Tecnológico de Monterrey

Embedded Systems
(TE 3059)

Second Partial Exam: SoC and FPGA

Perla Vanessa Jaime Gaytán
A00344428
ITE

Professor: Luis Fernando Gonzalez Perez

November 4, 2020

Contents

List of Figures

Nomenclature

AXI	Advanced eXtensible Interface
FPGA	Field Programmable Gate Arrays
GHRD	Golden Hardware Reference Design
HPS	Hard Processor System
HW	Hardware
SoC	System on Chip
SW	Software

1 Objective

The objective of this Lab was to learned how to use the HPS/ARM to communicate with FPGA. It introduce the GHRD project for DE10-Standard development board in order to develop one ARM C Project which demonstrates how HPS/ARM program controls the ten LEDs connected to FPGA. It shows how HPS controls the FPGA LED through Lightweight HPS-to-FPGA Bridge. The FPGA is configured by HPS through FPGA manager in HPS.

2 Tools

2.1 Software tools

- Intel Quartus Prime Lite Edition v.16.1
- Intel Qsys
- Intel SoC FPGA Embedded Development Suite v.19.1
- PuTTy Terminal
- Git

.

2.2 Hardware tools

- Intel DE10-Standard board
- Micro SD-Card
- RJ45 Ethernet cable
- Type A to Mini-B USB Cable
- Type A to B USB Cable
- Power DC Adapter (12V)

3 HPS + FPGA

3.1 AXI bridges in Intel SoC FPGA

In the board, the HPS logic is connected to FPGA through the AXI bridge. In order to communicate HPS logic with FPGA fabric, Intel system integration tool Qsys should be used for the system design to add HPS components. From the AXI master port of the HPS component, HPS can access those Qsys component whose memory-mapped slave ports are connected to the master port.
The HPS contains the following HPS-FPGA AXI bridges:

- FPGA-to-HPS Bridge
- HPS-to-FPGA Bridge
- Lightweight HPS-to-FPGA Bridge

Figure 1 shows a block diagram of the AXI bridges in the context of the FPGA fabric and the L3 interconnect to the HPS. Each master (M) and slave (S) interface is shown with its data width(s). The clock domain for each interconnect is noted in parentheses.
The HPS-to-FPGA bridge is mastered by the level 3 (L3) main switch and

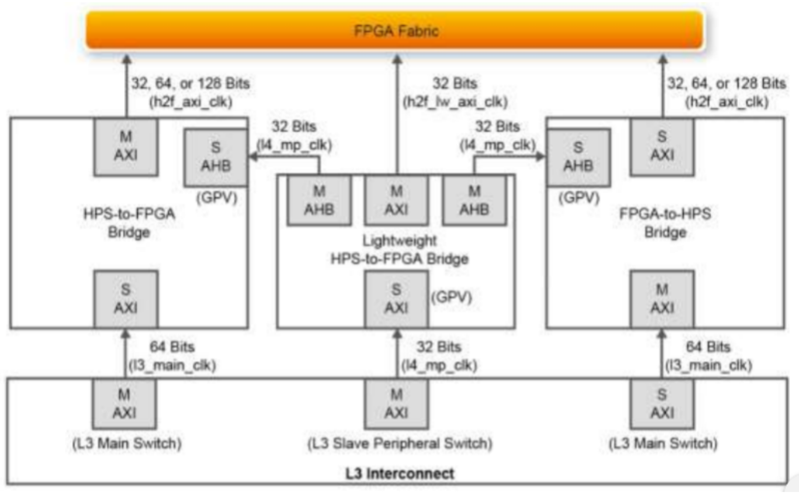


Figure 1: AXI Bridge Block Diagram.

the lightweight HPS-to-FPGA bridge is mastered by the L3 slave peripheral switch. The FPGA-to-HPS bridge masters the L3 main switch, allowing any master implemented in the FPGA fabric to access most slaves in the HPS. For example, the FPGA-to-HPS bridge can access the accelerator coherency. All three bridges contain global programmer view GPV register. The GPV register control the behavior of the bridge. It is able to access to the GPV registers of all three bridges through the lightweight HPS-to-FPGA bridge. This Demo introduces to users how to use the HPS/ARM to communicate with FPGA. This project includes GHRD project for the DE10-Standard one ARM C Project which demonstrates how HPS/ARM program controls the red LEDs connected to FPGA.

In Figure 2 we can see with more detail the SoC to FPGA bridges which are the blue arrows.

