UOC

Universitat Oberta de Catalunya

TRABAJO FINAL

GRADO

Diseño e implementación de un Dron Para la detección de incendios forestales

Alumno Miguel Ángel Iglesias Jiménez
Grado Ingeniería Telecomunicaciones, especialidad Sistemas de Telecomunicación

Profesor: Xavi Villajosana Guillen
Consultor: Pere Tuset Peiró

10 enero 2016 rev(27/01/16)

A MIS PADRES, HERMANOS Y FAMILIA, así como

a mi pareja POR EL APOYO DADO A LO LARGO DE ESTOS AÑOS

QUE HA HECHO POSIBLE LLEGAR HASTA AQUI.

Agradecer este proyecto:

A profesores de la UOC y en especial a Consultores por su ayuda, paciencia y respuestas siempre que las he necesitado, serian muchos para poder citarlos debido a lo largo de estos años.

A la especial figura de mi Tutor, Gracias *Joaquín* por esos consejos importantes en cada momento y que siempre me has dado.

A compañeros que igualmente serian muchos, el más reciente *Jordi* gracias por ejercer en apoyo para facilitarme el tener un punto de vista crítico siempre una inestimable ayuda para ver y localizar mejoras.

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Diseño e implementación Dron para la detección de incendios forestales*
Nombre del Autor:	*Miguel Ángel Iglesias Jiménez*
Nombre del consultor:	*Pere Tuset Peiró*
Fecha del libramiento (mm/aaaa):	*Enero/2016*
Área del trabajo Final:	*Sistemas Arduino*
Titulación:	*Grado Ingeniería Telecomunicaciones (Sistemas Telecomunicación)*

Abstrac

The serious damage caused by forest fires and the enormous size that reach them, make public administrations spend and increase economic resources in prevention on fire fighting as well as to remedy the negative consequences that these bring with them. Forest fires can cause irreparable damage to the environment as well as human losses and evictions of people who are reached by them.

An important aspect in fire prevention is its quick detection, since all the operating of extinction depends on the detection of the initial focus to act immediately and prevent it from becoming a large fire. It is therefore very necessary to detect the focus of a fire and determine the extension as soon as possible as well as to determine exactly the location of the same.

Different technologies have been used for these purposes, which include the use of cameras, optical and infrared satellite images as well as the use of the Integration Geographic (IGS) System. There are also other methods based on active laser (Salas, 2003), with very fast response times and ability to indicate the "actual location of the fire together with its intensity or virulence", whose main difference from the infrared is that this system is based on detecting smoke columns, getting faster in detention, while the infrared to detect heat.

There are basically two methods of detecting fires one based on the detection of the fire and another based on the detection of smoke in the use of cameras. The fire detection is simple but with the disadvantage that when an optical camera detects a fire, this is already at an advanced stage. On the other hand, smoke detection allows you to detect forest fires in the early stages while the analysis of the images is much more complex since the algorithms responsible for the analysis should include: background subtraction, filtering of quick elements, and detection by colour and analysis of related components. Thus from the images by analysing pixel to pixel you can see the smoke from controlled fires that corresponds to possible fire.

A system based on infrared and optical cameras, as it is the forest in Andalucía (SPAIN), (which entered into operation with the forest 2000 plan), consists of a series of infrared and optical cameras placed strategically throughout the territory to control, a central monitoring with monitoring and image processing unit monitors. Each checkpoint situated in different towers has devices of cameras that provide a view of 360 degrees, which sent the recorded images to the image-processing unit responsible for detecting a possible fire. When a possible fire is automatically detected the camera will stop at that position and the alarm is sent to the monitor's supervision.

From the initial study a system of Imaging for Assembly is proposed in mobile unit (Dron) for surveillance of areas which fall outside the scope of systems as described previously (by scheduled flight).

On the other hand the system provides some interesting new features since it integrates a number of sensors for the collection of meteorological data (atmospheric pressure, relative humidity, temperature...) that are important in subsequently determining a plan of attack against fire (such as using specific software SIDAEX).

Thus, with these objectives, and taking into account that we want the system to be mobile, various sensors are studied. We are looking for precision and low power consumption, which should be standard and if possible with minimum requirements of signal adaptation. They should have some of the communication systems that implement the node Coordinator integrated (Arduino).

The described platform will be equipped with various communication systems including radio control by RF (2.4 GHz) and the possibility of communication via GSM modem which makes it useful to use as a support unit for extinction once teams are located in the area. They can use it to display areas in the declared fire where people cannot access, while allowing contact control center to extinguishing equipment and lines of possibles it. This gives this project a special appeal since the unit may become a "patrol" fire service, while it can be used as a support unit by extinction equipments.

Key Word

Embedded systems, IGS (Systems Integration Geographic), forest fires, sensor network (WSN), systems wireless, Arduino and Dron.

Índice de Contenidos

Índice de Figuras

Índice de Figuras

1. Introducción

[Ir al índice]

1.1 Justificación y contexto

La necesidad de apoyo desde todos los ámbitos posibles a los medios existentes en la actualidad para evitar las catástrofes en el ámbito de los incendios forestales se hace cada vez más evidente. Los graves daños que producen en el medio ambiente así como el gran impacto social que llegan a alcanzar han hecho que las administraciones públicas dediquen importantes recursos económicos a la lucha y prevención contra el fuego y a paliar en la medida de lo posible las consecuencias que producen este tipo de incendios. Los incendios forestales no solo producen grandes daños en el medio ambiente en muchas ocasiones irreparable, sino que llegan incluso a producir el desalojo de zonas habitadas por la propagación del incendio y en ocasiones pérdidas de vidas humanas.

Para evitar estos daños los equipos de extinción de incendios cuentan con numerosos y modernos medios tanto terrestres como aéreos para la realización de las tares de lucha contra el fuego, por los que cada vez se hace más necesario el desarrollo de aplicaciones que utilizando las nuevas tecnologías optimicen el uso de dichos medios, ayuden en la toma de decisiones a los responsables de la lucha contra incendios, permitan evaluar las consecuencias del incendio sobre las áreas implicadas y su más adecuada restauración.

Para evitar que una alarma termine por ser un gran incendio forestal se pueden llevar a cabo tres acciones principales:

- Prevención: Esta consiste en evitar que se provoque el incendio y minimizar las consecuencias en caso de que se produzca. Son acciones como por ejemplo las quemas preventivas, o los cortafuegos.
- Detección: Consiste en alertar con la mayor celeridad posible de la existencia de un incendio con la finalidad de comenzar su extinción. Como puede ser la función realizada por los guardias forestales.
- Extinción: Es la acción directa contra el fuego que permite terminar con el incendio estas consisten en eliminar alguno de los tres elementos que lo constituyen el denominado triángulos del fuego: combustible (vegetación), comburente (oxigeno) encargado de oxidar el combustible favoreciendo la combustión y energía (calor) necesaria para que se produzca la reacción.

De otro lado en el sentido de detección de incendios la Consejería de Medio Ambiente a través del Plan INFOCA**[1]**, han llevado a cabo instalaciones y disponen de medios que han permitido desarrollar numerosas aplicaciones basadas en las nuevas tecnologías, a través de convenios con universidades y centros de investigación con la colaboración de empresas especializadas así como las aportaciones del Ministerio de Medio Ambiente.

Entre dichas aplicaciones se pueden citar:

- La detección automática de incendios forestales mediante el uso de cámaras infrarrojas situadas estratégicamente en puntos fijos. En este sentido existe el denominado “Sistema Bosque”[1] instalado en Andalucía y Galicia. También existe el denominado “Programa Fuego” en fase de experimentación el cual las cámaras son situadas en mini satélites.

- Técnicas de medición durante el incendio, de áreas quemadas por el fuego, por medio del “Sistema SALEIF”[2], el cual está basado en el vuelo de un helicóptero provisto de un ordenador con GPS que recorre el contorno del incendio.
- Técnicas de simulación de la evolución de un incendio con el “Programa CARDIN”[2] basado en utilización de SIG (Sistemas de Información Geográfica)

La aplicación de procesado de imágenes para llevar a cabo tareas de vigilancia es un campo ampliamente estudiado, por lo que deben de aplicarse esos conocimiento y la utilización de estas técnicas de procesado de imágenes en el ámbito de la detección de incendios. Es en este contexto que el presente proyecto pretende centrarse en la tarea de detección de incendios forestales mediante técnicas de vigilancia y procesado de imagen.

1.2 Descripción del Proyecto

Dadas las grandes extensiones de terreno que son propensas a incendios, bien sea por su vegetación o bien por los factores climáticos de la zona, hacen que sea muy difícil cuando no imposible que las acciones de detección mediante las funciones de vigilancia puedan ser llevadas a cabo mediante personas. De igual forma el despliegue de plataformas y redes de cámaras llegan a resultar proyectos altamente costosos y finalmente acaban por ser abandonados o no dotados de los recursos suficientes para llevar a cabo sus cometidos como ha ocurrido con el mencionado “Sistema Bosque” en Andalucía.

Esto lleva a proponer un UAV (*Unmanned Aerial Vehicle,* vehículo de vuelo no tripulado), también comúnmente denominado *Dron*, el cual permitirá realizar las funciones de vigilancia cubriendo mayores extensiones de terreno. Dicha unidad móvil estará dotada con un sistema de Geo-posicionamiento por GPS (*Global Positioning System*), capacidad de realizar grabaciones de video, captura y transmisión de imágenes, que pondrá a disposición del centro de control a través de sus diversos sistemas de comunicaciones. De igual forma podrá ser comandada en varias modalidades como son:

- Modalidad de vuelo programado: El sistema es programado para realizar planes de vuelo periódicos con unos recorridos preestablecidos, durante dichos vuelos tendrá la capacidad de realizar capturas de imágenes y/o video lo que permitirá poder conocer mediante visión el estado de la zona.
- Modalidad de vuelo por control manual (corto alcance): Situado en una zona de incendio podrá ser puesto a disposición de personal del equipo de extinción para ser pilotado manualmente y prestar funciones de apoyo que pudiera necesitar dicho equipo como puede ser: Visualización del incendio en zonas no accesibles por personas, toma de temperaturas, presión y humedad de esas zonas, extensión del incendio etc…

De esta forma mediante técnicas de procesado de imagen la unidad podrá detectar situaciones de posibles incendios de forma automática alertando al centro de control al que enviara los muestreos de datos atmosféricos obtenidos que podrán ser recabados y analizados para ayuda en la determinación de un plan de actuación contra incendio.

1.3 Objetivos principales

El objetivo principal del presente proyecto es desarrollar una plataforma que permita dar un paso más en ayudas a la prevención de incendios forestales. Plantea las bases para llevar a cabo un sistema de detección de incendios mediante un UAV el cual realizara la captura de imágenes para posterior detección automática de incendios mediante técnicas de procesado de imagen. Dicha unidad incluirá un sistema de sensores para medidas de fenómenos atmosféricos con indicador de localización vía GPS con la finalidad de ayudar a la determinación de un plan contra incendio y/o labores de prevención de incendios. Su desarrollo es llevado a cabo mediante sistemas embebidos, en el que podemos concretar cuatro bloques principales bloques principales:

- **El Sistema Central**: Lleva a cabo las funciones de coordinación de todos los subsistemas de la unidad Dron, permitiendo la interacción con el usuario a través de sus diversos interfaces y medios de comunicación disponibles. Ya sea en largo alcance con los centros de control o bien dando paso al enlace de corto alcance (aprox. 1Km) como puede ser en el caso de uso de la unidad Dron por equipos de extinción de incendios para funciones de apoyo.

- **Sistema Controlador de vuelo**: Realiza las funciones de piloto de a bordo de la UAV (*Unmanned Aerial Vehicle*). Este podrá ser programado de forma remota a través del sistema central para realizar vuelos periódicos de duración de 15 minutos. Aunque pueda parecer un tiempo corto dada la velocidad que puede alcanzar permitirá cubrir grandes extensiones de terreno. En modo de operación manual el sistema de control de vuelo permitirá ser controlado de haciendo uso de un enlace de radio-control pudiendo ser pilotado el Dron de forma manual (p.e. por algún miembro del equipo de extinción).

- **Sistema Captación video y Sensores atmosféricos**: El sistema será capaz de capturar imágenes en tiempo real así como informar de coordenadas de posicionamiento y fenómenos atmosféricos concretamente temperatura, presión atmosférica y luminosidad. Sobre dichas capturas se implementara una monitorización de la geo-localización por GPS conociendo así con exactitud las coordenadas donde se producen el posible inicio del incendio. Estos muestreos serán efectuados de forma periódica y los resultados obtenidos serán puestos a disposición del centro de control donde podrán ser analizados para la detección automática de incendios y disponiendo asi de los datos que le sean de utilidad y ayuda para determinar un plan de actuación.

- **Sistema de estación base**: Esta suministrara la energía eléctrica necesaria para la recarga de las baterías de la unidad *Dron* realizándose mediante placas solares lo que hace que puedan instalarse diversas bases e incluso en lugares remotos, de difícil acceso, donde la unidad pueda cargar baterías.

Del alcance de este proyecto queda excluido el montaje de las estaciones base donde la unidad *Dron* efectuara su aterrizaje para recarga de baterías y quedando a la espera de órdenes de vuelo. Si bien se aportara diseño y requisitos de dichas bases (circuitos para carga y comunicaciones con centro de control).

1.4. Enfoque y metodología

El presente proyecto está desarrollado con un enfoque didáctico y se lleva a cabo en diferentes ciclos los cuales, se realizan de forma secuencial si bien en todo momento se ha tenido una visión global lo que permite poder realizar trabajos de forma transversal cuyos resultados han sido utilizados en ciclos posteriores facilitando la solución requerida.

Así en una primera *fase de búsqueda e investigación,* se realiza el estudio de la tecnología y herramientas requeridas tanto en el ámbito del hardware (micro-controladores, sensores, componentes electrónicos y herramientas) como en software (sistemas operativos, lenguajes y entornos de programación). Por ultimo esta fase permite la identificación de los datos a tratar y la determinación de las metodologías utilizadas para la obtención del objetivo final.

Las fases *de montaje, de codificación y de testeo y verificación* son llevadas a cabo de una forma transversal, mediante pruebas empíricas para ir obteniendo los resultados deseados en cada uno de los dispositivos y sistemas instalados.

Por último la fase *de documentación* es llevada a cabo durante toda la duración del proyecto, desde su inicio recabando documentación e información y efectuando documentos tales como diagramas de bloques, diagramas de flujos, esquemas eléctricos, capturas de señales y pruebas efectuadas, que se aportan como anexos a esta memoria.

Se ha tenido presente la utilización de código abierto, así el proyecto se presenta en *Open Source* y de igual forma se beneficia de librerías de código de otros autores el cual ha sido convenientemente modificado conseguir los objetivos.

1.5. Planificación

1.5.1 Planteamiento inicial y seguimiento

El proyecto queda divido en tres fases bien definidas: Fase de estudio, fase de implementación y desarrollo y fase de pruebas y conclusiones. Estas fases en su mayoría son secuenciales aunque manteniendo una visión global del proyecto en ocasión se llevan a cabo de forma transversal interactuando entre ellas.

Fase de estudio.

En esta fase se realiza el estudio de la tecnología necesaria que se utilizara así como de las herramientas de las que haremos uso. Se identifican los dispositivos necesarios para la implementación efectuando la selección de estos para lo que se tiene en cuenta diversas consideraciones como calidad y precisión componentes sensores para que cumplan con el mínimo requerido en las condiciones del proyecto, estandarización de los mismos, y coste. De igual forma se determina el sistema operativo a utilizar, lenguajes de programación y aplicaciones, quedando definida la metodología utilizada presentando todo el itinerario a seguir en una planificación inicial para la obtención del objetivo final.

Fase de implementación y desarrollo.

Una vez determinada la tecnología a utilizar y seleccionados los componentes que se considerados como adecuados para el cumplimiento de requisitos se inicia la fase de montaje de un prototipo, que si bien se realiza el estudio y un primer esquemático de todo el sistema completo, el montaje no es efectuado en su totalidad en una única vez, sino que aplicamos metodología modular compaginándola con la fase de codificación del sistema y en su inicio con la parte final de la fase de estudio.

Así se inicia el montaje con el diseño y montaje de la estructura, motores y reguladores de potencia propulsión para proceder a su calibrado y ajuste. A continuación se anexa el modulo sistema control de vuelo, así como los módulos receptores de radiofrecuencia (RF 2.4GHz) para el control de la UAV mediante transmisor de radiofrecuencia. Para verificar su correcto funcionamiento se aprovecha la fase de estudio de la aplicación de control de piloto automático la cual nos permite verificar correctamente el funcionamiento de los módulos de comunicación de control, módulo de posicionamiento GPS, Modulo GSM-GPRS (*Global Sistem for Mobile communications - General Packet Radio Service*) y el bus de comunicaciones I2C a utilizar para sensores periféricos.

El siguiente paso será efectuar la correspondiente programación del módulo de comunicaciones de video y su depuración de forma independiente por considerar que este es el punto que puede resultar más vulnerable e importante en el proyecto.

Una vez obtenidos los resultados óptimos y realizadas todas las pruebas de comunicaciones que permitan disponer de un código de comunicaciones estable con un óptimo control de errores, se realizara el ensamblaje de los distintos sensores y módulos de transmisión de video con el sistema central quedando finalmente todos los sistemas programados e instalados sobre la estructura.

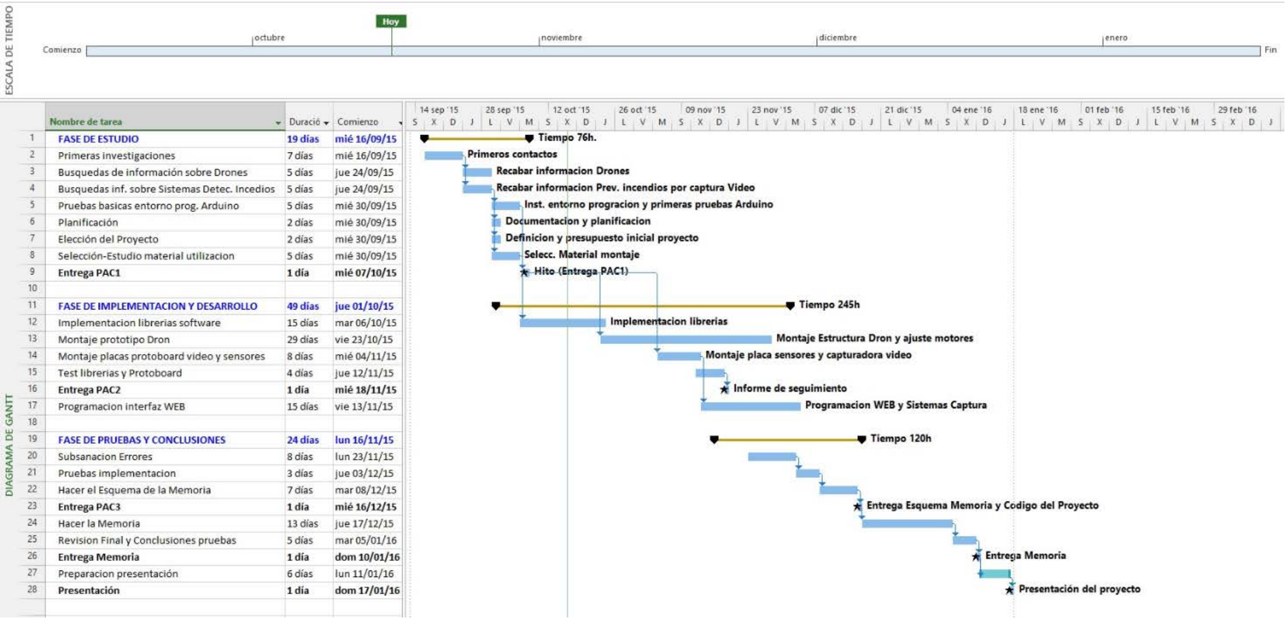
Fase de pruebas y conclusiones.

En esta fase se llevan a cabo los test de pruebas para verificación de todas las funciones descritas para este proyecto, se incluye en ella una parte de subsanación de posibles errores o desviaciones de los objetivos que puedan darse, asegurando así que las pruebas finales sean correctas.

Se termina con la verificación de que se cumplen todos los requisitos marcados y llevando a cabo una toma de datos para dejar constancia de ello y posterior aportación a la Memoria del proyecto.

La planificación detallada puede verse en el siguiente diagrama de Gantt

1.5.2. Diagrama de Gantt


ESCALA DE TIEMPO
Hoy
Comienzo
octubre
noviembre
diciembre
enero
Fin
DIAGRAMA DE GANTT
14 sep '15
28 sep '15
12 oct '15
26 oct '15
09 nov '15
23 nov '15
07 dic '15
21 dic '15
04 ene '16
18 ene '16
01 feb '16
15 feb '16
29 feb '16
S X D J L V M
Tiempo 76h.
Primeros contactos
Recabar informacion Drones
Recabar informacion Prev. incendios por captura Video
Inst. entorno progracion y primeras pruebas Arduino
Documentacion y planificacion
Definicion y presupuesto inicial proyecto
Selecc. Material montaje
Hito (Entrega PAC1)
Tiempo 245h
Implementacion librerias
Montaje Estructura Dron y ajuste motores
Montaje placa sensores y capturadora video
Informe de seguimiento
Programacion WEB y Sistemas Captura
Tiempo 120h
Entrega Esquema Memoria y Codigo del Proyecto
Entrega Memoria
Presentación del proyecto

	Nombre de tarea	Duració	Comienzo
1	FASE DE ESTUDIO	19 días	mié 16/09/15
2	Primeras investigaciones	7 días	mié 16/09/15
3	Busquedas de información sobre Drones	5 días	jue 24/09/15
4	Busquedas inf. sobre Sistemas Detec. Incedios	5 días	jue 24/09/15
5	Pruebas basicas entorno prog. Arduino	5 días	mié 30/09/15
6	Planificación	2 días	mié 30/09/15
7	Elección del Proyecto	2 días	mié 30/09/15
8	Selección-Estudio material utilizacion	5 días	mié 30/09/15
9	**Entrega PAC1**	**1 día**	**mié 07/10/15**
10			
11	FASE DE IMPLEMENTACION Y DESARROLLO	49 días	jue 01/10/15
12	Implementacion librerias software	15 días	mar 06/10/15
13	Montaje prototipo Dron	29 días	vie 23/10/15
14	Montaje placas protoboard video y sensores	8 días	mié 04/11/15
15	Test librerias y Protoboard	4 días	jue 12/11/15
16	**Entrega PAC2**	**1 día**	**mié 18/11/15**
17	Programacion interfaz WEB	15 días	vie 13/11/15
18			
19	FASE DE PRUEBAS Y CONCLUSIONES	24 días	lun 16/11/15
20	Subsanacion Errores	8 días	lun 23/11/15
21	Pruebas implementacion	3 días	jue 03/12/15
22	Hacer el Esquema de la Memoria	7 días	mar 08/12/15
23	**Entrega PAC3**	**1 día**	**mié 16/12/15**
24	Hacer la Memoria	13 días	jue 17/12/15
25	Revision Final y Conclusiones pruebas	5 días	mar 05/01/16
26	**Entrega Memoria**	**1 día**	**dom 10/01/16**
27	Preparacion presentación	6 días	lun 11/01/16
28	**Presentación**	**1 día**	**dom 17/01/16**

Fig. 1 Diagrama de Gantt

Esta planificación puede decirse que dada la envergadura del proyecto ha sufrido algunos retrasos en hitos de entrega y mantener documentación al día si bien en su entrega final el retraso ha sido de 1 días. Estos retrasos han sido mayormente por la carga de trabajo que supone el montaje de la unidad en su totalidad, pruebas modulares para verificación del correcto funcionamiento.

Las dificultades surgidas con proveedores los cuales indicaban plazos que finalmente no se han cumplido e incluso ha llevado a tener que realizar modificaciones en aspectos importantes del proyecto como ha sido en el sistema estabilizador de cámara de video.

1.5.3. Metodología de la memoria

La realización de la memoria del proyecto se realiza mediante la fase de documentación la cual es llevada a cabo a lo largo de todo el proyecto. Así en la fase de estudio se recopila documentación, *datasheet* de componentes, referencias de bibliografía. De igual forma en la fase de desarrollo se capturan tomas de señales, imágenes correspondientes al montaje de prototipo y capturas de pantalla de las pruebas de verificación efectuadas así como videos explicativos de cada uno de los subsistemas y montajes llevados a cabo. En la fase de codificación se implementa la documentación correspondiente a la programación efectuada sobre el sistema que se aporta como anexo a la presente memoria.

1.5.3. Metodología de aplicación

La implementación de la aplicación es realizada en cuatro fases que definimos como fase de análisis, fase de programación, fase de codificación, fase de depuración y control de errores, las cuales se describen detalladamente a lo largo de la presente memoria, el capítulo de Software.

a) Fase de análisis

En esta fase se efectúa el planteamiento del problema de una forma técnica, se determinan los sistemas y software a utilizar obteniendo tras el análisis un primer diagrama que aporta la información general del sistema y los dispositivos hardware del sistema que se utilizaran y un diagrama de flujo general de la aplicación. Se pueden diferenciar dos aplicativos uno para la unidad *Dron*, en adelante Sistema Central, y otro WEB para usuario..

Aplicación del Sistema Central

En el sistema se requiere que el mismo en su inicialización efectúe un auto chequeo y detección de los recursos disponibles (interfaces con usuario, interfaces de comunicación, sensores del sistema), enviando un comunicado al usuario si existe incidencia. Verificara y establecerá ser servidor proxy para la conexión con la unidad de Control de vuelo de abordo activando la comunicación con la misma para ponerla a disposición del usuario. Si la inicialización es correcta quedara en estado de reposo en espera de planes de vuelo o ser controlado mediante radio-control. Si el usuario quiere podrá activar muestreos periódicos de datos de los distintos sensores para control de la zona donde esté ubicado. Adquiridos estos efectúa el envío a la base de datos para su archivo y posterior tratamiento por parte del usuario si lo requiere. Cuando el sistema no tenga trabajo pasara a estado de bajo consumo dado que se encuentra alimentada por baterías o paneles solares en la base. El sistema central será accesible en todo momento mediante conexión SSH, al igual que mediante software específico de aplicación de realización de planes de vuelo. Igualmente será accesible en todo momento vía GSM disponiendo de servicio de voz (Nº telefónico Movil).

Aplicación WEB

Esta aplicación Web monitoriza los datos obtenidos por el sistema central y los presenta al usuario en un formato comprensible. De otro lado realiza las funciones de interfaz de comunicación *usuario-sistema* cuando se requiera configurar parámetros del sistema como puede ser activación de servicios NAS como puede ser servidores media, o servidores Cloud que la unidad pudiera utilizar, o también la configuración de puntos de acceso de red inalámbricos (esta es opcional implementada en parte para mejoras).

b) Fase de programación

En esta fase se instalan los sistemas y entornos de programación a utilizar, se determinan finalmente las estructuras y se realiza un diagrama de flujo detallado de las aplicaciones que será la base para la fase de codificación de las mismas. En este proyecto se utilizara:

-Sistema operativo *Raspbian* para *Raspberry Pi*
-Protocolo MAVLink para control y programación de planes de vuelo.
-Lenguaje Python para programación
-Openc Cv para tratamiento de imágenes.
-Lenguaje html y php para programación de aplicación WEB
-Lenguaje MySql para tratamiento de base de datos

1.6 Breve descripción de los capítulos de la memoria

La presente memoria se estructura de la siguiente forma. En primer lugar, el Capítulo 2 sirve al lector para dar a conocer la trayectoria que ha tenido la tecnología que se pretende implementar en la solución a lo largo de los últimos años y contextualizarla en la problemática que se aborda, exponiendo también la situación actual y posibles ventajas e inconvenientes que esta tecnología puede presentar.

El capítulo tres sirve al lector para familiarizarse con los conceptos teóricos que se mencionan en este trabajo, al igual que para mostrar un estudio de la plataforma a utilizar y del problema que se pretende abordar poniendo un especial énfasis en la problemática existente. Así en un primer apartado se hace una introducción al modelo de la dinámica del cuatri-motor desde el punto de vista del control de sistema, para pasar a detallar todos los componentes en su configuración básica. Pasando a describir la funcionalidad de los distintos sistemas que puede finalmente ser de utilidad en el ámbito del problema al que se pretende dar solución, describiendo detalladamente la interacción entre los distintos sub-sistemas que componen la posible solución así como las opciones que puede ofrecer la aplicación de control y monitorización para la interacción con el usuario y llevar a cabo el control de la UAV.

El Cuarto capítulo entra a describir los requisitos necesarios que deberá reunir el hardware para la solución pasando al análisis de las distintas opciones y la selección de cada uno de los módulos o componentes que conformaran finalmente la unidad *Dron* justificando y verificando en los casos que se requiere que las selección cumpla los requisitos necesarios para su selección.

El Quinto capítulo entra en la selección y descripción del software, sistema operativo que se utilizara, dado que muchas de las necesidades quedan cubiertas por el propio sistema operativo instalado y son simplemente instalaciones y uso de los módulos correspondientes del sistema o aplicaciones que solo requieren instalación y aprendizaje de manejo, estos se han llevado a un anexo como Manual de Instalación y Uso de aplicaciones Por lo que se entra directamente a ver algunos de los ejemplos de programación del uso de las interfaces como puede ser en lenguaje Phyton o la exposición de la conexión con protocolos de control de vuelo , para pasar finalmente a exponer las técnicas de procesado de video con Open CV que pueden ser de utilidad para la detección de incendios.

El capítulo 6 realiza un estudio de la viabilidad técnica exponiendo los principales inconvenientes observados así como sus puntos fuertes. El capítulo 7 presenta el presupuesto para finalmente en el capítulo 8 presentar las conclusiones y futuros trabajos.

2. Antecedentes

[Ir al índice]

2.1 Estado del Arte

Los UAV (*Unmanned Aerial Vehicle*) o vehículos aéreos no tripulados tienen su origen en fines militares dado que con ellos se puede tanto vigilar una zona de conflicto como atacar sin necesidad de poner en peligro vidas humanas. El UAV puede estar controlado remotamente por los operadores en tierra o pueden ser autónomos siguiendo una trayectoria previamente definida en su sistema de vuelo. Se pueden definir por tanto dos estaciones que pueden manejar la información del *Dron*, la estación de tierra y la estación de a bordo de la aeronave.

Existen diversos tipos de UAV los hay de tipo avión, tipo helicóptero, tipo dirigible etc… en las siguientes figuras podemos ver algunos de los utilizados para fines militares:

Figura 2 UAV de combate Barracuda (Fuente:Wikipedia Enlace)

Figura 3 UAV Predator (Fuente Wikipedia Enlace)

Sin embargo estas aeronaves también tienen uso en el ámbito civil, actualmente se trabaja en este sentido para desarrollar vehículos cada vez más pequeños y ligeros, con menor nivel tecnológico que empiezan a utilizarse en aplicaciones civiles las cuales pueden resultar peligrosas para que sean llevadas a cabo por tripulaciones, como es el caso de los incendios forestales.

En este ámbito en España existen diferentes proyectos de vehículos no tripulados orientados a la lucha contra incendios. Así la empresa FlightechSystem[3] ha llevado a cabo el desarrollo de diversas UAV para ayuda en problemas de deforestación causada por los incendios y para labores de vigilancia e incluso en detección de personas pirómanas, siendo capaz la aeronave de identificar actividad humana tanto de día como de noche a casi 2.000 metros de distancia.

El avance y desarrollo de la investigación en el área de los UAV, desde los años 90, en una gran medida es debido a la miniaturización de la tecnología, que ha permitido llevar a cabo diversos modelos y estudios de los distintos tipos de naves y su correcto control. Así desde los años 2000 se vienen aplicando nuevas estrategias de control que junto con la posibilidad de construcción e implementación de hardware y software más potente, con micro controladores capaces ejecutar millones de instrucciones por segundo, y la reducción del tamaño y por tanto peso de las baterías, principalmente ion-litio y Litio-Polímero han permitido una escalada en el ámbito de los *Drones*.

Las publicaciones recientes centran estas investigaciones mayormente en:

- Creación y validación de modelos en distintos tipos de software como puede ser MATLAB, simulink u otros de tipo CAD (*Computer Aided Desing*, Diseño Asistido por Computadora).
- Implementación de diferentes estrategias de creación y seguimiento de trayectorias incluyendo detección y evasión de obtaculos.
- Control basado en sensores de visión (Camaras).

Todo ello ha llevado los últimos años a conseguir desarrollar vehículos cada vez más pequeños y ligeros para su uso en el ámbito civil (denominados de baja escala), los cuales son utilizados para control de fronteras, operaciones contra narcotráfico, control de tráfico de carreteras, o en operaciones de emergencia de la cruz roja como es el caso del proyecto que la cruz roja lleva a cabo en cooperación con la empresa Zerintia[4] la cual dispone de una flota de *Drones* que pone a disposición de la Cruz Roja siempre que los necesite, como puede ser en casos de búsqueda de supervivientes o localización de equipos de rescate en grandes catástrofes.

2.2 Estudio de Mercado

Se ha podido ver los vehículos aéreos no tripulados UAV se encuentran en expansión, pero igualmente existe un estricto control y muchas limitaciones para su utilización con las actuales normativas, por tanto no es fácil su expansión y comercialización. Aun así la agencia *WinterGreen Research* [5], famosa por sus estudios de mercado sobre sectores como puede ser el de la energía, software o telecomunicaciones, ha hecho público un estudio sobre la previsión de mercado de los *Drones* comerciales para los años 2015 a 2021. En este indica que la siguiente generación de UAV conseguirá un reemplazo total de los sistemas aéreos existentes en la actualidad en diversos sectores como pueden ser: Vigilancia de instalaciones, patrullas fronterizas, envió de paquetería, fotografía y agricultura, apoyo en situaciones de catástrofes.

Estas unidades presenta claras ventajas como son: Mayor eficiencia energética, mayor vida útil así como un menor coste de operación que los actuales sistemas tripulados.

El mercado de las UAV (*Unmanned Aerial Vehicle*) presenta una fuerte tendencia a crecer principalmente por razones como son el abaratamiento de los sistemas de navegación y visualización, la experiencia de más de 3 millones de horas de vuelo que sistemas militares han llevado a cabo dotan a esta tecnología de la suficiente credibilidad y confianza para sentar las bases en su uso comercial.

En este sentido la citada agencia presenta unos datos de previsión de mercado para las UAV el cual **alcanzo los 609 millones de dólares en el año 2014 y se espera alcance la cifra de 4.800 millones para el año 2021**, debiéndose este crecimiento principalmente a la venta en sectores como son: Mapeo para petróleo y gas, inspección de infraestructuras energéticas, envió de paquetería, aplicaciones agrícolas y servicios de apoyo a catástrofes. Puede solicitarse una copia del citado informe en el siguiente enlace

Hay que indicar que al mencionado avance de las tecnologías y reducción del tamaño unido a un menor coste ha llevado a que estos dispositivos estén al alcance de un mayor público objetivo con menor poder adquisitivo. Así actualmente se están comercializando un tipo muy particular de UAV multi-rotor en el que los motores están en el mismo plano, y pueden encontrarse disponibles en distintos modelos según su número de motores sea de cuatro, seis y ocho motores.

Estos *Drones* de bajo coste se establecen para usos recreativos como pueden ser los modelos Parrot *AR Drone 2.0* o *DJI Phanton* Profesional los cuales están orientados al ocio y a la grabación de video mediante el uso de cámaras.

Hay que indicar que en España está **totalmente prohibido el uso de *Drones* para actividades civiles** así lo ha puesto de manifiesto AESA (Agencia Estatal de Seguridad Aérea) en diversas ocasiones mediante comunicados, donde indica "*…en España* ***no está permitido, y nunca lo ha estado, el uso de aeronaves pilotadas por control remoto con fines comerciales o profesionales****, para realizar actividades consideradas trabajos aéreos, como la fotogrametría, agricultura inteligente (detectar en una finca aquellas plantas específicas que necesitarían de una intervención, como riego, fumigación, para optimizar el cultivo), reportajes gráficos de todo tipo, inspección de líneas de alta tensión, ferroviarias, vigilancia de fronteras, detección de incendios forestales, reconocimiento de los lugares afectados por catástrofes naturales para dirigir las ayudas adecuadamente, etc.*
La realización de trabajos especializados (también llamados trabajos aéreos), como son las filmaciones aéreas, los de vigilancia, de detección y / o extinción de incendios, de cartografía, de inspección, etc., tal como indican los artículos 150 y 151 de la Ley 48/1960 sobre Navegación Aérea, requiere autorización por parte de AESA, y hasta que no esté aprobada la nueva normativa específica que regule el uso de este tipo de aparatos, AESA no puede emitir dichas autorizaciones porque carece de base legal para ello"[6]

Puede consultarse al respecto la citada legislación sobre Navegación Aérea en vigente en documento indicado en referencias [7].

Por tanto **la posibilidades introducir un diseño de una UAV, como puede ser el presente proyecto para fines comerciales y/o profesionales no es posible en España está prohibido**, la única comercialización posible de UAV en España se encontraría en el ámbito de uso recreativo y actividades deportivas y realmente está muy limitado por la mencionada legislación.

3. Descripción teórica y funcional

[Ir al índice]

3.1 Modelo UAV Quad-rotor

El presente capitulo se centrara en describir las partes que conforma un modelo de AUV, el *quad-rotor*, ya que es el modelo que se ha seleccionado para el presente proyecto por presentar una gran versatilidad y maniobrabilidad y ser actualmente la configuración más utilizada debido a que posee la misma naturaleza no lineal del helicóptero clásico y los mismos efectos aerodinámicos, pero con una mayor facilidad de control por tener los torques aerodinámicos bien definidos. La estructura *quad-rotor* se corresponde con la de un multi-cóptero de 4 motores y 4 hélices dispuestos en los extremos de unos brazos con la misma longitud. Las hélices delantera y trasera giran en un sentido mientras que las hélices derechas e izquierdas giran en el sentido opuesto. Esto lleva a una cancelación de los efectos giróscopos y torques aerodinámicos en estado de vuelo compensado.

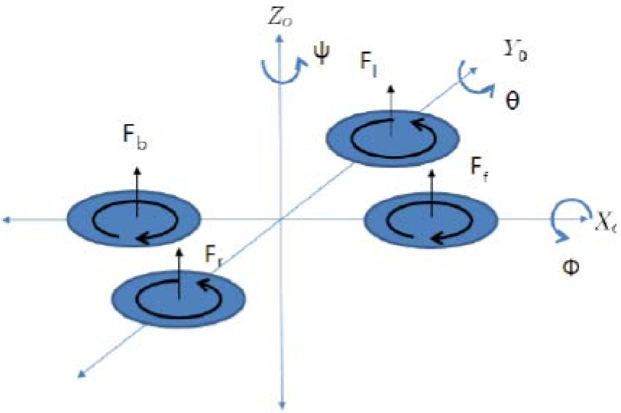


Figura 4 Esquema ejes rotación y fuerzas rotores estructura quad-rotor

A groso modo la estrategia de control del *quad-rotor* responderá ante serie de entradas (acciones de control) con una serie de salidas que corresponden a las coordenadas (x,y,z) y ángulos de orientación (θ,Ψ,ϕ), dicho sistema está sometido a perturbaciones externas v(t), por lo que un controlador deberá calcular las acciones a tomar para contrarrestar dichas perturbaciones puede representarse dicha estrategia de control mediante un esquema de control de lazo cerrado como se muestra en la siguiente figura

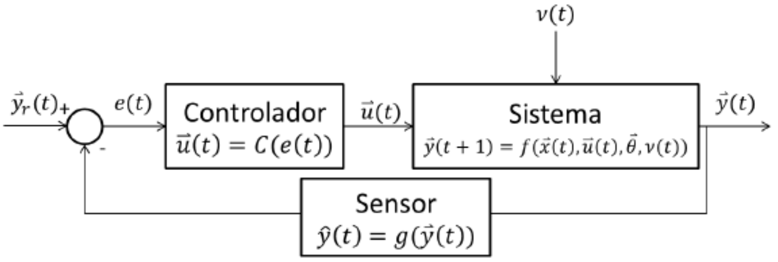


Figura 5 Esquema de control de lazo cerrado

Por tanto el cálculo de la acción de control dependerá de las medidas tomadas por los sensores, por lo que los rangos de tolerancias de dichos sensores harán que un sistema de control sea o no aceptable para una determinada aplicación.

3.2 Componentes UAV modelo Quad-Rotor

La configuración estándar para llevar a cabo la construcción y ensamblado de un *Dron* de cuatro motores implica los siguientes componentes.

3.2.1 Estructura

Conforma el cuerpo de la UAV, sirviendo de esqueleto para la instalación de componentes mecánicos, eléctricos y electrónicos de la aeronave. Esta debe de minimizar su peso y maximizar en lo posible su rigidez y resistencia con la finalidad de soportar y transmitir las fuerzas que actúan sobre la UAV. Es común que se usen materiales como fibra de vidrio, aluminio, o polímeros por su alta resistencia y bajo peso

3.2.2 Motores y Hélices

El motor más utilizado en las UAV de baja escala es de tipo eléctricos *brushless* DC, ya que estos no contemplan anillos de rozamiento para el cambio de polaridad en el rotor, realizando dicho cambio mediante la conmutación de la polaridad en los electroimanes incorporados en el motor. Son motores que presentan una reacción rápida y son alimentados mediante corriente continua (DC) la cual es convertida en una señal AC de tres fases que se aplica a los electroimanes y resultan ser tener una larga duración de uso[6].

Respecto de las hélices se fabrican de materiales compuestos como polímeros debido a las propiedades que le permiten resistir a fuerzas muy altas en ciertas direcciones. En el modelo Quad-rotor resulta muy útil ya que se conocen de antemano las fuerzas a las que estarán sometidas.

3.2.3 Controlador electrónico de velocidad ESC (*Electronic Speed Control*)

Estos dispositivos son los encargados de transformar las señales PWM (*Pulse Width Modulation*) en una señal de corriente alterna (AC) trifásica que es la que alimenta los motores. Cumplen también la función de amplificadores de voltaje salida a los valores nominales requeridos por el motor [7].

Para llevar a cabo la correcta regulación de velocidad estos monitorean la posición del imán permanente ubicado en el motor, efectuando los cambios de polaridad en los electroimanes del estator.

En la siguiente imagen se puede observar el esquema de un motor *Bushless*, donde el imán permanente se ubica en el rotor mientras que los electroimanes están fijados en el estator,

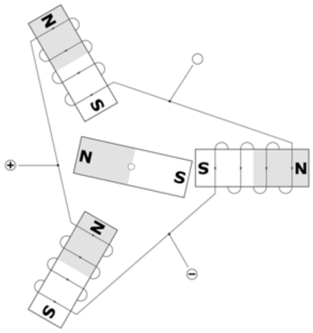


Figura 6 Esquema motor Brushless (Fuente:[7])

3.2.4 Controlador UAV

Realiza las funciones de control de la posición, velocidad y aceleración de la UAV, para ello cuentan con un microprocesador y unidades de medición inercial (IMU) y de forma general están dotados de puertos de comunicaciones con los diferentes protocolos como pueden ser I2C, UART, SPI, USB. Para el manejo de los motores hacen uso de las salidas PWM (*pulse width modulation*) de las que disponen. Actualmente existen una gran diversidad de controladores y diferentes comunidades que han colaborado para diseños de módulos *open-source*, si bien por lo general las empresas comerciales de *Drones* realizan sus propios diseños de estos módulos de control.

3.2.5 Sensores IMU (Unidades Medición Inercial)

Como se ha indicado en el apartado 3.1 el control de lazo cerrado requiere de un muestreo de lecturas actualizado de las coordenadas (sensores) para poder efectuar los cálculos de corrección. Los sensores más básicos que se utilizan en sistemas quad-rotor son los reunidos en las Unidades de Medición Inercial (IMU) que incluyen giroscopios y acelerómetro de tres ejes, lo que le permite obtener de una manera rápida mediante integración, la orientación de la aeronave. Estas unidades IMU suelen cumplimentarse con altímetros, sensores barométricos los cuales calculan la altitud en base a las medidas de presión, estos son costosos pero suelen presentar importantes desviaciones de precisión. Para aplicaciones que se requiera una mayor precisión se utilizan tecnologías sonar o laser entre otros. Para obtener la posición suelen hacer uso del magnetómetro (compás) el cual facilita la orientación respecto al sistema de referencia acoplado a la tierra, esto en conjunción con el uso de GPS permite obtener, en tiempo real, la posición hasta en 3 ejes según el número de satélites disponibles en cada momento.

3.2.6 Radiocontrol

Con la finalidad de comunicar y controlar la aeronave se requiere un sistema de comunicación por radio frecuencia (Transmisor-Receptor) el cual deberá disponer de un mínimo de canales igual al número de motores de la aeronave. Existen una gran variedad de módulos de comunicaciones y con distintos tipos de modulación. En el transmisor se deberá disponer de una serie de mandos usualmente palancas o *Sticks* de dos ejes que permitan el control de la aceleración (*throttle*) y la rotación del angulo Ψ (yaw) por un lado y el de los ángulos (roll) y (pitch). Actualmente existen dispositivos con mayor número de canales para control de otras opciones como modos de vuelo, tren de aterrizaje plegable, etc.. Siendo comunes los sistemas de 8, 16 y 32 canales.

3.2.7 Telemetría

Los sistemas de telemetría a bordo de la UAV como indica su propia definición permiten obtener datos físicos a bordo de la aeronave en tiempo real y enviarlos a la estación de control en tierra. Estos operan según la regulación local en el las frecuencias de 433Mhz y 915 MHz, realizando la comunicación con el controlador de vuelo de abordo a través de puerto de comunicación serial UART (*Universal Asynchronous Receiver-Transmiter*).

3.2.8 Batería

Estas unidades suelen utilizar para su autonomía de funcionamiento baterías de Polímero de litio, (Li-Po) ya que ofrecen una mayor densidad de carga a las tradicionales de ion-litio, un tamaño más reducido y menor peso.

3.3 Detección del fuego

Como se ha indicado en los primeros capítulos existen sistemas para la detección de incendios mediante cámaras situadas en el terreno y que hace uso de técnicas de procesado llevarla a cabo. Dicha detección se puede realizar mediante la detección del fuego o bien mediante la detección del humo ya que ambos son indicadores de los incendios forestales.

Sin embargo son características distintas ya que fuego se mantiene siempre en un mismo sitio pero este es muy variante por el avance del fuego y sus colores tienen la característica de ser muy vivos, mientras que el humo tiende a moverse y expandirse hacia la atmosfera y sus colores son muy apagados, esto hace que la detección de uno u otro sean procesos diferentes.

La detección del fuego se puede realizar de diversas formes en función de la cámara utilizada así:

- Mediante Cámara IR (Infrarroja): Los infrarrojos son una banda del espectro radioeléctrico por debajo de la luz visible sobre la que radian los objetos al producir calor.
- Mediante cámara de espectro visible: En estas el fuego presenta un color característico y diferenciado de otros elementos que puede facilitar su detección.

En este sentido la cámara infrarrojo presenta un nivel más altos de falsos positivos ya que la vegetación tiende a alcanzar mayores temperaturas los efectos solares acumulando calor y radiando un fuerte espectro infrarrojo, igualmente se producen falsos negativos por que la atmosfera atenúa la banda infrarroja en grabaciones a distancia.

3.3.1 Detección del fuego mediante su color

El modelo más usual para el almacenamiento de imágenes es el modelo RGB (Red, Green, Blue) el cual se basa en que cualquier color puede ser representado con una combinación de los tres colores denominados primarios, rojo, verde y azul. Dada la característica del color del fuego que varía entre naranja y amarillo hace que en el modelo RGB presente grandes cantidades de muy grandes de rojo, grandes de verde y muy escasos de azul, esta caracterización es la que permitirá llevar a cabo su detección entre los otros elementos de una imagen.

Sin embargo el modelo RGB presenta el problema está en que la unión entre el fuego y otros elementos si bien es posible el establecer la frontera resulta altamente complejo. Para solucionar este problema se hace uso del modelo de color HSV el cual se compone de tres variables: tono (*Hue*), saturación y brillo (*value***)**. Asi la saturación indica lo alejado que esta el valor del eje blanco y negó mientras que el brillo representa la altura del valor dentro del eje blanco y negro. Este modelo de color permite establecer las delimitaciones de una forma más sencilla y con un alto porcentaje de acierto permitiendo de una forma rápida determinar qué elementos son fuego en una imagen.

3.3.2 Detección del fuego por humo

Debido a las problemáticas expuestas puede ser más factible buscar humo que buscar fuego ya que este puede observarse desde una fase muy precoz del incendio y a mayores distancias mientras que el fuego

La detección del humo se hace en base a cuatro características:

- Novedad: El humo será un elemento nuevo que aparece en una secuencia de video
- Velocidad: El Humo se mueve a baja velocidad frente a otros nuevos elementos
- Tonalidad: Su color tiende a ser blanco-gris
- Volumen: El humo será un elemento grande.

De acuerdo a estas cuatro características el algoritmo para la detección del humo se realiza mediante cuatro etapas o procesos en la imagen que son:

- Sustracción de fondo: En una secuencia de video detectara aquellos elementos que sean nuevos.
- Eliminación de elementos rápidos: Se queda con los elementos rápidos
- Detección de color: Descarta aquellos colores que sean distintos del humo
- Análisis de componentes conexas: Los elementos deberán ser suficientemente grandes

Además existe una quinta etapa que es el no aprendizaje, lo que asegura que los elementos clasificados como humo no sean aprendidos por el algoritmo y dejen de ser detectado en la sustracción de fondo que detecta los elementos nuevos.

La detección de humo es más factible en sistemas fijo ya los movimientos capara pueden hacer que la etapa de sustracción de fondo no sea valida al entender el movimiento como un elemento nuevo y los descarte. Para contrarrestar los movimientos rápidos se pueden aplicar varias técnicas, existe una técnica que cosiste en eliminar las detecciones que no se producen de forma continuada.

3.4 Descripción Funcional de la UAV en la detección de incendios forestales

Se han visto en apartados anteriores los componentes básicos de la unidad Dron, para explicar la funcionalidad de la unidad se describen a continuación las funciones básicas de cada uno de los subsistemas que se incorporaran a la aeronave para cumplir los objetivos de detección de incendios forestales:

- **Estructura y sistema de propulsión:** Su estructura, en fibra de carbono, permitirá el acople del sistema de propulsión el cual se basa en el modelo de cuatro motores eléctricos descrito en apartados anteriores y debidamente regulados mediante dispositivos ESC. A esta estructura se incorporara un sistema de anclaje y estabilización (*Gimbal*) donde será instalada la unidad del sistema central la cual incorpora el dispositivo de captura de video (Cámara). Dicho sistema de anclaje permite la rotación sobre 2 ejes (opcional 3 ejes) proporcionando así al sistema de captura de video la estabilidad necesaria, independientemente de las variaciones que pueda sufrir la aeronave como consecuencia de su navegación, para efectuar la grabación y/o la visión imagen de video.

- **Sistema Central:** Esta es la encargada del control y coordinación de todos los subsistemas de la UAV. Para ello:
    - Realiza la gestión y control de los dispositivos de comunicaciones que permiten a la unidad su acceso a la red internet de forma inalámbrica, así como la interacción del sistema con el usuario final.
    - Estará dotada de diversos modos de comunicación como son: Adaptador red inalámbrica (Wi-Fi), sistema de radiocontrol de corto alcance (2.4Ghz) y modem GSM-GPRS. (*Global System Mobile-General Packet Radio Service*) y sistema de transmisión de video (5.8GHz).
    - Realizara las funciones de captura de video y grabación de imágenes, presentando dicha captura de video a través de su salida AV compuesto (Audio-Video Compuesto), a la cual le será superpuesta en un paso previo mediante el sistema OSD (*On Screen Display*) todos los datos de telemetría. En su conjunto la señal de video obtenida será aplicada al transmisor de video con lo que se obtendrá sobre dichas capturas el posicionamiento exacto de las mismas presentándolas en la propia transmisión de video realizada por la unidad.

- **Sistema Controlador de Vuelo**: Esta realiza las funciones de piloto de abordo de la unidad, al igual que las funciones de obtención de datos de geo-.posicionamiento y de telemetría poniéndolos a disposición de la unidad central mediante su interface UART. Igualmente llevara a cabo los planes de vuelo programado que el usuario le asigne a través de la unidad de sistema central y permitirá el control manual de vuelo mediante radio enlace de corto alcance (aprox. 1Km).

- **Sistema de sensores (US)** estos serán los encargados de la medición de los factores atmosféricos de la zona poniéndolos a disposición de la unidad central mediante su interface de comunicaciones I^{2}C (*Inter-Integrated Circuit*) ó SPI (*Serial Peripheral Interface*) por los cuales disponen de comunicación con el sistema central.

- **Aplicación de Monitorización y Control por el usuario**, dicha aplicación está ubicada en servidor WEB remoto proporcionando al usuario una interfaz para interacción con la unidad y visualización de la información en todo momento.

- **Unidad de alimentación (batería)**: Esta será la encargada de dotar a la unidad de la energía eléctrica necesaria para su correcto funcionamiento, proporcionando una autonomía de vuelo mínima de 15 minutos.

Su funcionamiento es el siguiente: Posicionada la unidad *Dron* en una estación base esta quedara en modo de espera para ser comandada en alguno de sus posibles modos los cuales son:

- Modo manual mediante emisora de radio control (2.4GHz) de corto alcance.
- Modo remoto mediante conexión remota para realización de vuelo programado.

El sistema en el estado de reposo en su base aceptara conexiones remotas desde el centro de control, permitiendo al usuario realizar la programación de planes de vuelo. Durante dicho plan de vuelo la unidad recabara los datos necesarios de los correspondientes subsistemas incorporados a la unidad. Efectuado el tratamiento de datos correspondiente serán enviados a la aplicación WEB para su monitorización a la vez que efectúa el registro de los datos correspondientes factores climáticos y de posicionamiento GPS en la base de datos para su posterior tratamiento, si se requiere, por parte del usuario.

Si el sistema en estado de reposo es controlado en modo manual, el transmisor de corto alcance estará dotado de un receptor de video a través del cual la persona que efectúa el control podrá disponer en pantalla de la captura de video realizada por el sistema a la vez que podrá pilotar el *Dron* mediante vuelo guiado por video FPV (*Firs Person View*).

3.4.1 Diagrama de Bloques de la aplicación

En la siguiente figura se represente el diagrama de bloque de la aplicación.

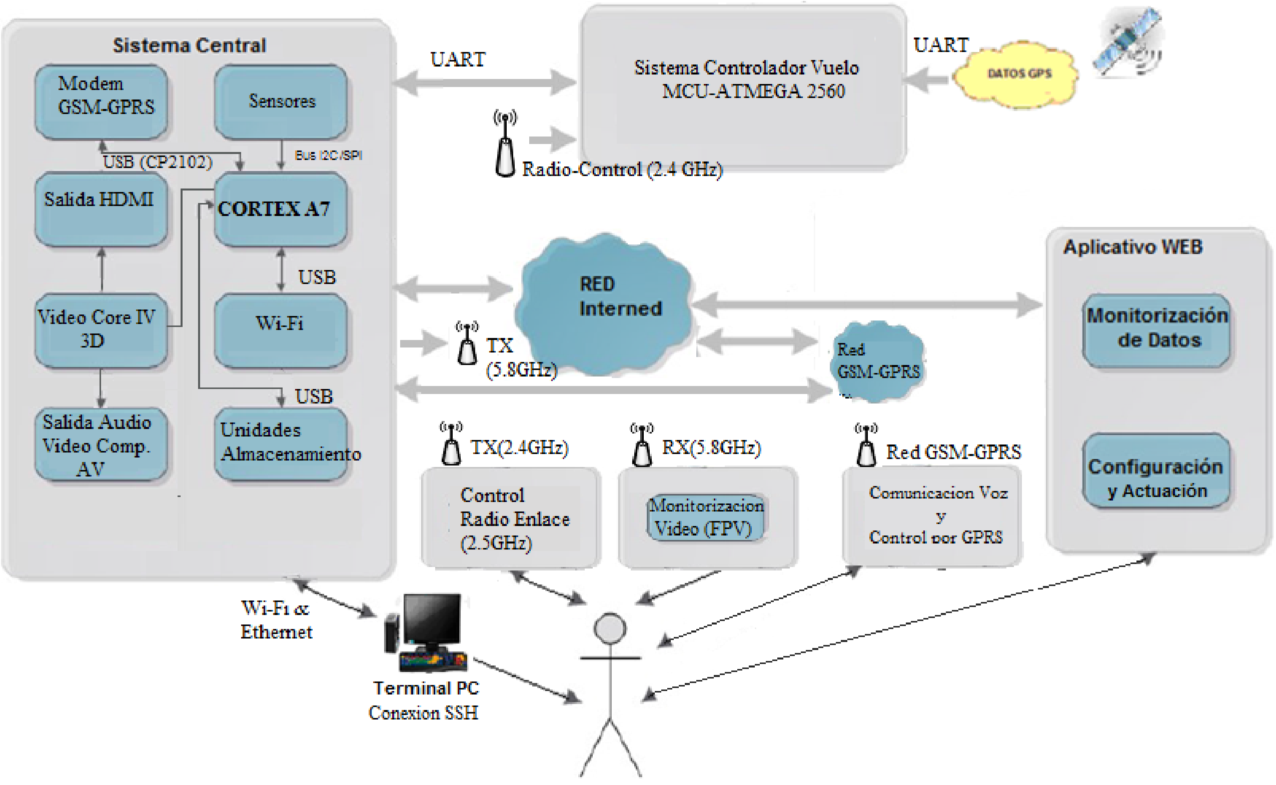


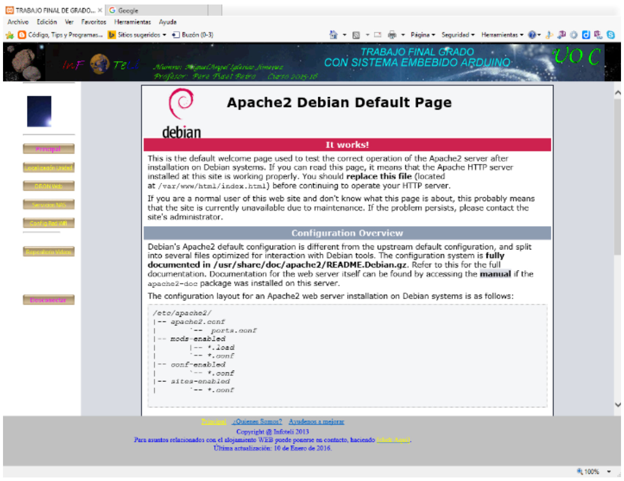
Figura 7 Diagrama bloque Aplicación

3.4.2 Posicionamiento y Comunicación en la Red

El sistema queda posicionado en la Red de las siguientes formas:

- Mediante la implementación de un servidor WEB (Local) propio, el cual puede ser configurado en sus valores de IP, y puerto de uso HTTP. Dicho servidor podrá ser debidamente programado para que a la unidad pueda presentar para su monitorización los datos atmosféricos obtenidos en el muestreo que realiza, el posicionamiento GPS de la estación, el estado de las redes de que dispone (*Wifly, GSM-GPRS*), e incluso si está en estado de reposo podrá activar servidor CAM permitiendo visionar su entorno. (En la figura 8 se puede ver captura la página de instalación del servidor Local de la unidad Dron).
- Mediante el uso de red GSM-GPRS, la unidad podrá efectuar envíos de mensajes SMS, y de tramas de datos via GPRS. Pudiendo ser también localizada mediante aplicaciones existentes para la localización y posición de dispositivos de telefonía móvil. En anexo B Manual de montaje pueden verse pruebas efectuadas del envío de SMS a mediante la unidad.

- Otra mediante el uso de un servidor WEB de aplicación (requiere bien un Equipo servidor WEB o bien un hosting en la red). Este debe ser instalado en la red, a través de este el sistema además de monitorizar los datos anteriormente indicados, permite la interacción del usuario con el sistema. (En la figura 9 puede observarse una captura de la página principal de la aplicación WEB).



Apache2 Debian Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

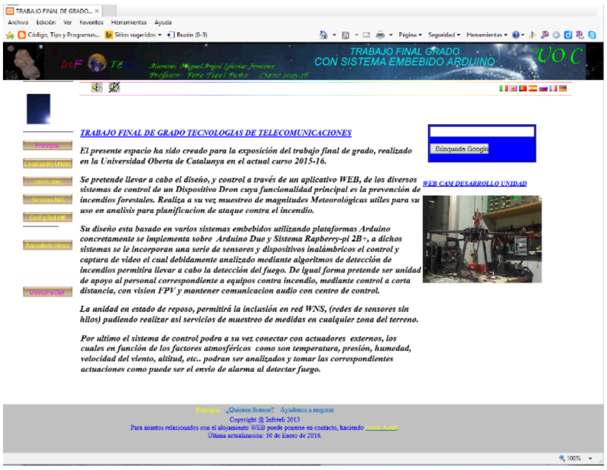
Principal ¿Quienes Somos? Ayudenos a mejorar

Copyright @ Infoteli 2013

Para asuntos relacionados con el alojamiento WEB puede ponerse en contacto, haciendo [click Aqui].

Última actualización: 10 de Enero de 2016.

Figura 8 Servicio WEB Unidad Dron



TRABAJO FINAL DE GRADO TECNOLOGIAS DE TELECOMUNICACIONES

El presente espacio ha sido creado para la exposición del trabajo final de grado, realizado en la Universidad Oberta de Catalunya en el actual curso 2015-16.

Se pretende llevar a cabo el diseño, y control a través de un aplicativo WEB, de los diversos sistemas de control de un Dispositivo Dron cuya funcionalidad principal es la prevención de incendios forestales. Realiza a su vez muestreo de magnitudes Meteorológicas utiles para su uso en analisis para planificacion de ataque contra el incendio.

Su diseño esta basado en varios sistemas embebidos utilizando plataformas Arduino concretamente se implementa sobre Arduino Due y Sistema Rapberry-pi 2B+, a dichos sistemas se le incorporan una serie de sensores y dispositivos inalámbricos el control y captura de video el cual debidamente analizado mediante algoritmos de detección de incendios permitira llevar a cabo la detección del fuego. De igual forma pretende ser unidad de apoyo al personal correspondiente a equipos contra incendio, mediante control a corta distancia, con vision FPV y mantener comunicacion audio con centro de control.

La unidad en estado de reposo, permitirá la inclusión en red WNS, (redes de sensores sin hilos) pudiendo realizar asi servicios de muestreo de medidas en cualquier zona del terreno.

Por ultimo el sistema de control podra a su vez conectar con actuadores externos, los cuales en función de los factores atmosféricos como son temperatura, presión, humedad, velocidad del viento, altitud, etc.. podran ser analizados y tomar las correspondientes actuaciones como puede ser el envio de alarma al detectar fuego.

Principal ¿Quienes Somos? Ayudenos a mejorar

Copyright @ Infoteli 2013

Para asuntos relacionados con el alojamiento WEB puede ponerse en contacto, haciendo [click Aqui].

Última actualización: 10 de Enero de 2016.

Figura 9 Aplicativo WEB

Estas Web se han posicionado y configurado en las siguientes direcciones:

- La aplicación WEB estará alojada en servidor instalado expresamente para el desarrollo de este proyecto en la URL: https://infoteli.org/proto.
- El servidor WEB local alojado en la unidad *Dron* será accesible mediante la URL: http://www.infoteli.org:8090 o bien desde el aplicativo WEB en la opción de menú WEB Dron. Dado que ha sido implementado como mejora con la finalidad de ver la posibilidad de ofrecer video las capturas de video mediante esta otra opción actualmente no se ha programado ninguna aplicación web para el presentando únicamente como puede verse en la figura 8 la página de instalación.

3.4.3 Interacción entre los diferentes Sub-Sistemas

Como puede verse en el diagrama de bloques aplicación (Figura 7), existe interconexión con diferentes dispositivos y aplicaciones, realizándose estos mediante los diversos medios de comunicación así los más destacados son:

- Entre dispositivo GPS y unidad de controlador de vuelo se lleva a cabo mediante un puerto UART
- Entre dispositivo Compas y Acelerómetro se realiza mediante bus I2C del controlador de vuelo los cuales son accesibles mediante el sistema central por UART, haciendo uso del protocolo MAVlink (*Micro Air Vehicule Comunication Protocol*).
- Entre dispositivo regulador potencia motores y controlador de vuelo a través de sus puertos PWM.
- Sistemas de comunicación con Usuario puede efectuarse mediante los siguientes medios:
    - Enlace Radio Control TX-RX (2.4GHz)
    - Enlace vía LAN-WAN (Protocolo MAVLink) haciendo uso del Software específico *Mission Planner* para planificación de planes de vuelo
    - Enlace vía LAN-WAN mediante conexión SSH se tendrá acceso al sistema central.
    - Enlace vía GSM-GPRS
- Sistema de Monitorización WEB LOCAL haciendo uso de dispositivo Wi-Fi.
- Sistema comunicación con usuario mediante PC hace uso de dispositivo USB
- Los dispositivos sensores instalados en el sistema efectúan comunicación a su vez por distintos buses esto son:
    - Sensor Temperatura-Humedad DTH22 utiliza BUS SPI (de un único hilo)
    - Sensores de Temperatura Presión, y Luminosidad mediante Bus I2C

En la siguiente figura pueden verse detallados la los distintos interfaces de comunicación de los que hace uso cada dispositivo y aplicativos.

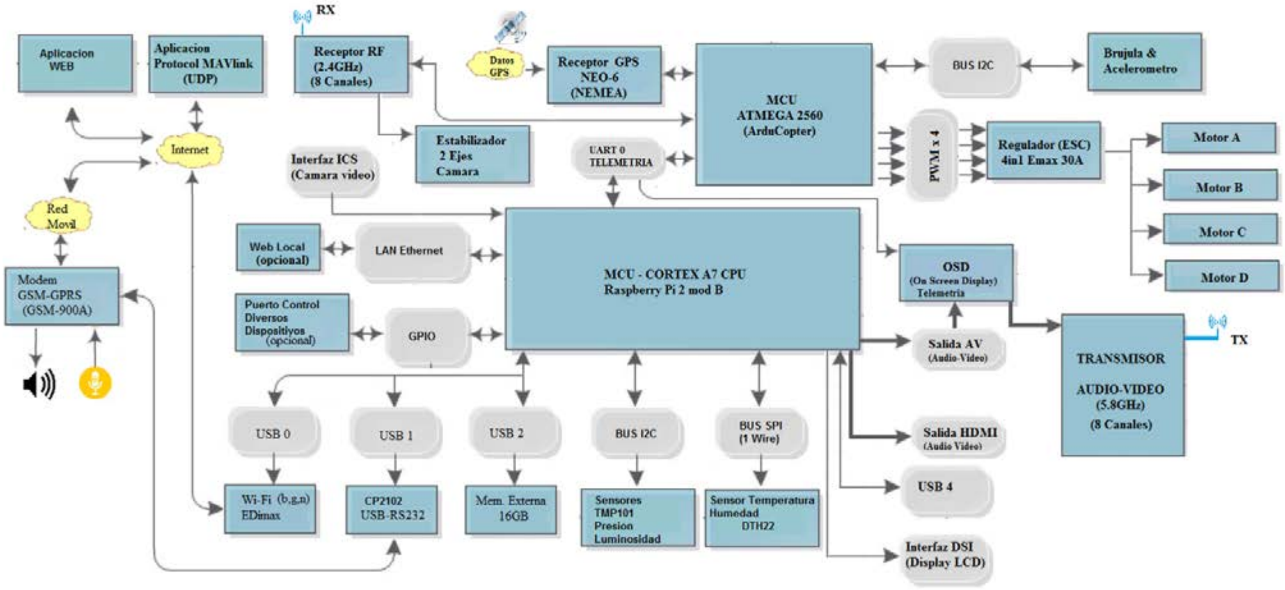

RX
Aplicacion WEB
Aplicacion Protocol MAVlink (UDP)
Receptor RF (2.4GHz) (8 Canales)
Datos GPS
Receptor GPS NEO-6 (NEMEA)
MCU ATMEGA 2560 (ArduCopter)
BUS I2C
Brujula & Acelerometro
Internet
Interfaz ICS (Camara video)
Estabilizador 2 Ejes Camara
UART 0 TELEMETRIA
PWM x 4
Regulador (ESC) 4in1 Emax 30A
Motor A
Motor B
Motor C
Motor D
Red Movil
Web Local (opcional)
LAN Ethernet
MCU - CORTEX A7 CPU Raspberry Pi 2 mod B
OSD (On Screen Display) Telemetria
Modem GSM-GPRS (GSM-900A)
Puerto Control Diversos Dispositiyos (opcional)
GPIO
Salida AV (Audio-Video)
TRANSMISOR AUDIO-VIDEO (5.8GHz) (8 Canales)
TX
USB 0
USB 1
USB 2
BUS I2C
BUS SPI (1 Wire)
Salida HDMI (Audio Video)
USB 4
Wi-Fi (b,g,n) EDimax
CP2102 USB-RS232
Mem. Externa 16GB
Sensores TMP101 Presion Luminosidad
Sensor Temperatura Humedad DTH22
Interfaz DSI (Display LCD)

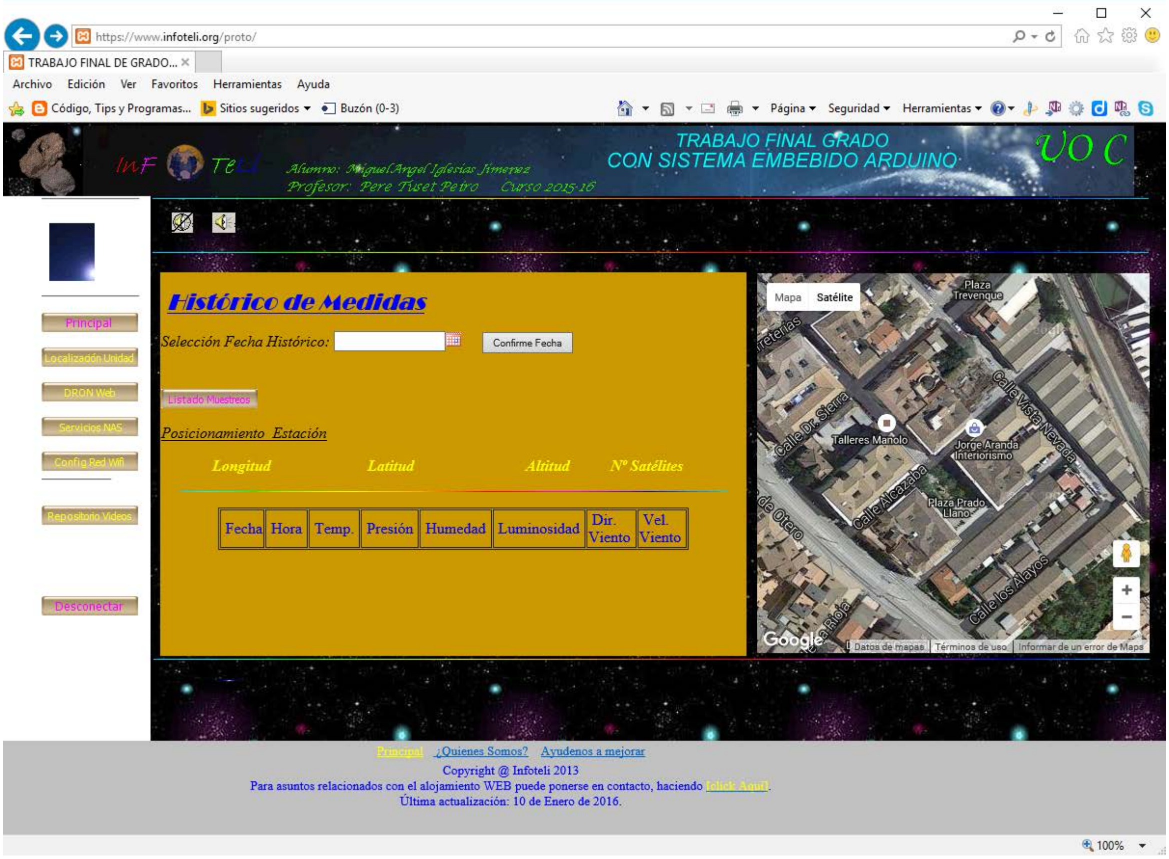
Figura 10 Diagrama Bloques interacción objetos del sistema

3.4 Aplicación control y monitorización

El sistema interactúa con el usuario de tres formas diferentes:

- Conexión directa mediante teclado y monitor con al sistema central, o bien mediante un PC de forma remota por conexión SSH, o haciendo uso igualmente de forma remota mediante software específico de aplicación (*Mission Planner*)
- Mediante el uso de emisora de control de RF(2.4GHz), para Radio-Control de la UAV..
- Mediante Aplicativo WEB, a través de internet.

3.4.1 Diseños interfaces usuarios



TRABAJO FINAL GRADO
CON SISTEMA EMBEBIDO ARDUINO

Alumno: Miguel Angel Iglesias Jimenez
Profesor: Pere Tuset Peiro Curso 2015-16

Histórico de Medidas

Selección Fecha Histórico:

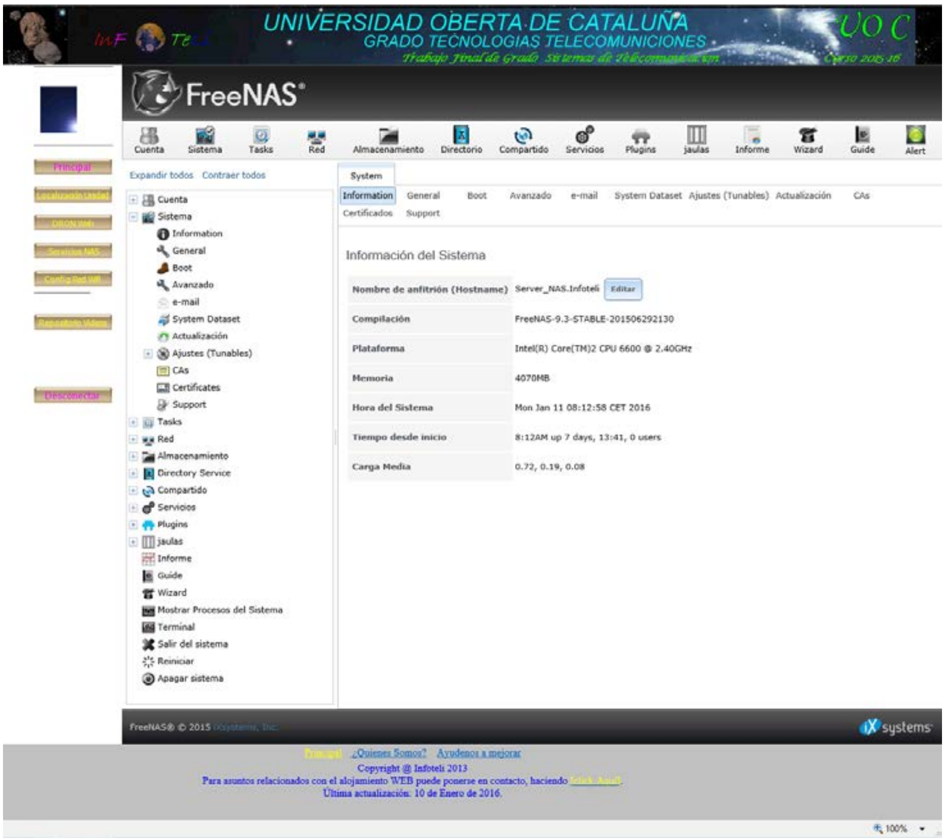
Posicionamiento Estación

Longitud	Latitud	Altitud	Nº Satélites

Fecha	Hora	Temp.	Presión	Humedad	Luminosidad	Dir. Viento	Vel. Viento

Principal ¿Quienes Somos? Ayudenos a mejorar
Copyright @ Infoteli 2013
Para asuntos relacionados con el alojamiento WEB puede ponerse en contacto, haciendo [click Aquí].
Última actualización: 10 de Enero de 2016.

Figura 11 Interfaz opción Menú Localización Unidad.

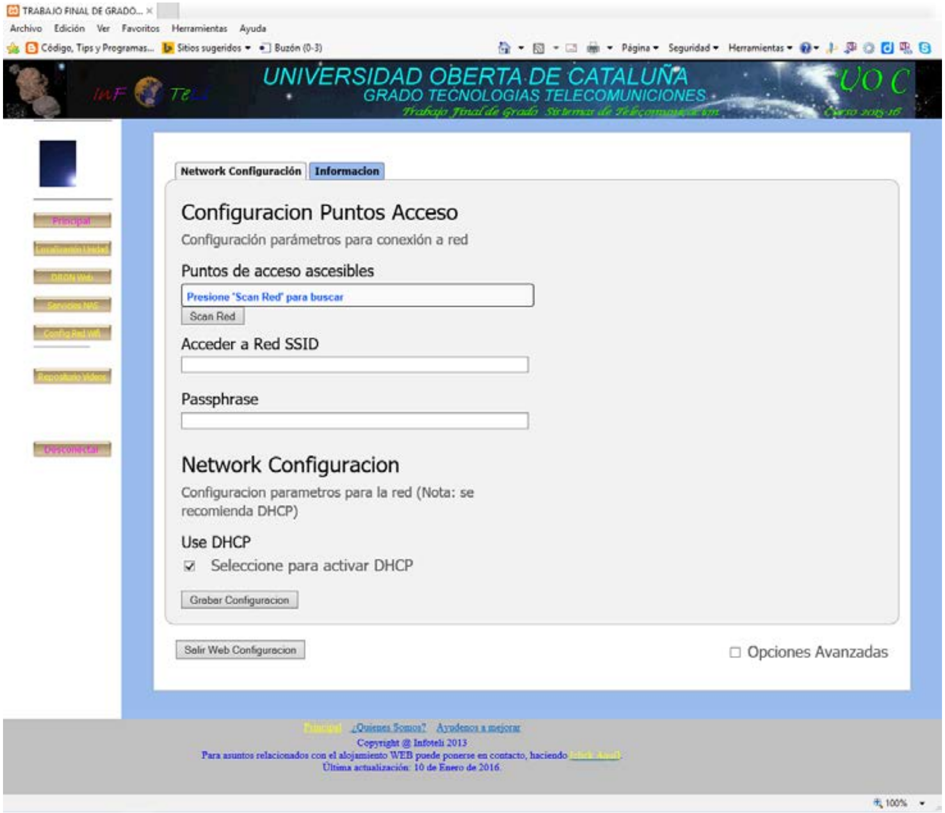


Información del Sistema

Nombre de anfitrión (Hostname)	Server_NAS.Infoteli
Compilación	FreeNAS-9.3-STABLE-201506292130
Plataforma	Intel(R) Core(TM)2 CPU 6600 @ 2.40GHz
Memoria	4070MB
Hora del Sistema	Mon Jan 11 08:12:58 CET 2016
Tiempo desde inicio	8:12AM up 7 days, 13:41, 0 users
Carga Media	0.72, 0.19, 0.08

Principal ¿Quienes Somos? Ayudenos a mejorar
Copyright @ Infoteli 2013
Para asuntos relacionados con el alojamiento WEB puede ponerse en contacto, haciendo [click Aquí].
Última actualización: 10 de Enero de 2016.

Figura 12 Interfaz Opción Menú Servicios NAS.



Configuracion Puntos Acceso

Configuración parámetros para conexión a red

Puntos de acceso accesibles

Presione 'Scan Red' para buscar

Acceder a Red SSID

Passphrase

Network Configuracion

Configuracion parametros para la red (Nota: se recomienda DHCP)

Use DHCP

☑ Seleccione para activar DHCP

□ Opciones Avanzadas

Principal ¿Quienes Somos? Ayudenos a mejorar
Copyright @ Infoteli 2013
Para asuntos relacionados con el alojamiento WEB puede ponerse en contacto, haciendo [click Aquí].
Última actualización: 10 de Enero de 2016.

Figura 13 Interfaz Opción Menú Configuración Red

3.3 Diagrama Estructura aplicativo WEB

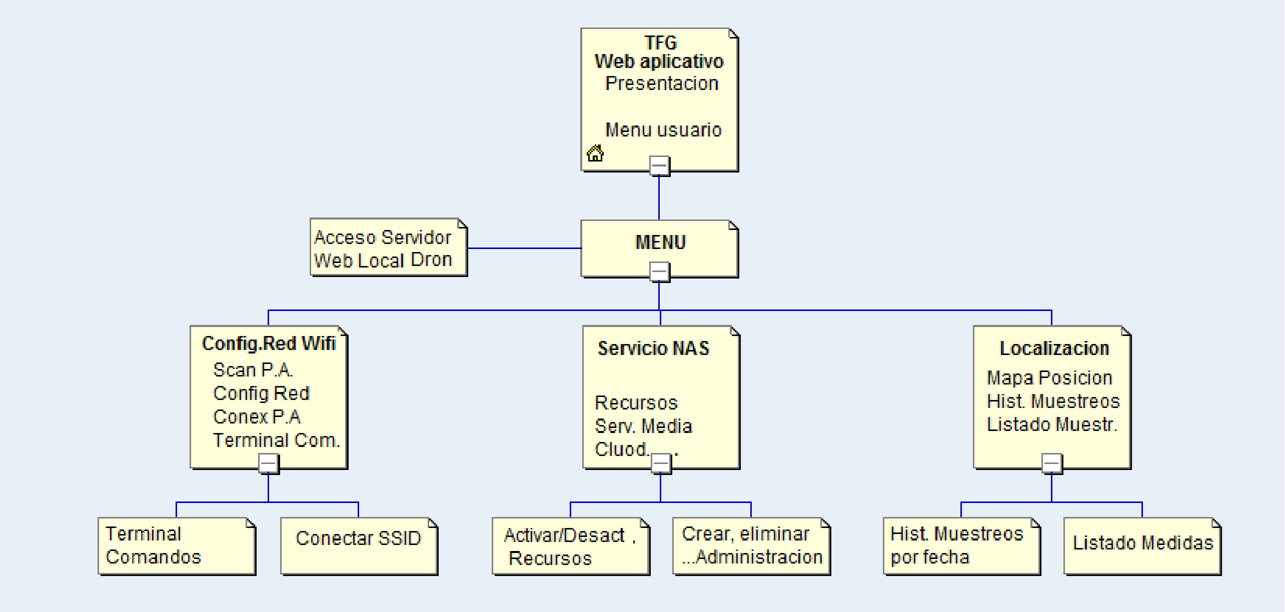


Figura 14 Diagrama Estructura Aplicación WEB

4. Descripción detallada del Hardware

4.1 Requisitos mínimos Sistema Central y Controlador de Vuelo

El Sistema Central deberá de cubrir unas mínimas prestaciones y cumplir con unos requisitos como son:

- Disponer de buses con los distintos protocolos como son UART, I2C, SPI,
- Disponer de la posibilidad video en alta definición
- Una velocidad de procesado que permita él envió mostrar en su salida de video las capturas realizadas en tiempo real y en una definición aceptable para posterior procesado del mismo.
- Reducido tamaño para poder ser ubicada en la unidad *Dron* y a ser posible con la posibilidad de incorporarse a esta como un módulo ensamblándose de forma fácil
- Posibilidad de almacenamiento externo que pudiera requerir para grabación de video e imágenes.
- Bajo consumo y reducido tamaño.

A estas mínimas prestaciones es deseable un mayor coste y que sea de fácil adquisición sin problemas en proveedores por las limitaciones de tiempo existentes al igual que disponga de comunidades que apoyen dichos proyectos y puedan aportar un soporte mínimo en sistemas operativos y librerías para su uso.

Se ha podido observar a través de búsquedas de información en internet ver que existen en el mercado amplia variedad de dispositivos con prestaciones muy similares a las de un PC, pero con unas dimensiones muy reducidas (mini-PC). A continuación se presentan tres dispositivos mini-PC que posibles candidatos a ser seleccionados para su uso como sistema central.

	Raspberry PI 2	**A20 OLinuXino Lime2**	**Radxa Rock Pro**
SoC	BroadCom BCM2836	A20 Dual Core T	
CPU	Quad-Core ARM Cortex-A7 900MHz	Dual Core ARM Cortex A7 1000MHz	Quad-Core ARM Cortex- 1,6MHz
GPU	Video Core IV	Mali 400 GPU	Mali 400
RAM	1GB	1GB	2GB 8GB NAND Flash
USB	4	2	2
Video	Compuesto y HDMI	HDMI	RGA, HDMI
Audio	Jack HDMI	HDMI	RGA
Boot	Micro SD	Micro SD	Digital S/PDIF, Jack, micro integrado en placa.
Red	Ethernet 10/100	1000	Ethernet 10/100M Wifi
Buses	UART(1),I2C(1), SPI(1)	UART(1),I2C(1), SPI(1)	UART(1),I2C(1), SPI(1)
GPIO	40	160	80
SATA	NO	SI	NO
Interfaces Adic.	CSI (Cámara) y LCD	NO	IR, USB-OTG,
Consumo	400mA a 800mA	500mA a 1200mA	800mA a 2000mA
Alimentación	Micro USB/GPIO	USB	
Tamaño	85 x 56 mm	84X60mm	100x80
Precio	50€	45€	95€

Tabla 1 Comparativa mini-PC

Actualmente existen diversas comunidades que trabajan en proyectos para el desarrollo de firmware y software para el control de vuelo de *Drones* de diverso tipo como son: Multi-rotores, aviones, helicópteros, etc... A se presenta una lista de los proyectos existentes actualmente en *open-source*.

PROYECTOS	**Controladores**	**Soporte**	**Sensores**	**Configuraciones**
APM Copter	APM:ATMEGA 2560 PixHawk STM32	Esquemas y PCB,s	Gyro, Acelerometro GPS, Mag, Alt, Temp	Tri, Quad, Hexa Octo.
AeroQuad	Arduino Pro Uno/Mini/Mega. STM 32	Esquemas y PCB,s	Gyro, Acelerometro GPS, Alt,	Tri, Quad, Hexa Octo.
MultiWiiCopter	Arduino Pro Mini Arduino Mega	Solo Esquemas	Gyro, Acelerometro GPS, Mag, Alt.	Tri, Quad, Hexa
OpenPilot	STM32	Esquemas y PCB,s	Gyro, Acelerometro GPS, Mag, Alt.	Tri, Quad, Hexa Octo
UAVP **(Universal Aerial Video Plataform)**	ATMEGA644	Solo Esquemas	Gyro, Acelerometro GPS, Mag, Alt.	Tri, Quad, Hexa Octo.

Tabla 2 Proyectos Controladores vuelo de código abierto

De estos hemos podido comprobar que APM Copter, anteriormente *ArduPilotMega* de donde ha tomado sus actuales APM, es la más consolidada trabajando activamente en desarrollo de vehículos aéreos y terrestres dispone de diferentes categorías[x] siendo su controlador clásico el ATMega2560 inspirado en Arduino. Esta comunidad desarrolla firmware diferenciado para cada tipo de configuración, siendo su software característico el APM:Mission Planner. Actualmente esta comunidad está desarrollando un controlador basado en microprocesador STM32 el cual presenta mejoras importantes en cuanto a capacidad de cómputo y con opciones para agregar segundos dispositivos de telemetría y GPS entre otros.

Para el desarrollo del proyecto se requerirán otros componentes electrónicos así como herramientas específicas tanto de electrónica como de informática. Por lo que a continuación se pasa a describir cada uno de los componentes seleccionados

4.2 Descripción de los módulos seleccionados

4.2.1 CPU Sistema Central y Captura-Procesado de video

Para el control del sistema central y el procesado de las captura de video se ha seleccionado la mini-PC Rasperry Pi 2 B. El motivo de su selección respecto a las otras presentadas es el disponer de su propia interfaz para cámara de video existiendo actualmente cámaras IR (Infrarroja) y NoIR (No Infrarroja) específicas de Raspberry Pi de un relativo bajo coste. Otro motivo ha sido el disponer de hasta 4 puertos USB de los que se tiene previsto utilizar tres.

Figura 15 Sistema embebido Raspberry pi 2+B

De igual manera se ha podido ver que existe un buen soporte de apoyo por parte de la comunidad *Raspberry Pi* con código open-source en la Red, diversos sistemas operativos a utilizar incluyendo el sistema operativo de tiempo real RTLinux, lo que permitirá en futuros trabajos poder llevar a cabo mejoras en el proyecto.

4.2.2 MCU Controlador de vuelo

Como sistema controlador de vuelo se ha seleccionado al APM Copter 2.6 con procesador ATmega 2560.

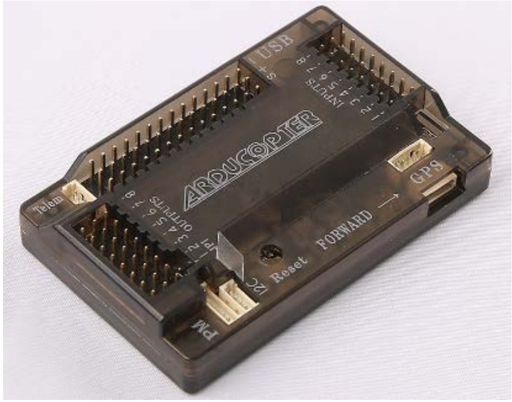


Figura. 16 Arduino APM Copter 2.6 (ArduCopter)

El sistema *ArduCopter* cuenta con capacidad de vuelo autónomo basado en *way-point*, planificación de la misión y telemetría en tiempo real a través de una estación en tierra. Motivo por el que se ha seleccionado ya que la mayoría de las soluciones para multi-rotores solo cuentan con soporte por radiocontrol.

Si bien existe una MCU superior actualmente como es la Pixhawk esta tiene un coste más elevado y aunque incluye mejoras sustanciales como son: La posibilidad de conexión de un segundo dispositivo GPS, un procesador con una mayor potencia de cálculo así como un número mayor de puestos dicho no son necesarios para este proyecto.

Por otro lado, el APM 2.6 ArduCopter presenta ventajas como ser multiplataforma por lo que es compatible con Windows, Mac y Linux. La utilidad de configuración y planificación puede ser utilizada en entorno gráfico o bien mediante línea de comandos utilizando el protocolo MAVLink lo que permite que pueda ser planificado a través de redes donde el ancho de banda este limitado e incluso con programación adecuada mediante GSM.

Por ultimo indicar que es totalmente compatible con estándares de la industria de robótica líderes como son el sistema ROS, o el protocolo MAVLink el cual permite a través de una librería poco pesada recibir los datos del dispositivo controlador de vuelo y enviarle comandos, pudiéndose empaquetas estructura C a través de canales seria con eficacia y enviar dichos paquetes a la estación de control en tierra. Razones estas que unidas a su bajo coste lo convierten en ideal para este proyecto.

4.2.3 Estructura *Quap-rotor* (TAROT IRON 650)

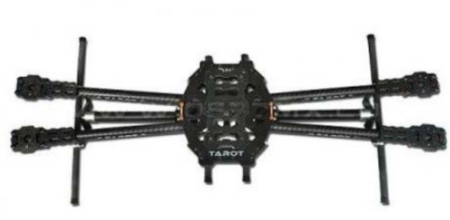


Figura. 17 Estructura TAROT IRON 650

La estructura se selecciona en fibra de carbono por sus características de peso-resistencia y dada la cantidad de dispositivos que se han de acoplar y teniendo presente que la unidad no debe superar 2,5kg de peso para que pueda quedar incluida en normativa vigente [7]. Se espera un peso total de 2Kg.

De las diversas estructuras vistas en este material se selecciona la referenciada como *TAROT IRON 650* por su coste y posibilidades de adaptación ya que esta será sometida a modificaciones para el correcto acople de los dispositivos de captura y diversos módulos de comunicaciones. Esta estructura tiene como especificaciones:

- Distancia entre sus ejes 650mm
- Peso total bruto: 450g/480g
- Configuración recomendada por fabricante Hélices de 14-17 pulgadas.

5.2.4 Motores propulsores

Figura 18 Motor brushless A2212/13T(1000Kv)

El motor seleccionado es un motor *Brushless* (sin escobillas) estos proporcionan movimientos suaves y rápidos, presentando ventajas frente a los motores comunes de DC (Corriente continua) como son:

- Mayor eficiencia (menos perdidas por calor)
- Mayor rendimiento (por tanto mayor duración baterías)
- Menor peso a la misma potencia
- Requieren un menor mantenimiento

Relación velocidad/par motor casi constante

- Mayor potencia en el mismo tamaño
- Mejor disipación de calor
- Rango de velocidad elevado a l no tener limitación mecánica
- Menor ruido electrónico

Como desventajas presentan un mayor costo de construcción, requieren de un circuito de control más complejo y siempre necesario para su funcionamiento. Por otro lado los motores normales con escobillas son más propensos, debido a los roces, a producir chispas y averías.

De la denominación de dada en un motor (p.e A2212/13T -1000KV) se puede conocer que:

- Los dos primeros dígitos (22): Es el diámetro de la parte interna del motor (estator) en milímetros. Indican el torque que puede ejercer a mayor número podrá mover hélices más grandes y con mayor pitch por tanto podrá levantar mayor peso con menor velocidad.
- Los dos siguientes dígitos (12): Diámetro de la parte internet del motor medio en milímetros.
- 1000KV: Indica las RPM (Revoluciones por minuto) por voltio. Así si se aplica una tensión de 11.1 voltios (batería de 3S) puede llegar a 11100 RPM.

Hay que tener en cuenta que a mayor RPM las hélices deben ser más pequeñas lo que permite grandes velocidades pero reducen en gran medida la eficiencia y por tanto el tiempo de vuelo. Así valores del orden de 2300 KV con hélices de 5 a 7 pulgadas suelen ser utilizados para carreras.

Mientras que motores del orden de 400KV a 900Kv, son utilizadas para *Drones* grandes con cargas pesadas, no son agiles ni rápidos pero si muy estables aunque necesitan de hélices más grandes por encima de 10 pulgadas.

Así para la selección del motor acorde a las RPM y fuerza de empuje que se requiere por motor puede efectuarse el cálculo mediante la aplicación de la fórmula:

$$F = 1.225\frac{\pi \cdot (0.0254 \cdot d)^2}{4}\left[\left(RPM_{prop} \cdot 0.0254\ \cdot pitch \cdot \frac{1}{60sec}\right)^2 - \left(RPM_{prop} \cdot 0.0254\ \cdot pitch \cdot \frac{1}{60sec}\right)V_o\right] \cdot \left(\frac{d}{3.29546 \cdot pitch}\right)^{1.5}$$

La cual simplificada:

$$F = 4.392399 \cdot 10^{-8} \cdot RPM\frac{(d)^{3.5}}{\sqrt{pitch}}\left(4.2333 \cdot 10^{-4} \cdot RPM_{prop} \cdot pitch\ \cdot pitch - V_0\right)$$

Dónde: F =es la Fuerza de empuje en Newtons pitch=Prop. Pich (pulgadas).
RPM= Rotaciones por minuto hélice. D=prop. Diámetro (pulgadas).
V_0=Velocidad avance en m/s

Para convertir Newtons en gramos basta con multiplicar por $\frac{1000}{9.81}gr$

De una forma mucho más simplificada los fabricantes recomiendan para una primera aproximación la formula general en base a la característica del motor que proporcionan (KV)

$$Empuje\ por\ motor\ (KV) = \frac{Peso\ total\ (gr)x2}{Nº\ motores} = \frac{2000x2}{4} = 1000Kv$$

En este caso se ha seleccionado el motor *Brushless* A2212/13T 1000Kv, según características del fabricante con un empuje de 900gr, y un consumo máximo de 20A.

En anexo correspondiente a manual de montaje se pueden observar los datos obtenidos de pruebas empíricas realizadas al motor seleccionado.

4.2.5 Hélices

Las hélices es un dispositivo el cual está constituido por un numero variable de aspas o palas (2, 3, 4,..) y que al girar alrededor de un eje producen una fuerza propulsora. Cada pala está formada por un conjunto de perfiles aerodinámicos los cuales cambian su Angulo de incidencia desde la raíz hasta extremo.

En este caso las hélices han sido proporcionadas en conjunción con los motores seleccionados por el propio fabricando las cuales son 12*4.5

4.2.6 Reguladores Velocidad Motores

El regulador de velocidad del motor (ESC) se ha seleccionado el módulo EMAX ESC 30Ax4, principalmente por guardar un amplio margen en su capacidad de suministrar intensidad (30A) frente al consumo máximo del motor seleccionado (25A) y permite baterías de 2s a 4s lo que posibilita en un futuro incluir si fuese necesario unos motores de mayor potencia haciendo uso de baterías de voltaje (14.8).

Por ultimo su diseño presentando los cuatro reguladores necesarios en un único bloque los cuatro reguladores permitirá su acople en el centro de la estructura utilizada y no en los brazos lo que mejora el diseño

Figura.19 ESC EMAX 4In1 30A

4.2.7 Modulo GPS y Brujula HMC5883L

Incluye GPS NEO-6M Compatible con el protocolo NMEA, (*National Marine Electronics Association*). Dispone de interface serie a través de la cual el sistema de control recibirá de la red satelital datos de posicionamiento, altitud etc...

Se ha seleccionado este módulo ya que Incluye una brújula digital con conexión por bus I2C mediante la cual se dota al sistema de la capacidad de indicación de dirección a donde se dirige y la orientación del dispositivo en todo momento. Igualmente el modulo es compatible con el APM Copter seleccionado

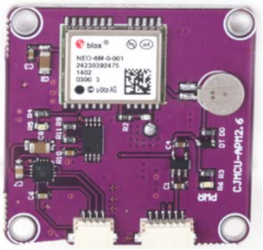


Figura.20 Modulo GPS NEO-6M

5.2.8 Modulo comunicación GSM-GPRS 900A

Modulo modem GSM dual banda, el cual permitirá la comunicación de voz con el dispositivo a la vez que posibilita el envío de mensajes y transferencia de datos mediante GPRS. Siendo así posible comunicar con la unidad desde cualquier localización.

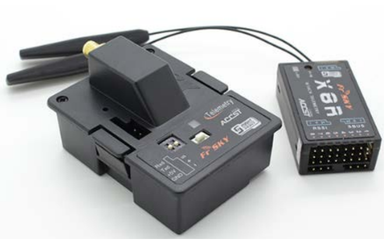
Figura.21 Modulo GPS NEO-6M

4.2.9 Sistema Radio-Control RF (2.4GHz)

Mediante este sistema de control se quiere obtener el control de la unidad en distancias cortas (para uso por el equipo de extinción como unidad de apoyo). Este se encuentra constituido por:

- Un emisor de radiofrecuencia Turningy 9XR PRO. Seleccionado principalmente por ser un dispositivo profesional el cual permite de forma fácil (mediante cambio de módulo), hacer uso de diversos tipos de módulos transmisores pudiendo ser utilizado para diversas frecuencias, igualmente permite conexionado de pantalla para FPV (Vuelo guiado por Video) teniendo en cuenta calidad precio no permite disponer de un control fiable.

Figura 22 Emisor RF Turnigy 9x Pro

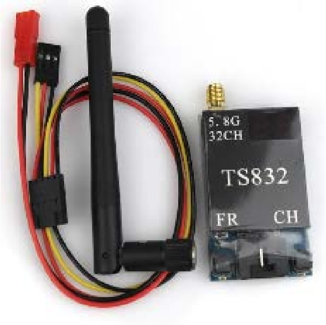


- Módulos TX RX (2,4Ghz). En la emisora se utilizara un Transmisor TX Fr Sky 2.4GHz, y en la unidad Dron módulo RX FrySky X8R en frecuencia de 2.4GHz el cual está dotado de 8 canales se ha seleccionado este modelo pues tiene la posibilidad de ampliar hasta 16 canales y dispone de telemetría. Igualmente su relación calidad precio lo hace asequible a las posibilidades del proyecto.

