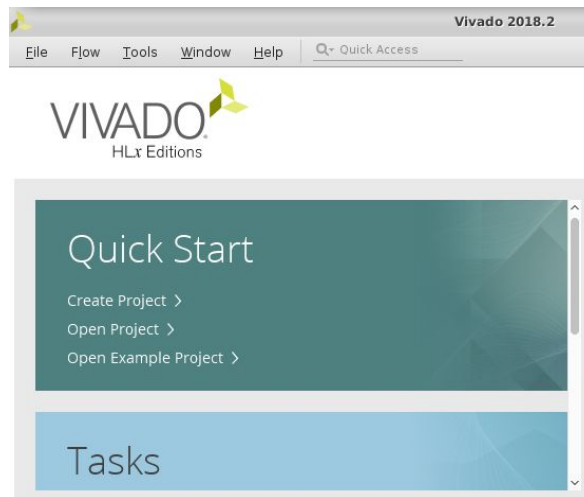


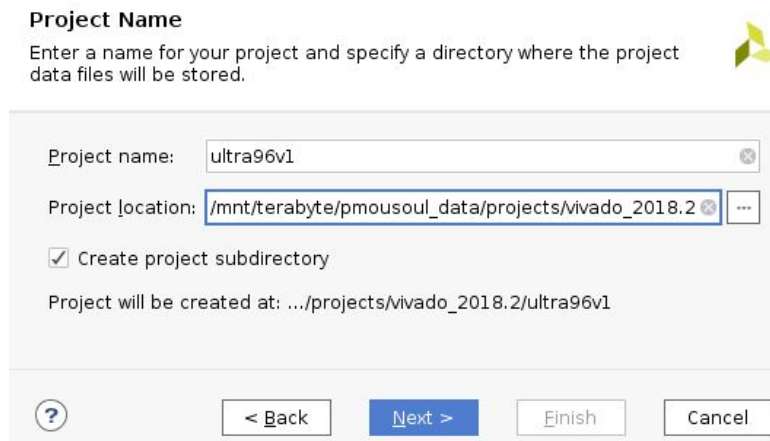
Platform Creation for Ultra96 v1.2 board using Xilinx Vivado, PetaLinux, and SDSoc

1. Create new Vivado project

Open Vivado IDE and select **Create Project** in Quick Start.



Click **Next**. In the New Project window, enter the name of the project (e.g. use **ultra96v1**) and select the location to be saved on the hard disk.



In the Project Type window, select **RTL Project**, select **Do not specify sources at this time**, and click on **Next** to create a new project using IP blocks (IP Integrator) without selecting any predefined source files.

Project Type

Specify the type of project to create.



RTL Project

You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.



Do not specify sources at this time



Post-synthesis Project: You will be able to add sources, view device resources, run design analysis, planning and implementation.



< Back

Next >

Finish

Cancel

In the Default Part window, select **Boards**, and type **ultra96v1** in the search field. Select the **Ultra96v1 Evaluation Platform**.

Default Part

Choose a default Xilinx part or board for your project. This can be changed later.



Parts | **Boards**

[Reset All Filters](#)

Vendor: **All** Name: **All** Board Rev: **Latest**

Search: **ultra96v1** (1 match)

Display Name	Preview	Vendor	File Ver...	Part	I/O Pir
Ultra96v1 Evaluation Platform		em.avnet.com	1.2	xczu3eg-sbva484-1-e	484

< >

< Back Next > Finish Cancel

The New Project Summary window should contain the same information as below.



New Project Summary

- A new RTL project named 'ultra96v1' will be created.
- The default part and product family for the new project:
Default Board: Ultra96v1 Evaluation Platform
Default Part: xczu3eg-sbva484-1-e
Product: Zynq UltraScale+
Family: Zynq UltraScale+ MPSoCs
Package: sbva484
Speed Grade: -1