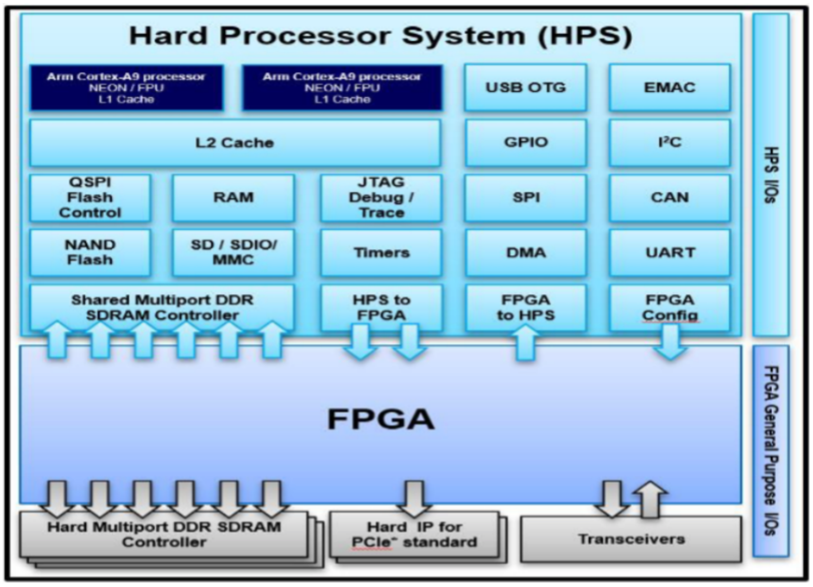

Hard Processor System (HPS)
Arm Cortex-A9 processor NEON / FPU L1 Cache
Arm Cortex-A9 processor NEON / FPU L1 Cache
USB OTG
EMAC
L2 Cache
GPIO
I²C
QSPI Flash Control
RAM
JTAG Debug / Trace
SPI
CAN
NAND Flash
SD / SDIO/ MMC
Timers
DMA
UART
Shared Multiport DDR SDRAM Controller
HPS to FPGA
FPGA to HPS
FPGA Config
HPS I/Os
FPGA
FPGA General Purpose I/Os
Hard Multiport DDR SDRAM Controller
Hard IP for PCIe standard
Transceivers

Figure 2: SoC-FPGA bridges.

4 Design Flow

In Figure 3 we can appreciated with more detail how HW and SW communicate in the board.

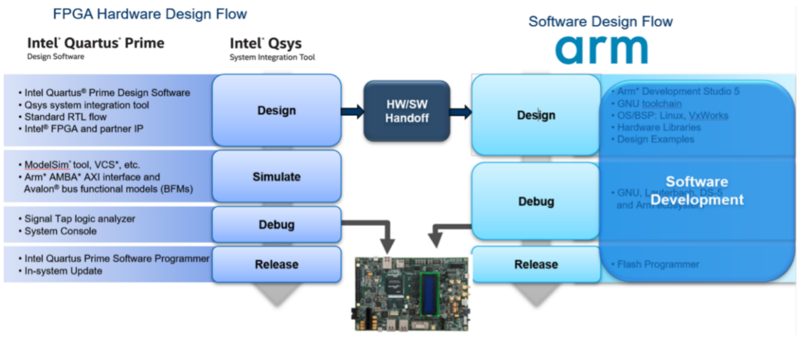

FPGA Hardware Design Flow
Intel Quartus Prime
Design Software
Intel Qsys
System Integration Tool
Software Design Flow
arm
• Intel Quartus® Prime Design Software
• Qsys system integration tool
• Standard RTL flow
• Intel® FPGA and partner IP
Design
• ModelSim tool, VCS*, etc.
• Arm* AMBA* AXI interface and Avalon® bus functional models (BFMs)
Simulate
• Signal Tap logic analyzer
• System Console
Debug
• Intel Quartus Prime Software Programmer
• In-system Update
Release
HW/SW Handoff
Design
Debug
Release
• Arm* Development Studio 5
• GNU toolchain
• OS/BSP: Linux, VxWorks
• Hardware Libraries
• Design Examples
Software Development
• GNU, Lauterbach, DS-5 and Arm ecosystem
• Flash Programmer

Figure 3: System Development Flow DE10-Standard

4.1 GHRD

The Golden Hardware Reference Design is an Intel® Quartus® Prime project that contains a full HPS design for the Cyclone® V SoC / Arria® V SoC Development Kit. The GHRD has connections to a boot source, SDRAM memory and other peripherals on the development board.
For every new released version of SoC EDS, the GHRD is included in the SoC EDS tools. The GHRD is regression tested with every major release of the Intel® Quartus® Prime Design Software and includes the latest bug fixes for known hardware issues. The GHRD serves as a known good configuration of an SoC FPGA hardware system.
The GHRD has a minimal set of peripherals in the FPGA fabric, because the HPS provides a substantial selection of peripherals. HPS-to-FPGA and FPGA-to-HPS interfaces are configured to a 64-bit data width.

