Embedded Systems

Ch 14A
Linux Kernel

Byung Kook Kim
Dept of EECS
Korea Advanced Institute of Science and Technology

Overview

- *1. Bootloader*
- *2. EzBoot*
- *3. Boot Sequence*

1. Bootloader

- **부트로더(Bootloader)란?**
  - A small program that loads the operating system into the computer memory when the system is booted and also starts the operating system (Webopedia)
  - Briefly, *boot loader* is the first software program that runs when a computer starts. It is responsible for loading and transferring control to the operating system *kernel* software (such as Linux kernel). The kernel, in turn, initializes the rest of the operating system (such as RedHat 9.0) (Gnu).
  - Examples
    - BIOS (Basic Input Output System) in PC
    - LILO (Linux Loader)
    - GNU GRUB is a multiboot boot loader. It was derived from GRUB, **GRand Unified Bootloader**
    - EzBoot in EZ-X5 board.

2. EzBoot

- 기능
  - 하드웨어에 대한 초기화 (CPI speed, 메모리, 인터럽트, UART serial 관련)
  - Kernel 이미지나 Root 이미지 (Ram disk image) 를 다운로딩 할 수 있는 기능 (Serial 및 tftp 이용) 및 다운로드한 image를 flash memory에 쓸 수 있는 기능 (Non-volatile storage)
  - 리눅스를 부팅 할 수 있게 해주는 기능.

- 구성
  - Start.S: Boot sequence assembly program
  - Main.c: C program

EzBoot (II)

- **EZBOOT implementation**
  - 기존에 공개된 부트로더의 장점을 취합하여 이지보드에 최적화된 부트로더
  - Blob 과 요피 부트로더와 PPC부트를 참조하여 작성되었다.

- **EzBoot의 기능**
  - 도움말 기능
  - 메모리 읽고 쓰기
  - 특정 번지로 이동
  - 커널 로드 및 부팅
  - ZMODEM을 이용한 이미지 다운로드 기능
  - TFTP를 이용한 이미지 다운로드 기능
  - 플래쉬 기록 및 읽기
  - 기본 환경 설정
  - 커널 전달 파라메터 설정
  - 명령어 히스토리 기능

EzBoot (III)

- **Memory map of EZ-X5 board**
  - Boot-flash (Static chip select 0) 512 Kbytes
    0x0000 0000 – 0x0008 0000
    - EZBOOT area 128 Kbytes 0x0000 0000 – 0x0002 0000
    - EZBOOT parameter area 64 Kbytes 0x0002 0000 – 0x0003 0000
  - NAND Flash memory (Static chip select 2) 64 Mbytes
    0x0800 0000 – 0x0C00 0000?
    - Kernel image 1 Mbytes
    - Compressed RAM disk image 3 Mbytes
    - Application area (yaffs) Rest
  - SDRAM (SDRAM Bank 0) 64 Mbytes
    0xA0000 0000 – 0xA400 0000
    - Kernel boot parameter 256 bytes 0xA000 0000 – 0xA000 0100
    - Kernel image 8M bytes 0xA000 0100 – 0xA080 0000
    - RAM disk image 7M bytes 0xA080 0000 – 0xA0F0 0000
    - EZBOOT Bootloader 128 Kbytes 0xA0F0 0000 – 0xA0F2 0000
    - EZBOOT stack area 796 Kbytes 0xA0F20000 – 0xA100 0000
    - EZBOOT working area Rest 0xA100 0000 – 0xA400 0000

EzBoot (IV)

- **EzBoot 명령어의 특징**
  - 명령어는 대소문자를 가리지 않는다.
  - 매우 게으른 프로그래머를 위한 명령 체계를 가지고 있다.
    - 기존의 부트로더와 다르게 약어를 사용한다.
    - 실제 부트로더를 쓰게 되면 같은 명령어를 계속 입력하여야 하는데, 3 글자 이하로 사용이 가능하도록 설계되어 있다.
  - 환경 파일이 따로 존재하며, 이곳에서 필요한 파일 이름과 IP 등을 수정, 저장하여 사용한다.
  - 명령어 히스토리 기능
    - 16개까지 이전에 입력했던 명령어를 기억한다.
    - 이전 히스토리로 이동은 CTRL-Z 키로
    - 이후 히스토리로 이동은 CTRL-X 키로

