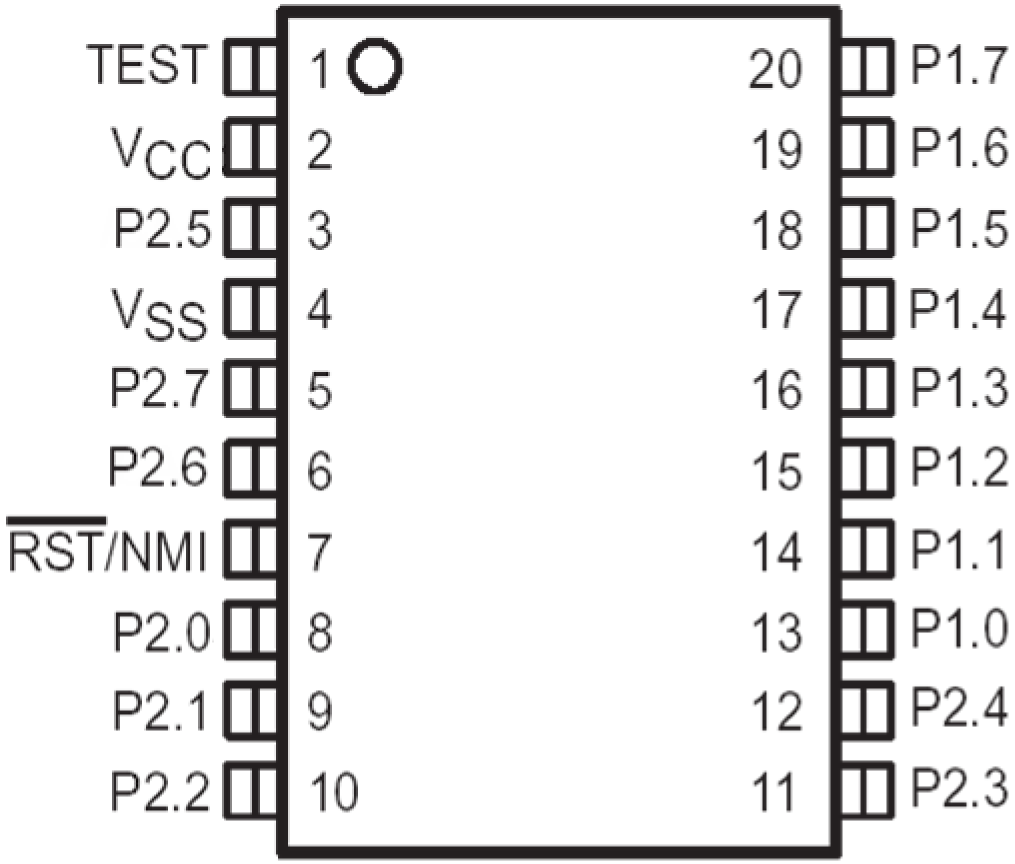
Session-V
GPIO programming concepts

Session outline

- I/O port representation in microcontroller
- Atmega328P I/O port structure
- Interfacing I/O devices with GPIO ports of Atmega328P
- GPIO register configuration and programming