Figura 23 Módulos Transmisor y Receptor Fr Sky 8Ch

5.2.10 Sistema Transmisión – Recepción Audio-Video (5,8Ghz)

Transmisor Audio-Video Boscam 5832

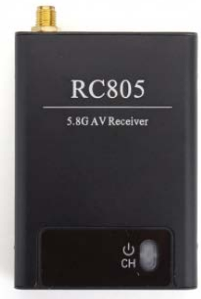


Se ha seleccionado por ser ideal por su potencia, que en principio no supera los 600mW lo que posibilita si es necesario hacer uso de amplificador existente en mercado pudiendo ampliar potencia hasta 2,5W, (hay que tener en cuenta la legalidad vigente en este sentido)[2].

Por otro lado haciendo uso de antena circular la cual según características indicadas por fabricante garantiza una alcance de 1Km a 1'5Km con una potencia de 600mW.

Figura 24 Módulos Transmisor Audio-Video 5'8Ghz

Receptor Audio-Video Boscam RC805



Se ha seleccionado el receptor RC805 el cual es acorde con los canales de transmisión del transmisor seleccionado, ambos han sido seleccionado principalmente por su relación calidad-precio, y teniendo en cuenta el alcance indicado en características por el fabricante.

Figura 25 Módulo Receptor Audio-Video 5'8Ghz

5.2.11Visualizador de datos en pantalla OSD (*On Screen Display*)

Es un sistema que puede ser programado para disponer los datos de telemetría así como el sistema de pilotaje de la UAV en pantalla. Lo que permitirá sin necesidad de consumo de recursos del sistema poder activarlo y sobre impresionar dichos los datos sobre la salida de video en paso previo a ser transmitidos.

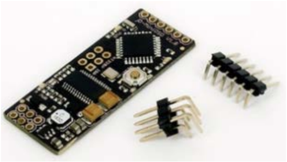


Figura 26 OSD (On Screen Display)

4.2.12 Sensor presión atmosférica

La presión se define como la fuerza que se ejerce sobre un área o superficie. Existen cinco tipos de presión: Presión absoluta, presión atmosférica o barométrica, presión diferencia, presión positiva relativa y presión negativa relativa, este proyecto se centra en la presión atmosférica o barométrica.

La presión barométrica es la que ejerce el aire en cualquier punto de la atmósfera, de ahí que se hable de forma general de presión atmosférica terrestre. Esta presión varía ligeramente con las condiciones meteorológicas y disminuye con la altitud. Existen diversos métodos para su medición como son mecánicos, neumáticos, electromagnéticos y elementos electrónicos. Dentro de estos se conocen cuatro tipos de sensores: Capacitivos, Sensor de Hall, sensor piezo-resistivo y sensor monolítico.

Sensor seleccionado: Efectuando algunas comparativas de los sensores vistos en el mercado se ha seleccionado el sensor MPL115A, principalmente por su bus de comunicación I2C, por su bajo consumo, alta precisión y reducido tamaño. El dispositivo esta calibrado en fabrica y se corresponde con el tipo de sensor monolítico.

Fig 27 Sensor Presión MPL115A

PRESION	Unidades	min.	Típica	Max
Resolución	kPa	-	0.15	-
	Bit	16	16	16
Rango	KPa	50	-	115
Conversión Time	ms		1.6	3
VDD	Voltios	2.3	3.3	5.5

Tabla 3 Características Sensor presión MPL115A

4.2.13 Sensor Iluminación y proximidad

La radiación solar se define como el conjunto de radiaciones electromagnéticas emitidas por el Sol, las cuales para alcanzar la superficie terrestre han de pasar la capa de ozono la cual absorbe la mayor parte de radiación. Su magnitud de medida es la irradiancia, la cual nos indica la energía por unidad de tiempo y área cuando alcanza la tierra. Su unidad son los W/m^2 aunque existe otra conocida como Lux. Esta es la luminosidad (lumen) por metro cuadrado. Esta será la que se utilizara a lo largo de este proyecto. Aunque no existe conversión directa entre Lux y W/m^2 si existe una equivalencia en función de la longitud de Onda.

Aquí hemos de indicar que en principio seleccionamos dos sensores independientes el S1087, pero que debido a plazos de entrega de proveedores se ha tenido que sustituir por uno más estándar el cual si fue recibido dentro de los plazos.

Sensor seleccionado: El sensor seleccionado para medición de Iluminación es el VCNL4010, debido a sus reducidas dimensiones, bajo consumo y una buena precisión.

Fig. 28 Sensor Iluminación-Proximidad

ILUMINACION	Unidades	min.	Típica	Max
Resolución	Luxes	-	0.25	-
	Bit	16	16	16
Rango	%HR	0.25	-	16383
PROXIMIDAD				
Resolución	mm		Up 200mm	
	Bit	16	16	16

Tabla 4 Características Sensor Iluminacion-Proximidad

4.2.14 Sensor Medidor Temperatura y Humedad relativa

La temperatura es una magnitud que hace referencia a la noción de calor o frío. Esta puede expresarse en el S.I de Unidades por grados Kelvin, aunque también fuera del ámbito científico se hace uso de otras escalas como la escala centígrado o Celsius, y la escala Fahrenheit, En este proyecto se hara uso de la escala Centígrada o Celsius. Para llevar a cabo la conversión la ecuación siguiente muestra cómo realizarla:

$$T(^{o}C) = T(^{o}K) - 273.16$$

La Humedad Relativa está definida como el porcentaje de vapor de agua que contiene el aire, este se mide en porcentajes, así a nivel del mar si nos encontramos con un porcentaje del 90% indica que el aire contiene ese porcentaje del vapor de agua que puede admitir. En zonas secas puede llegar a ser del 20%. Este factor está directamente relacionado con la temperatura y nivel de saturación del aire.

Sensor seleccionado: El sensor seleccionado para este proyecto ha sido el SHT 11, sin embargo debido a problemas con proveedores en plazos de entrega, los cuales superaban los plazos para poder cumplir la planificación se ha utilizado uno similar concretamente el DHT22.

Ambos son capaces de efectuar medidas de los dos parámetros indicados, temperatura y Humedad relativa, estando calibrados, se trata de un sensor de la familia CMOS el cual posee un conversor A/D (analógico/digital) de 14 bit, bajo consumo de energía y pequeñas dimensiones .

En la siguiente tabla se pueden ver sus características principales, tanto para la medición de Humedad Relativa como para la de Temperatura.

Fig 29 Sensor Temperatura Humedad Relativa DHT22

HUMEDAD	Unidades	min.	Tip.	Max
Resolución	%HR	0.5	0.03	0.03
	Bit	8	12	12
Repetitividad	%HR	-	±0.1	-
Rango	%HR	0	-	100
TEMPERATURA				
Resolución	ºC	0.04	0.01	0.01
	ºF	0.07	0.02	0.02
	Bit	12	14	14
Repetitividad	ºC		±0.1	-
	ºF	-	±0.2	-
Rango	ºC	-40	-	123.8
	ºF	-40		254.9

Tabla 5 Características Sensor Temperatura- Humedad Relativa

4.2.15 Sistema Estabilizador Cámara Video (Gimbal)

El sistema estabilizador adquirido ha sido el DJI, ya que el que se había previsto marca Tarot no se ha podido adquirir por problemas de con el proveedor el cual no lo surtirá hasta mes de febrero, este se utilizara de forma provisional ya que su tamaño no el apropiado (es pequeño para situar la Raspberry como se quiere hacer) en el soporte si bien se estudiara la forma en que puede modificarse sin que afecte al sistema estabilizador.

Figura 30 Sistema estabilizador (Gimbal)

4.2.15 Cámara Pi

La cámara capturadora de video se opta por adquirirla correspondiente al sistema Raspberry Pi, para asi poder hacer uso de las ventajas que presenta la CPU de disponer de un bus exclusivo DSI el cual esta conectado directamente al procesador de video obteniendo un mayor rendimiento. Sus características son:

- 5 megapíxeles del sensor,
- Soporta 1080p / 720p / 640x480p Video,
- Capaz de 2592 x 1944 píxeles Imágenes estáticas
- 25mm x 20mm x 9mm Huella
- Peso 3g
- Totalmente compatible con los casos ModMyPi

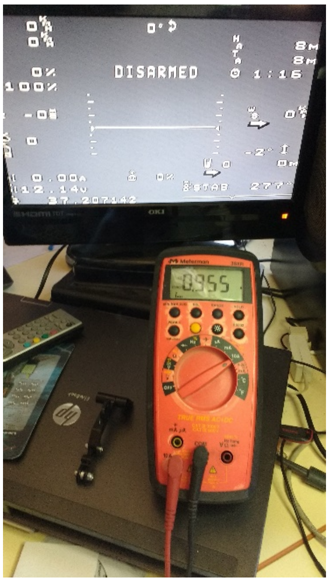
Figura 31 Pi Camera Rev 1.3

4.3 Sistema de Alimentación

5.3.1 Selección de Baterías

La batería a utilizar será de Polímero de litio (Li-Po) ya que estas ofrecen una mayor densidad de carga y tienen un tamaño más reducido así como menor peso que las de tradiciones de ion-litio.

Para la selección de la batería adecuada que proporciones los tiempos de vuelo que se necesitan (20min) se deberá tener él cuenta el consumo total de la unidad.



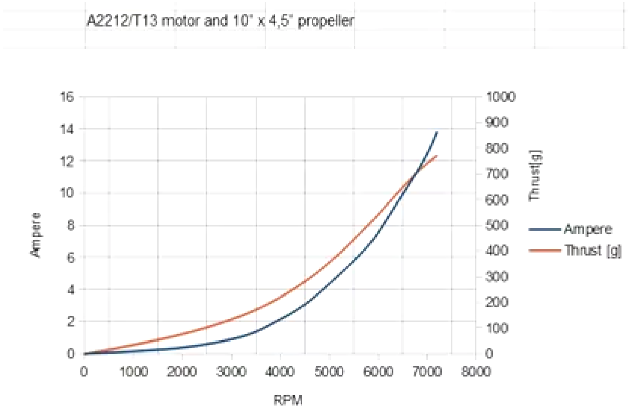
Para ello en primer lugar se toma medida del consumo en estado de reposo efectuando transmisión de video para una vez conocido este poder determinar cuál será el consumo que la unidad tendrá en vuelo incluyendo el que necesitan los distintos sistemas.

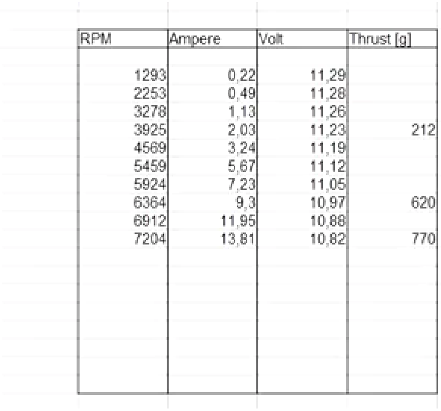
Tomando medida empíricamente pude observarse en la figura de la izquierda que el consumo efectuando transmisión de video es de prácticamente de 1 Amperio. Esta Intensidad sera el consumo de los sistemas de la unidad dron **($I_{Csistemas}$)**

Consumo sistemas y transmisión video= $I_{Csistemas}$ =965mA

Figura 32 Consumo Sistemas

En la siguiente figura se puede ver las características de trabajo del motor, relación consumo-rpm y fuerza de elevación en gramos correspondientes al motor utilizado A2212/T13 (1000Kv)





RPM	Ampere	Volt	Thrust [g]
1293	0,22	11,29	
2253	0,49	11,28	
3278	1,13	11,26	
3925	2,03	11,23	212
4569	3,24	11,19	
5459	5,67	11,12	
5924	7,23	11,05	
6364	9,3	10,97	620
6912	11,95	10,88	
7204	13,81	10,82	770

Figura 33 Relación consumo-rpm y fuerza elevación motores

Tabla 6x Relación RPM Motor y consumo.

Se puede extraer de las características mostradas (tabla xx) que los consumos de un único motor a máxima potencia es de 13,81 A, por lo que el consumo total de los motores a máxima potencia estará determinado por el nº de motores y dicho consumo más el consumo de los sistemas medido anteriormente.

$$I_{max} = I_{\max motor} \cdot Nº\ motores + Ic_{sistemas}$$

Así se tendrá que en el momento de despegue (máxima potencia) el consumo llegara a ser de:

$$I_{max} = I_{\max motor} \cdot Nº\ motores + Ic_{sistemas} = 13,81 \cdot 4 + 1 = 56A$$

Sin embargo los motores no se mantienen durante todo el vuelo a máxima potencia sino que una vez realizado el despegue se establecerá una velocidad de crucero que se estima en 60% (algo superior a la media velocidad). Por lo que el consumo de cada motor seria de aproximadamente 4.5 A , por tanto a dicha velocidad de crucero el consumo seria de:

$$I_{media\ pot} = 4,5A \cdot 4 + 1A = 18 + 1 = 20A$$

La vida útil de una batería se calcula en base a su corriente nominal en miliamperios por hora (mAh) inversamente proporcional a la corriente de carga soportada y aplicándole un factor de tolerancia de un 0.7 por lo que para tener una vida útil de 20minutos se requerirá una batería con un capacidad de:

$$Vida\ util = \frac{Capacidad_{bateria}(mAh)}{I_{c\ de\ carga}} \cdot 0.7 \Rightarrow Capacidad_{bateria}\ (mAh) = \frac{I_{c\ de\ carga} * Vida\ util}{0.7}$$

Por lo que para una autonomía de vuelo de 15 minutos=0.333 horas

$$Capacidad_{bateria}(mAh) = \frac{20000mAh * 0.25h}{0.7} = 7.145mAh$$

Determinada la capacidad de la batería se consulta modelos de baterías seleccionando dos posibles opciones, estas son:

Zyppy modelo Compact
Número de elementos: 3
Voltaje:11,1V
Capacidad: 5000 mAh
Descarga: 25C Continuo 125 A/ 35C Pico 175 A
Carga: 2C
Peso: 360 g
Medidas: 145*51*24mm
Conector de equilibrado: si
Conector batería: 4mm Bullet

Zyppy modelo Flightmax
Número de elementos: 3
Voltaje: 11,1V
Capacidad: 8000 mAh
Descarga: 30C Continuo 240 A/ 40C Pico 320 A
Carga: 2C
Peso: 642 gr
Medidas: 164*67*24mm
Conector de equilibrado: si
Conector batería: 5mm conectores oro.

Ambas puede observarse que la intensidad máxima en pico cubre con amplio margen el amperaje del requerido en la acción de despegue y que en modo de descarga continua igualmente pueden suministrar la intensidad necesaria igualmente con un amplio margen. Por lo que se puede optar por instalar 2 baterías de 5000mAh en paralelo o bien una batería de una capacidad de 8000mAh.

Finalmente se selecciona la batería Li.Po Zyppy modelo Flightmax ya que es suficiente para el tiempo de vuelo indicado y el coste es bastante inferior al que supone la compra de dos baterías del modelo Compact. Por otro lado dado se deja opción a una futura mejora la posibilidad de incorporar una segunda batería en paralelo que pueda proporcionar un tiempo de vuelo, siempre y cuando el peso final de la UAV no exceda de 2,5Kg.

Figura 34 Batería LI-PO Zippy 3S 8000mAh

4.3.2 Adaptación niveles tensión alimentación.

La unidad *Dron* como se ha visto será alimentada mediante batería LiPo de tres celdas, obteniendo así un voltaje de 11.1V. Para la reducción de tensión se hará uso de dos módulos de potencia con medición de intensidad y voltaje preparados para 5V y 3.3V. Concretamente se hará uso de los módulos específicos para Arduino *Arduflyer Power Module V1.0*

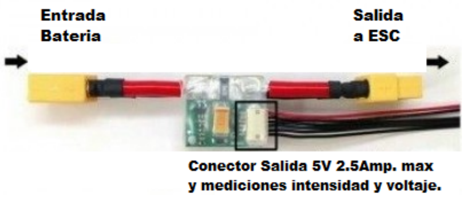

Entrada Bateria
Salida a ESC
Conector Salida 5V 2.5Amp. max y mediciones intensidad y voltaje.

Figura 35 Modulo de potencia V1.0

Este módulo de potencia ofrecerá una alimentación de 5V a partir de una batería LiPo (11.1V) y reportara la medición de consumo de corriente y voltaje de la batería a través del su conector de salida de 6 pines. Sus características son:

- Max Voltaje entrada 25V
- Max medición Amp. 90A
- Salida 5.3V y 3.3V.
- ***Intensidad Max salida 2.25A***

Estas características son suficientes para el consumo máximo que puede requerir tanto en el sistema de control de vuelo (Arduino) como en el sistema central y Transmisión de Video cuyo consumo máximo recomendado es de 2A máximo y verificado en uso empíricamente se ha visto es de 1 Amperio.

4.3.3 Estación base

La estación base deberá estar dotada de un sistema de carga de baterías el cual será alimentado mediante placas solares. El sistema de placas solares y sistema de carga deberá de ser capaz de suministras una intensidad máxima de 8000mA intensidad igual a la capacidad de la batería. Se determina asi ya que en la batería adquirida indica y recomienda sea cargado a un máxima de 1C, por tanto 8000mA.

Los bornes de alimentación para carga serán situados en placas en la zona de aterrizaje donde el Dron se posara sobre ellas y estando dotado de contactos en sus en su tren de aterrizaje servirán de contacto para efectuar la carga una vez haya aterrizado sobre la plataforma.

En la estación de base es deseable que exista un punto de acceso a red si existe la posibilidad.

5. Descripción Detallada del Software

[Ir al índice]

5.1 Sistema Operativo Central

En este último año se han presentado nuevas distribuciones para *Raspberry PI,* especialmente para la versión Rapberry Pi 2 B. La mayoría de los sistemas para este dispositivo están basados en Linux, siendo en gran medida distribuciones de Linux portadas a la arquitectura ARM, pero también soporta otras distribuciones incluso Windows ha sacado un versión para *Raspberry Pi 2 Windows 10 IoT (Internet of Things)* para *Raspberry pi*. Algunos de los sistemas son:

- ***FreeBSD*** el cual es un sistema de Unix-like open-source cuyas características en su versión para *Raspberry Pi* es haber sido compilado utilizando FreeBSD GCC 4.2.1, (*Berkeley Software Distribution*) incluyendo partición de intercambio de 512MB. Sin embargo actualmente no está considerado estable para *Raspberry PI*, presentando problemas con dispositivos de Red.
- ***OpenElec*** (Open Embedded Linux Entertainment Center) es una distribución de Linux la cual está diseñada para HTPC (Home Theater Personal Computer, computadora personal de cine en casa) y está basada en el reproductor de medios XBMC, aplicando el principio de JeOS (Just Enough Operating System) está diseñado para consumir pocos recursos, este proporciona un completo centro de recursos multimedia.
- ***Ubuntu Mate***, con entorno de escritorio, esta versión de Ubuntu se encuentra en fase de desarrollo sin embargo se ha presentado una distribución compilada para ARM. Unbuntu Mate 15.04.

*S*in embargo el primer sistema operativo fue *Raspbian* siendo el más extendido para uso genérico, es una sistema estable para la *Rasberry Pi* y que ha incorporado importantes mejoras para la versión *Rasberry Pi 2.* Por tanto es este proyecto se hará uso de este sistema operativo sobre el que se instalaran el resto de aplicaciones.

Raspbian es una distribución del sistema operativo GNU/Linux lanzado en junio del año 2012, es por tanto un software libre y basado en el sistema operativo *Debian 7.0*. Los motivos para su selección han sido que Raspbian@ dispone de soporte optimizado para cálculos en coma flotante por hardware lo que posibilita un mayor rendimiento en este proyecto. Por otro lado cuenta con herramientas de desarrollo ya incorporadas como los lenguajes de programación Phyton o Scratch, de igual forma dispone de utilidades iniciales simples y mediante menús interactivos con el usuario que permiten la configuración del sistema a usuarios sin necesidad de altos conocimientos en Linux. Así su menú de configuración *raspi-config* tiene opciones como son configuración de teclados, interfaces (I2C, Serial, Cámara, etc..), permite la expansión de la partición *root*, aplicar *overclock*, entre otras muchas.

Puede adquirirse mediante enlaces de descarga, directamente de internet en su web de desarrollo [9] donde también ofrece manuales en línea para su instalación y consulta. De igual forma puede descargarse de la web de *Raspberrypi*[2], donde se encuentra entre otros sistemas operativos disponibles para esta MCU. (http://www.raspberrypi.org/downloads)

Su instalación se realiza de forma sencilla sobre tarjeta micro SD, haciendo uso de las aplicaciones *SDFormatter* (Enlace descarga) y *Windisk32Imagen (*Enlace descarga*).* Ambas aplicaciones de software libre para formateo y escritura de imágenes en Tarjetas SD respectivamente.

Los procesos de instalación tanto del sistema operativo como del resto de aplicaciones de las que se hará uso se pueden consultar su instalación en ANEXO A adjunta a esta memoria, así una vez instaladas las aplicaciones y el Sistema Central configurado como se expone en el mencionado anexo el propio sistema pone disposición lenguajes de programación para la creación de los distintos scripts, y ejecutables en su conjunto bien mediante su uso por usuario o configurados para arranque automático no permitirá realizar todos las funciones de control y seguimiento de la aeronave.

5.2 Lenguaje de programación Python.

El sistema Rabian como se ha indicado incorpora el lenguaje de programación *Python* el cual es un lenguaje interpretado orientado a objetos, y que mediante sentencias permiten realizar tareas que implicarían una mayor complejidad así, en el caso en el caso de querer acceder a dispositivos conectados a la unidad Dron como por ejemplo puede obtener los datos de temperatura o presión de la zona en la que está vigilando se puede fácilmente realizar mediante un breve programación.

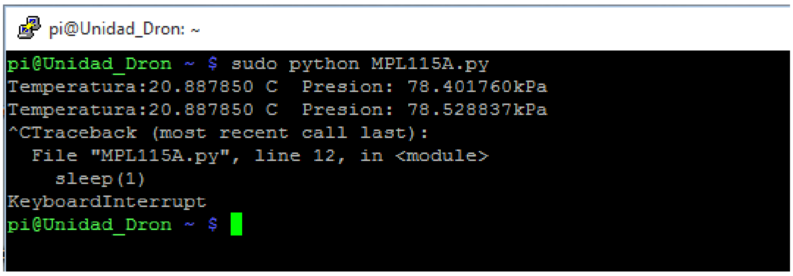
Así para acceder al sensor de Presión y temperatura MPL115A se realiza la siguiente codificación la cual guardaríamos como fichero ejecutable directamente con extensión *.py

```
#Lectura y presentación en pantalla de la presión y temperatura del sensor MPL115A.py
#!/usr/bin/phthon
import smbus
from time import sleep
bus = smbus.SMBus(1)
while(0==0):
    tempC=0
    pressure=0
    # solicita al sensor que efectué el muestreo
    bus.write_i2c_block_data(0x60,  0x12,  [0x01])
    sleep(3)
    #Solicita al sensor que
    bus.write_byte(0x60, 0x00)
    sleep(1)
    reading1=bus.read_i2c_block_data(0x60, 0x00)
    pressure=((reading1[0]<<2)+((reading1[1] & 0xc0) >>6))
    pressure=((65.0/1023.0)*pressure)+50
    tempC=(((reading1[2]<<2+((reading1[3] & 0xc0)>>6)) -510.0)/-5.35 + 25.0)
    if (tempC > 0) : print 'Temperatura:%1f C  Presion:% 1fkPa' %(tempC, pr$
```

Entrando en el sistema ejecutaríamos la sentencia (p.e por conexión SSH) ejecutando directamente el programa mediante *Python* con ejecutando el script mediante.

sudo phyton MPL115A.py

El sistema nos presentara la temperatura y presión de la zona donde se encuentre ubicado el Dron.



```
pi@Unidad_Dron ~ $ sudo python MPL115A.py
Temperatura:20.887850 C  Presion: 78.401760kPa
Temperatura:20.887850 C  Presion: 78.528837kPa
^CTraceback (most recent call last):
  File "MPL115A.py", line 12, in <module>
    sleep(1)
KeyboardInterrupt
pi@Unidad_Dron ~ $
```

Figura 36 presentación lectura Temperatura y presión

Este cuenta con una amplia librería así puede realizarse el servicio de envío de SMS, instala la librería SMS API, lo que permitirá que dichos datos si el sistema no dispone de otra red puedan ser enviados mediante un SMS.

Por otro lado Python permite la ejecución de sus programas en prácticamente toda la plataforma ofreciendo múltiples opciones para desarrollar interfaces graficas de usuario portables, como Tkinter, wxPython, PyGTK, PyQT. Los scripts de Python puede de una forma fácil comunicarse con otras parte de la aplicación del sistema ya que tiene mecanismos de integración para invocar librerías de leguaje C y C++ por lo que puede ser llamado desde C, CORBA (*Common Objetct Request Broker Architecture*) y NET pudiendo interactuar en la red con interfaces como SOAP, o comunicarse por medio de mecanismo de intercambio de datos en XML, lo que lo hace útil en esta aplicación para comunicación con la unidad Dron.

5.2 Biblioteca de visión artificial OpenCV.

Open CV (*Open Source Computer Visión*) es una biblioteca de visión artificial desarrollada por el centro de investigación Intel, es utilizada en aplicaciones de sistemas de seguridad para detección de movimientos, y reconocimiento de objeto.

Esta tiene muchas formas de poder detectar el color por contornos, por momentos, calculando áreas, etc…

Como se ha visto en el apartado 3.3 la detección de fuego por color es más complicada y que la detección por humo si bien

En este sentido no se ha podido profundizar por falta de tiempo material, únicamente se han podido realizar algunas pruebas en cuanto a reconocimiento de objetos por la forma, o reconocimiento facial, pero no ha sido posible disponer de más tiempo para llevar a cabo pruebas de detección del color y comprobar hasta qué punto en vuelo puede ser capaz de detecta color.

Se indica por detección de color pues la detección por humo cuando el sistema de visión, como es la UVA esta en vuelo es difícil implica turbulencias y movimientos aunque exista un estabilizador y como se explicado en el apartado 3 es más complicado y hay que aplicar otras técnicas.

Por tanto todo lo que se ha podido realizar a cabo a través de OpenCV y creando entornos virtuales es conseguir una captura de video alta definición durante el vuelo, a la vez que pone a disposición del servidor WEB local del dispositivo la grabación en almacenamiento USB.

La idea de instalar un servidor en la propia aeronave ha resultado factible en el sentido que si se ha verificado su funcionamiento realizándolo correctamente incluso mediante una captura con Raspistall, como se ha mostrado en videos de verificación de funcionamiento que se encuentra en repertorio del la web del presente trabajo de fin de grado. https://www.infoteli.org.

```
Raspìstill –o /media/archivo/www/video/fotograma.jpg   -tl 1000  t 0,
```

El sistema captura y transmite video por RF (5,8MHz) por tiempo indefinido y envía cada segundo graba un fotograma en el servidor WEB.

Igualmente mediante la grabación de video (Raspivid) se puede efectuar esta grabación de diversos formatos, como se ha efectuado en formato h.264, y realizar video streaming.

Por tanto como el sistema se ha programado mediante las aplicaciones y librerías con el sistema operativo para que pueda efectuar transmisión de video por RF a la vez que proporciona dichos video o bien fotogramas cada segundo para su análisis.

Pruebas que se esperan puedan presentarse en la presentación de este trabajo.

Indicar que todo ello se realiza controlado remotamente como se indica en el siguiente apartado.

5.3 Software *Mission Planner*

5.3.1 Conectando a Unidad Controladora de vuelo a través de Raspberry pi

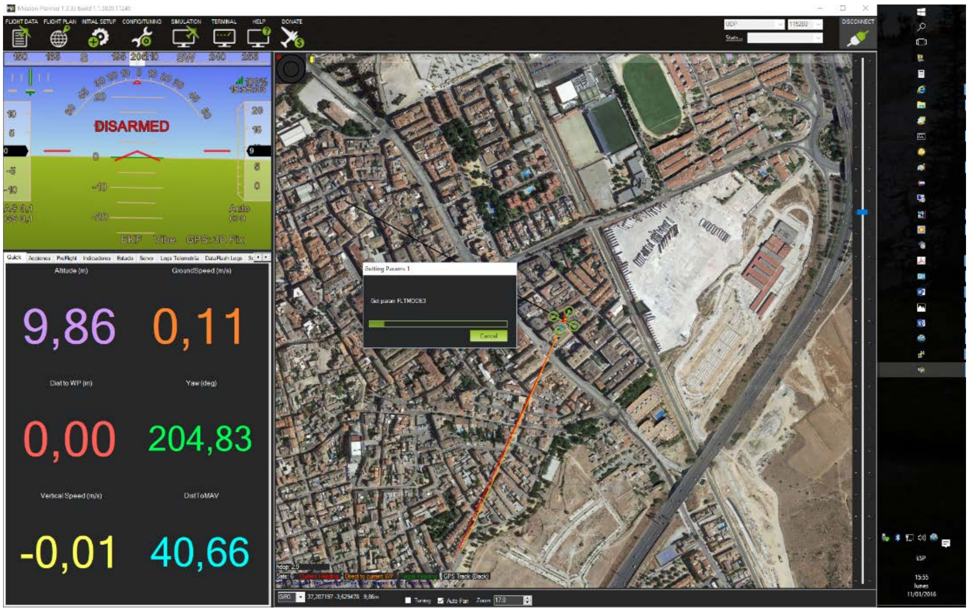
Para ello se hará uso de MAVLink (Micro Air Vehicule link). Este es un protocolo de comunicación para vehículos no tripulados, realizado por Lorenz Meier bajo licencia GPL 2009. Este es instalado como biblioteca en el sistema y permitirá disponer de una conexión y un control total del sistema control de vuelo (Arduino Pilot). Correctamente configurado, mediante *MavProxy* proporciona una conexión mediante protocolo UDP para aplicaciones como *Misión Planner* o bien simplemente mediante conexión SSH (p.e. WiFi) permitirá la comunicación con el controlador de vuelo en línea de comandos a través de MAVLink.

Así en modo supervisor (programador) este **permitirá conectar y comandar los sistemas ajustando parámetros de vuelo y configurando la unidad Dron para vuelo programado.** De igual forma en todo momento haciendo uso de MAVLink se pueden obtener todos los datos de telemetría que puedan requerirse para el control de vuelo, a través de los diferentes dispositivos periféricos de que dispone el sistema controlador de vuelo. (p.e GPS, Altitud, velocidad, posicionamiento, así como sensores tensión y consumo, del sistema en tiempo real.).

Para disponer de conexión remota se debera ejecutar mavproxy debidamente configurado en el inicio del sistema para lo cual agregaremos las siguientes líneas al fichero de inicio *etc/rc.local*

```
(
date
echo $PATH
PATH=$PATH:/bin:/sbin:/usr/bin:/usr/local/bin
export PATH
cd /home/pi
screen -d -m -s /bin/bash mavproxy.py --master=/dev/ttyAMA0 --baudrate 57600 --out 192.168.1.5:14550 --out 192.168.1.5:5760 --aircraft QuadCopter
) > /tmp/rc.log 2>&1
-----------------------------------------------------------------------------------------------------------------------------
```

Con ello se dispondrá de conexión mediante protocolo UDP en el puerto 14550 de forma remota para configuración como puede verse la conexión se establece correctamente pudiendo a partir de ella efectuar toda la programación y control de la UVA a través del sistema central Raspberry pi.



DISARMED

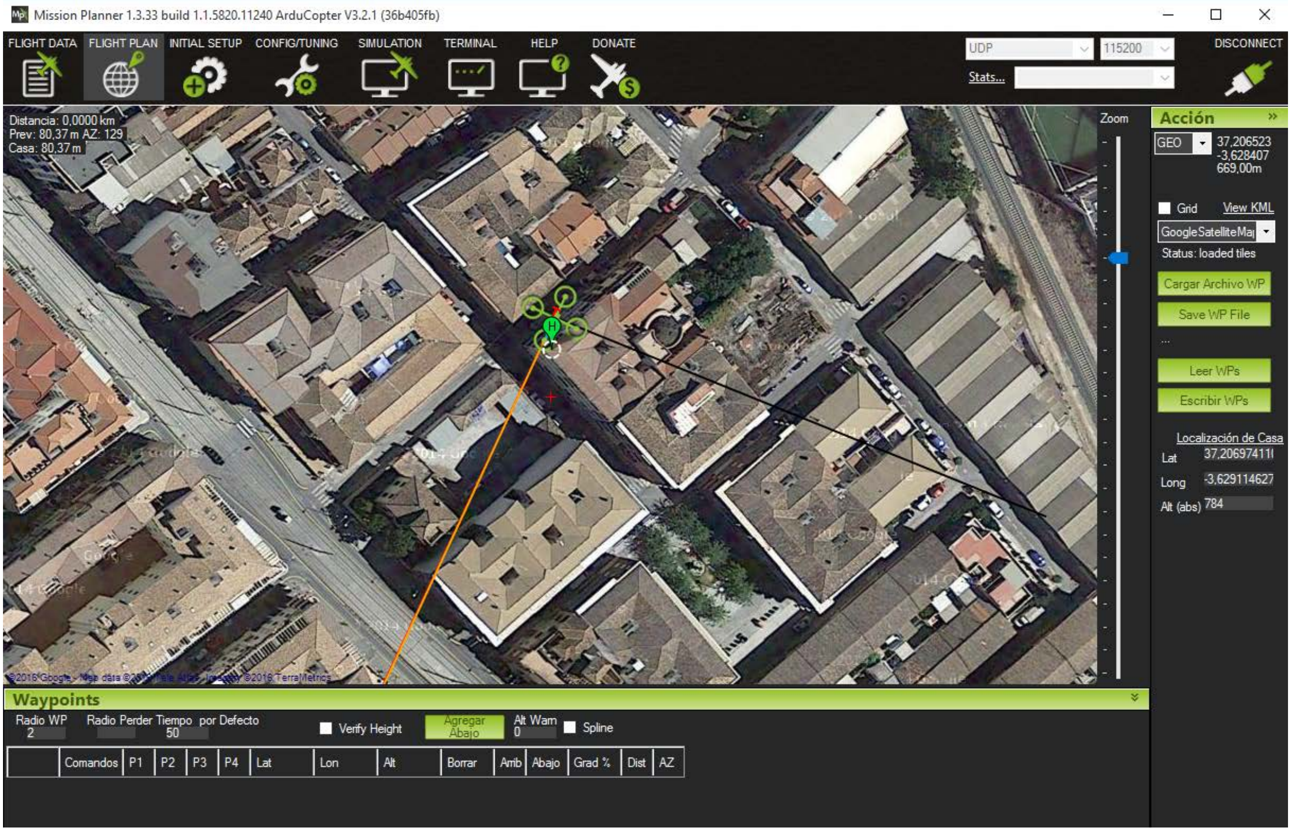
Getting Params 1

Got param FLTMODE3

Altitude (m)	GroundSpeed (m/s)
9,86	0,11
Dist to WP (m)	**Yaw (deg)**
0,00	204,83
Vertical Speed (m/s)	**DistToMAV**
-0,01	40,66

Figura 37 efectuando conexión remota mediante ***Mission Planner***.

En la siguiente figura se puede observar como la situación del GPS la marca correcta y se puede a directamente comenzar indicar los Way-points que definan un plan de vuelo



Mission Planner 1.3.33 build 1.1.5820.11240 ArduCopter V3.2.1 (36b405fb)

UDP 115200

Distancia: 0,0000 km
Prev: 80,37 m AZ: 129
Casa: 80,37 m

GEO 37,206523 -3,628407 669,00m

GoogleSatelliteMap
Status: loaded tiles

Localización de Casa
Lat 37,20697411(
Long -3,629114627
Alt (abs) 784

Radio WP 2 | Radio Perder Tiempo por Defecto 50 | Alt Warn 0

	Comandos	P1	P2	P3	P4	Lat	Lon	Alt	Borrar	Arrib	Abajo	Grad %	Dist	AZ

Figura 38 conexión a programación plan de vuelo.

5.5 Diagrama casos de Uso

En la aplicación del sistema pueden diferenciarse tres tipos de usuarios:

- Usuario Local: Interactúa directamente con el sistema de navegación como piloto in-situ.
- Usuario PC: Interactúa mediante SSH o Entorno Grafico (Escritorio remoto) con el sistema, pudiendo llevar a cabo toda la programación de sistema que se requiera. De igual forma podrá en modo SSH enviar comandos y planes de vuelo mediante MAVLinks por consola a la UAV.
- Usuario WEB: Interactúa con el sistema mediante aplicación WEB a través de red internet.

Para cada usuario podemos distinguir los siguientes casos de uso que se representan el siguiente diagrama de casos de uso

Diagrama casos de Uso

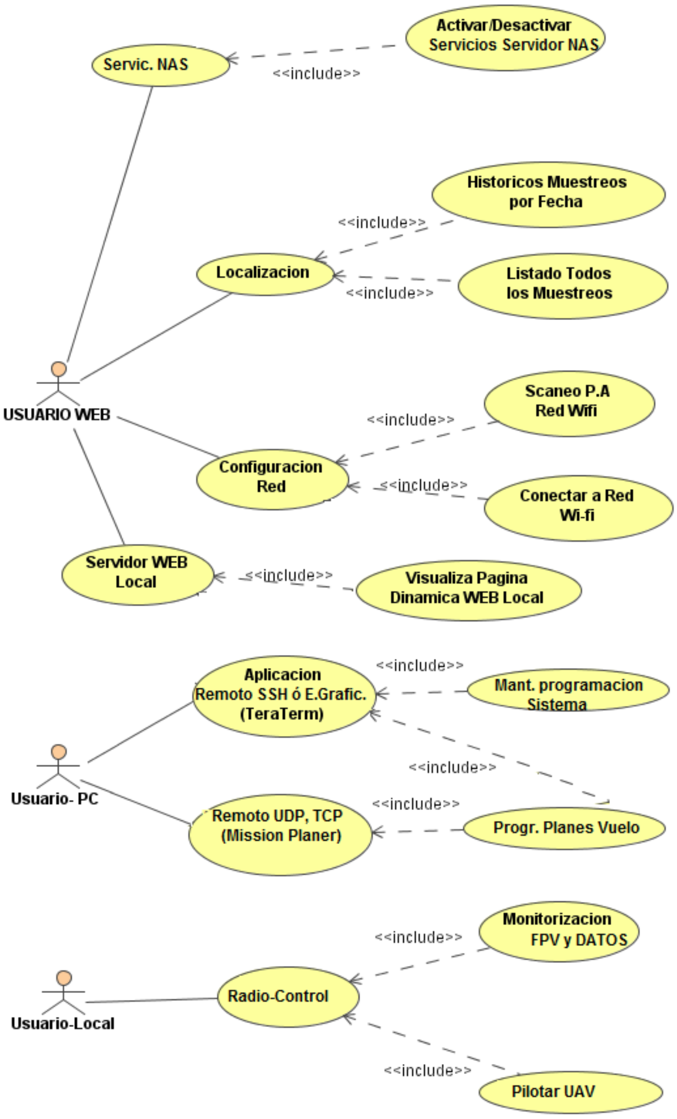


Figura 39 Diagrama casos de uso

6. VIABILIDAD TECNICA

[Ir al índice]

6.1 Estudio Viabilidad Técnica

El proyecto como puede comprobarse está falto de madurez, dado que no se han implementado los diseños de placa de sensores, ni los diseños de caja apropiada para un ensamblado más profesional, de igual forma la programación de interfaces de usuario se han realizado de una forma muy simple a modo de demostración, debido al corto tiempo disponible para su realización. Sin embargo, desde el punto de vista técnico, si puede considerarse un proyecto viable, ya que en la selección de sus componentes se ha tenido presente siempre la consecución de unos fines profesionales, teniendo en cuenta la calidad y precisión de los mismos. Por otro lado resuelve ampliamente el problema planteado, obteniendo un producto con capacidad para la captura de video y medida de fenómenos atmosféricos así, como posibilidad de registro de estos en base de datos para su tratamiento posterior y con posibilidades de mejoras sin prácticamente coste (solo se requiere tiempo) como puede ser la capacidad de tomar decisiones como causa-efecto de los datos recabados o la capacidad de mostrar presentar en su propio servidor web accedo a video a la vez que efectúa el vuelo pudiendo así poner a disposición la visión de la zona y esos datos recabados entiendo que da por tanto solución al problema que se planteaba en este proyecto.

Como puntos débiles, remarcar que el principal punto débil es su falta de madurez, a partir aquí se pueden observar otros como son:

- La autonomía de vuelo, el diseño debería ser repasado y más ajustado en peso para así poder reducir esos 200g que le sobran para poder incorporar una segunda batería lo que le daría una autonomía una mayor autonomía manteniéndose dentro de la legalidad vigente.
- El alto nivel de competencia en ambos sectores que pretende cubrir, ya que existen una amplia gama de productos tanto en el sector de medición de factores meteorológicos como en el sector de UAV unido a la actual legislación vigente hacen que desde el punto de vista comercial no tenga otras expectativas más que en actividades de recreo o deportivas.

Como puntos fuertes se puede considerar los siguientes puntos:

- Uso de una placa base con un procesador central el cual tiene una potencia considerable (cuatro núcleos), una amplia escalabilidad y con una tecnología que puede considerarse de las punteras en sistemas embebidos.
- La implementación de los distintos tipos de intercomunicación con otros dispositivos, no dejando al sistema limitado por el uso exclusivo de ninguno de ellos, esto lo hace fácilmente escalable y aplicable en diversos ámbitos tanto en apoyo a equipos de extinción de incendios como de apoyo a sistemas de toma de datos meteorológicos u otros sectores como puede ser el sector agrario.
- La implementación de distintos interfaces con el usuario, permitiendo el control total del sistema tanto de forma local como remota, y por otro que el sistema pueda informar en todo momento de su estado.
- La alta precisión de los sensores empleados estando todos ellos en un margen de error por debajo del 0.2% lo hacen un dispositivo de alta precisión.

- La realización de toda la programación de sistemas y aplicaciones en su totalidad de código libre reducen su coste en comparación con otros que pudieran dar las mismas prestaciones indicadas en el mercado.

Por lo que se puede concluir que: Implementando en profundidad su aplicación WEB y de control Local para que permita una correcta actuación sobre el sistema, cosa que no se ha podido llevar a cabo por falta de tiempo, se puede considerar que el proyecto es técnicamente viable.

7. COSTES

[Ir al índice]

8.1 Presupuesto Prototipo

PRESUPUESTO INVESTIGACION DESARROLLO Y MONTAJE PROTOTIPO

Descripción Componentes	Cantidad	Precio Unidad(€)	Precio Total(€)
Estructura Quap-Copter TAROT IRON 650	1	160.00	160.00
Placa Base Raspberry PI 2 B	1	45.00	45.00
Ardu Copter APM 2.6 (GPS NEO + MODULO POT V1.0)	1	62.00	62.00
Caja montaje Placa Base Raspberry PI	1	4.00	4.00
Emisora Radio Control Turning 9XR PRO (TX Módulo 16Canales TX + Modulo RX 8) con telemetria + Batería LIPO 11'1V 3S 2200mAh	1	250.00	250.00
Cargador LIPO IMAX B6	1	21.00	21.00
BATERIA LIPO 11'1V 3S 8000mAh	2	55.00	110.00
Modulo Dual Banda GSM-GPRS SIM900A Japon	1	14.75	14.75
Boscam 8CH 5.8G FPV AV RC805 Receiver +200mW TS351	1	60.00	60.00
Mini OSD MAVLink	1	9.00	9.00
Camara PI NoIR	1	18.00	18.00
Camara PI IR	1	14.00	14.00
Soporte Gimbal	1	10.50	10.50
Soporte plegable base alzadora GPS	1	10.00	10.00
Motores A2212/T13 (1000KV) + 4Helices	4	25.00	25.00
Gimbal DJI GO PRO	1	54.00	54.00
Soporte para Gimbal Taror RAIL	4	3.00	12.00
Soporte CNC FPV aluminio	1	6.90	6.90
CP2102	1	4.50'	4.50'
Cables diversos, conectores XT60 (x5Pares)y JST (x5Pares)	1	12.00	12.00
Placas Circuito impreso protoboard Soldable Ariston CEB31	1	13.00	13.00
Rollo Estaño sin plomo 100g	1	8.68	8.68
Sensor TMP 101	1	3.04	3.04
Sensor SMD Barómetro MPL115A	1	2.09	2.09
Sensor SMD Iluminación - Proximidad- Infrarojo VCNL4010	1	2.30	2.30
Sensor Temperatura-Humedad DTH-22	1	6.20	6.20
Total Coste Materiales			932.64 €
Horas Investigación, desarrollo, montaje y programación	480h.	15.00	7.200 €
COSTE BASE			8132,64 €
IVA 21%			1707.70 €
COSTE TOTAL			**9840.34 €**

8. Conclusiones

[Ir al índice]

8.1 Objetivos alcanzados

El desarrollo del proyecto ha resultado una tarea bastante complicada debido al gran número de horas que supone desarrollarlo con detalle y como contrapartida el poco tiempo disponible para la realización.

Hay que indicar que ha resultado un reto muy interesante y me ha permitido aprender bastante sobre el tema. La dificultad del tema escogido, y la falta de tiempo debida principalmente a compaginar el proyecto con otras tareas y estudios, no me han permitido llevar el proyecto a puntos de realización que me hubiera gustado vistas las grandes posibilidades que ofrece los dispositivos de que se han podido disponer.

Indicar que básicamente estoy satisfecho con el producto final obtenido el cual ha presentado un funcionamiento correcto consiguiendo alcanzar los objetivos marcados, si bien hay que reconocer que falta puntos importantes para alcanzar cierto nivel de madurez como diseño de placa, diseño de caja exterior, una programación de interfaces más exhaustiva que ponga al alcance del usuario a todas las posibilidades del sistema etc... y realizar algunas mejoras de las cuales se proponen a continuación.

8.2 Propuesta de mejoras

A lo largo del desarrollo del proyecto y en la presente memoria se han indicado varios puntos como mejoras del sistema tanto en software como en hardware, así se propondrían las siguientes mejoras:

<u>Software</u>:

- Programación adecuada y eficaz del modo bajo consumo (no efectuado por falta de tiempo)
- Programación de una Interfaz Usuario-sistema WEB mucho más detallada y avanzada.

<u>Hardware</u>:

- Inclusión de un 2º módulo GPS al sistema pudiendo dejar así el dispositivo *Bluetooth* para comunicaciones con dispositivos domótica cercanos, y no dependiendo el sistema de GPS externo.
- Inclusión de zócalo tarjeta *microSD a Sistema RED SENSORES*, lo que podría dotar al sistema de una total autonomía para registro de datos y poder realizar un servidor WEB Local más avanzado.
- Dotar al sistema de conexión 3G dando una total independencia al mismo.
- -Eliminación Modulo CP2102, ya que el sistema dispone de USB propio.

Estas son mejoras que básicamente en Hardware no incrementan el costo de un forma excesiva, ya que el dispositivo GPS interno es más económico que un GPS externo por *Bluetooth* (se ha efectuado así porque se disponía del externo), se ha consultado un módulo que incluye GPS y 3G para Arduino el cual su costo es de 105€. Por otro lado se ahorraría costo al eliminar el dispositivo CP2102 utilizando el USB incorporado en la MCU, y los costos de zócalo tarjeta microSD son mínimos (aprox. 10€).

8.3 Impacto ambiental

En el presente proyecto se ha tenido en cuenta a la hora de desarrollarlo el consumo energético. Las aeronaves son más eficientes y su nivel de contaminación es prácticamente nulo. Las bases allí donde se establezcan se alimentaran con placas solares por lo que el impacto en el medio ambiente es mínimo.

En cuanto al proyecto las plataformas de hardware utilizadas se ha cuidado que cumplan con la normativa RoHS la cual restringe el uso de materiales peligrosos como son mercurio, plomo, cadmio, cromo VI, etc.. Si hay que guardar las precauciones oportunas con las baterías así las baterías Li-Po cuando se acaba su ciclo de vida deben ser sumergidas en agua con sal y trasladas a los puntos de recogida de las baterías.

8.4 Autoevaluación.

Considerando que los requisitos mínimos básicamente consistían en:

- En el desarrollo mediante sistemas de Arduino, adquisición de datos mediante sensores
- Diseño e implementación de una UVA Programación de interfaces de comunicación
- Muestreo periódico de datos y envío a la aplicación local
- Conexión a Internet mediante Wifi

Creo que se han cumplido los objetivos y se han incorporado bastantes de las mejoras que se dieron como optativas en la planificación inicial. Intentando suplir las carencias que por falta de tiempo material no se ha podido llevar a cabo, asi se ha incorporado a la unidad un servidor WEB Local, sobre pendrive USB para no acortar la vida de la tarjeta micro SD, con capacidad de debidamente configurado presentar el video de vuelo en el momento que lo está realizando independientemente de que efectué la grabación para el posterior análisis por visión de imágenes.

Así este TFG me ha brindado la oportunidad de hacer una breve introducción en el mundo de los sistemas embebidos y así como de así como de los diversos sistemas de transmisión y control que pueden ser empleados, se han utilizado diversos sistemas de transmisiones en conjunción así como las diversas redes de datos GPRS, WAN, LAN, incluyendo en la unidad las red GSM, con la finalidad de conocer dentro de lo posible el funcionamiento de dichos dispositivos de comunicaciones.

De igual forma me ha permitido conocer y utilizar un sistema operativo como es Raspbian y de igual forma hacer una introducción en el lenguaje de programación Python. Si bein es cierto que por problemas ocurridos y falta de tiempo no se ha podido profundizar en la aplicación de Open CV, cosa que me hubiera gustado y que realizare para continuar con esas esas mejoras en este proyecto.

Por ultimo también me ha dado la oportunidad para conocer por encima las comunicaciones con dispositivos GPS, conociendo los distintos tipos de datos que pueden recabarse de la red de satélites mediante los distintos protocolos más en profundidad el protocolo utilizado NMEA0183 v2.20 (National Marine Electronics Association).

9. Glosario

[Ir al índice]

A

ADC: Conversión analógica digital.

B

Bluetooth: Es una especificación industrial para Redes Inalámbricas de Área Personal (WPAN) que posibilita la transmisión de voz y datos entre diferentes dispositivos.

Baudrate: Número de unidades de señal por segundo.

C

Cortex-A7: procesador de 32 bits de ARM.

CPU: Unidad central de procesamiento.

D

DHCP: protocolo de configuración dinámica de host.

Diagramas de Gantt: Herramienta grafica para representación de planificación de trabajos y tareas.

E

E/S: Entrada/Salida.

G

GUI: Interface gráfica de usuario.

GPIO: Puerta programable de entrada y salida de datos.

H

HTML: Siglas de HyperText Markup Language.

I

IDE: Entorno de desarrollo integrado.

IP: Identificación lógica y jerárquica, de interfaz de dispositivo dentro de una red que utilice el protocolo IP.

I2C: Bus de comunicaciones en serie.

L

Lan: Red de área local.

M

MCU: Microcontrolador circuito integrado programable, con capacidad de ejecución y memoria.

Mysql: Sistema de gestión de bases de datos relacional, y multiusuario

O

Open Source: Es el término con el que se conoce al software distribuido y desarrollado libremente.

P

PHP: Lenguaje de programación de uso general de script del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico.

R

RSSI: Indicador de fuerza de señal de recepción.

RTLinux: Sistema operativo en tiempo real el cual ejecuta Linux como un hilo.

S

SSID: Nombre identificación de una Red Inalambrica (Wi-Fi)

SPI :Serial Peripheral Interface. Estándar de comunicación utilizado para entre transferencia de información entre dispositivos

T

TCP/IP: conjunto de protocolos de red en los que se basa Internet y que permiten la transmisión de datos entre computadoras.

U

UART: Transmisor-Receptor Asíncrono Universal

W

Wifi: Mecanismo de conexión de dispositivos electrónicos de forma inalámbrica.

Wireless: Comunicación inalámbrica o sin cables

10. Bibliografia

[Ir al índice]

Se indican a continuación otras referencias utilizadas para el estudio inicial del proyecto y la realización que se han efectuado hasta la entrega de esta PAC3

- (Material Docente UOC) Universidad Oberta d Catalunya. Sistemas Empotrados, 2011. José María Gomez Cama, Francisco Hernández Ramírez, José López Vicario, Antoni Morell Pérez, Juan Daniel Prades García, Ignasi Vilajosana Guillen, Xavier Filajosana Guillen.

- (Material Docente UOC) Universidad Oberta de Catalunya. Presentación de documentos y elaboración de presentaciones. 2013 Roser Beneito Montagut

- Pérez García, Miguel Ángel . 2011 . Instrumentación Electrónica. Editorial Thomson-Paraninfo

- LOS SANTOS, 2004. Alberto Los Santos Aransay, Aplicación de las redes de sensores en el entorno vehicular. Mayo 2009.

  http://www.albertolsa.com/wp-content/uploads/2010/04/rsi-aplicacion-de-las-redes-de-sensores-en-el-entorno-vehicular-alberto-los-santos.pdf

- LORENZATI, 2010. Ing. Marcelo Lorenzati, Desarrollo de drivers y aplicaciones para FreeRTOS. Marzo 2010.
  http://laboratorios.fi.uba.ar/lse/sase/2010/slides/SASE-2010_-FreeRTOS_Drivers_-Lorenzati.pdf

- R**eal Decreto-ley 8/2014 Agencia estatal seguridad**. Legislación vigente sobre *Drones*. Boletin Oficial del Estado
  Enlace: *www.seguridadaerea.gob.es/media/4243006/rdl_8_2014_4julio.pdf*

- ***Aplicación WEB del proyecto (2016)****. Miguel Angel Iglesias Jimenez.*
  Enlace: https://www.infoteli.org/proto

11. Referencias

[Ir al índice]

[1] **Plan INFOCA**. Sistemas de Vigilancia y detección Red de comunicaciones. Junta Andalucía [en línea] Enlace documento . Última consulta: 08/10/2015

[2] **Plan INFOCA**. Aplicación de nuevas tecnologías. Junta de Andalucía. [en línea] Enlace documento . Última consulta: 08/10/2015

[3] **FlichtechSystem**. Proyectos y aplicaciones UAV empresa Española. [en línea] Enlace: *http://flightechspanish.weebly.com/aplicaciones.html* . Última consulta 08/10/2015

[4] **Zerinta Tecnologies**. Smart Drones. [en línea] Enlace: http://www.zerintia.com/productos-drones.html. Última consulta 08/10/2015

[5] **Wintergreen Research**. Smart Commercial Drones: Market Shares, Market Strategies, and Market Forecasts, 2015 to 2021 [en línea] Enlace: http://www.wintergreenresearch.com/commercial_drones última consulta 25/10/2015

[6] **AESA. Agencia estatal de seguridad Aerea (2014)**. El uso de *Drones* en España. [En línea] Enlace a documento . Ultima consulta: 25/10/2015

[7] R**eal Decreto-ley 8/2014 Agencia estatal seguridad**. Legislación vigente sobre *Drones*. Boletin Oficial del Estado Enlace: *www.seguridadaerea.gob.es/media/4243006/rdl_8_2014_4julio.pdf* Última consulta: 25/10/2015

[8] **Rasberry Pi – Teeach, Learn, and Make with RAspberry Pi**. [En línea] Enlace: https://www.raspberrypi.org/community/ Ultima consulta: 08/01/2016

[9] **I Gonzalez, S Salazar, and H Romero**, "Attitude control of a quad-rotor using speed sensing in brushless DC motors," in *2011 8th International Conference on Electrical Engineering Computing Science and Automatic Control (CCE)*, Merida City, 2011, pp. 1-6. Enlace documento. Ultima consulta: 30/12/2025

[10] **APC Propellers**. (2008) Materials Research. [Online]. Enlace: http://www.apcprop.com/v/Research/research.html#materials Última consulta: 30/12/2025

[11] **Gigahertz**. 2009-15 Juan Carlos lopez. ***La Jaula de Faraday*.** [En línea]

Enlace: http://www.gigahertz.es/la_jaula_de_faraday.html última consulta 25/10/2015

[12] Video prueba empírica motores utilizados [En línea]

URL: https://www.youtube.com/watch?v=D-eBnXrWJYw Última consulta 18/11/2015

[13] **Ardupilot.** Manuales consulta Arducopter. Instalación y manuales [En línea] http://copter.ardupilot.com/wiki/flying-arducopter/ Última consulta: 30/12/2025

[14] **Referencias a Diseños Sistemas Embebidos y Redes de sensores**. Estación meteorológica móvil con posicionamiento GPS. REPOSITORIO 02 UOC. URL: http://hdl.handle.net/10609/29781 Última consulta: 10/01/2016

Anexo A

[Ir al índice]

MANUAL USUARIO

1 Instalación

1.1 Creación de la tarjeta Micro SD Sistema Central

Para ello se deberá de disponer de una tarjeta micro SD con una capacidad mínima de 8GB. (Recomendada 32GB) y un PC. Mediante el PC se efectúa la grabación de la imagen del sistema Raspbian@, haciendo uso de las cualquiera de las aplicaciones existentes de formateado y grabación de imágenes en tarjetas USB. En este caso hemos utilizado SDFormatter para formateo de la micro tarjeta y Win32DiskImager para su grabación.

Efectuada la grabación de la Micro SD, se insertara en la placa Raspberry pi 2 B, a la que en su primer inicio se deberá conectar para realizar su configuración básica los siguientes dispositivos:

- Un monitor (HDMI),
- Un teclado y ratón (inalámbrico por puerto USB)
- Tarjeta USB WIFI (Edimax) o bien conectar a router mediante cable Ethernet

1.2 Primeras configuraciones del sistema

En el primer arranque el sistema se presenta mediante menú grafico como puede verse en la siguiente captura:

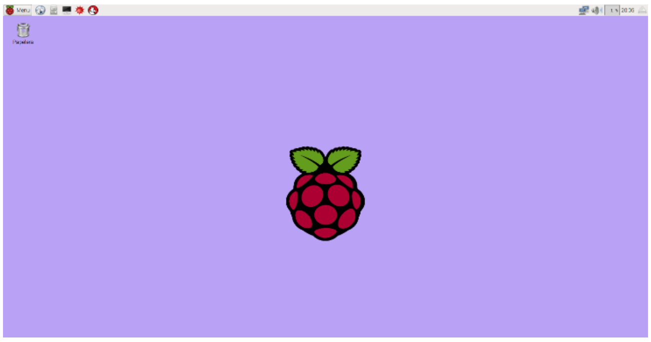
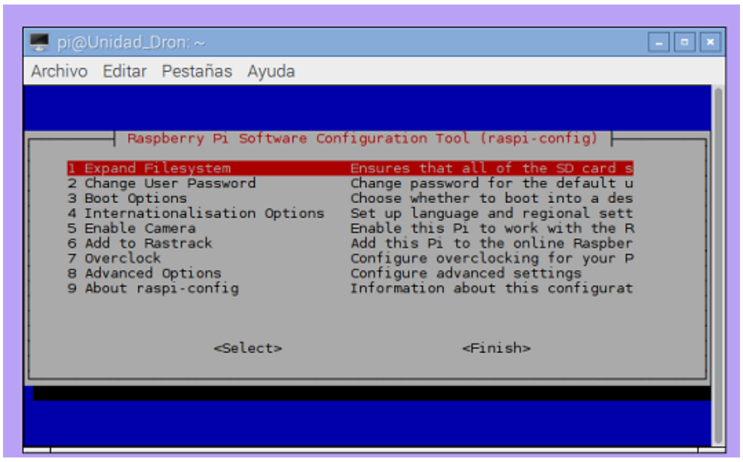


Fig xx Primer inicio Sistema Raspbian modo grafico

A continuación se inicia un terminal de consola en el cual se ejecuta el comando **sudo *raspi-config*** para la configuración básica del sistema el cual mediante su menú interactivo permitirá llevar a cabo la configuración básica del sistema como puede verse en la siguiente imagen



```
         Raspberry Pi Software Configuration Tool (raspi-config)

   1 Expand Filesystem               Ensures that all of the SD card s
   2 Change User Password            Change password for the default u
   3 Boot Options                    Choose whether to boot into a des
   4 Internationalisation Options    Set up language and regional sett
   5 Enable Camera                   Enable this Pi to work with the R
   6 Add to Rastrack                 Add this Pi to the online Raspber
   7 Overclock                       Configure overclocking for your P
   8 Advanced Options                Configure advanced settings
   9 About raspi-config              Information about this configurat


                   <Select>                     <Finish>
```

En este punto se realiza la siguiente configuración mínima:

- Internacionalización de la unidad: Asignar idioma, *timezone* y tipo de teclado del sistema.
- Activación del módulo de cámara
- En opciones avanzadas: Se activa el bus I2c, el acceso mediante SSH, el modo de audio (seleccionado 3.5mm headphone Jack) y se desactiva el acceso al puerto serial mediante Shell

Esta aplicación de configuración como puede observarse en la figura anterior permite muchas otras opciones como cambio de contraseña, definir el nombre del host ó expandir el sistema de archivos para que pueda utilizar la totalidad de la tarjeta micro SD entre otras.

1.3 Configuración salida Video

Efectuadas estas configuraciones mínimas, debido a que la señal de video será transmitida por RF (5.8Ghz) utilizaremos por tanto la salida de video compuesto, para ello se debe editar el fichero de configuración inicial /boot/config.txt. e indicar la opción de salida de video compuesto deseada. Esto se puede llevar a cabo mediante el editor de texto del sistema (aplicación *nano*). Así ejecutando la aplicación con privilegios root (sudo):

```
sudo nano /boot/config.txt
```

Se edita el fichero config.txt y se modifica el parámetro *stdv_mode* (el cual se encuentra en una línea comentada), eliminando el signo de línea comentada (#) se dejara asignado el modo de video AV en este caso modo PAL, asignándole el valor:

```
stdv_mode=2
```

Este tiene las opciones (0=NTSC normal, 1=NTSC Japon, 2=PAL Normal, 3 PAL Brasil 525/60)

Con estos pasos iniciales queda instalado el sistema Raspbian@ quedando configurado para su monitorización a través de su salida de video compuesto (para conexión al transmisor video RF 5.8Ghz) y podrá ser controlado mediante SSH.

1.4 Configuración redes LAN

Para llevar a cabo la configuración de los dispositivos de red (LAN), tanto la red lan cableada como la inalámbrica que se instala mediante adaptador USB Edimax, debe efectuarse la configuración mediante la edición del fichero */etc/network/interfaces*, asignando en este IP fijas y los distintos datos de conexión a la red inalámbrica como son SSID y contraseña.

A continuación se muestra la configuración utilizada en dicho fichero y que debe modificarse acorde a los datos de la red a la que efectué conexión. (Se requerirá de un teclado y monitor pudiendo utilizar directamente la salida HDMI la cual si esta es conectada en el encendido del sistema esta será activada por defecto).

El fichero *interfaces* utilizado es configurado como se muestra a continuación:

--Fichero /etc/network/interfaces --

```
# Please note that this file is written to be used with dhcpcd.
# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'.
auto lo
iface lo inet loopback
iface eth0 inet static
address 192.168.1.41
netmask 255.255.255.0
gateway 192.168.1.1
auto wlan0
iface wlan0 inet static
address 192.168.1.40
netmask 255.255.255.0
gateway 192.168.1.1
wpa-ssid "xxxxxxxxx"
wpa-psk "xxxxxxxxx"
```

***Importante si el adaptador se apagara puede configurarse el modo suspensión, modificando o creando el fichero */etc/modprobe.d/8192cu.conf* ajustando los valores:

- *rtw_power_mgnt*: Activa y desactiva el control de energía del adaptador wifi. Los posibles valores son 0, (desactivado), 1 (activado ahorro de energía) y 2 (máximo ahorro de energía).
- *rtw_enusbss:* Controla la auto suspensión del adaptador en caso de no ser usado. Los posibles valores son 0 (desactivado) y 1(activado).
- *rtw_ips_mode*: controla el consumo energético del adaptador cuando no se encuentra en uso. Los posibles valores son 0 (normal, por defecto) y 1 (máximo ahorro de energía).

1.5 Configuración auto montaje unidades almacenamiento externas mediante USB

Dada la capacidad limitada de la tarjeta SD y que este proyecto requerirá de espacio de almacenamiento para videos y otra información, instalamos una unidad de almacenamiento USB, para lo que con la finalidad de no tener que efectuar montaje manual de la misma se instala la herramienta *usbmount* para lo cual ejecutamos el comando

```
sudo apt-get install usbmount
```

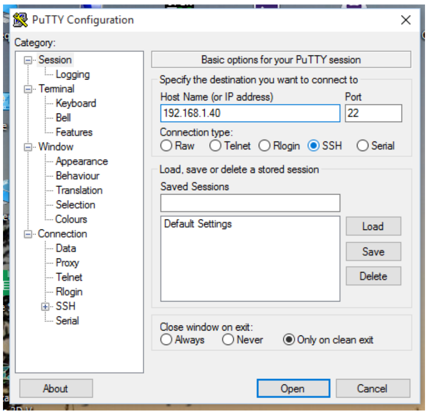
A partir de su instalación al conectar un dispositivo de almacenamiento en el USB este será montado por el sistema de forma automática, asignando el nombre USBx (x =0,1,.2…x)

** Si no se desea instalar la aplicación puede montarse una unidad USB de forma fija editando el fichero /etc/ftsab al que se añadirá la siguiente línea:

```
/dev/sda1    /media/archivo       vfat      defaults    0    0
```

Queda así montado en el arranque el dispositivo de almacenamiento USB al directorio /media/archivo.

Con estos pasos iniciales queda instalado y configurado el sistema Raspbian para su monitorización a través de su salida de video compuesto y podrá ser controlado mediante SSH como se muestra en las siguientes capturas del sistema. Como puede verse en la siguiente figura (izquierda), se puede efectuar conexión directamente mediante aplicaciones de conexión remota SSL (p.e PuTTy) indicando la dirección IP asignada al sistema.



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 192.168.1.40   Port: 22

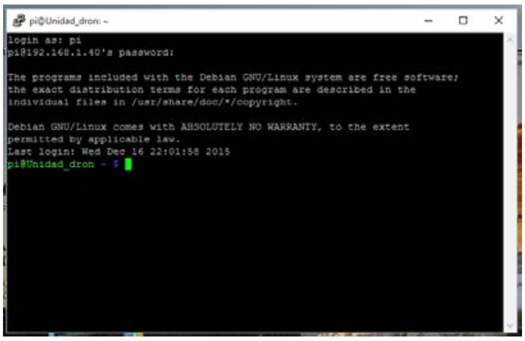
Connection type:
○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit:
○ Always ○ Never ● Only on clean exit



```
login as: pi
pi@192.168.1.40's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Dec 16 22:01:58 2015
pi@Unidad_dron ~ $
```

Fig 25. Conexión SSL sistema central.

Haciendo clic en Open se obtendrá una conexión segura con el sistema central. Este solicitara usuario y contraseña permitiendo así el acceso al sistema con todos los privilegios del usuario utilizado, como puede observarse en la figura anterior (derecha).

Igualmente puede interactuar mediante entorno activando las X, como puede verse en la siguiente figura mediante simple conexión de escritorio remota SSH. (Utilizando Xarchive del propio sistema Raspbian).

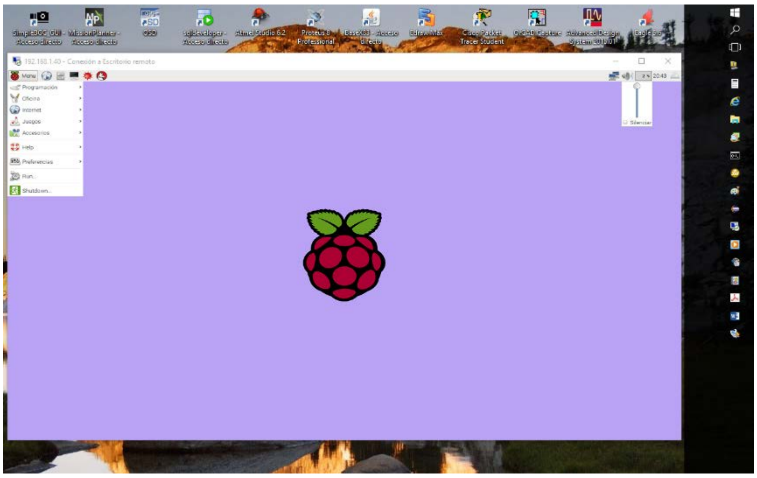


Fig 26. Conexión SSL entorno grafico (escritorio remoto).

Quedando así el sistema totalmente configurado para continuar con su programación y control de forma remota sin que se requiera de teclados ni monitores.

Una vez instalado el sistema *Raspbian* es conveniente actualizar el sistema y firmware de la Raspberry Pi para lo que se ejecutaran los siguientes comandos:

```
sudo apt-get update
sudo apt-get upgrade
sudo rpi-update
```

1.6 Instalación OpenCV y Phyton

Una vez instalado el sistema *Raspbian* se realiza la instalación del Software OpenCV y Python para lo cual se ejecutan las siguientes líneas de comando, siguiendo los siguientes pasos:

1- Se instalan paquetes de herramientas

```
sudo apt-get install build-essential cmake pkg-config
```

2- Se instalan paquetes necesarios para los diversos formatos de imagen (jpg, png, tiff, etc..)

```
sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
```

3- Se instala librería GTK developement, utilizada para interfaces graficas, posibilitando que OpenCV

```
sudo apt-get install libgtk2.0-dev
```

4- Instalación de los paquetes necesarios para I/O video

```
sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
sudo apt-get install libxvidcore-dev libx264-dev
```

5- Instalación librerías para optimización de operaciones con Open CV

```
sudo apt-get install libatlas-base-dev gfortran
```

3- Instalar PIP

```
wget https://bootstrap.pypa.io/get-pip.py
sudo python get-pip.py
```

7- Instalación *virtualenv* y *virtualenvwrapper*

```
sudo pip install virtualenv virtualenvwrapper
sudo rm -rf ~/.cache/pip
```

7- Actualizar el perfil para lo que se incluyen las siguientes líneas en el fichero ~/.profile

```
Sudo nano ~/.profile
export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python2.7
export WORKON_HOME=$HOME/.virtualenvs
source /usr/local/bin/virtualenvwrapper.sh
```

Cargando el nuevo perfil y creación entorno vircutal (cv)

```
source ~/.profile
mkvirtualenv cv
```

8- Instalación de *Python 2.7 development tools*

```
sudo apt-get install python2.7-dev
```

9- Instalar NumPy desde enlaces de OpenCV Phyton

```
Pip install numpy
```

10- Descarga de Open CV e instalación

```
cd ~
git clone https://github.com/Itseez/opencv.git
cd opencv

git checkout 3.0.0
cd ~
git clone https://github.com/Itseez/opencv_contrib.git
cd opencv_contrib

git checkout 3.0.0
```

Configuración y compilación

```
workon cv

mkdir build

cd build

cmake -D CMAKE_BUILD_TYPE=RELEASE

     -D CMAKE_INSTALL_PREFIX=/usr/local

    -D INSTALL_C_EXAMPLES=ON

    -D INSTALL_PYTHON_EXAMPLES=ON

    -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules

    -D BUILD_EXAMPLES=ON ..

sudo make –j4
```

Ejecución programa instalación

```
sudo make install

sudo ldconfig
```

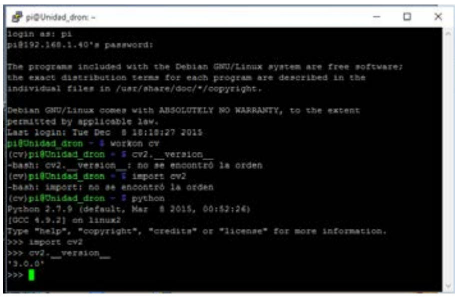
Por ultimo Open CV queda instalado en el directorio /usr/local/lib/python2.7/site-packages. Se efectúa copia de la compilación al directorio cv del entorno virtual para su uso

```
cd ~/.virtualenvs/cv/lib/python2.7/site-packages/

ln -s /usr/local/lib/python2.7/site-packages/cv2.so cv2.so

ln -s /usr/local/lib/python2.7/site-packages/cv.py cv.py
```

Reiniciando el sistema se efectua conexión mediante SSH se verifica OpenCV 3.0 y Phyton 2.7 está correctamente instalado como puede verse en la figura 27. En ambos entornos puede llevarse a cabo las mismas tareas, el entorno X enfocado a programación usuarios, y entorno SSH (TTY) para usuarios avanzados.



```
login as: pi
pi@192.168.1.40's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Dec  8 18:18:27 2015
pi@Unidad_dron ~ $ workon cv
(cv)pi@Unidad_dron ~ $ cv2.__version__
-bash: cv2.__version__: no se encontró la orden
(cv)pi@Unidad_dron ~ $ import cv2
-bash: import: no se encontró la orden
(cv)pi@Unidad_dron ~ $ python
Python 2.7.9 (default, Mar  8 2015, 00:52:26)
[GCC 4.9.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'3.0.0'
>>>
```

Figura 27. Verificación instalación *OpenCV 3.0 entorno SSH. (TTY)*

1.7 Instalación MAVlink (Micro Air Vehicule Communiction Protocol)

Se efectúa la instalación del protocolo *MAVlink*, así como de la aplicación *MAVproxy*, lo cual permitirá el acceso a la unidad de control de vuelo (APM 2.6) a través de la unidad central (*Raspberry pi 2B*) pudiendo así efectuar programación de planes de vuelo y operaciones de mantenimiento y ajustes del controlador de vuelo mediante software libre *Mission Planner* de forma remota.

Para ello ejecutamos los siguientes comandos:

```
sudo apt-get install screen python-wxgtk2.8 python-matplotlib python-opencv python-pip
                    python-numpy python-dev libxml2-dev libxslt-dev
sudo pip install pymavlink
sudo pip install mavproxy
```

**** Importante desactivar el acceso a la interfaz serial mediante el Shell, se realiza mediante el comando *raspi-config* en *opciones avanzadas* (como se ha indicado en punto 1 de este manual). De no realizarse el sistema enviara error por uso de la UART0 por diversos dispositivos.**

Para que MAVproxy se active en el inicio del sistema deberán agregarse las siguientes líneas al fichero del sistema */etc/rc.local,* lo cual se puede realizar mediante el propio editor del sistema *nano*

```
------------------------------------------------------------------------------------------------------------------------------
(
Date
echo $PATH
PATH=$PATH:/bin:/sbin:/usr/bin:/usr/local/bin
export PATH
cd /home/pi
screen -d -m -s /bin/bash mavproxy.py --master=/dev/ttyAMA0 --baudrate 57600 --out 192.168.1.5:14550  --
aircraft QuadCopter
) > /tmp/rc.log 2>&1
------------------------------------------------------------------------------------------------------------------------------
```

Donde en los parámetros a indicar a mavproxy son:

- **–master=/dev/ttyAMA0** el cual determina el puerto serie para la comunicación. En el caso de la Raspberry Pi, el puerto serie es 0
- **–baudrate 57600** velocidad del bus que debe coincidir con el predeterminado del controlado de vuelo
- **-out IP:Puerto (p.e. 192.168.1.5:14550):** IP del equipo donde se ejecutara la aplicación *Mission Planner*. Y puerto UDP que utilizara para la comunicación. (14550 es el utilizado por defecto por la aplicación *Mission Planner*).
- **–aircraft Planes_vuelo** nombre del modelo que pretendemos usar en la sesión y con el que se guardarán los nombres de los logs y configuraciones.

1.8 Configuración e instalación herramientas bus I2c

Para el uso de los dispositivos sensores conectados al bus I2c, se deberán instalar las correspondientes herramientas en el sistema para lo cual ejecutaremos los siguientes comandos:

```
sudo apt-get install i2c-tools
sudo apt-get update
```

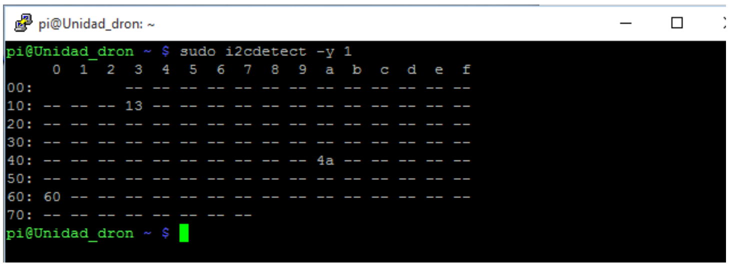
Para asegurar que el usuario pi es miembro del grupo i2c, ejecutaremos el comando:

```
sudo adduser pi i2c
```

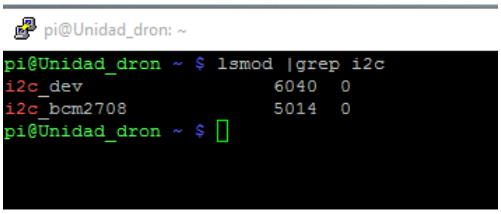
Una vez instaladas las herramientas se reiniciara el sistema para verificar si se obtiene un funcionamiento correcto y los dispositivos sensores son detectados por el sistema central. Para ello se puede verificar si se han cargado correctamente las librerías y comprobar que los dispositivos sensores son detectados por el sistema mediante los comandos:

```
lsmod |grep i2c
sudo i2cdetect – y 1
```

Se puede observar en la siguiente figura el resultado que ha de obtenerse si la controladora bcm2708 y el modulo i2c_dev se han instalado correctamente y que los dispositivos han sido detectados y se encuentra conectados al bus I2C.



```
pi@Unidad_dron ~ $ sudo i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- 13 -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- 4a -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: 60 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
pi@Unidad_dron ~ $
```



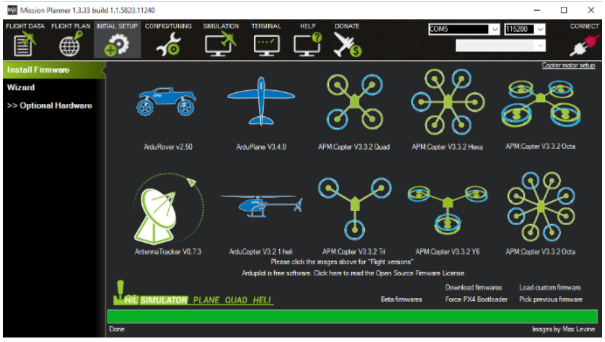
```
pi@Unidad_dron ~ $ lsmod |grep i2c
i2c_dev                 6040  0
i2c_bcm2708             5014  0
pi@Unidad_dron ~ $
```

Figura 27. Verificación reconocimiento dispositivos sensores

1.9. Primeros contactos con Software *Mission Planner*.

Se comienza la programación inicial del controlador automático de vuelo sobre la unidad Arduino (APM Ardu Copter 2.6) para lo cual en primer lugar se realiza una actualización del software de la controladora adquirida, para ello se hace uso del Software *Mision Planner*, realizando los siguiente pasos:

- Se instala en el PC el Software Mision Planner (Enlace descarga).
- Una vez instalado conectar la unidad mediante USB, esta deberá ser reconocida por el PC, (de no ser así deberán instalarse sus correspondientes drivers).
- Se ejecuta el programa *Mission Planner*, y seleccionar el puerto en la que está instalada la unidad o **bien la conexión UDP si se conecta a través de mavlink**. **NO debe hacer click en conectar (CONNECT).**
- En el menú **INITIAL SETUP**, seleccionar la opción **Install Firmware** esta presentara los distintos modelos de Unidades móviles para el cual se puede configurar la unidad de control, en este caso se selecciona APM Copter v3.0.1



Please click the images above for "Flight versions"

Ardupilot is free software. Click here to read the Open Source Firmware License.

Figura 26 Actualización Firmware controlador Vuelo.

Confirmando en el mensaje que presenta para actualizar realizara la actualización del firmware, una vez terminado presentara los correspondientes mensajes de los ajustes que deben realizarse para el correcto funcionamiento, siendo estos:

- Selección de la estructura modo de vuelo (Quapcopter, en X, en +).
- Ajustes del compas
- Ajuste del acelerómetro
- Ajuste de emisora (Seguir indicaciones en manual Transmisor RF 2.4Ghz. que se utilice)

A partir de estos momentos si no ha habido errores la unidad estará dispuesta para poder ser armada.

Anexo B

[Ir al índice]

MANUAL ENSAMBLAJE DRON
Y
VERIFICACION MODULAR DE FUNCIONAMIENTO

1. Equipos de desarrollo

Para el desarrollo del proyecto se requieren de equipos diversos y herramientas tanto de electrónica como de informática. A continuación se pasa a describir brevemente cada uno de los bloques que componen el sistema. El trabajo se desarrolla mediante un ordenador de sobremesa concretamente equipo Intel(R) Core(TM) i7 CPU 939 @ 2.80 GHz con 12Gb de memoria.

Los programas de soporte general utilizados han sido obtenidos bien mediante código abierto o bien a través de las versiones para estudio que nos facilita la propia UOC en su programa Microsoft DreamSpark para uso académico, así se utilizara:

- **Sistema Operativo Windows 7 64bits**
- **Microsoft Office** (incluyendo FrontPage)
- **Microsoft Project**
- **Visual Studio 12**
- **Microsoft Studio Mx** (para creación WEB)
- **MySql** (base de datos)
- **Apache** (para montaje Servidor WEB)

Programas específicos para implementación del sistema central del proyecto:

- **Sistema Operativo Raspbian** para Raspberry Pi 2
- **Open CV y Phyton** para tratamientos de imágenes y
- **Arduino**: Entorno de Desarrollo Integral para la programación y grabación de Arduino
- **Tera Term**: Como terminal de comunicación serie (USB).

Programas específicos para Diseño Electrónico:

- **OrCad Capture**: Para creación de esquemas eléctricos
- **FreeView**: Software Captura Osciloscopio PROMAX OD-571 y Multímetros PROMAX

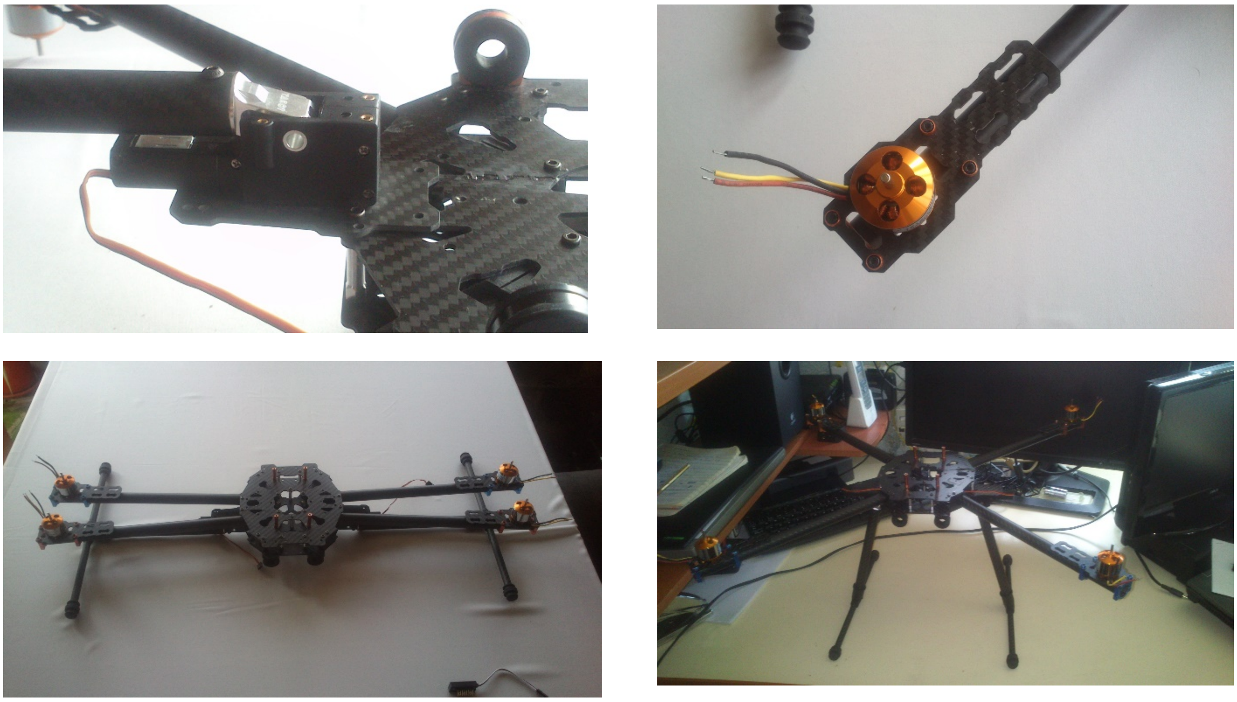
Como herramientas específicas para electrónica y microelectrónica:

- **Osciloscopio Promax OD-571 (150Mhz)**
- **Multímetro Materman 38XR**
- **LCR Meter PROMAX MZ-505C**
- **Estación Soldadora-Desoldadora ERSA Icon**
- **Corta hilos, destornilladores, pela hilos, pinzas sujeción placas, entre otros**

2 Ensamblaje estructura y sistemas de propulsión TAROT IRON 650

El montaje de la unidad Dron se efectúa sobre la estructura de cuatro motores TAROT IRON 650 (*Quapcopter*), efectuando modificaciones para el correcto acoplamiento de los módulos descritos anteriormente.

Entre dichas modificaciones se efectúa una bajada en altura de los soportes de fijación tipo RAIL con la finalidad de disponer de un espacio intermedio donde se situara el correspondiente sistema de control de velocidad de motores (ESC), quedando este entre la estructura central y el sistema de fijación los soportes tipo RAIL convenientemente aislado y protegido por placas de fibra de carbono. En las siguientes imágenes puede verse la estructura y ensamblaje de motores llevada a cabo.



Figuras 1 montajes Estructura y motores A2212/13T

En la imagen de la parte superior izquierda se puede apreciar la fijación del servo correspondiente al tren de aterrizaje plegable, con la finalidad de que el sistema de video situado en la parte inferior pueda disponer sin necesidad de cambiar el sentido de vuelo de una visión de 360º, cabe indicar que esto es en el caso de utilizar un sistema estabilizador del sistema de captura de video de tres ejes. (Este es opcional ya que en este caso se utilizara un sistema de dos ejes).

En la imagen superior derecha podemos observar la fijación de los motores utilizados y finalmente en las imágenes inferiores de la figura 12 la estructura montada con el tren de aterrizaje plegado e igualmente con el tren de aterrizaje desplegado.

Una vez llevado a cabo el ensamblaje de la estructura y motores se realiza el acople de los sistemas reguladores de motores (ESC) de la unidad Dron. Efectuando previamente las modificaciones mencionadas anteriormente que consisten en instalar un separador metálico entre de los soportes RAIL (silentblock) y su fijación a la estructura.

Igualmente se procede a cablear los motores realizando el cableado por el interior del brazo, previamente los cables han sido enfundados con maya protectora para evitar posibles problemas por deterioro de los mismos. Quedando finalmente como se muestra en la figura 13.

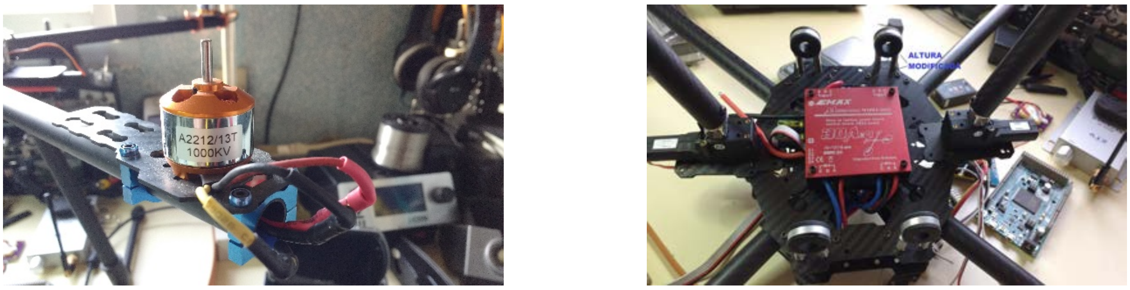
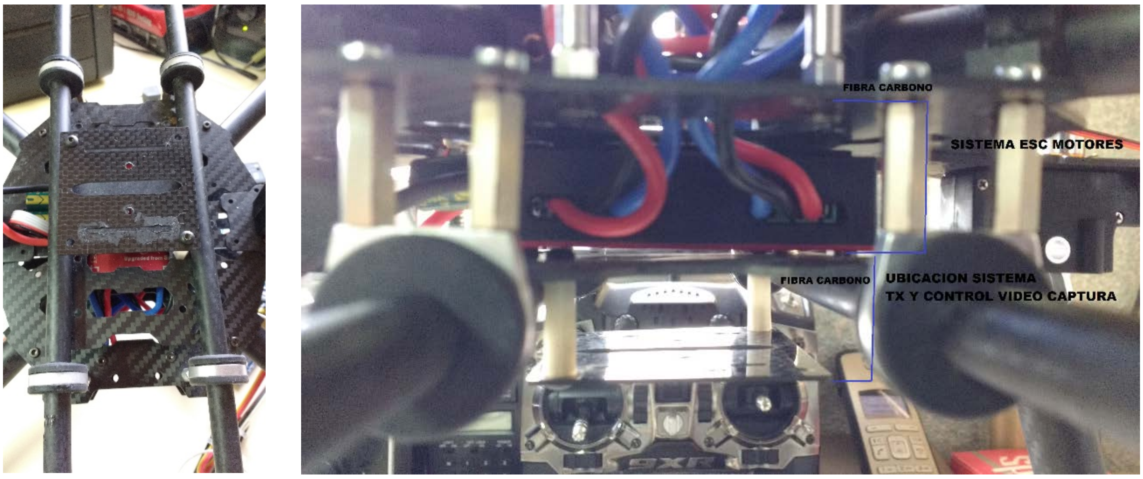


Figura 2 conexionado motores (Elaboración propia)

Para la fijación de los reguladores ESC se intercala una placa de carbono (para evitar posibles interferencias de los sistemas de trasmisión en los reguladores del sistema motor. Quedará así, debido a la conductividad del material (fibra de carbono), creada una *Jaula de Faraday* en cuyo interior se alojaran los mencionados reguladores. Igualmente se instalara otra base inferior donde será acoplado el modulo transmisión de video. Quedan así dos alojamientos para cada uno de los sistemas indicados como puede verse en las siguientes imágenes una vez efectuado el montaje.



Figuras 3. Encaje sistemas ESC y Sistema Transmisión Video

El montaje efectuado así como su funcionamiento puede verse detalladamente en video *Ensamblado estructura y propulsión* existente en repertorio de videos aportado para uso en el alojamiento Web del presente proyecto. Enlace a video

2 Verificación, ajuste y medida de consumos de los diferentes módulos.

Verificación sistema propulsor

La siguiente figura muestra el conexionado de los cables de control de motores, cuyo funcionamiento se verifica a continuación. Para ello se requiere de las herramientas electrónicas siguientes:

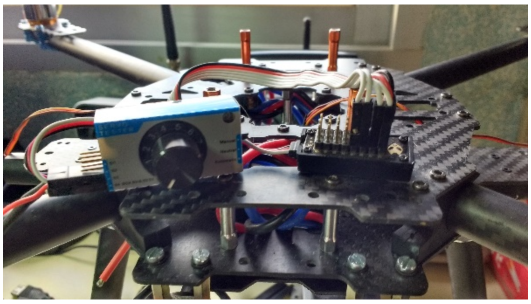


Figura 4 conexionado prueba motores propulsores

- 1 Switch para servos
- 1 Testeador Servos (Gen. y reg. de pulsos PWM)
- 1 Fuente alimentación corriente continua regulable ajustada a 11.7 y otra de 5V 1A.

Situando el control de motores en el CH3 del (Switch)y efectuando las conexiones de alimentación (11.7V a ESC motores y 5V a Testeador Servos), se calibra y ajusta el regulador *4IN1 ESC Emax 30A*.

Proceso de calibrado de motores

Para ello colocamos el regulador al máximo y conectamos alimentación, esperamos unos segundos y se oirán los pitidos de aviso de calibración terminada procedentes del ESC IMAX 30A.

A continuación se sitúa el regulador al mínimo y deberán de oírse los sonidos de ajuste de calibración de mínimos.

Figura 5 F.A. auxiliar para verificación

Una vez terminado el proceso, lo cual se sabrá por qué el sistema ESC deja de emitir sonidos, ***se retira la alimentación (batería de la unidad Dron o como en este caso F.A.) posteriormente asegurándose de situar regulador en posición mínimo*** se vuelve a alimentar la unidad.

Esta mediante una serie de sonidos nos indicara que está dispuesta para uso confirmando la activación del sistema controlador de potencia de los motores. Si se actúa sobre el regulador se debe observar que a mínimo los motores quedan parados y a máximo valor del regulador los motores alcanzan su máxima velocidad.

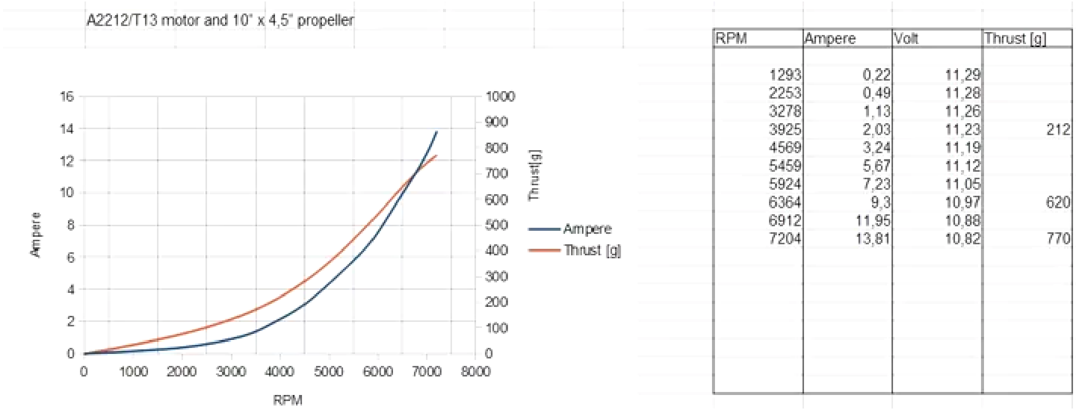
Realizados estos ajustes se obtiene un funcionamiento correcto concluyendo así el montaje y la verificación del sistema de control de los motores propulsores. Efectuando medida de consumo empíricamente se obtienen los siguientes resultados:

- **Consumo Sistema en estado de reposo: 250mA**
- **Consumo Sistema propulsión (motores máxima potencia) 2.5A (**sin carga)**

Respecto del comportamiento de los motores seleccionados, A221213T hacer referencia a las pruebas visualizadas (video existentes en internet y que se indican en referencias) “las pruebas empíricas marcan un rendimiento en medida de peso de elevación de 212g. a 770gr. es menor al indicado en características técnicas indicadas del motor 900gr.

Si bien hay que poner de manifiesto que según torque del motor por su referencia las hélices utilizadas en la prueba por lo que puede apreciarse no son las correctas, estas deberían de ser de 12” y como puede apreciarse en la figura 16 se han utilizado de 10” lo que produce una menor fuerza de elevación.

En la siguiente figura se puede ver las características de trabajo del motor, relación consumo-rpm y fuerza de elevación en gramos.



RPM	Ampere	Volt	Thrust [g]
1293	0,22	11,29	
2253	0,49	11,28	
3278	1,13	11,26	
3925	2,03	11,23	212
4569	3,24	11,19	
5459	5,67	11,12	
5924	7,23	11,05	
6364	9,3	10,97	620
6912	11,95	10,88	
7204	13,81	10,82	770

Figura 6 Relación consumo-rpm y fuerza elevación motores

3 Sistema Controlador de Vuelo (APM 2.6 ArduCopter)

El sistema embebido de control de vuelo y piloto automático, es instalado en la parte superior de la estructura. Con ello se pretende que la propia estructura dificulte que puedan interferir las señales correspondientes del sistema de transmisor de video (5.8GHz) sobre el sistema con control de vuelo (APM Ardu Copter 2.6) o sobre el receptor de radio-control igualmente ubicado en esta zona de la estructura.

El correspondiente cableado queda salvaguardado prácticamente en su totalidad en la parte central de la estructura bajo el sistema anti vibración instalada sobre el que se montara la unidad de control de vuelo, dejando únicamente la longitud suficiente para la interconexión ente los dispositivos que se utilizaran para el control los cuales son:

- Del receptor RX de Radio-control RF 2.4Ghz a unidad APM
- De la unidad APM, hasta el dispositivo de sistema central Raspberri, para control por MAVLink.

Igualmente a la controladora de vuelo se conectan los distintos dispositivos periféricos como es la unidad GPS para localización (RS232), y la brújula compas (mediante comunicación bus I2C).

El controlador de vuelo será conectado al sistema central y al OSD (On Screen Display) mediante su puerto de telemetría UART 0/1.

Quedando de esta forma el el sistema de control de vuelo disponible para poder ser comandado según necesidades, de alcance, maniobra, servicio, etc… mediante el uso de los correspondientes sistemas programados.

En las siguientes figura se puede observar la distribución que se ha dado en la parte superior de la estructura para anclaje del GPS, la fijación del soporte anti vibraciones para la APM, y los anclajes para la fijación del receptor de radio-control.

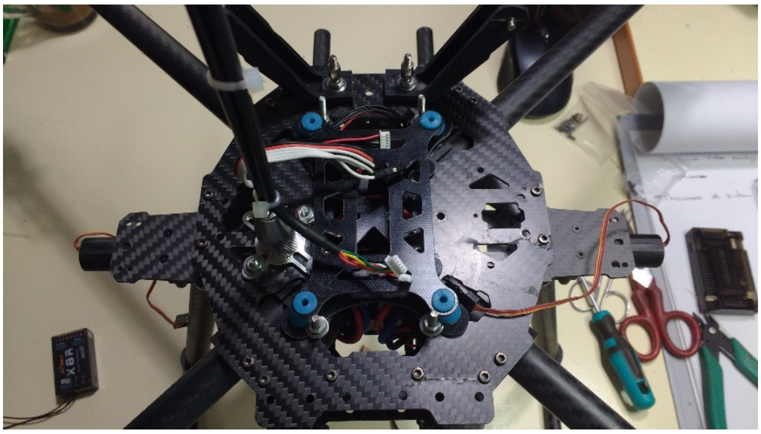


Figura 7 disposicion APM sobre estructura.

4 Instalación Sistema Estabilizador de cámara (*Gimbal 2 Ejes*)

Problemas de proveedores ha obligado a efectuar modificaciones no previstas, ya que no se ha podido obtener en tiempos el dispositivo estabilizador del sistema de video Tarot el cual estaba previsto en el diseño original. Se ha utilizado un sistema estabilizador (Gimbal de 2 Ejes) DJI para Go pro. Este es más pequeño lo que ha supuesto algunos problemas que de adaptación para poder calibrar y obtener un correcto funcionamiento del sistema estabilizado el cual se ha tenido que limitar los límites de ángulos Roll y Pitch para poder obtener un óptimo resultado y que la unidad estabilizadora quede calibrado.

Estos ángulos han quedado por tanto limitados a 30º, que si bien creemos son suficientes pues no se espera que la en vuelo llegue a esas inclinaciones, si ha llevado una serie de problemas que ha dejado la unidad limitada en una parte importante como es la estabilidad de la captura de imagen.

En las siguientes imágenes se muestra lo indicado se puede ver como el sistema de anclaje es pequeño por lo que se ha tenido que invertir el soporte de cámara y en vez que de la cámara (en este caso el sistema Raspberry) repose sobre el soporte se ha tenido que instalar quedando suspendido el de dicho soporte como puede apreciarse en la imagen de la derecha.

Ello limita los ángulos de giro tanto el Roll como puede verse en la imagen de la izquierda como el Pitch por llegar a un punto donde el sistema choca con el motor del sistema estabilizado.

