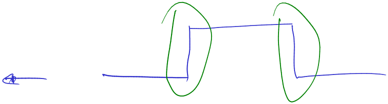
Embedded System Design and Modeling

IV. FSM Patterns

FSM Patterns

- **Design patterns**: reusable templates which appear often in applications
- Patterns
  - Operating on signal transitions
  - Debounce (one-sided, two-sided)

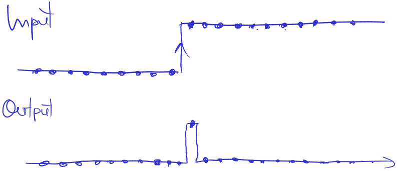


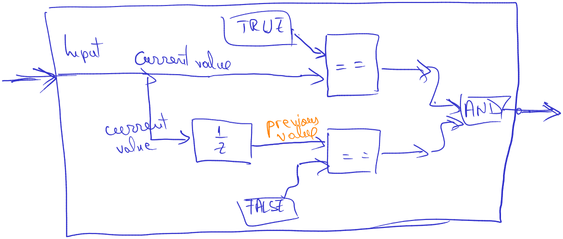
Operating on signal transitions

- For boolean signals/conditions
- Use when information is in the signals' "**fronts**" ? (edge / transition) rather than in its values
- Solution: detect signal transitions
  - rising edge
  - falling edge
  - both

Detect rising edge

- Draw here: detect rising edge

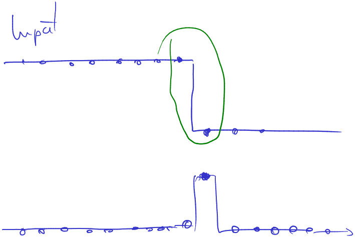


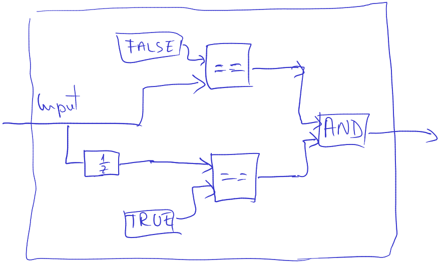


```
if ( current == True && prev. = = False)
        output = True;
else
        output = False
```

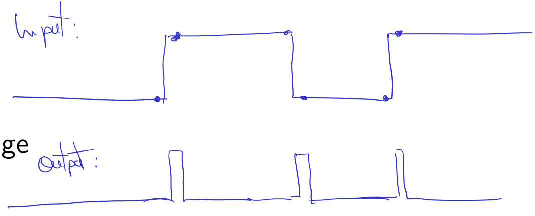
Detect falling edge

- Draw here: detect falling edge

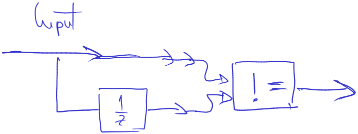




Detect any edge



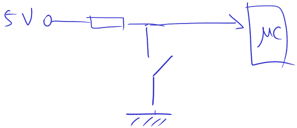
- Draw here: detect any edge

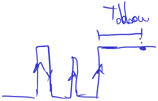


output = (current != prev.) ;

Debouncing

- For boolean signals/conditions
- Bouncing: real signals look like this:





