Vitis

Overview

Rui Machado

Embedded Systems Research Group (ESRG)
Centro ALGORITMI, Universidade do Minho

https://esrg.algoritmi.uminho.pt/

TABLE OF CONTENTS

01 Create the design

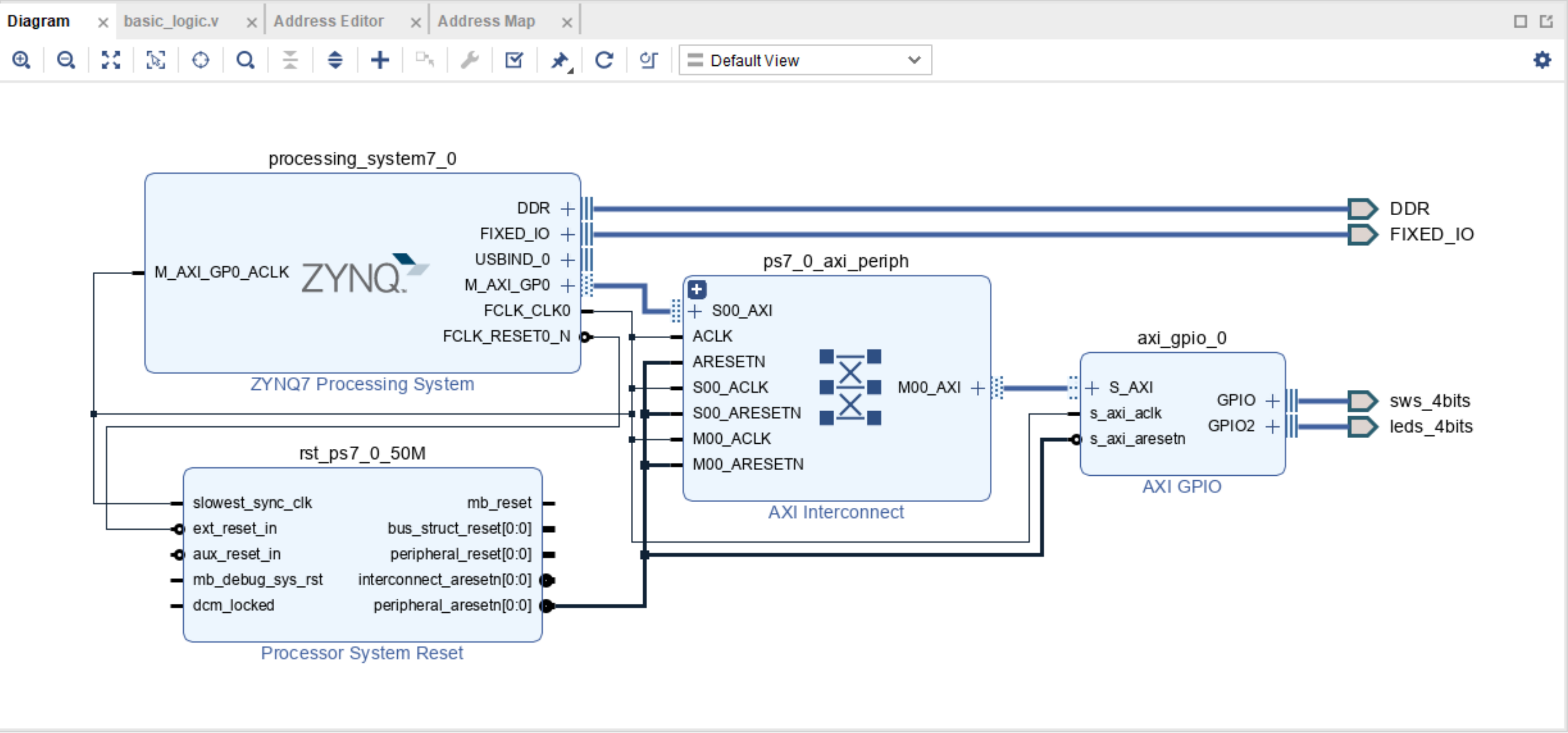
02 Create a Vitis project

03 GUI overview

04 Hello World

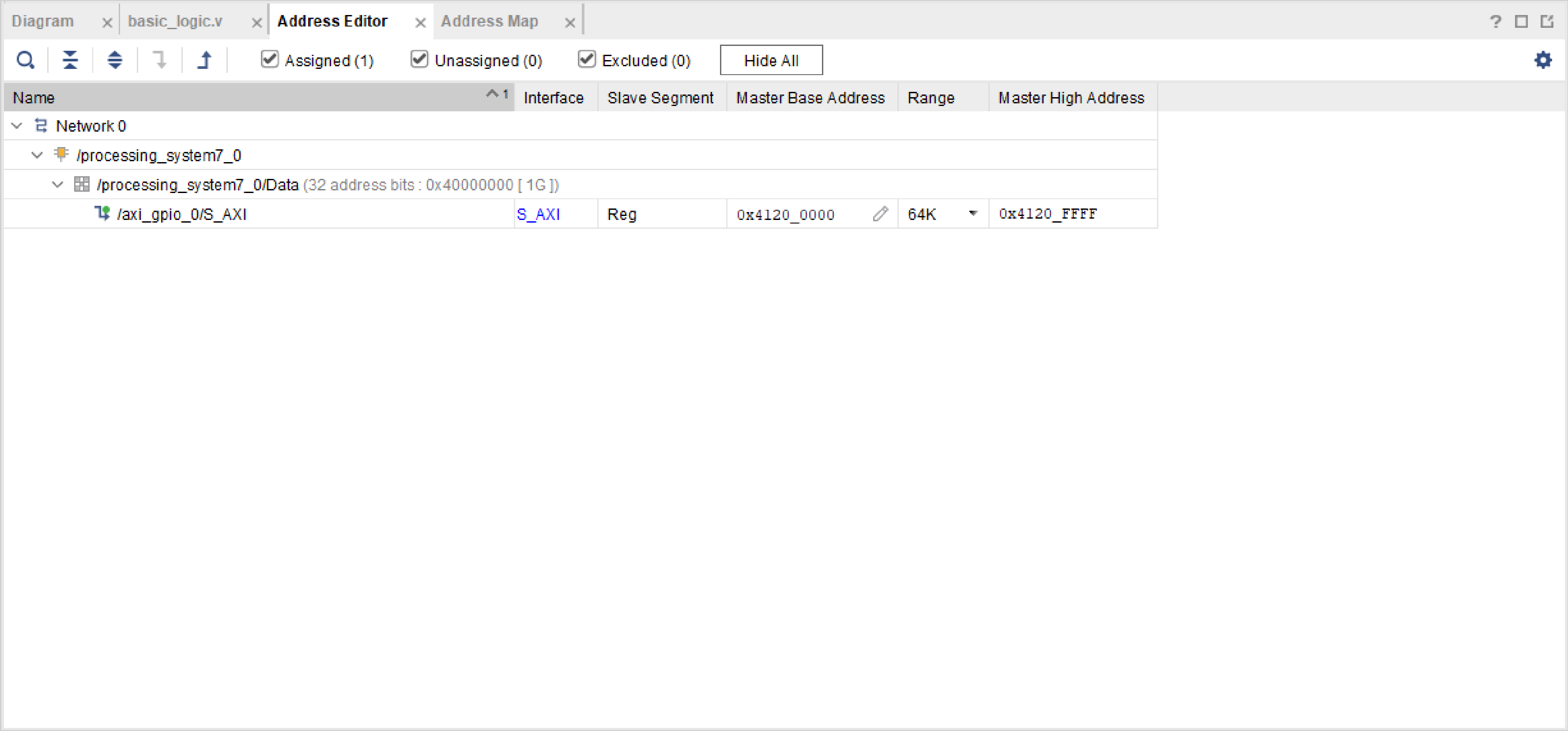
Create the design

- Create a new project and then create a new block design as the one presented



Create the design

- Check the address of the GPIO IP you instantiated and then generate the bitstream of the design

