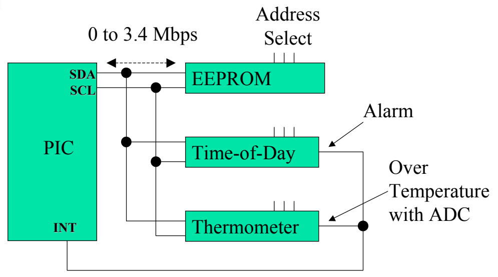
I²C – Inter-Integrated Circuit Communication

Embedded Systems Interfacing

Overview

- Specification
- Hardware
- Peripherals

Typical System



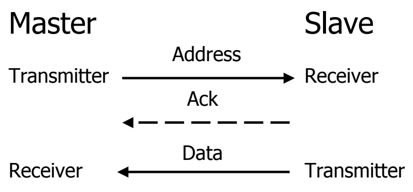
Specification

- Multiple versions
  - Version 1.0 1992
    - Up to 400 Kbps
    - 10 bit addresses
  - Version 2.0 1998
    - Up to 3.4 Mbps
    - New signal levels for High-speed operation
  - Version 2.1 2000
    - Some timing conditions relaxed
  - SMBus Version 2.0
    - Relaxed version of I²C specification for power management (10 Kbps to 100 Kbps)

Terminology

- Transmitter – device that sends data to the bus
- Receiver – device that accepts data from bus
- Master – device which initiates a transfer, generates clock and terminates transfer with NAK
- Slave – device addressed by master
- Multi-master – more then one device attempts control of bus at one time without corrupting message

Terminology

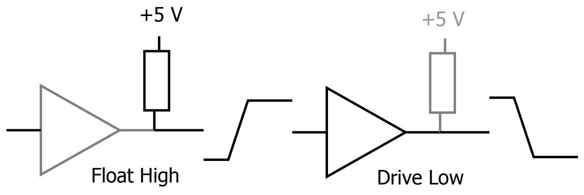


Terminology

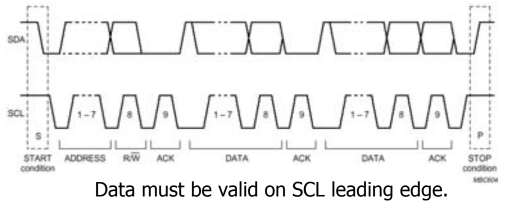
- Arbitration – procedure to ensure that, if more than one master simultaneously tries to control the bus, only one is allowed to do so and the winning message is not corrupted
- Synchronization – procedure to synchronize the clock signals of two or more devices

Signals Levels

- Float High (logic 1)
- Drive Low (logic 0)



Data Transfer on I²C-bus

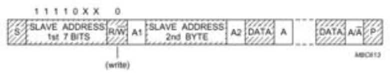


Data must be valid on SCL leading edge.

