Mathematics 3670: Laboratory 8

Pre-lab Exercises

1. Examine the assembly language program `lab8a.asm`. This program uses two **trap service routines** which use the assembly names `HALT` and `PUTS`. What are the corresponding 8-bit trap vector values for each of these? Using information from Table A.2 of Appendix A, determine the machine language encoding for each of these. Give the encoding as a 4-digit hexadecimal value.

   `HALT` _____ `PUTS` _____

2. In `lab8a.asm` the symbol `HELLO` corresponds to the address `x3007`. Given this information, what is the machine language encoding for the first instruction of the program?

   The encoding for `LEA R0,HELLO` is _____

3. Statements that begin with a period (`.ORIG`, `.STRINGZ`, and `.END` in our program) are "pseudo-ops." Refer to pp. 182–183 for an explanation of each of these.

   The value of the symbol `COURSE` is `x3016`. Convince yourself of this fact. It helps to know that `\n` stands for the "newline" character, which is the ASCII character `lf`.

   Determine the numerical address for the symbol `UNIV`.

Symbol	Address
`HELLO`	`x3007`
`COURSE`	`x3016`
`UNIV`	

4. Using Table E.2 (page 616), determine the values which will be loaded in memory, starting at address `x3016`, as a result of loading `lab8a.obj`, the assembled form of this program.

5. Examine the program shown in `lab8b.asm`. The program is intended to display N asterisks in one horizontal row. Carefully trace the execution of this program, keeping track of the values of `R1`, `R2`, and `R3` immediately before the instruction of line 18 has been performed. Does the program do what it claims?

6. Examine the program shown in `lab8c.asm`, which is intended to determine the minimum of two integer values. Carefully examine the structure of the program, then trace its execution, keeping track of all registers being used. Does the program do what it claims?

Lab Exercises

1. Obtain copies of this week's lab files and place them in your **lc3** folder.
2. Launch the LC-3 simulator, then perform the following actions:
   (a) Assemble `lab8a.asm`.
   (b) Load `lc3os.obj`.
   (c) Load `lab8a.obj`.
   (d) Set a breakpoint at `x3000`.
   (e) Inspect memory beginning at `x3000` and compare memory values with your pre-lab predictions.
   (f) Execute the program. To do this, hit the `Continue` button to reach the first instruction, then use the `Next` button to see the effect of each instruction. Using `Next` (instead of `Step`) allows us to treat the trap service routines as if they were single instructions. Carefully observe how the registers change and watch for output to appear in the lower-left window.
3. In a similar way, investigate `lab8b.asm` with the simulator. Here are some things to try:
   (a) Assemble and load the program.
   (b) Set a breakpoint at `STOP`.
   (c) Run the program to completion, verifying the output is correct.
   (d) Try different values of `N`.
   (e) Watch the program in "slow motion" by setting a breakpoint at the instruction of line 18. Carefully observe how the registers change over time.
4. Investigate the behavior of `lab8c.asm` with the simulator. By changing the values of `X` and `Y`, run the program with the following test cases:
   - `X = 6, Y = 8`
   - `X = 8, Y = 6`
   - `X = 8, Y = 8`
5. Using these programs as a model, implement a program which will output a triangular array of asterisks, consisting of N rows. The first row should have one asterisk and the last row should have N asterisks. For example, when $N = 4$, the output should be:

   ```
   *
   **
   ***
   ****
   ```

   Place your code in `lab8d.asm`.
6. Implement an LC-3 assembly language program that outputs a rectangular grid of asterisks, consisting of N rows and M columns. The values of N and M are to be found in two words of memory. Place your code in `lab8e.asm`.
7. Implement an LC-3 assembly language program that outputs a rectangular grid, as before, only now with a checkerboard pattern. For the character in row r and column c, output `*` if the sum is even and `.` if it is odd. Place your code in `lab8f.asm`.
8. Implement an LC-3 assembly language program, `lab8g.asm`, which forms the product NM, where N and M are found in two words of memory. Assume each is a non-negative value.

