EMBEDDED SYSTEMS

Assignment

Master of Science Robotics Engineering

Group
Ankur Kohli (5160903)
Ammar Iqbal (5183355)
Basit Akram (5161322)

Submitted to
Prof. ENRICO SIMETTI

Dipartimento di Informatica, Bioingegneria, Robotica e
Ingegneria dei Sistemi (DIBRIS)
Universit'a degli Studi di Genova (UniGe)

Preface

The work is based on Embedded Systems which involves the implementation of timers, interrupts, SPI, UART, and so on. This assignment deals with MPLAB IDE Software, XC16 Compiler, HTerm serial software and Microchip Microcontroller Board.

Acknowledgements

We take this opportunity to express our sincere thanks to *Prof. ENRICO SIMETTI* for his guidance and support. We would also like to express our gratitude towards him for showing confidence in us. It was a privilege to have a great experience working under him in a cordial environment. We are very much thankful to the *University of Genoa*, for providing us the opportunity to pursue *M.Sc in Robotics Engineering* in a peaceful environment with ample resources. In the end, we would like to acknowledge our parents, friends. Without their support, this work would not have been possible.

Abstract

This assignemnt report is about the ***Embedded Systems*** *in which the of* ***timers, interrupts, SPI, UART, and so on*** *to determine about the operations performance. The specific goal in this case is that to know the implementation of operations with the real-time hardware. This report to bring the light on* ***Embedded Systems*** *for* ***operations****. Furthermore, the purpose of this report is to provide the approaches used during the development of code and implementation on microcontroller board. For this assignment, MPLAB IDE Software, XC16 Compiler and Microchip Microcontroller Board, HTerm serial software is used and also Embedded C programming is platform for the development of code.*

Contents

1 Introduction

A microprocessor-based computer system with software that is intended to carry out a specific task, either independently or as a component of a larger system, is known as an ***embedded system***. An integrated circuit built to perform computing for real-time processes is at the heart of the system.

From a single microcontroller to a group of connected processors with networks and peripherals, complexity can range from having no user interface to having intricate graphical user interfaces. Depending on the task for which it is created, an embedded system's complexity varies greatly.

Applications for embedded systems include hybrid cars, avionics, digital watches, microwaves, and more. Embedded systems consume up to 98 percent of all produced microprocessors.

2 Methodlogy

In this assignment, our main task is to implement the operations such as timers, interrupts, SPI, UART, and so on. Also, this assignment deals with MPLAB IDE Software, XC16 Compiler, HTerm serial software and Microchip Microcontroller Board. We used ***dsPIC30F4011 Enhanced Flash 16-bit Digital Signal Microcontroller Board*** as shown in figure 1, for the real-time implementation of the developed code.

Peripheral Features of the board:

- High current sink/source I/O pins: 25 mA/25 mA
- Timer module with programmable prescaler: Five 16-bit timers/counters; optionally pair 16-bit timers into 32-bit timer modules
- 16-bit Capture input functions
- 16-bit Compare/PWM output functions
- 3-wire SPI™ modules (supports 4 Frame modes)
- I2C™ module supports Multi-Master/Slave mode and 7-bit/10-bit addressing

- 2 UART modules with FIFO Buffers
- 1 CAN modules, 2.0B compliant

Source: dsPIC30F4011 Datasheet https://ww1.microchip.com/downloads/en/devicedoc/70135c.pdf

