



Laboratorio 8:

Bus IIC

lectura/escritura de una EEPROM

Programación de sistemas y dispositivos

José Manuel Mendías Cuadros

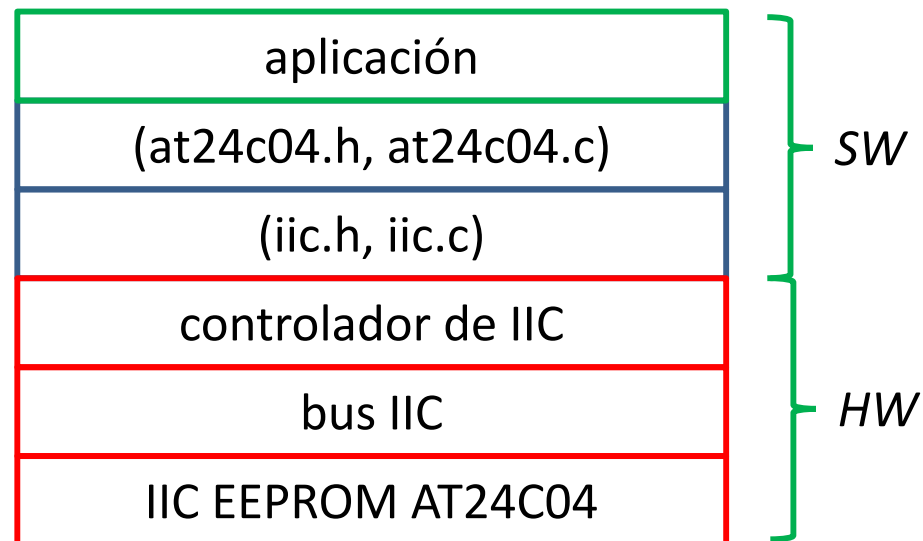
*Dpto. Arquitectura de Computadores y Automática
Universidad Complutense de Madrid*



Presentación



- Desarrollar 2 capas de firmware para el acceso por IIC a una EEPROM
 - o Una para el envío/recepción por pooling de bytes por bus IIC
 - Envío: `iic_putByte_start` / `iic_putByte` / `iic_putbyte_stop`
 - Recepción: `iic_putByte_start` / `iic_putByte` / `iic_putbyte_stop`
 - Inicialización: `iic_init` / `iic_open` / `iic_close`
 - o Otra para la lectura/escritura de una EPROM conectada a un bus IIC
 - Lectura/escritura aleatoria de 1 byte: `at24c04_bytewrite` / `at24c04_byte_read`
 - Lectura/escritura secuencial de toda la memoria: `at24c04_load` / `at24c04_store`
 - Inicialización: `at24c04_clear`



Bus IIC



- **IIC (Inter Integrated Circuits) es un bus serie síncrono multi-master**
 - Tiene 2 líneas bidireccionales: **SDA** (datos serie) y **SCL** (reloj).
 - Si el bus está libre ambas están en alta.
 - Comunicación master-slave
 - El maestro gobierna el inicio/fin de las transferencia y genera el reloj.
 - Puede haber varios maestros/esclavos conectados en un mismo bus
 - Existe un mecanismo de arbitraje.
 - Todos los datos transmitidos son de 8 bits (MSB first) y deben ser reconocidos individualmente (ACK).
 - Soporta altas tasas de transferencia (hasta 400 Kb/s)
- **Protocolo básico:**
 - El **maestro inicia la transmisión** generando la **start condition** (transición 1-0 en SDA).
 - todos los esclavos se ponen en alerta
 - El **maestro envía la dirección del esclavo** (7b) y el **tipo de operación** R/W (1b)
 - todos esclavos comparan la dirección con la suya y el esclavo aludido envía ACK
 - si la dirección es de 10b se envía en 2 trozos
 - Se **transmiten un número indefinido de datos** (8b) **reconocidos individualmente** (1b).
 - El **maestro finaliza la transmisión** generando la **stop condition** (transición 0-1 en SDA)

