AtomPalm, Inc.
Initial Release: 2020-10-29
Revision 2
2020-10-30

Switches in High Performance Peripherals

Dimitar Dimitrov, Jorel Lalicki
AtomPalm, Inc.

Abstract. In an ongoing effort to improve human performance in gaming among other demanding environments, we look at minimizing the technological barriers between the human and the subject. Our research explores the full breadth of variables that need consideration to optimize switches for low latency and identify factors that may reduce longevity. Based on the information, we draw conclusions and define a set of industry best practices.

Keywords: latency, switches, gaming, performance, peripherals, eSports

1. Introduction

The growing popularity of video games is in no small part due to eSports. In general, games are becoming more demanding of both the hardware and the player. To keep up with the demands, it is necessary to consider the factors that can cause mechanical, electrical, and latency issues. Precision is critical when it comes to clicking, and the latency is inimical to precision. This paper explores all the different aspects of switches. **Figure 1** shows the basic anatomy of an Omron Japan D2F-01F switch. *Zero Latency Switch*^{TM} is the term we use to describe a peripheral switch which is optimized for latency, meaning a switch with zero time-dependent elements as part of the circuit.

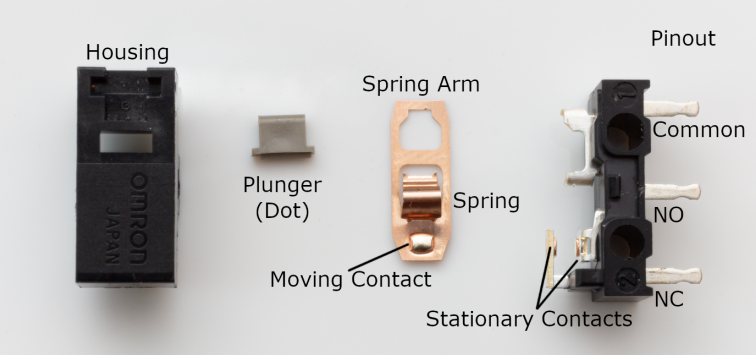

Figure 1. The anatomy of an Omron Japan D2F-01F switch

*dimitar@atompalm.com

2. Electromechanical

2.1. **Circuit.** There are two sets of contacts in a switch. *Stationary contacts*, also known as the *throw* contacts, refer to the fixed contacts on the *NO* and *NC* pins. *Moving contacts*, also known as the *pole* contacts, refer to the contacts moved by the spring, which alternate electrical connectivity with the *NO* or *NC* pins.

There are two common types of switch circuits to consider: **S**ingle **P**ole **S**ingle **T**hrow (SPST) and **S**ingle **P**ole **D**ouble **T**hrow (SPDT). In an SPST, the switch state is defined by a single output throw pin. In an SPDT, the switch state is defined by two output throw pins. In other words, in an SPST the *NC* pin from **Figure 1** is not a valid electrical connection - only the *NO* pin can be used to create the circuit. In an SPDT, both the *NO* and *NC* pins are valid and can be used to create the circuit.

It is important to emphasize that while many switches are technically capable of functioning as SPDT, they are rated as an SPST and should be used as such. In general, the reason for this is the lack of NC contacts on the switch. **Figure 2** shows a switch without the NC contacts. Using a switch which does not have NC contacts as an SPDT may initially appear to work but will result in unreliable and uncharacterized behavior.

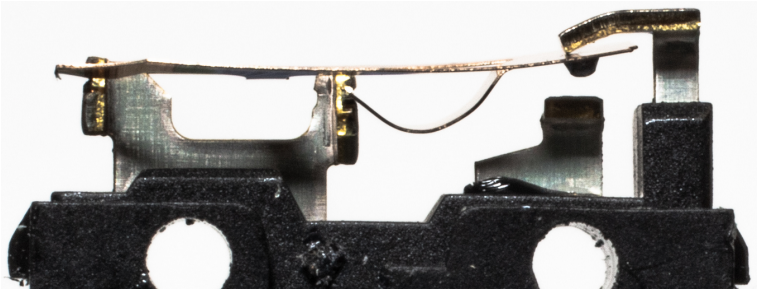

Figure 2. An Omron D2FC-F7-N switch.

