

Praktikum 1

Aufgabe 1

a)

Gibt nichts zu tun

b)

Behaviour (Verhalten):

- Man definiert wie sich Schaltung verhalten soll
- z.B. C artige Berechnung (ähnlich wie Coding)

Dataflow (Datenfluss): <- Klingt fürs erste am angenehmsten für die Umsetzung

- wie Theoretische Informatik
- z.B. a xor b (sagt dem Compiler welche Logik die Schaltung haben soll)

Timed-Dataflow:

- wie Dataflow nur das auch Verzögerungen pro Berechnung angegeben werden können

Structure:

- sehr hardwarenah
- Mit genauen Portmaps z.B. welcher Ausgang auf welchen Eingang geht, dafür aber viel mehr Arbeit wie z.B. Dataflow

c)

ghdl --dir

d)

ghdl -r half_adder_tb --wave=halfadder_tb.ghw

- Executes the simulation and produces the wave file which can be viewed later

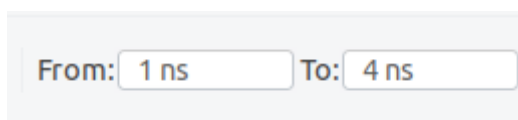
e)

gtkwave halfadder_tb.ghw

- view resulting wave file

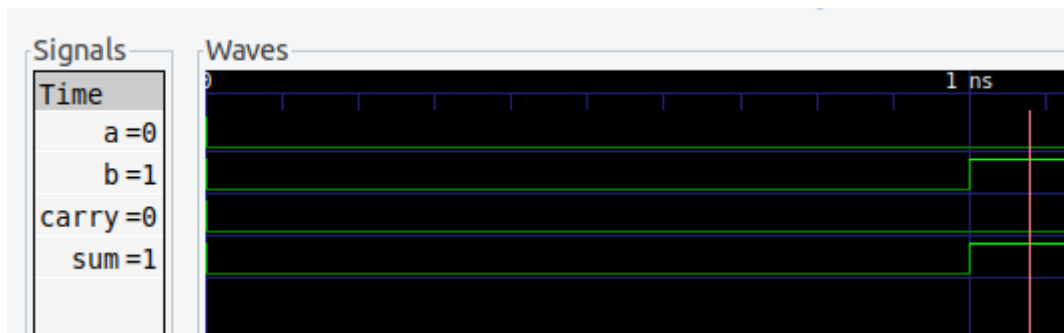
Signal auswählen: Doppelklick

Zeitinterval anzeigen:



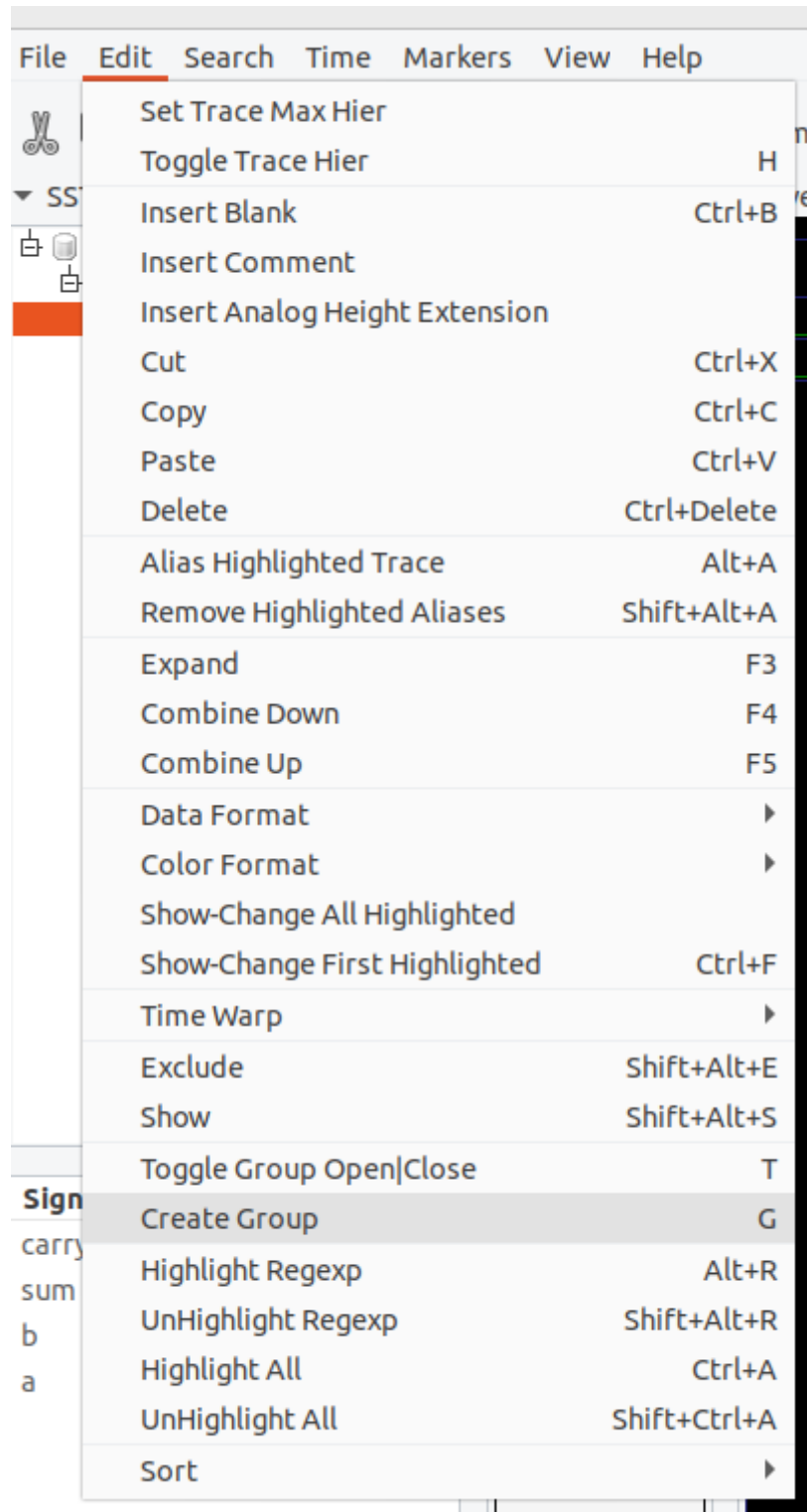
From: 1 ns To: 4 ns

Konkreten Wert anzeigen:

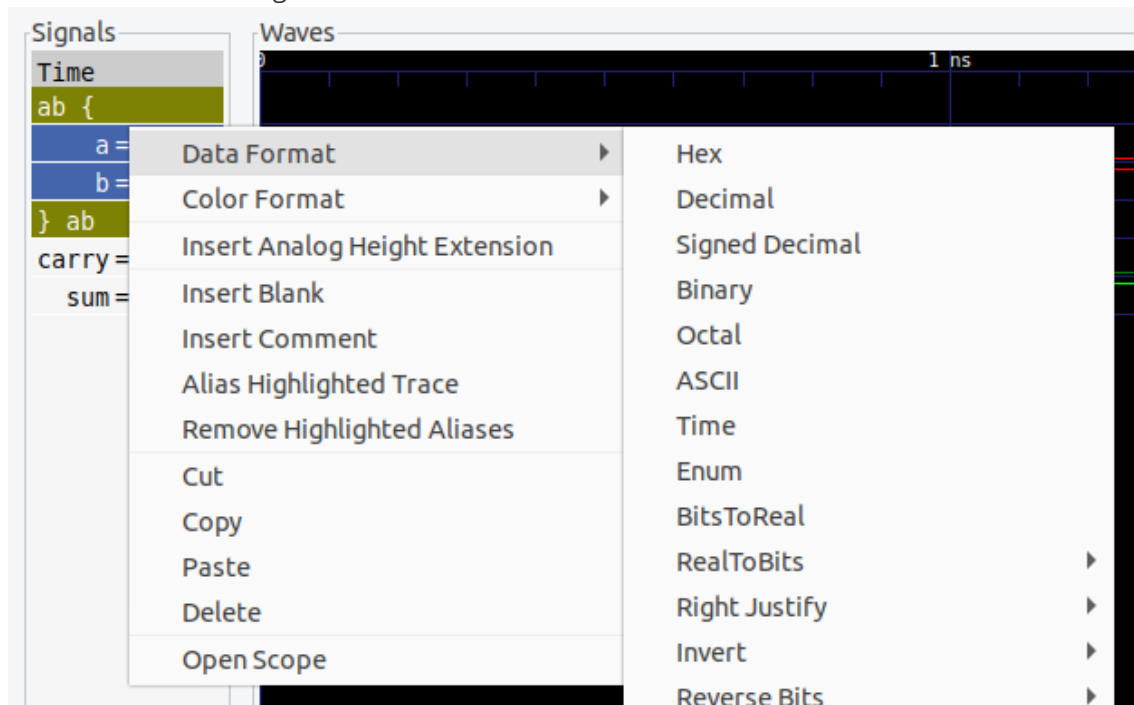


- Da wo geklickt wurde (orange Linie) wählt Zeitpunkt aus zu dem Werte links angezeigt werden