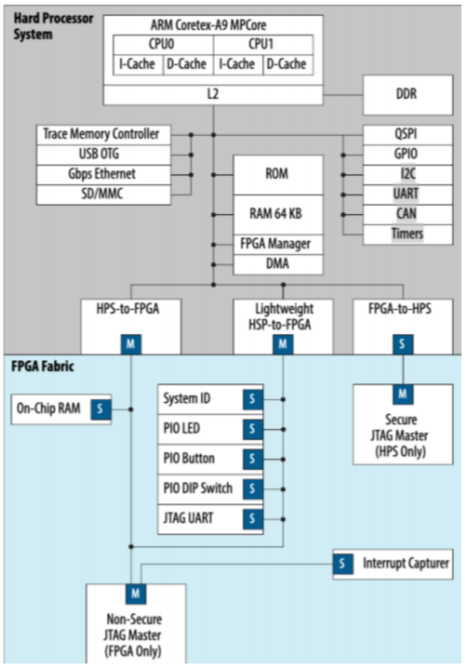


Figure 4: Cyclone® V / Arria® V SoC Golden Hardware Reference Design Overview

5 Lab Procedure

5.1 Open the project

After opening Quartus, I selected Open project on the file menu and select the DE10-Standard-GHRD as shown in Figure 5.

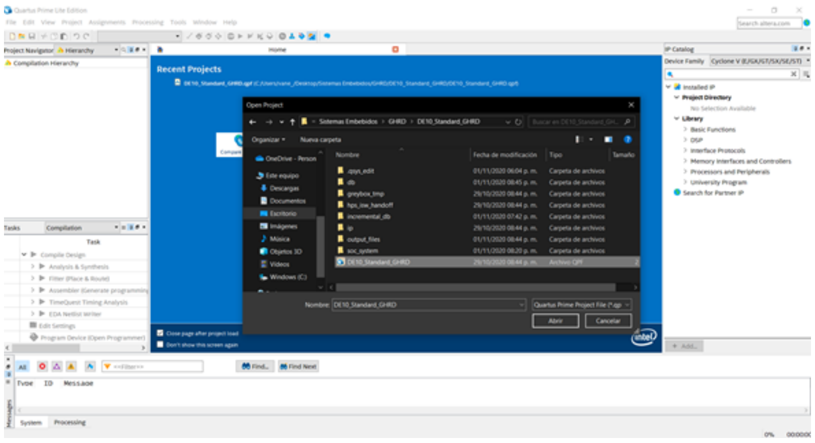


Figure 5: Selecting the project.

5.2 Generate Source Code

The next part is to open Qsys that is located in the Tools menu as shown in Figure 6. Then select soc-system.qsys and open it. After that, at the bottom, we click on the button "Generate HDL..." in order to created the source code. A window will pop up and then the modifications shown in Figure 7 were made and click on generate. When the HDL design is finished a window as shown in Figure 8 is shown an then we can close Qsys.

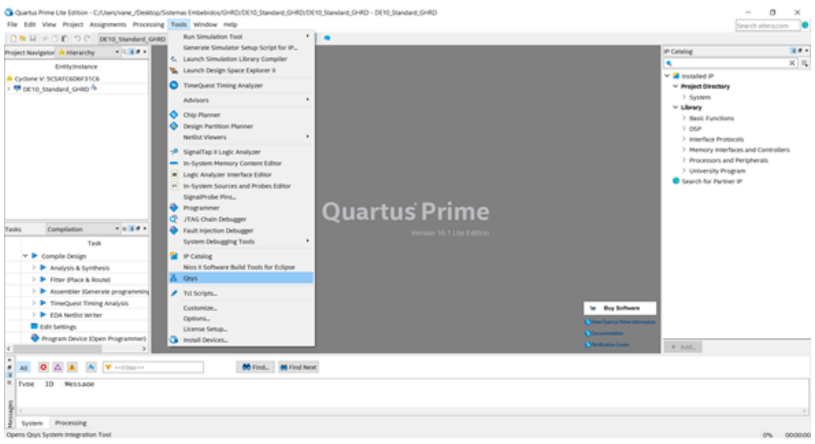
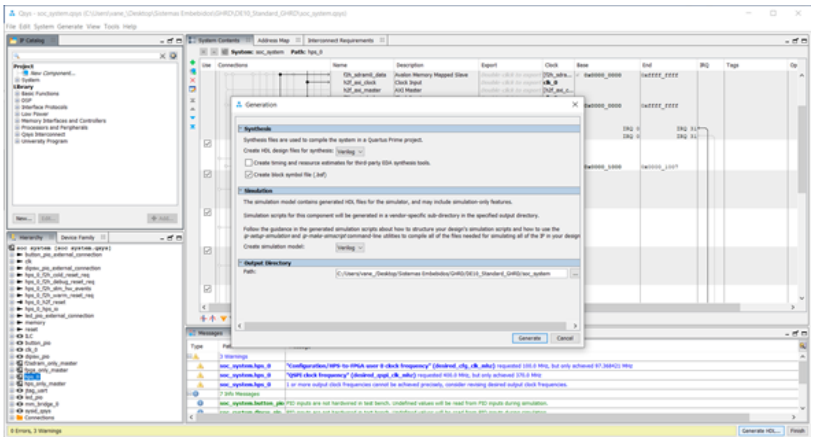


Figure 6: Tools Menu



Generation

Synthesis

Synthesis files are used to compile the system in a Quartus Prime project.

