

xbondgraphs*– drawing bond graphs using TikZ

Marcus J.W. Snippe[†]

May 4, 2018

Abstract

When using the `xbondgraphs`-package, the user is able to draw visually pleasing bond graphs¹, while mostly maintaining the standard notation of TikZ drawings. It defines two new PGF arrows, an accompanying decoration to ensure the direction of the barb, as well as a PGF shape for power (de-)mux elements. This package is based on the `bondgraphs` package by G. Folkertsma², but does not (yet) cover all its functions. It *might* result in more appealing bond graphs.

Contents

1	Introduction	2
1.1	Motivation	2
1.2	Alternatives	2
1.3	Known issues	3
2	Basic usage	3
2.1	Installation	3
2.2	Including the package	3
2.3	Simple example	3
3	Options	4
3.1	Global (package) options	4
3.2	Local (TikZ) options	5
4	Arrow tips	5
4.1	Single bond arrow tip	5
4.2	Multi bond arrow tip	6

*This document corresponds to `xbondgraphs` v0.0.1, dated 2018/05/02.

[†]E-mail: m.j.w.snippe@saxion.nl

¹https://en.wikipedia.org/wiki/Bond_graph

²<https://ctan.org/pkg/bondgraphs>

5 Examples	6
6 Implementation	7
6.1 Package definition	7
6.2 Required packages and libraries	7
6.3 Arrow tip definitions	7
7 Change History	15
8 Index	15

1 Introduction

1.1 Motivation

This package is a by-product of a project in which I was in need of a convenient way to draw bond graphs. At first, the `bondgraphs` package was sufficient, but as the delivery date of the final report approached, I became less and less satisfied by the aesthetic end result of my bond graphs, especially when using multi-bonds. Figure 1 shows a simple comparison between the `bondgraphs`- and the `xbondgraphs` package.

(a) Using the `bondgraphs` package (b) Using the `xbondgraphs` package

Figure 1 – Comparison of multi bond graph drawing.

Figure 1 shows the main motivation for this package. Although of course subjective, most of the differences between the `bondgraphs`- and the `xbondgraphs` package can be argued to be improvements. The drawing in figure 1b is overall more consistent. The causality stroke of figure 1a with flow-out causality is over-drawn by the inner line of the multi bond. This is fixed in figure 1b. Most flaws of the drawing in figure 1a can be traced back to the decoration being a postaction. This however is needed to inherit other options from the `\draw`-command, e.g. color.

Due to these reasons, I wrote the `xbondgraphs` package from scratch, re-using some parts but in a completely different setup.

1.2 Alternatives

As already mentioned, this package is based on the `bondgraphs` package, but does not (yet) cover all its functions. A comparison of main package functions is shown in table 1.

³See figure 1.

⁴This is optional.

Table 1 – Function comparison between bondgraphs and xbondgraphs.

	bondgraphs	xbondgraphs
Automatic arrow barb direction	✓	✓
Single bond drawings	✓	✓
Multi bond drawings ³	✓	✓
Power (de-)mux element	✗	✓
Multi-segment bonds	✗	✓
Curly bond barb	✓	✗
Colon between element and variable ⁴	✗	✓

A second alternative is the bondgraph⁵ package, but because it has nearly no documentation and an incomprehensible example file, I have never tried it personally.

1.3 Known issues

- None yet, but please submit issues to <https://github.com/MaxSnippe/xbondgraphs/issues>.

2 Basic usage

2.1 Installation

This package has not yet been included in popular L^AT_EX distributions, and therefore can be installed only by downloading the source (xbondgraphs.sty) from [the GitHub repository](#)⁶ to your local TEXMF tree. It should be placed under \$TEXMF\$/tex/latex/local.

2.2 Including the package

The package can be included with the well-known `\usepackage[<options>]{xbondgraphs}`, where *<options>* can be any of the options mentioned in section 3.1. Options that set the same keys to different values are treated in the order in which they are provided. The package works fine straight-out-of-the-box without setting any options.

2.3 Simple example

A simple example of an electric domain dynamic model shown as an iconic diagram, and its domain independent equal model shown as a bond graph.

