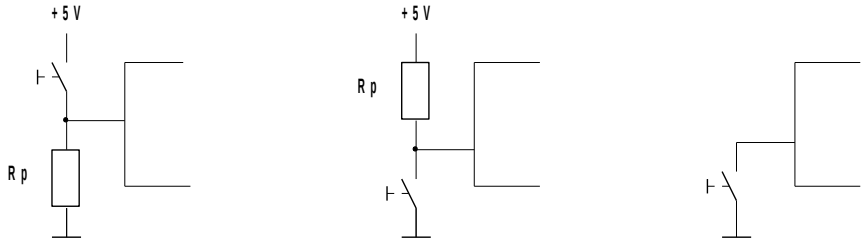
Arduino notes 3: Digital input via buttons

This is not a tutorial, but a collection of personal notes to remember the essentials of Arduino programming.
The program fragments are snippets that represent the essential pieces of code, in an example.
jean-claude.feltes@education.lu

Static input via button

There are 3 possible switch configurations: one with pulldown and two with pullup resistor:



It is a good idea to connect the switch **button to ground**, as in this case the internal pullup resistance can be used.

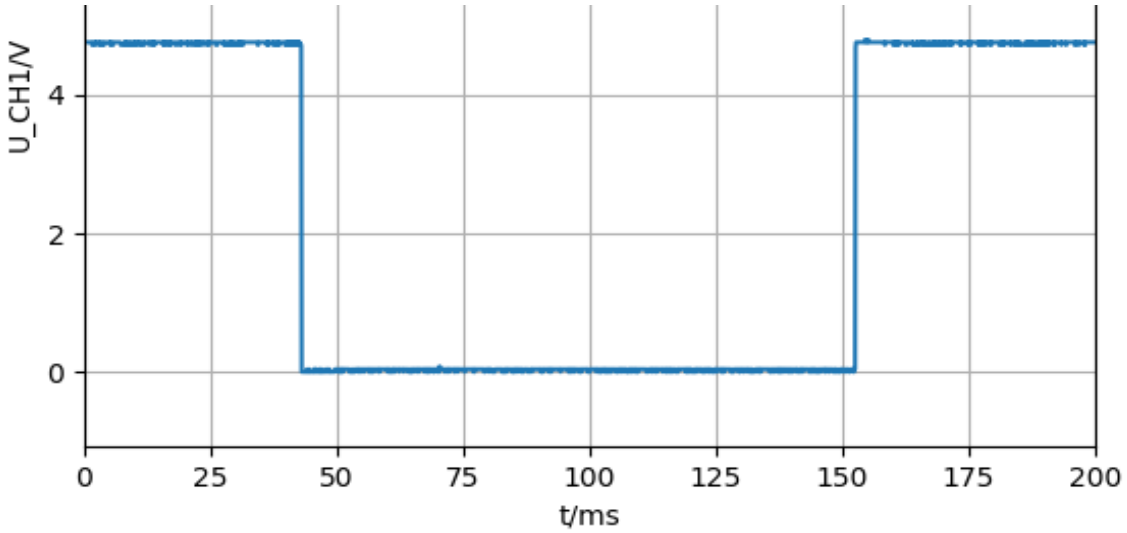
Remember that an active switch gives a digital 0!

The folowing example switches a LED on as long as the button is pushed:

```
const int button=7;
const int led=13;

void setup() {

  pinMode(button, INPUT_PULLUP);
  pinMode(led, OUTPUT);
}

void loop() {
   int buttonval = digitalRead(button);

  if (buttonval == 0) {
    digitalWrite(led, 1);}
  else {
    digitalWrite(led, 0);}
}
```

Counting button presses: Bouncing and debouncing

Oscillogram of a typical button press:



