# VILMA:
# Vehículo IP Liviano Multipropósito Autopropulsado

**Ingeniería Técnica de Sistemas**

**Estudiante**
Javier Ballester Gómez

**Consultor**
Jordi Bécares Ferrés

**Fecha**
23 de Enero de 2015

*A mi compañera Arantxa y nuestros pequeños*
*Iván y Amaya.*
*Gracias por regalarme vuestro tiempo*

# AGRADECIMIENTOS

Me gustaría agradecer a todos los miembros de la comunidad UOC, en especial, a los innumerables estudiantes y consultores con los que he tenido el placer de coincidir y de los que aprender y por cambiar el significado de los conceptos “on-line” y “a distancia” por las palabras proximidad y compañerismo

A Jordi Bécares Ferrés, consultor del proyecto, por sus consejos y recomendaciones, pero sobre todo por sus ánimo, paciencia y perseverancia.

A mi pareja Arantxa, por su generosidad, comprensión y sacrificio, sin el cual me hubiera sido imposible llevar a cabo estos estudios.

A nuestros pequeños Iván y Amaya a por todos los cuentos y horas de juegos que habéis perdonado.

A mis padres,  por enseñarme que el ingrediente que nunca falta en la receta del éxito es el esfuerzo.

A todos vosotros, mi más sincero agradecimiento.

# RESUMEN

Esta memoria pretende documentar los aspectos técnicos más relevantes del trabajo realizado para la elaboración del proyecto final de carrera de Ingeniería Técnica en Informática de Sistemas de la Universitat Oberta de Catalunya.

Este proyecto se enmarca en el campo de los sistemas embebidos, de modo que el trabajo se centra en dos áreas principalmente; Por un lado, el área de Hardware, en el que se trabajará con la electrónica del microcontrolador y la interconexión de éste con los diferentes sensores y dispositivos electrónicos y por el otro lado centrado en el desarrollo de la aplicación software que implemente las funciones del sistema.

Partiendo como base del microcontrolador y módulo de conexiones inalámbricas 802.11g proporcionado, así como otros elementos hardware que se detallan más adelante, se ha construido un prototipo de vehículo con ruedas, capaz de asociarse a una red inalámbrica WiFi para ser controlado mediante una red TCP/IP

El prototipo, que incorpora motores para el movimiento de sus ruedas, se encarga de recibir y ejecutar los movimientos ordenados desde una aplicación remota.

Además el vehículo, es capaz de proporcionar información relativa a su estado en cada momento al aplicativo de gestión remota.

# Tabla de contenido

# Índice de Ilustraciones



# Índice de Tablas

# Introducción

Formalmente, un sistema embebido (S.E) se entiende como una combinación de hardware y software de una computadora diseñada para realizar un número reducido de tareas concretas. En el apartado Hardware, estos sistemas suelen integrarse en plataformas con recursos limitados y están pensados para realizar tareas muy específicas donde prima la seguridad ante fallos y el consumo energético por encima de la flexibilidad operacional.

No obstante, en comparación con los sistemas hardware puros, los SS.EE cuentan con la ventaja de ser más flexibles, puesto que al contar con el componente software, éstos pueden ser reprogramados para realizar funciones diferentes.

En cuanto al software, estos dispositivos suelen contar con un sistema operativo en tiempo real (RTOS). En éste ámbito, el concepto de "tiempo real" significa que el tiempo de respuesta ante un evento es siempre finito y predecible.

Las características más destacables de los sistemas empotrados son:

- Interactúan con su entorno
- Son fiables y tolerantes a fallos
- Son eficientes tanto computacional como energéticamente
- Reducidas dimensiones y peso
- Bajo coste

## 1.1. Justificación

Actualmente los sistemas embebidos están presentes en todos los ámbitos que nos rodean, desde el punto de vista del hardware, su reducido tamaño unido a la creciente capacidad de proceso de los microcontroladores, hacen que estos tipos de sistemas se integren en aparatos de uso cotidiano tales como sistemas de entretenimiento en casa, aplicaciones tan críticas como aplicaciones de control industrial y robótica e incluso controlando los dispositivos de seguridad activa en los vehículos actuales. En el apartado hardware, se parte de la placa de desarrollo LPC1769, que integra un microprocesador ARM Cortex M3.

Desde el punto de vista software, en la actualidad la gestión de recursos en estos sistemas se lleva a cabo principalmente mediante sistemas operativo en tiempo real. En el caso particular de este desarrollo, el prototipo debe cumplir con la premisa de ser capaz de modificar de manera autónoma su comportamiento en función de su entorno. En este caso, el tiempo de respuesta ante un evento externo al sistema es crítico, por lo que la elección de un sistema operativo en tiempo real se estima la más acertada. Para el control del prototipo se ha elegido el sistema operativo en tiempo real FreeRTOS, ya que permite trabajar tanto en modo anticipativo como en modo colaborativo. Además se distribuye bajo licencia GPL y cuenta con soporte para el microprocesador empleado en la construcción de este prototipo

Por último, para la comunicación del vehículo prototipo con el entorno de control remoto, se optó por dotar al sistema de comunicación inalámbrica basada en el protocolo 802.11g que cuenta con la ventaja de poder extender el radio de acción a entornos de red LAN o incluso WAN, descartándose la utilización de otras tecnologías como el Bluetooth, ya que limitaría el radio de acción a unas decenas de metros.

## 1.2. Descripción

El proyecto desarrollado se ha denominado VILMA. (Vehículo Ip Liviano Multipropósito Autopropulsado).

VILMA es un prototipo de vehículo construido sobre una plataforma que incorpora cuatro ruedas controladas por dos pequeños motores de corriente continua. Sobre la plataforma se encuentra la placa de control principal, formado por:

- Módulo de comunicaciones inalámbricas: Cuya funcionalidad es la de posibilitar que el vehículo se conecte a un punto de acceso inalámbrico para poder establecer la comunicación con la aplicación externa de control remoto.

- Sensor de proximidad: Que proporciona una lectura de la distancia a la que se encuentra el obstáculo más próximo, con el fin de dotar al sistema de un mecanismo para evitar colisiones.

- Módulo controlador de motores: Es el actuador encargado dotar de movilidad a la plataforma

- Unidad de control principal: Almacena y ejecuta el sistema operativo así como aplicación principal del prototipo, además de gestionar el funcionamiento del resto de elementos.

- Módulo de suministro y regulación de energía: Que proporciona la alimentación estabilizada a todos los elementos del sistema.

En un ámbito externo al diseño del vehículo, se ha desarrollado una sencilla aplicación web que corre sobre un servidor PHP con el propósito de facilitar al usuario el control remoto del vehículo, además de representar la información relevante del estado del dispositivo.

Esta aplicación web proporciona la funcionalidad de envío de instrucciones de una manera gráfica además de proporcionar información del estado del prototipo tal como el nivel de señal inalámbrica recibido, dirección IP, etc. Por último la aplicación web ofrece una funcionalidad información acerca de los últimos movimientos realizados por el prototipo.

Tras el arranque del sistema, VILMA intentará establecer conexión con el punto de acceso inalámbrico configurado en su programación inicial. En caso de no conseguir conectar con la red inalámbrica configurada después de tres intentos, el vehículo entrará en modo HotSpot, esta funcionalidad se explicará más adelante. En caso de conseguir la conexión a la red, el vehículo abrirá una conexión TCP contra el servidor de la aplicación remota para proporcionar su dirección IP y puerto de escucha.

Finalizada la fase de inicialización, el control principal de VILMA quedará ejecutando constantemente la tarea de recepción de instrucciones.

Al solicitar la ejecución de una instrucción desde la aplicación remota, se efectuará la apertura de una conexión TCP hacia la dirección IP y puerto facilitado por el prototipo durante el arranque del sistema, enviando el mensaje que contiene la instrucción a ejecutar.

Por su parte, el control principal del prototipo, recibe y procesa el comando y pasa la información a la tarea que se encarga de ejecutar los comandos mediante el uso de una cola.

La tarea ejecutora comprueba que no hay impedimentos para llevar a cabo la instrucción, tales como una señal inalámbrica débil o un objeto muy próximo y ordena el movimiento de los motores para efectuar el movimiento solicitado. Después de realizar esta acción, la tarea ejecutora queda en estado de bloqueo para a la espera de ser nuevamente desbloqueada para la ejecución de una nueva instrucción.

## 1.3. Objetivos

Los objetivos planteados para el proyecto se han agrupado en dos categorías; objetivos principales, que son el grupo de objetivos prioritarios para el desarrollo del producto y objetivos secundarios que proporcionan valor añadido al proyecto.

En el grupo de los objetivos principales se destacan:

- Dotar al sistema de comunicación inalámbrica sobre el protocolo 802.11 g
- Crear un sistema de comunicaciones basado en el modelo cliente-servidor para el control del vehículo
- Implementar un interfaz GUI básico para el manejo control del vehículo
- Ofrecer sistema de control de movimiento hacia delante y hacia atrás, así como hacia derecha e izquierda.
- Desarrollar un sistema anti-colisión que detenga el vehículo en caso de detectar un obstáculo.

En el grupo de los objetivos secundarios se encuentran:

- Ampliar las funcionalidades de la aplicación de control remoto para la obtención de información del estado del prototipo y log de últimos movimientos.
- Implementación de alarma contra la pérdida de cobertura que evite la pérdida de control del vehículo al alejarse del punto de acceso inalámbrico.
- Desarrolla de sistema de trazador de recorrido que proporcionará información acerca de los últimos movimientos efectuados por el vehículo.
- Implantación de mecanismo para la recuperación autónoma de señal inalámbrica.

## 1.4. Enfoque y método seguido

El enfoque y metodología empleada para el desarrollo del proyecto se puede dividir en cuatro fases bien diferenciadas

**Fase de estudio y formación.**

Esta fase ha tenido una duración aproximada de 8 semanas y se ha centrado en la toma de contacto con el entorno de desarrollo, interfaz CMSIS así como las funcionalidades más importantes del sistema operativo FreeRTOS desde un punto de vista práctico.

Esta fase de aprendizaje se ha estructurado en 4 apartados que han abarcado los contenidos que se resumen a continuación:

Familiarización con el entorno integrado de desarrollo LPCXpresso y con la placa de desarrollo LPC1769, además de la librería CMSIS para la implementación de una librería simple que maneja el funcionamiento del puerto GPIO para el encendido y apagado de un led.

Estudio del funcionamiento del sistema inalámbrico RN-XV 171 (Wifly) para obtener conectividad a internet, el establecimiento de sockets TCP y el envío y obtención de datos a y desde un servidor HTTP remoto.

Desarrollo de un driver para el manejo de puertos UART para el envío y recepción de datos a través del puerto serie. Desarrollo de una librería para la gestión de las funcionalidades del módulo Wifly para el envío de instrucciones y la recepción de resultados apoyándose en la implementación del driver UART. Aprendizaje de la utilización de semáforos para la protección de accesos concurrentes a recursos en uso

Desarrollo de un sistema productor/consumidor mediante el empleo de colas como mecanismo de comunicación entre tareas o hilos de ejecución y diferencias entre las implementación de esperas activas y esperas pasivas

**Fase de planificación del Proyecto**

El enfoque de esta fase no se ha centrado en el producto final en sí, sino en la planificación y organización para llevar a cabo la totalidad de las tareas de las que consta este proyecto, con la finalidad de obtener un producto terminado. En esta fase, se ha llevado a cabo un estudio exhaustivo de los requisitos técnicos y económicos para la implementación del prototipo, intentando alcanzar un compromiso entre las funcionalidades y coste económico.

Además, se han fijado objetivos que se pretenden alcanzar con la elaboración del proyecto y finalmente se ha elaborado una planificación temporal de los plazos a cumplir para poder llevar un seguimiento y, en caso de ser necesario tomar las acciones correctivas que permitan reconducir las posibles desviaciones respecto a la planificación.