To create the project, click Finish



< Back

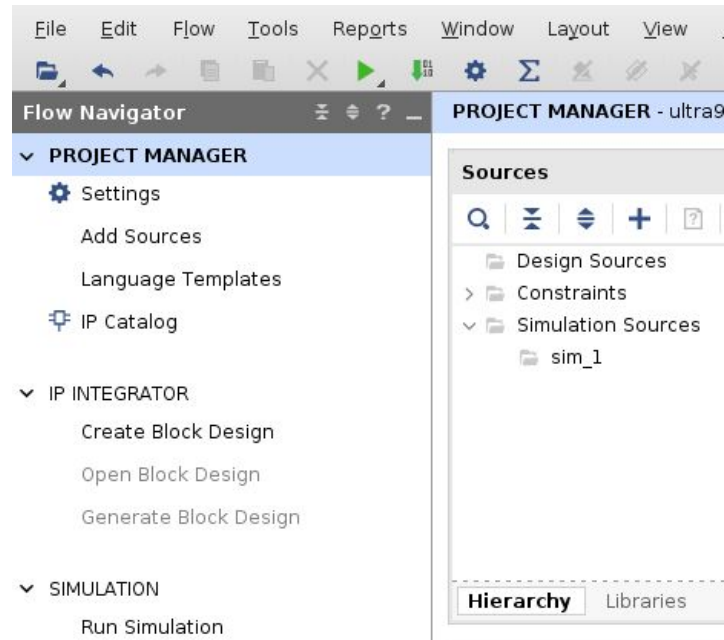
Next >

Finish

Cancel

2. Create design using IP blocks

Goto **Flow Navigator**, expand **IP INTEGRATOR** and select **Create Block Design**.



In the window that opens, we type the name of our design.

CATION: The design name we enter here must be the same as the SDSoc platform name that we will create later.

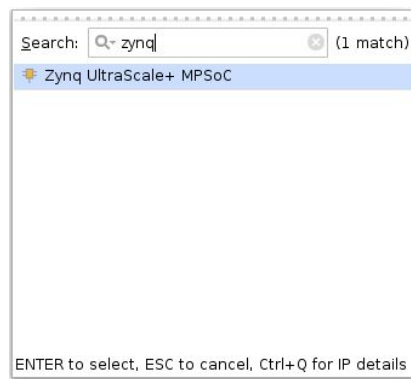
Enter **ultra96v1** and click **OK**.



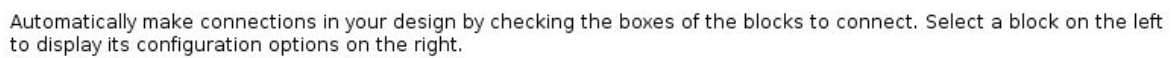
The following table gives a brief description of the IP blocks that will be included in our design.

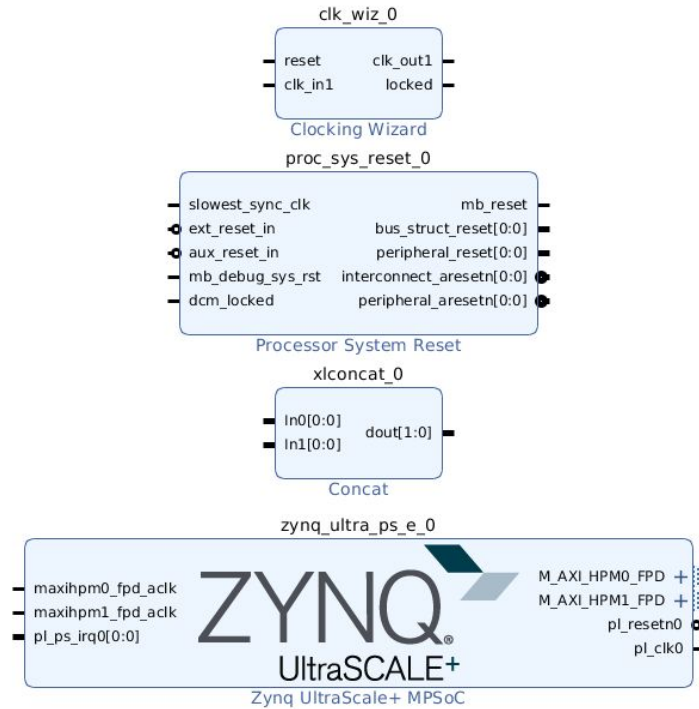
IP Block	Usage Summary
Zynq UltraScale+ MPSoC Processor System (PS)	<ul style="list-style-type: none"> • Dual-core Arm processor with cache hierarchy • Integrated I/O peripherals • DDR memory controller with external memory interface • PS to Programmable Logic (PL) interconnects • PL to PS interconnects
Processor System Reset Block (PL)	<ul style="list-style-type: none"> • Reset sequencing and synchronization block for PL logic
Clocking Wizard (PL)	<ul style="list-style-type: none"> • Multiple output clock generator to drive PL logic
Concat Block (PL)	<ul style="list-style-type: none"> • PL interrupt structure that feeds Zynq UltraScale+ MPSoC PS interrupt request input

