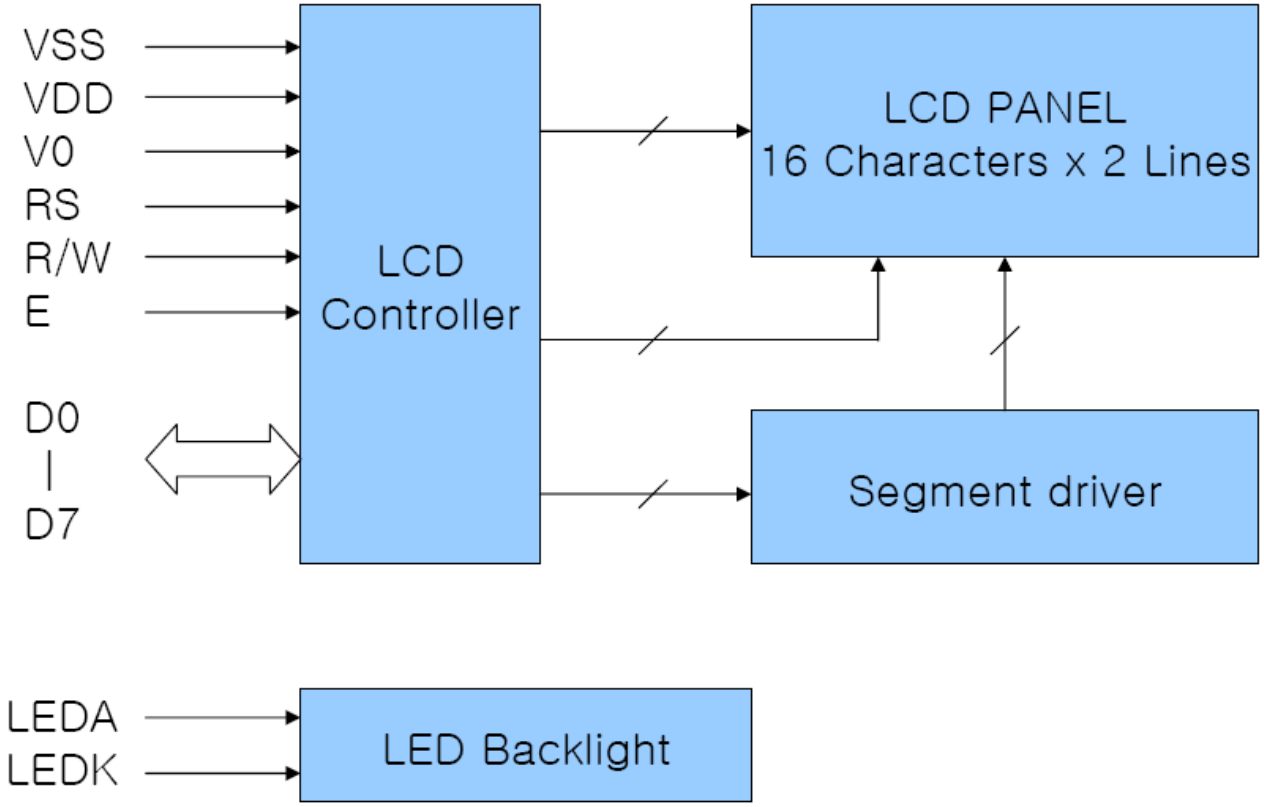
Lab4

Character LCD

(Text LCD)

Character LCD Block Diagram

LCD Controller, LCD Panel, Segment Driver, Backlight



Registers

- IR (Instruction Register)
- DR (Data Register)