**Fase de Implementación y pruebas**

En esta fase, se efectúa la programación de los drivers y librerías, para el manejo de los diferentes elementos hardware, así como de la aplicación principal del sistema embebido para la implementación de los requisitos funcionales del sistema. Durante esta fase, también se llevan a cabo las pruebas y depuración de los posibles errores así como la corrección de comportamientos no deseados en un entorno de simulación antes de la puesta en producción, una vez finalizada la construcción del prototipo.

Por otra parte, en esta fase también se llevan a cabo las tareas de preparación del servidor PHP que albergará la aplicación para la gestión remota del prototipo, así como la programación de las diferentes funcionalidades de ejecución y representación de información en el lado cliente aplicación.

A continuación, se efectuará el ensamblaje y la interconexión definitiva de todos los elementos que componen el sistema y se llevarán a cabo nuevamente las pruebas para confirmar o corregir cualquier eventualidad no prevista.

Durante el transcurso de esta fase, se han ido recabando las informaciones relevantes de cara a confeccionar la memoria del proyecto

**Fase de finalización de documentación, presentación y entrega**

En esta fase todo el código fuente está finalizado y es el momento de completar el informe técnico y de cerrar la documentación relacionada con la memoria, preparación de una presentación que resuma el contenido documentado de manera exhaustiva en la memoria y entrega del proyecto para su valoración.

## 1.5. Planificación

En este apartado se presentan los esquemas de tiempos de dedicación a cada una de las fases del proyecto con el fin de prever y corregir las posibles desviaciones en la consecución de los hitos marcados.

### 1.5.1. Planificación Inicial

La planificación del proyecto se ha llevado a cabo en tres fases:

- La primera fase: En la que se intenta cubrir los objetivos principales fijados para sistema, además de preparar el entorno para alojar la aplicación web de control remoto y la realización de la primera batería de pruebas.
- La segunda fase: Está dedicada al ensamblaje del sistema en la plataforma del prototipo, además del estudio para la implementación de las funcionalidades que permitan cubrir los objetivos secundarios tales como la integración de sensores de entorno.
- Por último la tercera fase está enteramente dedicada a la elaboración de informe técnico, finalización de la memoria y presentación del proyecto.

A continuación el diagrama de Gantt de la planificación inicial del proyecto se presenta el calendario con las duraciones así como cada uno de los hitos repartido en las tres fases mencionadas anteriormente

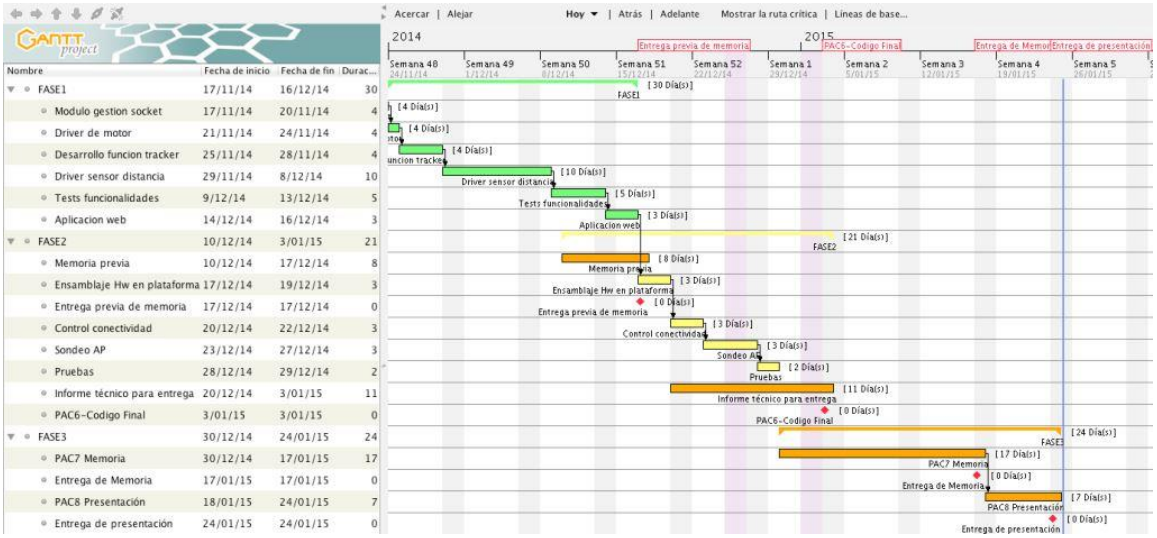



Ilustración 1 Planificación inicial del proyecto

### 1.5.2. Planificación Final

Durante el desarrollo del proyecto, se han producido ciertas desviaciones respecto al calendario presentado en la planificación del proyecto que han tenido que ser corregidas. A continuación se presenta el diagrama de Gantt donde se pueden apreciar los cambios sufridos respecto de la planificación inicial.

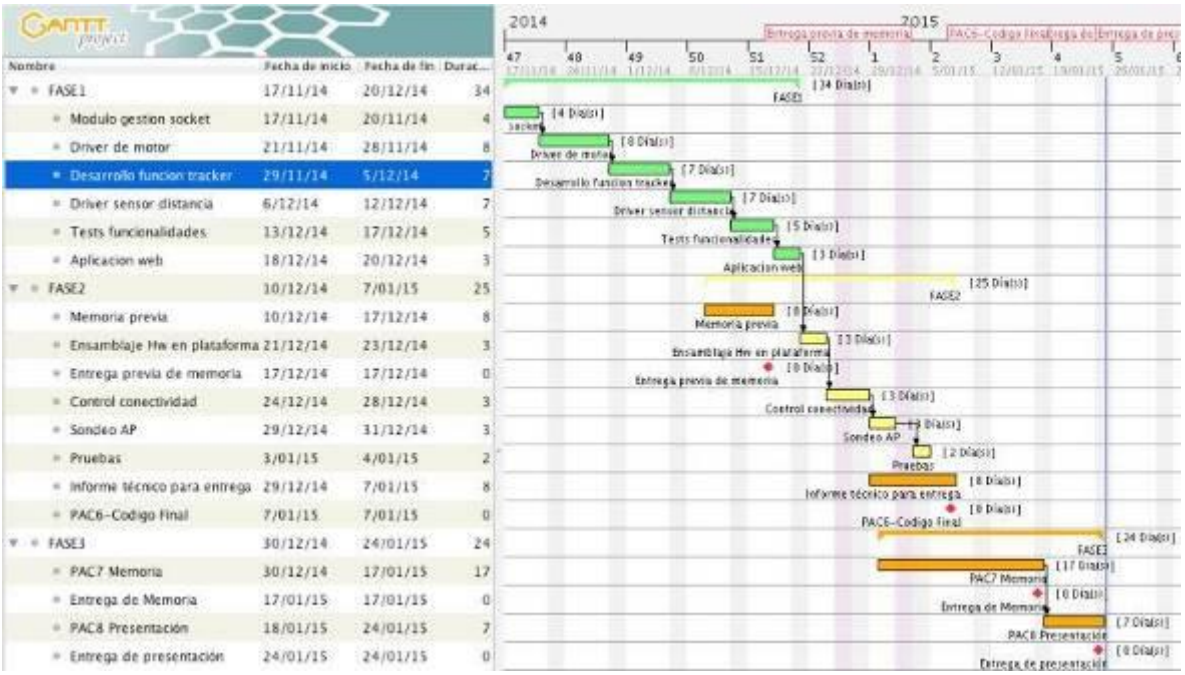


Ilustración 2 Planificación final del proyecto

## Consideraciones sobre la planificación final

Durante la realización del proyecto, ha habido ciertas circunstancias que ha provocado variaciones respecto del calendario presentado en la planificación inicial.

En un primer momento, se observó durante las pruebas de puesta en funcionamiento del driver de motor que el funcionamiento del hardware encargado de controlar de motores no era el correcto de modo que se optó por sustituir este actuador.

Como consecuencia del cambio en el hardware, hubo que afrontar dos problemas: El primero consistió en desarrollar una nueva librería para manejar los motores desde el principio lo que lo que acarreó un retraso de unos 4 días respecto a la planificación.

Además, el módulo de control de motores sustituido integraba un regulador de corriente continua que permitía la alimentación de la placa del microcontrolador directamente desde el, al sustituir estos componentes, fue necesario rediseñar el esquema de alimentación eléctrica del prototipo, lo que ocasiono un retraso adicional así como la inversión económica en la adquisición de un módulo de reductor dc-dc para la alimentación eléctrica del todo el conjunto.

Por otra parte, el desarrollo de la funcionalidad de tracker y su representación en el aplicativo de control remoto, me tomó más tiempo del previsto inicialmente, introduciendo una demora adicional de 3 días respecto a la planificación.

## 1.6. Recursos utilizados

Para la elaboración del proyecto se ha dedicado los elementos hardware y software que se detallarán a continuación:

### 1.6.1. Hardware Empleado

**Plataforma de evaluación LPC1769 con MCU ARM Cortex M3 :** Se trata de una placa de desarrollo de bajo coste basada en microprocesador ARM CORTEX M3 La placa también cuenta con el módulo LPC-LINK de depuración, que en este proyecto se ha separado de la placa principal para reducir el tamaño de la placa del sistema.


Figura 1: Placa de desarrollo LPC1769 de NXP

**Modulo inalámbrico RN-171 (WIFLY):** El módulo inalámbrico RX-171 (Wifly) del fabricante Rovin Networks. Este es un sistema embebido en sí mismo que integra la pila de protocolos TCP/IP para la comunicación inalámbrica en una plataforma que permite una sencilla configuración de su hardware

Figura 2: Modulo RN-171 de Microchip

**Módulo de control de Motores** Se trata de un módulo de tamaño muy reducido que permite controlar la dirección y velocidad de dos motores de escobillas mediante el puerto UART.

Figura 3 Qik 2s9v1 Dual serial motor

**Sensor de distancia por ultrasonidos**

El Max Sonar EZ0 es un sensor de distancia por ultrasonidos de alto rendimiento

Con numerosas posibilidades de conexión y un rango de medición entre 16cm y 6 m

Figura 4 MaxSonar EZ0

**Batería 6,6V 2000mAh:**

Batería de LiFePo de mayor eficiencia muy utilizadas en el campo del radiocontrol, debido a su relación rendimiento, tamaño y peso

Figura 5 Batería LiFe

**Convertidor DC-DC Step-Down regulable:**

El convertidor regulable proporcionará alimentación adaptando la tensión de la batería para el funcionamiento de la parte lógica del vehículo

Figura 6 Convertidor DC-DC

**Motores DC de imán permanente de 4,5V**

La plataforma del prototipo cuenta con dos motores, uno que proporciona la propulsión y el otro que se encarga de la dirección

Figura 7 Motor DC

**Plataforma del prototipo:**

La plataforma utilizada para la construcción del prototipo ya cuenta con motores DC a los respectivos conjuntos de engranajes para generar tracción y dirección

Figura 9- Plataforma

**Servidor web:** El hardware sobre el que se ha instalado el servidor de la aplicación remota es un ordenador portátil Toshiba Satellite Pro con procesador Intel Core i3 M330 2.14 GHz y 8 Gb de RAM corriendo sobre sistema operativo Windows Xp Virtualizado.

### 1.6.2. Software empleado

Para el desarrollo del proyecto se ha utilizado los siguientes elementos software

- **LPCXpresso v7.4.1**: El desarrollo tanto de la aplicación como las librerías que se programan en la mota LPC1769, se ha efectuado en lenguaje C utilizando el entorno de desarrollo integrado desarrollado por el fabricante de la placa NXP. Este IDE se ha implementado basándose en el conocido entorno multiplataforma Eclipse

- **CoolTerm 1.4.3:** Este software es un emulador de terminal que permite la comunicación con de un pc por el protocolo serial.

- **Notepad++:** este software es un editor de texto con funcionalidades avanzadas para la edición de código.

