



C – HW5

임베디드스쿨1기

Lv1과정

2020. 08. 29

강경수

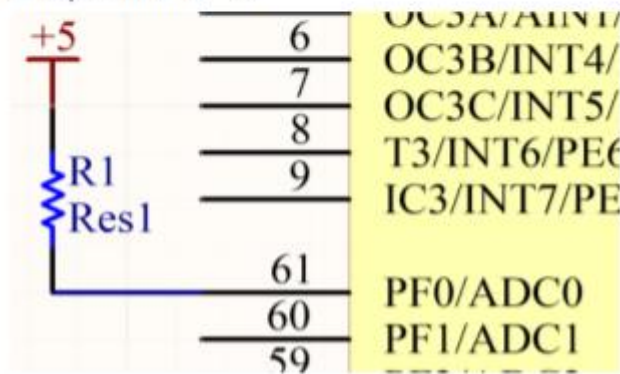
1. PULLUP & PULL DOWN

■ Pullup 저항 & Pulldown 저항

1. Pullup 저항, Pulldown 저항의 의미

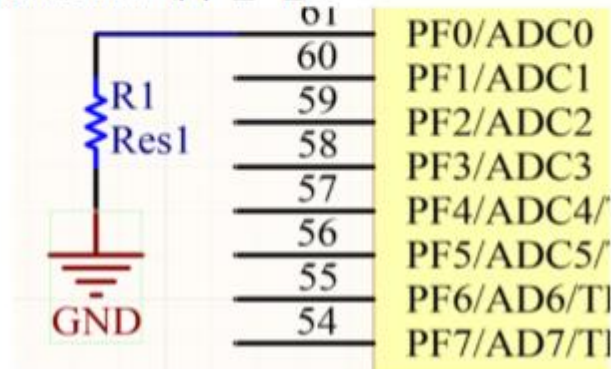
- 디지털레벨 1or0을 위하여 디지털IC PIN외부에 달아주는 저항

2. Pullup 해주는 법



- HIGH로 띄워야 하는 핀에 저항과 Vcc를 연결
- 저항값은 포트 내부의 임피던스의 값을 고려함
- 풀업저항이 너무 작을 경우 소비전류가 커짐
- 너무 클 경우에는 포트 내부의 임피던스 보다 값이 커져 HIGH신호를 인식하지 못함.
- 풀업저항이 큰 경우 RC충방전 시간이 길어져 응답 속도가 느려짐
- 일반적으로 MCU는 자체적으로 풀업저항기능 지원함

3. Pulldown 해주는 법



- PORT와 GND를 VCC를 통해 연결
- 저항을 두는 이유는 PORT가 HIGH가 됐을때 SHORT로 큰전류가 흐를을 방지 하기위함.

1. PULLUP & PULL DOWN

4. MCU 자체 지원 내부 풀업 저항

R_{PU}	I/O Pin Pull-up Resistor
----------	--------------------------

- 자체적으로 내부 풀업저항을 USE/UNUSED 할 수 있는 레지스터를 갖고 있음.

5. 실제 풀업저항 선정 방법

- 내부 임피던스의 값보다 풀업 저항이 클 경우,
값을 읽을때 외부 풀업저항에 많은 전압이 할당되어 제대로 측정 불가
따라서 내부임피던스 1/10 이하로 설계(ATMEGA328 DATA SHEET 내에선 확인하지 못하였음)

6. 실제 풀다운 저항 선정 방법

DC Current per I/O Pin 40.0mA

DC Current V_{CC} and GND Pins. 200.0mA

I/O PORT PULLDOWN 저항 40mA 넘지않게 설계

- 3.3V 기준 대략 10옴 이상이어야 MAX값 이내. 실제 설계시 이보다 훨씬 큰 값사용
- MCU 총 출력전류가 제한되어 있는 경우가 많다. 꼭 DATASHEET 확인할것.

1. PULLUP & PULL DOWN

7. PULLUP PULLDOWN 저항이 없을경우 생겨나는 현상

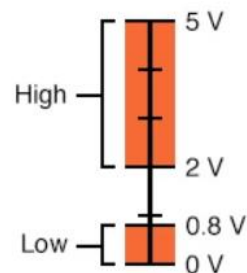
- 해당 포트는 HI-Z(하이 임피던스 즉, 공기로 인한 높은 임피던스를 갖는 개방) 상태가 되어, 얼마만큼의 전압에 포트내에서 읽혀지는지 알 수 없음.
이를 **floating** 상태라 함.

8. HI-Z상태란?

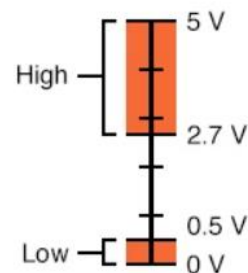
- 하이 임피던스 즉 높은 임피던스 상태를 이야기함.
일반적으로 공기중에 Digital pin이 개방되어 있는 상태를 이야기 한다.
이 상태에서는 Digital pin이 1(High) 인지 0인지 보증 할 수 없다.

