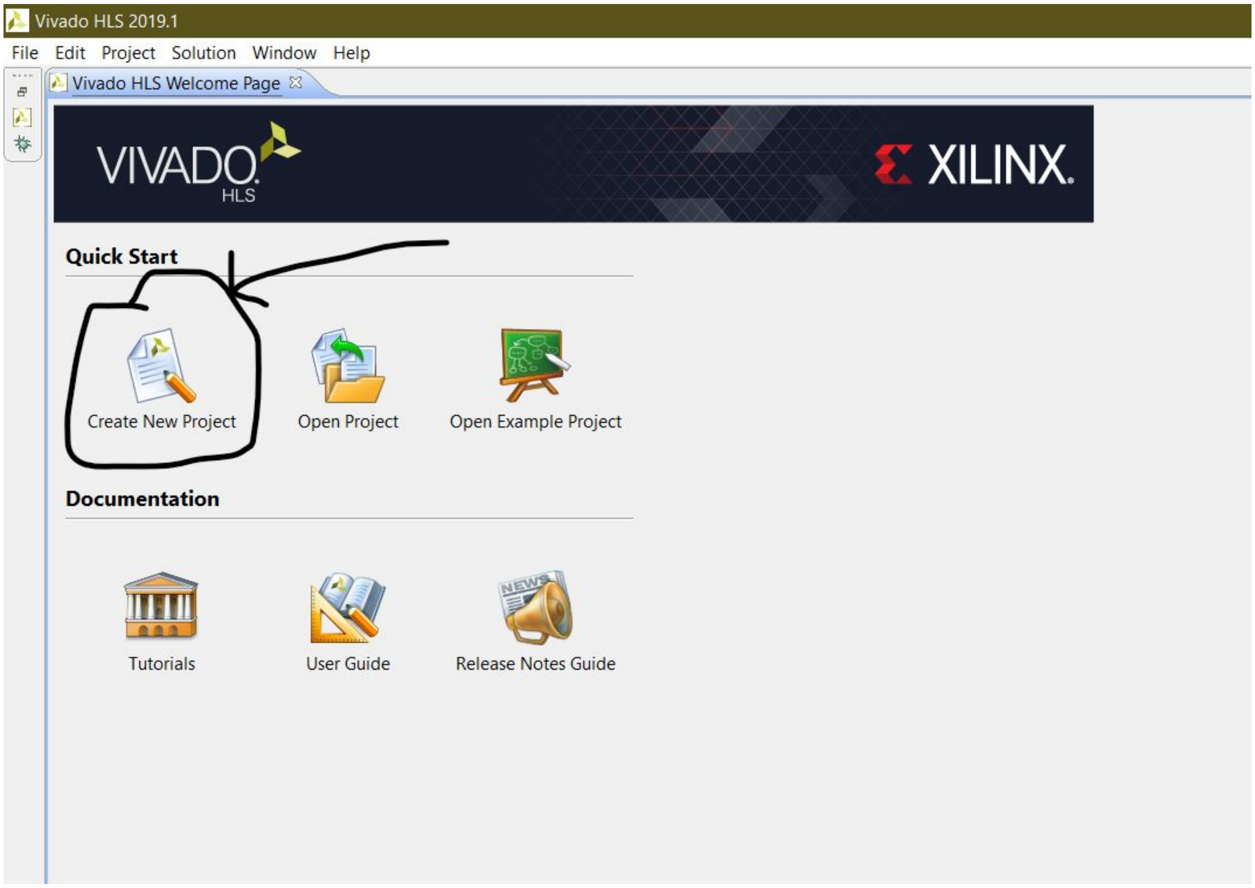
AES IMPLEMENTATION on FPGA

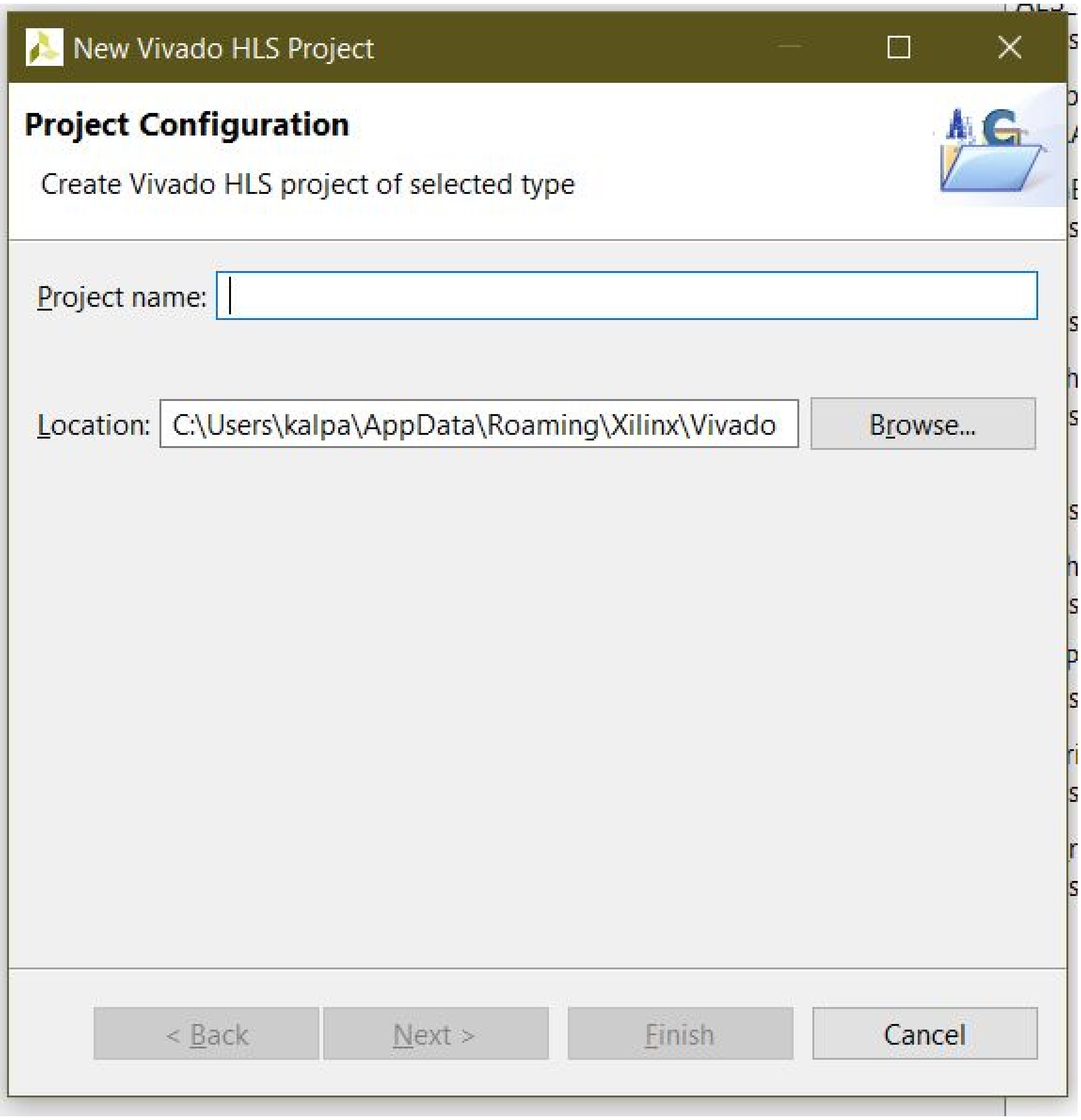
Kalpan Mehta - ksm469@nyu.edu

Step 1) Download the Tiny AES code from the given link

2) Open Vivado HLS (High-Level Synthesis) Tool and click on create new project.



3) Give the project name and location. Click Next after.