Controlador de IIC

configuración

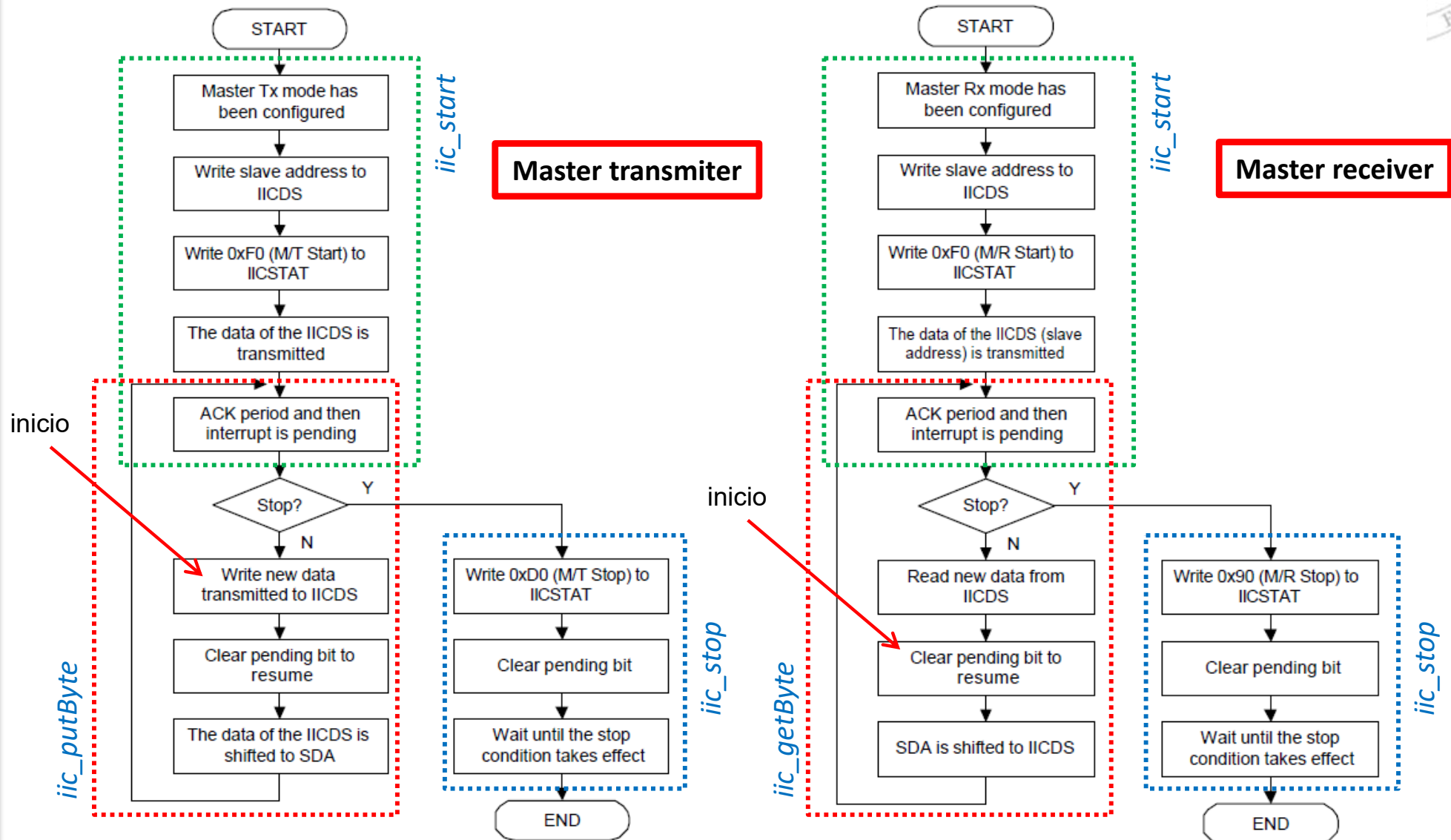


- Dirección como esclavo: no aplica
 - o IICADD = X
- Frecuencia de transmisión: 250 KHz
 - o IICCON[6] = 0 IICCLK = MCLK / 16
 - o IICCON[3:0] = 15 TxCLK = IICCLK / (15+1) = 64 MHz / 256
- Modo de transmisión: se define en cada transferencia
 - o IICSTAT[7:6] = XX
 - o ICCSTAT[4] = 1 lectura/escritura habilitada
- Generación ACK: habilitada
 - o ICCCON[7] = 1
- Interrupciones Tx/Rx: habilitadas
 - o IICCON[5] = 1

