Getting to Work with OpenPiton

Jonathan Balkind, Michael McKeown, Yaosheng Fu, Tri Nguyen, Yanqi Zhou, Alexey Lavrov, Mohammad Shahrad, Adi Fuchs, Samuel Payne, Xiaohua Liang, Matthew Matl, David Wentzlaff

Princeton University

http://openpiton.org
FPGA Prototyping
Supported Development Boards

4 boards supported by toolchain:

- Digilent Genesys2
- Xilinx VC707
- Digilent NexysVideo
- Digilent Nexys4DDR*

* doesn’t have DDR controller and FPU
# Comparison of Supported Boards

<table>
<thead>
<tr>
<th>Development Board, FPGA name, Part</th>
<th>Core Clock (1 core)</th>
<th># Cores</th>
<th>DDR Type, Size, Data Width</th>
<th>Price (nonacademic/academic)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xilinx VC707 Virtex-7 XC7VX485T-2FFG1761C</td>
<td>67 MHz</td>
<td>4</td>
<td>DDR3 1 GB 64 bits</td>
<td>$3,495</td>
</tr>
<tr>
<td>Digilent Genesys2 Kintex-7 XC7K325T-2FFG900C</td>
<td>60 MHz</td>
<td>2</td>
<td>DDR3 1 GB 32 bits</td>
<td>$1,299/ $600</td>
</tr>
<tr>
<td>Digilent Nexys Video Artix-7 XC7A200T-1SBG484C</td>
<td>29 MHz</td>
<td>1</td>
<td>DDR3 512MB 16 bits</td>
<td>$490/ $250</td>
</tr>
<tr>
<td>Digilent Nexys 4 DDR Artix-7 XC7A100T-ACSG324C</td>
<td>29 MHz</td>
<td>1</td>
<td>DDR2 128MiB 16 bits</td>
<td>$320/ $160</td>
</tr>
</tbody>
</table>
I/O Interfaces
I/O Interfaces

Wishbone SD Master:
- Up to 2GB SD (microSD) cards
- Storage for boot/OS/tests
- Optional

UART:
- Terminal I/O
- Loading of assembly test

DDR controller:
- Xilinx’s MIG
- Configurable data width
- Used as main memory
- Optional
Demo
Recommended Configurations

BRAM with assembly test
• fast verification of architecture
• directed tests
• main memory emulated in BRAMs

BRAM with OpenBoot PROM (OBP)
• verifies basic I/O operations
• main memory emulated in BRAMs

SD with OS image
• full power of SW stack
• performance tests with benchmarks
# Recommended Configurations

<table>
<thead>
<tr>
<th></th>
<th>SD with OS</th>
<th>BRAM_TEST</th>
<th>BRAM_BOOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRAM with test + bram_map module</td>
<td></td>
<td>✔️</td>
<td></td>
</tr>
<tr>
<td>BRAM with OBP + bram_map_obp module</td>
<td></td>
<td></td>
<td>✔️</td>
</tr>
<tr>
<td>DDR memory controller</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>SD card controller</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>UART 16550</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>
Tools

• **Source located at** piton/tools/src/proto/

• **protosyn**
  – All encompassing tool

• **make_mem_map**

• **image2stream**
Protosyn Flow

Legend
- Control Flow
- Data Flow
- pyv preprocessor
- Sims script
- Make_mem_map script
- Vivado
- input/output files
- flow step conditions

Project creation

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?

Implement?