New Vivado HLS Project

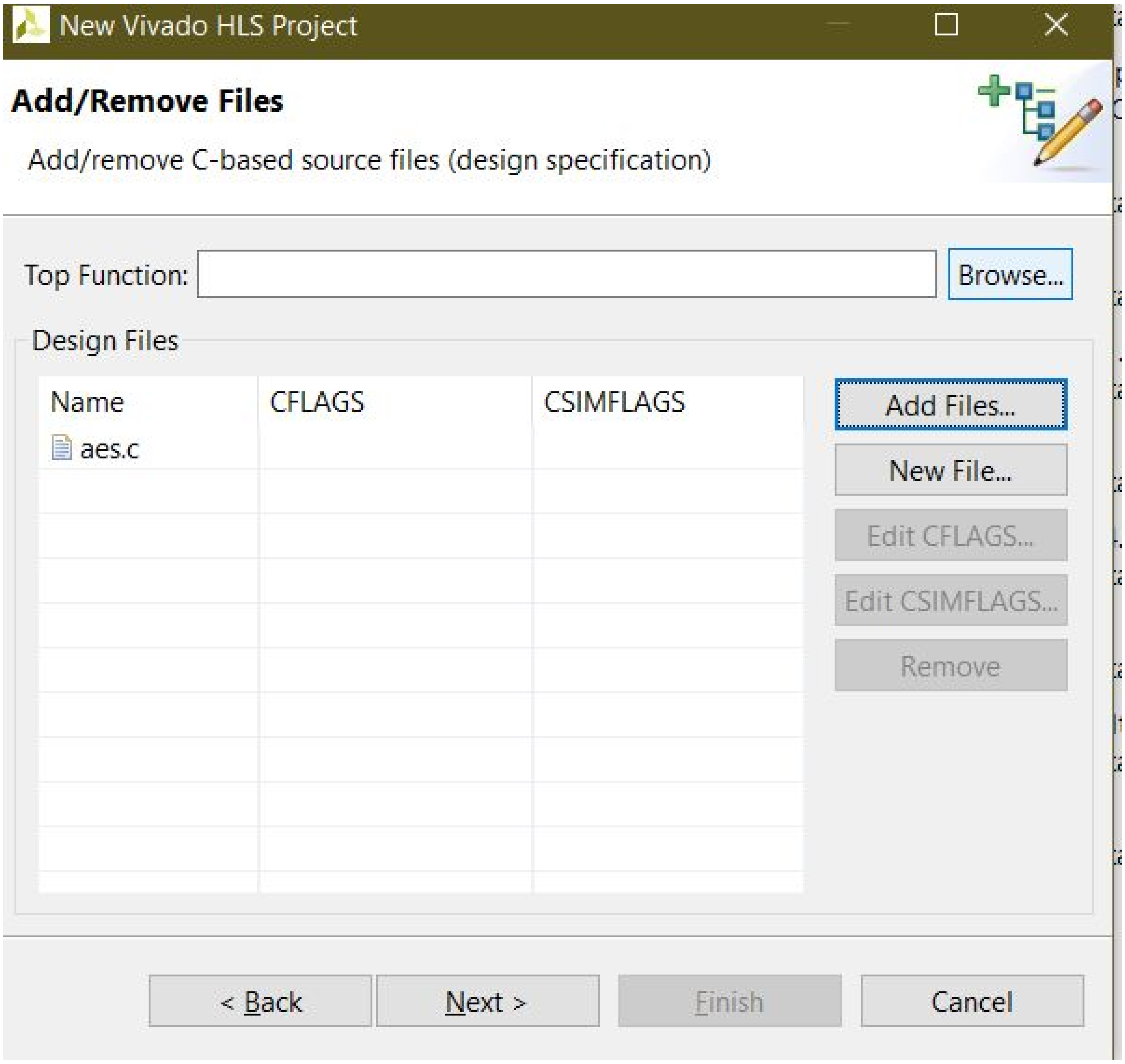
Project Configuration

Create Vivado HLS project of selected type

Project name:

Location: C:\Users\kalpa\AppData\Roaming\Xilinx\Vivado

4) In the Add/Remove C Source File click on the Add Files button and select the AES.c file.



Add/Remove Files

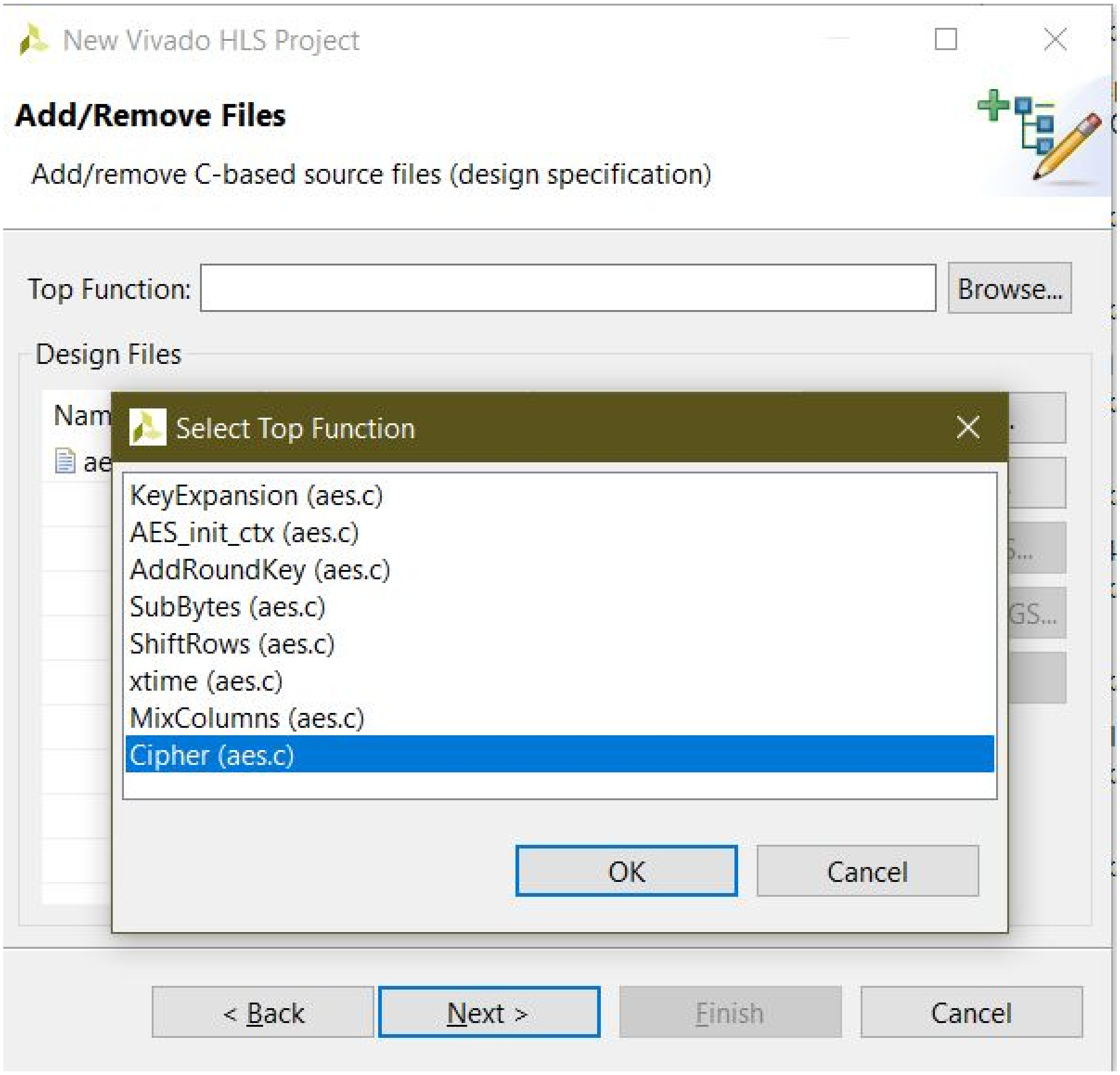
Add/remove C-based source files (design specification)

Top Function:

Design Files

Name	CFLAGS	CSIMFLAGS
aes.c		

5) Then click on Browse and select Cipher as the top function (Shown below). Click ok, Click Next.



