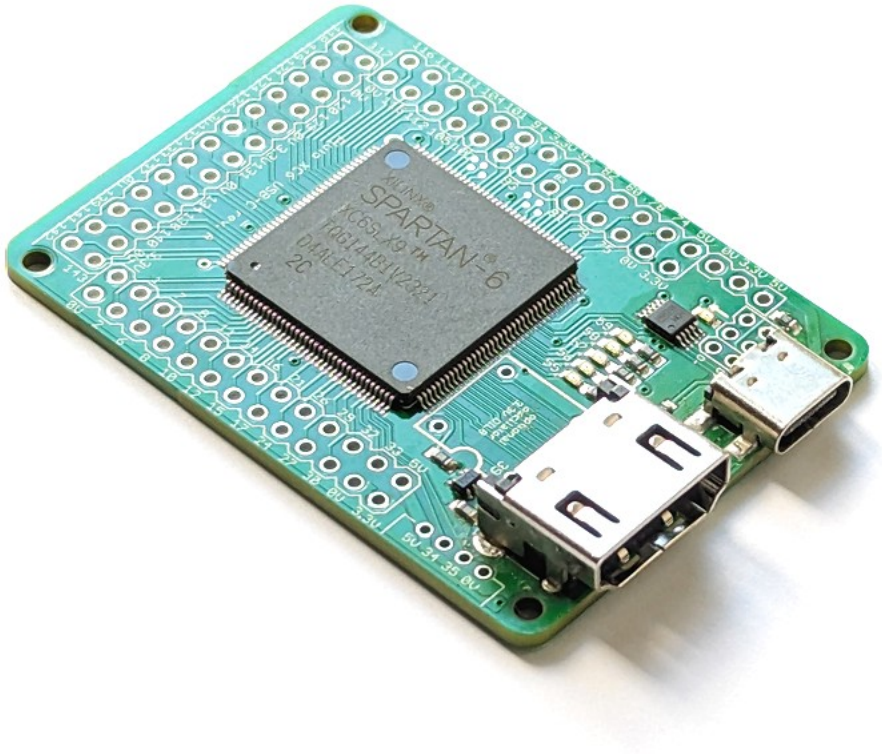
FPGA serial development boards

https://www.knjn.com/

This document applies to the following boards:

- Pluto rev. F
- Pluto-IIx
- Pluto-XC6
- Pluto-3 rev. B and C



Document last revision on **March 31, 2024**

Table of Contents

1 Welcome

1.1 This guide

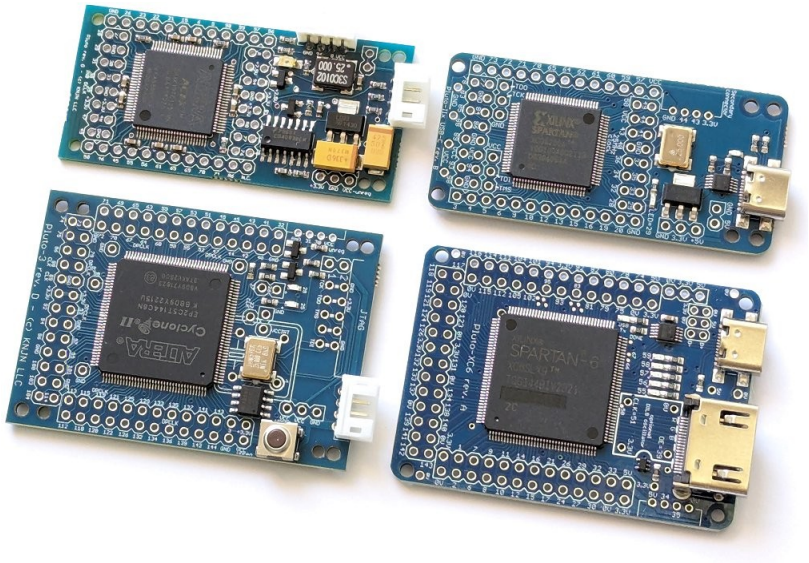
Welcome.

This document is partitioned in short chapters.

1.2 The Pluto boards

Although FPGA boards can be intimidating, the Pluto boards are easy to use as they are controlled through a serial port, either USB-C or TXDI.

The serial port is used for FPGA configuration, and after configuration it allows PC ↔ FPGA communication.



The Pluto boards with USB-C port use an CH340 interface chip and appear natively to the PC as a COM port, while the Pluto boards with TXDI require a small adapter board.

	Pluto	Pluto-IIx	Pluto-XC6 HDMI	Pluto-3
FPGA	EP1K10	XC3S50A or XC3S200A	XC6SLX9	EP2C5
Datasheet (PDF)	ACEX 1K	Spartan-3A	Spartan-6	Cyclone II
Serial interface	TXDI	USB-C	USB-C	TXDI
Logic cells	576	1584 or 4032	9152	4608
IO pins	41 (1)	48	62	65
PLL/DLL	-	DLL	PLL/DLL	PLL
External clocks	up to 2	up to 4	Up to 8	up to 4
Boot-PROM (2)	-	4Mbit	8Mbit	4Mbit
On-board oscillator	25MHz	25MHz	25MHz	25MHz
DIL8 oscillator header	-	-	Yes	Yes
JTAG connection (3)	-	Yes	Yes	Yes
LED(s) (4)	1	1	5	2
ADC board ready	Flashy / Widy	Flashy / Widy	FlashyD / WidyD	FlashyD / WidyD
Dimensions	58 x 28 mm	58 x 28 mm	58 x 41 mm	58 x 41 mm

(1) IOs on Pluto are 5V tolerant.
(2) Minimum boot-PROM size shown here. Actual product may use a higher capacity boot-PROM.
(3) See chapter 5.5.
(4) Pluto boards with a USB-C port have an additional activity LED.