EzBoot (V)

- **EzBoot Commands (I)**
  - **HELP or ?** Display help message
  - **MD [start-address] [count]: 메모리나 플래시의 내용을 읽어 표시**
    - MD 16 DWORD **읽어** 4line**에 표시하고 주소를** 16**증가 시킨다**.
    - MD start-addr 16 DWORD **읽고 넘겨진 주소를** 16**증가 시킨다**.
    - MD start-addr count count **만큼 읽고 넘겨진 주소를** count **만큼 증가 시킨다**.
  - **MW[B/W/L] address value: 메모리에 값을 쓴다**.
    - [B/W/L] Byte/Word/Long
    - MWL 0x001234 0x88 **주소** 0x001234**에** Long 0x88**을 쓴다**.
    - MWB 200 10 **주소** 0xC8**에** Byte 0x0A**를 쓴다**.
  - **RST: 소프트웨어적으로 프로그램을 리셋한다**.
    - **하드웨어는 리셋되지 않는다**.
  - **GK: 커널을 실행한다**.
    - **이 명령이 실행되기 전에 램의 커널영역에 커널이미지가 존재하여야 한다**.
  - **GO addr: 주어진 주소로 프로그램카운트가 이동한다**.
    - TM, ZM **명령으로 특정주소로** image**를 받은 후** test!

EzBoot (VI)

- **EzBoot Commands (II)**
  - **PING ddd.ddd.ddd.ddd [count]** ping 명령을 실행한다.
    - ping의 회수는 기본 3회이다.
  - **ARP ddd.ddd.ddd.ddd** Host의 MAC 주소를 얻는다.
    - Host IP가 설정되어 있어야 한다.
  - **FM[K/R]** 플래시의 내용을 메모리로 복사한다.
    - [K/R] K : 커널이미지, R : 램디스크이미지
    - FMK : 플래시에서 압축된 커널을 메모리로 복사한다.
    - FMR : 플래시에서 압축된 램디스크를 메모리로 복사한다.
    - FM flash-addr sdram-addr count : 플래시의 주소에서 메모리의 주소로 카운트 만큼 복사한다.
  - **MF[K/R]** 메모리의 내용을 플래시로 복사한다.
    - [K/R] : K : 커널이미지, R : 램디스크이미지
    - MFK : 메모리에서 압축된 커널영역을 플래시로 복사한다.
    - MFR : 메모리에서 압축된 램디스크영역을 플래시로 복사한다.
    - MF sdram-addr flash-addr count : 메모리의 주소에서 플래시 주소로 카운트 만큼 복사한다.

EzBoot (VII)

- **EzBoot Commands (III)**
  - **Z[M/F][B/K/R]** 씨리얼 Z-모뎀프로토콜로 파일을 다운로드한다.
    - [M/F] M : 메모리 F : 플래시
    - [B/K/R] B : 부트로더 K : 커널이미지 R : 램디스크이미지
    - ZMK 지정된 메모리 주소에 커널을 받는다.
    - ZMR 지정된 메모리 주소에 램디스크를 받는다.
    - ZFK 지정된 플래시 주소에 커널을 받는다.
    - ZFR 지정된 플래시 주소에 램디스크를 받는다.
    - ZFB 0x00000000 (플래시)에 부트로더를 받는다.
    - ZM sdram_addr 지정된 메모리 주소에 파일을 받는다.
    - ZF flash_addr 지정된 플래시 주소에 파일을 받는다.
  - **T[M/F][B/K/R]** 이더넷 tftp 프로토콜로 파일을 다운로드한다.
    - [M/F] M : 메모리 F : 플래시
    - [B/K/R] B : 부트로더 K : 커널 이미지 R : 램디스크 이미지
    - TMK 지정된 메모리 주소에 커널을 받는다.
    - TMR 지정된 메모리 주소에 램디스크를 받는다.
    - TFK 지정된 플래시 주소에 커널을 받는다.
    - TFR 지정된 플래시 주소에 램디스크를 받는다.
    - TFB 0x00000000 (플래시)에 부트로더를 받는다.
    - TM sdram_addr file : 지정된 메모리 주소에 파일을 받는다.
    - TF flash_addr file : 지정된 플래시 주소에 파일을 받는다.

