

Lecture 05

LCD 제어

LCD 모듈 소개

16x2 LCD(Liquid Crystal Display)

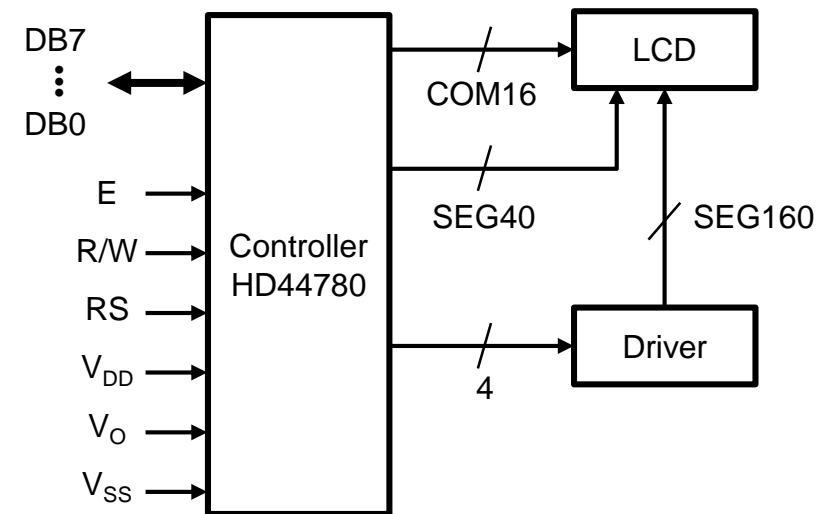


앞면

뒷면



- 16문자 2라인
- 4비트, 8비트 MCU와 인터페이스
- 5x7 도트, 5x10 도트 디스플레이 폰트
- 240 문자 폰트의 문자 발생기 ROM(CGROM)
- 64x8비트 문자발생기 RAM(CGRAM)



LCD 모듈 소개

■ 인터페이스 핀

1. VSS : GND(0V)
2. VDD : VCC(5V)
3. VO : 가변저항 10k를 달아서 글자의 밝기를 조절해줌
4. RS : 입력신호만 받는 단자로서 LCD의 제어 명령/데이터 입력 제어 신호를 설정함
 - 0이면 명령 입력 설정(IR 선택)
 - 1이면 데이터 입력 설정(DR 선택)
5. R/W : 읽기/쓰기 단자 Read/Write 신호를 관장함
 - 0이면 쓰기 동작으로서 MCU에서 LCD 모듈로의 데이터 쓰기
 - 1이면 읽기 동작으로서 LCD모듈에서 MCU로의 데이터 읽기
6. E : Enable 신호 명령이 하나 입력될 때마다 펄스를 하나씩 내줌
 - 0이면 LCD가 동작하지 않음
 - 1이면 LCD가 동작함
7. DB0~DB7 : 데이터 버스

LCD 모듈 소개

- 인터페이스 핀

핀번호	기호	신호레벨	핀설명	비고
1	VSS	-	Ground	0V
2	VDD	-	Vcc	5V±5%
3	VO	H/L	Text Light	by user
4	RS	H/L	Register selection	H:data L:instruction
5	R/W	H,H→L	Read/Write	H:read L:write
6	E	H/L	Enable signal	
7	DB0	H/L	Data 0	4bit사용시 DB4~DB7을 사용
8	DB1	H/L	Data 1	
9	DB2	H/L	Data 2	
10	DB3	H/L	Data 3	
11	DB4	H/L	Data 4	
12	DB5	H/L	Data 5	
13	DB6	H/L	Data 6	
14	DB7	H/L	Data 7	

LCD 모듈 소개

Instruction set

- I/D = 1 : 어드레스 자동 증가
- I/D = 0 : 어드레스 자동 감소
- S = 1 : 전체 시프트
- S = 0 : 시프트 안함
- S/C = 1 : 표시 시프트
- S/C = 0 : 커서 이동
- R/L = 1 : 오른쪽으로 시프트
- R/L = 0 : 왼쪽으로 시프트
- DL = 1 : 8비트
- DL = 0 : 4비트
- N = 1 : 2라인
- N = 0 : 1라인
- F = 1 : 5×10 도트
- F = 0 : 5×8 도트
- BF = 1 : 내부 동작 중
- BF = 0 : 명령/데이터 받기 가능

- DDRAM : 표시 데이터 RAM
- CGRAM : 폰트 제작용 RAM
- ACG : CGRAM 어드레스
- ADD : DDRAM 어드레스
- AC : 어드레스 카운터