2 Software tools

2.1 Essential downloads

First download the “startup-kit” from your order page. It includes utilities and example source code.

Then download the FPGA vendor software from one of the links below. The software is free but may require creating a free license.

Board	Software
Pluto	Quartus II Web Edition 9.0 SP2 (1.3GB)
Pluto-IIx	ISE WebPACK 14.7 (see note below)
Pluto-XC6	ISE WebPACK 14.7 (see note below)
Pluto-3	Quartus II Web Edition 13.0 SP1 (4.4GB)

Note that ISE WebPack 14.7 comes in two versions:

- ISE Design Suite for Windows 10 (virtual machine)
  Use this on Windows 10 or Windows 11.
- ISE Design Suite (native Win32/64 or Linux)
  Use this on Windows 10 or Linux.

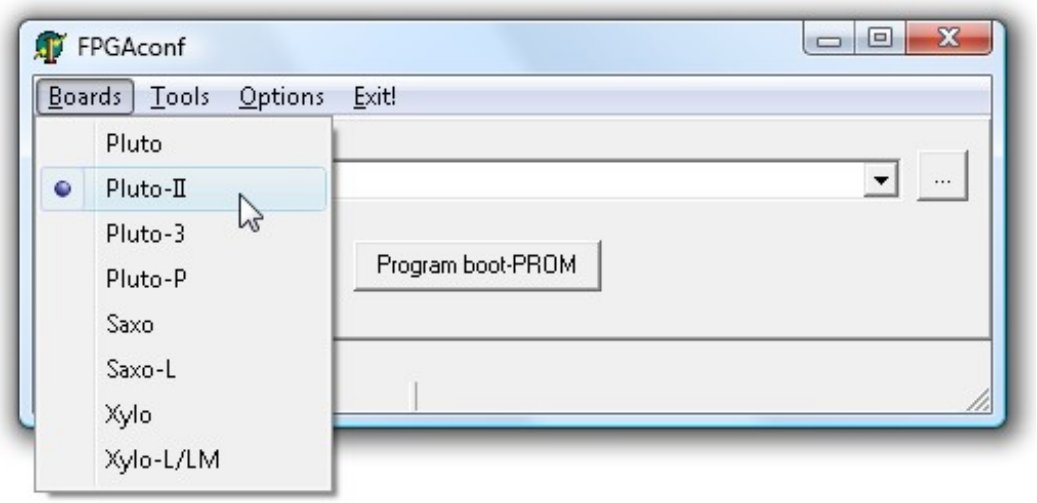
A C/C++ compiler is optional but you'll need one for many projects. Usable compilers include

- Microsoft Visual Studio Community
- Digital Mars C++
- Jacob Navia's lcc-win32

2.2 FPGAconf

FPGAconf is a utility provided in the startup-kit. Its main use is configuring the FPGA.

First make sure that your board is selected. For example, we select a Pluto-II below.



Then

1. Choose an FPGA bitfile (i.e. click on the browse button “...”).
2. Click “Configure FPGA”.

For your convenience, sample FPGA bitfiles are provided in the startup-kit. In particular, try “LEDblink” and “LEDglow”.

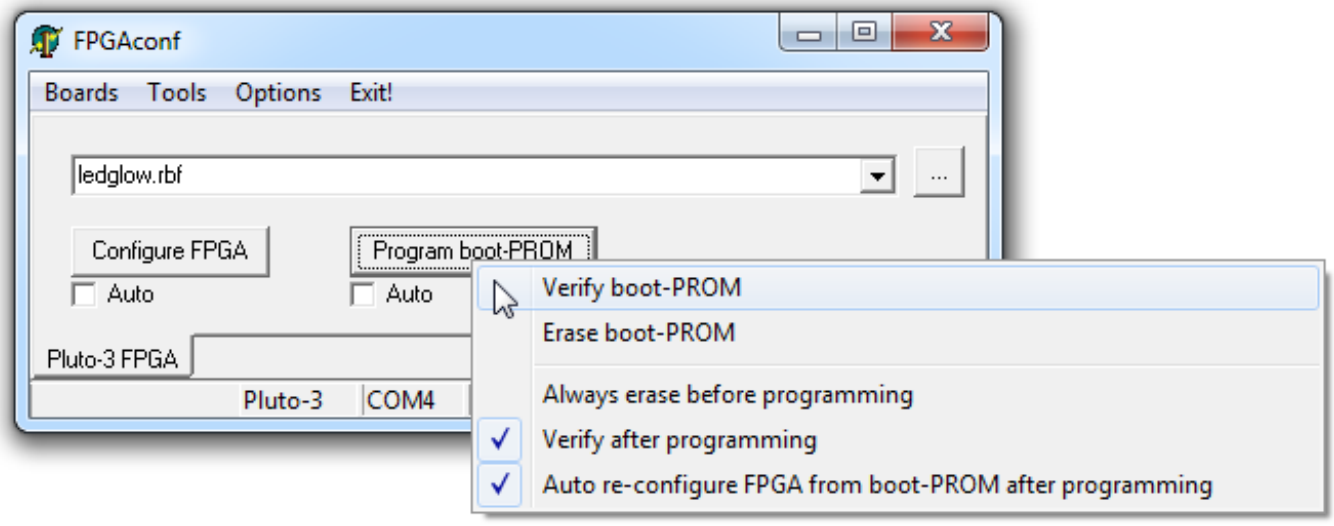
3 FPGA boot-PROM

3.1 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” for allowing manual FPGA configuration.

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.



3.2 Boot-PROM requirements

If your board has a USB-C port, you are all set.

If your board has a TXDI interface, note that 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.

