Arquitectura: Componentes

PID_00247326

José López Vicario
Màrius Montón Macián
Antoni Morell Pérez

Tiempo mínimo de dedicación recomendado: 3 horas

Universitat Oberta de Catalunya

Índice

Introducción

En este contenido se presentan los principales componentes de los sistemas empotrados. Se incluye la descripción de las interfaces con sensores más usadas, de los puertos de comunicación más habituales y de las memorias existentes en este campo; por último, se introduce el concepto de alimentación eléctrica y cómo manejarla en el caso de los sistemas empotrados.

1. Puertos de comunicación

Tal como se ha podido entrever en los diagramas anteriores, cualquier sistema empotrado, sea cual sea su arquitectura, necesita establecer comunicaciones entre las diferentes partes que lo forman.

Denominamos *buses* a estos canales de comunicación, y habitualmente esta información se transporta en forma de señales eléctricas. En su forma más simple, podemos entender el bus como un conjunto de «hilos» por donde viaja la información y normalmente un solo bus interconecta múltiples dispositivos.

Ejemplo

Un valor de +5 V suele representar un «1» lógico, mientras que un valor de 0 V representa el «0» lógico.

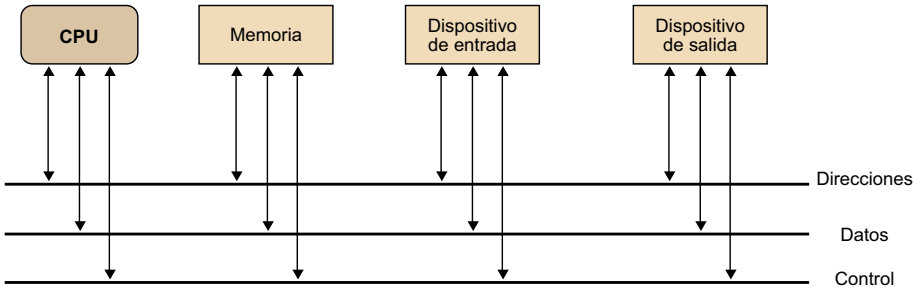
La información que viaja por ellos puede ser de tres tipos: **direcciones**, **datos** y **control**. En primer lugar, la dirección nos indica dónde hay que guardar la información o de dónde se tiene que extraer, ya sea de la memoria del sistema o de los dispositivos entrada-salida. En segundo lugar, los datos contienen la información que estamos procesando. Finalmente, en el bus de control encontramos señales que son necesarias para el funcionamiento correcto de los dispositivos que forman el sistema. Por ejemplo, cuando accedemos a la memoria del sistema para recoger unos datos determinados, aparte de saber la dirección, hay que saber en qué momento están preparados para ser leídos. En el mundo digital, es muy importante remarcar que todo aquello que no tiene dirección no existe, puesto que no hay manera de acceder a ello. Pensemos en cuando borramos un fichero del disco duro de nuestro PC: la información (datos) no se borra físicamente, pero sí que dejamos de recordar su ubicación (dirección) y, por lo tanto, no podemos ver la información aunque permanezca allí.

Ejemplo

En una máquina expendedora de bebidas nos puede interesar saber el número de unidades que quedan de un artículo determinado, que hemos de actualizar en el momento de hacer la reposición y que irá decreciendo cada vez que una bebida es dispensada. Este dato, que estará ubicado en una dirección determinada (posición de memoria), es el que viajará por el bus de datos.

La figura siguiente muestra los diferentes elementos de un sistema empotrado interconectados mediante los buses de direcciones, datos y control. En este caso, el bus de direcciones transporta 12 señales, el bus de datos 16 y el bus de control 4:

Figura 1. Los buses de direcciones, datos y control



1.1. Serie

Junto con el puerto paralelo, el puerto serie era uno de los puertos por excelencia en los primeros años que aparecieron los PC. Concretamente, este tipo de puerto se utiliza para establecer una comunicación punto por punto entre dos dispositivos. En cuanto al nombre de *puerto serie*, está determinado por el hecho de que solo se puede enviar un bit de datos a cada instante de tiempo. A pesar de que hay unos cuantos estándares para llevar a cabo comunicaciones mediante el puerto serie, normalmente, cuando se habla de este tipo de puerto (también conocido como puerto COM), se considera el estándar RS-232.

En la figura lateral se presenta, junto al cableado relacionado, el aspecto del puerto serie basado en RS-232 con el tipo de conector más utilizado, que es de 9 pines y se conoce con el nombre de DB-9 (existe una versión de 25 pines pero está menos extendida y resulta más cara). Este estándar concreto permite comunicaciones a 20 kbps hasta una distancia de 25 metros (de cable). Sin embargo, hay que señalar que la velocidad que se puede establecer depende de la longitud del cable y, en realidad, se pueden conseguir velocidades más altas (en torno a los 115.200 bps).

Puerto serie basado en RS-232 con conector de 9 pines DB-9.

En el caso de los sistemas empotrados, este puerto se utiliza normalmente para llevar a cabo tareas de programación o depuración del sistema empotrado. Es una opción atractiva en cuanto a coste y simplicidad y, por este motivo, se encuentra muy extendido en una amplia gama de dispositivos. Sin embargo, tiene como limitación la baja velocidad y el tamaño del puerto en sí; por este motivo, se usan actualmente también otras alternativas para hacer estas tareas, como es el caso del puerto USB (que se presenta más adelante).

1.2. Paralelo

Como se ha comentado en el caso anterior, el puerto paralelo es otro de los puertos clásicos en la informática convencional. Normalmente, se utilizaba para conectar el PC con impresoras. Sin embargo, este uso se está dejando actualmente al puerto USB.

Este tipo de puerto también establece una conexión punto por punto pero, a diferencia del puerto serie, en este caso se permite la transmisión de hasta 8 bits en paralelo (de ahí su nombre). También hay que comentar que sigue el estándar de comunicación IEEE 1284, que permite velocidades de comunicación que oscilan entre los 4 y los 16 Mbps a unas distancias de entre 6 y 10 metros (según la calidad del cable). En la figura 65 se presenta el puerto paralelo.

Puerto paralelo.

En cuanto a los sistemas empotrados, este tipo de puerto no se suele utilizar mucho. Es decir, raramente este tipo de puerto forma parte del hardware de un sistema empotrado. A veces, se incluye porque el sistema se tiene que conectar con una impresora u otro tipo de periférico, pero cada vez es más extraño encontrar este tipo de aplicación con la gran implantación del puerto USB.

Sin embargo, hay que comentar que una aplicación interesante del puerto paralelo en un sistema empotrado es su utilización para la fase de desarrollo, concretamente para llevar a cabo tareas de depuración del funcionamiento del sistema. Para hacer esto, se deben introducir órdenes de llamada al puerto paralelo en diferentes partes del código y, como el puerto paralelo permite que el último valor del pin se mantenga hasta que este no cambie, uno puede observar el desarrollo del código en el sistema cómodamente. Esta tarea viene facilitada por el hecho de que muchos sistemas operativos (como el caso de Linux) ofrecen controladores (*drivers*) para poder llevar a cabo estas tareas sin dificultad.

Ejemplo

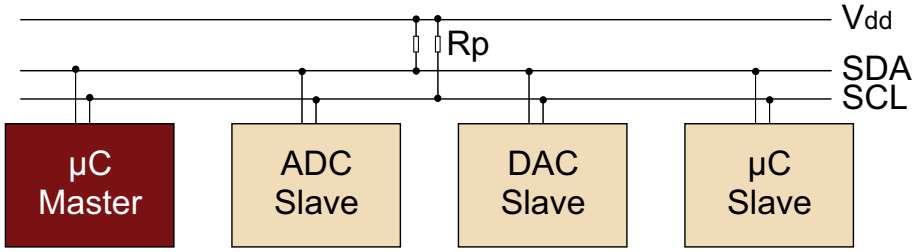
Es muy fácil insertar LED en los pines del puerto paralelo y usarlos para indicar en qué línea del código está.

1.3. I2C

La comunicación I2C es un bus de tipo serie síncrono de solo dos hilos, donde en un hilo se transmiten los datos (hilo SDA) de forma bidireccional y el otro hilo transmite la señal de reloj (hilo SCL). Este bus permite varios másteres (multi-máster) y varios esclavos (multi-slave) y está orientado a la comunicación entre dispositivos de baja velocidad actuando como esclavos y uno o varios microcontroladores actuando como másteres (figura 2). La velocidad habitual de un bus I2C es de 400 kHz y en algunos casos puede subir hasta 1 MHz [Jonathan Valdez (2015)].

Este bus dispone que cada dispositivo conectado tenga una dirección única y solamente los dispositivos tipo máster pueden iniciar una transmisión. Una transmisión típica consiste en enviar la dirección del esclavo al que se quiere acceder, una indicación de si se quiere escribir o leer en dicho esclavo, y o bien una dirección interna y un dato que escribir, o bien una dirección interna que se quiere leer. De esta forma es posible leer el estado o los registros del esclavo por parte del máster.

Figura 2. Ejemplo de un bus I2C típico



Dispositivos habituales que usan una comunicación I2C son sensores de baja velocidad, como acelerómetros o sensores de temperatura, dispositivos de expansión de GPIO o, en general, cualquier dispositivo que no necesite una alta transferencia de datos.

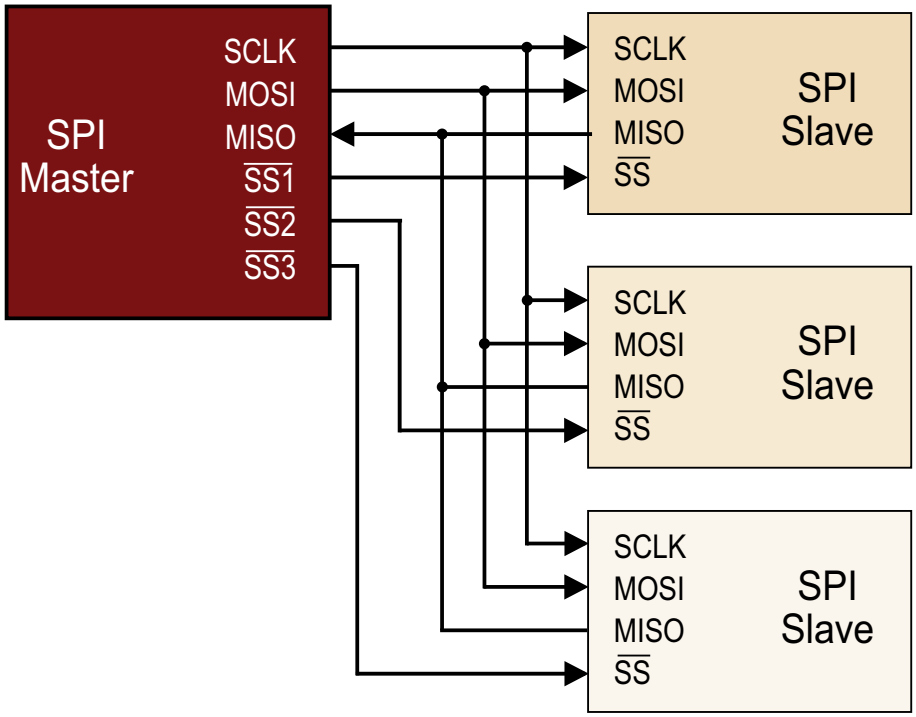
1.4. SPI

SPI* es también un protocolo serie para la interconexión de dispositivos. En este caso no hay ningún hilo que sea bidireccional, sino que hay un hilo de entrada al máster

* **Las siglas de *serial peripheral interface*.**

llamado MISO, otro de salia del máster hacia los dispositivos (MOSI) y un hilo que lleva el *clock* (SCLK). Opcionalmente hay una señal de Select (SS) distinta para cada dispositivo conectado al bus (figura 3).

