Logic Organization for the STAR Trigger Control Unit – Mark 4

E.G. Judd
May 24, 2013

This document contains block diagrams for all the logic implemented in the daughter card of the STAR Trigger Control Unit (TCU). It also contains a detailed description of how the VME communications path through the motherboard to the daughter card has been implemented. This is the 2nd version of the daughter card and therefore the 4th version of the TCU logic.

Contents

Top Level Logic Diagram

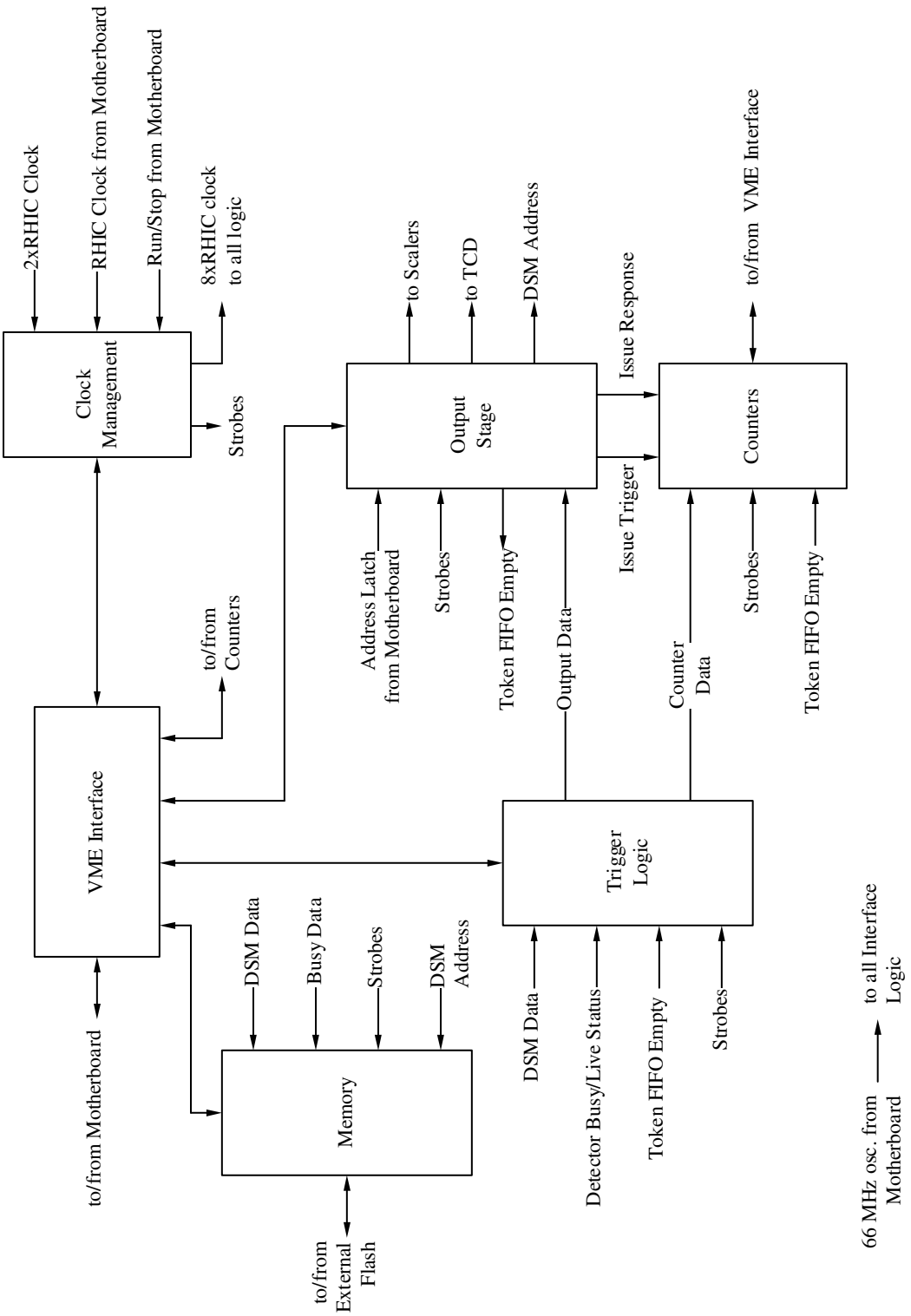

Figure 1: Top Level Logic Diagram

Clock Management Logic

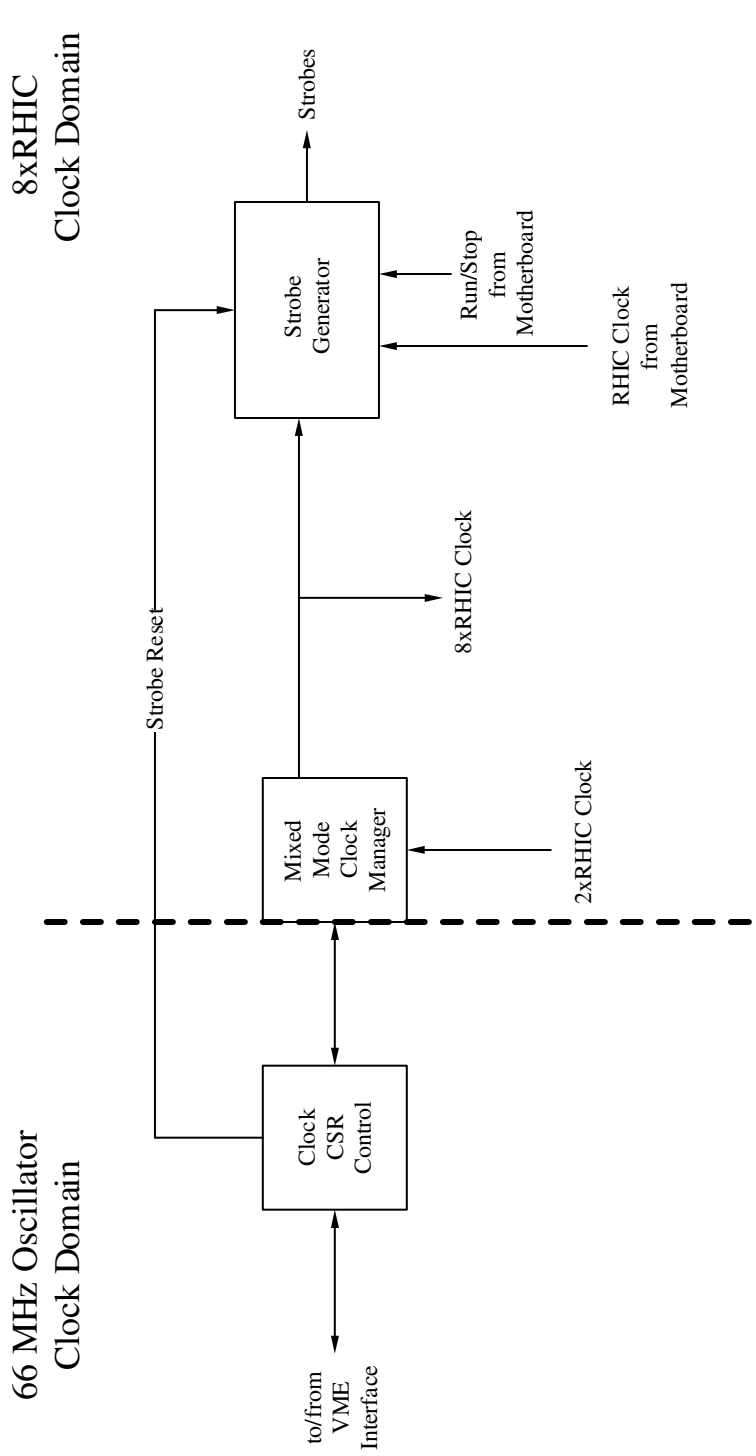

Figure 2: Clock Management Logic

Trigger Logic

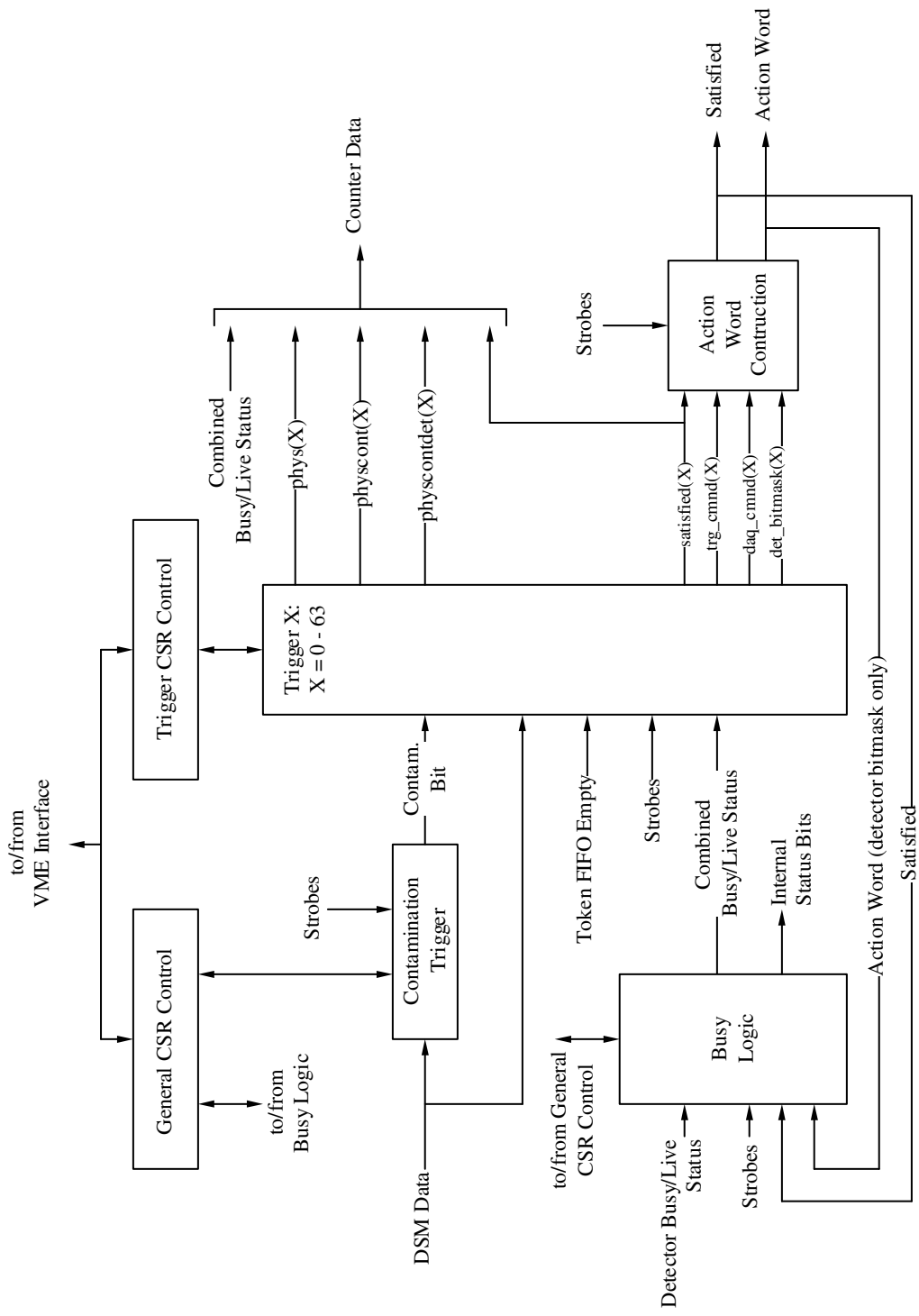

Figure 3: Trigger Logic

Output Stage Logic

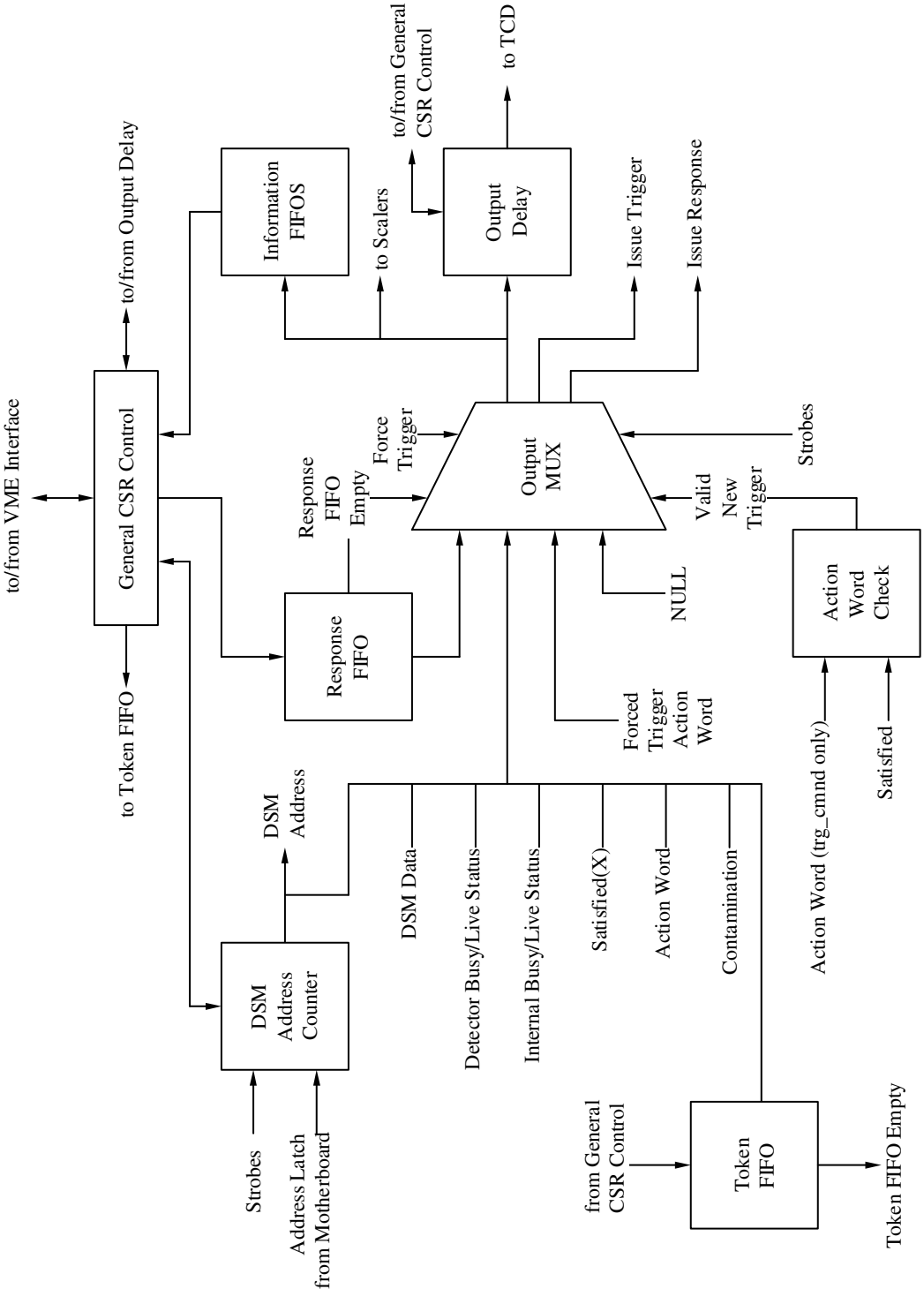