- Resumen:
 - o IICADD = 0x0 (XXXX.XXXX)
 - o ICCON = 0xAF (101X.1111)
 - o ICCSTAT = 0x10 (XXX1.XXXX)

Controlador de IIC

organigramas de operación



Controlador de IIC

control de operación



- Para **arrancar la transmisión/recepción** de un byte:
 - Escribir un 0 en ICCCON[4]
 - El dato a transmitir debe escribirse previamente en ICCDS
 - El dato recibido estará disponible en ICCDS
- Para **conocer el estado de la transmisión/recepción**:
 - Consultar ICCCON[4]: vale 1 si ha finalizado, vale 0 si está en curso
 - En el caso de transmisión, ésta finaliza cuando se recibe ACK
 - En el caso de la recepción, ésta finaliza cuando se recibe el dato
- Para **generar la start condition** en la primera transmisión
 - Escribir un 1 en ICCSTAT[5] antes de arrancar la transmisión
- Para **generar (o no) acknowledge** en la recepción de un dato
 - Escribir un 1 (o un 0) en IICCON[7]
- Para **generar la stop condition** después de la última recepción
 - Escribir un 0 en ICCSTAT[5]
 - Escribir un 0 en ICCCON[4] para arrancar la transmisión de la stop condition

Driver de controlador de IIC

icc.h



```
#ifndef __IIC_H__
#define __IIC_H__

#include <common_types.h>

#define IIC_Rx      (2)
#define IIC_Tx      (3) } Modos de funcionamiento del controlador actuando como master

#define RxACK       (1)
#define NO_RxACK    (0) } Macros para indicar si en recepción el controlador genera ACK o no

void iic_init( void );
void iic_start( uint8 mode, uint8 byte );
void iic_putByte( uint8 byte );
uint8 iic_getByte( uint8 ack );
void iic_stop( uint16 ms );

#endif
```

Driver de controlador de IIC

icc.c



```

void iic_start( uint8 mode, uint8 byte )
{
    IICDS    = ...; ..... escribe el dato (dirección del dispositivo) en el registro de transmisión
    IICSTAT  = ...; ..... fija modo de funcionamiento(mode), genera start condicion, habilita Tx/Rx
    IICCON  &= ...; ..... arranca la transmisión del dato
    while( ... ); ..... espera el fin de la transmisión (recepción de ACK)
}

void iic_putByte( uint8 byte )
{
    IICDS    = ...; ..... escribe el dato en el registro de transmisión
    IICCON  &= ...; ..... arranca la transmisión del dato
    while( ... ); ..... espera el fin de la transmisión (recepción de ACK)
}

uint8 iic_getByte( uint8 ack )
{
    IICCON  = ...; ..... indica si se genera o no ACK (ack)
    IICCON  &= ...; ..... arranca la recepción del dato
    while( ... ); ..... espera la recepción del dato
    return ...; ..... devuelve el dato
}

void iic_stop( uint16 ms )
{
    IICSTAT &= ...; ..... genera stop condicion
    IICCON  &= ...; ..... arranca la transmisión del bit de stop
    sw_delay_ms( ms ); ..... espera a que haga efecto la stop condition ya que no se recibe ACK
}
    
```