⁵<https://ctan.org/pkg/bondgraph>

⁶<https://github.com/MaxSnippe/xbondgraphs>

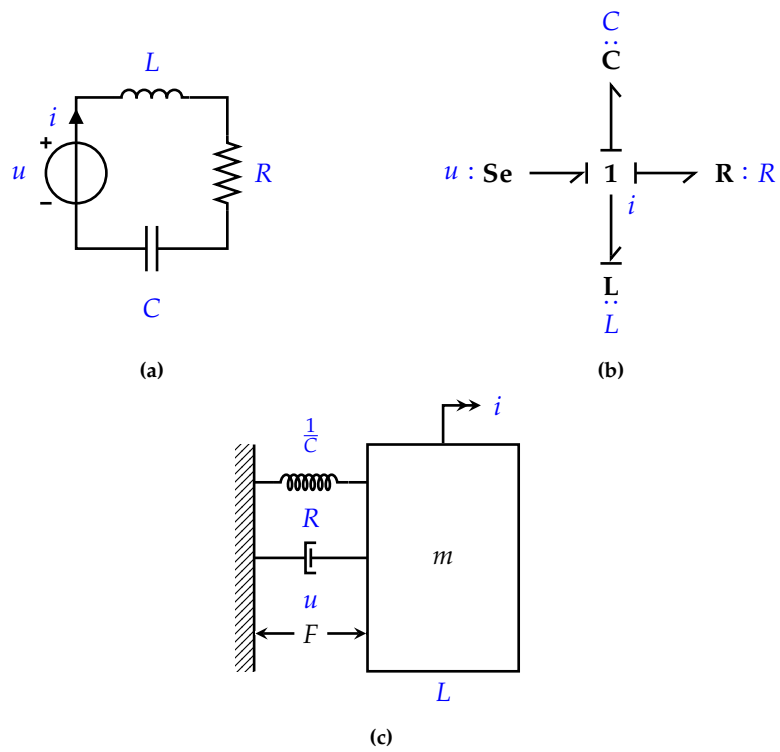
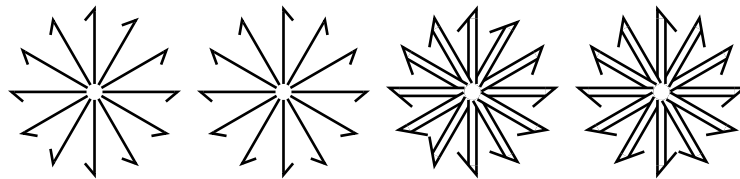


Figure 2 – Electric domain dynamic model and its bond graph representation.

