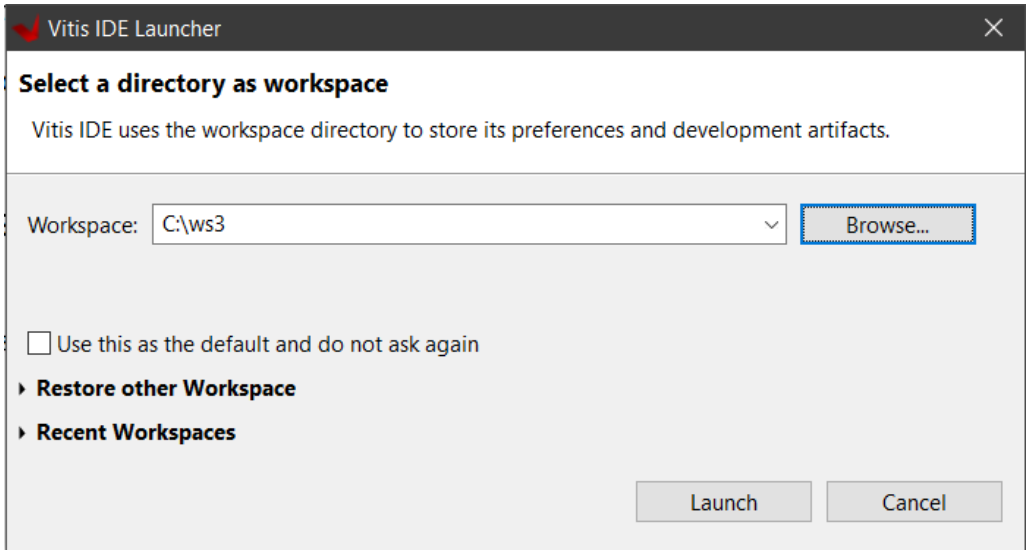
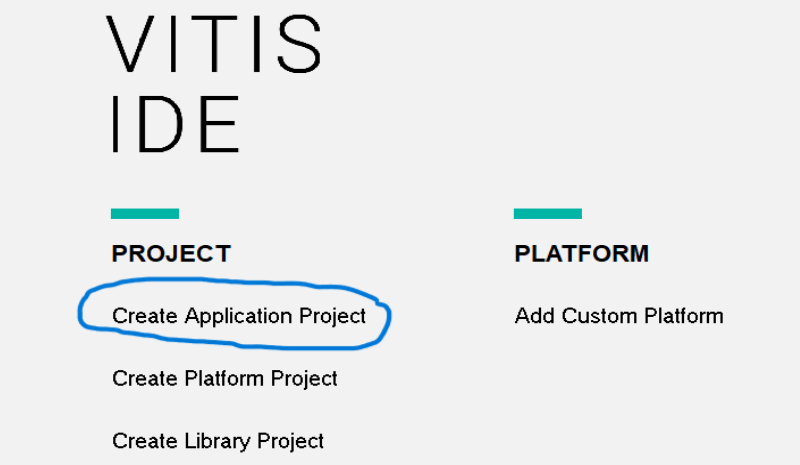
Matrix multiplication accelerator