In the **Diagram** section of the BLOCK DESIGN sub-window, we click on the **+** symbol to add IP blocks to our design. Or we can right-click in the **Diagram** section and select **Add IP**. We add a **Zynq UltraScale+ MPSoC** IP in our design.



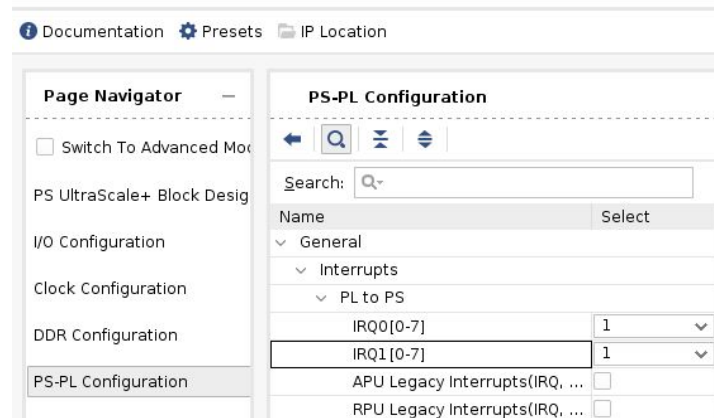
Click on the **Run Block Automation** of the **Designer Assistance** to **Apply Board Preset**. Click **OK**.





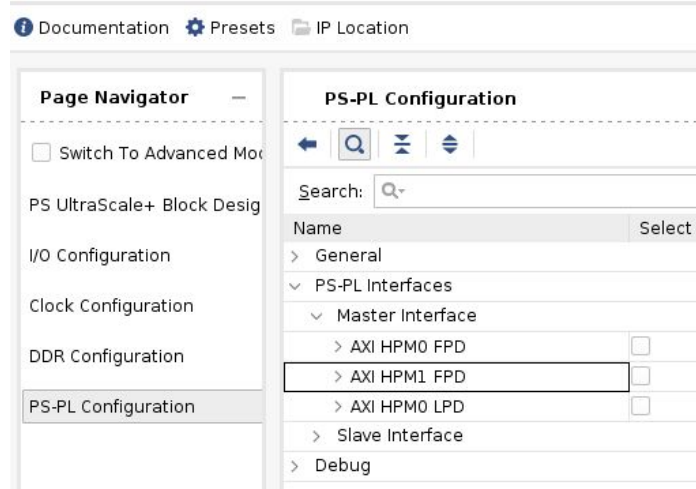
We configure the **Zynq UltraScale+ MPSoC** IP block by double-clicking on it. To enable interrupts from PL to PS we select **PS-PL Configuration** in the **Page Navigator** section of the **Re-customize IP** and we expand at the right of the **Page Navigator** the **General** -> **Interrupts** -> **PL to PS** and we enter **1** in **IRQ0[0-7]** and **IRQ1[0-7]**.

Zynq UltraScale+ MPSoC (3,2)



We select again the **PS-PL Configuration** in the **Page Navigator** section of the **Re-customize IP** window and we expand at the right of the **Page Navigator** the **PS-PL Interfaces** -> **Master Interface** and we de-select the **AXI HPM0 FPD** and **AXI HPM1 FPD**. We click on **OK** to close the **Re-customize IP** window.

Zynq UltraScale+ MPSoC (3.2)



We double-click the **Clocking Wizard** IP. Make sure that in the **Output Clocks** tab the **clk_out1** signal has the value 100MHz. Also make sure that the **Reset Type** signal is selected as **Active Low**. Click **OK** to close the **Re-customize IP** window.