How I/O port represented

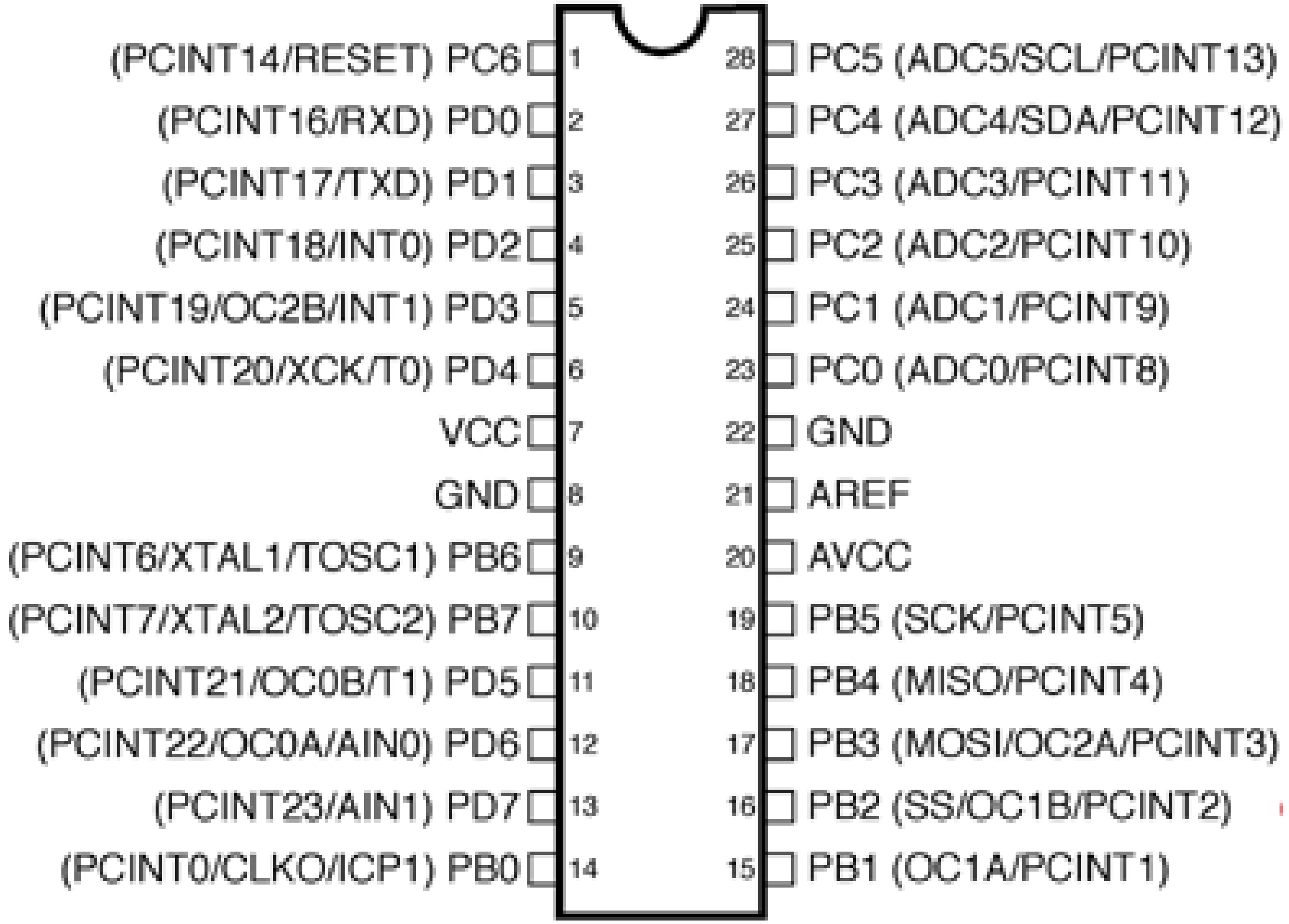


	P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
I/O Port	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

