VME64x I^2C to Wishbone bridge

Theodor-Adrian Stana
CERN, BE-CO-HT

June 5, 2013

Contents

1 Introduction

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

2 ELMA I^{2}C Protocol

Using the I^{2}C lines on the VME P1 connector, one can access boards placed in a VME crate. In this purpose, ELMA has defined a higher-level protocol [1] that uses I^{2}C as a low-level protocol.

Fig. 1 shows a write operation from the SysMon to a VME board. The process starts with the control byte, containing the board's I^{2}C slave address and the read/write bit cleared, indicating an I^{2}C write. After the slave's ACK, the following two bytes send the 12-bit address in little-endian order (most significant byte first). After the address has been acknowledged, the following four I^{2}C transfers are used to transmit the 32-bit data to be written to the board register. Data transmission occurs with the least significant byte first (big-endian).

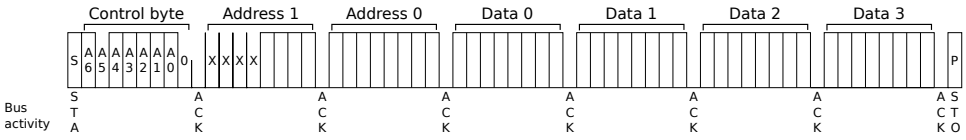

Figure 1: SysMon write operation

A read transfer (Fig. 2)from a VME board is similar to the write transfer. The differences lie in the retransmission of the control byte after the register address, this time with the read/write bit set, to indicate an I^{2}C read. Following the ACK from the slave, the transfer direction changes and the SysMon will read the four data bytes sent by the VME board. As with the write transfer, the data bytes are sent by the VME board in big-endian order.

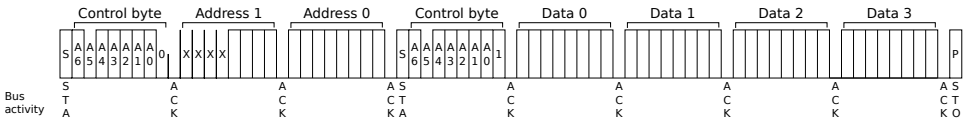

Figure 2: SysMon read operation

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), a simplified version of which is shown in Fig. 3. Table 1 also lists the states of the state machine.

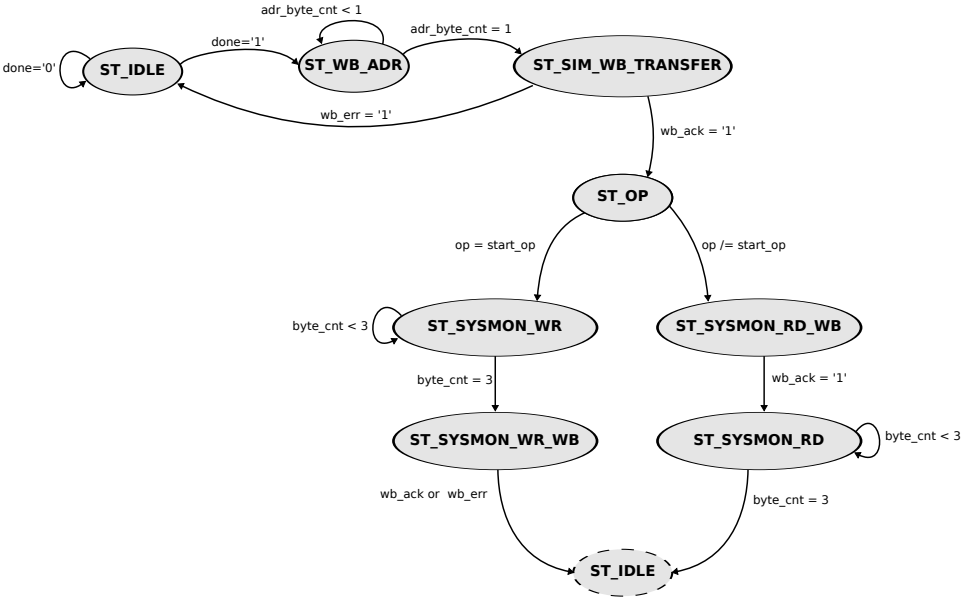

Figure 3: Main FSM of *vme64x_i2c* module

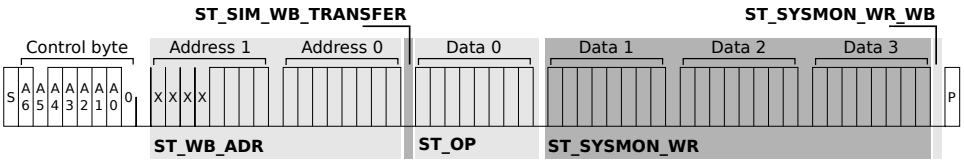

Figure 4: FSM states when the SysMon writes to the *vme64x_i2c*

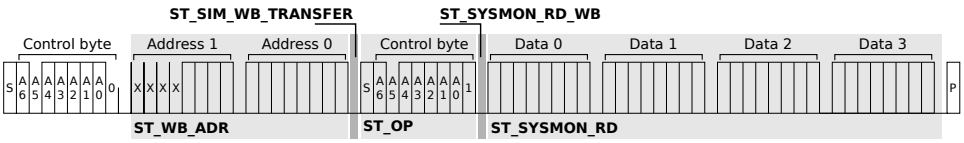

Figure 5: FSM states when the SysMon reads from the *vme64x_i2c*

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. For example, if the state of the FSM is *ST_SYSMON_WR*, this means the SysMon is writing and the *vme64x_i2c* is reading.

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 *vme64x_i2c*), otherwise continue shifting in bytes (SysMon writing to *vme64x_i2c*)
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

To better understand how the FSM operates, Figures 4 and 5 can be consulted, where the state of the FSM is shown during reads and writes from the SysMon.

When reading from the SysMon (Fig. 4), 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 (Fig. 5), 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.

References

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