Submissions

Before submitting your work, make sure your name appears in each of the programs you wrote. Also, ensure that each of your programs is generously commented.

Create a **lab8** folder and place copies of all `.asm` programs you wrote in this folder. Submit the folder by dragging it onto the EIU submission icon.

Appendix

Contents of lab8a.asm

```
1           ;;
2           ;; Author: Bill Slough
3           ;;
4           ;; MAT 3670
5           ;;
6           ;; A first assembly-language LC-3 program
7           ;;
8           ;; This is a variation on the world-famous "Hello, world" program
9           ;; known to many programmers.
10          ;;
11          ;; Notes:
12          ;; 1. HALT is equivalent to TRAP x25  (see Table A.2, page 543)
13          ;; 2. PUTS is equivalent to TRAP x22
14          ;; 3. LEA (Load Effective Address) uses the IMMEDIATE addressing mode
15          ;; 4. \n represents the "newline" character
16
17          .ORIG x3000             ; specify the "origin"; i.e., where to load in memory
18
19          LEA R0,HELLO            ; R0 = address of output string
20          PUTS                    ; write("Hello, world!\n")
21
22          LEA R0,COURSE           ; R0 = address of output string
23          PUTS                    ; write("MAT 3670")
24
25          LEA R0,UNIV             ; R0 = address of output string
26          PUTS                    ; write("EIU")
27          HALT                    ;
28
29  HELLO   .STRINGZ "Hello, world!\n"
30  COURSE  .STRINGZ "MAT 3670\n"
31  UNIV    .STRINGZ "EIU\n"
32          .END
```

Contents of lab8b.asm

```
1           ;;
2           ;; Author: Bill Slough
3           ;;
4           ;; MAT 3670
5           ;;
6           ;; Example assembly language program with a simple loop
7           ;;
8           ;; Displays a line of N asterisks, where N is a given value in memory.
9           ;;
10
11          .ORIG x3000             ; specify the "origin"; i.e., where to load in memory
12
13          LD      R1,N            ;
14          NOT     R1,R1           ;
15          ADD     R1,R1,#1        ; R1 = -N
16
17          AND     R2,R2,#0        ; R2 = 0;
18  LOOP    ADD     R3,R2,R1        ; while (R2 < N)
19          BRzp    ELOOP           ;
20          LD      R0,STAR         ;   R0 = '*'
21          OUT                     ;   write('*')
22          ADD     R2,R2,#1        ;   R2 = R2 + 1
23          BRnzp   LOOP            ; end while
24  ELOOP
25          LEA     R0,NEWLN        ;
26          PUTS                    ; write('\n')
27
28  STOP    HALT                    ;
29
30  N       .FILL   6               ; how many characters to display?
31  STAR    .FILL   x2A             ; the character to display
32  NEWLN   .STRINGZ "\n"
33          .END
```

Contents of lab8c.asm

```
1           ;;
2           ;; Author: Bill Slough
3           ;;
4           ;; MAT 3670
5           ;;
6           ;; Example assembly language program with a conditional statement
7           ;;
8           ;; Determines the minimum value of two integers.
9           ;;
10
11          .ORIG x3000             ; specify the "origin"; i.e., where to load in memory
12
13  IF      LD      R0,X            ; if (x - y < 0)
14          LD      R1,Y            ;
15          NOT     R2,R1           ;
16          ADD     R2,R2,#1        ;
17          ADD     R3,R0,R2        ;
18          BRzp    ELSE            ; then
19  THEN    ST      R0,MIN          ;    min = x
20          BRnzp   EIF             ; else
21  ELSE    ST      R1,MIN          ;    min = y
22  EIF                             ; end if
23  STOP    HALT                    ;
24
25  X       .FILL   6               ; first value
26  Y       .FILL   8               ; second value
27  MIN     .BLKW   1               ; reserved for min(X, Y)
28          .END
```