9. Digital pin의 High Low 기준

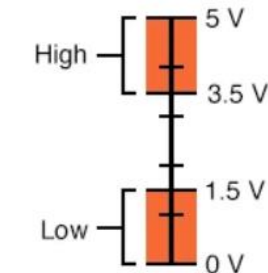
Acceptable TTL Gate
Input Signal Levels



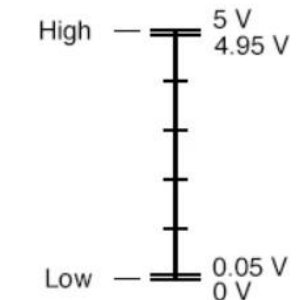
Acceptable TTL Gate
Output Signal Levels



Acceptable CMOS Gate
Input Signal Levels



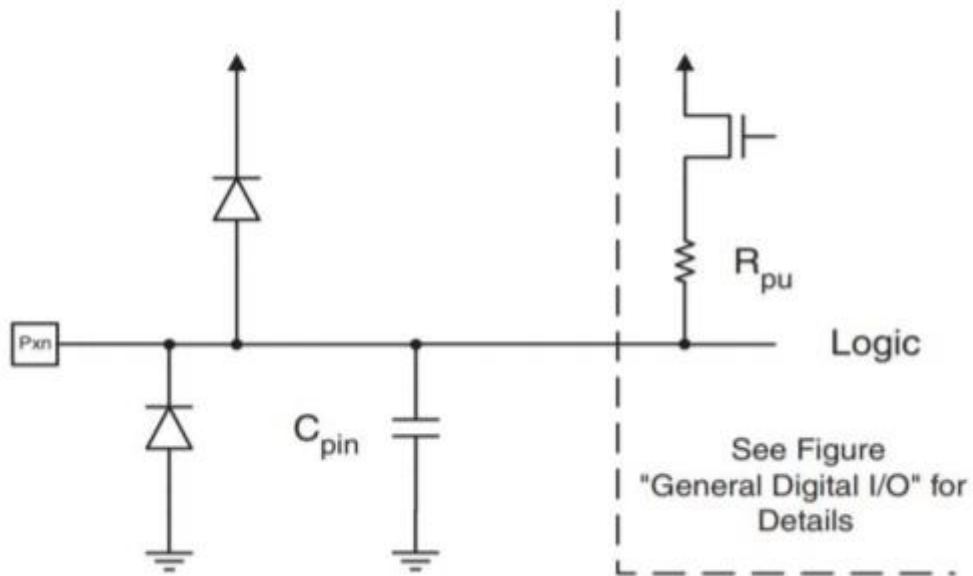
Acceptable CMOS Gate
Output Signal Levels



2. ATMEGA328 DIGITAL I/O

■ ATMEGA 328 DIGITAL I/O

1. I/O PIN 등가 회로



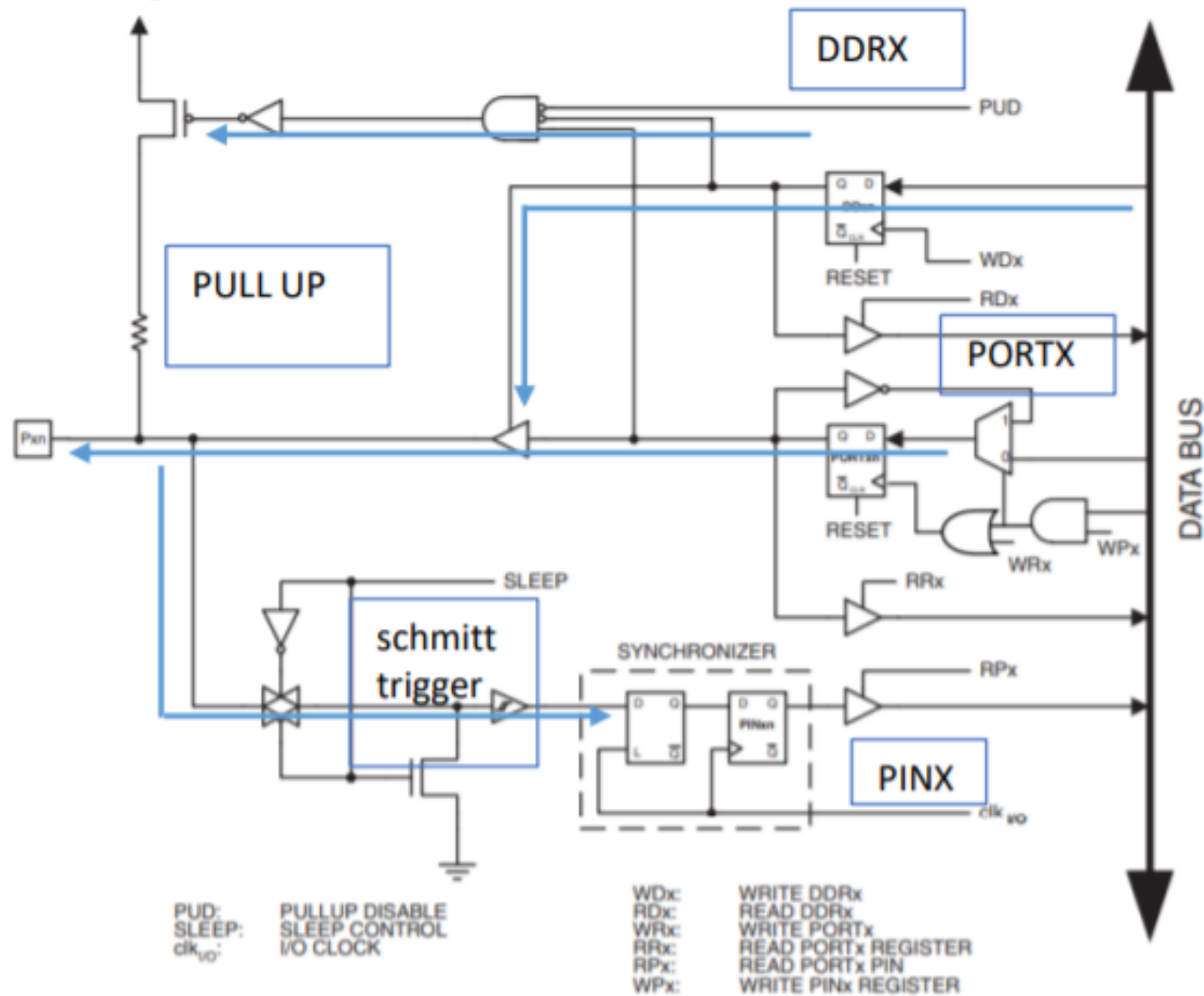
- 2개의 다이오드
→ 정격이상 혹은 이하의 전압
인가될시 port보호하기 위함.