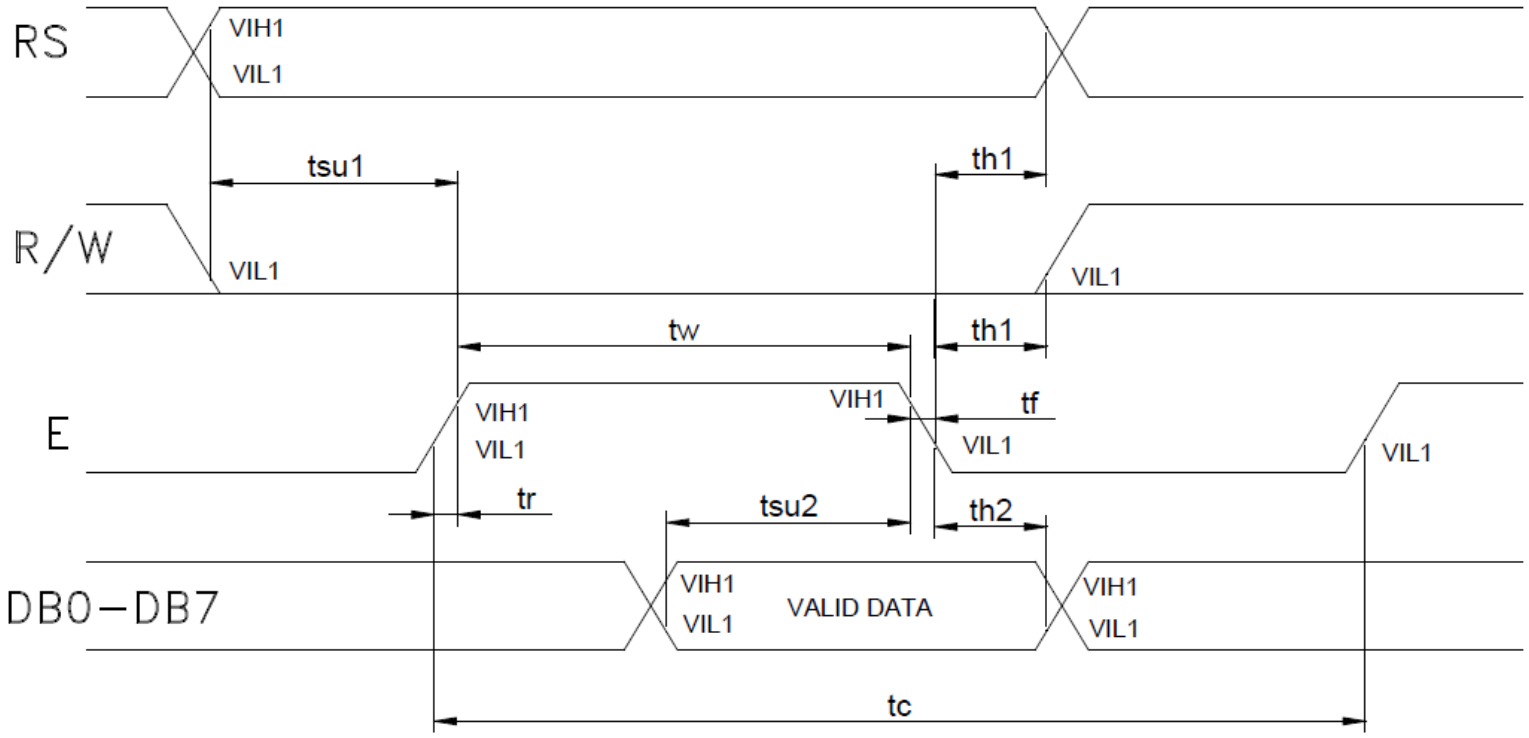
Table 1. Various Kinds of Operations according to RS and R/W Bits

RS	R/W	Operation
L	L	Instruction Write operation (MPU writes Instruction code into IR)
L	H	Read Busy Flag (DB7) and address counter (DB0 - DB6)
H	L	Data Write operation (MPU writes data into DR)
H	H	Data Read operation (MPU reads data from DR)

