Xilinx Zynq FPGA, TI DSP, MCU 기반의 프로그래밍 및 회로 설계 전문가 과정

강사 – Innova Lee(이상훈)

gcccompil3r@gmail.com

학생 – hoseong Lee(이호성)

hslee00001@naver.com

OS없이 pwm, gpio, adc 등을 모두 동작시키면 우리가 원하는 데로 동작하지 않고 모두 꼬이게 된다. OS가 서포트 해주는 역할 중에 가장 중요한것은 context switching 과 멀티태스킹이 무엇을 기반으로 정의한 것일까? 스케쥴링이다.

(각 프로세스 별로 얼마나 시간을 줄 것인지.. 만약 시간이 만료되면 그때 context switching을 하고 제어권을 넘겨주게되는 것)

결론은 스케쥴링이 안돼면 모두 개털이 된다.

* 그러면 스케쥴링은 어떤 것일까?

→ 타이밍을 모두 계산해준다.

결론은 운영체제를 쓰라는 것이다.

우리가 사용하는 펌웨어에는 캐시가 있다. 캐시가 접근하면 메모리가 접근하지 않는다. 만약 캐시가 접근하지 않으면 메모리가 접근하게 된다. 이때 메모리가 접근하는 시간과 메모리가 접근하지 않는 시간 값은 다르다. → 어떻게 처리할까? 알 수 없다. 캐시가 접근할지 안할지는 cpu밖에 모르므로 답이 없다. 한마디로 작업을 제어하기위해서는 스케쥴링이 필요하다. 스케쥴링을 하려면 운영체제가 필요하다.

zimage는 리눅스 커널 압축된 이미지이다. (스케줄링, 파일시스템, 메모리 관리기능) 밑의 기능들을 모두 포함함.

운영체제뿐아니라 모든 하드웨어들은 전원이 들어가면 모든 장치들을 초기화 시킨다.

→ 초기화를 반드시 해야함. (0도 아니고 1도아닌 상태인 프로팅 상태(쓰레기값)에 있으므로 초기화를 하지 않으면 정상적인 역할을 할 수 없다.)

Bootload가 이 초기화를 담당한다. 그리고, 뭐가 없으면 c언어를 돌릴 수 없을까? 스택이다.부트로드가 가상메모리 레이아웃을 잡는다. 그다음 리눅스 커널을 올린다.

반면 펌웨어(코어텍스 a, avr, 코어텍스 r등) MMU가 없을 때는 가상메모리를 사용하지 않는다. 가상 메모리가 없으면 물리 메모리에 직접 맵핑하게 된다.

Root filesystem은 linux porting 을 할 수 있는 시스템들만 가질 수 있다. 루트파일 시스템안에는 디스크에 들어있는 구조를 트리형태로 들어 있다. 리눅스 커널을 올리고 그다음에는 운영체제가 관리하는 5대요소

1.파일시스템, 2메모리, 3 네트워크 4 디바이스 5 프로세스(디스크) 이 커널을 관리하는 것들이 zimage에 들어간다.

시스템 디자인 - A라는 신호가 들어왔을 때, 이장치를 어떤식으로 구동하게 할 것인가 자체를 말한다. → fpga설계

디바이스 드라이버 - 어떤장치가 꽂히면 이 장치를 분석하고, 이 장치를 구동 하기위해 필요한 소프트웨어가 돌아간다. 이것 덕분에 시스템 디자인에만 집중 할 수 있게 해준다. → 이는 os전문가가 한다.

데몬을 만든다. 확인은?

Ps -ef grep → 섹션부분이 물음표로 나왔다. 이 물음표 나온 부분을 보고 아 이부분은 데몬이구나 생각 했었다.

컴퓨터 전원을 누르기 전까지 이 데몬은 죽지 않는다. 데몬은 일을 시키지 않더라도 해야하는 일이 생기면 Runnig 상태가 되어 자신에게 주어진 일을 수행한다. → 데몬은 강제로 작업을 쉬라고 할 수 있다. 시그널이 들어오면 데몬을 깨울 수 있는 것 ,또한 리얼타임 시그널 가지고 특정상황에서만 동작하도록 할 수 있다.

35쪽 cpu에 플랫폼에 따라 바뀌어야 하는 것이 arch에 들어 있따!!

```
lee@lee-Lenovo-YOGA-720-13IKB:~/kernel/linux-4.4$ ls arch
Kconfig  arc  arm64  blackfin  cris  h8300    ia64  m68k   microblaze  mn10300  openrisc  powerpc  score  sparc  um        x86
alpha    arm  avr32  c6x       frv   hexagon  m32r  metag  mips        nios2    parisc    s390     sh     tile   unicore32  xtensa
```

openrisc

x86 → intel cpu

c6x → ti dxp

ia64 → intel에 멸망당한

microblaze → fpga에서 구동되는 가상의 cpu (소프트웨어 cpu)

