Developing Zynq Software with Xilinx SDK

Lab 11

TE HTU21D Pmod Standalone Environment

Feb 2018
Version 02

AVNET
Reach Further™

Lab 11 Overview

The goal of this lab will be to familiarize yourself with the basic I2C communication process taking place between your MiniZed and the TE HTU21D Pmod. A basic application that reads the data off the HTU21D Pmod and displays it in a terminal program will be imported into SDK. We will then add the needed math libraries to the application and then run the application.

Lab 11 Objectives

When this tutorial is complete, you will be able to:

- Create a bare metal BSP and import application files into SDK
- Add libraries to your standalone applications
- Run and explore a PL I2C Controller Pmod application based of the TE HTU21D Pmod

Experiment 1: Create a Standalone BSP and an Application for the HTU21D Pmod

Experiment 1 General Instruction:

Explore the IIC driver and generate blank software application project. Import code from the following folder:

C:\Speedway\ZynqSW\2017_4\Support_documents\Lab11

Experiment 1 Step-by-Step Instructions:

1. Launch Xilinx Software Development Kit (SDK) if not already open. **Start → All Programs → Xilinx Design Tools → SDK 2017.4 →Xilinx SDK 2017.1**.

Figure 1 – The SDK Application Icon

2. Set or switch the workspace to the following folder and then click the **OK** button:

   C:\Speedway\ZynqSW\2017_4\SDK_Workspace

3. Before we add our application, we are going to explore the PL I2C peripheral by looking through the documentation associated with that driver. **Open** the **system.mss** file located under the standalone_bsp_0 in the Project Explorer window.

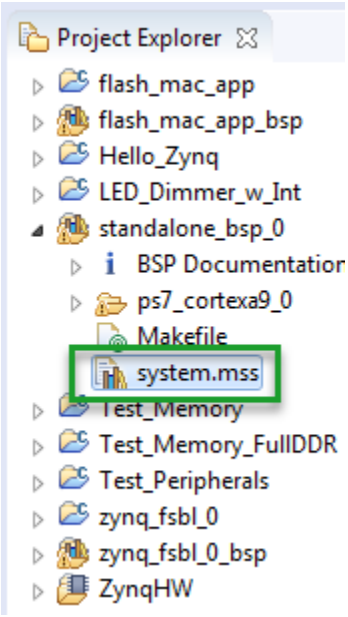

Figure 2 -- Open System.mss

4. If you look in the **system.mss** under Peripheral Drivers, you will find one for "**axi_iic_2 iic**" with Documentation and Import Examples hyperlinks. Click on the **Documentation** link to read a bit about the driver that will be used to exercise this peripheral. Close document when finished

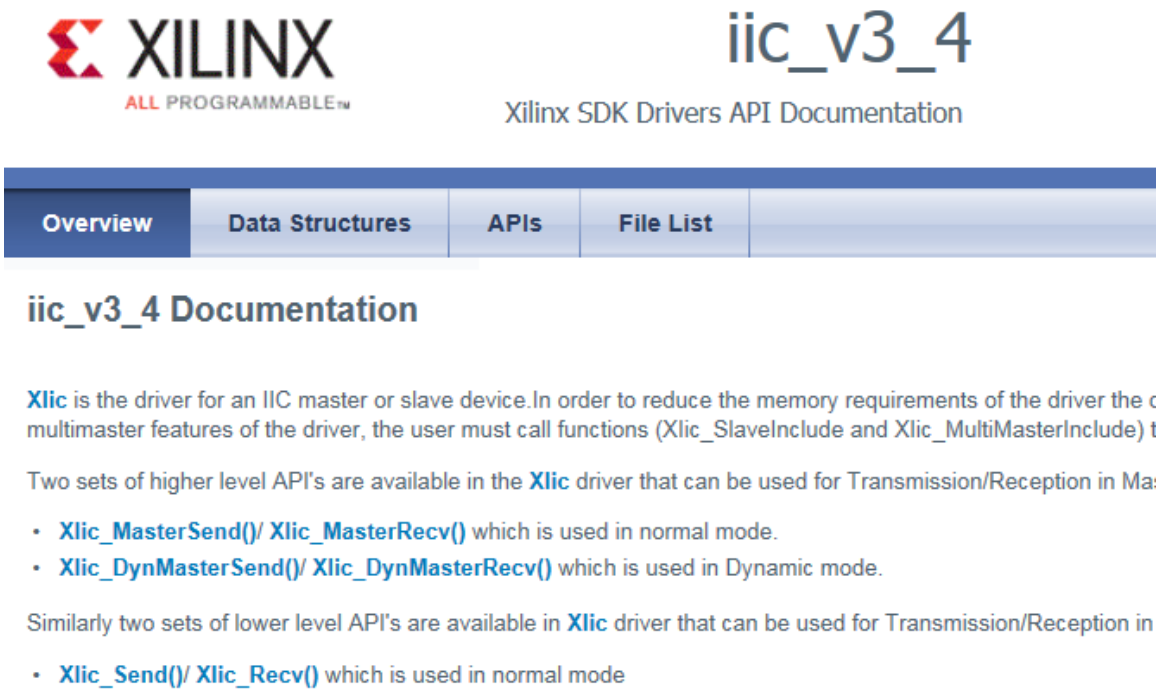

Figure 3 – IIC Driver Documentation

5. Next we will create a new Application Project by selecting **File→New→Application Project**. This will open up a new Application Project wizard.

6. Next, set the Project name to **HTU21D** and the Board Support Package to **Use existing standalone_bsp_0.** Click **Next** to open up available templates. The reason we wish to use the standalone_bsp_0 and not either the FSBL or Flash Mac App BSP is because we do not need either of the libraries that were added to those BSPs. The standalone_bsp_0 will suffice.

