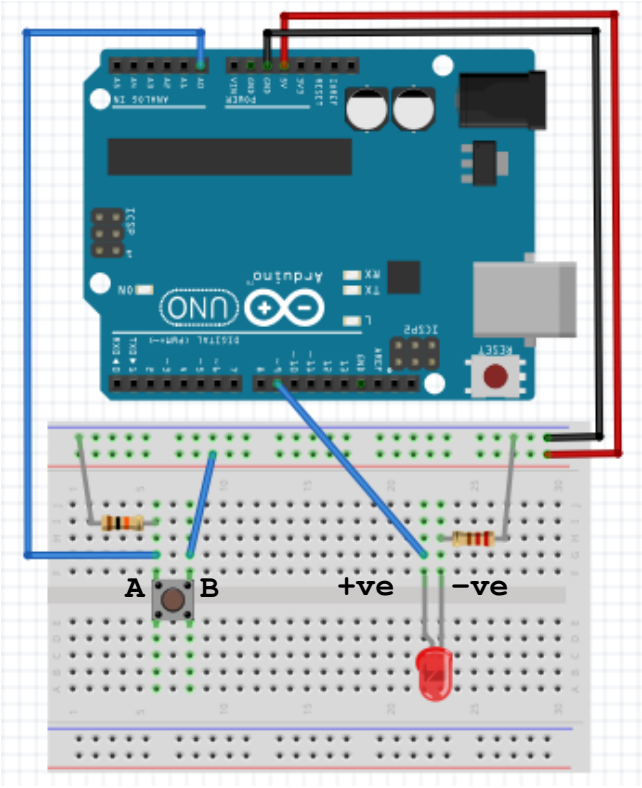
**ATMEGA328/Arduino Uno – BASE INPUT/OUTPUT HARDWARE SET-UP with "SUPER LOOP"**
https://github.com/teaksoon/lmaewapm

## BASE INPUT/OUTPUT HARDWARE SET-UP




**CORE HARDWARE**
1xComputer with Arduino IDE Software
1xUSB 2.0 Type A/B Data Cable
1xArduino Uno Board
Jumper Wires

**INPUT HARDWARE**
*1xTactile Switch
1x10K Resistor

**OUTPUT HARDWARE**
*1x5mm LED
1x220 Ohm Resistor

---

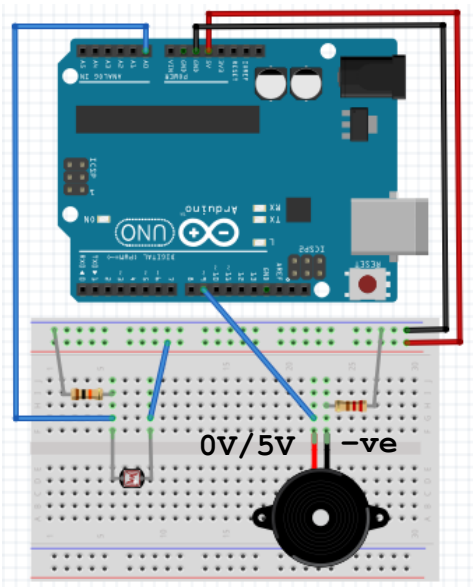
This is our "skeleton" BASE INPUT/OUTPUT HARDWARE set-up, similar to our "skeleton" SUPER LOOP PROGRAM

We will expand from here...

**This BASE INPUT/OUTPUT HARDWARE SET-UP can easily be modified to cater for other I/O hardware, like the ones below**
( most of the I/O devices will be set-up more or less the same )
We can use this design above as our Base

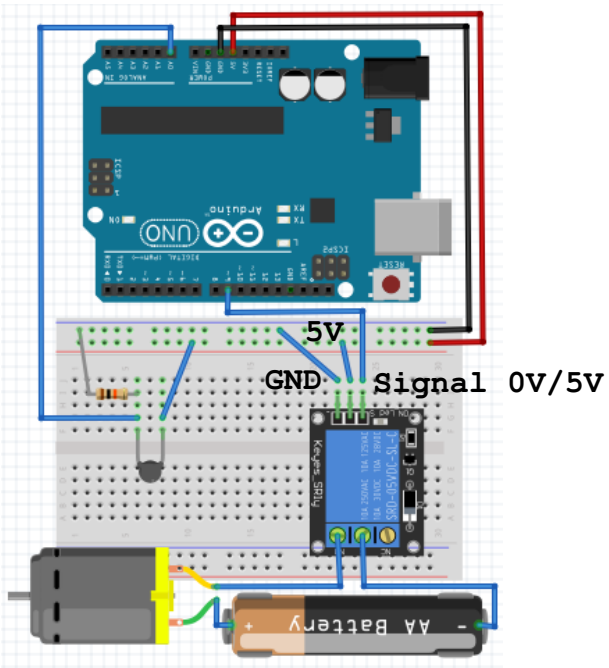
**INPUT HARDWARE**
*Replaced with 1xLDR
1x10K Resistor