2.2. **Wetting Current & Surface Material.** The wetting current is the minimum electric current to break through the surface film resistance on a contact. If wetting current is not taken into account, the switch may become unreliable over time. Most switches have some contact drag, which can mechanically reduce the wetting current by scrubbing off the surface oxide layer. This is only acceptable for switches which have a large amount of contact surface material with similar metals between the contacts. Switches which have contact drag without the proper surface will experience significant wear on the contact surface; This wear increases the wetting current and will ultimately break the switch altogether. In order to maximize the longevity of a switch, the contact drag should be minimized.

The surface material of the contacts is a large factor of the wetting current and the electrical integrity of the switch over time. Switches such as the Huano “white dot” / “red dot” which use dissimilar metals on the stationary and moving contacts will cause the softer metal to wear down quickly, which will compromise the function of the switch.

There are countless surface material options. Most commonly we see copper, nickel, ruthenium, silver, and gold alloys. The electrical and the mechanical properties need to be considered when choosing the material. Gold is the optimal surface material due to its minimal reactivity. Peripheral switches are usually used in a moderately humid and dusty environment, so reactivity is a moderate concern. The more reactive the material, the more a surface oxide layer will develop, and therefore the higher the wetting current requirements will be. Especially in low contact force

switches, designing for wetting current requirements is crucial to obtaining long term performance and reliability.

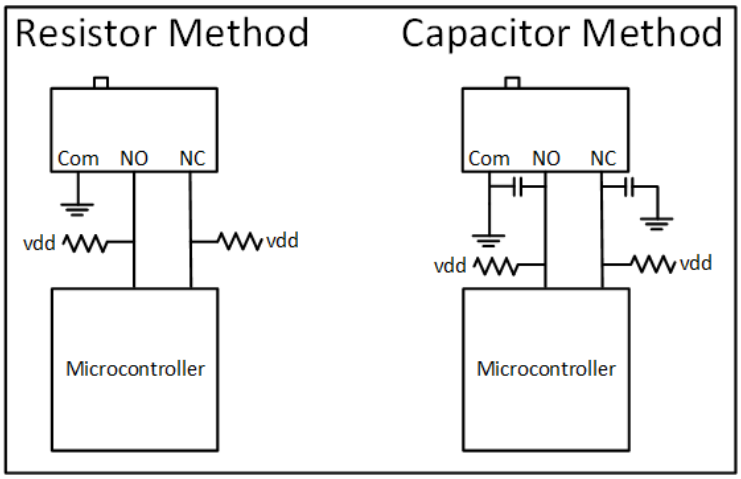

Figure 3. Example resistor and capacitor circuits to address wetting current issues in an SPDT switch.

The trend in modern microcontrollers is to use weak pull up resistors on the GPIO pins. Usually, the current output is significantly below the wetting current the switch is rated for. **Figure 3** shows two example circuits as potential solutions for meeting the wetting current requirements; The resistor method typically requires a resistor with a lower value than the one internal to the microcontroller. The resistor method is simple because the power calculations can be done using Ohm's law $V = IR$. For a battery-powered device, this will generally result in an unreasonably high amount of current being "wasted" in the switches, especially in an SPDT switch, where this current would be active all the time. For an SPST switch, the resistor method is recommended if the appropriate wetting current can be achieved, and the duty cycle of switch presses is relatively low (switch is expected to be pressed in short clicks, rather than held for long periods of time). The capacitor method adds a capacitor to store charge. When the switch is open, the capacitor charges through the pullup resistor until it is at VDD. When the switch closes, the charge stored in the capacitor rushes through the switch contacts, providing a substantial amount of current (enough to meet wetting current requirements).

Selecting an appropriate capacitor for this method is not simple to calculate, as the current is very brief and is limited by the impedance of the traces, solder junctions, component leads, etc. Typical values for computer peripherals are expected to be in the range of 10nF to 1uF, however, calculating an "ideal" value is impossible, as the capacitance only impacts the overall amount of energy stored rather than the peak current. To calculate the peak current delivered, you again use Ohm's law, and the peak current will be when the capacitor is fully charged ($V = VDD$). Larger capacitance results in a "longer" pulse of wetting current. Most switches do not have a manufacturer-recommended minimum wetting current duration, so this value is somewhat arbitrary, although increasing capacitance serves to increase power consumption and (at an extreme) limit switching speed. Low ESR capacitors should be selected, and care should be taken to locate the capacitor close to the switch, and with sufficient trace width, to minimize unnecessary impedance that would limit current flow. For an SPDT switch, the capacitor method is recommended because it minimizes the continuous power draw, as the wetting current is only reached as the capacitor discharges. These methods can also be used in conjunction, if necessary.

