Lab 3: Interrupts & Timers

Tuesday 1/27 2:15pm

Jp Marhefka, Aveed Gorji, & Gwyn Anawalt

Part 1

Program that triggers an interrupt each time the switch is pressed:

```
volatile int counter = 0;
volatile int flag = 0;

void setup() {
 Serial.begin(9600);

 pinMode(2, INPUT_PULLUP);
 attachInterrupt(0, isr0, FALLING);

}

void loop() {
 // put your main code here, to run repeatedly:
 noInterrupts();
 if(flag==1)
 {
   Serial.println(counter);
   flag=0;
 }
 interrupts();
}

void isr0() {
 counter++;
 flag=1;
}
```

You will probably get multiple counts for each button press. Be prepared to explain why:

Initially, when we pressed the button, the counter value was displayed multiple times. After implementing the noInterrupts(); function, we were able to allow the flag==1 if statement to run completely before it could be interrupted again by another “button press,” which would sometimes cause numbers to be skipped.

Part 2

Interrupt code detects the rising edge of the signal at pin 2:

```
int counter = 0;
volatile int flag = 0;

void setup() {
 Serial.begin(9600);

 pinMode(2, INPUT_PULLUP);
 attachInterrupt(0, isr0, RISING);

}

void loop() {
 // put your main code here, to run repeatedly:
 noInterrupts();
 if(flag==1)
 {
   Serial.println(counter);
   counter++;
   flag=0;
 }
 interrupts();
}

void isr0() {
 flag=1;
}
```

Part 3

Generate a timer one interrupt every 100ms:

```
#include <TimerOne.h>
volatile int msCount=0;
volatile int count=0;
volatile int flag=0;

void setup()
{
 Serial.begin(9600);
 Timer1.initialize(100000);
 Timer1.attachInterrupt(timerIsr);
}
void loop()
{
 if (flag==1)
 {
   Serial.print(msCount);
   Serial.println(" ms");
   flag = 0;
   count = 0;
 }
}
void timerIsr()
{
 msCount++;
 if(count < 9 ){
   count++;
 }else{
   flag = 1;
 }
}
```

With pushbutton:

```
#include <TimerOne.h>
volatile int msCount=0;
volatile int count=0;
volatile int flag=0;
volatile int flag2=0;
```

```
void setup()
{
 Serial.begin(9600);
 Timer1.initialize(100000);
 Timer1.attachInterrupt(timerIsr);

 pinMode(2, INPUT_PULLUP);
 attachInterrupt(0, isr0, RISING);
}
void loop()
{
 if (flag==1 && flag2%2==0)
 {
   Serial.print(msCount);
   Serial.println(" ms");
   flag = 0;
   count = 0;
 }

}
void timerIsr()
{
 msCount++;
 if(count < 9 ){
   count++;
 }else{
   flag = 1;
 }
}

void isr0() {
flag2++;
}
```

Part 4

Generate an interrupt on every rising edge of channel A:

```
volatile int B_Value = 0;
volatile int count = 0;

void setup() {
 Serial.begin(9600);

 pinMode(2, INPUT);
 attachInterrupt(0, isr0, RISING);
 pinMode(3, INPUT);

}

void loop() {
 // put your main code here, to run repeatedly:
 Serial.println(count);
}

void isr0() {
 B_Value = digitalRead(3);
 if (B_Value == HIGH){
   count++;
 }
 if (B_Value == LOW){
   count--;
 }
}
```

How many counts correspond to one full wheel rotation: 966

The distance travelled per count:

The wheel diameter is 60mm

Circumference = 188.49mm

188.49mm per count

Part 5

LED on pin 13 blink for 200ms out of every 500ms

```
#include <TimerOne.h>
volatile int B_Value = 0;
volatile float count = 0;
volatile int count2 = 0;
float distance = 0;


void setup() {
Serial.begin(9600);

pinMode(2, INPUT);
attachInterrupt(0, isr0, RISING);
pinMode(3, INPUT);
pinMode(13, OUTPUT);

 Timer1.initialize(100000);
Timer1.attachInterrupt(timerIsr);
}


void loop() {
// put your main code here, to run repeatedly:
distance = (count/966) * 188.49;
Serial.print(distance);
Serial.println("mm");

if (count2<2)
{
 digitalWrite(13, HIGH);
}
else
{
 digitalWrite(13, LOW);
}

}

void isr0() {
```

```
B_Value = digitalRead(3);
if (B_Value == HIGH){
  count++;
}
if (B_Value == LOW){
  count--;
}
}

void timerIsr()
{
count2++;
if(count2 == 4 ){
  count2=0;
}
}
```

Modify the pin 2 interrupt configuration

```
pinMode(2, INPUT);
attachInterrupt(0, isr0, RISING);
```

How many counts correspond to one full wheel rotation: 966

The distance travelled per count:

The wheel diameter is 60mm

Circumference = 188.49mm

188.49mm per count

Problems We Encountered:

For part 3, we initially had the condition be where count < 10, but this actually made our serial output increment by 11. But then we realized this was the case because we had an else that would change the flag. This made it 10+1, and 11. So we changed the condition to be count < 9 and it worked properly.

For part 4, we were unsure of the logic of what pins are analog/digital read/write. We re-read the instructions and realized that pin 2 is actually supposed to be digitally reading and not writing. After this, it worked properly.

For part 5, we accidentally used the diameter of the wheel and not the circumference. This caused our value to be approximately divided by pi. We also had an issue with reusing the same count variable for both interrupts and this made it so the LED would stop flashing after the wheel started moving. So we made the count2 variable. Once we fixed these errors, everything worked properly.