

Sistemas Electrónicos Digitales Avanzados



Introducción al protocolo I2C

© 2013 Departamento de Electrónica. Universidad de Alcalá

D. Julio Pastor Mendoza (pastor@depeca.uah.es)



Universidad
de Alcalá



Protocolo de Comunicación I²C



B
U
S

I
2
C

Índice de la Lección

- **Introducción**
- **Características del bus I²C**
- **Conexión de dispositivos al bus: nivel físico**
- **Intercambio de información: nivel de enlace**
- **Generación del protocolo desde un microcontrolador**
- **Ejemplos de dispositivos que utiliza el bus I²C**
- **Unidad I2C del LPC17xx**
- **Conclusiones**
- **Bibliografía**



Protocolo de Comunicación I²C



B
U
S

I
2
C

Índice de la Lección

- **Introducción**
- **Características del bus I²C**
- **Conexión de dispositivos al bus: nivel físico**
- **Intercambio de información: nivel de enlace**
- **Generación del protocolo desde un microcontrolador**
- **Ejemplos de dispositivos que utiliza el bus I²C**
- **Conclusiones**
- **Bibliografía**



Introducción



B U S I 2 C

Interconexión de dispositivos mediante bus serie

Ventajas

- Pocos cables de interconexión
- Componentes con encapsulado reducido
- Tarjetas reducidas
- Conexión de dispositivos a distancia

Inconvenientes

- Velocidad inferior a un bus paralelo
- Disponibilidad de circuitos que soporten el bus

Parámetros generales

- Número de hilos de conexión
- Velocidad (bits/segundo)
- Distancia máxima y número de dispositivos
- Protocolo de acceso al medio compartido
- Política de direccionamiento



Introducción



B U S I 2 C

Origen del bus I²C (Inter Integrated Circuits Bus)

- **Desarrollado por Philips** a principios de los 80 como medio de interconexión entre una CPU y dispositivos periféricos dentro de la electrónica de consumo.
 - Simplificar las conexiones entre los periféricos (pistas, decodificadores, ..)
 - Aumentar de la inmunidad al ruido
 - Control de sistemas de audio y vídeo (baja velocidad)
- **Actualmente diseñan dispositivos** basados en I²C muchos fabricantes
 - Xicor, SGS-Thomson, Siemens, Intel, TI, Maxim, Atmel, Analog Devices



Protocolo de Comunicación I²C



B
U
S

I
2
C

Índice de la Lección

- Introducción
- **Características del bus I²C**
- Conexión de dispositivos al bus: nivel físico
- Intercambio de información: nivel de enlace
- Generación del protocolo desde un microcontrolador
- Ejemplos de dispositivos que utiliza el bus I²C
- Unidad I2C del LPC17xx
- Conclusiones
- Bibliografía



Características del bus I²C



B U S I 2 C

Características del bus I²C

- **Bus de comunicación síncrono**
 - La comunicación es controlada por una señal de reloj común
- **Bus formado por 2 hilos**
 - SDA (**S**erial **D**ata Line): datos
 - SCL (**S**erial **C**Lock line): reloj
 - También es necesaria una referencia común de masa
- **Velocidad de transmisión**
 - *Standard*: hasta 100 Kbits/s
 - *Fast*: hasta 400 Kbits/s
 - *High-speed*: hasta 3,4 Mbits/s
- **Cada dispositivo del bus tiene una dirección única**
- **Distancia y número de dispositivos**
 - Limitado por la capacidad del bus (inferior a 400pF). Normalmente 2 o 3 metros
- **Protocolo de acceso al bus:**
 - Maestro – esclavo
 - I²C soporta protocolo multimaestro



Protocolo de Comunicación I²C



B
U
S

I
2
C

Índice de la Lección

- Introducción
- Características del bus I²C
- **Conexión de dispositivos al bus: nivel físico**
- Intercambio de información: nivel de enlace
- Generación del protocolo desde un microcontrolador
- Ejemplos de dispositivos que utiliza el bus I²C
- Unidad I2C del LPC17xx
- Conclusiones
- Bibliografía



Conexión de dispositivos al bus: nivel físico

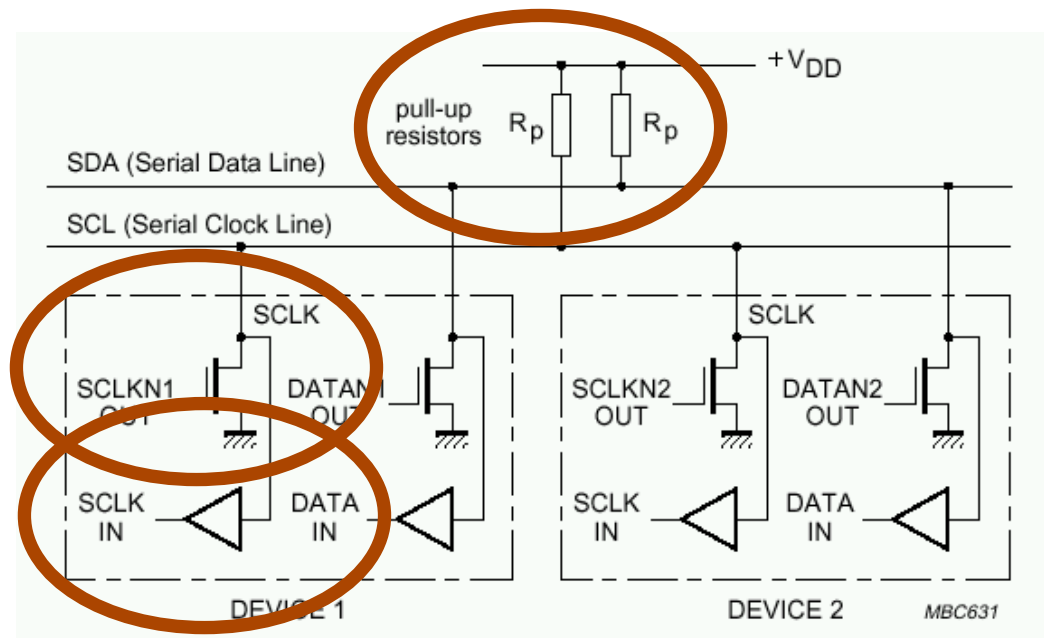


B
U
S

I
2
C

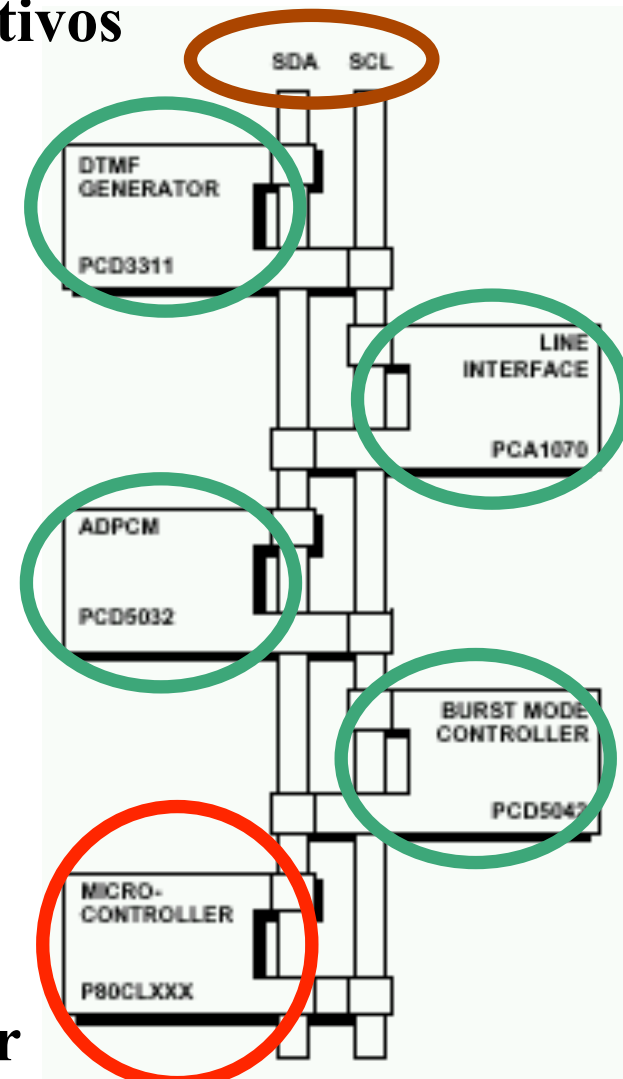
Conexión en bus

- Todos los dispositivos conectados a las mismas líneas
- Las salidas deben ser en colector o drenador abierto



Microcontrolador

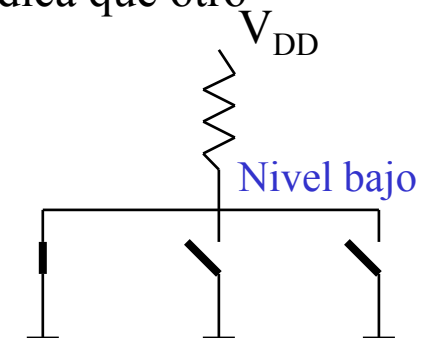
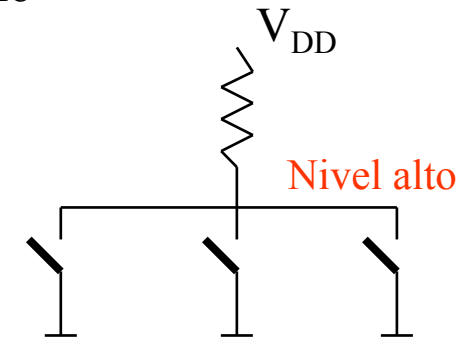
Dispositivos





Características de una conexión en colector abierto

- Permite conectar varias fuentes de datos a un mismo hilo
- Nivel alto en el bus
 - Si ningún dispositivo accede al bus
 - Si ningún dispositivo transmite un cero
- Nivel bajo en el bus
 - Si un dispositivo pone un nivel bajo
- Si dos dispositivos escriben a la vez siempre prevalecen los ceros
 - AND cableada
- Si un dispositivo escribe un nivel alto pero lee un cero indica que otro dispositivo está también accediendo al bus





Conexión de dispositivos al bus: nivel físico

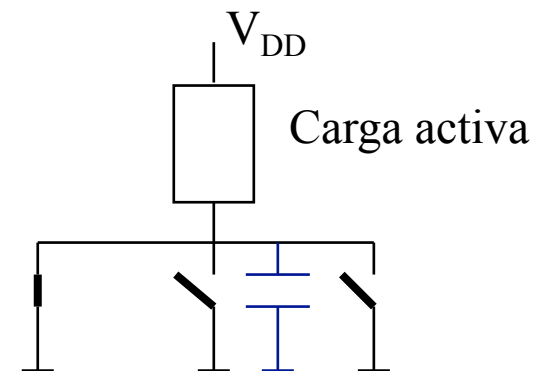
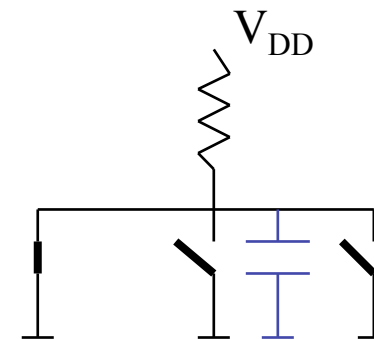
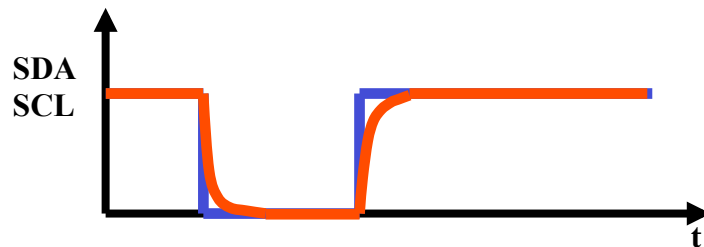
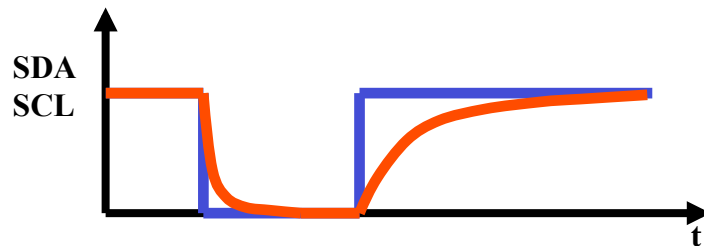


B
U
S

I
2
C

Inconvenientes de la conexión en colector abierto

- Las capacidades de la línea se cargan a través del *pull-up*
- Se puede solucionar utilizando una carga activa en lugar de un resistor





Índice de la Lección

- Introducción a la comunicación entre dispositivos mediante un bus serie
- Características del bus I²C
- Conexión de dispositivos al bus: nivel físico
- **Intercambio de información: nivel de enlace**
- Generación del protocolo desde un microcontrolador
- Ejemplos de dispositivos que utiliza el bus I²C
- Unidad I2C del LPC17xx
- Conclusiones
- Bibliografía



Protocolo de acceso al medio: maestro - esclavo

- El maestro controla la comunicación
 - Genera la señal de reloj del bus (SCL)
 - Inicia y termina la comunicación
 - Direcciona a los esclavos
 - Establece el sentido de la comunicación
- El protocolo requiere que cada byte de información sea confirmado por el destinatario

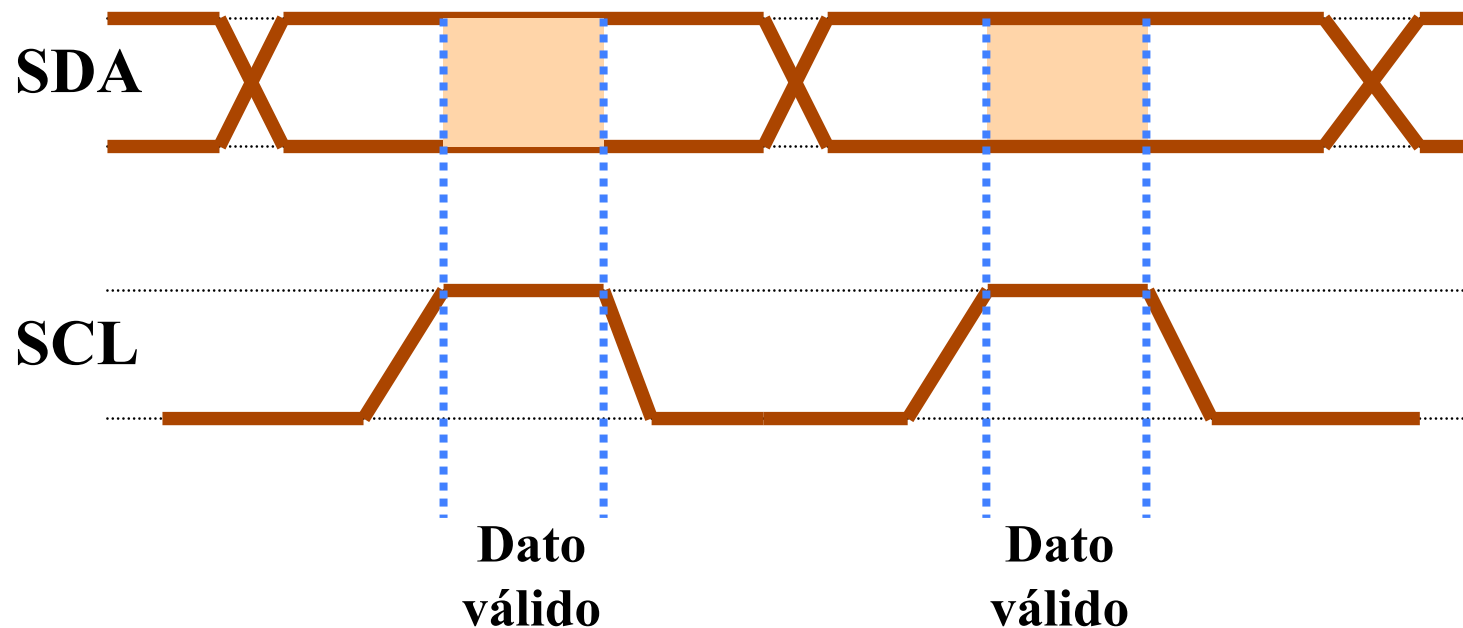
Nomenclatura

- **Emisor:** Dispositivo que envía datos al bus
- **Receptor:** Dispositivo que recibe datos del bus
- **Maestro:** Dispositivo que inicia una transferencia, genera las señales de reloj y termina la transferencia
- **Esclavo:** Dispositivo direccionado por un maestro