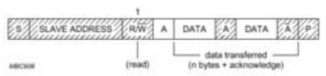
Start

sto**P**

Data Transfer on I²C-bus

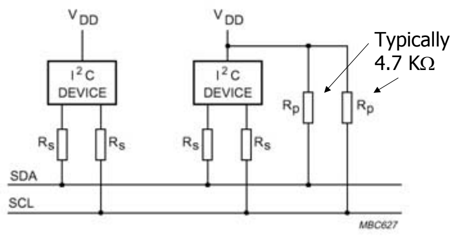


10-bit Address

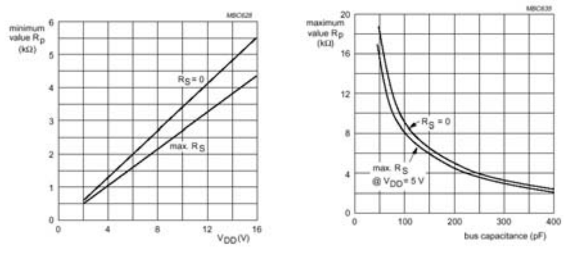


7-bit Address

Signal Conditioning

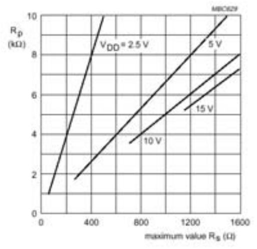


Pull-up Resistance – Standard Mode



Copyright James Grover, 2008

Series Resistance – Standard Mode



Other topics

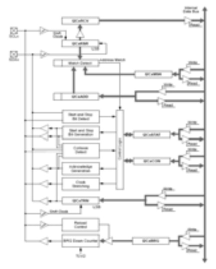
- General call
- High-speed mode (Hs-mode) – 0 to 3.4 Mbps
- Signal parameters
  - Levels
  - Timing

Hardware

- Six hardware registers to support I²C
  - I²C Control Register (I2CxCON)
  - I²C Status Register (I2CxSTAT)
  - I²C Transmit Register (I2CxTRN)
  - I²C Receive Rgister (I2CxRSR
  - I²C Address Register (I2CxADD)
  - I²C Mask Register (I2CxMSK)
- Tool Kit
  - Start
  - stoP
  - Data
  - Ack

Hardware Modes

- Three mode of I²C supported
  - I²C Master Mode in single master system
  - I²C Slave Mode
  - I²C master/Slave Mode in multi-master system
- Baud Rate Generator
  - 100 Kbps
  - 400 Kbps
  - 1 Mbps



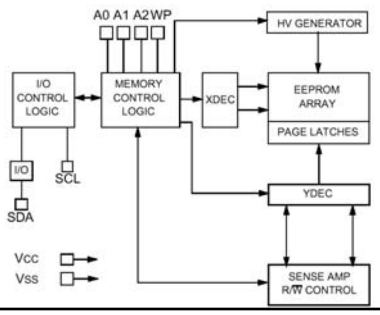
I²C Master Mode Initialization

```
void initI2C1(unsigned char addrMode){
        I2C1CONbits.I2CEN=1;            //start module and configure
                                        //SDA and SCK
        if(addrMode==10)
                I2C1CONbits.A10M=1;     //set 10-bit address mode
        else
                I2C1CONbits.A10M=0;     //set 7-bit address mode
        I2C1BRG=0x27;                   //Fosc=8 MHz, SCK=100 KHz
        }
```

Peripherals

- Device Manufactures
  - Microchip
  - National Semiconductor
  - XPS (Phillips)
  - Seimens
  - Xicor
  - Zilog
- Devices
  - Memory
  - Clock / Calendar
  - Bus Expanders
  - LCD Controllers
  - ADC
  - Audio / Video Devices

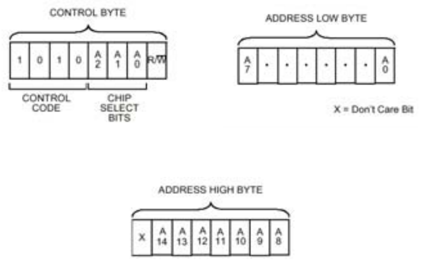
24AA256 / 24LC256 / 24FC256



Characteristics

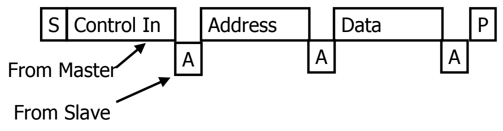
- 256 Kbit as 32 K by 8
- Self timed 5 ms Erase / Write cycle
- 64-byte page write mode (5 ms cycle time)
- 100,000 Erase / Write cycles
- > 200 year data retention
- Transfer rates
  - 400 Kbps for 24AA256 and 24LC256
  - 1 Mbps 24FC256

