pFREYA DAQ documentation

Paolo Lazzaroni
µLab
Università degli Studi di Bergamo

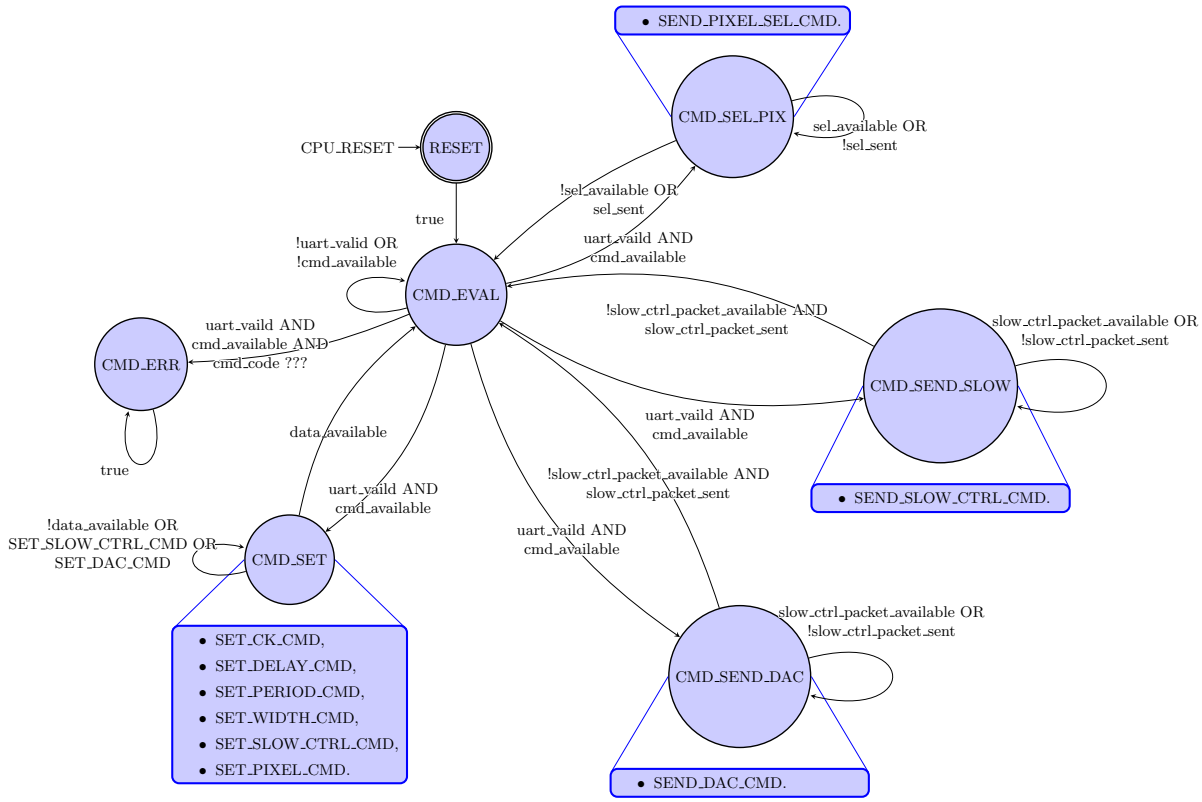
October 23, 2023

Summary

This document provides information on the Data AQuisition (DAQ) system to test the pFREYA16 and pFREYATS ASICs. The system is based on a Xilinx Ultrascale+ FPGA Evaluation Board (KCU116) and it is written in SystemVerilog/Verilog.

1 State machine

This section illustrates the state machine on the FPGA.



2 General directions

The state machine might seem messy, but the fact is that it is just hard to render with that instrument the way the FPGA works. So a few words of clarification. When saying first, second, ... it is intended as "starting from the MSB".

When the FPGA is turned up it should be in RESET state. If a layer of assurance is needed one can push the CPU_RESET button on the FPGA.

Right after the reset is released, the FPGA goes in the CMD_EVAL state, waiting for UART communication. It stays there until a UART communication is available. If the communication is not one of those envisioned, it goes in the CMD_ERR state and stays there forever (or until the CPU_RESET button is pressed).