Transmisión de bits

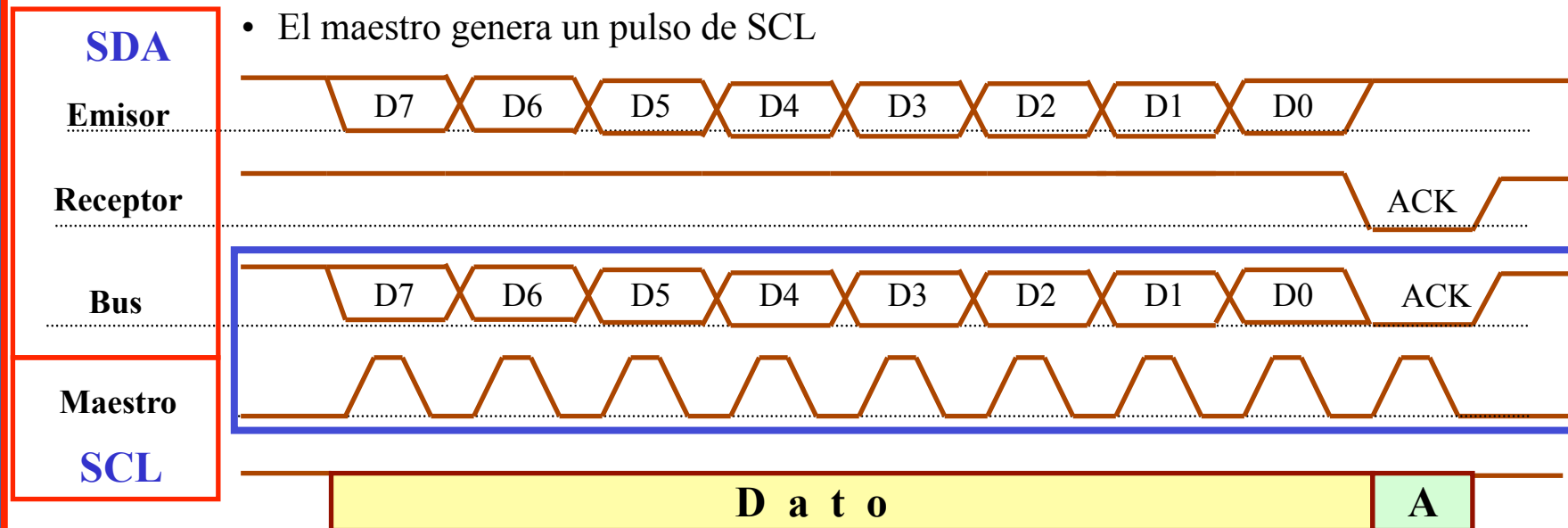
- Los bits de datos van por SDA
- Por cada bit de información es necesario un pulso de SCL
- Los datos sólo pueden cambiar cuando SCL está a nivel bajo





Transmisión de datos

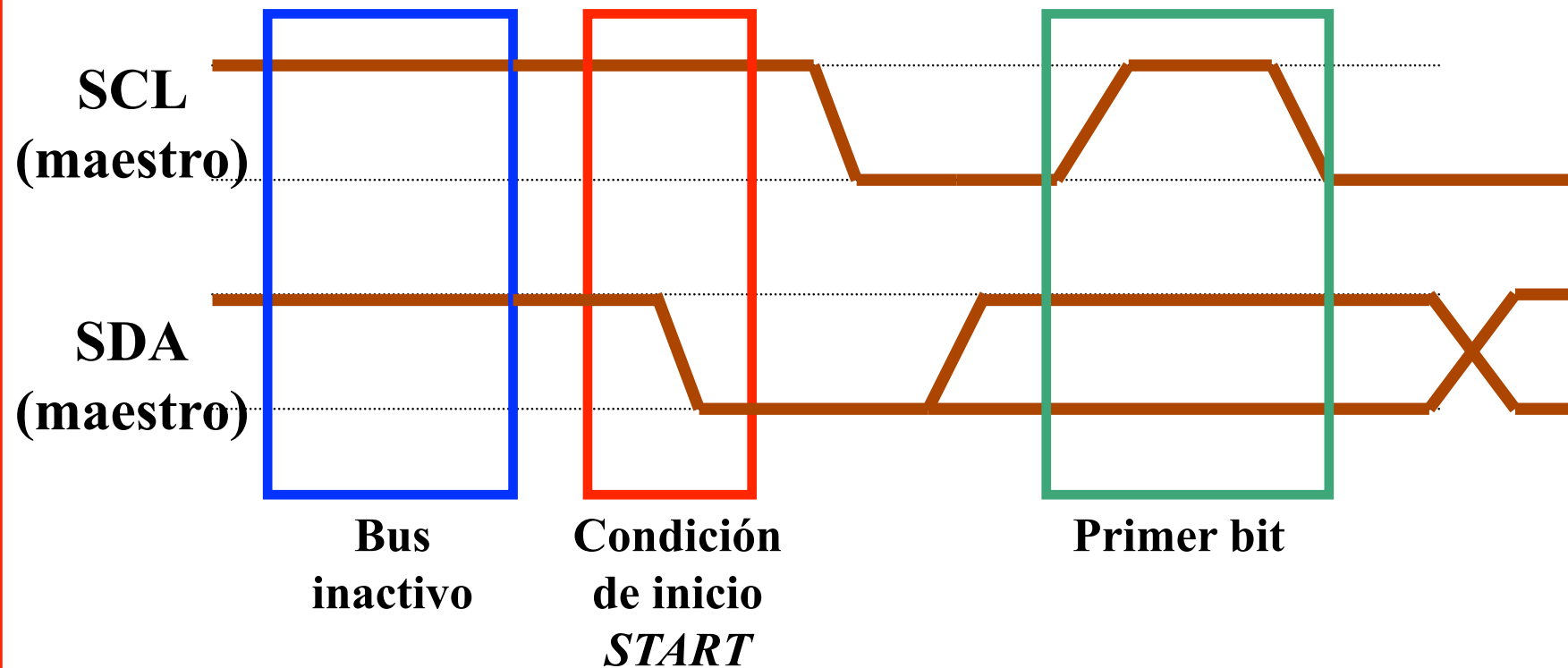
- La unidad básica de transmisión es el byte
- Las transferencias de datos son de 8 bits
- Cada byte enviado requiere una respuesta de confirmación
 - **ACK**: el destinatario (maestro o esclavo) mantiene SDA a nivel bajo durante un tiempo de bit (si no lo hace → **NACK**)
 - El maestro genera un pulso de SCL





Inicio de transmisión

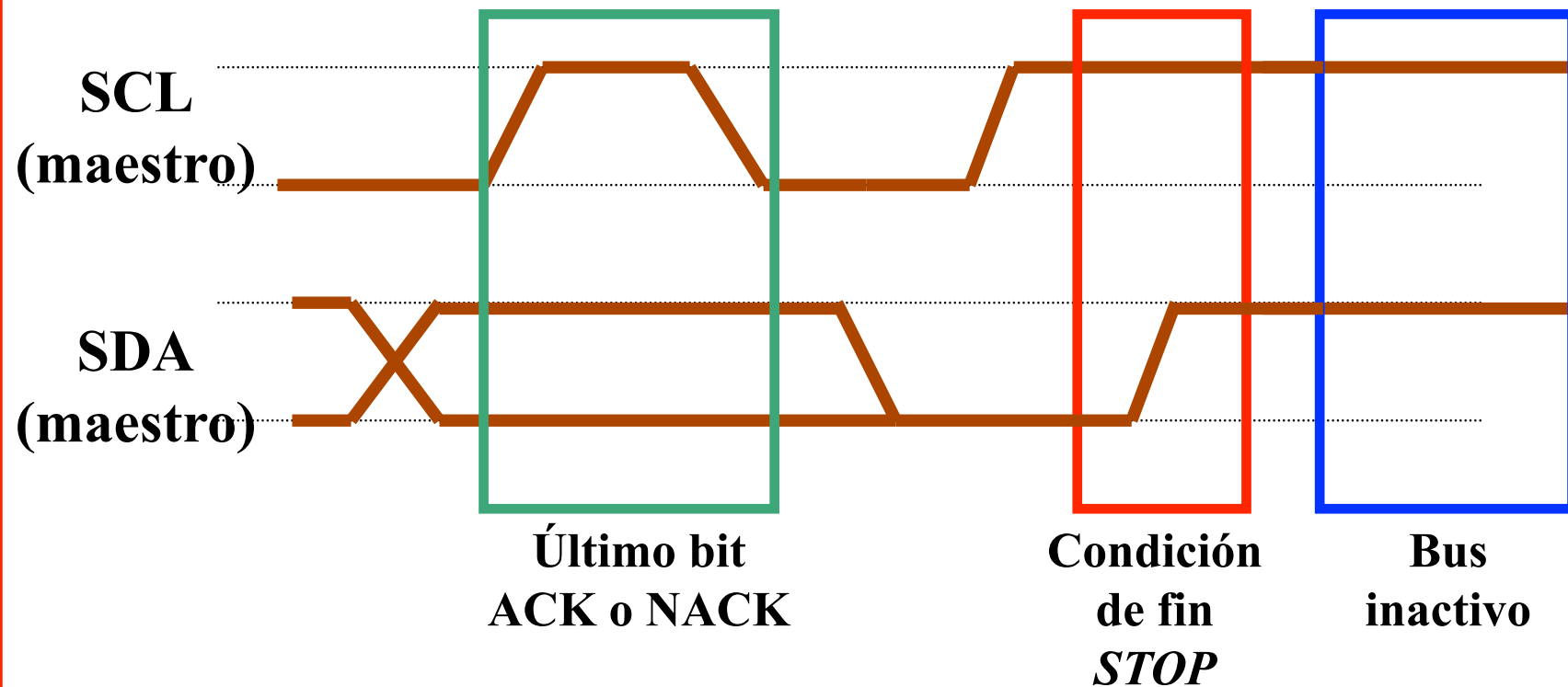
- La transmisión la inicia el maestro
- Flanco de bajada en SDA con SCL a nivel alto
- Cuando nadie accede al bus hay un nivel alto en SCL y SDA





Finalización de transmisión

- La transmisión la finaliza el maestro
- Flanco de subida en SDA con SCL a nivel alto

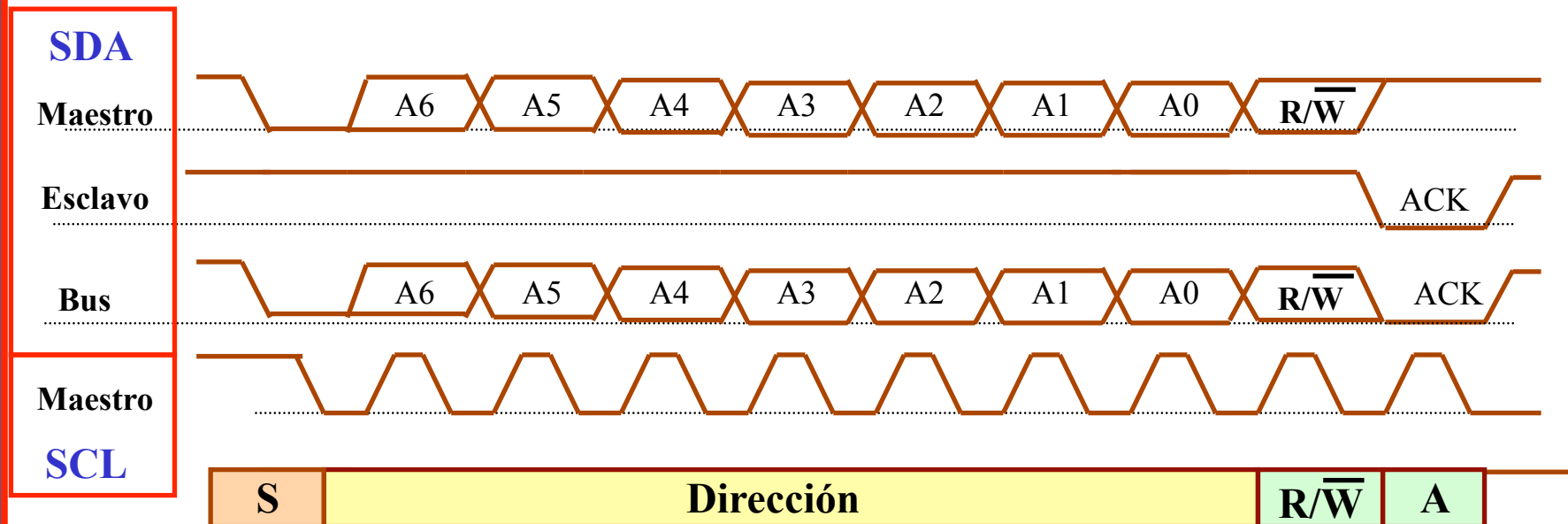




Intercambio de datos

Direccionamiento

- Tras la condición de inicio el maestro envía:
 - Dirección del esclavo (7 bits)
 - Comando de lectura o escritura ($R=1$ – $W=0$)





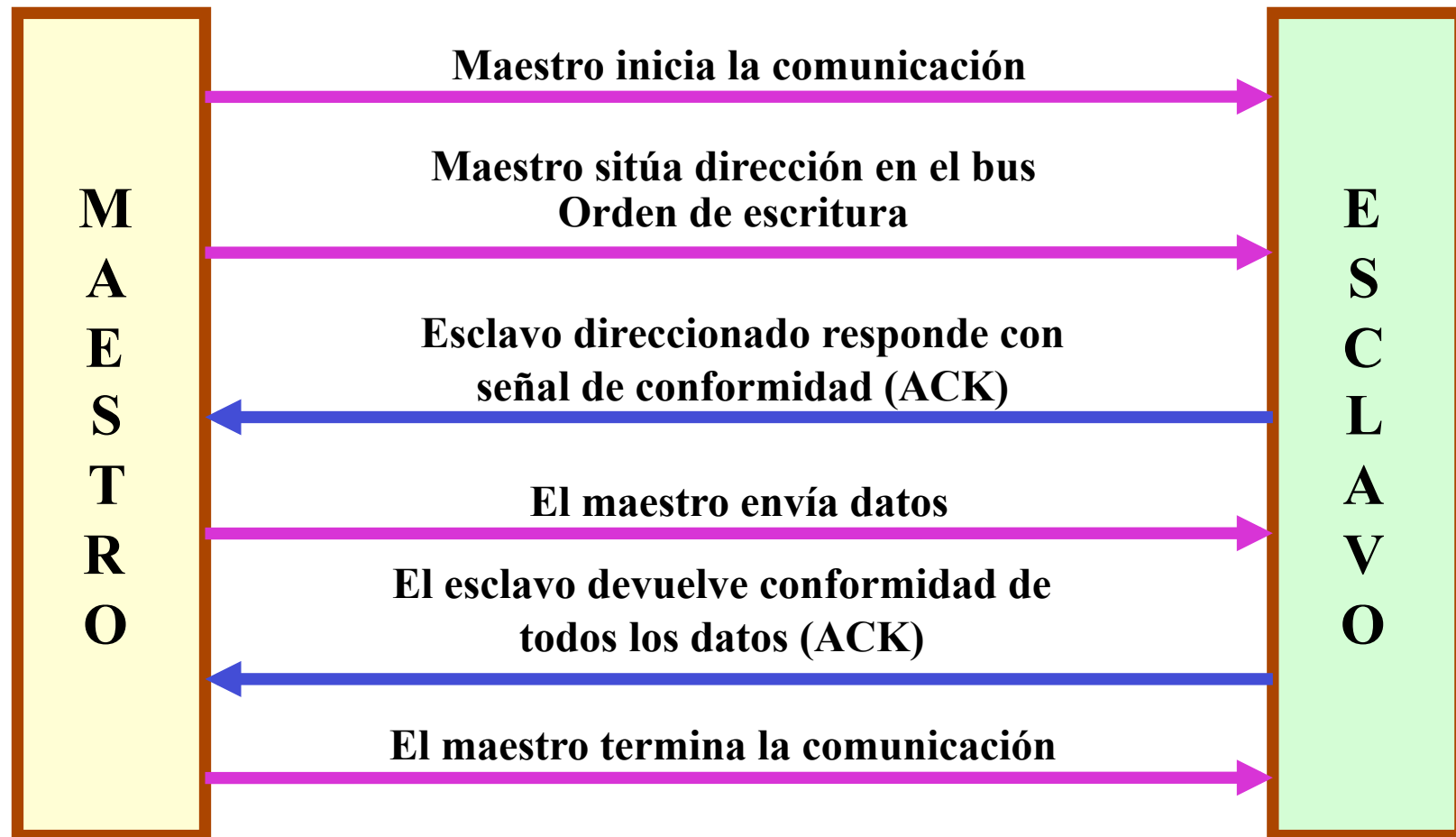
Intercambio de información: nivel de enlace



B
U
S

I
2
C

Maestro envía datos a un esclavo





Intercambio de información: nivel de enlace



B
U
S

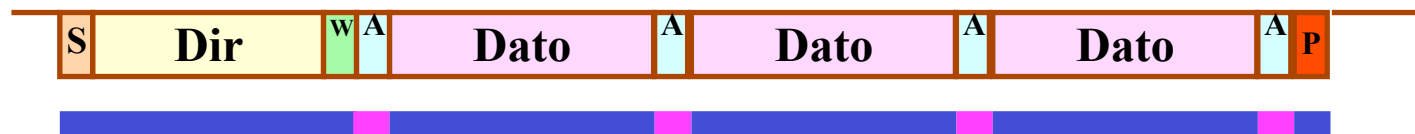
I
2
C

Intercambio de datos

Maestro envía un dato a un esclavo



Maestro envía varios datos a un esclavo





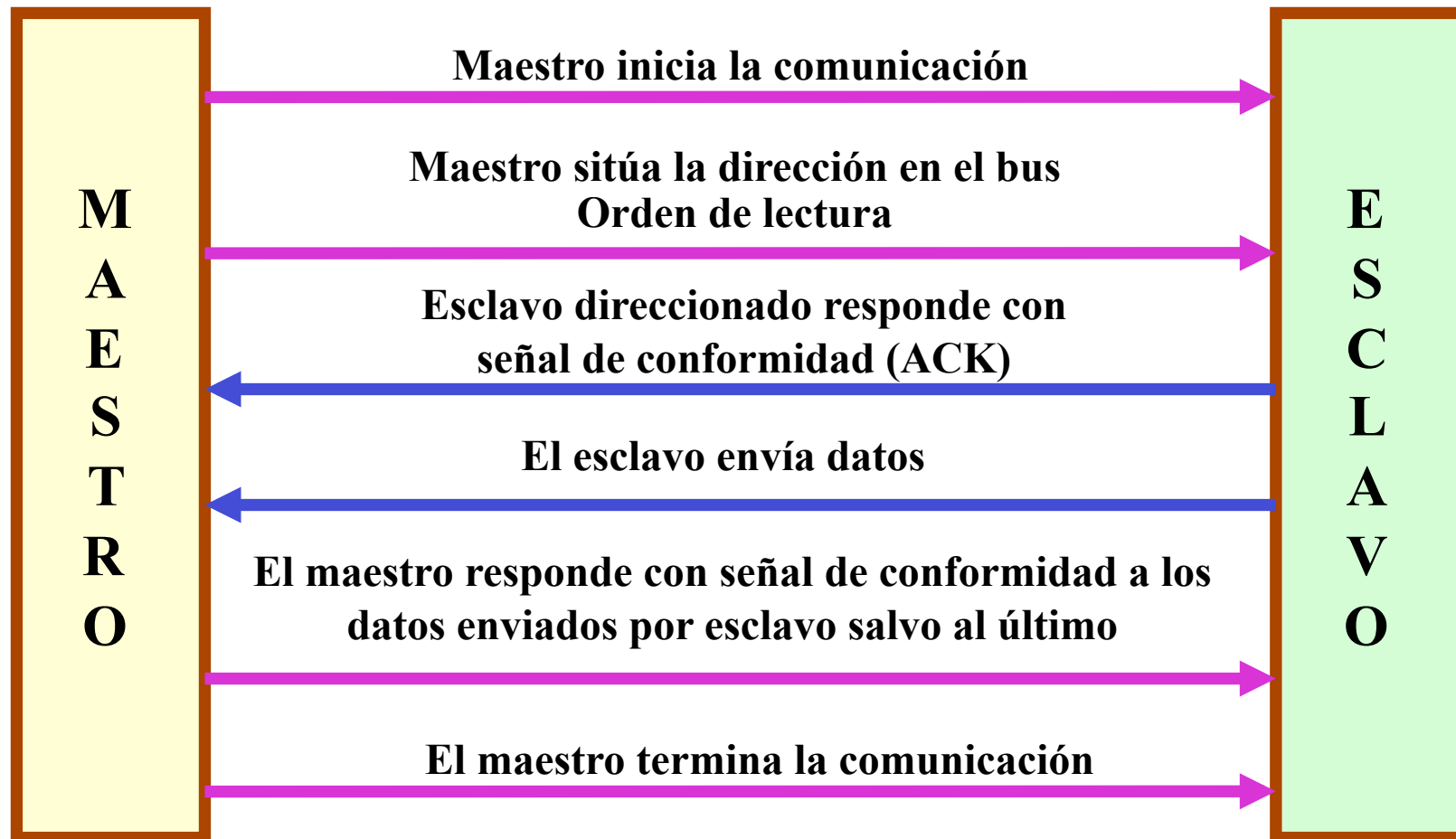
Intercambio de información: nivel de enlace



