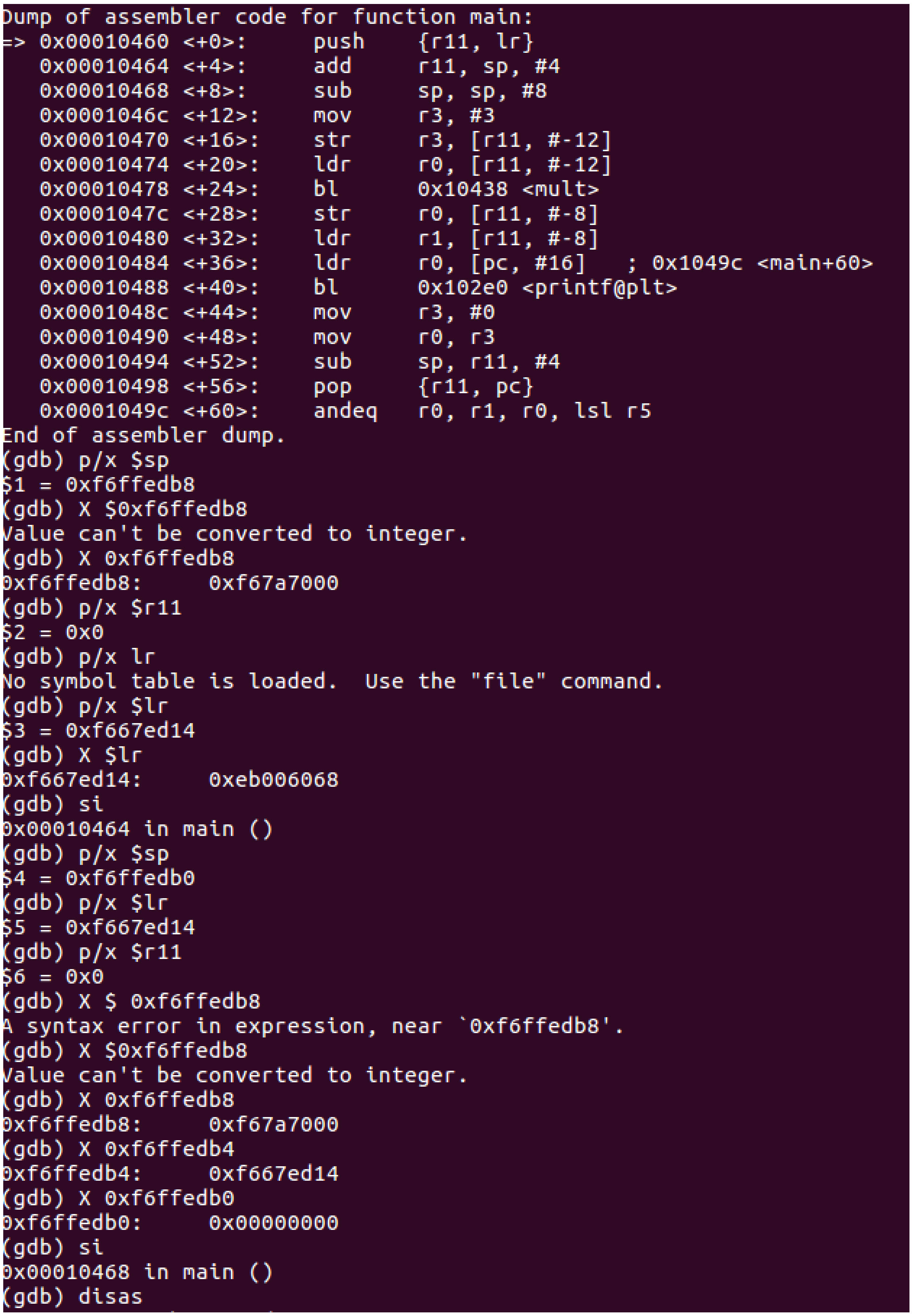
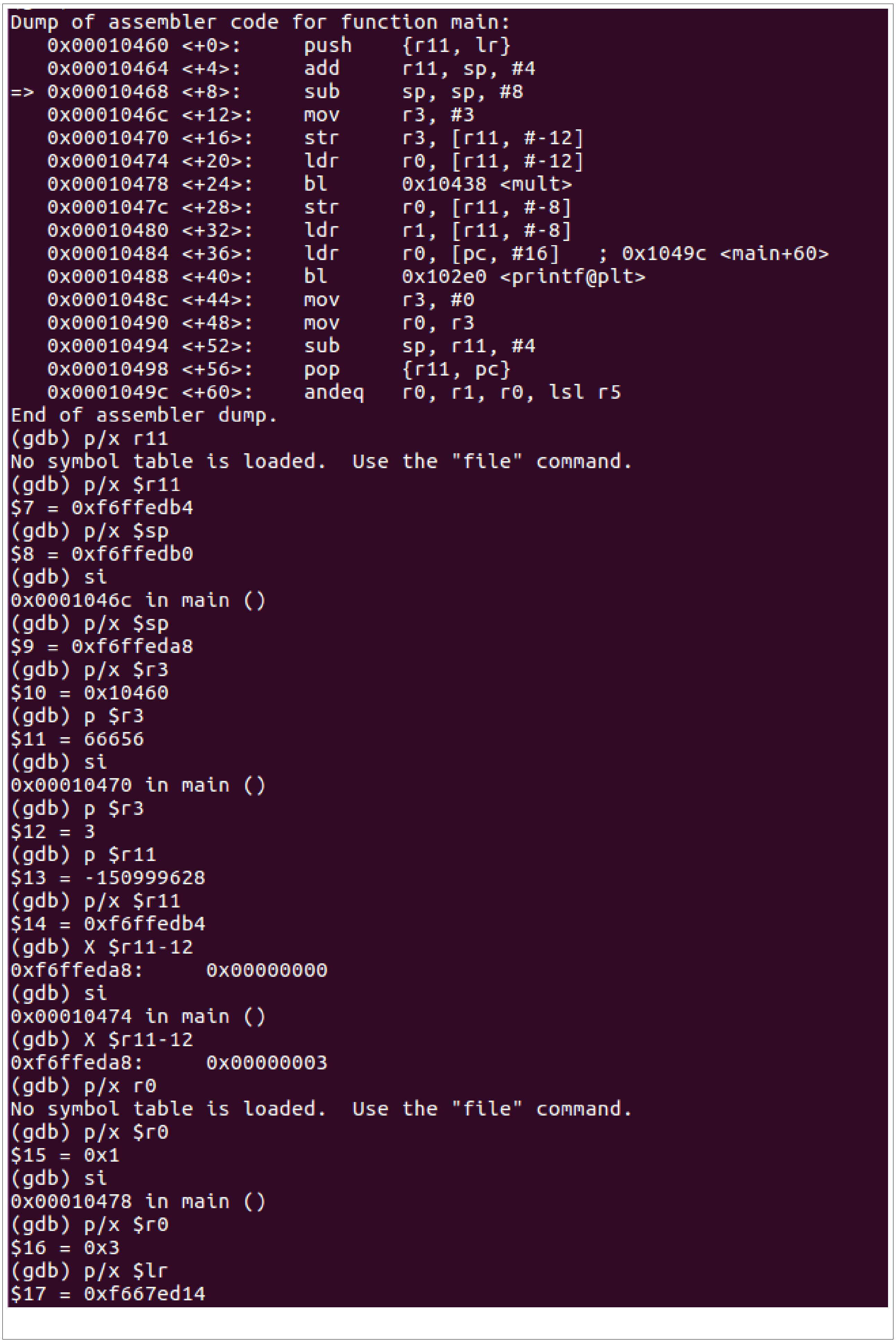
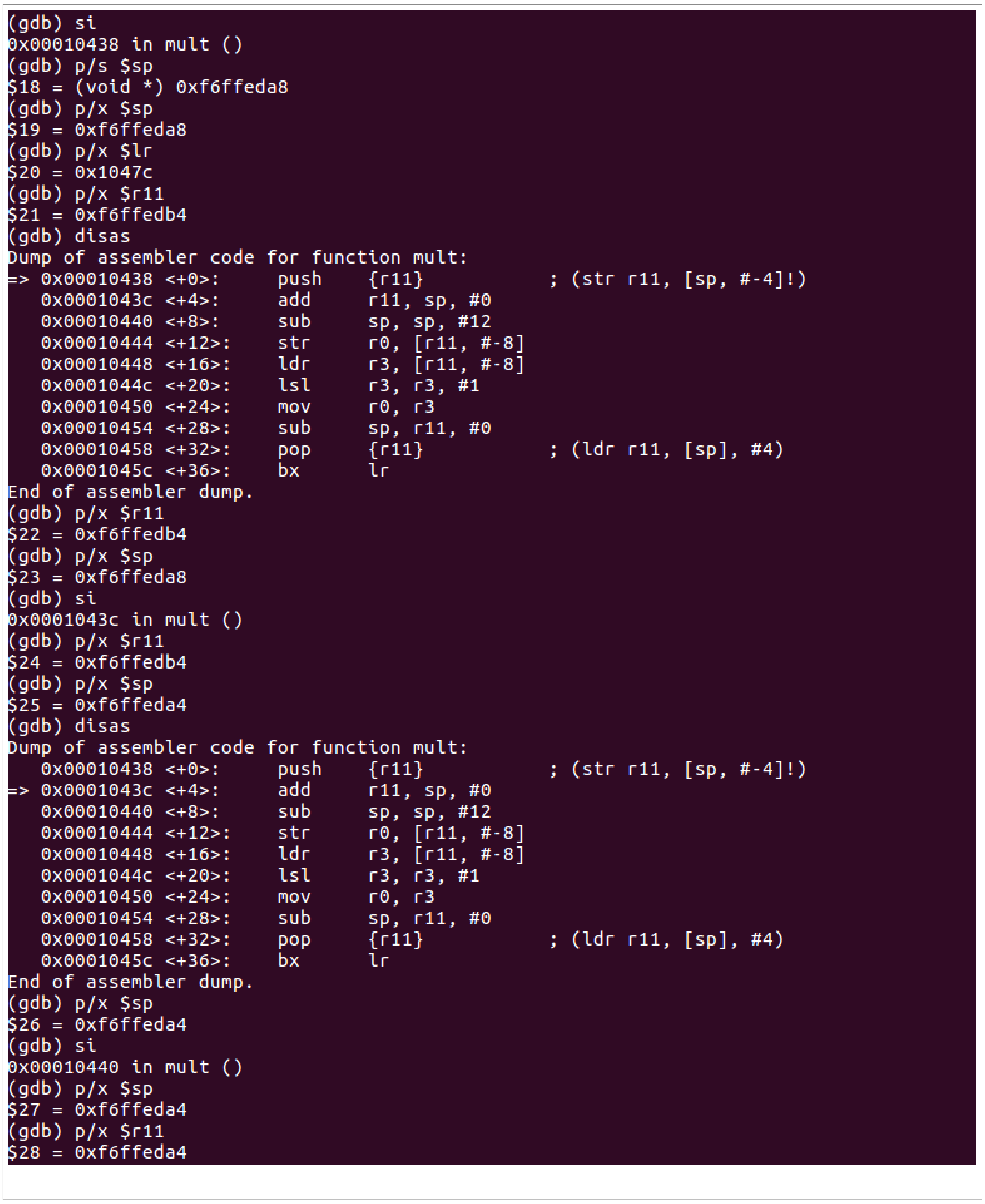
1. asm-func.c 실행파일 디버깅



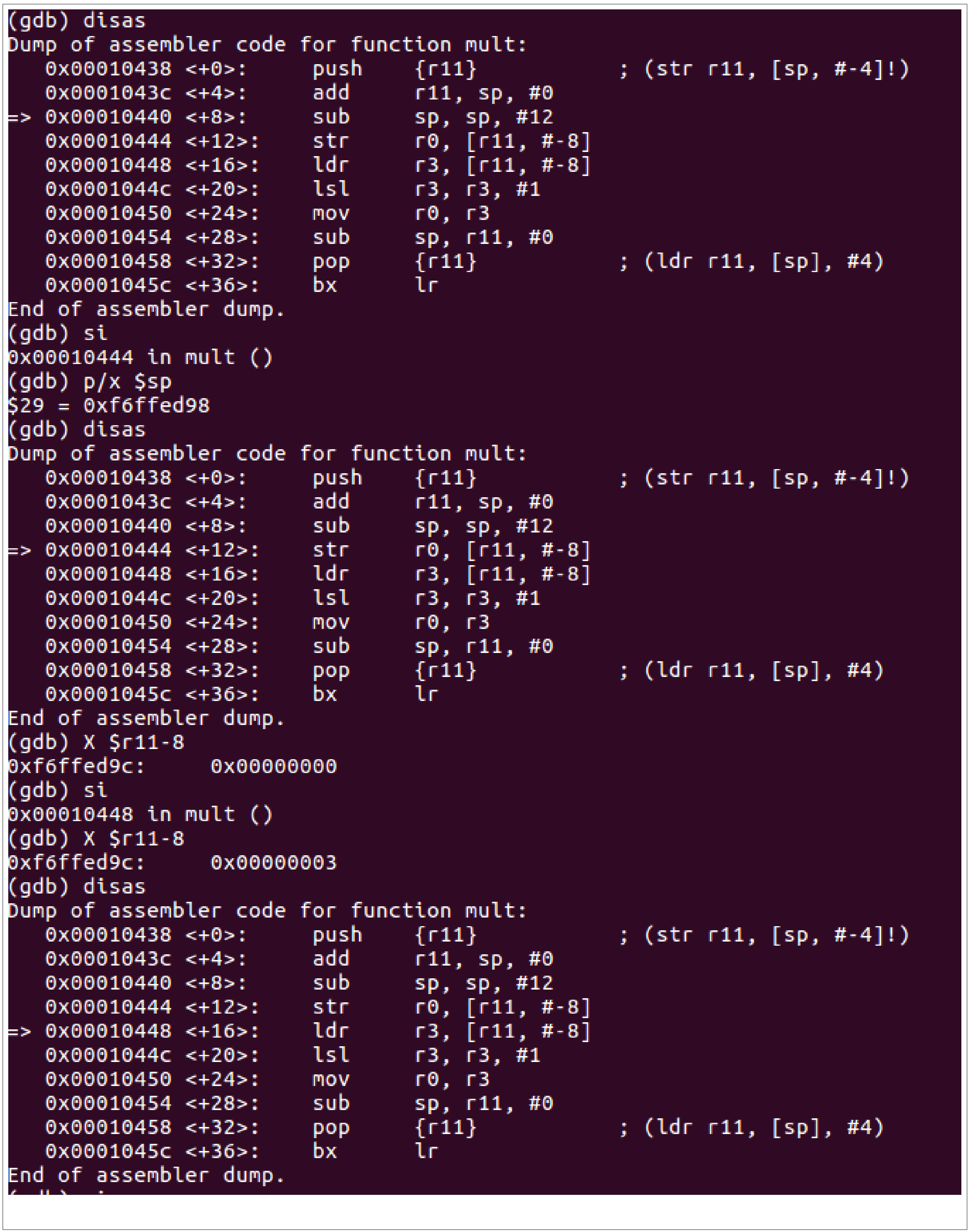
```
Dump of assembler code for function main:
=> 0x00010460 <+0>:     push    {r11, lr}
   0x00010464 <+4>:     add     r11, sp, #4
   0x00010468 <+8>:     sub     sp, sp, #8