2.3. **Contact Shape.** Some switches, such as those in the Omron Japan D2F series, use a large crossbar style contact where the diameter matches the deflection of the spring arm. This effectively changes the contact overtravel motion from a drag to a roll - significantly reducing wear on the contacts while still maintaining the benefits of scrubbing. **Figure 4** Shows various different moving contact styles commonly used.

Some switches do not have stationary contact points at all. They simple rely on the conductivity of the *NO* pin's inner structure. The entire Omron D2FC series of switch is guilty of this and can be seen in **Figure 2**. The improper contact surface causes asymmetric wear on the moving contacts, which partly leads to the double-clicking issues commonly found in mice using these switches.

Switches such as the Kailh GM 6.0 use a grid of pyramid-shaped contact points instead of a single smooth surface. The purpose of such a design is to reduce bouncing of the contact. From an electrical standpoint, there is no issue with such a design since, even on smooth contacts, the surfaces will only meet at asperities as modeled in Liu et al. (2013). From a mechanical standpoint, there is less material on the contact overall, so it will wear down more quickly. The same surface material considerations stated in the previous section apply to this style of contact as well.

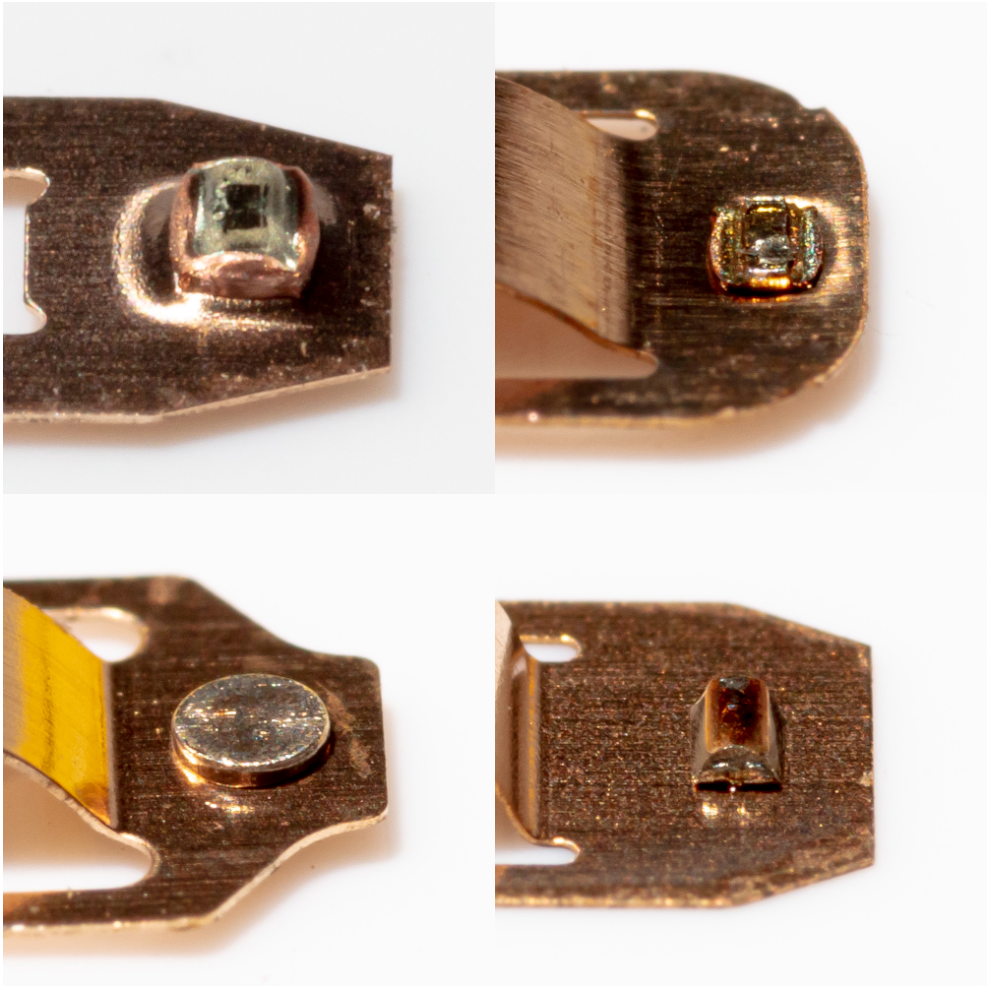

Figure 4. A comparison of the moving contacts on various switches. Top Left: Omron D2F-01F, Top Right: Huano Red Dot, Bottom Left: Kailh White Dot, Bottom Right: D2FC-F-7N

2.4. **Springs.** The feeling of a switch depends almost entirely on the spring. As pressure is applied to the plunger, it will begin to depress. The distance the plunger depresses prior to the actuation

is called the *pretravel*. The force on the switch at the moment of actuation is called the *operating force*. The distance the plunger is allowed to move after the actuation is called *overtravel*. The minimum amount of force necessary to keep the plunger depressed is called the *releasing force*.

