ECD 322 Open Source FPGA Board

Github: https://github.com/jdambra10/ECD322FPGA
PCB software: EasyEDA
Programming software: iCECube2, Diamond Programmer (NOT Lattice Diamond, they are different)

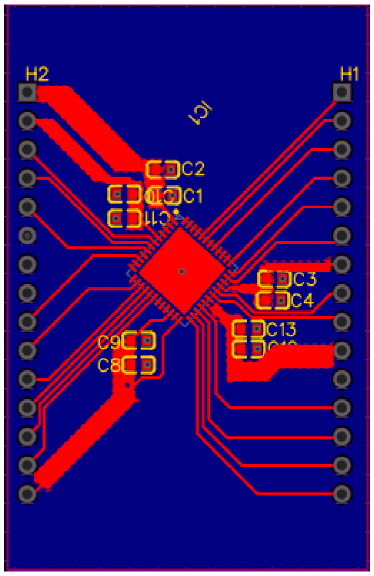
Hardware

FPGA Chip

In this project, we used the iCE5LP4K SG48 chip because it has a simple design with less pins, and we can easily solder it. We chose to use a QFN package instead of a BGA package because for our test socket, we created a PCB. The PCB software we used does not allow us to route BGA pins because the clearance between pins is too small.

Test Socket PCB Design

The test socket design we chose allows for the decoupling capacitors to be soldered close to the FPGA, which is an important design rule. We also attached the pins we use for the design to a header, so that we did not need to route all pins. This allowed for a simpler design. The pins are routed to headers as such:



Notice that H2 is to the left of H1. (This part of the design was not on purpose, but it was difficult to change once we created the main PCB design). The pins routed are as follows:

H2	H1

3V3	Clock1
1V2	Switch
IO_1	LED
CDONE	Clock2
CRESET	GPIO5
GND	GPIO7
Blank	3V3_3
Blank	GPIO8
IO_3	GPIO6
SPI_SO	1V2_1
SPI_SCK	Blank
SPI_CSN	GPIO4
SPI_SI	GPIO2
IO	GPIO1
3V3_2	GPIO3

The 1V2 power pins are the core for the FPGA, and the 3V3 pins are for the IOs. More specifications for the FPGA are in the iCE40 Ultra datasheet.

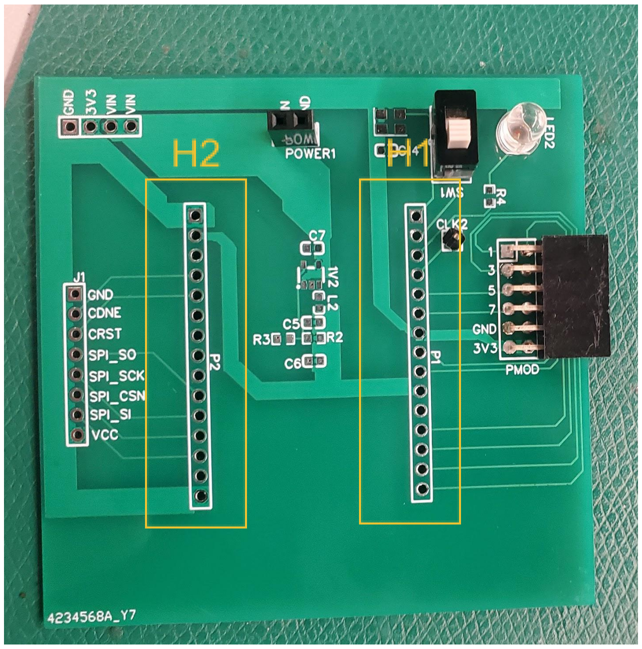
The FPGA chip can be soldered onto the board using the reflow oven in the Watson Fab Lab. We recommend getting a stencil for the chip, along with a squeegee and solder paste. This will make the chip soldering process much simpler.

Main PCB

The main PCB contains all of the requirements. The only potential issues include the on-board 8 MHz oscillator, as we never had the chance to test it. Other than the oscillator, all other components have been tested on the board.

Another issue could be the soldering process of the 1.2V regulator circuit. We recommend buying a stencil with the PCB to use the reflow oven. We soldered the circuit by hand under a microscope, which also works but is more time consuming.

Note the test socket will go in the following orientation, with H2 on P2 and H1 on P1:



P1 and P2 should have female headers soldered to them, and the test socket with male headers to fit onto the board.

Note that in our images and the pre-soldered PCBs, the CDONE and CRST pins do not have a pullup resistor. The pull up resistors are necessary for the project to work (it was not programming properly without them). Unless you order the newly modified PCB which includes it, the old PCBs do not have them and you would need to hard solder a pull up resistor. All of our PCB files have been updated to include the resistors.

Software

iCECube2

We recommend using this guide to using the iCECube2 software. This is the software you will use to generate the bitstream for the project. It is similar to Vivado but less user friendly and does not include a hardware manager. To connect to the hardware you will need a separate software, Diamond Programmer.

iCECube2 requires a license, but it is free, you just have to sign up for it. They will then send you an email with instructions to set up the license. Once you download the software, it also includes an example project. The user guide walks you through how to generate a bitstream with the example. We highly recommend using the example first to gain an idea of how to use the software.

