Lab #11

*Interface to a Hexadecimal Keypad*

**PURPOSE**

This lab introduces interfacing a keypad to a 68000 microprocessor through a 6821 peripheral interface adapter (PIA). Upon completion of this lab, students will be able to do the following:

- Write program to perform input/output operations using a 6821 PIA.
- Write programs that scan rows and columns of a keypad to determine if a key is pressed.
- Write programs to perform software debouncing.

**PREPARATION**

Prior to the scheduled lab session, read the following sections from your textbook:
- Section 9.11 (6821 Peripheral Interface Adapter)
- Section 9.12 (Hexadecimal Keypad Interface)

**MATERIALS**

*Hardware:*
- 68KMB 68000-based computer
- I/O Board #5 for the 68KMB
- I/O Board #2 for the 68KMB
- PC host computer
- RS232C serial interface cable
- 20-conductor ribbon cable
**MS-DOS Software:**
- A68K 68000 cross assembler
- XLINK 68000 linker, locator, conversion utility
- PC-VT VT100 terminal emulator
- EDIT MS-DOS text editor (or equivalent)

**68000 Programs:**
- KEYPAD provided in 68KMB directory
- KEYPAD2 to be written

**INTRODUCTION**

This is the fifth in a series of labs to explore interfacing with the 68KMB. I/O Board #5 is quite different from the boards used in the preceding labs. Instead of interfacing to the 68681 on the 68KMB, I/O Board #5 contains its own peripheral interface IC – a 6821 peripheral interface adapter (PIA). The connection to the 68KMB is through J2. J2 includes the required address decoding and control signals to interface to any 8-bit peripheral interface device from the 6800-family. The signals on J2 of the 68KMB are illustrated in Figure 8-16 in *The 68000 Microprocessor*.

The connection between I/O Board #5 and connector J2 of the 68KMB is illustrated in Figure 11-1.

![Figure 11-1. Connecting I/O Board #5 to J2 on the 68KMB](image-url)
The hexadecimal keypad has 16 pressure-sensitive switches arranged in four rows and four columns. These connect to Port A on the 6821 PIA. The complete interface is shown in Figure 11-2.

Figure 11-2. I/O Board #5

The 6821 PIA and the keypad interface are discussed in Section 9.11 and Section 9.12 of your textbook. Review this material before proceeding.

PROCEDURE

1. With the 68KMB powered-off, connect I/O Board #5 to J2.
2. Power-on the 68KMB and the PC host computer. Execute PC-VT and obtain the MON68K prompt from the 68KMB.

3. An example program called KEYPAD is presented in Example 9-12 in your textbook. KEYPAD scans the rows and columns of the keypad to determine if a key is pressed. The program includes software debouncing to ensure the mechanical contacts in the keypad have stabilized. When a clean key closure is detected, the ASCII code for the key is sent to the console. Review the software listing and the description of the program to gain an understanding of its operation.

The program is in the directory 68KMB on the PC host computer. Run the program and demonstrate it to your lab instructor. Be prepared to answer questions on the operation of this program.

4. Power-off the 68KMB. For the next part of this lab, we will use two I/O Boards – I/O Board #5 and I/O Board #2. Connect I/O Board #2 to J1 on the 68KMB. Power-on the 68KMB. Execute PC-VT and obtain the MON68K prompt from the 68KMB.

5. Make a copy of KEYPAD.SRC and save it in a file called KEYPAD2.SRC. Put your name and the date in comment lines at the top. Modify the program such that the output is sent to the 7-segment display on I/O Board #2 (rather than to the console).

Run the new program and demonstrate it to your lab instructor.

CONCLUSION

This lab has introduced interfacing to a hexadecimal keypad, including scanning the rows and columns of the keypad and debouncing mechanical switches through software.
Lab #12

Interface to a Digital-to-Analog Converter

PURPOSE

In this lab a digital-to-analog converter (DAC) is interfaced to a 68000 microprocessor through a 6821 peripheral interface adapter. Upon completion of this lab, students will be able to do the following:

- Use monitor commands to read and write registers inside a 6821 PIA.
- Use monitor commands to control a DAC connected to a 6821 PIA.
- Calibrate the output voltage range of a DAC.
- Write programs to create waveforms at the output of a DAC.
- Write programs to create musical tones using a loudspeaker driven by a DAC.