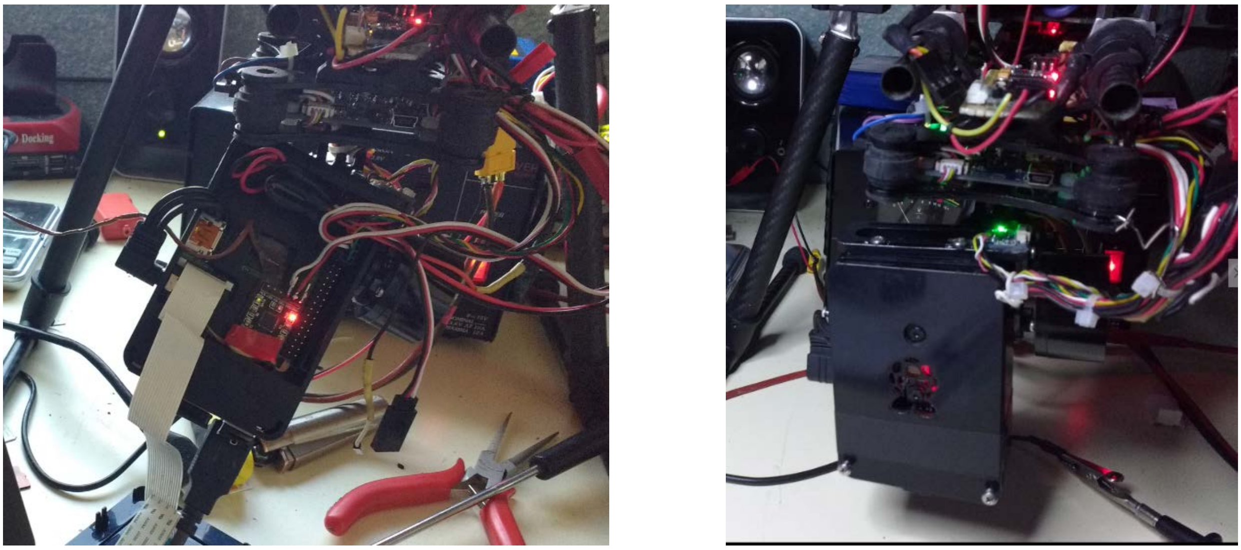
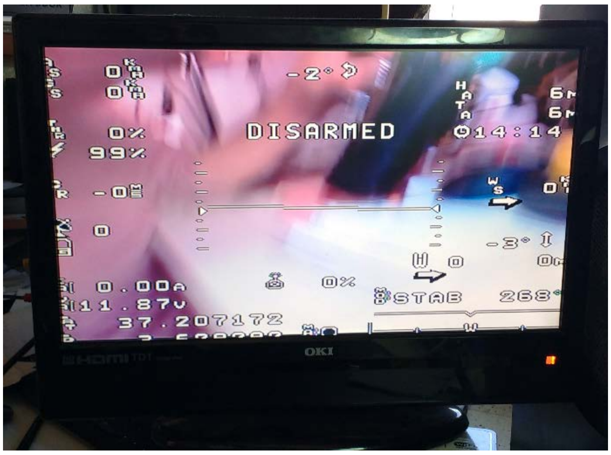


Figura 8 modificaciones en sistema estabilizado.

Finalmente adaptando la estructura del Gimbal, y efectuando fijación mediante tornillos de la caja del sistema Raspberry Pi se ha conseguido aunque con las limitaciones en los ángulos como se ha indicado cosa que no puede remediarse sin efectuar el cambio de la estructura Gimbal por la que inicialmente se había planteado y que no presentaba estos problemas pues es mayor.

Esto suponía un problema sería pues como puede verse en la siguiente figura con la cámara sin estabilizar es imposible pues visionarse y mucho menos usar técnicas detección de color como se pretende llevar a cabo para la detección de incendios.



Figuran 8 Problemas de vibración en Sistema estabilizador de camara

La solución ha pasado por la limitación de los ángulos, invertir el acelerómetro instalado en el dispositivo estabilizado, y realizar la calibración correspondiente de la controladora de Gimbal mediante su programación, ya que invertir el soporte el sistema lo entiende como girado sobre el Pitch no consiguiendo estabilizar el sistema ni tan siquiera estando el Dron posado con las consecuencias que pueden verse en la figura 8.

Una vez corregido puede verse en la siguiente figura como ya la imagen queda estabilizada y fijada incluso aunque varie el Angulo de inclinación (dentro de los márgenes indicados).

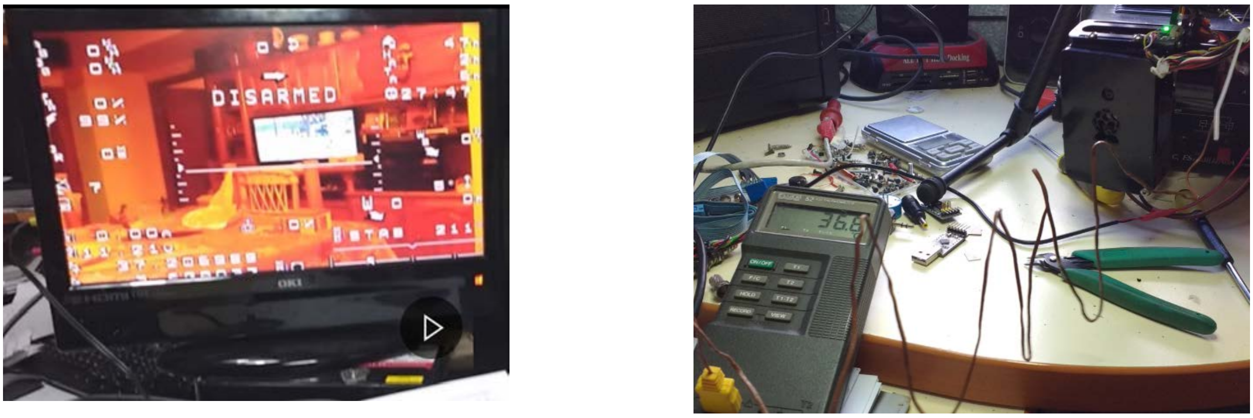


Figura 9 Problemas de vibración en Sistema estabilizador de cámara

En la Web del proyecto puede verse mediante video el montaje y su correcto funcionamiento, quedando así por tanto calibrado el sistema estabilizador.

Este se ha verificado dejando el sistema en funcionamiento largo tiempo (entre 4-5 horas) con sonda de temperatura colocada en motores para verificar que la temperatura de estos no sea elevada como indica el fabricante ya que al una alta temperatura puede dañar alguno de sus imanes internos y por tanto inutilizar el sistema estabilizador definitivamente.

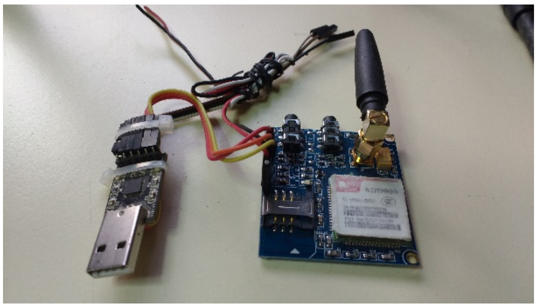
La temperatura obtenida no ha llegado nunca a los 39º, teniendo en cuenta que en la ubicación la temperatura ambiente es de unos 23º se considera que el funcionamiento es correcto.

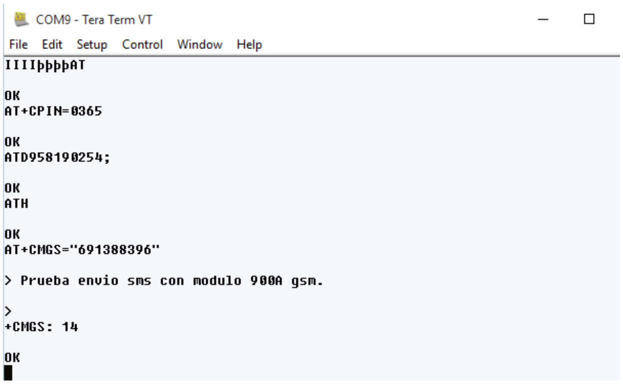
5 Verificación Módulo GSM-900A

El modulo adquirido por cuestiones de precio ha sido el módulo GSM900A de doble banda 900-1800Mhz (importado de Japón) ya que los aquí existentes *Quadband*, dichas bandas no son útiles en Europa y el costo es mucho más elevado. Cabe indicar que se debe de actualizar su firmware al de Europa para su correcto funcionamiento en Europa. Proceso que llevamos a cabo.

Una vez actualizado el mismo, efectuamos la conexión del módulo GSM900A mediante herramienta electrónica conversor (RS232-USB) CP2101 al PC con la finalidad de verificar su correcto funcionamiento.

En la siguiente figura puede verse el cableado para su conexión directamente al PC (a través de USB). Haciendo uso del *Software TeraTem*, (como se observa en la figura de la derecha), se efectúa la conexión con el dispositivo, se le indica el PIN efectuando una llamada de prueba a teléfono fijo y se realiza envió de un mensaje SMS a teléfono móvil, obteniendo resultado positivos.





```
IIIIþþþþAT

OK
AT+CPIN=0365

OK
ATD958190254;

OK
ATH

OK
AT+CMGS="691388396"

> Prueba envio sms con modulo 900A gsm.

>
+CMGS: 14

OK
```

Figura 10 Cableado GSM-CP2102 y Captura comunicación con PC via GSM

En dichas pruebas se efectúa toma empírica de medidas, mediante interconexión de Multímetro. Obteniendo los siguientes resultados de consumo en reposo y efectuando transmisión (llamada telefónica) y envió de SMS:

- **Consumo GSM en estado de reposo: 18-20mA**
- **Consumo GSM en estado de transmisión: 60mA**

Datos que a posteriori serán necesarios para los correspondientes de consumo y así poder determinar la potencia necesaria de las baterías a utilizar para determinar el tiempo final de vuelo y uso a pleno rendimiento de la unidad.

Queda por tanto verificado el módulo de comunicación GSM-GPRS que será utilizado el cual se instara sobre una placa de baquelita junto con la placa de sensores atmosféricos, la cual se ubicara sobre los soportes preparados como puede observarse en la figura 7 anterior, quedando así todos los dispositivos acoplados a la estructura

6 Ensamblaje sensores datos atmosféricos

Por último se lleva a cabo el montaje de los sensores para lo cual se ha efectuado un pequeño montaje sobre baquelita del sensor de temperatura, medidos de luminosidad y proximidad, y medido de presión barométrica, como puede verse en la siguiente figura   el sensor de humedad se fija sobre el propio chasis. Estos sensores comunican con el sistema central (Raspberry Pi) mediante bus I2C..

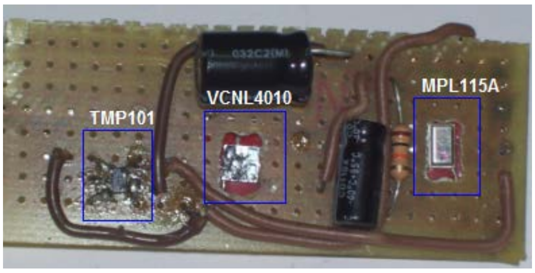
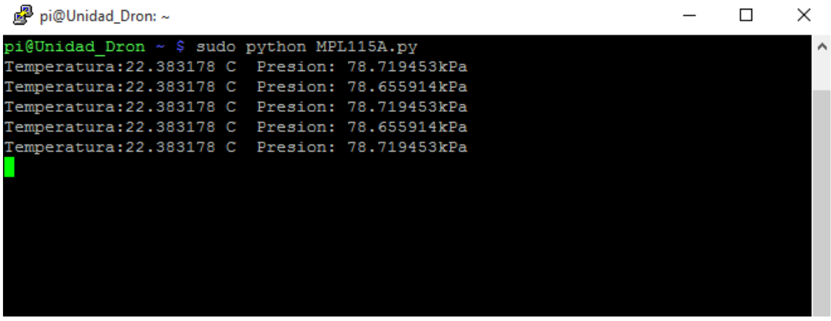


Figura 11 Placa de sensores



```
pi@Unidad_Dron ~ $ sudo python MPL115A.py
Temperatura:22.383178 C  Presion: 78.719453kPa
Temperatura:22.383178 C  Presion: 78.655914kPa
Temperatura:22.383178 C  Presion: 78.719453kPa
Temperatura:22.383178 C  Presion: 78.655914kPa
Temperatura:22.383178 C  Presion: 78.719453kPa
```

Figura. 12 placas de sensores

Dado que conviven en la unidad diversos sistemas de RF y se hace uso del osciloscopio, el cual se conecta al bus I2C y activando un bucle de petición de muestreo se pasa a medir la señal para observar si le están llegando las señales correctamente sin interferencias.

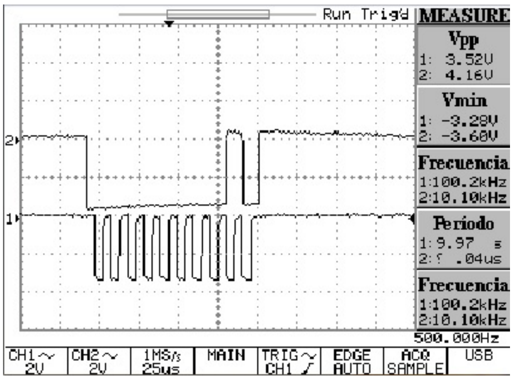


Figura. 13 Señal Bus I2C

Puede verse que efectivamente la señal llega correctamente a la frecuencia de 100Khz velocidad del bus y que no presenta interferencias, incluso armando motores por si estos le afectara.

Verificando el peso de la UAV, vemos que ha pasado de los 2Kg que se esperaba como máximo,

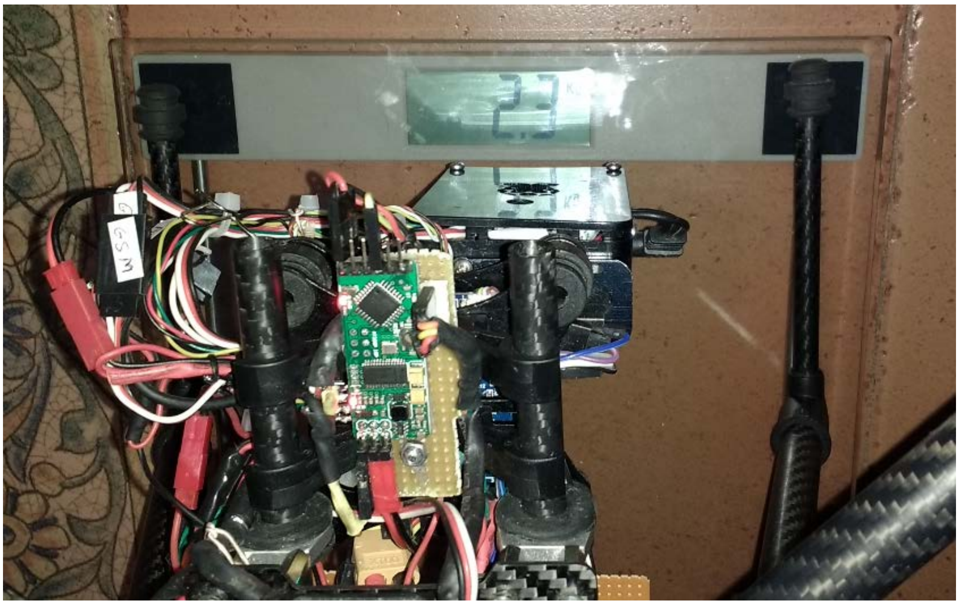


Figura. 14 Peso final de la unidad.

Estando como puede apreciarse su peso en 2,3Kg, con batería sería conveniente en un futuro revisar partes del montaje y ver en qué medida puede rebajarse dicho peso. Quedan por tanto verificados todos los módulos y componente que dando finalmente el ensamblaje como puede verse en la siguiente figura:

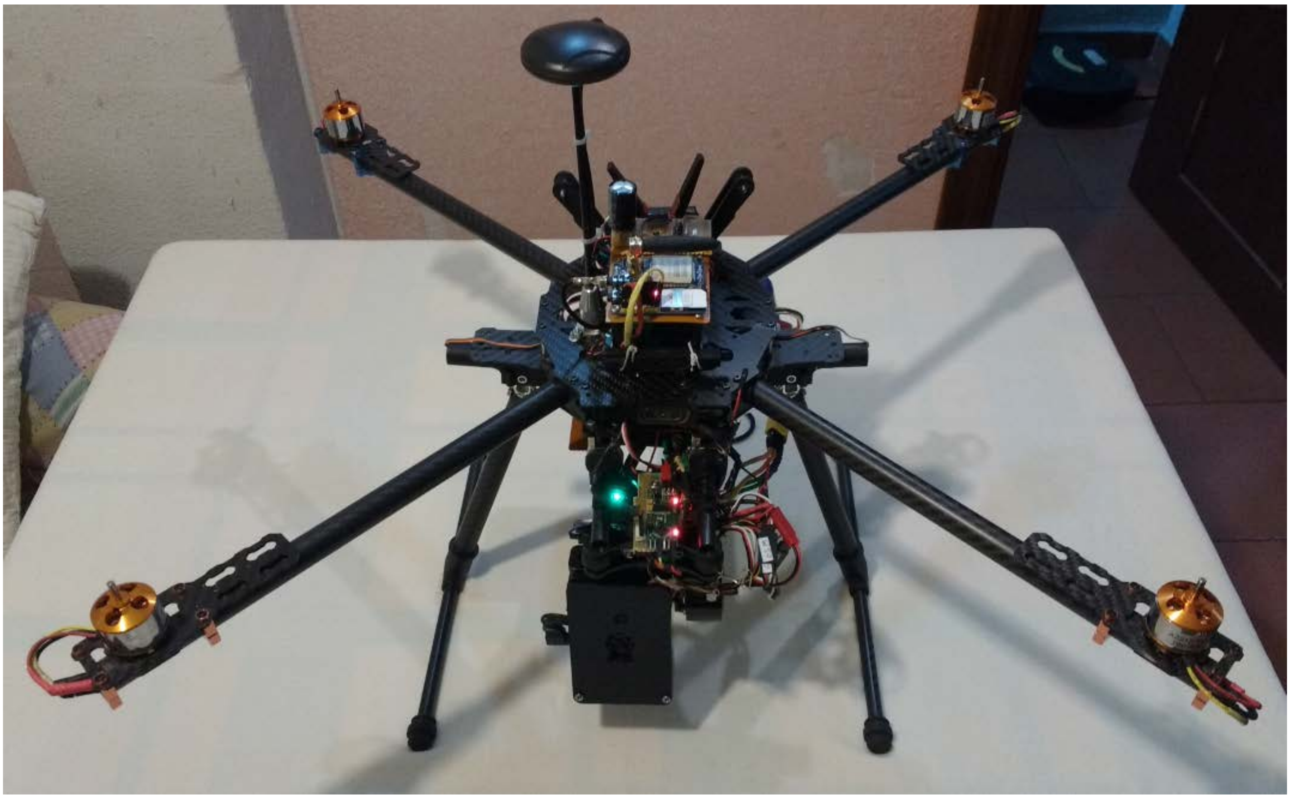
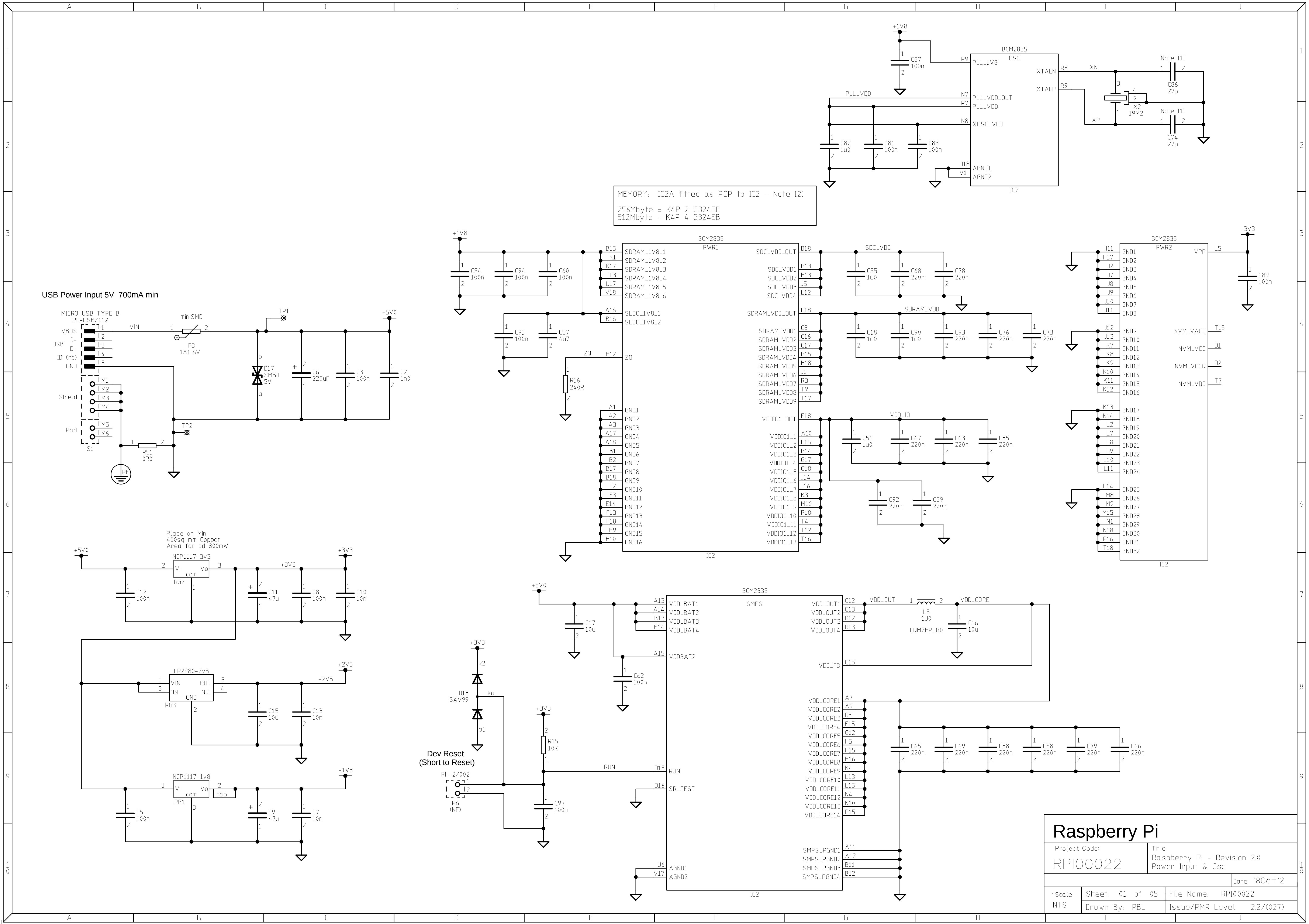


Figura. 15 montaje final de la UAV.

Anexo B

[Ir al índice]

ESQUEMAS ELECTRICOS
Y
DATASHEET



Raspberry Pi

Project Code: RPI00022

Title: Raspberry Pi - Revision 2.0 Power Input & Osc

Date: 18Oct12

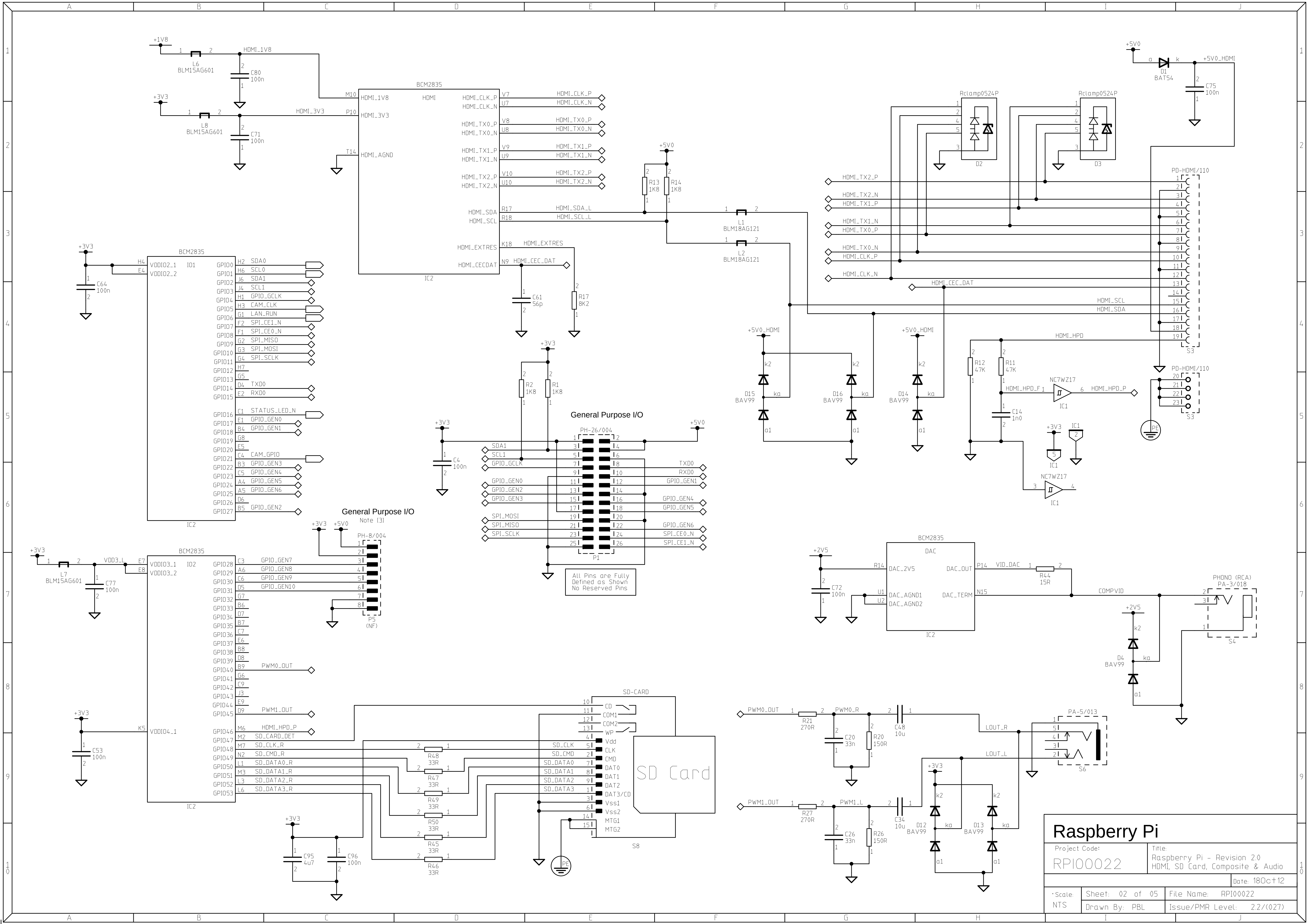
Scale: NTS

Sheet: 01 of 05

File Name: RPI00022

Drawn By: PBL

Issue/PMR Level: 2.2(027)



Raspberry Pi

Project Code: RPI00022

Title: Raspberry Pi - Revision 2.0
HDMI, SD Card, Composite & Audio

Date: 18Oct12

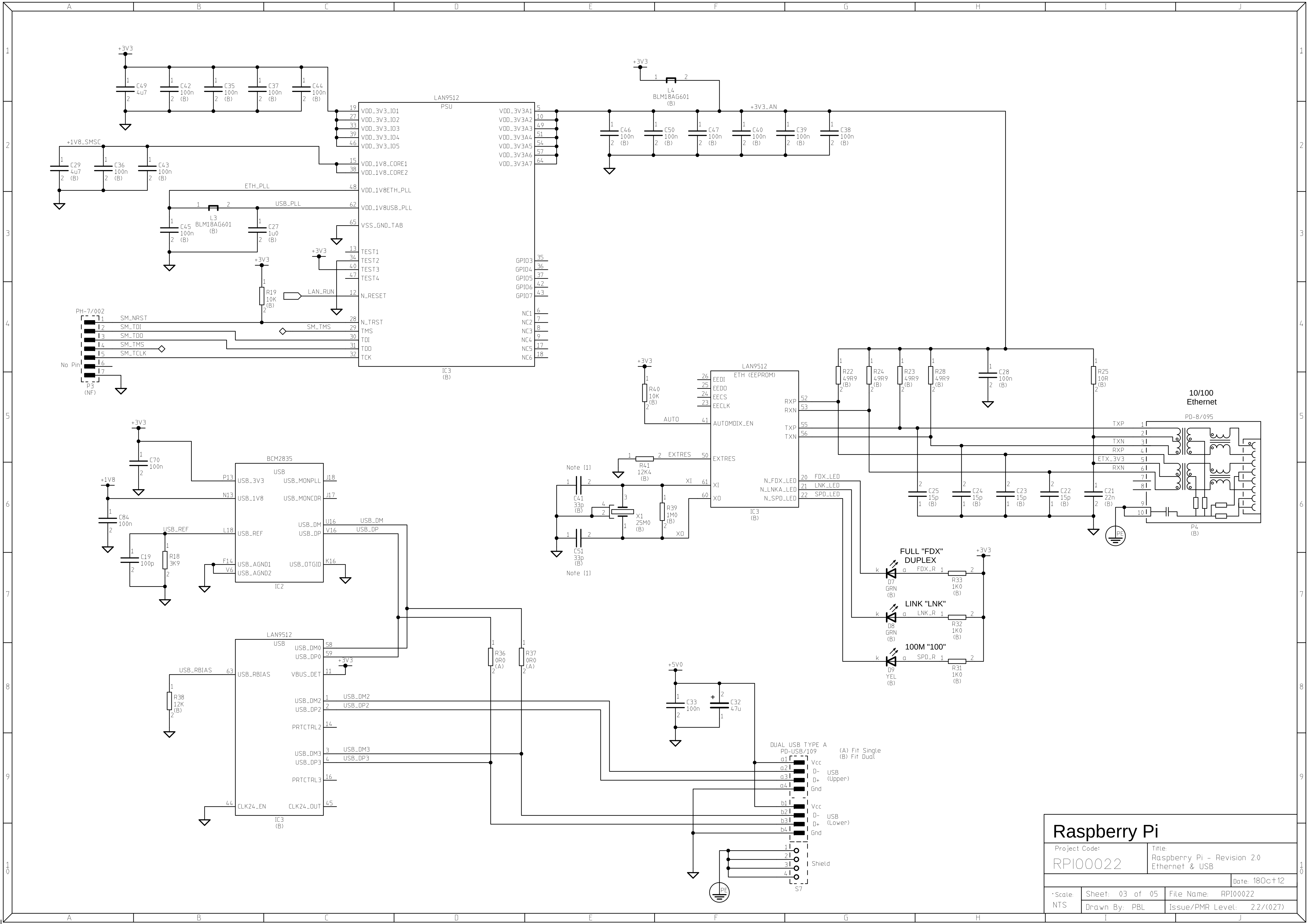
Scale: NTS

Sheet: 02 of 05

File Name: RPI00022

Drawn By: PBL

Issue/PMR Level: 2.2(027)



Raspberry Pi

Project Code: RPI00022

Title: Raspberry Pi - Revision 2.0 Ethernet & USB

Date: 18Oct12

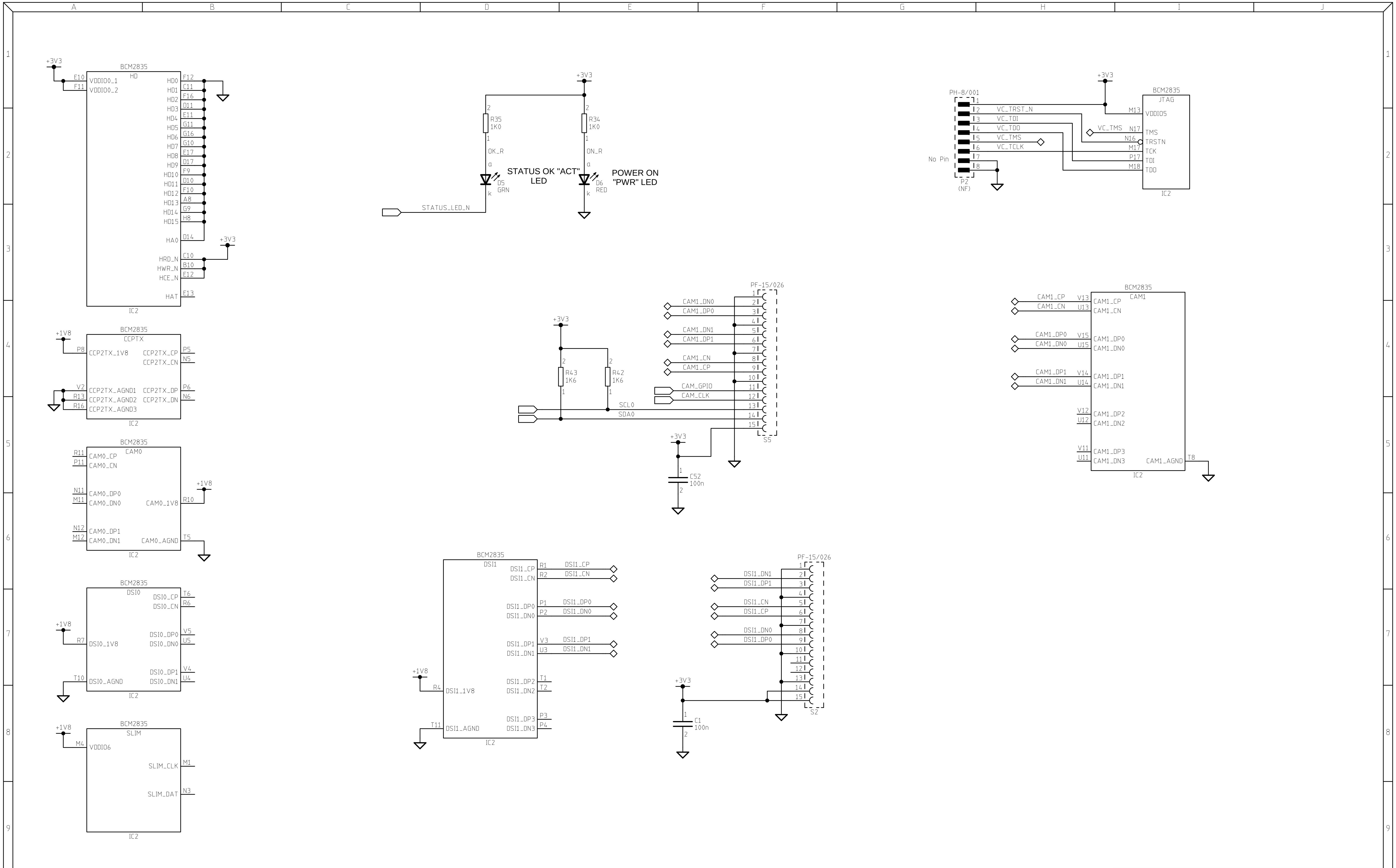
Scale: NTS

Sheet: 03 of 05

File Name: RPI00022

Drawn By: PBL

Issue/PMR Level: 2.2(027)



Raspberry Pi

Project Code: RPI00022	Title: Raspberry Pi - Revision 2.0 User I/O	
		Date: 18Oct12
Scale: NTS	Sheet: 04 of 05 Drawn By: PBL	File Name: RPI00022 Issue/PMR Level: 2.2(027)

PCB Build Variations

Each Build variant is assigned a letter A-Z. Components affected by a variant have that letter, in parentheses, associated with it on the schematic pages.

Build Variant	Build Description

Notes: A PCB assembly option ("Build Option") is formed by summing the required variants of circuit sections. Eg. A+D+G.

PCB Assembly Options

List of Assembly options (valid combinations of Build variants, for reference only.

Build variants	Build Description
A	Components only fitted to Model A (Basic)
B	Components only fitted to Model B (Full System)

PCB Parts List Entry

- Project Code - Issue

BD-RPI-00022/001

PCB Layout Requirements

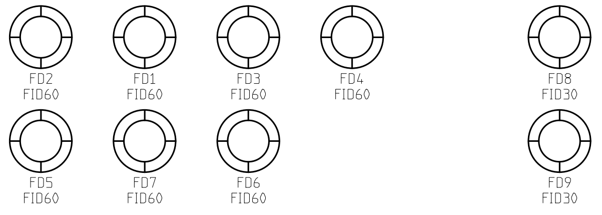
Layout notes referring to specific components or groups of components are indicated on the schematics by the note number, enclosed in braces, adjacent to the component. Eg {4}.

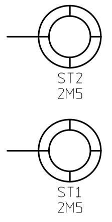
Note No.	Description

General Notes

These are general notes relating to the build of the product.

[Number]	Details
[1]	Capacitor values tied to crystal specification. Refer to production site specific instructions and BOM for additional information.
[2]	Use flux gel or solder paste for PoP attachment process.
[3]	Connector designed to be mounted undeside, DO NOT FILL holes in process.

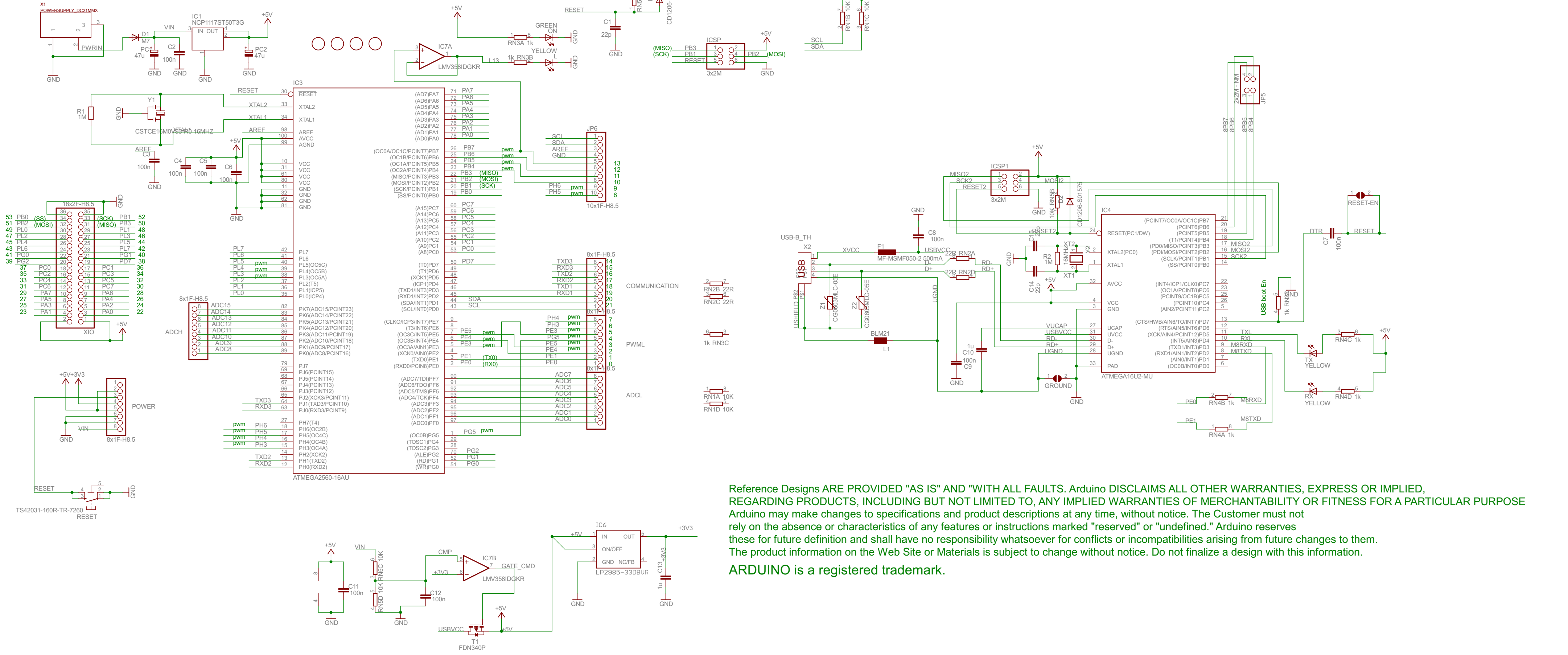




Raspberry Pi

Project Code: RPI00022	Title: Raspberry Pi - Revision 2.0 Build Options/PCB layout instructions.		
Client: n/a		Date: 18Oct12	
Scale: NTS	Sheet: 05 of 05	File Name: RPI00022	
	Drawn By: PBL	Issue/PMR Level: 2.2/(027)	

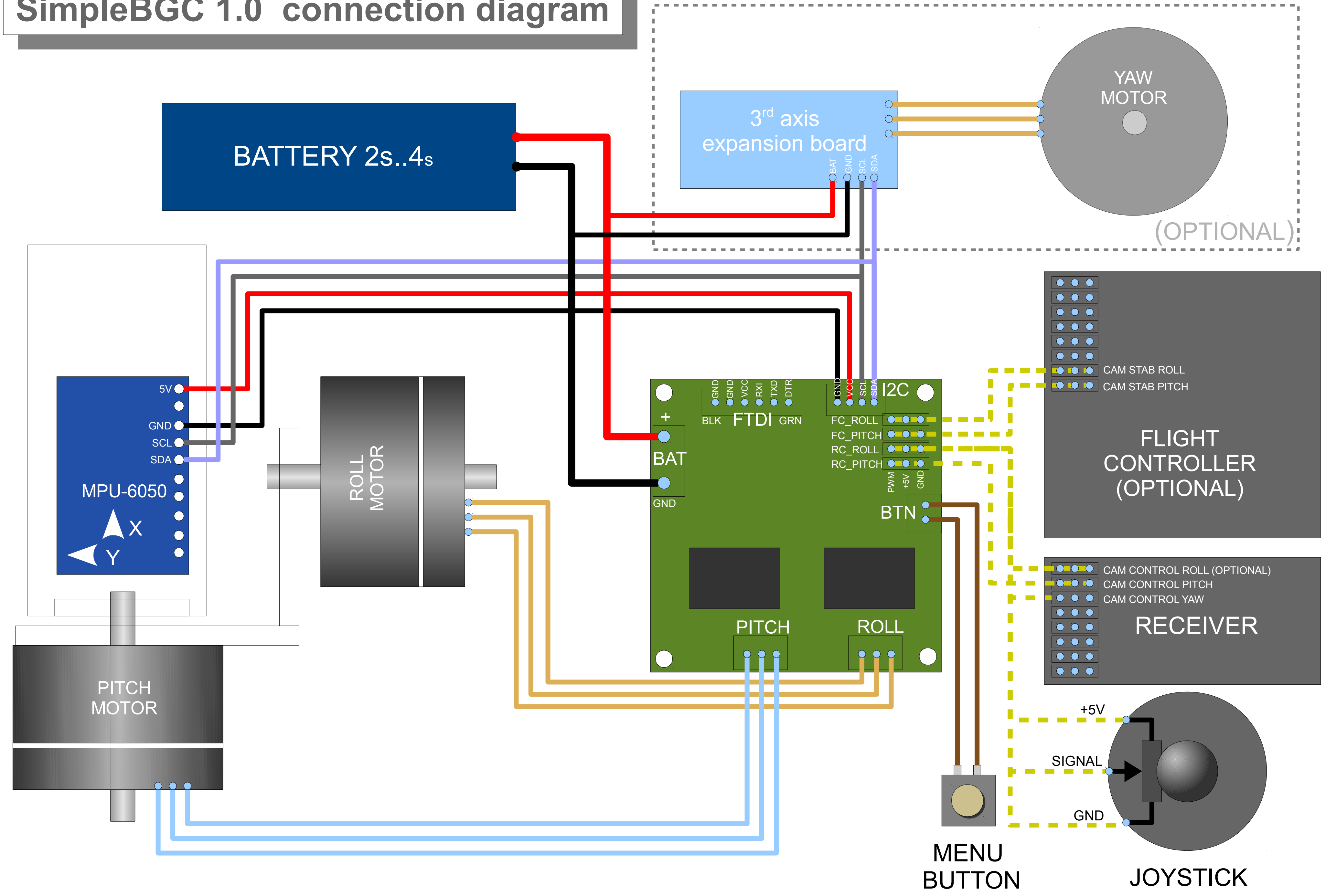
Arduino™ MEGA 2560



Reference Designs ARE PROVIDED "AS IS" AND "WITH ALL FAULTS. Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE Arduino may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Arduino reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

ARDUINO is a registered trademark.

SimpleBGC 1.0 connection diagram



BCM2835 ARM Peripherals

Broadcom Europe Ltd. 406 Science Park Milton Road Cambridge CB4 0WW

Table of Contents

1 Introduction

1.1 Overview

BCM2835 contains the following peripherals which may safely be accessed by the ARM:

- Timers
- Interrupt controller
- GPIO
- USB
- PCM / I2S
- DMA controller
- I2C master
- I2C / SPI slave
- SPI0, SPI1, SPI2
- PWM
- UART0, UART1

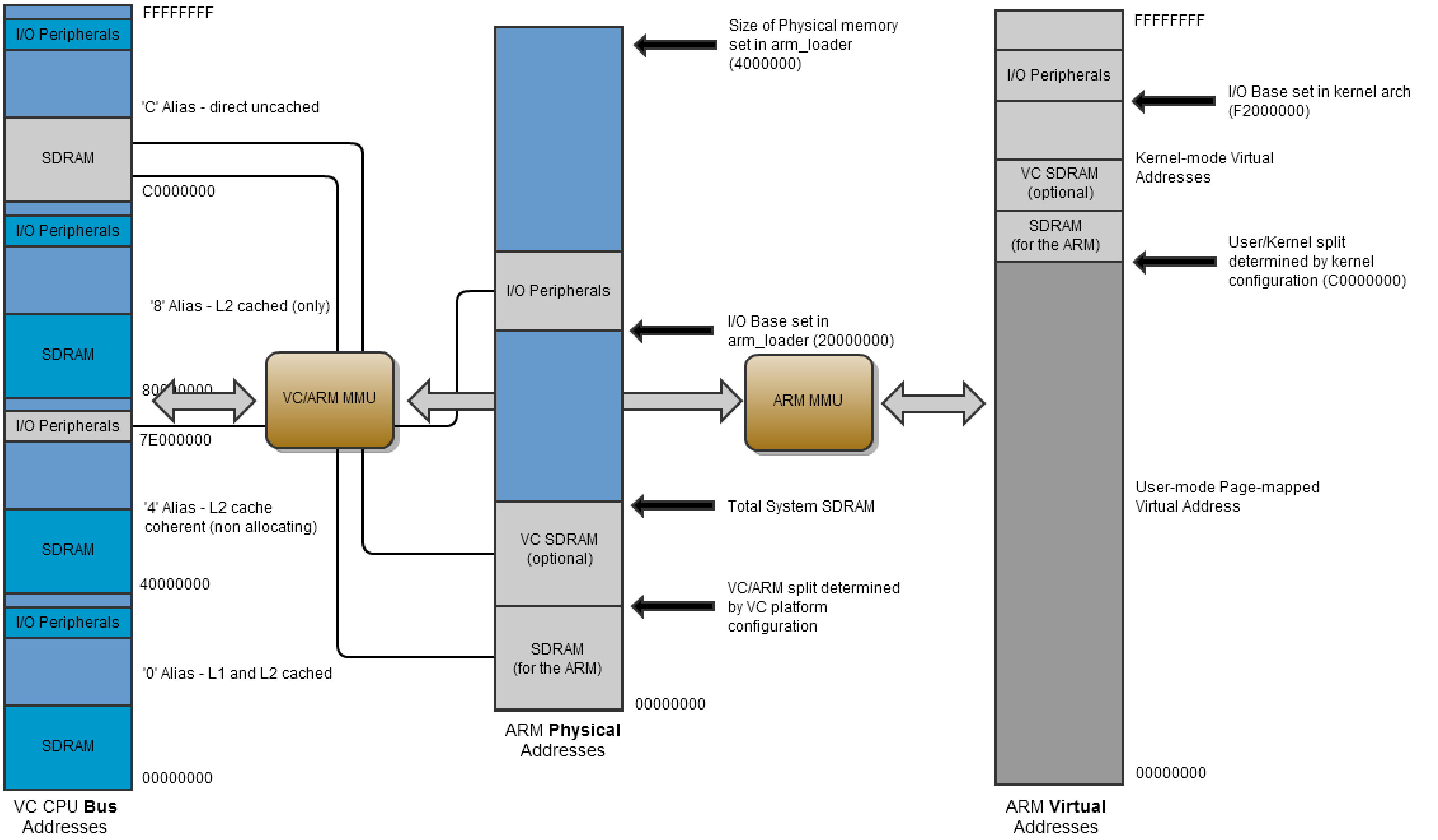
The purpose of this datasheet is to provide documentation for these peripherals in sufficient detail to allow a developer to port an operating system to BCM2835.

There are a number of peripherals which are intended to be controlled by the GPU. These are omitted from this datasheet. Accessing these peripherals from the ARM is not recommended.

1.2 Address map

1.2.1 Diagrammatic overview

In addition to the ARM's MMU, BCM2835 includes a second coarse-grained MMU for mapping ARM physical addresses onto system bus addresses. This diagram shows the main address spaces of interest:


FFFFFFFF
I/O Peripherals
'C' Alias - direct uncached
SDRAM
C0000000
I/O Peripherals
'8' Alias - L2 cached (only)
SDRAM
80000000
I/O Peripherals
7E000000
'4' Alias - L2 cache coherent (non allocating)
SDRAM
40000000
I/O Peripherals
'0' Alias - L1 and L2 cached
SDRAM
00000000
VC CPU Bus Addresses
VC/ARM MMU
Size of Physical memory set in arm_loader (4000000)
I/O Peripherals
I/O Base set in arm_loader (20000000)
Total System SDRAM
VC SDRAM (optional)
VC/ARM split determined by VC platform configuration
SDRAM (for the ARM)
00000000
ARM Physical Addresses
ARM MMU
FFFFFFFF
I/O Peripherals
I/O Base set in kernel arch (F2000000)
Kernel-mode Virtual Addresses
VC SDRAM (optional)
SDRAM (for the ARM)
User/Kernel split determined by kernel configuration (C0000000)
User-mode Page-mapped Virtual Address
00000000
ARM Virtual Addresses

Addresses in ARM Linux are:

- issued as virtual addresses by the ARM core, then
- mapped into a physical address by the ARM MMU, then
- mapped into a bus address by the ARM mapping MMU, and finally
- used to select the appropriate peripheral or location in RAM.

1.2.2 ARM virtual addresses (standard Linux kernel only)

As is standard practice, the standard BCM2835 Linux kernel provides a contiguous mapping over the whole of available RAM at the top of memory. The kernel is configured for a 1GB/3GB split between kernel and user-space memory.

The split between ARM and GPU memory is selected by installing one of the supplied start*.elf files as start.elf in the FAT32 boot partition of the SD card. The minimum amount of memory which can be given to the GPU is 32MB, but that will restrict the multimedia performance; for example, 32MB does not provide enough buffering for the GPU to do 1080p30 video decoding.

Virtual addresses in kernel mode will range between 0xC0000000 and 0xEFFFFFFF.

Virtual addresses in user mode (i.e. seen by processes running in ARM Linux) will range between 0x00000000 and 0xBFFFFFFF.

Peripherals (at physical address 0x20000000 on) are mapped into the kernel virtual address space starting at address 0xF2000000. Thus a peripheral advertised here at bus address 0x7Ennnnnn is available in the ARM kenel at virtual address 0xF2nnnnnn.

1.2.3 ARM physical addresses

Physical addresses start at 0x00000000 for RAM.

- The ARM section of the RAM starts at 0x00000000.
- The VideoCore section of the RAM is mapped in only if the system is configured to support a memory mapped display (this is the common case).

The VideoCore MMU maps the ARM physical address space to the bus address space seen by VideoCore (and VideoCore peripherals). The bus addresses for RAM are set up to map onto the uncached[1] bus address range on the VideoCore starting at 0xC0000000.

Physical addresses range from 0x20000000 to 0x20FFFFFF for peripherals. The bus addresses for peripherals are set up to map onto the peripheral bus address range starting at 0x7E000000. Thus a peripheral advertised here at bus address 0x7Ennnnnn is available at physical address 0x20nnnnnn.

1.2.4 Bus addresses

The peripheral addresses specified in this document are bus addresses. Software directly accessing peripherals must translate these addresses into physical or virtual addresses, as described above. Software accessing peripherals using the DMA engines must use bus addresses.

[1] BCM2835 provides a 128KB system L2 cache, which is used primarily by the GPU. Accesses to memory are routed either via or around the L2 cache depending on senior two bits of the bus address.

Software accessing RAM directly must use physical addresses (based at 0x00000000). Software accessing RAM using the DMA engines must use bus addresses (based at 0xC0000000).

1.3 Peripheral access precautions for correct memory ordering

The BCM2835 system uses an AMBA AXI-compatible interface structure. In order to keep the system complexity low and data throughput high, the BCM2835 AXI system does not always return read data in-order[2]. The GPU has special logic to cope with data arriving out-of-order; however the ARM core does not contain such logic. Therefore some precautions must be taken when using the ARM to access peripherals.

Accesses to the same peripheral will always arrive and return in-order. It is only when switching from one peripheral to another that data can arrive out-of-order. The simplest way to make sure that data is processed in-order is to place a memory barrier instruction at critical positions in the code. You should place:

- A memory write barrier before the first write to a peripheral.
- A memory read barrier after the last read of a peripheral.

It is **not** required to put a memory barrier instruction after **each** read or write access. Only at those places in the code where it is possible that a peripheral read or write may be followed by a read or write of a **different** peripheral. This is normally at the entry and exit points of the peripheral service code.

As interrupts can appear anywhere in the code so you should safeguard those. If an interrupt routine reads from a peripheral the routine should start with a memory read barrier. If an interrupt routine writes to a peripheral the routine should end with a memory write barrier.

[2] Normally a processor assumes that if it executes two read operations the data will arrive in order. So a read from location X followed by a read from location Y should return the data of location X first, followed by the data of location Y. Data arriving out of order can have disastrous consequences. For example:

```
a_status = *pointer_to_peripheral_a;
b_status = *pointer_to_peripheral_b;
```

Without precuations the values ending up in the variables a_status and b_status can be swapped around.

It is theoretical possible for writes to go 'wrong' but that is far more difficult to achieve. The AXI system makes sure the data always arrives in-order at its intended destination. So:

```
*pointer_to_peripheral_a = value_a;
*pointer_to_peripheral_b = value_b;
```

will always give the expected result. The only time write data can arrive out-of-order is if two different peripherals are connected to the same external equipment.

2 Auxiliaries: UART1 & SPI1, SPI2

2.1 Overview

The Device has three Auxiliary peripherals: One mini UART and two SPI masters. These three peripheral are grouped together as they share the same area in the peripheral register map and they share a common interrupt. Also all three are controlled by the auxiliary enable register.

Auxiliary peripherals Register Map (offset = 0x7E21 5000)			
Address	**Register Name**[3]	**Description**	**Size**
0x7E21 5000	AUX_IRQ	Auxiliary Interrupt status	3
0x7E21 5004	AUX_ENABLES	Auxiliary enables	3
0x7E21 5040	AUX_MU_IO_REG	Mini Uart I/O Data	8
0x7E21 5044	AUX_MU_IER_REG	Mini Uart Interrupt Enable	8
0x7E21 5048	AUX_MU_IIR_REG	Mini Uart Interrupt Identify	8
0x7E21 504C	AUX_MU_LCR_REG	Mini Uart Line Control	8
0x7E21 5050	AUX_MU_MCR_REG	Mini Uart Modem Control	8
0x7E21 5054	AUX_MU_LSR_REG	Mini Uart Line Status	8
0x7E21 5058	AUX_MU_MSR_REG	Mini Uart Modem Status	8
0x7E21 505C	AUX_MU_SCRATCH	Mini Uart Scratch	8
0x7E21 5060	AUX_MU_CNTL_REG	Mini Uart Extra Control	8
0x7E21 5064	AUX_MU_STAT_REG	Mini Uart Extra Status	32
0x7E21 5068	AUX_MU_BAUD_REG	Mini Uart Baudrate	16
0x7E21 5080	AUX_SPI0_CNTL0_REG	SPI 1 Control register 0	32
0x7E21 5084	AUX_SPI0_CNTL1_REG	SPI 1 Control register 1	8
0x7E21 5088	AUX_SPI0_STAT_REG	SPI 1 Status	32
0x7E21 5090	AUX_SPI0_IO_REG	SPI 1 Data	32
0x7E21 5094	AUX_SPI0_PEEK_REG	SPI 1 Peek	16
0x7E21 50C0	AUX_SPI1_CNTL0_REG	SPI 2 Control register 0	32
0x7E21 50C4	AUX_SPI1_CNTL1_REG	SPI 2 Control register 1	8

[3] These register names are identical to the defines in the AUX_IO header file. For programming purposes these names should be used wherever possible.

0x7E21 50C8	AUX_SPI1_STAT_REG	SPI 2 Status	32
0x7E21 50D0	AUX_SPI1_IO_REG	SPI 2 Data	32
0x7E21 50D4	AUX_SPI1_PEEK_REG	SPI 2 Peek	16

2.1.1 AUX registers

There are two Auxiliary registers which control all three devices. One is the interrupt status register, the second is the Auxiliary enable register. The Auxiliary IRQ status register can help to hierarchically determine the source of an interrupt.

AUXIRQ Register (0x7E21 5000)

SYNOPSIS The AUXIRQ register is used to check any pending interrupts which may be asserted by the three Auxiliary sub blocks.

Bit(s)	Field Name	Description	Type	Reset
31:3		Reserved, write zero, read as don’t care		
2	SPI 2 IRQ	If set the SPI 2 module has an interrupt pending.	R	0
1	SPI 1 IRQ	If set the SPI1 module has an interrupt pending.	R	0
0	Mini UART IRQ	If set the mini UART has an interrupt pending.	R	0

AUXENB Register (0x7E21 5004)

SYNOPSIS The AUXENB register is used to enable the three modules; UART, SPI1, SPI2.

Bit(s)	Field Name	Description	Type	Reset
31:3		Reserved, write zero, read as don’t care		
2	SPI2 enable	If set the SPI 2 module is enabled. If clear the SPI 2 module is disabled. That also disables any SPI 2 module register access	R/W	0
1	SPI 1 enable	If set the SPI 1 module is enabled. If clear the SPI 1 module is disabled. That also disables any SPI 1 module register access	R/W	0
0	Mini UART enable	If set the mini UART is enabled. The UART will immediately start receiving data, especially if the UART1_RX line is *low*. If clear the mini UART is disabled. That also disables any mini UART register access	R/W	0

If the enable bits are clear you will have ***no access*** to a peripheral. You can not even read or write the registers!

GPIO pins should be set up first the before enabling the UART. The UART core is build to emulate 16550 behaviour. So when it is enabled any data at the inputs will immediately be received . If the UART1_RX line is low (because the GPIO pins have not been set-up yet) that will be seen as a start bit and the UART will start receiving 0x00-characters.

Valid stops bits are not required for the UART. (See also Implementation details). Hence any bit status is acceptable as stop bit and is only used so there is clean timing start for the next bit.

Looking after a reset: the baudrate will be zero and the system clock will be 250 MHz. So only 2.5 µseconds suffice to fill the receive FIFO. The result will be that the FIFO is full and overflowing in no time flat.

2.2 Mini UART

The mini UART is a secondary low throughput[4] UART intended to be used as a console. It needs to be enabled before it can be used. It is also recommended that the correct GPIO function mode is selected before enabling the mini Uart.

The mini Uart has the following features:

- 7 or 8 bit operation.
- 1 start and 1 stop bit.
- No parities.
- Break generation.
- 8 symbols deep FIFOs for receive and transmit.
- SW controlled RTS, SW readable CTS.
- Auto flow control with programmable FIFO level.
- 16550 *like* registers.
- Baudrate derived from system clock.

This is a *mini* UART and it does NOT have the following capabilities:

- Break detection
- Framing errors detection.
- Parity bit
- Receive Time-out interrupt
- DCD, DSR, DTR or RI signals.

The implemented UART is *not* a 16650 compatible UART However as far as possible the first 8 control and status registers are laid out *like* a 16550 UART. Al 16550 register bits which are not supported can be written but will be ignored and read back as 0. All control bits for simple UART receive/transmit operations are available.

[4] The UART itself has no throughput limitations in fact it can run up to 32 Mega baud. But doing so requires significant CPU involvement as it has shallow FIFOs and no DMA support.

2.2.1 Mini UART implementation details.

The UART1_CTS and UART1_RX inputs are synchronised and will take 2 system clock cycles before they are processed.

The module does not check for any framing errors. After receiving a start bit and 8 (or 7) data bits the receiver waits for one half bit time and then starts scanning for the next start bit. The mini UART does *not* check if the stop bit is high or wait for the stop bit to appear. As a result of this a UART1_RX input line which is continuously low (a break condition or an error in connection or GPIO setup) causes the receiver to continuously receive 0x00 symbols.

The mini UART uses 8-times oversampling. The Baudrate can be calculated from:

$$baudrate = \frac{system\_clock\_freq}{8 * (baudrate\_reg + 1)}$$

If the system clock is 250 MHz and the baud register is zero the baudrate is 31.25 Mega baud. (25 Mbits/sec or 3.125 Mbytes/sec). The lowest baudrate with a 250 MHz system clock is 476 Baud.

When writing to the data register only the LS 8 bits are taken. All other bits are ignored. When reading from the data register only the LS 8 bits are valid. All other bits are zero.

2.2.2 Mini UART register details.

AUX_MU_IO_REG Register (0x7E21 5040)

SYNOPSIS The AUX_MU_IO_REG register is primary used to write data to and read data from the UART FIFOs.
If the DLAB bit in the line control register is set this register gives access to the LS 8 bits of the baud rate. (Note: there is easier access to the baud rate register)

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7:0	LS 8 bits Baudrate read/write, DLAB=1	Access to the LS 8 bits of the 16-bit baudrate register. (Only If bit 7 of the line control register (DLAB bit) is set)	R/W	0
7:0	Transmit data write, DLAB=0	Data written is put in the transmit FIFO (Provided it is not full) (Only If bit 7 of the line control register (DLAB bit) is clear)	W	0
7:0	Receive data read, DLAB=0	Data read is taken from the receive FIFO (Provided it is not empty) (Only If bit 7 of the line control register (DLAB bit) is clear)	R	0

AUX_MU_IIR_REG Register (0x7E21 5044)

SYNOPSIS The AUX_MU_IER_REG register is primary used to enable interrupts
If the DLAB bit in the line control register is set this register gives access to the MS 8 bits of the baud rate. (Note: there is easier access to the baud rate register)

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7:0	MS 8 bits Baudrate read/write, DLAB=1	Access to the MS 8 bits of the 16-bit baudrate register. (Only If bit 7 of the line control register (DLAB bit) is set)	R/w	0
7:2		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		
1	Enable receive interrupt (DLAB=0)	If this bit is set the interrupt line is asserted whenever the receive FIFO holds at least 1 byte. If this bit is clear no receive interrupts are generated.	R	0
0	Enable transmit interrupt (DLAB=0)	If this bit is set the interrupt line is asserted whenever the transmit FIFO is empty. If this bit is clear no transmit interrupts are generated.	R	0

AUX_MU_IER_REG Register (0x7E21 5048)

SYNOPSIS The AUX_MU_IIR_REG register shows the interrupt status.
It also has two FIFO enable status bits and (when writing) FIFO clear bits.

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7:6	FIFO enables	Both bits always read as 1 as the FIFOs are always enabled	R	11
5:4	-	Always read as zero	R	00
3	-	Always read as zero as the mini UART has no timeout function	R	0
2:1	READ: Interrupt ID bits WRITE: FIFO clear bits	On read this register shows the interrupt ID bit 00 : No interrupts 01 : Transmit holding register empty 10 : Receiver holds valid byte 11 : <Not possible> On write: Writing with bit 1 set will clear the receive FIFO Writing with bit 2 set will clear the transmit FIFO	R/W	00
0	Interrupt pending	This bit is clear whenever an interrupt is pending	R	1

AUX_MU_LCR_REG Register (0x7E21 504C)

SYNOPSIS The AUX_MU_LCR_REG register controls the line data format and gives access to the baudrate register

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7	DLAB access	If set the first to Mini UART register give access the the Baudrate register. During operation this bit must be cleared.	R/W	0
6	Break	If set high the UART1_TX line is pulled low continuously. If held for at least 12 bits times that will indicate a break condition.	R/W	0
5:1		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		0
0	data size	If clear the UART works in 7-bit mode If set the UART works in 8-bit mode	R/W	0

AUX_MU_MCR_REG Register (0x7E21 5050)

SYNOPSIS The AUX_MU_MCR_REG register controls the 'modem' signals.

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7:2		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		0
1	RTS	If clear the UART1_RTS line is high If set the UART1_RTS line is low This bit is ignored if the RTS is used for auto-flow control. See the Mini Uart Extra Control register description)	R/W	0
0		Reserved, write zero, read as don't care *This bit has a function in a 16550 compatible UART but is ignored here*		0

AUX_MU_LSR_REG Register (0x7E21 5054)

SYNOPSIS The AUX_MU_LSR_REG register shows the data status.

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7		Reserved, write zero, read as don't care *This bit has a function in a 16550 compatible UART but is ignored here*		0
6	Transmitter idle	This bit is set if the transmit FIFO is empty and the transmitter is idle. (Finished shifting out the last bit).	R	1
5	Transmitter empty	This bit is set if the transmit FIFO can accept at least one byte.	R	0
4:2		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		0
1	Receiver Overrun	This bit is set if there was a receiver overrun. That is: one or more characters arrived whilst the receive FIFO was full. The newly arrived charters have been discarded. This bit is cleared each time this register is read. To do a non-destructive read of this overrun bit use the Mini Uart Extra Status register.	R/C	0
0	Data ready	This bit is set if the receive FIFO holds at least 1 symbol.	R	0

AUX_MU_MSR_REG Register (0x7E21 5058)

SYNOPSIS The AUX_MU_MSR_REG register shows the 'modem' status.

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7:6		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		0
5	CTS status	This bit is the inverse of the UART1_CTS input Thus : If set the UART1_CTS pin is low If clear the UART1_CTS pin is high	R	1
3:0		Reserved, write zero, read as don't care *Some of these bits have functions in a 16550 compatible UART but are ignored here*		0

AUX_MU_SCRATCH Register (0x7E21 505C)

SYNOPSIS The AUX_MU_SCRATCH is a single byte storage.

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7:0	Scratch	One whole byte extra on top of the 134217728 provided by the SDC	R/W	0

AUX_MU_CNTL_REG Register (0x7E21 5060)

SYNOPSIS The AUX_MU_CNTL_REG provides access to some extra useful and nice features not found on a normal 16550 UART .

Bit(s)	Field Name	Description	Type	Reset
31:8		Reserved, write zero, read as don't care		
7	CTS assert level	This bit allows one to invert the CTS auto flow operation polarity. If set the CTS auto flow assert level is low* If clear the CTS auto flow assert level is high*	R/W	0
6	RTS assert level	This bit allows one to invert the RTS auto flow operation polarity. If set the RTS auto flow assert level is low* If clear the RTS auto flow assert level is high*	R/W	0
5:4	RTS AUTO flow level	These two bits specify at what receiver FIFO level the RTS line is de-asserted in auto-flow mode. 00 : De-assert RTS when the receive FIFO has 3 empty spaces left. 01 : De-assert RTS when the receive FIFO has 2 empty spaces left. 10 : De-assert RTS when the receive FIFO has 1 empty space left. 11 : De-assert RTS when the receive FIFO has 4 empty spaces left.	R/W	0
3	Enable transmit Auto flow-control using CTS	If this bit is set the transmitter will stop if the CTS line is de-asserted. If this bit is clear the transmitter will ignore the status of the CTS line	R/W	0

2	Enable receive Auto flow-control using RTS	If this bit is set the RTS line will de-assert if the receive FIFO reaches it 'auto flow' level. In fact the RTS line will behave as an RTR *(Ready To Receive)* line. If this bit is clear the RTS line is controlled by the AUX_MU_MCR_REG register bit 1.	R/W	0
1	Transmitter enable	If this bit is set the mini UART transmitter is enabled. If this bit is clear the mini UART transmitter is disabled	R/W	1
0	Receiver enable	If this bit is set the mini UART receiver is enabled. If this bit is clear the mini UART receiver is disabled	R/W	1

Receiver enable
If this bit is set no new symbols will be accepted by the receiver. Any symbols in progress of reception will be finished.

Transmitter enable
If this bit is set no new symbols will be send the transmitter. Any symbols in progress of transmission will be finished.

Auto flow control
Automatic flow control can be enabled independent for the receiver and the transmitter.

CTS auto flow control impacts the transmitter only. The transmitter will not send out new symbols when the CTS line is de-asserted. Any symbols in progress of transmission when the CTS line becomes de-asserted will be finished.

RTS auto flow control impacts the receiver only. In fact the name RTS for the control line is incorrect and should be RTR (Ready to Receive). The receiver will de-asserted the RTS (RTR) line when its receive FIFO has a number of empty spaces left. Normally 3 empty spaces should be enough.

If looping back a mini UART using full auto flow control the logic is fast enough to allow the RTS auto flow level of '10' (De-assert RTS when the receive FIFO has 1 empty space left).

Auto flow polarity
To offer full flexibility the polarity of the CTS and RTS (RTR) lines can be programmed. This should allow the mini UART to interface with any existing hardware flow control available.

AUX_MU_STAT_REG Register (0x7E21 5064)

SYNOPSIS The AUX_MU_STAT_REG provides a lot of useful information about the internal status of the mini UART not found on a normal 16550 UART.

Bit(s)	Field Name	Description	Type	Reset
31:28		Reserved, write zero, read as don't care		
27:24	Transmit FIFO fill level	These bits shows how many symbols are stored in the transmit FIFO The value is in the range 0-8	R	0
23:20		Reserved, write zero, read as don't care		
19:16	Receive FIFO fill level	These bits shows how many symbols are stored in the receive FIFO The value is in the range 0-8	R	0
15:10		Reserved, write zero, read as don't care		
9	Transmitter done	This bit is set if the transmitter is idle and the transmit FIFO is empty. It is a logic AND of bits 2 and 8	R	1
8	Transmit FIFO is empty	If this bit is set the transmitter FIFO is empty. Thus it can accept 8 symbols.	R	1
7	CTS line	This bit shows the status of the UART1_CTS line.	R	0
6	RTS status	This bit shows the status of the UART1_RTS line.	R	0
5	Transmit FIFO is full	This is the inverse of bit 1	R	0
4	Receiver overrun	This bit is set if there was a receiver overrun. That is: one or more characters arrived whilst the receive FIFO was full. The newly arrived characters have been discarded. This bit is cleared each time the AUX_MU_LSR_REG register is read.	R	0
3	Transmitter is idle	If this bit is set the transmitter is idle. If this bit is clear the transmitter is idle.	R	1
2	Receiver is idle	If this bit is set the receiver is idle. If this bit is clear the receiver is busy. This bit can change unless the receiver is disabled	R	1
1	Space available	If this bit is set the mini UART transmitter FIFO can accept at least one more symbol. If this bit is clear the mini UART transmitter FIFO is full	R	0

0	Symbol available	If this bit is set the mini UART receive FIFO contains at least 1 symbol If this bit is clear the mini UART receiver FIFO is empty	R	0

Receiver is idle
This bit is only useful if the receiver is disabled. The normal use is to disable the receiver. Then check (or wait) until the bit is set. Now you can be sure that no new symbols will arrive. (e.g. now you can change the baudrate...)

Transmitter is idle
This bit tells if the transmitter is idle. Note that the bit will set only for a short time if the transmit FIFO contains data. Normally you want to use bit 9: Transmitter done.

RTS status
This bit is useful only in receive Auto flow-control mode as it shows the status of the RTS line.

AUX_MU_BAUD Register (0x7E21 5068)

SYNOPSIS The AUX_MU_BAUD register allows direct access to the 16-bit wide baudrate counter.

Bit(s)	Field Name	Description	Type	Reset
31:16		Reserved, write zero, read as don't care		
15:0	Baudrate	mini UART baudrate counter	R/W	0

This is the same register as is accessed using the LABD bit and the first two register, but much easier to access.

2.3 Universal SPI Master (2x)

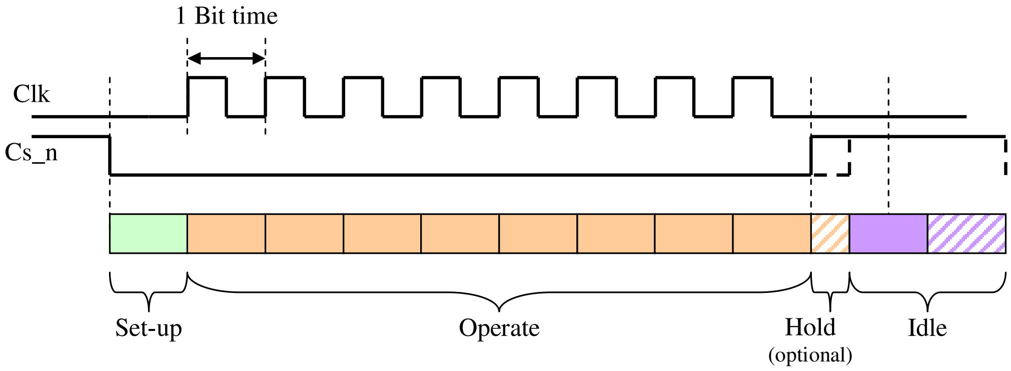
The two universal SPI masters are secondary low throughput[5] SPI interfaces. Like the UART the devices needs to be enabled before they can be used. Each SPI master has the following features:

- Single beat bit length between 1 and 32 bits.
- Single beat variable bit length between 1 and 24 bits
- Multi beat infinite bit length.
- 3 independent chip selects per master.
- 4 entries 32-bit wide transmit and receive FIFOs.
- Data out on rising or falling clock edge.
- Data in on rising or falling clock edge.
- Clock inversion (Idle high or idle low).
- Wide clocking range.
- Programmable data out hold time.
- Shift in/out MS or LS bit first

A major issue with an SPI interface is that there is no SPI standard in any form. Because the SPI interface has been around for a long time some pseudo-standard rules have appeared mostly when interfacing with memory devices. The universal SPI master has been developed to work even with the most 'non-standard' SPI devices.

2.3.1 SPI implementation details

The following diagrams shows a typical SPI access cycle. In this case we have 8 SPI clocks.



One bit time before any clock edge changes the CS_n will go low. This makes sure that the MOSI signal has a full bit-time of set-up against any changing clock edges.

The operation normally ends after the last clock cycle. Note that at the end there is one half-bit time where the clock does not change but which still is part of the operation cycle.

There is an option to add a half bit cycle hold time. This makes sure that any MISO data has at least a full SPI bit time to arrive. (Without this hold time, data clocked out of the SPI device on the last clock edge would have only half a bit time to arrive).

[5] Again the SPIs themselves have no throughput limitations in fact they can run with an SPI clock of 125 MHz. But doing so requires significant CPU involvement as they have shallow FIFOs and no DMA support.

Last there is a guarantee of at least a full bit time where the spi chip select is high. A longer CS_n high period can be programmed for another 1-7 cycles.

The SPI clock frequency is:

$$SPIx\_CLK = \frac{system\_clock\_freq}{2*(speed\_field+1)}$$

If the system clock is 250 MHz and the speed field is zero the SPI clock frequency is 125 MHz. The practical SPI clock will be lower as the I/O pads can not transmit or receive signals at such high speed. The lowest SPI clock frequency with a 250 MHz system clock is 30.5 KHz.

The hardware has an option to add hold time to the MOSI signal against the SPI clk. This is again done using the system clock. So a 250 MHz system clock will add hold times in units of 4 ns. Hold times of 0, 1, 4 and 7 system clock cycles can be used. (So at 250MHz an additional hold time of 0, 4, 16 and 28 ns can be achieved). The hold time is *additional* to the normal output timing as specified in the data sheet.

2.3.2 Interrupts

The SPI block has two interrupts: TX FIFO is empty, SPI is Idle.

TX FIFO is empty:
This interrupt will be asserted as soon as the last entry has been read from the transmit FIFO. At that time the interface will still be busy shifting out that data. This also implies that the receive FIFO will not yet contain the last received data. It is possible at that time to fill the TX FIFO again and read the receive FIFO entries which have been received. Note that there is no "receive FIFO full" interrupt as the number of entries received is always equal to the number of entries transmitted.

SPI is IDLE:
This interrupt will be asserted when the transmit FIFO is empty and the SPI block has finished all actions (including the CS-high time) By this time the receive FIFO will have all received data as well.

2.3.3 Long bit streams

The SPI module works in bursts of maximum 32 bits. Some SPI devices require data which is longer the 32 bits. To do this the user must make use of the two different data TX addresses: Tx data written to one address cause the CS to remain asserted. Tx data written to the other address cause the CS to be de-asserted at the end of the transmit cycle. So in order to exchange 96 bits you do the following:
Write the first two data words to one address, then write the third word to the other address.

2.3.4 SPI register details.

AUXSPI0/1_CNTL0 Register (0x7E21 5080,0x7E21 50C0)

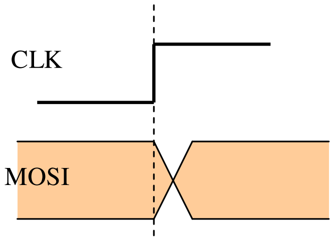
SYNOPSIS The AUXSPIx_CNTL0 register control many features of the SPI interfaces.

Bit(s)	Field Name	Description	Type	Reset
31:20	Speed	Sets the SPI clock speed. spi clk freq = system_clock_freq/2*(speed+1)	R/W	0
19:17	chip selects	The pattern output on the CS pins when active.	R/W	111
16	post-input mode	If set the SPI input works in post input mode. For details see text further down	R/W	0
15	Variable CS	If 1 the SPI takes the CS pattern and the data from the TX fifo If 0 the SPI takes the CS pattern from bits 17-19 of this register Set this bit only if also bit 14 (variable width) is set	R/W	0
14	Variable width	If 1 the SPI takes the shift length and the data from the TX fifo If 0 the SPI takes the shift length from bits 0-5 of this register	R/W	0
13:12	DOUT Hold time	Controls the extra DOUT hold time in system clock cycles. 00 : No extra hold time 01 : 1 system clock extra hold time 10 : 4 system clocks extra hold time 11 : 7 system clocks extra hold time	R/W	0
11	Enable	Enables the SPI interface. Whilst disabled the FIFOs can still be written to or read from This bit should be 1 during normal operation.	R/W	0
10	In rising	If 1 data is clocked in on the rising edge of the SPI clock If 0 data is clocked in on the falling edge of the SPI clock	R/W	0
9	Clear FIFOs	If 1 the receive and transmit FIFOs are held in reset (and thus flushed.) This bit should be 0 during normal operation.	R/W	0

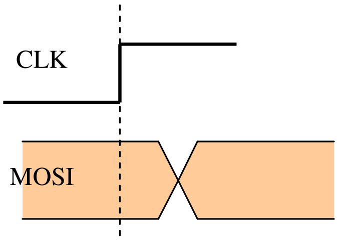
8	Out rising	If 1 data is clocked out on the rising edge of the SPI clock If 0 data is clocked out on the falling edge of the SPI clock	R/W	0
7	Invert SPI CLK	If 1 the 'idle' clock line state is high. If 0 the 'idle' clock line state is low.	R/W	0
6	Shift out MS bit first	If 1 the data is shifted out starting with the MS bit. (bit 15 or bit 11) If 0 the data is shifted out starting with the LS bit. (bit 0)	R/W	0
5:0	Shift length	Specifies the number of bits to shift This field is ignored when using 'variable shift' mode	R/W	0

Invert SPI CLK
Changing this bit will immediately change the polarity of the SPI clock output. It is recommended not to do this when also the CS is active as the connected devices will see this as a clock change.

DOUT hold time
Because the interface runs of fast silicon the MOSI hold time against the clock will be very short. This can cause considerable problems on SPI slaves. To make it easier for the slave to see the data the hold time of the MOSI out against the SPI clock out is programmable.



No hold time



With hold time

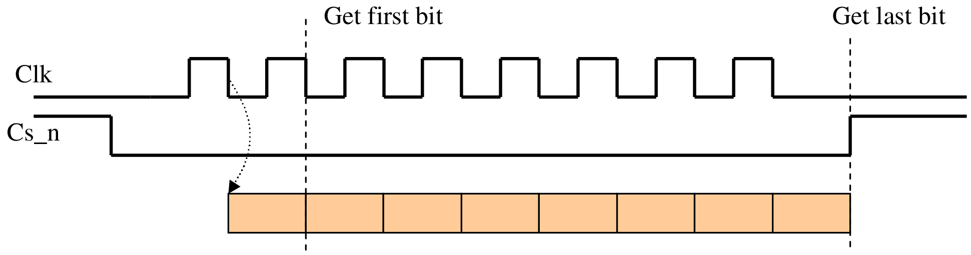
Variable width
In this mode the shift length is taken from the transmit FIFO. The transmit data bits 28:24 are used as shift length and the data bits 23:0 are the actual transmit data. If the option 'shift MS out first' is selected the first bit shifted out will be bit 23. The receive data will arrive as normal.

Variable CS
This mode is used together with the variable width mode. In this mode the CS pattern is taken from the transmit FIFO. The transmit data bits 31:29 are used as CS and the data bits 23:0 are the actual transmit data. This allows the CPU to write to different SPI devices without having to change the CS bits. However the data length is limited to 24 bits.

Post-input mode
Some rare SPI devices output data on the falling clock edge which then has to be picked up on the next falling clock edge. There are two problems with this:

1. The very first falling clock edge there is no valid data arriving.
2. After the last clock edge there is one more 'dangling' bit to pick up.

The post-input mode is specifically to deal with this sort of data. If the post-input mode bit is set, the data arriving at the first falling clock edge is ignored. Then after the last falling clock edge the CS remain asserted and after a full bit time the last data bit is picked up. The following figure shows this behaviour:



In this mode the CS will go high 1 full SPI clock cycle after the last clock edge. This guarantees a full SPI clock cycle time for the data to settle and arrive at the MISO input.

AUXSPI0/1_CNTL1 Register (0x7E21 5084,0x7E21 50C4)

SYNOPSIS The AUXSPIx_CNTL1 registers control more features of the SPI interfaces.

Bit(s)	Field Name	Description	Type	Reset
31:18	-	Reserved, write zero, read as don't care		
10:8	CS high time	Additional SPI clock cycles where the CS is high.	R/W	0
7	TX empty IRQ	If 1 the interrupt line is high when the transmit FIFO is empty	R/W	0
6	Done IRQ	If 1 the interrupt line is high when the interface is idle	R/W	0
5:2	-	Reserved, write zero, read as don't care		
1	Shift in MS bit first	If 1 the data is shifted in starting with the MS bit. (bit 15) If 0 the data is shifted in starting with the LS bit. (bit 0)	R/W	0
0	Keep input	If 1 the receiver shift register is NOT cleared. Thus new data is concatenated to old data. If 0 the receiver shift register is cleared before each transaction.	R/W	0

Keep input
Setting the 'Keep input' bit will make that the input shift register is not cleared between transactions. However the contents of the shift register is still written to the receive FIFO at the end of each transaction. E.g. if you receive two 8 bit values 0x81 followed by 0x46 the receive FIFO will contain: 0x0081 in the first entry and 0x8146 in the second entry. This mode may save CPU time concatenating bits (4 bits followed by 12 bits).

CS high time
The SPI CS will always be high for at least 1 SPI clock cycle. Some SPI devices need more time to process the data. This field will set a longer CS-high time. So the actual CS high time is (CS_high_time + 1) (In SPI clock cycles).

Interrupts
The SPI block has two interrupts: TX FIFO is empty, SPI is Idle.

TX FIFO is empty:
This interrupt will be asserted as soon as the last entry has been read from the transmit FIFO. At that time the interface will still be busy shifting out that data. This also implies that the receive FIFO will not yet contain the last received data.

It is possible at that time to fill the TX FIFO again and read the receive FIFO entries which have been received. There is a RX FIFO level field which tells exactly how many words are in the receive FIFO. In general at that time the receive FIFO should contain the number of Tx items minus one (the last one still being received). Note that there is no "receive FIFO full" interrupt or "receive FIFO overflow" flag as the number of entries received can never be more then the number of entries transmitted.

AUX is IDLE:
This interrupt will be asserted when the module has finished all activities, including waiting the minimum CS high time. This guarantees that any receive data will be available and `transparent' changes can be made to the configuration register (e.g. inverting the SPI clock polarity).

AUXSPI0/1_STAT Register (0x7E21 5088,0x7E21 50C8)

SYNOPSIS The AUXSPIx_STAT registers show the status of the SPI interfaces.

Bit(s)	Field Name	Description	Type	Reset
31:24	TX FIFO level	The number of data units in the transmit data FIFO	R/W	0
23:12	RX FIFO level	The number of data units in the receive data FIFO.	R/W	0
11:5	-	Reserved, write zero, read as don't care		
4	TX Full	If 1 the transmit FIFO is full If 0 the transmit FIFO can accept at least 1 data unit.	R/W	0
3	TX Empty	If 1 the transmit FIFO is empty If 0 the transmit FIFO holds at least 1 data unit.	R/W	0
2	RX Empty	If 1 the receiver FIFO is empty If 0 the receiver FIFO holds at least 1 data unit.	R/W	0
6	Busy	Indicates the module is busy transferring data.	R/W	0
5:0	Bit count	The number of bits still to be processed. Starts with 'shift-length' and counts down.	R/W	0

Busy
This status bit indicates if the module is busy. It will be clear when the TX FIFO is empty and the module has finished all activities, including waiting the minimum CS high time.

AUXSPI0/1_PEEK Register (0x7E21 508C,0x7E21 50CC)

SYNOPSIS The AUXSPIx_PEEK registers show received data of the SPI interfaces.

Bit(s)	Field Name	Description	Type	Reset
31:16	-	Reserved, write zero, read as don't care		
15:0	Data	Reads from this address will show the top entry from the receive FIFO, but the data is not taken from the FIFO. This provides a means of inspecting the data but not removing it from the FIFO.	RO	0

AUXSPI0/1_IO Register
(0x7E21 50A0-0x7E21 50AC
0x7E21 50E0-0x7E21 50EC)

SYNOPSIS The AUXSPIx_IO registers are the primary data port of the SPI interfaces
These four addresses all write to the same FIFO.

Writing to any of these addresses causes the SPI CS_n pins to be de-asserted at the end of the access

Bit(s)	Field Name	Description	Type	Reset
31:16	-	Reserved, write zero, read as don't care		
15:0	Data	Writes to this address range end up in the transmit FIFO. Data is lost when writing whilst the transmit FIFO is full. Reads from this address will take the top entry from the receive FIFO. Reading whilst the receive FIFO is will return the last data received.	R/W	0

AUXSPI0/1_TXHOLD Register
(0x7E21 50B0-0x7E21 50BC
0x7E21 50F0-0x7E21 50FC)

SYNOPSIS The AUXSPIx_TXHOLD registers are the extended CS port of the SPI interfaces These four addresses all write to the same FIFO.

Writing to these addresses causes the SPI CS_n pins to remain asserted at the end of the access

Bit(s)	Field Name	Description	Type	Reset
31:16	-	Reserved, write zero, read as don't care		
15:0	Data	Writes to this address range end up in the transmit FIFO. Data is lost when writing whilst the transmit FIFO is full. Reads from this address will take the top entry from the receive FIFO. Reading whilst the receive FIFO is will return the last data received.	R/W	0

3 BSC

3.1 Introduction

The Broadcom Serial Controller (BSC) controller is a master, fast-mode (400Kb/s) BSC controller. The Broadcom Serial Control bus is a proprietary bus compliant with the Philips® I2C bus/interface version 2.1 January 2000.

- I^2C single master only operation (supports clock stretching wait states)
- Both 7-bit and 10-bit addressing is supported.
  - Timing completely software controllable via registers

3.2 Register View

The BSC controller has eight memory-mapped registers. All accesses are assumed to be 32-bit. Note that the BSC2 master is used dedicated with the HDMI interface and should not be accessed by user programs.

There are three BSC masters inside BCM. The register addresses starts from

- BSC0: 0x7E20_5000
- BSC1: 0x7E80_4000
- BSC2 : 0x7E80_5000

The table below shows the address of I^2C interface where the address is an offset from one of the three base addreses listed above.

I2C Address Map			
Address Offset	**Register Name**	**Description**	**Size**
0x0	C	Control	32
0x4	S	Status	32
0x8	DLEN	Data Length	32
0xc	A	Slave Address	32
0x10	FIFO	Data FIFO	32
0x14	DIV	Clock Divider	32
0x18	DEL	Data Delay	32

0x1c	CLKT	Clock Stretch Timeout	32

C Register

Synopsis The control register is used to enable interrupts, clear the FIFO, define a read or write operation and start a transfer.
The READ field specifies the type of transfer.
The CLEAR field is used to clear the FIFO. Writing to this field is a one-shot operation which will always read back as zero. The CLEAR bit can set at the same time as the start transfer bit, and will result in the FIFO being cleared just prior to the start of transfer. Note that clearing the FIFO during a transfer will result in the transfer being aborted.
The ST field starts a new BSC transfer. This has a one shot action, and so the bit will always read back as 0 .
The INTD field enables interrupts at the end of a transfer the DONE condition. The interrupt remains active until the DONE condition is cleared by writing a 1 to the I2CS.DONE field. Writing a 0 to the INTD field disables interrupts on DONE.
The INTT field enables interrupts whenever the FIFO is or more empty and needs writing (i.e. during a write transfer) - the TXW condition. The interrupt remains active until the TXW condition is cleared by writing sufficient data to the FIFO to complete the transfer. Writing a 0 to the INTT field disables interrupts on TXW.
The INTR field enables interrupts whenever the FIFO is or more full and needs reading (i.e. during a read transfer) - the RXR condition. The interrupt remains active until the RXW condition is cleared by reading sufficient data from the RX FIFO. Writing a 0 to the INTR field disables interrupts on RXR.
The I2CEN field enables BSC operations. If this bit is 0 then transfers will not be performed. All register accesses are still permitted however.

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved** - Write as 0, read as don't care*		
15	I2CEN	I2C Enable 0 = BSC controller is disabled 1 = BSC controller is enabled	RW	0x0
14:11		***Reserved** - Write as 0, read as don't care*		
10	INTR	INTR Interrupt on RX 0 = Don t generate interrupts on RXR condition. 1 = Generate interrupt while RXR = 1.	RW	0x0
9	INTT	INTT Interrupt on TX 0 = Don t generate interrupts on TXW condition. 1 = Generate interrupt while TXW = 1.	RW	0x0

8	INTD	INTD Interrupt on DONE 0 = Don t generate interrupts on DONE condition. 1 = Generate interrupt while DONE = 1.	RW	0x0
7	ST	ST Start Transfer 0 = No action. 1 = Start a new transfer. One shot operation. Read back as 0.	RW	0x0
6		***Reserved*** *- Write as 0, read as don't care*		
5:4	CLEAR	CLEAR FIFO Clear 00 = No action. x1 = Clear FIFO. One shot operation. 1x = Clear FIFO. One shot operation. If CLEAR and ST are both set in the same operation, the FIFO is cleared before the new frame is started. Read back as 0. Note: 2 bits are used to maintain compatibility to previous version.	RW	0x0
3:1		***Reserved*** *- Write as 0, read as don't care*		
0	READ	READ Read Transfer 0 = Write Packet Transfer. 1 = Read Packet Transfer.	RW	0x0

S Register

Synopsis The status register is used to record activity status, errors and interrupt requests.
The TA field indicates the activity status of the BSC controller. This read-only field returns a 1 when the controller is in the middle of a transfer and a 0 when idle.
The DONE field is set when the transfer completes. The DONE condition can be used with I2CC.INTD to generate an interrupt on transfer completion. The DONE field is reset by writing a 1 , writing a 0 to the field has no effect.
The read-only TXW bit is set during a write transfer and the FIFO is less than full and needs writing. Writing sufficient data (i.e. enough data to either fill the FIFO more than full or complete the transfer) to the FIFO will clear the field. When the I2CC.INTT control bit is set, the TXW condition can be used to generate an interrupt to write more data to the FIFO to complete the current transfer. If the I2C controller runs out of data to send, it will wait for more data to be written into the FIFO.
The read-only RXR field is set during a read transfer and the FIFO is or more full and needs reading. Reading sufficient data to bring the depth below will clear the field. When I2CC.INTR control bit is set, the RXR condition can be used to generate an interrupt to read data from the FIFO before it becomes full. In the event that the FIFO does become full, all I2C operations will stall until data is removed from the FIFO.
The read-only TXD field is set when the FIFO has space for at least one byte of data. TXD is clear when the FIFO is full. The TXD field can be used to check that the FIFO can accept data before any is written. Any writes to a full TX FIFO will be ignored.
The read-only RXD field is set when the FIFO contains at least one byte of data. RXD is cleared when the FIFO becomes empty. The RXD field can be used to check that the FIFO contains data before reading. Reading from an empty FIFO will return invalid data.
The read-only TXE field is set when the FIFO is empty. No further data will be transmitted until more data is written to the FIFO.
The read-only RXF field is set when the FIFO is full. No more clocks will be generated until space is available in the FIFO to receive more data.
The ERR field is set when the slave fails to acknowledge either its address or a data byte written to it. The ERR field is reset by writing a 1 , writing a 0 to the field has no effect.
The CLKT field is set when the slave holds the SCL signal high for too long (clock stretching). The CLKT field is reset by writing a 1 , writing a 0 to the field has no effect.

Bit(s)	Field Name	Description	Type	Reset
31:10		***Reserved** - Write as 0, read as don't care*		
9	CLKT	CLKT Clock Stretch Timeout 0 = No errors detected. 1 = Slave has held the SCL signal low (clock stretching) for longer and that specified in the I2CCLKT register Cleared by writing 1 to the field.	RW	0x0
8	ERR	ERR ACK Error 0 = No errors detected. 1 = Slave has not acknowledged its address. Cleared by writing 1 to the field.	RW	0x0
7	RXF	RXF - FIFO Full 0 = FIFO is not full. 1 = FIFO is full. If a read is underway, no further serial data will be received until data is read from FIFO.	RO	0x0

6	TXE	TXE - FIFO Empty 0 = FIFO is not empty. 1 = FIFO is empty. If a write is underway, no further serial data can be transmitted until data is written to the FIFO.	RO	0x1
5	RXD	RXD - FIFO contains Data 0 = FIFO is empty. 1 = FIFO contains at least 1 byte. Cleared by reading sufficient data from FIFO.	RO	0x0
4	TXD	TXD - FIFO can accept Data 0 = FIFO is full. The FIFO cannot accept more data. 1 = FIFO has space for at least 1 byte.	RO	0x1
3	RXR	RXR - FIFO needs Reading (full) 0 = FIFO is less than full and a read is underway. 1 = FIFO is or more full and a read is underway. Cleared by reading sufficient data from the FIFO.	RO	0x0
2	TXW	TXW - FIFO needs Writing (full) 0 = FIFO is at least full and a write is underway (or sufficient data to send). 1 = FIFO is less then full and a write is underway. Cleared by writing sufficient data to the FIFO.	RO	0x0
1	DONE	DONE Transfer Done 0 = Transfer not completed. 1 = Transfer complete. Cleared by writing 1 to the field.	RW	0x0
0	TA	TA Transfer Active 0 = Transfer not active. 1 = Transfer active.	RO	0x0

DLEN Register

Synopsis The data length register defines the number of bytes of data to transmit or receive in the I2C transfer. Reading the register gives the number of bytes remaining in the current transfer.

The DLEN field specifies the number of bytes to be transmitted/received. Reading the DLEN field when a transfer is in progress (TA = 1) returns the number of bytes still to be transmitted or received. Reading the DLEN field when the transfer has just completed (DONE = 1) returns zero as there are no more bytes to transmit or receive. Finally, reading the DLEN field when TA = 0 and DONE = 0 returns the last value written. The DLEN field can be left over multiple transfers.

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved*** *- Write as 0, read as don't care*		

15:0	DLEN	Data Length. Writing to DLEN specifies the number of bytes to be transmitted/received. Reading from DLEN when TA = 1 or DONE = 1, returns the number of bytes still to be transmitted or received. Reading from DLEN when TA = 0 and DONE = 0, returns the last DLEN value written. DLEN can be left over multiple packets.	RW	0x0

A Register

Synopsis The slave address register specifies the slave address and cycle type. The address register can be left across multiple transfers
The ADDR field specifies the slave address of the I2C device.

Bit(s)	Field Name	Description	Type	Reset
31:7		***Reserved*** *- Write as 0, read as don't care*		
6:0	ADDR	Slave Address.	RW	0x0

FIFO Register

Synopsis The Data FIFO register is used to access the FIFO. Write cycles to this address place data in the 16-byte FIFO, ready to transmit on the BSC bus. Read cycles access data received from the bus.
Data writes to a full FIFO will be ignored and data reads from an empty FIFO will result in invalid data. The FIFO can be cleared using the I2CC.CLEAR field.
The DATA field specifies the data to be transmitted or received.

Bit(s)	Field Name	Description	Type	Reset
31:8		***Reserved*** *- Write as 0, read as don't care*		
7:0	DATA	Writes to the register write transmit data to the FIFO. Reads from register reads received data from the FIFO.	RW	0x0

DIV Register

Synopsis The clock divider register is used to define the clock speed of the BSC peripheral. The CDIV field specifies the core clock divider used by the BSC.

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved** - Write as 0, read as don't care*		
15:0	CDIV	Clock Divider SCL = core clock / CDIV Where core_clk is nominally 150 MHz. If CDIV is set to 0, the divisor is 32768. CDIV is always rounded down to an even number. The default value should result in a 100 kHz I2C clock frequency.	RW	0x5dc

DEL Register

Synopsis The data delay register provides fine control over the sampling/launch point of the data.
The REDL field specifies the number core clocks to wait after the rising edge before sampling the incoming data.
The FEDL field specifies the number core clocks to wait after the falling edge before outputting the next data bit.
Note: Care must be taken in choosing values for FEDL and REDL as it is possible to cause the BSC master to malfunction by setting values of CDIV/2 or greater. Therefore the delay values should always be set to less than CDIV/2.

Bit(s)	Field Name	Description	Type	Reset
31:16	FEDL	FEDL Falling Edge Delay Number of core clock cycles to wait after the falling edge of SCL before outputting next bit of data.	RW	0x30
15:0	REDL	REDL Rising Edge Delay Number of core clock cycles to wait after the rising edge of SCL before reading the next bit of data.	RW	0x30

CLKT Register

Synopsis The clock stretch timeout register provides a timeout on how long the master waits for the slave to stretch the clock before deciding that the slave has hung.
The TOUT field specifies the number I2C SCL clocks to wait after releasing SCL high and finding that the SCL is still low before deciding that the slave is not responding and moving the I2C machine forward. When a timeout occurs, the I2CS.CLKT bit is set.
Writing 0x0 to TOUT will result in the Clock Stretch Timeout being disabled.

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved** - Write as 0, read as don't care*		
15:0	TOUT	TOUT Clock Stretch Timeout Value Number of SCL clock cycles to wait after the rising edge of SCL before deciding that the slave is not responding.	RW	0x40

3.3 10 Bit Addressing

10 Bit addressing is an extension to the standard 7-bit addressing mode. This section describes in detail how to read/write using 10-bit addressing with this I2C controller.

10-bit addressing is compatible with, and can be combined with, 7 bit addressing. Using 10 bits for addressing exploits the reserved combination 1111 0xx for the first byte following a START (S) or REPEATED START (Sr) condition.

The 10 bit slave address is formed from the first two bytes following a S or Sr condition.

The first seven bits of the first byte are the combination 11110XX of which the last two bits (XX) are the two *most significant* bits of the 10-bit address. The eighth bit of the first byte is the R/W bit. If the R/W bit is '0' (write) then the following byte contains the remaining 8 bits of the 10-bit address. If the R/W bit is '1' then the next byte contains data transmitted from the slave to the master.