Synthesis

Mapping, placing, routing, bitstream generation, STA

Create project?
Running FPGA Flow

```

-b, --board <board_type>
   Name of a supported Xilinx's development board. Available options are:
   nexys4ddr*
   vc707
   genesys2
   nexysVideo

   * current configuration of design doesn't fit on this board

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

--bram-test <test_name>
   Name of the test to be mapped into BRAM

--bram-boot
   Implement design with OpenBoot mapped to a BRAM

--no-ddr
   Implement design without DDR memory

--uart-dmw
   Implement design with Direct Memory Write (DMW) from UART module turned on

--asic-rtl
   Implement design with ASIC RTL.

--chip-bridge
   Implement full system including the chip bridge

--inc-passthru
   Include passthru between chip and chipset (requires --chip-bridge)

--make-mem-map
   Create a mapping of test specified by --bram-test option to a BRAM

--from <step>
   Start FPGA flow from a specified step. Available options are:
   project
   impl

--to <step>
   Run FPGA flow to a specified step (including). Available options are:
   project
   impl

--slurm
   Run steps of flow using SLURM job scheduler
```
Running FPGA Flow


-b, --board <board_type>
Name of a supported Xilinx’s development board. Available options are:
- nexys4ddr
- vc707
- genesys2
- nexysVideo

* current configuration of design doesn’t fit on this board

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

--bram-test <test_name>
Name of the test to be mapped into BRAM

--bram-boot
Implement design with OpenBoot mapped to a BRAM

--no-ddr
Implement design without DDR memory

--uart-dmw
Implement design with Direct Memory Write (DMW) from UART module turned on

--asic-rtl
Implement design with ASIC RTL.

--chip-bridge
Implement full system including the chip bridge

--inc-passthru
Include passthru between chip and chipset (requires --chip-bridge)

--make-mem-map
Create a mapping of test specified by --bram-test option to a BRAM

--from <step>
Start FPGA flow from a specified step. Available options are:
- project
- impl

--to <step>
Run FPGA flow to a specified step (including). Available options are:
- project
- impl

--slurm
Run steps of flow using SLURM job scheduler
Running FPGA Flow


- b, --board <board_type>
   Name of a supported Xilinx's development board. Available options are:
   nexys4ddr
   vc707
   genesys2
   nexysVideo

   * current configuration of design doesn't fit on this board

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

--bram-test <test_name>
   Name of the test to be mapped into BRAM

--bram-boot
   Implement design with OpenBoot mapped to a BRAM

--no-ddr
   Implement design without DDR memory

--uart-dmw
   Implement design with Direct Memory Write (DMW) from UART module turned on

--asic-rtl
   Implement design with ASIC RTL.

--chip-bridge
   Implement full system including the chip bridge

--inc-passthru
   Include passthru between chip and chipset (requires --chip-bridge)

--make-mem-map
   Create a mapping of test specified by --bram-test option to a BRAM

--from <step>
   Start FPGA flow from a specified step. Available options are:
   project
   impl

--to <step>
   Run FPGA flow to a specified step (including). Available options are:
   project
   impl

--slurm
   Run steps of flow using SLURM job scheduler
Running FPGA Flow

```

-b, --board <board_type>
  Name of a supported Xilinx's development board. Available options are:
  nexys4ddr*
  vc707
  genesys2
  nexysVideo

# Format:
<table>
<thead>
<tr>
<th>BlockID</th>
<th>BlockPath</th>
<th>Supported Board, Frequency(MHz), DDRSize(Mbytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>system</td>
<td>.</td>
<td>vc707, 67, 1024; genesys2, 50, 1024; nexysVideo, 30, 512</td>
</tr>
<tr>
<td>chipset</td>
<td>.</td>
<td>genesys2, 50, 1024</td>
</tr>
<tr>
<td>chip</td>
<td>.</td>
<td>genesys2, 50, 1024</td>
</tr>
</tbody>
</table>

Name of the test to be mapped into BRAM

--bram-boot
  Implement design with OpenBoot mapped to a BRAM

--no-ddr
  Implement design without DDR memory

--uart-dmw
  Implement design with Direct Memory Write (DMW) from UART module turned on

--asic-rtl
  Implement design with ASIC RTL.

--chip-bridge
  Implement full system including the chip bridge

--inc-passthru
  Include passthru between chip and chipset (requires --chip-bridge)

--make-mem-map
  Create a mapping of test specified by --bram-test option to a BRAM

--from <step>
  Start FPGA flow from a specified step. Available options are:
  project
  impl

--to <step>
  Run FPGA flow to a specified step (including). Available options are:
  project
  impl

--slurm
  Run steps of flow using SLURM job scheduler
```