New Vivado HLS Project

Add/Remove Files

Add/remove C-based source files (design specification)

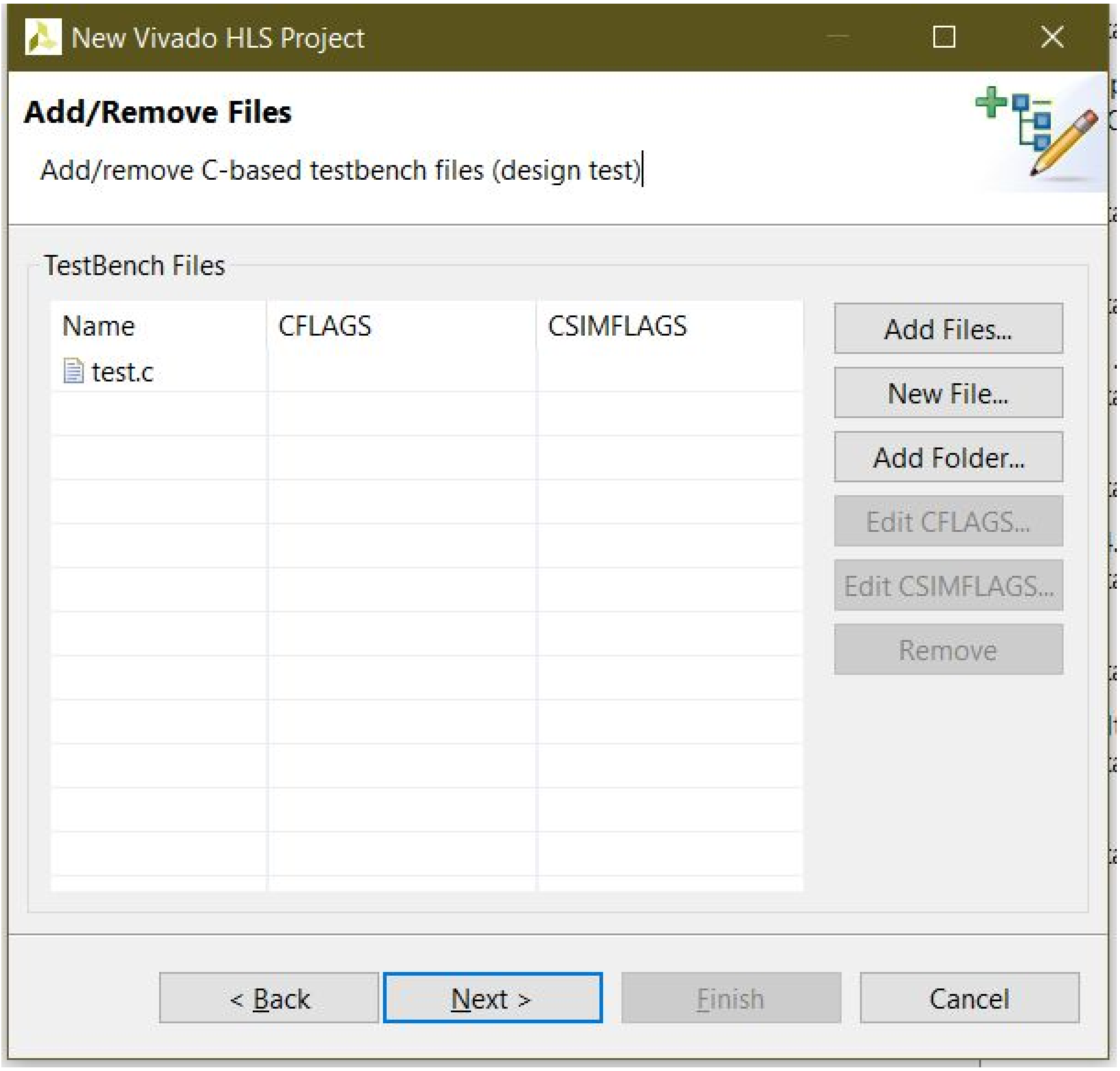
Top Function:

Design Files

Select Top Function

KeyExpansion (aes.c)
AES_init_ctx (aes.c)
AddRoundKey (aes.c)
SubBytes (aes.c)
ShiftRows (aes.c)
xtime (aes.c)
MixColumns (aes.c)
Cipher (aes.c)

6) Add the testbench file test.c . Click Next and then click on Finish. (You can



New Vivado HLS Project

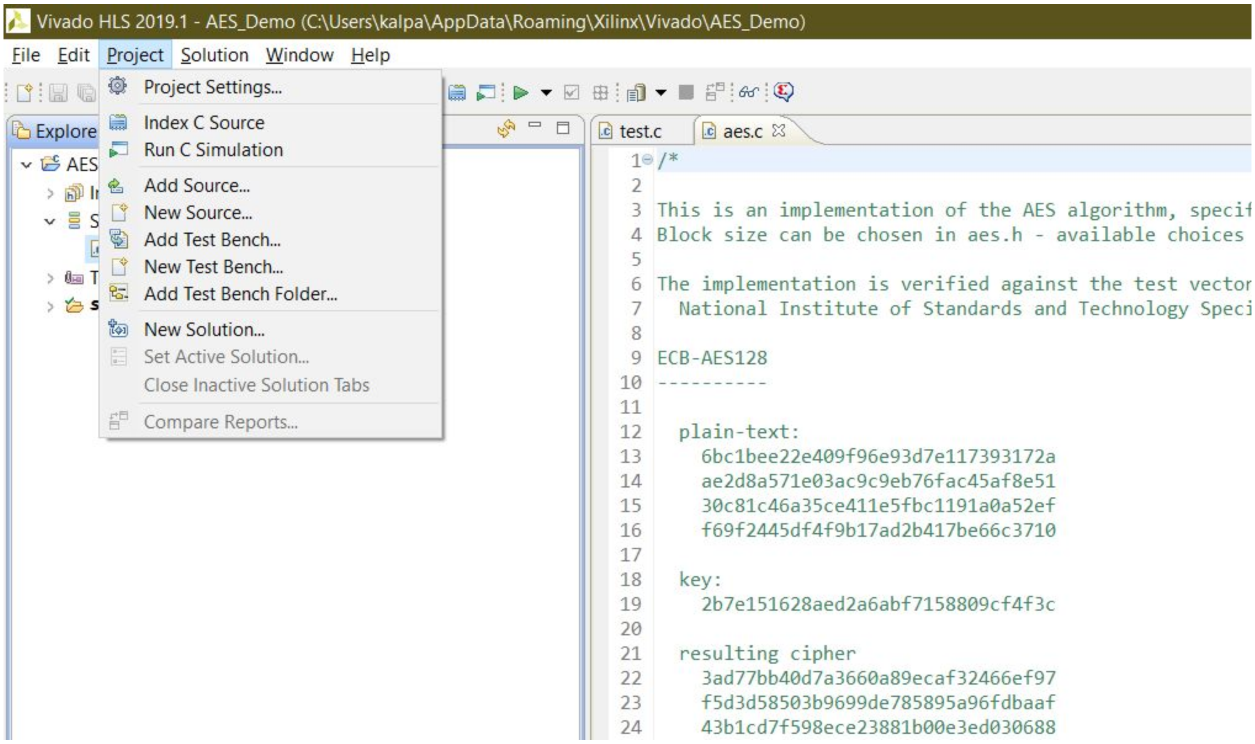
Add/Remove Files

Add/remove C-based testbench files (design test)