3.3 Boot-PROM on-demand FPGA configuration

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

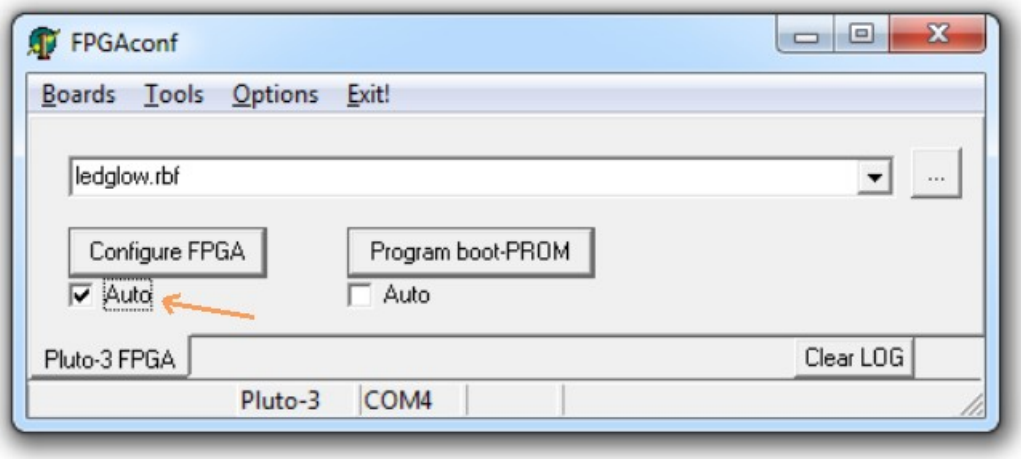
Here's a summary of all the boot-PROM features.

Boot-PROM	Pluto	Pluto-IIx	Pluto-XC6	Pluto-3
Configures FPGA at power-up	N/A	Yes	Yes	Yes
Can be programmed through the serial port	N/A	Yes	Yes	Yes
Can be programmed through JTAG	N/A	Yes	Yes	Yes
Can configure the FPGA on-demand (after power-up)	N/A	Yes	Yes	Yes

4 FPGAconf extras

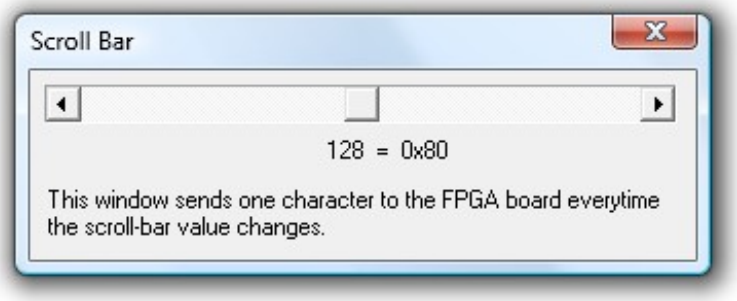
4.1 Auto configuration mode

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,.



4.2 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.

4.3 Terminal

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

5 FPGA connections

5.1 FPGA pins

The main FPGA signals are:

Pin name	Pluto	Pluto-IIx	Pluto-XC6	Pluto-3	Direction	Comment
CLK0	91	40	127	17	FPGA input	25MHz on-board SMD oscillator
CLK1			51	18	FPGA input	Optional DIL8 oscillator
CLK2			88		FPGA input	Optional SMD oscillator
LED1	7	29	59	28	FPGA output	Red LED (active high)
LED2			58	25	FPGA output	Red LED (active high)
PB				9	FPGA input	Push-button (active low)
RxD	77	21	61	21	FPGA input	FPGA receives from PC
TxD	78	30	62	24	FPGA output	FPGA transmits to PC

The RxD and TxD pins are used as an asynchronous serial port and allow communication with the PC. Many other IO signals are available on unpopulated headers, see the drawings on chapter 11. For Pluto-XC6 HDMI, check the demo project in its startup-kit.

5.2 Boot-PROM connection

The boot-PROM is an SPI flash W25Q or equivalent that is connected to the FPGA.

SPI flash pin	Pluto-IIx FPGA pin	Pluto-XC6 FPGA pin	Pluto-3 FPGA pin
Clock	53	70	15
Data In	46	64	1
Data Out	51	65	14
nCS	27	38	2
nHOLD	31	69	8

5.3 Secondary connector

This 4-pins connector is located on the side of the board. It can be easily soldered (available as KNJN item#1804 or from DigiKey). It has 2 power pins and 2 IOs.

Pin	Pluto	Pluto-IIx	Pluto-XC6	Pluto-3	Comment
1	VCC-unreg (1)	3.3V	5V	VCC-unreg (1)	Power
2	FPGA IO pin 8	FPGA IO pin 43	FPGA IO pin 34	FPGA IO pin 30	RxD or serclk or other use
3	FPGA IO pin 96	FPGA IO pin 44	FPGA IO pin 35	FPGA IO pin 31	TxD or serdata or other use
4	GND	GND	GND	GND	Ground

(1) VCC-unreg is the voltage that you power your FPGA board with, typically +5V to +10V.

5.4 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 Pluto board).

5.5 JTAG connection

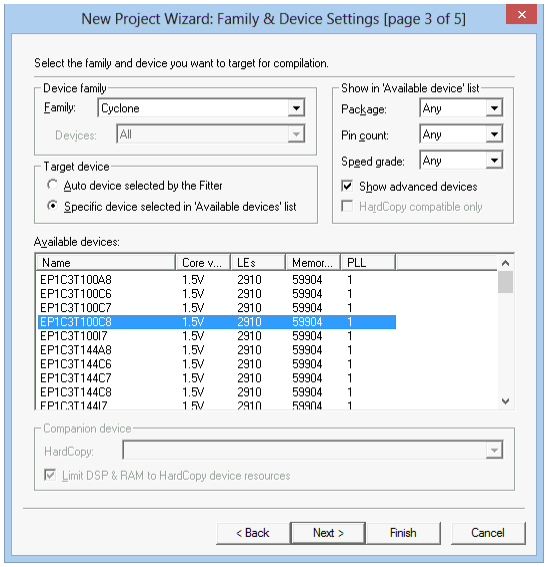
Board	JTAG
Pluto	JTAG is not available.
Pluto-IIx	The JTAG signals are accessible on pin headers next to the FPGA.
Pluto-XC6	The JTAG signals are accessible on pin headers on the bottom side of the board (below the FPGA).
Pluto-3	Pluto-3 has a full-size Altera-style 10 pins header. A matching shrouded 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.