While these are all properties of a switch, the peripheral housing can be used to fine tune them. However, it is not recommended unless there is very careful consideration of tolerances. The switch should be the primary focus to achieve the desires button feel characteristics.

The most important characteristic is the operating force. The force should be high enough so that no accidental clicks are made but minimized otherwise. The mouse grip style which creates the highest resting force on a switch, is a palm grip with relaxed fingers. From our tests, the average finger in this configuration weighs approximately 20 ± 10 grams - This number should be taken only as an estimate. It is highly dependent on switch position, mouse shape, hand size, etc. The pretravel and overtravel are the characteristics responsible for a switch feeling "mushy", and should be minimized.

2.4.1. *Spring Types.* The Honeywell ZX10C series uses a helical tension spring which moves a contact plate. The primary benefit of this design is that there is no contact drag or roll, thereby reducing wear and increasing the longevity of the switch.

The Omron Japan D2F series uses a leaf spring with a 2 part construction. The contacts are attached to a flat metal arm, which is used as a mechanical advantage against a leaf spring. This decreases the mechanical fatigue on the spring and allows for a tighter tolerance on the operating force.

The most common construction, found in the Omron D2FC series among many others, uses a leaf spring made from a single piece of stamped metal.

2.4.2. *Mechanical Latency.* There is a minimum latency inherent to every switch. It is the amount of time it takes for the moving contacts to travel the distance between the two stationary contacts on actuation. This latency is a function of the stationary contact gap width, the moving contact width, the input force, and many aspects of the spring such as tension. **Table 1** shows some data comparing the maximum mechanical latency across various switches. While the measurements varied significantly between different switches, this measurement is not particularly relevant to computer peripheral performance. In 'typical' usage, the actuation force is much higher, resulting in virtually identical mechanical latency for all the switches tested (approximately 1500μs, although this is very dependent on technique and individual user).

(in μs)	**D2F-01**	**D2F-01F**	**D2FC-F-7N**	**Huano Green Dot**	**Black TTC**
Average	3715.5	3334.5	6032.3	2619.9	6609.2
Std Dev	427.8	483.3	998.0	345.4	1676.3

Table 1. Contact gap travel times for new switches using the minimum actuation pressure. *data was collected with 20 samples of a single switch using a logic analyzer recording 2048 samples at 400kHz, measured between the rising edge on the NC contact and the falling edge of the NO contact. Note that not all switches tested have NC contacts, potentially resulting in some additional variability in data.

2.5. **Bouncing.** When a switch is actuated, there is a physical motion that has to occur for the switch to change state. Since the spring moves the contacts very quickly, when they first meet, the moving contact literally bounces off the stationary contact. The bouncing period describes this state of transition after the initial connection until the energy has been dissipated, and the moving contact has settled. **Figure 5** shows how the physical latency characteristics are exhibited

between the two contacts on the switch. **Table 2** compares data on bounce timings from various new switches.

(in μs)	**D2F-01**	**D2F-01F**	**D2FC-F-7N**	**Huano Green Dot**	**Black TTC**
Average	757.0	771.0	687.0	629.8	998.9
Std Dev	83.7	87.2	79.9	67.3	225.9

Table 2. Bounce times of new switches. *data was collected with 20 samples of a single switch using a logic analyzer recording 2048 samples at 400kHz, triggered on the falling edge of the NO contact.

