Lezione 3 – Esempi di programmi in linguaggio macchina LC-2

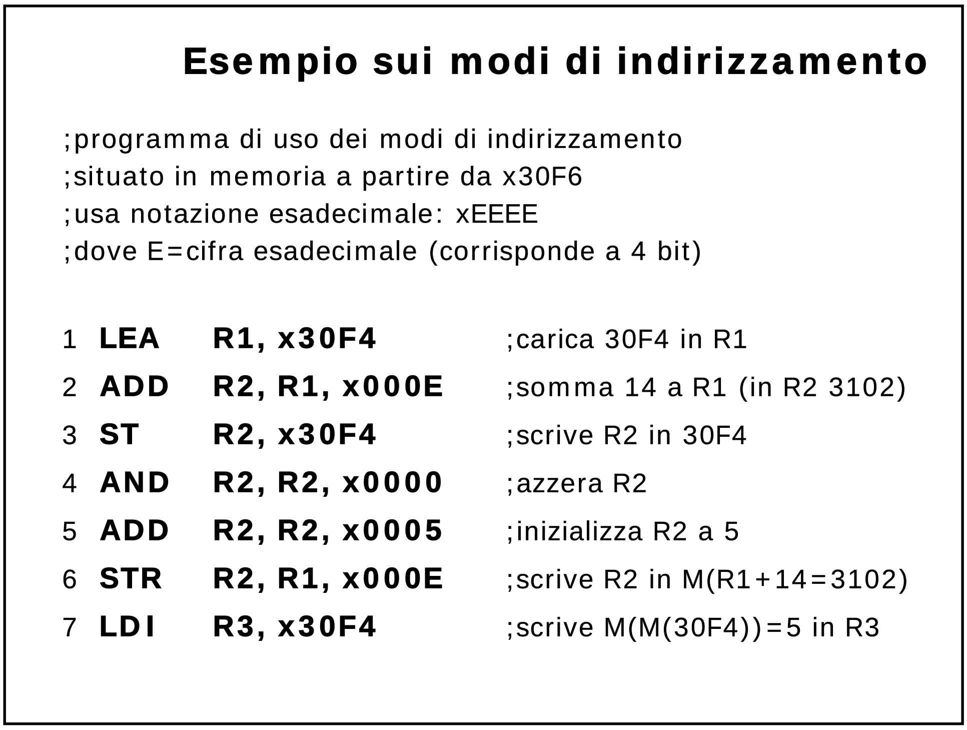
Architettura degli elaboratori

Modulo 2 – Linguaggio macchina

Unità didattica 2 – CPU LC-2

Nello Scarabottolo

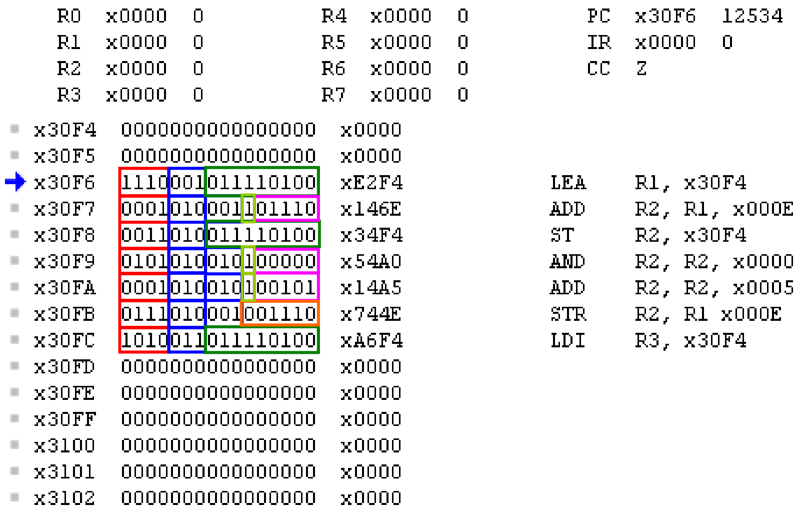
Università degli Studi di Milano - Ssri - CDL ONLINE



Esempio sui modi di indirizzamento

```
;programma di uso dei modi di indirizzamento
;situato in memoria a partire da x30F6
;usa notazione esadecimale: xEEEE
;dove E=cifra esadecimale (corrisponde a 4 bit)

LEA   R1, x30F4          ;carica 30F4 in R1
ADD   R2, R1, x000E      ;somma 14 a R1 (in R2 3102)
ST    R2, x30F4          ;scrive R2 in 30F4
AND   R2, R2, x0000      ;azzera R2
ADD   R2, R2, x0005      ;inizializza R2 a 5
STR   R2, R1, x000E      ;scrive R2 in M(R1+14=3102)
LDI   R3, x30F4          ;scrive M(M(30F4))=5 in R3
```

