Xilinx Zynq FPGA,TI DSP,
MCU 기반의
프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – 정한별
hanbulkr@gmail.com

<15 일차 정한별 >

디버그 하기

```
#include<stdio.h>

int myfunc(int num){
 // return num * 2;
  return num + 3;
 // return num << 1;
}

int main(void){
        int num = 3, res;
        res = myfunc(num);
        printf("res=%d\n",res);

        return 0;
}
```

-O0 대문자 O 숫자 0은 넣으면 최적화는 안한 debug 파일이 나온다.

① gcc -g -o 다른이름 컴파일할 C 파일
- -g → debug 용으로 만듦.
- -o → 다른이름 지정

② gdb debug — debug를 디버깅모드로 보겠다!
- debug → 이름

③ b main — main에 break point를 잡겠다.

④ r — run 해서 동작을 시작한다.

⑤ disas — display assembly 하겠다!

⑥ b* 주소 — 브레이크를 주소 쓴 곳 앞으로 걸어준다. (rip)를 옮겨준 것.

⑦ run 하고 disas 를 해서 rip가 옮겨진 위치를 볼 수 있다.

⑧ 다음 이셈 명령어를 시작하려면 ni, si 를 눌러 실행한다.
- n은 next로 다음 줄로 가져 함수 내부로 들어가진 않는다.
- s는 step으로 한 단계 다음 실행할 함수 내부로 이동한다.

⑨ print 명령어 → $rsp, $rbp 를 출력
- p/x 16진수
- p/d 정수 10진수
- p/c 문자. (아스키)
- p/s 문자열.

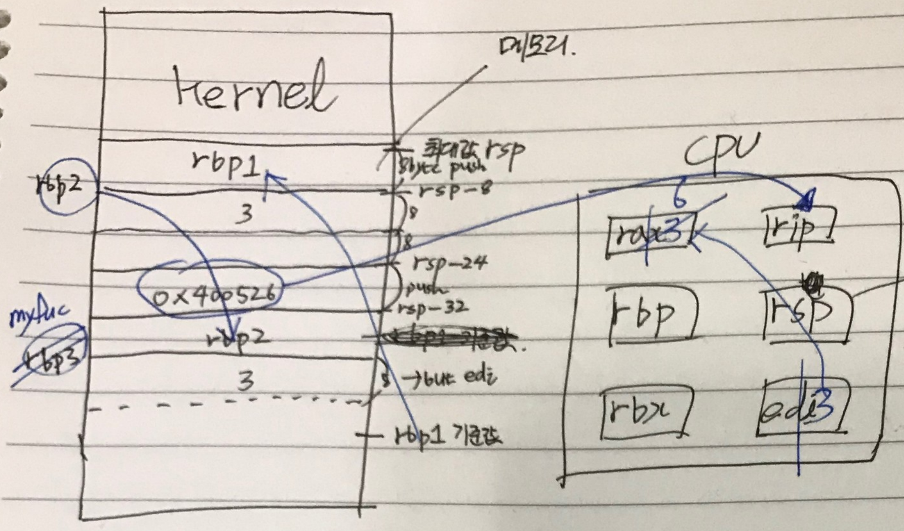
⑩ c는 break point 까지 이동

⑪ bt는 오류가 발생한 함수를 역으로 찾아간다

⑫ x $rsp → x는 메모리를 조사하는 것.

x 한 줄 실행하면 포인터는 8byte

디버깅 오류시.
① b main
② r
③ c
④ bt
순으로 하면 된다.



rsp는 (push pop) 에서만 직접 건드려 값을 바꿀 수 있다.

Callq = push + jmp

retq = pop + rip pop rip

pop %rbp == pop은 스택의 제일 높은 곳에 들어간다 (낮은)