TestBench Files

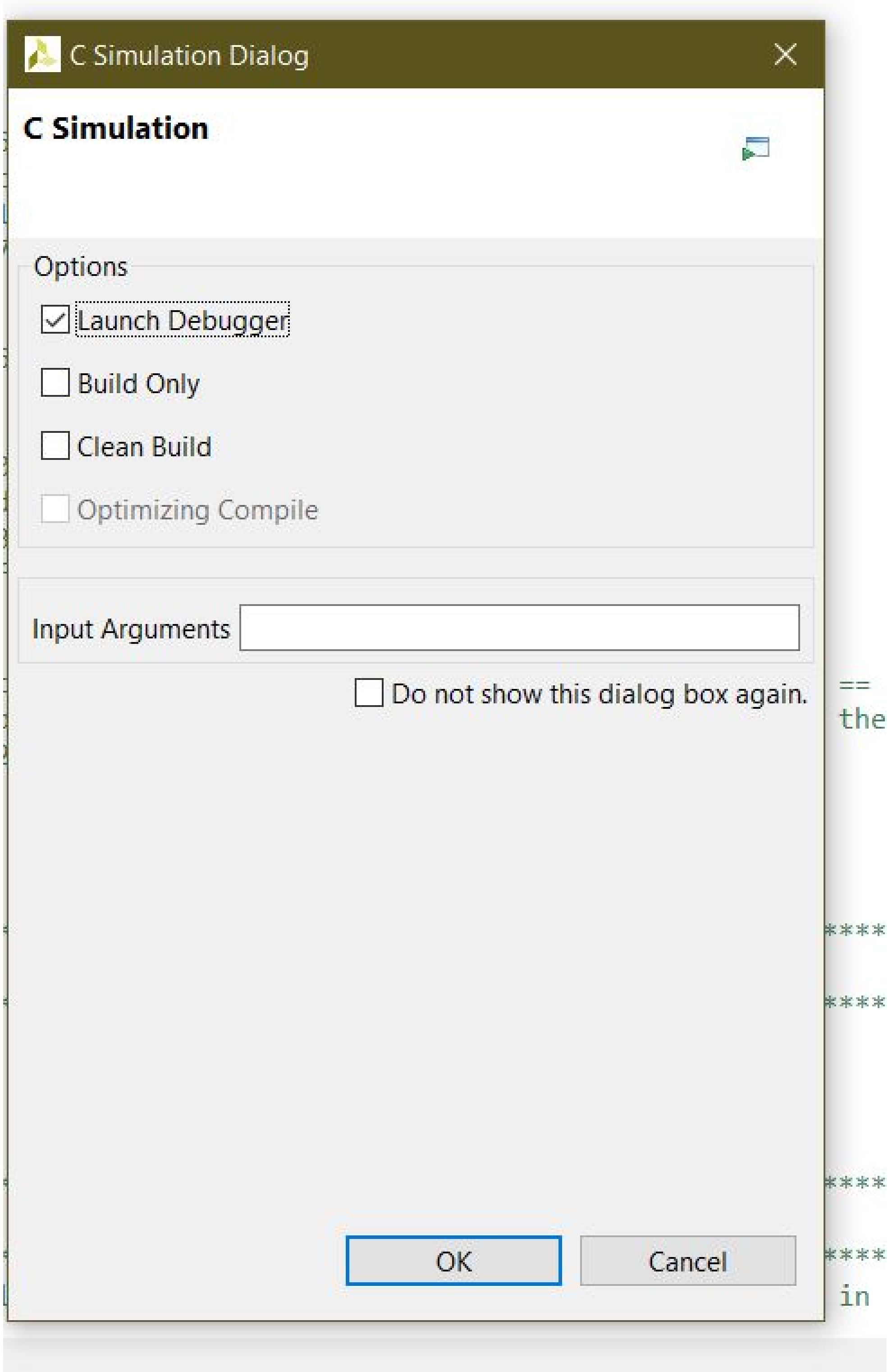
Name	CFLAGS	CSIMFLAGS
test.c		

7) Click on Project and then click on Run C Simulation.



```
/*

This is an implementation of the AES algorithm, specif
Block size can be chosen in aes.h - available choices

The implementation is verified against the test vector
  National Institute of Standards and Technology Speci

ECB-AES128
----------

  plain-text:
    6bc1bee22e409f96e93d7e117393172a
    ae2d8a571e03ac9c9eb76fac45af8e51
    30c81c46a35ce411e5fbc1191a0a52ef
    f69f2445df4f9b17ad2b417be66c3710

  key:
    2b7e151628aed2a6abf7158809cf4f3c

  resulting cipher
    3ad77bb40d7a3660a89ecaf32466ef97
    f5d3d58503b9699de785895a96fdbaaf
    43b1cd7f598ece23881b00e3ed030688
```

8) In the C Simulation dialog, make sure to check Launch Debugger. Then click ok.



Options

☑ Launch Debugger

□ Build Only

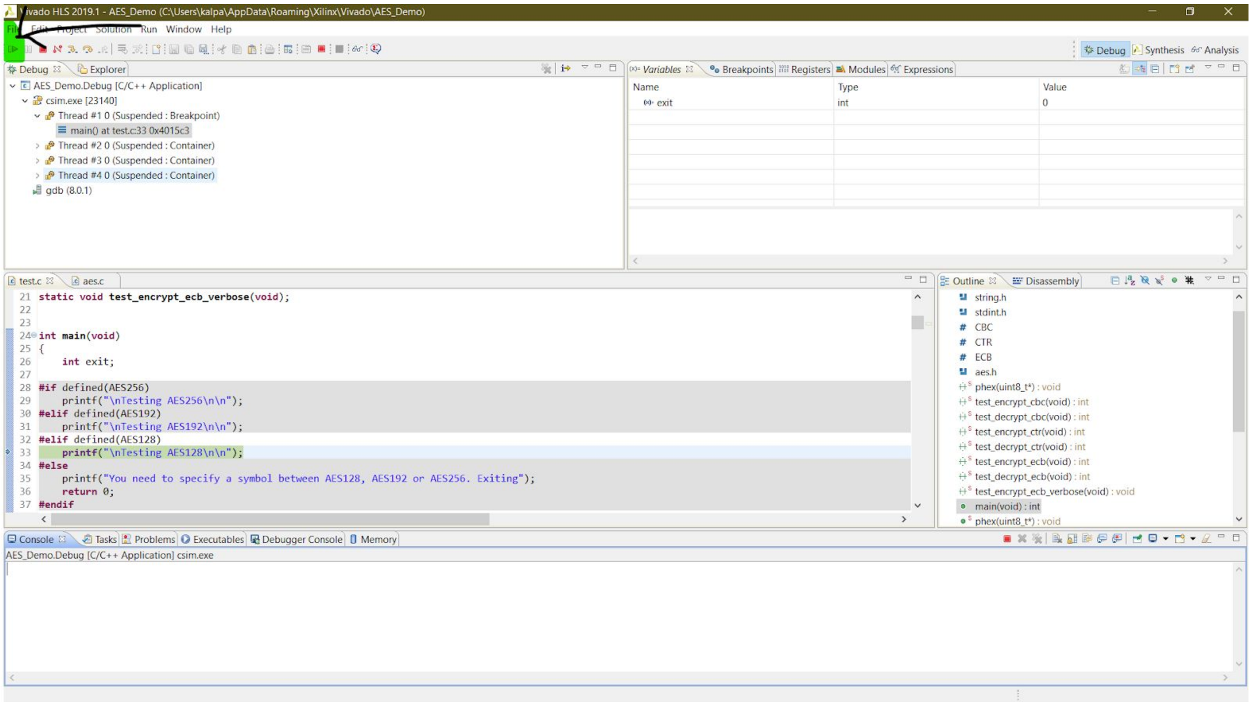
□ Clean Build

□ Optimizing Compile

Input Arguments

□ Do not show this dialog box again.

9) Once compiled, click on the run/resume button highlighted in green. You will see the output in the console at the bottom.



