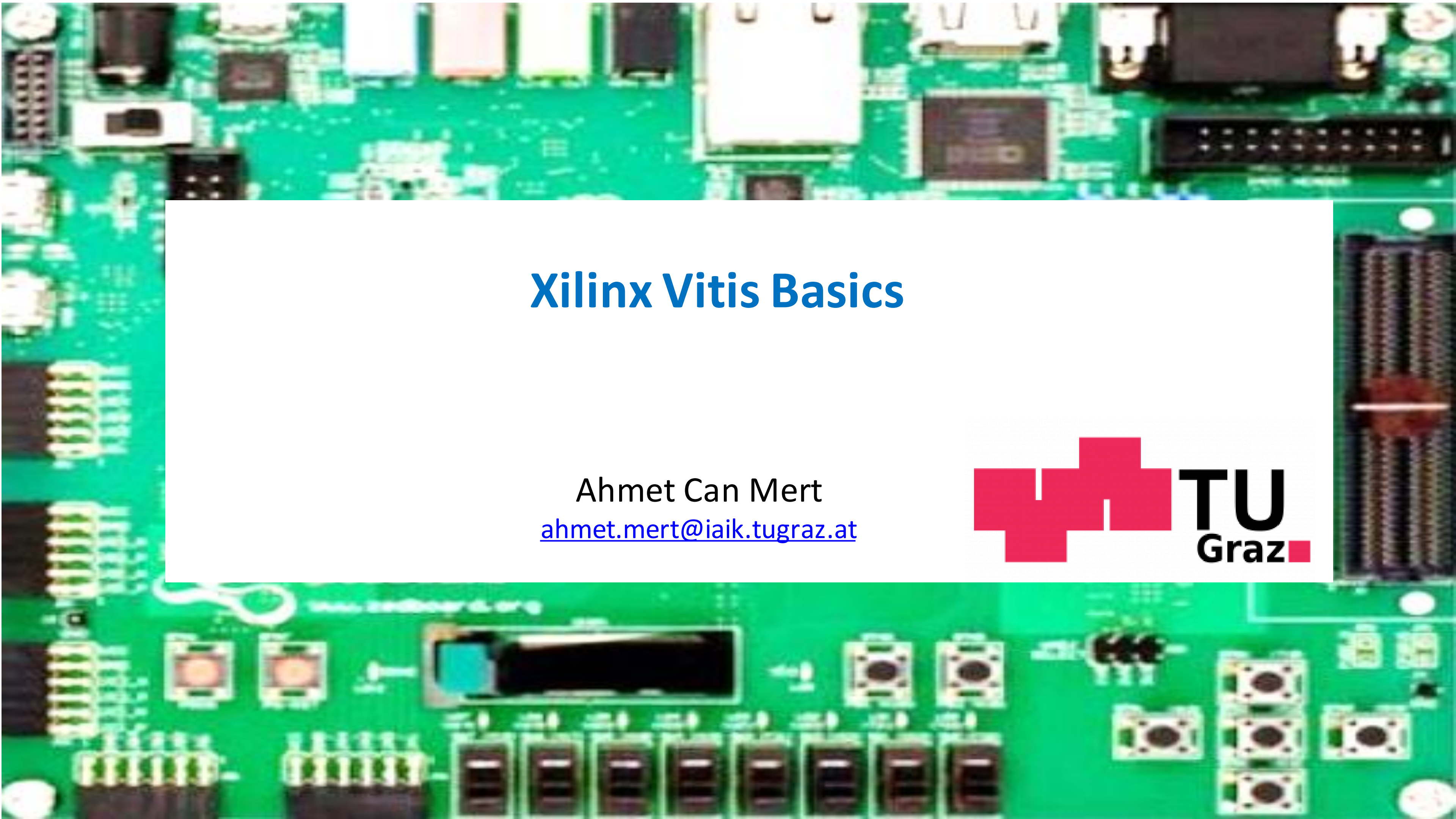


Xilinx Vitis Basics

Ahmet Can Mert
ahmet.mert@iaik.tugraz.at

Overview

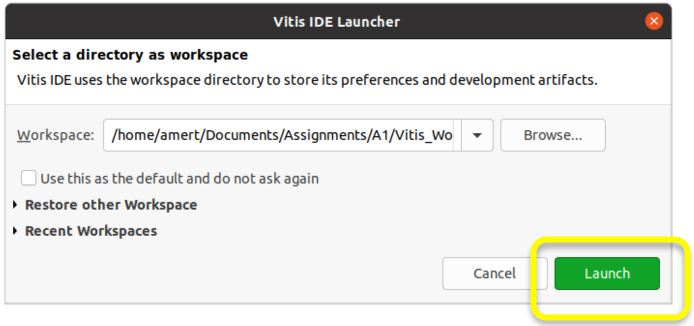
- We will use Xilinx Vitis tool as a platform for developing SW applications on embedded Arm processors (on our target Pynq board).