Figure 4: Output Stage Logic

Counter Logic

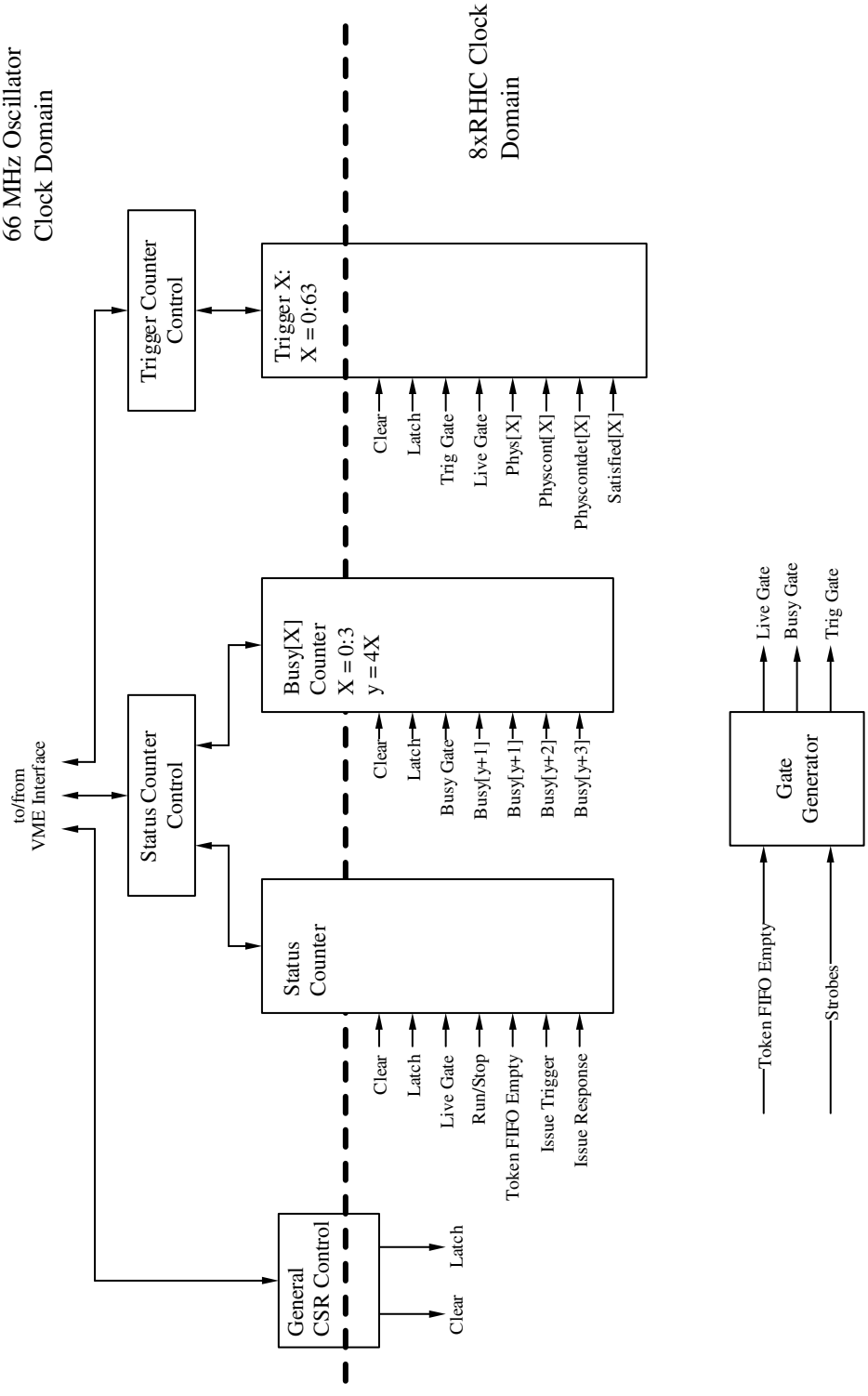

Figure 6: Counter Logic

Input / Output Memory Logic

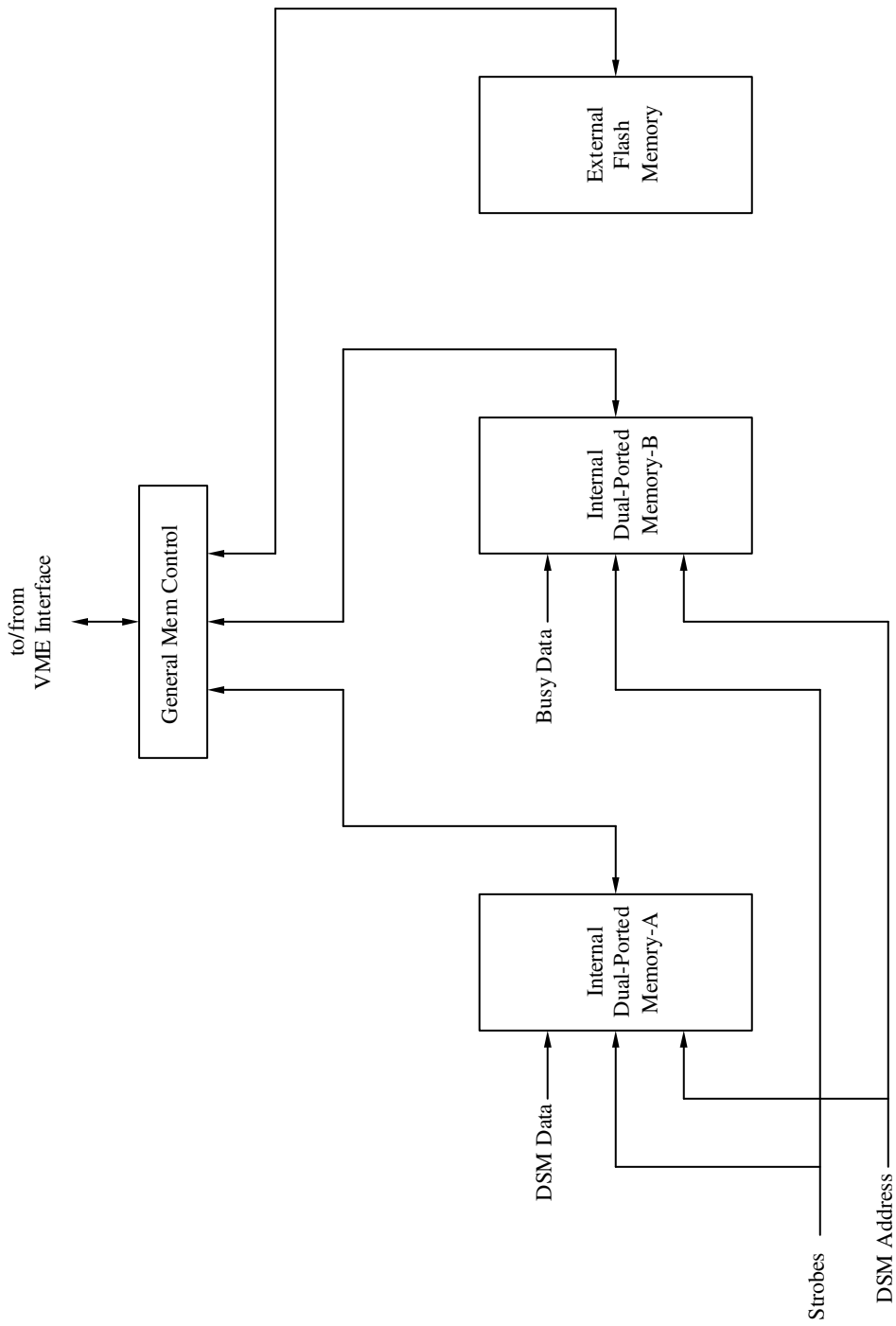

Figure 7: Input and Output Memory Logic

VME Interface Logic

The communications path is quite complex. VME communications involve separate address and data buses as well as a large number of control signals. The daughter card only has a single external bus available to carry both address and data, and a limited number of control pins. Internally however the daughter card FPGA has enough space to implement separate address and data buses again. Procedures have been developed to allow these three domains to connect properly so an external VME Master can communicate with the FPGA in the TCU daughter card.

Communication with the TCU4 daughter card goes through an FPGA on the motherboard. The communications logic in the motherboard FPGA is split into the two modules shown in Figure 8. One module receives and processes all of the VME address, data and control signals. The other module deals with the control signals and the single multiplexed bus that connects to the daughter card. The actions taken in each state are shown in the following table.

