

OpenPiton in Action

Princeton University

<http://openpiton.org>

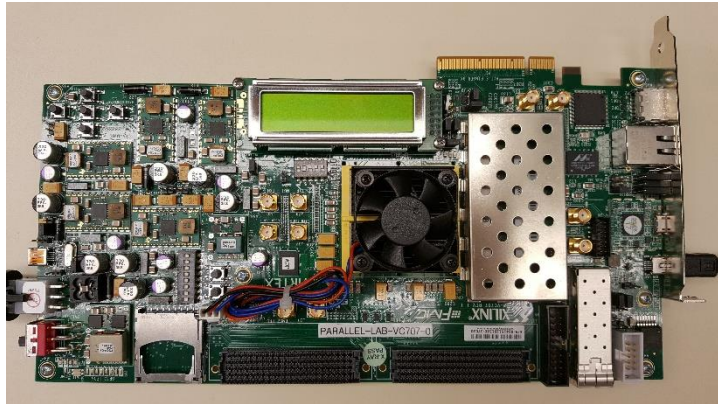


OpenPiton

FPGA Prototyping

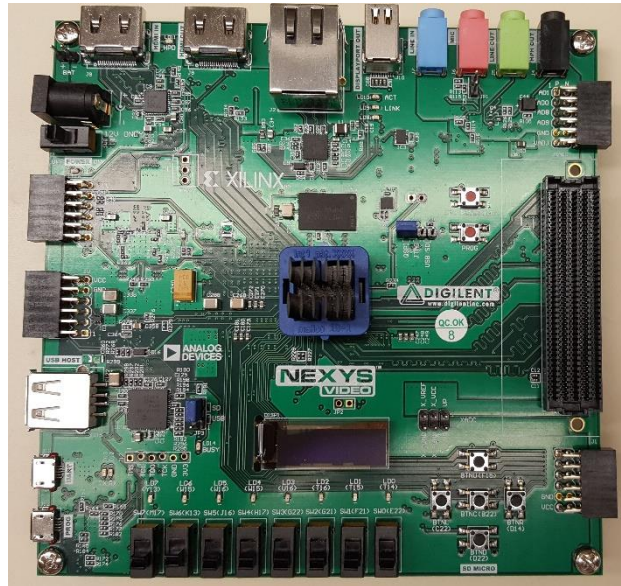
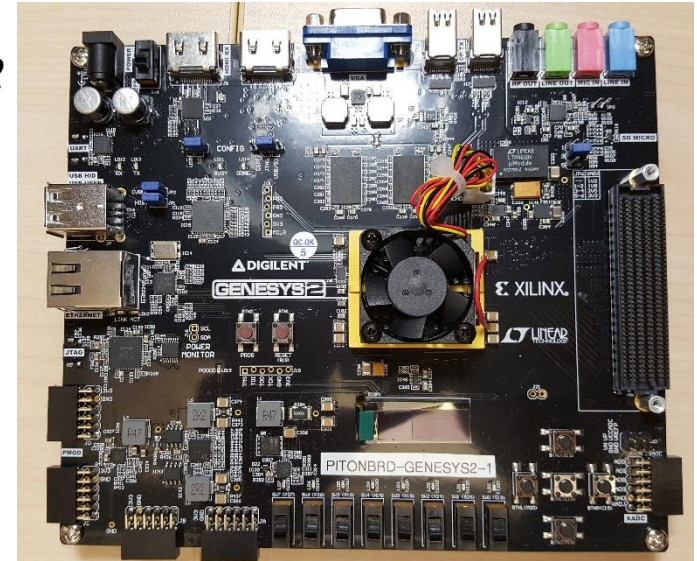
Supported Development Boards

Boards supported by toolchain:

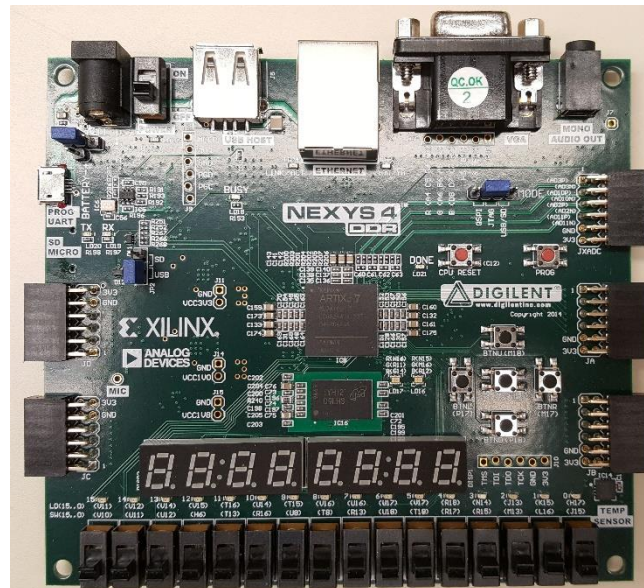


Xilinx VC707

Digilent *Genesys2*



Digilent NexysVideo



Digilent Nexys4DDR*

* doesn't have DDR controller and FPU

Comparison of Supported Boards

Development Board, FPGA name, Part	Core Clock (1 core)	Max # of Cores	DDR Type, Size, Data Width	Price (nonacademic/ academic)
Xilinx VC707 Virtex-7 XC7VX485T-2FFG1761C	60 MHz	3	DDR3 1 GB 64 bits	\$3,495
Digilent Genesys2 Kintex-7 XC7K325T-2FFG900C	67 MHz	2	DDR3 1GB 32 bits	\$1,299/ \$600
Digilent NexysVideo Artix-7 XC7A200T-1SBG484C	30 MHz	1	DDR3 512MB 16 bits	\$490/ \$250
Digilent Nexys 4 DDR Artix-7 XC7A100T-ACSG324C	30 MHz	1	DDR2 128MiB 16 bits	\$320/ \$160

Prototype Architecture

DDR controller*:

- Xilinx's MIG 7 IP core
- Configurable data width
- Used as main memory

Wishbone SD Master*:

- Up to 32GB SD/SDHC cards
- Storage for HV/OS/tests

UART:

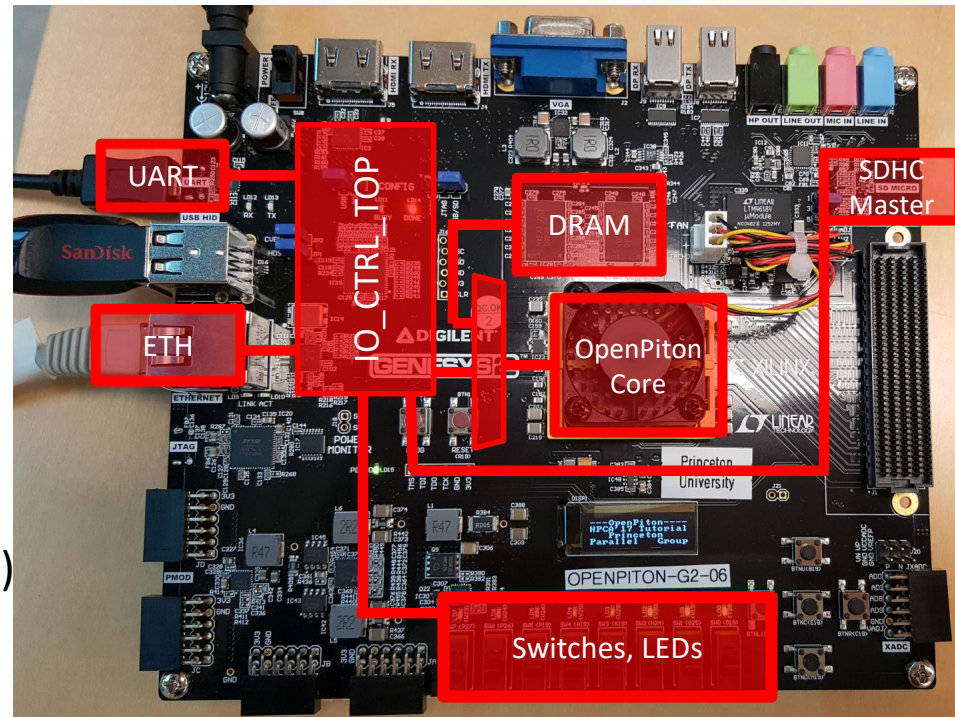
- Terminal I/O
- Loading of assembly test

(DMW - Direct Memory Write from a host)

Ethernet controller*:

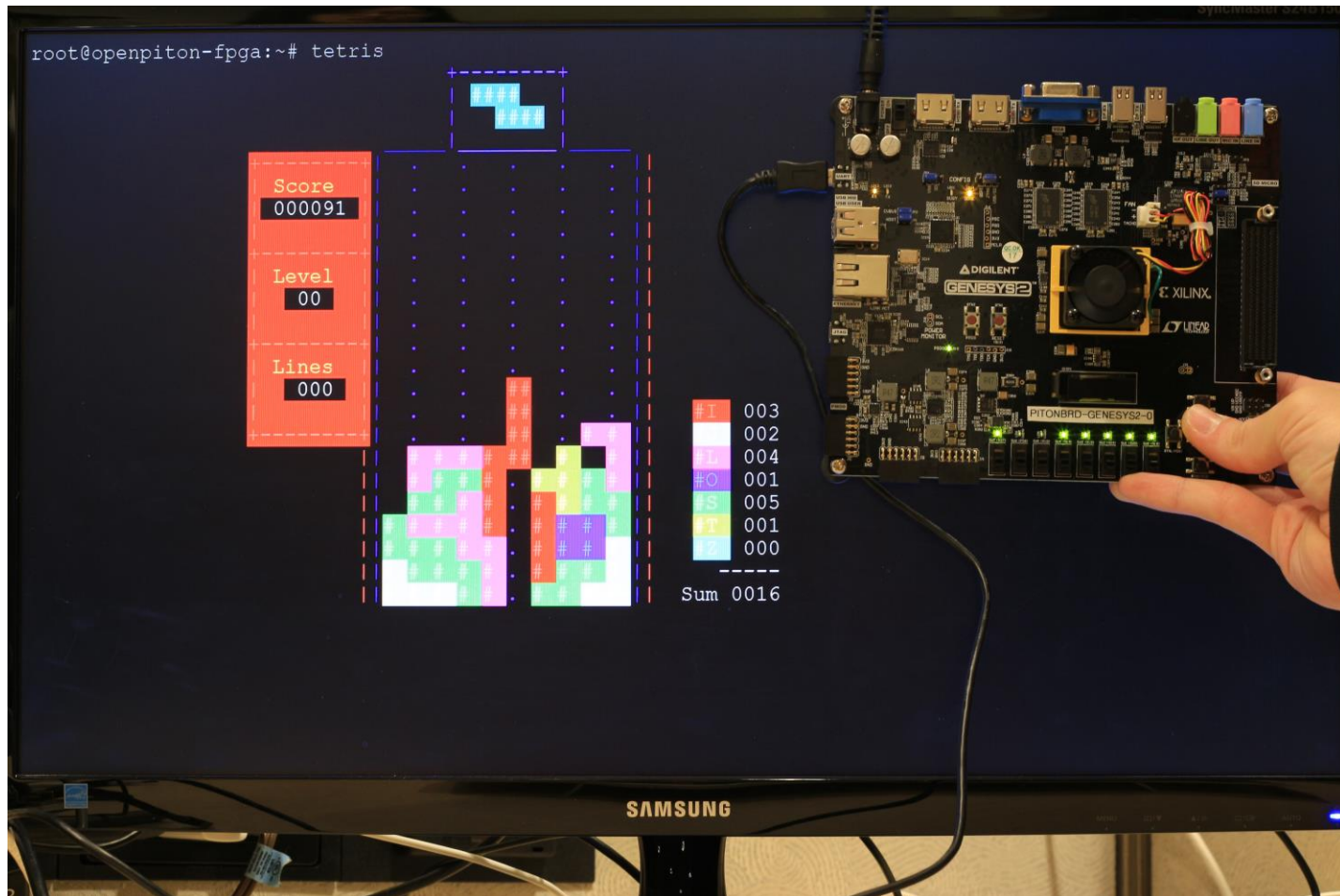
- Xilinx's Ethernet Lite MAC IP Core
- Driver from Linux kernel
- 100 Mb/s