**OUTPUT HARDWARE**
*Replaced with 1xActive Buzzer
*1x220 Ohm Resistor(optional)



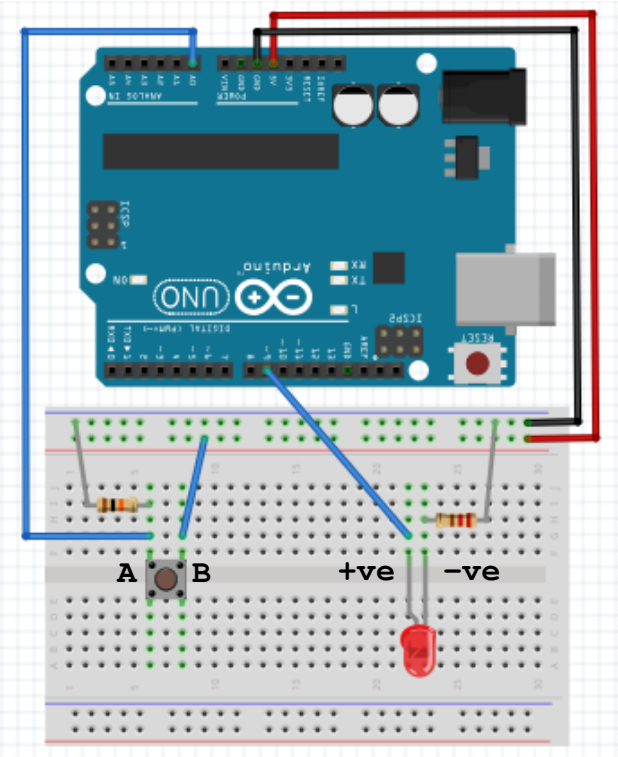

**INPUT HARDWARE**
*Replaced with 1x10K Thermistor
1x10K Resistor

**OUTPUT HARDWARE**
*Replaced with 1xRelay Module
*does not need Resistor



**ATMEGA328/Arduino Uno – BASE INPUT/OUTPUT HARDWARE SET-UP with "SUPER LOOP"**
https://github.com/teaksoon/lmaewapm

We start with our **BASE HARDWARE**




**CORE HARDWARE**
1xComputer with Arduino IDE Software
1xUSB 2.0 Type A/B Data Cable
1xArduino Uno Board
Jumper Wires

**INPUT HARDWARE**
*1xTactile Switch
1x10K Resistor

**OUTPUT HARDWARE**
*1x5mm LED
1x220 Ohm Resistor

---
Switch "A" to A0
Switch "B" to 5V
Switch "A" to 10KOhm Resistor to GND

LED +ve to Pin 9
LED -ve to 220Ohm Resistor to GND

We start with our **"skeleton" SUPER LOOP PROGRAM**

```
void setup(){}
void loop(){}
```

From this "skeleton" PROGRAM, we expand...

**OUTPUT HARDWARE in SUPER LOOP – "BLINKING LED"(**with **blocking-delays)**
This should be easy. The easiest way is use code like the following ( this is the code we normally see all over the place )

```
#define LED_PIN 9

void setup(){
  pinMode(LED_PIN, OUTPUT);
}
void loop(){
  digitalWrite(LED_PIN,HIGH);
  delay(500);
  digitalWrite(LED_PIN,LOW);
  delay(500);
}
```

This is simple and it is working perfectly fine. However, if we also have other hardware in our set-up and depending on what we want to do, we may face a problem. The **delay() function is a "blocking-delays"**, meaning nothing else can run, while the delay() function is not completed

**In this example, the instruction code "delay(500)" will pause the PROGRAM for 500ms, doing nothing.** If we have a Tactile Switch, we cannot process the Tactile Switch button within that 500ms while the delay() function is still running. That will make our Tactile Switch very hard to use

So, we to write our PROGRAM differently, using a **"NON blocking delays"** to do the "BLINKING LED" in our "SUPER LOOP"

**ATMEGA328/Arduino Uno – BASE INPUT/OUTPUT HARDWARE SET-UP with "SUPER LOOP"**
https://github.com/teaksoon/lmaewapm

**OUTPUT HARDWARE in SUPER LOOP – "BLINKING LED"(with NON blocking-delays)**
This is the better way to code for our OUTPUT HARDWARE

Arduino IDE|Save PROGRAM as: **c_super_loop_hardware_output_led**
Enter the codes below and Upload

```
#define LED_PIN 9

int led_state;
unsigned long blink_timer;
unsigned long blink_ms = 500; // blink in milliseconds

void setup(){
  pinMode(LED_PIN, OUTPUT);
  led_state = HIGH;
  digitalWrite(LED_PIN,led_state); blink_timer = millis();
}

void loop(){
  // Start LED Blink process...
  if (millis()-blink_timer > blink_ms ) {
    led_state = (led_state == HIGH) ? LOW : HIGH;
    digitalWrite(LED_PIN,led_state); blink_timer = millis();
  }
  // End LED Blink process...

}
```

Watch the LED. We are getting the same "BLINKING" effect like our regular delay() based "BLINKING LED" PROGRAM.