LCD Connector

Pin no.	Symbol	External connection	Function
1	Vss	Power supply	Signal ground for LCM
2	V_{DD}		Power supply for logic for LCM
3	V_0		Contrast adjust
4	RS	MPU	Register select signal
5	R/W	MPU	Read/write select signal
6	E	MPU	Operation (data read/write) enable signal
7~10	DB0~DB3	MPU	Four low order bi-directional three-state data bus lines. Used for data transfer between the MPU and the LCM. These four are not used during 4-bit operation.
11~14	DB4~DB7	MPU	Four high order bi-directional three-state data bus lines. Used for data transfer between the MPU
15	LED+	LED BKL power supply	Power supply for BKL
16	LED-		Power supply for BKL

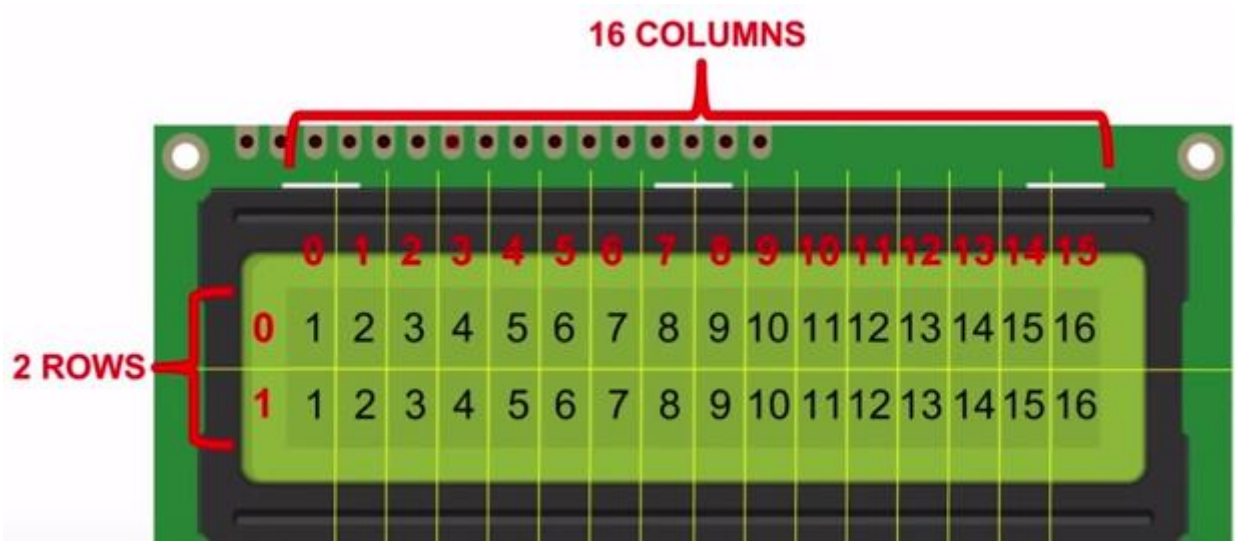
Write Mode Timing



Instruction	Instruction Code										Description	Execution time (fosc=270KHz)
	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM and set DDRAM address to "00H" from AC	1.52ms
Return Home	0	0	0	0	0	0	0	0	1	-	Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.52ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Assign cursor moving direction and enable the shift of entire display	38μs
Display ON/ OFF Control	0	0	0	0	0	0	1	D	C	B	Set display(D), cursor(C), and blinking of cursor(B) on/off control bit.	38μs
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	-	-	Set cursor moving and display shift control bit, and the direction, without changing of DDRAM data.	38μs
Function Set	0	0	0	0	1	DL	N	F	-	-	Set interface data length (DL: 8-bit/4-bit), numbers of display line (N: 2-line/1-line) and, display font type (F:5x10 dots/5x8 dots)	38μs
Set CGRAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter.	38μs
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in counter	38μs
Read Busy Flag and Address Counter	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM).	38μs
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM).	38μs

DDRAM Address

Display position	Column	1	2	- - -	15	16
DD RAM Address (Hex-Decimal)	1-Line	00H	01H	- - -	0EH	0FH
	2-Line	40H	41H	- - -	4EH	4FH