Component Name

Clk Options	Output Clocks	MMCM Settings	Summary			
<input checked="" type="checkbox"/> clk_out1	clk_out1 <input type="button" value="X"/>	100.000 <input type="button" value="X"/>	100.000	0.000 <input type="button" value="X"/>	0.000	50.000 <input type="button" value="X"/>
<input type="checkbox"/> clk_out2	clk_out2	100.000	N/A	0.000	N/A	50.000
<input type="checkbox"/> clk_out3	clk_out3	100.000	N/A	0.000	N/A	50.000
<input type="checkbox"/> clk_out4	clk_out4	100.000	N/A	0.000	N/A	50.000
<input type="checkbox"/> clk_out5	clk_out5	100.000	N/A	0.000	N/A	50.000
<input type="checkbox"/> clk_out6	clk_out6	100.000	N/A	0.000	N/A	50.000
<input type="checkbox"/> clk_out7	clk_out7	100.000	N/A	0.000	N/A	50.000

☐ USE CLOCK SEQUENCING

Output Clock	Sequence Number
clk_out1	1
clk_out2	1
clk_out3	1
clk_out4	1
clk_out5	1
clk_out6	1
clk_out7	1

Enable Optional Inputs / Outputs for MMCM/PLL
☒ reset
 ☐ power_down
 ☐ input_clk_stopped

Reset Type
☐ Active High
 ☒ Active Low

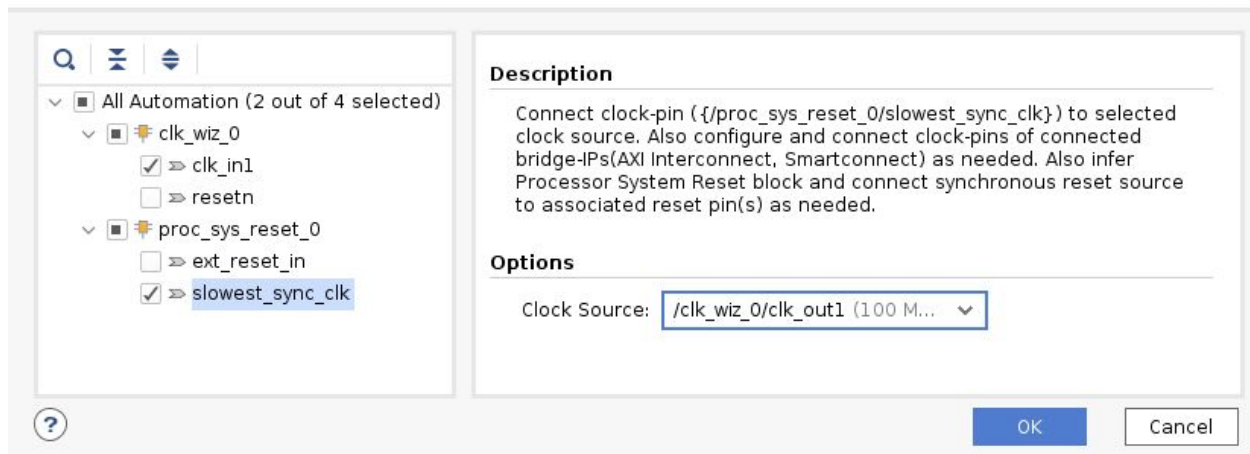
Phase Shift Mode
☐ WAVEFORM
 ☒ LATENCY

☒ locked
 ☐ clkfbstopped

Double click the **Concat** IP and set a value equal to **1** in the **Number of Ports** field. Click **OK** to close the **Re-customize IP** window.

Select the **Run Connection Automation** of the **Designer Assistance** to connect the **Clocking Wizard** and the **Processor System Reset** IP blocks with the **Zynq UltraScale+ MPSoC** IP block. The **clk_in1** port of the **Clocking Wizard** IP must be connected to the **pl_clk0** port of the PS and the **slowest_sync_clk** setting port of the **Processor System Reset** IP must be connected to the **clk_out1** port of the **Clocking Wizard** IP.

