

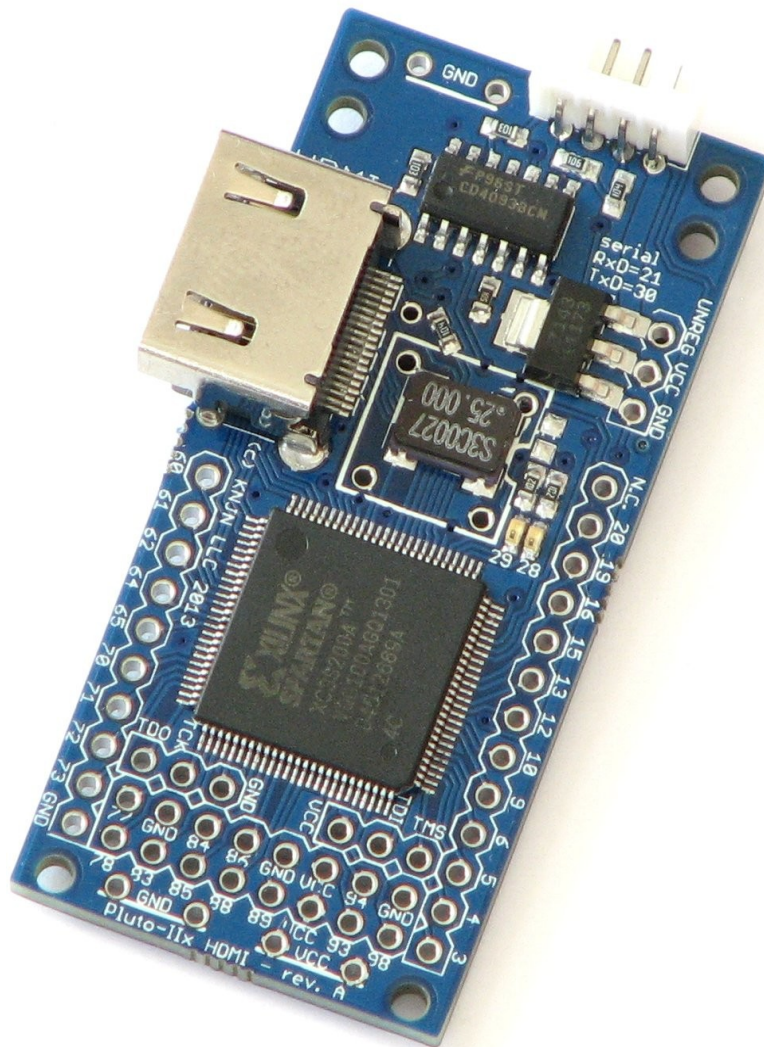
# FPGA serial development boards

© 2004 to 2023 KNJN LLC

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

This document applies to the following boards:

- Pluto rev. F
- Pluto-IIx / HDMI / USB-C
- Pluto-3 rev. B and C



Document last revision on **October 28, 2023**

---

# Table of Contents

1	Welcome.....	5
1.1	This guide.....	5
1.2	Why serial FPGA boards?.....	5
1.3	The Pluto boards.....	5
1.4	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 (Pluto-IIx/Pluto-3).....	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
4.5	Boot-PROM on-demand FPGA configuration.....	8
5	FPGAconf extras.....	9
5.1	Auto configuration mode.....	9
5.2	Scrollbar.....	9
5.3	Terminal.....	9
6	FPGA configuration using Quartus-II JTAG support (Pluto-3).....	10
6.1	JTAG requirements.....	10
6.2	JTAG configuration.....	10
7	FPGA project using Quartus-II (Pluto/Pluto-3).....	11
7.1	Create a new project.....	11
7.2	A simple start.....	11
8	FPGA projects with Xilinx's ISE (Pluto-IIx).....	12
8.1	Create a new project.....	12
8.2	A simple start.....	12
9	FPGA connections.....	13
9.1	FPGA pins.....	13
9.2	IO headers.....	13
9.3	Boot-PROM connection (Pluto-IIx/Pluto-3).....	13
9.4	HDMI (Pluto-IIx).....	13
9.5	Power header.....	13
9.6	TXDI connector.....	14
9.7	Secondary connector.....	14
10	JTAG connection.....	15
10.1	JTAG on Pluto.....	15
10.2	JTAG on Pluto-IIx.....	15
10.3	JTAG on Pluto-3.....	15
11	Flashy boards.....	16
11.1	FlashyMini design.....	16
11.2	FlashyDemo design.....	16
12	FPGA configuration through the serial port.....	17
12.1	Pluto/-IIx FPGA configuration.....	17
	With a UART.....	17
	Without a UART.....	17
12.2	Pluto-3 FPGA configuration.....	17
13	Quartus-II JTAG indirect mode (Pluto-3).....	18
13.1	What is it?.....	18
13.2	Create a "JTAG indirect configuration" file.....	18
13.3	Program the boot-PROM.....	18
14	Power requirements.....	19
14.1	Wall adapter.....	19
14.2	USB to power jack cable.....	19
14.3	Power consumption.....	19

---

14.4	Voltage regulator temperature.....	19
15	Connecting the Pluto boards to a PC (TXDI interface).....	20
15.1	Serial connection.....	20
15.2	With a TXDI.....	20
15.3	With a TXDI/MAX232 or TXDI/FTDI.....	20
15.4	Without a TXDI.....	20
16	Sample C code for serial Win32 send & receive.....	21
17	Board checklist.....	22
17.1	The FPGA doesn't configure?.....	22
17.2	Boot-PROM problem?.....	22
18	Board connectors and headers, with IO pin assignments.....	23
18.1	Pluto.....	23
18.2	Pluto-IIx.....	24
18.3	Pluto-IIx USB-C.....	25
18.4	Pluto-IIx HDMI.....	26
18.5	Pluto-IIx HDMI USB-C.....	27
18.6	Pluto-3.....	28
19	Mechanical drawings.....	29
19.1	Pluto.....	29
19.2	Pluto-IIx (all versions).....	30
19.3	Pluto-3.....	31



# 1 Welcome

## 1.1 This guide

Welcome to the KNJN FPGA serial 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 Why serial FPGA boards?

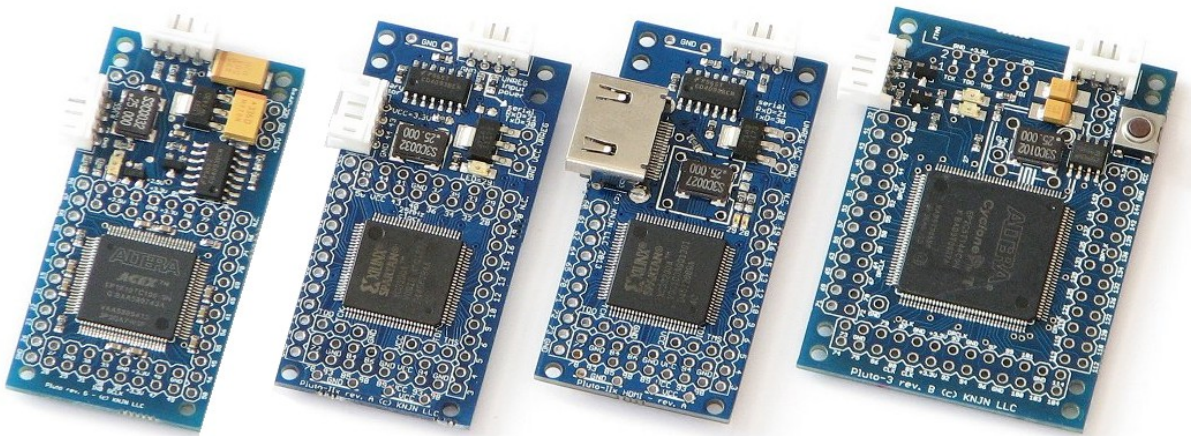
Serial interfaces provide an easy way to interact with FPGAs, and so are valuable as learning tools.

The serial interface is either a TXDI (similar to RS-232) or USB-C.

