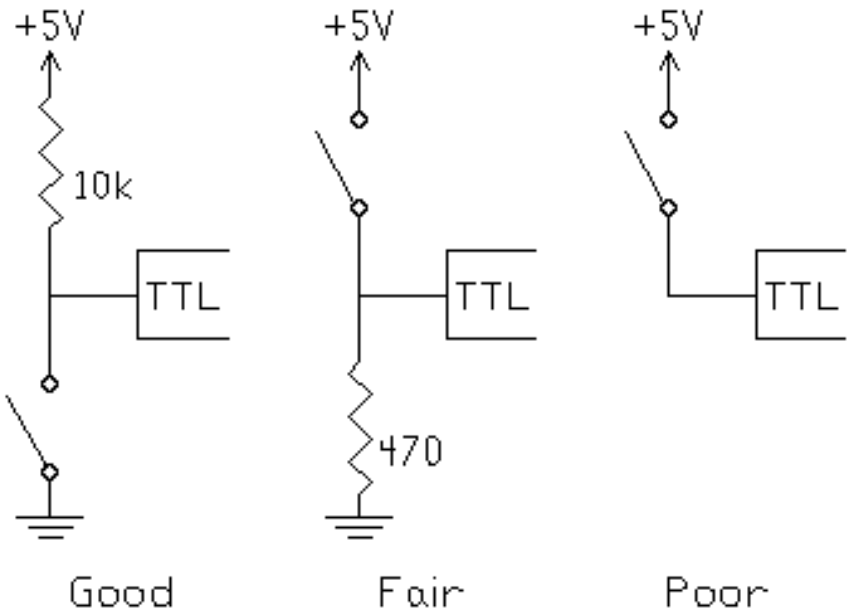
Switches



Quelle: http://www.freewebs.com/maheshwankhede/ports.html

- Pullup-Resistor probably provided in uC

Debouncing

- Capacitor
- Code
  - waiting

```
if (KEY1_Get()) {
    WAIT1_Waitms(50); /* simple debounce */
    if (KEY1_Get()) { /* still pressed? */
    …
    }
}
```

  - Edge detection

uC configuration

PTADD

Read: / Write:	PTADD7	PTADD6	PTADD5	PTADD4	PTADD3	PTADD2	PTADD1	PTADD0
Reset:	0	0	0	0	0	0	0	0

Input or Output
1: Output

PTAD

	Bit 7	6	5	4	3	2	1	Bit 0
Read: / Write:	PTAD7	PTAD6	PTAD5	PTAD4	PTAD3	PTAD2	PTAD1	PTAD0
Reset:	0	0	0	0	0	0	0	0

Data

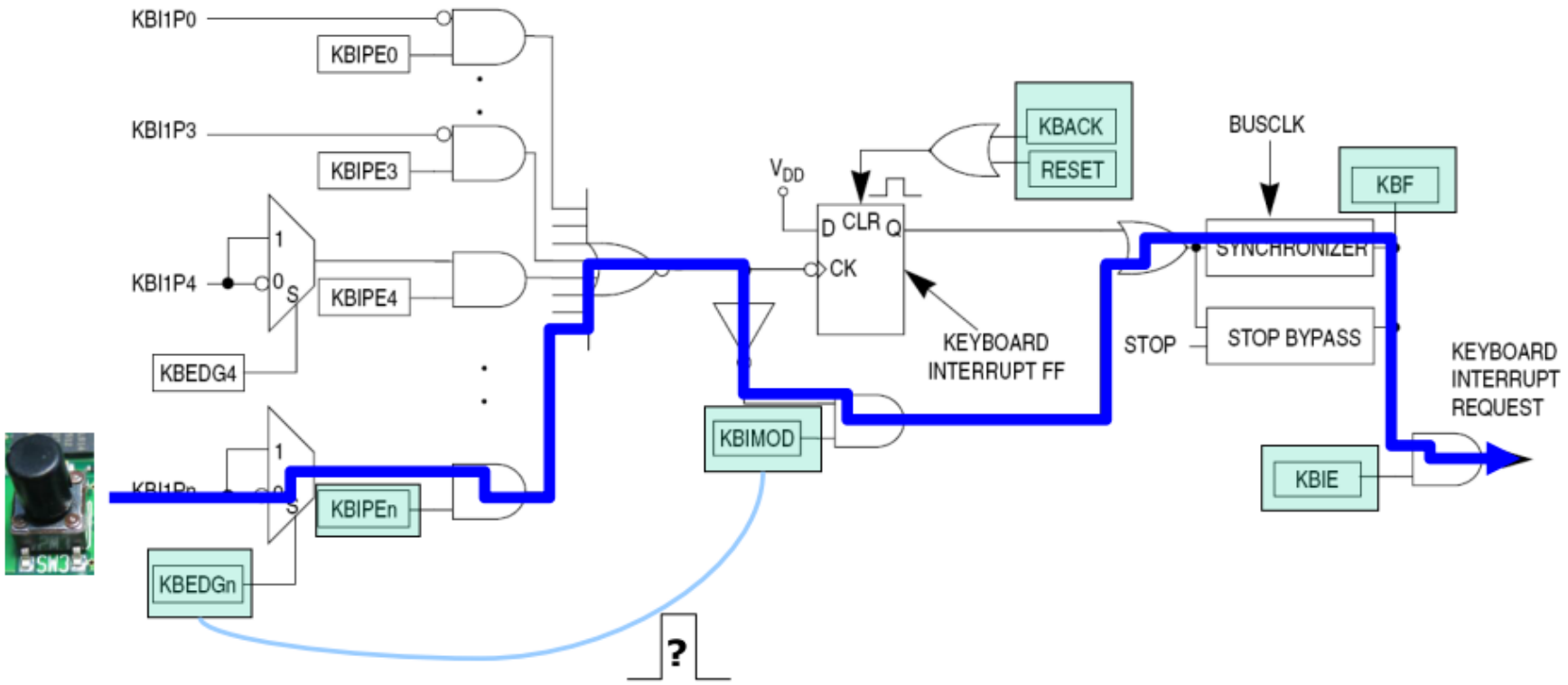
PTAPE

Read: / Write:	PTAPE7	PTAPE6	PTAPE5	PTAPE4	PTAPE3	PTAPE2	PTAPE1	PTAPE0
Reset:	0	0	0	0	0	0	0	0

Pull-Up (for Input)
0: no pullup

➔ **Processor Expert**

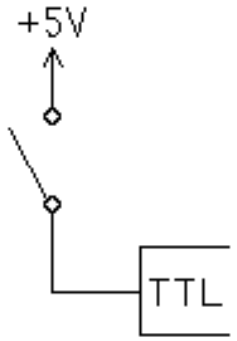
Switches with Interupts



+ no Polling needed

- not sure which switch pressed

Questions



1. Why is this a bad solution?
2. What is debouncing used for?
3. Pros and cons of polling?
4. Pros and cons of interrupt?

Solutions

1. No defined state if button is not pressed.
2. Nullification of the bouncing mechanical parts of a switch/button.
3. + easy programming
   - resource intensive
4. + work capacity saved for other tasks
   - sometimes not easy/possible to implement