

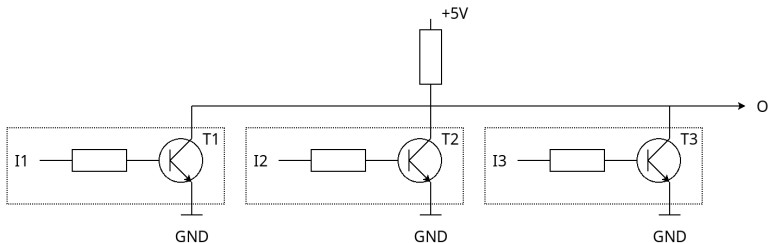
STD_LOGIC / IEEE 1164

Dr.-Ing. Volkmar Sieh

Department Informatik 4
Systemsoftware
Friedrich-Alexander-Universität Erlangen-Nürnberg

WS 2024/2025





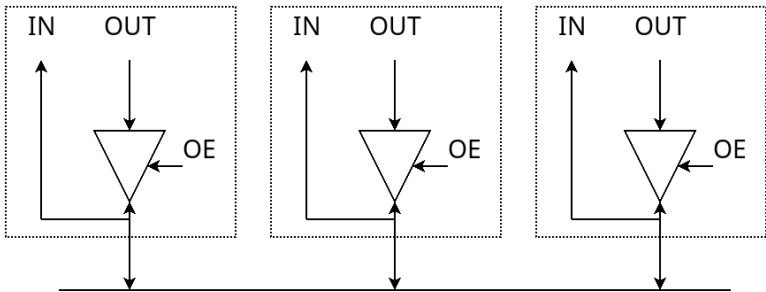
Output O ist $0V$ gdw. Transistor $T1$ *oder* $T2$ *oder* $T3$ durchgeschaltet ist (sonst ist O $5V$).

Also: $O = \text{nor}(I_1, I_2, I_3)$

Hier: Die drei Transistoren berechnen **gemeinsam** einen Wert O obwohl sie ggf. in **unterschiedlichen** Komponenten sitzen!

„wired-nor“



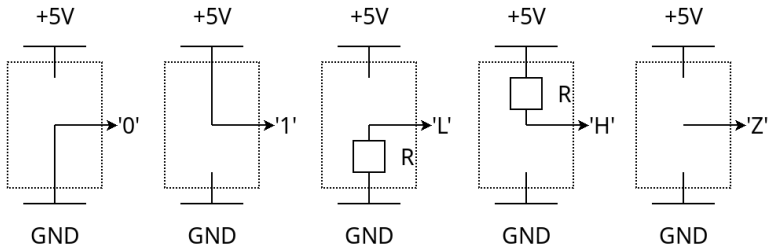


Komponenten müssen „0“, „1“ und „nichts“ ausgeben können.

OE: „Output Enable“

„tristate“





Mögliche Ausgaben einer Komponente:

'0' starke 0 (strong 0)

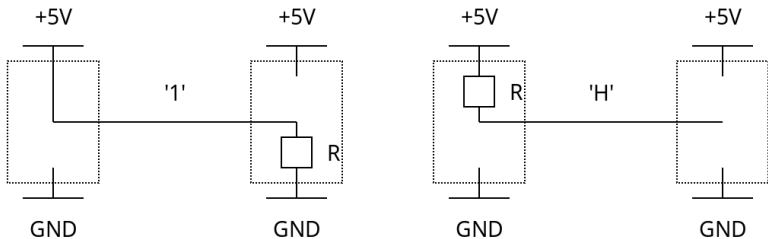
'1' starke 1 (strong 1)

'L' schwache 0; Pull-down-Widerstand (weak 0)

'H' schwache 1; Pull-up-Widerstand (weak 1)

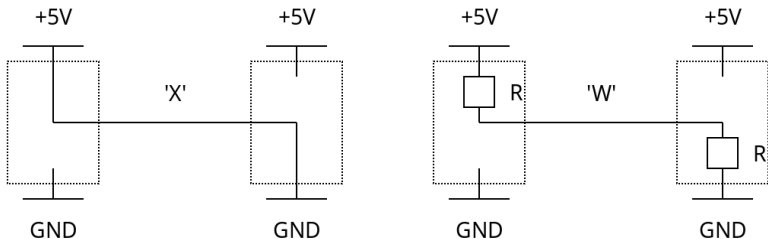
'Z' hochohmig (high impedance)





	'0'	'1'	'L'	'H'	'Z'
'0'	'0'	?	'0'	'0'	'0'
'1'	?	'1'	'1'	'1'	'1'
'L'	'0'	'1'	'L'	?	'L'
'H'	'0'	'1'	?	'H'	'H'
'Z'	'0'	'1'	'L'	'H'	'Z'





'X' undefiniert

'W' schwach undefiniert



	'0'	'1'	'L'	'H'	'Z'
'0'	'0'	'X'	'0'	'0'	'0'
'1'	'X'	'1'	'1'	'1'	'1'
'L'	'0'	'1'	'L'	'W'	'L'
'H'	'0'	'1'	'W'	'H'	'H'
'Z'	'0'	'1'	'L'	'H'	'Z'

Zusätzlich:

'U' uninitialisiert



Tabelle nach STD_LOGIC / IEEE 1164 (ohne '-'; don't care):

