Joseph Hyatt
CSE 2130-01

Homework 3

Chapter(7): 7.4, 7.9

7.4) Create the symbol table entries generated by the assembler when translating the following routine into machine code:

```
          .ORIG   x301C
          ST      R3, SAVE3
          ST      R2, SAVE2
          AND     R2, R2, #0
TEST      IN
          BRz     TEST
          ADD     R1, R0, #-10
          BRn     FINISH
          ADD     R1, R0, #-15
          NOT     R1, R1
          BRn     FINISH
          HALT
FINISH    ADD     R2, R2, #1
          HALT
SAVE3     .FILL   X0000
SAVE2     .FILL   X0000
          .END
```

The address generated by each instruction in the routine:

x301C: ST R3, SAVE3
x301D: ST R2, SAVE2
x301E: AND R2, R2, #0
x301F: TEST IN
x3020: BRz TEST
x3021: ADD R1, R0, #-10
x3022: BRn FINISH
x3023: ADD R1, R0, #-15
x3024: NOT R1, R1
x3025: BRn FINISH
x3026: HALT
x3027: FINISH ADD R2, R2, #1
x3028: HALT
x3029: SAVE3 .FILL X0000
x302A: SAVE2 .FILL X0000
.END

The entries of the symbol table generated by the assembler for the above routine:

Symbol	Address
TEST	x301F
FINISH	x3027
SAVE3	x3029
SAVE2	x302A

The best address or origin starts at address **x301C**. The address is incremented by 1 for every instruction in the routine. Each label in a routine is a symbol in the symbol table. The current address is allocated to the symbol in the symbol table. The routine ends using the directive .END.

7.9) What is the purpose of the . END pseudo-op? How does it differ from the HALT instruction?

The .END pseudo-op tells the assembler where the program ends. Any string that occurs after that will be disregarded and not processed by the assembler. It's the difference from HALT instruction in very fundamentals aspects, such as it's not an instruction, it can never be executed. It doesn't stop the machine, and it is just a marker that helps the assembler to know where to stop assembling.

Chapter(8): 8.5

8.5) What is the purpose of bit [15] in the KBSR?

The purpose of bit 15 in the KBSR is the bit 15 of KBSR is a ready bit which is used to check whether a new character has been received by the keyboard or not. It resides in the address "xFE00" and lets the processor know whether input has occurred through the synchronization mechanism. If bit 15 of the keyboard status register is 0, then the keyboard is enabled else the keyboard is disabled. As long as bit 15 of the KBSR is 0, then no key has been struck since the last time the processor has read the data from the data register.

Chapter(9): 9.2

9.2)

a. How many trap service routines can be implemented in the LC-3? Why?

In LC3 we have 256 TRAP service routines as there are 256 vector locations. These locations are located at x0 to xFF in the memory.

b. Why must a RET instruction be used to return from a TRAP routine? Why won't a BR (Unconditional Branch) instruction work instead?

Using BRnzp instructions we won't know the return location or address and doesn't restore the value of Program Counter, once TRAP routine is executed. Using RET instruction executing trap routing the control is jumped to the code that called TRAP instruction or TRAP routine, coping the value in register R7.

c. How many accesses to memory are made during the processing of a TRAP instruction? Assume the TRAP is already in the IR.

One access to memory is made during the TRAP process instruction which is already in IR.

Chapter(10): 10.8

10.8)

The following operations are performed on a stack: PUSH A, PUSH B, POP, PUSH C, PUSH D, POP, PUSH E, POP, POP, PUSH F

a. What does the stack contain after the PUSH F?

Operation	Stack
PUSH A	A
PUSH B	A B
POP	A

PUSH C	A C
PUSH D	A C D
POP	A C
PUSH E	A C E
POP	A C
POP	A
PUSH F	A F

Initially, the stack was empty. After PUSH A the stack contains 1 element which is A. So, the stack contents after the last push statement are: A F. The top of the stack points to F

b. At which point does the stack contain the most elements? Without removing the elements left on the stack from the previous operations, we perform: PUSH G, PUSH H, PUSH I, PUSH J, POP, PUSH K, POP, POP, POP, PUSH L, POP, POP, PUSH M

Operation	**Stack**
PUSH G	A F G
PUSH H	A B G H
PUSH I	A B G H I
PUSH J	A B G H I J
POP	A B G H I
PUSH K	A B G H I K
POP	A B G H I
POP	A B G H
POP	A B G
PUSH L	A B G L

POP	A B G
POP	A B
PUSH M	A B M

The stack contains the maximum elements after PUSH J and PUSH K operations. The number of elements in the stack at both of these points is 6.

c. What does the stack contain now?

Now, the stack contains 3 elements: A B M