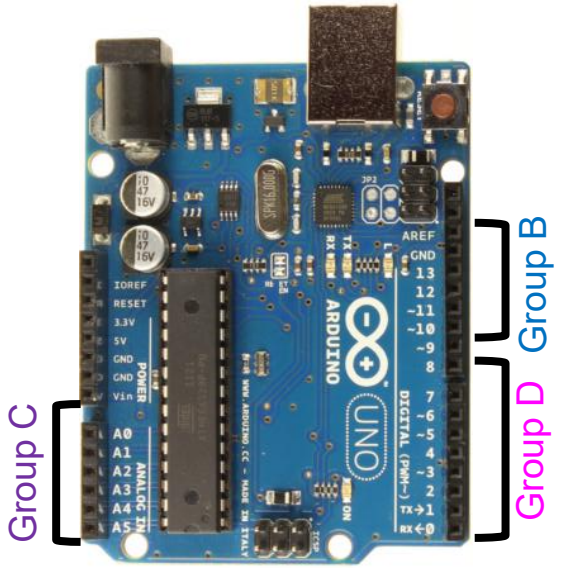
EE 109 Lab 3

Quick-Reference

Arduino Port/Pin Mapping

- When building your circuit, we use the Arduino connector bit names
- When writing your software, you must use the Group B, C, D (DDRx, PORTx, PINx) names.

SW	Arduino	SW	Arduino	SW	Arduino
PortB, bit0	DIG8	PortC, bit0	AN0	PortD, bit0	DIG0
PortB, bit1	DIG9	PortC, bit1	AN1	PortD, bit1	DIG1
PortB, bit2	DIG10	PortC, bit2	AN2	PortD, bit2	DIG2
PortB, bit3	DIG11	PortC, bit3	AN3	PortD, bit3	DIG3
PortB, bit4	DIG12	PortC, bit4	AN4	PortD, bit4	DIG4
PortB, bit5	DIG13	PortC, bit5	AN5	PortD, bit5	DIG5
PortB, bit6	**Clock1 (don't use)**	PortC, bit6	**Reset (don't use)**	PortD, bit6	DIG6
PortB, bit7	**Clock2 (don't use)**			PortD, bit7	DIG7



Software to Arduino Name Mapping

Group B bit5-bit0 = DIG13-DIG8

Group C bit5-bit0 = A5-A0

Group D bit7-bit0 = DIG7-DIG0

Follow the Recipes

- There 6 common actions you MUST know how to perform, but luckily there are simple ***recipes/patterns*** for each shown below.

Action	Register & Bitwise Operation	Description
Set Pin as Output	`DDRB |= (1 << 4);`	Configures B4 as an output pin
Set Pin as Input	`DDRB &= ~(1 << 4);`	Configures B4 as an input pin
Set Output Value to 1	`PORTB |= (1 << 4);`	Sets B4 high (logic 1)
Clear Output Value to 0	`PORTB &= ~(1 << 4);`	Sets B4 low (logic 0)
Check if input is 1	`(PINB & (1 << 4)) != 0`	Reads B4 and checks if it's high
Check if input is 0	`(PINB & (1 << 4)) == 0`	Reads B4 and checks if it's low

- Also, recall to enable the **built-in** pullup resistor for an input, set the corresponding PORT register bit to 1.

Common Mistakes

- Don't make these mistakes
- Instead remember:
  - Never shift a 0 when trying to do an AND or OR (e.g. 0 << x)
  - Correctly parenthesize your comparisons
  - To check if a bit is 1, check if the result of the ANDing is not-equal to 0

```
// Clearing a bit to 0
// Wrong
PORTD &= (0 << 3);
PORTD |= (0 << 3);

// Right
PORTD &= ~(1 << 3);


// Checking a bit
// Wrong
if(PIND & (1 << 3) == 0)

// Right
if( (PIND & (1 << 3)) == 0)

// Checking if a bit is 1
// Wrong
if( (PIND & (1 << 3)) == 1)

// Right
if( (PIND & (1 << 3)) != 0)
```

Practice

- Interactive visualization (link)
- Practice problem generator (link)
  - Generate random problems for you to code to ensure you can write appropriate commands to SET, CLEAR, FLIP and CHECK bits