- **Suite servidores Wampp:** Se trata de una suite multiplataforma que integra un servidor web Apache2, un servidor FTP y una base de datos MySql. Sobre esta plataforma se desarrolla la aplicación de control remoto del vehículo

- **VMWare Fusión para OSX:** Software utilizado para correr la máquina virtual Windows XP que aloja el servidor PHP y Apache

- **GANTT Project:** Herramienta gráfica para gestión de proyectos utilizada para elaborar los diagramas de GANTT y el seguimiento de la evolución del proyecto.

## 1.7. Producto obtenido

Por una parte se obtendrá una aplicación para la gestión remota del vehículo. Esta es una aplicación GUI básica desarrollada en lenguaje HTML y PHP que permite ser portada a prácticamente cualquier dispositivo móvil. Este interfaz de control posibilita el envío de comandos al prototipo de una manera cómoda para el usuario además de proporcionarle la información más relevante del estado del vehículo

Por otro lado se obtiene la plataforma del vehículo, capaz de establecer de manera autónoma una conexión a una red inalámbrica y ejecutar las órdenes de movimiento proporcionadas desde la aplicación de control remoto. Además se ha incorporado en la parte frontal del prototipo, un sensor de distancia que permite al prototipo calcular a qué distancia se encuentra de un objeto para evitar choques con obstáculos.

## 1.8. Resumen de los siguientes capítulos

Una vez introducidas las características generales del proyecto, es el momento de presentar brevemente cuáles serán los contenidos a abarcar en los capítulos siguientes de esta memoria.

En el capítulo 2, se da una visión del estado actual de la fase de desarrollo en el que se encuentran las áreas tecnológicas abarcadas en este proyecto, por otra parte, se presentarán algunas de las soluciones presentes actualmente en el mercado en comparación con el prototipo presentado en ésta memoria

El capítulo 3 está centrado en la descripción funcional de los elementos Hardware y Software implicados en el funcionamiento del prototipo de un modo general.

El capítulo 4 amplía la información del capítulo anterior, presentándose los detalles técnicos acerca del funcionamiento de cada subsistema en detalles en el funcionamiento.

En los capítulos 5 y 6 se discutirá la viabilidad técnica y económica del prototipo

El capítulo 7 expondré las conclusiones obtenidas del proceso de desarrollo del proyecto.

Por último en los capítulos 8,9 y 10 serán los dedicados al glosario, bibliografía y anexos

# 2. Antecedentes

Desde el considerado primer sistema embebido, desarrollado en el laboratorio del Massachusetts Institute of Tecnology para guiar el alunizaje del Apolo 11 en 1969 hasta la actualidad, los sistemas embebidos han experimentado una evolución radical, motivada por los avances conseguidos en dos áreas principalmente: La industria de la electrónica que ha permitido integrar cada vez un mayor gran número de semiconductores en circuitos cada vez más pequeños y eficientes y las redes de telecomunicaciones que ha cambiado drásticamente tanto modo como la necesidad de comunicarnos con nuestro entorno gracias a Internet.

El incremento de potencia y eficiencia energética de microprocesadores así como de otros dispositivos electrónicos unido a las posibilidades de conectividad que proporcionan además de su reducido tamaño y precio, han convertido este tipo de sistemas en soluciones ideales para el desarrollo de productos, no solo en el ámbito profesional, sino también en el ámbito doméstico, ya que en los últimos tiempos ha habido un aumento de las personas interesadas en desarrollar sus propios proyectos de robótica, seguridad, etc. En donde los sistemas empotrados todavía tienen mucho que aportar

## 2.1. Estado del Arte

En este apartado se presenta al lector un repaso del estado actual de desarrollo de las diferentes tecnologías abordadas en este proyecto:

### 2.1.1. Sistemas Operativos:

A continuación se enumerarán algunos de los sistemas operativos más utilizados en plataformas embebidas

**Windows CE:** Desarrollado por Microsoft, cuenta con soporte para plataformas x86, ARM, MIPS. Aún se puede encontrar en modelos de Smartphone, notebooks, GPS y verificadores de precios. Ocupa unos 21MB. Como inconvenientes, El software es propietario aunque el núcleo cuenta con licencia Shared Source y no cuenta actualmente con soporte.

**Embedded Linux:** Este S.O está formado por un núcleo Linux combinado con un conjunto de utilidades de software libre. En la actualidad multitud de las denominadas "Single Board Computer" (SCB) lo utilizan. Entre las principales ventajas se encuentra su reducido tamaño, ya que ocupa de media 2MB y se distribuye con licencia GPL, además cuenta con el respaldo del ELC (Embedded Linux Consortium) formado por empresas como IBM, Intel, Motorola, etc.

**FreeRTOS:** Sistema operativo en tiempo real diseñado específicamente para pequeños sistemas microcontroladores, ya que requiere entre 5 y 10KB de espacio. Muy portable ya que la mayor parte está escrito en lenguaje C y publicado como software libre. Cuenta con soporte para más de 30 arquitecturas.

### 2.1.2. Plataformas hardware:

En cuanto a las plataformas embebidas, actualmente se puede encontrar en el mercado tal variedad de plataformas que sería imposible enumerar todas en este documento, no obstante merece la pena prestar atención a las más conocidas.

Arduino: Se basa en una plataforma de electrónica abierta, que permite la construcción y evolución del hardware con total libertad, es por este motivo que se puedan encontrar en el mercado hasta quince configuraciones oficiales. El proyecto Arduino tiene el apoyo de multitud de desarrolladores en todo el mundo, ya que cuenta con un entorno de desarrollo muy intuitivo que posibilita introducirse en el mundo de los sistemas embebidos de una manera muy sencilla, este hecho además de su bajo coste, ha convertido en esta plataforma en todo un éxito en el ámbito domestico

La configuración más extendida es la modelo UNO que cuenta con un microcontrolador de 8 bits y 2KB de RAM ATmega328, proporciona 14 puertos E/S Digitales,6 E/S analógicas y PWM por un coste inferior a 25 €

Figura 8 Placa Arduino UNO rev3

Raspberry Pi: Este concepto es algo distinto al anterior, ya que el RPI es un ordenador de placa única (SCB) de bajo coste desarrollado con el fin de estimular la enseñanza de ciencias de computación en las escuelas. El diseño de incluye un SoC (system-on-a-chip) Broadcom BMC2835 que integra una CPU, GPU, DSP, USB y 512MB de memoria RAM (modelo B).

Esta placa es capaz de hacer funcionar una distribución de Linux adaptada del mismo modo que un ordenador personal y tiene un precio de 35€

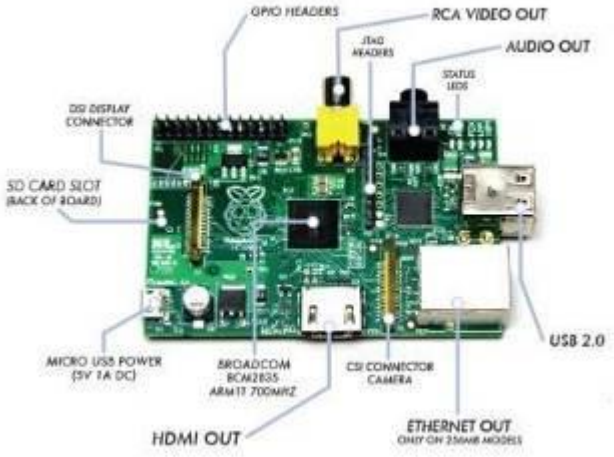



Figura 9 Placa Raspberry Pi

### 2.1.3. Módulos de comunicaciones inalámbricas:

**Módulos Bluetooth v4.0**: Trabajan a una frecuencia de 2,4Ghz Permiten establecer una Personal Área Network entre dispositivos para su comunicación inalámbrica una distancia máxima de 30 metros. Incorpora un puerto de conexión UART que facilita la configuración y gestionan mediante comandos

Figura 10 Modulo Bluetooth

**Módulos GSM/GPRS**: Funcionan mediante redes GSM, estos módulos permiten la conexión remota de dispositivos limitada únicamente a la cobertura de telefonía móvil, aunque requieren de una tarjeta SIM para poder establecer conexión

Figura 11 Modulo GSM SM5100B

## 2.2. Estudio de mercado

El mercado hay una gran variedad de productos de características similares al presentado en esta memoria, incluso incorporando más funcionalidades. Aunque inicialmente el propósito para el que fueron diseñados este tipo de vehículos dentro de un ámbito profesional para la exploración y prospección de entornos. Este tipo de productos se han extendido en el ámbito doméstico y lúdico.

A continuación se presentan algunos ejemplos de vehículos para entornos doméstico que se pueden encontrar en el mercado y posteriormente se presentará algún producto a mayor escala para usos profesionales

**Rovospy:** Este producto consiste en un vehículo controlable mediante un Smartphone vía WIFI. Cuenta con una cámara que permite transmitir imágenes con una resolución de 640x480 y sonido en tiempo real. Dispone de iluminación led y visión nocturna.

El vehículo establece un enlace punto a punto con el dispositivo de control remoto (Smartphone) con un alcance de unos 60 metros. Este producto se puede conseguir por un precio inferior a 150€

Figura 12 RovoSpy

**Cloud Rover iSpy**: Este es un prototipo parecido pero en formato de 4 ruedas, con funcionalidades similares al anterior, cámara de 640x480 ajustable de 0 a 40º en vertical. Su precio aproximado es de unos 100€

Figura 13 CloudRover iSpy

**Inspectorbots 4WD FPV Spider Mite Robot:** Este vehículo está especialmente diseñado para vigilancia, y prospección. Cuenta con unos motores de alto par motor que le proporcionan potencia para superar pendientes con ángulos de 40º. Dispone de cámara con visión nocturna y una capacidad de arrastre de


Figura 14 Spider Mite de Inspectorbots

**Inspectorbots 4WD HD:** Este producto está pensado para remolcar otros vehículos o para transportar personas accidentadas ya que tiene una potencia de arrastre de uno 1400Kg

Figura 15 Plataforma 4WD de Inspectorbots

# 3. Descripción funcional

## 3.1. Vehículo IP Liviano Multipropósito Autopropulsado

El sistema está compuesto por:

- Una plataforma dotada de cuatro ruedas que incorpora los motores y mecanismos de engranajes que posibilitan su movimiento y soporta el resto de los elementos que forman el prototipo.

- La unidad de control del sistema que integra todos los dispositivos sensores y actuadores necesarios para el funcionamiento del vehículo y sus interconexiones, así como la electrónica encargada de regular el suministro de energía de todos los elementos.

- Una aplicación web ubicada en un servidor remoto que permite el control y la monitorización del estado del vehículo través de una red TCP/IP

- En la memoria flash del MCU, se encuentran sistema operativo FreeRTOS, librerías desarrolladas para la gestión de los diferentes actuadores/sensores así como la capa de abstracción de Hardware (HAL) para procesadores ARM Cortex CMSIS. Estos elementos proporcionan funcionalidades y drivers necesarios para el funcionamiento de aplicación principal que implementa el comportamiento del prototipo.

### 3.1.1. Descripción mecánica de la plataforma

La plataforma sobre la que se ha construido el prototipo es el cuerpo de un coche de radiocontrol al que se le han efectuado las modificaciones oportunas con el fin de adaptar su funcionamiento al producto que se pretendía desarrollar. El coche incorpora dos motores eléctricos de imán permanente, uno de los motores proporciona el giro al eje trasero del coche para mover las dos ruedas traseras. El otro motor ataca a un grupo de engranajes que mueven las ruedas delanteras hacia la derecha o la izquierda

Figura 17 Sistema de dirección

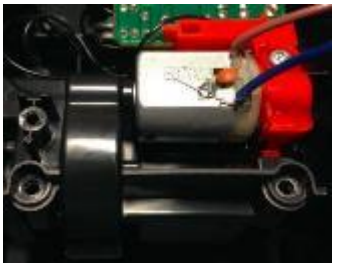


Figura 17 Detalle del motor de tracción

### 3.1.2. Diagrama de Bloques del sistema

En la figura 10 se representa a modo esquemático el funcionamiento general de la aplicación. Por un lado el prototipo se encuentra ubicado en un emplazamiento y dentro del alcance de señal inalámbrica del punto de acceso WIFI que previamente habrá configurado. Por otra parte, el usuario puede interactuar con la aplicación de control remoto en otra localización distinta a la del prototipo. Al insertar un nuevo comando en la aplicación remota, el módulo servidor integrado en la aplicación que ha sido informado de la dirección IP y el puerto de escucha de conexiones de la plataforma, establece una conexión (socket) mediante el protocolo TCP a través de la cual introduce el comando seleccionado. El prototipo recibe el comando y lo procesa de la manera adecuada para su ejecución.

