

; main program

;

main move Param2, -(sp)
 move Param1, -(sp)

Level1 Call Sub1
 move (sp), Result
 Add #8, sp

...
...

Sub1 moveM.l R0-R3, -(sp)
 move.l Param3, -(sp)

....

 Call Sub2
Level2 move (sp)+, R2

 move (sp)+, R2

...

...

 move R3, 20(sp)
 moveM (sp)+, R0-R3
 Return

Sub2 moveM.l R0-R1, -(sp)

....

 move 12(sp), R0

...

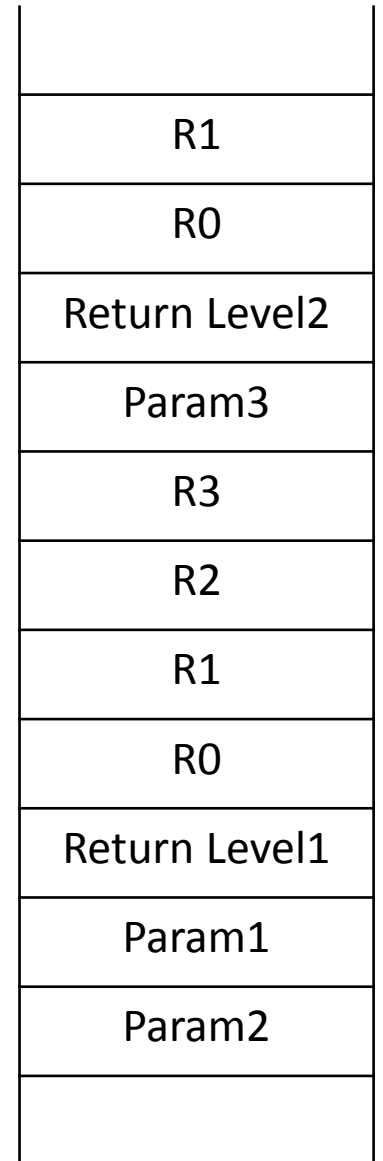
...

 move R1, 12(sp)
 moveM (sp)+, R0-R1
 Return

SP →

SP →

SP →



All moves above are longword moves

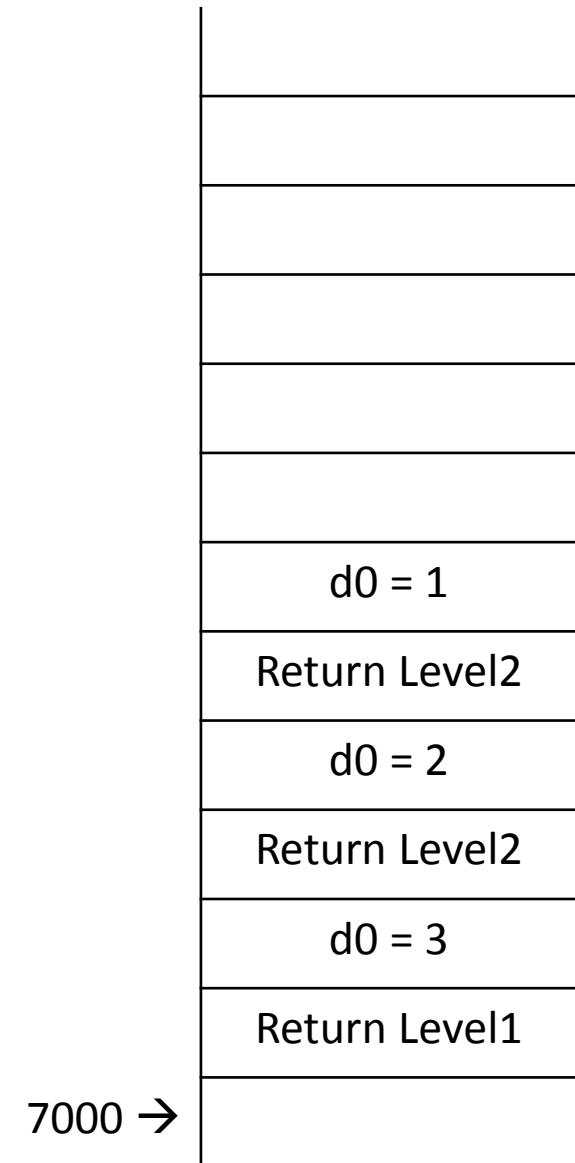
; Recursive Subroutine FACTOR

```
;
data      equ $6000
program   equ $4000
stack     equ $7000

          org data
number    dc.w    3
result    ds.w    1
          org program
main      movea.w #stack, sp
          move.w  number, d0
          bsr    FACTOR
Level1    move.w  d0, result
          move.w  #228, d7
          trap #14
```

```
FACTOR   move.w  d0, -(sp)
          subq.w  #1, d0
          bne    push
          move.w  (sp)+, d0
          bra    return
push      bsr    FACTOR
Level2    mulu   (sp)+, d0
return    rts

          end
```