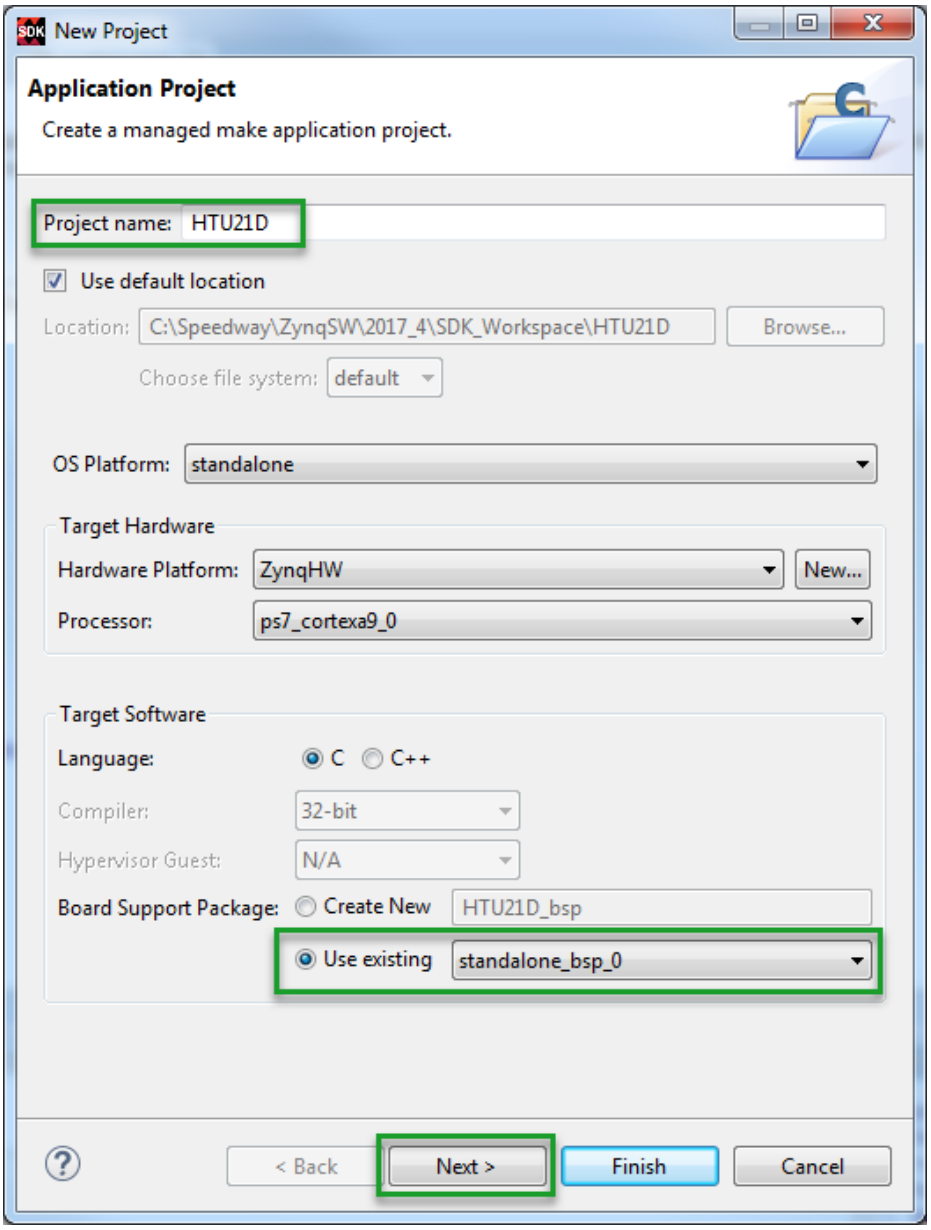

Figure 4 – Create New Application Project

7. For this application we are going to select **Empty Application**. Then click **Finish** to generate the HTU21D application project.

You should now be able to see the HTU21D application in the Project Explorer window

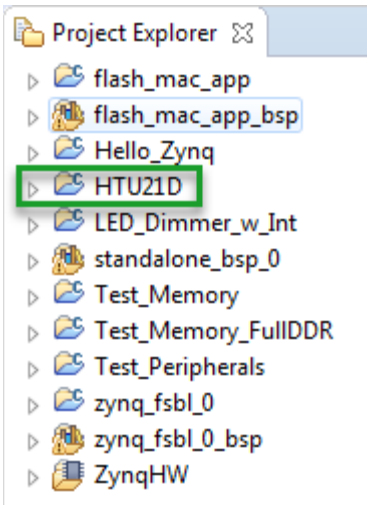

Figure 5 – HTU21D Application in Project Explorer Window

8. We now want to import the HTU21D application code from the supporting documents. **Expand** your HTU21D application in the Project Explorer window and **right click** src and select **Import**.