## 1.3 The Pluto boards

The boards are mainly differentiated by

- The FPGA model
- The serial interface (TXDI or USB-C)
- HDMI port



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

## 1.4 Boards characteristics

	Pluto	Pluto-IIx	Pluto-IIx HDMI	Pluto-3
FPGA	EP1K10	XC3S50A or XC3S200A	XC3S200A	EP2C5
Datasheet (PDF)	<a href="#">ACEX 1K</a>	<a href="#">Spartan-3A</a>	<a href="#">Spartan-3A</a>	<a href="#">Cyclone II</a>
Serial interface	TXDI	TXDI or USB-C	TXDI or USB-C	TXDI
Logic cells	576	1584 or 4032	4032	4608
IO pins	41	48	36	65
PLL/DLL	-	DLL	DLL	PLL
External clocks	up to 2	up to 4	up to 4	up to 4
Boot-PROM (1)	-	4Mbit	8Mbit	4Mbit
On-board oscillator	25MHz	25MHz	25MHz	25MHz
DIL8 oscillator header	-	-	Yes	Yes
Push-button	-	-	-	Yes
JTAG header	-	(see chapter 10)	(see chapter 10)	Yes
LED(s)	1	1	2	2
ADC board ready	Flashy / Widy	Flashy / Widy	Flashy / Widy	FlashyD / WidyD
Dimensions	58 x 28 mm	58 x 28 mm	58 x 28 mm	58 x 41 mm

(1) Minimum boot-PROM size shown here. Actual product may use a higher capacity boot-PROM.

---

## 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.

Board	Software
Pluto	<a href="#">Quartus II Web Edition 9.0 SP2</a> (1.3GB)
Pluto-IIx	<a href="#">ISE WebPACK 14.7</a>
Pluto-3	<a href="#">Quartus II Web Edition 13.0 SP1</a> (4.4GB)

### 2.3 C/C++ compiler

A C/C++ compiler is optional but you'll need one for many projects.

Here are examples of different compilers that can be used:

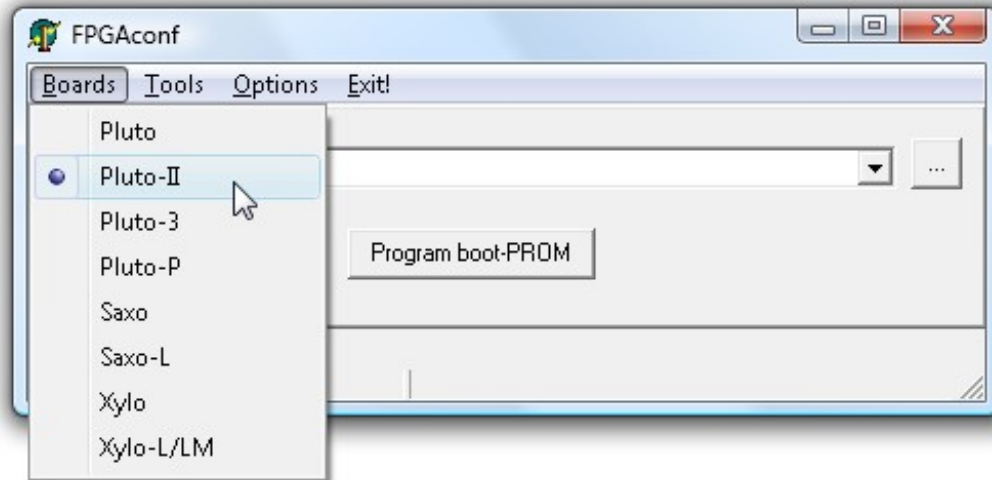
- [Microsoft Visual Studio Community](#)
- [Digital Mars C++](#)
- Jacob Navia's [lcc-win32](#)

## 3 FPGAconf

FPGAconf is a multifunction software provided with your 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, BIT or BIN file (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 files are provided in the startup-kit. In particular, try "LEDblink" and "LEDglow" from the LED directory.

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)

## 4 FPGA boot-PROM (Pluto-IIx/Pluto-3)

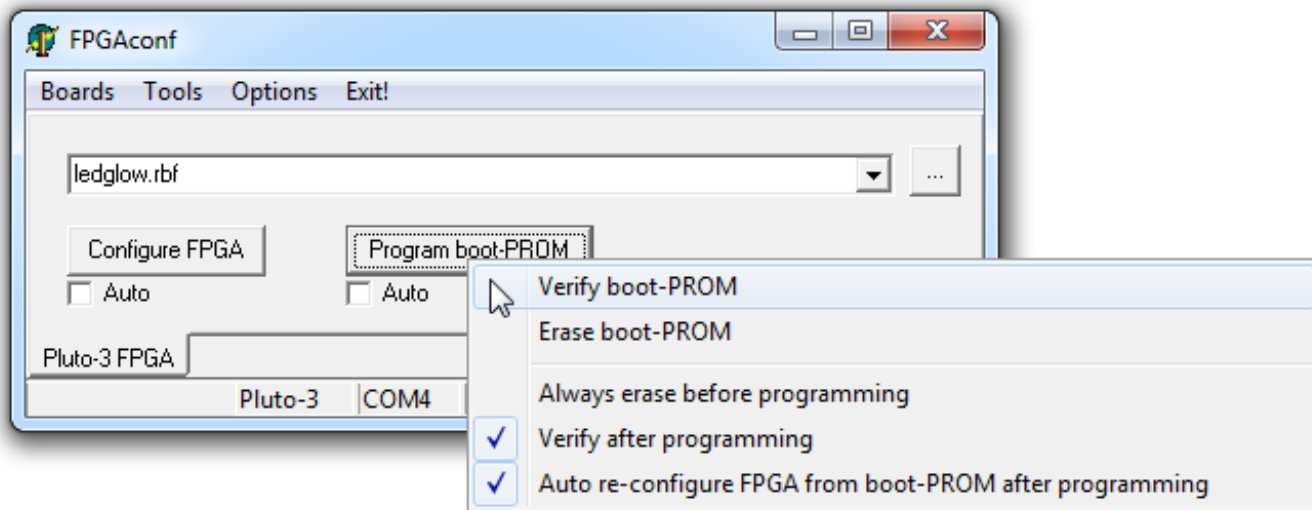
### 4.1 What's the boot-PROM?

The boot-PROM is a serial flash memory that is read by the FPGA at power-up to get configuration data. If the boot-PROM is empty or its content is invalid, the FPGA stays un-configured and the boot-PROM gets “out of the way” to allow FPGA configuration from another source (serial or JTAG).

### 4.2 Boot-PROM actions

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

- To program the boot-PROM, select a bitfile and left-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, try the SerialRxTx project to diagnose the communication.

### 4.4 Boot-PROM and JTAG

The boot-PROM can also be programmed from JTAG, although it is usually less convenient.

- For Pluto-3, use Quartus-II JTAG indirect mode (chapter 13).
- For Pluto-IIx, use ISE iMPACT (part of ISE WebPACK).

### 4.5 Boot-PROM on-demand FPGA configuration

The boot-PROM can also configure the FPGA “on-demand” (i.e. under software control after power-up).

Here's a summary of all the boot-PROM features.

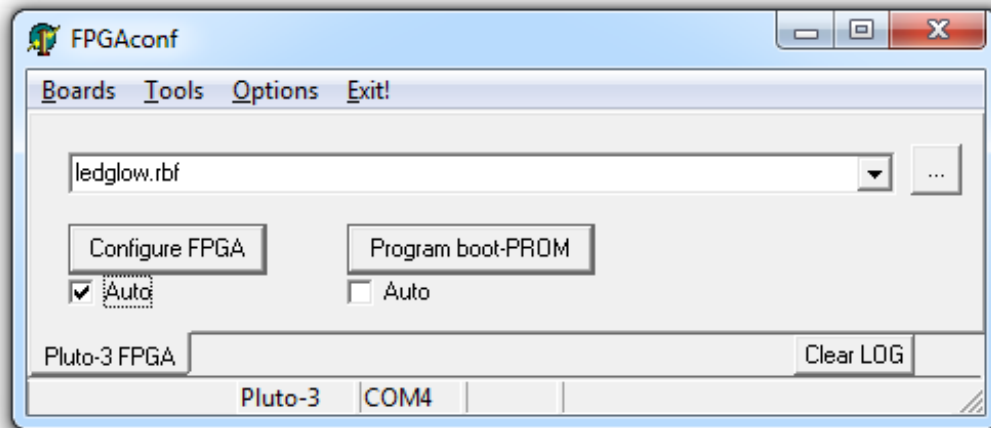
Boot-PROM	Pluto	Pluto-IIx	Pluto-3
Configures FPGA at power-up	N/A	Yes	Yes
Can be programmed through the serial port	N/A	Yes	Yes
Can be programmed through JTAG	N/A	Yes	Yes
Can configure the FPGA on-demand (after power-up)	N/A	Yes	Yes



## 5 FPGAconf extras

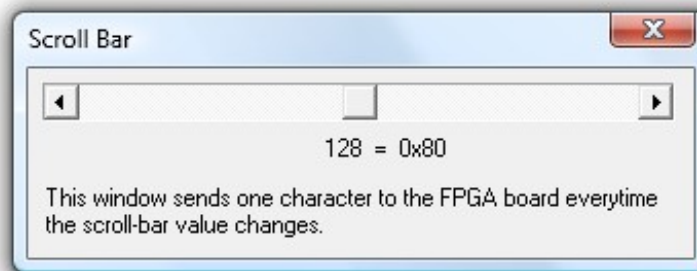
### 5.1 Auto configuration mode

When the “Auto mode” is enabled, FPGAconf monitors the bitfile and takes action each time the file is updated. Useful for example if you want to re-configure the FPGA automatically after each ISE or 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 an serial terminal window that is activated by pressing CTRL-T.