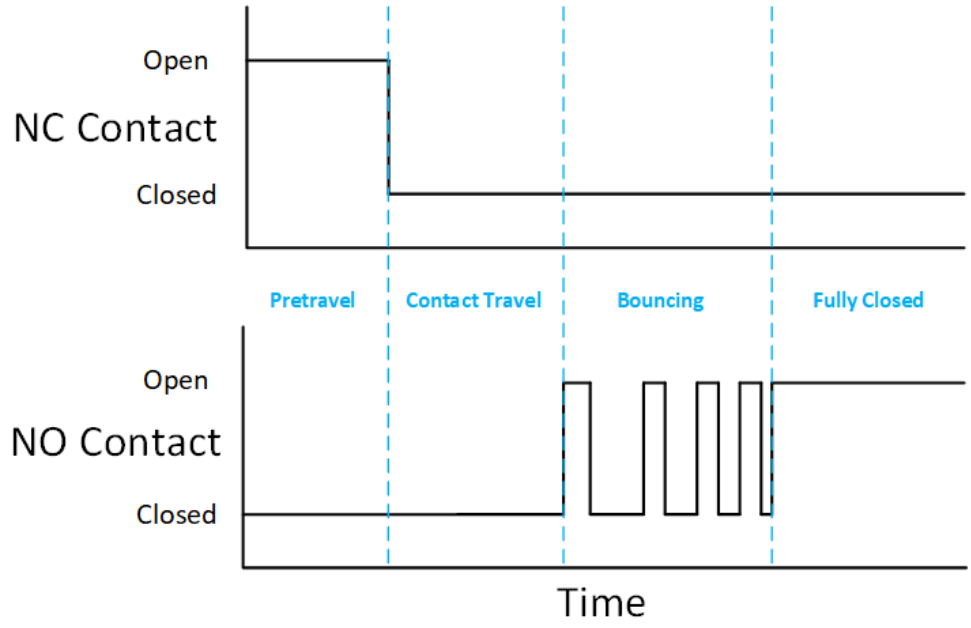

Figure 5. Physical latency characteristics of a switch

From the perspective of the microcontroller, there is no difference between multiple bounces and multiple clicks. If bouncing is not handled properly, multiple clicks are transmitted on a single actuation of the switch. As a switch ages, the stiffness of the spring is reduced, and the contact surface wears down. Due to this, the bouncing characteristics will also change. It is important to take these factors into account when addressing bouncing. See the Click State Logic section for information on how to properly handle bouncing.

2.6. **Optical Switches.** Another type of switch which has recently gained some popularity is an optical switch. Optical switches typically function similarly to garage door safety mechanisms. On one side of the switch, there is a light emitter (typically a LED). On the other, there is a light sensor (typically photodiode or phototransistor) - when the switch is actuated, the spring moves to either obstruct or unobstruct the line of sight between these components.

In one popular design shown in **Figure 6**, when the switch is actuated, the spring moves a plastic window to unobstruct the line of sight. The motion is not stopped instantly, it bounces similarly to its non-optical counterparts. However, the small amount of bouncing which occurs has virtually no effect on the signal from the light sensor, as the window is substantially larger than the amplitude of any potential mechanical bouncing.

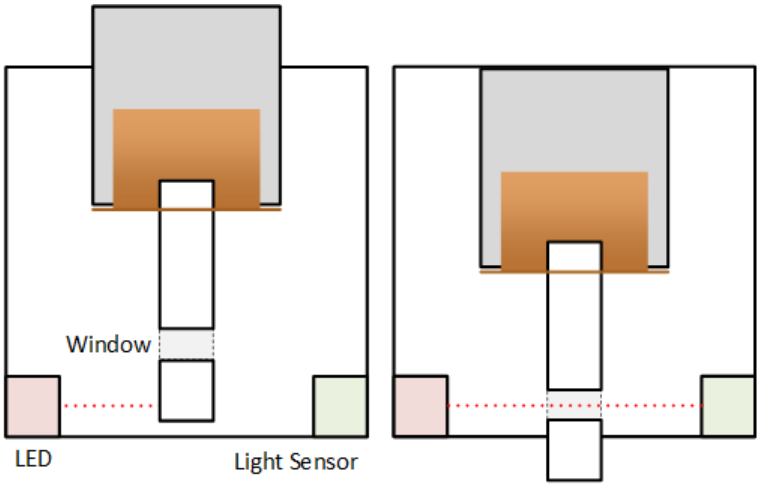

Figure 6. Optical switch design using a moving window to unobstruct light

The moving window could also be replaced with a moving prism or mirror, effectively accomplishing the same task of allowing light to reach the sensor when the switch is actuated. An example is shown in **Figure 7**

