[05_ATmega128 LCD & ADC]

한국폴리텍대학교 성남캠퍼스

1 LCD(Liquid Crystal Display)

1 LCD(Liquid Crystal Display)

Text LCD

액정

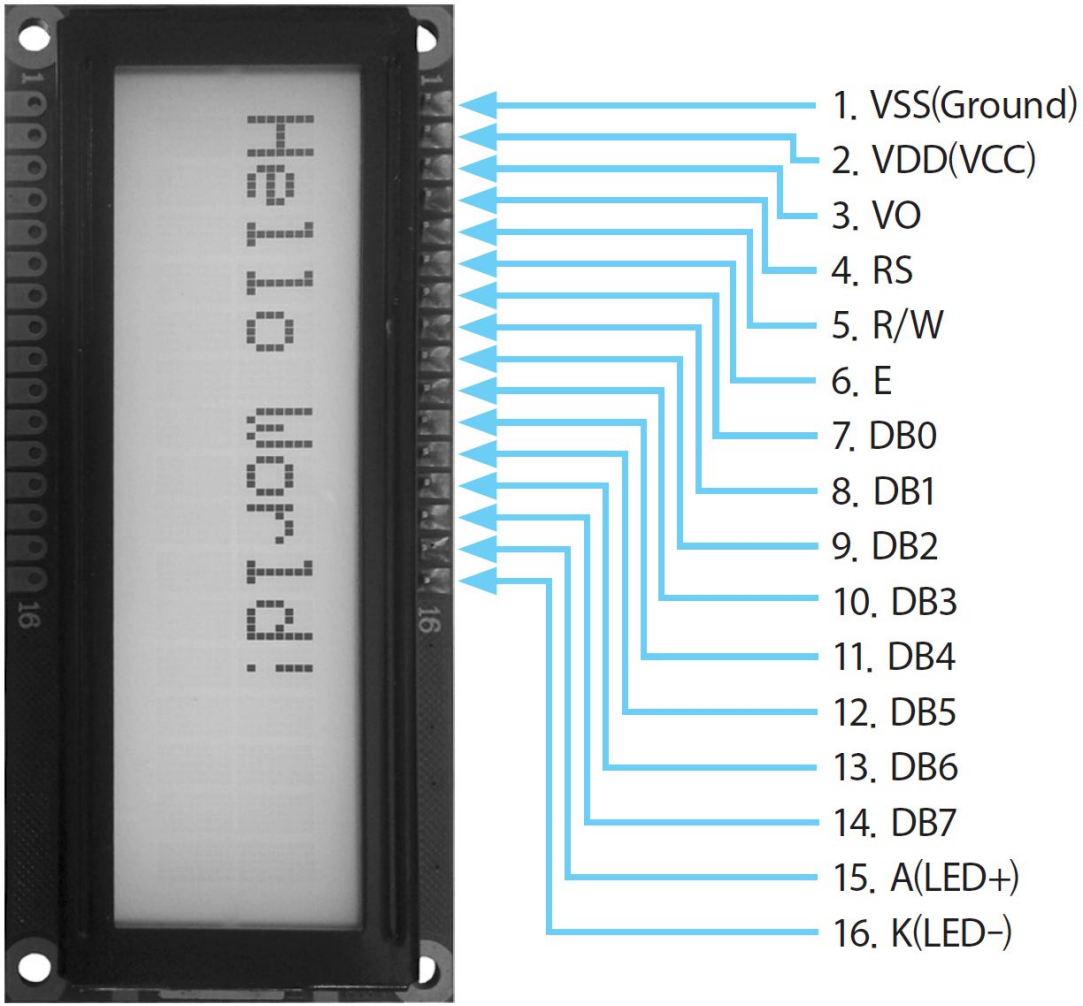
- 액체와 고체의 중간 형태 물질
- 1960년대 이후 디스플레이로 사용되기 시작

텍스트 LCD

- 글자 단위의 정보 출력
- ASCII 코드 정의 문자를 기본으로 함
- 2줄 16글자씩 총 32글자를 표시할 수 있는 텍스트 LCD가 일반적임

1 LCD(Liquid Crystal Display)

- Text LCD



1 LCD(Liquid Crystal Display)

Text LCD

핀 번호	이름	설명
1	VSS	그라운드(GND)
2	VDD	5V 동작 전원(VCC)
3	VO	LCD 전원으로 가변저항을 통해 0~5V 사이 입력
4	RS	레지스터 선택(Register Select)
5	R/W	읽기/쓰기(Read/Write)
6	E	활성화(Enable)
7	DB0	데이터 신호 핀
8	DB1	
9	DB2	
10	DB3	
11	DB4	
12	DB5	
13	DB6	
14	DB7	
15	A(LED+)	백라이트 전원
16	K(LED-)	

