How to Document for BTP: Communication System with Hardware Encryption

Mohil Patel - 160070002

Guide:
Prof. Madhav Desai

Spring 2020

Contents

1 How to Use this Document

This document will help in guiding to build the complete project from the base files. It includes all the details and files necessary to build the complete project yourself. This guide uses TIVA TMC4C123G Launchpad as its Microcontroller and Nexys 4 DDR Board as the FPGA. The details concerning each setup is individually covered in the sections and final section contains the details on how to join the each individual components to build the complete project and test it.
This document is accompanied with a folder named "*How_to_Guide*", for all the directory listing this will serve as the base address, i.e. if in the document it is written to copy the file from *How_to_Guide/PC_client_files/Client_loopback_flow.py* this implies the complete location of the file is current location of the *How_to_Guide* folder + the address ahead.
This document is written with a hope to ease up the process of setting up the complete environment required to further develop the project. This will help speed up the environment setup and thus fasten the process of further develop. Hope it serves you well.

2 Analog Circuit Setup

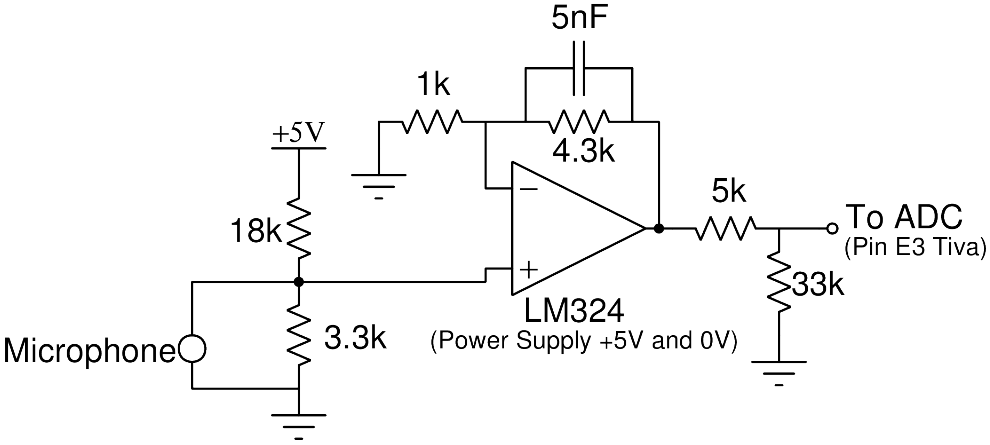


Figure 1: Microphone Circuit Diagram

Figure 1 shows the circuit diagram for the microphone circuit used in the system. Power supplies to the microphone is only +5V and 0V. The Microcontroller (TIVA TM4C123G Launchpad Board) is used to supply the power directly to the circuit, it does not use external source for the power supply. Depending on the variation of the microphone used one may need to re-tune the amplification of the OPAMP circuit (i.e. 1k and 4.3k resistor). In our setup as we are using the onboard ADC of the TIVA Board (Microcontroller), we need to ensure that the audio signal is properly captured within the ADC voltage limits which are 3.3V to 0V. This tuning can be achieved by varying the amplification resistor (1k and 4.3k) and limiting resistors (5k and 33k).

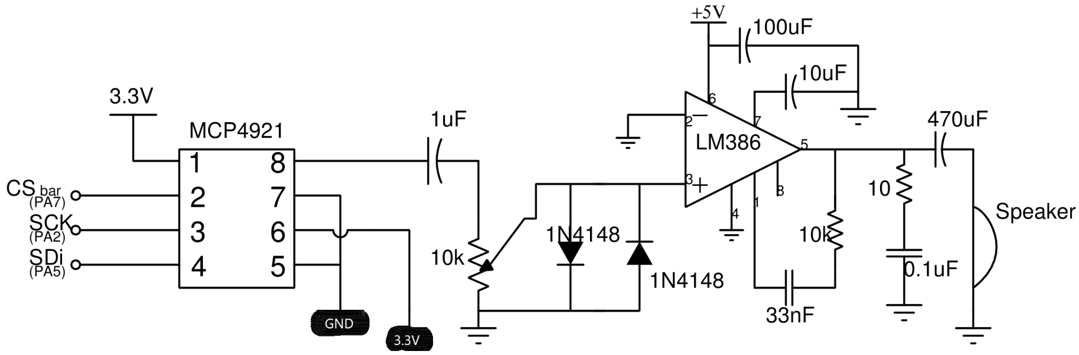


Figure 2: Speaker Circuit Diagram

Figure 2 shows the circuit diagram of the speaker circuit along with the DAC (MCP4921) setup. Note all the power supplies for the circuit are being provided by the Microcontroller Board (TIVA TM4C123G Launchpad) itself. Here as we have used a 10k pot directly in the circuit it can be easily used to control the levels of the signal and thus the output volume. This will help in tuning the volume as required without requiring to change the resistors in the circuit.
Both the figure 1 and 2 shows the pins corresponding to the TIVA TM4C123G Launchpad Board (Microcontroller) where the connections need to be made to ease up the connection process.

