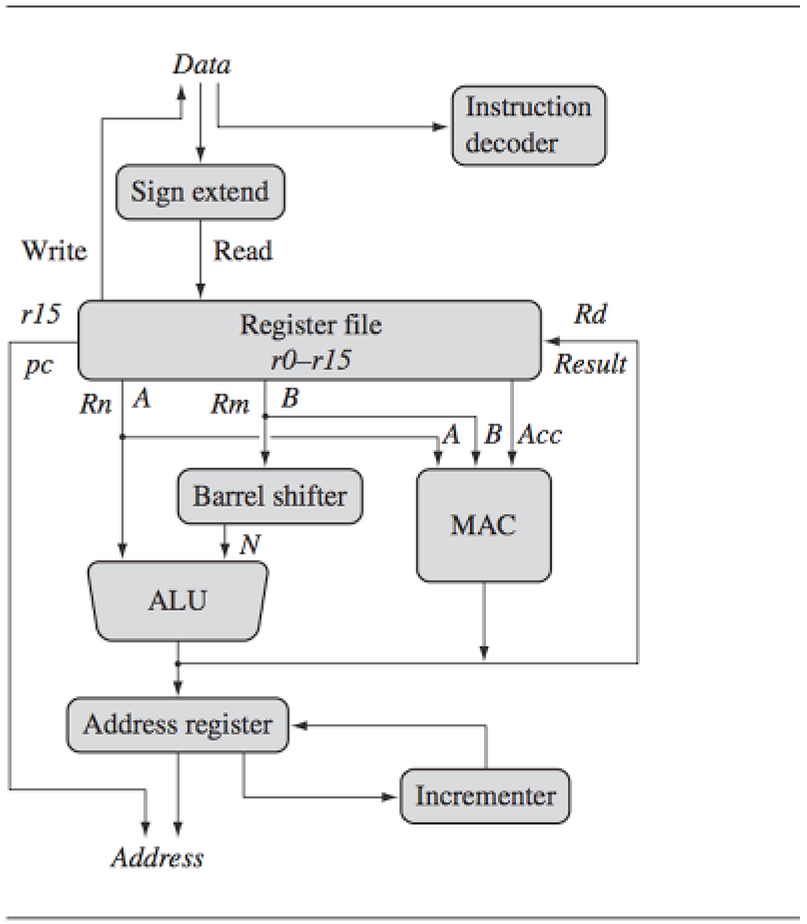
TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – GJ (박현우)
uc820@naver.com

목차

1. ARM ARCHITECTURE 1



ARM core dataflow model.

ARM

옵션 – MAC이 있으면 DSP

MAC

--> 곱셈연산으로 덧셈 클록 수를 줄이기 위함.

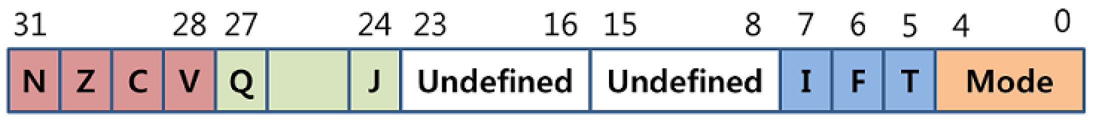
--> 곱셈 덧셈 처리를 1클록에 함.

cpsr -> intel eflags register랑 같음

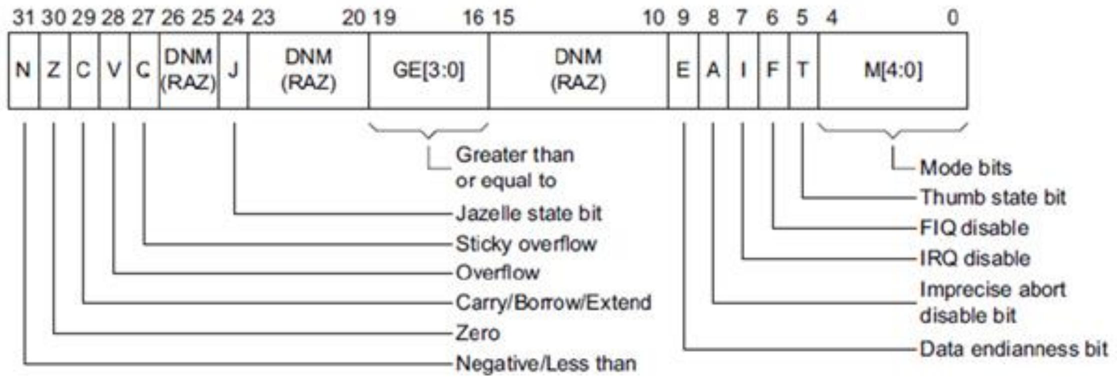
satuartion --> core 연산

1. ARM ARCHITECTURE 2

CPSR 구성



0~4 비트 - 프로세서의 동작 모드
(User – 10000, FIQ – 10001, IRQ – 10010, SVC – 10011, 등등)
5 비트 – Thumb 상태
6 비트 – FIQ, 0-enable, 1-Disable
7 비트 – IRQ, 0-enable, 1-Disable
24 비트 – JAVA 명령 실행
27 비트 – 포화 연산 수행 결과
28~31 비트 – 연산 결과 (Overflow, Carry flag, Zero, Negative)

