SOTTOPROGRAMMI E MECCANISMI PER IL COLLEGAMENTO

CALCOLATORI ELETTRONICI I

CdL Ingegneria Biomedica (A-I)

DIS - Università degli Studi di Napoli “Federico II”

Procedura o subroutine

Problematiche da affrontare:

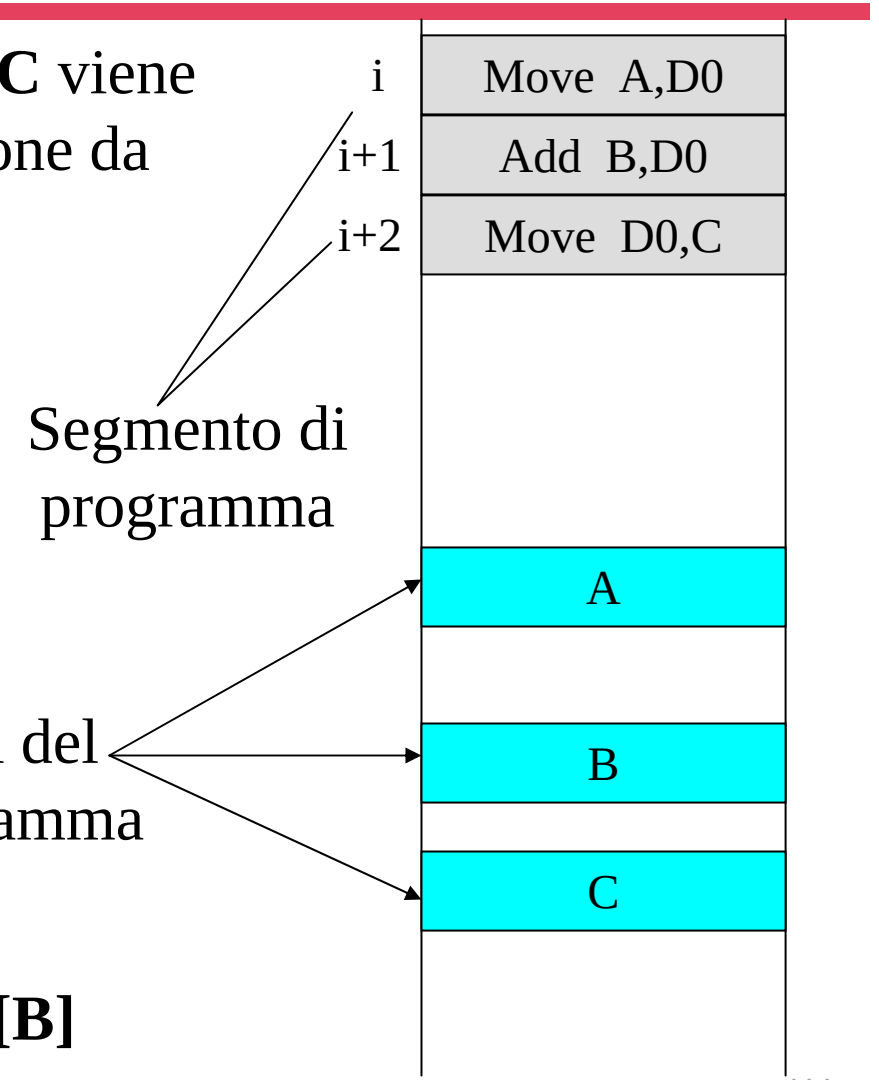
Collegamento (o ***Linkage***)**:** individuazione delle istruzioni da eseguire per realizzare le operazioni di ***chiamata*** e di ***ritorno*** delle ***procedure***

Passaggio dei Parametri**:** individuazione delle istruzioni da eseguire per realizzare il ***passaggio delle informazioni*** dal ***modulo chiamante*** alla ***procedure*** e viceversa

Esecuzione in sequenza lineare

Prelevata l'istruzione da eseguire, il registro **PC** viene incrementato per puntare alla prossima istruzione da eseguire:

```
while (TRUE){
    Fetch;
    Operand Assembly;
    Execute
                }
```

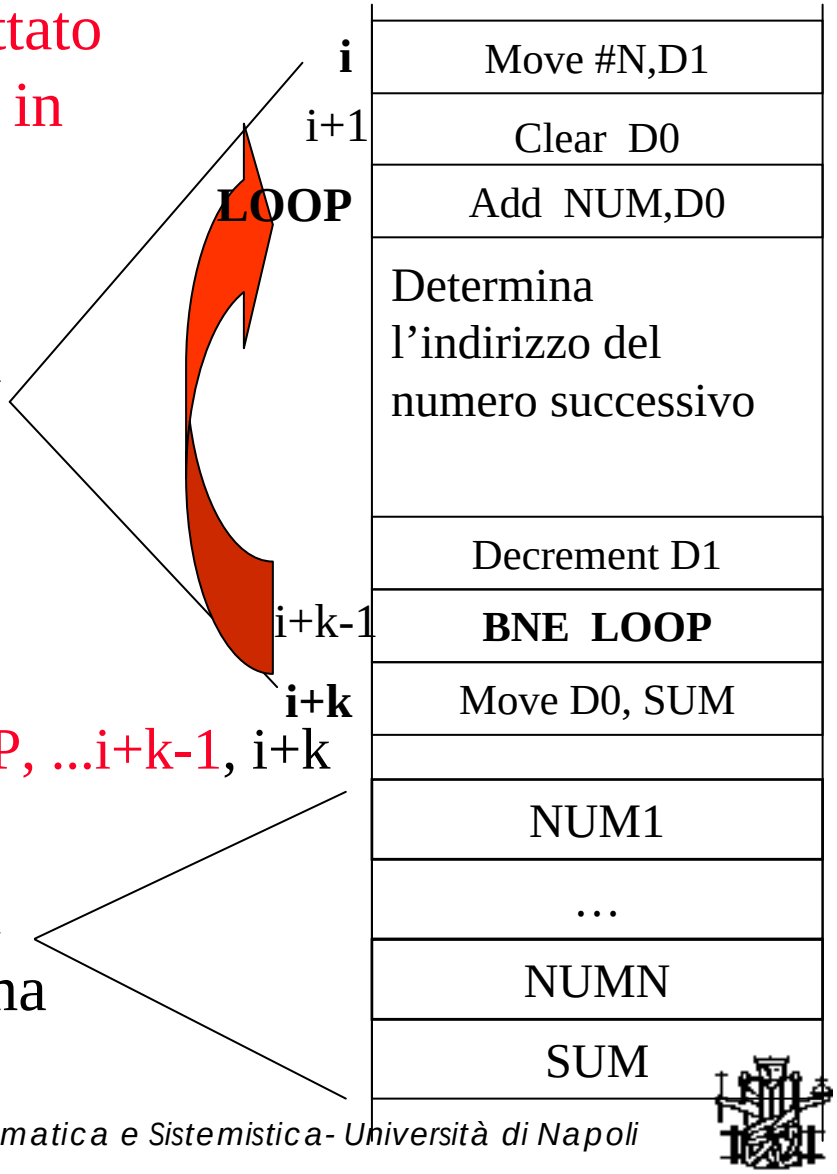


Esempio somma di due numeri: **C ⟵ [A] + [B]**

Istruzione di salto

Con le istruzioni di salto il registro **PC** viene settato (ad esempio in base al risultato di un confronto) in modo da puntare ad una specifica istruzione.

Es: somma di una lista di numeri



PC= i, i+1, LOOP, ...i+k-1, LOOP, ...i+k-1,…, LOOP, ...i+k-1, i+k

Linkage mediante istruzioni di salto

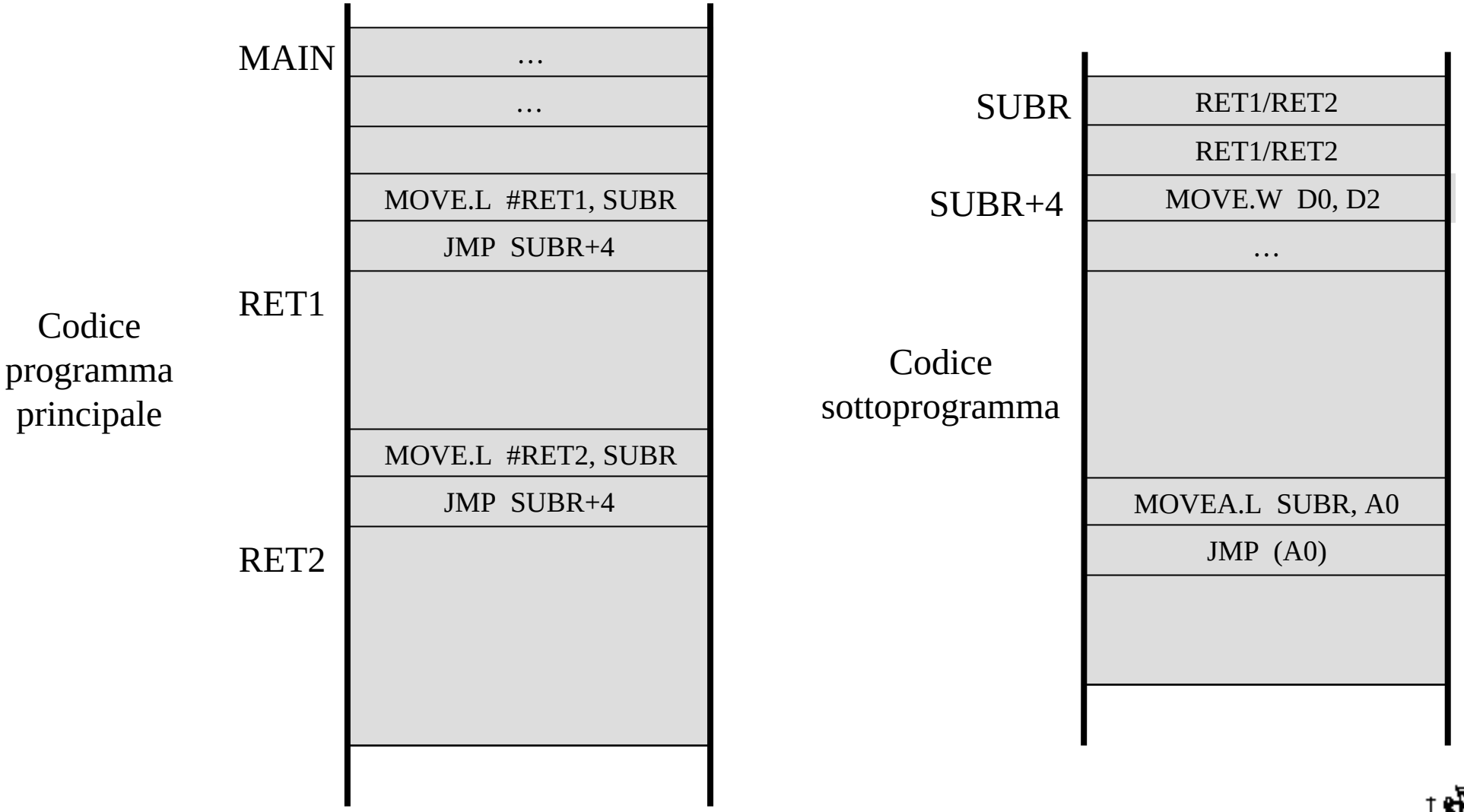
```
            ORG         $8000
MAIN        MOVE.L      #RET1,SUBR   Salva l'indirizzo di ritorno
            JMP         SUBR+4       Salta alla subroutine
RET1        NOP                      Indirizzo di ritorno dalla chiamata
            MOVE.L      #RET2,SUBR   Salva l'indirizzo di ritorno
            JMP         SUBR+4       Salta alla subroutine
RET2        NOP                      Indirizzo di ritorno dalla chiamata

SUBR        DS.L        1            Riserva una LongWord per ind.ritorno
            MOVE.W      D0,D2        Prima istruzione della procedura
            MOVEA.L     SUBR,A0      Carica A0 con l'indirizzo di ritorno
            JMP (A0)                 Salta all'indirizzo contenuto in A0
```

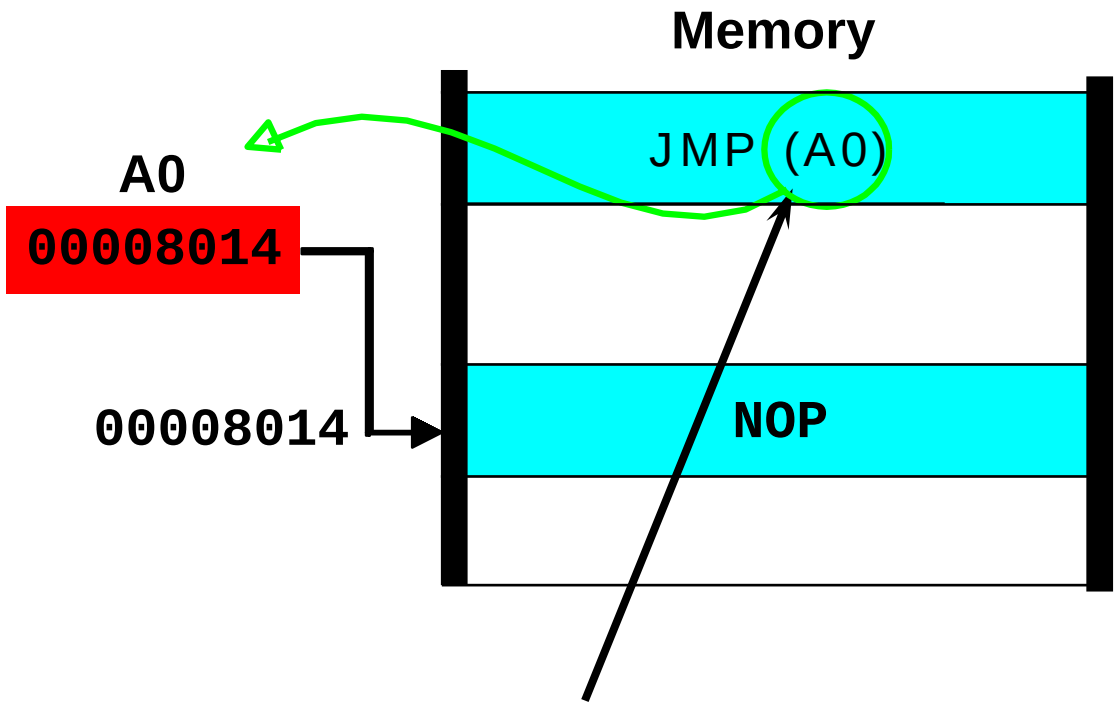
sottoprogramma

Realizza una chiamata a sottoprogrammi utilizzando le istruzioni di salto previste dal set di istruzioni del processore

Mappa della memoria



Ritorno - Schema



Questa istruzione significa: salta all'istruzione puntata dal registro indirizzo A0

L'istruzione specifica l'operando come (A0)

Linkage mediante istruzioni di salto

Con questa soluzione

- non occorre
  - * definire istruzioni nuove
  - * dedicare risorse hardware

- non è possibile
  - *memorizzare la subroutine in una ROM (non è possibile memorizzare l'indirizzo di ritorno)
  - *effettuare chiamate ricorsive (conserva un solo indirizzo di ritorno)

Linkage mediante registro

- Il processore è dotato di apposite istruzioni:
  - **JSR** (***Jump to Subroutine***)
  - **RTS** (***Return from Subroutine***)

- L'indirizzo di ritorno è salvato in un apposito registro (***registro di collegamento*** o ***link register***)

