

# GPIO

(General Purpose Inputs Outputs)

# GPIO (General Purpose Input Output)

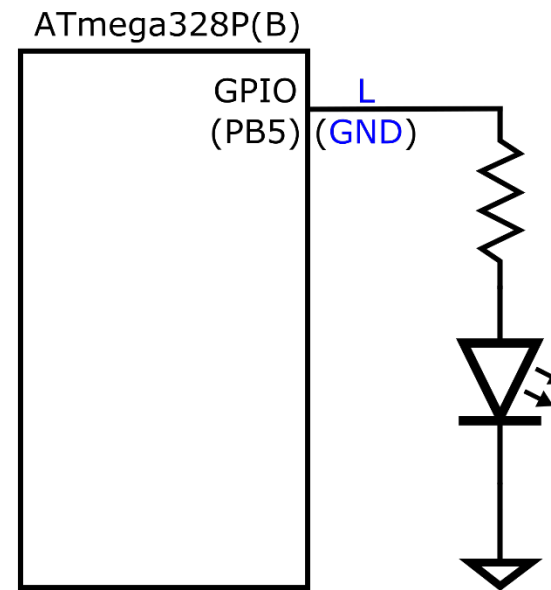
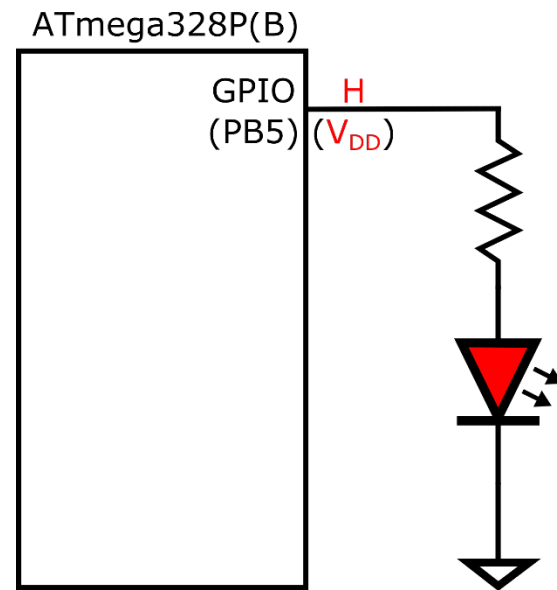
---

- General Purpose Input Output

- Is used for input or output **binary** data from/to a device whose communication protocol is not standard.
- Each port pin can be individually selectable for input or output mode.
- All port pins have individually selectable pull-up resistors with a supply-voltage invariant resistance.

# GPIO (General Purpose Input Output)

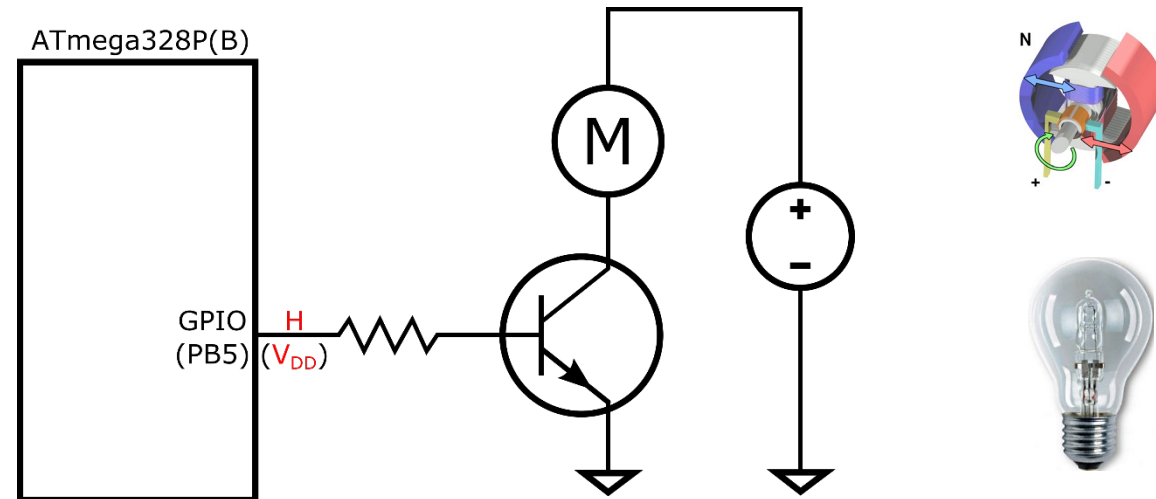
- General Purpose **Output** is used
  - to control a device whose function is controlled by binary value, i.e. '0' or '1'.
  - Examples: small DC power devices – Turn on/off **LEDs**



# GPIO (General Purpose Input Output)

- General Purpose **Output** Example

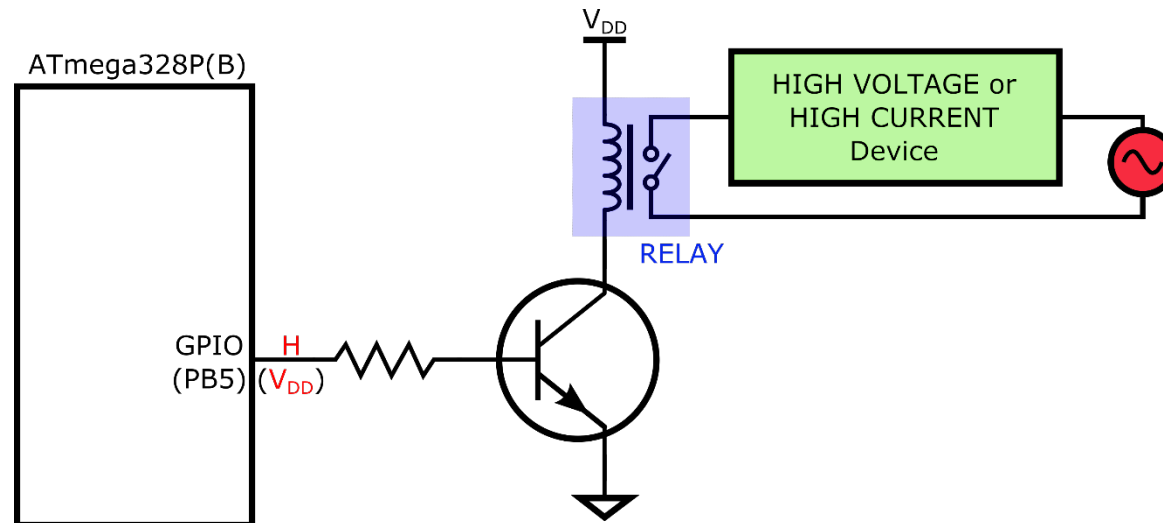
- to control a device whose function is controlled by binary value, i.e. '0' or '1'.
- Example: medium or high DC power devices - Turn on/off **motors or lamps**