6 FPGA project using Quartus-II (Pluto/Pluto-3)

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

6.1 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.

6.2 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 5.1). 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".

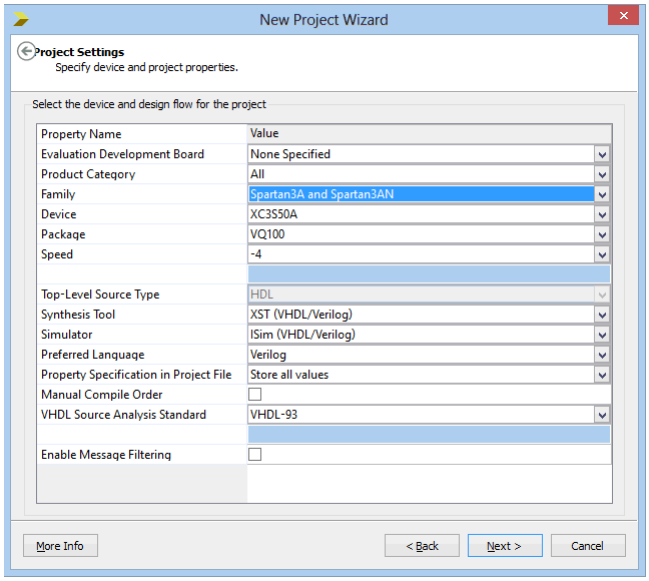
Option 2.a makes sure RBF files are generated (used for serial FPGA configuration). Otherwise only SOF files are generated (used for JTAG).

Option 2.b 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.

7 FPGA projects with ISE (Pluto-IIx/Pluto-XC6)

The Pluto-IIx and Pluto-XC6 boards are configured from BIT files generated by Xilinx's ISE software.

7.1 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. For Pluto-XC6, select the Spartan-6 family, then the XC6SLX9 device in TQG144 package. For Pluto-IIx, select the “Spartan3A and Spartan3AN” family and the device on your board (XC3S50A or 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.

7.2 A simple start

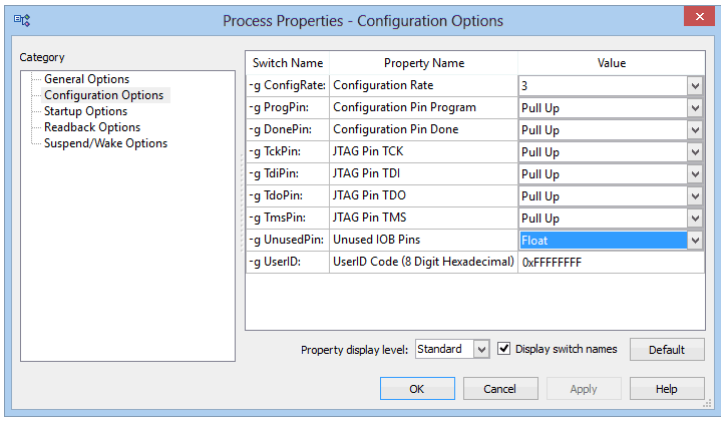
Here’s a simple Verilog file:

```
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 in your project.
Now add a UCF file to the project. For example, use this for a Pluto-IIx

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

Finally right-click on “Generate Programming File” and choose “Process Properties”. Click on “Enable BitStream Compression” and allow “Unused IOB Pins” to “Float” or “Pull-up”.



8 Flashy boards

With an optional Flashy board, the system becomes a digital oscilloscope.

8.1 FlashyMini design

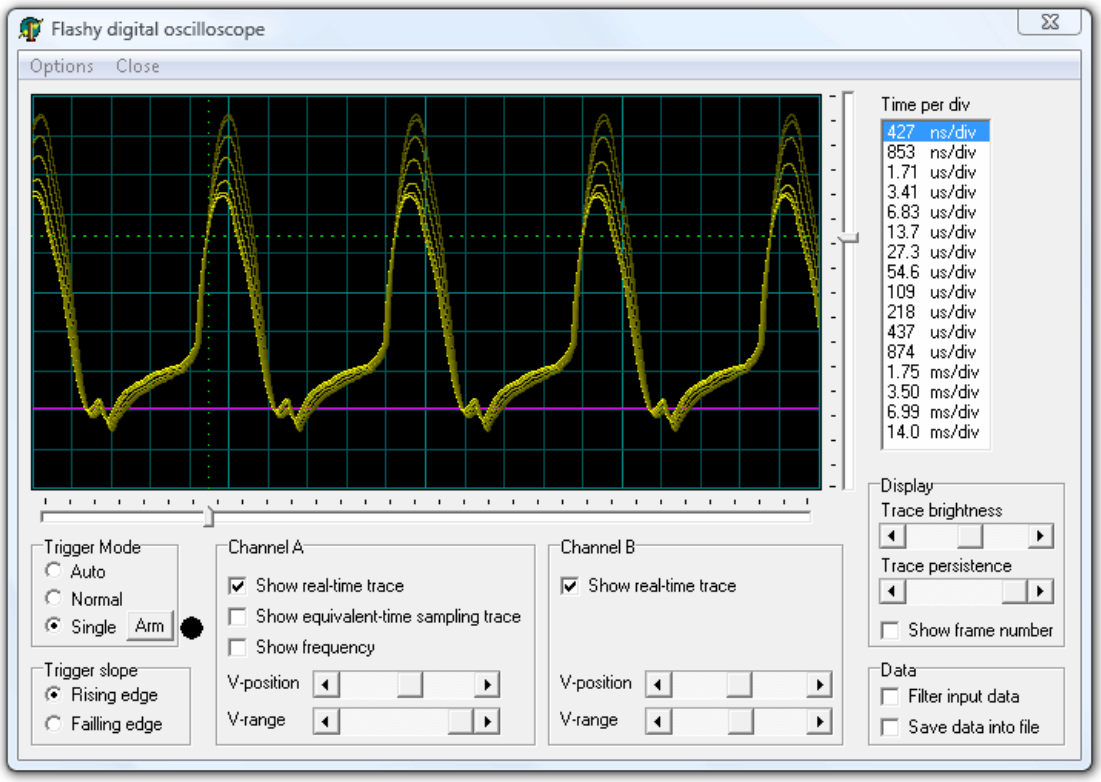
FlashyMini is provided with full source code (HDL + C). It shows how to get data from Flashy and can be used as a skeleton to develop your own acquisition system.

8.2 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.
2. Configure the FPGA with the FlashyDemo bitfile.
3. Go to Menu → Tools → Flashy Oscilloscope (or press CTRL-F).



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

	Flash	**Flashy**	**FlashD**	**FlashyD**	**LCD (2)**
Pluto	Limited (1)	Limited (1)	No	No	No
Pluto-IIx	Yes	Yes	No	No	Yes
Pluto-XC6	Yes	Yes	Yes	Yes	Yes
Pluto-3	Yes	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 the Flashy acquisition board page.

9 FPGA configuration

In case you don’t want to use FPGAconf.

9.1 Pluto/-IIx/-XC6 FPGA configuration

Set the baud speed at 115200 bauds for TXDI and 3000000 bauds for USB-C, and run the following C pseudo-code:

For each byte of the bitfile, do:

```
    for(j=0; j<8; j++)   serial.write(~(bufbyte >> j & 1 ^ 1));  // for Altera (LSB first)
or
    for(j=0; j<8; j++)   serial.write(~(bufbyte >> (j^7) & 1));  // for Xilinx (MSB first)
```

A complete example for Altera could be:

```
       FILE *fpIn = fopen("LEDblink.rbf", "rb");
       char buf[0x100000];
       int len = fread(buf, 1, sizeof(buf), fpIn);
       fclose(fpIn);

       OpenCom(115200);
       SetCommBreak(hCom);  Sleep(50);  ClearCommBreak(hCom);  // un-configure FPGA

       for(int i=0; i<len; i++)
        for(int j=0; j<8; j++)
         WriteComChar(~(buf[i] >> j & 1 ^ 1));

       CloseCom();
```

See also the chapter 10 for some extra source code.

9.2 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, a script can be used:

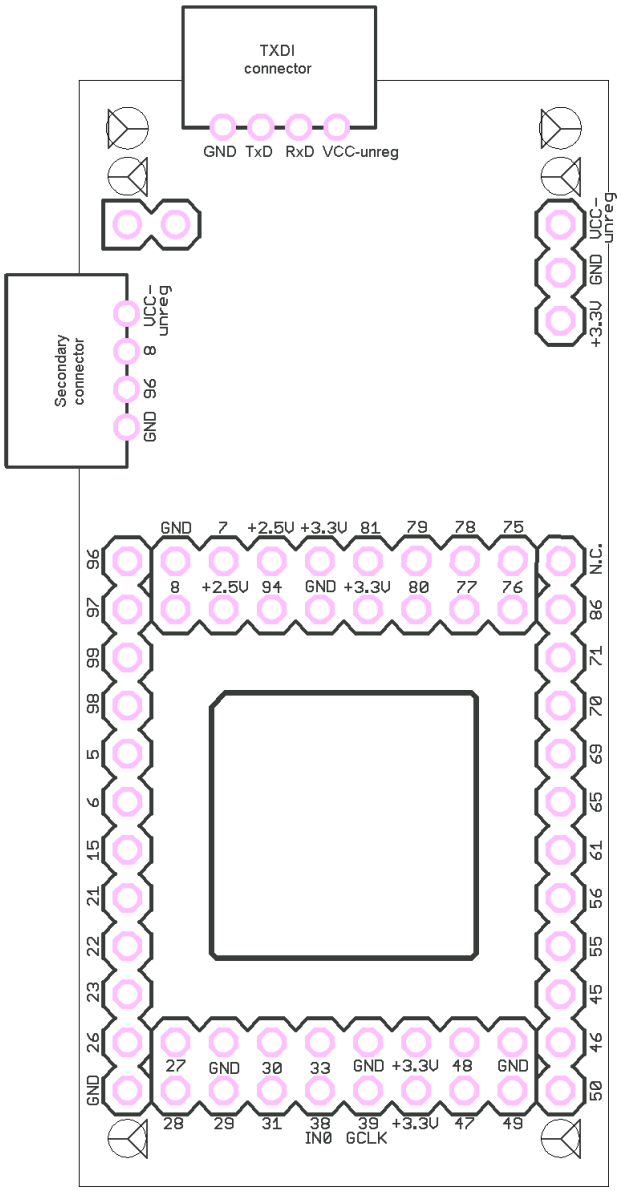
```
#!/bin/bash
#
# pluto3configure :: send an rbf file to a Pluto-3 FPGA board
#