- In this tutorial, we will use Vivado project/Vitis code provided for Assignment 1 as an example (see course website for project files: https://www.iaik.tugraz.at/chw)

- Vitis User Guide: https://docs.xilinx.com/r/en-US/ug1400-vitis-embedded/Embedded-Design-Tutorials

Step 1 - Launching Vitis

- Launch Vitis IDE (see installation guide to launch Vitis), set your workspace (to any location) and click on Launch.
    - You will see an empty project.

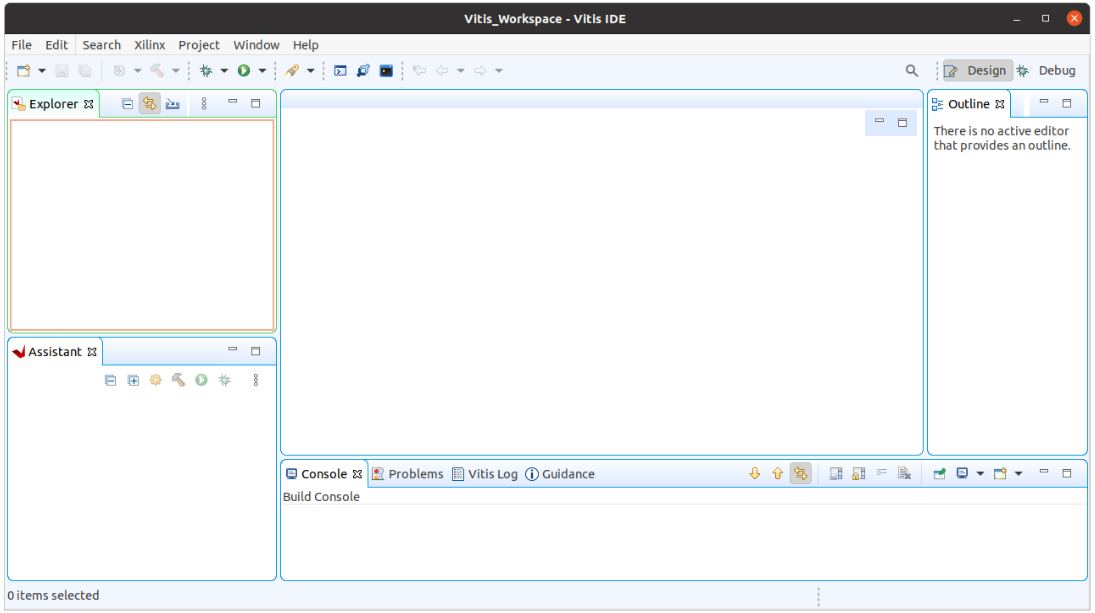


Vitis IDE Launcher

Select a directory as workspace

Vitis IDE uses the workspace directory to store its preferences and development artifacts.

Workspace: /home/amert/Documents/Assignments/A1/Vitis_Wo



Vitis_Workspace - Vitis IDE

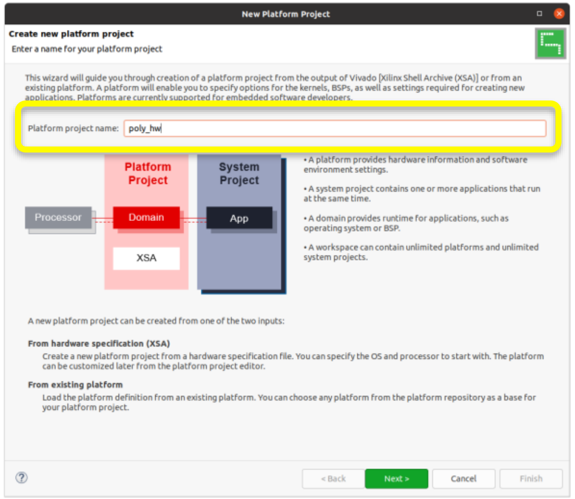
There is no active editor that provides an outline.

Build Console

0 items selected

Step 2 – Creating Platform Project

- First, we have to create a Platform Project.
  - Go to File -> New -> Platform Project.
  - Give a name to your Platform Project (i.e., poly_hw) and click on Next.
  - Click on Browse and select XSA file generated by your Vivado project (for this tutorial, it is the pre-generated XSA file in Cryptoprocessor project provided for Assignment1, located at `CryptoProcessor/CryptoProcessor_2022_23/project_1`).
  - Finally, click on Finish.