IIC EEPROM AT24C04

características



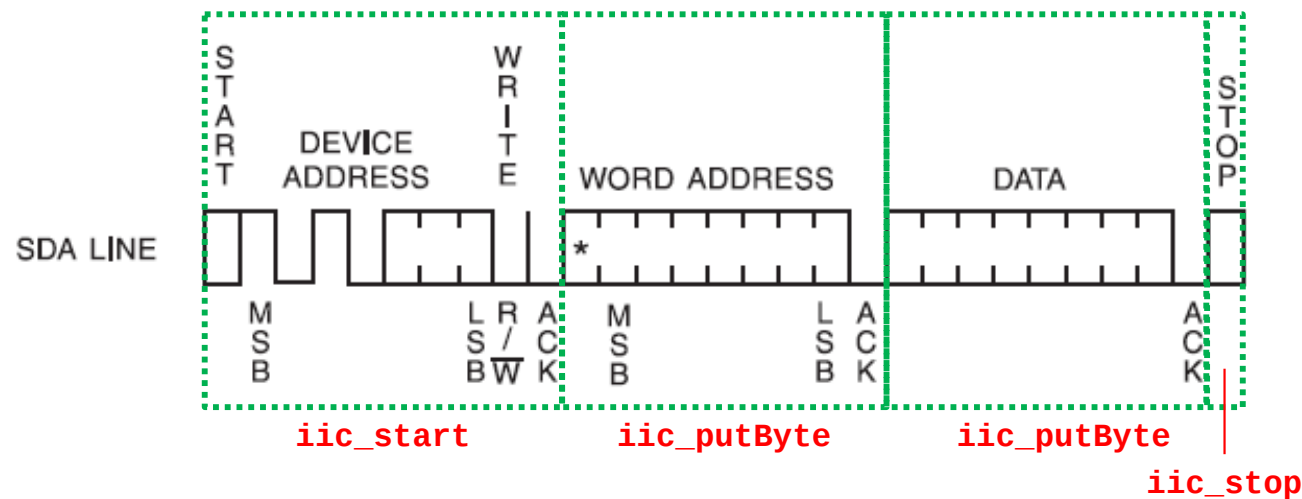
- Memoria **EEPROM 4Kb** con interfaz IIC (AT24C04)
 - Organizada lógicamente como 512 palabras de 8 bits (32 páginas de 16B)
 - 8 bits de datos y 9 bits de dirección
 - Frecuencia máxima de transmisión: 400 Kb/s
- Dirección del dispositivo: 1010000
 - 1010: genérico de las EEPROM
 - 000: por la conexión del los pines A0..A2
- 2 tipos de escritura:
 - 1 byte aleatorio
 - 1 página (16B) aleatoria
- 3 tipos de lectura:
 - 1 byte aleatorio
 - Secuencial
 - Última dirección accedida + 1



IIC EEPROM AT24C04

cronogramas (i)

- Para hacer una **escritura aleatoria** de un dato en una dirección dada
 - Es necesario enviar 3 bytes por IIC



1K/2K	1	0	1	0	A ₂	A ₁	A ₀	R/W
	MSB				LSB			
4K	1	0	1	0	A ₂	A ₁	P0	R/W
8K	1	0	1	0	A ₂	P1	P0	R/W
16K	1	0	1	0	P2	P1	P0	R/W

dirección del dispositivo

- Se envía la start condition seguida del 1er. byte que incluye:
 - los 6 bits más significativos de la dirección del dispositivo IIC (101000)
 - el bit 8 de la dirección del dato
 - un 0 para indicar la operación de escritura
- Se envía el 2do. byte que incluye los bits 7..0 (8 bits) de la dirección del dato.
- Se envía el 3er. byte que incluye el dato (8 bits)
- Se envía la stop condition.

