

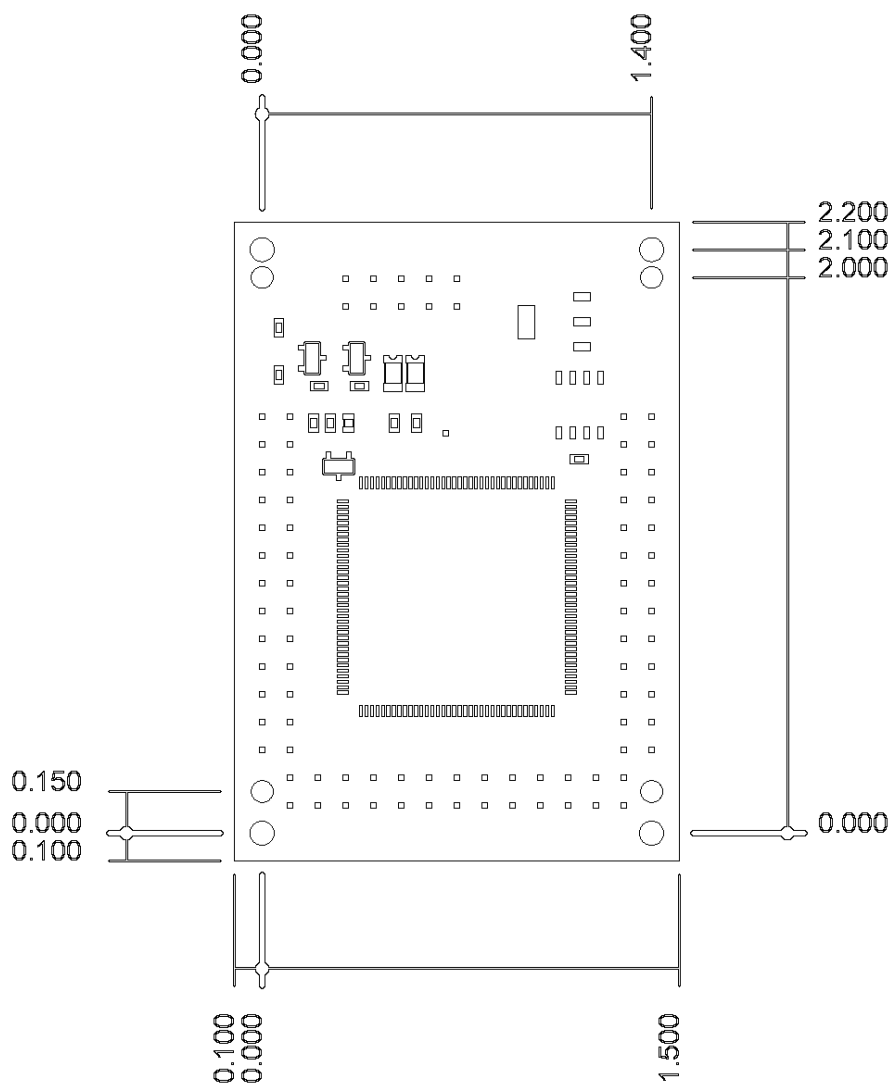
# FPGA RS232 development boards

© 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011 KNJN LLC

<http://www.knjin.com/>

This document applies to the following boards:

- Pluto rev. F
- Pluto-II rev. E
- Pluto-3 rev. B



# Table of Contents

1 Welcome.....	5
1.1 This guide.....	5
1.2 The Pluto boards.....	5
1.3 Boards characteristics.....	5
2 Software tools.....	6
2.1 Important downloads.....	6
2.2 FPGA software.....	6
2.3 C/C++ compiler.....	6
3 FPGAconf.....	7
3.1 Board selection.....	7
3.2 Configuring the FPGA.....	7
3.3 FPGAconf options.....	7
4 FPGA boot-PROM.....	8
4.1 What's the boot-PROM?.....	8
4.2 Boot-PROM actions.....	8
4.3 Boot-PROM requirements.....	8
4.4 Boot-PROM and JTAG.....	8
5 FPGAconf extras.....	9
5.1 Auto configuration mode.....	9
5.2 Scrollbar.....	9
5.3 Terminal.....	9
6 FPGA configuration using JTAG.....	10
6.1 JTAG requirements.....	10
6.2 JTAG configuration.....	10
7 FPGA projects.....	11
7.1 Generate your own FPGA files.....	11
7.2 A simple project.....	11
7.3 The serial interface.....	11
8 FPGA connections.....	12
8.1 FPGA pins.....	12
8.2 IO headers.....	12
8.3 Boot-PROM connection (Pluto-II and Pluto-3 only).....	12
9 Connectors and headers.....	13
9.1 Power header.....	13
9.2 TXDI connector.....	13
9.3 Secondary connector.....	13
10 JTAG connection.....	14
10.1 JTAG on Pluto.....	14
10.2 JTAG on Pluto-II.....	14
10.3 JTAG on Pluto-3.....	14
11 Flashy boards.....	15
11.1 Flashy daughter-boards.....	15
11.2 FlashyMini design.....	15
11.3 FlashyDemo design.....	15
12 FPGA configuration through RS232.....	16
12.1 Pluto FPGA configuration.....	16
With a UART.....	16
Without a UART.....	16
12.2 Pluto-II FPGA configuration.....	16
12.3 Pluto-3 FPGA configuration.....	16
13 Quartus-II JTAG indirect mode.....	17
13.1 What is it?.....	17
13.2 Create a "JTAG indirect configuration" file.....	17
13.3 Program the boot-PROM.....	17
14 Power requirements.....	18
14.1 Wall adapter.....	18
14.2 USB to power jack cable.....	18
14.3 Power consumption.....	18
14.4 Voltage regulator temperature.....	18
15 Connecting the Pluto boards to a PC.....	19

---

15.1 Serial connection.....	19
15.2 With a TXDI.....	19
15.3 With a TXDI/MAX232 or TXDI/FTDI.....	19
15.4 Without a TXDI.....	19
16 Sample C code for RS-232 Win32 send & receive.....	20
17 Board checklist.....	21
17.1 The FPGA doesn't configure?.....	21
17.2 Boot-PROM problem?.....	21
18 Board connectors and headers, with IO pin assignments.....	22
18.1 Pluto.....	22
18.2 Pluto-II.....	23
18.3 Pluto-3.....	24
19 Mechanical drawings.....	25
19.1 Pluto.....	25
19.2 Pluto-II.....	26
19.3 Pluto-3.....	27



# 1 Welcome

## 1.1 This guide

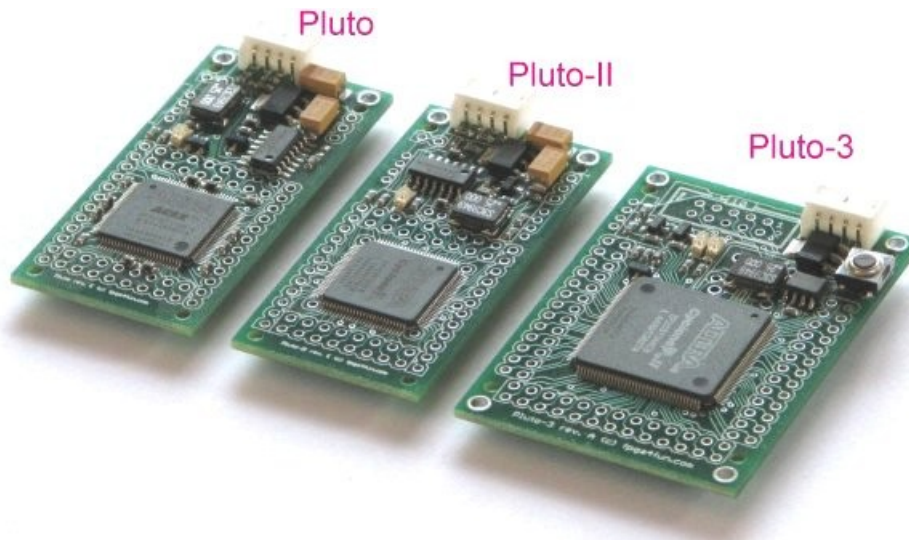
Welcome to the KNJN FPGA RS232 development boards guide.

Although FPGA boards can be intimidating, the Pluto FPGA boards are easy to use. This document is partitioned in short and easy to read chapters, and will explain all you need to know about your new FPGA board.

## 1.2 The Pluto boards

This guide applies to three different boards (Pluto, Pluto-II and Pluto-3).

Along this document, when a paragraph apply to all three of them, they are collectively named “the Pluto boards” or “the FPGA boards”.