1 LCD(Liquid Crystal Display)

Text LCD

- 3개의 제어핀으로 동작 설정
- 4개 또는 8개 데이터 핀으로 제어 또는 표시 데이터 전송
- 제어 순서
  - ✓ RS 및 R/W 핀의 데이터 설정 (1비트 단위)
  - ✓ DBn 핀에 데이터 설정 (4비트 또는 8비트 단위)
  - ✓ E 핀의 하향 Edge에서 실제 동작 발생

제어 핀	설명
RS	텍스트 LCD를 제어하기 위해 제어 레지스터와 데이터 레지스터, 2개의 레지스터를 사용하며, RS 신호는 명령을 담고 있는 레지스터(RS = LOW)와 데이터를 담고 있는 레지스터(RS = HIGH) 중 하나를 선택하기 위해 사용한다.
R/W	읽기 (R/W = HIGH) 및 쓰기 (R/W = LOW) 모드를 선택하기 위해 사용한다. 일반적으로 LCD는 데이터를 쓰기 위한 용도로만 사용하므로 R/W 신호를 GND에 연결하여 사용할 수 있다.
E	하강 에지에서 LCD 드라이버가 레지스터의 내용을 바탕으로 처리를 시작하도록 지시하기 위한 신호로 사용한다.

- 제어 명령 : 표 22-3, 22-4, 22-5 참고

1 LCD(Liquid Crystal Display)

Text LCD

Command List

명령	명령 코드										설명
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
Clear Display	0	0	0	0	0	0	0	0	0	1	공백문자(코드 0x20)로 화면을 지우고 커서를 홈 위치(주소 0번)로 이동시킨다.
Return Home	0	0	0	0	0	0	0	0	1	-	커서를 홈 위치로 이동시키고, 표시 영역이 이동된 경우 초기 위치로 이동시킨다. 화면에 출력된 내용(DDRAM의 값)은 변하지 않는다.
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	데이터 읽기 또는 쓰기 후 메모리의 증가(Increment) 또는 감소(Decrement) 방향을 지정한다. DDRAM에서 I/D = 1이면 커서를 오른쪽으로, I/D = 0이면 왼쪽으로 옮긴다. S = 1이면 I/D 값에 따라 디스플레이를 왼쪽 또는 오른쪽으로 옮기며, 이때 커서는 고정된 위치에 나타난다.
Display on/ off Control	0	0	0	0	0	0	1	D	C	B	디스플레이(D), 커서(C), 커서 깜빡임(B)의 ON/OFF를 설정한다.
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	-	-	화면에 출력된 내용의 변경 없이 커서와 화면을 이동시킨다(표 22-4).
Function Set	0	0	0	0	1	DL	N	F	-	-	데이터 비트 크기(DL = 1이면 8비트, DL = 0이면 4비트), 디스플레이 행 수(N), 폰트 크기(F)를 설정한다(표 22-5).
Set CGRAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	주소 카운터에 CGRAM 주소를 설정한다.
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	주소 카운터에 DDRAM 주소를 설정한다.
Read Busy Flag & Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	드라이버에서 현재 명령어를 실행 중인지의 여부를 나타내는 동작 중 플래그(Busy Flag) 값과 주소 카운터의 값을 읽어 온다.
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	램(DDRAM 또는 CGRAM)에 데이터를 기록한다.
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	램(DDRAM 또는 CGRAM)에서 데이터를 읽어 온다.

1 LCD(Liquid Crystal Display)

Text LCD

❖ 커서 및 화면이동

S/C	R/L	설명
0	0	커서를 왼쪽으로 옮긴다. 메모리 주소는 1 감소한다.
0	1	커서를 오른쪽으로 옮긴다. 메모리 주소는 1 증가한다.
1	0	화면을 왼쪽으로 옮긴다. 커서 역시 왼쪽으로 이동한다.
1	1	화면을 오른쪽으로 옮긴다. 커서 역시 오른쪽으로 이동한다.

