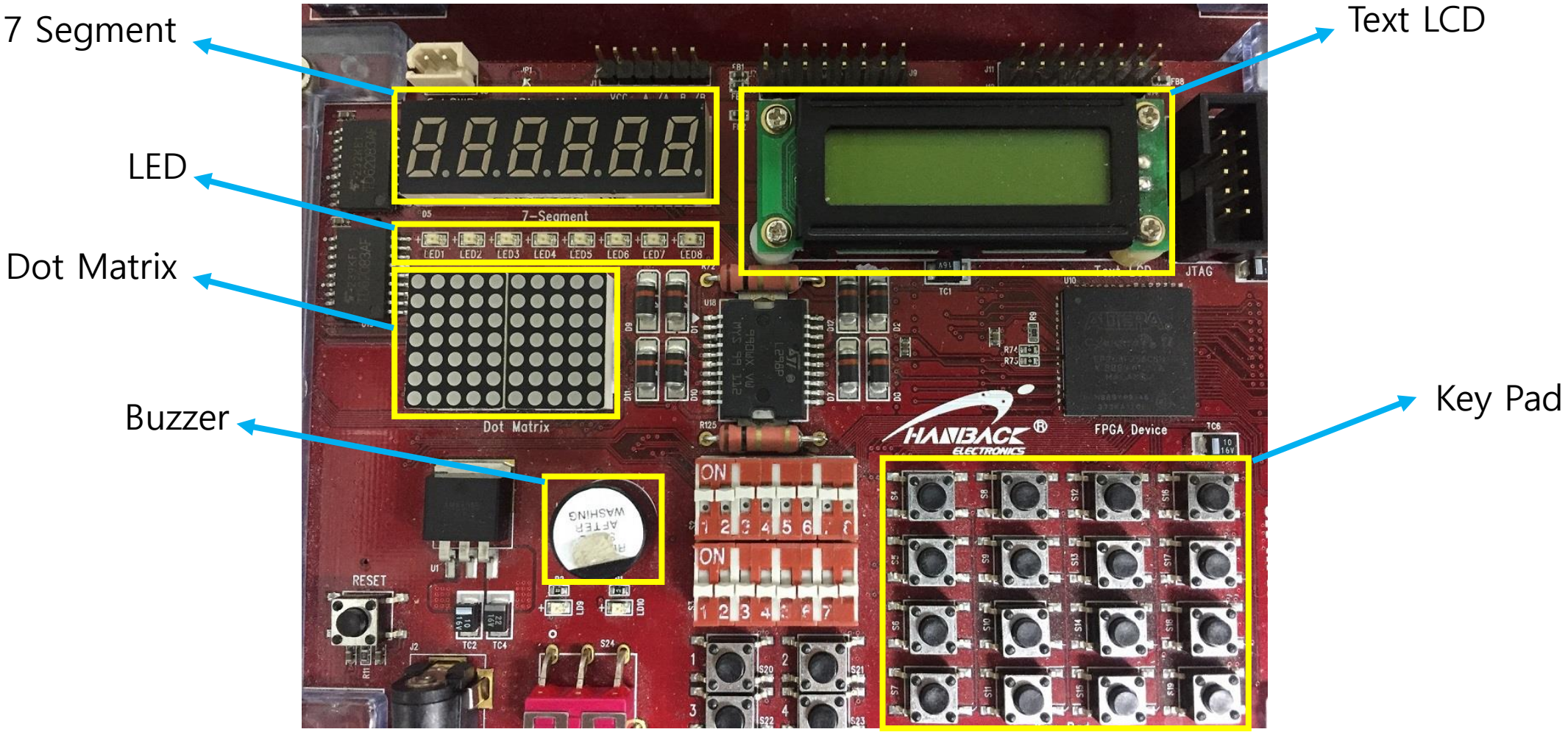
modified date : 5.31.2018

teacher assistant : SungHoon Im

SV210 장치 제어

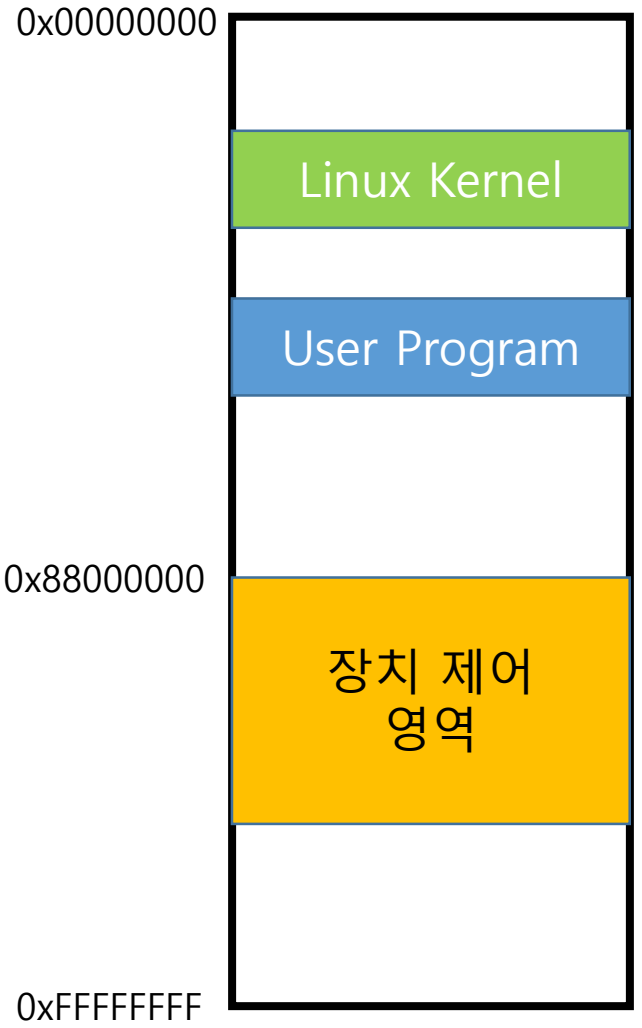
Index

1. 장치 소개


7 Segment
LED
Dot Matrix
Buzzer
Text LCD
Key Pad
7-Segment
LED1 LED2 LED3 LED4 LED5 LED6 LED7 LED8
Dot Matrix
FPGA Device
JTAG
RESET
ON

2. 장치 제어 기본 개념

- 장치 제어는 기본적으로 특정 메모리 영역에 제어할 값을 Write 하여 제어
- SV210 장비는 0x88000000 번지부터 시작으로 장치 제어 영역이 존재
- 예) LED 제어 영역은 0x88000000으로부터 0x20 만큼 떨어진 위치에 존재



물리 메모리 예시

2. 장치 제어 기본 개념

자세한 것은 Linux의
open, mmap 함수를 찾아볼 것

open 함수는 linux programming 기본 개념

mmap 함수는 유저 영역, 커널 영역의 이해가 필요

```
open("/dev/mem", O_RDWR | O_SYNC)
```

- 장치를 제어하기 위해서는 가장 먼저 물리 메모리를 Open 해야함