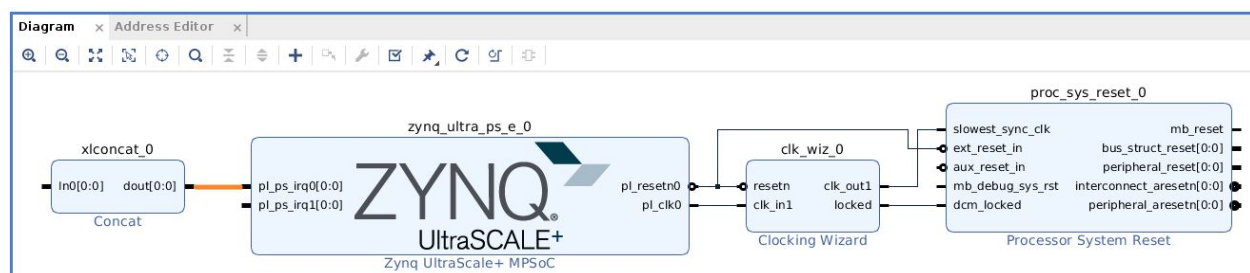
Automatically make connections in your design by checking the boxes of the interfaces to connect. Select an interface on the left to display its configuration options on the right.



We make “by-hand” the rest of the connections - using “drag-n-drop” beginning from the port of one IP block and ending to the port of another IP block. We must make the following connections:

- Connect the **pl_resetsn0** port of **Zynq UltraScale+ MPSoC** IP to the **resetsn** port of the **Clocking Wizard** IP
- Connect the **pl_resetsn0** port of **Zynq UltraScale+ MPSoC** IP to the **ext_reset_in** port of the **Processor System Reset** IP
- Connect the **locked** port of **Clocking Wizard** IP to the **dcm_locked** port of the **Processor System Reset** IP
- Connect the **dout[0:0]** port of the **Concat** IP to the **pl_ps_irq0[0:0]** port of the **Zynq UltraScale+ MPSoC** IP

Clicking on the **Regenerate Output** our design must look like the following diagram.

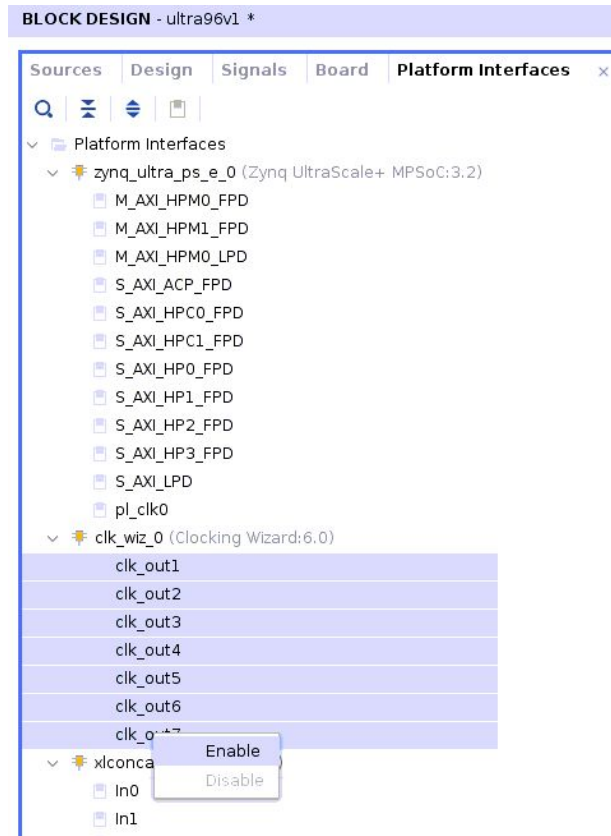


Finally, we click on the **Validate Design** to ensure that our design is error-free.

