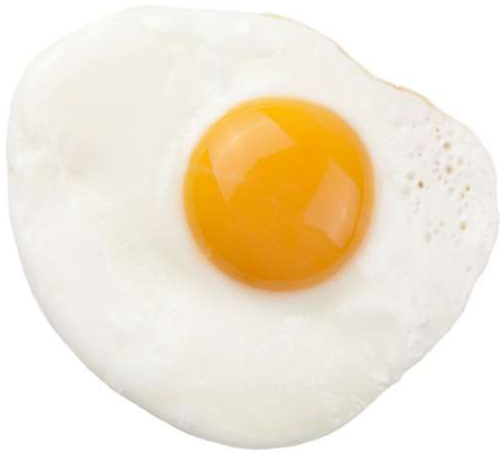
Arduino over Easy

Dr. Jim Grover



Overview

What is Arduino?

Basic Hardware Architecture

Shields

Integrated Development Environment

Program Template

Polling I/O

Interrupt Driven I/O

Useful Links

Disclaimer

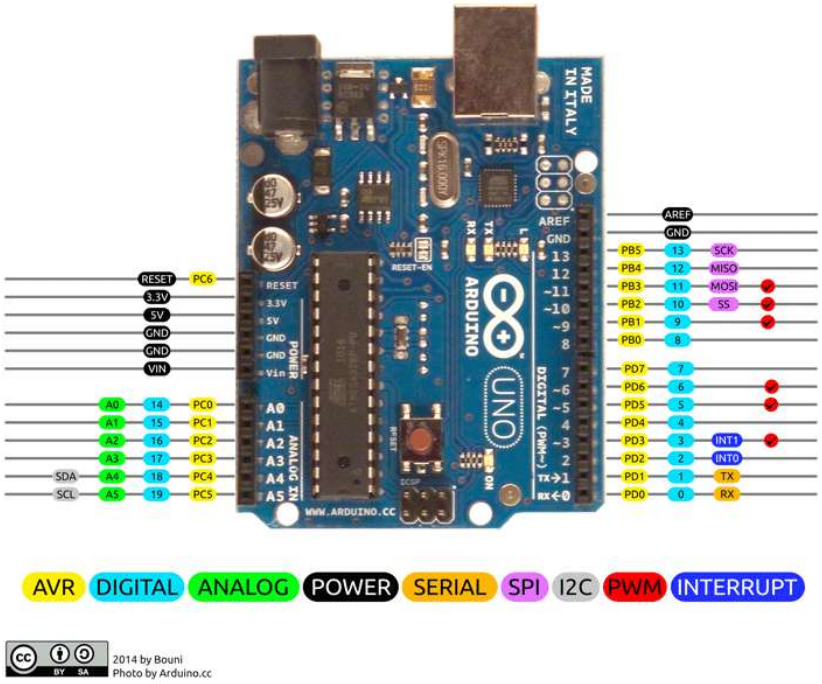
- Will talk about ATmega based environment
- Will not discuss ARM based boards
- Will not discuss Python based development
- Will not discuss, “What is the best Arduino board?”
- Will not discuss whether you should buy a licensed boards or China Inc knock offs

What is Arduino

- Open Source Computer Hardware
- Integrated Development Environment
- Community
- Popular IoT Development Platform

Basic Hardware Architecture

- Digital I/O (Uno 17)
- Analog In (Uno 5)
- Serial (Uno 1 Pair)
- SPI (Uno 1 interface)
- I2C (Uno 1 interface)
- PWM(Uno 6)
- External Interrupts (Uno 2)