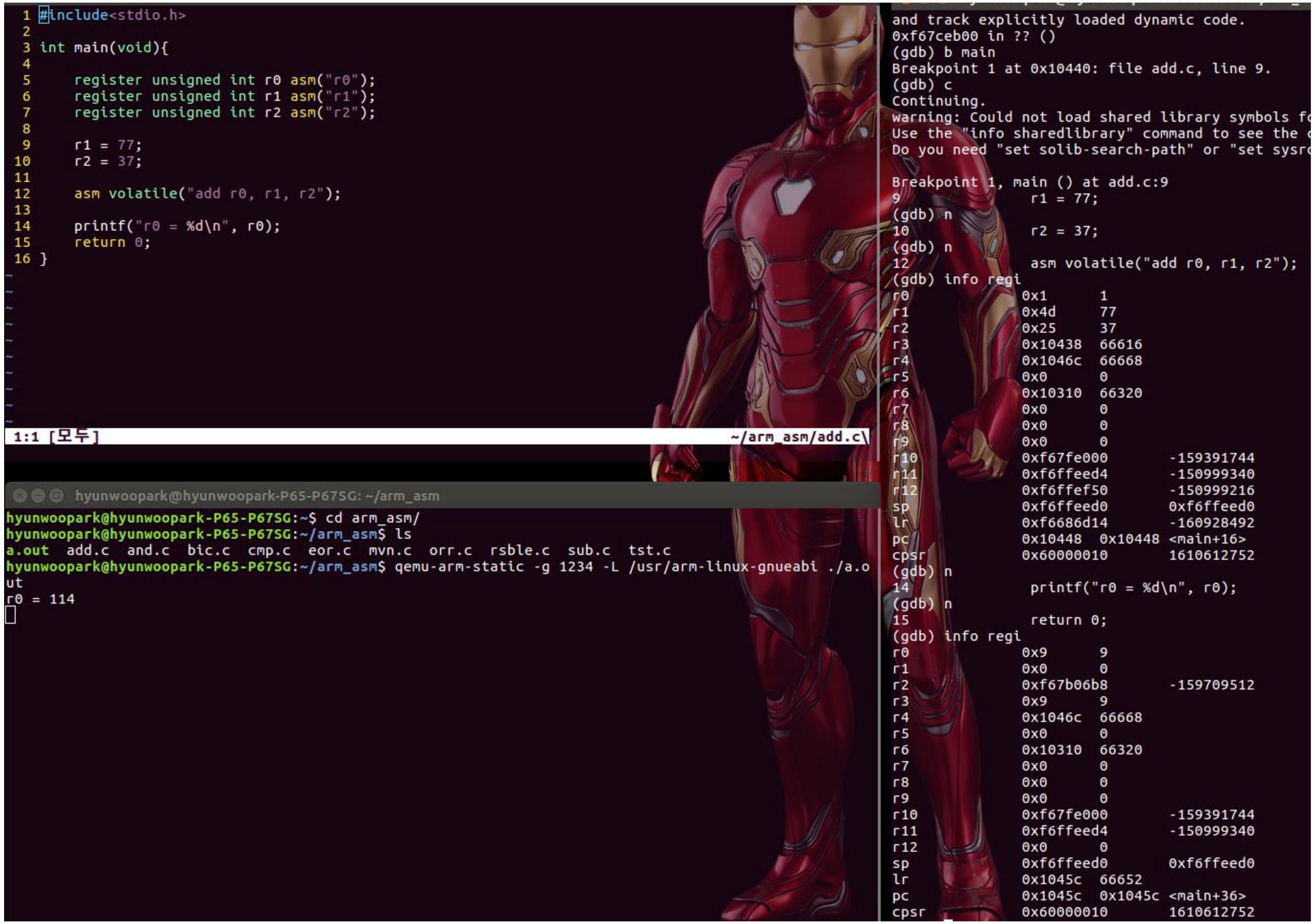


arm 4byte 단위씩 명령어 구성.

Shift를 이용해서 모자란 bit수를 활용함.

lt : less than
gt : greater than
eq : equal to
ne : not equal to
le : less than or equal to
ge : greater than or equal to

2. ARM ASM – ADD

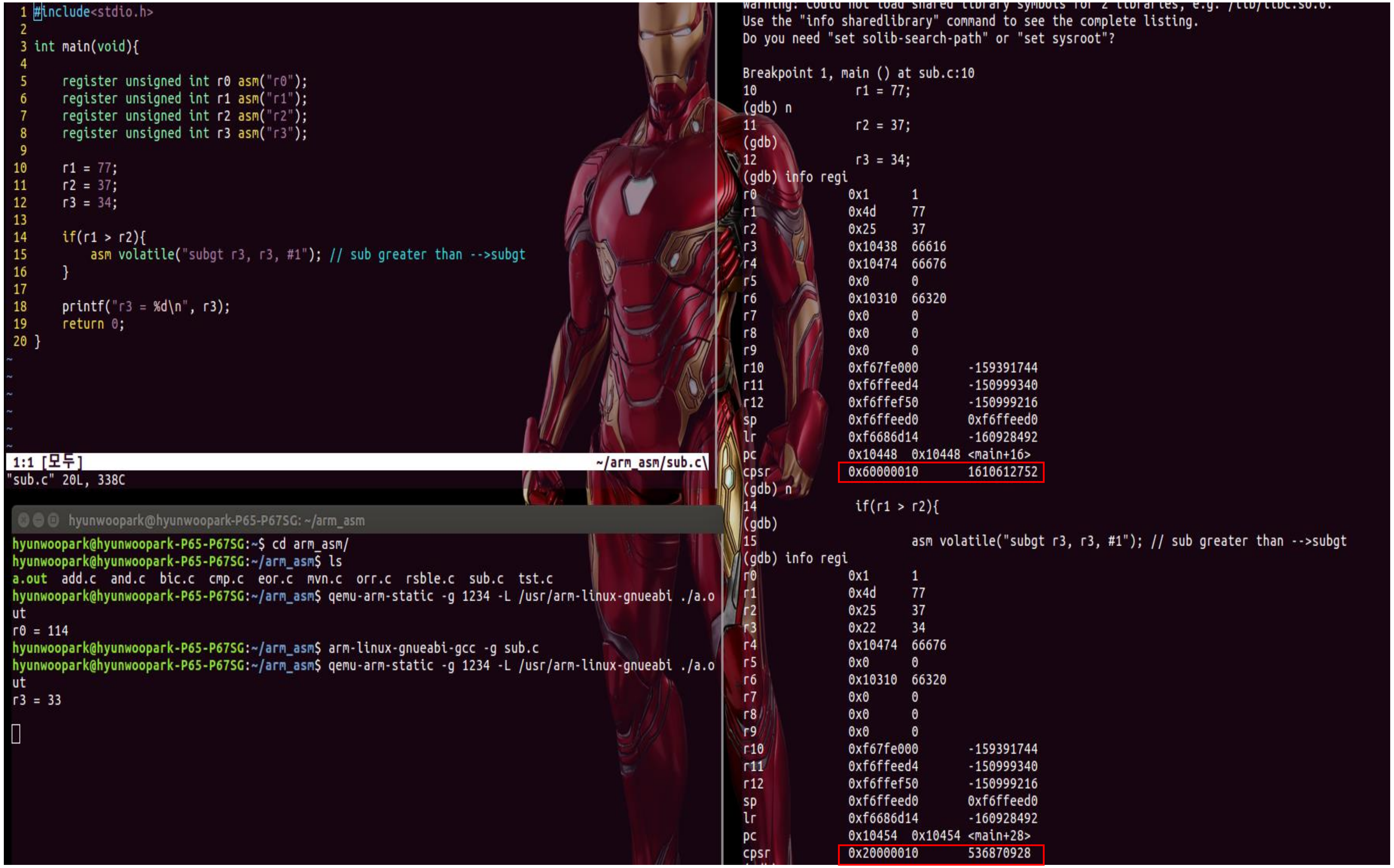


```
 1 #include<stdio.h>
 2 
 3 int main(void){
 4 
 5     register unsigned int r0 asm("r0");
 6     register unsigned int r1 asm("r1");
 7     register unsigned int r2 asm("r2");
 8 
 9     r1 = 77;
10     r2 = 37;
11 
12     asm volatile("add r0, r1, r2");
13 
14     printf("r0 = %d\n", r0);
15     return 0;
16 }
1:1 [모두]                                              ~/arm_asm/add.c\
```

```
hyunwoopark@hyunwoopark-P65-P67SG: ~/arm_asm
hyunwoopark@hyunwoopark-P65-P67SG:~$ cd arm_asm/
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ ls
a.out  add.c  and.c  bic.c  cmp.c  eor.c  mvn.c  orr.c  rsble.c  sub.c  tst.c
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ qemu-arm-static -g 1234 -L /usr/arm-linux-gnueabi ./a.o
ut
r0 = 114
```

```
and track explicitly loaded dynamic code.
0xf67ceb00 in ?? ()
(gdb) b main
Breakpoint 1 at 0x10440: file add.c, line 9.
(gdb) c
Continuing.
warning: Could not load shared library symbols fo
Use the "info sharedlibrary" command to see the c
Do you need "set solib-search-path" or "set sysro

Breakpoint 1, main () at add.c:9
9               r1 = 77;
(gdb) n
10              r2 = 37;
(gdb) n
12              asm volatile("add r0, r1, r2");
(gdb) info regi
r0             0x1      1
r1             0x4d     77
r2             0x25     37
r3             0x10438  66616
r4             0x1046c  66668
r5             0x0      0
r6             0x10310  66320
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeed0       0xf6ffeed0
lr             0xf6686d14       -160928492
pc             0x10448  0x10448 <main+16>
cpsr           0x60000010       1610612752
(gdb) n
14              printf("r0 = %d\n", r0);
(gdb) n
15              return 0;
(gdb) info regi
r0             0x9      9
r1             0x0      0
r2             0xf67b06b8       -159709512
r3             0x9      9
r4             0x1046c  66668
r5             0x0      0
r6             0x10310  66320
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0x0      0
sp             0xf6ffeed0       0xf6ffeed0
lr             0x1045c  66652
pc             0x1045c  0x1045c <main+36>
cpsr           0x60000010       1610612752
```