Slave Address

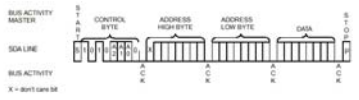


Byte Write

- S → Start
- A → Acknowledge
- P → stoP

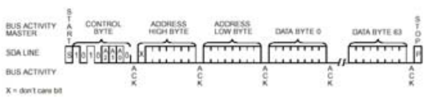


Byte and Page Write

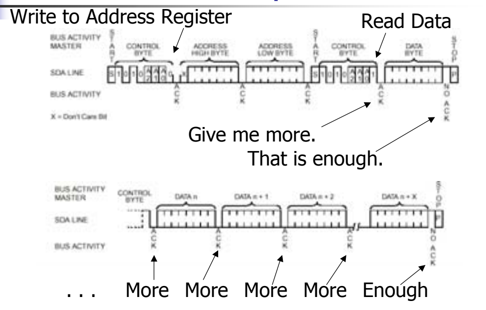


•Slave generates ACK

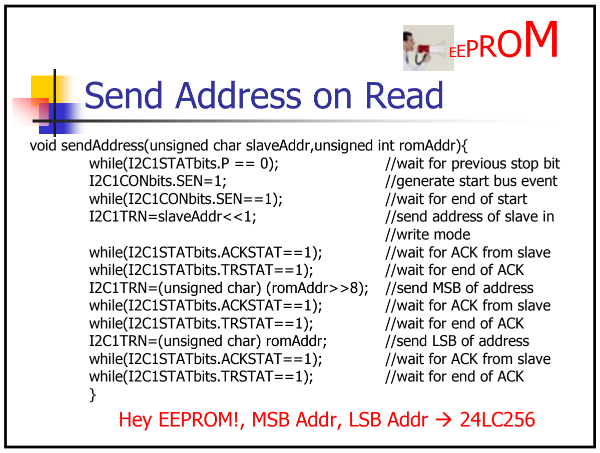
•Slave accepts data until stoP.



Random and Sequential Read



Embedded Systems Interfacing

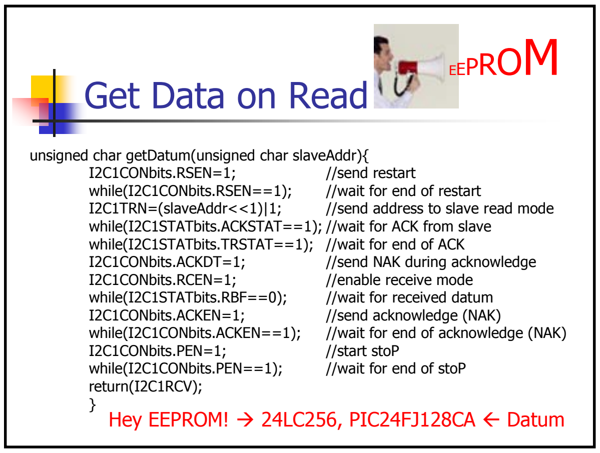


EEPROM

Send Address on Read

```
void sendAddress(unsigned char slaveAddr,unsigned int romAddr){
        while(I2C1STATbits.P == 0);                //wait for previous stop bit
        I2C1CONbits.SEN=1;                         //generate start bus event
        while(I2C1CONbits.SEN==1);                 //wait for end of start
        I2C1TRN=slaveAddr<<1;                      //send address of slave in
                                                   //write mode
        while(I2C1STATbits.ACKSTAT==1);            //wait for ACK from slave
        while(I2C1STATbits.TRSTAT==1);             //wait for end of ACK
        I2C1TRN=(unsigned char) (romAddr>>8);      //send MSB of address
        while(I2C1STATbits.ACKSTAT==1);            //wait for ACK from slave
        while(I2C1STATbits.TRSTAT==1);             //wait for end of ACK
        I2C1TRN=(unsigned char) romAddr;           //send LSB of address
        while(I2C1STATbits.ACKSTAT==1);            //wait for ACK from slave
        while(I2C1STATbits.TRSTAT==1);             //wait for end of ACK
        }
```

