VME64x I^2C to Wishbone bridge

Theodor-Adrian Stana
CERN, BE-CO-HT

June 4, 2013

Contents

1 Introduction **3**

2 ELMA I^2C Protocol **3**

3 Implementation **3**

1 Introduction

This document describes the *vme64x_i2c* module, an I^2C to Wishbone bridge for the VME64x crates. The module implements an I^2C slave and translates the protocol defined by ELMA in [1] into Wishbone accesses to a Wishbone slave device of choice.

2 ELMA I^2C Protocol

WRITE THIS THING

3 Implementation

In order to perform low-level I^2C transfers, the *i2c_slave* module is instantiated and used within the *vme64x_i2c* module. The outputs of the *i2c_slave* module REFERENCE? are used as controls for an eight-state finite state machine (FSM), shown in Table 1. When the *i2c_slave* module finishes a transfer (signaled by a *done_p_o* pulse), the status is checked and if it is as expected (e.g., a *address good* in the *ST_IDLE* state), the FSM advances to the next state.

It should be noted that where the SysMon appears in the state names, it indicates what the SysMon action is. That is, if the state of the FSM is *ST_SYSMON_WR*, this means the SysMon is writing and the CONV-TTL-BLO is reading.

To better understand how the FSM operates, BYTE ORDER FIGURES can be consulted, where each I^2C transfer is correlated to the current state of the FSM. When reading from the SysMon, the *vme64x_i2c* module will wait in the *ST_IDLE* state while the I^2C address is sent, then go to the *ST_WB_ADR* state to shift in the address. A Wishbone transfer is simulated and if the address exists (a Wishbone *ack* is received), the first byte is shifted in while in the *ST_OP* state, followed by the next three bytes while in the *ST_SYSMON_WR* state. After that, the register is written to in the *ST_SYSMON_WR_WB* state.

When writing to the SysMon, the first few steps are the same as for reading from it. The address is shifted in and checked in the Wishbone transfer simulation state. In the case of a SysMon reading from a board, however, the I^2C transfer is restarted and the order is reversed (SysMon starts reading). Thus, while in *ST_OP*, the FSM detects a different value of *op_o* and goes into the *ST_SYSMON_RD_WB* state. Here, an actual Wishbone read transfer is executed, the value of the register is read and sent via I^2C in the
ST_SYSMON_RD state.

Table 1: *vme64x_i2c* state machine

State	Description
ST_IDLE	Wait for the *i2c_slave* module to receive the I^2C address and go to *ST_WB_ADR*. The starting value at the *op_o* output of the *i2c_slave* module is stored for checking in *ST_OP*
ST_WB_ADR	Shift in the two address bytes sent via I^2C and go to *ST_SIM_WB_TRANSF*
ST_SIM_WB_TRANSF	Start a Wishbone read transfer from address received in previous state and go to *ST_OP* if Wishbone address exists (Wishbone *ack* received, or *ST_IDLE* otherwise (Wishbone *err* received)
ST_OP	Check the *op_o* output of the *i2c_slave* module. If different from the value at the start, go to *ST_SYSMON_RD_WB* state (SysMon is reading from CONV-TTL-BLO), otherwise continue shifting in bytes (SysMon writing to CONV-TTL-BLO)
ST_SYSMON_WR	Continue reading up to four bytes sent by the SysMon and go to *ST_SYSMON_WR_WB*
ST_SYSMON_WR_WB	Perform a Wishbone write transfer to the register with the address obtained in *ST_WB_ADR*
ST_SYSMON_RD_WB	Perform a Wishbone read transfer from the address obtained in *ST_WB_ADR* and go to *ST_SYSMON_RD*
ST_SYSMON_RD	Shift out the four bytes of the Wishbone register when the *i2c_slave* module successfully finishes a write

References

[1] ELMA, "Access to board data using SNMP and I2C." `http://www.ohwr.org/documents/227`.