B
U
S

I
2
C

Maestro lee datos de un esclavo





Intercambio de información: nivel de enlace



B
U
S

I
2
C

Intercambio de datos

Maestro lee un dato de un esclavo



Maestro lee varios datos de un esclavo





Intercambio de información: nivel de enlace



B
U
S

I
2
C

Intercambio de datos

Maestro escribe y lee cambiando de dirección





Casos particulares de respuesta

- Esclavo ocupado y no responde al *ACK* de dirección
 - Maestro genera condición de *STOP*
- El esclavo quiere interrumpir una recepción periódica de datos
 - El esclavo no responde a un dato con *ACK*
 - El maestro genera la condición de *STOP*
- El maestro quiere interrumpir una recepción de datos de un esclavo
 - El maestro no responde con *ACK*
 - El esclavo deja de transmitir
 - El maestro genera condición de *STOP*



Protocolo multimaestro

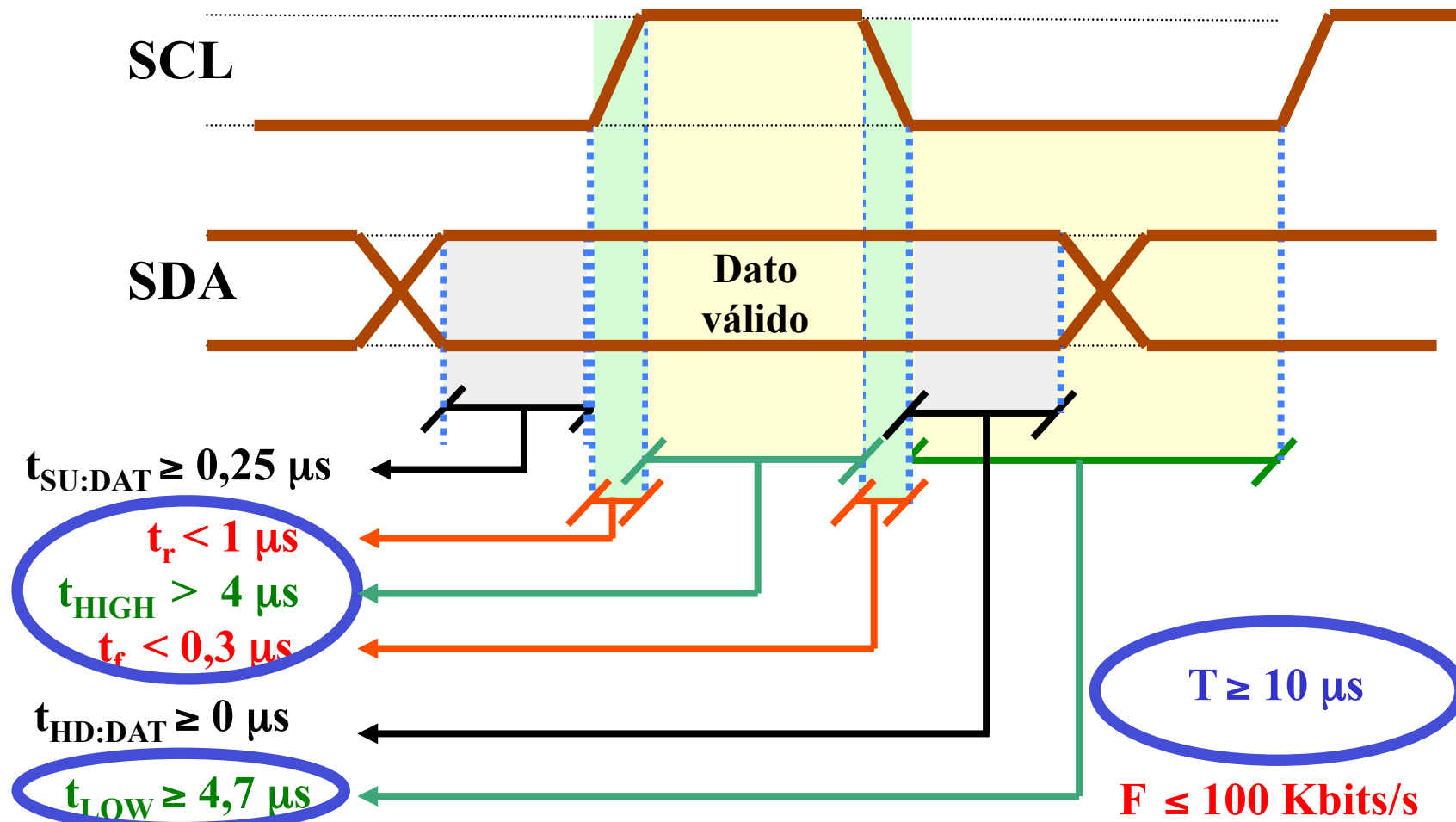
- Con dos maestros en el bus existe posibilidad de conflicto

Arbitración: procedimiento para asegurar que sólo un maestro tiene el control del bus en un instante

- Si un maestro está utilizando el bus no puede ser interrumpido por otro.
 - Desde *START* hasta *STOP*
- Si dos maestros intentan comenzar a utilizar el bus a la vez:
 - Conexión del bus en colector abierto → prevalecen los ceros
 - A la vez que ponen datos en el bus, escuchan la línea.
 - Si un maestro está intentando enviar un nivel alto y lee un nivel bajo
 - Existe otro maestro utilizando el bus
 - Deja de transmitir esperando que la línea quede libre (condición de *STOP*)

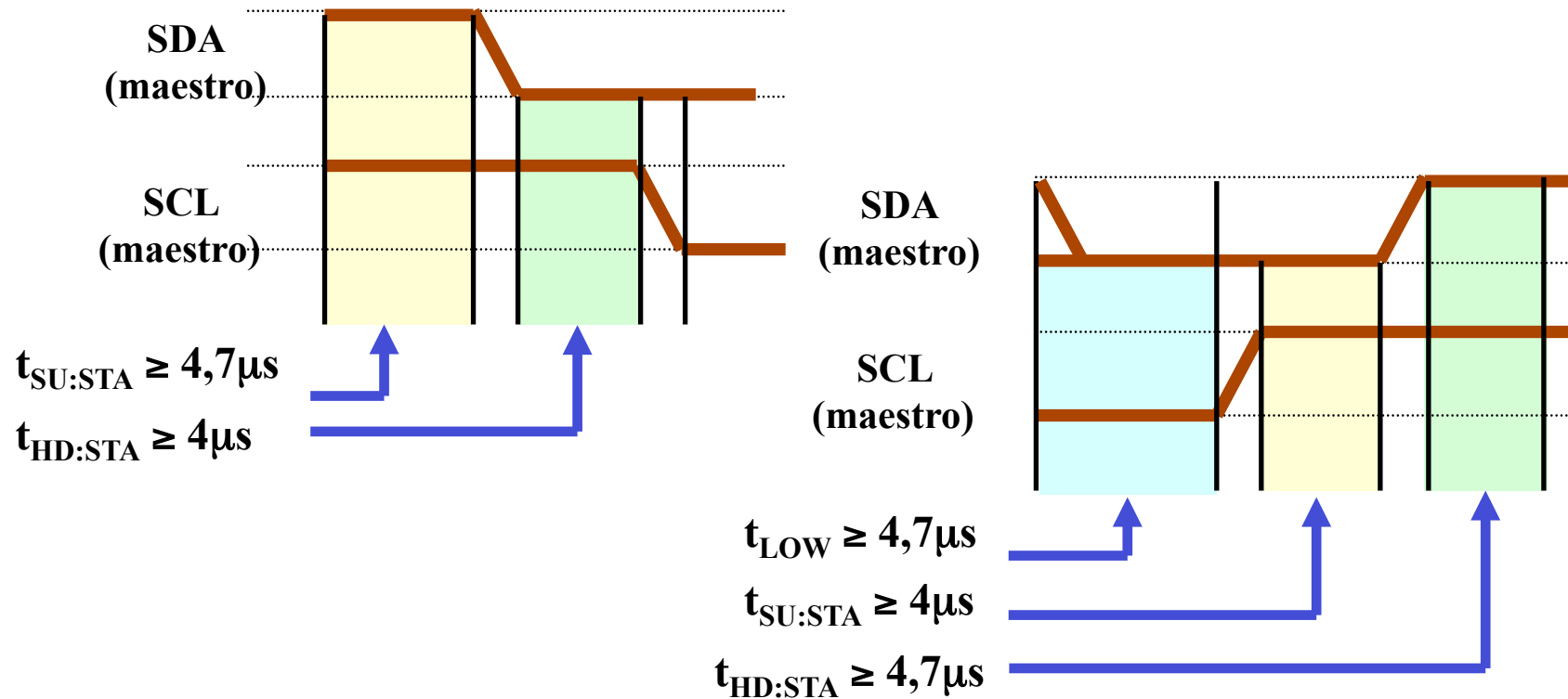


Transmisión de bits: Temporización





Temporización en *START* y en *STOP*





Índice de la Lección

- Introducción a la comunicación entre dispositivos mediante un bus serie
- Características del bus I²C
- Conexión de dispositivos al bus: nivel físico
- Intercambio de información: nivel de enlace
- **Generación del protocolo desde un microcontrolador**
- Ejemplos de dispositivos que utiliza el bus I²C
- Unidad I2C del LPC17xx
- Conclusiones
- Bibliografía



Control del bus I2C desde un microcontrolador

– Sin una unidad hardware específica

- El protocolo I2C se puede controlar utilizando puertos de entrada y salida
- El maestro controla la velocidad del bus
 - No debe generar señales demasiado rápidas
 - Puede ralentizar el bus cuando quiera
- Complicado utilizarlo como esclavo.
 - Debería cumplir la norma de tiempos
 - Sólo procesadores muy rápidos o dedicados.
- No se permite multimaestro

– Con una unidad hardware específica

- El procesador no debe preocuparse de gestionar el protocolo
- Se puede aprovechar toda la funcionalidad del bus



Control del bus I2C utilizando puertos de E/S

– Eventos a programar

- Generación de *START*
- Generación de *STOP*
- Enviar dirección o dato y recibir *ACK*
- Recibir dato y enviar *ACK* o *NACK*

– Consideraciones temporales

- Velocidad del procesador
 - El reloj del 8051 típico se puede considerar 12 MHz
 - Ciclo máquina en modo X1: 1 μs .
 - Duración media de una instrucción: 2 ciclos (2 μs)



Generación del protocolo desde un μC

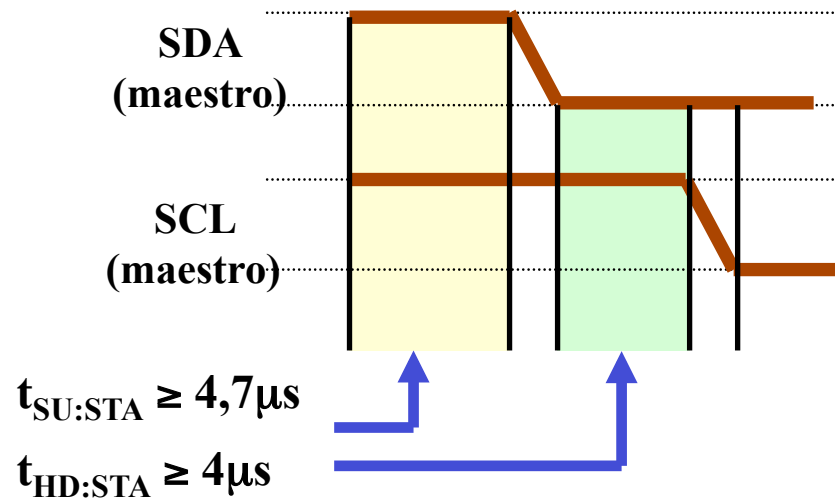


B
U
S

I
2
C

Control del bus I2C utilizando puertos de E/S

I2C_START



Pseudocódigo

```
SDA = 1  
SCL = 1  
delay(4,7 $\mu\text{s}$ )  
SDA = 0  
delay(4 $\mu\text{s}$ )  
SCL = 0
```



Generación del protocolo desde un μC

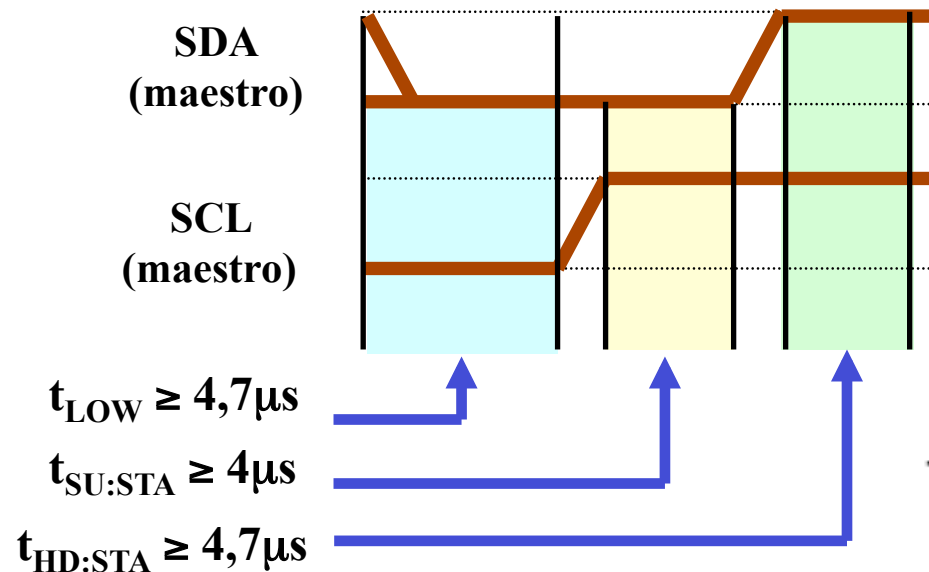


B
U
S

I
2
C

Control del bus I2C utilizando puertos de E/S

I2C_STOP



Pseudocódigo

```
SDA = 0  
SCL = 0  
delay(4,7 $\mu\text{s}$ )  
SCL = 1  
delay(4 $\mu\text{s}$ )  
SDA = 1  
delay(4,7 $\mu\text{s}$ )
```

```
void I2CSendStop(void)  
{  
    SDA = 0;  
    I2Cdelay();  
    SCL=1;  
    I2Cdelay();  
    SDA = 1;  
    I2Cdelay();  
}
```




Generación del protocolo desde un μC

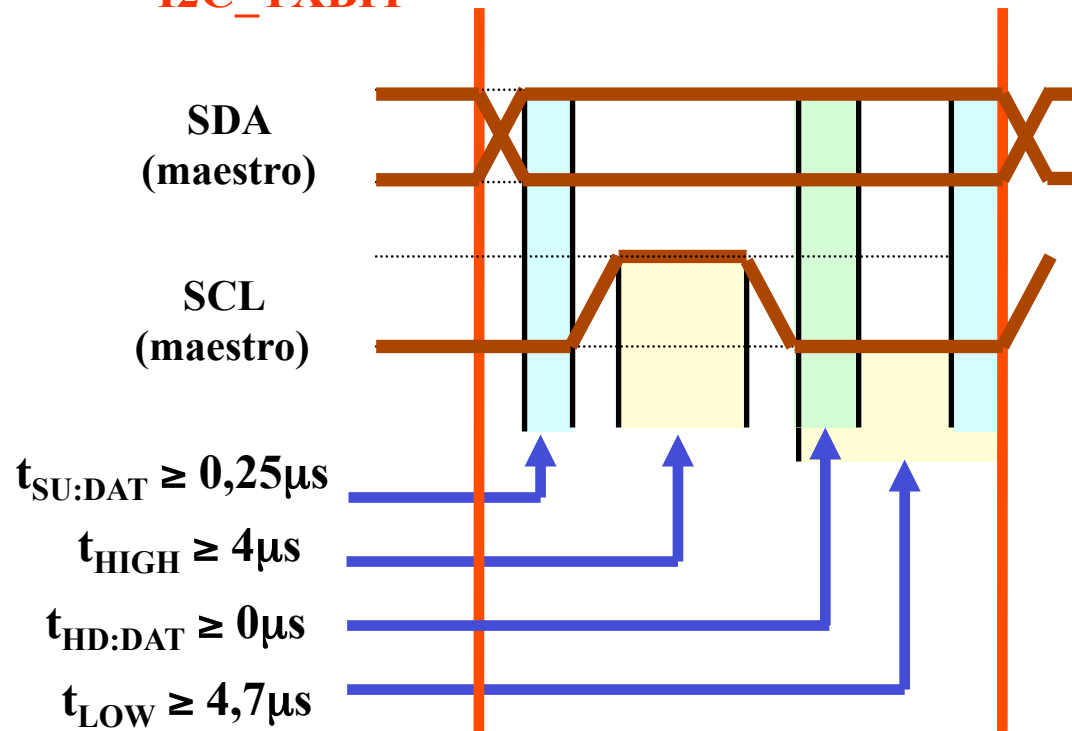


B
U
S

I
2
C

Control del bus I2C utilizando puertos de E/S

I2C_TXBIT



Pseudocódigo

```
SDA = BIT  
SCL = 1  
delay(4μs )  
SCL = 0  
delay(4,7μs )
```