- 물리 메모리 디바이스의 위치는 /dev/mem 에 위치
- 메모리를 Open할 때에 옵션은 Read/Write(O_RDWR)이 가능해야함
- 또한 다른 프로세서에서 변경된 값을 바로 확인 가능하도록 동기화(O_SYNC) 함
- Open 함수는 File Descriptor를 반환, 실패시 음수 반환

```
mmap(NULL, 4096, PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0x88000000)
```

- 물리메모리에 접근하도록 하기 위해서는 mmap 함수를 통해 접근 가능한 주소를 받아야 함
- 첫 번째 인자는 시작 주소로 보통 0 사용
- 두 번째 인자는 사용할 메모리의 크기로 , 여기서는 4KB 사용
- 세 번재 인자는 메모리의 접근 권한의 옵션으로 여기서는 읽기/쓰기가 가능하도록 함
- 네 번째 인자는 해당 영역의 특성으로 다른 프로세서에서 해당 영역을 공유할 수 있도록 함
- 다섯 번째 인자는 File Descriptor
- 마지막 인자는 시작주소로부터 떨어진 offse을 의미

3. LED 제어

- LED는 8개로 구성되어 있으며 On/Off 상태를 1bit로 표현 가능.
- 8bit를 통해 8개의 LED를 제어할 수 있음
- 예) 0xFF, 모든 LED가 ON 상태
- 예) 0x80, 왼쪽 끝의 LED 하나만 ON 상태
- 예) 0x01, 오른쪽 끝의 LED 하나만 ON 상태