*optional



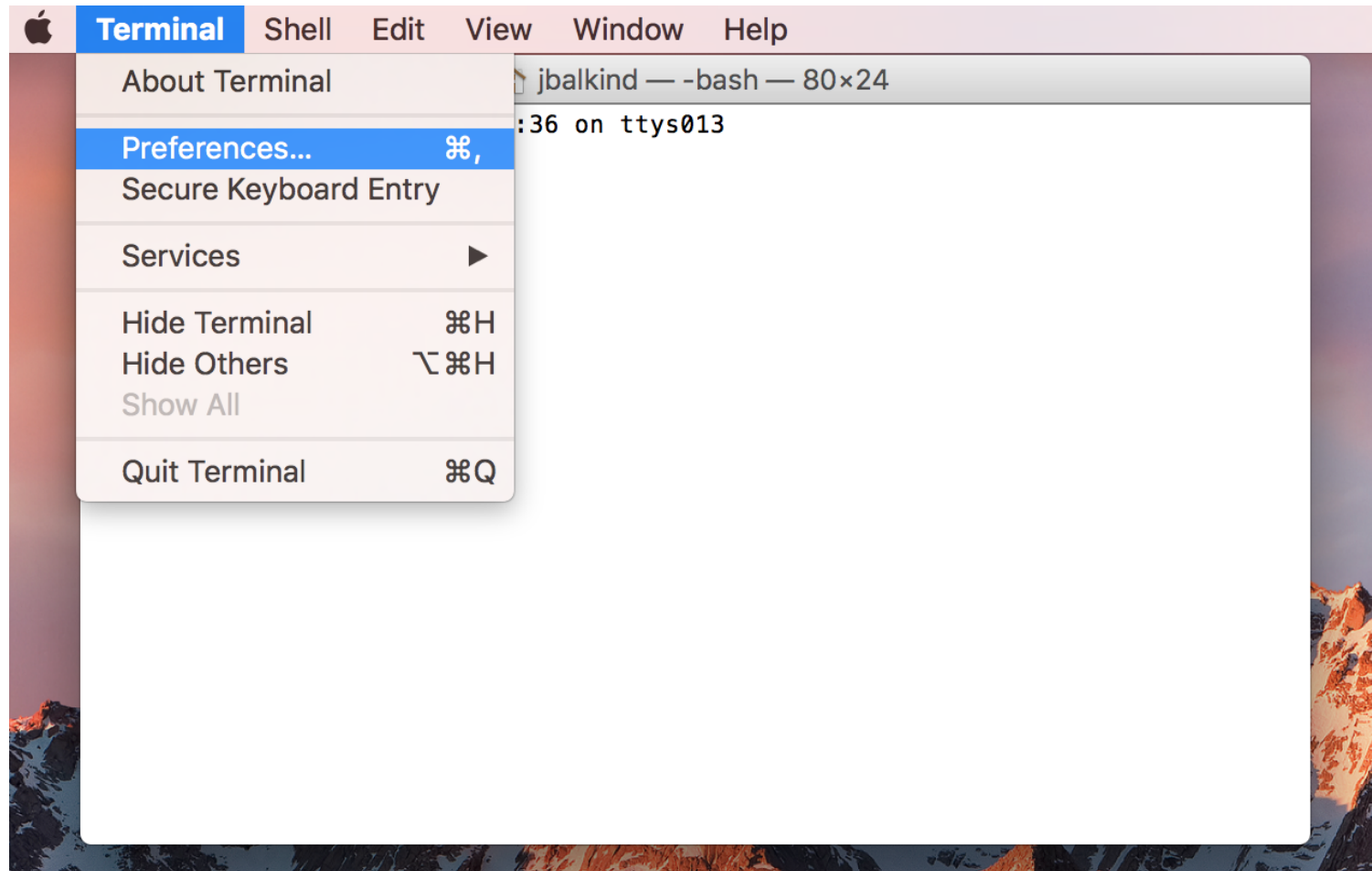
Digilent *Genesys2*

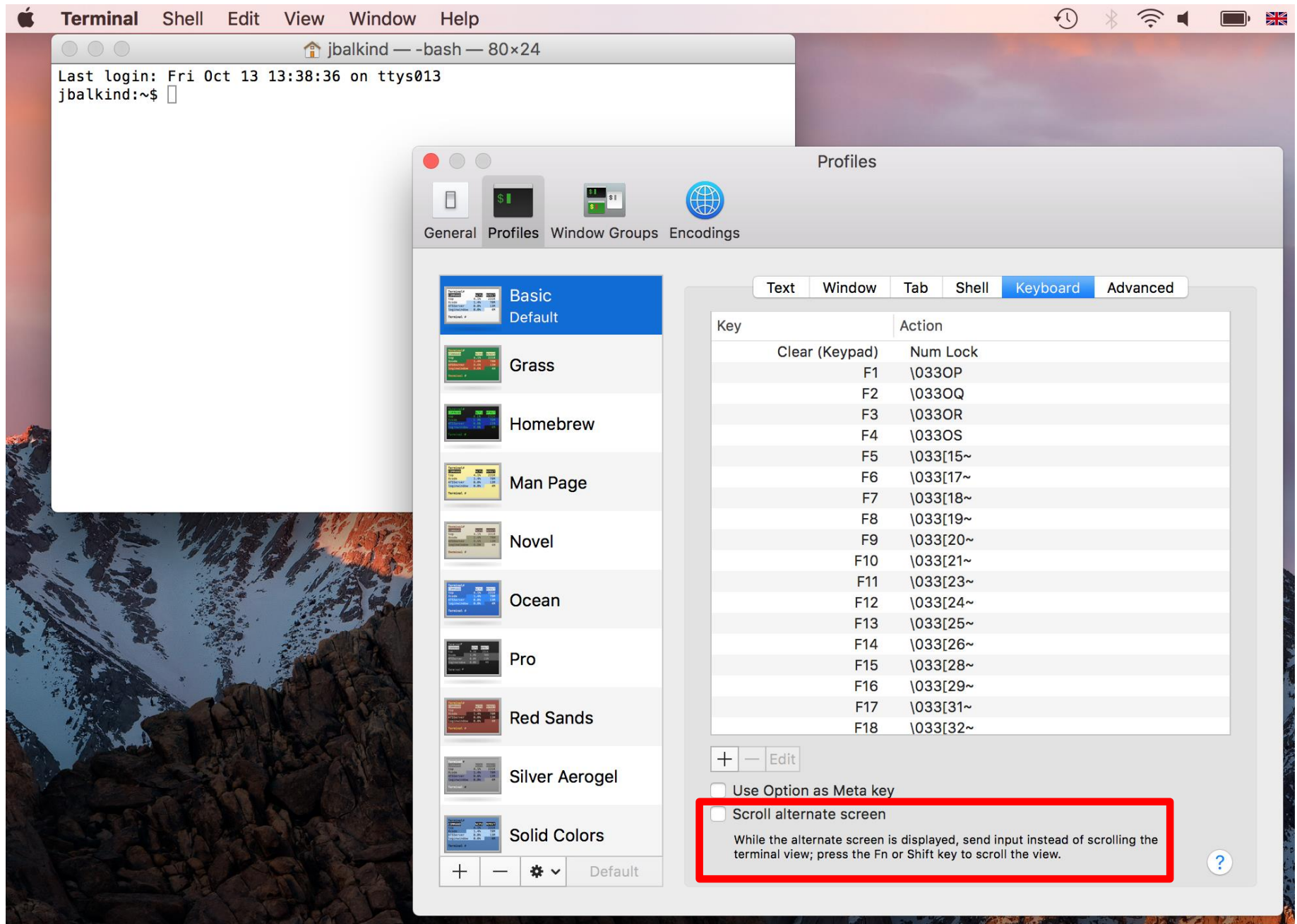
Demo



Setup for Hands-on with FPGA

Setting Up Terminal (MAC)





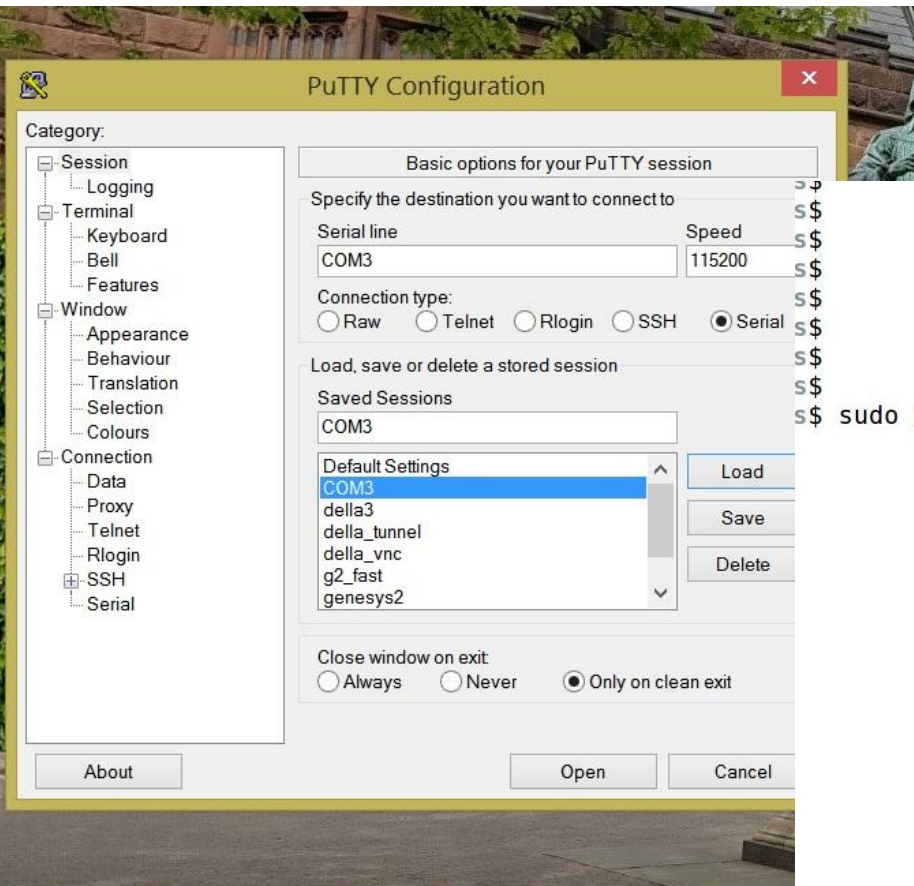
Setting Up Terminal

- Find serial device:
 - Windows: Device Manager
 - Linux: /dev/
- Unplug/Plug back USB cable to determine the right one