2. ARM ASM – SUB

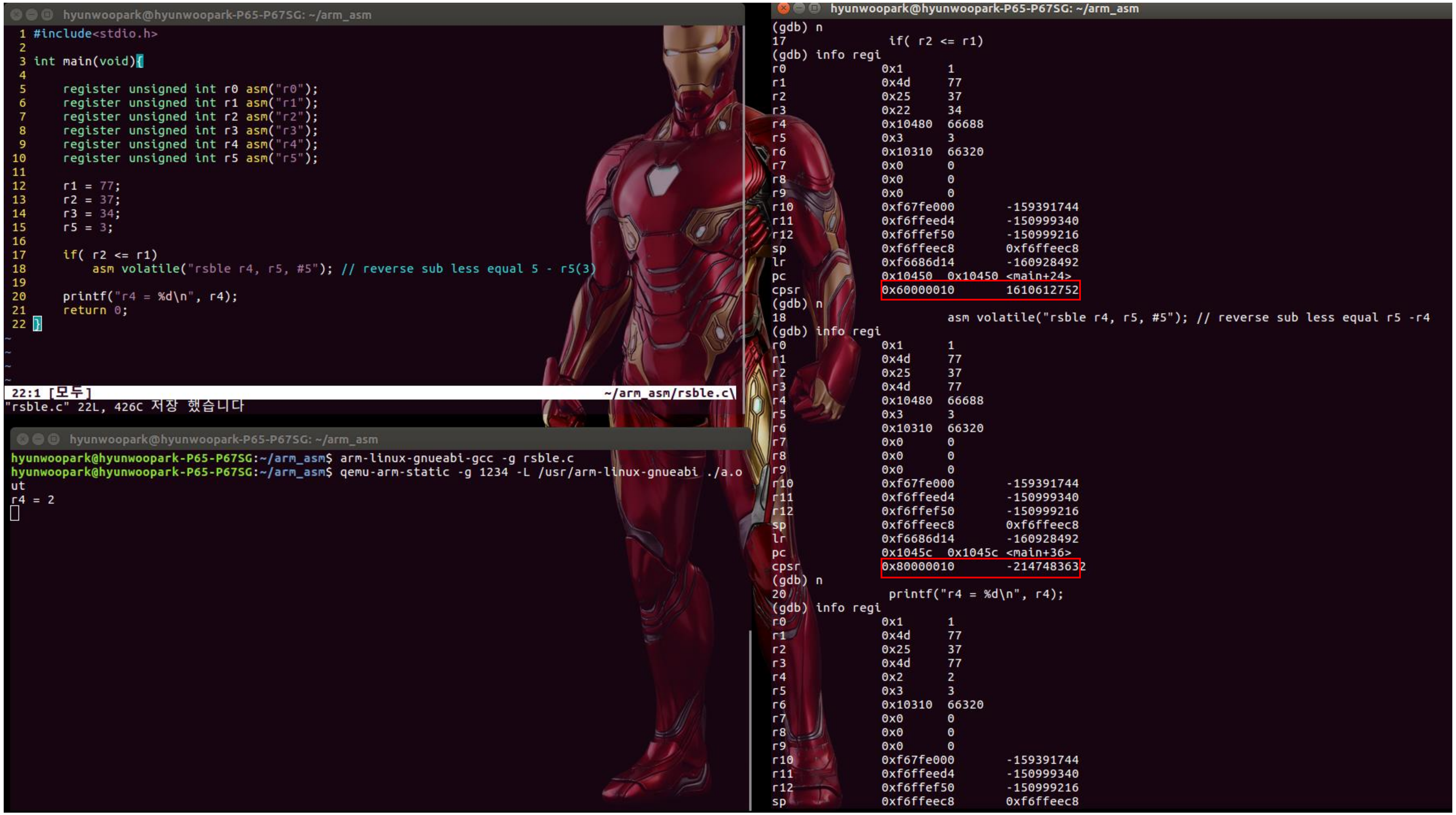


```
 1 #include<stdio.h>
 2 
 3 int main(void){
 4 
 5     register unsigned int r0 asm("r0");
 6     register unsigned int r1 asm("r1");
 7     register unsigned int r2 asm("r2");
 8     register unsigned int r3 asm("r3");
 9 
10     r1 = 77;
11     r2 = 37;
12     r3 = 34;
13 
14     if(r1 > r2){
15         asm volatile("subgt r3, r3, #1"); // sub greater than -->subgt
16     }
17 
18     printf("r3 = %d\n", r3);
19     return 0;
20 }
~
~
~
~
~
~
 1:1 [모두]                                                  ~/arm_asm/sub.c
"sub.c" 20L, 338C
```

```
hyunwoopark@hyunwoopark-P65-P67SG: ~/arm_asm
hyunwoopark@hyunwoopark-P65-P67SG:~$ cd arm_asm/
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ ls
a.out  add.c  and.c  bic.c  cmp.c  eor.c  mvn.c  orr.c  rsble.c  sub.c  tst.c
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ qemu-arm-static -g 1234 -L /usr/arm-linux-gnueabi ./a.o
ut
r0 = 114
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ arm-linux-gnueabi-gcc -g sub.c
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ qemu-arm-static -g 1234 -L /usr/arm-linux-gnueabi ./a.o
ut
r3 = 33
```

```
warning: Could not load shared library symbols for 2 libraries, e.g. /lib/libc.so.6.
Use the "info sharedlibrary" command to see the complete listing.
Do you need "set solib-search-path" or "set sysroot"?

Breakpoint 1, main () at sub.c:10
10              r1 = 77;
(gdb) n
11              r2 = 37;
(gdb)
12              r3 = 34;
(gdb) info regi
r0             0x1      1
r1             0x4d     77
r2             0x25     37
r3             0x10438  66616
r4             0x10474  66676
r5             0x0      0
r6             0x10310  66320
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeed0       0xf6ffeed0
lr             0xf6686d14       -160928492
pc             0x10448  0x10448 <main+16>
cpsr           0x60000010       1610612752
(gdb) n
14              if(r1 > r2){
(gdb)
15                      asm volatile("subgt r3, r3, #1"); // sub greater than -->subgt
(gdb) info regi
r0             0x1      1
r1             0x4d     77
r2             0x25     37
r3             0x22     34
r4             0x10474  66676
r5             0x0      0
r6             0x10310  66320
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeed0       0xf6ffeed0
lr             0xf6686d14       -160928492
pc             0x10454  0x10454 <main+28>
cpsr           0x20000010       536870928
```

