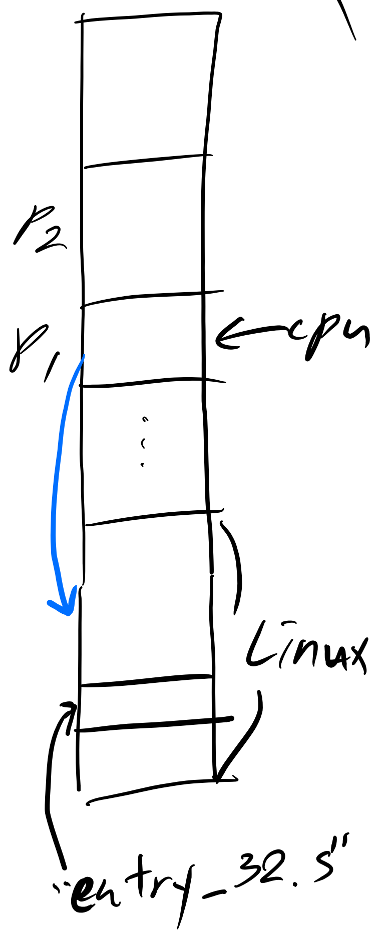
System call interrupt

- system call
- ia32_sysenter_target
- making new system call.



$\langle P_1 \rightarrow INT\ x \rightarrow ISR_1 \rightarrow ISR_2 \rightarrow P_1 \rangle$

X INT

① HW INT:

array 형.

$P_1 \rightarrow$ key press $\rightarrow$ INT 33 $\rightarrow$ <u>interrupt</u>[ ] $\rightarrow$ atkbd_interrupt() $\rightarrow P_1$

② Exception INT:

$P_1 \rightarrow$ % 0 $\rightarrow$ INT ∅ $\rightarrow$ divide_error $\rightarrow$ do_divide_error() $\rightarrow$ ~~P1~~ $\rightarrow P_2$

③ System call INT

<프로그램이 운영체제 관련 함수를 호출했을 때>

↳ ① $P_1 \rightarrow$ printf() INT128 →

write()

syscall_call → sys_write()

→ P_1 (=ISR1)

② $P_1 \rightarrow$ printf()

↓

write() → SYSENTER

→ ia32_system_target

→ sys_write() → P_1

얘가 ①보다
좀 더 빠르다!!
ㅋMS사에서
만들어 놓은
알고리즘임!

※ syscall_call :

...

...

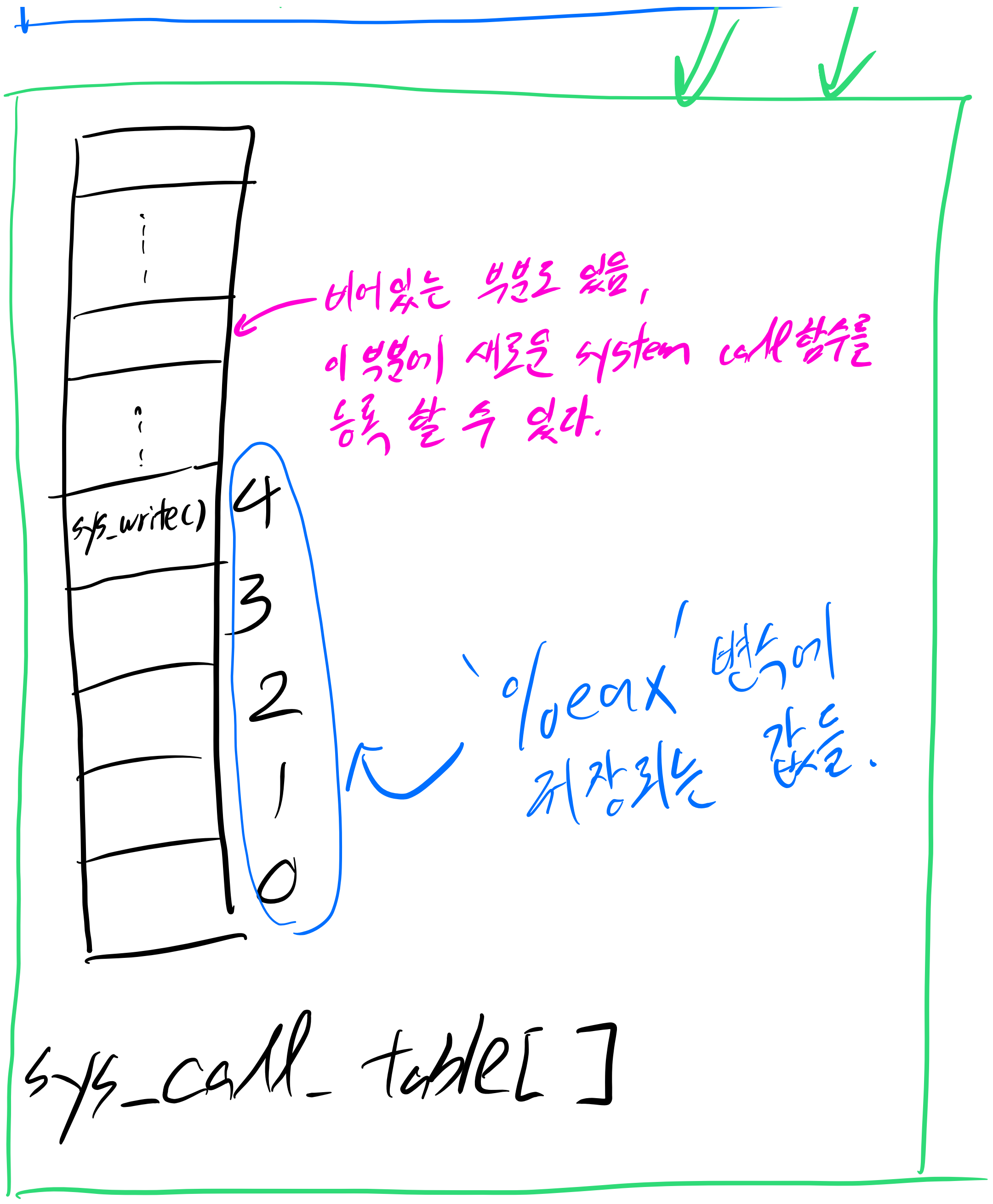
call *sys_call_table(,%eax,4)

