CONV-TTL-BLO HDL specifications

Carlos Gil Soriano
BE-CO-HT
carlos.gil.soriano@cern.ch

February 13, 2012

Abstract

This document tackles with:

- HDL development priorities
- Memory mapping

Contents

1 HDL schema

The following schema is used as a reference for the HDL development:

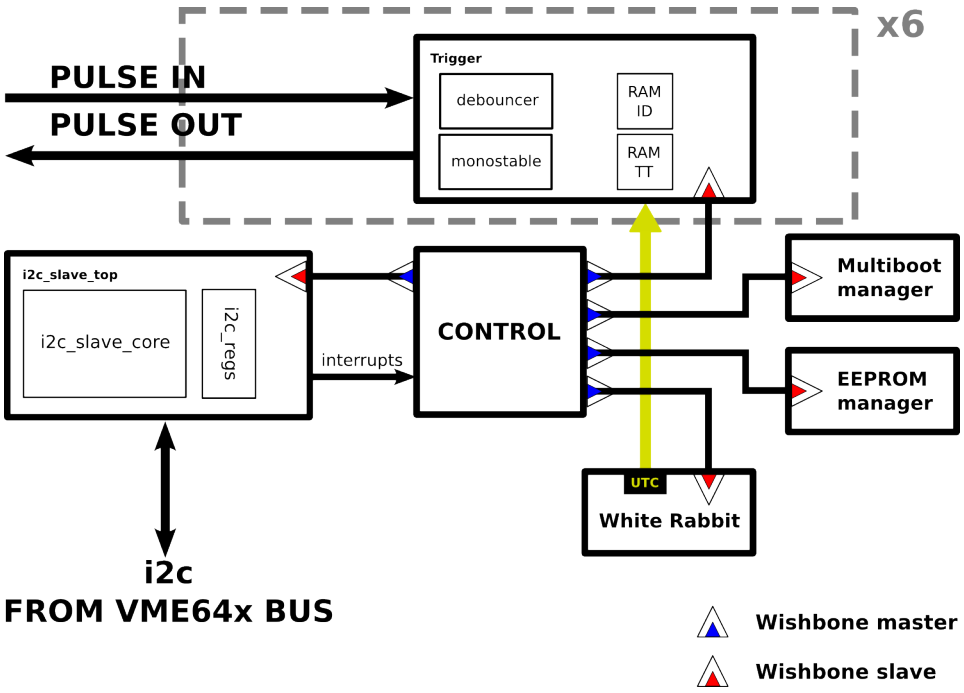

Figure 1: CONV-TTL-BLO HDL structure

1.1 Control

It is the part that bridges the I2C frames to the correct wishbone module. The tasks it is responsible of are:

- Correctly power-up the rest of the modules.
- Provide connectivity of all the wishbone registers via I2C. It manages control access to the registers.

1.2 I2C slave

An I2C slave is needed to receive the frames from the VME64x SERA and SERB pins in P1 connector. This module will communicates with *control hdl module* in the following fashion:

- It connects as a wishbone **slave** to control hdl core and provides interrupt lines for data reception and transmission.

The main reason of implementing a wishbone slave is that by dividing data reception -*i2c slave*- from control access -*control*- the development is more reliable and clear.
This module offers configuration registers to ease the task of data assembly –for instance, the communication schema in ELMA crate.

1.3 Trigger

The trigger manages the pulse repetition. The parameters handled that affect the pulse repetition are:

- **Debouncing** stages of the input pulse.
- **Pulse length** of the output pulse according to Standard Blocking definition.
- **Minimun spacing between pulses** to let the magnetizing current of the transformer be drained off. It is also refered in other documents as *Inactivity timeout upon output pulse is outputted*.

The pulses must be time-tagged. It can be achieved either by a lossy time-tagging via i2c, or with a precise one via White Rabbit. Every time-tag has appended event identifiers (metadata).

1.3.1 Time-tagging Format

The format of the time-tags should be defined. At this moment, an implementation with 96 bits for timestamping with 32 bits of metadata is the default. A record of the last 256 time-tags per channel is hold in the FPGA.

1.4 Multiboot manager

The task of the *Multiboot manager* is to manage a golden bitstream and another one to update the FPGA from. From within this module a FPGA reprogramming command is issued.

1.5 EEPROM manager

The module responsible to write into the EEPROM, read it back and reprogramming the memory module. It should be targeted to interface directly with a MICRON M25P32-VMF6P memory. It will be able to write the MAC address that will be used by White Rabbit and block memory parts of the EEPROM.

1.6 White Rabbit core

Provides precise timestamping.

2 HDL development milestones

The HDL development is scheduled to tackle with several milestones:

1. *i2c slave* module: verification
2. *Control* module: reset the rest of HDL cores. Bypassing i2c instrucctions to wishbone interface.
3. *Multiboot manager* module: multiboot between different precharged bitstreams.
4. *EEPROM manager* module: write a bitstream through the i2c interface.
5. Integration of White Rabbit core: fine timestamping

3 Wishbone memory map

So as to access the devices thanks to *CTDAH control* the following memory map is proposed:

NUMBER	**DEVICE**	**FIRST WISHBONE ADDRESS**	**LAST WISHBONE ADDRESS**
0	Control	0x0100	0x01FF
1	I2C slave	0x0200	0x02FF
2	Trigger 1	0x0300	0x03FF
3	Trigger 2	0x0400	0x04FF
4	Trigger 3	0x0500	0x05FF
5	Trigger 4	0x0600	0x06FF
6	Trigger 5	0x0700	0x07FF
7	Trigger 6	0x0800	0x08FF
8	Multiboot manager	0x0900	0x09FF
9	EEPROM manager	0x0A00	0x0AFF
10	White Rabbit core	0x0B00	0x0BFF
11	EEPROM memory	0x1000	0x1FFF