AES_Demo.Debug [C/C++ Application]
csim.exe [23140]
Thread #1 0 (Suspended : Breakpoint)
main() at test.c:33 0x4015c3
Thread #2 0 (Suspended : Container)
Thread #3 0 (Suspended : Container)
Thread #4 0 (Suspended : Container)
gdb (8.0.1)

Name	Type	Value
exit	int	0

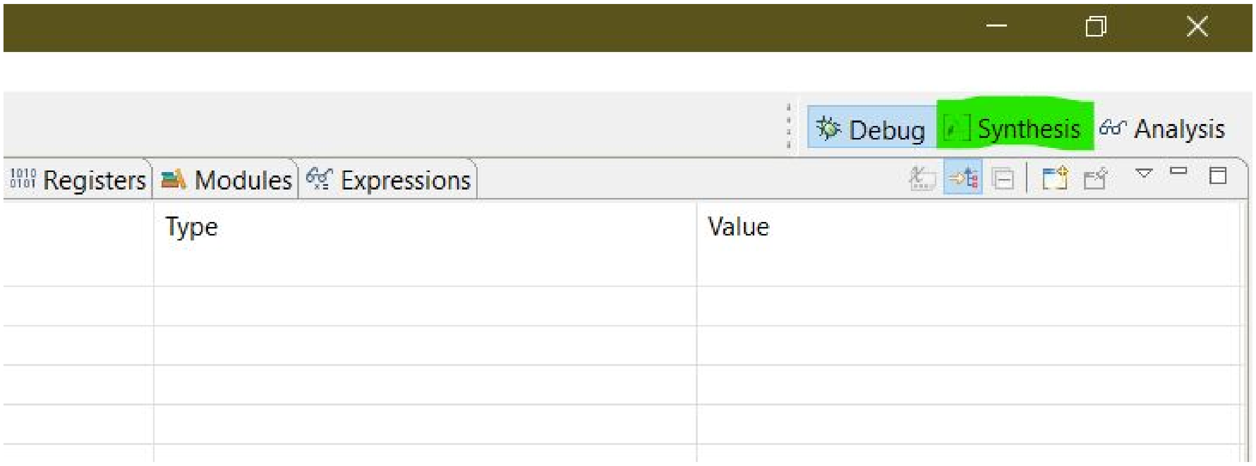
```
static void test_encrypt_ecb_verbose(void);


int main(void)
{
    int exit;

#if defined(AES256)
    printf("\nTesting AES256\n\n");
#elif defined(AES192)
    printf("\nTesting AES192\n\n");
#elif defined(AES128)
    printf("\nTesting AES128\n\n");
#else
    printf("You need to specify a symbol between AES128, AES192 or AES256. Exiting");
    return 0;
#endif
```

string.h
stdint.h
CBC
CTR
ECB
aes.h
phex(uint8_t*) : void
test_encrypt_cbc(void) : int
test_decrypt_cbc(void) : int
test_encrypt_ctr(void) : int
test_decrypt_ctr(void) : int
test_encrypt_ecb(void) : int
test_decrypt_ecb(void) : int
test_encrypt_ecb_verbose(void) : void
main(void) : int
phex(uint8_t*) : void

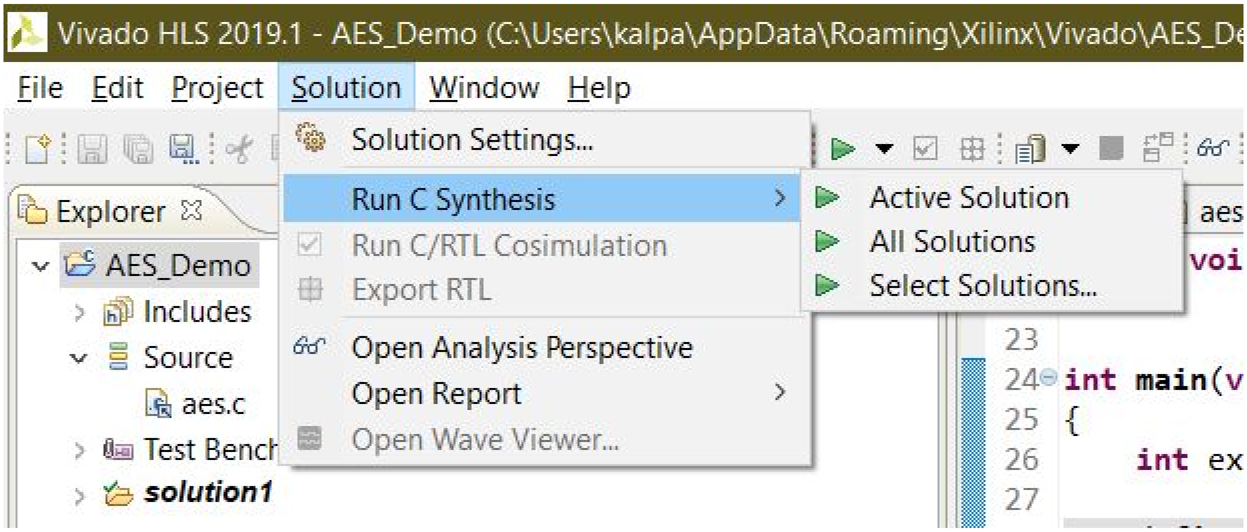
AES_Demo.Debug [C/C++ Application] csim.exe

If the console shows Failure or there is any error in compiling, make sure that aes.h file is included. If not, include it from the sources.

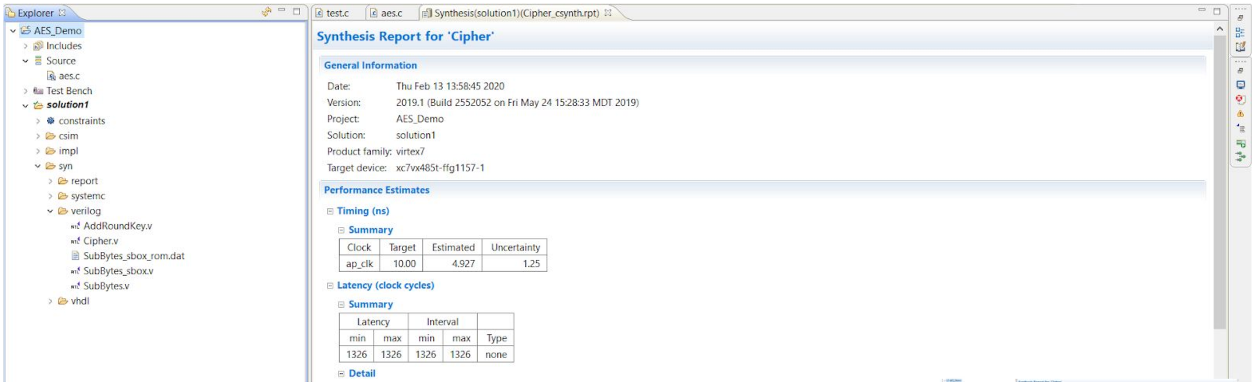
10) Click on the synthesis tab on the top right corner to get out from the debug mode.



11) Click on Solution > Run C Synthesis > Active Solution.



12) A synthesis report will be generated. Study the report to understand the latency and are requirements of your implementation.