*The Pluto boards*

Check them also on the [KNJN website](#).

## 1.3 Boards characteristics

	Pluto	Pluto-II	Pluto-3
FPGA (vendor page)	<a href="#">EP1K10</a>	<a href="#">EP1C3</a>	<a href="#">EP2C5</a>
Datasheet (PDF)	<a href="#">ACEX 1K</a>	<a href="#">Cyclone</a>	<a href="#">Cyclone II</a>
Logic cells	576	2910	4608
IO pins	41	51	65
PLL	-	Yes	Yes
External clocks	up to 2	up to 4	up to 4
Boot-PROM	-	1 Mbits	4 Mbits
On-board oscillator	25MHz	25MHz	25MHz
DIL8 oscillator header	-	-	Yes
Push-button	-	-	Yes
JTAG header	-	(see chapter 10)	Yes
LED(s)	1	1	2
Flashy ready	Flashy	Flashy	FlashyD
Dimensions	58x28mm	58x28mm	58x41mm

---

## 2 Software tools

### 2.1 Important downloads

Each KNJN FPGA board is provided with a “startup kit” that includes the board documentation and other files (mainly example source code). The startup kit doesn't include some important software tools that are required as you experiment with your FPGA board:

- The FPGA software
- A C/C++ compiler

### 2.2 FPGA software

The FPGA software is required to generate FPGA bitfiles. You have to get the software that matches your FPGA.

- For Pluto, get [Quartus II Web Edition 9.0 Service Pack 2](#)
- For Pluto-II, get [Quartus II Web Edition 11.0 Service Pack 1](#)
- For Pluto-3, get the latest [Quartus II Web Edition](#)

These software are free, don't require a license and don't expire. They have lots of features but are big so you might want to download and install them in advance.

### 2.3 C/C++ compiler

A C/C++ compiler is optional but you'll need one for many projects. Here are different compilers that can be used:

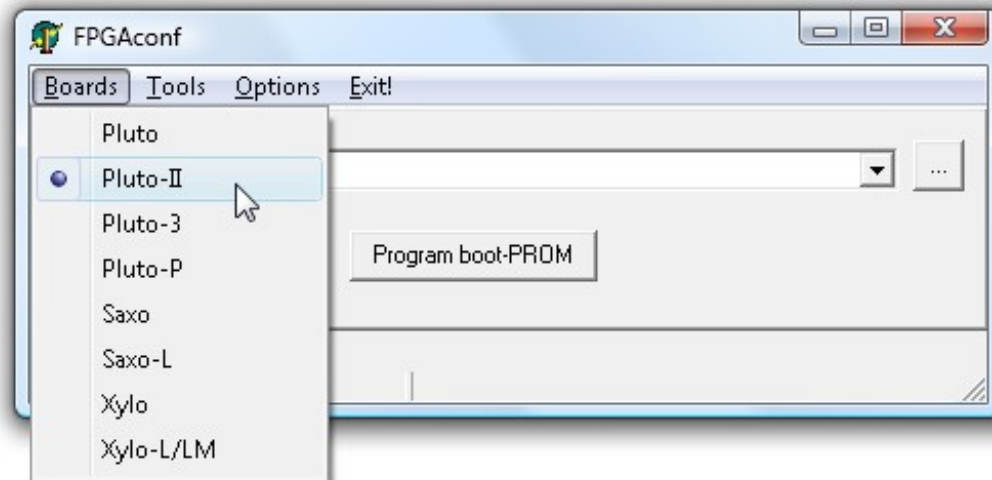
- Microsoft Visual C++ 5.0 or 6.0
- Microsoft [Visual C++ express edition](#) (free download)
- [Digital Mars](#) (free download)
- Jacob Navia's [lcc-win32](#) (free download)

## 3 FPGAconf

FPGAconf is a multi-function software provided with your KNJN FPGA board.

### 3.1 Board selection

FPGAconf can be used with different boards - make sure that your board is selected. For example, we select Pluto-II below.



### 3.2 Configuring the FPGA

FPGAconf makes FPGA configuration very easy.

1. Select an RBF bitfile (I.e. click on the browse button to select a file - the browse button is shown as "...").
2. Click "Configure".  
After a few seconds, the FPGA should be configured. If not, see the board checklist (chapter 17).

For your convenience, sample RBF files are provided in the startup-kit. In particular, try "ledblink.rbf" and "ledglow.rbf". For more information on RBF files, check paragraph 7.1.

When the FPGA is not configured, the board's LED glows slightly. With practice, you'll be able to recognize immediately if the FPGA is configured or not.

### 3.3 FPGAconf options

The different settings available are:

1. "Beep after configuration"
2. "COM port" (choose from COM1 to COM32)
3. "Look for any COM port available" (if the COM port specified fails, FPGAconf uses the next port it can open)
4. "Use alternate COM port for the terminal" (useful if you want the terminal window to use a different port)
5. "Keep COM port open after configuration" (some PCs reset the Pluto board unless this is enabled)
6. "Turbo mode" (allows faster FPGA configuration, works on most boards)
7. "Send trimmed bit-file" (send a shortened bit-file, required for some applications)

Pluto-II has an extra option:

8. Pluto-II clock speed: FPGAconf needs to know the on-board clock speed (for the boot-PROM programming). This should be set to 25MHz (some older Pluto-II revisions use 32MHz).

## 4 FPGA boot-PROM

This feature applies to Pluto-II and Pluto-3 only.

### 4.1 What's the boot-PROM?

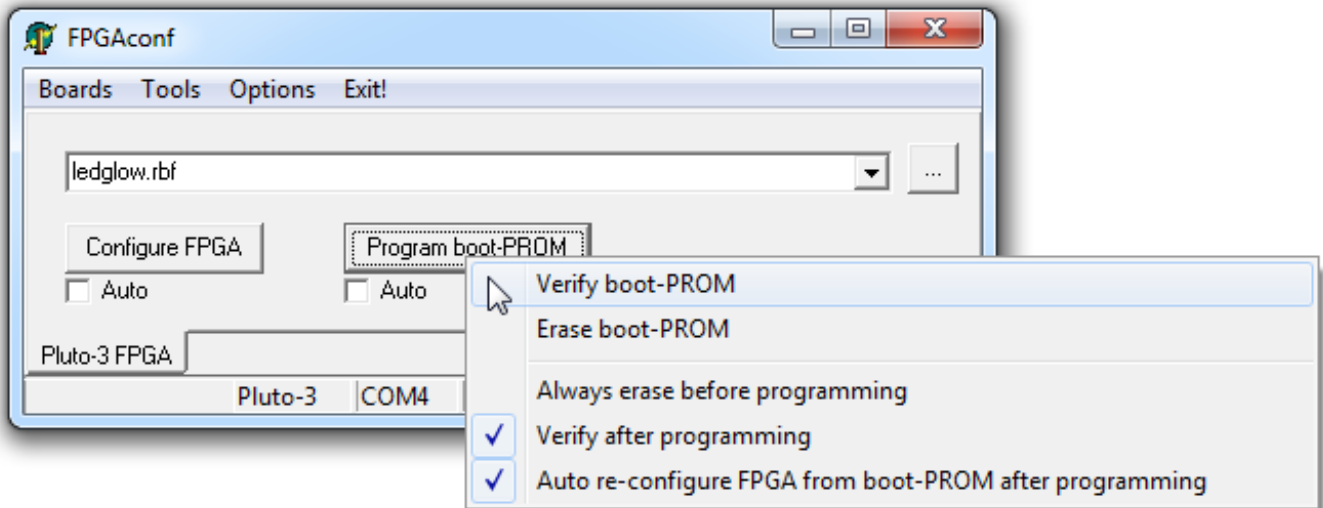
Pluto-II and Pluto-3 have an FPGA boot-PROM.

- At power-up, the boot-PROM tries to configure the FPGA.
- If the boot-PROM is empty or its content is invalid, the FPGA stays un-configured. The FPGA can still be configured manually using the PC.

### 4.2 Boot-PROM actions

The boot-PROM can be programmed, verified and erased.

- To program the boot-PROM, select an RBF file and click on the “Program boot-PROM” button.
- To verify or erase the boot-PROM, right-click on the button and use the drop-down menu.



### 4.3 Boot-PROM requirements

FPGAconf requires bi-directional communication with the PC to access the boot-PROM.

If the boot-PROM is not recognized by FPGAconf:

- Make sure bidirectional communication from the PC works correctly (try the SerialRxTx project to verify bidirectional communication - paragraph 7.3).
- Make sure the menu → Options → Pluto-II clock speed is set correctly (Pluto-II only).

