Microprocessors

Peripheries - timers, buttons, keypads, displays

Stanislav Vítek
Department of Radioelectronics
Czech Technical University in Prague

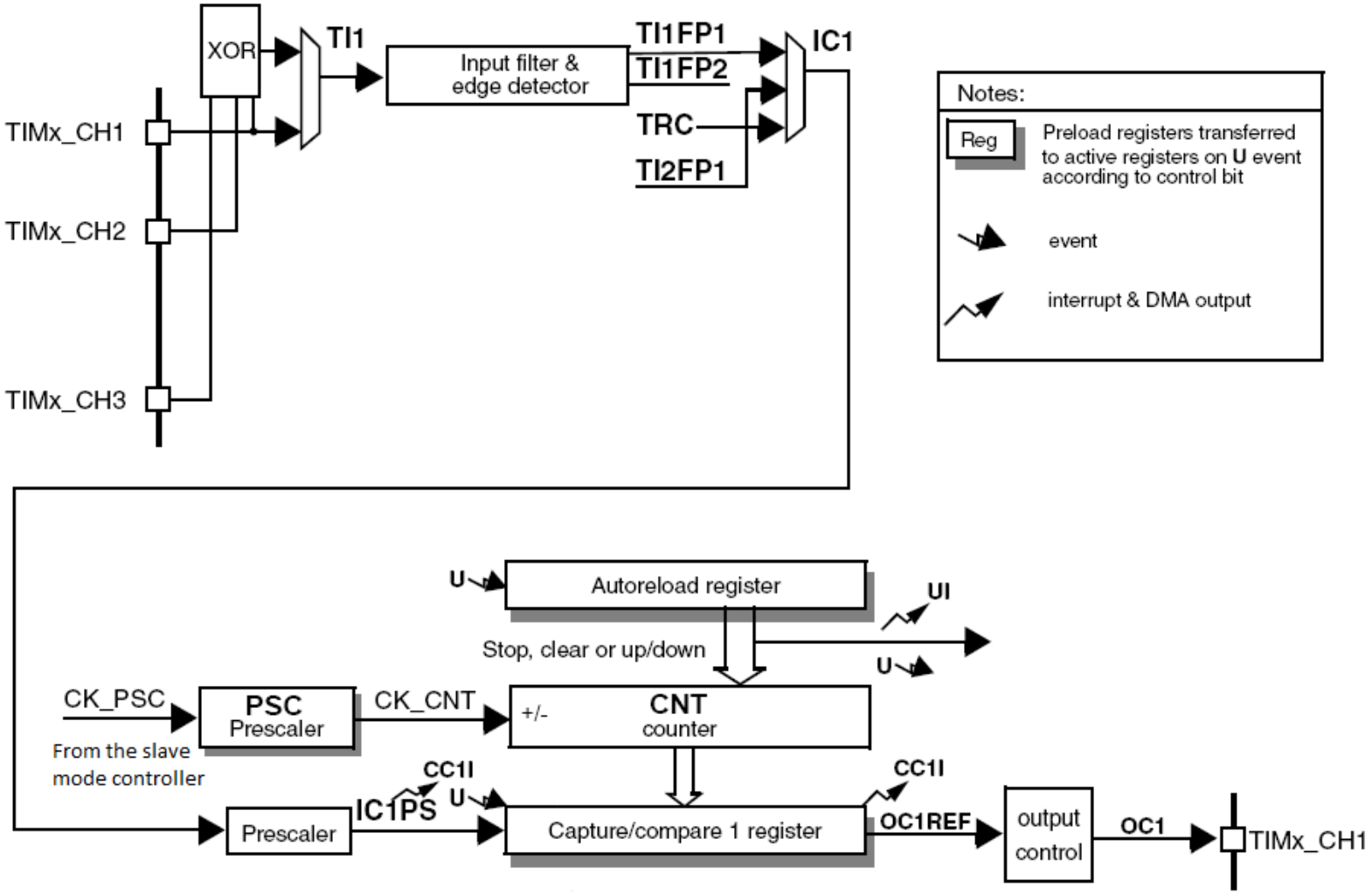
Timers

Hardware timers are used to:

- Generate
  - signals of various frequencies
  - pulse-width-modulated (PWM) outputs
  - accurate time base
- Trigger events at known frequencies
- Measure elapsed time between two events
- Count events

Without accurate timing, digital control engineering is not possible - the control signals (controller action) have to happen at the exact right moment in time, e.g. timing control of an engine, etc.

Example of timer (STM32F4)



Timer modes

1. **Capture Mode** allows timing for the duration of an event. This circuit gives insight into the current state of a register which constantly changes its value. In this case, it is the timer register.
2. **Compare Mode** compares values contained in two registers at some point. One of them is the timer register. This circuit also allows the user to trigger an external event when a predetermined amount of time has expired.
3. **Pulse Width Modulation (PWM)** can generate signals of varying frequency and duty cycle.