Create HDL design files for synthesis: Verilog

☐ Create timing and resource estimates for third-party EDA synthesis tools.

☑ Create block symbol file (.bsf)

Simulation

The simulation model contains generated HDL files for the simulator, and may include simulation-only features.

Simulation scripts for this component will be generated in a vendor-specific sub-directory in the specified output directory.

Follow the guidance in the generated simulation scripts about how to structure your design's simulation scripts and how to use the ip-setup-simulation and ip-make-simscript command-line utilities to compile all of the files needed for simulating all of the IP in your design

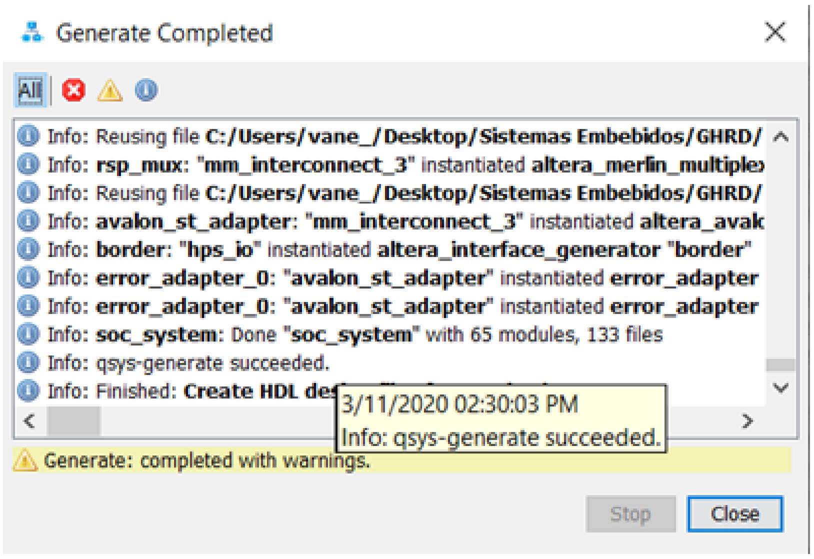
Create simulation model: Verilog

Output Directory

Path: C:/Users/vane_/Desktop/Sistemas Embebidos/GHRD/DE10_Standard_GHRD/soc_system

```
3 Warnings
soc_system.hps_0   "Configuration/HPS-to-FPGA user 0 clock frequency" (desired_cfg_clk_mhz) requested 100.0 MHz, but only achieved 97.3684...
soc_system.hps_0   "QSPI clock frequency" (desired_qspi_clk_mhz) requested 400.0 MHz, but only achieved 370.0 MHz
soc_system.hps_0   1 or more output clock frequencies cannot be achieved precisely, consider revising desired output clock frequencies.
7 Info Messages
soc_system.button_pio   PIO inputs are not hardwired in test bench. Undefined values will be read from PIO inputs during simulation.
```

Figure 7: Generating HDL Window



Generate Completed

```
Info: Reusing file C:/Users/vane_/Desktop/Sistemas Embebidos/GHRD/
Info: rsp_mux: "mm_interconnect_3" instantiated altera_merlin_multiplex
Info: Reusing file C:/Users/vane_/Desktop/Sistemas Embebidos/GHRD/
Info: avalon_st_adapter: "mm_interconnect_3" instantiated altera_avalo
Info: border: "hps_io" instantiated altera_interface_generator "border"
Info: error_adapter_0: "avalon_st_adapter" instantiated error_adapter
Info: error_adapter_0: "avalon_st_adapter" instantiated error_adapter
Info: soc_system: Done "soc_system" with 65 modules, 133 files
Info: qsys-generate succeeded.
Info: Finished: Create HDL des
3/11/2020 02:30:03 PM
Info: qsys-generate succeeded.
Generate: completed with warnings.
```

Figure 8: Generation of HDL Completed

5.3 Run TCL files

In order to submit the pin assigment to the board we need to run the TCL files and for this we choose from the Tools menu the option of Tcl Scripts as shown in Figure 9. After that, a window as shown in Figure 10 will pop up,then we select the first one click on Run, then choose the second one and click run.If the Tcl Scripts File were executed correctly, a window as shown in Figure 11 will pop up.