## 6 FPGA configuration using Quartus-II JTAG support (Pluto-3)

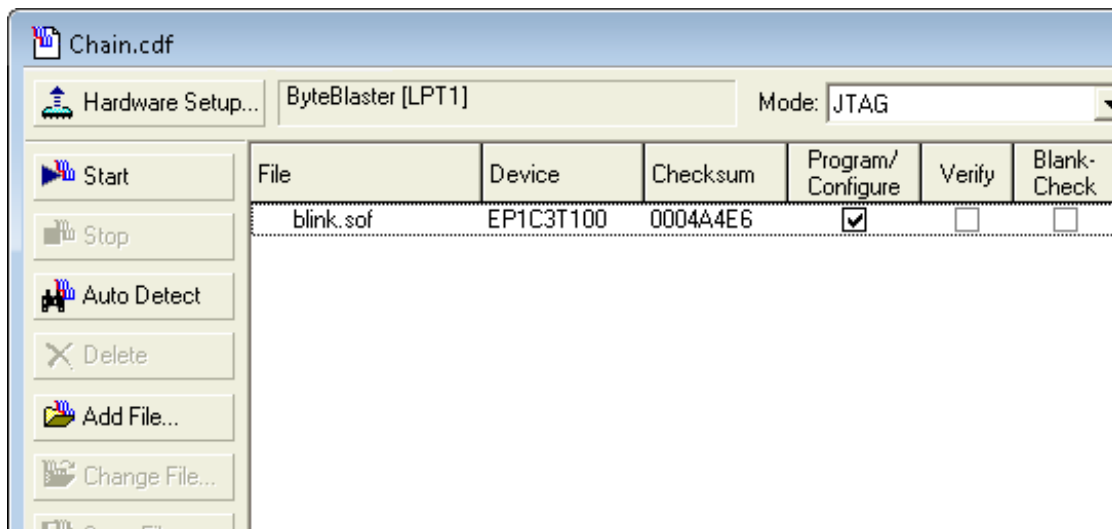
### 6.1 JTAG requirements

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 7.2.

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

## 7 FPGA project using Quartus-II (Pluto/Pluto-3)

Pluto, Pluto-II and Pluto-3 are configured from SOF or RBF files generated by Altera's Quartus-II software.

### 7.1 Create a new project

1. Run Quartus-II, and click on menu → File → New Project Wizard.
2. Select the project location, choose a project name, and click Next.
3. Choose files to add to the project. Just click next if you don't have files to add now.
4. Now is time to choose the device (you can also do that later using menu → Assignments → Device)
  - a. For Pluto, choose family “APEX1K” and device “EP1K10TC100-3”.
  - b. For Pluto-3, choose family “Cyclone-II” and device “EP2C5T144C8”.
5. Click Finish.

A graphical work-through is also available on [this fpga4fun page](#).

### 7.2 A simple start

Here's a simple Verilog file:

```
module LEDblink(  
    input clk,  
    output LED  
);  
  
reg [31:0] cnt;  
always @(posedge clk) cnt <= cnt + 1;    // 32 bits counter  
  
assign LED = cnt[23];  
endmodule
```

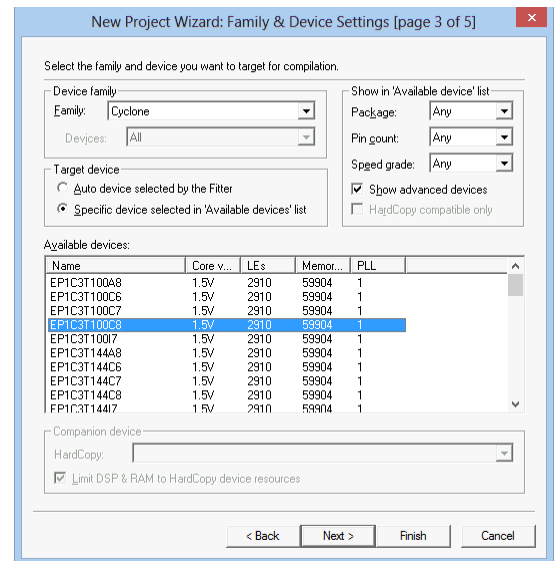
Add it to the project and select it as the top-level design. Next make the correct pin assignments in Quartus-II menu → Assignments/Pins or “Pin planner” (using the info from paragraph 9.1). This project uses only 2 pins, so it should be fast.

You also want to specify the outputs and what happens to unused pins.

1. Select menu → Assignments → Device
2. 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” or “As inputs with weak pull-up”.
  - c. Click “OK”.
3. Click “OK”.

Option 2.a makes sure RBF files are generated (used for serial FPGA configuration). Otherwise only SOF files are generated (used for JTAG).

Option 2.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.



## 8 FPGA projects with Xilinx's ISE (Pluto-Ilx)

The Pluto-Ilx board is configured from BIT files generated by Xilinx's ISE software.

### 8.1 Create a new project

1. Run ISE Project Navigator, and click on menu → File → New Project.
2. Choose a project name, select the project location, and click Next.
3. Select the “Spartan3A and Spartan3AN” family and the device on your board, either
  - XC3S50A in VQ100 package
  - XC3S200A in VQ100 package
4. Click Next twice and Finish to close the wizard.

You can now create or add source files in the project.

A graphical work-through is also available on [this fpga4fun page](#).

### 8.2 A simple start

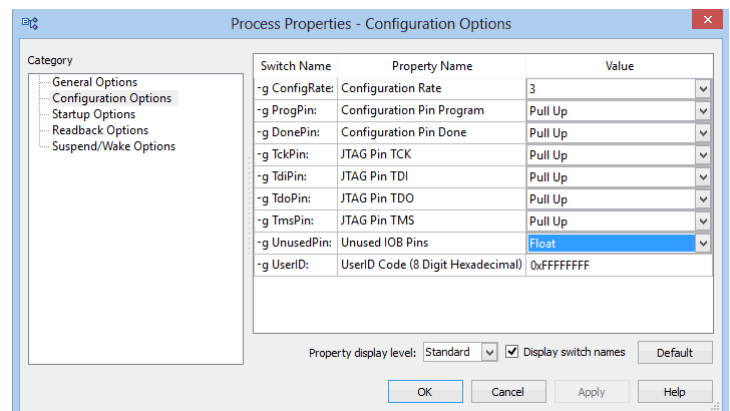
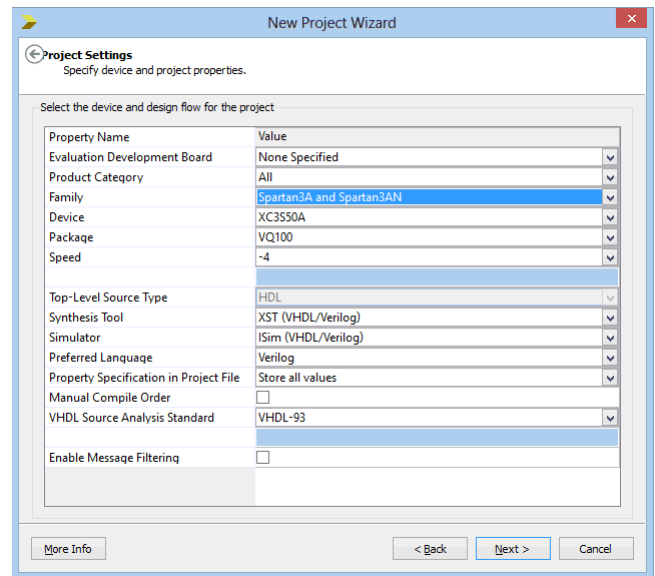
Here's a simple Verilog file:

```
module LEDblink(  
    input clk,  
    output LED  
);  
  
reg [31:0] cnt;  
always @(posedge clk) cnt <= cnt + 1;    // 32 bits counter  
  
assign LED = cnt[23];  
endmodule
```

Add it to the project and select it as the top-level design in your project.  
Now add this UCF file to the project.

```
NET "clk" LOC = P40;  
NET "LED" LOC = P29;
```

Finally right-click on “Generate Programming File” and choose “Process Properties”. In “Configuration Options”, allow “Unused IOB Pins” to “Float”. That prevents the FPGA from driving the pins that are not used in your project. Otherwise grounding all the unused pins (the default) often ends-up creating IO contentions.