Entrando en un mayor nivel de detalle, el funcionamiento del conjunto del sistema, es el que se presenta con en la figura 11

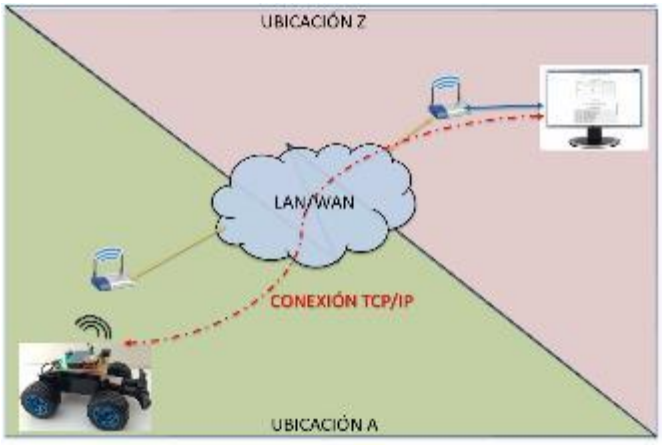



Figura 18 Esquema general de funcionamiento del sistema

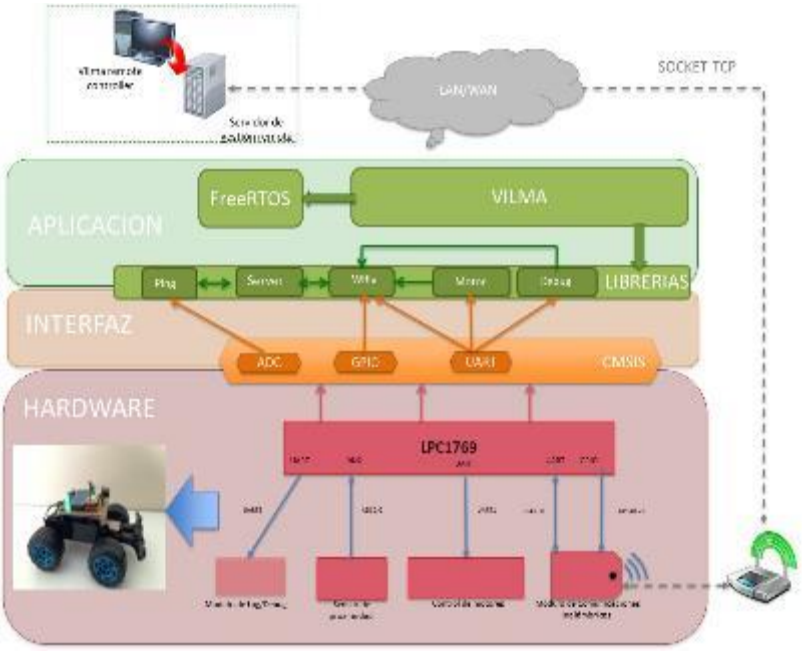



Figura 19 Diagrama de bloques del sistema VILMA

La capa Hardware del sistema, está formada por los módulos electrónicos conectados a través de diferentes interfaces a la placa de control principal LPC1769, control de motores, sensor de proximidad y módulo de comunicaciones. El módulo de comunicaciones es el dispositivo que posibilita la interacción del prototipo con el mundo exterior.

La capa Middleware o de Interfaz: En esta capa se encuentra la capa de abstracción de hardware (HAL) CMSIS que proporciona herramientas para manejar los recursos del microcontrolador

La capa aplicación: En esta capa se encuentra el programa principal de la aplicación que modela el comportamiento del prototipo y el sistema operativo FreeRTOS, que proporciona herramientas para la gestión de tareas, prioridades, semáforos y colas.

Por último, entre la capa de aplicación y la de interfaz se ubica las librerías y drivers que gestionan los módulos hardware conectados a la placa LPC1769.

### 3.1.3. Descripción del protocolo de comunicaciones de VILMA

La comunicación entre la aplicación de control y la plataforma, se lleva a cabo mediante el protocolo Transport Control Protocol. (TCP). Se ha preferido la utilización de éste protocolo frente a su alternativa UDP porque a diferencia de éste, TCP es un protocolo orientado a conexión, de modo que para que dos dispositivos puedan intercambiarse mensajes, primero ha tenido que negociarse y establecerse un canal de comunicación entre ellos. Además TCP implementa otros mecanismos como el control de flujo numeración de secuencia de tramas y la retransmisión de paquetes.

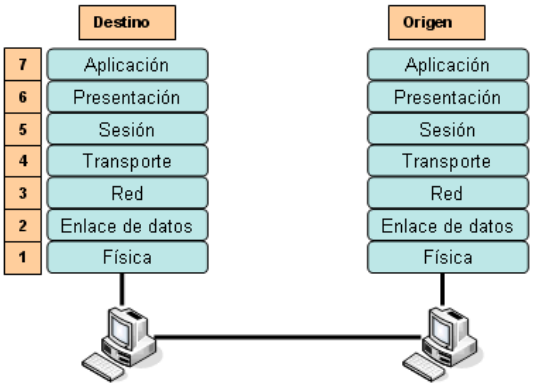



Figura 20- Modelo de capas OSI

La comunicación entre dispositivos mediante TCP se establece con el denominado "negociación en tres pasos" representada en la figura 13, la finalización también puede darse en tres pasos aunque no es obligatorio

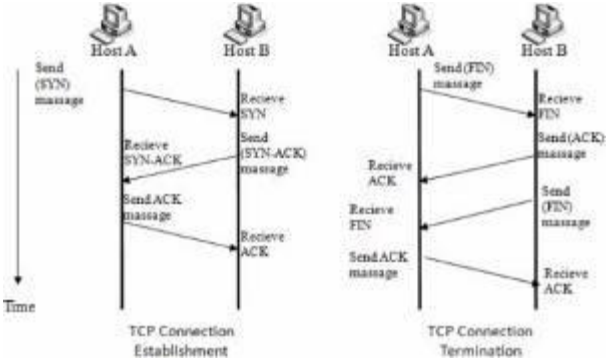



Figura 21 Establecimiento y cierre de conexión TCP

Al solicitar el envío de un comando, la aplicación de control establece la conexión, envía el mensaje a la plataforma y cierra el canal de comunicación. Por su parte, el prototipo envía realiza las mismas operaciones al enviar la información de su estado a la aplicación de control. En ambos casos la secuencia es la representada en la figura 14

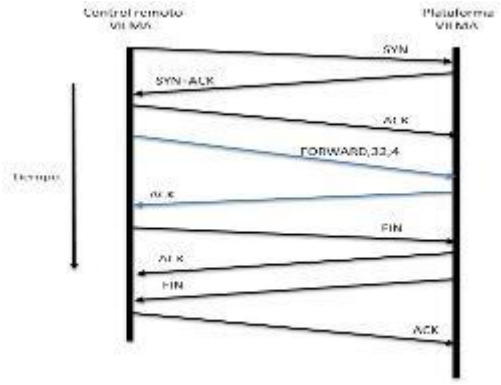



Figura 22 Envío de comando remoto

### 3.1.4. Comunicaciones con la unidad de control principal

A continuación se especifica el modo en el que cada uno de los subsistemas se conecta al sistema de control principal LPC1769 de acuerdo con la figura 15

- Módulo de comunicaciones Wifly, se dialoga con el control principal mediante conexión UART seleccionable, por defecto se ha elegido el puerto UART0, el baudrate es configurable.
- Driver de motor se comunica con la placa LPC1769 también mediante puerto UART, para esta conexión se ha seleccionado el UART1. El módulo de control de motores detecta automáticamente el baudrate, no obstante la velocidad máxima soportada para la comunicación serie es de 38400 bps.
- Sensor de proximidad, en este caso el sensor se conecta al sistema principal mediante un puerto ADC0.0 para obtención del valor de distancia escalada. La resolución del conversor ADC en la placa LPC1769 es de 12 bits

- Puerto para depuración: Este módulo solo se utiliza para propósitos de depuración y no se utiliza en modo de funcionamiento normal, no obstante es posible la conexión de un adaptador USB/UART al puerto 3 de la placa 1769 para revisar la información proporcionada por el sistema.

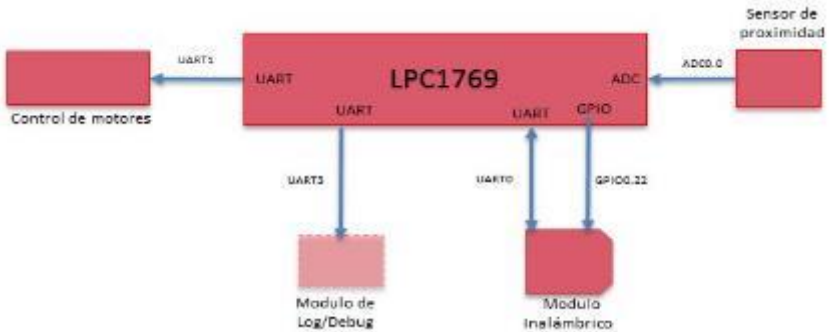



Figura 23-Interconexión entre módulos del sistema

## 3.2. Diseño de la Interfaz de usuario

La aplicación para el control remoto del prototipo se ha diseñado un entorno gráfico basado en página web basado en formularios y programado en lenguaje PHP, que corre en una máquina servidora.

Este entorno es muy básico y su objetivo es el envío de comandos al prototipo así como su representación de una forma cómoda y funcional para el usuario, entendiéndose que el desarrollo de una aplicación de entorno gráfico de aspecto profesional, excede el alcance de este proyecto.

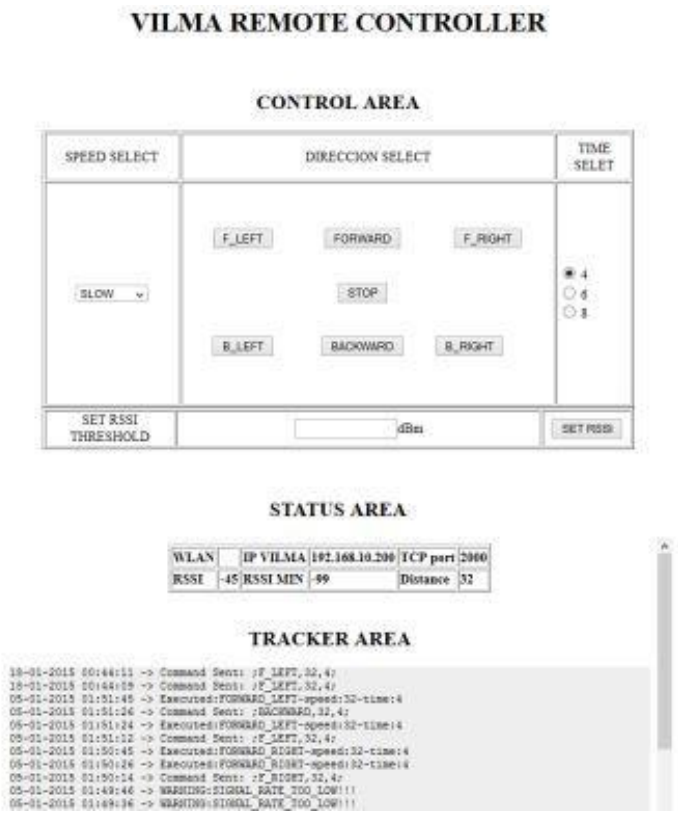



Figura 24 Captura de pantalla del entorno GUI

La aplicación de control remoto está formada por tres áreas de información como se puede apreciar en la figura 16.