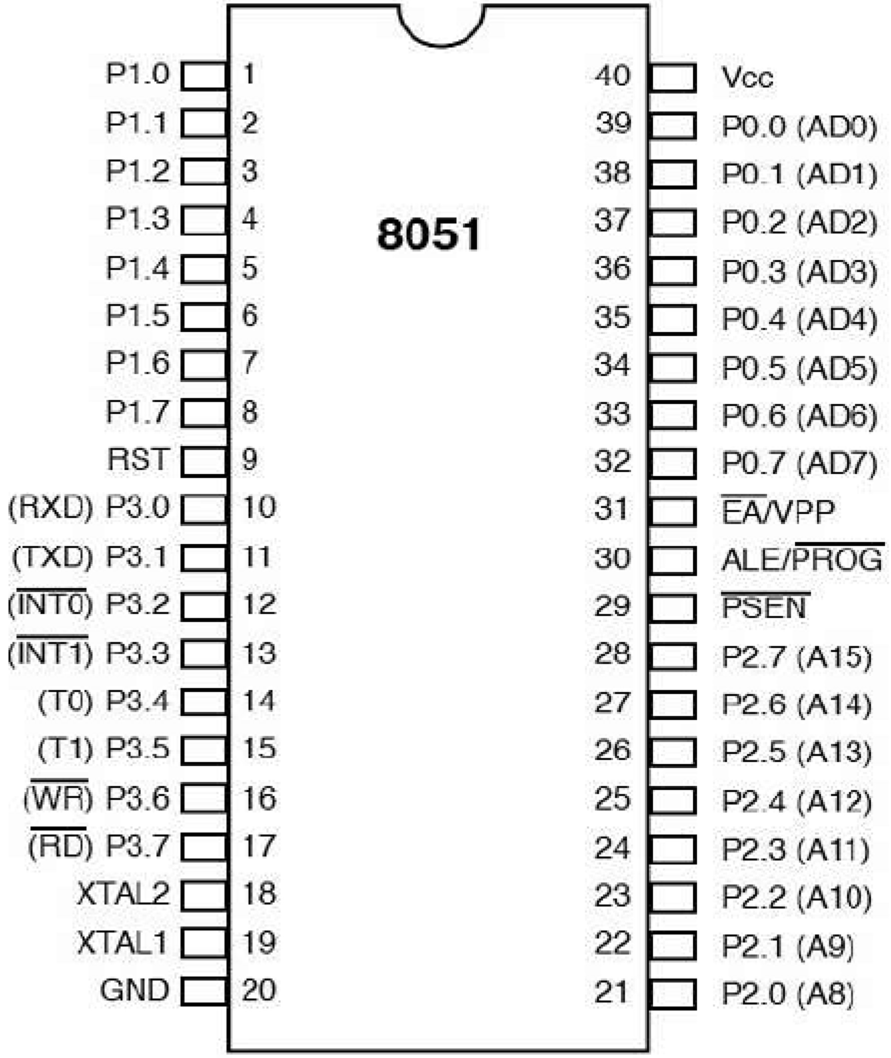
8 bit I/O port

Example: Atmega328P Vs 8051

Atmega328P



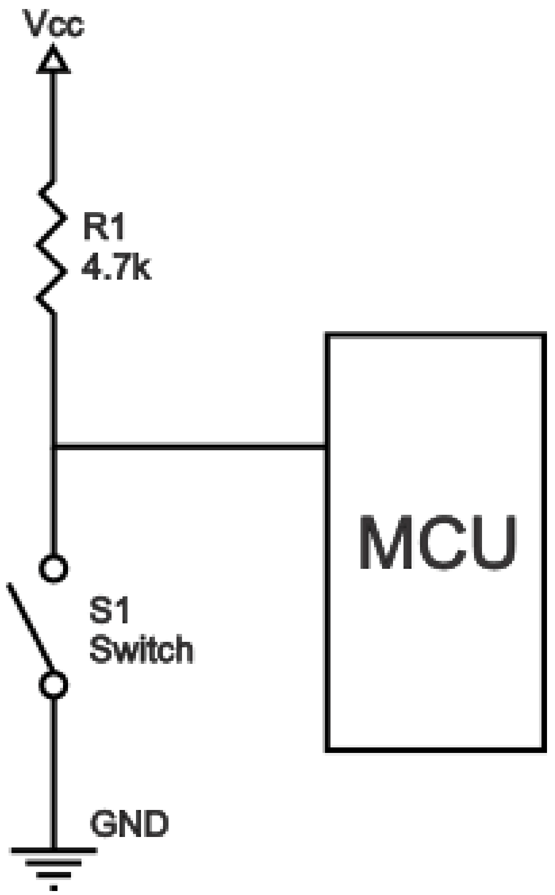
8051

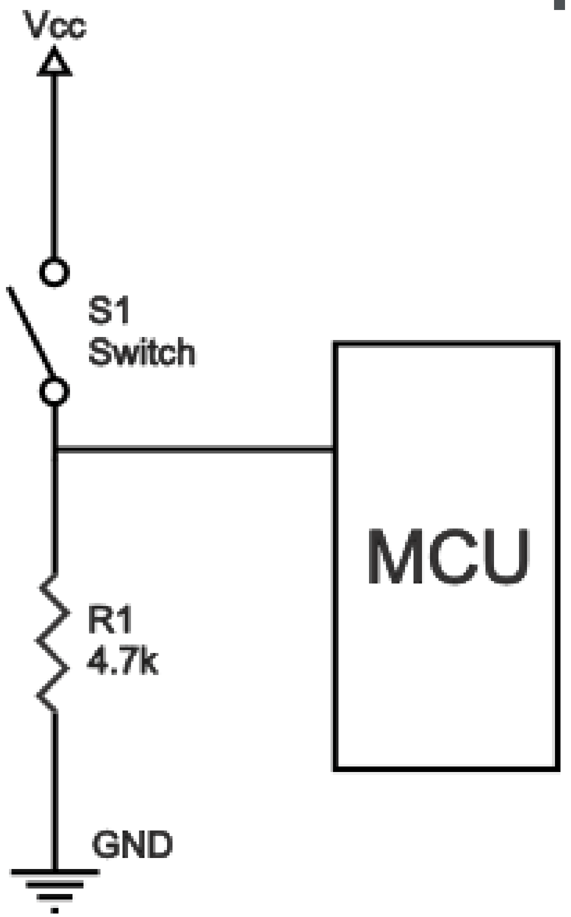


Role of I/O ports

- I/O ports can be configured as INPUT or OUTPUT
- For example,
- When connecting to DIP switch, push button, the I/O pin can be configured as INPUT
- When connecting to LED or Relay, I/O pin can be configured as OUTPUT
- Here DIP switch, push buttons are input devices, whereas LED or Relay are output devices.