New Platform Project

Create new platform project

Enter a name for your platform project

This wizard will guide you through creation of a platform project from the output of Vivado [Xilinx Shell Archive (XSA)] or from an existing platform. A platform will enable you to specify options for the kernels, BSPs, as well as settings required for creating new applications. Platforms are currently supported for embedded software developers.

Platform project name: poly_hw

- A platform provides hardware information and software environment settings.
- A system project contains one or more applications that run at the same time.
- A domain provides runtime for applications, such as operating system or BSP.
- A workspace can contain unlimited platforms and unlimited system projects.

A new platform project can be created from one of the two inputs:

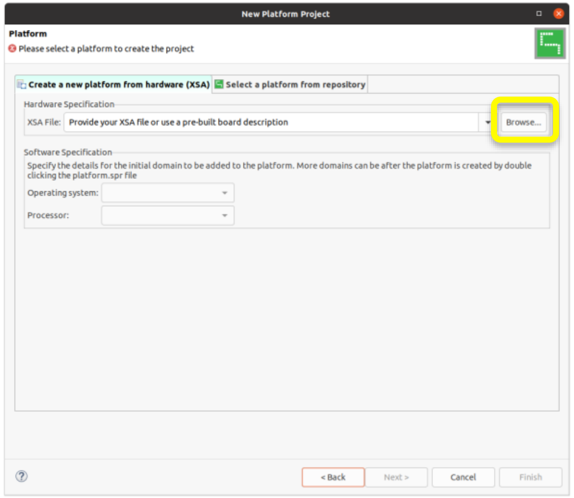
From hardware specification (XSA)

Create a new platform project from a hardware specification file. You can specify the OS and processor to start with. The platform can be customized later from the platform project editor.

From existing platform

Load the platform definition from an existing platform. You can choose any platform from the platform repository as a base for your platform project.

< Back
Next >
Cancel
Finish



New Platform Project

Platform

Please select a platform to create the project

Create a new platform from hardware (XSA)
Select a platform from repository

Hardware Specification

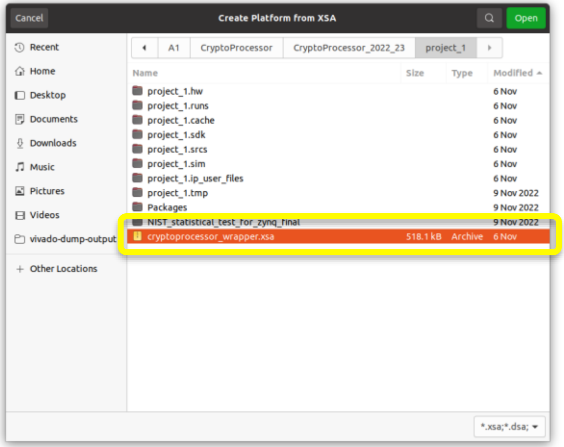
XSA File: Provide your XSA file or use a pre-built board description

Software Specification

Specify the details for the initial domain to be added to the platform. More domains can be after the platform is created by double clicking the platform.spr file

Operating system:

Processor:



Create Platform from XSA

Name	Size	Type	Modified
project_1.hw			6 Nov
project_1.runs			6 Nov
project_1.cache			6 Nov
project_1.sdk			6 Nov
project_1.srcs			6 Nov
project_1.sim			6 Nov
project_1.ip_user_files			6 Nov
project_1.tmp			9 Nov 2022
Packages			9 Nov 2022
NIST_statistical_test_for_zynq_final			9 Nov 2022
cryptoprocessor_wrapper.xsa	518.1 kB	Archive	6 Nov

