Department of Computer Engineering
Digital Hardware Systems
CpE 3201 - Embedded Systems

Practical Activity # 5
Analog to Digital Converter

Exercise Objectives:
1. Configure Analog to Digital (A/D) Converter of the PIC16F877A
2. Use A/D converter module to convert analog input signal
3. Applying A/D converter in basic embedded systems application

Student Outcomes:
At the end of the exercise, students must be able to:
1. learn how to configure the Analog to Digital (A/D) Converter module
2. able to use A/D converter in converting analog signal to digital values
3. develop an embedded systems application utilizing A/D conversion

Tools Required:
The following will be provided for the students:
1. MPLAB IDE v8.92 + MPLAB XC8 v1.33
2. Proteus v8.11

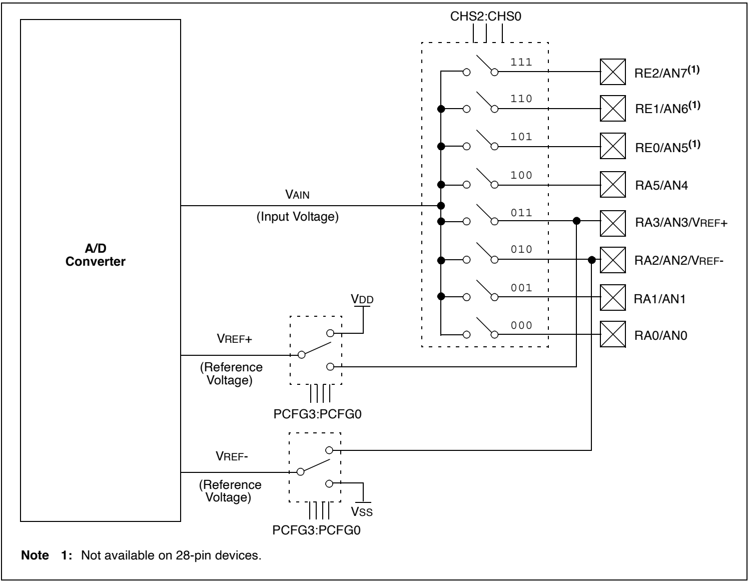
Delivery Process:
The instructor will conduct a briefing for this practical activity. Notes from the lecture class will provided for reference purposes. All inquiries pertaining to the latter must be referred to this manual.

Note:
Images and other contents in this manual are copyright protected. Use of these without consent from the respective authors is unauthorized.

Part I. Analog to Digital Converter Module of PIC16F87&A

The 8-/10-bit A/D converter module of the PIC16F877A has 8-analog input channels. It has a 8 or 10-bit digital output.The high and low-voltage reference input that is software selectable to some combination of VDD, VSS, RA2 or RA3. A unique feature of the A/D converter module is that it is able to operate while the device is in Sleep mode.

FIGURE 11-1: A/D BLOCK DIAGRAM



A/D Module Registers

The following are the registers associated with the module:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register0 (ADCON0)
- A/D Control Register1 (ADCON1)

ADRESH:ADRESL registers contain the 10-bit result of the A/D conversion. When the A/D conversion is complete, the result is loaded into this A/D Result register pair, the GO/DONE bit (ADCON0<2>) is cleared and the A/D interrupt flag bit ADIF is set. Interrupt enable ADIE is at PIE1 register and interrupt flag ADIF is at PIR1 register.

Setting Up A/D Conversion

The following steps must be followed in order to implement A/D conversion in PIC16F877A. Enabling the A/D conversion interrupt is optional.

1. Configure the A/D module:
    - Configure analog pins/voltage reference and digital I/O (ADCON1)
    - Select A/D input channel (ADCON0)
    - SelectA/D conversion clock (ADCON0)
    - Turn on A/D module (ADCON0)
2. Configure A/D interrupt (if desired):
    - Clear ADIF bit
    - Set ADIE bit
    - Set PEIE bit
    - Set GIE bit
3. Wait the required acquisition time.
4. Start conversion:
5. Set GO/DONE bit (ADCON0)
6. Wait for A/D conversion to complete by either:
7. Polling for the GO/DONE bit to be cleared (interrupts disabled); OR
8. Waiting for the A/D interrupt
9. Read A/D Result register pair (ADRESH:ADRESL), clear bit ADIF if required.
10. For the next conversion, go to step 1 or step 2 as required. The A/D conversion time per bit is defined as T_{AD}.