EzBoot (VIII)

- **EZBoot Commands (IV)**
  - FEK NAND flash 의 kernel partition을 지운다
  - FER NAND flash 의 Ram disk partition을 지운다
  - SET 환경설정모드로 들어간다
    - MAC address [00:a2:55:f2:26:25]
    - Local IP [192.168.10.155]
    - Host IP [192.168.10.1]
    - Host tftp directory
    - zImage filename [zImage.x5]
    - Ramdisk filename [ramdisk.x5.gz]
    - Bootloader filename [ezboot.x5]
    - Autoboot wait time [3]
    - Boot menu key [ ]
    - Copy ramdisk [Y]
    - Architecture number [303]
    - Serial FF/BT/ST [2: Standard]
    - Kernel command 1st
    - Kernel command 2nd
    - Kernel command 3rd
    - Load default
    - Apply & Exit
    - Save
    - Exit

3. Boot Sequence

- **Reset**
  - Reset exception: Starts at address 0x00000000
    - EzBoot: entry.S, main.c
  - Program entry.S performs
    - Mask all interrupts
    - Determine the CPU speed
    - Initialize LED display
    - Activate instruction cache
    - Initialize memory
    - Test boot loader area in memory
    - Copy the bootloader to memory
    - Initialize the stack pointer
    - Jump to main.c

Boot Sequence (II)

- **Start.S (I)**

```
/*=====================================================================
  화일명 : start.S
  설  명 : 리셋시 호출되는 루틴
=====================================================================*/
#include   <ez_x5.h>
.text
//---------------------------------------------------------------------
//  인터럽트  벡터 테이블
//---------------------------------------------------------------------
.globl                  _start
_start:                 b       reset                   // MCU가 리셋된후 실행된다.
                        b       undefined_instruction   // 정의되지 않는 명령이 수행 되었을때 실행
                        b       software_interrupt      // SWI가 수행되었을 때 실행된다.
                        b       prefetch_abort          // 데이타를 프리 패치 버스 에러가 발생시.
                        b       data_abort              // 데이타 에러가 발생하면 실행된다.
                        b       not_used                // 사용되지 않는다.
                        b       IRQ                     // IRQ 인터럽트가 발생되면 실행된다.
                        b       FIQ                     // IRQ 인터럽트보다 우선권이 있는 인터럽트
가 발생되면 실행된다.
```

Boot Sequence (III)

- **Start.S (II)**

```
//-------------------------------------------------------------------
//  RESET이 발생하거나 전원이 들어 왔을때 맨 처음 수행되는 루틴이다.
//-------------------------------------------------------------------
reset:
                    //-----------------------------------------
                    // change Supervisor mode & IRQ/FIQ Disable
                    //-----------------------------------------
                    mrs         r0, CPSR
                    bic         r0, r0, #0x1f
                    orr         r0, r0, #(0x13 | 0xc0)
                    msr         CPSR, r0
                    //-----------------------------------------
                    // Change CPU SPEED
                    //-----------------------------------------
                    // CCCR 레지스터를 설정하여 CPU 속도를 맞춘다.
                    ldr         r0, =PXA_REG_CCCR
                    ldr         r1, =CPU_SPEED
                    str         r1, [r0]
                    // 동작속도를 변경한다.
                    mov         r0, #PXA_COP_CCLKCFG_FCS
                    mcr         p14, 0, r0, c6, c0, 0
                    // 터보모드가 있는지 확인한다.
                    and         r1, r1, #(0x7 << 7)     // CCCR_BF_N Mask
                    cmp         r1, #CCCR_BF_N_RUN_X10
                    beq         10f
                    // 터보모드를 활성화 한다.
                    mov         r0, #PXA_COP_CCLKCFG_TURBO
                    mcr         p14, 0, r0, c6, c0, 0
```

Boot Sequence (IV)

- **Start.S (III)**

```
10:
                    //-----------------------------------------
                    // Active I-Chache
                    //-----------------------------------------
                    // I-Cache 비활성화
                    mcr         p15, 0, r1, c7, c5, 0
                    CPWAIT
                    // I-Cache 활성화
                    mrc         p15, 0, r0, c1, c0, 0
                    orr         r0, r0, #0x1000
                    mcr         p15, 0, r0, c1, c0, 0
                    CPWAIT
                    mov         r5, #DEBUG_START
                    bl          led_out

                    //-----------------------------------------
                    //  GPIO 설정
                    //-----------------------------------------
                    bl          gpio_init
```