ia32_sysenter_target :

...

...

call *sys_call_table(%eax,4)



비어있는 부분도 있음,
이 부분에 새로운 system call 함수를
등록 할 수 있다.

'%eax' 변수에
저장되는 값들.

_/ (8)

-hw(…)

· System call in interrupt.

· printf( ) ⇒ sys_write( )

즉, 뭔가를 입력할 때는 이 함수가 호출된다!

-hw(9)

① write(1, "hi", 2) → $\overline{|\text{hi}}$

|| ↓

syscall(4, 1, "hi", 2) → $\overline{|\text{hi}}$

↓

sys_write(···)

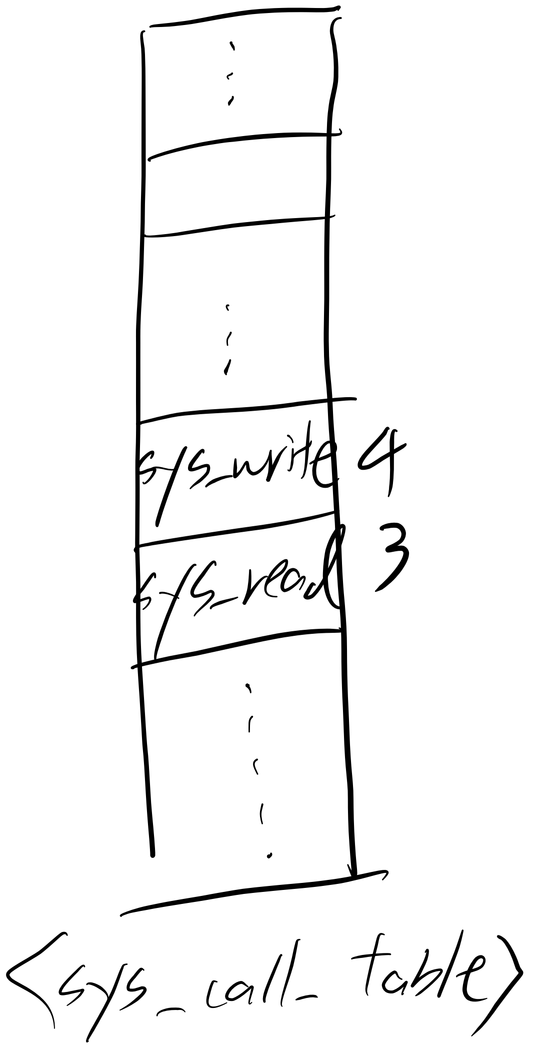
② read(0, buf, n) :

|| ↓

syscall(3, 0, buf, n).

↓

sys_read(...)



※ 어떤 사용자 함수를 사용해야 특정 system call 함수가 호출되는지 알아야 한다!!

→ ~~sys~~_write : 보통 system call 함수명의 앞 "sys"를 뺀 것이, 해당 system call 함수를 호출하는 사용자 함수명이다.