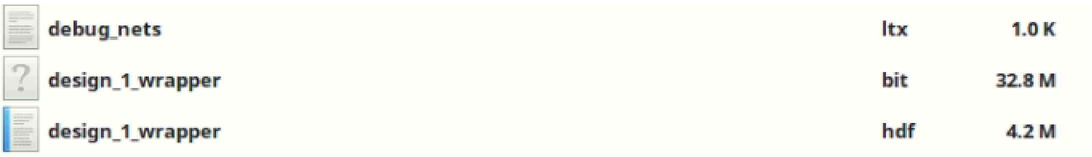
RFSoC and Xilinx SDK – QUICK GUIDE

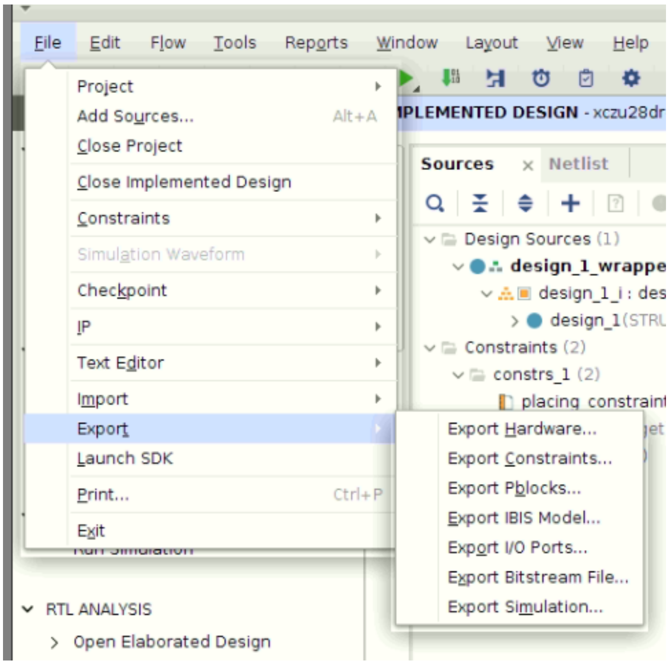
What do yo need from Vivado:

-Bitstream
-Hardware Definition (hdf)
-Debug Nets



debug_nets	ltx	1.0 K
design_1_wrapper	bit	32.8 M
design_1_wrapper	hdf	4.2 M

How to generate the hdf:

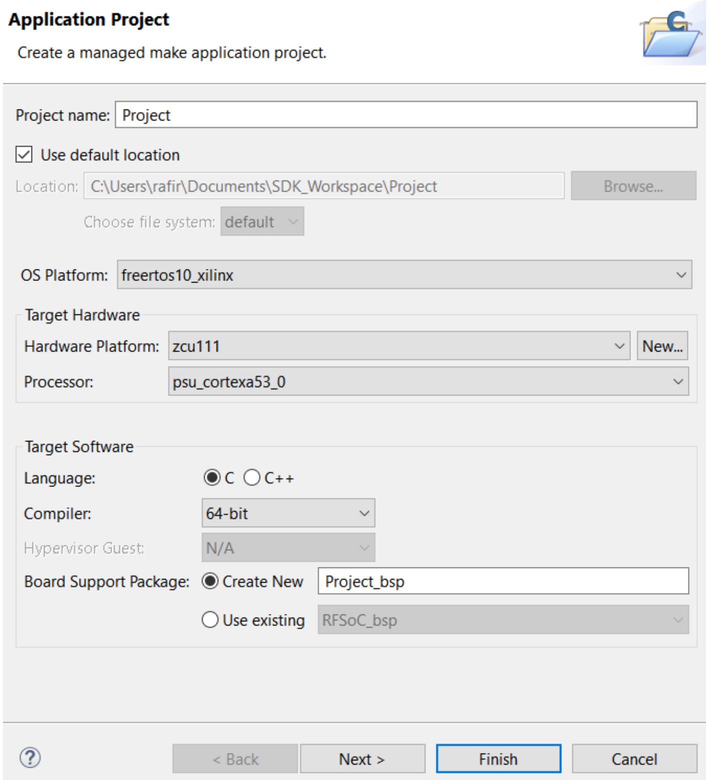


Select export Hardware and you will find the file at .\Nameoftheproject.sdk

What do yo need at SDK?

