FPGA Lecture 1

Adam Duncan Ph.D.

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Class Introduction

About Me: Electrical/Computer Engineer at NSWC Crane

Class Goals: 1) Learn about FPGAs. 2) Program FPGAs to do fun things.

Class SW Materials: https://github.com/adamdunc/whs

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

This week we will do fun things with FPGAs!

- **Mon:**
  - Build basic logic circuits on breadboards
  - Build basic logic circuits on FPGA
  - Introduce FPGA design project
- **Tues/Wed/Thurs:**
  - Work on FPGA design project
- **Monday**
  - Project Brief Outs

Field-Programmable Gate Arrays (FPGAs) are “**Programmable Hardware**” devices

CPU

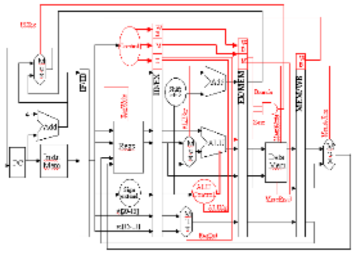
- ~1 inch^2 with ~10 billion transistors
- Fast (~3GHz)
- Hardware does one thing really well (execute code)

FPGA

- ~1 inch^2 with ~10 billion transistors
- Not as Fast (~500MHz)
- Hardware can be programmed to do anything

FPGA hardware behavior configured by “**Bitstream**” file

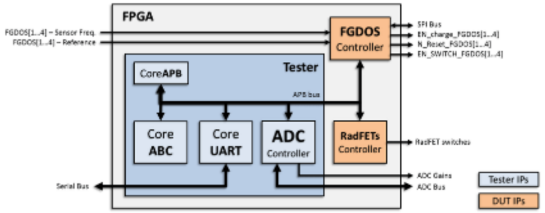
CPU



CPU is always CPU Datapath

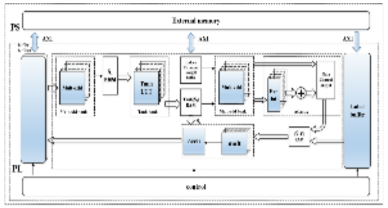
FPGA

Custom Sensor Bitstream



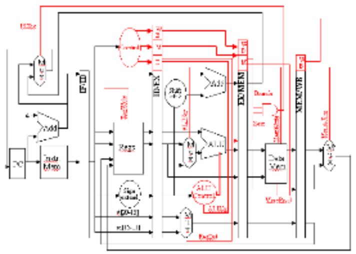
FPGA is now customized sensor driver/sampler

Neutral Network Bitstream



FPGA is now Neural Network Datapath

CPU Bitstream



FPGA is now CPU Datapath

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

FPGAs in the Wild

Ease of use:

- New HW designs in ~1 day
- Design portability

Cost:

- $1 for low-cost FPGAs
- $100K for rugged/large FPGAs

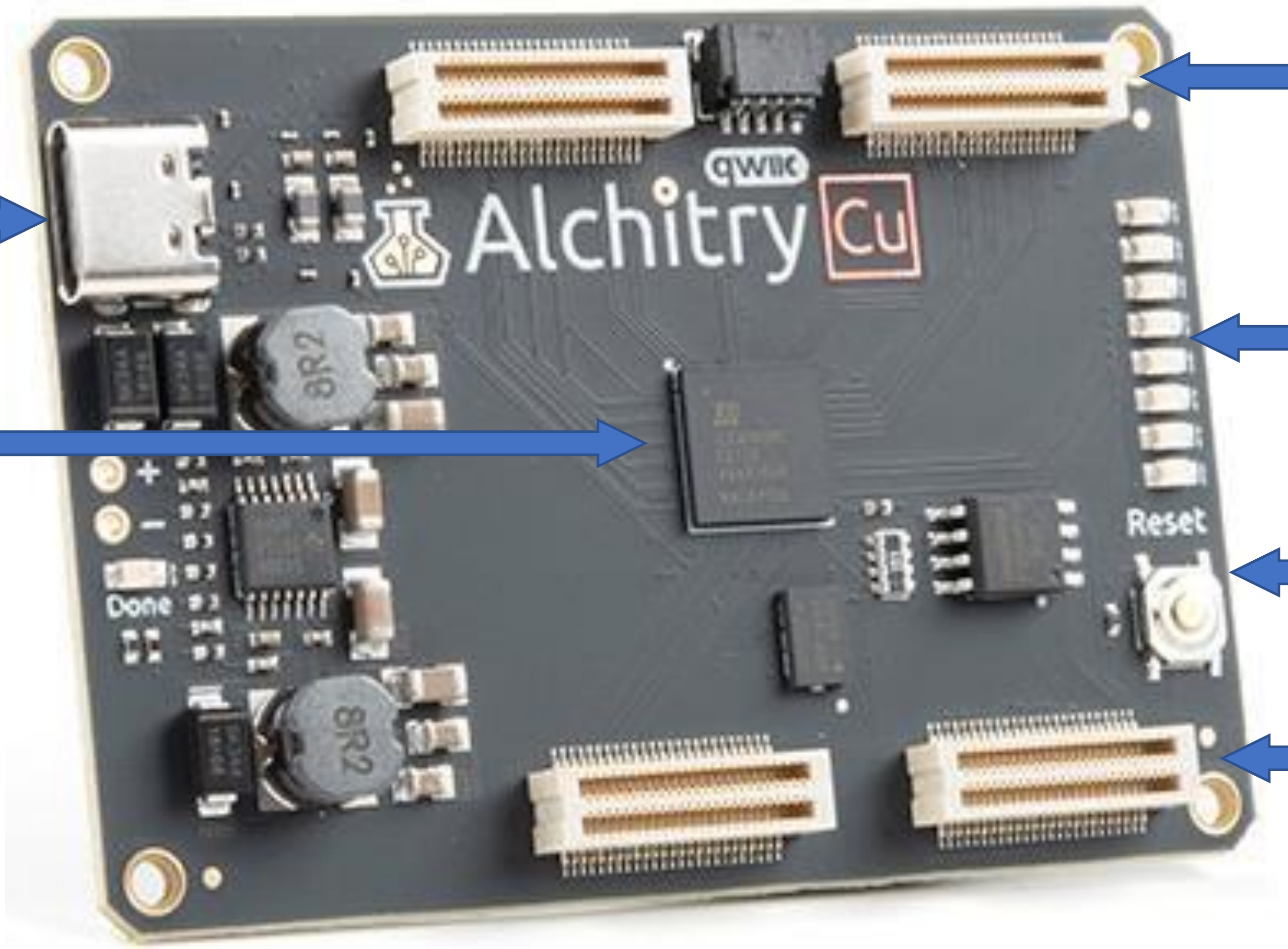
Products with FPGAs:

- Consumer electronics
- DoD Programs
- Automotive
- Networking equipment
- Test/Measurement equipment

FPGA Benefits for Department of Defense

- **HW ultimately drives performance** (SW optimizations limited by HW)
- **Parallelization** (i.e. 35 parallel ALUs executing at same time)
- **Unique needs** (read non-standard sensors at deterministic time intervals)
- **Consolidates PCB circuitry** (CPU + sensor logic + display driving)
- **Prototype Custom Integrated Circuits** (wafers >$10M)
- **Update hardware in field** (security updates, new features, bug fixes)
- **Replace obsolete hardware** (configure FPGA to "emulate" obsolete component"

Meet Our FPGA: Lattice ICE40 ($9 FPGA)



USB-C (Power)

$9 FPGA

Input/Output Pins

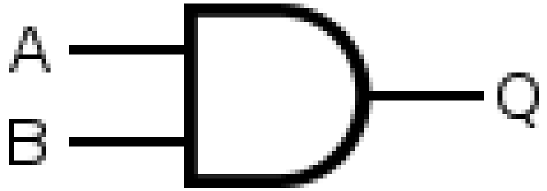
LEDs

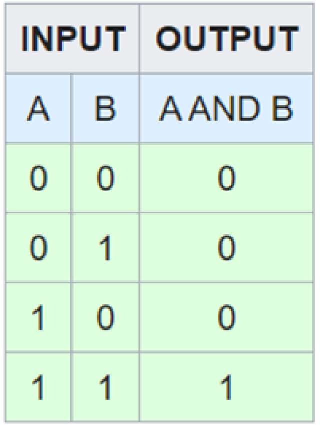
Reset Button

Input/Output Pins

https://alchitry.com/boards/cu

FPGAs use “logic gates” that implement Boolean equations





INPUT		OUTPUT
A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

“AND” logic gate

FPGAs can contain up to 20 million logic gates!

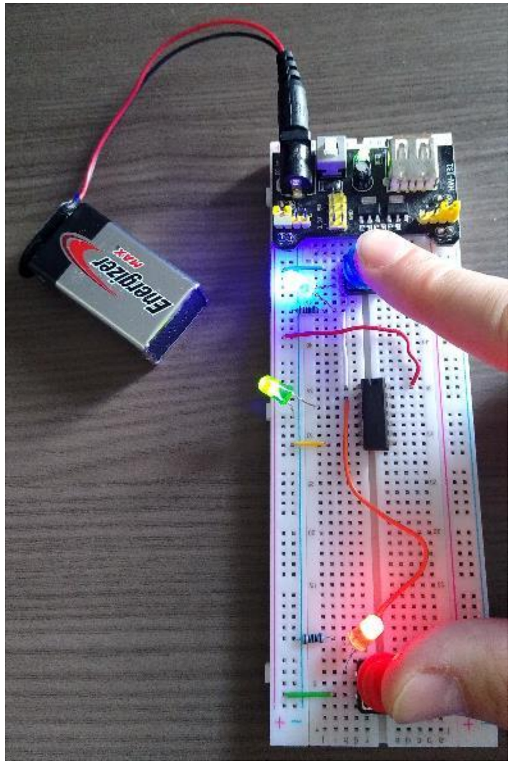
(Our FPGA has ~3000 logic gates)

https://en.wikipedia.org/wiki/Logic_gate

Project 1: AND gate with a breadboard

Input (Blue)	Input (Red)	Output (Green)
0	0	0
0	1	0
1	0	0
1	1	1

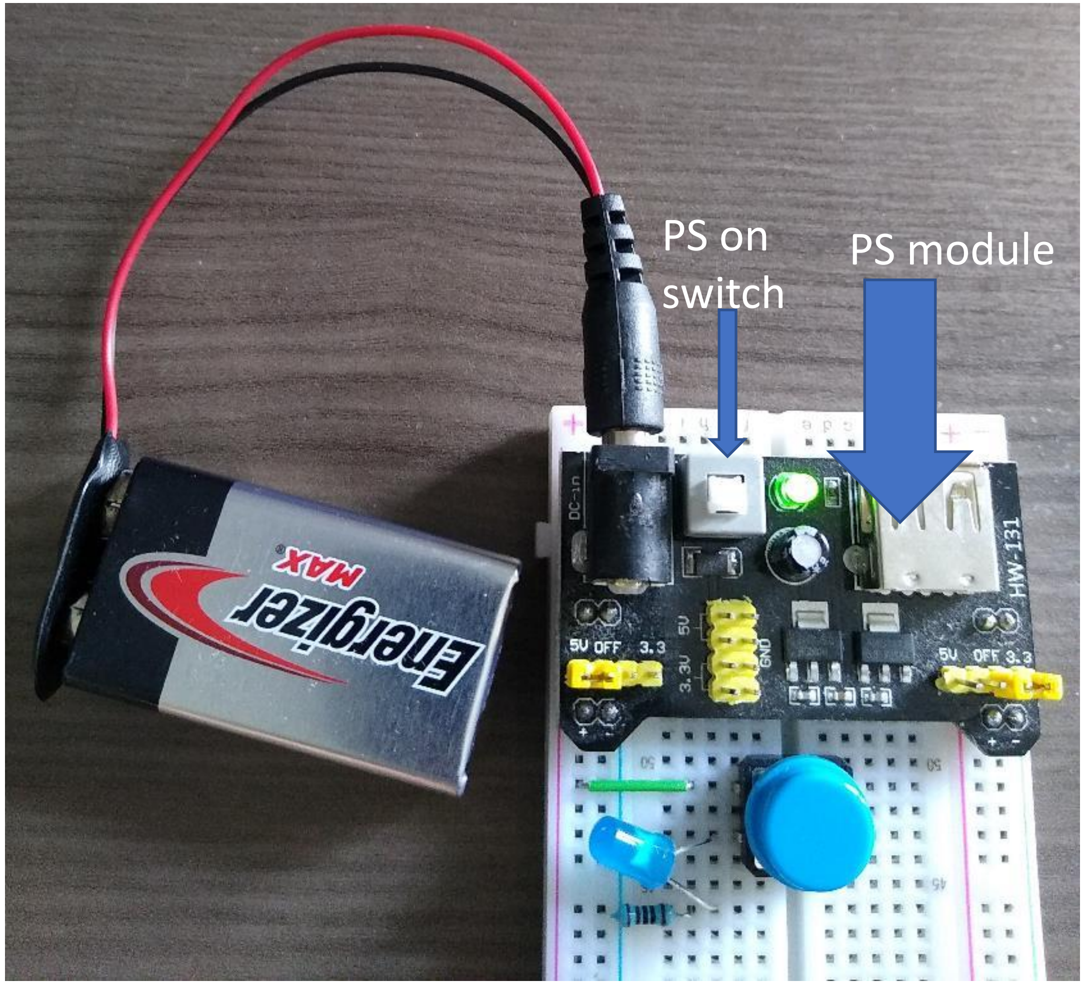
AND gate truth table



Blue LED lit if **blue button** pushed
Red Led lit if **red button** pushed
Green LED lit only if **blue button AND red button** pushed