명령	제어명령		데이터									설명	실행 시간	
	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	hex			
명령 쓰기	화면 클리어	0	0	0	0	0	0	0	0	0	1	0x01	표시클리어, 커서홈	1,64ms
	커서홈	0	0	0	0	0	0	0	0	1	*	0x02	커서홈	1,64ms
	엔트리 모드셀	0	0	0	0	0	0	0	1	I/D	S	0x06	커서방향, 표시이동(x)	40us
													0x07	
	표시 온/오프	0	0	0	0	0	0	1	D Display 표시	C Cursor 커서	B Blink 블링크	0x08	표시(x), 커서(x), 블링크(x)	40us
												0x0c	표시(0), 커서(x), 블링크(x)	
												0x0e	표시(0), 커서(0), 블링크(x)	
												0x0f	표시(0), 커서(0), 블링크(0)	
	표시 시프트	0	0	0	0	0	1	S/C	R/L	*	*	0x18	커서, 표시내용을 좌로 이동	40us
												0x1c	커서, 표시내용을 우로 이동	
	평선셀	0	0	0	0	1	DL	N	F	*	*	0x28	Data선4비트, 2줄표시, 5x7도트	40us
												0x38	Data선8비트, 2줄표시, 5x7도트	
	CG 램주소	0	0	0	1	Address						0x40~0x7f	CG램 어드레스 설정	40us
	DD 램주소	0	0	1	Address						0x80~0xff	DD램 어드레스 설정	40us	
명령 읽기	BF주소 읽기	0	1	BF	Address							BF, AC 내용읽기	40us	
데이터 쓰기	데이터 쓰기	1	0	쓸 DATA							데이터를 쓴다	40us		
데이터 읽기	데이터 읽기	1	1	읽을 DATA							데이터를 읽는다	40us		

LCD 모듈 소개

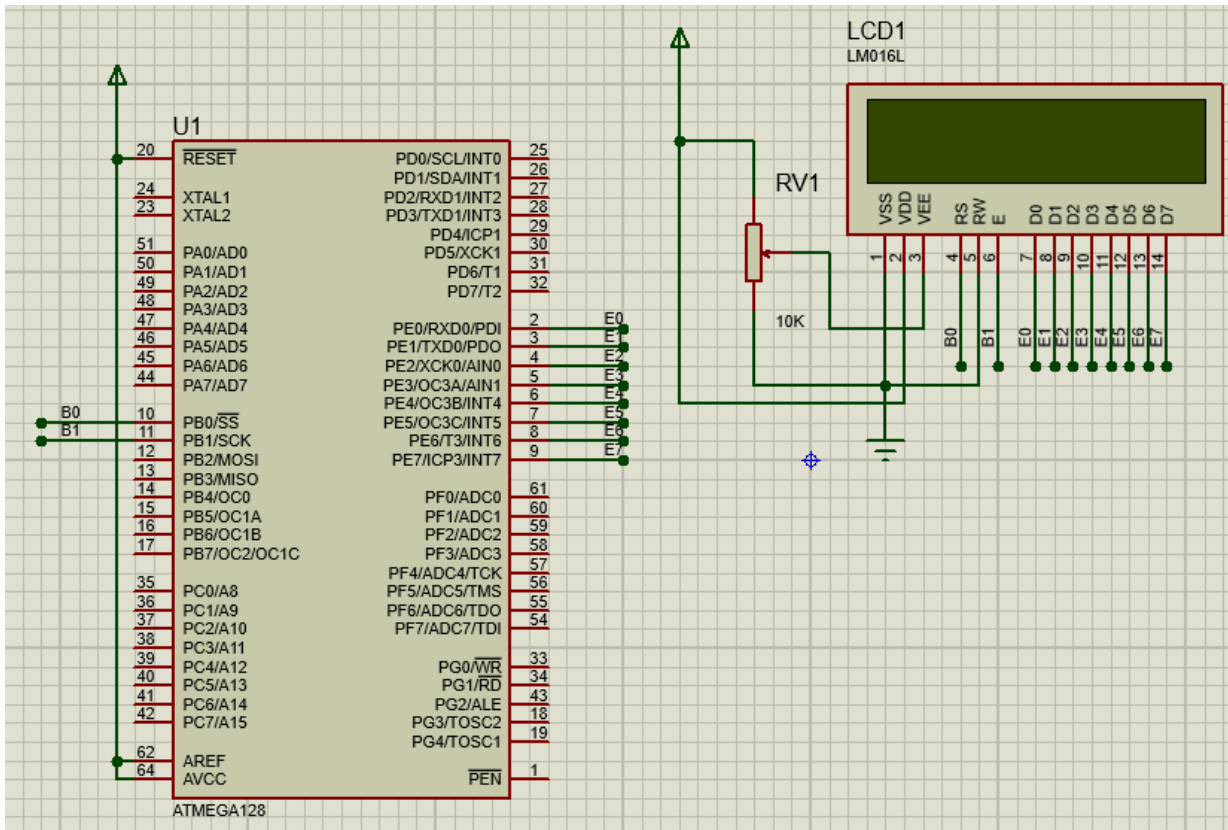
- 레지스터
 - 명령 레지스터(IR)
 - CPU로부터 LCD 표시에 관련된 제어명령을 쓰는 용도
 - LCD 모듈의 제어상태를 확인하는 용도
 - 데이터 레지스터(DR)
 - DDRAM 혹은 CGRAM에 써 넣기 위한 데이터를 저장하는 용도
 - DDRAM 혹은 CGRAM으로부터 읽어진 데이터의 일시적인 저장 용도