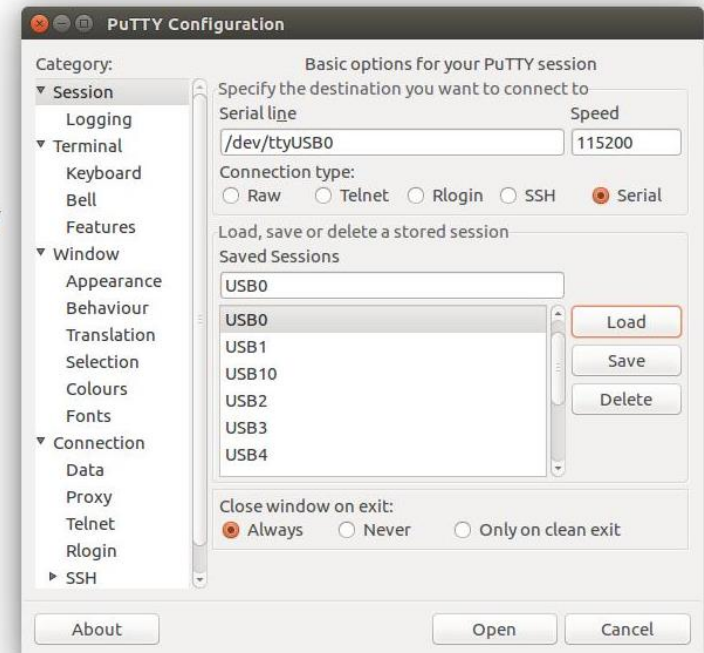
Setting Up Terminal (Windows, Linux)

Serial Line: /dev/ttyUSBX or COMX, where X is a number depending on your system

Speed: 115200



sudo putty



The image shows a Digilent Genesys2 development board. Key components and connections include:

- Power and Data Connectors (Red Boxes):**
 - Top left: Power and ground pins.
 - Left side: USB-A to UART bridge and a SanDisk SD card.
 - Bottom left: Ethernet port.
 - Right side: SD card slot.
- Board Components:**
 - Processor:** Xilinx Zynq-7010 SoC.
 - Display:** 3.5-inch LCD screen showing "Princeton University".
 - Connectors:** USB-A, USB-B, HDMI, VGA, DP, and various pin headers.
 - Buttons:** Power button, reset button, and several push buttons.
 - LEDs:** Status LEDs for power, data, and other functions.
- Overlays:**
 - A large yellow "GO!" is overlaid on the bottom center.
 - A "QC OK 2" sticker is near the top center.
 - A "Princeton University" label is on the display.
 - A "GENESYS2" label is on the top left.
 - A "DIGILENT" label is on the top center.
 - A "XILINX" label is on the top right.
 - A "LINEAR TECHNOLOGY" label is on the bottom right.

Booting Linux

```
Alive and well ...
Strand start set = 0x1
Total physical mem = 0x40000000
Scrubbing the rest of memory
Number of strands = 0x1
membase = 0x0
memsize = 0x1000000
physmem = 0x40000000
done
returned status 0x0
setup everything else
Setting remaining details
Start heart beat for control domain
```

WARNING: Unable to connect to Domain Service providers

WARNING: Unable to get LDOM Variable Updates

WARNING: Unable to update LDOM Variable

```
OpenPiton, No Keyboard
Copyright 2007 Sun Microsystems, Inc. All rights reserved.
OpenBoot 4.x.build_122***PROTOTYPE BUILD***, 1008 MB memory available, Serial #66711024.
[greddy obp #0]
Ethernet address 0:e0:81:5f:2c:ab, Host ID: 83f9edf0.
```

ok boot Linux **After ~10s**

FPGA Linux Boot

```
ok boot Linux  
Boot device: /virtual-devices/disk@110  File and args: Linux  
SILO Version 1.4.14
```

```
Allocated 64 Megs of memory at 0x40000000 for kernel
```

After ~2min systemd starts

Coffee Break

FPGA Linux Boot

```
[FAILED] Failed to start Journal Service.
See 'systemctl status systemd-journald.service' for details.
[ 1506.007339] systemd[1]: systemd-journald.service: Unit entered failed state.
[ 1506.047520] systemd[1]: systemd-journald.service: Failed with result 'timeout'.
[ 1506.199855] systemd[1]: systemd-journald.service: Service has no hold-off time, scheduling restart.
[ 1506.385968] systemd[1]: Stopped Journal Service.
[  OK ] Stopped Journal Service.
[ 1509.358259] systemd[1]: Starting Journal Service...
        Starting Journal Service...
[**    ] A start job is running for Journal Service (17s / 1min 31s)[ 1524.683177] systemd[1]: Started Journal Service.
[  OK ] Started Journal Service.
[  OK ] Reached target System Initialization.
[  OK ] Listening on D-Bus System Message Bus Socket.
[  OK ] Reached target Sockets.
[  OK ] Started Daily apt activities.
[  OK ] Started Daily Cleanup of Temporary Directories.
[  OK ] Reached target Timers.
[  OK ] Reached target Basic System.
        Starting Permit User Sessions...
        Starting OpenBSD Secure Shell server...
[  OK ] Started D-Bus System Message Bus.
        Starting LSB: Start NTP daemon...
        Starting Login Service...
[  OK ] Started Permit User Sessions.
[FAILED] Failed to start OpenBSD Secure Shell server.
See 'systemctl status ssh.service' for details.
[FAILED] Failed to start Login Service.
See 'systemctl status systemd-logind.service' for details.
[  OK ] Stopped Login Service.
        Starting Login Service...
[  OK ] Stopped OpenBSD Secure Shell server.
        Starting OpenBSD Secure Shell server...
        Starting Cleanup of Temporary Directories...
[  OK ] Started Getty on tty1.
[  OK ] Started Console Getty.
[  OK ] Reached target Login Prompts.
```

Debian GNU/Linux stretch/sid piton-0 console

piton-0 login:

Hands on: Login to the System

```
piton-0 login: root
Password: 
Linux piton-0 4.7.0-rc7-openpiton #50 SMP Thu Jan 26 14:43:38 EST 2017 sparc64
```










The programs included with the Debian GNU/Linux system are free software; the exact distribution terms for each program are described in the individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.
root@piton-0:~#

Login: root

Password: root

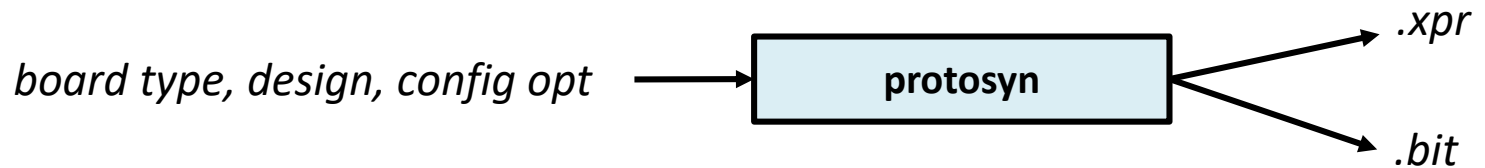
Suggested Configurations

	BRAM_TEST	SD with OS + Eth	UART DMW to DDR
BRAM with hardwired test			
DRAM memory controller			
SD card controller			
UART 16550			
Ethernet Lite MAC			
UART support for test streaming			

Tools

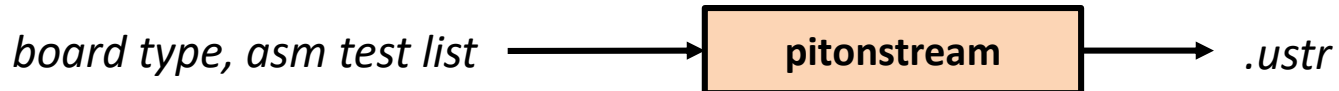
- `protosyn`

All encompassing tool for creation of FPGA project and generating programming file



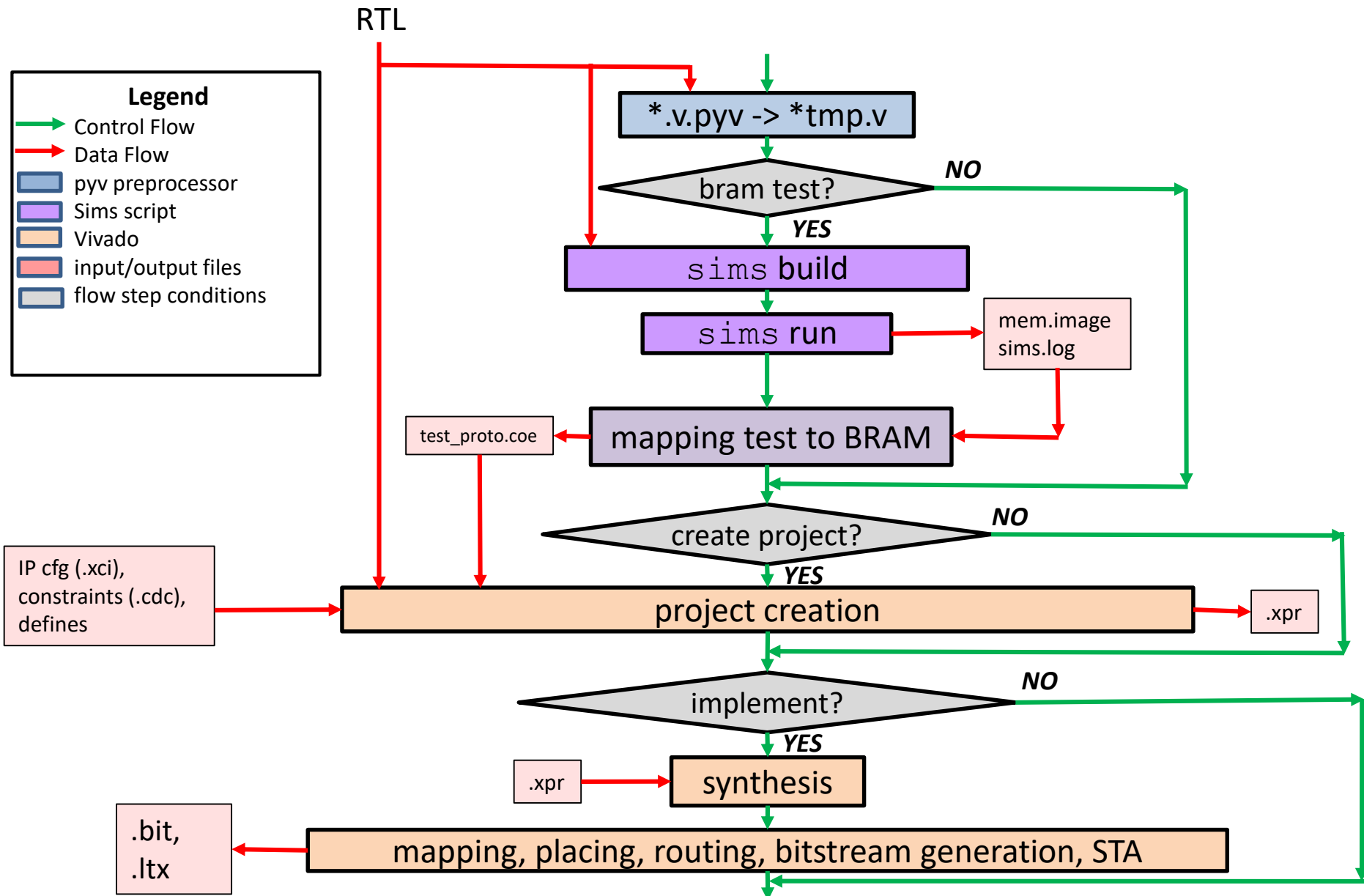
- `pitonstream`

Tool for running assembly tests on FPGA



Sources are located at `piton/tools/src/proto/`

protosyn Flow



Bringing up Network

```
root@piton-0:~# ifconfig eth0 hw ether 52:54:00:a1:ec:30
root@piton-0:~# dhclient -v eth0
Internet Systems Consortium DHCP Client 4.3.5b1
Copyright 2004-2016 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/
```