Generación del protocolo desde un μC

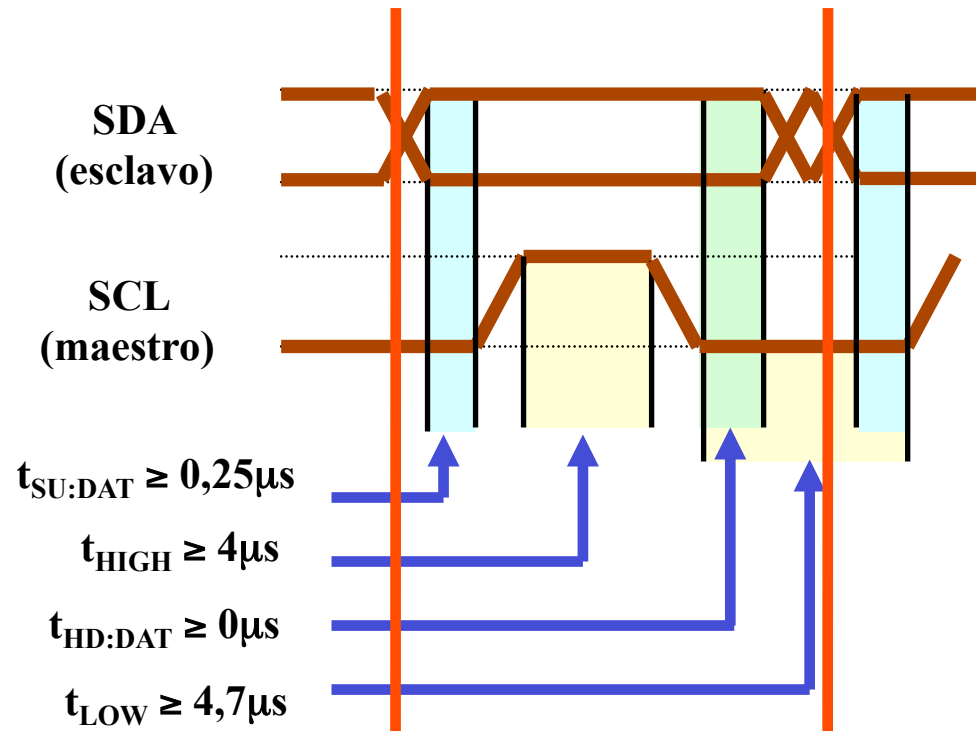


B
U
S

I
2
C

Control del bus I2C utilizando puertos de E/S

I2C_RXBIT



Pseudocódigo

```
SCL = 1  
delay(4 $\mu\text{s}$ )  
BIT = SDA  
SCL = 0  
delay(4,7 $\mu\text{s}$ )
```



Generación del protocolo desde un μC

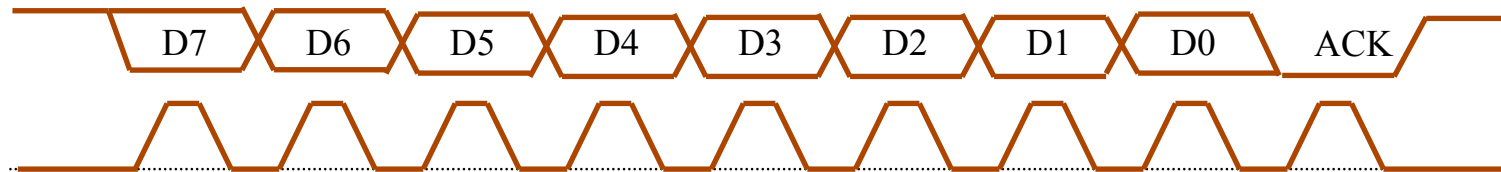


B
U
S

I
2
C

Control del bus I2C utilizando puertos de E/S

I2C_TXBYTE



Pseudocódigo

Repite 8

DESPLAZA DATO ←

I2C_TXBIT

I2C_RXACK

Devuelve ACK



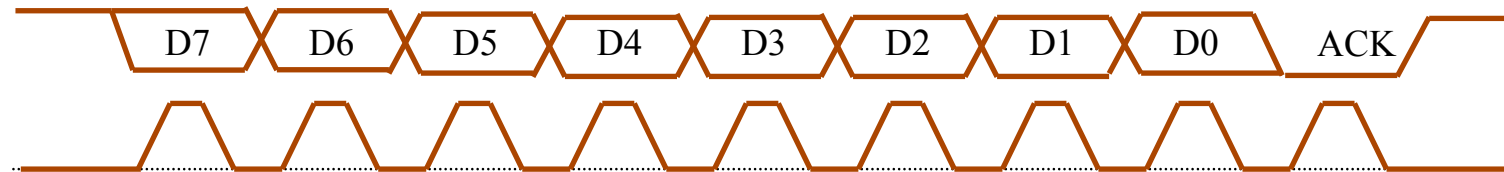
Generación del protocolo desde un μ C



B U S I 2 C

Control del bus I2C utilizando puertos de E/S

I2C_TXBYTE



```
void I2CSendByte(unsigned char byte)
{
    unsigned char i;
    for (i=0; i<8; i++)
    {
        if (byte & 0x80) SDA = 1; //envia cada bit, comenzando por el MSB
        else SDA = 0;
        SCL=1;
        I2Cdelay();
        SCL = 0;
        I2Cdelay();
        byte = byte << 1;
    }

    SDA = 1; // espera ACK (config. pin como lectura)
    SCL=1;
    I2Cdelay();
    SCL = 0;
    I2Cdelay();
}
```



Generación del protocolo desde un μ C

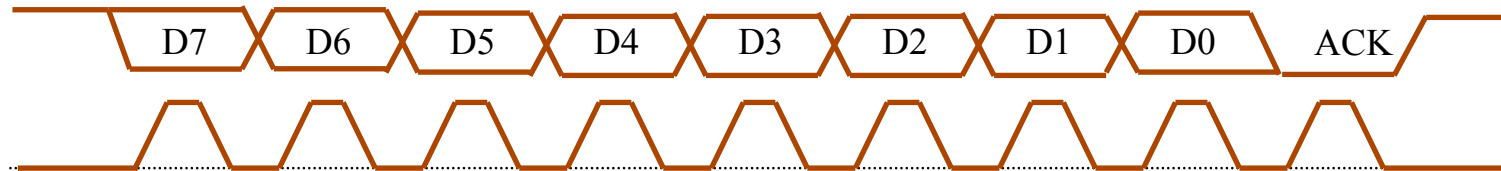


B
U
S

I
2
C

Control del bus I2C utilizando puertos de E/S

I2C_RXBYTE



Pseudocódigo

Repite 8

I2C_RXBIT

DESPLAZA DATO ←

I2C_TXACK

Devuelve DATO



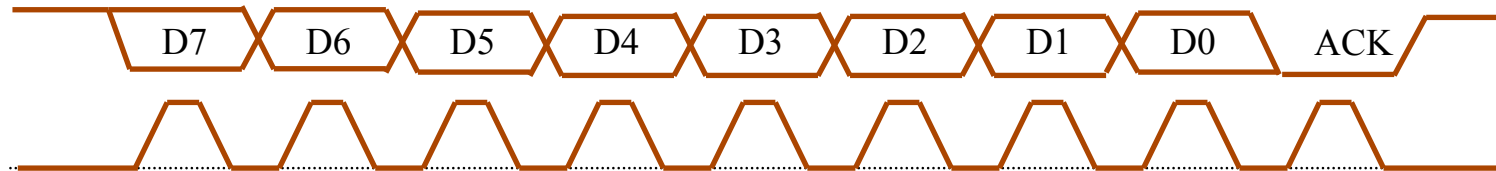
Generación del protocolo desde un μ C



B U S I 2 C

Control del bus I2C utilizando puertos de E/S

I2C_RXBYTE



```
unsigned char I2CGetByte(unsigned char ultimo)
// ultimo = NACK = 1 para el ultimo byte
// ultimo = ACK = 0 para cualquier byte que no sea el último.
{
    unsigned char i, byte;
    SDA=1;
    for (i=0;i<8;i++)          // lee un bit comenzando por el MSB
    {
        SCL=1;
        I2Cdelay();
        byte=byte<<1;
        if (SDA) byte++;
        SCL = 0;
        I2Cdelay();
    }
    SDA = ultimo;              // ACK es funcion de ultimo
    SCL=1;
    I2Cdelay();
    SCL = 0;
    I2Cdelay();
    return(byte);
}
```



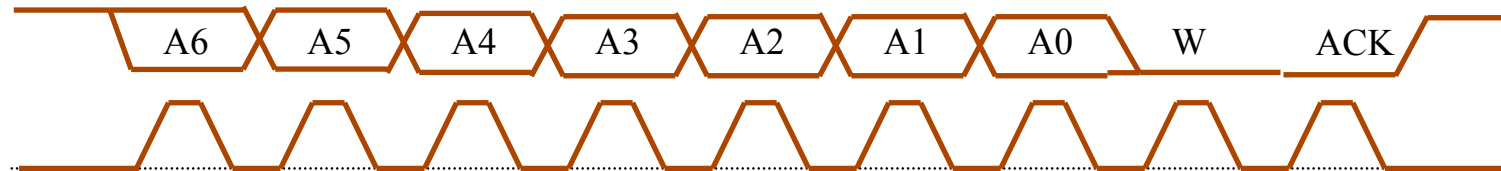
Generación del protocolo desde un μC



B
U
S

I
2
C

I2C_DIR_W



Pseudocódigo

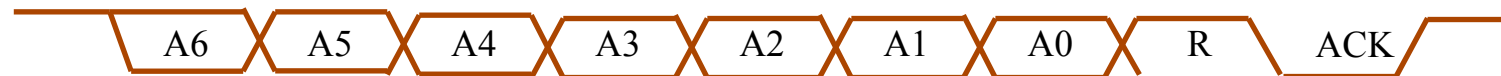
$A = \text{DIR} \ll 1 + W(0)$

I2C_START

I2C_TXBYTE

Devuelve ACK

I2C_DIR_R



Pseudocódigo

$A = \text{DIR} \ll 1 + R(1)$

I2C_START

I2C_TXBYTE

Devuelve ACK



Generación del protocolo desde un μC

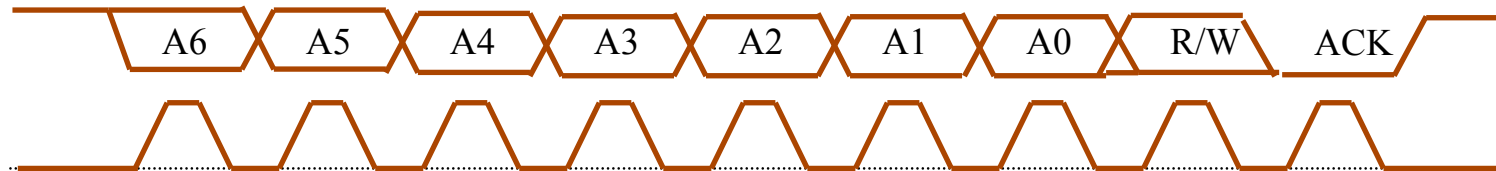


B
U
S

I
2
C

I2C_DIR_W

I2C_DIR_R



```
void I2CSendAddr(unsigned char addr, unsigned char rw)
{
    SDA = 1;
    SCL = 1;
    I2Cdelay();
    SDA = 0;                      // condicion de START
    I2Cdelay();
    SCL = 0;
    I2Cdelay();
    I2CSendByte((addr=addr<<1) + rw); // envia byte de direccion
                                        // addr, dirección (7 bits)
                                        // rw=1, lectura
                                        // rw=0, escritura
}
```




Índice de la Lección

- Introducción a la comunicación entre dispositivos mediante un bus serie
- Características del bus I²C
- Conexión de dispositivos al bus: nivel físico
- Intercambio de información: nivel de enlace
- Generación del protocolo desde un microcontrolador
- **Ejemplos de dispositivos que utiliza el bus I²C**
- Unidad I2C del LPC17xx
- Conclusiones
- Bibliografía



Factores a tener en cuenta

- Los dispositivos tendrán **dos partes**.
 - Bloque funcional del dispositivo
 - Interfaz I²C
- **Interfaz I²C**
 - Direccionamiento
 - Acceso a la funcionalidad del dispositivo
 - En dispositivos sencillos – lectura y escritura de datos.
 - En dispositivos complejos
 - » Configuración de modo de funcionamiento
 - » Comando + dato
- Ejemplos de dispositivo
 - puerto de E/S de 8 bits
 - Memoria serie
 - Sensor de temperatura - termostato