# GPIO (General Purpose Input Output)

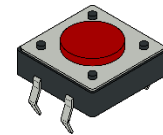
- General Purpose **Output** Example

- to control a device whose function is controlled by binary value, i.e. '0' or '1'.
- Example: low to high DC or AC power devices – Turn on/off **relays (heater, aircon)**

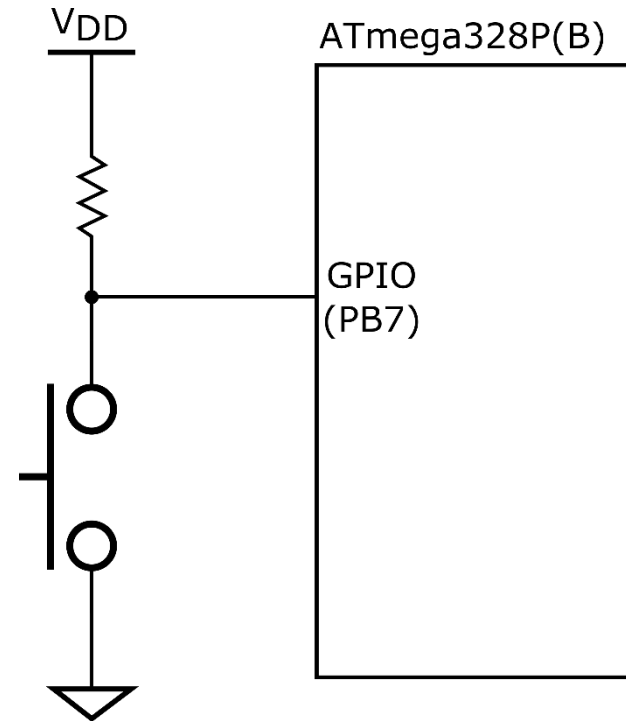


# GPIO (General Purpose Input Output)

- General Purpose **Input** is used
  - to accept signal from a device whose output is binary value, i.e. '0' or '1'.
  - Examples: switches

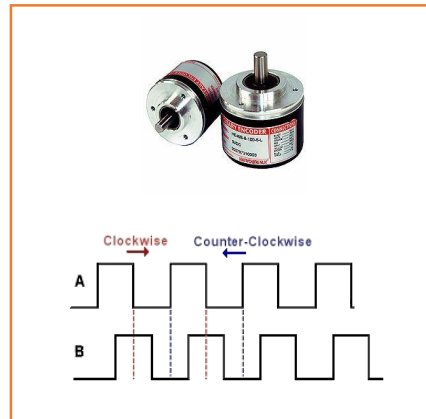
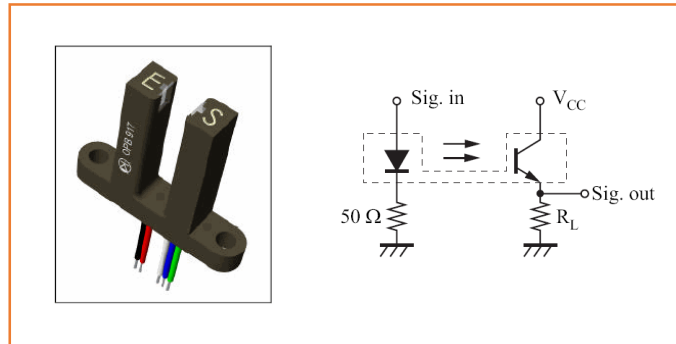


Switch

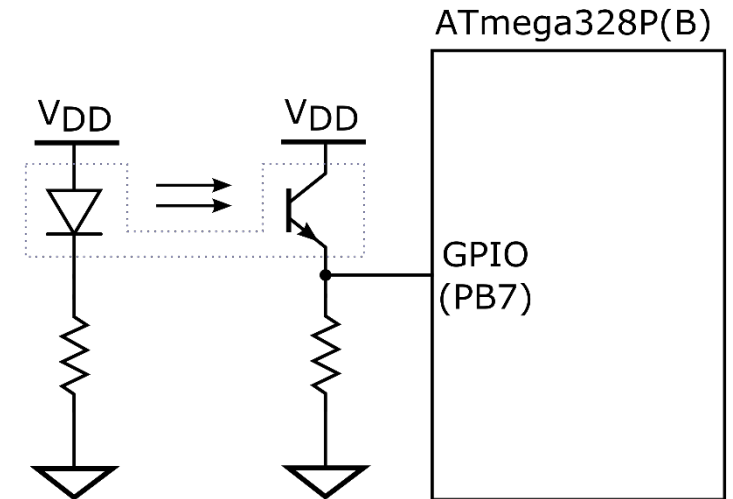


# GPIO (General Purpose Input Output)

- General Purpose **Input** is used
  - to accept signal from a device whose output is binary value, i.e. '0' or '1'.
  - Examples: opto-interrupter, rotary encoder

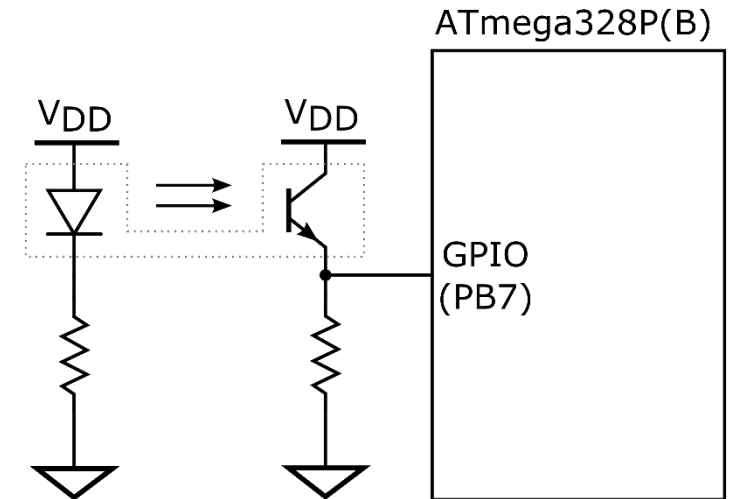
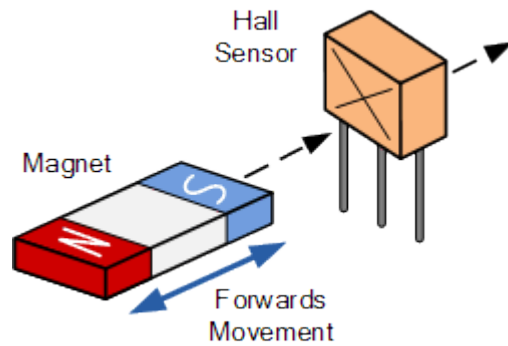