3. LED 제어

- 먼저 메모리 디바이스를 Open, 음수 반환시 오류
- mmap을 통해 원하는 주소의 접근 권한을 얻음, 음수 반환시 오류
- 제어영역 주소로부터 LED 제어 영역(0x20) 계산
- LED 전체 1초마다 ON/OFF 반복 제어 코드 동작

```
fd = open("/dev/mem", O_RDWR | O_SYNC);
if(fd < 0){
    exit(1);
}

addr_fpga = (unsigned short *)mmap(NULL,
                                   4096,
                                   PROT_READ | PROT_WRITE,
                                   MAP_SHARED,
                                   fd,
                                   0x88000000);

if(*addr_led == (unsigned short)-1){
    close(fd);
    printf("mmap error₩n");
    exit(1);
}

addr_led = addr_fpga + 0x20 / sizeof(unsigned short);

while(1){
    *addr_led = 0xFF; // LED All On
    sleep(1); // sleep 1 second
    *addr_led = 0x00; // LED All Off
    sleep(1); // sleep 1 second
}
```

4. 7-Segment 제어

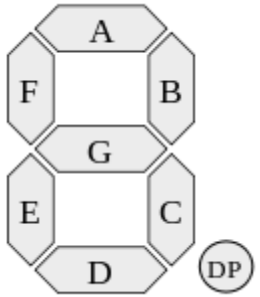


그림 1

- 7-Segment는 간단하게 설명하면 8개의 LED가 그림 1과 같이 배열되어 있는 것
- SV210은 6개의 7-Segment가 존재
- 하나의 7-Segment는 8bit로 ON/OFF 제어 가능
- 비트의 순서는 abcd efgh 순서로 h는 그림 1에서 DP와 같음
- 예) 0x81, 그림 1에서 A와 DP를 ON
- 예) 0xfc, 그림 1에서 A, B, C, D, E, F를 ON(숫자 0과 유사)

4. 7-Segment 제어

- 앞서 메모리 디바이스를 열고 mmap를 하였다고 가정
- 7-Segment는 총 6개로 각각 1, 2, 4, 8, 16, 32로 제어
- 7-Segment의 표현으로 0~9까지 ON/OFF 상태 값
- 장치 제어 주소로부터 7-Segment 제어 주소 계산
- 몇 번째 7-Segment를 제어할지에 대한 주소(addr_grid)
- 선택된 세그먼트의 ON/OFF 상태의 값을 저장할 주소(addr_data)
- 6개의 7-Segment에 순서대로 값 출력

```
....

unsigned short led_pos[6]={
                            0x01, 0x02, 0x04,
                            0x08, 0x10, 0x20
                          };
unsigned short digit[10]={
                        //  0     1     2     3     4
                            0xfc, 0x60, 0xda, 0xf2, 0x66,
                        //  5     6     7     8     9
                            0xb6, 0xbe, 0xe4, 0xf3, 0xf6
                          };

addr_grid = addr_fpga + 0x30 / sizeof(unsigned short);
addr_data = addr_fpga + 0x32 / sizeof(unsigned short);

while(1){
    for(i = 0; i < 6; ++i){
        *addr_grid = led_pos[i]; // set segment position
        *addr_data = digit[i]; // set segment data
        usleep(10);
    }
}
```

5. Key pad 제어

- 첫 번째 Row를 선택 후
- Col의 키 4개 중 어디가 눌렸는지 확인
- 두 번째 Row를 선택 후
- Col의 키 4개 중 어디가 눌렸는지 확인

