How to use Cir2Tikz

Xiaoning Bian
Tsinghua University

March 27, 2026

1 Introduction

Cir2Tikz is tool for converting a circuit encoding to a tikz file for use in Latex and visual editing in TikZiT.
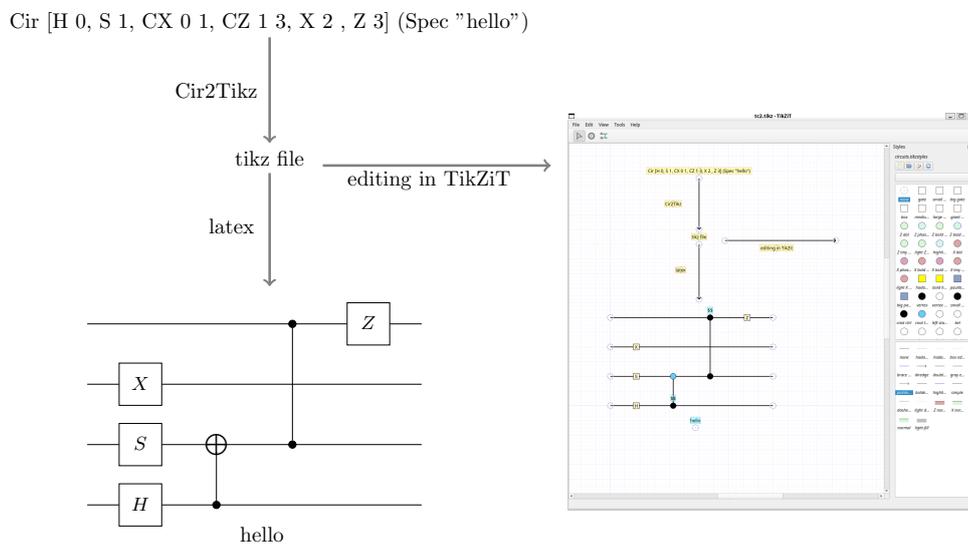

Figure 1: Left part of this figure is drawn in TikZiT whose GUI is the right part

2 Steps of using Cir2Tikz

1. Open the app here: `https://onlinegdb.com/ZV-cO_wwF`

2. Modify "your_cir" (around line 12) in the app as you like.

3. Click "Run" button on the top left corner of the app. Try again if it outputs nothing.

4. Select the output starting with "\begin" line ending with "\end" line; Copy it; Paste it to an empty .tikz file; Save.

3 Compiling with latex

This is the minimal tex file setup:

```
\documentclass{article}

\usepackage{tikzit}
\input{circuits.tikzstyles}

\definecolor{normGreen}{RGB}{214, 243, 221}
```

```
\definecolor{normRed}{RGB}{249, 196, 196}
```

```
\begin{document}
```

```
 \scalebox{0.8}{\tikzfig{test_cir}}
```

```
\end{document}
```

That means you need download the file "circuits.tikzstyles" and save it in the same folder as your .tex file. Here is the file:

https://www.dropbox.com/scl/fi/5ubc4uo7g4z6yv7cmyp9p/circuits.tikzstyles?rlkey=abndq6652j3t1gawzpbg2c1ud&st=lezzil4l&dl=0

Similarly, download and save the file "tikzit.sty" here:

https://www.dropbox.com/scl/fi/riqyuvj1sbn48osxbnlk9/tikzit.sty?rlkey=y4zhykbgq3y5adkk0q6llqx3e&st=5bztui95&dl=0

Note that sometime you might need to install missing packages such as `xcolor`.

4   Editing in TikZiT

1. Install TikZiT from here: `https://tikzit.github.io/`

2. Load the style file "circuits.tikzstyles" you just downloaded by clicking this button:
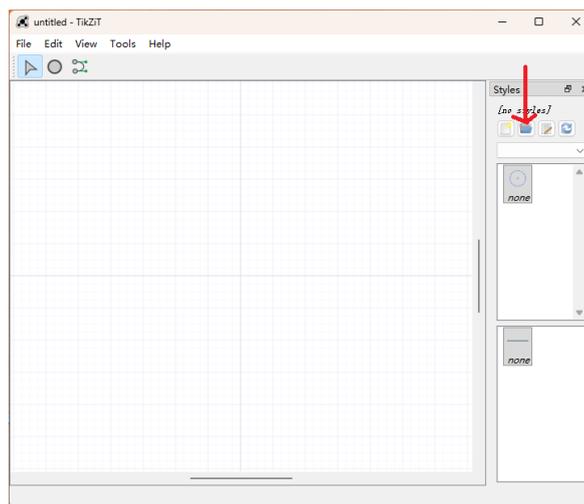
3. Click "File → Open" to open your tikz file and then edit it.