Setting Up the Programmer

To set up the board, the Vin requires a 5V supply to power the board. Once the board is powered you need to connect the lattice programmer (this). The pins should be connected as follows:

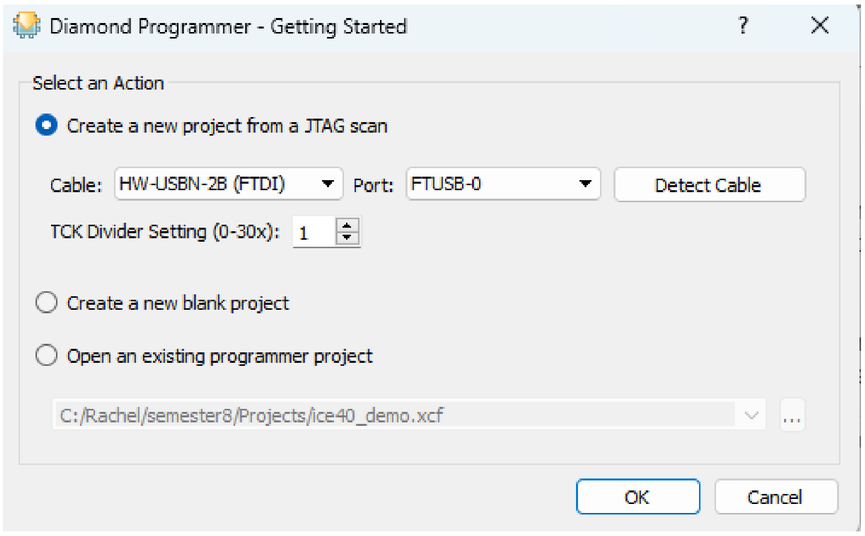
Color	**Name on Programmer**	**FPGA Pin Name**	**Pin Number**
Orange	TDI/SI	SPI_SI	17
Brown	TDO/SO	SPI_SO	14
White	TCK	SPI_SCK	15
Yellow	ISPEN/PROG	SPI_SS/CSN	16
Blue	DONE	CDONE	7
Green	TRST	CRESET_B	8
Red	VCC	SPI_VCCIO1 (3.3V)	22
Black	GND	GND	GND

The three other wires will remain unconnected. Later in this tutorial, we will discuss the possibility of disconnecting the CDONE and CRST pins to program.

The “pin numbers” in the table are showing the pin numbers on the FPGA.

Diamond Programmer

When you open Diamond Programmer, it will give you this screen:


Diamond Programmer - Getting Started

Select an Action

(●) Create a new project from a JTAG scan

Cable: HW-USBN-2B (FTDI) Port: FTUSB-0

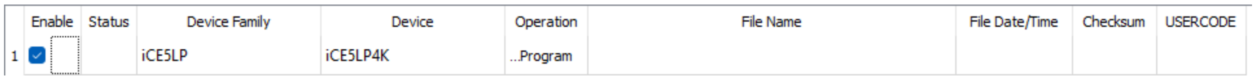
TCK Divider Setting (0-30x): 1

( ) Create a new blank project

( ) Open an existing programmer project

C:/Rachel/semester8/Projects/ice40_demo.xcf

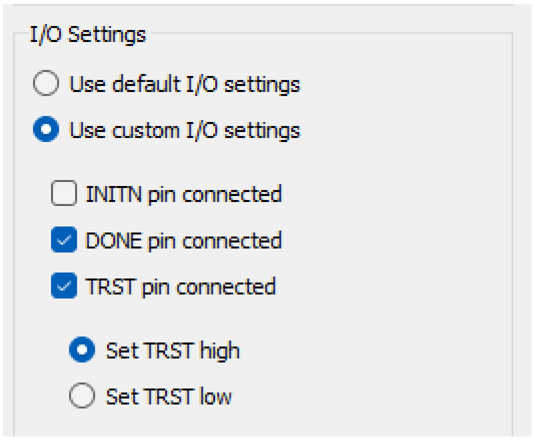
If the programmer is connected to the laptop, all programming pins are connected, and the board is on, you should be able to hit “detect cable,” OK, and it will scan the device. Once Diamond Programmer is open, it should detect the device and read this:



	Enable	Status	Device Family	Device	Operation	File Name	File Date/Time	Checksum	USERCODE
1	✓		iCE5LP	iCE5LP4K	...Program				

If it reads “Generic JTAG Device,” the device has not been properly detected. If it is, then select the “file name” category and select your bitstream. First, ensure that the programming settings (Edit->settings->programming) are not on “NVCM”, otherwise the chip is in single programming mode and will be permanently programmed. You should want to program the “CRAM.”

If the device is not programming, there are a couple of possible options. First, in the right hand column you can set custom I/O settings:



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

If you select these options, make sure the programmer is connected to the CDONE and CRST pins. If this does not work, try disconnecting the two pins on the programmer and deselecting the options for the custom I/O settings. This is where the pull up resistors on those pins are keeping those pins high, rather than the programmer. Sometimes it worked for us to use the custom I/O settings, sometimes it worked to disconnect the pins from the programmer. We are unsure as to why this is.