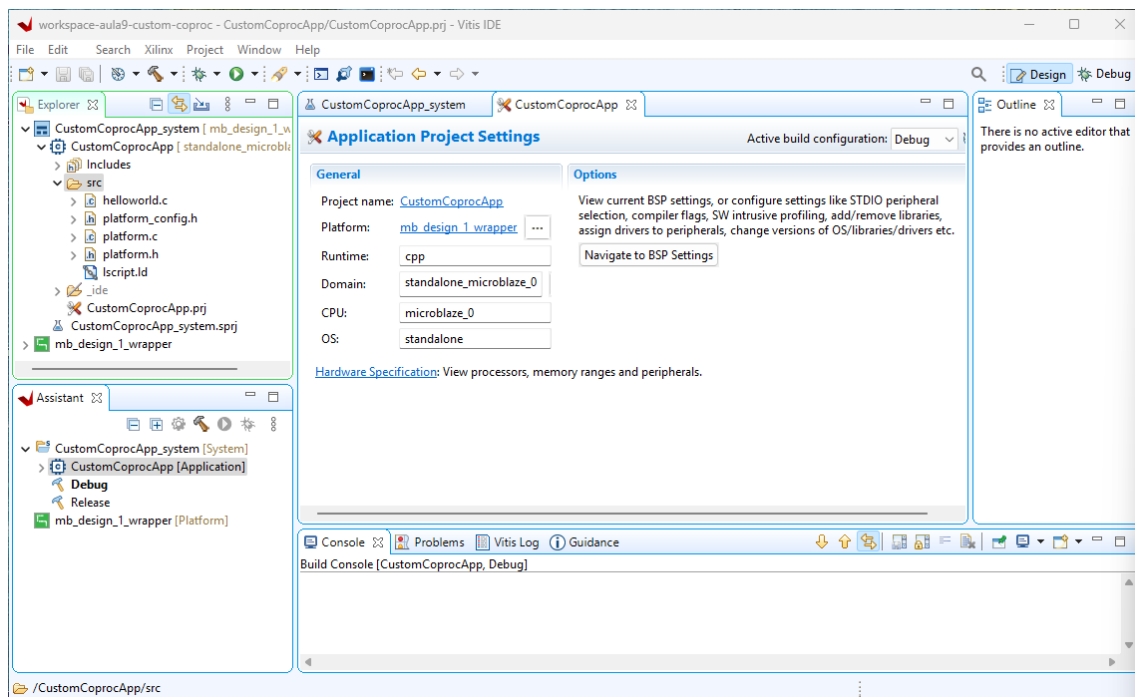


Tutorial 4

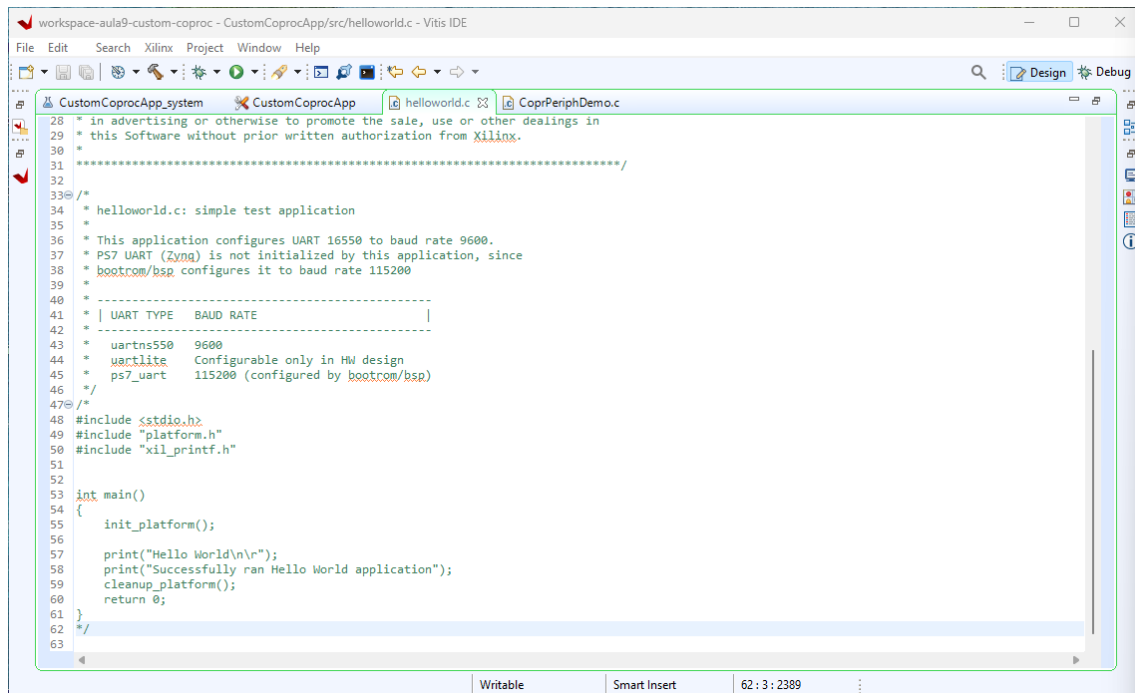
Update MicroBlaze Platform with a Custom Coprocessor (3-register adder) connected via AXI-Lite – SW Project (Vitis)

