SSD Firmware Implementation Project
- Lab. #7-

Sang-Phil Lim (lsfeel0204@gmail.com)
SKKU VLDB Lab.

2011-05-26

Lab. Time Schedule

Lab.	Title
#1	FTL Simulator Development Guide
#2	FTL Simulation Guide
#3	Project 1 Presentation
#4	Jasmine OpenSSD platform tutorial #1
#5	Jasmine OpenSSD platform tutorial #2
#6	FTL Porting Guide
#7	**Firmware Debugging Guide**
#8	SSD Performance Evaluation Guide
#9	Project 2 Presentation

Firmware Debugging Guide

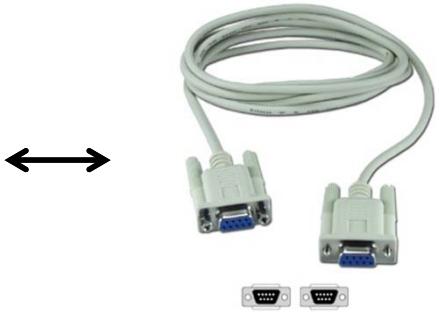
- Ways to debug your code
  1. **Serial communication (UART interface)**
     - Print debug message
     - ASSERT function
  2. ICE + RVD (JTAG interface)
     - Line-by-line debugging
  3. On-board indicator (LED on Position D4)

Introduction of UART Debugging

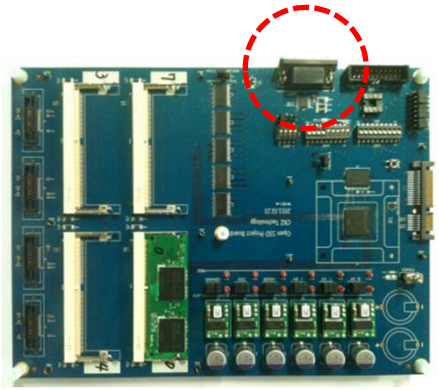
- UART(Universal Asynchronous Receiver/Transmitter)
  - Memory/Register dump message
  - Measure I/O response time using ‘Timer function’ (`./target_spw/misc.c`)

UART Debugging: Hardware Setting

Serial port



RS232 Serial Cable



Jasmine UART port

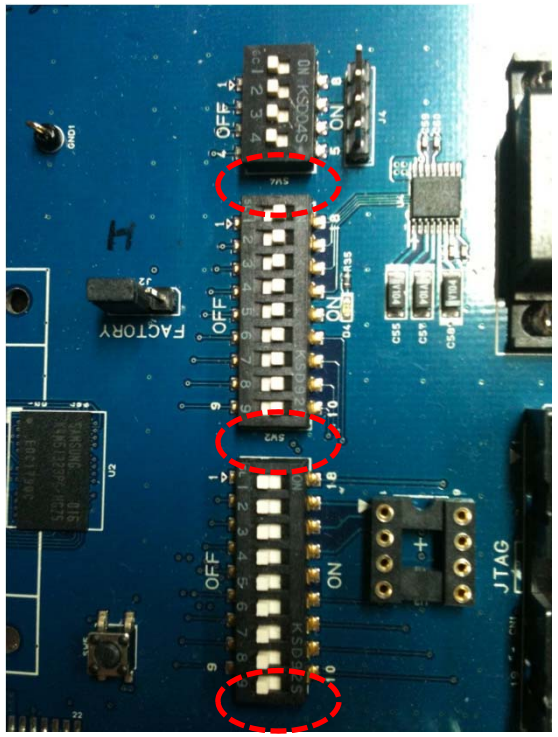
OR

USB to RS232 Cable

UART Debugging: Hardware Setting

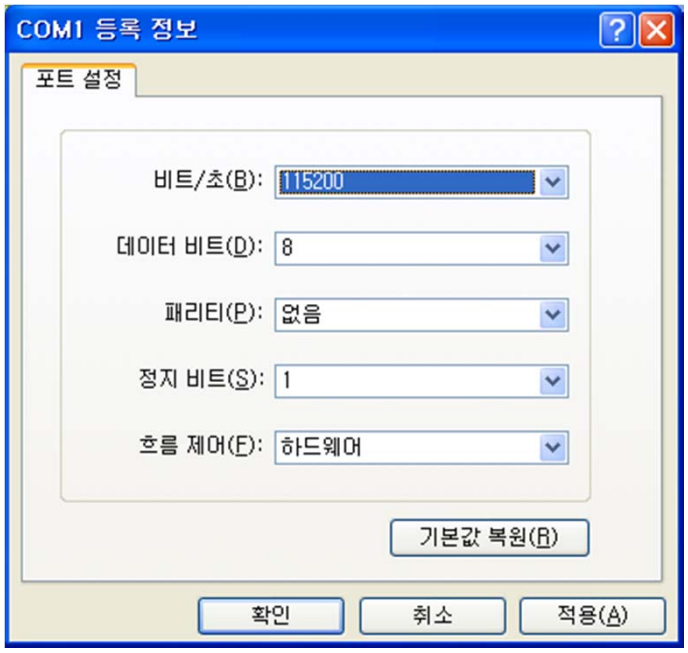
- Setting On-board Switch on Jasmine board