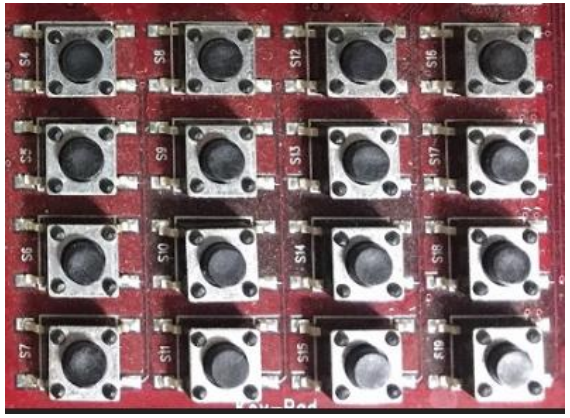
```
...

*keypad_row_addr = 0x01;
usleep(1000);
value = (*keypad_col_addr & 0x0f);
*keypad_row_addr = 0x00;
switch(value){
    case 0x01 : value = 0x01; break;
    case 0x02 : value = 0x02; break;
    case 0x04 : value = 0x03; break;
    case 0x08 : value = 0x04; break;
}
if(value != 0x00) goto stop_poll;

*keypad_row_addr = 0x02;
usleep(1000);
value = (*keypad_col_addr & 0x0f);
*keypad_row_addr = 0x00;
switch(value){
    case 0x01 : value = 0x05; break;
    case 0x02 : value = 0x06; break;
    case 0x04 : value = 0x07; break;
    case 0x08 : value = 0x08; break;
}
if(value != 0x00) goto stop_poll;

...
```

