LABORATORY 14:
ANALOG TO DIGITAL CONVERSION USING ADC0809

NAME:                                STUDENT ID#:  

Objectives
- To understand how the ADC0809 works for analog-to-digital signal conversion.
- To program loop tests for analog-to-digital signal conversion.

Part 1: Understanding how the ADC0809 works

The complement to the digital-to-analog (D/A) converter is the **analog-to-digital (A/D) converter**. An A/D converter converts analog voltages to digital information that can be used by a computer. It is useful to note that the digital data produced by an A/D converter is only approximately proportional to the analog input. This is because a perfect conversion is impossible due to the fact that digital information changes in steps, whereas analog is virtually continuous.

A good example of the limitation is the 8-bit A/D converter that provides 8-bit numbers ranging from 0 to 255 to represent 256 different analog values. The situation improves as the number of bits increases. A 20-bit converter gets a lot closer to real world with 1,048,575 steps. However, it's important to remember that there are infinitely more analog values possible between a single step of any A/D converter.

One important element in analog to digital converters such as the ADC0809 is the analog comparator. The analog comparator accepts analog inputs but produces a digital output. Another important component in the ADC0809 is a digital to analog converter (DAC) similar to the one discussed in LAB13. The digital representation of the input can be determined by connecting the analog voltage input to be converted to the - input of the comparator, and the output of the DAC to the + input. Figure L14.1 shows the block diagram of ADC0809.
The DAC is initialized by storing only the high-order bit (bit 7 of the 8-bit converter) in it. If the comparator output is low indicating the test analog level from the DAC is still lower than the input, the bit is kept. If the comparator output is high indicating the test analog level from the DAC is too high, then the bit is dropped. The other bits are tested in the same manner and either kept or dropped according to the status of the comparator. The process is called successive approximation is illustrated in Figure L14.2.
Now consider the task of providing a digital number for an analog input of 3.21 volts. It will be assumed that the D/A converter provides a voltage and that the comparator compares voltages. The flow chart says to start with bit 7 which has a weight of 128, then keep or drop bits according to the output of the comparator. The process used to convert 3.21 volts to a number is shown in the Table L14.1:

<table>
<thead>
<tr>
<th>Test Bit</th>
<th>DAC Binary Value</th>
<th>Decimal</th>
<th>((5 \times \text{DAC value})/256)</th>
<th>Comparison Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>10000000</td>
<td>10000000</td>
<td>128</td>
<td>2.5</td>
<td>low - keep bit</td>
</tr>
<tr>
<td>01000000</td>
<td>11000000</td>
<td>192</td>
<td>3.75</td>
<td>high - drop bit</td>
</tr>
<tr>
<td>00100000</td>
<td>10100000</td>
<td>160</td>
<td>3.125</td>
<td>low - keep bit</td>
</tr>
<tr>
<td>00010000</td>
<td>10110000</td>
<td>176</td>
<td>3.4375</td>
<td>high - drop bit</td>
</tr>
<tr>
<td>00001000</td>
<td>10101000</td>
<td>168</td>
<td>3.28125</td>
<td>high - drop bit</td>
</tr>
<tr>
<td>00000100</td>
<td>10100100</td>
<td>164</td>
<td>3.203125</td>
<td>low - keep bit</td>
</tr>
<tr>
<td>00000010</td>
<td>10100110</td>
<td>166</td>
<td>3.2421875</td>
<td>high - drop bit</td>
</tr>
<tr>
<td>00000001</td>
<td>10100101</td>
<td>165</td>
<td>3.22265625</td>
<td>high - drop bit</td>
</tr>
</tbody>
</table>

Three bits were kept, leaving 10100100 in the DAC converter. The analog to digital converter will provide 164 as its numerical representation for 3.21 volts.

Figure L14.3 shows the ADC0809 decoder circuit. With this information we can design assembly instructions to test the ADC0809.

![Figure L14.3 The ADC0809 decoder circuit](attachment:image)
Part 2: Programming the ADC for an Analog Input

Here we will begin our practice of converting analog voltage by ADC0809 with a simple program. We will analyze, run, and modify the program. Check off each step as it is completed.

<table>
<thead>
<tr>
<th>Check</th>
<th>Step</th>
<th>Procedure</th>
</tr>
</thead>
</table>
| _____ | 1.   | Examine the data segment and constant definitions in the program L14P1.ASM.  
      |      | a. Make sure DIP SW3 switched to INT4-INT7.  
      |      | b. Wire CON1 to POWER SUPPLY (+).  
      |      | c. Wire common ground.  
      |      | d. Describe the reason why we need to do step1.a to step1.c for this lab? |
| _____ | 2.   | Examine the part of the program in L14P1.ASM. If the address of ADC0809 INT0 is 3FCAH. (Hint: Refer to Figure L14.3),  
      |      | a. What is the address of ADC0809 INT1? _______________________________  
      |      | b. What is the address of ADC0809 INT2? _______________________________  
      |      | c. What is the address of ADC0809 INT3? _______________________________  
      |      | d. What is the address of ADC0809 INT4? _______________________________ |
| _____ | 3.   | Examine the following part of the program in L14P1.ASM.  
      |      | MOV   DX, OFFSET MSG  
      |      | MOV   AH, 9  
      |      | INT   21H  
      |      | J1:   MOV   AX, 0  
      |      | MOV   DX, ADC1  
      |      | OUT   DX, AL  
      |      | MOV   CX, 20H  
      |      | LOOP   $  
      |      | IN     AL, DX  
      |      | MOV   DX, BPORT3  
      |      | OUT   DX, AL  

【LAB14-4】
MOV    DH, 4
INT    10H

MOV    CX, 0
LOOP    $  

MOV    DX, OFFSET BACK
MOV    AH, 9
INT    21H

JMP    J1

a. What is the purpose of the instructions ‘MOV AH, 9’ and ‘INT 21H’?

b. What is the purpose of the instructions ‘MOV DH, 4’ and ‘INT 10H’?

c. What is the purpose of the following instruction sequence?

MOV    AX, 0
MOV    DX, ADC1
OUT    DX, AL

d. What is the purpose of the following instruction sequence?

MOV    CX, 20H
LOOP    $

4. Swing POWER SUPPLY (+) from +5V to 0V, and describe what you see. (Notice: don’t exceed +5.3V)
5. Modify the program L14P1.ASM to L14P1A.ASM to convert your voice from microphone and show the converted digital number on your screen. Reassemble, relink, and generate L14P1A.EXE. Use the DEBUG program and describe what you see. Download the new program to MTS-86C and verify the program that it works correctly. (Hint: INT3 is used for microphone). Hand in your program listing with this lab report. Describe briefly what you have done in modifying L14P1.ASM to L14P1A.ASM.