How to connect I/O device with microcontroller pins

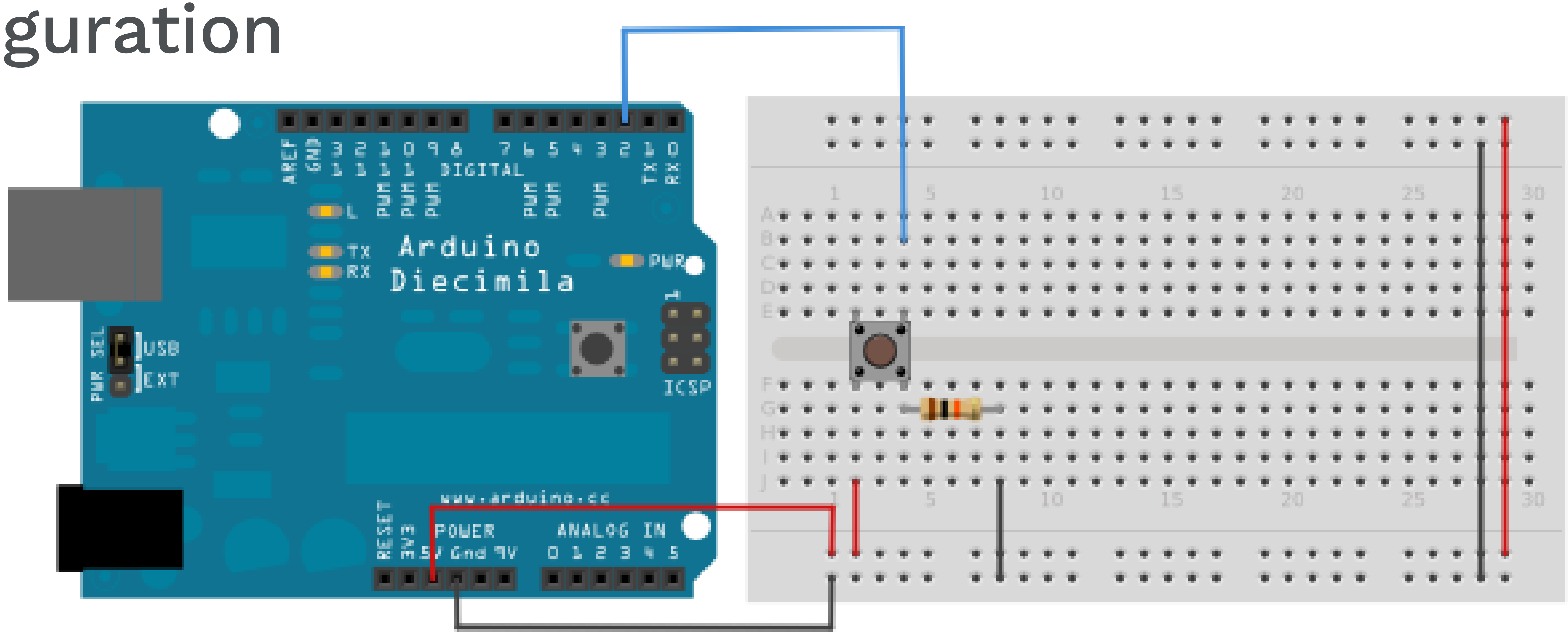




Pull-up resistors are resistors which are used to ensure that a wire is pulled to a high logical level in the absence of an input signal.

Pull-down resistors are resistors which are used to ensure that a wire is pulled to a Low logical level in the absence of an input signal.

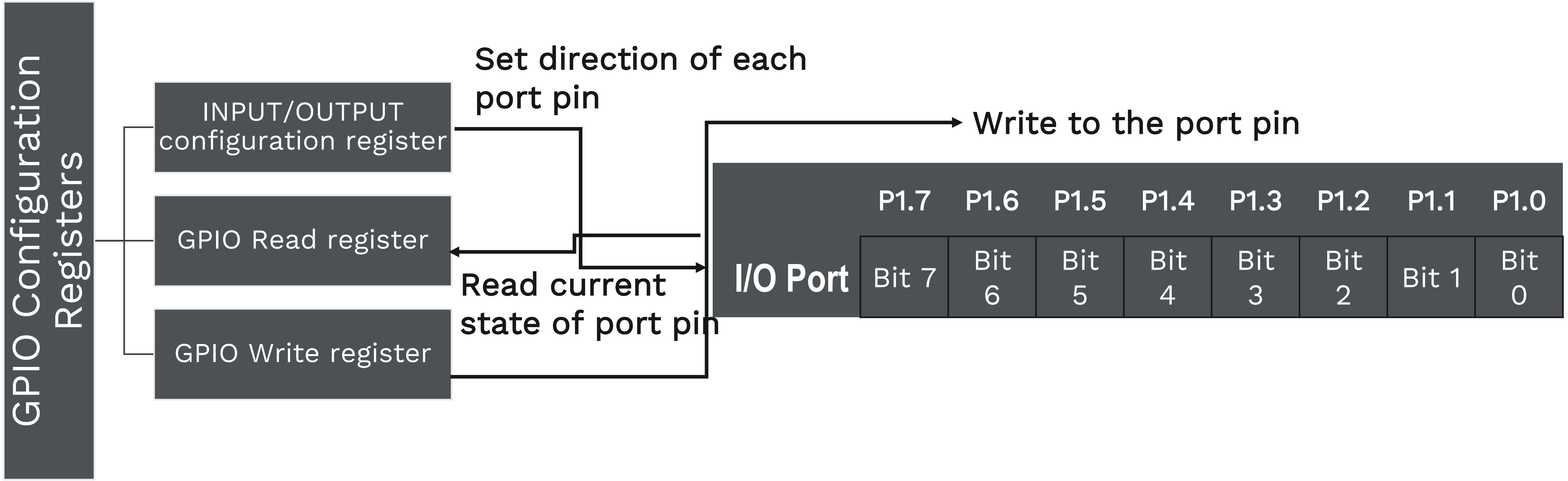
Example circuit: A switch connected using pull-down configuration



When the pushbutton is un-pressed there is no connection between the two legs of the pushbutton, so the pin is connected to ground (through the pull-down resistor) and we read a LOW.