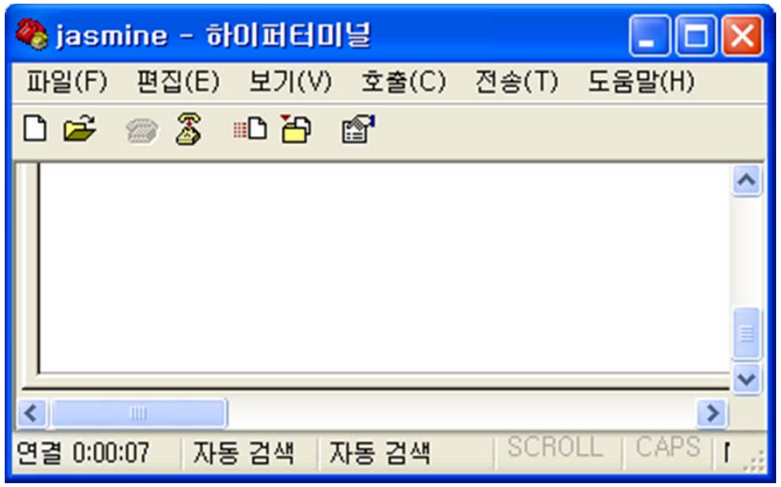
	Part	No.	Status
UART	SW2	1,2,3,4	ON
	SW3	1,2,3,4	OFF
	SW4	1,2,3,4	OFF



UART Debugging: Software Setting

- Terminal program configuration
  - BAUD_115200/8/N/1/YES



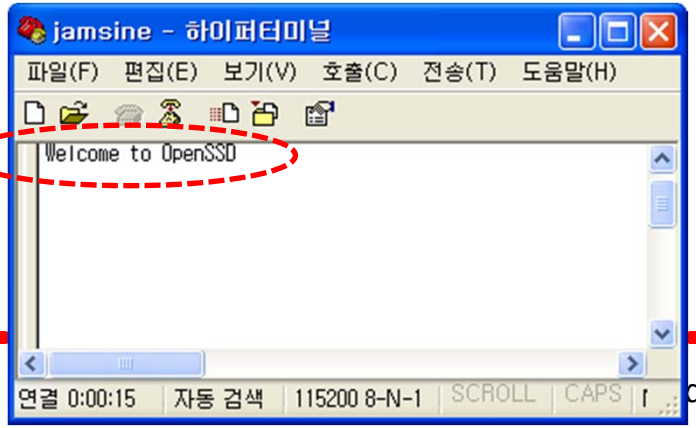


UART Debugging: Software Setting

- Enable UART debug option & re-build Jasmine firmware
  - `./include/jasmine.h`

```
#define OPTION_UART_DEBUG    1 // 1 = enable UART message output
```

- Install new firmware image to Jasmine board
- Boot the Jasmine board on 'normal mode', then you can observe the following message through terminal



UART Debugging

- Available APIs
  - **./target_spw/misc.h & misc.c**

```
// timer start -> stop -> print latency(us) to UART
void ptimer_start(void);
void ptimer_stop_and_uart_print(void);

// print unsigned integer value to UART
void uart_print_32(UINT32); // by sprintf function
```

c.f.) ASSERT Function

- `./include/jasmine.h`

```
#define OPTION_ENABLE_ASSERT  0   // 1 = enable
```

- `./target_spw/target.h`

```
#if OPTION_ENABLE_ASSERT
    #define ASSERT(X)                       \
    {                                       \
        if (!(X))                           \
        {                                   \
            // add your debug message       \
            while (1);                      \
        }                                   \
    }
#else
    #define ASSERT(X)
#endif
```