**Put a MAC from
your board!**

```
[ 6527.207484] xilinx_emaclite f026d9b0 eth0: Link is Down
[ 6529.353806] xilinx_emaclite f026d9b0 eth0: Link is Up - 100Mbps/Full - flow control rx/tx
Listening on LPF/eth0/52:54:00:a1:ec:30
Sending on   LPF/eth0/52:54:00:a1:ec:30
Sending on   Socket/fallback
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 4
DHCPREQUEST of 192.168.0.104 on eth0 to 255.255.255.255 port 67
DHCPOFFER of 192.168.0.104 from 192.168.0.254
DHCPACK of 192.168.0.104 from 192.168.0.254
bound to 192.168.0.104 -- renewal in -7956855 seconds.
root@piton-0:~#
```

Running protosyn

Usage:
protosyn **-b <board_type>** **[-d <design>]** **[--bram-test <test_name>]** [--from <FPGA flow step>]
led <string>]

-b, --board <board_type>

Name of a supported Xilinx's development board. Available options are:

vc707

genesys2

nexysVideo

-d, --design <design>

Name of design module to synthesize. The default is 'system', which synthesizes a full system with chip and chipset. See \$DV_ROOT/tools/src/proto/block.list for supported design modules

--bram-test <test_name>

Name of the test to be mapped into BRAM

--no-ddr

Implement design without DDR memory

--eth

Add Ethernet controller to implementation

...

more options are in FPGA manual

Example protosyn run

```
[openpiton@della2]$ protosyn -b genesys2 -d system --bram-test=uart16550-hello-world.s
INFO: Synthesizing a test: uart16550-hello-world.s
INFO: Compilation started
INFO: Simulation started
INFO: Using core clock frequency: 50 MHz
INFO: Test Passed!
INFO: Starting mapping of a test to BRAM
INFO: Length of image file: 52597
INFO: Checking correctness of section mapping...
INFO: Correct!
INFO: Used 96 out of 16384 blocks of storage
INFO: Creating UART stream for a test...
INFO: Creating project for design 'system' on board 'genesys2'
INFO: Running FPGA implementation down to bitstream generation
INFO: Implementation finished!
INFO: All timing constraints are met!
INFO: Protosyn finished!
[openpiton@della2]$
```

FPGA Flow Runtimes

- System including DDR controller
 - ~1.5 hour including IP generation
 - ~40 mins excluding IP generation

FPGA Flow Outputs

```
[alavrov@della2 system]$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/system
[alavrov@della2 system]$ ls
additional_defines.tcl  vivado_18131.backup.jou  webtalk_10265.backup.log
genesys2_system        vivado_18131.backup.log  webtalk_12480.backup.jou
protosyn_logs          vivado_18300.backup.jou  webtalk_12480.backup.log
vivado_iou             vivado_18300.backup.log  webtalk_13970.backup.jou
[alavrov@della2 protosyn_logs]$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/system/protosyn_logs
[alavrov@della2 protosyn_logs]$ ls
implementation.log  make_project.log
[alavrov@della2 protosyn_logs]$
```

FPGA Flow Outputs

```
[alavrov@della2 genesys2_system]$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/system/genesys2_system
[alavrov@della2 genesys2_system]$ ls
genesys2_system.cache      genesys2_system.runs      vivado.log
genesys2_system.hw         genesys2_system.xpr
genesys2_system.ip_user_files  vivado.jou
[alavrov@della2 genesys2_system]$
```

```
[alavrov@della2 genesys2_system.runs]$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/system/genesys2_system/genesys2_system.runs
[alavrov@della2 genesys2_system.runs]$ ls
impl_1  synth_1
[alavrov@della2 genesys2_system.runs]$
```

FPGA Flow Outputs

```
[alavrov@della2 impl_1]$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/system/genesys2_system/genesys2_system.runs/impl_1
[alavrov@della2 impl_1]$ ls
ISEWrap.js          runme.sh
ISEWrap.sh          system.bit
gen_run.xml         system.tcl
htr.txt             system.vdi
init_design.pb      system_clock_utilization_routed.rpt
opt_design.pb       system_control_sets_placed.rpt
place_design.pb     system_drc_opted.rpt
project.wdf         system_drc_routed.pb
route_design.pb     system_drc_routed.rpt
rundef.js          system_io_placed.rpt
runme.bat           system_opt.dcp
runme.log           system_placed.dcp
[alavrov@della2 impl_1]$
```

system_power_routed.rpt
system_power_summary_routed.pb
system_route_status.pb
system_route_status.rpt
system_routed.dcp
system_timing_summary_routed.rpt
system_timing_summary_routed.rpx
system_utilization_placed.pb
system_utilization_placed.rpt
vivado.jou
vivado.pb
write_bitstream.pb

Example `pitonstream` Run

```
root@piton-laptop-2:build# pitonstream -b genesys2 -f tests.txt
```

UART will be configured for 115200 baud rate

UART DIV Latch value: 27

Press reset button on FPGA

Waiting...

Loading a test...

100%

TEST OUTPUT >>>

Hi! I'm OpenPiton

<<< END OF TEST OUTPUT

```
uart16550-hello-world.s : PASSED
```

```
=====
All tests finished
```

Exiting...

```
root@piton-laptop-2:build#
```

of 1 test

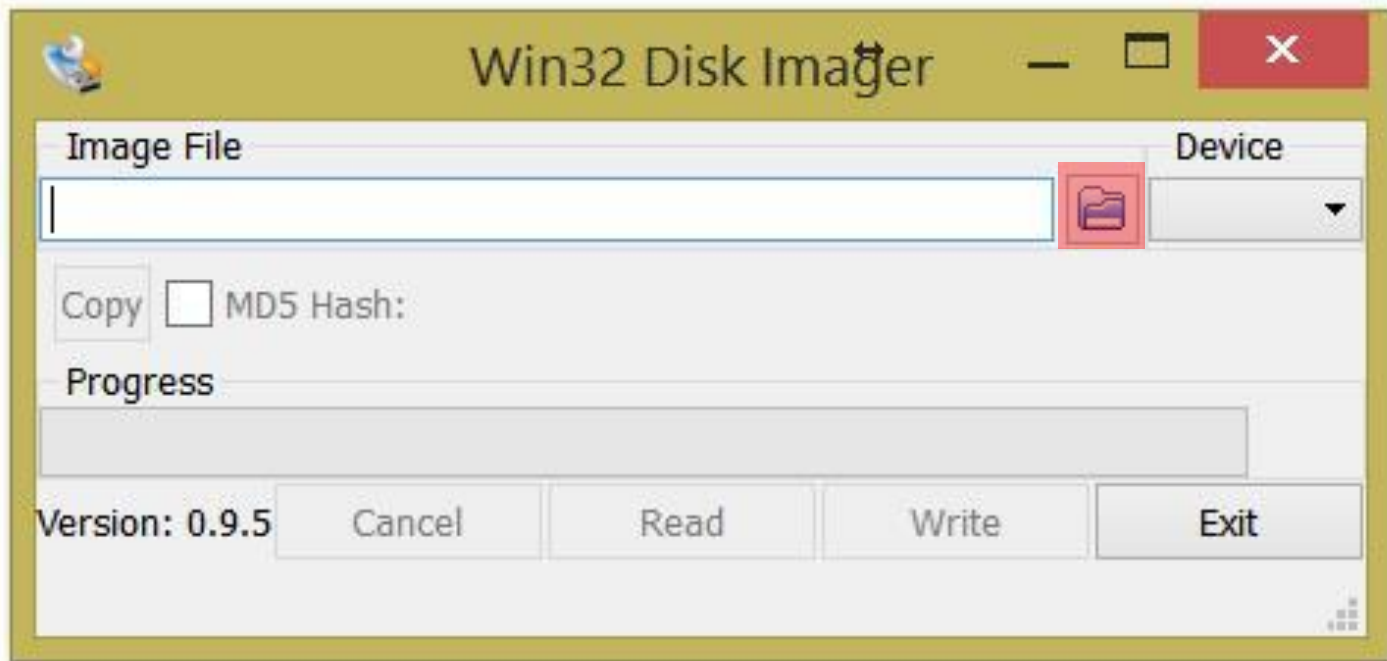
```
... Correct!  
rage
```

uart16550-hello-world.s

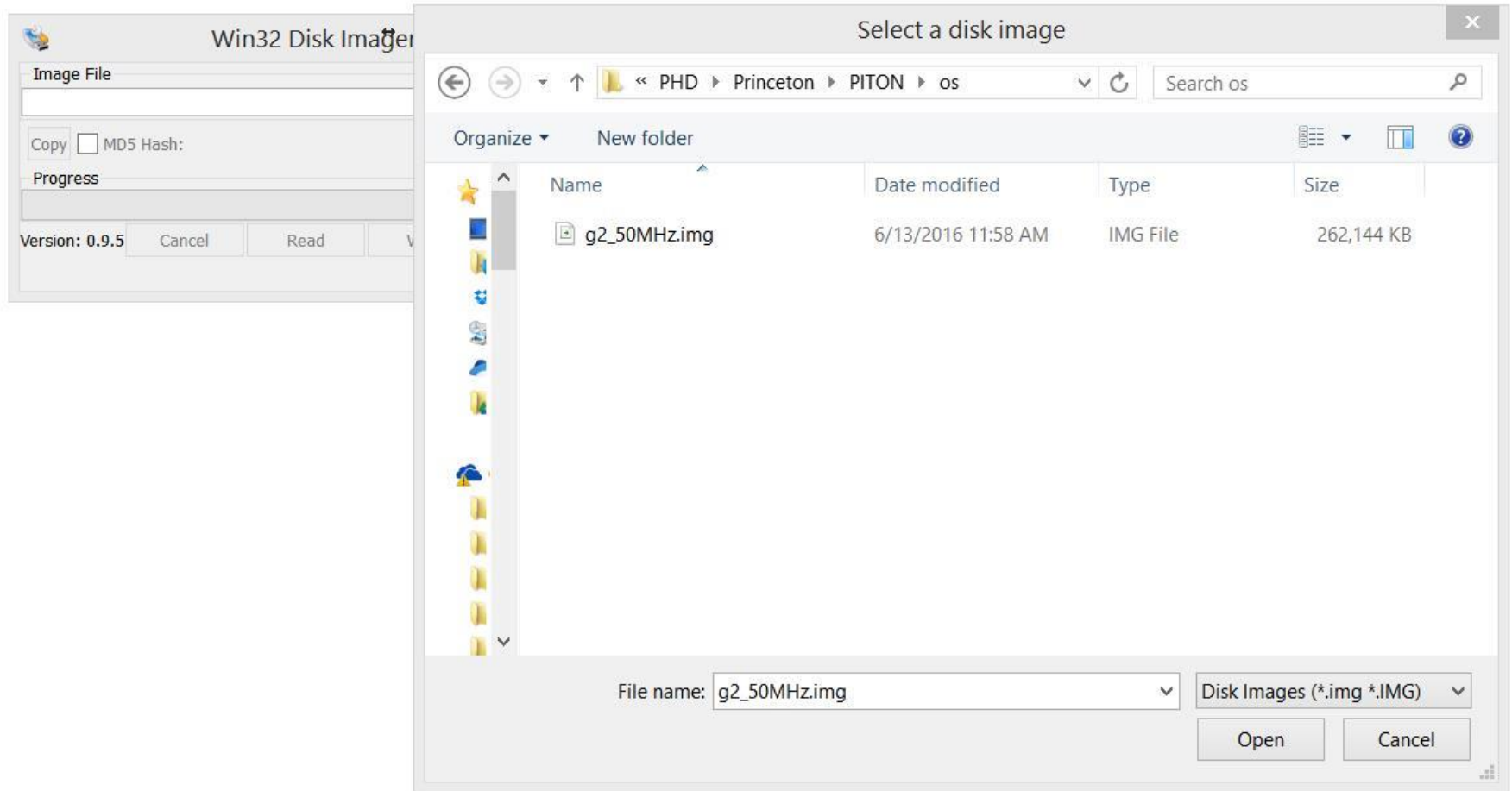
[illegible]

```
"tests.txt" [New] 1L, 24C written
```

