POLITECNICO
DI MILANO

www.polimi.it





STM32 – Encoder

Federica Villa

Incremental encoders

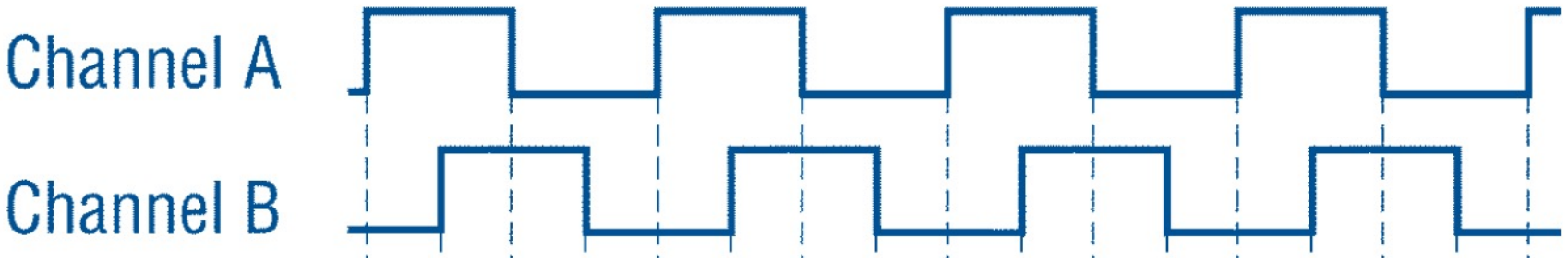
Provide a specific number of pulses per revolution (PPR) in rotary motion, or per inch or millimeter in linear motion.

- **single channel output** → doesn't provide direction of movement
- **quadrature output** → provides direction sensing
  (two channels 90° out of phase)

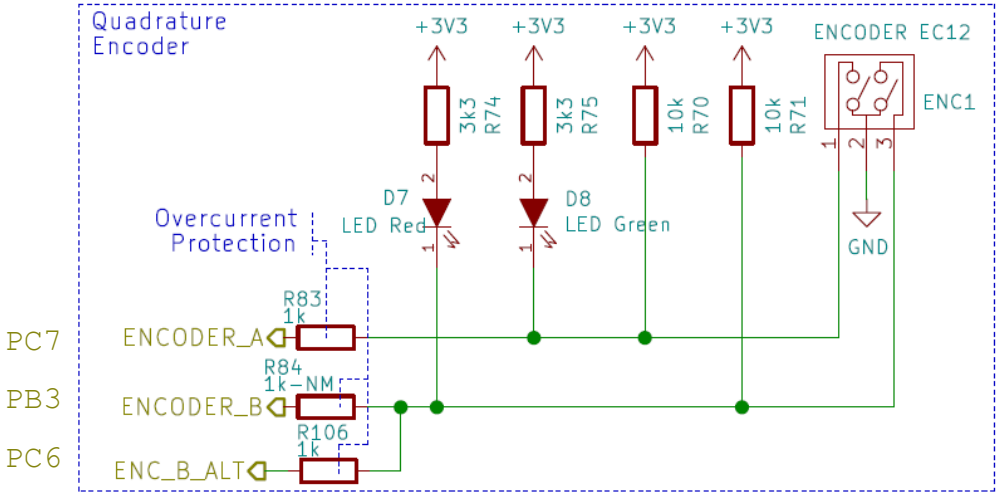
To determine position, its pulses must be accumulated by a counter.

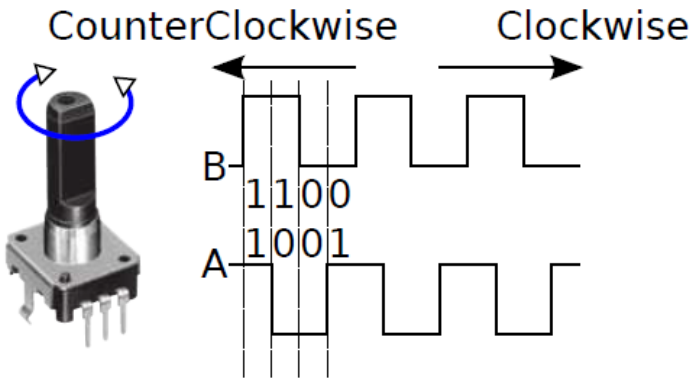
When starting up, the equipment must be driven to a reference or home position to initialize the position counters.