Writing

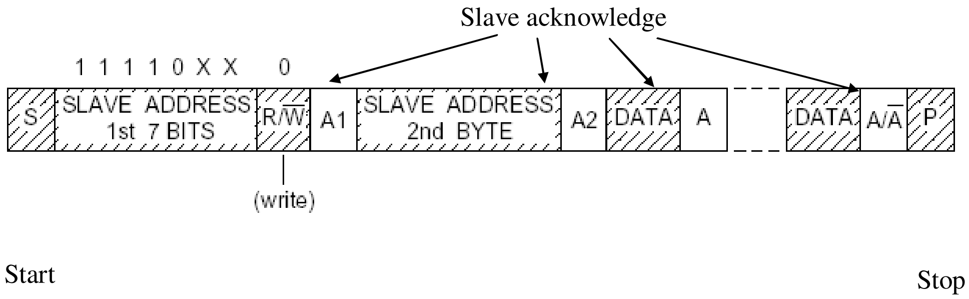


Figure 3-1 Write to a slave with 10 bit address

Figure 3-1 shows a write to a slave with a 10-bit address, to perform this using the controller one must do the following:

Assuming we are in the 'stop' state: (and the FIFO is empty)

1. Write the number of data bytes to written (plus one) to the I2CDLEN register.
2. Write 'XXXXXXXX' to the FIFO where 'XXXXXXXX' are the least 8 significant bits of the 10-bit slave address.
3. Write other data to be transmitted to the FIFO.
4. Write '11110XX' to Slave Address Register where 'XX' are the two most significant bits of the 10-bit address. Set I2CC.READ = 0 and I2CC.ST = 1, this will start a write transfer.

Reading

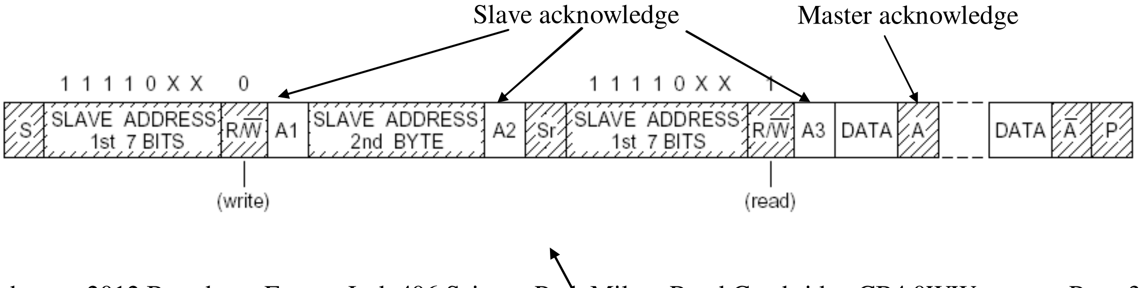


Figure 3-2 Read from slave with 10 bit address

Figure 3-2 shows how a read from a slave with a 10-bit address is performed. Following is the procedure for performing a read using the controller:

1. Write 1 to the I2CDLEN register.
2. Write ‘XXXXXXXX’ to the FIFO where ‘XXXXXXXX’ are the least 8 significant bits of the 10-bit slave address.
3. Write ‘11110XX’ to the Slave Address Register where ‘XX’ are the two most significant bits of the 10-bit address. Set I2CC.READ = 0 and I2CC.ST = 1, this will start a write transfer.
4. Poll the I2CS.TA bit, waiting for the transfer has started.
5. Write the number of data bytes to read to the I2CDLEN register.
6. Set I2CC.READ = 1 and I2CC.ST = 1, this will send the repeat start bit, new slave address and R/W bit (which is ‘1’) initiating the read.

4 DMA Controller

4.1 Overview

The majority of hardware pipelines and peripherals within the BCM2835 are bus masters, enabling them to efficiently satisfy their own data requirements. This reduces the requirements of the DMA controller to block-to-block memory transfers and supporting some of the simpler peripherals. In addition, the DMA controller provides a *read only* prefetch mode to allow data to be brought into the L2 cache in anticipation of its later use.

Beware that the DMA controller is direcly connected to the peripherals. Thus the DMA controller must be set-up to use the Physical (harware) addresses of the peripherals.

The BCM2835 DMA Controller provides a total of 16 DMA channels. Each channel operates independently from the others and is internally arbitrated onto one of the 3 system busses. This means that the amount of bandwidth that a DMA channel may consume can be controlled by the arbiter settings.

Each DMA channel operates by loading a *Control Block* (CB) data structure from memory into internal registers. The *Control Block* defines the required DMA operation. Each *Control Block* can point to a further *Control Block* to be loaded and executed once the operation described in the current *Control Block* has completed. In this way a linked list of *Control Blocks* can be constructed in order to execute a sequence of DMA operations without software intervention.

The DMA supports AXI read bursts to ensure efficient external SDRAM use. The DMA control block contains a burst parameter which indicates the required burst size of certain memory transfers. In general the DMA doesn't do write bursts, although wide writes will be done in 2 beat bursts if possible.

Memory-to-Peripheral transfers can be paced by a *Data Request* (DREQ) signal which is generated by the peripheral. The DREQ signal is level sensitive and controls the DMA by gating its AXI bus requests.

A peripheral can also provide a *Panic* signal alongside the DREQ to indicate that there is an imminent danger of FIFO underflow or overflow or similar critical situation. The *Panic* is used to select the AXI apriority level which is then passed out onto the AXI bus so that it can be used to influence arbitration in the rest of the system.

The allocation of peripherals to DMA channels is programmable.

The DMA can deal with byte aligned transfers and will minimise bus traffic by buffering and packing misaligned accesses.

Each DMA channel can be fully disabled via a top level power register to save power.

4.2 DMA Controller Registers

The DMA Controller is comprised of several identical DMA Channels depending upon the required configuration. Each individual DMA channel has an identical register map (although LITE channels have less functionality and hence less registers).

DMA Channel 0 is located at the address of 0x7E007000, Channel 1 at 0x7E007100, Channel 2 at 0x7E007200 and so on. Thus adjacent DMA Channels are offset by `0x100`.

DMA Channel 15 however, is physically removed from the other DMA Channels and so has a different address base of 0x7EE05000.

DMA Channel Offsets		
DMA Channels 0 – 14 Register Set Offsets from DMA0_BASE		
0x000		*DMA Channel 0 Register Set*
0x100		*DMA Channel 1 Register Set*
0x200		*DMA Channel 2 Register Set*
0x300		*DMA Channel 3 Register Set*
0x400		*DMA Channel 4 Register Set*
0x500		*DMA Channel 5 Register Set*
0x600		*DMA Channel 6 Register Set*
0x700		*DMA Channel 7 Register Set*
0x800		*DMA Channel 8 Register Set*
0x900		*DMA Channel 9 Register Set*
0xa00		*DMA Channel 10 Register Set*
0xb00		*DMA Channel 11 Register Set*
0xc00		*DMA Channel 12 Register Set*
0xd00		*DMA Channel 13 Register Set*
0xe00		*DMA Channel 14 Register Set*
DMA Channel 15 Register Set Offset from DMA15_BASE		
0x000		*DMA Channel 15 Register Set*

Table 4-1 – DMA Controller Register Address Map

4.2.1 DMA Channel Register Address Map

Each DMA channel has an identical register map, only the base address of each channel is different.

There is a global enable register at the top of the Address map that can disable each DMA for powersaving.

Only three registers in each channels register set are directly writeable (CS, CONBLK_AD *and DEBUG*). The other registers (TI, SOURCE_AD, DEST_AD, TXFR_LEN, STRIDE & NEXTCONBK), are automatically loaded from a *Control Block* data structure held in external memory.

4.2.1.1 Control Block Data Structure

Control Blocks (CB) are 8 words (256 bits) in length and must start at a 256-bit aligned address. The format of the CB data structure in memory, is shown below.

Each 32 bit word of the control block is automatically loaded into the corresponding 32 bit DMA control block register at the start of a DMA transfer. The descriptions of these registers also defines the corresponding bit locations in the CB data structure in memory.

32-bit Word Offset	Description	Associated Read-Only Register
0	Transfer Information	TI
1	Source Address	SOURCE_AD
2	Destination Address	DEST_AD
3	Transfer Length	TXFR_LEN
4	2D Mode Stride	STRIDE
5	Next Control Block Address	NEXTCONBK
6-7	*Reserved – set to zero.*	*N/A*

Table 4-2 – DMA Control Block Definition

The DMA is started by writing the address of a CB structure into the CONBLK_AD register and then setting the ACTIVE bit. The DMA will fetch the CB from the address set in the SCB_ADDR field of this reg and it will load it into the *read-only* registers described below. It will then begin a DMA transfer according to the information in the CB.

When it has completed the current DMA transfer (length => 0) the DMA will update the CONBLK_AD register with the contents of the NEXTCONBK register, fetch a new CB from that address, and start the whole procedure once again.

The DMA will stop (and clear the ACTIVE bit) when it has completed a DMA transfer and the NEXTCONBK register is set to `0x0000_0000`. It will load this value into the CONBLK_AD reg and then stop.

Most of the control block registers cannot be written to directly as they loaded automatically from memory. They can be read to provide status information, and to indicate the progress of the current DMA transfer. The value loaded into the NEXTCONBK register can be overwritten so that the linked list of Control Block data structures can be dynamically altered. However it is only safe to do this when the DMA is paused.

4.2.1.2 Register Map

DMA Address Map			
Address Offset	**Register Name**	**Description**	**Size**
0x0	0_CS	DMA Channel 0 Control and Status	32
0x4	0_CONBLK_AD	DMA Channel 0 Control Block Address	32
0x8	0_TI	DMA Channel 0 CB Word 0 (Transfer Information)	32
0xc	0_SOURCE_AD	DMA Channel 0 CB Word 1 (Source Address)	32
0x10	0_DEST_AD	DMA Channel 0 CB Word 2 (Destination Address)	32
0x14	0_TXFR_LEN	DMA Channel 0 CB Word 3 (Transfer Length)	32
0x18	0_STRIDE	DMA Channel 0 CB Word 4 (2D Stride)	32
0x1c	0_NEXTCONBK	DMA Channel 0 CB Word 5 (Next CB Address)	32
0x20	0_DEBUG	DMA Channel 0 Debug	32
0x100	1_CS	DMA Channel 1 Control and Status	32
0x104	1_CONBLK_AD	DMA Channel 1 Control Block Address	32
0x108	1_TI	DMA Channel 1 CB Word 0 (Transfer Information)	32
0x10c	1_SOURCE_AD	DMA Channel 1 CB Word 1 (Source Address)	32
0x110	1_DEST_AD	DMA Channel 1 CB Word 2 (Destination Address)	32
0x114	1_TXFR_LEN	DMA Channel 1 CB Word 3 (Transfer Length)	32

0x118	1_STRIDE	DMA Channel 1 CB Word 4 (2D Stride)	32
0x11c	1_NEXTCONBK	DMA Channel 1 CB Word 5 (Next CB Address)	32
0x120	1_DEBUG	DMA Channel 1 Debug	32
0x200	2_CS	DMA Channel 2 Control and Status	32
0x204	2_CONBLK_AD	DMA Channel 2 Control Block Address	32
0x208	2_TI	DMA Channel 2 CB Word 0 (Transfer Information)	32
0x20c	2_SOURCE_AD	DMA Channel 2 CB Word 1 (Source Address)	32
0x210	2_DEST_AD	DMA Channel 2 CB Word 2 (Destination Address)	32
0x214	2_TXFR_LEN	DMA Channel 2 CB Word 3 (Transfer Length)	32
0x218	2_STRIDE	DMA Channel 2 CB Word 4 (2D Stride)	32
0x21c	2_NEXTCONBK	DMA Channel 2 CB Word 5 (Next CB Address)	32
0x220	2_DEBUG	DMA Channel 2 Debug	32
0x300	3_CS	DMA Channel 3 Control and Status	32
0x304	3_CONBLK_AD	DMA Channel 3 Control Block Address	32
0x308	3_TI	DMA Channel 3 CB Word 0 (Transfer Information)	32
0x30c	3_SOURCE_AD	DMA Channel 3 CB Word 1 (Source Address)	32
0x310	3_DEST_AD	DMA Channel 3 CB Word 2 (Destination Address)	32
0x314	3_TXFR_LEN	DMA Channel 3 CB Word 3 (Transfer Length)	32
0x318	3_STRIDE	DMA Channel 3 CB Word 4 (2D Stride)	32
0x31c	3_NEXTCONBK	DMA Channel 3 CB Word 5 (Next CB Address)	32
0x320	3_DEBUG	DMA Channel 0 Debug	32

0x400	4_CS	DMA Channel 4 Control and Status	32
0x404	4_CONBLK_AD	DMA Channel 4 Control Block Address	32
0x408	4_TI	DMA Channel 4 CB Word 0 (Transfer Information)	32
0x40c	4_SOURCE_AD	DMA Channel 4 CB Word 1 (Source Address)	32
0x410	4_DEST_AD	DMA Channel 4 CB Word 2 (Destination Address)	32
0x414	4_TXFR_LEN	DMA Channel 4 CB Word 3 (Transfer Length)	32
0x418	4_STRIDE	DMA Channel 4 CB Word 4 (2D Stride)	32
0x41c	4_NEXTCONBK	DMA Channel 4 CB Word 5 (Next CB Address)	32
0x420	4_DEBUG	DMA Channel 0 Debug	32
0x500	5_CS	DMA Channel 5 Control and Status	32
0x504	5_CONBLK_AD	DMA Channel 5 Control Block Address	32
0x508	5_TI	DMA Channel 5 CB Word 0 (Transfer Information)	32
0x50c	5_SOURCE_AD	DMA Channel 5 CB Word 1 (Source Address)	32
0x510	5_DEST_AD	DMA Channel 5 CB Word 2 (Destination Address)	32
0x514	5_TXFR_LEN	DMA Channel 5 CB Word 3 (Transfer Length)	32
0x518	5_STRIDE	DMA Channel 5 CB Word 4 (2D Stride)	32
0x51c	5_NEXTCONBK	DMA Channel 5 CB Word 5 (Next CB Address)	32
0x520	5_DEBUG	DMA Channel 5 Debug	32
0x600	6_CS	DMA Channel 6 Control and Status	32
0x604	6_CONBLK_AD	DMA Channel 6 Control Block Address	32
0x608	6_TI	DMA Channel 6 CB Word 0 (Transfer Information)	32

0x60c	6_SOURCE_AD	DMA Channel 6 CB Word 1 (Source Address)	32
0x610	6_DEST_AD	DMA Channel 6 CB Word 2 (Destination Address)	32
0x614	6_TXFR_LEN	DMA Channel 6 CB Word 3 (Transfer Length)	32
0x618	6_STRIDE	DMA Channel 6 CB Word 4 (2D Stride)	32
0x61c	6_NEXTCONBK	DMA Channel 6 CB Word 5 (Next CB Address)	32
0x620	6_DEBUG	DMA Channel 6 Debug	32
0x700	7_CS	DMA Channel 7 Control and Status	32
0x704	7_CONBLK_AD	DMA Channel 7 Control Block Address	32
0x708	7_TI	DMA Channel 7 CB Word 0 (Transfer Information)	32
0x70c	7_SOURCE_AD	DMA Channel 7 CB Word 1 (Source Address)	32
0x710	7_DEST_AD	DMA Channel 7 CB Word 2 (Destination Address)	32
0x714	7_TXFR_LEN	DMA Channel 7 CB Word 3 (Transfer Length)	32
0x71c	7_NEXTCONBK	DMA Channel 7 CB Word 5 (Next CB Address)	32
0x720	7_DEBUG	DMA Channel 7 Debug	32
0x800	8_CS	DMA Channel 8 Control and Status	32
0x804	8_CONBLK_AD	DMA Channel 8 Control Block Address	32
0x808	8_TI	DMA Channel 8 CB Word 0 (Transfer Information)	32
0x80c	8_SOURCE_AD	DMA Channel 8 CB Word 1 (Source Address)	32
0x810	8_DEST_AD	DMA Channel 8 CB Word 2 (Destination Address)	32
0x814	8_TXFR_LEN	DMA Channel 8 CB Word 3 (Transfer Length)	32
0x81c	8_NEXTCONBK	DMA Channel 8 CB Word 5 (Next CB Address)	32

0x820	8_DEBUG	DMA Channel 8 Debug	32
0x900	9_CS	DMA Channel 9 Control and Status	32
0x904	9_CONBLK_AD	DMA Channel 9 Control Block Address	32
0x908	9_TI	DMA Channel 9 CB Word 0 (Transfer Information)	32
0x90c	9_SOURCE_AD	DMA Channel 9 CB Word 1 (Source Address)	32
0x910	9_DEST_AD	DMA Channel 9 CB Word 2 (Destination Address)	32
0x914	9_TXFR_LEN	DMA Channel 9 CB Word 3 (Transfer Length)	32
0x91c	9_NEXTCONBK	DMA Channel 9 CB Word 5 (Next CB Address)	32
0x920	9_DEBUG	DMA Channel 9 Debug	32
0xa00	10_CS	DMA Channel 10 Control and Status	32
0xa04	10_CONBLK_AD	DMA Channel 10 Control Block Address	32
0xa08	10_TI	DMA Channel 10 CB Word 0 (Transfer Information)	32
0xa0c	10_SOURCE_AD	DMA Channel 10 CB Word 1 (Source Address)	32
0xa10	10_DEST_AD	DMA Channel 10 CB Word 2 (Destination Address)	32
0xa14	10_TXFR_LEN	DMA Channel 10 CB Word 3 (Transfer Length)	32
0xa1c	10_NEXTCONBK	DMA Channel 10 CB Word 5 (Next CB Address)	32
0xa20	10_DEBUG	DMA Channel 10 Debug	32
0xb00	11_CS	DMA Channel 11 Control and Status	32
0xb04	11_CONBLK_AD	DMA Channel 11 Control Block Address	32
0xb08	11_TI	DMA Channel 11 CB Word 0 (Transfer Information)	32
0xb0c	11_SOURCE_AD	DMA Channel 11 CB Word 1 (Source Address)	32

0xb10	11_DEST_AD	DMA Channel 11 CB Word 2 (Destination Address)	32
0xb14	11_TXFR_LEN	DMA Channel 11 CB Word 3 (Transfer Length)	32
0xb1c	11_NEXTCONBK	DMA Channel 11 CB Word 5 (Next CB Address)	32
0xb20	11_DEBUG	DMA Channel 11 Debug	32
0xc00	12_CS	DMA Channel 12 Control and Status	32
0xc04	12_CONBLK_AD	DMA Channel 12 Control Block Address	32
0xc08	12_TI	DMA Channel 12 CB Word 0 (Transfer Information)	32
0xc0c	12_SOURCE_AD	DMA Channel 12 CB Word 1 (Source Address)	32
0xc10	12_DEST_AD	DMA Channel 12 CB Word 2 (Destination Address)	32
0xc14	12_TXFR_LEN	DMA Channel 12 CB Word 3 (Transfer Length)	32
0xc1c	12_NEXTCONBK	DMA Channel 12 CB Word 5 (Next CB Address)	32
0xc20	12_DEBUG	DMA Channel 12 Debug	32
0xd00	13_CS	DMA Channel 13 Control and Status	32
0xd04	13_CONBLK_AD	DMA Channel 13 Control Block Address	32
0xd08	13_TI	DMA Channel 13 CB Word 0 (Transfer Information)	32
0xd0c	13_SOURCE_AD	DMA Channel 13 CB Word 1 (Source Address)	32
0xd10	13_DEST_AD	DMA Channel 13 CB Word 2 (Destination Address)	32
0xd14	13_TXFR_LEN	DMA Channel 13 CB Word 3 (Transfer Length)	32
0xd1c	13_NEXTCONBK	DMA Channel 13 CB Word 5 (Next CB Address)	32
0xd20	13_DEBUG	DMA Channel 13 Debug	32
0xe00	14_CS	DMA Channel 14 Control and Status	32

0xe04	14_CONBLK_AD	DMA Channel 14 Control Block Address	32
0xe08	14_TI	DMA Channel 14 CB Word 0 (Transfer Information)	32
0xe0c	14_SOURCE_AD	DMA Channel 14 CB Word 1 (Source Address)	32
0xe10	14_DEST_AD	DMA Channel 14 CB Word 2 (Destination Address)	32
0xe14	14_TXFR_LEN	DMA Channel 14 CB Word 3 (Transfer Length)	32
0xe1c	14_NEXTCONBK	DMA Channel 14 CB Word 5 (Next CB Address)	32
0xe20	14_DEBUG	DMA Channel 14 Debug	32
0xfe0	INT_STATUS	Interrupt status of each DMA channel	32
0xff0	ENABLE	Global enable bits for each DMA channel	32

0_CS 1_CS 2_CS 3_CS 4_CS 5_CS 6_CS 7_CS 8_CS 9_CS 10_CS 11_CS 12_CS 13_CS 14_CS Register

Synopsis DMA Control And Status register contains the main control and status bits for this DMA channel.

Bit(s)	Field Name	Description	Type	Reset
31	RESET	DMA Channel Reset Writing a 1 to this bit will reset the DMA. The bit cannot be read, and will self clear.	W1SC	0x0
30	ABORT	Abort DMA Writing a 1 to this bit will abort the current DMA CB. The DMA will load the next CB and attempt to continue. The bit cannot be read, and will self clear.	W1SC	0x0
29	DISDEBUG	Disable debug pause signal When set to 1, the DMA will not stop when the debug pause signal is asserted.	RW	0x0

28	WAIT_FOR_OUTSTANDING_WRITES	Wait for outstanding writes When set to 1, the DMA will keep a tally of the AXI writes going out and the write responses coming in. At the very end of the current DMA transfer it will wait until the last outstanding write response has been received before indicating the transfer is complete. Whilst waiting it will load the next CB address (but will not fetch the CB), clear the active flag (if the next CB address = zero), and it will defer setting the END flag or the INT flag until the last outstanding write response has been received. In this mode, the DMA will pause if it has more than 13 outstanding writes at any one time.	RW	0x0
27:24		***Reserved*** *- Write as 0, read as don't care*		
23:20	PANIC_PRIORITY	AXI Panic Priority Level Sets the priority of panicking AXI bus transactions. This value is used when the panic bit of the selected peripheral channel is 1. Zero is the lowest priority.	RW	0x0
19:16	PRIORITY	AXI Priority Level Sets the priority of normal AXI bus transactions. This value is used when the panic bit of the selected peripheral channel is zero. Zero is the lowest priority.	RW	0x0
15:9		***Reserved*** *- Write as 0, read as don't care*		
8	ERROR	DMA Error Indicates if the DMA has detected an error. The error flags are available in the debug register, and have to be cleared by writing to that register. 1 = DMA channel has an error flag set. 0 = DMA channel is ok.	RO	0x0
7		***Reserved*** *- Write as 0, read as don't care*		

6	WAITING_FOR_OUTSTANDING_WRITES	DMA is Waiting for the Last Write to be Received Indicates if the DMA is currently waiting for any outstanding writes to be received, and is not transferring data. 1 = DMA channel is waiting.	RO	0x0
5	DREQ_STOPS_DMA	DMA Paused by DREQ State Indicates if the DMA is currently paused and not transferring data due to the DREQ being inactive.. 1 = DMA channel is paused. 0 = DMA channel is running.	RO	0x0
4	PAUSED	DMA Paused State Indicates if the DMA is currently paused and not transferring data. This will occur if: the active bit has been cleared, if the DMA is currently executing wait cycles or if the debug_pause signal has been set by the debug block, or the number of outstanding writes has exceeded the max count. 1 = DMA channel is paused. 0 = DMA channel is running.	RO	0x0
3	DREQ	DREQ State Indicates the state of the selected DREQ (Data Request) signal, ie. the DREQ selected by the PERMAP field of the transfer info. 1 = Requesting data. This will only be valid once the DMA has started and the PERMAP field has been loaded from the CB. It will remain valid, indicating the selected DREQ signal, until a new CB is loaded. If PERMAP is set to zero (un-paced transfer) then this bit will read back as 1. 0 = No data request.	RO	0x0
2	INT	Interrupt Status This is set when the transfer for the CB ends and INTEN is set to 1. Once set it must be manually cleared down, even if the next CB has INTEN = 0. Write 1 to clear.	W1C	0x0

1	END	DMA End Flag Set when the transfer described by the current control block is complete. Write 1 to clear.	W1C	0x0
0	ACTIVE	Activate the DMA This bit enables the DMA. The DMA will start if this bit is set and the CB_ADDR is non zero. The DMA transfer can be paused and resumed by clearing, then setting it again. This bit is automatically cleared at the end of the complete DMA transfer, ie. after a NEXTCONBK = 0x0000_0000 has been loaded.	RW	0x0

0_CONBLK_AD 1_CONBLK_AD 2_CONBLK_AD 3_CONBLK_AD 4_CONBLK_AD 5_CONBLK_AD 6_CONBLK_AD 7_CONBLK_AD 8_CONBLK_AD 9_CONBLK_AD 10_CONBLK_AD 11_CONBLK_AD 12_CONBLK_AD 13_CONBLK_AD 14_CONBLK_AD Register

Synopsis DMA Control Block Address register.

Bit(s)	Field Name	Description	Type	Reset
31:0	SCB_ADDR	Control Block Address This tells the DMA where to find a Control Block stored in memory. When the ACTIVE bit is set and this address is non zero, the DMA will begin its transfer by loading the contents of the addressed CB into the relevant DMA channel registers. At the end of the transfer this register will be updated with the ADDR field of the NEXTCONBK control block register. If this field is zero, the DMA will stop. Reading this register will return the address of the currently active CB (in the linked list of CB s). The address must be 256 bit aligned, so the bottom 5 bits of the address must be zero.	RW	0x0

0_TI 1_TI 2_TI 3_TI 4_TI 5_TI 6_TI Register

Synopsis DMA Transfer Information.

Bit(s)	Field Name	Description	Type	Reset

31:27		***Reserved*** *- Write as 0, read as don't care*		
26	NO_WIDE_BURSTS	Don t Do wide writes as a 2 beat burst This prevents the DMA from issuing wide writes as 2 beat AXI bursts. This is an inefficient access mode, so the default is to use the bursts.	RW	0x0
25:21	WAITS	Add Wait Cycles This slows down the DMA throughput by setting the number of dummy cycles burnt after each DMA read or write operation is completed. A value of 0 means that no wait cycles are to be added.	RW	0x0
20:16	PERMAP	Peripheral Mapping Indicates the peripheral number (1-31) whose ready signal shall be used to control the rate of the transfers, and whose panic signals will be output on the DMA AXI bus. Set to 0 for a continuous un-paced transfer.	RW	0x0
15:12	BURST_LENGTH	Burst Transfer Length Indicates the burst length of the DMA transfers. The DMA will attempt to transfer data as bursts of this number of words. A value of zero will produce a single transfer. Bursts are only produced for specific conditions, see main text.	RW	0x0
11	SRC_IGNORE	Ignore Reads 1 = Do not perform source reads. In addition, destination writes will zero all the write strobes. This is used for fast cache fill operations. 0 = Perform source reads..	RW	0x0
10	SRC_DREQ	Control Source Reads with DREQ 1 = The DREQ selected by PER_MAP will gate the source reads. 0 = DREQ has no effect.	RW	0x0
9	SRC_WIDTH	Source Transfer Width 1 = Use 128-bit source read width. 0 = Use 32-bit source read width.	RW	0x0
8	SRC_INC	Source Address Increment 1 = Source address increments after each read. The address will increment by 4, if S_WIDTH=0 else by 32. 0 = Source address does not change.	RW	0x0

7	DEST_IGNORE	Ignore Writes 1 = Do not perform destination writes. 0 = Write data to destination.	RW	0x0
6	DEST_DREQ	Control Destination Writes with DREQ 1 = The DREQ selected by PERMAP will gate the destination writes. 0 = DREQ has no effect.	RW	0x0
5	DEST_WIDTH	Destination Transfer Width 1 = Use 128-bit destination write width. 0 = Use 32-bit destination write width.	RW	0x0
4	DEST_INC	Destination Address Increment 1 = Destination address increments after each write The address will increment by 4, if DEST_WIDTH=0 else by 32. 0 = Destination address does not change.	RW	0x0
3	WAIT_RESP	Wait for a Write Response When set this makes the DMA wait until it receives the AXI write response for each write. This ensures that multiple writes cannot get stacked in the AXI bus pipeline. 1= Wait for the write response to be received before proceeding. 0 = Don t wait; continue as soon as the write data is sent.	RW	0x0
2		***Reserved*** *- Write as 0, read as don't care*		
1	TDMODE	2D Mode 1 = 2D mode interpret the TXFR_LEN register as YLENGTH number of transfers each of XLENGTH, and add the strides to the address after each transfer. 0 = Linear mode interpret the TXFR register as a single transfer of total length {YLENGTH ,XLENGTH}.	RW	0x0
0	INTEN	Interrupt Enable 1 = Generate an interrupt when the transfer described by the current Control Block completes. 0 = Do not generate an interrupt.	RW	0x0

0_SOURCE_AD 1_SOURCE_AD 2_SOURCE_AD 3_SOURCE_AD 4_SOURCE_AD 5_SOURCE_AD 6_SOURCE_AD 7_SOURCE_AD 8_SOURCE_AD 9_SOURCE_AD 10_SOURCE_AD 11_SOURCE_AD 12_SOURCE_AD 13_SOURCE_AD 14_SOURCE_AD Register

Synopsis DMA Source Address

Bit(s)	Field Name	Description	Type	Reset
31:0	S_ADDR	DMA Source Address Source address for the DMA operation. Updated by the DMA engine as the transfer progresses.	RW	0x0

0_DEST_AD 1_DEST_AD 2_DEST_AD 3_DEST_AD 4_DEST_AD 5_DEST_AD 6_DEST_AD 7_DEST_AD 8_DEST_AD 9_DEST_AD 10_DEST_AD 11_DEST_AD 12_DEST_AD 13_DEST_AD 14_DEST_AD Register

Synopsis DMA Destination Address

Bit(s)	Field Name	Description	Type	Reset
31:0	D_ADDR	DMA Destination Address Destination address for the DMA operation. Updated by the DMA engine as the transfer progresses.	RW	0x0

0_TXFR_LEN 1_TXFR_LEN 2_TXFR_LEN 3_TXFR_LEN 4_TXFR_LEN 5_TXFR_LEN 6_TXFR_LEN Register

Synopsis DMA Transfer Length. This specifies the amount of data to be transferred in bytes.
In normal (non 2D) mode this specifies the amount of bytes to be transferred.
In 2D mode it is interpreted as an X and a Y length, and the DMA will perform Y transfers, each of length X bytes and add the strides onto the addresses after each X leg of the transfer.
The length register is updated by the DMA engine as the transfer progresses, so it will indicate the data left to transfer.

Bit(s)	Field Name	Description	Type	Reset
31:30		***Reserved*** *- Write as 0, read as don't care*		
29:16	YLENGTH	When in 2D mode, This is the Y transfer length, indicating how many xlength transfers are performed. When in normal linear mode this becomes the top bits of the XLENGTH	RW	0x0
15:0	XLENGTH	Transfer Length in bytes.	RW	0x0

0_STRIDE 1_STRIDE 2_STRIDE 3_STRIDE 4_STRIDE 5_STRIDE 6_STRIDE Register

Synopsis DMA 2D Stride

Bit(s)	Field Name	Description	Type	Reset
31:16	D_STRIDE	Destination Stride (2D Mode) Signed (2 s complement) byte increment to apply to the destination address at the end of each row in 2D mode.	RW	0x0
15:0	S_STRIDE	Source Stride (2D Mode) Signed (2 s complement) byte increment to apply to the source address at the end of each row in 2D mode.	RW	0x0

0_NEXTCONBK 1_NEXTCONBK 2_NEXTCONBK 3_NEXTCONBK 4_NEXTCONBK 5_NEXTCONBK 6_NEXTCONBK 7_NEXTCONBK 8_NEXTCONBK 9_NEXTCONBK 10_NEXTCONBK 11_NEXTCONBK 12_NEXTCONBK 13_NEXTCONBK 14_NEXTCONBK Register

Synopsis DMA Next Control Block Address

The value loaded into this register can be overwritten so that the linked list of Control Block data structures can be altered. However it is only safe to do this when the DMA is paused. The address must be 256 bit aligned and so the bottom 5 bits cannot be set and will read back as zero.

Bit(s)	Field Name	Description	Type	Reset
31:0	ADDR	Address of next CB for chained DMA operations.	RW	0x0

0_DEBUG 1_DEBUG 2_DEBUG 3_DEBUG 4_DEBUG 5_DEBUG 6_DEBUG Register

Synopsis DMA Debug register.

Bit(s)	Field Name	Description	Type	Reset
31:29		***Reserved*** *- Write as 0, read as don't care*		

28	LITE	DMA Lite Set if the DMA is a reduced performance LITE engine.	RO	0x0
27:25	VERSION	DMA Version DMA version number, indicating control bit filed changes.	RO	0x2
24:16	DMA_STATE	DMA State Machine State Returns the value of the DMA engines state machine for this channel.	RO	0x0
15:8	DMA_ID	DMA ID Returns the DMA AXI ID of this DMA channel.	RO	0x0
7:4	OUTSTANDING_WRITES	DMA Outstanding Writes Counter Returns the number of write responses that have not yet been received. This count is reset at the start of each new DMA transfer or with a DMA reset.	RO	0x0
3		***Reserved*** *- Write as 0, read as don't care*		
2	READ_ERROR	Slave Read Response Error Set if the read operation returned an error value on the read response bus. It can be cleared by writing a 1,	RW	0x0
1	FIFO_ERROR	Fifo Error Set if the optional read Fifo records an error condition. It can be cleared by writing a 1,	RW	0x0
0	READ_LAST_NOT_SET_ERROR	Read Last Not Set Error If the AXI read last signal was not set when expected, then this error bit will be set. It can be cleared by writing a 1.	RW	0x0

7_TI 8_TI 9_TI 10_TI 11_TI 12_TI 13_TI 14_TI Register

Synopsis DMA Transfer Information.

Bit(s)	Field Name	Description	Type	Reset
31:26		***Reserved*** *- Write as 0, read as don't care*		

25:21	WAITS	Add Wait Cycles This slows down the DMA throughput by setting the number of dummy cycles burnt after each DMA read or write operation is completed. A value of 0 means that no wait cycles are to be added.	RW	0x0
20:16	PERMAP	Peripheral Mapping Indicates the peripheral number (1-31) whose ready signal shall be used to control the rate of the transfers, and whose panic signals will be output on the DMA AXI bus. Set to 0 for a continuous un-paced transfer.	RW	0x0
15:12	BURST_LENGTH	Burst Transfer Length Indicates the burst length of the DMA transfers. The DMA will attempt to transfer data as bursts of this number of words. A value of zero will produce a single transfer. Bursts are only produced for specific conditions, see main text.	RW	0x0
11	SRC_IGNORE		RW	0x0
10	SRC_DREQ	Control Source Reads with DREQ 1 = The DREQ selected by PER_MAP will gate the source reads. 0 = DREQ has no effect.	RW	0x0
9	SRC_WIDTH	Source Transfer Width 1 = Use 128-bit source read width. 0 = Use 32-bit source read width.	RW	0x0
8	SRC_INC	Source Address Increment 1 = Source address increments after each read. The address will increment by 4, if S_WIDTH=0 else by 32. 0 = Source address does not change.	RW	0x0
7	DEST_IGNORE		RW	0x0
6	DEST_DREQ	Control Destination Writes with DREQ 1 = The DREQ selected by PERMAP will gate the destination writes. 0 = DREQ has no effect.	RW	0x0
5	DEST_WIDTH	Destination Transfer Width 1 = Use 128-bit destination write width. 0 = Use 32-bit destination write width.	RW	0x0

4	DEST_INC	Destination Address Increment 1 = Destination address increments after each write The address will increment by 4, if DEST_WIDTH=0 else by 32. 0 = Destination address does not change.	RW	0x0
3	WAIT_RESP	Wait for a Write Response When set this makes the DMA wait until it receives the AXI write response for each write. This ensures that multiple writes cannot get stacked in the AXI bus pipeline. 1= Wait for the write response to be received before proceeding. 0 = Don t wait; continue as soon as the write data is sent.	RW	0x0
2:1		***Reserved** - Write as 0, read as don't care*		
0	INTEN	Interrupt Enable 1 = Generate an interrupt when the transfer described by the current Control Block completes. 0 = Do not generate an interrupt.	RW	0x0

7_TXFR_LEN 8_TXFR_LEN 9_TXFR_LEN 10_TXFR_LEN 11_TXFR_LEN 12_TXFR_LEN 13_TXFR_LEN 14_TXFR_LEN Register

Synopsis DMA Transfer Length

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved** - Write as 0, read as don't care*		
15:0	XLENGTH	Transfer Length Length of transfer, in bytes. Updated by the DMA engine as the transfer progresses.	RW	0x0

7_DEBUG 8_DEBUG 9_DEBUG 10_DEBUG 11_DEBUG 12_DEBUG 13_DEBUG 14_DEBUG Register

Synopsis DMA Lite Debug register.

Bit(s)	Field Name	Description	Type	Reset

31:29		***Reserved*** *- Write as 0, read as don't care*		
28	LITE	DMA Lite Set if the DMA is a reduced performance LITE engine.	RO	0x1
27:25	VERSION	DMA Version DMA version number, indicating control bit filed changes.	RO	0x2
24:16	DMA_STATE	DMA State Machine State Returns the value of the DMA engines state machine for this channel.	RO	0x0
15:8	DMA_ID	DMA ID Returns the DMA AXI ID of this DMA channel.	RO	0x0
7:4	OUTSTANDING_WRITES	DMA Outstanding Writes Counter Returns the number of write responses that have not yet been received. This count is reset at the start of each new DMA transfer or with a DMA reset.	RO	0x0
3		***Reserved*** *- Write as 0, read as don't care*		
2	READ_ERROR	Slave Read Response Error Set if the read operation returned an error value on the read response bus. It can be cleared by writing a 1,	RW	0x0
1	FIFO_ERROR	Fifo Error Set if the optional read Fifo records an error condition. It can be cleared by writing a 1,	RW	0x0
0	READ_LAST_NOT_SET_ERROR	Read Last Not Set Error If the AXI read last signal was not set when expected, then this error bit will be set. It can be cleared by writing a 1.	RW	0x0

INT_STATUS Register

Synopsis Interrupt status of each DMA engine

Bit(s)	Field Name	Description	Type	Reset

31:16		***Reserved*** *- Write as 0, read as don't care*		
15	INT15	Interrupt status of DMA engine 15	RW	0x0
14	INT14	Interrupt status of DMA engine 14	RW	0x0
13	INT13	Interrupt status of DMA engine 13	RW	0x0
12	INT12	Interrupt status of DMA engine 12	RW	0x0
11	INT11	Interrupt status of DMA engine 11	RW	0x0
10	INT10	Interrupt status of DMA engine 10	RW	0x0
9	INT9	Interrupt status of DMA engine 9	RW	0x0
8	INT8	Interrupt status of DMA engine 8	RW	0x0
7	INT7	Interrupt status of DMA engine 7	RW	0x0
6	INT6	Interrupt status of DMA engine 6	RW	0x0
5	INT5	Interrupt status of DMA engine 5	RW	0x0
4	INT4	Interrupt status of DMA engine 4	RW	0x0
3	INT3	Interrupt status of DMA engine 3	RW	0x0
2	INT2	Interrupt status of DMA engine 2	RW	0x0
1	INT1	Interrupt status of DMA engine 1	RW	0x0
0	INT0	Interrupt status of DMA engine 0	RW	0x0

ENABLE Register

Synopsis Global enable bits for each channel

Bit(s)	Field Name	Description	Type	Reset
31:15		***Reserved*** *- Write as 0, read as don't care*		

14	EN14	enable dma engine 14	RW	0x1
13	EN13	enable dma engine 13	RW	0x1
12	EN12	enable dma engine 12	RW	0x1
11	EN11	enable dma engine 11	RW	0x1
10	EN10	enable dma engine 10	RW	0x1
9	EN9	enable dma engine 9	RW	0x1
8	EN8	enable dma engine 8	RW	0x1
7	EN7	enable dma engine 7	RW	0x1
6	EN6	enable dma engine 6	RW	0x1
5	EN5	enable dma engine 5	RW	0x1
4	EN4	enable dma engine 4	RW	0x1
3	EN3	enable dma engine 3	RW	0x1
2	EN2	enable dma engine 2	RW	0x1
1	EN1	enable dma engine 1	RW	0x1
0	EN0	enable dma engine 0	RW	0x1

4.2.1.3 Peripheral DREQ Signals

A DREQ (Data Request) mechanism is used to pace the data flow between the DMA and a peripheral.

Each peripheral is allocated a permanent DREQ signal. Each DMA channel can select which of the DREQ signals should be used to pace the transfer by controlling the DMA reads, DMA writes or both. Note that DREQ 0 is permanently enabled and can be used if no DREQ is required.

When a DREQ signal is being used to pace the DMA reads, the DMA will wait until it has sampled DREQ high before launching a single or burst read operation. It will then wait for all the read data to be returned before re-checking the DREQ and starting the next read. Thus once a peripheral receives the read request it should remove its DREQ as soon as possible to prevent the DMA from re-sampling the same DREQ assertion.

DREQ's are not required when reading from AXI peripherals. In this case, the DMA will request data from the peripheral and the peripheral will only send the data when it is available. The DMA will not request data that is does not have room for, so no pacing of the data flow is required.

DREQ's are required when reading from APB peripherals as the AXI-to-APB bridge will not wait for an APB peripheral to be ready and will just perfom the APB read regardless. Thus an APB peripheral needs to make sure that it has all of its read data ready before it drives its DREQ high.

When writing to peripherals, a DREQ is always required to pace the data. However, due to the pipelined nature of the AXI bus system, several writes may be in flight before the peripheral receives any data and withdraws its DREQ signal. Thus the peripheral must ensure that it has sufficient room in its input FIFO to accommodate the maximum amount of data that it might receive. If the peripheral is unable to do this, the DMA WAIT_RESP mechanism can be used to ensure that only one write is in flight at any one time, however this is less efficient transfer mechanism.

The mapping of peripherals to DREQ's is as follows:

DREQ	Peripheral
0	DREQ = 1 This is always on so use this channel if no DREQ is required.
1	DSI
2	PCM TX
3	PCM RX
4	SMI
5	PWM
6	SPI TX
7	SPI RX

8	BSC/SPI Slave TX
9	BSC/SPI Slave RX
10	unused
11	*e.MMC*
12	*UART TX*
13	SD HOST
14	UART RX.
15	DSI
16	SLIMBUS MCTX.
17	HDMI
18	SLIMBUS MCRX
19	SLIMBUS DC0
20	SLIMBUS DC1
21	SLIMBUS DC2
22	SLIMBUS DC3
23	SLIMBUS DC4
24	Scaler FIFO 0 & SMI *
25	Scaler FIFO 1 & SMI *
26	Scaler FIFO 2 & SMI *
27	SLIMBUS DC5
28	SLIMBUS DC6
29	SLIMBUS DC7
30	SLIMBUS DC8
31	SLIMBUS DC9

* The SMI element of the Scaler FIFO 0 & SMI DREQs can be disabled by setting the SMI_DISABLE bit in the DMA_DREQ_CONTROL register in the system arbiter control block.

4.3 AXI Bursts

The DMA supports bursts under specific conditions. Up to 16 beat bursts can be accommodated.

Peripheral (32 bit wide) read bursts are supported. The DMA will generate the burst if there is sufficient room in its read buffer to accommodate all the data from the burst. This limits the burst size to a maximum of 8 beats.

Read bursts in destination ignore mode (DEST_IGNORE) are supported as there is no need for the DMA to deal with the data. This allows wide bursts of up to 16 beats to be used for efficient L2 cache fills.

DMA channel 0 and 15 are fitted with an external 128 bit 8 word read FIFO. This enables efficient memory to memory transfers to be performed. This FIFO allows the DMA to accommodate a wide read burst up to the size of the FIFO. In practice this will allow a 128 bit wide read burst of 9 as the first word back will be immediately read into the DMA engine (or a 32 bit peripheral read burst of 16 – 8 in the input buffer and 8 in the fifo). On any DMA channel, if a read burst is selected that is too large, the AXI read bus will be stalled until the DMA has written out the data. This may lead to inefficient system operation, and possibly AXI lock up if it causes a circular dependancy.

In general write bursts are not supported. However to increase the efficiency of L2 cache fills, src_ignore (SRC_IGNORE) transfers can be specified with a write burst. In this case the DMA will issue a write burst address sequence followed by the appropriate number of zero data, zero strobe write bus cycles, which will cause the cache to pre-fetch the data. To improve the efficiency of the 128 bit wide bus architecture, and to make use of the DMAs internal 256 bit registers, the DMA will generate 128 bit wide writes as 2 beat bursts wherever possible, although this behaviour can be disabled.

4.4 Error Handling

If the DMA detects a Read Response error it will record the fact in the READ_ERROR flag in the debug register. This will remain set until it is cleared by writing a 1 to it. The DMA will clear its active flag and generate an interrupt. Any outstanding read data transactions (remainder of a burst) will be honoured. This allows the operator to either restart the DMA by clearing the error bit and setting the active bit, or to abort the DMA transfer by clearing the NEXTCONBK register and restarting the DMA with the ABORT bit set.

The DMA will also record any errors from an external read FIFO. These will be latched in the FIFO_ERROR bit in the debug register until they are cleared by writing a '1' to the bit. (note that only DMA0 and 15 have an external read fifo)

If the DMA detects that a read occurred without the AXI rlast set as expected then it will set the READ_LAST_NOT_SET_ERROR bit in the debug register. This can be cleared by writing a '1' to it.

The error bits are logically OR'd together and presented as a general ERROR bit in the CS register.

4.5 DMA LITE Engines

Several of the DMA engines are of the LITE design. This is a reduced specification engine designed to save space. The engine behaves in the same way as a normal DMA engine except for the following differences.

1. The internal data structure is 128 bits instead of 256 bits. This means that if you do a 128 bit wide read burst of more than 1 beat, the DMA input register will be full and the read bus will be stalled. The normal DMA engine can accept a read burst of 2 without stalling. If you do a narrow 32 bit read burst from the peripherals then the lite engine can cope with a burst of 4 as opposed to a burst of 8 for the normal engine. Note that stalling the read bus will potentially reduce the overall system performance, and may possible cause a system lockup if you end up with a conflict where the DMA cannot free the read bus as the read stall has prevented it writing out its data due to some circular system relationship.
2. The Lite engine does not support 2D transfers. The TDMODE, S_STRIDE, D_STRIDE and YLENGTH registers will all be removed. Setting these registers will have no effect.
3. The DMA length register is now 16 bits, limiting the maximum transferrable length to 65536 bytes.
4. Source ignore (SRC_IGNORE) and destination ignore (DEST_IGNORE) modes are removed.

The Lite engine will have about half the bandwidth of a normal DMA engine, and are intended for low bandwith peripheral servicing.

5 External Mass Media Controller

o Introduction

The External Mass Media Controller (EMMC) is an embedded MultiMedia™ and SD™ card interface provided by Arasan™. It is compliant to the following standards:

- SD™ Host Controller Standard Specification Version 3.0 Draft 1.0
- SDIO™ card specification version 3.0
- SD™ Memory Card Specification Draft version 3.0
- SD™ Memory Card Security Specification version 1.01
- MMC™ Specification version 3.31,4.2 and 4.4

For convenience in the following text card is used as a placeholder for SD™, embedded MultiMedia and SDIO™ cards.

For detailed information about the EMMC internals please refer to the Arasan™ document SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf but make sure to read the following chapter which lists the changes made to Arasan™'s IP.

Because the EMMC module shares pins with other functionality it must be selected in the GPIO interface. Please refer to the GPIO section for further details.

The interface to the card uses its own clock clk_emmc which is provided by the clock manager module. The frequency of this clock should be selected between 50 MHz and 100 MHz. Having a separate clock allows high performance access to the card even if the VideoCore runs at a reduced clock frequency. The EMMC module contains its own internal clock divider to generate the card's clock from clk_emmc.

Additionally can the sampling clock for the response and data from the card be delayed in up to 40 steps with a configurable delay between 200ps to 1100ps per step typically. The delay is intended to cancel the internal delay inside the card (up to 14ns) when reading. The delay per step will vary with temperature and supply voltage. Therefore it is better to use a bigger delay than necessary as there is no restriction for the maximum delay.

The EMMC module handles the handshaking process on the command and data lines and all CRC processing automatically.

Command execution is commenced by writing the command plus the appropriate flags to the CMDTM register after loading any required argument into the ARG1 register. The EMMC module calculates the CRC checksum, transfers the command to the card, receives the response and checks its CRC. Once the command has executed or timed-out bit 0 of register INTERRUPT will be set. Please note that the INTERRUPT register is not self clearing, so the software has first to reset it by writing 1 before using it to detect if a command has finished.

The software is responsible for checking the status bits of the card's response in order to verify successful processing by the card.

In order to transfer data from/to the card register DATA is accessed after configuring the host and sending the according commands to the card using CMDTM. Because the EMMC module doesn't interpret the commands sent to the card it is important to configure it identical to the card setup using the CONTROL0 register. Especial care should be taken to make sure that the width of the data bus is configured identical for host and card. The card is synchronized to the data flow by switching off its clock appropriately. A handshake signal dma_req is available for paced data transfers. Bit 1 of the INTERRUPT register can used to determine whether a data transfer has finished. Please note that the INTERRUPT register is not self clearing, so the software has first to reset it by writing 1 before using it to detect if a data transfer has finished.

The EMMC module restricts the maximum block size to the size of the internal data FIFO which is 1k bytes. In order to get maximum performance for data transfers it is necessary to use multiple block data transfers. In this case the EMMC module uses two FIFOs in ping-pong mode, i.e. one is used to transfer data to/from the card while the other is simultaneously accessed by DMA via the AXI bus. If the EMMC module is configured for single block transfers only one FIFO is used, so no DMA access is possible while data is transferred to/from the card and vice versa resulting in long dead times.

○ Registers

Contrary to Arasan™'s documentation the EMMC module registers can only be accessed as 32 bit registers, i.e. the two LSBs of the address are always zero.

The EMMC register base address is 0x7E300000

EMMC Address Map			
Address Offset	**Register Name**	**Description**	**Size**
0x0	ARG2	ACMD23 Argument	32
0x4	BLKSIZECNT	Block Size and Count	32
0x8	ARG1	Argument	32
0xc	CMDTM	Command and Transfer Mode	32
0x10	RESP0	Response bits 31 : 0	32
0x14	RESP1	Response bits 63 : 32	32

0x18	RESP2	Response bits 95 : 64	32
0x1c	RESP3	Response bits 127 : 96	32
0x20	DATA	Data	32
0x24	STATUS	Status	32
0x28	CONTROL0	Host Configuration bits	32
0x2c	CONTROL1	Host Configuration bits	32
0x30	INTERRUPT	Interrupt Flags	32
0x34	IRPT_MASK	Interrupt Flag Enable	32
0x38	IRPT_EN	Interrupt Generation Enable	32
0x3c	CONTROL2	Host Configuration bits	32
0x50	FORCE_IRPT	Force Interrupt Event	32
0x70	BOOT_TIMEOUT	Timeout in boot mode	32
0x74	DBG_SEL	Debug Bus Configuration	32
0x80	EXRDFIFO_CFG	Extension FIFO Configuration	32
0x84	EXRDFIFO_EN	Extension FIFO Enable	32
0x88	TUNE_STEP	Delay per card clock tuning step	32
0x8c	TUNE_STEPS_STD	Card clock tuning steps for SDR	32
0x90	TUNE_STEPS_DDR	Card clock tuning steps for DDR	32
0xf0	SPI_INT_SPT	SPI Interrupt Support	32
0xfc	SLOTISR_VER	Slot Interrupt Status and Version	32

ARG2 Register

Synopsis This register contains the argument for the SD card specific command ACMD23 (SET_WR_BLK_ERASE_COUNT). ARG2 must be set before the ACMD23 command is issued using the CMDTM register.

Bit(s)	Field Name	Description	Type	Reset
31:0	ARGUMENT	Argument to be issued with ACMD23	RW	0x0

BLKSIZECNT Register

Synopsis This register must not be accessed or modified while any data transfer between card and host is ongoing.
It contains the number and size in bytes for data blocks to be transferred. Please note that the EMMC module restricts the maximum block size to the size of the internal data FIFO which is 1k bytes.
BLKCNT is used to tell the host how many blocks of data are to be transferred. Once the data transfer has started and the TM_BLKCNT_EN bit in the CMDTM register is set the EMMC module automatically decreases the BNTCNT value as the data blocks are transferred and stops the transfer once BLKCNT reaches 0.

Bit(s)	Field Name	Description	Type	Reset
31:16	BLKCNT	Number of blocks to be transferred	RW	0x0
15:10		***Reserved** - Write as 0, read as don't care*		
9:0	BLKSIZE	Block size in bytes	RW	0x0

ARG1 Register

Synopsis This register contains the arguments for all commands except for the SD card specific command ACMD23 which uses ARG2. ARG1 must be set before the command is issued using the CMDTM register.

Bit(s)	Field Name	Description	Type	Reset
31:0	ARGUMENT	Argument to be issued with command	RW	0x0

CMDTM Register

Synopsis This register is used to issue commands to the card. Besides the command it also contains flags informing the EMMC module what card response and type of data transfer to expect. Incorrect flags will result in strange behaviour.
For data transfers two modes are supported: either transferring a single block of data or several blocks of the same size. The SD card uses two different sets of commands to differentiate between them but the host needs to be additionally configured using TM_MULTI_BLOCK. It is important that this bit is set correct for the command sent to the card, i.e. 1 for CMD18 and CMD25 and 0 for CMD17 and CMD24. Multiple block transfer gives a better performance.
The BLKSIZECNT register is used to configure the size and number of blocks to be transferred. If bit TM_BLKCNT_EN of this register is set the transfer stops automatically after the number of data blocks configured in the BLKSIZECNT register has been transferred.
The TM_AUTO_CMD_EN bits can be used to make the host to send automatically a command to the card telling it that the data transfer has finished once the BLKCNT bits in the BLKSIZECNT register are 0.

Bit(s)	Field Name	Description	Type	Reset
31:30		***Reserved** - Write as 0, read as don't care*		
29:24	CMD_INDEX	Index of the command to be issued to the card	RW	0x0
23:22	CMD_TYPE	Type of command to be issued to the card: 00 = normal 01 = suspend (the current data transfer) 10 = resume (the last data transfer) 11 = abort (the current data transfer)	RW	0x0
21	CMD_ISDATA	Command involves data transfer: 0 = no data transfer command 1 = data transfer command	RW	0x0
20	CMD_IXCHK_EN	Check that response has same index as command: 0 = disabled 1 = enabled	RW	0x0
19	CMD_CRCCHK_EN	Check the responses CRC: 0 = disabled 1 = enabled	RW	0x0
18		***Reserved** - Write as 0, read as don't care*		
17:16	CMD_RSPNS_TYPE	Type of expected response from card: 00 = no response 01 = 136 bits response 10 = 48 bits response 11 = 48 bits response using busy	RW	0x0

15:6		***Reserved** - Write as 0, read as don't care*		
5	TM_MULTI_BLOCK	Type of data transfer 0 = single block 1 = multiple block	RW	0x0
4	TM_DAT_DIR	Direction of data transfer: 0 = from host to card 1 = from card to host	RW	0x0
3:2	TM_AUTO_CMD_EN	Select the command to be send after completion of a data transfer: 00 = no command 01 = command CMD12 10 = command CMD23 11 = reserved	RW	0x0
1	TM_BLKCNT_EN	Enable the block counter for multiple block transfers: 0 = disabled 1 = enabled	RW	0x0
0		***Reserved** - Write as 0, read as don't care*		

RESP0 Register

Synopsis This register contains the status bits of the SD card s response. In case of commands CMD2 and CMD10 it contains CID[31:0] and in case of command CMD9 it contains CSD[31:0].
Note: this register is only valid once the last command has completed and no new command was issued.

Bit(s)	Field Name	Description	Type	Reset
31:0	RESPONSE	Bits 31:0 of the card s response	RW	0x0

RESP1 Register

Synopsis In case of commands CMD2 and CMD10 this register contains CID[63:32] and in case of command CMD9 it contains CSD[63:32].
Note: this register is only valid once the last command has completed and no new command was issued.

Bit(s)	Field Name	Description	Type	Reset
31:0	RESPONSE	Bits 63:32 of the card s response	RW	0x0

RESP2 Register

Synopsis In case of commands CMD2 and CMD10 this register contains CID[95:64] and in case of command CMD9 it contains CSD[95:64].
Note: this register is only valid once the last command has completed and no new command was issued.

Bit(s)	Field Name	Description	Type	Reset
31:0	RESPONSE	Bits 95:64 of the card s response	RW	0x0

RESP3 Register

Synopsis In case of commands CMD2 and CMD10 this register contains CID[127:96] and in case of command CMD9 it contains CSD[127:96].
Note: this register is only valid once the last command has completed and no new command was issued.

Bit(s)	Field Name	Description	Type	Reset
31:0	RESPONSE	Bits 127:96 of the card s response	RW	0x0

DATA Register

Synopsis This register is used to transfer data to/from the card.
Bit 1 of the INTERRUPT register can be used to check if data is available. For paced DMA transfers the high active signal dma_req can be used.

Bit(s)	Field Name	Description	Type	Reset
31:0	DATA	Data to/from the card	RW	0x0

STATUS Register

Synopsis This register contains information intended for debugging. Its values change automatically according to the hardware. As it involves resynchronisation between different clock domains it changes only after some latency and it is easy sample the values too early.
Therefore it is not recommended to use this register for polling. Instead use the INTERRUPT register which implements a handshake mechanism which makes it impossible to miss a change when polling.

Bit(s)	Field Name	Description	Type	Reset
31:29		***Reserved*** *- Write as 0, read as don't care*		
28:25	DAT_LEVEL1	Value of data lines DAT7 to DAT4	RW	0xf
24	CMD_LEVEL	Value of command line CMD	RW	0x1
23:20	DAT_LEVEL0	Value of data lines DAT3 to DAT0	RW	0xf
19:10		***Reserved*** *- Write as 0, read as don't care*		
9	READ_TRANSFER	New data can be read from EMMC: 0 = no 1 = yes	RW	0x0
8	WRITE_TRANSFER	New data can be written to EMMC: 0 = no 1 = yes	RW	0x0
7:3		***Reserved*** *- Write as 0, read as don't care*		
2	DAT_ACTIVE	At least one data line is active: 0 = no 1 = yes	RW	0x0
1	DAT_INHIBIT	Data lines still used by previous data transfer: 0 = no 1 = yes	RW	0x0
0	CMD_INHIBIT	Command line still used by previous command: 0 = no 1 = yes	RW	0x0

CONTROL0 Register

Synopsis This register is used to configure the EMMC module.
For the exact details please refer to the Arasan documentation SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf. Bits marked as reserved in this document but not by the Arasan documentation refer to functionality which has been disabled due to the changes listed in the previous chapter.

Bit(s)	Field Name	Description	Type	Reset
31:23		***Reserved** - Write as 0, read as don't care*		
22	ALT_BOOT_EN	Enable alternate boot mode access: 0 = disabled 1 = enabled	RW	0x0
21	BOOT_EN	Boot mode access: 0 = stop boot mode access 1 = start boot mode access	RW	0x0
20	SPI_MODE	SPI mode enable: 0 = normal mode 1 = SPI mode	RW	0x0
19	GAP_IEN	Enable SDIO interrupt at block gap (only valid if the HCTL_DWIDTH bit is set): 0 = disabled 1 = enabled	RW	0x0
18	READWAIT_EN	Use DAT2 read-wait protocol for SDIO cards supporting this: 0 = disabled 1 = enabled	RW	0x0
17	GAP_RESTART	Restart a transaction which was stopped using the GAP_STOP bit: 0 = ignore 1 = restart	RW	0x0
16	GAP_STOP	Stop the current transaction at the next block gap: 0 = ignore 1 = stop	RW	0x0
15:6		***Reserved** - Write as 0, read as don't care*		
5	HCTL_8BIT	Use 8 data lines: 0 = disabled 1 = enabled	RW	0x0

4:3		***Reserved*** *- Write as 0, read as don't care*		
2	HCTL_HS_EN	Select high speed mode (i.e. DAT and CMD lines change on the rising CLK edge): 0 = disabled 1 = enabled	RW	0x0
1	HCTL_DWIDTH	Use 4 data lines: 0 = disabled 1 = enabled	RW	0x0
0		***Reserved*** *- Write as 0, read as don't care*		

CONTROL1 Register

Synopsis This register is used to configure the EMMC module. For the exact details please refer to the Arasan documentation SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf. Bits marked as reserved in this document but not by the Arasan documentation refer to functionality which has been disabled due to the changes listed in the previous chapter. CLK_STABLE seems contrary to its name only to indicate that there was a rising edge on the clk_emmc input but not that the frequency of this clock is actually stable.

Bit(s)	Field Name	Description	Type	Reset
31:27		***Reserved*** *- Write as 0, read as don't care*		
26	SRST_DATA	Reset the data handling circuit: 0 = disabled 1 = enabled	RW	0x0
25	SRST_CMD	Reset the command handling circuit: 0 = disabled 1 = enabled	RW	0x0
24	SRST_HC	Reset the complete host circuit: 0 = disabled 1 = enabled	RW	0x0
23:20		***Reserved*** *- Write as 0, read as don't care*		
19:16	DATA_TOUNIT	Data timeout unit exponent: 1111 = disabled x = TMCLK * 2^(x+13)	RW	0x0
15:8	CLK_FREQ8	SD clock base divider LSBs	RW	0x0

7:6	CLK_FREQ_MS2	SD clock base divider MSBs	RW	0x0
5	CLK_GENSEL	Mode of clock generation: 0 = divided 1 = programmable	RW	0x0
4:3		***Reserved*** *- Write as 0, read as don't care*		
2	CLK_EN	SD clock enable: 0 = disabled 1 = enabled	RW	0x0
1	CLK_STABLE	SD clock stable: 0 = no 1 = yes	RO	0x0
0	CLK_INTLEN	Clock enable for internal EMMC clocks for power saving: 0 = disabled 1 = enabled	RW	0x0

INTERRUPT Register

Synopsis This register holds the interrupt flags. Each flag can be disabled using the according bit in the IRPT_MASK register.
For the exact details please refer to the Arasan documentation SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf. Bits marked as reserved in this document but not by the Arasan documentation refer to functionality which has been disabled due to the changes listed in the previous chapter.
ERR is a generic flag and is set if any of the enabled error flags is set.

Bit(s)	Field Name	Description	Type	Reset
31:25		***Reserved*** *- Write as 0, read as don't care*		
24	ACMD_ERR	Auto command error: 0 = no error 1 = error	RW	0x0
23		***Reserved*** *- Write as 0, read as don't care*		
22	DEND_ERR	End bit on data line not 1: 0 = no error 1 = error	RW	0x0

21	DCRC_ERR	Data CRC error: 0 = no error 1 = error	RW	0x0
20	DTO_ERR	Timeout on data line: 0 = no error 1 = error	RW	0x0
19	CBAD_ERR	Incorrect command index in response: 0 = no error 1 = error	RW	0x0
18	CEND_ERR	End bit on command line not 1: 0 = no error 1 = error	RW	0x0
17	CCRC_ERR	Command CRC error: 0 = no error 1 = error	RW	0x0
16	CTO_ERR	Timeout on command line: 0 = no error 1 = error	RW	0x0
15	ERR	An error has occured: 0 = no error 1 = error	RO	0x0
14	ENDBOOT	Boot operation has terminated: 0 = no 1 = yes	RW	0x0
13	BOOTACK	Boot acknowledge has been received: 0 = no 1 = yes	RW	0x0
12	RETUNE	Clock retune request was made: 0 = no 1 = yes	RO	0x0
11:9		***Reserved*** *- Write as 0, read as don't care*		
8	CARD	Card made interrupt request: 0 = no 1 = yes	RO	0x0
7:6		***Reserved*** *- Write as 0, read as don't care*		

5	READ_RDY	DATA register contains data to be read: 0 = no 1 = yes	RW	0x0
4	WRITE_RDY	Data can be written to DATA register: 0 = no 1 = yes	RW	0x0
3		***Reserved*** *- Write as 0, read as don't care*		
2	BLOCK_GAP	Data transfer has stopped at block gap: 0 = no 1 = yes	RW	0x0
1	DATA_DONE	Data transfer has finished: 0 = no 1 = yes	RW	0x0
0	CMD_DONE	Command has finished: 0 = no 1 = yes	RW	0x0

IRPT_MASK Register

Synopsis This register is used to mask the interrupt flags in the INTERRUPT register. For the exact details please refer to the Arasan documentation SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf. Bits marked as reserved in this document but not by the Arasan documentation refer to functionality which has been disabled due to the changes listed in the previous chapter.

Bit(s)	**Field Name**	**Description**	**Type**	**Reset**
31:25		***Reserved*** *- Write as 0, read as don't care*		
24	ACMD_ERR	Set flag if auto command error: 0 = no 1 = yes	RW	0x0
23		***Reserved*** *- Write as 0, read as don't care*		
22	DEND_ERR	Set flag if end bit on data line not 1: 0 = no 1 = yes	RW	0x0

21	DCRC_ERR	Set flag if data CRC error: 0 = no 1 = yes	RW	0x0
20	DTO_ERR	Set flag if timeout on data line: 0 = no 1 = yes	RW	0x0
19	CBAD_ERR	Set flag if incorrect command index in response: 0 = no 1 = yes	RW	0x0
18	CEND_ERR	Set flag if end bit on command line not 1: 0 = no 1 = yes	RW	0x0
17	CCRC_ERR	Set flag if command CRC error: 0 = no 1 = yes	RW	0x0
16	CTO_ERR	Set flag if timeout on command line: 0 = no 1 = yes	RW	0x0
15		***Reserved*** *- Write as 0, read as don't care*		
14	ENDBOOT	Set flag if boot operation has terminated: 0 = no 1 = yes	RW	0x0
13	BOOTACK	Set flag if boot acknowledge has been received: 0 = no 1 = yes	RW	0x0
12	RETUNE	Set flag if clock retune request was made: 0 = no 1 = yes	RW	0x0
11:9		***Reserved*** *- Write as 0, read as don't care*		
8	CARD	Set flag if card made interrupt request: 0 = no 1 = yes	RW	0x0
7:6		***Reserved*** *- Write as 0, read as don't care*		
5	READ_RDY	Set flag if DATA register contains data to be read: 0 = no 1 = yes	RW	0x0

4	WRITE_RDY	Set flag if data can be written to DATA register: 0 = no 1 = yes	RW	0x0
3		***Reserved*** *- Write as 0, read as don't care*		
2	BLOCK_GAP	Set flag if data transfer has stopped at block gap: 0 = no 1 = yes	RW	0x0
1	DATA_DONE	Set flag if data transfer has finished: 0 = no 1 = yes	RW	0x0
0	CMD_DONE	Set flag if command has finished: 0 = no 1 = yes	RW	0x0

IRPT_EN Register

Synopsis This register is used to enable the different interrupts in the INTERRUPT register to generate an interrupt on the int_to_arm output.
For the exact details please refer to the Arasan documentation SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf. Bits marked as reserved in this document but not by the Arasan documentation refer to functionality which has been disabled due to the changes listed in the previous chapter.

Bit(s)	Field Name	Description	Type	Reset
31:25		***Reserved*** *- Write as 0, read as don't care*		
24	ACMD_ERR	Create interrupt if auto command error: 0 = no 1 = yes	RW	0x0
23		***Reserved*** *- Write as 0, read as don't care*		
22	DEND_ERR	Create interrupt if end bit on data line not 1: 0 = no 1 = yes	RW	0x0
21	DCRC_ERR	Create interrupt if data CRC error: 0 = no 1 = yes	RW	0x0

20	DTO_ERR	Create interrupt if timeout on data line: 0 = no 1 = yes	RW	0x0
19	CBAD_ERR	Create interrupt if incorrect command index in response: 0 = no 1 = yes	RW	0x0
18	CEND_ERR	Create interrupt if end bit on command line not 1: 0 = no 1 = yes	RW	0x0
17	CCRC_ERR	Create interrupt if command CRC error: 0 = no 1 = yes	RW	0x0
16	CTO_ERR	Create interrupt if timeout on command line: 0 = no 1 = yes	RW	0x0
15		***Reserved*** *- Write as 0, read as don't care*		
14	ENDBOOT	Create interrupt if boot operation has terminated: 0 = no 1 = yes	RW	0x0
13	BOOTACK	Create interrupt if boot acknowledge has been received: 0 = no 1 = yes	RW	0x0
12	RETUNE	Create interrupt if clock retune request was made: 0 = no 1 = yes	RW	0x0
11:9		***Reserved*** *- Write as 0, read as don't care*		
8	CARD	Create interrupt if card made interrupt request: 0 = no 1 = yes	RW	0x0
7:6		***Reserved*** *- Write as 0, read as don't care*		
5	READ_RDY	Create interrupt if DATA register contains data to be read: 0 = no 1 = yes	RW	0x0

4	WRITE_RDY	Create interrupt if data can be written to DATA register: 0 = no 1 = yes	RW	0x0
3		***Reserved*** *- Write as 0, read as don't care*		
2	BLOCK_GAP	Create interrupt if data transfer has stopped at block gap: 0 = no 1 = yes	RW	0x0
1	DATA_DONE	Create interrupt if data transfer has finished: 0 = no 1 = yes	RW	0x0
0	CMD_DONE	Create interrupt if command has finished: 0 = no 1 = yes	RW	0x0

CONTROL2 Register

Synopsis This register is used to enable the different interrupts in the INTERRUPT register to generate an interrupt on the int_to_arm output.
For the exact details please refer to the Arasan documentation SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf. Bits marked as reserved in this document but not by the Arasan documentation refer to functionality which has been disabled due to the changes listed in the previous chapter.

Bit(s)	Field Name	Description	Type	Reset
31:24		***Reserved*** *- Write as 0, read as don't care*		
23	TUNED	Tuned clock is used for sampling data: 0 = no 1 = yes	RW	0x0
22	TUNEON	Start tuning the SD clock: 0 = not tuned or tuning complete 1 = tuning	RW	0x0
21:19		***Reserved*** *- Write as 0, read as don't care*		

18:16	UHSMODE	Select the speed mode of the SD card: 000 = SDR12 001 = SDR25 010 = SDR50 011 = SDR104 100 = DDR50 other = reserved	RW	0x0
15:8		***Reserved*** *- Write as 0, read as don't care*		
7	NOTC12_ERR	Error occurred during auto command CMD12 execution: 0 = no error 1 = error	RO	0x0
6:5		***Reserved*** *- Write as 0, read as don't care*		
4	ACBAD_ERR	Command index error occurred during auto command execution: 0 = no error 1 = error	RO	0x0
3	ACEND_ERR	End bit is not 1 during auto command execution: 0 = no error 1 = error	RO	0x0
2	ACCRC_ERR	Command CRC error occurred during auto command execution: 0 = no error 1 = error	RO	0x0
1	ACTO_ERR	Timeout occurred during auto command execution: 0 = no error 1 = error	RO	0x0
0	ACNOX_ERR	Auto command not executed due to an error: 0 = no 1 = yes	RO	0x0

FORCE_IRPT Register

Synopsis This register is used to fake the different interrupt events for debugging. For the exact details please refer to the Arasan documentation SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf. Bits marked as reserved in this document but not by the Arasan documentation refer to functionality which has been disabled due to the changes listed in the previous chapter.

Bit(s)	Field Name	Description	Type	Reset
31:25		***Reserved*** *- Write as 0, read as don't care*		
24	ACMD_ERR	Create auto command error: 0 = no 1 = yes	RW	0x0
23		***Reserved*** *- Write as 0, read as don't care*		
22	DEND_ERR	Create end bit on data line not 1: 0 = no 1 = yes	RW	0x0
21	DCRC_ERR	Create data CRC error: 0 = no 1 = yes	RW	0x0
20	DTO_ERR	Create timeout on data line: 0 = no 1 = yes	RW	0x0
19	CBAD_ERR	Create incorrect command index in response: 0 = no 1 = yes	RW	0x0
18	CEND_ERR	Create end bit on command line not 1: 0 = no 1 = yes	RW	0x0
17	CCRC_ERR	Create command CRC error: 0 = no 1 = yes	RW	0x0
16	CTO_ERR	Create timeout on command line: 0 = no 1 = yes	RW	0x0
15		***Reserved*** *- Write as 0, read as don't care*		
14	ENDBOOT	Create boot operation has terminated: 0 = no 1 = yes	RW	0x0
13	BOOTACK	Create boot acknowledge has been received: 0 = no 1 = yes	RW	0x0

12	RETUNE	Create clock retune request was made: 0 = no 1 = yes	RW	0x0
11:9		***Reserved*** *- Write as 0, read as don't care*		
8	CARD	Create card made interrupt request: 0 = no 1 = yes	RW	0x0
7:6		***Reserved*** *- Write as 0, read as don't care*		
5	READ_RDY	Create DATA register contains data to be read: 0 = no 1 = yes	RW	0x0
4	WRITE_RDY	Create data can be written to DATA register: 0 = no 1 = yes	RW	0x0
3		***Reserved*** *- Write as 0, read as don't care*		
2	BLOCK_GAP	Create interrupt if data transfer has stopped at block gap: 0 = no 1 = yes	RW	0x0
1	DATA_DONE	Create data transfer has finished: 0 = no 1 = yes	RW	0x0
0	CMD_DONE	Create command has finished: 0 = no 1 = yes	RW	0x0

BOOT_TIMEOUT Register

Synopsis This register configures after how many card clock cycles a timeout for e.MMC cards in boot mode is flagged

Bit(s)	Field Name	Description	Type	Reset
31:0	TIMEOUT	Number of card clock cycles after which a timeout during boot mode is flagged	RW	0x0

DBG_SEL Register

Synopsis This register selects which submodules are accessed by the debug bus. For the exact details please refer to the Arasan documentation SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf. Bits marked as reserved in this document but not by the Arasan documentation refer to functionality which has been disabled due to the changes listed in the previous chapter.

Bit(s)	Field Name	Description	Type	Reset
31:1		***Reserved** - Write as 0, read as don't care*		
0	SELECT	Submodules accessed by debug bus: 0 = receiver and fifo_ctrl 1 = others	RW	0x0

EXRDFIFO_CFG Register

Synopsis This register allows fine tuning the dma_req generation for paced DMA transfers when reading from the card. If the extension data FIFO contains less than RD_THRSH 32 bits words dma_req becomes inactive until the card has filled the extension data FIFO above threshold. This compensates the DMA latency.
When writing data to the card the extension data FIFO feeds into the EMMC module s FIFO and no fine tuning is required Therefore the RD_THRSH value is in this case ignored.

Bit(s)	Field Name	Description	Type	Reset
31:3		***Reserved** - Write as 0, read as don't care*		
2:0	RD_THRSH	Read threshold in 32 bits words	RW	0x0

EXRDFIFO_EN Register

Synopsis This register enables the extension data register. It should be enabled for paced DMA transfers and be bypassed for burst DMA transfers.

Bit(s)	Field Name	Description	Type	Reset
31:1		***Reserved** - Write as 0, read as don't care*		

0	ENABLE	Enable the extension FIFO: 0 = bypass 1 = enabled	RW	0x0

TUNE_STEP Register

Synopsis This register is used to delay the card clock when sampling the returning data and command response from the card.
DELAY determines by how much the sampling clock is delayed per step.

Bit(s)	Field Name	Description	Type	Reset
31:3		***Reserved*** *- Write as 0, read as don't care*		
2:0	DELAY	Sampling clock delay per step: 000 = 200ps typically 001 = 400ps typically 010 = 400ps typically 011 = 600ps typically 100 = 700ps typically 101 = 900ps typically 110 = 900ps typically 111 = 1100ps typically	RW	0x0

TUNE_STEPS_STD Register

Synopsis This register is used to delay the card clock when sampling the returning data and command response from the card. It determines by how many steps the sampling clock is delayed in SDR mode.

Bit(s)	Field Name	Description	Type	Reset
31:6		***Reserved*** *- Write as 0, read as don't care*		
5:0	STEPS	Number of steps (0 to 40)	RW	0x0

TUNE_STEPS_DDR Register

Synopsis This register is used to delay the card clock when sampling the returning data and command response from the card. It determines by how many steps the sampling clock is delayed in DDR mode.

Bit(s)	Field Name	Description	Type	Reset
31:6		***Reserved** - Write as 0, read as don't care*		
5:0	STEPS	Number of steps (0 to 40)	RW	0x0

SPI_INT_SPT Register

Synopsis This register controls whether assertion of interrupts in SPI mode is possible independent of the card select line.
For the exact details please refer to the Arasan documentation SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf. Bit marked as reserved in this document but not by the Arasan documentation refer to functionality which has been disabled due to the changes listed in the previous chapter.

Bit(s)	Field Name	Description	Type	Reset
31:8		***Reserved** - Write as 0, read as don't care*		
7:0	SELECT	Interrupt independent of card select line: 0 = no 1 = yes	RW	0x0

SLOTISR_VER Register

Synopsis This register contains the version information and slot interrupt status.
For the exact details please refer to the Arasan documentation SD3.0_Host_AHB_eMMC4.4_Usersguide_ver5.9_jan11_10.pdf. Bit marked as reserved in this document but not by the Arasan documentation refer to functionality which has been disabled due to the changes listed in the previous chapter.

Bit(s)	Field Name	Description	Type	Reset
31:24	VENDOR	Vendor Version Number	RW	0x0
23:16	SDVERSION	Host Controller specification version	RW	0x0

15:8		***Reserved*** *- Write as 0, read as don't care*		
7:0	SLOT_STATUS	Logical OR of interrupt and wakeup signal for each slot	RW	0x0

6 General Purpose I/O (GPIO)

There are 54 general-purpose I/O (GPIO) lines split into two banks. All GPIO pins have at least two alternative functions within BCM. The alternate functions are usually peripheral IO and a single peripheral may appear in each bank to allow flexibility on the choice of IO voltage. Details of alternative functions are given in section 6.2. Alternative Function Assignments.

The block diagram for an individual GPIO pin is given below :

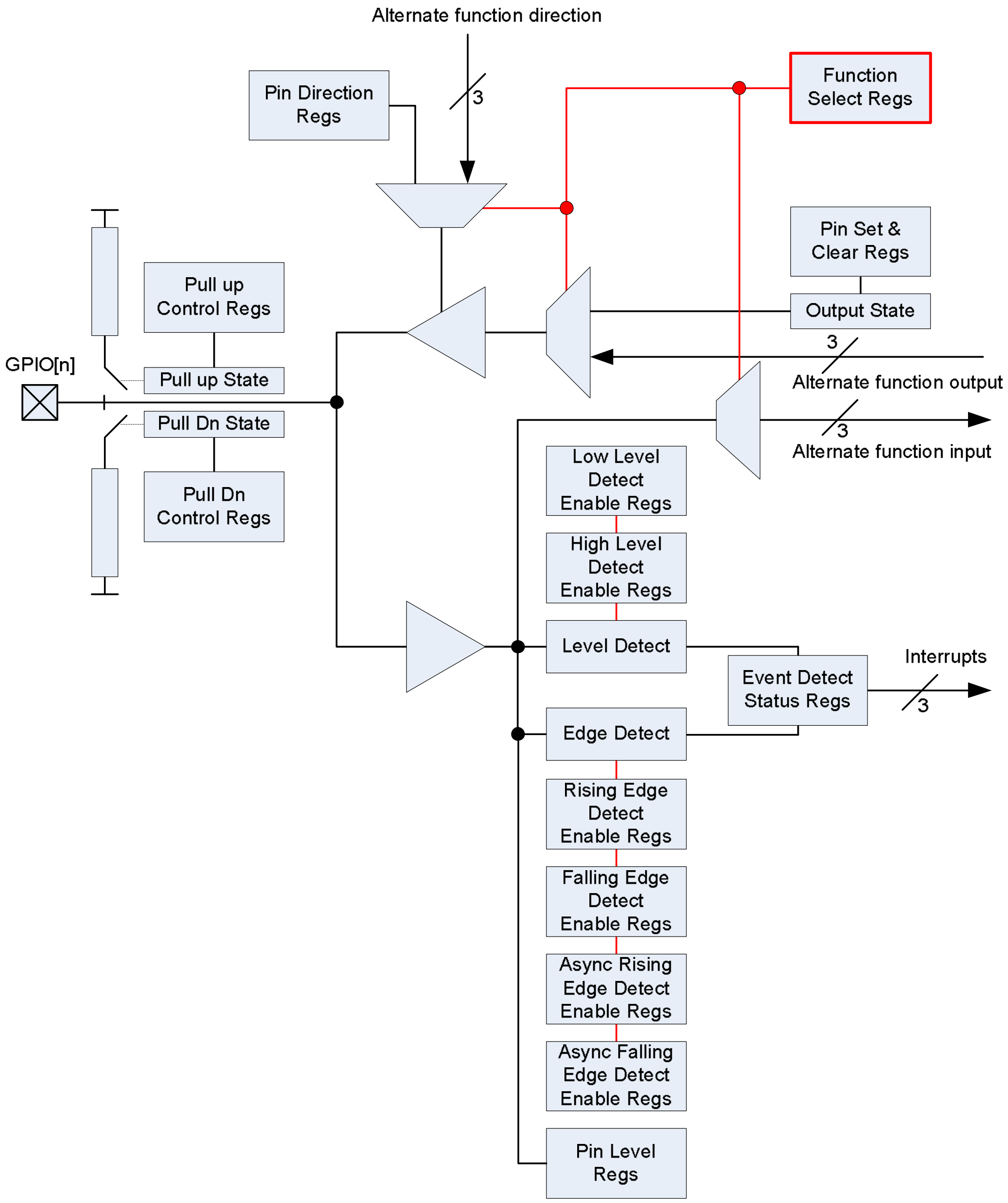

Alternate function direction
3
Pin Direction Regs
Function Select Regs
Pin Set & Clear Regs
Output State
3
Alternate function output
3
Alternate function input
Pull up Control Regs
Pull up State
GPIO[n]
Pull Dn State
Pull Dn Control Regs
Low Level Detect Enable Regs
High Level Detect Enable Regs
Level Detect
Event Detect Status Regs
Interrupts
3
Edge Detect
Rising Edge Detect Enable Regs
Falling Edge Detect Enable Regs
Async Rising Edge Detect Enable Regs
Async Falling Edge Detect Enable Regs
Pin Level Regs

Figure 6-1 GPIO Block Diagram

The GPIO peripheral has three dedicated interrupt lines. These lines are triggered by the setting of bits in the event detect status register. Each bank has its' own interrupt line with the third line shared between all bits.

The Alternate function table also has the pull state (pull-up/pull-down) which is applied after a power down.

6.1 Register View

The GPIO has 41 registers. All accesses are assumed to be 32-bit.

Address	Field Name	Description	Size	Read/ Write
0x 7E20 0000	GPFSEL0	GPIO Function Select 0	32	R/W
0x 7E20 0000	GPFSEL0	GPIO Function Select 0	32	R/W
0x 7E20 0004	GPFSEL1	GPIO Function Select 1	32	R/W
0x 7E20 0008	GPFSEL2	GPIO Function Select 2	32	R/W
0x 7E20 000C	GPFSEL3	GPIO Function Select 3	32	R/W
0x 7E20 0010	GPFSEL4	GPIO Function Select 4	32	R/W
0x 7E20 0014	GPFSEL5	GPIO Function Select 5	32	R/W
0x 7E20 0018	-	Reserved	-	-
0x 7E20 001C	GPSET0	GPIO Pin Output Set 0	32	W
0x 7E20 0020	GPSET1	GPIO Pin Output Set 1	32	W
0x 7E20 0024	-	Reserved	-	-
0x 7E20 0028	GPCLR0	GPIO Pin Output Clear 0	32	W
0x 7E20 002C	GPCLR1	GPIO Pin Output Clear 1	32	W
0x 7E20 0030	-	Reserved	-	-
0x 7E20 0034	GPLEV0	GPIO Pin Level 0	32	R
0x 7E20 0038	GPLEV1	GPIO Pin Level 1	32	R
0x 7E20 003C	-	Reserved	-	-
0x 7E20 0040	GPEDS0	GPIO Pin Event Detect Status 0	32	R/W
0x 7E20 0044	GPEDS1	GPIO Pin Event Detect Status 1	32	R/W
0x 7E20 0048	-	Reserved	-	-
0x 7E20 004C	GPREN0	GPIO Pin Rising Edge Detect Enable 0	32	R/W
0x 7E20 0050	GPREN1	GPIO Pin Rising Edge Detect Enable 1	32	R/W
0x 7E20 0054	-	Reserved	-	-
0x 7E20 0058	GPFEN0	GPIO Pin Falling Edge Detect Enable 0	32	R/W
0x 7E20 005C	GPFEN1	GPIO Pin Falling Edge Detect Enable 1	32	R/W

Address	Field Name	Description	Size	Read/ Write
0x 7E20 0060	-	Reserved	-	-
0x 7E20 0064	GPHEN0	GPIO Pin High Detect Enable 0	32	R/W
0x 7E20 0068	GPHEN1	GPIO Pin High Detect Enable 1	32	R/W
0x 7E20 006C	-	Reserved	-	-
0x 7E20 0070	GPLEN0	GPIO Pin Low Detect Enable 0	32	R/W
0x 7E20 0074	GPLEN1	GPIO Pin Low Detect Enable 1	32	R/W
0x 7E20 0078	-	Reserved	-	-
0x 7E20 007C	GPAREN0	GPIO Pin Async. Rising Edge Detect 0	32	R/W
0x 7E20 0080	GPAREN1	GPIO Pin Async. Rising Edge Detect 1	32	R/W
0x 7E20 0084	-	Reserved	-	-
0x 7E20 0088	GPAFEN0	GPIO Pin Async. Falling Edge Detect 0	32	R/W
0x 7E20 008C	GPAFEN1	GPIO Pin Async. Falling Edge Detect 1	32	R/W
0x 7E20 0090	-	Reserved	-	-
0x 7E20 0094	GPPUD	GPIO Pin Pull-up/down Enable	32	R/W
0x 7E20 0098	GPPUDCLK0	GPIO Pin Pull-up/down Enable Clock 0	32	R/W
0x 7E20 009C	GPPUDCLK1	GPIO Pin Pull-up/down Enable Clock 1	32	R/W
0x 7E20 00A0	-	Reserved	-	-
0x 7E20 00B0	-	Test	4	R/W

Table 6-1 GPIO Register Assignment

GPIO Function Select Registers (GPFSELn)

SYNOPSIS The function select registers are used to define the operation of the general-purpose I/O pins. Each of the 54 GPIO pins has at least two alternative functions as defined in section 16.2. The FSEL{n} field determines the functionality of the nth GPIO pin. All unused alternative function lines are tied to ground and will output a "0" if selected. All pins reset to normal GPIO input operation.

Bit(s)	Field Name	Description	Type	Reset
31-30	---	Reserved	R	0

29-27	FSEL9	FSEL9 - Function Select 9 000 = GPIO Pin 9 is an input 001 = GPIO Pin 9 is an output 100 = GPIO Pin 9 takes alternate function 0 101 = GPIO Pin 9 takes alternate function 1 110 = GPIO Pin 9 takes alternate function 2 111 = GPIO Pin 9 takes alternate function 3 011 = GPIO Pin 9 takes alternate function 4 010 = GPIO Pin 9 takes alternate function 5	R/W	0
26-24	FSEL8	FSEL8 - Function Select 8	R/W	0
23-21	FSEL7	FSEL7 - Function Select 7	R/W	0
20-18	FSEL6	FSEL6 - Function Select 6	R/W	0
17-15	FSEL5	FSEL5 - Function Select 5	R/W	0
14-12	FSEL4	FSEL4 - Function Select 4	R/W	0
11-9	FSEL3	FSEL3 - Function Select 3	R/W	0
8-6	FSEL2	FSEL2 - Function Select 2	R/W	0
5-3	FSEL1	FSEL1 - Function Select 1	R/W	0
2-0	FSEL0	FSEL0 - Function Select 0	R/W	0

Table 6-2 – GPIO Alternate function select register 0

Bit(s)	Field Name	Description	Type	Reset
31-30	---	Reserved	R	0
29-27	FSEL19	FSEL19 - Function Select 19 000 = GPIO Pin 19 is an input 001 = GPIO Pin 19 is an output 100 = GPIO Pin 19 takes alternate function 0 101 = GPIO Pin 19 takes alternate function 1 110 = GPIO Pin 19 takes alternate function 2 111 = GPIO Pin 19 takes alternate function 3 011 = GPIO Pin 19 takes alternate function 4 010 = GPIO Pin 19 takes alternate function 5	R/W	0
26-24	FSEL18	FSEL18 - Function Select 18	R/W	0
23-21	FSEL17	FSEL17 - Function Select 17	R/W	0
20-18	FSEL16	FSEL16 - Function Select 16	R/W	0
17-15	FSEL15	FSEL15 - Function Select 15	R/W	0
14-12	FSEL14	FSEL14 - Function Select 14	R/W	0
11-9	FSEL13	FSEL13 - Function Select 13	R/W	0
8-6	FSEL12	FSEL12 - Function Select 12	R/W	0
5-3	FSEL11	FSEL11 - Function Select 11	R/W	0
2-0	FSEL10	FSEL10 - Function Select 10	R/W	0

Table 6-3 – GPIO Alternate function select register 1

Bit(s)	Field Name	Description	Type	Reset
31-30	---	Reserved	R	0
29-27	FSEL29	FSEL29 - Function Select 29 000 = GPIO Pin 29 is an input 001 = GPIO Pin 29 is an output 100 = GPIO Pin 29 takes alternate function 0 101 = GPIO Pin 29 takes alternate function 1 110 = GPIO Pin 29 takes alternate function 2 111 = GPIO Pin 29 takes alternate function 3 011 = GPIO Pin 29 takes alternate function 4 010 = GPIO Pin 29 takes alternate function 5	R/W	0
26-24	FSEL28	FSEL28 - Function Select 28	R/W	0
23-21	FSEL27	FSEL27 - Function Select 27	R/W	0
20-18	FSEL26	FSEL26 - Function Select 26	R/W	0
17-15	FSEL25	FSEL25 - Function Select 25	R/W	0
14-12	FSEL24	FSEL24 - Function Select 24	R/W	0
11-9	FSEL23	FSEL23 - Function Select 23	R/W	0
8-6	FSEL22	FSEL22 - Function Select 22	R/W	0
5-3	FSEL21	FSEL21 - Function Select 21	R/W	0
2-0	FSEL20	FSEL20 - Function Select 20	R/W	0

Table 6-4 – GPIO Alternate function select register 2

Bit(s)	Field Name	Description	Type	Reset
31-30	---	Reserved	R	0
29-27	FSEL39	FSEL39 - Function Select 39 000 = GPIO Pin 39 is an input 001 = GPIO Pin 39 is an output 100 = GPIO Pin 39 takes alternate function 0 101 = GPIO Pin 39 takes alternate function 1 110 = GPIO Pin 39 takes alternate function 2 111 = GPIO Pin 39 takes alternate function 3 011 = GPIO Pin 39 takes alternate function 4 010 = GPIO Pin 39 takes alternate function 5	R/W	0
26-24	FSEL38	FSEL38 - Function Select 38	R/W	0
23-21	FSEL37	FSEL37 - Function Select 37	R/W	0
20-18	FSEL36	FSEL36 - Function Select 36	R/W	0
17-15	FSEL35	FSEL35 - Function Select 35	R/W	0
14-12	FSEL34	FSEL34 - Function Select 34	R/W	0
11-9	FSEL33	FSEL33 - Function Select 33	R/W	0
8-6	FSEL32	FSEL32 - Function Select 32	R/W	0

5-3	FSEL31	FSEL31 - Function Select 31	R/W	0
2-0	FSEL30	FSEL30 - Function Select 30	R/W	0

Table 6-5 – GPIO Alternate function select register 3

Bit(s)	Field Name	Description	Type	Reset
31-30	---	Reserved	R	0
29-27	FSEL49	FSEL49 - Function Select 49 000 = GPIO Pin 49 is an input 001 = GPIO Pin 49 is an output 100 = GPIO Pin 49 takes alternate function 0 101 = GPIO Pin 49 takes alternate function 1 110 = GPIO Pin 49 takes alternate function 2 111 = GPIO Pin 49 takes alternate function 3 011 = GPIO Pin 49 takes alternate function 4 010 = GPIO Pin 49 takes alternate function 5	R/W	0
26-24	FSEL48	FSEL48 - Function Select 48	R/W	0
23-21	FSEL47	FSEL47 - Function Select 47	R/W	0
20-18	FSEL46	FSEL46 - Function Select 46	R/W	0
17-15	FSEL45	FSEL45 - Function Select 45	R/W	0
14-12	FSEL44	FSEL44 - Function Select 44	R/W	0
11-9	FSEL43	FSEL43 - Function Select 43	R/W	0
8-6	FSEL42	FSEL42 - Function Select 42	R/W	0
5-3	FSEL41	FSEL41 - Function Select 41	R/W	0
2-0	FSEL40	FSEL40 - Function Select 40	R/W	0

Table 6-6 – GPIO Alternate function select register 4

Bit(s)	Field Name	Description	Type	Reset
31-12	---	Reserved	R	0
11-9	FSEL53	FSEL53 - Function Select 53 000 = GPIO Pin 53 is an input 001 = GPIO Pin 53 is an output 100 = GPIO Pin 53 takes alternate function 0 101 = GPIO Pin 53 takes alternate function 1 110 = GPIO Pin 53 takes alternate function 2 111 = GPIO Pin 53 takes alternate function 3 011 = GPIO Pin 53 takes alternate function 4 010 = GPIO Pin 53 takes alternate function 5	R/W	0
8-6	FSEL52	FSEL52 - Function Select 52	R/W	0
5-3	FSEL51	FSEL51 - Function Select 51	R/W	0
2-0	FSEL50	FSEL50 - Function Select 50	R/W	0

Table 6-7 – GPIO Alternate function select register 5

GPIO Pin Output Set Registers (GPSETn)

SYNOPSIS The output set registers are used to set a GPIO pin. The SET{n} field defines the respective GPIO pin to set, writing a “0” to the field has no effect. If the GPIO pin is being used as in input (by default) then the value in the SET{n} field is ignored. However, if the pin is subsequently defined as an output then the bit will be set according to the last set/clear operation. Separating the set and clear functions removes the need for read-modify-write operations

Bit(s)	Field Name	Description	Type	Reset
31-0	SETn (n=0..31)	0 = No effect 1 = Set GPIO pin *n*	R/W	0

Table 6-8 – GPIO Output Set Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0
21-0	SETn (n=32..53)	0 = No effect 1 = Set GPIO pin *n.*	R/W	0

Table 6-9 – GPIO Output Set Register 1

GPIO Pin Output Clear Registers (GPCLRn)

SYNOPSIS The output clear registers) are used to clear a GPIO pin. The CLR{n} field defines the respective GPIO pin to clear, writing a “0” to the field has no effect. If the GPIO pin is being used as in input (by default) then the value in the CLR{n} field is ignored. However, if the pin is subsequently defined as an output then the bit will be set according to the last set/clear operation. Separating the set and clear functions removes the need for read-modify-write operations.

Bit(s)	Field Name	Description	Type	Reset
31-0	CLRn (n=0..31)	0 = No effect 1 = Clear GPIO pin *n*	R/W	0

Table 6-10 – GPIO Output Clear Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0

21-0	CLRn (n=32..53)	0 = No effect 1 = Set GPIO pin *n*	R/W	0

Table 6-11 – GPIO Output Clear Register 1

GPIO Pin Level Registers (GPLEVn)

SYNOPSIS The pin level registers return the actual value of the pin. The LEV{n} field gives the value of the respective GPIO pin.

Bit(s)	Field Name	Description	Type	Reset
31-0	LEVn (n=0..31)	0 = GPIO pin *n* is low 0 = GPIO pin *n* is high	R/W	0

Table 6-12 – GPIO Level Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0
21-0	LEVn (n=32..53)	0 = GPIO pin *n* is low 0 = GPIO pin *n* is high	R/W	0

Table 6-13 – GPIO Level Register 1

GPIO Event Detect Status Registers (GPEDSn)

SYNOPSIS The event detect status registers are used to record level and edge events on the GPIO pins. The relevant bit in the event detect status registers is set whenever: 1) an edge is detected that matches the type of edge programmed in the rising/falling edge detect enable registers, or 2) a level is detected that matches the type of level programmed in the high/low level detect enable registers. The bit is cleared by writing a "1" to the relevant bit.

The interrupt controller can be programmed to interrupt the processor when any of the status bits are set. The GPIO peripheral has three dedicated interrupt lines. Each GPIO bank can generate an independent interrupt. The third line generates a single interrupt whenever any bit is set.

Bit(s)	Field Name	Description	Type	Reset
31-0	EDSn (n=0..31)	0 = Event not detected on GPIO pin n 1 = Event detected on GPIO pin n	R/W	0

Table 6-14 – GPIO Event Detect Status Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0
21-0	EDSn (n=32..53)	0 = Event not detected on GPIO pin n 1 = Event detected on GPIO pin n	R/W	0

Table 6-15 – GPIO Event Detect Status Register 1

GPIO Rising Edge Detect Enable Registers (GPRENn)

SYNOPSIS The rising edge detect enable registers define the pins for which a rising edge transition sets a bit in the event detect status registers (GPEDSn). When the relevant bits are set in both the GPRENn and GPFENn registers, any transition (1 to 0 and 0 to 1) will set a bit in the GPEDSn registers. The GPRENn registers use synchronous edge detection. This means the input signal is sampled using the system clock and then it is looking for a "011" pattern on the sampled signal. This has the effect of suppressing glitches.

Bit(s)	Field Name	Description	Type	Reset
31-0	RENn (n=0..31)	0 = Rising edge detect disabled on GPIO pin *n*. 1 = Rising edge on GPIO pin *n* sets corresponding bit in EDSn.	R/W	0

Table 6-16 – GPIO Rising Edge Detect Status Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0
21-0	RENn (n=32..53)	0 = Rising edge detect disabled on GPIO pin *n*. 1 = Rising edge on GPIO pin *n* sets corresponding bit in EDSn.	R/W	0

Table 6-17 – GPIO Rising Edge Detect Status Register 1

GPIO Falling Edge Detect Enable Registers (GPRENn)

SYNOPSIS The falling edge detect enable registers define the pins for which a falling edge transition sets a bit in the event detect status registers (GPEDSn). When the relevant bits are set in both the GPRENn and GPFENn registers, any transition (1 to 0 and 0 to 1) will set a bit in the GPEDSn registers. The GPFENn registers use synchronous edge detection. This means the input signal is sampled using the system clock and then it is looking for a "100" pattern on the sampled signal. This has the effect of suppressing glitches.

Bit(s)	Field Name	Description	Type	Reset
31-0	FENn (n=0..31)	0 = Falling edge detect disabled on GPIO pin *n.* 1 = Falling edge on GPIO pin *n* sets corresponding bit in EDSn.	R/W	0

Table 6-18 – GPIO Falling Edge Detect Status Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0
21-0	FENn (n=32..53)	0 = Falling edge detect disabled on GPIO pin *n.* 1 = Falling edge on GPIO pin *n* sets corresponding bit in EDSn.	R/W	0

Table 6-19 – GPIO Falling Edge Detect Status Register 1

GPIO High Detect Enable Registers (GPHENn)

SYNOPSIS The high level detect enable registers define the pins for which a high level sets a bit in the event detect status register (GPEDSn). If the pin is still high when an attempt is made to clear the status bit in GPEDSn then the status bit will remain set.