powerpc → 자동차

arm64 → arm도 64비트가 나옴 차량용

avr32 → 우리도 리눅스를 사용해보잣 망하지말잣

nios2 → altera

- 우리는 코어텍스a , dxp, fpga (arm64, arm, c6x) 를 다루게된다

```
lee@lee-Lenovo-YOGA-720-13IKB:~/kernel/linux-4.4$ ls arch/arm/
Kconfig         kernel          mach-cns3xxx     mach-highbank    mach-lpc18xx     mach-nomadik     mach-rockchip    mach-stm32       mach-zx          plat-samsung
Kconfig-nommu   kvm             mach-davinci     mach-hisi        mach-lpc32xx     mach-nspire      mach-rpc         mach-sunxi       mach-zynq        plat-versatile
Kconfig.debug   lib             mach-digicolor   mach-imx         mach-mediatek    mach-omap1       mach-s3c24xx     mach-tegra       mm               probes
Makefile        mach-alpine     mach-dove        mach-integrator  mach-meson       mach-omap2       mach-s3c64xx     mach-u300        net              tools
boot            mach-asm9260    mach-ebsa110     mach-iop13xx     mach-mmp         mach-orion5x     mach-s5pv210     mach-uniphier    nwfpe            vdso
common          mach-at91       mach-efm32       mach-iop32x      mach-moxart      mach-picoxcell   mach-sa1100      mach-ux500       oprofile         vfp
configs         mach-axxia      mach-ep93xx      mach-iop33x      mach-mv78xx0     mach-prima2      mach-shmobile    mach-versatile   plat-iop         xen
crypto          mach-bcm        mach-exynos      mach-ixp4xx      mach-mvebu       mach-pxa         mach-socfpga     mach-vexpress    plat-omap
firmware        mach-berlin     mach-footbridge  mach-keystone    mach-mxs         mach-qcom        mach-spear       mach-vt8500      plat-orion
include         mach-clps711x   mach-gemini      mach-ks8695      mach-netx        mach-realview    mach-sti         mach-w90x900     plat-pxa
```

arm 하위 호환이안돼다?

그래서 많다

mach-omap1,2 → ti

mach-zynq → 자일링스

mach-s3c24xx → 삼성

mach-keystone → 투는 레이더용임.

mach-lpc32xx → 차량용

mach-bcm → 라즈베리파이

mach-davinci → ti - 블랙박스용 - cctv 등

mach-stm32 →

mach-tegra → NVIDA에서 사용하는 리눅스플랫폼

→ arch에 있는 커널이 하는 일은 구동시키고 있는

```
lee@lee-Lenovo-YOGA-720-13IKB:~/kernel/linux-4.4$ ls
COPYING        Kbuild       Makefile        arch   crypto        drivers   include  kernel  net       security  tools
CREDITS        Kconfig      README          block  cscope.files  firmware  init     lib     samples   sound     usr
Documentation  MAINTAINERS  REPORTING-BUGS  certs  cscope.out    fs        ipc      mm      scripts   tags      virt
```

tools -

block - block device에 관련되 있는

init - 커널의 초기화

usr - 유저가 사용할 때 관계되는 정보들

certs - 암호화 알고리즘

ipc - mute, lock 등

scripts - 커널을 구동시키는 스크립트

virt - 운영체제위에 운영체제 즉, 클라우드

drivers - device driver

task 관리자

```
#ifdef CONFIG_COMPAT_BRK
        unsigned brk_randomized:1;
#endif

        unsigned long atomic_flags; /* Flags needing atomic access. */

        struct restart_block restart_block;

        pid_t pid;
        pid_t tgid;

#ifdef CONFIG_CC_STACKPROTECTOR
```

pid

tgid

→ 태스크 생성, 실행, 상태전이

ipc , 스케줄링, 시그널 처리등

```
lee@lee-Lenovo-YOGA-720-13IKB:~/my_proj/lhs/network/4_4/load$ file serv
serv: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=258ba
be7b5c549655063cab32aec05334ea49f1f, not stripped
lee@lee-Lenovo-YOGA-720-13IKB:~/my_proj/lhs/network/4_4/load$ readelf -h serv
ELF Header:
  Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF64
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x400b50
  Start of program headers:          64 (bytes into file)
  Start of section headers:          11992 (bytes into file)
  Flags:                             0x0
  Size of this header:               64 (bytes)
  Size of program headers:           56 (bytes)
  Number of program headers:         9
  Size of section headers:           64 (bytes)
  Number of section headers:         31
  Section header string table index: 28
```

mm_struct 로 들어가기

pgd - 페이지 테이블을 관리

start_brk, brk, start_stack; // 8byte 안에 어디에 오느냐 하면 힙영역인지 스택영역인지 등 파악해서 페이징하는 것

메모리 관리자

→ 물리 메모리 관리, 가상메모리관리, 세그먼테이션, 페이징, 세그먼테이션 폴트처리

파일 시스템 관리자

→ 파일 생성, 접근제어, inode관리, 디렉토리 관리, super block관리

네트워크 관리자

→ 소켓 인터페이스, tcp/ip 등의 통신 프로토콜

디바이스 드라이버

→ 디스크,터미널, cd, 랜카드 등 주변장치 드라이버

task란??

Vi –t task_struct 명령어로 task_struct 내부를 살펴보게 되면 pid 와 tgid 가 선언 되어있는 것을 볼 수 있다. pid 는 프로세스 아이디이고, tgid 는 쓰레드 그룹 아이디이다. 프로세스와 쓰레드의 차이점은 pid 와 tgid 가 같으면 프로세스이고 pid 와 tgid 가 다르면 쓰레드로 프로세스 구성원이다. 쓰레드 리더일 때 프로세스라고 볼 수 있다.

실행파일 포맷

PE - 윈도우

LF64 - 엘프 리눅스 ,유닉스

DWF64 - 드워프 윈도우