UART Debugging Example

① Print debug message (**./ftl_greedy/ftl.c**)

```
void ftl_test(void)
{
    ...
    // STEP 2 - read and verify
    for (j = 0; j < num_sectors; j++)
    {
        UINT32 sample = read_dram_32(rd_buf_addr);

        if (sample != data)
        {
            uart_print("ftl test fail...");
            led_blink();
        }
        ...
    }
}
```

UART Debugging Example

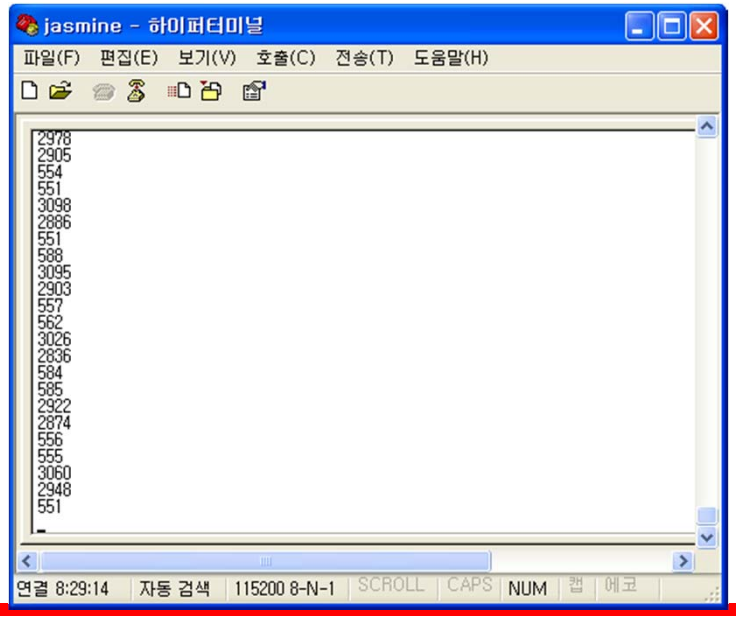
② Register dump (**./ftl_greedy/ftl.c**)

```
// BSP interrupt service routine
void ftl_isr(void)
{
    UINT32 bank, bsp_intr_flag;

    uart_print("BSP interrupt occured...");
    // interrupt pending clear (ICU)
    SETREG(APB_INT_STS, INTR_FLASH);

    for (bank = 0; bank < NUM_BANKS; bank++) {
        // get interrupt flag from BSP
        bsp_intr_flag = BSP_INTR(bank);

        if (bsp_intr_flag == 0) continue;

        // dump current bank's BSP registers (BSP contains last accepted FC)
        uart_print_32(bank);
        uart_print("----BSP register dump---");
        uart_print_32(BSP_CMD(bank));
        uart_print_32(BSP_OPTION(bank));
        uart_print_32(BSP_DMA_ADDR(bank));
        ...
    }
}
```

UART Debugging Example

③ Measuring IO response time

```
ptimer_start();
ftl_write(lba, num_sectors);
ptimer_stop_and_uart_print();
```



c.f.) Debugging with LED

- Control LED on position D4
  - LED On/Off/Blink
  - **`./target_spw/misc.h & misc.c`**

```
void led(BOOL32 on);
void led_blink(void);
```

Supplement of Previous Lab.