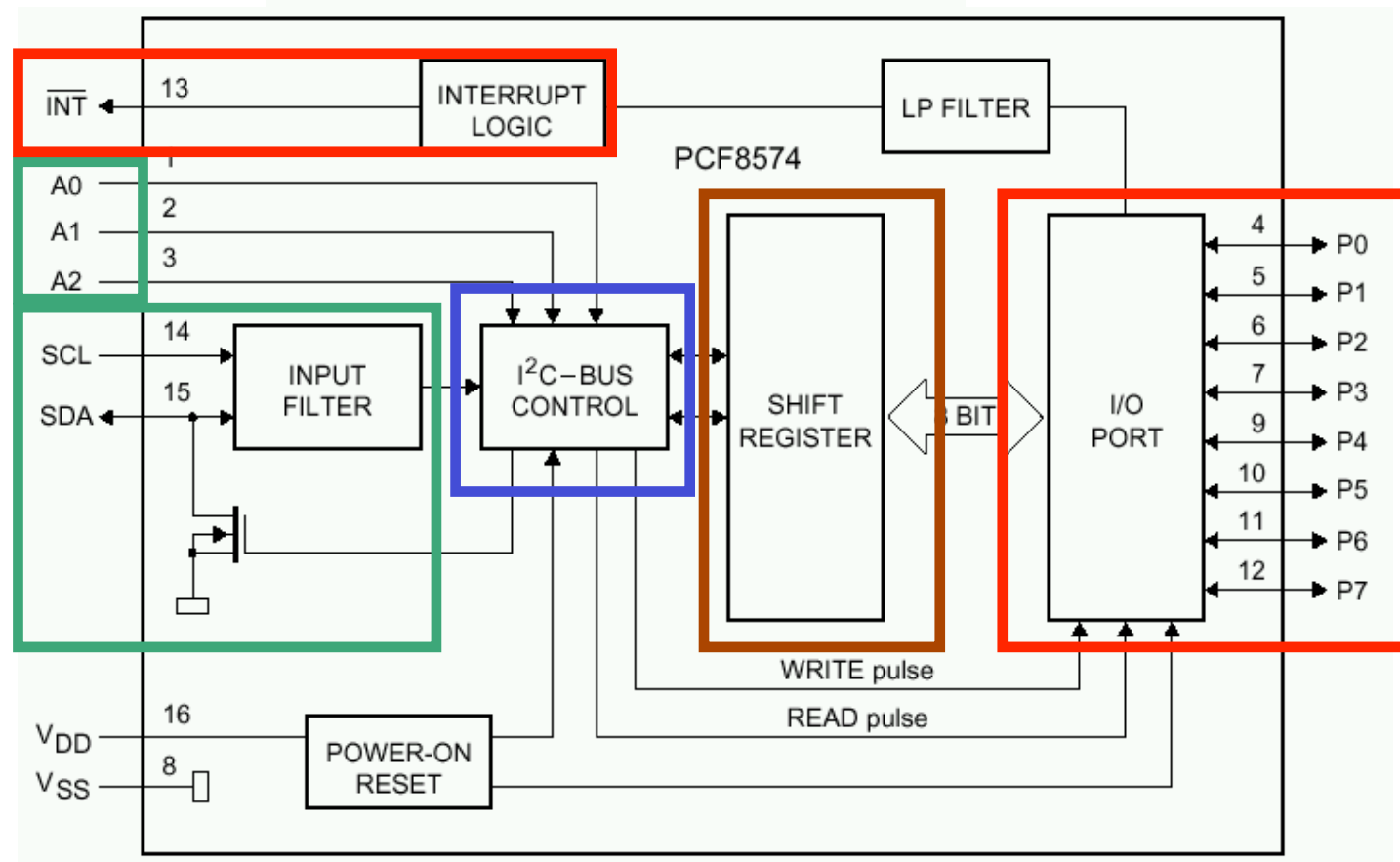


Ejemplos de dispositivos que utilizan I²C

B
U
S

I
2
C

Puerto de 8 bits de entrada/salida – PCF8574





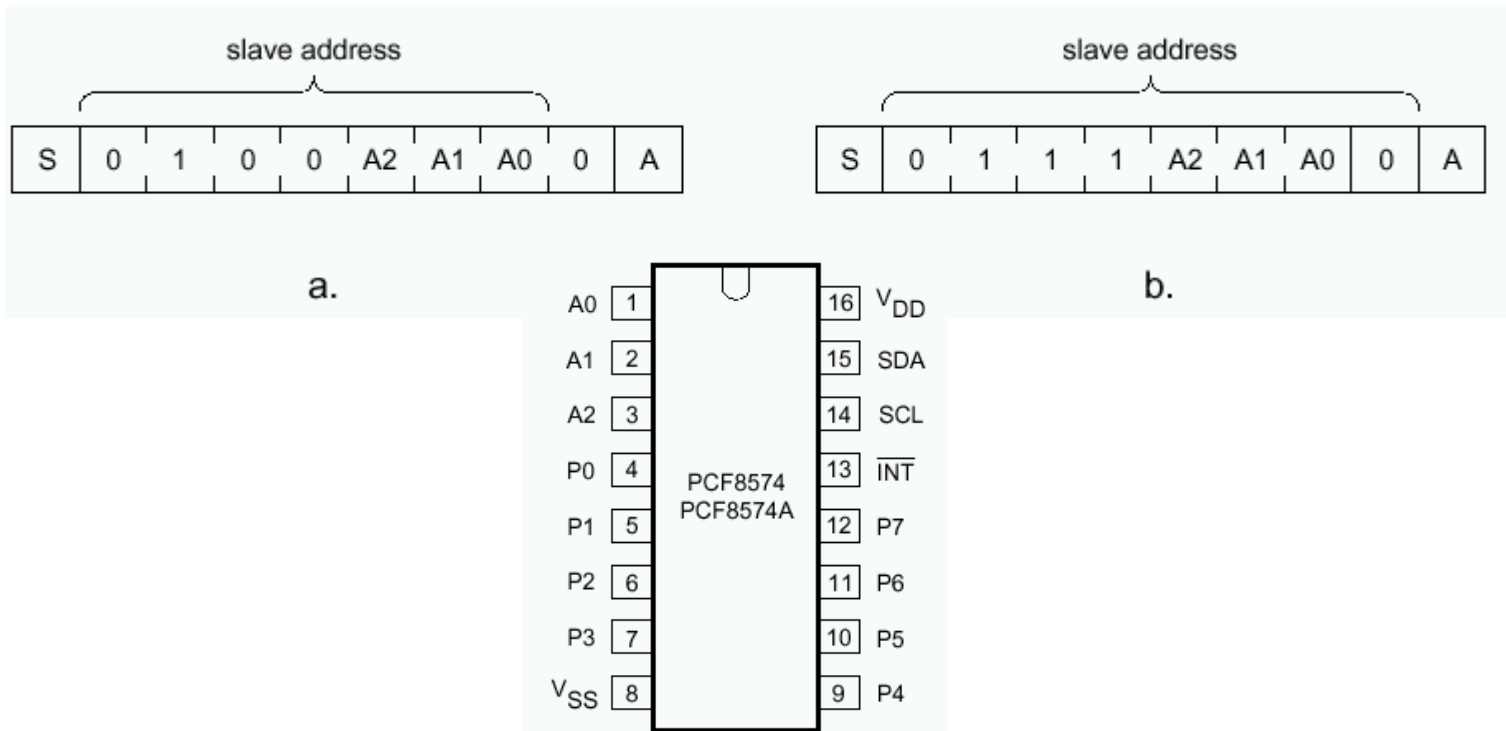
Ejemplos de dispositivos que utilizan I²C

B U S I 2 C

Puerto de 8 bits de entrada/salida – PCF8574

– Direcccionamiento

- Parte de la dirección es fija. Depende del dispositivo
- Los tres bits de menor peso se configuran con pines del dispositivo





Ejemplos de dispositivos que utilizan I²C

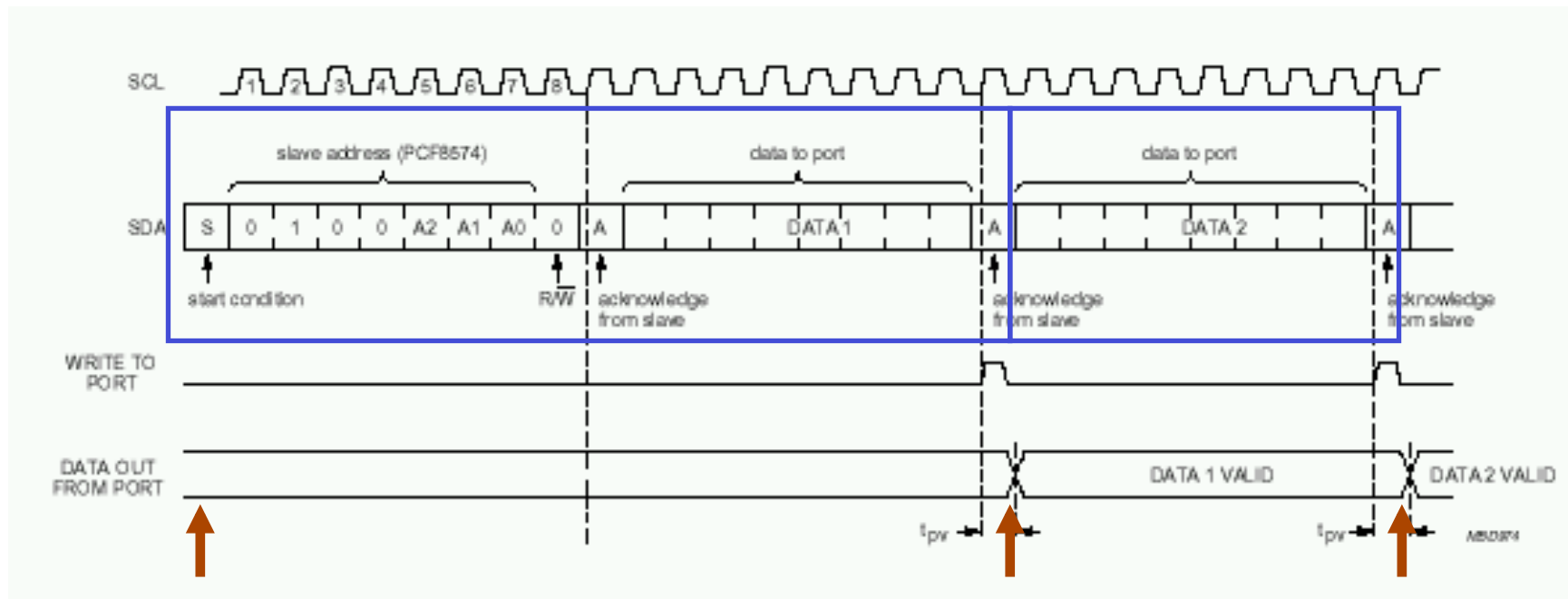


B
U
S

I
2
C

Puerto de 8 bits de entrada/salida – PCF8574

– Escritura en el puerto



Inicio de escritura

Dato disponible



Ejemplos de dispositivos que utilizan I²C

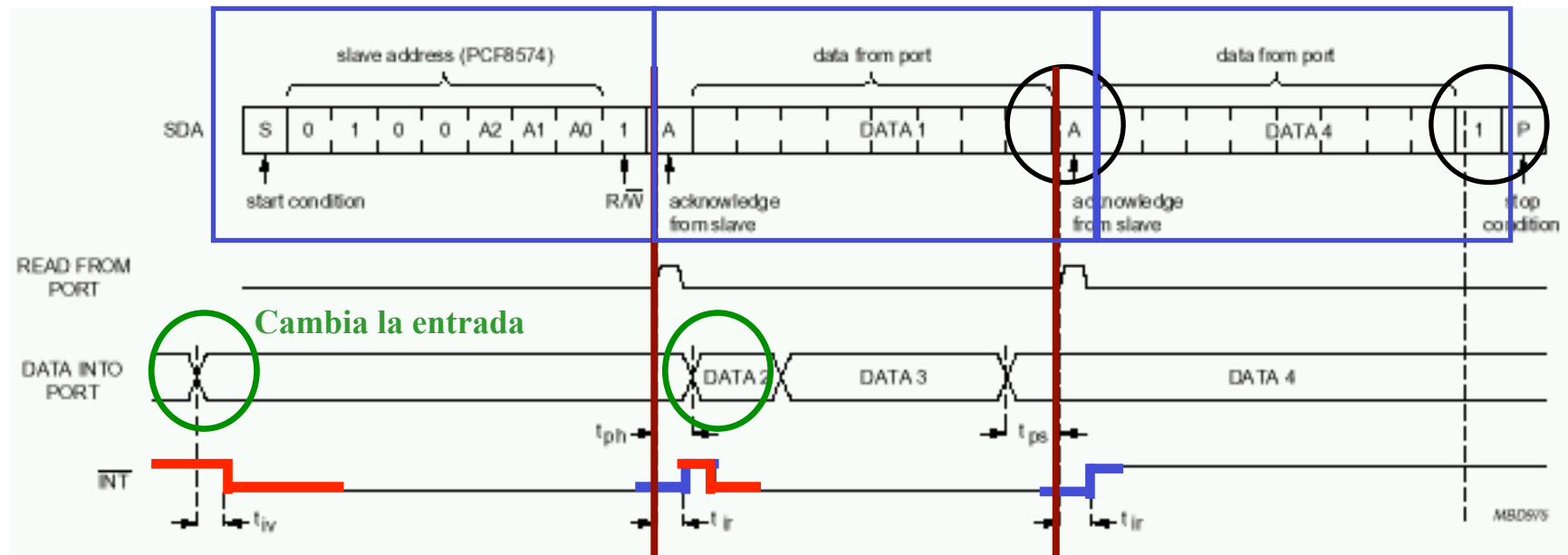


B U S I 2 C

Puerto de 8 bits de entrada/salida – PCF8574

– Lectura del puerto

- Si se modifica una entrada se activa una interrupción
- La línea de interrupción tiene salida en colector abierto.



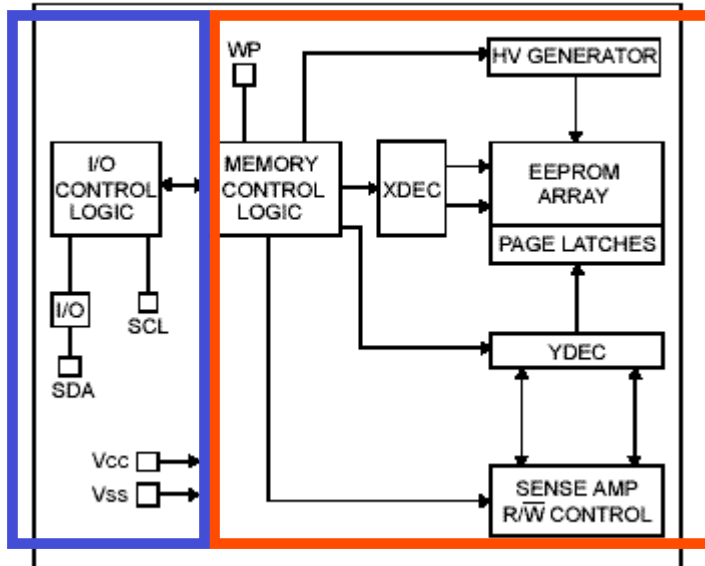


Memoria serie

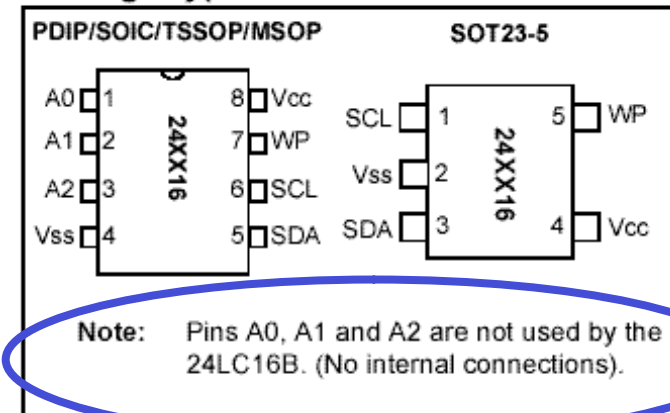
24AA16/24LC16B

16K I²C™ Serial EEPROM

Block Diagram



Package Types





Ejemplos de dispositivos que utilizan I²C



B U S I 2 C

Memoria serie - Comando

Memoria de 16K bits (2KBytes)

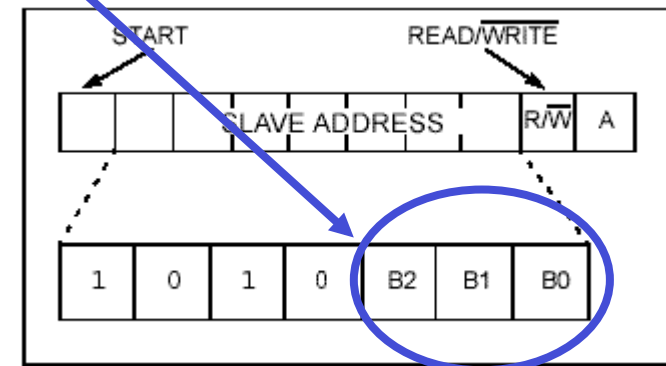
Organizada en 8 bloques de 256 bytes

Dirección de bloque (3 bits)

Dirección de byte (8 bits)

Lectura / escritura

Operation	Control Code	Block Select	R/ \overline{W}
Read	1010	Block Address	1
Write	1010	Block Address	0





Ejemplos de dispositivos que utilizan I²C



BUS I²C

Memoria serie - Escritura

FIGURE 4-1: BYTE WRITE

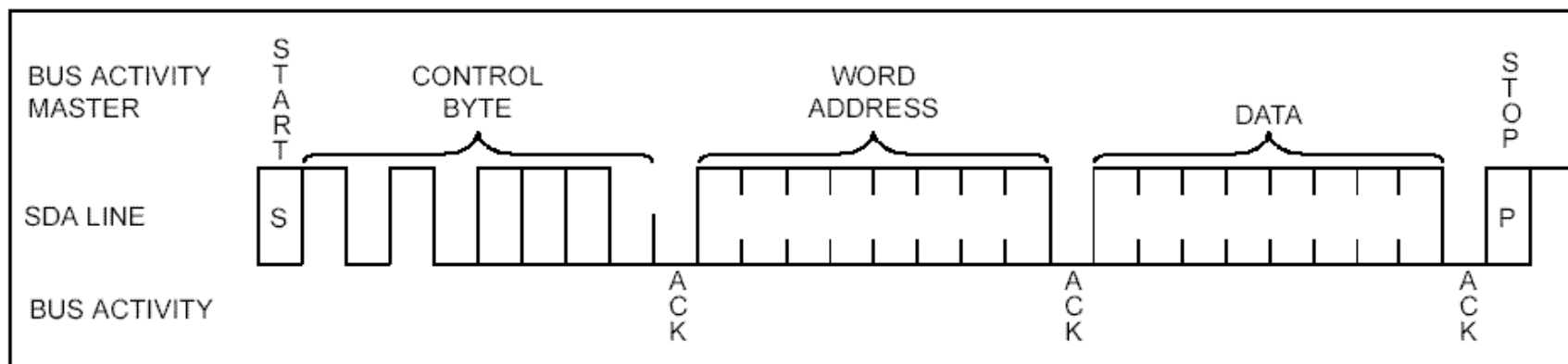
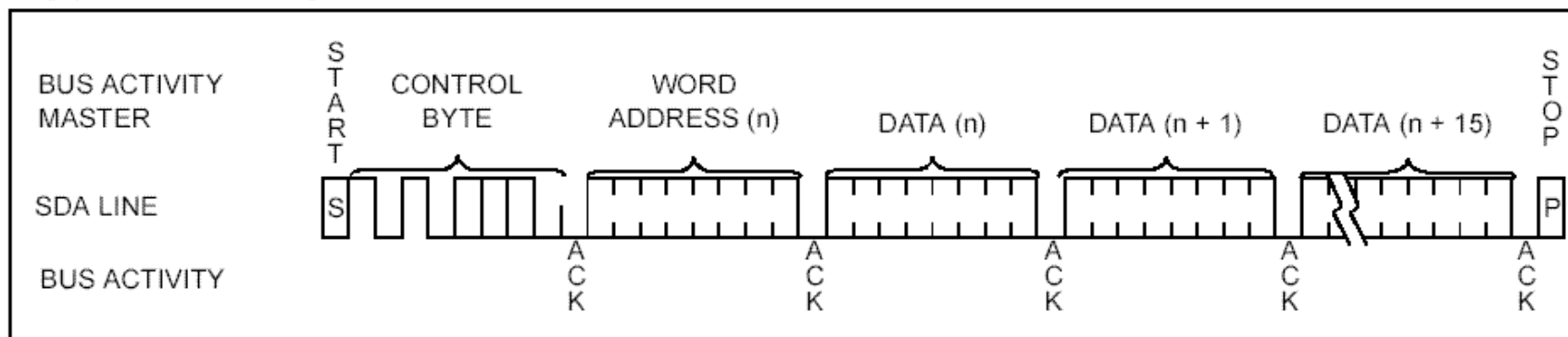