LCD Command

■ 화면 클리어 : LCD 의 화면을 깨끗이 지우고 어드레스 카운터의 값을 '0'으로 만드어 1열 첫 번째 번지부터 글자를 나타낼 수 있다. (실행시간 : 1.53ms)

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	0	1

■ 커서 홈 : 화면을 지우지 않고 어드레스 카운터를 '0'으로 만들어 주고 디스플레이 쉬프트 되었던 것을 본래의 위치로 복귀시킨다. (실행시간 : 1.53ms)

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0	1	*

■ 엔트리 모드 셋 : 커서의 움직이는 방향을 설정하거나, 글자가 표기된 부분을 쉬프트 할 것인지를 결정한다. (실행시간 : 39㎲)

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	1	I/D	S

-. I/D = '1' -> 자동으로 어드레스 증가
-. I/D = '0' -> 자동으로 어드레스 감소
-. S = '0' 으로 설정하는 것이 정상이고 '1'로 설정하면 화면이 쉬프트 되면서 글자가 쓰여지는 매우 혼란스러운 동작을 한다.

LCD Command

■ 디스플레이 ON/OFF : 전체 화면의 ON/OFF, 커서의 ON/OFF, 커서의 위치가 있는 부분의 글자를 깜박이는 것을 설정한다.(실행시간 : 39㎲)

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	1	D	C	B

-. D = '1' 이면, 전체화면이 ON 되어 글자가 나타난다.
-. D = '0' 이면, 전체화면이 OFF 되어 어떤 글자도 나타나지 않는다.
-. C = '1' 이면, 커서가 나타난다.
-. C = '0' 이면, 커서가 나타나지 않는다.
-. B = '1' 이면, 커서는 글자위치에서 깜박이게 된다.
-. B = '0' 이면, 커서는 깜박이지 않는다.

LCD Command

■ 커서 디스플레이 쉬프트 : 현재 LCD 창에 표시된 내용을 좌우로 쉬프트 시키거나 커서를 좌우로 이동시킨다.(실행시간 : 39㎲)

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	1	S/C	R/L	*	*

S/C	R/L	동 작 상 태
0	0	커서는 왼쪽으로 쉬프트 된다.
0	1	커서는 오른쪽으로 쉬프트 된다.
1	0	LCD 창에 표시된 내용과 커서가 왼쪽으로 쉬프트 된다.
1	1	LCD 창에 표시된 내용과 커서가 오른쪽으로 쉬프트 된다.

(단, 디스플레이의 쉬프트만 행해질 때 어드레스 카운터는 변하지 않는다.)

LCD Command

■ LCD 기능설정 : LCD 가 CPU 와 연결되는 데이터 라인, 디스플레이 라인, 글자 폰트 등을 설정한다.(실행시간 : 39㎲)

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	1	DL	N	F	*	*

-. DL = '1' 일 때 8비트(D0~D7)의 데이터 라인을 사용한다.
-. DL = '0' 일 때 4비트(D0~D7)의 데이터 라인을 사용한다.
-. N 은 LCD 디스플레이 라인 수를 설정한다.
-. F는 글자 폰트를 결정한다.