- C_{pin}
→ parastic C 기생

- R_{pu} 게이트에 신호를 주어
풀업저항 on/off

2. ATMEGA328 DIGITAL I/O

Figure 14-2. General Digital I/O⁽¹⁾



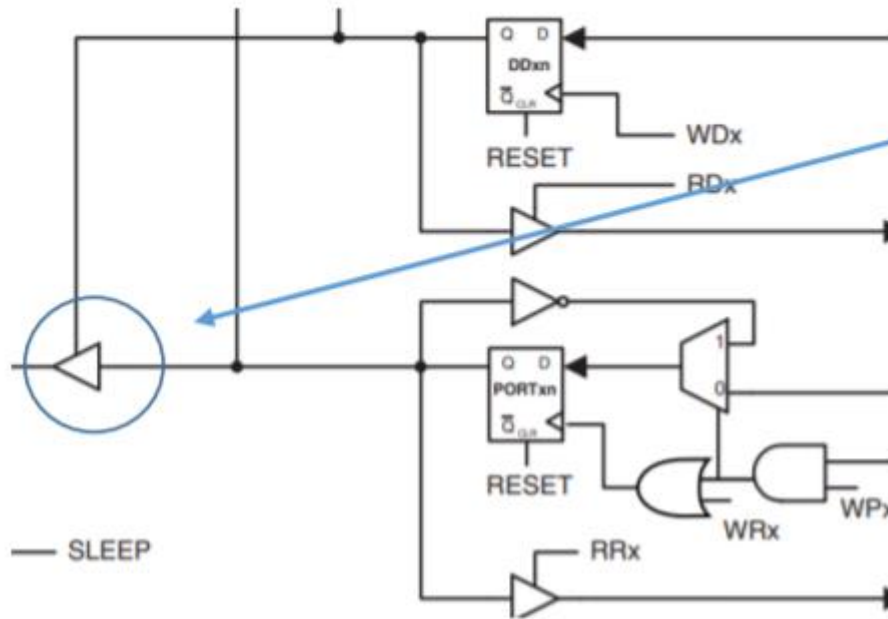
2. ATMEGA328 DIGITAL I/O

2. 핀 구성

- 모든 핀은 레지스터 DDxn, PORTxn, PINxn으로 구성되어 있음.
- DDxn 레지스터는 핀의 방향을 결정한다 .
- DDxn이 1이면 Pxn은 output으로, 0이면 Pxn은 input으로 된다.
- PORTxn이 1이고 input(즉 DDxn이 0)이면 내부 풀업저항이 활성화 된다.
- DDxn이 1이고 PORTxn이 1이면, high신호를 출력한다.
- DDxn이 1이고 PORTxn이 0이면, low신호를 출력한다.

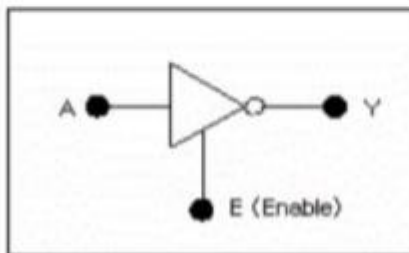
2. ATMEGA328 DIGITAL I/O

3. Tri-state buffer



- 3 State Buffer를 통해서 Pxn이
컨트롤됨을 확인 할 수 있음
DDRxn → CE
PORTXN → 출력

- 3 State Buffer 진리표



E(Enable)	A	Y
High	Low	High
	High	Low
Low	Low	High-Impedance
	High	

2. ATMEGA328 DIGITAL I/O

4. Toggling the pin

- PIN_{xn}은 PORT_{xn}을 토글할 수 있다. DDR_{xn}의 값과 상관없음.

5. Switching Between Input and Output

- DD_{xn}, PORT_{xn} = 0b00 → 0b11 전환시 0b10 or 0b01로 전환된 이후 전환되어야 함.
- 위와같은 순서를 따르지 않으면 MCUCR의 PUD비트가 모든 포트의 풀업저항을 비활성화 함.
- 마찬가지로 DD_{xn}, PORT_{xn} = 0b00 → 0b10 전환시 중간단계로 0b00 or 0b11을 사용해야함.

