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

<리눅스 커널>

2018.04.19 - 41 일차

강사 - 이상훈

gcccompil3r@gmail.com

학생 - 안상재

sangjae2015@naver.com

<커널 내부 소스코드 드라이빙>

1. read_cr0() 함수 분석

1) arch/x86/include/asm/special_insns.h 파일로 이동

2) 아래 그림의 native_read_cr0 함수로 이동

```
static inline unsigned long read_cr0(void)
{
    return native_read_cr0();
}
```

3) native_read_cr0 함수의 정의 부

```
static inline unsigned long native_read_cr0(void)
{
    unsigned long val;
    asm volatile("mov %%cr0,%0\n\t" : "=r" (val), "=m" (__force_order));
    return val;
}
```

→ val 이 %0 으로 대입됨.
→ cr0 레지스터 값이 val 로 들어가게 되므로 read 가 가능하게 됨. (mov 연산은 왼쪽에서 오른쪽으로 이동)
→ 메모리의 값은 __force_order 로 갱신하게 됨.
→ “=r” 은 레지스터, “=m” 은 메모리 를 의미함.

2. write_cr0() 함수 분석

```
static inline void native_write_cr0(unsigned long val)
{
    asm volatile("mov %0,%%cr0": : "r" (val), "m" (__force_order));
}
```

→ val 값이 %0 으로 들어가게 되고 val 이 cr0 에 write 가 됨.

3. kallsyms_lookup_name() 함수 분석
1) 커널 내부에서 인자로 들어온 이름의 심볼을 모두 찾는 함수

```
unsigned long kallsyms_lookup_name(const char *name)
{
    char namebuf[KSYM_NAME_LEN];
    unsigned long i;
    unsigned int off;

    for (i = 0, off = 0; i < kallsyms_num_syms; i++) {
        off = kallsyms_expand_symbol(off, namebuf, ARRAY_SIZE(namebuf));

        if (strcmp(namebuf, name) == 0)
            return kallsyms_addresses[i];
    }
    return module_kallsyms_lookup_name(name);
}
```

2) kallsyms_lookup_name 안의 kallsyms_expand_symbol 함수로 들어감.
→ 찾고 있는 심볼 값이 있는지 루프를 돌면서 체크함.

```
static unsigned int kallsyms_expand_symbol(unsigned int off,
                       char *result, size_t maxlen)
{
    int len, skipped_first = 0;
    const u8 *tptr, *data;

    /* Get the compressed symbol length from the first symbol byte. */
    data = &kallsyms_names[off];
    len = *data;
    data++;

    /*
     * Update the offset to return the offset for the next symbol on
     * the compressed stream.
     */
    off += len + 1;

    /*
     * For every byte on the compressed symbol data, copy the table
     * entry for that byte.
     */
    while (len) {   // 심볼 값이 있는지 찾는 중
        tptr = &kallsyms_token_table[kallsyms_token_index[*data]];
        data++;
        len--;

        while (*tptr) { // 드라이버 중에 심볼이 있는지 찾음
            if (skipped_first) {
                if (maxlen <= 1)
                    goto tail;
                *result = *tptr;
                result++;
                maxlen--;
            } else
                skipped_first = 1;
            tptr++;
        }
    }

tail:
    if (maxlen)
        *result = '\0';

    /* Return to offset to the next symbol. */
    return off;
}
```

4. set_memory_rw 심볼 분석

1) set_memory_rw 함수의 정의부

```
int set_memory_rw(unsigned long addr, int numpages)
{
    return change_page_attr_set(&addr, numpages, __pgprot(_PAGE_RW), 0);
}
```

2) _PAGE_RW 의 정의

```
#define _PAGE_RW    (_AT(pteval_t, 1) << _PAGE_BIT_RW)
```

→ _AT 함수 정의

```
#ifdef __ASSEMBLY__
#define _AC(X,Y)    X
#define _AT(T,X)    X
#else
#define __AC(X,Y)   (X##Y)
#define _AC(X,Y)    __AC(X,Y)
#define _AT(T,X)    ((T)(X))
#endif
```

4) __pgprot 함수정의

```
#define pgprot_val(x)   ((x).pgprot)
#define __pgprot(x) ((pgprot_t) { (x) } )
```

5) change_page_attr_set 함수의 정의

```
static inline int change_page_attr_set(unsigned long *addr, int numpages,
                       pgprot_t mask, int array)
{
    return change_page_attr_set_clr(addr, numpages, mask, __pgprot(0), 0,
        (array ? CPA_ARRAY : 0), NULL);
}
```