Input Capture

Capture the times of events

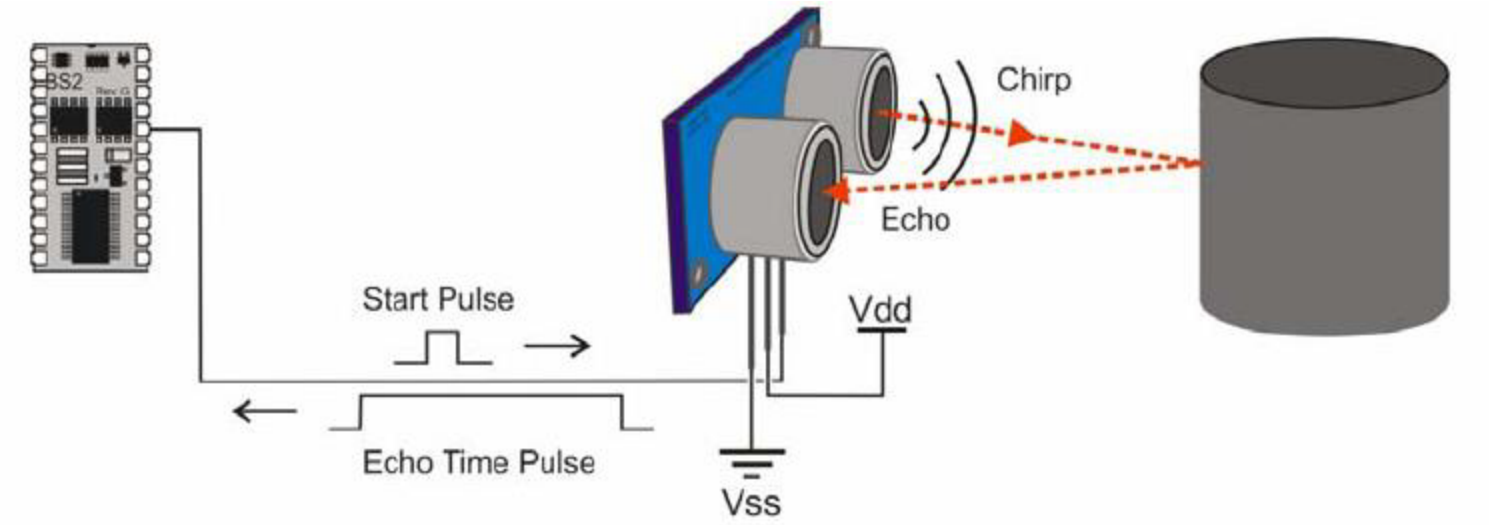
Many applications in microcontroller applications:

- Measure rotation rate
- Remote control
- Sonar devices
- Communications

Generally, any input that can be treated as a series of events, where the precise measure of event times is important

Application: Sonar Device

Ping sensor: ultrasound distance detection device



- Speed of sound: ~340m/s
- Echo Pulse width proportional to round trip distance
- Temperature affects on sound speed negligible in temperature controlled room

Sonar principle

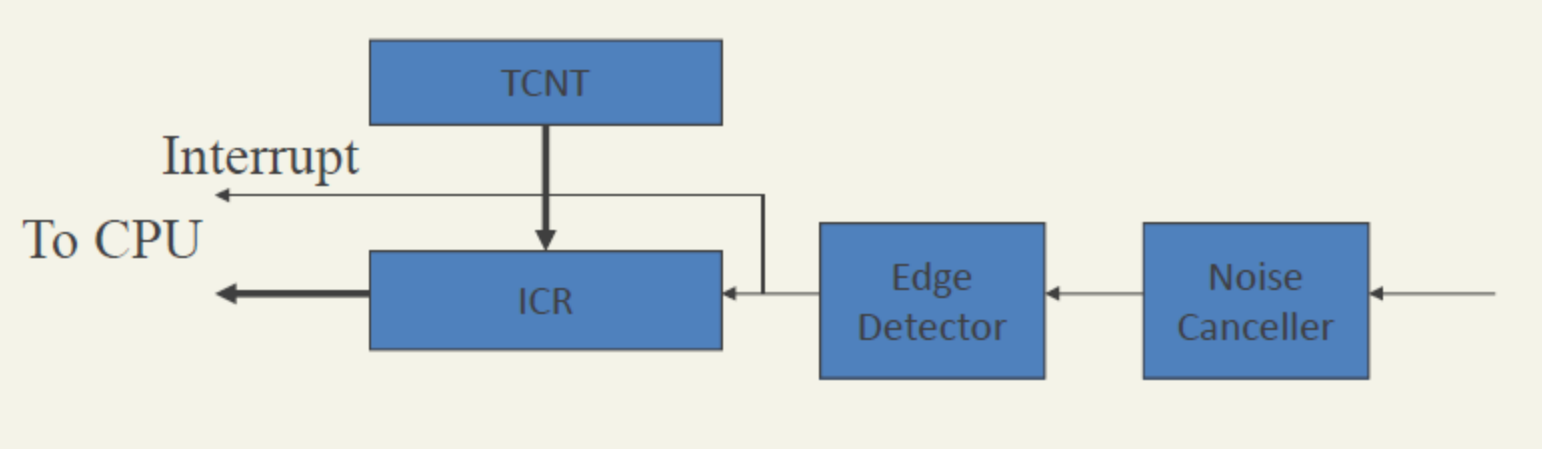
- Assume 62.5KHz Input Capture clock
- 1ms <=> 62.5 clock ticks/periods <=> 34cm

Time Diff.	Clock Count	One-way Distance
2ms	125	0.34m
4ms	250	0.68m

How to capture the times of rising edge and falling edge?

