| Class: | CPE301L Digital Systems Architecture and Design 1001 | | Semester: | Spring 2025 |
|---|---|---|---|---|
| Points | | Document author: | Narek Kalikian | |
| | | Author's email: | kalikn1@unlv.nevada.edu | |
| | | | | |
| | | Document topic: | Postlab 10 | |
| Instructor's comments: | | | | |

## 1. Introduction / Theory of Operation

In this lab, we worked on inter-intergrated circuits (I2C). I2C is another serial communication protocol that is used for transferring data between two integrated circuits. I2C is a two-wire interface that relies on the Serial Data and Serial Clock pins in AVR and also supports master-slave relations among its devices. It uses the following two-wire interface AVR registers: TWAR, TWBR, TWCR, TWDR, and TWSR. Specifically, in the lab, we integrated the DS1307 and the MAX232N chips together with our ATmega328p microcontroller to read the current date and time and display it on the LCD every 30 seconds.

## 2. Prelab Content

**DS1307 Operation:** The DS1307 is a real-time clock chip that keeps track of time and date (in both 12-hour and 24-hour modes) using an I2C interface. It stores data and includes 56 bytes of non-volatile RAM. It can continue working using a backup battery even when the main power is lost. It communicates with a microcontroller to read and write to and from time and date registers.

**Function of All Registers:**

- TWAR (TWI Slave Address Register): Stores the 7-bit address of the AVR when it operates as a slave device in I2C communication
- TWBR (TWI Bit Rate Register): Controls the speed of the I2C bus by setting the bit rate for the SCL clock frequency
- TWCR (TWI Control Register): Manages and controls the operation of the TWI module
- TWDR (TWI Data Register): Holds the data being transmitted or received over the I2C bus
- TWSR (TWI Status Register): Reflects the current status of the TWI logic