2. ARM ASM – rsble

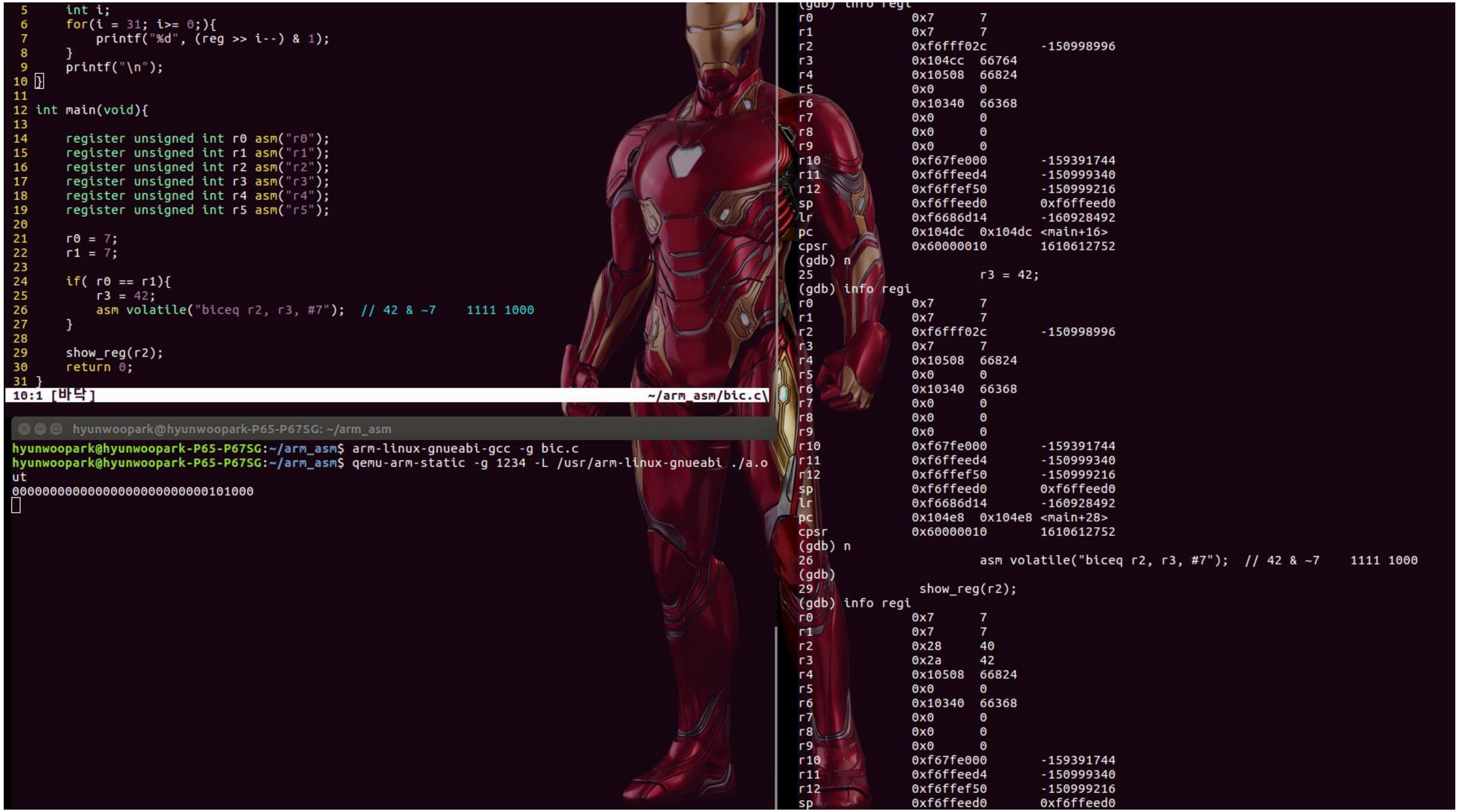


```
 1 #include<stdio.h>
 2
 3 int main(void){
 4
 5     register unsigned int r0 asm("r0");
 6     register unsigned int r1 asm("r1");
 7     register unsigned int r2 asm("r2");
 8     register unsigned int r3 asm("r3");
 9     register unsigned int r4 asm("r4");
10     register unsigned int r5 asm("r5");
11
12     r1 = 77;
13     r2 = 37;
14     r3 = 34;
15     r5 = 3;
16
17     if( r2 <= r1)
18         asm volatile("rsble r4, r5, #5"); // reverse sub less equal 5 - r5(3)
19
20     printf("r4 = %d\n", r4);
21     return 0;
22 }
~
~
~
~
 22:1 [모두]                                                    ~/arm_asm/rsble.c\
"rsble.c" 22L, 426C 저장 했습니다
```

```
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ arm-linux-gnueabi-gcc -g rsble.c
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ qemu-arm-static -g 1234 -L /usr/arm-linux-gnueabi ./a.o
ut
r4 = 2
```

```
(gdb) n
17              if( r2 <= r1)
(gdb) info regi
r0             0x1      1
r1             0x4d     77
r2             0x25     37
r3             0x22     34
r4             0x10480  66688
r5             0x3      3
r6             0x10310  66320
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeec8       0xf6ffeec8
lr             0xf6686d14       -160928492
pc             0x10450  0x10450 <main+24>
cpsr           0x60000010       1610612752
(gdb) n
18                      asm volatile("rsble r4, r5, #5"); // reverse sub less equal r5 -r4
(gdb) info regi
r0             0x1      1
r1             0x4d     77
r2             0x25     37
r3             0x4d     77
r4             0x10480  66688
r5             0x3      3
r6             0x10310  66320
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeec8       0xf6ffeec8
lr             0xf6686d14       -160928492
pc             0x1045c  0x1045c <main+36>
cpsr           0x80000010       -2147483632
(gdb) n
20              printf("r4 = %d\n", r4);
(gdb) info regi
r0             0x1      1
r1             0x4d     77
r2             0x25     37
r3             0x4d     77
r4             0x2      2
r5             0x3      3
r6             0x10310  66320
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeec8       0xf6ffeec8
```