0x01

0x02

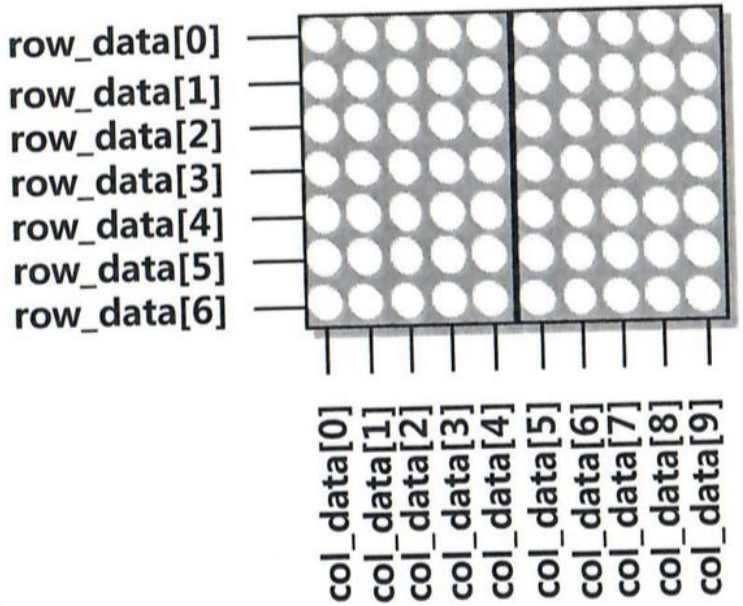
0x04

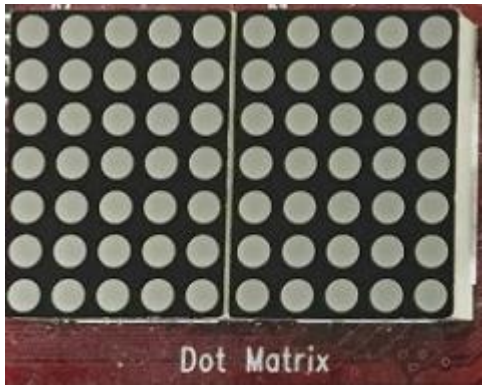
0x08



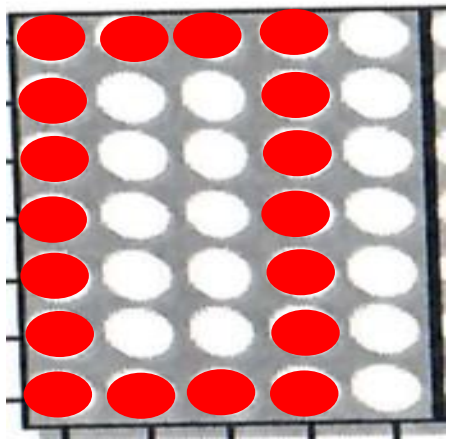
6. Dot matrix 제어

- 도트 매트릭스는 col_data 7bit를 통해 ON/OFF 제어
- 하나의 도트 매트릭스는 7x5 로 구성됨

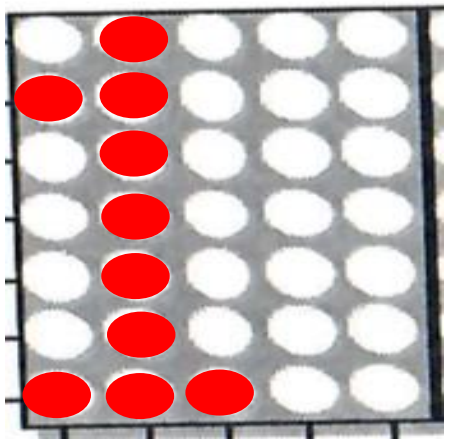




0x7f 0x41 0x41 0x7f



0x42 0x7f 0x40 0x00



6. Dot matrix 제어

- 0~9까지 표현하는 On/Off 상태를 배열에 정의
- 첫 번째 Dot Matrix에 숫자 1 출력
- 두 번째 Dot Matrix에 숫자 5 출력

```
unsigned short font_num[40] = { 0x7f, 0x41, 0x41, 0x7f, // 0
                                0x40, 0x7f, 0x42, 0x00, // 1
                                0x4f, 0x49, 0x49, 0x79, // 2
                                0x7f, 0x49, 0x49, 0x49, // 3
                                0x10, 0x7f, 0x10, 0x1f, // 4
                                0x79, 0x49, 0x49, 0x4f, // 5
                                0x79, 0x49, 0x49, 0x7f, // 6
                                0x7f, 0x01, 0x01, 0x01, // 7
                                0x7f, 0x49, 0x49, 0x7f, // 8
                                0x7f, 0x49, 0x49, 0x4f, // 9
                                };

while(1){
    const int dot1_num = 1;
    const int dot2_num = 5;
    int j;

    for(j = 0; j < 4; j++){
        *dot_row_addr = 0x008 >> j;
        *dot_col_addr = font_num[dot1_num * 4 + j];
        usleep(500);
    }
    for(j = 0; j < 4; j++){
        *dot_row_addr = 0x100 >> j;
        *dot_col_addr = font_num[dot2_num * 4 + j];
        usleep(500);
    }
}
```

7. Text LCD 제어

- Text LCD는 그림 1과 같이 2X16 배열 형태로 글자를 출력할 수 있음
- 문자를 출력하기 위한 제어 명령이 별도로 존재
- SV210은 그림2에서 RS, RW, E, D7~D0 로 총 11개의 라인을 사용함
- 예) 0x0100, 그림 3에서 E 비트를 ON
- 예) 0x0200, 그림 3에서 RW 비트를 ON
- 예) 0x0400, 그림 3에서 RS 비트를 ON