When the button is closed (pressed), it makes a connection between its two legs,

Conceptual GPIO port structure



GPIO Configuration Registers

INPUT/OUTPUT configuration register

GPIO Read register

GPIO Write register

Set direction of each port pin

Read current state of port pin

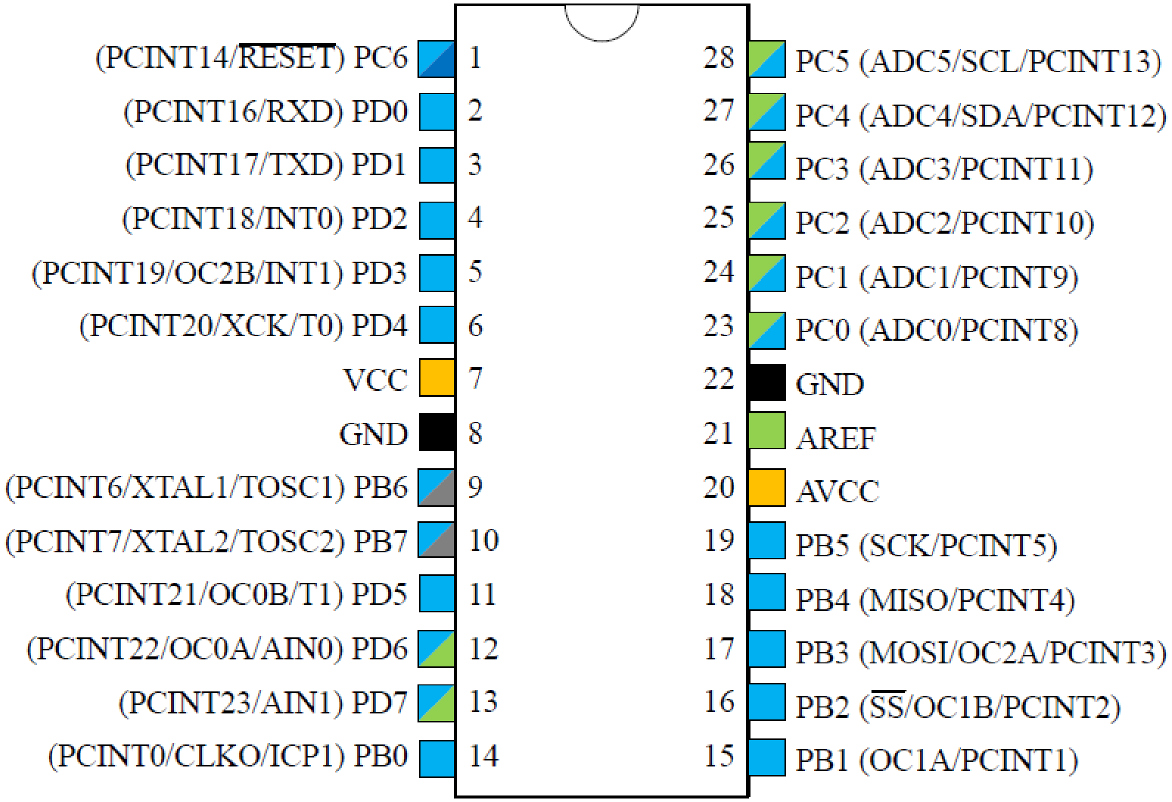
Write to the port pin

I/O Port	P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

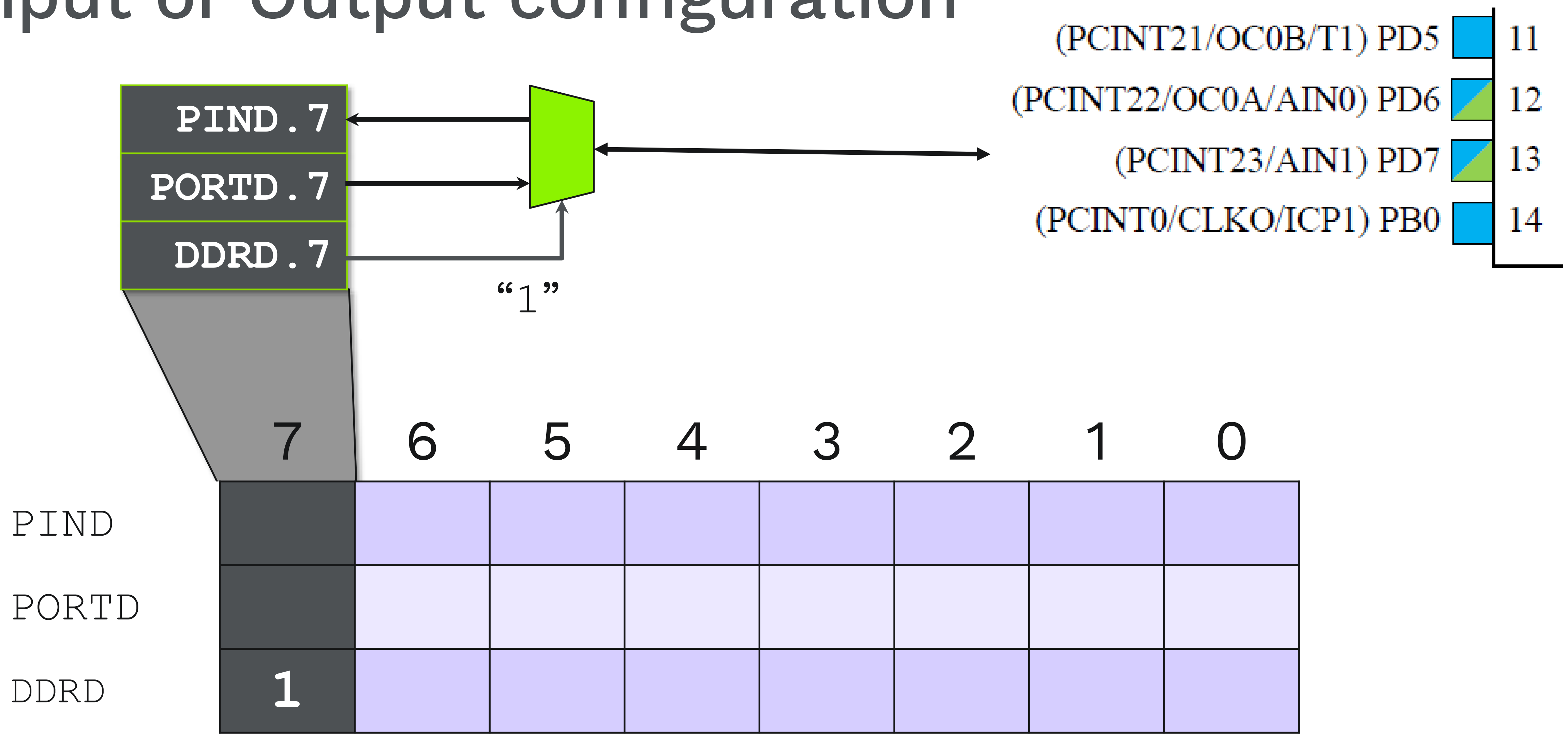
Understanding GPIO port configuration for Atmega328P