**Control Área**: Donde se indica el movimiento a ejecutar, así como la velocidad y duración del desplazamiento. La aplicación establece un socket con el módulo de comunicaciones del prototipo, envía el comando y posteriormente cierra el socket

**Status Área**: Que representa la información proporcionada por el prototipo. El módulo de comunicaciones genera cadenas de mensajes de tipo variable=valor y las envía al recurso status.php almacenado en el servidor mediante un método http GET. El servidor extrae los valores de las variables y las representa en el área de estado de la aplicación. A continuación se muestra un ejemplo de mensaje enviado por el prototipo a la aplicación remota:

**Tracker Área**: Donde se presenta la información a modo de log de los movimientos ordenados desde la aplicación, así como mensajes recibidos desde el prototipo, una vez se han ejecutado los comandos o en caso de producirse algún evento destacable. Tras la ejecución de un comando, el prototipo envía un mensaje empleando el mismo mecanismo que el caso del área de estado. La aplicación incluye un timestamp y representa la información recibida.

## 3.3. Diseño de la Aplicación Embebida

La aplicación que ejecuta el sistema embebido, se puede resumir del modo que se presenta en el diagrama de boques de la figura 15

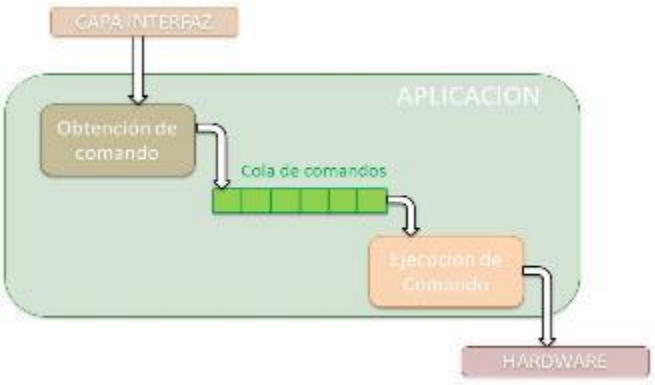



Figura 25 Diagrama de bloques de la aplicación

En líneas generales, el funcionamiento de la aplicación embebida es el de un sistema con una tarea productor de datos, en este caso comandos de movimiento y otra tarea consumidora que se encarga del procesado y ejecución de los mismos, controlando los elementos que actúan sobre el hardware.

La comunicación entre ambas tareas se lleva a cabo a través de una cola FIFO. Cada vez que la tarea productora recibe un dato lo introduce en la cola, por su parte la tarea consumidora es desbloqueada inmediatamente para extraerlo y proceder con la ejecución de la instrucción.

## 3.4. Suministro de potencia de la plataforma

Los elementos que proporcionan energía a todo el prototipo son básicamente dos:

En primer lugar se cuenta con una batería de $LiFePo_4$ de 6,6V de reducido tamaño y peso capaz de entregar 12,5Wh. Esta batería se encargará de proporcionar energía a los motores, además de alimentación a todo el conjunto electrónico.

Para poder adaptar la tensión de la batería a los distintos módulos electrónicos, se ha incorporado un conversor DC-DC step-down que se encarga regular proporcionada por la batería hasta 3,3V, ya que tanto la placa LPC1769 como el resto de módulos electrónicos funcionan a esta tensión.

Los motores del vehículo se alimentan directamente a la tensión de salida, mediante los pines habilitados en el driver de motor a tal efecto.

En la figura 18 se puede observar el modo en el que se proporciona alimentación a los diferentes módulos del sistema

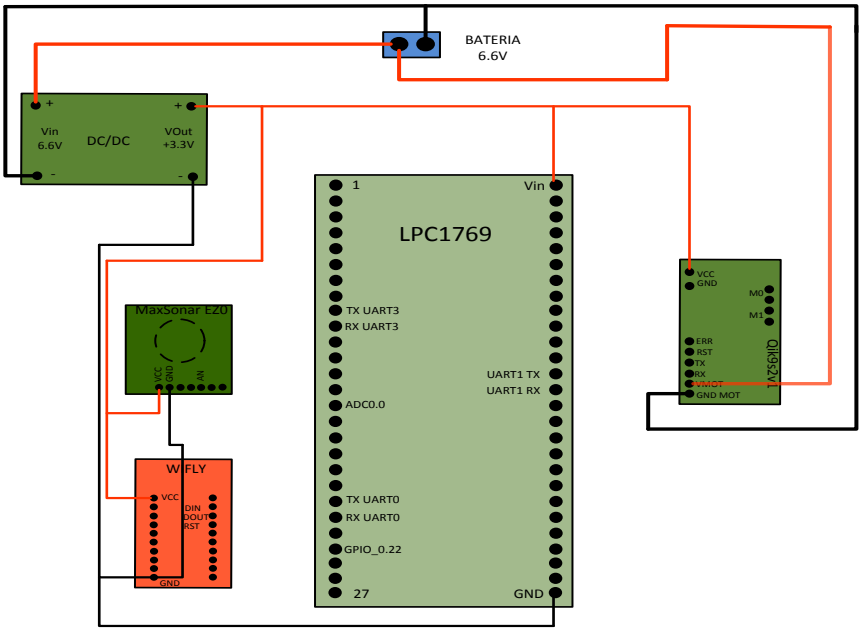



Figura 26 Esquema de conexiones eléctrica del prototipo

# 4. Descripción detallada del sistema

A continuación se aborda el detalle de funcionamiento general desde un punto de vista esquemático, para posteriormente profundizar en las tareas más relevantes llevadas a cabo por estas unidades de software.

Por último, se profundizará sobre los elementos Hardware empleados en el proyecto.

## 4.1. Funcionamiento global del sistema

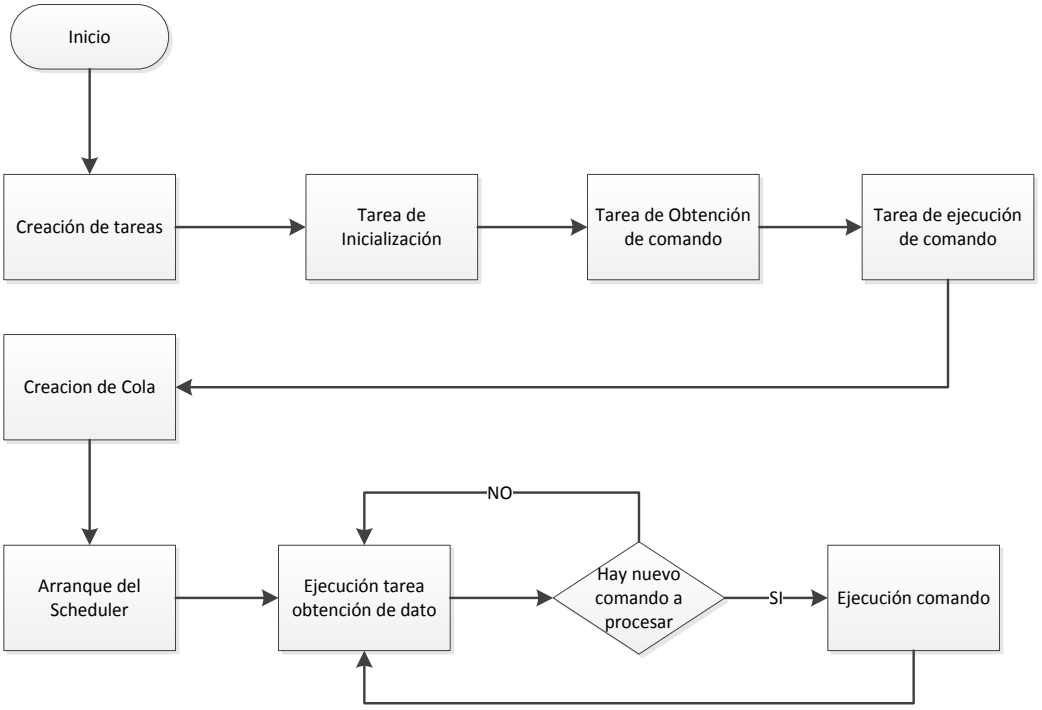



Figura 27 Esquema de funcionamiento general

El sistema operativo freeRTOS sobre el que gestiona el funcionamiento de la aplicación embebida permite la gestión de tareas o hilos de ejecución con la capacidad de asignación de prioridades de un modo flexible. Además permite el manejo de colas, que permite la comunicación de información entre las diversas tareas en ejecución. La aplicación principal utiliza entre otros estos dos recursos indicados anteriormente.

En la figura 19 se puede observar el diagrama de flujo de la aplicación. La función principal de la aplicación está basada en 3 tareas.  Inicialización, obtención de comandos y ejecución de comandos.

Una vez se han iniciado las tareas que participan en el funcionamiento de la aplicación, se crea una cola que, como se ha comentado con anterioridad, será utilizará para pasar comandos desde la tarea de obtención de datos hasta la tarea de ejecución. Por último se inicia el planificador que es el encargado de gestionar el orden de ejecución de las tareas, en función de la prioridad asignada a cada una de ellas.

A continuación se presentan las funciones que realiza cada una de estas tareas

## Tarea de Inicialización

Esta tarea es la que se ejecuta con mayor prioridad y se encarga de iniciar todos los sensores y actuadores para el funcionamiento de la plataforma. Además, proporciona los datos necesarios al módulo de comunicaciones para establecer una conexión inalámbrica al punto de acceso configurado.

También se encarga de la configuración de los datos sobre el servidor remota así como de la configuración del módulo de comunicaciones con el puerto de escucha para las peticiones de conexión TCP.

Una de las primeras acciones que se lleva a cabo en la tarea, es bloquear la ejecución de las otras dos. El motivo de esta acción es la de forzar a que la tarea de inicialización se ejecute de inicio a fin, ya que de lo contrario, el planificador forzaría saltos de contexto al resto de las tareas en caso de algún evento de pausa o retardo en la ejecución.

Las dos últimas acciones llevadas a cabo por esta tarea son la de desbloquear la siguiente tarea con mayor prioridad, la de obtención de datos y finalmente la tarea se inicialización se destruye así misma para liberar recursos de memoria y tiempo de ejecución. En la figura 20 se puede observar el diagrama de flujo de esta tarea

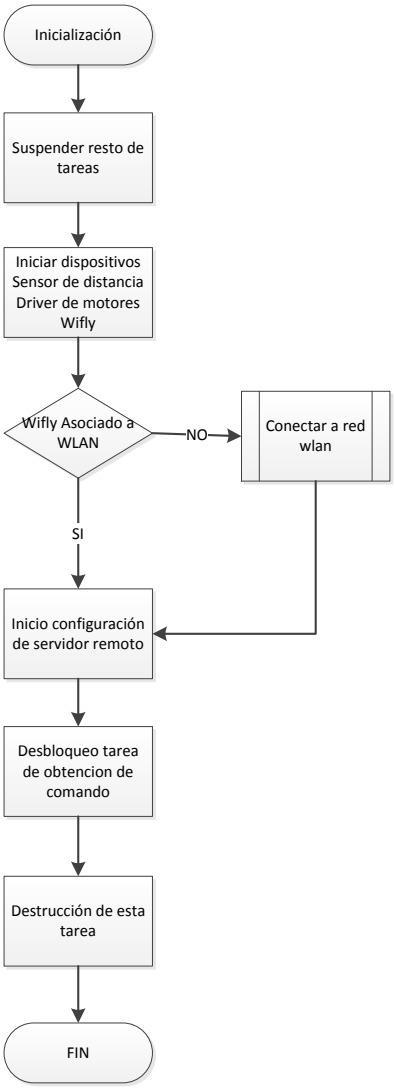



Figura 28 Diagrama de flujo de la tarea de inicialización del sistema

## Tarea de Obtención de comandos

Esta tarea se encarga de recibir los comandos enviados por la aplicación remota. Esta tarea se ejecutará con mayor prioridad durante todo el ciclo de vida del programa manteniéndose en escucha para introducir en la cola de datos el comando recibido a través del socket.