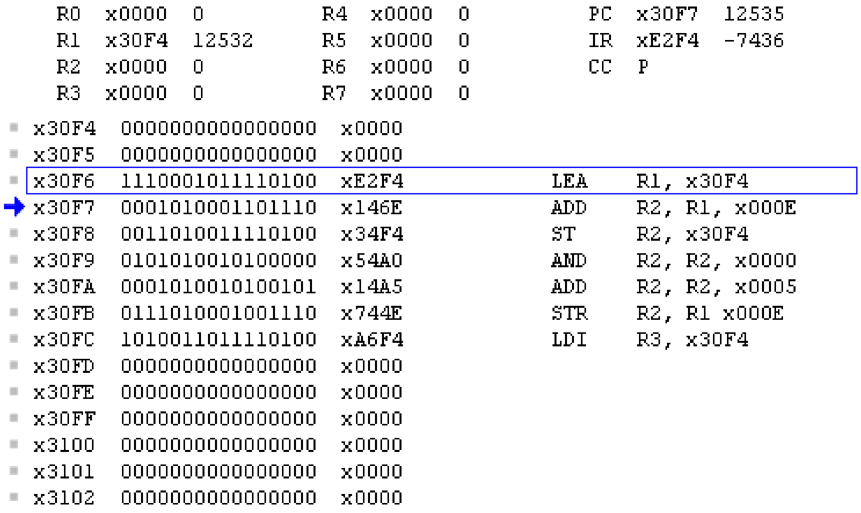
CPU e memoria all'inizio del programma



```
    R0  x0000  0          R4  x0000  0          PC  x30F6  12534
    R1  x0000  0          R5  x0000  0          IR  x0000  0
    R2  x0000  0          R6  x0000  0          CC  Z
    R3  x0000  0          R7  x0000  0

  x30F4  0000000000000000  x0000
  x30F5  0000000000000000  x0000
→ x30F6  1110001011110100  xE2F4          LEA    R1, x30F4
  x30F7  0001010001101110  x146E          ADD    R2, R1, x000E
  x30F8  0011010011110100  x34F4          ST     R2, x30F4
  x30F9  0101010010100000  x54A0          AND    R2, R2, x0000
  x30FA  0001010010100101  x14A5          ADD    R2, R2, x0005
  x30FB  0111010001001110  x744E          STR    R2, R1 x000E
  x30FC  1010011011110100  xA6F4          LDI    R3, x30F4
  x30FD  0000000000000000  x0000
  x30FE  0000000000000000  x0000
  x30FF  0000000000000000  x0000
  x3100  0000000000000000  x0000
  x3101  0000000000000000  x0000
  x3102  0000000000000000  x0000
```

Dopo istruzione 1

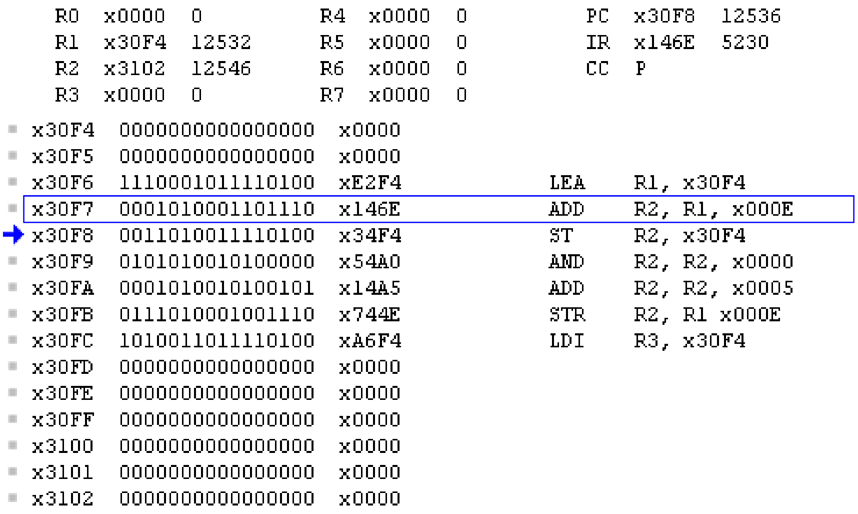
LEA R1, x30F4 ;carica 30F4 in R1



```
    R0  x0000  0          R4  x0000  0          PC  x30F7  12535
    R1  x30F4  12532      R5  x0000  0          IR  xE2F4  -7436
    R2  x0000  0          R6  x0000  0          CC  P
    R3  x0000  0          R7  x0000  0

  x30F4  0000000000000000  x0000
  x30F5  0000000000000000  x0000
  x30F6  1110001011110100  xE2F4          LEA    R1, x30F4
→ x30F7  0001010001101110  x146E          ADD    R2, R1, x000E
  x30F8  0011010011110100  x34F4          ST     R2, x30F4
  x30F9  0101010010100000  x54A0          AND    R2, R2, x0000
  x30FA  0001010010100101  x14A5          ADD    R2, R2, x0005
  x30FB  0111010001001110  x744E          STR    R2, R1 x000E
  x30FC  1010011011110100  xA6F4          LDI    R3, x30F4
  x30FD  0000000000000000  x0000
  x30FE  0000000000000000  x0000
  x30FF  0000000000000000  x0000
  x3100  0000000000000000  x0000
  x3101  0000000000000000  x0000
  x3102  0000000000000000  x0000
```