# 3. Description of Experiments

## Experiment 1 - Code:

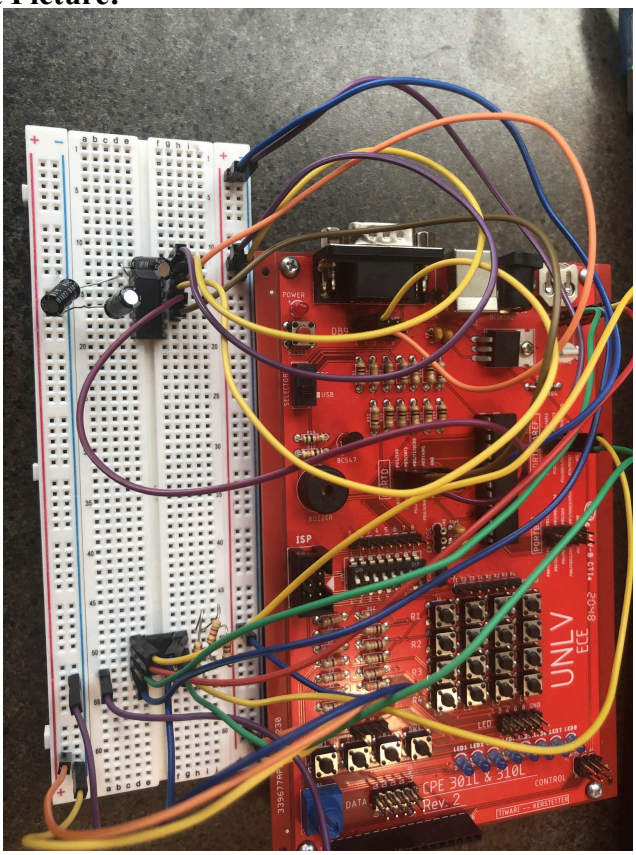
```
#define F_CPU 8000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <stdio.h>

// --- USART Functions ---

void Initialize_usart()
{
    UBRR0H = 0;
    UBRR0L = 51; // 9600 baud @ 8 MHz
    UCSR0B = (1 << TXEN0) | (1 << RXEN0);
    UCSR0C = (1 << UCSZ01) | (1 << UCSZ00); // 8-bit, no parity, 1 stop
}

void USART_send(unsigned char data)
{
    while (!(UCSR0A & (1 << UDRE0)));
    UDR0 = data;
}

void USART_send_string(const char* str)
{
    while (*str)
    {
        USART_send(*str++);
    }
}

// --- I2C Functions ---

void i2c_init()
{
    TWSR = 0x00;
    TWBR = 0x48; // 50kHz SCL
    TWCR = (1 << TWEN);
}

void i2c_start()
{
    TWCR = (1 << TWINT) | (1 << TWSTA) | (1 << TWEN);
    while (!(TWCR & (1 << TWINT)));
}
```

```
void i2c_write(unsigned char data)
{
    TWDR = data;
    TWCR = (1 << TWINT) | (1 << TWEN);
    while (!(TWCR & (1 << TWINT)));
}

unsigned char i2c_read(unsigned char ack)
{
    if (ack)
    TWCR = (1 << TWINT) | (1 << TWEN) | (1 << TWEA);
    else
    TWCR = (1 << TWINT) | (1 << TWEN);

    while (!(TWCR & (1 << TWINT)));
    return TWDR;
}

// --- Set Initial Time (Optional) ---

void set_time()
{
    i2c_start();
    i2c_write(0xD0);
    i2c_write(0x00); // Start at register 0
    i2c_write(0x00); // Seconds
    i2c_write(0x52); // Minutes
    i2c_write(0x12); // Hours
    i2c_write(0x05); // Day of week (1=Sunday...7=Saturday)
    i2c_write(0x25); // Date (25)
    i2c_write(0x04); // Month (April)
    i2c_write(0x24); // Year (2024)
    i2c_stop();
}

// --- Read Full Time/Date ---

void read_full_time(unsigned char* hr, unsigned char* min, unsigned char* sec,
unsigned char* day, unsigned char* date,
unsigned char* month, unsigned char* year)
{
    i2c_start();
    i2c_write(0xD0); // Write mode
    i2c_write(0x00); // Start at seconds register
    i2c_start();
    i2c_write(0xD1); // Read mode

    *sec   = i2c_read(1);
    *min   = i2c_read(1);
    *hr    = i2c_read(1);
    *day   = i2c_read(1);
```

```
    *date  = i2c_read(1);
    *month = i2c_read(1);
    *year  = i2c_read(0); // Last byte: no ACK

    i2c_stop();
}

// --- Main Program ---

int main(void)
{
    unsigned char hr, min, sec, day, date, month, year;
    char buffer[80];

    Initialize_usart();
    i2c_init();
    set_time(); // Only run once to initialize DS1307

    while (1)
    {
        read_full_time(&hr, &min, &sec, &day, &date, &month, &year);

        sprintf(buffer, "Time: %02x:%02x:%02x  Date: %02x/%02x/20%02x \r\n",
        hr, min, sec, month, date, year);
        USART_send_string(buffer);

        _delay_ms(30000); // Wait 1 second
    }
}
```

```
Output
Show output from: Build
    Task "RunCompilerTask"
        Shell Utils Path C:\Program Files (x86)\Atmel\Studio\7.0\shellUtils
        C:\Program Files (x86)\Atmel\Studio\7.0\shellUtils\make.exe all --jobs 16 --output-sync
        make: Nothing to be done for 'all'.
    Done executing task "RunCompilerTask".
    Task "RunOutputFileVerifyTask"
                Program Memory Usage     :   2260 bytes   6.9 % Full
                Data Memory Usage        :   48 bytes   2.3 % Full
                Warning: Memory Usage estimation may not be accurate if there are sections other than .text sections in ELF file
    Done executing task "RunOutputFileVerifyTask".
Done building target "CoreBuild" in project "Lab10.cproj".
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\Narek\Documents\Atmel Studio\7.0\Lab10\Lab10\Lab10.cproj" (entry point):
Done building target "Build" in project "Lab10.cproj".
Done building project "Lab10.cproj".

Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
```

## Experiment 1 - Circuit Picture:




**Experiment 1 - Video Link:** https://www.youtube.com/watch?v=_jY9a11tqd4

## 4. Questions and Answers

**I2C:** I2C, also known as inter–intergrated circuit, is a two-wire serial communication protocol that supports multiple target devices and controllers for communicating: sending and receiving data. It uses a serial data line and serial clock line to operate.

**TWI:** TWI, also known as two-wire interface, is Atmel's implementation of I2C. It describes their version of I2C interfacing and protocols. Essentially, it's another way to describe I2C and isn't much different outside of terminology.

**SPL Uses:**

- Communication between devices
    - Microcontrollers
    - Display controllers like LCD
    - Memory devices like SD cards
    - Sensors
    - Other peripherals

## 5. Conclusions

This lab taught us how to use inter-intergrated circuits. We implemented two ICs (DS1307 and MAX232N) and our ATmega328p microcontroller to read and display the current date and time using the USART interface. We didn't really face many challenges during this lab, as it was limited to one experiment. However, the hardware setup was quite complex since we had to integrate two chips.