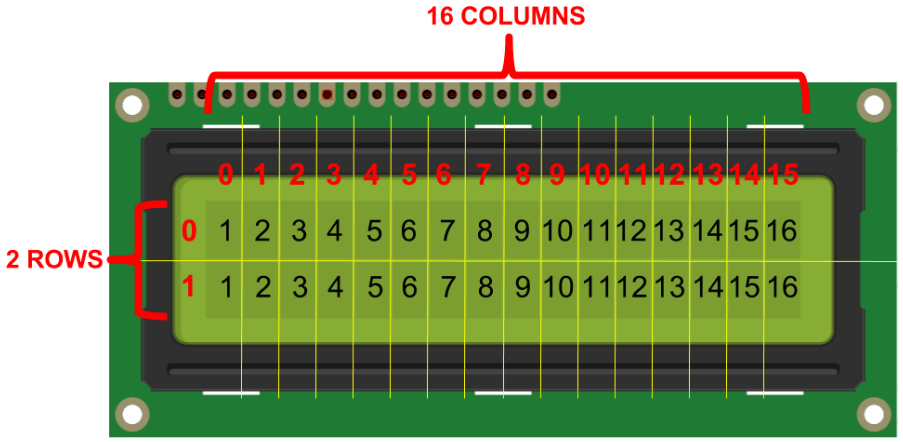


그림 1

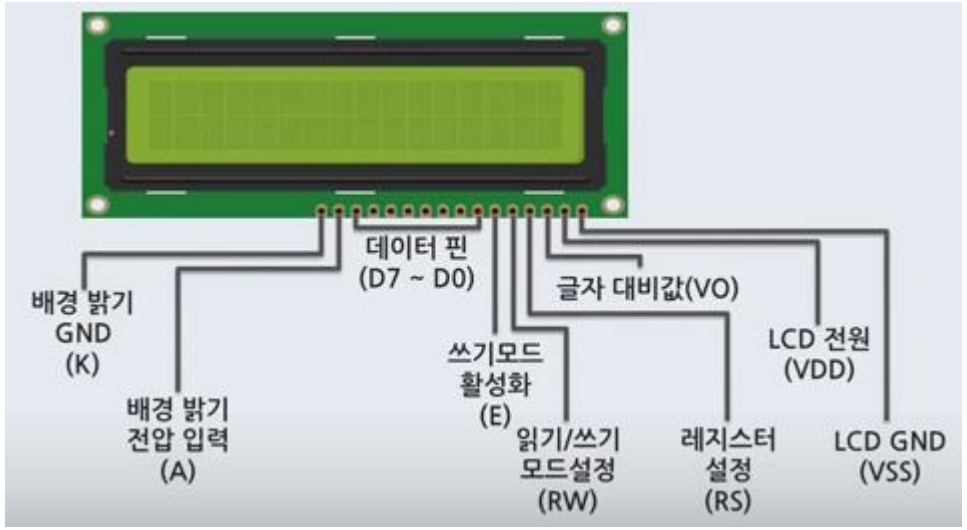


그림 2

RS	RW	E	D7	D6	D5	D4	D3	D2	D1	D0

그림 3

7. Text LCD 제어

기 능	제어 신호		제어 명령 (D0 ~ D7)								실행 시간
	RS	R/W	7	6	5	4	3	2	1	0	
Clear Display	0	0	0	0	0	0	0	0	0	1	1.64ms
Return Home	0	0	0	0	0	0	0	0	1	0	40us
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	40us
Display on/off control	0	0	0	0	0	0	1	D	C	S	40us
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	0	0	40us
Function Set	0	0	0	0	1	D/L	N	F	0	0	40us
Set CG RAM Address	0	0	0	1	CG RAM Address						40us
Set DD RAM Address	0	0	1	DD RAM Address							40us
Read Busy Flag and Address	0	1	BF	Address Counter							40us
Data Write to CG RAM or DD RAM	1	0	Write Address								40us
Data Read to CG RAM or DD RAM	1	1	Read Address								40us

```
void set_command(unsigned short command){
    command &= 0x00FF;

    *ptextlcd = command | 0x0100;
    usleep(1000);
    *ptextlcd = command | 0x0000;
    usleep(1000);
}
```

제어명령을 실행하기 위해서는 E 비트를 ON 해야함

7. Text LCD 제어

- Clear Display
- 모든 디스플레이 상태를 제거하고 커서를 Home 위치로 옮긴다.

```
void clear_display(){
    unsigned short command = 0x01;
    set_command(command);
}
```

기 능	제어 신호		제어 명령 (D0 ~ D7)								실행 시간
	RS	R/W	7	6	5	4	3	2	1	0	
Clear Display	0	0	0	0	0	0	0	0	0	1	1.64ms
Return Home	0	0	0	0	0	0	0	0	1	0	40us
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	40us
Display on/off control	0	0	0	0	0	0	1	D	C	S	40us
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	0	0	40us
Function Set	0	0	0	0	1	D/L	N	F	0	0	40us
Set CG RAM Address	0	0	0	1	CG RAM Address						40us
Set DD RAM Address	0	0	1	DD RAM Address							40us
Read Busy Flag and Address	0	1	BF	Address Counter							40us
Data Write to CG RAM or DD RAM	1	0	Write Address								40us
Data Read to CG RAM or DD RAM	1	1	Read Address								40us

7. Text LCD 제어

- Return Display
- 디스플레이 상태는 그대로 두고 커서만 Home으로 옮긴다.

```
void return_home(){
    unsigned short command = 0x02;
    set_command(command);
}
```

기 능	제어 신호		제어 명령 (D0 ~ D7)								실행 시간
	RS	R/W	7	6	5	4	3	2	1	0	
Clear Display	0	0	0	0	0	0	0	0	0	1	1.64ms
Return Home	0	0	0	0	0	0	0	0	1	0	40us
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	40us
Display on/off control	0	0	0	0	0	0	1	D	C	S	40us
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	0	0	40us
Function Set	0	0	0	0	1	D/L	N	F	0	0	40us
Set CG RAM Address	0	0	0	1	CG RAM Address						40us
Set DD RAM Address	0	0	1	DD RAM Address							40us
Read Busy Flag and Address	0	1	BF	Address Counter							40us
Data Write to CG RAM or DD RAM	1	0	Write Address								40us
Data Read to CG RAM or DD RAM	1	1	Read Address								40us