Vivado version: 2022.2

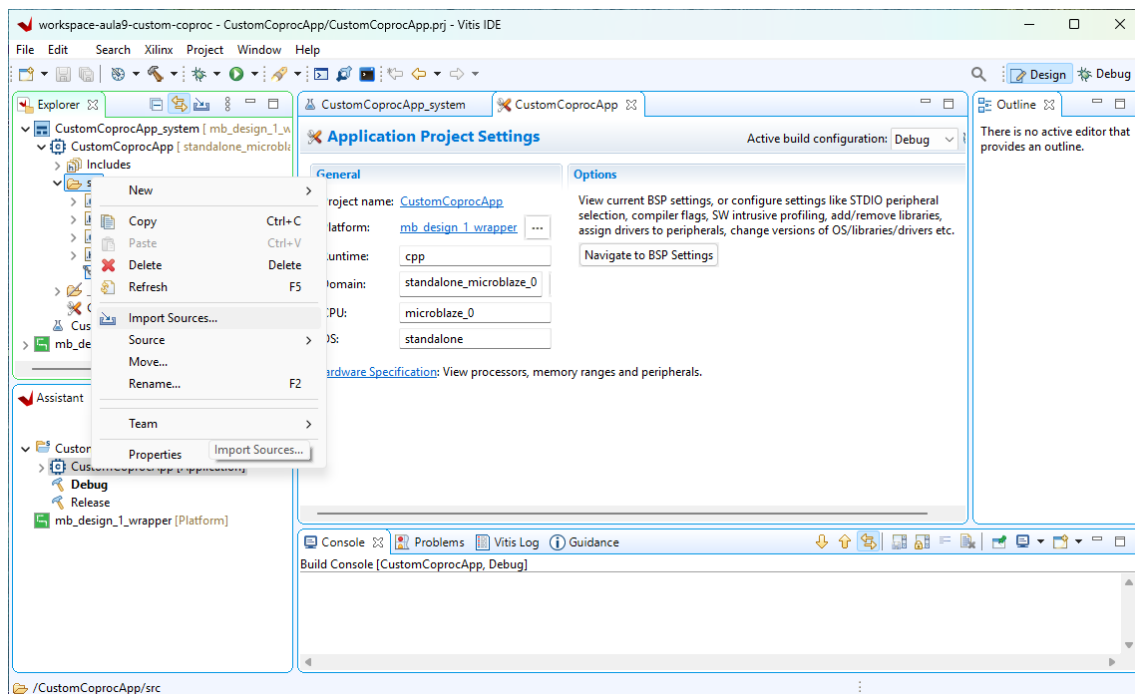
1. Open **Vitis**. Create a new application project. Import the XSA of the project you just created in **Vivado**. The application project template can be an empty C++ project or, if in doubt, the **Hello World** application template. Name the application at your discretion (e.g., **CustomCoproApp**).



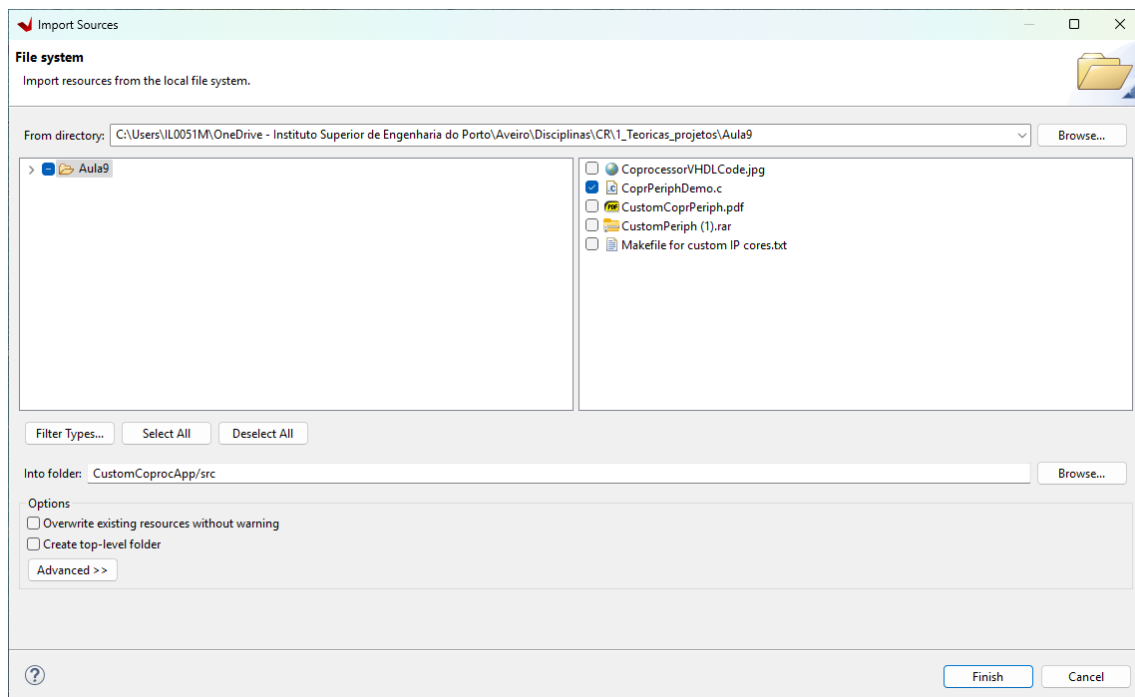
2. In case you selected the **Hello World** application template, open the **helloworld.c** file and comment it fully.