Bit(s)	Field Name	Description	Type	Reset
31-0	HENn (n=0..31)	0 = High detect disabled on GPIO pin *n* 1 = High on GPIO pin *n* sets corresponding bit in GPEDS	R/W	0

Table 6-20 – GPIO High Detect Status Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0
21-0	HENn (n=32..53)	0 = High detect disabled on GPIO pin *n* 1 = High on GPIO pin *n* sets corresponding bit in GPEDS	R/W	0

Table 6-21 – GPIO High Detect Status Register 1

GPIO Low Detect Enable Registers (GPLENn)

SYNOPSIS The low level detect enable registers define the pins for which a low level sets a bit in the event detect status register (GPEDSn). If the pin is still low when an attempt is made to clear the status bit in GPEDSn then the status bit will remain set.

Bit(s)	Field Name	Description	Type	Reset
31-0	LENn (n=0..31)	0 = Low detect disabled on GPIO pin *n* 1 = Low on GPIO pin *n* sets corresponding bit in GPEDS	R/W	0

Table 6-22 – GPIO Low Detect Status Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0
21-0	LENn (n=32..53)	0 = Low detect disabled on GPIO pin *n* 1 = Low on GPIO pin *n* sets corresponding bit in GPEDS	R/W	0

Table 6-23 – GPIO Low Detect Status Register 1

GPIO Asynchronous rising Edge Detect Enable Registers (GPARENn)

SYNOPSIS The asynchronous rising edge detect enable registers define the pins for which a asynchronous rising edge transition sets a bit in the event detect status registers (GPEDSn).

Asynchronous means the incoming signal is not sampled by the system clock. As such rising edges of very short duration can be detected.

Bit(s)	Field Name	Description	Type	Reset
31-0	ARENn (n=0..31)	0 = Asynchronous rising edge detect disabled on GPIO pin *n.* 1 = Asynchronous rising edge on GPIO pin *n* sets corresponding bit in EDSn.	R/W	0

Table 6-24 – GPIO Asynchronous rising Edge Detect Status Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0
21-0	ARENn (n=32..53)	0 = Asynchronous rising edge detect disabled on GPIO pin *n.* 1 = Asynchronous rising edge on GPIO pin *n* sets corresponding bit in EDSn.	R/W	0

Table 6-25 – GPIO Asynchronous rising Edge Detect Status Register 1

GPIO Asynchronous Falling Edge Detect Enable Registers (GPAFENn)

SYNOPSIS The asynchronous falling edge detect enable registers define the pins for which a asynchronous falling edge transition sets a bit in the event detect status registers (GPEDSn). Asynchronous means the incoming signal is not sampled by the system clock. As such falling edges of very short duration can be detected.

Bit(s)	Field Name	Description	Type	Reset
31-0	AFENn (n=0..31)	0 = Asynchronous falling edge detect disabled on GPIO pin *n*. 1 = Asynchronous falling edge on GPIO pin *n* sets corresponding bit in EDSn.	R/W	0

Table 6-26 – GPIO Asynchronous Falling Edge Detect Status Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0
21-0	AFENn (n=32..53)	0 = Asynchronous falling edge detect disabled on GPIO pin *n*. 1 = Asynchronous falling edge on GPIO pin *n* sets corresponding bit in EDSn.	R/W	0

Table 6-27 – GPIO Asynchronous Falling Edge Detect Status Register 1

GPIO Pull-up/down Register (GPPUD)

SYNOPSIS The GPIO Pull-up/down Register controls the actuation of the internal pull-up/down control line to ALL the GPIO pins. This register must be used in conjunction with the 2 GPPUDCLKn registers.

Note that it is not possible to read back the current Pull-up/down settings and so it is the users' responsibility to 'remember' which pull-up/downs are active. The reason for this is that GPIO pull-ups are maintained even in power-down mode when the core is off, when all register contents is lost.

The Alternate function table also has the pull state which is applied after a power down.

Bit(s)	Field Name	Description	Type	Reset

31-2	---	Unused	R	0
1-0	PUD	PUD - GPIO Pin Pull-up/down 00 = Off – disable pull-up/down 01 = Enable Pull Down control 10 = Enable Pull Up control 11 = Reserved *Use in conjunction with GPPUDCLK0/1/2	R/W	0

Table 6-28 – GPIO Pull-up/down Register (GPPUD)

GPIO Pull-up/down Clock Registers (GPPUDCLKn)

SYNOPSIS

The GPIO Pull-up/down Clock Registers control the actuation of internal pull-downs on the respective GPIO pins. These registers must be used in conjunction with the GPPUD register to effect GPIO Pull-up/down changes. The following sequence of events is required:

1. Write to GPPUD to set the required control signal (i.e. Pull-up or Pull-Down or neither to remove the current Pull-up/down)
2. Wait 150 cycles – this provides the required set-up time for the control signal
3. Write to GPPUDCLK0/1 to clock the control signal into the GPIO pads you wish to modify – NOTE only the pads which receive a clock will be modified, all others will retain their previous state.
4. Wait 150 cycles – this provides the required hold time for the control signal
5. Write to GPPUD to remove the control signal
6. Write to GPPUDCLK0/1 to remove the clock

Bit(s)	Field Name	Description	Type	Reset
(31-0)	PUDCLKn (n=0..31)	0 = No Effect 1 = Assert Clock on line *(n)* *Must be used in conjunction with GPPUD	R/W	0

Table 6-29 – GPIO Pull-up/down Clock Register 0

Bit(s)	Field Name	Description	Type	Reset
31-22	-	Reserved	R	0
21-0	PUDCLKn (n=32..53)	0 = No Effect 1 = Assert Clock on line *(n)* *Must be used in conjunction with GPPUD	R/W	0

Table 6-30 – GPIO Pull-up/down Clock Register 1

6.2 Alternative Function Assignments

Every GPIO pin can carry an alternate function. Up to 6 alternate function are available but not every pin has that many alternate functions. The table below gives a quick over view.

	Pull	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5
GPIO0	High	SDA0	SA5	<reserved>			
GPIO1	High	SCL0	SA4	<reserved>			
GPIO2	High	SDA1	SA3	<reserved>			
GPIO3	High	SCL1	SA2	<reserved>			
GPIO4	High	GPCLK0	SA1	<reserved>			ARM_TDI
GPIO5	High	GPCLK1	SA0	<reserved>			ARM_TDO
GPIO6	High	GPCLK2	SOE_N / SE	<reserved>			ARM_RTCK
GPIO7	High	SPI0_CE1_N	SWE_N / SRW_N	<reserved>			
GPIO8	High	SPI0_CE0_N	SD0	<reserved>			
GPIO9	Low	SPI0_MISO	SD1	<reserved>			
GPIO10	Low	SPI0_MOSI	SD2	<reserved>			
GPIO11	Low	SPI0_SCLK	SD3	<reserved>			
GPIO12	Low	PWM0	SD4	<reserved>			ARM_TMS
GPIO13	Low	PWM1	SD5	<reserved>			ARM_TCK
GPIO14	Low	TXD0	SD6	<reserved>			TXD1
GPIO15	Low	RXD0	SD7	<reserved>			RXD1
GPIO16	Low	<reserved>	SD8	<reserved>	CTS0	SPI1_CE2_N	CTS1
GPIO17	Low	<reserved>	SD9	<reserved>	RTS0	SPI1_CE1_N	RTS1
GPIO18	Low	PCM_CLK	SD10	<reserved>	BSCSL SDA / MOSI	SPI1_CE0_N	PWM0
GPIO19	Low	PCM_FS	SD11	<reserved>	BSCSL SCL / SCLK	SPI1_MISO	PWM1
GPIO20	Low	PCM_DIN	SD12	<reserved>	BSCSL / MISO	SPI1_MOSI	GPCLK0
GPIO21	Low	PCM_DOUT	SD13	<reserved>	BSCSL / CE_N	SPI1_SCLK	GPCLK1
GPIO22	Low	<reserved>	SD14	<reserved>	SD1_CLK	ARM_TRST	
GPIO23	Low	<reserved>	SD15	<reserved>	SD1_CMD	ARM_RTCK	
GPIO24	Low	<reserved>	SD16	<reserved>	SD1_DAT0	ARM_TDO	
GPIO25	Low	<reserved>	SD17	<reserved>	SD1_DAT1	ARM_TCK	
GPIO26	Low	<reserved>	<reserved>	<reserved>	SD1_DAT2	ARM_TDI	
GPIO27	Low	<reserved>	<reserved>	<reserved>	SD1_DAT3	ARM_TMS	
GPIO28	-	SDA0	SA5	PCM_CLK	<reserved>		
GPIO29	-	SCL0	SA4	PCM_FS	<reserved>		
GPIO30	Low	<reserved>	SA3	PCM_DIN	CTS0		CTS1
GPIO31	Low	<reserved>	SA2	PCM_DOUT	RTS0		RTS1
GPIO32	Low	GPCLK0	SA1	<reserved>	TXD0		TXD1
GPIO33	Low	<reserved>	SA0	<reserved>	RXD0		RXD1
GPIO34	High	GPCLK0	SOE_N / SE	<reserved>	<reserved>		
GPIO35	High	SPI0_CE1_N	SWE_N / SRW_N		<reserved>		
GPIO36	High	SPI0_CE0_N	SD0	TXD0	<reserved>		
GPIO37	Low	SPI0_MISO	SD1	RXD0	<reserved>		
GPIO38	Low	SPI0_MOSI	SD2	RTS0	<reserved>		
GPIO39	Low	SPI0_SCLK	SD3	CTS0	<reserved>		
GPIO40	Low	PWM0	SD4		<reserved>	SPI2_MISO	TXD1
	Pull	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5

GPIO41	Low	PWM1	SD5	<reserved>	<reserved>	SPI2_MOSI	RXD1
GPIO42	Low	GPCLK1	SD6	<reserved>	<reserved>	SPI2_SCLK	RTS1
GPIO43	Low	GPCLK2	SD7	<reserved>	<reserved>	SPI2_CE0_N	CTS1
GPIO44	-	GPCLK1	SDA0	SDA1	<reserved>	SPI2_CE1_N	
GPIO45	-	PWM1	SCL0	SCL1	<reserved>	SPI2_CE2_N	
GPIO46	High	<Internal>					
GPIO47	High	<Internal>					
GPIO48	High	<Internal>					
GPIO49	High	<Internal>					
GPIO50	High	<Internal>					
GPIO51	High	<Internal>					
GPIO52	High	<Internal>					
GPIO53	High	<Internal>					

Table 6-31 GPIO Pins Alternative Function Assignment

Entries which are white should ***not*** be used. These may have unexpected results as some of these have special functions used in test mode. e.g. they may drive the output with high frequency signals.

Special function legend:

Name	Function	See section
SDA0	BSC[6] master 0 data line	BSC
SCL0	BSC master 0 clock line	BSC
SDA1	BSC master 1 data line	BSC
SCL1	BSC master 1 clock line	BSC
GPCLK0	General purpose Clock 0	**<TBD>**
GPCLK1	General purpose Clock 1	**<TBD>**
GPCLK2	General purpose Clock 2	**<TBD>**
SPI0_CE1_N	SPI0 Chip select 1	SPI
SPI0_CE0_N	SPI0 Chip select 0	SPI
SPI0_MISO	SPI0 MISO	SPI
SPI0_MOSI	SPI0 MOSI	SPI
SPI0_SCLK	SPI0 Serial clock	SPI
PWMx	Pulse Width Modulator 0..1	Pulse Width Modulator
TXD0	UART 0 Transmit Data	UART
RXD0	UART 0 Receive Data	UART
CTS0	UART 0 Clear To Send	UART
RTS0	UART 0 Request To Send	UART
PCM_CLK	PCM clock	PCM Audio
PCM_FS	PCM Frame Sync	PCM Audio
PCM_DIN	PCM Data in	PCM Audio
PCM_DOUT	PCM data out	PCM Audio
SAx	Secondary mem Address bus	Secondary Memory Interface
SOE_N / SE	Secondary mem. Controls	Secondary Memory Interface
SWE_N / SRW_N	Secondary mem. Controls	Secondary Memory Interface
SDx	Secondary mem. data bus	Secondary Memory Interface
BSCSL SDA / MOSI	BSC slave Data, SPI salve MOSI	BSC ISP slave
BSCSL SCL / SCLK	BSC slave Clock, SPI slave clock	BSC ISP slave
BSCSL - / MISO	BSC <not used>,SPI MISO	BSC ISP slave
BSCSL - / CE_N	BSC <not used>, SPI CSn	BSC ISP slave

[6] The Broadcom Serial Control bus is a proprietary bus compliant with the Philips® I2C bus/interface

Name	Function	See section
SPI1_CEx_N	SPI1 Chip select 0-2	Auxiliary I/O
SPI1_MISO	SPI1 MISO	Auxiliary I/O
SPI1_MOSI	SPI1 MOSI	Auxiliary I/O
SPI1_SCLK	SPI1 Serial clock	Auxiliary I/O
TXD0	UART 1 Transmit Data	Auxiliary I/O
RXD0	UART 1 Receive Data	Auxiliary I/O
CTS0	UART 1 Clear To Send	Auxiliary I/O
RTS0	UART 1 Request To Send	Auxiliary I/O
SPI2_CEx_N	SPI2 Chip select 0-2	Auxiliary I/O
SPI2_MISO	SPI2 MISO	Auxiliary I/O
SPI2_MOSI	SPI2 MOSI	Auxiliary I/O
SPI2_SCLK	SPI2 Serial clock	Auxiliary I/O
ARM_TRST	ARM JTAG reset	**<TBD>**
ARM_RTCK	ARM JTAG return clock	**<TBD>**
ARM_TDO	ARM JTAG Data out	**<TBD>**
ARM_TCK	ARM JTAG Clock	**<TBD>**
ARM_TDI	ARM JTAG Data in	**<TBD>**
ARM_TMS	ARM JTAG Mode select	**<TBD>**

6.3 General Purpose GPIO Clocks

The General Purpose clocks can be output to GPIO pins. They run from the peripherals clock sources and use clock generators with noise-shaping MASH dividers. These allow the GPIO clocks to be used to drive audio devices.

The fractional divider operates by periodically dropping source clock pulses, therefore the output frequency will periodically switch between:

$$\frac{source\_frequency}{DIVI} \quad \& \quad \frac{source\_frequency}{DIVI+1}$$

Jitter is therefore reduced by increasing the source clock frequency. In applications where jitter is a concern, the fastest available clock source should be used.

The General Purpose clocks have MASH noise-shaping dividers which push this fractional divider jitter out of the audio band.

MASH noise-shaping is incorporated to push the fractional divider jitter out of the audio band if required. The MASH can be programmed for 1, 2 or 3-stage filtering. MASH filter, the frequency is spread around the requested frequency and the user must ensure that the module is not exposed to frequencies higher than 25MHz. Also, the MASH filter imposes a low limit on the range of DIVI.

MASH	min DIVI	min output freq	average output freq	max output freq
0 (int divide)	1	source / (DIVI)	source / (DIVI)	source / (DIVI)
1	2	source / (DIVI)	source / (DIVI + DIVF / 1024)	source / (DIVI + 1)
2	3	source / (DIVI - 1)	source / (DIVI + DIVF / 1024)	source / (DIVI + 2)
3	5	source / (DIVI - 3)	source / (DIVI + DIVF / 1024)	source / (DIVI + 4)

Table 6-32 Effect of MASH Filter on Frequency

The following example illustrates the spreading of output clock frequency resulting from the use of the MASH filter. Note that the spread is greater for lower divisors.

PLL freq (MHz)	target freq (MHz)	MASH	divisor	DIVI	DIVF	min freq (MHz)	ave freq (MHz)	max freq (MHz)	error
650	18.32	0	35.480	35	492	18.57	18.57	18.57	ok
650	18.32	1	35.480	35	492	18.06	18.32	18.57	ok
650	18.32	2	35.480	35	492	17.57	18.32	19.12	ok
650	18.32	3	35.480	35	492	16.67	18.32	20.31	ok
400	18.32	0	21.834	21	854	19.05	19.05	19.05	ok
400	18.32	1	21.834	21	854	18.18	18.32	19.05	ok
400	18.32	2	21.834	21	854	17.39	18.32	20.00	ok
400	18.32	3	21.834	21	854	16.00	18.32	22.22	ok
200	18.32	0	10.917	10	939	20.00	20.00	20.00	ok
200	18.32	1	10.917	10	939	18.18	18.32	20.00	ok
200	18.32	2	10.917	10	939	16.67	18.32	22.22	ok
200	18.32	3	10.917	10	939	14.29	18.32	28.57	error

Table 6-33 Example of Frequency Spread when using MASH Filtering

It is beyond the scope of this specification to describe the operation of a MASH filter or to determine under what conditions the available levels of filtering are beneficial.

Operating Frequency

The maximum operating frequency of the General Purpose clocks is ~125MHz at 1.2V but this will be reduced if the GPIO pins are heavily loaded or have a capacitive load.

Clock Manager General Purpose Clocks Control (CM_GP0CTL, GP1CTL & GP2CTL)

Address 0x 7e10 1070 CM_GP0CTL
0x 7e10 1078 CM_GP1CTL
0x 7e10 1080 CM_GP2CTL

Bit Number	Field Name	Description	Read/ Write	Reset
31-24	PASSWD	Clock Manager password "5a"	W	0
23-11	-	Unused	R	0
10-9	MASH	MASH control 0 = integer division 1 = 1-stage MASH (equivalent to non-MASH dividers) 2 = 2-stage MASH 3 = 3-stage MASH To avoid lock-ups and glitches do not change this control while BUSY=1 and do not change this control at the same time as asserting ENAB.	R/W	0
8	FLIP	Invert the clock generator output This is intended for use in test/debug only. Switching this control will generate an edge on the clock generator output. To avoid output glitches do not switch this control while BUSY=1.	R/W	0
7	BUSY	Clock generator is running Indicates the clock generator is running. To avoid glitches and lock-ups, clock sources and setups must not be changed while this flag is set.	R	0
6	-	Unused	R	0
5	KILL	Kill the clock generator 0 = no action 1 = stop and reset the clock generator This is intended for test/debug only. Using this control may cause a glitch on the clock generator output.	R/W	0
4	ENAB	Enable the clock generator This requests the clock to start or stop without glitches. The output clock will not stop immediately because the cycle must be allowed to complete to avoid glitches. The BUSY flag will go low when the final cycle is completed.	R/W	0
3-0	SRC	Clock source 0 = GND 1 = oscillator 2 = testdebug0 3 = testdebug1 4 = PLLA per 5 = PLLC per 6 = PLLD per 7 = HDMI auxiliary 8-15 = GND To avoid lock-ups and glitches do not change this control while BUSY=1 and do not change this control at the same time as asserting ENAB.	R/W	0

Table 6-34 General Purpose Clocks Control

Clock Manager General Purpose Clock Divisors (CM_GP0DIV, CM_GP1DIV & CM_GP2DIV)

Address 0x 7e10 1074 CM_GP0DIV
0x 7e10 107c CM_GP1DIV
0x 7e10 1084 CM_GP2DIV

Bit Number	Field Name	Description	Read/ Write	Reset
31-24	PASSWD	Clock Manager password "5a"	W	0
23-12	DIVI	Integer part of divisor This value has a minimum limit determined by the MASH setting. See text for details. To avoid lock-ups and glitches do not change this control while BUSY=1.	R/W	0
11-0	DIVF	Fractional part of divisor To avoid lock-ups and glitches do not change this control while BUSY=1.	R/W	0

Table 6-35 General Purpose Clock Divisors

7 Interrupts

7.1 Introduction

The ARM has two types of interrupt sources:

1. Interrupts coming from the GPU peripherals.
2. Interrupts coming from local ARM control peripherals.

The ARM processor gets three types of interrupts:

1. Interrupts from ARM specific peripherals.
2. Interrupts from GPU peripherals.
3. Special events interrupts.

The ARM specific interrupts are:

- One timer.
- One Mailbox.
- Two Doorbells.
- Two GPU halted interrupts.
- Two Address/access error interrupt

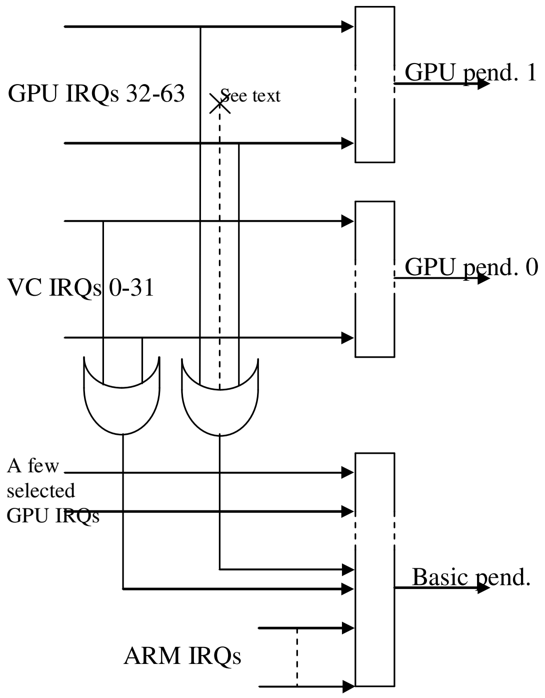
The Mailbox and Doorbell registers are not for general usage.

For each interrupt source (ARM or GPU) there is an interrupt enable bit (read/write) and an interrupt pending bit (Read Only). All interrupts generated by the arm control block are level sensitive interrupts. Thus all interrupts remain asserted until disabled or the interrupt source is cleared.

Default the interrupts from doorbell 0,1 and mailbox 0 go to the ARM this means that these resources should be written by the GPU and read by the ARM. The opposite holds for doorbells 2, 3 and mailbox 1.

7.2 Interrupt pending.

An interrupt vector module has NOT been implemented. To still have adequate interrupt processing the interrupt pending bits are organized as follows:



There are three interrupt pending registers.

One basic pending register and two GPU pending registers.

Basic pending register.

The basic pending register has interrupt pending bits for the ARM specific interrupts .

To speed up the interrupt processing it also has a number of selected GPU interrupts which are deemed most likely to be required in ARM drivers.

Further there are two special GPU pending bits which tell if any of the two other pending registers has bits set, one bit if a GPU interrupt 0-31 is pending, a second bit if a GPU interrupt 32-63 is pending. The 'selected GPU interrupts' on the basic pending registers are NOT taken into account for these two status bits. So the two pending 0,1 status bits tell you that 'there are more interrupt which you have not seen yet'.

GPU pending registers.

There are two GPU pending registers with one bit per GPU interrupt source.

7.3 Fast Interrupt (FIQ).

The ARM also supports a Fast Interrupt (FIQ). One interrupt sources can be selected to be connected to the ARM FIQ input. There is also one FIQ enable. An interrupt which is selected as FIQ should have its normal interrupt enable bit cleared. Otherwise an normal and a FIQ interrupt will be fired at the same time. Not a good idea!

7.4 Interrupt priority.

There is no priority for any interrupt. If one interrupt is much more important then all others it can be routed to the FIQ. Any remaining interrupts have to be processed by polling the pending registers. It is up to the ARM software to device a strategy. e.g. First start looking for specific pending bits or process them all shifting one bit at a time.

As interrupt may arrive whilst this process is ongoing the usual care for any 'race-condition critical' code must be taken. The following ARM assembly code has been proven to work:

```
  .macro get_irqnr_preamble, base, tmp
  ldr  \base, =IO_ADDRESS(ARMCTRL_IC_BASE)
  .endm

  .macro get_irqnr_and_base, irqnr, irqstat, base, tmp
  ldr  \irqstat, [\base, #(ARM_IRQ_PEND0 - ARMCTRL_IC_BASE)] @ get masked status
  mov  \irqnr, #(ARM_IRQ0_BASE + 31)
  and  \tmp, \irqstat, #0x300           @ save bits 8 and 9
  bics \irqstat, \irqstat, #0x300       @ clear bits 8 and 9, and test
  bne  1010f

  tst  \tmp, #0x100
  ldrne \irqstat, [\base, #(ARM_IRQ_PEND1 - ARMCTRL_IC_BASE)]
  movne \irqnr, #(ARM_IRQ1_BASE + 31)
  @ Mask out the interrupts also present in PEND0 - see SW-5809
  bicne \irqstat, #((1<<7) | (1<<9) | (1<<10))
  bicne \irqstat, #((1<<18) | (1<<19))
  bne  1010f

  tst  \tmp, #0x200
  ldrne \irqstat, [\base, #(ARM_IRQ_PEND2 - ARMCTRL_IC_BASE)]
  movne \irqnr, #(ARM_IRQ2_BASE + 31)
  @ Mask out the interrupts also present in PEND0 - see SW-5809
  bicne \irqstat, #((1<<21) | (1<<22) | (1<<23) | (1<<24) | (1<<25))
  bicne \irqstat, #((1<<30))
  beq 1020f

1010:
  @ For non-zero x, LSB(x) = 31 - CLZ(x^(x-1))
  @ N.B. CLZ is an ARM5 instruction.
  sub  \tmp, \irqstat, #1
  eor  \irqstat, \irqstat, \tmp
  clz  \tmp, \irqstat
  sub  \irqnr, \tmp

1020: @ EQ will be set if no irqs pending
```

```
.endm
```

7.5 Registers

The base address for the ARM interrupt register is 0x7E00B000.

Registers overview:

Address offset[7]	Name	Notes
0x200	IRQ basic pending	
0x204	IRQ pending 1	
0x208	IRQ pending 2	
0x20C	FIQ control	
0x210	Enable IRQs 1	
0x214	Enable IRQs 2	
0x218	Enable Basic IRQs	
0x21C	Disable IRQs 1	
0x220	Disable IRQs 2	
0x224	Disable Basic IRQs	

The following is a table which lists all interrupts which can come from the peripherals which can be handled by the ARM.

[7] This is the offset which needs to be added to the base address to get the full hardware address.

ARM peripherals interrupts table.

#	IRQ 0-15	#	IRQ 16-31	#	IRQ 32-47	#	IRQ 48-63
0		16		32		48	smi
1		17		33		49	gpio_int[0]
2		18		34		50	gpio_int[1]
3		19		35		51	gpio_int[2]
4		20		36		52	gpio_int[3]
5		21		37		53	i2c_int
6		22		38		54	spi_int
7		23		39		55	pcm_int
8		24		40		56	
9		25		41		57	uart_int
10		26		42		58	
11		27		43	i2c_spi_slv_int	59	
12		28		44		60	
13		29	Aux int	45	pwa0	61	
14		30		46	pwa1	62	
15		31		47		63	

The table above has many empty entries. These should not be enabled as they will interfere with the GPU operation.

ARM peripherals interrupts table.

0	ARM Timer
1	ARM Mailbox
2	ARM Doorbell 0
3	ARM Doorbell 1
4	GPU0 halted (Or GPU1 halted if bit 10 of control register 1 is set)
5	GPU1 halted
6	Illegal access type 1
7	Illegal access type 0

Basic pending register.

The basic pending register shows which interrupt are pending. To speed up interrupts processing, a number of 'normal' interrupt status bits have been added to this register. This makes the *'IRQ pending base'* register different from the other *'base'* interrupt registers

Name: IRQ pend base		Address: 0x200	Reset: 0x000
Bit(s)	**R/W**	**Function**	
31:21	-	<unused>	
20	R	GPU IRQ 62	

Name: IRQ pend base		Address: 0x200	Reset: 0x000
19	R	GPU IRQ 57	
18	R	GPU IRQ 56	
17	R	GPU IRQ 55	
16	R	GPU IRQ 54	
15	R	GPU IRQ 53	
14	R	GPU IRQ 19	
13	R	GPU IRQ 18	
12	R	GPU IRQ 10	
11	R	GPU IRQ 9	
10	R	GPU IRQ 7	
9	R	One or more bits set in pending register 2	
8	R	One or more bits set in pending register 1	
7	R	Illegal access type 0 IRQ pending	
6	R	Illegal access type 1 IRQ pending	
5	R	GPU1 halted IRQ pending	
4	R	GPU0 halted IRQ pending (Or GPU1 halted if bit 10 of control register 1 is set)	
3	R	ARM Doorbell 1 IRQ pending	
2	R	ARM Doorbell 0 IRQ pending	
1	R	ARM Mailbox IRQ pending	
0	R	ARM Timer IRQ pending	

GPU IRQ x (10,11..20)

These bits are direct interrupts from the GPU. They have been selected as interrupts which are most likely to be useful to the ARM. The GPU interrupt selected are 7, 9, 10, 18, 19, 53,54,55,56,57,62. For details see the *GPU interrupts table*.

Bits set in pending registers (8,9)

These bits indicates if there are bits set in the pending 1/2 registers. The pending 1/2 registers hold ALL interrupts 0..63 from the GPU side. Some of these 64 interrupts are also connected to the basic pending register. Any bit set in pending register 1/2 which is NOT connected to the basic pending register causes bit 8 or 9 to set. Status bits 8 and 9 should be seen as "There are some interrupts pending which you don't know about. They are in pending register 1 /2."

Illegal access type-0 IRQ (7)

This bit indicate that the address/access error line from the ARM processor has generated an interrupt. That signal is asserted when either an address bit 31 or 30 was high or when an access was seen on the ARM Peripheral bus. The status of that signal can be read from Error/HALT status register bit 2.

Illegal access type-1 IRQ (6)

This bit indicates that an address/access error is seen in the ARM control has generated an interrupt. That can either be an address bit 29..26 was high or when a burst access was seen on the GPU Peripheral bus. The status of that signal can be read from Error/HALT status register bits 0 and 1.

GPU-1 halted IRQ (5)

This bit indicate that the GPU-1 halted status bit has generated an interrupt. The status of that signal can be read from Error/HALT status register bits 4.

GPU-0 (or any GPU) halted IRQ (4)

This bit indicate that the GPU-0 halted status bit has generated an interrupt. The status of that signal can be read from Error/HALT status register bits 3.

In order to allow a fast interrupt (FIQ) routine to cope with GPU 0 OR GPU-1 there is a bit in control register 1 which, if set will also route a GPU-1 halted status on this bit.

Standard peripheral IRQs (0,1,2,3)

These bits indicate if an interrupt is pending for one of the ARM control peripherals.

GPU pending 1 register.

Name: IRQ pend base		Address: 0x204	Reset: 0x000
Bit(s)	**R/W**	**Function**	
31:0	R	IRQ pending source 31:0 (See IRQ table above)	

This register holds ALL interrupts 0..31 from the GPU side. Some of these interrupts are also connected to the basic pending register. Any interrupt status bit in here which is NOT connected to the basic pending will also cause bit 8 of the basic pending register to be set. That is all bits except 7, 9, 10, 18, 19.

GPU pending 2 register.

Name: IRQ pend base		Address: 0x208	Reset: 0x000
Bit(s)	**R/W**	**Function**	
31:0	R	IRQ pending source 63:32 (See IRQ table above)	

This register holds ALL interrupts 32..63 from the GPU side. Some of these interrupts are also connected to the basic pending register. Any interrupt status bit in here which is NOT connected to the basic pending will also cause bit 9 of the basic pending register to be set. That is all bits except . register bits 21..25, 30 (Interrupts 53..57,62).

FIQ register.

The FIQ register control which interrupt source can generate a FIQ to the ARM. Only a single interrupt can be selected.

Name: FIQ		Address: 0x20C	Reset: 0x000
Bit(s)	**R/W**	**Function**	
31:8	R	<unused>	
7	R	FIQ enable. Set this bit to 1 to enable FIQ generation. If set to 0 bits 6:0 are don't care.	
6:0	R/W	Select FIQ Source	

FIQ Source.

The FIQ source values 0-63 correspond to the GPU interrupt table. (See above)

The following values can be used to route ARM specific interrupts to the FIQ vector/routine:

FIQ index	Source
0-63	GPU Interrupts (See GPU IRQ table)
64	ARM Timer interrupt
65	ARM Mailbox interrupt
66	ARM Doorbell 0 interrupt
67	ARM Doorbell 1 interrupt
68	GPU0 Halted interrupt (Or GPU1)
69	GPU1 Halted interrupt
70	Illegal access type-1 interrupt
71	Illegal access type-0 interrupt
72-127	Do Not Use

Interrupt enable register 1.

Name: IRQ enable 1		Address: 0x210	Reset: 0x000
Bit(s)	**R/W**	**Function**	
31:0	R/Wbs	Set to enable IRQ source 31:0 (See IRQ table above)	

Writing a 1 to a bit will set the corresponding IRQ enable bit. All other IRQ enable bits are unaffected. Only bits which are enabled can be seen in the interrupt pending registers. There is no provision here to see if there are interrupts which are pending but not enabled.

Interrupt enable register 2.

Name: IRQ enable 2		Address: 0x214	Reset: 0x000
Bit(s)	R/W	Function	
31:0	R/Wbs	Set to enable IRQ source 63:32 (See IRQ table above)	

Writing a 1 to a bit will set the corresponding IRQ enable bit. All other IRQ enable bits are unaffected. Only bits which are enabled can be seen in the interrupt pending registers. There is no provision here to see if there are interrupts which are pending but not enabled.

Base Interrupt enable register.

Name: IRQ enable 3		Address: 0x218	Reset: 0x000
Bit(s)	R/W	Function	
31:8	R/Wbs	<Unused>	
7	R/Wbs	Set to enable Access error type -0 IRQ	
6	R/Wbs	Set to enable Access error type -1 IRQ	
5	R/Wbs	Set to enable GPU 1 Halted IRQ	
4	R/Wbs	Set to enable GPU 0 Halted IRQ	
3	R/Wbs	Set to enable ARM Doorbell 1 IRQ	
2	R/Wbs	Set to enable ARM Doorbell 0 IRQ	
1	R/Wbs	Set to enable ARM Mailbox IRQ	
0	R/Wbs	Set to enable ARM Timer IRQ	

Writing a 1 to a bit will set the corresponding IRQ enable bit. All other IRQ enable bits are unaffected. Again only bits which are enabled can be seen in the basic pending register. There is no provision here to see if there are interrupts which are pending but not enabled.

Interrupt disable register 1.

Name: IRQ disable 1		Address: 0x21C	Reset: 0x000
Bit(s)	R/W	Function	
31:0	R/Wbc	Set to disable IRQ source 31:0 (See IRQ table above)	

Writing a 1 to a bit will clear the corresponding IRQ enable bit. All other IRQ enable bits are unaffected.

Interrupt disable register 2.

Name: IRQ disable 2		Address: 0x220	Reset: 0x000
Bit(s)	**R/W**	**Function**	
31:0	R/Wbc	Set to disable IRQ source 63:32 (See IRQ table above)	

Writing a 1 to a bit will clear the corresponding IRQ enable bit. All other IRQ enable bits are unaffected.

Base disable register.

Name: IRQ disable 3		Address: 0x224	Reset: 0x000
Bit(s)	**R/W**	**Function**	
31:8	-	<Unused>	
7	R/Wbc	Set to disable Access error type -0 IRQ	
6	R/Wbc	Set to disable Access error type -1 IRQ	
5	R/Wbc	Set to disable GPU 1 Halted IRQ	
4	R/Wbc	Set to disable GPU 0 Halted IRQ	
3	R/Wbc	Set to disable ARM Doorbell 1 IRQ	
2	R/Wbc	Set to disable ARM Doorbell 0 IRQ	
1	R/Wbc	Set to disable ARM Mailbox IRQ	
0	R/Wbc	Set to disable ARM Timer IRQ	

Writing a 1 to a bit will clear the corresponding IRQ enable bit. All other IRQ enable bits are unaffected.

8 PCM / I2S Audio

The PCM audio interface is an APB peripheral providing input and output of telephony or high quality serial audio streams. It supports many classic PCM formats including I2S.

The PCM audio interface has 4 interface signals;

PCM_CLK	- bit clock.
PCM_FS	- frame sync signal.
PCM_DIN	- serial data input.
PCM_DOUT	- serial data output.

PCM is a serial format with a single bit data_in and single bit data_out. Data is always serialised MS-bit first.

The frame sync signal (PCM_FS) is used to delimit the serial data into individual frames. The length of the frame and the size and position of the frame sync are fully programmable.

Frames can contain 1 or 2 audio/data channels in each direction. Each channel can be between 8 and 32 bits wide and can be positioned anywhere within the frame as long as the two channels don't overlap. The channel format is separately programmable for transmit and receive directions.

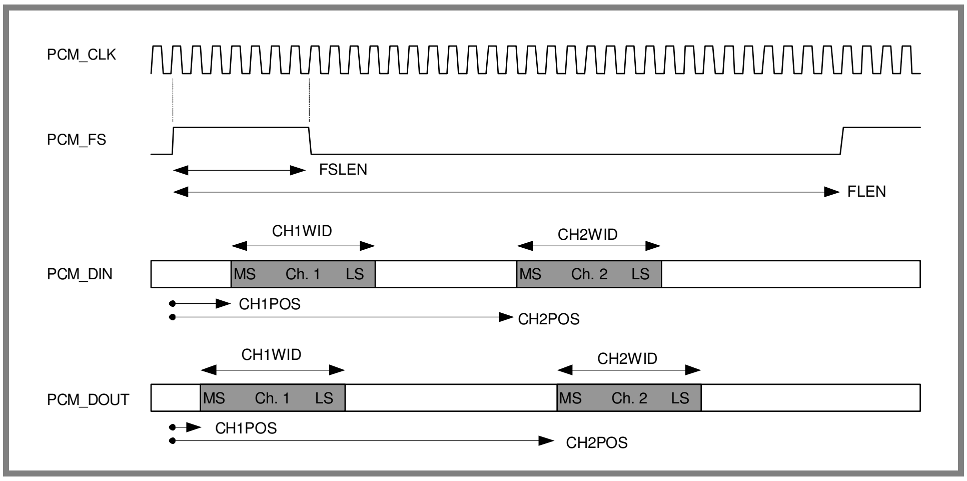


Figure 8-1 PCM Audio Interface Typical Timing

The PCM_CLK can be asynchronous to the bus APB clock and can be logically inverted if required.

The direction of the PCM_CLK and PCM_FS signals can be individually selected, allowing the interface to act as a master or slave device.

The input interface is also capable of supporting up to 2 PDM microphones, as an alternative to the classic PCM input format, in conjunction with a PCM output.

8.1 Block Diagram

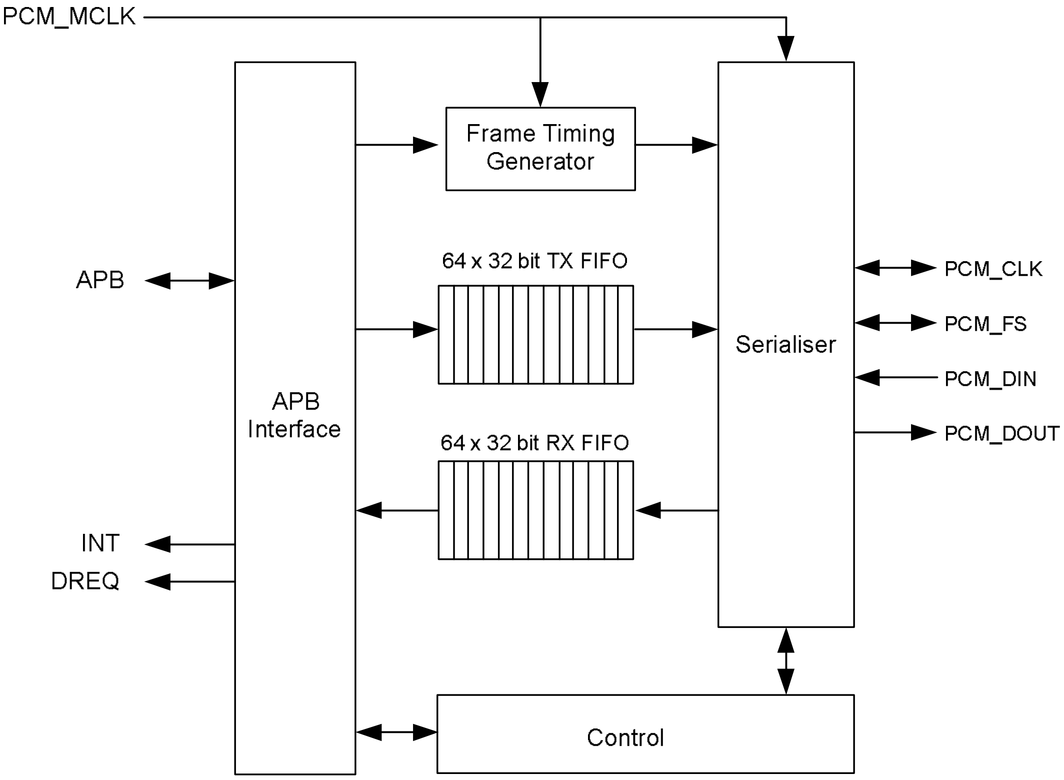


Figure 8-2 PCM Audio Interface Block Diagram

The PCM audio interface contains separate transmit and receive FIFOs. Note that if the frame contains two data channels, they must share the same FIFO and so the channel data will be interleaved. The block can be driven using simple polling, an interrupt based method or direct DMA control.

8.2 Typical Timing

Figure 8-1 shows typical interface timing and indicates the flexibility that the peripheral offers.

Normally PCM output signals change on the rising edge of PCM_CLK and input signals are sampled on its falling edge. The frame sync is considered as a data signal and sampled in the same way.

The front end of the PCM audio interface is run off the PCM_CLK and the PCM signals are timed against this clock. However, the polarity of the PCM_CLK can be physically inverted, in which case the edges are reversed.

In clock master mode (CLKM=0), the PCM_CLK is an output and is driven from the PCM_MCLK clock input.

In clock slave mode (CLKM=1), the PCM_CLK is an input, supplied by some external clock source.

In frame sync master mode (FSM=0), the PCM_FS is internally generated and is treated as a data output that changes on the positive edge of the clock. The length and polarity of the frame sync is fully programmable and it can be used as a standard frame sync signal, or as an L-R signal for I2S.

In frame sync slave mode (FSM=1), the PCM_FS is treated as a data input and is sampled on the negative edge of PCM_CLK. The first clock of a frame is taken as the first clock period where PCM_FS is sampled as a 1 following a period or periods where it was previously a 0. The PCM audio interface locks onto the incoming frame sync and uses this to indicate where the data channels are positioned. The precise timing at the start of frame is shown in Figure 8-3.

Note that in frame sync slave mode there are two synchronising methods. The legacy method is used when the frame length = 0. In this case the internal frame logic has to detect the incoming PCM_FS signal and reset the internal frame counter at the start of every frame. The logic relies on the PCM_FS to indicate the length of the frame and so can cope with adjacent frames of different lengths. However, this creates a short timing path that will corrupt the PCM_DOUT for one specific frame/channel setting.

The preferred method is to set the frame length to the expected length. Here the incoming PCM_FS is used to resynchronise the internal frame counter and this eliminates the short timing path.

8.3 Operation

The PCM interface runs asynchronously at the PCM_CLK rate and automatically transfers transmit and receive data across to the internal APB clock domain. The control registers are NOT synchronised and should be programmed before the device is enabled and should NOT be changed whilst the interface is running.

Only the EN, RXON and TXON bits of the PCMCS register are synchronised across the PCM - APB clock domain and are allowed to be changed whilst the interface is running.

The EN bit is a global power-saving enable. The TXON and RXON bits enable transmit and receive, and the interface is running whenever either TXON or RXON is enabled.

In operation, the PCM format is programmed by setting the appropriate frame length, frame sync, channel position values, and signal polarity controls. The transmit FIFO should be preloaded with data and the interface can then be enabled and started, and will run continuously until stopped. If the transmit FIFO becomes empty or the receive FIFO becomes full, the RXERR or TXERR error flags will be set, but the interface will just continue. If the RX FIFO overflows, new samples are discarded and if the TX FIFO underflows, zeros are transmitted.

Normally channel data is read or written into the appropriate FIFO as a single word. If the channel is less than 32 bits, the data is right justified and should be padded with zeros. If the RXSEX bit is set then the received data is sign extended up to the full 32 bits. When a frame is programmed to have two data channels, then each channel is written/read as a separate word in the FIFO, producing an interleaved data stream. When initialising the interface, the first word read out of the TX FIFO will be used for the first channel, and the data from the first channel on the first frame to be received will be the first word written into the RX FIFO.

If a FIFO error occurs in a two channel frame, then channel synchronisation may be lost which may result in a left right audio channel swap. RXSYNC and TXSYNC status bits are provided to help determine if channel slip has occurred. They indicate if the number of words in the FIFO is a multiple of a full frame (taking into account where we are in the current frame being transferred). This assumes that an integer number of frames data has been sent/read from the FIFOs.

If a frame is programmed to have two data channels and the packed mode bits are set (FRXP FTXP) then the FIFOs are configured so that each word contains the data for both channels (2x 16 bit samples). In this mode each word written to the TX FIFO contains 2 16 bit samples, and the Least Significant sample is transmitted first. Each word read from the RX FIFO will contain the data received from 2 channels, the first channel received will be in the Least Significant half of the word. If the channels size is less than 16 bits, the TX data will be truncated and RX data will be padded to 16 bits with zeros.

Note that data is always serialised MS-bit first. This is well-established behaviour in both PCM and I2S.

If the PDM input mode is enabled then channel 1 is sampled on the negative edge of PCM_CLK whilst channel 2 is sampled on the positive edge of PCM_CLK.

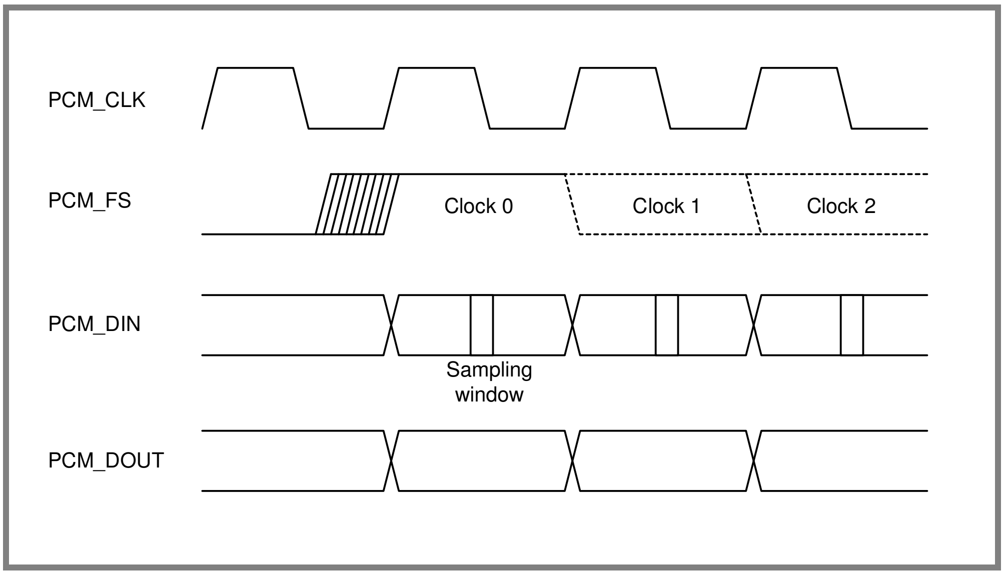


Figure 8-3 Timing at Start of Frame

Note that the precise timing of FS (when it is an input) is not clearly defined and it may change state before or after the positive edge of the clock. Here the first clock of the frame is defined as the clock period where the PCM_FS is sampled (negative edge) as a 1 where it was previously sampled as a 0.

8.4 Software Operation

8.4.1 Operating in Polled mode

Set the EN bit to enable the PCM block. Set all operational values to define the frame and channel settings. Assert RXCLR and/or TXCLR wait for 2 PCM clocks to ensure the FIFOs are reset. The SYNC bit can be used to determine when 2 clocks have passed. Set RXTHR/TXTHR to determine the FIFO thresholds.

If transmitting, ensure that sufficient sample words have been written to PCMFIFO before transmission is started. Set TXON and/or RXON to begin operation. Poll TXW writing sample words to PCMFIFO and RXR reading sample words from PCMFIFO until all data is transferred.

8.4.2 Operating in Interrupt mode

a) Set the EN bit to enable the PCM block. Set all operational values to define the frame and channel settings. Assert RXCLR and/or TXCLR wait for 2 PCM clocks to ensure the FIFOs are reset. The SYNC bit can be used to determine when 2 clocks have passed. Set RXTHR/TXTHR to determine the FIFO thresholds.

b) Set INTR and/or INTT to enable interrupts.

c) If transmitting, ensure that sufficient sample words have been written to PCMFIFO before transmission is started. Set TXON and/or RXON to begin operation.

d) When an interrupt occurs, check RXR. If this is set then one or more sample words are available in PCMFIFO. If TXW is set then one or more sample words can be sent to PCMFIFO.

8.4.3 DMA

a) Set the EN bit to enable the PCM block. Set all operational values to define the frame and channel settings. Assert RXCLR and/or TXCLR wait for 2 PCM clocks to ensure the FIFOs are reset. The SYNC bit can be used to determine when 2 clocks have passed.

b) Set DMAEN to enable DMA DREQ generation and set RXREQ/TXREQ to determine the FIFO thresholds for the DREQs. If required, set TXPANIC and RXPANIC to determine the level at which the DMA should increase its AXI priority,

c) In the DMA controllers set the correct DREQ channels, one for RX and one for TX. Start the DMA which should fill the TX FIFO.

d) Set TXON and/or RXON to begin operation.

8.5 Error Handling.

In all software operational modes, the possibility of FIFO over or under run exists. Should this happen when using 2 channels per frame, there is a risk of losing sync with the channel data stored in the FIFO. If this happens and is not detected and corrected, then the data channels may become swapped.

The FIFO's will automatically detect an error condition caused by a FIFO over or under-run and this will set the appropriate latching error bit in the control/status register. Writing a '1' back to this error bit will clear the latched flag.

In a system using a polled operation, the error bits can be checked manually. For an interrupt or DMA based system, setting the INTE bit will cause the PCM interface to generate an interrupt when an error is detected.

If a FIFO error occurs during operation in which 2 data channels are being used then the synchronisation of the data may be lost. This can be recovered by either of these two methods:

a) Disable transmit and receive (TXON and RXON =0). Clear the FIFO's (RXCLR and TXCLR =1). Note that it may take up to 2 PCM clocks for the FIFOs to be physically cleared after initiating a clear. Then preload the transmit FIFO and restart transmission. This of course loses the data in the FIFO and further interrupts the data flow to the external device.

b) Examine the TXSYNC and RXSYNC flags. These flags indicate if the amount of data in the FIFO is a whole number of frames, automatically taking into account where we are in the current frame being transmitted or received. Thus, providing an even number of samples was read or written to the FIFOs, then if the flags are set then this indicates that a single word needs to be written or read to adjust the data. Normal exchange of data can then proceed (where the first word in a data pair is for channel 1). This method should cause less disruption to the data stream.

8.6 PDM Input Mode Operation

The PDM input mode is capable of interfacing with two digital half-cycle PDM microphones and implements a 4th order CIC decimation filter with a selectable decimation factor. The clock input of the microphones is shared with the PCM output codec and it should be configured to provide the correct clock rate for the microphones. As a result it may be necessary to add a number of padding bits into the PCM output and configure the output codec to allow for this.

When using the PDM input mode the bit width and the rate of the data received will depend on the decimation factor used. Once the data has been read from the peripheral a further decimation and filtering stage will be required and can be implemented in software. The software filter should also correct the droop introduced by the CIC filter stage. Similarly a DC correction stage should also be employed.

PDMN	PCM_CLK (MHz)	Peripheral Output Format	OSR	Fs
0 (N=16)	3.072	16 bits unsigned	4	48kHz
1 (N=32)	3.072	20 bits unsigned	2	48kHz

Table 8-1 PDM Input Mode Configuration

8.7 GRAY Code Input Mode Operation

GRAY mode is used for an incoming data stream only. GRAY mode is selected by setting the enable bit (EN) in the PCM_GRAY register.

In this mode data is received on the PCM_DIN (data) and the PCM_FS (strobe) pins. The data is expected to be in data/strobe format. In this mode data is detected when either the data or the strobe change state. As each bit is received it is written into the RX buffer and when 32 bits are received they are written out to the RXFIFO as a 32 bit word. In order for this mode to work the user must program a PCM clock rate which is 4 times faster then the gray data rate. Also the gray coded data input signals should be clean.

The normal RXREQ and RXTHR FIFO levels will apply as for normal PCM received data.

If a message is received that is not a multiple of 32 bits, any data in the RX Buffer can be flushed out by setting the flush bit (FLUSH). Once set, this bit will read back as zero until the flush operation has completed. This may take several cycles as the APB clock may be many times faster than the PCM clock. Once the flush has occurred, the bits are packed up to 32 bits with zeros and written out to the RXFIFO. The flushed field (FLUSHED) will indicate how many of bits of this word are valid.

Note that to get an accurate indication of the number of bits currently in the rx shift register (RXLEVEL) the APB clock must be at least 2x the PCM_CLK.

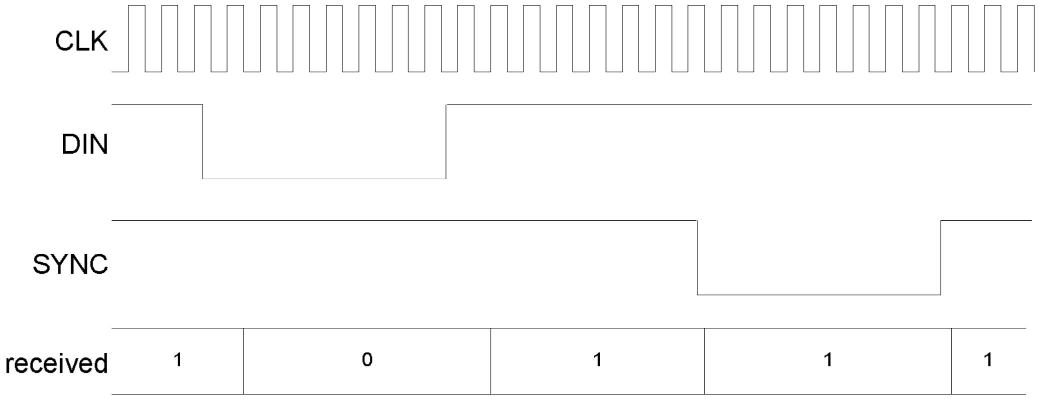


Figure 8-4 Gray mode input format

8.8 PCM Register Map

There is only PCM module in the BCM2835. The PCM base address for the registers is 0x7E203000.

PCM Address Map			
Address Offset	**Register Name**	**Description**	**Size**
0x0	CS_A	PCM Control and Status	32
0x4	FIFO_A	PCM FIFO Data	32
0x8	MODE_A	PCM Mode	32
0xc	RXC_A	PCM Receive Configuration	32
0x10	TXC_A	PCM Transmit Configuration	32
0x14	DREQ_A	PCM DMA Request Level	32

0x18	INTEN_A	PCM Interrupt Enables	32
0x1c	INTSTC_A	PCM Interrupt Status & Clear	32
0x20	GRAY	PCM Gray Mode Control	32

CS_A Register

Synopsis This register contains the main control and status bits for the PCM. The bottom 3 bits of this register can be written to whilst the PCM is running. The remaining bits cannot.

Bit(s)	Field Name	Description	Type	Reset
31:26		***Reserved** - Write as 0, read as don't care*		
25	STBY	RAM Standby This bit is used to control the PCM Rams standby mode. By default this bit is 0 causing RAMs to start initially in standby mode. Rams should be released from standby prior to any transmit/receive operation. Allow for at least 4 PCM clock cycles to take effect. This may or may not be implemented, depending upon the RAM libraries being used.	RW	0x0
24	SYNC	PCM Clock sync helper. This bit provides a software synchronisation mechanism to allow the software to detect when 2 PCM clocks have occurred. It takes 2 PCM clocks before the value written to this bit will be echoed back in the read value.	RW	0x0
23	RXSEX	RX Sign Extend 0 = No sign extension. 1 = Sign extend the RX data. When set, the MSB of the received data channel (as set by the CHxWID parameter) is repeated in all the higher data bits up to the full 32 bit data width.	RW	0x0
22	RXF	RX FIFO is Full 0 = RX FIFO can accept more data. 1 = RX FIFO is full and will overflow if more data is received.	RO	0x0

21	TXE	TX FIFO is Empty 0 = TX FIFO is not empty. 1 = TX FIFO is empty and underflow will take place if no more data is written.	RO	0x1
20	RXD	Indicates that the RX FIFO contains data 0 = RX FIFO is empty. 1 = RX FIFO contains at least 1 sample.	RO	0x0
19	TXD	Indicates that the TX FIFO can accept data 0 = TX FIFO is full and so cannot accept more data. 1 = TX FIFO has space for at least 1 sample.	RO	0x1
18	RXR	Indicates that the RX FIFO needs reading 0 = RX FIFO is less than RXTHR full. 1 = RX FIFO is RXTHR or more full. This is cleared by reading sufficient data from the RX FIFO.	RO	0x0
17	TXW	Indicates that the TX FIFO needs Writing 0 = TX FIFO is at least TXTHR full. 1 = TX FIFO is less then TXTHR full. This is cleared by writing sufficient data to the TX FIFO.	RO	0x1
16	RXERR	RX FIFO Error 0 = FIFO has had no errors. 1 = FIFO has had an under or overflow error. This flag is cleared by writing a 1.	RW	0x0
15	TXERR	TX FIFO Error 0 = FIFO has had no errors. 1 = FIFO has had an under or overflow error. This flag is cleared by writing a 1.	RW	0x0
14	RXSYNC	RX FIFO Sync 0 = FIFO is out of sync. The amount of data left in the FIFO is not a multiple of that required for a frame. This takes into account if we are halfway through the frame. 1 = FIFO is in sync.	RO	0x0
13	TXSYNC	TX FIFO Sync 0 = FIFO is out of sync. The amount of data left in the FIFO is not a multiple of that required for a frame. This takes into account if we are halfway through the frame. 1 = FIFO is in sync.	RO	0x0
12:10		***Reserved*** *- Write as 0, read as don't care*		

9	DMAEN	DMA DREQ Enable 0 = Don t generate DMA DREQ requests. 1 = Generates a TX DMA DREQ requests whenever the TX FIFO level is lower than TXREQ or generates a RX DMA DREQ when the RX FIFO level is higher than RXREQ.	RW	0x0
8:7	RXTHR	Sets the RX FIFO threshold at which point the RXR flag is set 00 = set when we have a single sample in the RX FIFO 01 = set when the RX FIFO is at least full 10 = set when the RX FIFO is at least 11 = set when the RX FIFO is full	RW	0x0
6:5	TXTHR	Sets the TX FIFO threshold at which point the TXW flag is set 00 = set when the TX FIFO is empty 01 = set when the TX FIFO is less than full 10 = set when the TX FIFO is less than full 11 = set when the TX FIFO is full but for one sample	RW	0x0
4	RXCLR	Clear the RX FIFO . Assert to clear RX FIFO. This bit is self clearing and is always read as clear Note that it will take 2 PCM clocks for the FIFO to be physically cleared.	WO	0x0
3	TXCLR	Clear the TX FIFO Assert to clear TX FIFO. This bit is self clearing and is always read as clear. Note that it will take 2 PCM clocks for the FIFO to be physically cleared.	WO	0x0
2	TXON	Enable transmission 0 = Stop transmission. This will stop immediately if possible or else at the end of the next frame. The TX FIFO can still be written to to preload data. 1 = Start transmission. This will start transmitting at the start of the next frame. Once enabled, the first data read from the TX FIFO will be placed in the first channel of the frame, thus ensuring proper channel synchronisation. The frame counter will be started whenever TXON or RXON are set. This bit can be written whilst the interface is running.	RW	0x0

1	RXON	Enable reception. 0 = Disable reception. This will stop on the next available frame end. RX FIFO data can still be read. 1 = Enable reception. This will be start receiving at the start of the next frame. The first channel to be received will be the first word written to the RX FIFO. This bit can be written whilst the interface is running.	RW	0x0
0	EN	Enable the PCM Audio Interface 0 = The PCM interface is disabled and most logic is gated off to save power. 1 = The PCM Interface is enabled. This bit can be written whilst the interface is running.	RW	0x0

FIFO_A Register

Synopsis This is the FIFO port of the PCM. Data written here is transmitted, and received data is read from here.

Bit(s)	Field Name	Description	Type	Reset
31:0		***Reserved*** *- Write as 0, read as don't care*		

MODE_A Register

Synopsis This register defines the basic PCM Operating Mode. It is used to configure the frame size and format and whether the PCM is in master or slave modes for its frame sync or clock. This register cannot be changed whilst the PCM is running.

Bit(s)	Field Name	Description	Type	Reset
31:29		***Reserved*** *- Write as 0, read as don't care*		

28	CLK_DIS	PCM Clock Disable 1 = Disable the PCM Clock. This cleanly disables the PCM clock. This enables glitch free clock switching between an internal and an uncontrollable external clock. The PCM clock can be disabled, and then the clock source switched, and then the clock re-enabled. 0 = Enable the PCM clock.	RW	0x0
27	PDMN	PDM Decimation Factor (N) 0 = Decimation factor 16. 1 = Decimation factor 32. Sets the decimation factor of the CIC decimation filter.	RW	0x0
26	PDME	PDM Input Mode Enable 0 = Disable PDM (classic PCM input). 1 = Enable PDM input filter. Enable CIC filter on input pin for PDM inputs. In order to receive data RXON must also be set.	RW	0x0
25	FRXP	Receive Frame Packed Mode 0 = The data from each channel is written into the RX FIFO. 1 = The data from both RX channels is merged (1st channel is in the LS half) and then written to the RX FIFO as a single 2x16 bit packed mode word. First received channel in the frame goes into the LS half word. If the received data is larger than 16 bits, the upper bits are truncated. The maximum channel size is 16 bits.	RW	0x0
24	FTXP	Transmit Frame Packed Mode 0 = Each TX FIFO word is written into a single channel. 1 = Each TX FIFO word is split into 2 16 bit words and used to fill both data channels in the same frame. The maximum channel size is 16 bits. The LS half of the word is used in the first channel of the frame.	RW	0x0
23	CLKM	PCM Clock Mode 0 = Master mode. The PCM CLK is an output and drives at the MCLK rate. 1 = Slave mode. The PCM CLK is an input.	RW	0x0

22	CLKI	Clock Invert this logically inverts the PCM_CLK signal. 0 = Outputs change on rising edge of clock, inputs are sampled on falling edge. 1 = Outputs change on falling edge of clock, inputs are sampled on rising edge.	RW	0x0
21	FSM	Frame Sync Mode 0 = Master mode. The PCM_FS is an output and we generate the frame sync. 1 = Slave mode. The PCM_FS is an input and we lock onto the incoming frame sync signal.	RW	0x0
20	FSI	Frame Sync Invert This logically inverts the frame sync signal. 0 = In master mode, FS is normally low and goes high to indicate frame sync. In slave mode, the frame starts with the clock where FS is a 1 after being a 0. 1 = In master mode, FS is normally high and goes low to indicate frame sync. In slave mode, the frame starts with the clock where FS is a 0 after being a 1.	RW	0x0
19:10	FLEN	Frame Length Sets the frame length to (FLEN+1) clocks. Used only when FSM == 0. 1 = frame length of 2 clocks. 2 = frame length of 3 clocks. etc	RW	0x0
9:0	FSLEN	Frame Sync Length Sets the frame sync length to (FSLEN) clocks. This is only used when FSM == 0. PCM_FS will remain permanently active if FSLEN >= FLEN. 0 = frame sync pulse is off. 1 = frame sync pulse is 1 clock wide. etc	RW	0x0

RXC_A Register

Synopsis Sets the Channel configurations for Receiving. This sets the position and width of the 2 receive channels within the frame. The two channels cannot overlap, however they channel 1 can come after channel zero, although the first data will always be from the first channel in the frame. Channels can also straddle the frame begin end boundary as that is set by the frame sync position. This register cannot be changed whilst the PCM is running.

Bit(s)	Field Name	Description	Type	Reset

31	CH1WEX	Channel 1 Width Extension Bit This is the MSB of the channel 1 width (CH1WID). It allows widths greater than 24 bits to be programmed and is added here to keep backwards compatibility with older versions of the PCM	RW	0x0
30	CH1EN	Channel 1 Enable 0 = Channel 1 disabled and no data is received from channel 1 and written to the RX FIFO. 1 = Channel 1 enabled.	RW	0x0
29:20	CH1POS	Channel 1 Position This sets the bit clock at which the first bit (MS bit) of channel 1 data occurs in the frame. 0 indicates the first clock of frame.	RW	0x0
19:16	CH1WID	Channel 1 Width This sets the width of channel 1 in bit clocks. This field has been extended with the CH1WEX bit giving a total width of (CH1WEX* 16) + CH1WID + 8. The Maximum supported width is 32 bits. 0 = 8 bits wide 1 = 9 bits wide	RW	0x0
15	CH2WEX	Channel 2 Width Extension Bit This is the MSB of the channel 2 width (CH2WID). It allows widths greater than 24 bits to be programmed and is added here to keep backwards compatibility with older versions of the PCM	RW	0x0
14	CH2EN	Channel 2 Enable 0 = Channel 2 disabled and no data is received from channel 2 and written to the RX FIFO. 1 = Channel 2 enabled.	RW	0x0
13:4	CH2POS	Channel 2 Position This sets the bit clock at which the first bit (MS bit) of channel 2 data occurs in the frame. 0 indicates the first clock of frame.	RW	0x0
3:0	CH2WID	Channel 2 Width This sets the width of channel 2 in bit clocks. This field has been extended with the CH2WEX bit giving a total width of (CH2WEX* 16) + CH2WID + 8. The Maximum supported width is 32 bits. 0 = 8 bits wide 1 = 9 bits wide	RW	0x0

TXC_A Register

Synopsis Sets the Channel configurations for Transmitting. This sets the position and width of the 2 transmit channels within the frame. The two channels cannot overlap, however they channel 1 can come after channel zero, although the first data will always be used in the first channel in the frame. Channels can also straddle the frame begin end boundary as that is set by the frame sync position. This register cannot be changed whilst the PCM is running.

Bit(s)	Field Name	Description	Type	Reset
31	CH1WEX	Channel 1 Width Extension Bit This is the MSB of the channel 1 width (CH1WID). It allows widths greater than 24 bits to be programmed and is added here to keep backwards compatibility with older versions of the PCM	RW	0x0
30	CH1EN	Channel 1 Enable 0 = Channel 1 disabled and no data is taken from the TX FIFO and transmitted on channel 1. 1 = Channel 1 enabled.	RW	0x0
29:20	CH1POS	Channel 1 Position This sets the bit clock at which the first bit (MS bit) of channel 1 data occurs in the frame. 0 indicates the first clock of frame.	RW	0x0
19:16	CH1WID	Channel 1 Width This sets the width of channel 1 in bit clocks. This field has been extended with the CH1WEX bit giving a total width of (CH1WEX* 16) + CH1WID + 8. The Maximum supported width is 32 bits. 0 = 8 bits wide 1 = 9 bits wide	RW	0x0
15	CH2WEX	Channel 2 Width Extension Bit This is the MSB of the channel 2 width (CH2WID). It allows widths greater than 24 bits to be programmed and is added here to keep backwards compatibility with older versions of the PCM	RW	0x0
14	CH2EN	Channel 2 Enable 0 = Channel 2 disabled and no data is taken from the TX FIFO and transmitted on channel 2. 1 = Channel 2 enabled.	RW	0x0

13:4	CH2POS	Channel 2 Position This sets the bit clock at which the first bit (MS bit) of channel 2 data occurs in the frame. 0 indicates the first clock of frame.	RW	0x0
3:0	CH2WID	Channel 2 Width This sets the width of channel 2 in bit clocks. This field has been extended with the CH2WEX bit giving a total width of (CH2WEX* 16) + CH2WID + 8. The Maximum supported width is 32 bits. 0 = 8 bits wide 1 = 9 bits wide	RW	0x0

DREQ_A Register

Synopsis Set the DMA DREQ and Panic thresholds. The PCM drives 2 DMA controls back to the DMA, one for the TX channel and one for the RX channel. DMA DREQ is used to request the DMA to perform another transfer, and DMA Panic is used to tell the DMA to use its panic level of priority when requesting thins on the AXI bus. This register cannot be changed whilst the PCM is running.

Bit(s)	Field Name	Description	Type	Reset
31		***Reserved*** *- Write as 0, read as don't care*		
30:24	TX_PANIC	TX Panic Level This sets the TX FIFO Panic level. When the level is below this the PCM will assert its TX DMA Panic signal.	RW	0x10
23		***Reserved*** *- Write as 0, read as don't care*		
22:16	RX_PANIC	RX Panic Level This sets the RX FIFO Panic level. When the level is above this the PCM will assert its RX DMA Panic signal.	RW	0x30
15		***Reserved*** *- Write as 0, read as don't care*		
14:8	TX	TX Request Level This sets the TX FIFO DREQ level. When the level is below this the PCM will assert its DMA DREQ signal to request more data is written to the TX FIFO.	RW	0x30
7		***Reserved*** *- Write as 0, read as don't care*		

6:0	RX	RX Request Level This sets the RX FIFO DREQ level. When the level is above this the PCM will assert its DMA DREQ signal to request that some more data is read out of the RX FIFO.	RW	0x20

INTEN_A Register

Synopsis Set the reasons for generating an Interrupt. This register cannot be changed whilst the PCM is running.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved*** - *Write as 0, read as don't care*		
3	RXERR	RX Error Interrupt Setting this bit enables interrupts from PCM block when RX FIFO error occurs.	RW	0x0
2	TXERR	TX Error Interrupt Setting this bit enables interrupts from PCM block when TX FIFO error occurs.	RW	0x0
1	RXR	RX Read Interrupt Enable Setting this bit enables interrupts from PCM block when RX FIFO level is greater than or equal to the specified RXTHR level.	RW	0x0
0	TXW	TX Write Interrupt Enable Setting this bit enables interrupts from PCM block when TX FIFO level is less than the specified TXTHR level.	RW	0x0

INTSTC_A Register

Synopsis This register is used to read and clear the PCM interrupt status. Writing a 1 to the asserted bit clears the bit. Writing a 0 has no effect.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved*** - *Write as 0, read as don't care*		

3	RXERR	RX Error Interrupt Status / Clear This bit indicates an interrupt occurred on RX FIFO Error. Writing 1 to this bit clears it. Writing 0 has no effect.	RW	0x0
2	TXERR	TX Error Interrupt Status / Clear This bit indicates an interrupt occurred on TX FIFO Error. Writing 1 to this bit clears it. Writing 0 has no effect.	RW	0x0
1	RXR	RX Read Interrupt Status / Clear This bit indicates an interrupt occurred on RX Read. Writing 1 to this bit clears it. Writing 0 has no effect.	RW	0x0
0	TXW	TX Write Interrupt Status / Clear This bit indicates an interrupt occurred on TX Write. Writing 1 to this bit clears it. Writing 0 has no effect.	RW	0x0

GRAY Register

Synopsis This register is used to control the gray mode generation. This is used to put the PCM into a special data/strobe mode. This mode is under 'best effort ' contract.

Bit(s)	Field Name	Description	Type	Reset
31:22		***Reserved*** *- Write as 0, read as don't care*		
21:16	RXFIFOLEVEL	The Current level of the RXFIFO This indicates how many words are currently in the RXFIFO.	RO	0x0
15:10	FLUSHED	The Number of bits that were flushed into the RXFIFO This indicates how many bits were valid when the flush operation was performed. The valid bits are from bit 0 upwards. Non-valid bits are set to zero.	RO	0x0
9:4	RXLEVEL	The Current fill level of the RX Buffer This indicates how many GRAY coded bits have been received. When 32 bits are received, they are written out into the RXFIFO.	RO	0x0

3		***Reserved*** *- Write as 0, read as don't care*		
2	FLUSH	Flush the RX Buffer into the RX FIFO This forces the RX Buffer to do an early write. This is necessary if we have reached the end of the message and we have bits left in the RX Buffer. Flushing will write these bits as a single 32 bit word, starting at bit zero. Empty bits will be packed with zeros. The number of bits written will be recorded in the FLUSHED Field. This bit is written as a 1 to initiate a flush. It will read back as a zero until the flush operation has completed (as the PCM Clock may be very slow).	RW	0x0
1	CLR	Clear the GRAY Mode Logic This Bit will reset all the GRAY mode logic, and flush the RX buffer. It is not self clearing.	RW	0x0
0	EN	Enable GRAY Mode Setting this bit will put the PCM into GRAY mode. In gray mode the data is received on the data in and the frame sync pins. The data is expected to be in data/strobe format.	RW	0x0

9 Pulse Width Modulator

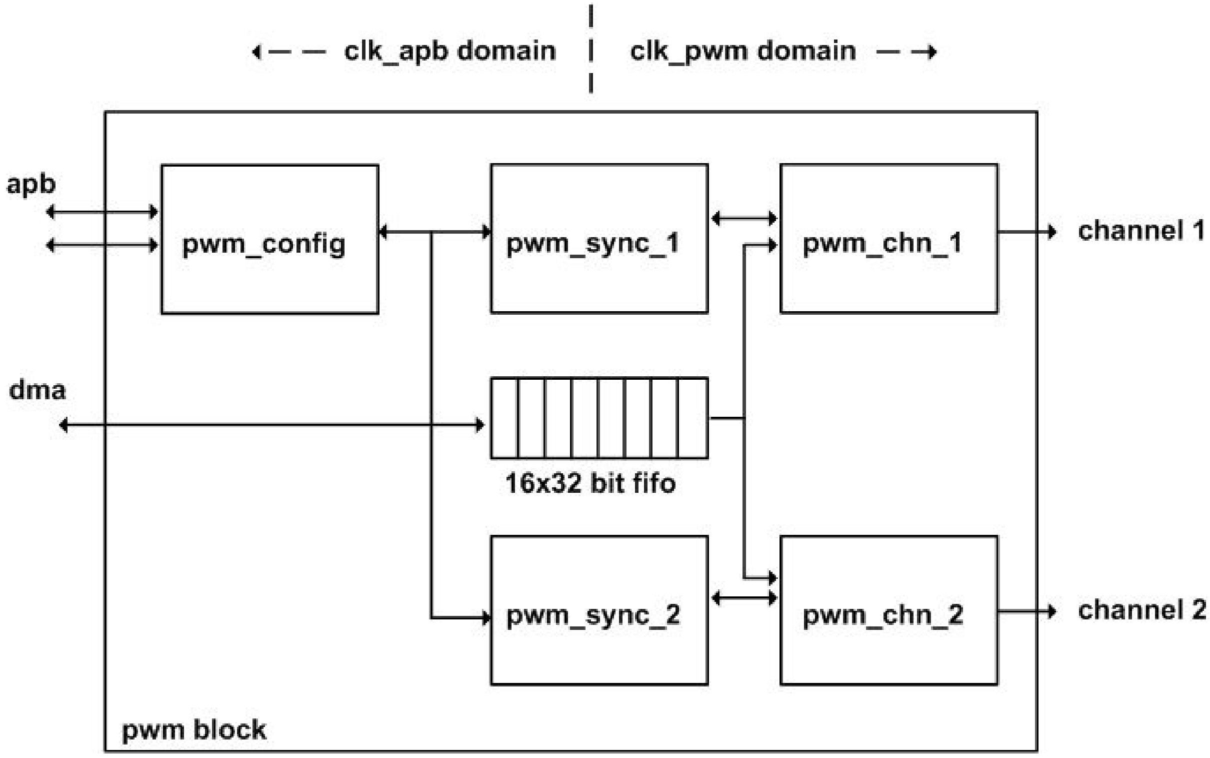
9.1 Overview

This section specifies in detail the functionality provided by the device Pulse Width Modulator (PWM) peripheral.

The PWM controller incorporates the following features:

- Two independent output bit-streams, clocked at a fixed frequency.
- Bit-streams configured individually to output either PWM or a serialised version of a 32-bit word.
- PWM outputs have variable input and output resolutions.
- Serialise mode configured to load data to and/or read data from a FIFO storage block, which can store up to eight 32-bit words.
- Both modes clocked by clk_pwm which is nominally 100MHz, but can be varied by the clock manager.

9.2 Block Diagram



9.3 PWM Implementation

A value represented as a ratio of N/M can be transmitted along a serial channel with pulse width modulation in which the value is represented by the duty cycle of the output signal. To send value N/M within a periodic sequence of M cycles, output should be 1 for N cycles and 0 for (M-N) cycles. The desired sequence should have 1's and 0's spread out as even as possible so that during any arbitrary period of time duty cycle achieves closest approximation of the value. This can be shown in the following table where 4/8 is modulated (N= 4, M= 8).

Bad	0	0	0	0	1	1	1	1	0	0	0	0
Fair	0	0	1	1	0	0	1	1	0	0	1	1
Good	0	1	0	1	0	1	0	1	0	1	0	1

Sequence which gives the 'good' approximation from the table above can be achieved by the following algorithm:

```
1. Set context = 0
2. context = context + N
3.  if (context >= M)
        context = context – M
        send 1
    else
```

where context is a register which stores the result of the addition/subtractions.

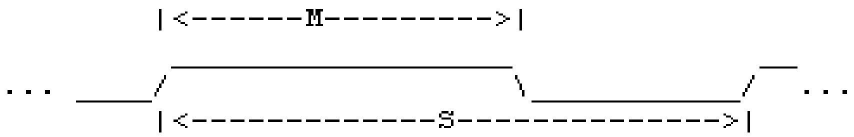
9.4 Modes of Operation

PWM controller consists of two independent channels (pwm_chn in block diagram) which implement the pwm algorithm explained in section 1.3. Each channel can operate in either pwm mode or serialiser mode.

PWM mode: There are two sub-modes in PWM mode: MSEN=0 and MSEN=1.

When MSEN=0, which is the default mode, data to be sent is interpreted as the value N of the algorithm explained above. Number of clock cycles (range) used to send data is the value M of the algorithm. Pulses are sent within this range so that the resulting duty cycle is N/M. Channel sends its output continuously as long as data register is used, or buffer is used and it is not empty.

When MSEN=1, PWM block does not use the algorithm explained above, instead it sends serial data with the M/S ratio as in the picture below. M is the data to be sent, and S is the range. This mode may be preferred if high frequency modulation is not required or has negative effects. Channel sends its output continuously as long as data register is used, or buffer is used and it is not empty.



Serial bit transmission when M/S Mode enabled

Serialiser mode: Each channel is also capable of working as a serialiser. In this mode data written in buffer or the data register is sent serially.

9.5 Quick Reference

- PWM DMA is mapped to DMA channel 5.
- GPIOs are assigned to PWM channels as below. Please refer to GPIO section for further details:

	PWM0	PWM1
GPIO 12	Alt Fun 0	-
GPIO 13	-	Alt Fun 0
GPIO 18	Alt Fun 5	-
GPIO 19	-	Alt Fun 5
GPIO 40	Alt Fun 0	-
GPIO 41	-	Alt Fun 0
GPIO 45	-	Alt Fun 0
GPIO 52	Alt Fun 1	-
GPIO 53	-	Alt Fun 1

- PWM clock source and frequency is controlled in CPRMAN.

9.6 Control and Status Registers

PWM Address Map			
Address Offset	**Register Name**	**Description**	**Size**
0x0	CTL	PWM Control	32
0x4	STA	PWM Status	32
0x8	DMAC	PWM DMA Configuration	32
0x10	RNG1	PWM Channel 1 Range	32
0x14	DAT1	PWM Channel 1 Data	32
0x18	FIF1	PWM FIFO Input	32
0x20	RNG2	PWM Channel 2 Range	32
0x24	DAT2	PWM Channel 2 Data	32

CTL Register

Synopsis PWENi is used to enable/disable the corresponding channel. Setting this bit to 1 enables the channel and transmitter state machine. All registers and FIFO is writable without setting this bit.

MODEi bit is used to determine mode of operation. Setting this bit to 0 enables PWM mode. In this mode data stored in either PWM_DATi or FIFO is transmitted by pulse width modulation within the range defined by PWM_RNGi. When this mode is used MSENi defines whether to use PWM algorithm. Setting MODEi to 1 enables serial mode, in which data stored in either PWM_DATi or FIFO is transmitted serially within the range defined by PWM_RNGi. Data is transmitted MSB first and truncated or zero-padded depending on PWM_RNGi. Default mode is PWM.

RPTLi is used to enable/disable repeating of the last data available in the FIFO just before it empties. When this bit is 1 and FIFO is used, the last available data in the FIFO is repeatedly sent. This may be useful in PWM mode to avoid duty cycle gaps. If the FIFO is not used this bit does not have any effect. Default operation is do-not-repeat.

SBITi defines the state of the output when no transmission takes place. It also defines the zero polarity for the zero padding in serialiser mode. This bit is padded between two consecutive transfers as well as tail of the data when PWM_RNGi is larger than bit depth of data being transferred. this bit is zero by default.

POLAi is used to configure the polarity of the output bit. When set to high the final output is inverted. Default operation is no inversion.

USEFi bit is used to enable/disable FIFO transfer. When this bit is high data stored in the FIFO is used for transmission. When it is low, data written to PWM_DATi is transferred. This bit is 0 as default.

CLRF is used to clear the FIFO. Writing a 1 to this bit clears the FIFO. Writing 0 has no effect. This is a single shot operation and reading the bit always returns 0.

MSENi is used to determine whether to use PWM algorithm or simple M/S ratio transmission. When this bit is high M/S transmission is used. This bit is zero as default. When MODEi is 1, this configuration bit has no effect.

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved*** *- Write as 0, read as don't care*		
15	MSEN2	Channel 2 M/S Enable 0: PWM algorithm is used 1: M/S transmission is used.	RW	0x0
14		***Reserved*** *- Write as 0, read as don't care*		
13	USEF2	Channel 1 Use Fifo 0: Data register is transmitted 1: Fifo is used for transmission	RW	0x0
12	POLA2	Channel 1 Polarity 0 : 0=low 1=high 1: 1=low 0=high	RW	0x0
11	SBIT2	Channel 1 Silence Bit Defines the state of the output when no transmission takes place	RW	0x0

10	RPTL2	Channel 1 Repeat Last Data 0: Transmission interrupts when FIFO is empty 1: Last data in FIFO is transmitted repetedly until FIFO is not empty	RW	0x0
9	MODE2	Channel 1 Mode 0: PWM mode 1: Serialiser mode	RW	0x0
8	PWEN2	Channel 1 Enable 0: Channel is disabled 1: Channel is enabled	RW	0x0
7	MSEN1	Channel 1 M/S Enable 0: PWM algorithm is used 1: M/S transmission is used.	RW	0x0
6	CLRF1	Clear Fifo 1: Clears FIFO 0: Has no effect This is a single shot operation. This bit always reads 0	RO	0x0
5	USEF1	Channel 1 Use Fifo 0: Data register is transmitted 1: Fifo is used for transmission	RW	0x0
4	POLA1	Channel 1 Polarity 0 : 0=low 1=high 1: 1=low 0=high	RW	0x0
3	SBIT1	Channel 1 Silence Bit Defines the state of the output when no transmission takes place	RW	0x0
2	RPTL1	Channel 1 Repeat Last Data 0: Transmission interrupts when FIFO is empty 1: Last data in FIFO is transmitted repetedly until FIFO is not empty	RW	0x0
1	MODE1	Channel 1 Mode 0: PWM mode 1: Serialiser mode	RW	0x0
0	PWEN1	Channel 1 Enable 0: Channel is disabled 1: Channel is enabled	RW	0x0

STA Register

Synopsis FULL1 bit indicates the full status of the FIFO. If this bit is high FIFO is full.
EMPT1 bit indicates the empty status of the FIFO. If this bit is high FIFO is empty.
WERR1 bit sets to high when a write when full error occurs. Software must clear this bit by writing 1. Writing 0 to this bit has no effect.
RERR1 bit sets to high when a read when empty error occurs. Software must clear this bit by writing 1. Writing 0 to this bit has no effect.
GAPOi. bit indicates that there has been a gap between transmission of two consecutive data from FIFO. This may happen when FIFO gets empty after state machine has sent a word and waits for the next. If control bit RPTLi is set to high this event will not occur. Software must clear this bit by writing 1. Writing 0 to this bit has no effect.
BERR sets to high when an error has occurred while writing to registers via APB. This may happen if the bus tries to write successively to same set of registers faster than the synchroniser block can cope with. Multiple switching may occur and contaminate the data during synchronisation. Software should clear this bit by writing 1. Writing 0 to this bit has no effect.
STAi bit indicates the current state of the channel which is useful for debugging purposes. 0 means the channel is not currently transmitting. 1 means channel is transmitting data.

Bit(s)	Field Name	Description	Type	Reset
31:13		***Reserved** - Write as 0, read as don't care*		
12	STA4	Channel 4 State	RW	0x0
11	STA3	Channel 3 State	RW	0x0
10	STA2	Channel 2 State	RW	0x0
9	STA1	Channel 1 State	RW	0x0
8	BERR	Bus Error Flag	RW	0x0
7	GAPO4	Channel 4 Gap Occurred Flag	RW	0x0
6	GAPO3	Channel 3 Gap Occurred Flag	RW	0x0
5	GAPO2	Channel 2 Gap Occurred Flag	RW	0x0
4	GAPO1	Channel 1 Gap Occurred Flag	RW	0x0
3	RERR1	Fifo Read Error Flag	RW	0x0
2	WERR1	Fifo Write Error Flag	RW	0x0

1	EMPT1	Fifo Empty Flag	RW	0x1
0	FULL1	Fifo Full Flag	RW	0x0

DMAC Register

Synopsis ENAB bit is used to start DMA.
PANIC bits are used to determine the threshold level for PANIC signal going active. Default value is 7.
DREQ bits are used to determine the threshold level for DREQ signal going active. Default value is 7.

Bit(s)	Field Name	Description	Type	Reset
31	ENAB	DMA Enable 0: DMA disabled 1: DMA enabled	RW	0x0
30:16		***Reserved*** *- Write as 0, read as don't care*		
15:8	PANIC	DMA Threshold for PANIC signal	RW	0x7
7:0	DREQ	DMA Threshold for DREQ signal	RW	0x7

RNG1 Register

Synopsis This register is used to define the range for the corresponding channel. In PWM mode evenly distributed pulses are sent within a period of length defined by this register. In serial mode serialised data is transmitted within the same period. If the value in PWM_RNGi is less than 32, only the first PWM_RNGi bits are sent resulting in a truncation. If it is larger than 32 excess zero bits are padded at the end of data. Default value for this register is 32.
Note: Channels 3 and 4 are not available in B0 and corresponding Channel Range Registers are ignored.

Bit(s)	Field Name	Description	Type	Reset
31:0	PWM_RNGi	Channel i Range	RW	0x20

DAT1 Register

Synopsis This register stores the 32 bit data to be sent by the PWM Controller when USEFi is 0. In PWM mode data is sent by pulse width modulation: the value of this register defines the number of pulses which is sent within the period defined by PWM_RNGi. In serialiser mode data stored in this register is serialised and transmitted.
Note: Channels 3 and 4 are not available in B0 and corresponding Channel Data Registers are ignored.

Bit(s)	Field Name	Description	Type	Reset
31:0	PWM_DATi	Channel i Data	RW	0x0

FIF1 Register

Synopsis This register is the FIFO input for the all channels. Data written to this address is stored in channel FIFO and if USEFi is enabled for the channel i it is used as data to be sent. This register is write only, and reading this register will always return bus default return value, pwm0 .
When more than one channel is enabled for FIFO usage, the data written into the FIFO is shared between these channels in turn. For example if the word series A B C D E F G H I .. is written to FIFO and two channels are active and configured to use FIFO then channel 1 will transmit words A C E G I .. and channel 2 will transmit words B D F H .. .
Note that requesting data from the FIFO is in locked-step manner and therefore requires tight coupling of state machines of the channels. If any of the channel range (period) value is different than the others this will cause the channels with small range values to wait between words hence resulting in gaps between words. To avoid that, each channel sharing the FIFO should be configured to use the same range value.
Also note that RPTLi are not meaningful when the FIFO is shared between channels as there is no defined channel to own the last data in the FIFO. Therefore sharing channels must have their RPTLi set to zero.
If the set of channels to share the FIFO has been modified after a configuration change, FIFO should be cleared before writing new data.

Bit(s)	Field Name	Description	Type	Reset
31:0	PWM_FIFO	Channel FIFO Input	RW	0x0

RNG2 Register

Synopsis This register is used to define the range for the corresponding channel. In PWM mode evenly distributed pulses are sent within a period of length defined by this register. In serial mode serialised data is transmitted within the same period. If the value in PWM_RNGi is less than 32, only the first PWM_RNGi bits are sent resulting in a truncation. If it is larger than 32 excess zero bits are padded at the end of data. Default value for this register is 32.
Note: Channels 3 and 4 are not available in B0 and corresponding Channel Range Registers are ignored.

Bit(s)	Field Name	Description	Type	Reset
31:0	PWM_RNGi	Channel i Range	RW	0x20

DAT2 Register

Synopsis This register stores the 32 bit data to be sent by the PWM Controller when USEFi is 1. In PWM mode data is sent by pulse width modulation: the value of this register defines the number of pulses which is sent within the period defined by PWM_RNGi. In serialiser mode data stored in this register is serialised and transmitted.
Note: Channels 3 and 4 are not available in B0 and corresponding Channel Data Registers are ignored.

Bit(s)	Field Name	Description	Type	Reset
31:0	PWM_DATi	Channel i Data	RW	0x0

10 SPI

10.1 Introduction

This Serial interface peripheral supports the following features:

- Implements a 3 wire serial protocol, variously called Serial Peripheral Interface (SPI) or Synchronous Serial Protocol (SSP).
- Implements a 2 wire version of SPI that uses a single wire as a bidirectional data wire instead of one for each direction as in standard SPI.
- Implements a LoSSI Master (Low Speed Serial Interface)
- Provides support for polled, interrupt or DMA operation.

10.2 SPI Master Mode

10.2.1 Standard mode

In Standard SPI master mode the peripheral implements the standard 3 wire serial protocol described below.

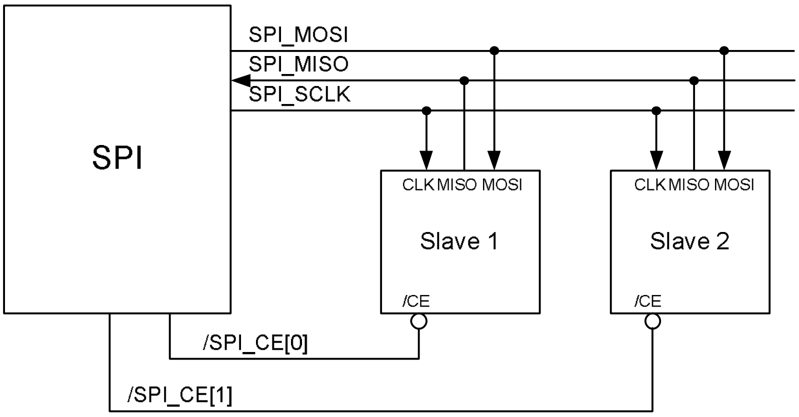
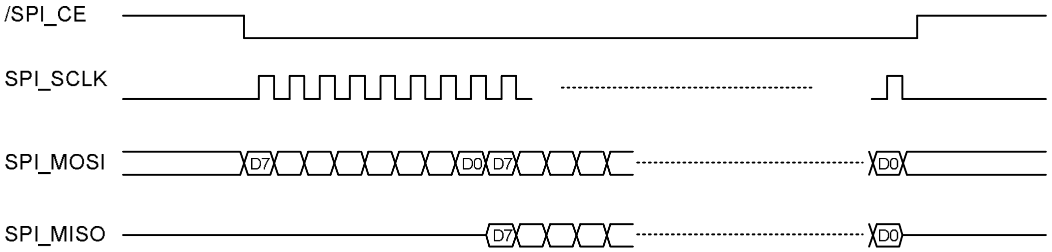


Figure 10-1 SPI Master Typical Usage



Notes
1. Slave enables itself onto SPI_MISO only when it is outputting data. At other times, output is tristate.
2. Different SCLK polarity and phase values are possible. Case illustrated is CPOL = 0, CPHA = 0.
3. Data is transmitted MSB first.
4. Transactions can be from a single byte to hundreds of bytes.

Figure 10-2 SPI Cycle

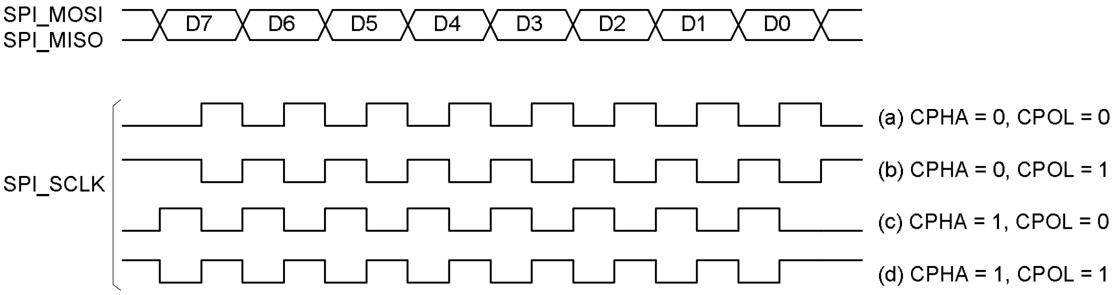


Figure 10-3 Different Clock Polarity/Phase

10.2.2 Bidirectional mode

In bidirectional SPI master mode the same SPI standard is implemented except that a single wire is used for the data (MIMO) instead of the two as in standard mode (MISO and MOSI). Bidirectional mode is used in a similar way to standard mode, the only difference is that before attempting to read data from the slave, you must set the read enable (SPI_REN) bit in the SPI control and status register (SPI_CS). This will turn the bus around, and when you write to the SPI_FIFO register (with junk) a read transaction will take place on the bus, and the read data will appear in the FIFO.

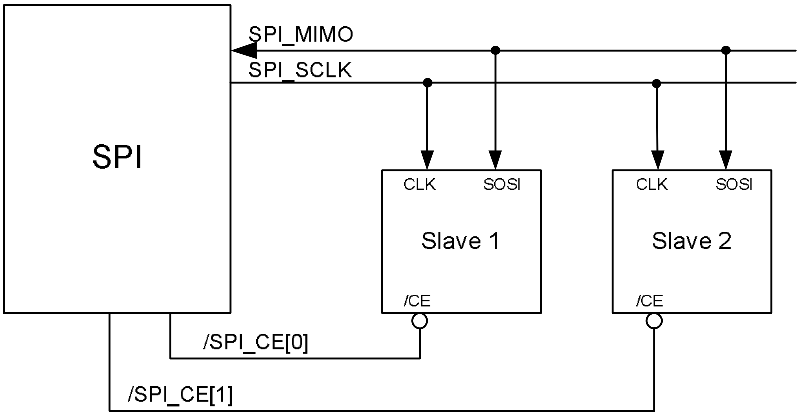


Figure 10-4 Bidirectional SPI Master Typical Usage

10.3 LoSSI mode

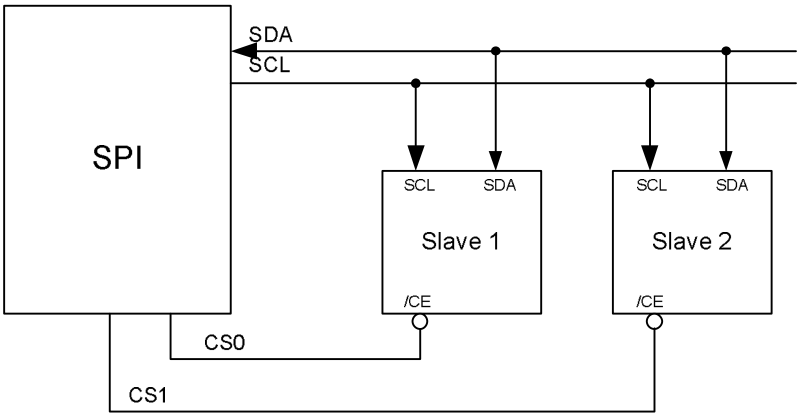
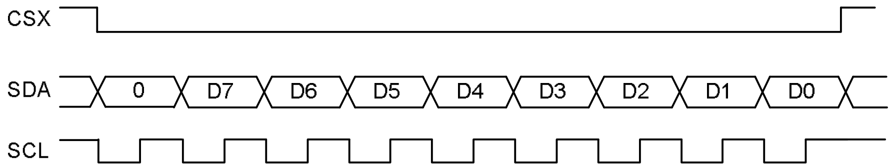


Figure 10-5 LoSSI mode Typical usage

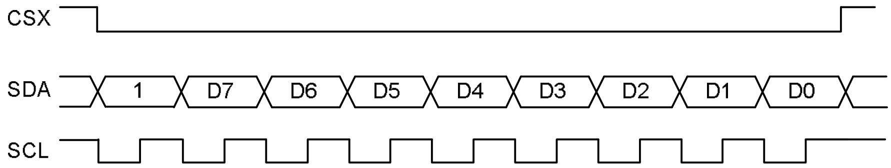
The LoSSI standard allows us to issue commands to peripherals and to transfer data to and from them. LoSSI commands and parameters are 8 bits long, but an extra bit is used to indicate whether the byte is a command or data. This extra bit is set high for a parameter and low for a command. The resulting 9-bit value is serialized to the output. When reading from a LoSSI peripheral the standard allows us to read bytes of data, as well as 24 and 32 bit words.

Commands and parameters are issued to a LoSSI peripheral by writing the 9-bit value of the command or data into the SPI_FIFO register as you would for SPI mode. Reads are automated in that if the serial interface peripheral detects a read command being issued, it will issue the command and complete the read transaction, putting the received data into the FIFO.

10.3.1 Command write

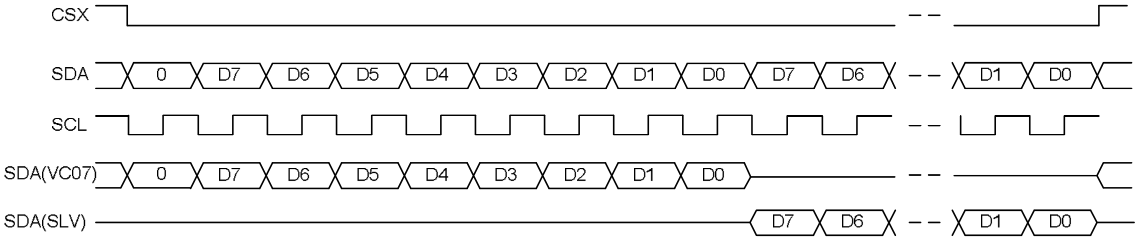


10.3.2 Parameter write



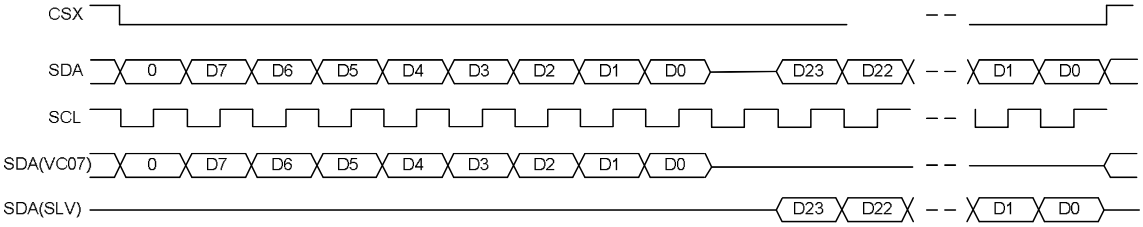
10.3.3 Byte read commands

Byte read commands are 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f, 0xda, 0xdb, 0xdc.