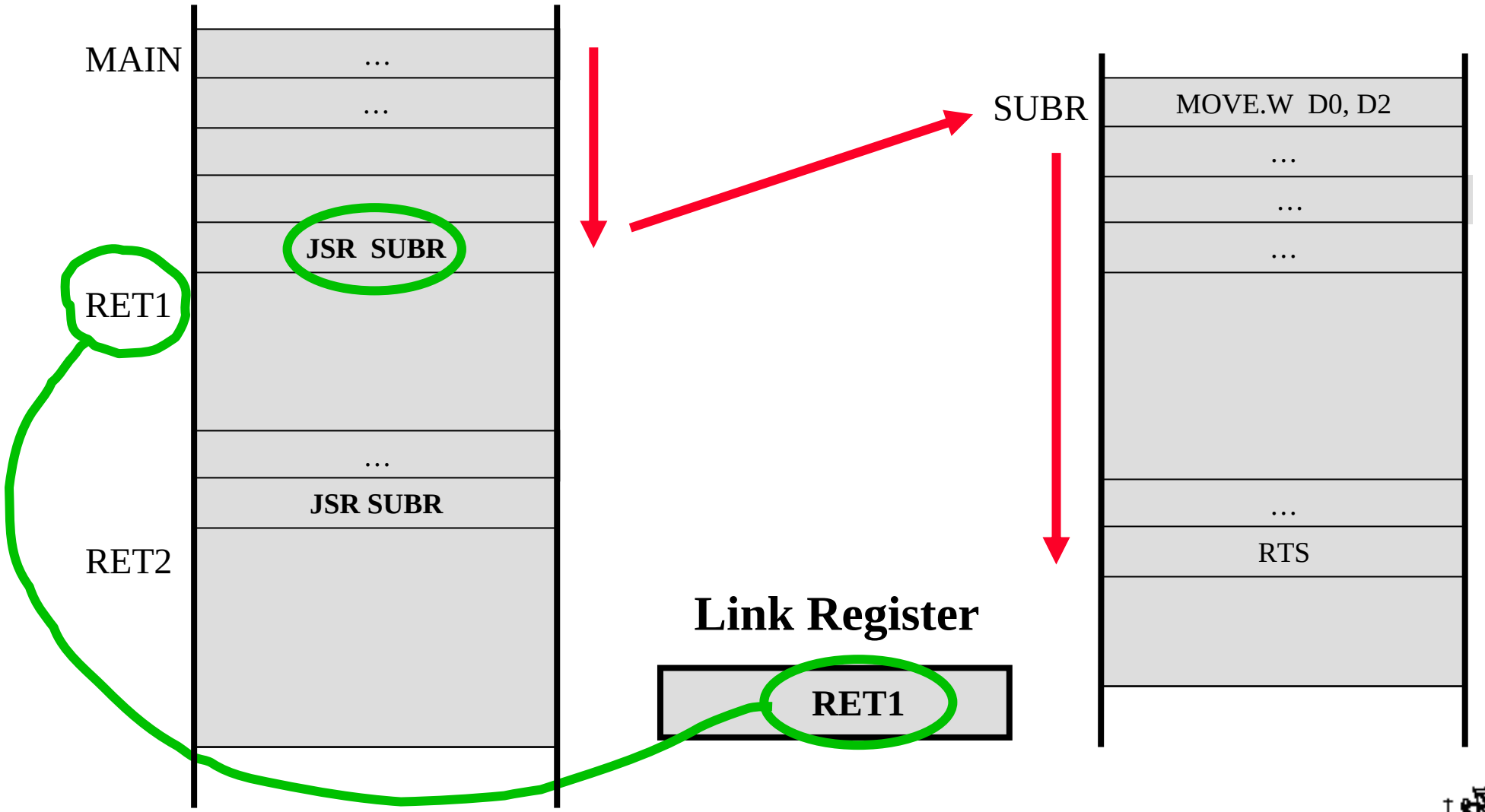
L'uso di **procedure annidate** è a carico del programmatore

Esempio

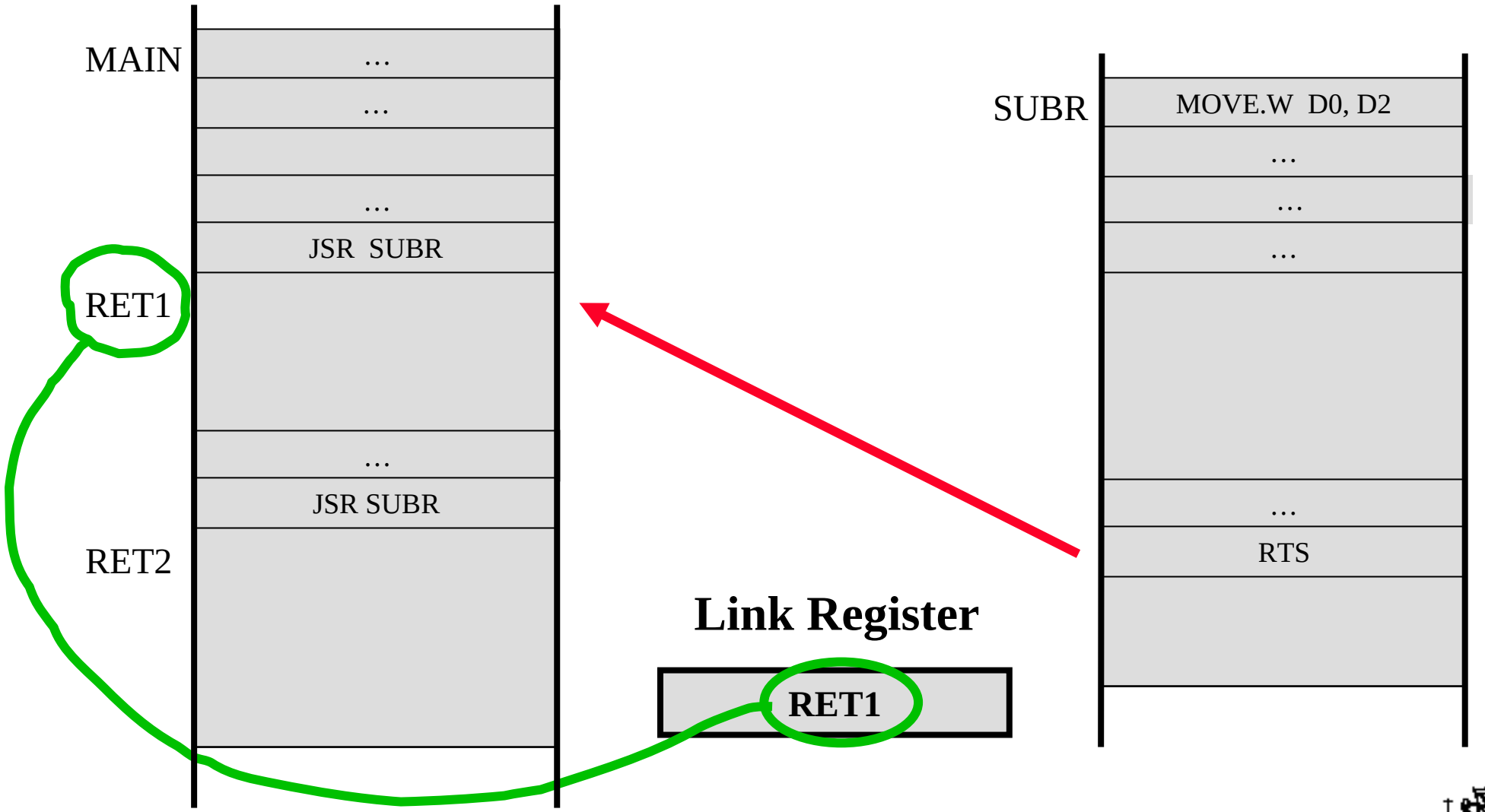
```
MAIN        … Inizio programma principale
…
JSR SUBR    Salva l'indirizzo di ritorno nel LR e salta alla sub
…
JSR SUBR    Salva l'indirizzo di ritorno nel LR e salta alla sub
…


SUBR    MOVE.W D0,D2        Prima istruzione della procedura
…
RTS     Salta all'indirizzo di ritorno contenuto in LR
```

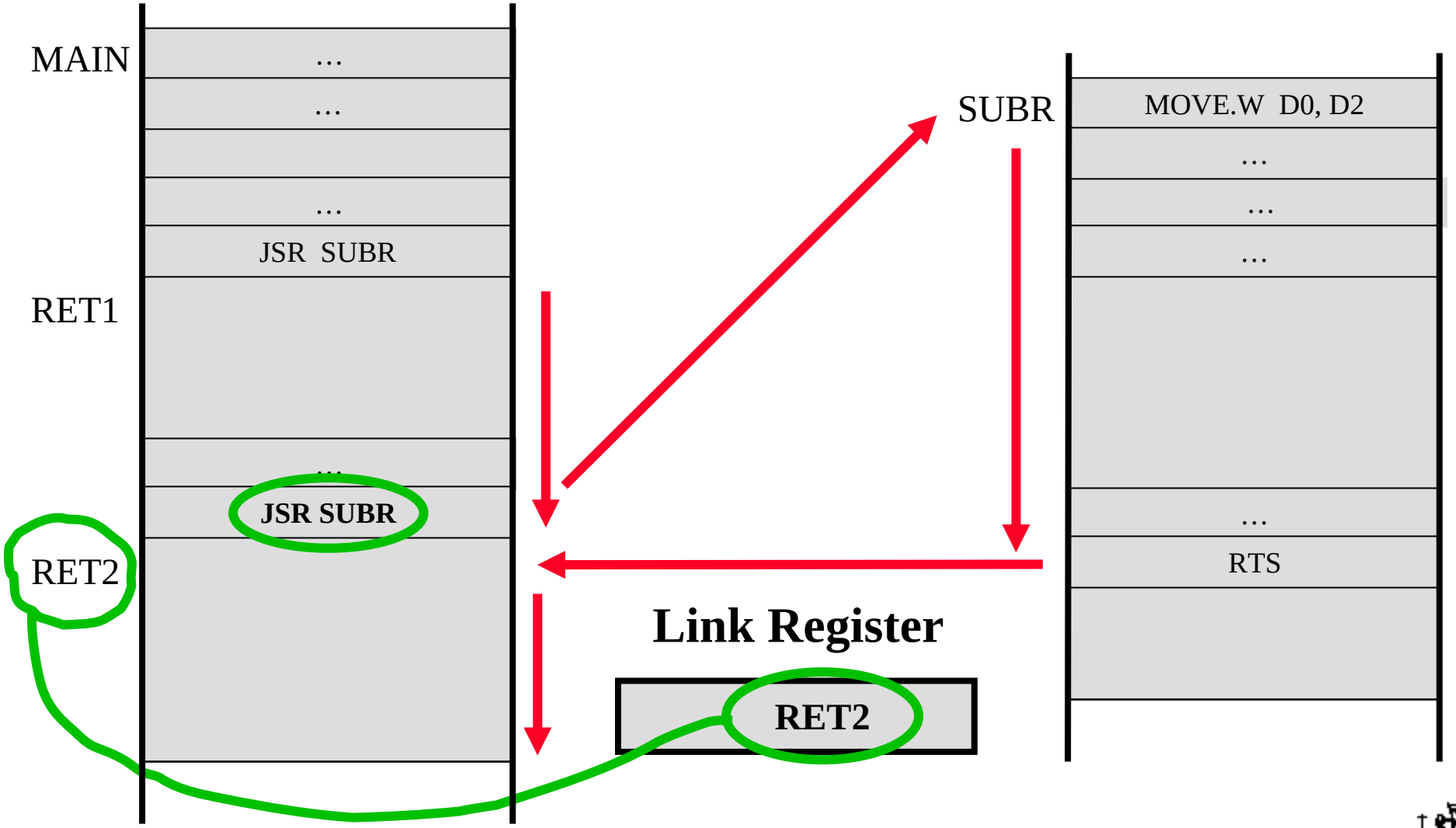
Mappa della memoria



Mappa della memoria



Mappa della memoria



Linkage mediante registro

Differisce dal collegamento con istruzioni di salto, perché l'indirizzo di ritorno viene salvato nel ***Link Register*** anziché nella locazione di memoria SUBR

- Vantaggi:
  - È possibile memorizzare la subroutine anche in una ROM
  - È veloce
  - Richiede poche risorse hardware

- Svantaggi:
  - Non è possibile effettuare chiamate ricorsive (le chiamate ricorsive devono essere gestite esplicitamente dal programmatore)

Linkage con gestione esplicita dello stack

Gli indirizzi di ritorno vengono salvati in una area stack ed un registro particolare, lo Stack Pointer, punta all'ultima posizione occupata della pila

La subroutine preleva dallo stack l'indirizzo di ritorno ponendolo in un registro A, che viene usato per effettuare il ritorno al chiamante (indrizz. indiretto tramite registro)

L'indirizzo di ritorno viene salvato nello stack dal programma chiamante ***prima*** di saltare alla subroutine

```
* Subroutine
                ORG   $8000
SUBR            MOVE.W  D0,D2
                MOVEA.L (SP)+,A0
                JMP     (A0)

* Main Program
                ORG   $8010
START           MOVE.L #RET1,-(SP)
                JMP     SUBR
RET1            NOP

                MOVE.L #RET2,-(SP)
                JMP     SUBR
RET2            NOP
                END    START
```

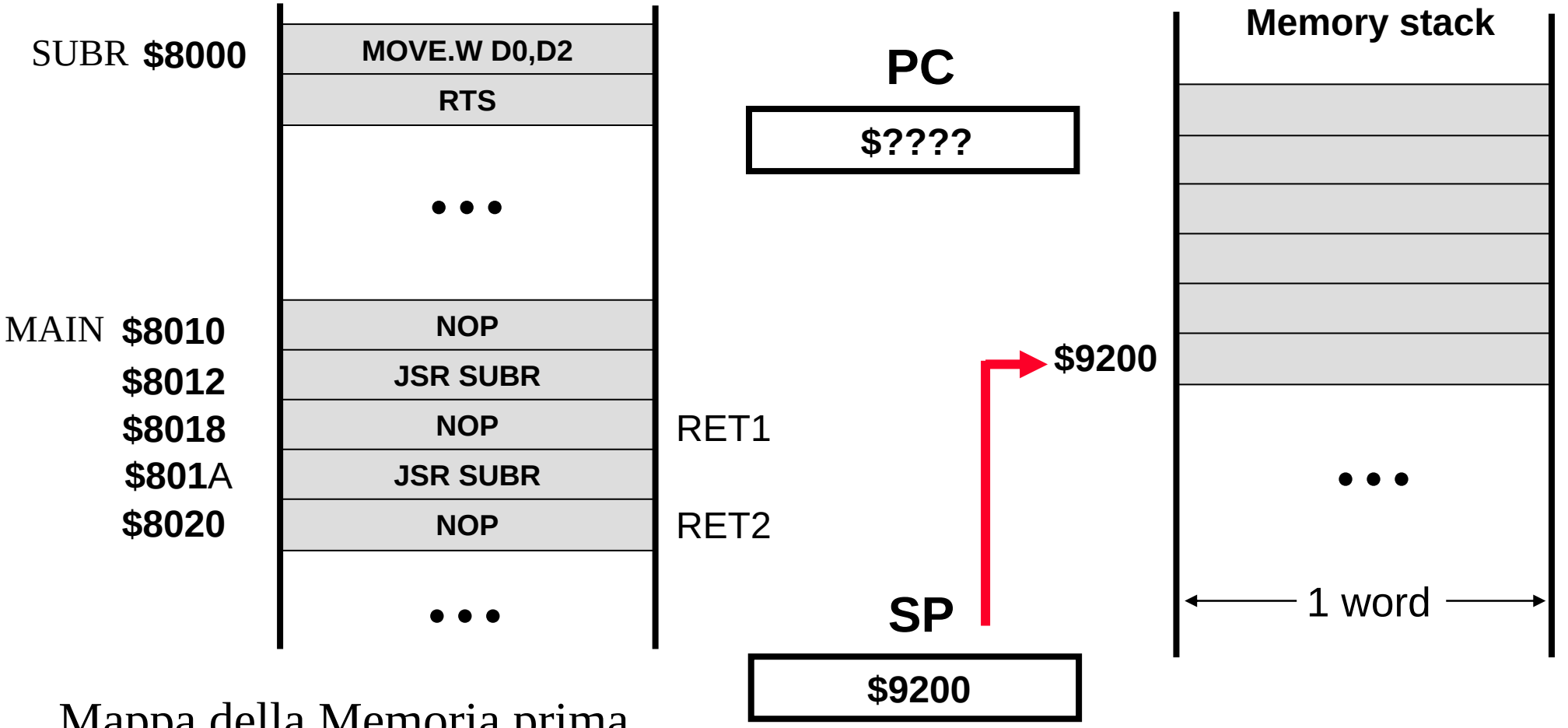
Linkage con gestione implicita (istruzioni dedicate) dello stack

