B38DF Computer Architecture and Embedded Systems. Part II. Tutorial 2 – coding

AVR Atmel Instruction Set (incomplete)

Instruction	Description
LDI Rp, K	Load integer value K to register Rp
LDS Rp, k	Load contents of data at location k into Rp
CPI Rp, K CP Rp, Rq	Compare the contents of register Rp with value K Compare the contents of register Rp with the contents of Rq
ADD Rp, Rq	Adds the contents of Rq with Rp stores the result into Rp
SUB Rp, Rq	Subtracts the contents of Rq from Rp and stores the result into Rp
INC Rp	Increments the contents of Rp
DEC Rp	Decrements the contents of Rp
BRBS Z, k BRBS N, k	Jump to location k if ALU operation results in a zero flag being set Jump to location k if ALU operation results in a negative flag being set
BRGE k BRLT k	Jump to location k if ALU operation results in Greater than or Equal to Jump to location k if ALU operation results in Less than
BREQ k BRNE k	Jump to location k if ALU operation results in Equal to … Jump to location k if ALU operation results in Not Equal to …
RJMP k	Jump to location k

1. Translate the following C-code into AVR Atmel assembly code.

```
sum=0,term=1;
for(i=1; i<=10; i++)
{
  sum += term;
  term += 1;
}
```

```
x=1;
while (x<16) do
{
 // instructions to be repeated
 x++;
}
```

```
.def sum = r16
.def term = r17
.def i = r18
      ldi sum,0
      ldi term,1
      ldi i,1
again:
      cpi i,11
      brge exit
      add sum,term
      inc term
      inc i
      rjmp again
exit:
      nop
```

```

LDI R18,1
L1:
   CPI R18,16
   BRGE L3
   ;instructions to be repeated
   INC R18
   RJMP L1
L3:
```

2. Using the HWU instruction set write a programme to execute the following formula

a) **A = ((B + C) AND D) – E** where D0 = B = 2, D1 = C = 5, D2 = D = 15 and D3 = E = 3.

b) Compile it as a series of hexadecimal numbers.

Solution:

a)

D0 = B = 2, D1 = C = 5, D2= D=15, D3=E =3

LD R0, D0 // B

```
LD R1 D1   // C
LD R2, D2  // D
LD R3, D3  // E
ADD R0,R1  // B+ C
AND R0, R2
SUB R0,R3   // R0 = A
```

b)

```
Coded
100060
110162
120262
130362
209062
50A062
30B062
```

3. Using the HWU machine mnemonics write a programme to execute this programme fragment

```
sum = 0xAA;
i = 0;
while (i < 20)
{
  sum = sum ^ 0x20;
  i= i+1;
}
```

where D2=0x20 is hexadecimal 20 and D0=0xAA is hexadecimal AA and ^ is bit exclusive OR and D1=20 keeps the number of iterations.

Solution:

D0 = 0xAA, D1 =20 // terminator, D2= 0x20

```
LD R0, D0   // sum
SUB R1, R1  //  zero index
L1:  LD R2, D2
XOR R0,R2  // sum = sum ^ 0x20
INC R1        // index ++
LD R2,D1    //
SUB R2,R1 // ?20 iterations?
JZ exit   // yes
JP L1   //
```

Exit : NOP

4. Use the HW machine instruction set to code the following statements

a) `if (a < 0) sum = sum + 3; else sum = sum - 1;`

b)
```
case (a)
     2: x = x + 2;
     3: x = x - 3;
     default x = 2;
endcase
```

c) `y =10; x = 9; while (y > 0) { x = x – 1; y = y - 1; }`

Solution:

a)

```
D0 =a

     SUB R2,R2  // zero sum
     LD R0, D0
     SUB R1,R1  // zero test
     SUB R1,R0
     JL   L1  // a<0 ?
     JP L2
 L1: INC R2
     INC R2
     INC R2  // sum = sum +3
     JP exit
 L2:  DEC R2  // sum = sum -1
exit:  NOP
```

b)

```
D0 = a,  R3 = x, D1 = 2, D2 = 3

     LD R1, D0
     LD R2, D1
     SUB R2 R1  // == 2?
     JZ L1 //  yes
     LD R1, D0
     LD R2, D2
     SUB R2,R1 // == 3?
     JZ L2
     LD R3, D1  x=2  default
     JP exit
     L1: LD R2,D1
     ADD R3,R2    // x= x+2
     JP exit
     L2: DEC R3
     DEC R3
```

```
DEC R3    // x = x-3
Exit: NOP
```

c)

D0 = 10 ,y the terminator D1 = x

```
SUB R0, R0 // zero the index
LD R1, D1
LD R2,D0  // y= 10
L1 : DEC R1     // x= x-1
DEC R2      // decrement the iteration counter y
JZ  exit      //  y >0 ?
JP L1
Exit: NOP
```