7. Text LCD 제어

- Entry Mode Set
- 데이터를 Read하거나 Write 할 때 커서의 위치를 증가시킬 것인가(I/D=1) 감소시킬 것인가(I/D=0) 결정
- 또 이때 화면을 이동할 것인지(S=1), 아닌지(S=0)를 결정

```
void entry_mode_set(int increase, int nshift){
    unsigned short command = 0x04;
    command = increase ? (command | 0x2) : command;
    command = nshift ? (command | 0x1) : command;
    set_command(command);
}
```

기 능	제어 신호		제어 명령 (D0 ~ D7)								실행 시간
	RS	R/W	7	6	5	4	3	2	1	0	
Clear Display	0	0	0	0	0	0	0	0	0	1	1.64ms
Return Home	0	0	0	0	0	0	0	0	1	0	40us
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	40us
Display on/off control	0	0	0	0	0	0	1	D	C	S	40us
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	0	0	40us
Function Set	0	0	0	0	1	D/L	N	F	0	0	40us
Set CG RAM Address	0	0	0	1	CG RAM Address						40us
Set DD RAM Address	0	0	1	DD RAM Address							40us
Read Busy Flag and Address	0	1	BF	Address Counter							40us
Data Write to CG RAM or DD RAM	1	0	Write Address								40us
Data Read to CG RAM or DD RAM	1	1	Read Address								40us

7. Text LCD 제어

- Display On/Off Control
- 화면 표시를 On/Off 하거나(D), 커서를 On/Off 하거나(C), 커서를 깜빡이게 할 것인지(S) 결정

```
void display_control(int display_enable, int cursor_enable, int nblink){
    unsigned short command = 0x08;
    command = display_enable ? (command | 0x04) : command;
    command = cursor_enable ? (command | 0x02) : command;
    command = nblink ? (command | 0x01) : command;
    set_command(command);
}
```

기 능	제어 신호		제어 명령 (D0 ~ D7)								실행 시간
	RS	R/W	7	6	5	4	3	2	1	0	
Clear Display	0	0	0	0	0	0	0	0	0	1	1.64ms
Return Home	0	0	0	0	0	0	0	0	1	0	40us
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	40us
Display on/off control	0	0	0	0	0	0	1	D	C	S	40us
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	0	0	40us
Function Set	0	0	0	0	1	D/L	N	F	0	0	40us
Set CG RAM Address	0	0	0	1	CG RAM Address						40us
Set DD RAM Address	0	0	1	DD RAM Address							40us
Read Busy Flag and Address	0	1	BF	Address Counter							40us
Data Write to CG RAM or DD RAM	1	0	Write Address								40us
Data Read to CG RAM or DD RAM	1	1	Read Address								40us

7. Text LCD 제어

- Cursor or Display Shift
- 화면(S/C=1) 또는 커서(S/C=0)를 오른쪽(R/L=1) 또는 왼쪽(R/L=0)으로 이동한다.

```
void cursor_shift(int set_screen, int set_right_shift){
    unsigned short command = 0x10;
    command = set_screen ? (command | 0x08) : command;
    command = set_right_shift ? (command | 0x04) : command;
    set_command(command);
}
```

기 능	제어 신호		제어 명령 (D0 ~ D7)								실행 시간
	RS	R/W	7	6	5	4	3	2	1	0	
Clear Display	0	0	0	0	0	0	0	0	0	1	1.64ms
Return Home	0	0	0	0	0	0	0	0	1	0	40us
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	40us
Display on/off control	0	0	0	0	0	0	1	D	C	S	40us
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	0	0	40us
Function Set	0	0	0	0	1	D/L	N	F	0	0	40us
Set CG RAM Address	0	0	0	1	CG RAM Address						40us
Set DD RAM Address	0	0	1	DD RAM Address							40us
Read Busy Flag and Address	0	1	BF	Address Counter							40us
Data Write to CG RAM or DD RAM	1	0	Write Address								40us
Data Read to CG RAM or DD RAM	1	1	Read Address								40us