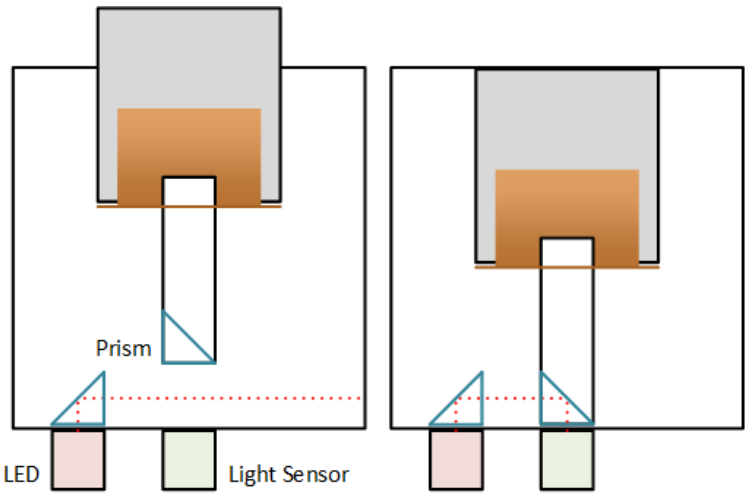

Figure 7. Optical switch design using a moving prism to redirect light

Typical designs utilize a light sensor and light emitter mounted directly to the device PCB, with a purely mechanical "switch" body mounted above them. The switch body does not have any electrical properties, so solder joints are unnecessary. This theoretically has an advantage in terms of clean-ability, upgrade-ability, and it eliminates the potential failure point of the electrical leads/solder connection on a traditional switch at the cost of mechanical stability.

Another potential advantage of optical switching is the ability to read analog position (with the correct supporting electronics). The signal from the light sensor is analog. As the window slides open (or closed, depending on configuration), it is receiving a changed amount of the light from the photodiode. Analog switching is likely more applicable to a keyboard, where it would allow a user to adjust the threshold for registering a keypress, allowing for an adjustable pretravel. These benefits are not likely to affect the average user and are of more interest to the modding/enthusiast

communities who may want to customize their peripherals. Additionally, on a computer mouse, the overall travel is short enough to render the adjustable pretravel essentially useless.

Most optical sensors consist of photodiodes or phototransistors, which output current that varies with the amount of light hitting the sensor. Depending on the photodiode and slew rate desired for switching, the parasitic capacitance can be a limiting factor. In order to reduce this capacitance, the photodiode is generally held in reverse bias, which adds the complexity and cost of additional electrical circuit components - typically operational amplifiers. A high-performance optical sensor may consist of a photodiode held in reverse bias, connected to a transimpedance amplifier, which converts the current mode signal to voltage. This voltage can then be compared to an analog reference voltage (with a comparator/op amp) to obtain a high bandwidth digital signal with an accurate threshold. For most anticipated computer peripherals, however, this approach yields little benefit for the increased cost and complexity.

The simplest optical sensors are simply a phototransistor (PN transistor that is activated by light), connected through a high-value resistor to ground. As light hits the phototransistor, current flows through the resistor, creating a measurable voltage signal. This simple solution can be read as a digital signal by a microcontroller - the "on" signal voltage can be set to some extent by changing the value of the resistor. Photodiodes can also be read in this manner, but the gain is substantially lower, and the maximum voltage obtainable may be too low to be read as a digital signal.

RC filtering is not needed for optical switches, both due to the already-present parasitic capacitance of the optical sensor (which behaves like an RC filter when used with a series resistor to create a measurable signal) and due to the mechanical attributes of an optical switch that result in the switch opening/closing slowly over a period of time, rather than instantly at the point of contact.

Unlike mechanical switches, optical switches use power continuously. Lower output light emitters use less power but require higher gain light sensors and increase the likelihood of signal to noise issues. In a battery-powered device, this can be a substantial design constraint, as maximizing battery life is generally a priority. There are many strategies for minimizing power consumption by optical switches. One is simply to make sure the light path is well shielded from external interference. By preventing stray light from hitting the light sensor, we minimize the amount of external noise being detected, which allows use of a lower intensity light emitter. Most photodiodes used for switching are provided with wavelength filters to block visible light - although you could use any wavelength for the switching, provided the emitter and sensor are both at the same wavelength and that appropriate measures are taken to avoid interference from external light sources. Another strategy for minimizing power consumption is to simply not have the emitter on all the time. With a high bandwidth system, the optical sensor and emitter can be operated at very high frequency and at a low duty cycle. For example, instead of the emitter being on continuously, it could be pulsed at 8kHz, with a duty cycle of 10%. This would result in an approximate power savings of 90% by the switch, although the microcontroller is potentially going to draw more power to deal with the increased number of events. A more advanced system would dynamically alter the switching rate based on usage - after sitting idle for a period of time, a peripheral could revert to a lower read frequency for optical switches (for example, 200Hz). At this low frequency, with the same time period active as the above 8kHz example (12.5ns), power consumption is drastically reduced by 99.75% of that of a continuously on solution.