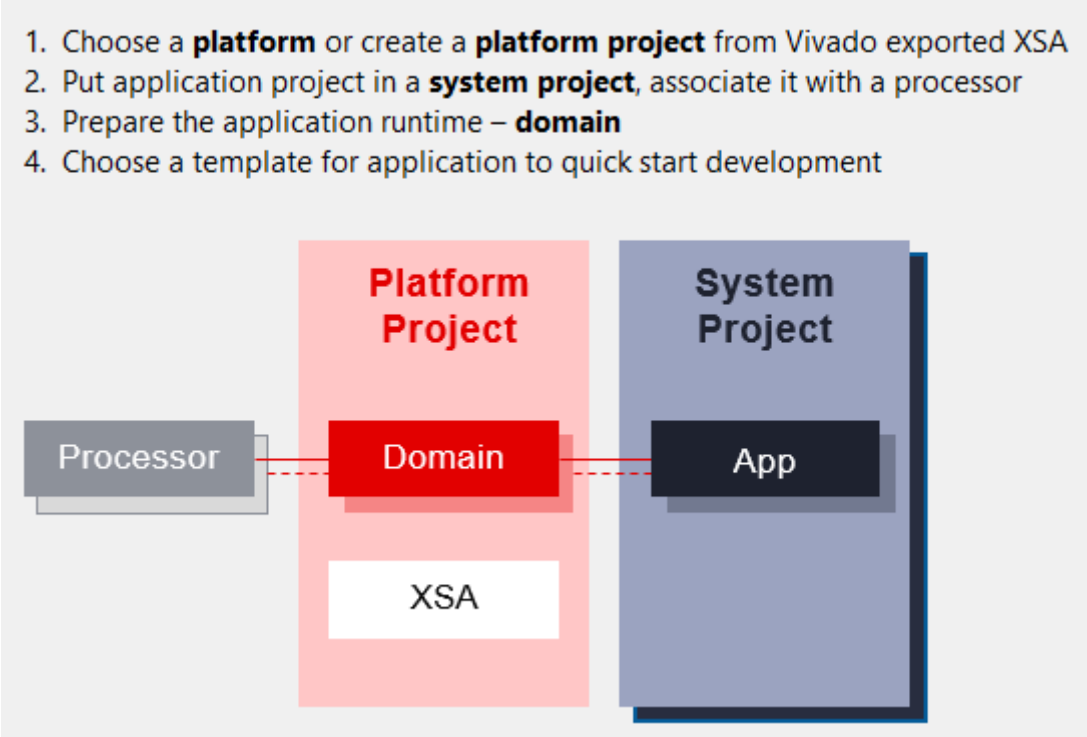
From C++ to hardware

Part 3: Using the accelerator

1. Start Vitis, create a workspace ("C:\ws3") and create a new "Application Project".

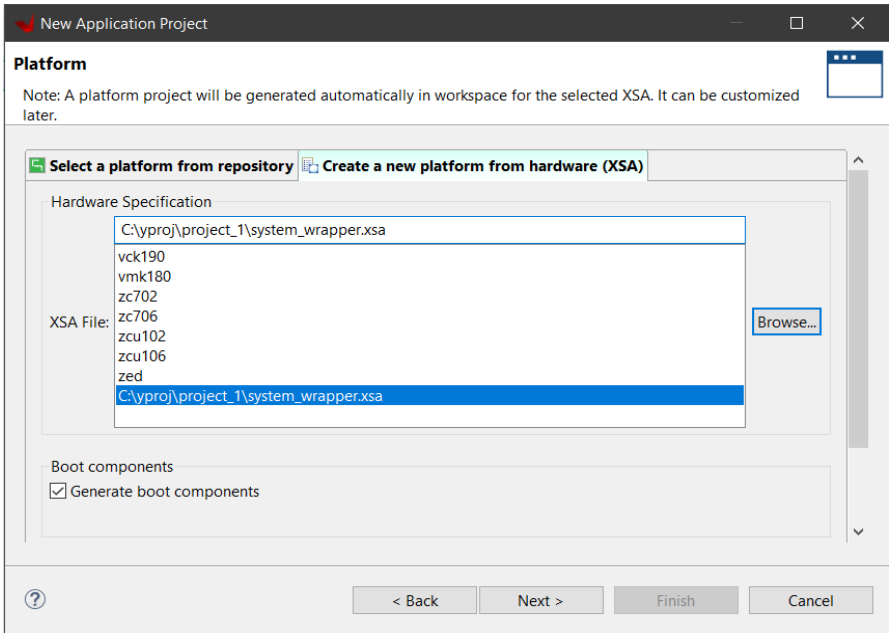




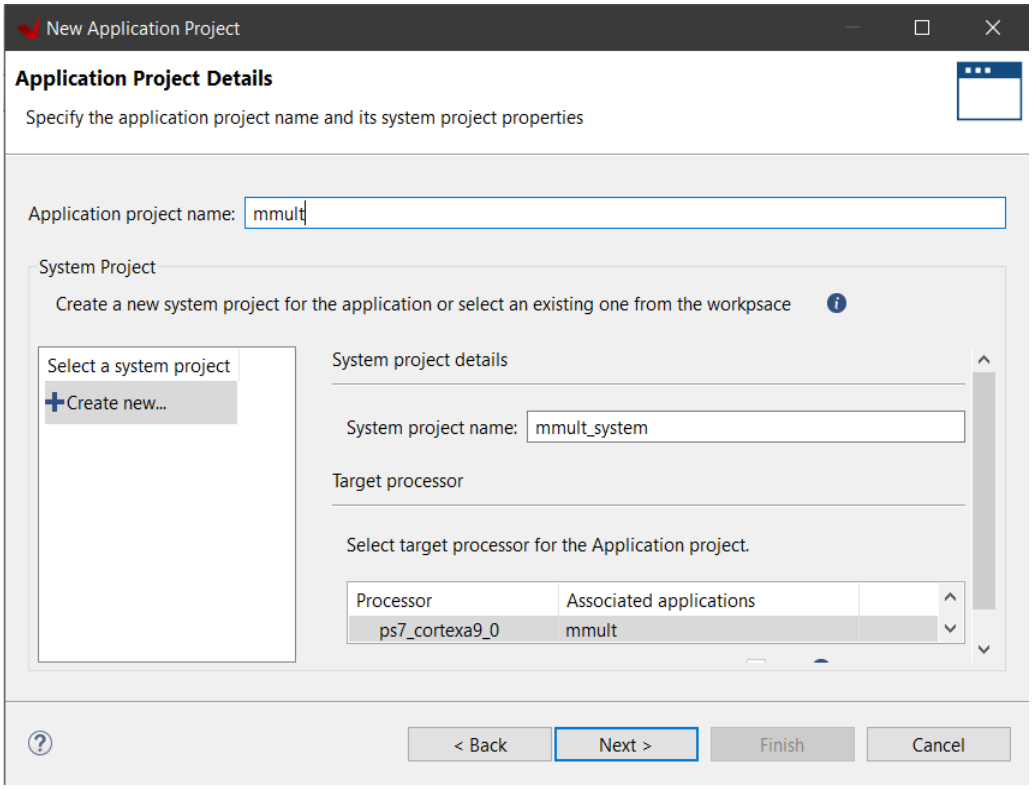


2. We will create a platform project based on