Project 1 Step1: **Blue Button**

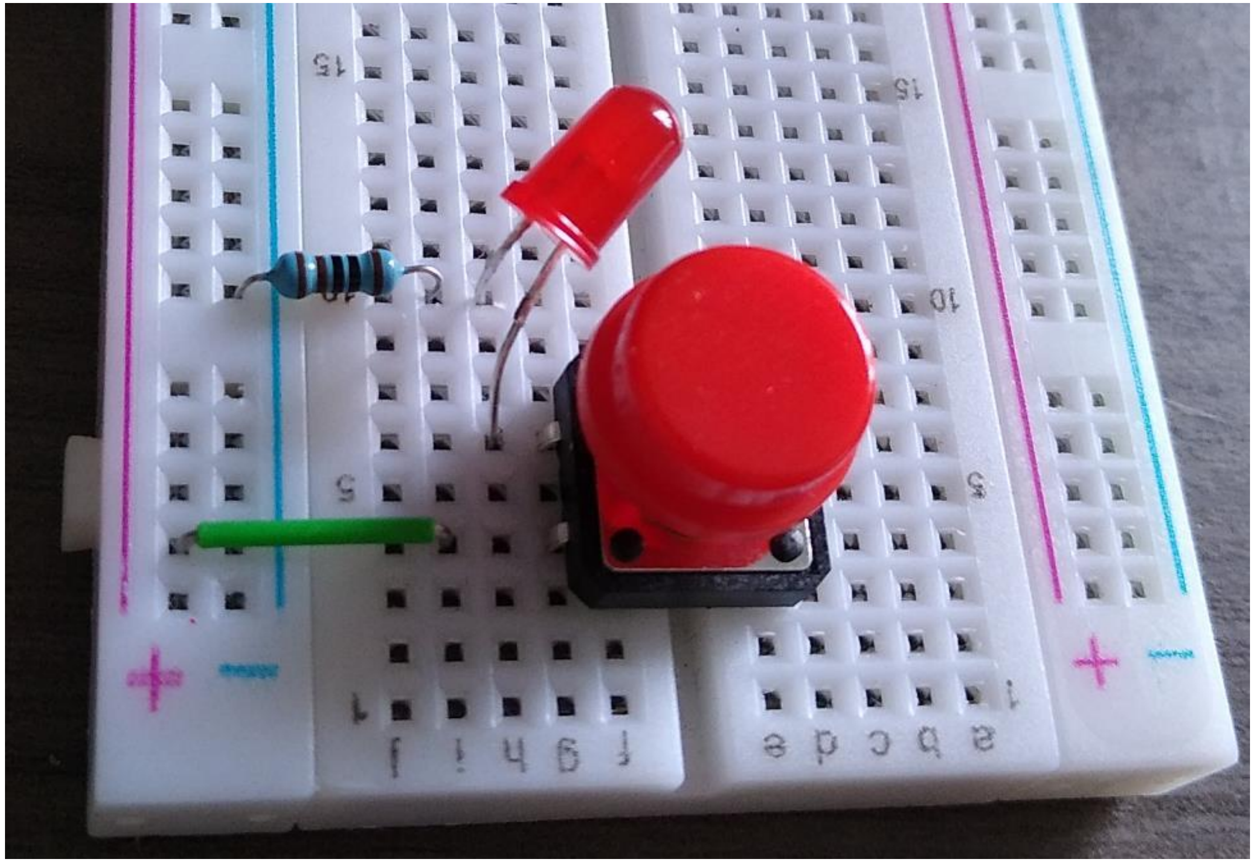
1. Connect PS module to breadboard
    1. Make sure PS “+ and –” line up with breadboard “+ and –”
2. Connect 9V adapter to PS module
3. Connect 9V to 9V adapter
4. Connect blue button just below PS module
5. Connect wire between “+” column and top button pin
6. Connect blue LED between bottom button pin and lower row on breadboard
    1. Make sure longer blue LED lead is the lead connected to top button pin
7. Connect 1000 ohm resistor from bottom LED lead to “-” column
8. Turn on the “PS on switch”
9. **Press the blue button and verify the blue LED is lit when pressed**



Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

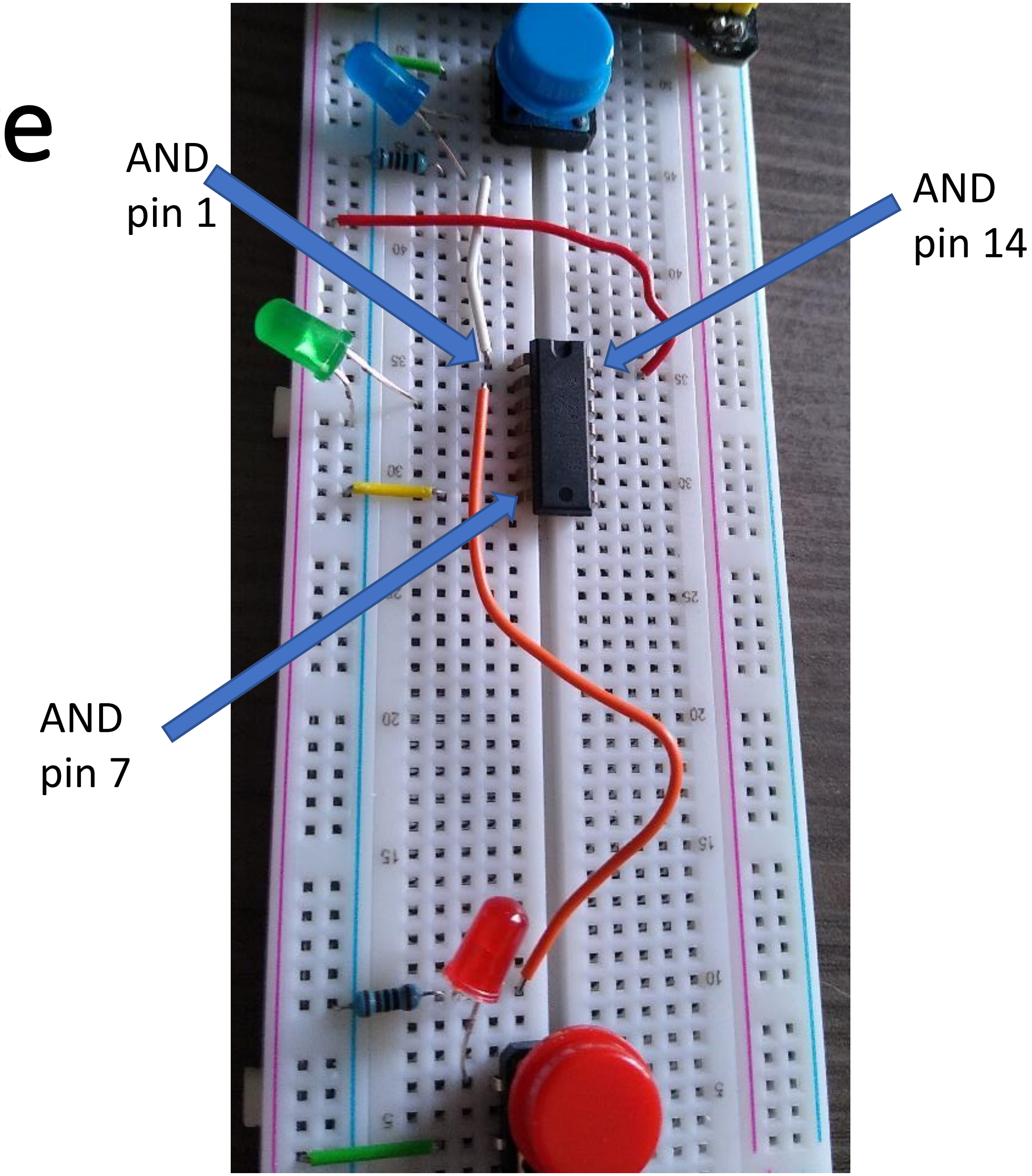
Project 1 Step2: **Red Button**

1. Connect red button near bottom of breadboard
2. Connect wire between "+" column and bottom button pin
3. Connect blue LED between bottom switch pin and lower row on breadboard
    1. Make sure longer red LED lead is the lead connected to bottom button pin
4. Connect 1000 ohm resistor from top LED lead to "-" column
5. Turn on the "PS on switch"
6. **Press the red switch and verify the red LED is lit when pressed**



Project 1 Step3: **AND Gate**

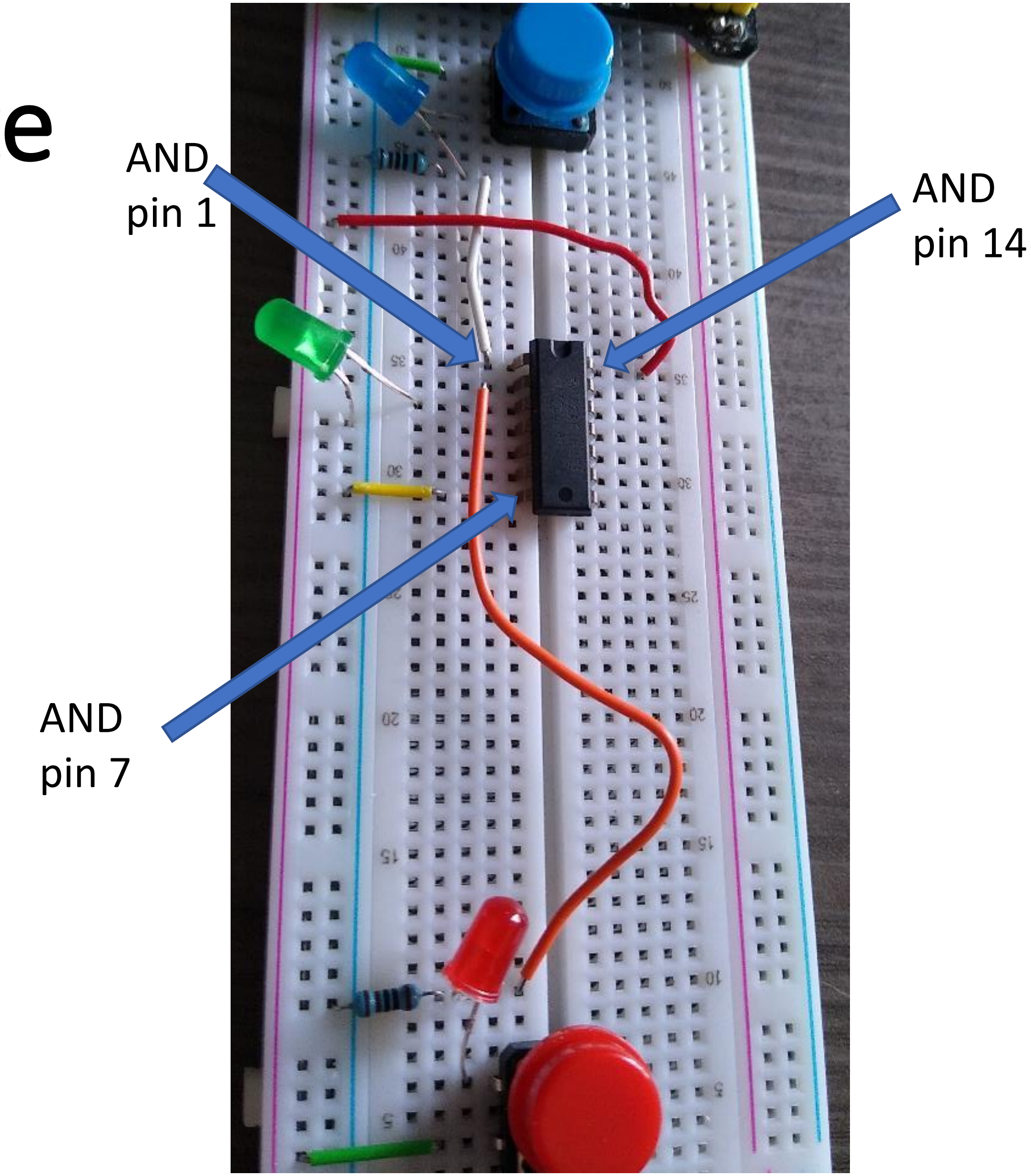
1. Place 14-pin AND Gate in middle of breadboard
2. Connect wire between AND pin 1 and bottom blue LED pin
3. Connect wire between AND pin 2 and top red LED pin
4. Connect wire between AND pin 7 and "-" column
5. Connect wire between AND pin 14 and "+" column
6. Connect LED from AND pin 3 to "-" column"
   1. Make sure longer LED pin is connected to AND pin 3
7. **Demonstrate circuit functions as AND**
   1. **i.e. Green LED only on if blue and red buttons pushed together**



Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Project 1 Step3: **AND Gate**

1. Place 14-pin AND Gate in middle of breadboard
2. Connect wire between AND pin 1 and bottom blue LED pin
3. Connect wire between AND pin 2 and top red LED pin
4. Connect wire between AND pin 7 and “-” column
5. Connect wire between AND pin 14 and “+” column
6. Connect LED from AND pin 3 to “-” column”
   1. Make sure longer LED pin is connected to AND pin 3
7. **Demonstrate circuit functions as AND**
   1. **i.e. Green LED only on if blue and red buttons pushed together**



AND pin 1

AND pin 14

AND pin 7

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Project 1 Step4: **other Gates**

1. Replace AND gate with either NAND, OR, XOR, XNOR
2. Write out the truth table below for the gate you chose.
3. How is it different than the AND Truth table?

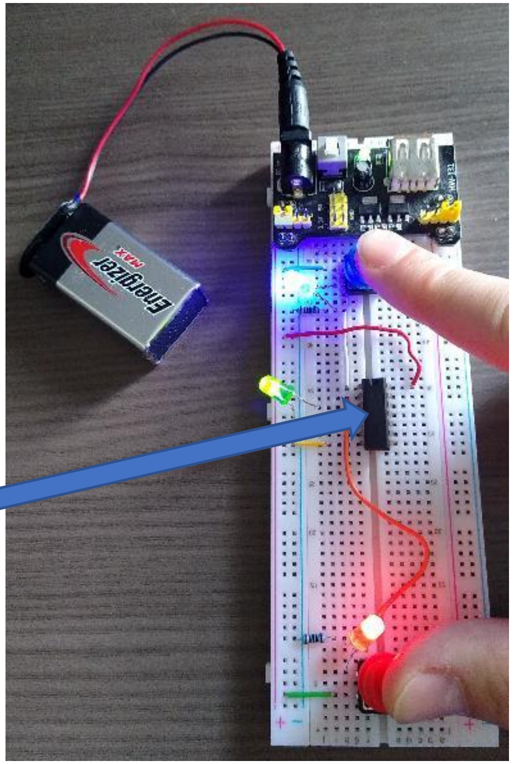
Input (Blue)	Input (Red)	Output (Green)
0	0	0
0	1	0
1	0	0
1	1	1

AND gate truth table

Input (Blue)	Input (Red)	Output (Green)
0	0	?
0	1	?
1	0	?
1	1	?

Truth table for your gate

Replace this with NAND, OR, XOR, XNOR



FPGA basics

Lattice **“iCEcube2”** is a software program used to design and compile code to program your FPGA in this class.

We write a code in a hardware description language (HDL) like **Verilog** to specify the intended hardware behavior of the FPGA.

iCEcube2 compiles the Verilog and automatically places logic gates inside the FPGA to realize the intended FPGA functionality.

iCEcube2 produces a **bitstream** file that is loaded into the physical FPGA to configure the FPGA behavior.

Lattice **“Diamond Programmer”** is a software program used to physically load a bitstream into your FPGA in this class

Verilog basics

A Verilog **module** is a hierarchal block in Verilog within input/output ports

An **assign** statement performs assignments and arithmetic operations on ports and wires within a module.