N	F	라 인	폰트	비고
0	0	1	5*8	
0	1	1	5*10	
1	0	2	5*8	5*10 도트는 2 Line 디스플레이 불가능

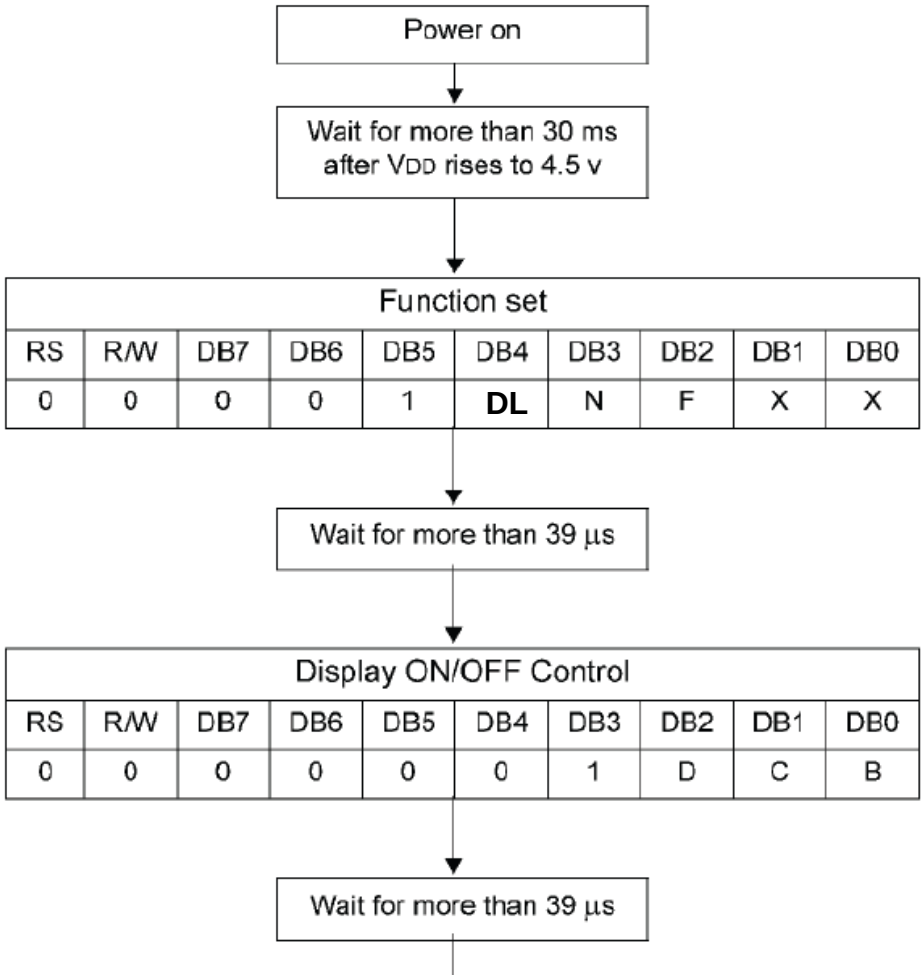
LCD Command

■ D.D RAM 어드레스 설정 : 자신이 원하는 곳에 문자를 디스플레이 하기위해서는 먼저 'Set D.D RAM Address' 명령으로 위치를 지정해 주어야 한다. 램으로 비유하면 데이터가 들어갈 번지를 미리 지정하는 것이 된다. D.D RAM 어드레스는 디스플레이 라인을 설정하는 N 값에 따라 N='0 이면 D.D RAM 어드레스는 0x00~0x47이다. N='1'이면 D.D RAM 어드레스는 첫 번째 라인이 0x00~0x27 이고, 두 번째 라인은 0x40~0x67 이다. (실행시간 : 39㎲)

RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0*

(단, D.D RAM 어드레스 설정은 7비트까지 가능하다.)

Initialization(1)