Dopo istruzione 2

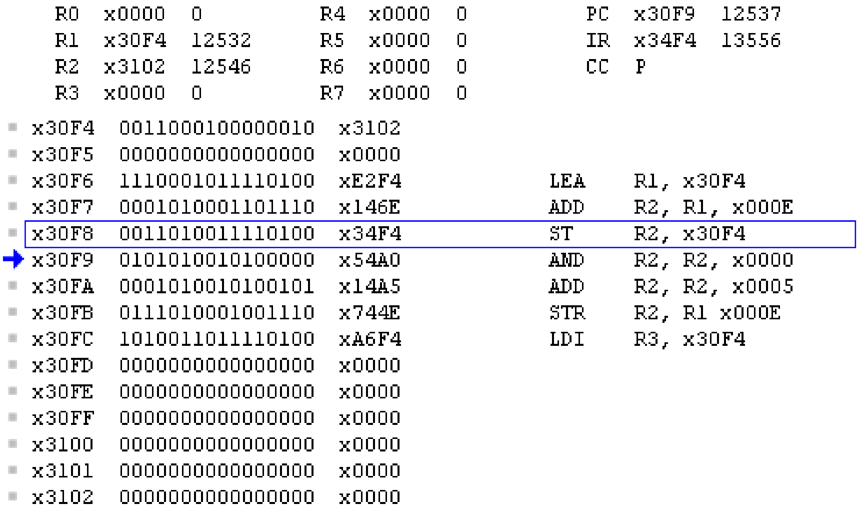
ADD R2, R1, x000E ;somma 14 a R1 (in R2 3102)



```
R0  x0000  0         R4  x0000  0          PC  x30F8  12536
R1  x30F4  12532     R5  x0000  0          IR  x146E  5230
R2  x3102  12546     R6  x0000  0          CC  P
R3  x0000  0         R7  x0000  0

x30F4  0000000000000000  x0000
x30F5  0000000000000000  x0000
x30F6  1110001011110100  xE2F4          LEA   R1, x30F4
x30F7  0001010001101110  x146E          ADD   R2, R1, x000E
x30F8  0011010011110100  x34F4          ST    R2, x30F4
x30F9  0101010010100000  x54A0          AND   R2, R2, x0000
x30FA  0001010010100101  x14A5          ADD   R2, R2, x0005
x30FB  0111010001001110  x744E          STR   R2, R1 x000E
x30FC  1010011011110100  xA6F4          LDI   R3, x30F4
x30FD  0000000000000000  x0000
x30FE  0000000000000000  x0000
x30FF  0000000000000000  x0000
x3100  0000000000000000  x0000
x3101  0000000000000000  x0000
x3102  0000000000000000  x0000
```

Dopo istruzione 3

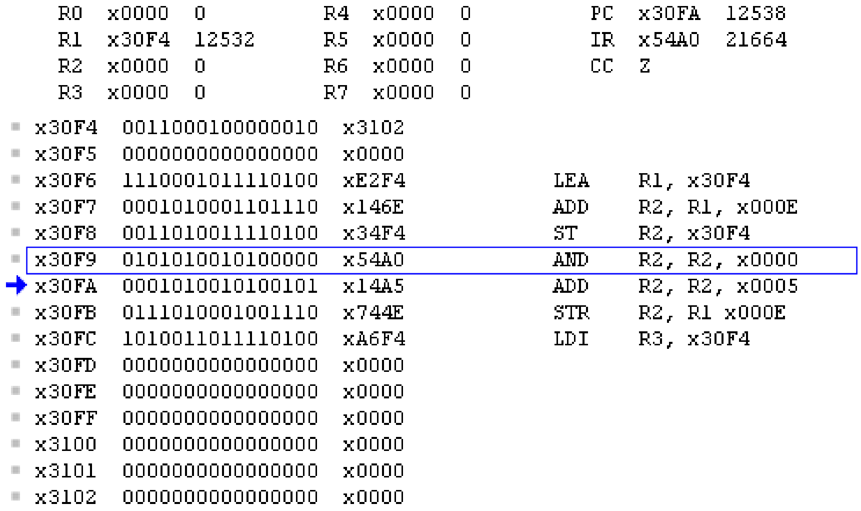
ST R2, x30F4 ;scrive R2 in 30F4



```
R0  x0000  0         R4  x0000  0          PC  x30F9  12537
R1  x30F4  12532     R5  x0000  0          IR  x34F4  13556
R2  x3102  12546     R6  x0000  0          CC  P
R3  x0000  0         R7  x0000  0

x30F4  0011000100000010  x3102
x30F5  0000000000000000  x0000
x30F6  1110001011110100  xE2F4          LEA   R1, x30F4
x30F7  0001010001101110  x146E          ADD   R2, R1, x000E
x30F8  0011010011110100  x34F4          ST    R2, x30F4
x30F9  0101010010100000  x54A0          AND   R2, R2, x0000
x30FA  0001010010100101  x14A5          ADD   R2, R2, x0005
x30FB  0111010001001110  x744E          STR   R2, R1 x000E
x30FC  1010011011110100  xA6F4          LDI   R3, x30F4
x30FD  0000000000000000  x0000
x30FE  0000000000000000  x0000
x30FF  0000000000000000  x0000
x3100  0000000000000000  x0000
x3101  0000000000000000  x0000
x3102  0000000000000000  x0000
```