FIGURE 4-2: PAGE WRITE





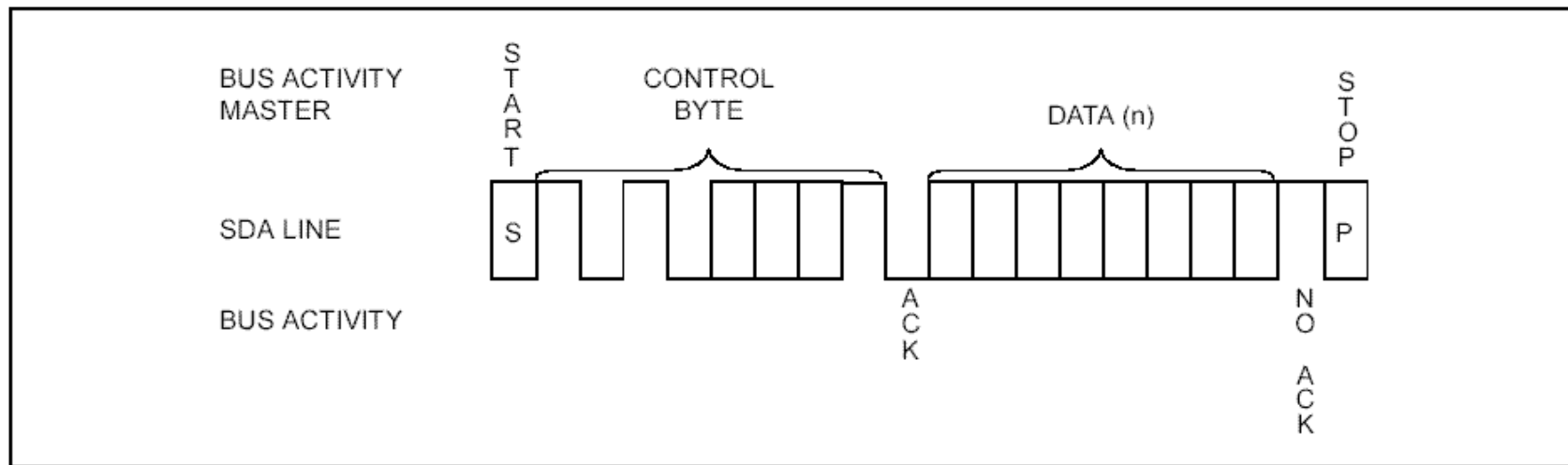
Ejemplos de dispositivos que utilizan I²C



B U S I 2 C

Memoria serie - lectura

FIGURE 7-1: CURRENT ADDRESS READ



Con cada lectura se autoincrementa un puntero interno



Ejemplos de dispositivos que utilizan I²C



BUS I²C

Memoria serie - lectura

FIGURE 7-2: RANDOM READ

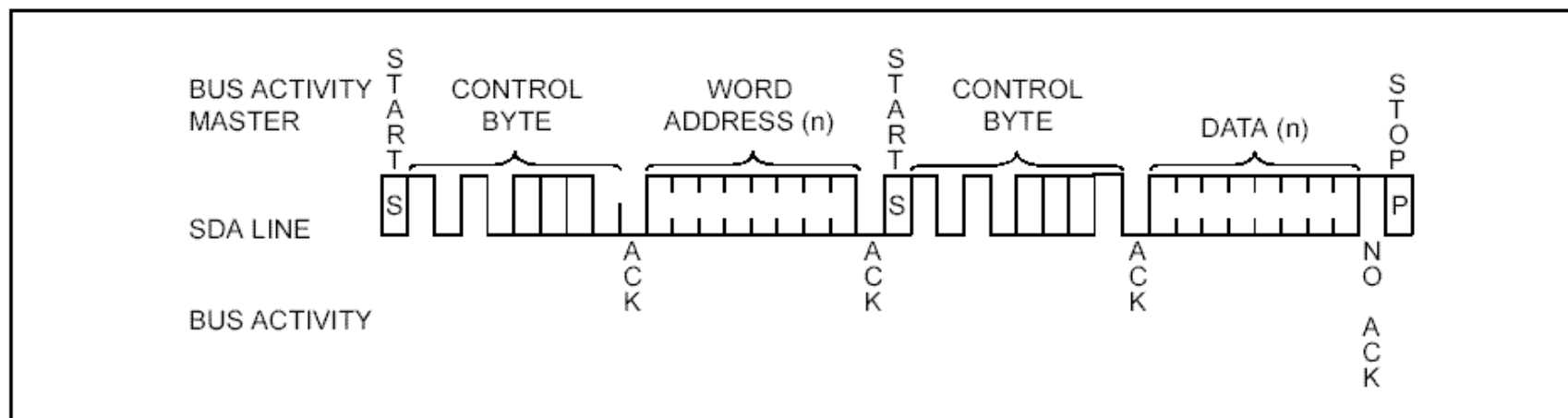
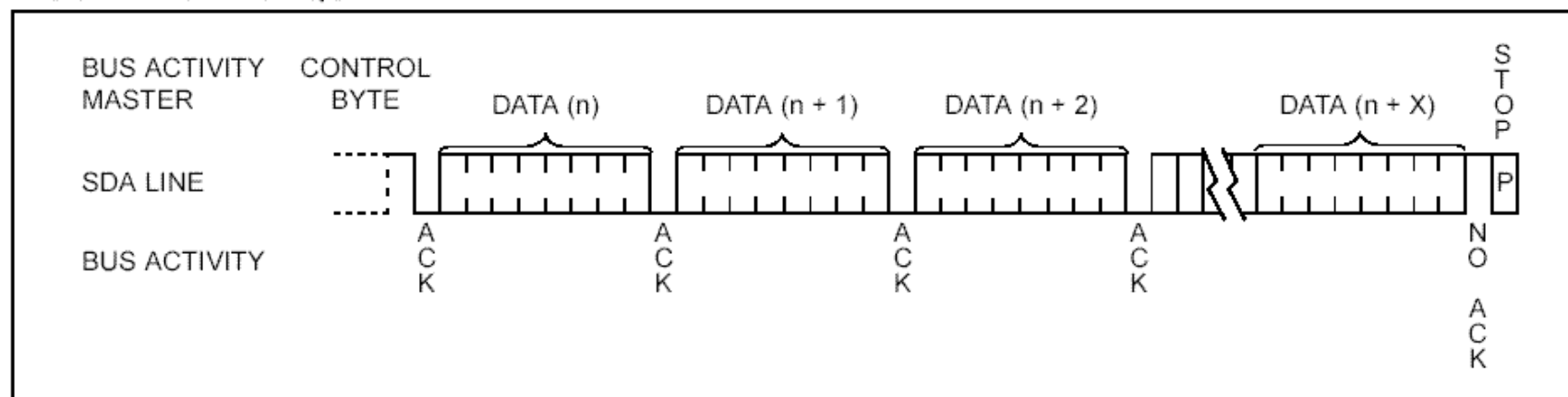


FIGURE 7-3: SEQUENTIAL READ





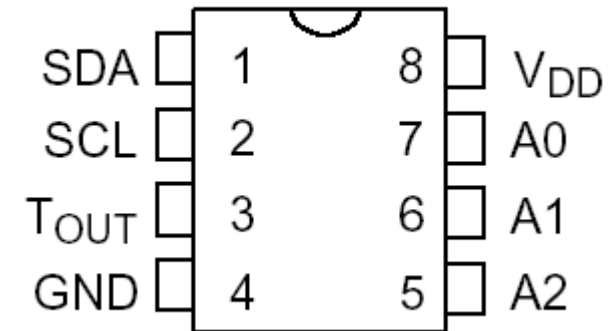
Ejemplos de dispositivos que utilizan I²C



B U S I 2 C

Termómetro / Termostato DS1621

- **Margen de temperatura: -55 a +125°C**
- **Resolución: 0.5°C**
- **T^a almacenada en 9 bits (2 bytes)**
- **Rango de alimentación: 2.7-5.5V**
- **Tiempo de conversión: 1 seg.**
- **Parámetros del termostato (TH y TL) almacenados en memoria no volátil (eeprom)**
- **No requiere circuitería externa.**





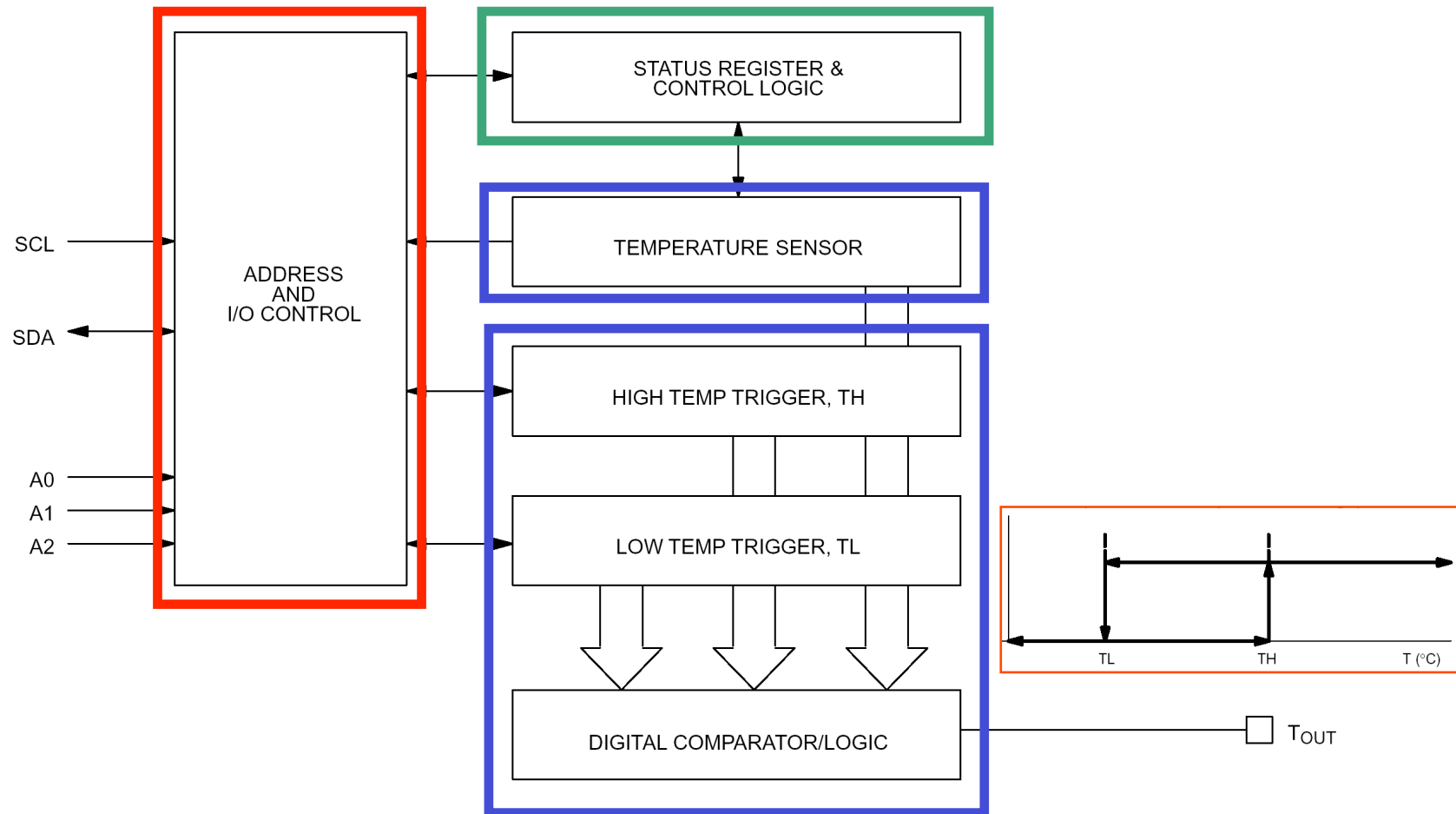
Ejemplos de dispositivos que utilizan I²C



B
U
S

I
2
C

Termómetro / Termostato





Termómetro / Termostato

MSB

LSB

1	1	1	0	0	1	1	1	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

$$T = -25^{\circ}\text{C}$$

TEMPERATURE	DIGITAL OUTPUT (Binary)	DIGITAL OUTPUT (Hex)
+125°C	01111101 00000000	7B00h
+25°C	00011001 00000000	1900h
+ $\frac{1}{2}$ °C	00000000 10000000	0080h
+0°C	00000000 00000000	0000h
- $\frac{1}{2}$ °C	11111111 10000000	FF80h
-25°C	11100111 00000000	E700h
-55°C	11001001 00000000	C900h



Termómetro / Termostato

Registro de Configuración

DONE	THF	TLF	NVB	1	0	POL	1SHOT
------	-----	-----	-----	---	---	-----	-------

- **DONE.** Flag de fin de conversión.
- **THF.** Flag ($T^a > TH$).
- **TLF.** Flag ($T^a < TL$).
- **NVB.** Flag indicador de que la grabación en eeprom está en proceso.
- **POL.** Polaridad de la salida Tout.
- **1SHOT.** Selección del modo de conversión única o continua. En el modo de conversión continua, el termostato puede funcionar de forma autónoma, sin necesidad de uC, una vez configurados TH y TL.

Comandos (1 byte):

- Lectura de la temperatura [AAh]
- Acceso a TH [A1h]
- Acceso a TL [A2h]
- Acceso al Reg. de Config. [Ach]
- Inicio de conversión [EEh]
- Parar la conversión [22h]



Ejemplos de dispositivos que utilizan I²C

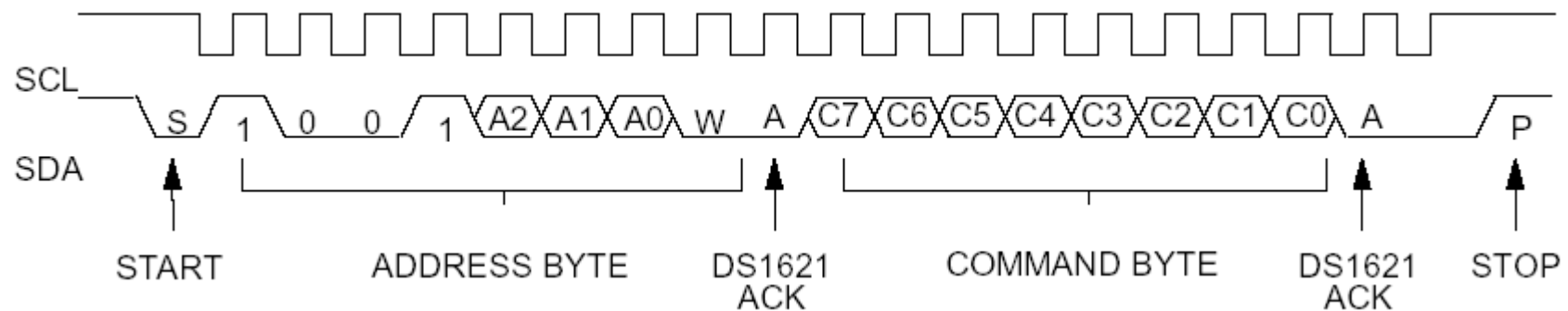


B
U
S

I
2
C

DS1621. Protocolo de acceso

Comando Inicio/Stop de conversión





Ejemplos de dispositivos que utilizan I²C

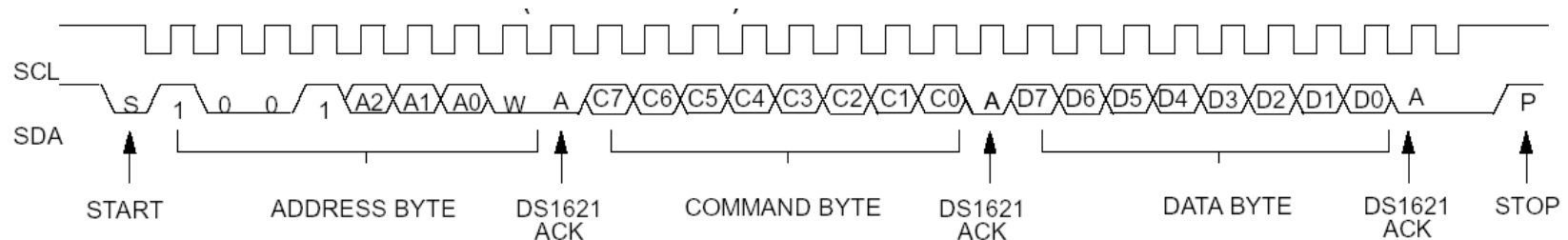


B
U
S

I
2
C

DS1621. Protocolo de acceso

**Escritura de un dato en un registro (1byte)
(Reg. De Config.)**





Ejemplos de dispositivos que utilizan I²C

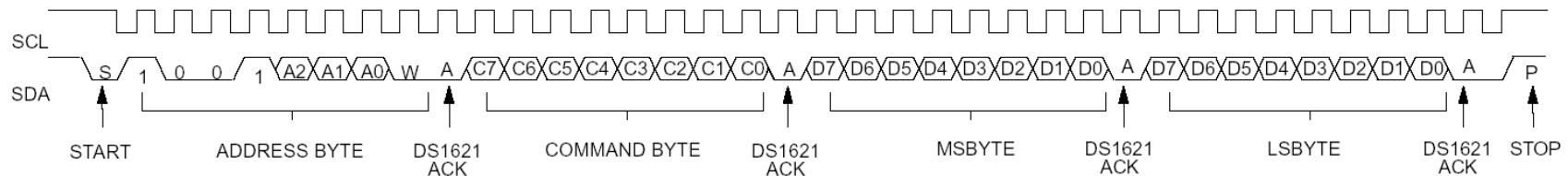


B
U
S

I
2
C

DS1621. Protocolo de acceso

Escritura de un dato en un registro (2bytes) (TH, TL)





Ejemplos de dispositivos que utilizan I²C

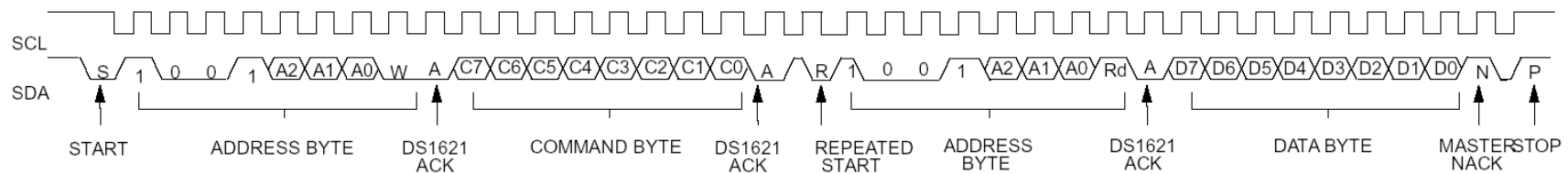


B
U
S

I
2
C

DS1621. Protocolo de acceso

Lectura de un dato (1byte) de un registro (Reg. De Config.)