2. ARM ASM – bic

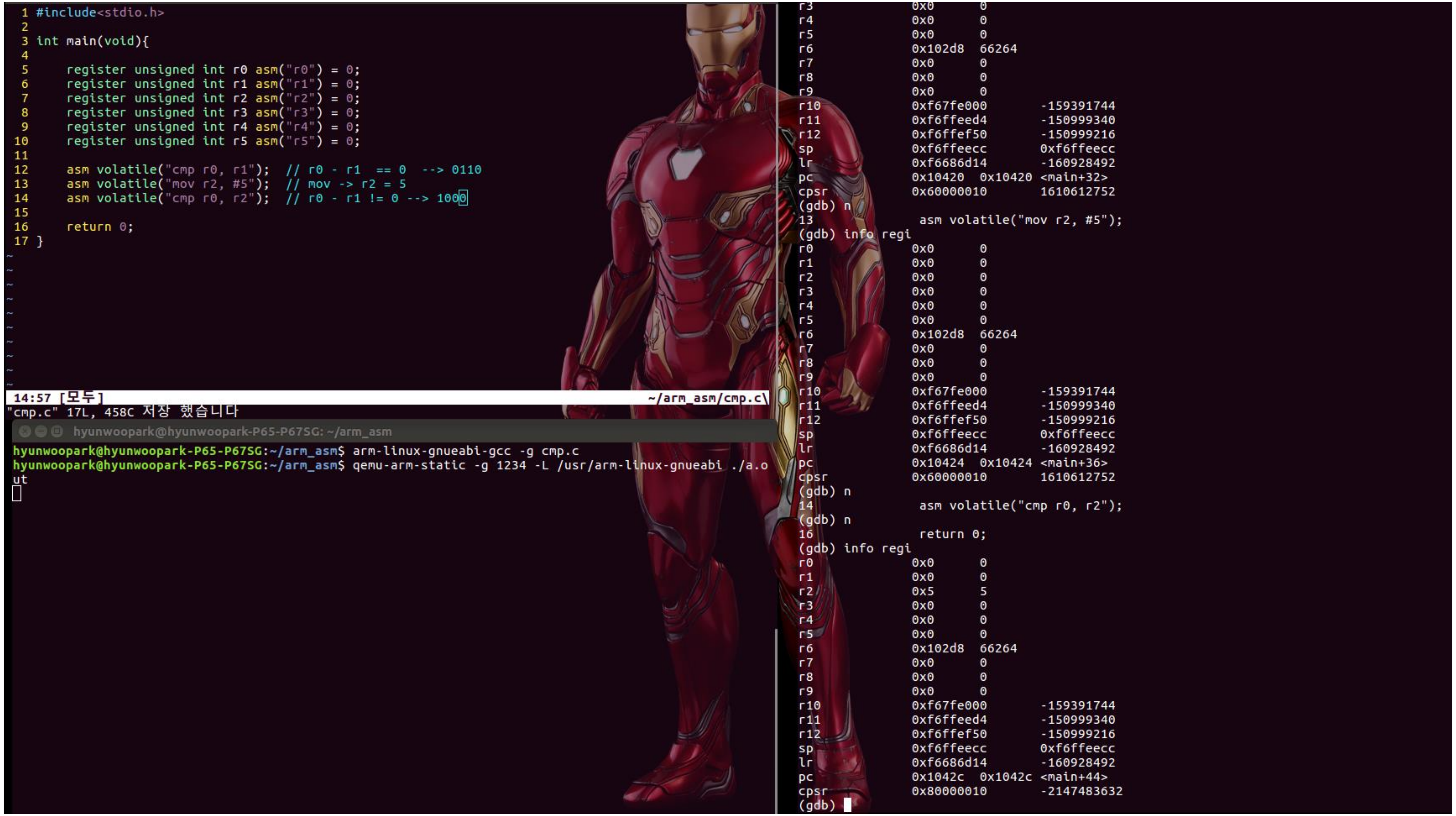


```
 5     int i;
 6     for(i = 31; i>= 0;){
 7         printf("%d", (reg >> i--) & 1);
 8     }
 9     printf("\n");
10 }
11
12 int main(void){
13
14     register unsigned int r0 asm("r0");
15     register unsigned int r1 asm("r1");
16     register unsigned int r2 asm("r2");
17     register unsigned int r3 asm("r3");
18     register unsigned int r4 asm("r4");
19     register unsigned int r5 asm("r5");
20
21     r0 = 7;
22     r1 = 7;
23
24     if( r0 == r1){
25         r3 = 42;
26         asm volatile("biceq r2, r3, #7");  // 42 & ~7    1111 1000
27     }
28
29     show_reg(r2);
30     return 0;
31 }
10:1 [바닥]                                                    ~/arm_asm/bic.c
```

```
hyunwoopark@hyunwoopark-P65-P67SG: ~/arm_asm
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ arm-linux-gnueabi-gcc -g bic.c
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ qemu-arm-static -g 1234 -L /usr/arm-linux-gnueabi ./a.o
ut
00000000000000000000000000101000
```

```
(gdb) info regi
r0             0x7      7
r1             0x7      7
r2             0xf6fff02c       -150998996
r3             0x104cc  66764
r4             0x10508  66824
r5             0x0      0
r6             0x10340  66368
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeed0       0xf6ffeed0
lr             0xf6686d14       -160928492
pc             0x104dc  0x104dc <main+16>
cpsr           0x60000010       1610612752
(gdb) n
25                      r3 = 42;
(gdb) info regi
r0             0x7      7
r1             0x7      7
r2             0xf6fff02c       -150998996
r3             0x7      7
r4             0x10508  66824
r5             0x0      0
r6             0x10340  66368
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeed0       0xf6ffeed0
lr             0xf6686d14       -160928492
pc             0x104e8  0x104e8 <main+28>
cpsr           0x60000010       1610612752
(gdb) n
26                      asm volatile("biceq r2, r3, #7");  // 42 & ~7    1111 1000
(gdb)
29              show_reg(r2);
(gdb) info regi
r0             0x7      7
r1             0x7      7
r2             0x28     40
r3             0x2a     42
r4             0x10508  66824
r5             0x0      0
r6             0x10340  66368
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeed0       0xf6ffeed0
```