   0x0001046c <+12>:    mov     r3, #3
   0x00010470 <+16>:    str     r3, [r11, #-12]
   0x00010474 <+20>:    ldr     r0, [r11, #-12]
   0x00010478 <+24>:    bl      0x10438 <mult>
   0x0001047c <+28>:    str     r0, [r11, #-8]
   0x00010480 <+32>:    ldr     r1, [r11, #-8]
   0x00010484 <+36>:    ldr     r0, [pc, #16]   ; 0x1049c <main+60>
   0x00010488 <+40>:    bl      0x102e0 <printf@plt>
   0x0001048c <+44>:    mov     r3, #0
   0x00010490 <+48>:    mov     r0, r3
   0x00010494 <+52>:    sub     sp, r11, #4
   0x00010498 <+56>:    pop     {r11, pc}
   0x0001049c <+60>:    andeq   r0, r1, r0, lsl r5
End of assembler dump.
(gdb) p/x $sp
$1 = 0xf6ffedb8
(gdb) X $0xf6ffedb8
Value can't be converted to integer.
(gdb) X 0xf6ffedb8
0xf6ffedb8:     0xf67a7000
(gdb) p/x $r11
$2 = 0x0
(gdb) p/x lr
No symbol table is loaded.  Use the "file" command.
(gdb) p/x $lr
$3 = 0xf667ed14
(gdb) X $lr
0xf667ed14:     0xeb006068
(gdb) si
0x00010464 in main ()
(gdb) p/x $sp
$4 = 0xf6ffedb0
(gdb) p/x $lr
$5 = 0xf667ed14
(gdb) p/x $r11
$6 = 0x0
(gdb) X $ 0xf6ffedb8
A syntax error in expression, near `0xf6ffedb8'.
(gdb) X $0xf6ffedb8
Value can't be converted to integer.