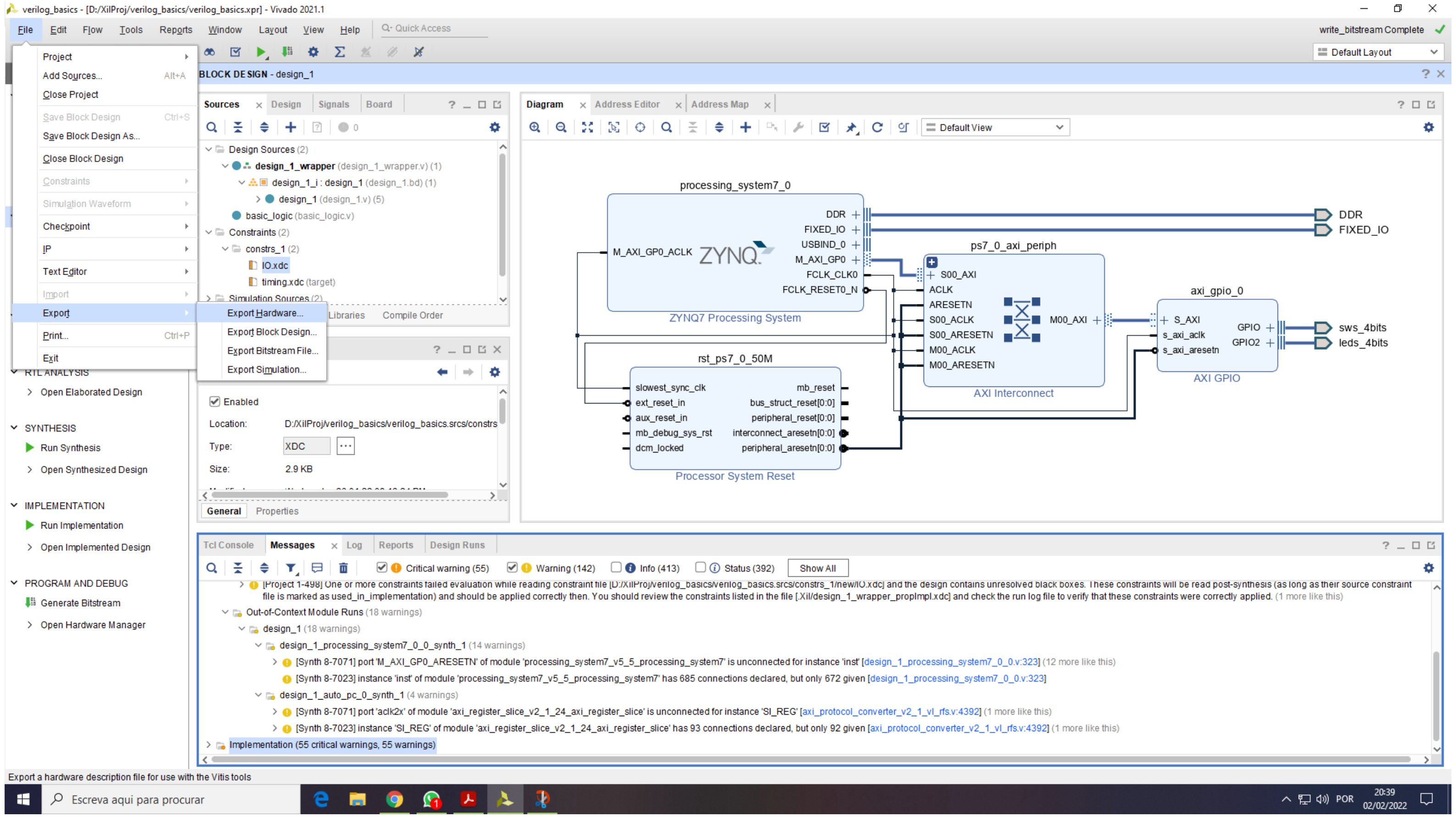


☑ Assigned (1) ☑ Unassigned (0) ☑ Excluded (0)

Name	Interface	Slave Segment	Master Base Address	Range	Master High Address
Network 0					
/processing_system7_0					
/processing_system7_0/Data (32 address bits : 0x40000000 [1G])					
/axi_gpio_0/S_AXI	S_AXI	Reg	0x4120_0000	64K	0x4120_FFFF

Create the design

- After finishing the bitstream generation export the hardware

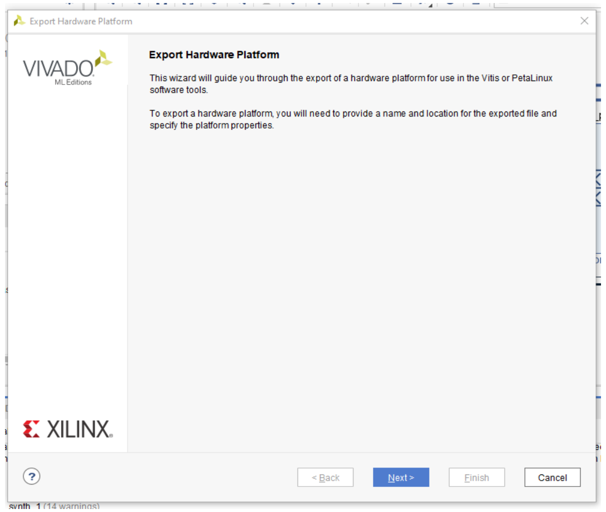


```
[Project 1-498] One or more constraints failed evaluation while reading constraint file [D:/XilProj/verilog_basics/verilog_basics.srcs/constrs_1/new/IO.xdc] and the design contains unresolved black boxes. These constraints will be read post-synthesis (as long as their source constraint file is marked as used_in_implementation) and should be applied correctly then. You should review the constraints listed in the file [.Xil/design_1_wrapper_propImpl.xdc] and check the run log file to verify that these constraints were correctly applied. (1 more like this)
Out-of-Context Module Runs (18 warnings)
  design_1 (18 warnings)
    design_1_processing_system7_0_0_synth_1 (14 warnings)
      [Synth 8-7071] port 'M_AXI_GP0_ARESETN' of module 'processing_system7_v5_5_processing_system7' is unconnected for instance 'inst' [design_1_processing_system7_0_0.v:323] (12 more like this)
      [Synth 8-7023] instance 'inst' of module 'processing_system7_v5_5_processing_system7' has 685 connections declared, but only 672 given [design_1_processing_system7_0_0.v:323]
    design_1_auto_pc_0_synth_1 (4 warnings)
      [Synth 8-7071] port 'aclk2x' of module 'axi_register_slice_v2_1_24_axi_register_slice' is unconnected for instance 'SI_REG' [axi_protocol_converter_v2_1_vl_rfs.v:4392] (1 more like this)
      [Synth 8-7023] instance 'SI_REG' of module 'axi_register_slice_v2_1_24_axi_register_slice' has 93 connections declared, but only 92 given [axi_protocol_converter_v2_1_vl_rfs.v:4392] (1 more like this)
Implementation (55 critical warnings, 55 warnings)
```

Export a hardware description file for use with the Vitis tools

Create the design

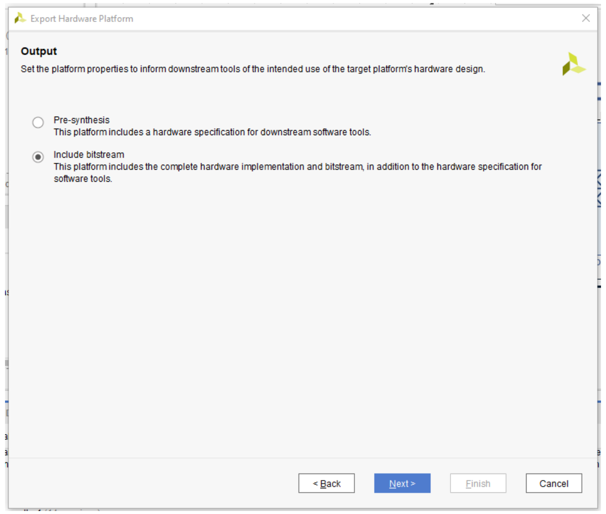
- Follow the Export Hardware platform widget. Make sure you select include bitstream and select a directory to store the .xsa that will be created then finish



Export Hardware Platform

This wizard will guide you through the export of a hardware platform for use in the Vitis or PetaLinux software tools.

To export a hardware platform, you will need to provide a name and location for the exported file and specify the platform properties.

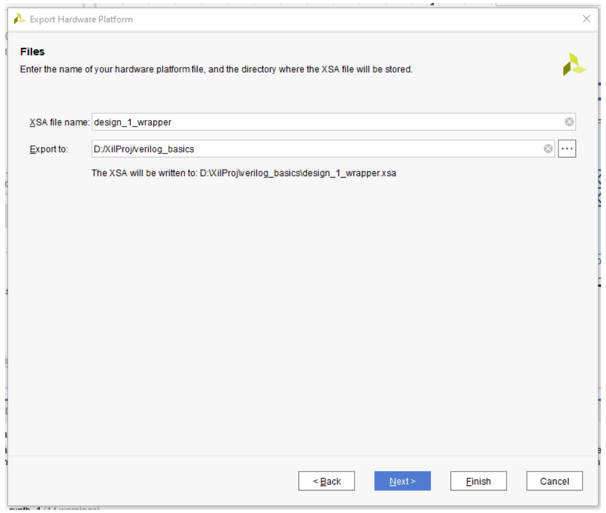


Output

Set the platform properties to inform downstream tools of the intended use of the target platform's hardware design.

○ Pre-synthesis
This platform includes a hardware specification for downstream software tools.

● Include bitstream
This platform includes the complete hardware implementation and bitstream, in addition to the hardware specification for software tools.



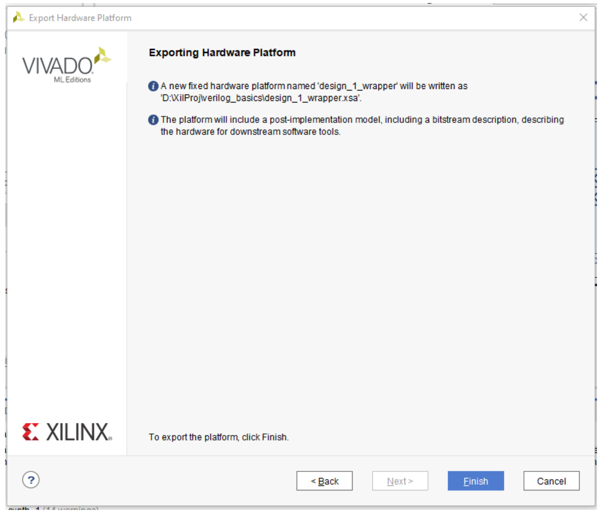
Files

Enter the name of your hardware platform file, and the directory where the XSA file will be stored.