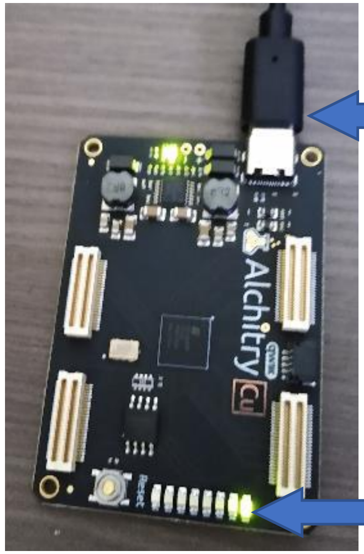
```
module top(
   output LED
   );
assign LED = 1;
endmodule
```

Arithmetic operators can be added to Verilog statements

```
module top(
   input button1,
   input button2,
   output LED
   );
assign LED = button1 & button2; // Logical AND arithmetic operator
endmodule
```

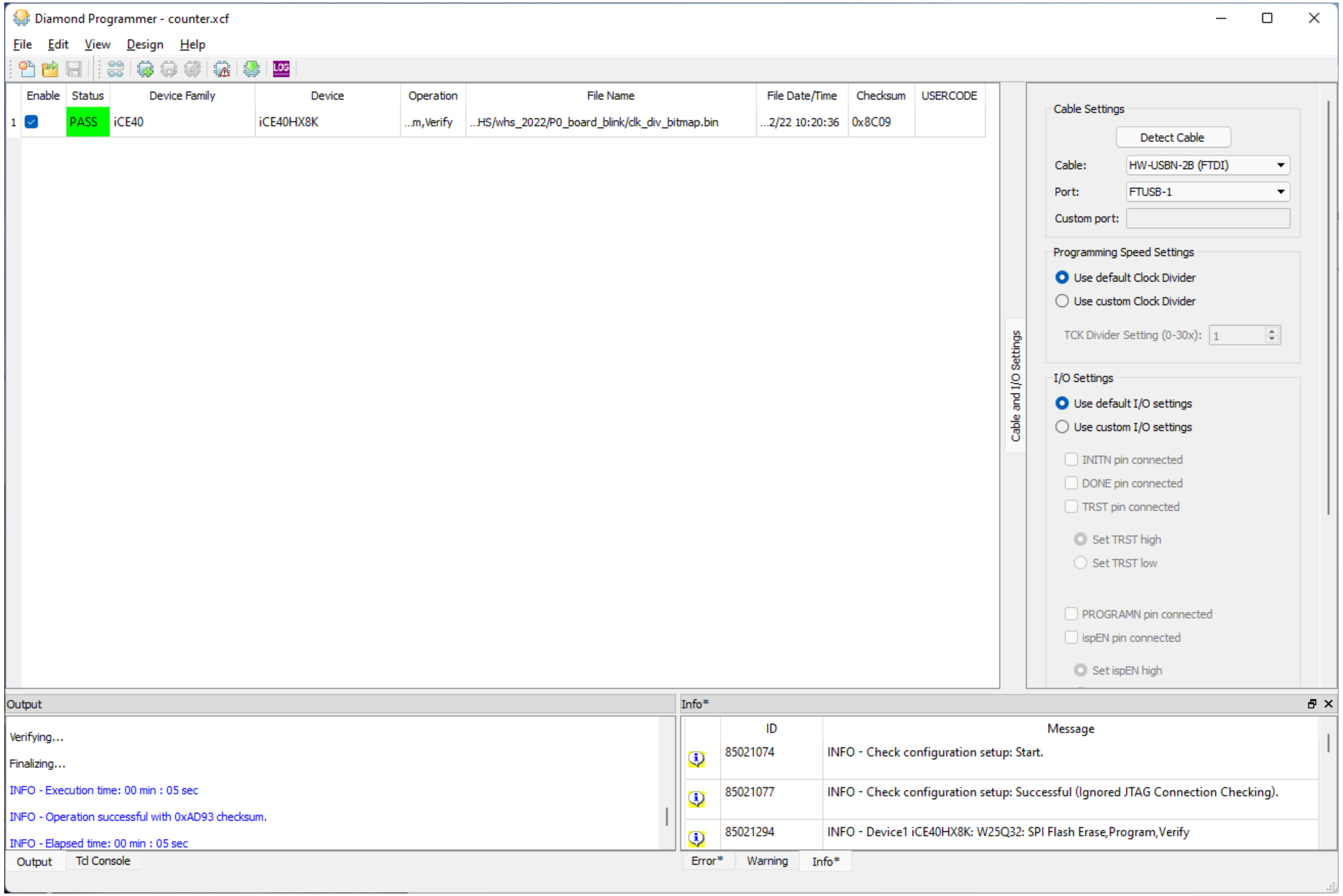
Project 2: Program the FPGA using Diamond Programmer

1. Connect Alchitry Cu to laptop USB-C
2. Open Diamond Programmer SW
3. Click “Open an existing project” or “File>Open File”
   1. P2_board_blink/Counter.xcf
4. Design>Program
5. Should see blue text appear in lower left and then green LED should blink



USB-C

LED that should blink



	Enable	Status	Device Family	Device	Operation	File Name	File Date/Time	Checksum	USERCODE
1	☑	PASS	iCE40	iCE40HX8K	...m,Verify	...HS/whs_2022/P0_board_blink/clk_div_bitmap.bin	...2/22 10:20:36	0x8C09	

Cable Settings

Cable: HW-USBN-2B (FTDI)

Port: FTUSB-1

Custom port:

Programming Speed Settings

● Use default Clock Divider

○ Use custom Clock Divider

TCK Divider Setting (0-30x): 1

I/O Settings

● Use default I/O settings

○ Use custom I/O settings

□ INITN pin connected

□ DONE pin connected

□ TRST pin connected

○ Set TRST high

○ Set TRST low

□ PROGRAMN pin connected

□ ispEN pin connected

○ Set ispEN high

Output

```
Verifying...
Finalizing...
INFO - Execution time: 00 min : 05 sec
INFO - Operation successful with 0xAD93 checksum.
INFO - Elapsed time: 00 min : 05 sec
```

Info*

ID	Message
85021074	INFO - Check configuration setup: Start.
85021077	INFO - Check configuration setup: Successful (Ignored JTAG Connection Checking).
85021294	INFO - Device1 iCE40HX8K: W25Q32: SPI Flash Erase,Program,Verify

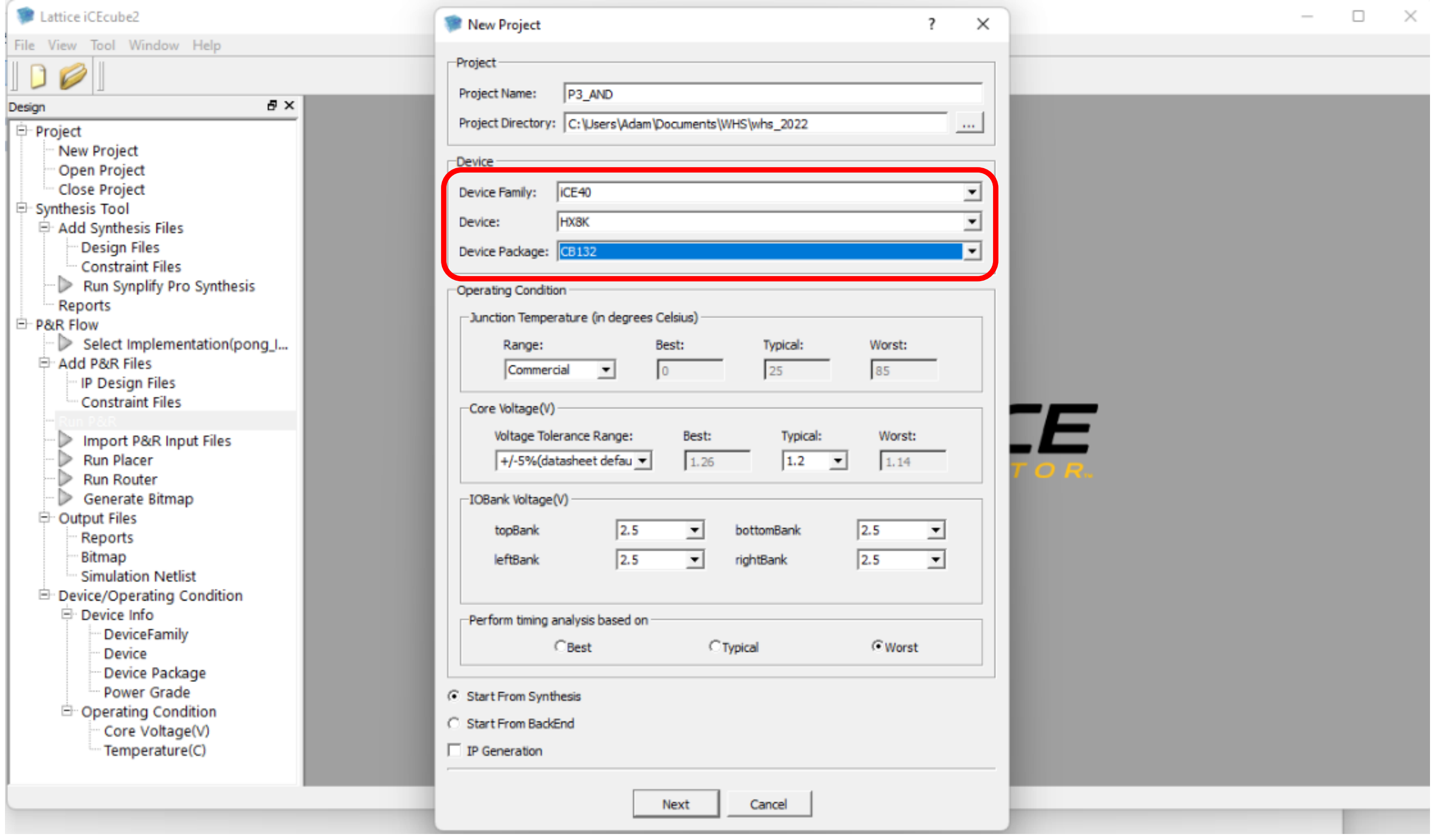
Distribution Statement A: Approved for Public Release; Distribution is unlimited.

Project 3: Program basic logic gate behavior into the FPGA

- **Goal: Use the FPGA to imitate the breadboard behavior**
  - Write Verilog code to tell the FPGA how to behave in hardware
  - Compile the Verilog code into a **“bitstream”** file
  - Load the bitstream into the FPGA
  - Connect some “buttons” to show the FPGA in action

Project 3 Step 1: Create AND project in ICEcube2

1. Open iCEcube2 SW
2. File>New Project
3. Type Project Name = P3_AND
4. Set Device Family = ICE40
5. Set Device = HX8K
6. Set Device Package = CB132
7. Click Next



New Project

Project

Project Name: P3_AND

Project Directory: C:\Users\Adam\Documents\WHS\whs_2022

Device

Device Family: iCE40

Device: HX8K

Device Package: CB132

Operating Condition

Junction Temperature (in degrees Celsius)

Range:	Best:	Typical:	Worst:
Commercial	0	25	85

Core Voltage(V)

Voltage Tolerance Range:	Best:	Typical:	Worst:
+/-5%(datasheet defau	1.26	1.2	1.14

IOBank Voltage(V)

topBank	2.5	bottomBank	2.5
leftBank	2.5	rightBank	2.5

Perform timing analysis based on

○ Best ○ Typical ● Worst

● Start From Synthesis

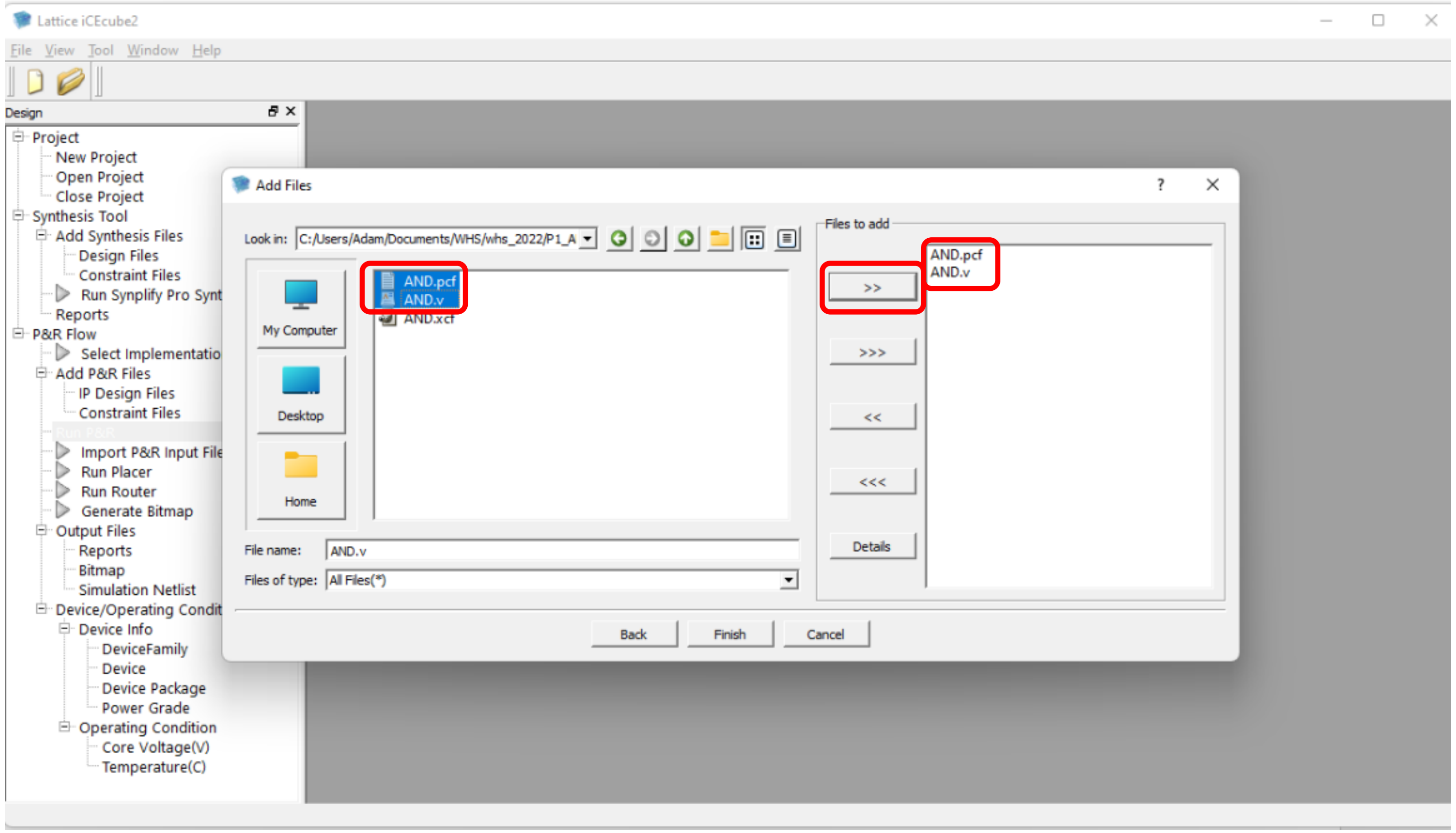
○ Start From BackEnd

□ IP Generation

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Project 3 Step 2: Create AND project in ICEcube2

1. Add Files should pop up
2. Select files in P3_AND:
    1. AND.pcf
    2. AND.v
3. Click the “>>” icon so they show up in the “Files to Add”
4. Click Finish



