pinouTikz

v1.1.2

User's manual

Robert Blazek
robert.h.blazek@gmail.com

Abstract

This package which requires ε-TEX, provides macros for creating pinout diagrams of chips.

Contents

1 Introduction

1.1 Description

This package defines macros for generating symbolic pinout diagrams for different package classes, such as DIP, PLCC, etc.

1.2 Motivation

Whoever has ever had to do with FPGA or MCUs (whether for living or leisure), it's just natural he or she might have been in a need to document some pins. So was my case and since I failed in finding any package in LaTeX to suit my needs, I opted for creating one myself.
I hope others will find it as useful as it was to me and my colleagues.
This is my first latex package documentation ever - and since I hate reinventing the wheel - this manual has been based upon that of **xstrings** - with the courtesy of the author, of course.

2 The macros

For a better understanding, let's see first the macros with the simpler arguments possible. No special catcode, no exotic token, no control sequence either: only alphanumeric chars will be contained in the arguments.

In the following chapters, all the macros will be presented this way:

- a short description of the operation;
- the operation under special conditions. For each conditions considered, the operation described has priority on that (those) below;
- finally, several examples are given. I tried to find them most easily comprehensible and most representative of the situations met in normal use.

Important: in the following, a ⟨*number*⟩ can be an integer written with numeric chars, a counter, or the result of an arithmetic operation made with the command \numexpr.

All the macros of pinouTikz are displayed in blue.

2.1 The pinout diagrams

2.1.1 \PDIP

\PDIP(⟨*pincount*⟩){⟨*pinarray*⟩}

Draws a PDIP package with generic number of pins, as a standalone glyph.
\pctPDIP(⟨*pincount*⟩){⟨*pinarray*⟩}

Draws a PDIP package with generic number of pins, as a picture sub-element.

- ⟨*pincount*⟩ the number of pins of a DIP package and should be an even number.
- {⟨*pinarray*⟩} is a comma-separated list of pins - each pin definition is as follows: ⟨*pinnumber*⟩/{⟨*pinlabel*⟩}.

```
\begin{figure}
  \centering
  \PDIP(4){%
    1/{E},2/B,3/NC,4/C}
  \caption{NPN-Transistor, 4-pin PDIP package, as a glyph} \label{fig:X_DIP4}
\end{figure}
```

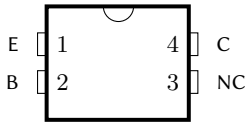

Figure 1: NPN-Transistor, 4-pin PDIP package, as a glyph

```
\begin{figure}[ht!]
  \centering
  \begin{tikzpicture}
  % an arbitrary picture laid over the PDIP diagram
  \draw[thick,rounded corners=8pt](0,0) (0,2) (1,3.25) (2,2) (2,0) (0,2) (2,2) (0,0);
  % a PDIP diagram itself
  \pctPDIP(4){%
    1/{E},
    2/B,
    3/NC,
    4/C
    }%
  % a PDIP diagram itself
  \begin{scope}[shift={(3.5,0.5)}]
  \pctPDIP(4){%
    1/{E},
    2/B,
    3/NC,
    4/C
    }%
  \end{scope}
  \end{tikzpicture}
  \caption{NPN-Transistor, 4-pin PDIP package, placed within another picture, twice} \label{fig:X_DIP8}
\end{figure}
```

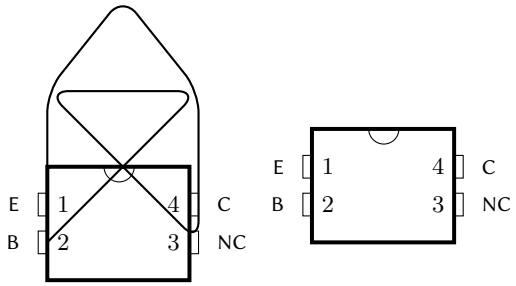

Figure 2: NPN-Transistor, 4-pin PDIP package, placed within another picture, twice

```
\begin{figure}
  \centering
  \PDIP(8){%
    1/CLK,
    2/A,
    3/B,
    4/GND,
    5/Y,
    6/{\FormatPinLabel{~Y~}/RESET},
    7/NC,
    8/$V_{cc}$%
```

```
    }
  \caption{TTL logic chip, 8-pin PDIP package, as a glyph} \label{fig:X_DIP8}
\end{figure}
```

