Slow Control Documentation

Martim Rosado
martim.rosado@cern.ch

August 30, 2022

Contents

1 Motivation

This IP was designed to route slow control transactions from the backend to the frontend of HGCAL. These transactions target two ASICs: the lpGBT and the GBT-SCA. The transactions consist in read/write operations to the internal registers of either. The main purpose of these transactions is the configuration and monitoring of the several ASICs used in the frontend.

This IP was designed to be used in the final HGCAL system in which there is the need to interface a large number of ASICs in the frontend per backend FPGA. However, due to its architecture, it can also be adapted to the use in smaller test systems through the use of pre-synthesis parameters. Hence, upon instatiation is necessary to configure the IP regarding the system in use and number of target ASICs.

2 Description

Due to the high number of ASICs that need to be interfaced, the Slow Control is made of independent small blocks each attatched to its own protocol (lpGBT or SCA). Each small block is able to drive a certain number of frontend ASICs (default lpGBT 16 | SCA 40). This allows for a certain level of parallelism to be attained that will allow for a faster configuration of the frontend.

Each small block is equipped with BRAM buffers that store the data of both the transactions to send to the frontend and the respective received responses. Besides the buffers, each small block is able to decode the transactions and send the expected data to the respective ASIC to interface.

The Slow Control IP has two AXI interfaces to communicate with the software. One AXI Full (axi) that acesses the BRAM buffers and allow transaction data to be written and read. One AXI Lite (SC_AXI_L) that allows to write and read control and status registers that control the flow of data in the IP.

The software procedure to use this IP is the following for each small block in use:

1. Store the transactions to send to the frontend on the BRAM buffers.
2. Define how many transactions need to be executed.
3. Assert start bit. The IP will start sending the transactions stored in the BRAM buffers to the frontend. Afterwards, the repplies from the frontend will be stored in BRAM buffers as well.
4. Wait until busy bit is deasserted.

5. Read the responses from the BRAM buffers and analyse the data.

6. Repeat as many times as needed.

The outputs of the Slow Control to the frontend are 2 signals that consist of a concatenation of several 2 bit streams each corresponding to an ASIC on the frontend (lpGBT or SCA). One of such signals corresponds to the lpGBTs and the other to the SCAs. Complementary inputs should be connected to the Slow Control in order to trasmit the frontend responses.

These signals cannot be directly connected to the GT of the FPGA. They must be connected to an lpGBT framer that will encode lpGBT frames, usually lpGBT firmware (lpGBT FPGA). It should be noted that each 2 bit stream in each signal corresponds to an lpGBT IC or EC stream. Hence, per lpGBT firmware, a maximum of 2 such streams may be connected (1 for IC, the other for EC). To connect more streams, instantiate more lpGBT firmware.

3 IP Generic Parameters

The Slow Control has some generic parameters that allow, pre-synthesis, to adapt the number of generated small blocks to the number of ASICs to drive. Each small block has its own independent control and status registers, and BRAM buffers.

Parameter Name	**Description**
N_SCA_CTRL	Number of SCA blocks to generate.
N_SCA_CH	Number of SCAs each SCA block can drive. (40)
N_IC_CTRL	Number of lpGBT blocks to generate.
N_IC_CH	Number of lpGBTs each lpGBT block can drive. (16)
N_MEM	Number of memories. MUST be 2*IC_CTRL+2*SCA_CTRL

4 Ports

Port Name	**Description**
clk40_i	40 MHz clock. Related with GT clock
sca_tx_o	Output data from the SCA. Connect to lpGBT fw.
ic_tx_o	Output data from the lpGBT. Connect to lpGBT fw.
sca_rx_o	Input data from the SCA. Connect to lpGBT fw.
ic_rx_o	Input data from the lpGBT. Connect to lpGBT fw.
axi	signals of the AXI Full. Access to BRAMs.
SC_AXI_L	signals of the AXI Lite. Access to control/status registers.

The width of the input output ports to connect to the lpGBT firmware will vary regarding how many small blocks are being instantiated. Per small block, a 2 bit stream will be provided for the input (rx signal) and another 2 bit stream will also be provided for the output signal (tx port) for each ASIC that block can drive (16 lpGBT | 40 SCA). If in the target design it is not needed to use all the 2 bit streams, is ok to leave the unused ones unconnected.