Figura 3. Ejemplo de un bus SPI básico



Master-Slave

Se denomina máster al dispositivo que es capaz de iniciar una transferencia. Un dispositivo esclavo (*slave*) no puede iniciar una transferencia y solo responde a una transferencia iniciada por un máster.

En este caso, la variedad de protocolos de acceso a los datos o control de cada dispositivo es muy variada y cada fabricante sigue su propio criterio. De este modo, no existe el concepto de dirección, ya que se selecciona a qué dispositivo le llega la transferencia mediante el uso de las señales de SS conectadas a cada dispositivo por separado.

La velocidad de transferencia en este caso es más elevada que la del bus I2C, y son habituales frecuencias de reloj de pocos MHz. Así, este tipo de bus se usa en dispositivos de mayores prestaciones o que necesiten mayor tasa de transferencia de datos.

Los dispositivos más habituales en usar SPI son las memorias tipo flash, ADC (*analog-to-digital converter*) y DAC (*digital-to-analog converter*) de altas prestaciones, controladores de pantallas LCD (*liquid-cristal display*) y, en general, cualquier dispositivo que necesita una alta velocidad en la transferencia de datos.

También se usa este protocolo para la conexión de las tarjetas tipo *SD card* (*secure digital*). En este caso se puede usar un SPI habitual o una versión mejorada con hasta cuatro hilos para el envío de los datos.

1.5. GPIO

GPIO es el acrónimo de *general-purpose input/output*. Los distintos pines de entrada/salida de un microcontrolador pueden, en general, configurarse para que actúen como entrada o salida digital. También es posible configurar un pin concreto para que sea la entrada o salida de un periférico.

En algunos casos, es posible también configurar ciertas características eléctricas del pin, como puede ser el voltaje de trabajo, la cantidad de corriente máxima que puede proporcionar, el *slew rate* o configurar un *pull-up* o *pull-down* para el pin. Esto puede permitir cumplir ciertos estándares para interfaces concretas [Silicon Labs (2015a)].

Figura 4. *Slew rate*

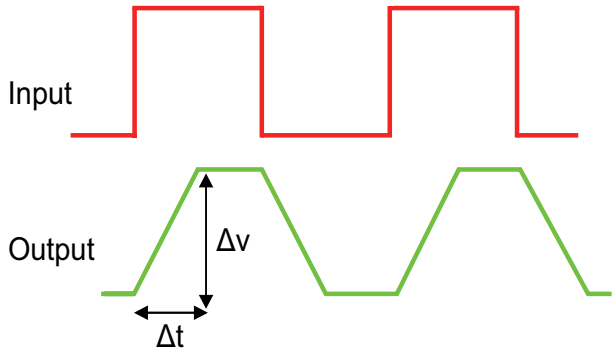
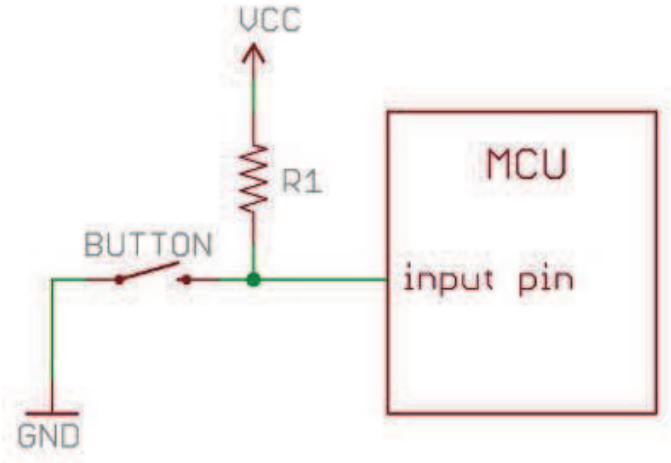


Figura 4

Slew rate es una medida de cómo de rápido puede cambiar el valor de una salida. Normalmente se puede configurar a 2 o 3 velocidades (rápido, normal, lento).

Una resistencia de *pull-up* (o *pull-down*) es una resistencia conectada entre el voltaje de funcionamiento (o a tierra) y una línea de señal. Así, si un dispositivo desea introducir un valor lógico '1' en una línea con un *pull-up*, puede dejar la línea sin ningún valor, ya que la resistencia forzará el valor '1' (figura 5). En cambio, si el mismo dispositivo quiere introducir un '0', conectando la señal a tierra hará que la señal tenga el valor '0', ya que la resistencia es de un valor suficientemente alto para que no «gane» el valor '0' conectado directamente a tierra. El mismo razonamiento puede hacerse para una resistencia de *pull-down*.

Figura 5. Pulsador con *pull-up*



2. Interfaces externas

Una vez introducidos los puertos de comunicación entre dispositivos de corto alcance, se describen en esta sección las principales interfaces de comunicación de mayor alcance para la comunicación con dispositivos externos al sistema en sí.

2.1. USB

Este puerto, desarrollado por el consorcio de compañías USB Implementers Forum (USB-IF), nació con el objetivo de sustituir a los puertos paralelo y serie. El tipo de puerto utilizado es de un tamaño más reducido (tal como se puede observar en la figura lateral) y sus principales características son el bajo coste de la solución, las velocidades de datos más altas que se pueden conseguir y la facilidad de interconexión entre dispositivos, que se hace siguiendo la filosofía de la integración automática, con una configuración simplificada y sin la necesidad de reiniciar el sistema (concepto conocido como ***plug-and-play***).

Puerto USB.

El tipo de conexión USB se basa en el uso de una topología en estrella donde un dispositivo hace de ordenador central (*host*) y el resto se conectan a él. Además, se ofrece la posibilidad de que un ordenador central se conecte a otro ordenador central, lo cual da como resultado la creación de una topología en árbol con diferentes ramas con un máximo de cinco niveles y de 127 dispositivos conectados al ordenador central principal (o *root host*). A pesar de que este número máximo de dispositivos puede parecer limitante en redes de gran escala, esto no suele representar un problema en las típicas configuraciones encontradas en los sistemas empotrados.

En cuanto a la velocidad que se puede conseguir con longitudes de cables no superiores a 5 metros, hay diferentes límites, que dependen de la versión del puerto:

- USB 1.0: 1,5 Mbps.
- USB 1.1: 12 Mbps.
- USB 2.0: 480 Mbps, pero normalmente se trabaja a 125 Mbps (esta es la versión de USB más extendida actualmente).
- USB 3.0: 4,8 Gbps (con una longitud de cable recomendable de 3 metros en este caso).

Otra de las ventajas de este tipo de puerto es que los dispositivos conectados pueden ser alimentados por el mismo puerto, siempre que el nivel de alimentación requerida por el dispositivo no sea demasiado elevado.

Fruto de las ventajas comentadas, el puerto USB es hoy en día una de las soluciones más comunes para poder ofrecer interconexión de numerosos dispositivos y se ha convertido, en muchos casos, en puerto estándar, como sucede en el caso de las impresoras y cámaras digitales.

Ejemplo

Entre los dispositivos que se pueden conectar mediante el USB, podemos mencionar impresoras, cámaras digitales, teléfonos móviles, cámaras de vídeo, dispositivos de almacenamiento, grabadoras, teclados, ratones, etc.

A pesar de que este tipo de puerto está principalmente orientado a la informática de consumo, su utilización en sistemas empotrados ha proliferado en los últimos años. Tal como se ha comentado, el puerto USB se está utilizando como sustitutivo del puerto serie para llevar a cabo tareas de programación y depuración, ya sea porque el conector es más simple, por la posibilidad de detección y configuración automática, por el bajo coste, por la posibilidad de utilizar el mismo puerto para alimentar el sistema o porque tiene más velocidad.

Otra utilidad del puerto USB, dada la mayor velocidad que puede ofrecer al sistema, es la conexión del sistema empotrado con otro dispositivo para transferir datos.

Por otro lado, dadas las posibilidades de interconexión de este sistema, se puede establecer una red de recogida de datos de los diferentes dispositivos empotrados utilizando USB. Siguiendo con el ejemplo anterior, se podría formar una red tipo estrella en la que los diferentes sensores se conectarían con el dispositivo recolector de datos.

Finalmente, dada la gran popularidad del puerto USB en los dispositivos de consumo, este puerto se puede utilizar para conectar dispositivos adicionales al sistema empotrado, como pueden ser cámaras digitales, de vídeo u otros tipos de periféricos según la aplicación del dispositivo en sí.

Ejemplo

Como muestra de esto, podríamos señalar las tareas de monitorización o vigilancia.

2.2. Ethernet

La creación de Ethernet data de la década de 1970 y su estandarización viene determinada por la norma IEEE 802.3. Ethernet ofrece diferentes variantes que trabajan a distintas velocidades; las versiones más populares son las de 10 Mbps, 100 Mbps y 1 Gbps. Estas variantes utilizan también diferentes versiones de cableado físico y ofrecen, por lo tanto, distintos comportamientos que se ven reflejados en la longitud máxima que pueden tener los cables.

Ejemplo

En el caso de 10 Mbps, algunas variantes son las referidas como 10Base2 (cable coaxial con longitud máxima de 185 m) y 10BaseT (par trenzado con 100 m); esta última es la variante más utilizada.

El acceso al medio es compartido y se emplea el protocolo de acceso múltiple con escucha de portadora y detección de colisión (CSMA-CD, *carrier sense multiple access with collision detection*). Los dispositivos que utilizan este protocolo escuchan el medio antes de transmitir sus paquetes para ver si hay alguna otra transmisión activa y, por lo tanto, para evitar colisiones. A la vez que va transmitiendo los datos, el dispositivo escucha también el medio para ver si hay colisión con otro dispositivo, es decir,

ver si otro dispositivo también ha empezado a transmitir. Si se detecta una colisión, se para la transmisión y no se vuelve a intentar transmitir hasta que pasa un tiempo aleatorio.

Inicialmente, Ethernet se basaba en *colgar* todos los dispositivos del mismo cable (cable coaxial) y, por este motivo, se usaba este tipo de protocolo de acceso compartido. Este sistema ofrecía muy buenos resultados pero era bastante ineficiente cuando se aplicaba en redes relativamente grandes. Para reducir el número de colisiones y hacer la red más robusta, se cambió a una topología en estrella, en la que todos los dispositivos se conectaban a un nodo central. En cuanto a este nodo central, hay dispositivos de diferente naturaleza para llevar a cabo esta tarea y ofrecen la posibilidad de dividir la red en diferentes segmentos (o subredes), y por lo tanto se dividen el tráfico global y facilitan la escalabilidad. Según las funcionalidades que incorporan, los dispositivos más comunes son:

- **Concentrador (*hub*)**: actúa como unión física de los diferentes dispositivos. Es decir, es un simple repetidor; si recibe un paquete de un dispositivo, lo retransmite a todos los dispositivos que tiene conectados.

- **Conmutador (*switch*)**: es un dispositivo más sofisticado que el concentrador, puesto que analiza el paquete recibido para ver la dirección de destino. De este modo, reenvía solo el paquete al dispositivo de destino o al segmento donde se encuentra el dispositivo de destino o el segmento donde se encuentra.

- **Encaminador (*router*)**: es un tipo de dispositivo que se suele utilizar en redes grandes donde diferentes subredes (o segmentos) están interconectadas. Básicamente, tiene la misma funcionalidad que el conmutador, pero también tiene la habilidad de encaminar el paquete por la mejor salida en términos de camino más corto y menos congestionado.

Ethernet es una de las opciones más sencillas y con un coste más bajo de las existentes para montar una red de dispositivos empotrados con una velocidad relativamente alta. Sin duda, el puerto Ethernet es el más utilizado para conectar dispositivos en redes de computadores y por esta razón este tipo de puerto también se encuentra implementado en muchas soluciones empotradas. Básicamente, ofrece la posibilidad de conectar el dispositivo de manera sencilla a otros computadores, impresoras, bases de datos e internet.

UAV

Una opción interesante es utilizar Ethernet para conectar el sistema empotrado a internet para ofrecer una solución remota de monitorización y control.