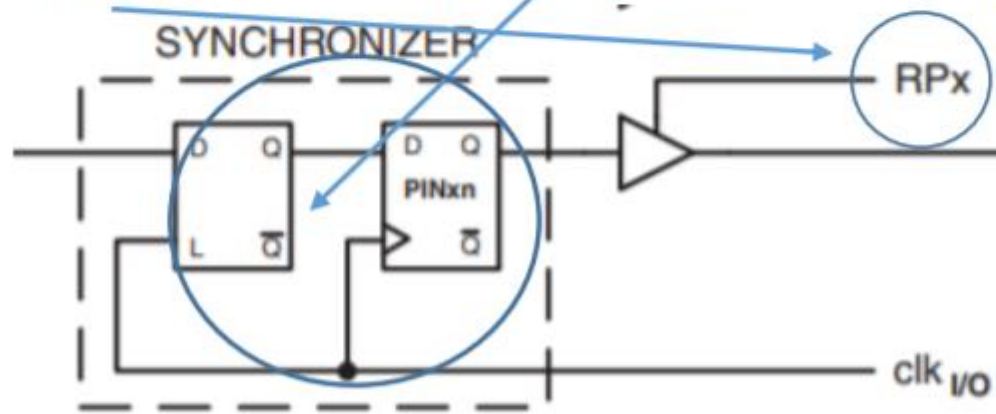
Table 14-1. Port Pin Configurations

DD _{xn}	PORT _{xn}	PUD (in MCUCR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	P _{xn} will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

2. ATMEGA328 DIGITAL I/O

6. Reading the Pin Value

- 데이터 방향비트 DDxn과 별개로 포트핀은 PINxn레지스터 비트를 통해 읽을 수 있음.
- **PINxn레지스터와 비트와 선행래치**가 싱크로나이징(동기화)함.

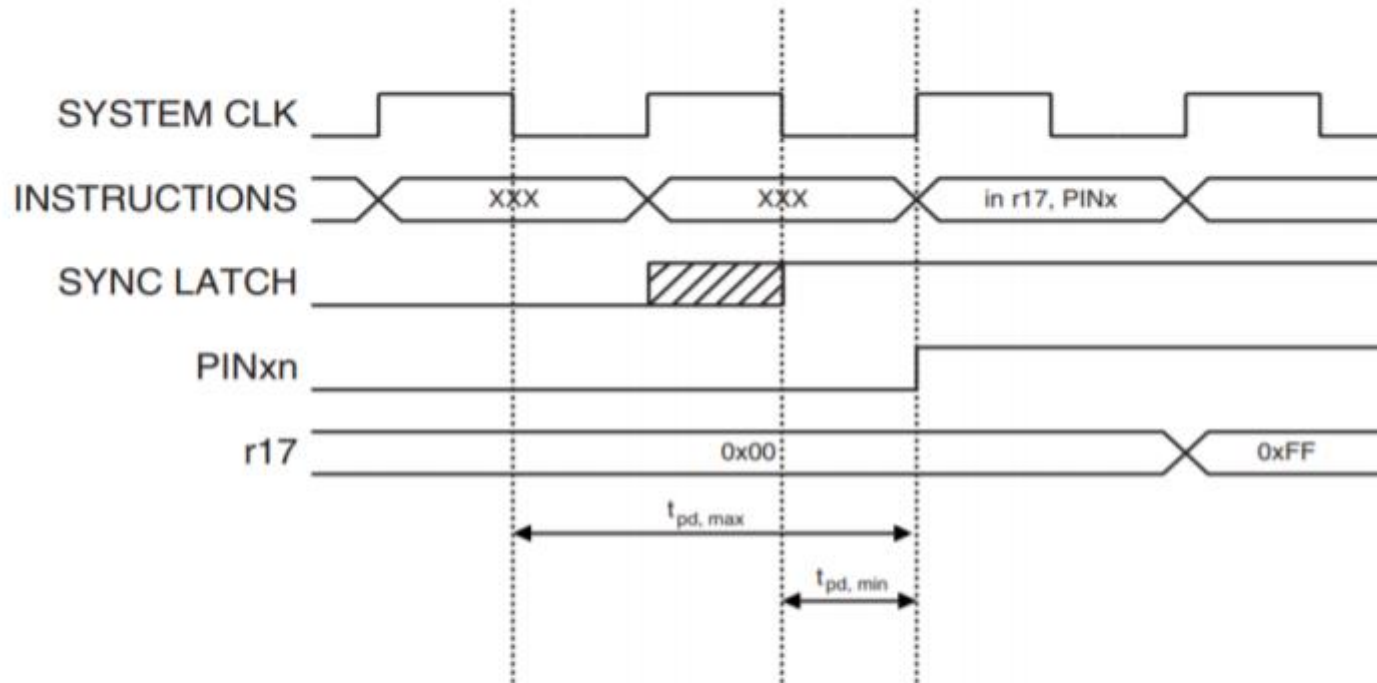


- 이유는 내부클럭 근처에서 핀이 값을 바꿀때 metasability를 피하기 위함.
(metasability = 1인지 0인지 모르는 상태)
이로 인하여 metasability를 피할 수 있지만, 지연시간이 발생한다.

