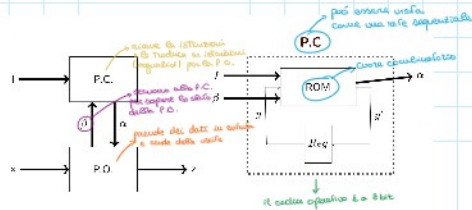


lunedì 18 maggio 2020 18:41

- 1) codificare in RTL
- 2) descrivere come modificare la P.O.
- 3) valore degli  $\alpha$  nell'esecuzione
- 4) cosa deve fare la P.C.



- 1: Codice operativo a 8 bit.
- Num.  $b$  e  $y = \lceil \log_2 (\text{Num. passi nella } a\text{-sequenza più lunga}) \rceil$
- Righe della ROM: 

COP	$\beta$	$y$	$y'$	$\alpha$
Address			Output	

Esercizio tratto dalla prova scritta del 18/04/2007

IR: C=O 1702  $\rightarrow$  exemplo

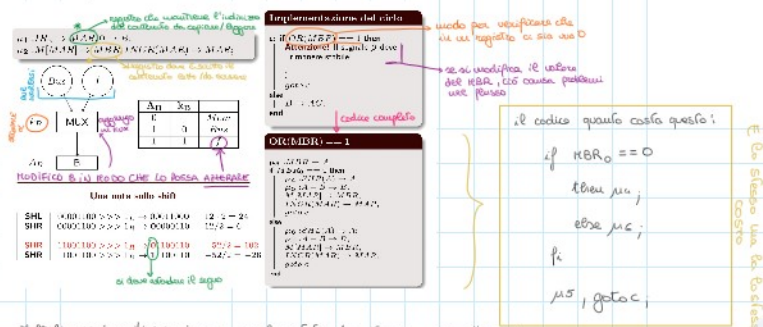
1002	$m_2$	0
1003	$m_1$	2
1004	$m_2$	10
1005	$m_8$	0
1006	$m_1$	3
1007	$m_5$	7
1008	$m_6$	0

```
res = 0
while X <= N do
  if X % 2 == 0 then
    res = res + X / 2
  else
    res = res + X * 2
  end
  X = X + 1
end while
```

AC: 47

2) occur before

→ rapporto che mantiene l'indignità



Moltiplicazioni e divisioni sono implementate tramite somme e sottrazioni

```

(1)  $\vdash R_1 : \text{MAR}_0 \rightarrow \text{MAR}_1$ 
(2)  $\vdash \text{MAR} : \text{MAR}_1 \rightarrow \text{MDR}(\text{INCR}(\text{MAR}_1)) \rightarrow \text{MAR}_1$ 
c: if  $\text{OR}(\text{MDR}) = 1$  then
  if  $\text{MDR} = 1$  then
     $\mu_1 : \text{SHL}(A) : A$ 
     $\mu_2 : A \rightarrow B, \text{MAR} : \text{MDR}(\text{INCR}(\text{MAR})) \rightarrow \text{MAR}, \text{goto c}$ 
  else
     $\mu_3 : \text{SHL}(A) : A$ 
     $\mu_4 : A \rightarrow B, \text{MAR} : \text{MDR}(\text{INCR}(\text{MAR})) \rightarrow \text{MAR}, \text{goto c}$ 
end
else
   $\mu_5 : B \rightarrow AC$ 
end

```

$M_2$	$G_1$	$M_3$	OPERAZIONE
0	1	0	$\Gamma$
0	2	1	$B \cup A$
0	1	0	$B$
0	1	1	$A \cup B$
1	2	0	$A \cap B$
1	2	1	$A \cup B$
1	1	0	$A \cap B$
1	1	1	$A \cap B$

L'ALU non è strutturata per far scrivere da A

$$M: IR_X \rightarrow KAR, 0 \rightarrow B$$

$\mu_2: M[HAR] \rightarrow MBR, INC(MAR) \rightarrow MAR$

$$\mu_2: \text{NBR} \rightarrow A$$
$$\text{ma! } \text{SHR}(A) \rightarrow A$$
[illegible]

[illegible]