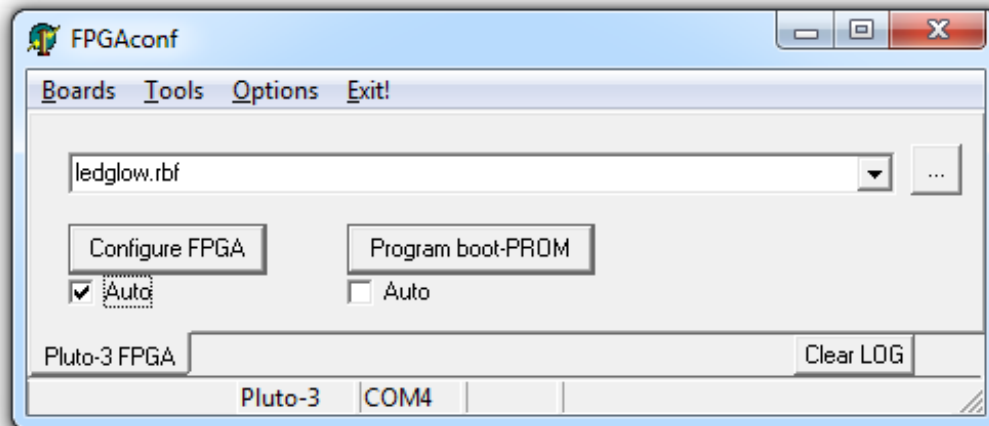
### 4.4 Boot-PROM and JTAG

The boot-PROM can also be programmed from JTAG, although it is usually less convenient than with FPGAconf. For details, check chapter 13.

## 5 FPGAconf extras

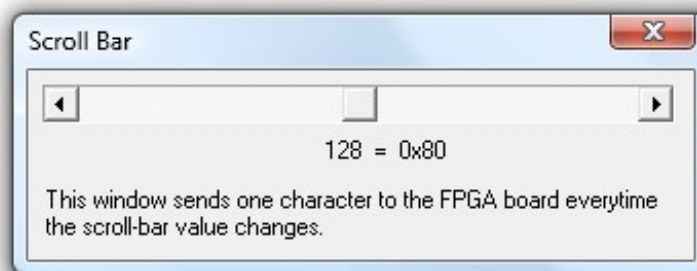
### 5.1 Auto configuration mode

When the “Auto mode” is enabled, FPGAconf monitors the RBF file and takes action each time the file is updated. Useful for example if you want to re-configure the FPGA automatically after each Quartus-II compilation,.



### 5.2 Scrollbar

FPGAconf has a “scrollbar window” that is activated by pressing CTRL-S. Every time the scrollbar position is changed, a byte between 0 and 255 is sent to the Pluto board (depending of the bar position).



That can be used to control easily a servomotor for example, or other simple applications that can be controlled by a single byte.

### 5.3 Terminal

FPGAconf has a RS-232 terminal window that is activated by pressing CTRL-T.

Note: you can also use a thrid-party terminal, but it is recommended in this case to use a TXDI with MAX232. See paragraph 15.3 and KNJN's [TXDI](#) page for more information.

## 6 FPGA configuration using JTAG

This feature applies to Pluto-II and Pluto-3 only.

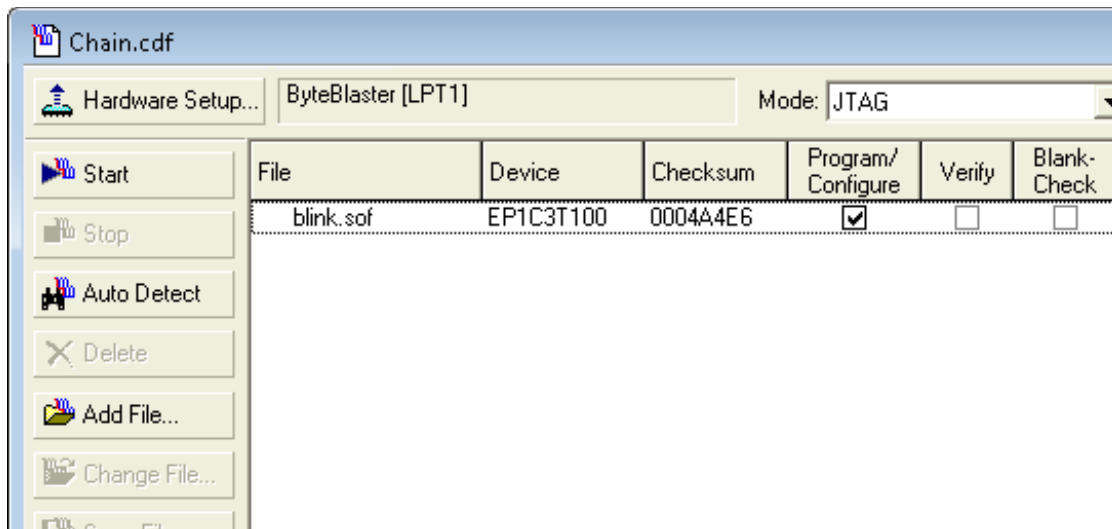
### 6.1 JTAG requirements

To use the JTAG port, you need a compatible JTAG cable (like an Altera ByteBlaster-MV/II or USB-Blaster) connected to your board's JTAG connector (chapter 10).

### 6.2 JTAG configuration

Follow these steps:

- In Quartus-II, open the “Programmer” window (Menu → Tools → Programmer)
- Click on the “Hardware Setup” button and select the JTAG cable you’re using.
- Select “JTAG” in the “Mode” drop-down list.
- Load the “.sof” file.
- Check the “Program/Configure” check-box and click “Start”.



Note how JTAG configuration is accomplished through the Quartus-II software using SOF files instead of RBF files. Both files are generated by Quartus-II – more details in paragraph 7.1.

The boot-PROM can also be programmed using JTAG, check chapter 13 for more information.

## 7 FPGA projects

### 7.1 Generate your own FPGA files

The Pluto FPGA boards use SOF or RBF files generated by Altera's Quartus-II software.

1. Create a Quartus-II project, and go to "Assignments/Device"
2. Choose the device:
  - a. For Pluto, choose family "APEX1K" and device "EP1K10TC100-3".
  - b. For Pluto-II, choose family "Cyclone" and device "EP1C3T100C8".
  - c. For Pluto-3, choose family "Cyclone-II" and device "EP2C5T144C8".
3. Click on "Device & Pin Options..."
  - a. Go to the "Programming Files" tab, select "Raw Binary File (.rbf)".
  - b. Go to "Unused Pins", select "As inputs, tri-stated".
  - c. Click "OK".
4. Click "OK".

Option 3.a makes sure RBF files are generated (used for RS232 FPGA configuration). Otherwise only SOF files are generated (used for JTAG FPGA configuration).

Option 3.b is optional but highly recommended. It prevents the FPGA from driving pins that are not used in your project. Otherwise, Quartus-II drives all the unused pins to ground, which often ends-up creating IO contentions. You may change this option back once you know that all the pins of your project are correctly assigned.

### 7.2 A simple project

Here's one simple Verilog module that makes an LED blink:

```
module ledblink(clk, LED);
  input clk;
  output LED;

  // 32 bits counter
  reg [31:0] cnt;
  always @(posedge clk) cnt <= cnt + 32'h1;

  assign LED = cnt[23];
endmodule
```

To see how to create a Quartus-II project, check the fpga4fun's [Quartus-II quick-start guide](#).

Before compiling it in Quartus-II, don't forget to make the correct pin assignments in Quartus-II menu → Assignments/Pins or "Pin planner". This project uses only 2 pins, so doesn't require much work (using the info from paragraph 8.1).

### 7.3 The serial interface

The Pluto boards can easily send and receive characters to/from a PC's RS-232 port – see the startup-kit's SerialRxTx project. This tests bi-directional communication.

Note that

- More information on the RS-232 HDL code is available at <http://www.fpga4fun.com/SerialInterface.html>
- For PC RS232 sample code, see chapter 16 (Sample C code for RS-232 Win32 send & receive).

## 8 FPGA connections

### 8.1 FPGA pins

The main FPGA signals are:

Pin name	Pluto	Pluto-II	Pluto-3	Direction	Comment
CLK0	91	10	17	FPGA input	25MHz on-board SMD oscillator, always present
CLK1			18	FPGA input	Optional DIL8 oscillator (not present by default, to be soldered above the on-board SMD oscillator) (Pluto-3 only)
LED1	7	25	28	FPGA output	Red LED (active high)
LED2			25	FPGA output	Red LED (active high) (Pluto-3 only)
PB			9	FPGA input	Push-button (active low) (Pluto-3 only)
RxD	77	4	21	FPGA input	RS-232 receive from PC
TxD	78	29	24	FPGA output	RS-232 transmit to PC

### 8.2 IO headers

Many IO signals are available on headers, see the drawings on chapter 18.