Input Capture: Design Principle

Time value (clock tick count) is captured first then read by the CPU



TCNT: Timer/Counter

ICR: Input Capture Register

Input Capture: Design Principle

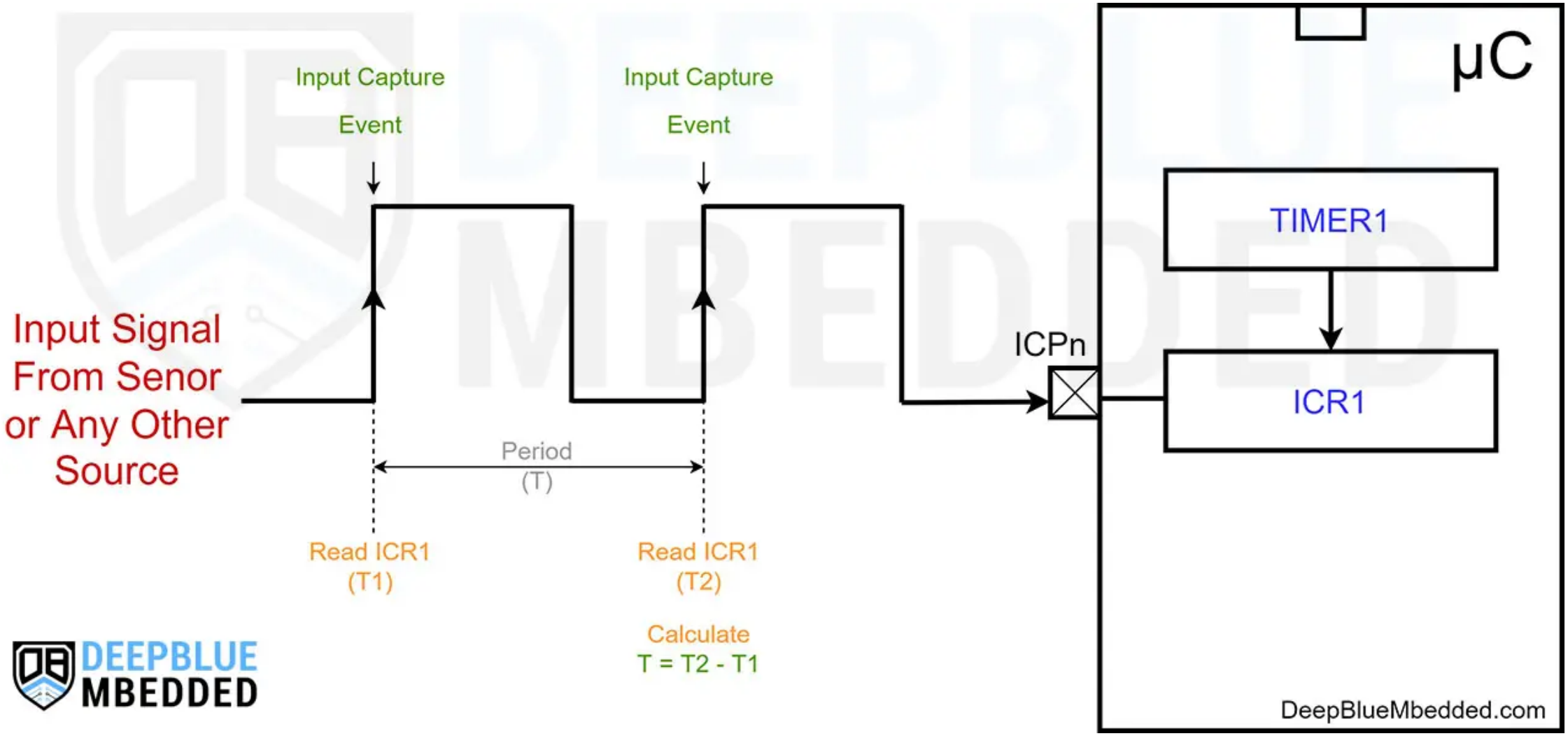
What happens in hardware and software when and after an event occurs

- The event's time is captured in an ICR (input capture register)
- An interrupt is raised to the CPU
- CPU executes the input capture ISR, which reads the ICR and completes the related processing

The captured time is precise because it's captured immediately when the event occurs

The ISR should read the ICR and complete its processing fast enough to avoid loss of events