## 9 FPGA connections

### 9.1 FPGA pins

The main FPGA signals are:

Pin name	Pluto	Pluto-IIx	Pluto-IIx HDMI	Pluto-3	Direction	Comment
CLK0	91	40	40	17	FPGA input	25MHz on-board SMD oscillator, always present
CLK1			41	18	FPGA input	Optional DIL8 oscillator (1)
LED1	7	29	29	28	FPGA output	Red LED (active high)
LED2			28	25	FPGA output	Red LED (active high)
PB				9	FPGA input	Push-button (active low)
RxD	77	21	21	21	FPGA input	FPGA receives from PC
TxD	78	30	30	24	FPGA output	FPGA transmits to PC

(1) The optional oscillator is not present by default. It is to be added above the on-board SMD oscillator.

### 9.2 IO headers

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

### 9.3 Boot-PROM connection (Pluto-IIx/Pluto-3)

The boot-PROM is an M25P10 (1Mbit) to M25P80 (8MBit), or equivalent.

The pinout is as follow:

Boot-PROM pin	Pluto-IIx FPGA pin	Pluto-3 FPGA pin
Clock	53	15
Data In	46	1
Data Out	51	14
CSn	27	2
HOLDn	31	8
Wn	N.A.	N.A.

### 9.4 HDMI (Pluto-IIx)

Each HDMI port uses 8 FPGA pins (4 differential pairs), plus some auxiliary lines (DDC I2C and “hot plug detect”).

- Pluto-IIx board: it doesn't have a native HDMI port but can be fitted with an optional HDMI adapter.
- Pluto-IIx HDMI board: it has a native HDMI port, and can also use an optional HDMI adapter, so in effect has two possible HDMI outputs.

	TMDS clock	TMDS 0 (blue)	TMDS 1 (green)	TMDS 2 (red)	I2C SDA	I2C SCL	Hot plug detect
Pluto-IIx HDMI (native output)	34 / 35	36 / 37 (1)	43 / 44 (1)	48 / 49	50	33	32
Pluto-IIx optional HDMI adapter	77 / 78	83 / 84 (1)	88 / 89 (1)	93 / 94	(2)	(2)	(2)

(1) Inverted TMDS lane – place an inverter in the FPGA. See the HDMI source code “TMDS\_encoder” instantiations.

(2) If desired, the HDMI auxiliary lines can be wired manually to the FPGA.

### 9.5 Power header

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

On Pluto, the power header pinout is:

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

On Pluto-IIx and Pluto-3, the power header pinout is:

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

---

## 9.6 TXDI connector

It is usually used to power the Pluto board and for serial 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.

## 9.7 Secondary connector

The secondary connector is placed on the left side of the board. If it is not present, it can be easily soldered (available as KNJN [item#1804](#) or from [DigiKey](#)).

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

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

The pin assignments are:

Pin	Pluto	Pluto-Ilx	Pluto-Ilx HDMI	Pluto-3	Comment
1	VCC-unreg	3.3V	3.3V	VCC-unreg	Power
2	FPGA IO pin 8	FPGA IO pin 43	FPGA IO pin 57	FPGA IO pin 30	RxD or serclk or other use
3	FPGA IO pin 96	FPGA IO pin 44	FPGA IO pin 59	FPGA IO pin 31	TxD or serdata or other use
4	GND	GND	GND	GND	Ground

In more details:

- VCC-unreg is the voltage that you power your FPGA board with, typically +5V to +10V. On Pluto, 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 (Pluto/-3 only).

---

## 10 JTAG connection

### 10.1 JTAG on Pluto

JTAG is not available on Pluto.

### 10.2 JTAG on Pluto-IIx

The Pluto-IIx JTAG signals are accessible on pin headers next to the FPGA.

### 10.3 JTAG on Pluto-3

Pluto-3 has the regular Altera-style 10 pins header. A matching shrouded connector must be added to the board, like KNJN items [2450 or 2451](#), so that an Altera or compatible JTAG cable can easily be used.

## 11 Flashy boards

When a Flashy board is used in combination with a KNJN FPGA board, the system becomes a digital oscilloscope.

### 11.1 FlashyMini design

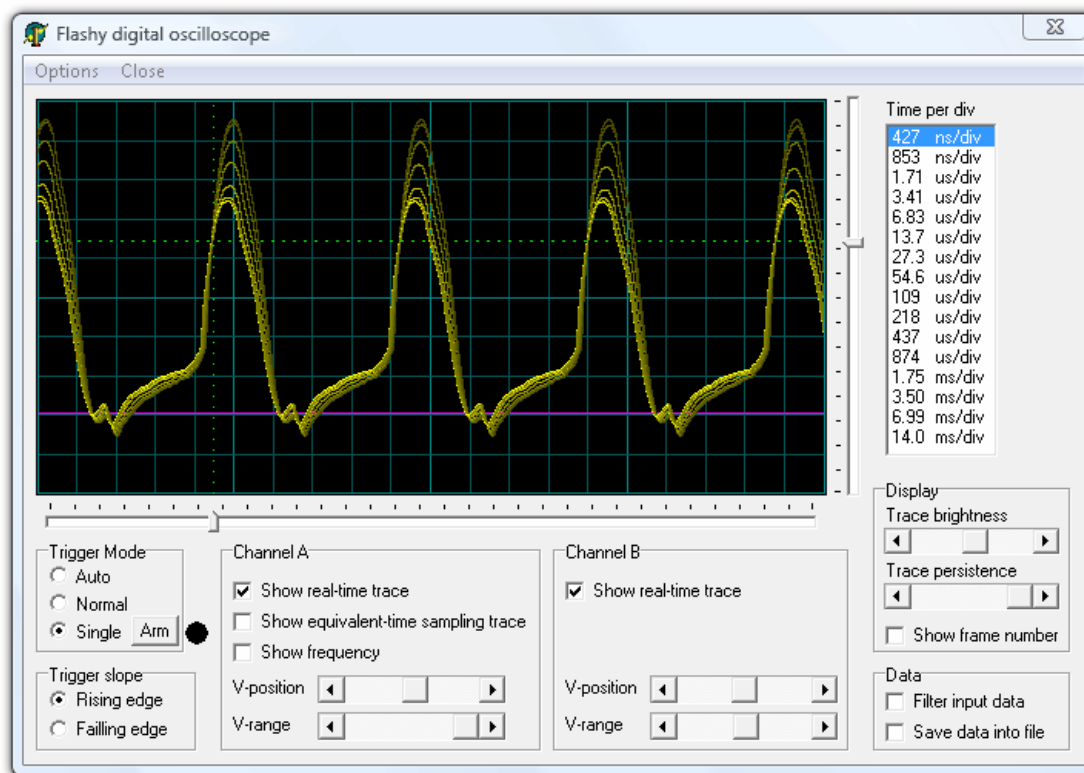
FlashyMini is provided with full source code (HDL + C). It shows how to get data from Flashy and can be used as a skeleton to develop your own acquisition system.

### 11.2 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 (see also the [Hands-on - A digital oscilloscope](#) page).
2. Configure the FPGA with the FlashyDemo bitfile.
3. Go to Menu → Tools → Flashy Oscilloscope (or press CTRL-F).



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

	Flash	Flashy	FlashyD	LCD (2)
Pluto	Limited (1)	Limited (1)	No	No
Pluto-IIx / HDMI	Yes	Yes	No	Yes
Pluto-3	Yes	Yes	Yes	Yes

(1) Pluto's FPGA cannot hold all the FlashyDemo functionality at once, so two FlashyDemo bitfiles are provided. Each covers a different set of features.

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

For more information, check KNJN's [Flashy acquisition board](#) page.



## 12 FPGA configuration through the serial port

### 12.1 Pluto-Ilx FPGA configuration

Unlike Pluto-Ilx 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);      // un-configure 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 un-configure Pluto (before sending the RBF), send a "break" condition (a high signal) for about 50ms.

The RBF file is a binary file which is usually compressible, in case you run out of memory in the microcontroller.