Il processore viene dotato di apposite istruzioni:

- **JSR**: salva automaticamente nello stack l'indirizzo di ritorno
- **RTS**: ripristina automaticamente nel PC l'indirizzo di ritorno presente nello statck

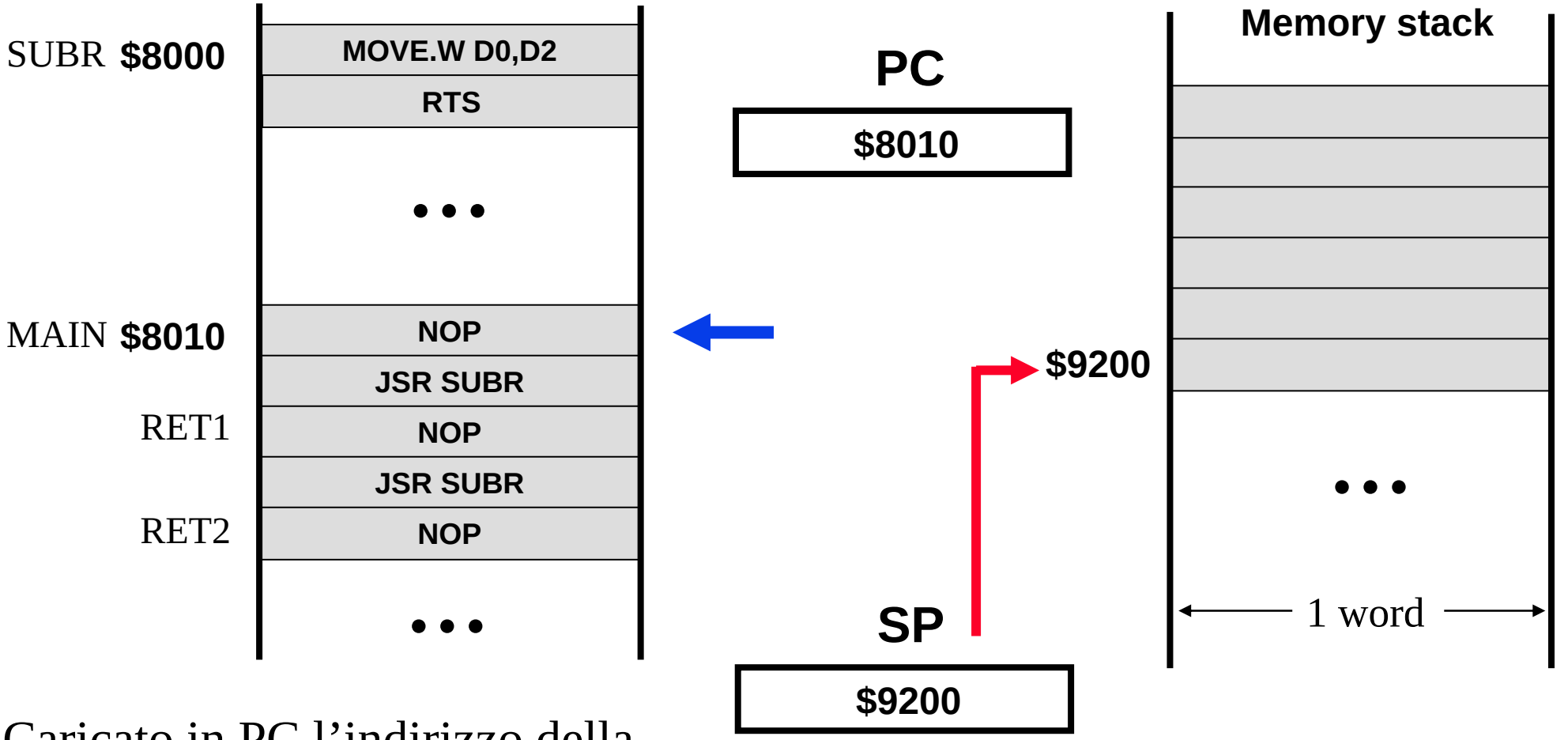
```
*  Subroutine
                ORG $8000
SUBR            MOVE.W  D0,D2
                RTS
* Main Program
                ORG $8010
START           NOP
                JSR   SUBR
                NOP
                JSR   SUBR
                NOP
                END  START
```

Esempio - Mappa della memoria e stack



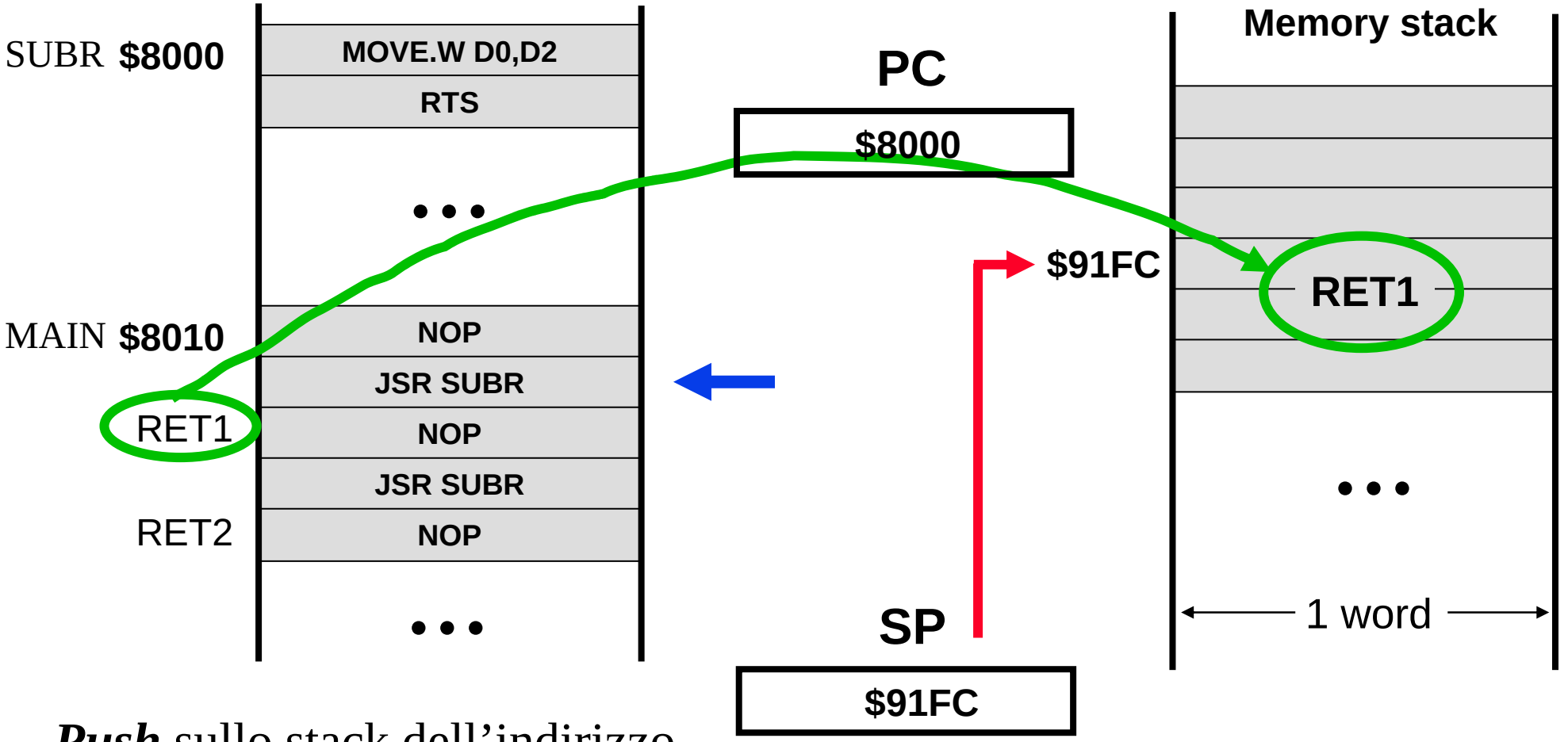
Mappa della Memoria prima dell'esecuzione del programma

Esempio - Mappa della memoria e stack



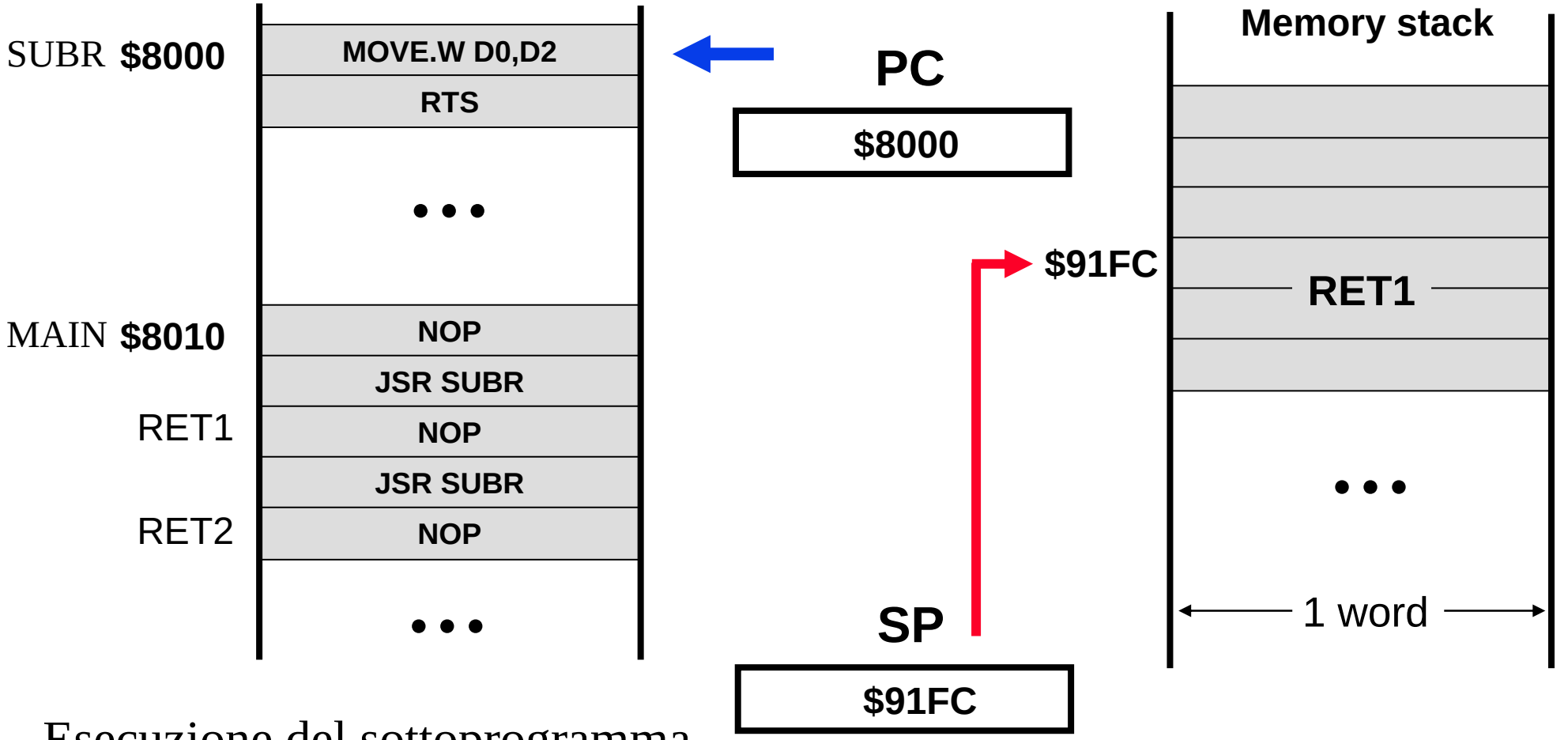
Caricato in PC l'indirizzo della prima Istruzione del programma, inizia l'esecuzione

Esempio - Mappa della memoria e stack



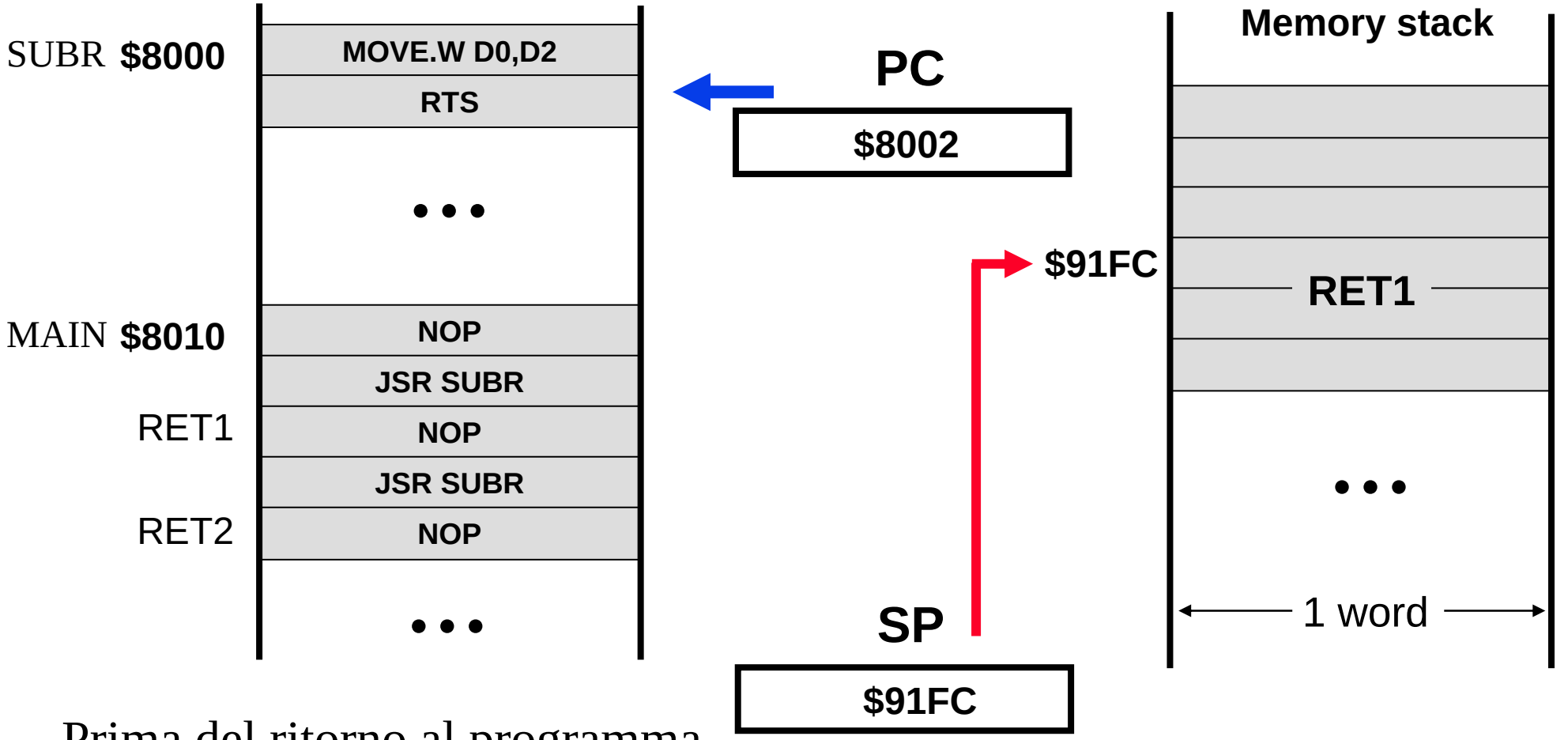
Push sullo stack dell'indirizzo di ritorno