Frequency Measurement



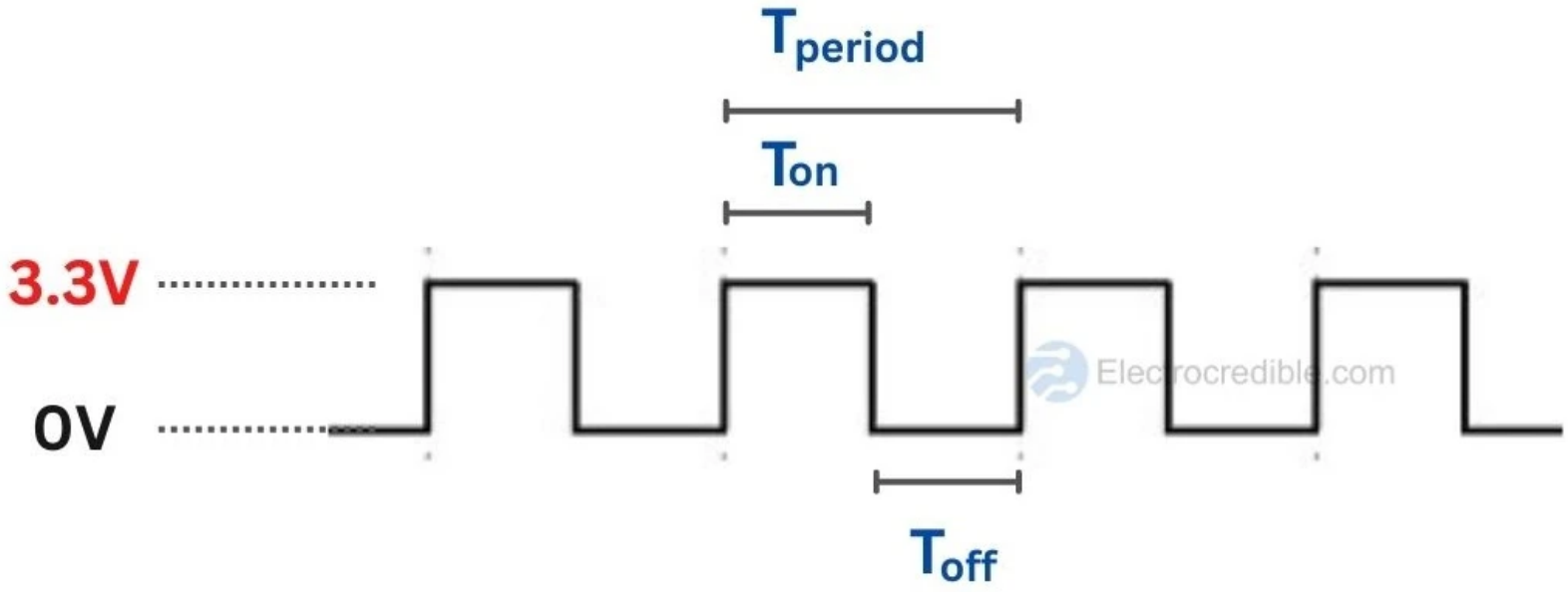
Output Compare Mode - PWM

PWM is a modulation technique using which the width of waves in a pulse train can be varied. It has a wide variety of applications such as:

- Generation of analog voltages using a digital signal.
- Speed control of electric motors.
- Telecommunications.
- Audio synthesis and amplification.

Duty Cycle, frequency, and period are three notable parameters of a PWM signal. Frequency relates to the number of times a PWM signal switches between ON and OFF state per second.

