Middle East Technical University
Department of Computer Engineering

CENG 336

Spring 2006

Term Project - Phase 2

Rev 1.1

Humidity Data Acquisition and Control with PICos18

Assigned: May 23 2007, Wednesday
Due: May 31 2007, Thursday, 23:55

1 Introduction

This part of the term project aims you to make you familiar with A/D Converter and UART Modules of PIC18F4520 by designing a communication program for a basic humitidy control process. You will also use some parts of *Phase*1 and combine them with this phase of the term project.

2 Your Mission

This phase of the term project includes two parts. Each part will be submitted seperately as explained in Secion 5.

2.1 Part I

In this part, you will use A/D Converter module, LCD and buttons. You will use the first phase of the term project with two minor differences:

1. The count value will be your `set value` and you will write this value to the last line of the LCD with the following format where XX is the set value (00 – 99):
`Set value = XX %`

2. You will not show the set value on the 7−segment displays. It is excluded in this phase of the project.

Initial value of the set value and the button functions are the same as in Phase 1.

What is new in this part is that you will use the potentiometer on the development board, Figure 1, to simulate a humidity sensor output and get an analog input for A/D Converter module.

The potentiometer is an electrical device which has a user-adjustable resistance. Usually, it has three terminals, like the one we have, as shown in Figure 2. The resistance on the *adjustable terminal* can be changed using a *sliding contact*. Therefore, we can use the potentiometer as a voltage dividing device. In this project, you will use the potentiometer to simulate a humidity sensor output by turning its contact (white circle in the center of potentiometer) and hence

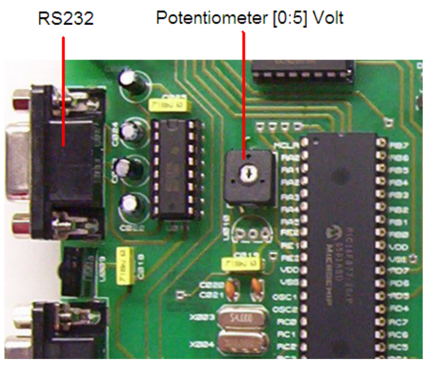

Figure 1: RS232 port and the potentiometer. You will use the potentiometer to simulate the output of a humidity sensor by turning its sliding contact and changing its output voltage.

changing its output voltage between 0 and 5 volts [1]. The adjustable terminal of the potentiometer is connected to the $RA0/AN0$ pin of PIC18F4520.

You will convert the output voltage of simulated humidity sensor to digital in order to determine the humidity level of the environment and use this digital value as explained below:

- ADC will have a $10-bit$ resolution. The result can be right or left justified. It is up to your implementation. ADC's reference voltage settings will be adjusted to use V_{DD} (5V) as V_{REF+} and V_{SS} (0V) as V_{REF-}.
- Interrupt feature of A/D Converter module **should** be used.
- If the value obtained from ADC is decimal $10(\sim 48.83mV)$, it will mean that the humidity is exactly 0 %. If the value is decimal $1000(\sim 4926.76mV)$, it will mean that the humidity is exactly 99 %. You can assume that the sensor output changes linearly.
- If the ADC result is less than decimal 10 or greater than decimal 1000, it indicates a sensor error.
- The state of the environment is determined as follows. Taking H as the current humidity of the environment and S as the set value,
  - **if $S > H$, the state is `Low humidity`**
  - if $S = H$, the state is `Normal humidity`
  - **if $S < H$, the state is `High humidity`**
  - if there is a sensor error, the state is `Sensor failure!`

You will make the A/D conversion with 50ms intervals and update the LCD accordingly as follows[2]:

- Current humidity value should be written to the first line of the LCD beginning from the first character with the following format where XX is the current humidity value $(00-99)$:
  `Humidity = XX %`

[1]When you turn the sliding contact to the left, the voltage on the $RA0/AN0$ pin will increase. If you turn it to the right, the voltage will decrease.

[2]Only the updates related to ADC is mentioned here. Do not forget the `set value` explained above!

- Current state (`Low humidity`, `Normal humidity`, `High humidity` or `Sensor failure!`) should be written to the second line of the LCD beginning from the first character.

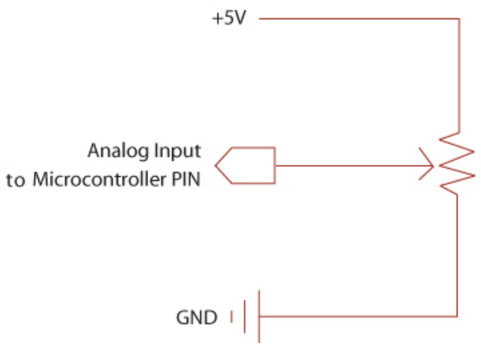

Figure 2: Schematic representation of a potentiometer.

2.2 Part II

In this part, you will add a communication feature to your project using EUSART Module of the *PIC*18*F*4520 in asynchronous mode.