RS	R/W	동 작 기 능
0	0	IR 선택하여 제어명령쓰기(디스플레이 클리어 등)
0	1	DB7로부터 비지플래그를 읽기/어드레스카운터를 DB0-6으로부터 읽기
1	0	DR선택하여 데이터 값을 쓰기(DR에서 DDRAM혹은 CGRAM으로)
1	1	DR선택하여 데이터 값을 읽기 (DDRAM혹은 CGRAM에서 DR로)

실습1: 8비트 버스 사용



■ Proteus





실습1: 8비트 버스 사용

▪ LCD 제어 핀 할당

- RS, E : 포트B
- DB7, ..., DB0: 포트E
- R/W = 0 : MCU에서 LCD 모듈로의 데이터 쓰기

```
#define LCD_DATA_DIR DDRE
#define LCD_DATA_PORT PORTE
#define LCD_CTRL_DIR DDRB
#define LCD_CTRL_PORT PORTB
```

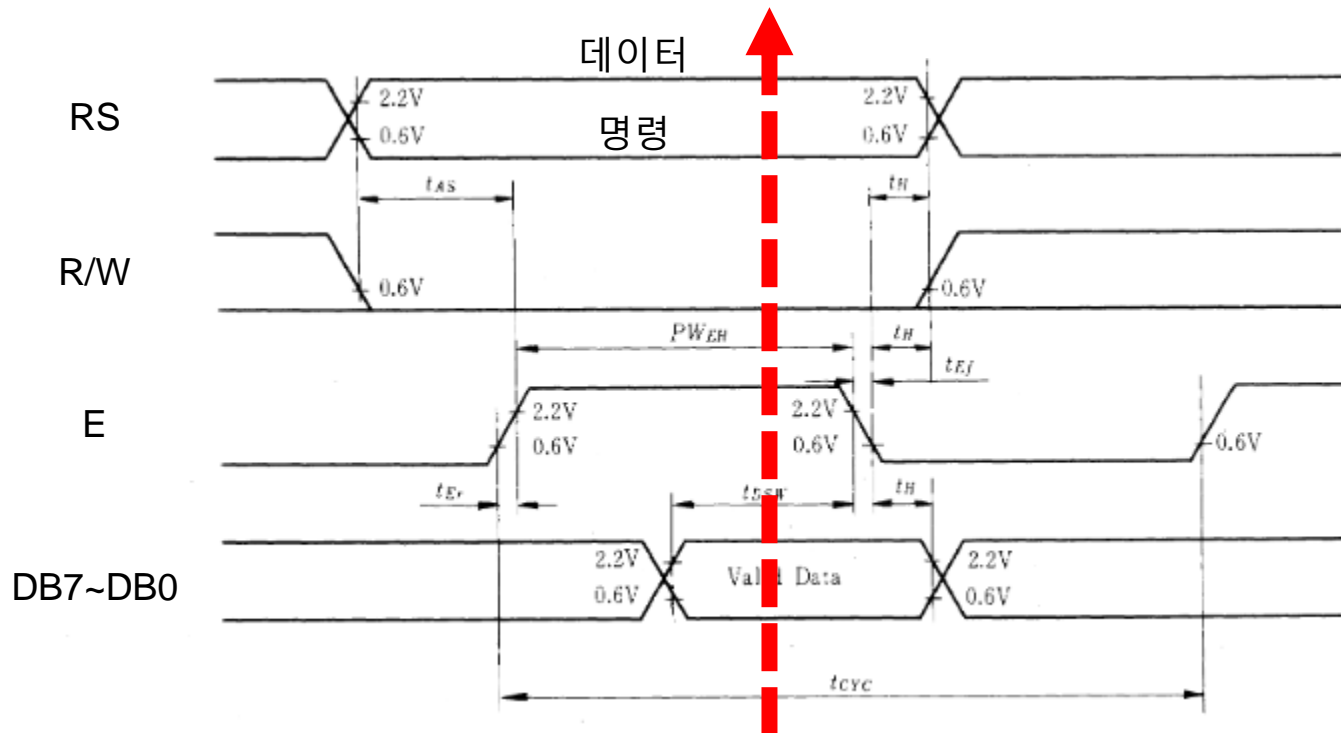
▪ 자주 사용하는 RS와 E 신호 설정

```
#define LCD_E_HIGH (LCD_CTRL_PORT |= 0x02)
#define LCD_E_LOW (LCD_CTRL_PORT &= 0xfd)
#define LCD_RS_HIGH (LCD_CTRL_PORT |= 0x01)
#define LCD_RS_LOW (LCD_CTRL_PORT &= 0xfe)
```