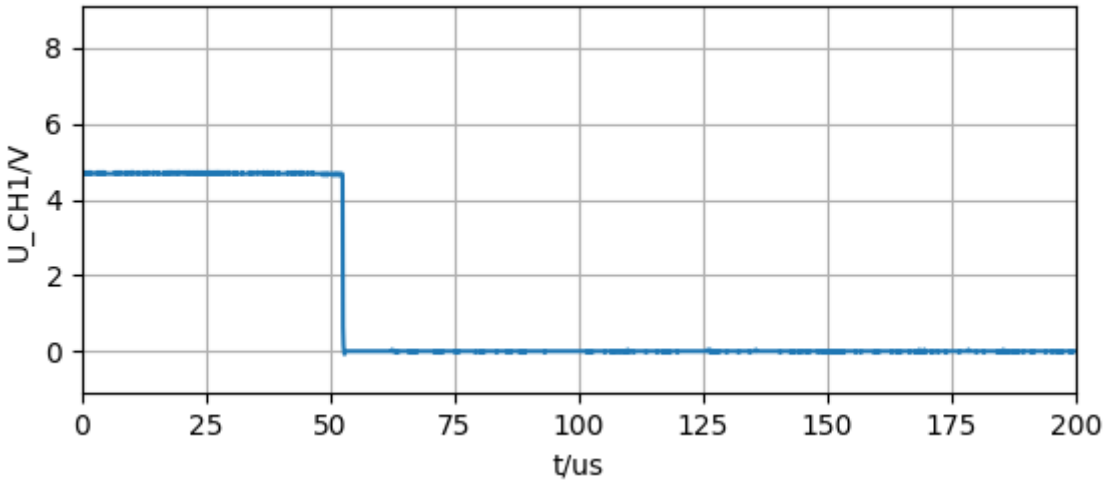
The length of the low time is typically 100 to 200ms. (It is difficult to get shorter times). During this time, the controller must wait until the button is released.

A sample program to count pulses, with transmission of the count via serial interface:

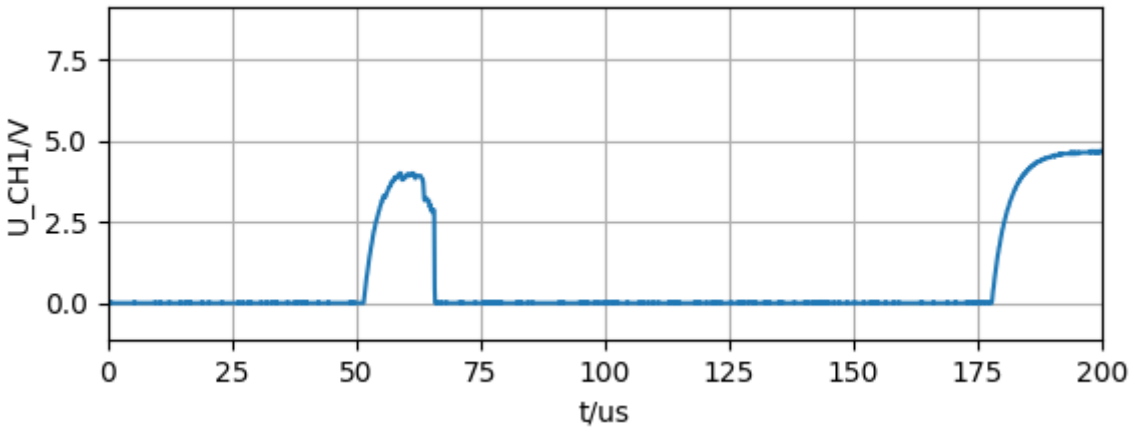
```
const int button=7;
const int led=13;
int x=0;

void setup() {
  Serial.begin(115200);
  pinMode(button, INPUT_PULLUP);
  pinMode(led, OUTPUT);
}

void loop() {

  if (digitalRead(button) == 0) {
      delay(10);
      // wait until button is released
      while (digitalRead(button)==0){
       delay(4);
       }

      // react to button pushed
      Serial.println(x);
      x+=1;}
}
```

When testing this program, I was prepared to see some bouncing effects, but there seemed to be none. Maybe the delay of the Serial.println was enough to debounce? Or would I have such a fantastic switch, with no bouncing?