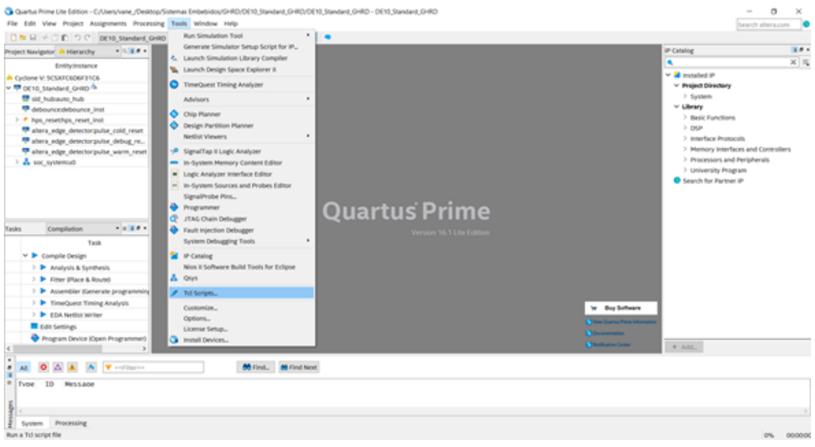
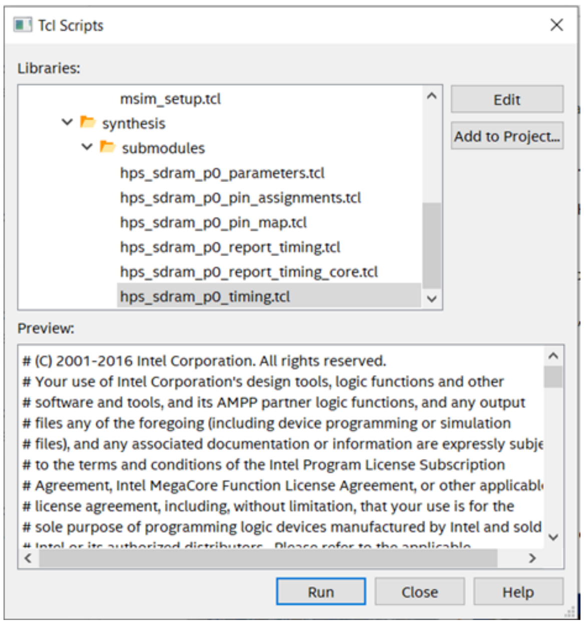

File Edit View Project Assignments Processing Tools Window Help
Run Simulation Tool
Generate Simulator Setup Script for IP...
Launch Simulation Library Compiler
Launch Design Space Explorer II
TimeQuest Timing Analyzer
Advisors
Chip Planner
Design Partition Planner
Netlist Viewers
SignalTap II Logic Analyzer
In-System Memory Content Editor
Logic Analyzer Interface Editor
In-System Sources and Probes Editor
SignalProbe Pins...
Programmer
JTAG Chain Debugger
Fault Injection Debugger
System Debugging Tools
IP Catalog
Nios II Software Build Tools for Eclipse
Qsys
Tcl Scripts...
Customize...
Options...
License Setup...
Install Devices...
Quartus Prime

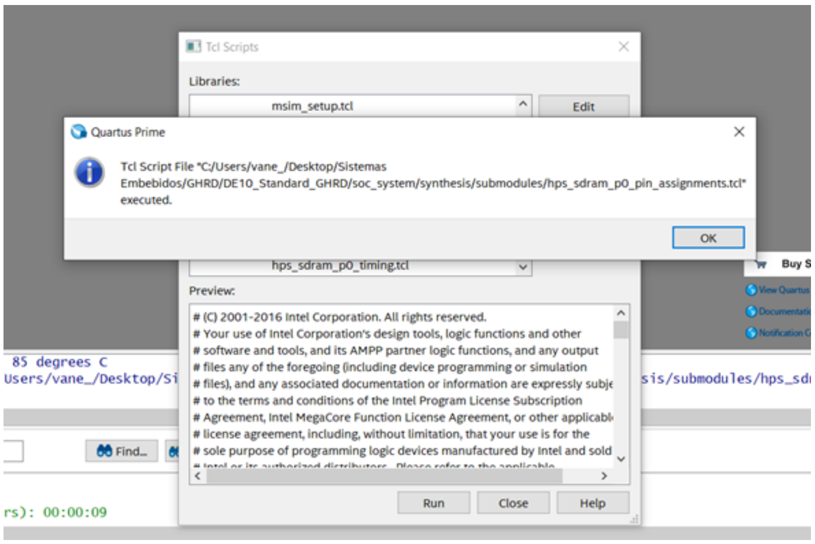
Figure 9: Tools Menu.



msim_setup.tcl
synthesis
submodules
hps_sdram_p0_parameters.tcl
hps_sdram_p0_pin_assignments.tcl
hps_sdram_p0_pin_map.tcl
hps_sdram_p0_report_timing.tcl
hps_sdram_p0_report_timing_core.tcl
hps_sdram_p0_timing.tcl

```
# (C) 2001-2016 Intel Corporation. All rights reserved.
# Your use of Intel Corporation's design tools, logic functions and other
# software and tools, and its AMPP partner logic functions, and any output
# files any of the foregoing (including device programming or simulation
# files), and any associated documentation or information are expressly subje
# to the terms and conditions of the Intel Program License Subscription
# Agreement, Intel MegaCore Function License Agreement, or other applicabl
# license agreement, including, without limitation, that your use is for the
# sole purpose of programming logic devices manufactured by Intel and sold
```

