FreeRTOS/lwIP (XAPP1026) for Xilinx Zynq devices using Vivado 2014.2

This port is compatible with Xilinx Vivado 2014.2 and was tested on a Digilent Zedboard, but should work on a Xilinx ZC702 board also.

The FreeRTOS port is modified from v1_02_a found on this forum. It is based on version 7.0.2 of FreeRTOS.

The lwIP library is modified from v2.1 found in the Vivado 2014.2 tools. The lwIP library is referenced in the FreeRTOS .mss file directly to work properly and is modified to fix a bug in the TCL script that does not setup the lwipopts.h file properly for FreeRTOS.

The application note from Xilinx can be found here:

XAPP1026 PDF

Using FreeRTOS in the Xilinx SDK environment

A stand-alone board support package (BSP) is a library generated by the Xilinx SDK that is specific to a hardware design. It contains initialization code for bringing up the ARM CPUs in Zynq and also contains software drivers for all available Zynq peripherals. However it is not FreeRTOS aware.

The FreeRTOS port provided in this package extends the stand-alone BSP described above to also include FreeRTOS source files. After using this port in a Xilinx SDK environment, the user gets all the FreeRTOS source files in a FreeRTOS BSP library. This library uses the Xilinx SDK generated stand-alone BSP library. None of the standalone drivers included in the stand-alone BSP library is thread-safe. The demo applications provided as part of this package are based on the FreeRTOS BSP.

The Xilinx Zynq repository in this package has the following structure –

```
FreeRTOS_Zynq_Vivado
¦
+-sw Repository used to integrate FreeRTOS related files and related apps in to SDK
¦
+- repo
¦
```

+- bsp ***FreeRTOS and lwip library Source files***
¦
+- sw_apps ***Contains Apps for Hello World, Blink LED using Semaphore, Blink LED using Mutex , lwip socket, and lwip raw IO apps***

The **bsp** folder contains all FreeRTOS and lwip source files. These source files include the generic FreeRTOS source (that this port has not changed) and Zynq related source files (which are newly written as per the Zynq hardware requirements).

The **sw_apps** contains demo applications that the user can run to test the FreeRTOS port. It contains a simple hello world application that prints messages from multiple tasks. It contains four LED examples that blink a LED on ZC702 board or Zedboard, two using mutex and the other two using timer and semaphore. There are also two apps to test the lwIP library, a standalone raw I/O version not using FreeRTOS and a socket I/O version that uses FreeRTOS.

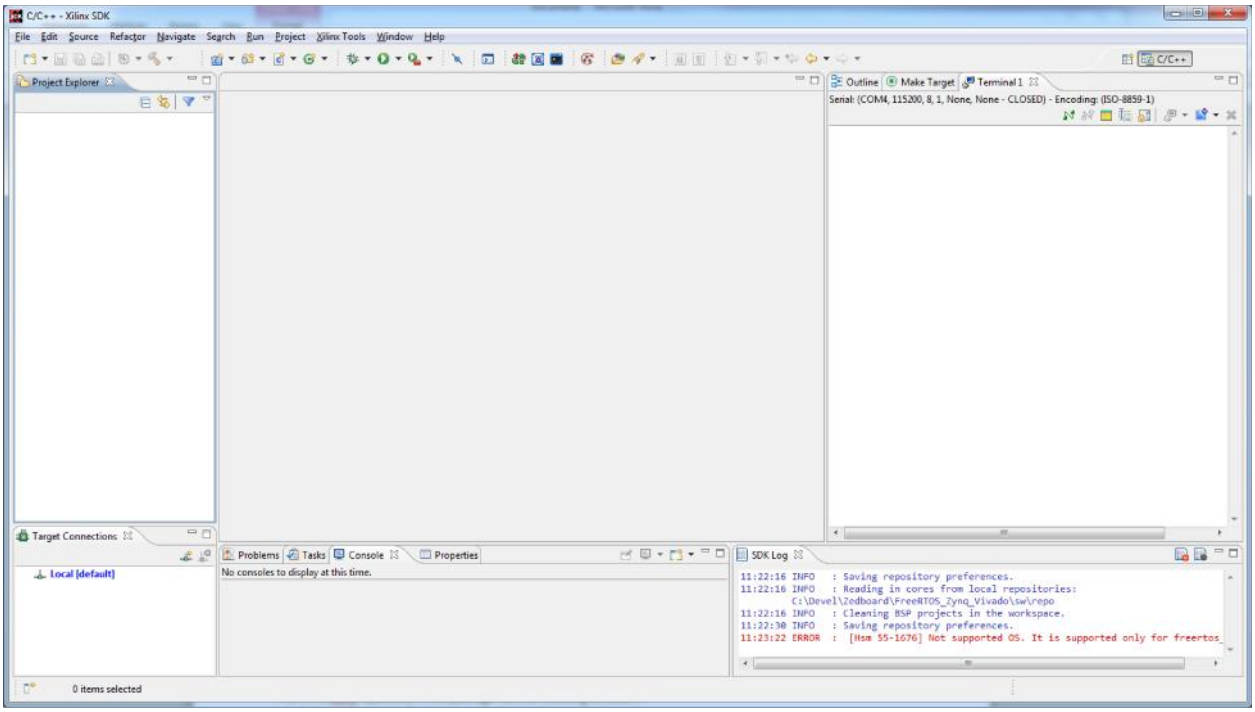
This repository has be lightly tested but at least works in Vivado 2014.2 and actually talks TCP/IP and UDP over a working Ethernet link.