Page Copy-back Operation

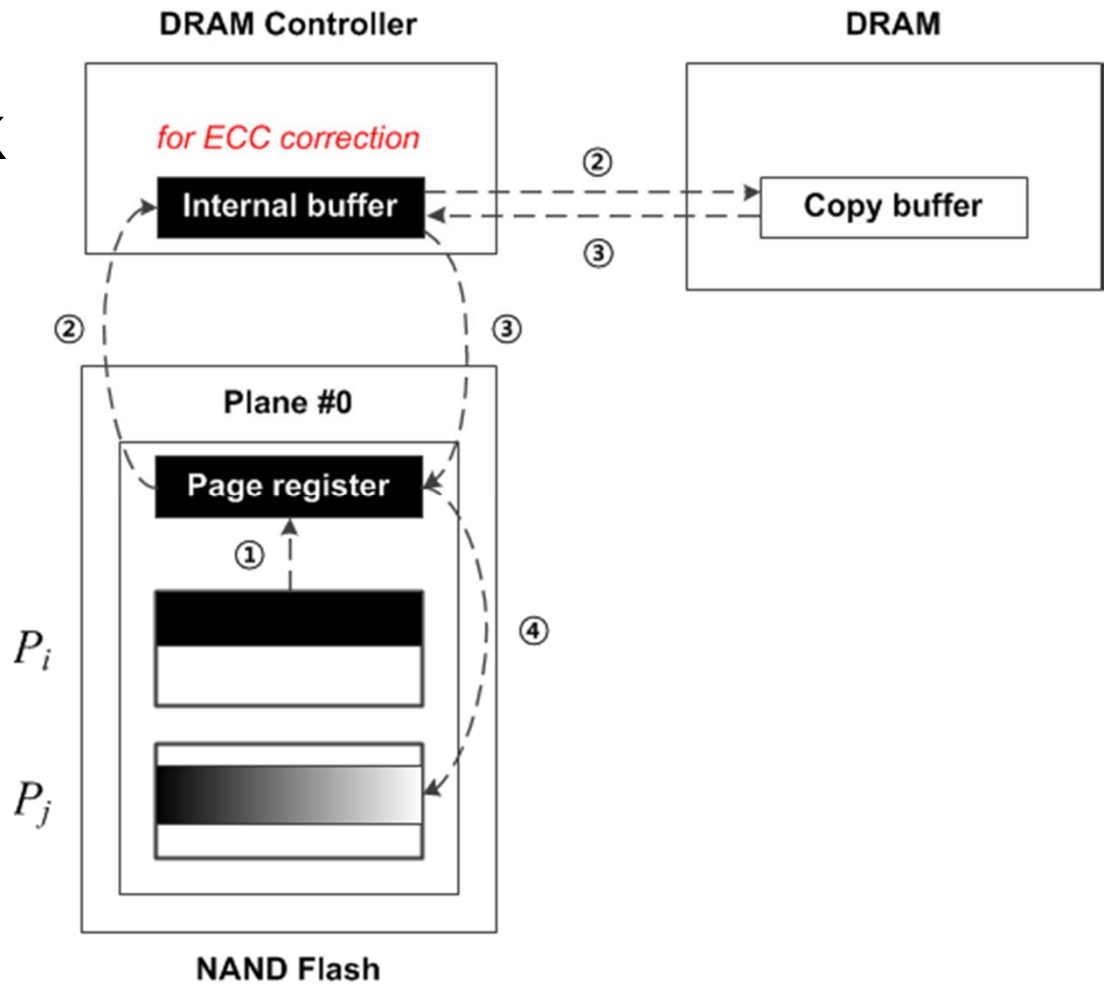
```
void nand_page_copyback(UINT32 const bank, UINT32 const src_vblock, UINT32 const src_page,
UINT32 const dst_vblock, UINT32 const dst_page)
{
    #if NAND_SPEC_DIE == NAND_SPEC_DIE_2
    {
        #if NAND_SPEC_PLANE == NAND_SPEC_PLANE_3 && OPTION_2_PLANE == FALSE
        {
            if (src_row / (PAGES_PER_BANK / 4) != dst_row / (PAGES_PER_BANK / 4))
                do_copyback = FALSE;
        }
        #else
        {
            // check same plane
            if (src_row / (PAGES_PER_BANK / 2) != dst_row / (PAGES_PER_BANK / 2))
                do_copyback = FALSE;
        }
        #endif
    }
    ...
    if (do_copyback)
    {
        SETREG(FCP_CMD, FC_COPYBACK);
        SETREG(FCP_OPTION, FO_P | FO_E | FO_B_W_DRDY);
        ...
    }
    ...
```

Page Copy-back Operation

```
    ...
    // no internal copyback
    else
    {
        SETREG(FCP_CMD, FC_COL_ROW_READ_OUT);
        SETREG(FCP_OPTION, FO_P | FO_E);
        SETREG(FCP_DMA_ADDR, COPY_BUF(bank));
        SETREG(FCP_DMA_CNT, BYTES_PER_PAGE);
        SETREG(FCP_ROW_L(bank), src_row);
        SETREG(FCP_ROW_H(bank), src_row);
        SETREG(FCP_COL, 0);
        SETREG(FCP_BANK, REAL_BANK(bank));
        FLASH_POLL_WR;
        FLASH_ISSUE;

        SETREG(FCP_CMD, FC_COL_ROW_IN_PROG);
        SETREG(FCP_OPTION, FO_P | FO_E | FO_B_W_DRDY);
        SETREG(FCP_ROW_L(bank), dst_row);
        SETREG(FCP_ROW_H(bank), dst_row);
        SETREG(FCP_BANK, REAL_BANK(bank));
        FLASH_POLL_WR;
        FLASH_ISSUE;
    }
}// end of nand_page_copyback function
```

Page Copy-back Internal

- FC_CPBACK
- FC_MODIFY_CPBACK
- Copy page P_i to P_j



Any Questions?