Cabe comentar que, hace tiempo, incluir Ethernet en un sistema empotrado era bastante complicado porque un dispositivo basado en este estándar tenía una alta complejidad, tanto en cuanto al circuito como a los componentes requeridos. Sin embargo, esta tarea se ha facilitado bastante, dado el alto nivel de integración logrado últimamente. En concreto, hay implementaciones de chips de controladores Ethernet específicamente diseñadas para su implementación en sistemas empotrados y tienen como característica principal ofrecer interfaces Ethernet de bajo coste y simples. Las dos más populares son las dadas por los chips Realtek 8019 y Cirrus CS8900A.

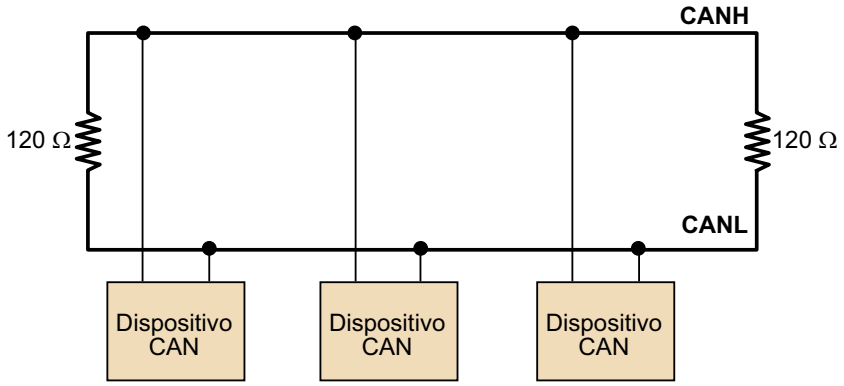
2.3. CAN

Esta sección la concluimos presentando una tecnología utilizada en entornos industriales. Estos tipos de entornos están caracterizados por su hostilidad en términos de ruido electromagnético severo. Por ello, son diseñados teniendo en cuenta otros criterios. En esta sección concreta, nos centramos en la tecnología *controller area network* (CAN), que se suele utilizar en entornos severamente afectados por interferencia electromagnética.

En las décadas de 1970 y 1980, la automoción sufrió un cambio importante en el sentido de que los automóviles disponían cada vez de más dispositivos electrónicos, como es el caso del ABS, la transmisión electrónica, el sistema de climatización, el cierre centralizado, etc. Naturalmente, todas estas partes de los vehículos se debían interconectar para poder llevar a cabo una interacción entre ellos, ya sea para llevar a cabo órdenes de control o para intercambiar datos. Originalmente, todos estos dispositivos se conectaban usando cableado punto por punto, pero esto limitaba tanto las prestaciones del sistema como su escalabilidad y, además, aumentaban el coste del vehículo y su peso. Para solucionar esto, la empresa Bosch desarrolló CAN a finales de la década de 1980. Aunque estaba orientado principalmente para la industria de la automoción, fue adoptado después por otras aplicaciones industriales. En particular, CAN está confirmada por la International Standards Organization (ISO) en forma de la estándar ISO 11898 e ISO 11518-2.

La solución CAN se basa en reemplazar la complejidad del múltiple cableado por el uso de un único bus en el que todos los dispositivos se conectan y en el que se pueden conseguir velocidades de hasta 1 Mbps con cables de longitud inferiores a 40 m (el mínimo de velocidad es de 40 kbps, que se puede lograr con cables de 1 km). Concretamente, este bus consiste simplemente en una línea balanceada formada por un par de cables (podéis ver la figura 6). En la figura, se puede observar también que las líneas están terminadas con resistencias de 120 Ω y el motivo de esto es evitar reflexiones de las transmisiones efectuadas sobre la línea.

Figura 6. Red CAN



En una red CAN, los dispositivos se pueden añadir o desconectar en cualquier momento y se permite que diferentes dispositivos actúen como máster, donde cada uno

de estos másteres se puede encargar del control local de la red. A pesar de ello, el tipo de acceso al bus se lleva a cabo usando un mecanismo de contención basado en CSMA/CD + AMP (*carrier sense multiple access with collision detection and arbitration message priority*). Básicamente, este esquema es similar al caso de CSMA/CD utilizado en Ethernet en el sentido de que un nodo mira la línea antes de transmitir para ver si está libre. Si es el caso, va transmitiendo los diferentes bits de información y, a la vez, va comparando si los bits existentes en la línea son los mismos que los que va transmitiendo. En caso de que no sea así, es porque hay colisión. Entonces se para la transmisión y se inicia un periodo de arbitraje que no se da en el protocolo CSMA/CD, y por ello se denomina CSMA/CD+AMP. Este arbitraje se basa en dar paso al dispositivo con prioridad más alta, en el que la prioridad de los diferentes dispositivos ha sido asignada durante el diseño del sistema. En cuanto al número de dispositivos que se pueden conectar a una línea, el máximo es 30.

Teniendo en cuenta que en un automóvil se encuentra, desde el punto de vista electromagnético, en un entorno muy ruidoso dada la presencia de motores eléctricos, sistemas de ignición, emisiones de radio, etc., la comunicación establecida tiene que ser altamente robusta. Para poder ofrecer esto, se emplea un sistema de comunicación balanceada. Cuando se usa este tipo de comunicación, la información se envía variando los niveles de voltaje de dos líneas.

En este caso, las líneas utilizadas son las líneas CANH y CANL (podéis verlas en la figura 6). En el receptor se mide la tensión diferencial de estas dos líneas, es decir, se mide la tensión CANH-CANL para determinar el bit enviado. Esto difiere de esquemas convencionales en los que el receptor extrae la información mirando el voltaje de una sola línea. En el caso de CAN concreto, un bit 0 se envía induciendo las tensiones 3,5 V a la línea CANH y 1,5 V a la línea CANL. Por lo tanto, en el receptor se tendrá una tensión diferencial CANH-CANL igual a 2 V. En el caso de querer enviarse un bit 1, las dos líneas se ponen a 2,5 V teniendo como resultado una tensión diferencial de 0 V. El objetivo para utilizar este mecanismo diferencial es reducir el ruido inducido por otras fuentes electromagnéticas. Es decir, si hay un ruido de este tipo, este será inducido del mismo modo a las dos líneas CANH y CANL. Por lo tanto, al obtener la tensión diferencial CANH-CANL, este ruido será compensado.

Ejemplo

Si se quiere enviar un 0 lógico, CANH será igual a 3,5 V y CANL a 1,5 V. Si tenemos un ruido inducido a un instante de tiempo igual a +1 V, lo que se recibe de las líneas CANH y CANL será 4,5 V y 2,5 V, respectivamente. Al obtener la tensión diferencial, sin embargo, se obtendrán los 2 V nominales correspondientes a un bit 0.

Hay que comentar que este tipo de puerto se encuentra muy a menudo en dispositivos utilizados en entornos industriales. Por lo tanto, es una opción muy adecuada si la aplicación del sistema empotrado es de este tipo. Sobre todo en caso de que trabaje en un entorno muy hostil en términos de ruido e interferencia, y la limitación de velocidad no sea un impedimento para la aplicación concreta.

2.4. Modbus

Modbus es un protocolo de comunicación serie usado sobre todo en PLC y que está pensado para la comunicación entre todo tipo de máquinas industriales; normalmente

se usa para conectar las distintas máquinas de una planta industrial hacia un supervisor o a un sistema SCADA [MODICON Inc. (1996)].

Los datos se envían en serie según distintos estándares, siendo los más habituales los protocolos serie RS-232 y RS-485. Sobre esta capa física de transmisión de datos, se añade una capa de comunicación común para todos los componentes. Hay también una variante que usa comunicación vía cable de red Ethernet estándar.

Esta capa permite el acceso al estado de cada uno de los dispositivos conectados y acceder a nivel de bit si es necesario. En esta capa hay distintos estándares, los más habituales de los cuales son:

RS-232 y RS-485

RS-232 es un estándar para una comunicación serie del estilo de la descripción hecha en el apartado 1.1 de este material. RS-485 es otro estándar de comunicación serie, en este caso con una comunicación diferencial de los datos.

- Modbus RTU: comunicación binaria y compacta de los datos con corrección de errores.
- Modbus ASCII: Variante de la anterior donde los datos se envían en texto plano.
- Modbus TCP/IP: Usado con Ethernet donde se encapsulan los datos dentro de una trama TCP/IP estándar.

Aunque el propósito original fue el de la comunicación entre maquinaria industrial, también se usa para la comunicación entre sensores y dispositivo, normalmente usando RS-485 y Modbus RTU debido a su sencillez, su robustez a interferencias y la capacidad de usar cables de gran longitud.

2.5. 4-20 mA

Esta interfaz se basa en un lazo de corriente entre el dispositivo transmisor y el dispositivo receptor, de manera que el dato que se va a transmitir está codificado en la cantidad de corriente que fluye entre los dos dispositivos. Así, tal como su nombre indica, los valores mínimos y máximos de esta interfaz son 4 mA (miliamperios) y 20 mA de corriente [DataQ Instruments (2017)].

Al tener un mínimo distinto de 0 permite detectar cuándo hay un fallo o corte en la comunicación, ya que entonces la corriente que se detecte en el receptor será 0 mA y eso será un indicador de un fallo en el mecanismo de comunicación.

Este tipo de transmisión por lazo de corriente hace esta transmisión bastante inmune al ruido electromagnético, a la vez que permite grandes distancias entre transmisor y receptor (llegando en algunas instalaciones a centenares de metros o incluso kilómetros).

Este tipo de interfaz se usa habitualmente para la comunicación de sensores hacia los dispositivos de control y es propio de cada fabricante o sistema definir la correspon-

dencia exacta entre el valor sensado y la cantidad de corriente enviada hacia el receptor (siempre cumpliendo con el mínimo y el máximo).

2.6. 0-10 V

Este tipo de interfaz es de naturaleza muy simple, ya que solo consta de una señal analógica de voltaje variable entre los 0 y los 10 Voltios. Esta señal se genera en la parte sensora según la magnitud del valor físico.

La gran ventaja de este tipo de interfaz es su simplicidad en ambos lados de la transmisión y su sencilla adaptación al campo digital usando un ADC.

Por otra parte, sus principales desventajas son su limitado rango de funcionamiento debido a las interferencias y a las limitaciones físicas de la transmisión de un voltaje.

2.7. Dry contact

Esta interfaz es quizá la más sencilla de todas, ya que se basa en un circuito que se abre o cierra según el estado del sensor. Consta de dos hilos que tienen conexión eléctrica o no según el sensor, pudiéndose ver como un interruptor que se conecta o desconecta.

Por ello, esta interfaz puede generar directamente una interrupción cada vez que cambia su estado (véase el apartado «Interrupciones» dentro del material titulado «Microcontroladores I: CPU» para más detalles).

Sus características y ventajas e inconvenientes son muy similares a un sensor tipo 0-10 (véase el apartado 2.6 de este material).

3. Memorias

El subsistema de memoria es una parte esencial de cualquier sistema empotrado, puesto que es donde guardamos tanto los datos como las instrucciones que forman parte de nuestras aplicaciones. El diseño correcto de este subsistema en cuanto a dimensionado, gestión de los propios recursos y adecuación de los tiempos de ejecución afecta directamente al rendimiento global del sistema. Es más, asignaciones de memoria inadecuadas pueden incluso dejar nuestra aplicación en un punto muerto, «colgada». En general, nos podemos encontrar en dos situaciones claramente diferenciadas: si nuestra aplicación es pequeña, la memoria interna del procesador puede resultar suficiente y, por lo tanto, no tendremos problemas de compatibilidad. En cambio, en aplicaciones más grandes, tendremos que dotar al sistema de un módulo externo de memoria. En este caso, la compatibilidad, sobre todo en cuanto a tiempo de ejecución, es fundamental, tal como veremos más adelante.

