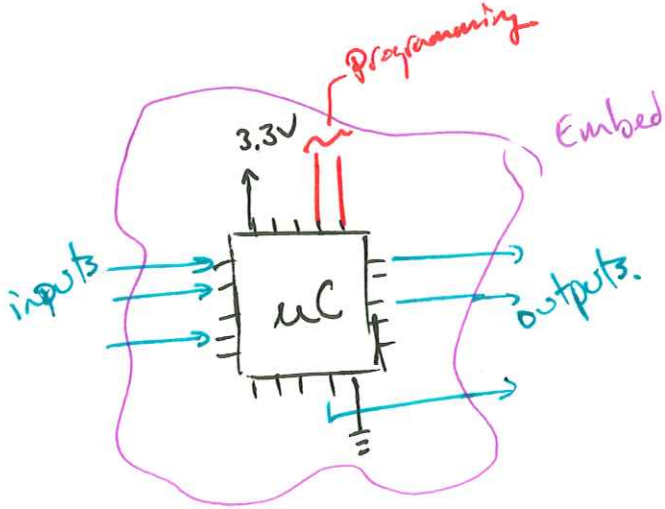
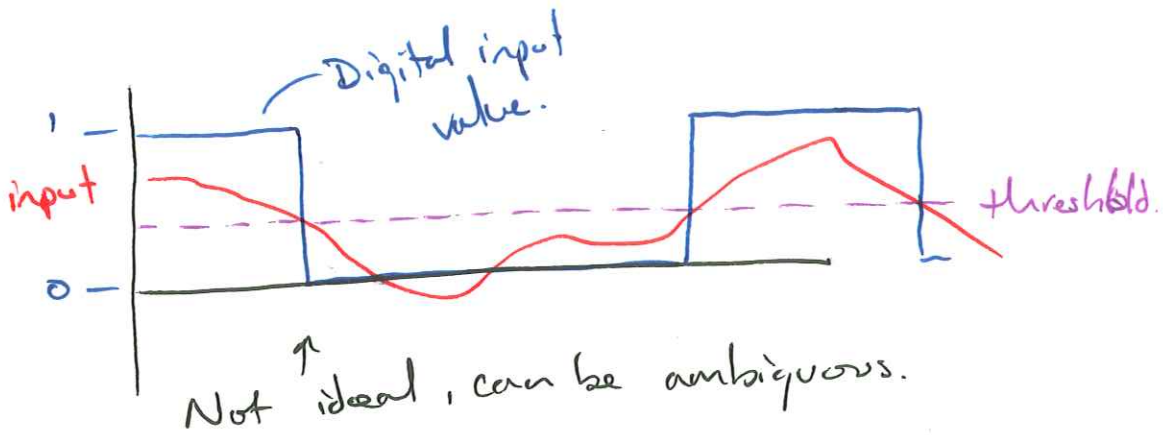
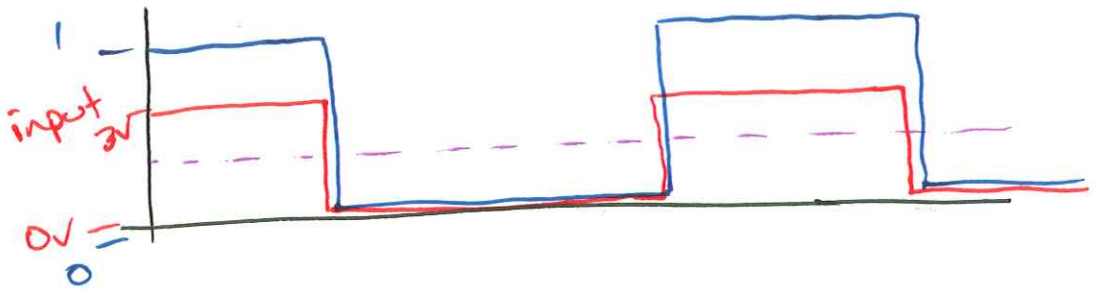


Digital inputs/outputs : DIO

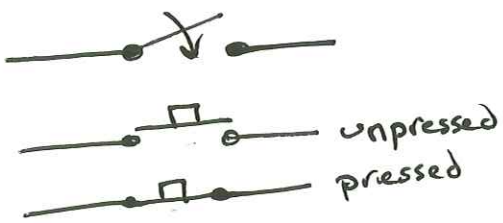


Not ideal, can be ambiguous.

want input to have digital states as well...