Create FreeRTOS Application and BSP using Xilinx SDK

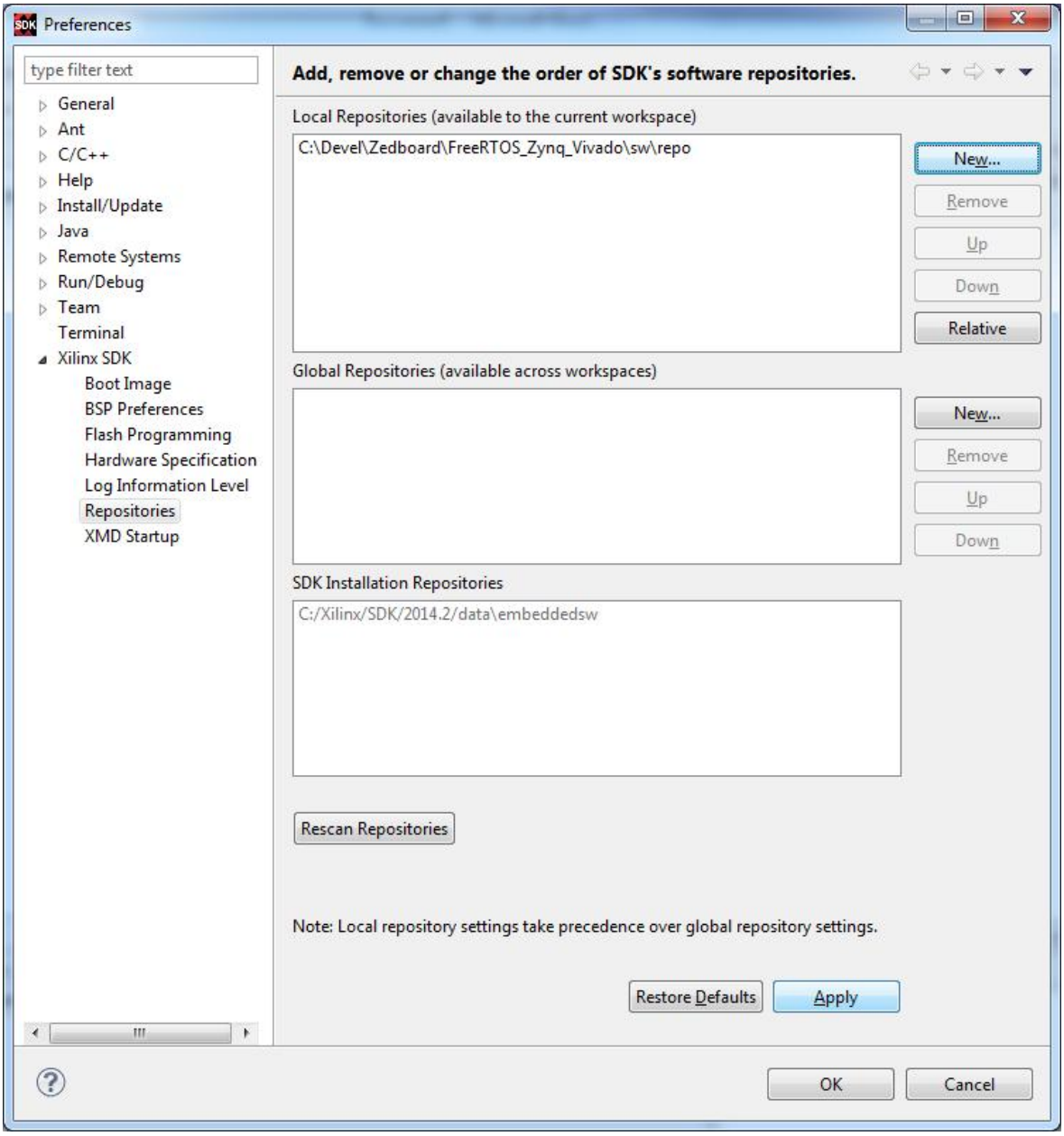
Extract the zip file available in the package to get the FreeRTOS_Zynq_Vivado directory.

Create a folder for SDK workspace with a suitable name, say *sdk_projects*.

Open SDK with *sdk_projects* as workspace.