XSA file name: design_1_wrapper

Export to: D:/XilProj/verilog_basics

The XSA will be written to: D:\XilProj\verilog_basics\design_1_wrapper.xsa



Exporting Hardware Platform

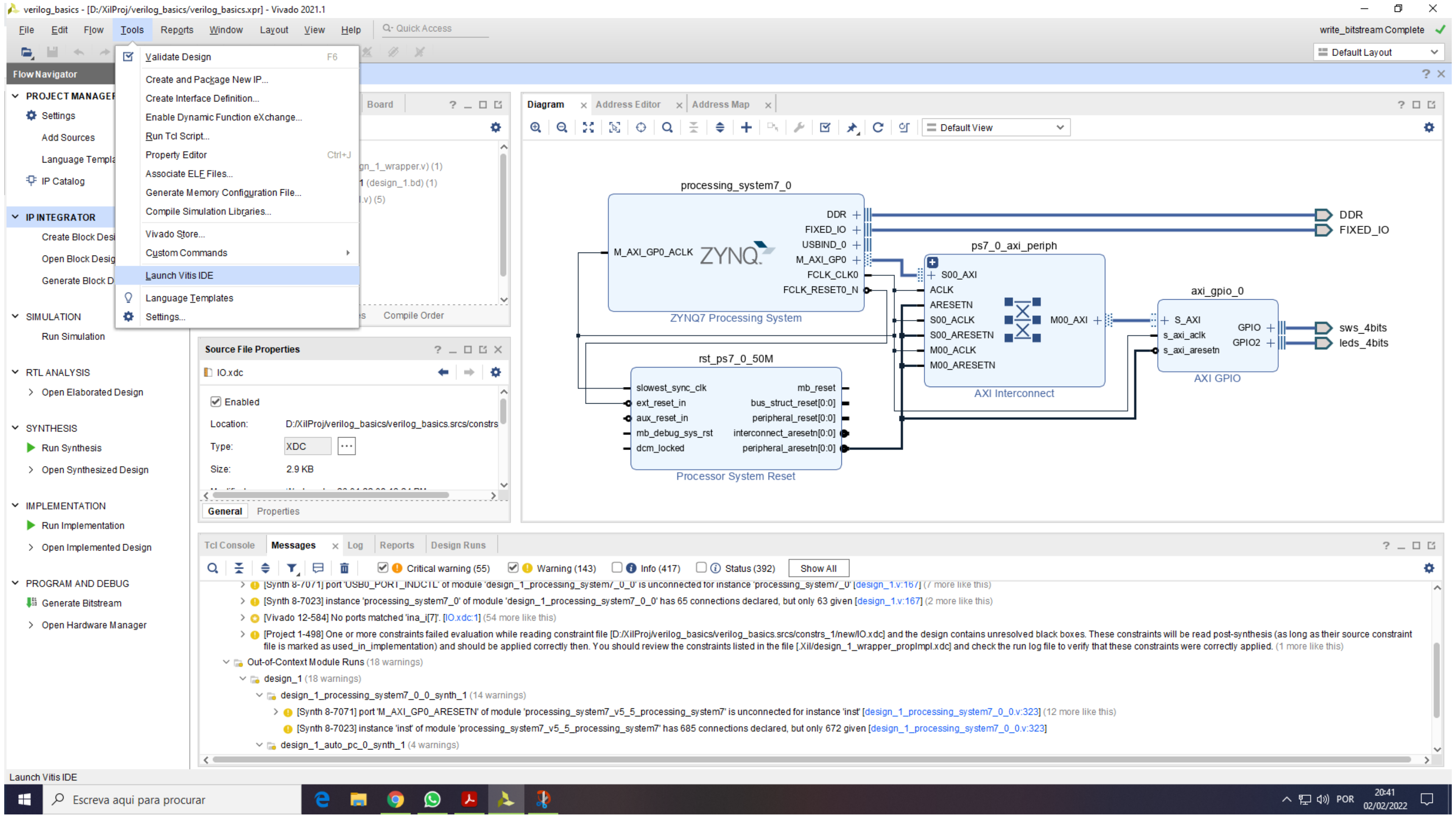
A new fixed hardware platform named 'design_1_wrapper' will be written as 'D:\XilProj\verilog_basics\design_1_wrapper.xsa'.

The platform will include a post-implementation model, including a bitstream description, describing the hardware for downstream software tools.

To export the platform, click Finish.

Create the design

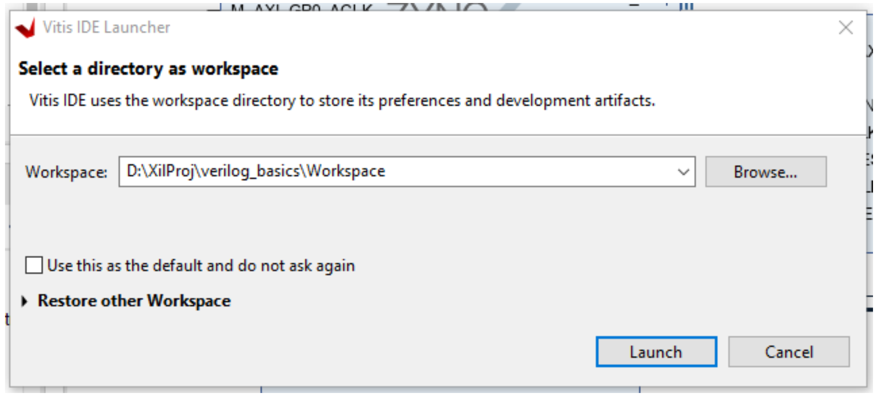
- Launch Vitis



```
[Synth 8-7071] port 'USB0_PORT_INDCTL' of module 'design_1_processing_system7_0_0' is unconnected for instance 'processing_system7_0' [design_1.v:167] (7 more like this)
[Synth 8-7023] instance 'processing_system7_0' of module 'design_1_processing_system7_0_0' has 65 connections declared, but only 63 given [design_1.v:167] (2 more like this)
[Vivado 12-584] No ports matched 'ina_i[7]'. [IO.xdc:1] (54 more like this)
[Project 1-498] One or more constraints failed evaluation while reading constraint file [D:/XilProj/verilog_basics/verilog_basics.srcs/constrs_1/new/IO.xdc] and the design contains unresolved black boxes. These constraints will be read post-synthesis (as long as their source constraint file is marked as used_in_implementation) and should be applied correctly then. You should review the constraints listed in the file [.Xil/design_1_wrapper_propImpl.xdc] and check the run log file to verify that these constraints were correctly applied. (1 more like this)
Out-of-Context Module Runs (18 warnings)
  design_1 (18 warnings)
    design_1_processing_system7_0_0_synth_1 (14 warnings)
      [Synth 8-7071] port 'M_AXI_GP0_ARESETN' of module 'processing_system7_v5_5_processing_system7' is unconnected for instance 'inst' [design_1_processing_system7_0_0.v:323] (12 more like this)
      [Synth 8-7023] instance 'inst' of module 'processing_system7_v5_5_processing_system7' has 685 connections declared, but only 672 given [design_1_processing_system7_0_0.v:323]
    design_1_auto_pc_0_synth_1 (4 warnings)
```