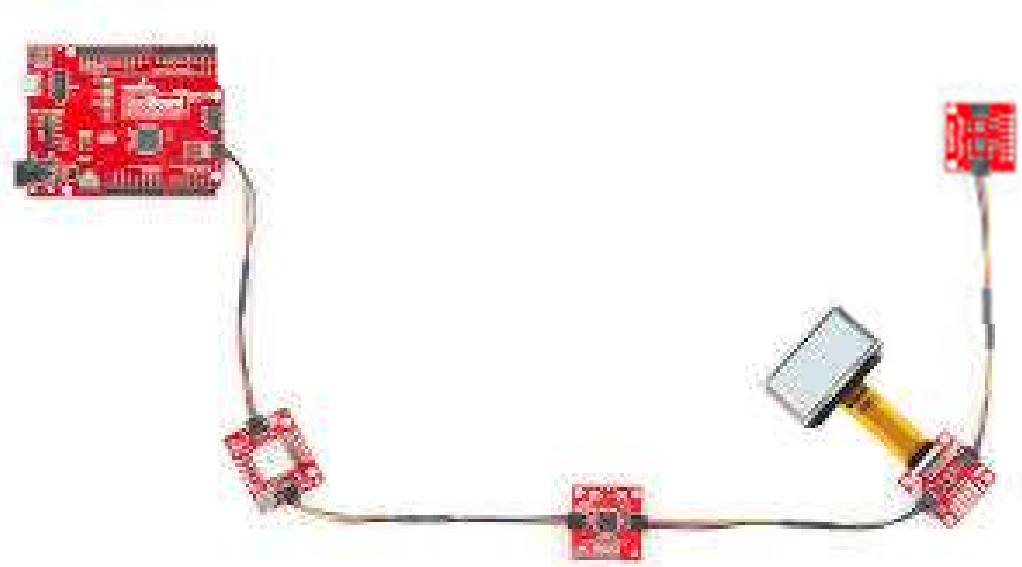
Shields

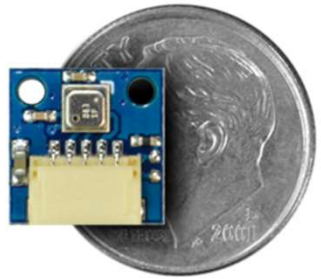
- 370+ Shields[1]
  - Adafruit Industries (9 shields)
  - Sparkfun(26 shields & Qwiic Interface)
  - Seeed Studio (14 shields)
- 125+ Makers
- Some board house have shield templates

[1] shieldlist.org

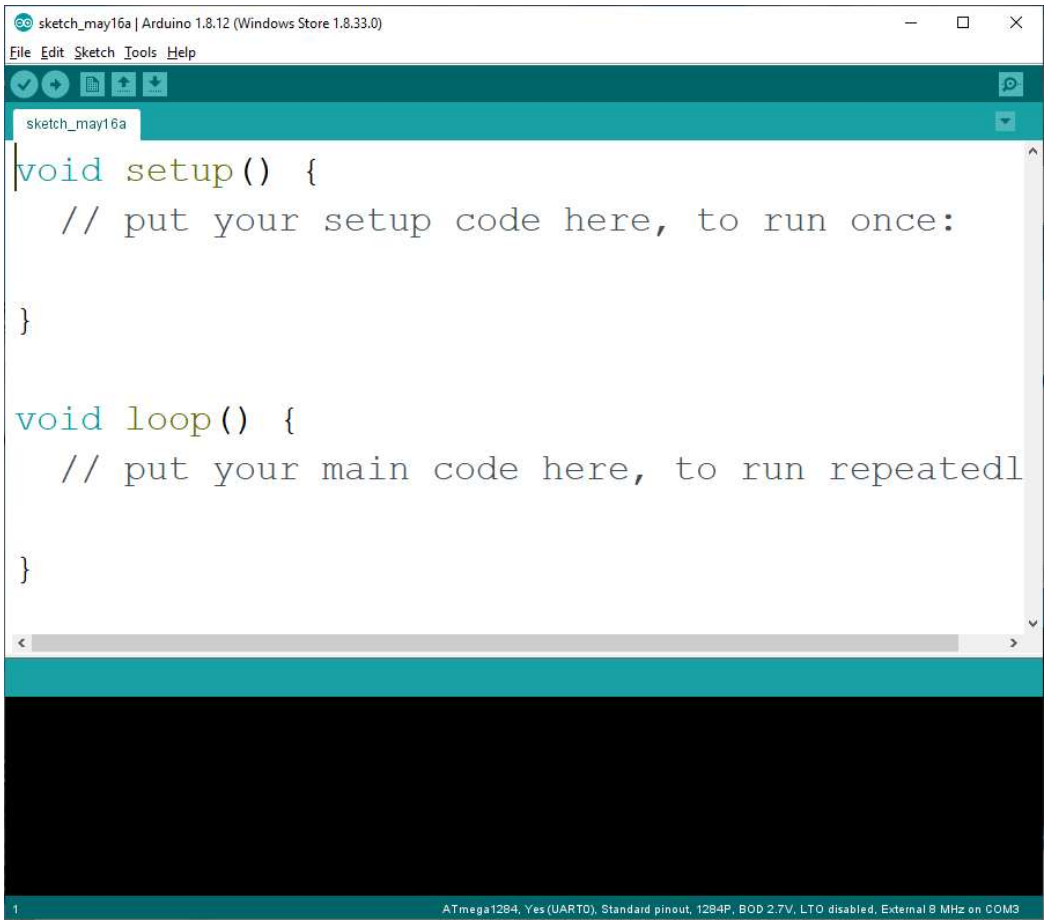
Getting Off The Shield Stack

- Tiny Circuits Wireling
- Sparkfun Qwiic
- Roll Your Own





Integrated Development Environment



Program Template – Blinky Sketch

- Boot loader calls main( )
- Main calls setup( ) once
- Main calls loop( ) in endless loop

```
void setup( ) {
  // put your setup code here, to run once:
}
void loop( ) {
  //put your code here to run repeatedly:
}
```