Rotary Encoder



# GPIO (General Purpose Input Output)

- General Purpose **Input** is used
  - to accept signal from a device whose output is binary value, i.e. '0' or '1'.
  - Examples: Hall-effect sensor, ultra-sonic sensor

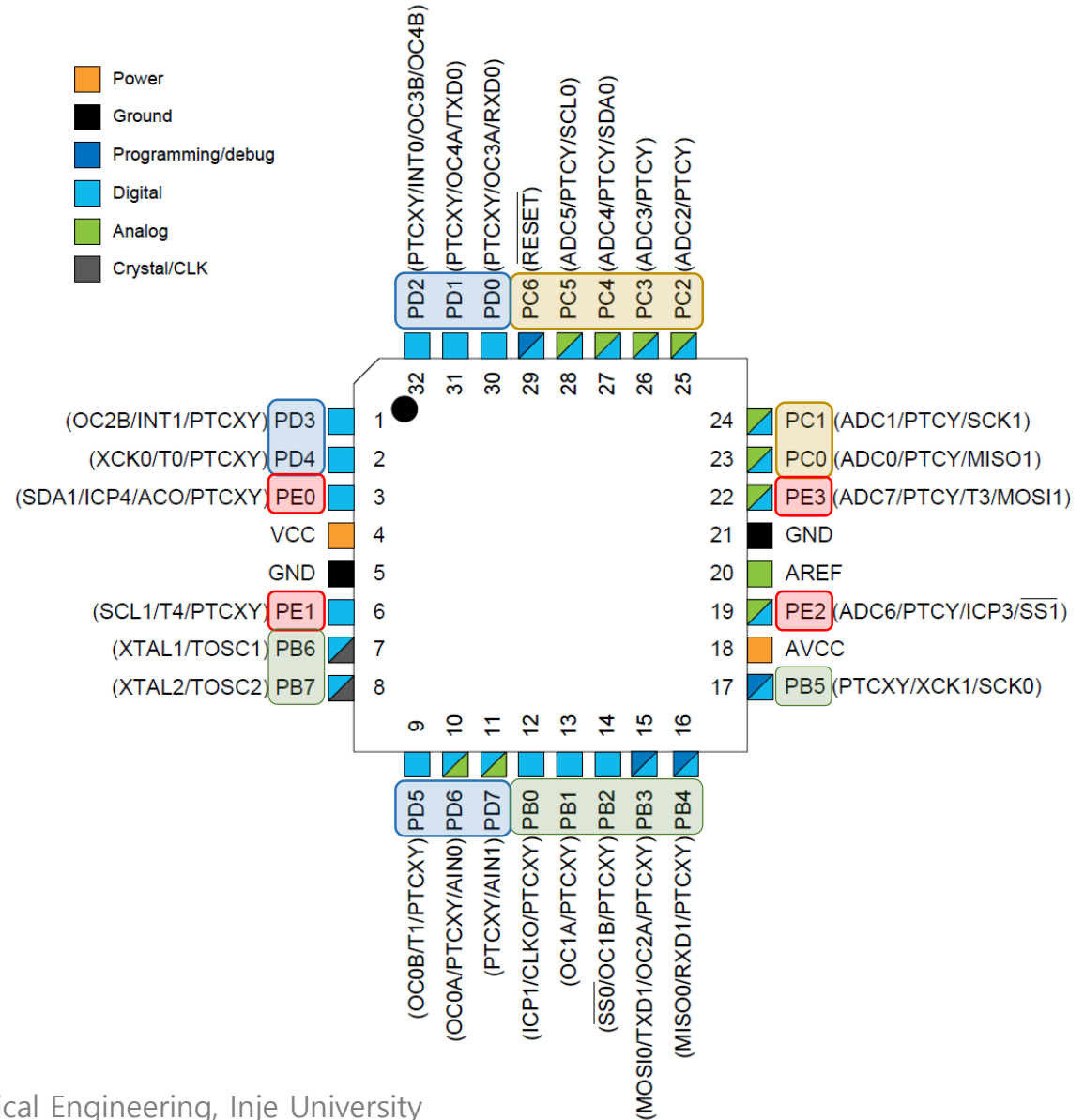




# GPIO (General Purpose Input Output)

ATmega328PB의 GPIO로 사용 가능한 핀

p14. ATmega328PB datasheet



# GPIO (General Purpose Input Output)

---

- GPIO Registers

- **DDRx** register

- Data direction of GPIO pins

- **PORTx** register

- Output data value

- **PINx** register

- Input data value

# GPIO (General Purpose Input Output)

## DDR<sub>x</sub>

x: Port name

DDRB, DDRC, DDRD, DDRE

DDRB

Bit No.	7	6	5	4	3	2	1	0
Name	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
Reset Value	0	0	0	0	0	0	0	0

- Determines data direction of GPIO pins
  - 1 → Output
  - 0 → Input

### Example: Port B

- Bit 5 and 3: output
- Remaining bits: input

```
DDRB = 0b00101000;
```

Bit No.	7	6	5	4	3	2	1	0
Name	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
Value	0	0	1	0	1	0	0	0

# GPIO (General Purpose Input Output)

## PORTx

x: Port name

PORTB, PORTC, PORTD, PORTE

PORTB

Bit No.	7	6	5	4	3	2	1	0
Name	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
Reset Value	0	0	0	0	0	0	0	0

➤ Holds output data value of GPIO pins

- 1 → High ( $V_{dd}$ )
- 0 → Low (GND)

### Example: Port B

- Bit 5: High, Bit 3: Low
- Remaining bits: input

```
PORTB = 0b00100000;
```

Bit No.	7	6	5	4	3	2	1	0
Name	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
Value	0	0	1	0	0	0	0	0