2. ATMEGA328 DIGITAL I/O

7. Reading an Externally Applied Pin value

Synchronization when Reading an Externally Applied Pin value

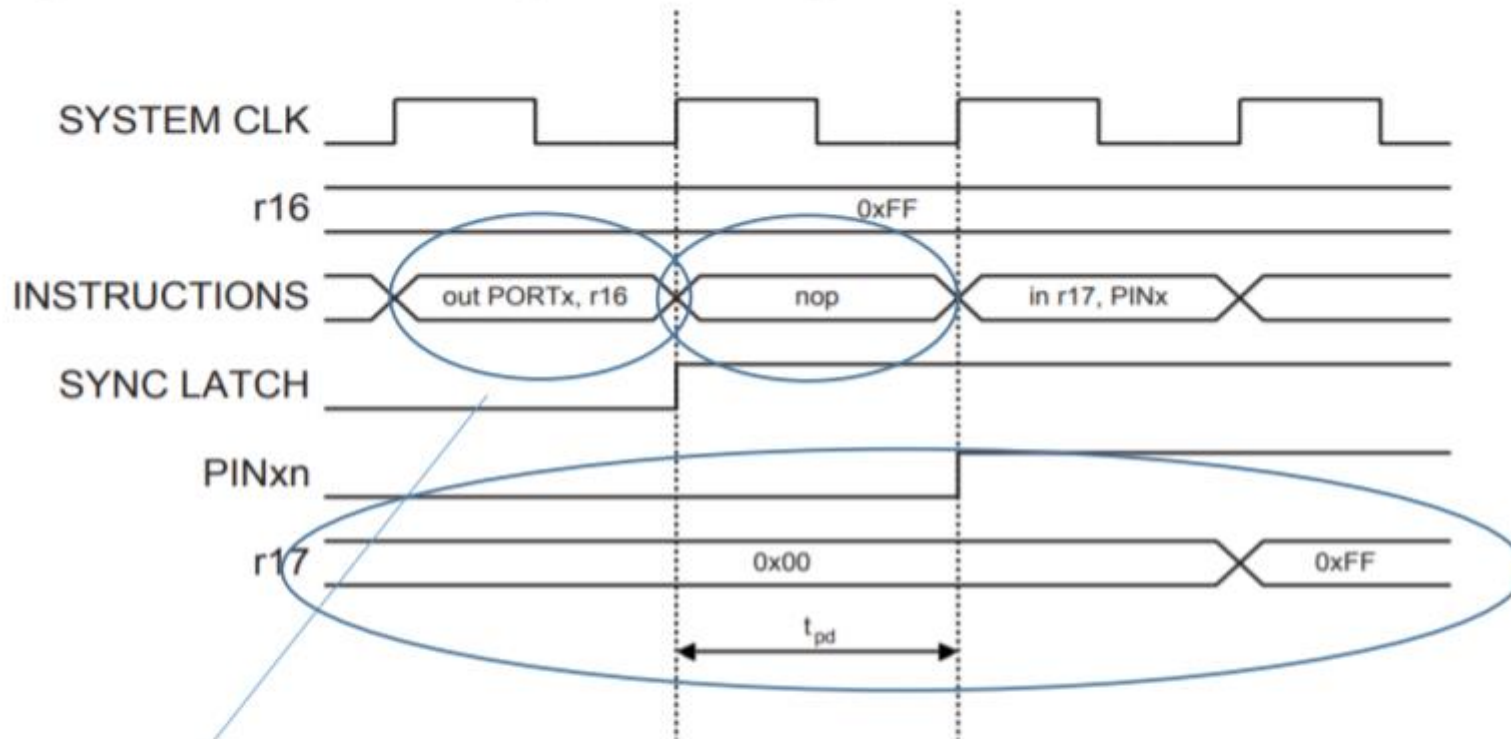


- 래치신호는 클록 하강신호일때 래치된다.
- 싱크로나이징 회로로 인하여 래치 신호 이후 PINxn값이 변한다.
- 화살표에서 보듯이 PINxn 변화는 2/1 그리고 1+1/2 시스템클록만큼 늦어진다.

2. ATMEGA328 DIGITAL I/O

8. Reading a Software Assigned Pin Value

Synchronization when Reading a Software Assigned Pin Value



- 소프트웨어적으로 할당된 Pin값을 읽을때에는 반드시 `nop` 명령어를 삽입해야 한다.
- `out` 명령어 이후 래치신호는 클럭 상승에지일때 변화되며 이때 싱크로나이저를 통한 지연은 1system clock만큼 지연된다.
- 16Mhz 기준 주기는 62.5ns이므로 외부 핀변화 감지는 약 31.25ns~ 87.5ns 지연 소프트웨어로 인한 핀변화 감지는 62.5ns만큼 지연된다.