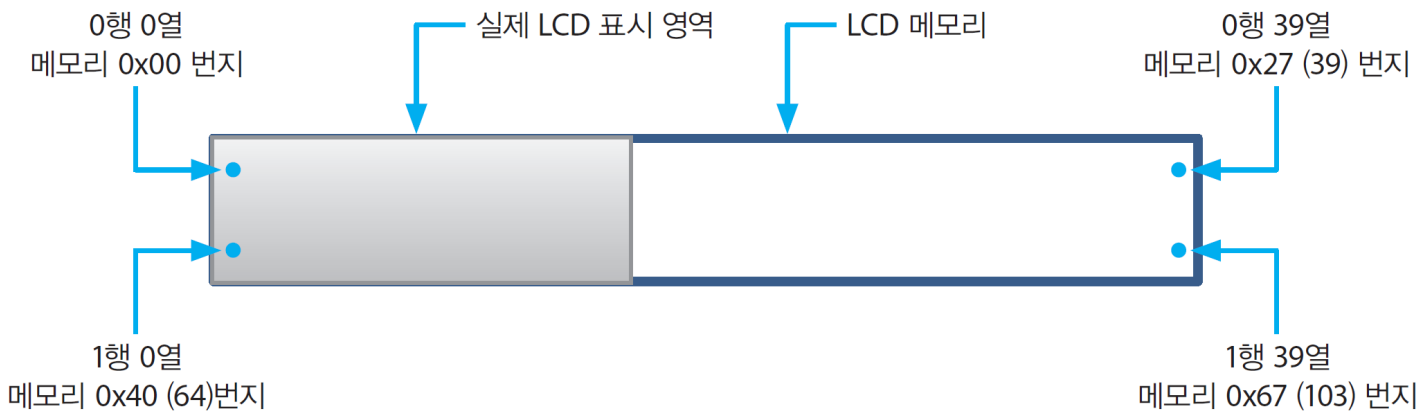
❖ 행 수와 폰트 크기

N	F	행 수	폰트 크기	비고
0	0	1	5×8	
0	1	1	5×10	
1	-	2	5×8	2행 출력에서는 5×10 크기 폰트를 사용할 수 없다.

1 LCD(Liquid Crystal Display)

Text LCD

- 메모리 영역 및 표시 영역



최대 80문자 저장이 가능하며
그 중 일부인 32문자만 표시됨

1 LCD(Liquid Crystal Display)

Text LCD

8비트 모드 – 상수 정의

```
#define RS_PIN                  0                       // RS 제어 핀의 비트 번호
#define RW_PIN                  1                       // R/W 제어 핀의 비트 번호
#define E_PIN                   2                       // E 제어 핀의 비트 번호

#define COMMAND_CLEAR_DISPLAY       0x01
#define COMMAND_8_BIT_MODE          0x38        // 8비트, 2라인, 5×8 폰트
#define COMMAND_4_BIT_MODE          0x28        // 4비트, 2라인, 5×8 폰트

#define COMMAND_DISPLAY_ON_OFF_BIT      2
#define COMMAND_CURSOR_ON_OFF_BIT       1
#define COMMAND_BLINK_ON_OFF_BIT        0
```

1 LCD(Liquid Crystal Display)

Text LCD

8비트 모드 – 제어 함수

```
void LCD_pulse_enable(void)                    // 하강 에지에서 동작
{
    PORT_CONTROL |= (1 << E_PIN);              // E를 HIGH로
    _delay_us(1);
    PORT_CONTROL &= ~(1 << E_PIN);             // E를 LOW로
    _delay_ms(1);
}

void LCD_write_data(uint8_t data)
{
    PORT_CONTROL |= (1 << RS_PIN);             // 문자 출력에서 RS는 1
    PORT_DATA = data;                          // 출력할 문자 데이터
    LCD_pulse_enable();                        // 문자 출력
    _delay_ms(2);
}

void LCD_write_command(uint8_t command)
{
    PORT_CONTROL &= ~(1 << RS_PIN);            // 명령어 실행에서 RS는 0
    PORT_DATA = command;                       // 데이터 핀에 명령어 전달
    LCD_pulse_enable();                        // 명령어 실행
    _delay_ms(2);
}

void LCD_clear(void)
{
    LCD_write_command(COMMAND_CLEAR_DISPLAY);
    _delay_ms(2);
}
```

1 LCD(Liquid Crystal Display)

Text LCD

8비트 모드 – 초기화 함수

```
void LCD_init(void)
{
    _delay_ms(50);                                  // 초기 구동 시간

    // 연결 핀을 출력으로 설정
    DDR_DATA = 0xFF;
    PORT_DATA = 0x00;
    DDR_CONTROL |= (1 << RS_PIN) | (1 << RW_PIN) | (1 << E_PIN);

    // R/W 핀으로 LOW를 출력하여 쓰기 전용으로 사용
    PORT_CONTROL &= ~(1 << RW_PIN);

    LCD_write_command(COMMAND_8_BIT_MODE);          // 8비트 모드

    // display on/off control
    // 화면 on, 커서 off, 커서 깜빡임 off
    uint8_t command = 0x08 | (1 << COMMAND_DISPLAY_ON_OFF_BIT);
    LCD_write_command(command);

    LCD_clear();                                    // 화면 지움

    // Entry Mode Set
    // 출력 후 커서를 오른쪽으로 옮김, 즉 DDRAM의 주소가 증가하며 화면 이동은 없음
    LCD_write_command(0x06);
}
```