`piton/tools/src/proto/block.list`
Running FPGA Flow


-b, --board <board_type>
Name of a supported Xilinx's development board. Available options are:
nexys4ddr*
vc707
genesys2
nexysVideo

* current configuration of design doesn’t fit on this board
-d, --design <design>
Name of design module to synthesize. The default is 'system', which synthesizes a full system with chip and chipset. See $PITON_ROOT/tools/src/proto/block.list for supported design modules

--bram-test <test_name>
Name of the test to be mapped into BRAM

--bram-boot
Implement design with OpenBoot mapped to a BRAM

--no-ddr
Implement design without DDR memory

--uart-dmw
Implement design with Direct Memory Write (DMW) from UART module turned on

--asic-rtl
Implement design with ASIC RTL.

--chip-bridge
Implement full system including the chip bridge

--inc-passthru
Include passthru between chip and chipset (requires --chip-bridge)

--make-mem-map
Create a mapping of test specified by --bram-test option to a BRAM

--from <step>
Start FPGA flow from a specified step. Available options are:
project
impl

--to <step>
Run FPGA flow to a specified step (including). Available options are:
project
impl

--slurm
Run steps of flow using SLURM job scheduler
Running FPGA Flow


-b, --board <board_type>
   Name of a supported Xilinx's development board. Available options are:
   nexys4ddr*
   vc707
   genesys2
   nexsysVideo

   * current configuration of design doesn't fit on this board
-d, --design <design>
   Name of design module to synthesize. The default is 'system', which
   synthesizes a full system with chip and chipset. See
   $PITON_ROOT/tools/src/proto/block.list for supported design modules

--bram-test <test_name>
   Name of the test to be mapped into BRAM

--bram-boot
   Implement design with OpenBoot mapped to a BRAM

--no-ddr
   Implement design without DDR memory

--uart-dmw
   Implement design with Direct Memory Write (DMW) from UART module turned on

--asic-rtl
   Implement design with ASIC RTL.

--chip-bridge
   Implement full system including the chip bridge

--inc-passthru
   Include passthru between chip and chipset (requires --chip-bridge)

--make-mem-map
   Create a mapping of test specified by --bram-test option to a BRAM

--from <step>
   Start FPGA flow from a specified step. Available options are:
   project
   impl

--to <step>
   Run FPGA flow to a specified step (including). Available options are:
   project
   impl

--slurm
   Run steps of flow using SLURM job scheduler
Running FPGA Flow


-b, --board <board_type>
Name of a supported Xilinx's development board. Available options are:
  nexys4ddr*
  vc707
  genesys2
  nexysVideo

* current configuration of design doesn't fit on this board
-d, --design <design>
Name of design module to synthesize. The default is 'system', which
synthesizes a full system with chip and chipset. See
$PITON_ROOT/tools/src/proto/block.list for supported design modules

--bram-test <test_name>
Name of the test to be mapped into BRAM

--bram-boot
Implement design with OpenBoot mapped to a BRAM

--no-ddr
Implement design without DDR memory

--uart-dmw
Implement design with Direct Memory Write (DMW) from UART module turned on

--asic-rtl
Implement design with ASIC RTL.

--chip-bridge
Implement full system including the chip bridge

--inc-passthr
Include passthr between chip and chipset (requires --chip-bridge)

--make-mem-map
Create a mapping of test specified by --bram-test option to a BRAM

--from <step>
Start FPGA flow from a specified step. Available options are:
  project
  impl

--to <step>
Run FPGA flow to a specified step (including). Available options are:
  project
  impl

--slurm
Run steps of flow using SLURM job scheduler
Running FPGA Flow


-b, --board <board_type>
Name of a supported Xilinx’s development board. Available options are:

nexys4ddr
vc707
genesis2
nexysVideo

* current configuration of design doesn’t fit on this board

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

--bram-test <test_name>
Name of the test to be mapped into BRAM

--bram-boot
Implement design with OpenBoot mapped to a BRAM

--no-ddr
Implement design without DDR memory

--uart-dmw
Implement design with Direct Memory Write (DMW) from UART module turned on

--asic-rtl
Implement design with ASIC RTL.