Look in: C:/Users/Adam/Documents/WHS/whs_2022/P1_A

AND.pcf
AND.v
AND.xcf

File name: AND.v

Files of type: All Files(*)

Files to add

AND.pcf
AND.v

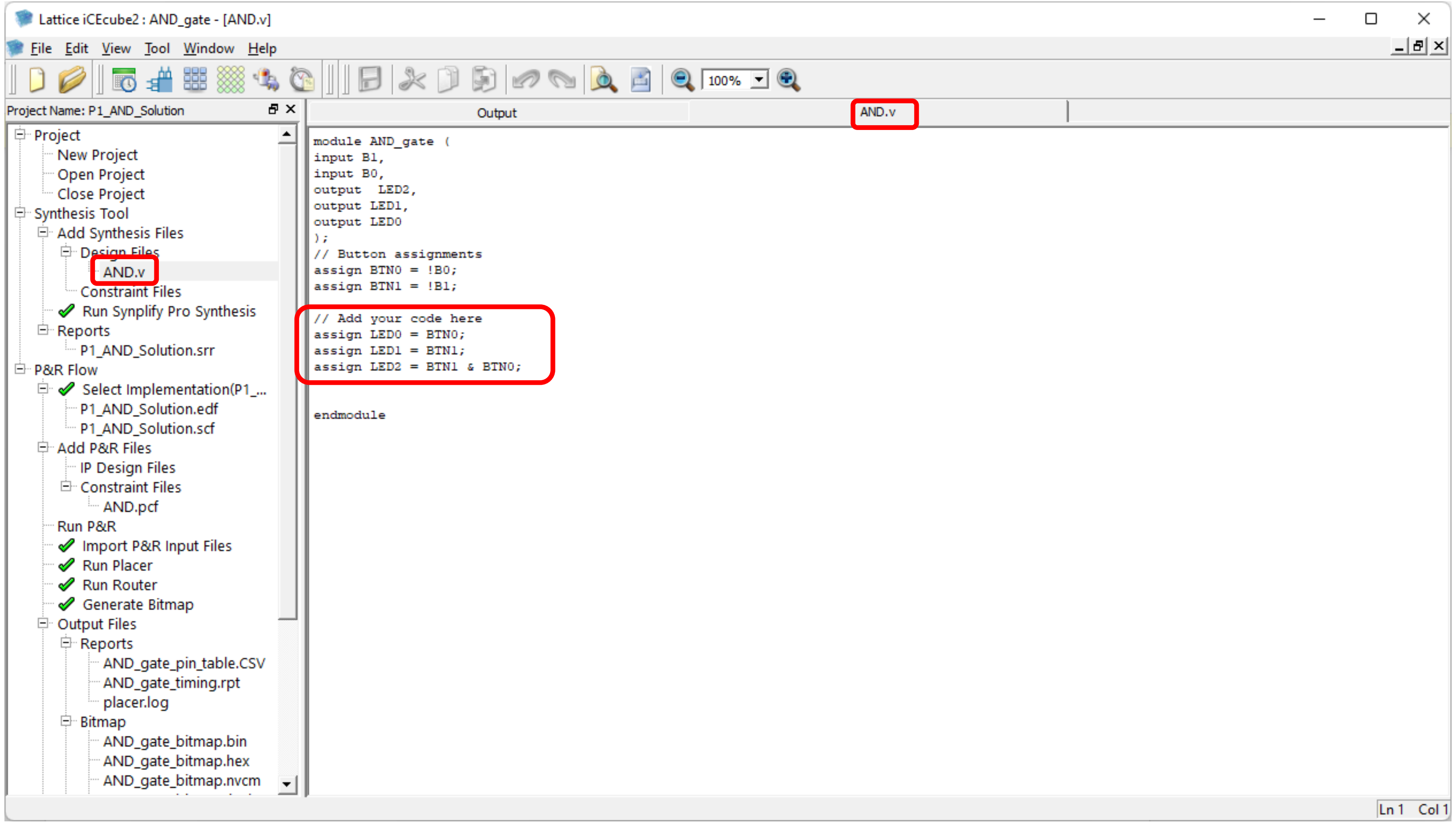
Project 3 Step 4: Create AND gate design in FPGA

1. Double click AND.v
2. Add three lines of code

    assign LED0 = BTN0;

    assign LED1 = BTN1;

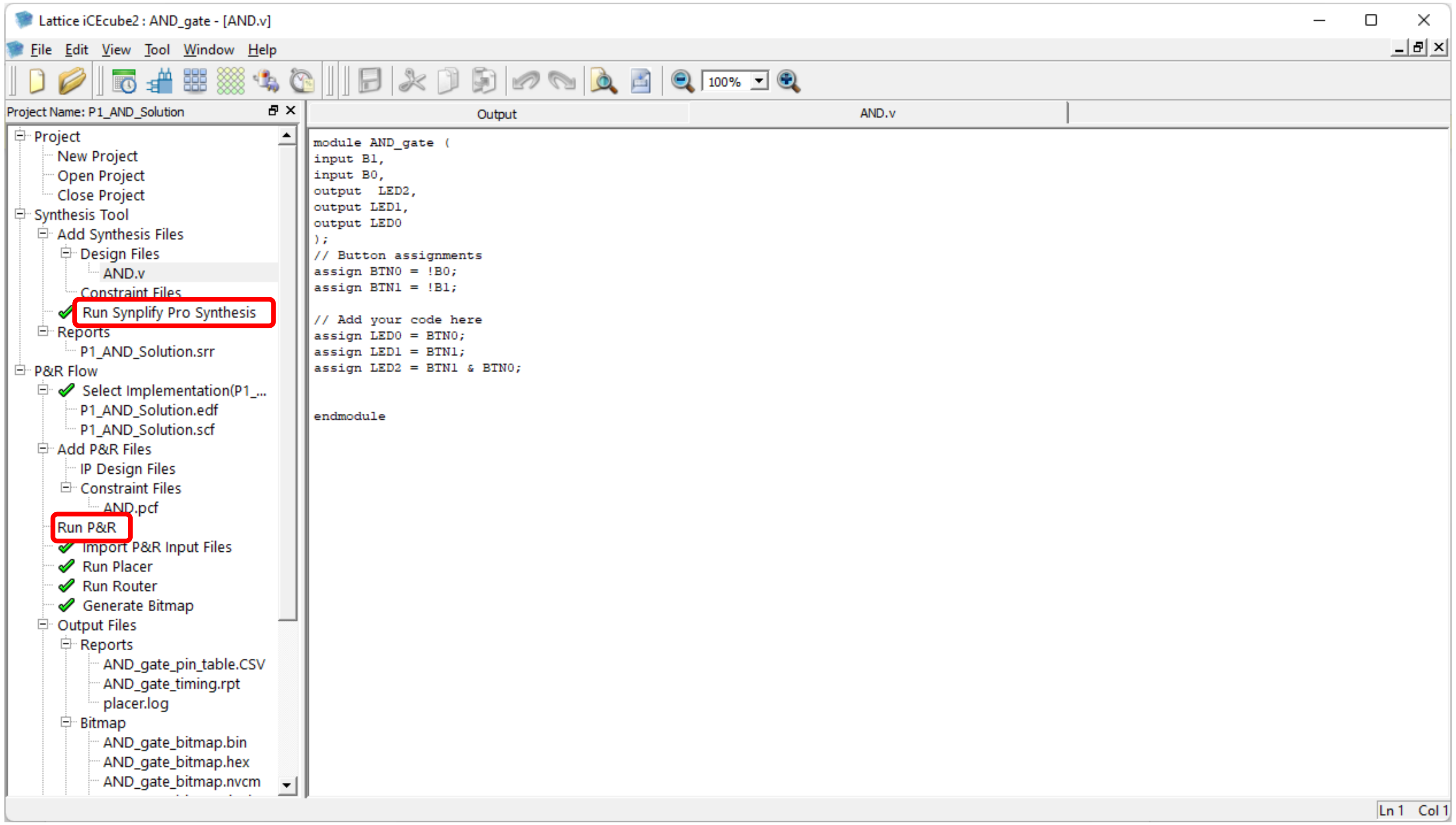
    assign LED2 = BTN1 & BTN0;



```
module AND_gate (
input B1,
input B0,
output  LED2,
output LED1,
output LED0
);
// Button assignments
assign BTN0 = !B0;
assign BTN1 = !B1;

// Add your code here
assign LED0 = BTN0;
assign LED1 = BTN1;
assign LED2 = BTN1 & BTN0;


endmodule
```

Project 3 Step 5: Run “Synplify” and run “P&R”

1. Make iCEcube2 SW open
2. ;
3. Click “run Synplify”
    1. Wait for green check mark
4. Click “run P&R”
    1. Wait for green check mark

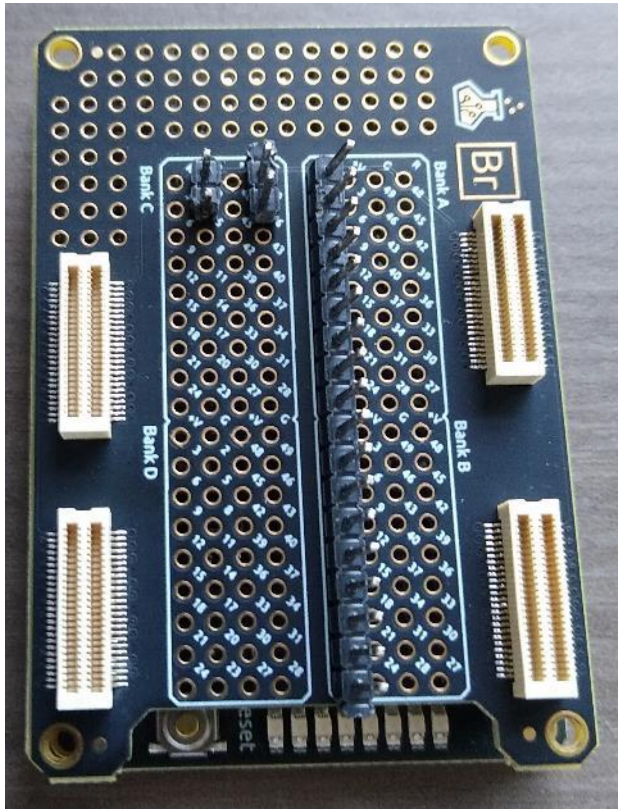


```
module AND_gate (
input B1,
input B0,
output  LED2,
output LED1,
output LED0
);
// Button assignments
assign BTN0 = !B0;
assign BTN1 = !B1;

// Add your code here
assign LED0 = BTN0;
assign LED1 = BTN1;
assign LED2 = BTN1 & BTN0;


endmodule
```

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

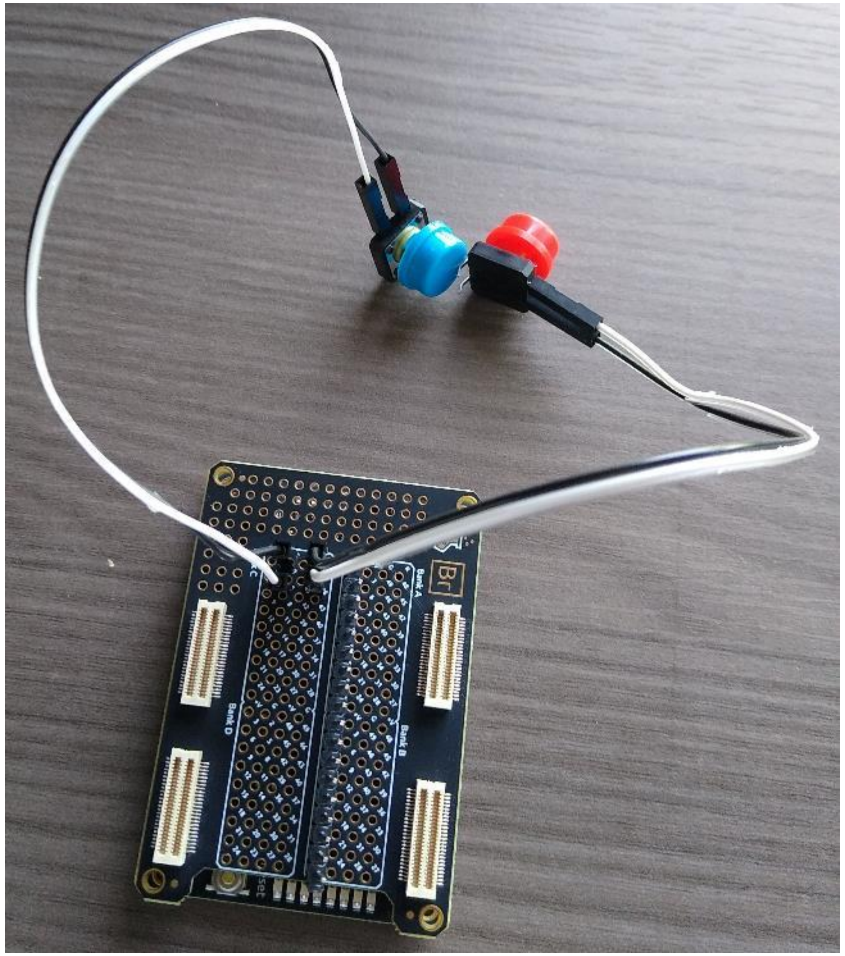
Project 3 Step 6: Place Alchitry Br on Alchitry Cu