1 LCD(Liquid Crystal Display)

Text LCD

8비트 모드 – 출력 함수

```
void LCD_write_string(char *string)
{
    uint8_t i;
    for(i = 0; string[i]; i++)                    // 종료 문자를 만날 때까지
    LCD_write_data(string[i]);                    // 문자 단위 출력
}

void LCD_goto_XY(uint8_t row, uint8_t col)
{
    col %= 16;                                    // [0 15]
    row %= 2;                                     // [0 1]

    // 첫째 라인 시작 주소는 0x00, 둘째 라인 시작 주소는 0x40
    uint8_t address = (0x40 * row) + col;
    uint8_t command = 0x80 + address;

    LCD_write_command(command);                   // 커서 이동
}
```

1 LCD(Liquid Crystal Display)

Text LCD

8비트 모드 – Main 함수

```
int main(void)
{
    LCD_init();                                     // 텍스트 LCD 초기화

    LCD_write_string("Hello World!");               // 문자열 출력

    _delay_ms(1000);                                // 1초 대기

    LCD_clear();                                    // 화면 지움

    // 화면에 보이는 영역은 기본 값으로 0~1행, 0~15열로 설정되어 있다.
    LCD_goto_XY(0, 0);                              // 0행 0열로 이동
    LCD_write_data('1');                            // 문자 단위 출력
    LCD_goto_XY(0, 5);
    LCD_write_data('2');
    LCD_goto_XY(1, 0);
    LCD_write_data('3');
    LCD_goto_XY(1, 5);
    LCD_write_data('4');

    while(1);
    return 0;
}
```