2. ATMEGA328 DIGITAL I/O

9. Digital Input Enable and Sleep Mode

- 디지털입력신호는 Schmitt trigger buffer 입력측에서 접지로 고정될 수 있다.
- MCU슬립컨트롤러가 SLEEPMODE들을 설정하여 floating상태의 신호, VCC/2레벨 입력신호를 무시하여 저전력 모드를 구현할 수 있다.
- 외부 인터럽트 신호에 의하여 SLEEP MODE는 갱신됨 (아마 일반모드로?)
- 외부 인터럽트가 활성화 되지 않으면 SLEEP MODE는 계속 활성화 상태

10. Unconnected Pin

- 모든 포트는 1 혹은 0으로 정의 되는게 좋다.
- 플로팅된 input pin의 경우 sleep mode때 비 활성화 된다고 하더라도 Reset, Active,idle모드때 전류소모를 야기 할 수 있으므로 피해야 한다.
- 가장 추천하는 방법은 사용하지 않는 핀에 대하여 내부 풀업저항 사용 할것.
- Reset시에 내부 풀업저항이 잠시 비활성화 되는데 이때의 전력소모도 고려하여야 한다면 그냥 외부 풀업저항 달아서 사용할 것.

3. ATMEGA328 Pin Configurations

■ ATMEGA 328 Pin Configurations

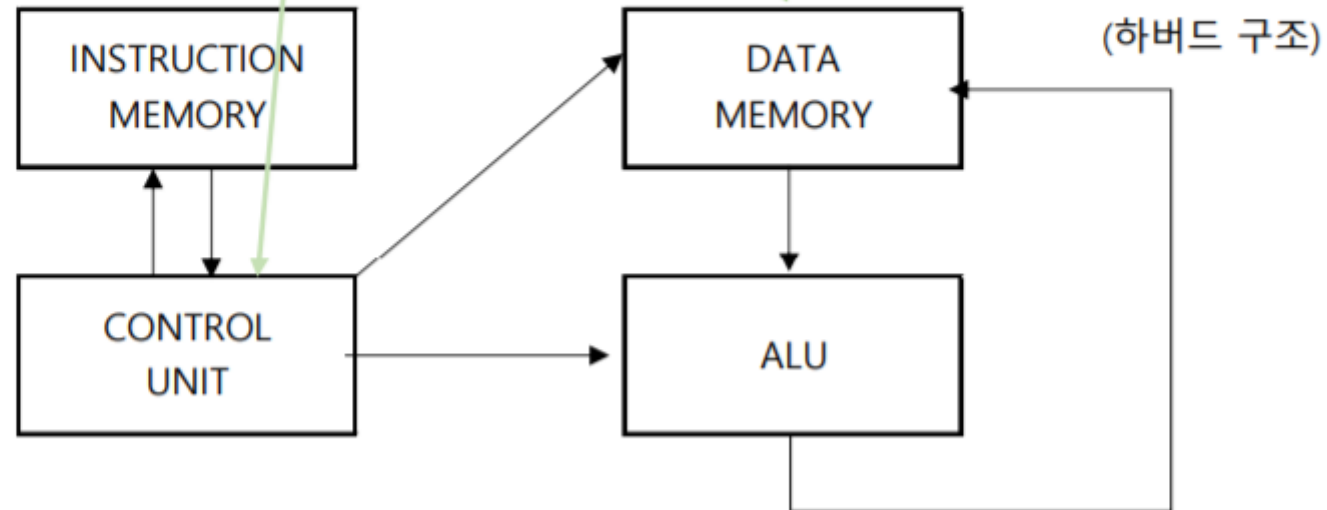
1. Vcc : 전원공급
2. GND : 기준전위
3. PORTB : DIGITAL I/O / PB6 PB7 외부 Oscillator로 사용 가능 / 내부 RC Oscialtor 사용시 PB7..6은 TOSC2..1으로서 동기화 타이머로서 사용됨.
4. PORTC : Digital I/O
5. PC6/RESETW : DIGITAL I/O / RSTDISBL Fuse 세팅에 따라서 reset input으로 사용 가능
일정 시간의 low pulse 인가시 reset이 발생한다. Clock이 발생하고 있지 않더라도.
6. PORTD : Digital I/O 다른 포트에 비해 높은 전류를 drive 및 sink 할 수 있음.
7. AVcc : A/D컨버터를 위한 전원공급 핀. PC3:0 그리고 ADC7:6은 외부 VCC에 연결.
ADC를 사용하지 않는경우에도. ADC를 사용하는 경우에 외부 LOW PASS FILTER를
거치도록 설계.
8. AREF : 아날로그 레퍼런스 전압
9. ADC7:6 10bit ADC INPUT