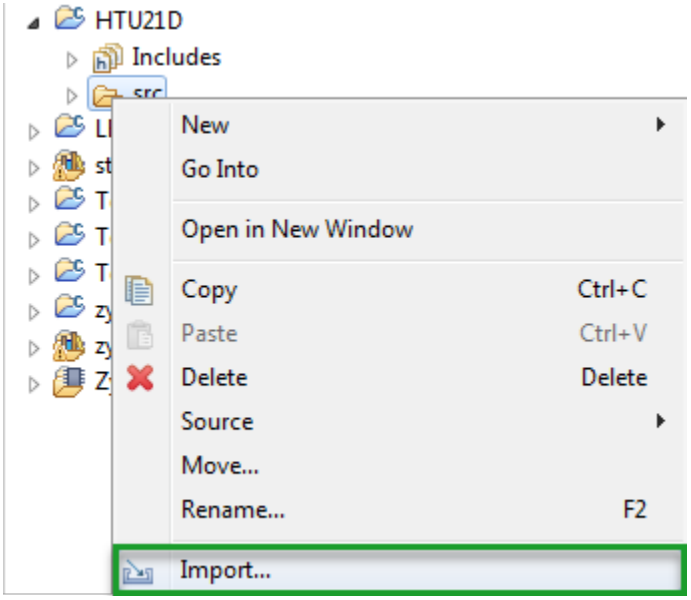

Figure 6 – Import into SRC Folder

9. Now that the Import window is open, select **General→File System** and click **Next**.

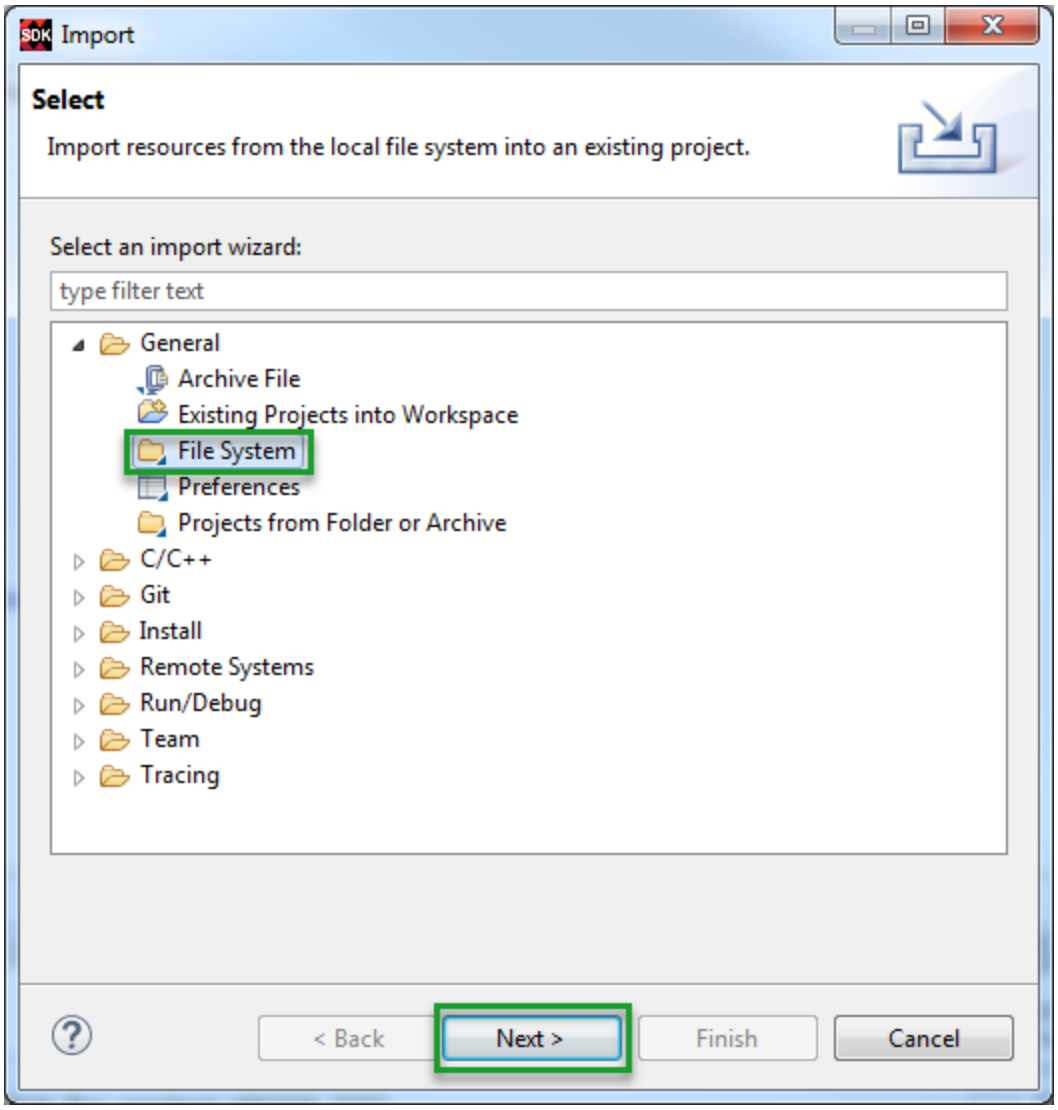

Figure 7 – Import File System

10. We are now in the Import window in which we will specify the directory we wish to import from. Select the upper **Browse** button. Browse to the following directory and select **OK** : **C:\Speedway\ZynqSW\2017_4\Support_documents\Lab11**
    Now click on **Select All** then click **Finish**.