- GPIO configuration registers are as follows :
- (x can be replaced by A,B,C,D as per the microcontroller)

- DDRx register: I/O Configuration register
- PORTx register: GPIO write register
- PINx register: GPIO read register



DDRx: Input or Output configuration



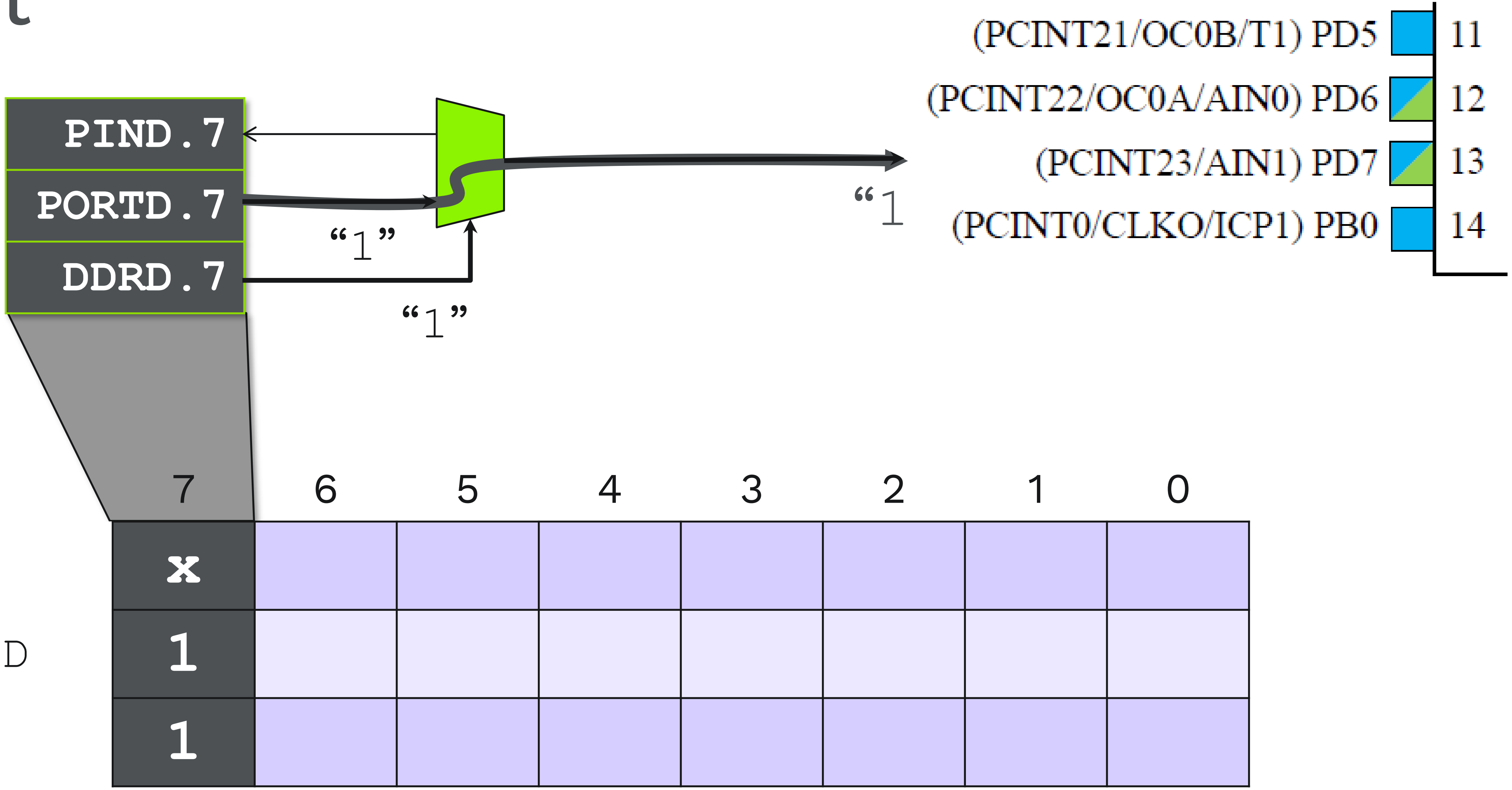
	7	6	5	4	3	2	1	0
PIND								
PORTD								
DDRD	1							

- DDRD.y: 0 = input
  1 = output

- Register access example:
  `DDRD |= 0x80;`

Each bit in each DDRx register selects the direction of the corresponding I/O pin, regardless of the selected function for the pin.