실습1: 8비트 버스 사용

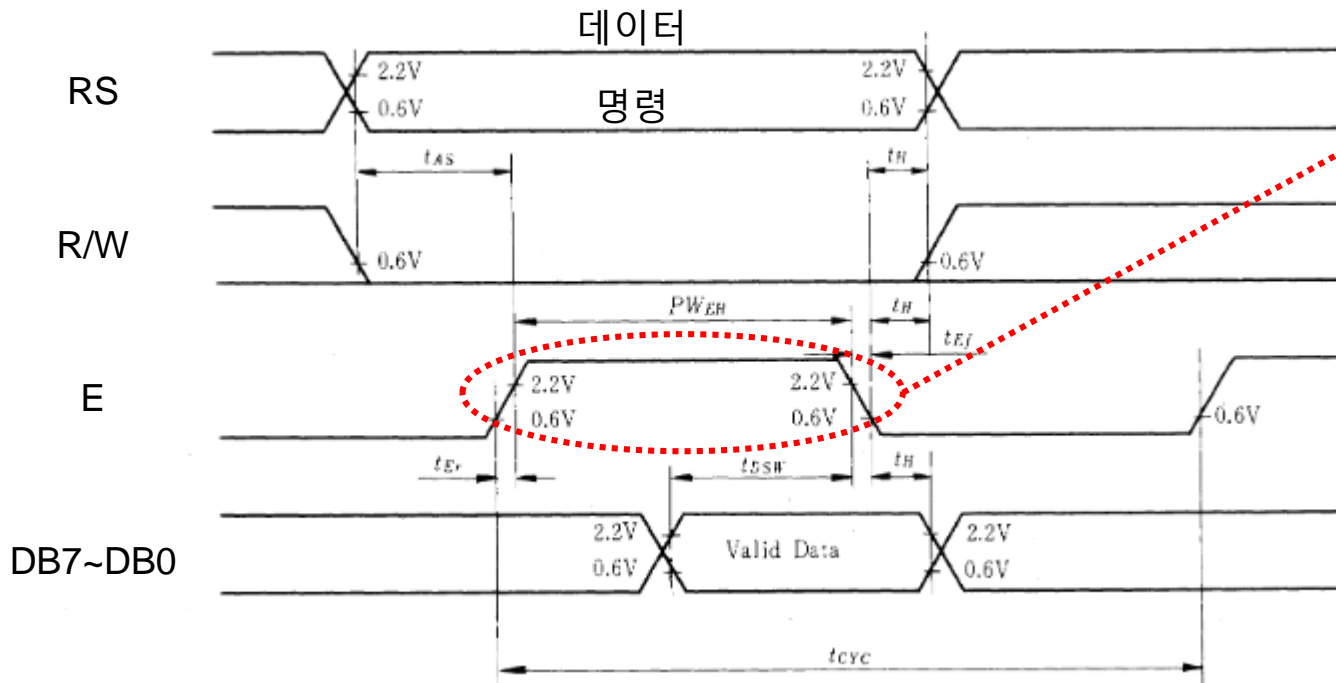
명령/데이터 쓰기

데이터 쓰기 : RS = 1, R/W = 0, E = 1, 데이터를 DB7~DB0에 넣음
 명령 쓰기 : RS = 0, R/W = 0, E = 1, 명령을 DB7~DB0에 넣음