### 8.3 Boot-PROM connection (Pluto-II and Pluto-3 only)

The boot-PROM is an EPCS1 or EPCS4, also known as M25P10 or M25P40.

The pinout is as follow:

Boot-PROM pin	Pluto-II FPGA pin	Pluto-3 FPGA pin
Clock	20	15
Data In	17	1
Data Out	5	14
CSn	6	2
HOLDn	40	8
Wn	89	N.A.

## 9 Connectors and headers

### 9.1 Power header

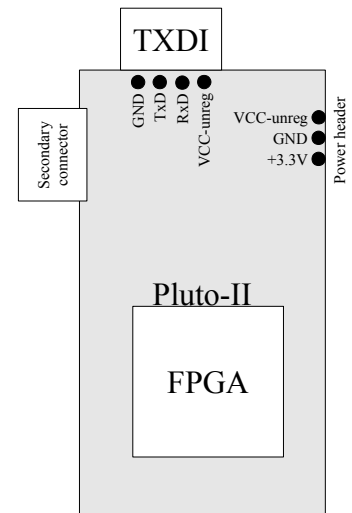
This 3-pins header provides access to the board power signals. It is often used as an output (to power other boards) but can also be used as an input (to power the FPGA board).

On Pluto and Pluto-II, the power header is located on the right side of the board and its pinout is:

1. VCC-unreg (board power, typically +5V to +10V)
2. GND
3. +3.3V (regulated by Pluto/Pluto-II)

On Pluto-3, the power header is located below the TXDI connector and its pinout is:

1. GND
2. +3.3V (regulated by Pluto-3)
3. VCC-unreg (board power, typically +5V to +10V)



### 9.2 TXDI connector

That's the white connector at the top of your FPGA board.

TXDI has 4 pins. It is used to power the FPGA board, and for RS-232 communication.

1. GND
2. TxD (FPGA → PC)
3. RxD (PC → FPGA)
4. VCC-unreg (board power, typically +5V to +10V)

See chapter 15 for more information.

### 9.3 Secondary connector

The secondary connector is an optional connector on the left side of the board. If it is not present, it can be easily soldered. The connector can be purchased from KNJN ([item#1803](#)) or [DigiKey](#).

The secondary connector has only 4 pins (2 powers and 2 IOs), and so is limited to applications requiring a serial bus. Possible applications include:

- Graphic LCD interface
- I<sup>2</sup>C interface

The 4 pins are:

Pin	Pluto	Pluto-II	Pluto-3	Comment
1	Power	Power	Power	Board power (VCC-unreg)
2	FPGA IO pin 8	FPGA IO pin 26	FPGA IO pin 30	RxD or serclk or other use
3	FPGA IO pin 96	FPGA IO pin 27	FPGA IO pin 31	TxD or serdata or other use
4	GND	GND	GND	Ground

In more details:

- Pin 1 is VCC-unreg. This is the voltage that you power your FPGA board with. On Pluto and Pluto-II, VCC-unreg can be disconnected from the secondary connector by cutting the trace placed between the two pads close to the VCC-unreg pin.
- Pins 2 & 3 are two FPGA IO pins. The board includes series termination resistors, and [Zener](#) protection diodes on these 2 signals.
- Some revisions of Pluto-II have another secondary connector on the right side of the board, below the power header.

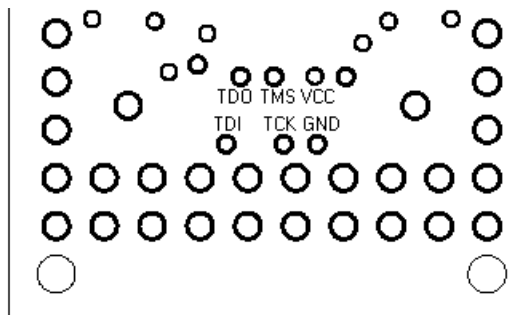
## 10 JTAG connection

### 10.1 JTAG on Pluto

JTAG is not available on Pluto.

### 10.2 JTAG on Pluto-II

The Pluto-II's FPGA JTAG signals are only accessible at the bottom of the board.



To use JTAG, get a JTAG cable (like a ByteBlaster or USB-Blaster), and solder the JTAG signals to the pads shown above.

### 10.3 JTAG on Pluto-3

Pluto-3 has the regular Altera-style 10-pins header, so any Altera or compatible JTAG cable can easily be used. To use it, add a 10-pins shrouded header to the board, like [item#2450 or 2451](#).

# 11 Flashy boards

## 11.1 Flashy daughter-boards

When a Flashy board is used in combination with a KNJN FPGA board, the system becomes a digital oscilloscope (see below the two designs called FlashyMini and FlashyDemo). Check also the [Hands-on - A digital oscilloscope](#) and [Flashy acquisition board](#) pages for more information.

## 11.2 FlashyMini design

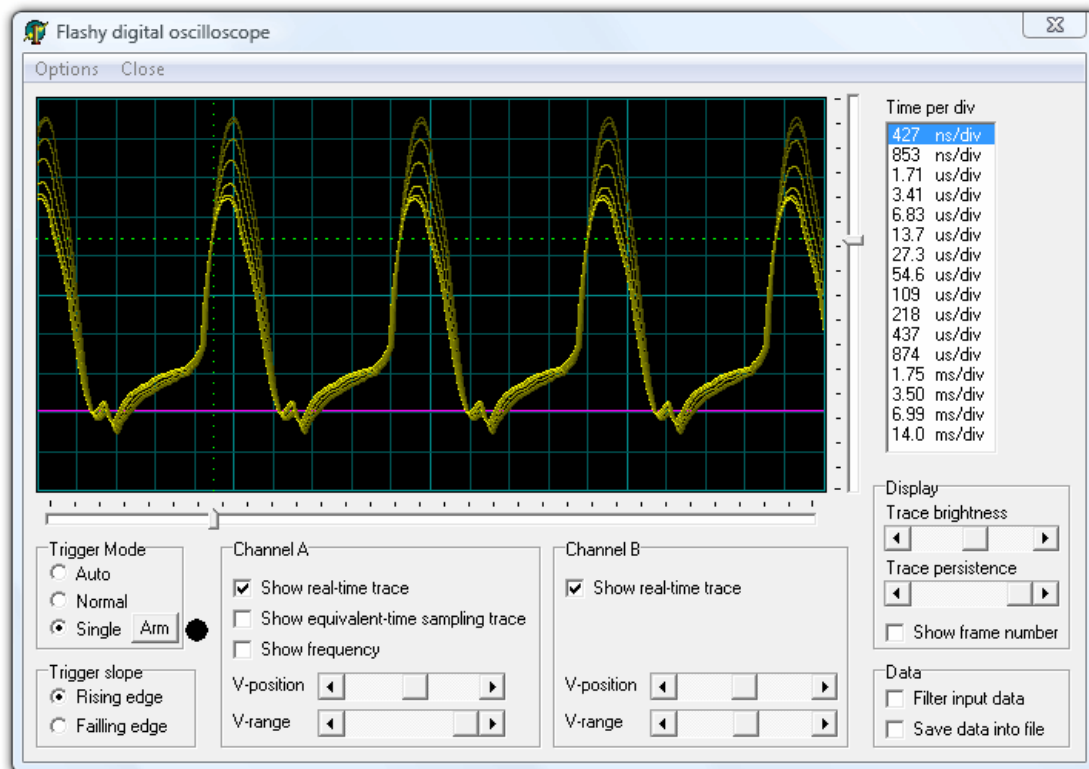
“FlashyMini” is provided with full source code (HDL + C). It shows how to get data from Flashy and can be used as a skeleton to develop your own acquisition system.

## 11.3 FlashyDemo design

FlashyDemo is provided in binary form. It is a showcase of Flashy possibilities, implementing features found in digital oscilloscopes like pre-trigger acquisition and equivalent-time-sampling.

To run FlashyDemo:

1. Mount Flashy on the Pluto board, and power it up.
2. Configure the FPGA with the “FlashyDemo.rbf” FPGA bitfile.
3. Go to Menu → Tools → Flashy Oscilloscope (or press CTRL-F).



Note that there are actually three Flashy boards available (named Flash, Flashy and FlashyD). Here's the FlashyDemo compatibility table.