En cuanto al tipo de memoria, de entrada podemos hacer dos grandes grupos: memoria de acceso aleatorio o RAM y memoria solo de lectura o ROM. A continuación, se presenta una clasificación más detallada de los diferentes tipos de memoria, si bien no es exhaustiva. Tenemos:

- **RAM**. Su nombre indica que podemos acceder a cualquier posición de memoria en cualquier momento, a diferencia de lo que sucede en otros tipos de memoria (normalmente antiguos), como pueden ser las cintas magnéticas, donde el acceso a una posición determinada pasa forzosamente por el acceso a posiciones anteriores. Además, los tiempos de lectura y escritura son del mismo orden de magnitud.

- **ROM**. La memoria ROM es una memoria de acceso aleatorio como la RAM pero está pensada, como su nombre indica, solo para lectura. Se puede escribir en ella, obviamente, pero el proceso de escritura es siempre mucho más lento que el de lectura y normalmente nos referimos a él como programación de la ROM.

En cuanto a la interacción entre memoria y sistema, cabe señalar que la interfaz de cualquier memoria está formada por tres tipos de señales, que, coincidiendo con los tipos de buses, son: direcciones, datos y señales de control. Independientemente del tipo de memoria y del mecanismo de almacenamiento que emplee, la interfaz más básica con la memoria la podemos ver en la figura 7, en la que de momento se han obviado las señales de control. Fijémonos en que un conjunto de bits del bus de dirección identifica una palabra de la memoria, que se lee o se escribe mediante los *buffers* (seguidores de tensión) de entrada/salida. Sin embargo, y por cuestiones de fabricación, es posible que las memorias no se adecuen siempre a las necesidades del sistema en términos de tamaño, es decir, que no haya suficientes posiciones o direcciones o que

la longitud de la palabra en memoria no sea suficiente. Más adelante veremos cómo se pueden solucionar las dos situaciones, entre otras.

3.1. RAM

Las memorias RAM son dispositivos en los que se puede almacenar datos y leerlos posteriormente, así como modificar su contenido.

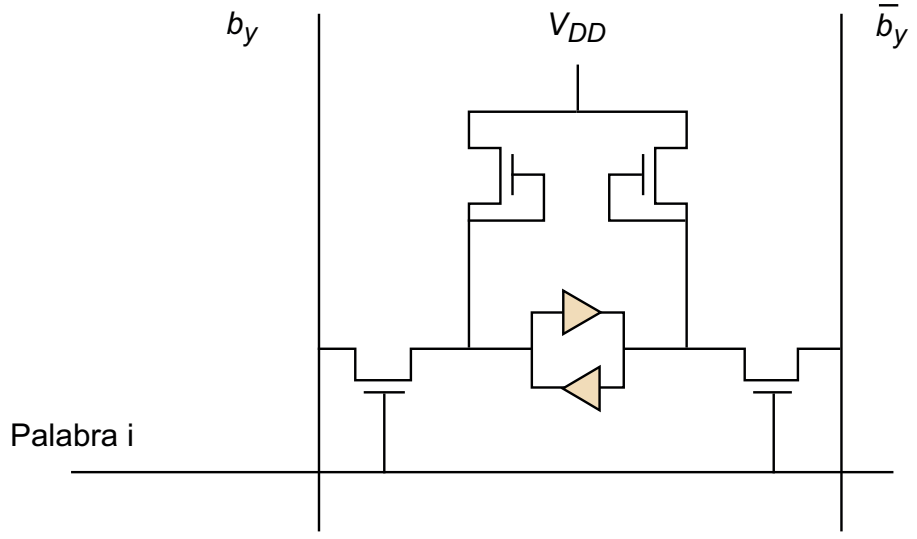
En este punto nos centraremos en los dos tipos básicos de memoria RAM, es decir, memoria RAM estática y memoria RAM dinámica. En primer lugar, describiremos la celda básica de memoria en cada uno de los casos y, a continuación, trataremos diferentes aspectos relacionados con la integración de la memoria en un sistema empotrado.

3.1.1. Memoria RAM estática (SRAM)

La celda básica de memoria SRAM se puede ver en la figura siguiente y está formada por seis transistores. Recordemos que cada inversor está formado por dos transistores y, además, son necesarios los dos transistores de carga conectados a la tensión de referencia. Además, se requieren dos transistores más de acceso para poder ejecutar las operaciones de lectura y escritura. La parte esencial de la memoria y donde se almacena el bit de información es el *latch** que forman los dos inversores.

*** *Latch* es sinónimo de biestable o *flip flop*.**

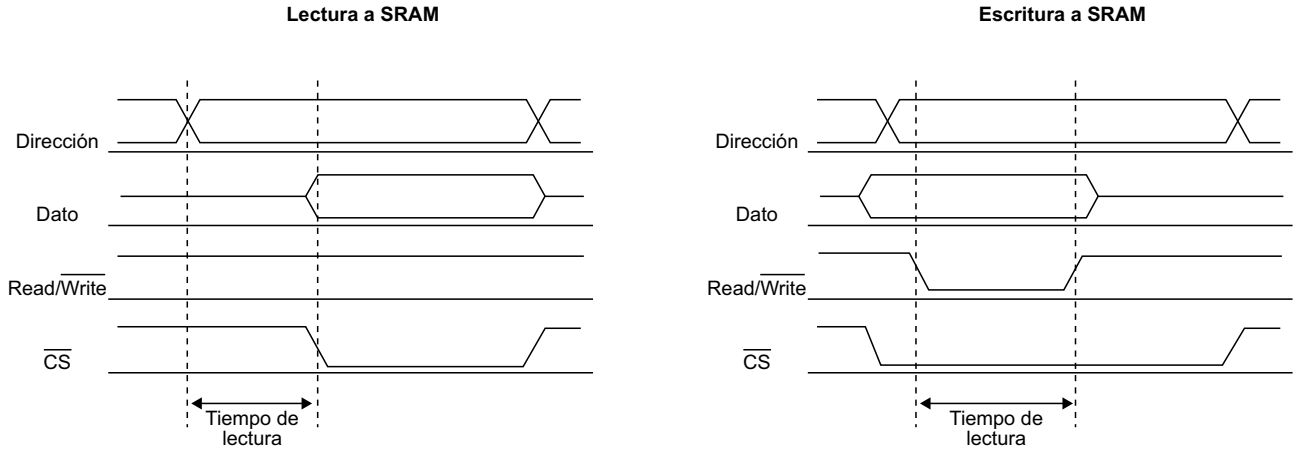
Figura 7. Celda básica de una memoria SRAM



Para leer de la memoria, hay que tener precargadas las líneas b_i y $\bar{b}_i$ a una tensión intermedia entre masa y la de referencia. Entonces, cuando se activan los transistores de acceso seleccionando la dirección de memoria correspondiente, la línea b_i y la $\bar{b}_i$ toman los valores almacenados. Finalmente, el valor del bit tiene que ser sensato, amplificado y transmitido al exterior. A la hora de escribir, simplemente cargamos las

líneas b_i y $\bar{b}_i$ mediante los amplificadores de escritura. Una vez activados los transistores de acceso, el bit de información quedará almacenado en el *latch* de la figura. En la figura siguiente vemos la temporización típica en los procesos de lectura y escritura en una memoria SRAM. Fijémonos en que se distinguen los tiempos de acceso a memoria para cada una de las operaciones, puesto que no han de ser iguales.

Figura 8. Temporización de las señales en una memoria SRAM



En el momento de integrar una memoria determinada en un sistema empotrado, es posible que el procesador tenga un tamaño del bus de datos o de direcciones que coincida con el número de líneas disponibles en la memoria. En este caso, la integración es inmediata. Pero esto no es siempre así. Nos podemos encontrar con que esta condición no se cumpla por alguna de las dos partes, es decir, puede que el número de líneas del procesador no sea suficiente o que la memoria se quede corta en cuanto al tamaño de palabra o en cuanto al número de direcciones posibles. A continuación, se plantea un ejemplo de diseño lo más general posible y se propone una solución de integración.

Supongamos que las especificaciones del sistema en cuestión requieren una memoria SRAM capaz de almacenar 4 K palabras de 16 bits. Sin embargo, los módulos que tenemos disponibles son de 1 K palabras y solo 8 bits. Así pues, serán necesarios 8 módulos para poder cumplir las especificaciones. Además, el procesador tiene un bus de datos de 8 bits de ancho y un bus de direcciones también de 8 bits. Sabiendo que para poder distinguir 4 K direcciones diferentes necesitamos 12 bits, necesitaremos dos transferencias tanto para las direcciones como para los datos.

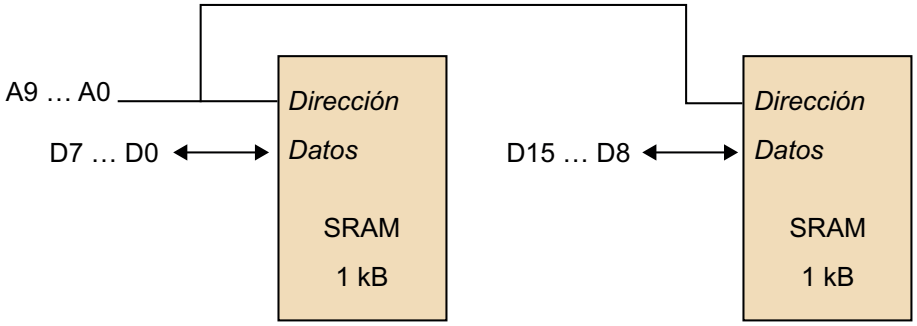
Ejemplo

La dirección 001011001111 corresponde al primer bloque de memoria (00) y, dentro de este, tenemos que mirar la dirección 1011001111.

En primer lugar, utilizaremos bloques de dos módulos de memoria para obtener 1 K palabras de 16 bits de ancho, tal como se muestra en la figura siguiente. Como se puede apreciar, las líneas de dirección son comunes a los dos módulos, mientras que los bits de datos se dividen entre ellos, es decir, un módulo se encarga de los bits D7...D0 y el otro de los bits D15...D8. Una vez obtenidos los bloques de 1 K × 16 bits, resulta más o menos sencillo ampliar el diseño, en este caso cuadruplicar, a un sistema SRAM de 4 K × 16 bits. Si disponemos de 4 bloques de memoria de 1 K × 16 bits, solo hay que seleccionar en cuál de los 4 bloques tenemos que escribir o leer los datos en cada acceso. Fijémonos en que 4 K palabras se representan con 12 bits.

