cetz-timing

A Typst Package for Timing Diagrams

v0.1.0 2024-11-28

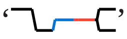
Johannes Schiffer

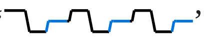
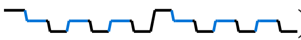
Table of contents

Part I.
Introduction

This package uses CeTZ to produce timing diagrams inside text.

It is a port of `tikz-timing` by Martin Scharrer to Typst.

The signal levels of the timing diagram can be given by corresponding characters/letters like '`H`' for *Logical High* or '`L`' for *Logical Low*. So e.g. '`HLZXD`' gives ''. In order to fit (in)to normal text size the diagram size (i.e. its height, width and line width) is defined relatively to the size of the character 'A' in the current context.

This way the diagram can also be scaled width the font size. (Example: Hello , Hello). A single timing character produces a diagram width a width identical to its height ('`H`' → ' '). Longer diagrams can be produced by either using the same character multiple times ('`HHH`' → ' ') or writing the width as number in front of the character ('`3H`' → ' '). Recurring character combinations can be repeated using character groups ('`3{HLZ}`' → '') . Character groups can be nested arbitrarily ('`2{H3{ZL}}`' → ).

Part II. Usage

II.1. Timing Characters

The logic levels are described by so called timing characters. Actually all of them are letters, but the general term character is used here. Table 1 shows all by default defined logic characters and Table 2 all possible two-character transitions.

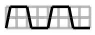
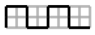
Character	Description	Diagram	Transition Example
`H`	High		
`L`	Low		
`Z`	High impedance		
`X`	Don’t care		
`D`	Data		
`U`	Unknown data		
`T`	Toggle	or	
`C`	Clock	or	
`M`	Metastable condition		
`G`	Glitch	-	-
`S`	Space	-	-

Table 1: Timing Characters

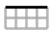
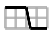
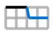
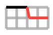
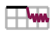
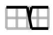
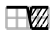
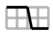
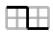
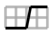
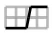
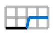
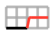
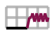
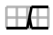
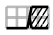
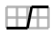
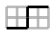
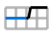
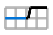
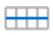
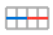
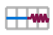
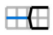
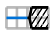
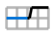
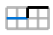
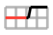
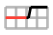
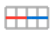
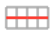
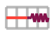
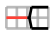
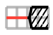
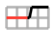
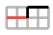
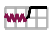
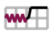
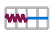
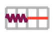
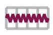
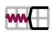
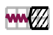
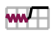
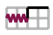
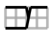
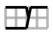
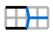
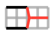
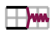
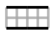
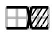
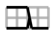
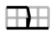
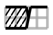
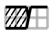
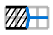
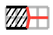
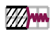
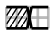
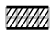
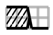
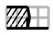
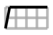
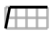
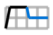
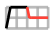
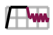
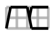
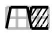
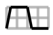
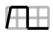
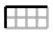
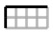
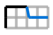
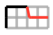
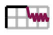
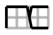
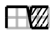
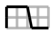
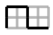
From	H	L	Z	X	M	D	U	T	C
H									
L									
Z									
X									
M									
D									
U									
T									
C									

Table 2: Overview over all transitions

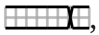
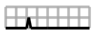
Modifier Syntax	Description
`D|D`	Produces an explicit transition. By default, repeating signals don’t have a transition. *E.g.:* ‘`3D|D`’ → , ‘`L|LLL`’ → 

Table 3: Modifiers for Timing Characters

II.2. Timing Diagram Table

Using the `timingtable` command, a timing diagram with several logic lines can be drawn to a CeTZ canvas.

The used layout is shown in Figure 1.

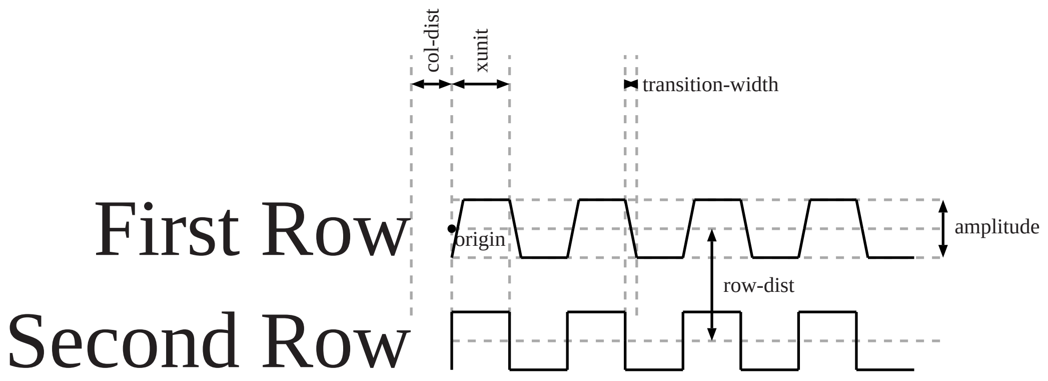


Figure 1: Distances inside a `timingtable`

Part III. Available Commands

`#texttiming(⟨strok⟩: 1pt + luma(0%), ⟨initchar⟩: none, ⟨draw-grid⟩: false, ⟨sequence⟩: string)`

This macro places a single timing diagram line into the current text. The signal have the same height has an uppercase letter (like 'X') of the current font, i.e. they scale with the font size. The macro argument must contain only valid logic characters and modifiers which define the logical levels of the diagram line.