Synthesis Report for 'Cipher'

General Information

Date: Thu Feb 13 13:58:45 2020
Version: 2019.1 (Build 2552052 on Fri May 24 15:28:33 MDT 2019)
Project: AES_Demo
Solution: solution1
Product family: virtex7
Target device: xc7vx485t-ffg1157-1

Performance Estimates

Timing (ns)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	4.927	1.25

Latency (clock cycles)

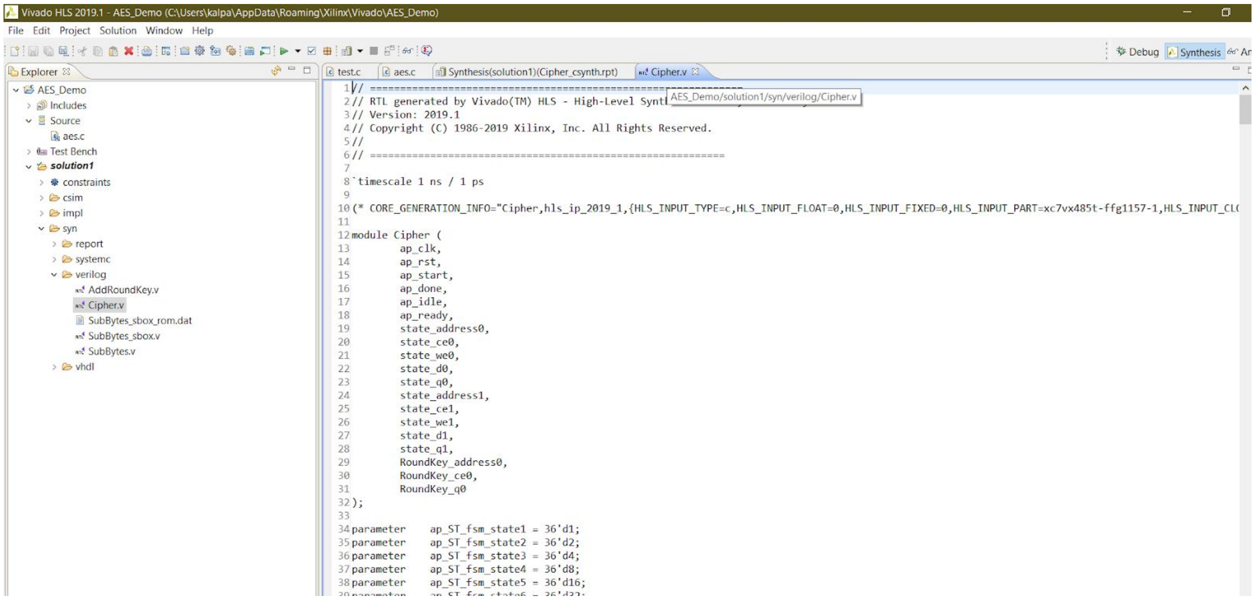
Summary

Latency		Interval		
min	max	min	max	Type
1326	1326	1326	1326	none

Detail

13) On the explorer window, notice that a folder named syn has been created. Expand it then expand the Verilog Folder (As shown on the left). There will be Verilog files created by the Vivado HLS tool.

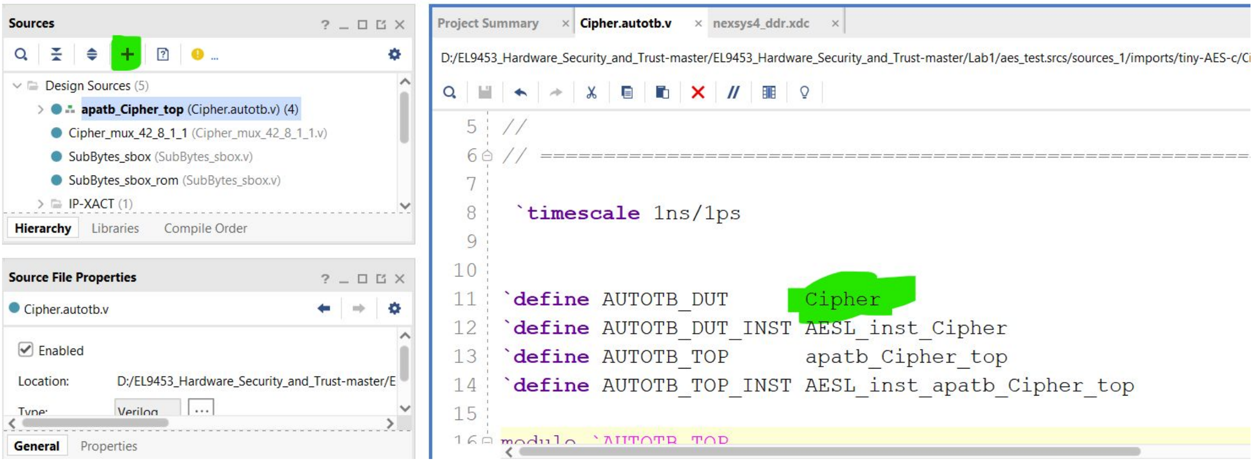
14) Double click on Cipher.v file. Take a quick look to understand how it is implemented and what are the inputs and the outputs. Hover over the title to know the location of this Verilog file. Shown below.


Vivado HLS 2019.1 - AES_Demo (C:\Users\kalpa\AppData\Roaming\Xilinx\Vivado\AES_Demo)
File Edit Project Solution Window Help
Debug
Synthesis
Explorer
AES_Demo
Includes
Source
aes.c
Test Bench
solution1
constraints
csim
impl
syn
report
systemc
verilog
AddRoundKey.v
Cipher.v
SubBytes_sbox_rom.dat
SubBytes_sbox.v
SubBytes.v
vhdl
test.c
aes.c
Synthesis(solution1)(Cipher_csynth.rpt)
Cipher.v
AES_Demo/solution1/syn/verilog/Cipher.v

```
// ============================================================
// RTL generated by Vivado(TM) HLS - High-Level Synt
// Version: 2019.1
// Copyright (C) 1986-2019 Xilinx, Inc. All Rights Reserved.
//
// ============================================================

`timescale 1 ns / 1 ps

(* CORE_GENERATION_INFO="Cipher,hls_ip_2019_1,{HLS_INPUT_TYPE=c,HLS_INPUT_FLOAT=0,HLS_INPUT_FIXED=0,HLS_INPUT_PART=xc7vx485t-ffg1157-1,HLS_INPUT_CL

module Cipher (
        ap_clk,
        ap_rst,
        ap_start,
        ap_done,
        ap_idle,
        ap_ready,
        state_address0,
        state_ce0,
        state_we0,
        state_d0,
        state_q0,
        state_address1,
        state_ce1,
        state_we1,
        state_d1,
        state_q1,
        RoundKey_address0,
        RoundKey_ce0,
        RoundKey_q0
);

parameter    ap_ST_fsm_state1 = 36'd1;
parameter    ap_ST_fsm_state2 = 36'd2;
parameter    ap_ST_fsm_state3 = 36'd4;
parameter    ap_ST_fsm_state4 = 36'd8;
parameter    ap_ST_fsm_state5 = 36'd16;
parameter    ap_ST_fsm_state6 = 36'd32;
```

15) Download the Lab 1 code from the Github link and open the project (.xpr file) in the Vivado.

16) open the cipher.autotb.v file and on Line 11 make sure that there is the Cipher module as DUT.



☑ Enabled

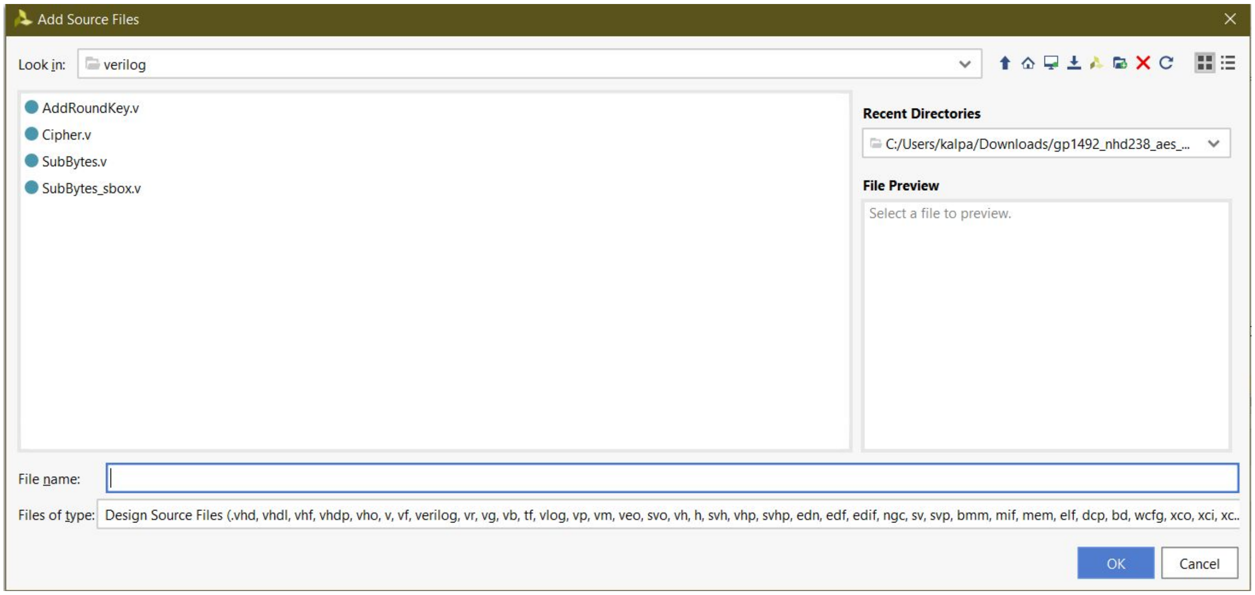
Location: D:/EL9453_Hardware_Security_and_Trust-master/E