Create the design

- Select your working directory and then create a new application project



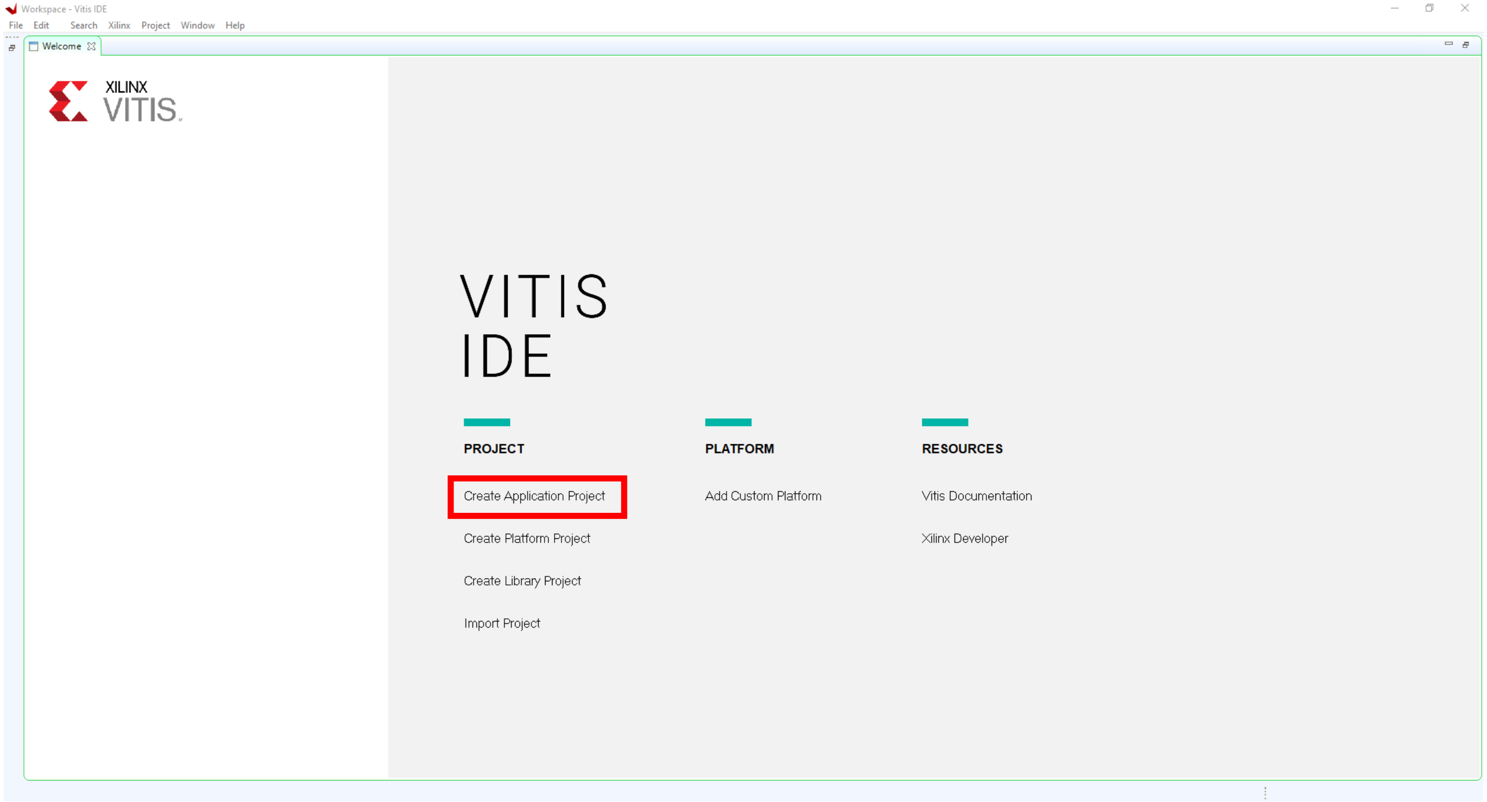
Vitis IDE Launcher

Select a directory as workspace

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

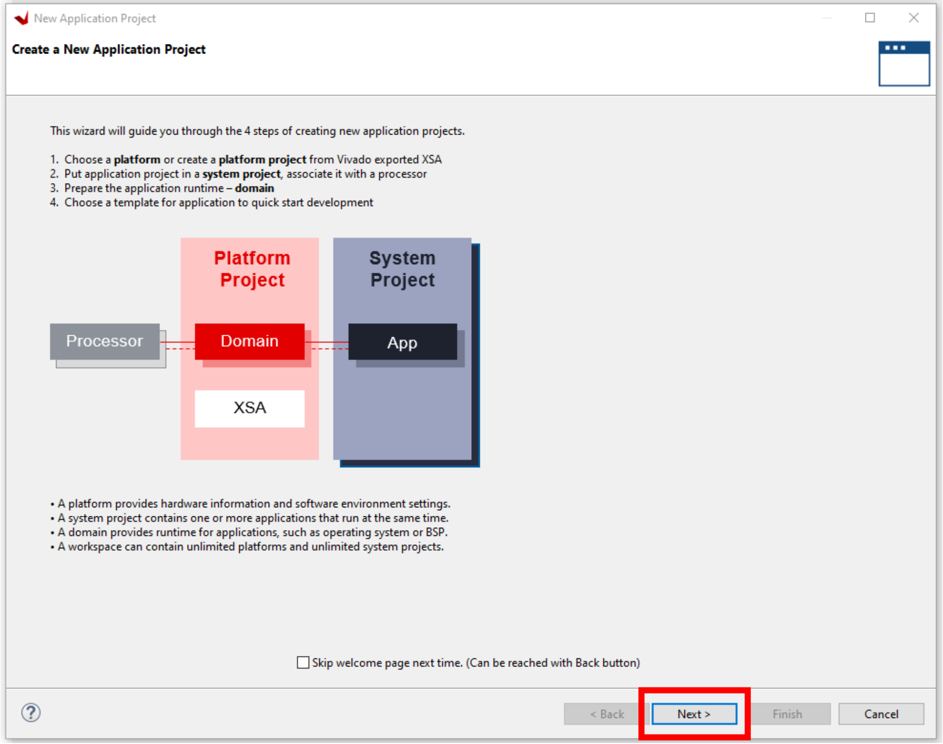
Workspace: D:\XilProj\verilog_basics\Workspace

□ Use this as the default and do not ask again



Create the design

- Follow the New Application project Widget.
- Select the Hardware platform that you want to use (the one you just exported)



New Application Project

Create a New Application Project

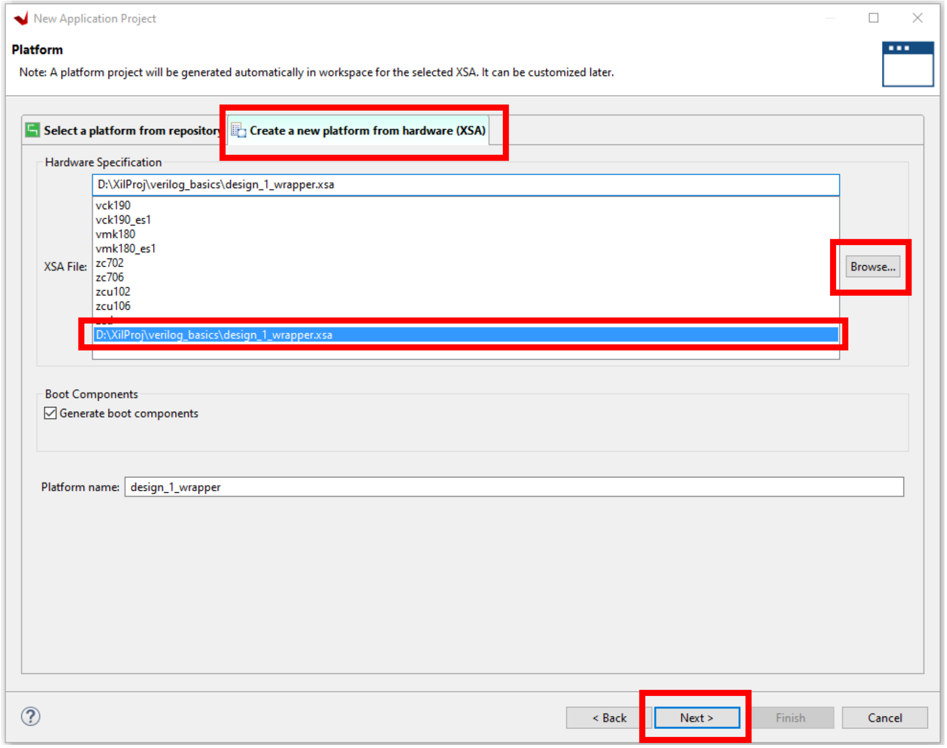
This wizard will guide you through the 4 steps of creating new application projects.

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)

< Back
Next >
Finish
Cancel



New Application Project

Platform

Note: A platform project will be generated automatically in workspace for the selected XSA. It can be customized later.

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

Hardware Specification

XSA File: D:\XilProj\verilog_basics\design_1_wrapper.xsa

vck190
vck190_es1
vmk180
vmk180_es1
zc702
zc706
zcu102
zcu106
D:\XilProj\verilog_basics\design_1_wrapper.xsa

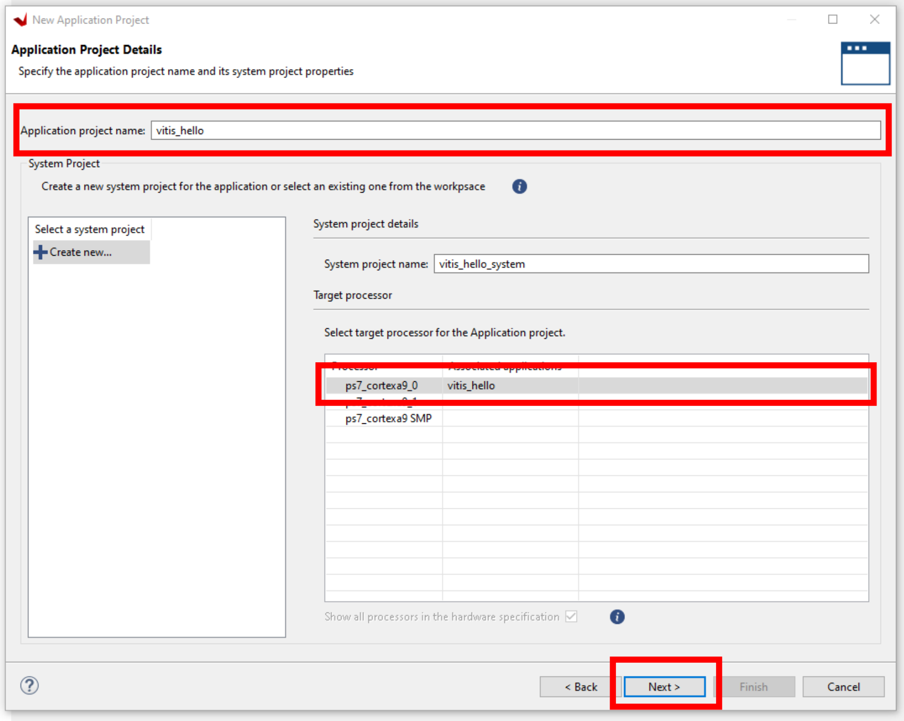
Boot Components

☑ Generate boot components

Platform name: design_1_wrapper

Create the design

- Chose a name for the application and the core in which you want to run the application. Then do Next



New Application Project

Application Project Details

Specify the application project name and its system project properties

Application project name: vitis_hello

System Project

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

Select a system project

System project details

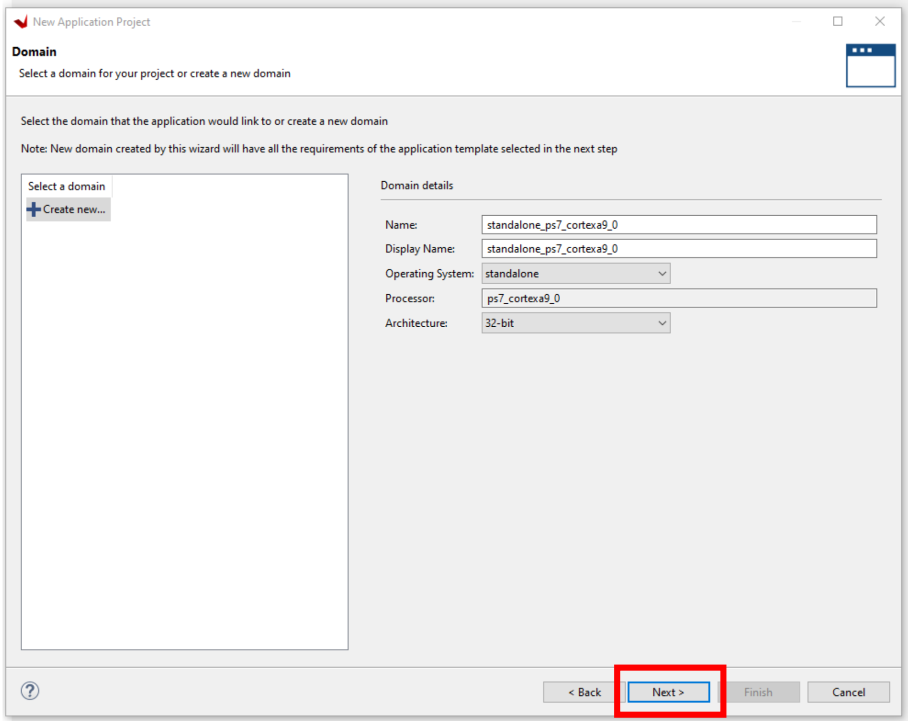
System project name: vitis_hello_system

Target processor

Select target processor for the Application project.