4. Click "Tools → Make Preview" to preview your circuit. (For this to work, you need latex installed in your system so that TikZiT can call it; see Previewing for help) A preview would look like:

Cir [H 0, S 1, CX 0 1, CZ 1 3, X 2 , Z 3] (Spec "hello")

Cir2Tikz

tikz file → editing in TikZit

latex

hello
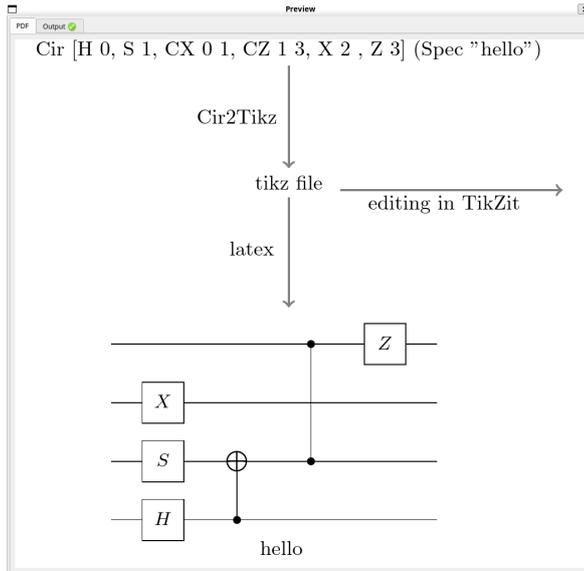
5 Advanced use

5.1 Draw a list of circuit equations (with specifications)

For example, the encoding of a list of equations in Fig 2: will be compiled to Fig 3.

```
eg_rels =
    [
    (Rel CT.Symplectic (Cir [CZ 0 1, D 1 "{0b}", D 0 "{0d}"] (Spec "")) (Cir [D 1 "{0b}", D 0 "{0d}", CZ 1 2] (Spec "")) (Spec "")),
    (Rel CT.Symplectic (Cir [CZ 0 1, D 1 "{ab}", D 0 "{0d}"] (Spec "")) (Cir [D 1 "{ab}", D 0 "{0,d-a}", Sep, He 2 "3", CZ 1 2, H 2] (Spec "")) (Spec "$a\\neq 0$"))
    (Rel CT.Symplectic (Cir [CZ 0 1, D 1 "{0b}", D 0 "{cd}"] (Spec "")) (Cir [D 1 "{0,b-c}", D 0 "{c,d}" ,He 1 "3", CZ 1 2, H 1] (Spec "")) (Spec "$c\\neq 0$")),
    (Rel CT.Symplectic (Cir [CZ 0 1, D 1 "{ab}", D 0 "{cd}"] (Spec "")) (Cir [D 1 "{a,b-c}", D 0 "{c,d-a}", Sep, He 1 "3", Se 1 "-a/c", He 2 "3", Se 2 "-c/a", CZ 1
    ]
```

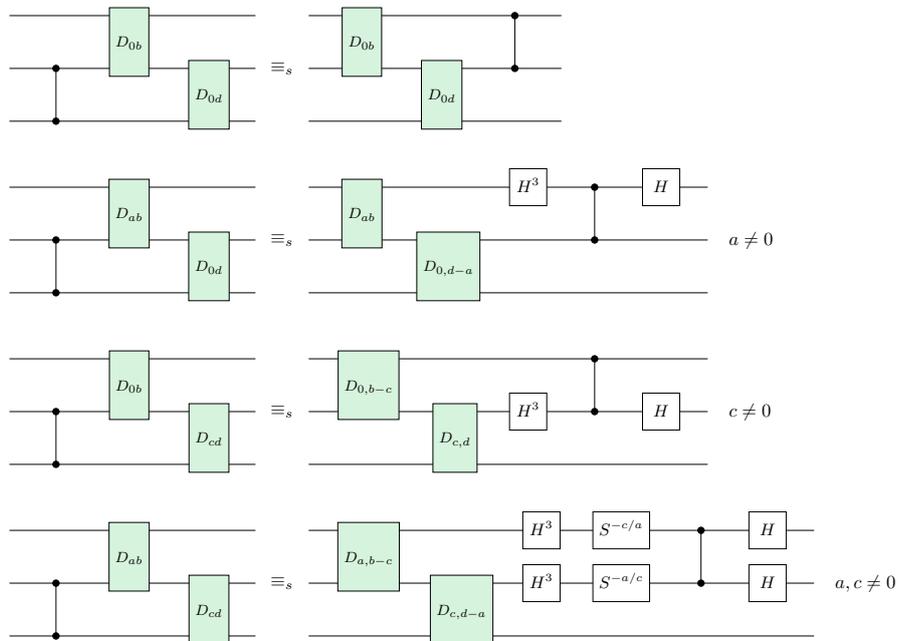Figure 2: Encoding of a list of circuit equations