--chip-bridge
Implement full system including the chip bridge

--inc-passthru
Include passthru between chip and chipset (requires --chip-bridge)

--make-mem-map
Create a mapping of test specified by --bram-test option to a BRAM

--from <step>
Start FPGA flow from a specified step. Available options are:

project
impl

--to <step>
Run FPGA flow to a specified step (including). Available options are:

project
impl

--slurm
Run steps of flow using SLURM job scheduler
FPGA Flow Hello World Example

[openpiton]alavrov@della2$ pwd
/tigress/alavrov/chip/openpiton
[openpiton]alavrov@della2$ protosyn -b genesys2 --bram-test uart16550-hello-world.s
Generating .tmp.v files
Synthesizing a test: uart16550-hello-world.s
Compilation started
Simulation started
Using core clock frequency: 50 MHz
Test Passed!
Starting mapping of a test
Length of image file: 44423
Automatically created 19 sections for test in bram memory
Checking correctness of section mapping...
Correct!
Used 96 out of 256 rows of BRAM
Creating project for design 'system' on board 'genesys2'
Running FPGA implementation down to bitstream generation
Implementation finished!

Protosyn finished!
[openpiton]alavrov@della2$
FPGA Flow Hello World Example

[openpiton]alavrov@della2$ pwd
/tigress/alavrov/chip/openpiton
[openpiton]alavrov@della2$ protosyn -b genesys2 --bram-test uart16550-hello-world.s
Generating .tmp.v files
Synthesizing a test: uart16550-hello-world.s
Compilation started
Simulation started
Using core clock frequency: 50 MHz
Test Passed!
Starting mapping of a test
Length of image file: 44423
Automatically created 19 sections for test in bram memory
Checking correctness of section mapping...
Correct!
Used 96 out of 256 rows of BRAM
Creating project for design 'system' on board 'genesys2'
Running FPGA implementation down to bitstream generation
Implementation finished!

Protosyn finished!
[openpiton]alavrov@della2$
FPGA Flow Hello World Example

[openpiton]alavrov@della2$ pwd
/tigress/alavrov/chip/openpiton

[openpiton]alavrov@della2$ protosyn -b genesys2 --bram-test uart16550-hello-world.s
Generating .tmp.v files

Synthesizing a test: uart16550-hello-world.s
Compilation started
Simulation started
Using core clock frequency: 50 MHz
Test Passed!
Starting mapping of a test
Length of image file: 44423
Automatically created 19 sections for test in bram memory
Checking correctness of section mapping...
Correct!
Used 96 out of 256 rows of BRAM
Creating project for design 'system' on board 'genesys2'
Running FPGA implementation down to bitstream generation
Implementation finished!

Protosyn finished!

[openpiton]alavrov@della2$
FPGA Flow Hello World Example

[openpiton]alavrov@della2$ pwd
/tigress/alavrov/chip/openpiton
[openpiton]alavrov@della2$ protosyn -b genesys2 --bram-test uart16550-hello-world.s
Generating .tmp.v files

Synthesizing a test: uart16550-hello-world.s
Compilation started
Simulation started
Using core clock frequency: 50 MHz
Test Passed!
Starting mapping of a test
Length of image file: 44423
Automatically created 19 sections for test in bram memory
Checking correctness of section mapping...
Correct!
Used 96 out of 256 rows of BRAM
Creating project for design 'system' on board 'genesys2'
Running FPGA implementation down to bitstream generation
Implementation finished!