La figura 21 muestra el diagrama de flujo de la tarea.

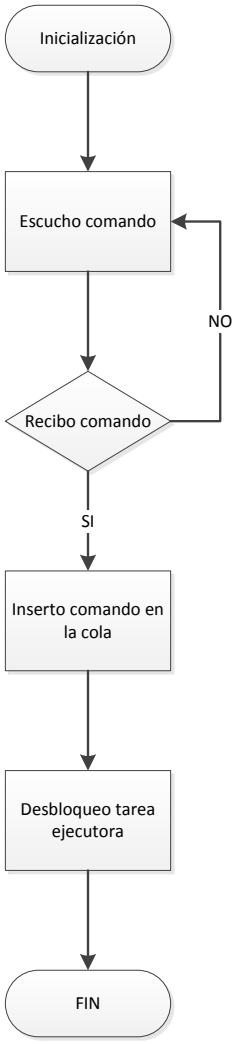



Figura 29 Diagrama de flujo de la tarea de obtención de comando

## Tarea de ejecución de comandos

Esta tarea se encarga procesar el comando obtenido en la cola de mensajes, una vez procesado, evalúa el comando a ejecutar y estima la posibilidad de ejecución, en función de si se cumplen o no los criterios de señal inalámbrica mínima y distancia respecto a obstáculo.

Uno de los comandos que se pueden ordenar desde el aplicativo remoto es el de fijar un nuevo umbral inalámbrico. Durante la ejecución de una instrucción de movimiento, el sistema evalúa si el nivel de señal detectado por la plataforma permite su desplazamiento sin riesgo de perder la cobertura y por tanto el control del prototipo. En caso de existir riesgo de pérdida de control remoto, la aplicación remota consulta el sistema de tracking que contiene los últimos movimientos de la plataforma y deshace el último movimiento efectuado con el fin de recuperar nuevamente la señal inalámbrica

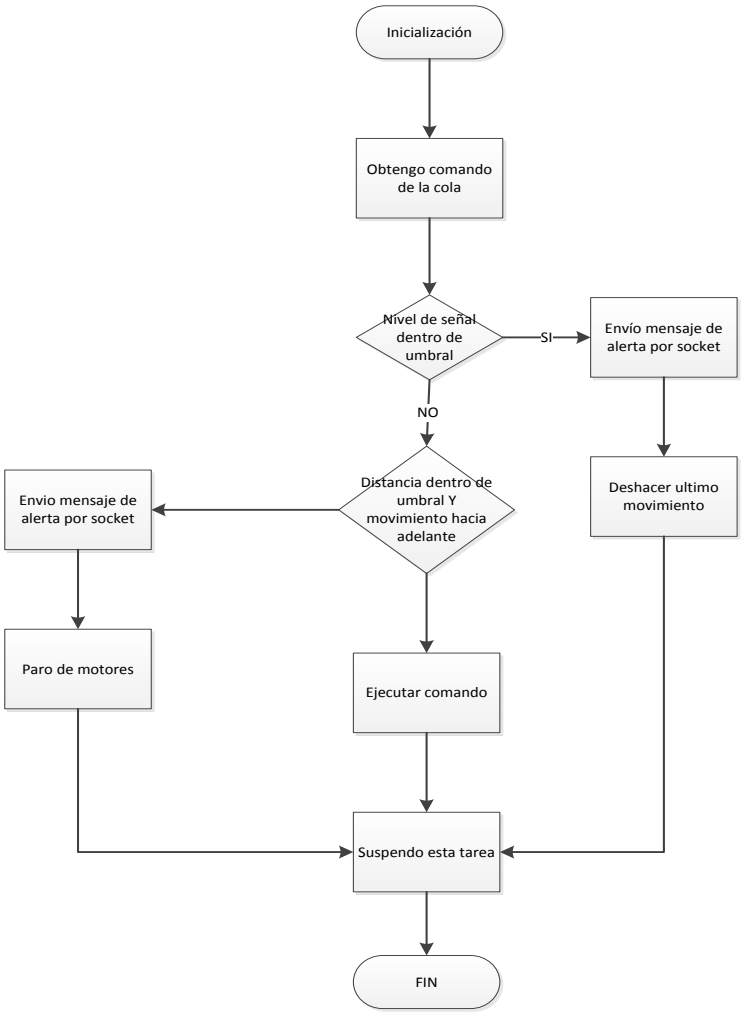



Figura 30 Diagrama de flujo de la tarea de ejecución

## 4.2. Aplicación de control remoto

La aplicación para el control remoto del prototipo está programada en lenguaje PHP y su funcionamiento, como se puede apreciar en el diagrama de la figura 23 es muy básico.

Al ordenarse ejecución de un comando, la aplicación genera un mensaje del tipo

Que es enviado a través de la conexión TCP para que el prototipo ejecute el comando. Por su parte, el prototipo envía información de su estado y alertas relacionadas con umbrales de funcionamiento. Además este mecanismo de comunicación es el utilizado para el envío de información para su representación en área del tracker de la aplicación remota.

El formato del mensaje enviado por la plataforma es el que se presenta a continuación

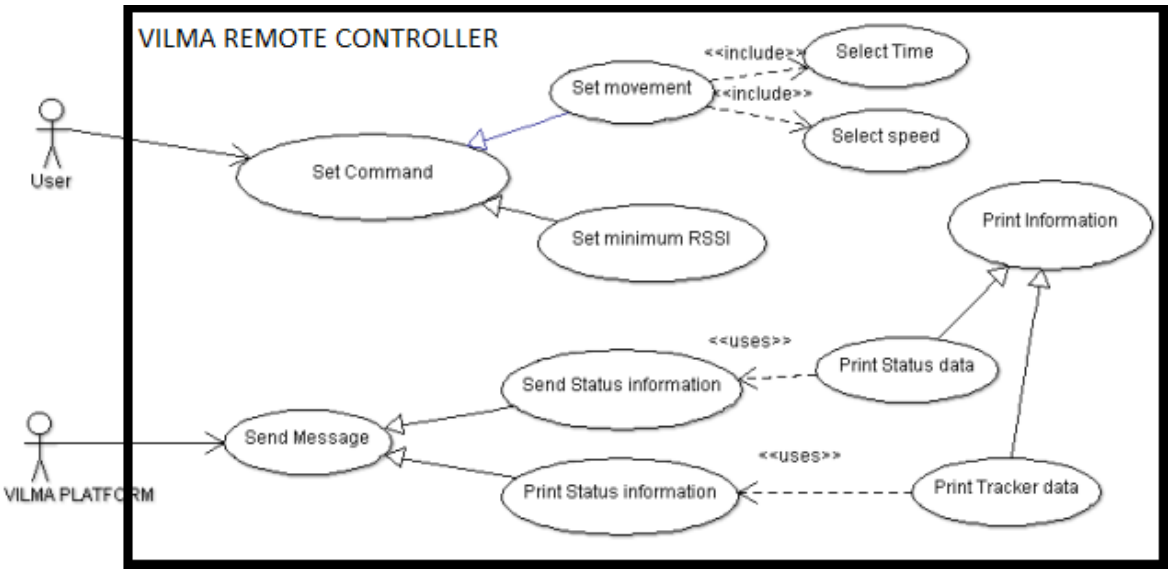



Figura 31 Diagrama de casos de uso de la aplicación par control remoto

Los mensajes desde el prototipo a la aplicación se envían mediante un método HTTP GET. El fichero de aplicación remota status.php es el encargado de obtener los mensajes recibidos por éste método y genera la información en el fichero status_values.php que contienen las variables y los valores recibidos.

Por último está el fichero show status lee la información de status_values y representa los valores obtenidos en el área de notificación o tracker según corresponda.

## 4.3. Hardware

### 4.3.1. Sistema de control principal

La placa de desarrollo LPC1769 se basa en microcontrolador para aplicaciones embebidas caracterizado por un alto nivel de integración y muy bajo consumo, entre sus características más importantes destacan:

- Opera a una tensión entre -0.5 y 4.5V
- Procesador Cortex M3 de 32 bits hasta 120Mhz
- Memoria RAM de 64kB y Memoria Flash de 256kB
- 2 Bloques de memoria SRAM de 16kB
- USB 2.0 con controlador DMA dedicado
- 4 Puertos UART
- 3 interfaces de Bus $I^2C$
- Conversor ADC de 12 bits y Conversor DAC de 10bits
- Generador de PWM (Pulse Width Modulation)

A continuación se presenta el esquema de conexionado de la placa LPC1769

| LPC1769 | | |
|---|---|---|
| **Pin** | **Nombre** | **Conexión** |
| 9 | UART3 TX | Pin debug |
| 10 | UART3 RX | NC |
| 15 | ADC0.0 | distancia out |
| 21 | UART 0 TX | Wifly Rx |
| 22 | UART 0 RX | Wifly TX |
| 24 | GPIO 0.22 | Wifly Reset |
| 28 | Vin (+3.3V) | DC/DC |
| 40 | UART1 TX | Motor RX |
| 41 | UART1 RX | NC |
| 54 | GND | GND |

Tabla 1 Conexiones LPC1769

El diagrama de bloques de la figura 19 permite hacerse una idea de la flexibilidad y potencia que ofrece esta plataforma

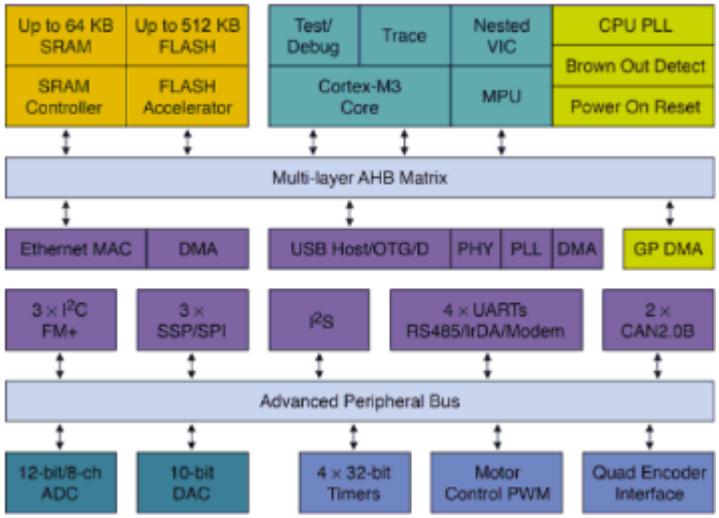



Figura 32 Diagrama de bloques LPC1769

Durante la fase previa, y también posteriormente en la fase de realización del proyecto se ha efectuado un estudio de las características de la placa, su funcionamiento y la configuración de los elementos a utilizar en este proyecto. Cabe destacar que en el propio instalador del entorno de desarrollo, denominado LPCXpresso, se ofrece una serie de programas de ejemplo para su estudio y comprensión dentro de la carpeta denominada *Examples* cuya ubicación depende de la plataforma donde sea instalado el mencionado IDE.
Además, el fabricante ofrece en internet páginas y foros de usuarios donde se comparte ejemplos así como información útil para la configuración del sistema y periféricos.