Pushbutton / Momentary Switch (for lab switches/pushbuttons - cut endsoff!)

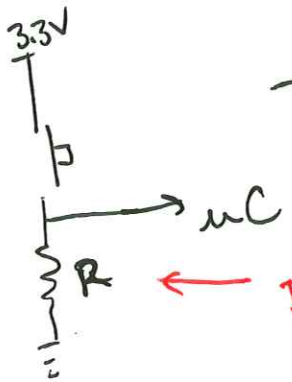


want input to be high V unpressed
low V pressed
(or switched)

→ 3.3V pressed
→ ??? unpressed!

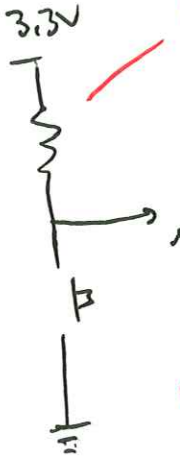
floating input!
voltage undefined.

fix:



→ 3.3V pressed
0V unpressed.

pull-down resistor

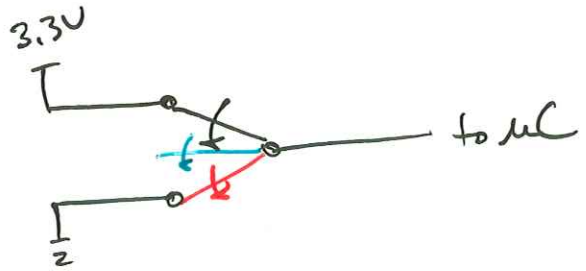


pull-up resistor.

→ 3.3V unpressed
0V pressed

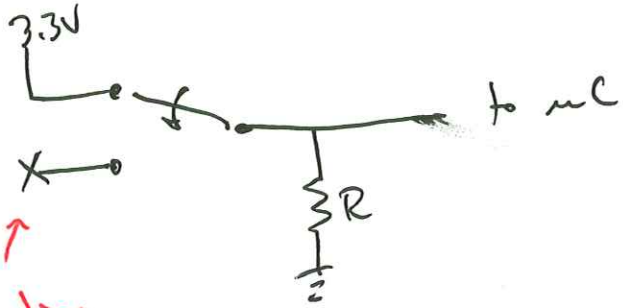
pull-up/down resistors
can be supplied by uC!

Slideswitch (latching Switch)

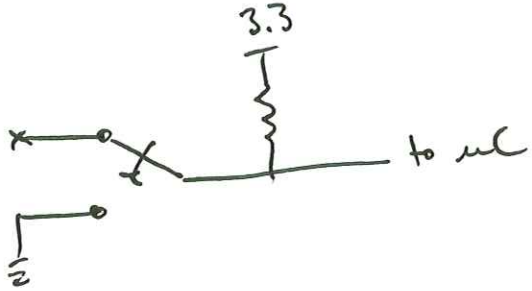


problem: when in middle of switching, signal is floating!

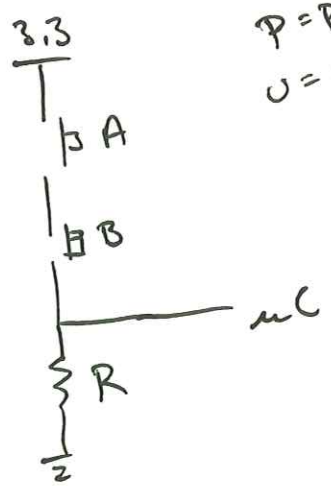
fix:



no connection.