Protosyn finished!
[openpiton]alavrov@della2$
FPGA Flow Hello World Example

```
[openpiton]alavrov@della2$ pwd
/tigress/alavrov/chip/openpiton
[openpiton]alavrov@della2$ protosyn -b genesys2 --bram-test uart16550-hello-world.s
Generating .tmp.v files
Synthesizing a test: uart16550-hello-world.s
Compilation started
Simulation started
Using core clock frequency: 50 MHz
Test Passed!
Starting mapping of a test
Length of image file: 44423
Automatically created 19 sections for test in bram memory
Checking correctness of section mapping...
Correct!
Used 96 out of 256 rows of BRAM
Creating project for design 'system' on board 'genesys2'
Running FPGA implementation down to bitstream generation
Implementation finished!
Protosyn finished!
[openpiton]alavrov@della2$
```
FPGA Flow SD Example

[openpiton]alavrov@della2$ pwd
/tigress/alavrov/chip/openpiton
[openpiton]alavrov@della2$ protosyn -b genesys2
Generating .tmp.v files
Creating project for design 'system' on board 'genesys2'
Running FPGA implementation down to bitstream generation
Implementation finished!

Protosyn finished!
[openpiton]alavrov@della2$
FPGA Flow SD Example

```
[openpiton]alavrov@della2$ pwd
/tigress/alavrov/chip/openpiton
[openpiton]alavrov@della2$ protosyn -b genesys2
Generating .tmp.v files
Creating project for design 'system' on board 'genesys2'
Running FPGA implementation down to bitstream generation
Implementation finished!

Protosyn finished!
[openpiton]alavrov@della2$
```
FPGA Flow SD Example

[openpiton]alavrov@della2$ pwd
/tigress/alavrov/chip/openpiton
[openpiton]alavrov@della2$ protosyn -b genesys2
Generating .tmp.v files
Creating project for design 'system' on board 'genesys2'
Running FPGA implementation down to bitstream generation
Implementation finished!

Protosyn finished!
[openpiton]alavrov@della2$
FPGA Flow SD Example

[openpiton]alavrov@della2$ pwd
/tigress/alavrov/chip/openpiton
[openpiton]alavrov@della2$ protosyn -b genesys2
Generating .tmp.v files
Creating project for design 'system' on board 'genesys2'
Running FPGA implementation down to bitstream generation
Implementation finished!
Protosyn finished!
[openpiton]alavrov@della2$
FPGA Flow Runtimes

• System including DDR controller
  – ~1 hour including IP generation
  – ~20 mins excluding IP generation
FPGA Flow Outputs

```
$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/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.jou              vivado_18300.backup.log  webtalk_13970.backup.jou
vivado.log              vivado_26785.backup.log  webtalk_13970.backup.log
vivado_15396.backup.jou vivado_26785.backup.log  webtalk_15500.backup.jou
vivado_15396.backup.log webtalk.jou               webtalk_15500.backup.log
vivado_15433.backup.jou webtalk.log               webtalk_19091.backup.jou
vivado_15433.backup.log webtalk_10265.backup.jou  webtalk_19091.backup.log
```

FPGA Flow Outputs

```bash
[alavrov@della2 system]$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/system

[alavrov@della2 system]$ ls
additionalDefines.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.jou            vivado_18300.backup.log   webtalk_13970.backup.jou
vivado.log             vivado_26785.backup.log   webtalk_13970.backup.log
vivado_15396.backup.jou vivado_26785.backup.log   webtalk_15500.backup.jou
vivado_15396.backup.log webtalk.jou               webtalk_15500.backup.log
vivado_15433.backup.jou webtalk.log               webtalk_19091.backup.jou
vivado_15433.backup.log webtalk_10265.backup.jou  webtalk_19091.backup.log
```

FPGA Flow Outputs

```
[alavrov@della2 system]$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/system
[alavrov@della2 system]$ ls
additionalDefines.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.jou            vivado_18300.backup.log   webtalk_13970.backup.jou
vivado.log            vivado_26785.backup.log   webtalk_13970.backup.log
vivado_15396.backup.jou  vivado_26785.backup.log   webtalk_15500.backup.jou
vivado_15396.backup.log           webtalk.jou                webtalk_15500.backup.log
vivado_15433.backup.jou          webtalk.log                   webtalk_19091.backup.jou
vivado_15433.backup.log          webtalk_10265.backup.jou  webtalk_19091.backup.log
[alavrov@della2 system]$ 
```
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
genesis2_system       vivado_18131.backup.log  webtalk_12480.backup.jou
protosyn_logs          vivado_18300.backup.jou  webtalk_12480.backup.log
vivado_jou             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
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
genesis2_system       vivado_18131.backup.log   webtalk_12480.backup.jou
protosyn_logs          vivado_18300.backup.jou   webtalk_12480.backup.log
vivado.jou             vivado_18300_backup_log  webtalk_13970_backup_log