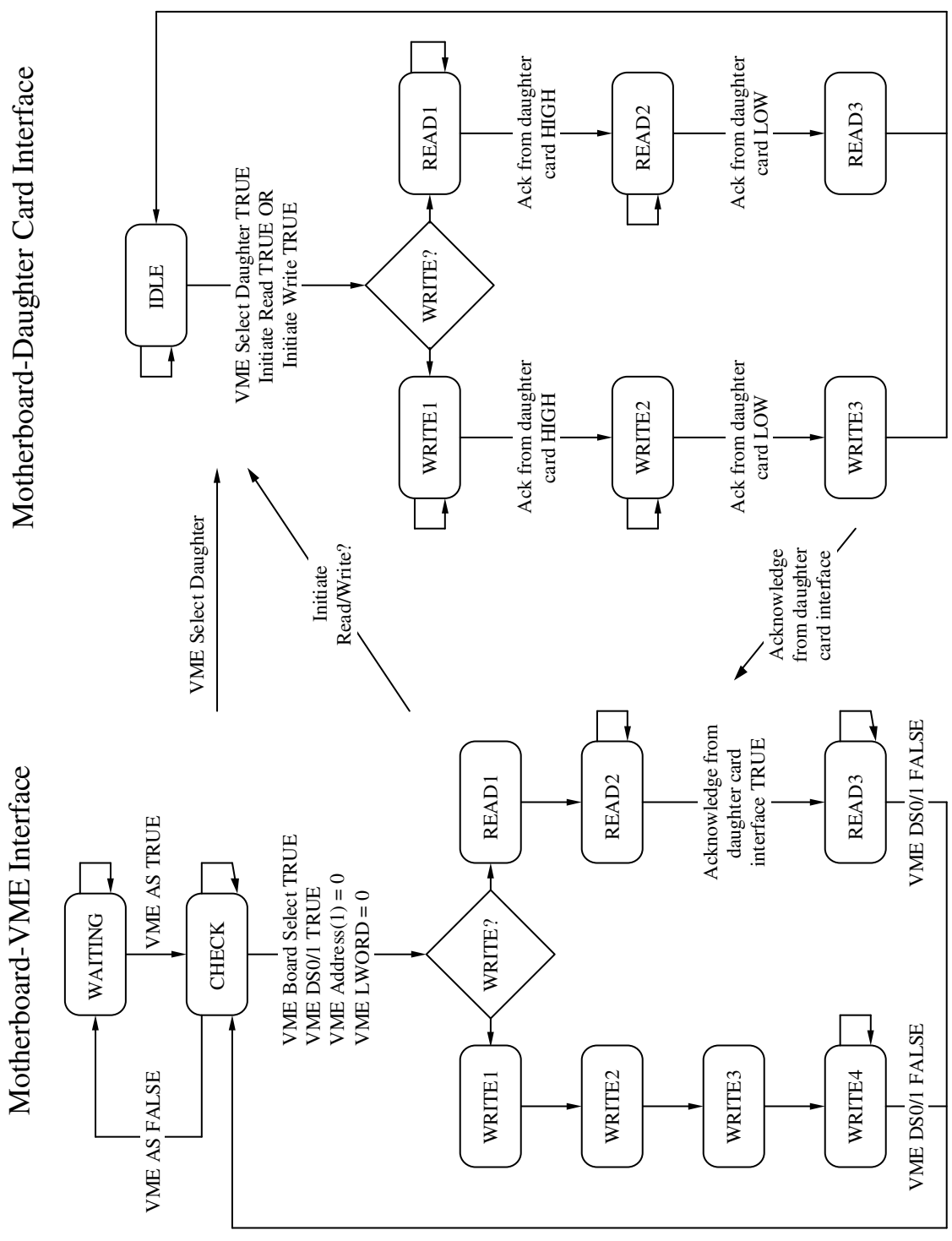

Figure 8: State Machines for the TCU Motherboard Communications Modules.

Module	State	Action
Motherboard-VME Interface	WAITING	Latch in VME address and control signals. Disable external VME Data Bus drivers.
	CHECK	Decode upper address bits to select daughter card or internal motherboard registers.
	READ1	Generate initiate_read pulse
	READ2	Enable external VME Data Bus drivers Drive data received from daughter to VME Data Bus
	READ3	Continue to drive data to VME Data Bus Also drive VME DTACK low (TRUE).
	WRITE1	Enable external VME Data Bus drivers
	WRITE2	Latch data received from VME Data Bus to internal bus
	WRITE3	Generate initiate_write pulse
	WRITE4	Drive VME DTACK low (TURE)
Motherboard-Daughter Card Interface	IDLE	Allow multiplexed bus to float Drive daughter card control signals low Drive ack_n to VME interface high (FALSE)
	READ1	Drive VME address on multiplexed bus Drive SYN high to indicate the start of an access
	READ2	Allow multiplexed bus to float so daughter card can drive its data on the bus. Drive BUS_DIR high to tell daughter card that bus is available Receive that data from the daughter card
	READ3	Continue to receive data from daughter card Drive SYN and BUS_DIR low to tell daughter card that its data has been received Drive ack_n to VME Interface low (TRUE)
	WRITE1	Drive VME address on multiplexed bus Drive SYN and RW high to indicate the start of an access
	WRITE2	Drive data from internal bus on to multiplexed bus Drive SYN and RW low to tell the daughter card that the data is available
	WRITE3	Drive ack_n to VME Interface low (TRUE)

Timing diagrams for VME read and write accesses to the daughter card are shown next in Figure 9. The absolute times are not correct but the relative timing between transitions in the control signals and the multiplexed bus are correct.