A/D Converter Resolution

The digital result is 10-bits at ADRESH:ADRESL. Depending on the A/D Result Format Select (ADFM) bit in the ADCON1 register, the 10-bit data can either be right or left justified.

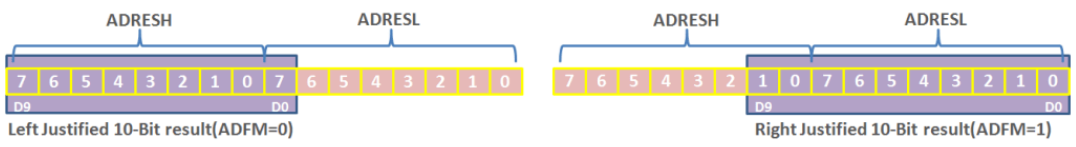


Fig. 1. Right and Left justify of the ADRESH:ADRESL

Let’s say that the VREF is 5V. Converting the analog input using the 10-bit resolution will have a step voltage of:

$$Step\ Voltage = \frac{V_{REF}}{2^{resolution}}$$

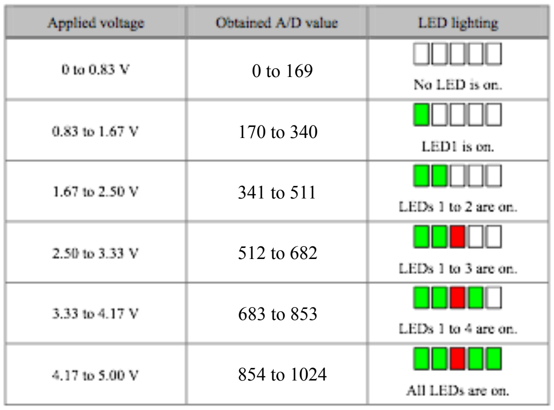
$$Step\ Voltage = \frac{5V}{2^{10}} = 4.882mV$$

Therefore, the digital value range of 0-169 will have an analog value range of:

$$169\ x\ 4.882mV = 0.825V$$
$$0V\ -\ 0.825V$$

A/D Converter Exercise

A variable resistor is connected to AN0 (RA0) with a reference voltage of +5V and five LEDs connected to PORTB <RB4:RB0>. The LEDs shall light up depending on the applied analog signal.



Applied voltage	Obtained A/D value	LED lighting
0 to 0.83 V	0 to 169	⬜⬜⬜⬜⬜ No LED is on.
0.83 to 1.67 V	170 to 340	🟩⬜⬜⬜⬜ LED1 is on.
1.67 to 2.50 V	341 to 511	🟩🟩⬜⬜⬜ LEDs 1 to 2 are on.
2.50 to 3.33 V	512 to 682	🟩🟩🟥⬜⬜ LEDs 1 to 3 are on.
3.33 to 4.17 V	683 to 853	🟩🟩🟥🟩⬜ LEDs 1 to 4 are on.
4.17 to 5.00 V	854 to 1024	🟩🟩🟥🟩🟩 All LEDs are on.

Fig. 2. LED output for range of analog input voltage.

The output of the LEDs are shown in Figure 2. Since VREF is 5V and the resolution is 10 bits, the digital value range can be obtained. Below is the `main()` with the A/D converter configuration and the interrupt service routine for A/D conversion "done" interrupt.

```
void main(void)
{
      int d_value = 0;

      TRISB = 0x00;     // set all PORTB as output
      PORTB = 0x00;     // all LEDs are off
      ADCON1 = 0x80;    // result: right Justified, clock: FOSC/2
                        // all ports in PORTA are analog
                        // VREF+: VDD, VREF-: VSS
      ADCON0 = 0x01;    // clock: FOSC/2, analog channel: AN0,
                        // A/D conversion: STOP, A/D module: ON

      for(;;)           // foreground routine
      {
            d_value = readADC();   // get ADC value

            /* setting the LEDs */
            if(d_value>=0 && d_value<=170)
                  PORTB = 0x00;         // all LEDs OFF
            else if(d_value>=171 && d_value<=341)
                  PORTB = 0x01;         // RB5 LED ON
            …
            …

      }
}
```