Figure 10: Tcl Scripts Selection



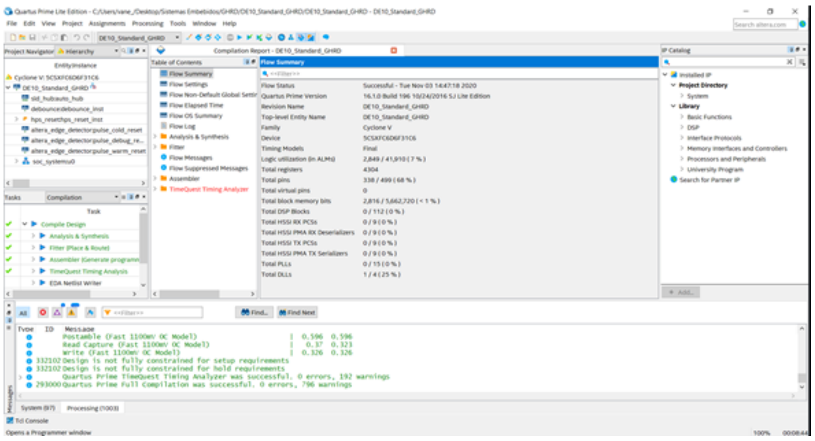
```
Tcl Script File "C:/Users/vane_/Desktop/Sistemas Embebidos/GHRD/DE10_Standard_GHRD/soc_system/synthesis/submodules/hps_sdram_p0_pin_assignments.tcl" executed.
```

```
# (C) 2001-2016 Intel Corporation. All rights reserved.
# Your use of Intel Corporation's design tools, logic functions and other
# software and tools, and its AMPP partner logic functions, and any output
# files any of the foregoing (including device programming or simulation
# files), and any associated documentation or information are expressly subje
# to the terms and conditions of the Intel Program License Subscription
# Agreement, Intel MegaCore Function License Agreement, or other applicabl
# license agreement, including, without limitation, that your use is for the
# sole purpose of programming logic devices manufactured by Intel and sold
```

Figure 11: Tcl file executed

5.4 Compilation

The next part is to do the compilation and if it successful it will be as shown in Figure 12.



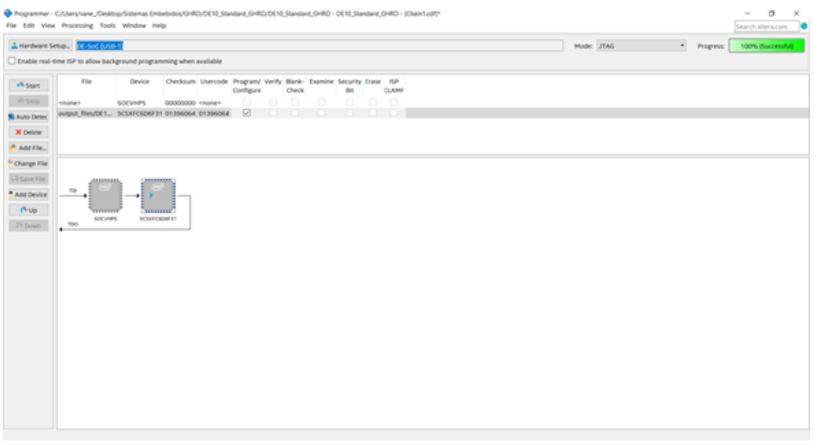
Flow Summary	
Flow Status	Successful - Tue Nov 03 14:47:18 2020
Quartus Prime Version	16.1.0 Build 196 10/24/2016 SJ Lite Edition
Revision Name	DE10_Standard_GHRD
Top-level Entity Name	DE10_Standard_GHRD
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	2,849 / 41,910 (7 %)
Total registers	4304
Total pins	338 / 499 (68 %)
Total virtual pins	0
Total block memory bits	2,816 / 5,662,720 (< 1 %)
Total DSP Blocks	0 / 112 (0 %)
Total HSSI RX PCSs	0 / 9 (0 %)
Total HSSI PMA RX Deserializers	0 / 9 (0 %)
Total HSSI TX PCSs	0 / 9 (0 %)
Total HSSI PMA TX Serializers	0 / 9 (0 %)
Total PLLs	0 / 15 (0 %)
Total DLLs	1 / 4 (25 %)

```
Type  ID      Message
              Postamble (Fast 1100mV 0C Model)                  |  0.596  0.596
              Read Capture (Fast 1100mV 0C Model)               |   0.37  0.323
              Write (Fast 1100mV 0C Model)                      |  0.326  0.326
      332102  Design is not fully constrained for setup requirements
      332102  Design is not fully constrained for hold requirements
              Quartus Prime TimeQuest Timing Analyzer was successful. 0 errors, 192 warnings
      293000  Quartus Prime Full Compilation was successful. 0 errors, 796 warnings
```

Figure 12: Compilation

5.5 Uploading to FPGA

In order to upload the program to the FPGA we need to open the programmer, then select the board and the program and click on Start. If this is successful it will be as shown in Figure 13.