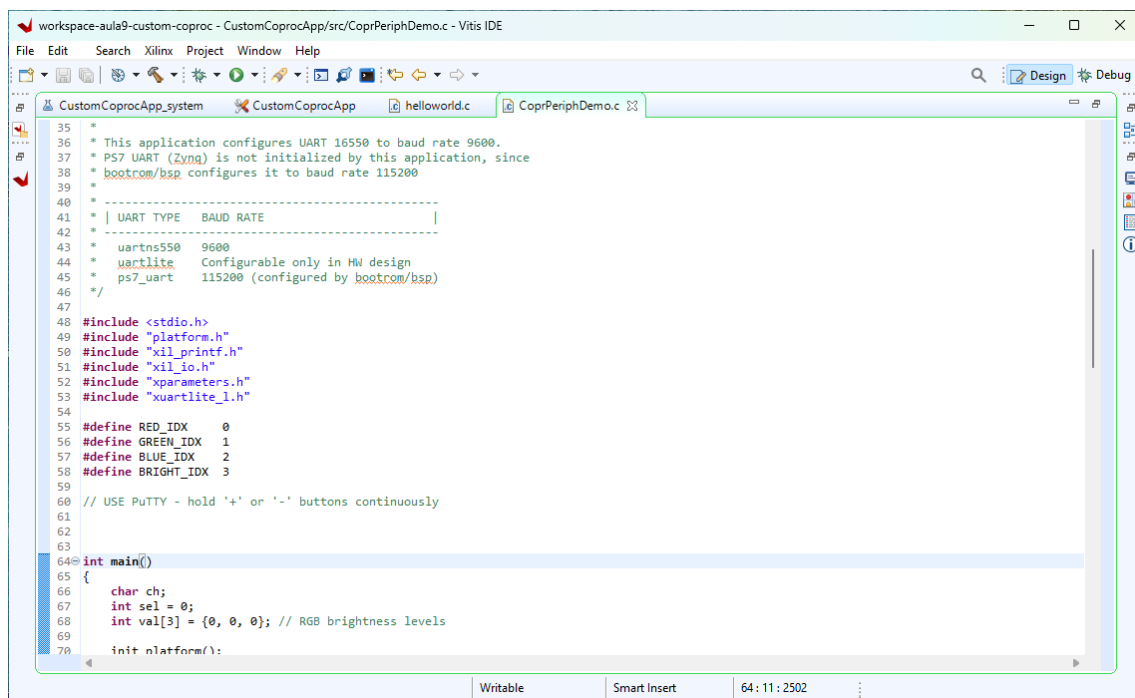
3. Click on **Import Sources**, on the **src** folder.



4. Import file **CoprPeriphDemo.c** – available in the Moodle.



5. Confirm that the file has been imported into the project.



6. There may be errors flagged in the file, concerning macros that represent the co-processor parameters (e.g., base address). This occurs because this example file assumes a name for the co-processor that may not match the one you choose.

```

53 #include "xuartlite.h"
54
55 #define RED_IDX 0
56 #define GREEN_IDX 1
57 #define BLUE_IDX 2
58 #define BRIGHT_IDX 3
59
60 // USE PuTTY - hold '+' or '-' buttons continuously
61
62
63
64 int main()
65 {
66     char ch;
67     int sel = 0;
68     int val[3] = {0, 0, 0}; // RGB brightness levels
69
70     init_platform();
71
72     print("Hello Custom Coprocessor\n\n");
73
74     for (int i = 0; i < 10; i++)
75     {
76         Xil_Out32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 0, i);
77         Xil_Out32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 4, i + 1);
78         Xil_Out32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 8, i + 2);
79         xil_printf("%d + %d + %d = %d\n", Xil_In32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 0),
80             Xil_In32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 4),
81             Xil_In32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 8),
82             Xil_In32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 12));
83
84         if (Xil_In32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 12) == (3 * i + 3))
85             xil_printf("OK\n");
86         else
87             xil_printf("ERROR\n");
88     }
89 }

```

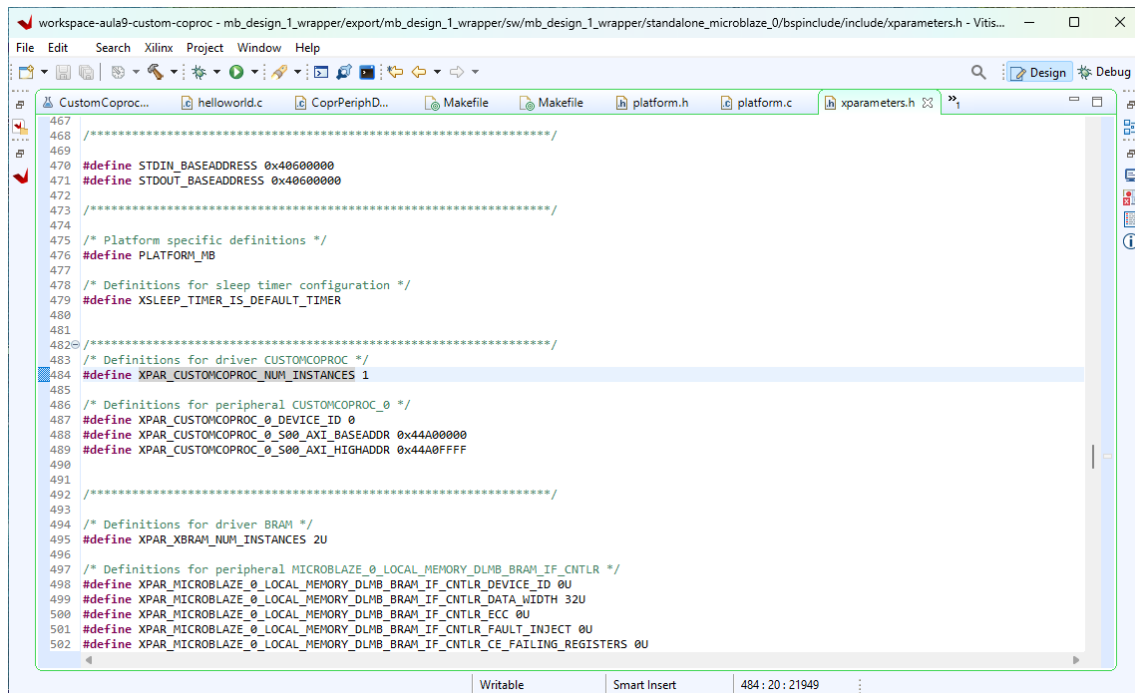
7. Right-click on `#include "xparameters.h"`, and select **Open Declaration**.

```

44 * uartlite
45 * ps7_uart
46 */
47
48 #include <stdio.h>
49 #include "platform.h"
50 #include "xil_printf.h"
51 #include "xil_io.h"
52 #include "xparameters.h"
53 #include "xuartlite.h"
54
55 #define RED_IDX 0
56 #define GREEN_IDX 1
57 #define BLUE_IDX 2
58 #define BRIGHT_IDX 3
59
60 // USE PuTTY - hold '+' or '-' buttons continuously
61
62
63
64 int main()
65 {
66     char ch;
67     int sel = 0;
68     int val[3] = {0, 0, 0};
69
70     init_platform();
71
72     print("Hello Custom Coprocessor\n\n");
73
74     for (int i = 0; i < 10; i++)
75     {
76         Xil_Out32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 0, i);
77         Xil_Out32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 4, i + 1);
78         Xil_Out32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 8, i + 2);
79         xil_printf("%d + %d + %d = %d\n", Xil_In32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 0),
80             Xil_In32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 4),
81             Xil_In32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 8),
82             Xil_In32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 12));
83
84         if (Xil_In32(XPAR_CUSTOMCOPR_0_S00_AXI_BASEADDR + 12) == (3 * i + 3))
85             xil_printf("OK\n");
86         else
87             xil_printf("ERROR\n");
88     }
89 }