Type: Verilog

D:/EL9453_Hardware_Security_and_Trust-master/EL9453_Hardware_Security_and_Trust-master/Lab1/aes_test.srcs/sources_1/imports/tiny-AES-c/Ci

```
 5  //
 6  // ============================================================
 7
 8   `timescale 1ns/1ps
 9
10
11  `define AUTOTB_DUT       Cipher
12  `define AUTOTB_DUT_INST  AESL_inst_Cipher
13  `define AUTOTB_TOP       apatb_Cipher_top
14  `define AUTOTB_TOP_INST  AESL_inst_apatb_Cipher_top
15
16  module `AUTOTB_TOP
```

17) Click on the + (Highlighted) to add the Verilog files. Click Add/Create design sources and click next.
18) Click on Add files and go to the location of the Cipher.v file you discovered on step 14.
19)Select all the files on that location. Click ok. Click Finish.



Look in: verilog

AddRoundKey.v
Cipher.v
SubBytes.v
SubBytes_sbox.v

C:/Users/kalpa/Downloads/gp1492_nhd238_aes_...

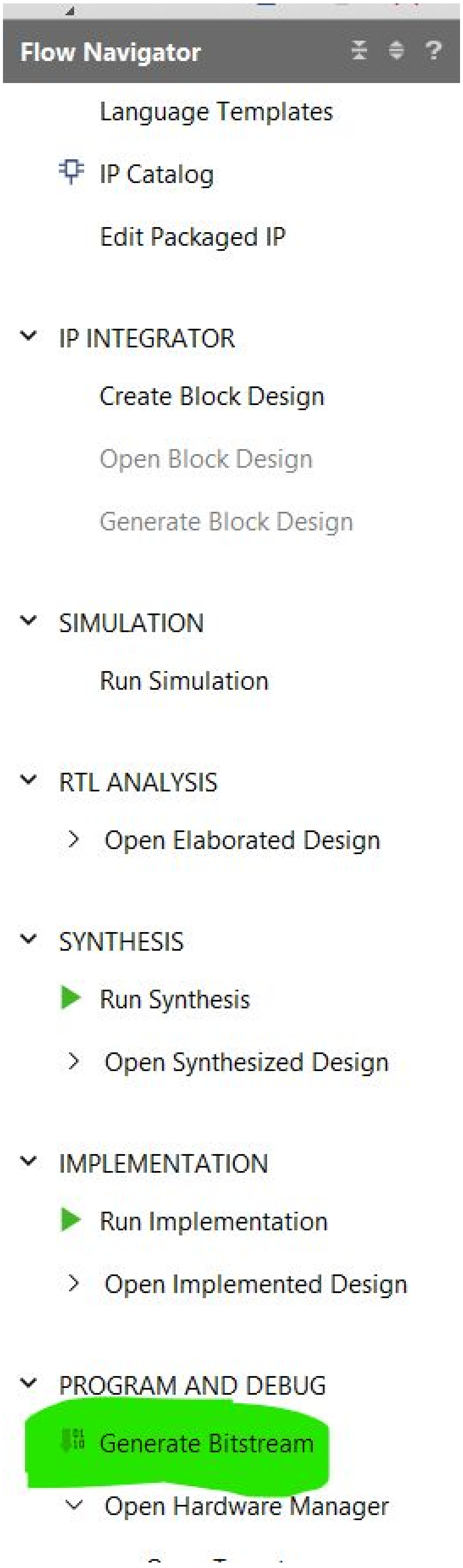
Select a file to preview.

File name:

Files of type: Design Source Files (.vhd, vhdl, vhf, vhdp, vho, v, vf, verilog, vr, vg, vb, tf, vlog, vp, vm, veo, svo, vh, h, svh, vhp, svhp, edn, edf, edif, ngc, sv, svp, bmm, mif, mem, elf, dcp, bd, wcfg, xco, xci, xc..

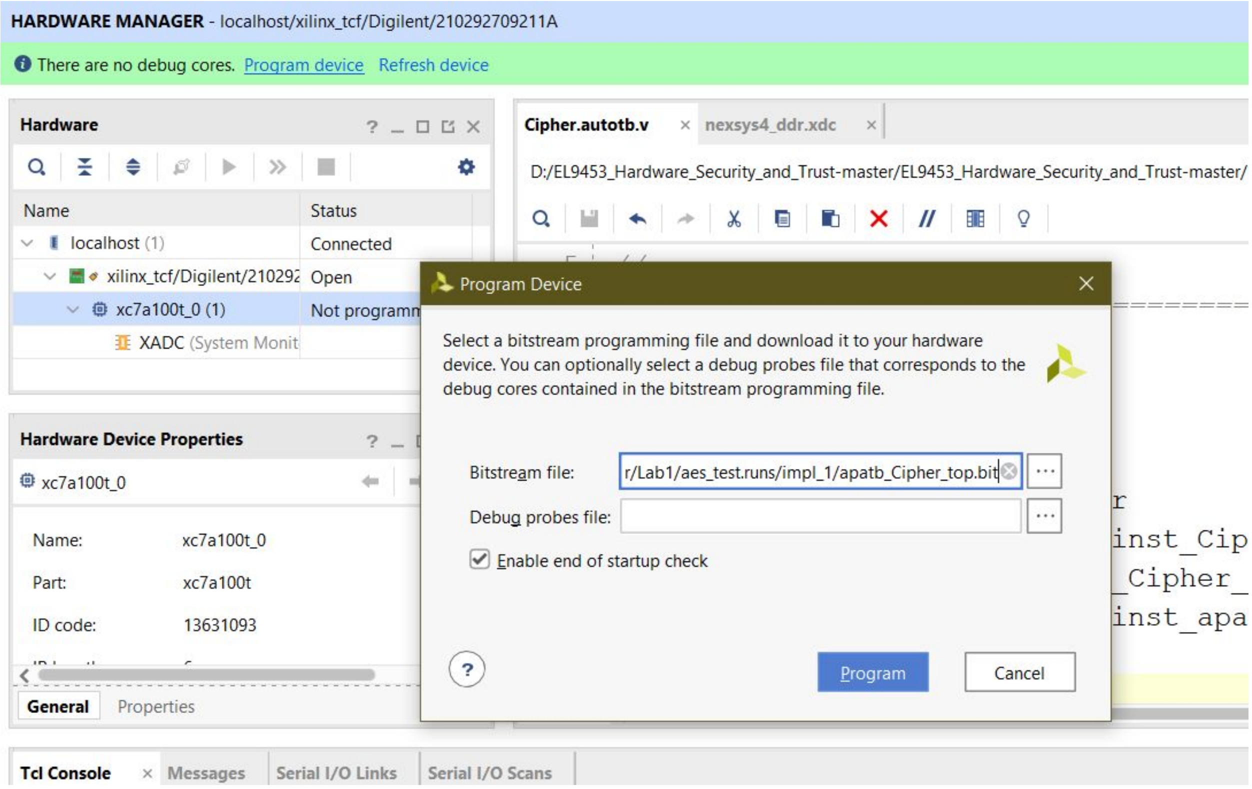
20) you will see that the 4 files have now been merged into the project.