Processor	Associated applications
ps7_cortexa9_0	vitis_hello
ps7_cortexa9_1	
ps7_cortexa9 SMP	

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

Domain details

Name: standalone_ps7_cortexa9_0

Display Name: standalone_ps7_cortexa9_0

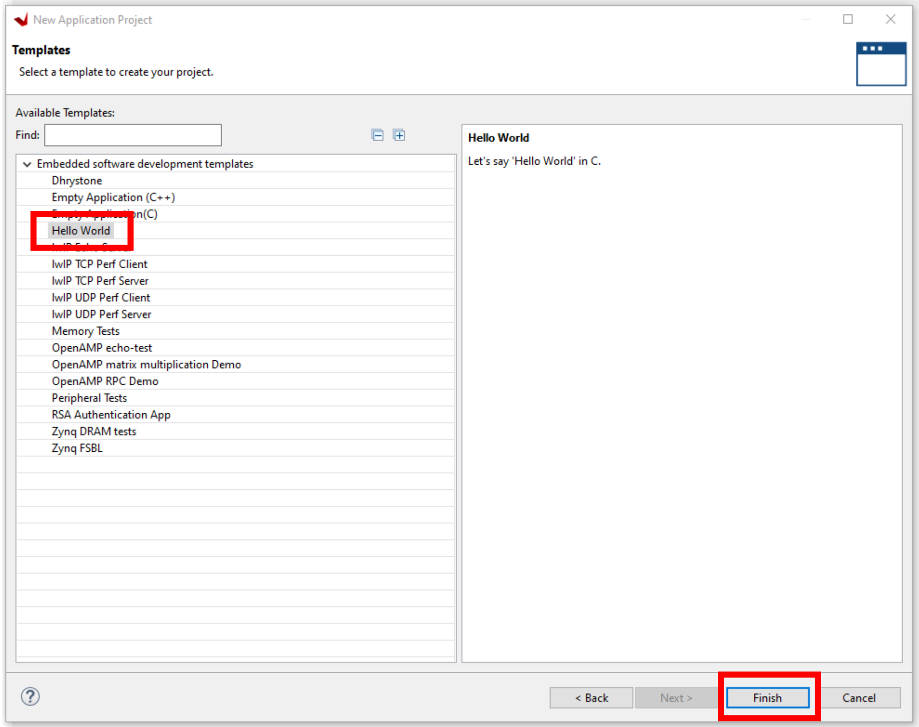
Operating System: standalone

Processor: ps7_cortexa9_0

Architecture: 32-bit

Create the design

- For this example, we will start we a Hello World example
- Select the Hello World example and click finish to create the platform and application project



New Application Project

Templates

Select a template to create your project.

Available Templates:

Find:

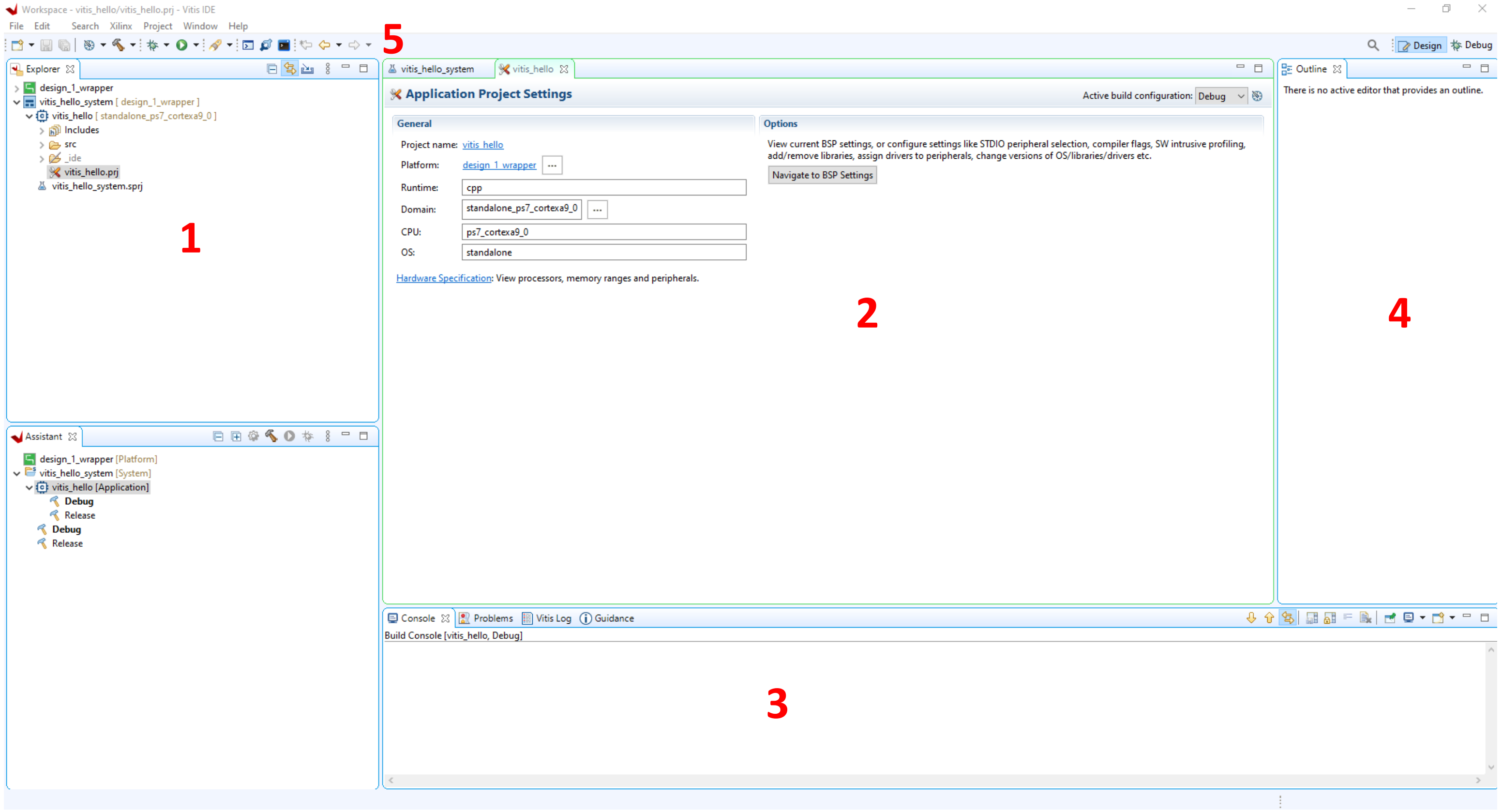
- Embedded software development templates
  - Dhrystone
  - Empty Application (C++)
  - Empty Application(C)
  - Hello World
  - lwIP Echo Server
  - lwIP TCP Perf Client
  - lwIP TCP Perf Server
  - lwIP UDP Perf Client
  - lwIP UDP Perf Server
  - Memory Tests
  - OpenAMP echo-test
  - OpenAMP matrix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

Hello World

Let's say 'Hello World' in C.

Create the design

- Welcome to Vitis!
  - Project explorer (1)
  - File viewer (2)
  - Console, terminal, etc. (3)
  - Outline (4)
  - Toolbar (5)
- All this can be edited and moved around



Application Project Settings

Active build configuration: Debug

General

Project name: vitis_hello

Platform: design_1_wrapper ...

Runtime: cpp

Domain: standalone_ps7_cortexa9_0 ...

CPU: ps7_cortexa9_0

OS: standalone

Hardware Specification: View processors, memory ranges and peripherals.

Options

View current BSP settings, or configure settings like STDIO peripheral selection, compiler flags, SW intrusive profiling, add/remove libraries, assign drivers to peripherals, change versions of OS/libraries/drivers etc.