PREPARATION

Prior to the scheduled lab session, read the following sections from your textbook:
- Section 9.13 (Analog Output)
- Section 9.14 (Digital Sine Wave Generator)
- Section 9.15 (Music Output From a Digital-to-Analog Converter)
MATERIALS

Hardware:
• 68KMB 68000-based computer
• I/O Board #6
• power supply (±12 volt or ±15 volt)
• oscilloscope
• screwdriver
• PC host computer
• RS232C serial interface cable
• 20-conductor ribbon cable
• amplified external speaker (optional)

MS-DOS Software:
• PC-VT VT100 terminal emulator
• A68K 68000 cross assembler
• XLINK 68000 linker, locator, conversion utility

68000 Programs:
• SAWTOOTH provided in 68KMB directory
• TRIANGLE to be written
• SINEWAVE provided in 68KMB directory
• AMAJOR provided in 68KMB directory

INTRODUCTION

This is the sixth in a series of labs to explore interfacing with the 68KMB. I/O Board #6 contains a single 6821 PIA and two separate interfaces. Port A of the PIA interfaces to an 8-bit digital-to-analog converter (DAC), and Port B of the PIA interfaces to an 8-bit analog-to-digital converter (ADC). The DAC is an MC1408L8 device, and the ADC is an ADC0804 device. This lab is concerned with the DAC interface.

The complete schematic of I/O Board #6 is spread over three pages. These appear in Figure 12-1, at the end of this lab. Note that I/O Board #6 requires an external ±12 volt or ±15 volt power supply.

The DAC interface is discussed in Section 9.13 of your textbook. Review this before proceeding.
PROCEDURE

1. With the 68KMB powered off, connect I/O Board #6 to J2.

2. With the external power supply switched off, connect the power supply cable to I/O Board #6.

   Have your lab instructor verify the connections, before turning the power on.

3. Power-on all equipment and execute PC-VT on the PC host computer. The monitor prompt should appear on the console.

4. Calibrate the MC1408L8 as follows:

   (a) Reset the 68KMB.

   (b) Configure Port A of the 6821 as an output port by writing $FF to Data Direction Register A (DDRA) at address $010001. (Hint: Use MON68K's memory modify command.)

   (c) Enable access to Port A of the 6821 by writing $04 to Control Register A (CRA) at address $010003.

   (d) Write $FF to Port A at address $010001. (This places the byte $FF at the input to the MC1408L8 DAC, generating the full-scale output voltage at Test Point 1, TP1 (see Figure 12-1a).

   (e) Adjust trimpot R1 while observing the voltage at TP1. Calibrate for a full-scale output of 10 volts.
5. Write the following values to the DAC and measure the analog result with an oscilloscope. Tabulate the results:

$00 __________

$7F __________

$80 __________

$B0 __________

$FF __________

Demonstrate to your lab instructor that you can configure the 6821 PIA using MON68K's memory modify command and manually write data to the MC1408L8 DAC.

6. A program called SAWTOOTH is presented in Example 9-13 in your textbook. The program creates a sawtooth waveform at TP1 by continually sending a count to the DAC. Review the software listing and the description of the program to gain an understanding of its operation.

The program is in the directory 68KMB on the PC host computer. Run the program and demonstrate it to your lab instructor.

What is the frequency of the sawtooth waveform that you observe on the oscilloscope? ________________

What is the duration of each output step? ________________

7. Use MON68K's DI command to disassemble the program. Compare the result with the listing in your textbook.

At what address within the program is the variable COUNT located? ________________
8. Using MON68K’s memory modify command, change the variable COUNT to $000A. Re-run the program.

What is the frequency of the sawtooth waveform? 

With COUNT = $000A, what is the duration of each output step? 

9. Leave COUNT = $000A, but change the variable STEP to $0040.

What is the frequency of the output waveform? 

What is the duration of each output step? 

10. Make a copy of SAWTOOTH.SRC and call it TRIANGLE.SRC. Put your name and the date in comment lines at the top. Modify the new program to create a triangle waveform instead of a sawtooth waveform. Demonstrate the new program to your lab instructor.