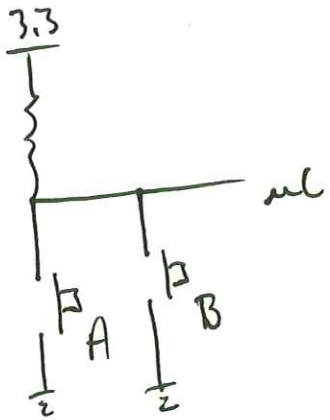
Input logic



P = pressed
U = unpressed

A	B	to μC
U	U	0V
U	P	0V
P	U	0V
P	P	3.3V

AND



A	B	to μC
U	U	3.3V
U	P	0V
P	U	0V
P	P	0V

NOR (~OR)

output value from uC

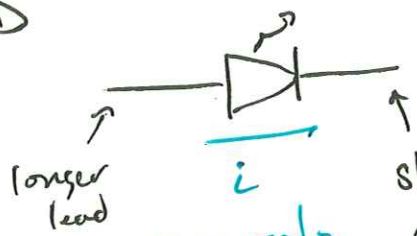
always ~ 3.3V

~ 0V

limited amount of current!

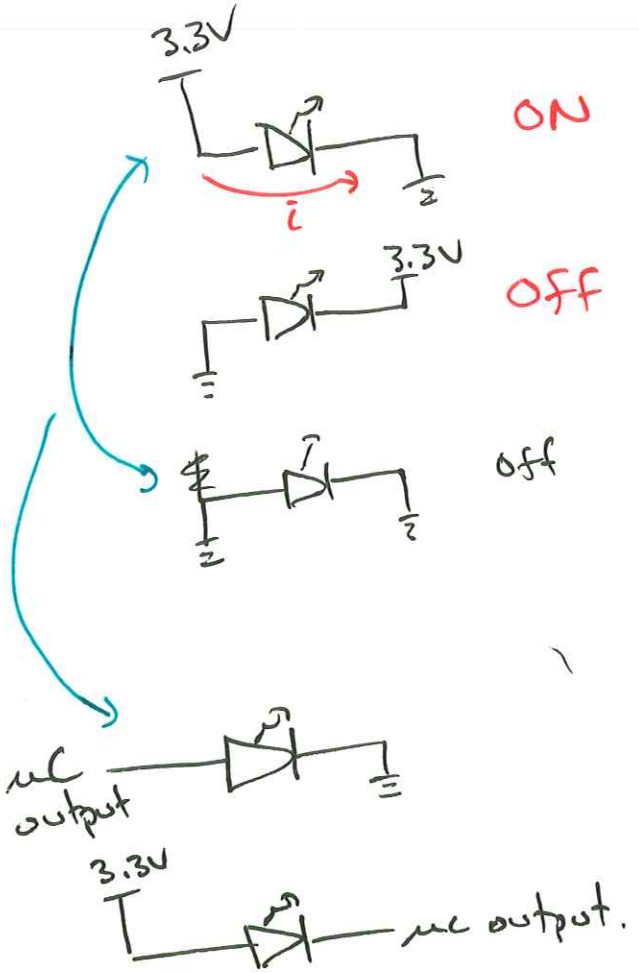
e.g. < 10mA

LED



can only flow in this direction

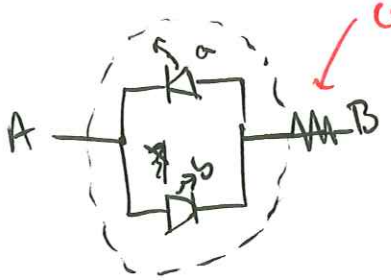
current flowing → produce light.



Need current limiting Resistor! (to prevent LED from failing)

↑ current limiting Resistor.