Esempio - Mappa della memoria e stack



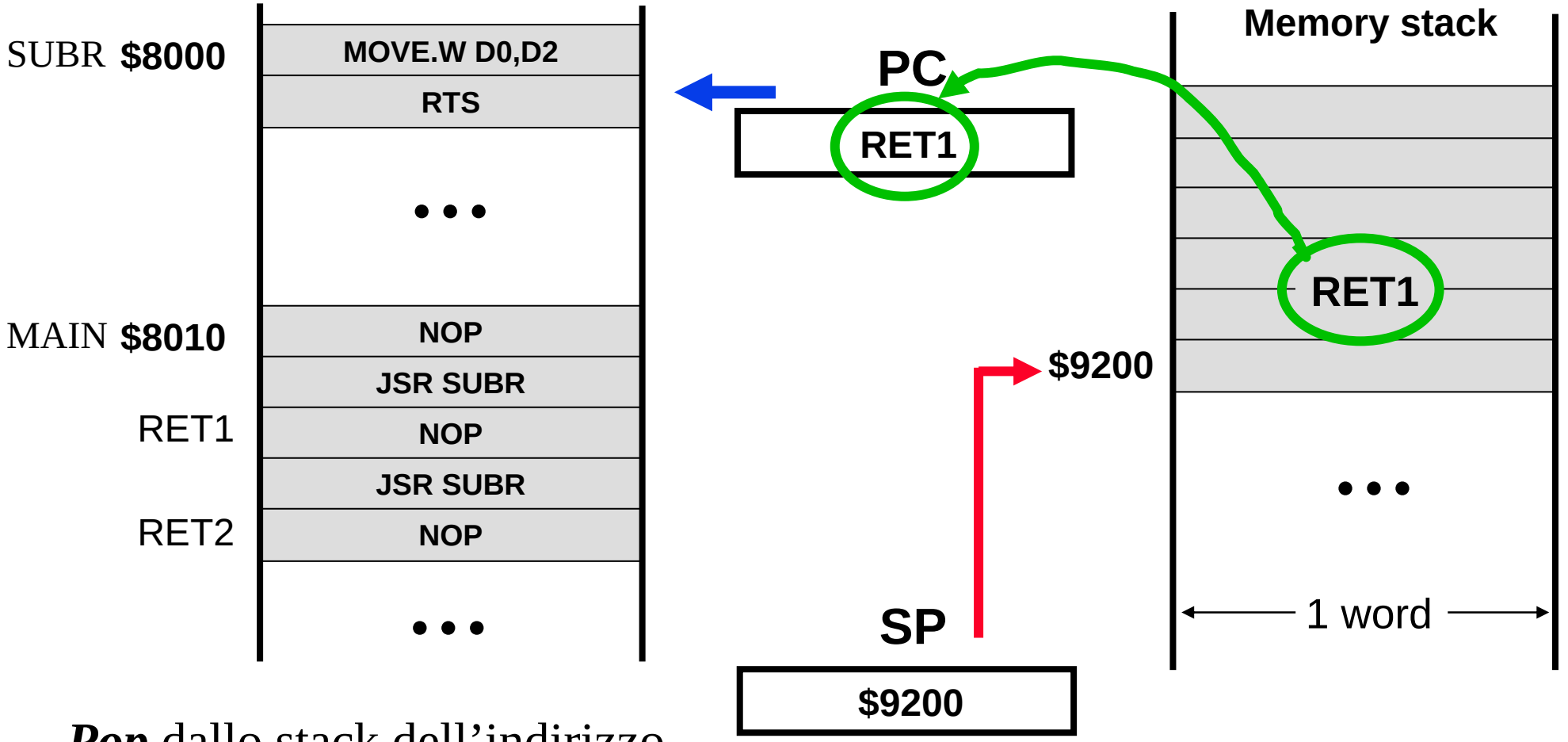
Esecuzione del sottoprogramma

Esempio - Mappa della memoria e stack



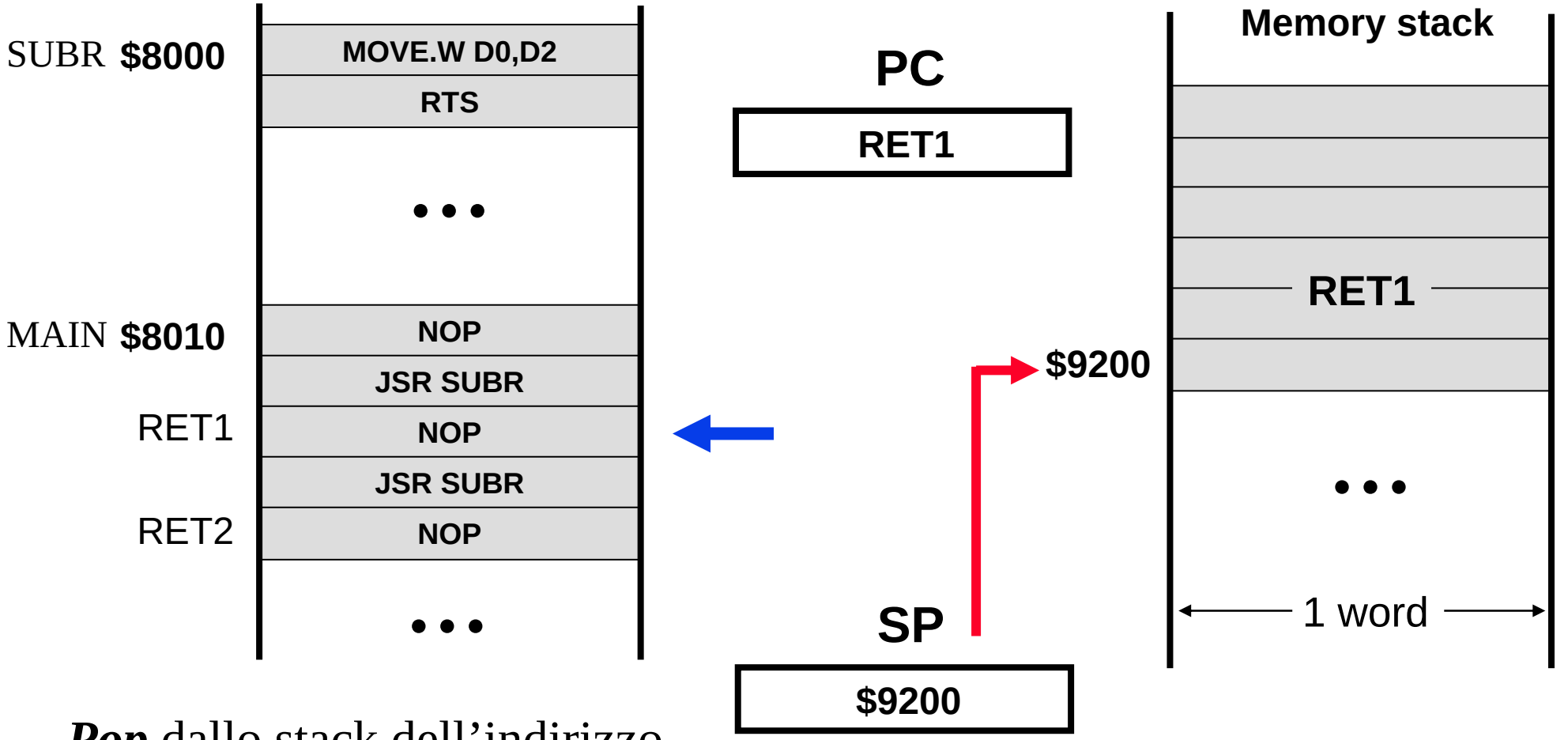
Prima del ritorno al programma chiamante

Esempio - Mappa della memoria e stack



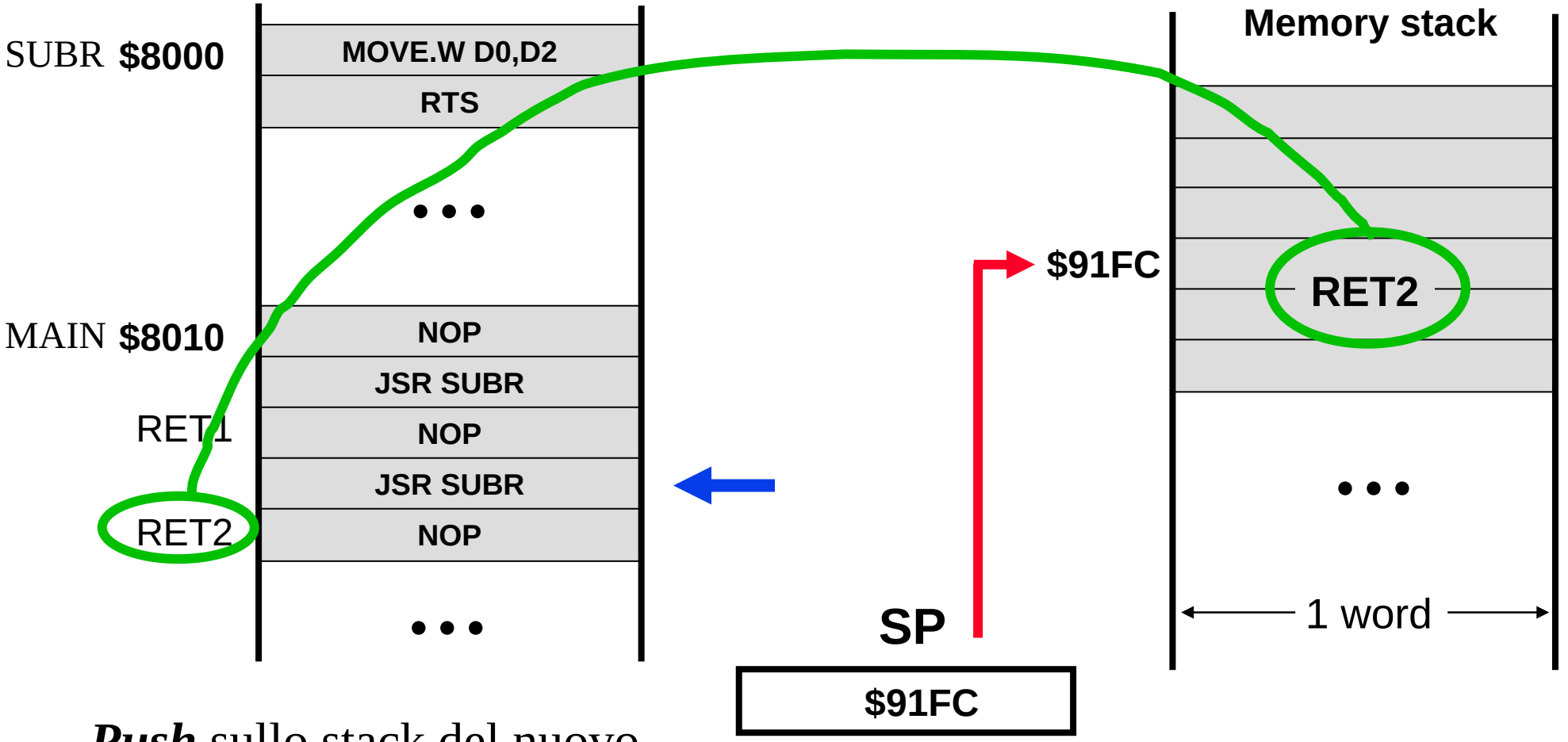
Pop dallo stack dell'indirizzo di ritorno

Esempio - Mappa della memoria e stack



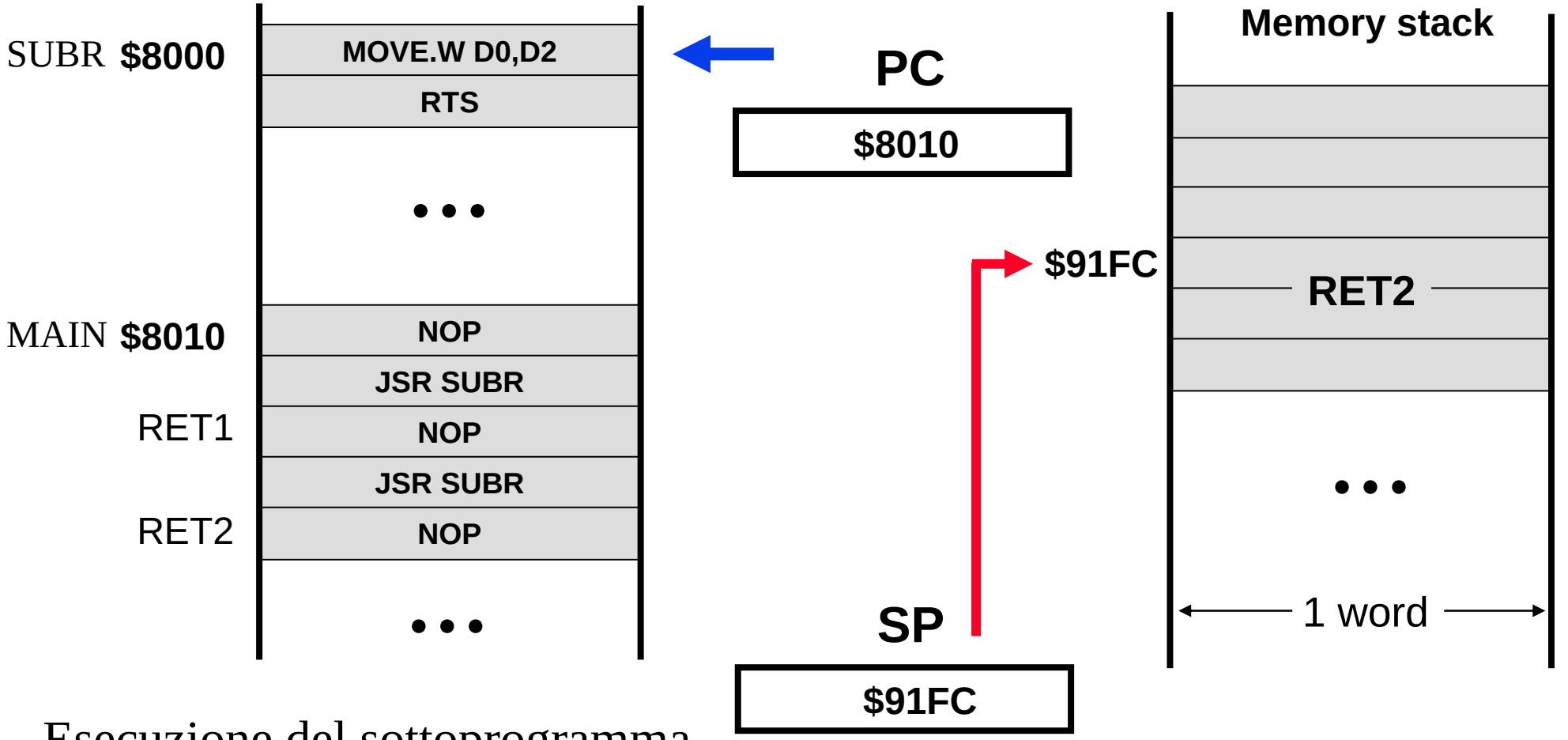
Pop dallo stack dell'indirizzo di ritorno