GPIO Output

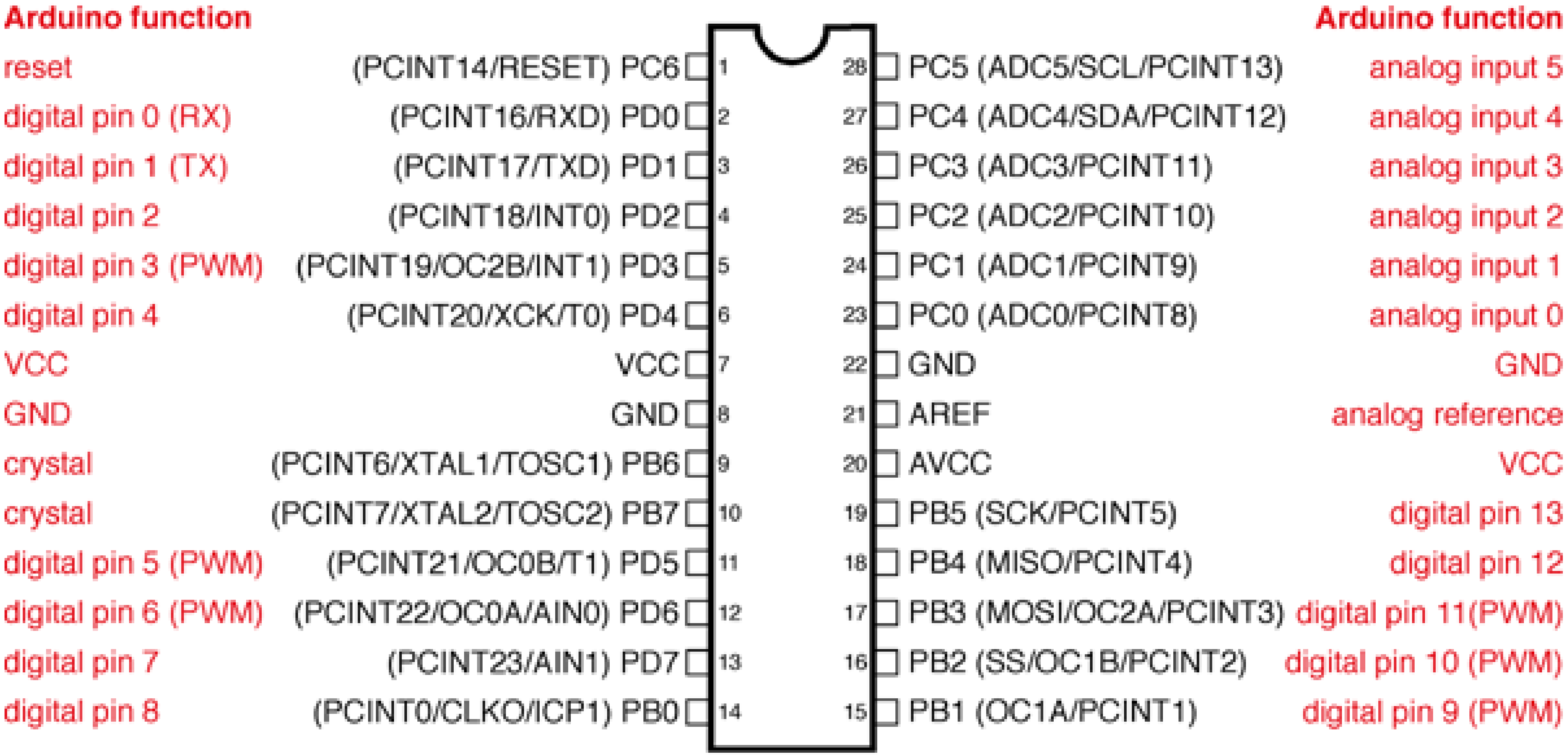


	7	6	5	4	3	2	1	0
PIND	x							
PORTD	1							
DDRD	1							

- PORTD.y: 0 = low
  1 = high
- Register example:
  `PORTD |= 0x80;`

Each bit in each PORTx register is the value to be output on the corresponding I/O pin when the pin is configured as I/O function, output direction.

Arduino Uno and ATmega328P pin mapping



Arduino function					Arduino function
reset	(PCINT14/RESET) PC6	1	28	PC5 (ADC5/SCL/PCINT13)	analog input 5
digital pin 0 (RX)	(PCINT16/RXD) PD0	2	27	PC4 (ADC4/SDA/PCINT12)	analog input 4
digital pin 1 (TX)	(PCINT17/TXD) PD1	3	26	PC3 (ADC3/PCINT11)	analog input 3
digital pin 2	(PCINT18/INT0) PD2	4	25	PC2 (ADC2/PCINT10)	analog input 2
digital pin 3 (PWM)	(PCINT19/OC2B/INT1) PD3	5	24	PC1 (ADC1/PCINT9)	analog input 1
digital pin 4	(PCINT20/XCK/T0) PD4	6	23	PC0 (ADC0/PCINT8)	analog input 0
VCC	VCC	7	22	GND	GND
GND	GND	8	21	AREF	analog reference
crystal	(PCINT6/XTAL1/TOSC1) PB6	9	20	AVCC	VCC
crystal	(PCINT7/XTAL2/TOSC2) PB7	10	19	PB5 (SCK/PCINT5)	digital pin 13
digital pin 5 (PWM)	(PCINT21/OC0B/T1) PD5	11	18	PB4 (MISO/PCINT4)	digital pin 12
digital pin 6 (PWM)	(PCINT22/OC0A/AIN0) PD6	12	17	PB3 (MOSI/OC2A/PCINT3)	digital pin 11(PWM)
digital pin 7	(PCINT23/AIN1) PD7	13	16	PB2 (SS/OC1B/PCINT2)	digital pin 10 (PWM)
digital pin 8	(PCINT0/CLKO/ICP1) PB0	14	15	PB1 (OC1A/PCINT1)	digital pin 9 (PWM)