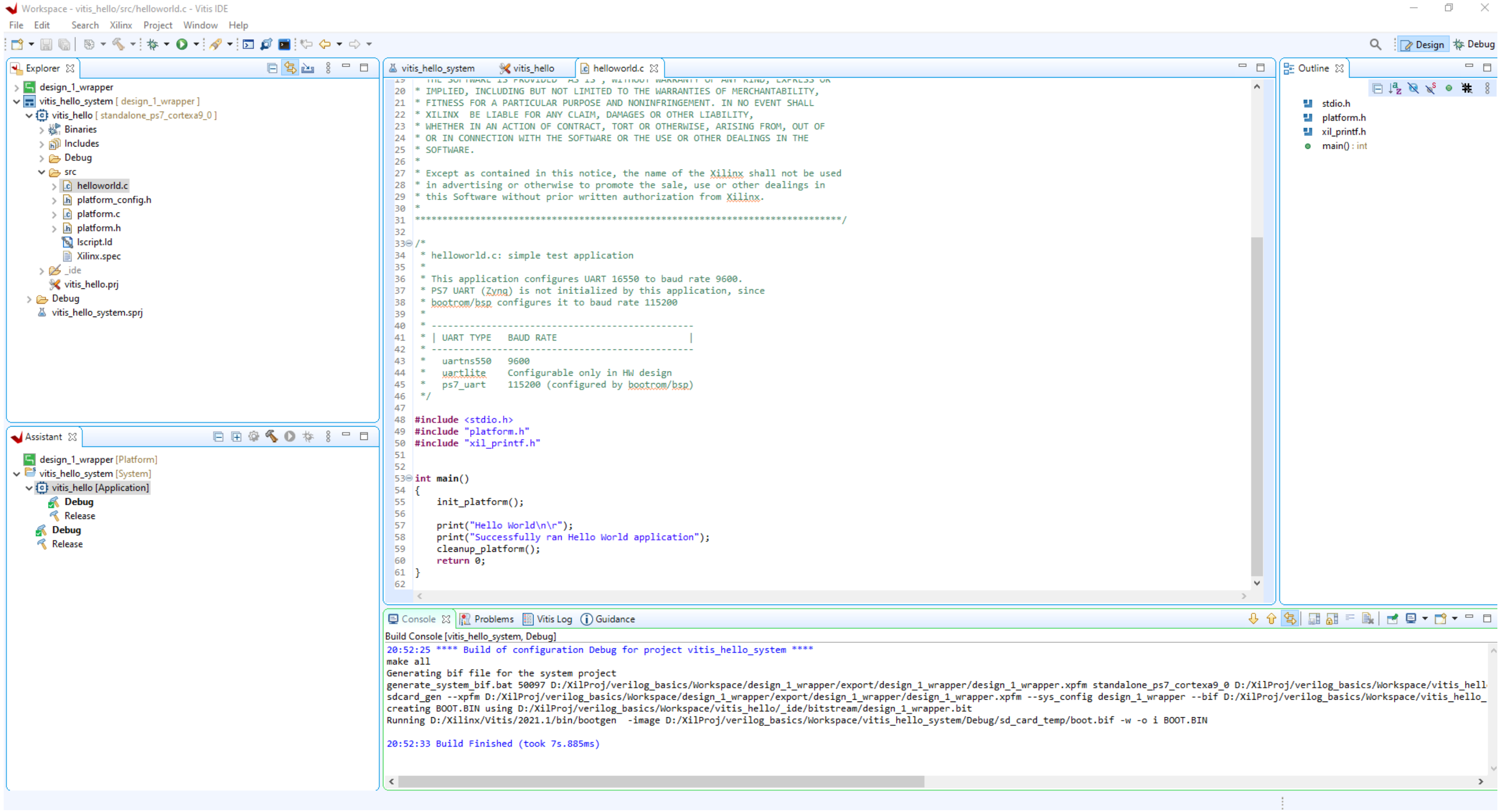
Navigate to BSP Settings

There is no active editor that provides an outline.

Build Console [vitis_hello, Debug]

Create the design

- Open the helloworld.c file



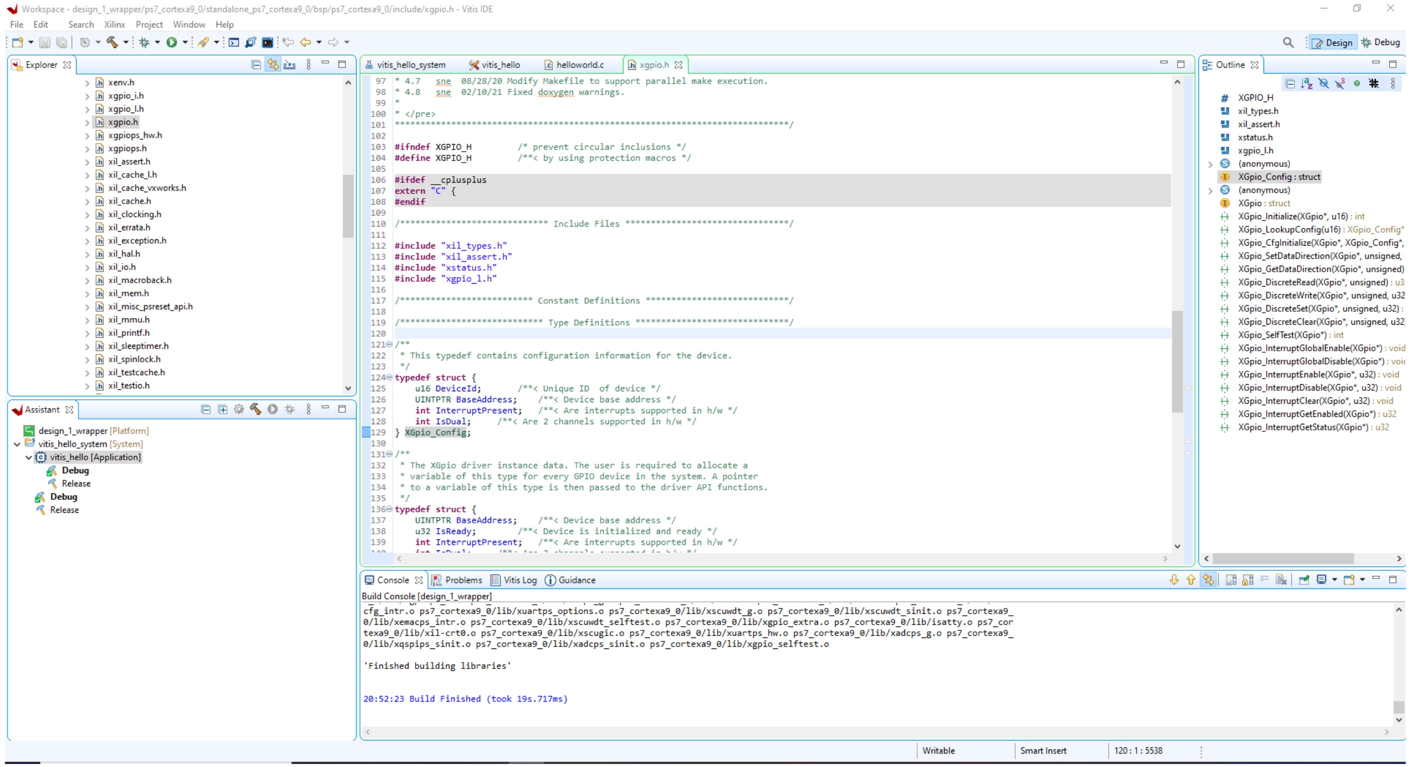
```c
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
 * XILINX  BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY,
 * WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF
 * OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
 * SOFTWARE.
 *
 * Except as contained in this notice, the name of the Xilinx shall not be used
 * in advertising or otherwise to promote the sale, use or other dealings in
 * this Software without prior written authorization from Xilinx.
 *
******************************************************************************/

/*
 * helloworld.c: simple test application
 *
 * This application configures UART 16550 to baud rate 9600.
 * PS7 UART (Zynq) is not initialized by this application, since
 * bootrom/bsp configures it to baud rate 115200
 *
 * ------------------------------------------------
 * | UART TYPE   BAUD RATE                        |
 * ------------------------------------------------
 *   uartns550   9600
 *   uartlite    Configurable only in HW design
 *   ps7_uart    115200 (configured by bootrom/bsp)
 */

#include <stdio.h>
#include "platform.h"
#include "xil_printf.h"


int main()
{
    init_platform();

    print("Hello World\n\r");
    print("Successfully ran Hello World application");
    cleanup_platform();
    return 0;
}
```

Build Console [vitis_hello_system, Debug]

```
20:52:25 **** Build of configuration Debug for project vitis_hello_system ****
make all
Generating bif file for the system project
generate_system_bif.bat 50097 D:/XilProj/verilog_basics/Workspace/design_1_wrapper/export/design_1_wrapper/design_1_wrapper.xpfm standalone_ps7_cortexa9_0 D:/XilProj/verilog_basics/Workspace/vitis_hell
sdcard_gen --xpfm D:/XilProj/verilog_basics/Workspace/design_1_wrapper/export/design_1_wrapper/design_1_wrapper.xpfm --sys_config design_1_wrapper --bif D:/XilProj/verilog_basics/Workspace/vitis_hello_
creating BOOT.BIN using D:/XilProj/verilog_basics/Workspace/vitis_hello/_ide/bitstream/design_1_wrapper.bit
Running D:/Xilinx/Vitis/2021.1/bin/bootgen  -image D:/XilProj/verilog_basics/Workspace/vitis_hello_system/Debug/sd_card_temp/boot.bif -w -o i BOOT.BIN

20:52:33 Build Finished (took 7s.885ms)
```

Create the design

- Explore the include files from your platform (xparameters.h, xgpio.h and xgpio.c)
- What are these files?



Workspace - design_1_wrapper/ps7_cortexa9_0/standalone_ps7_cortexa9_0/bsp/ps7_cortexa9_0/include/xgpio.h - Vitis IDE
File Edit Search Xilinx Project Window Help
Design
Debug
Explorer
xenv.h
xgpio_i.h
xgpio_l.h
xgpio.h
xgpiops_hw.h
xgpiops.h
xil_assert.h
xil_cache_l.h
xil_cache_vxworks.h
xil_cache.h
xil_clocking.h
xil_errata.h
xil_exception.h
xil_hal.h
xil_io.h
xil_macroback.h
xil_mem.h
xil_misc_psreset_api.h
xil_mmu.h
xil_printf.h
xil_sleeptimer.h
xil_spinlock.h
xil_testcache.h
xil_testio.h
Assistant
design_1_wrapper [Platform]
vitis_hello_system [System]
vitis_hello [Application]
Debug
Release
Debug
Release
vitis_hello_system
vitis_hello
helloworld.c
xgpio.h
Outline
XGPIO_H
xil_types.h
xil_assert.h
xstatus.h
xgpio_l.h
(anonymous)
XGpio_Config : struct
(anonymous)
XGpio : struct
XGpio_Initialize(XGpio*, u16) : int
XGpio_LookupConfig(u16) : XGpio_Config*
XGpio_CfgInitialize(XGpio*, XGpio_Config*,
XGpio_SetDataDirection(XGpio*, unsigned,
XGpio_GetDataDirection(XGpio*, unsigned)
XGpio_DiscreteRead(XGpio*, unsigned) : u3
XGpio_DiscreteWrite(XGpio*, unsigned, u32
XGpio_DiscreteSet(XGpio*, unsigned, u32) :
XGpio_DiscreteClear(XGpio*, unsigned, u32
XGpio_SelfTest(XGpio*) : int
XGpio_InterruptGlobalEnable(XGpio*) : void
XGpio_InterruptGlobalDisable(XGpio*) : voi
XGpio_InterruptEnable(XGpio*, u32) : void
XGpio_InterruptDisable(XGpio*, u32) : void
XGpio_InterruptClear(XGpio*, u32) : void
XGpio_InterruptGetEnabled(XGpio*) : u32
XGpio_InterruptGetStatus(XGpio*) : u32
Console
Problems
Vitis Log
Guidance
Writable
Smart Insert
120 : 1 : 5538