Writing OS Image to SD Card (Windows)



Writing OS Image to SD Card (Windows)



Writing OS Image to SD Card (Windows)

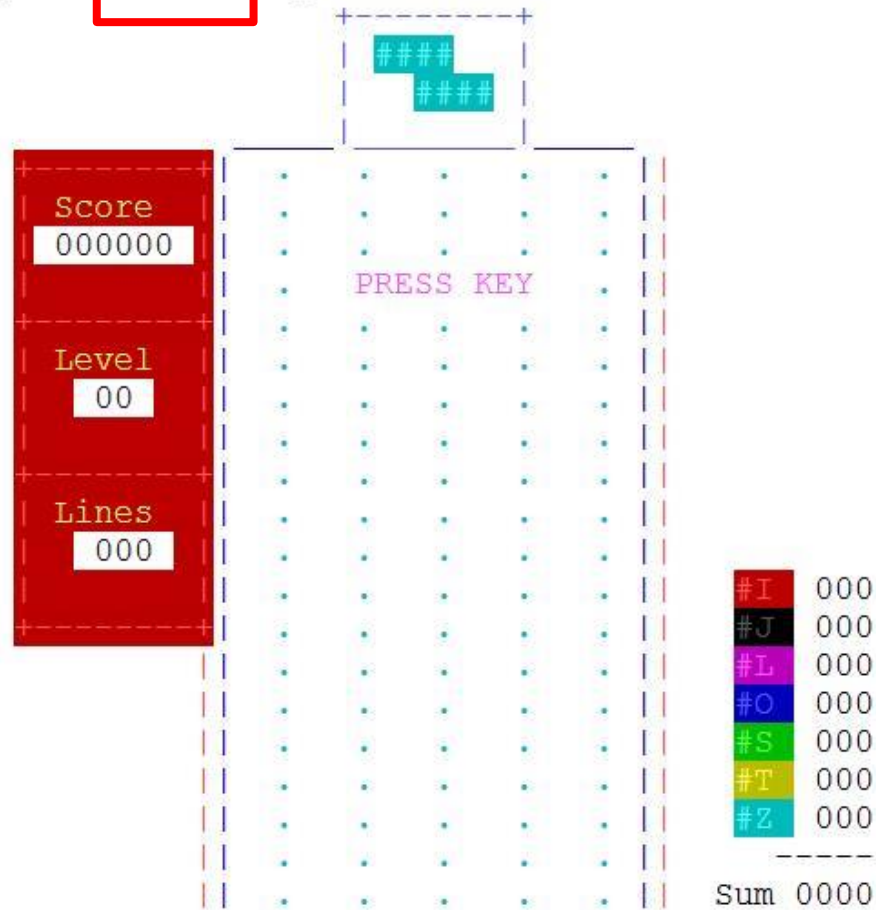


Hands-on with FPGA

Running Tetris on OpenPiton

COM5 - PuTTY

```
root@openpiton-fpga:~# tetris -bg white
```



Browsing OpenPiton web page on OpenPiton

```
root@piton-0:~# lynx www.openpiton.org
```

```
<<<
```

(p2 of 6)

OpenPiton

OpenPiton is the worlds first open source, general-purpose, multithreaded, manycore processor and framework. It is based on the Princeton Piton processor which was designed and taped-out in March 2015 by the Princeton Parallel Group. OpenPiton is open source across the entire computing stack, from the hardware to the firmware and software. Researchers and industry experts from many fields can utilize OpenPiton to modify any part of the stack and evaluate their ideas at scale. The hardware can be easily synthesized to FPGA and run an OS and applications at reasonable speeds for realistic evaluations. OpenPiton is designed to be highly configurable, including core count, cache sizes, and NoC topology, enabling it to adapt to different use cases. OpenPiton has an active community of users and is supported by the Princeton Parallel Group. Some of the features of OpenPiton include:

- * Open source (GPL core, BSD uncore) manycore
- * Written in Verilog HDL
- * Scalable up to 1/2 Billion Cores
- * Configurable core and uncore

```
-- press space for next page --
```

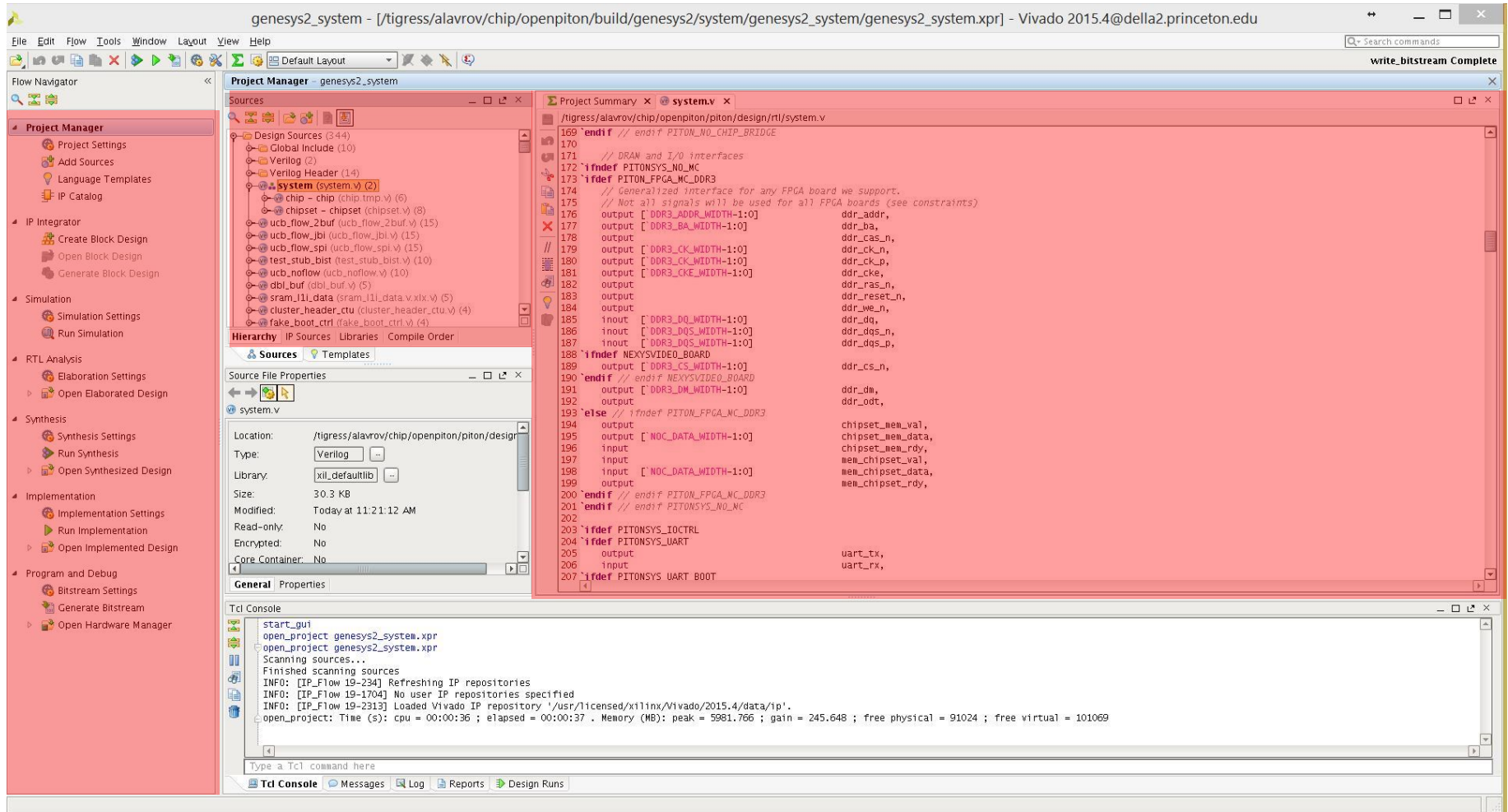
Arrow keys: Up and Down to move. Right to follow a link; Left to go back.
H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list

Backup Slides

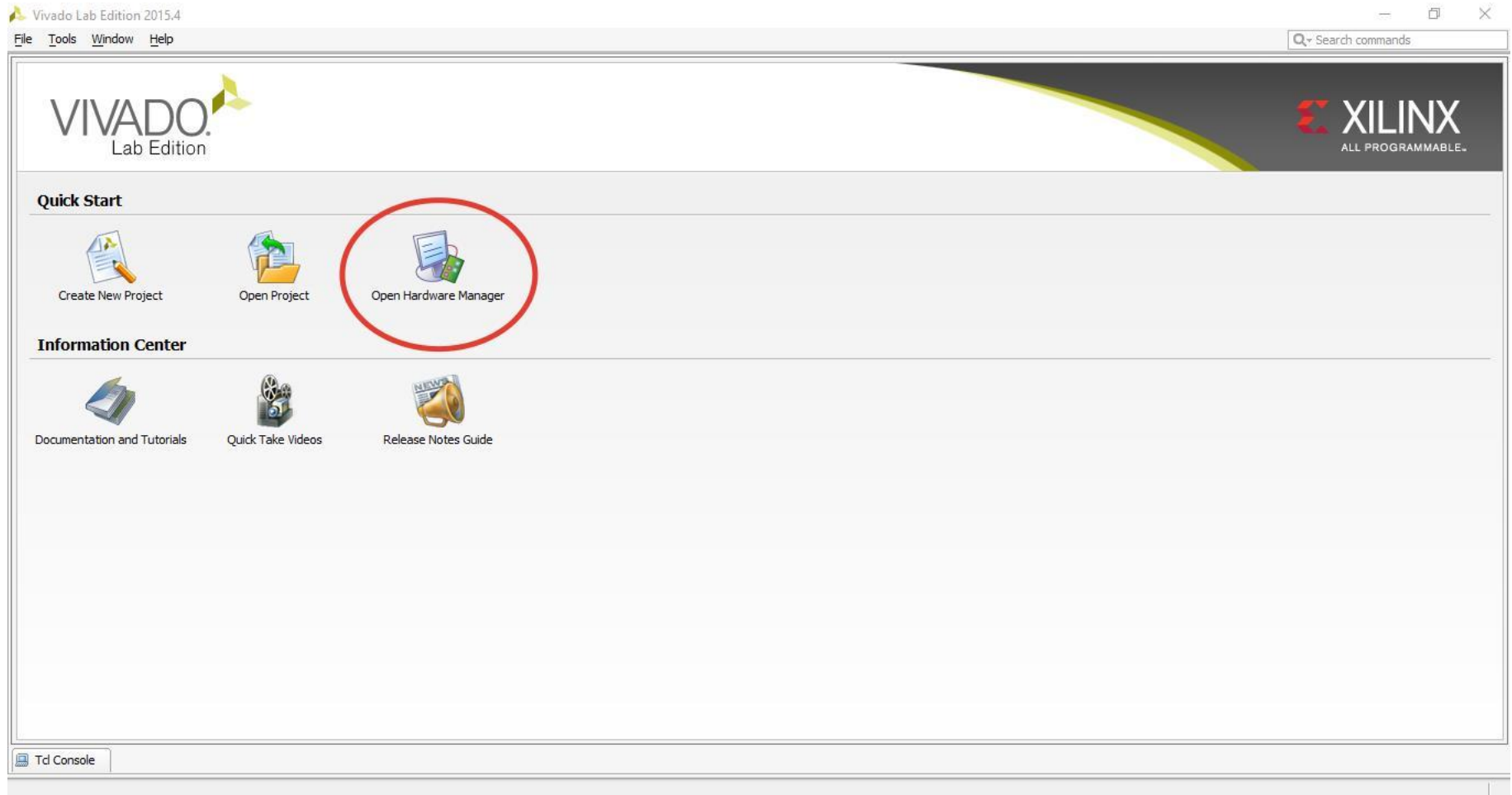
Opening FPGA Design

```
[alavrov@della2 genesys2_system]$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/system/genesys2_system
[alavrov@della2 genesys2_system]$ ls
genesys2_system.cache  genesys2_system.ip_user_files  genesys2_system.xpr  vivado.log
genesys2_system.hw     genesys2_system.runs           vivado.jou
[alavrov@della2 genesys2_system]$ vivado genesys2_system.xpr &
```