2. ARM ASM – cmp

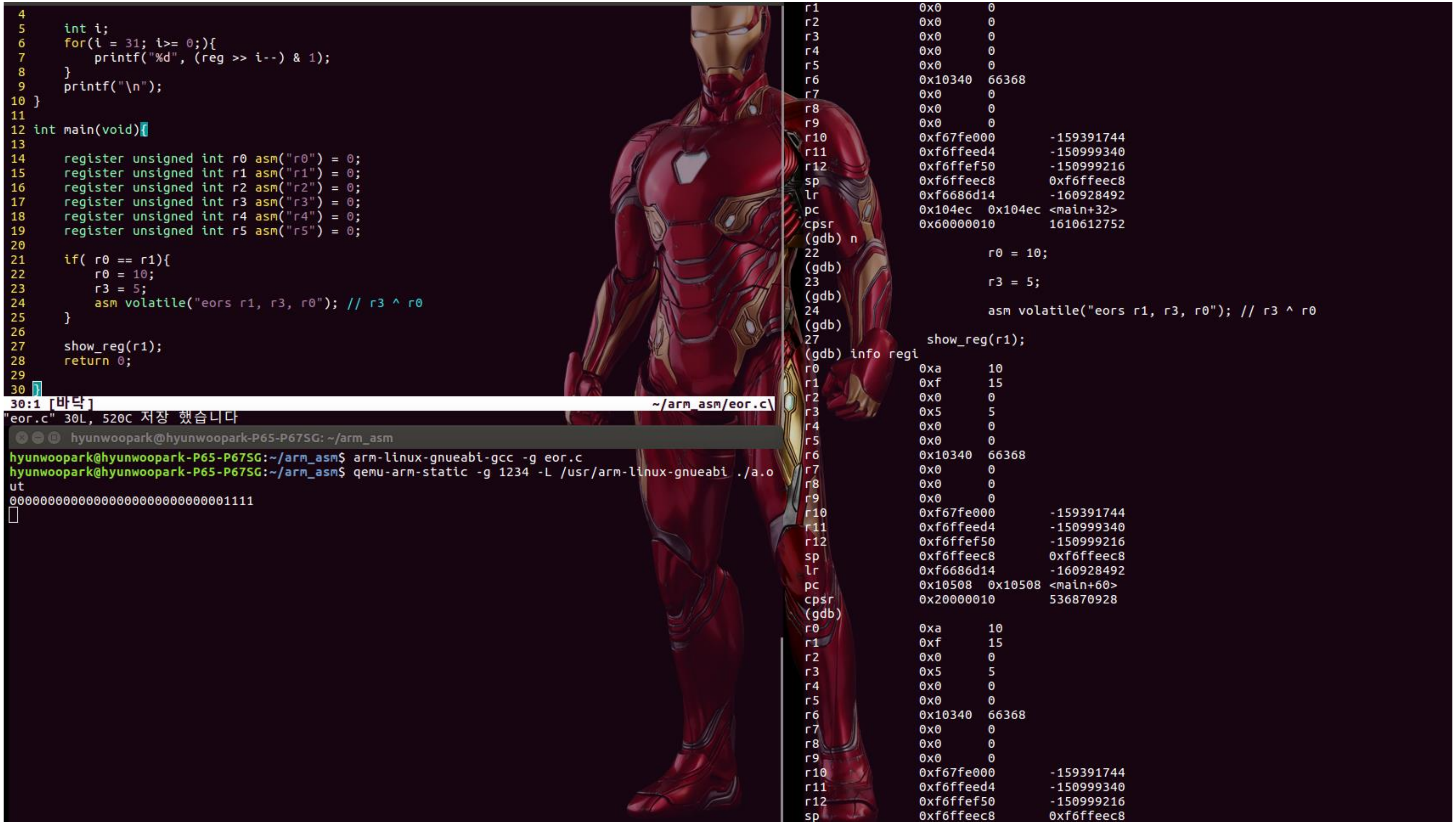


```
 1 #include<stdio.h>
 2 
 3 int main(void){
 4 
 5     register unsigned int r0 asm("r0") = 0;
 6     register unsigned int r1 asm("r1") = 0;
 7     register unsigned int r2 asm("r2") = 0;
 8     register unsigned int r3 asm("r3") = 0;
 9     register unsigned int r4 asm("r4") = 0;
10     register unsigned int r5 asm("r5") = 0;
11 
12     asm volatile("cmp r0, r1");  // r0 - r1  == 0  --> 0110
13     asm volatile("mov r2, #5");  // mov -> r2 = 5
14     asm volatile("cmp r0, r2");  // r0 - r1 != 0 --> 1000
15 
16     return 0;
17 }
~
~
~
~
~
~
~
~
~
~
 14:57 [모두]                                                        ~/arm_asm/cmp.c\
"cmp.c" 17L, 458C 저장 했습니다
```

```
hyunwoopark@hyunwoopark-P65-P67SG: ~/arm_asm
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ arm-linux-gnueabi-gcc -g cmp.c
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ qemu-arm-static -g 1234 -L /usr/arm-linux-gnueabi ./a.o
ut
```

```
r3             0x0      0
r4             0x0      0
r5             0x0      0
r6             0x102d8  66264
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeecc       0xf6ffeecc
lr             0xf6686d14       -160928492
pc             0x10420  0x10420 <main+32>
cpsr           0x60000010       1610612752
(gdb) n
13              asm volatile("mov r2, #5");
(gdb) info regi
r0             0x0      0
r1             0x0      0
r2             0x0      0
r3             0x0      0
r4             0x0      0
r5             0x0      0
r6             0x102d8  66264
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeecc       0xf6ffeecc
lr             0xf6686d14       -160928492
pc             0x10424  0x10424 <main+36>
cpsr           0x60000010       1610612752
(gdb) n
14              asm volatile("cmp r0, r2");
(gdb) n
16              return 0;
(gdb) info regi
r0             0x0      0
r1             0x0      0
r2             0x5      5
r3             0x0      0
r4             0x0      0
r5             0x0      0
r6             0x102d8  66264
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeecc       0xf6ffeecc
lr             0xf6686d14       -160928492
pc             0x1042c  0x1042c <main+44>
cpsr           0x80000010       -2147483632
(gdb) 
```