3 Options

3.1 Global (package) options

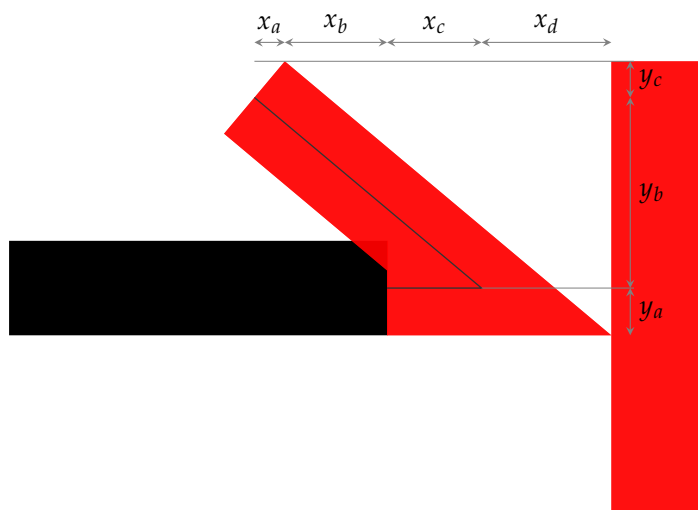
barbdirection



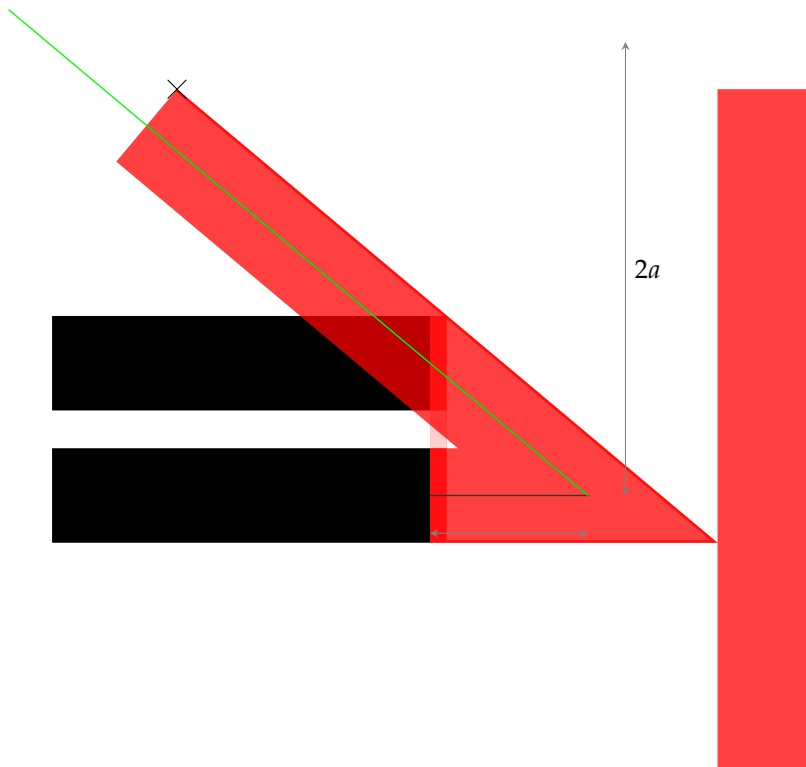
3.2 Local (TikZ) options

4 Arrow tips

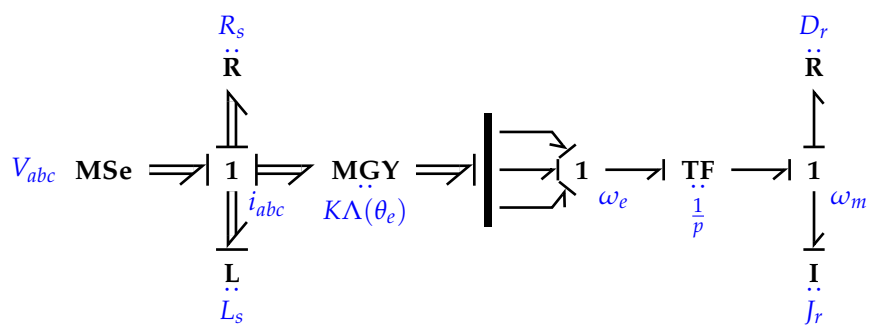
4.1 Single bond arrow tip



4.2 Multi bond arrow tip



5 Examples



6 Implementation

6.1 Package definition

```
1 \*package>
2 \NeedsTeXFormat{LaTeX2e}[2017/04/15]
3 \ProvidesPackage{xbondgraphs}
4 [2018/05/02 v0.0.1 Bond graph drawing using TikZ]
```

6.2 Required packages and libraries

This package uses the pgfplots package to be able to use pgfkeys as package options. All the actual drawing is done by TikZ. The amsfonts package is used for the `\mathbb` font.

```
5 \RequirePackage{pgfplots}
6 \RequirePackage{tikz}
7 \RequirePackage{amsfonts}
8
9 \usetikzlibrary{arrows.meta,decorations.markings,shapes}
```

6.3 Arrow tip definitions

The arrow tips are defined using `\pgfdeclarearrow`.

Single Bond Barb First the single bond barb is defined. The definition of this arrow is elaborated in section 4.1.

```
10 \pgfdeclarearrow{
11   name = {Single Bond Barb},
12   setup code = {
```

First locally define the line width of a single bond, a multibond, and the (absolute) angle the barb makes with the bond.

```
13   \pgfmathsetlengthmacro{\sbw}{\pgflinewidth}
14   \pgfmathsetlengthmacro{\mbw}{\xbondgraphs@multibondwidth}
15   \pgfmathsetlengthmacro{\ba}{\xbondgraphs@barbangle}
```

Calculate the x - and y position of the points that the barb will follow. If one was walking along the bond from startpoint to endpoint, the origin of this scope would be the endpoint, the x direction would be forward, and the y direction would be leftward.

```
16   \pgfmathsetlengthmacro{\tipx}{\sbw}
17   \pgfmathsetlengthmacro{\tipy}{0pt}
18   \pgfmathsetlengthmacro{\backx}{-1/\tan(\ba)*(\mbw-0.5*\cos(\ba)*\sbw)%
19     + \sbw}
20   \pgfmathsetlengthmacro{\backy}{\mbw - 0.5*\cos(\ba)*\sbw}
```

PGF needs the outer points of the arrow tip to accurately determine the bounding box. Also, the actual tip of the arrow is needed, so the drawn bond will end exactly at the endpoint (the bond TikZ styles use a `shorten > = <dimen>` and `shorten > = <dimen>` so they will not end exactly at the endpoint).

```

21 \pgfmathsetlengthmacro{\hullpointx}{\backx + 0.5*\sbw*sin(\ba)}
22 \pgfmathsetlengthmacro{\hullpointy}{\mbw}
23 \pgfmathsetlengthmacro{\tipendx}{0.5*\sbw/tan(\ba/2) + \tipx}
24 \pgfmathsetlengthmacro{\tipendy}{-0.5*\sbw}

```

These commands are used to set the outer dimensions that TikZ/PGF needs.

```

25 \pgfarrowssettipend{\tipendx}
26 \pgfarrowssetbackend{\backx}
27 \pgfarrowssethullpoint{\hullpointx}{\hullpointy}
28 \pgfarrowssethullpoint{\tipendx}{\tipendy}
29 },
30 drawing code = {