	Flash	Flashy	FlashyD	LCD (2)
Pluto	Yes	Limited (1)	No	No
Pluto-II	Yes	Yes	No	Yes
Pluto-3	Yes	Yes	Yes	Yes

(1) Pluto's FPGA cannot hold all the FlashyDemo functionality at once, so Pluto is provided with two FlashyDemo bitfiles that each covers some of Flashy features.

(2) The KNJN color LCD [item#5300](#) option can work as a FlashyDemo external display.

---

## 12 FPGA configuration through RS232

### 12.1 Pluto FPGA configuration

Unlike Pluto-II and Pluto-3, Pluto doesn't have an FPGA boot-PROM, so Pluto needs to be configured after each power-up. This is usually done through the serial port of a PC, but it can also be accomplished using a microcontroller (using only one output pin).

There are 2 techniques, depending on the presence or absence of a UART in your microcontroller.

#### With a UART

Set the microcontroller UART at 115200 bauds, and run the following C pseudo-code:

For each byte of the RBF file, do:

```
for(j=0; j<8; j++)    serial.write(((rbfbyte >> j) & 1) ? 0xFF : 0xFE);
```

A more complete example could be:

```
int i, j;
FILE *fpIn = fopen("LEDblink.rbf", "rb" );
char buf[0x10000];
int len = fread(buf, 1, sizeof(buf), fpIn);
fclose(fpIn);

OpenCom();
SetCommBreak(hCom); Sleep(50);      // unconfigure FPGA
ClearCommBreak(hCom);

for(i=0; i<len; i++)
  for(j=0; j<8; j++)
    WriteComChar(((buf[i] >> j) & 1) ? 0xFF : 0xFE); // 8.6µs or 17.3µs pulses

CloseCom();
```

This code also work from a PC, see the chapter 16 for some COM source code.

#### Without a UART

If your microcontroller doesn't have a UART, you can just send pulses on one IO pin. Sending 0xFF above is equivalent to sending a pulse of 8.6µs, while sending 0xFE sends a pulse twice that long. Pulses are positive (inactive level is "0") and they need to be separated by 30µs or so between them. To unconfigure Pluto (before sending the RBF), send a "break" condition (a high signal) for about 50ms.

The RBF file is a binary file. Pluto's RBF file weights close to 20KB (uncompressed). It is usually highly compressible (full of 0x00's), in case you run out of memory in the microcontroller.

### 12.2 Pluto-II FPGA configuration

Pluto-II uses the same configuration scheme as Pluto.

Pluto-II also has a boot-PROM so doesn't require configuration at each power-up if the boot-PROM has been programmed.

### 12.3 Pluto-3 FPGA configuration

Pluto-3's configuration scheme is simpler than Pluto/-II's. To configure the FPGA, just send the RBF binary content through RS232 at 115200 bauds in 8-bits mode. To unconfigure the FPGA (before sending the RBF), send a "break" condition (a high signal) for about 50ms.

Pluto-3 also has a boot-PROM so doesn't require configuration at each power-up if the boot-PROM has been programmed.

## 13 Quartus-II JTAG indirect mode

### 13.1 What is it?

The JTAG indirect mode allows programming the FPGA boot-PROM through JTAG.

It is a two steps process: first a .jic (“JTAG indirect configuration”) file is created, and then the .jic file is used to program the boot-PROM

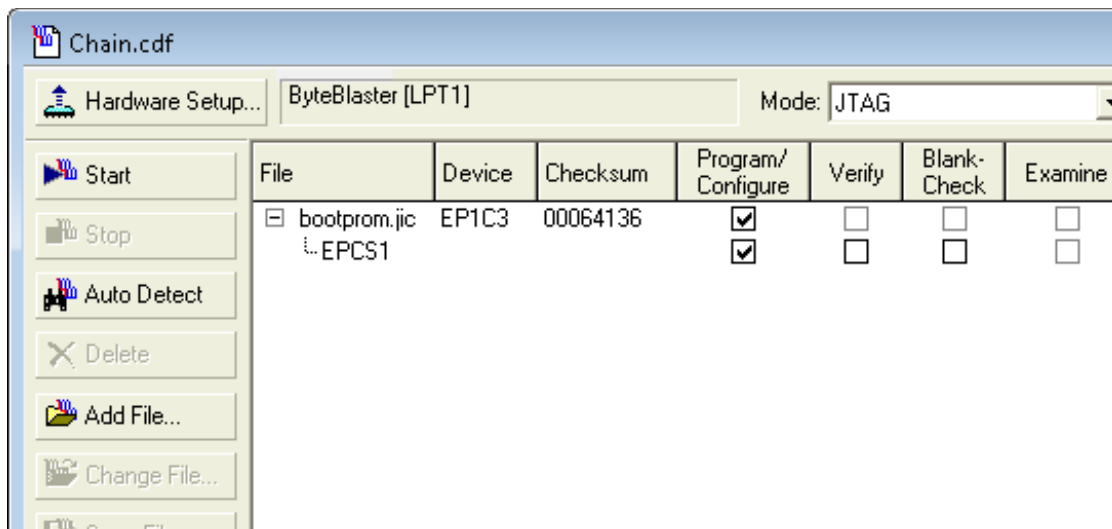
### 13.2 Create a “JTAG indirect configuration” file

1. In Quartus-II, go to File → Convert programming file
2. In the “Output programming file” panel, select “JTAG Indirect Configuration File”, EPCS1 for Pluto-II or EPCS4 for Pluto-3, and a filename for a “.jic” file.
3. In “Input file to convert”, select the Cyclone EP1C3 for Pluto-II or EP2C5 for Pluto-3 as “Flash loader”, and the SOF file that you want to use for the boot-PROM as “SOF Data”.
4. Click “OK”. This creates the “.jic” file.

The boot-PROM can now be programmed.

### 13.3 Program the boot-PROM

1. Open the “Programmer” window (in Tools → Programmer)
2. Select the JTAG cable you are using (“Hardware Setup” button)
3. Select JTAG as “Mode”
4. Load the “.jic” file
5. Select configure for both the FPGA (EP1C3/EP2C5) and the boot-PROM (EPCS1/EPCS4)
6. Click the “Start” button.



7. You can also verify and erase the boot-PROM if you want (note: the FPGA doesn't need to be re-configured if it is already configured from the jic).

For more information, check Altera's [AN-370](#).

## 14 Power requirements

The Pluto boards have their own voltage regulator, so don't have stringent requirements on a power supply.

### 14.1 Wall adapter

Most common (semi-regulated) 5V to 10V "wall adapter" DC supply works fine.



In practice, the Pluto boards work with an input voltage as low as 4.5V, or as high as 15V (although the voltage regulator might become hot in the later case, so it is better to keep the input voltage low).

### 14.2 USB to power jack cable

The "USB to power jack cable" (picture on the right) is another simple way to power the Pluto boards. It is available on KNJN's [power cables](#) page (item#6025).



### 14.3 Power consumption

The Pluto boards don't consume much by themselves (usually less than 100mA). The consumption depends more of what you connect to them. A supply that provides a few 100mA is adequate for most applications.

### 14.4 Voltage regulator temperature

The Pluto boards voltage regulator can get hot in some instances. Here are the two things to do in this case:

1. Check your DC-adapter output voltage. The higher it is, the warmer the voltage regulator gets, so try to use one adapter with a low voltage output (+5V is ideal).
2. Check the current consumption: the more current drawn out of the regulator, the warmer it gets. So if your FPGA board is heavily loaded (lots of IOs connected), the regulator may get hot.

If the voltage regulator gets too hot, it shuts down automatically and the board stops working temporarily.

## 15 Connecting the Pluto boards to a PC

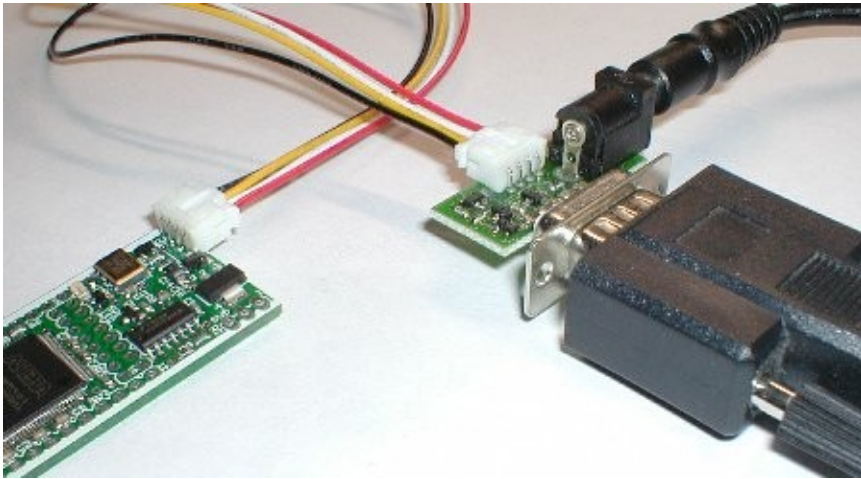
### 15.1 Serial connection

The Pluto boards connects to a PC's RS-232 port.