For this example the step voltage is 4.88mV. Therefore from 0 to 169, the voltage range is from 0V to 0.83V (see Figure 2).

```
void readADC(void)
{
      int temp = 0;
      delay(1000);            // delay to get the hold capacitor charged

      GO = 1;                 // start conversion

      while(GO_DONE==1);      // wait for conversion to finish

      /* read result register */
      temp = ADRESH;          // read ADRESH
      temp = temp << 8;       // move to correct position
      temp = temp | ADRESL;   // read ADRESL

      return temp;
}

void delay(int int)
{
      while(cnt--);
}
```

GO bit is automatically reset when conversion is done (GO_DONE='0'). Therefore, it has to be set to '1' again when starting a new conversion.

Open Proteus and construct the required circuit with filename "LE5-1.pdsprj". Use the *POT-HG* for the variable resistor and *LED-BARGRAPH-RED* for the LEDs. Connect a voltmeter to the analog input pin AN0 (RA0) to monitor the voltage. In MPLAB, create a new project using the project wizard with the name "CpE 3201-LE5". Create a new source file with the filename "LE5-1.c" and add it to the project. Write the source code with the pre-processor directives. Complete the interrupt handler for the A/D converter interrupt. Build the program and debug if necessary. After successfully building the source code, simulate the program in Proteus ("LE5-1.pdsprj") by loading the generated .hex file to the microcontroller. Observe the LEDs by adjusting the variable resistor. Verify the input voltage against the digital value range and the corresponding LEDs.

Part IV. Hands-on Exercise

Open the firmware project file "LE5-1.pdsprj" and save it as "LE5-2.pdsprj". Remove the LED bar graph and connect two (2) 7-segment displays at PORTB with BCD to 7-segment decoders. The lower nibble of PORTB will be the decimal while the upper nibble will the whole number. Tie the decimal point of the whole number to always on. Reference voltage for the A/D converter is 5V.

In MPLAB, create a new source file with the filename "LE5-2.c" and add it to the project "CpE 3201-LE5". Write a program that will display the voltage input with a single decimal digit. For example if the input voltage is 1.231V, then the display on the 7-segment will be "1.2". The display will update in real-time in response to the changes in the input voltage. Use interrupts in reading and updating the display.

Build the program and debug if necessary. After successfully building the source code, simulate the program in Proteus ("LE5-2.pdsprj") by loading the generated .hex file to the microcontroller. Observe the display and compare to voltage in the voltmeter. They should have the same whole number and decimal. Gather 10 input voltage samples from 0V to 5V and record the displayed voltage.

Instructions for submission:

- Write a report that contains the data gathered. The filename of the document should be "<LAST NAME>_LE5.pdf".
- Submit the report together with the following:
    - LE5-2.pdsprj
    - LE5-2.c
- Submit the files in Canvas in the correct order and file format.

Assessment

Criteria	Outstanding (4 pts)	Competent (3 pts)	Marginal (2 pts)	Not Acceptable (1 pt)	None (0 pts)
Feature Configuration	Configuration was properly done.	Configuration has minor errors.	-	Configuration is incorrect.	No project created.
Functionality	The systems function perfectly.	The system functions with minor issues.	The system has several issues which already affect the functionality.	The presented system is non-functioning.	No project created.
Firmware	Firmware created successfully without issues and followed the requirements.	Firmware created successfully with some issues and followed the requirements.	Firmware has issues and missing some of the requirements.	Firmware has several issues and did not follow the requirements.	No project created.

Copyright Information

Author: Van B. Patiluna (vbpatiluna@usc.edu.ph)
Contributors: none
Date of Release: March 18, 2021
Version: 1.0.3

References

PIC16F87X Data Sheet, Microchip Technology Inc. 2003.
CpE 3201 Lecture Notes on A/D Converter in PIC16F877A.

Change Log:

Date	Version	Author	Changes
March 15, 2021	1.0	Van B. Patiluna	- Initial draft.
March 16, 2021	1.0.1	Van B. Patiluna	- Revised the source code. - Added component details for the variable resistor and LEDs.
March 18, 2021	1.0.3	Van B. Patiluna	- Corrected some typographical errors. - Required to use interrupts in LE 5-2. - Factual erros corrected.