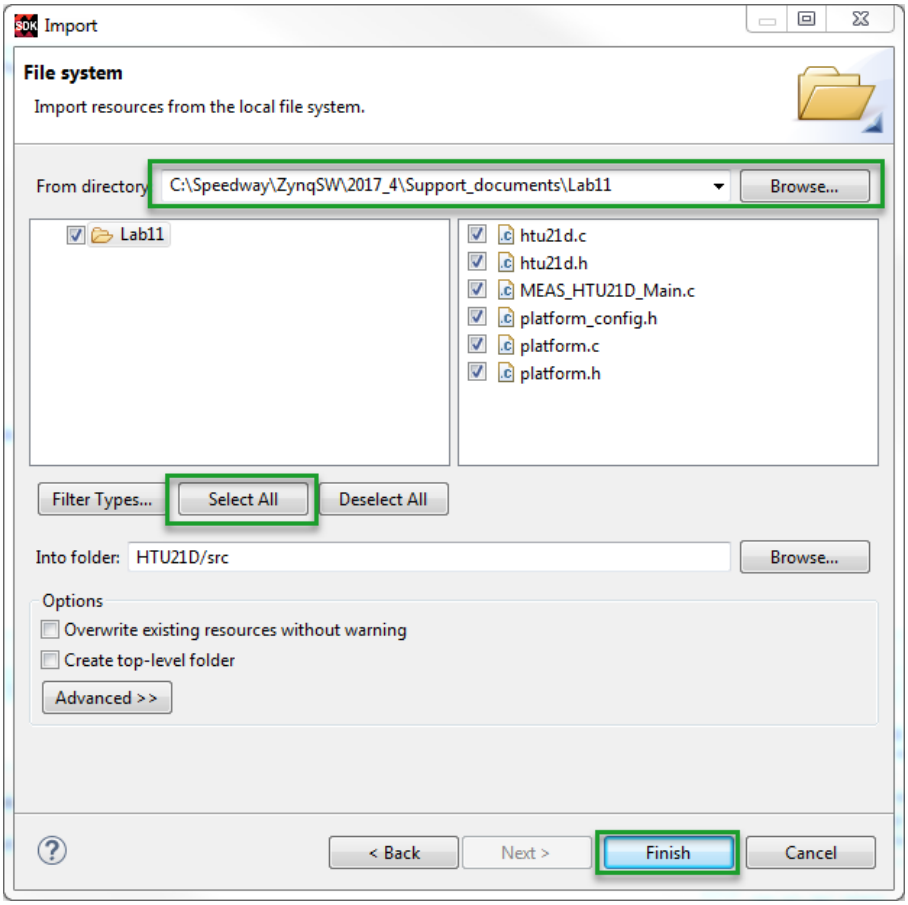

Figure 8 –Importing Supporting Documents

11. Your application will now build. Please refer to the Console tab where you can see everything build. You will notice some build errors in relation to some math functions such as pow and log10. This is due to the math library not being added.

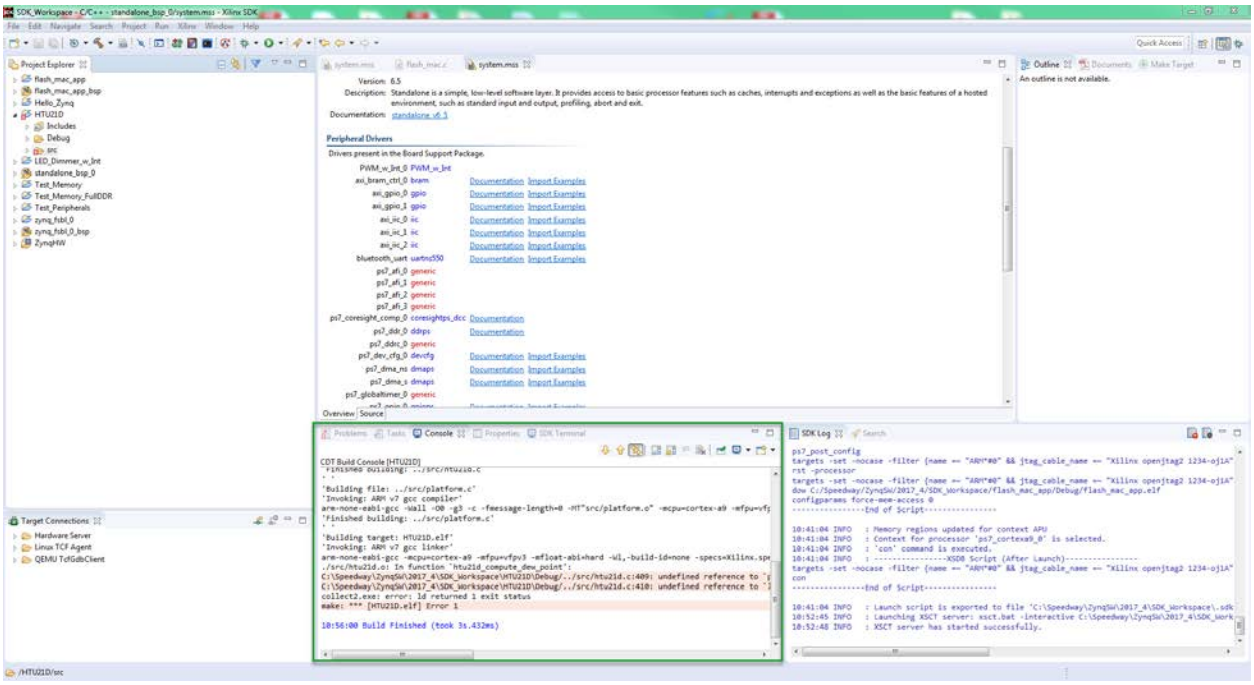

Figure 9 -- Application Build Errors

12. We can fix these errors by adding the math library to the application. Right click on the HTU21D application and select C/C++ Build Settings

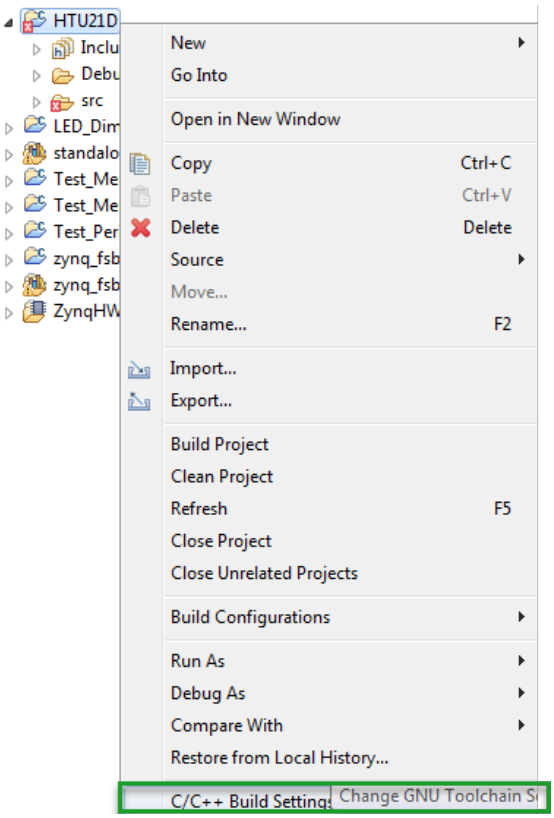

Figure 10 -- Modify Build Settings

13. Now select **Libraries** under ARM v7 gcc linker, then select the **add libraries icon**  and type m for math. Click **OK.**