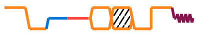
Argument

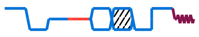
`⟨strok⟩: 1pt + luma(0%)` — stroke

Stroke of the diagram line. This does not affect `X`, `Z`, and `M` logic levels.

Note: I couldn't manage to get `stroke` to work, so it is named `strok` for now.

Examples

`#texttiming(strok: orange + 1pt, "HLZXDUTCM")` → 

`#texttiming(strok: blue + 1pt, "HLZXDUTCM")` → 

Argument

`⟨initchar⟩: none` — str

Initial logical level. This is used to draw a transition right at the beginning. It must be `none` or one of the logic levels.

Examples

`#texttiming(initchar: "L", "Z")` →

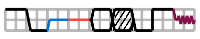
`#texttiming(initchar: "H", "Z")` →

Argument

`⟨draw-grid⟩: false` — bool

Draw a gray grid on the CeTZ canvas background.

Examples

`#texttiming(draw-grid: true, "HLZXDUTCM")` → 

Argument

`⟨sequence⟩` — str

The timing sequence to visualize.

```
#timingtable(
  ⟨col-dist⟩: 10pt,
  ⟨row-dist⟩: auto,
  ⟨xunit⟩: 2.0,
  ⟨amplitude⟩: 2.0,
  ⟨draw-grid⟩: false,
  ..⟨body⟩
)
```

This macro draws a timing diagram table to a CeTZ canvas.

Argument

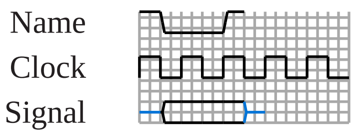
`⟨col-dist⟩: 10pt` length

The distance between columns.

Example

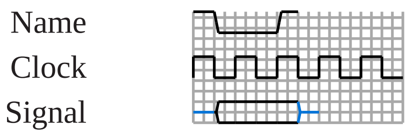
20 pt

```
#timingtable(col-dist: 20pt,
show-grid: true,
  [Name], [HLLLH],
  [Clock], [10{C}],
  [Signal], [Z4DZ],
)
```



40 pt

```
#timingtable(col-dist: 40pt,
show-grid: true,
  [Name], [HLLLH],
  [Clock], [10{C}],
  [Signal], [Z4DZ],
)
```



Argument

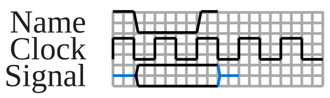
`⟨row-dist⟩: auto` length

The distance between rows.

Example

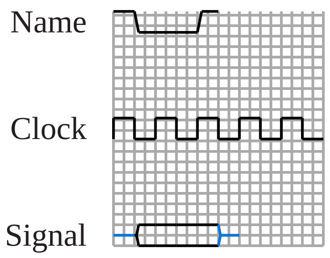
10 pt

```
#timingtable(row-dist: 10pt,
show-grid: true,
  [Name], [HLLLH],
  [Clock], [10{C}],
  [Signal], [Z4DZ],
)
```



40 pt

```
#timingtable(row-dist: 40pt,
show-grid: true,
  [Name], [HLLLH],
  [Clock], [10{C}],
  [Signal], [Z4DZ],
)
```



Argument

`⟨xunit⟩: 2.0` float

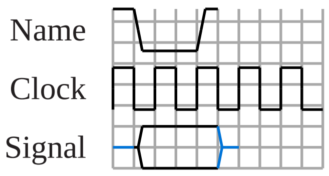
Number of CeTZ units per timing character.

Example

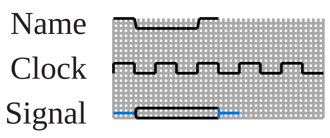
1.0

3.2

```
#timingtable(xunit: 1.0, show-
grid: true,
  [Name], [HLLLH],
  [Clock], [10{C}],
  [Signal], [Z4DZ],
)
```



```
#timingtable(xunit: 4.2, show-
grid: true,
  [Name], [HLLLH],
  [Clock], [10{C}],
  [Signal], [Z4DZ],
)
```



Argument

⟨amplitude⟩: 2.0 float

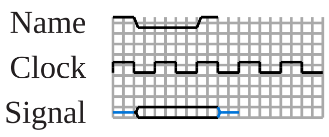
Signal aplitude (peak-to-peak) in CeTZ units.