### 4.3.2. Sistema de comunicaciones

El módulo encargado de las comunicaciones es el sistema RN-170 de la compañía Microchip (anteriormente Rovin Networks). Como se mención en el apartado 1.7 se trata de un sistema empotrado en sí mismo. Entre sus características destacan:

- Opera a un tensión de 3,3 Vcc
- Muy bajo consumo 4uA
- Señal de transmisión configurable entre 0 dBm y +12dBm
- Bajo consumo:35mA en operación 4uA en reposo
- Incluye 14 GPIO pines (4 se comparten con el puerto UART)
- 8 entradas analógicas de 14 bits de resolución
- Aplicaciones integradas como ICMP, cliente HTTP, servidor y cliente TCP, servidor y cliente DHCP, cliente FTP
- Soporta configuración WI-FI protected setup (WPS)

El diagrama de bloques del sistema inalámbrico se presenta a continuación


Figura 33 Esquema de módulos del

Las conexiones físicas de este módulo empleadas en el proyecto son las que se indican a continuación

| PIN | USO | DETALLES |
|---|---|---|
| 1 | VIN | 3,3Vcc |
| 2 | UART IN | |
| 3 | UART OUT | |
| 5 | RESET | Ton>=160us |
| 10 | GND | |

Tabla 2 Pin-out módulo Wifly

El sistema Wifly cuenta con un desarrollado API que permite su configuración de un modo muy flexible. Una vez configurados los parámetros mínimos para la conexión a una red inalámbrica, tales como en nombre del punto de acceso (SSID) y la contraseña para la

encriptación, el dispositivo se encuentra en disposición de asociarse a una red inalámbrica de manera automática cada vez que éste es alimentado.

Una particularidad del dispositivo es que incluye un servicio de SoftAp, estableciéndose como punto de acceso inalámbrico al que cualquier dispositivo puede conectarse. En este modo, el sistema también actuar como servidor DHCP proporcionando direccionamiento IP y permitiendo iniciar un entorno web desde el que el dispositivo muestra las redes inalámbricas cercanas y permite realizar la configuración inicial de conexión a una red inalámbrica nueva. Lo que dota al sistema de una gran portabilidad.

En el apartado de bibliográfica se puede encontrar el enlace al documento de referencia de comandos donde se explica el proceso para habilitar esta funcionalidad.

### 4.3.3. Sistema controlador de motores

El módulo de control de motores está basado en el circuito Toshiba TB6612FNG que integra un doble puente en H para control de motores. Este circuito integrado, es capaz de suministrar hasta 1 amperio por motor en modo continuo y hasta 3 A de pico. El sistema cuenta con led de estado que indica la presencia de un error en la configuración o comando enviado

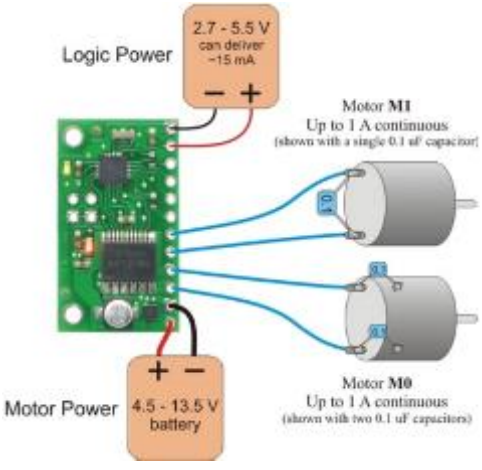



Figura 35 Esquema de conexión de Qik 2v9s1

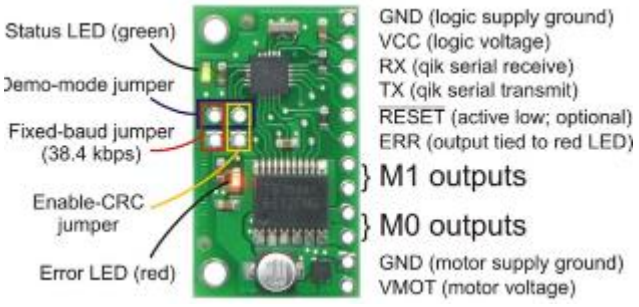



Figura 34- Detalle del pin-out del driver de motores

### 4.3.4. Sensor de proximidad Maxsonar EZ0

Se trata de un sensor de ultrasonidos capaz de medir distancias y detectar objetos entre 15,2cm (6'') y 6 metros. Funciona a una frecuencia de 42Khz y un consumo de apenas 2mA. El sensor permite la función cooperativa con otros sensores de proximidad para realizar medidas en cascada. Permite medir distancias empleando los puertos serie, mediante PWM. En este proyecto el sensor funcionará en modo analógico proporcionando un valor que cumple con la fórmula VCC/512, 98mV/pulgada con una alimentación de 5V. El valor en centímetros para esa tensión de alimentación es de aproximadamente 38,58 mv/cm

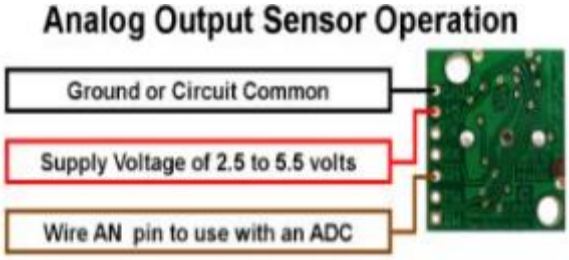



Figura 36- Detalle conexión del sensor de distancia por ultrasonidos

## 4.4. Interconexión de los elementos del sistema

En la figura 24 se muestra el modo en el que los módulos hardware presentados en los capítulos anteriores se encuentran interconectados en la placa de circuito impreso.

El módulo depurador de la tarjeta LPC1769 se ha separado del módulo microcontrolador, denominado target, de este modo se obtiene una placa algo más reducida que utilizando la placa de desarrollo al completo.

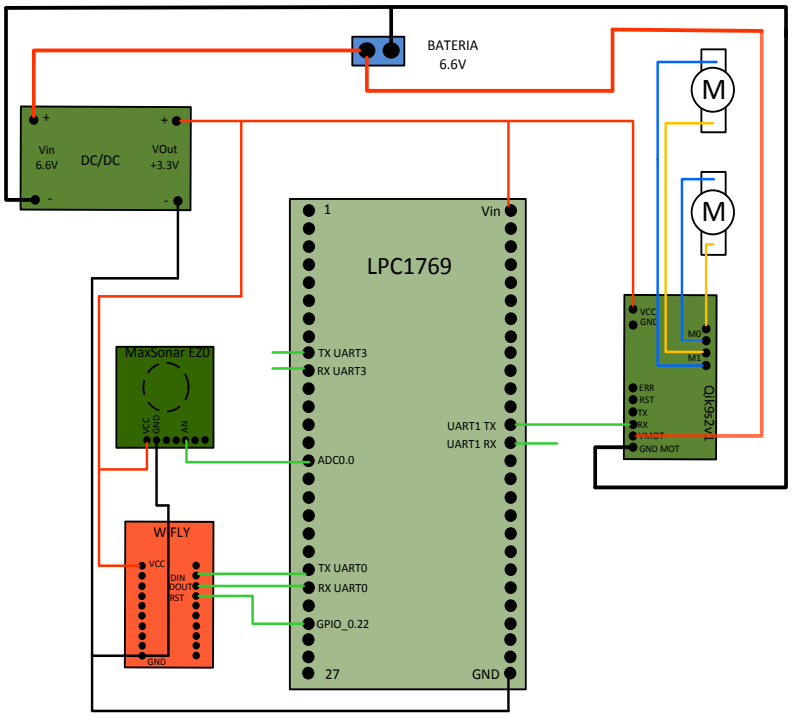



Figura 37 Esquema de interconexión entre módulos

.El puerto UART3 se utiliza como módulo de depuración de errores enviando información sobre las tareas que está realizando el sistema, por lo que no se ha conectado el pin de recepción a ningún punto para su utilización.

El caso de del puerto UART1 es similar ya que se encuentra conectado al driver de motor y aunque este dispositivo envía códigos informativos, no se ha estimado de utilidad la obtención de estos para su tratamiento, de modo que el pin de recepción tampoco se ha conectado.

# 5. Viabilidad Técnica y económica

Una vez finalizado el proyecto, se efectúa una valoración económica de materiales y el coste aproximado de precios de desarrollo. Tablas 3 y 4

| Elemento | Cantidad | Precio unidad | Precio total |
|---|---|---|---|
| LPC 1769 Board | 1 | 23,80 | 23,80 |
| RN-XV Wifly Module | 1 | 30,83 | 30,83 |
| Qik2s9v1 Motor driver | 1 | 24.95 | 24.95 |
| Sensor de distancia por ultrasonidos | 1 | 25.79 | 25.79 |
| Battery Pack LiFE 6,6V 2000ma/A | 1 | 37.99 | 37.99 |
| Resistencias ¼ E24 varios valores | 10 | 0,04 | 0,40 |
| Conversor DC-DC | 1 | 5.60 | 5.60 |
| Interruptor placa C.I | 1 | 0,37 | 0,37 |
| diodo led verde A.L | 1 | 0,67 | 0,67 |
| Placa CI topos | 1 | 5.30 | 5.30 |
| Gastos de envío | 2 | 20,00 | 40,00 |
| TOTAL MATERIALES | | | 195.50€ |

Tabla 3 Valoración económica de materiales

| Elemento | Cantidad | Precio unidad | Precio total |
|---|---|---|---|
| Ingeniería y documentación | 20 | 60 | 1200 |
| Horas desarrollo | 290 | 20 | 5800 |
| Puesta en marcha | 25 | 25 | 625 |
| TOTAL | | | 7625 |

Tabla 4 Valoración económica de dedicación

# 6. Consideraciones finales

Como cierre de esta memoria se discutirá a continuación las conclusiones que se han obtenido del desarrollo de este proyecto

## 6.1. Conclusiones

A continuación se enumeran los objetivos fijados al inicio del proyecto en el apartado 1.3 de esta memoria así como sus conclusiones.

En el apartado de objetivos principales destacaban se han conseguido todos los objetivos propuestos:

- **Dotar al sistema de comunicación inalámbrica sobre el protocolo 802.11 b/g**: Este objetivo se ha cumplido totalmente, llegando a desarrollar una librería funcional para el módulo Wifly (RN171V) que abarca la práctica totalidad de funcionalidades.

- **Crear un sistema de comunicaciones basado en el modelo cliente-servidor para el control del vehículo:** Se ha implementado un mecanismo de comunicación en el que tanto el aplicativo de control remoto, como el prototipo se intercambian roles de cliente y servidor. Si bien es cierto que la comunicación podría haberse implementado mediante un solo socket que permaneciera abierto constantemente, el resultado de la comunicación es fluido y no presenta retrasos, al menos en el entorno de pruebas basado en una red LAN

- **Implementar un interfaz GUI básico para el manejo control del vehículo**: Se ha desarrollado un sistema muy básico pero que cumple sobradamente su cometido de enviar comandos al prototipo y presentación de información acerca del mismo ampliándose también uno de los objetivos secundarios de representar información del estado del prototipo así como log de últimos movimientos.

- **Ofrecer sistema de control de movimiento hacia delante y hacia atrás, así como hacia derecha e izquierda:** Este objetivo se ha cumplido en su totalidad

- **Desarrollar un sistema anti-colisión que detenga el vehículo en caso de detectar un obstáculo**. Esta funcionalidad se ha conseguido desarrollando una librería para la gestión del sensor de proximidad.

Referente a los objetivos secundarios, solo ha quedado pendiente de completar la funcionalidad de recuperación de cobertura autónoma que no se ha llevado a cabo por falta de tiempo.

## 6.2. Propuestas de Mejora

Son muchas las mejoras que proporcionarían un valor añadido al producto que se ha presentado.

En primer lugar el control del módulo de comunicaciones a nivel de programación es bastante complicado conseguir sincronizar sus respuestas con las del resto del sistema, y se ha tenido que realizar el ajuste a base de prueba-error, este inconveniente provoca que en ocasiones una en una respuesta no capturada correctamente por otro módulo funcional puede producir inestabilidades en el sistema.

Sería deseable la implementación de una aplicación de control remoto de un aspecto profesional como las existentes en el mercado actualmente.