```

The actual drawing of the arrow.

```

31 \pgfpathmoveto{\pgfpointorigin}
32 \pgfpathlineto{\pgfpoint{\tipx}{\tipy}}
33 \pgfpathlineto{\pgfpoint{\backx}{\backy}}
34 \pgfusepathqstroke
35 },
36 }

```

Multi Bond Barb Repeat all for the multi bond barb. The definition of this arrow is elaborated in section 4.2.

```

37 \pgfdeclarearrow{
38   name = {Multi Bond Barb},
39   setup code = {

```

Note that the single bond line width is now read from its PGF key and not from `\pgflinewidth`. The latter now holds the multi bond line width.

```

40 \pgfmathsetlengthmacro{\sbw}{\xbondgraphs@singlebondwidth}
41 \pgfmathsetlengthmacro{\mbw}{\pgflinewidth}
42 \pgfmathsetlengthmacro{\ba}{\xbondgraphs@barbangle}

```

The starting point of the drawing of the actual arrow tip is now were the ‘bottom’ line ends. The tip end location is calculated such that the centerline of the barb passes through the endpoint of the ‘top’ double line.

```

43 \pgfmathsetlengthmacro{\startx}{0pt}
44 \pgfmathsetlengthmacro{\starty}{-0.5*\mbw+0.5*\sbw}
45 \pgfmathsetlengthmacro{\tipx}{(\mbw-\sbw)/tan(\ba)}
46 \pgfmathsetlengthmacro{\tipy}{-0.5*\mbw + 0.5*\sbw}
47 \pgfmathsetlengthmacro{\backy}{1.5*\mbw - 0.5*\sbw*cos(\ba)}
48 \pgfmathsetlengthmacro{\backx}{-(\backy+\tipy)/tan(\ba)}

```

The outer dimensions of the arrow are slightly different than for the Single Bond Barb, but not much.

```

49 \pgfmathsetlengthmacro{\hullpointx}{\backx + 0.5*\sbw*sin(\ba)}
50 \pgfmathsetlengthmacro{\hullpointy}{1.5*\mbw}
51 \pgfmathsetlengthmacro{\tipendx}{0.5*\sbw/tan(\ba/2) + \tipx}
52 \pgfmathsetlengthmacro{\tipendy}{-0.5*\mbw}

```

Again set the PGF dimensions needed for the definition of the arrow tip.


```

53 \pgfarrowssettipend{\tipendx}
54 \pgfarrowssetbackend{\backx}
55 \pgfarrowshullpoint{\hullpointx}{\hullpointy}
56 \pgfarrowshullpoint{\tipendx}{\tipendy}
57 },
58 drawing code = {

```

The drawing is the same as for the Single Bond Barb, except for the \pgfsetlinewidth that sets the line width to the single bond line width.

```

59 \pgfpathmoveto{\pgfpoint{\startx}{\starty}}
60 \pgfpathlineto{\pgfpoint{\tipx}{\tipy}}
61 \pgfpathlineto{\pgfpoint{\backx}{\backy}}
62 \pgfsetlinewidth{\sbw}
63 \pgfusepathqstroke
64 }
65 }

66
67 % BOND DECORATION
68 \pgfdeclaredecoration{bond}{initial}{
69 \state{initial}[width=\pgfdecoratedinputsegmentlength+1pt]{
70 \pgfpathlineto{\pgfpointdecoratedinputsegmentlast}
71 }
72 \state{final}{
73 %
74 \pgfmathparse{int((\pgfdecoratedangle+\xbondgraphs@bond@barbdirectionflipangle)/90)}
75 \ifcase\pgfmathresult
76 \pgfkeys{/xbondgraphs/bond/barb direction=right}
77 \or
78 \pgfkeys{/xbondgraphs/bond/barb direction=left}
79 \or
80 \pgfkeys{/xbondgraphs/bond/barb direction=left}
81 \else
82 \pgfkeys{/xbondgraphs/bond/barb direction=right}
83 \fi
84 \ifxbondgraphs@bond@causality@eout
85 \tikzset{-{\xbondgraphs@bond@barbarrowhead[\xbondgraphs@bond@barbdirection]}|[/tikz/causal
86 \else
87 \ifxbondgraphs@bond@causality@fout
88 \tikzset{{|[/tikz/causal stroke style]}-{\xbondgraphs@bond@barbarrowhead[\xbondgraphs@bond@
89 \else
90 \tikzset{-{\xbondgraphs@bond@barbarrowhead[\xbondgraphs@bond@barbdirection]}}
91 \fi
92 \fi
93 \path[/xbondgraphs/bond/template]\pgfextra{\pgfpathlineto{\pgfpointdecoratedinputsegmentlas
94 }
95 }
96
97 % ifs for the bond options
98 \newif\ifxbondgraphs@bond@causality@eout