Bicolor LED



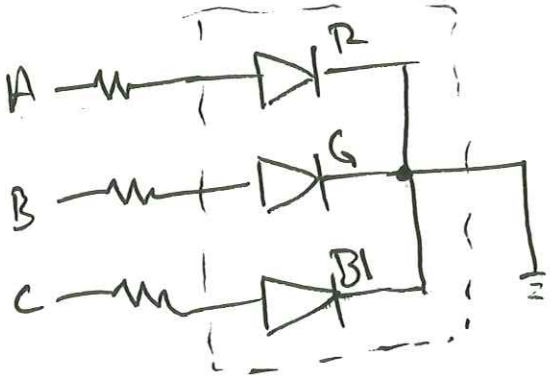
A	B	a	b
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

requires two outputs!

↑ Green?

↑ Red?

RGB LED



General Purpose Input/Output (GPIO)

→ Hardware pins on uC that can be Digital Inputs/Outputs

→ Most pins on uC will be GPIO

MSPM0G3507

- 64 total pins
- 60 GPIO (programming pins included!)
- Each of these coresponed to Bits 0 or 1 (single bit)
- Pins usually grouped into "ports"
  - 32 pins per port
  - 2 ports! Port A: pins 0-31 (32 total)
    Port B: pins 0-27 (28 total)

labelled as Pmn, m is port "A", "B"
n is pin 0-31

ex. PA2 is port A pin 2

ex. PB9 is port B pin 9

- All GPIO independent!

PA2 does not affect PA3

⋮

Special function Register (SFR)

variable that interacts with hardware functionality or memory region.

for Pin / Port Interface:

- → Direction (input or output)
- → Events (setup*)
- → Pullup / Pulldown Resistors*
- → Alternative function select*
- → Output type selection (PP vs. OD)
- → Set outputs
- → Read inputs (and outputs!)
- → Read Events*

Configuration.
* multiple bits per pin.
Most of these 1-bit-per-pin!

usage.

GPIO SFRs for MSPM0G3507 (all uint32_t)

DOE31_0 → configures pin to be an input or output. for a port. (bit=0 input, bit=1 output)
DOUT31_0 → set output value of a pin (if pin is output)
DIN31_0 → read values of pins (both inputs & outputs)

for port A:

GPIOA -> DOE31_0

use this as variable!

GPIOB -> DIN31_0

GPIOB -> DOE31_0 = 0x...76 ← 32-bit number, only looking at LSB

... 01110110

pin 0 is input
pin 4 is an output.
outputs!

GPIOB -> DOUT31_0 = 0b...00110100

pin 1 is low (0V)
pin 4 is high (3.3V)

Problem! Modifying registers as shown on previous page changes/sets all bits/pins! want to modify 1 at a time.

Ex. GPIOB->DOUT31_0 → Assume starts as unknown.

... X X X X X X X X X = 0 or 1 (unknown)

Bitmasking!

let's say we want to set pin 4 = 0, pin 1 = 1

... X X X 0 X X 1 X

set pin 4:

X X X X X X X X
& 1 1 1 0 1 1 1 1 = 0xEF

X X X 0 X X X X

set pin 1:

X X X 0 X X X X
| 0 0 0 0 0 0 1 0 = 0x02

X X X 0 X X 1 X

A	B	A & B	A | B
X	0	0	X
X	1	X	1

force bits LOW

force bits HIGH

Do similar things for DOE31_0

GPIOB->DOUT31_0 = GPIOB->DOUT31_0 & 0x...Ef;

OR GPIOB->DOUT31_0 &= 0x..Ef;

GPIOB->DOUT31_0 |= 0x...02;

ORDER DOES NOT MATTER! as long as pins aren't set in both.

Read value of pin PA7 — pin 7 in bits is ...0010000000

= 0x...080

want to isolate value of PA7

GPIOA→DIN31_0 → ... X x x x x x x x

isolate this bit!

best way: set all other bits to 0...

... X 0 0 0 0 0 0 0

```
if ( GPIOA -> DIN31_0 & 0x80 ) {
        :
}
```

Instead of Bitmasking...

Use "Drivers". for our case, the "EmCon HAL"

→ functions that do the bitmasking (and other things) for you...

for example:

```
GPIO_setOutput( GPIOA , GPIO_PIN_7);
```