SERIAL=/dev/ttyUSB0

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

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

10 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();
}
```

11 Board connectors and headers, with IO pin assignments

11.1 Pluto

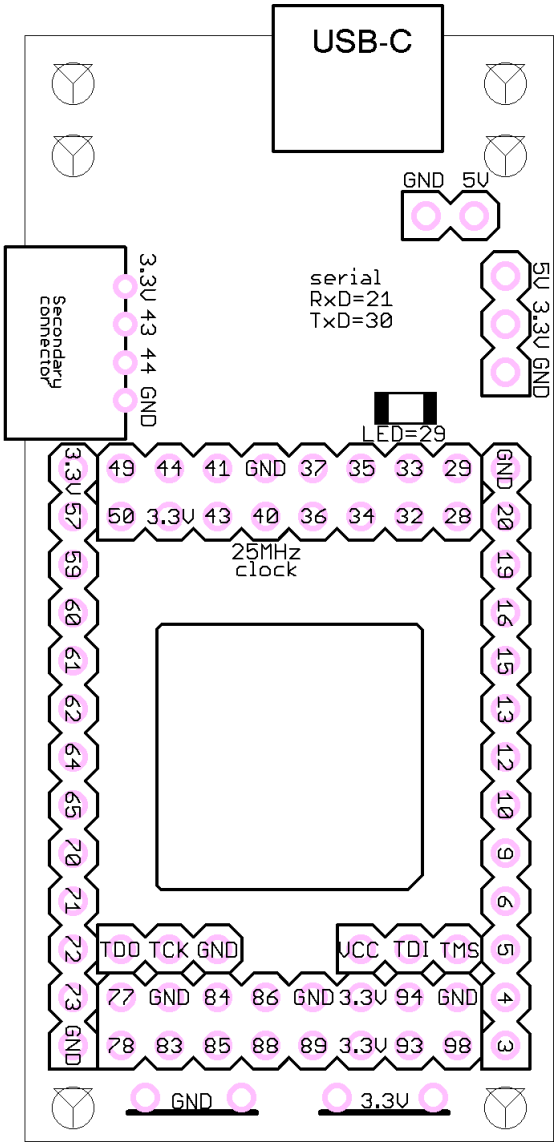


All IOs use 3.3V powered banks and are 5V input tolerant.

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.

Check the ACEX family datasheet for more details.

11.2 Pluto-IIx



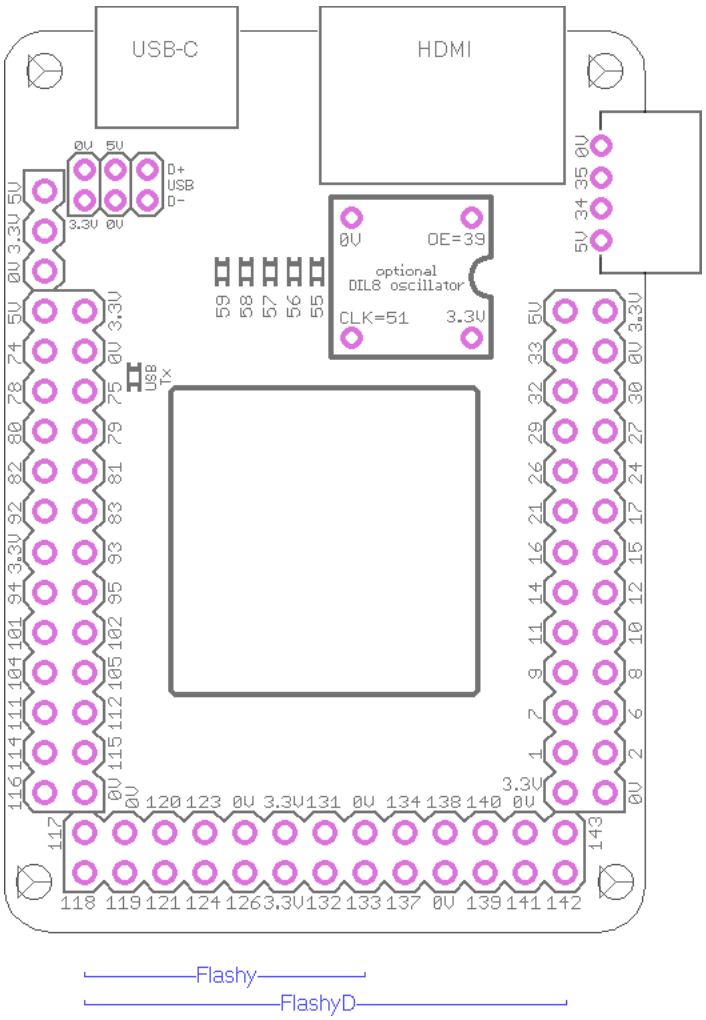
All IOs use 3.3V powered banks and are not 5V tolerant.

Many pins can be used as clock: LHCLK pins 9, 10, 12, 13, 15, 16, RHCLK 59~62, 64, 65, and GCLK 83~86, 88, 89.

