# EXPERIMENENT #4

# CODE CONVERSION and BIT MANIPULATION

## 1.0 Purpose

The purpose of this experiment is to accomplish the following:

- perform ASCII, BCD and Hexadecimal Code Conversion
- gain familiarity with the 68000's bit manipulation instructions
- learn how to download programs from a host computer into the SANPER-1 ELU.

## 2.0 Component Requirements

None.

## 3.0 Background

### A. Bit Manipulation

Bit manipulation is the ability to modify each bit according to some algorithm. The 68000 has the following four Bit Manipulation Instructions:

- BCHG Test a Bit and Change
- BCLR Test a Bit and Clear
- BSET Test a Bit and Set
- BTST Test a Bit

Bit manipulation can also be performed with Logical Instructions such as:

- AND Logical AND
- ANDI Logical AND Immediate
- OR, Logical Inclusive OR
- ORI Logical Inclusive OR Immediate
- EOR Logical Exclusive OR
- EORI Logical Exclusive OR Immediate
- NOT Logical Complement

Lastly, bit manipulation can be performed with Shift and Rotate Instructions such as:

- ASL Arithmetic Shift Left
- ASR Arithmetic Shift Right
- LSL Logical Shift Left

- LSR Logical Shift Right
- ROL Rotate Left
- ROR Rotate Right
- ROXL Rotate Left with Extend
- ROXR Rotate Right with Extend

B. Downloading Capability

Through a combination of hardware and software, the SANPER-1 ELU is capable of receiving MC68000 programs from an external computer, and storing these programs into the SANPER-1 ELU's memory. This downloading capability is achieved in hardware by connecting the serial port of the computer to one of the serial ports of the SANPER-1 ELU. The download functionality is achieved in software through the TUTOR firmware. Invoking TUTOR's Transparent Mode Command ("TM") sets up the SANPER-1 hardware to wait for data to arrive through one of its serial ports. The external computer then transmits a file out of its serial port. The file is sent in Motorola S-Record format. The TUTOR firmware reads in the data from its serial ports and stores it into memory.

The procedure to download a program from a personal computer to the SANPER-1 ELU is described in the SANPER-1 Educational Lab Unit User's Manual.

## 4.0 Statement of Problem

In the first part of this experiment the student will download a sample program to the SAMPER-1 ELU and then execute it. This exercise will provide the student with practice in downloading programs to the SANPER-1 ELU.

In the second part of this experiment the student will download a bit manipulation program to the SANPER-1 ELU. This program will accept terminal input of a decimal number from 0 to 255, pass it through a bit manipulation algorithm, and output the new data to the terminal, and to the SANPER-1 ELU's User Data Display.

## 5.0 Preliminary Assignment

A. Sample Downloading Program

1. Create a file on the host computer and enter the program listed in Table 4.1. Some assembler-specific directives may need to be added to the program in order for it to be assembled by the host computer's MC68000 cross-assembler. A list of directives is available in the User's Manual for the specific assembler.

2. Assemble the program using the appropriate host computer commands. The assembled program should be free of syntax and assembler errors. The MC68000

cross-assembler will generate two output files: a LIST files and a TAPE file. The TAPE file consists of Motorola S-Records, and is the file which will be downloaded to the SANPER-1 ELU. The LIST file is a listing of the assembled source code. Bring two copies of each file to the lab.

3. Read Appendix A of the MC68000 Educational Computer Board User's Manual, entitled "S-Record Output Format" which discusses the Motorola S-Record data format. Closely examine the S-Records in your TAPE file. You should be able to identify each of the six fields within each record. Copy these records onto another sheet of paper, label each field and explain its function.

B. Bit Manipulation Program

4. Write a subroutine to input a decimal number (range: 0 to 255) from the terminal. Students are permitted to use TUTOR's TRAP 14 Handler for inputting data from the terminal. Make sure the TRAP 14 Handler's initialization requirements have been satisfied.

5. Write a subroutine to perform bit manipulation on the input data per the bit manipulation algorithm provided by the Course Instructor. A sample of the algorithm is shown in Figure 4.1.

   Note: The student is not permitted to use any of TUTOR's TRAP 14 conversion routines. All conversion routines must be written in 68000 assembly language.

6. Write a subroutine to output the manipulated value (in decimal) to the terminal. Students are permitted to use TUTOR's TRAP 14 Handler for outputting data to the terminal. Make sure the TRAP 14 Handler's initialization requirements have been satisfied.

7. Write a subroutine to output the manipulated value (in decimal) to the User Data Display on the Front Panel of the SANPER-1 Educational Lab Unit. Refer to the SANPER-1 Educational Lab Unit's Memory Map for the addresses of the User Data Display.

8. Write the main program, which calls all the subroutines written above. A block diagram of the entire system is shown in Figure 4.2.