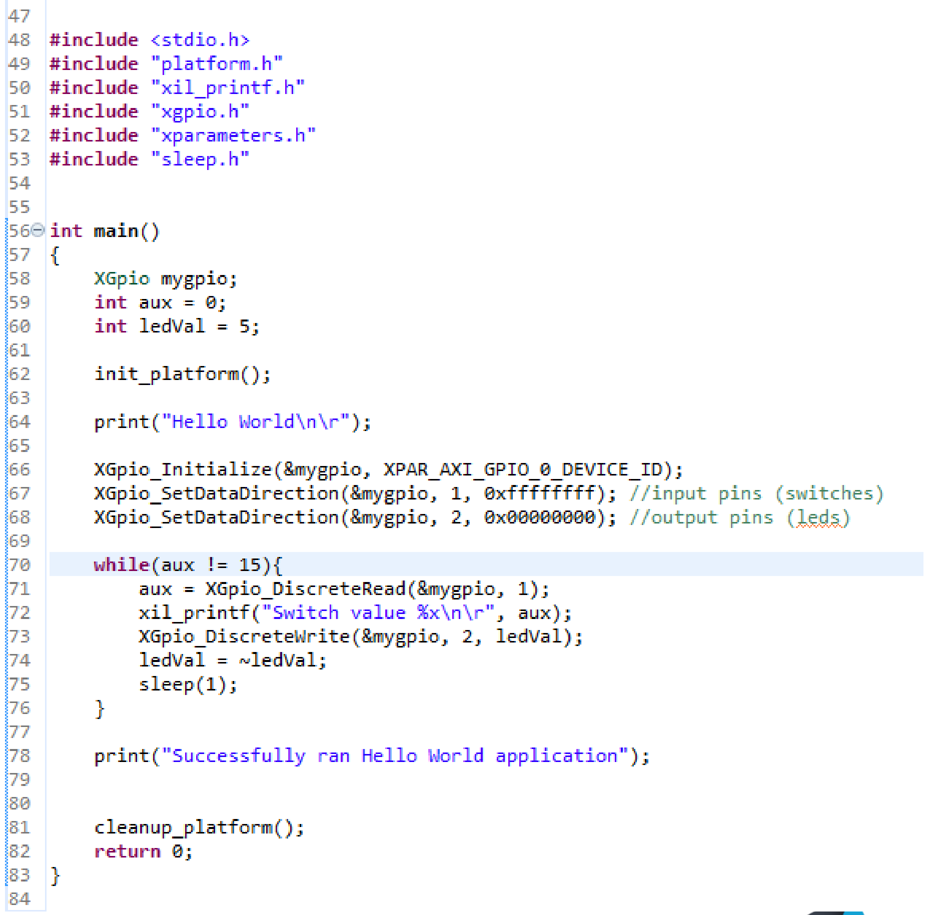
```c
* 4.7   sne  08/28/20 Modify Makefile to support parallel make execution.
* 4.8   sne  02/10/21 Fixed doxygen warnings.
*
* </pre>
******************************************************************************/

#ifndef XGPIO_H         /* prevent circular inclusions */
#define XGPIO_H         /**< by using protection macros */

#ifdef __cplusplus
extern "C" {
#endif

/***************************** Include Files *********************************/

#include "xil_types.h"
#include "xil_assert.h"
#include "xstatus.h"
#include "xgpio_l.h"

/************************** Constant Definitions *****************************/

/**************************** Type Definitions *******************************/

/**
 * This typedef contains configuration information for the device.
 */
typedef struct {
	u16 DeviceId;		/**< Unique ID  of device */
	UINTPTR BaseAddress;	/**< Device base address */
	int InterruptPresent;	/**< Are interrupts supported in h/w */
	int IsDual;		/**< Are 2 channels supported in h/w */
} XGpio_Config;

/**
 * The XGpio driver instance data. The user is required to allocate a
 * variable of this type for every GPIO device in the system. A pointer
 * to a variable of this type is then passed to the driver API functions.
 */
typedef struct {
	UINTPTR BaseAddress;	/**< Device base address */
	u32 IsReady;		/**< Device is initialized and ready */
	int InterruptPresent;	/**< Are interrupts supported in h/w */
```

Build Console [design_1_wrapper]

```
cfg_intr.o ps7_cortexa9_0/lib/xuartps_options.o ps7_cortexa9_0/lib/xscuwdt_g.o ps7_cortexa9_0/lib/xscuwdt_sinit.o ps7_cortexa9_
0/lib/xemacps_intr.o ps7_cortexa9_0/lib/xscuwdt_selftest.o ps7_cortexa9_0/lib/xgpio_extra.o ps7_cortexa9_0/lib/isatty.o ps7_cor
texa9_0/lib/xil-crt0.o ps7_cortexa9_0/lib/xscugic.o ps7_cortexa9_0/lib/xuartps_hw.o ps7_cortexa9_0/lib/xadcps_g.o ps7_cortexa9_
0/lib/xqspips_sinit.o ps7_cortexa9_0/lib/xadcps_sinit.o ps7_cortexa9_0/lib/xgpio_selftest.o

'Finished building libraries'

20:52:23 Build Finished (took 19s.717ms)
```

Create the design

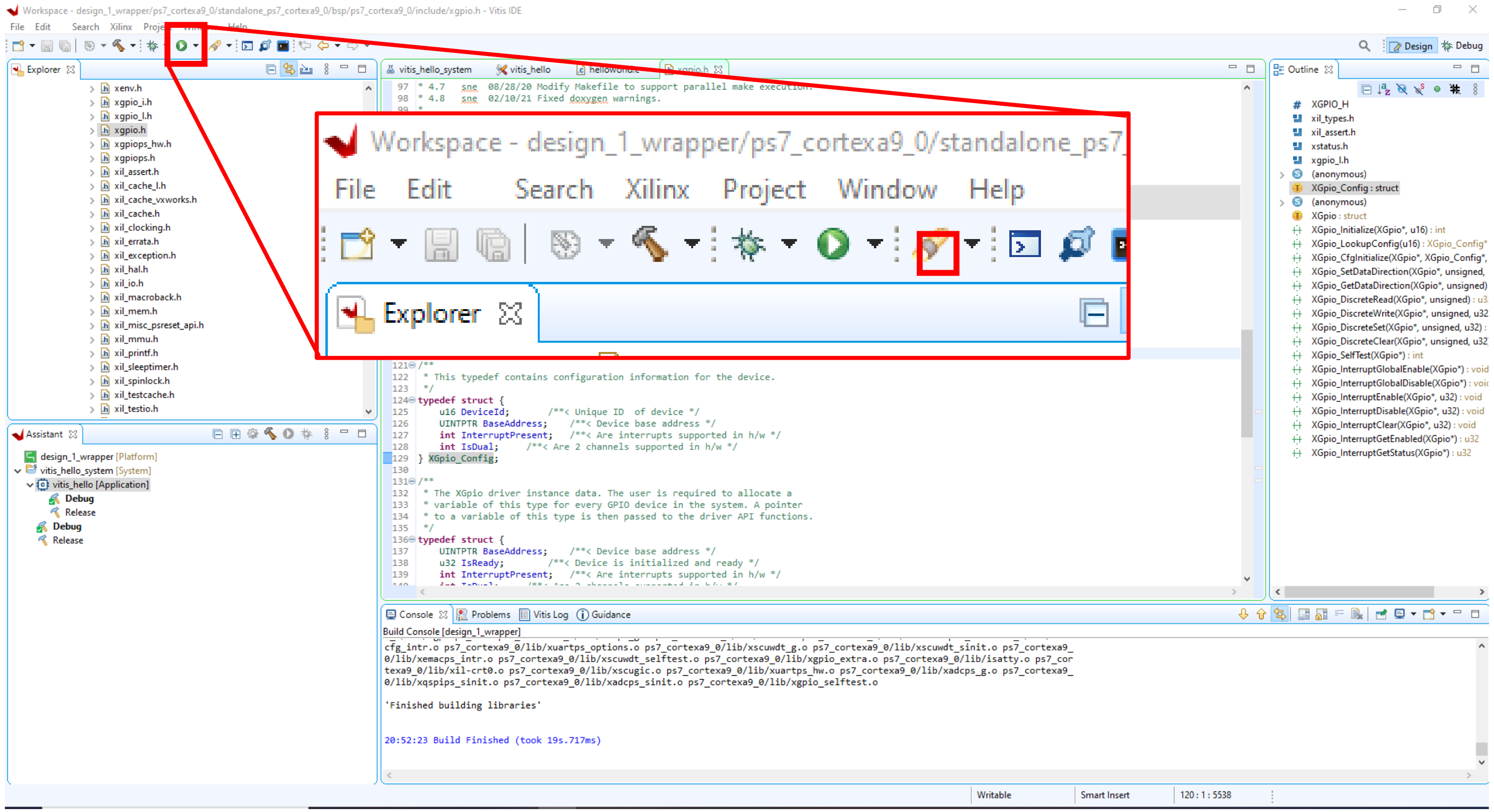
- Open the helloworld.c file and change it with the following code
- Study the code. What does it do?



```c
#include <stdio.h>
#include "platform.h"
#include "xil_printf.h"
#include "xgpio.h"
#include "xparameters.h"
#include "sleep.h"


int main()
{
    XGpio mygpio;
    int aux = 0;
    int ledVal = 5;

    init_platform();

    print("Hello World\n\r");

    XGpio_Initialize(&mygpio, XPAR_AXI_GPIO_0_DEVICE_ID);
    XGpio_SetDataDirection(&mygpio, 1, 0xffffffff); //input pins (switches)
    XGpio_SetDataDirection(&mygpio, 2, 0x00000000); //output pins (leds)

    while(aux != 15){
        aux = XGpio_DiscreteRead(&mygpio, 1);
        xil_printf("Switch value %x\n\r", aux);
        XGpio_DiscreteWrite(&mygpio, 2, ledVal);
        ledVal = ~ledVal;
        sleep(1);
    }

    print("Successfully ran Hello World application");


    cleanup_platform();
    return 0;
}
```