1. Connect Alchitry Br on top of Alchitry Cu
2. Make sure all four connectors are snug



Project 3 Step 7: Connect buttons to Alchitry Br

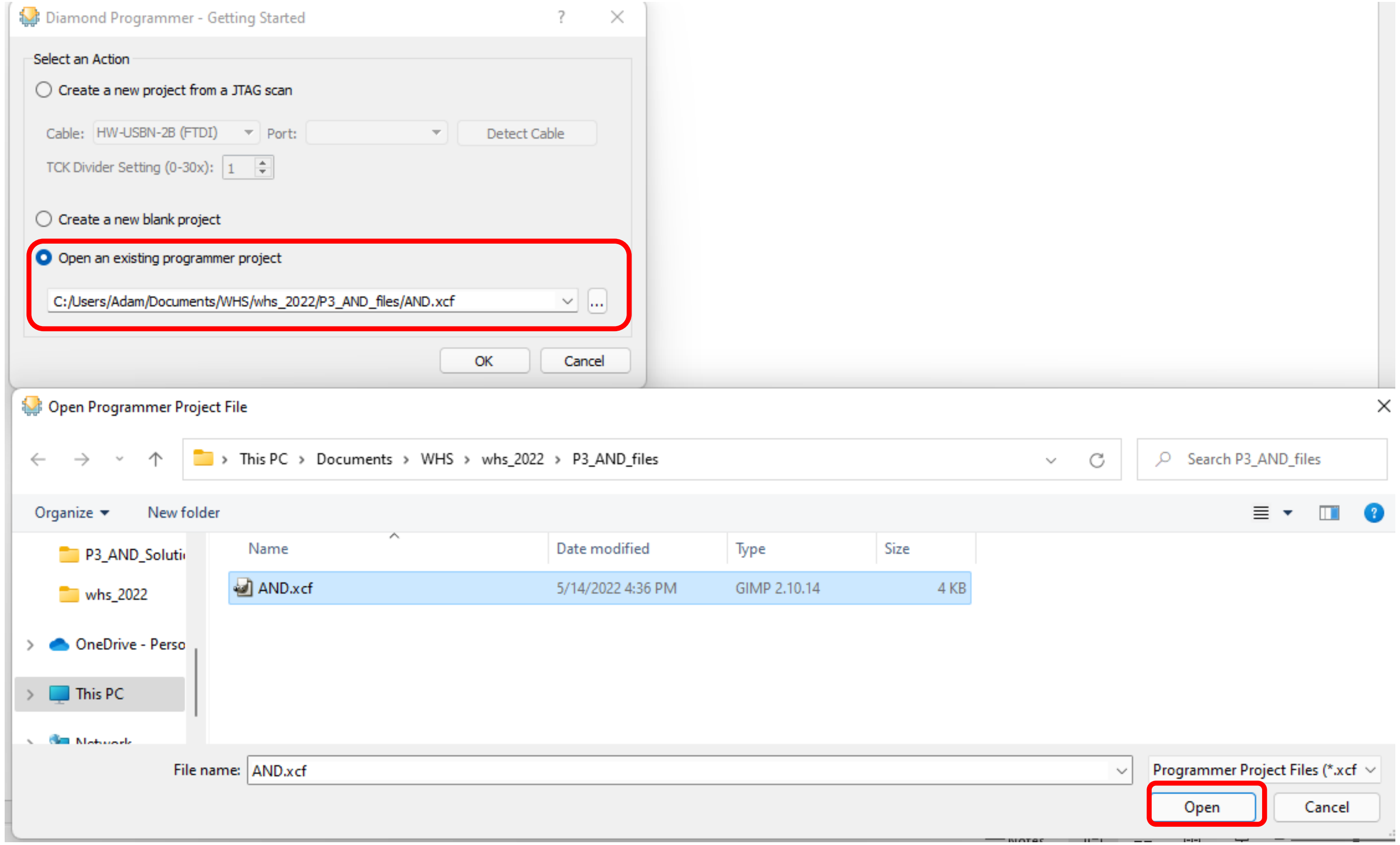
1. Connect white and black wire pair to Blue button
2. Connect white and black wire pair to Red button
3. Connect one pair to the set of two pins on the Br board
4. Connect the other pair to the other set of two pins on the Br board
5. Final assembly should look like the picture on the right



Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Project 3 Step 8: Launch Diamond Programmer and open AND.xcf

1. Open Diamond Programmer SW
2. Select “Open an existing programmer project”
3. Choose **P3_AND_files/AND.xcf**
4. Click “Open”



Diamond Programmer - Getting Started

Select an Action

○ Create a new project from a JTAG scan

Cable: HW-USBN-2B (FTDI)   Port: Detect Cable

TCK Divider Setting (0-30x): 1

○ Create a new blank project

● Open an existing programmer project

C:/Users/Adam/Documents/WHS/whs_2022/P3_AND_files/AND.xcf

Open Programmer Project File

This PC › Documents › WHS › whs_2022 › P3_AND_files

Name	Date modified	Type	Size
AND.xcf	5/14/2022 4:36 PM	GIMP 2.10.14	4 KB

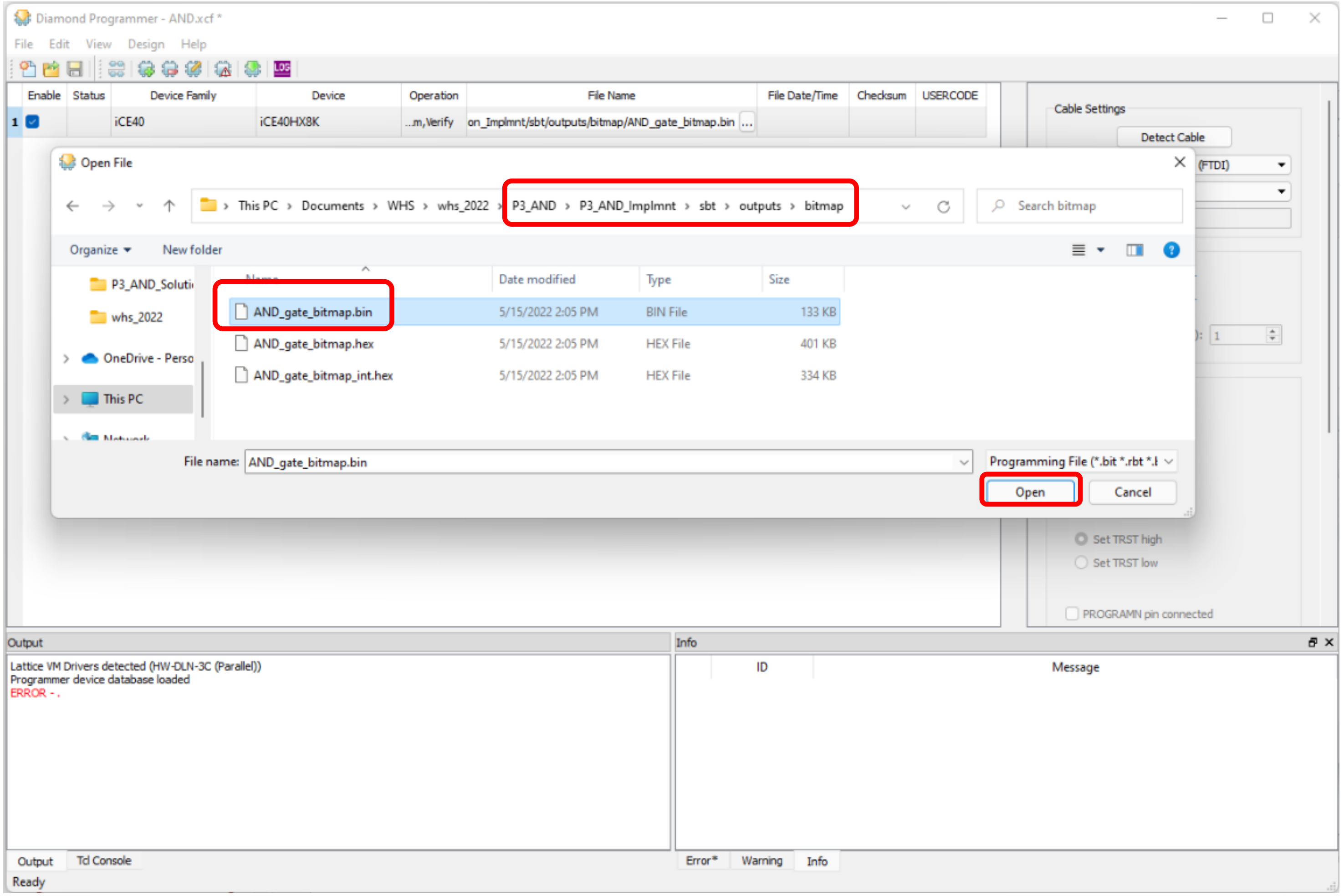
File name: AND.xcf

Programmer Project Files (*.xcf

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Project 3 Step 9: Select AND .bin file

1. Make sure Diamond Programmer SW is open
2. Select File under File Name
3. Navigate to: **P3_AND/P3_AND_Implement/sbt/outputs/bitmap/**
4. Select **AND_gate_bitmap.bin**
5. Click Open



	Enable	Status	Device Family	Device	Operation	File Name	File Date/Time	Checksum	USERCODE
1	☑		iCE40	iCE40HX8K	...m,Verify	on_Implmnt/sbt/outputs/bitmap/AND_gate_bitmap.bin			

This PC › Documents › WHS › whs_2022 › P3_AND › P3_AND_Implmnt › sbt › outputs › bitmap

Name	Date modified	Type	Size
AND_gate_bitmap.bin	5/15/2022 2:05 PM	BIN File	133 KB
AND_gate_bitmap.hex	5/15/2022 2:05 PM	HEX File	401 KB
AND_gate_bitmap_int.hex	5/15/2022 2:05 PM	HEX File	334 KB

File name: AND_gate_bitmap.bin

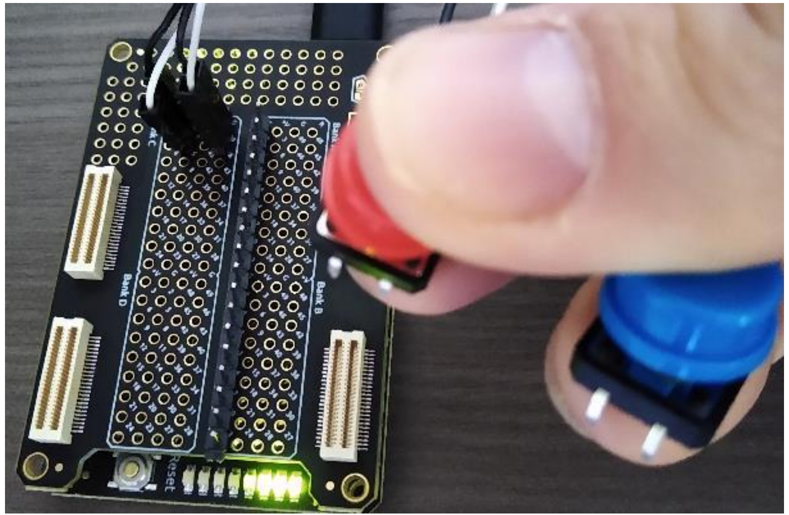
Programming File (*.bit *.rbt *.i

Output

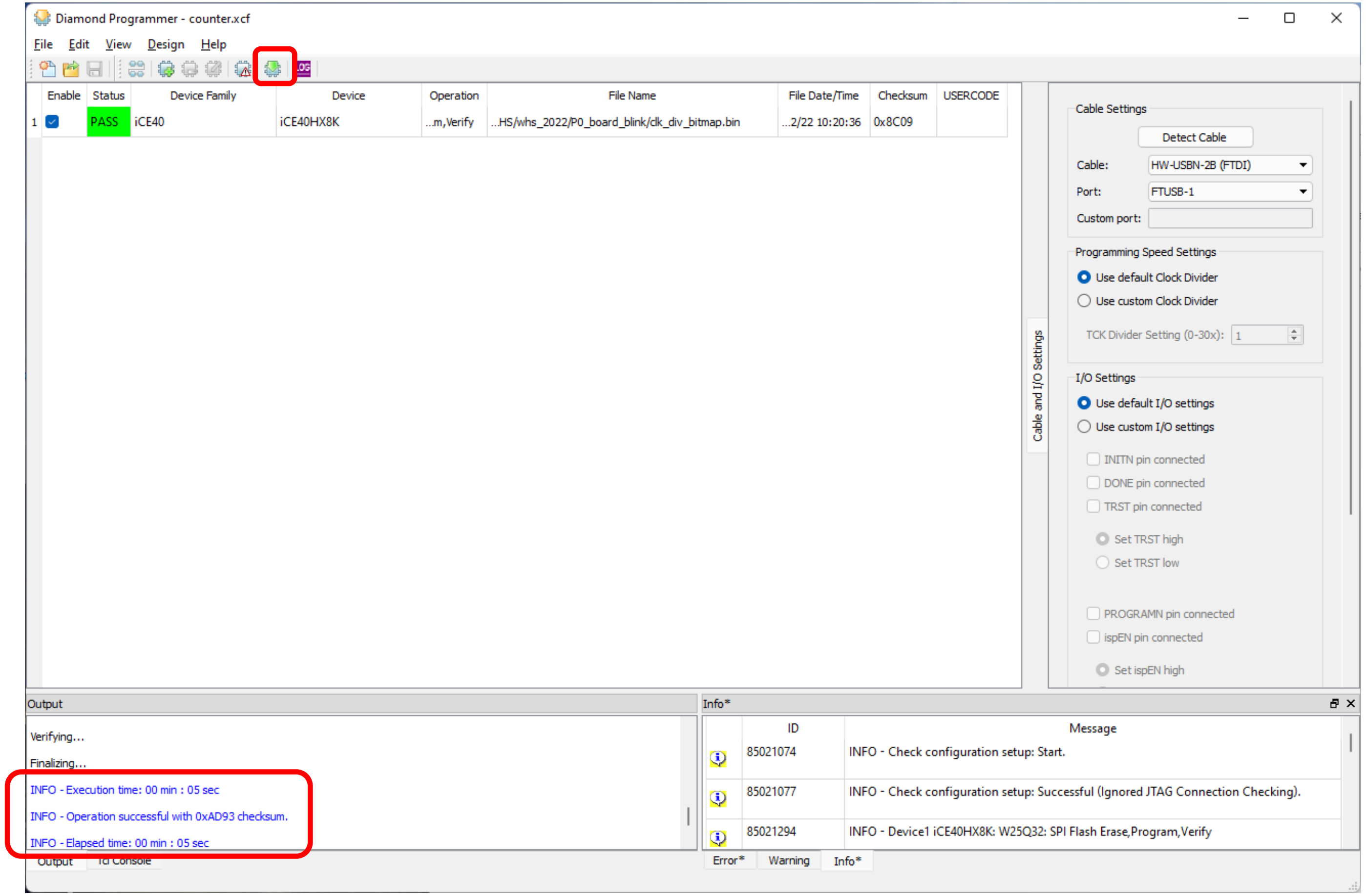
```
Lattice VM Drivers detected (HW-DLN-3C (Parallel))
Programmer device database loaded
ERROR - .
```

Project 3 Step 10: Program AND into FPGA

1. Click Program Icon (or Design>Program)
2. Click buttons and observe AND behavior
3. Make sure you see blue text with “operation successful” in bottom left



LEDs for AND behavior



	Enable	Status	Device Family	Device	Operation	File Name	File Date/Time	Checksum	USERCODE
1	☑	PASS	iCE40	iCE40HX8K	...m,Verify	...HS/whs_2022/P0_board_blink/clk_div_bitmap.bin	...2/22 10:20:36	0x8C09	

Cable Settings

Detect Cable

Cable: HW-USBN-2B (FTDI)

Port: FTUSB-1

Custom port:

Programming Speed Settings

● Use default Clock Divider

○ Use custom Clock Divider

TCK Divider Setting (0-30x): 1

I/O Settings

● Use default I/O settings

○ Use custom I/O settings

□ INITN pin connected

□ DONE pin connected

□ TRST pin connected

○ Set TRST high

○ Set TRST low

□ PROGRAMN pin connected

□ ispEN pin connected

○ Set ispEN high

```
Verifying...
Finalizing...
INFO - Execution time: 00 min : 05 sec
INFO - Operation successful with 0xAD93 checksum.
INFO - Elapsed time: 00 min : 05 sec
```

ID	Message
85021074	INFO - Check configuration setup: Start.
85021077	INFO - Check configuration setup: Successful (Ignored JTAG Connection Checking).
85021294	INFO - Device1 iCE40HX8K: W25Q32: SPI Flash Erase,Program,Verify