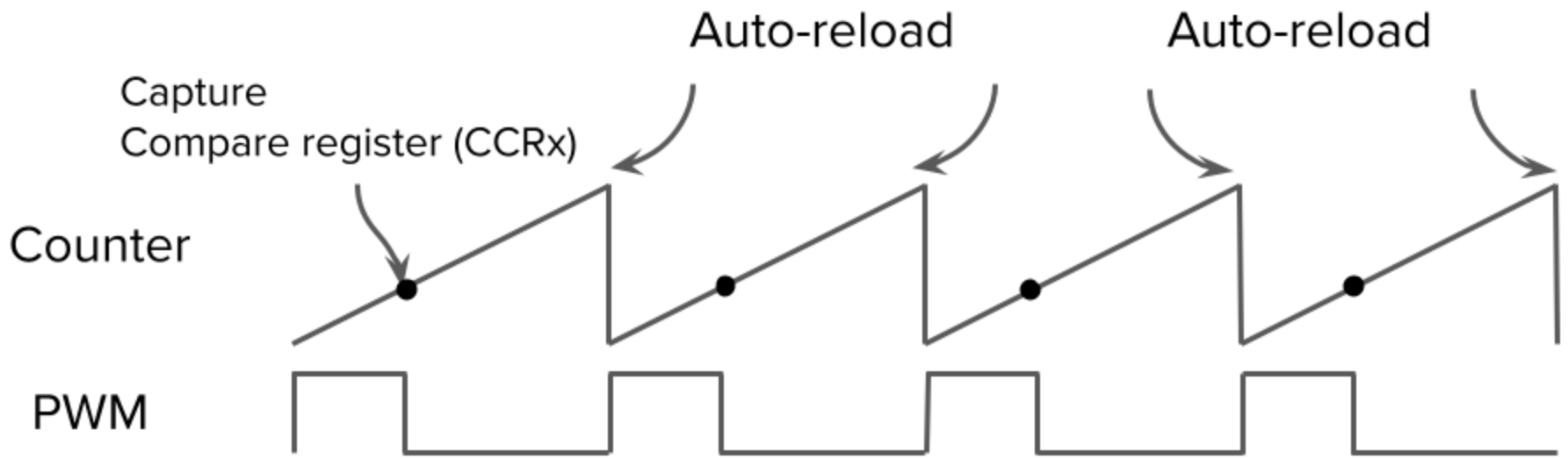
Output Compare Mode - PWM



$$Duty\,cycle = \frac{T_{on}}{T_{on}+T_{off}} \times 100\%$$

Output Compare Mode - PWM

PWM implementation, STM32

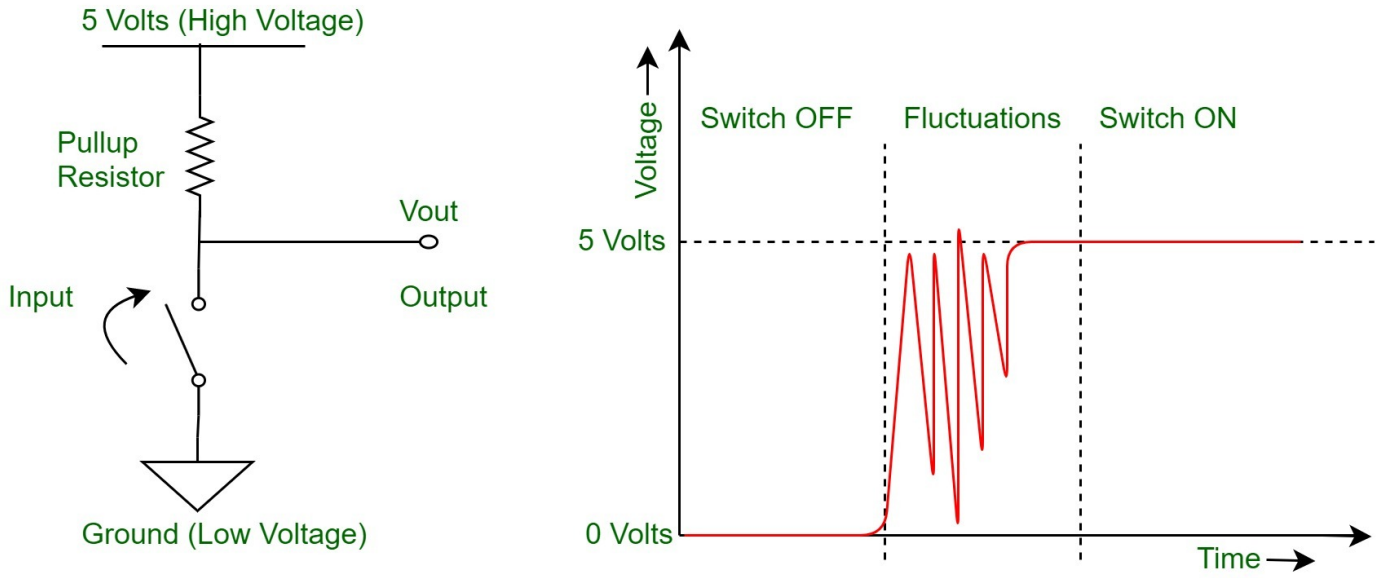


$$\text{Duty cycle (\%)} = 100 \times \frac{\text{CCRx}}{\text{Auto-reload} + 1}$$