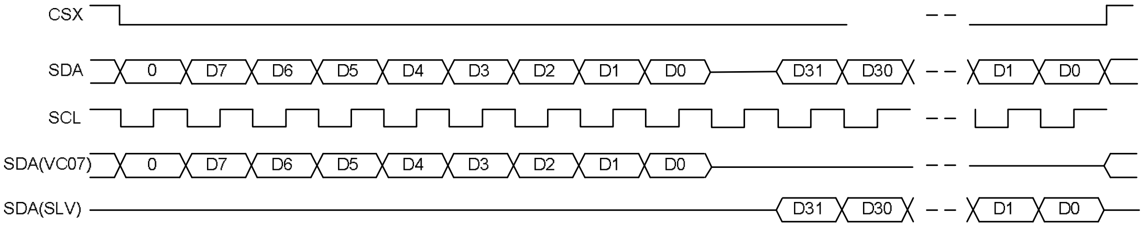
10.3.4 24bit read command

A 24 bit read can be achieved by using the command 0x04.

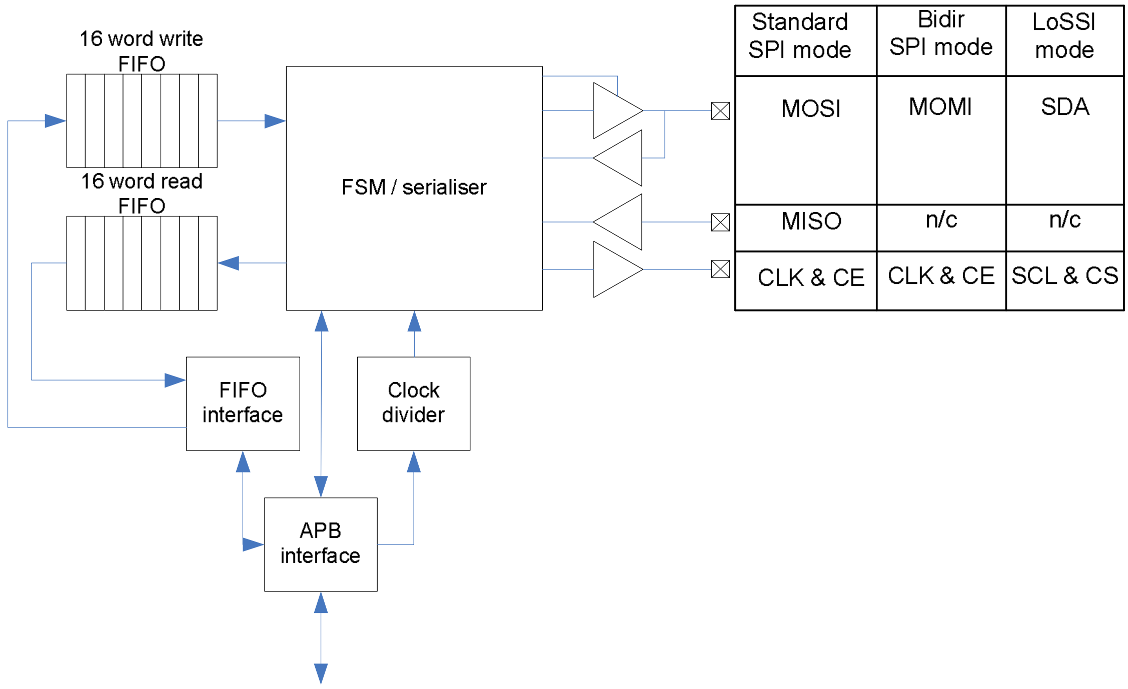


10.3.5 32bit read command

A 32bit read can be achieved by using the command 0x09.



10.4 Block Diagram



Standard SPI mode	Bidir SPI mode	LoSSI mode
MOSI	MOMI	SDA
MISO	n/c	n/c
CLK & CE	CLK & CE	SCL & CS

Figure 10-6 Serial interface Block Diagram

10.5 SPI Register Map

The BCM2835 devices has only one SPI interface of this type. It is referred to in all the documentation as SPI0. It has two additional mini SPI interfaces (SPI1 and SPI2). The specification of those can be found under *2.3 Universal SPI Master (2x)*.

The base address of this SPI0 interface is 0x7E204000.

SPI Address Map			
Address Offset	**Register Name**	**Description**	**Size**
0x0	CS	SPI Master Control and Status	32
0x4	FIFO	SPI Master TX and RX FIFOs	32
0x8	CLK	SPI Master Clock Divider	32

0xc	DLEN	SPI Master Data Length	32
0x10	LTOH	SPI LOSSI mode TOH	32
0x14	DC	SPI DMA DREQ Controls	32

CS Register

Synopsis This register contains the main control and status bits for the SPI.

Bit(s)	Field Name	Description	Type	Reset
31:26		***Reserved*** *- Write as 0, read as don't care*		
25	LEN_LONG	Enable Long data word in Lossi mode if DMA_LEN is set 0= writing to the FIFO will write a single byte 1= wrirng to the FIFO will write a 32 bit word	RW	0x0
24	DMA_LEN	Enable DMA mode in Lossi mode	RW	0x0
23	CSPOL2	Chip Select 2 Polarity 0= Chip select is active low. 1= Chip select is active high.	RW	0x0
22	CSPOL1	Chip Select 1 Polarity 0= Chip select is active low. 1= Chip select is active high.	RW	0x0
21	CSPOL0	Chip Select 0 Polarity 0= Chip select is active low. 1= Chip select is active high.	RW	0x0
20	RXF	RXF - RX FIFO Full 0 = RXFIFO is not full. 1 = RX FIFO is full. No further serial data will be sent/ received until data is read from FIFO.	RO	0x0
19	RXR	RXR RX FIFO needs Reading (full) 0 = RX FIFO is less than full (or not active TA = 0). 1 = RX FIFO is or more full. Cleared by reading sufficient data from the RX FIFO or setting TA to 0.	RO	0x0

18	TXD	TXD TX FIFO can accept Data 0 = TX FIFO is full and so cannot accept more data. 1 = TX FIFO has space for at least 1 byte.	RO	0x1
17	RXD	RXD RX FIFO contains Data 0 = RX FIFO is empty. 1 = RX FIFO contains at least 1 byte.	RO	0x0
16	DONE	Done transfer Done 0 = Transfer is in progress (or not active TA = 0). 1 = Transfer is complete. Cleared by writing more data to the TX FIFO or setting TA to 0.	RO	0x0
15	TE_EN	Unused	RW	0x0
14	LMONO	Unused	RW	0x0
13	LEN	LEN LoSSI enable The serial interface is configured as a LoSSI master. 0 = The serial interface will behave as an SPI master. 1 = The serial interface will behave as a LoSSI master.	RW	0x0
12	REN	REN Read Enable read enable if you are using bidirectional mode. If this bit is set, the SPI peripheral will be able to send data to this device. 0 = We intend to write to the SPI peripheral. 1 = We intend to read from the SPI peripheral.	RW	0x1
11	ADCS	ADCS Automatically Deassert Chip Select 0 = Don t automatically deassert chip select at the end of a DMA transfer chip select is manually controlled by software. 1 = Automatically deassert chip select at the end of a DMA transfer (as determined by SPIDLEN)	RW	0x0
10	INTR	INTR Interrupt on RXR 0 = Don t generate interrupts on RX FIFO condition. 1 = Generate interrupt while RXR = 1.	RW	0x0
9	INTD	INTD Interrupt on Done 0 = Don t generate interrupt on transfer complete. 1 = Generate interrupt when DONE = 1.	RW	0x0

8	DMAEN	DMAEN DMA Enable 0 = No DMA requests will be issued. 1 = Enable DMA operation. Peripheral generates data requests. These will be taken in four-byte words until the SPIDLEN has been reached.	RW	0x0
7	TA	Transfer Active 0 = Transfer not active./CS lines are all high (assuming CSPOL = 0). RXR and DONE are 0. Writes to SPIFIFO write data into bits -0 of SPICS allowing DMA data blocks to set mode before sending data. 1 = Transfer active. /CS lines are set according to CS bits and CSPOL. Writes to SPIFIFO write data to TX FIFO.TA is cleared by a dma_frame_end pulse from the DMA controller.	RW	0x0
6	CSPOL	Chip Select Polarity 0 = Chip select lines are active low 1 = Chip select lines are active high	RW	0x0
5:4	CLEAR	CLEAR FIFO Clear 00 = No action. x1 = Clear TX FIFO. One shot operation. 1x = Clear RX FIFO. One shot operation. If CLEAR and TA are both set in the same operation, the FIFOs are cleared before the new frame is started. Read back as 0.	RW	0x0
3	CPOL	Clock Polarity 0 = Rest state of clock = low. 1 = Rest state of clock = high.	RW	0x0
2	CPHA	Clock Phase 0 = First SCLK transition at middle of data bit. 1 = First SCLK transition at beginning of data bit.	RW	0x0
1:0	CS	Chip Select 00 = Chip select 0 01 = Chip select 1 10 = Chip select 2 11 = Reserved	RW	0x0

FIFO Register

Synopsis This register allows TX data to be written to the TX FIFO and RX data to be read from the RX FIFO.

Bit(s)	Field Name	Description	Type	Reset
31:0	DATA	DMA Mode (DMAEN set) If TA is clear, the first 32-bit write to this register will control SPIDLEN and SPICS. Subsequent reads and writes will be taken as four-byte data words to be read/written to the FIFOs Poll/Interrupt Mode (DMAEN clear, TA set) Writes to the register write bytes to TX FIFO. Reads from register read bytes from the RX FIFO	RW	0x0

CLK Register

Synopsis This register allows the SPI clock rate to be set.

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved*** *- Write as 0, read as don't care*		
15:0	CDIV	Clock Divider SCLK = Core Clock / CDIV If CDIV is set to 0, the divisor is 65536. The divisor must be a power of 2. Odd numbers rounded down. The maximum SPI clock rate is of the APB clock.	RW	0x0

DLEN Register

Synopsis This register allows the SPI data length rate to be set.

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved*** *- Write as 0, read as don't care*		
15:0	LEN	Data Length The number of bytes to transfer. This field is only valid for DMA mode (DMAEN set) and controls how many bytes to transmit (and therefore receive).	RW	0x0

LTOH Register

Synopsis This register allows the LoSSI output hold delay to be set.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved** - Write as 0, read as don't care*		
3:0	TOH	This sets the Output Hold delay in APB clocks. A value of 0 causes a 1 clock delay.	RW	0x1

DC Register

Synopsis This register controls the generation of the DREQ and Panic signals to an external DMA engine The DREQ signals are generated when the FIFOs reach their defined levels and need servicing. The Panic signals instruct the external DMA engine to raise the priority of its AXI requests.

Bit(s)	Field Name	Description	Type	Reset
31:24	RPANIC	DMA Read Panic Threshold. Generate the Panic signal to the RX DMA engine whenever the RX FIFO level is greater than this amount.	RW	0x30
23:16	RDREQ	DMA Read Request Threshold. Generate A DREQ to the RX DMA engine whenever the RX FIFO level is greater than this amount, (RX DREQ is also generated if the transfer has finished but the RXFIFO isn t empty).	RW	0x20
15:8	TPANIC	DMA Write Panic Threshold. Generate the Panic signal to the TX DMA engine whenever the TX FIFO level is less than or equal to this amount.	RW	0x10
7:0	TDREQ	DMA Write Request Threshold. Generate a DREQ signal to the TX DMA engine whenever the TX FIFO level is less than or equal to this amount.	RW	0x20

10.6 Software Operation

10.6.1 Polled

a) Set CS, CPOL, CPHA as required and set TA = 1.

b) Poll TXD writing bytes to SPI_FIFO, RXD reading bytes from SPI_FIFO until all data written.

c) Poll DONE until it goes to 1.

d) Set TA = 0.

10.6.2 Interrupt

e) Set INTR and INTD. These can be left set over multiple operations.

f) Set CS, CPOL, CPHA as required and set TA = 1. This will immediately trigger a first interrupt with DONE == 1.

g) On interrupt:

h) If DONE is set and data to write (this means it is the first interrupt), write up to 16 bytes to SPI_FIFO. If DONE is set and no more data, set TA = 0. Read trailing data from SPI_FIFO until RXD is 0.

i) If RXR is set read 12 bytes data from SPI_FIFO and if more data to write, write up to 12 bytes to SPIFIFO.

10.6.3 DMA

Note: In order to function correctly, each DMA channel must be set to perform 32-bit transfers when communicating with the SPI. Either the Source or the Destination Transfer Width field in the DMA TI register must be set to 0 (i.e. 32-bit words) depending upon whether the channel is reading or writing to the SPI.

Two DMA channels are required, one to read from and one to write to the SPI.

j) Enable DMA DREQ's by setting the DMAEN bit and ADCS if required.

k) Program two DMA control blocks, one for each DMA controller.

l) DMA channel 1 control block should have its PER_MAP set to x and should be set to write 'transfer length' + 1 words to SPI_FIFO. The data should comprise:

  i) A word with the transfer length in bytes in the top sixteen bits, and the control register settings [7:0] in the bottom eight bits (i.e. TA = 1, CS, CPOL, CPHA as required.)

  ii) 'Transfer length' number in words of data to send.

m) DMA channel 2 control block should have its PER_MAP set to y and should be set to read 'transfer length' words from SPI_FIFO.

n) Point each DMA channel at its CB and set its ACTIVE bit to 1.

o) On receipt of an interrupt from DMA channel 2, the transfer is complete.

10.6.4 Notes

1. The SPI Master knows nothing of the peripherals it is connected to. It always both sends and receives bytes for every byte of the transaction.
2. SCLK is only generated during byte serial transfer. It pauses in the rest state if the next byte to send is not ready or RXF is set.
3. Setup and Hold times related to the automatic assertion and de-assertion of the CS lines when operating in DMA mode (DMAEN and ADCS set) are as follows:

   The CS line will be asserted at least 3 core clock cycles before the msb of the first byte of the transfer.

   The CS line will be de-asserted no earlier than 1 core clock cycle after the trailing edge of the final clock pulse.

   If these parameters are insufficient, software control should alleviate the problem. ADCS should be 0 allowing software to manually control the assertion and de-assertion of the CS lines.

11 SPI/BSC SLAVE

11.1 Introduction

The BSC interface can be used as either a Broadcom Serial Controller (BSC) or a Serial Peripheral Interface (SPI) controller. The BSC bus is a proprietary bus compliant with the Philips® I2C bus/interface version 2.1 January 2000. Both BSC and SPI controllers work in the slave mode. The BSC slave controller has specially built in the Host Control and Software Registers for a Chip booting. The BCS controller supports fast-mode (400Kb/s) and it is compliant to the I^{2}C bus specification version 2.1 January 2000 with the restrictions:

- I^{2}C slave only operation
- clock stretching is not supported
- 7-bit addressing only
-

There is only one BSC/SPI slave. The registers base addresses is 0x7E21_4000.

11.2 Registers

The SPI controller implements 3 wire serial protocol variously called Serial Peripheral Interface (SPI) or Synchronous Serial Protocol (SSP). BSC and SPI controllers do not have DMA connected, hence DMA is not supported.

I2C_SPI_SLV Address Map			
Address Offset	**Register Name**	**Description**	**Size**
0x0	DR	Data Register	32
0x4	RSR	The operation status register and error clear register	32
0x8	SLV	The I2C SPI Address Register holds the I2C slave address value	32
0xc	CR	The Control register is used to configure the I2C or SPI operation	32
0x10	FR	Flag register	32
0x14	IFLS	Interrupt fifo level select register	32
0x18	IMSC	Interupt Mask Set Clear Register	32

0x1c	RIS	Raw Interupt Status Register	32
0x20	MIS	Masked Interupt Status Register	32
0x24	ICR	Interupt Clear Register	32
0x28	DMACR	DMA Control Register	32
0x2c	TDR	FIFO Test Data	32
0x30	GPUSTAT	GPU Status Register	32
0x34	HCTRL	Host Control Register	32
0x38	DEBUG1	I2C Debug Register	32
0x3c	DEBUG2	SPI Debug Register	32

DR Register

Synopsis The I2C SPI Data Register is used to transfer/receive data characters and provide a Status and Flag information. Status and Flag information is also available via individual registers.

Bit(s)	Field Name	Description	Type	Reset
31:27	RXFLEVEL	RXFLEVEL RX FIFO Level Returns the current level of the RX FIFO use	RO	0x0
26:22	TXFLEVEL	TXFLEVEL TX FIFO Level Returns the current level of the TX FIFO use	RO	0x0
21	RXBUSY	RXBUSY Receive Busy 0 Receive operation inactive 1 Receive operation in operation	RO	0x0
20	TXFE	TXFE TX FIFO Empty 0 TX FIFO is not empty 1 When TX FIFO is empty	RO	0x1
19	RXFF	RXFE RX FIFO Full 0 FX FIFO is not full 1 When FX FIFO is full	RO	0x0

18	TXFF	TXFF TX FIFO Full 0 TX FIFO is not full 1 When TX FIFO is full	RO	0x0
17	RXFE	RXFE RX FIFO Empty 0 FX FIFO is not empty 1 When FX FIFO is empty	RO	0x1
16	TXBUSY	TXBUSY Transmit Busy 0 Transmit operation inactive 1 Transmit operation in operation	RO	0x0
15:10		***Reserved*** *- Write as 0, read as don't care*		
9	UE	TXUE TX Underrun Error 0 - No error case detected 1 Set when TX FIFO is empty and I2C master attempt to read a data character from I2C slave. Cleared by writing 0 to I2C SPI Status register .	RO	0x0
8	OE	RXOE RX Overrun Error 0 No error case detected 1 Set when RX FIFO is full and a new data character is received. Cleared by writing 0 to I2C SPI Status register .	RO	0x0
7:0	DATA	DATA Received/Transferred data characters Data written to this location is pushed into the TX FIFO. Data read from this location is fetched from the RX FIFO.	RW	0x0

RSR Register

Synopsis The operation status register and error clear register.

Bit(s)	Field Name	Description	Type	Reset
31:6		***Reserved*** *- Write as 0, read as don't care*		
5	RXDMABREQ	Unsupported, write zero, read as don't care	RO	0x0
4	RXDMAPREQ	Unsupported, write zero, read as don't care	RO	0x0
3	TXDMABREQ	Unsupported, write zero, read as don't care	RO	0x0

2	TXDMAPREQ	Unsupported, write zero, read as don't care	RO	0x0
1	UE	TXUE TX Underrun Error 0 - No error case detected 1 Set when TX FIFO is empty and I2C master attempt to read a data character from I2C slave. Cleared by writing 0 to it.	RW	0x0
0	OE	RXOE RX Overrun Error 0 No error case detected 1 Set when RX FIFO is full and a new data character is received. Cleared by writing 0 to it.	RW	0x0

SLV Register

Synopsis The I2C SPI Address Register holds the I2C slave address value. NOTE: It is of no use in SPI mode.

Bit(s)	Field Name	Description	Type	Reset
31:7		***Reserved*** *- Write as 0, read as don't care*		
6:0	ADDR	SLVADDR I2C Slave Address Programmable I2C slave address Note: In case HOSTCTRLEN bit is set from the I2C SPI Control Register bit SLVADDR[0] chooses the following: 0 - selects normal operation, i.e. accessing RX and TX FIFOs. 1 - selects access to I2C SPI SW Status Register or I2C SPI Host Control Register	RW	0x0

CR Register

Synopsis The Control register is used to configure the I2C or SPI operation.

Bit(s)	Field Name	Description	Type	Reset
31:14		***Reserved*** *- Write as 0, read as don't care*		

13	INV_TXF	INV-RX Inverse TX status flags 0 = default status flags When this bit is 0, bit 6 (TXFE - TX FIFO Empty) will reset to a 1 1 = inverted status flags When this bit is set, bit 6 (TXFE - TX FIFO Full) will reset to a 0 * Note: INV_TX bit changes the default values of 6 bit as it is specified for I2C SPI GPU Host Status Register .	RW	0x0
12	HOSTCTRLEN	HOSTCTRLEN Enable Control for Host 0 = Host Control disabled 1 = Host Control enabled Note: HOSTCTRLEN allows Host to request GPUSTAT or HCTRL register. The same behaviour is achieved from the GPU side using ENSTAT and ENCTRL.	RW	0x0
11	TESTFIFO	TESTFIFO TEST FIFO 0 = TESTT FIFO disabled 1 = TESTT FIFO enabled	RW	0x0
10	INV_RXF	INV-RX Inverse RX status flags 0 = default status flags When this bit is 0, bit 6 (RXFF - RX FIFO Full) will reset to a 0 1 = inverted status flags When this bit is 0, bit 6 (RXFF - RX FIFO Empty) will reset to a 1 * NOTE: INV_RX bit changes the default values of 7 bit as it is specified for I2C SPI GPU Host Status Register .	RW	0x0
9	RXE	RXE Receive Enable 0 = Receive mode disabled 1 = Receive mode enabled	RW	0x0
8	TXE	TXE Transmit Enable 0 = Transmit mode disabled 1 = Transmit mode enabled	RW	0x0
7	BRK	BRK Break current operation 0 = No effect. 1 = Stop operation and clear the FIFOs.	RW	0x0

6	ENCTRL	ENCTRL ENABLE CONTROL 8bit register 0 = Control register disabled. Implies ordinary I2C protocol. 1 = Control register enabled. When enabled the control register is received as a first data character on the I2C bus. NOTE: The same behaviour is achieved from the Host side by using bit SLVADDR[6] of the slave address.	RO	0x0
5	ENSTAT	ENSTAT ENABLE STATUS 8bit register 0 = Status register disabled. Implies ordinary I2C protocol. 1 = Status register enabled. When enabled the status register is transferred as a first data character on the I2C bus. Status register is transferred to the host. NOTE: The same behaviour is achieved from the Host side by using bit SLVADDR[6] of the slave address.	RW	0x0
4	CPOL	CPOL Clock Polarity 0 = 1 = SPI Related	RW	0x0
3	CPHA	CPHA Clock Phase 0 = 1 = SPI Related	RW	0x0
2	I2C	SPI Mode 0 = Disabled I2C mode 1 = Enabled I2C mode	RW	0x0
1	SPI	SPI Mode 0 = Disabled SPI mode 1 = Enabled SPI mode	RW	0x0
0	EN	EN Enable Device 1 = Enable I2C SPI Slave. 0 = Disable I2C SPI Slave.	RW	0x0

FR Register

Synopsis The flag register indicates the current status of the operation.

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved*** *- Write as 0, read as don't care*		

15:11	RXFLEVEL	RXFLEVEL RX FIFO Level Returns the current level of the RX FIFO use	RW	0x0
10:6	TXFLEVEL	TXFLEVEL TX FIFO Level Returns the current level of the TX FIFO use	RW	0x0
5	RXBUSY	RXBUSY Receive Busy 0 Receive operation inactive 1 Receive operation in operation	RW	0x0
4	TXFE	TXFE TX FIFO Empty 0 TX FIFO is not empty 1 When TX FIFO is empty	RW	0x1
3	RXFF	RXFE RX FIFO Full 0 FX FIFO is not full 1 When FX FIFO is full	RW	0x0
2	TXFF	TXFF TX FIFO Full 0 TX FIFO is not full 1 When TX FIFO is full	RW	0x0
1	RXFE	RXFE RX FIFO Empty 0 FX FIFO is not empty 1 When FX FIFO is empty	RW	0x1
0	TXBUSY	TXBUSY Transmit Busy 0 Transmit operation inactive 1 Transmit operation in operation	RW	0x0

IFLS Register

Synopsis The flag register indicates the current status of the operation.

Bit(s)	Field Name	Description	Type	Reset
31:12		***Reserved** - Write as 0, read as don't care*		
11:9	RXIFPSEL	Unsupported, write zero, read as don't care	RO	0x0
8:6	TXIFPSEL	Unsupported, write zero, read as don't care	RO	0x0

5:3	RXIFLSEL	RXIFLSEL RX Interrupt FIFO Level Select Interrupt is triggered when : 000 RX FIFO gets 1/8 full 001 RX FIFO gets 1/4 full 010 RX FIFO gets 1/2 full 011 RX FIFO gets 3/4 full 100 RX FIFO gets 7/8 full 101 111 not used	RW	0x0
2:0	TXIFLSEL	TXIFLSEL TX Interrupt FIFO Level Select Interrupt is triggered when : 000 TX FIFO gets 1/8 full 001 TX FIFO gets 1/4 full 010 TX FIFO gets 1/2 full 011 TX FIFO gets 3/4 full 100 TX FIFO gets 7/8 full 101 111 not used	RW	0x0

IMSC Register

Synopsis Interrupt Mask Set/Clear Register. On a read this register returns the current value of the mask on the relevant interrupt. On a write of 1 to the particular bit, it sets the corresponding mask of that interrupt. A write of 0 clears the corresponding mask.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved*** *- Write as 0, read as don't care*		
3	OEIM	Overrun error interrupt mask. A read returns the current mask for the interrupt. On a write of 1, the mask of the OEINTR interrupt is set. A write of 0 clears the mask.	RW	0x0
2	BEIM	Break error interrupt mask. A read returns the current mask for the BEINTR interrupt. On a write of 1, the mask of the interrupt is set. A write of 0 clears the mask.	RW	0x0
1	TXIM	Transmit interrupt mask. A read returns the current mask for the TXINTR interrupt. On a write of 1, the mask of the interrupt is set. A write of 0 clears the mask.	RW	0x0
0	RXIM	Receive interrupt mask. A read returns the current mask for the RXINTR interrupt. On a write of 1, the mask of the interrupt is set. A write of 0 clears the mask.	RW	0x0

RIS Register

Synopsis The Raw Interrupt Status Register returns the current raw status value, prior to masking, of the corresponding interrupt.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved** - Write as 0, read as don't care*		
3	OERIS	Overrun error interrupt status. Returns the raw interrupt state of the OEINTR interrupt.	RW	0x0
2	BERIS	Break error interrupt status. Returns the raw interrupt state of the BEINTR interrupt.	RW	0x0
1	TXRIS	Transmit interrupt status. Returns the raw interrupt state of the TXINTR interrupt.	RW	0x0
0	RXRIS	Receive interrupt status. Returns the raw interrupt state of the RXINTR interrupt.	RW	0x0

MIS Register

Synopsis The Masked Interrupt Status Register returns the current masked status value of the corresponding interrupt.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved** - Write as 0, read as don't care*		
3	OEMIS	Overrun error masked interrupt status. Returns the masked interrupt state of the OEINTR interrupt.	RW	0x0
2	BEMIS	Break error masked interrupt status. Returns the masked interrupt state of the BEINTR interrupt.	RW	0x0
1	TXMIS	Transmit masked interrupt status. Returns the masked interrupt state of the TXINTR interrupt.	RW	0x0
0	RXMIS	Receive masked interrupt status. Returns the masked interrupt state of the RXINTR interrupt.	RW	0x0

ICR Register

Synopsis The Interrupt Clear Register.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved*** *- Write as 0, read as don't care*		
3	OEIC	Overrun error interrupt clear. Clears the OEINTR interrupt.	RW	0x0
2	BEIC	Break error interrupt clear. Clears the BEINTR interrupt.	RW	0x0
1	TXIC	Transmit interrupt clear. Clears the TXINTR interrupt.	RW	0x0
0	RXIC	Receive masked interrupt status. Returns the masked interrupt state of the RXINTR interrupt.	RW	0x0

DMACR Register

Synopsis The DMA Control register is not supported in this version.

Bit(s)	Field Name	Description	Type	Reset
31:3		***Reserved*** *- Write as 0, read as don't care*		
2	DMAONERR	Unsupported, write zero, read as don't care	RW	0x0
1	TXDMAE	Unsupported, write zero, read as don't care	RW	0x0
0	RXDMAE	Unsupported, write zero, read as don't care	RW	0x0

TDR Register

Synopsis The Test Data Register enables data to be written into the receive FIFO and read out from the transmit FIFO for test purposes.

Bit(s)	Field Name	Description	Type	Reset
31:8		***Reserved*** *- Write as 0, read as don't care*		
7:0	DATA	Test data is written into the receive FIFO and read out of the transmit FIFO.	RW	0x0

GPUSTAT Register

Synopsis The GPU SW Status Register to be passed via I2C bus to a Host.
NOTE: GPU SW Status Register is combined with the status bit coming from within I2C SPI Slave device. Hence, the I2C SPI GPU Host Status Register as it is seen by a Host is depicted on Table 1 14.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved*** *- Write as 0, read as don't care*		
3:0	DATA	GPUSTAT GPU to Host Status Register SW controllable	RW	0x0

HCTRL Register

Synopsis The Host Control register is received from the host side via I2C bus. When ENCTRL - enable control register bit is set, the host control register is received as the first data character after the I2C address.

Bit(s)	Field Name	Description	Type	Reset
31:8		***Reserved*** *- Write as 0, read as don't care*		
7:0	DATA	HCTRL Host Control Register SW processing received via I2C bus	RW	0x0

DEBUG1 Register

Synopsis I2C Debug Register

Bit(s)	Field Name	Description	Type	Reset
31:26		***Reserved*** *- Write as 0, read as don't care*		
25:0	DATA		RW	0xe

DEBUG2 Register

Synopsis SPI Debug Register

Bit(s)	Field Name	Description	Type	Reset
31:24		***Reserved*** *- Write as 0, read as don't care*		
23:0	DATA		RW	0x400000

12 System Timer

The System Timer peripheral provides four 32-bit timer channels and a single 64-bit free running counter. Each channel has an output compare register, which is compared against the 32 least significant bits of the free running counter values. When the two values match, the system timer peripheral generates a signal to indicate a match for the appropriate channel. The match signal is then fed into the interrupt controller. The interrupt service routine then reads the output compare register and adds the appropriate offset for the next timer tick. The free running counter is driven by the timer clock and stopped whenever the processor is stopped in debug mode.

The Physical (hardware) base address for the system timers is 0x7E003000.

12.1 System Timer Registers

ST Address Map

Address Offset	Register Name	Description	Size
0x0	CS	System Timer Control/Status	32
0x4	CLO	System Timer Counter Lower 32 bits	32
0x8	CHI	System Timer Counter Higher 32 bits	32
0xc	C0	System Timer Compare 0	32
0x10	C1	System Timer Compare 1	32
0x14	C2	System Timer Compare 2	32
0x18	C3	System Timer Compare 3	32

CS Register

Synopsis System Timer Control / Status.

This register is used to record and clear timer channel comparator matches. The system timer match bits are routed to the interrupt controller where they can generate an interrupt.

The M0-3 fields contain the free-running counter match status. Write a one to the relevant bit to clear the match detect status bit and the corresponding interrupt request line.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved*** *- Write as 0, read as don't care*		
3	M3	System Timer Match 3 0 = No Timer 3 match since last cleared. 1 = Timer 3 match detected.	RW	0x0
2	M2	System Timer Match 2 0 = No Timer 2 match since last cleared. 1 = Timer 2 match detected.	RW	0x0
1	M1	System Timer Match 1 0 = No Timer 1 match since last cleared. 1 = Timer 1 match detected.	RW	0x0
0	M0	System Timer Match 0 0 = No Timer 0 match since last cleared. 1 = Timer 0 match detected.	RW	0x0

CLO Register

Synopsis System Timer Counter Lower bits.
The system timer free-running counter lower register is a read-only register that returns the current value of the lower 32-bits of the free running counter.

Bit(s)	Field Name	Description	Type	Reset
31:0	CNT	Lower 32-bits of the free running counter value.	RW	0x0

CHI Register

Synopsis System Timer Counter Higher bits.
The system timer free-running counter higher register is a read-only register that returns the current value of the higher 32-bits of the free running counter.

Bit(s)	Field Name	Description	Type	Reset
31:0	CNT	Higher 32-bits of the free running counter value.	RW	0x0

C0 C1 C2 C3 Register

Synopsis System Timer Compare.
The system timer compare registers hold the compare value for each of the four timer channels. Whenever the lower 32-bits of the free-running counter matches one of the compare values the corresponding bit in the system timer control/status register is set.
Each timer peripheral (minirun and run) has a set of four compare registers.

Bit(s)	Field Name	Description	Type	Reset
31:0	CMP	Compare value for match channel n.	RW	0x0

13 UART

The BCM2835 device has two UARTS. On mini UART and and PL011 UART. This section describes the PL011 UART. For details of the mini UART see *2.2 Mini UART*.

The PL011 UART is a Universal Asynchronous Receiver/Transmitter. This is the ARM UART (PL011) implementation. The UART performs serial-to-parallel conversion on data characters received from an external peripheral device or modem, and parallel-to-serial conversion on data characters received from the Advanced Peripheral Bus (APB).

The ARM PL011 UART has some optional functionality which can be included or left out.

The following functionality is ***not supported*** :

- Infrared Data Association (IrDA)
- Serial InfraRed (SIR) protocol Encoder/Decoder (ENDEC)
- Direct Memory Access (DMA).

The UART provides:

- Separate 16x8 transmit and 16x12 receive FIFO memory.
- Programmable baud rate generator.
- Standard asynchronous communication bits (start, stop and parity). These are added prior to transmission and removed on reception.
- False start bit detection.
- Line break generation and detection.
- Support of the modem control functions CTS and RTS. However DCD, DSR, DTR, and RI are not supported.
- Programmable hardware flow control.
- Fully-programmable serial interface characteristics:

  data can be 5, 6, 7, or 8 bits

  even, odd, stick, or no-parity bit generation and detection

  1 or 2 stop bit generation

  baud rate generation, dc up to UARTCLK/16

The UART clock source and associated dividers are controlled by the Clock Manager.

For the in-depth UART overview, please, refer to the ARM PrimeCell UART (PL011) Revision: r1p5 Technical Reference Manual.

13.1 Variations from the 16C650 UART

The UART varies from the industry-standard 16C650 UART device as follows:

- Receive FIFO trigger levels are 1/8, 1/4, 1/2, 3/4, and 7/8
- Transmit FIFO trigger levels are 1/8, 1/4, 1/2, 3/4, and 7/8
- The internal register map address space, and the bit function of each register differ

- The deltas of the modem status signals are not available.

The following 16C650 UART features are not supported:

- 1.5 stop bits (1 or 2 stop bits only are supported)
- Independent receive clock.

13.2 Primary UART Inputs and Outputs

The UART has two primary inputs RXD, nCTS and two primary outputs TXD, nRTS. The remaining signals like SRIN, SROUT, OUT1, OUT2, DSR, DTR, and RI are not supported in this implementation. The following table shows the UART signals map on the General Purpose I/O (GPIO). For the insight on how to program alternate function refer to the GPIO paragraph.

	Pull	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5
GPIO14	Low	TXD0					
GPIO15	Low	RXD0					
GPIO16	Low				CTS0		
GPIO17	Low				RTS0		
GPIO30	Low				CTS0		
GPIO31	Low				RTS0		
GPIO32	Low				TXD0		
GPIO33	Low				RXD0		
GPIO36	High			TXD0			
GPIO37	Low			RXD0			
GPIO38	Low			RTS0			
GPIO39	Low			CTS0			

Table 13-1 UART Assignment on the GPIO Pin map

13.3 UART Interrupts

The UART has one intra-chip interrupt UARTINTR generated as the OR-ed function of the five individual interrupts.

- UARTINTR, this is an OR function of the five individual masked outputs:
  - UARTRXINTR
  - UARTTXINTR
  - UARTRTINTR
  - UARTMSINTR, that can be caused by:
    - — UARTCTSINTR, because of a change in the nUARTCTS modem status
    - — UARTDSRINTR, because of a change in the nUARTDSR modem status.
  - UARTEINTR, that can be caused by an error in the reception:

— UARTOEINTR, because of an overrun error

— UARTBEINTR, because of a break in the reception

— UARTPEINTR, because of a parity error in the received character

— UARTFEINTR, because of a framing error in the received character.

One can enable or disable the individual interrupts by changing the mask bits in the Interrupt Mask Set/Clear Register, UART_IMSC. Setting the appropriate mask bit HIGH enables the interrupt.

UARTRXINTR:

The transmit interrupt changes state when one of the following events occurs:

- If the FIFOs are enabled and the transmit FIFO is equal to or lower than the programmed trigger level then the transmit interrupt is asserted HIGH. The transmit interrupt is cleared by writing data to the transmit FIFO until it becomes greater than the trigger level, or by clearing the interrupt.

- If the FIFOs are disabled (have a depth of one location) and there is no data present in the transmitters single location, the transmit interrupt is asserted HIGH. It is cleared by performing a single write to the transmit FIFO, or by clearing the interrupt.

UARTRTINTR:

The receive interrupt changes state when one of the following events occurs:

- If the FIFOs are enabled and the receive FIFO reaches the programmed trigger level. When this happens, the receive interrupt is asserted HIGH. The receive interrupt is cleared by reading data from the receive FIFO until it becomes less than the trigger level, or by clearing the interrupt.

- If the FIFOs are disabled (have a depth of one location) and data is received thereby filling the location, the receive interrupt is asserted HIGH. The receive interrupt is cleared by performing a single read of the receive FIFO, or by clearing the interrupt.

13.4 Register View

The PL011 USRT is mapped on base adderss 0x7E20100. It has the following memory-mapped registers.

UART Address Map			
Address Offset	**Register Name**	**Description**	**Size**
0x0	DR	Data Register	32
0x4	RSRECR		32
0x18	FR	Flag register	32

0x20	ILPR	not in use	32
0x24	IBRD	Integer Baud rate divisor	32
0x28	FBRD	Fractional Baud rate divisor	32
0x2c	LCRH	Line Control register	32
0x30	CR	Control register	32
0x34	IFLS	Interupt FIFO Level Select Register	32
0x38	IMSC	Interupt Mask Set Clear Register	32
0x3c	RIS	Raw Interupt Status Register	32
0x40	MIS	Masked Interupt Status Register	32
0x44	ICR	Interupt Clear Register	32
0x48	DMACR	DMA Control Register	32
0x80	ITCR	Test Control register	32
0x84	ITIP	Integration test input reg	32
0x88	ITOP	Integration test output reg	32
0x8c	TDR	Test Data reg	32

DR Register

Synopsis The UART_DR Register is the data register. For words to be transmitted:
if the FIFOs are enabled, data written to this location is pushed onto the transmit FIFO.
if the FIFOs are not enabled, data is stored in the transmitter holding register (the bottom word of the transmit FIFO).
The write operation initiates transmission from the UART. The data is prefixed with a start bit, appended with the appropriate parity bit (if parity is enabled), and a stop bit. The resultant word is then transmitted.
For received words:
if the FIFOs are enabled, the data byte and the 4-bit status (break, frame, parity, and overrun) is pushed onto the 12-bit wide receive FIFO
if the FIFOs are not enabled, the data byte and status are stored in the receiving holding register (the bottom word of the receive FIFO).

Bit(s)	Field Name	Description	Type	Reset
31:12		***Reserved** - Write as 0, read as don't care*		
11	OE	Overrun error. This bit is set to 1 if data is received and the receive FIFO is already full. This is cleared to 0 once there is an empty space in the FIFO and a new character can be written to it.	RW	0x0
10	BE	Break error. This bit is set to 1 if a break condition was detected, indicating that the received data input was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits). In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (marking state), and the next valid start bit is received.	RW	0x0
9	PE	Parity error. When set to 1, it indicates that the parity of the received data character does not match the parity that the EPS and SPS bits in the Line Control Register, UART_LCRH select. In FIFO mode, this error is associated with the character at the top of the FIFO.	RW	0x0

8	FE	Framing error. When set to 1, it indicates that the received character did not have a valid stop bit (a valid stop bit is 1). In FIFO mode, this error is associated with the character at the top of the FIFO.	RW	0x0
7:0	DATA	Receive (read) data character. Transmit (write) data character.	RW	0x0

RSRECR Register

Synopsis The UART_RSRECR Register is the receive status register/error clear register. If the status is read from this register, then the status information for break, framing and parity corresponds to the data character read from the Data Register, UART_DR. The status information for overrun is set immediately when an overrun condition occurs. NOTE: The received data character must be read first from the Data Register, UART_DR on before reading the error status associated with that data character from this register.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved*** *- Write as 0, read as don't care*		
3	OE	Overrun error. This bit is set to 1 if data is received and the receive FIFO is already full. This is cleared to 0 once there is an empty space in the FIFO and a new character can be written to it.	RW	0x0

2	BE	Break error. This bit is set to 1 if a break condition was detected, indicating that the received data input was held LOW for longer than a full-word transmission time (defined as start, data, parity and stop bits). In FIFO mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (marking state), and the next valid start bit is received.	RW	0x0
1	PE	Parity error. When set to 1, it indicates that the parity of the received data character does not match the parity that the EPS and SPS bits in the Line Control Register, UART_LCRH select. In FIFO mode, this error is associated with the character at the top of the FIFO.	RW	0x0
0	FE	Framing error. When set to 1, it indicates that the received character did not have a valid stop bit (a valid stop bit is 1). In FIFO mode, this error is associated with the character at the top of the FIFO.	RW	0x0

FR Register

Synopsis The UART_FR Register is the flag register.

Bit(s)	**Field Name**	**Description**	**Type**	**Reset**
31:9		***Reserved** - Write as 0, read as don't care*		
8	RI	Unsupported, write zero, read as don't care	RW	0x0

7	TXFE	Transmit FIFO empty. The meaning of this bit depends on the state of the FEN bit in the Line Control Register, UARTLCR_ LCRH. If the FIFO is disabled, this bit is set when the transmit holding register is empty. If the FIFO is enabled, the TXFE bit is set when the transmit FIFO is empty. This bit does not indicate if there is data in the transmit shift register.	RW	0x1
6	RXFF	Receive FIFO full. The meaning of this bit depends on the state of the FEN bit in the UARTLCR_ LCRH Register. If the FIFO is disabled, this bit is set when the receive holding register is full. If the FIFO is enabled, the RXFF bit is set when the receive FIFO is full.	RW	0x0
5	TXFF	Transmit FIFO full. The meaning of this bit depends on the state of the FEN bit in the UARTLCR_ LCRH Register. If the FIFO is disabled, this bit is set when the transmit holding register is full. If the FIFO is enabled, the TXFF bit is set when the transmit FIFO is full.	RW	0x0
4	RXFE	Receive FIFO empty. The meaning of this bit depends on the state of the FEN bit in the UARTLCR_H Register. If the FIFO is disabled, this bit is set when the receive holding register is empty. If the FIFO is enabled, the RXFE bit is set when the receive FIFO is empty.	RW	0x0
3	BUSY	UART busy. If this bit is set to 1, the UART is busy transmitting data. This bit remains set until the complete byte, including all the stop bits, has been sent from the shift register. This bit is set as soon as the transmit FIFO becomes non-empty, regardless of whether the UART is enabled or not.	RW	0x0
2	DCD	Unsupported, write zero, read as don't care	RW	0x0
1	DSR	Unsupported, write zero, read as don't care	RW	0x0

0	CTS	Clear to send. This bit is the complement of the UART clear to send, nUARTCTS, modem status input. That is, the bit is 1 when nUARTCTS is LOW.	RW	0x0

ILPR Register

Synopsis This is the disabled IrDA register, writing to it has not effect and reading returns 0.

Bit(s)	Field Name	Description	Type	Reset
31:0	ILPR	Reserved - write zero, read as don't care.	RW	0x0

IBRD Register

Synopsis The UART_IBRD Register is the integer part of the baud rate divisor value.

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved*** *- Write as 0, read as don't care*		
15:0	IBRD	The integer baud rate divisor.	RW	0x0

FBRD Register

Synopsis The UART_FBRD Register is the fractional part of the baud rate divisor value.
The baud rate divisor is calculated as follows:
Baud rate divisor BAUDDIV = (FUARTCLK/(16 Baud rate))
where FUARTCLK is the UART reference clock frequency. The BAUDDIV is comprised of the integer value IBRD and the fractional value FBRD. NOTE: The contents of the IBRD and FBRD registers are not updated until transmission or reception of the current character is complete.

Bit(s)	Field Name	Description	Type	Reset
31:6		***Reserved*** *- Write as 0, read as don't care*		
5:0	FBRD	The fractional baud rate divisor.	RW	0x0

LCRH Register

Synopsis The UARTLCR_ LCRH Register is the line control register.
NOTE: The UART_LCRH, UART_IBRD, and UART_FBRD registers must not be changed:
when the UART is enabled
when completing a transmission or a reception when it has been programmed to become disabled.

Bit(s)	Field Name	Description	Type	Reset
31:8		***Reserved*** *- Write as 0, read as don't care*		
7	SPS	Stick parity select. 0 = stick parity is disabled 1 = either: if the EPS bit is 0 then the parity bit is transmitted and checked as a 1 if the EPS bit is 1 then the parity bit is transmitted and checked as a 0. See Table 25 9.	RO	0x0

6:5	WLEN	Word length. These bits indicate the number of data bits transmitted or received in a frame as follows:
		b11 = 8 bits
		b10 = 7 bits
		b01 = 6 bits
		b00 = 5 bits.
			RW	0x0

4	FEN	Enable FIFOs: 0 = FIFOs are disabled (character mode) that is, the FIFOs become 1-byte-deep holding registers 1 = transmit and receive FIFO buffers are enabled (FIFO mode).	RW	0x0

3	STP2	Two stop bits select. If this bit is set to 1, two stop bits are transmitted at the end of the frame. The receive logic does not check for two stop bits being received.	RW	0x0
2	EPS	Even parity select. Controls the type of parity the UART uses during transmission and reception: 0 = odd parity. The UART generates or checks for an odd number of 1s in the data and parity bits. 1 = even parity. The UART generates or checks for an even number of 1s in the data and parity bits. This bit has no effect when the PEN bit disables parity checking and generation. See Table 25 9.	RW	0x0
1	PEN	Parity enable: 0 = parity is disabled and no parity bit added to the data frame 1 = parity checking and generation is enabled. See Table 25 9.	RW	0x0
0	BRK	Send break. If this bit is set to 1, a low-level is continually output on the TXD output, after completing transmission of the current character.	RW	0x0

CR Register

Synopsis The UART_CR Register is the control register.
NOTE:
To enable transmission, the TXE bit and UARTEN bit must be set to 1.
Similarly, to enable reception, the RXE bit and UARTEN bit, must be set to 1.
NOTE:
Program the control registers as follows:
1. Disable the UART.
2. Wait for the end of transmission or reception of the current character.
3. Flush the transmit FIFO by setting the FEN bit to 0 in the Line Control Register, UART_LCRH.
4. Reprogram the Control Register, UART_CR.
5. Enable the UART.

Bit(s)	Field Name	Description	Type	Reset
31:16		***Reserved*** *- Write as 0, read as don't care*		
15	CTSEN	CTS hardware flow control enable. If this bit is set to 1, CTS hardware flow control is enabled. Data is only transmitted when the nUARTCTS signal is asserted.	RW	0x0
14	RTSEN	RTS hardware flow control enable. If this bit is set to 1, RTS hardware flow control is enabled. Data is only requested when there is space in the receive FIFO for it to be received.	RW	0x0
13	OUT2	Unsupported, write zero, read as don't care	RO	0x0
12	OUT1	Unsupported, write zero, read as don't care	RO	0x0
11	RTS	Request to send. This bit is the complement of the UART request to send, nUARTRTS, modem status output. That is, when the bit is programmed to a 1 then nUARTRTS is LOW.	RW	0x0
10	DTR	Unsupported, write zero, read as don't care	RO	0x0
9	RXE	Receive enable. If this bit is set to 1, the receive section of the UART is enabled. Data reception occurs for UART signals. When the UART is disabled in the middle of reception, it completes the current character before stopping.	RW	0x1
8	TXE	Transmit enable. If this bit is set to 1, the transmit section of the UART is enabled. Data transmission occurs for UART signals. When the UART is disabled in the middle of transmission, it completes the current character before stopping.	RW	0x1
7	LBE	Loopback enable. If this bit is set to 1, the UARTTXD path is fed through to the UARTRXD path. In UART mode, when this bit is set, the modem outputs are also fed through to the modem inputs. This bit is cleared to 0 on reset, to disable loopback.	RW	0x0

6:3		***Reserved** - Write as 0, read as don't care*		
2	SIRLP	Unsupported, write zero, read as don't care	RO	0x0
1	SIREN	Unsupported, write zero, read as don't care	RO	0x0
0	UARTEN	UART enable: 0 = UART is disabled. If the UART is disabled in the middle of transmission or reception, it completes the current character before stopping. 1 = the UART is enabled.	RW	0x0

IFLS Register

Synopsis The UART_IFLS Register is the interrupt FIFO level select register. You can use this register to define the FIFO level that triggers the assertion of the combined interrupt signal.
The interrupts are generated based on a transition through a level rather than being based on the level. That is, the interrupts are generated when the fill level progresses through the trigger level.
The bits are reset so that the trigger level is when the FIFOs are at the half-way mark.

Bit(s)	Field Name	Description	Type	Reset
31:12		***Reserved** - Write as 0, read as don't care*		
11:9	RXIFPSEL	Unsupported, write zero, read as don't care	RO	0x0
8:6	TXIFPSEL	Unsupported, write zero, read as don't care	RO	0x0

5:3	RXIFLSEL	Receive interrupt FIFO level select. The trigger points for the receive interrupt are as follows:
		b000 = Receive FIFO becomes 1/8 full
		b001 = Receive FIFO becomes 1/4 full
		b010 = Receive FIFO becomes 1/2 full
		b011 = Receive FIFO becomes 3/4 full
		b100 = Receive FIFO becomes 7/8 full
		b101-b111 = reserved.
			RW	0x0

2:0	TXIFLSEL	Transmit interrupt FIFO level select. The trigger points for the transmit interrupt are as follows:
		b000 = Transmit FIFO becomes 1/8 full
		b001 = Transmit FIFO becomes 1/4 full
		b010 = Transmit FIFO becomes 1/2 full
		b011 = Transmit FIFO becomes 3/4 full
		b100 = Transmit FIFO becomes 7/8 full
		b101-b111 = reserved.
			RW	0x0

IMSC Register

Synopsis The UART_IMSC Register is the interrupt mask set/clear register. It is a read/write register. On a read this register returns the current value of the mask on the relevant interrupt. On a write of 1 to the particular bit, it sets the corresponding mask of that interrupt. A write of 0 clears the corresponding mask.

Bit(s)	Field Name	Description	Type	Reset
31:11		***Reserved** - Write as 0, read as don't care*		
10	OEIM	Overrun error interrupt mask. A read returns the current mask for the interrupt. On a write of 1, the mask of the UARTOEINTR interrupt is set. A write of 0 clears the mask.	RW	0x0
9	BEIM	Break error interrupt mask. A read returns the current mask for the UARTBEINTR interrupt. On a write of 1, the mask of the interrupt is set. A write of 0 clears the mask.	RW	0x0
8	PEIM	Parity error interrupt mask. A read returns the current mask for the UARTPEINTR interrupt. On a write of 1, the mask of the interrupt is set. A write of 0 clears the mask.	RW	0x0
7	FEIM	Framing error interrupt mask. A read returns the current mask for the UARTFEINTR interrupt. On a write of 1, the mask of the interrupt is set. A write of 0 clears the mask.	RW	0x0

6	RTIM	Receive timeout interrupt mask. A read returns the current mask for the UARTRTINTR interrupt. On a write of 1, the mask of the interrupt is set. A write of 0 clears the mask.	RW	0x0
5	TXIM	Transmit interrupt mask. A read returns the current mask for the UARTTXINTR interrupt. On a write of 1, the mask of the interrupt is set. A write of 0 clears the mask.	RW	0x0
4	RXIM	Receive interrupt mask. A read returns the current mask for the UARTRXINTR interrupt. On a write of 1, the mask of the interrupt is set. A write of 0 clears the mask.	RW	0x0
3	DSRMIM	Unsupported, write zero, read as don't care	RO	0x0
2	DCDMIM	Unsupported, write zero, read as don't care	RO	0x0
1	CTSMIM	nUARTCTS modem interrupt mask. A read returns the current mask for the UARTCTSINTR interrupt. On a write of 1, the mask of the interrupt is set. A write of 0 clears the mask.	RW	0x0
0	RIMIM	Unsupported, write zero, read as don't care	RO	0x0

RIS Register

Synopsis The UART_RIS Register is the raw interrupt status register. It is a read-only register. This register returns the current raw status value, prior to masking, of the corresponding interrupt.
NOTE: All the bits, except for the modem status interrupt bits (bits 3 to 0), are cleared to 0 when reset. The modem status interrupt bits are undefined after reset.

Bit(s)	Field Name	Description	Type	Reset
31:11		***Reserved*** *- Write as 0, read as don't care*		

10	OERIS	Overrun error interrupt status. Returns the raw interrupt state of the UARTOEINTR interrupt.	RW	0x0
9	BERIS	Break error interrupt status. Returns the raw interrupt state of the UARTBEINTR interrupt.	RW	0x0
8	PERIS	Parity error interrupt status. Returns the raw interrupt state of the UARTPEINTR interrupt.	RW	0x0
7	FERIS	Framing error interrupt status. Returns the raw interrupt state of the UARTFEINTR interrupt.	RW	0x0
6	RTRIS	Receive timeout interrupt status. Returns the raw interrupt state of the UARTRTINTR interrupt.	RW	0x0
5	TXRIS	Transmit interrupt status. Returns the raw interrupt state of the UARTTXINTR interrupt.	RW	0x0
4	RXRIS	Receive interrupt status. Returns the raw interrupt state of the UARTRXINTR interrupt.	RW	0x0
3	DSRRMIS	Unsupported, write zero, read as don't care	RW	0x0
2	DCDRMIS	Unsupported, write zero, read as don't care	RW	0x0
1	CTSRMIS	nUARTCTS modem interrupt status. Returns the raw interrupt state of the UARTCTSINTR interrupt.	RW	0x0
0	RIRMIS	Unsupported, write zero, read as don't care	RW	0x0

MIS Register

Synopsis The UART_MIS Register is the masked interrupt status register. This register returns the current masked status value of the corresponding interrupt. NOTE: All the bits, except for the modem status interrupt bits (bits 3 to 0), are cleared to 0 when reset. The modem status interrupt bits are undefined after reset.

Bit(s)	Field Name	Description	Type	Reset
31:11		***Reserved** - Write as 0, read as don't care*		
10	OEMIS	Overrun error masked interrupt status. Returns the masked interrupt state of the UARTOEINTR interrupt.	RW	0x0
9	BEMIS	Break error masked interrupt status. Returns the masked interrupt state of the UARTBEINTR interrupt.	RW	0x0
8	PEMIS	Parity error masked interrupt status. Returns the masked interrupt state of the UARTPEINTR interrupt.	RW	0x0
7	FEMIS	Framing error masked interrupt status. Returns the masked interrupt state of the UARTFEINTR interrupt.	RW	0x0
6	RTMIS	Receive timeout masked interrupt status. Returns the masked interrupt state of the UARTRTINTR interrupt.	RW	0x0
5	TXMIS	Transmit masked interrupt status. Returns the masked interrupt state of the UARTTXINTR interrupt.	RW	0x0
4	RXMIS	Receive masked interrupt status. Returns the masked interrupt state of the UARTRXINTR interrupt.	RW	0x0
3	DSRMMIS	Unsupported, write zero, read as don't care	RW	0x0
2	DCDMMIS	Unsupported, write zero, read as don't care	RW	0x0
1	CTSMMIS	nUARTCTS modem masked interrupt status. Returns the masked interrupt state of the UARTCTSINTR interrupt.	RW	0x0
0	RIMMIS	Unsupported, write zero, read as don't care	RW	0x0

ICR Register

Synopsis The UART_ICR Register is the interrupt clear register.

Bit(s)	Field Name	Description	Type	Reset
31:11		***Reserved*** *- Write as 0, read as don't care*		
10	OEIC	Overrun error interrupt clear. Clears the UARTOEINTR interrupt.	RW	0x0
9	BEIC	Break error interrupt clear. Clears the UARTBEINTR interrupt.	RW	0x0
8	PEIC	Parity error interrupt clear. Clears the UARTPEINTR interrupt.	RW	0x0
7	FEIC	Framing error interrupt clear. Clears the UARTFEINTR interrupt..	RW	0x0
6	RTIC	Receive timeout interrupt clear. Clears the UARTRTINTR interrupt.	RW	0x0
5	TXIC	Transmit interrupt clear. Clears the UARTTXINTR interrupt.	RW	0x0
4	RXIC	Receive masked interrupt status. Returns the masked interrupt state of the UARTRXINTR interrupt.	RW	0x0
3	DSRMIC	Unsupported, write zero, read as don't care	RW	0x0
2	DCDMIC	Unsupported, write zero, read as don't care	RW	0x0
1	CTSMIC	nUARTCTS modem masked interrupt status. Returns the masked interrupt state of the UARTCTSINTR interrupt.	RW	0x0
0	RIMIC	Unsupported, write zero, read as don't care	RW	0x0

DMACR Register

Synopsis This is the disabled DMA Control Register, writing to it has not effect and reading returns 0.

Bit(s)	Field Name	Description	Type	Reset
31:3		***Reserved** - Write as 0, read as don't care*		
2	DMAONERR	Unsupported, write zero, read as don't care	RW	0x0
1	TXDMAE	Unsupported, write zero, read as don't care	RW	0x0
0	RXDMAE	Unsupported, write zero, read as don't care	RW	0x0

ITCR Register

Synopsis This is the Test Control Register UART_ITCR.

Bit(s)	Field Name	Description	Type	Reset
31:2		***Reserved** - Write as 0, read as don't care*		
1	ITCR1	Test FIFO enable. When this bit it 1, a write to the Test Data Register, UART_DR writes data into the receive FIFO, and reads from the UART_DR register reads data out of the transmit FIFO. When this bit is 0, data cannot be read directly from the transmit FIFO or written directly to the receive FIFO (normal operation).	RW	0x0
0	ITCR0	Integration test enable. When this bit is 1, the UART is placed in integration test mode, otherwise it is in normal operation.	RW	0x0

ITIP Register

Synopsis This is the Test Control Register UART_ITIP.

Bit(s)	Field Name	Description	Type	Reset
31:4		***Reserved*** *- Write as 0, read as don't care*		
3	ITIP3	Reads return the value of the nUARTCTS primary input.	RW	0x0
2:1		***Reserved*** *- Write as 0, read as don't care*		
0	ITIP0	Reads return the value of the UARTRXD primary input.	RW	0x0

ITOP Register

Synopsis This is the Test Control Register UART_ITOP.

Bit(s)	Field Name	Description	Type	Reset
31:12		***Reserved*** *- Write as 0, read as don't care*		
11	ITOP11	Intra-chip output. Writes specify the value to be driven on UARTMSINTR. Reads return the value of UARTMSINTR at the output of the test multiplexor.	RW	0x0
10	ITOP10	Intra-chip output. Writes specify the value to be driven on UARTRXINTR. Reads return the value of UARTRXINTR at the output of the test multiplexor.	RW	0x0
9	ITOP9	Intra-chip output. Writes specify the value to be driven on UARTTXINTR. Reads return the value of UARTTXINTR at the output of the test multiplexor.	RW	0x0

8	ITOP8	Intra-chip output. Writes specify the value to be driven on UARTRTINTR. Reads return the value of UARTRTINTR at the output of the test multiplexor.	RW	0x0
7	ITOP7	Intra-chip output. Writes specify the value to be driven on UARTEINTR. Reads return the value of UARTEINTR at the output of the test multiplexor.	RW	0x0
6	ITIP6	Intra-chip output. Writes specify the value to be driven on UARTINTR. Reads return the value of UARTINTR at the output of the test multiplexor.	RW	0x0
5:4		***Reserved*** *- Write as 0, read as don't care*		
3	ITIP3	Primary output. Writes specify the value to be driven on nUARTRTS.	RW	0x0
2:1		***Reserved*** *- Write as 0, read as don't care*		
0	ITIP0	Primary output. Writes specify the value to be driven on UARTTXD.	RW	0x0

TDR Register

Synopsis UART_TDR is the test data register. It enables data to be written into the receive FIFO and read out from the transmit FIFO for test purposes. This test function is enabled by the ITCR1bit in the Test Control Register, UART_ITCR.

Bit(s)	**Field Name**	**Description**	**Type**	**Reset**
31:11		***Reserved*** *- Write as 0, read as don't care*		
10:0	TDR10_0	When the ITCR1 bit is set to 1, data is written into the receive FIFO and read out of the transmit FIFO.	RW	0x0

14 Timer (ARM side)

14.1 Introduction

The ARM Timer is *based* on a ARM AP804, but it has a number of differences with the standard SP804:

- There is only one timer.
- It only runs in continuous mode.
- It has a extra clock pre-divider register.
- It has a extra stop-in-debug-mode control bit.
- It also has a 32-bit free running counter.

The clock from the ARM timer is derived from the system clock. This clock can change dynamically e.g. if the system goes into reduced power or in low power mode. Thus the clock speed adapts to the overal system performance capabilities. For accurate timing it is recommended to use the system timers.

14.2 Timer Registers:

The base address for the ARM timer register is 0x7E00B000.

Address offset[8]	Description
0x400	Load
0x404	Value (Read Only)
0x408	Control
0x40C	IRQ Clear/Ack (Write only)
0x410	RAW IRQ (Read Only)
0x414	Masked IRQ (Read Only)
0x418	Reload
0x41C	*Pre-divider (Not in real 804!)*
0x420	*Free running counter (Not in real 804!)*

Timer Load register

The timer load register sets the time for the timer to count down. This value is loaded into the timer value register after the load register has been written or if the timer-value register has counted down to 0.

[8] This is the offset which needs to be added to the base address to get the full hardware address.

Timer Value register:

This register holds the current timer value and is counted down when the counter is running. It is counted down each timer clock until the value 0 is reached. Then the value register is re-loaded from the timer load register and the interrupt pending bit is set. The timer count down speed is set by the timer pre-divide register.

Timer control register:

The standard SP804 timer control register consist of 8 bits but in the BCM implementation there are more control bits for the extra features. Control bits 0-7 are identical to the SP804 bits, albeit some functionality of the SP804 is not implemented. All new control bits start from bit 8 upwards. Differences between a real 804 and the BCM implementation are shown in italics.

Name: Timer control		**Address: base + 0x40C**	**Reset: 0x3E0020**
Bit(s)	**R/W**	**Function**	
31:10	-	<Unused>	
23:16	*R/W*	*Free running counter pre-scaler. Freq is sys_clk/(prescale+1)* ***These bits do not exists in a standard 804! Reset value is 0x3E***	
15:10	-	<Unused>	
9	*R/W*	*0 : Free running counter Disabled* *1 : Free running counter Enabled* ***This bit does not exists in a standard 804 timer!***	
8	*R/W*	*0 : Timers keeps running if ARM is in debug halted mode* *1 : Timers halted if ARM is in debug halted mode* ***This bit does not exists in a standard 804 timer!***	
7	R/W	0 : Timer disabled 1 : Timer enabled	
6	*R/W*	***Not used**, The timer is always in free running mode.* *If this bit is set it enables periodic mode in a standard 804. That mode is not supported in the BC2835M.*	
5	R/W	0 : Timer interrupt disabled 1 : Timer interrupt enabled	
4	R/W	<Not used>	
3:2	R/W	Pre-scale bits: 00 : pre-scale is clock / 1 (No pre-scale) 01 : pre-scale is clock / 16 10 : pre-scale is clock / 256 11 : pre-scale is clock / 1 *(Undefined in 804)*	
1	R/W	0 : 16-bit counters 1 : 23-bit counter	
0	*R/W*	***Not used**, The timer is always in wrapping mode.* *If this bit is set it enables one-shot mode in real 804. That mode is not supported in the BCM2835.*	

Timer IRQ clear register:

The timer IRQ clear register is write only. When writing this register the interrupt-pending bit is cleared.

When reading this register it returns 0x544D5241 which is the ASCII reversed value for "ARMT".

Timer Raw IRQ register

The raw IRQ register is a read-only register. It shows the status of the interrupt pending bit.

Name: Raw IRQ		Address: base + 0x40C	Reset: 0x3E0020
Bit(s)	**R/W**	**Function**	
31:0	R	0	
0	R	0 : The interrupt pending bits is clear 1 : The interrupt pending bit is set.	

The interrupt pending bits is set each time the value register is counted down to zero. The interrupt pending bit can not by itself generates interrupts. Interrupts can only be generated if the interrupt enable bit is set.

Timer Masked IRQ register:

The masked IRQ register is a read-only register. It shows the status of the interrupt signal. It is simply a logical AND of the interrupt pending bit and the interrupt enable bit.

Name: Masked IRQ		Address: base + 0x40C	Reset: 0x3E0020
Bit(s)	**R/W**	**Function**	
31:0	R	0	
0	R	0 : Interrupt line not asserted. 1 :Interrupt line is asserted, (the interrupt pending and the interrupt enable bit are set.)	

Timer Reload register:

This register is a copy of the timer load register. The difference is that a write to this register does not trigger an immediate reload of the timer value register. Instead the timer load register value is only accessed if the value register has finished counting down to zero.

The timer pre-divider register:

Name: pre-divide		Address: base + 0x41C	Reset: 0x07D
Bit(s)	**R/W**	**Function**	
31:10	-	<Unused>	
9:0	R/W	Pre-divider value.	

The Pre-divider register is not present in the SP804.

The pre-divider register is 10 bits wide and can be written or read from. This register has been added as the SP804 expects a 1MHz clock which we do not have. Instead the pre-divider takes the APB clock and divides it down according to:

timer_clock = apb_clock/(pre_divider+1)

The reset value of this register is 0x7D so gives a divide by 126.

Free running counter

Name: Free running		Address: base + 0x420	Reset: 0x000
Bit(s)	**R/W**	**Function**	
31:0	R	Counter value	

The free running counter is not present in the SP804.

The free running counter is a 32 bits wide read only register. The register is enabled by setting bit 9 of the Timer control register. The free running counter is incremented immediately after it is enabled. The timer can not be reset but when enabled, will always increment and roll-over. The free running counter is also running from the APB clock and has its own clock pre-divider controlled by bits 16-23 of the timer control register.

This register will be halted too if bit 8 of the control register is set and the ARM is in Debug Halt mode.

15 USB

The USB core used in the Videocore is build from Synopsys IP. Details about the block can be found in DWC_otg_databook.pdf (Which can also be downloaded from https://www.synopsys.com/dw/ipdir.php?ds=dwc_usb_2_0_hs_otg) .

15.1 Configuration

A number of features of the block are specified before the block is build and thus can not be changed using software. The above mentioned document has a list of these under the chapter "Configuration Parameters". The following table list all configuration parameters mentioned in that chapter and the values which have been chosen.

Feature/Parameter	Selected value
Mode of Operation	0: HNP- and SRP-Capable OTG (Device and Host)
LPM Mode of Operation	0: Non-LPM-capable core
HSIC Mode of Operation	0: Non-HSIC-capable core
Architecture	2: Internal DMA
Point-to-Point Application Only	0: No
High-Speed PHY Interfaces	1: UTMI+
USB 1.1 Full-Speed Serial Transceiver Interface	1: Dedicated FS
USB IC_USB Transceiver Interface	0: Non-IC_USB-capable
Default (Power on) Interface selection: FS_USB/IC_USB	0 : FS_USB interface
Data Width of the UTMI+ Interface	0: 8 bits
Enable I2C Interface	0: None
Enable ULPI Carkit	0: No
Enable PHY Vendor Control Interface	0: No

Feature/Parameter	Selected value
Number of Device Mode Endpoints in Addition to Control Endpoint 0	7
Enable Dedicated Transmit FIFOs for Device IN Endpoints	1: Yes
Enable descriptor based scatter/gather DMA	0: No
Enable Option for Endpoint- Specific Interrupt	0: No
Number of Device Mode Periodic IN Endpoints	0
Number of Device Mode IN Endpoints including Control Endpoi 0	8
Number of Device Mode Control Endpoints in Addition to Endpoint 0	0
Number of Host Mode Channels	8
Is Periodic OUT Channel Support Needed in Host Mode	1: Yes
Total Data FIFO RAM Depth	4096
Enable Dynamic FIFO Sizing	1: Yes
Largest Rx Data FIFO Depth	4096
Largest Non-Periodic Host Tx Data FIFO Depth	1024
Largest Non-periodic Tx Data FIFO Depth	4096
Largest Host Mode Tx Periodic Data FIFO Dept	4096
Non-periodic Request Queue Depth	8
Host Mode Periodic Request Queue Depth	8
Device Mode IN Token Sequence Learning Queue Depth	8
Width of Transfer Size Counters	19

Feature/Parameter	Selected value
Width of Packet Counters	10
Remove Optional Features	0: No
Power-on Value of User ID Register	0x2708A000
Enable Power Optimization	0: No
Is Minimum AHB Operating Frequency Less than 60 MHz	1: Yes
Reset Style of Clocked always Blocks in RTL	0: Asynchronous
Instantiate Double- Synchronization Flops	1: Yes
Enable Filter on “iddig” Signal from PHY	1: Yes
Enable Filter on “vbus_valid” Signal from PHY	1: Yes
Enable Filter on “a_valid” Signal from PHY	1: Yes
Enable Filter on “b_valid” Signal from PHY	1: Yes
Enable Filter on “session_end” Signal from PHY	1: Yes
Direction of Endpoints	Mode is {IN and OUT} for all endpoints
Largest Device Mode Periodic Tx Data FIFO n Depth	768 for all endpoints (Except 0)
Largest Device Mode IN Endpoint Tx FIFOn Depth (n = 0 to 15) when using dynamic FIFO sizing	0=32 1..5=512 6,7=768

15.2 Extra / Adapted registers.

Besides the registers as specified in the documentation of Synopsys a number of extra registers have been added. These control the Analogue USB Phy and the connections of the USB block into the Video core bus structure. Also the USB_GAHBCFG register has an alternative function for the bits [4:1].

Base Address of the USB block – 0x7E98_0000

Offset Address	Description		Size	Read/ Write
0x080	USB_MDIO_CNTL	MDIO interface control		R/W
0x084	USB_MDIO_GEN	Data for MDIO interface	32	R/W
0x088	USB_VBUS_DRV	Vbus and other Miscellaneous controls		R/W

USB MDIO Control (USB_MDIO_CNTL)

Address 0x 7E98 0080

Bit Number	Field Name	Description	Read/ Write	Reset
31	mdio_busy	1= MDIO read or write in progress 0= MDIO Idle	R	0
30-24	-	Unused	-	0
23	bb_mdo	Direct write (bitbash) MDO output	R/W	0
22	bb_mdc	Direct write (bitbash) MDC output	R/W	0
21	bb_enbl	1= MDIO bitbash enable 0= MDIO under control of the phy	R/W	0
20	freerun	1= MDC is continous active 0 = MDC only active during data transfer	R/W	0
19:16	mdc_ratio	MDC clock freq is sysclk/mdc_ratio	R/W	0
15:0	mdi	16-bit read of MDIO input shift register. Updates on falling edge of MDC	RO	0

Table 15-1 MDIO Control

USB MDIO Data (USB_MDIO_DATA)

Address 0x 7E98 0084

Bit Number	Field Name	Description	Read/ Write	Reset
31-0	mdio_data	32-bit sequence to send over MDIO bus	W	0
31-0	mdio_data	32-bit sequence received from MDIO bus	R	0

Table 15-2 USB MDIO data

A Preamble is not auto-generated so any MDIO access must be preceded by a write to this register of 0xFFFFFFFF. Furthermore, a bug in the USB PHY requires an extra clock edge so a write of 0x00000000 must follow the actual access.

USB VBUS (USB_VBUS)

Address 0x 7E98 0088

Bit Number	Field Name	Description	Read/ Write	Reset
31-20	-	Unused	-	0
19-16	axi_priority	Sets the USB AXI priority level	R/W	0
15:10	-	Unused	-	0
9	vbus_irq	1=one or more bits of [6:4] have changed since last read. This bit is cleared when the register is read.	RC	0
8	vbus_irq_en	1=Enable IRQ on VBUS status change	R/W	0
7	afe_non_driving	1=USB PHY AFE pull ups/pull downs are off 0=Normal USB AFE operation (Has no effect if MDIO mode is enabled in the phy)	R/W	0
6	utmisrp_dischrgvbus	Drive VBUS	R	0
5	utmisrp_chrgvbus	Charge VBUS	R	0
4	utmiotg_drvvbus	Discharge VBUS	R	0
3	utmiotg_avalid	A session Valid	R/W	0
2	utmiotg_bvalid	B session Valid	R/W	0
1	utmiotg_vbusvalid	VBUS valid	R/W	0
0	utmisrp_sessend	Session end	R/W	0

Table 15-3 USB MDIO data

The RW bits in this register are fed into the USB2.0 controller and the RO bits are coming out of it. In the real device, it will be up to the software to communicate this information between the USB2.0 controller and external VBUS device (some of these have I2C control, others will have to interface via GPIO).

USB AHB configuration (USB_GAHBCFG)

Address 0x 7E98 0008

The USB_GAHBCFG register has been adapted. Bits [4:1] which are marked in the Synopsys documentation as "Burst Length/Type (HBstLen)" have been used differently.

[4] 1 = Wait for all outstanding AXI writes to complete before signalling (internally) that DMA is done.
0 = don't wait.

[3] Not used

[2:1] Sets the maximum AXI burst length, but the bits are inverted,
00 = maximum AXI burst length of 4,
01 = maximum AXI burst length of 3,
10 = maximum AXI burst length of 2
11 = maximum AXI burst length of 1

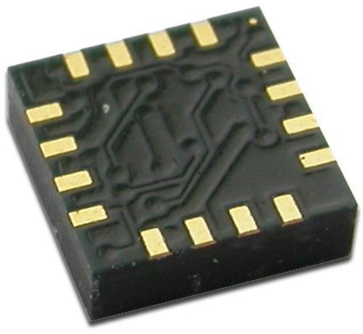
Personal Notes:

3-Axis Digital Compass IC HMC5883L

Honeywell

Advanced Information

The Honeywell HMC5883L is a surface-mount, multi-chip module designed for low-field magnetic sensing with a digital interface for applications such as low-cost compassing and magnetometry. The HMC5883L includes our state-of-the-art, high-resolution HMC118X series magneto-resistive sensors plus an ASIC containing amplification, automatic degaussing strap drivers, offset cancellation, and a 12-bit ADC that enables 1° to 2° compass heading accuracy. The I^2C serial bus allows for easy interface. The HMC5883L is a 3.0x3.0x0.9mm surface mount 16-pin leadless chip carrier (LCC). Applications for the HMC5883L include Mobile Phones, Netbooks, Consumer Electronics, Auto Navigation Systems, and Personal Navigation Devices.



The HMC5883L utilizes Honeywell's Anisotropic Magnetoresistive (AMR) technology that provides advantages over other magnetic sensor technologies. These anisotropic, directional sensors feature precision in-axis sensitivity and linearity. These sensors' solid-state construction with very low cross-axis sensitivity is designed to measure both the direction and the magnitude of Earth's magnetic fields, from milli-gauss to 8 gauss. Honeywell's Magnetic Sensors are among the most sensitive and reliable low-field sensors in the industry.

FEATURES	BENEFITS
▸ 3-Axis Magnetoresistive Sensors and ASIC in a 3.0x3.0x0.9mm LCC Surface Mount Package	▸ Small Size for Highly Integrated Products. Just Add a Micro-Controller Interface, Plus Two External SMT Capacitors Designed for High Volume, Cost Sensitive OEM Designs Easy to Assemble & Compatible with High Speed SMT Assembly
▸ 12-Bit ADC Coupled with Low Noise AMR Sensors Achieves 2 milli-gauss Field Resolution in ±8 Gauss Fields	▸ Enables 1° to 2° Degree Compass Heading Accuracy
▸ Built-In Self Test	▸ Enables Low-Cost Functionality Test after Assembly in Production
▸ Low Voltage Operations (2.16 to 3.6V) and Low Power Consumption (100 μA)	▸ Compatible for Battery Powered Applications
▸ Built-In Strap Drive Circuits	▸ Set/Reset and Offset Strap Drivers for Degaussing, Self Test, and Offset Compensation
▸ I^2C Digital Interface	▸ Popular Two-Wire Serial Data Interface for Consumer Electronics
▸ Lead Free Package Construction	▸ RoHS Compliance
▸ Wide Magnetic Field Range (+/-8 Oe)	▸ Sensors Can Be Used in Strong Magnetic Field Environments with a 1° to 2° Degree Compass Heading Accuracy
▸ Software and Algorithm Support Available	▸ Compassing Heading, Hard Iron, Soft Iron, and Auto Calibration Libraries Available
▸ Fast 160 Hz Maximum Output Rate	▸ Enables Pedestrian Navigation and LBS Applications

SPECIFICATIONS (* Tested at 25°C except stated otherwise.)

Characteristics	Conditions*	Min	Typ	Max	Units
Power Supply					
Supply Voltage	VDD Referenced to AGND	2.16	2.5	3.6	Volts
	VDDIO Referenced to DGND	1.71	1.8	VDD+0.1	Volts
Average Current Draw	Idle Mode	-	2	-	μA
	Measurement Mode (7.5 Hz ODR;	-	100	-	μA
	No measurement average, MA1:MA0 = 00)				
	VDD = 2.5V, VDDIO = 1.8V (Dual Supply)				
	VDD = VDDIO = 2.5V (Single Supply)				
Performance					
Field Range	Full scale (FS)	-8		+8	gauss
Mag Dynamic Range	3-bit gain control	±1		±8	gauss
Sensitivity (Gain)	VDD=3.0V, GN=0 to 7, 12-bit ADC	230		1370	LSb/gauss
Digital Resolution	VDD=3.0V, GN=0 to 7, 1-LSb, 12-bit ADC	0.73		4.35	milli-gauss
Noise Floor (Field Resolution)	VDD=3.0V, GN=0, No measurement average, Standard Deviation 100 samples (See typical performance graphs below)		2		milli-gauss
Linearity	±2.0 gauss input range			0.1	±% FS
Hysteresis	±2.0 gauss input range		±25		ppm
Cross-Axis Sensitivity	Test Conditions: Cross field = 0.5 gauss, Happlied = ±3 gauss		±0.2%		%FS/gauss
Output Rate (ODR)	Continuous Measurment Mode	0.75		75	Hz
	Single Measurement Mode			160	Hz
Measurement Period	From receiving command to data ready		6		ms
Turn-on Time	Ready for I2C commands		200		μs
	Analog Circuit Ready for Measurements		50		ms
Gain Tolerance	All gain/dynamic range settings		±5		%
I^2C Address	8-bit read address		0x3D		hex
	8-bit write address		0x3C		hex
I^2C Rate	Controlled by I^2C Master			400	kHz
I^2C Hysteresis	Hysteresis of Schmitt trigger inputs on SCL and SDA - Fall (VDDIO=1.8V)		0.2*VDDIO		Volts
	Rise (VDDIO=1.8V)		0.8*VDDIO		Volts
Self Test	X & Y Axes		±1.16		gauss
	Z Axis		±1.08		
	X & Y & Z Axes (GN=5) Positive Bias	243		575	LSb
	X & Y & Z Axes (GN=5) Negative Bias	-575		-243	
Sensitivity Tempco	T_A = -40 to 125°C, Uncompensated Output		-0.3		%/°C
General					
ESD Voltage	Human Body Model (all pins)			2000	Volts
	Charged Device Model (all pins)			750	
Operating Temperature	Ambient	-30		85	°C
Storage Temperature	Ambient, unbiased	-40		125	°C

Characteristics	Conditions*	Min	Typ	Max	Units
Reflow Classification	MSL 3, 260 °C Peak Temperature				
Package Size	Length and Width	2.85	3.00	3.15	mm
Package Height		0.8	0.9	1.0	mm
Package Weight			18		mg

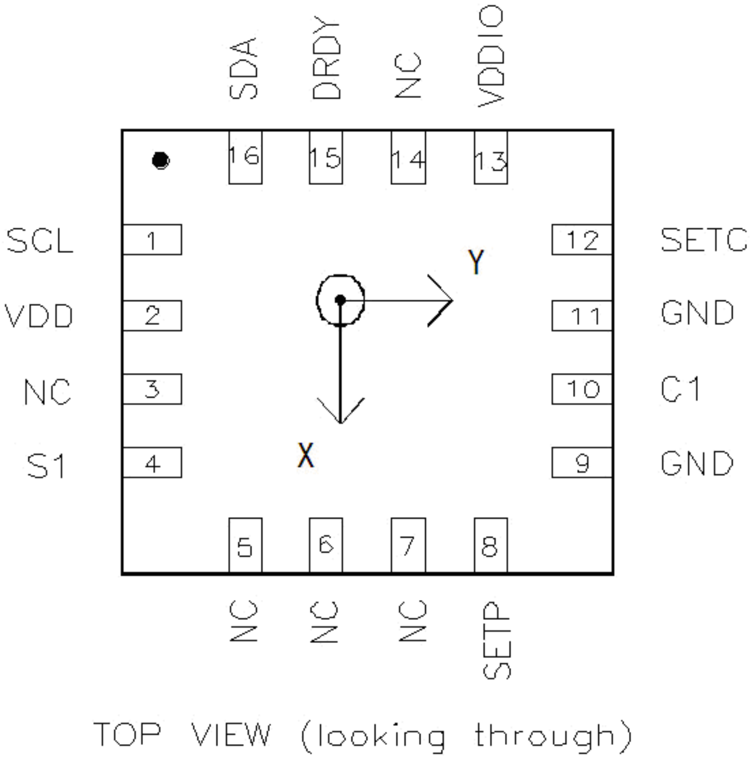
Absolute Maximum Ratings (* Tested at 25°C except stated otherwise.)

Characteristics	Min	Max	Units
Supply Voltage VDD	-0.3	4.8	Volts
Supply Voltage VDDIO	-0.3	4.8	Volts

PIN CONFIGURATIONS

Pin	Name	Description
1	SCL	Serial Clock – I^{2}C Master/Slave Clock
2	VDD	Power Supply (2.16V to 3.6V)
3	NC	Not to be Connected
4	S1	Tie to VDDIO
5	NC	Not to be Connected
6	NC	Not to be Connected
7	NC	Not to be Connected
8	SETP	Set/Reset Strap Positive – S/R Capacitor (C2) Connection
9	GND	Supply Ground
10	C1	Reservoir Capacitor (C1) Connection
11	GND	Supply Ground
12	SETC	S/R Capacitor (C2) Connection – Driver Side
13	VDDIO	IO Power Supply (1.71V to VDD)
14	NC	Not to be Connected
15	DRDY	Data Ready, Interrupt Pin. Internally pulled high. Optional connection. Low for 250 μsec when data is placed in the data output registers.
16	SDA	Serial Data – I^{2}C Master/Slave Data

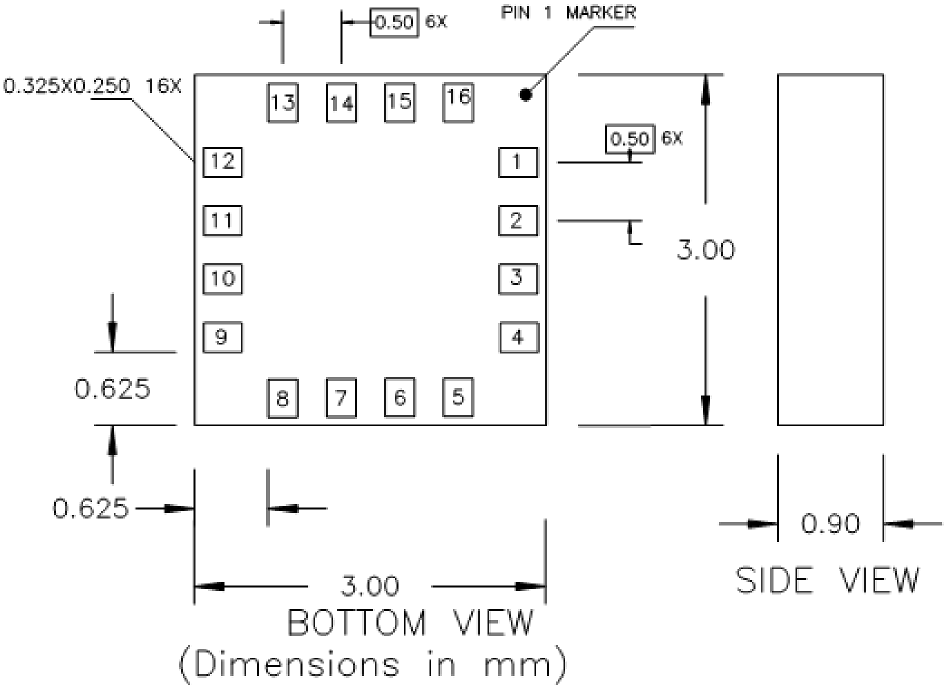
Table 1: Pin Configurations



Arrow indicates direction of magnetic field that generates a positive output reading in Normal Measurement configuration.

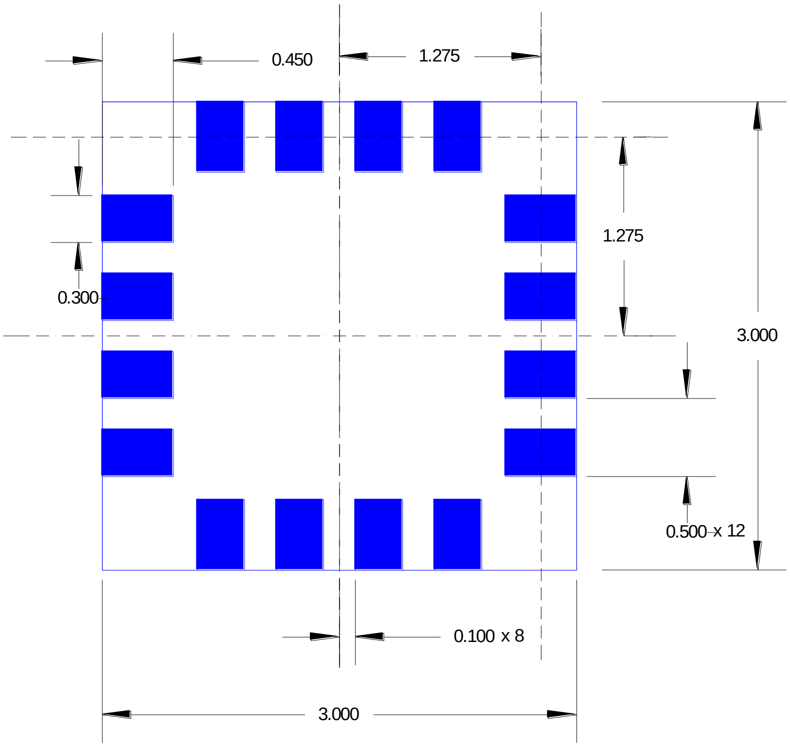
PACKAGE OUTLINES

PACKAGE DRAWING HMC5883L (16-PIN LPCC, dimensions in millimeters)



MOUNTING CONSIDERATIONS

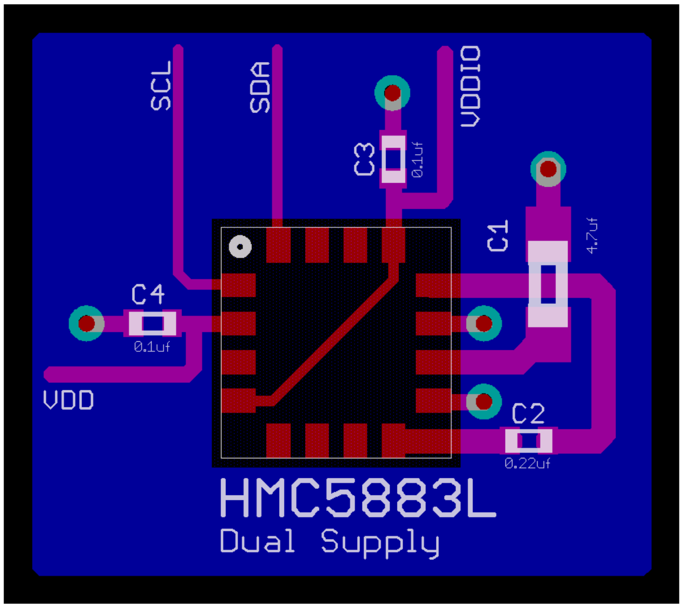
The following is the recommend printed circuit board (PCB) footprint for the HMC5883L.

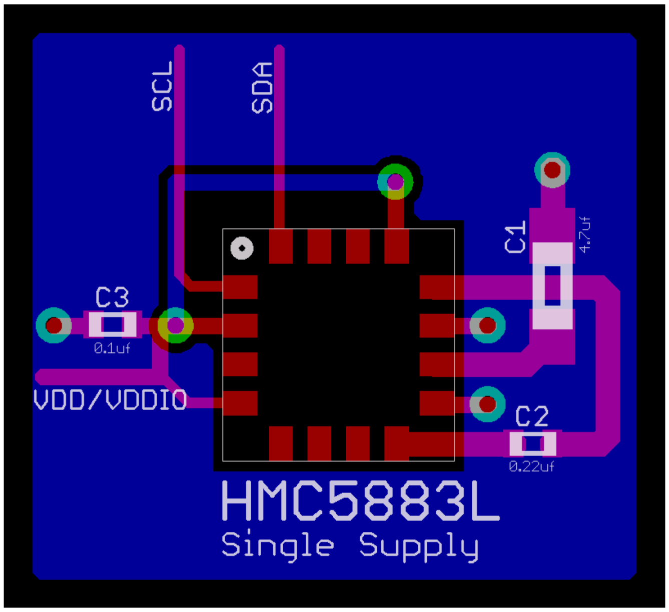


HMC5883 Land Pad Pattern
(All dimensions are in mm)

LAYOUT CONSIDERATIONS

Besides keeping all components that may contain ferrous materials (nickel, etc.) away from the sensor on both sides of the PCB, it is also recommended that there is no conducting copper under/near the sensor in any of the PCB layers. See recommended layout below. Notice that the one trace under the sensor in the dual supply mode is not expected to carry active current since it is for pin 4 pull-up to VDDIO. Power and ground planes are removed under the sensor to minimize possible source of magnetic noise. For best results, use non-ferrous materials for all exposed copper coding.





PCB Pad Definition and Traces

The HMC5883L is a fine pitch LCC package. Refer to previous figure for recommended PCB footprint for proper package centering. Size the traces between the HMC5883L and the external capacitors (C1 and C2) to handle the 1 ampere peak current pulses with low voltage drop on the traces.

Stencil Design and Solder Paste

A 4 mil stencil and 100% paste coverage is recommended for the electrical contact pads.

Reflow Assembly

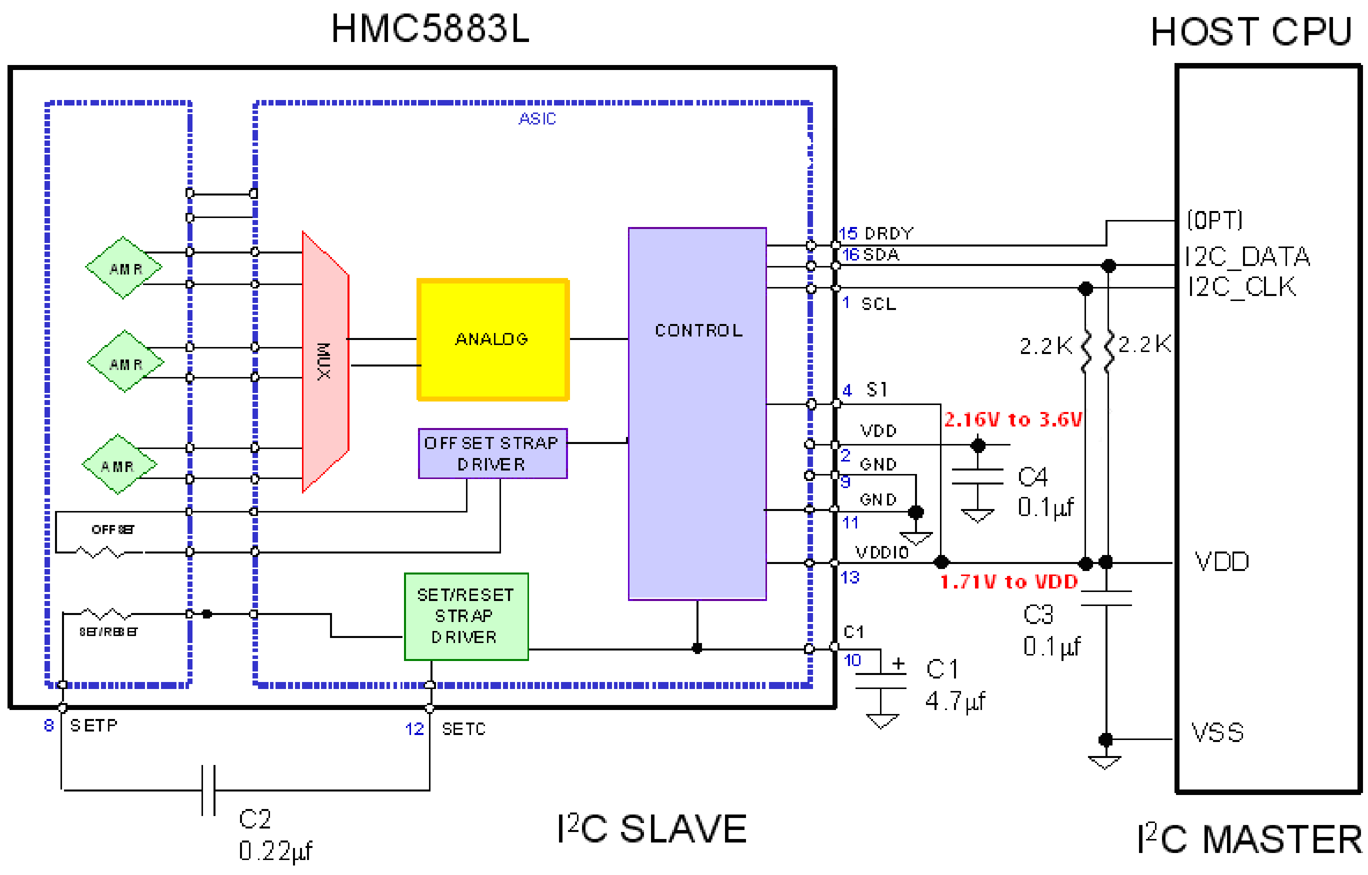
This device is classified as MSL 3 with 260°C peak reflow temperature. A baking process (125°C, 24 hrs) is required if device is not kept continuously in a dry (< 10% RH) environment before assembly. No special reflow profile is required for HMC5883L, which is compatible with lead eutectic and lead-free solder paste reflow profiles. Honeywell recommends adherence to solder paste manufacturer’s guidelines. Hand soldering is not recommended. Built-in self test can be used to verify device functionalities after assembly.

External Capacitors

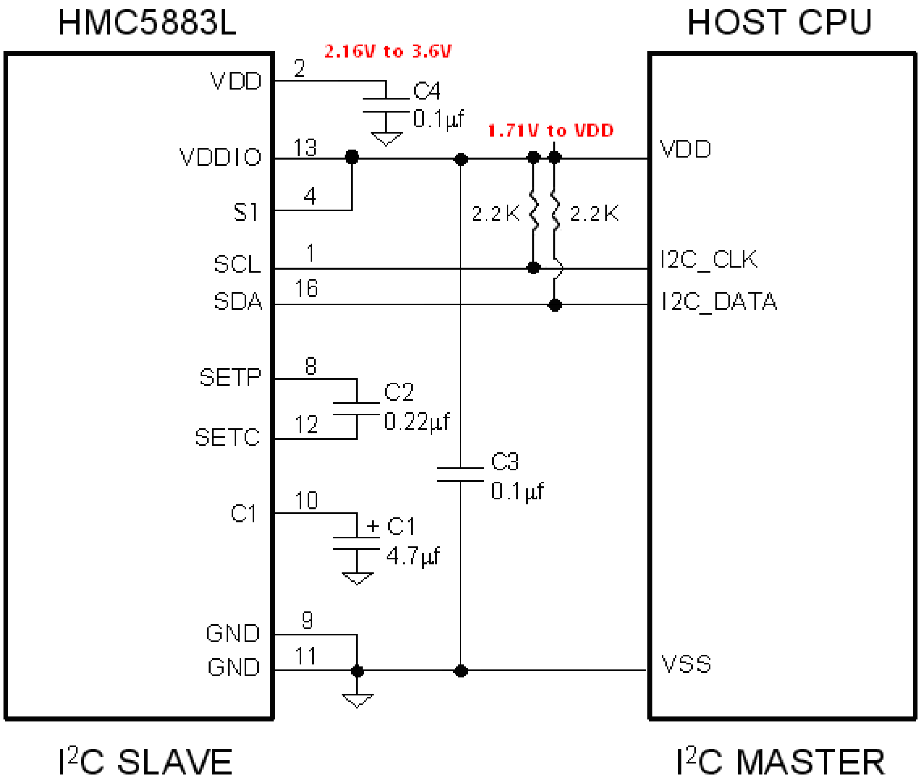
The two external capacitors should be ceramic type construction with low ESR characteristics. The exact ESR values are not critical but values less than 200 milli-ohms are recommended. Reservoir capacitor C1 is nominally 4.7 µF in capacitance, with the set/reset capacitor C2 nominally 0.22 µF in capacitance. Low ESR characteristics may not be in many small SMT ceramic capacitors (0402), so be prepared to up-size the capacitors to gain Low ESR characteristics.

INTERNAL SCHEMATIC DIAGRAM

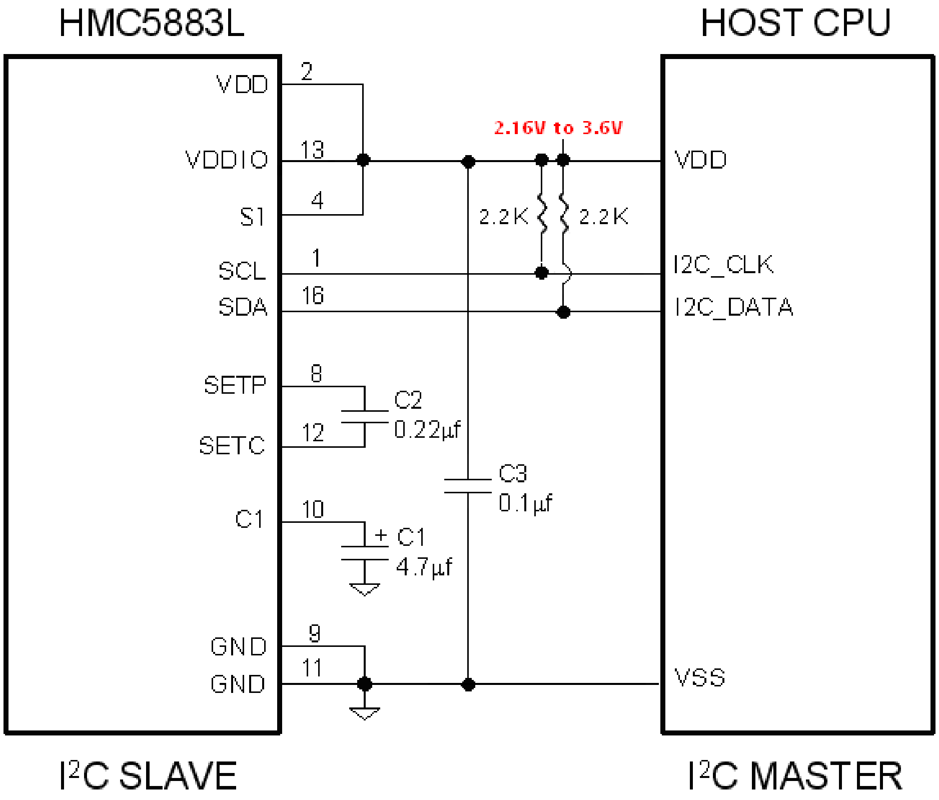
HMC5883L



DUAL SUPPLY REFERENCE DESIGN



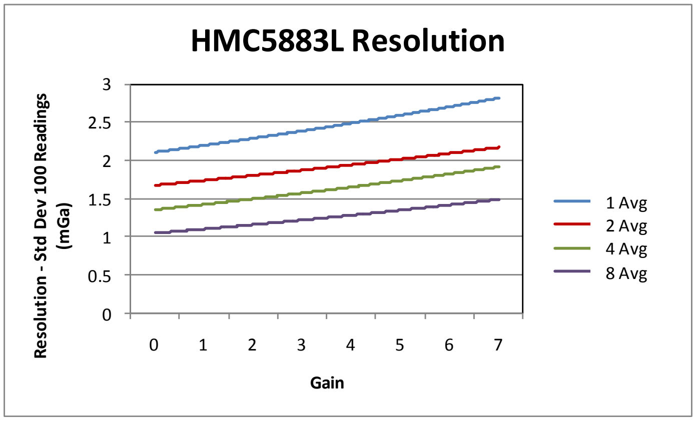
SINGLE SUPPLY REFERENCE DESIGN

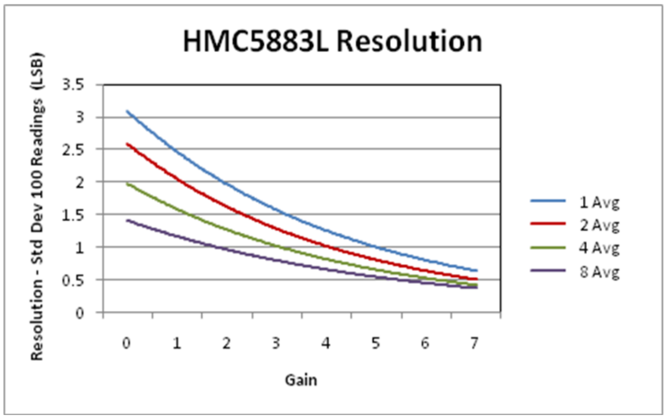


PERFORMANCE

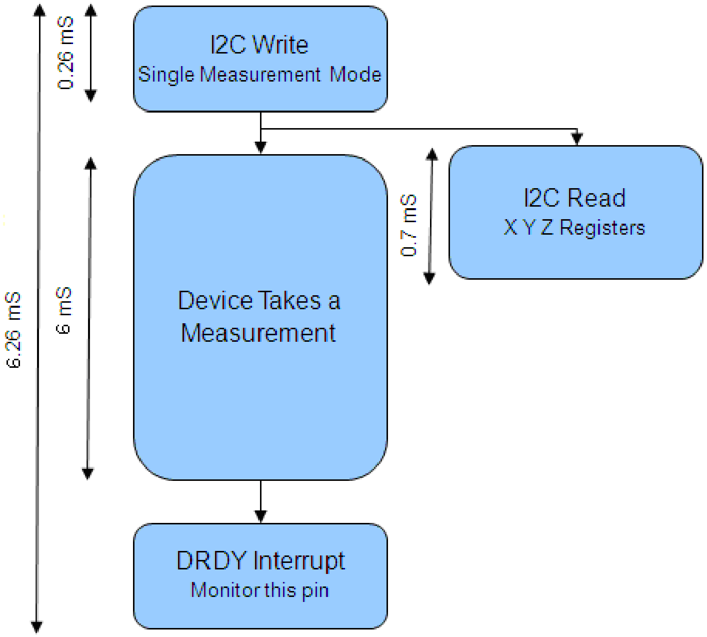
The following graph(s) highlight HMC5883L's performance.

