Andrew Hedy

Professor Hou

CSE 313

11/30/2016

Homework 4

Chapter 9:

9.2a.) How many trap service routines can be implemented in the LC-3? Why? **The LC-3 was designed so that up to 256 service routines can be specified. The TRAP instruction is made up of two parts: the TRAP opcode (1111) and the trap vector (bits [7:0]). Since there are 8 bits making up the trap vector that gives us 256 service routines.**

9.2b.) Why must a RET instruction be used to return from a TRAP routine? Why won’t a BR (Unconditional Branch) instruction work instead? **The unconditional branch does not know the memory location of the TRAP routine. The RET instruction holds that information in R7 and is able to find its way back to its original location.**

9.2c.) How many accesses to memory are made during the processing of a TRAP instruction? Assume the TRAP is already in the IR. **If the trap memory is in the IR than only 1 access to memory would be made.**

9.12.) Assuming that the memory locations at DATA get filled in before the program executes, what is the relationship between the final values at DATA and the initial values at DATA? (hint: what kind of sort is this?)

.ORIG x3000

LEA R0, DATA ;Load address of DATA into R0

AND R1, R1, #0 ;Clear R1

ADD R1, R1, #9 ;Add 9 to R1

LOOP1 ADD R2, R0, #0 ;Add value of DATA into R2

ADD R3, R1, #0 ;Add value of R1 into R3

LOOP2 JSR SUB1 ;Jump to SUB1

ADD R4, R4, #0 ;Check R4

BRzp LABEL ;If zero or positive jump to Label

JSR SUB2 ;Jump to Sub2

LABEL ADD R2, R2, #1 ;Increment R2

ADD R3, R3, #-1 ;Decrement R3

BRp LOOP2 If R3 is positive jump to LOOP2

ADD R1, R1, #-1 ;Decrement R1

BRp LOOP1 ;If R1 is positive jump to LOOP1

HALT

DATA .BLKW 10 x0000 ;Reserve 10 memory locations for DATA

SUB1 LDR R5, R2, #0 ;Load R2 into R5 with an offset of 0

NOT R5, R5 ;Negate R5

ADD R5, R5, #1 ;Add 1 to R5 for 2’s complement

LDR R6, R2, #1 ;Load value of R2 with offset #1 into R6

ADD R4, R5, R6 ;Add R5 and R6 and store into R4

RET ;Return from subroutine “SUB1”

SUB2 LDR R4, R2, #0 ;

LDR R5, R2, #1 ;

STR R4, R2, #1 ;

STR R5, R2, #0 ;

RET ;

.END

**This program sorts the input data and outputs the same data into ascending order.**

Chapter 10:

10.4.) Write a function that implements another stack function, peek. Peek returns the value of the first element on the stack without removing the element from the stack. Peek should also do underflow error checking. Why is overflow checking unnecessary? (hint: remember to check underflow, overflow checking is not necessary since we are not adding anything to the stack)

**JSR PEEK**

**PEEK ST R1, SAVE1 ; SAVE R2**

**ST R2, SAVE2 ; SAVE R1**

**LD R1, BASE ;R1 CONTAINS ADDRESS TO BOTTOM OF STACK**

**LD R3, MAX ;R3 CONTAINS ADDRESS TO TOP OF STACK**

**ADD R2, R3, R1 ;COMPARE TOP AND BOTTOM**

**BRz FAIL ;UNDERFLOW (STACK IS EMPTY)**

**LDR R0, R3, #0 ;STORE R3 INTO R0 WITH OFFSET #0**

**BRnzp SUCCESS ;BRANCH**

**BASE .FILL xC001**

**MAX .FILL xC005**

**SAVE1 .FILL x0000**

**SAVE2 .FILL x0000**

**The hint for overflow was given. Overflow checking is not necessary since we are not adding anything to the stack. Underflow is necessary to make sure the stack is not empty.**

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

10.8a.) What does the stack contain after the PUSH F? A stack is first in, last out. After the first pop only A is left. After the second pop A and C are left. After the third and fourth pop only A is left and then the last F is pushed. **The stack contains A and F after the PUSH F.**

10.8b.) 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

**Without removing the elements left on the stack from the previous operations we would have the most elements before the first pop. Before the first pop we would have elements A, F, G, H, I, and J. After the first pop we would still have more elements than any other time throughout the operation with elements A, F, G, H, and I.**