2 ADC

2 ADC

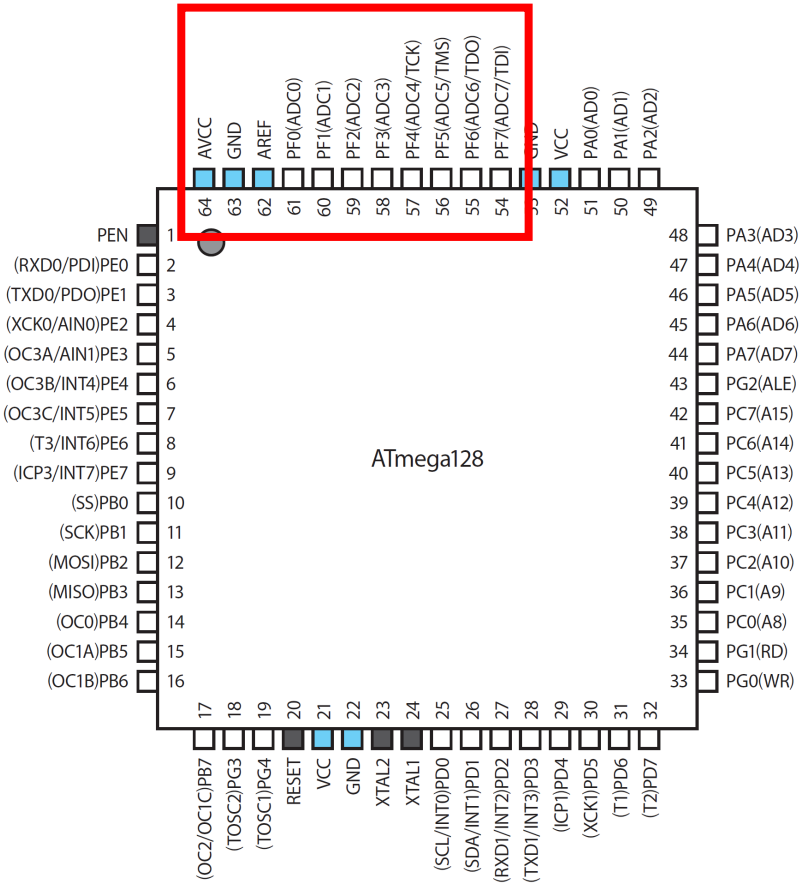
Analog-Digital Converter

❖ 아날로그-디지털 변환의 필요성

- 주변의 모든 데이터는 아날로그 데이터
- 마이크로 컨트롤러가 처리할 수 있는 데이터는 디지털 데이터

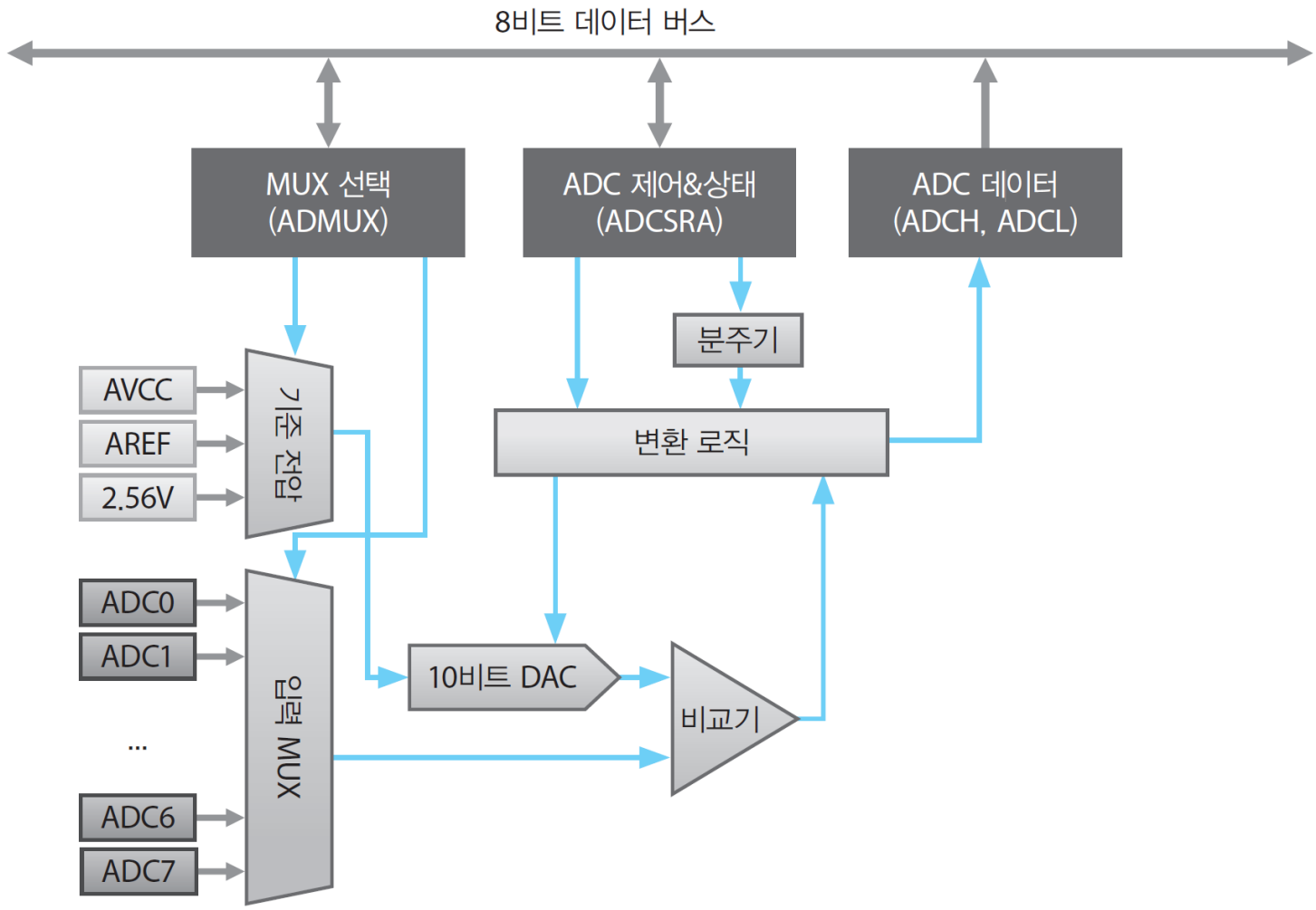
❖ ATmega128의 아날로그-디지털 변환기(ADC)

- 10비트 해상도 : 아날로그 전압을 0~1,023 사이 디지털 값으로 변환
- 1,023으로 변환되는 기준 전압은 AVCC, AREF, 내부 2.56V 중 선택 사용
- 8개 채널 제공 : 포트 F, MUX를 통해 연결되고 한 번에 하나씩 만 사용 가능