Polling Blinky

```
void setup() {
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13,HIGH);
  delay(500);
  digitalWrite(13,LOW);
  delay(500);
}
```

- setup( )
  - Configure pin 13 (on board LED) as OUTPUT
- loop( )
  - Write high to pin 13
  - Wait for 500 milliseconds
  - Write low to pin 13
  - Wait for 500 milliseconds

I/O Style -- Polling

- Repeatedly ask peripheral if it has data
- Example: Serial.available( )
- Time wasted either polling status or loop waiting for peripheral to change status

- Serial.readBytes(buffer, length);

- if(Serial.available()){

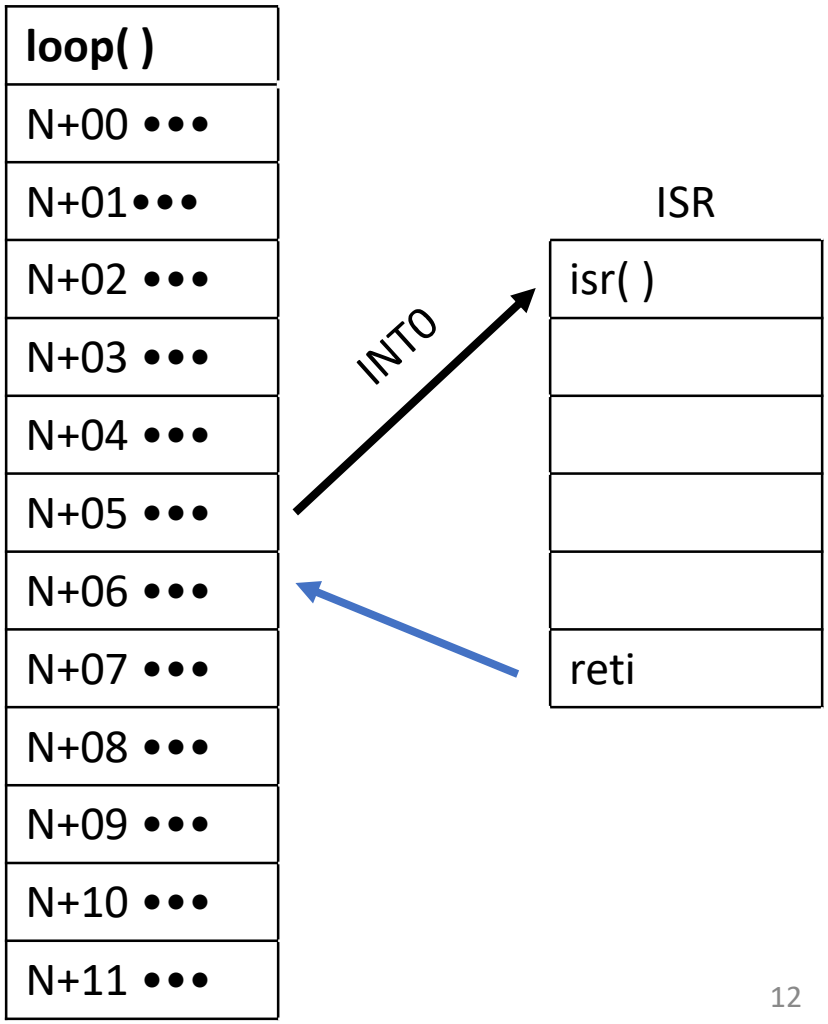
  inByte = Serial.read( );

  •••

  }

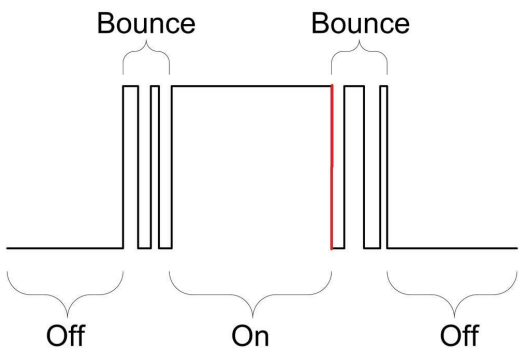
I/O Style -- Interrupt (Hardware Event Driven) I/O

- Peripheral notifies CPU when data available.
- Must return from interrupt service routine clean