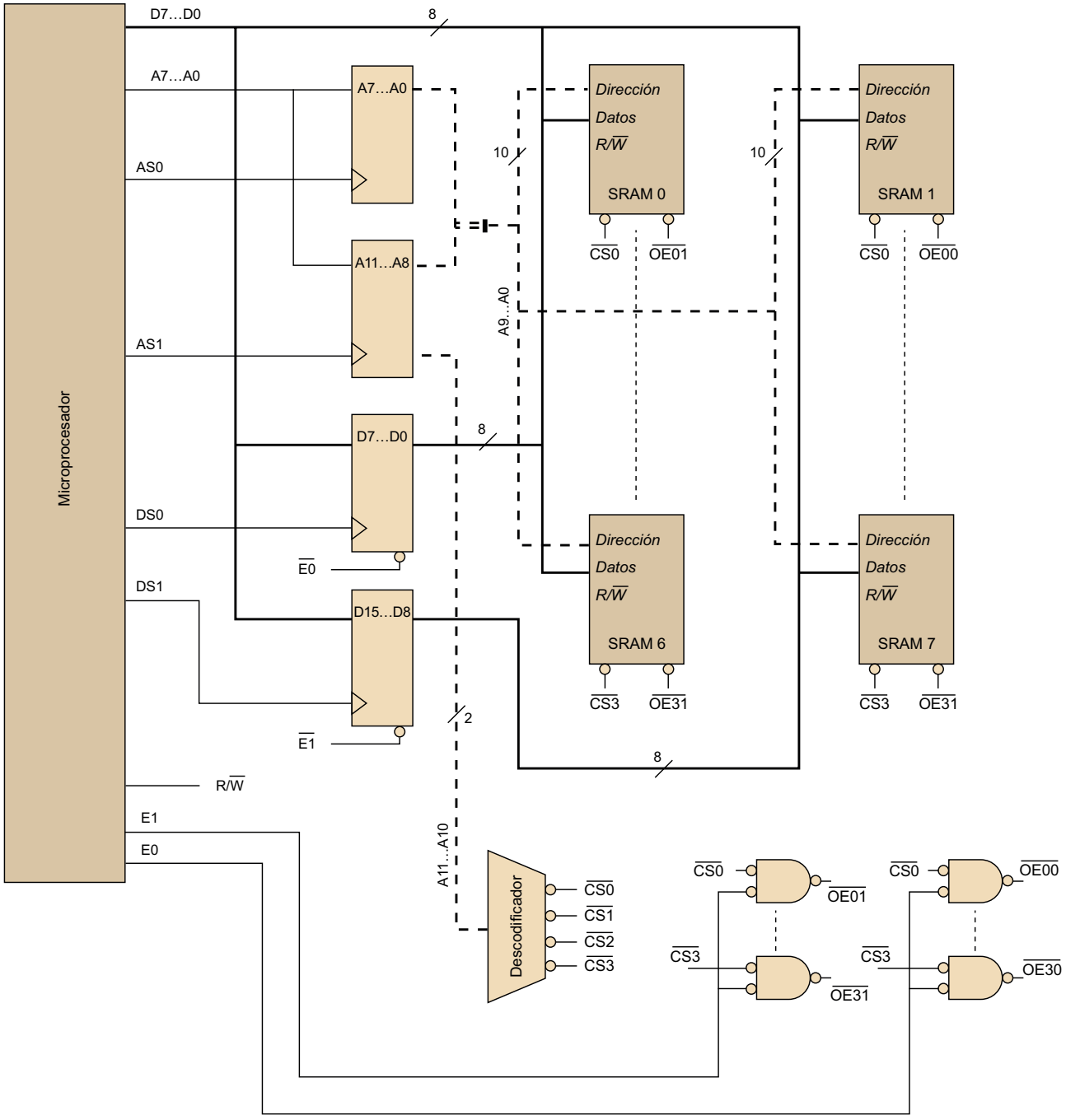
De estos, 10 bits (A9...A0) serán necesarios para especificar la palabra deseada dentro del bloque de memoria. Los otros dos bits (A11, A10) servirán para determinar de manera unívoca qué bloque de memoria está activo en cada momento. Fijémonos en que los bits de dirección A9...A0 deben ser comunes a los cuatro bloques de memoria y, por lo tanto, es imprescindible desactivar los tres últimos bloques para no acceder a una posición indeseada. Un sencillo bloque de lógica combinacional nos servirá para llevar a cabo esta misión.

Figura 9. Memoria SRAM de 16 bits a partir de dos bloques de 8 bits



Finalmente, hemos de solucionar el hecho de que el procesador disponga de solo ocho líneas tanto para los datos como para las direcciones. Recordemos que las palabras (16 bits) y las direcciones (12 bits) del sistema en cuestión requieren dos transferencias. Sin embargo, necesitamos los 16 bits de datos y los 12 bits de dirección disponibles a la hora de acceder a la memoria, al menos para escribir. Con esta finalidad utilizaremos cuatro registros (dos para los datos y dos para las direcciones). A pesar de que hablaremos de los registros más adelante, de momento quedémonos con la idea de que son elementos de memoria de tamaño pequeño (unos cuantos bits). Cuando hay una señal de activación, almacenan los bits que se encuentran a su entrada en aquel momento y, a continuación, se encuentran en disposición de ser leídos a la salida hasta que se aparezca la señal siguiente, momento en el que los bits guardados serán reemplazados por los nuevos bits de entrada. El diseño completo del sistema de memoria y su interacción con el microprocesador se puede ver en la figura siguiente. A continuación detallamos el funcionamiento de las operaciones de escritura y lectura.

Figura 10. Sistema de memoria SRAM 4 K × 16 bits

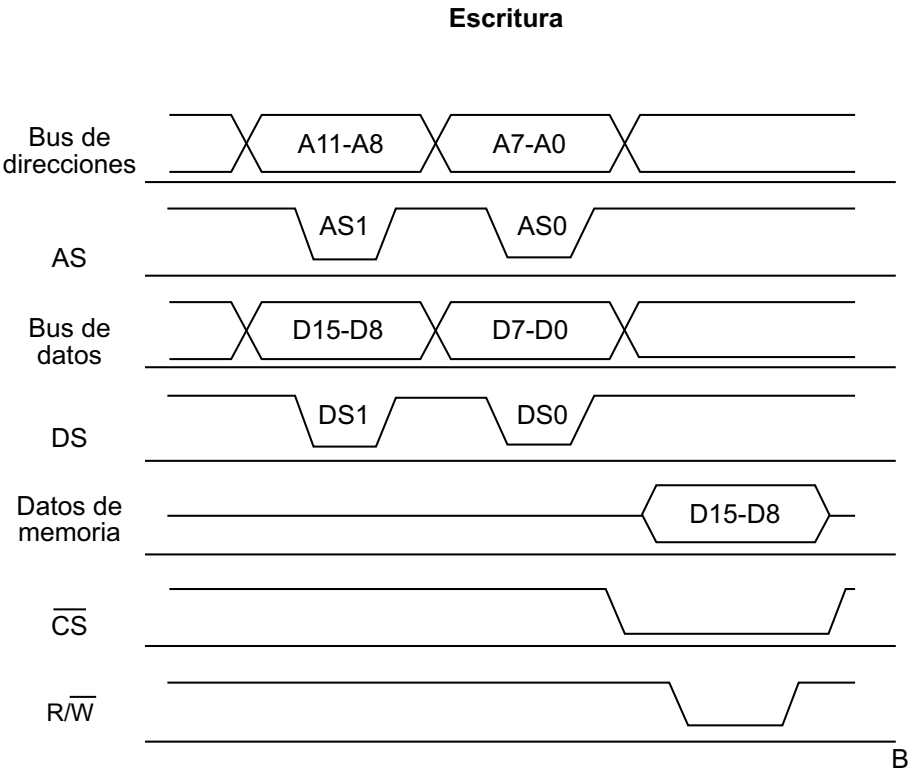


Escritura en memoria

El proceso de escritura en memoria se puede resumir en tres pasos que siguen el diagrama temporal de la figura siguiente. En primer lugar, se cargan los registros tanto de direcciones como de datos. Para hacerlo, utilizamos las señales *address strobe* (AS) y *data strobe* (DS) para indicar cuándo las señales están disponibles en las líneas de salida del procesador. Gracias a los bits A11 y A10, leídos en la primera transferencia, se calcula el bloque de memoria que hay que activar por medio de la correspondiente señal de *chip select* (CS). Finalmente, solo hay que dar la orden de escritura para ejecutar definitivamente la operación. Aparte, hay que mantener el *buffer* de salida de

datos de las memorias en alta impedancia durante todo el proceso de escritura para evitar un conflicto en el bus de datos. El lector puede comprobar en el diagrama de bloques de la figura siguiente cómo se ha tenido en cuenta esta cuestión mediante las señales E1 y E0 del microprocesador, que, tal como veremos a continuación, se utilizan para la lectura de datos en memoria.

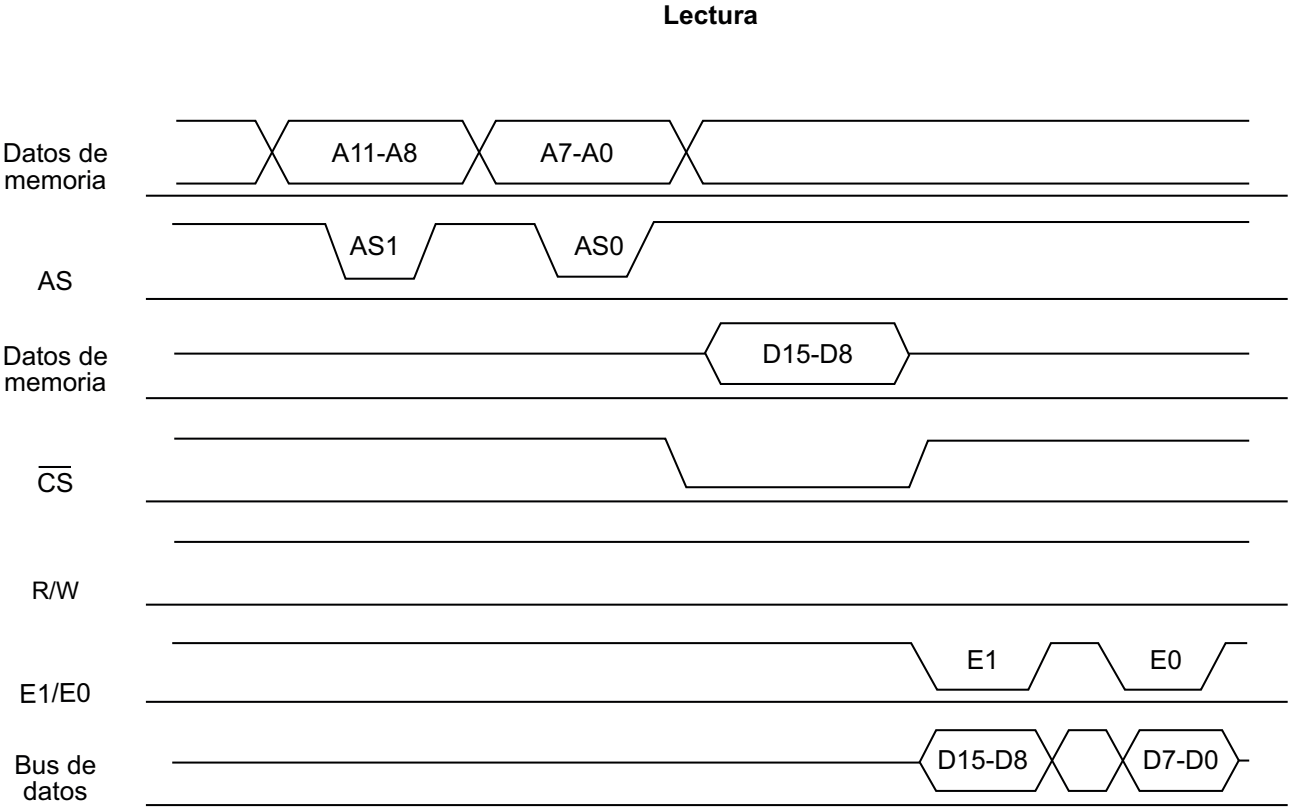
Figura 11. Proceso de escritura del sistema basado en SRAM



Lectura de memoria

El proceso de lectura se encuentra descrito en el diagrama de la figura siguiente. En este caso, solo utilizamos los registros de direcciones como memoria temporal, puesto que los bits de datos viajarán directamente de la memoria al procesador en dos transferencias de 8 bits cada una. Igual que en el caso de escritura, hay que evitar un conflicto en el bus de datos y, por ello, en esta ocasión nos debemos preocupar por dejar las salidas de los registros de datos en alta impedancia en el momento de la lectura de los bits. Podemos dividir el proceso de lectura en tres etapas. En primer lugar, se carga la dirección deseada a los registros correspondientes del mismo modo que en el caso anterior. Esto provoca, en segundo lugar, la habilitación del bloque de memoria donde se ubica el dato, si bien no está todavía disponible, puesto que los *buffers* de salida permanecen desactivados. Finalmente, las señales de habilitación que emite el procesador (E1 y E0) activan sucesivamente los *buffers* de salida de los dos módulos de memoria donde se guarda el dato para acabarlo transfiriendo al microprocesador.