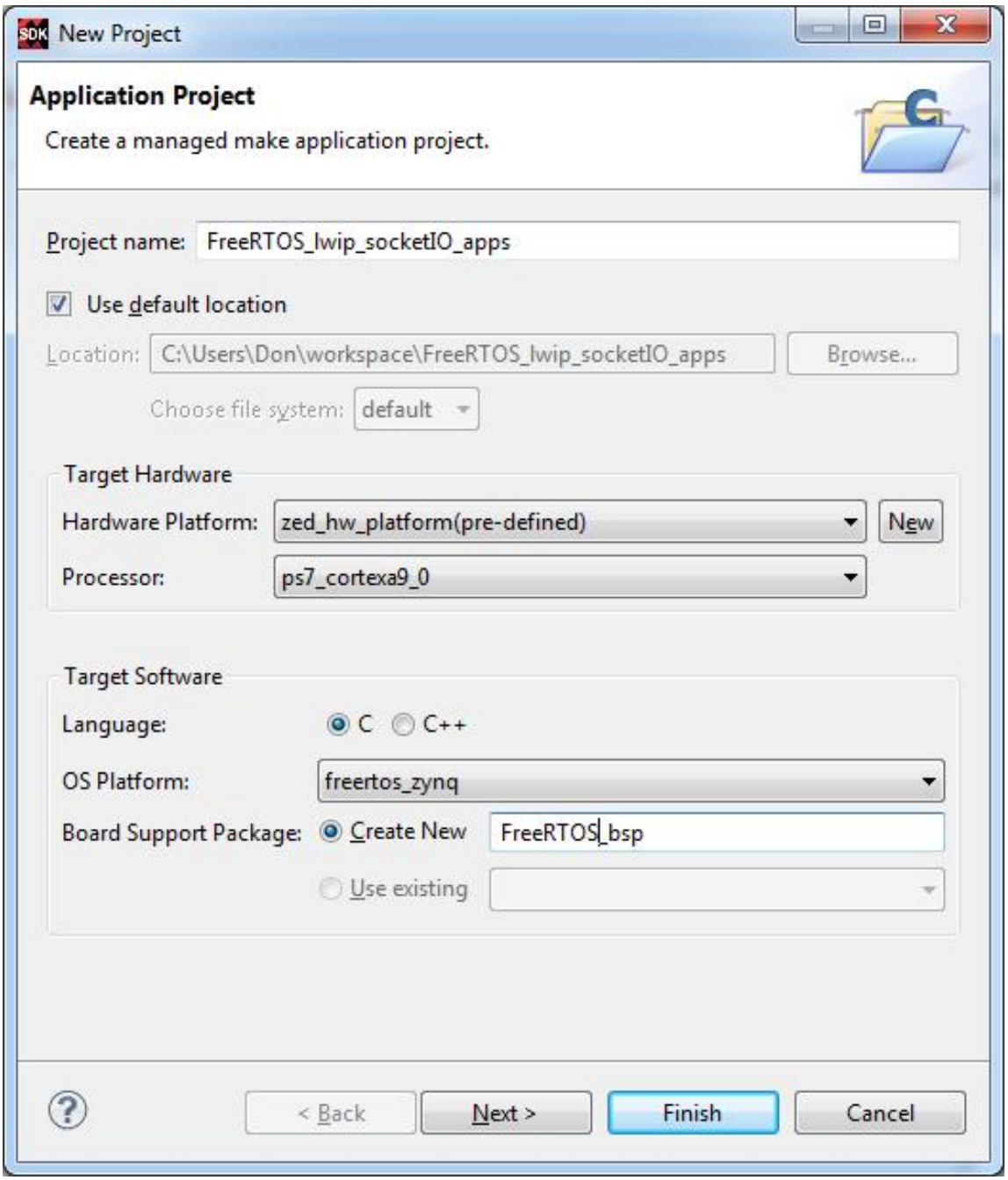
Select Xilinx Tools -> Repositories. Preferences Window pops up.
Click New under Local Repositories section and give the path to FreeRTOS_Zynq_Vivado/sw/repo/ directory.
Click Rescan Repositories, then select Apply and then OK. This will ensure that the Xilinx SDK knows about the FreeRTOS and lwIP BSPs and the applications available to it.


Preferences
type filter text
General
Ant
C/C++
Help
Install/Update
Java
Remote Systems
Run/Debug
Team
Terminal
Xilinx SDK
Boot Image
BSP Preferences
Flash Programming
Hardware Specification
Log Information Level
Repositories
XMD Startup
Add, remove or change the order of SDK's software repositories.
Local Repositories (available to the current workspace)
C:\Devel\Zedboard\FreeRTOS_Zynq_Vivado\sw\repo
New...
Remove
Up
Down
Relative
Global Repositories (available across workspaces)
New...
Remove
Up
Down
SDK Installation Repositories
C:/Xilinx/SDK/2014.2/data\embeddedsw
Rescan Repositories
Note: Local repository settings take precedence over global repository settings.
Restore Defaults
Apply
OK
Cancel