### 12.2 Pluto-3 FPGA configuration

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

On Linux, this script could be used.

```
#!/bin/bash
#
# pluto3configure :: send an rbf file to a Pluto-3 FPGA board
#

SERIAL=/dev/ttyUSB0

if [ ! -f "$1" ]
then
  echo "Usage: $(basename $0) filename.rbf" >&2
  exit 1
fi

(stty 115200 raw cs8 -cstopb -parenb -ixon -crtcts 0<&1 ; sendbreak; dd "if=$1" bs=1k) > $SERIAL
```

## 13 Quartus-II JTAG indirect mode (Pluto-3)

### 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

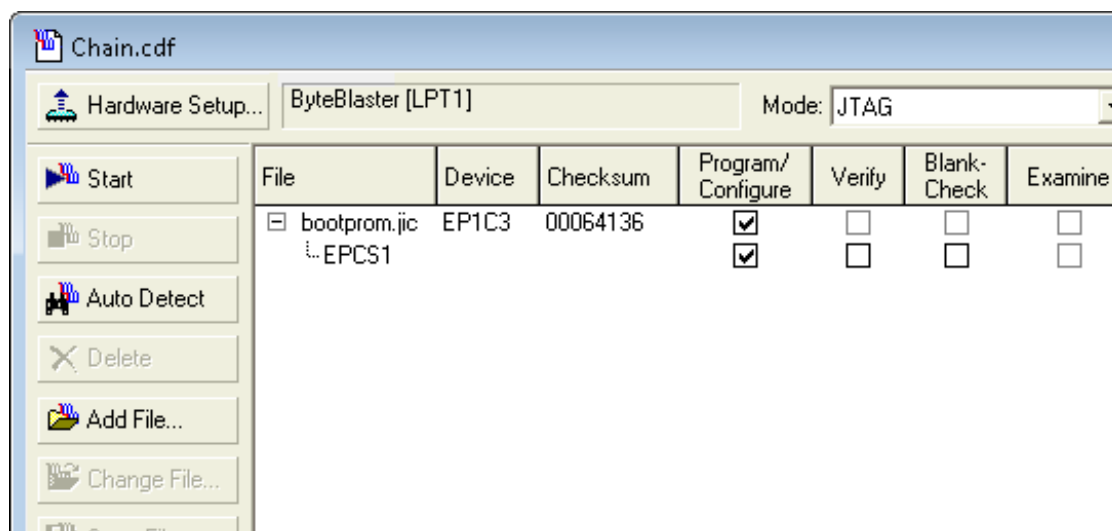
### 13.2 Create a "JTAG indirect configuration" file

1. In Quartus-II, go to File → Convert programming file
2. In the "Output programming file" panel, select "JTAG Indirect Configuration File", EPCS4 for Pluto-3, and a filename for a ".jic" file.
3. In "Input file to convert", select the 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 (EP2C5) and the boot-PROM (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 (TXDI interface)

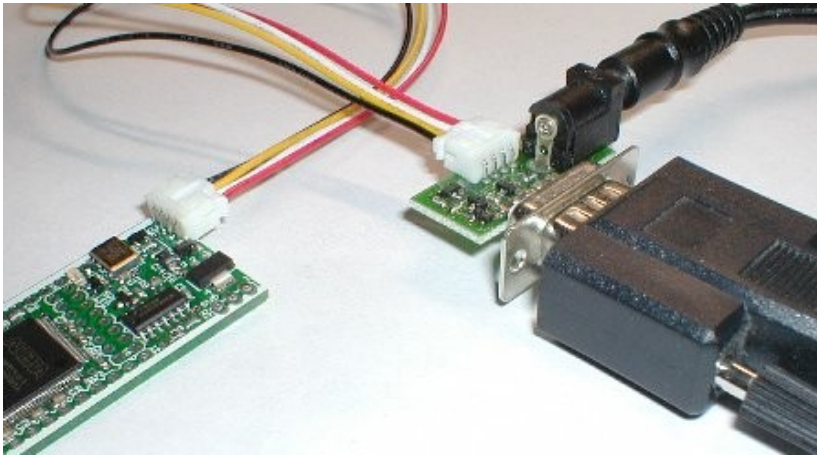
### 15.1 Serial connection

The Pluto boards connects to the PC's serial 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 RS-232 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 serial 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 serial data to the FPGA, but the FPGA cannot send data to the PC) so cannot be used to program the Pluto-II/-IIx/-3 boot-PROM.

## 16 Sample C code for serial 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:

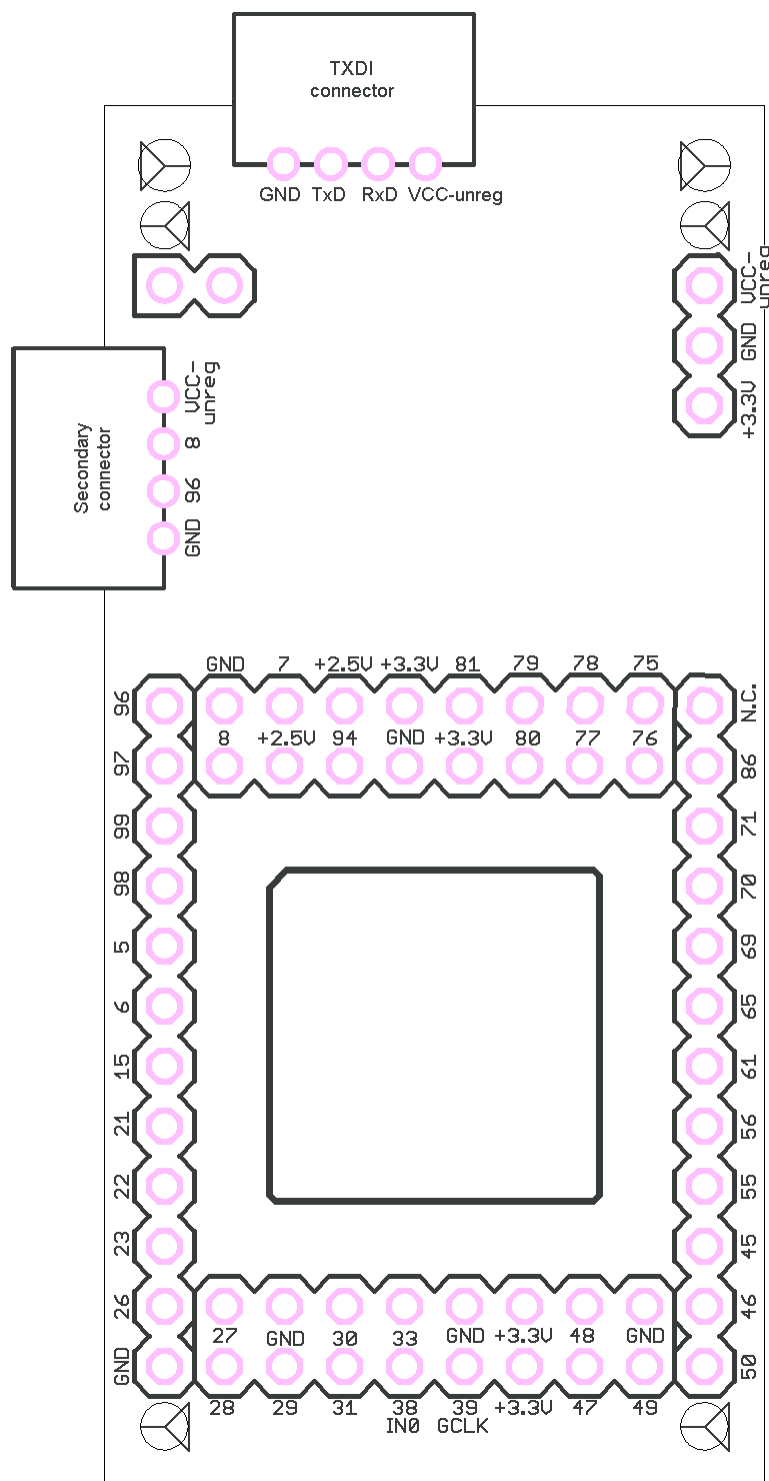
1. Make sure the board is powered with a correct voltage. You need 5V minimum at the input of your Pluto board. If you use a TXDI with an integrated 5V regulator, it requires at least 7V to be able to provide 5V to Pluto.
2. Try a known good bitfile file, like LEDblink from the startup-kit.
3. Make sure 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 serial port 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 serial connection works unreliably. With laptops, that's usually due to low-level RS-232 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").