The UART communication is an 8-bit packet that comes in two flavours: command and data packet. A command to be sent in the UART is composed of an initial 0, to identify a command packet, followed by four bits identifying the command and three bits identifying the signal the command is referring to, if any. A data packet is instead introduced by a 1, followed by 7 bits representing the data content. Please refer to `pFREYA_defs.sv` for which bit sequence corresponds to which command/data.

If the command is one between SET_CK_CMD, SET_DELAY_CMD, SET_HIGH_CMD, SET_LOW_CMD, SET_SLOW_CTRL_CMD, SET_DAC_CMD, and SET_PIXEL_CMD, the FPGA goes in the CMD_SET state. In this state the command and the signal are checked:

- SET_CK_CMD sets a clock, so a signal with 50% duty cycle, and the three following bits represent a clock signal (SLOW_CTRL_CK_CODE, SEL_CK_CODE, ADC_CK_CODE, INJ_STB_CODE, SER_CK_CODE, DAC_SCK_CODE) to be set.
- SET_DELAY_CMD, SET_HIGH_CMD, and SET_LOW_CMD set a fast signal characteristic, that are delay and how long will it stay high or low. The three following bits represent a fast control signal (CSA_RESET_N_CODE, SH_INF_CODE, SH_SUP_CODE, ADC_START_CODE, and INJ_START which is not used right now).

After one of these commands, the FPGA expect a data packet transmission, representing a time length expressed in FPGA clocks. The FPGA clock is a 200 MHz clock generated by the SYSTEM CLOCKs (P and N) running at 300 MHz through a clock wizard instance. However, since each event is triggered on the rising edge of the base clock, the fastest signal attainable has a frequency of 100 MHz (200 MHz divided by 2, which is triggering just on the rising edge).

The data packet expected are composed of 7-bit binary data representing the number of FPGA clocks that the signal property (period, delay, high, or low) has to be set to. Be mindful that the clocks are set by period, while fast signal are set by high and low, which is different. The data packet received set the proper divider with which the counter for each property is checked in order to commute the state of the wanted signal.

The commands SET_SLOW_CTRL_CMD and SET_DAC_CMD still end up in the CMD_SET state, but need more than a 7-bit data to work with. Specifically for the SET_SLOW_CTRL_CMD a 122-bit data is needed (rounded up to 128-bit in FPGA registers, where the MSB are not interesting), while for SET_DAC_CMD a 24-bit data is needed (rounded up to 32-bit in FPGA registers, where the MSB are not interesting). Both the sub-state expect a different kind of data packet: the first bit identifies again the data packet, and it is set to 1, while the second bit is either 1 or 0, if this is the last packet or not respectively. The effective number of bits available for data is therefore 6. The SET_SLOW_CTRL_CMD expect 19 data packets, while SET_DAC_CMD needs 4 of them. Be mindful of setting the right second bit for each packet. The data are stored in FPGA registers and once ready can be sent through the proper command. Please refer to the ASIC documentation of the DAC documentation for the meaning of each bit. A summary is also done in the `pFREYA_defs.sv` file.

The command SET_PIXEL_CMD is yet another exception. It needs a signal, representing either row or column of the pixel to be readout. The data packet is the same as those in the clocks or fast controls, but represents the number of the pixel to be selected. The selection is then sent through the proper command.

The commands SEND_SLOW_CTRL_CMD and SEND_DAC_CMD are used to forward the content of the respective register to the ASIC and the DAC on the pFREYA mother board. They both work in the same way, meaning like a SPI communication. Basically the bits are sent in a serial way, with a clock together with them to provide the time basis on the shift registers in the ASIC or the DAC. A signal enabling the communication is also provided.

Finally, the command SEND_PIXEL_SEL_CMD sends the pixel selection to the ASIC, based on the pixel row and col set. The pixel selection is done by moving a high state in a shift register, therefore a number of clock equal to the pixel row or col to be selected is generated by the FPGA, together with the signal enabling the communication.