■ AVR CPU CORE

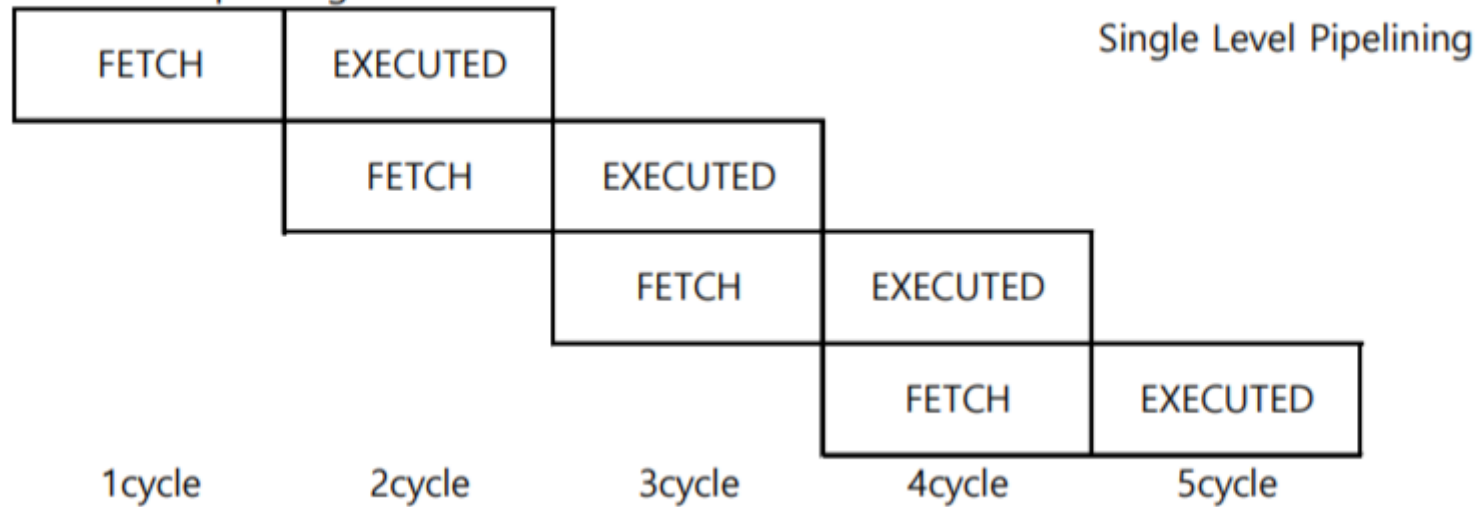


4. AVR CPU CORE

2. Harvard Architecutre



3. AVR CORE Pipelining



4. AVR CPU CORE

4. Register

- 32X8bit의 매우 빠른 범용목적의 레지스터를 갖고 있다.(레지스터 접근 시간 1cycle)
- 32개의 레지스터중 6개를 2개씩 사용하여 16bit의 X,Y,Z 레지스터로 사용

5. ALU

- ALU는 상수와 레지스터간 혹은 레지스터간의 산술 및 논리 연산을 지원함.
- 레지스터 단 하나 단독 동작은 ALU내부에서 해결
- 산술 연산후 상태 레지스터를 업데이트하며 연산 결과에 대한 정보를 반영함.

6. Program Flash Memory

- 플래쉬 프로그램 메모리는 book section 과 application code section으로 나뉘어져 있음.
즉 AVR시리즈 사용시 작성한 코드는 application code section에 저장됨.

5. Arduino

■ Arduino 실습

```
int trig = 6;
int echo = 5;

void setup() {
  pinMode(3, OUTPUT);
  Serial.begin(9600);
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
}

void loop() {

  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);

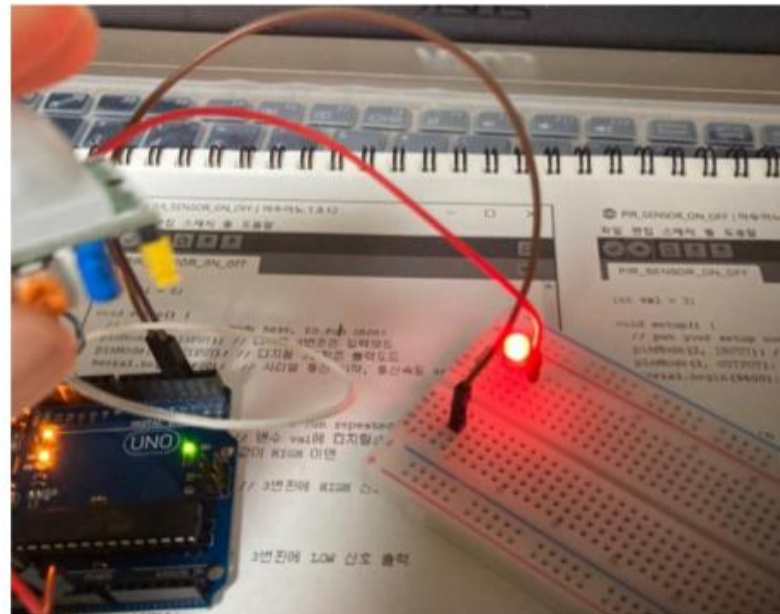
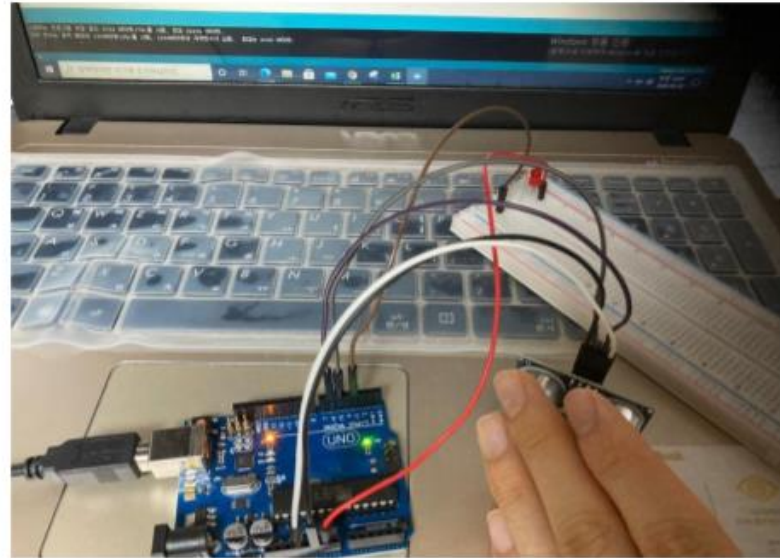
  int distance = pulseIn(echo, HIGH) * 340/2/10000;

  Serial.print(distance);
  Serial.println("cm");
  delay(100);

  if(distance < 20)
  {
    digitalWrite(3, HIGH);
  }
  else
  {
    digitalWrite(3, LOW);
  }
}
```