```

```

99 \newif\ifxbondgraphs@bond@causality@fout
100
101 % ifs for the element options
102 \newif\ifxbondgraphs@element@word
103 \newif\ifxbondgraphs@element@multiport
104
105 % Define 'xbondgraphs' as key family for this package
106 \pgfkeys{
107   xbondgraphs/.is family,
108   xbondgraphs,
109   % Two key families are mainly used, first is 'bond':
110   bond/.is family,
111   bond,
112   template/.style={
113     shorten < = 3pt,
114     shorten > = 3pt,
115     draw,
116     line width = \xbondgraphs@singlebondwidth,
117   },
118   barb direction/.store in=\xbondgraphs@bond@barbdirection,
119   barb direction flip angle/.store in=\xbondgraphs@bond@barbdirectionflipangle,
120   eout/.is if=xbondgraphs@bond@causality@eout,
121   eout=false,
122   fout/.is if=xbondgraphs@bond@causality@fout,
123   fout=false,
124   effort out/.code=\pgfkeys{
125     /xbondgraphs/bond/.cd,
126     eout=true,
127     fout=false,
128     /tikz/causal stroke style/.append style={#1}
129   },
130   flow out/.code=\pgfkeys{
131     /xbondgraphs/bond/.cd,
132     eout=false,
133     fout=true,
134     /tikz/causal stroke style/.append style={#1}
135   },
136   effort in/.code=\pgfkeys{/xbondgraphs/bond/flow out={#1}},
137   flow in/.code=\pgfkeys{/xbondgraphs/bond/effort out={#1}},
138   multi/.code=\pgfkeys{
139     /xbondgraphs/bond/causality stroke scale=3,
140     /xbondgraphs/bond/barb arrow head={Multi Bond Barb},
141     /xbondgraphs/bond/template/.append style={
142       double,double distance={\xbondgraphs@multibondwidth-2*\xbondgraphs@singlebondwidth}
143     },
144     /tikz/line width = \xbondgraphs@multibondwidth,
145   },
146   causality stroke scale/.store in=\xbondgraphs@causalitystrokescale,
147   causality stroke scale=2,
148   barb arrow head/.store in=\xbondgraphs@bond@barbarrowhead,

```

```

149 barb arrow head={Single Bond Barb},
150 label/.style = {
151   \xbondgraphs@bondlabelcolor,
152 },
153 /xbondgraphs,
154 % Second key family is 'element':
155 element/.is family,
156 element,
157 n/.store in=\xbondgraphs@element@n,
158 n=1,
159 word/.is if=\xbondgraphs@element@word,
160 word=false,
161 multiport boolean/.is if=\xbondgraphs@element@multiport,
162 multiport boolean=false,
163 multiport/.code=\pgfkeys{
164   /xbondgraphs/element/multiport boolean=true,
165 },
166 label/.style={
167   \xbondgraphs@bgelementlabelcolor,
168 },
169 % The 'XBG' keys are used as package options
170 /XBG/.cd,
171 barbangle/.store in=\xbondgraphs@barbangle,
172 barbangle=40,
173 singlebondwidth/.store in=\xbondgraphs@singlebondwidth,
174 singlebondwidth=1pt,
175 multibondwidth/.store in=\xbondgraphs@multibondwidth,
176 multibondwidth=4pt,
177 bgelementlabelcolor/.store in=\xbondgraphs@bgelementlabelcolor,
178 bgelementlabelcolor=blue,
179 bondlabelcolor/.store in=\xbondgraphs@bondlabelcolor,
180 bondlabelcolor=green!50!black,
181 gray/.code={
182   \pgfkeys{
183     /XBG/.cd, bondlabelcolor=gray, bgelementlabelcolor=gray
184   }
185   \colorlet{diff}{white!60!black}
186   \colorlet{error}{white!30!black}
187 },
188 barbdirection/.is choice,
189 barbdirection/leftbelow/.code={\pgfkeys{/xbondgraphs/bond/barb direction flip angle=45}},
190 barbdirection/alwaysbelow/.code={\pgfkeys{/xbondgraphs/bond/barb direction flip angle=-1}},
191 barbdirection/alwaysbelow,
192 /tikz/.cd,
193 bond/.style={
194   /xbondgraphs/bond,
195   #1,
196   /tikz,
197   draw = none,
198   decoration={bond},