Boot Sequence (V)

- **gpio.S**

```
/*=====================================================
  화일명 : gpio.S
  설  명 : EZ-X5 의 GPIO를 제어 하는 루틴이다.
=====================================================*/
#include   "ez_x5.h"
.text
//-----------------------------------------------------
//  gpio 초기화
//-----------------------------------------------------
.globl gpio_init
gpio_init:              // GPIO Alternate function
                        ldr         r0, =PXA_REG_GP_BASE

                        ldr         r1, =GAFR0_L_VALUE
                        str         r1, [r0, #PXA_REG_OFFSET_GAFR0_L]
                        ldr         r1, =GAFR0_U_VALUE
                        str         r1, [r0, #PXA_REG_OFFSET_GAFR0_U]
                                        // Repeat for GAFR1, GAFR2
                        ldr         r1, =GPDR0_VALUE
                        str         r1, [r0, #PXA_REG_OFFSET_GPDR0]
                                        // Repeat for GPDR1, GPDR2
                        // PSSR 설정 - GPIO 입력 활성화
                        ldr         r0, =PXA_REG_PSSR
                        ldr         r1, =PSSR_VALUE
                        str         r1, [r0]
                        // debug led off
                        ldr         r0, =PXA_REG_GP_BASE
                        mov         r1, #(_LED_3|_LED_2|_LED_1|_LED_0)
                        str         r1, [r0, #PXA_REG_OFFSET_GPSR0]

                        mov         pc, lr                          // Return!
```

Boot Sequence (VI)

- **Start.S (IV)**

```
//-------------------------------------------
//  SDRAM, Static MEM 설정
//-------------------------------------------
bl          mem_config                  // In memory.S
mov         r5, #DEBUG_READY_MEMTEST
bl          led_out
//-------------------------------------------
//  Memory Test
//-------------------------------------------
bl          mem_test
mov         r5, #DEBUG_MEM_OK
bl          led_out
//-------------------------------------------
//  부트로더를 메모리에 올린다.
//-------------------------------------------
bl          mem_clear
bl          copy_loader_to_ram
```

Boot Sequence (VII)

- **memory.S**

```
//-------------------------------------------------------------------
//  롬영역의 부트 로더를 램에 올린다.
//-------------------------------------------------------------------
.globl copy_loader_to_ram
copy_loader_to_ram:
                    ldr     r0, =EZ_X5_RAM_BOOT          // 메모리 베이스 어드레스
                    ldr     r1, =EZ_X5_RAM_BOOT_END      // 128kbyte
                    mov     r2, #EZ_X5_BASE_ROM          // 롬 베이스 어드레스

10:                 ldr     r3, [r2], #4                 // Get
                    str     r3, [r0], #4                 // Put
                    cmp     r0, r1                       // Complete?
                    bne     10b                          // Loop if not

                    ldr     r0, =EZ_X5_BASE_ROM          // 저장된 내용을 비교한다
                    ldr     r1, =EZ_X5_RAM_BOOT
                    ldr     r4, =EZ_X5_RAM_BOOT_END

20:                 ldr     r2, [r0], #4
                    ldr     r3, [r1], #4
                    cmp     r2, r3
                    bne     comp_error                   //Jump to  Error routine!
                    cmp     r1, r4
                    bne     20b

                    mov     pc, lr                       // Return

comp_error:         mov     r5, #DEBUG_MEM_ERROR
                    bl      led_out
                    b       error_loop
```

Boot Sequence (VIII)

- **Start.S (V)**

```
//-----------------------------------------
//  STACK 포인터를 설정한다.
//-----------------------------------------
ldr         r0, =EZ_X5_RAM_BOOT_END
sub         sp, r0, #0x04
mov         r5, #DEBUG_JUMP_C
bl          led_out
//-----------------------------------------
//  ezboot 임을 메모리에 표시한다.
//-----------------------------------------
mov         r1, #0xA0000000
mov         r2, #0x0000
str         r2, [r1]
//-----------------------------------------
//  main.c 로 이동한다.
//-----------------------------------------
ldr         r0, =EZ_X5_RAM_BOOT
add         r0, r0, #EZ_X5_C_MAIN_OFFSET
mov         pc, r0
// 이곳으로는 진행되지 않는다.
b           error_loop
```