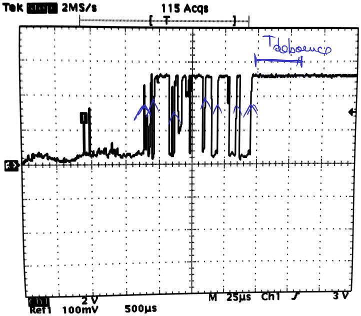
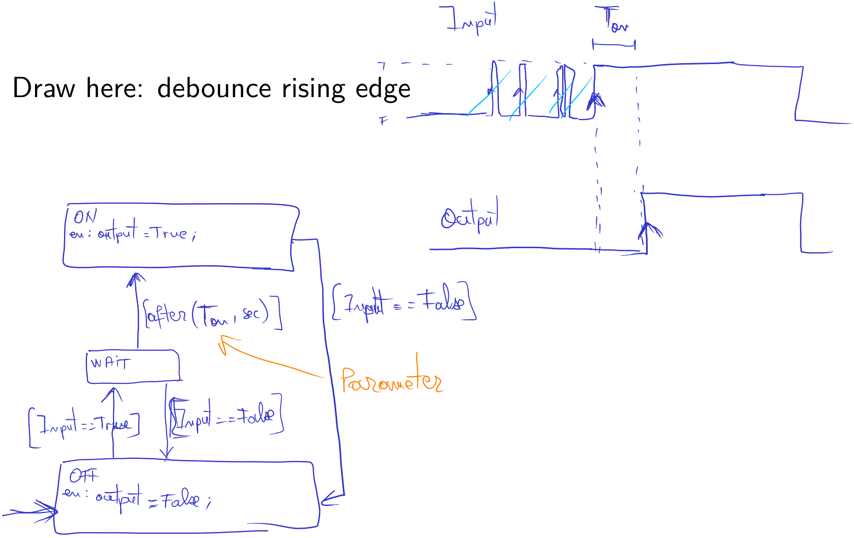


Figure 1: Signal change when pushing a button

- Use debouncing to avoid spurious transitions

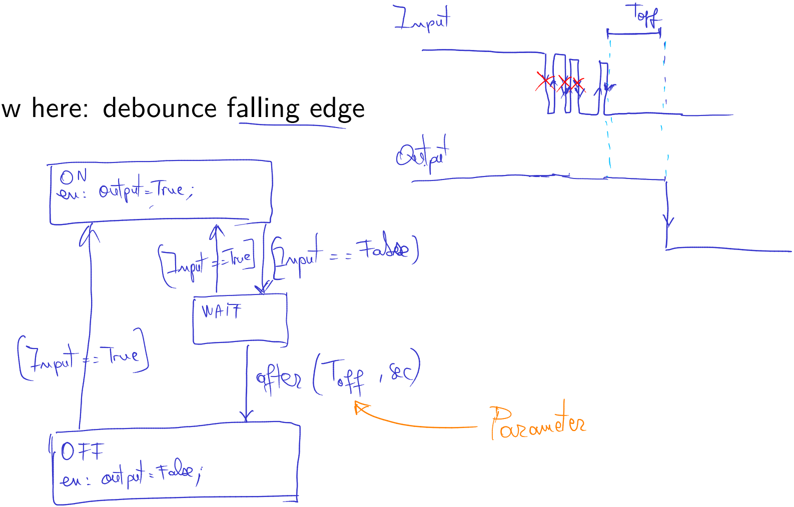
Debouncing rising edge

- Draw here: debounce rising edge



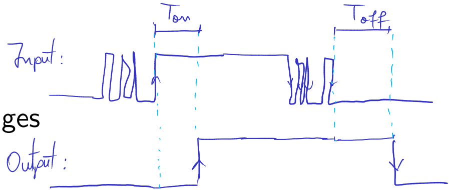
Debouncing falling edge

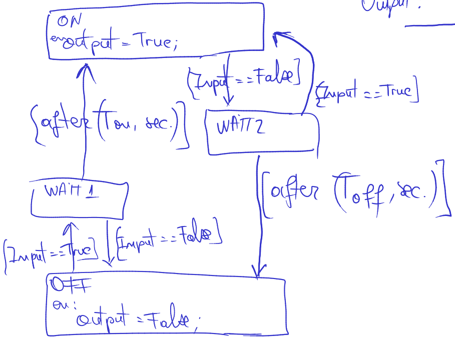
- Draw here: debounce falling edge

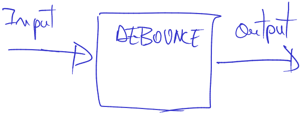


Debouncing both edges

- Draw here: debounce both edges


Ton
Toff
Input:
Output:


ON
output = True;
[Input == False]
[Input == True]
[after (Ton, sec.)]
WAIT 2
[after (Toff, sec.)]
WAIT 1
[Input == True]
[Input == False]
OFF
en:
output = False;


Input
DEBOUNCE
Output