1)

Solution: Since MDR is only 8 bits, the memory is organized into addressable bytes.

Fetch the first two bytes of the instruction

Step 1: MAR ← PC;

Step 2: MDR ← M(MAR), PC ← PC+1 ; Get the high byte of the instruction and increment PC

Step 3: IR(15…8) ← MDR

Step 4: MAR ← PC; Return

Step 5: MDR ← M(MAR), PC ← PC+1 ; Get the low byte of the instruction and increment PC

Step 6: IR(7…0) ← MDR

Step 7: MAR ← PC;

Step 8: MDR ← M(MAR), PC ← PC+1

Step 9: TEMP(15…8) ← MDR

Step 10: MAR ← PC;

Step 11: MDR ← M(MAR), PC ← PC+1

Step 12: TEMP(7…0) ← MDR ; The return address is pushed onto the stack

Step 13: MDR ← PC(7…0)

Step 14: MAR ← SP

Step 15: M(MAR) ← MDR, SP ← SP-1 ; Push the lower byte of return address onto stack

Step 16: MDR ← PC(15…8)

Step 17: MAR ← SP

Step 18: M(MAR) ← MDR, SP ← SP-1 ;

Step 19: PC ← TEMP ; Goto fetch and Execute cycle.

2)

Solution Since ADIW ZH:ZL,32 is a 16-bit instruction and memory is organized into consecutive bytes, the instruction occupies two consecutive bytes. Thus, two memory accesses are needed to fetch the instruction into the IR.

Fetch cycle

Step 1: MAR ← PC;

Step 2: MDR ← M(MAR), PC ← PC+1 ; Get the high byte of the instruction and increment PC

Step 3: IR ← MDR ;

Step 4: MAR ← PC;

Step 5: MDR ← M(MAR), PC ← PC+1 ;

Execute cycle

Step 6: AC ← R30

Step 7: AC ← AC + MDR ; Add 32 to ZL

Step 8: R30 ← AC ; Write back to register file

Step 9: AC ← R31

Step 10: If (C==1) then AC ← AC +1 ; Increment ZH if there was a carry

Step 11: R31 ← AC ; Write it back to register file

3)

a)

3.	MAIN:	LDI YL, low(addrB)
4.		LDI YH, high(addrB)
5.		LDI YL, low(LAddrP)
6.		LDI YH, high(LAddrP)
8.		LDI YL, low(addrA)
9.		LDI YH, high(addrA)

b)The two hex numbers being multiplied are 0203 and 010C

c)

Lines 11-13 result in 00 24

Lines 14-19 result in 00 00 24

Then lines 20-22 stores these bytes in

104 24

105 00

106 00

d)

Lines 11-13 result in 00 18

Lines 14-19 result in 00 00 18

Then lines 20-22 stores three bytes in

104 24

105 18

106 00

107 00

e) ldi oloop, 2;

4)

Program Address	Binary			
0000	1100	0000	0000	0011
-	-	-	-	-
0002	1101	-	-	-
0003	1001	0101	0001	1000
0004				
-	-	-	-	-
000B	1110	0000	1011	0000
000C	1110	0110	1010	0000
000D	1110	0000	1101	0000
000E	1110	0110	1100	0001
000F	1100	1111	1111	1111
-	-	-	-	-
100F	1011	0111	0000	1001
1010	1001	0011	0000	1001
1011	1001	0101	0001	0011
1012	1001	0011	0001	1101
1013	1001	0101	0000	1000

5)