# DE0 GCD Calculator v2

# What you will need:

* Python2 (tested on 2.7.13)
* TerasIC DE0 Development Board

# Setting up the Program

## Windows

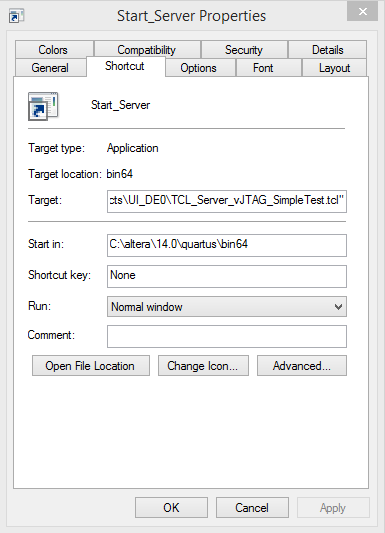
The first thing that needs to be done, is to change the Start\_Server.lnk properties.

1. Under the Shortcut Tab, change the Target to point to the quartus\_stp.exe and the provided TCL\_Server.tcl in the following manner:

Target:

“Path to quartus”\quartus\bin64\quartus\_stp.exe -t C:\<Path to TCL\_Server.tcl>"

Start in:

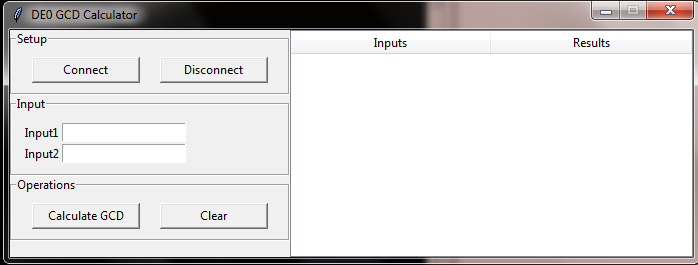
“Path to quartus”\quartus\bin64

Some things to keep in mind about this:

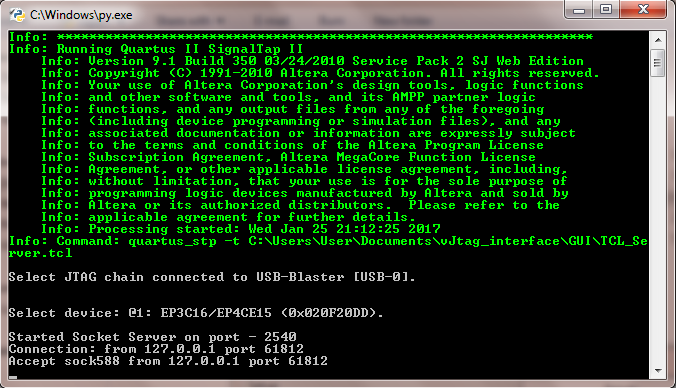
* The version of Quartus should not matter. It has been tested with version 9, 13 and 14. With any version later than 13, you cannot program the Cyclone III FPGA, as it is no longer supported.
* If you move the TCL\_Server.tcl file, you must update the shortcut file.

# Using the GUI

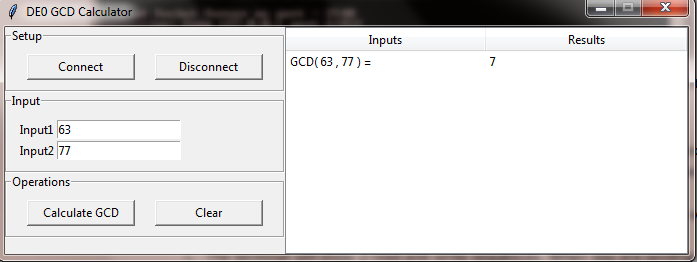
1. To run the program, open the main.py file. This will launch the GUI and the Python terminal.



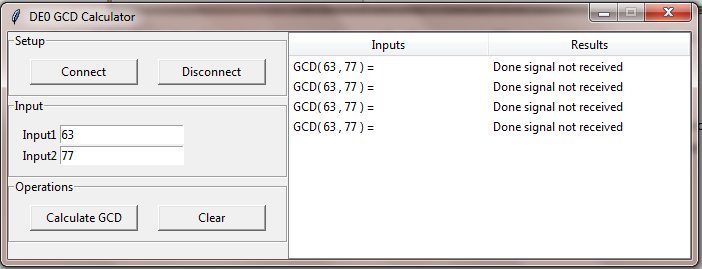
1. Click on “Connect” button, which will run the TCL\_Server through quartus\_stp.exe. If you need to reprogram the DE0, you’ll need to press “Disconnect” or else the reprogramming will fail.



1. To calculate a new GCD, enter “Input1” and “Input2”, and then click on the “Calculate GCD” button. For now, the inputs are restricted from between 1 and 32767.



**Note:** If the done signal is not implemented correctly, the message “Done signal not received” will display instead of the GCD result.

****

# How it works

The GUI is written in python and runs a TCL, TCP/IP server to communicate with the board. The port that is opened is 2540. Once the connection is made to the board from the computer, the python scripts will send binary data to the port number listed. At this point, the TCL script takes over and sends the data in serial. With the TCL script, there are a few main commands that are used that are supported by Quartus.

|  |  |
| --- | --- |
| Open\_device –hardware\_name $usbblaster\_name –device\_name $test\_device | This command opens the JTAG Device so that it will accept further commands |
| device\_virtual\_ir\_shift –instance\_index 0 –ir\_value 1 –no\_captured\_ir\_value | Setups up to send in data |
| device\_virtual\_dr\_shift –dr\_value $send\_data –instance\_index 0 -length 8 –no\_captured\_dr\_value | Data to be sent |
| device\_virtual\_ir\_shift –instance\_index 0 –ir\_value 0 –no\_captured\_ir\_value | Set IR register back to 0 which is bypass mode |
| device\_virtual\_dr\_shift –dr\_value $send\_data –instance\_index 0 -length 8] | This command sends data into the device and also reads back in a value from the tdo pin. |