실습1: 8비트 버스 사용

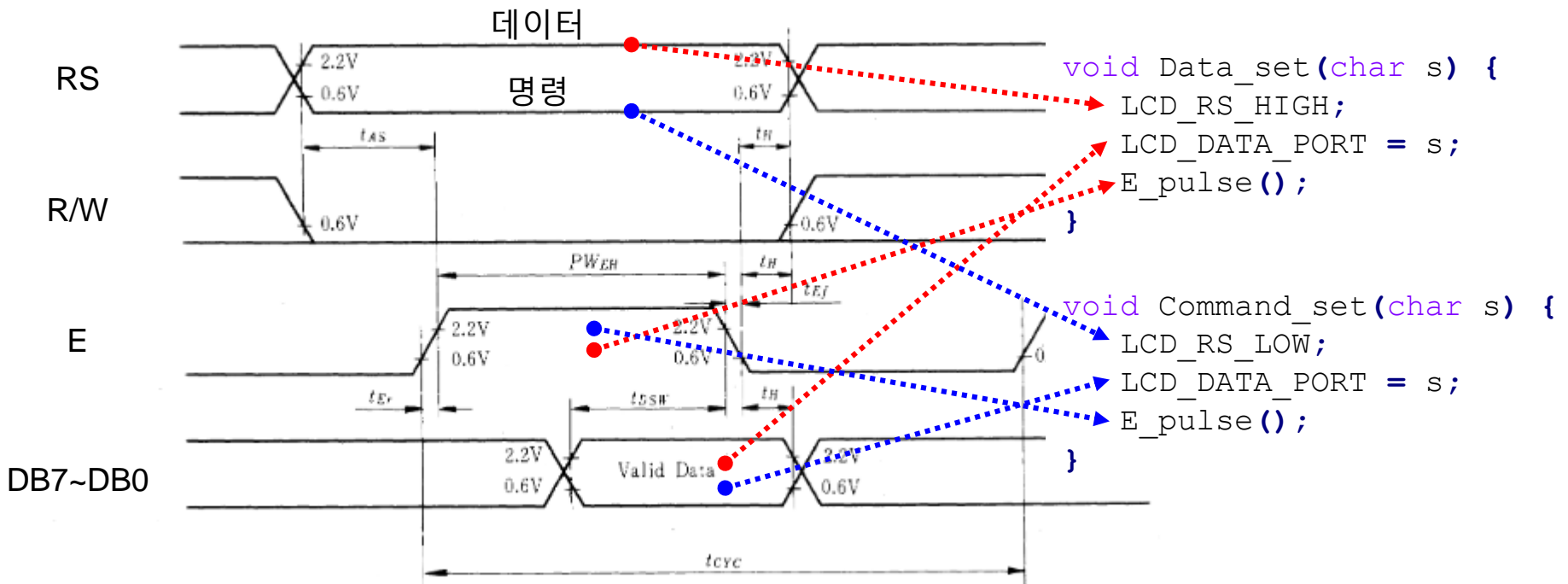
명령/데이터 쓰기



```
void E_pulse(void) {
    LCD_E_HIGH;
    delay(5);
    LCD_E_LOW;
}
```

실습1: 8비트 버스 사용

명령/데이터 쓰기



실습1: 8비트 버스 사용

■ LCD 초기화

```
void init_lcd(void) {
    Command_set(0x38);
    Command_set(0x0f);
    Command_set(0x06);
}
```

명령	제어명령		데이터									설명	실행 시간	
	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	hex			
명령 쓰기	화면 클리어	0	0	0	0	0	0	0	0	0	1	0x01	표시클리어, 커서홈	1,64ms
	커서홈	0	0	0	0	0	0	0	0	1	*	0x02	커서홈	1,64ms
	엔트리 모드셀	0	0	0	0	0	0	0	1	I/D	S	0x06	커서방향, 표시이동(x)	40us
												0x07	커서방향, 표시이동(0)	
	표시 온/오프	0	0	0	0	0	0	1	D Display 표시	C Cursor 커서	B Blink 블링크	0x08	표시(x), 커서(x), 블링크(x)	40us
												0x0c	표시(0), 커서(x), 블링크(x)	
												0x0e	표시(0), 커서(0), 블링크(x)	
												0x0f	표시(0), 커서(0), 블링크(0)	
	표시 시프트	0	0	0	0	0	1	S/C	R/L	*	*	0x18	커서, 표시내용을 좌로 이동	40us
												0x1c	커서, 표시내용을 우로 이동	
	평선셀	0	0	0	0	1	DL	N	F	*	*	0x28	Data선4비트, 2줄표시, 5x7도트	40us
												0x38	Data선8비트, 2줄표시, 5x7도트	
CG 램주소	0	0	0	1	Address						0x40~0x7f	CG램 어드레스 설정	40us	
DD 램주소	0	0	1	Address						0x80~0xff	DD램 어드레스 설정	40us		
명령 읽기	BF주소 읽기	0	1	BF	Address							BF, AC 내용읽기	40us	
데이터 쓰기	데이터 쓰기	1	0	쓸 DATA							데이터를 쓴다	40us		
데이터 읽기	데이터 읽기	1	1	읽을 DATA							데이터를 읽는다	40us		

실습1: 8비트 버스 사용

▪ LCD에 글자 출력

```
void cursor_at(char x, char y)
{
    switch (y) {
        case 0: y = 0x80; break;
        case 1: y = 0xc0; break;
        case 2: y = 0x94; break;
        case 3: y = 0xd4; break;
    }
    y = y+x;
    Command_set (y);
}

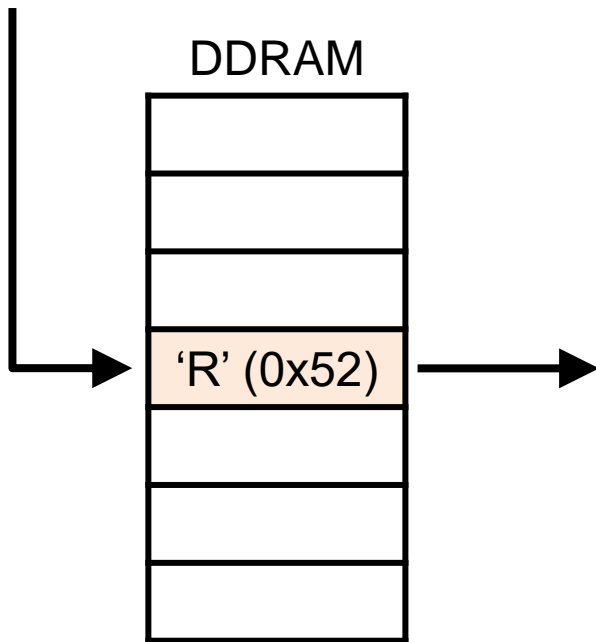
void writeString_lcd(
    char x,
    char y,
    const char *str)
{
    cursor_at(x, y);
    while (*str)
        Data_set (*str++);
}
```