Dopo istruzione 4

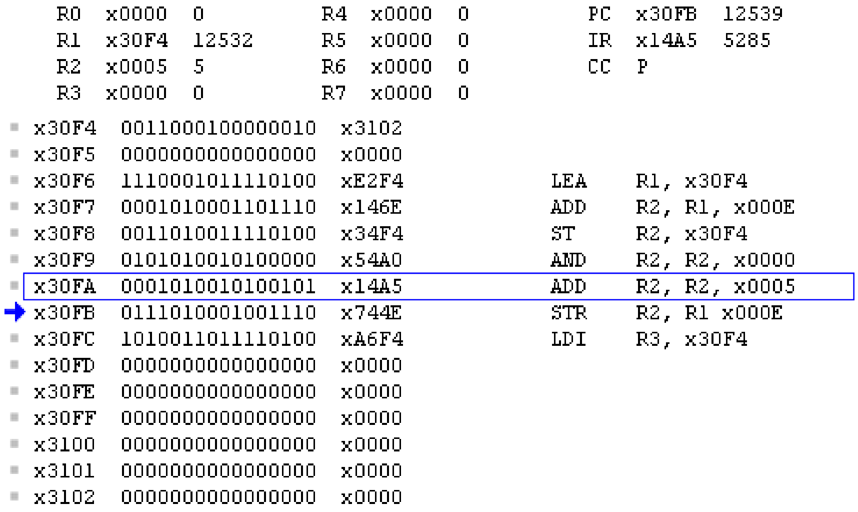
AND R2, R2, x0000 ;azzera R2



```
     R0  x0000  0          R4  x0000  0          PC  x30FA  12538
     R1  x30F4  12532      R5  x0000  0          IR  x54A0  21664
     R2  x0000  0          R6  x0000  0          CC  Z
     R3  x0000  0          R7  x0000  0

 ▪ x30F4  0011000100000010  x3102
 ▪ x30F5  0000000000000000  x0000
 ▪ x30F6  1110001011110100  xE2F4              LEA    R1, x30F4
 ▪ x30F7  0001010001101110  x146E              ADD    R2, R1, x000E
 ▪ x30F8  0011010011110100  x34F4              ST     R2, x30F4
 ▪ x30F9  0101010010100000  x54A0              AND    R2, R2, x0000
 → x30FA  0001010010100101  x14A5              ADD    R2, R2, x0005
 ▪ x30FB  0111010001001110  x744E              STR    R2, R1 x000E
 ▪ x30FC  1010011011110100  xA6F4              LDI    R3, x30F4
 ▪ x30FD  0000000000000000  x0000
 ▪ x30FE  0000000000000000  x0000
 ▪ x30FF  0000000000000000  x0000
 ▪ x3100  0000000000000000  x0000
 ▪ x3101  0000000000000000  x0000
 ▪ x3102  0000000000000000  x0000
```