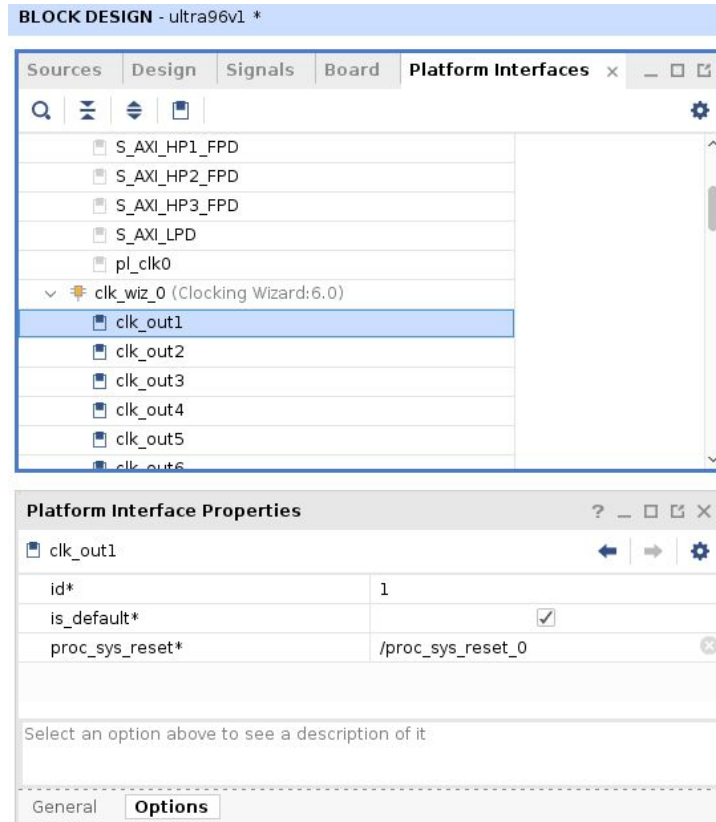
3. Setting the hardware interfaces

We select **Window -> Platform Interfaces** and click on **Enable platform interfaces**.

The **Platform Interfaces** section shows all the port that could be available to our platform. We can choose the ports we would like to include by right-clicking on them and selecting **Enable**.



Using the section **Platform Interface Properties** we can set the default clock signal for use by the SDSoC. We select **clk_out1** (100MHz) and in the **Platform Interface Properties** we select **is_default***.



Enable all the AXI interfaces except **pl_clk0**, **S_AXI_ACP_FPD**, and **S_AXI_LPD**.

Finally, enable the interrupt ports 0 to 7 of the **Concat** IP.

4. HDL and DSA file generation

In the **Sources** section, we right-click the **ultra96v1** (**ultra96v1.bd**) and select **Generate Output Products**. In the window that opens we select **Generate**.

The following output products will be generated.



Preview

Q [] [] []

▼ ultra96v1.bd (OOC per IP)

- Synthesis
- Implementation
- Simulation

Synthesis Options

☐ Global

☒ Out of context per IP

☐ Out of context per Block Design

Run Settings

☒ On local host: Number of jobs: 8 ▼

☐ On remote hosts:

☐ Use LSF:

After the previous action's end, we right-click on the **ultra96v1 (ultra96v1.bd)**, select **Create HDL Wrapper**, and click **OK**.

You can either add or copy the HDL wrapper file to the project. Use copy option if you would like to modify this file.



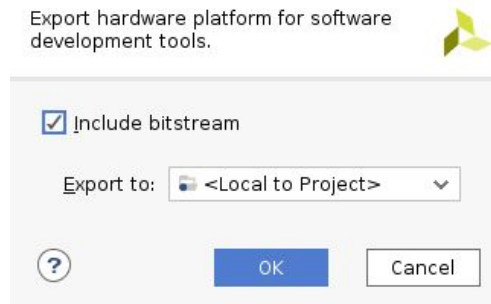
Options

☐ Copy generated wrapper to allow user edits

☒ Let Vivado manage wrapper and auto-update

In the **Flow Navigator** section we select **PROGRAM AND DEBUG -> Generate Bitstream**. We select **Yes** for executing **synthesis and implementation** and in the next window we click **OK**.

After the end of the previous action, we select **Cancel** in the window that opens. We select **File -> Export > Export Hardware** from the main menu and check the **Include Bitstream**.