IIC EEPROM AT24C04

cronogramas (ii)



```
#define DEVICE_ADDR (( 0xA << 4) | (0 << 1))

#define READ (1)
#define WRITE (0)

void at24c04_bytewrite( uint16 addr, uint8 data )
{
    uint8 page;

    page = (addr & 0x100) >> 8; ..... Se queda con el bit 8 de la dirección

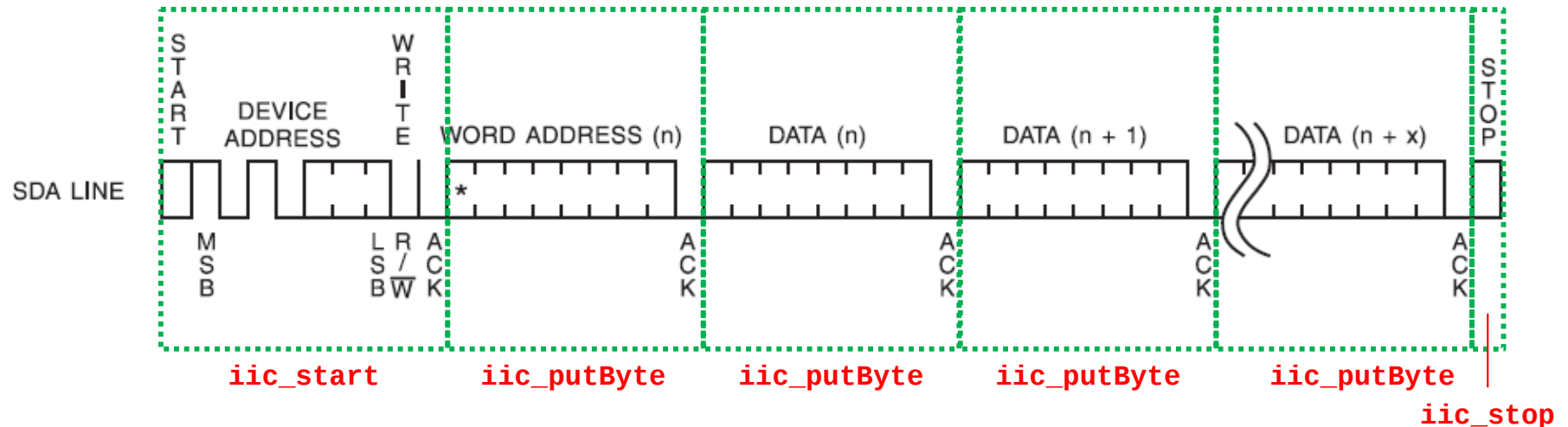
    iic_start( IIC_Tx, DEVICE_ADDR | (page << 1) | WRITE );
    iic_putByte( addr & 0xFF ); ..... Envía los bits 7..0 de la dirección
    iic_putByte( data );
    iic_stop( 5 ); ..... En el AT24C04, twr = 5 ms
}
```



IIC EEPROM AT24C04

cronogramas (ii)

- Para hacer una **escritura de página** (16 bytes) desde una dirección dada
 - Es necesario enviar 18 bytes por IIC



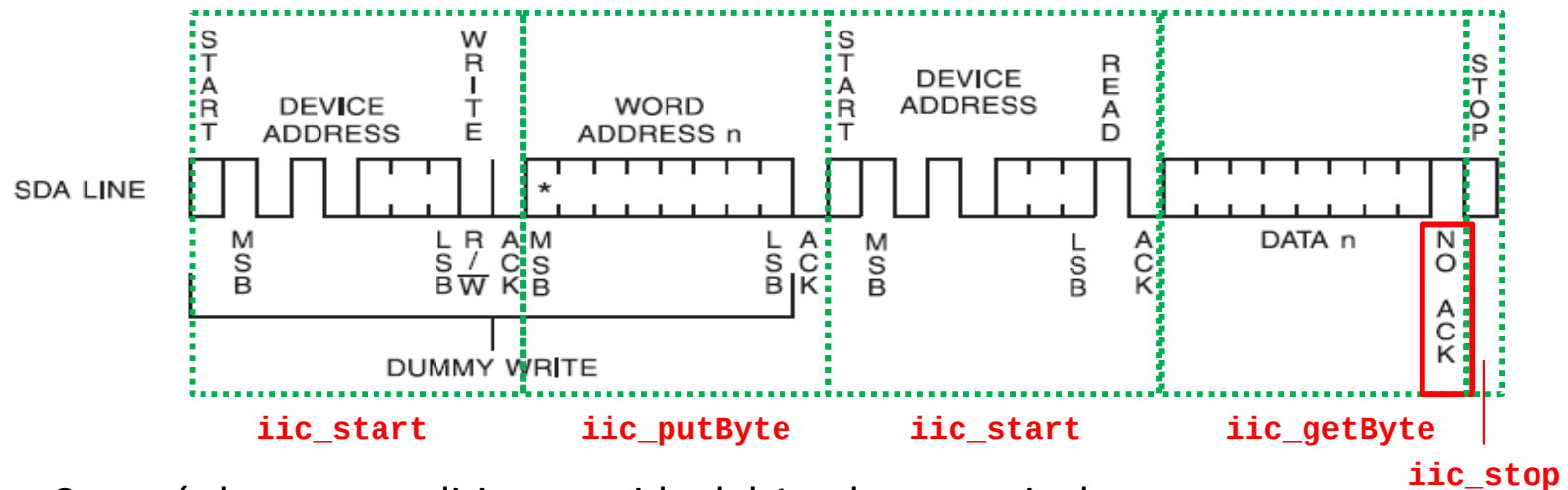
- Se envía la start condition seguida del 1er. byte que incluye:
 - los 6 bits más significativos de la dirección del dispositivo IIC (101000)
 - el bit 8 de la dirección del dato
 - un 0 para indicar la operación de escritura
 - Se envía el 2do. byte que incluye los bits 7..0 (8 bits) de la dirección del dato.
 - Se envían del 3er. al 18vo. byte que incluyen los 16 primeros datos (8 bits cada uno).
 - Se envía la stop condition.
- Para escribir la memoria completa el proceso se repite 32 veces