In fact the push phase didn‘t show any bounces (even at higher time resolution):



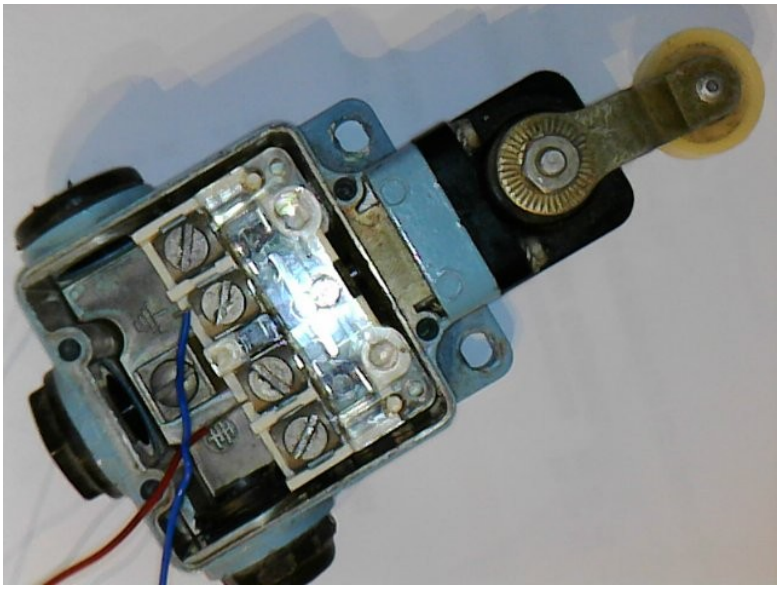
For the release phase it was mostly similar, only one of many tests showed real bouncing:



As the bouncing period here is about 120us, the effect is neutralised by the delay of the println.

So, this little cheap switch does a better job than expected!