the hardware platform previously defined in Vivado.

2.1. Choose “Create a new platform from hardware (XSA)”
2.2. Use the “Browse…” button to select the file “system_wrapper.xsa” (which is inside the “project_1” folder).



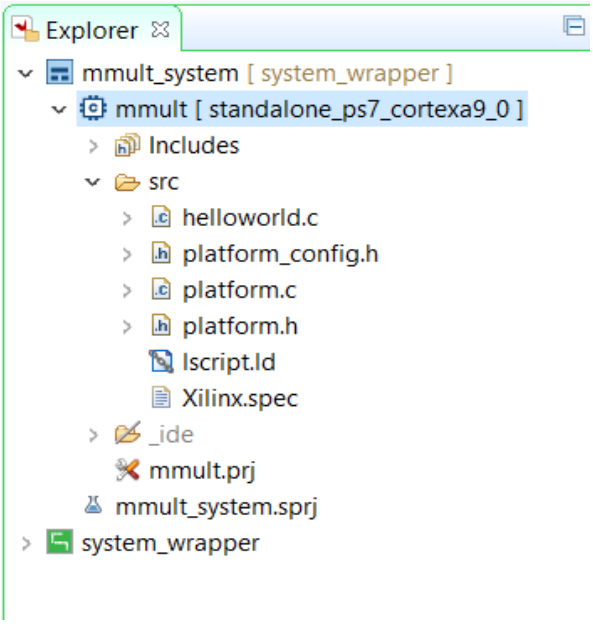
2.3. Specify the application project name: “mmult”