(gdb) X 0xf6ffedb8
0xf6ffedb8:     0xf67a7000
(gdb) X 0xf6ffedb4
0xf6ffedb4:     0xf667ed14
(gdb) X 0xf6ffedb0
0xf6ffedb0:     0x00000000
(gdb) si
0x00010468 in main ()
(gdb) disas
```



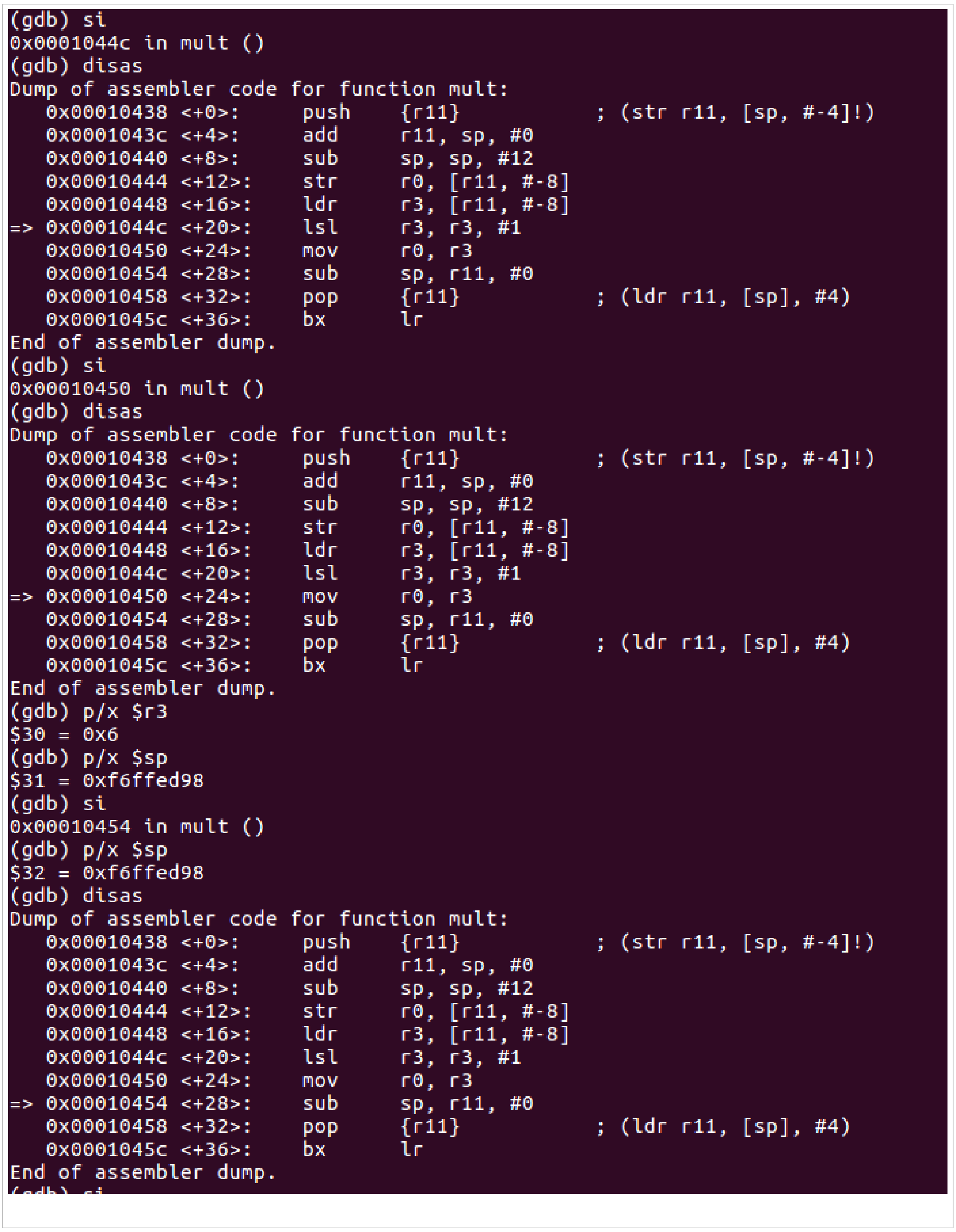
```
Dump of assembler code for function main:
   0x00010460 <+0>:     push    {r11, lr}
   0x00010464 <+4>:     add     r11, sp, #4
=> 0x00010468 <+8>:     sub     sp, sp, #8
   0x0001046c <+12>:    mov     r3, #3
   0x00010470 <+16>:    str     r3, [r11, #-12]
   0x00010474 <+20>:    ldr     r0, [r11, #-12]
   0x00010478 <+24>:    bl      0x10438 <mult>
   0x0001047c <+28>:    str     r0, [r11, #-8]
   0x00010480 <+32>:    ldr     r1, [r11, #-8]
   0x00010484 <+36>:    ldr     r0, [pc, #16]   ; 0x1049c <main+60>
   0x00010488 <+40>:    bl      0x102e0 <printf@plt>
   0x0001048c <+44>:    mov     r3, #0
   0x00010490 <+48>:    mov     r0, r3
   0x00010494 <+52>:    sub     sp, r11, #4
   0x00010498 <+56>:    pop     {r11, pc}
   0x0001049c <+60>:    andeq   r0, r1, r0, lsl r5
End of assembler dump.
(gdb) p/x r11
No symbol table is loaded.  Use the "file" command.
(gdb) p/x $r11
$7 = 0xf6ffedb4
(gdb) p/x $sp
$8 = 0xf6ffedb0
(gdb) si
0x0001046c in main ()
(gdb) p/x $sp
$9 = 0xf6ffeda8
(gdb) p/x $r3
$10 = 0x10460
(gdb) p $r3
$11 = 66656
(gdb) si
0x00010470 in main ()
(gdb) p $r3
$12 = 3
(gdb) p $r11
$13 = -150999628
(gdb) p/x $r11
$14 = 0xf6ffedb4
(gdb) X $r11-12
0xf6ffeda8:     0x00000000
(gdb) si
0x00010474 in main ()
(gdb) X $r11-12
0xf6ffeda8:     0x00000003
(gdb) p/x r0
No symbol table is loaded.  Use the "file" command.