Typical Noise Floor (Field Resolution)





Typical Measurement Period in Single-Measurement Mode



** Monitoring of the DRDY Interrupt pin is only required if maximum output rate is desired.*

BASIC DEVICE OPERATION

Anisotropic Magneto-Resistive Sensors

The Honeywell HMC5883L magnetoresistive sensor circuit is a trio of sensors and application specific support circuits to measure magnetic fields. With power supply applied, the sensor converts any incident magnetic field in the sensitive axis directions to a differential voltage output. The magnetoresistive sensors are made of a nickel-iron (Permalloy) thin-film and patterned as a resistive strip element. In the presence of a magnetic field, a change in the bridge resistive elements causes a corresponding change in voltage across the bridge outputs.

These resistive elements are aligned together to have a common sensitive axis (indicated by arrows in the pinout diagram) that will provide positive voltage change with magnetic fields increasing in the sensitive direction. Because the output is only proportional to the magnetic field component along its axis, additional sensor bridges are placed at orthogonal directions to permit accurate measurement of magnetic field in any orientation.

Self Test

To check the HMC5883L for proper operation, a self test feature in incorporated in which the sensor is internally excited with a nominal magnetic field (in either positive or negative bias configuration). This field is then measured and reported. This function is enabled and the polarity is set by bits MS[n] in the configuration register A. An internal current source generates DC current (about 10 mA) from the VDD supply. This DC current is applied to the offset straps of the magneto-resistive sensor, which creates an artificial magnetic field bias on the sensor. The difference of this measurement and the measurement of the ambient field will be put in the data output register for each of the three axes. By using this built-in function, the manufacturer can quickly verify the sensor's full functionality after the assembly without additional test setup. The self test results can also be used to estimate/compensate the sensor's sensitivity drift due to temperature.

For each "self test measurement", the ASIC:

1. Sends a "Set" pulse
2. Takes one measurement (M1)
3. Sends the (~10 mA) offset current to generate the (~1.1 Gauss) offset field and takes another measurement (M2)
4. Puts the difference of the two measurements in sensor's data output register:

   Output = [M2 – M1] (i.e. output = offset field only)

See SELF TEST OPERATION section later in this datasheet for additional details.

Power Management

This device has two different domains of power supply. The first one is VDD that is the power supply for internal operations and the second one is VDDIO that is dedicated to IO interface. It is possible to work with VDDIO equal to VDD; Single Supply mode, or with VDDIO lower than VDD allowing HMC5883L to be compatible with other devices on board.

I^{2}C Interface

Control of this device is carried out via the I^{2}C bus. This device will be connected to this bus as a slave device under the control of a master device, such as the processor.

This device is compliant with *I^{2}C-Bus Specification*, document number: 9398 393 40011. As an I^{2}C compatible device, this device has a 7-bit serial address and supports I^{2}C protocols. This device supports standard and fast modes, 100kHz and 400kHz, respectively, but does not support the high speed mode (Hs). External pull-up resistors are required to support these standard and fast speed modes.

Activities required by the master (register read and write) have priority over internal activities, such as the measurement. The purpose of this priority is to not keep the master waiting and the I^{2}C bus engaged for longer than necessary.

Internal Clock

The device has an internal clock for internal digital logic functions and timing management. This clock is not available to external usage.

H-Bridge for Set/Reset Strap Drive

The ASIC contains large switching FETs capable of delivering a large but brief pulse to the Set/Reset strap of the sensor. This strap is largely a resistive load. There is no need for an external Set/Reset circuit. The controlling of the Set/Reset function is done automatically by the ASIC for each measurement. One half of the difference from the measurements taken after a set pulse and after a reset pulse will be put in the data output register for each of the three axes. By doing so, the sensor's internal offset and its temperature dependence is removed/cancelled for all measurements. The set/reset pulses also effectively remove the past magnetic history (magnetism) in the sensor, if any.

For each "measurement", the ASIC:

1. Sends a "Set" pulse
2. Takes one measurement (Mset)
3. Sends a "Reset" pulse
4. Takes another measurement (Mreset)
5. Puts the following result in sensor's data output register:

   Output = [Mset – Mreset] / 2

Charge Current Limit

The current that reservoir capacitor (C1) can draw when charging is limited for both single supply and dual supply configurations. This prevents drawing down the supply voltage (VDD).

MODES OF OPERATION

This device has several operating modes whose primary purpose is power management and is controlled by the Mode Register. This section describes these modes.

Continuous-Measurement Mode

During continuous-measurement mode, the device continuously makes measurements, at user selectable rate, and places measured data in data output registers. Data can be re-read from the data output registers if necessary; however, if the master does not ensure that the data register is accessed before the completion of the next measurement, the data output registers are updated with the new measurement. To conserve current between measurements, the device is placed in a state similar to idle mode, but the Mode Register is not changed to Idle Mode. That is, MD[n] bits are unchanged. Settings in the Configuration Register A affect the data output rate (bits DO[n]), the measurement configuration (bits MS[n]), when in continuous-measurement mode. All registers maintain values while in continuous-measurement mode. The I^2C bus is enabled for use by other devices on the network in while continuous-measurement mode.

Single-Measurement Mode

This is the default power-up mode. During single-measurement mode, the device makes a single measurement and places the measured data in data output registers. After the measurement is complete and output data registers are updated, the device is placed in idle mode, and the Mode Register *is* changed to idle mode by setting MD[n] bits. Settings in the configuration register affect the measurement configuration (bits MS[n])when in single-measurement mode. All registers maintain values while in single-measurement mode. The I^2C bus is enabled for use by other devices on the network while in single-measurement mode.

Idle Mode

During this mode the device is accessible through the I^2C bus, but major sources of power consumption are disabled, such as, but not limited to, the ADC, the amplifier, and the sensor bias current. All registers maintain values while in idle mode. The I^2C bus is enabled for use by other devices on the network while in idle mode.

REGISTERS

This device is controlled and configured via a number of on-chip registers, which are described in this section. In the following descriptions, *set* implies a logic 1, and *reset* or *clear* implies a logic 0, unless stated otherwise.

Register List
The table below lists the registers and their access. All address locations are 8 bits.

Address Location	Name	Access
00	Configuration Register A	Read/Write
01	Configuration Register B	Read/Write
02	Mode Register	Read/Write
03	Data Output X MSB Register	Read
04	Data Output X LSB Register	Read
05	Data Output Z MSB Register	Read
06	Data Output Z LSB Register	Read
07	Data Output Y MSB Register	Read
08	Data Output Y LSB Register	Read
09	Status Register	Read
10	Identification Register A	Read
11	Identification Register B	Read
12	Identification Register C	Read

Table2: Register List

Register Access

This section describes the process of reading from and writing to this device. The devices uses an address pointer to indicate which register location is to be read from or written to. These pointer locations are sent from the master to this slave device and succeed the 7-bit address (0x1E) plus 1 bit read/write identifier, i.e. 0x3D for read and 0x3C for write.

To minimize the communication between the master and this device, the address pointer updated automatically without master intervention. The register pointer will be incremented by 1 automatically after the current register has been read successfully.

The address pointer value itself cannot be read via the I^{2}C bus.
Any attempt to read an invalid address location returns 0's, and any write to an invalid address location or an undefined bit within a valid address location is ignored by this device.

To move the address pointer to a random register location, first issue a "write" to that register location with no data byte following the commend. For example, to move the address pointer to register 10, send 0x3C 0x0A.

Configuration Register A

The configuration register is used to configure the device for setting the data output rate and measurement configuration. CRA0 through CRA7 indicate bit locations, with *CRA* denoting the bits that are in the configuration register. CRA7 denotes the first bit of the data stream. The number in parenthesis indicates the default value of that bit.CRA default is 0x10.

CRA7	CRA6	CRA5	CRA4	CRA3	CRA2	CRA1	CRA0
(0)	MA1(0)	MA0(0)	DO2 (1)	DO1 (0)	DO0 (0)	MS1 (0)	MS0 (0)

Table 3: Configuration Register A

Location	Name	Description
CRA7	CRA7	Bit CRA7 is reserved for future function. Set to 0 when configuring CRA.
CRA6 to CRA5	MA1 to MA0	Select number of samples averaged (1 to 8) per measurement output. 00 = 1(Default); 01 = 2; 10 = 4; 11 = 8
CRA4 to CRA2	DO2 to DO0	Data Output Rate Bits. These bits set the rate at which data is written to all three data output registers.
CRA1 to CRA0	MS1 to MS0	Measurement Configuration Bits. These bits define the measurement flow of the device, specifically whether or not to incorporate an applied bias into the measurement.

Table 4: Configuration Register A Bit Designations

The Table below shows all selectable output rates in continuous measurement mode. All three channels shall be measured within a given output rate. Other output rates with maximum rate of 160 Hz can be achieved by monitoring DRDY interrupt pin in single measurement mode.

DO2	DO1	DO0	Typical Data Output Rate (Hz)
0	0	0	0.75
0	0	1	1.5
0	1	0	3
0	1	1	7.5
1	0	0	15 (Default)
1	0	1	30
1	1	0	75
1	1	1	Reserved

Table 5: Data Output Rates

MS1	MS0	Measurement Mode
0	0	Normal measurement configuration (Default). In normal measurement configuration the device follows normal measurement flow. The positive and negative pins of the resistive load are left floating and high impedance.
0	1	Positive bias configuration for X, Y, and Z axes. In this configuration, a positive current is forced across the resistive load for all three axes.
1	0	Negative bias configuration for X, Y and Z axes. In this configuration, a negative current is forced across the resistive load for all three axes..
1	1	This configuration is reserved.

Table 6: Measurement Modes

Configuration Register B

The configuration register B for setting the device gain. CRB0 through CRB7 indicate bit locations, with *CRB* denoting the bits that are in the configuration register. CRB7 denotes the first bit of the data stream. The number in parenthesis indicates the default value of that bit. CRB default is 0x20.

CRB7	CRB6	CRB5	CRB4	CRB3	CRB2	CRB1	CRB0
GN2 (0)	GN1 (0)	GN0 (1)	(0)	(0)	(0)	(0)	(0)

Table 7: Configuration B Register

Location	Name	Description
CRB7 to CRB5	GN2 to GN0	Gain Configuration Bits. These bits configure the gain for the device. The gain configuration is common for all channels.
CRB4 to CRB0	0	These bits must be cleared for correct operation.

Table 8: Configuration Register B Bit Designations

The table below shows nominal gain settings. Use the “Gain” column to convert counts to Gauss. The “Digital Resolution” column is the theoretical value in term of milli-Gauss per count (LSb) which is the inverse of the values in the “Gain” column. The effective resolution of the usable signal also depends on the noise floor of the system, i.e.

Effective Resolution = Max (Digital Resolution, Noise Floor)

Choose a lower gain value (higher GN#) when total field strength causes overflow in one of the data output registers (saturation). Note that the very first measurement after a gain change maintains the same gain as the previous setting. **The new gain setting is effective from the second measurement and on.**

GN2	GN1	GN0	Recommended Sensor Field Range	Gain (LSb/ Gauss)	Digital Resolution (mG/LSb)	Output Range
0	0	0	± 0.88 Ga	1370	0.73	0xF800–0x07FF (-2048–2047)
0	0	1	± 1.3 Ga	1090 (default)	0.92	0xF800–0x07FF (-2048–2047)
0	1	0	± 1.9 Ga	820	1.22	0xF800–0x07FF (-2048–2047)
0	1	1	± 2.5 Ga	660	1.52	0xF800–0x07FF (-2048–2047)
1	0	0	± 4.0 Ga	440	2.27	0xF800–0x07FF (-2048–2047)
1	0	1	± 4.7 Ga	390	2.56	0xF800–0x07FF (-2048–2047)
1	1	0	± 5.6 Ga	330	3.03	0xF800–0x07FF (-2048–2047)
1	1	1	± 8.1 Ga	230	4.35	0xF800–0x07FF (-2048–2047)

Table 9: Gain Settings

Mode Register

The mode register is an 8-bit register from which data can be read or to which data can be written. This register is used to select the operating mode of the device. MR0 through MR7 indicate bit locations, with *MR* denoting the bits that are in the mode register. MR7 denotes the first bit of the data stream. The number in parenthesis indicates the default value of that bit. Mode register default is 0x01.

MR7	MR6	MR5	MR4	MR3	MR2	MR1	MR0
HS(0)	(0)	(0)	(0)	(0)	(0)	MD1 (0)	MD0 (1)

Table 10: Mode Register

Location	Name	Description
MR7 to MR2	HS	Set this pin to enable High Speed I2C, 3400kHz.
MR1 to MR0	MD1 to MD0	Mode Select Bits. These bits select the operation mode of this device.

Table 11: Mode Register Bit Designations

MD1	MD0	Operating Mode
0	0	Continuous-Measurement Mode. In continuous-measurement mode, the device continuously performs measurements and places the result in the data register. RDY goes high when new data is placed in all three registers. After a power-on or a write to the mode or configuration register, the first measurement set is available from all three data output registers after a period of $2/f_{DO}$ and subsequent measurements are available at a frequency of f_{DO}, where f_{DO} is the frequency of data output.
0	1	Single-Measurement Mode (Default). When single-measurement mode is selected, device performs a single measurement, sets RDY high and returned to idle mode. Mode register returns to idle mode bit values. The measurement remains in the data output register and RDY remains high until the data output register is read or another measurement is performed.
1	0	Idle Mode. Device is placed in idle mode.
1	1	Idle Mode. Device is placed in idle mode.

Table 12: Operating Modes

Data Output X Registers A and B

The data output X registers are two 8-bit registers, data output register A and data output register B. These registers store the measurement result from channel X. Data output X register A contains the MSB from the measurement result, and data output X register B contains the LSB from the measurement result. The value stored in these two registers is a 16-bit value in 2's complement form, whose range is 0xF800 to 0x07FF. DXRA0 through DXRA7 and DXRB0 through DXRB7 indicate bit locations, with *DXRA* and *DXRB* denoting the bits that are in the data output X registers. DXRA7 and DXRB7 denote the first bit of the data stream. The number in parenthesis indicates the default value of that bit.

In the event the ADC reading overflows or underflows for the given channel, or if there is a math overflow during the bias measurement, this data register will contain the value -4096. This register value will clear when after the next valid measurement is made.

DXRA7	**DXRA6**	**DXRA5**	**DXRA4**	**DXRA3**	**DXRA2**	**DXRA1**	**DXRA0**
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)
DXRB7	**DXRB6**	**DXRB5**	**DXRB4**	**DXRB3**	**DXRB2**	**DXRB1**	**DXRB0**
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

Table 13: Data Output X Registers A and B

Data Output Y Registers A and B

The data output Y registers are two 8-bit registers, data output register A and data output register B. These registers store the measurement result from channel Y. Data output Y register A contains the MSB from the measurement result, and data output Y register B contains the LSB from the measurement result. The value stored in these two registers is a 16-bit value in 2's complement form, whose range is 0xF800 to 0x07FF. DYRA0 through DYRA7 and DYRB0 through DYRB7 indicate bit locations, with *DYRA* and *DYRB* denoting the bits that are in the data output Y registers. DYRA7 and DYRB7 denote the first bit of the data stream. The number in parenthesis indicates the default value of that bit.

In the event the ADC reading overflows or underflows for the given channel, or if there is a math overflow during the bias measurement, this data register will contain the value -4096. This register value will clear when after the next valid measurement is made.

DYRA7	**DYRA6**	**DYRA5**	**DYRA4**	**DYRA3**	**DYRA2**	**DYRA1**	**DYRA0**
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)
DYRB7	**DYRB6**	**DYRB5**	**DYRB4**	**DYRB3**	**DYRB2**	**DYRB1**	**DYRB0**
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

Table 14: Data Output Y Registers A and B

Data Output Z Registers A and B

The data output Z registers are two 8-bit registers, data output register A and data output register B. These registers store the measurement result from channel Z. Data output Z register A contains the MSB from the measurement result, and data output Z register B contains the LSB from the measurement result. The value stored in these two registers is a 16-bit value in 2's complement form, whose range is 0xF800 to 0x07FF. DZRA0 through DZRA7 and DZRB0 through DZRB7 indicate bit locations, with *DZRA* and *DZRB* denoting the bits that are in the data output Z registers. DZRA7 and DZRB7 denote the first bit of the data stream. The number in parenthesis indicates the default value of that bit.

In the event the ADC reading overflows or underflows for the given channel, or if there is a math overflow during the bias measurement, this data register will contain the value -4096. This register value will clear when after the next valid measurement is made.

DZRA7	DZRA6	DZRA5	DZRA4	DZRA3	DZRA2	DZRA1	DZRA0
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)
DZRB7	**DZRB6**	**DZRB5**	**DZRB4**	**DZRB3**	**DZRB2**	**DZRB1**	**DZRB0**
(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)

Table 15: Data Output Z Registers A and B

Data Output Register Operation

When one or more of the output registers are read, new data cannot be placed in any of the output data registers until all six data output registers are read. This requirement also impacts DRDY and RDY, which cannot be cleared until new data is placed in all the output registers.

Status Register

The status register is an 8-bit read-only register. This register is used to indicate device status. SR0 through SR7 indicate bit locations, with *SR* denoting the bits that are in the status register. SR7 denotes the first bit of the data stream.

SR7	SR6	SR5	SR4	SR3	SR2	SR1	SR0
(0)	(0)	(0)	(0)	(0)	(0)	LOCK (0)	RDY(0)

Table 16: Status Register

Location	Name	Description
SR7 to SR2	0	These bits are reserved.
SR1	LOCK	Data output register lock. This bit is set when: 1.some but not all for of the six data output registers have been read, 2. Mode register has been read. When this bit is set, the six data output registers are locked and any new data will not be placed in these register until one of these conditions are met: 1.all six bytes have been read, 2. the mode register is changed, 3. the measurement configuration (CRA) is changed, 4. power is reset.
SR0	RDY	Ready Bit. Set when data is written to all six data registers. Cleared when device initiates a write to the data output registers and after one or more of the data output registers are written to. When RDY bit is clear it shall remain cleared for a 250 μs. DRDY pin can be used as an alternative to the status register for monitoring the device for measurement data.

Table 17: Status Register Bit Designations

Identification Register A

The identification register A is used to identify the device. IRA0 through IRA7 indicate bit locations, with *IRA* denoting the bits that are in the identification register A. IRA7 denotes the first bit of the data stream. The number in parenthesis indicates the default value of that bit.

The identification value for this device is stored in this register. This is a read-only register.
Register values. ASCII value *H*

IRA7	IRA6	IRA5	IRA4	IRA3	IRA2	IRA1	IRA0
0	1	0	0	1	0	0	0

Table 18: Identification Register A Default Values

Identification Register B

The identification register B is used to identify the device. IRB0 through IRB7 indicate bit locations, with *IRB* denoting the bits that are in the identification register A. IRB7 denotes the first bit of the data stream.

Register values. ASCII value *4*

IRB7	IRB6	IRB5	IRB4	IRB3	IRB2	IRB1	IRB0
0	0	1	1	0	1	0	0

Table 19: Identification Register B Default Values

Identification Register C

The identification register C is used to identify the device. IRC0 through IRC7 indicate bit locations, with *IRC* denoting the bits that are in the identification register A. IRC7 denotes the first bit of the data stream.

Register values. ASCII value *3*

IRC7	IRC6	IRC5	IRC4	IRC3	IRC2	IRC1	IRC0
0	0	1	1	0	0	1	1

Table 20: Identification Register C Default Values

I^{2}C COMMUNICATION PROTOCOL

The HMC5883L communicates via a two-wire I^{2}C bus system as a slave device. The HMC5883L uses a simple protocol with the interface protocol defined by the I^{2}C bus specification, and by this document. The data rate is at the standard-mode 100kbps or 400kbps rates as defined in the I^{2}C Bus Specifications. The bus bit format is an 8-bit Data/Address send and a 1-bit acknowledge bit. The format of the data bytes (payload) shall be case sensitive ASCII characters or binary data to the HMC5883L slave, and binary data returned. Negative binary values will be in two's complement form. The default (factory) HMC5883L 8-bit slave address is 0x3C for write operations, or 0x3D for read operations.

The HMC5883L Serial Clock (SCL) and Serial Data (SDA) lines require resistive pull-ups (Rp) between the master device (usually a host microprocessor) and the HMC5883L. Pull-up resistance values of about 2.2K to 10K ohms are recommended with a nominal VDDIO voltage. Other resistor values may be used as defined in the I^{2}C Bus Specifications that can be tied to VDDIO.

The SCL and SDA lines in this bus specification may be connected to multiple devices. The bus can be a single master to multiple slaves, or it can be a multiple master configuration. All data transfers are initiated by the master device, which is responsible for generating the clock signal, and the data transfers are 8 bit long. All devices are addressed by I^{2}C's unique 7-bit address. After each 8-bit transfer, the master device generates a 9th clock pulse, and releases the SDA line. The receiving device (addressed slave) will pull the SDA line low to acknowledge (ACK) the successful transfer or leave the SDA high to negative acknowledge (NACK).

Per the I^{2}C spec, all transitions in the SDA line must occur when SCL is low. This requirement leads to two unique conditions on the bus associated with the SDA transitions when SCL is high. Master device pulling the SDA line low while the SCL line is high indicates the Start (S) condition, and the Stop (P) condition is when the SDA line is pulled high while the SCL line is high. The I^{2}C protocol also allows for the Restart condition in which the master device issues a second start condition without issuing a stop.

All bus transactions begin with the master device issuing the start sequence followed by the slave address byte. The address byte contains the slave address; the upper 7 bits (bits7-1), and the Least Significant bit (LSb). The LSb of the address byte designates if the operation is a read (LSb=1) or a write (LSb=0). At the 9th clock pulse, the receiving slave device will issue the ACK (or NACK). Following these bus events, the master will send data bytes for a write operation, or the slave will clock out data with a read operation. All bus transactions are terminated with the master issuing a stop sequence.

I^{2}C bus control can be implemented with either hardware logic or in software. Typical hardware designs will release the SDA and SCL lines as appropriate to allow the slave device to manipulate these lines. In a software implementation, care must be taken to perform these tasks in code.

OPERATIONAL EXAMPLES

The HMC5883L has a fairly quick stabilization time from no voltage to stable and ready for data retrieval. The nominal 56 milli-seconds with the factory default single measurement mode means that the six bytes of magnetic data registers (DXRA, DXRB, DZRA, DZRB, DYRA, and DYRB) are filled with a valid first measurement.

To change the measurement mode to continuous measurement mode, after the power-up time send the three bytes:

0x3C 0x02 0x00

This writes the 00 into the second register or mode register to switch from single to continuous measurement mode setting. With the data rate at the factory default of 15Hz updates, a 67 milli-second typical delay should be allowed by the I^{2}C master before querying the HMC5883L data registers for new measurements. To clock out the new data, send:

0x3D, and clock out DXRA, DXRB, DZRA, DZRB, DYRA, and DYRB located in registers 3 through 8. The HMC5883L will automatically re-point back to register 3 for the next 0x3D query. All six data registers must be read properly before new data can be placed in any of these data registers.

Below is an example of a (power-on) initialization process for “continuous-measurement mode”:

1. Write CRA (00) – send **0x3C** **0x00** **0x70** (8-average, 15 Hz default, normal measurement)
2. Write CRB (01) – send **0x3C** **0x01** **0xA0** (Gain=5, or any other desired gain)
3. Write Mode (02) – send **0x3C** **0x02** **0x00** (Continuous-measurement mode)
4. Wait 6 ms or monitor status register or DRDY hardware interrupt pin
5. Loop

        Send **0x3D 0x06** (Read all 6 bytes. If gain is changed then this data set is using previous gain)
        Convert three 16-bit 2’s compliment hex values to decimal values and assign to X, Z, Y, respectively.
        Send **0x3C 0x03** (point to first data register 03)
        Wait about 67 ms (if 15 Hz rate) or monitor status register or DRDY hardware interrupt pin

    End_loop

Below is an example of a (power-on) initialization process for “single-measurement mode”:

1. Write CRA (00) – send **0x3C** **0x00** **0x70** (8-average, 15 Hz default or any other rate, normal measurement)
2. Write CRB (01) – send **0x3C** **0x01** **0xA0** (Gain=5, or any other desired gain)
3. For each measurement query:

        Write Mode (02) – send **0x3C** **0x02** **0x01** (Single-measurement mode)
        Wait 6 ms or monitor status register or DRDY hardware interrupt pin
        Send **0x3D 0x06** (Read all 6 bytes. If gain is changed then this data set is using previous gain)
        Convert three 16-bit 2’s compliment hex values to decimal values and assign to X, Z, Y, respectively.

SELF TEST OPERATION

To check the HMC5883L for proper operation, a self test feature in incorporated in which the sensor offset straps are excited to create a nominal field strength (bias field) to be measured. To implement self test, the least significant bits (MS1 and MS0) of configuration register A are changed from 00 to 01 (positive bias) or 10 (negetive bias).

Then, by placing the mode register into single or continuous-measurement mode, two data acquisition cycles will be made on each magnetic vector. The first acquisition will be a set pulse followed shortly by measurement data of the external field. The second acquisition will have the offset strap excited (about 10 mA) in the positive bias mode for X, Y, and Z axes to create about a 1.1 gauss self test field plus the external field. The first acquisition values will be subtracted from the second acquisition, and the net measurement will be placed into the data output registers.

Since self test adds ~1.1 Gauss additional field to the existing field strength, using a reduced gain setting prevents sensor from being saturated and data registers overflowed. For example, if the configuration register B is set to 0xA0 (Gain=5), values around +452 LSb (1.16 Ga * 390 LSb/Ga) will be placed in the X and Y data output registers and around +421 (1.08 Ga * 390 LSb/Ga) will be placed in Z data output register. To leave the self test mode, change MS1 and MS0 bit of the configuration register A back to 00 (Normal Measurement Mode). Acceptable limits of the self test values depend on the gain setting. Limits for Gain=5 is provided in the specification table.

Below is an example of a "positive self test" process using continuous-measurement mode:

```
1.  Write CRA (00) – send 0x3C 0x00 0x71 (8-average, 15 Hz default, positive self test measurement)
2.  Write CRB (01) – send 0x3C 0x01 0xA0 (Gain=5)
3.  Write Mode (02) – send 0x3C 0x02 0x00 (Continuous-measurement mode)
4.  Wait 6 ms or monitor status register or DRDY hardware interrupt pin
5.  Loop
            Send 0x3D 0x06 (Read all 6 bytes. If gain is changed then this data set is using previous gain)
            Convert three 16-bit 2's compliment hex values to decimal values and assign to X, Z, Y, respectively.
            Send 0x3C 0x03 (point to first data register 03)
            Wait about 67 ms (if 15 Hz rate) or monitor status register or DRDY hardware interrupt pin
    End_loop
6.  Check limits –
    If all 3 axes (X, Y, and Z) are within reasonable limits (243 to 575 for Gain=5, adjust these limits basing on the
    gain setting used. See an example below.) Then
            All 3 axes pass positive self test
            Write CRA (00) – send 0x3C 0x00 0x70 (Exit self test mode and this procedure)
    Else
            If Gain<7
                    Write CRB (01) – send 0x3C 0x01 0x_0 (Increase gain setting and retry, skip the next data set)
            Else
                    At least one axis did not pass positive self test
                    Write CRA (00) – send 0x3C 0x00 0x70 (Exit self test mode and this procedure)
    End If
```

Below is an example of how to adjust the "positive self" test limits basing on the gain setting:

1. If Gain = 6, self test limits are:
   Low Limit = 243 * 330/390 = 206
   High Limit = 575 * 330/390 = 487

2. If Gain = 7, self test limits are:
   Low Limit = 243 * 230/390 = 143
   High Limit = 575 * 230/390 = 339

SCALE FACTOR TEMPERATURE COMPENSATION

The built-in self test can also be used to periodically compensate the scaling errors due to temperature variations. A compensation factor can be found by comparing the self test outputs with the ones obtained at a known temperature. For example, if the self test output is 400 at room temperature and 300 at the current temperature then a compensation factor of (400/300) should be applied to all current magnetic readings. A temperature sensor is not required using this method.

Below is an example of a temperature compensation process using positive self test method:

1. If self test measurement at a temperature "when the last magnetic calibration was done":
   X_STP = 400
   Y_STP = 410
   Z_STP = 420
2. If self test measurement at a different tmperature:
   X_STP = 300 (Lower than before)
   Y_STP = 310 (Lower than before)
   Z_STP = 320 (Lower than before)
   Then
   X_TempComp = 400/300
   Y_TempComp = 410/310
   Z_TempComp = 420/320
3. Applying to all new measurements:
   X = X * X_TempComp
   Y = Y * Y_TempComp
   Z = Z * Z_TempComp
   Now all 3 axes are temperature compensated, i.e. sensitivity is same as "when the last magnetic calibration was done"; therefore, the calibration coefficients can be applied without modification.
4. Repeat this process periodically or,for every Δt degrees of temperature change measured, if available.

ORDERING INFORMATION

Ordering Number	Product
HMC5883L-T HMC5883L-TR	Cut Tape Tape and Reel 4k pieces/reel

Caution
This part is sensitive to damage by electrostatic discharge. Use ESD precautionary procedures when touching, removing or inserting.

CAUTION: ESDS CAT. 1B

FIND OUT MORE

For more information on Honeywell's Magnetic Sensors visit us online at www.magneticsensors.com or contact us at 1-800-323-8295 (763-954-2474 internationally).

The application circuits herein constitute typical usage and interface of Honeywell product. Honeywell does not warranty or assume liability of customer-designed circuits derived from this description or depiction.

Honeywell reserves the right to make changes to improve reliability, function or design. Honeywell does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others.

U.S. Patents 4,441,072, 4,533,872, 4,569,742, 4,681,812, 4,847,584 and 6,529,114 apply to the technology described

Honeywell
12001 Highway 55
Plymouth, MN 55441
Tel: 800-323-8295
www.magneticsensors.com

Honeywell

Burr-Brown Products
from Texas Instruments

TMP100
TMP101

SBOS231G − JANUARY 2002 − REVISED NOVEMBER 2007

Digital Temperature Sensor with I2C™ Interface

FEATURES

- **DIGITAL OUTPUT: I2C Serial 2-Wire**
- **RESOLUTION: 9- to 12-Bits, User-Selectable**
- **ACCURACY:**
  ±2.0°C from −25°C to +85°C (max)
  ±3.0°C from −55°C to +125°C (max)
- **LOW QUIESCENT CURRENT:**
  45μA, 0.1μA Standby
- **WIDE SUPPLY RANGE: 2.7V to 5.5V**
- **TINY SOT23-6 PACKAGE**

APPLICATIONS

- **POWER-SUPPLY TEMPERATURE MONITORING**
- **COMPUTER PERIPHERAL THERMAL PROTECTION**
- **NOTEBOOK COMPUTERS**
- **CELL PHONES**
- **BATTERY MANAGEMENT**
- **OFFICE MACHINES**
- **THERMOSTAT CONTROLS**
- **ENVIRONMENTAL MONITORING AND HVAC**
- **ELECTROMECHANICAL DEVICE TEMPERATURE**

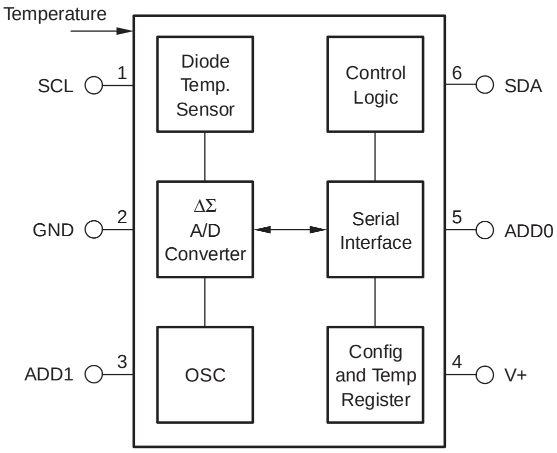
DESCRIPTION

The TMP100 and TMP101 are two-wire, serial output temperature sensors available in SOT23-6 packages. Requiring no external components, the TMP100 and TMP101 are capable of reading temperatures with a resolution of 0.0625°C.

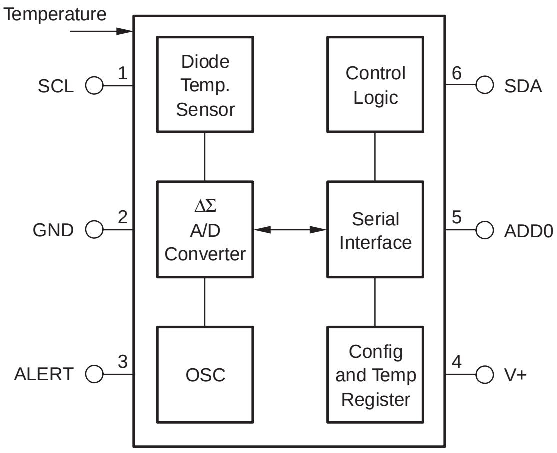
The TMP100 and TMP101 feature SMBus and I2C interface compatibility, with the TMP100 allowing up to eight devices on one bus. The TMP101 offers SMBus alert function with up to three devices per bus.

The TMP100 and TMP101 are ideal for extended temperature measurement in a variety of communication, computer, consumer, environmental, industrial, and instrumentation applications.

The TMP100 and TMP101 are specified for operation over a temperature range of −55°C to +125°C.



TMP100



TMP101

Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

I2C is a trademark of NXP Semiconductors. All other trademarks are the property of their respective owners.

ABSOLUTE MAXIMUM RATINGS[1]

Power Supply, V+	7.5V
Input Voltage[2]	−0.5V to 7.5V
Operating Temperature Range	−55°C to +125°C
Storage Temperature Range	−60°C to +150°C
Junction Temperature (T_J max)	+150°C
ESD Rating, Human Body Model	2000V
Machine Model	200V

[1] Stresses above these ratings may cause permanent damage. Exposure to absolute maximum conditions for extended periods may degrade device reliability. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those specified is not supported.

[2] Input voltage rating applies to all TMP100 and TMP101 input voltages.

This integrated circuit can be damaged by ESD. Texas Instruments recommends that all integrated circuits be handled with appropriate precautions. Failure to observe proper handling and installation procedures can cause damage.

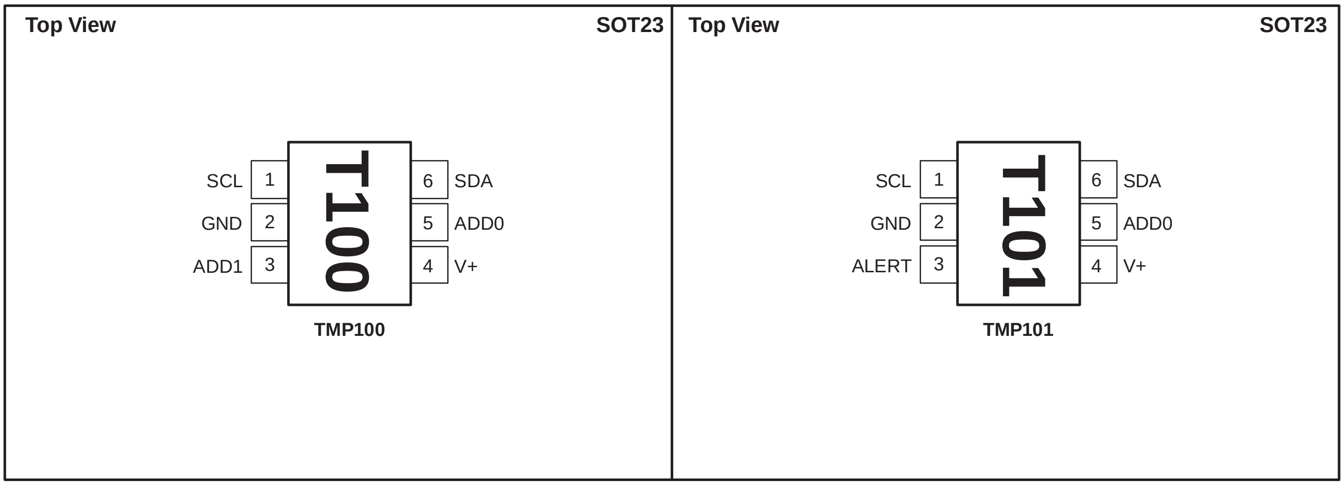
ESD damage can range from subtle performance degradation to complete device failure. Precision integrated circuits may be more susceptible to damage because very small parametric changes could cause the device not to meet its published specifications.

ORDERING INFORMATION[1]

PRODUCT	PACKAGE-LEAD	PACKAGE DESIGNATOR	PACKAGE MARKING
TMP100	SOT23-6	DBV	T100
TMP101	SOT23-6	DBV	T101

[1] For the most current package and ordering information, see the Package Option Addendum at the end of this document, or see the TI web site at www.ti.com.

PIN CONFIGURATION



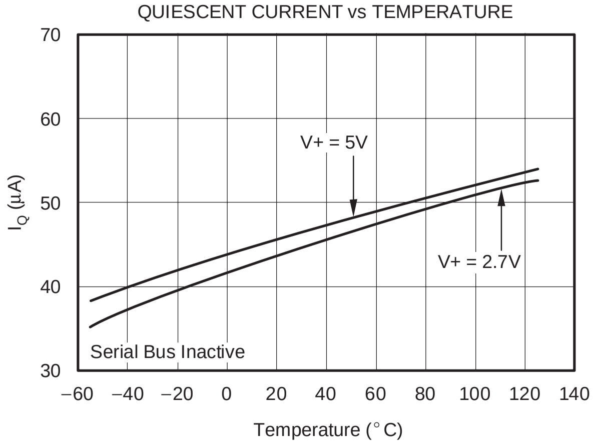
ELECTRICAL CHARACTERISTICS

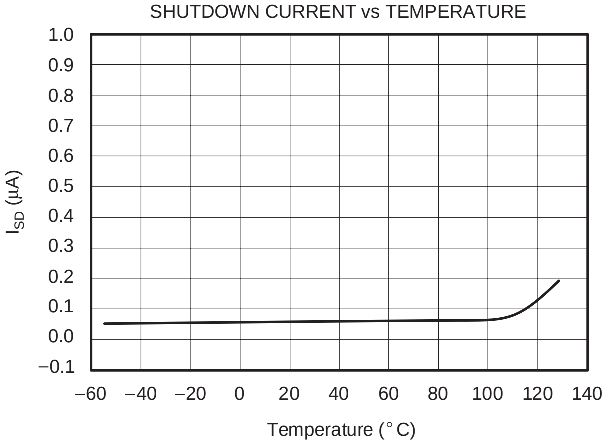
At T_A = −55°C to +125°C and V+ = 2.7V to 5.5V, unless otherwise noted.

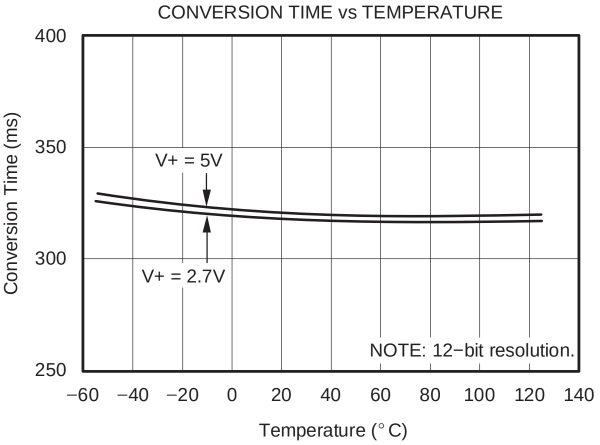
PARAMETER	TEST CONDITIONS	TMP100, TMP101			UNIT
		MIN	TYP	MAX	
TEMPERATURE INPUT					
Range		−55		+125	°C
Accuracy (temperature error)	−25°C to +85°C		±0.5	±2.0	°C
	−55°C to +125°C		±1.0	±3.0	°C
Resolution	Selectable		±0.0625		°C
DIGITAL INPUT/OUTPUT					
Input Logic Levels:					
V_{IH}		0.7(V+)		6.0	V
V_{IL}		−0.5		0.3(V+)	V
Input Current, I_{IN}	$0V \leq V_{IN} \leq 6V$			1	µA
Output Logic Levels:					
V_{OL} SDA	I_{OL} = 3mA	0	0.15	0.4	V
V_{OL} ALERT	I_{OL} = 4mA	0	0.15	0.4	V
Resolution	Selectable		9 to 12		Bits
Conversion Time	9-Bit		40	75	ms
	10-Bit		80	150	ms
	11-Bit		160	300	ms
	12-Bit		320	600	ms
Conversion Rate	9-Bit		25		s/s
	10-Bit		12		s/s
	11-Bit		6		s/s
	12-Bit		3		s/s
POWER SUPPLY					
Operating Range		2.7		5.5	V
Quiescent Current I_Q	Serial Bus Inactive		45	75	µA
	Serial Bus Active, SCL Frequency = 400kHz		70		µA
	Serial Bus Active, SCL Frequency = 3.4MHz		150		µA
Shutdown Current I_{SD}	Serial Bus Inactive		0.1	1	µA
	Serial Bus Active, SCL Frequency = 400kHz		20		µA
	Serial Bus Active, SCL Frequency = 3.4MHz		100		µA
TEMPERATURE RANGE					
Specified Range		−55		+125	°C
Storage Range		−60		+150	°C
Thermal Resistance θ_{JA}	SOT23-6 Surface-Mount		200		°C/W

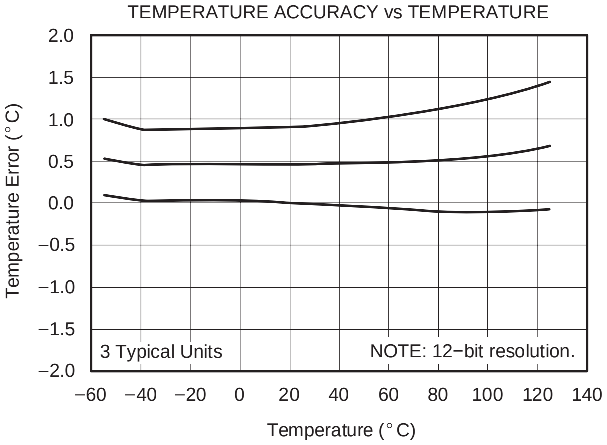
TYPICAL CHARACTERISTICS

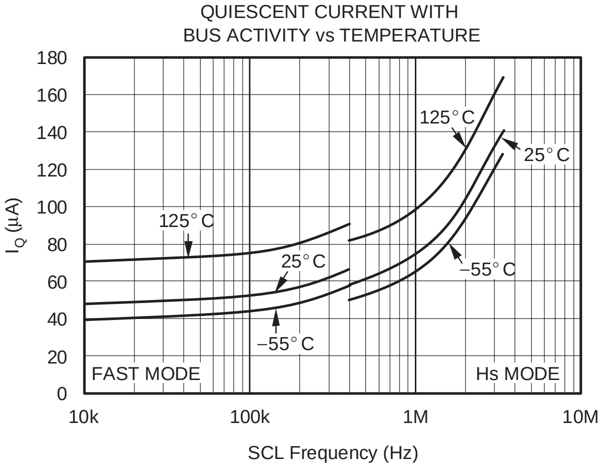
At T_A = +25°C and V+ = 5.0V, unless otherwise noted.







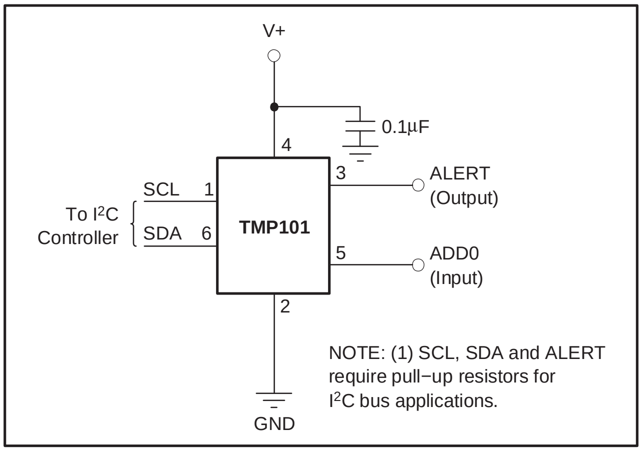




APPLICATIONS INFORMATION

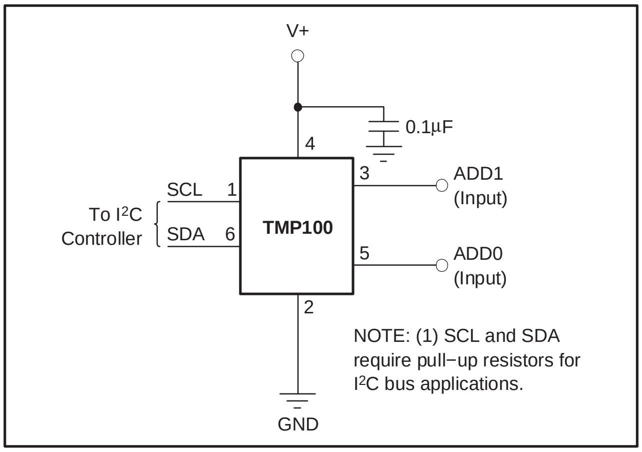
The TMP100 and TMP101 are digital temperature sensors optimal for thermal management and thermal protection applications. The TMP100 and TMP101 are I^{2}C and SMBus interface-compatible and are specified over a temperature range of −55°C to +125°C.

The TMP100 and TMP101 require no external components for operation except for pull-up resistors on SCL, SDA, and ALERT, although a 0.1µF bypass capacitor is recommended, as shown in Figure 1 and Figure 2.



NOTE: (1) SCL, SDA and ALERT require pull−up resistors for I^{2}C bus applications.

Figure 1. Typical Connections of the TMP101



NOTE: (1) SCL and SDA require pull−up resistors for I^{2}C bus applications.

Figure 2. Typical Connections of the TMP100

The die flag of the lead frame is connected to pin 2. The sensing device of the TMP100 and TMP101 is the chip itself. Thermal paths run through the package leads as well as the plastic package. The lower thermal resistance of metal causes the leads to provide the primary thermal path. The GND pin of the TMP100 or TMP101 is directly connected to the metal lead frame, and is the best choice for thermal input.

To maintain the accuracy in applications requiring air or surface temperature measurement, care should be taken to isolate the package and leads from ambient air temperature. A thermally-conductive adhesive will assist in achieving accurate surface temperature measurement.

POINTER REGISTER

Figure 3 shows the internal register structure of the TMP100 and TMP101. The 8-bit Pointer Register of the TMP100 and TMP101 is used to address a given data register. The Pointer Register uses the two LSBs to identify which of the data registers should respond to a read or write command. Table 1 identifies the bits of the Pointer Register byte. Table 2 describes the pointer address of the registers available in the TMP100 and TMP101. Power-up Reset value of P1/P0 is 00.

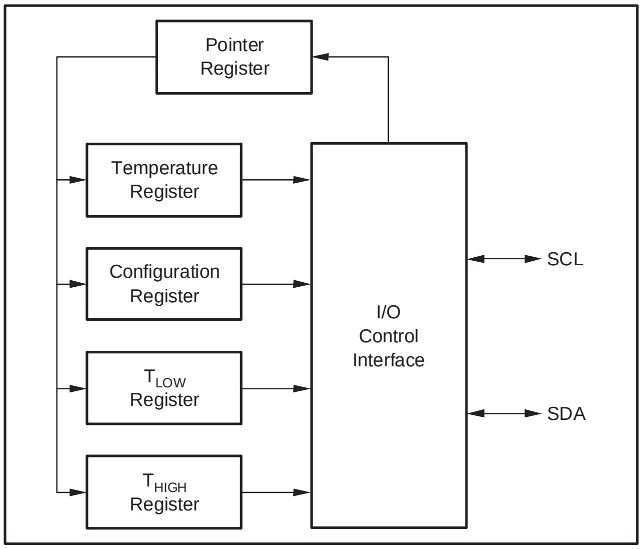


Figure 3. Internal Register Structure of the TMP100 and TMP101

Table 1. Pointer Register Type

P7	P6	P5	P4	P3	P2	P1	P0
0	0	0	0	0	0	Register Bits	

Table 2. Pointer Addresses of the TMP100 and TMP101 Registers

P1	P0	REGISTER
0	0	Temperature Register (READ Only)
0	1	Configuration Register (READ/WRITE)
1	0	T_{LOW} Register (READ/WRITE)
1	1	T_{HIGH} Register (READ/WRITE)

TEMPERATURE REGISTER

The Temperature Register of the TMP100 or TMP101 is a 12-bit read-only register that stores the output of the most recent conversion. Two bytes must be read to obtain data and are described in Table 3 and Table 4. The first 12 bits are used to indicate temperature with all remaining bits equal to zero. Data format for temperature is summarized in Table 5. Following power-up or reset, the Temperature Register will read 0°C until the first conversion is complete.

Table 3. Byte 1 of Temperature Register

D7	D6	D5	D4	D3	D2	D1	D0
T11	T10	T9	T8	T7	T6	T5	T4

Table 4. Byte 2 of Temperature Register

D7	D6	D5	D4	D3	D2	D1	D0
T3	T2	T1	T0	0	0	0	0

Table 5. Temperature Data Format

TEMPERATURE (°C)	DIGITAL OUTPUT (BINARY)	HEX
128	0111 1111 1111	7FF
127.9375	0111 1111 1111	7FF
100	0110 0100 0000	640
80	0101 0000 0000	500
75	0100 1011 0000	4B0
50	0011 0010 0000	320
25	0001 1001 0000	190
0.25	0000 0000 0100	004
0.0	0000 0000 0000	000
−0.25	1111 1111 1100	FFC
−25	1110 0111 0000	E70
−55	1100 1001 0000	C90
−128	1000 0000 0000	800

The user can obtain 9, 10, 11, or 12 bits of resolution by addressing the Configuration Register and setting the resolution bits accordingly. For 9-, 10-, or 11-bit resolution, the most significant bits in the Temperature Register are used with the unused LSBs set to zero.

CONFIGURATION REGISTER

The Configuration Register is an 8-bit read/write register used to store bits that control the operational modes of the temperature sensor. Read/write operations are performed MSB first. The format of the Configuration Register for the TMP100 and TMP101 is shown in Table 6, followed by a breakdown of the register bits. The power-up/reset value of the Configuration Register is all bits equal to 0. The OS/ALERT bit will read as 1 after power-up/reset.

Table 6. Configuration Register Format

BYTE	D7	D6	D5	D4	D3	D2	D1	D0
1	OS/ALERT	R1	R0	F1	F0	POL	TM	SD

SHUTDOWN MODE (SD)

The Shutdown Mode of the TMP100 and TMP101 allows the user to save maximum power by shutting down all device circuitry other than the serial interface, which reduces current consumption to less than 1µA. For the TMP100 and TMP101, Shutdown Mode is enabled when the SD bit is 1. The device will shutdown once the current conversion is completed. For SD equal to 0, the device will maintain continuous conversion.

THERMOSTAT MODE (TM)

The Thermostat Mode bit of the TMP101 indicates to the device whether to operate in Comparator Mode (TM = 0) or Interrupt Mode (TM = 1). For more information on comparator and interrupt modes, see the *HIGH and LOW Limit Registers* section.

POLARITY (POL)

The Polarity Bit of the TMP101 allows the user to adjust the polarity of the ALERT pin output. If POL = 0, the ALERT pin will be active LOW, as shown in Figure 4. For POL = 1 the ALERT pin will be active HIGH, and the state of the ALERT pin is inverted.

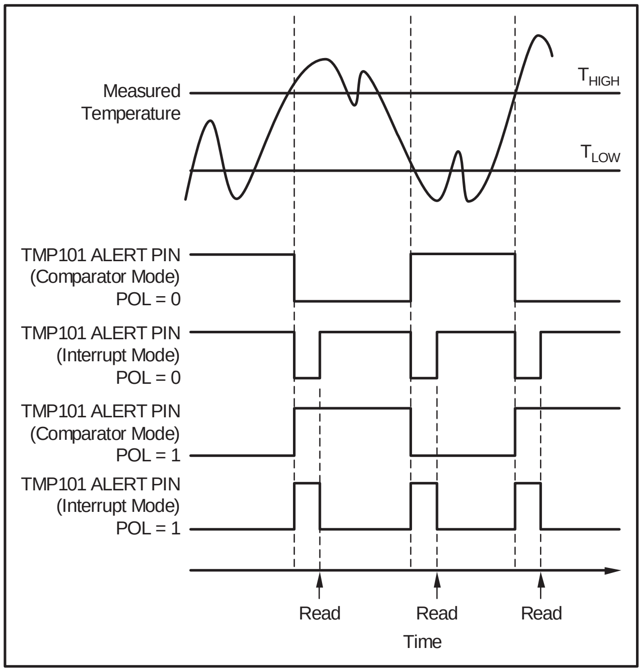

Measured Temperature
T_{HIGH}
T_{LOW}
TMP101 ALERT PIN (Comparator Mode) POL = 0
TMP101 ALERT PIN (Interrupt Mode) POL = 0
TMP101 ALERT PIN (Comparator Mode) POL = 1
TMP101 ALERT PIN (Interrupt Mode) POL = 1
Read
Read
Read
Time

Figure 4. Output Transfer Function Diagrams

FAULT QUEUE (F1/F0)

A fault condition occurs when the measured temperature exceeds the user-defined limits set in the T_{HIGH} and T_{LOW} Registers. Additionally, the number of fault conditions required to generate an alert may be programmed using the Fault Queue. The Fault Queue is provided to prevent a false alert due to environmental noise. The Fault Queue requires consecutive fault measurements in order to trigger the alert function. If the temperature falls below T_{LOW}, prior to reaching the number of programmed consecutive faults limit, the count is reset to 0. Table 7 defines the number of measured faults that may be programmed to trigger an alert condition in the device.

Table 7. Fault Settings of the TMP100 and TMP101

F1	F0	CONSECUTIVE FAULTS
0	0	1
0	1	2
1	0	4
1	1	6

CONVERTER RESOLUTION (R1/R0)

The Converter Resolution Bits control the resolution of the internal Analog-to-Digital (A/D) converter. This allows the user to maximize efficiency by programming for higher resolution or faster conversion time. Table 8 identifies the Resolution Bits and relationship between resolution and conversion time.

Table 8. Resolution of the TMP100 and TMP101

R1	R0	RESOLUTION	CONVERSION TIME (typical)
0	0	9 Bits (0.5°C)	40ms
0	1	10 Bits (0.25°C)	80ms
1	0	11 Bits (0.125°C)	160ms
1	1	12 Bits (0.0625°C)	320ms

OS/ALERT (OS)

The TMP100 and TMP101 feature a One-Shot Temperature Measurement Mode. When the device is in Shutdown Mode, writing a 1 to the OS/ALERT bit will start a single temperature conversion. The device will return to the shutdown state at the completion of the single conversion. This is useful to reduce power consumption in the TMP100 and TMP101 when continuous monitoring of temperature is not required.

Reading the OS/ALERT bit will provide information about the Comparator Mode status. The state of the POL bit will invert the polarity of data returned from the OS/ALERT bit. For POL = 0, the OS/ALERT will read as 1 until the temperature equals or exceeds T_{HIGH} for the programmed number of consecutive faults, causing the OS/ALERT bit to read as 0. The OS/ALERT bit will continue to read as 0 until the temperature falls below T_{LOW} for the programmed number of consecutive faults when it will again read as 1. The status of the TM bit does not affect the status of the OS/ALERT bit.

HIGH AND LOW LIMIT REGISTERS

In Comparator Mode (TM = 0), the ALERT pin of the TMP101 becomes active when the temperature equals or exceeds the value in T_{HIGH} and generates a consecutive number of faults according to fault bits F1 and F0. The ALERT pin will remain active until the temperature falls below the indicated T_{LOW} value for the same number of faults.

In Interrupt Mode (TM = 1) the ALERT Pin becomes active when the temperature equals or exceeds T_{HIGH} for a consecutive number of fault conditions. The ALERT pin remains active until a read operation of any register occurs or the device successfully responds to the SMBus Alert Response Address. The ALERT pin will also be cleared if the device is placed in Shutdown Mode. Once the ALERT pin is cleared, it will only become active again by the temperature falling below T_{LOW}. When the temperature falls below T_{LOW}, the ALERT pin will become active and remain active until cleared by a read operation of any register or a successful response to the SMBus Alert Response Address. Once the ALERT pin is cleared, the above cycle will repeat with the ALERT pin becoming active when the temperature equals or exceeds T_{HIGH}. The ALERT pin can also be cleared by resetting the device with the General Call Reset command. This will also clear the state of the internal registers in the device returning the device to Comparator Mode (TM = 0).

Both operational modes are represented in Figure 4. Table 9 and Table 10 describe the format for the T_{HIGH} and T_{LOW} registers. Power-up Reset values for T_{HIGH} and T_{LOW} are: T_{HIGH} = 80°C and T_{LOW} = 75°C. The format of the data for T_{HIGH} and T_{LOW} is the same as for the Temperature Register.

Table 9. Bytes 1 and 2 of T_{HIGH} Register

BYTE	D7	D6	D5	D4	D3	D2	D1	D0
1	H11	H10	H9	H8	H7	H6	H5	H4

BYTE	D7	D6	D5	D4	D3	D2	D1	D0
2	H3	H2	H1	H0	0	0	0	0

Table 10. Bytes 1 and 2 of T_{LOW} Register

BYTE	D7	D6	D5	D4	D3	D2	D1	D0
1	L11	L10	L9	L8	L7	L6	L5	L4

BYTE	D7	D6	D5	D4	D3	D2	D1	D0
2	L3	L2	L1	L0	0	0	0	0

All 12 bits for the Temperature, T_{HIGH}, and T_{LOW} registers are used in the comparisons for the ALERT function for all converter resolutions. The three LSBs in T_{HIGH} and T_{LOW} can affect the ALERT output even if the converter is configured for 9-bit resolution.

SERIAL INTERFACE

The TMP100 and TMP101 operate only as slave devices on the I^2C bus and SMBus. Connections to the bus are made via the open-drain I/O lines SDA and SCL. The TMP100 and TMP101 support the transmission protocol for fast (up to 400kHz) and high-speed (up to 3.4MHz) modes. All data bytes are transmitted most significant bit first.

SERIAL BUS ADDRESS

To program the TMP100 and TMP101, the master must first address slave devices via a slave address byte. The slave address byte consists of seven address bits, and a direction bit indicating the intent of executing a read or write operation.

The TMP100 features two address pins to allow up to eight devices to be addressed on a single I^{2}C interface. Table 11 describes the pin logic levels used to properly connect up to eight devices. *Float* indicates the pin is left unconnected. The state of pins ADD0 and ADD1 is sampled on the first I^{2}C bus communication and should be set prior to any activity on the interface.

Table 11. Address Pins and Slave Addresses for the TMP100

ADD1	ADD0	SLAVE ADDRESS
0	0	1001000
0	Float	1001001
0	1	1001010
1	0	1001100
1	Float	1001101
1	1	1001110
Float	0	1001011
Float	1	1001111

The TMP101 features one address pin and an ALERT pin, allowing up to three devices to be connected per bus. Pin logic levels are described in Table 12. The address pins of the TMP100 and TMP101 are read after reset or in response to an I^{2}C address acquire request. Following reading, the state of the address pins is latched to minimize power dissipation associated with detection.

Table 12. Address Pins and Slave Addresses for the TMP101

ADD0	SLAVE ADDRESS
0	1001000
Float	1001001
1	1001010

BUS OVERVIEW

The device that initiates the transfer is called a *master*, and the devices controlled by the master are *slaves*. The bus must be controlled by a master device that generates the serial clock (SCL), controls the bus access, and generates the START and STOP conditions.

To address a specific device, a START condition is initiated, indicated by pulling the data-line (SDA) from a HIGH to LOW logic level while SCL is HIGH. All slaves on the bus shift in the slave address byte, with the last bit indicating whether a read or write operation is intended. During the ninth clock pulse, the slave being addressed responds to the master by generating an Acknowledge and pulling SDA LOW.

Data transfer is then initiated and sent over eight clock pulses followed by an Acknowledge Bit. During data transfer SDA must remain stable while SCL is HIGH, as any change in SDA while SCL is HIGH will be interpreted as a control signal.

Once all data have been transferred, the master generates a STOP condition indicated by pulling SDA from LOW to HIGH, while SCL is HIGH.

WRITING/READING TO THE TMP100 AND TMP101

Accessing a particular register on the TMP100 and TMP101 is accomplished by writing the appropriate value to the Pointer Register. The value for the Pointer Register is the first byte transferred after the I^{2}C slave address byte with the $R/\overline{W}$ bit LOW. Every write operation to the TMP100 and TMP101 requires a value for the Pointer Register. (Refer to Figure 6.)

When reading from the TMP100 and TMP101, the last value stored in the Pointer Register by a write operation is used to determine which register is read by a read operation. To change the register pointer for a read operation, a new value must be written to the Pointer Register. This is accomplished by issuing an I^{2}C slave address byte with the $R/\overline{W}$ bit LOW, followed by the Pointer Register Byte. No additional data are required. The master can then generate a START condition and send the I^{2}C slave address byte with the $R/\overline{W}$ bit HIGH to initiate the read command. See Figure 7 for details of this sequence. If repeated reads from the same register are desired, it is not necessary to continually send the Pointer Register bytes as the TMP100 and TMP101 will remember the Pointer Register value until it is changed by the next write operation.

SLAVE MODE OPERATIONS

The TMP100 and TMP101 can operate as slave receivers or slave transmitters.

Slave Receiver Mode:

The first byte transmitted by the master is the slave address, with the $R/\overline{W}$ bit LOW. The TMP100 or TMP101 then acknowledges reception of a valid address. The next byte transmitted by the master is the Pointer Register. The TMP100 or TMP101 then acknowledges reception of the Pointer Register byte. The next byte or bytes are written to the register addressed by the Pointer Register. The TMP100 and TMP101 will acknowledge reception of each data byte. The master may terminate data transfer by generating a START or STOP condition.

Slave Transmitter Mode:

The first byte is transmitted by the master and is the slave address, with the $R/\overline{W}$ bit HIGH. The slave acknowledges reception of a valid slave address. The next byte is transmitted by the slave and is the most significant byte of the register indicated by the Pointer Register. The master acknowledges reception of the data byte. The next byte transmitted by the slave is the least significant byte. The master acknowledges reception of the data byte. The master may terminate data transfer by generating a Not-Acknowledge on reception of any data byte, or generating a START or STOP condition.

SMBus ALERT FUNCTION

The TMP101 supports the SMBus Alert function. When the TMP101 is operating in Interrupt Mode (TM = 1), the ALERT pin of the TMP101 may be connected as an SMBus Alert signal. When a master senses that an ALERT condition is present on the ALERT line, the master sends an SMBus Alert command (00011001) on the bus. If the ALERT pin of the TMP101 is active, the TMP101 will acknowledge the SMBus Alert command and respond by returning its slave address on the SDA line. The eighth bit (LSB) of the slave address byte will indicate if the temperature exceeding T_{HIGH} or falling below T_{LOW} caused the ALERT condition. For POL = 0, this bit will be LOW if the temperature is greater than or equal to T_{HIGH}. This bit will be HIGH if the temperature is less than T_{LOW}. The polarity of this bit will be inverted if POL = 1. Refer to Figure 8 for details of this sequence.

If multiple devices on the bus respond to the SMBus Alert command, arbitration during the slave address portion of the SMBus alert command will determine which device will clear its ALERT status. If the TMP101 wins the arbitration, its ALERT pin will become inactive at the completion of the SMBus Alert command. If the TMP101 loses the arbitration, its ALERT pin will remain active.

The TMP100 will also respond to the SMBus ALERT command if its TM bit is set to 1. Since it does not have an ALERT pin, the master needs to periodically poll the device by issuing an SMBus Alert command. If the TMP100 has generated an ALERT, it will acknowledge the SMBus Alert command and return its slave address in the next byte.

GENERAL CALL

The TMP100 and TMP101 respond to the I^2C General Call address (0000000) if the eighth bit is 0. The device will acknowledge the General Call address and respond to commands in the second byte. If the second byte is 00000100, the TMP100 and TMP101 will latch the status of their address pins, but will not reset. If the second byte is 00000110, the TMP100 and TMP101 will latch the status of their address pins and reset their internal registers.

POR (POWER-ON RESET)

The TMP100 and TMP101 both have on-chip power-on reset circuits that reset the device to default settings when the device is powered on. This circuit activates when the power supply is less than 0.3V for more than 100ms. If the TMP100 and TMP101 are powered down by removing supply voltage from the device, but the supply voltage is not assured to be less than 0.3V, it is recommended to issue a General Call reset command on the I^2C interface bus to ensure that the TMP100 and TMP101 are completely reset.

HIGH-SPEED MODE

In order for the I^2C bus to operate at frequencies above 400kHz, the master device must issue an Hs-mode master code (00001XXX) as the first byte after a START condition to switch the bus to high-speed operation. The TMP100 and TMP101 will not acknowledge this byte as required by the I^2C specification, but will switch their input filters on SDA and SCL and their output filters on SDA to operate in Hs-mode, allowing transfers at up to 3.4MHz. After the Hs-mode master code has been issued, the master will transmit an I^2C slave address to initiate a data transfer operation. The bus will continue to operate in Hs-mode until a STOP condition occurs on the bus. Upon receiving the STOP condition, the TMP100 and TMP101 will switch their input and output filters back to fast-mode operation.

TIMING DIAGRAMS

The TMP100 and TMP101 are I^{2}C and SMBus compatible. Figure 5 to Figure 8 describe the various operations on the TMP100 and TMP101. Bus definitions are given below. Parameters for Figure 5 are defined in Table 13.

Bus Idle: Both SDA and SCL lines remain HIGH.

Start Data Transfer: A change in the state of the SDA line, from HIGH to LOW, while the SCL line is HIGH, defines a START condition. Each data transfer is initiated with a START condition.

Stop Data Transfer: A change in the state of the SDA line from LOW to HIGH while the SCL line is HIGH defines a STOP condition. Each data transfer is terminated with a repeated START or STOP condition.

Data Transfer: The number of data bytes transferred between a START and a STOP condition is not limited and is determined by the master device. The receiver acknowledges the transfer of data.

Acknowledge: Each receiving device, when addressed, is obliged to generate an Acknowledge bit. A device that acknowledges must pull down the SDA line during the Acknowledge clock pulse in such a way that the SDA line is stable LOW during the HIGH period of the Acknowledge clock pulse. Setup and hold times must be taken into account. On a master receive, the termination of the data transfer can be signaled by the master generating a Not-Acknowledge on the last byte that has been transmitted by the slave.

Table 13. Timing Diagram Definitions

PARAMETER		FAST MODE		HIGH-SPEED MODE		UNITS
		MIN	MAX	MIN	MAX	
SCLK Operating Frequency	$f_{(SCLK)}$		0.4		3.4	MHz
Bus Free TIme Between STOP and START Conditions	$t_{(BUF)}$	600		160		ns
Hold time after repeated START condition. After this period, the first clock is generated.	$t_{(HDSTA)}$	600		160		ns
Repeated START Condition Setup Time	$t_{(SUSTA)}$	600		160		ns
STOP Condition Setup Time	$t_{(SUSTO)}$	600		160		ns
Data HOLD Time	$t_{(HDDAT)}$	0		0		ns
Data Setup Time	$t_{(SUDAT)}$	100		10		ns
SCLK Clock LOW Period	$t_{(LOW)}$	1300		160		ns
SCLK Clock HIGH Period	$t_{(HIGH)}$	600		60		ns
Clock/Data Fall Time	t_F		300		160	ns
Clock/Data Rise Time	t_R		300		160	ns
for SCLK ≤ 100kHz	t_R		1000			ns

I2C TIMING DIAGRAMS

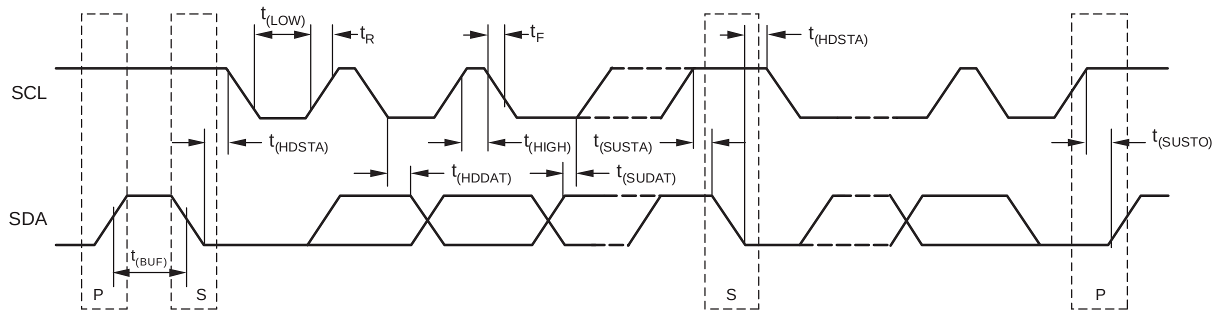


Figure 5. I2C Timing Diagram

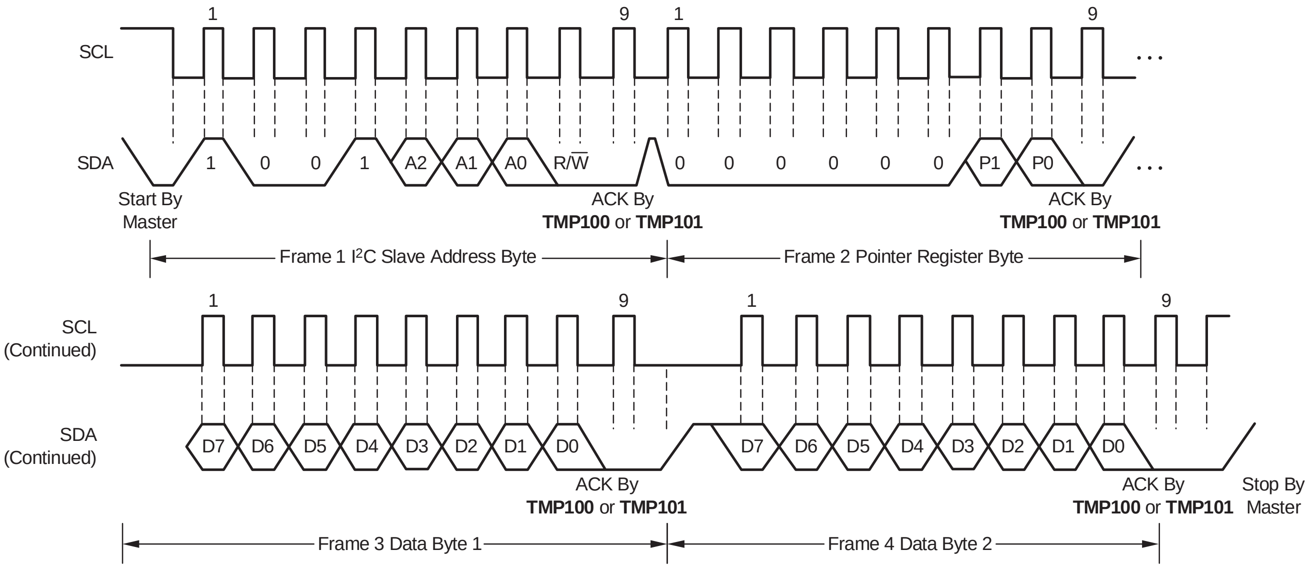


Figure 6. I2C Timing Diagram for Write Word Format

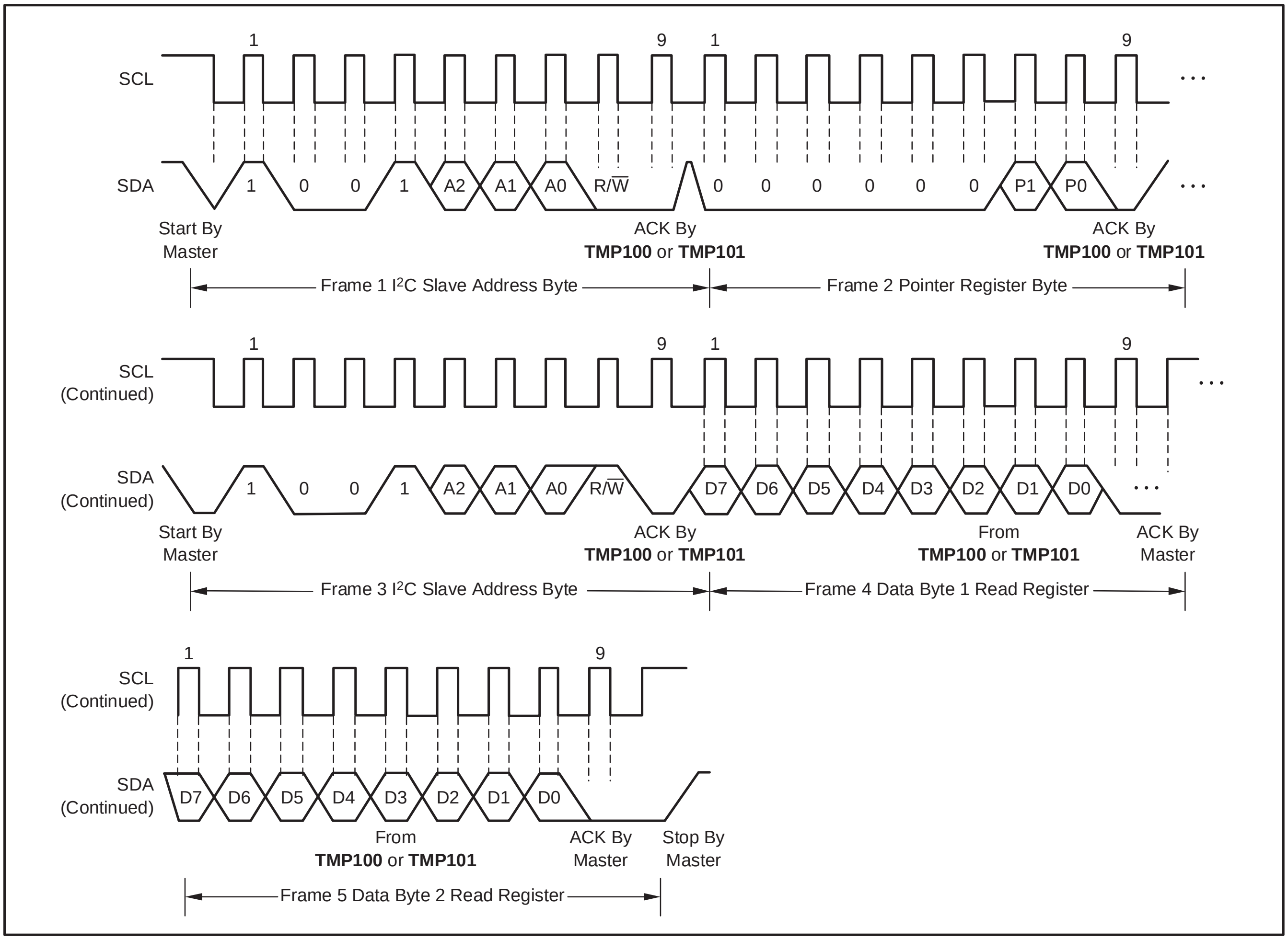


Figure 7. I^{2}C Timing Diagram for Read Word Format

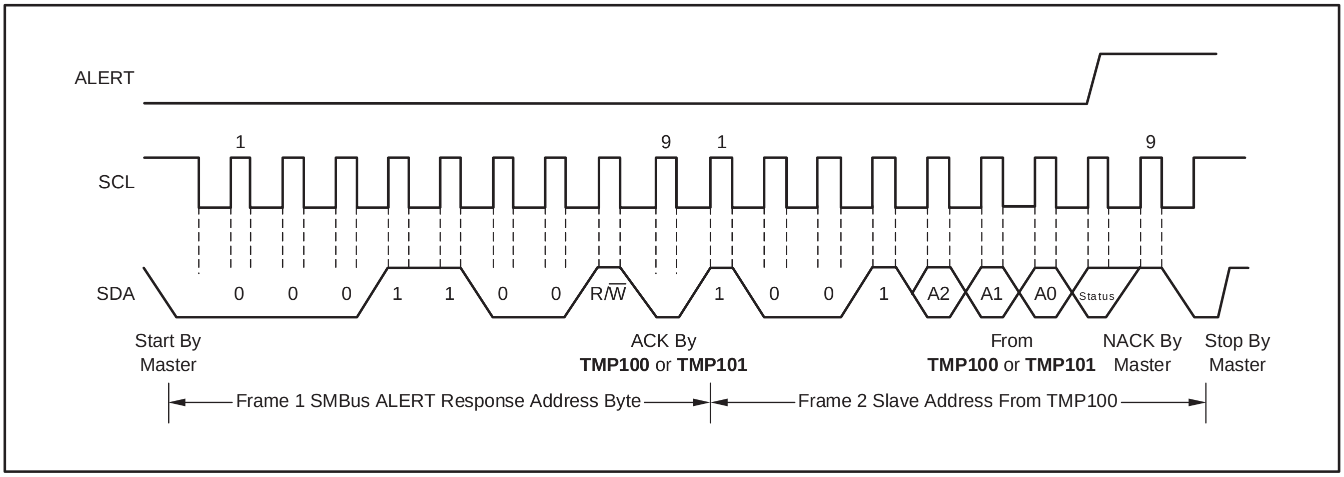


Figure 8. Timing Diagram for SMBus ALERT

PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/Ball Finish	MSL Peak Temp (3)	Op Temp (°C)	Top-Side Markings (4)	Samples
SN0312100DBVR	ACTIVE	SOT-23	DBV	6	3000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-55 to 125	T100	Samples
TMP100NA/250	ACTIVE	SOT-23	DBV	6	250	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR		T100	Samples
TMP100NA/250G4	ACTIVE	SOT-23	DBV	6	250	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR		T100	Samples
TMP100NA/3K	ACTIVE	SOT-23	DBV	6	3000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-55 to 125	T100	Samples
TMP100NA/3KG4	ACTIVE	SOT-23	DBV	6	3000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-55 to 125	T100	Samples
TMP101NA/250	ACTIVE	SOT-23	DBV	6	250	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-55 to 125	T101	Samples
TMP101NA/250G4	ACTIVE	SOT-23	DBV	6	250	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-55 to 125	T101	Samples
TMP101NA/3K	ACTIVE	SOT-23	DBV	6	3000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-55 to 125	T101	Samples
TMP101NA/3KG4	ACTIVE	SOT-23	DBV	6	3000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-55 to 125	T101	Samples

(1) The marketing status values are defined as follows:
ACTIVE: Product device recommended for new designs.
LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.
PREVIEW: Device has been announced but is not in production. Samples may or may not be available.
OBSOLETE: TI has discontinued the production of the device.

(2) Eco Plan - The planned eco-friendly classification: Pb-Free (RoHS), Pb-Free (RoHS Exempt), or Green (RoHS & no Sb/Br) - please check http://www.ti.com/productcontent for the latest availability information and additional product content details.
TBD: The Pb-Free/Green conversion plan has not been defined.
Pb-Free (RoHS): TI's terms "Lead-Free" or "Pb-Free" mean semiconductor products that are compatible with the current RoHS requirements for all 6 substances, including the requirement that lead not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, TI Pb-Free products are suitable for use in specified lead-free processes.
Pb-Free (RoHS Exempt): This component has a RoHS exemption for either 1) lead-based flip-chip solder bumps used between the die and package, or 2) lead-based die adhesive used between the die and leadframe. The component is otherwise considered Pb-Free (RoHS compatible) as defined above.
Green (RoHS & no Sb/Br): TI defines "Green" to mean Pb-Free (RoHS compatible), and free of Bromine (Br) and Antimony (Sb) based flame retardants (Br or Sb do not exceed 0.1% by weight in homogeneous material)

[3] MSL, Peak Temp. -- The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

[4] Multiple Top-Side Markings will be inside parentheses. Only one Top-Side Marking contained in parentheses and separated by a "~" will appear on a device. If a line is indented then it is a continuation of the previous line and the two combined represent the entire Top-Side Marking for that device.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

OTHER QUALIFIED VERSIONS OF TMP100, TMP101 :

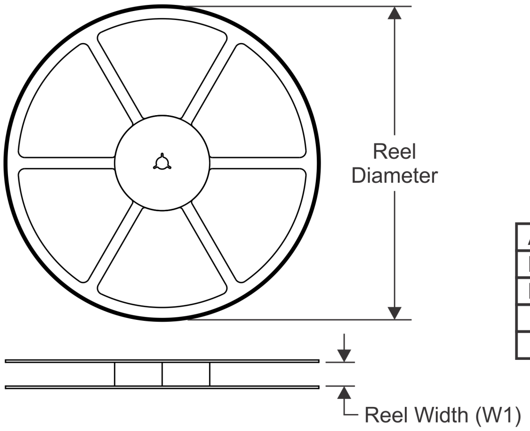
- Automotive: TMP101-Q1
- Enhanced Product: TMP100-EP

NOTE: Qualified Version Definitions:

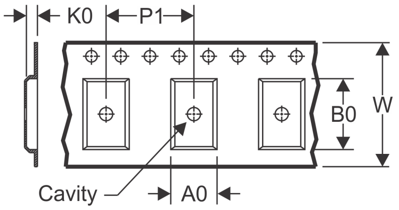
- Automotive - Q100 devices qualified for high-reliability automotive applications targeting zero defects
- Enhanced Product - Supports Defense, Aerospace and Medical Applications

TAPE AND REEL INFORMATION

REEL DIMENSIONS

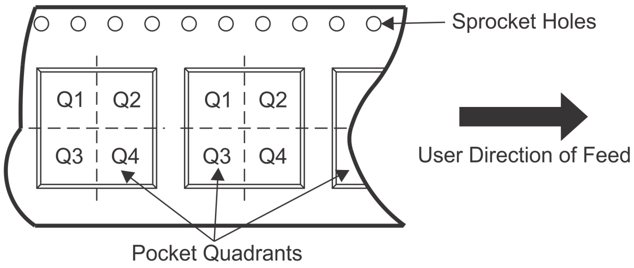


TAPE DIMENSIONS



A0	Dimension designed to accommodate the component width
B0	Dimension designed to accommodate the component length
K0	Dimension designed to accommodate the component thickness
W	Overall width of the carrier tape
P1	Pitch between successive cavity centers

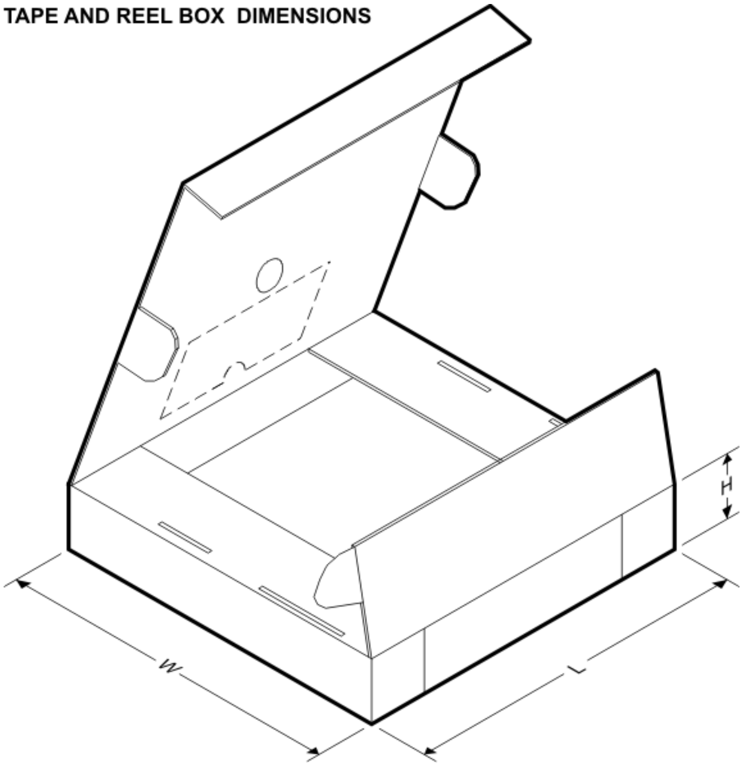
QUADRANT ASSIGNMENTS FOR PIN 1 ORIENTATION IN TAPE



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Reel Diameter (mm)	Reel Width W1 (mm)	A0 (mm)	B0 (mm)	K0 (mm)	P1 (mm)	W (mm)	Pin1 Quadrant
TMP100NA/250	SOT-23	DBV	6	250	178.0	9.0	3.23	3.17	1.37	4.0	8.0	Q3
TMP100NA/3K	SOT-23	DBV	6	3000	178.0	9.0	3.23	3.17	1.37	4.0	8.0	Q3

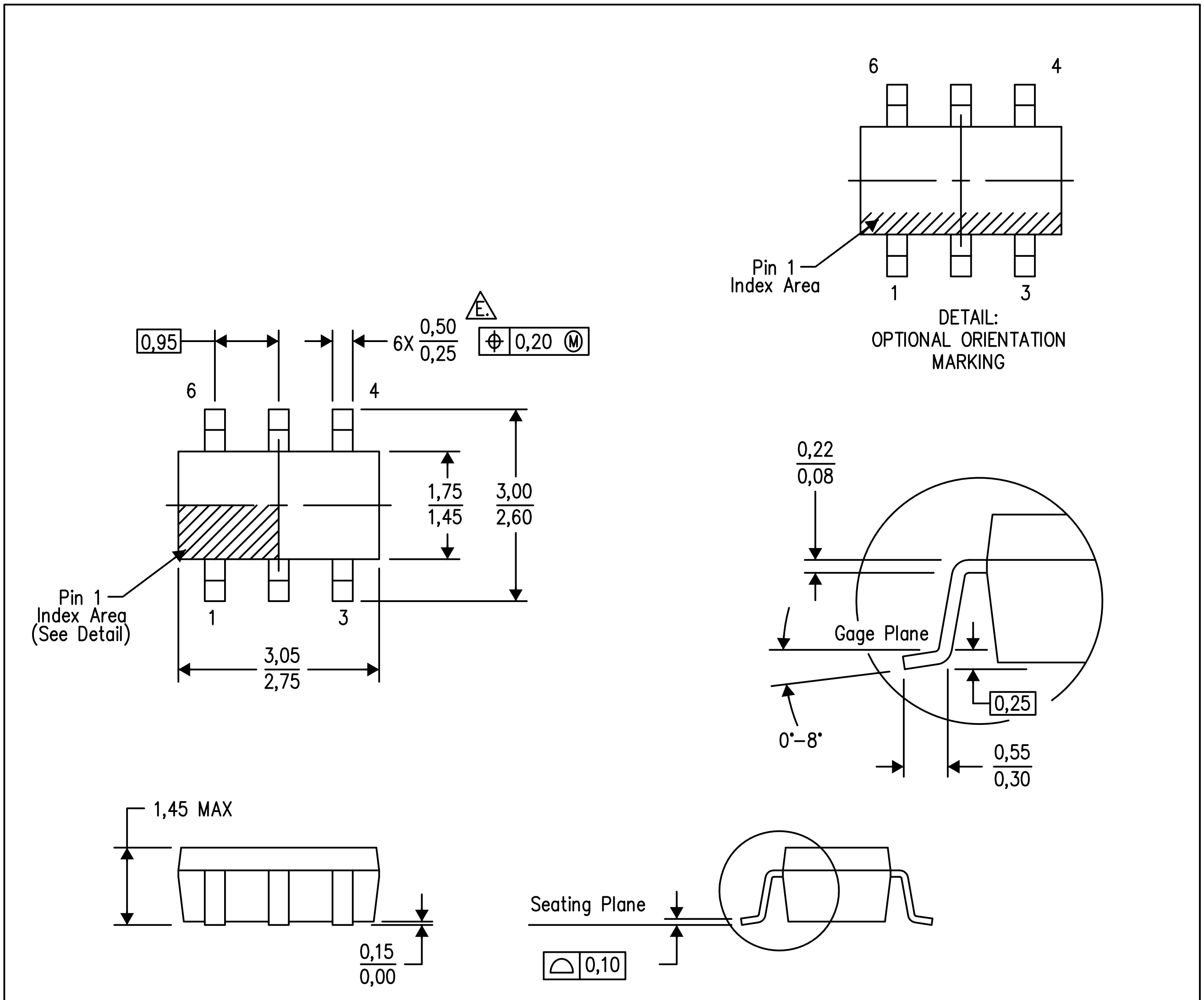
TAPE AND REEL BOX DIMENSIONS



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Length (mm)	Width (mm)	Height (mm)
TMP100NA/250	SOT-23	DBV	6	250	180.0	180.0	18.0
TMP100NA/3K	SOT-23	DBV	6	3000	180.0	180.0	18.0

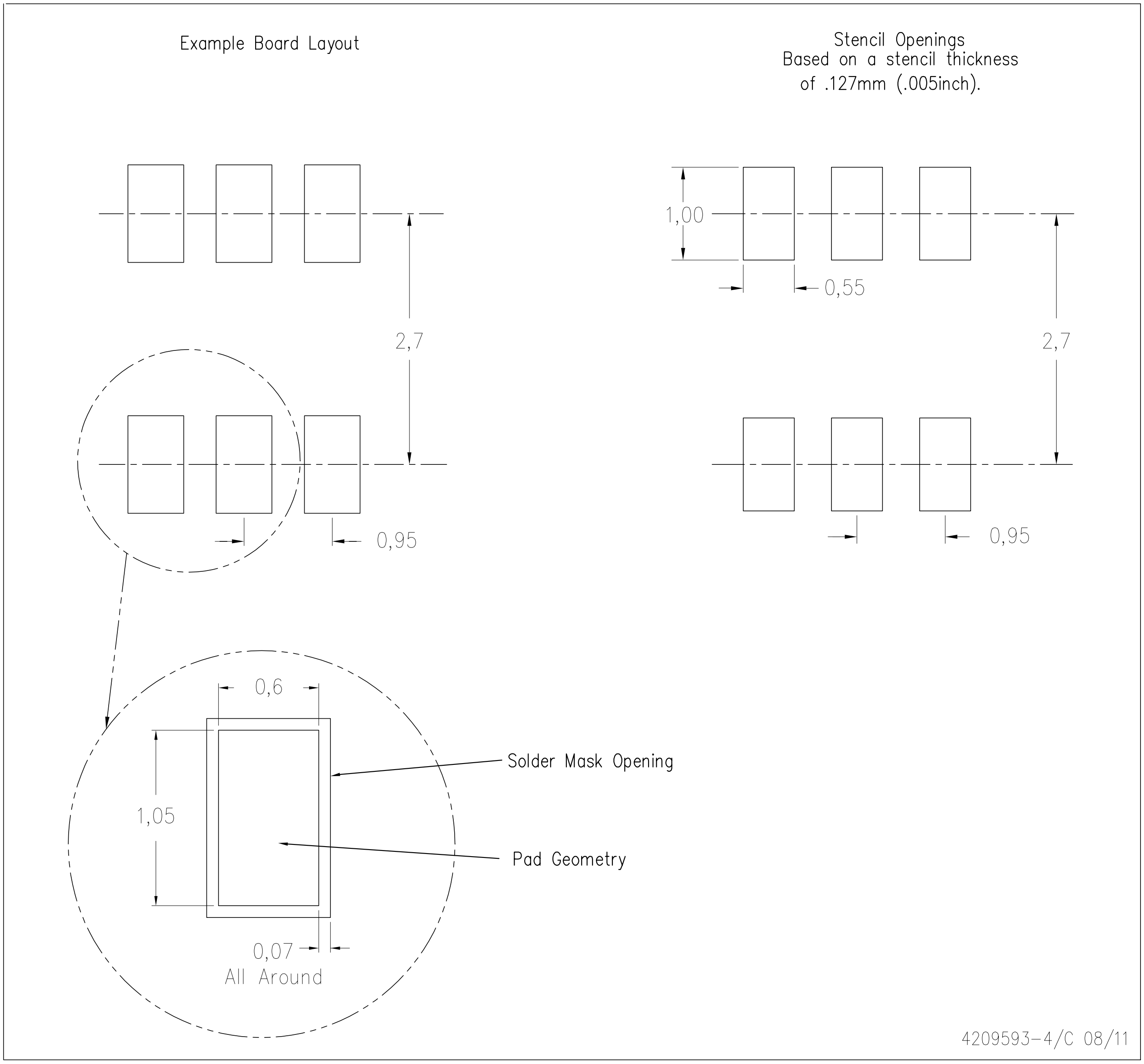
DBV (R–PDSO–G6) PLASTIC SMALL–OUTLINE PACKAGE



4073253–5/L 08/2013

NOTES:
A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Body dimensions do not include mold flash or protrusion. Mold flash and protrusion shall not exceed 0.15 per side.
D. Leads 1,2,3 may be wider than leads 4,5,6 for package orientation.
E. Falls within JEDEC MO–178 Variation AB, except minimum lead width.

DBV (R–PDSO–G6) PLASTIC SMALL OUTLINE



NOTES:
A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Customers should place a note on the circuit board fabrication drawing not to alter the center solder mask defined pad.
D. Publication IPC–7351 is recommended for alternate designs.
E. Laser cutting apertures with trapezoidal walls and also rounding corners will offer better paste release. Customers should contact their board assembly site for stencil design recommendations. Example stencil design based on a 50% volumetric metal load solder paste. Refer to IPC–7525 for other stencil recommendations.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

Freescale Semiconductor
Data Sheet: Technical Data

Document Number: MPL115A2
Rev 8, 06/2012

RoHS

Miniature I^{2}C Digital Barometer

The MPL115A2 is an absolute pressure sensor with a digital I^{2}C output targeting low cost applications. A miniature 5 by 3 by 1.2 mm LGA package is ideally suited for the space constrained requirements of portable electronic devices. Low current consumptions of 5 µA during Active mode and 1 µA during Shutdown (Sleep) mode are essential when focusing on low-power applications. The wide operating temperature range spans from -40°C to +105°C to fit demanding environmental conditions.

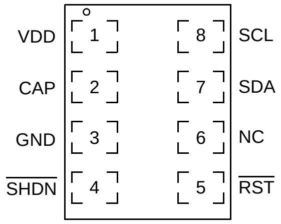
The MPL115A2 employs a MEMS pressure sensor with a conditioning IC to provide accurate pressure measurements from 50 to 115 kPa. An integrated ADC converts pressure and temperature sensor readings to digitized outputs via a I^{2}C port. Factory calibration data is stored internally in an on-board ROM. Utilizing the raw sensor output and calibration data, the host microcontroller executes a compensation algorithm to render *Compensated Absolute Pressure* with ±1 kPa accuracy.

The MPL115A2 pressure sensor's small form factor, low power capability, precision, and digital output optimize it for barometric measurement applications.

MPL115A2
50 to 115 kPa

MPL115A2
5.0 mm by 3.0 mm by 1.2 mm

Top View



Pin Connections

Features

- Digitized pressure and temperature information together with programmed calibration coefficients for host micro use.
- Factory Calibrated
- 50 kPa to 115 kPa Absolute Pressure
- ±1 kPa Accuracy
- 2.375V to 5.5V Supply
- Integrated ADC
- I^{2}C Interface (operates up to 400 kHz)
- 7 bit I^{2}C address = 0x60
- Monotonic Pressure and Temperature Data Outputs
- Surface Mount RoHS Compliant Package

Application Examples

- Barometry (portable and desktop)
- Altimeters
- Weather Stations
- Hard Disk-Drives (HDD)
- Industrial Equipment
- Health Monitoring
- Air Control Systems

ORDERING INFORMATION

Device Name	Package Options	Case No.	# of Ports			Pressure Type			Digital Interface
			None	Single	Dual	Gauge	Differential	Absolute	
MPL115A2	Tray	2015	•					•	I^{2}C
MPL115A2T1	Tape & Reel (1000)	2015	•					•	I^{2}C
MPL115A2T2	Tape & Reel (5000)	2015	•					•	I^{2}C

1 Block Diagram and Pin Descriptions

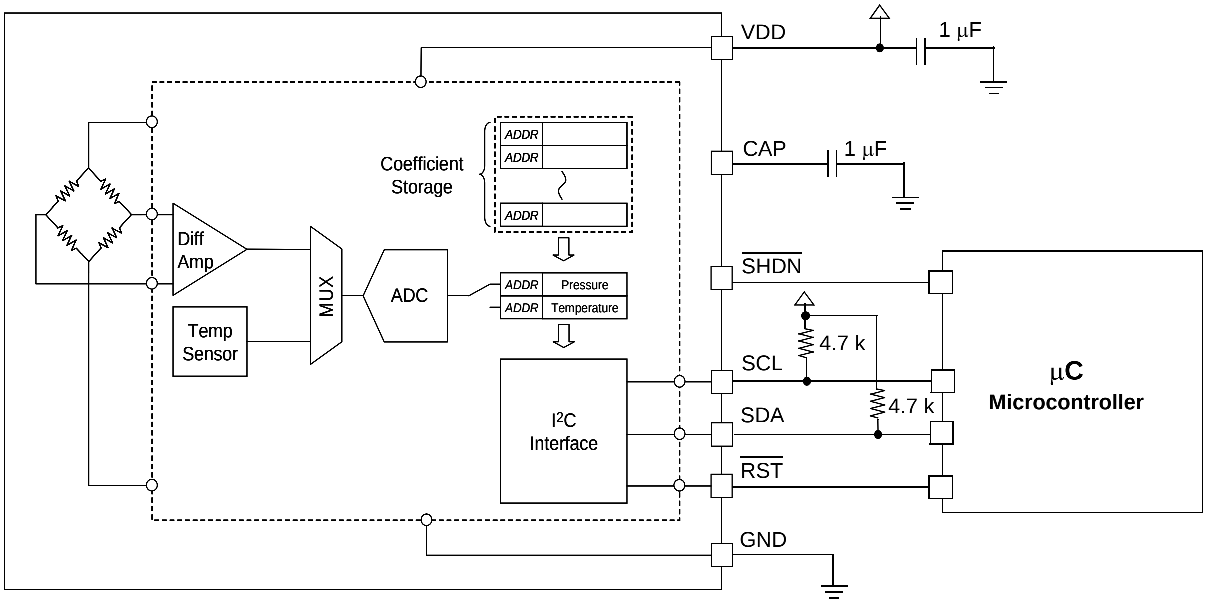


Figure 1. Block Diagram and Pin Connections

Table 1. Pin Description

Pin	Name	Function
1	VDD	VDD Power Supply Connection: VDD range is 2.375V to 5.5V.
2	CAP	1 µF connected to ground.
3	GND	Ground
4	$\overline{\text{SHDN}}$	Shutdown: Connect to GND to disable the device. When in shutdown, the part draws no more than 1 µA supply current and all communications pins ($\overline{\text{RST}}$, SCL, SDA) are high impedance. Connect to VDD for normal operation.
5	$\overline{\text{RST}}$	Reset: Connect to ground to disable I^{2}C communications.
6	NC	NC: No connection
7	SDA[1]	SDA: Serial data I/O line
8	SCL[1]	I^{2}C Serial Clock Input.