The main oscillator (pin 40) is always enabled.

Check the Spartan-3A user guide for more details.

11.3 Pluto-XC6



All IOs use 3.3V powered banks and are not 5V tolerant.

Many pins can be used as clock: pins 14~17, 21, 24, 51 (optional DIL8 oscillator), 88 (optional SMD oscillator at the bottom of the board), 92~95.

The main oscillator (pin 127) is always enabled by default but it can be disabled under pin 5 control if a wire jumper or a small value resistor is added at the bottom of the board (see mark “osc OE”).

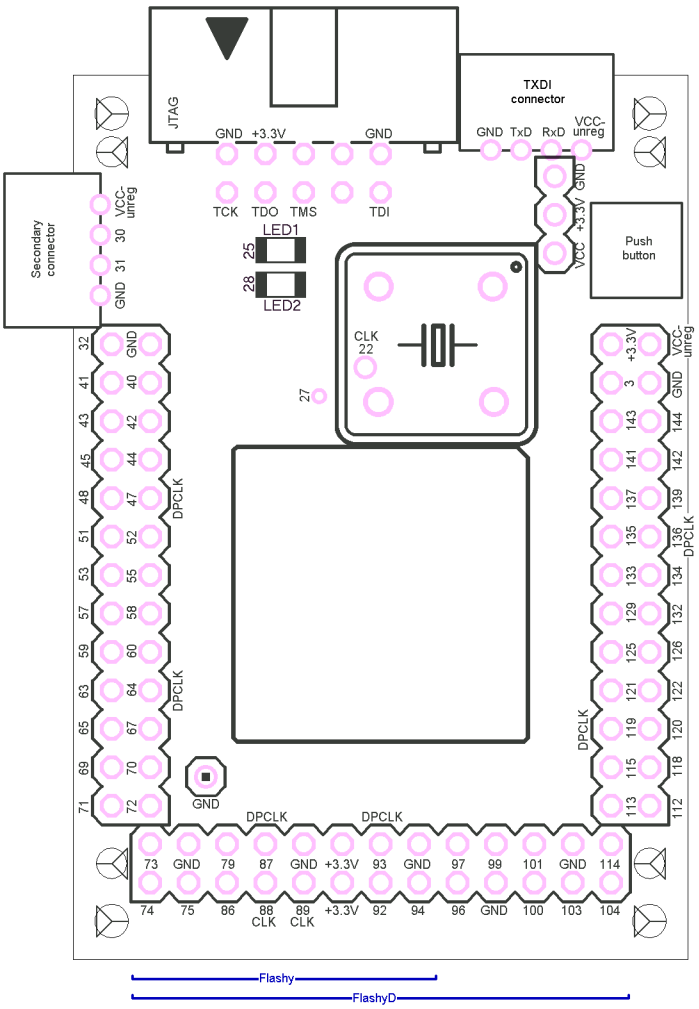
The USB signals are available on a 2mm pitch header below the USB-C connector.

An extra secondary connector header with 3.3V power is available at the bottom of the board below the FPGA.