(gdb) p/x $r0
$15 = 0x1
(gdb) si
0x00010478 in main ()
(gdb) p/x $r0
$16 = 0x3
(gdb) p/x $lr
$17 = 0xf667ed14
```



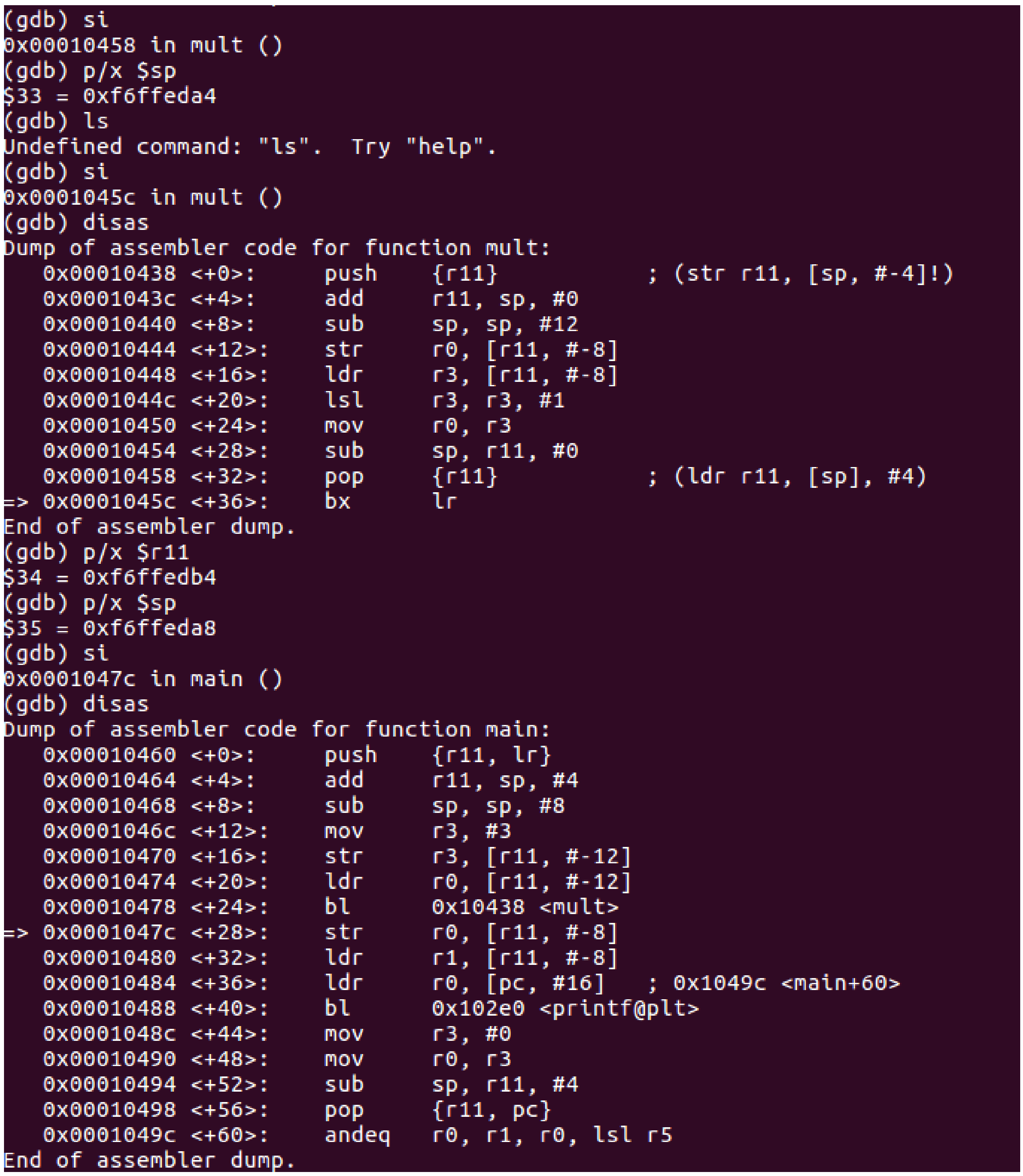
```
(gdb) si
0x00010438 in mult ()
(gdb) p/s $sp
$18 = (void *) 0xf6ffeda8
(gdb) p/x $sp
$19 = 0xf6ffeda8
(gdb) p/x $lr
$20 = 0x1047c
(gdb) p/x $r11
$21 = 0xf6ffedb4
(gdb) disas
Dump of assembler code for function mult:
=> 0x00010438 <+0>:     push    {r11}           ; (str r11, [sp, #-4]!)