11. A program called SINEWAVE is presented in Example 9-14 in your textbook. The program creates a sine wave at TP1 by continually outputting data read from a look-up table. Review the software listing and the description of the program to gain an understanding of its operation.

The program is in the directory 68KMB on the PC host computer. Run the program. Connect channel A of the oscilloscope to TP1 and channel B to TP2. The circuit between TP1 and TP2 is a 4 kHz low-pass filter (see Figure 12-1a). Demonstrate the program to your lab instructor.
What is the frequency of the sine wave that you observe on the oscilloscope?

12. Using MON68K's memory modify command, change the variable STEP to $0020. Re-run the program.

What is the frequency of the sine wave?

13. Change the variable STEP to $0080 and re-run the program.

What is the frequency of the sine wave?

14. The DAC circuit includes additional output stages to create audio tones for music or speech output. Section 9.15 in your textbook describes the circuit and presents a technique to create musical tones. Review this section before proceeding.

15. A program called AMAJOR is presented in Example 9-15 in your textbook. The program uses the console keyboard to create musical tones on the loudspeaker on I/O Board #6. The volume is controlled by potentiometer R2. If an amplified external speaker is available, connect it to J1 (on I/O Board #6).

Review the software listing and the description of the program to gain an understanding of its operation.

The program is in the directory 68KMB on the PC host computer. Connect channel A of the oscilloscope to TP1 and channel B to TP2. Run the program and demonstrate your musical skill to your lab instructor.
CONCLUSION

This lab has introduced interfacing to a digital-to-analog converter using a 6821 PIA and an MC1408L8 8-bit DAC.
Figure 12-1. I/O Board #6 (a) CPU Interface (b) DAC output (c) ADC input
Figure 12-1. (continued)
Figure 12-1. (continued)
Lab #13

Interface to an Analog-to-Digital Converter

PURPOSE

In this lab an analog-to-digital converter (ADC) is interfaced to a 68000 microprocessor through a 6821 peripheral interface adapter. Upon completion of this lab, students will be able to do the following:

• Write programs to interface to an ADC through a 6821 PIA.
• Write programs to input speech samples through an ADC and store them in a buffer.
• Write a program to output the content of a buffer to an ADC.

PREPARATION

Prior to the scheduled lab session, read the following sections from your textbook:
• Section 9.16 (Analog Input)
• Section 9.17 (Digitized Speech Input and Playback)
MATERIALS

*Hardware:*
- 68KMB 68000-based computer
- I/O Board #6
- I/O Board #3
- microphone (200 Ω)
- power supply (±12 volt or ±15 volt)
- screwdriver
- PC host computer
- RS232C serial interface cable
- two 20-conductor ribbon cable

*MS-DOS Software:*
- PC-VT VT100 terminal emulator

*68000 Programs:*
- ADCTEST provided in the 68KMB directory
- ADCTEST2 to be written
- SPEECH provided in the 68KMB directory
- SPEECH2 to be written

INTRODUCTION

This lab is a continuation of lab #12. Our main focus is on the analog input channel on I/O Board #6. An ADC0804 8-bit analog-to-digital converter connects to Port B of the 6821 PIA, as shown in Figure 12-1 in lab #12. The interface also includes three op amps and a sample-and-hold circuit to allow input from a microphone.

The ADC interface is discussed in Sections 9.16 and 9.17 of your textbook. Review these before proceeding.

PROCEDURE

1. With the 68KMB powered off, connect I/O Board #6 to J2.

2. With the external power supply switched off, connect the power supply cable to I/O Board #6.
Have your lab instructor verify the connections, before turning the power on.

3. Notice in Figure 12-1c (lab #12) that jumpers X1, X2, and X3 select between three possible inputs to the ADC. If X1 is installed, the ADC’s input is from the microphone and the conditioning circuitry. If X2 is installed, the ADC’s input is from the 10K trimpot labeled R3 in Figure 12-1c. If X3 is installed, input is from an external transducer connected to J5 on I/O Board #6.

Install a jumper in X2. (Remove the jumper from X1, if it is present.)