Dopo istruzione 5

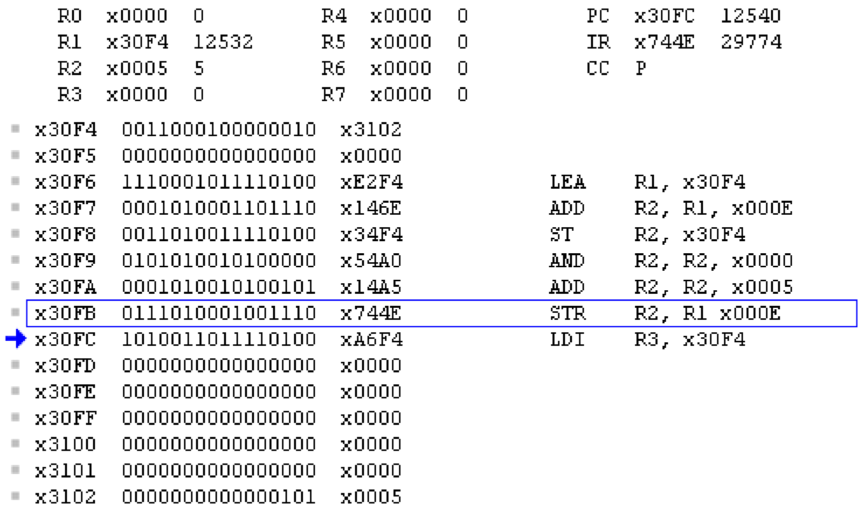
ADD R2, R2, x0005 ;inizializza R2 a 5



```
     R0  x0000  0          R4  x0000  0          PC  x30FB  12539
     R1  x30F4  12532      R5  x0000  0          IR  x14A5  5285
     R2  x0005  5          R6  x0000  0          CC  P
     R3  x0000  0          R7  x0000  0

 ▪ x30F4  0011000100000010  x3102
 ▪ x30F5  0000000000000000  x0000
 ▪ x30F6  1110001011110100  xE2F4              LEA    R1, x30F4
 ▪ x30F7  0001010001101110  x146E              ADD    R2, R1, x000E
 ▪ x30F8  0011010011110100  x34F4              ST     R2, x30F4
 ▪ x30F9  0101010010100000  x54A0              AND    R2, R2, x0000
 ▪ x30FA  0001010010100101  x14A5              ADD    R2, R2, x0005
 → x30FB  0111010001001110  x744E              STR    R2, R1 x000E
 ▪ x30FC  1010011011110100  xA6F4              LDI    R3, x30F4
 ▪ x30FD  0000000000000000  x0000
 ▪ x30FE  0000000000000000  x0000
 ▪ x30FF  0000000000000000  x0000
 ▪ x3100  0000000000000000  x0000
 ▪ x3101  0000000000000000  x0000
 ▪ x3102  0000000000000000  x0000
```

Dopo istruzione 6

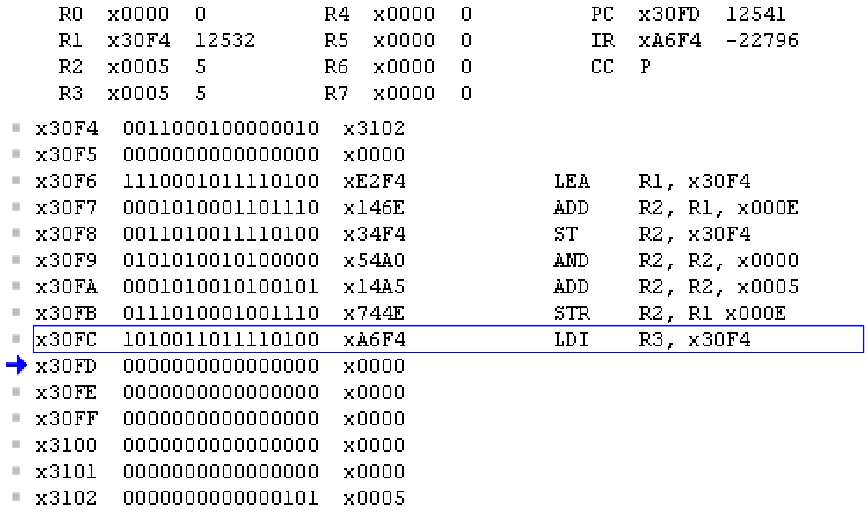
STR R2, R1, x000E ;scrive R2 in M(R1+14=3102)



```
R0  x0000  0          R4  x0000  0         PC  x30FC  12540
R1  x30F4  12532      R5  x0000  0         IR  x744E  29774
R2  x0005  5          R6  x0000  0         CC  P
R3  x0000  0          R7  x0000  0

▪ x30F4  0011000100000010  x3102
▪ x30F5  0000000000000000  x0000
▪ x30F6  1110001011110100  xE2F4             LEA   R1, x30F4
▪ x30F7  0001010001101110  x146E             ADD   R2, R1, x000E
▪ x30F8  0011010011110100  x34F4             ST    R2, x30F4
▪ x30F9  0101010010100000  x54A0             AND   R2, R2, x0000
▪ x30FA  0001010010100101  x14A5             ADD   R2, R2, x0005
▪ x30FB  0111010001001110  x744E             STR   R2, R1 x000E
→ x30FC  1010011011110100  xA6F4             LDI   R3, x30F4
▪ x30FD  0000000000000000  x0000
▪ x30FE  0000000000000000  x0000
▪ x30FF  0000000000000000  x0000
▪ x3100  0000000000000000  x0000
▪ x3101  0000000000000000  x0000
▪ x3102  0000000000000101  x0005
```