5 Software Interface

The BRAM buffers of a Slow Control small block, from the AXI perspective, have 4096 32 bit addresses. Each 4 of these addresses form a 128 bit word that has information corresponding to 1 transaction. So, it is possible to launch a maximum of 1024 transactions per lpGBT/SCA small block at a time.

5.1 lpGBT

In each lpGBT transaction it is possible to write or read up to 8 consecutive registers on a lpGBT. In each transaction the IP waits for the lpGBT response before starting the next one (except in case in a timeout). It is also possible to broadcast the same transaction to several lpGBTs. However in that case, the response will be read from only one. Moreover, use the address "0000000" as the chip address of the lpGBT when broadcasting.

The TX format to write on the lpGBT TX Buffer is the following:

Bit Mask	**Description**
127:112	Broadcast Address.
111:100	Reserved.
99:96	Reply Address of the lpGBT. 0x7 corresponds to lpGBT 8/16.
95:88	Chip address of the lpGBT[7:1] + operation type R/W̃[0].
87:72	Address of the 1st lpGBT register to read/write.
71:68	Reserved.
67:64	Number of bytes/registers to read/write. Up to 8.
63:0	Bytes to write.

Each bit of the broadcast address corresponds to 1/16 lpGBTs. The least significant bit of this field will correspond to the least significant 2 bit stream that is outputed by the small block. To send a transaction to an lpGBT, assert its respective bit on the broadcast address (possible to assert more than one for each transaction).

The reply address field allows to select which 2 bit stream is selected to read the frontend response among the possible 16. 0x0 corresponds to the least significant 2 bit stream.

The RX format to be expected on the lpGBT RX Buffer is the following:

Bit Mask	Description
127:125	Error Flags. See below.
124:120	Reserved.
119:112	Chip address of the lpGBT[7:1] + operation type R/$\tilde{W}$[0].
111:104	lpGBT Command. Last bit parity ok.
103:92	Reserved.
91:88	Number of bytes/registers to read/write. Up to 8.
87:72	Address of the 1st lpGBT register to read/write.
71:64	Parity Word from the lpGBT.
63:0	Written/Read bytes.

When a transaction fails, only the error flags will be written in the RX buffer. The lpGBT error flags are the following:

Bit	Description
127	Timeout. When active, no reply was received.
126	When active, a write transaction was specified with no data to send.
125	When active, error in broadcast address. No target ASIC was selected.

5.2 SCA

In each SCA transaction, the IP waits for the SCA response similarly to the lpGBT. The TX format to write on the SCA TX Buffer is the following:

Bit Mask	Description
127:88	Broadcast Address. Each bit corresponds of 1/40 SCAs.
87:74	Reserved.
73:68	Reply Address of the SCA. 0x7 corresponds to SCA 8/40.
67	Reserved.
66:64	Command ID. See below.
63:56	SCA address.
55:48	Transaction ID.
47:40	SCA channel address.
39:32	Command.
31:0	Data.

The broadcast address and reply address fields behave in the same way as in the lpGBT TX format.

The Command ID field should follow the following specification:

Bit Code	Description
000	Reserved.
001	Send Connect CMD to SCA.
010	Send Reset CMD to SCA.
011	Reserved.
100	Send Start CMD to SCA.
101	Reserved.
110	Reserved.
111	Reserved.

Use a start command for all transactions except resets and connects.

The RX format to be expected on the SCA RX Buffer is the following:

Bit Mask	Description
127:83	Reserved.
82:80	Error Flags. See below.
79:72	SCA address.
71:74	Control.
63:56	Transaction ID.
55:48	SCA channel address.
47:40	Nbr of bytes in Data.
39:32	Error.
31:0	Data.

When a transaction fails, only the error flags will be written in the RX buffer. The SCA error flags are the following:

Bit	Description
82	Timeout. When active, no reply was received.
81	When active, error in Command ID field. Don't use reserved value.
80	When active, error in broadcast address. No target ASIC was selected.

6 Aditional Information

The transactions start when a positive edge is detected on the start bit of the control registers. Hence, it is advised to deassert the start bit after its assertion

on the begininng of sending transactions to the frontend.

7 Register Map

7.1 Control and Status Registers Addressing

Inside the address space of the SC_AXI_L interface, the addresses are divided as follows from 0 to 2*N_SCA_CTRL+2*N_IC_CTRL-1:

1. 1st Quarter: SCA Control registers 0 to N_SCA_CTRL-1.
2. 2nd Quarter: lpGBT Control registers 0 to N_IC_CTRL-1.
3. 3rd Quarter: SCA Status registers 0 to N_SCA_CTRL-1.
4. 4th Quarter: lpGBT Status registers 0 to N_IC_CTRL-1.