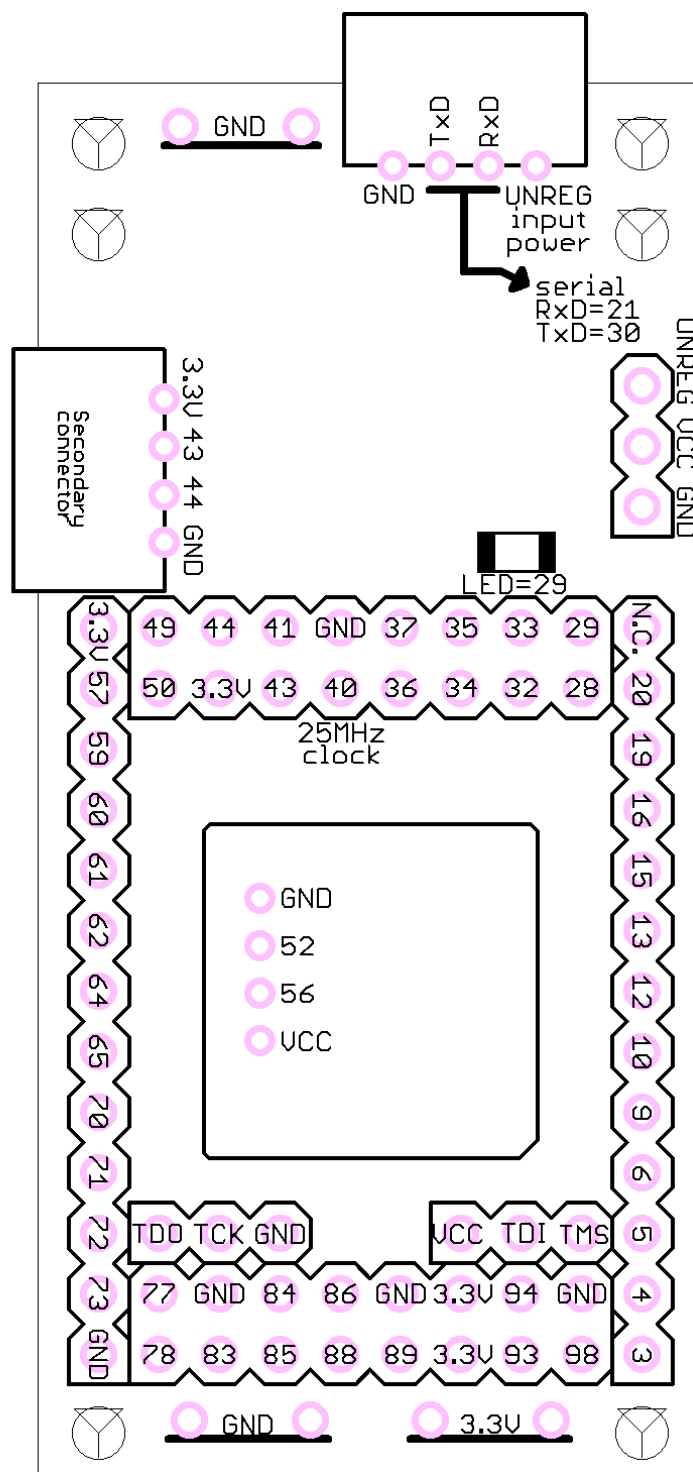
## 18 Board connectors and headers, with IO pin assignments

### 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.

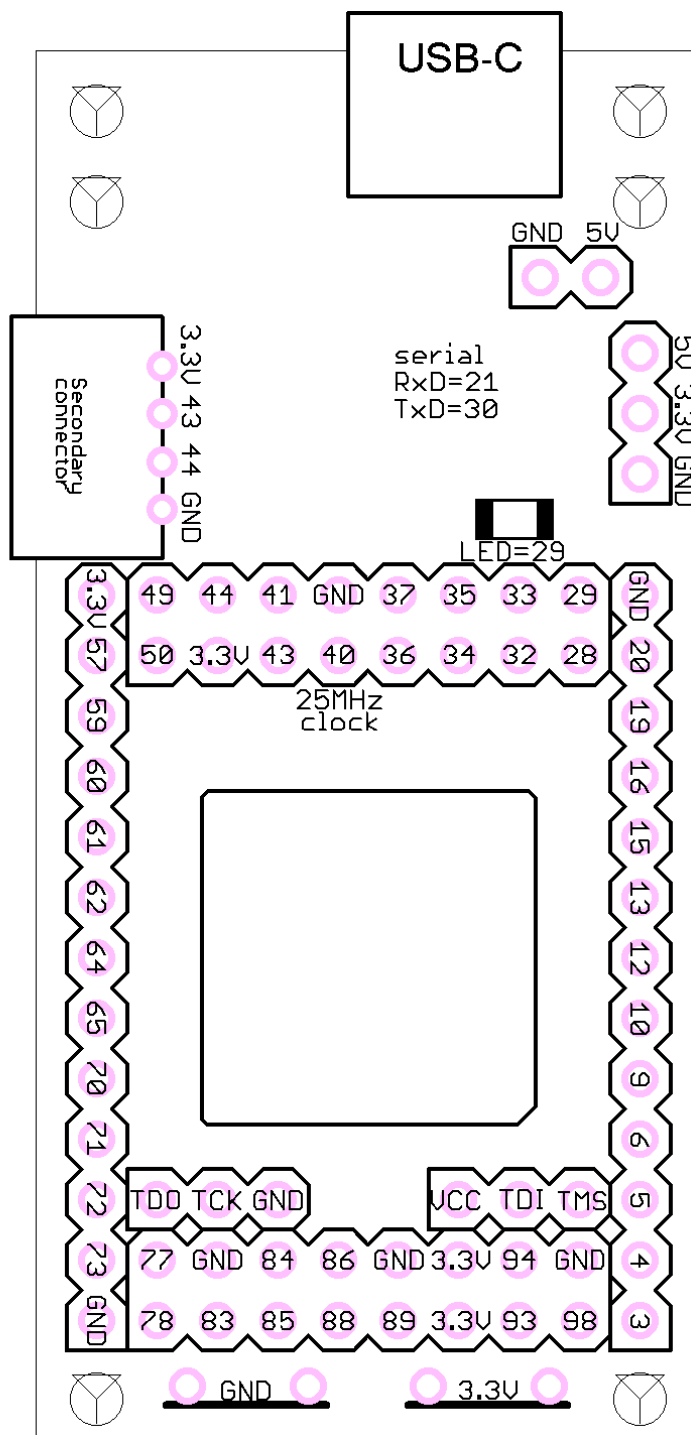
## 18.2 Pluto-Ilx



All IOs use 3.3V powered banks that can use different IO standards like LVTTL3.3 or LVCMOS3.3. Pluto-IIx IOs are not directly 5V tolerant. Check the [Spartan-3A user guide](#) for more details.