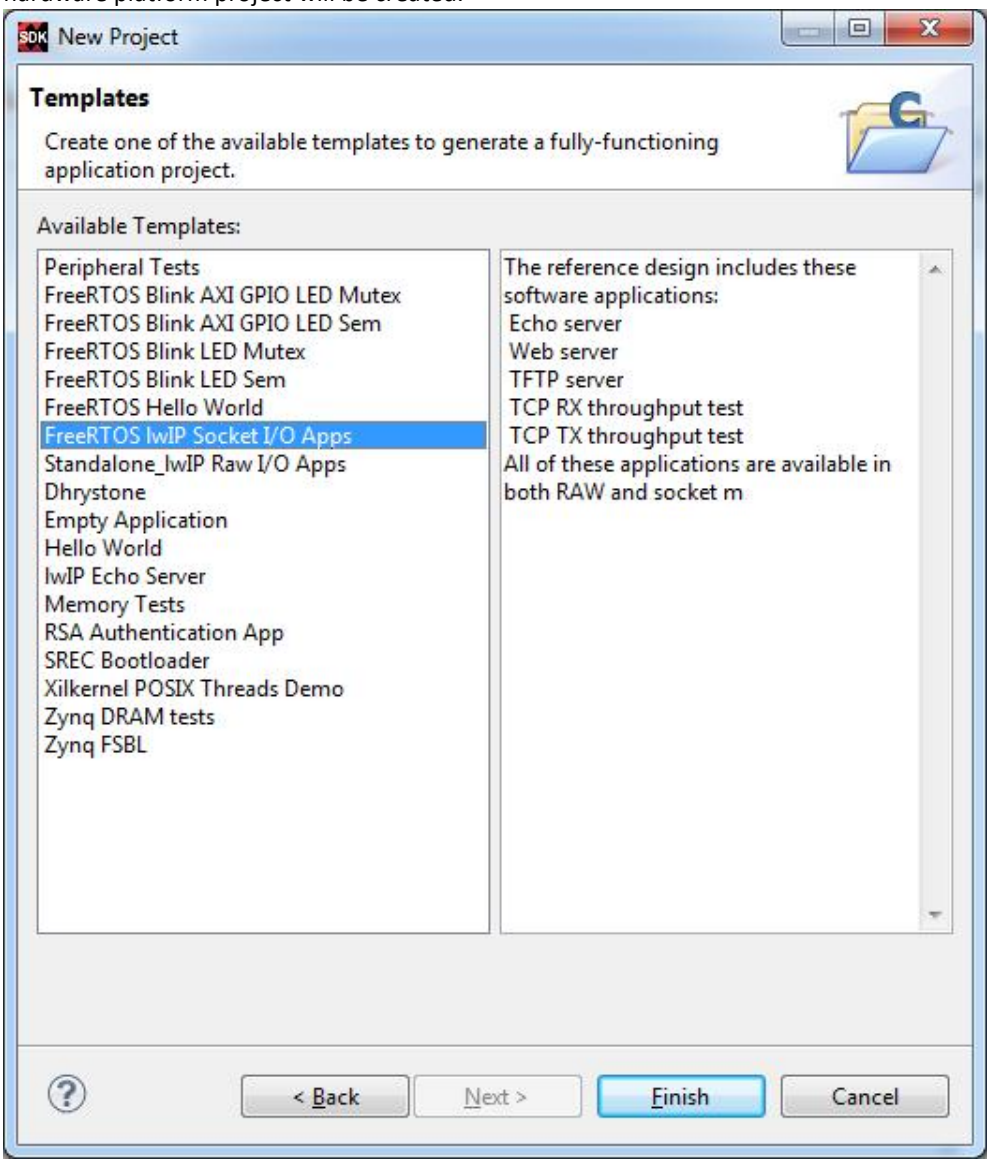
Choose File -> New -> Application Project -> Select.
New Application Project window opens up. Provide Project Name.