Sería interesante añadir sensores de distancia en la parte trasera de la plataforma, para intentar evitar colisiones en desplazamientos marcha atrás. Además de que ambos sensores, delantero y trasero podrían moverse mediante plataformas pan and tilt controladas por servomotores para implementar un sistema anticolisión con búsqueda de ruta alternativa.

Por último y ya que en el mercado existen estén plataformas que incorporan cámaras IP desde las que se pueden obtener imágenes y audio tomadas desde el vehículo lo que proporcionaría un alto valor añadido al prototipo.

## 6.3. Autoevaluación

El motivo principal por el que me interesé por esta área de proyecto fue porque abarca dos campos que realmente me apasionan como son la electrónica y la informática.

A pesar de que a priori no contaba con experiencia en microcontroladores ni mucho menos Sistemas embebidos, creo la elaboración de este proyecto me ha proporcionado una buena base para continuar investigando y aprendiendo en este campo.

Por otra parte, he de reconocer que el camino no ha sido nada sencillo, a mi falta de experiencia en programación, se ha unido mi total desconocimiento del lenguaje de programación C, de modo que he tenido que ir aprendiendo no solo a programar los aspectos propios de los sistemas embebidos si también éste lenguaje de programación. Estas carencias las he tenido suplir con muchas horas de trabajo, pruebas, búsquedas y en algún momento frustraciones. No obstante, también es justo señalar que el hecho de afrontar estas debilidades me ha proporcionado habilidades que denomino “transversales” que seguro me serán muy útiles en mi vida profesional.

Valoro la experiencia durante el desarrollo de este proyecto como muy enriquecedora, tanto en el ámbito docente, que me ha descubierto un mundo que realmente me ha “enganchado” como en el aspecto de crecimiento personal, ya que en ocasiones me he visto obligado a sacar lo mejor de mí mismo para superar ciertas adversidades durante la elaboración del proyecto.

Mi nota para esta experiencia es un Notable alto.

# 7. Glosario

***8***

802.11g
Estandar que define el uso de las capas física y enlace de datos de la arquitectura OSI especificando sus normas de funcionamiento en una WLAN 4

***B***

Bluetooth
Especificación industrial para redes inalámbricas de área personal que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace de radiofrecuencia 9, 25

bps
Bits por segundo 32

***C***

CMSIS
Cortex Microcontroller Software Interface Standard 12, 28, 30

CPU
Unidad central de proceso 24

***D***

DSP
Procesador digital de señales 24

***F***

FIFO
First in first out 35

***G***

GPIO
Generic Purpose Input/Output 12, 44

GPL
Generic Public License 8, 23

GPRS
Servicio general de paquetes via radio 25

GPU
Unidad de procesamiento gráfico 24

GSM
Sistema global de comunicaciones móviles 25

GUI
Graphical User Interface 6, 11, 21, 34, 49

***H***

HotSpot
Punto de acceso que ofrece acceso a una red de manera inlámbrica 10

***L***

LAN
Local Area Network 9, 49

led

# 8. Bibliografía

Using the FreeRTOS Real Time Kernel - A Practical Guide de Richard Barry

FreeRTOS Reference Manual Varios Autores Real Time Engineers Ltd.

Programación en C/C++ Manuel Alfonseca Moreno y Alejandro Sierra Urrecho Editorial Anaya Multimedia

PHP5 Luis Miguel Cabezas Granado Anaya Multimedia

# 9. Recursos on-line

| | |
|---|---|
| Apuntes de Sistemes Encastats | Universitat Oberta de Catalunya |
| Documentacion wifly | Ir a web de Microchip |
| Documentación LPCXpresso | Ir a la web de LPC Ware |
| Manual LPC1769 | Descargar manual |
| Manual Driver motores Qik2s9v1 | Descargar Manual |
| Manual MaxSonar EZ0 | Descargar Manual |

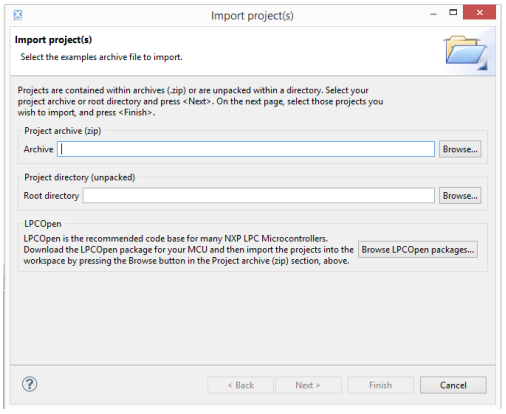
# 10. Anexos

## 10.1. Importación de proyecto en LPCXpresso

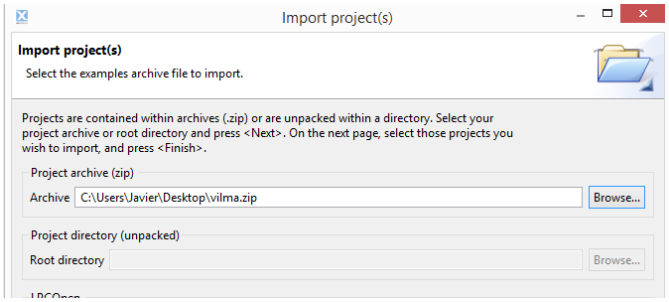
A continuación se describen los pasos para importar el código del proyecto al entorno de desarrollo integrado LPCXPRESSO. Para la descarga e instalación de esta aplicación puede consultarse los manuales que aparecen en las referencias del punto 9 de éste documento

1. Para la compilación del código adjunto al proyecto, será necesario importar éste en el espacio de trabajo del entorno de desarrollo LPCXPRESSO, para ello se

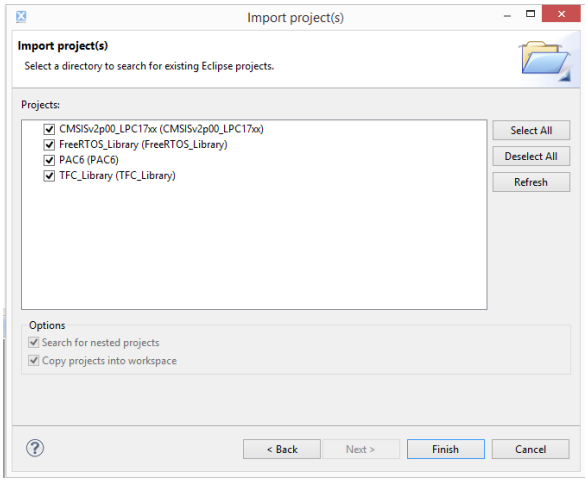

2. Seguidamente aparecerá una ventana en la que se elegirá si el proyecto a importar se encuentra en una estructura de directorios o comprimido



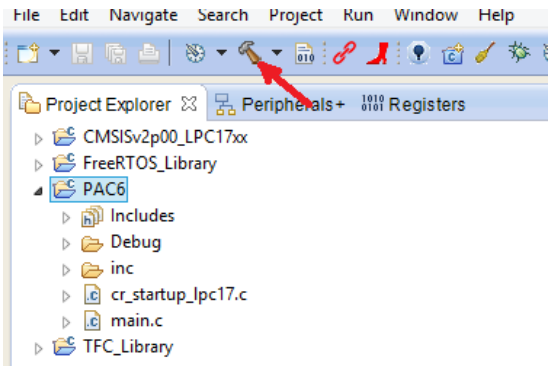

3. En la ventana de importación hay que seleccionar Project Archive (Zip) para este caso y pulsar el botón “browse” para encontrar la ubicación del fichero comprimido.



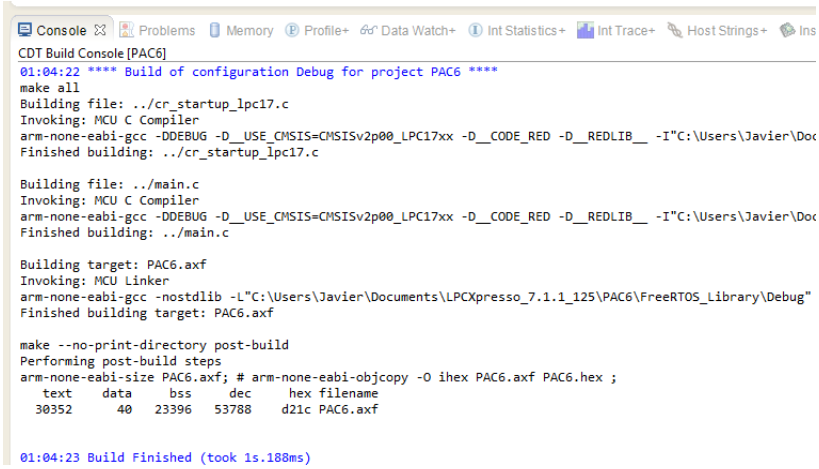

4. A continuación aparecerá una ventana done podremos seleccionar qué librerías se deben importar al espacio de trabajo. Se seleccionan todas y se hace clic en el botón “Finish”



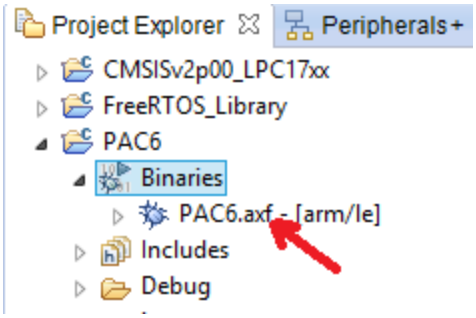

5. Una vez finalizada la importación, el explorador de proyecto mostrará toda la estructura de carpetas del proyecto. El paso siguiente sería compilar todo el código, seleccionando el icono de un martillo, marcado en la siguiente captura



Finalizado el proceso de compilación del código, se podrá comprobar si el proceso de compilación finalizó correctamente o con algún tipo de error.



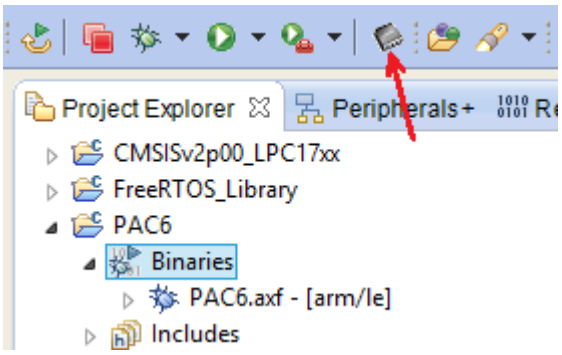

Si el proceso de compilación finaliza correctamente, se generará un fichero ejecutable con extensión axf



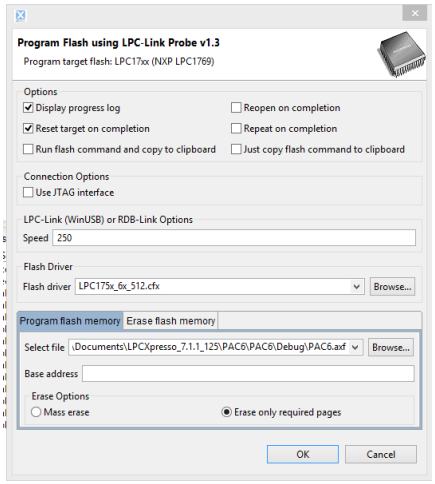

## 10.2. Carga del programa en memoria de la placa

Una vez obtenido el fichero ejecutable, éste debe cargarse en la memoria de la placa LPC1769

1. Se hará clic en el icono del chip del panel superior de la aplicación



2. Aparecerá una ventana con un asistente donde se deberá seleccionar la pestaña “Program flash memory” y a continuación se elegirá la ubicación del fichero ejecutable con extensión axf obtenido tras la compilación. Para ello será necesario pulsar el botón “Browse”. Una vez elegido el fichero pulsar OK para continuar




3. El proceso de programación se iniciará inmediatamente