Figura 12. Proceso de escritura del sistema basado en SRAM



Ya para acabar, hay que remarcar que este es solo un ejemplo de diseño de un sistema de memoria SRAM. Se ha procurado hacerlo lo más completo posible, pero, obviamente, habrá que estudiar cada caso por separado y tener en cuenta las características tanto del procesador como de la memoria con la que trabajamos.

Ejemplo

Nos podemos encontrar con que el procesador tenga suficientes líneas tanto para direcciones como para datos, si bien esta situación no es lo habitual en sistemas empotrados. En el otro extremo, también nos podemos encontrar con un bus para datos y direcciones compartido, que requerirá un diseño parecido al expuesto, con un registro para direcciones y otro para datos. En resumen, pues, hay que examinar cada caso con detenimiento y adaptarse a las características del hardware.

3.1.2. Memoria RAM dinámica (DRAM)

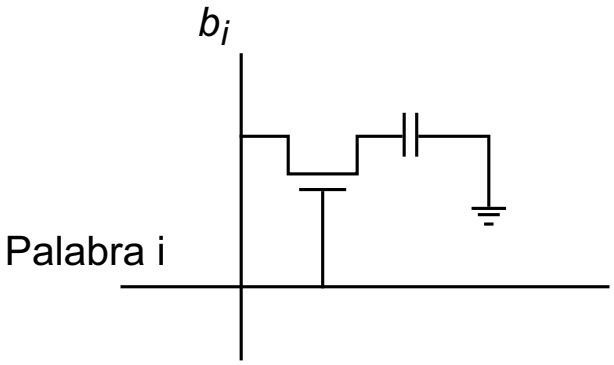
La celda básica de memoria DRAM es mucho más simple que la de SRAM y se puede ver en la figura siguiente. Como podemos comprobar, se trata de un solo transistor MOS con los *buffers* de entrada/salida correspondientes, y el mecanismo de almacenamiento de información se deriva de la capacidad parásita del propio transistor. De este hecho se pueden extraer dos características importantes de las memorias DRAM:

- Permiten integrar mayores cantidades de memoria que las SRAM porque son menos complejas.

- Como el mecanismo de almacenamiento es volátil, puesto que la carga acumulada drena por medio de la resistencia del transistor, hay que renovar continuamente la información guardada. Para hacerlo, se efectúa una lectura de los bits para reescribirlos inmediatamente después. Esta renovación de la información se conoce con el nombre de refresco o ciclo de refresco, y es el fabricante de la memoria

quien debe determinar el tiempo máximo permitido entre refrescos o intervalos de refresco.

Figura 13. Celda básica de una memoria DRAM



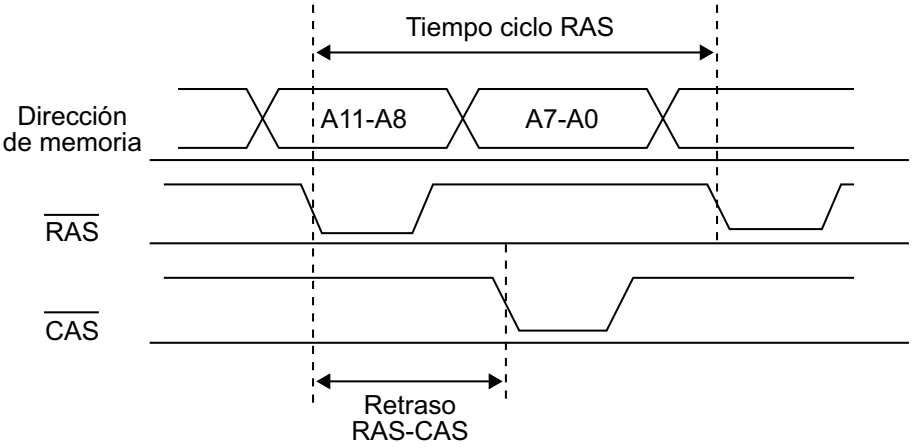
La lectura del bit guardado se hace precargando b_i a una tensión intermedia entre masa y la tensión de referencia. Después se activa la puerta del transistor y se comparte así la carga almacenada. Por lo tanto, si el valor guardado era un 0, la tensión b_i baja y si era un 1, la tensión sube. Finalmente, hay que detectar y amplificar esta variación para traerla al exterior de la memoria. Sin embargo, es obvio que el proceso de lectura deteriora la información almacenada y es la circuitería de la propia memoria la que se encarga de restaurarla o reescribirla. La operación de escritura es todavía más simple: solo se carga la línea b_i al valor deseado y se activa la puerta del transistor, con lo cual se carga o descarga la capacidad parásita para guardar el valor lógico que queremos.

En cuanto al diseño del sistema de memoria, todo lo que se ha visto para la memoria SRAM se puede trasladar al caso que nos ocupa, si bien ahora hemos de tener en cuenta algunas consideraciones adicionales. La primera es consecuencia del empaquetado del chip, puesto que muchas veces no es posible incluir un pivote para cada línea o bit de dirección. La solución en estos casos es multiplexar los bits de dirección aprovechando la estructura matricial de la memoria, es decir, primero se suministran los bits de dirección que identifican la fila donde se encuentra nuestra palabra y después se suministran los bits que identifican la columna. Para gestionarlo, utilizamos las señales RAS y CAS, que indican, respectivamente, cuándo los bits de la fila o de la columna están disponibles. Asociado a estas señales, definimos los parámetros temporales siguientes:

- Tiempo de ciclo del RAS: indica el periodo de esta señal.

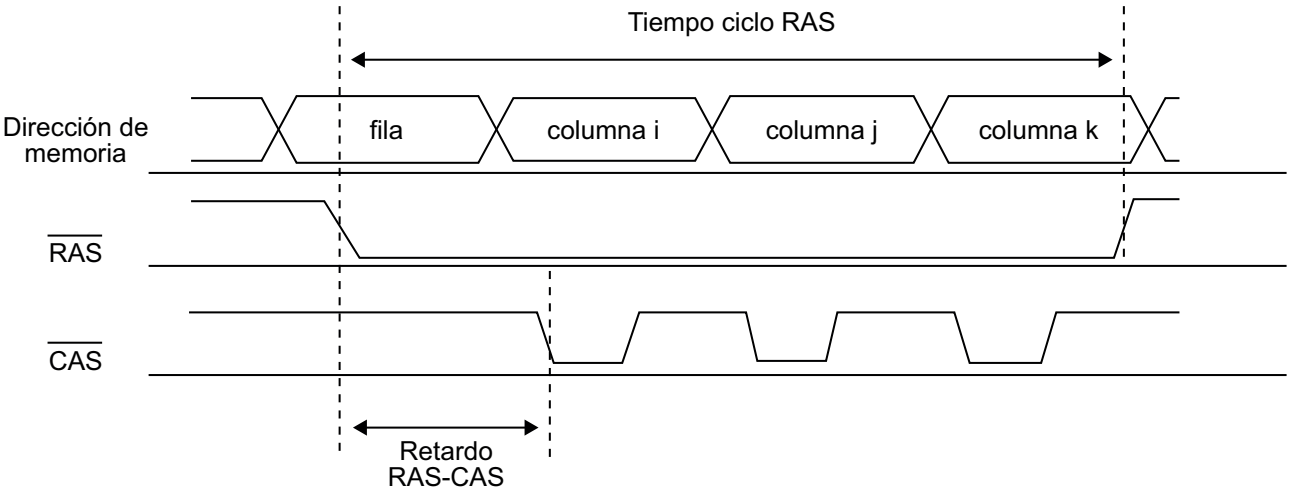
- Retardo RAS-CAS: indica la separación que hay entre las señales RAS y CAS.

Figura 14. Temporización de las señales de dirección en una memoria DRAM



La figura muestra la señalización multiplexada de direcciones junto a los parámetros que acabamos de definir. Finalmente, hay que comentar que los diferentes tipos de memoria DRAM se distinguen básicamente por cómo se organizan internamente y por cómo se accede a los datos. Por poner un ejemplo, mencionamos las memorias DRAM que soportan EDO (*extended data output*) y que permiten acceder a varias palabras a la vez. En particular, deben ser palabras que tengan la parte de la dirección correspondiente a la fila en común y cuya señalización empleada sea la de la figura siguiente.

Figura 15. Temporización de las señales en una memoria DRAM con EDO

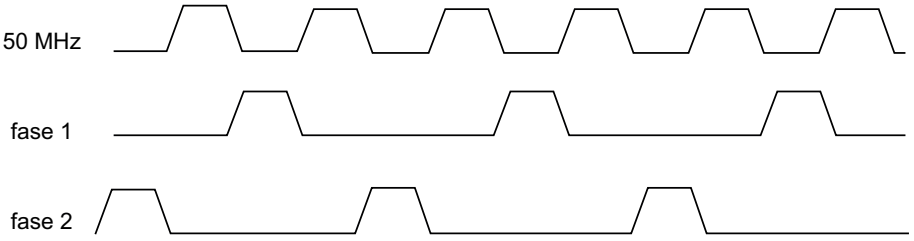


La segunda consideración es consecuencia de la necesidad de refresco de las memorias. Definiremos el periodo de refresco como el periodo temporal máximo con el que las celdas de memoria deben ser refrescadas para evitar pérdidas de información. Existen varias maneras de refrescar la memoria y hay que hacerlo de la manera más eficiente posible, puesto que el refresco en sí es un trabajo extra que no nos aporta ningún beneficio desde el punto de vista funcional. Esto implica toda una circuitería adicional que se encargue de recorrer las posiciones de memoria con los tiempos adecuados y de evitar conflictos con el funcionamiento normal de la memoria. A con-

tinuación, presentamos un posible diseño para una memoria de 4 M palabras de 16 bits cada una.

La memoria está organizada en 4 K filas por 1 K columna y es posible refrescar al mismo tiempo todas las palabras que constituyen una única fila, con lo cual conseguimos que el refresco sea más eficiente. En el supuesto que nos ocupa, asumimos que el periodo de refresco es de 64 ms y que disponemos de un reloj de 50 MHz con dos fases diferentes, tal como muestra la figura siguiente.

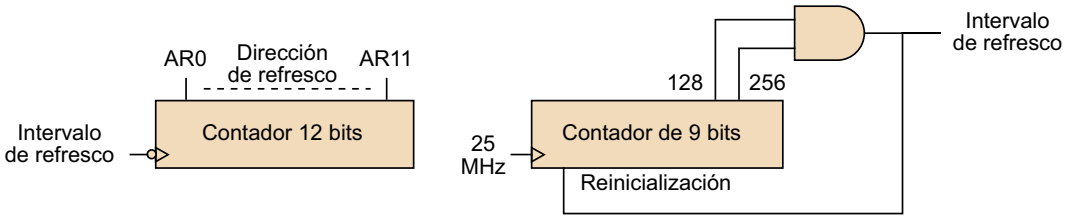
Figura 16. Reloj de dos fases empleado



En primer lugar, nos ocupamos del tiempo de refresco. Si tenemos en cuenta las 4 K filas de la memoria, vemos que hay que refrescar una fila cada 16 μs. Esto se puede conseguir con un contador de 9 bits alimentado por una de las fases del reloj (25 MHz). Fijémonos en que 400 golpes de reloj hacen los 16 μs que necesitamos. Si queremos incluir un cierto margen de seguridad y a la vez facilitar la implementación, podemos considerar 384 golpes de reloj, equivalentes a 15,36 μs. Poniendo una puerta AND entre los dos bits de más peso del contador conseguimos una señal que se mantiene a 0 hasta que no han transcurrido los 384 periodos de reloj deseados. Entonces, toma el valor 1 y se mantiene en él hasta que el contador no da la vuelta, es decir, cuando pasa de 511 a 0. Fijémonos en que, si no hiciéramos nada más, tendríamos una señal que se activaría cada 512 golpes de reloj, lo cual no nos interesa. Por lo tanto, una solución será optar por un contador con entrada de inicialización síncrona, que será alimentada por la señal de salida de la puerta AND, el mismo que marcará el intervalo de refresco deseado.