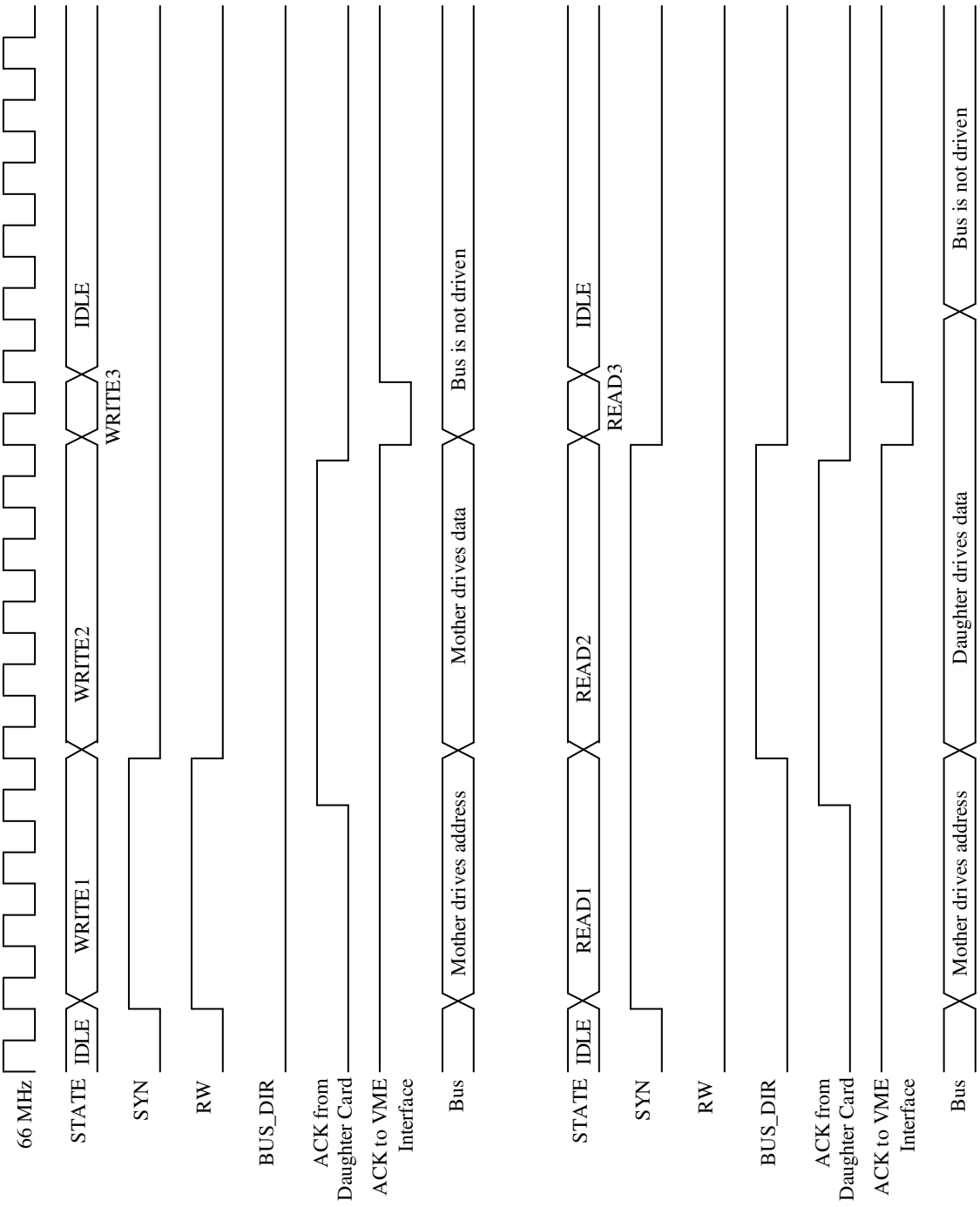

Figure 9: Timing Diagrams for the Communications Bus connecting the TCU Motherboard to the Daughter Card.

In this diagram “STATE” corresponds to the states of the Motherboard-Daughter Card Interface state machine shown in Figure 8. The 66 MHz clock along with SYN, RW and BUS_DIR are all driven from the Motherboard to the Daughter Card. The Daughter Card sends back an acknowledge signal.