명령	제어명령		데이터									설명	실행 시간	
	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	hex			
화면 클리어 커서홈	0	0	0	0	0	0	0	0	0	0	1	0x01	표시클리어, 커서호출	1.64ms
	0	0	0	0	0	0	0	0	0	1	I/D	S	I/D = 1 : 어드레스 자동 증가	4ms
	0	0	0	0	0	0	0	0	1	I/D	S	0x06 0x07	커서방향, 표시이동(x) 커서방향, 표시이동(0)	40us
명령 쓰기	0	0	0	0	0	0	0	1	D Display 표시	C Cursor 커서	B Blink 블링크	0x08 0x0c 0x0e 0x0f	표시(x), 커서(x), 블링크(x) 표시(0), 커서(x), 블링크(x) 표시(0), 커서(0), 블링크(x) 표시(0), 커서(0), 블링크(0)	40us
	0	0	0	0	0	1	S/C	R/L	*	*	0x18 0x1c	커서, 표시내용을 좌로 이동 커서, 표시내용을 우로 이동	40us	
	0	0	0	0	1	DL	N	F	*	*	0x28 0x38	Data선4비트, 2줄표시, 5x7도트 Data선8비트, 2줄표시, 5x7도트	40us	
	0	0	0	1	Address						0x40 ~ 0x7f	CG램 어드레스 설정	40us	
	0	0	1	Address						0x80~ 0xff	DD램 어드레스 설정	40us		
	0	1	BF	Address								BF, AC 내용읽기	40us	
데이터 쓰기	1	0	쓸 DATA									데이터를 쓴다	40us	
데이터 읽기	1	1	읽을 DATA									데이터를 읽는다	40us	

실습1: 8비트 버스 사용

- LCD에 글자 출력 : DDRAM과 CGROM의 역할

```
Data_set('R');
```



Lower 4 Bits	Upper 4 Bits	0000	0001	0010	0011	0100	0101	110	0111	1000	1001	1010	1011	1100	1101	1110	1111		
xxxx0000	CG ROM (1)				0	a	P	^	P					-	9	≡	α	p	
xxxx0001	(2)			!	1	A	Q	a	9					。	ア	チ	4	ä	q
xxxx0010	(3)			"	2	B	R	b	r					「	イ	ツ	×	β	θ
xxxx0011	(4)			#	3	C	S	c	s					」	ウ	テ	ε	ε	∞
xxxx0100	(5)			\$	4	D	T	d	t					、	イ	ト	フ	μ	Ω
xxxx0101	(6)			%	5	E	U	e	u					・	オ	ナ	1	ε	Ü
xxxx0110	(7)			&	6	F	V	f	v					ヲ	カ	ニ	ヨ	ρ	Σ
xxxx0111	(8)			'	7	G	W	g	w					ア	キ	ヌ	ラ	g	π
xxxx1000	(1)			(8	H	X	h	x					イ	ク	ネ	リ	フ	×
xxxx1001	(2))	9	I	Y	i	y					ウ	ケ	ル	レ	レ	γ
xxxx1010	(3)			*	:	J	Z	j	z					エ	コ	ン	レ	j	〒
xxxx1011	(4)			+	;	K	[k	[オ	サ	ヒ	ロ	*	π
xxxx1100	(5)			,	<	L	¥	l	l					カ	シ	フ	ワ	φ	円
xxxx1101	(6)			-	=	M]	m]					ユ	ズ	ヘ	ン	も	÷
xxxx1110	(7)			.	>	N	^	n	→					ヨ	セ	ホ	°	ñ	
xxxx1111	(8)			/	?	O	_	o	←					ツ	ソ	マ	°	ö	■

CGROM

CGROM 0x52 アドレス에서 미리 저장한 패턴

1	1	1	1	0
1	0	0	0	1
1	0	0	0	1
1	1	1	1	0
1	0	1	0	0
1	0	0	1	0
1	0	0	0	1
0	0	0	0	0

실습1: 8비트 버스 사용



▪ 다른 보조 함수

```
void cirscr(void) {
    Command_set(0x01);
    delay(5);
}

void cursor_home(void) {
    Command_set(0x02);
    delay(5);
}

void move_display(char p) {
    if (p==LEFT) Command_set(0x18);
    else if (p==RIGHT) Command_set(0x1c);
}

void move_cursor(char p) {
    if (p==RIGHT) Command_set(0x14);
    else if (p==LEFT) Command_set(0x10);
}
```

```
void Entry_shift(char p) {
    if (p==RIGHT) Command_set(0x05);
    else if (p==LEFT) Command_set(0x07);
    else if (p==NO) Command_set(0x06);
}

void display_onoff(
    unsigned char d,
    unsigned char c,
    unsigned char b)
{
    unsigned char display = 0x08;
    if (d==ON) d = 0x04;
    else d = 0x00;
    if (c==ON) c = 0x02;
    else c = 0x00;
    if (b==ON) b = 0x01;
    else b = 0x00;
    display = display | d | c | b;
    Command_set(display);
}
```