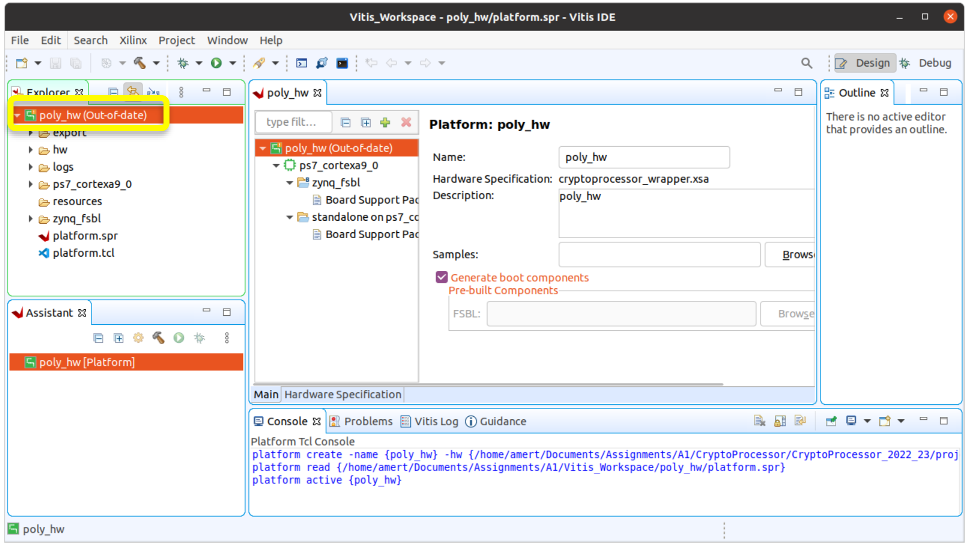
Step 2 – Creating Platform Project

How to generate XSA file?

1. Go to Vivado and open your project.
2. Make changes to your RTL, verify changes using RTL simulation.
3. Synthesize, implement and generate bitstream
4. Export your design, File -> Export -> Export Hardware.
   In the Export window, Next -> Select Include bitstream and Next.

Step 2 – Creating Platform Project

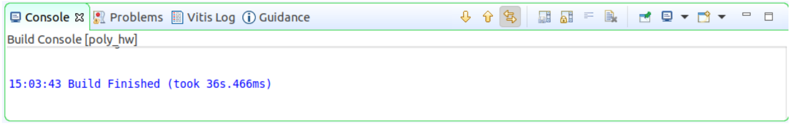
- Your project platform is out-of-date. Right click on its name and then click on Build Project.
  - You should see Build is finished without any error.



Platform: poly_hw

Name: poly_hw
Hardware Specification: cryptoprocessor_wrapper.xsa
Description: poly_hw
Samples:
Generate boot components
Pre-built Components
FSBL:

```
Platform Tcl Console
platform create -name {poly_hw} -hw {/home/amert/Documents/Assignments/A1/CryptoProcessor/CryptoProcessor_2022_23/proj
platform read {/home/amert/Documents/Assignments/A1/Vitis_Workspace/poly_hw/platform.spr}
platform active {poly_hw}
```



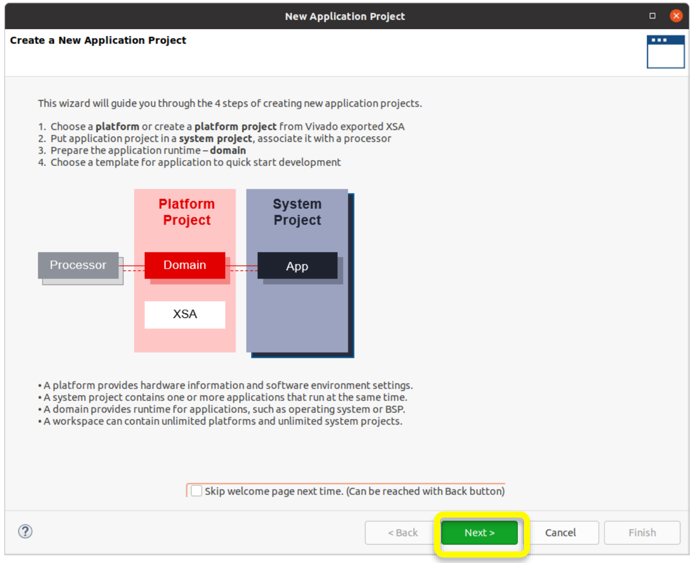
```
Build Console [poly_hw]

15:03:43 Build Finished (took 36s.466ms)
```

If you use Windows OS and face compilation error in this step, please see the following thread for the solution: https://support.xilinx.com/s/question/0D52E00006hpOx5SAE/drivers-and-makefiles-problems-in-vitis-20202?language=en_US

Step 3 – Creating Application Project

- Now, we have to create Application Project.
  - Go to File -> New -> Application Project.
  - Select your Platform Project.



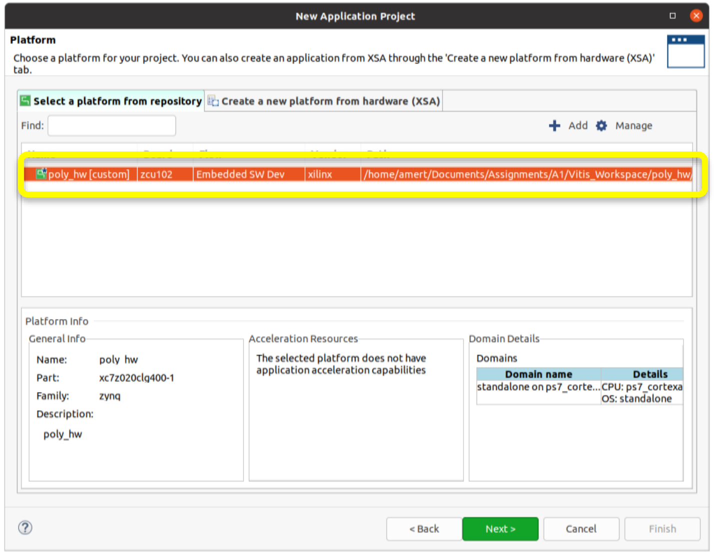
Create a New Application Project

This wizard will guide you through the 4 steps of creating new application projects.