9. Create a file on the computer and enter your Bit Manipulation program into this file.

10. Assemble the Bit Manipulation program using the appropriate host computer commands. The assembled program should be free of syntax and assembler errors.

## 6.0 Procedure

1. Download the sample program from Table 4.1 to the SANPER-1 ELU using the downloading procedure outlined in the SANPER-1 ELU User's Manual.
2. Execute the sample program.
3. Record the results of running this program.
4. Download your Bit Manipulation program to the lab unit.
5. Execute your program.
6. Enter the required test input data, and verify that your program executes correctly. If it doesn't execute correctly, use TUTOR's debugging capabilities (breakpoints and tracing) to locate the problem.
7. Demonstrate to your Lab Instructor that your program works correctly.

## 7.0 Discussion

Submit the following items to your Lab Instructor as a Final Report:

1. A listing of your program with both global and local comments.
2. A description of the S-Record fields for the sample program.
3. Draw a block diagram illustrating how you implemented your program. Include all conversion routines required for implementation of this program.

TABLE 4.1 Sample Downloading Program

```
            ORG $900
MSGSTR      DC.B "IT WORKS!"
MSGEND      DS.B 1

            ORG $1000
            LEA $2FFF,A7
LOOP        LEA MSGSTR,A5
            LEA MSGEND,A6
            MOVE.B #243,D7
            TRAP #14
            MOVE.B #241,D7
            TRAP #14
            MOVE.B #227,D7
            TRAP #14
            BRA LOO
```

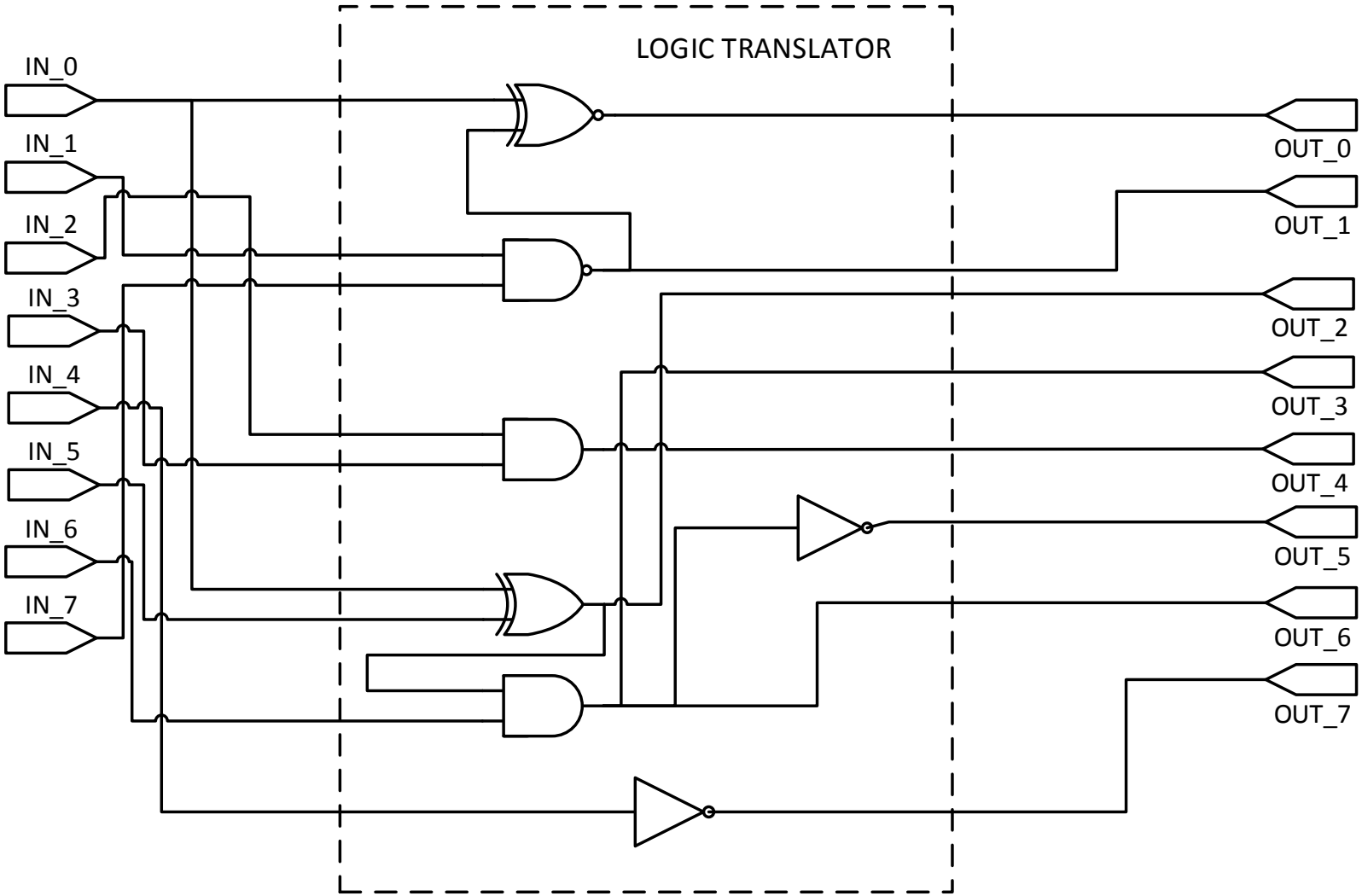
LOGIC TRANSLATOR
IN_0
IN_1
IN_2
IN_3
IN_4
IN_5
IN_6
IN_7
OUT_0
OUT_1
OUT_2
OUT_3
OUT_4
OUT_5
OUT_6
OUT_7