# GPIO (General Purpose Input Output) 예제 1

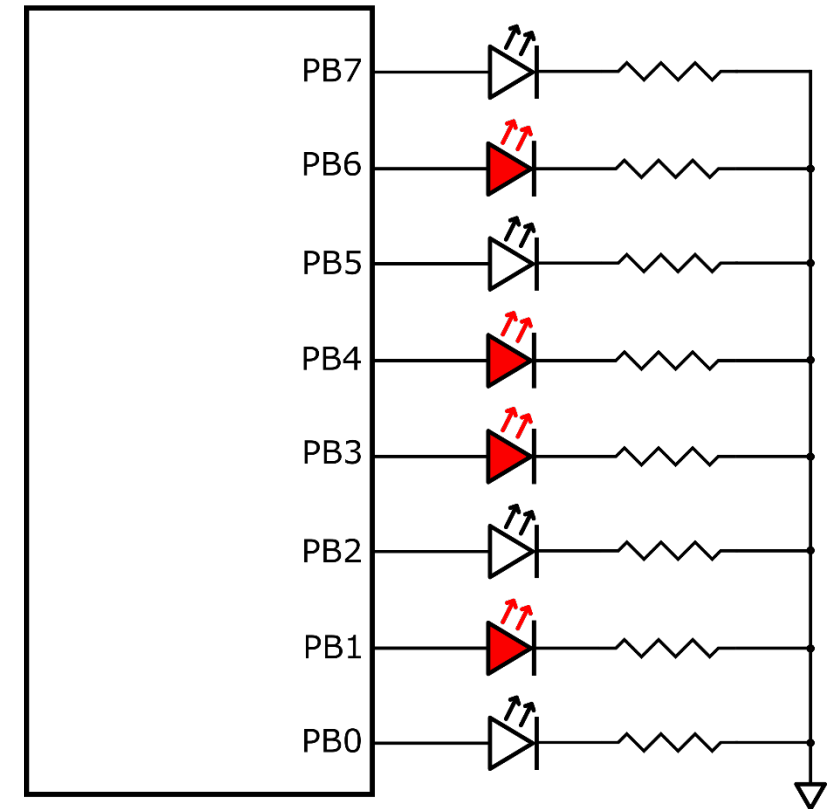
## Example: Port B

- Turn on LEDs at PB6, PB4, PB3 and PB1
- Turn off LEDs at the remaining bits

```
PORTB = 0b01011010;
```

Bit No.	7	6	5	4	3	2	1	0
Name	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
Value	0	1	0	1	1	0	1	0

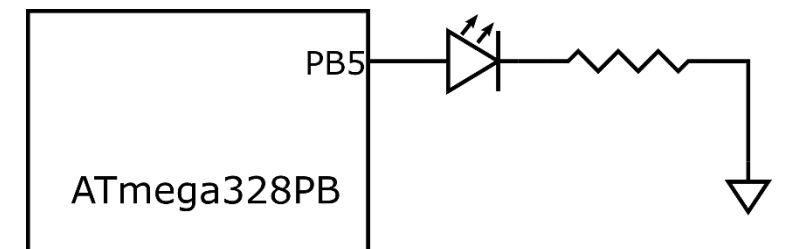
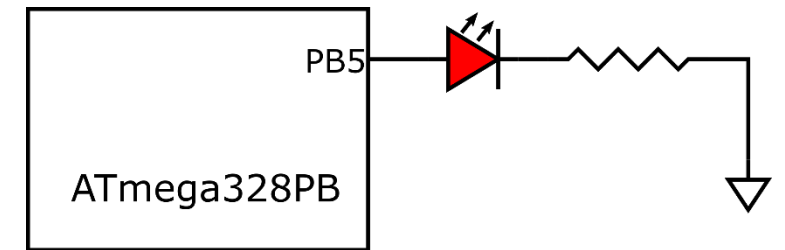
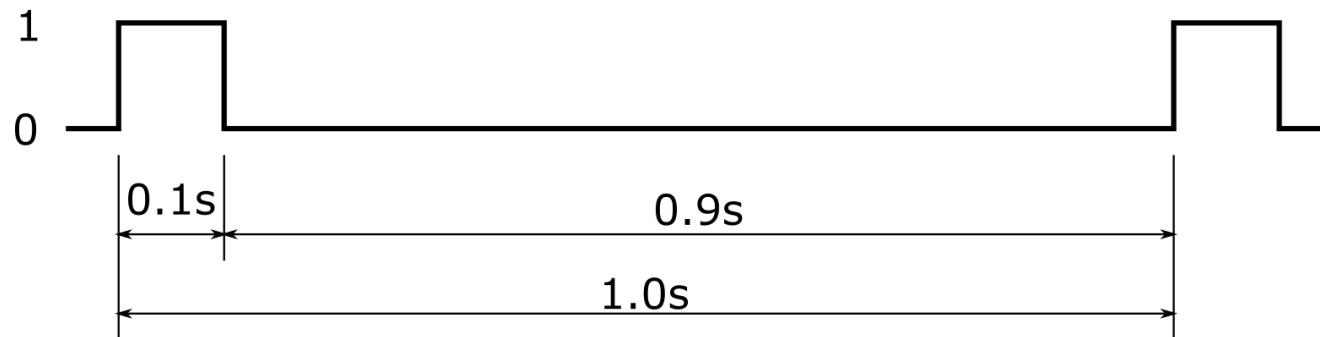
ATmega328PB



# GPIO (General Purpose Input Output) 예제 2 (1)

## Example: Port B로 펄스 신호 출력

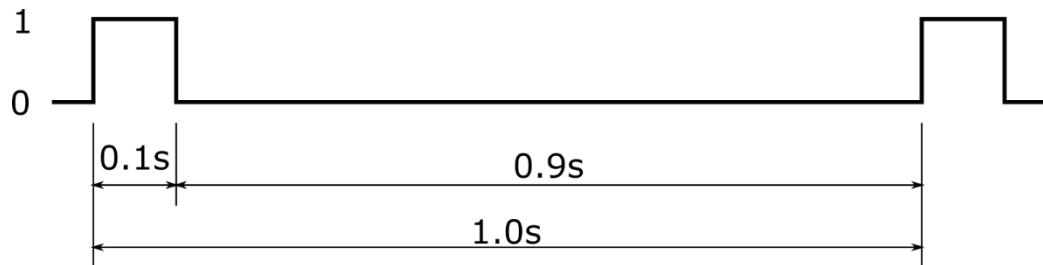
- PB5에 연결된 LED를 0.1초간 점등하고, 이어서 0.9초간 소등하는 동작을 무한히 반복하는 장치
- AVR-GCC에서 제공하는 `_delay_ms()` 함수 사용