N	0	1-line mode
	1	2-line mode

F	0	display off
	1	display on

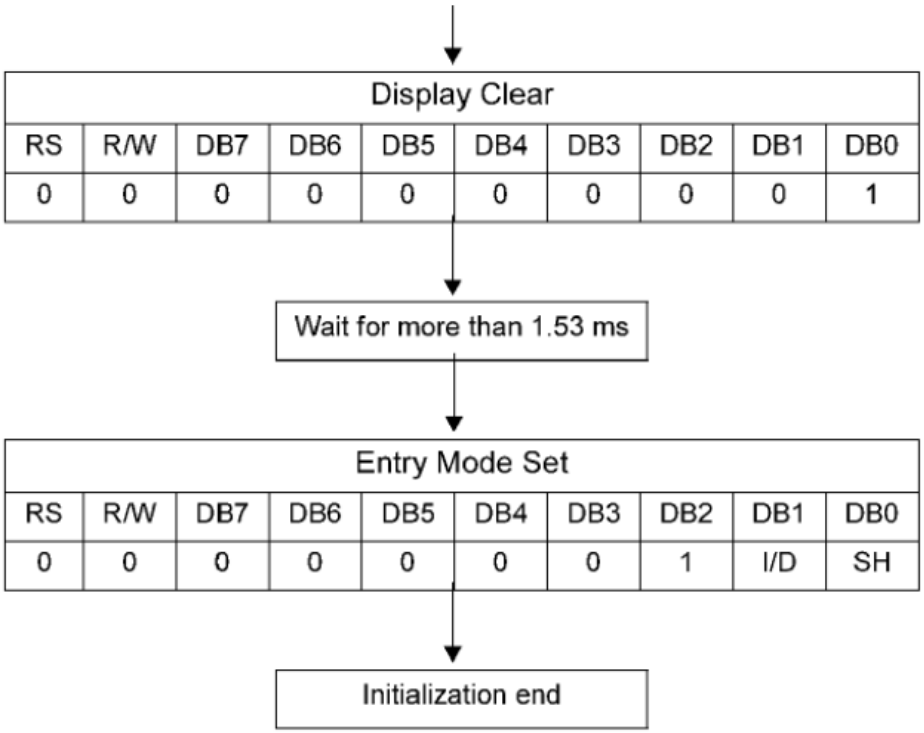
D	0	display off
	1	display on

C	0	cursor off
	1	cursor on

B	0	blink off
	1	blink on

Initialization(2)

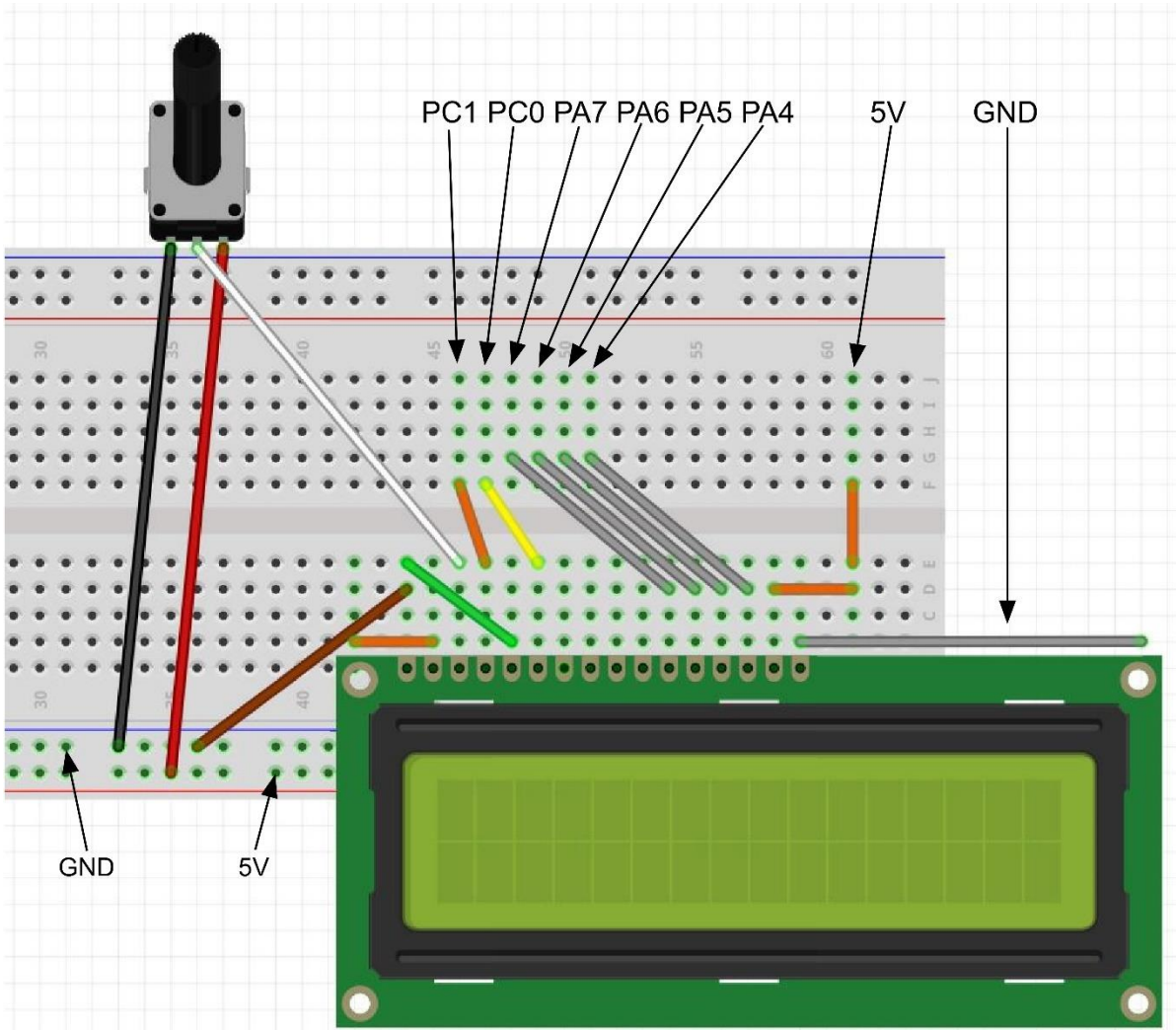
(Continued)