The recommended way is through a TXDI board, although it is not mandatory. The TXDI board simplifies the Pluto boards connection as it combines the bulky RS232 connection and the power connection into one cable. If a TXDI is not desired, a simple DB-9 connector can be used.

### 15.2 With a TXDI

Connect the Pluto and TXDI has shown below (the small multi-color cable between the two is provided).



**IMPORTANT:** Make sure the DC-plug is center-positive.

### 15.3 With a TXDI/MAX232 or TXDI/FTDI

The TXDI/MAX232 and TXDI/FTDI are more versatile than the regular TXDI as they can also be used with the secondary connector (to create a second RS232 port).

TXDI connector:

- With TXDI/MAX232, place the jumper on the "B" position.
- With TXDI/FTDI, TxD needs to be inverted (use FTDI's [FT\\_PROG](#) utility if required).  
Note: the TXDI/FTDI board resets the FPGA when the USB cable is plugged, which may prevent or disturb the FPGA boot-PROM configuration process at start-up.

Secondary connector:

- With TXDI/MAX232, place the jumper on the "M" position.
- With TXDI/FTDI, don't invert the TxD line (use FTDI's [FT\\_PROG](#) utility if required).

### 15.4 Without a TXDI

In the absence of a TXDI, the Pluto boards are shipped with a small cable. Connect it to a DB-9 female connector as follow:

- White wire to DB-9 pin 3.
- Black wire to DB-9 pin 5 and power supply GND.
- Red wire to DC power supply (+5V to +10V).

In short, the board is powered using the black and red wires, and the PC sends data to the board through the white wire. Note that this method provides only unidirectional communication with the FPGA board (the PC can send RS232 data to the FPGA, but the FPGA cannot send data to the PC) so cannot be used to program the Pluto-II/-3 boot-PROM.

## 16 Sample C code for RS-232 Win32 send & receive

```
#include <windows.h>
HANDLE hCom;

void ExitOnError(char*message)
{
    printf("%s error", message);
    exit(1);
}

void OpenCom(char* COM_name)
{
    DCB dcb;
    COMMTIMEOUTS ct;

    hCom = CreateFile(COM_name, GENERIC_READ | GENERIC_WRITE, 0, NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
    if(hCom==INVALID_HANDLE_VALUE) ExitOnError(COM_name); // can't open COM port
    if(!SetupComm(hCom, 4096, 4096)) ExitOnError("SetupComm");

    if(!GetCommState(hCom, &dcb)) ExitOnError("GetCommState");
    dcb.BaudRate = 115200;
    ((DWORD*)&dcb)[2] = 0x1001; // set port properties for TXDI + no flow-control
    dcb.ByteSize = 8;
    dcb.Parity = NOPARITY;
    dcb.StopBits = 2;
    if(!SetCommState(hCom, &dcb)) ExitOnError("SetCommState");

    // set the timeouts to 0
    ct.ReadIntervalTimeout = MAXDWORD;
    ct.ReadTotalTimeoutMultiplier = 0;
    ct.ReadTotalTimeoutConstant = 0;
    ct.WriteTotalTimeoutMultiplier = 0;
    ct.WriteTotalTimeoutConstant = 0;
    if(!SetCommTimeouts(hCom, &ct)) ExitOnError("SetCommTimeouts");
}

void CloseCom()
{
    CloseHandle(hCom);
}

DWORD WriteCom(char* buf, int len)
{
    DWORD nSend;
    if(!WriteFile(hCom, buf, len, &nSend, NULL)) exit(1);

    return nSend;
}

void WriteComChar(char b)
{
    WriteCom(&b, 1);
}

int ReadCom(char *buf, int len)
{
    DWORD nRec;
    if(!ReadFile(hCom, buf, len, &nRec, NULL)) exit(1);

    return (int)nRec;
}

char ReadComChar()
{
    DWORD nRec;
    char c;
    if(!ReadFile(hCom, &c, 1, &nRec, NULL)) exit(1);

    return nRec ? c : 0;
}

void main()
{
    OpenCom("COM1:"); // change that to use a different COM port
    WriteComChar(0x41);
    CloseCom();
}
```

---

## 17 Board checklist

### 17.1 The FPGA doesn't configure?

If the FPGA doesn't configure, check the following:

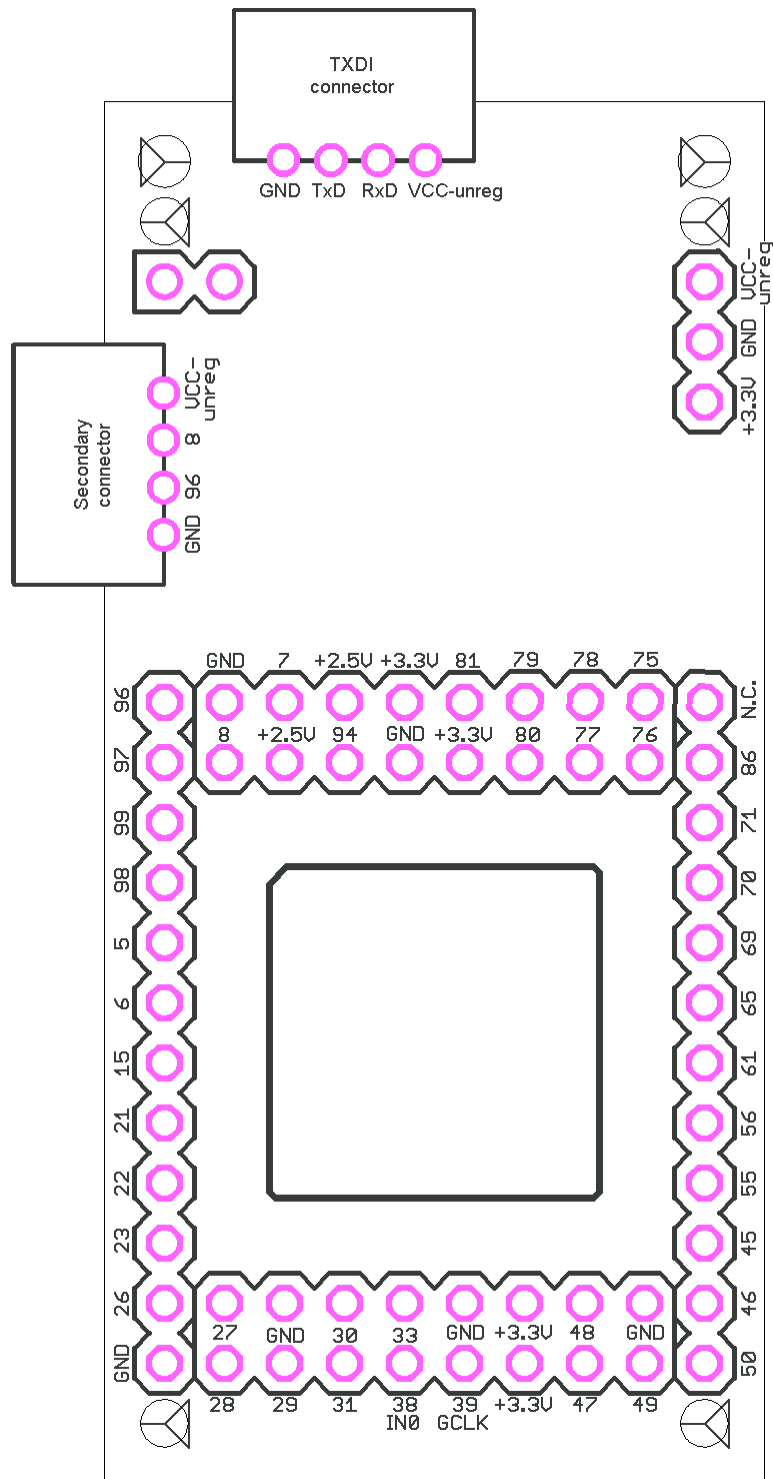
1. Make sure the board is powered with a correct voltage. You need 5V minimum at the input of your Pluto board. If you use a TXDI with an integrated 5V regulator, it requires at least 7V to be able to provide 5V to Pluto.
2. Try a known good RBF file, like LEDblink.rbf or LEDglow.rbf
3. Make sure you are using the right COM port on the back of your PC (or try another COM port in FPGAconf / Options / COM port).
4. Make sure the voltage on RxD (the white wire) is negative when the RS-232 is idle. If you are using a TXDI/MAX-232, check that the jumper position is on "bypass".
5. If you're using an RS-232 cable, try without it (connect the TXDI/DB9 directly on the back of your computer).
6. Enable the option "Keep COM port open after configuration" in FPGAconf.
7. Disable the option "Turbo Mode" in FPGAconf.
8. Try a different computer, in case your computer has some troubles communicating at 115200 bauds. Also, some computers RS-232 levels may be too low, but Pluto/Pluto-II can be adapted in such cases by removing a resistor.