Ejemplos de dispositivos que utilizan I²C

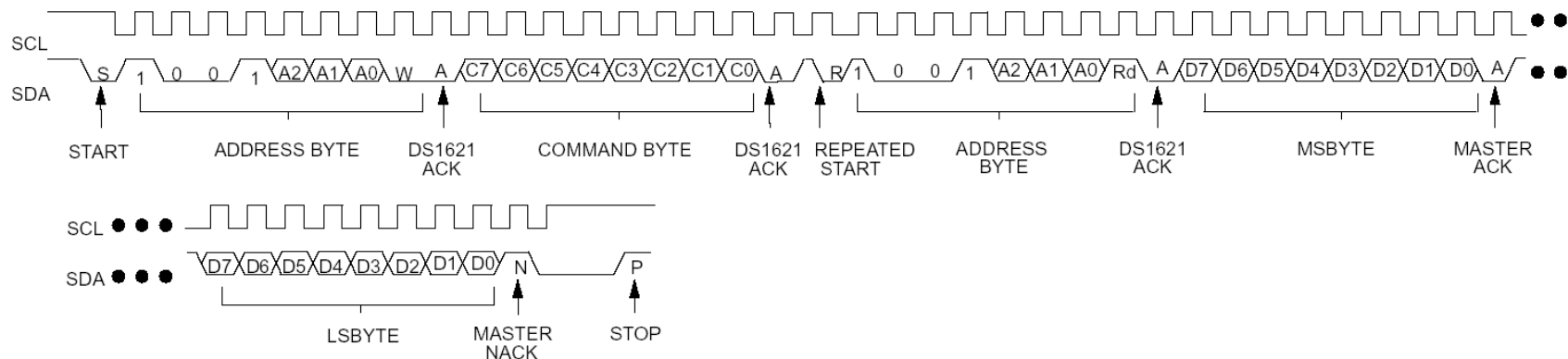


B
U
S

I
2
C

DS1621. Protocolo de acceso

Lectura de un dato (2 bytes) de un registro (T^a, T_H, T_L)





B
U
S

I
2
C

DS1621. Protocolo de acceso

Configuración del dispositivo (modo de conversión continua).

Configurar valores de TH y TL (si opción termostato).

Comando de Inicio de conversión (1ª vez)

Llamada a rutina de lectura de la Tª:



PCF8591. Conversor A/D y D/A

1 FEATURES

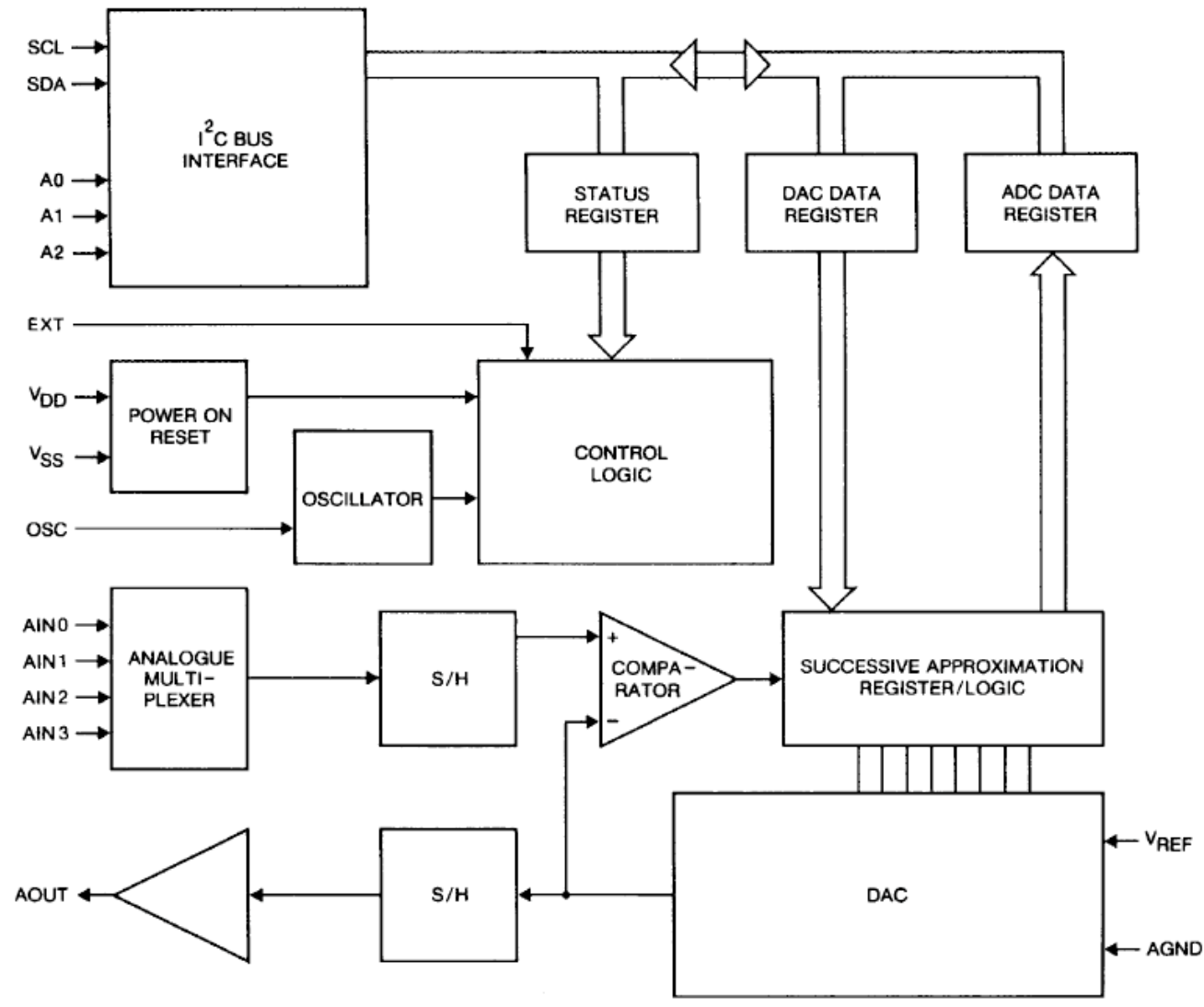
- Single power supply
- Operating supply voltage 2.5 V to 6 V
- Low standby current
- Serial input/output via I²C-bus
- Address by 3 hardware address pins
- Sampling rate given by I²C-bus speed
- 4 analog inputs programmable as single-ended or differential inputs
- Auto-incremented channel selection
- Analog voltage range from V_{SS} to V_{DD}
- On-chip track and hold circuit
- 8-bit successive approximation A/D conversion
- Multiplying DAC with one analog output.



Ejemplos de dispositivos que utilizan I²C

B
U
S

I
2
C





Ejemplos de dispositivos que utilizan I²C



B U S I 2 C

PCF8591. Conversor A/D y D/A

SYMBOL	PIN	DESCRIPTION
AIN0	1	analog inputs (A/D converter)
AIN1	2	
AIN2	3	
AIN3	4	
A0	5	hardware address
A1	6	
A2	7	
V _{SS}	8	negative supply voltage
SDA	9	I ² C-bus data input/output
SCL	10	I ² C-bus clock input
OSC	11	oscillator input/output
EXT	12	external/internal switch for oscillator input
AGND	13	analog ground
V _{REF}	14	voltage reference input
AOUT	15	analog output (D/A converter)
V _{DD}	16	positive supply voltage

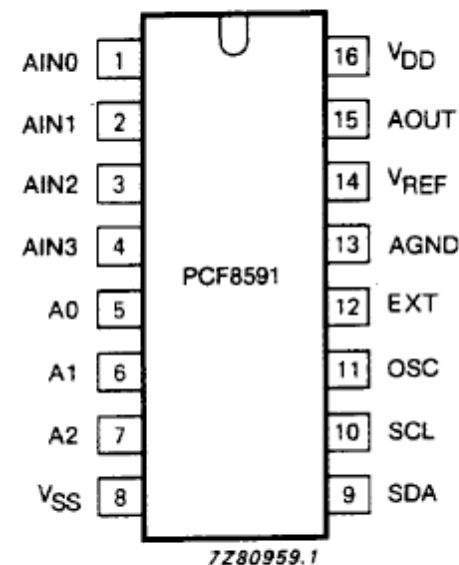


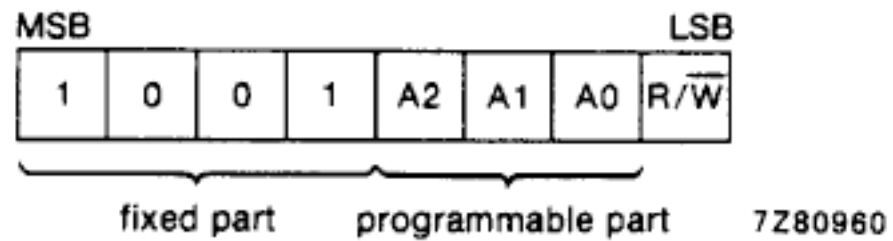
Fig.2 Pinning diagram.



Ejemplos de dispositivos que utilizan I²C

B U S I 2 C

PCF8591. Conversor A/D y D/A



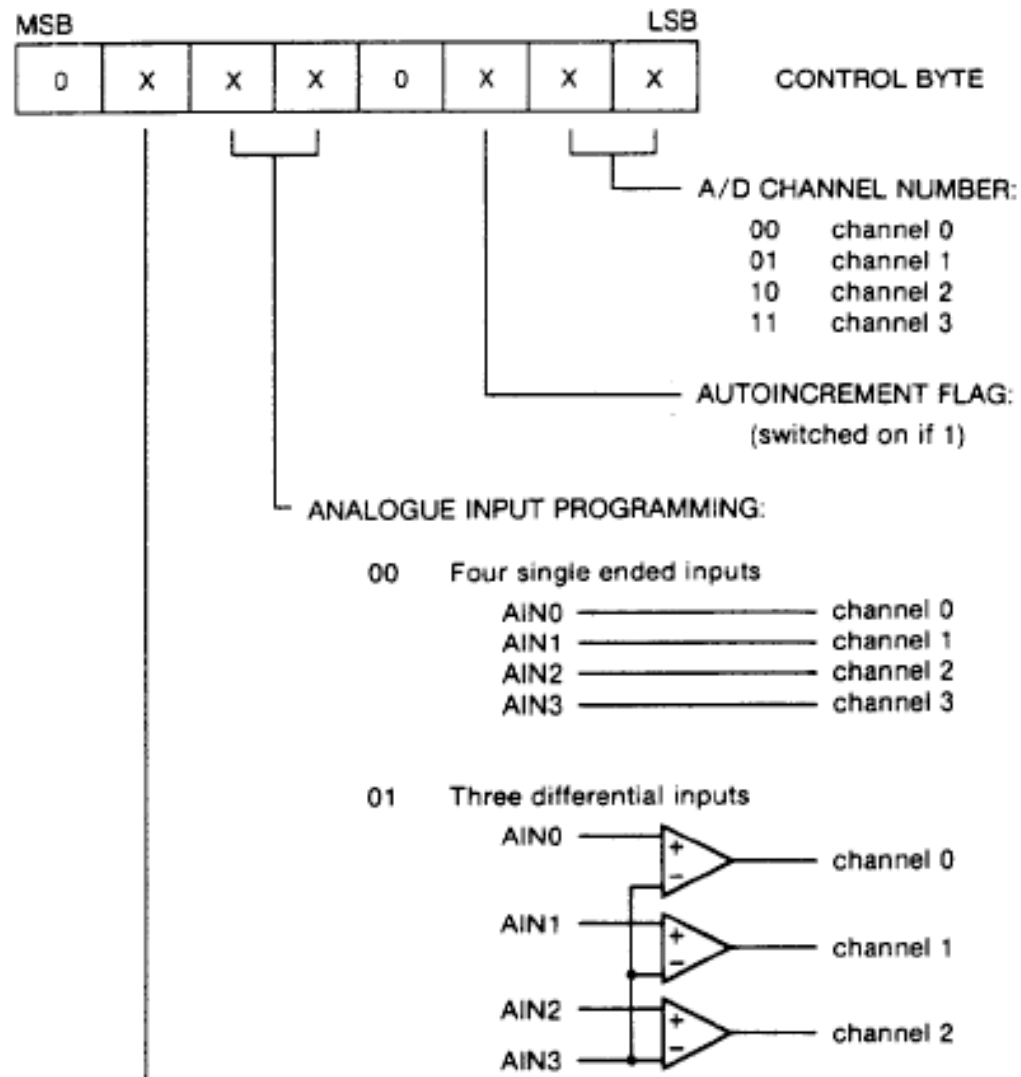
7280960

Fig.3 Address byte.



Ejemplos de dispositivos que utilizan I²C

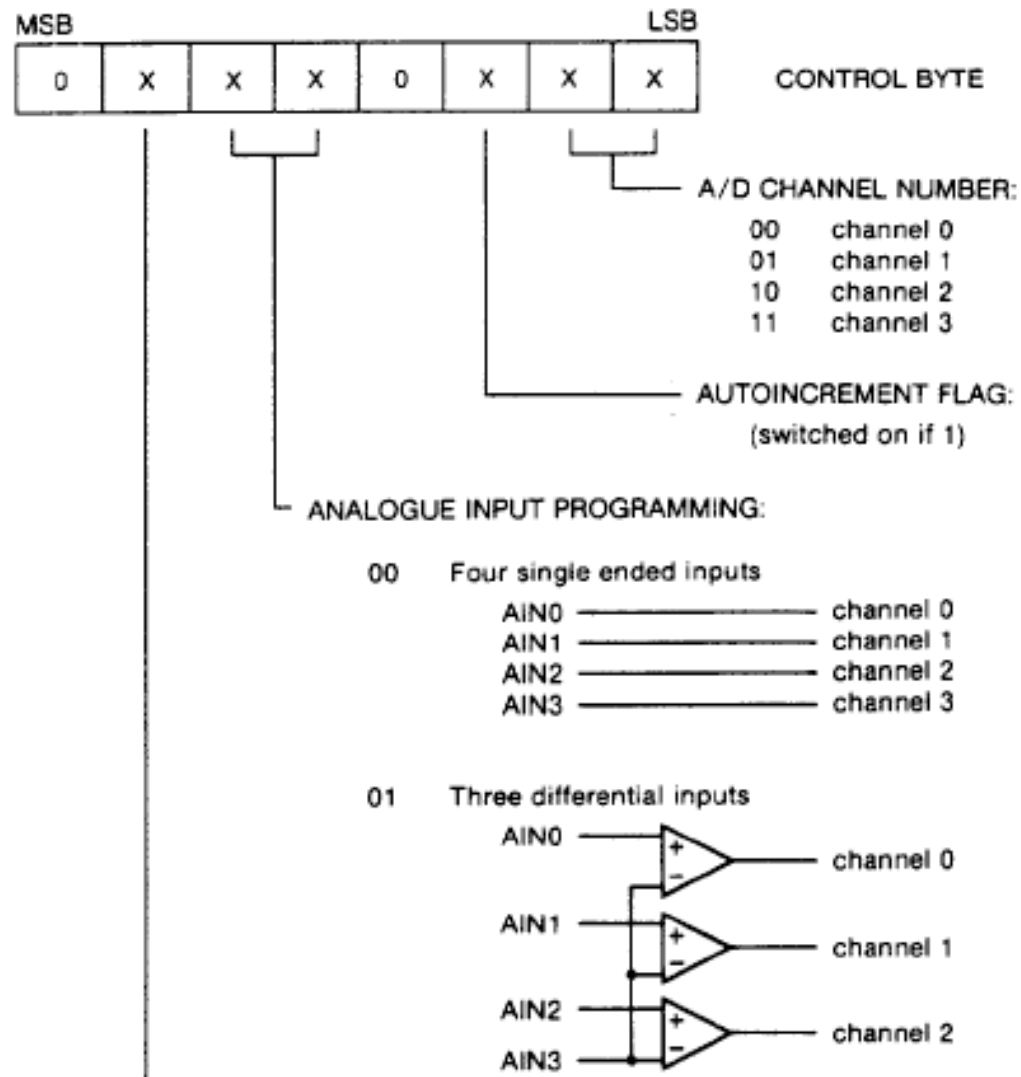
BUS I²C





Ejemplos de dispositivos que utilizan I²C

BUS I²C





Ejemplos de dispositivos que utilizan I²C



B U S I 2 C

PCF8591. Conversor A/D y D/A

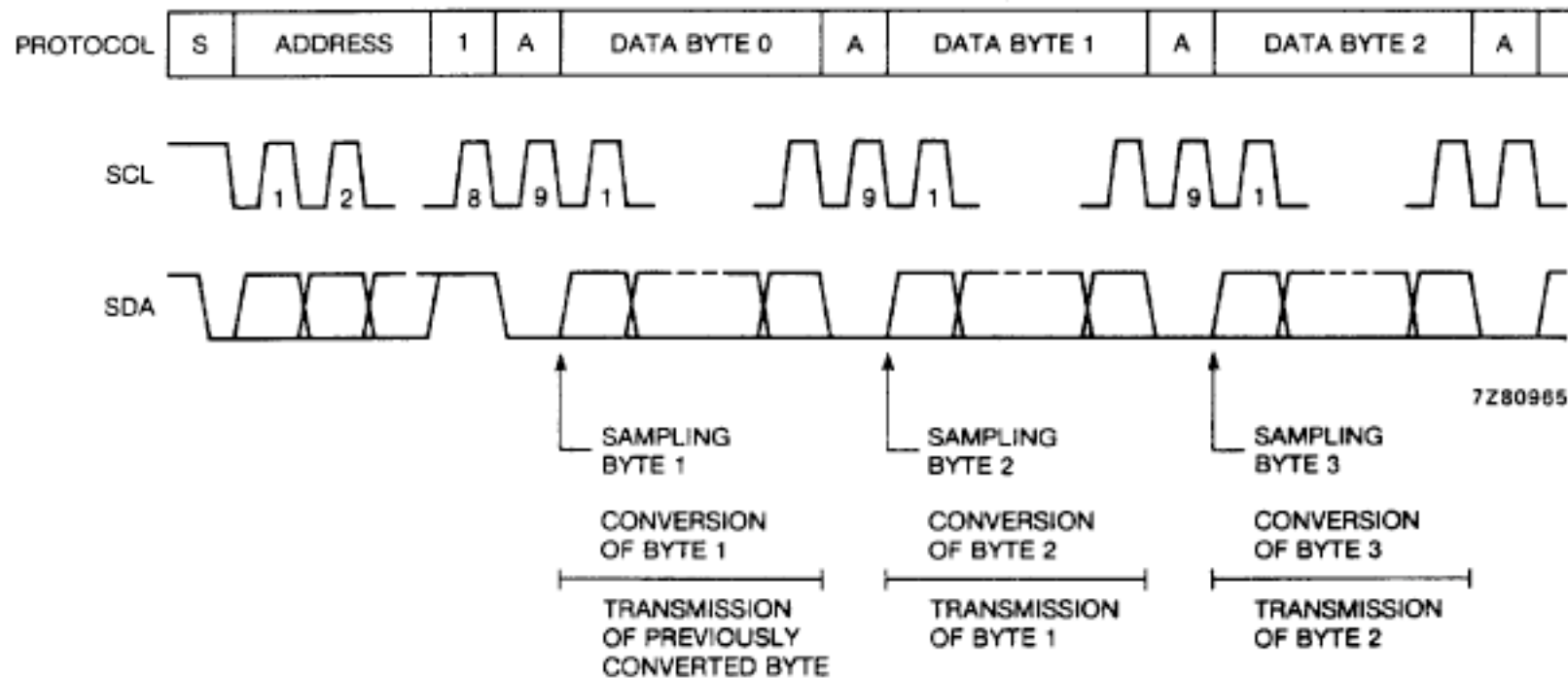


Fig.8 A/D conversion sequence.