Signal $\beta$			Signal $\alpha$		
$\beta$	$\text{CR}(\beta) \backslash \text{NR}(\beta)$	$\text{NR}(\beta)$	$\beta \in \mathcal{V}$	$\beta \notin \mathcal{V}$	$\beta$
$\text{CR}^0_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^1_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^2_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^3_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^4_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^5_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^6_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^7_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^8_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^9_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{10}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{11}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{12}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{13}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{14}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{15}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{16}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{17}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{18}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{19}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{20}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{21}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{22}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{23}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{24}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{25}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{26}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{27}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{28}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{29}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{30}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{31}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{32}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{33}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{34}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{35}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{36}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{37}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{38}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{39}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{40}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{41}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{42}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{43}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{44}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{45}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{46}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{47}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{48}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{49}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{50}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{51}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{52}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{53}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{54}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{55}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{56}_\beta$	1	1	CR	CR	$\mu_1$
$\text{CR}^{57}_\beta$	1	1	CR	CR	$\mu_1$
$\text{$					

[illegible]

## Soluzione:

```

 $\mu_1 : IR_x \rightarrow MAR, 0 \rightarrow A;$ 
 $\mu_2 : M[MAR] \rightarrow MBR, INCR(MAR) \rightarrow MAR;$ 
 $\mu_3 : MBR \rightarrow T_1;$ 
 $\mu_4 : SHR(T_1) \rightarrow T_1, T_1 \rightarrow T_1;$ 
c1: if  $OR(T_1) == 1$  then
     $\mu_2 : M[MAR] \rightarrow MBR, INCR(MAR) \rightarrow MAR;$ 
    // Se il valore letto è pari
    if  $MBR_0 == 0$  then
         $\mu_5 : MBR \rightarrow B;$ 
         $\mu_6 : A + B \rightarrow A, DECR(T_1) \rightarrow T_1, DECR(T_2) \rightarrow T_2, goto c1;$ 
    else
         $\mu_7 : DECR(T_1) \rightarrow T_1, DECR(T_2) \rightarrow T_2, goto c1;$ 
    end
else
     $\mu_0 : \phi;$ 
end
c2: if  $OR(T_2) == 1$  then
     $\mu_2 : M[MAR] \rightarrow MBR, INCR(MAR) \rightarrow MAR;$ 
    if  $MBR_0 == 1$  then
         $\mu_5 : MBR \rightarrow B;$ 
         $\mu_8 : A - B \rightarrow A, DECR(T_2) \rightarrow T_2, goto c2;$ 
    else
         $\mu_9 : DECR(T_2) \rightarrow T_2, goto c2;$ 
    end
else
     $\mu_{10} : A \rightarrow AC;$ 
end

```

## Esercizio tratto dalla traccia del 07/04/05

Estendere il set di istruzioni della macchina con l'operazione **CNTVn X** che, dati due vettori  $V$  e  $W$  di 32 elementi, posti in RAM a partire rispettivamente dagli indirizzi memorizzati nelle locazioni  $M[X]$  e  $M[X+1]$ , restituisce nell'accumulatore il numero di coppie di elementi  $V[i]$  e  $W[i]$  ( $0 \leq i \leq 31$ ) che occupano la stessa posizione nei rispettivi vettori e tali che  $V[i] < W[i]$ .

Memory:

102	ind1	226
103	ind2	113
...	...	...
113	W[0]	-2
114	W[1]	4
115	W[2]	-1
...	...	...
144	W[31]	-3
...	...	...
226	V[0]	6
227	V[1]	3
228	V[2]	4
...	...	...
257	V[31]	-1

Pseudocode **CNTVn(X)**:

```

ind1  $\leftarrow M[X];$ 
ind2  $\leftarrow M[X+1];$ 
cnt  $\leftarrow 0; sz \leftarrow 32;$ 
while  $sz > 0$  do
     $vi \leftarrow M[ind1];$ 
     $wi \leftarrow M[ind2];$ 
    if  $vi < wi$  then
        cnt ++;
    ind1 ++; ind2 ++; sz --;
end

```

```

 $\mu_1 : IR_x \rightarrow MAR;$ 
 $\mu_2 : M[MAR] \rightarrow MBR, INCR(MAR) \rightarrow MAR;$ 
 $\mu_3 : MBR \rightarrow IND1, M[MAR] \rightarrow MBR;$ 
 $\mu_4 : MBR \rightarrow IND2, 32 \rightarrow T1, 0 \rightarrow AC;$ 