### 17.2 Boot-PROM problem?

1. If you cannot program the FPGA boot-PROM and get an error "No boot-PROM found" or "Error communication timeout", check the Boot-PROM requirements in paragraph 4.3.
2. If the boot-PROM programming starts but fails in the verify phase, that's because your RS232 connection works unreliably. With laptops, that's usually due to low-level RS232 signals. The workaround is to use a [TXDI/MAX232](#).
3. If the boot-PROM programming works but the FPGA fails to configure at power-up, make sure you didn't change the programming option in your Quartus-II project (it should be "Active serial").

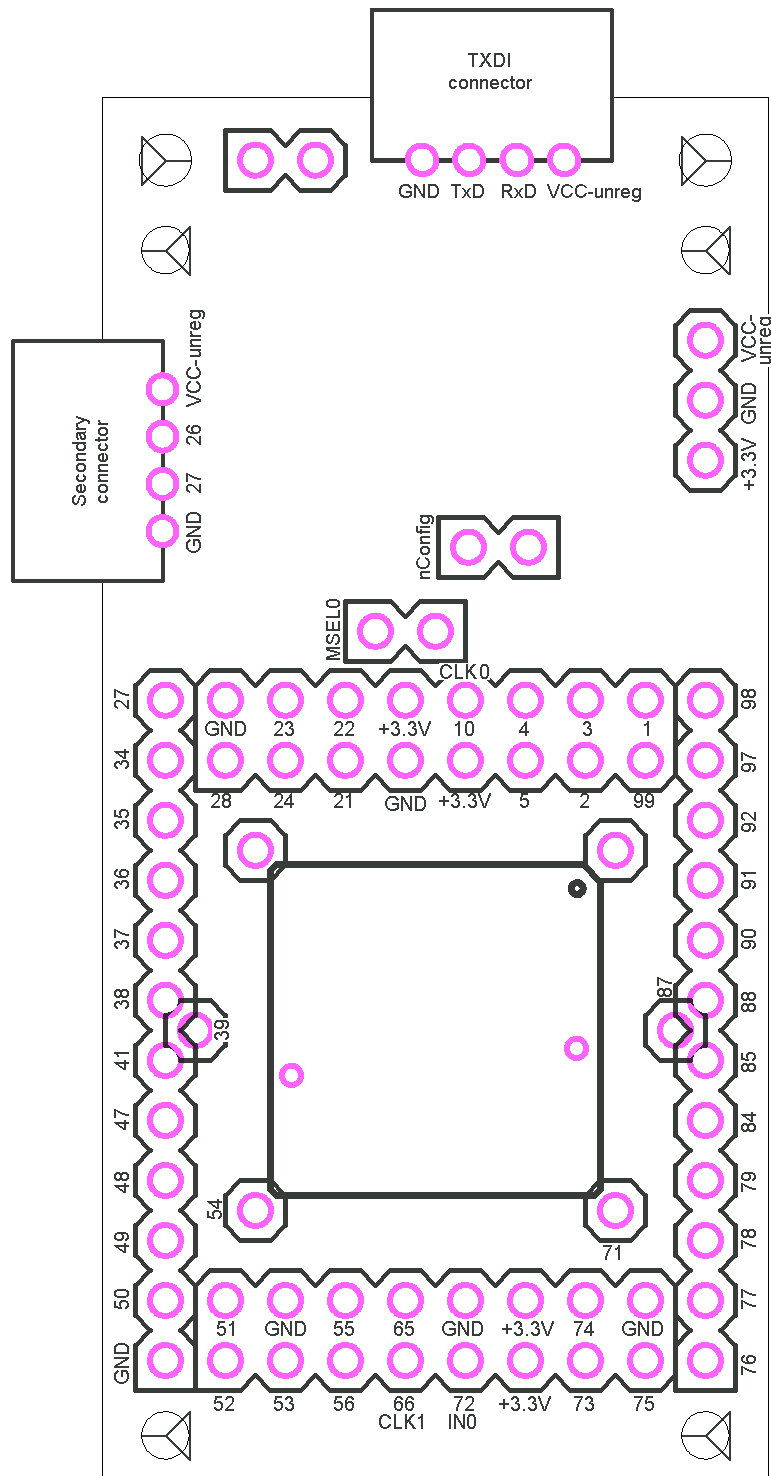
# 18 Board connectors and headers, with IO pin assignments

## 18.1 Pluto



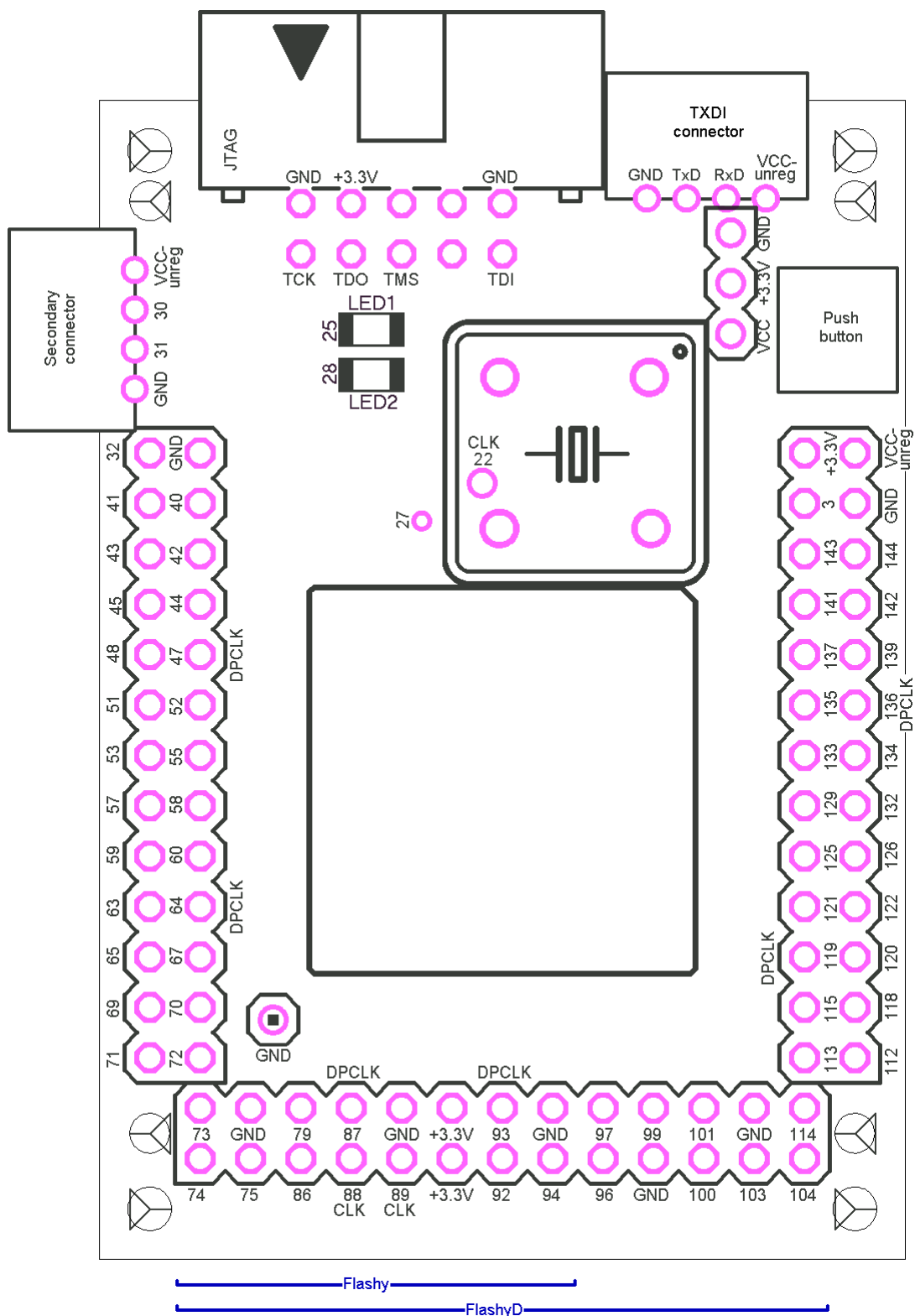
- Pluto has 39+2 IOs available (the +2 are one clock and one dedicated input). Pin 39 is a dedicated clock input. Pin 38 can also be used as a clock input.
- All IOs use 3.3V powered banks and are 5V input tolerant. Check [Altera ACEX family datasheet](#) for more details.

