Lab Report

Microcontrollers

ELEC2850 Spring 2019

Laboratory Exercise # Final project

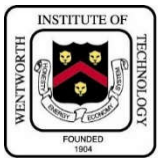
Laboratory Name: Remote sensing and control

Name: Jesus Esgueva

Lab Partner: Daniel Arpide

Lab Partner: Marc Ghannam

Date of Lab Exercise: 04/17/2019



WENTWORTH
Institute of Technology

Objective:

In this project, we aimed to achieve a simulation of sensing a remote sensor (external board) and displaying the value on our main board. This scenario is typical throughout the industry where a computer receives data from multiple boards to ensure that the devices(sensors) are working as expected.

The objective of the flow of data gathering. The voltage from the sensor is gathered by the ADC in the external board, these values are then sent over to the main board where it is converted and displayed in the 7-segment display.

Project requirements:

- Refresh data every 5 seconds.
- Data accuracy below ± 0.25V.

Procedure & Results:

The project was then divided into 4 steps:

- Program a timer.
- Program Analog to Digital converter
- Program 7 segment display.
- Program USART for Transmission and Reception of data.

Timer:

The timer was programmed with a comparator. The PIC16F877A uses an internal 20 MHz oscillator. The speed of this clock is 200ns. Then a comparator is used to compare the value of the counter to the value we want to reach before interrupting the CPU. In this case, the CPU needs to be interrupted every 10ms. The times 200ns go into 10ms is 50000 which is C350 in hexadecimal. When the timer reaches this value, it interrupts the CPU.

Once the comparator interrupts the CPU, the software is adjusted to loop through the program every 5 seconds. This is achieved by using a counter. Every time the comparator interrupts (10ms) the CPU the counter increases until it reaches 500 loops.

Analog to Digital converter:

The 10-bit analog to digital converter of the PIC16F877A is set up so there is one analog input (RA0) and the voltage references are obtained from the board power supply (VDD and VSS). The result is left justified meaning that the High register (ADRESH) contains the 8 most significant bit of the AD conversion and the low register (ADRESL) contains the 2 least significant bits.

The AD conversion is triggered every 5 seconds when conversion ends the data is sent over to USART buffer to be sent.

7 segment display:

The voltages received from the external board are displayed in the format [V.VVV] where the first display is going to be dotted. The 7-segment display works by outputting a high

or low signal to each of the segments. For this purpose, an 8-bit register is used. Then two arrays with the hexadecimal value for the 10 different values with and without the decimal point.

The USART receives two bytes. They need to be converted to decimal and to an appropriate form so they can be displayed in the 7-segment display. First, regroup both registers into an INT. ADRESH is multiplied by 2^2 to shift it two spaces to the left. Then ADRESL has shifted 6 spaces to the right and added together. These are the 10-bit value of the reading. To convert it to decimal, it is multiplied by 5 and divided by 1024. These number would have decimal values. But we want an INT. So, the value is multiplied by 1000. If before we had 1.345 V it then becomes 1345 V. These number is then split into four digits. Which are then passed to each display respectively and gives the output register the hex value of the digit?

Each display is lighted and turned off after displaying the digit, this part is run on the infinite loop so the displays are constantly turning On and OFF so fast that we can't notice it.

USART:

The USART is used as an asynchronous serial transmitter-receiver. It is set to use the serial ports.

For the transmission and receiving of data, we used a baud rate of 9600 which is how many bits per second are being transmitted. We used 9.6k because it was tied for the most accurate for baud rate with just 0.16% error, and we didn't need to transfer at very high speeds, so 19.2k would have been overkill.

Transmission:

Once the ADC is done. The transmission register checks, if it is empty it loads the first byte to the transmission register (TXREG) and waits until it is emptied to the transmission buffer. Then load the second byte and the transmission is finalized.

Reception:

the program checks if there is any data coming in if it is, it loads the values in the same order they come. First in First out (FIFO). These values are then forwarded to the Binary to decimal conversion.

If there is an overrun error, it resets the receiving device and the value showed in the 7-segment is 0.000 V.

Conclusion:

Overall, the project is working successfully. We were able to transmit and receive voltages and display them in the 7-segment display with a percentage error of 1% - 2%. The ADC is gathering data every 5 seconds then sending it, receiving it and ultimately displaying it. There was no need for calibration as the voltages obtained are very accurate and within range.

The project was a little challenging a first, it did help us learn how to read the datasheet. It isn't always easy to understand what it says, and small details make a big difference in the overall result of the project. Also, it is very important to understand the flow of the code. So, the right variable or flags are called or checked in the proper part of the code.

This lab can be applied to many areas. Nowadays the IoT is based on the same principle, you check the status of sensors across different devices on a network.

This lab has helped to improve our coding skills, a better understanding of the program flow and understanding how to apply all the resources given in the datasheet.

Annex:

	Task	Start date	End date	Days	Weeks	Completed
	Project report	3/21/2019	4/17/2019	27	3	YES
	Timer 5 Second	3/21/2019	4/4/2019	14	2	YES
J	Control registers	3/28/2019	4/3/2019	6	0	YES
E	Code implementation	4/4/2019	4/10/2019	6	0	YES
S	**Analog to Digital Converter**	3/21/2019	4/10/2019	20	2	YES
U	Control registers	3/29/2019	4/5/2019	7	1	YES
S	**Calibration-not needed**	4/8/2019	4/10/2019	2	0	NO
	Control registers EEprom	4/8/2019	4/9/2019	1	0	NO
	Implement code and save	4/9/2019	4/10/2019	1	0	NO
M	**7 Segment Display**	3/29/2019	4/16/2019	18	2	YES
A	Control registers	3/29/2019	4/16/2019	18	2	YES
R	Conversion to format (V.VVV)	4/5/2019	4/15/2019	10	1	YES
C	Code implementation	3/21/2019	4/16/2019	26	3	YES
D	**Transmissions (UART)**	3/21/2019	4/11/2019	21	3	YES
A	Control Registers for Transmitting	3/21/2019	3/28/2019	7	1	YES
N	Control Registers for Receiving	3/21/2019	3/29/2019	8	1	YES
I	Implementation with ADC	3/29/2019	4/4/2019	6	0	YES
E	Receive Data from other Micro	4/4/2019	4/11/2019	7	1	YES
L	Give received data to seven-seg code	4/4/2019	4/11/2019	7	1	YES
	Debug	4/11/2019	4/17/2019	6	0	YES
	Test software	4/11/2019	4/17/2019	6	0	YES


3/21/2019
3/26/2019
3/31/2019
4/5/2019
4/10/2019
4/15/2019
Project report
Timer 5 Second
Control registers
Code implementation
Analog to Digital Converter
Control registers
Calibration-not needed
Control registers EEprom
Implement code and save
7 Segment Display
Control registers
Conversion to format (V.VVV)
Code implementation
Transmissions (UART)
Control Registers for Transmitting
Control Registers for Receiving
Implementation with ADC
Receive Data from other Micro
Give received data to seven-seg code
Debug
Test sofware