3. Click State Logic

3.1. **Resistor-Capacitor Filters.** Also known as an RC Filter, is a time-dependent solution to debouncing - meaning there is a minimum amount of delay built into the circuit between the physical actuation of the switch and when the microcontroller will receive a click or a release trigger. **Figure 8** shows a basic RC filter circuit as well as an example of the electrical characteristics exhibited by such a circuit.

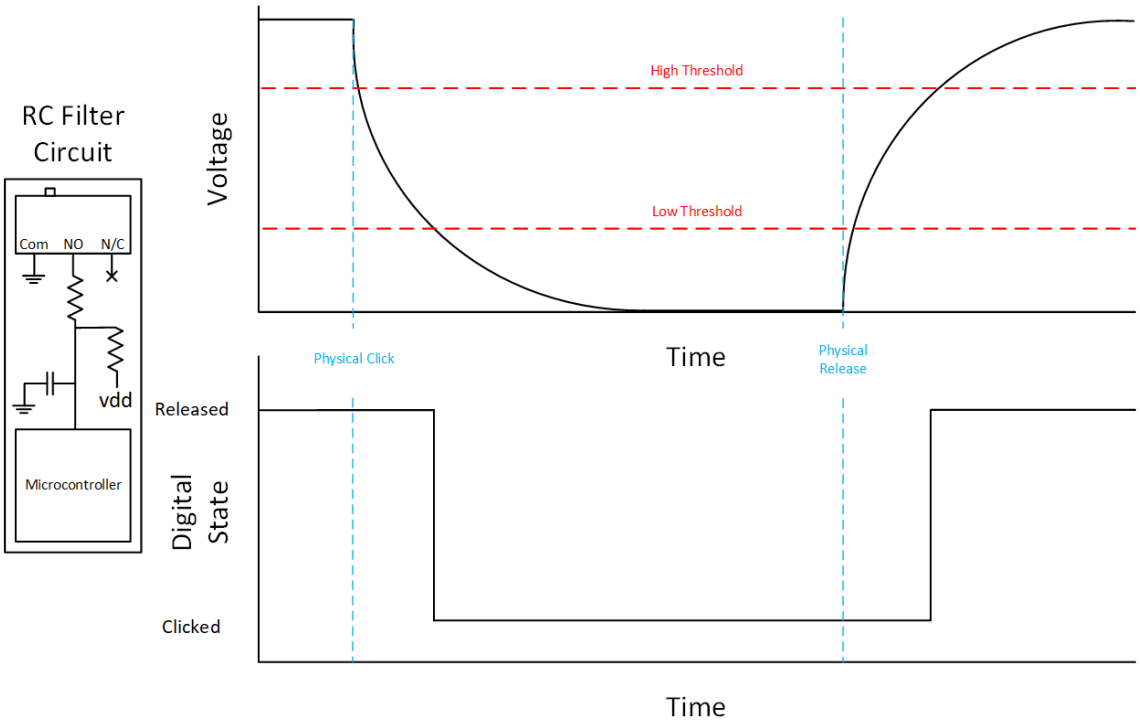

Figure 8. Electrical characteristics of an RC filter

The desired time delay must be defined in order to calculate the appropriate values for the resistor and capacitor. The time delay should be long enough so that all bouncing is contained within it, but minimized otherwise. The following equation gives a simplified method for calculating the corresponding values. (Where V_o is vdd, and $V(t)$ is the microcontroller's state voltage threshold)

$$R = \frac{t * V_o}{C * ln(V(t))} \tag{3.1}$$

One caveat to this equation is that it does not account for the periods of no electrical connection during the bouncing. The capacitor will fluctuate between charging and discharging, thereby increasing the required time value significantly. As stated in the Bouncing section, the characteristics of the bounce change over time, so this must be accounted for as well. The time value should be chosen to correspond to the least optimal conditions in order to ensure proper function through the lifetime of the switch.

3.2. **Firmware Method.** A slight improvement from the RC filter is to connect the switch directly to the GPIO on the microcontroller and debounce it with firmware.