FIGURE 4.1

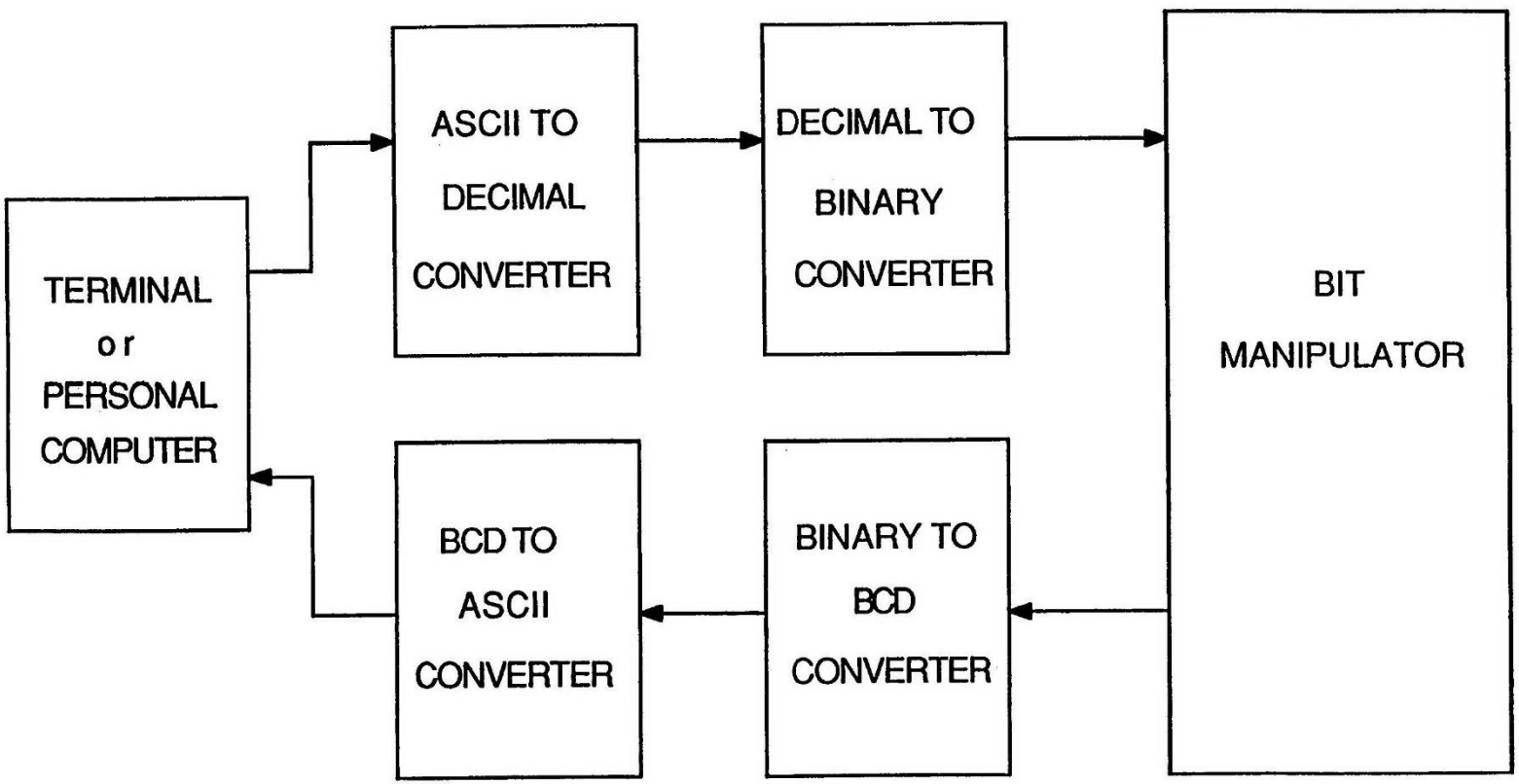


FIGURE 4.2 BIT MANIPULATION BLOCK DIAGRAM