Finally, we execute in the **Tcl Console** the following commands for generating and validating the dsa file:

```
write_dsa -force -include_bit <τοποθεσία_αρχείου>/όνομα_αρχείου.dsa
```

```
validate_dsa <τοποθεσία_αρχείου>/όνομα_αρχείου.dsa
```

e.g.

```
write_dsa -force -include_bit  
/mnt/terabyte/pmousoul_data/projects/vivado_2018.2/ultra96v1/ultra96v1.dsa
```

```
validate_dsa /mnt/terabyte/pmousoul_data/projects/vivado_2018.2/ultra96v1/ultra96v1.dsa
```

Close Vivado.

5. PetaLinux build

1. `petalinux-create -t project -n <project_name> -s <path_to_base_BSP>` , [e.g. `petalinux-create -t project -n ultra96v1 -s ultra96v1_full_2018_2.bsp`]
2. `petalinux-config -p <project_name> --get-hw-description=<HDF_path>` , [e.g. `petalinux-config -p ultra96v1 --get-hw-description=./`]
3. `petalinux-config -p <project_name> -c kernel`
 - Device Drivers → Generic Driver Options → Size in MB(1024)
 - Device Drivers → Staging drivers (ON)
 - Device Drivers → Staging drivers → Xilinx APF Accelerator driver (ON)
 - Device Drivers → Staging drivers → Xilinx APF Accelerator driver → Xilinx APF DMA engines support (ON)
 - CPU Power Management → CPU idle → CPU idle PM support (OFF)
 - CPU Power Management → CPU Frequency scaling → CPU Frequency scaling (OFF)

4. `petalinux-config -p <project_name> -c rootfs`
 - Filesystem Packages → misc → gcc-runtime → libstdc++ (ON)
5. `nano`
`ultra96v1/project-spec/meta-user/recipes-bsp/device-tree/files/system-user.dts`
`i`

`/ {`
`xlnk {`
`compatible = "xlnx,xlnk-1.0";`
`};`
`};`
6. `petalinux-build -p <project_name>`, για να κάνουμε build την PetaLinux διανομή.

6. Platform Creation

Prepare the PetaLinux files:

```
cd <petalinux_project>/images/linux
mkdir boot
mkdir image
cp u-boot.elf boot/u-boot.elf
cp *fsbl.elf boot/fsbl.elf
cp bl31.elf boot/bl31.elf
cp pmufw.elf boot/pmufw.elf
cp image.ub image/image.ub
```

Create file:

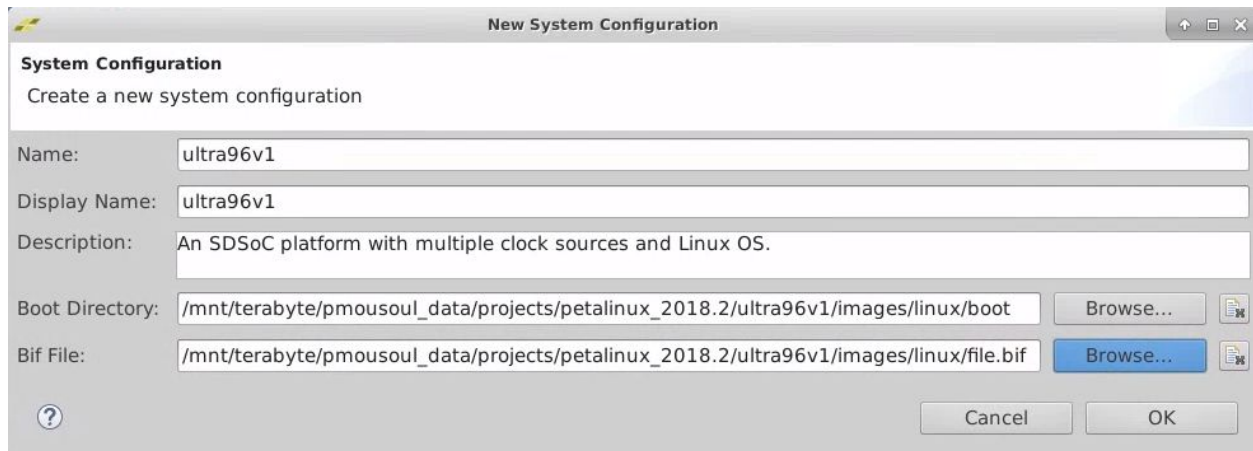
```
nano file.bif
```

```
the_ROM_image:
{
    [fsbl_config] a53_x64
    [bootloader]<fsbl.elf>
    [pmufw_image]<pmufw.elf>
    [destination_device=pl] <bitstream>
    [destination_cpu=a53-0, exception_level=e1-3, trustzone] <bl31.elf>
    [destination_cpu=a53-0, exception_level=e1-2] <u-boot.elf>
}
```

Open SDSoc and select **File -> New SDx Project**. We select **Platform** and click **Next**.