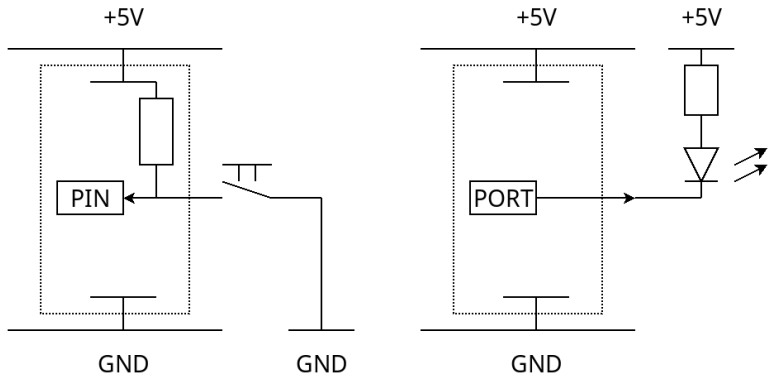
	'U'	'X'	'0'	'1'	'W'	'L'	'H'	'Z'
'U'	'U'	'X'	'U'	'U'	'U'	'U'	'U'	'U'
'X'	'X'	'X'	'X'	'X'	'X'	'X'	'X'	'X'
'0'	'U'	'X'	'0'	'X'	'0'	'0'	'0'	'0'
'1'	'U'	'X'	'X'	'1'	'1'	'1'	'1'	'1'
'W'	'U'	'X'	'0'	'1'	'W'	'W'	'W'	'W'
'L'	'U'	'X'	'0'	'1'	'W'	'L'	'W'	'L'
'H'	'U'	'X'	'0'	'1'	'W'	'W'	'H'	'H'
'Z'	'U'	'X'	'0'	'1'	'W'	'L'	'H'	'Z'



ATmega:

DDR-Bit	PORT-Bit	Ausgang	
'0'	'0'	'Z'	Input
'0'	'1'	'H'	Pull-up
'1'	'0'	'0'	Output
'1'	'1'	'1'	Output





Konfiguration:

Taster: DDR=0 (Input), PORT=1 (Pull-up), PIN=0/H

LEDs: DDR=1 (Output), PORT=0/1, (PIN=0/1)

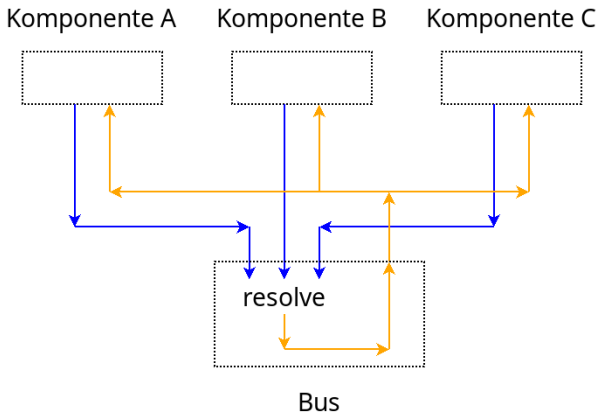


Framework-Implementierung stellt zur Verfügung:

- Konstanten `SIG_STD_LOGIC_0`, `SIG_STD_LOGIC_1`, ...
- Ausgabefunktion `sig_std_logic_set`
- Konfigurationsfunktionen `sig_std_logic_connect_{in,out}`
- Bus-Resolution



Bus-Resolution:



Framework-Implementierung ist so voreingestellt, dass `chip_atmel_atmega32_port_{a-d}{0-7}_out_set`-Methoden aufgerufen werden, wenn sich Signal am Port ändert.

Übergebener Wert `val` ist `SIG_STD_LOGIC_0` oder `SIG_STD_LOGIC_1` oder ...



Chip-Simulation muss SIG_STD_LOGIC_0, SIG_STD_LOGIC_1, usw. ausgeben, wenn sich

- PORT-Wert ändert oder
- DDR-Wert ändert



Update von DDR- bzw. PORT-Bit (pro Bit!):

```
...
if (ddr == 0) {
    /* Input */
    if (port == 0) {
        /* High impedance input. */
        sig_std_logic_set(cpssp->port_XX, cpssp, SIG_STD_LOGIC_Z);
    } else {
        /* Input with pull-up. */
        sig_std_logic_set(cpssp->port_XX, cpssp, SIG_STD_LOGIC_H);
    }
} else {
    /* Output */
    if (port == 0) {
        /* Output '0'. */
        sig_std_logic_set(cpssp->port_XX, cpssp, SIG_STD_LOGIC_0);
    } else {
        /* Output '1'. */
        sig_std_logic_set(cpssp->port_XX, cpssp, SIG_STD_LOGIC_1);
    }
}
...
```



Aufruf Callback (pro Bit!):

```
void
..._out_set(void *_cpssp, unsigned int val)
{
    switch (val) {
        case SIG_STD_LOGIC_0:
        case SIG_STD_LOGIC_L:
            PIN = 0;
            break;
        case SIG_STD_LOGIC_1:
        case SIG_STD_LOGIC_H:
            PIN = 1;
            break;
        default:
            ???
    }
}
```



Für AD-Wandler:

In val hineincodiert:

- Spannung (in mV)
- Stromstärke (in mA)

```
void
..._out_set(void *_cpssp, unsigned int val)
{
    unsigned int voltage = SIG_mV(val);
    unsigned int current = SIG_mA(val);

    ...
}
```