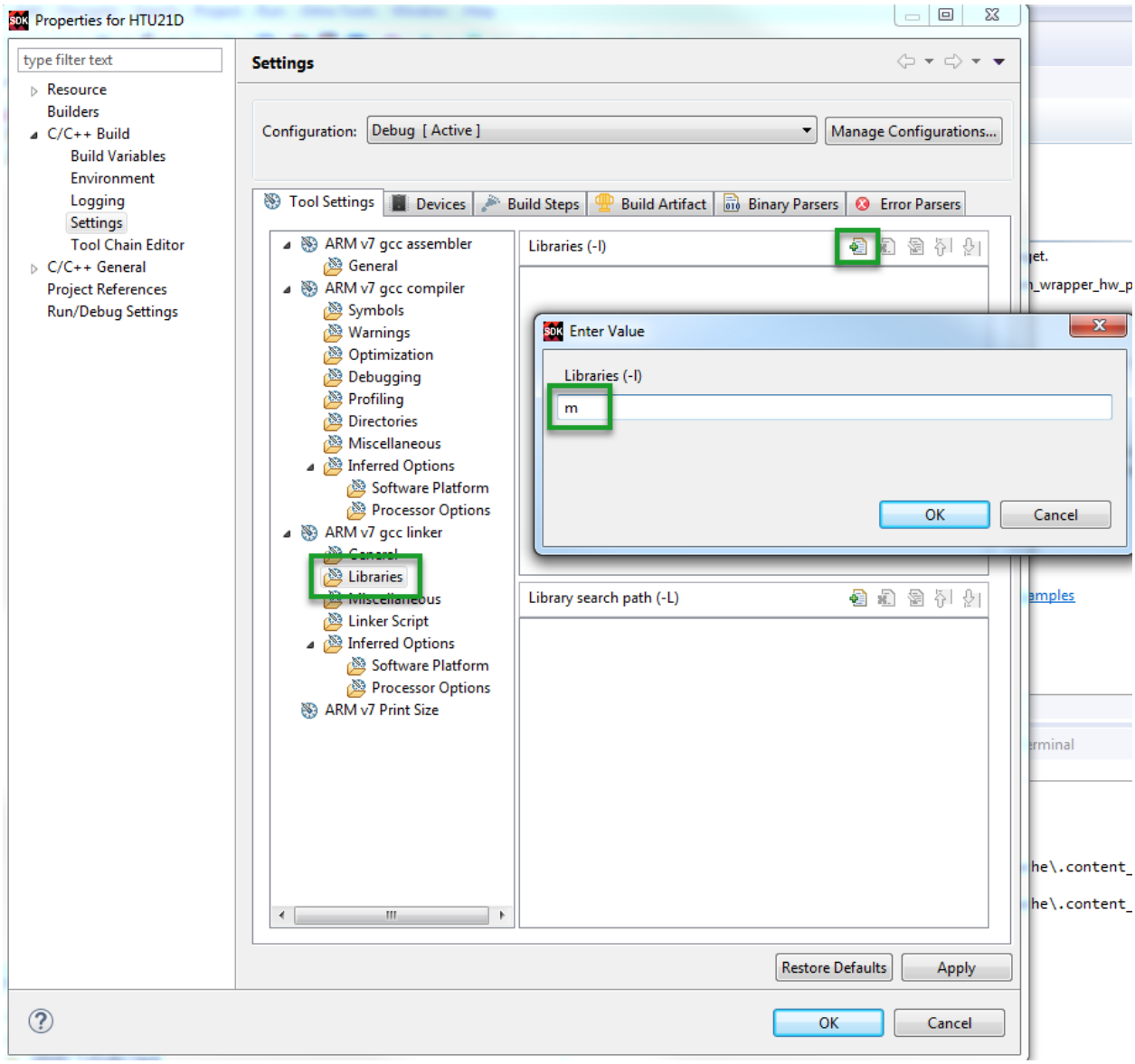

Figure 11 -- Add Math Library

14. Now to finish adding the library select **Apply** then **OK** to return to the SDK window. Now looking in the SDK Console you will see the Application built correctly without any errors. We are now ready to setup the hardware to run the application on.

Experiment 2: Hardware Setup to Run HTU21D Pmod Application

Experiment 2 General Instruction:

Setup hardware with HTU21D on Pmod 2 for MiniZed.

Experiment 2 Step-by-Step Instructions:

1. Set the MiniZed boot mode switch SW1 to JTAG mode ('J' for JTAG) as shown below.
2. If not already, disconnect the MiniZed USB-JTAG/UART port J2 to power off the board