Each register in each quarter corresponds to a Slow Control small block.

7.2 Control and Status Registers Data Format

7.2.1 Control Registers

1. Bit 0: Start bit. Starts transactions on positive edge detection.
2. Bit 1: Enable. Unconnected.
3. Bits 12 downto 2: Number of transactions to perform. At the start of the procedure to send transactions, this register needs to have the number of transactions to perform up to date regarding the number of transactions written on the BRAM buffers.

7.2.2 Status Registers

1. Bit 0: Negated Timeout. If 0, some transaction failed due to timeout.
2. Bit 1: Busy. If 1, IP is processing ongoing transactions. Don't change BRAM contents until Busy is 0.
3. Bit 2: zero_cmd. Indicates if upon start the number of transactions was zero. If so, change the number of transactions to a valid value and try again.
4. Bits 12 downto 3: Number of successful transactions.

7.3 BRAM Buffers Addressing

Inside the address space of the axi interface, the addresses are divided per BRAM from 0 to 2*N_SCA_CTRL+2*N_IC_CTRL-1 BRAMS. Each BRAM has 4096 32 bit addresses:

1. 1st Quarter: SCA TX buffers 0 to N_SCA_CTRL-1.
2. 2nd Quarter: SCA RX buffers 0 to N_SCA_CTRL-1.
3. 3rd Quarter: IC TX buffers 0 to N_IC_CTRL-1.
4. 4th Quarter: IC RX buffers 0 to N_IC_CTRL-1.

Each BRAM in each quarter corresponds to a Slow Control small block.

8 uHAL XML Generation

When using uHAL to access this IP it is needed to provide XML files describing the accessible memory positions. Such files are already provided in the repo for the default values of 1 small block for lpGBTs and 1 small block for SCAs.

Should the number of small blocks change, the XMLs need to be updated. To ensure an automatic generation it is just needed to compile the file XML-TemplateGenerator.c and run the executable. The arguments should be either 0, 1, 2, or 3 followed by N_IC_CTRL followed by N_SCA_CTRL. The first number corresponds to the XML to generate. 0 will generate the status and control register mapping and 1 the BRAM mapping. Copy the output to the target XML.

Should the contents of the control and status registers change, that also needs to be reflected on the XMLs. Change the C file accordingly and compile. To generate the XML of the control interface use the first argument 2 and for status use the last option, 3.

9 EMP Serenity Usage

To use the Slow Control in Serenity without AXI interfaces don't use the IP in its packaged form. Instead use the top module slow_control_top.vhd. This file will provide the Slow Control small blocks with the interfaces to memory buffers and control/status registers without the buffers and regiters themselves (that are connected to an AXI interface).

10 Developer Additional Notes

In the VHDL, each small block is separated from its buffers and control/status registers. In this separated form, each small block comprises three components:

1. The transactor (lpGBT or SCA),
2. The engine (lpGBT or SCA), and
3. The xpoint.

The engine (IC engine for lpGBT and SCA engine for SCA) is the data processing part of this circuit. Is it provided by the lpGBT group and changes should be avoided (at the point of writing, only a FIFO has been altered in the lpGBT engine). The engines translate data into the 2 bit streams needed to interface the ASICs and translate back such stream when processing a response from the frontend.

The transactors drive the respective engines and control the data flow inside each block. Mainly a state machine, the transactor accesses the BRAM buffers and decodes each transaction in order to drive the engine. It is also the transactor's responsability to enconde in the RX frames the information received from the engine once a response reaches the IP and the subsequent write of that data on the RX BRAM buffer.

The xpoint multiplexes the output of the engine to the correct 2 bit output stream and multiplexes back the stream of the response to the input of the engine. The xpoint also ensures that each stream sends the idle patern of the corresponding ASIC when valid data is not present.

10.1 To upload changes to Main BE DAQ

It is needed to update the Slow Control fw on the main BE DAQ when a bug is corrected or a feature is added. Due to some differences regarding the hierarchy for the test systems and the main BE DAQ, the changes should be done manually. Don't update the files slow_control_top.vhd and bus_addr_decoder.vhd. Files present on the Slow Control and absent on the main BE DAQ should also not be added.