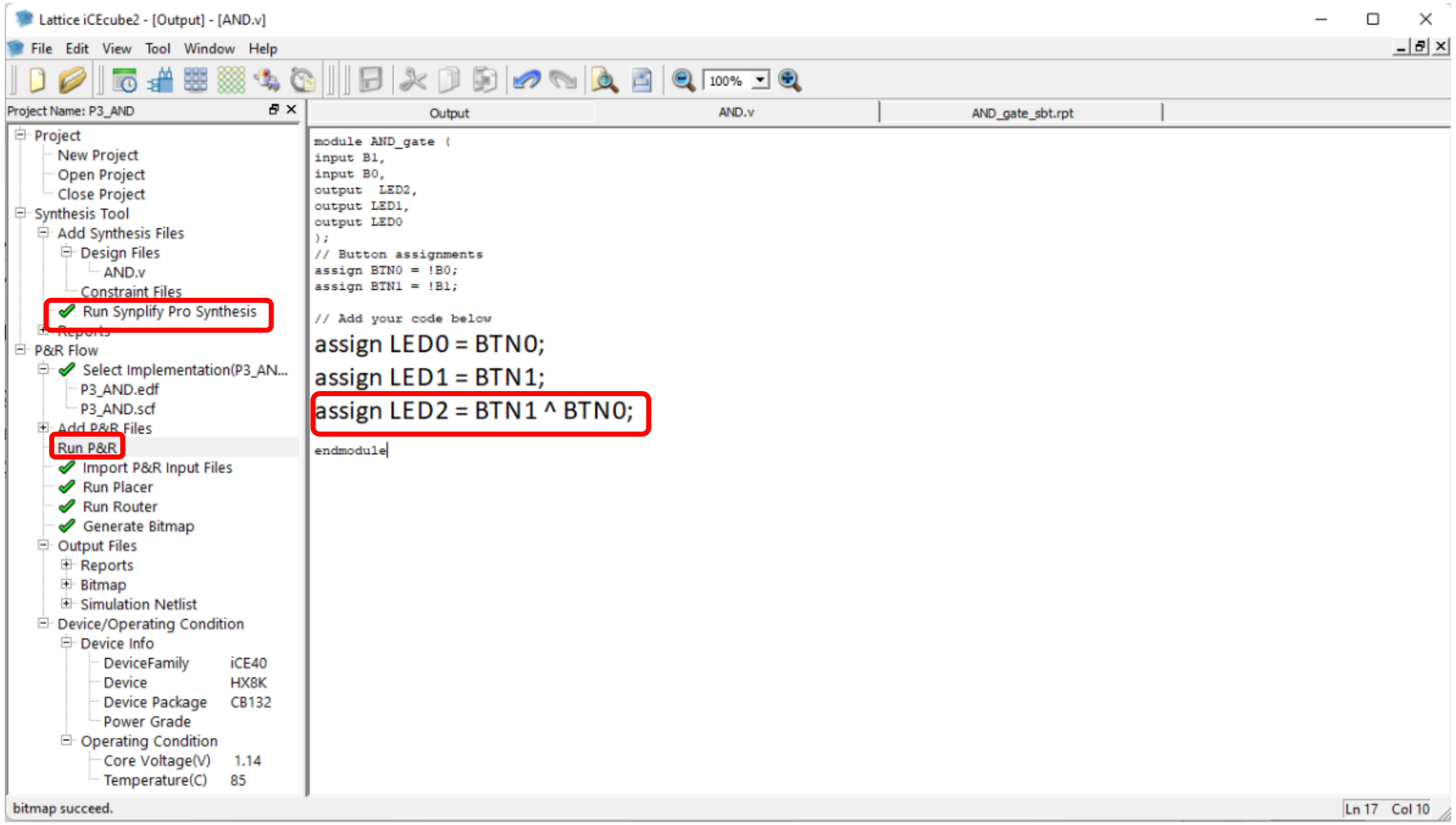
Distribution Statement A: Approved for Public Release; Distribution is unlimited.

Project 3 Step 11: Change AND to XOR

1. Change arithmetic operate from AND (&) to XOR (^)
    1. Assign LED2 = BTN1 ^ BTN0;
2. Click save to save AND.v
3. Rerun “Run Synplify”
4. Rerun “Run P&R”
5. Make sure all check marks green
6. Repeat step “Program into FPGA” on previous slide
7. Press the Red and Blue buttons to verify that the FPGA incorporated your change
8. Try others below as well:

&	reduction AND
|	reduction OR
~&	reduction NAND
~|	reduction NOR
^	reduction XOR
~^ or ^~	reduction XNOR

https://class.ece.uw.edu/cadta/verilog/operators.html



DeviceFamily	iCE40
Device	HX8K
Device Package	CB132
Power Grade	
Core Voltage(V)	1.14
Temperature(C)	85

```
module AND_gate (
input B1,
input B0,
output  LED2,
output LED1,
output LED0
);
// Button assignments
assign BTN0 = !B0;
assign BTN1 = !B1;

// Add your code below
assign LED0 = BTN0;
assign LED1 = BTN1;
assign LED2 = BTN1 ^ BTN0;

endmodule
```

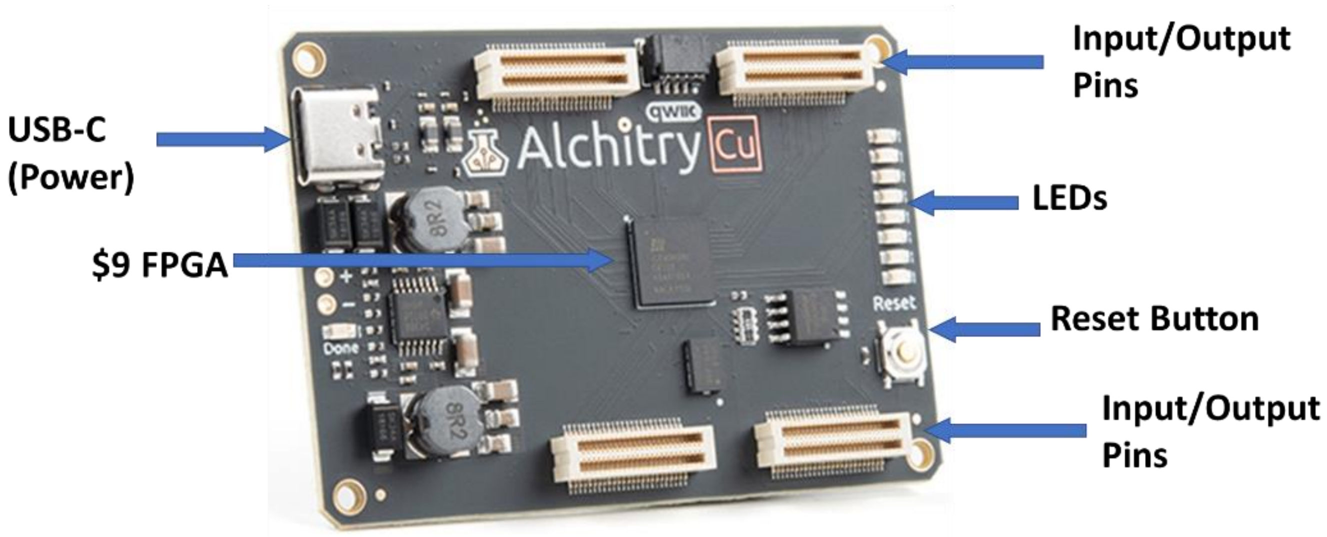
bitmap succeed.

Ln 17 Col 10

Distribution Statement A: Approved for Public Release; Distribution is unlimited.

Project 4: Build a video game system!

- **Turn the Alchitry Cu into a video game system by programming FPGA to:**
  - Run game engine
  - Create VGA video
  - Implement a controller
  - Transmit score to LEDs



USB-C (Power)

$9 FPGA

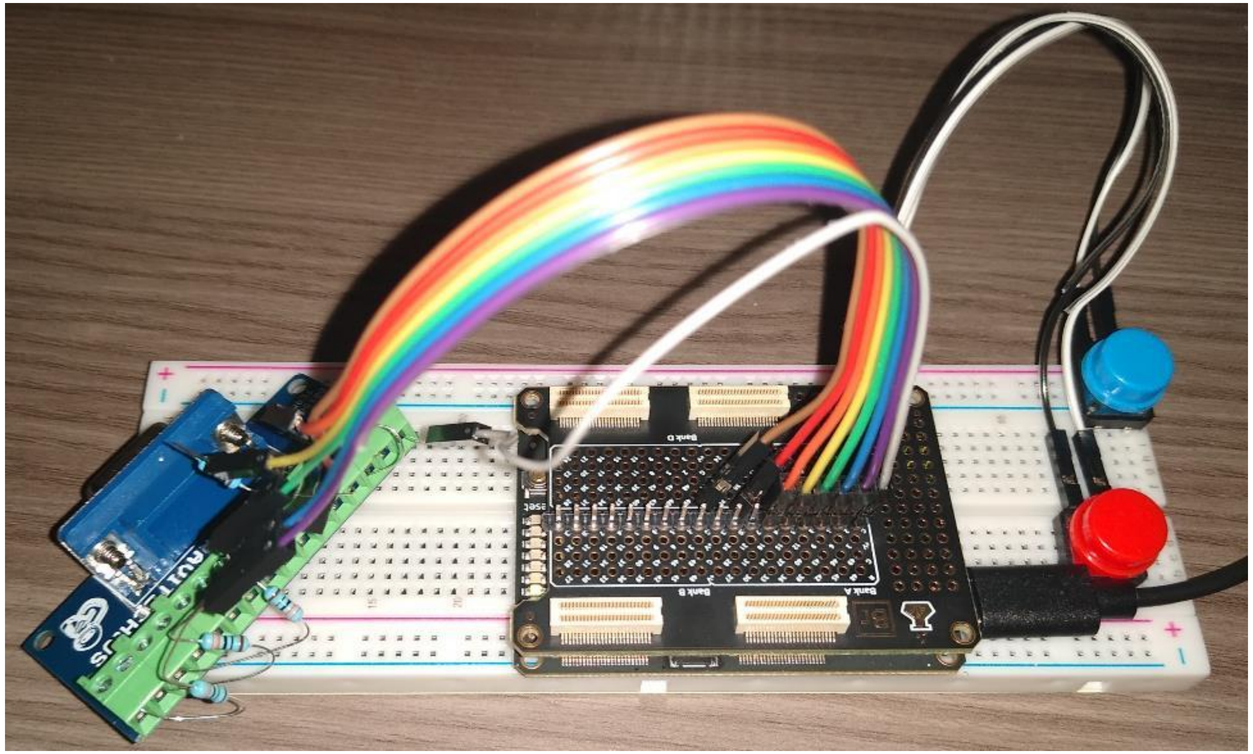
Input/Output Pins

LEDs

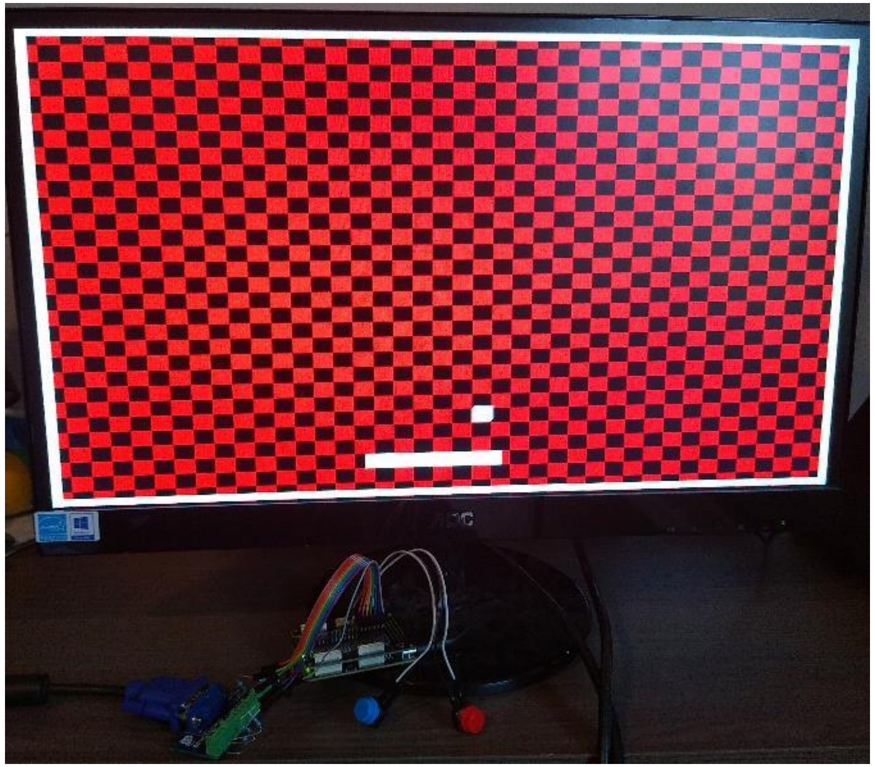
Reset Button

Input/Output Pins

Project 4: What it could look like …



Final base design will look something like this …



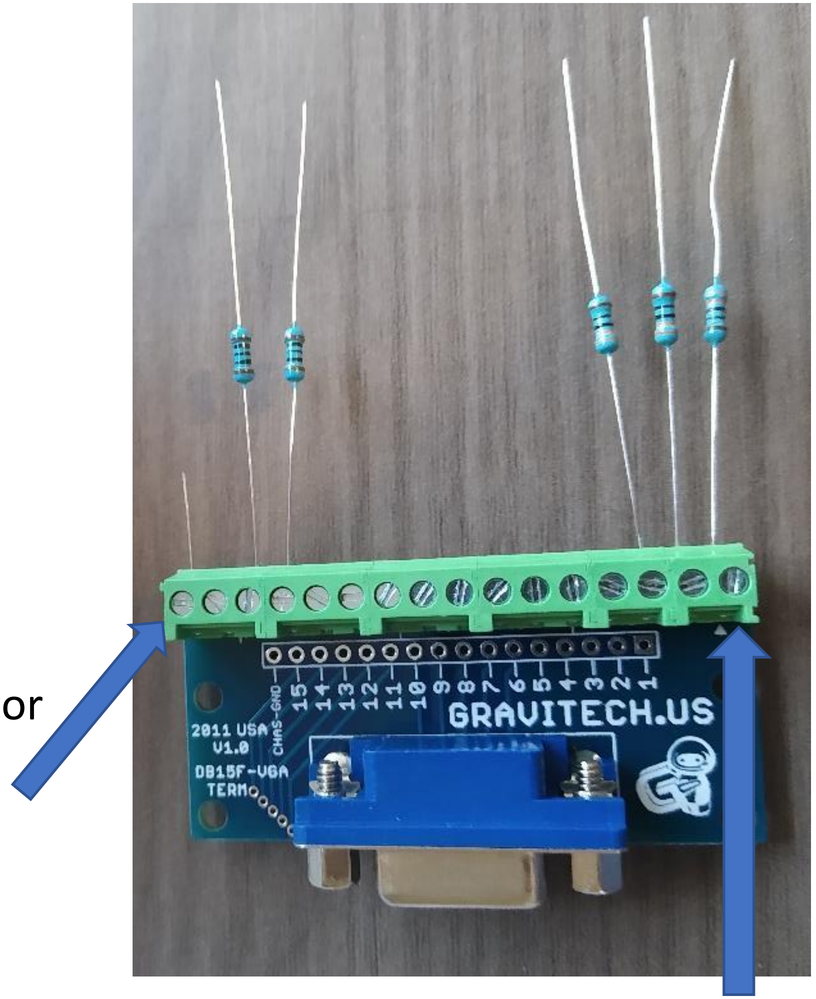
The VGA output should look like this …

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Project 4 Step 1: Connect to VGA module