4. An example program called ADCTEST is presented in Example 9-16 in your textbook. The program continually reads the ADC0804 and reports the result of the conversion on the console. Review the software listing and the description of the program to gain an understanding of its operation.

The program is in the directory 68KMB on the PC host computer. Run the program and connect an oscilloscope or voltmeter to test point #6 (TP6). Demonstrate the program to your lab instructor.

Adjust the trimpot R3 to the following voltages, as measured at TP6. For each voltage, what is the converted result read from the ADC0804?

<table>
<thead>
<tr>
<th>Voltage</th>
<th>Converted Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 volts</td>
<td></td>
</tr>
<tr>
<td>1 volt</td>
<td></td>
</tr>
<tr>
<td>2 volts</td>
<td></td>
</tr>
<tr>
<td>3 volts</td>
<td></td>
</tr>
<tr>
<td>4 volts</td>
<td></td>
</tr>
<tr>
<td>5 volts</td>
<td></td>
</tr>
</tbody>
</table>
5. Power-off the 68KMB by removing the AC adapter jack and by switching off the external power supply connected to I/O Board #6.

For the next part of this lab, we will add I/O Board #3 – the 4-digit LED display. Connect I/O Board #3 to J1 on the 68KMB. Power-on the 68KMB and switch on the external power supply connected to I/O Board #6. Execute PC-VT and obtain the MON68K prompt.

6. Make a copy of ADCTEST.SRC and call it ADCTEST2.SRC. Put your name and the date in comment lines at the top. Modify the new program such that the output is sent to the 4-digit display on I/O Board #3. The display should vary from 0 to 255 as the input varies from 0 volts to 5 volts. Use subroutines from COUNT4.SRC in lab #9, as appropriate.

Hint: Begin by writing a subroutine that converts a hexadecimal byte to three BCD digits. This can be done by first dividing the byte by 100 to obtain the hundreds digit, and then dividing the remainder by 10 to obtain the tens digit. The final remainder is the ones digit.

Demonstrate the new program to your lab instructor.

7. Move the jumper from position X2 to position X1 on I/O Board #6. With jumper X1 installed, the trimpot is disconnected from the ADC0804. Input is now obtained through the microphone and its associated conditioning circuitry.

8. A program called SPEECH is presented in Example 9-17 in your textbook. The program has two parts. When started at address $008000, speech samples are gathered from the ADC0804 and placed in a buffer. When started at address $00800C, the samples in the buffer are sent to the DAC output channel on I/O
Board #6. Review the software listing and the description of the program to gain an understanding of its operation.

The program is in the directory 68KMB on the PC host computer. Connect the microphone provided to connector J2 on I/O Board #6. Run the program and demonstrate it to your lab instructor.

Notes:
1. Adjust trimpot R2 to vary the volume of the output.
2. Install an amplified external speaker in the auxiliary connector (J1) to improve the quality of audio output.
3. If the speech output is very noisy, follow the calibrate/checkout procedure for I/O Board #6 given in Appendix A of this lab manual.

9. Make a copy of SPEECH.SRC and save it in a file called SPEECH2.SRC. Place your name and the date in comment lines at the top. Modify the new program as follows. Place the playback routine in a loop such that the content of the RAM buffer is continually sent to the DAC. During playback, monitor the console for keyboard input and respond in the following way when a key is pressed:

   u the frequency of playback increases (UP)
   d the frequency of playback decreases (DOWN)
   SPACE the frequency of playback is restored to normal
   q quit to MON68K

Ignore any other keystrokes. (Hint: The frequency of playback can be controlled by altering the timer count.)

Demonstrate the modified program to your lab instructor.
CONCLUSIONS

This lab has demonstrated an interface between an analog-to-digital converter and the 68000 microprocessor through a 6821 PIA.
Lab #14

**Purpose**

This lab introduces a variety of concepts relevant to developing large assembly language programs.

Upon completion of this lab, students will be able to do the following:

- Define and give examples of the following terms: modular programming, relocatable module, absolute module, code segment, data segment, external symbol, public symbol, linker, and locator.
- Write a 68000 assembly language program consisting of multiple relocatable modules containing code and data segments.
- Assemble, link, and locate a 68000 program, creating a single absolute output module.