# GPIO (General Purpose Input Output) 예제 2 (2)

## Example: Port B로 펄스 신호 출력

- PB5에 연결된 LED를 0.1초간 점등하고, 이어서 0.9초간 소등하는 동작을 무한히 반복하는 장치
- AVR-GCC에서 제공하는 `_delay_ms()` 함수 사용



```
#define F_CPU 16000000UL

#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
    DDRB = 0b00100000;

    while (1)
    {
        PORTB = 0b00100000; // Set PB5 to High
        _delay_ms(100); // for 0.1s

        PORTB = 0b00000000; // Set PB5 to High
        _delay_ms(900); // for 0.9s
    }
}
```

# GPIO (General Purpose Input Output)

## PINx Registers

x: Port name

PINB, PINC, PIND, PINE

Bit No.	7	6	5	4	3	2	1	0
Name	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0
Reset Value	0	0	0	0	0	0	0	0

- Used to read **input data** value of GPIO pins
  - High ( $V_{dd}$ ) → 1
  - Low (GND) → 0



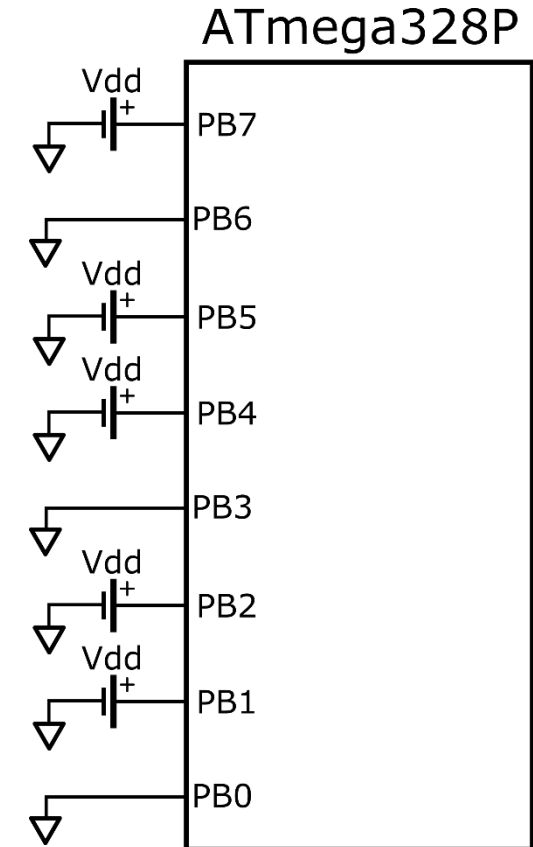
# GPIO (General Purpose Input Output)

## Example: PINB register value

- Assume that all bits of Port B are input mode.
- PINB register value for the right-hand side circuit is

PINB == 0b10110110

Bit No.	7	6	5	4	3	2	1	0
Name	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0
Value	1	0	1	1	0	1	1	0



# GPIO (General Purpose Input Output)

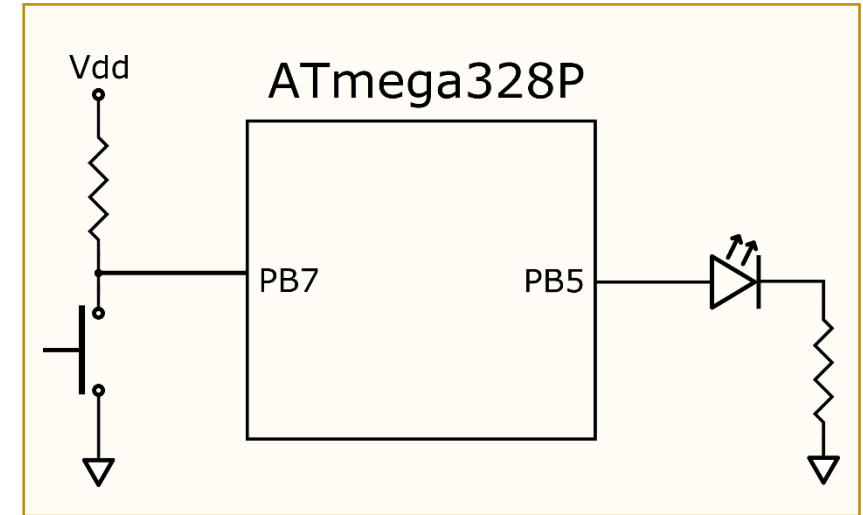
## Example: **DDRB**, **PORTB** and **PINB** registers

- Turn on LED at PB5 while the switch at PB7 is pressed (on).

```
#include <avr/io.h>

int main(void)
{
    DDRB = 0b00100000;           // set PB5 as OUTPUT mode

    while (1)
    {
        if ((PINB & 0b10000000) == 0) // if switch at PB7 is pressed
            PORTB = 0b00100000;       // turn ON LED at PB5
        else                            // else
            PORTB = 0b00000000;       // turn OFF LED at PB5
    }
}
```



# GPIO (General Purpose Input Output)

Example: **DDRB**, **PORTB** and **PINB** registers

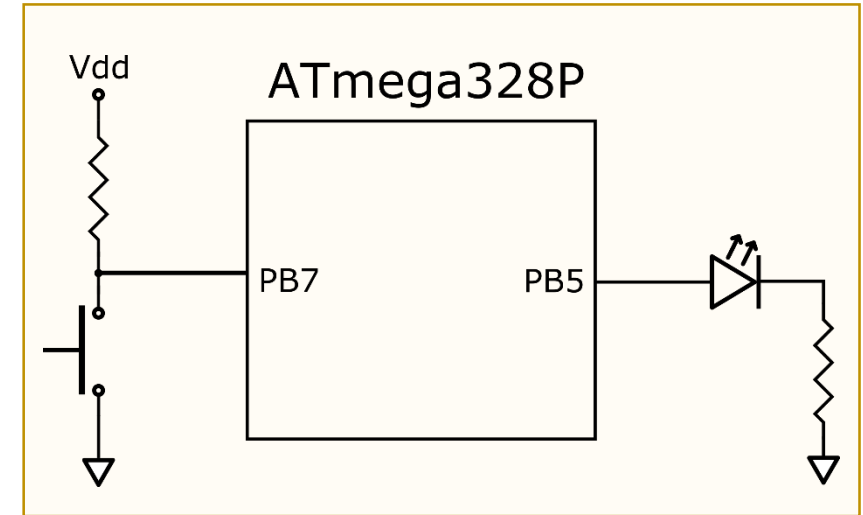
- Turn on LED at PB5 while the switch at PB7 is pressed (on).

```
#define SWITCH 7           // switch is connected at PB7
#define LED 5             // LED is connected at PB5

#include <avr/io.h>

int main(void)
{
    DDRB |= 1 << LED;      // set PB5 as OUTPUT mode

    while (1)
    {
        if ((PINB & (1 << SWITCH)) == 0) // if switch at PB7 is pressed
            PORTB |= 1 << LED;           // turn ON LED at PB5
        else
            PORTB &= ~(1 << LED);       // turn OFF LED at PB5
    }
}
```