Figure 3: Draw a list of circuit equations

5.2 Draw a list of positioned circuits

For example, the encoding of a list of positioned circuits in Fig 4 will be compiled to Fig 5.

```
eg_pcirs :: [(Cir , PositionSpec)]
eg_pcirs =
  [
    (Cir eg1 (Spec "eg1") , PositionSpec Up "\\Rightarrow"),
    (Cir eg2 (Spec "eg2") , PositionSpec Right "\\mapsto"),
    (Cir eg1 (Spec "eg1 again") , PositionSpec RU "\\rightarrow"),
    (Cir eg2 (Spec "eg2 again") , PositionSpec RU "\\rightarrow")
  ]
```

Figure 4: Encoding of a list of positioned circuits

Note that the supported "directions" are `Up` , `Down` , `Right` , `Left` , `LU` , `LD` , `RU` , `RD`. Also note that in `[(cir1, pspec1), (cir2, pspec2) ... ]`, `pspec1` specifies the relative position of `cir2` w.r.t `cir1`, and the connective between them. The last pspec is discarded.
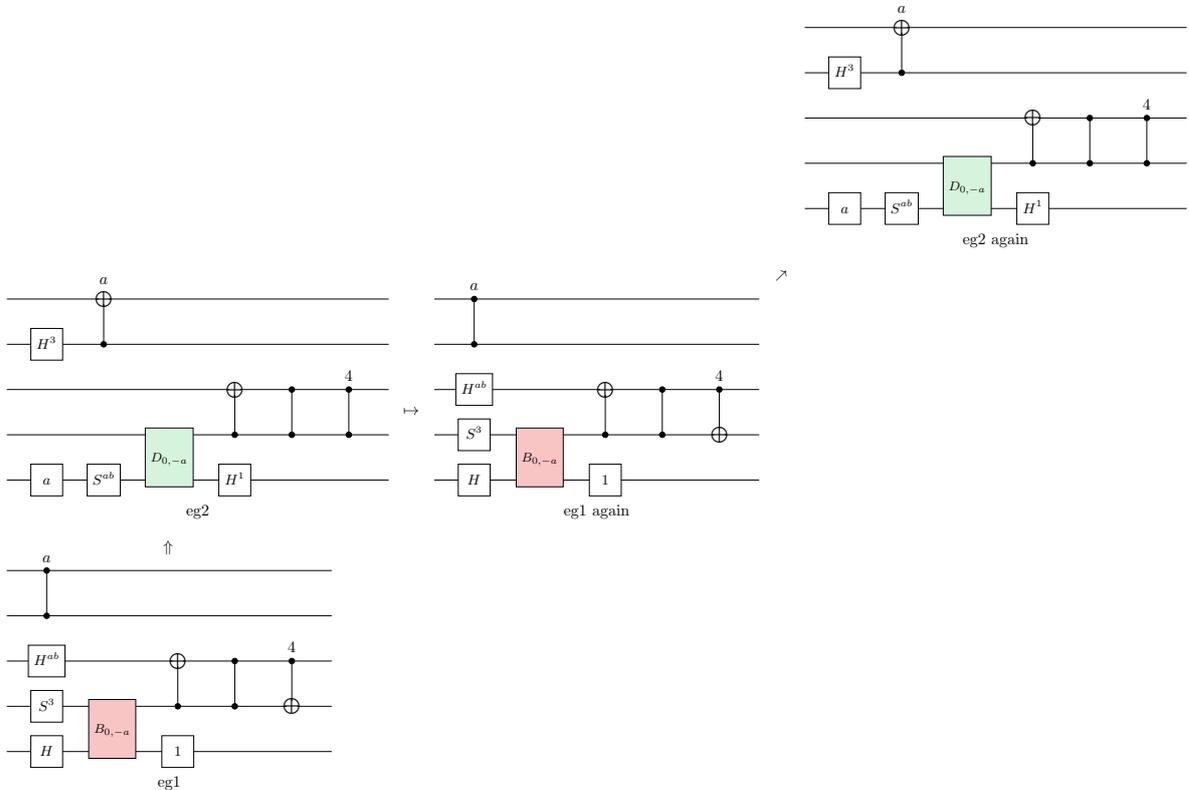

Figure 5: Draw a list of positioned circuits

6 Pros and Cons

6.1 Pros

1. Lightweight: the number of loc is less than 1000.

2. Qiskit's `latex_code = qc.draw(output='latex_source')` outputs `qcircuit` format .tex code which lacks TikZiT visual editing tool.

3. `qcircuit` format .tex code also lacks support for drawing multiple circuit in one figure. So it cannot support advanced use as in Section 5.

4. Extensible: we welcome customization requests to suit your specific needs.

6.2   Cons

1. Limited functionality: for the moment, it does not support measurement, initialization, termination etc. It only supports unitary circuit.

2. Coded in Haskell which is less used.

3. This is the first release. So it might be buggy.