3 Microcontroller Setup

The Microcontroller used in this project is TIVA TM4C123G Launchpad Board, from now onwards we will refer to it as Tiva Board for rest of the section. The Tiva Board is programmed using Code Composer Studio (CCS 9.1.10) and it used Tivaware Peripheral Driver Library (TivaWare_C_Series-2.1.4.178) for accessing the necessary functionalities. This guide will help setup the project environment and build the code using the files provided in the *How_To_Guide/ Microcontroller_Files/Main_Files*.
In this guide we will extensively use take the help from the Document located at *How_To_Guide/Documents/TM4C123G_LaunchPad_Workshop_Workbook.pdf* (Refered to as PDF in this section). The guide will step by step instruct you to go to specific locations in the PDF and do the steps. If you are new to the Microcontroller programming or are using Tiva Board for the first time, we would strongly recommend you to at least complete first 2 chapters of the PDF (i.e.

everything before page 3-1) before starting this setup. Also you may go through Timer, ADC and UART chapters but they are needed only if you plan to develop further and can be reviewed at a later stage.

1. In the PDF, starting from page 1-11 complete the steps 2. till 11. finishing at page 1-15. This will complete the installation of Code Composer Studio.
2. After this complete the step 12. on page 1-16 to install the TivaWare.
3. Now go to page 2-15 in the PDF, complete the steps from step 2. to step 9. finishing at page 2-19. This is create a new empty project and setup the TivaWare library for the project.
4. Now follow the step 10. at page 2-20 but instead of copy the files from the location *How_To_Guide/Microcontroller_files/Main_Files* into the project. You will need to add all the files from the folder into the project. Keep a note to select "copy the files option" while importing. Also note that you will be replacing the *tm4c123gh6pm_startup_ccs.c* from the project. This is done basically to set the interrupt service routine for Timer 1 Interrupt. (Line 108, in the *tm4c123gh6pm_startup_ccs.c*).
5. After adding file complete steps 11. and 12. on page 2-21 and 2-22. This will setup the complete project.
6. Now Build the project and load it in the TIVA board. This will later be connected to in the complete system, details concerning the same will be covered in section 7.

4 AHIR System Setup

The following section will cover the details regarding the installation of the AHIR tools which can help convert the AA code directly to vhdl files. We have coded the encryption module using AA language and thus we will require AHIR tool to convert it to vhdl file. Following is step by step process to install AHIR Tool. We will be using the documents provided by Prof. Madhav Desai for EE789-Algorithmic Design of Digital Systems, to guide the installation of the AHIR tool. Please refer to the document located at *How_To_Guide/Documents/AHIR_Intro.pdf*, it will be refered as the PDF for rest of the section.

1. To install the AHIR tool you will require to install ubuntu 12.04, you can either install it along side your already existing OS or you can use virtual box to install ubuntu 12.04. For our setup we had used a virtualbox setup, there are many guides available online to install the same refer to them for installation. Link to one such guide: Guide for Ubuntu 12.04 installation.
2. Once installed refer to the last page in the PDF stated earlier. This will guide you to install necessary components, follow it from start till install java run-time (openjdk), complete all the install till then.

3. For the ghdl installation you will need to specifically install ghdl 0.31, you can refer to this guide here for installation.

4. Continue with the last step. Note it is possible that you might not be able to build the release as stated in the PDF as the GIT Repository is still being developed and thus you may be need to refer to the guidelines provided on the repository for proper installation. In case you are not able to build the new release properly, we have put a already build release in the location *How_To_Guide/AHIR/ahir_release.tgz*, you can use that directly (Note it still need the installation of all the necessary dependencies stated in the PDF). Using this will let to skip the last step of the PDF.

5. This complete the installation of the AHIR tool, for a small example please refer to *How_To_Guide/Documents/AHIR/upload.tgz* taken for EE789-Algorithmic Design of Digital Systems course.

5 AA Code to VHDL

Once AHIR tool is setup properly, follow the following instructions to build the vhdl file from the AA files.