21) Click on generate bitstream in the flow navigator.



Flow Navigator
Language Templates
IP Catalog
Edit Packaged IP
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager

22) if there are any errors at all, make sure you have selected the right Nexyx/Basys Board and added the constraint file as well.
23) Once Bitstream is generated, open hardware manager > open target > Autoconnect.
24) Connect your FPGA to your device.
25) once the hardware manager is open, click on program device. The .bit file generated should be already visible.



HARDWARE MANAGER - localhost/xilinx_tcf/Digilent/210292709211A

There are no debug cores. Program device Refresh device

Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/210292	Open
xc7a100t_0 (1)	Not programm
XADC (System Monit	

Hardware Device Properties

xc7a100t_0

Name: xc7a100t_0

Part: xc7a100t

ID code: 13631093

D:/EL9453_Hardware_Security_and_Trust-master/EL9453_Hardware_Security_and_Trust-master/

Program Device

Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.

Bitstream file: r/Lab1/aes_test.runs/impl_1/apatb_Cipher_top.bit

Debug probes file:

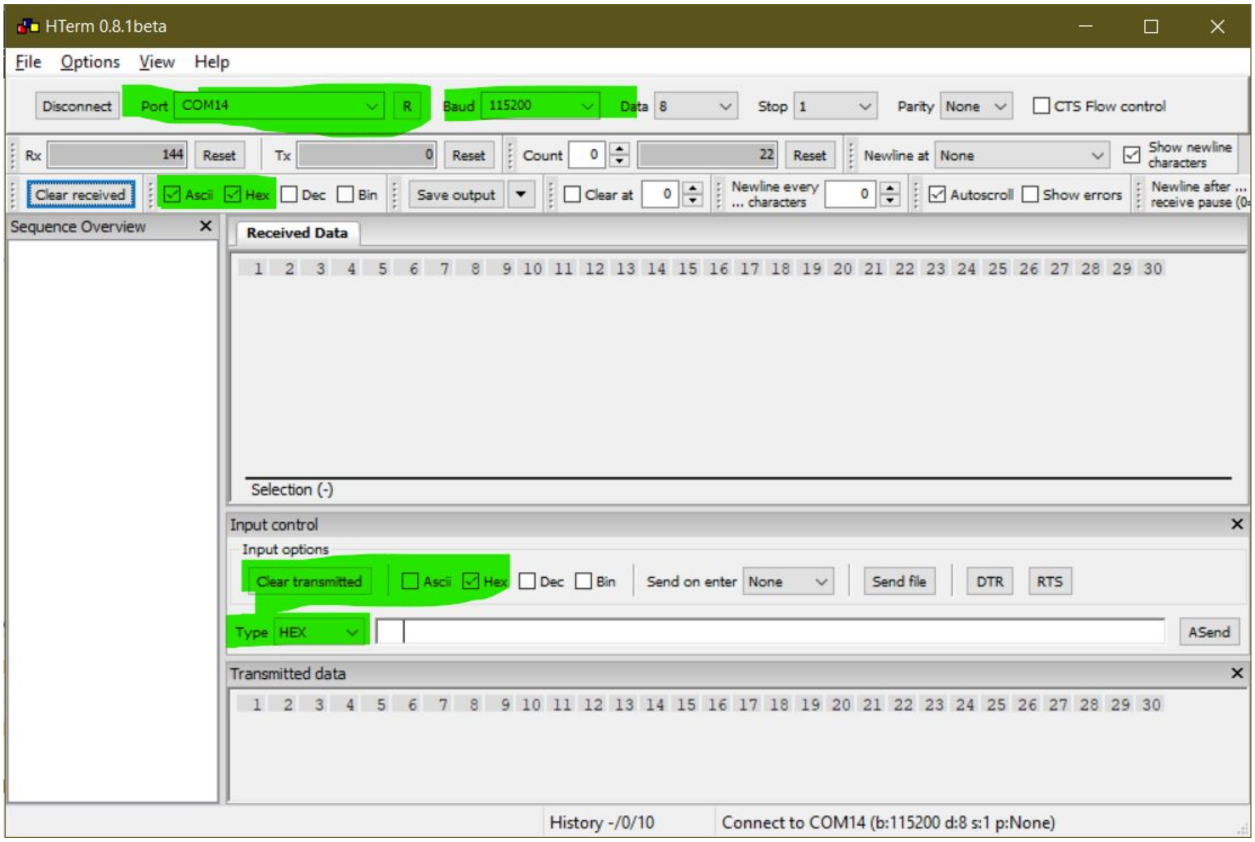
☑ Enable end of startup check

If not, click on (...) open project folder> .runs>impl_1. > search for .bit file. Select it and click program.
26) Your FPGA should show IDLE over its LED display.

27) Install any of the available tools available for serial communication. Make sure that the tool lets you communicate in hex. For windows, we recommend hterm.exe while for Linux we recommend moserial.
I have uploaded HTerm on the drive HERE.

28) Open HTerm/ Moserial and select the port to communicate to the FPGA. You can find which port your FPGA is connected to by opening the device manager> Ports.

29) For HTerm, make sure the following details match. Check whether Hex is ticked out, Baudrate is set correctly & Port is connected or not.



Port COM14 | Baud 115200 | Data 8 | Stop 1 | Parity None | □ CTS Flow control

Rx 144 | Tx 0 | Count 0 | 22 | Newline at None | ☑ Show newline characters

☑ Ascii ☑ Hex □ Dec □ Bin | □ Clear at 0 | Newline every ... characters 0 | ☑ Autoscroll □ Show errors | Newline after ... receive pause (0-

Input options: ☐ Ascii ☑ Hex ☐ Dec ☐ Bin | Send on enter None

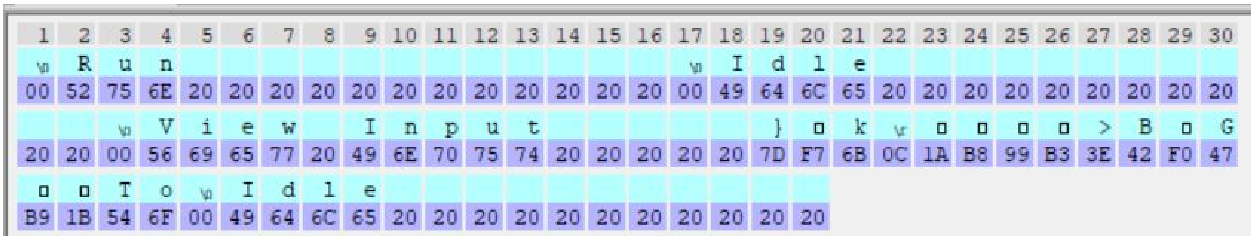
Type HEX

History -/0/10 | Connect to COM14 (b:115200 d:8 s:1 p:None)

30) Controls on your FPGA:

SW 15 and 14:Displays input and output
SW 13: Displays the key
SW 12: Run AES
SW 11 and 10: Enter Input
SW 9: Enter Key

31) Turn on SW 15 to see what the input is default input is 0. The default key is the given in the test.c file.



```
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
 \0  R  u  n                                      \0  I  d  l  e
00 52 75 6E 20 20 20 20 20 20 20 20 20 20 20 20 00 49 64 6C 65 20 20 20 20 20 20 20 20 20
       \0  V  i  e  w     I  n  p  u  t                 }  □  k \r  □  □  □  □  >  B  □  G
20 20 00 56 69 65 77 20 49 6E 70 75 74 20 20 20 20 20 7D F7 6B 0C 1A B8 99 B3 3E 42 F0 47
 □  □  T  o \0  I  d  l  e
B9 1B 54 6F 00 49 64 6C 65 20 20 20 20 20 20 20 20 20 20 20
```

32) press run (Turn on SW 12).

33) Display the output. The output starts with 7D F7 6B…..

34) You can verify your AES output at http://aes.online-domain-tools.com/

For any issues with the implementation Pl. email me on ksm469@nyu.edu