```

Modifiche architetturali:

- Aggiunta e collegamento al bus dei registri **IND1** e **IND2**.
- Supporto all'istruzione  $0 \rightarrow AC$
- Supporto all'istruzione  $32 \rightarrow T1$

$A_{T1}$	$K'_{T1}$	$K'_{T1}$	f
0	-	-	Mem
1	0	0	Bus
1	0	1	INCR
1	1	0	DECR
1	1	1	Load 32

$\mu$	$A_{IR}$	$Z_{IR}$	$A_{AC}$	$K_{AC}$	$A_{MAR}$	$K_{MAR}$	$A_{MBR}$	$S$	$L$	$E$	$A_{T1}$	$K'_{T1}$	$K'_{T1}$	$A_{IND1}$	$A_{IND2}$
$\mu_1$	0	-	0	-	1	0	0	0	0	-	0	-	-	0	0
$\mu_2$	0	-	0	-	1	1	0	1	-	0	-	-	-	0	0
$\mu_3$	0	-	0	-	0	-	1	1	0	-	0	-	-	0	0
$\mu_4$	0	-	1	1	0	-	0	0	0	-	1	1	1	0	1

## Implementazione del ciclo

```

c: if  $OR(T1) == 1$  then
     $\mu_5 : IND1 \rightarrow MAR, INCR(IND1) \rightarrow IND1;$ 
     $\mu_6 : M[MAR] \rightarrow MBR, IND2 \rightarrow MAR, INCR(IND2) \rightarrow IND2;$ 
     $\mu_7 : MBR \rightarrow A, M[MAR] \rightarrow MBR;$ 
     $\mu_8 : MBR \rightarrow B;$ 
     $\mu_9 : A - B \rightarrow A;$ 
    if  $A_{31} == 1$  then
         $\mu_{10} : INCR(AC) \rightarrow AC, DECR(T1) \rightarrow T1, goto c;$ 
    else
         $\mu_{11} : DECR(T1) \rightarrow T1, goto c;$ 
    end
else
     $\mu_0 : \phi;$ 
end

```

Posso fare questo?

$A - MBR \rightarrow A$

```

if  $A_{31} == 1$  then
     $\mu_{10} : INCR(AC) \rightarrow AC;$ 
     $\mu_{11} : DECR(T1) \rightarrow T1, goto c;$ 
end

```

```

if  $A_{31} == 1$  then
     $\mu_{10} : INCR(AC) \rightarrow AC;$ 
else
     $\mu_0 : \phi;$ 
     $\mu_{11} : DECR(T1) \rightarrow T1, goto c;$ 
end

```



### Soluzione:

```

 $\mu_1 : IR_x \rightarrow MAR;$ 
 $\mu_2 : M[MAR] \rightarrow MBR, INCR(MAR) \rightarrow MAR;$ 
 $\mu_3 : MBR \rightarrow IND1, M[MAR] \rightarrow MBR;$ 
 $\mu_4 : MBR \rightarrow IND2, 32 \rightarrow T1, 0 \rightarrow AC;$ 
c: if  $OR(T1) == 1$  then
     $\mu_5 : IND1 \rightarrow MAR, INCR(IND1) \rightarrow IND1;$ 
     $\mu_6 : M[MAR] \rightarrow MBR, IND2 \rightarrow MAR, INCR(IND2) \rightarrow IND2;$ 
     $\mu_7 : MBR \rightarrow A, M[MAR] \rightarrow MBR;$ 
     $\mu_8 : MBR \rightarrow B;$ 
     $\mu_9 : A - B \rightarrow A;$ 
    if  $A_{31} == 1$  then
         $\mu_{10} : INCR(AC) \rightarrow AC, DECR(T1) \rightarrow T1, goto c;$ 
    else
         $\mu_{11} : DECR(T1) \rightarrow T1, goto c;$ 
else
     $\mu_0 : \phi;$ 

```

### Esercizio tratto dalla traccia del 07/07/11

Estendere il set di istruzioni della macchina con l'operazione **Count.Active X**. A partire dalla locazione di indirizzo  $X$  è presente in RAM un vettore  $V$  di 32 elementi. L'accumulatore contiene un numero a 32 bit. L' $i$ -esimo elemento del vettore  $V$  è attivo se l' $i$ -esimo bit dell'accumulatore vale 1. L'istruzione restituisce nell'accumulatore la somma degli elementi attivi del vettore  $V$ .