```

```

199     postaction=decorate,
200 },
201 bond graph element/.code 2 args={
202     \pgfkeys{
203         /xbondgraphs/element,
204         #2
205     }
206     \tikzset{
207         shape=rounded rectangle,
208         inner sep = 1.5pt,
209         node contents = {%
210             \ifxbondgraphs@element@multiport%
211             \ifnum\xbondgraphs@element@n=1
212             \ensuremath{\mathbb{b}{#1}}}%
213             \else
214             \ensuremath{\mathbb{b}{#1}_{\xbondgraphs@element@n}}%
215             \fi
216             \else%
217             \ifnum\xbondgraphs@element@n=1
218             \ensuremath{\mathbf{b}{#1}}}%
219             \else
220             \ensuremath{\mathbf{b}{#1}_{\xbondgraphs@element@n}}%
221             \fi
222             \fi%
223         },
224         prefix after command={
225             \pgfextra{
226                 \tikzset{
227                     every pin/.style={
228                         /xbondgraphs/element/label,
229                         pin distance = 2pt,
230                         pin edge={
231                             draw = none,
232                             decoration={
233                                 markings,
234                                 mark = at position 0.5 with {
235                                     \node[rotate=\pgfdecoratedangle,inner sep = 0pt,/xbondgraphs/element/label]
236                                     },
237                                 },
238                                 decorate,
239                             },
240                         },
241                     every label/.style={
242                         /xbondgraphs/element/label,
243                     },
244                 },
245             }
246         },
247     }
248     \ifxbondgraphs@element@word

```

```

249 \tikzset{draw,line width = 0.75\xbondgraphs@singlebondwidth,shape=ellipse}
250 \fi
251 },
252 bond label/.style={
253     font=\small,
254     /xbondgraphs/bond/label,
255     sloped,
256 },
257 effort/.style={
258     edge node={node [bond label,above]{#1}}
259 },
260 flow/.style={
261     edge node={node [bond label,below]{#1}}
262 },
263 causal stroke style/.style={
264     width=\xbondgraphs@causalitystrokescale*\xbondgraphs@multibondwidth,
265 },
266 }
267
268 % MUX SHAPE
269 \pgfkeys{
270 /tikz/mux/.code={
271     \pgfkeys{
272         % /tikz/shape=mux,
273         /mux/.cd,
274         #1
275     }
276     \tikzset{
277         outer sep = 0pt,
278         inner sep = 0pt,
279         minimum width = \pgfkeysvalueof{/mux/width},
280         node contents = {},
281         fill=black,
282         shape=mux,
283     }
284 },
285 /mux/.is family,
286 mux,
287 inputs/.initial=2,
288 outputs/.initial=2,
289 io spacing/.initial=5mm,
290 width/.initial=3pt,
291 }
292 \pgfdeclareshape{mux}{
293     \savedanchor\centerpoint{%
294         \pgf@x=0%
295         \pgf@y=0%
296     }%
297     \inheritssavedanchors[from=rectangle]
298     \inheritanchorborder[from=rectangle]