Esempio - Mappa della memoria e stack



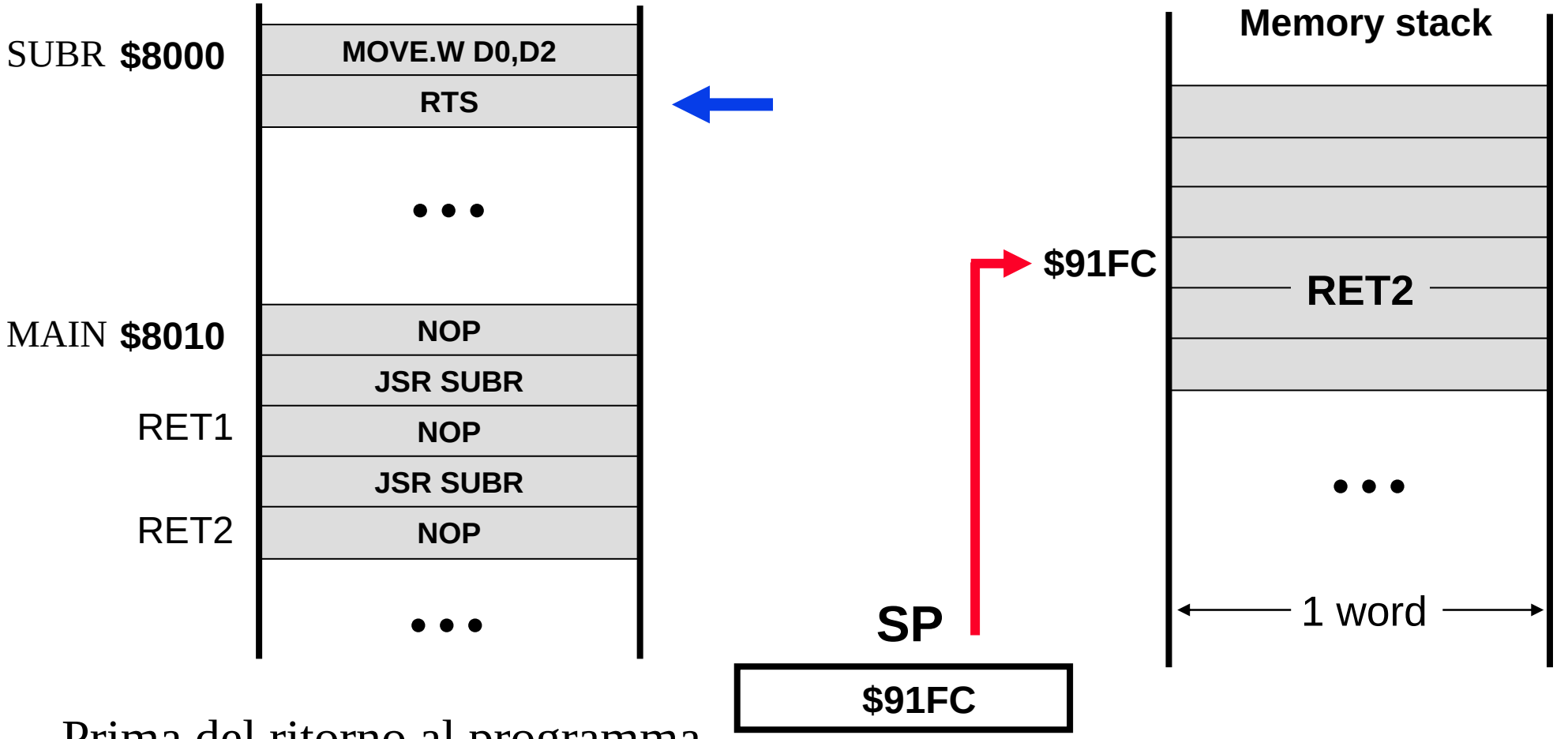
Push sullo stack del nuovo indirizzo di ritorno

Esempio - Mappa della memoria e stack



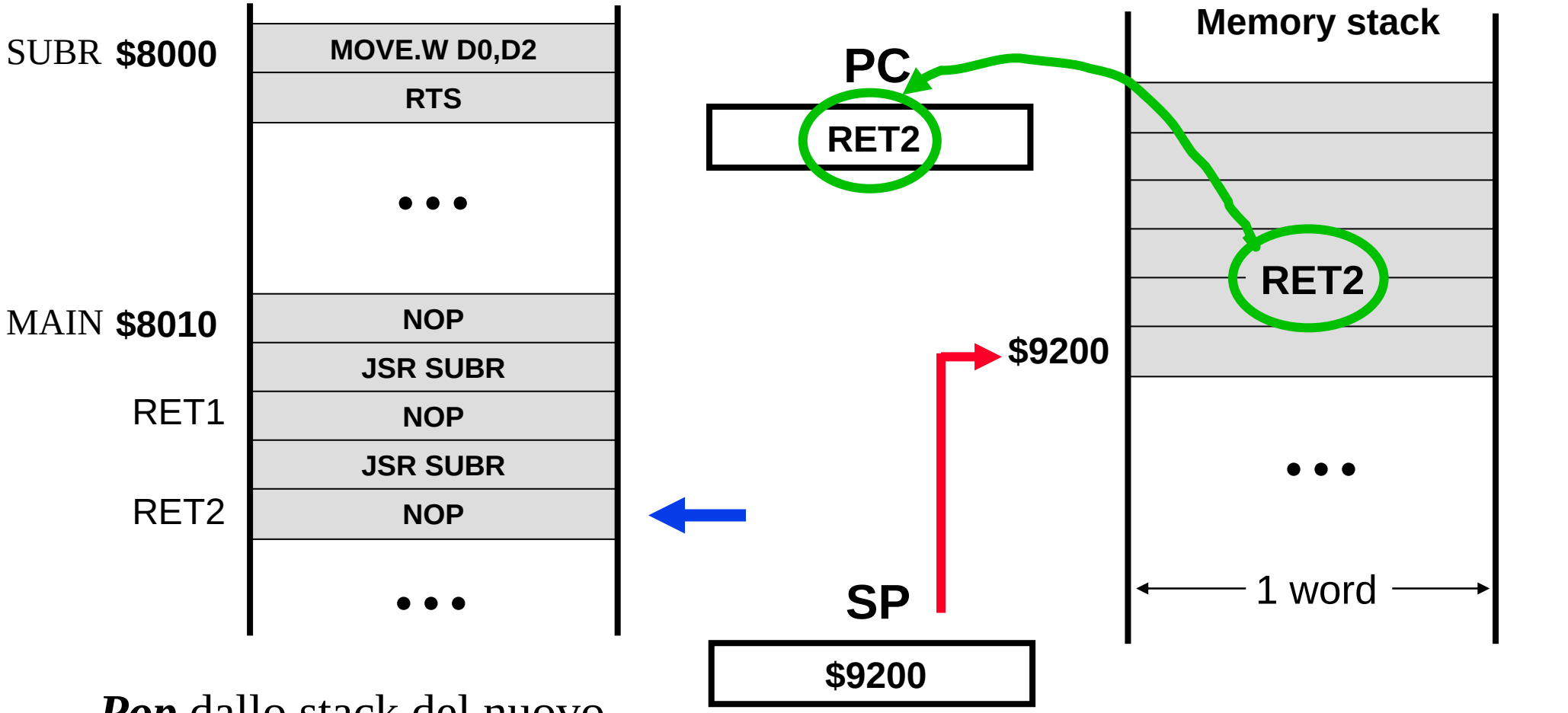
Esecuzione del sottoprogramma

Esempio - Mappa della memoria e stack



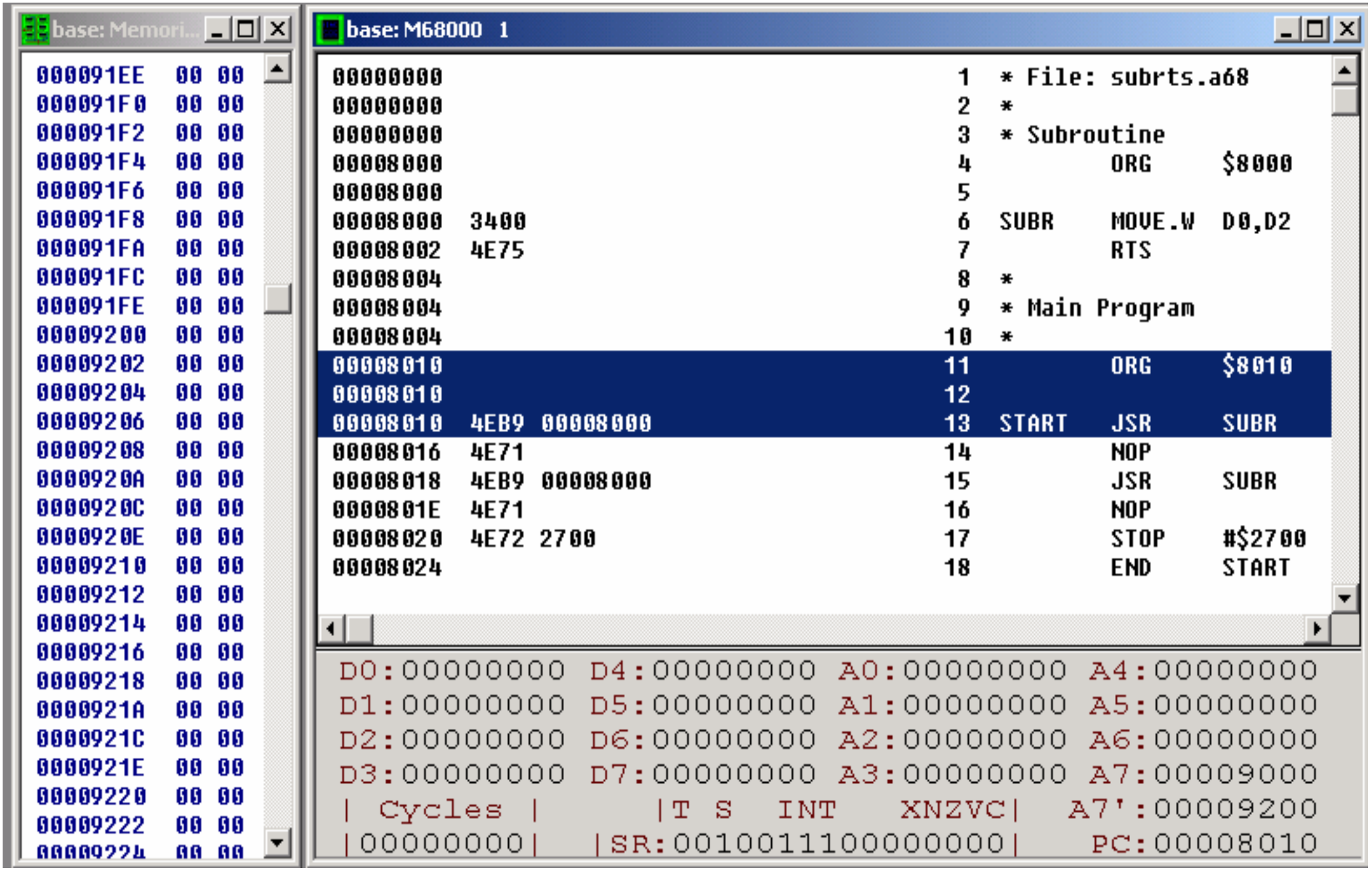
Prima del ritorno al programma chiamante