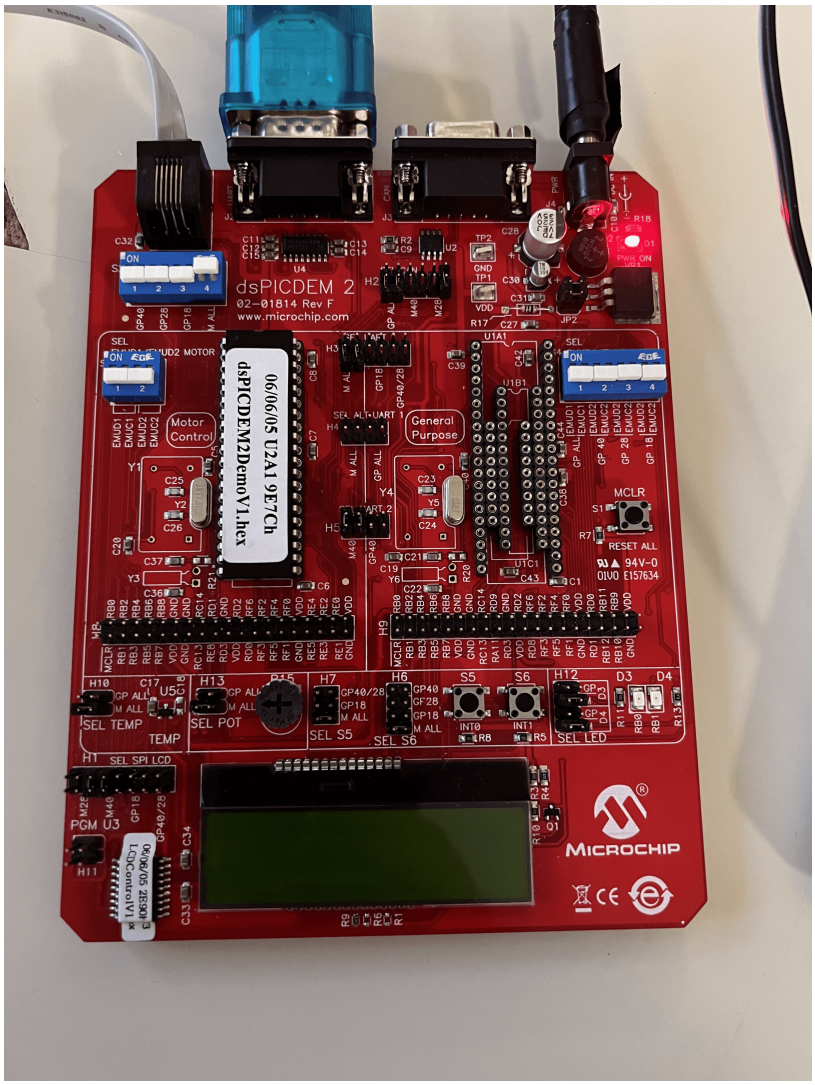

Figure 1: dsPIC30F4011 Board

DSP Engine Features:

- Dual data fetch
- Accumulator write back for DSP operations
- Modulo and Bit-Reversed Addressing modes
- Two, 40-bit wide accumulators with optional saturation logic

- 17-bit x 17-bit single cycle hardware fractional/ integer multiplier
- All DSP instructions single cycle
- ± 16-bit single cycle shift

Source: dsPIC30F4011 Datasheet https://ww1.microchip.com/downloads/en/devicedoc/70135c.pdf

Porgramm Burner: pickit3 Programmer

Figure 2, shows the program burner.

Figure 2: pickit3 Programmer

Requirements for the assignment

1. Simulate an algorithm that needs 7 ms for its execution, and needs to work at 100 Hz.
2. Read characters from UART and display the characters received on the first row of the LCD.
3. When the end of the row has been reached, clear the first row and start writing again from the first row first column.
4. Whenever a CR '\r' or LF '\n' character is received, clear the first row.
5. On the second row, write "Char Recv: XXX", where XXX is the number of characters received from the UART2. (use sprintf(buffer, "%d", value) to convert an integer to a string to be displayed.
6. Whenever button S5 is pressed, send the current number of chars received to UART2.

7. Whenever button S6 is pressed, clear the first row and reset the characters received counter.

Tips for Writing to LCD with SPI

The following power-up sequence should be observed by the user's application firmware when writing characters to the LCD:

1. After any reset operation wait 1000 milliseconds to allow the LCD to begin normal operation. The cursor on the LCD will be positioned at the top row on the left-most column.
2. Configure SPI1 module on your dsPIC30F device to operate in 8-bit Master mode. The serial clock may be set for any frequency up to 1 MHz.
3. To write an ASCII character to the LCD at the location pointed to by the cursor, load the SPIBUF register with the ASCII character byte.
4. After the character is displayed on the LCD, the cursor is automatically relocated to the next position on the LCD.
5. To reposition the cursor to another column on any of the two rows, write the address of the desired location to the SPIBUF register. Addresses in the first row of the LCD range from 0x80 to 0x8F, while addresses on the second row range from 0xC0 through 0xCF.
6. After 16 characters are written to the first row on the LCD, it is necessary for the user's application to write the address 0xC0 of the second row to the SPIBUF in order to roll the cursor over to the second row.
7. The user application must wait for a minimum of (8 bits / SPI Frequency) between writing two successive characters or addresses.