Programmer - C:/Users/vane_/Desktop/Sistemas Embebidos/GHRD/DE10_Standard_GHRD/DE10_Standard_GHRD - DE10_Standard_GHRD - [Chain1.cdf]*

Hardware Setup... DE-SoC [USB-1] Mode: JTAG Progress: 100% (Successful)

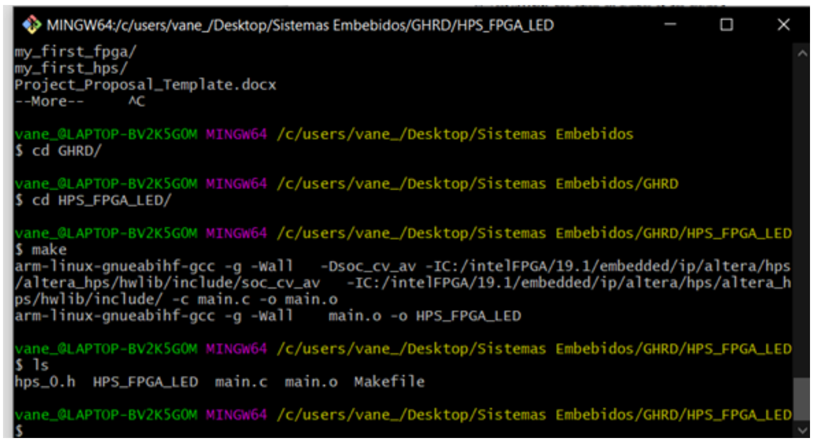
Enable real-time ISP to allow background programming when available

File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank-Check	Examine	Security Bit	Erase	ISP CLAMP
<none>	SOCVHPS	00000000	<none>	☐	☐	☐	☐	☐	☐	☐
output_files/DE1...	5CSXFC6D6F31	01396064	01396064	✓	☐	☐	☐	☐	☐	☐

Figure 13: FPGA Programmer

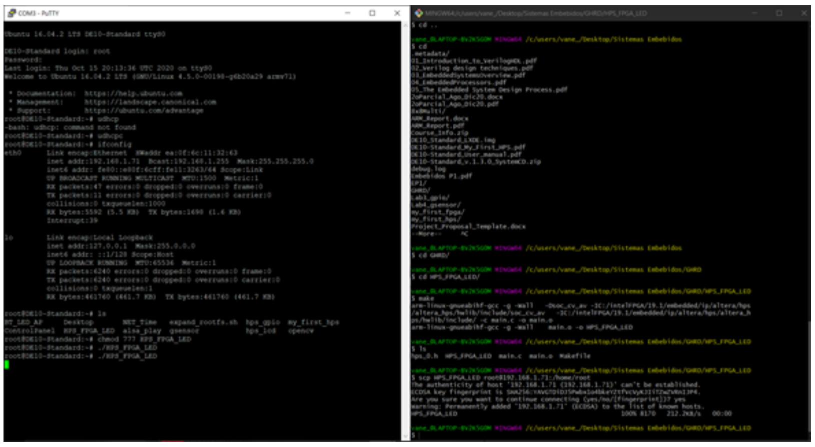
5.6 Executing HPS program

In this section we need to open the embedded command shell that could be found on the embedded folder iside of our folder instalation of intelFPGA as shown in Figure 14. Then we use the command cd to access the folder were we keep our project and executed the make command. After that, we connect our FPGA to our network with the ethernet cable, and to our computer with the mini-USB cable in order to communicate via UART. After the communications is stablish we requested a ip address. When the ip address is given we move to our command shell and move the project to the FPGA. Then run the project with the command shown in Figure 15. After that the leds will start blinking.



```
MINGW64:/c/users/vane_/Desktop/Sistemas Embebidos/GHRD/HPS_FPGA_LED
my_first_fpga/
my_first_hps/
Project_Proposal_Template.docx
--More--      ^C

vane_@LAPTOP-BV2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos
$ cd GHRD/

vane_@LAPTOP-BV2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos/GHRD
$ cd HPS_FPGA_LED/

vane_@LAPTOP-BV2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos/GHRD/HPS_FPGA_LED
$ make
arm-linux-gnueabihf-gcc -g -Wall   -Dsoc_cv_av -IC:/intelFPGA/19.1/embedded/ip/altera/hps
/altera_hps/hwlib/include/soc_cv_av   -IC:/intelFPGA/19.1/embedded/ip/altera/hps/altera_h
ps/hwlib/include/ -c main.c -o main.o
arm-linux-gnueabihf-gcc -g -Wall    main.o -o HPS_FPGA_LED

vane_@LAPTOP-BV2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos/GHRD/HPS_FPGA_LED
$ ls
hps_0.h  HPS_FPGA_LED  main.c  main.o  Makefile

vane_@LAPTOP-BV2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos/GHRD/HPS_FPGA_LED
$
```

