VME64x Core Specifications

May 19, 2010

1 Introduction

This document provides an overview of the features VME64x slave supports. All implemented functionalities conform to the standards defined by ANSI/VITA VME64x Standard [1] [2], but not all are implemented. It also provides an overview of the default power-up configuration and configuration procedure.

The implementation follows the design rules set by Design rules for custom VME modules in CMS [3].

2 System clock

Due to several timing requirements and constraints 100 MHz system clock is suggested. If the core is driven with a clock of a lower frequency, erroneous data in 2eSST transfers might occur. Higher frequencies are in conflict with implementation timing constraints and are therefore not advised.

3 VME64x features

This chapter lists and explains features that VME64x slave implements.

3.1 CR/CSR space

For the sake of a “plug and play” capability CR/CSR space is implemented as defined by ANSI/VITA Standards for VME64 Extensions [2].

In order to provide “plug and play” capability VME64x provides a mechanism very similar to PCI. A dedicated “Configuration ROM / Control & Status Register” (CR/CSR) address space has been introduced. It consists of ROM and RAM regions with a set of well defined registers. It is addressed with the address modifier 0x2F in the A24 address space.

Every VME module occupies a 512 kB page in this address space. The location of this page in the A24 space is defined by geographical address lines on the backplane: each slot is provided with a unique geographical five bit address at the J1 connector (row d). From these bits A23...A19 of the CR/CSR page are derived. If the geographical address is not available (GA parity bit does not match), base address is set to 0x00, which indicates a faulty condition.

The CR/CSR space can be accessed with the data width D08(EO). Please note that in compliance with the CR/CSR definition, only every fourth location in the CR/CSR space is used.

In addition to the standard CSR space, user CSR space is also implemented and it contains IRQ Status/ID registers. User CSR space is located in address space from 0x7FBFF to 0x7FBE7.

Configuration ROM (CR) and Configuration RAM (CRAM) are implemented externally. Designers should see to it that data contained in the CR depicts the correct VME64x core specifications as presented in this document.

The layout of the configuration space is depicted in Figure 1. The location of the user defined CR and CSR regions as well as the CRAM region are programmable. For each of these, six bytes defining the start and the end address (with respect to the start of the configuration space) are reserved in the CR region. Designers are free to use these regions for module specific purposes. Dedicated software must deal with the content. If a user defined region is not implemented the start and end address must be set to 0x000000.

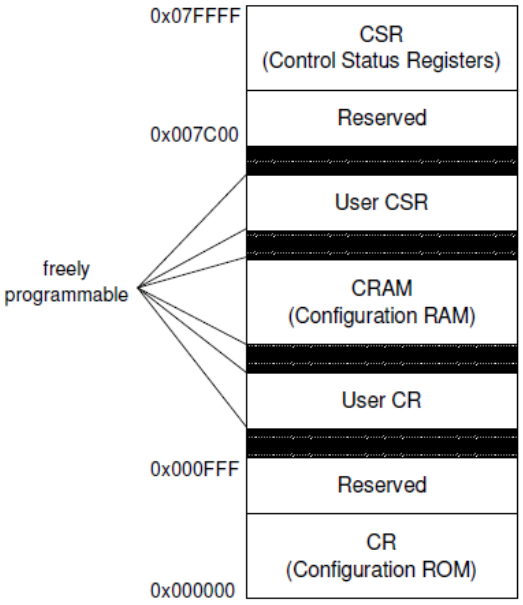

Figure 1: Configuration space organization in VME64x

Please refer to ANSI/VITA VME64 Extensions [2] for further information on the CR/CSR space.

3.2 Data types

This VME64x core supports D08(OE), D16, D32, D64 and unaligned data transfers. The implementation assumes that the target data memory is 8-bit wide. Upon D16 access, only every other byte is addressed, upon D32 every fourth and upon D64 data access only every eighth byte is addressed. Designers who do not need the 8-bit granularity and would prefer bigger data widths can choose to ignore least significant bits of the local address.