$$\text{Frequency} = \frac{\text{Timer's frequency}}{\text{Auto-reload} + 1}$$

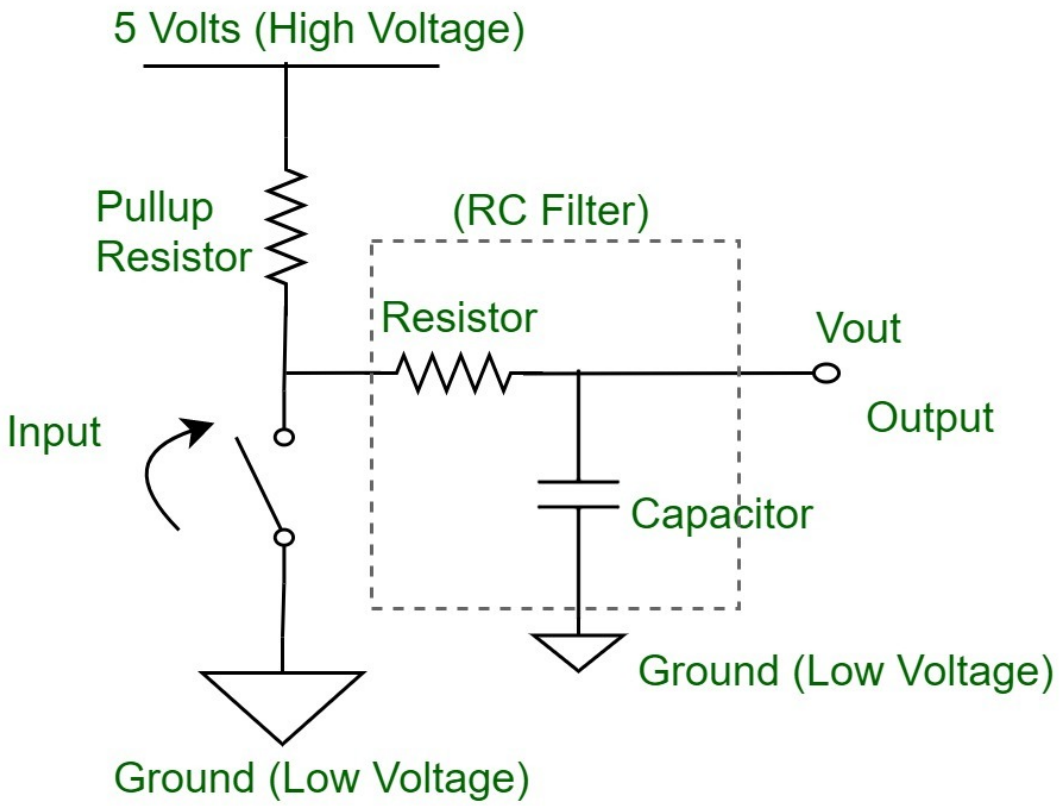
Switches and push buttons

Push button is a very simple device. However, due to its mechanical nature, it is not so easy in practice… When the contacts strike together, their momentum and elasticity cause bounce. The voltage is not just simple jump from one logic value to the other, but spikes appears in its waveform. µC may evaluate this as series of logic 1 and logic 0. It may last even tents of miliseconds!



Switch debounce using RC filter

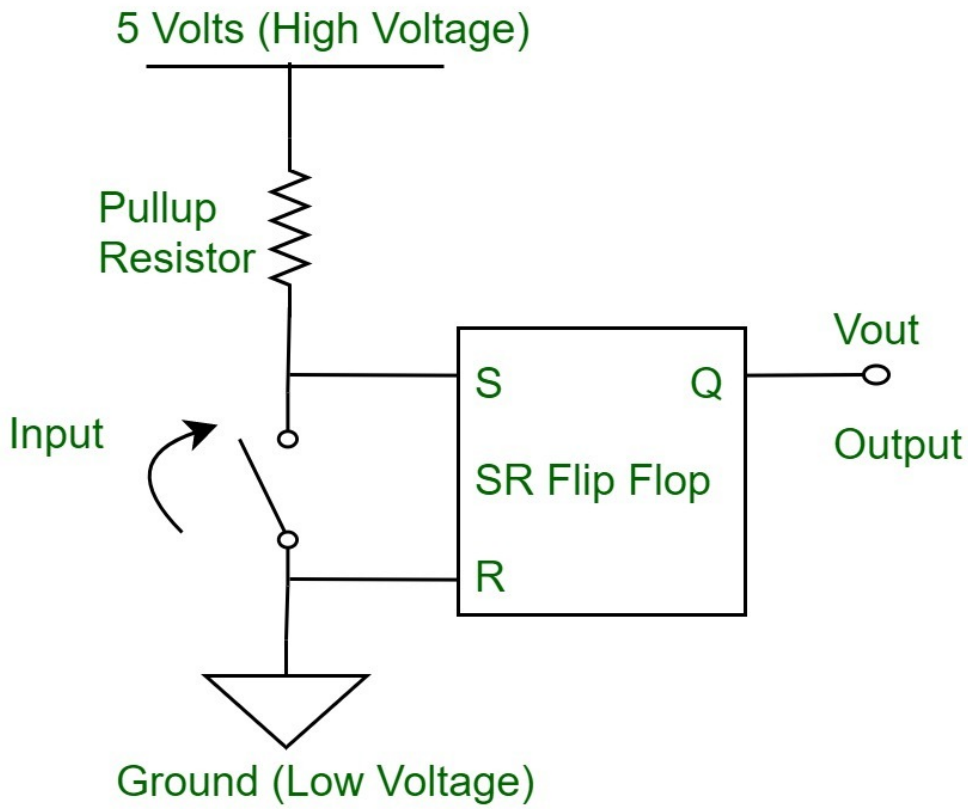
- This circuit involves the combination of a resistor and a capacitor circuit to act as a filter to smooth out the output glitch for the switch.
- **Disadvantage** – voltage transition is very slow (time constant must be comparable with the period during which spikes occur).
- Schmitt trigger may be used to shape



Switch debounce using SR flip-flop

S-R Flip Flop Latch circuit

- The circuit when introduced in the output part of the switch, it will retain the voltage level of the input as the output state.
- Thus, latching to the input, when change in state is introduced.
- This method is useful, but adds to the bulkiness of the simple circuit.