IIC EEPROM AT24C04

cronogramas (iii)



- Para hacer una **lectura aleatoria** de un dato de una dirección dada
 - Es necesario enviar 3 bytes por IIC y recibir 1



- Se envía la start condition seguida del 1er. byte que incluye:
 - los 6 bits más significativos de la dirección del dispositivo IIC (101000)
 - el bit 8 de la dirección del dato
 - un 0 para indicar la operación de escritura (escritura ficticia)
- Se envía el 2do. byte que incluye los bits 7..0 (8 bits) de la dirección del dato.
- Se envía la start condition seguida del 3er. byte que incluye:
 - La misma información que el 1er. byte pero con un 1 final para indicar la operación de lectura
- Se recibe el byte que incluye el dato (sin enviar ACK).
- Se envía la stop condition

- o Se envía la start condition seguida del 1er. byte que incluye:
 - los 6 bits más significativos de la dirección del dispositivo IIC (101000)
 - el bit 8 de la dirección del dato
 - un 0 para indicar la operación de escritura (escritura ficticia)
- o Se envía el 2do. byte que incluye los bits 7..0 (8 bits) de la dirección del dato.
- o Se envía la start condition seguida del 3er. byte que incluye:
 - La misma información que el 1er. byte pero con un 1 final para indicar la operación de lectura
- o Se reciben n-1 bytes que incluyen los datos (enviando ACK en cada uno)
- o Se recibe el último byte que incluye datos (sin enviar ACK).
- o Se envía la stop condition.

Driver de IIC EEPROM AT24C04

at24c04.h



```
#ifndef __AT24C04_H__
#define __AT24C04_H__

#include <common_types.h>

#define AT24C04_WIDTH (8)
#define AT24C04_DEPTH (512) } Tamaño de la EEPROM

void at24c04_clear( void ); ..... Realizará 32 escrituras de páginas
void at24c04_bytewrite( uint16 addr, uint8 data );
void at24c04_byteread( uint16 addr, uint8 *data ); } Lectura/escritura aleatoria
void at24c04_store( uint8 *buffer );
void at24c04_load( uint8 *buffer ); ..... Realizará una lectura secuencial de datos

#endif
```

Acerca de *Creative Commons*



■ Licencia CC (*Creative Commons*)

o Ofrece algunos derechos a terceras personas bajo ciertas condiciones. Este documento tiene establecidas las siguientes:



Reconocimiento (*Attribution*):

En cualquier explotación de la obra autorizada por la licencia hará falta reconocer la autoría.



No comercial (*Non commercial*):

La explotación de la obra queda limitada a usos no comerciales.



Compartir igual (*Share alike*):

La explotación autorizada incluye la creación de obras derivadas siempre que mantengan la misma licencia al ser divulgadas.

Más información: <https://creativecommons.org/licenses/by-nc-sa/4.0/>