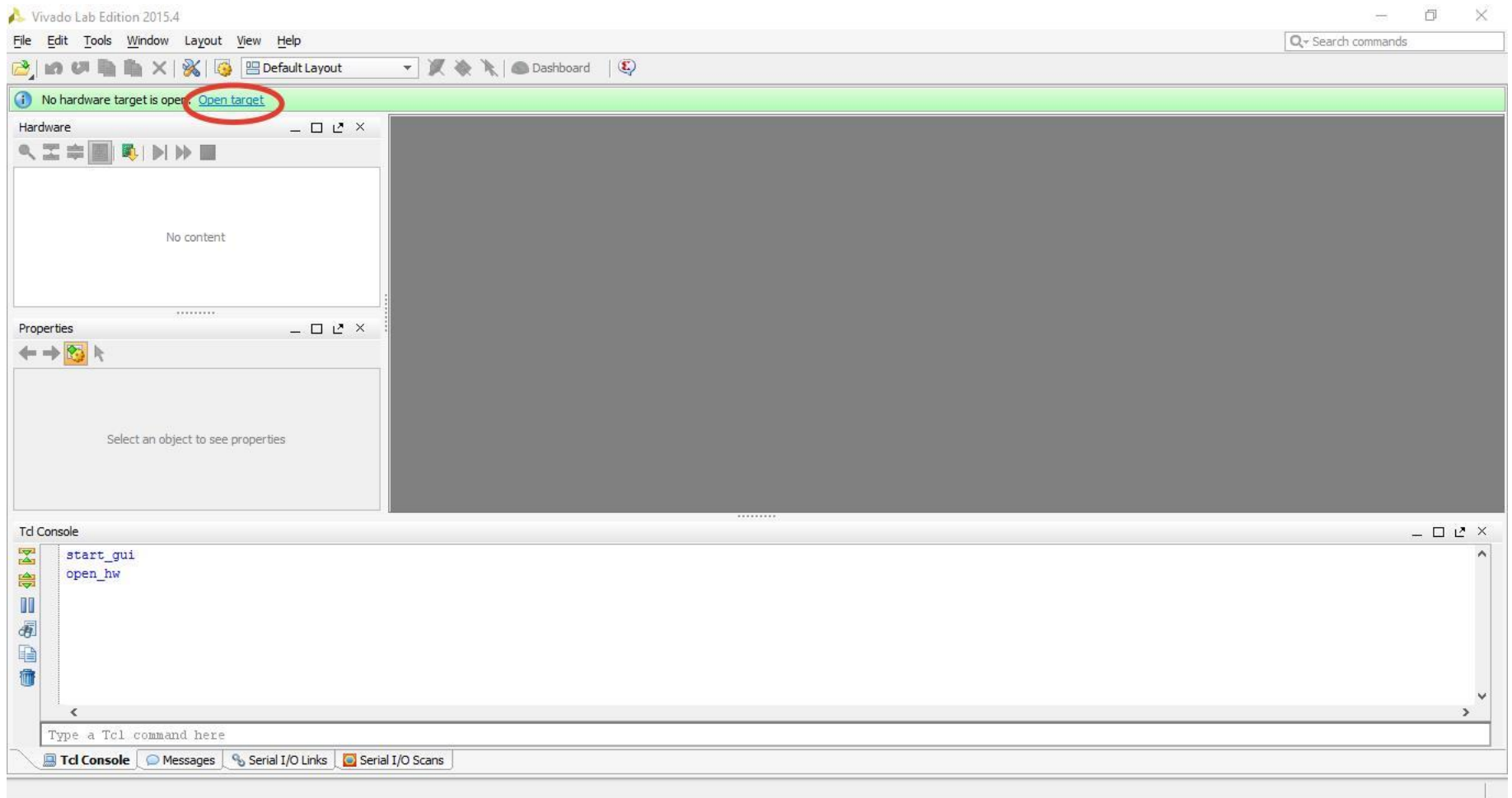
Opening FPGA Design



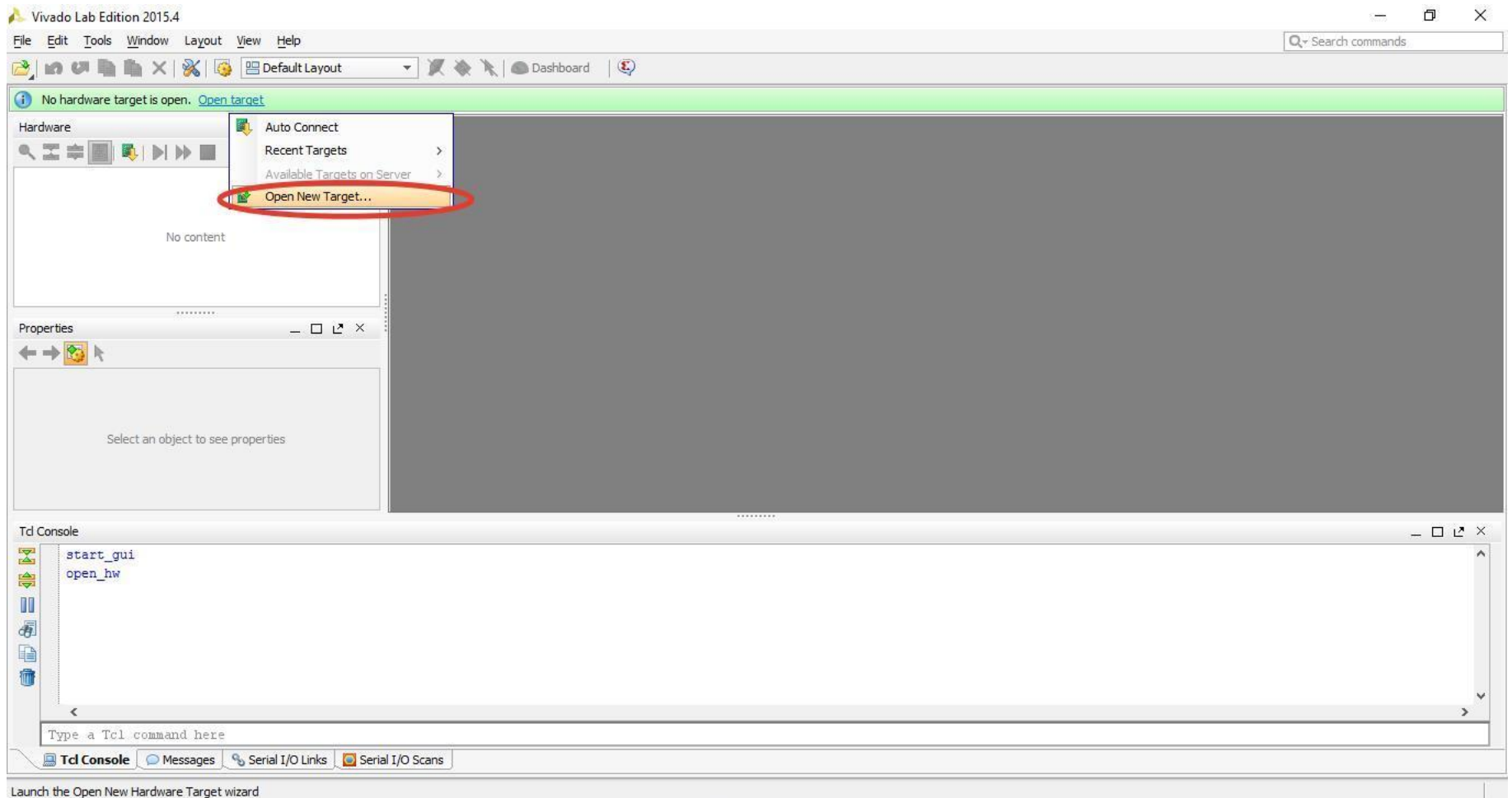
FPGA Programming



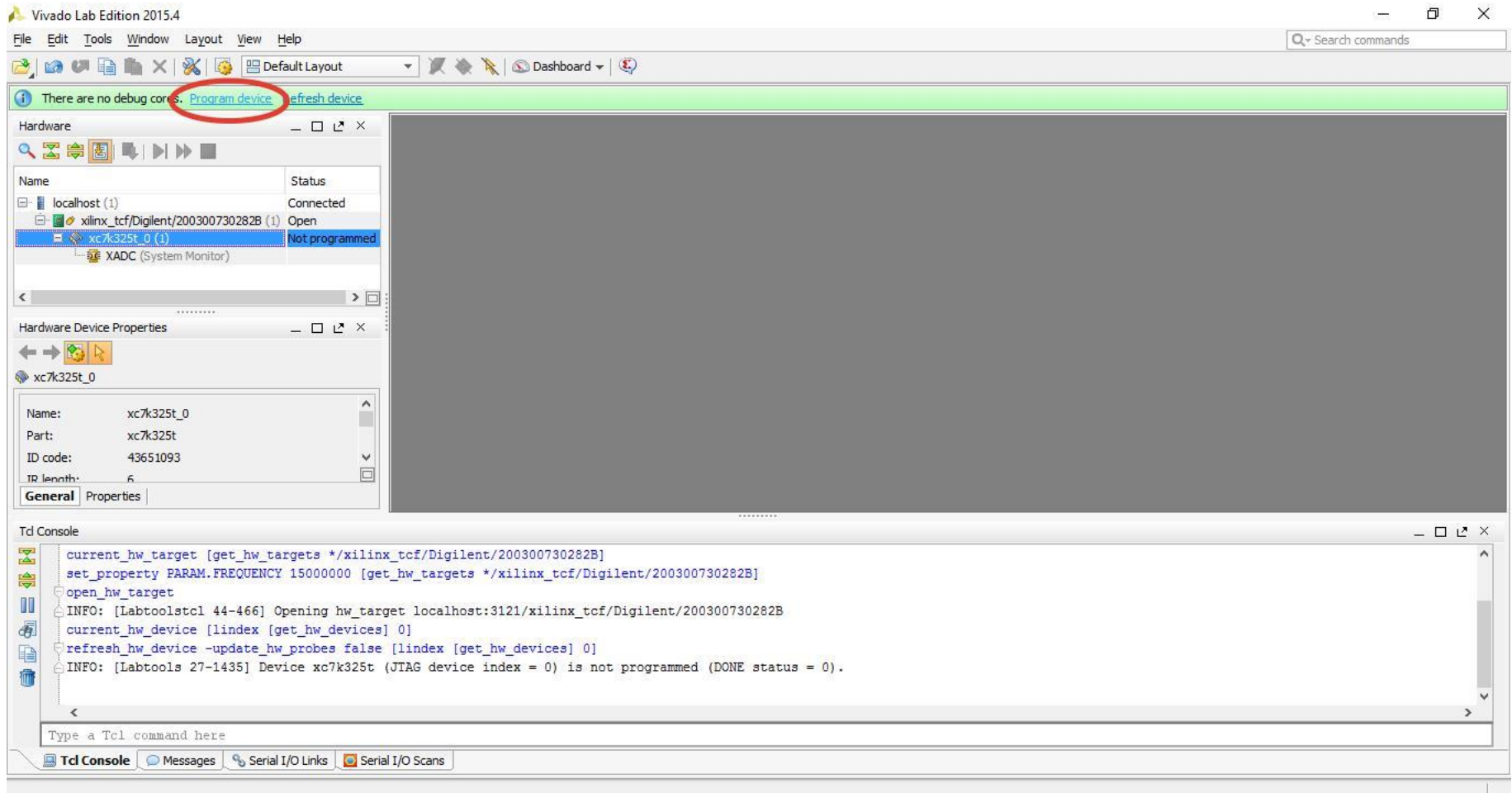
FPGA Programming



FPGA Programming



FPGA Programming



FPGA Programming

The screenshot displays the Vivado Lab Edition 2015.4 interface. The 'Hardware' panel on the left shows a tree view with 'xc7k325t_0 (1)' selected, indicating it is 'Not programmed'. The 'Hardware Device Properties' panel shows details for 'xc7k325t_0', including Name, Part, ID code, and IR length. The 'Program Device' dialog box is open in the center, prompting the user to select a bitstream file and debug probes file. It includes a checkbox for 'Enable end of startup check'. A yellow tooltip points to the 'Program' button, stating: 'Check End Of Startup (EOS) configuration status flag after programming is complete.' The 'Tcl Console' at the bottom shows a series of commands and responses, including 'current_hw_target', 'set_property', 'open_hw_target', 'current_hw_device', 'refresh_hw_device', and an 'INFO' message stating the device is not programmed.

Vivado Lab Edition 2015.4

File Edit Tools Window Layout View Help

Search commands

Default Layout Dashboard