Figure 14: Embedded Command Shell



```
Ubuntu 16.04.2 LTS DE10-Standard ttyS0

DE10-Standard login: root
Password:
Last login: Thu Oct 15 20:13:36 UTC 2020 on ttyS0
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.5.0-00198-g6b20a29 armv7l)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
root@DE10-Standard:~# udhcp
-bash: udhcp: command not found
root@DE10-Standard:~# udhcpc
root@DE10-Standard:~# ifconfig
eth0      Link encap:Ethernet  HWaddr ea:0f:6c:11:32:63
          inet addr:192.168.1.71  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::e80f:6cff:fe11:3263/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:47 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5592 (5.5 KB)  TX bytes:1698 (1.6 KB)
          Interrupt:39

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:6240 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6240 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:461760 (461.7 KB)  TX bytes:461760 (461.7 KB)

root@DE10-Standard:~# ls
BT_LED_AP     Desktop       NET_Time   expand_rootfs.sh  hps_gpio  my_first_hps
ControlPanel  HPS_FPGA_LED  alsa_play  gsensor           hps_lcd   opencv
root@DE10-Standard:~# chmod 777 HPS_FPGA_LED
root@DE10-Standard:~# ./HPS_FPGA_LED
root@DE10-Standard:~# ./HPS_FPGA_LED
```

```
$ cd ..

vane_@LAPTOP-8V2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos
$ cd
.metadata/
01_Introduction_to_VerilogHDL.pdf
02_Verilog design techniques.pdf
03_EmbeddedSystemsOverview.pdf
04_EmbeddedProcessors.pdf
05_The Embedded System Design Process.pdf
2oParcial_Ago_Dic20.docx
2oParcial_Ago_Dic20.pdf
8x8Multi/
ARM_Report.docx
ARM_Report.pdf
Course_Info.zip
DE10_Standard_LXDE.img
DE10-Standard_My_First_HPS.pdf
DE10-Standard_User_manual.pdf
DE10-Standard_v.1.1.0_SystemCD.zip
debug.log
Embebidos P1.pdf
EP1/
GHRD/
Lab1_gpio/
Lab4_gsensor/
my_first_fpga/
my_first_hps/
Project_Proposal_Template.docx
--More--     ^C

vane_@LAPTOP-8V2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos
$ cd GHRD/

vane_@LAPTOP-8V2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos/GHRD
$ cd HPS_FPGA_LED/

vane_@LAPTOP-8V2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos/GHRD/HPS_FPGA_LED
$ make
arm-linux-gnueabihf-gcc -g -Wall   -Dsoc_cv_av -IC:/intelFPGA/19.1/embedded/ip/altera/hps
/altera_hps/hwlib/include/soc_cv_av   -IC:/intelFPGA/19.1/embedded/ip/altera/hps/altera_h
ps/hwlib/include/ -c main.c -o main.o
arm-linux-gnueabihf-gcc -g -Wall    main.o -o HPS_FPGA_LED

vane_@LAPTOP-8V2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos/GHRD/HPS_FPGA_LED
$ ls
hps_0.h  HPS_FPGA_LED  main.c  main.o  Makefile

vane_@LAPTOP-8V2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos/GHRD/HPS_FPGA_LED
$ scp HPS_FPGA_LED root@192.168.1.71:/home/root
The authenticity of host '192.168.1.71 (192.168.1.71)' can't be established.
ECDSA key fingerprint is SHA256:[illegible].
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.71' (ECDSA) to the list of known hosts.
HPS_FPGA_LED                                  100% 8170   212.2KB/s   00:00

vane_@LAPTOP-8V2K5G0M MINGW64 /c/users/vane_/Desktop/Sistemas Embebidos/GHRD/HPS_FPGA_LED
$
```

Figure 15: UART Communication and moving the file via internet

6 Problems Encountered

The first problem that I encountered was related with the version of Quartus. For some reason that I am not able to explain, the 19.1 version didn't did create the files from Qsys correctly, so, since the project was made in Quartus 16.1, the professor suggested that we downloaded that version. With the 16.1 version I did not have issues with Qsys, however, when I compiled the project, I realised that Quartus stop working. This happened to me at least 4 times, then I decided to erase the version 19.1 from my computer but that did not resolve the problem. At the end I figure out that the problem was that I wasn't running the tcl files that the program needed. The rest of the laboratory was pretty straight forward and did not give me any trouble.

7 Conclusion

I think that the communication of the HPS with the FPGA is a very important aspect when we are doing a development of embedded systems. That is the main reason why I think this lab help me to understand more the process of it. I think that is important to know the bridge between HPS and the FPGA in order to program it and used them in further projects. With this Laboratory I learned more about how HPS is connected with the FPGA and I have more idea on how we can developed our final project.

8 References

AN 796: Cyclone V and Arria V SoC Device Design Guidelines. (2020). Intel.
Recovery at: `https://www.intel.com/content/www/us/en/programmable/documentation/doq1481305867183.html#git1481305300309`
User Manual.(2017). Intel. Recovery at: `https://www.intel.com/content/dam/altera-www/global/en_US/portal/dsn/42/doc-us-dsnbk-42-5505271707235-de10-standard-use`
`pdf`