```

8. Navigate the file until you find the right name of your macro. Replace the names correspondingly in the **CoprPeriphDemo.c** file.

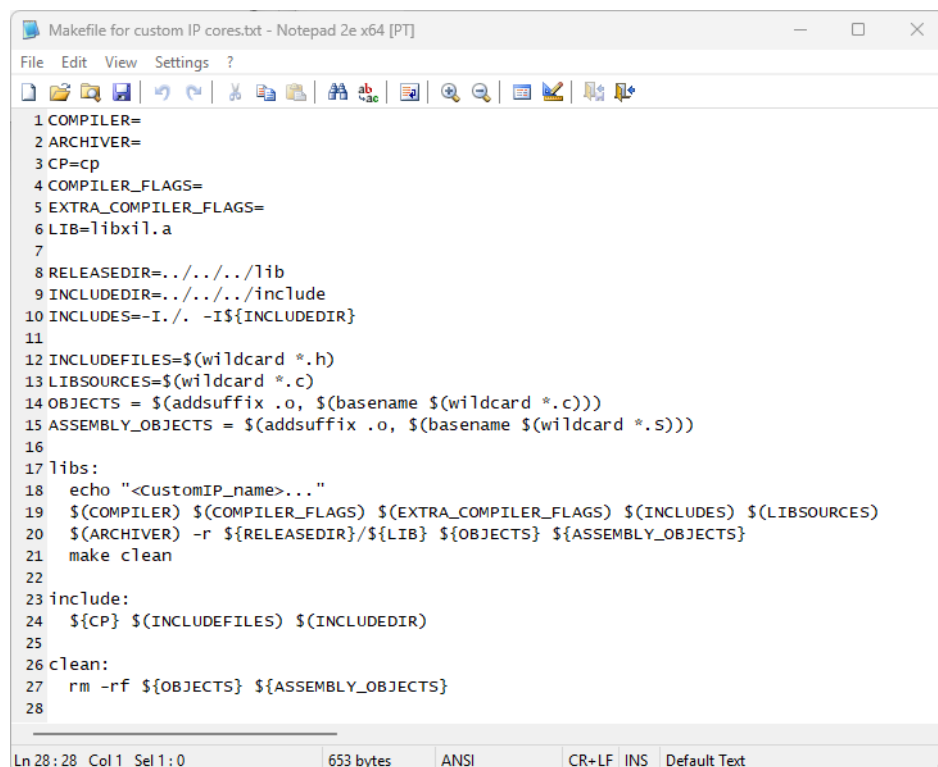


```

467
468
469
470 #define STDIN_BASEADDRESS 0x40600000
471 #define STDOUT_BASEADDRESS 0x40600000
472
473
474
475 /* Platform specific definitions */
476 #define PLATFORM_MB
477
478 /* Definitions for sleep timer configuration */
479 #define XSLEEP_TIMER_IS_DEFAULT_TIMER
480
481
482
483
484 #define XPAR_CUSTOMCOPROC_NUM_INSTANCES 1
485
486 /* Definitions for peripheral CUSTOMCOPROC_0 */
487 #define XPAR_CUSTOMCOPROC_0_DEVICE_ID 0
488 #define XPAR_CUSTOMCOPROC_0_S00_AXI_BASEADDR 0x44A00000
489 #define XPAR_CUSTOMCOPROC_0_S00_AXI_HIGHADDR 0x44A0FFFF
490
491
492
493
494 /* Definitions for driver BRAM */
495 #define XPAR_XBRAM_NUM_INSTANCES 2U
496
497 /* Definitions for peripheral MICROBLAZE_0 LOCAL_MEMORY_DLMB_BRAM_IF_CNTRL */
498 #define XPAR_MICROBLAZE_0_LOCAL_MEMORY_DLMB_BRAM_IF_CNTRL_DEVICE_ID 0U
499 #define XPAR_MICROBLAZE_0_LOCAL_MEMORY_DLMB_BRAM_IF_CNTRL_DATA_WIDTH 32U
500 #define XPAR_MICROBLAZE_0_LOCAL_MEMORY_DLMB_BRAM_IF_CNTRL_ECC 0U
501 #define XPAR_MICROBLAZE_0_LOCAL_MEMORY_DLMB_BRAM_IF_CNTRL_FAULT_INJECT 0U
502 #define XPAR_MICROBLAZE_0_LOCAL_MEMORY_DLMB_BRAM_IF_CNTRL_CE_FAILING_REGISTERS 0U

```

9. At this point, due to a bug in Vitis, we need to update two **Makefiles** used for compilation of the Custom Coprocessor. In the Moodle, you should find the file **Makefile for custom IP cores.txt**. Open it. You should use it for all projects involving co-processors.