There are no "pause" anywhere in this "SUPER LOOP" PROGRAM. We can now process the Tactile Switch without the need to wait like the earlier PROGRAM that uses the delay() function

**Keep this code as our "base" for future PROGRAM that uses "BLINKING LED"**
(no need to re-invent this code over and over again)

**ATMEGA328/Arduino Uno – BASE INPUT/OUTPUT HARDWARE SET-UP with "SUPER LOOP"**
https://github.com/teaksoon/lmaewapm

**INPUT HARDWARE in SUPER LOOP – "TACTILE SWITCH"**(with **debounce filter**)
The debounce filter is special for tactile switch, other input hardware may need their own kind of filter

Arduino IDE|Save PROGRAM as: **c_super_loop_hardware_input_tactile**
Enter the codes below and Upload

```
#define TSW_PIN       A0
#define DEBOUNCE_MS   50 // debounce time in milliseconds

int TSW_data;
int TSW_prev_data;
unsigned long TSW_timer;

void setup() {
  pinMode(TSW_PIN, INPUT);
  Serial.begin(9600);
  Serial.println("\nTactile Switch Button is waiting to be pressed...");
  TSW_prev_data = digitalRead(TSW_PIN);
}

void loop() {
int pin_data;

  // Start TSW process ---
  pin_data = digitalRead(TSW_PIN);
  if (pin_data != TSW_prev_data) TSW_timer = millis();
  if ((millis()-TSW_timer) > DEBOUNCE_MS) {
    if (pin_data != TSW_data) {TSW_data = pin_data; process_TSW_btn();}
  }
  TSW_prev_data = pin_data;
  // End TSW process ---
}

void process_TSW_btn() {
  if (TSW_data == HIGH) {
    // code for button pressed
    Serial.println("Switch Button is Pressed Down");
  } else {
    // code for button released
    Serial.println("Switch Button is Released");
  }
}
```

Open the Arduino IDE Software Serial Monitor. Watch the Serial Monitor Screen while Pressing and Releasing the Switch Button

**DEBOUNCE:**
When the switch button has already been fully pressed down or when the switch button has already been released for a period of time, VOLTAGE will stay stable at either 5V or 0V. We do not have problem with them

However, it was during the time, "half-way pressing" or "half-way releasing" of the Tactile Switch Button. We may experience some "VOLTAGE jitters" caused by the mechanical nature. VOLTAGE will simply goes HIGH/LOW randomly within a very short period of time(in micro/milli-seconds). The INPUT Pin will experience many "FAKE button press/release" and that will effect our INPUT Pin readings. That is why we have the debounce filter code above to filter out those unwanted "VOLTAGE jitters"

**Keep this code as our "base" for future PROGRAM that uses Tactile Switch**
(no need to re-invent this code over and over again)

**ATMEGA328/Arduino Uno – BASE INPUT/OUTPUT HARDWARE SET-UP with "SUPER LOOP"**
https://github.com/teaksoon/lmaewapm

**INPUT/OUTPUT HARDWARE in SUPER LOOP – "TACTILE SWITCH"**(with **debounce filter**) **and "BLINKING LED"**(with **NON blocking-delay**)
This is what we will be coding in "real-world" making commercial products with our micrp-controllers

Arduino IDE|Save PROGRAM as: **c_super_loop_hardware_io**
Enter the codes below and Upload. Watch the LED and Press the Tactile Switch

```
#define LED_PIN        9
#define TSW_PIN       A0
#define DEBOUNCE_MS   50 // debounce in milliseconds

int blink_started;
int led_state;
unsigned long blink_timer;
unsigned long blink_ms = 500; // blink in milliseconds
int TSW_data;
int TSW_prev_data;
unsigned long TSW_timer;

void setup() {
  pinMode(TSW_PIN, INPUT);
  pinMode(LED_PIN, OUTPUT);
  TSW_prev_data = digitalRead(TSW_PIN);
  blink_started = 1; led_state = HIGH;
  digitalWrite(LED_PIN,led_state); blink_timer = millis();
}

void loop() {
int pin_data;

  // Start LED process ---
  if (blink_started == 1) {
    if (millis()-blink_timer > blink_ms ) {
      led_state = (led_state == HIGH) ? LOW : HIGH;
      digitalWrite(LED_PIN,led_state); blink_timer = millis();
    }
  }
  // End LED blink process ---

  // Start TSW process ---
  pin_data = digitalRead(TSW_PIN);
  if (pin_data != TSW_prev_data) TSW_timer = millis();
  if ((millis()-TSW_timer) > DEBOUNCE_MS) {
    if (pin_data != TSW_data) {TSW_data = pin_data; process_TSW_btn();}
  }
  TSW_prev_data = pin_data;
  // End TSW process ---
}

void process_TSW_btn() {
  if (TSW_data == HIGH) {
    // code for button pressed
    if (blink_started == 0) {
      blink_started = 1; led_state = HIGH;
      digitalWrite(LED_PIN,led_state); blink_timer = millis();
    } else {
      blink_started = 0; led_state = LOW;
      digitalWrite(LED_PIN,led_state);
    }
  } else {
    // code for button released
  }
}
```