실습1: 8비트 버스 사용



- LCD 제어 소스 코드

```
#include <xc.h>
#include <util/delay.h>
#include <avr/io.h>
#include "lcd1.h"

int main(void) {
    DDRE = DDRB = 0xff;

    init_lcd();

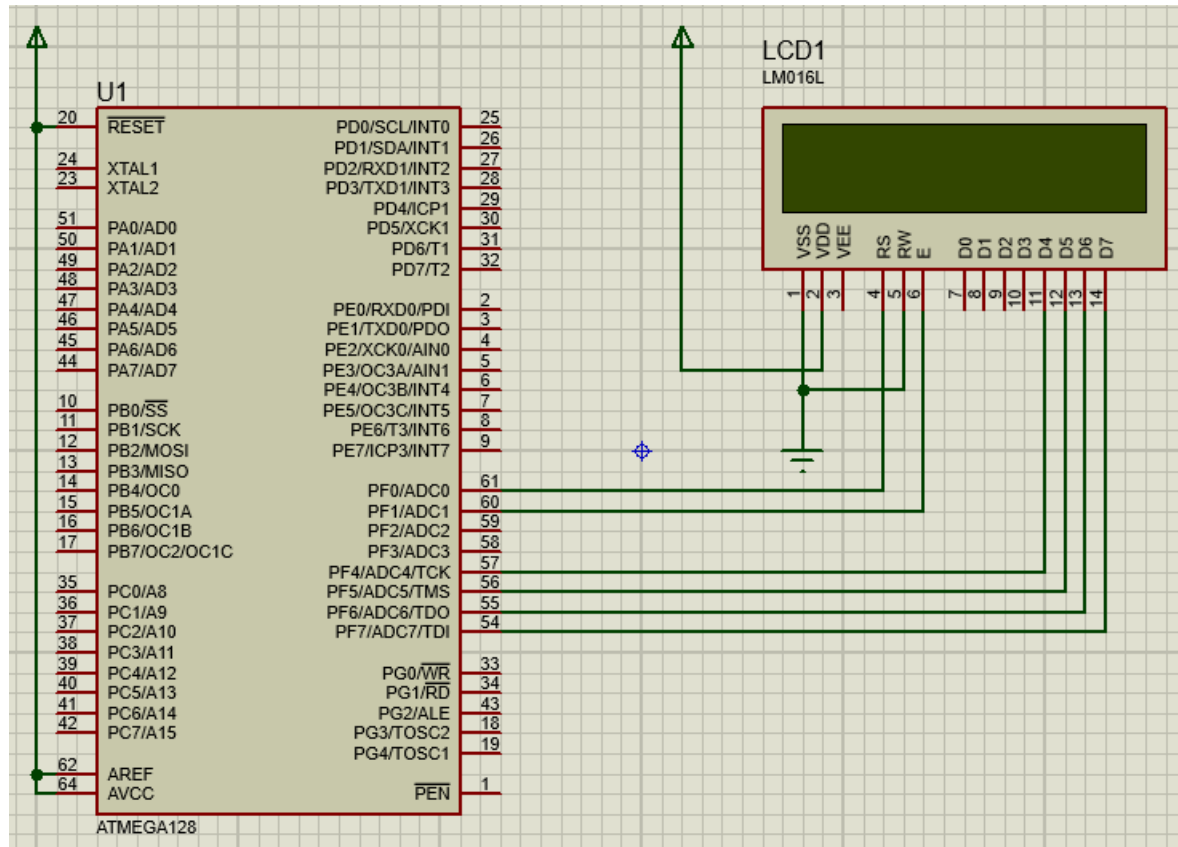
    writeString_lcd(0, 0, "Welcome");
    writeString_lcd(0, 1, "Have a nice day");

    while(1);
}
```


실습2: 4비트 버스 사용



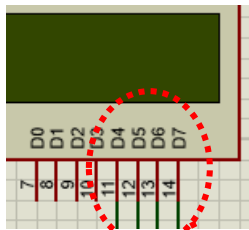
■ Proteus



실습2: 4비트 버스 사용

▪ LCD 초기화

```
void init_lcd4(void) {
    LCD_DATA_PORT = 0xff;
    Command_set4(0x28);
    Command_set4(0x06);
    Command_set4(0x0c);
    cirscr4();
}
```



DB3~DB0 연결 안 되는데
8비트 데이터 전송은???