# GPIO (General Purpose Input Output)

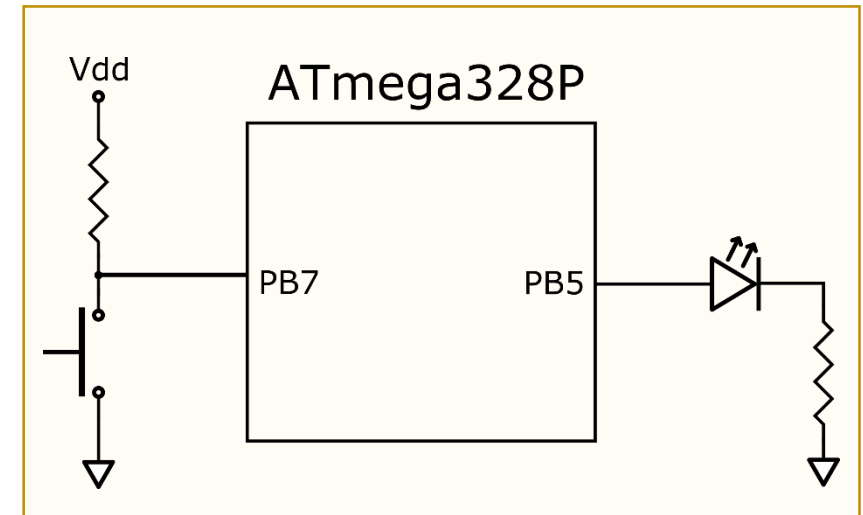
Example: **DDRB**, **PORTB** and **PINB** registers

- **Toggle** LED at PB5 whenever the switch at PB7 is pressed.

```
// toggle by XOR with '1'
#include <avr/io.h>

int main(void)
{
    DDRB |= (1 << 5);           // set PB5 as OUTPUT mode

    while (1)
    {
        if (!(PINB & (1 << 7))) // if switch at PB7 is pressed
        {
            PORTB ^= 1 << 5;    // toggle LED at PB5
            while (!(PINB & (1 << 7))); // wait until switch is released
        }
    }
}
```



# GPIO (General Purpose Input Output)

---

## **PIN<sub>x</sub>** Registers (used to **toggle**)

x: Port name

PIN<sub>B</sub>, PIN<sub>C</sub>, PIN<sub>D</sub>, PIN<sub>E</sub>

- Used to **toggle** data in the PORT<sub>x</sub> register
  - Writing '1' to PIN<sub>xn</sub> → **toggle** n bit in the PORT<sub>x</sub> register
  - Writing '0' to PIN<sub>xn</sub> → no change in the PORT<sub>x</sub> register

# GPIO (General Purpose Input Output)

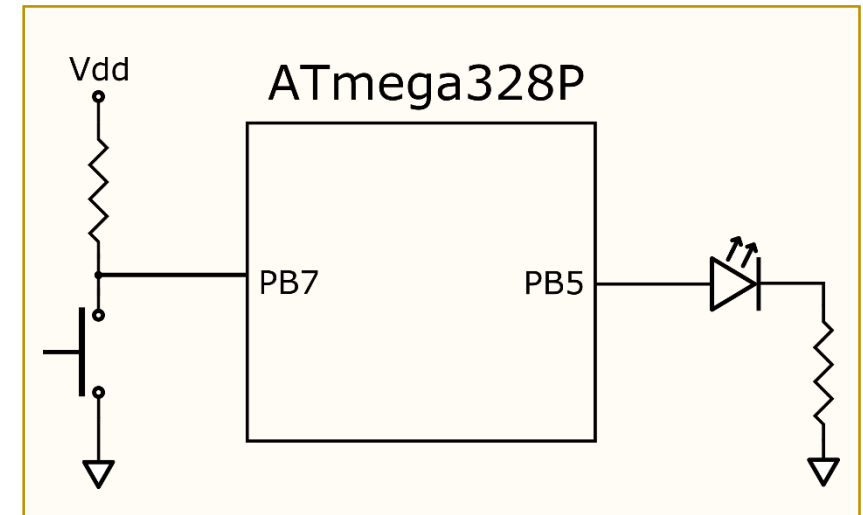
Example: **DDRB**, **PORTB** and **PINB** registers

- **Toggle** LED at PB5 whenever the switch at PB7 is pressed.

```
// toggle by writing '1' to PINB
#include <avr/io.h>

int main(void)
{
    DDRB |= (1 << 5);           // set PB5 as OUTPUT mode

    while (1)
    {
        if (!(PINB & (1 << 7))) // if switch at PB7 is pressed
        {
            PINB |= 1 << 5;     // toggle LED at PB5
            while (!(PINB & (1 << 7))); // wait until switch is released
        }
    }
}
```

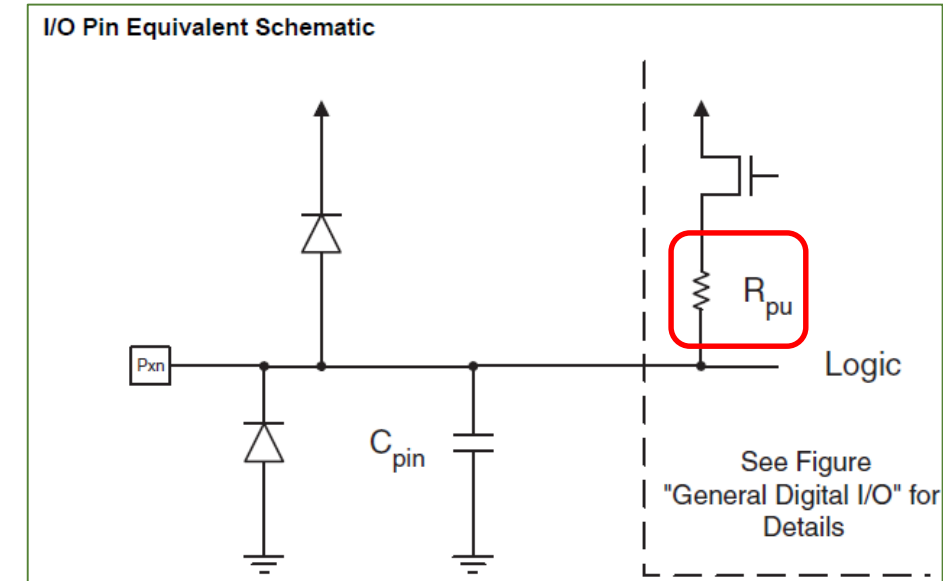


# GPIO (General Purpose Input Output)

- Global **pull-up** control: **PUD** bit in **MCUCR** register
  - **0** → Pull-up **enable**
  - **1** → All pull-ups in the GPIO are **disabled**.