Software Switch Debouncing

- Solution, which does not require any hardware circuitry, is a quite simple software technique.
- When the program detects a change at the state of input value, it simply waits for defined period and read the value again (or even several times).
  - If both values are the same, we assume the button was pressed.
  - Delay must be similar to the time constant of the RC filter, that is comparable to the maximum supposed period for which some spikes occur.
  - Of course, if this interval is poorly set, errors still occur.
  - Moreover, this time should be in tens of milliseconds, so it may consume a considerable amount of processing power (although timers and interrupts may be used for that).

Push button keyboard

More buttons are coupled to keyboards with matrix alignment. From the point of view of keyboard operation, there are two basic principles:

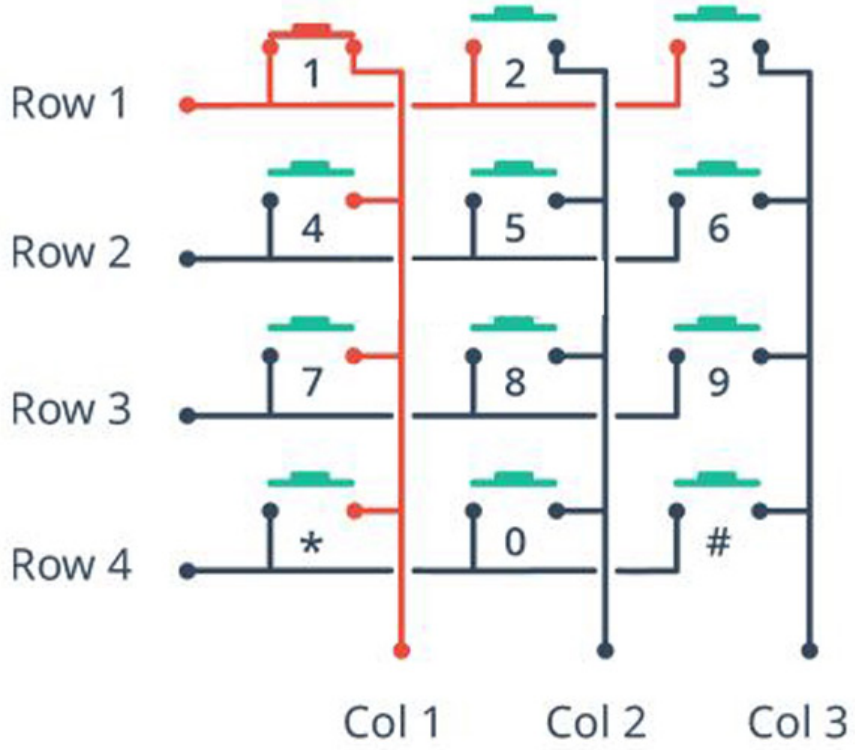
- Cyclic reading
- Interrupt handling

From the point of view of hardware treatment, following questions must be taken into account:

- How a spikes after the button is pressed will be treated (RC with Schmitt trigger, RS flip-flop, software)
- How many keys may be pushed simultaneously

Push button keyboard

- We need 7 GPIO pins
  - 4 are configured as output (rows)
  - 3 are configured as inputs (cols)
- Logic value of inputs is defined by an internal pull-up resistors
- In IDLE, outputs should be in high impedance state



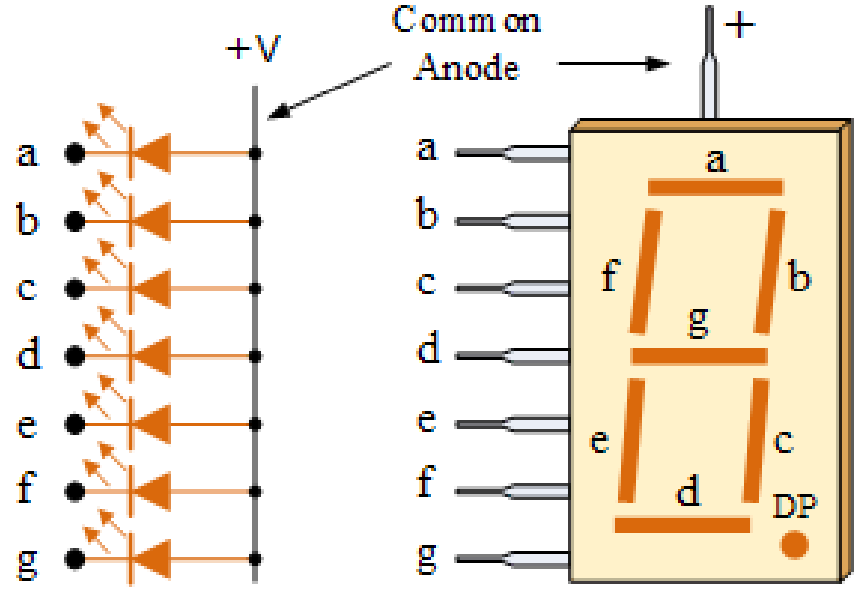
Push button keyboard

Scan periodically keyboard as follows:

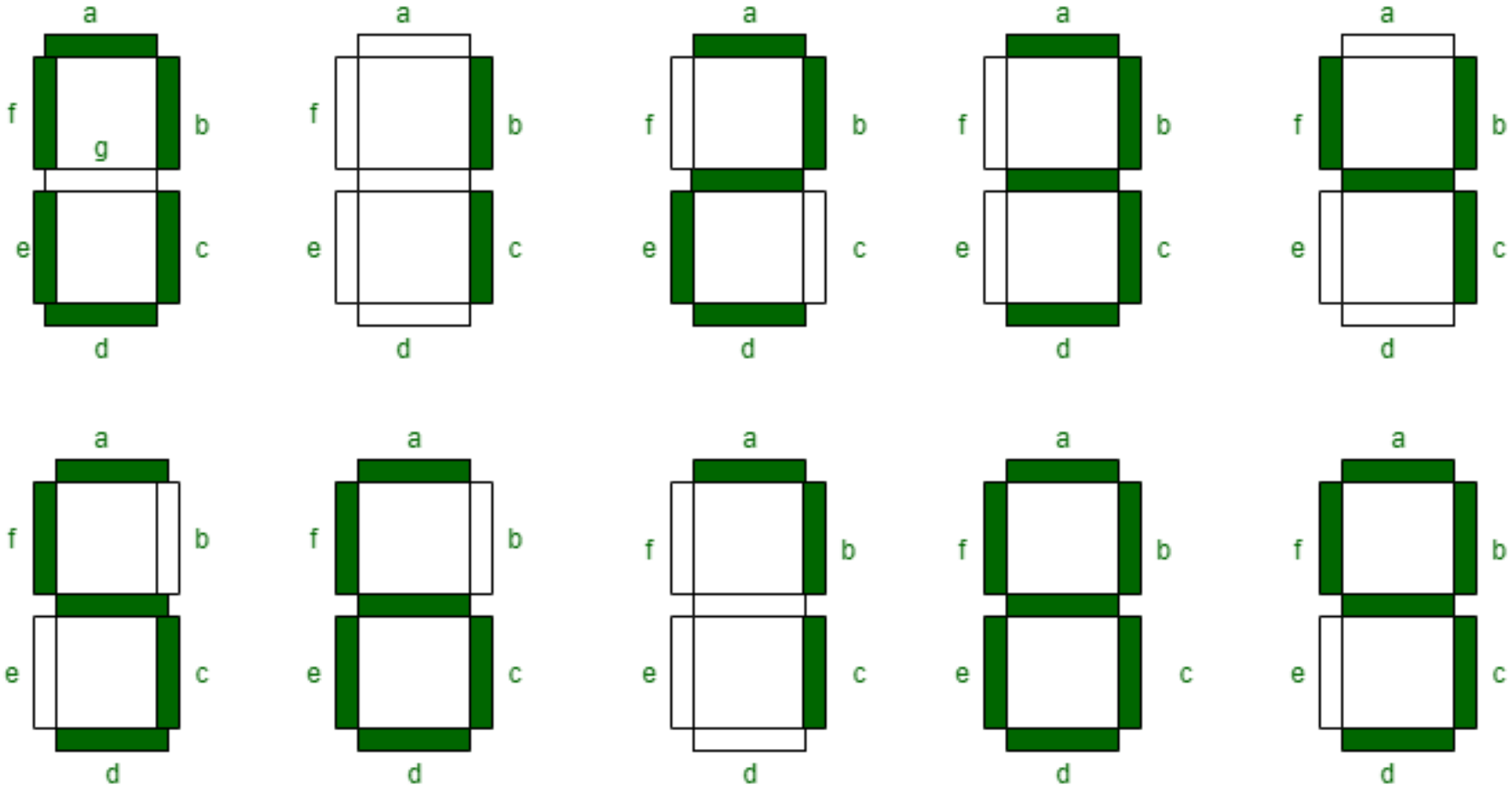
1. Assert the first row low
2. Read the state of input port B; bits, which are in a logic 0, corresponds to keys, which were pressed, so you handle keys “0”, “1”, …, “A”
3. Return the first row to high impedance. Assert second row low.
4. Read the state of input port B again, so you handle keys “3”, “4”, …, “B”
5. Repeat the same procedure with third and fourth row to handle keys “6”, …, “D”
6. Wait a few (tents) of milliseconds and repeat points 1, … 5 again. Compare both results to prevent spikes. If all columns are logic 1 in all readings, no key is pressed.

7-Segment Displays

- The 7-segment display, consists of seven LEDs arranged in a rectangular fashion
- In the common anode display, all the anode connections of the LED segments are joined together to logic “1”.
- The individual segments are illuminated by applying a ground, logic “0” or “LOW”



Digital Segments for all Numbers



Driving a display