1. Go to AHIR release library via terminal and do:
   source ahir_bashrc

2. Copy the contents of *How_To_Guide/AA_files* into the folder where you want to build the project.

3. From the AHIR release folder cd into the new folder made in the make step.

4. Apply the following command to build the project:
   make all

5. After the project is build completely copy the files *ahir_system.vhdl* & *ahir_system_global_packages.vhdl* and store them properly as they will be required for building the project.

6 FGPA Setup

For this section we will we taking help from the Nexys 4 DDR Programming Guide, which can be found here. From here on the programming guide will be referred to as the guide till the end of the section.

1. Install the Vivado available from the Xilinx website. (While building this demo we were using Vivado 2019.1)

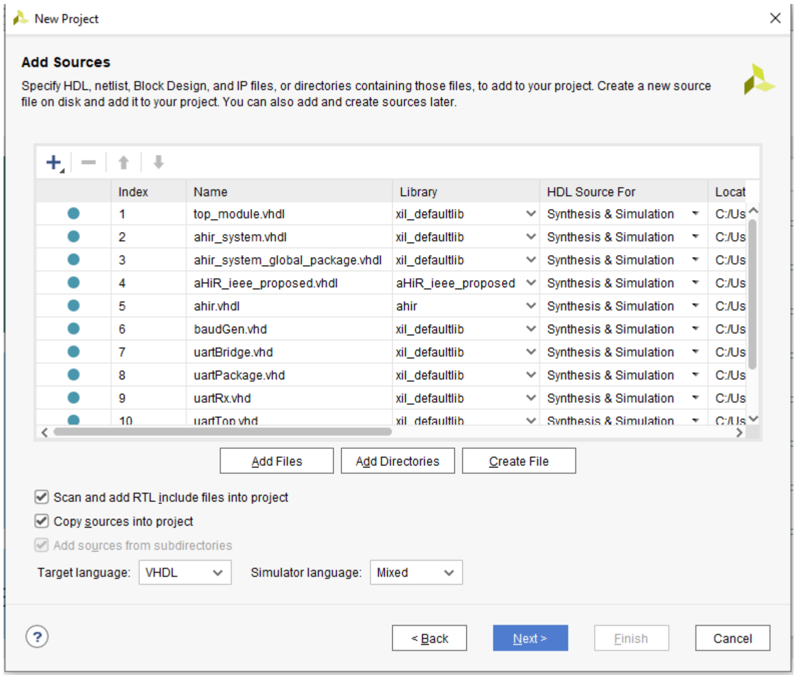
2. Go to the guide referred above and start following the tutorial to create a new project. Complete the steps from 1.1 to 1.3.

3. For the step 1.4 you will need to add files from the location *How_To_Guide/ FPGA_files*. Please note that you will not be adding the file from the constraints folder here, it will be done in step 1.6. Also note that the *ahir_system.vhdl* & *ahir_system_global_packages.vhdl* files are the same that you create during the end of the AA Code to VHDL, you can replace this files with the once generate after section 5.

4. The files *ahir.vhdl* & *aHiR_ieee_proposed.vhdl* are files taken from the AHIR release library and are necessary for code synthesis. You can find them from */release/vhdl* location. These files need to set to specific library in step 1.4, for that you will need to change the library section for this files. Refer to figure 3, and change the library as shown. The library for other files will remain the same.

5. Now continue with the guide from step 1.5 onwards (we do not add any existing IP in this project) till the end of section 2 in the guide, i.e. till step 2.10.

6. After completion of step 2.10 you will need to select the way you will program the device. We would suggest using JTAG (i.e. section 3) if you want to program the device while debugging, where once the code has been been finalize you can do Quad SPI (section 5).



Add Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those files, to add to your project. Create a new source file on disk and add it to your project. You can also add and create sources later.

	Index	Name	Library	HDL Source For	Locat
●	1	top_module.vhdl	xil_defaultlib	Synthesis & Simulation	C:/Us
●	2	ahir_system.vhdl	xil_defaultlib	Synthesis & Simulation	C:/Us
●	3	ahir_system_global_package.vhdl	xil_defaultlib	Synthesis & Simulation	C:/Us
●	4	aHiR_ieee_proposed.vhdl	aHiR_ieee_proposed	Synthesis & Simulation	C:/Us
●	5	ahir.vhdl	ahir	Synthesis & Simulation	C:/Us
●	6	baudGen.vhd	xil_defaultlib	Synthesis & Simulation	C:/Us
●	7	uartBridge.vhd	xil_defaultlib	Synthesis & Simulation	C:/Us
●	8	uartPackage.vhd	xil_defaultlib	Synthesis & Simulation	C:/Us
●	9	uartRx.vhd	xil_defaultlib	Synthesis & Simulation	C:/Us
●	10	uartTop.vhd	xil_defaultlib	Synthesis & Simulation	C:/Us

☑ Scan and add RTL include files into project

☑ Copy sources into project

☑ Add sources from subdirectories

Target language: VHDL

Simulator language: Mixed

Figure 3: Add Files Vivado

By the end of this section you will have programmed the FPGA properly. In the final section we will discuss who to combine all the elements properly and start the system. For now keep the programmed FPGA with you.