1. Choose a **platform** or create a **platform project** from Vivado exported XSA
2. Put application project in a **system project**, associate it with a processor
3. Prepare the application runtime – **domain**
4. Choose a template for application to quick start development

- A platform provides hardware information and software environment settings.
- A system project contains one or more applications that run at the same time.
- A domain provides runtime for applications, such as operating system or BSP.
- A workspace can contain unlimited platforms and unlimited system projects.

☐ Skip welcome page next time. (Can be reached with Back button)


New Application Project
Select a platform from repository
Create a new platform from hardware (XSA)
Find:
Add
Manage
< Back
Next >
Cancel
Finish

Platform

Choose a platform for your project. You can also create an application from XSA through the 'Create a new platform from hardware (XSA)' tab.

poly_hw [custom]	zcu102	Embedded SW Dev	xilinx	/home/amert/Documents/Assignments/A1/Vitis_Workspace/poly_hw/

Platform Info

General Info

Name:	poly hw
Part:	xc7z020clg400-1
Family:	zynq
Description:	

poly_hw

Acceleration Resources

The selected platform does not have application acceleration capabilities

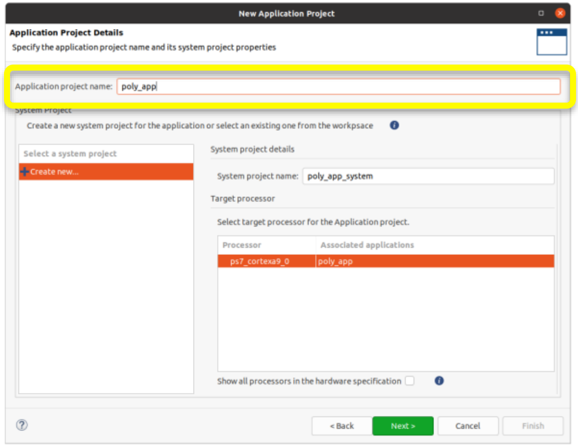
Domain Details

Domains

Domain name	Details
standalone on ps7_corte...	CPU: ps7_cortexa OS: standalone

Step 3 – Creating Application Project

- Now, we have to create Application Project.
  - Go to File -> New -> Application Project.
  - Select your Platform Project.
  - Give Application Project a name (i.e., poly_app) and select Empty Application (C++).



New Application Project

Application Project Details

Specify the application project name and its system project properties

Application project name: poly_app

System Project

Create a new system project for the application or select an existing one from the workpsace

Select a system project
- Create new...

System project details

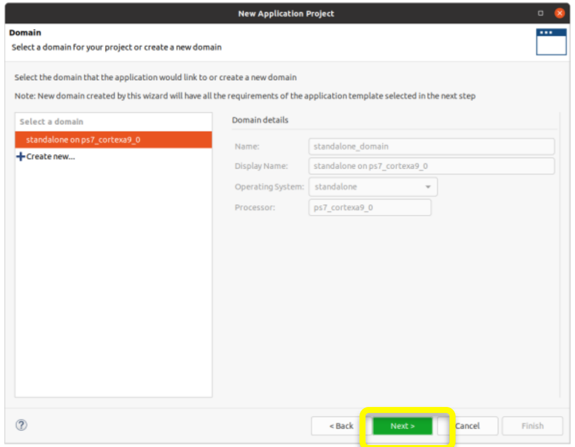
System project name: poly_app_system

Target processor

Select target processor for the Application project.