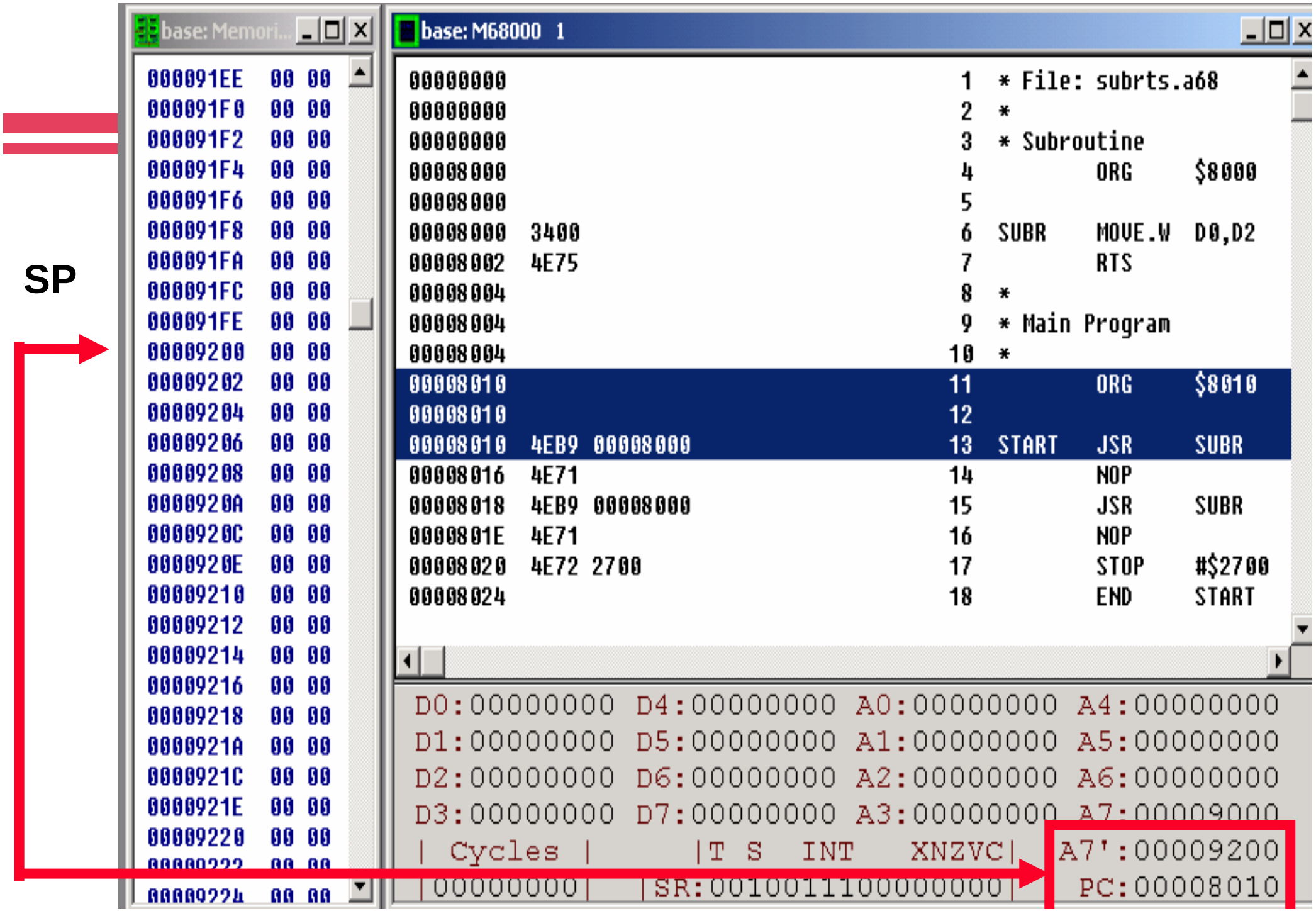
Esempio - Mappa della memoria e stack

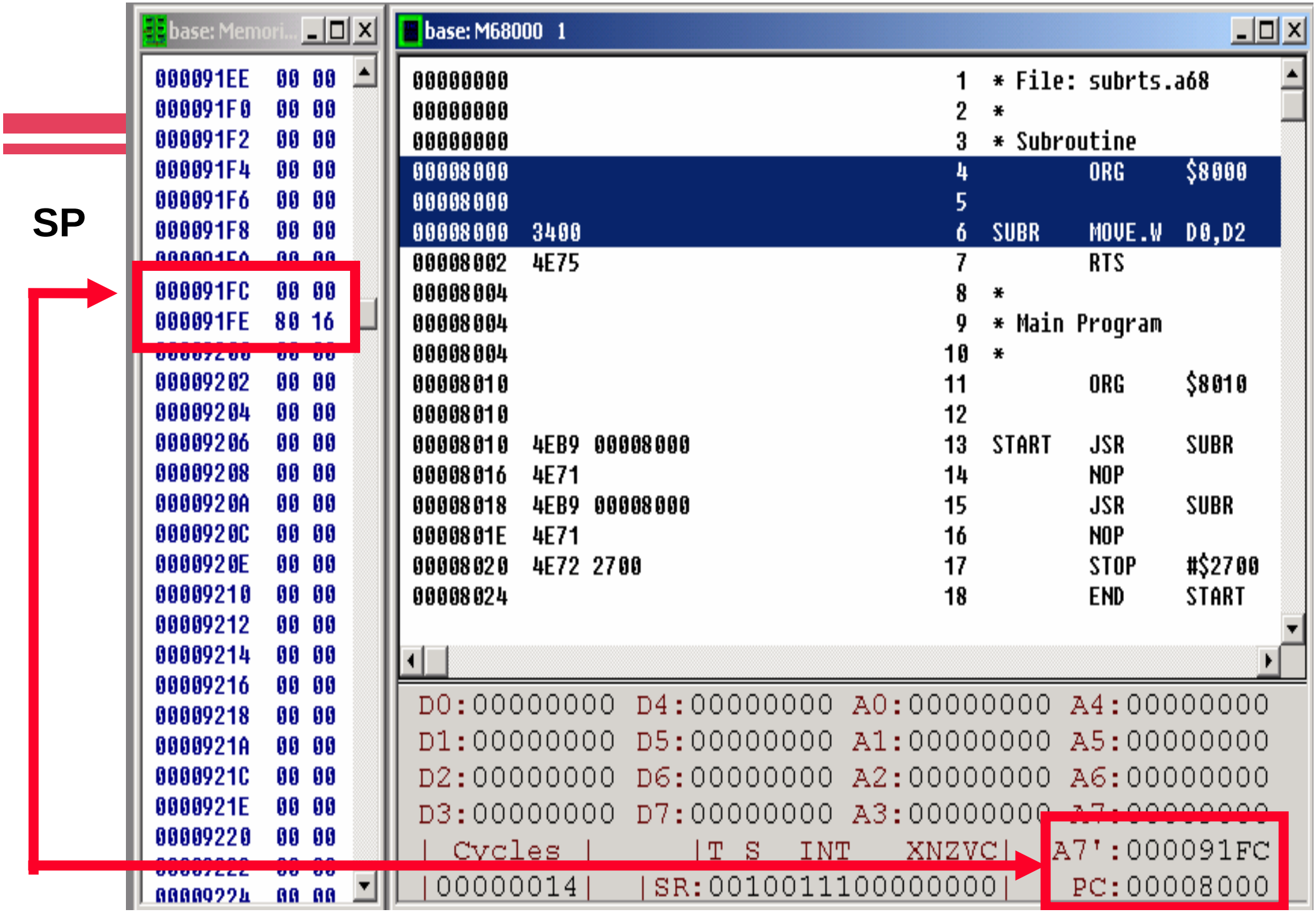


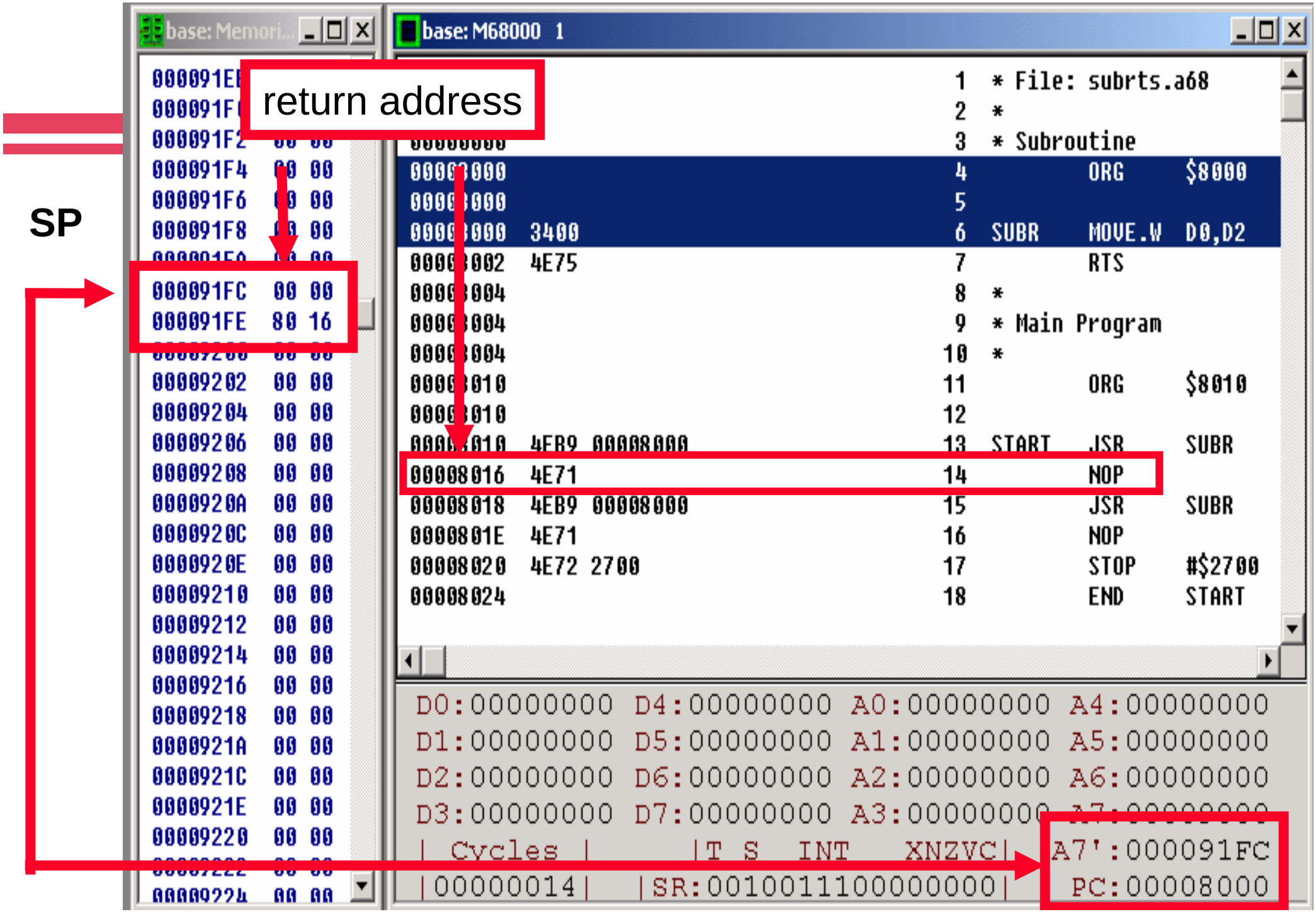
Pop dallo stack del nuovo indirizzo di ritorno

Esempio - esecuzione

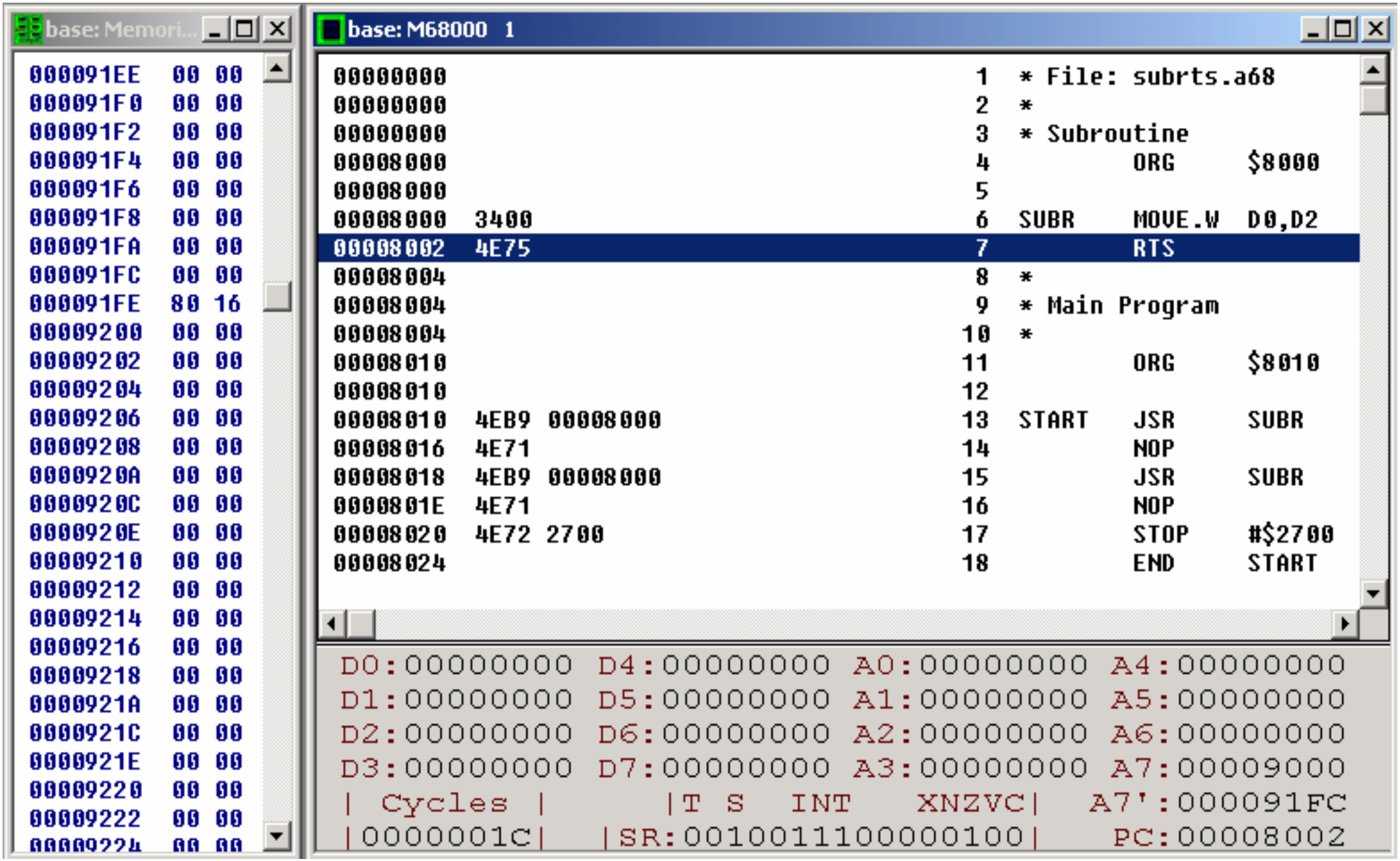


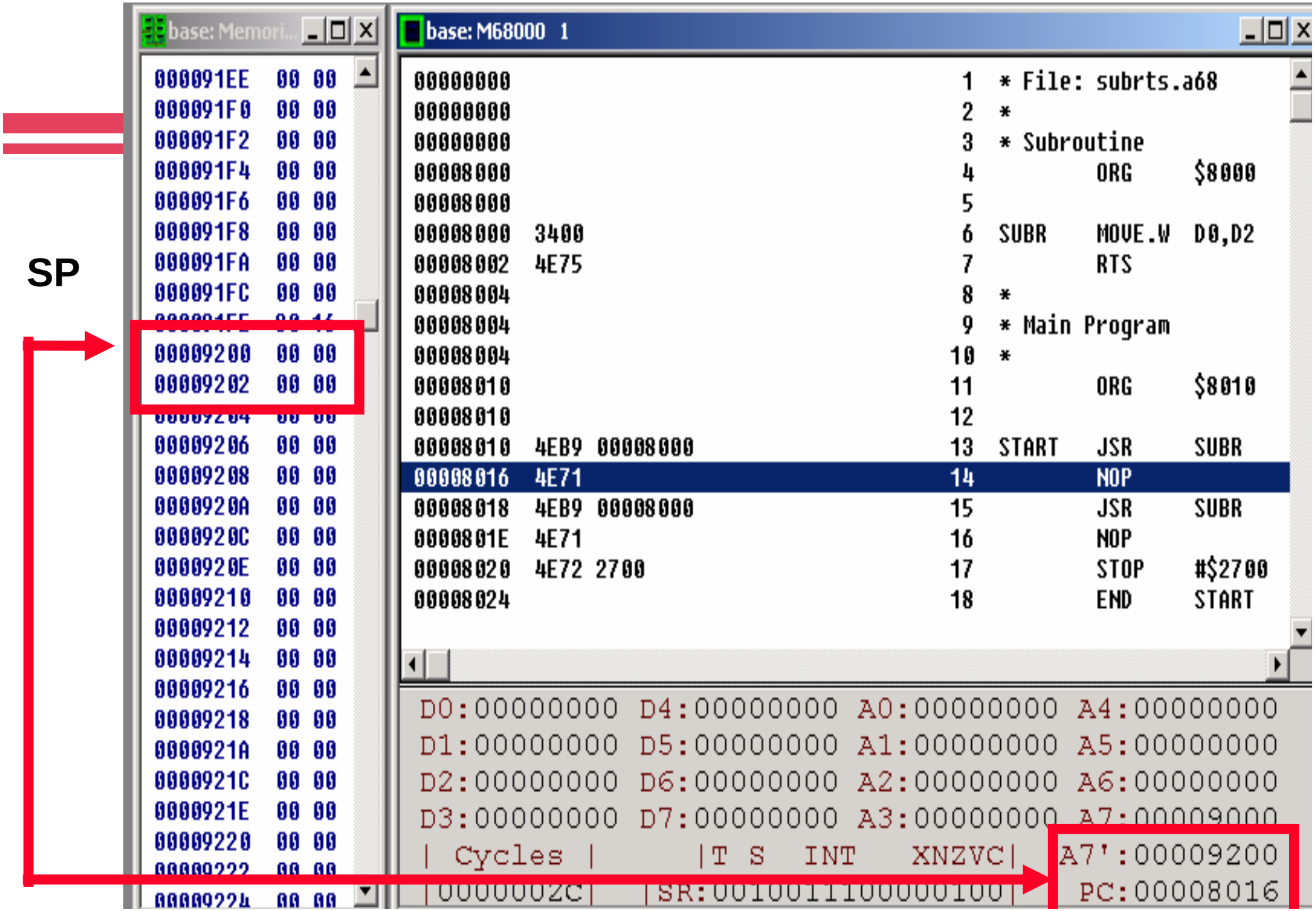

SP
base: Memori...
000091EE 00 00
000091F0 00 00
000091F2 00 00
000091F4 00 00
000091F6 00 00
000091F8 00 00
000091FA 00 00
000091FC 00 00
000091FE 00 00
00009200 00 00
00009202 00 00
00009204 00 00
00009206 00 00
00009208 00 00
0000920A 00 00
0000920C 00 00
0000920E 00 00
00009210 00 00
00009212 00 00
00009214 00 00
00009216 00 00
00009218 00 00
0000921A 00 00
0000921C 00 00
0000921E 00 00
00009220 00 00
00009224 00 00
base: M68000 1
00000000 1 * File: subrts.a68
00000000 2 *
00000000 3 * Subroutine
00008000 4 ORG $8000
00008000 5
00008000 3400 6 SUBR MOVE.W D0,D2
00008002 4E75 7 RTS
00008004 8 *
00008004 9 * Main Program
00008004 10 *
00008010 11 ORG $8010
00008010 12
00008010 4EB9 00008000 13 START JSR SUBR
00008016 4E71 14 NOP
00008018 4EB9 00008000 15 JSR SUBR
0000801E 4E71 16 NOP
00008020 4E72 2700 17 STOP #$2700
00008024 18 END START
D0:00000000 D4:00000000 A0:00000000 A4:00000000
D1:00000000 D5:00000000 A1:00000000 A5:00000000
D2:00000000 D6:00000000 A2:00000000 A6:00000000
D3:00000000 D7:00000000 A3:00000000 A7:00009000
| Cycles | |T S INT XNZVC|
|00000000| |SR:0010011100000000|
A7':00009200
PC:00008010