-Hardware Specification Platform
-Board Package Support (BSP)
-Main project

You will need to generate each of them. Go to File\New\Application Project. The OS platform should be freertos10_xilinx. Also, you will need to define if you are going to create a new BSP or use one that already exists.



Application Project

Create a managed make application project.

Project name: Project

☑ Use default location

Location: C:\Users\rafir\Documents\SDK_Workspace\Project

Choose file system: default

OS Platform: freertos10_xilinx

Target Hardware

Hardware Platform: zcu111

Processor: psu_cortexa53_0

Target Software

Language: ● C ○ C++

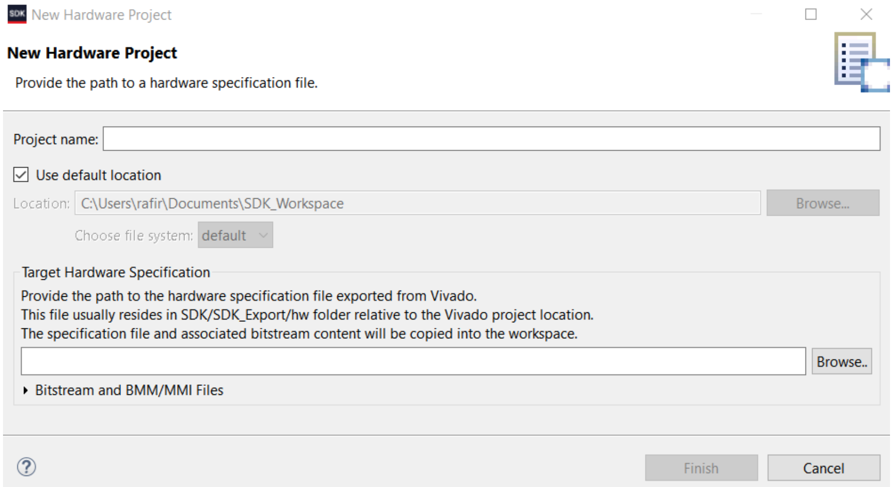
Compiler: 64-bit

Hypervisor Guest: N/A

Board Support Package: ● Create New Project_bsp

○ Use existing RFSoC_bsp

In order to generate a new Hardware Specification, click new Target Hardware and select the bitstream and hdf of your project.



New Hardware Project

New Hardware Project

Provide the path to a hardware specification file.

Project name:

☑ Use default location

Location: C:\Users\rafir\Documents\SDK_Workspace

Choose file system: default

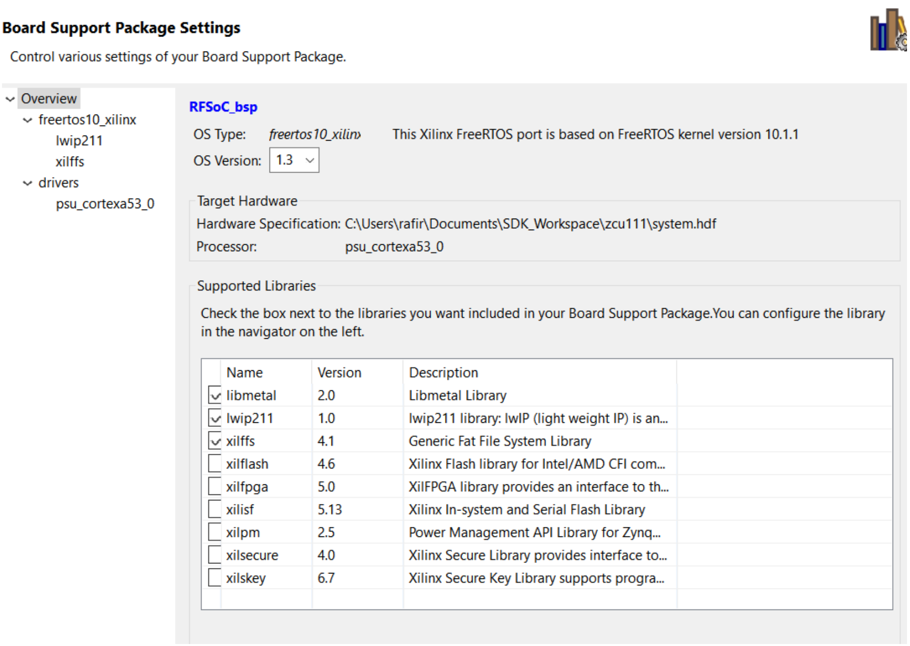
Target Hardware Specification

Provide the path to the hardware specification file exported from Vivado.
This file usually resides in SDK/SDK_Export/hw folder relative to the Vivado project location.
The specification file and associated bitstream content will be copied into the workspace.

Choose an empty application for your project. You can move the files (.c and .h) to the src of the new project.

Board Support Settings

Right click in the BSP generated in the project explorer and select *Board Support Package Settings.* This screen will come up:



Board Support Package Settings

Control various settings of your Board Support Package.

RFSoC_bsp

OS Type: *freertos10_xilinx* This Xilinx FreeRTOS port is based on FreeRTOS kernel version 10.1.1

OS Version: 1.3

Target Hardware

Hardware Specification: C:\Users\rafir\Documents\SDK_Workspace\zcu111\system.hdf

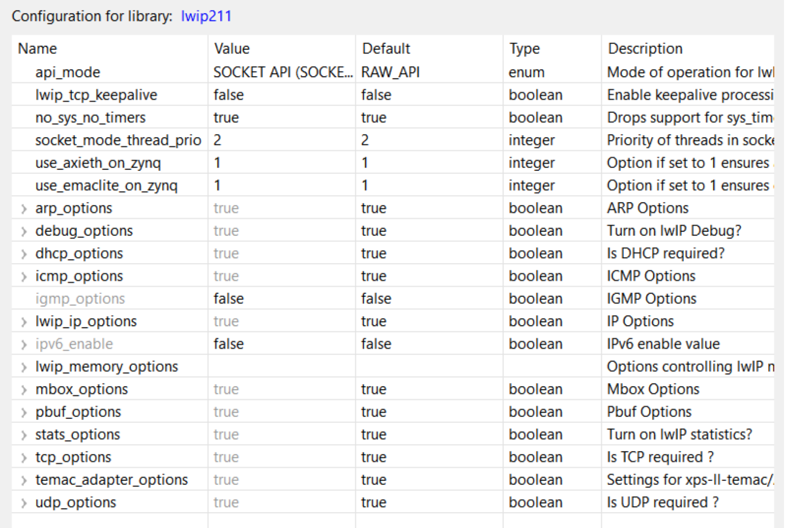
Processor: psu_cortexa53_0

Supported Libraries

Check the box next to the libraries you want included in your Board Support Package.You can configure the library in the navigator on the left.

Name	Version	Description
☑ libmetal	2.0	Libmetal Library
☑ lwip211	1.0	lwip211 library: lwIP (light weight IP) is an...
☑ xilffs	4.1	Generic Fat File System Library
☐ xilflash	4.6	Xilinx Flash library for Intel/AMD CFI com...
☐ xilfpga	5.0	XilFPGA library provides an interface to th...
☐ xilisf	5.13	Xilinx In-system and Serial Flash Library
☐ xilpm	2.5	Power Management API Library for Zynq...
☐ xilsecure	4.0	Xilinx Secure Library provides interface to...
☐ xilskey	6.7	Xilinx Secure Key Library supports progra...