In the next window we set the location of the DSA file we created. We also select **Import Software Platform Components** and then we select **Finish**.

We click on the **Define System Configuration** and we type the information displayed in the next figure.

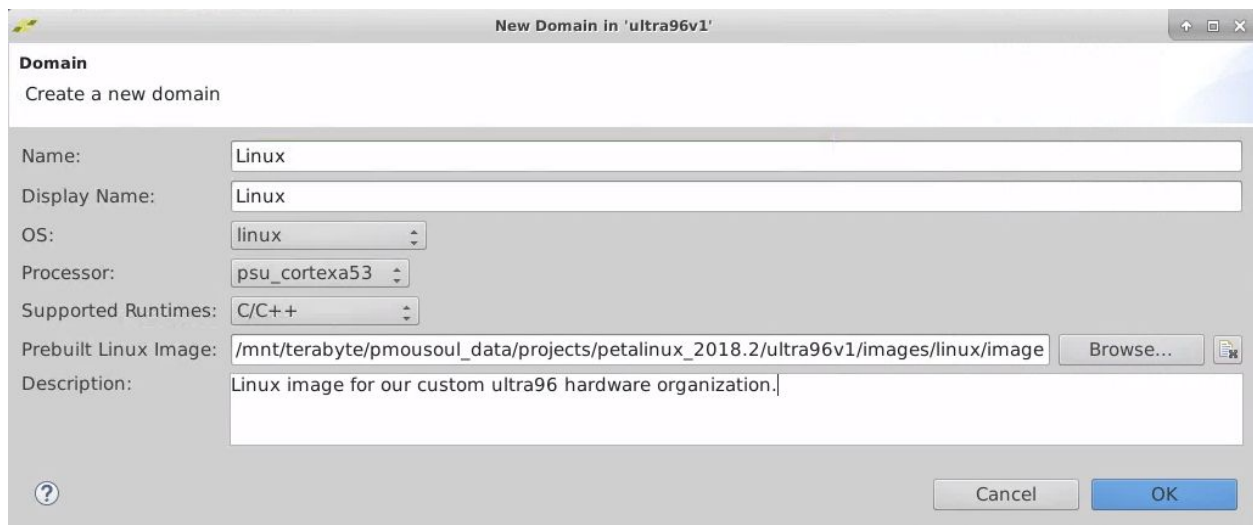


The screenshot shows a dialog box titled "New System Configuration". It contains the following fields and values:

- Name:** ultra96v1
- Display Name:** ultra96v1
- Description:** An SDSoc platform with multiple clock sources and Linux OS.
- Boot Directory:** /mnt/terabyte/pmousoul_data/projects/petalinux_2018.2/ultra96v1/images/linux/boot
- Bif File:** /mnt/terabyte/pmousoul_data/projects/petalinux_2018.2/ultra96v1/images/linux/file.bif

At the bottom, there are "Cancel" and "OK" buttons.

We click **OK** to close the window and we click on the **Add Processor Group/Domain**. we type the information displayed in the next figure.



The screenshot shows a dialog box titled "New Domain in 'ultra96v1'". It contains the following fields and values:

- Name:** Linux
- Display Name:** Linux
- OS:** linux
- Processor:** psu_cortexa53
- Supported Runtimes:** C/C++
- Prebuilt Linux Image:** /mnt/terabyte/pmousoul_data/projects/petalinux_2018.2/ultra96v1/images/linux/image
- Description:** Linux image for our custom ultra96 hardware organization.

At the bottom, there are "Cancel" and "OK" buttons.

Next, we select the **Generate Platform** and click **OK** in the window that opens. Finally we select the **Add to Custom Repositories** in order to make the platform available in SDSoc and click **OK**.