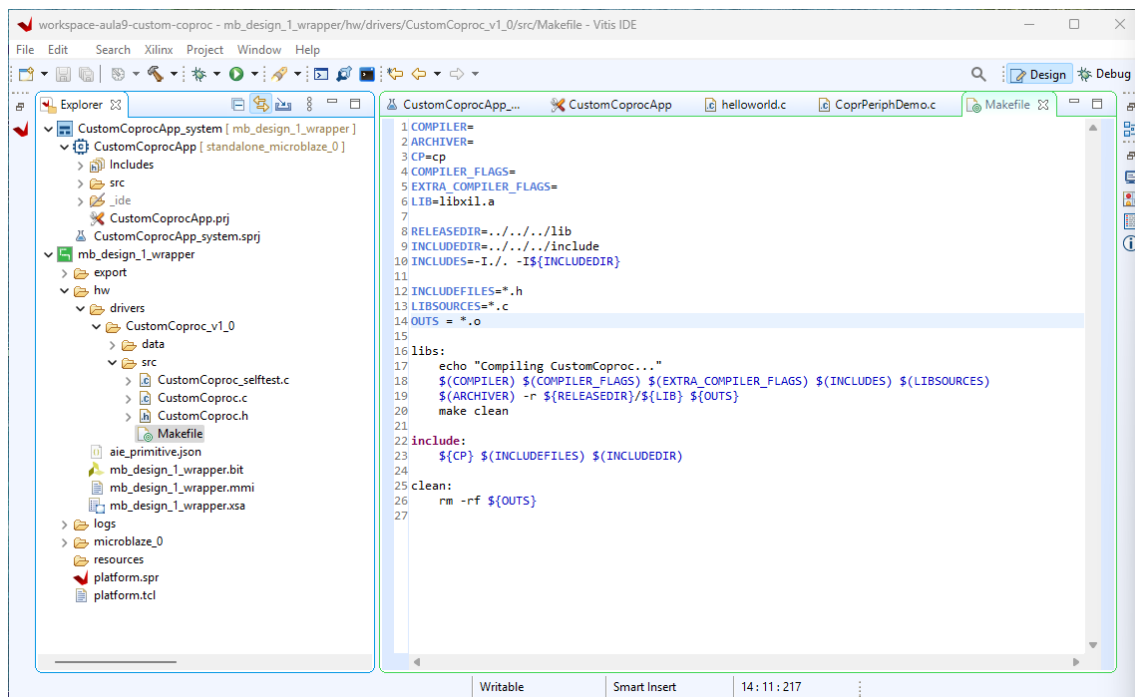


```

1 COMPILER=
2 ARCHIVER=
3 CP=cp
4 COMPILER_FLAGS=
5 EXTRA_COMPILER_FLAGS=
6 LIB=libxil.a
7
8 RELEASEDIR=../../lib
9 INCLUDEDIR=../../include
10 INCLUDES=-I./ -I${INCLUDEDIR}
11
12 INCLUDEFILES=$(wildcard *.h)
13 LIBSOURCES=$(wildcard *.c)
14 OBJECTS = $(addsuffix .o, $(basename $(wildcard *.c)))
15 ASSEMBLY_OBJECTS = $(addsuffix .o, $(basename $(wildcard *.s)))
16
17 libs:
18 echo "<CustomIP_name>..."
19 $(COMPILER) $(COMPILER_FLAGS) $(EXTRA_COMPILER_FLAGS) $(INCLUDES) $(LIBSOURCES)
20 $(ARCHIVER) -r ${RELEASEDIR}/${LIB} ${OBJECTS} ${ASSEMBLY_OBJECTS}
21 make clean
22
23 include:
24 ${CP} $(INCLUDEFILES) $(INCLUDEDIR)
25
26 clean:
27 rm -rf ${OBJECTS} ${ASSEMBLY_OBJECTS}
28