명령	제어명령		데이터									설명	실행 시간	
	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	hex			
명령 쓰기	화면 클리어	0	0	0	0	0	0	0	0	0	1	0x01	표시클리어, 커서홈	1,64ms
	커서홈	0	0	0	0	0	0	0	0	1	*	0x02	커서홈	1,64ms
	엔트리 모드셀	0	0	0	0	0	0	0	1	I/D	S	0x06	커서방향, 표시이동(x)	40us
												0x07	커서방향, 표시이동(0)	
	표시 온/오프	0	0	0	0	0	0	1	D	C	B	0x08	표시(x), 커서(x), 블링크(x)	40us
												0x0c	표시(0), 커서(x), 블링크(x)	
												0x0e	표시(0), 커서(0), 블링크(x)	
	표시 시프트	0	0	0	0	0	1	S/C	R/L	*	*	0x0f	표시(0), 커서(0), 블링크(0)	40us
												0x18	커서, 표시내용을 좌로 이동	
												0x1c	커서, 표시내용을 우로 이동	
	평선셀	0	0	0	0	1	DL	N	F	*	*	0x28	Data선4비트, 2줄표시, 5x7도트	40us
												0x38	Data선8비트, 2줄표시, 5x7도트	
	CG 램주소	0	0	0	1	Address						0x40~0x7f	CG램 어드레스 설정	40us
DD 램주소	0	0	1	Address						0x80~0xff	DD램 어드레스 설정	40us		
명령 읽기	BF주소 읽기	0	1	BF	Address							BF, AC 내용읽기	40us	
데이터 쓰기	데이터 쓰기	1	0	쓸 DATA							데이터를 쓴다	40us		
데이터 읽기	데이터 읽기	1	1	읽을 DATA							데이터를 읽는다	40us		

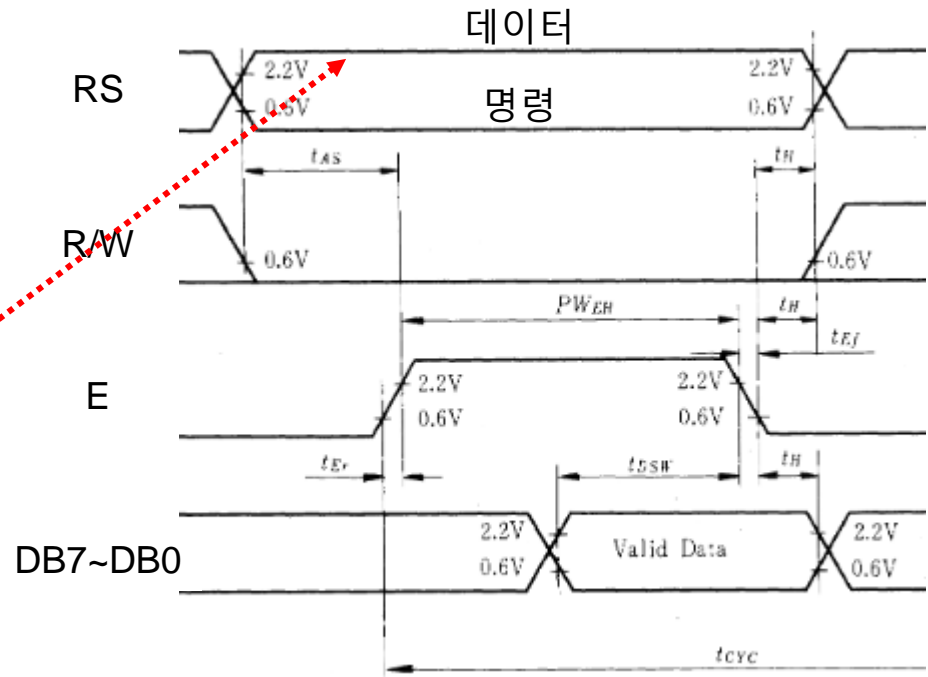
실습2: 4비트 버스 사용

명령/데이터 쓰기

```

void Command_set4(char s) {
    LCD_RS_LOW;
    LCD_DATA_PORT = (s&0xf0);
    E_pulse();
    LCD_DATA_PORT = (s<<4);
    E_pulse();
}

void Data_set4(char s) {
    LCD_RS_HIGH;
    LCD_DATA_PORT = (s&0xf0) | 0x01;
    E_pulse();
    LCD_DATA_PORT = (s<<4) | 0x01;
    E_pulse();
}
    
```



실습2: 4비트 버스 사용



- LCD 제어 소스 코드

```
#include <xc.h>
#include <avr/io.h>
#include <util/delay.h>
#include "lcd4.h"

int main(void) {
    DDRF = 0xff;

    init_lcd4();

    writeString_lcd4(0, 0, "Welcome");
    writeString_lcd4(0, 1, "Have a nice day");

    while(1);
}
```