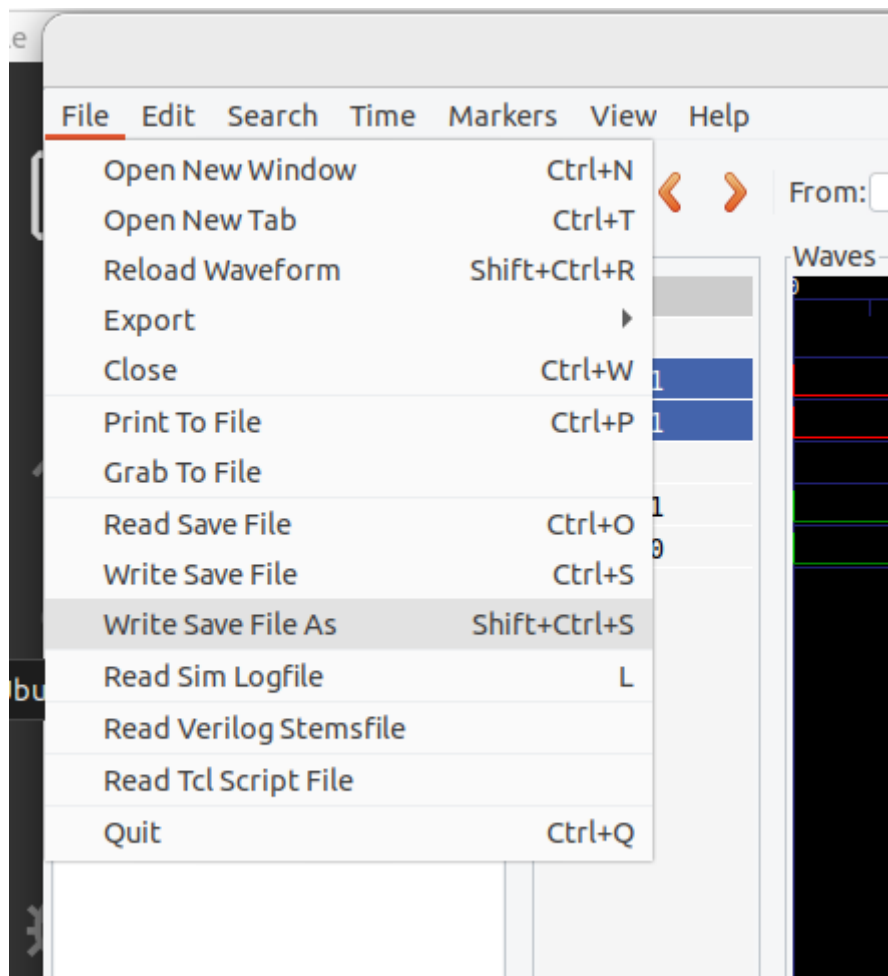
Mehrere bitsignale zusammenfassen:



- Gruppe mit gewünschten Signalen erzeugen (davor Signal mit strg + click auswählen die man in einer Gruppe haben möchte)
- Wert numerisch anzeigen:



- Datei speichern:



f)

sum als Ausgang entfernt und neu kompiliert (mit ghdl -a half_adder.vhdl):

```
mark@mark-VirtualBox:~/repos/hardwaresysteme$ ghdl -a half_adder.vhdl
half_adder.vhdl:22:5:error: no declaration for "sum"
    sum <= result(0);    -- output 'sum' = lower bit
    ^
half_adder.vhdl:32:3:error: no declaration for "sum"
    sum <= a xor b;
    ^
half_adder.vhdl:32:12:error: no function declarations for operator "xor"
    sum <= a xor b;
           ^
half_adder.vhdl:52:42:error: no declaration for "sum"
    I0: XOR2 port map(x => a, y => b, z => sum);
                                   ^
mark@mark-VirtualBox:~/repos/hardwaresysteme$
```

Aufgabe 2

a)

Verhalten scheint sich nicht zu ändern und eine Änderung des waits von 1 auf 10 der Testbench scheint auch nichts zu bewirken. AND/XOR schaltet immer noch sofort und ohne Verzögerung

b)

or2 kann von and2 zum Großteil abgeschaut werden

c)

fulladder programmieren

d)

Testbench für fulladder programmieren

e)

Todo: Testbench korrigieren da es scheint das noch was nicht ganz stimmt

f)

- **'U'**: Uninitialized – Dies zeigt an, dass das Signal noch keinen Wert erhalten hat.
- **'X'**: Unknown – Dies repräsentiert einen unbekannten Wert, der z.B. das Ergebnis eines Konflikts zwischen mehreren treibenden Signalen sein kann.

Danke @ChatGPT -> kontrollieren ob das stimmt

g)

AND/OR: 3ns

XOR: 2ns

$2 + 3 + 3 = 8\text{ns?}$

$f_{\text{max}} = 1 / T_{\text{max}}$

$= 1 / 8\text{ns}$

$= 125\text{ MHz?}$

h)

Bei $a = 1$, $b = 0$ und $c = 1$ -> Gesamtverzögerung relevant da der längste Pfad über XOR, AND, OR genommen werden muss (für Carry)

i)

Verzögerung überprüfen geht aktuell nicht, da wie oben beschrieben das nicht richtig funktioniert hat (mit der Verzögerung bei den Gattern)

j)

Gleiches Problem wie bei i mit der Verzögerung