The row distance is automatically adjusted.

Example

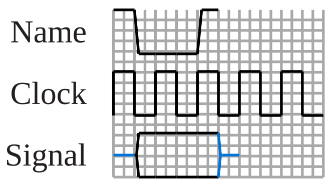
1.0

```
#timingtable(amplitude: 1.0,
show-grid: true,
  [Name], [HLLLH],
  [Clock], [10{C}],
  [Signal], [Z4DZ],
)
```



3.2

```
#timingtable(amplitude: 4.2,
show-grid: true,
  [Name], [HLLLH],
  [Clock], [10{C}],
  [Signal], [Z4DZ],
)
```



Argument

..⟨body⟩ none

Signal names and character sequences that describe the timing diagram.

```
#wave(
  ⟨origin⟩: (x: 0, y: 0),
  ⟨initchar⟩: none,
  ⟨stroke⟩: 1pt + luma(0%),
  ⟨xunit⟩: 2.0,
  ⟨amplitude⟩: 2.0,
  ⟨sequence⟩
)
```

Draw wave specified by character sequence in cetz.canvas context.

Example

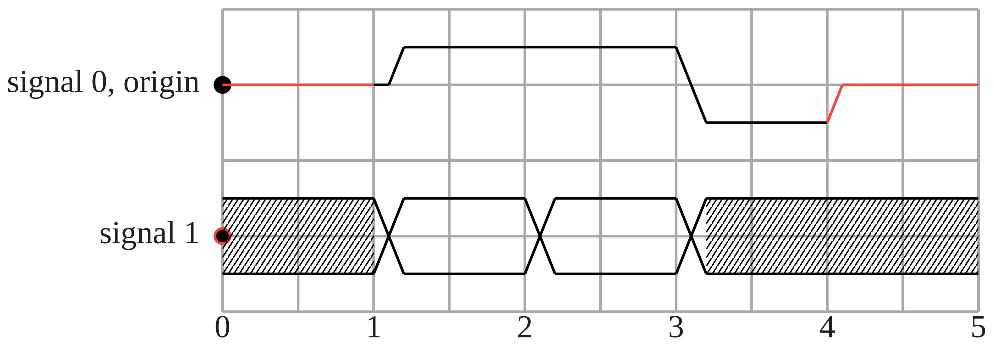
```
#cetz.canvas({
  import cetz.draw: *
```

```
    grid((0, 1), (10, -3), stroke: gray)

    circle((0, 0), radius: 0.1, fill: black, name: "origin")
    content("origin", anchor: "east", padding: .3, [signal 0, origin])
    wave("XHHLX")

    circle((0, -2), radius: 0.1, fill: black, name: "sig1")
    content("sig1", anchor: "east", padding: .3, [signal 1])
    wave(origin: (x: 0, y: -2), "UD|DUU")

    for i in range(6) {
      content((i * 2, -3.2), [#i])
    }
  })
```



Argument

⟨origin⟩: (x: 0, y: 0) dictionary

A dictionary that specifies the origin position on the CeTZ canvas.

Argument

⟨initchar⟩: none str

Initial logical level. This is used to draw a transition right at the beginning. It must be none or one of the logic levels.

Argument

⟨stroke⟩: 1pt + luma(0%) stroke

Stroke of the wave.

Argument

⟨xunit⟩: 2.0 float

Size of one character in the sequence in CeTZ coordinate space.

Argument

⟨amplitude⟩: 2.0 float

Size of the peak-to-peak amplitude in CeTZ coordinate space.

Argument

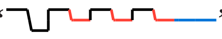
⟨sequence⟩ str

The character sequence to visualize.

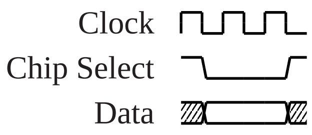
Part IV.
TODO

- Add data labels: `D[MISO]`. `content` in braces.
- Add CeTZ anchors for diagram.
- Add optional CeTZ anchors for individual signals: `D<miso>`, `D<miso>[MISO]`.
- Make anchors available so users can do custom arrows and annotations -> leave drawing CeTZ `canvas` to the user?
- Apply color to `U` pattern.
- Add option to omit first column of timing table.
- [*Optional*] Add caption to timing table.
- [*Optional*] Add table header to timing table.
- [*Optional*] Add tick marks.
- [*Optional*] Add grouping of table rows.
- [*Optional*] Add highlighting of row groups and ticks.
- [*Optional*] Correct `strok` argument.
- [*Optional*] Resolve `mantys` warnings.
- [*Optional*] Allow non-integer lengths for logic levels.

Part V.
Examples

I am an inline timing diagram: ‘’.

I am a basic timing diagram with multiple rows:



Part VI. Index