CoE 115 Lab 6 – I^2C

Objectives

- Properly setup and utilize the I^2C functionality of the PIC24FJ microcontroller
- Interface an I^2C EEPROM to the microcontroller

Materials

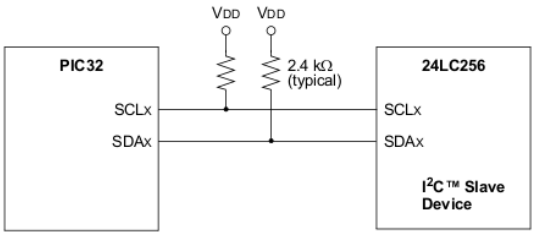
- 24LC64 I^2C EEPROM
- LEDs, Jumper wires, Pull-up resistors

Overview and Setup

The Inter-Integrated Circuit™ (I^2C™) is a serial interface useful for communicating with other peripheral or microcontroller devices. The I^2C bus is a two-wire serial interface, where the two lines are designated as SCL and SDA. The **SDA** is the data line while the **SCL** is the clock signal and is used to synchronize the events over the I^2C bus.

The primary I^2C lines in PIC24FJ are found in pins 17 (SCL) and 18 (SDA). The corresponding SCL and SDA pins of the slave device (ie. EEPROM) should be connected to these lines as well. External pull-up resistors are needed to ensure a high level when no device is pulling the line down, as shown in the sample diagram below. Note that only one set of pull-up resistors is needed for the entire I^2C system.

For the 24LC64 EEPROM device, connect all address bits to GND. Also, wire up the EEPROM such that write-protect is disabled.



We will use **RB3, RB2, RB1, RB0** to display the values read from the EEPROM chip. Set up an active-low LED with external pull-up resistors for each RB output. The LED connected to RB3 should be the most significant bit while RB0 should be the least significant bit.

Note on I^2C I/O: When the module is enabled, assuming no other module with higher priority has control, the module will assume control of the SDAx and SCLx pins. The module software need not be concerned with the state of the port I/O of the pins, the module overrides the port state and direction.

I^2C Configuration in PIC24FJ

I^2C operation for the PIC24FJ microcontroller is controlled by the following registers:

- I2C1CON (or I2C1CONL) enables control of the I^2C operation
- I2C1STAT contains status flags indicating the I^2C module's state during operation
- I2C1BRG holds the Baud Rate Generator (BRG) reload value
- I2C1TRN bytes are written to this register during a transmit operation
- I2C1RCV buffer register from which data bytes can be read

You must initialize the I^2C module of your microcontroller such that following are satisfied:

- PIC24FJ microcontroller will act as the I2C master.
- I2C interrupts will be used.
- Discontinue module operation when device enters an idle mode.
- Send ACK during Acknowledge.
- Use FSCL = 400 kHz.

Seatwork

- You are required to perform read/write operations on the EEPROM device. Refer to the EEPROM datasheet for the operational details.
- To do so, you will need to implement the basic routines that would allow you to transmit and receive data over the I^2C bus. You may use "lab06_guide.c" as a reference to complete the required I2C primitive functions.
- Present your EEPROM device to your instructor for data initialization.

A. EEPROM Write

- Write the string **"CoE_115"** (null character included) on your EEPROM. The string be written starting on address 0x1000.

B. EEPROM Read

- You will be each given an EEPROM address. Read 32 consecutive bytes starting from this address.
- Once you have read the 32-byte input sequence, compute for the hash code using the hash function below.

```
// DJB2 hash algorithm
unsigned long hash(unsigned char *str)
    {
        unsigned long hash = 5381;
        int c;

        while (c = *str++)
```

```
            hash = ((hash << 5) + hash) + c;
        return hash;
    }
```

- The resulting hash value will be shorter than the original sequence and is represented as an unsigned long (4 bytes).
- You need to display the value of the hash code by flashing each hex digit using the LEDs. The equivalent binary value must be displayed through the LEDs connected to RB[3:0].
    - Example: The resulting hash code is 0xADF27302.
    - Display the hash code sequence as: 0xA, 0xD, 0xF, 0x2, 0x7, 0x3, 0x0, 0x2