2.4. In the next window (“Domain”), just leave the deafults.
2.5.In the “Templates” window select “Hello World”.

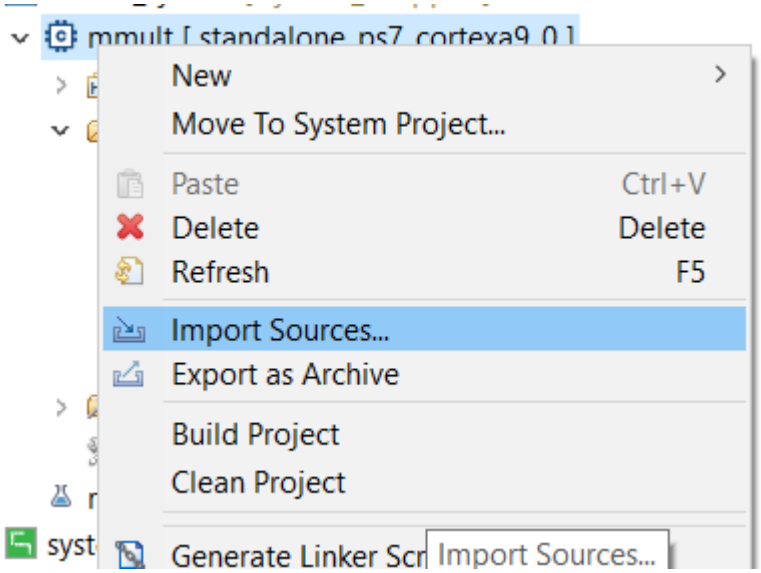
3. Adding the C source code to project

3.1. In the “Explorer” tab, look at the folder “src”. It has some predefined source files.

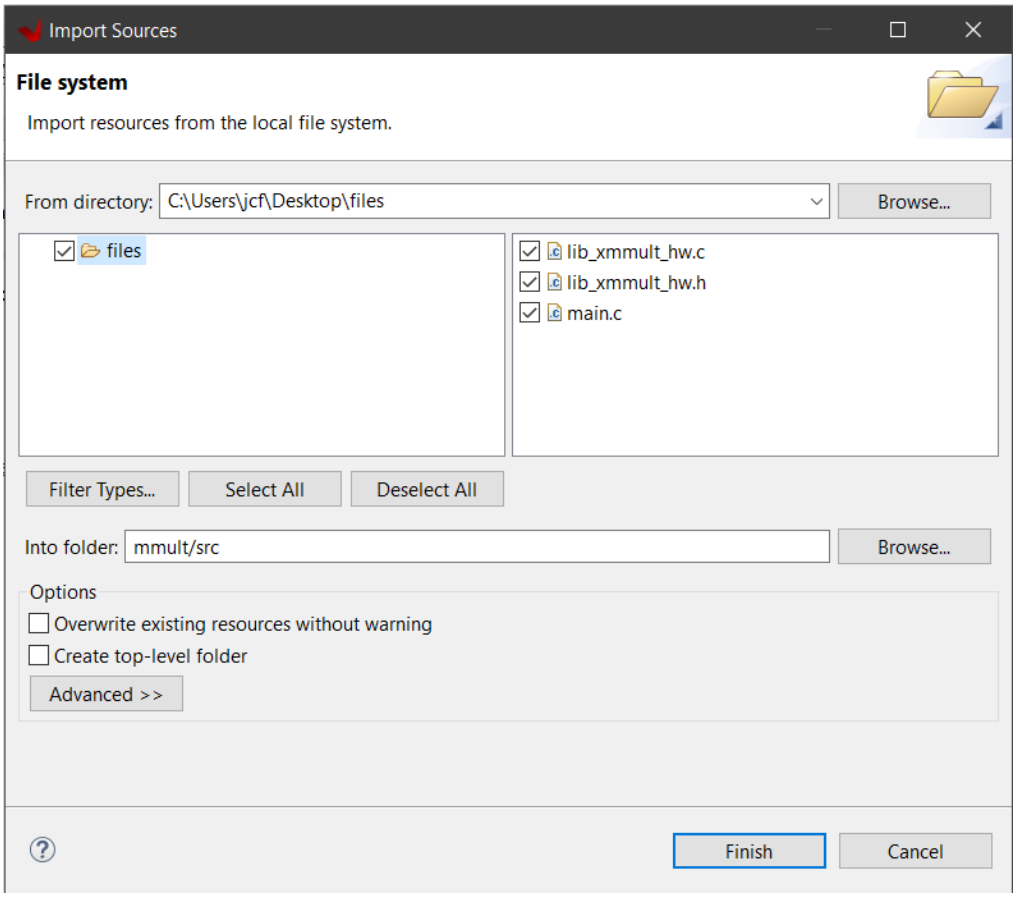


Delete the “helloworld.c” file (use a right-click).

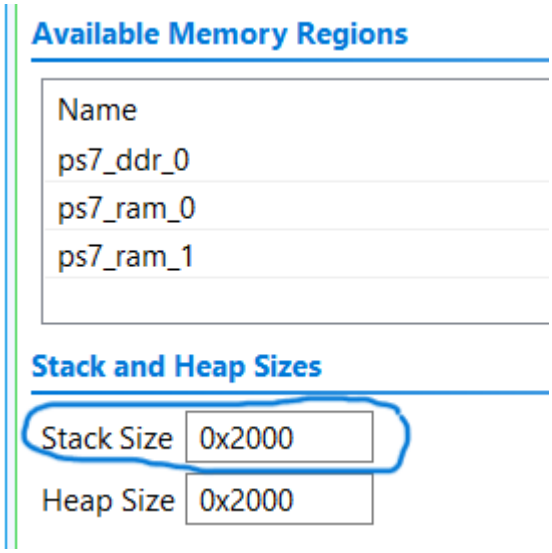
3.2. Select “mmult [standalone_ps_7_cortexa9_0]” icon and use the right mouse button to select “Import Sources”.