MCUCR

Bit No.	7	6	5	4	3	2	1	0
Name	-	BODS	BODSE	<b>PUD</b>	-	-	IVSEL	IVCE
Reset Value	0	0	0	0	0	0	0	0



- Individual pull-up control: each bit in PORTx register for input mode
  - **1** → Pull-up **enable**
  - **0** → Pull-up in the GPIO pin is **disabled**.

# GPIO (General Purpose Input Output)

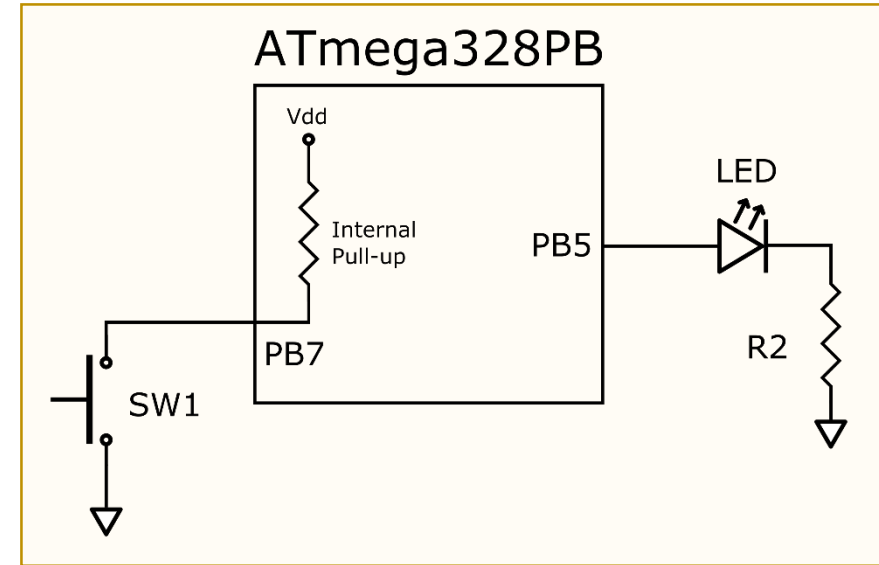
## Example: DDRB, PORTB and PINB registers

- Turn on LED at PB5 while the switch at PB7 is pressed (on).
- External pull-up resistor was replaced by an **internal pull-up** resistor.

```
#include <avr/io.h>

int main(void)
{
    DDRB  |= 1 << 5;           // set PB5 as OUTPUT mode
    PORTB |= 1 << 7;           // enable Pull-Up at PB7

    while (1)
    {
        if ((PINB & (1 << 7)) == 0) // if switch at PB7 is pressed
            PORTB |= 1 << 5;         // turn ON LED at PB5
        else
            PORTB &= ~(1 << 5);      // turn OFF LED at PB5
    }
}
```



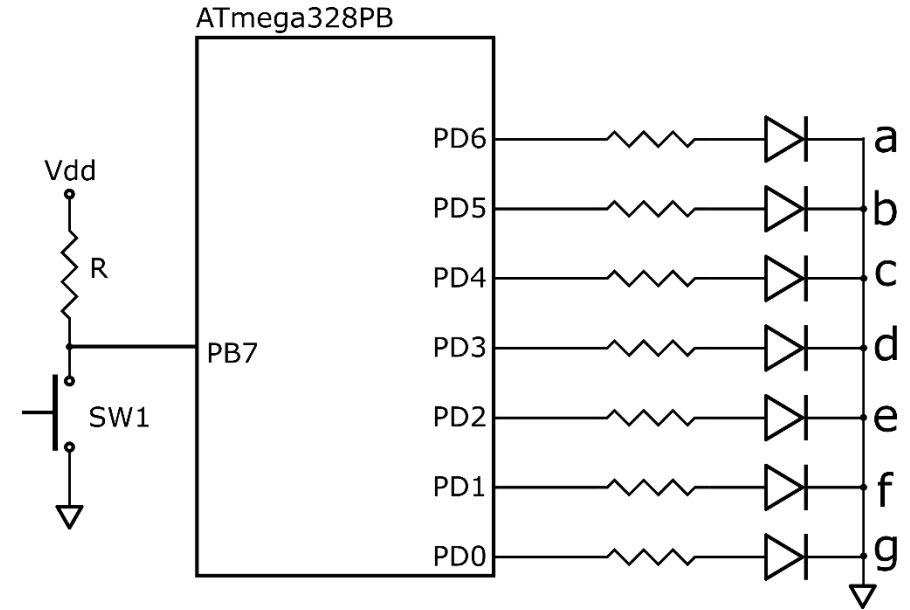
atmega328p\_gpio\_input\_output\_internal\_pull\_up.png



# GPIO 과제

## (문제) Mod-10 counter 구현

- mod-10 counter는 0부터 9까지 상향 계수.
- PB7에 연결된 스위치(SW1)을 누를 때마다 mod-10 counter의 값이 1씩 증가.
- mod-10 counter의 값이 10이 되는 순간에 0으로 되돌아감.
- mod-10 counter의 값을 Port D에 연결된 Seven-Segment Display(SSD)에 출력
- SSD에 출력될 값(8-비트 상수)을 미리 배열로 선언해서 사용할 것. 즉, 이 배열은 10개의 원소로 구성되며, 각각의 원소는 8-비트 상수이다.
- 프로그램의 중요한 부분에는 자신만의 주석문을 반드시 추가할 것.