I/D	0	decrement mode
	1	increment mode

SH	0	entire shift off
	1	entire shift on

Connection Diagram



Connection Table

PIN	LCD	ATmega128
1	GND	GND
2	5V	5V
3	Contrast	Potentiometer
4	RS	PC1
5	R/W	GND
6	E	PC0
11	DB4	PA7
12	DB5	PA6
13	DB6	PA5
14	DB7	PA4
15	LED+	5V
16	LED-	GND

Sample Code(1)

```
/* PORTA4:D4, PORTA5:D5, PORTA6:D6, PORTA7:D7 */
/* PORTC0:E, PORTC1:RS, GND:R/W */
//#define F_CPU 16000000U
#include <avr/io.h>
//#include <util/delay.h>
#include <stdio.h>
#define LCD_DATA PORTA //LCD데이터 포트 정의
#define LCD_INST PORTA
#define LCD_CTRL PORTC //LCD제어 포트 정의
#define RS 0x01
#define EN 0x00
void delay_us(unsigned char time_us)
{
	register unsigned char i;
	for(i=0;i<time_us;i++) //4 cycle
	{
		asm volatile("PUSH R0"); //2 cycle
		asm volatile("POP R0"); //2 cycle
		asm volatile("PUSH R0"); //2 cycle
		asm volatile("POP R0"); //2 cycle
		asm volatile("PUSH R0"); //2 cycle
		asm volatile("POP R0"); //2 cycle
		// Sum = 16 cycle=1 us for 16MHz
	}
}
```

Sample Code(2)

```
void delay_ms(unsigned int time_ms)
{
	register unsigned int i;
	for(i=0;i<time_ms;i++) //4 cycle
	{
		delay_us(250);
		delay_us(250);
		delay_us(250);
		delay_us(250);
	}
}
char flip_bits(char ch)
{
	char return_ch=0;
	return_ch=(ch>>3) & 0x10;
	return_ch=return_ch | ((ch>>1) & 0x20);
	return_ch=return_ch | ((ch<<1) & 0x40);
	return_ch=return_ch | ((ch<<3) & 0x80);
	return return_ch;
}
```