```

10. The first **Makefile** can be found in the following location.



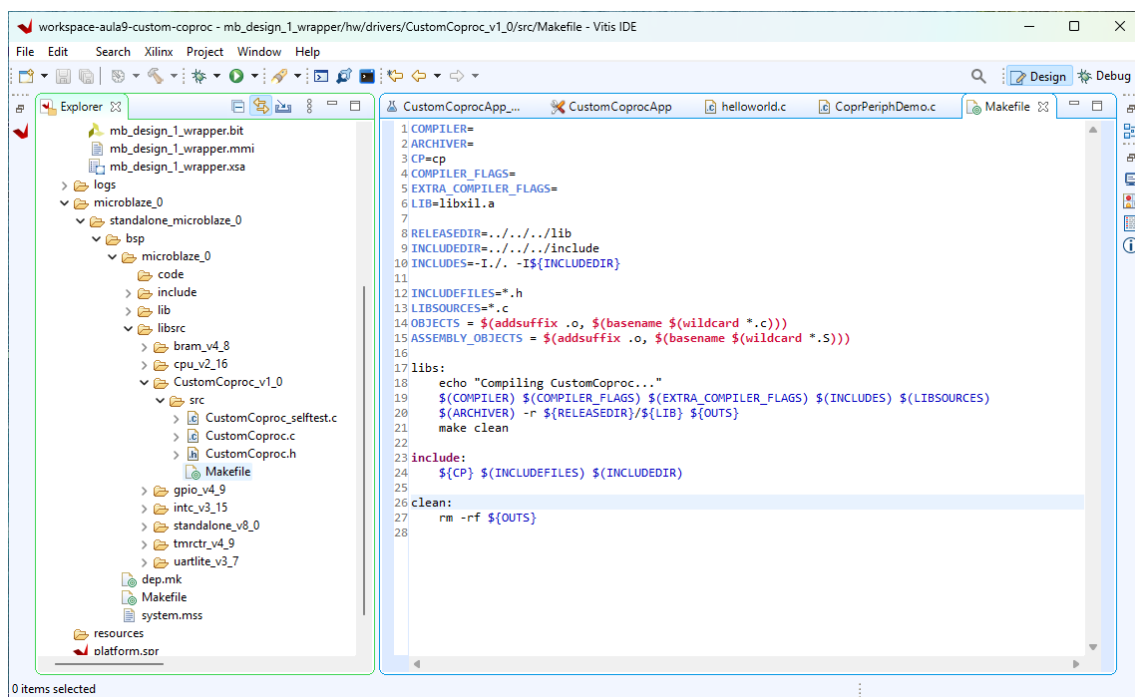
11. Replace line 14 with the following lines. Also, select and copy the whole file (Ctrl-A + Ctrl-C).

```

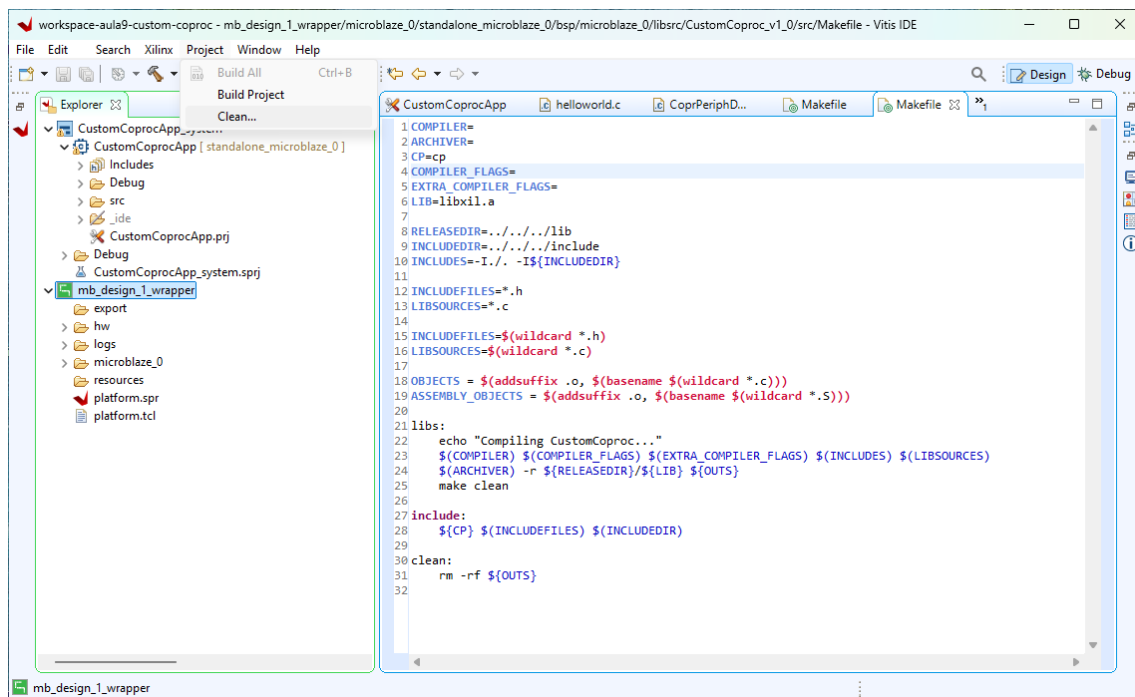
OBJECTS = $(addsuffix .o, $(basename $(wildcard *.c)))
ASSEMBLY_OBJECTS = $(addsuffix .o, $(basename $(wildcard *.S)))

```

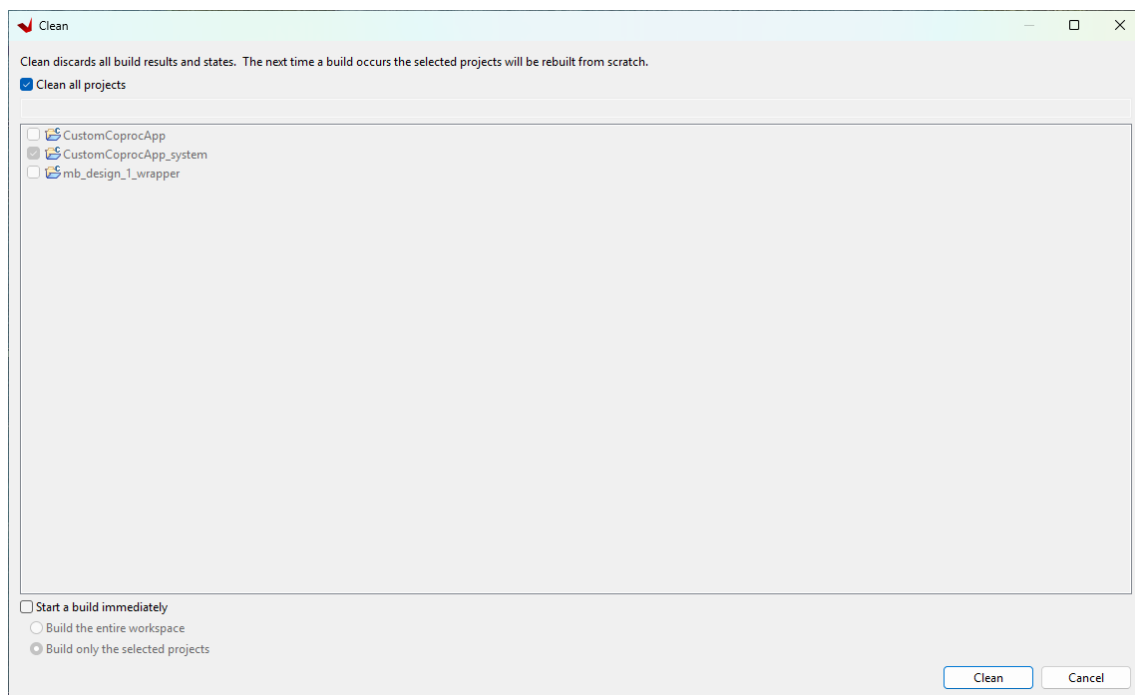
12. The second **Makefile** can be found in the following location. Replace its contents with the previous **Makefile** (Ctrl-A + Ctrl-V).



