Assembly Language Homework

7.5a)

.ORIG x3000 ;Placement of instructions in memory

LD R2, ZERO ;Load zero (.FILL x0000) to register 2

LD R0, M0 ;Load M0(.FILL x0004) into register 0

LD R1, M1 ;Load M1(.FILL x0803) into register 1

LOOP BRz DONE ;Loop until R1 is zero, then jump to DONE

ADD R2, R2, R0 ;Add R0 to R2

ADD R1, R1, -1 ;Add -1 to R1

BR LOOP ;Loop multiples M1 and M0

DONE ST R2, RESULT ;R2 value is stored into RESULT

HALT ;Stop

RESULT .FILL X0000 ;Label

ZERO .FILL x0000 ;Label

M0 .FILL x0004 ;Label

M1 .FILL x0803 ;Label

.END ;End of program

7.5b) The value stored in RESULT will be x200C since x0803 is 2051 in decimal and 2051 times 4 is 8204. 8204 is x200C.

7.7

.ORIG x3000

AND R2, R2, #0

ADD R2, R2, #1

AND R1, R1, #0

AND R3, R3, #0

AND R4, R4, #0

ADD R4, R4, #-16

LOOP AND R5, R0, R2

BRz NEXT

ADD R5, R5, #1

NEXT ADD R2, R2, R2

ADD R4, R4, #1

BRnp LOOP

ADD R1, R1, R2

HALT

.END

R0 is value

R1 is result

R2 is mask

R3 is 1 counter

R4 is loop counter

R5 reads bit

7.9 The difference between HALT and .END is that HALT is an instruction that stops the machine while the .END is a pseudo-op that is never executed and does not stop the machine. The .END is what tells the assembler to stop assembling essentially.

7.12

.ORIG x3000 ;Placement of instructions in memory

AND R5, R5, #0 ;Clear R5

AND R3, R3 , #0 ;Clear R3

ADD R3, R3, #8 ;Add 8 to R3

LDI R1, A ;Load A(x4000) to R1

ADD R2 , R1, #0 ;Adds R1 to R2

AG ADD R2, R2 , R2

ADD R3 , R3, #-1 ;Subtracts R3 by 1 8 times

BRnp AG ;Adds R2 8 times

LD R4, B ;Load B(xFF00) to R4

AND R1, R1, R4 ;R1 = R1(A) AND R4(B)

NOT R1, R1 ;Negates R1

ADD R1, R1, #1 ;Add 1 to R1

ADD R2, R2, R1 ;R2 ;Adds Negation of R1 to R2

BRnp NO ;Checks if Result overflows and zeroes

ADD R5, R5, #1 ;Adds 1 to R5 if it does

NO HALT ;Stop

B .FILL xFF00 ;Label

A . FILL x4000 ;Label

.END ;End of program

Compares if highest 8 bits are equal to lowest 8 bits.

7.13

Line No.

1 .ORIG x3000

2 ONE LD R0, A

3 ADD R1 , R1, R0

4 TWO LD R0, B

5 ADD R1, R1, R0

6 THREE LD R0 , C

7 ADD R1, R1, R0

8 ST R1, SUM

9 TRAP x25

10 A .FILL x0001

11 B .FILL x0002

12 C .FILL x0003

13 D .FILL x0004

14 .END

The two errors are at line 3 and 8. At line 3, R1 is not initialized before being used and needs to be detected by debugging at run time. At line 8, SUM is not defined, which is an assembly time error.

7.21 The program counts the number of negative values from x4000 – x4009 and stores it in x5000.

7.23

.ORIG x3000

LD R0, PTR

ADD R1, R0, #0

AGAIN LDR R2 , R1, #0

BRz CONT

ADD R1, R1, #1

BRnzp AGAIN

CONT (a) ADD R1, R1, #-1

LOOP LDR R3, R0, #0

(b) LDR R4, R1, #0

NOT R4, R4

ADD R4, R4, #1

ADD R3, R3, R4

BRnp NO

(c) ADD R0, R0, #1

(d) ADD R1, R1, #-1

NOT R2, R0

ADD R2, R2 , #1

ADD R2 , R1, R2

BRnz YES

(e) BRnzp LOOP

YES AND R5, R5, #0

ADD R5, R5, #1

BRnzp DONE

NO AND R5, R5, #0

DONE HALT

PTR .FILL X4000

.END