AC: 

1011001000001000010110010010101
---------------------------------

Memory

1052	V[0]	9
1053	V[1]	-2
1054	V[2]	1
1055	V[3]	5
1056	V[4]	3
1057	V[5]	-7
1058	V[6]	0
...	...	...
1058	V[31]	1

```

 $\mu_1 : IR_x \rightarrow MAR, 32 \rightarrow T1, 0 \rightarrow B;$ 
c: if  $OR(T1) == 1$  then
    if  $AC_{31} == 1$  then
         $\mu_2 : M[MAR] \rightarrow MBR, INCR(MAR) \rightarrow MAR;$ 
         $\mu_3 : MBR \rightarrow A;$ 
         $\mu_4 : A + B \rightarrow B, DECR(T1) \rightarrow T1,$ 
         $SHL(AC) \rightarrow AC, goto c;$ 
    else
         $\mu_5 : INCR(MAR) \rightarrow MAR, DECR(T1) \rightarrow T1,$ 
         $SHL(AC) \rightarrow AC, goto c;$ 
else
     $\mu_6 : B \rightarrow AC$ 

```

### Esercizio tratto dalla traccia del 30/04/14 (riadattamento)

Estendere il set di istruzioni della macchina con l'operazione **MULT2K X** che verifica se il numero all'indirizzo  $X$  è multiplo di  $2^k$ , dove  $k$  è il valore presente nell'accumulatore. Se la proprietà è verificata l'istruzione restituisce 1 nell'accumulatore, altrimenti restituisce 0.

```

if  $M[X] \% 2^k == 0$  then
     $AC \leftarrow 1;$ 
else
     $AC \leftarrow 0;$ 

```

#### Soluzione 1

```

num  $\leftarrow M[X];$ 
while  $AC \neq 0$  do
    if  $num_0 == 1$  then
        return 0;
    num  $\leftarrow SHR(num);$ 
     $AC \leftarrow AC - 1;$ 
return 1;

```

#### Soluzione 2

Usare una maschera

0000...0	11...1
$k$	

```

num  $\wedge$  00000000...011...1 =
{ 1  $num_{[0...k]} = 00...0$ 
  0  $1 \in num_{[0...k]}$ 

```

#### Soluzione 1

```
 $\mu_1 : IR_x \rightarrow MAR, AC \rightarrow T2;$   
 $\mu_2 : M[MAR] \rightarrow MBR;$   
 $\mu_3 : MBR \rightarrow T1;$   
c: if  $OR(T2) == 1$  then  
  if  $T1_0 == 1$  then  
     $\mu_4 : 0 \rightarrow AC;$   
  else  
     $\mu_5 : SHR(T1) \rightarrow T1,$   
     $DECR(T1) \rightarrow T2, goto c;$   
     $DECR(T2) \rightarrow T2, goto c;$   
else  
   $\mu_6 : 1 \rightarrow AC;$ 
```

**N.B.**  $DECR(T1) \rightarrow T2$  è errato anche da un punto di vista tecnico: il decremento è un'operazione interna al registro, perciò non si può scrivere il risultato del decremento di T1 direttamente in T2.

#### Soluzione 2

```
 $\mu_1 : IR_x \rightarrow MAR, AC \rightarrow T2, 1 \rightarrow B;$   
c:if  $OR(T2) == 1$  then  
   $\mu_2 : SHL(B) \rightarrow B,$   
   $DECR(T2) \rightarrow T2, goto c;$   
else  
   $\mu_3 : DECR(B) \rightarrow B;$   
 $\mu_4 : M[MAR] \rightarrow MBR;$   
 $\mu_5 : MBR \rightarrow A;$   
 $\mu_6 : A \wedge B \rightarrow T2;$   
if  $OR(T2) == 0$  then  
   $\mu_7 : 1 \rightarrow AC;$   
else  
   $\mu_8 : 0 \rightarrow AC;$ 
```