3.3. Select the folder where you unzipped the files (“src_files.zip”) and choose all three files. Click “Finish”.



3.4. In the Explorer, click on “main.c” and look at the application code.
3.4.1. What is the purpose of the code on lines 143–164? (Check the comments.)
3.4.2. What is the purpose of the code on lines 181–192?
3.4.3. What is the purpose of the code on lines 195–208?
3.4.4. What is the purpose of the code on lines 211–230?
4. Double click on “lscript.ld” (the linker script).
4.1. Change the stack size from 0x2000 to 0x3000 and save the file.



5. The files generated by Vivado contain errors! You will need to change the files called "Makefile" in the following directories (they have the same contents):

```
/ws3/system_wrapper/ps7_cortexa9_0/standalone_ps7_cortexa9_0/bsp/ps7_cortexa9_0/
libsrc/HLS_accel_v1_0/src
```

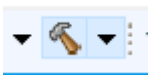
```
/ws3/system_wrapper/zynq_fsbl/zynq_fsbl_bsp/ps7_cortexa9_0/libsrc/HLS_accel_v1_0/src
```

Remove the hash character (#) that is at the start of the lines shown in red below (32, 35, 40).

```
# Rules to compile and link
$(OBJDIR)/%.o: %.c
	echo "Compiling $@ from $<"
	$(COMPILER) $(COMPILER_FLAGS) $(EXTRA_COMPILER_FLAGS) $(INCLUDES) -o $@ $<
$(LIB): $(OBJS)
	echo "Linking $@"
	$(ARCHIVER) $(ARCHIVER_FLAGS) -r $@ $^

# Rule to release include files
$(INCLUDEDIR)/%.h: %.h
	echo "Copying '$<' to '$@'"
	$(CP) $< $@
```

Do this for **both** files.

6. Build everything

Use the icon  to build project "mmult". This will also build the "mmult_system" project and the "system_wrapper" platform. It will take some time…

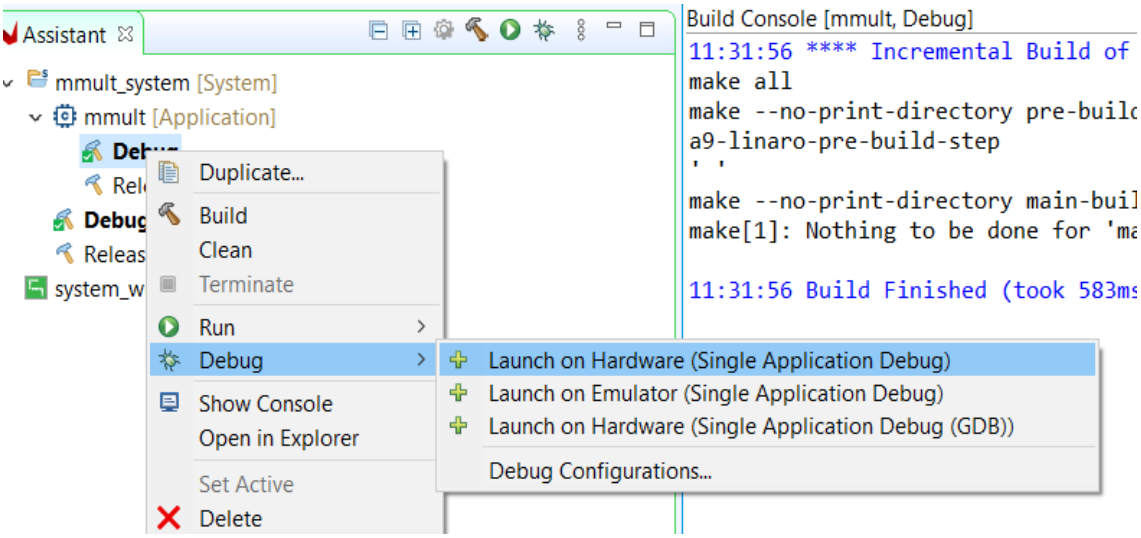
The compiled code wil be located in file “mmult.elf”.

```
'

Building target: mmult.elf'
Invoking: ARM v7 gcc linker'
rm-none-eabi-gcc -mcpu=cortex-a9 -mfpu=vfpv3 -mfloat-
Finished building target: mmult.elf'
 '

Invoking: ARM v7 Print Size'
rm-none-eabi-size mmult.elf  |tee "mmult.elf.size"
   text    data     bss     dec     hex filename
  46180    3048   28696   77924   13064 mmult.elf
Finished building: mmult.elf.size'
 '
```