Hey EEPROM!, MSB Addr, LSB Addr → 24LC256

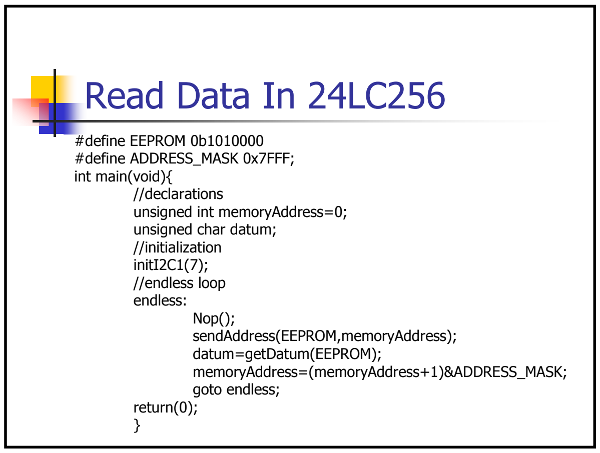


EEPROM

Get Data on Read

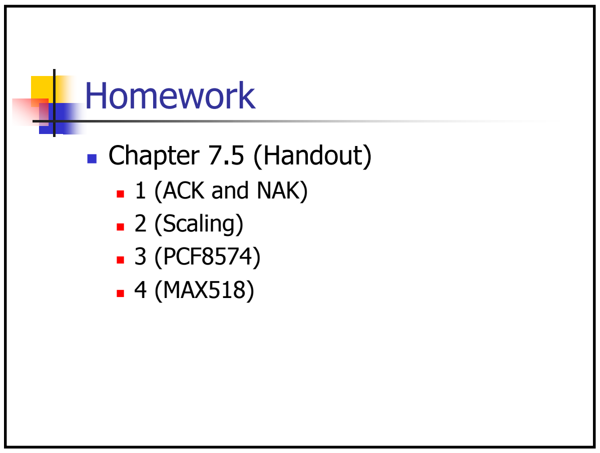
```
unsigned char getDatum(unsigned char slaveAddr){
        I2C1CONbits.RSEN=1;                  //send restart
        while(I2C1CONbits.RSEN==1);          //wait for end of restart
        I2C1TRN=(slaveAddr<<1)|1;            //send address to slave read mode
        while(I2C1STATbits.ACKSTAT==1); //wait for ACK from slave
        while(I2C1STATbits.TRSTAT==1);       //wait for end of ACK
        I2C1CONbits.ACKDT=1;                 //send NAK during acknowledge
        I2C1CONbits.RCEN=1;                  //enable receive mode
        while(I2C1STATbits.RBF==0);          //wait for received datum
        I2C1CONbits.ACKEN=1;                 //send acknowledge (NAK)
        while(I2C1CONbits.ACKEN==1);         //wait for end of acknowledge (NAK)
        I2C1CONbits.PEN=1;                   //start stoP
        while(I2C1CONbits.PEN==1);           //wait for end of stoP
        return(I2C1RCV);
        }
```

Hey EEPROM! → 24LC256, PIC24FJ128CA ← Datum



Read Data In 24LC256

```
    #define EEPROM 0b1010000
    #define ADDRESS_MASK 0x7FFF;
    int main(void){
            //declarations
            unsigned int memoryAddress=0;
            unsigned char datum;
            //initialization
            initI2C1(7);
            //endless loop
            endless:
                    Nop();
                    sendAddress(EEPROM,memoryAddress);
                    datum=getDatum(EEPROM);
                    memoryAddress=(memoryAddress+1)&ADDRESS_MASK;
                    goto endless;
            return(0);
            }
```



Homework

- Chapter 7.5 (Handout)
  - 1 (ACK and NAK)
  - 2 (Scaling)
  - 3 (PCF8574)
  - 4 (MAX518)