Some incremental encoders also produce another signal, the "marker," produced once per revolution.



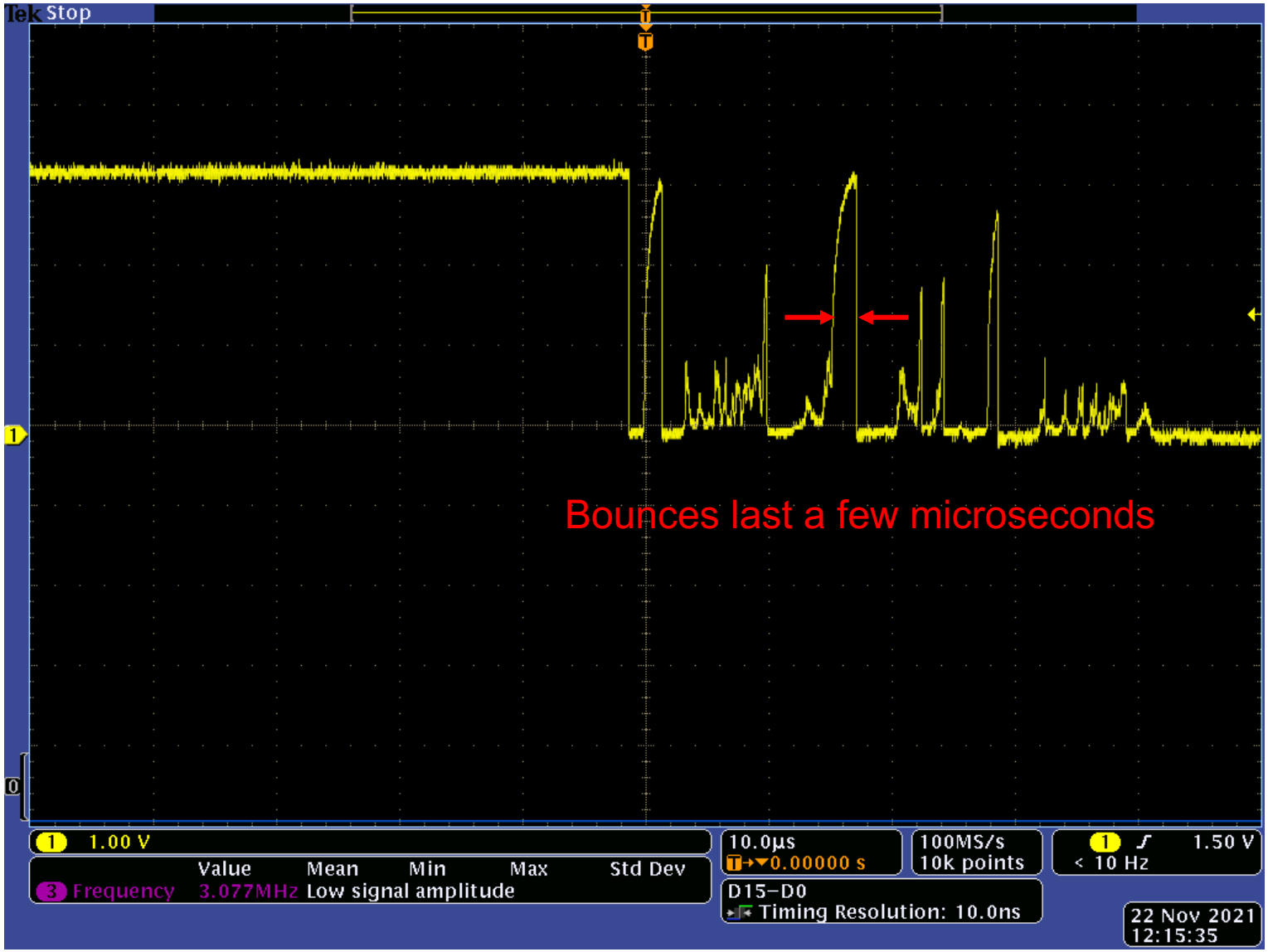
PMDB16: encode





- The encoder is connected to pins PC6 / PC7 of the STM32
- No hardware debouncing: we will use **digital filtering**
- STM32 Timer peripherals feature dedicated encoder mode. Let's setup the hardware

PMDB16: encoder debouncing



STM32 Timer input digital filter

IC1F: Input capture 1 filter

This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output:

0000: No filter, sampling is done at f_{DTS}
0001: $f_{SAMPLING}=f_{CK\_INT}$, N=2
0010: $f_{SAMPLING}=f_{CK\_INT}$, N=4
0011: $f_{SAMPLING}=f_{CK\_INT}$, N=8
0100: $f_{SAMPLING}=f_{DTS}/2$, N=6
0101: $f_{SAMPLING}=f_{DTS}/2$, N=8
0110: $f_{SAMPLING}=f_{DTS}/4$, N=6
0111: $f_{SAMPLING}=f_{DTS}/4$, N=8
1000: $f_{SAMPLING}=f_{DTS}/8$, N=6
1001: $f_{SAMPLING}=f_{DTS}/8$, N=8
1010: $f_{SAMPLING}=f_{DTS}/16$, N=5
1011: $f_{SAMPLING}=f_{DTS}/16$, N=6
1100: $f_{SAMPLING}=f_{DTS}/16$, N=8
1101: $f_{SAMPLING}=f_{DTS}/32$, N=5
1110: $f_{SAMPLING}=f_{DTS}/32$, N=6
1111: $f_{SAMPLING}=f_{DTS}/32$, N=8

Minimum pulse duration
@ f_{CK_INT} = 84 MHz, CKD = 1 (CKD = 4)