2. ARM ASM – eor

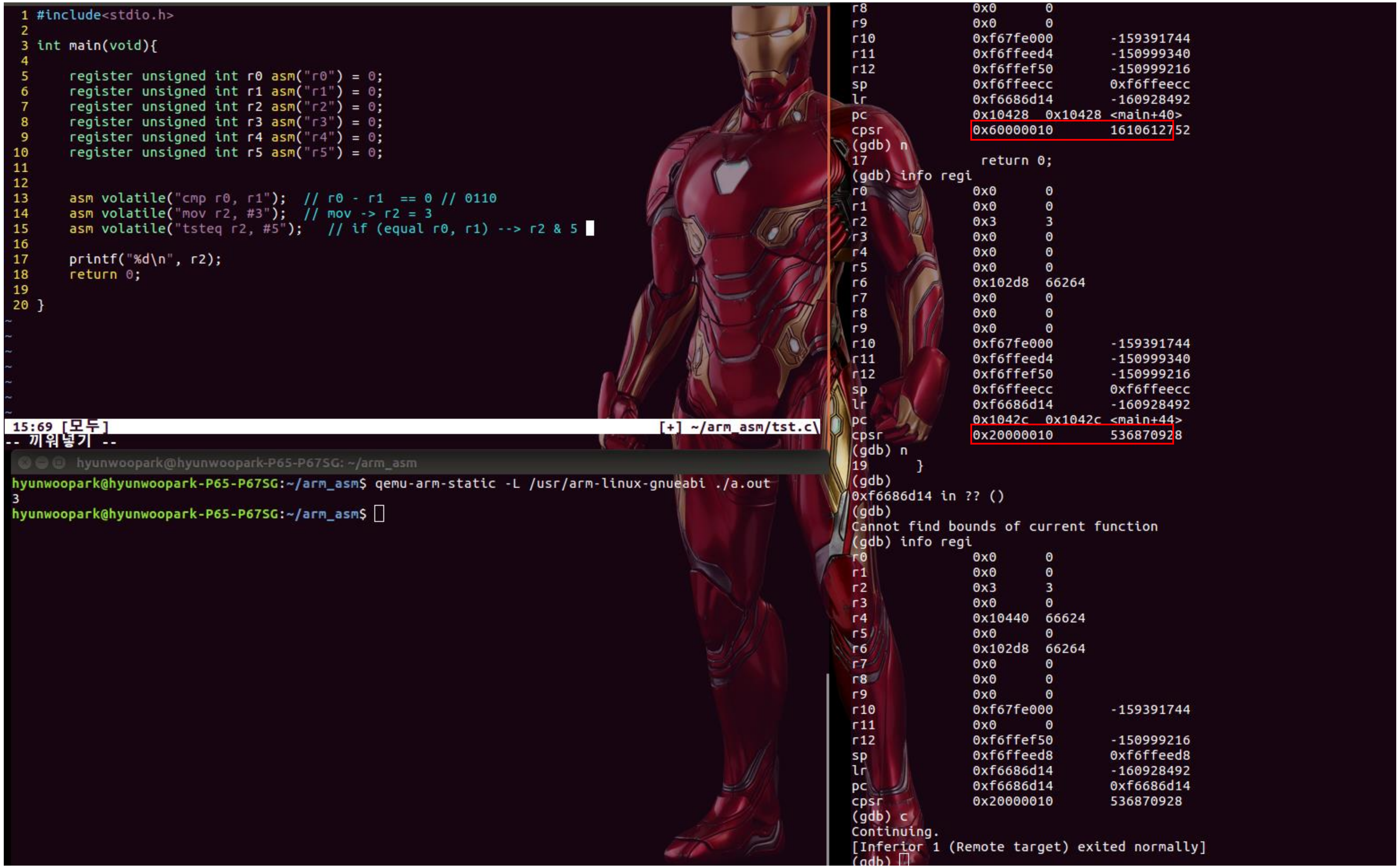


```

    int i;
    for(i = 31; i>= 0;){
        printf("%d", (reg >> i--) & 1);
    }
    printf("\n");
}

int main(void){

    register unsigned int r0 asm("r0") = 0;
    register unsigned int r1 asm("r1") = 0;
    register unsigned int r2 asm("r2") = 0;
    register unsigned int r3 asm("r3") = 0;
    register unsigned int r4 asm("r4") = 0;
    register unsigned int r5 asm("r5") = 0;

    if( r0 == r1){
        r0 = 10;
        r3 = 5;
        asm volatile("eors r1, r3, r0"); // r3 ^ r0
    }

    show_reg(r1);
    return 0;

}
30:1 [바닥]                                              ~/arm_asm/eor.c\
"eor.c" 30L, 520C 저장 했습니다
```

```
hyunwoopark@hyunwoopark-P65-P67SG: ~/arm_asm
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ arm-linux-gnueabi-gcc -g eor.c
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ qemu-arm-static -g 1234 -L /usr/arm-linux-gnueabi ./a.o
ut
00000000000000000000000000001111
```

```
r1             0x0      0
r2             0x0      0
r3             0x0      0
r4             0x0      0
r5             0x0      0
r6             0x10340  66368
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeec8       0xf6ffeec8
lr             0xf6686d14       -160928492
pc             0x104ec  0x104ec <main+32>
cpsr           0x60000010       1610612752
(gdb) n
22                      r0 = 10;
(gdb)
23                      r3 = 5;
(gdb)
24                      asm volatile("eors r1, r3, r0"); // r3 ^ r0
(gdb)
27              show_reg(r1);
(gdb) info regi
r0             0xa      10
r1             0xf      15
r2             0x0      0
r3             0x5      5
r4             0x0      0
r5             0x0      0
r6             0x10340  66368
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeec8       0xf6ffeec8
lr             0xf6686d14       -160928492
pc             0x10508  0x10508 <main+60>
cpsr           0x20000010       536870928
(gdb)
r0             0xa      10
r1             0xf      15
r2             0x0      0
r3             0x5      5
r4             0x0      0
r5             0x0      0
r6             0x10340  66368
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeec8       0xf6ffeec8
```