26cm
30cm
31cm
35cm
33cm
33cm
32cm
32cm
32cm
27cm
33cm
33cm
31cm
31cm

☒ 자동 스크롤 ☐ 타임스탬프 표시



6. Quick sort

■ Quick_Sort

1. 아래와같은 배열이 있다고 가정



9	331	90	11	13
---	-----	----	----	----

2. 이를 가장 빠르게 오름차순 혹은 내림차순으로 정렬하는 방법은? → Quick Sort

1) 위 표에서 아무 위치에 Pivot값 할당.(박샘은 맨 오른쪽으로 진행)

9	331	90	11	13 (Pivot)
---	-----	----	----	---------------

2) right값 pivot보다 작을때까지 search / left값 pivot 보다 클때까지 search

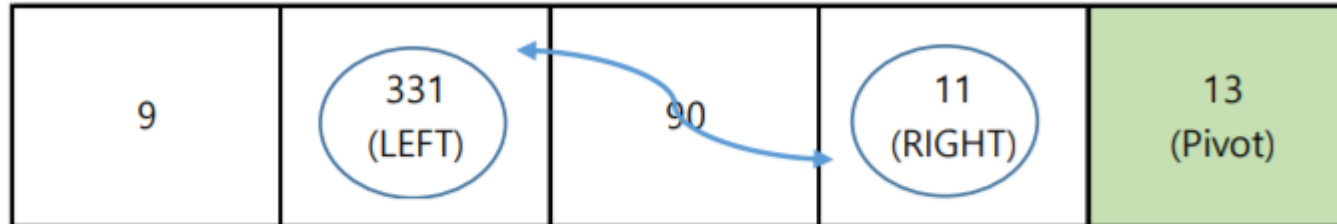
9 (LEFT)	331	90	11 (RIGHT)	13 (Pivot)
				

즉 RIGHT 11은 PIVOT 13보다 작으므로 RIGHT SEARCH STOP

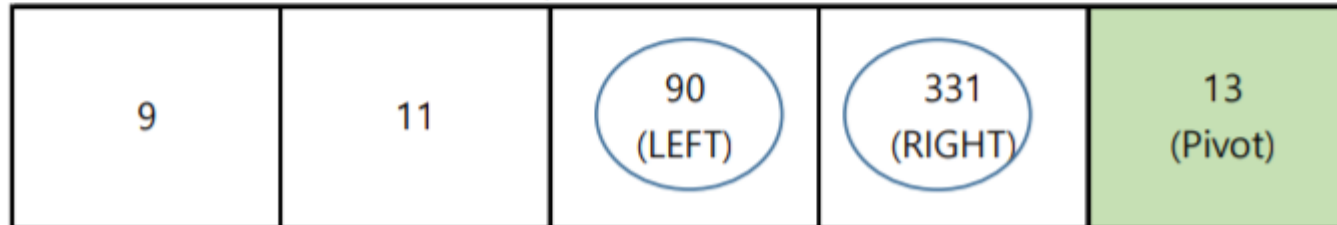
LEFT 9는 PIVOT보다 작으므로 LEFT SEARCH 331까지 진행

6. Quick sort

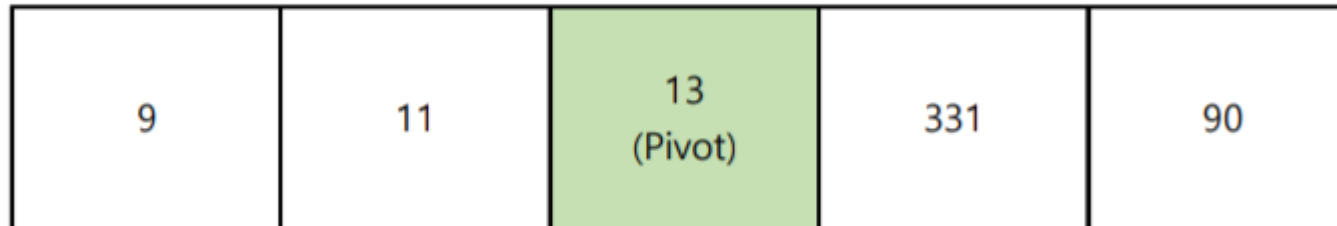
3) 이때 331과 11이 각각 PIVOT보다 크고 작음을 만족하므로 SWAP



4) SWAP 이후 LEFT와 RIGHT는 각각 ++, --하게 되므로 90에서 LEFT와 RIGHT는 서로 겹침



5) 위와같이 LEFT와 RIGHT가 하나 차이 나게 되면 PIVOT과 LEFT를 SWAP해줌



6) 이후 양 옆의 값들에 대해서도 재귀함수로 quick_sort 진행