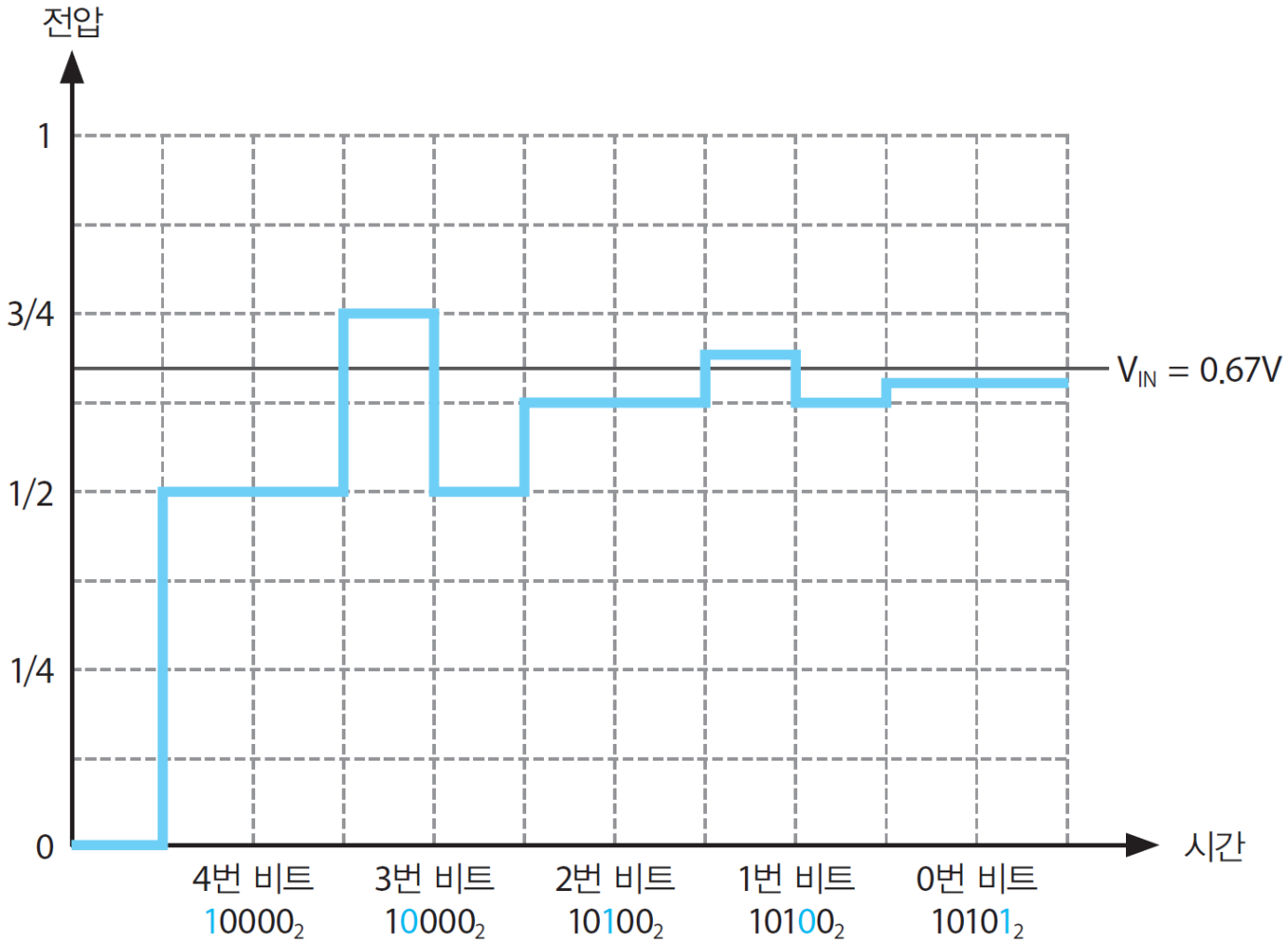
2 ADC

Analog-Digital Converter



2 ADC

축차 비교 방식


전압
1
3/4
1/2
1/4
0
V_{IN} = 0.67V
시간
4번 비트
10000_2
3번 비트
10000_2
2번 비트
10100_2
1번 비트
10100_2
0번 비트
10101_2

2 ADC

ADC Option

❖ 단일 입력과 차동 입력

- 단일 입력 : 1개의 입력과 GND를 사용
- 차동 입력 : 2개의 입력 차이와 GND를 사용

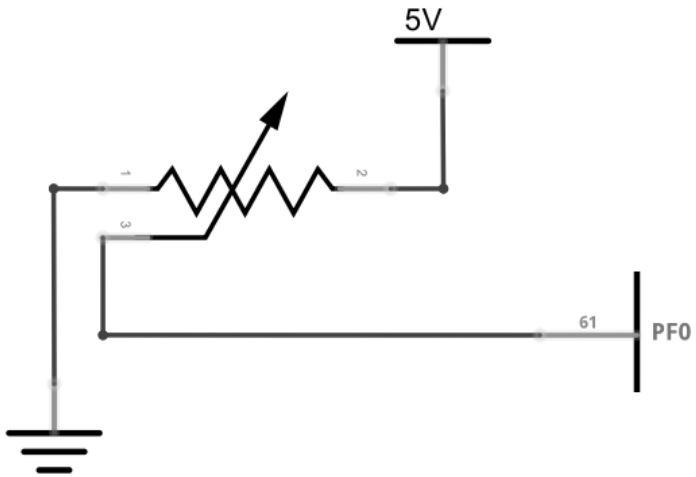
❖ 이득(gain)