There are no debug cores. Program device Refresh device

Hardware

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/200300730282B (1)	Open
xc7k325t_0 (1)	Not programmed
XADC (System Monitor)	

Hardware Device Properties

xc7k325t_0

Name: xc7k325t_0
Part: xc7k325t
ID code: 43651093
IR length: 6

General Properties

Program Device

Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.

Bitstream file:

Debug probes file:

☒ Enable end of startup check

Program Cancel

Check End Of Startup (EOS) configuration status flag after programming is complete.

Tcl Console

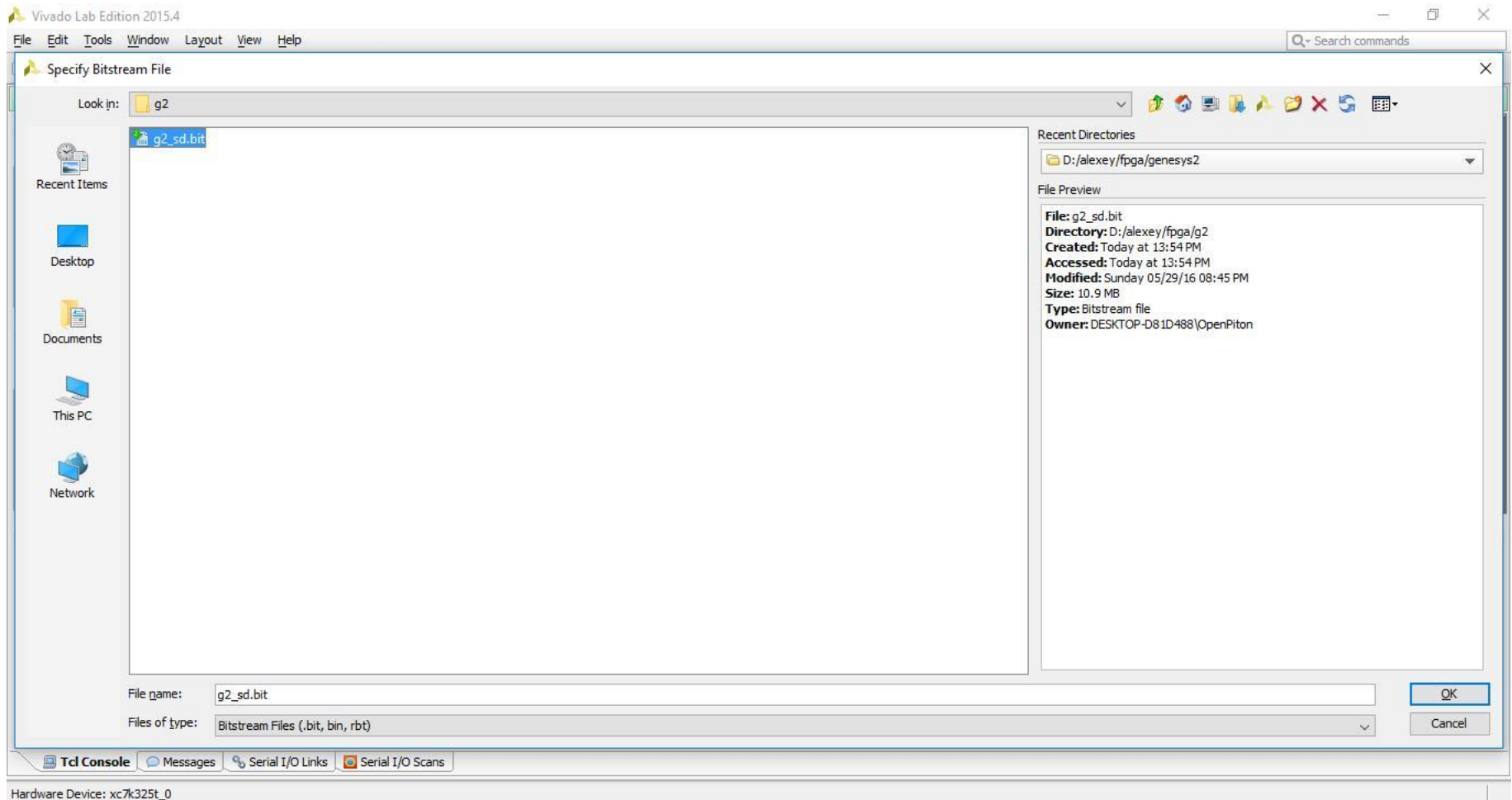
```
current_hw_target [get_hw_targets */xilinx_tcf/Digilent/200300730282B]
set_property PARAM.FREQUENCY 15000000 [get_hw_targets */xilinx_tcf/Digilent/200300730282B]
open_hw_target
INFO: [Labtoolstcl 44-466] Opening hw_target localhost:3121/xilinx_tcf/Digilent/200300730282B
current_hw_device [lindex [get_hw_devices] 0]
refresh_hw_device -update_hw_probes false [lindex [get_hw_devices] 0]
INFO: [Labtools 27-1435] Device xc7k325t (JTAG device index = 0) is not programmed (DONE status = 0).
```

Type a Tcl command here

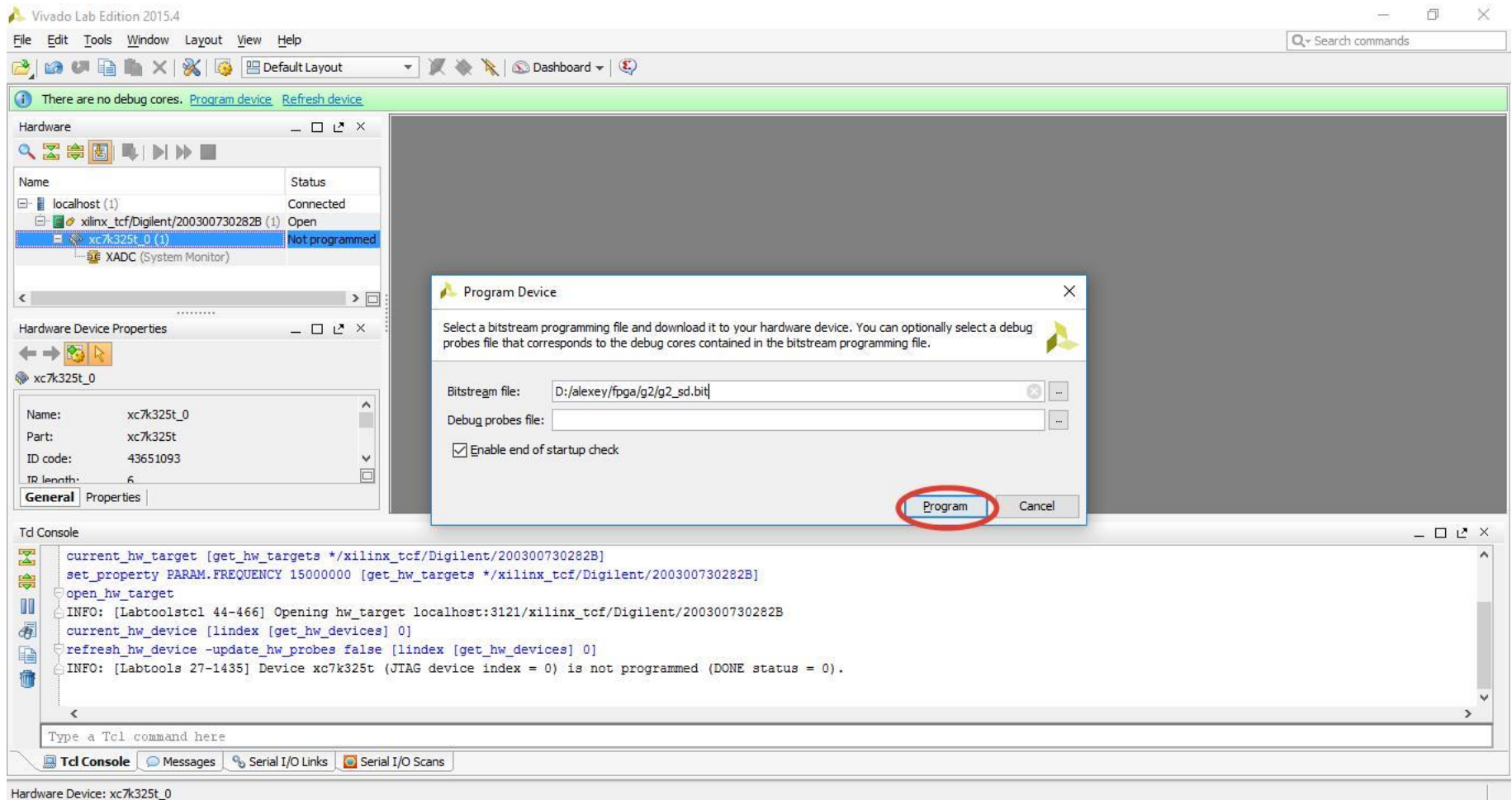
Tcl Console Messages Serial I/O Links Serial I/O Scans