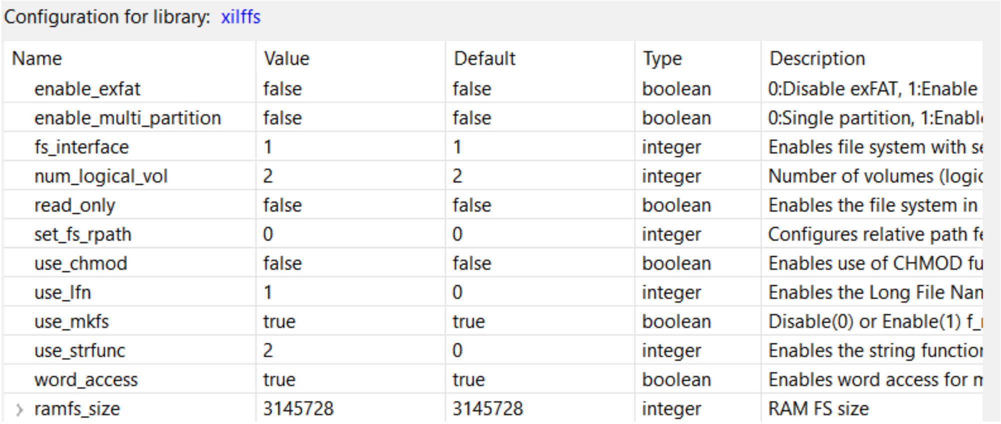
Make sure these libraries are included: libmetal, lwip211 (TCP\IP stack) and xilffs (SD driver).



Configuration for library: lwip211

Name	Value	Default	Type	Description
api_mode	SOCKET API (SOCKE...	RAW_API	enum	Mode of operation for lwI
lwip_tcp_keepalive	false	false	boolean	Enable keepalive processi
no_sys_no_timers	true	true	boolean	Drops support for sys_tim
socket_mode_thread_prio	2	2	integer	Priority of threads in sock
use_axieth_on_zynq	1	1	integer	Option if set to 1 ensures
use_emaclite_on_zynq	1	1	integer	Option if set to 1 ensures
› arp_options	true	true	boolean	ARP Options
› debug_options	true	true	boolean	Turn on lwIP Debug?
› dhcp_options	true	true	boolean	Is DHCP required?
› icmp_options	true	true	boolean	ICMP Options
igmp_options	false	false	boolean	IGMP Options
› lwip_ip_options	true	true	boolean	IP Options
› ipv6_enable	false	false	boolean	IPv6 enable value
› lwip_memory_options				Options controlling lwIP n
› mbox_options	true	true	boolean	Mbox Options
› pbuf_options	true	true	boolean	Pbuf Options
› stats_options	true	true	boolean	Turn on lwIP statistics?
› tcp_options	true	true	boolean	Is TCP required ?
› temac_adapter_options	true	true	boolean	Settings for xps-ll-temac/
› udp_options	true	true	boolean	Is UDP required ?

Go to lwip211 configuration and change the *api_mode* to SOCKET.