Processor	Associated applications
ps7_cortexa9_0	poly_app

Show all processors in the hardware specification ☐



New Application Project

Domain

Select a domain for your project or create a new domain

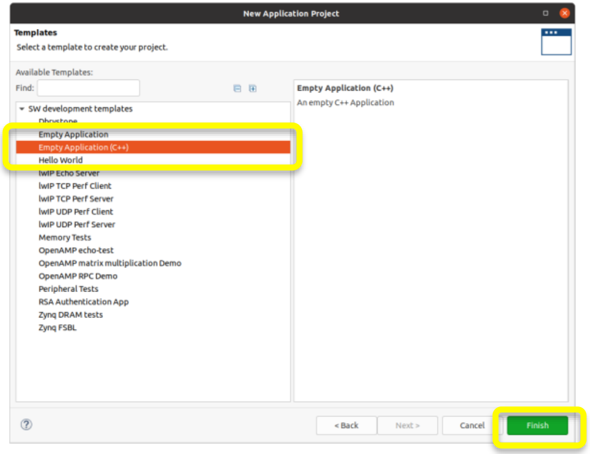
Select the domain that the application would link to or create a new domain

Note: New domain created by this wizard will have all the requirements of the application template selected in the next step

Select a domain
- standalone on ps7_cortexa9_0
- Create new...

Domain details

Name: standalone_domain
Display Name: standalone on ps7_cortexa9_0
Operating System: standalone
Processor: ps7_cortexa9_0



New Application Project

Templates

Select a template to create your project.

Available Templates:

Find:

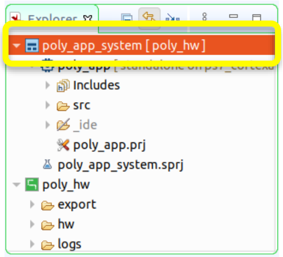
SW development templates
- Dhrystone
- Empty Application
- Empty Application (C++)
- Hello World
- lwIP Echo Server
- lwIP TCP Perf Client
- lwIP TCP Perf Server
- lwIP UDP Perf Client
- lwIP UDP Perf Server
- Memory Tests
- OpenAMP echo-test
- OpenAMP matrix multiplication Demo
- OpenAMP RPC Demo
- Peripheral Tests
- RSA Authentication App
- Zynq DRAM tests
- Zynq FSBL

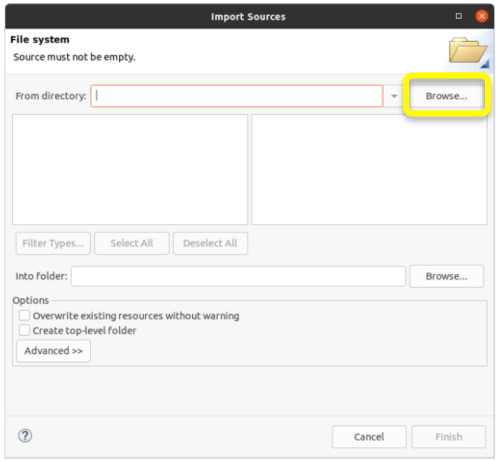
Empty Application (C++)

An empty C++ Application

Step 3 – Creating Application Project

- Import source files (Vitis_code), provided for Assignment 1 on course website, to the Application Project.
  - Right click on Application Project -> Import Sources
  - Click on Browse (first one) and select Vitis_code folder.
  - For destination folder, click on Browse (second one) and select src folder of Application Project.

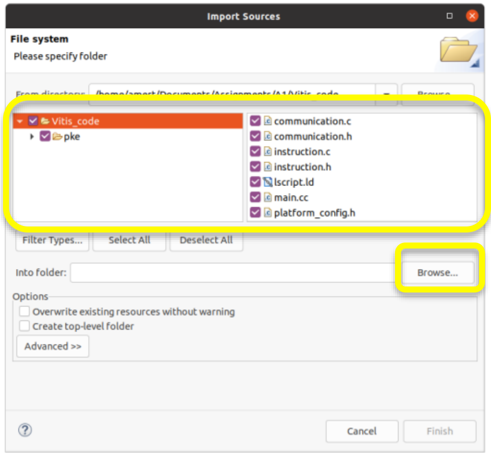



Import Sources

File system

Source must not be empty.

From directory:

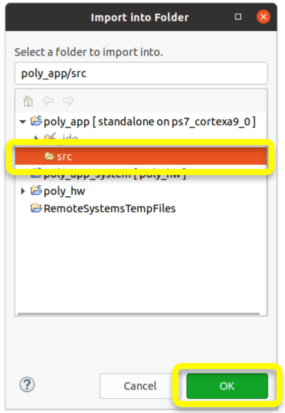

Import Sources

File system

Please specify folder

From directory: /home/amert/Documents/Assignments/A1/Vitis_code

- [✓] Vitis_code
  - [✓] pke
- [✓] communication.c
- [✓] communication.h
- [✓] instruction.c
- [✓] instruction.h
- [✓] lscript.ld
- [✓] main.cc
- [✓] platform_config.h