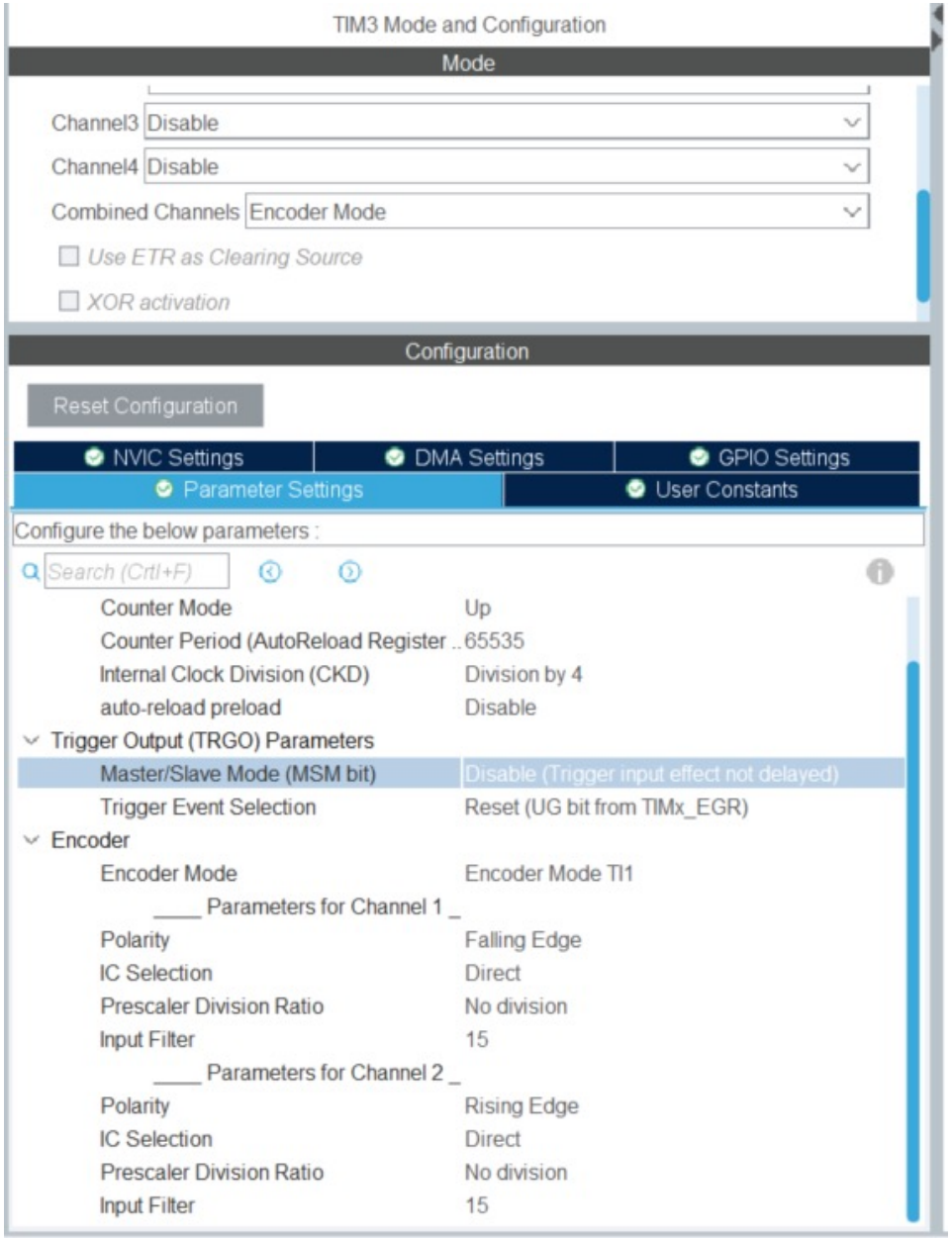
Unfiltered: 11.9 ns (47.7 ns)

381 ns (1.52 µs)

3.05 µs (**12.2 µs**)

$f_{DTS} = f_{CK\_INT}$ / CKD (Internal clock division)

STM32CubeIDE: setup



Encoder working principle

Active edge	Level on opposite signal (TI1FP1 for TI2, TI2FP2 for TI1)	TI1FP1 signal		TI2FP2 signal	
		Rising	Falling	Rising	Falling
Counting on TI1 only	High	Down	Up	No Count	No Count
	Low	Up	Down	No Count	No Count
Counting on TI2 only	High	No Count	No Count	Up	Down
	Low	No Count	No Count	Down	Up
Counting on TI1 and TI2	High	Down	Up	Up	Down
	Low	Up	Down	Down	Up

When a transition on one channel happens, the system observes the level of the other signal.

Selecting the polarity of the channel and the mode of the encoder, the microcontroller decodes the data as depicted in this picture, either rising or decreasing the counter value.

Project 1a – Encoder readout

Objective

Read the encoder position
and send to the PC the
rotation speed in rpm

Aim of the project

Objective of the project is to readout a quadrature encoder, using the specific modality of STM32 timers, in order to provide the rotation frequency (expressed in rpm / rotations per minute) and direction (“+” for clockwise and “-” for counterclockwise).

The result must be displayed using the remote terminal.

Project hints

- Identify the encoder pins, and enable then in TIMx_CHy mode
- Setup the timer to operate in encoder mode, with the correct input filter applied. Start the timer in encoder mode.
- Within the while(1) loop, poll the counter value every second and compute the delta from the previous read, then convert it to rpms. How many counts does a full rotation of the encoder provide?
- Beware of overflow and underflow of the timer. How to solve this issue?
- Compile and debug the code.

Project 1b – Encoder readout

Objective

Read the encoder position
and send to the PC the
rotation speed in rpm

Use a timer as a timebase and DMA to transfer the UART data