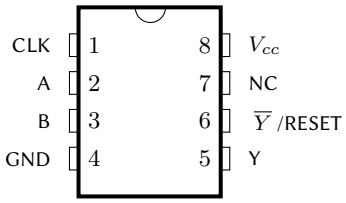

Figure 3: TTL logic chip, 8-pin PDIP package, as a glyph

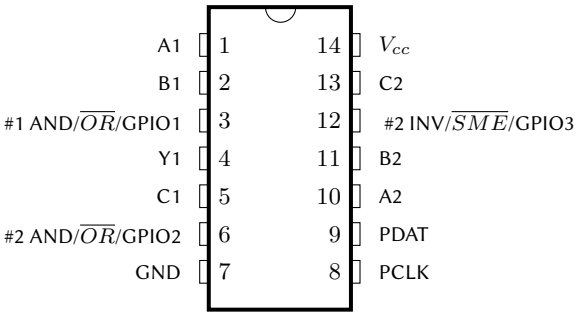

Figure 4: Generic programmable TTL logic chip, 14-pin PDIP package

```
\begin{figure}[ht!]
  \centering
  \PDIP(14){%
    1/A1,
    2/B1,
    3/\FormatPinLabel{\#1 AND/~OR~/GPIO1},
    4/Y1,
    5/C1,
    6/\FormatPinLabel{\#2 AND/~OR~/GPIO2},
    7/GND,
    8/PCLK,
    9/PDAT,
    10/A2,
    11/B2,
    12/\FormatPinLabel{\#2 INV/~SME~/GPIO3},
    13/C2,
    14/$V_{cc}$}
  \caption{Generic programmable TTL logic chip, 14-pin PDIP
      package} \label{fig:X_DIP14}
\end{figure}
```

2.1.2 `\TQFP`

`\TQFP(`⟨*pinnumber*⟩`){`⟨*pinarray*⟩`}`

Draws a TQFP package with generic number of pins, as a standalone glyph.

`\pctTQFP(`⟨*pinnumber*⟩`){`⟨*pinarray*⟩`}`

Draws a TQFP package with generic number of pins, as a picture sub-element.

- ⟨*pincount*⟩ the number of pins of a DIP package and should be an even number.
- {⟨*pinarray*⟩} is a comma-separated list of pins - each pin definition is as follows: ⟨*pinnumber*⟩/{⟨*pinlabel*⟩}.

```
\begin{figure}[ht!]
  \centering
  \TQFP(32){%
    1/{PD.0/RTX1},
    2/{PA.0/STX1},
    3/PA.1,
    4/PA.2,
    5/PA.3,
    6/PA.4,
    7/PA.5,
    8/GND,
    9/PA.6,
    10/PA.7,
    11/{PB.0/RTX1},
    12/{PB.1/STX1},
    13/PB.2,
    14/PB.3,
    15/PB.4,
    16/PB.5,
    17/PB.6,
    18/PB.7,
    19/\FormatPinLabel{PC.0/~ALE~/PLPBC0},
    20/PC.1,
    21/PC.2,
    22/PC.3,
    23/PC.4,
    24/PC.5,
    25/PC.6,
    26/PC.7,
    27/XTAL1/PD.3,
    28/XTAL2/PD.4,
    29/RST,
    30/PD.1,
    31/PD.2,
    32/$V_{cc}$}
  \caption{A generic MCU chip, 32-pin TQFP package} \label{fig:X_
      TQFP32}
\end{figure}
```