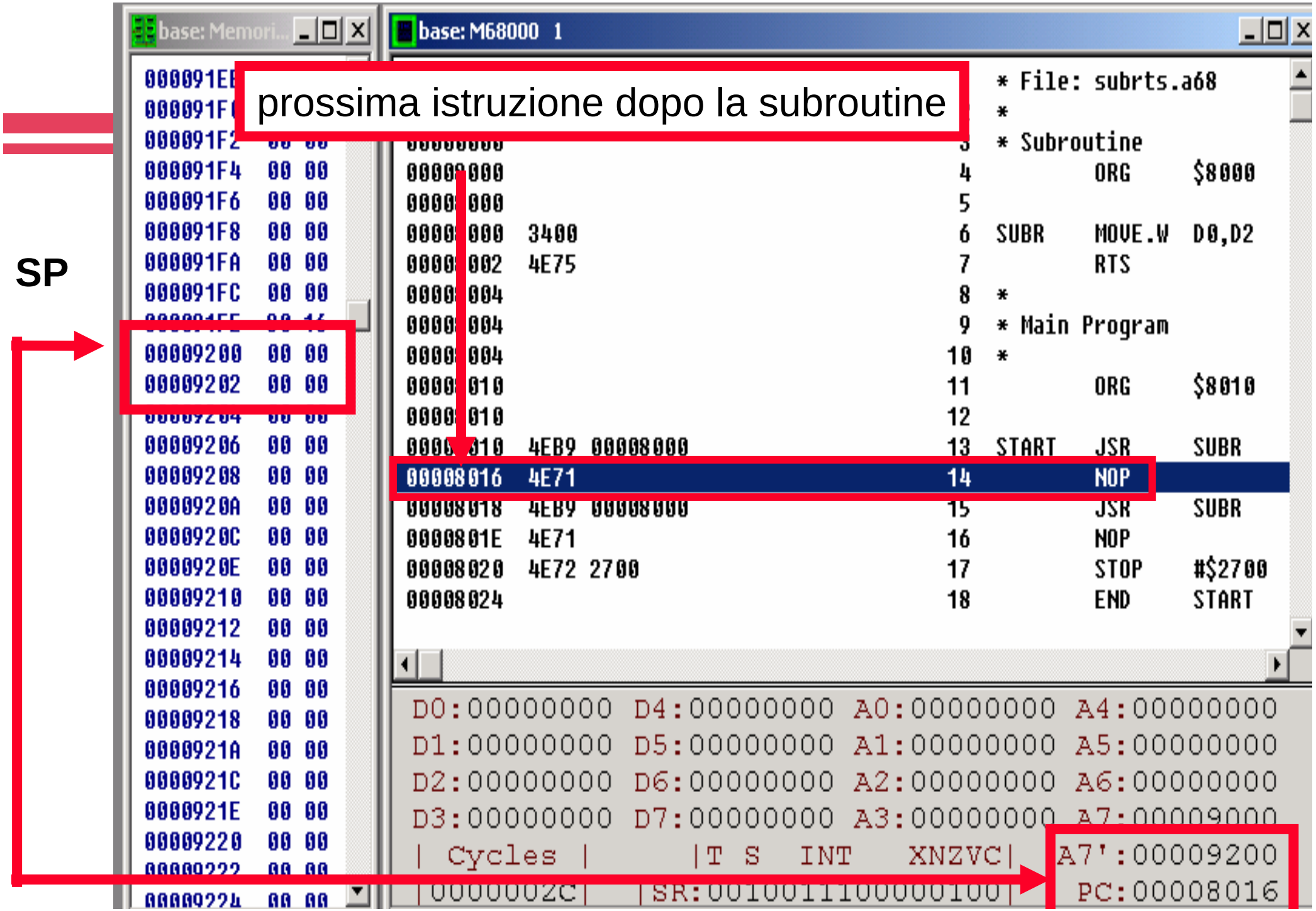

SP
base: Memori...
000091EE 00 00
000091F0 00 00
000091F2 00 00
000091F4 00 00
000091F6 00 00
000091F8 00 00
000091FC 00 00
000091FE 80 16
00009202 00 00
00009204 00 00
00009206 00 00
00009208 00 00
0000920A 00 00
0000920C 00 00
0000920E 00 00
00009210 00 00
00009212 00 00
00009214 00 00
00009216 00 00
00009218 00 00
0000921A 00 00
0000921C 00 00
0000921E 00 00
00009220 00 00
base: M68000 1
00000000 1 * File: subrts.a68
00000000 2 *
00000000 3 * Subroutine
00008000 4 ORG $8000
00008000 5
00008000 3400 6 SUBR MOVE.W D0,D2
00008002 4E75 7 RTS
00008004 8 *
00008004 9 * Main Program
00008004 10 *
00008010 11 ORG $8010
00008010 12
00008010 4EB9 00008000 13 START JSR SUBR
00008016 4E71 14 NOP
00008018 4EB9 00008000 15 JSR SUBR
0000801E 4E71 16 NOP
00008020 4E72 2700 17 STOP #$2700
00008024 18 END START
D0:00000000 D4:00000000 A0:00000000 A4:00000000
D1:00000000 D5:00000000 A1:00000000 A5:00000000
D2:00000000 D6:00000000 A2:00000000 A6:00000000
D3:00000000 D7:00000000 A3:00000000
| Cycles | |T S INT XNZVC|
|00000014| |SR:0010011100000000|
A7':000091FC
PC:00008000


SP
base: Memori...
base: M68000 1
return address
000091FC 00 00
000091FE 80 16
00009202 00 00
00009204 00 00
00009206 00 00
00009208 00 00
0000920A 00 00
0000920C 00 00
0000920E 00 00
00009210 00 00
00009212 00 00
00009214 00 00
00009216 00 00
00009218 00 00
0000921A 00 00
0000921C 00 00
0000921E 00 00
00009220 00 00
1 * File: subrts.a68
2 *
3 * Subroutine
00008000 4 ORG $8000
00008000 5
00008000 3400 6 SUBR MOVE.W D0,D2
00008002 4E75 7 RTS
00008004 8 *
00008004 9 * Main Program
00008004 10 *
00008010 11 ORG $8010
00008010 12
00008010 4EB9 00008000 13 START JSR SUBR
00008016 4E71 14 NOP
00008018 4EB9 00008000 15 JSR SUBR
0000801E 4E71 16 NOP
00008020 4E72 2700 17 STOP #$2700
00008024 18 END START
D0:00000000 D4:00000000 A0:00000000 A4:00000000
D1:00000000 D5:00000000 A1:00000000 A5:00000000
D2:00000000 D6:00000000 A2:00000000 A6:00000000
D3:00000000 D7:00000000 A3:00000000
| Cycles | |T S INT XNZVC|
|00000014| |SR:0010011100000000|
A7':000091FC
PC:00008000