   0x0001043c <+4>:     add     r11, sp, #0
   0x00010440 <+8>:     sub     sp, sp, #12
   0x00010444 <+12>:    str     r0, [r11, #-8]
   0x00010448 <+16>:    ldr     r3, [r11, #-8]
   0x0001044c <+20>:    lsl     r3, r3, #1
   0x00010450 <+24>:    mov     r0, r3
   0x00010454 <+28>:    sub     sp, r11, #0
   0x00010458 <+32>:    pop     {r11}           ; (ldr r11, [sp], #4)
   0x0001045c <+36>:    bx      lr
End of assembler dump.
(gdb) p/x $r11
$22 = 0xf6ffedb4
(gdb) p/x $sp
$23 = 0xf6ffeda8
(gdb) si
0x0001043c in mult ()
(gdb) p/x $r11
$24 = 0xf6ffedb4
(gdb) p/x $sp
$25 = 0xf6ffeda4
(gdb) disas
Dump of assembler code for function mult:
   0x00010438 <+0>:     push    {r11}           ; (str r11, [sp, #-4]!)
=> 0x0001043c <+4>:     add     r11, sp, #0
   0x00010440 <+8>:     sub     sp, sp, #12
   0x00010444 <+12>:    str     r0, [r11, #-8]
   0x00010448 <+16>:    ldr     r3, [r11, #-8]
   0x0001044c <+20>:    lsl     r3, r3, #1
   0x00010450 <+24>:    mov     r0, r3
   0x00010454 <+28>:    sub     sp, r11, #0
   0x00010458 <+32>:    pop     {r11}           ; (ldr r11, [sp], #4)
   0x0001045c <+36>:    bx      lr
End of assembler dump.
(gdb) p/x $sp
$26 = 0xf6ffeda4
(gdb) si
0x00010440 in mult ()
(gdb) p/x $sp
$27 = 0xf6ffeda4
(gdb) p/x $r11
$28 = 0xf6ffeda4
```



```
(gdb) disas
Dump of assembler code for function mult:
   0x00010438 <+0>:     push    {r11}           ; (str r11, [sp, #-4]!)
   0x0001043c <+4>:     add     r11, sp, #0
=> 0x00010440 <+8>:     sub     sp, sp, #12
   0x00010444 <+12>:    str     r0, [r11, #-8]
   0x00010448 <+16>:    ldr     r3, [r11, #-8]
   0x0001044c <+20>:    lsl     r3, r3, #1
   0x00010450 <+24>:    mov     r0, r3
   0x00010454 <+28>:    sub     sp, r11, #0
   0x00010458 <+32>:    pop     {r11}           ; (ldr r11, [sp], #4)
   0x0001045c <+36>:    bx      lr
End of assembler dump.
(gdb) si
0x00010444 in mult ()
(gdb) p/x $sp
$29 = 0xf6ffed98
(gdb) disas
Dump of assembler code for function mult:
   0x00010438 <+0>:     push    {r11}           ; (str r11, [sp, #-4]!)
   0x0001043c <+4>:     add     r11, sp, #0
   0x00010440 <+8>:     sub     sp, sp, #12
=> 0x00010444 <+12>:    str     r0, [r11, #-8]
   0x00010448 <+16>:    ldr     r3, [r11, #-8]
   0x0001044c <+20>:    lsl     r3, r3, #1
   0x00010450 <+24>:    mov     r0, r3
   0x00010454 <+28>:    sub     sp, r11, #0
   0x00010458 <+32>:    pop     {r11}           ; (ldr r11, [sp], #4)
   0x0001045c <+36>:    bx      lr
End of assembler dump.
(gdb) X $r11-8
0xf6ffed9c:     0x00000000
(gdb) si
0x00010448 in mult ()
(gdb) X $r11-8
0xf6ffed9c:     0x00000003
(gdb) disas
Dump of assembler code for function mult:
   0x00010438 <+0>:     push    {r11}           ; (str r11, [sp, #-4]!)
   0x0001043c <+4>:     add     r11, sp, #0
   0x00010440 <+8>:     sub     sp, sp, #12
   0x00010444 <+12>:    str     r0, [r11, #-8]
=> 0x00010448 <+16>:    ldr     r3, [r11, #-8]
   0x0001044c <+20>:    lsl     r3, r3, #1
   0x00010450 <+24>:    mov     r0, r3
   0x00010454 <+28>:    sub     sp, r11, #0
   0x00010458 <+32>:    pop     {r11}           ; (ldr r11, [sp], #4)
   0x0001045c <+36>:    bx      lr
End of assembler dump.
```



```
(gdb) si
0x0001044c in mult ()
(gdb) disas
Dump of assembler code for function mult:
   0x00010438 <+0>:     push    {r11}           ; (str r11, [sp, #-4]!)