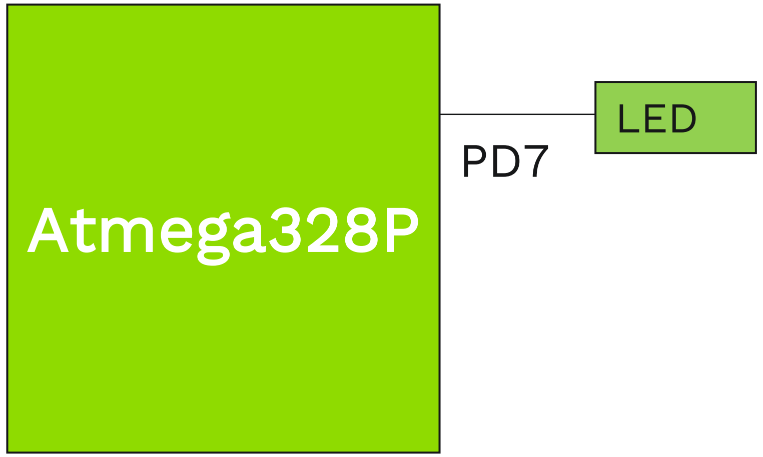
Example

Draw circuit in to connect LED to PD7 pin

Write a program to Blink LED connected to port pin PD7 every 1 second delay

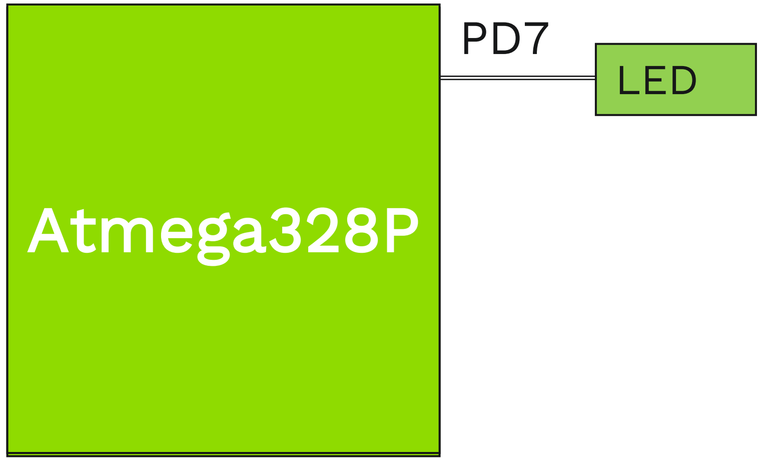
```
void main()
{
    DDRD = 0xFF;           //PD as output
    PORTD= 0x00;           //keep all LEDs off

    while(1) //super loop
    {
        PORTD &= 0b01111111;       //turn LED off
        delay_ms(1000);    //wait for 1 second
        PORTD |= 0b10000000;       //turn LED on
        delay_ms(1000);    //wait for 1 second
    };
}
```



Super loop is required as we do not have operating system to return to

Exercise 1



Blink LED connected to port pin PD7 every 1 second delay

Use below approaches:

A. Modify the code that uses Hexadecimal value representation in stead of binary values for assigning data to GPIO port registers PORTD.

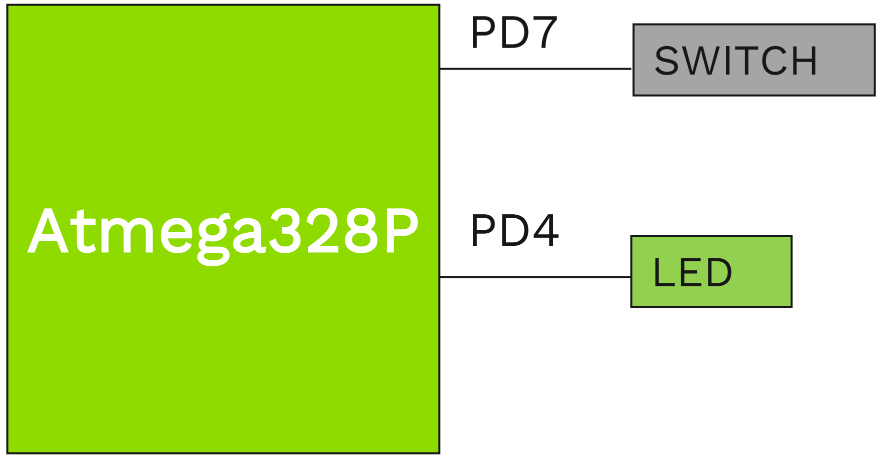
B. Modify the code to use bit wise operators to perform the GPIO port access for PORTD.

Exercise 2

Step-I: Draw circuit as per shown representation

Step-II: Write a C program to perform the following:

A. When switch is pressed, make LED On.
B. When switch is not pressed, make LED off.


Atmega328P
PD7
SWITCH
PD4
LED