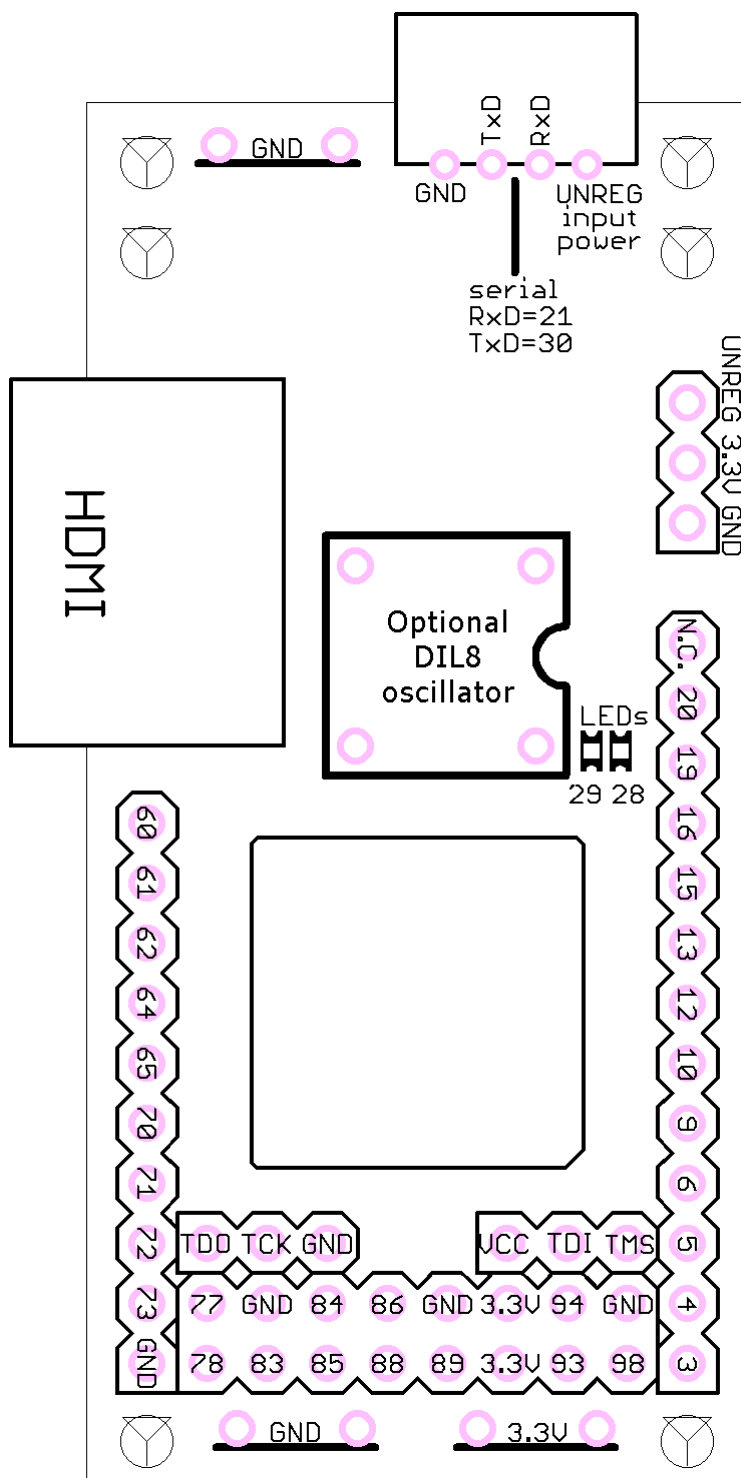


## 18.3 Pluto-Ilx USB-C



All IOs use 3.3V powered banks that can use different IO standards like LVTTTL3.3 or LVCMOS3.3. Pluto-Ilx IOs are not directly 5V tolerant. Check the [Spartan-3A user guide](#) for more details.

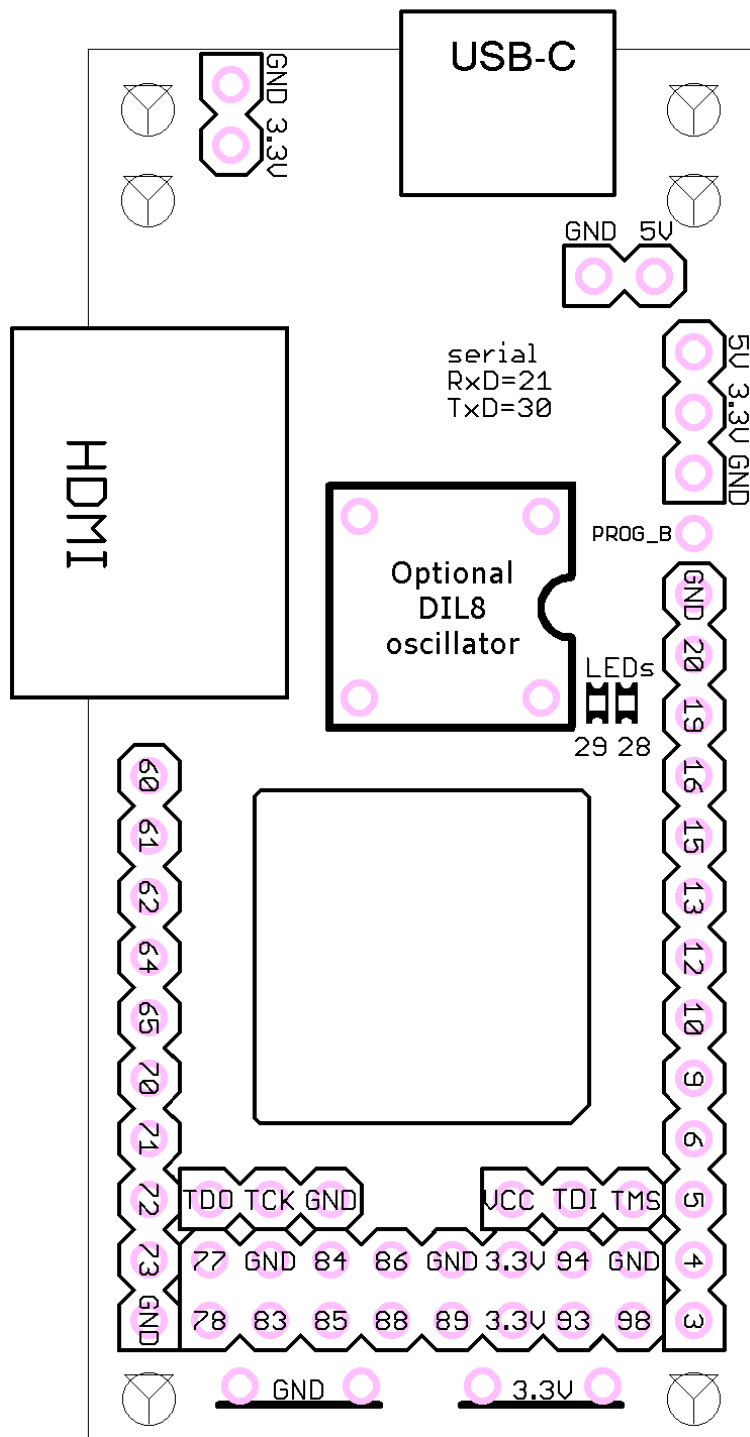
## 18.4 Pluto-IIx HDMI



All IOs use 3.3V powered banks that can use different IO standards like LVTTTL3.3 or LVCMOS3.3. Pluto-IIx IOs are not directly 5V tolerant. Check the [Spartan-3A user guide](#) for more details.

An optional DIL8 oscillator can be added in the middle of the board (feeds FPGA clock pin 41).

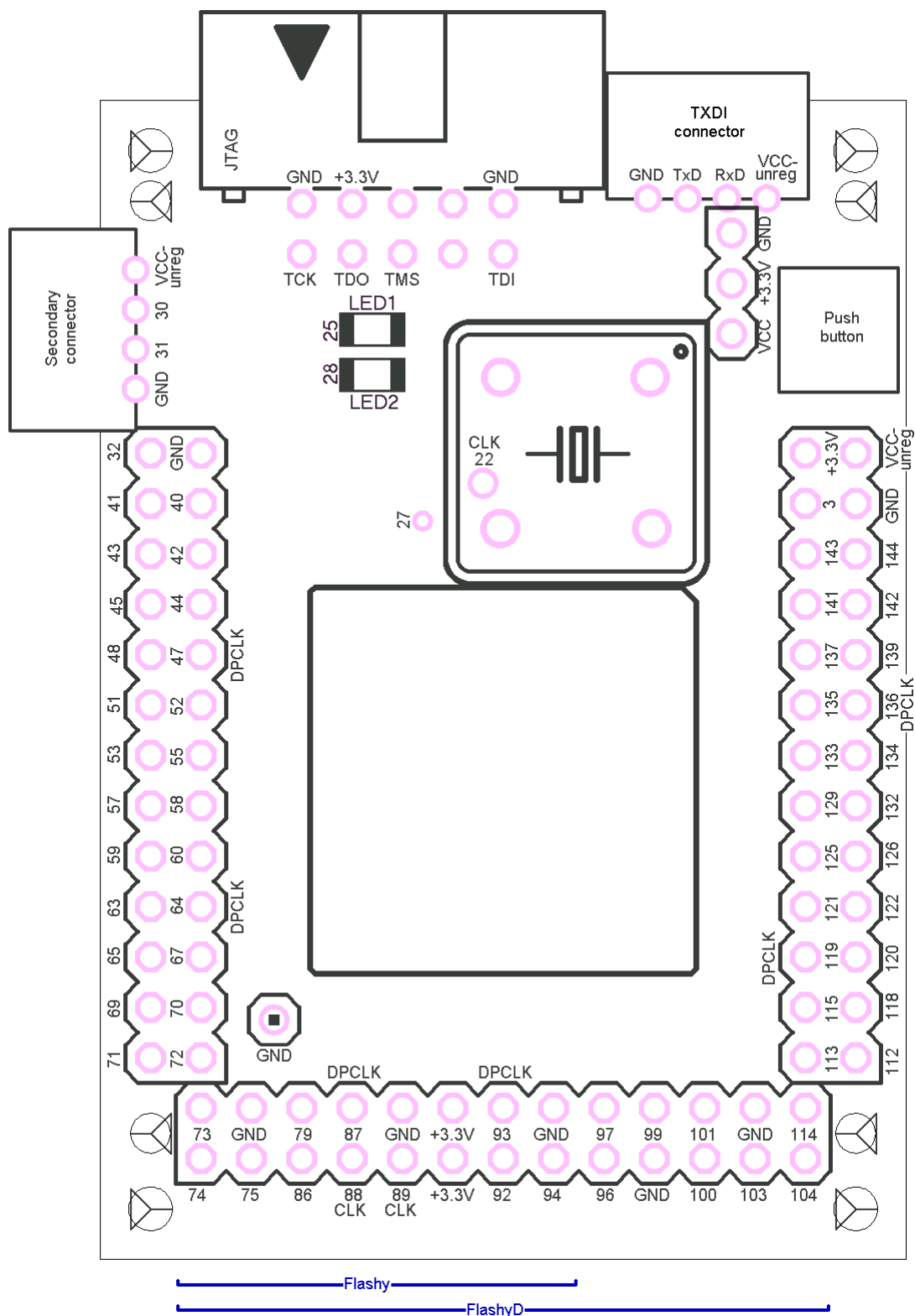
## 18.5 Pluto-IIx HDMI USB-C



All IOs use 3.3V powered banks that can use different IO standards like LVTTTL3.3 or LVCMOS3.3. Pluto-IIx IOs are not directly 5V tolerant. Check the [Spartan-3A user guide](#) for more details.