2. ARM ASM – eor

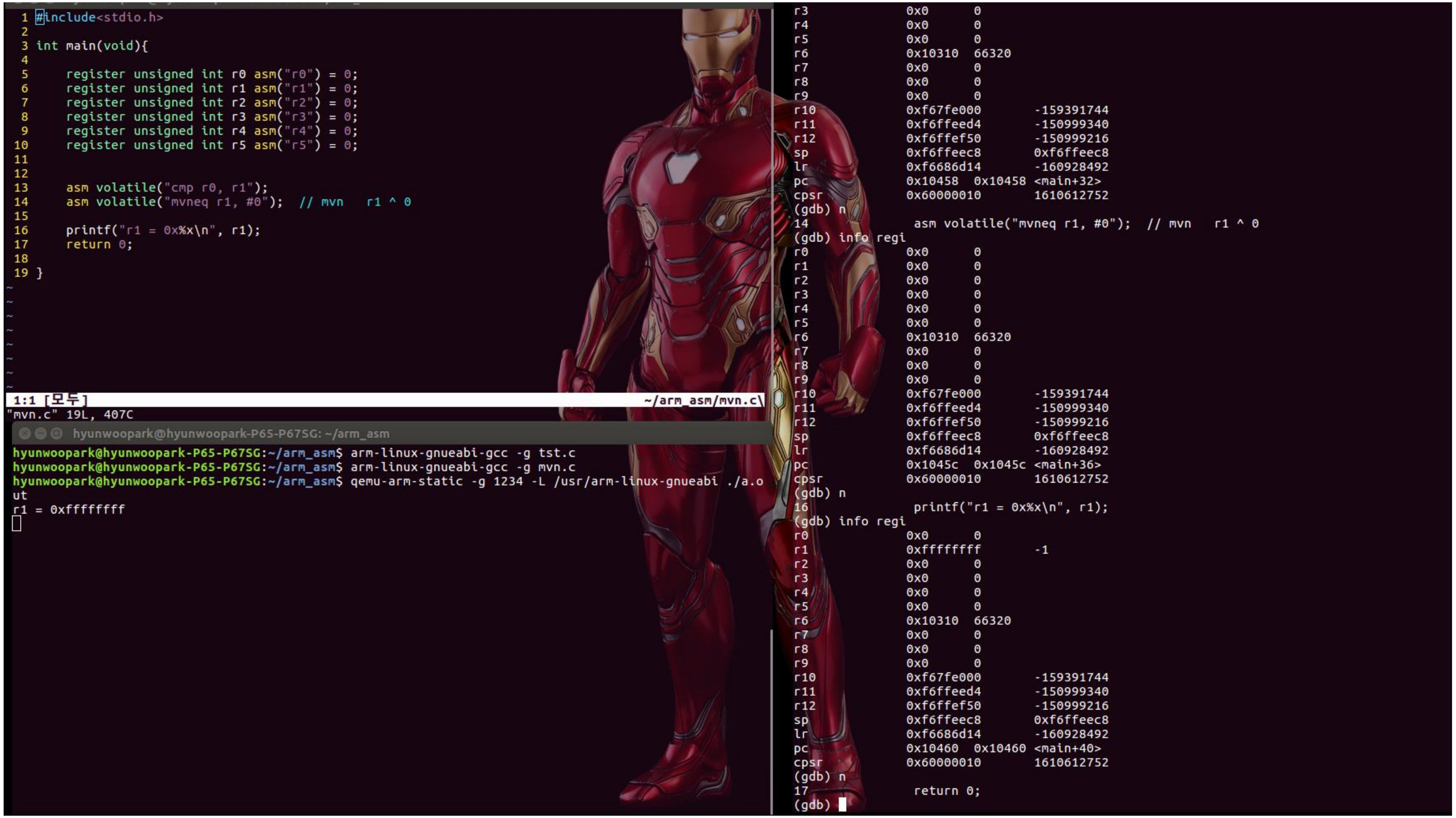


```c
#include<stdio.h>

int main(void){

    register unsigned int r0 asm("r0") = 0;
    register unsigned int r1 asm("r1") = 0;
    register unsigned int r2 asm("r2") = 0;
    register unsigned int r3 asm("r3") = 0;
    register unsigned int r4 asm("r4") = 0;
    register unsigned int r5 asm("r5") = 0;


    asm volatile("cmp r0, r1");  // r0 - r1  == 0 // 0110
    asm volatile("mov r2, #3");  // mov -> r2 = 3
    asm volatile("tsteq r2, #5");   // if (equal r0, r1) --> r2 & 5

    printf("%d\n", r2);
    return 0;

}
~
~
~
~
~
~
~
15:69 [모두]                                                  [+] ~/arm_asm/tst.c\
-- 끼워넣기 --
```

```
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ qemu-arm-static -L /usr/arm-linux-gnueabi ./a.out
3
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$
```

```
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeecc       0xf6ffeecc
lr             0xf6686d14       -160928492
pc             0x10428  0x10428 <main+40>
cpsr           0x60000010       1610612752
(gdb) n
17              return 0;
(gdb) info regi
r0             0x0      0
r1             0x0      0
r2             0x3      3
r3             0x0      0
r4             0x0      0
r5             0x0      0
r6             0x102d8  66264
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeecc       0xf6ffeecc
lr             0xf6686d14       -160928492
pc             0x1042c  0x1042c <main+44>
cpsr           0x20000010       536870928
(gdb) n
19      }
(gdb)
0xf6686d14 in ?? ()
(gdb)
Cannot find bounds of current function
(gdb) info regi
r0             0x0      0
r1             0x0      0
r2             0x3      3
r3             0x0      0
r4             0x10440  66624
r5             0x0      0
r6             0x102d8  66264
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0x0      0
r12            0xf6ffef50       -150999216
sp             0xf6ffeed8       0xf6ffeed8
lr             0xf6686d14       -160928492
pc             0xf6686d14       0xf6686d14
cpsr           0x20000010       536870928
(gdb) c
Continuing.
[Inferior 1 (Remote target) exited normally]
(gdb)
```

2. ARM ASM – eor



```
#include<stdio.h>

int main(void){

    register unsigned int r0 asm("r0") = 0;
    register unsigned int r1 asm("r1") = 0;
    register unsigned int r2 asm("r2") = 0;
    register unsigned int r3 asm("r3") = 0;
    register unsigned int r4 asm("r4") = 0;
    register unsigned int r5 asm("r5") = 0;


    asm volatile("cmp r0, r1");
    asm volatile("mvneq r1, #0");  // mvn   r1 ^ 0

    printf("r1 = 0x%x\n", r1);
    return 0;

}
~
~
~
~
~
~
~
~
 1:1 [모두]                                                  ~/arm_asm/mvn.c
"mvn.c" 19L, 407C
```

hyunwoopark@hyunwoopark-P65-P67SG: ~/arm_asm

```
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ arm-linux-gnueabi-gcc -g tst.c
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ arm-linux-gnueabi-gcc -g mvn.c
hyunwoopark@hyunwoopark-P65-P67SG:~/arm_asm$ qemu-arm-static -g 1234 -L /usr/arm-linux-gnueabi ./a.o
ut
r1 = 0xffffffff
```

```
r3             0x0      0
r4             0x0      0
r5             0x0      0
r6             0x10310  66320
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeec8       0xf6ffeec8
lr             0xf6686d14       -160928492
pc             0x10458  0x10458 <main+32>
cpsr           0x60000010       1610612752
(gdb) n
14              asm volatile("mvneq r1, #0");  // mvn   r1 ^ 0
(gdb) info regi
r0             0x0      0
r1             0x0      0
r2             0x0      0
r3             0x0      0
r4             0x0      0
r5             0x0      0
r6             0x10310  66320
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeec8       0xf6ffeec8
lr             0xf6686d14       -160928492
pc             0x1045c  0x1045c <main+36>
cpsr           0x60000010       1610612752
(gdb) n
16              printf("r1 = 0x%x\n", r1);
(gdb) info regi
r0             0x0      0
r1             0xffffffff       -1
r2             0x0      0
r3             0x0      0
r4             0x0      0
r5             0x0      0
r6             0x10310  66320
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0xf67fe000       -159391744
r11            0xf6ffeed4       -150999340
r12            0xf6ffef50       -150999216
sp             0xf6ffeec8       0xf6ffeec8
lr             0xf6686d14       -160928492
pc             0x10460  0x10460 <main+40>
cpsr           0x60000010       1610612752
(gdb) n
17              return 0;
(gdb) 
```