- 입력되는 아날로그 신호 크기가 작은 경우 증폭 기능 사용 가능
- 10배, 200배 신호 증폭 가능

예제 코드 - 1

❖ 가변저항 읽기

- ADC 0번 채널인 PF0에 가변저항 연결
- 0~5V 아날로그 전압을 0~1,023 디지털 값으로 변환하여 읽음



2 ADC

예제 코드 - 1

- 가변저항 읽기

```
void ADC_init(unsigned char channel)
{
    ADMUX |= (1 << REFS0);                  // AVCC를 기준 전압으로 선택

    ADCSRA |= 0x07;                         // 분주비 설정

    ADCSRA |= (1 << ADEN);                  // ADC 활성화
    ADCSRA |= (1 << ADFR);                  // 프리러닝 모드

    ADMUX = ((ADMUX & 0xE0) | channel);     // 채널 선택
    ADCSRA |= (1 << ADSC);                  // 변환 시작
}
```

```
int read_ADC(void)
{
    while(!(ADCSRA & (1 << ADIF)));         // 변환 종료 대기

    return ADC;                             // 10비트 값을 반환
}
```

2 ADC

ADC (= ADCH + ADCL) **레지스터**

❖ ATmega128의 레지스터는 8비트, ADC 변환 값은 10비트

- 2개의 8비트 레지스터를 사용하여 ADC 변환 값 저장
- ADCH : 상위 2비트 또는 8비트
- ADCH : 하위 8비트 또는 2비트
- 6비트는 사용하지 않음
- ADMUX 레지스터의 ADLAR 비트로 정렬 방식 지정

비트	15	14	13	12	11	10	9	8
ADCH	-	-	-	-	-	-	ADC9	ADC8
ADCL	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0
비트	7	6	5	4	3	2	1	0

(a) 오른쪽 정렬

비트	15	14	13	12	11	10	9	8
ADCH	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2
ADCL	ADC1	ADC0	-	-	-	-	-	-
비트	7	6	5	4	3	2	1	0

(b) 왼쪽 정렬

2 ADC

ADMUX 레지스터

❖ AD 변환 기준 전압과 입력 채널 선택

비트	7	6	5	4	3	2	1	0
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0
읽기/쓰기	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
초깃값	0	0	0	0	0	0	0	0

- REFSn : 기준 전압 설정

REFS1	REFS0	설명
0	0	외부 AREF 핀 입력을 기준 전압으로 사용한다.
0	1	외부 AVCC 핀 입력을 기준 전압으로 사용한다.
1	0	-
1	1	내부 2.56V를 기준 전압으로 사용한다.

- ADLAR : 정렬 방식 설정
- MUXn : 채널 선택
  - 차동 입력 및 이득 설정
  - 단일 입력은 000002 ~ 001112까지 8개

2 ADC

ADMUX 레지스터

- MUXn 비트 설정에 따른 입력 채널 및 이득

MUX[4:0]	단일 입력	차동 입력 (+)	차동 입력 (-)	이득
00000	ADC0			
00001	ADC1			
00010	ADC2			
00011	ADC3			
00100	ADC4			
00101	ADC5			
00110	ADC6			
00111	ADC7			
01000		ADC0	ADC0	10x
01001		ADC1	ADC0	10x
01010		ADC0	ADC0	200x
01011		ADC1	ADC0	200x
01100		ADC2	ADC2	10x
01101		ADC3	ADC2	10x
01110		ADC2	ADC2	200x
01111		ADC3	ADC2	200x
10000		ADC0	ADC1	1x
10001		ADC1	ADC1	1x
10010		ADC2	ADC1	1x
10011		ADC3	ADC1	1x
10100		ADC4	ADC1	1x
10101		ADC5	ADC1	1x
10110		ADC6	ADC1	1x
10111		ADC7	ADC1	1x
11000		ADC0	ADC2	1x
11001		ADC1	ADC2	1x
11010		ADC2	ADC2	1x
11011		ADC3	ADC2	1x
11100		ADC4	ADC2	1x
11101		ADC5	ADC2	1x
11110	1.23V(V_{BG})			
11111	0V(GND)			

ADCSRA 레지스터

❖ AD 변환 상태 표시 및 AD 변환 제어

비트	7	6	5	4	3	2	1	0
	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0
읽기/쓰기	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
초깃값	0	0	0	0	0	0	0	0

비트	이름	설명
7	ADEN	ADC Enable: ADC를 활성화시킨다.
6	ADSC	ADC Start Conversion: AD 변환을 시작한다.
5	ADFR	ADC Free Running Selection: 단일 변환 모드 또는 프리러닝 모드를 설정한다.
4	ADIF	ADC Interrupt Flag: AD 변환 종료 시 1로 설정된다.
3	ADIE	ADC Interrupt Enable: AD 변환이 종료되면 인터럽트 발생을 허용한다.
2	ADPS2	ADC Prescaler Select: ADC를 위한 분주율을 설정한다.
1	ADPS1	
0	ADPS0	

