Experiment No. 04:

Code Conversion and Bit Manipulation

By: \*\*\*\*\*\*\*\*\*\*

Lab Partner: \*\*\*\*\*\*\*\*\*

Instructor: Dr. Jafar Saniie

ECE 441-001

Lab Date: 02-14-2024

Due Date: 03-16-2024

Acknowledgment: I acknowledge all of the work (including figures and codes) belongs to me and/or persons who are referenced.

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**I. Introduction**

**A. Purpose**

The purpose of this experiment is to acquaint students with ASCII, BCD and Hexadecimal code conversion. In addition, students will gain familiarity with the 68000’s bit manipulation instructions and with downloading programs from a host program to the SANPER-1 ELU.

**B. Background**

Bit manipulation is the ability to modify each bit of a binary number according to an algorithm. The 68000 has four bit manipulation routines: BCHG, BCLR, BSET and BTST. Bit manipulation can also be accomplished with logical instructions such as, AND, OR and NOT. The third way bit manipulation can be performed is with the shift and rotate instructions, such as ASL, ASR, LSL, ROR, and ROL.

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

**II. Lab Procedure and Equipment List**

**A. Equipment**

*Equipment*

* SANPER-1 system
* PC with TUTOR software

**B. Procedure**

1. Download the sample program from Table 4.1 in the lab manual 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 the Bit Manipulation program from the preliminary exercise section to the lab unit.
5. Execute the 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

**III. Results and Analysis**

**\*** See attached text files for results

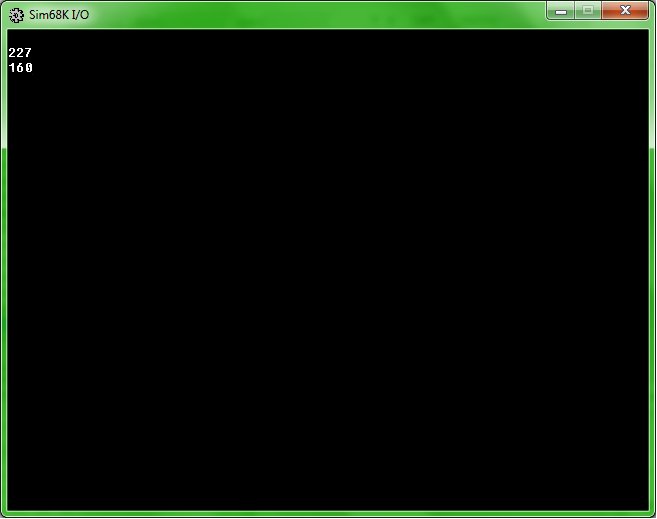


Figure 1: Results for Bit Manipulation Program using Sim68K

**A. Discussion**

1. A listing of your program with both global and local comments.

* *Original Program with comments attached*

1. A description of the S-Record fields for the sample program.

S Records:

S748392174983274983274972398473298479237498237948723984723984739284798237984ecnc4ej23hd23ud238d2389jd823j988dj239

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type (2) | Record Length (2) | Address (4,6,8) | Code/Data (0-2n) | Checksum (2) |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Descriptions:

S0:???

S1: ???

S8: ????

1. Draw a block diagram illustrating how you implemented your program. Include all conversion routines required for implementation of this program



**IV. Conclusions**

Thus, the purpose of this experiment was accomplished. By implementing a bit manipulation routine for a given input, students gained familiarity with the bit manipulation instructions from the MC6800 instruction set. In addition, students learned how to download programs from a host computer and perform ASCII, BCD and Binary code conversion. The experiment was successful and ultimately allowed students to develop coding skills for future labs.

**References**

[1] Experiment 4 Lab Manual

[2] Educational Computer Board manual appendix