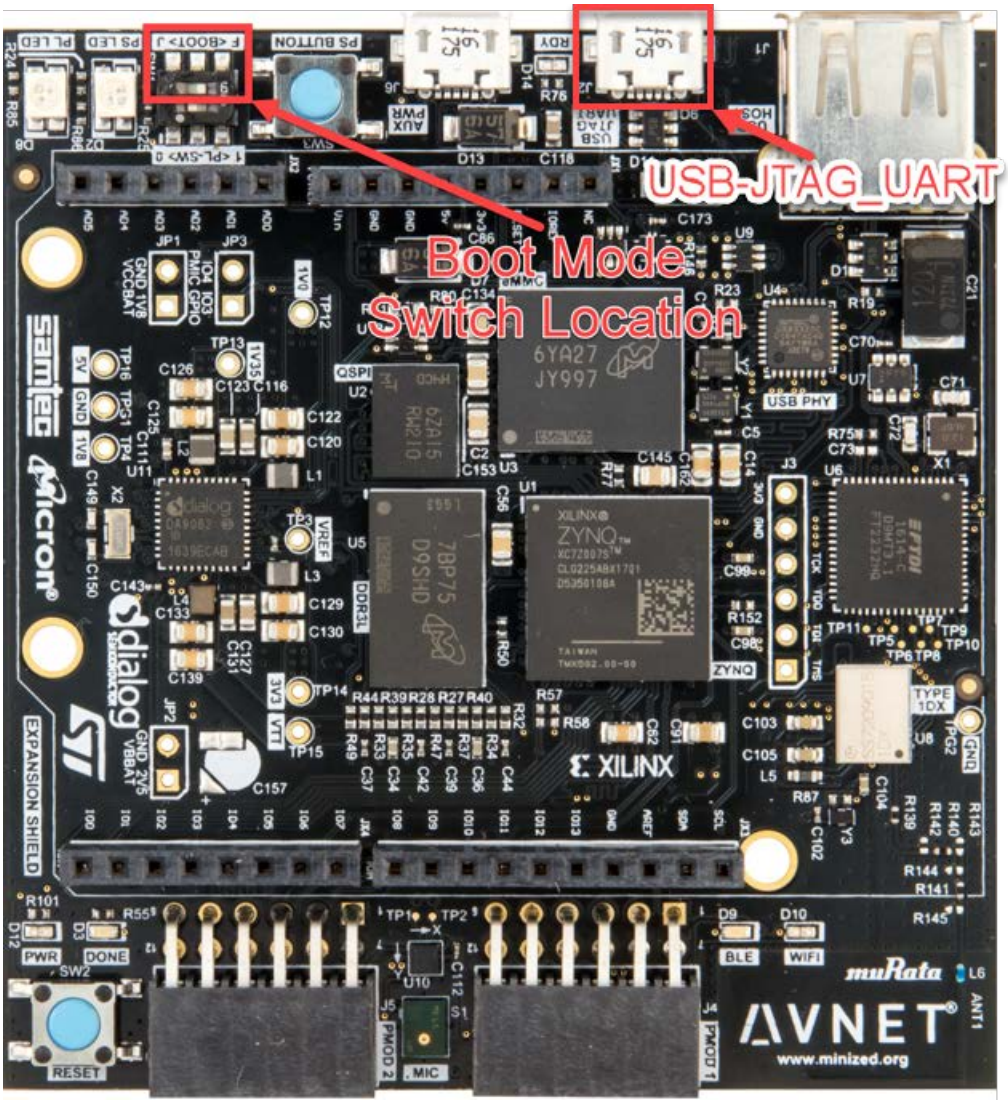

Figure 12 – MiniZed Switch / USB-JTAG-UART micro-USB Connector Location

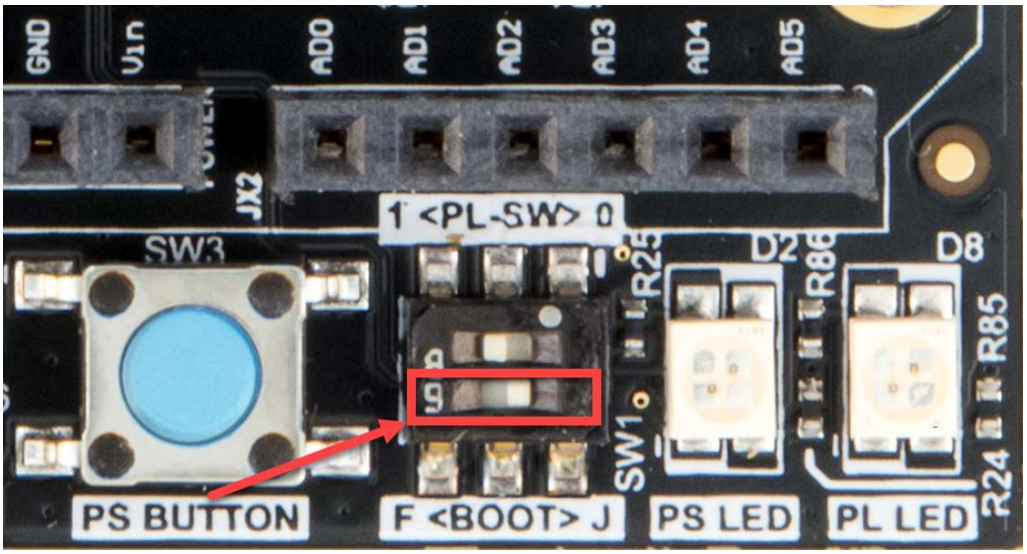

Figure 13 – JTAG Boot Mode

3. Now connect your HTU21D Pmod to the Pmod 2 connector on the MiniZed

4. Reconnect the MiniZed USB-JTAG/UART port J2 to power on board.

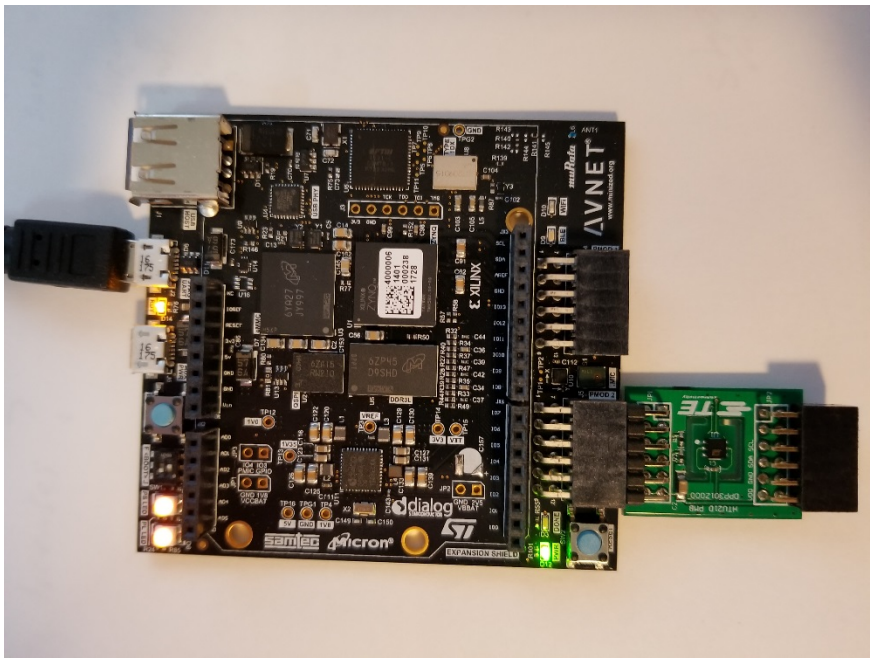

Figure 14 – HTU21D Pmod Installed on MiniZed

5. Use Device Manager to determine the COM port for your MiniZed. In Windows 7, click **Start → Control Panel**, and then click **Device Manager**. Click **Yes** to confirm.

6. Expand ***Ports***. Note the COM port number for the USB Serial Port device. This example shows COM25.

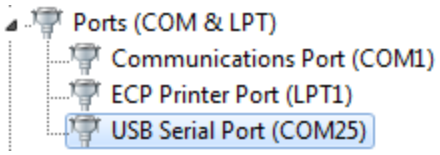

Figure 19 – Find the COM port number for the USB Serial Port device

7. Open a Terminal, such as **Tera Term**, and set the **COM port** to active COM setting for your board and set the **Baud Rate of 115,200**.