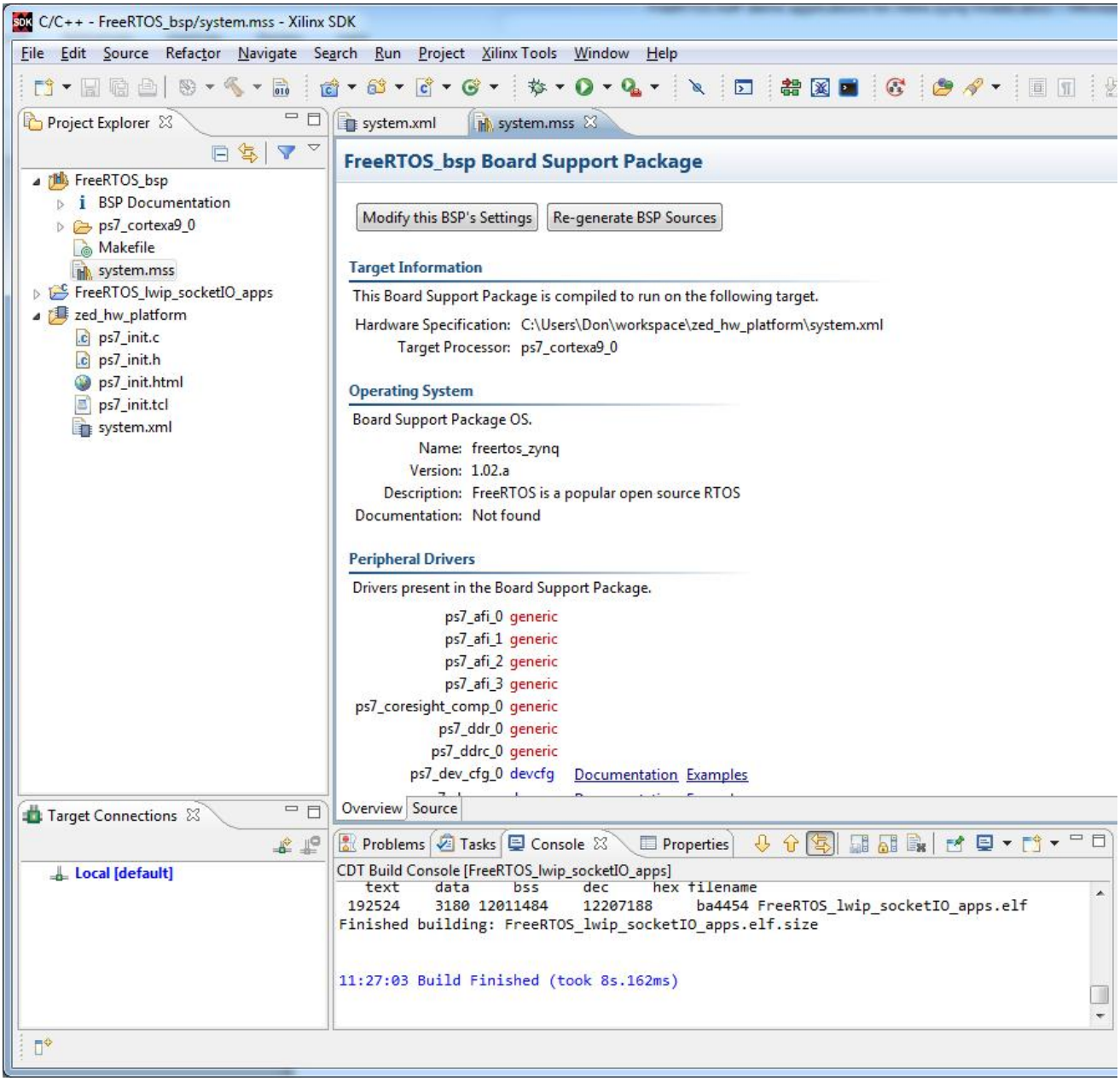
Under Target Hardware tab, choose a Hardware Platform from dropdown list against Hardware Platform attribute. Choice can be made to use pre-defined hardware platforms or create new hardware project, say zed_hw_platform. Choose the processor for which the application should be targeted.

Under Target Software tab, select freertos_zynq as OS Platform. Name for Board Support Package will be populated based on the application project name. Accept the default or edit and ... Click Next.



Select any of the available FreeRTOS applications, such as FreeRTOS lwIP Socket I/O Apps. Click Finish. Three sdk projects – FreeRTOS lwIP socket application with the provided project name, board support package and Zedboard hardware platform project will be created.




C/C++ - FreeRTOS_bsp/system.mss - Xilinx SDK
File Edit Source Refactor Navigate Search Run Project Xilinx Tools Window Help
Project Explorer
FreeRTOS_bsp
BSP Documentation
ps7_cortexa9_0
Makefile
system.mss
FreeRTOS_lwip_socketIO_apps
zed_hw_platform
ps7_init.c
ps7_init.h
ps7_init.html
ps7_init.tcl
system.xml
system.xml
system.mss
FreeRTOS_bsp Board Support Package
Modify this BSP's Settings
Re-generate BSP Sources
Target Information
This Board Support Package is compiled to run on the following target.
Hardware Specification: C:\Users\Don\workspace\zed_hw_platform\system.xml
Target Processor: ps7_cortexa9_0
Operating System
Board Support Package OS.
Name: freertos_zynq
Version: 1.02.a
Description: FreeRTOS is a popular open source RTOS
Documentation: Not found
Peripheral Drivers
Drivers present in the Board Support Package.
ps7_afi_0 generic
ps7_afi_1 generic
ps7_afi_2 generic
ps7_afi_3 generic
ps7_coresight_comp_0 generic
ps7_ddr_0 generic
ps7_ddrc_0 generic
ps7_dev_cfg_0 devcfg Documentation Examples
Overview Source
Target Connections
Local [default]
Problems Tasks Console Properties
CDT Build Console [FreeRTOS_lwip_socketIO_apps]
text data bss dec hex filename
192524 3180 12011484 12207188 ba4454 FreeRTOS_lwip_socketIO_apps.elf
Finished building: FreeRTOS_lwip_socketIO_apps.elf.size
11:27:03 Build Finished (took 8s.162ms)