Create the design

- Time to configure the run/debug configuration. Open the Run Configurations menu



```
 * 4.7   sne  08/28/20 Modify Makefile to support parallel make execution.
 * 4.8   sne  02/10/21 Fixed doxygen warnings.
```

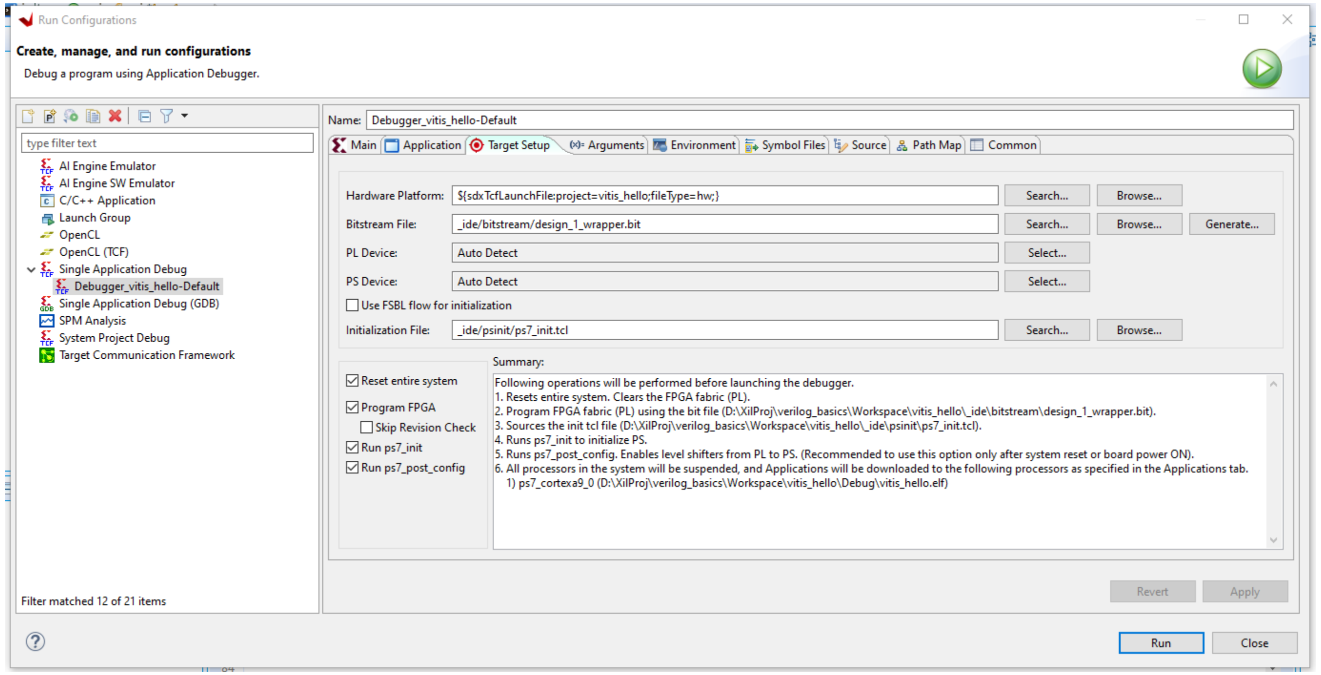
```
/**
 * This typedef contains configuration information for the device.
 */
typedef struct {
	u16 DeviceId;		/**< Unique ID  of device */
	UINTPTR BaseAddress;	/**< Device base address */
	int InterruptPresent;	/**< Are interrupts supported in h/w */
	int IsDual;		/**< Are 2 channels supported in h/w */
} XGpio_Config;

/**
 * The XGpio driver instance data. The user is required to allocate a
 * variable of this type for every GPIO device in the system. A pointer
 * to a variable of this type is then passed to the driver API functions.
 */
typedef struct {
	UINTPTR BaseAddress;	/**< Device base address */
	u32 IsReady;		/**< Device is initialized and ready */
	int InterruptPresent;	/**< Are interrupts supported in h/w */
```

Build Console [design_1_wrapper]

```
cfg_intr.o ps7_cortexa9_0/lib/xuartps_options.o ps7_cortexa9_0/lib/xscuwdt_g.o ps7_cortexa9_0/lib/xscuwdt_sinit.o ps7_cortexa9_
0/lib/xemacps_intr.o ps7_cortexa9_0/lib/xscuwdt_selftest.o ps7_cortexa9_0/lib/xgpio_extra.o ps7_cortexa9_0/lib/isatty.o ps7_cor
texa9_0/lib/xil-crt0.o ps7_cortexa9_0/lib/xscugic.o ps7_cortexa9_0/lib/xuartps_hw.o ps7_cortexa9_0/lib/xadcps_g.o ps7_cortexa9_
0/lib/xqspips_sinit.o ps7_cortexa9_0/lib/xadcps_sinit.o ps7_cortexa9_0/lib/xgpio_selftest.o

'Finished building libraries'


20:52:23 Build Finished (took 19s.717ms)
```

Create the design

- Time to configure the run/debug configuration
- Open the Run Configurations menu and create a new "Simple Application Debug" configuration
- Run it and see the result



Create, manage, and run configurations

Debug a program using Application Debugger.

Name: Debugger_vitis_hello-Default

Hardware Platform: ${sdxTcfLaunchFile:project=vitis_hello;fileType=hw;}

Bitstream File: _ide/bitstream/design_1_wrapper.bit

PL Device: Auto Detect

PS Device: Auto Detect

☐ Use FSBL flow for initialization

Initialization File: _ide/psinit/ps7_init.tcl

☑ Reset entire system

☑ Program FPGA

☐ Skip Revision Check

☑ Run ps7_init

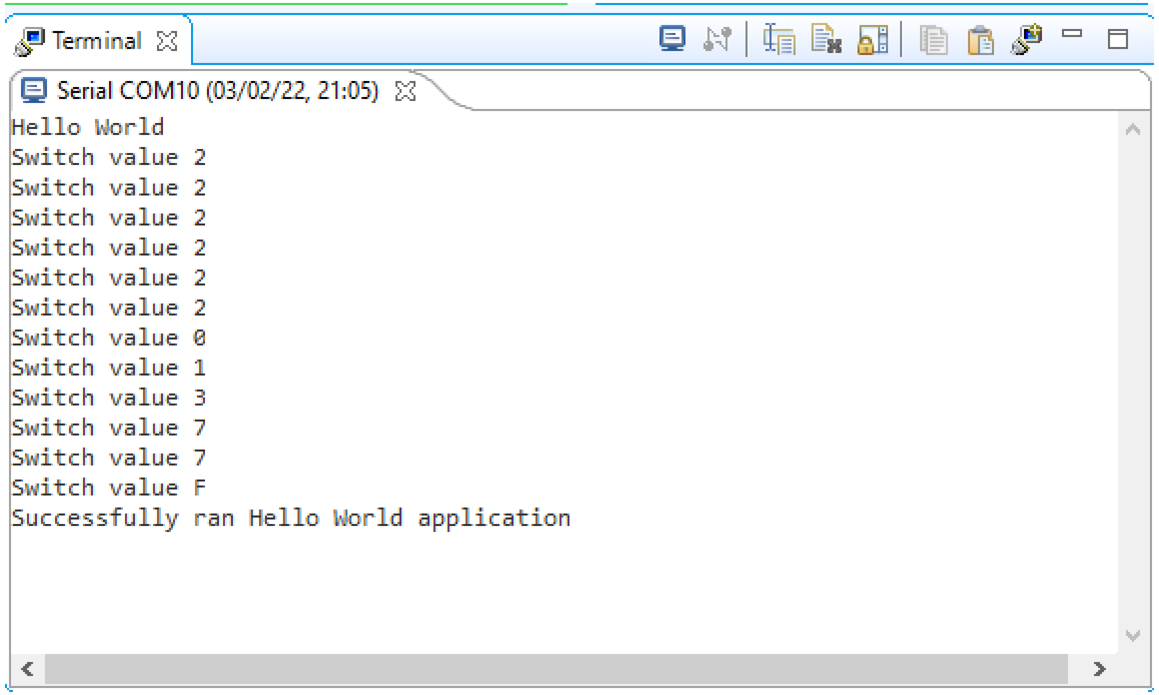
☑ Run ps7_post_config

Summary:

Following operations will be performed before launching the debugger.
1. Resets entire system. Clears the FPGA fabric (PL).
2. Program FPGA fabric (PL) using the bit file (D:\XilProj\verilog_basics\Workspace\vitis_hello_ide\bitstream\design_1_wrapper.bit).
3. Sources the init tcl file (D:\XilProj\verilog_basics\Workspace\vitis_hello_ide\psinit\ps7_init.tcl).
4. Runs ps7_init to initialize PS.
5. Runs ps7_post_config. Enables level shifters from PL to PS. (Recommended to use this option only after system reset or board power ON).
6. All processors in the system will be suspended, and Applications will be downloaded to the following processors as specified in the Applications tab.
   1) ps7_cortexa9_0 (D:\XilProj\verilog_basics\Workspace\vitis_hello\Debug\vitis_hello.elf)

Create the design

- You can configure a terminal (Serial port with a Baud rate of 115200)
- Congratulations. We have done your first application with PS and PL integration



```
Hello World
Switch value 2
Switch value 2
Switch value 2
Switch value 2
Switch value 2
Switch value 2
Switch value 0
Switch value 1
Switch value 3
Switch value 7
Switch value 7
Switch value F
Successfully ran Hello World application
```

THANK YOU!

ANY QUESTIONS?

rui.machado@dtx-colab.pt