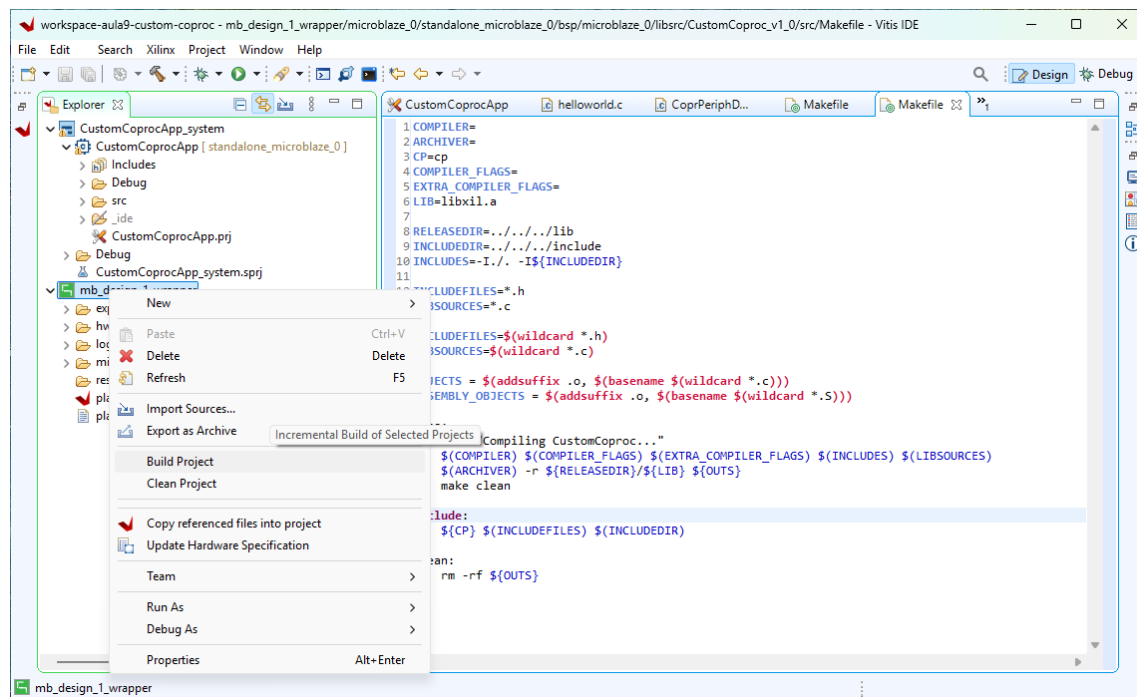
13. Click on **Project** > **Clean**.



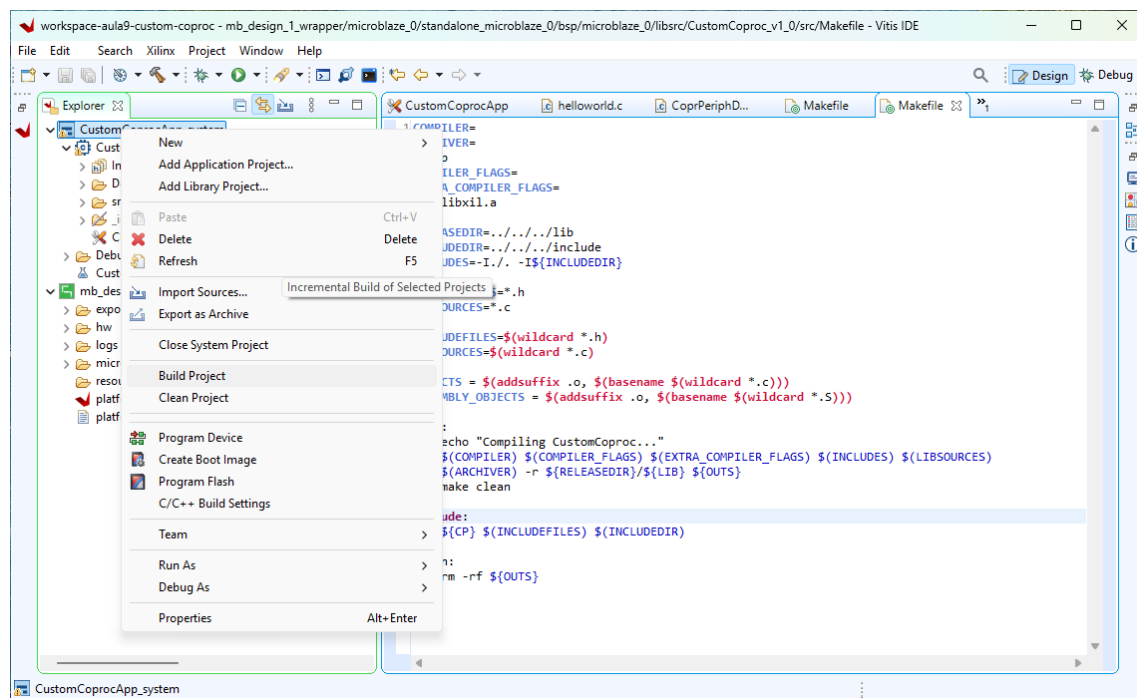
14. Clean all projects.