3.3 Addressing types

Table 1 list supported addressing types with their address modifier codes:

Mnemonic	AM code (hex)	Description
A24	3D	24-bit addressing
A24_BLT	3B	24-bit addressing block read/write (limited to 256 cycles)
A24_MBLT	3C	24-bit addressing multiplexed block read/write for D64 data access
A24_LCK	32	24-bit addressing ADOH lock cycle
A16	2D	16-bit addressing
A16_LCK	2C	16-bit addressing ADOH lock cycle
A32	0D	32bit addressing
A32_BLT	0F	32-bit addressing block read/write (limited to 256 cycles)
A32_MBLT	0C	32-bit addressing multiplexed block read/write for D64 data access
A32_LCK	05	32-bit addressing ADOH lock cycle
A64	01	64-bit addressing
A64_BLT	03	64-bit addressing block read/write (limited to 256 cycles)
A64_MBLT	03	64-bit addressing multiplexed block read/write for D64 data access
A64_LCK	04	64-bit addressing ADOH lock cycle
CR_CSR	2F	24-bit addressing for CR/CSR access

Table 1: Supported addressing types

Lock ADOH (address only with handshake) cycles are used to lock out the addressed resource for the period of the current VME bus grant (during which BBSY signal is low).

3.4 2e transfers

In addition to the addressing and transfer types listed in Table 1, two edge transfers are also supported (2eVME and 2eSST transfers). Address modifier 0x20 is used for these transfers.

With 2e transfers, master supplies the address along with some additional data in three address phases as depicted in Table 2.

Signal Line	Address Phase 1	Address Phase 2	Address Phase 3	Data Phase
AM[5:0]	0x20	0x20	0x20	0x20
A[7:0]	XAM Code	A[3:0] = 0 Device Address A[7:4]	Reserved	D[39:32]
A[15:8]	Device Address A[15:8]	Beat count for 2eVME transfers Cycle Count for 2eSST transfers	Reserved	D[47:40]
A[23:16]	Device Address A[23:16]	A[23:21] = 0 A[20:16] = GA of Master	Reserved	D[55:48]
A[31:24]	Device Address A[31:24]	Subunit Number in Master	Reserved	D[63:56]
D[31:0]	Device Address A[63:32] (= 0 for A32)	D[3:0] = Transfer Rate for 2eSST D[4] = Odd Bit for 2eSST D[31:5] = Reserved	Reserved	D[31:0]

Table 2: 2e address cycles

Address bits 7 down to 0 (note: signal LWORD is regarded as address bit 0) in the first address phase carry an extended address modifier or XAM. Table 3 shows the supported XAMs.

Extended Address Modifier Code (XAM)	Address/Data Mode
0x01	A32/D64 2eVME
0x02	A64/D64 2eVME
0x11	A32/D64 2eSST
0x12	A64/D64 2eSST

Table 3: Supported extended address modifiers

Please note, that in order to use these types of transfer, ADER registers in CSR space have to be configured accordingly.

3.5 Signals

This section focuses on functionality of certain VME bus signals.

3.5.1 RESET

RESET resets the entire core to the default configuration.

3.5.2 BERR

BERR signal is used to signal a bus error. A transfer cycle is terminated with assertion of this signal if the VME64x slave does not recognize the data or addressing type used in the transfer cycle, if master attempts to write to a read-only memory (CR) or if error is received from the module which is addressed.

3.5.3 RETRY

RETRY signal terminates the transfer cycle if VME64x slave receives a retry request from the addressed module (via the WishBone bus), signaling that the read/write request cannot be completed at this time.

3.6 Interrupts

Interrupt controller is a ROACK type controller.

Upon synchronously detecting a rising edge on the interrupt request signal input, the VME64x core drives the IRQ request line on the VME bus low thus issuing an interrupt request. VME master acknowledges the interrupt in a form of an IACK cycle. After the interrupt is acknowledged, the VME IRQ line is released.

There are seven VME IRQ lines but only one interrupt request input. For the purpose of configuring which of the seven IRQ lines the VME64x core will drive (in response to a rising edge on the IRQ input), an IRQ Level register has been implemented in the user CSR space. The value of this register corresponds to the number of the IRQ line on the VME bus that is to be used (note that on the VME master side priorities are taken into an account, IRQ7 having the highest priority and IRQ1 the lowest). If the IRQ level register is set to 0x00 or values above 0x07, interrupts are disabled (which is also the default power-up and reset configuration).

After each IACK cycle, 8-bit Status/ID register is presented on the data bus by the VME64x slave. IRQ Status/ID register is located in the user CSR space and can be written or read by the VME master as any other register in the CSR space.