[alavrov@della2 protosyn_logs]$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/system/protosyn_logs
[alavrov@della2 protosyn_logs]$ ls
implementation.log  make_project.log
```

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.jou              vivado_18300.backup.log  webtalk_13970.backup.jou
vivado.log              vivado_26785.backup.log  webtalk_13970.backup.log
vivado_15396.backup.jou vivado_26785.backup.log  webtalk_15500.backup.jou
vivado_15396.backup.log webtalk.jou                webtalk_15500.backup.log
vivado_15433.backup.jou webtalk.log                webtalk_19091.backup.jou
vivado_15433.backup.log webtalk_10265.backup.jou   webtalk_19091.backup.log
```

FPGA Flow Outputs

```
[alavrov@della2 genesys2_system]$ pwd
/tigress/alavrov/chip/openpiton/build/genesys2/system/genesys2_system
[alavrov@della2 genesys2_system]$ ls
genesis2_system.cache genesis2_system.runs vivado.log
genesis2_system.hw genesis2_system.xpr
genesis2_system.ip_user_files vivado.jou
[alavrov@della2 genesys2_system]$ 
```
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.xpr
 genesys2_system.hw
 genesys2_system.ip_user_files vivado.jou
[alavrov@della2 genesys2_system]$ 
```
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
genesis2_system.hw     genesys2_system.xpr
genesys2_system.ip_user_files  vivado.jou
[alavrov@della2 genesys2_system]$
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.hw
genesys2_system.ip_user_files
genesys2_system.runs
vivado.log
vivado.jou

[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/openpitoon/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
system_power_routed.rpt
system_power_summary_routed.pb
system_route_status.bp
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
```
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]$ ls
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
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]$
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
oxpt_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]$ ```
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

```
[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

```
[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
Opening FPGA Design
Opening FPGA Design
Opening FPGA Design
Writing OS Image to SD Card
Writing OS Image to SD Card
Writing OS Image to SD Card
Writing OS Image to SD Card

![Win32 Disk Imager](image)
Writing OS Image to SD Card
Hands-on with FPGA
FPGA Programming
FPGA Programming
FPGA Programming
FPGA Programming
FPGA Linux Boot

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

Sun Fire T1000, No Keyboard
Copyright 2007 Sun Microsystems, Inc. All rights reserved.
OpenBoot 4.8.x.build_122***PROTOTYPE BUILD***, 1000 MB memory available, Serial #66711024.
[greddy obp #0]
Ethernet address 0:e0:81:5f:2c:ab, Host ID: 83f9edf0.

ok
FPGA Linux Boot

Alive and well ...
Strand start set = 0x1
Total physical mem = 0x40000000
Scrubbing the rest of memory
Number of strands = 0x1
membase = 0x0
memsize = 0x10000000
phymem = 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

Sun Fire T1000, No Keyboard
Copyright 2007 Sun Microsystems, Inc. All rights reserved.
OpenBoot 4.x.build_122***PROTOTYPE BUILD***, 1000 MB memory available, Serial #66711024.
[greddy obp #0]
Ethernet address 0:e0:81:5f:2c:ab, Host ID: 83f9edf0.
FPGA Linux Boot

ok boot Linux
Boot device: /virtual-devices/disk@0  File and args: Linux
SILO Version 1.4.13