7. Text LCD 제어

- Function Set
- 데이터의 길이가 8Bit(DL=1)인지 4Bit(DL=0) 인지 결정
- 행의 길이가 1행(N=0) 인지, 2행(N=1) 인지 결정
- 문자 폰트가 5x7 도트(F=0) 인지, 5x10 도트(F=1) 인지 결정
- Text LCD의 전원이 들어오면 이 명령을 실행해줘야 함
- 이 명령이 실행 완료되기 위해서는 50ms 소요

```
void init_textlcd(){
    function_set(2, 0);
    display_control(1, 0, 0);
    clear_display();
    entry_mode_set(1, 0);
    return_home();
}
```

SV210의 Text LCD 초기화 예시

```
void function_set(int rows, int nfonts){
    unsigned short command = 0x30;
    if(rows == 2){
        command |= 0x08;
    }
    else if(rows == 1){
        command &= 0xf7;
    }
    else{
        return -1;
    }
    command = nfonts ? (command | 0x04) : command;
    set_command(command);
}
```

기 능	제어 신호		제어 명령 (D0 ~ D7)								실행 시간
	RS	R/W	7	6	5	4	3	2	1	0	
Clear Display	0	0	0	0	0	0	0	0	0	1	1.64ms
Return Home	0	0	0	0	0	0	0	0	1	0	40us
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	40us
Display on/off control	0	0	0	0	0	0	1	D	C	S	40us
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	0	0	40us
Function Set	0	0	0	0	1	D/L	N	F	0	0	40us
Set CG RAM Address	0	0	0	1	CG RAM Address						40us
Set DD RAM Address	0	0	1	DD RAM Address							40us
Read Busy Flag and Address	0	1	BF	Address Counter							40us
Data Write to CG RAM or DD RAM	1	0	Write Address								40us
Data Read to CG RAM or DD RAM	1	1	Read Address								40us

7. Text LCD 제어

- Data Write to CG RAM or DD RAM
- 원하는 문자를 출력하는 기능

```
void write_byte(char ch){
    unsigned short data;
    data = ch & 0x00FF;

    *ptextlcd = data | 0x0500;
    usleep(1000);
    *ptextlcd = data | 0x0400;
    usleep(1000);
}
```

기 능	제어 신호		제어 명령 (D0 ~ D7)								실행 시간
	RS	R/W	7	6	5	4	3	2	1	0	
Clear Display	0	0	0	0	0	0	0	0	0	1	1.64ms
Return Home	0	0	0	0	0	0	0	0	1	0	40us
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	40us
Display on/off control	0	0	0	0	0	0	1	D	C	S	40us
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	0	0	40us
Function Set	0	0	0	0	1	D/L	N	F	0	0	40us
Set CG RAM Address	0	0	0	1	CG RAM Address						40us
Set DD RAM Address	0	0	1	DD RAM Address							40us
Read Busy Flag and Address	0	1	BF	Address Counter							40us
Data Write to CG RAM or DD RAM	1	0	Write Address								40us
Data Read to CG RAM or DD RAM	1	1	Read Address								40us

7. Text LCD 제어

- Set DD RAM Address
- 문자를 출력할 DDRAM의 주소를 지정
- 0x00은 Row1, 0x40은 Row2

```
int set_ddram_address(int pos){
    unsigned short command = 0x80;
    command += pos;
    set_command(command);
    return 1;
}
```

기 능	제어 신호		제어 명령 (D0 ~ D7)								실행 시간
	RS	R/W	7	6	5	4	3	2	1	0	
Clear Display	0	0	0	0	0	0	0	0	0	1	1.64ms
Return Home	0	0	0	0	0	0	0	0	1	0	40us
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	40us
Display on/off control	0	0	0	0	0	0	1	D	C	S	40us
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	0	0	40us
Function Set	0	0	0	0	1	D/L	N	F	0	0	40us
Set CG RAM Address	0	0	0	1	CG RAM Address						40us
Set DD RAM Address	0	0	1	DD RAM Address							40us
Read Busy Flag and Address	0	1	BF	Address Counter							40us
Data Write to CG RAM or DD RAM	1	0	Write Address								40us
Data Read to CG RAM or DD RAM	1	1	Read Address								40us