   0x0001043c <+4>:     add     r11, sp, #0
   0x00010440 <+8>:     sub     sp, sp, #12
   0x00010444 <+12>:    str     r0, [r11, #-8]
   0x00010448 <+16>:    ldr     r3, [r11, #-8]
=> 0x0001044c <+20>:    lsl     r3, r3, #1
   0x00010450 <+24>:    mov     r0, r3
   0x00010454 <+28>:    sub     sp, r11, #0
   0x00010458 <+32>:    pop     {r11}           ; (ldr r11, [sp], #4)
   0x0001045c <+36>:    bx      lr
End of assembler dump.
(gdb) si
0x00010450 in mult ()
(gdb) disas
Dump of assembler code for function mult:
   0x00010438 <+0>:     push    {r11}           ; (str r11, [sp, #-4]!)
   0x0001043c <+4>:     add     r11, sp, #0
   0x00010440 <+8>:     sub     sp, sp, #12
   0x00010444 <+12>:    str     r0, [r11, #-8]
   0x00010448 <+16>:    ldr     r3, [r11, #-8]
   0x0001044c <+20>:    lsl     r3, r3, #1
=> 0x00010450 <+24>:    mov     r0, r3
   0x00010454 <+28>:    sub     sp, r11, #0
   0x00010458 <+32>:    pop     {r11}           ; (ldr r11, [sp], #4)
   0x0001045c <+36>:    bx      lr
End of assembler dump.
(gdb) p/x $r3
$30 = 0x6
(gdb) p/x $sp
$31 = 0xf6ffed98
(gdb) si
0x00010454 in mult ()
(gdb) p/x $sp
$32 = 0xf6ffed98
(gdb) disas
Dump of assembler code for function mult:
   0x00010438 <+0>:     push    {r11}           ; (str r11, [sp, #-4]!)
   0x0001043c <+4>:     add     r11, sp, #0
   0x00010440 <+8>:     sub     sp, sp, #12
   0x00010444 <+12>:    str     r0, [r11, #-8]
   0x00010448 <+16>:    ldr     r3, [r11, #-8]
   0x0001044c <+20>:    lsl     r3, r3, #1
   0x00010450 <+24>:    mov     r0, r3
=> 0x00010454 <+28>:    sub     sp, r11, #0
   0x00010458 <+32>:    pop     {r11}           ; (ldr r11, [sp], #4)
   0x0001045c <+36>:    bx      lr
End of assembler dump.
```



```
(gdb) si
0x00010458 in mult ()
(gdb) p/x $sp
$33 = 0xf6ffeda4
(gdb) ls
Undefined command: "ls".  Try "help".
(gdb) si
0x0001045c in mult ()
(gdb) disas
Dump of assembler code for function mult:
   0x00010438 <+0>:     push    {r11}           ; (str r11, [sp, #-4]!)
   0x0001043c <+4>:     add     r11, sp, #0
   0x00010440 <+8>:     sub     sp, sp, #12
   0x00010444 <+12>:    str     r0, [r11, #-8]
   0x00010448 <+16>:    ldr     r3, [r11, #-8]
   0x0001044c <+20>:    lsl     r3, r3, #1
   0x00010450 <+24>:    mov     r0, r3
   0x00010454 <+28>:    sub     sp, r11, #0
   0x00010458 <+32>:    pop     {r11}           ; (ldr r11, [sp], #4)
=> 0x0001045c <+36>:    bx      lr
End of assembler dump.
(gdb) p/x $r11
$34 = 0xf6ffedb4
(gdb) p/x $sp
$35 = 0xf6ffeda8
(gdb) si
0x0001047c in main ()
(gdb) disas
Dump of assembler code for function main:
   0x00010460 <+0>:     push    {r11, lr}
   0x00010464 <+4>:     add     r11, sp, #4
   0x00010468 <+8>:     sub     sp, sp, #8
   0x0001046c <+12>:    mov     r3, #3
   0x00010470 <+16>:    str     r3, [r11, #-12]
   0x00010474 <+20>:    ldr     r0, [r11, #-12]
   0x00010478 <+24>:    bl      0x10438 <mult>
=> 0x0001047c <+28>:    str     r0, [r11, #-8]
   0x00010480 <+32>:    ldr     r1, [r11, #-8]
   0x00010484 <+36>:    ldr     r0, [pc, #16]   ; 0x1049c <main+60>
   0x00010488 <+40>:    bl      0x102e0 <printf@plt>
   0x0001048c <+44>:    mov     r3, #0
   0x00010490 <+48>:    mov     r0, r3
   0x00010494 <+52>:    sub     sp, r11, #4
   0x00010498 <+56>:    pop     {r11, pc}
   0x0001049c <+60>:    andeq   r0, r1, r0, lsl r5
End of assembler dump.
```