Make the following connections

VGA pin screw connector	Connection	Name
1	330 ohm resistor	R
2	330 ohm resistor	G
3	330 ohm resistor	B
13	1000 ohm resistor	Hsync
14	1000 ohm resistor	Vsync
16 (GND)	Wire (cut wire from resistor)	GND



Screw connector pin 16 (GND)

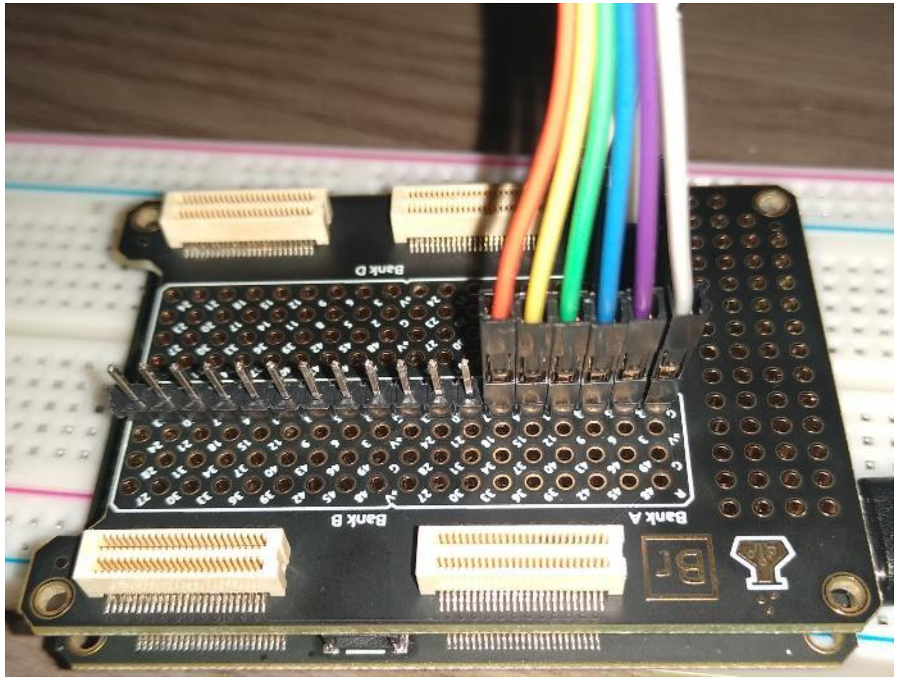
Screw connector pin 1

Distribution Statement A: Approved for Public Release; Distribution is unlimited.

Project 4 Step 2: Connect ribbon cable to Alchitry Br

Make the following connections

Alchitry Br Pin #	Ribbon Connector Color
1	Grey
2	Purple
3	Blue
4	Green
5	Yellow
6	Orange

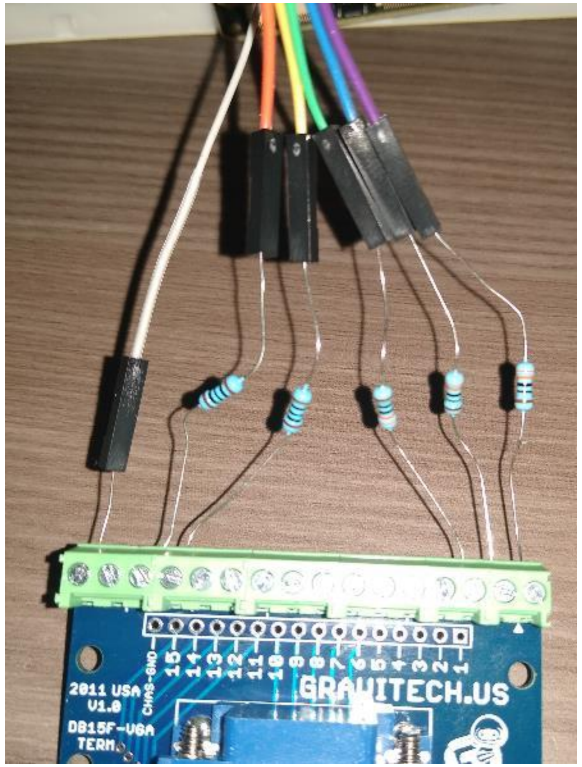


Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Project 4 Step 3: Connect ribbon cable to resistors and wires on VGA board

Make the following connections

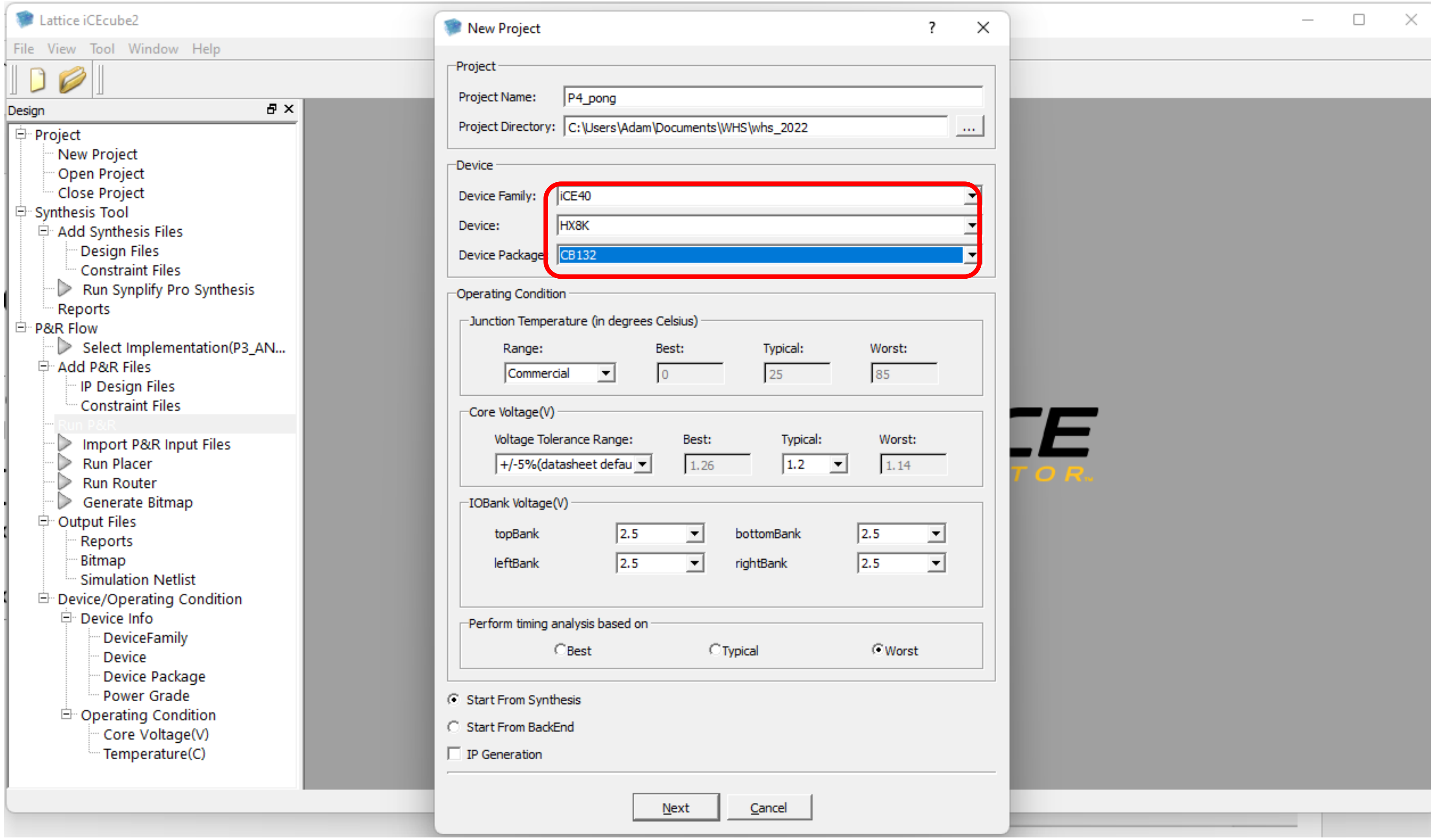
VGA pin screw connector	Connection	Ribbon Connector Color	Name
1	330 ohm resistor	Purple	R
2	330 ohm resistor	Blue	G
3	330 ohm resistor	Green	B
13	1000 ohm resistor	Yellow	Hsync
14	1000 ohm resistor	Orange	Vsync
16 (GND)	Wire (cut wire from resistor)	Grey	GND



Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Project 4 Step 4: Create pong project in ICEcube2

1. Open iCEcube2 SW
2. File>New Project
3. Type Project Name = P4_pong
    1. Set Device Family = ICE40
    2. Set Device = HX8K
    3. Set Device Package = CB132
4. Click Next



Project Name: P4_pong

Project Directory: C:\Users\Adam\Documents\WHS\whs_2022

Device Family: iCE40

Device: HX8K

Device Package: CB132

Range: Commercial Best: 0 Typical: 25 Worst: 85

Voltage Tolerance Range: +/-5%(datasheet defau Best: 1.26 Typical: 1.2 Worst: 1.14

topBank 2.5 bottomBank 2.5

leftBank 2.5 rightBank 2.5

○ Best ○ Typical ● Worst

● Start From Synthesis

○ Start From BackEnd

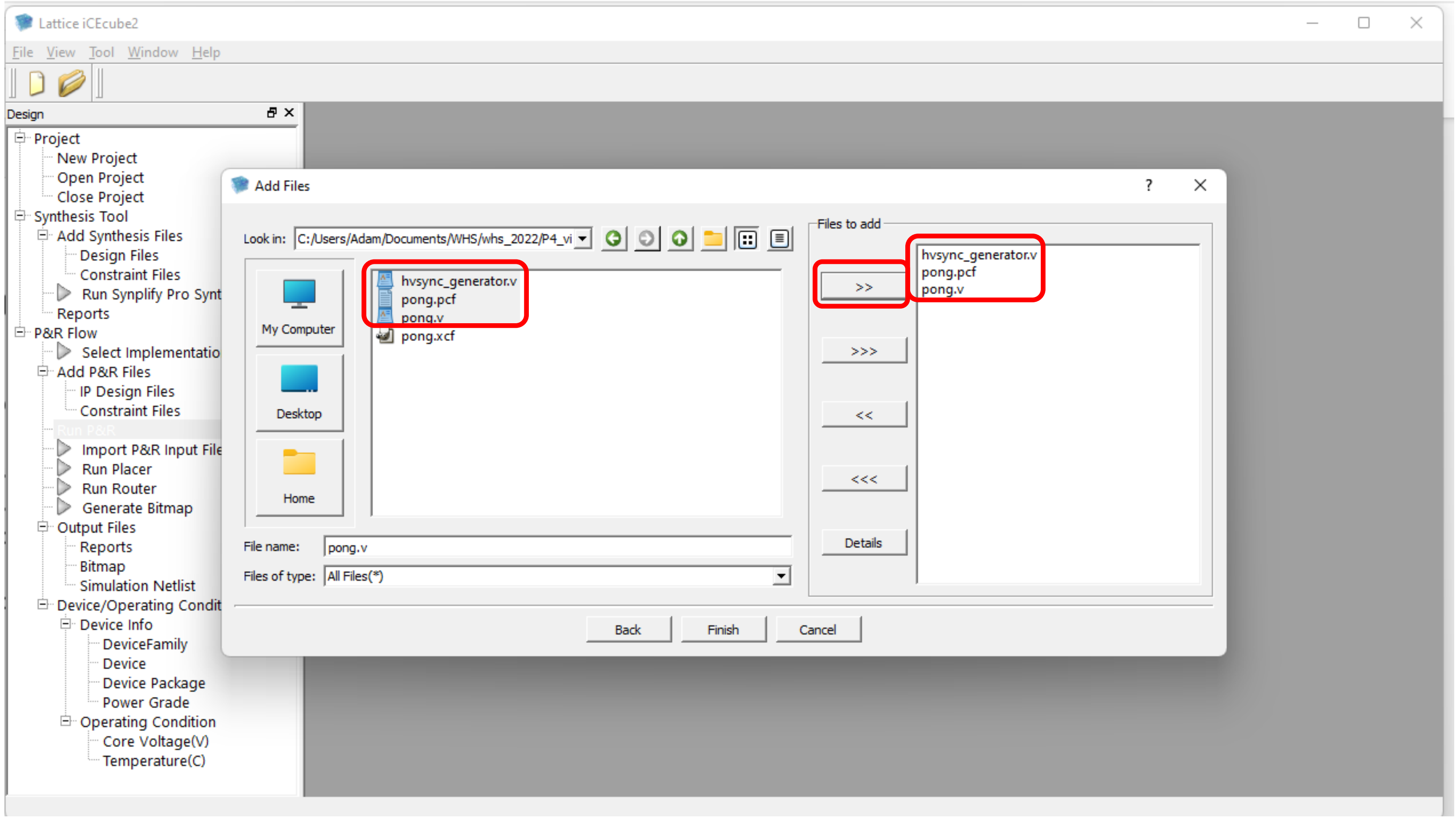
□ IP Generation

Next Cancel

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Project 4 Step 5: Create pong project in ICEcube2

1. Add Files should pop up
2. Select files in P4_pong:
   1. pong.pcf
   2. pong.v
   3. hysnc_generator.v
3. Click the “>>” icon so they show up in the “Files to Add”
4. Click Finish