Communication will be at 115200bps baud rate using 8N1 protocol (8 data bits and 1 stop bit -you will not use 9th bit.). You will connect your development card to PC via serial port (RS232 9-pin) cable (Figure 1). You should use interrupt features in both transmit and receive. You may receive the messages at any time.

In this last part, different from other parts, your program should start in a `waiting state` such that it should wait until receiving a `$` character from PC without taking any interrupts from buttons, without making any A/D conversion and without responding any serial commands other than `$` character. In the `waiting state`, only `Waiting...` message should be seen on the first line of the LCD as shown in Figure 3. After getting first `$` character, your program should start the measurements, update the LCD accordingly, control the buttons as explained in **Part I**, and additionally serve the commands explained below until getting a `#` character. When your program receives a `#` character, it should switch to the `waiting state` and reset the set value to 00 and A/D conversion interval to 50ms [3].

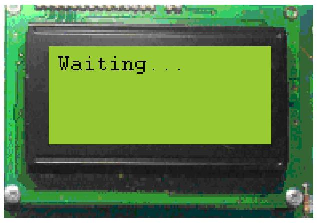

Figure 3: In the `waiting state`, only `Waiting...` message should be seen on the first line of the LCD.

There are four commands which your program should serve.

[3]Be aware of that the humidity value depends on the potentiometer position and that the state is determined after getting the current humidity value.

1. **Change set value** command: This command is used to change the set value to a new value given with the command. The command starts with `cvs` string followed by new set value and ends with `;` character.

   `cvs35;` is a valid command which changes the set value to 35.
   `cvs05;` is another valid command which changes the set value to 05.

   Note: Here 3, 0 and 5 are not hexadecimal values. They are ASCII numeric characters given with $0x33$, $0x30$ and $0x35$ hexadecimal values, respectively. Check the ASCII table in the course web site. **This convention is valid for all other numeric characters used in other commands.**

2. **Change measurement interval** command: This command is used to change the time interval according to which the A/D conversion is performed. The command starts with `cmi` string followed by new time interval value (in ms) and ends with `;` character.

   `cmi75;` is a valid command which changes the time interval to 75ms.
   `cmi150;` is another valid command which changes the time interval to 150ms.

   You can assume that new time interval value is greater than or equal to 50ms and does not exceed 5000ms.

3. **Get current humidity value** command: This command is used to get the current humidity value obtained from A/D conversion. The command is `geth;`. Your program should response to the command by immediately sending the current humidity value.

   If the current value of humidity is 45, then you program should send `45` string immediately after taking `geth;` command. Note that, your program should send two hexadecimal values representing each digit of the current humidity value.

   As another example, if the current value of humidity is 07, then your program should send `07` string not $0x07$ hexadecimal value.

4. **Get current set value** command: This command is used to get the current set value. The command is `gets;`. Your program should response to the command by immediately sending the current set value.

   If the current set value is 09, then your program should send `09` string (not $0x09$ hexadecimal value) immediately after taking `gets;` command.

Your programs will be tested sendind both valid and invalid commands. **In addition to the normal tests, a performance test will be performed to determine the real time performance of your program**.

3 Hints

- You are free to use any driver available for *PICos*18 or *MCC*18 library functions.

- For serial communication you can use *HyperTerminal* in Windows. You can access it from *Accessories* → *Communications* → *HyperTerminal*. There is also another program prepared for *Homework 4* last year. It was specialized for the homework, but it can help you. You may download it from below link:

  http://www.ceng.metu.edu.tr/courses/ceng336/2005/_labs/Lab4TestProgram.rar

- To adjust the configuration settings you may put the below code segment prior to any `#define` or `include` directives in your main file.

  ```
  #pragma config OSC = HS, FCMEN = OFF, IESO = OFF, PWRT = OFF, BOREN = OFF, WDT
  = OFF, MCLRE = ON, LPT1OSC = OFF, PBADEN = OFF, LVP = OFF, XINST = OFF, DEBUG
  = OFF
  ```

- We recommend to use 20MHz oscillator.

4 Evaluation

If your code is not compiled or you do not use interrupt features of the PIC while controlling the buttons, you get zero from this part of the term project and will not be able to object your score.

You will **certainly** use PICos18 and MCC18 compiler.

Total of the term project is worth 100 points.

- Phase 1: 30 points.
- Part I of Phase 2: 30 points.
- Part II of Phase 2: 40 points.

5 Hand In Instructions

- You will submit a single file for each part of this phase (`phase2-1.tar.gz` for Part I and `phase2-2.tar.gz` for Part II) including all of `.c` and `.h` files in your project folders and the kernel files if you make any changes. We will create two different homeworks for each part of Phase 2.
- Make sure that you have included your names and group ID in a comment at the top of `main.c` file.
- You will submit your file(s) using `http://cow.ceng.metu.edu.tr`.
- **No late submission is allowed.**
- For any clarification please follow the newsgroup.