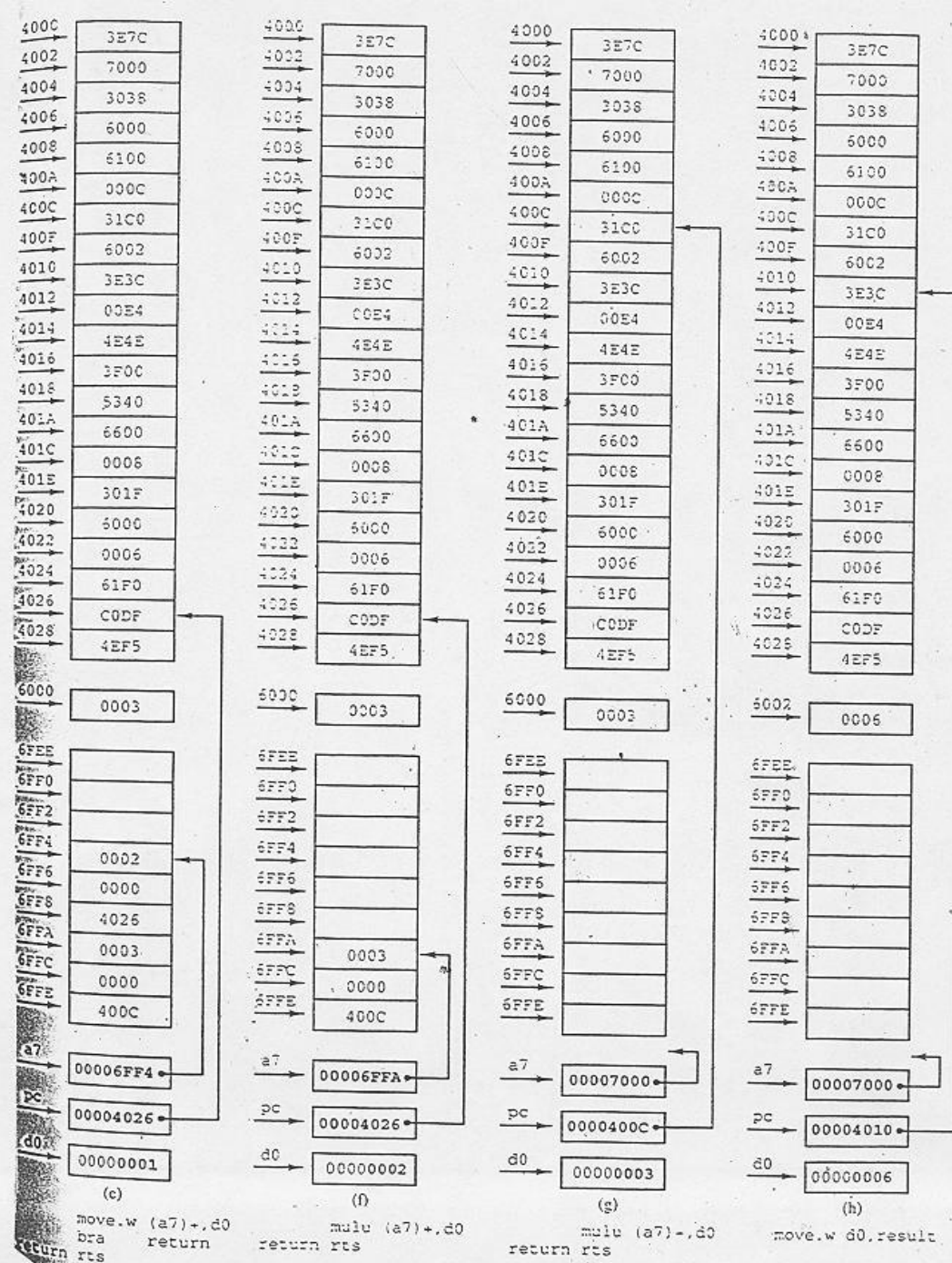


Figure 8.6 (Continued) Tracing recursive routine factor