An optional DIL8 oscillator can be added in the middle of the board (feeds FPGA clock pin 41).

## 18.6 Pluto-3



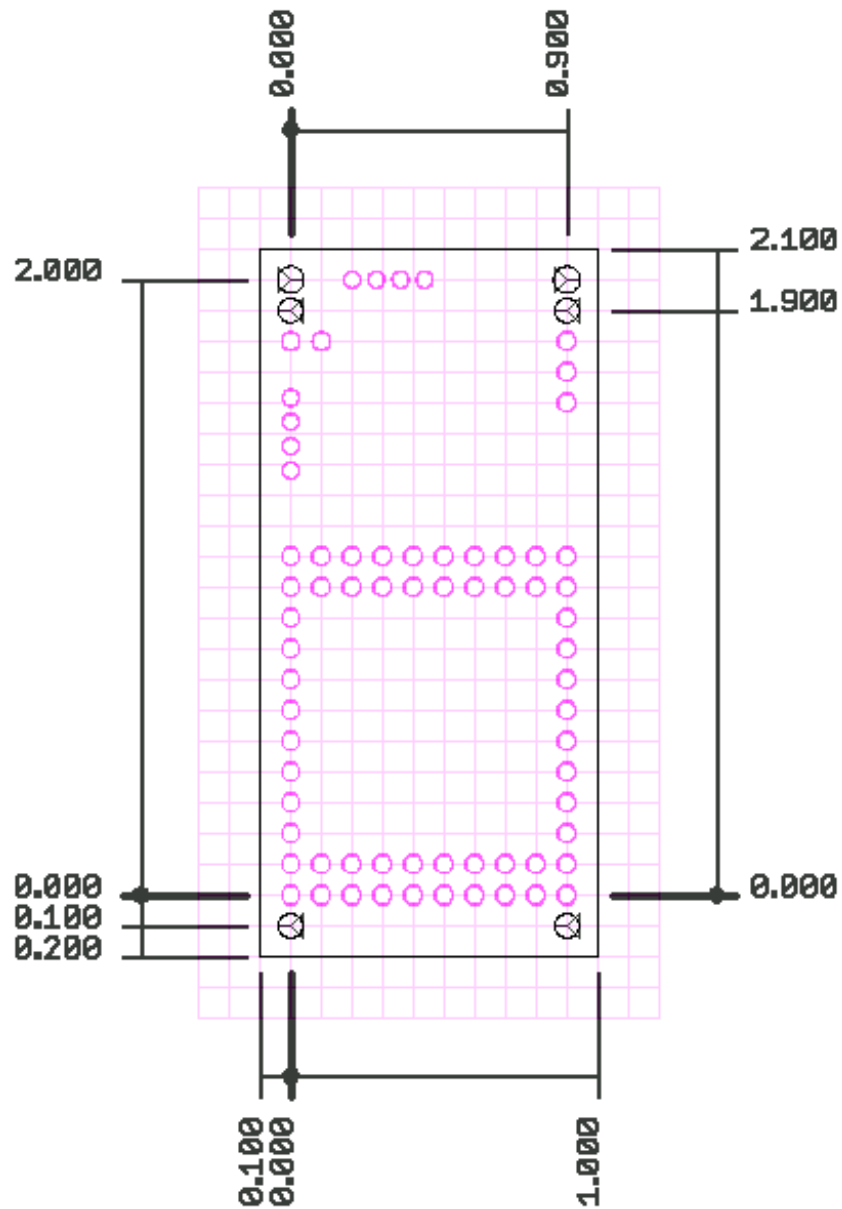
- Many pins can be used as clock: CLK6 (pin 89), CLK7 (pin 88), DPCLK2 (pin 47), DPCLK4 (pin 64), DPCLK6 (pin 87), DPCLK7 (pin 93), DPCLK8 (pin 119), DPCLK10 (pin 136). Also CLK3 (pin 22) is available on a pad.
- All IOs use 3.3V powered banks that can use different IO standards like LVTTTL3.3 or LVCMOS3.3. They are not directly 5V tolerant. Check the [Altera Cyclone-II device handbook](#) for more details.

## 19 Mechanical drawings

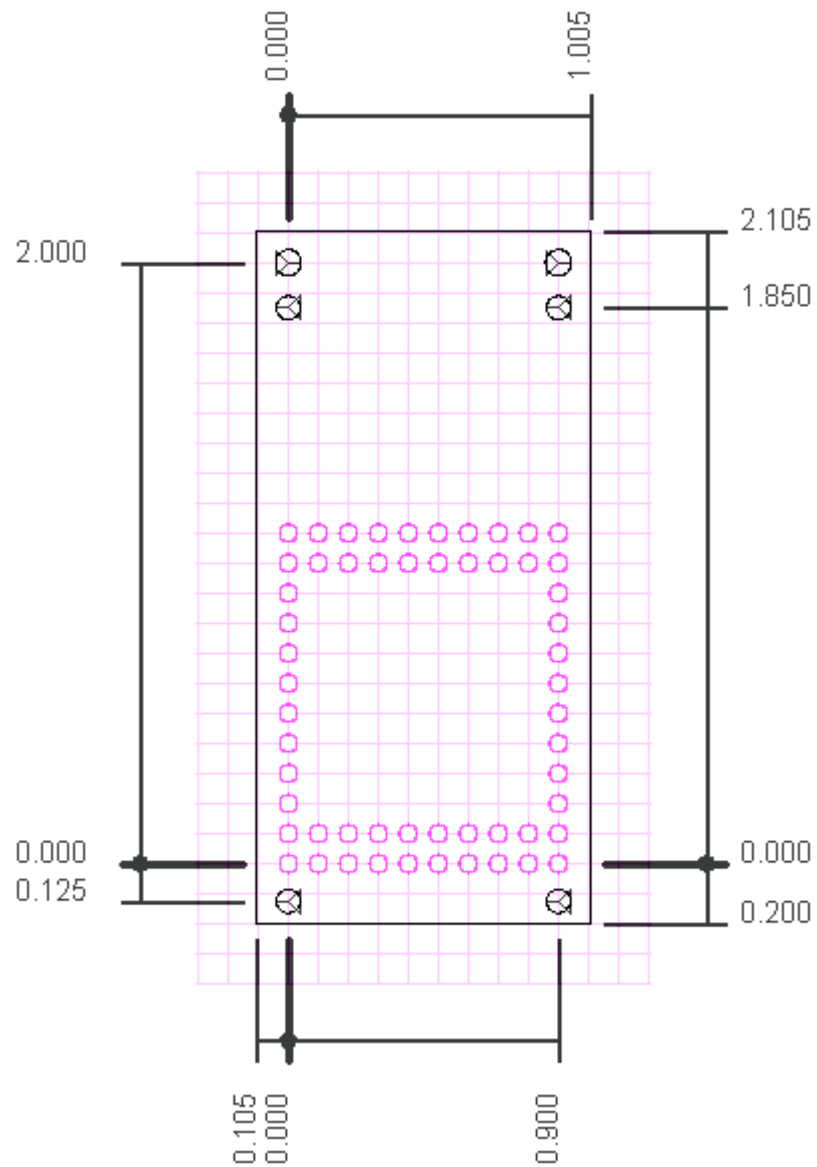
All dimensions are given in inches (1" = 25.4mm).

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

## 19.1 Pluto



## 19.2 Pluto-llx (all versions)



## 19.3 Pluto-3