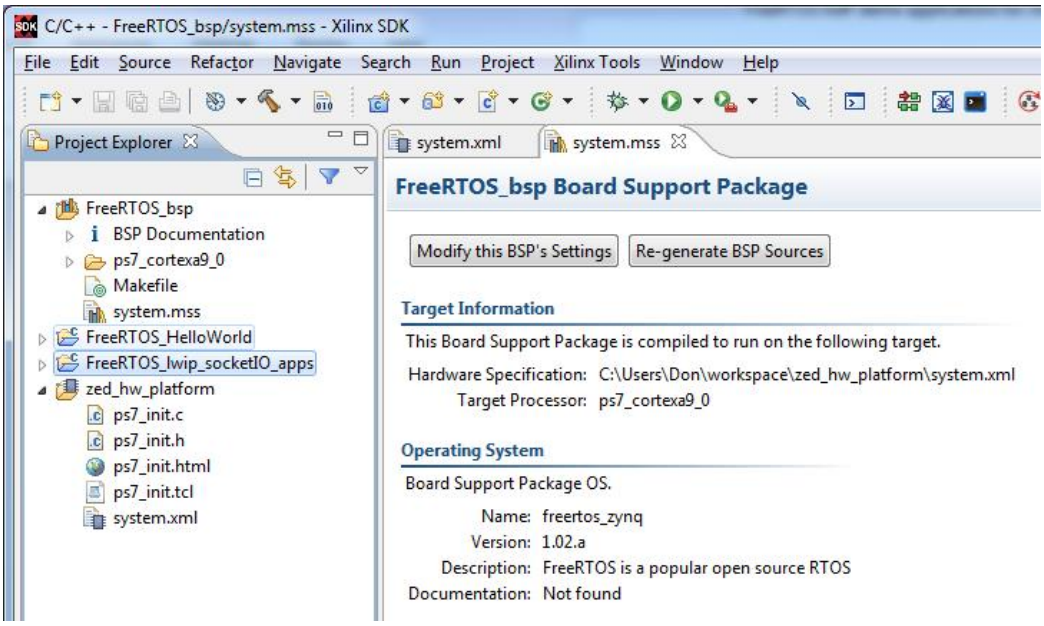
Create other FreeRTOS APPS

Choose File -> New -> Application Project -> Select.

Give Project Name, choose OS Platform as FreeRTOS Zynq and for Board Support Package select the radio button ‘Use existing’, which points to already created BSP project. Click Next.

Choose one of the FreeRTOS Applications available. Click Finish.

FreeRTOS Hello World – It simply creates two tasks with a print statement in each and of equal priorities. The prints should be observed on terminal according to scheduling policy. Expected output is prints from both tasks one after another.