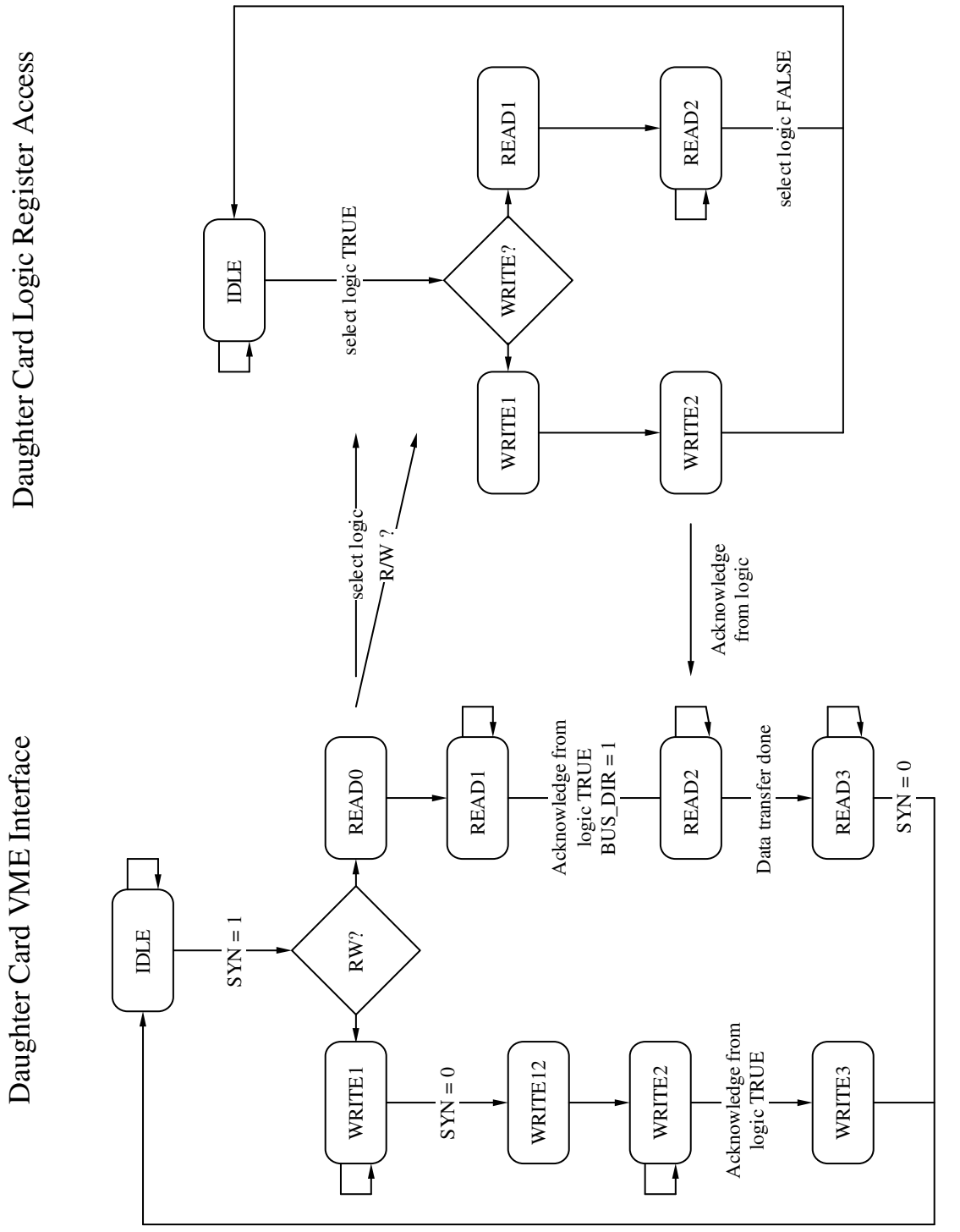

Figure 10: State Machines for the TCU Daughter Card Communications Modules.

At the other end of the link the daughter card communication logic is also split into multiple blocks. The first block, labeled VME Interface in the Top Level Logic Diagram on page 2,

receives the control signals from the motherboard, deals with the multiplexed bus, decodes the address and handles communication to the rest of the logic. The state machines are shown in Figure 10.

Each of the other logic blocks then has a sub-module that handles read and write access to their registers. These modules are typically named “General CSR Control”, or something similar. All these blocks are small variations on the same logic, and their state machine is shown here.
The actions taken in each state are shown in the following table:

Module	State	Action
Daughter Card VME Interface	IDLE	Drive Acknowledge to motherboard low Set internal R/W strobe to 0 (READ) Latch data from multiplexed bus onto internal address bus
	READ0	Decode address and select logic module
	READ1	Drive Acknowledge to motherboard high
	READ2	Continue to drive Acknowledge high Drive data from selected logic module onto multiplexed bus. Wait long enough for data to become stable on the bus.
	READ3	Drive acknowledge to motherboard low Continue to drive selected data onto multiplexed bus
	WRITE1	Drive Acknowledge to motherboard high Set internal R/W strobe to 1 (WRITE)
	WRITE12	Latch data from multiplexed bus onto internal data bus Decode address and select logic module
	WRITE2	Wait for acknowledge from logic module
	WRITE3	Drive acknowledge to motherboard low
Daughter Card General CSR Control	IDLE	Drive Acknowledge to VME Interface low Set internal read/write flags to 0
	READ1	Set internal read flag to 1
	READ2	Drive Acknowledge to VME Interface high Set internal read flag to 0 Decode address and drive selected data to VME Interface
	WRITE1	Drive Acknowledge to VME Interface high Set internal write flag to 1
	WRITE2	Continue to drive Acknowledge high Set internal write flag to 0 Decode address and set selected register value to data received from VME Interface n internal data bus.