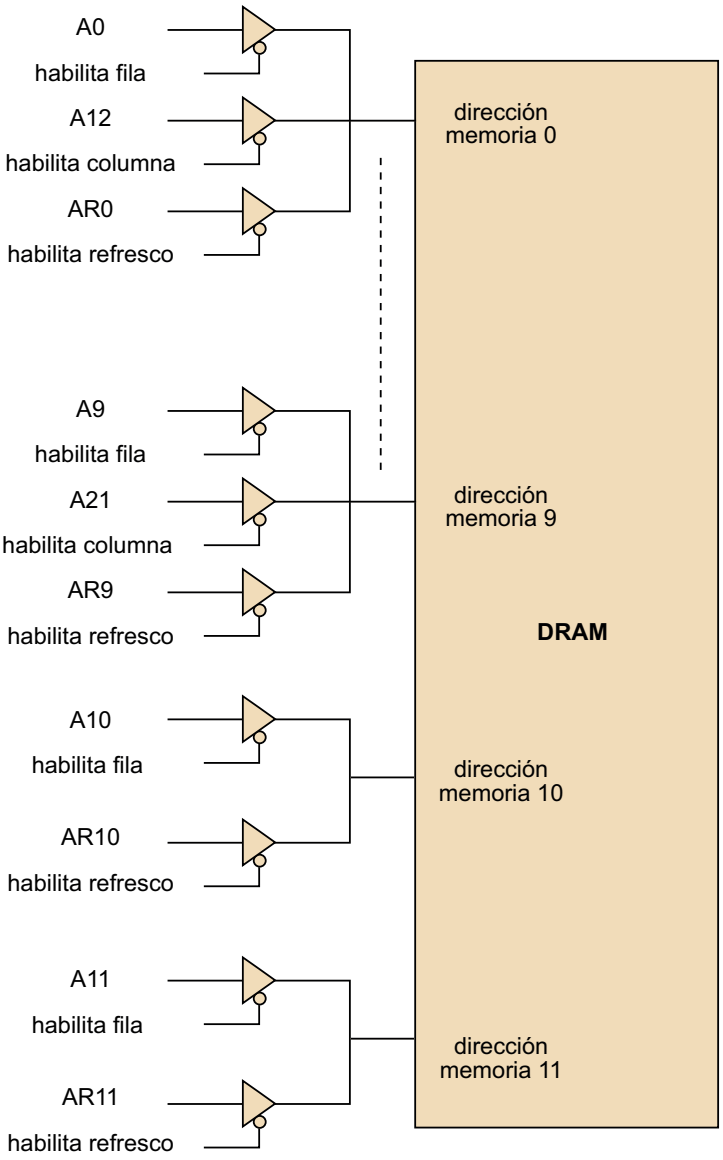
Una vez establecido el tiempo de refresco, hemos de calcular la posición o posiciones de memoria que hay que refrescar. En el supuesto que nos ocupa, donde refrescamos una fila de memoria cada vez, un contador de 12 bits será suficiente para recorrer todas las posiciones, es decir, las 4 K filas. Además, activaremos el contador en cada flanco de bajada de la señal intervalo de refresco, puesto que de este modo la dirección que hay que refrescar estará disponible con toda certeza a la hora de ejecutar la operación. En la figura 34 podemos ver tanto el generador de direcciones de refresco como el generador del intervalo de refresco.

Figura 17. Generador de direcciones de refresco (izquierda) y del intervalo de refresco (derecha)



Como tenemos 12 bits para indicar la dirección en la memoria DRAM y estos bits pueden tener hasta tres funciones diferentes (fila de memoria para lectura/escritura, columna de memoria para lectura/escritura o fila para refresco), utilizaremos *buffers* de salida para diferenciar cada una de estas funciones y evitar conflictos, tal como se muestra en la figura siguiente. De este modo, podremos dejar los *buffers* en estado de alta impedancia cuando no se utilicen. Esto implica el diseño de circuitería adicional para generar las señales de control de los *buffers*.

Figura 18. Control de las direcciones (normales y refresco)



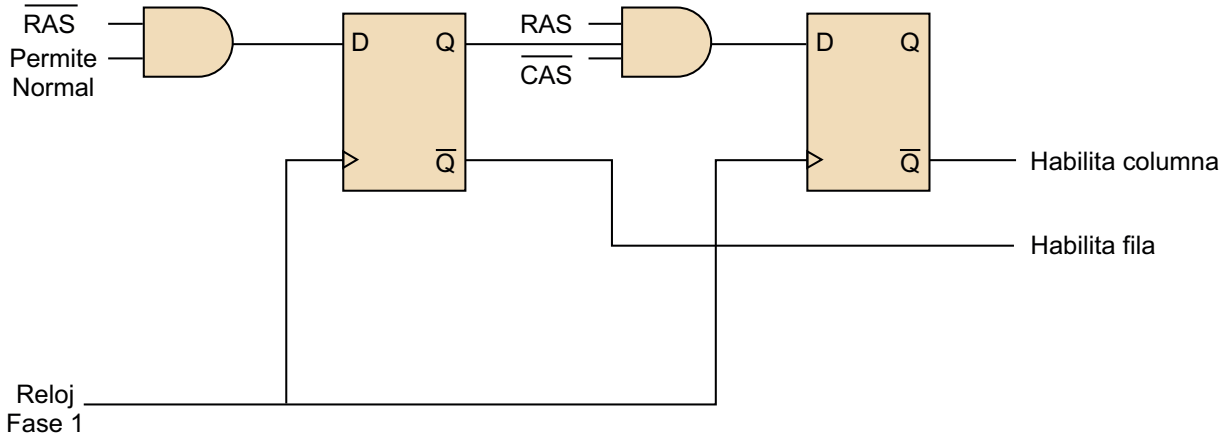
Ejemplo

En la figura siguiente vemos una manera de generar las señales de habilitación de fila y columna cuando la señal Permite Normal (que permite las operaciones normales de lectura o escritura) es válida. En la misma figura, podemos ver un diagrama temporal de las señales que intervienen. En este ejemplo, las señales RAS y CAS provenientes del procesador están sincronizados con la fase 2 del reloj, mientras que las señales de habilitación que generamos son gobernadas por la fase 1. Este mecanismo evita interpretar erróneamente las señales tomando algún valor del transitorio.

Llegados a este punto, hemos sido capaces de adecuar la memoria DRAM para que pueda soportar procesos de lectura/escritura y también de refresco. No obstante, todavía falta lo más importante: la gestión temporal de estos procesos. Fijémonos en que el refresco de la memoria es una acción más controlada que las operaciones de lectura/escritura habituales, puesto que podemos predecir qué posiciones de memoria se verán afectadas en cada momento. En cambio, el acceso normal puede ser más arbitrario, y dejar abierta la posibilidad de querer acceder a memoria y refrescar al mismo tiempo. Para evitar este tipo de conflictos, deberemos incluir un módulo de arbitraje. Una posible definición funcional de este es la siguiente:

- Cuando empieza una operación de lectura/escritura, se deja acabar.
- Si una operación de refresco ha empezado, se deja acabar recordando la operación normal.
- Si las dos operaciones coinciden, se da prioridad a la operación normal.

Figura 19. Ejemplo de generación de las señales de habilitación de los *buffers* de dirección



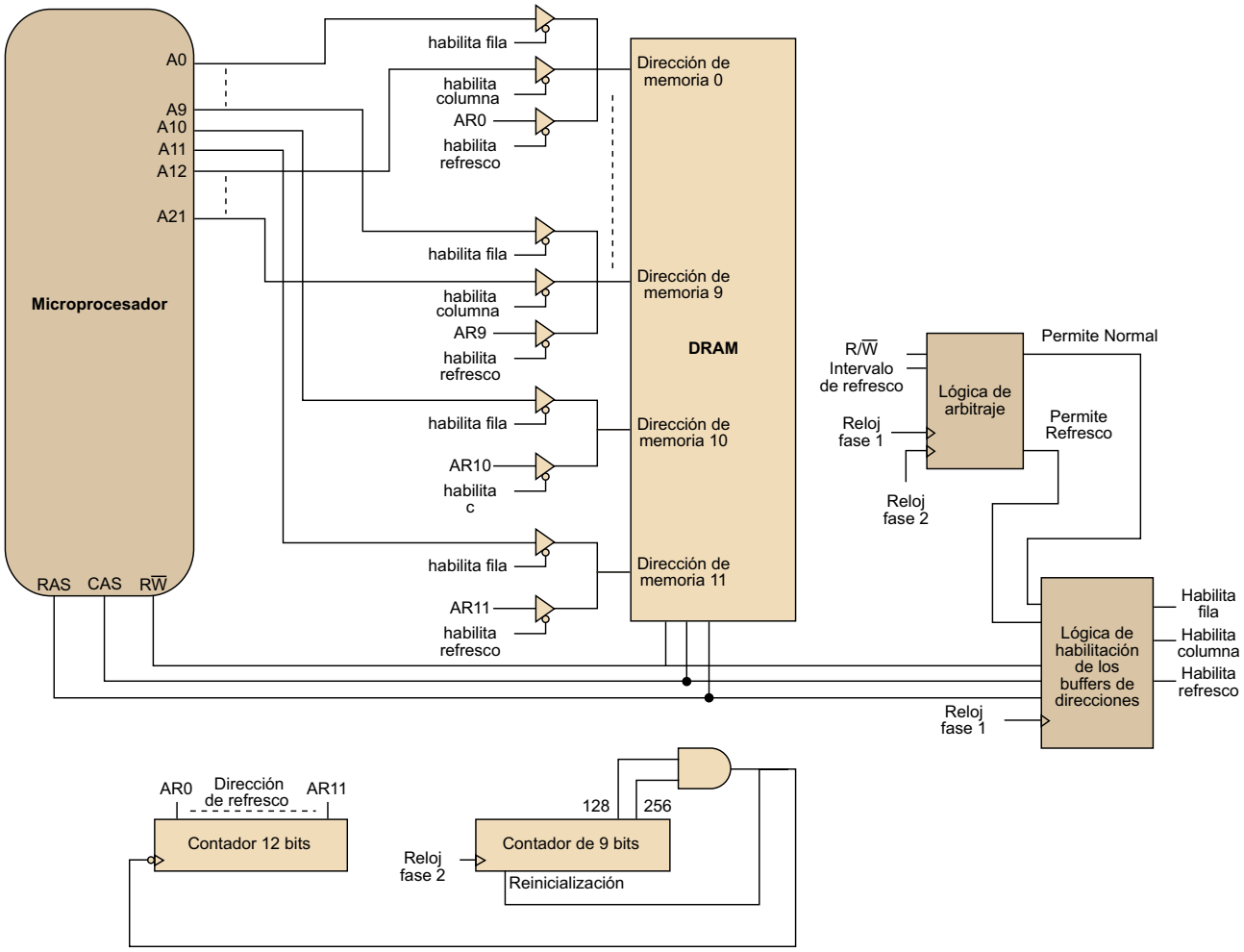
Finalmente, la figura siguiente muestra el diseño completo del sistema de memoria basado en DRAM, en el que usamos las señales siguientes:

- **Lectura-escritura**: indica que se ha iniciado un proceso normal de lectura/ escritura.

- **Permite Normal**: indica que el módulo de arbitraje permite las operaciones normales de lectura/escritura.

- **Permite Refresco**: indica que el módulo de arbitraje permite las operaciones de refresco.