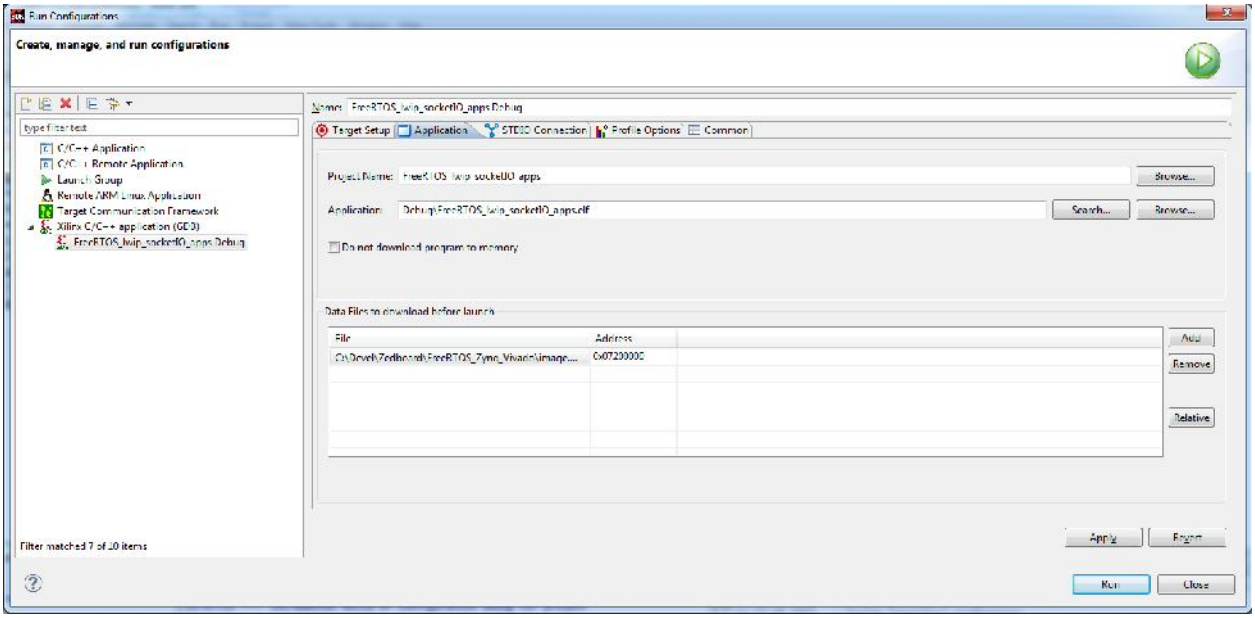
Set up the serial terminal in the SDK

Choose Window -> Show View -> Terminal if the terminal window is not already displayed. Select the connection to be serial and the baud rate to be 115k.

Execute the lwIP FreeRTOS APPS

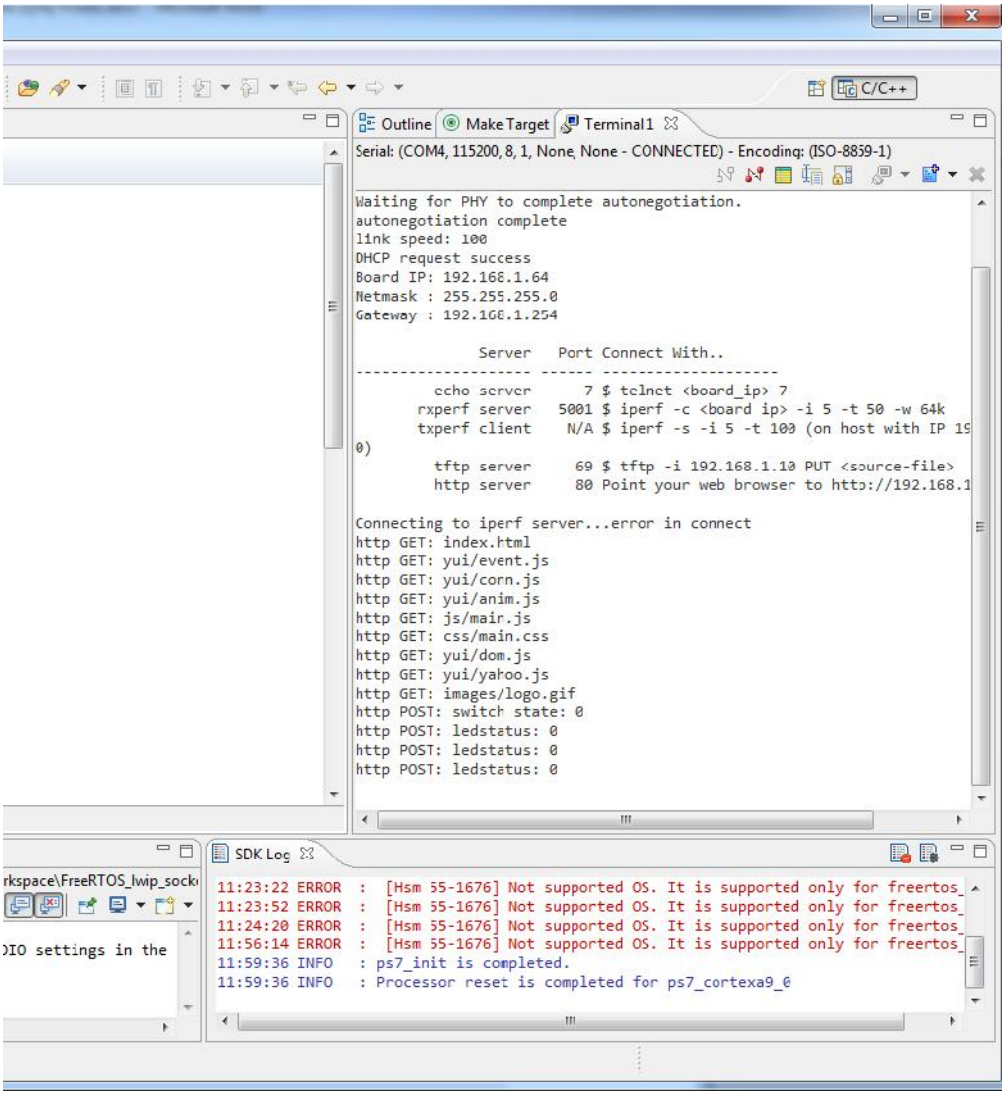
Choose Run -> Run Configurations …

Under Xilinx C/ C++ application select New from the context menu and choose the lwip Socket app created earlier. In the Application tab click the “Add” button and browse to the location of the image.mfs file from the XAPP1026 files retrieved from the Xilinx website. Set the address to 0x7200000 and hit “Apply” then “Run”.