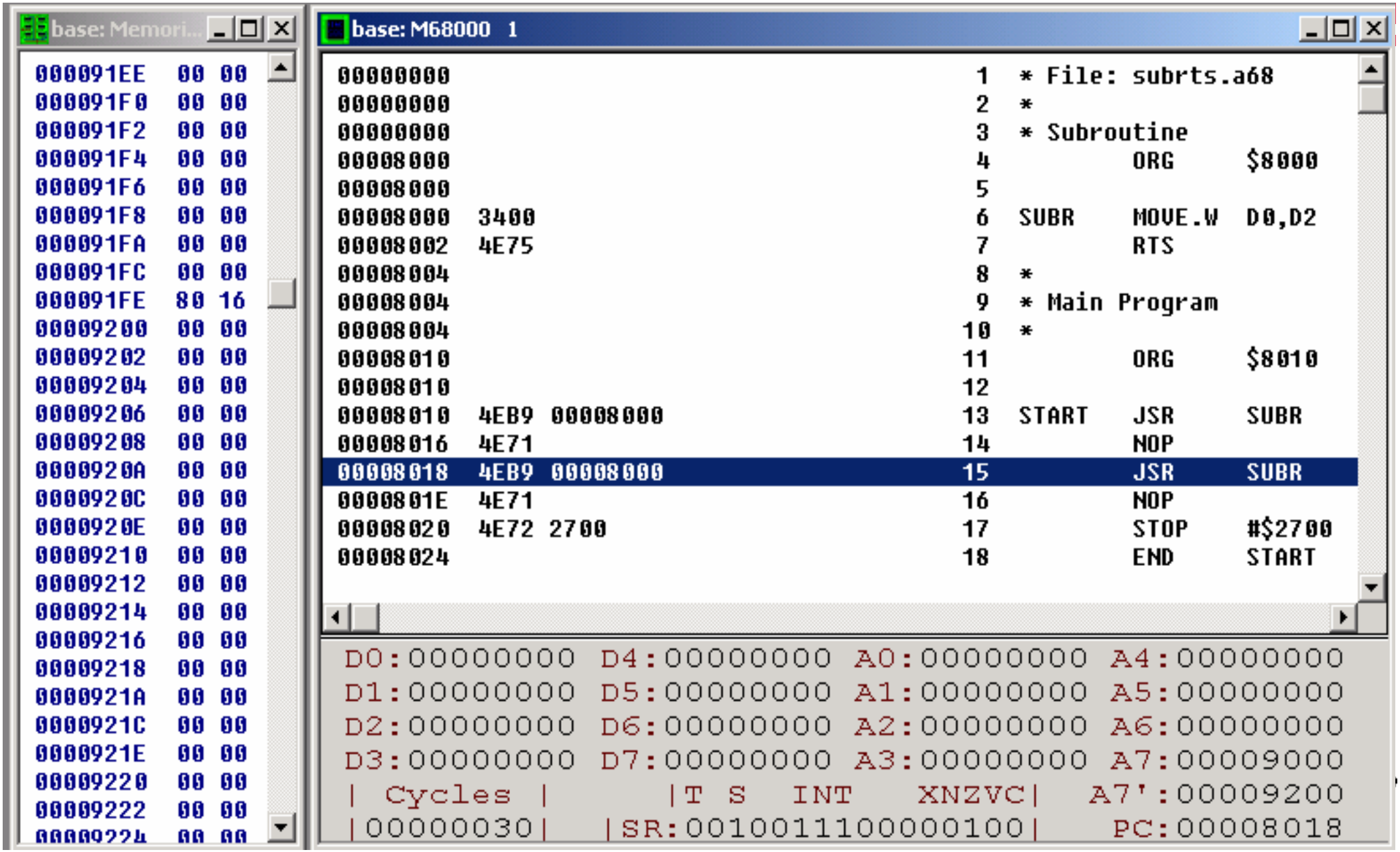
Esempio - esecuzione

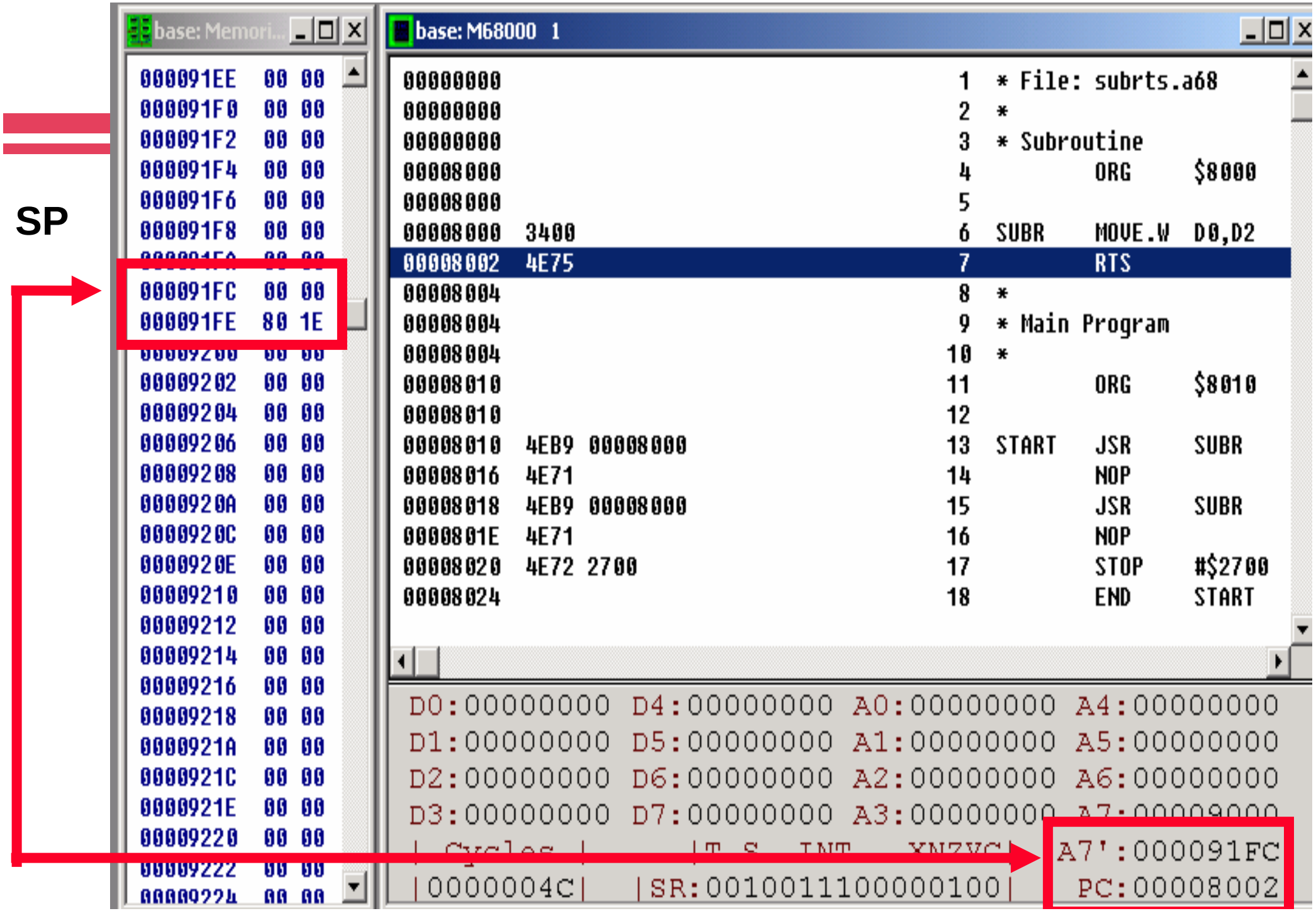


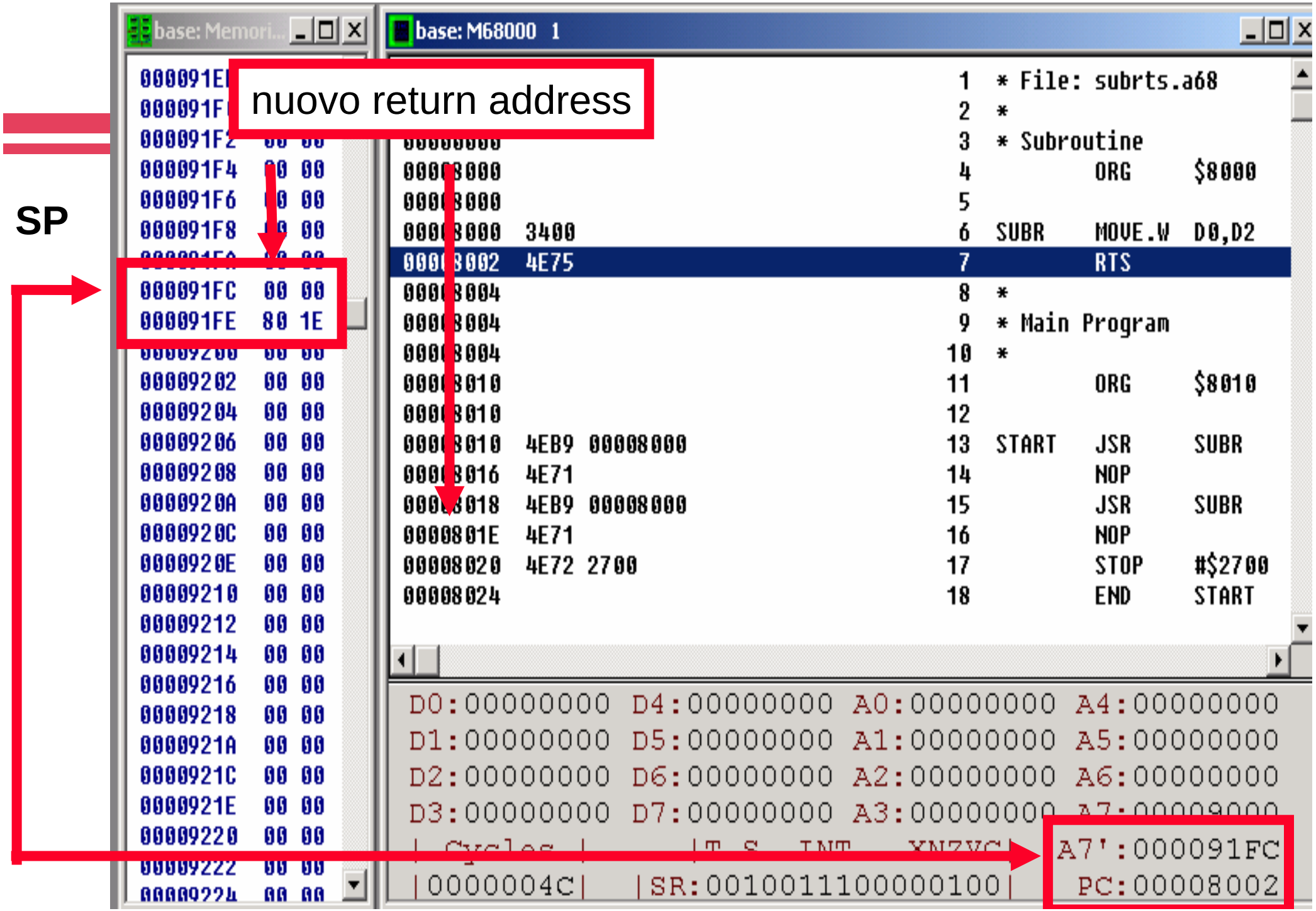

SP
base: Memori...
000091EE 00 00
000091F0 00 00
000091F2 00 00
000091F4 00 00
000091F6 00 00
000091F8 00 00
000091FA 00 00
000091FC 00 00
00009200 00 00
00009202 00 00
00009206 00 00
00009208 00 00
0000920A 00 00
0000920C 00 00
0000920E 00 00
00009210 00 00
00009212 00 00
00009214 00 00
00009216 00 00
00009218 00 00
0000921A 00 00
0000921C 00 00
0000921E 00 00
00009220 00 00
base: M68000 1
00000000 1 * File: subrts.a68
00000000 2 *
00000000 3 * Subroutine
00008000 4 ORG $8000
00008000 5
00008000 3400 6 SUBR MOVE.W D0,D2
00008002 4E75 7 RTS
00008004 8 *
00008004 9 * Main Program
00008004 10 *
00008010 11 ORG $8010
00008010 12
00008010 4EB9 00008000 13 START JSR SUBR
00008016 4E71 14 NOP
00008018 4EB9 00008000 15 JSR SUBR
0000801E 4E71 16 NOP
00008020 4E72 2700 17 STOP #$2700
00008024 18 END START
D0:00000000 D4:00000000 A0:00000000 A4:00000000
D1:00000000 D5:00000000 A1:00000000 A5:00000000
D2:00000000 D6:00000000 A2:00000000 A6:00000000
D3:00000000 D7:00000000 A3:00000000 A7:00009000
| Cycles | |T S INT XNZVC| A7':00009200
|0000002C| |SR:0010011100000100| PC:00008016

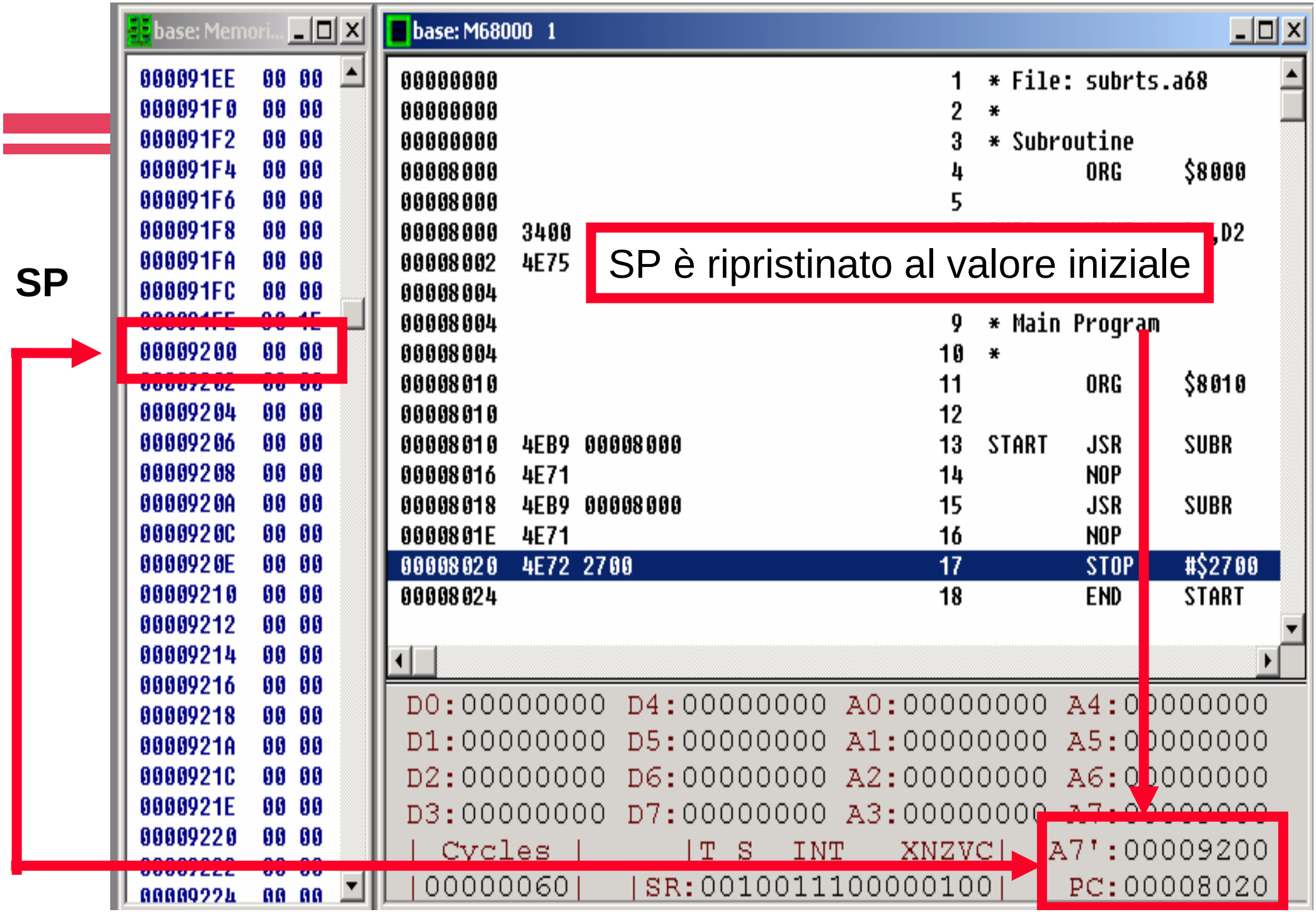

prossima istruzione dopo la subroutine
SP
base: Memori...
base: M68000 1
00009200 00 00
00009202 00 00
00009206 00 00
00009208 00 00
0000920A 00 00
0000920C 00 00
0000920E 00 00
00009210 00 00
00009212 00 00
00009214 00 00
00009216 00 00
00009218 00 00
0000921A 00 00
0000921C 00 00
0000921E 00 00
00009220 00 00
* File: subrts.a68
*
* Subroutine
00008000 4 ORG $8000
00008000 5
00008000 3400 6 SUBR MOVE.W D0,D2
00008002 4E75 7 RTS
00008004 8 *
00008004 9 * Main Program
00008004 10 *
00008010 11 ORG $8010
00008010 12
00008010 4EB9 00008000 13 START JSR SUBR
00008016 4E71 14 NOP
00008018 4EB9 00008000 15 JSR SUBR
0000801E 4E71 16 NOP
00008020 4E72 2700 17 STOP #$2700
00008024 18 END START
D0:00000000 D4:00000000 A0:00000000 A4:00000000
D1:00000000 D5:00000000 A1:00000000 A5:00000000
D2:00000000 D6:00000000 A2:00000000 A6:00000000
D3:00000000 D7:00000000 A3:00000000 A7:00009000
| Cycles | |T S INT XNZVC| A7':00009200
|0000002C| |SR:0010011100000100| PC:00008016

Esempio - esecuzione




SP
base: Memori...
000091EE 00 00
000091F0 00 00
000091F2 00 00
000091F4 00 00
000091F6 00 00
000091F8 00 00
000091FC 00 00
000091FE 80 1E
00009202 00 00
00009204 00 00
00009206 00 00
00009208 00 00
0000920A 00 00
0000920C 00 00
0000920E 00 00
00009210 00 00
00009212 00 00
00009214 00 00
00009216 00 00
00009218 00 00
0000921A 00 00
0000921C 00 00
0000921E 00 00
00009220 00 00
base: M68000 1
00000000 1 * File: subrts.a68
00000000 2 *
00000000 3 * Subroutine
00008000 4 ORG $8000
00008000 5
00008000 3400 6 SUBR MOVE.W D0,D2
00008002 4E75 7 RTS
00008004 8 *
00008004 9 * Main Program
00008004 10 *
00008010 11 ORG $8010
00008010 12
00008010 4EB9 00008000 13 START JSR SUBR
00008016 4E71 14 NOP
00008018 4EB9 00008000 15 JSR SUBR
0000801E 4E71 16 NOP
00008020 4E72 2700 17 STOP #$2700
00008024 18 END START
D0:00000000 D4:00000000 A0:00000000 A4:00000000
D1:00000000 D5:00000000 A1:00000000 A5:00000000
D2:00000000 D6:00000000 A2:00000000 A6:00000000
D3:00000000 D7:00000000 A3:00000000
A7':000091FC
|0000004C| |SR:0010011100000100| PC:00008002


nuovo return address
SP
base: Memori...
base: M68000 1
000091FC 00 00
000091FE 80 1E
00009202 00 00
00009204 00 00
00009206 00 00
00009208 00 00
0000920A 00 00
0000920C 00 00
0000920E 00 00
00009210 00 00
00009212 00 00
00009214 00 00
00009216 00 00
00009218 00 00
0000921A 00 00
0000921C 00 00
0000921E 00 00
00009220 00 00
1 * File: subrts.a68
2 *
3 * Subroutine
4 ORG $8000
5
00008000 3400 6 SUBR MOVE.W D0,D2
00008002 4E75 7 RTS
8 *
9 * Main Program
10 *
11 ORG $8010
12
00008010 4EB9 00008000 13 START JSR SUBR
00008016 4E71 14 NOP
00008018 4EB9 00008000 15 JSR SUBR
0000801E 4E71 16 NOP
00008020 4E72 2700 17 STOP #$2700
00008024 18 END START
D0:00000000 D4:00000000 A0:00000000 A4:00000000
D1:00000000 D5:00000000 A1:00000000 A5:00000000
D2:00000000 D6:00000000 A2:00000000 A6:00000000
D3:00000000 D7:00000000 A3:00000000
A7':000091FC
PC:00008002
|0000004C| |SR:0010011100000100|


base: Memori...
base: M68000 1
SP
00009200 00 00
00000000 1 * File: subrts.a68
00000000 2 *
00000000 3 * Subroutine
00008000 4 ORG $8000
00008000 5
00008000 3400 ,D2
00008002 4E75
00008004
SP è ripristinato al valore iniziale
00008004 9 * Main Program
00008004 10 *
00008010 11 ORG $8010
00008010 12
00008010 4EB9 00008000 13 START JSR SUBR
00008016 4E71 14 NOP
00008018 4EB9 00008000 15 JSR SUBR
0000801E 4E71 16 NOP
00008020 4E72 2700 17 STOP #$2700
00008024 18 END START
D0:00000000 D4:00000000 A0:00000000 A4:00000000
D1:00000000 D5:00000000 A1:00000000 A5:00000000
D2:00000000 D6:00000000 A2:00000000 A6:00000000
D3:00000000 D7:00000000 A3:00000000
| Cycles | |T S INT XNZVC|
|00000060| |SR:0010011100000100|
A7':00009200
PC:00008020

Considerazioni su quest'ultima soluzione

- Vantaggi:
  - È possibile memorizzare la subroutine in una ROM
  - È possibile effettuare chiamate ricorsive
- Svantaggi:
  - Per essere veloce, richiede risorse hardware extra
- E' la soluzione più diffusa