7. Running the program

7.1. Make sure the Zedboard is connected to your computer and turned on.

7.2. Run the program in Debug mode.

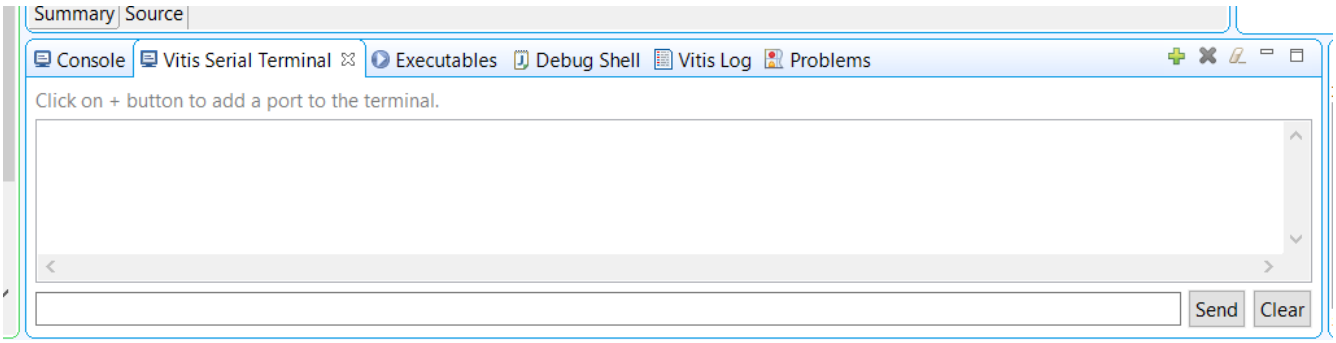


The FPGA will be configured with the hardware (generated by Vivado) and the standalone program created in Vitis.

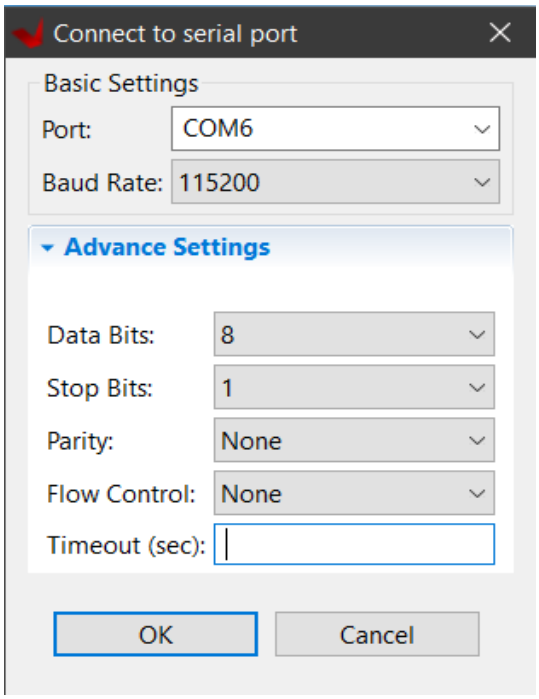
The program will stop at the start of function “main()” due to a breakpoint.

7.3. Configure the serial terminal. (The program’s output will appear here.)

7.3.1. Near the bottom, select “Vitis Serial Terminal” and click on the green plus sign.



Configure the terminal as shown next. Note that the communication port (COM6 in this case) will be different on your computer.



7.3.2. On the top bar, click the “Resume” button 

to proceed with the execution of the application.

7.3.3. Check the messages that appear in the terminal window.

```
Total run time for AXI DMA + HW accelerator is 5236 cycles over 1024 tests

Acceleration factor: [redacted]


SW and HW results match!
```

8. DONE.