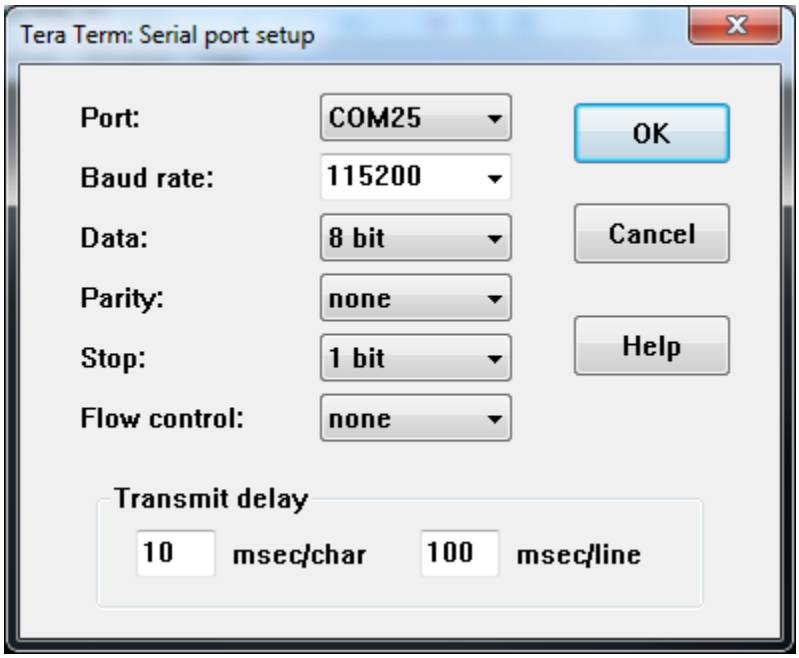

Figure 15 -- Terminal Setup

8. In SDK program the PL first by clicking the  icon or selecting **Xilinx Tools → Program FPGA**. The default options are acceptable. Click **Program**. When complete, the Blue DONE LED should light.

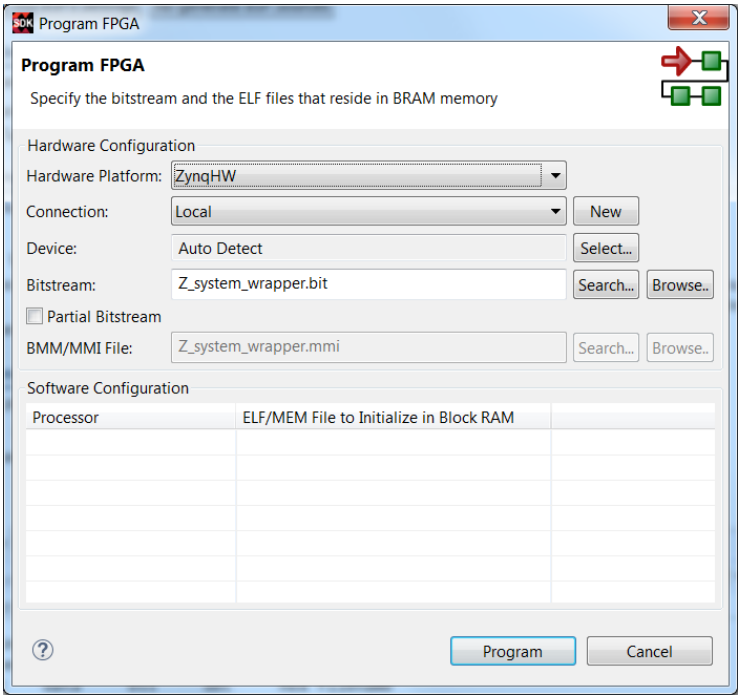

Figure 16 – Program FPGA

Experiment 3: Run and Explore HTU21D I2C Pmod Application

Experiment 3 General Instruction:

Launch the HTU21D application on the target hardware and experiment with reading the temperature, humidity, and dew point. Explore the application source code.

Experiment 3 Step-by-Step Instructions:

1. **Right click** on your HTU21D application in SDK then **select** Run as → **Launch on Hardware (System Debugger)** This will launch your HTU21D application.
    a. After startup, it is a good idea to reset the sensor. This puts it in a known state. Do this by selecting (1) in the console application.