Dopo istruzione 7

LDI R3, x30F4 ;scrive M(M(30F4))=5 in R3

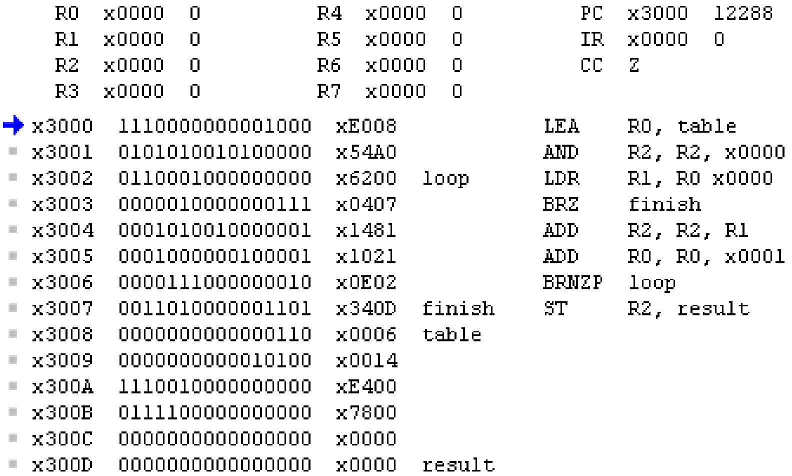


```
R0  x0000  0          R4  x0000  0         PC  x30FD  12541
R1  x30F4  12532      R5  x0000  0         IR  xA6F4  -22796
R2  x0005  5          R6  x0000  0         CC  P
R3  x0005  5          R7  x0000  0

▪ x30F4  0011000100000010  x3102
▪ x30F5  0000000000000000  x0000
▪ x30F6  1110001011110100  xE2F4             LEA   R1, x30F4
▪ x30F7  0001010001101110  x146E             ADD   R2, R1, x000E
▪ x30F8  0011010011110100  x34F4             ST    R2, x30F4
▪ x30F9  0101010010100000  x54A0             AND   R2, R2, x0000
▪ x30FA  0001010010100101  x14A5             ADD   R2, R2, x0005
▪ x30FB  0111010001001110  x744E             STR   R2, R1 x000E
▪ x30FC  1010011011110100  xA6F4             LDI   R3, x30F4
→ x30FD  0000000000000000  x0000
▪ x30FE  0000000000000000  x0000
▪ x30FF  0000000000000000  x0000
▪ x3100  0000000000000000  x0000
▪ x3101  0000000000000000  x0000
▪ x3102  0000000000000101  x0005
```

Esempio sulle istruzioni di controllo

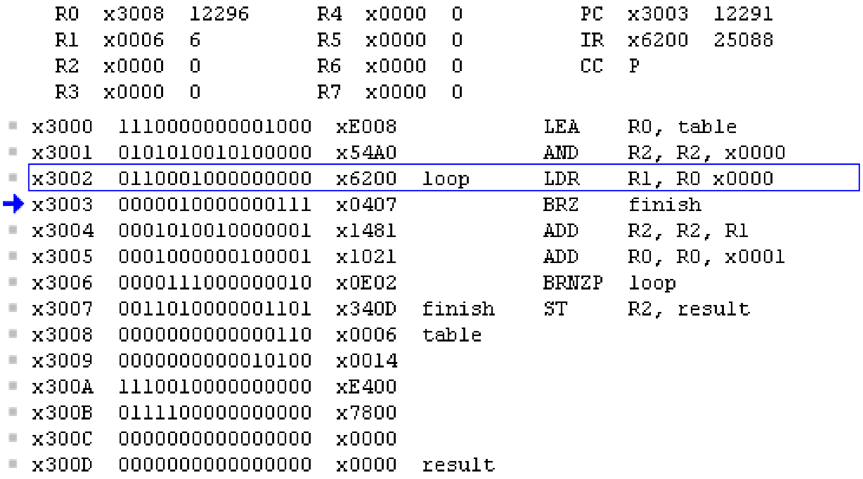
```
;programma che somma un vettore di numeri in memoria
;la somma termina appena si incontra un valore nullo
;programma situato in memoria a partire da x3000
;vettore situato in memoria a partire da x3008

        LEA     R0, table     ;carica 3008 in puntatore
        AND     R2, R2, #0    ;azzera totalizzatore
loop    LDR     R1, R0, #0    ;legge prossimo numero
        BRZ     finish        ;se nullo ha finito
        ADD     R2, R2, R1    ;somma a totalizzatore
        ADD     R0, R0, #1    ;incrementa puntatore
        BRNZP   loop          ;prossimo numero
finish  ST      R2, result    ;scrive risultato in memoria
table                         ;vettore di numeri
result                        ;risultato
```