Source: Assignemnt pdf https://2022.aulaweb.unige.it/mod/assign/view.php?id=70551

Algorithm used for the accomplishment of the goal

```
void algorithm()
{
tmr wait ms(TIMER2, 7);
```

```
}
int main()
{
tmr setup period(TIMER1, 10);
while(1)
{
algorithm() ;
// code to handle the assignment
tmr wait period(TIMER1);
}
}
```

Approaches used to acomplish the task

I/O Pin Registers: This section provides information on the I/O ports for the dsPIC30F4011 device.

- ***TRISx*** is the register used to select which pins which pins of a register are to be used as input and which as output.
- ***LATx*** is register is used to write a logic value as pins output.

Example: read from pin D3

```
int pinValue;
TRISDbits.TRISD3 = 1; // set the pin as input
pinValue = PORTDbits.RD3; // read from pin
```

Timers:

TxCON: Timer X Control register

TON: initates the timer

TCKPS: used to set prescaler

TCS: clock source

Example: TxCON register

T1CON = 0; // sets all bits to zero

T1CONbits.TON = 1; // sets only the bit TON to 1

T1CONbits.TCKPS = 2; // sets TCKPS to binary 10

Interrupts: Interrupt controller reduces numerous peripheral interrupt requests to a single interrupt request signal to the CPU. Architecture of the interrupts is shown below:

void __attribute__((__interrupt__, __auto_psv__)) _INT0Interrupt()
{
IFS0bits.INT0IF = 0; // reset interrupt flag
... do some action
}

UART: The Universal Asynchronous Receiver Transmitter (UART) module is one of the serial I/O modules available in the dsPIC30F Switch Mode Power Supply (SMPS) device family. The UART is a full–duplex, asynchronous communication channel that communicates with peripheral devices and personal computers, using protocols such as RS–232, RS–485, LIN 1·2 and IrDA. The module also includes the IrDA encoder and decoder.

Example of sending character using UART
U1BRG = 11; // (7372800 /4) /(16 ∗9600) − 1
$U1MODEbits.UARTEN = 1; //enableUART$
$U1STAbits.UTXEN = 1; //enablingU1TX(mustbeafterUARTEN)$
$U1TXREG =' C'; //sendscharacter'C'$

SPI: The Serial Peripheral Interface (SPI) module is a synchronous serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be Serial EEPROMs, shift registers, display drivers, A/D converters, etc. The SPI module is compatible with Motorola's SPI and SIOP interfaces.

Example of writing character using SPI:
SPI1CONbits $\cdot MSTEN = 1;$
$SPI1CONbits \cdot MODE16 = 0;$

$SPI1CONbits \cdot PPRE = 3;$

$SPI1CONbits \cdot SPRE = 3;$

$SPI1STATbits \cdot SPIEN = 1;$

$while(SPI1STATbits.SPITBF == 1);$

$SPI1BUF =' x';$

$while(SPI1STATbits.SPITBF == 1);$

$SPI1BUF = 22;$

Here, in the above example firstly, master mode activated, secondly, 8 bit mode is set, than, primary prescaler set, also, secondary prescaler sets, now SPI is, here we waits for until not full, lastly, sets the character to be send and value to be sends, and the byte containing the value 22.

Note: All the aforementioned examples and codes referenced from the lecture notes of Embedded Systems *(source: https://2022.aulaweb.unige.it/course/view.php?id=4560).*

3 Results Discussion

In this section, we will discuss about the results accomplished during the development of the code and real-time implementation on **dsPIC30f4011** board. Figure 3, shows the outcomes of the task.

Figure 3: Result

In figure 3, first row shows the characters displayed on the first row of the LCD. First row

highlights the name of the group mates such as Basit Akram as Basit, Ankur Kohli as Ankur, and Ammar Iqbal as Ammar. In the message ***"BasitAnkurAmmar"*** there are total 15 number of characters and these are the characters received through UART and displayed using SPI on LCD. These characters displayed on the second row of the LCD as shown in figure 3.

4 Conclusion

The main objective of this assignment is to understand the concept of timers, interrupts, UART, SPI, and so on. By using the same we achieve the goal of the assign task. We able to receive and display message on the LCD screen and also able to dispaly the number of characters. Also, by pressing button S5, we manage to sends the current number of chars received to UART2. ALso, by pressing button S6, we manage to clear the first row and reset the characters received counter.