Now the sensor and the software are setup and ready to use. This first step only needs to be performed at power up.

b. The console application option (2) displays a menu that allows the user to select from the four possible resolution modes of the sensor.

c. The console application option (3) reads both the temperature and relative humidity values and displays each of them once.

d. The console application option (4) reads the temperature and relative humidity 20 times each at approximately two measurement pairs per second and displays them to the screen in real time.

e. The console application option (5) computes the dew point from the last measured temperature and relative humidity values.

f. The console application option (6) reads the HTU21D's battery status and displays it to the console.

g. The console application option (7) reads the HTU21D's heater status and displays it to the console.

h. The console application option (8) sends the I2C command to the HTU21D device that enables the on-chip heater.

i. The console application option (9) sends the I2C command to the HTU21D device that disables the on-chip heater.

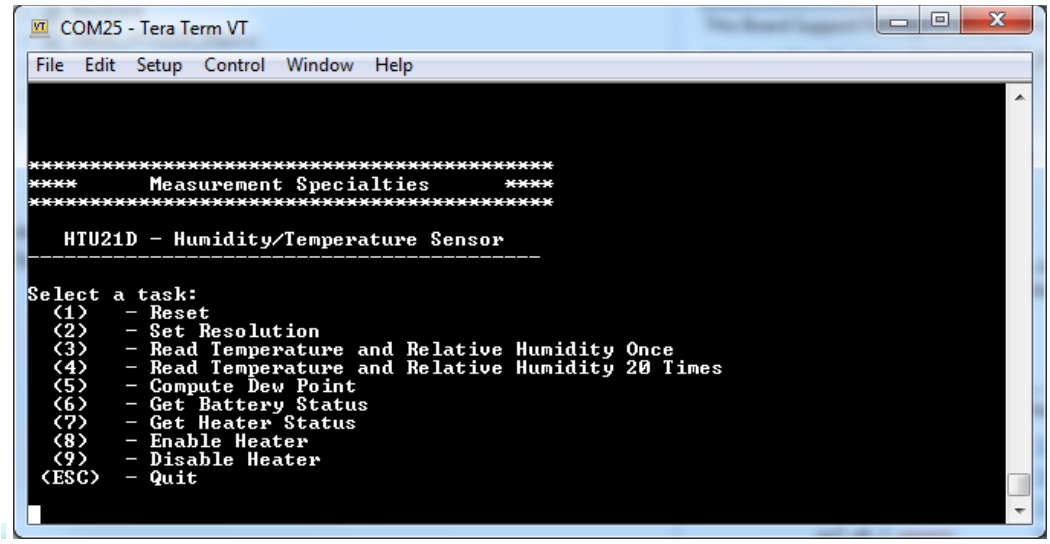

Figure 17 – Example of Terminal Output

2. Now that we have our HTU21D application running we shall examine the contents of the Pmod application. **Expand** the SRC folder of the HTU21D Application in the Project Explorer window. In there, you will see the six files you imported from your supporting documents.

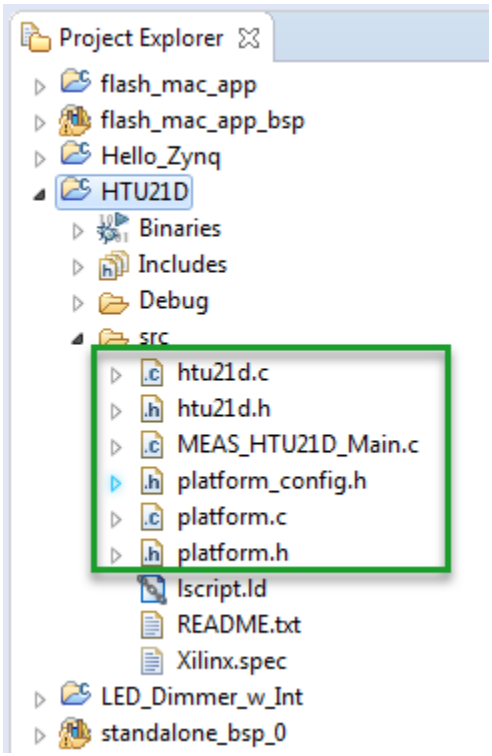

Figure 18 – HTU21D Project Explorer Window

a. MEAS_HTU21D_Main.c – Contains all the menu's and calls for user input

b. htu21d.h – Defines all the HTU21D Pmod registers

c. htu21d.c – Contains the high level operations of data acquisition and also defines the characteristics of the HTU21D Pmod and the AXI I2C controller

d. Platform_config.h - This is where the UART device is being defined as well as determine where the programs output data is being written

e. Platform.c - Hardware interface definition used by the Utilities functions to access the Processing System Hardware.

f. Platform.h – Header file for Processing System Hardware Interface

3. Now explore each file and try to understand how the I2C communication works between the PL I2C Controller and your HTU21D Pmod. Each file of interest is commented to help facilitate your understanding of what is actually happening.

   Start by looking through the htu21d.h file, in there you will notice all the HTU21D I2C registers are defined. Next move onto the htu21d.c file, here is where all the registers read and written to in order to enable the various functions. As you explore the application in TeraTerm, try and match up the various functions in htu21d.c to what you are doing. Now finally take a look in the MEAS_HTU21D_Main.c file in which the TeraTerm menu is described along with all the calls to the various functions we just looked at.

Question:

Answer the following question:

- *What peripheral does the iic_v3_4 driver support? (Hint : Refer to the MDD file)*

_____

Revision History

Date	Version	Revision
10 Jul 2017	1.0	Initial Avnet release Vivado 2017.1
01 Feb 2018	2.0	Updated to Vivado/SDK 2017.4

Answer

Experiment 2

- *What peripheral does the iic_v3_4 driver support? (Hint : Refer to the MDD file)*

Looking in the MDD file located at C:\Xilinx\SDK\2017.4\data\embeddedsw\XilinxProcessorIPLib\drivers\iic_v3_4\data it states **OPTION supported_peripherals = (axi_iic);** indicating it supports the AXI IIC IP Block