Hardware Device: xc7k325t_0

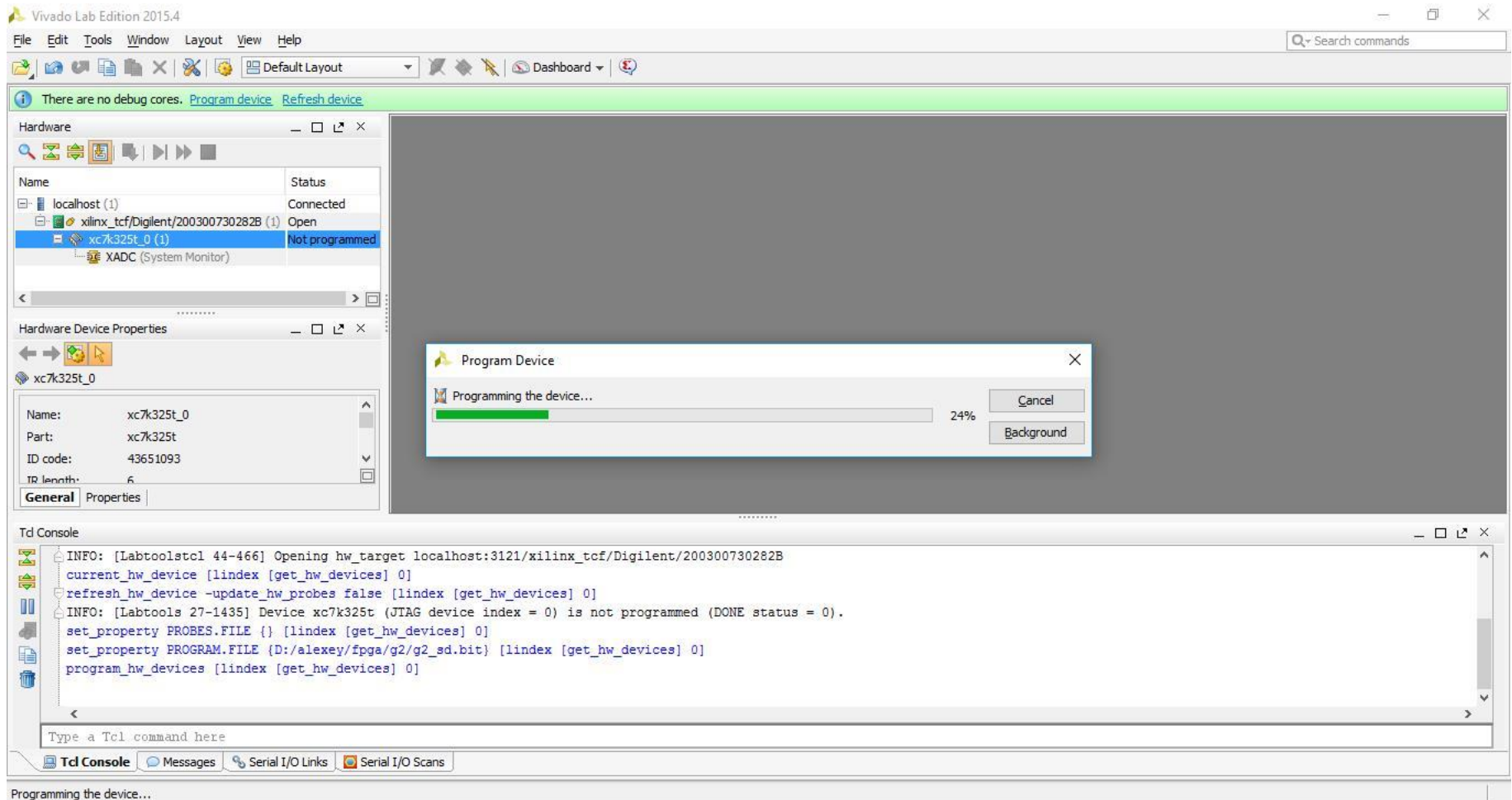
FPGA Programming



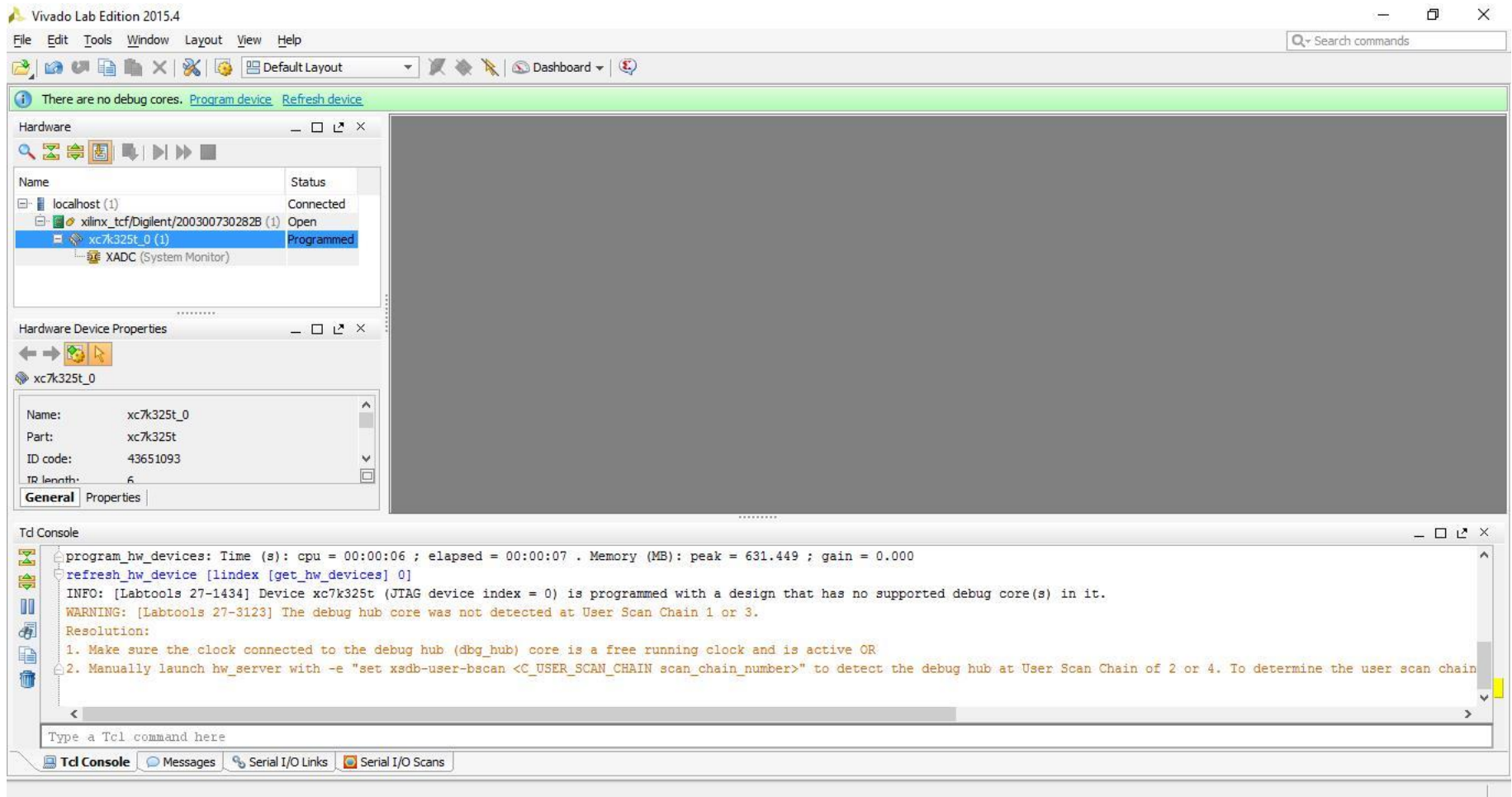
FPGA Programming



FPGA Programming

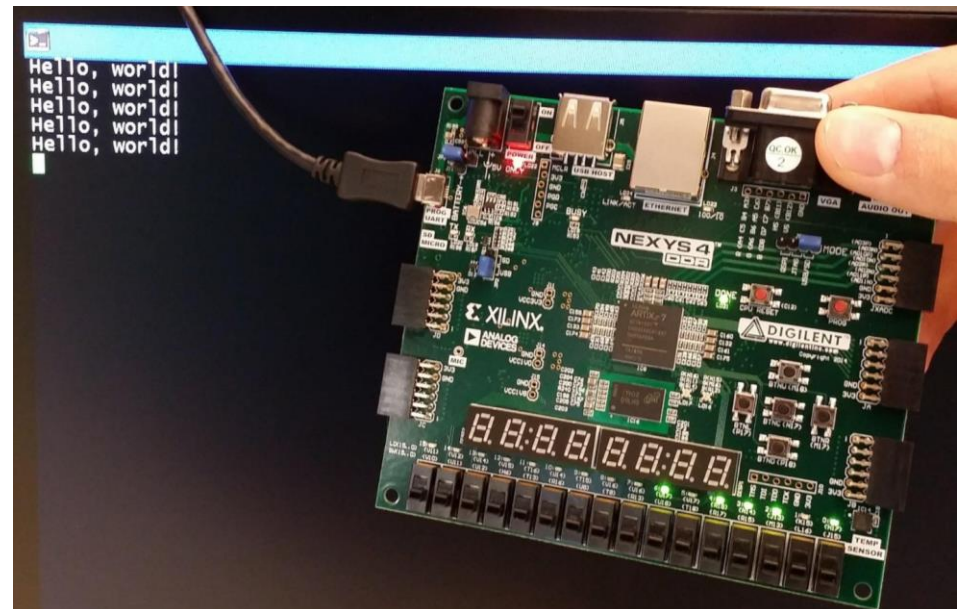


FPGA Programming



Synthesizing the *Hello, World!* Assembly Test

- download `tar.gz` of the release
- extract archive and set up environment
- run `protosyn -b genesys2 --no-ddr --bram-test uart-hello-world.s`
- wait until bit file is generated
- open Hardware Manager in Vivado or Vivado Lab Edition connected to Genesys2 board
- open a target and program the board with a generated `.bit` file
- open serial port on host machine
- press reset



Booting Debian Linux and Playing Tetris

- download `tar.gz` archive of OpenPiton release
- extract it and set up your environment and tools
- run `protosyn -b genesys2`
- wait until bit file is generated
- open Hardware Manager in Vivado or Vivado Lab Edition connected to Genesys2 board
- open a target and program the board with a generated `.bit` file
- write `.bin` file with OpenBoot and OS image on SD card
- insert the SD card into the board and press reset
- wait for Open Boot to start OK boot prompt
- print `boot Linux` command in OK boot prompt
- wait for Linux to boot
- use `root` both as login and password
- print `tetris` in Linux prompt and play the game!