Sample Code(3)

```
void LCD_data(char ch)
{
	LCD_DATA=flip_bits(ch); //upper nibble 데이터 출력
	LCD_CTRL |= (1<<RS);
	LCD_CTRL |= 1<<EN;
	delay_us(1);
	LCD_CTRL &= ~(1<<EN);
	delay_us(20);

	LCD_DATA=flip_bits(ch<<4); //lower nibble 데이터 출력
	LCD_CTRL |= (1<<RS);
	LCD_CTRL |= 1<<EN;
	delay_us(1);
	LCD_CTRL &= ~(1<<EN);
	delay_us(50);
}
```

Sample Code(4)

```
void LCD_comm(char ch)
{
	LCD_INST=flip_bits(ch); //upper nibble 명령어 쓰기
	LCD_CTRL &= ~(1<<RS);
	LCD_CTRL |= 1<<EN;
	delay_us(1);
	LCD_CTRL &= ~(1<<EN);
	delay_us(5);

	LCD_INST=flip_bits(ch<<4); //lower nibble 명령어 쓰기
	LCD_CTRL &= ~(1<<RS);
	LCD_CTRL |= 1<<EN;
	delay_us(1);
	LCD_CTRL &= ~(1<<EN);
	delay_ms(5);
}
void LCD_CHAR(char c) // 한문자 출력 함수
{
	delay_ms(1);
	LCD_data(c); // DDRAM으로 데이터 전달
}
void LCD_STR(char *str) // 위의 문자열을 한문자씩 출력함수로 전달 ,출력해줄 문자열을 받음
{
	while(*str)
	LCD_CHAR(*str++);
}
```

Sample Code(5)

```
void LCD_pos(char col, char row) // LCD 포지션 설정
{
    LCD_comm(0x80|(col+row*0x40)); //문자 처음 출력해줄 위치 설정
}
void LCD_clear(void) // 화면 클리어
{
    LCD_comm(1);
}
void LCD_init(void) // LCD 초기화
{
    LCD_comm(0x20); //4bit 초기화
    delay_ms(5);
    LCD_comm(0x28); // Function Set 데이터 4비트 사용, 5X7도트 , LCD2열로 사용
    delay_ms(5);
    LCD_comm(0x0C); // Display ON/OFF
    delay_ms(5);
    LCD_comm(0x06); // Entry Mode Set 주소 +1 , 커서를 우측으로 이동
    delay_ms(5);
    LCD_clear(); // Display Clear
}
```

Sample Code(6)

```
int main(void)
{
	char str[20]="LCD test..	";
	DDRA=0xF0;
	DDRC |= (1<<RS)|(1<<EN); // PC0~1까지 출력으로 사용
	LCD_init(); //LCD 초기화
	LCD_init(); //LCD 초기화
	LCD_pos(0,0);
	sprintf(str,"Hello");
	LCD_STR(str);
	LCD_pos(0,1);
	sprintf(str,"World");
	LCD_STR(str);
}
```

Exercise

- 먼저, 다음과 같은 함수를 완성하고 시험해 본다.

```
void display_digits(unsigned int count, unsigned int number) {  }
```

- 위 함수는 2개의 입력 변수 count, number 를 가지며 각각은 0~99 까지의 숫자이다.
- 입력 변수 count의 숫자는 LCD의 1번째 줄에 2자리 숫자로, 입력 변수 number의 숫자는 LCD의 2번째 줄에 2자리 숫자로 나타낸다.
- 예:count=5, number=12 **(한자리 숫자에는 leading zero 추가)**

```
CNT:05
NUM:12
```

Exercise

- Lab3의 버튼 스위치 카운터 예제를 seven segment 가 아닌 character LCD를 이용하여 구현한다.
- 앞 페이지에서 구현한 LCD 디스플레이 함수를 사용한다. Lab3와 마찬가지로, 함수의 count 변수는 버튼 스위치를 누른 횟수이며, 함수의 number는 스위치를 누름에 따라 증가하거나 멈추는 숫자이다.