Digital Input Interrupt

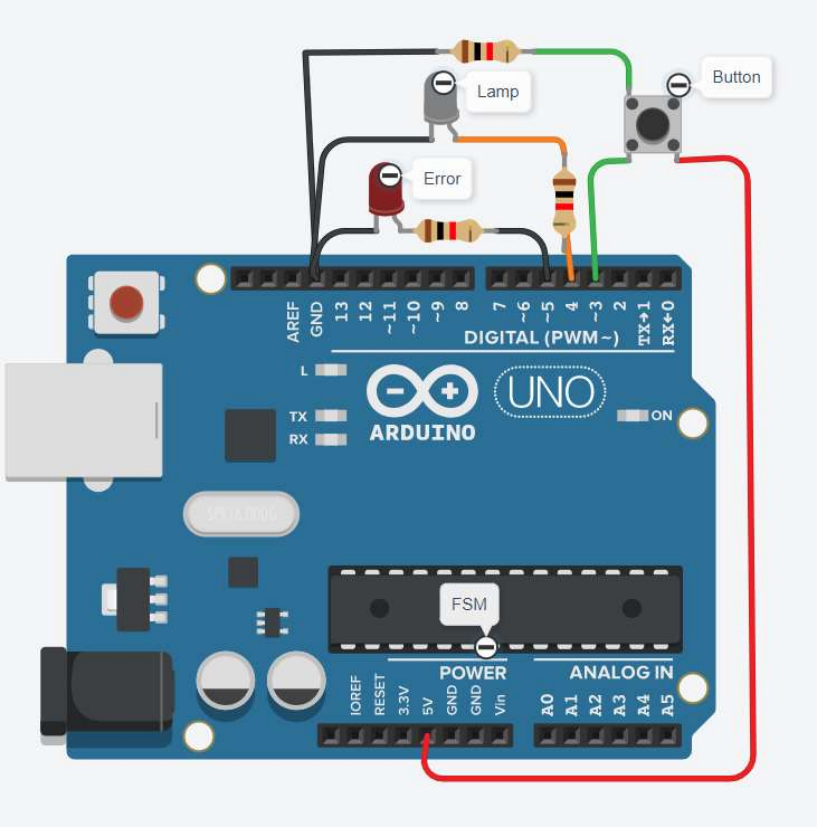
- Hardware called function (subroutine)
- Must associate interrupt service routine with interrupt pin.
- May enable and disable interrupts

- 

- attachInterrupt(digitalPinToInterrupt(INT0), getButton, FALLING);
- interrupts( ) & nointerrupts( )

TinkerCAD.com

- Finite State Machine in loop( )
- Interrupt falling edge



Interrupt Driven I/O

```
// ================================================
// Name: Jim Grover
// Design: Illuminating Example Using If Based FSM
// Date: 11 April 2020
// ================================================

// Define I/O pins
#define LAMP 4
#define ERROR 5
#define BUTTON 3
// Define states
enum stateType{OFF,ON};

// Declare interrupt service routine & flag
void getButton();
volatile bool button = false;
```

Interrupt Driven I/O

```
void setup(){
  // Configure I/O
  pinMode(LAMP, OUTPUT);
  pinMode(BUTTON, INPUT);
  //Attach interrupt service routine
  attachInterrupt(digitalPinToInterrupt(BUTTON),
                  getButton, FALLING);
}
```

Interrupt Driven I/O

```
void loop(){
  // Initialize state variable
  static stateType state = OFF;

  // Finite State Machine
  // case statement specifies presentstate
  // if statements specify next state based on
  // transition arrows
```

Interrupt Driven I/O

```
switch(state){
  case OFF:
            if(button == false){
     state = OFF;
     button = false;
    }else if(button == true){
     state = ON;
     button = false;
    }else{
                        // Indicate error & lock-out
                        while(1)
        digitalWrite(ERROR,HIGH);
    }
            break;
```

Interrupt Driven I/O

```
case ON:
  if(button == false){
        state = ON;
        button = false;
      }else if(button == true){
        state = OFF;
        button = false;
      }else{
  // Indicate error & lock-out
  while(1) digitalWrite(ERROR,HIGH); }
}
```

Interrupt Driven I/O

```
 // update output
  if(state == ON)
    digitalWrite(LAMP, HIGH);
  else
    digitalWrite(LAMP,LOW);
}
```

Interrupt Driven I/O

```
void getButton(){
  // Button is true on falling edge of BUTTON input
  button = true;
  delayMicroseconds(50000); // debounce time
  }
```

Useful Links

- Arduino.cc
- TinkerCAD.com
- YouTube.com
- Fritzing.org