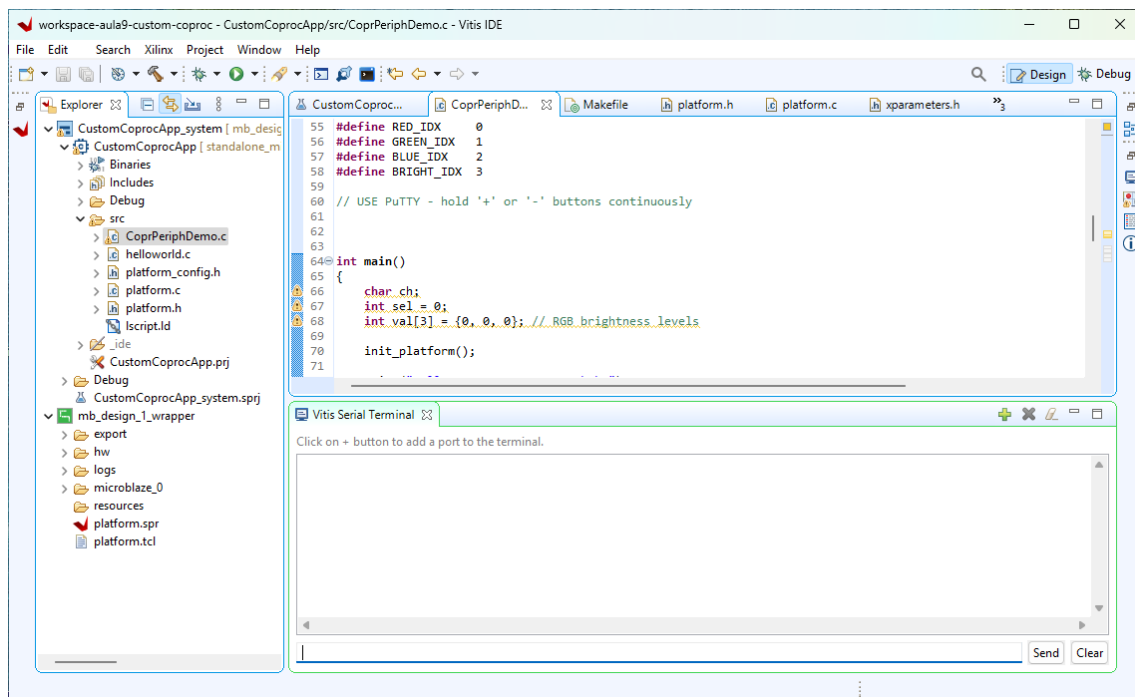
15. Build the platform



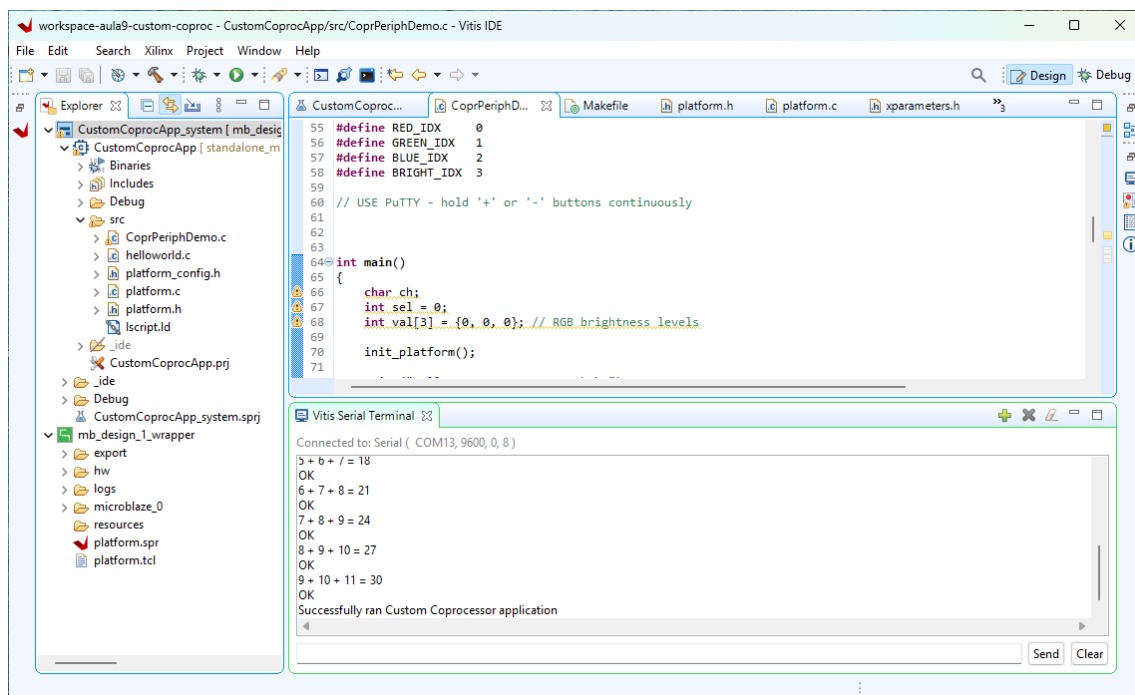
16. Build the application.



17. Open the Vitis Serial Terminal. Select the right configuration parameters.



18. You should observe the program output.



Congratulations!

Sources: AMD Xilinx documentation.