2 ADC

ADCSRA 레지스터

비트	7	6	5	4	3	2	1	0
	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0
읽기/쓰기	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
초깃값	0	0	0	0	0	0	0	0

비트	이름	설명
7	ADEN	ADC Enable: ADC를 활성화시킨다.
6	ADSC	ADC Start Conversion: AD 변환을 시작한다.
5	ADFR	ADC Free Running Selection: 단일 변환 모드 또는 프리러닝 모드를 설정한다.
4	ADIF	ADC Interrupt Flag: AD 변환 종료 시 1로 설정된다.
3	ADIE	ADC Interrupt Enable: AD 변환이 종료되면 인터럽트 발생을 허용한다.
2	ADPS2	ADC Prescaler Select: ADC를 위한 분주율을 설정한다.
1	ADPS1	
0	ADPS0	

2 ADC

ADCSRA 레지스터

- ADEN : ADC 활성화, 디폴트값은 비활성화 상태

- ADSC : 변환 시작
  - ✓ 단일 변환 모드 : AD 변환 시작
  - ✓ 프리러닝 모드 : 첫 번째 변환 시작

비트	7	6	5	4	3	2	1	0
	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0
읽기/쓰기	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W
초깃값	0	0	0	0	0	0	0	0

- ADFR : 단일 변환 모드 or 프리러닝 모드
  - ✓ 단일 변환 모드 : AD 변환 시작 비트가 세트되면 1번 AD 변환 후 종료
  - ✓ 프리러닝 모드 : AD 변환이 시작된 이후, 이전 AD 변환이 끝나면 다음 AD 변환을 자동으로 시작

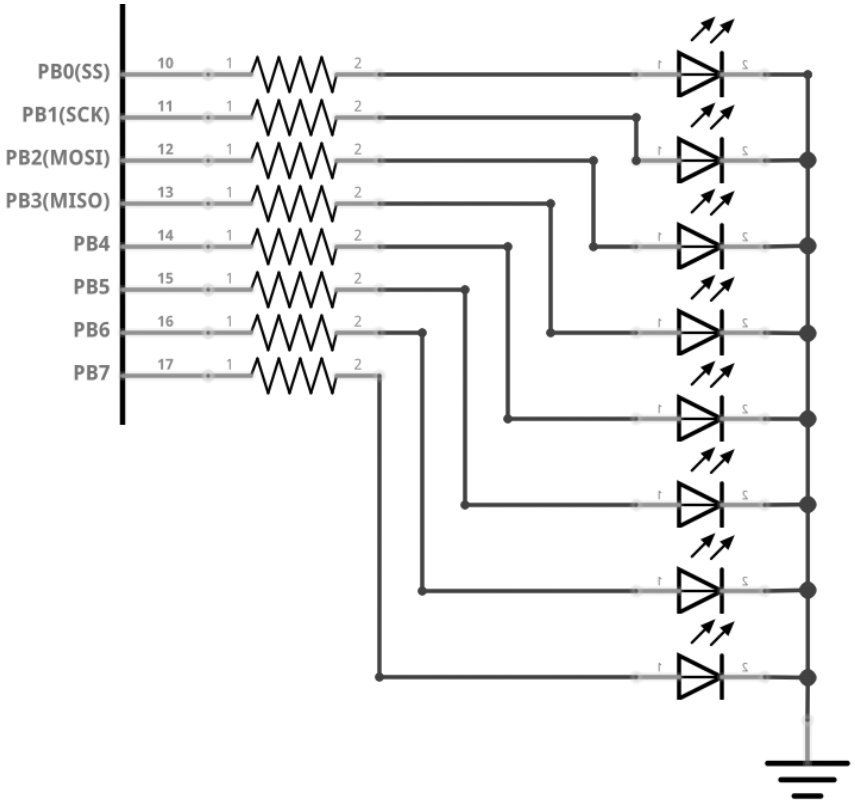
- ADIE : AD 변환이 종료될 때 인터럽트 발생 허용

- ADPSn : AD 변환을 위한 분주비 설정
  - ✓ AD 변환을 위한 주파수는 50~200KHz 추천

ADPS2	ADPS1	ADPS0	분주율
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

예제코드 -2

- 가변저항으로 LED 제어



예제코드 -2

- 가변저항으로 LED 제어

```
while(1)
{
    read = read_ADC();                          // 가변저항 값 읽기

    uint8_t pattern = 0;                        // LED 제어값
    int LED_count = (read >> 7) + 1;            // 켜질 LED의 개수

    for(int i = 0; i < LED_count; i++){         // LED 제어값 생성
        pattern |= (0x01 << i);
    }

    PORTB = pattern;                            // LED 켜기



    _delay_ms(1000);
}
```


Q & A