Allocated 8 Megs of memory at 0x40000000 for kernel

/
FPGA Linux Boot

Allocated 8 Megs of memory at 0x40000000 for kernel
Loaded kernel version 4.1.0

[ 0.000000] PROMLIB: Sun IEEE Boot Prom 'OBP 4.x.build_122***PROTOTYPE BUILD*** 2009/03/18 13:50'
[ 0.000000] PROMLIB: Root node compatible: sun4v
[ 0.000000] Linux version 4.1.0-openpilton+ (mmatl@cloud-t2000) (gcc version 4.6.3 (Debian 4.6.3-14) ) #1 SMP Sat Apr 2 22:38:48 EDT 2016
[ 0.000000] bootconsole [earlyprom0] enabled
[ 0.000000] ARCH: SUN4V
[ 0.000000] Ethernet address: 00:e0:81:5f:2c:ab
[ 0.000000] MM: PAGE OFFSET is 0xffffffff0000000000 (max_phys_bits == 39)
[ 0.000000] MM: VMAALLOC [0x0000000000000000 -- 0x0000000000000000] --> 0x00000000000000000000000000000000
[ 0.000000] MM: VMEMMAP [0x0000000000000000 -- 0x00000000000000000000000000000000]
[ 0.000000] Kernel: Using 2 locked TLB entries for main kernel image.
[ 0.000000] Remapping the kernel...
[ 0.000000] done.
[ 0.000000] OF stdout device is: /virtual-devices@100/console@01
[ 0.000000] PROM: Built device tree with 29726 bytes of memory.
[ 0.000000] MDESC: Size is 8704 bytes.
[ 0.000000] PLATFORM: banner-name [Sun Fire T1000]
[ 0.000000] PLATFORM: name [SUNW,Sun-Fire-T1000]
[ 0.000000] PLATFORM: hostid [83f9edf0]
[ 0.000000] PLATFORM: serial# [00ab4130]
[ 0.000000] PLATFORM: stick-frequency [02faf080]
[ 0.000000] PLATFORM: mac-address [e081f2cab]
[ 0.000000] PLATFORM: watchdog-resolution [1000 ms]
[ 0.000000] PLATFORM: watchdog-max-timeout [3153600000 ms]
[ 0.000000] PLATFORM: max-cpus [32]
[ 0.000000] Top of RAM: 0x3ff0a000, Total RAM: 0x3eb08000
[ 0.000000] Memory hole size: 20MB
FPGA Linux Boot