Boot Sequence (IX)

- **Start.S (VI)**

```
//-------------------------------------------------------------------------
//  Reset 이외의 인터럽트 처리
//-------------------------------------------------------------------------
data_abort:
                        mov     r5, #DEBUG_DATA_ABORT
                        bl      led_out
                        b       error_loop

undefined_instruction:
software_interrupt:
prefetch_abort:
not_used:
IRQ:
FIQ:
                        mov     r5, #DEBUG_OTHER_EXCEPT
                        bl      led_out
                        b       error_loop
```


Boot Sequence (X)

- **Main program of EzBoot (main.c)**

```
int             main( void )
{
                // 환경값을 불러온다.
                LoadConfig();
                // 타이머를 초기화 한다.
                TimerInit();
                // GPIO를 초기화 한다.
                GPIOInit();
                // 시리얼을 초기화 한다.
                SerialInit( BAUD_115200 );
                ZModem_Baudrate = 115200;
                // 이지부트 시작 메세지를 보여준다.
                printf( "₩n₩n");
                printf( "WELCOME EZBOOT.X5 V1.6.....for PXA255₩n");
                printf( "Program by You Young-chang, fooji (FALinux Co.,Ltd)₩n");
                printf( "Last Modify 2004.09.06₩n₩n");
                // LED 를 깜박여 정상임을 표시한다.
                LedBlink();
                // 부트 플래쉬 초기화
                BootFlash_Init();       printf("₩n");
                // NAND 플래쉬 초기화
                NandFlash_Init();       printf("₩n");
                // CS8900을 초기화 한다.
                CS8900_Init();          printf("₩n");
```

Boot Sequence (XI)

- **main.c (II)**

```
// eztiny 에서 넘어온것이라면 메모리의 내용을 플래시에 복사한다.
switch ( start_option )
{
// [ezboot]
case 0x0000 :
        // Autoboot ======================
        printf( "Quickly Autoboot [ENTER] / " );
        if( Cfg.BootMenuKey == ' ' )
                printf( "Goto BOOT-MENU press [space bar]");
        else    printf( "Goto BOOT-MENU press [%c]", Cfg.BootMenuKey );

        if ( getc_timed( Cfg.BootMenuKey, Cfg.AutoBootWaitTime*1000 ) )
        {
                printf( "₩n");
                CopyImage();
                GoKernel( 1, NULL );
        }
        break;
// [eztiny]
case 0x0001 : SelfCopy_BootLoader();
        break;
}

printf( "₩n₩n");
```

Boot Sequence (XII)

- **main.c (III)**

```
        // Boot propmt ======================
        // 명령을 대기한다.
        while (1) {
                printf( "EZBOOT>");
                memset( ReadBuffer, 0 , sizeof( ReadBuffer ) );

                // 시리얼에서 명령어가 입력될때까지 대기한다.
                gets_his( ReadBuffer );
                printf( "₩n");
                argc = parse_args( ReadBuffer, argv );

                if (argc) {
                        UpperStr( argv[0] );
                        cmdlp = 0;
                        while( Cmds[cmdlp].CmdStr ) {
                                if( strcmp( argv[0], Cmds[cmdlp].CmdStr ) == 0 ) {
                                        Cmds[cmdlp].func( argc, argv );
                                        printf( "₩n");
                                        break;
                                }
                                cmdlp++;
                        }
                }
        }
}
```


References

- Bootloader, EzBoot
  - EZ-X5 User's Manual, http://www.ffalinux.com
- Boot seqence
  - EX-X5 CD, sw/ezboot.x5.v16/v16.org.tar.gz
- *Step by step one goes a long way.*

국화 옆에서

서정주

한 송이의 국화꽃을 피우기 위해
봄부터 소쩍새는
그렇게 울었나 보다.

한 송이의 국화꽃을 피우기 위해
천둥은 먹구름 속에서
또 그렇게 울었나 보다.

그립고 아쉬움에 가슴 조이던
머언 먼 젊음의 뒤안길에서
인제는 돌아와 거울 앞에 선
내 누님같이 생긴 꽃이여.

노오란 네 꽃잎이 피려고
간밤엔 무서리가 저리 내리고
내게는 잠도 오지 않았나 보다.