[1] Use 4.7k pullup resistors for I^{2}C communication.

2 Mechanical and Electrical Specifications

2.1 Maximum Ratings

Voltage (with respect to GND unless otherwise noted)

V_{DD}	-0.3 V to +5.5 V
$\overline{\text{SHDN}}$, RST, SDA, SCL	-0.3 V to V_{DD}+0.3 V
Operating Temperature Range	-40°C to +105°C
Storage Temperature Range	-40°C to +125°C
Overpressure	1000 kPa

2.2 Operating Characteristics

V_{DD} = 2.375 V to 5.5 V, T_A = -40°C to +105°C, unless otherwise noted. Typical values are at V_{DD} = 3.3 V, T_A = +25°C.

Ref	Parameters	Symbol	Conditions	Min	Typ	Max	Units
1	Operating Supply Voltage	V_{DD}		2.375	3.3	5.5	V
2	Supply Current	I_{DD}	Shutdown ($\overline{\text{SHDN}}$ = GND)	—	—	1	μA
			Standby	—	3.5	10	μA
			Average – at one measurement per second	—	5	6	μA
Pressure Sensor							
3	Range			50	—	115	kPa
4	Resolution			—	0.15	—	kPa
5	Accuracy		-20ºC to 85ºC	—	—	±1	kPa
6	Power Supply Rejection		Typical operating circuit at DC		0.1	—	kPa/V
			100 mV p-p 217 Hz square wave plus 100 mV pseudo random noise with 10 MHz bandwidth		0.1	—	kPa
7	Conversion Time *(Start Pressure and Temperature Conversion*)	tc	Time between start convert command and data available in the Pressure and Temperature registers	—	1.6	3	ms
8	Wakeup Time	tw	Time between leaving Shutdown mode ($\overline{\text{SHDN}}$ goes high) and communicating with the device to issue a command or read data.	—	3	5	ms
I^2C I/O Stages: SCL, SDA							
9	SCL Clock Frequency	f_{SCL}		—	—	400	kHz
10	Low Level Input Voltage	VIL		—	—	$0.3V_{DD}$	V
11	High Level Input Voltage	VIH		$0.7V_{DD}$	—	—	V
I^2C Outputs: SDA							
12	Data Setup Time	t_{SU}	Setup time from command receipt to ready to transmit	0	—	0.4	s
I^2C Addressing							
MPL115A2 uses 7-bit addressing, does not acknowledge the general call address 0000000. Slave address has been set to 0x60 or 1100000.							

3 Overview of Functions/Operation

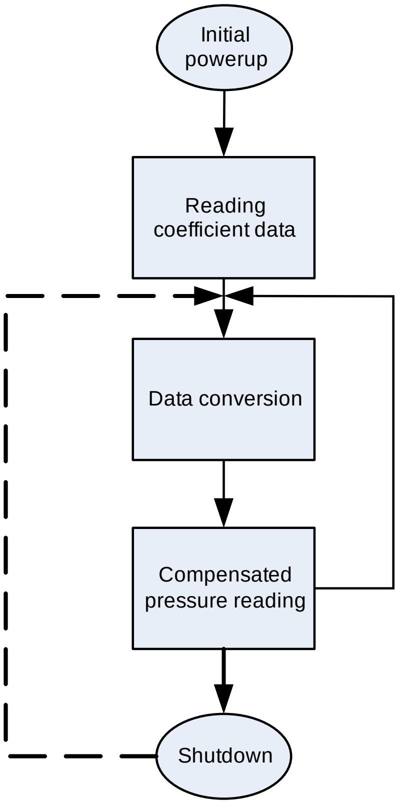


Figure 2. Sequence Flow Chart

The MPL115A interfaces to a host (or system) microcontroller in the user's application. All communications are via I^2C. A typical usage sequence is as follows:

Initial Power-up

All circuit elements are active. I^2C port pins are high impedance and associated registers are cleared. The device then enters standby mode.

Reading Coefficient Data

The user then typically accesses the part and reads the coefficient data. The main circuits within the slave device are disabled during read activity. The coefficients are usually stored in the host microcontoller local memory but can be re-read at any time.

It is not necessary to read the values stored in the host microcontroller multiple times because the coefficients within a device are constant and do not change. However, note that the coefficients will be different from device to device, and cannot be used for another part.

Data Conversion

This is the first step that is performed each time a new pressure reading is required which is initiated by the host sending the CONVERT command. The main system circuits are activated (wake) in response to the command and after the conversion completes, the result is placed into the Pressure and Temperature ADC output registers.

The conversion completes within the maximum conversion time, tc (see Row 7, in the Operating Characteristics Table). The device then enters standby mode.

Compensated Pressure Reading

After the conversion has been given sufficient time to complete, the host microcontroller reads the result from the ADC output registers and calculates the Compensated Pressure, a barometric/atmospheric pressure value which is compensated for changes in temperature and pressure sensor linearity. This is done using the coefficient data from the MPL115A and the raw sampled pressure and temperature ADC output values, in a compensation equation (detailed later). Note that this is an absolute pressure measurement with a vacuum as a reference.

From this step the host controller may either wait and then return to the Data Conversion step to obtain the next pressure reading or it may go to the Shutdown step.

Shutdown

For longer periods of inactivity the user may assert the $\overline{\text{SHDN}}$ input by driving this pin low to reduce system power consumption. This removes power from all internal circuits, including any registers. In the shutdown state, the Pressure and Temperature registers will be reset, losing any previous ADC output values.

This step is exited by taking the $\overline{\text{SHDN}}$ pin high. Wait for the maximum wakeup time, tw (see Row 8, in the Operating Characteristics Table), after which another pressure reading can be taken by transitioning to the data Conversion step.

Table 2. Device Memory Map

Address	Name	Description	Size (bits)
0x00	Padc_MSB	10-bit Pressure ADC output value MSB	8
0x01	Padc_LSB	10-bit Pressure ADC output value LSB	2
0x02	Tadc_MSB	10-bit Temperature ADC output value MSB	8
0x03	Tacd_LSB	10-bit Temperature ADC output value LSB	2
0x04	a0_MSB	a0 coefficient MSB	8
0x05	a0_LSB	a0 coefficient LSB	8
0x06	b1_MSB	b1 coefficient MSB	8
0x07	b1_LSB	b1 coefficient LSB	8
0x08	b2_MSB	b2 coefficient MSB	8
0x09	b2_LSB	b2 coefficient LSB	8
0x0A	c12_MSB	c12 coefficient MSB	8
0x0B	c12_LSB	c12 coefficient LSB	8
0x0C	Reserved*	—	—
0x0D	Reserved*	—	—
0x0E	Reserved*	—	—
0x0F	Reserved*	—	—
0x10	Reserved	—	—
0x11	Reserved	—	—
0x12	CONVERT	Start Pressure and Temperature Conversion	—

*These registers are set to 0x00. These are reserved, and were previously utilized as Coefficient values, c11 and c22, which were always 0x00.

For values with less than 16 bits, the lower LSBs are zero. For example, c12 is 14 bits and is stored into 2 bytes as follows:

$$c12\text{ MS byte} = c12[13:6] = [c12_{b13}, c12_{b12}, c12_{b11}, c12_{b10}, c12_{b9}, c12_{b8}, c12_{b7}, c12_{b6}]$$

$$c12\text{ LS byte} = c12[5:0]\ \&\ \text{“00”} = [c12_{b5}, c12_{b4}, c12_{b3}, c12_{b2}, c12_{b1}, c12_{b0}, 0, 0]$$

3.1 Pressure, Temperature and Coefficient Bit-Width Specifications

The table below specifies the initial coefficient bit-width specifications for the compensation algorithm and the specifications for Pressure and Temperature ADC values.

Pressure, Temperature and Compensation Coefficient Specifications

	a0	b1	b2	c12	Padc	Tadc
Total Bits	16	16	16	14	10	10
Sign Bits	1	1	1	1	0	0
Integer Bits	12	2	1	0	10	10
Fractional Bits	3	13	14	13	0	0
dec pt zero pad	0	0	0	9	0	0

Example Binary Format Definitions:

a0 Signed, Integer Bits = 12, Fractional Bits = 3 :	Coeff a0 = S $I_{11}\ I_{10}\ I_9\ I_8\ I_7\ I_6\ I_5\ I_4\ I_3\ I_2\ I_1\ I_0\ .\ F_2\ F_1\ F_0$
b1 Signed, Integer Bits = 2, Fractional Bits = 7 :	Coeff b1 = S $I_1\ I_0\ .\ F_{12}\ F_{10}\ F_9\ F_8\ F_7\ F_6\ F_5\ F_4\ F_3\ F_2\ F_1\ F_0$
b2 Signed, Integer Bits = 1, Fractional Bits = 14 :	Coeff b2 = S $I_0\ .\ F_{13}\ F_{12}\ F_{10}\ F_9\ F_8\ F_7\ F_6\ F_5\ F_4\ F_3\ F_2\ F_1\ F_0$
c12 Signed, Integer Bits = 0, Fractional Bits = 13, dec pt zero pad = 9 :	Coeff c12 = S 0 . 000 000 000 $F_{12}\ F_{10}\ F_9\ F_8\ F_7\ F_6\ F_5\ F_4\ F_3\ F_2\ F_1\ F_0$
Padc Unsigned, Integer Bits = 10 :	Padc U = $I_9\ I_8\ I_7\ I_6\ I_5\ I_4\ I_3\ I_2\ I_1\ I_0$
Tadc Unsigned, Integer Bits =10 :	Tadc U = $I_9\ I_8\ I_7\ I_6\ I_5\ I_4\ I_3\ I_2\ I_1\ I_0$

NOTE: Negative coefficients are coded in 2’s complement notation.

3.2 Compensation

The 10-bit compensated pressure output, Pcomp, is calculated as follows:

$$\mathrm{Pcomp} = \mathrm{a0} + (\mathrm{b1} + \mathrm{c12} \cdot \mathrm{Tadc}) \cdot \mathrm{Padc} + \mathrm{b2} \cdot \mathrm{Tadc} \qquad \textbf{Eqn. 1}$$

Where:

Padc is the 10-bit pressure ADC output of the MPL115A

Tadc is the 10-bit temperature ADC output of the MPL115A

a0 is the pressure offset coefficient

b1 is the pressure sensitivity coefficient

b2 is the temperature coefficient of offset (TCO)

c12 is the temperature coefficient of sensitivity (TCS)

Pcomp will produce a value of 0 with an input pressure of 50 kPa and will produce a full-scale value of 1023 with an input pressure of 115 kPa.

$$\text{Pressure (kPa)} = \mathrm{Pcomp} \cdot \left[\frac{115 - 50}{1023}\right] + 50 \qquad \textbf{Eqn. 2}$$

3.3 Evaluation Sequence, Arithmetic Circuits

The following is an example of the calculation for Pcomp, the compensated pressure output. Input values are in **bold.**

c12x2 = **c12** * **Tadc**

a1 = **b1** + c12x2

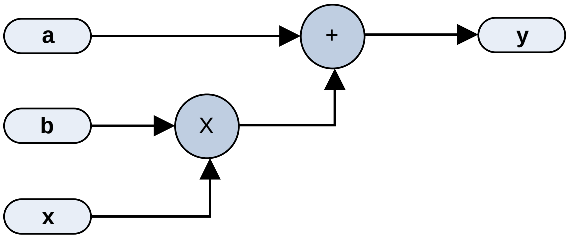
a1x1 = a1 * **Padc**

y1 = **a0** + a1x1

a2x2 = **b2** * **Tadc**

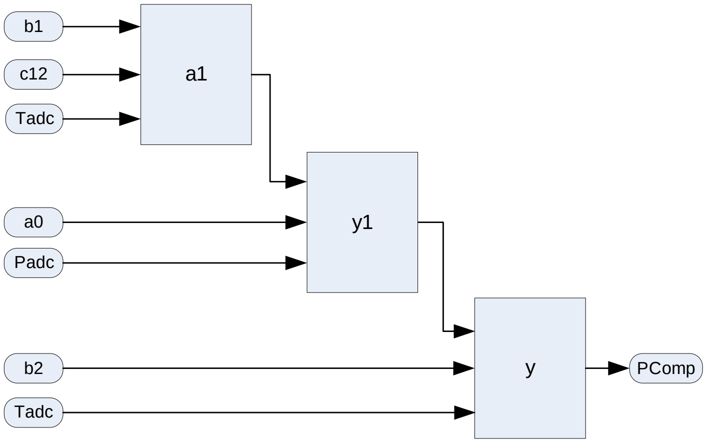
Pcomp = y1 + a2x2

This can be calculated as a succession of Multiply Accumulates (MACs) operations of the form *y = a + b * x:*



The polynomial can be evaluated (Equation 1) as a sequence of 3 MACs:

$$Pcomp = a0 + (b1 + c12 \cdot Tadc) \cdot Padc + b2 \cdot Tadc$$



Please refer to Freescale application note AN3785 for more detailed notes on implementation.

3.4 I^{2}C Device Read/Write Operations

All device read/write operations are memory mapped. Device actions e.g. “Start Conversions” are controlled by writing to the appropriate memory address location.

- For I^{2}C the 7-bit Device Address (from Table 2) has a read/write toggle bit, where the least significant bit is ‘1’ for read operations or ‘0’ for write operations. The Device Address is 0xC0 for a *Write* and the Device Address is 0xC1 for a *Read*.
- The most significant bit in the Command tables below is not used and is don't care (X). In examples given it’s set to ‘0’.

Refer to Sensor I^{2}C Setup and FAQ Application Note AN4481 for more information on I^{2}C communication between the sensor and host controller.

Table 3. I^{2}C Write Commands

Command	Binary	HEX(1)
Devices Address + Write bit	1100 0000	0xC0
Start Conversions	X001 0010	0x12

X = Don’t care

1 = The command byte needs to be paired with a 0x00 as part of the I^{2}C exchange to complete the passing of Start Conversions.

The actions taken by the part in response to each command are as follows:

Table 4. I^{2}C Write Command Description

Command	Action Taken
Start Conversions	Wake main circuits. Start clock. Allow supply stabilization time. Select pressure sensor input. Apply positive sensor excitation and perform A to D conversion. Select temperature input. Perform A to D conversion. Load the Pressure and Temperature registers with the result. Shut down main circuits and clock.

Table 5. I^{2}C Read Command Description

Command	Binary	HEX[1]
Device Address + Read bit	1100 0001	0xC1
Read Pressure MSB	X000 0000	0x00
Read Pressure LSB	X000 0001	0x01
Read Temperature MSB	X000 0010	0x02
Read Temperature LSB	X000 0011	0x03
Read Coefficient data byte 1	X000 0100	0x04

X = don't care

These are MPL115A2 I^{2}C commands to read coefficients, execute Pressure and Temperature conversions, and to read Pressure and Temperature data. The sequence of the commands for the interaction is given as an example to operate the MPL115A2.

Utilizing this gathered data, an example of the calculating the Compensated Pressure reading is given in floating point notation.

I^{2}C Commands (simplified for communication)

Device Address + write bit "To Write" = 0xC0

Device Address + read bit "To Read" = 0xC1

Command to Write "Convert Pressure and Temperature" = 0x12

Command to Read "Pressure ADC High byte" = 0x00

Command to Read "Pressure ADC Low byte" = 0x01

Command to Read "Temperature ADC High byte" = 0x02

Command to Read "Temperature ADC Low byte" = 0x03

Command to Read "Coefficient data byte 1 High byte" = 0x04

Read Coefficients:

[0xC0], [0x04], [0xC1], [0x3E], [0xCE], [0xB3], [0xF9], [0xC5], [0x17], [0x33], [0xC8]

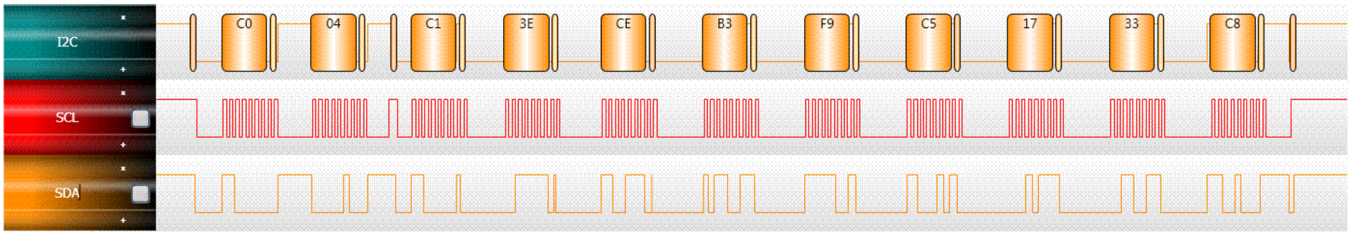


Figure 3. I^{2}C Read Coefficient Datagram

a0 coefficient MSB	=	0x3E					
a0 coefficient LSB	=	0xCE	a0 coefficient	=	0x3ECE	=	2009.75
b1 coefficient MSB	=	0xB3					
b1 coefficient LSB	=	0xF9	b1 coefficient	=	0xB3F9	=	-2.37585
b2 coefficient MSB	=	0xC5					
b2 coefficient LSB	=	0x17	b2 coefficient	=	0xC517	=	-0.92047
c12 coefficient MSB	=	0x33					
c12 coefficient LSB	=	0xC8	c12 coefficient	=	0x33C8	=	0.000790

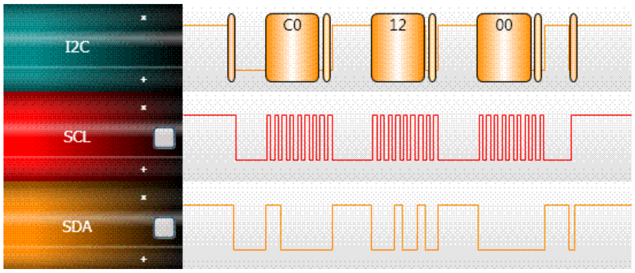


Figure 4. I^{2}C Start Conversion Datagram

Command to I^{2}C Start Conversion, 0x12

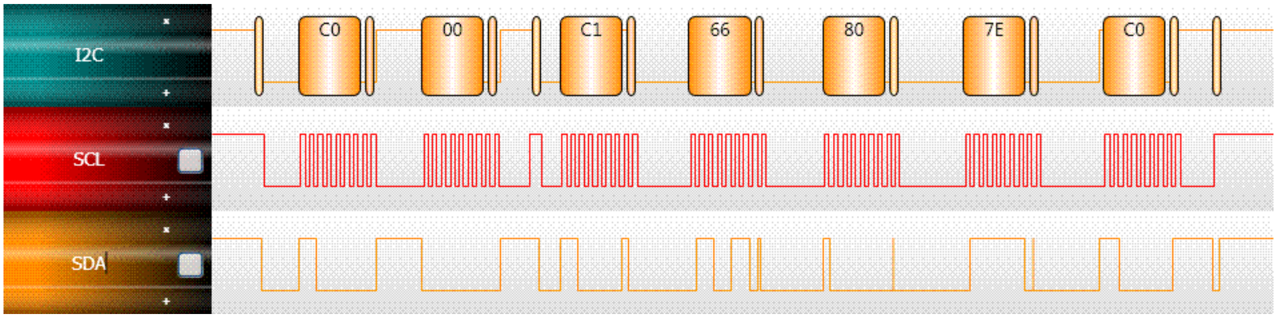


Figure 5. I^{2}C Read Results Datagram

Pressure MSB	=	0x66					
Pressure LSB	=	0x80	Pressure	=	0x6680	=	0110 0110 1100 0000
						=	410 ADC counts
Temperature MSB	=	0x7E					
Temperature LSB	=	0xC0	Temperature	=	0x7EC0	=	0111 1110 1100 0000
						=	507 ADC counts

3.5 Example of Pressure Compensated Calculation in Floating-point Notation

a0 coefficient	=	2009.75
b1 coefficient	=	-2.37585
b2 coefficient	=	-0.92047
c12 coefficient	=	0.000790
Pressure	=	410 ADC counts
Temperature	=	507 ADC counts

Pressure Compensation:

$$\text{Pcomp} = \text{a0} + (\text{b1} + \text{c12} \cdot \text{Tadc}) \cdot \text{Padc} + \text{b2} \cdot \text{Tadc}$$

Using the evaluation sequence shown in Section 3.3:

c12x2	=	c12 * Tadc	=	0.000790 * 507	=	0.40053
a1	=	b1 + c12x2	=	-2.37585 + 0.40053	=	-1.97532
a1x1	=	a1 * Padc	=	-1.97532 * 410	=	-809.8812
y1	=	a0 + a1x1	=	2009.75 + (-809.8812)	=	1199.8688
a2x2	=	b2 * Tadc	=	-0.92047 * 507	=	-466.67829
PComp	=	y1 + a2x2	=	1199.8688 + (-466.67829)	=	733.19051

$$\text{Pressure (kPa)} = \text{Pcomp.}\left[\frac{115-50}{1023}\right] + 50$$

$$= 733.19.\left[\frac{115-50}{1023}\right] + 50$$

$$= 96.59\text{kPa}$$

4 Solder Recommendations

1. Use SAC solder alloy (i.e., Sn-Ag-Cu) with a melting point of about 217°C. It is recommended to use SAC305 (i.e., Sn-3.0 wt.% Ag-0.5 wt.% Cu).
2. Reflow
   - Ramp up rate: 2 to 3°C/s.
   - Preheat flat (soak): 110 to 130s.
   - Reflow peak temperature: 250°C to 260°C (depends on exact SAC alloy composition).
   - Time above 217°C: 40 to 90s (depends on board type, thermal mass of the board/quantities in the reflow).
   - Ramp down: 5 to 6°C/s.
   - Using an inert reflow environment (with O_2 level about 5 to 15 ppm).

NOTE: The stress level and signal offset of the device also depends on the board type, board core material, board thickness and metal finishing of the board.

Please refer to Freescale application note AN3150, Soldering Recommendations for Pressure Sensor Devices for any additional information.

5 Handling Recommendations

It is recommended to handle the MPL115A pressure sensor with a vacuum pick and place tool. Sharp objects utilized to move the MPL115A pressure sensor increase the possibility of damage via a foreign object/tool into the small exposed port.

The sensor die is sensitive to light exposure. Direct light exposure through the port hole can lead to varied accuracy of pressure measurement. Avoid such exposure to the port during normal operation.

Please note that the Pin 1 designator is on the bottom of the package. Do not use the port as a orientation reference in production.

6 Soldering/Landing Pad Information

The LGA package is compliant with the RoHS standard. It is recommended to use a no-clean solder paste to reduce cleaning exposure to high pressure and chemical agents that can damage or reduce life span of the Pressure sensing element.

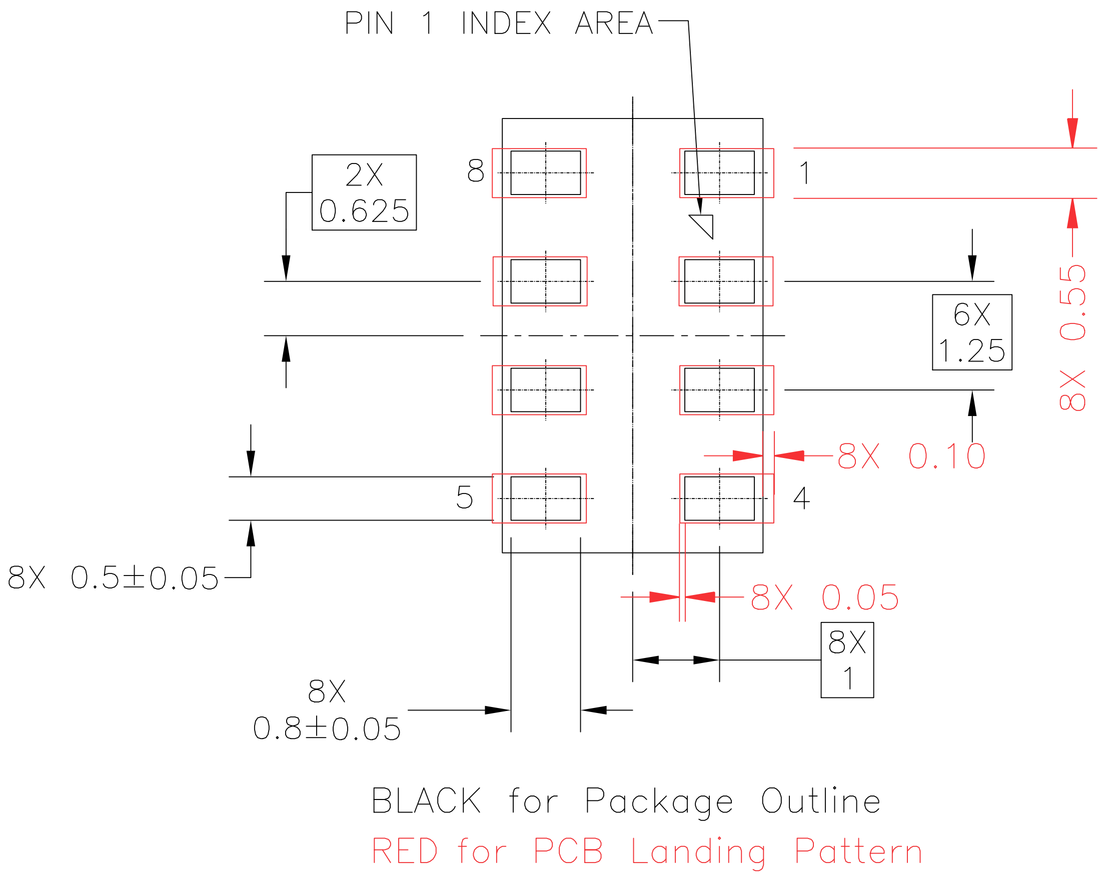
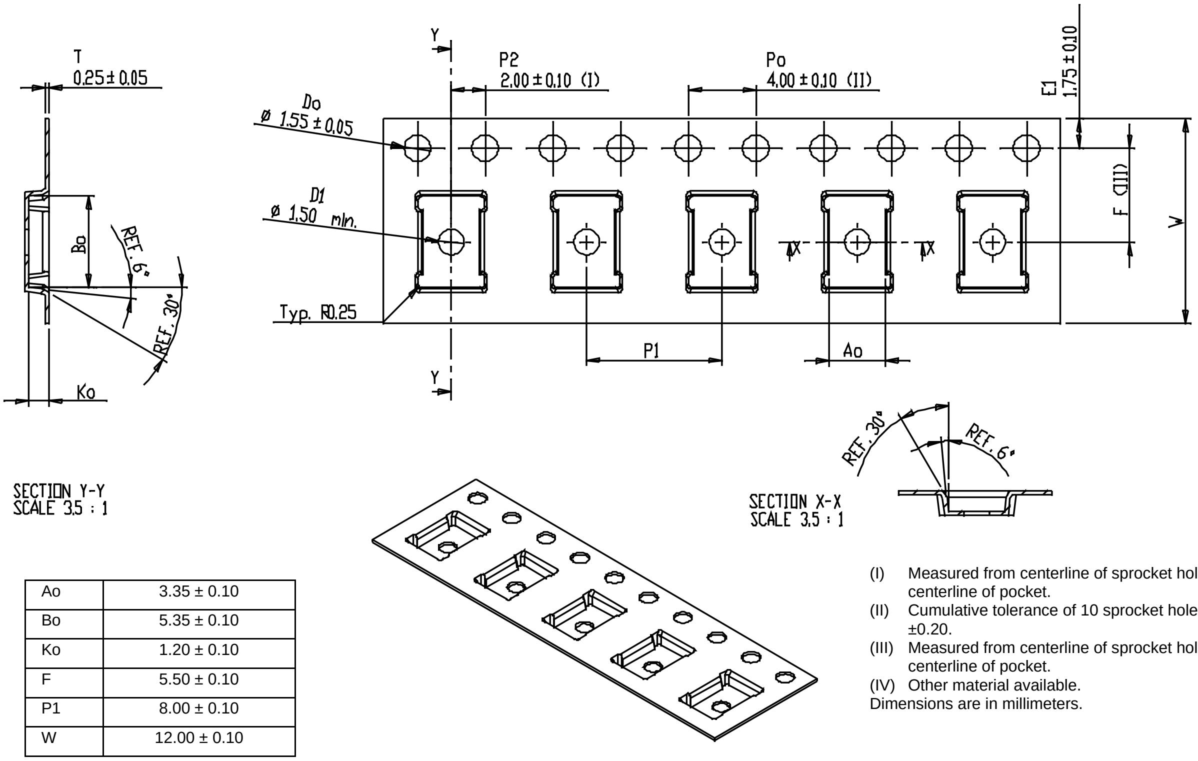


Figure 6. MPL115A2 Recommended PCB Landing Pattern

7 Tape and Reel Specifications



Ao	3.35 ± 0.10
Bo	5.35 ± 0.10
Ko	1.20 ± 0.10
F	5.50 ± 0.10
P1	8.00 ± 0.10
W	12.00 ± 0.10

(I) Measured from centerline of sprocket hole to centerline of pocket.
(II) Cumulative tolerance of 10 sprocket holes is ±0.20.
(III) Measured from centerline of sprocket hole to centerline of pocket.
(IV) Other material available.
Dimensions are in millimeters.

Figure 7. LGA (3 by 5) Embossed Carrier Tape Dimensions

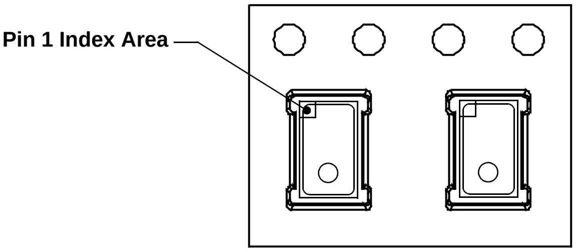
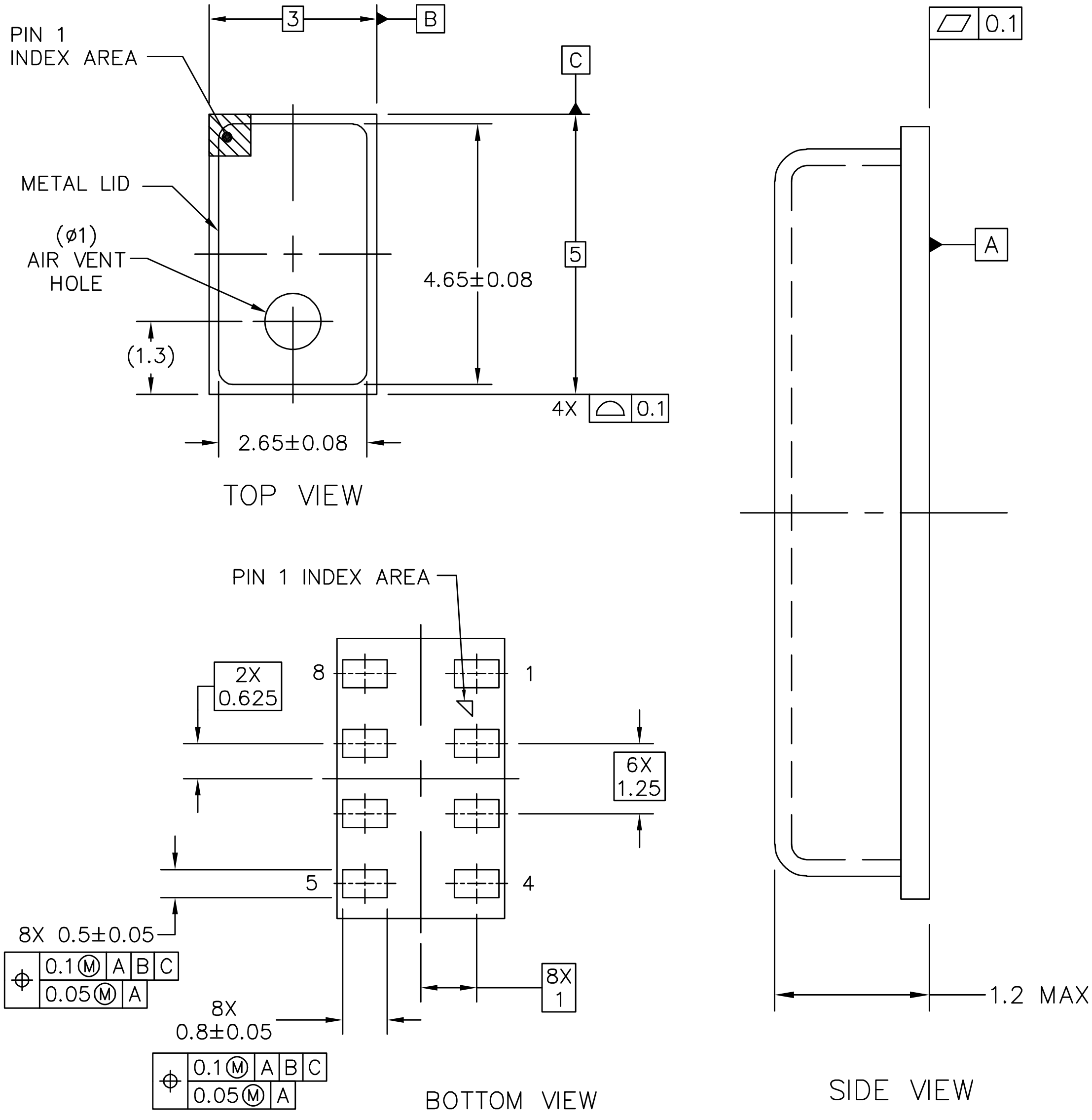


Figure 8. Device Orientation in Chip Carrier

PACKAGE DIMENSIONS



NOTES:

1. ALL DIMENSIONS IN MILLIMETERS.
2. DIMENSIONING AND TOLERANCING PER ASME Y14. 5M–1994

© FREESCALE SEMICONDUCTOR, INC. ALL RIGHTS RESERVED.	**MECHANICAL OUTLINE**	PRINT VERSION NOT TO SCALE
TITLE: LGA 8 I/O, 3 X 5 X 1.25 PITCH, SENSOR 1.2MAX MM PKG	DOCUMENT NO: 98ASA10829D	REV: A
	CASE NUMBER: 2015–02	10 MAR 2010
	STANDARD: NON–JEDEC	

CASE 2015-02
ISSUE A
LGA PACKAGE

Related Documentation

The MPL115A2 device features and operations are described in a variety of reference manuals, user guides, and application notes. To find the most-current versions of these documents:

1. Go to the Freescale homepage at:

   http://www.freescale.com/

2. In the Keyword search box at the top of the page, enter the device number MPL115A2.
3. In the Refine Your Result pane on the left, click on the Documentation link.

Table 6. Revision History

Revision number	Revision date	Description of changes
8	06/2012	• Updated graphic on page 1, Section 2.2 Operating Characteristics: Ref 7: Conversion Time: changed Typ from 3.0 to 1.6, Section 3.0 Overview of Functions/Operation: Reading Coefficient Data deleted statement that reading of coefficients may be executed only once, Table 2: added Size (bits) column in table, added new Section 3.4 I^2C Device Read/Write Operations
		•
		•

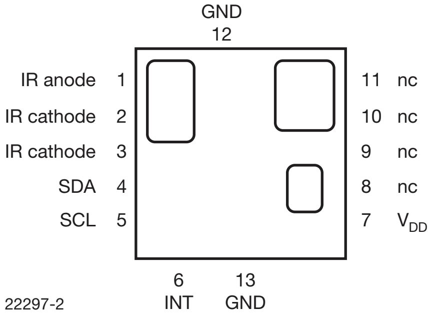
Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

.

freescale™

Fully Integrated Proximity and Ambient Light Sensor with Infrared Emitter, I^{2}C Interface, and Interrupt Function



DESCRIPTION

The VCNL4010 is a fully integrated proximity and ambient light sensor. Fully integrated means that the infrared emitter is included in the package. It has 16 bit resolution. It includes a signal processing IC and features standard I^{2}C communication interface. It features an interrupt function.

APPLICATIONS

- Proximity sensor for mobile devices (e.g. smart phones, touch phones, PDA, GPS) for touch screen locking, power saving, etc.
- Integrated ambient light function for display/keypad contrast control and dimming of mobile devices
- Proximity/optical switch for consumer, computing and industrial devices and displays
- Dimming control for consumer, computing and industrial displays

FEATURES

- Package type: surface mount
- Dimensions (L x W x H in mm): 3.95 x 3.95 x 0.75
- Integrated infrared emitter, ambient light sensor, proximity sensor, and signal conditioning IC
- Interrupt function
- Supply voltage range V_{DD}: 2.5 V to 3.6 V
- Supply voltage range IR anode: 2.5 V to 5 V
- Communication via I^{2}C interface
- I^{2}C Bus H-level range: 1.7 V to 5 V
- Floor life: 168 h, MSL 3, acc. J-STD-020
- Low stand by current consumption: 1.5 μA
- Material categorization: For definitions of compliance please see www.vishay.com/doc?99912

Pb
Pb-free
e4
RoHS
COMPLIANT
HALOGEN
FREE
GREEN
(5-2008)

PROXIMITY FUNCTION

- Built-in infrared emitter and photo-pin-diode for proximity function
- 16 bit effective resolution for proximity detection range ensures excellent cross talk immunity
- Programmable LED drive current from 10 mA to 200 mA in 10 mA steps
- Excellent ambient light suppression by modulating the infrared signal
- Proximity distance up to 200 mm

AMBIENT LIGHT FUNCTION

- Built-in ambient light photo-pin-diode with close-to-human-eye sensitivity
- 16 bit dynamic range from 0.25 lx to 16 klx
- 100 Hz and 120 Hz flicker noise rejection

PRODUCT SUMMARY

PART NUMBER	OPERATING RANGE (mm)	OPERATING VOLTAGE RANGE (V)	I^{2}C BUS VOLTAGE RANGE (V)	LED PULSE CURRENT [1] (mA)	AMBIENT LIGHT RANGE (lx)	AMBIENT LIGHT RESOLUTION (lx)	OUTPUT CODE
VCNL4010	1 to 200	2.5 to 3.6	1.7 to 5	10 to 200	0.25 to 16 383	0.25	16 bit, I^{2}C

Note

[1] Adjustable through I^{2}C interface

ORDERING INFORMATION

ORDERING CODE	PACKAGING	VOLUME [(1)]	REMARKS
VCNL4010-GS08	Tape and reel	MOQ: 1800 pcs	3.95 mm x 3.95 mm x 0.75 mm
VCNL4010-GS18		MOQ: 7000 pcs	
VCNL4000Demokit [(2)]	-	MOQ: 1 pc	-

Notes

[(1)] MOQ: minimum order quantity

[(2)] VCNL4000 Demokit provides USB dongle, basic software including Vishay licence. The VCNL4010 sensor board could be ordered free of charge by contacting sensorstechsupport@vishay.com. Software updates for VCNL4010 can be downloaded from our web site: www.vishay.com/???/

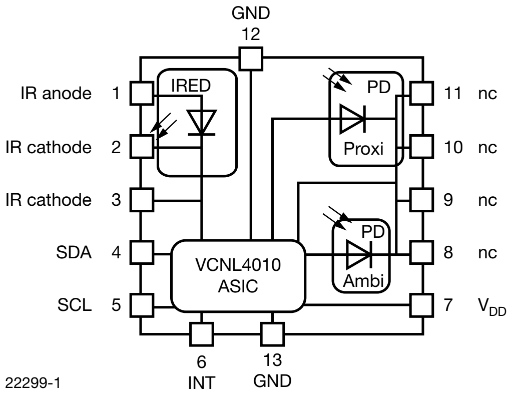
ABSOLUTE MAXIMUM RATINGS (T_{amb} = 25 °C, unless otherwise specified)

PARAMETER	TEST CONDITION	SYMBOL	MIN.	MAX.	UNIT
Supply voltage		V_{DD}	- 0.3	5.5	V
Operation temperature range		T_{amb}	- 25	+ 85	°C
Storage temperature range		T_{stg}	- 40	+ 85	°C
Total power dissipation	$T_{amb} \leq 25$ °C	P_{tot}		50	mW
Junction temperature		T_j		100	°C

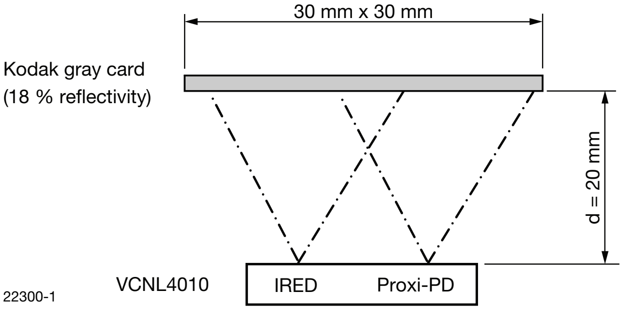
BASIC CHARACTERISTICS (T_{amb} = 25 °C, unless otherwise specified)

PARAMETER	TEST CONDITION	SYMBOL	MIN.	TYP.	MAX.	UNIT
Supply voltage V_{DD}			2.5		3.6	V
Supply voltage IR anode			2.5		5	V
I^2C Bus H-level range			1.7		5	V
INT H-level range			1.7		5	V
INT low voltage	3 mA sink current				0.4	V
Current consumption	Standby current, no IRED-operation			1.5	2	µA
Current consumption proximity mode incl. IRED (averaged)	2 measurements per second, IRED current 20 mA			5		µA
	250 measurements per second, IRED current 20 mA			520		µA
	2 measurements per second, IRED current 200 mA			35		µA
	250 measurements per second, IRED current 200 mA			4.0		mA
Current consumption ambient light mode	2 measurements per second averaging = 1			2.5		µA
	8 measurements per second averaging = 1			10		µA
	2 measurements per second averaging = 64			160		µA
	8 measurements per second averaging = 64			640		µA
Ambient light resolution	Digital resolution (LSB count)		0.25			lx
Ambient light output	E_V = 100 lx averaging = 64			400		counts
I^2C clock rate range		f_{SCL}			3400	kHz

CIRCUIT BLOCK DIAGRAM



TEST CIRCUIT



Note

- nc must not be electrically connected
  Pads 8 to 11 are only considered as solder pads

BASIC CHARACTERISTICS (T_{amb} = 25 °C, unless otherwise specified)

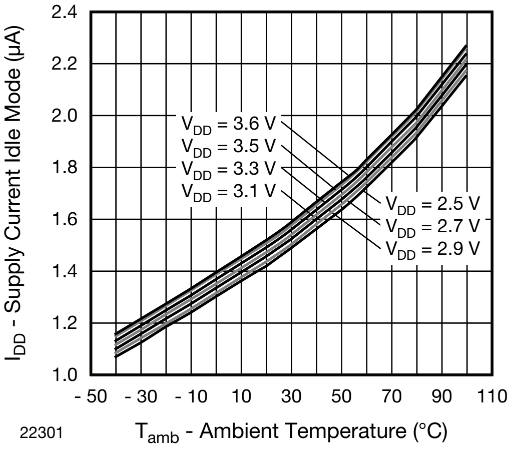


Fig. 1 - Idle Current vs. Ambient Temperature

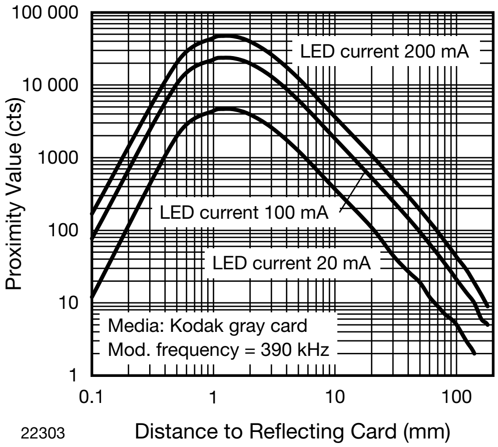


Fig. 3 - Proximity Value vs. Distance

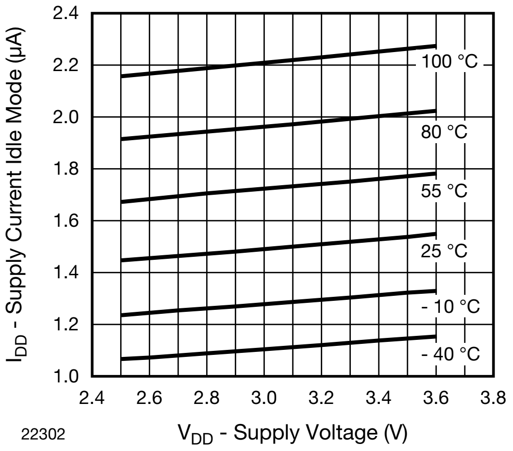


Fig. 2 - Idle Current vs. V_{DD}

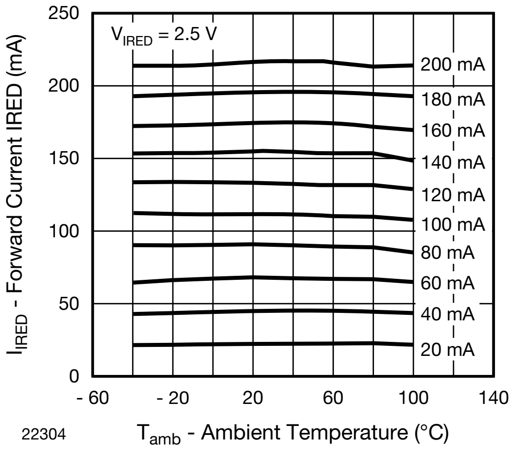


Fig. 4 - Forward Current vs. Temperature

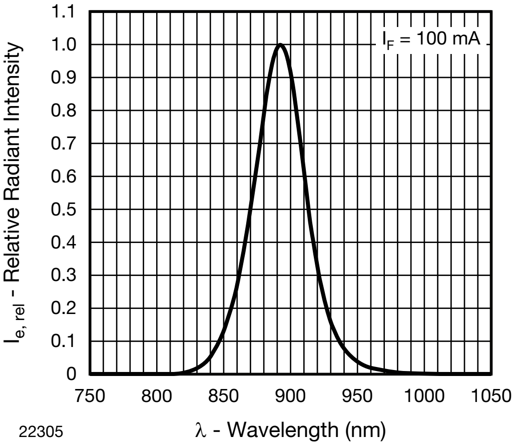


Fig. 5 - Relative Radiant Intensity vs. Wavelength

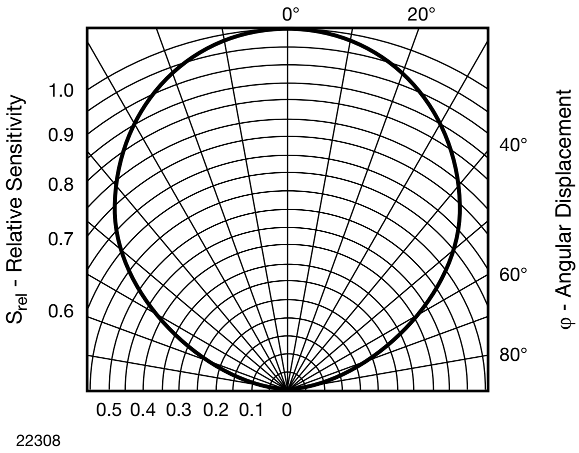


Fig. 8 - Relative Radiant Sensitivity vs. Angular Displacement (Proximity Sensor)

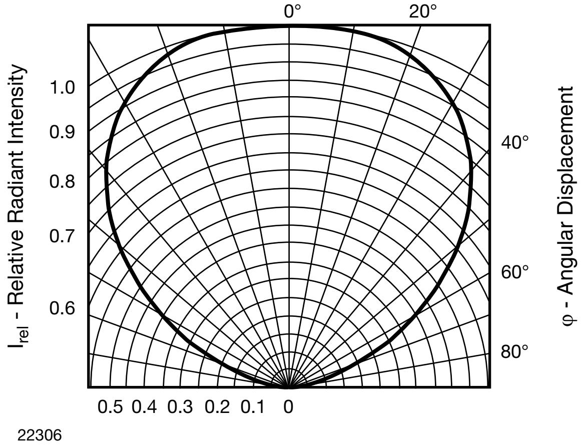


Fig. 6 - Relative Radiant Intensity vs. Angular Displacement

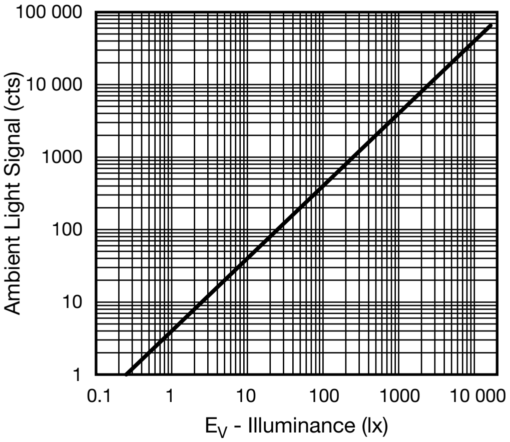


Fig. 9 - Ambient Light Value vs. Illuminance

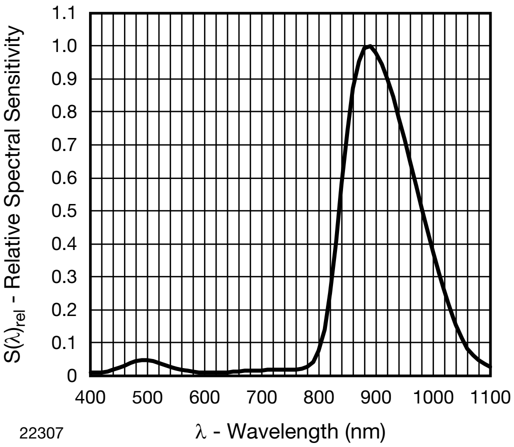


Fig. 7 - Relative Spectral Sensitivity vs. Wavelength (Proximity Sensor)

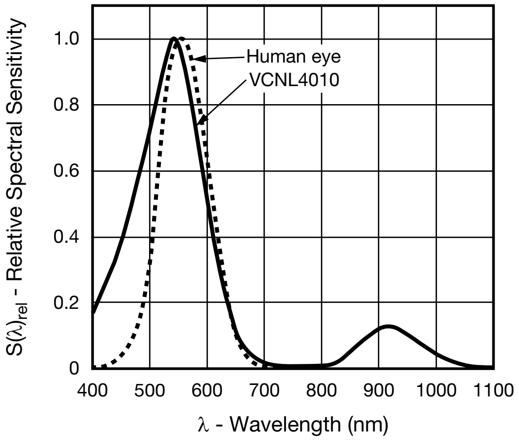


Fig. 10 - Relative Spectral Sensitivity vs. Wavelength (Ambient Light Sensor)

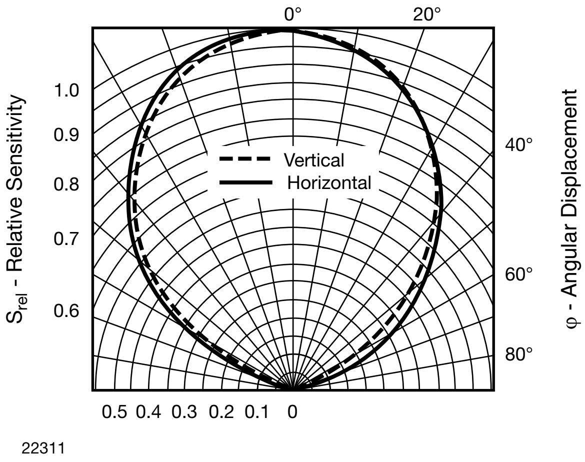


Fig. 11 - Relative Radiant Sensitivity vs. Angular Displacement (Ambient Light Sensor)

APPLICATION INFORMATION

VCNL4010 is a cost effective solution of proximity and ambient light sensor with I^2C bus interface. The standard serial digital interface is easy to access "Proximity Signal" and "Light Intensity" without complex calculation and programming by external controller. Beside the digital output also a flexible programmable interrupt pin is available.

1. Application Circuit

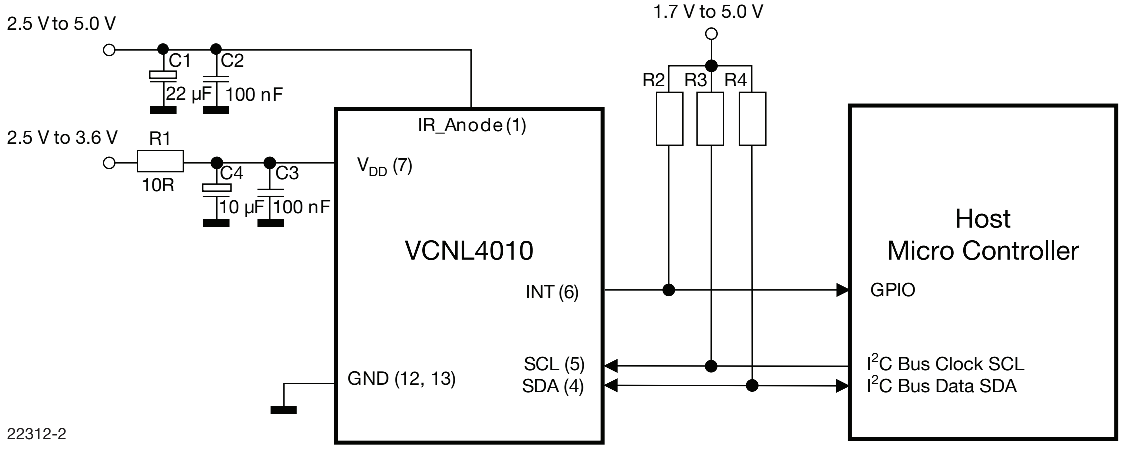


Fig. 12 - Application Circuit
(x) = Pin Number

Note

- The interrupt pin is an open drain output. The needed pull-up resistor may be connected to the same supply voltage as the application controller and the pull-up resistors at SDA/SCL. Proposed value R2 should be >1 kΩ , e.g. 10 kΩ to 100 kΩ.
  Proposed value for R3 and R4, e.g. 2.2 kΩ to 4.7 kΩ, depend also on the I^2C bus speed.
  For detailed description about set-up and use of the interrupt as well as more application related information see AN: "Designing VCNL4010 into an Application".

2. I²C Interface

The VCNL4010 contains seventeen 8 bit registers for operation control, parameter setup and result buffering. All registers are accessible via I²C communication. Figure 13 shows the basic I²C communication with VCNL4010.

The built in I²C interface is compatible with all I²C modes (standard, fast and high speed).

I²C H-level range = 1.7 V to 5 V.

Please refer to the I²C specification from NXP for details.

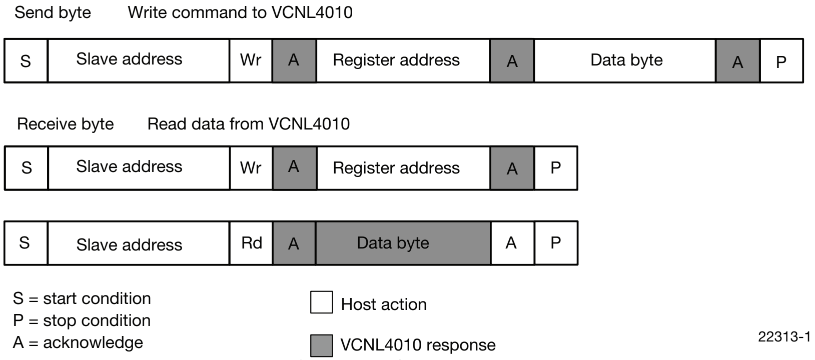


Fig. 13 - Send Byte/Receive Byte Protocol

Device Address

The VCNL4010 has a fix slave address for the host programming and accessing selection. The predefined 7 bit I²C bus address is set to 0010 011 = 13h. The least significant bit (LSB) defines read or write mode. Accordingly the bus address is set to 0010 011x = 26h for write, 27h for read.

Register Addresses

VCNL4010 has seventeen user accessible 8 bit registers. The register addresses are 80h (register #0) to 90h (register #16).

REGISTER FUNCTIONS

Register #0 Command Register

Register address = 80h
The register #0 is for starting ambient light or proximity measurements. This register contains 2 flag bits for data ready indication.

TABLE 1 - COMMAND REGISTER #0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
config_lock	als_data_rdy	prox_data_rdy	als_od	prox_od	als_en	prox_en	selftimed_en

Description	
config_lock	Read only bit. Value = 1
als_data_rdy	Read only bit. Value = 1 when ambient light measurement data is available in the result registers. This bit will be reset when one of the corresponding result registers (reg #5, reg #6) is read.
prox_data_rdy	Read only bit. Value = 1 when proximity measurement data is available in the result registers. This bit will be reset when one of the corresponding result registers (reg #7, reg #8) is read.
als_od	R/W bit. Starts a single on-demand measurement for ambient light. If averaging is enabled, starts a sequence of readings and stores the averaged result. Result is available at the end of conversion for reading in the registers #5(HB) and #6(LB).
prox_od	R/W bit. Starts a single on-demand measurement for proximity. Result is available at the end of conversion for reading in the registers #7(HB) and #8(LB).
als_en	R/W bit. Enables periodic als measurement
prox_en	R/W bit. Enables periodic proximity measurement
selftimed_en	R/W bit. Enables state machine and LP oscillator for self timed measurements; no measurement is performed until the corresponding bit is set

Note

- With setting bit 3 and bit 4 at the same write command, a simultaneously measurement of ambient light and proximity is done. Beside als_en and/or prox_en first selftimed_en needs to be set. On-demand measurement modes are disabled if selftimed_en bit is set. For the selftimed_en mode changes in reading rates (reg #4 and reg #2) can be made only when b0 (selftimed_en bit) = 0. For the als_od mode changes to the reg #4 can be made only when b4 (als_od bit) = 0; this is to avoid synchronization problems and undefined states between the clock domains. In effect this means that it is only reasonable to change rates while no selftimed conversion is ongoing.

Register #1 Product ID Revision Register

Register address = 81h. This register contains information about product ID and product revision.

Register data value of current revision = 21h.

TABLE 2 - PRODUCT ID REVISION REGISTER #1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Product ID				Revision ID			
Description							
Product ID		Read only bits. Value = 2					
Revision ID		Read only bits. Value = 1					

Register #2 Rate of Proximity Measurement

Register address = 82h.

TABLE 3 - PROXIMITY RATE REGISTER #2

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n/a					Rate of Proximity Measurement (no. of measurements per second)		
Description							
Proximity rate		R/W bits. 000 - 1.95 measurements/s (DEFAULT) 001 - 3.90625 measurements/s 010 - 7.8125 measurements/s 011 - 16.625 measurements/s 100 - 31.25 measurements/s 101 - 62.5 measurements/s 110 - 125 measurements/s 111 - 250 measurements/s					

Note

- If self_timed measurement is running, any new value written in this register will not be taken over until the mode is actualy cycled.

Register #3 LED Current Setting for Proximity Mode

Register address = 83h. This register is to set the LED current value for proximity measurement.

The value is adjustable in steps of 10 mA from 0 mA to 200 mA.

This register also contains information about the used device fuse program ID.

TABLE 4 - IR LED CURRENT REGISTER #3

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Fuse prog ID		IR LED current value					
Description							
Fuse prog ID		Read only bits. Information about fuse program revision used for initial setup/calibration of the device.					
IR LED current value		R/W bits. IR LED current = Value (dec.) x 10 mA. Valid Range = 0 to 20d. e.g. 0 = 0 mA , 1 = 10 mA, …., 20 = 200 mA (2 = 20 mA = DEFAULT) LED Current is limited to 200 mA for values higher as 20d.					

Register #4 Ambient Light Parameter Register

Register address = 84h.

TABLE 5 - AMBIENT LIGHT PARAMETER REGISTER #4

<table>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>Cont. conv. mode</td><td colspan="3">als_rate</td><td>Auto offset compensation</td><td colspan="3">Averaging function
(number of measurements per run)</td></tr>
<tr><td colspan="8"><b>Description</b></td></tr>
<tr><td colspan="2">Cont. conversion mode</td><td colspan="6">R/W bit. Continuous conversion mode.
Enable = 1; Disable = 0 = DEFAULT
This function can be used for performing faster ambient light measurements. Please refer to the application information chapter 3.3 for details about this function.</td></tr>
<tr><td colspan="2">Ambient light measurement rate</td><td colspan="6">R/W bits. Ambient light measurement rate
000 - 1 samples/s
001 - 2 samples/s = DEFAULT
010 - 3 samples/s
011 - 4 samples/s
100 - 5 samples/s
101 - 6 samples/s
110 - 8 samples/s
111 - 10 samples/s</td></tr>
<tr><td colspan="2">Auto offset compensation</td><td colspan="6">R/W bit. Automatic offset compensation.
Enable = 1 = DEFAULT; Disable = 0
In order to compensate a technology, package or temperature related drift of the ambient light values there is a built in automatic offset compensation function.
With active auto offset compensation the offset value is measured before each ambient light measurement and subtracted automatically from actual reading.</td></tr>
<tr><td colspan="2">Averaging function</td><td colspan="6">R/W bits. Averaging function.
Bit values sets the number of single conversions done during one measurement cycle. Result is the average value of all conversions.
Number of conversions = $2^{\text{decimal\_value}}$ e.g. 0 = 1 conv., 1 = 2 conv, 2 = 4 conv., ….7 = 128 conv.
DEFAULT = 32 conv. (bit 2 to bit 0: 101)</td></tr>
</table>

Note

- If self_timed measurement is running, any new value written in this register will not be taken over until the mode is actualy cycled.

Register #5 and #6 Ambient Light Result Register

Register address = 85h and 86h. These registers are the result registers for ambient light measurement readings.

The result is a 16 bit value. The high byte is stored in register #5 and the low byte in register #6.

TABLE 6 - AMBIENT LIGHT RESULT REGISTER #5

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description							
Read only bits. High byte (15:8) of ambient light measurement result							

TABLE 7 - AMBIENT LIGHT RESULT REGISTER #6

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description							
Read only bits. Low byte (7:0) of ambient light measurement result							

Register #7 and #8 Proximity Measurement Result Register

Register address = 87h and 88h. These registers are the result registers for proximity measurement readings.

The result is a 16 bit value. The high byte is stored in register #7 and the low byte in register #8.

TABLE 8 - PROXIMITY RESULT REGISTER #7

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description							
Read only bits. High byte (15:8) of proximity measurement result							

TABLE 9 - PROXIMITY RESULT REGISTER #8

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description							
Read only bits. Low byte (7:0) of proximity measurement result							

Register #9 Interrupt Control Register

Register address = 89h.

TABLE 10 - INTERRUPT CONTROL REGISTER #9

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Int count exceed			n/a	INT_PROX_ ready_EN	INT_ALS_ ready_EN	INT_THRES_EN	INT_THRES_ SEL

Description	
Int count exceed	R/W bits. These bits contain the number of consecutive measurements needed above/below the threshold 000 - 1 count = DEFAULT 001 - 2 count 010 - 4 count 011 - 8 count 100 -16 count 101 - 32 count 110 - 64 count 111 - 128 count
INT_PROX_ready_EN	R/W bit. Enables interrupt generation at proximity data ready
INT_ALS_ ready_EN	R/W bit. Enables interrupt generation at ambient data ready
INT_THRES_EN	R/W bit. Enables interrupt generation when high or low threshold is exceeded
INT_THRES_SEL	R/W bit. If 0: thresholds are applied to proximity measurements If 1: thresholds are applied to als measurements

Register #10 and #11 Low Threshold

Register address = 8Ah and 8Bh. These registers contain the low threshold value. The value is a 16 bit word. The high byte is stored in register #10 and the low byte in register #11.

TABLE 11 - LOW THRESHOLD REGISTER #10							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description							
R/W bits. High byte (15:8) of low threshold value							

TABLE 12 - LOW THRESHOLD REGISTER #11							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description							
R/W bits. Low byte (7:0) of low threshold value							

Register #12 and #13 High Threshold

Register address = 8Ch and 8Dh. These registers contain the high threshold value. The value is a 16 bit word. The high byte is stored in register #12 and the low byte in register #13.

TABLE 13 - HIGH THRESHOLD REGISTER #12							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description							
R/W bits. High byte (15:8) of high threshold value							

TABLE 14 - HIGH THRESHOLD REGISTER #13							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Description							
R/W bits. Low byte (7:0) of high threshold value							

Register #14 Interrupt Status Register

Register address = 8Eh. This register contains information about the interrupt status for either proximity or ALS function and indicates if high or low going threshold exceeded.

TABLE 15 - INTERRUPT STATUS REGISTER #14							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
n/a				int_prox_ready	int_als_ready	int_th_low	int_th_hi
Description							
int_prox_ready	R/W bit. Indicates a generated interrupt for proximity						
int_als_ready	R/W bit. Indicates a generated interrupt for als						
int_th_low	R/W bit. Indicates a low threshold exceed						
int_th_hi	R/W bit. Indicates a high threshold exceed						

Note

- Once an interrupt is generated the corresponding status bit goes to 1 and stays there unless it is cleared by writing a 1 in the corresponding bit. The int pad will be pulled down while at least one of the status bit is 1.

Register #15 Proximity Modulator Timing Adjustment

Register address = 8Fh.

<table>
<tr><th colspan="8">TABLE 16 - PROXIMITY MODULATOR TIMING ADJUSTMENT #15</th></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td colspan="3">Modulation delay time</td><td colspan="2">Proximity frequency</td><td colspan="3">Modulation dead time</td></tr>
<tr><td colspan="8"><b>Description</b></td></tr>
<tr><td colspan="2">Modulation delay time</td><td colspan="6">R/W bits. Setting a delay time between IR LED signal and IR input signal evaluation.
This function is for compensation of delays from IR LED and IR photo diode. Also in respect to the possibility for setting different proximity signal frequency. Correct adjustment is optimizing measurement signal level. (DEFAULT = 0)</td></tr>
<tr><td colspan="2">Proximity frequency</td><td colspan="6">R/W bits. Setting the proximity IR test signal frequency
The proximity measurement is using a square IR signal as measurement signal. Four different values are possible:
00 = 390.625 kHz (DEFAULT)
01 = 781.25 kHz
10 = 1.5625 MHz
11 = 3.125 MHz</td></tr>
<tr><td colspan="2">Modulation dead time</td><td colspan="6">R/W bits. Setting a dead time in evaluation of IR signal at the slopes of the IR signal. (DEFAULT = 1)
This function is for reducing of possible disturbance effects.
This function is reducing signal level and should be used carefully.</td></tr>
</table>

Note

- The settings for best performance will be provided by Vishay. With first samples this is evaluated to:
  Delay Time = 0 ; Dead Time = 1 and Prox Frequency = 0 . With that register#15 should be programmed with 1 (= default value).

Register #16 Ambient IR Light Level Register

Register address = 90h.

This register is not intended to be used by customer.

3. IMPORTANT APPLICATION HINTS AND EXAMPLES

3.1 Receiver standby mode

In standby mode the receiver has the lowest current consumption of about 1.5 μA. In this mode only the I^{2}C interface is active. This is always valid, when there are no measurement demands for proximity and ambient light executed. Also the current sink for the IR-LED is inactive, so there is no need for changing register #3 (IR LED current).

3.2 Data Read

In order to get a certain register value, the register has to be addressed without data like shown in the following scheme. After this register addressing, the data from the addressed register is written after a subsequent read command.

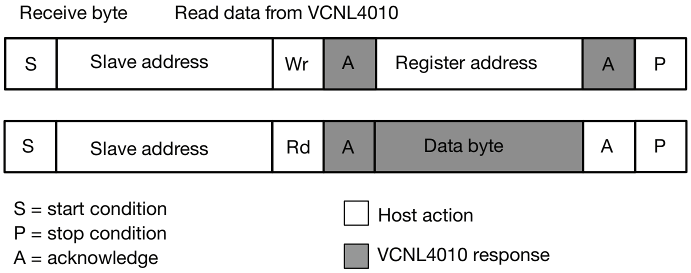


Fig. 14 - Send Byte/Receive Byte Protocol

The stop condition between these write and read sequences is not mandatory. It works also with a repeated start condition.

Note

- For reading out 2 (or more) subsequent registers like the result registers, it is not necessary to address each of the registers separately. After one read command the internal register counter is increased automatically and any subsequent read command is accessing the next register.

Example: read register “Ambient Light Result Register” #5 and #6:

Addressing:command: 26h, 85h (VCNL4010_I^{2}C_Bus_Write_Adr., Ambient Light Result Register #5 [85])

Read register #5:command: 27h, data (VCNL4010_I^{2}C_Bus_Read_Adr., {High Byte Data of Ambient Light Result register #5 [85])}

Read register #6:command: 27h, data (VCNL4010_I^{2}C_Bus_Read_Adr., {Low Byte Data of Ambient Light Result register #6 [86])}

3.3 Continuous Conversion Mode in Ambient Light Measurement

In the following is a detail description of the function “continuous conversion” (bit 7 of register #4)

Standard mode (bit 7 of reg #4 = 0):
In standard mode the ambient light measurement is done during a fixed time frame of 100 ms. The single measurement itself takes actually only appr. 300 µs.

The following figures show examples of this measurement timing in standard mode using averaging function 2 and 8 as examples for illustration (possible values up to 128).

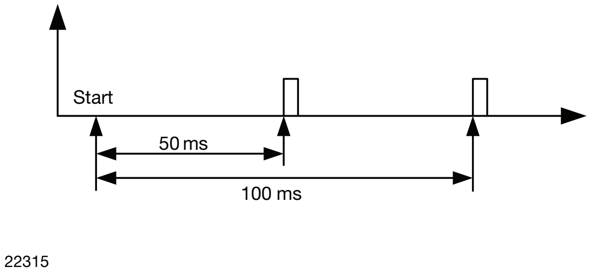


Fig. 15 - Ambient Light Measurement with Averaging = 2; Final Measurement Result = Average of these 2 Measurements

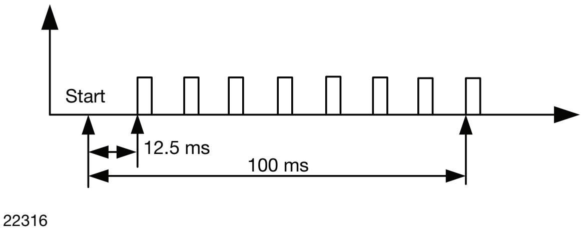


Fig. 16 - Ambient Light Measurement with Averaging = 8; Final Measurement Result = Average of these 8 Measurements

Note

- ≥ Independent of setting of averaging the result is available only after 100 ms.

Continuous conversion mode (bit 7 of register #4 = 1):
In continuous conversion mode the single measurements are done directly subsequent after each other.

See following examples in figure 17 and 18

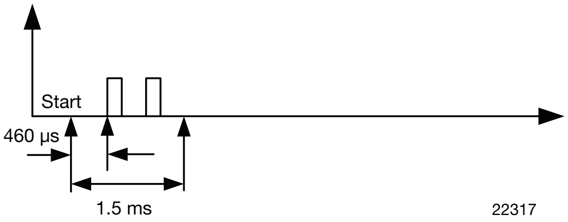


Fig. 17 - Ambient Light Measurement with Averaging = 2; using Continuous Conversion Mode

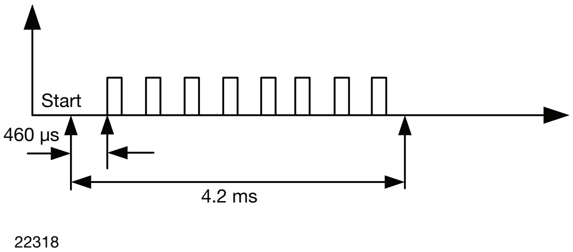
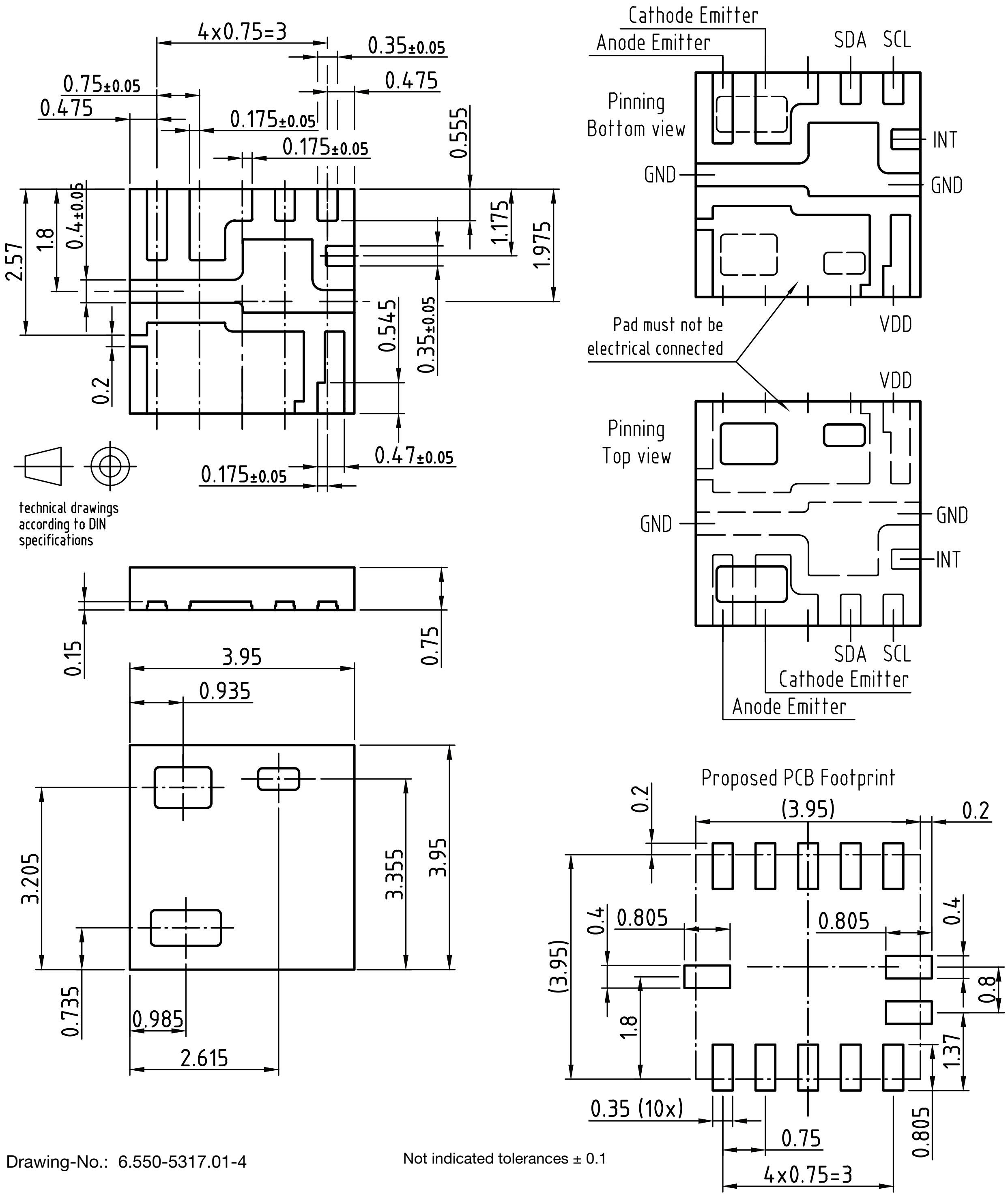


Fig. 18 - Ambient Light Measurement with Averaging = 8; using Continuous Conversion Mode

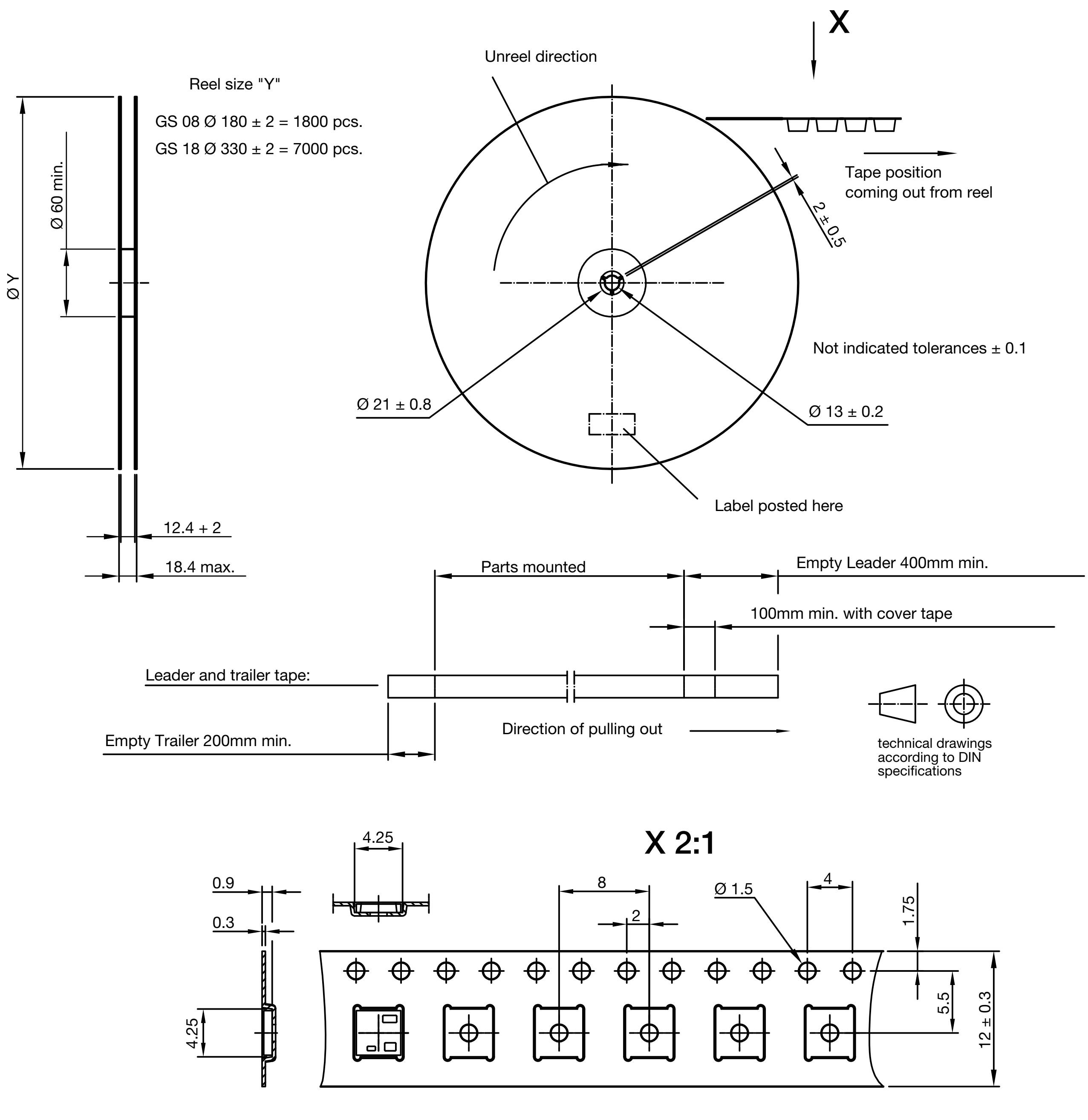
PACKAGE DIMENSIONS in millimeters



Drawing-No.: 6.550-5317.01-4

Not indicated tolerances ± 0.1

TAPE AND REEL DIMENSIONS in millimeters



Drawing-No.: 9.800-5103.01-4

SOLDER PROFILE

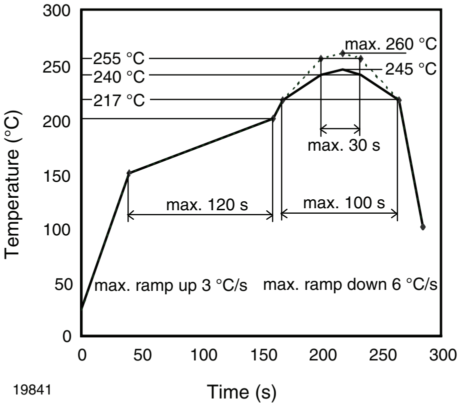


Fig. 19 - Lead (Pb)-free Reflow Solder Profile acc. J-STD-020

DRYPACK

Devices are packed in moisture barrier bags (MBB) to prevent the products from moisture absorption during transportation and storage. Each bag contains a desiccant.

FLOOR LIFE

Floor life (time between soldering and removing from MBB) must not exceed the time indicated on MBB label:

Floor life: 168 h

Conditions: T_{amb} < 30 °C, RH < 60 %

Moisture sensitivity level 3, according to J-STD-020

DRYING

In case of moisture absorption devices should be baked before soldering. Conditions see J-STD-020 or label. Devices taped on reel dry using recommended conditions 192 h at 40 °C (+ 5 °C), RH < 5 %.

Disclaimer

ALL PRODUCT, PRODUCT SPECIFICATIONS AND DATA ARE SUBJECT TO CHANGE WITHOUT NOTICE TO IMPROVE RELIABILITY, FUNCTION OR DESIGN OR OTHERWISE.

Vishay Intertechnology, Inc., its affiliates, agents, and employees, and all persons acting on its or their behalf (collectively, “Vishay”), disclaim any and all liability for any errors, inaccuracies or incompleteness contained in any datasheet or in any other disclosure relating to any product.

Vishay makes no warranty, representation or guarantee regarding the suitability of the products for any particular purpose or the continuing production of any product. To the maximum extent permitted by applicable law, Vishay disclaims (i) any and all liability arising out of the application or use of any product, (ii) any and all liability, including without limitation special, consequential or incidental damages, and (iii) any and all implied warranties, including warranties of fitness for particular purpose, non-infringement and merchantability.

Statements regarding the suitability of products for certain types of applications are based on Vishay’s knowledge of typical requirements that are often placed on Vishay products in generic applications. Such statements are not binding statements about the suitability of products for a particular application. It is the customer’s responsibility to validate that a particular product with the properties described in the product specification is suitable for use in a particular application. Parameters provided in datasheets and/or specifications may vary in different applications and performance may vary over time. All operating parameters, including typical parameters, must be validated for each customer application by the customer’s technical experts. Product specifications do not expand or otherwise modify Vishay’s terms and conditions of purchase, including but not limited to the warranty expressed therein.

Except as expressly indicated in writing, Vishay products are not designed for use in medical, life-saving, or life-sustaining applications or for any other application in which the failure of the Vishay product could result in personal injury or death. Customers using or selling Vishay products not expressly indicated for use in such applications do so at their own risk. Please contact authorized Vishay personnel to obtain written terms and conditions regarding products designed for such applications.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document or by any conduct of Vishay. Product names and markings noted herein may be trademarks of their respective owners.

Material Category Policy

Vishay Intertechnology, Inc. hereby certifies that all its products that are identified as RoHS-Compliant fulfill the definitions and restrictions defined under Directive 2011/65/EU of The European Parliament and of the Council of June 8, 2011 on the restriction of the use of certain hazardous substances in electrical and electronic equipment (EEE) - recast, unless otherwise specified as non-compliant.

Please note that some Vishay documentation may still make reference to RoHS Directive 2002/95/EC. We confirm that all the products identified as being compliant to Directive 2002/95/EC conform to Directive 2011/65/EU.

Vishay Intertechnology, Inc. hereby certifies that all its products that are identified as Halogen-Free follow Halogen-Free requirements as per JEDEC JS709A standards. Please note that some Vishay documentation may still make reference to the IEC 61249-2-21 definition. We confirm that all the products identified as being compliant to IEC 61249-2-21 conform to JEDEC JS709A standards.

Digital relative humidity & temperature sensor RHT03

1. Feature & Application:

*High precision
*Capacitive type
*Full range temperature compensated
*Relative humidity and temperature measurement
*Calibrated digital signal
*Outstanding long-term stability
*Extra components not needed
*Long transmission distance, up to 100 meters
*Low power consumption
*4 pins packaged and fully interchangeable

2. Description:

RHT03 output calibrated digital signal. It applys exclusive digital-signal-collecting-technique and humidity sensing technology, assuring its reliability and stability. Its sensing elements is connected with 8-bit single-chip computer.

Every sensor of this model is temperature compensated and calibrated in accurate calibration chamber and the calibration-coefficient is saved in type of programme in OTP memory, when the sensor is detecting, it will cite coefficient from memory.

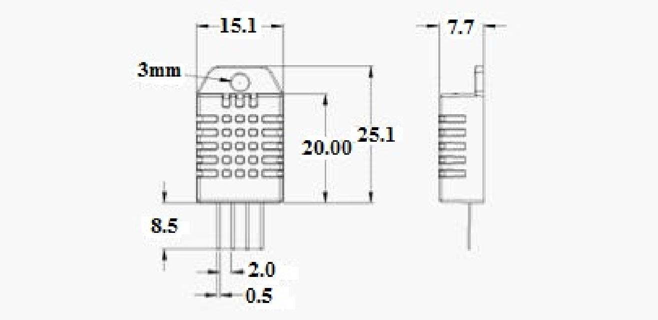
Small size & low consumption & long transmission distance(100m) enable RHT03 to be suited in all kinds of harsh application occasions. Single-row packaged with four pins, making the connection very convenient.

3. Technical Specification:

Model	RHT03	
Power supply	3.3-6V DC	
Output signal	digital signal via MaxDetect 1-wire bus	
Sensing element	Polymer humidity capacitor	
Operating range	humidity 0-100%RH;	temperature -40~80Celsius
Accuracy	**humidity +-2%RH**(Max +-5%RH);	temperature +-0.5Celsius
Resolution or sensitivity	humidity 0.1%RH;	temperature 0.1Celsius
Repeatability	humidity +-1%RH;	temperature +-0.2Celsius

Humidity hysteresis	+-0.3%RH
Long-term Stability	+-0.5%RH/year
Interchangeability	fully interchangeable

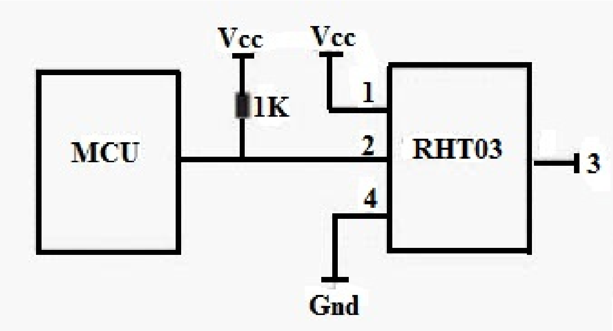
4. Dimensions: (unit----mm)



Pin sequence number: 1 2 3 4 (from left to right direction).

Pin	Function
1	VDD----power supply
2	DATA--signal
3	NULL
4	GND

5. Electrical connection diagram:



6. Operating specifications:

(1) Power and Pins

Power's voltage should be 3.3-6V DC. When power is supplied to sensor, don't send any instruction to the sensor within one second to pass unstable status. One capacitor valued 100nF can be added between VDD and GND for wave filtering.

(2) Communication and signal

MaxDetect 1-wire bus is used for communication between MCU and RHT03. (MaxDetect 1-wire bus is specially designed by MaxDetect Technology Co., Ltd. , it's different from Maxim/Dallas 1-wire bus, so it's incompatible with Dallas 1-wire bus.)

Illustration of MaxDetect 1-wire bus:
Data is comprised of integral and decimal part, the following is the formula for data.
DATA=8 bit integral RH data+8 bit decimal RH data+8 bit integral T data+8 bit decimal T data+8 bit check-sum

If the data transmission is right, check-sum should be:
Check sum=8 bit integral RH data+8 bit decimal RH data+8 bit integral T data+8 bit decimal T data

Example: MCU has received 40 bits data from RHT03 as

0000 0010 1000 1100 **0000 0001 0101 1111** **1110 1110**

16 bits RH data 16 bits T data check sum

Check sum=0000 0010+1000 1100+0000 0001+0101 1111=1110 1110

RH= (0000 0010 1000 1100)/10=65.2%RH

T=(0000 0001 0101 1111)/10=35.1□

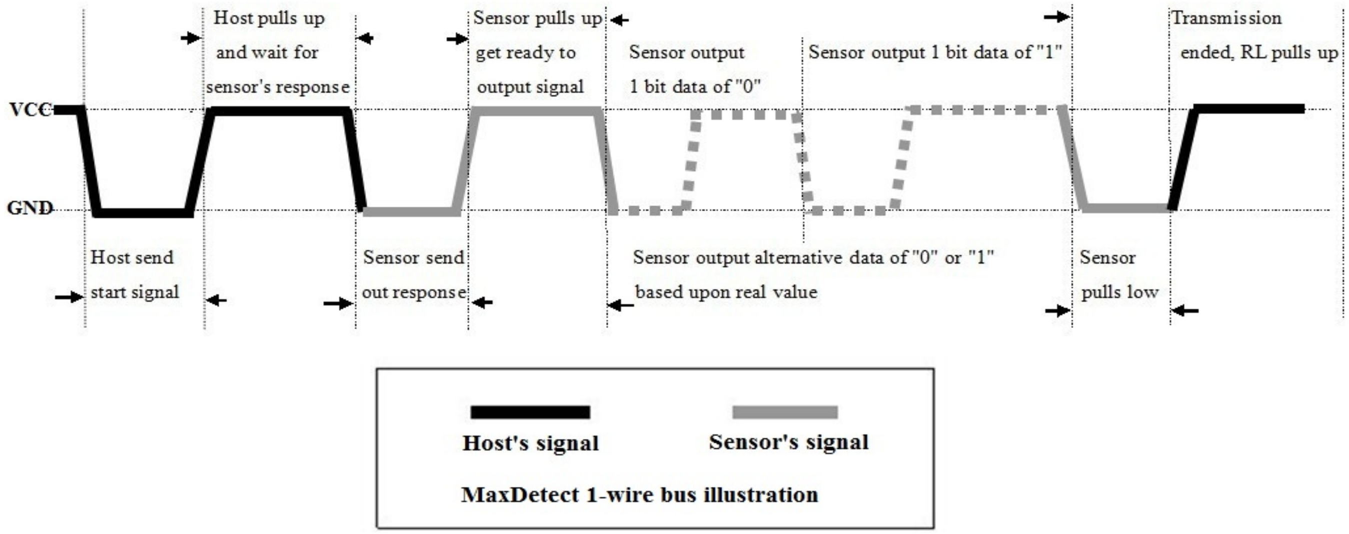
When highest bit of temperature is 1, it means the temperature is below 0 degree Celsius.

Example: 1000 0000 0110 0101, T= minus 10.1□

16 bits T data

When MCU send start signal, RHT03 change from standby-status to running-status. When MCU finishs sending the start signal, RHT03 will send response signal of 40-bit data that reflect the relative humidity and temperature to MCU. Without start signal from MCU, RHT03 will not give response signal to MCU. One start signal for one response data from RHT03 that reflect the relative humidity and temperature. RHT03 will change to standby status when data collecting finished if it don't receive start signal from MCU again.

See below figure for overall communication process, **the interval of whole process must beyond 2 seconds.**

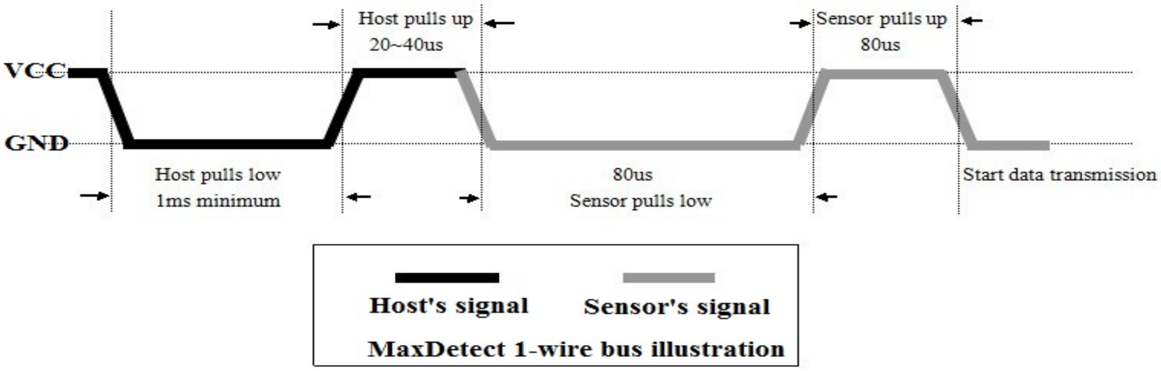

Host pulls up and wait for sensor's response
Sensor pulls up get ready to output signal
Sensor output 1 bit data of "0"
Sensor output 1 bit data of "1"
Transmission ended, RL pulls up
VCC
GND
Host send start signal
Sensor send out response
Sensor output alternative data of "0" or "1" based upon real value
Sensor pulls low
Host's signal
Sensor's signal
MaxDetect 1-wire bus illustration

----

1) Step 1: MCU send out start signal to RHT03 and RHT03 send response signal to MCU

Data-bus's free status is high voltage level. When communication between MCU and RHT03 begins, MCU will pull low data-bus and this process must beyond at least 1~10ms to ensure RHT03 could detect MCU's signal, then MCU will pulls up and wait 20-40us for RHT03's response.

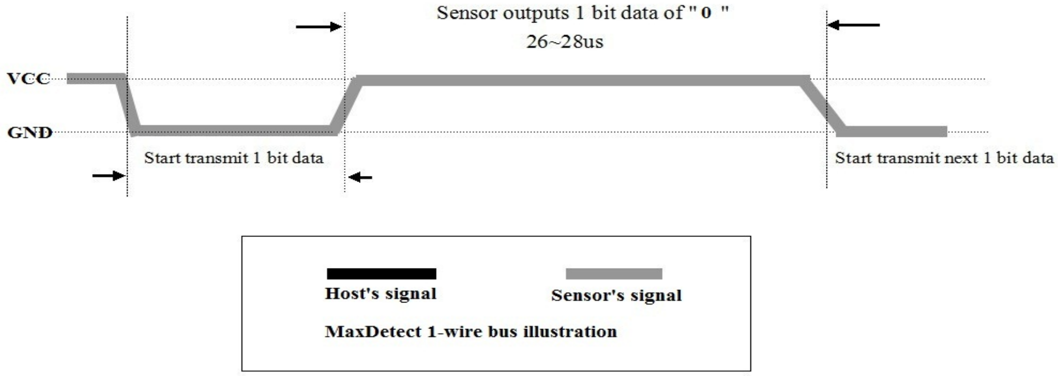
When RHT03 detect the start signal, RHT03 will pull low the bus 80us as response signal, then RHT03 pulls up 80us for preparation to send data. See below figure:

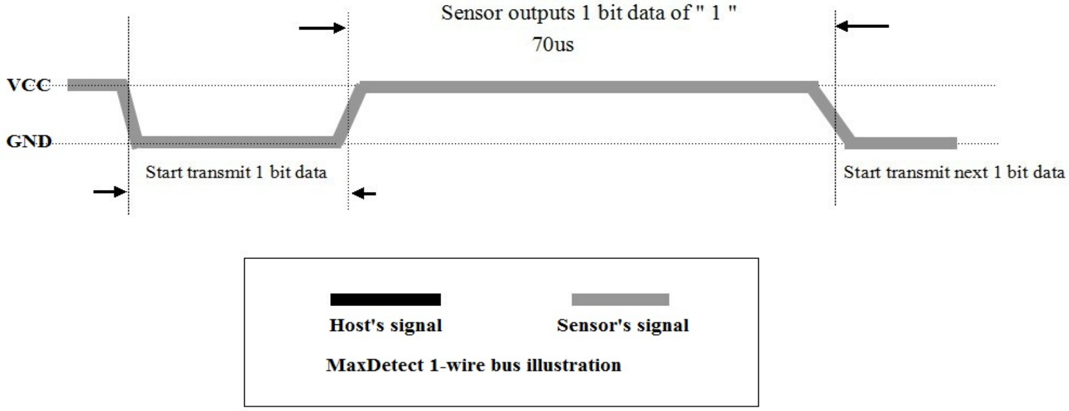


----

2). Step 2: RHT03 send data to MCU

When RHT03 is sending data to MCU, every bit's transmission begin with low-voltage-level that last 50us, the following high-voltage-level signal's length decide the bit is "1" or "0". See below figures:


Sensor outputs 1 bit data of " 0 "
26~28us
VCC
GND
Start transmit 1 bit data
Start transmit next 1 bit data
Host's signal
Sensor's signal
MaxDetect 1-wire bus illustration


Sensor outputs 1 bit data of " 1 "
70us
VCC
GND
Start transmit 1 bit data
Start transmit next 1 bit data
Host's signal
Sensor's signal

MaxDetect 1-wire bus illustration

Attention:

If signal from RHT03 is always high-voltage-level, it means RHT03 is not working properly, please check the electrical connection status.

7. Electrical Characteristics:

Items	Condition	Min	Typical	Max	Unit
Power supply	DC	3.3	5	6	V
Current supply	Measuring	1		1.5	mA
	Stand-by	40	Null	50	uA
Collecting period	Second		2		Second

8. Attentions of application:

(1) Operating and storage conditions

We don't recommend the applying RH-range beyond the range stated in this specification. The RHT03 sensor can recover after working in abnormal operating condition to calibrated status, but will accelerate sensors' aging.

(2) Attentions to chemical materials

Vapor from chemical materials may interfere RHT03's sensitive-elements and debase RHT03's sensitivity.

(3) Disposal when (1) & (2) happens

Step one: Keep the RHT03 sensor at condition of Temperature 50~60Celsius, humidity <10%RH for 2 hours;

Step two: After step one, keep the RHT03 sensor at condition of Temperature 20~30Celsius, humidity >70%RH for 5 hours.

(4) Attention to temperature's affection

Relative humidity strongly depend on temperature, that is why we use temperature compensation technology to ensure accurate measurement of RH. But it's still be much better to keep the sensor at same temperature when sensing.

RHT03 should be mounted at the place as far as possible from parts that may cause change to temperature.

(5) Attentions to light

Long time exposure to strong light and ultraviolet may debase RHT03's performance.

(6) Attentions to connection wires

The connection wires' quality will effect communication's quality and distance, high quality shielding-wire is recommended.

(7) Other attentions

* Welding temperature should be bellow 260Celsius.

* Avoid using the sensor under dew condition.

* Don't use this product in safety or emergency stop devices or any other occasion that failure of RHT03 may cause personal injury.