COM3 - PUTTY

[ 224.910198] Testing NMI watchdog ... OK.
[ 224.991523] Supported PMU type is 'niagara'
[ 225.009955] clocksource jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns: 7645041785100000 ns
[ 225.159223] VIO: Adding device channel-devices
[ 225.166612] VIO: Adding device vldc-port-0-0
[ 225.172457] VIO: Adding device vldc-port-0-1
[ 225.618634] Switched to clocksource stick
[ 226.124675] futex hash table entries: 8192 (order: 6, 524288 bytes)
[ 226.200150] HugeTLB registered 8 MB page size, pre-allocated 0 pages
[ 226.886799] fuse init (API version 7.23)
[ 227.447101] Block layer SCSI generic (bsg) driver version 0.4 loaded (major 253)
[ 227.450415] io scheduler noop registered
[ 227.472866] io scheduler cfq registered (default)
[ 229.452124] HDLC line discipline maxframe=4096
[ 229.454444] N_HDLC line discipline registered.
[ 229.463973] f026b770: ttyS0 at I/O 0x0 (irq = 1, base_baud = 115200) is a SUN4V HCONS
[ 229.988421] console [ttyHV0] enabled
[ 230.744884] brd: module loaded
[ 231.073423] loop: module loaded
[ 231.088368] sunhv_disk:v1.0 Mar 20, 2009
[ 231.165238] mousedev: PS/2 mouse device common for all mice
[ 231.251899] rtc-sun4v rtc-sun4v: rtc core: registered sun4v as rtc0
[ 231.275242] rtc rtc1: invalid alarm value: 1900-1-2 2022213768:1073741856:0
[ 231.291172] rtc-test rtc-test.0: rtc core: registered test as rtc1
[ 231.307508] rtc rtc2: invalid alarm value: 1900-1-2 2022213768:1073741856:0
[ 231.323392] rtc-test rtc-test.1: rtc core: registered test as rtc2
[ 231.420201] rtc-sun4v rtc-sun4v: setting system clock to 1970-01-01 00:00:00 UTC (0)
[ 231.521132] kjournald starting. Commit interval 5 seconds
[ 231.531618] EXT3-fs (sunhv_disk): mounted filesystem with ordered data mode
[ 231.543047] VFS: Mounted root (ext3 filesystem) readonly on device 253:0.
FPGA Linux Boot

```plaintext
COM3 - PUTTY

[ 229.988421] console [ttyHVO] enabled
[ 230.744884] brd: module loaded
[ 231.073423] loop: module loaded
[ 231.088368] sunhv_disk:vl.0 Mar 20, 2009
[ 231.165238] mousedev: PS/2 mouse device common for all mice
[ 231.251899] rtc-sun4v rtc-sun4v: rtc core: registered sun4v as rtc0
[ 231.275242] rtc rtc1: invalid alarm value: 1900-1-2 2022213768:1073741856:0
[ 231.291172] rtc-test rtc-test.0: rtc core: registered test as rtc1
[ 231.307508] rtc rtc2: invalid alarm value: 1900-1-2 2022213768:1073741856:0
[ 231.323392] rtc-test rtc-test.1: rtc core: registered test as rtc2
[ 231.420201] rtc-sun4v rtc-sun4v: setting system clock to 1970-01-01 00:00:00 UTC (0)
[ 231.521132] kjournald starting. Commit interval 5 seconds
[ 231.531618] EXT3-fs (sunhv_disk): mounted filesystem with ordered data mode
[ 231.543047] VFS: Mounted root (ext3 filesystem) readonly on device 253:0.
INIT: version 2.68 booting
[ ok ] Activating swap...done.
[ ok ] Checking root file system...fsck from util-linux 2.20.1
/run/rootdev: clean, 10586/61440 files, 188286/245760 blocks
done.
[ 348.408009] EXT3-fs (sunhv_disk): using internal journal
[ ok ] Cleaning up temporary files.../tmp.
[ ok ] Activating lvm and md swap...done.
[ ok ] Checking file systems...fsck from util-linux 2.20.1
done.
[ ok ] Mounting local filesystems...done.
[ ok ] Activating swapfile swap...done.
[ ok ] Cleaning up temporary files....
[ 593.552118] random: dd urandom read with 0 bits of entropy available
INIT: Entering runlevel: 2

Debian GNU/Linux 7 openpitol-fpga ttyS0
openpitol-fpga login:  
```
FPGA Linux Boot

[ 231.531618] EXT3-fs (sunhv_disk): mounted filesystem with ordered data mode
[ 231.543047] VFS: Mounted root (ext3 filesystem) readonly on device 253:0.
INIT: version 2.88 booting
[ ok ] Activating swap...done.
[....] Checking root file system...fsck from util-linux 2.20.1
/run/rootdev: clean, 10586/61440 files, 188286/245760 blocks
done.
[ 348.408009] EXT3-fs (sunhv_disk): using internal journal
[ ok ] Cleaning up temporary files.../tmp.
[ ok ] Activating lvm and md swap...done.
[....] Checking file systems...fsck from util-linux 2.20.1
done.
[ ok ] Mounting local filesystems...done.
[ ok ] Activating swapfile swap...done.
[ ok ] Cleaning up temporary files....
[ 593.552118] random: dd urandom read with 0 bits of entropy available
INIT: Entering runlevel: 2

Debian GNU/Linux 7 openpiton-fpga ttyS0

openpiton-fpga login: root
Password:
Linux openpiton-fpga 4.1.0-openpiton+ #11 SMP Sat Apr 2 22:38:48 EDT 2016 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@openpiton-fpga:~# date
Thu Jan 1 00:09:48 UTC 1970
root@openpiton-fpga:~#
FPGA Linux Boot

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