CPU e memoria all'inizio del programma



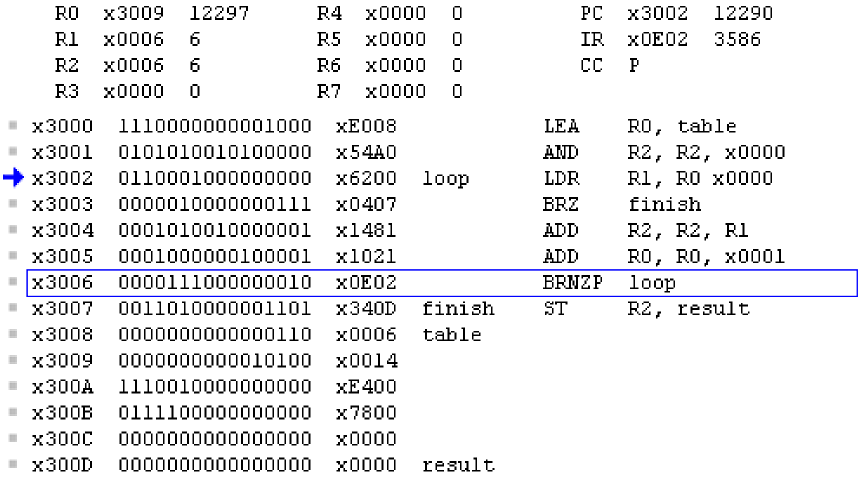
```
    R0  x0000  0         R4  x0000  0         PC  x3000  12288
    R1  x0000  0         R5  x0000  0         IR  x0000  0
    R2  x0000  0         R6  x0000  0         CC  Z
    R3  x0000  0         R7  x0000  0

➜ x3000  1110000000001000  xE008            LEA    R0, table
▪ x3001  0101010010100000  x54A0            AND    R2, R2, x0000
▪ x3002  0110001000000000  x6200  loop      LDR    R1, R0 x0000
▪ x3003  0000010000000111  x0407            BRZ    finish
▪ x3004  0001010010000001  x1481            ADD    R2, R2, R1
▪ x3005  0001000000100001  x1021            ADD    R0, R0, x0001
▪ x3006  0000111000000010  x0E02            BRNZP  loop
▪ x3007  0011010000001101  x340D  finish    ST     R2, result
▪ x3008  0000000000000110  x0006  table
▪ x3009  0000000000010100  x0014
▪ x300A  1110010000000000  xE400
▪ x300B  0111100000000000  x7800
▪ x300C  0000000000000000  x0000
▪ x300D  0000000000000000  x0000  result
```

Dopo la lettura del primo numero



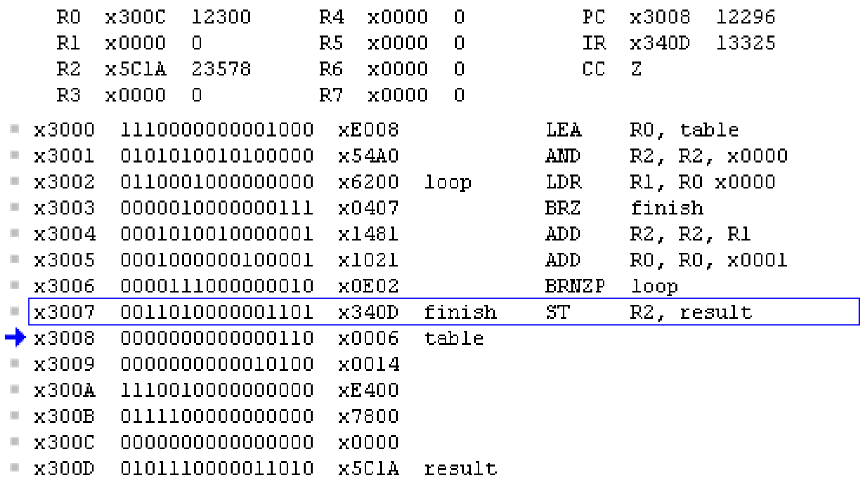
```
     R0   x3008   12296      R4   x0000   0          PC   x3003   12291
     R1   x0006   6          R5   x0000   0          IR   x6200   25088
     R2   x0000   0          R6   x0000   0          CC   P
     R3   x0000   0          R7   x0000   0
▪  x3000   1110000000001000   xE008             LEA     R0, table
▪  x3001   0101010010100000   x54A0             AND     R2, R2, x0000
▪  x3002   0110001000000000   x6200   loop      LDR     R1, R0 x0000
→  x3003   0000010000000111   x0407             BRZ     finish
▪  x3004   0001010010000001   x1481             ADD     R2, R2, R1
▪  x3005   0001000000100001   x1021             ADD     R0, R0, x0001
▪  x3006   0000111000000010   x0E02             BRNZP   loop
▪  x3007   0011010000001101   x340D   finish    ST      R2, result
▪  x3008   0000000000000110   x0006   table
▪  x3009   0000000000010100   x0014
▪  x300A   1110010000000000   xE400
▪  x300B   0111100000000000   x7800
▪  x300C   0000000000000000   x0000
▪  x300D   0000000000000000   x0000   result
```