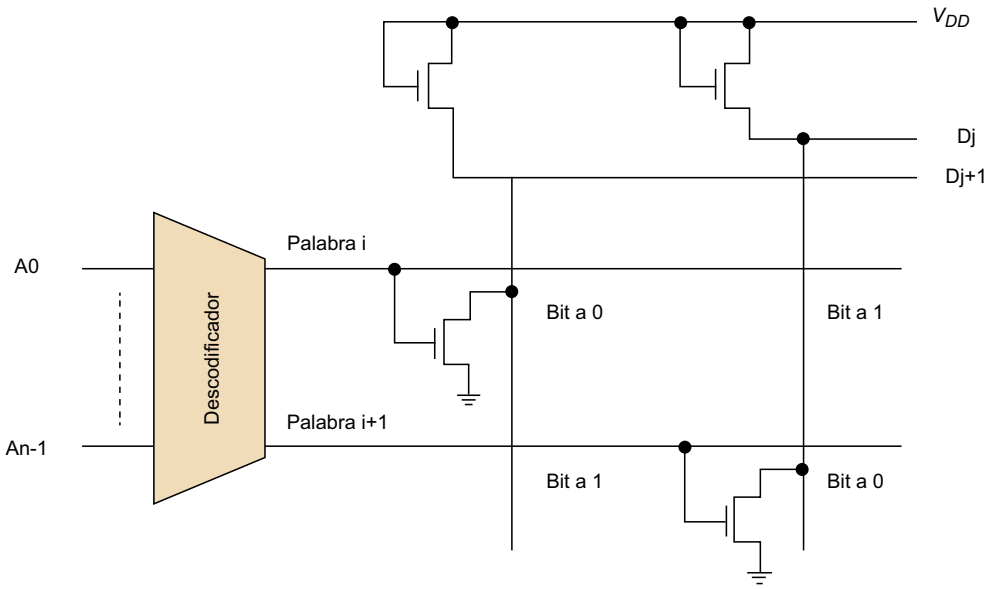
Figura 20. Sistema de memoria DRAM completo



3.2. ROM/FLASH

En este apartado consideraremos la memoria ROM un dispositivo de solo lectura, es decir, obviaremos la parte de programación o escritura. En cuanto a la implementación, se puede conseguir almacenar un bit con la presencia o no de un transistor. En la figura siguiente vemos un posible esquema de memoria ROM utilizando únicamente transistores NMOS. Fijémonos en que cada fila interna de la memoria es activada o seleccionada por una única combinación de los bits de entrada, que forman la dirección. *A priori*, todas las columnas están cargadas a un valor lógico 1 (transistores NMOS con la puerta conectada a la tensión de referencia) y, por lo tanto, si no hay ningún otro elemento, todos los bits de la palabra seleccionada tomarían un valor lógico 1. Sin embargo, la colocación de un transistor NMOS conectado a tierra provoca que la tensión final de la celda o bit en cuestión se sitúe en un valor lógico 0, puesto que se drena la corriente a masa.

Figura 21. Memoria ROM implementada con transistores NMOS



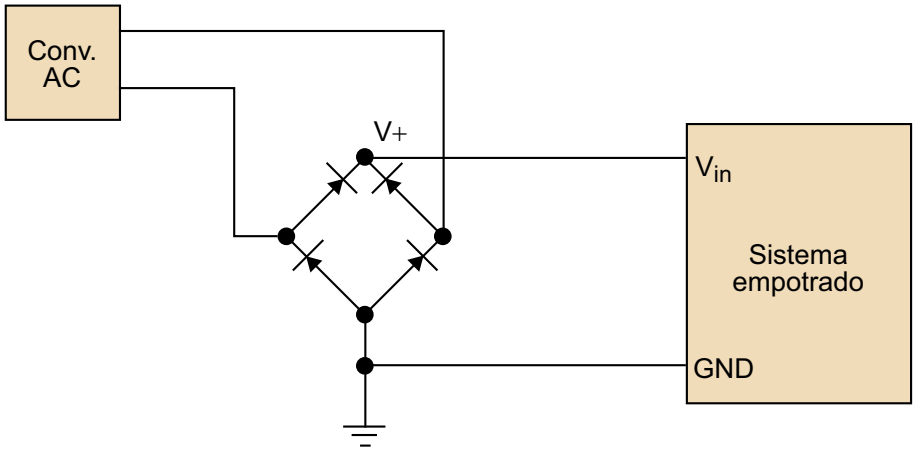
En cuanto a la interacción con el dispositivo, deberemos indicar en primer lugar la dirección de la que se quiere hacer la lectura y también de la que queremos hacer la operación.

4. Alimentación

Lógicamente, cualquier dispositivo electrónico necesita una fuente de alimentación para poder funcionar. Dependiendo del tipo de dispositivo, la selección del tipo de fuente y el diseño del subsistema de alimentación varía.

En caso de que el dispositivo no necesite ser portable y disponga de una toma de corriente cercana, la mejor opción es conectarlo directamente a la red eléctrica. Al trabajar la red eléctrica en corriente alterna (ca) y con altos niveles de voltaje (230 V en el caso de Europa), se debe transformar para que pueda ser útil para el sistema empotrado, que requiere corriente continua (cc) y un voltaje bastante inferior. Una de las opciones más viables para hacer esto, en términos de simplicidad y bajo coste, es el uso de un adaptador ca, que es fácilmente adquirible por cualquier distribuidor. Concretamente, es como el que se utiliza en el caso de los teléfonos móviles y otros dispositivos de informática de consumo. Estos dispositivos transforman la corriente alterna de entrada en corriente continua con un voltaje que suele estar entre los 5 y los 12 V. Un aspecto que se debe tener en cuenta con el uso de este tipo de adaptadores es que el conector que tienen de salida (que se conecta al sistema empotrado) puede tener diferente polaridades según el fabricante. Por lo tanto, antes de conectarlo al sistema empotrado, hay que asegurarse de que la polaridad sea la correcta para no provocar desperfectos en el sistema empotrado. Esto se puede verificar mediante el uso de un **multímetro**. Otra opción es proveer el sistema empotrado de un **puente de diodos**. Un puente de diodos, o puente rectificador, es un circuito que da siempre la misma polaridad a la salida, independientemente de la polaridad de entrada (podéis ver la figura siguiente, en la que las entradas V_{in} y GND se refieren a las entradas de alimentación y masa, respectivamente). Esto se debe a la característica propia de los diodos, que solo dejan pasar corriente eléctrica en una única dirección.

Figura 22. Esquema del subsistema de alimentación diseñado



En el resto de los casos, la opción más común es el uso de baterías. En primer lugar, se debe hacer una buena selección de la batería para asegurar que ofrece el voltaje correcto y un nivel de corriente suficiente. En caso contrario, el sistema puede funcionar erróneamente o directamente no funcionar. Por otro lado, se debe considerar, aparte del nivel de corriente media que ofrece la batería, cuál es la corriente de pico que puede ofrecer.

Algunos sistemas empotrados pueden necesitar solo una corriente media de 20 mA, pero requerir en algunos momentos corrientes de pico de 100 mA. Esto suele suceder en sistemas con memorias flash, que requieren altas corrientes cuando llevan a cabo operaciones de escritura.

En segundo lugar, se debe tener un sistema bien diseñado, puesto que esto puede condicionar bastante la duración de la batería, que puede ir desde unos minutos (si está diseñado bastante mal) hasta algunos años, según el tipo de hardware del sistema empotrado y la aplicación. Para llevar a cabo este diseño, el primer paso es tener un sistema empotrado que utilice dispositivos de bajo consumo. El consumo de potencia entre chips diferentes puede variar mucho, y muy a menudo hay versiones de bajo consumo del dispositivo que se quiere integrar. Por otro lado, muchos periféricos y chips de memoria entran de manera automática en modo de bajo consumo cuando no están en uso. Otros pueden ser puestos en este modo al accionar una entrada digital o mediante una orden de software. Dependiendo de la aplicación, una opción también puede ser implementar una solución en la que se separen los circuitos de alimentación de algunos dispositivos del sistema para poderlos desconectar, vía control de software, cuando estos no sean utilizados. Finalmente, algunos dispositivos de muy bajo consumo, como los sensores, necesitan muy poco nivel de corriente para funcionar. Por lo tanto, una opción es alimentarlos directamente con las líneas de entrada y salida propias del microcontrolador o microprocesador del sistema. Es decir, se utiliza una línea de entrada y salida para alimentar el dispositivo, puesto que estas líneas pueden dar niveles de corrientes suficientes (20 mA en algunos casos) y, simultáneamente, se aprovecha esta línea para activar o desactivar el dispositivo cuando sea necesario.

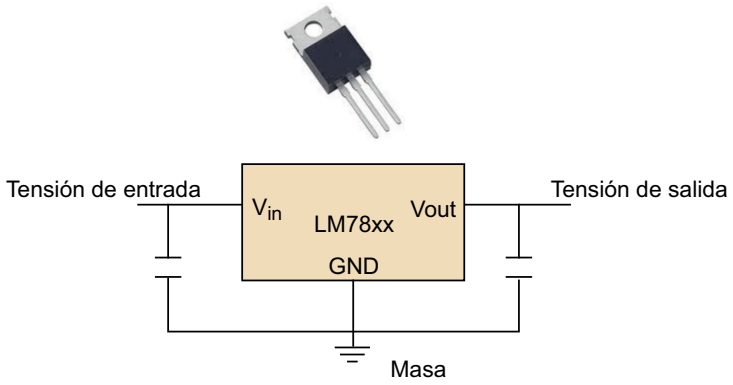
Dejando ahora de lado el tipo de fuente de alimentación que utilice el sistema, es recomendable que el subsistema de alimentación del sistema empotrado use reguladores de voltaje. Un regulador de voltaje es un dispositivo que asegura un nivel fijo de tensión continua a su salida a pesar de que la tensión a la entrada adopte diferentes valores o sufra variaciones o ruido. Esta opción es recomendable para asegurar que el nivel de tensión que se aplica a los componentes del sistema empotrado es la deseada y se mantiene constante. A pesar de que muchos de los componentes son reacios a variaciones del nivel de tensión, otros son muy sensibles, como por ejemplo los convertidores analógico-digitales que hacen la conversión de la entrada analógica comparando el nivel de tensión de entrada con el nivel de tensión de alimentación del mismo convertidor. Es decir, la cuantificación de la señal de entrada se efectúa tomando la tensión de alimentación como referencia. Por otro lado, si el sistema empotrado trabaja con

baterías, el regulador puede ser un mecanismo eficaz para combatir las variaciones de tensiones ocasionadas por el mismo consumo de la batería. En la figura siguiente se muestra un ejemplo de regulador comercial, en el que las tres patas se corresponden con la tensión de entrada (referida normalmente como Vin), la tensión de salida (Vout) y la conexión de masa (GND). Este regulador concreto se conoce con el nombre de LM78xx, lo fabrican varias compañías (como Fairchild y ST Microelectronics) y xx se refiere al voltaje de salida. En la parte derecha de la misma figura se muestra el circuito que se suele utilizar para su integración, que se conoce como regulador lineal. Tal como se puede ver, se utiliza un condensador a la entrada y otro a la salida. En esta configuración, los condensadores se denominan condensadores desacopladores y su objetivo es filtrar el ruido electromagnético presente en la línea de alimentación. Hay que mencionar también que algunos reguladores tienen entradas adicionales que actúan como interruptores para la activación o desactivación del circuito. Esto puede simplificar el proceso de optimización de consumo comentado en el párrafo anterior en cuanto a desactivar dispositivos que no estén siendo utilizados.

Ejemplo

En el caso de LM7805, la salida sería igual a 5 V.

Figura 23. Regulador comercial (arriba) y circuito de un regulador lineal (abajo)



Bibliografía

DataQ Instruments (2017). «How To Make 4-20 mA Current Loop Measurements».
URL: https://www.dataq.com/blog/data-acquisition/ 4-20-ma-current-loop-measurements/.

Jonathan Valdez, J. B. (2015). *Application Report SLVA704*. Texas Instruments.
URL: http://www.ti.com/lit/an/slva704/slva704.pdf.

MODICON Inc. (1996). *Modicon Modbus ProtocolReference Guide*. MODICON, Inc.
URL: http://modbus.org/docs/PI_MBUS_300.pdf.

Silicon Labs (2015a). *AN0012: General Purpose Input Output*. Silicon Labs.
URL: https://www.silabs.com/documents/public/application-notes/AN0012.pdf.