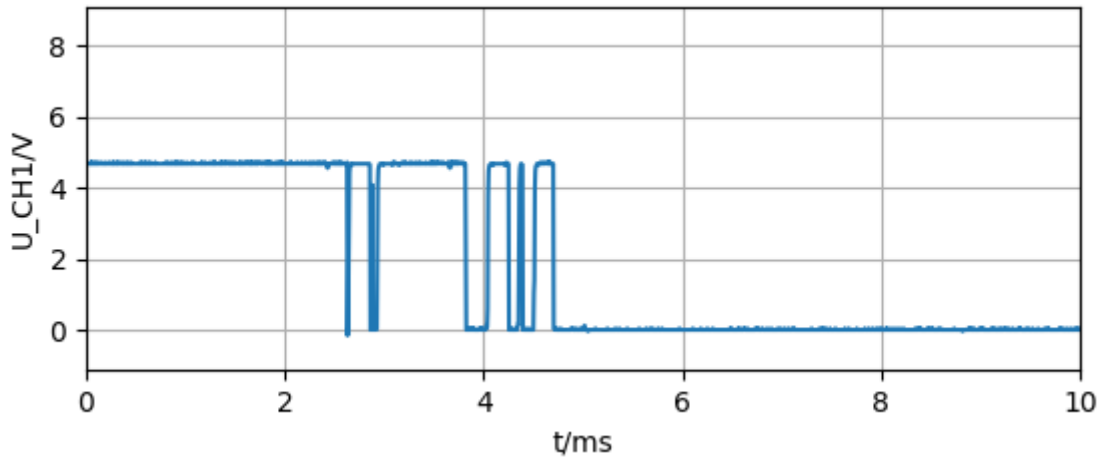
Small switch behaving well



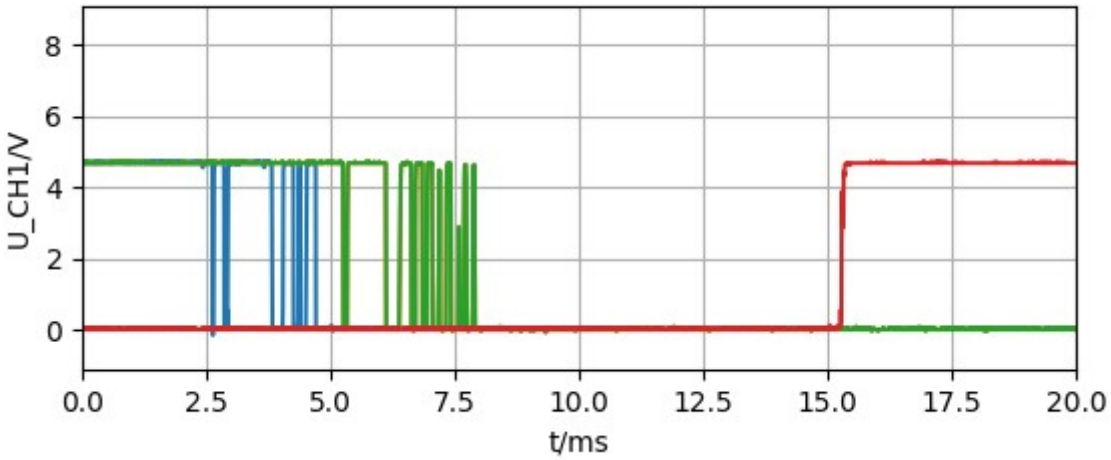
Industrial endswitch with really much bouncing

How would an industrial end switch behave?

Well, if I would have wanted to demonstrate bouncing, this would have been a good choice:



Several push / release events:



And also, the serial monitor shows multiple counts for one button push.

With this switch, a debouncing strategy is clearly needed!

As we see, the bouncing occurs mainly when the input goes from high to low (though we cannot be shure to have none during the low to high switching, and in fact there seems to be some, only much faster).

The strategy is the following:

- When the input first goes 1→ 0:
  wait ca. 10ms until the bouncing has terminated
- Loop until the input goes high, with a small delay in the loop that works against the 0→ 1 bouncing.

```
void loop() {

  if (digitalRead(button) == 0) {
      // anti-bouncing delay:
      delay(10);
      // wait until button is released
      while (digitalRead(button)==0){
       delay(4);
       }

      // react to button pushed
      toggle_led();
      Serial.println(x);
      x+=1;}
}
```

I tested this with the „bad“ switch, and 100 of 100 pushes were counted right.

Eventually the two delay times must be varied if the used switch has longer bouncing times.

This code fragment uses a small function to toggle the led. This is very practical to visualise the correct function:

```
void toggle_led(){
    if (digitalRead(led) == 1){
      digitalWrite(led,0);
    }
    else{
      digitalWrite(led,1);
    }
}
```

A button read function with debouncing

To modularise, we can put the debouncing code into a function buttonPushed that returns true when the button is pushed:

```
boolean buttonPushed(int button){
     if (digitalRead(button) == 0) {
          // anti-bouncing delay:
          delay(15);
          // wait until button is released
          while (digitalRead(button)==0){
               delay(5);
          }
          return true;
     }
     else
```

```
            {return false; }
}

void loop() {
      if (buttonPushed(button)==true){

            toggle_led();
            Serial.println(x);
            x+=1;}
}
```

This function can also be used for other inputs, as button is a variable.

Eventually the timing must be adapted to the used switch.

Arduino libraries for debouncing

See under Sketch – Include Library – Manage Libraries

There are several libraries to be found, but in my eyes they are not as straightforward to use as the one in BASCOM.

An article on debouncing is found here:

https://www.arduino.cc/en/Tutorial/Debounce

The method is slightly more complicated than the one described here. I don‘t know if it gets better results.