```

```

299 \inheritanchor[from=rectangle]{north}
300 \inheritanchor[from=rectangle]{north west}
301 \inheritanchor[from=rectangle]{north east}
302 \inheritanchor[from=rectangle]{center}
303 \inheritanchor[from=rectangle]{west}
304 \inheritanchor[from=rectangle]{east}
305 \inheritanchor[from=rectangle]{mid}
306 \inheritanchor[from=rectangle]{mid west}
307 \inheritanchor[from=rectangle]{mid east}
308 \inheritanchor[from=rectangle]{south}
309 \inheritanchor[from=rectangle]{south west}
310 \inheritanchor[from=rectangle]{south east}
311 \savedmacro\inputs{\pgfmathtruncatemacro\inputs{\pgfkeysvalueof{/mux/inputs}}}%
312 \savedmacro\outputs{\pgfmathtruncatemacro\outputs{\pgfkeysvalueof{/mux/outputs}}}%
313 \savedmacro\numio{\pgfmathparse{max(\inputs,\outputs)}\pgfmathtruncatemacro\numio\pgfmathresult}
314 \saveddimen\height{%
315   \pgfmathparse{max(\pgfkeysvalueof{/mux/inputs},\pgfkeysvalueof{/mux/outputs})}
316   \pgfmathparse{(\pgfmathresult) * \pgfkeysvalueof{/mux/io spacing}}
317   \pgfmathsetlength\pgf@x{\pgfmathresult}
318 }
319 \saveddimen\halfwidth{\pgfmathsetlength\pgf@x{\pgfkeysvalueof{/mux/width}/2}\pgfmathresult}
320 \saveddimen\iospacing{\pgfmathsetlength\pgf@x{\pgfkeysvalueof{/mux/io spacing}}\pgfmathresult}
321 \backgroundpath{
322   \pgfpathrectanglecorners{
323     \pgfpointadd{\centerpoint}{\pgfpoint{-\halfwidth}{\height/2}}
324   }{
325     \pgfpointadd{\centerpoint}{\pgfpoint{\halfwidth}{-\height/2}}
326   }
327 }
328 \pgfutil@g@addto@macro\pgf@sh@s@mux{%
329   % Start with the maximum input number and go backwards.
330   % If the anchor is undefined, create it. Otherwise stop.
331   \c@pgf@counta=\pgfkeysvalueof{/mux/inputs}\relax%
332   \pgfmathloop%
333   \ifnum\c@pgf@counta>0\relax%
334     \pgfutil@ifundefined{pgf@anchor@mux@input\the\c@pgf@counta}{%
335       \expandafter\xdef\csname pgf@anchor@mux@input\the\c@pgf@counta\endcsname{%
336         \noexpand\ioanchor{\the\c@pgf@counta}{-1}{((\inputs-\numio)/2+0.5)}%
337       }%
338     }\c@pgf@counta=0\relax}%
339     \advance\c@pgf@counta-1\relax%
340     \repeatpgfmathloop%
341 }%
342 \pgfutil@g@addto@macro\pgf@sh@s@mux{%
343   % Start with the maximum output number and go backwards.
344   % If the anchor is undefined, create it. Otherwise stop.
345   \c@pgf@counta=\pgfkeysvalueof{/mux/outputs}\relax%
346   \pgfmathloop%
347   \ifnum\c@pgf@counta>0\relax%
348     \pgfutil@ifundefined{pgf@anchor@mux@output\the\c@pgf@counta}{%

```

```

349     \expandafter\xdef\csname pgf@anchor@mux@output\the\c@pgf@counta\endcsname{%
350         \noexpand\ioanchor{\the\c@pgf@counta}{1}{((\outputs-\numio)/2+0.5)}%
351     }%
352     }{\c@pgf@counta=0\relax}%
353     \advance\c@pgf@counta-1\relax%
354     \repeatpgfmathloop%
355 }%
356 }
357
358 \def\ioanchor#1#2#3{%
359     \pgfpointadd{\centerpoint}{\pgfpoint{#2*\halfwidth}{\height/2-#1*\iospacing+#3*\iospacing}}%
360 }
361
362 \colorlet{diff}{orange}
363 \colorlet{error}{red}
364
365 % Proces all /XBG keys as package options
366 \ProcessPgfPackageOptions{/XBG}
367 \</package>

```

7 Change History

v0.0.1

General: Initial version 1

8 Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

A		336, 338, 339,	H
\advance	339, 353	345, 347, 348,	\halfwidth
		349, 350, 352, 353	.. 319, 323, 325, 359
B		\centerpoint	\height 314, 323, 325, 359
\ba . 15, 18, 20, 21, 23,		.. 293, 323, 325, 359	\hullpointx 21, 27, 49, 55
42, 45, 47, 48, 49, 51		\colorlet	\hullpointy 22, 27, 50, 55
\backgroundpath . . .	321	.. 185, 186, 362, 363	
\backx	18, 21,	\csname	I
26, 33, 48, 49, 54, 61		335, 349	\ifnum . 211, 217, 333, 347
\backy .. 20, 33, 47, 48, 61			\ifxbondgraphs@bond@causality@eout
\barbdirection	4	E	.. 84, 98
		\endcsname	\ifxbondgraphs@bond@causality@fout
		335, 349	.. 87, 99
C		\ensuremath	\ifxbondgraphs@element@multiport
\c@pgf@counta ..	331,	.. 212, 214, 218, 220	.. 103, 210
333, 334, 335,		\expandafter . . .	
		335, 349	