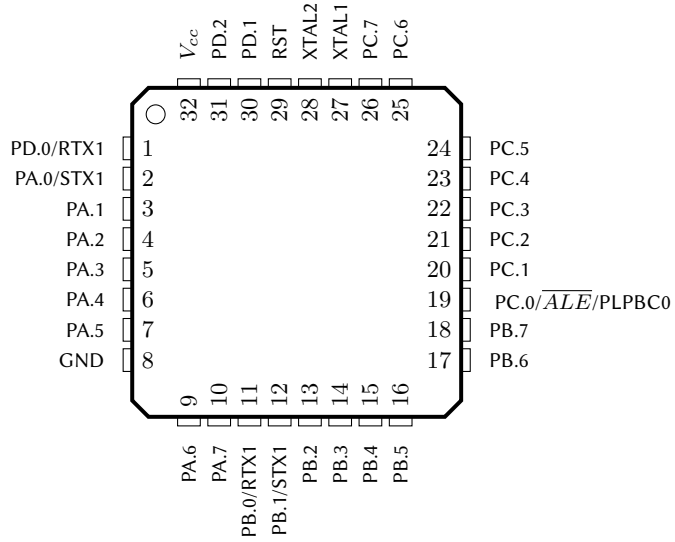

Figure 5: A generic MCU chip, 32-pin TQFP package

2.1.3 \PLCC

\PLCC(⟨*pinnumber*⟩){⟨*pinarray*⟩}

Draws a PLCC package with generic number of pins.
\pctPLCC(⟨*pinnumber*⟩){⟨*pinarray*⟩}
Draws a PLCC package with generic number of pins, as a picture sub-element.

- ⟨*pincount*⟩ the number of pins of a DIP package and should be an even number.
- {⟨*pinarray*⟩} is a comma-separated list of pins - each pin definition is as follows: ⟨*pinnumber*⟩/{⟨*pinlabel*⟩}.

```
\begin{figure}[ht!]
  \centering
  \PLCC(28){%
    1/{PD.0/RTX1},
    2/{PA.0/STX1},
    3/PA.1,
    4/PA.2,
    5/PA.3,
    6/PA.4,
    7/PA.5,
    8/GND,
    9/PA.6,
    10/PA.7,
    11/{PB.0/RTX1},
    12/{PB.1/STX1},
    13/PB.2,
    14/PB.3,
    15/PB.4,
    16/PB.5,
    17/PB.6,
    18/PB.7,
    19/\FormatPinLabel{PC.0/~ALE~/PLPBC0},
    20/PC.1,
    21/PC.2,
    22/PC.3,
    23/{XTAL1/PD.3},
    24/{XTAL2/PD.4},
    25/RST,
    26/PD.1,
    27/PD.2,
    28/$V_{cc}$}
  \caption{A generic MCU chip, 28-pin PLCC package} \label{fig:X_
      PLCC28}
\end{figure}
```

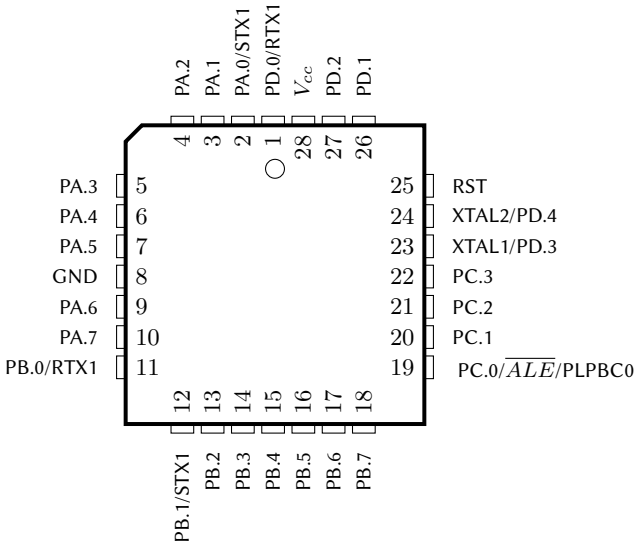

Figure 6: A generic MCU chip, 28-pin PLCC package

⋆
⋆ ⋆

That's all, I hope you will find this package useful!
Please, send me an email if you find a bug or if you have any idea of improvement...

Robert Blazek