7 PC client and server setup

As the Code Composer Studio and Vivado were installed in windows, We installed PyCharm to run the python client which is used as the PC client to communicate to the server. For this guide we had deployed the server as the local host and the python client is set for the loop-back operation. This can be easily changed by changing the mqtt server subscriber and publisher topics. You will need to read about the mqtt server before you are able to edit the code. In this guide the server is deployed using Windows Subsystem for Linux and the python script is doing a loop-back operation.

1. Install PyCharm to run the Python script located at How_To_Guide/PC_client_files.
2. Instal Windows Subsystem for Linux to run the mqtt server.
3. To install mqtt server, type in terminal following commands:
   sudo apt-get update
   sudo apt-get install mosquitto
4. Start the server, type in terminal following command:
   mosquitto

8 Combining All Elements for Final System

The final section is divided into 2 parts, first part will focus on connections between different components, and the second part will focus on programming and running the system. we would suggest that you first properly connect all the peripherally before programming the devices and running the python script.

8.1 Connections in between different components

As we have noted earlier, this demo is built specifically for Tiva TM4C123G Launchpad Board as Microcontroller and Nexys 4 DDR as the FPGA Board. Whenever we refer to a pin on microcontroller (or FPGA) it would imply the pin on Tiva TMC123G Launchpad Board (or Nexys 4 DDR Board). Please keep this in mind.

8.1.1 Analog Circuit to Microcontroller Connections

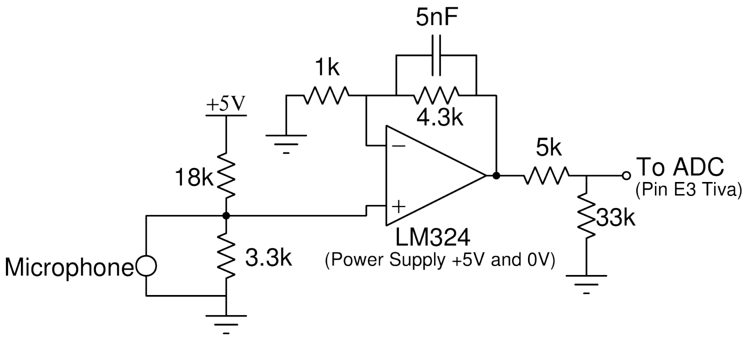


Figure 4: Microphone Circuit Connections

As can be clearly seen in figure 4, you will be required to connect the ADC pin E3 of microcontroller to the point shown in the figure. Also the power supply to LM324 as well as +5V is provided from the Tiva TM4C123G Board itself.

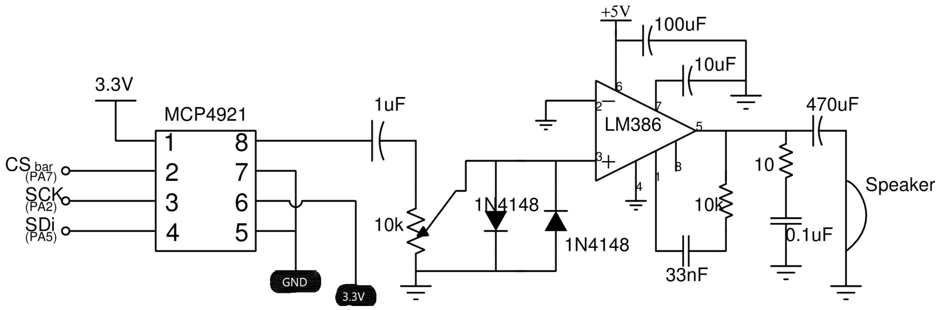


Figure 5: Speaker Circuit Connections

As shown in figure 5, connections to MCP4921 to Tiva TM4C123G is as shown. The power supply i.e. both 3.3V and 5V are also provided by the Microcontroller itself.

8.1.2 Microcontroller to FPGA Board

All the connections are in between Nexys 4 DDR Board and TIVA TM4C123G Board.