<code>\ifxbondgraphs@element@word</code>	<code>\pgfkeys</code> .. 76, 78, 80,	333, 338, 339,
..... 102, 248	82, 106, 124, 130,	345, 347, 352, 353
<code>\inheritanchor</code>	136, 137, 138,	<code>\repeatpgfmathloop</code>
..... 299, 300,	163, 182, 189, 340, 354
301, 302, 303,	190, 202, 269, 271	<code>\RequirePackage</code> .. 5, 6, 7
304, 305, 306,	<code>\pgfkeysvalueof</code> ...	
307, 308, 309, 310 279, 311,	S
<code>\inheritanchorborder</code> 298	312, 315, 316,	<code>\savedanchor</code> 293
<code>\inheritsavedanchors</code> 297	319, 320, 331, 345	<code>\saveddimen</code> 314, 319, 320
<code>\inputs</code> 311, 313, 336	<code>\pgflinewidth</code> 13, 41	<code>\savedmacro</code> 311, 312, 313
<code>\ioanchor</code> .. 336, 350, 358	<code>\pgfmathloop</code> ... 332, 346	<code>\sbw</code> . 13, 16, 18, 19, 20,
<code>\iospacing</code> 320, 359	<code>\pgfmathparse</code>	21, 23, 24, 40, 44,
	... 74, 313, 315, 316	45, 46, 47, 49, 51, 62
M	<code>\pgfmathresult</code>	<code>\SingleBondBarb</code> .. 10
<code>\mathbb</code> 212, 214 75, 313,	<code>\small</code> 253
<code>\mathbf</code> 218, 220	316, 317, 319, 320	<code>\startx</code> 43, 59
<code>\mbw</code> . 14, 18, 20, 22, 41,	<code>\pgfmathsetlength</code> .	<code>\starty</code> 44, 59
44, 45, 46, 47, 50, 52 317, 319, 320	<code>\state</code> 69, 72
<code>\MultiBondBarb</code> ... 37	<code>\pgfmathsetlengthmacro</code>	T
 13, 14, 15,	<code>\the</code> 334, 335,
N	16, 17, 18, 20, 21,	336, 348, 349, 350
<code>\NeedsTeXFormat</code> 2	22, 23, 24, 40, 41,	<code>\tikzset</code> 85, 88,
<code>\newif</code> ... 98, 99, 102, 103	42, 43, 44, 45, 46,	90, 206, 226, 249, 276
<code>\node</code> 235	47, 48, 49, 50, 51, 52	<code>\tipendx</code>
<code>\noexpand</code> 336, 350	<code>\pgfmathtruncatemacro</code>	23, 25, 28, 51, 53, 56
<code>\numio</code> 313, 336, 350 311, 312, 313	<code>\tipendy</code> ... 24, 28, 52, 56
	<code>\pgfpathlineto</code>	<code>\tipx</code> . 16, 23, 32, 45, 51, 60
O	32, 33, 60, 61, 70, 93	<code>\tipy</code> ... 17, 32, 46, 48, 60
<code>\outputs</code> ... 312, 313, 350	<code>\pgfpathmoveto</code> ... 31, 59	
	<code>\pgfpathrectanglecorners</code>	U
P 322	<code>\usetikzlibrary</code> 9
<code>\path</code> 93	<code>\pgfpoint</code> . 32, 33, 59,	X
<code>\pgf@sh@smux</code> .. 328, 342	60, 61, 323, 325, 359	<code>\xbondgraphs@barbangle</code>
<code>\pgf@x</code> . 294, 317, 319, 320	<code>\pgfpointadd</code> 323, 325, 359	<code>\xbondgraphs@bgelementlabelcolor</code>
<code>\pgf@y</code> 295	<code>\pgfpointdecoratedinputsegmentlast</code> 15, 42, 171 167, 177
<code>\pgffarrowshullpoint</code> 70, 93	<code>\xbondgraphs@bond@barbarrowhead</code>
..... 27, 28, 55, 56	<code>\pgfpointorigin</code> 31 85, 88, 90, 148
<code>\pgffarrowssetbackend</code>	<code>\pgfsetlinewidth</code> ... 62	<code>\xbondgraphs@bond@barbdirection</code>
..... 26, 54	<code>\pgfusepathqstroke</code> 85, 88, 90, 118
<code>\pgffarrowssettipend</code> 34, 63	<code>\xbondgraphs@bond@barbdirectionflipangle</code>
..... 25, 53	<code>\pgfutil@g@addto@macro</code> 74, 119
<code>\pgfdeclarearrow</code> . 10, 37 328, 342	<code>\xbondgraphs@bondlabelcolor</code>
<code>\pgfdeclaredecoration</code>	<code>\pgfutil@ifundefined</code> 151, 179
..... 68 334, 348	<code>\xbondgraphs@causalitystrokescale</code>
<code>\pgfdeclareshape</code> .. 292	<code>\ProcessPgfpPackageOptions</code> 146, 264
<code>\pgfdecoratedangle</code> 366	<code>\xbondgraphs@element@n</code>
..... 74, 235	<code>\ProvidesPackage</code> 3 157,
<code>\pgfdecoratedinputsegmentlength</code>	R	211, 214, 217, 220
..... 69	<code>\relax</code> 331,	
<code>\pgfextra</code> 93, 225		

<code>\xbondgraphs@multibondwidth</code>	<code>\xbondgraphs@singlebondwidth</code>	<code>\xdef</code>	335, 349
14, 142, 144, 175, 264	40, 116, 142, 173, 249		