Look in: C:/Users/Adam/Documents/WHS/whs_2022/P4_vi

hvsync_generator.v
pong.pcf
pong.v
pong.xcf

Files to add

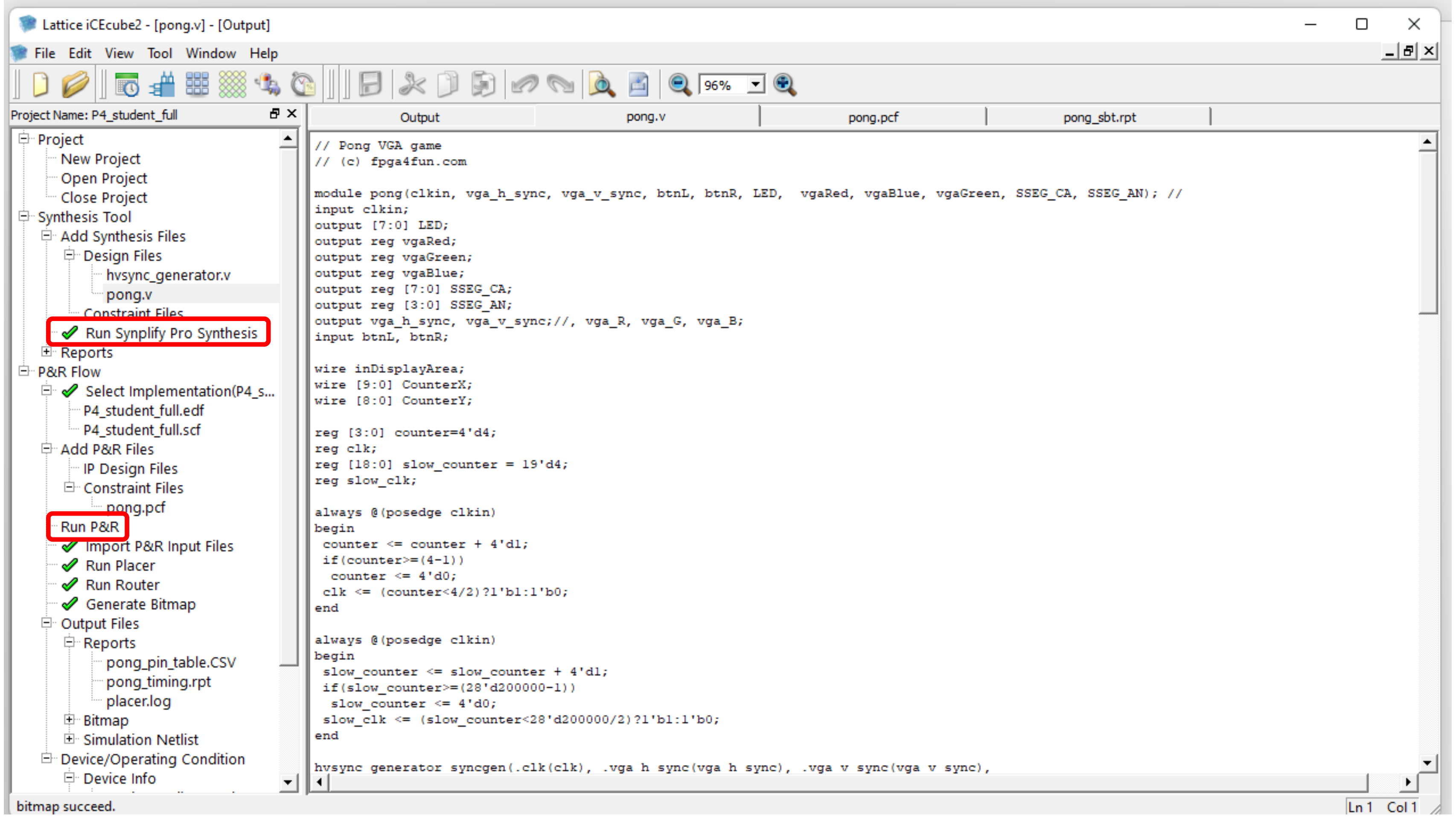
hvsync_generator.v
pong.pcf
pong.v

File name: pong.v

Files of type: All Files(*)

Project 4 Step 5: Run “Synplify” and run “P&R”

1. Click “run Synplify”
   1. Wait for green check mark
2. Click “run P&R”
   1. Wait for green check mark

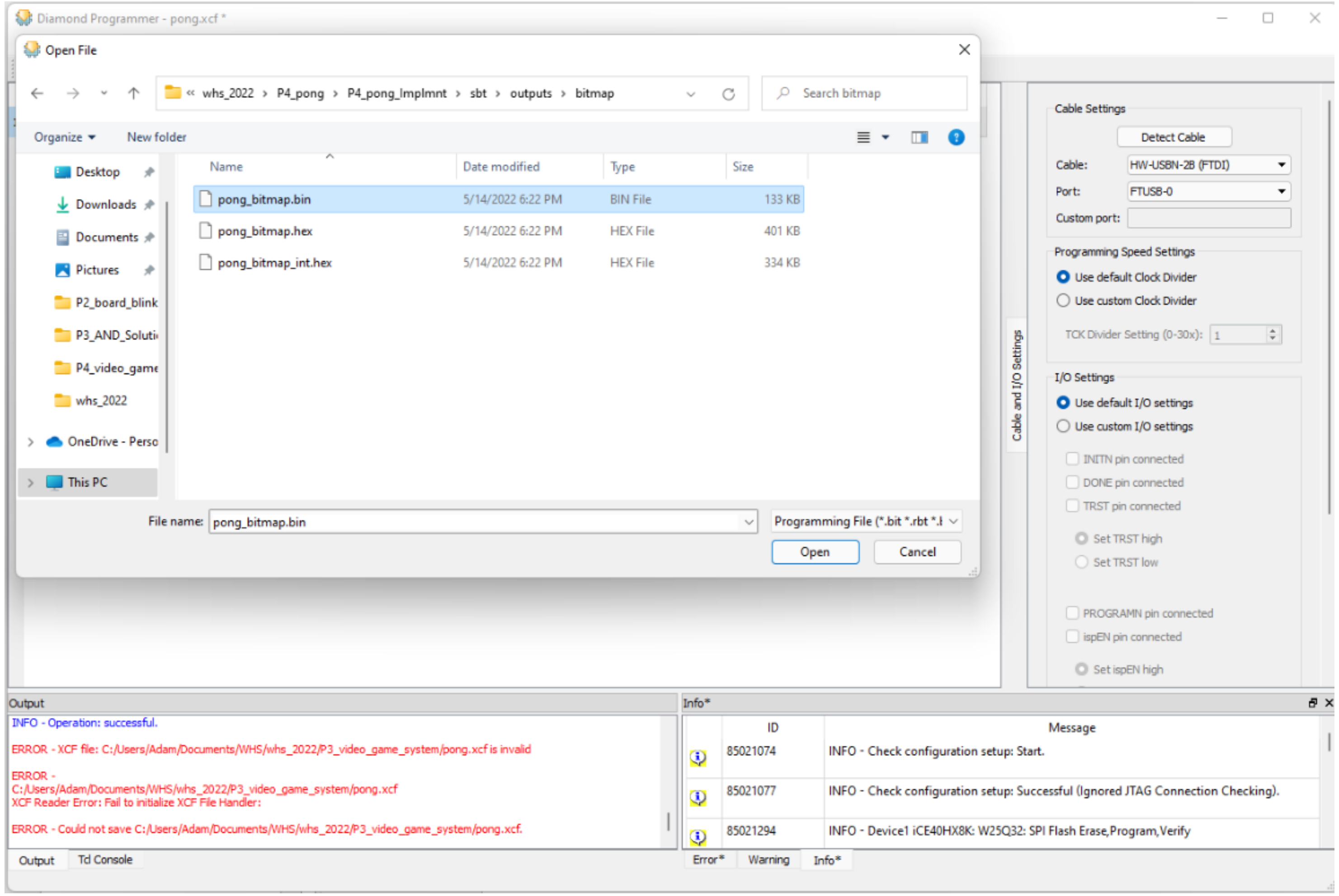


```
// Pong VGA game
// (c) fpga4fun.com

module pong(clkin, vga_h_sync, vga_v_sync, btnL, btnR, LED,  vgaRed, vgaBlue, vgaGreen, SSEG_CA, SSEG_AN); //
input clkin;
output [7:0] LED;
output reg vgaRed;
output reg vgaGreen;
output reg vgaBlue;
output reg [7:0] SSEG_CA;
output reg [3:0] SSEG_AN;
output vga_h_sync, vga_v_sync;//, vga_R, vga_G, vga_B;
input btnL, btnR;

wire inDisplayArea;
wire [9:0] CounterX;
wire [8:0] CounterY;

reg [3:0] counter=4'd4;
reg clk;
reg [18:0] slow_counter = 19'd4;
reg slow_clk;

always @(posedge clkin)
begin
 counter <= counter + 4'd1;
 if(counter>=(4-1))
  counter <= 4'd0;
 clk <= (counter<4/2)?1'b1:1'b0;
end

always @(posedge clkin)
begin
 slow_counter <= slow_counter + 4'd1;
 if(slow_counter>=(28'd200000-1))
  slow_counter <= 4'd0;
 slow_clk <= (slow_counter<28'd200000/2)?1'b1:1'b0;
end

hvsync generator syncgen(.clk(clk), .vga h sync(vga h sync), .vga v sync(vga v sync),
```

```
bitmap succeed.
```

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.

Project 4 Step 6: Load files into IceCube2



Name	Date modified	Type	Size
pong_bitmap.bin	5/14/2022 6:22 PM	BIN File	133 KB
pong_bitmap.hex	5/14/2022 6:22 PM	HEX File	401 KB
pong_bitmap_int.hex	5/14/2022 6:22 PM	HEX File	334 KB

File name: pong_bitmap.bin

Programming File (*.bit *.rbt *.I

Cable Settings

Cable: HW-USBN-2B (FTDI)

Port: FTUSB-0

Custom port:

Programming Speed Settings

● Use default Clock Divider

○ Use custom Clock Divider

TCK Divider Setting (0-30x): 1

I/O Settings

● Use default I/O settings

○ Use custom I/O settings

□ INITN pin connected

□ DONE pin connected

□ TRST pin connected

○ Set TRST high

○ Set TRST low

□ PROGRAMN pin connected

□ ispEN pin connected

○ Set ispEN high

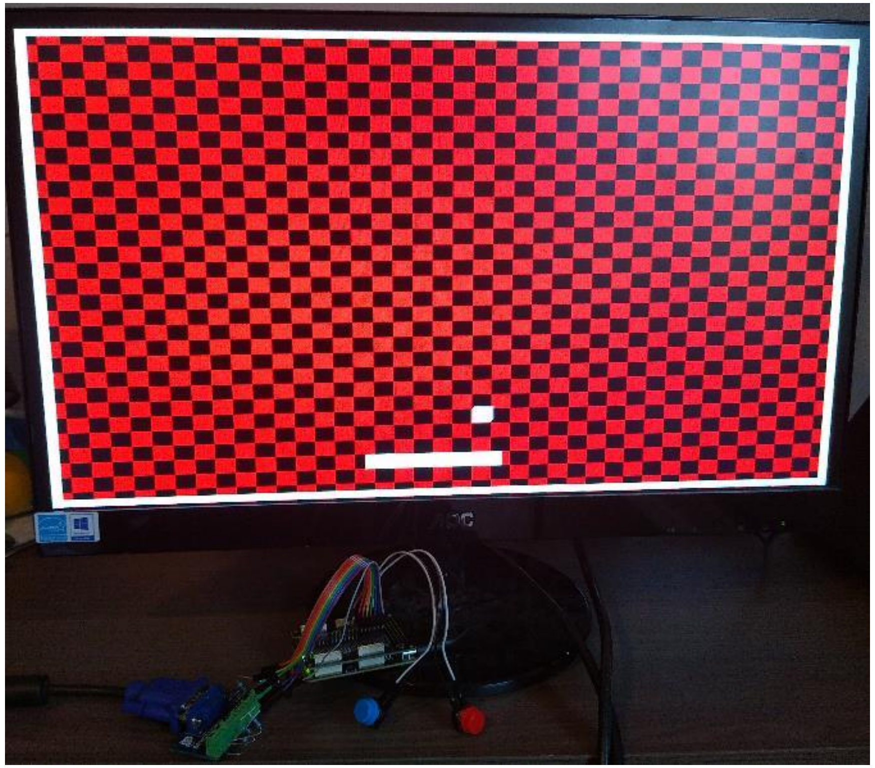
Output

```
INFO - Operation: successful.

ERROR - XCF file: C:/Users/Adam/Documents/WHS/whs_2022/P3_video_game_system/pong.xcf is invalid

ERROR -
C:/Users/Adam/Documents/WHS/whs_2022/P3_video_game_system/pong.xcf
XCF Reader Error: Fail to initialize XCF File Handler:

ERROR - Could not save C:/Users/Adam/Documents/WHS/whs_2022/P3_video_game_system/pong.xcf.
```

Info*

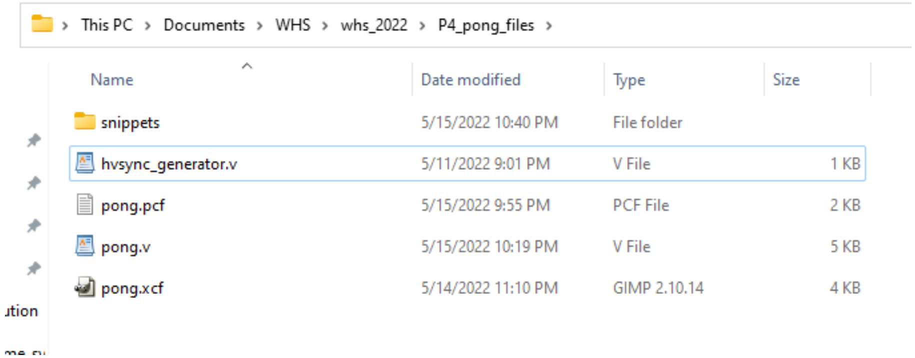
	ID	Message
ⓘ	85021074	INFO - Check configuration setup: Start.
ⓘ	85021077	INFO - Check configuration setup: Successful (Ignored JTAG Connection Checking).
ⓘ	85021294	INFO - Device1 iCE40HX8K: W25Q32: SPI Flash Erase,Program,Verify

Project 4 Step 7: Make edits to pong.v



The VGA output should look like this …

1. Check to see if you see output on VGA monitor
2. Verify buttons work to move paddle
3. Go into the snippets directory and use files to make modifications to pong.v to add features to your design



› This PC › Documents › WHS › whs_2022 › P4_pong_files ›

Name	Date modified	Type	Size
snippets	5/15/2022 10:40 PM	File folder	
hvsync_generator.v	5/11/2022 9:01 PM	V File	1 KB
pong.pcf	5/15/2022 9:55 PM	PCF File	2 KB
pong.v	5/15/2022 10:19 PM	V File	5 KB
pong.xcf	5/14/2022 11:10 PM	GIMP 2.10.14	4 KB

Backup

IceCube2 License (Should be done ahead of lesson)

- Free license at link below:
- https://www.latticesemi.com/Support/Licensing/DiamondAndiCEcube2SoftwareLicensing/iceCube2

“Review your Web Account information below. [Edit]Name: John Doe
Email: john.doe@gmail.com

Fill in the Software License Request Form and Submit.
Finding the Host NIC:
For Windows, from an MS-DOS window, use the ipconfig /all command
For Linux, from the command prompt, use the ifconfig -a command

After submitting the form successfully, a new license file with instructions on how to install will be emailed to you.”

Distribution Statement A: Approved for Public Release;
Distribution is unlimited.