Laser Control FPGA User Guide



Table of Contents

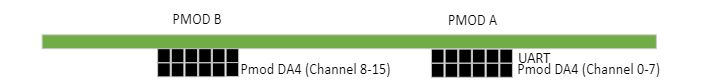
1. Hardware Description
   1. Eclypse Z7 Board / Pmod ports
   2. Pmod DA4
   3. UART Interface
2. Software Description
   1. Required packages
   2. Script overview
   3. User instructions

**Hardware Description**

**Eclypse Z7 board and Pmod ports**

The main board for the device is the Eclypse Z7 board which, among other things, contains a Xilinx Z-7020 FPGA and 4 “Pmod” ports. The Pmod ports are located at the edge of the board. The ports are labelled “PMOD A” and “PMOD B” on the board each are 2 rows of 6 GPIO ports.

These Pmod ports will be used for the UART communication between the FPGA and the CPU, as well as for connecting to the Pmod DA4 parts. PMOD A is used for the UART communication (top row) and one of the Pmod DA4 parts (bottom row). PMOD B is used only for one of the Pmod DA4 parts (bottom row), although another could be plugged in to the top row.



**Pmod DA4**

The Pmod DA4 is an eight channel, 12-bit resolution DAC. The primary IC for the part is the AD5628. The data sheet can be found [here](chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https:/www.analog.com/media/en/technical-documentation/data-sheets/AD5628_5648_5668.pdf?_ga=2.81451287.1891390565.1673221710-1176562786.1665952048). The most relevant information can be found on pages 22-26. In summary, the Pmod is connected to the FPGA with a SPI interface to write commands from the FPGA to the Pmod. For this part, data cannot be read back from the Pmod. The FPGA controls the part by sending commands such as power on/off, write to register, update DAC register, and more. The pmod has 6 pins for the SPI interface that need to be plugged into the board. As shown above, plug one pmod each into the bottom rows of PMOD A and B if not already connected.

The output of the Pmod DA4 looks identical to PMOD A and B and are the DC outputs from the DAC. Each Pmod has 8 DC channels labelled A-H. In the software implementation, the DC channels are labelled 0-15. The pmod plugged into PMOD A maps to channels 0-7, where channel A is channel 0 and channel H is channel 7. Likewise, the pmod plugged into PMOD B is channels 8-15. Again, channel 8 is mapped to channel A and so on. The pmod also has two VCC and two GND outputs.

A picture containing diagram

Description automatically generated

**UART Interface**

The UART interface provides communication between the FPGA and the CPU. The UART currently supports reading a writing 32 bits of data from a 16 bit address space. Messages have the following format:  
Write: “WAAAADDDDDDD

Read : RAAAADDDDDDDD

The first character of the message is either W or R as a Unicode character (i.e send 0x57 for a W). Next is the 16 bit address, shown above in hexadecimal form (each A represents 4 bits). Finally the 32 bit data, also shown in hexadecimal form.

The write commands are all that is needed in order to control the 16 DC DAC channels. There is a specific address for each channel of the DAC, as well as a few extra address for specific commands.

The value of bits 3 to 5 of the address corresponds to different commands.

|  |  |
| --- | --- |
| Value (bits 3-5) | Command |
| 0 | Selects PMOD A Pmod (Channels 0-7) |
| 1 | Selects PMOD B Pmod (Channels 8-15) |
| 4 | Selects ALL channels |
| 6 | Powers on/off ALL channels |

If a specific Pmod is selected (bits 3-5 value is 0 or 1), then bits 0-2 of the address select which channel of the Pmod to update. A value of 0 updates channel A, 1 channel B, and so on.

Once a channel is selected with the address, the voltage is set to the desired value using the bottom 12 data bits of the UART message.

The UART is connected to the FPGA via a USB-UART cable. The USB-UART cable has three wires, one black, one white, and one green. The black is GND, the white is Tx, and the green is Rx. The cable should be connected to PMOD A on the FPGA as shown below:

A picture containing text

Description automatically generated

**Software Description**

**Required Packages**

The following packages must be installed on the host machine to run the python script:

1. Serial
2. Time
3. PySimpleGUI

**Script Overview**

The python script generates a GUI for the user to input voltages to each individual channel. Given a value, the script will generate a UART message as described in the UART Interface section to send to the FPGA. The FPGA receives the UART message, decodes it, and generates a SPI message to send a command to the Pmod and update the DAC channels.

The script has 4 main helper functions:

1. Write: Given an address and data value, writes a single message out on the UART interface.
2. Get\_value: Given a decimal value representing a voltage, a reference voltage, and a resolution, return a binary number with resolution number of bits representing the voltage value based on the reference voltage.
3. Get\_address: Given a channel number, returns the values to be used for constructing the address of the UART message.
4. Update\_channel: Given a channel number and a voltage value in decimal, updates that channel to the voltage using the 3 above functions.

**User instructions**

First, identify the COM3 port of the host PC and connect the USB-UART cable to that port. Alternatively, you can plug in the USB to any port, but you will have to update the script to use that port instead.

Text

Description automatically generated

Ensure all the UART pins are connected to the FPGA and the Pmod DA4 parts are plugged in as described in the Hardware Description sections. Power on the Eclypse Board.

Ensure all required packages are installed on the machine and then simply run the script from any terminal. The following GUI should appear:

A screenshot of a computer

Description automatically generated with medium confidence

To update a specific channel, enter a decimal value into the corresponding box for the voltage that you want and hit Enter Channel X. Alternatively, you can type in the value of all the channels you want to update and hit the Enter button labelled “Update All”. Finally, you can use the Set All Channels box to update all channels to the same value, this is easiest way to reset all channels to 0V.