Dopo la prima esecuzione del ciclo



```
     R0   x3009   12297      R4   x0000   0          PC   x3002   12290
     R1   x0006   6          R5   x0000   0          IR   x0E02   3586
     R2   x0006   6          R6   x0000   0          CC   P
     R3   x0000   0          R7   x0000   0
▪  x3000   1110000000001000   xE008             LEA     R0, table
▪  x3001   0101010010100000   x54A0             AND     R2, R2, x0000
→  x3002   0110001000000000   x6200   loop      LDR     R1, R0 x0000
▪  x3003   0000010000000111   x0407             BRZ     finish
▪  x3004   0001010010000001   x1481             ADD     R2, R2, R1
▪  x3005   0001000000100001   x1021             ADD     R0, R0, x0001
▪  x3006   0000111000000010   x0E02             BRNZP   loop
▪  x3007   0011010000001101   x340D   finish    ST      R2, result
▪  x3008   0000000000000110   x0006   table
▪  x3009   0000000000010100   x0014
▪  x300A   1110010000000000   xE400
▪  x300B   0111100000000000   x7800
▪  x300C   0000000000000000   x0000
▪  x300D   0000000000000000   x0000   result
```

Termine del programma

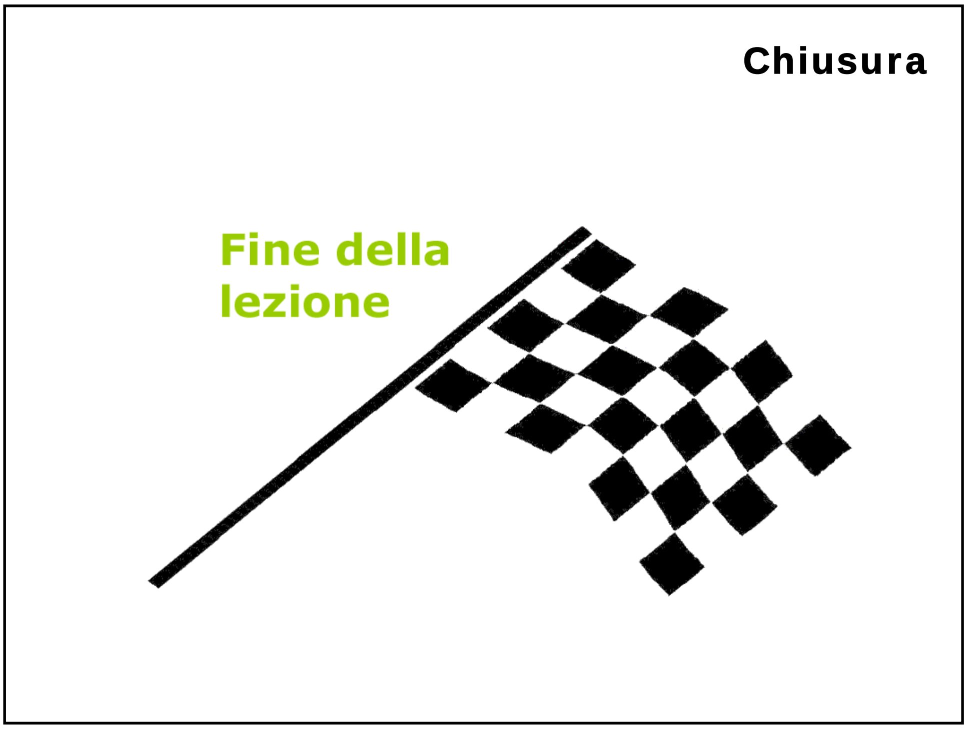


```
   R0   x300C   12300      R4   x0000   0          PC   x3008   12296
   R1   x0000   0          R5   x0000   0          IR   x340D   13325
   R2   x5C1A   23578      R6   x0000   0          CC   Z
   R3   x0000   0          R7   x0000   0

   x3000  1110000000001000  xE008           LEA    R0, table
   x3001  0101010010100000  x54A0           AND    R2, R2, x0000
   x3002  0110001000000000  x6200  loop     LDR    R1, R0 x0000
   x3003  0000010000000111  x0407           BRZ    finish
   x3004  0001010010000001  x1481           ADD    R2, R2, R1
   x3005  0001000000100001  x1021           ADD    R0, R0, x0001
   x3006  0000111000000010  x0E02           BRNZP  loop
   x3007  0011010000001101  x340D  finish   ST     R2, result
 → x3008  0000000000000110  x0006  table
   x3009  0000000000010100  x0014
   x300A  1110010000000000  xE400
   x300B  0111100000000000  x7800
   x300C  0000000000000000  x0000
   x300D  0101110000011010  x5C1A  result
```

In sintesi...

- Abbiamo visto due esempi di esecuzione di programmi in linguaggio macchina.
- Il comportamento dei due programmi è stato discusso mostrando il contenuto della CPU LC-2 e della sua memoria di lavoro.
- Abbiamo dato una rappresentazione simbolica dei programmi (con i nomi delle istruzioni LC-2 invece della loro codifica binaria).
- Abbiamo adottato sia un'analisi passo passo sia un'analisi per blocchi di istruzioni.

Vedremo gli strumenti software
che ci consentono queste attività.



Chiusura

Fine della lezione