A few extra IOs are available on SMD test pads around the FPGA.

Check the Spartan-6 family overview and the Spartan-6 datasheet for information on the FPGA.

11.4 Pluto-3



All IOs use 3.3V powered banks and are not 5V tolerant.

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.

The main oscillator (pin 17) is always enabled.

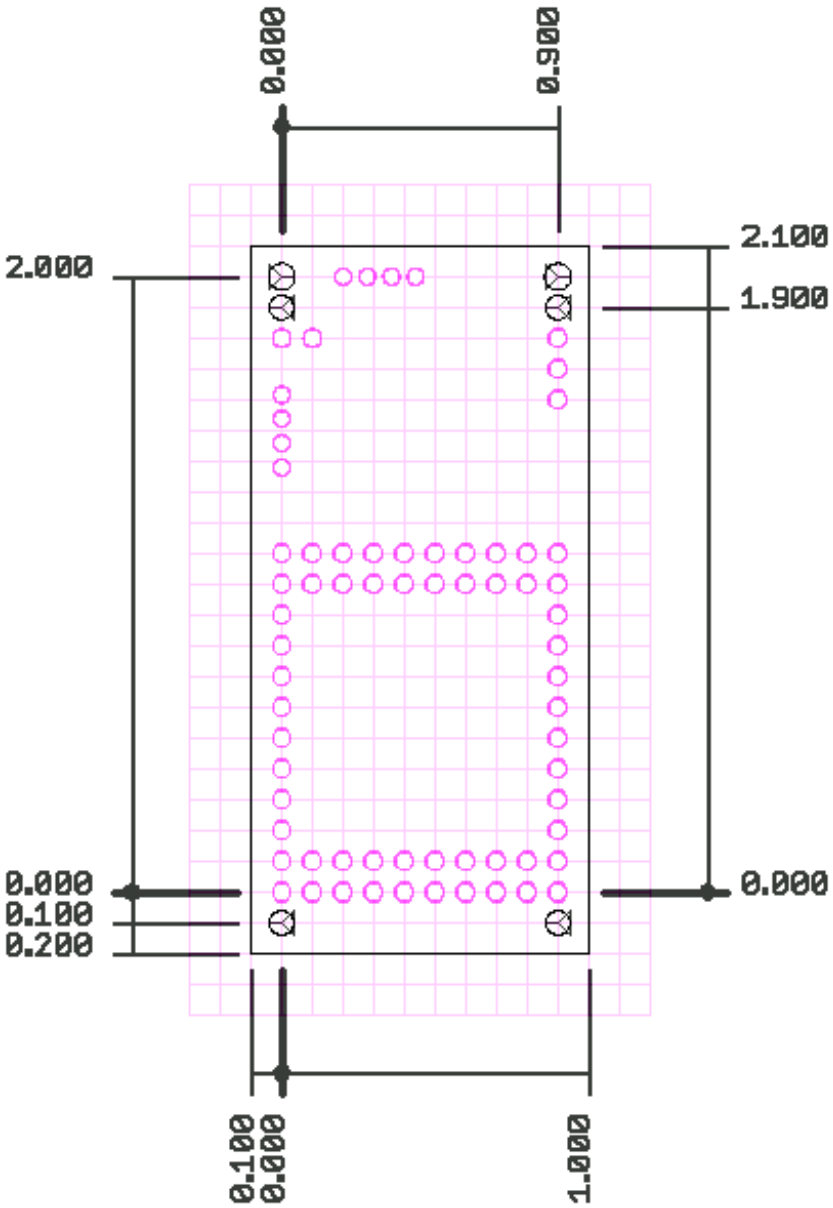
Check the Cyclone-II device handbook for more details.

12 Mechanical drawings

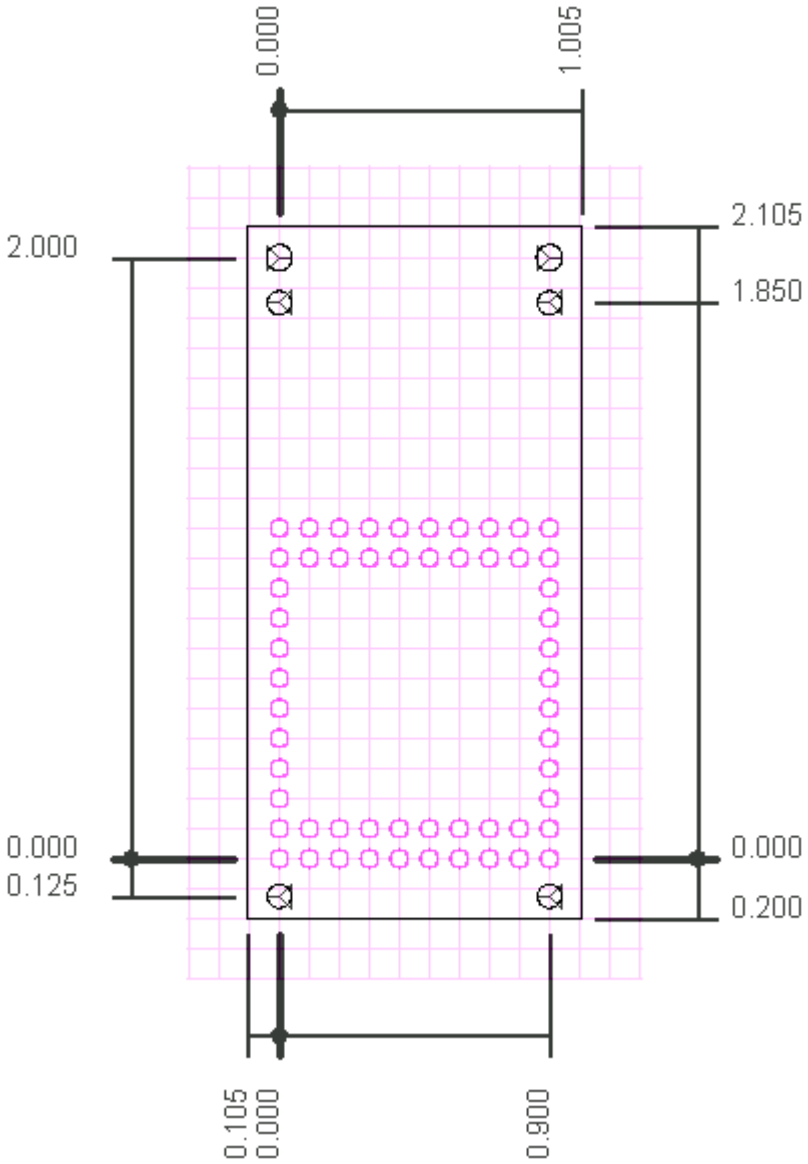
All dimensions are given in inches (1” = 25.4mm).

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

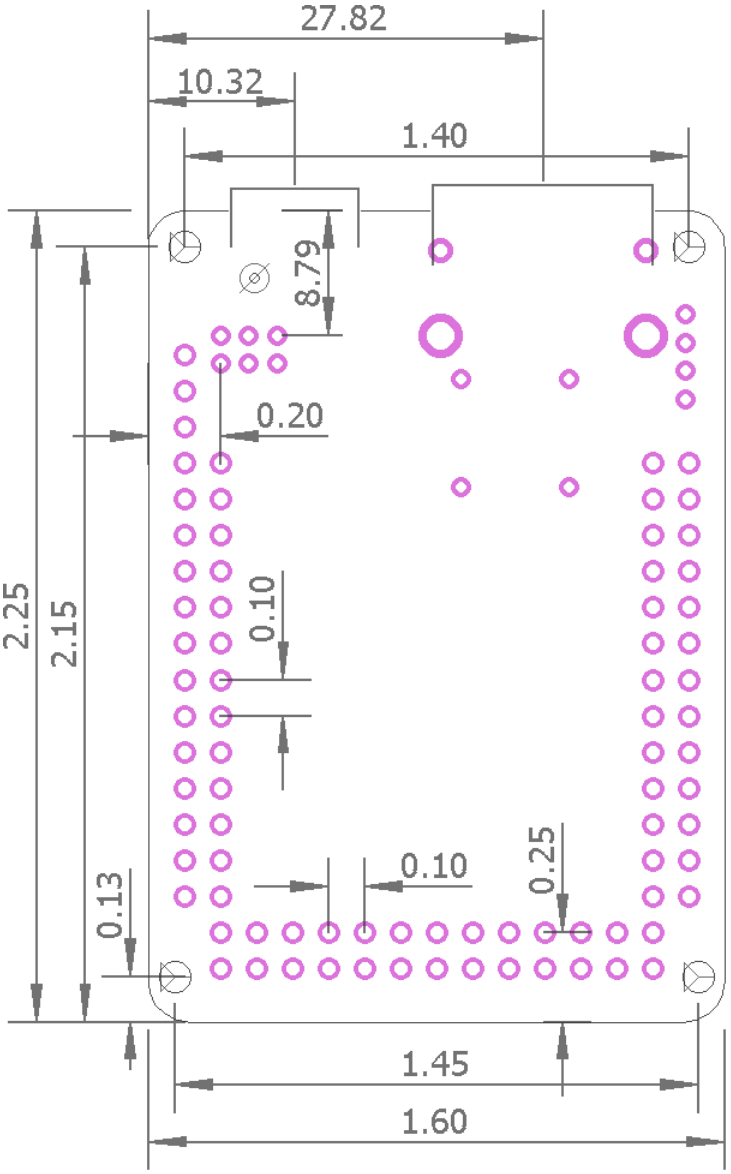
12.1 Pluto



12.2 Pluto-IIx



12.3 Pluto-XC6



12.4 Pluto-3