**Preparation**

Prior to the scheduled lab session, read the following section from Chapter 4 of your textbook:
- Section 4.7 (Modular Programming)

**Materials**

*Hardware:*
- 68KMB 68000-based computer
- PC host computer
- RS232C serial interface cable
**MS-DOS Software:**
- A68K 68000 cross assembler
- XLINK 68000 linker/locator
- EDIT MS-DOS editor (or equivalent)
- PC-VT VT100 terminal emulator

**68000 Programs:**
- ECHO provided in the 68KMB directory
- MYLIB provided in the 68KMB directory
- ECHO2 to be written
- MYLIB2 to be written

**INTRODUCTION**

A simple program is used in this lab to illustrate modular programming. The program is split across two files which must be assembled separately and then linked together to form a single absolute object module.

The program does the following:

1. Output the prompt "Enter a command: ".

2. Input a line from the keyboard. (Store the line in an input buffer. Echo each character as it is typed.)

3. When RETURN is entered, echo the entire input line (again).

4. Repeat.

When a line beginning with "q" or "Q" is entered, the program terminates to MON68K.

The main part of the program is in a file called ECHO.SRC and the subroutines are in a file called MYLIB.SRC.

Section 4.7 in your textbook contains a detailed discussion on modular programming using the above program as an example. Review this section before proceeding.
PROCEDURE

1. The files ECHO.SRC and MYLIB.SRC are in the directory 68KMB on the PC host computer. Assemble each of these.

Examine the listing files and answer the following questions:

What is the name of the code segment? ________________________________

What is the name of the data segment? ________________________________

What is the first address of the prompt string? (Note: this address is relative to the start of ECHO.OBJ.) ________________________________

What does the symbol CR stand for? ________________________________

What is the opcode for the RTS instruction? ________________________________

2. The file ECHO.XLK is a batch file for XLINK to combine ECHO.OBJ and MYLIB.OBJ into a single executable object program. ECHO.XLK is also located in the directory 68KMB on the PC host computer. Use the TYPE command to examine ECHO.XLK.

Do you understand the purpose of each line in ECHO.XLK? If not, review Section 4.7 in your textbook or ask your lab instructor for assistance.

3. Issue the appropriate command to link and locate ECHO.OBJ and MYLIB.OBJ. Review Section 4.7 of your text if you are not sure how to do this.

What output files were created? ________________________________

Examine the link map and answer the following questions:

What is the absolute address of the beginning of the prompt string? ________________________________

100
What is the first and last address of your program?  

What is the first and last address of the RAM buffer used in the program?  

What is the address of the OUTSTR subroutine?  

4. Execute PC-VT and transfer ECHO.HEX to the 68KMB. Verify that the program is in the 68KMB's memory.

What MON68K command did you enter?  

5. Run the program and demonstrate it to your lab instructor.

6. Now, we'll modify ECHO. First, make copies of ECHO.SRC, MYLIB.SRC, ECHO.XLK and save them in ECHO2.SRC, MYLIB2.SRC, and ECHO2.XLK respectively. Put your name and the date in comments lines at the top of each source file. Change the prompt to "Sharon's program, Enter a command: ". (Use your name, please.)

Modify the program to interpret the first character on each line as follows:

Q Quit to MON68K (this is already supported)

U Echo the entire line in uppercase characters. Convert lowercase characters to uppercase characters. Leave graphic characters as is.

L Echo the entire line in lowercase characters. Convert uppercase characters to lowercase characters. Leave punctuation characters as is.

F Echo the input line forwards

B Echo the input line backwards
Display a description of the commands supported

For any other command character, do not echo the line. Re-issue the prompt and repeat. Recognize commands in both uppercase and lowercase.

Continue to use a modular approach in the modified program. Put the code to determine the command in ECHO2. Put the code to execute commands in subroutines in MYLIB2. Remember to change the filenames in ECHO2.XLK as appropriate.

Demonstrate the new program to your lab instructor.

14.3

CONCLUSION

Having completed this lab, students are familiar with modular programming in 68000 assembly language.