Nexys 4 DDR Pin (FPGA)	TIVA TM4C123G Pin (Microcontroller)
Pin E0	Pin JA[2]
Pin E1	Pin JA[1]
Pin F3	Pin JA[3]
GND	GND

Table 1: Connections

8.1.3 Rest of the connections

- To power the TIVA TM4C123G it needs to be connected to PC via a USB cable
- The FPGA Board is connected to the PC via a USB cable, this cable will be used to transmit data to PC client via UART. Note that depending on the port USB port connected, you will need to update the *serial.port* in the python script. You can find the serial port by going into device manager and identifying the USB port assigned to the FPGA cable under the ports tab.

8.2 Programming and starting the system

In this we will demonstrate the start up sequence to start the communication system. But first complete the following steps before we can start:

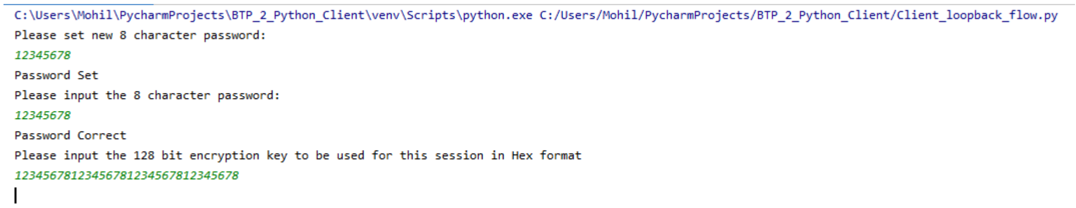
1. Complete all the physical connections as noted in section 8.1.
2. Program the TIVA TM4C123G as noted in section 3.
3. Convert the AA code to vhdl and program the Nexys 4 DDR (FPGA) Board as noted in section 6.
4. Go to the python script at How_To_Guide/PC_client_files/Client_loopback_flow.py and edit the serial port to the port which is assgined to the FPGA USB Cable. This can be found out from device manager under ports tab.
5. Start the MQTT server as noted is section 7.

There are 2 distinct startup sequence one may observe, each has been described below in details.

8.2.1 First Time users

From first time we mean the conditions when the Microcontroller is just programmed and system has been never started after that. Basically a first time user will be first asked to set a password on the microcontroller. This will basically lock the device and whenever the user needs to access the device he/she will be required to provide the correct password set first time. This is to ensure additional security. Following will be sequence for the steps required to follow to start the system for first time user.

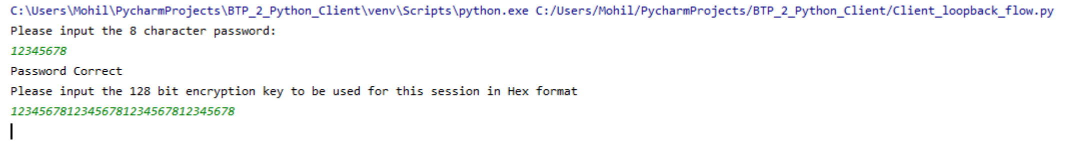
1. Hold the reset of the microcontroller, and while holding start the python script.
2. Let go the reset and wait for around 10 seconds, you will see a prompt asking to set a new password as can be seen in figure 6.
3. After setting the new password, you will be prompted to input the password.
4. Once correctly inputted, system will prompt you to provide encryption key for this session. This key must be hexadecimal character.
5. After the key is added the system will start. In case it does not repeat from the start again.



```
C:\Users\Mohil\PycharmProjects\BTP_2_Python_Client\venv\Scripts\python.exe C:/Users/Mohil/PycharmProjects/BTP_2_Python_Client/Client_loopback_flow.py
Please set new 8 character password:
12345678
Password Set
Please input the 8 character password:
12345678
Password Correct
Please input the 128 bit encryption key to be used for this session in Hex format
12345678123456781234567812345678
|
```

Figure 6: First Time User Instructions

8.2.2 Returning User



```
C:\Users\Mohil\PycharmProjects\BTP_2_Python_Client\venv\Scripts\python.exe C:/Users/Mohil/PycharmProjects/BTP_2_Python_Client/Client_loopback_flow.py
Please input the 8 character password:
12345678
Password Correct
Please input the 128 bit encryption key to be used for this session in Hex format
12345678123456781234567812345678
|
```

Figure 7: Returning User Instructions

For the returning user the instructions are mostly the same except for the step where it will be asked to set a new password. Following are the steps:

1. Hold the reset of the microcontroller, and while holding start the python script.
2. Let go the reset and within around 10 seconds you will see a prompt asking for the password as seen in figure 7.
3. Once the password is inputted correctly you will be prompted to input the encryption key.
4. After the encryption key is added, the system will start. (In case it does not repeat the above steps)