## 18.2 Pluto-II



- Pluto-II has 50+1 IOs available (the +1 is one dedicated clock input). Pin 66 is a dedicated clock input. Pins 34, 72, 84 and 92 can also be used as clock inputs.
- All IOs use 3.3V powered banks that can use different IO standards like LVTTTL or LVCMOS. Unlike Pluto, Pluto-II IOs are not directly 5V tolerant. Check the [Altera Cyclone device handbook](#) for more details.

## 18.3 Pluto-3



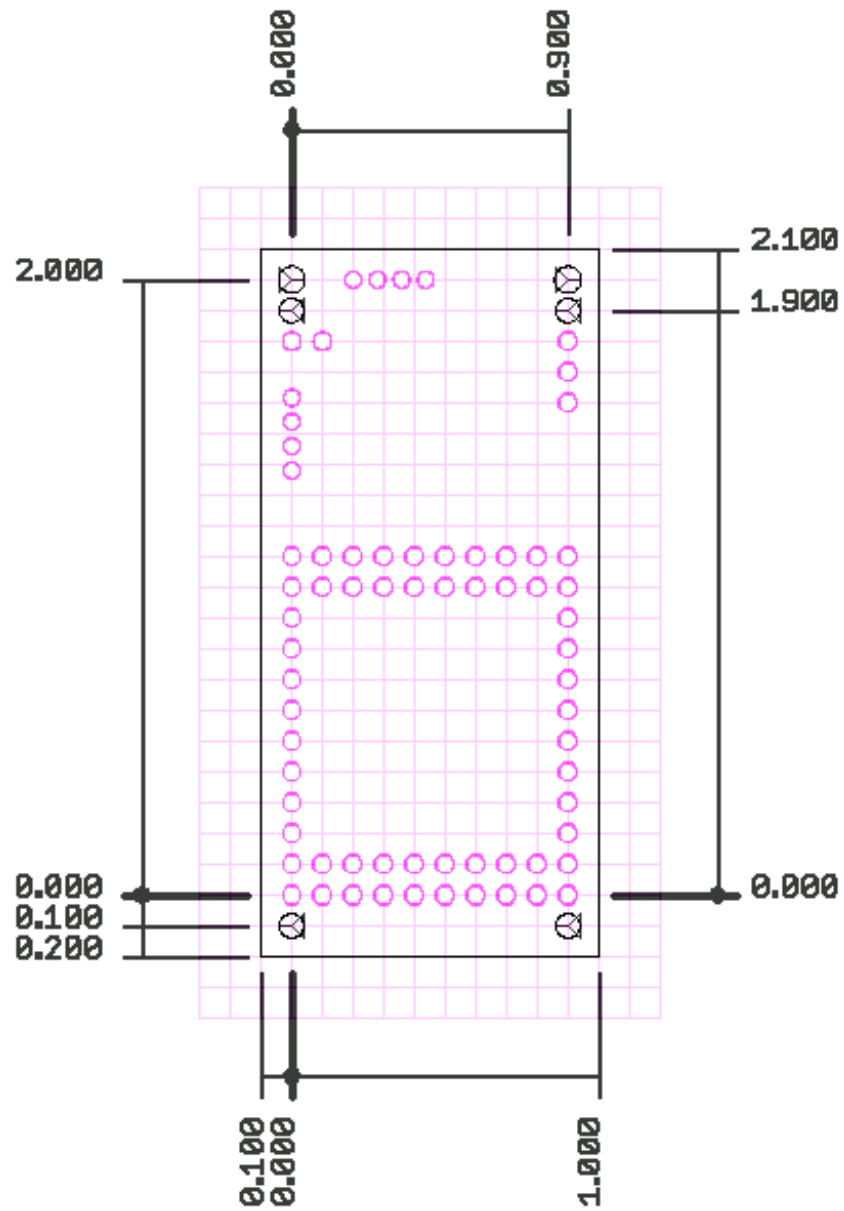
- Many pins can be used as clock: CLK6 (pin 89), CLK7 (pin 88), DPCLK2 (pin 47), DPCLK4 (pin 64), DPCLK6 (pin 87), DPCLK7 (pin 93), DPCLK8 (pin 119), DPCLK10 (pin 136). Also CLK3 (pin 22) is available on a pad.
- All IOs use 3.3V powered banks that can use different IO standards like LVTTTL or LVC MOS. They are not directly 5V tolerant. Check the [Altera Cyclone-II device handbook](#) for more details.
- Pluto-3 revision A and B are very similar. The revision B shown here rotates the JTAG connector to allow for a straight as well as a right-angle JTAG connector. All other pin assignments are the same.

## 19 Mechanical drawings

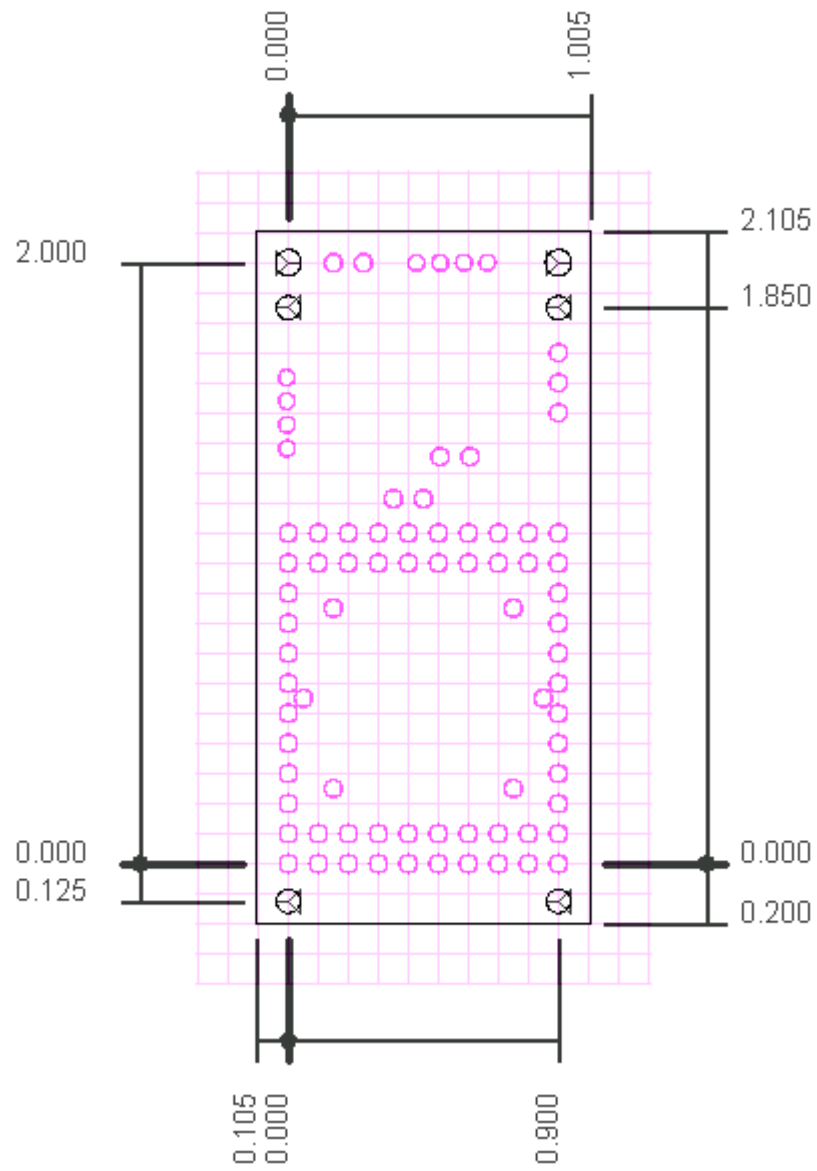
All dimensions are given in inches (1" = 25.4mm).

The grid is drawn using 0.1" steps (2.54mm).

### 19.1 Pluto



## 19.2 Pluto-II



### 19.3 Pluto-3