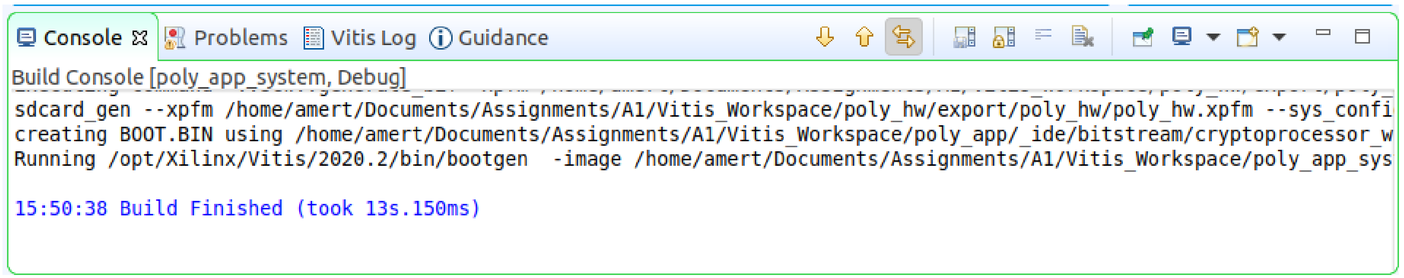

Import into Folder

Select a folder to import into.

poly_app/src
poly_app [standalone on ps7_cortexa9_0]
_ide
src
poly_app_system [poly_hw]
poly_hw
RemoteSystemsTempFiles
Cancel
OK

Step 3 – Creating Application Project

- Right click on Application Project and click on Build Project.
    - You should see Build is finished without any error.



Build Console [poly_app_system, Debug]

```
sdcard_gen --xpfm /home/amert/Documents/Assignments/A1/Vitis_Workspace/poly_hw/export/poly_hw/poly_hw.xpfm --sys_confi
creating BOOT.BIN using /home/amert/Documents/Assignments/A1/Vitis_Workspace/poly_app/_ide/bitstream/cryptoprocessor_w
Running /opt/Xilinx/Vitis/2020.2/bin/bootgen  -image /home/amert/Documents/Assignments/A1/Vitis_Workspace/poly_app_sys

15:50:38 Build Finished (took 13s.150ms)
```

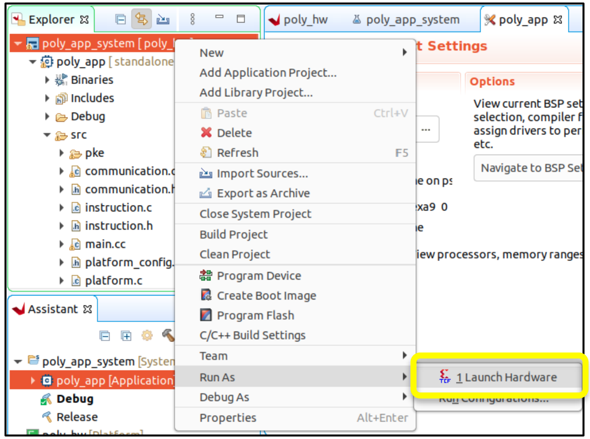
If you use Windows OS and face compilation error in this step, please see the following thread for the solution:
https://support.xilinx.com/s/question/0D52E00006iHqrQSAS/vitis-project-wont-build?language=en_US

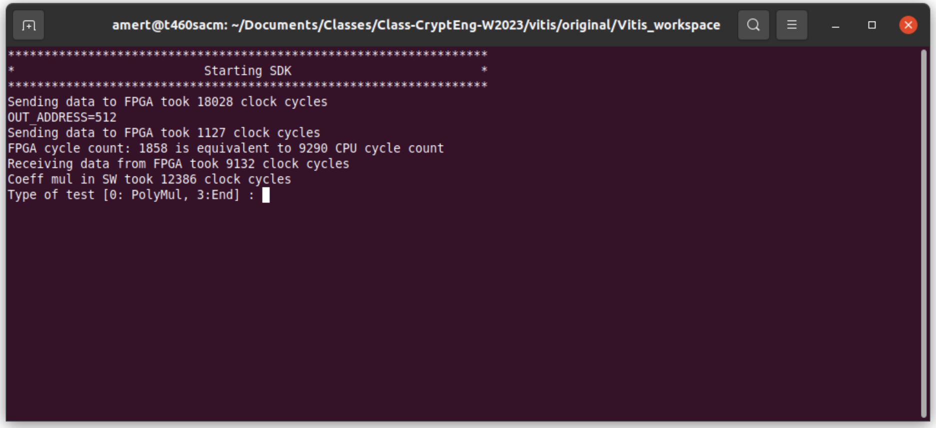
Step 4 - Running the example code

- To observe FPGA output, you should observe the serial port output on the Ubuntu terminal using the following command (first connect your FPGA to your PC).

```
sudo screen /dev/serial/by-id/usb-Xilinx_TUL_1234-tul-if01-port0 115200,cs8,-parenb,-cstopb,-
hupcl
```