Configuration for library: xilffs

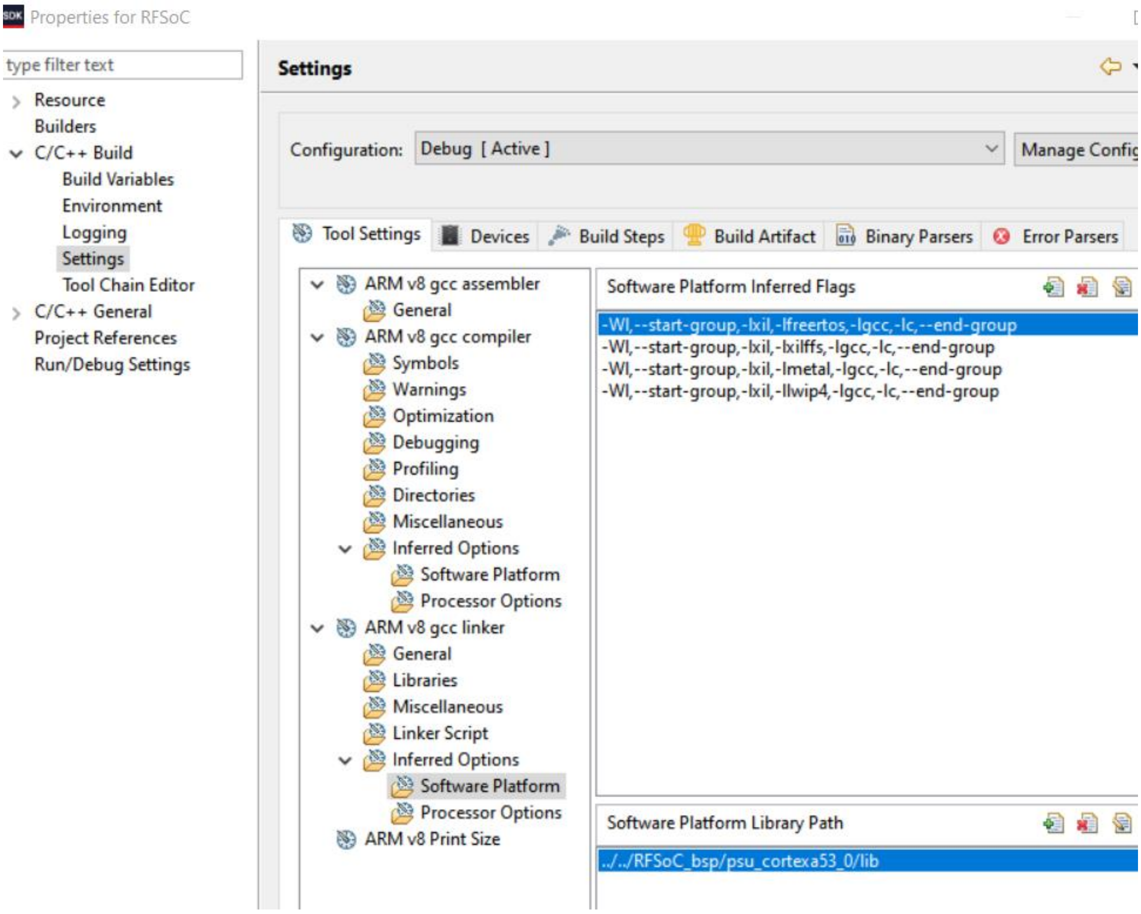
Name	Value	Default	Type	Description
enable_exfat	false	false	boolean	0:Disable exFAT, 1:Enable
enable_multi_partition	false	false	boolean	0:Single partition, 1:Enabl
fs_interface	1	1	integer	Enables file system with se
num_logical_vol	2	2	integer	Number of volumes (logic
read_only	false	false	boolean	Enables the file system in
set_fs_rpath	0	0	integer	Configures relative path fe
use_chmod	false	false	boolean	Enables use of CHMOD fu
use_lfn	1	0	integer	Enables the Long File Nan
use_mkfs	true	true	boolean	Disable(0) or Enable(1) f_
use_strfunc	2	0	integer	Enables the string functior
word_access	true	true	boolean	Enables word access for n
› ramfs_size	3145728	3145728	integer	RAM FS size

For the xilffs library you need to modify *use_strfunc* to 2 and *use_lfn* to 1.

Project Settings

Go to project properties (Right click/Properties), then in the C/C++ Build Menu go to settings. Make sure the linker flags of the libraries appear in the Software Platform Inferred Flags Menu.

```
-Wl,--start-group,-lxil,-lfreertos,-lgcc,-lc,--end-group
-Wl,--start-group,-lxil,-lxilffs,-lgcc,-lc,--end-group
-Wl,--start-group,-lxil,-lmetal,-lgcc,-lc,--end-group
-Wl,--start-group,-lxil,-llwip4,-lgcc,-lc,--end-group
```



Configuration: Debug [Active]