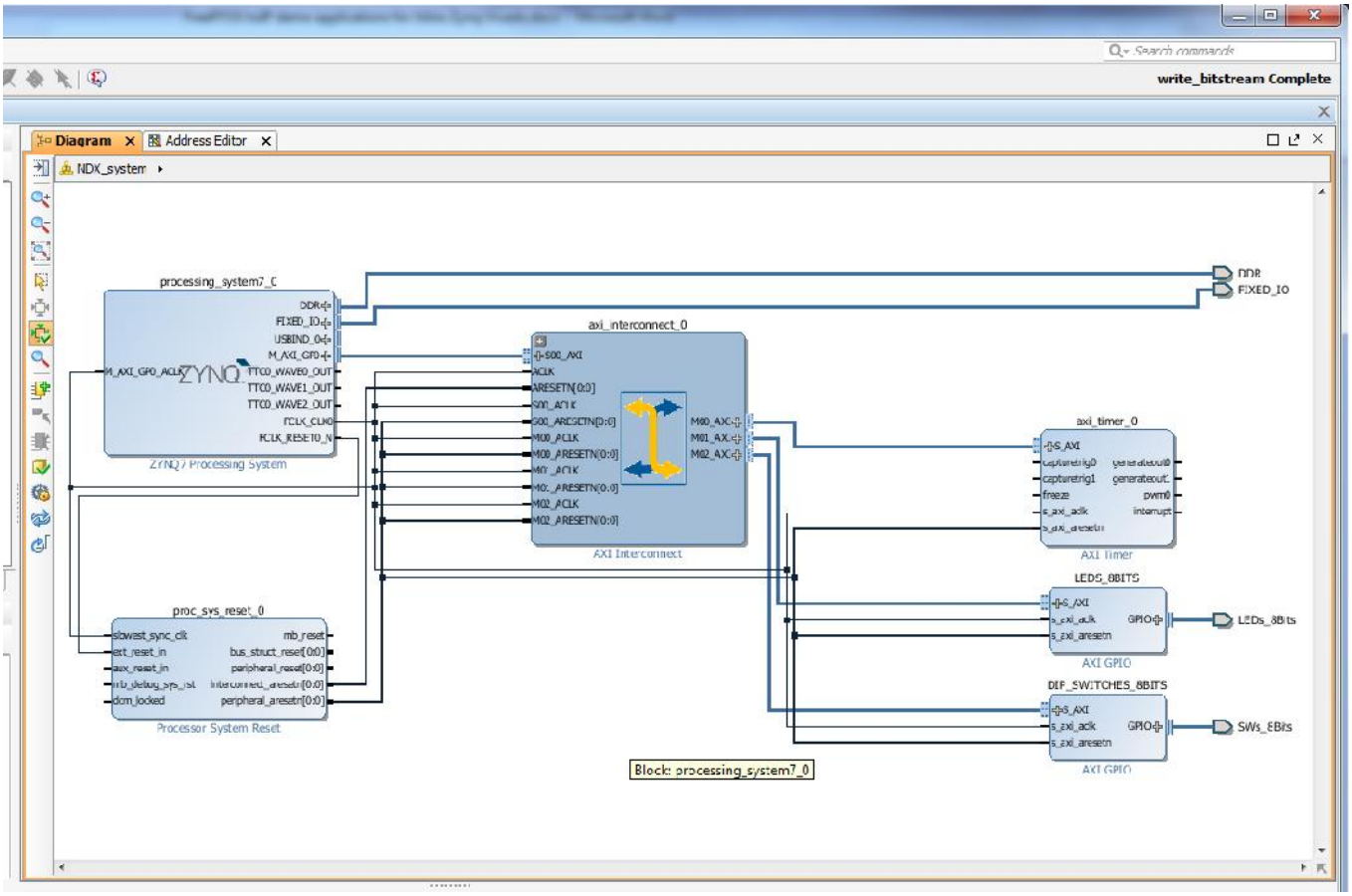


Once this is set up, future launches do not need this dialog. Just run from the run menu directly.

From a browser on the same subnet as the Zedboard, navigate to the DHCP or fixed address as reported by the serial terminal output. The terminal will display information about the application.



The LED display on the Zedboard will not work without an FPGA bitstream. To get the LED and switch status to display you will need to configure a Vivado project with a block design and the minimum GPIO to connect the Zedboard LEDs and switches to generate a hardware platform and bitstream. An example block design with AXI GPIO is shown below.


write_bitstream Complete
Diagram
Address Editor
NDX_system
processing_system7_0
ZYNQ7 Processing System
axi_interconnect_0
AXI Interconnect
axi_timer_0
AXI Timer
LEDS_8BITS
AXI GPIO
LEDs_8Bits
DIP_SWITCHES_8BITS
SWs_8Bits
proc_sys_reset_0
Processor System Reset
DDR
FIXED_IO
Block: processing_system7_0