# GPIO (General Purpose Input Output)

- Symmetrical drive characteristics with both high sink and source capability.
  - $I_{OH\ max}$  = 20mA at VCC = 5V, 10mA at VCC = 3V
    - The sum of all IOH, for ports C0 - C5, D0- D4, ADC7, RESET should not exceed 150mA.
    - The sum of all IOH, for ports B0 - B5, D5 - D7, ADC6, XTAL1, XTAL2 should not exceed 150mA.
  - $I_{OL\ max}$  = 20mA at VCC = 5V, 10mA at VCC = 3V
    - The sum of all IOL, for ports C0 - C5, ADC7, ADC6 should not exceed 100mA.
    - The sum of all IOL, for ports B0 - B5, D5 - D7, XTAL1, XTAL2 should not exceed 100mA.
    - The sum of all IOL, for ports D0 - D4, RESET should not exceed 100mA.

# GPIO (General Purpose Input Output)

## ATmega328P GPIO Pin Driver Strength

Figure 35-22 ATmega328PB: I/O Pin Output Voltage vs. Sink Current ( $V_{CC} = 3V$ )

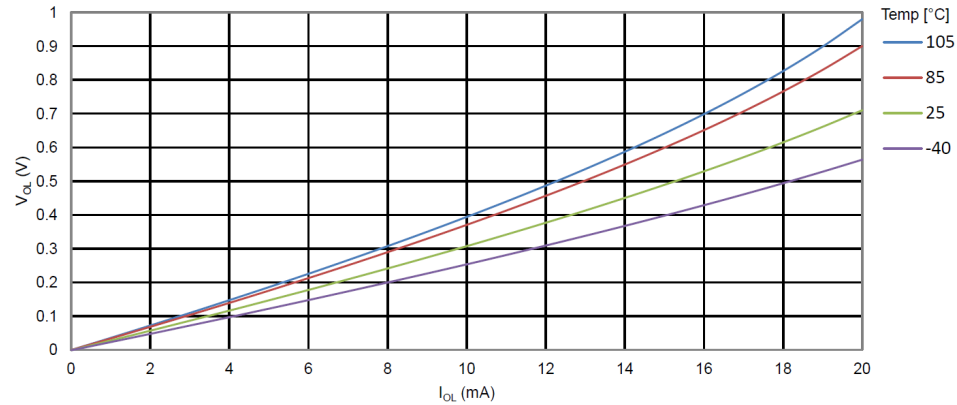


Figure 35-23 ATmega328PB: I/O Pin Output Voltage vs. Sink Current ( $V_{CC} = 5V$ )

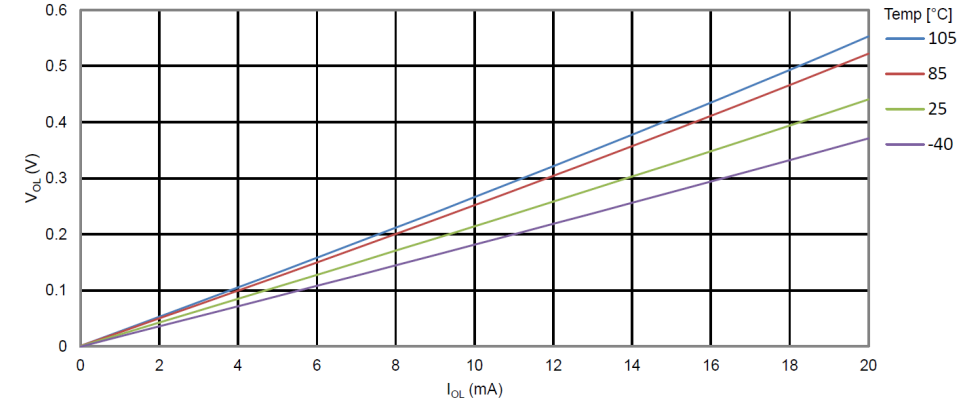


Figure 35-24 ATmega328PB: I/O Pin Output Voltage vs. Source Current ( $V_{CC} = 3V$ )

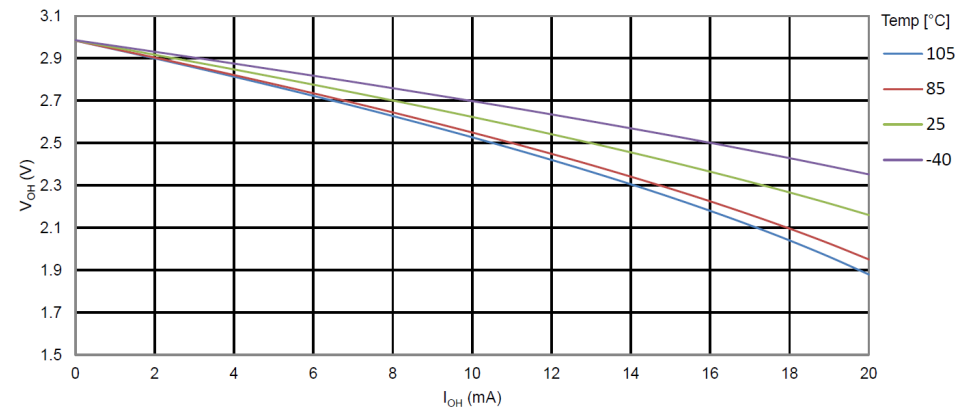
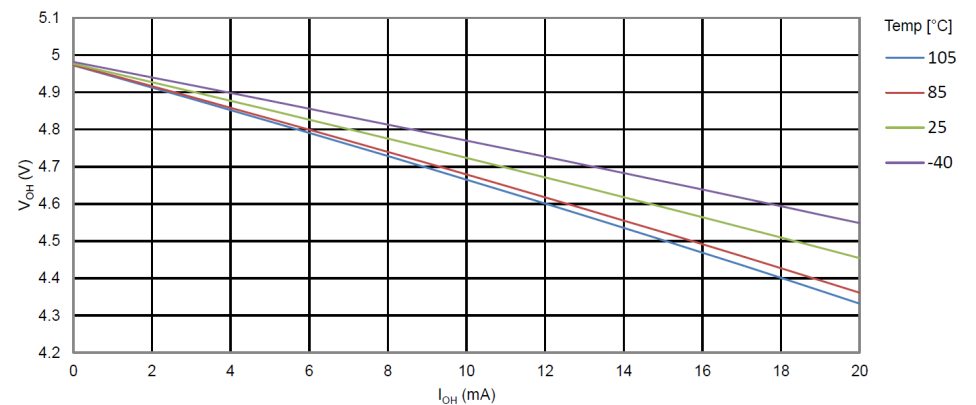


Figure 35-25 ATmega328PB: I/O Pin Output Voltage vs. Source Current ( $V_{CC} = 5V$ )



# What's next?