Protocolo de Comunicación I²C



B
U
S

I
2
C

Índice de la Lección

- Introducción a la comunicación entre dispositivos mediante un bus serie
- Características del bus I²C
- Conexión de dispositivos al bus: nivel físico
- Intercambio de información: nivel de enlace
- Generación del protocolo desde un microcontrolador
- Ejemplos de dispositivos que utiliza el bus I²C
- **Unidad I2C del LPC17xx**
- Conclusiones
- Bibliografía



Unidad hardware específica de I2C

- El LPC 17xx dispone de 3 módulos de I2C (I2C0, I2C1, I2C2)
- Configuración como colector abierto
 - I2C0 se configura automáticamente
 - I2C1 e I2C2 necesitan configuración manual.
- Velocidad de comunicación
 - I2C0 permite alcanzar 1Mbps
 - I2C1 e I2C 2 pueden trabajar con 100K o 400K
- Puede trabajar como Maestro o como Esclavo
- Identifica colisiones en protocolo multimaestro



Modos de funcionamiento

- Master Trasnmmitter Mode
- Master Receiver Mode
- Slave Transmitter Mode
- Slave Receiver Mode

La comunicación debe gestionarse por una función de atención a interrupción que normalmente implementa una máquina de estados

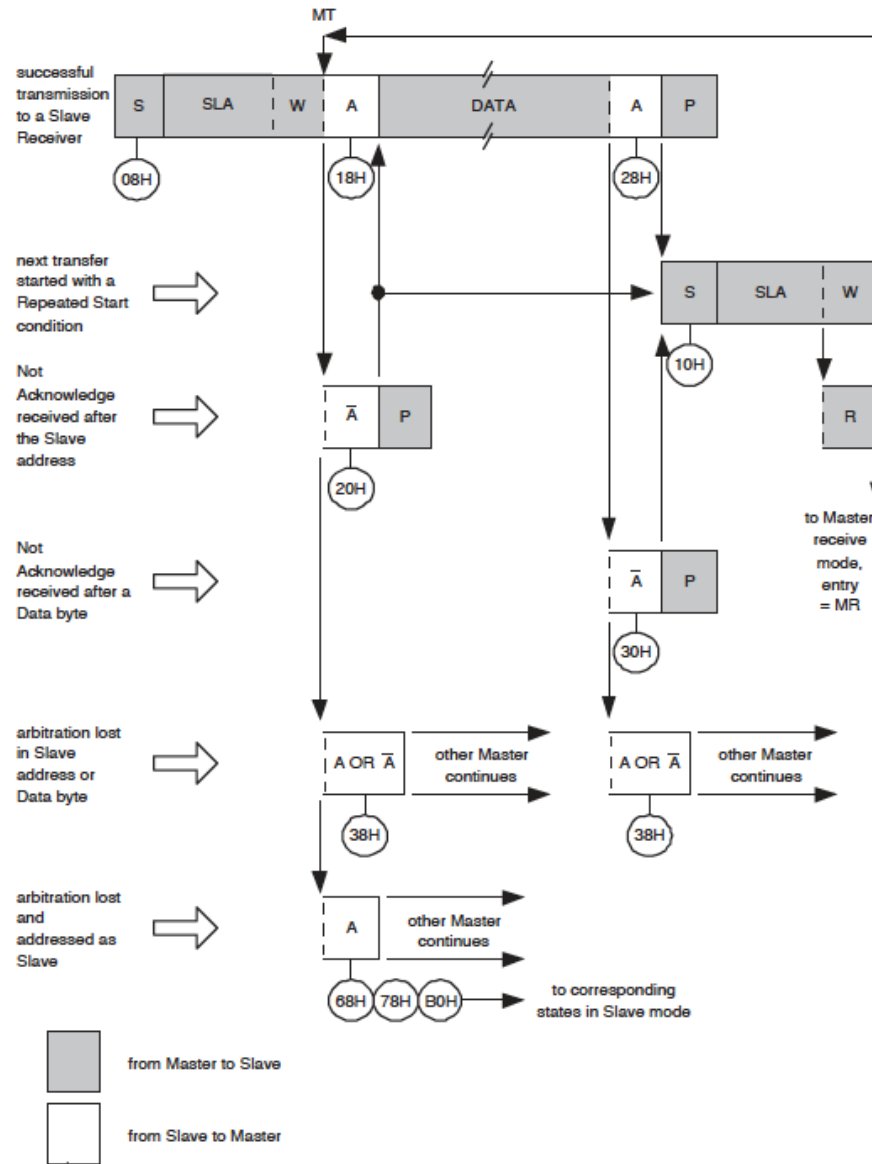


I2C en el LPC17xx



BUS I2C

Master Transmitter Mode



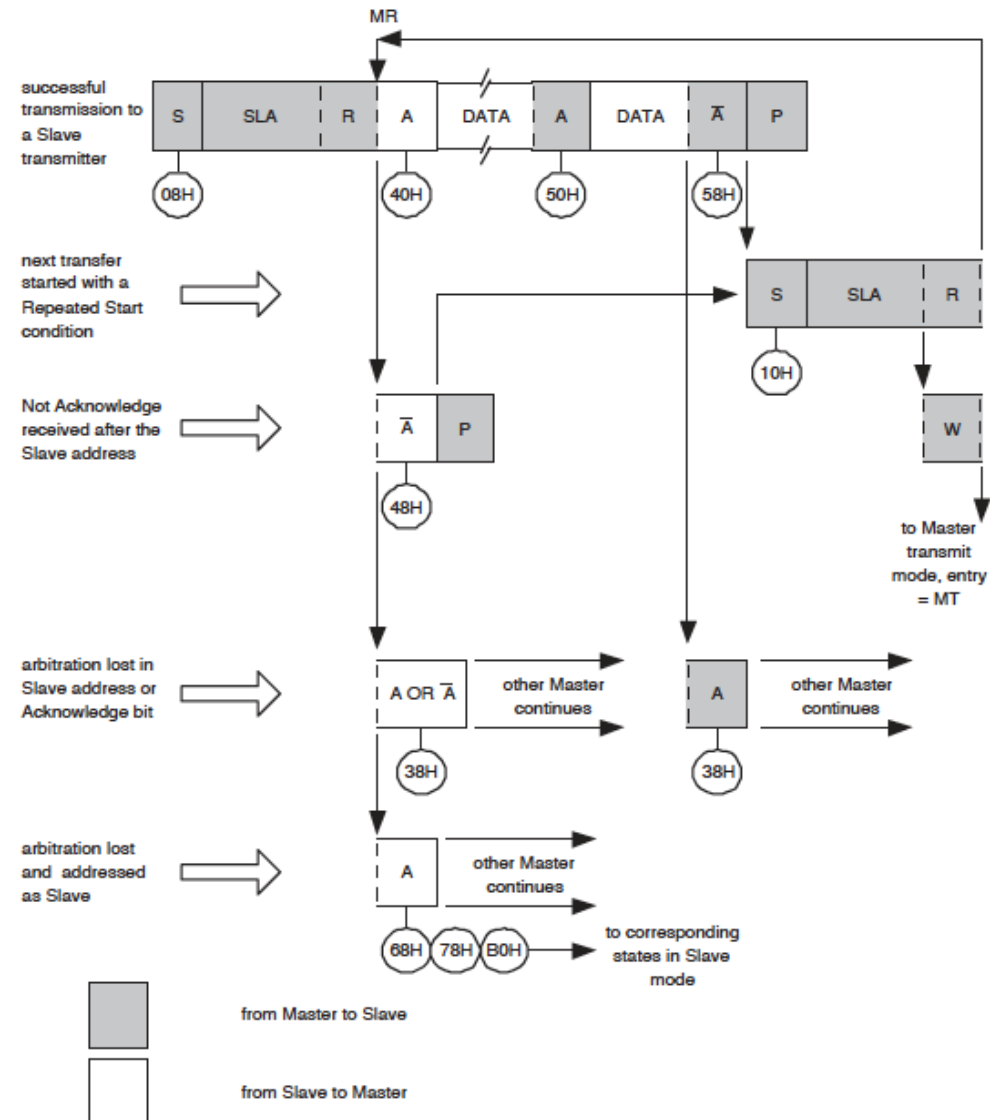


I2C en el LPC17xx



B U S I 2 C

Master Receiver Mode



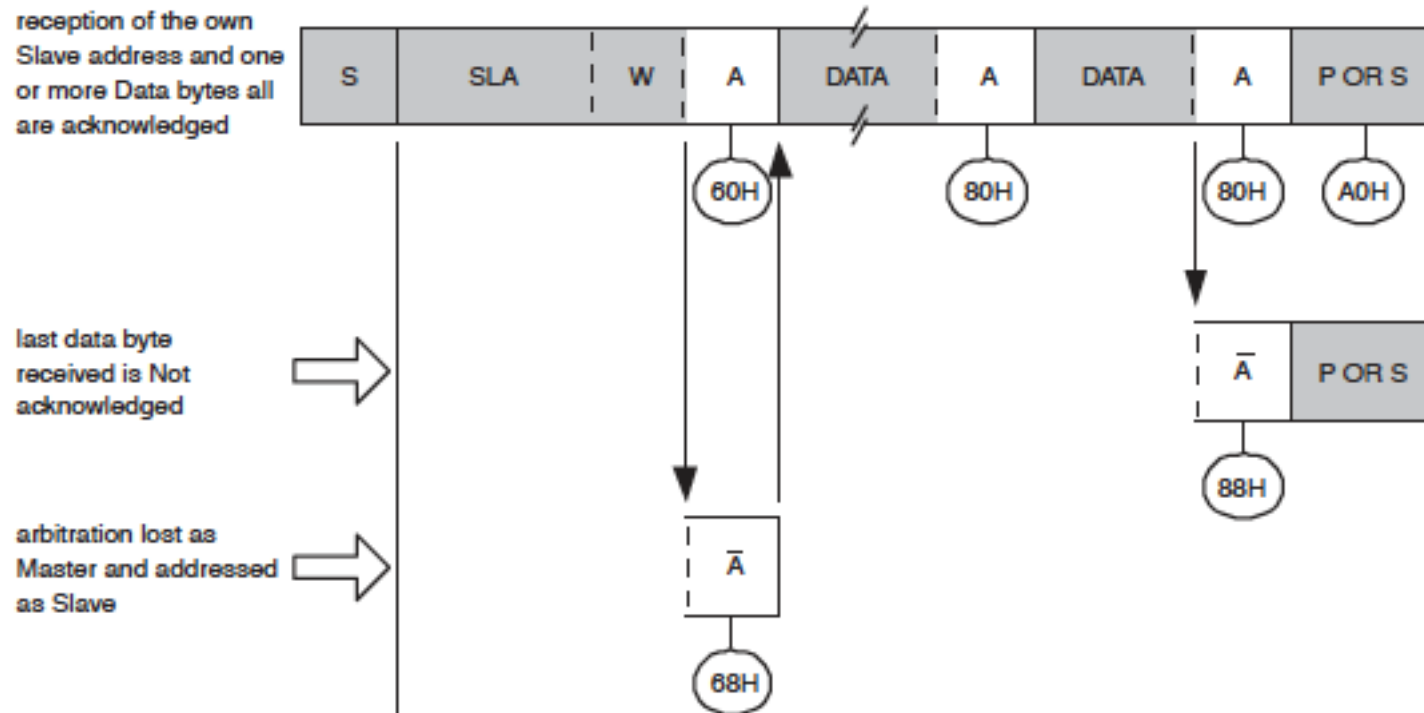


I2C en el LPC17xx



B U S I 2 C

Slaver Transmitter Mode





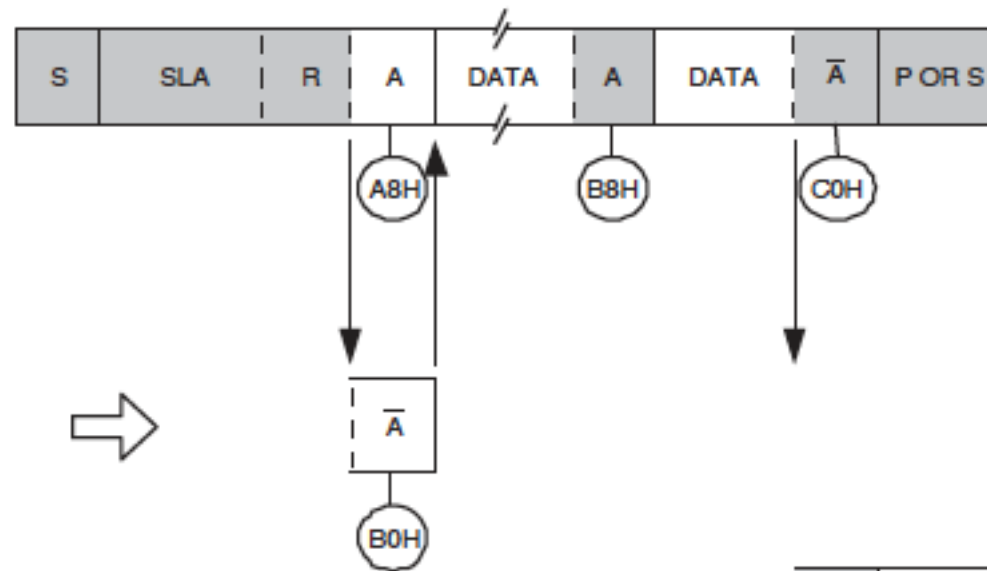
I2C en el LPC17xx



B U S I 2 C

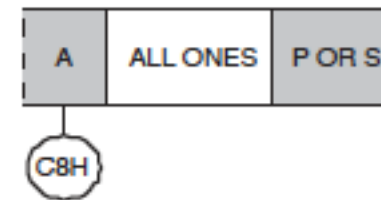
Slave Receiver Mode

reception of the own
Slave address and
one or more Data
bytes all are
acknowledged



arbitration lost as
Master and
addressed as Slave

last data byte
transmitted. Switched
to Not Addressed
Slave (AA bit in
I2CON = "0")





Índice de la Lección

- **Introducción a la comunicación entre dispositivos mediante un bus serie**
- **Características del bus I²C**
- **Conexión de dispositivos al bus: nivel físico**
- **Intercambio de información: nivel de enlace**
- **Generación del protocolo desde un microcontrolador**
- **Ejemplos de dispositivos que utiliza el bus I²C**
- **Unidad I2C del LPC17xx**
- **Conclusiones**
- **Bibliografía**



Conclusiones



B U S I 2 C

Conclusiones

Un bus serie permite reducir el tamaño general del circuito

El protocolo del bus I²C contempla

Conexión de nivel físico

Intercambio de datos

El protocolo se puede implementar

Utilizando puertos de entrada/salida → maestro

Unidad específica

Gran cantidad de dispositivos compatibles con I²C



Índice de la Lección

- **Introducción a la comunicación entre dispositivos mediante un bus serie**
- **Características del bus I²C**
- **Conexión de dispositivos al bus: nivel físico**
- **Intercambio de información: nivel de enlace**
- **Generación del protocolo desde un microcontrolador**
- **Ejemplos de dispositivos que utiliza el bus I²C**
- **Unidad I2C del LPC17xx**
- **Conclusiones**
- **Bibliografía**



Referencias

Bibliografía

“El bus I2C. De la teoría a la práctica”

Dominique Paret Ed. Paraninfo. 1995. (ISBN: 84-283-2167-1)

“The I2C-BUS Specification” Versión 2.1 Enero 2000

Hojas de características de los componentes explicados



Protocolo de Comunicación I²C



B
U
S

I
2
C

Referencias

Enlaces de Internet

The I2C FAQ

<http://www.ping.be/~ping0751/i2cfaq/i2cindex.htm>

Single Master I2C driver routines

<http://www.specs.de/users/danni/appl/hard/i2c/>

I2C Bus Technical Overview and Frequently Asked Questions (FAQ)

by Axel Wolf, ESAcademy based on the I2C FAQ by Vince Himpe

Embedded Systems Academy:

<http://www.esacademy.com/faq/i2c/index.htm>