Listing 1. Firmware implementation of debouncing

```
1
2  if (!button_value) {
3      /* Released state */;
4  } else {
5      if (!previous_button_value) {
6          transition_time = time(NULL);
7      }
8      if (transition_time > debounce_time) {
9          /* Released state */;
10     }
11 }
12 previous_button_value = button_value;
```

In this example code, the switch clicks will be debounced with a time delay of *debounce_time*, but the releases will be instant. This method is only considered a slight improvement because release latency is not nearly as important as click latency for the vast majority of applications. The code can be modified shift the delay to release. All the same considerations from the RC Filter apply when choosing an appropriate value for *debounce_time*.

Furthermore, some switches will bounce as the contact is opening as well. The example firmware implementation shown above will not function properly for such a switch, an additional delay period will need to be added for the release.

3.3. **Latch Method.** An SPDT switch is a prerequisite to use the latch method. Rather than relying on the state from a single pin, the latch method determines the digital state based on which pin received the most recent signal. This makes the latch method the only time-independent solution, meaning there is zero additional latency from the circuit on the switch.

There are two primary ways to implement a latch - with a physical circuit such as an $\overline{SR}$ latch or with a firmware equivalent. There are a few compromises to be considered when deciding which implementation to use. A firmware latch requires fewer physical components, adds additional processing burden to each poll, and uses twice the number of GPIO pins on the microcontroller. Depending on the hardware chosen, a hardware solution can also introduce a non-negligible amount of latency into the circuit. In general, if performance is not affected, the firmware latch should be utilized.

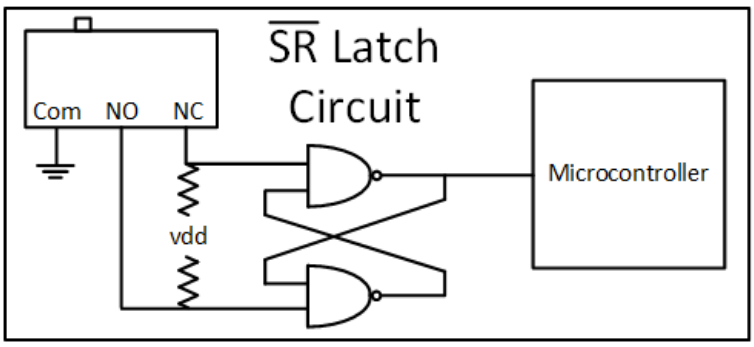

Figure 9. $\overline{SR}$ Latch circuit diagram

3.4. **Other Electrical Considerations to Debouncing.** In addition to removing bounces, typically debouncing circuits accomplish other tasks, depending on application. One such consideration is ESD protection. If the microcontroller I/O pins are not adequately protected, an additional ESD protection IC or circuit may be necessary. In many cases, debouncing-specific ICs, as mentioned in Anonymous (2000) incorporate ESD protection, transient suppression, and overvoltage protection, and may be a lower-cost solution than implementing a "plain" latch or RC filter. The cost of building an RC filtered input with transient and ESD protection from discrete components is likely higher than a purpose built (and superior performance) latch based IC. Overall, care should be taken to meet design requirements regarding robust electrical design while maximizing performance. The "simple" RC filter often is the more expensive option.

4. Conclusion

While initial performance is similar for many switches, one factor which was not studied here is longevity (and failure mode). While the latch debouncing method generally allows usage of a switch that has increased bouncing well beyond initial specifications, rated lifetime for switches can easily be exceeded by service life. Whichever switch is chosen should use the capacitor method for meeting the wetting current requirements and be implemented with a time-independent debouncing solution - preferably using the latch firmware method if there are sufficient IO pins. Selecting an appropriate switch involves examining the mechanical construction, longevity (relying both on manufacturer specification and real-world data), physical characteristics (pretravel distance, spring force, click feel, sound, etc.), and cost. While we highly recommend switches from the Omron D2F or Honeywell ZX10C series, application and budget varies greatly, and it is impossible to make a general recommendation. Overall robust engineering is more important than a specific switch. There are many switch manufacturers with unique products, many of which are very well suited for general use or specialized for individual applications.

References

Anonymous. (2000). Switch bounce and other dirty little secrets. *Maxim Integrated*.

Liu, H., Leray, D., Pons, P., & Colin, S. (2013). An asperity-based finite element model forelectrical contact of microswitches. *HAL*.