Software Platform Inferred Flags

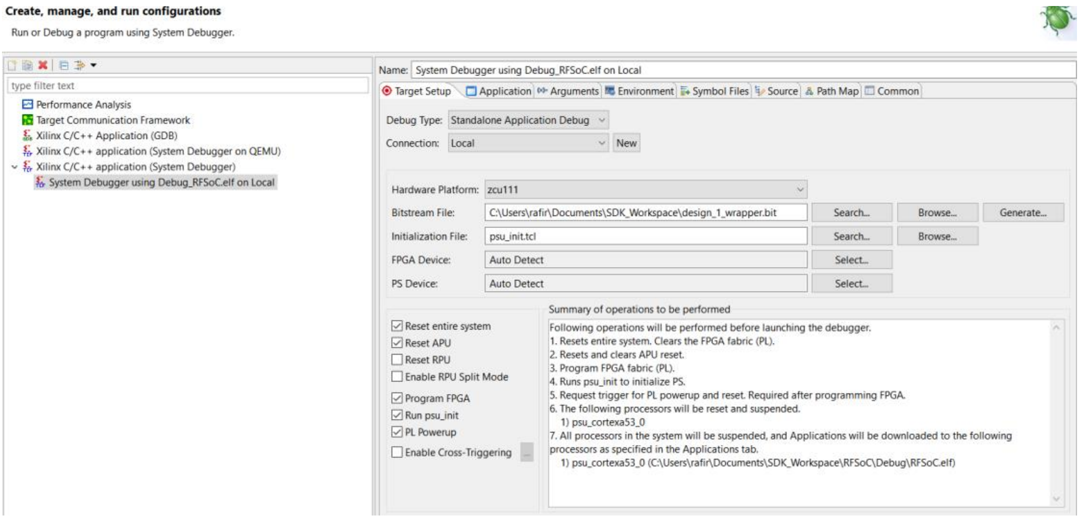
```
-Wl,--start-group,-lxil,-lfreertos,-lgcc,-lc,--end-group
-Wl,--start-group,-lxil,-lxilffs,-lgcc,-lc,--end-group
-Wl,--start-group,-lxil,-lmetal,-lgcc,-lc,--end-group
-Wl,--start-group,-lxil,-llwip4,-lgcc,-lc,--end-group
```

Software Platform Library Path

../../RFSoC_bsp/psu_cortexa53_0/lib

Debug Configurations

Its time to setup the debug configurations. Select the System Debugger mode.


type filter text
Performance Analysis
Target Communication Framework
Xilinx C/C++ Application (GDB)
Xilinx C/C++ application (System Debugger on QEMU)
Xilinx C/C++ application (System Debugger)
System Debugger using Debug_RFSoC.elf on Local
Target Setup
Application
Arguments
Environment
Symbol Files
Source
Path Map
Common
New
Search...
Browse...
Generate...
Search...
Browse...
Select...
Select...

Create, manage, and run configurations

Run or Debug a program using System Debugger.

Name: System Debugger using Debug_RFSoC.elf on Local

Debug Type: Standalone Application Debug

Connection: Local

Hardware Platform: zcu111

Bitstream File: C:\Users\rafir\Documents\SDK_Workspace\design_1_wrapper.bit

Initialization File: psu_init.tcl

FPGA Device: Auto Detect

PS Device: Auto Detect

- ☑ Reset entire system
- ☑ Reset APU
- □ Reset RPU
- □ Enable RPU Split Mode
- ☑ Program FPGA
- ☑ Run psu_init
- ☑ PL Powerup
- □ Enable Cross-Triggering

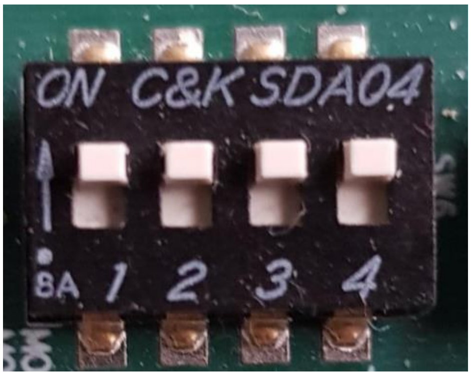
Summary of operations to be performed

```
Following operations will be performed before launching the debugger.
1. Resets entire system. Clears the FPGA fabric (PL).
2. Resets and clears APU reset.
3. Program FPGA fabric (PL).
4. Runs psu_init to initialize PS.
5. Request trigger for PL powerup and reset. Required after programming FPGA.
6. The following processors will be reset and suspended.
   1) psu_cortexa53_0
7. All processors in the system will be suspended, and Applications will be downloaded to the following
processors as specified in the Applications tab.
   1) psu_cortexa53_0 (C:\Users\rafir\Documents\SDK_Workspace\RFSoC\Debug\RFSoC.elf)
```

You need to define the Hardware platform, the bitstream and the application. An .elf file must be generated by building the project before going to debug mode. To select the application you are going to flash to the PS, select the Application tab and choose the executable file.

Flash the Board

Connect the board by JTAG through its micro USB port. Put the SW6 to this configuration in order to switch to JTAG Flash Mode.



Click the debug button 🐞▾ and the project will change to Debug perspective. The FPGA will be programed. To run the application click ▶ to start.