- Right click on your Application Project and then click on Run As -> Launch Hardware.
  - Then, the FPGA will be programmed and you will observe the following output.




amert@t460sacm: ~/Documents/Classes/Class-CryptEng-W2023/vitis/original/Vitis_workspace

```
******************************************************************
*                         Starting SDK                           *
******************************************************************
Sending data to FPGA took 18028 clock cycles
OUT_ADDRESS=512
Sending data to FPGA took 1127 clock cycles
FPGA cycle count: 1858 is equivalent to 9290 CPU cycle count
Receiving data from FPGA took 9132 clock cycles
Coeff mul in SW took 12386 clock cycles
Type of test [0: PolyMul, 3:End] :
```

Step 4 - Running the example code

- When you make changes in your code, build your Application Project again before running your code (by right clicking on your Application Project -> Build Project).

- For observing output, you can also use Terminal function of Vitis to observe output. See the following Video for steps: https://youtu.be/3D2-OPArCiA?t=524

Step 5 - Updating your HW design

- When you update your design on Xilinx Vivado, you have to re-generate bitstream and import its XSA for Vitis. In Vivado:

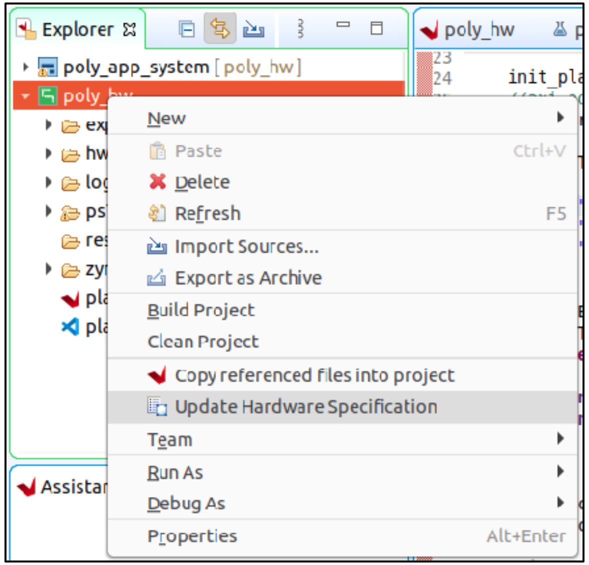
1. Make changes to your RTL, verify changes using RTL simulation.
2. Synthesize, implement and generate bitstream
3. Export your design, File -> Export -> Export Hardware.
   In the Export window, Next -> Select Include bitstream and Next.

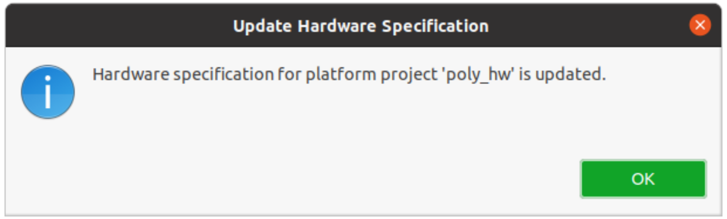
Now, go back to Vitis.

Step 5 - Updating your HW design

- In Vitis:

1. Right click on your platform project and click on Update Hardware Specification.
2. Select XSA exported from Vivado and click on OK.



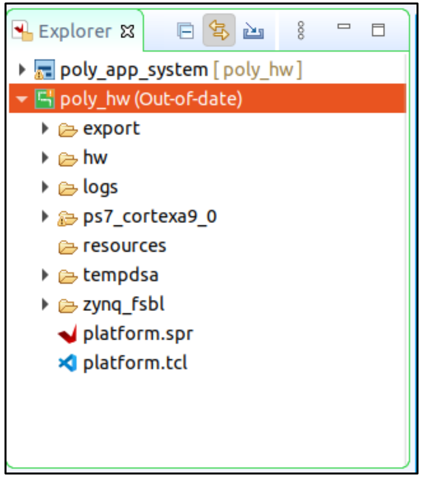


Hardware specification for platform project 'poly_hw' is updated.

Step 5 - Updating your HW design

- In Vitis:

1. After the update, you will see an Out-of-date tag near the platform project.
2. Right click on the Platform Project and select Build Project.
3. Right click on the Application Project and select Build Project.
4. You can now re-run the code and observe your output (as shown in Step 4)


Explorer
poly_app_system [poly_hw]
poly_hw (Out-of-date)
export
hw
logs
ps7_cortexa9_0
resources
tempdsa
zynq_fsbl
platform.spr
platform.tcl