Register	Address
IRQ Status/ID	0x7FBFF
IRQ Level	0x7FBFB

Table 4: IRQ-related registers memory mapping

4 Configuration

Upon power-up or reset, module is disabled and only its CR/CSR space can be accessed. Software must then first map the module memory in the 64-bit address space by setting Address Decoder Compare (ADER) registers in CSR, which, together with Address Decoder Mask (ADEM) registers in CR relocate the module memory to the desired address range. Please note that since 64-bit address space is supported, two consecutive ADEMs and ADERs form address relocation for one “function”, so designers can implement up to four different memory relocations, but addressing whichever one of them will trigger a memory request to the module. ADER for each function also contains an AM or XAM code to which it responds.

After the module has been placed in the desired address space, it can be enabled by writing into Bit Set Register in the CSR and thus setting the correct enable bit.

5 VME bus transceivers

The VME64x slave core also includes output signals that drive external hardware transceivers. These signals are DTACK OE, DATA DIR, DATA OE, ADDR DIR and ADDR OE.

Direction (DIR) signals specify the direction of data transmission. For MOSI (master out, slave in) directed data DIR is low and for MISO (master in, slave out) directed data DIR is high.

Output enable (OE) signals are used to disable the transceivers so that the buses are effectively isolated. Transceiver is disabled when OE is low.

6 WishBone master

This section describes the functional operation of the WishBone (WB) master core.

6.1 Single, BLT and MBLT transfers

When the core is addressed with these types of transfers, WB master operates in accordance with official WB specifications document [5].

6.2 2eSST and 2eVME transfers

When the VME64x core is addressed with a 2eVME or 2eSST transfer the WB master assumes a slightly modified way of operation.

WishBone master component modification for use with fast 2eVME and 2eSST transfers is based on the WishBone specifications [5] with a few changes that add support for pipelined WB transfers. This section will only specify these differences, while the basic WB operation is extensively explained in the official WB specifications document [5].

For WB slaves to achieve greater burst read/write speeds, address (and data for writes) is clocked to the WB slaves with STB signal without waiting for the slave to acknowledge each cycle with asserting ACK signal. This enables the dataflow to be pipelined and when the pipeline is full, slaves begin to acknowledge each of the received address/data (along with putting valid data on the bus for reads). Only after the number of ACK pulses matches the number of previously issued STB pulses will the WB master terminate the block transfer (with CYC going low). It must be assured that WB slaves will acknowledge the appropriate number of cycles under all circumstances.

For the purpose of reporting to the master that the pipeline cannot receive new data at the moment, STALL signal is introduced. Master will not clock any address/data while the STALL signal is high.

Designers should note that even though RTY and ERR signals are supported (and are propagated to the VME bus in a form of RETRY and BERR signals), the VME master will not receive the information carried by these signals in “real-time” because the VME64x slave core and the WB master core function independently.

7 I/O ports

Port name	Direction	Comment
clk_i	in	System clock for the entire core
VME_AS_n_i	in	VME bus signal, for additional info see [1]
VME_RST_n_i	in	VME bus signal, for additional info see [1]
VME_WRITE_n_i	in	VME bus signal, for additional info see [1]
VME_AM_i (5:0)	in	VME bus signal, for additional info see [1]
VME_DS_n_i (1:0)	in	VME bus signal, for additional info see [1]
VME_GA_i (5:0)	in	VME bus signal, for additional info see [1]
VME_BERR_n_o	out	VME bus signal, for additional info see [1]
VME_DTACK_n_o	out	VME bus signal, for additional info see [1]
VME_RETRY_n_o	out	VME bus signal, for additional info see [1]
VME_LWORD_n_b	in/out	VME bus signal, for additional info see [1]
VME_ADDR_b (31:1)	in/out	VME bus signal, for additional info see [1]
VME_DATA_b (31:0)	in/out	VME bus signal, for additional info see [1]
VME_BBSY_n_i	in	VME bus signal, for additional info see [1]
VME_IRQ_n_o (6:0)	out	VME bus signal, for additional info see [1]
VME_IACKIN_n_i	in	VME bus signal, for additional info see [1]
VME_IACKOUT_n_o	out	VME bus signal, for additional info see [1]
CRaddr_o	out	CR address bus
CRdata_i (7:0)	in	CR data bus
CRAMaddr_o (18:0)	out	CRAM address bus
CRAMdata_o (7:0)	out	CRAM data out bus
CRAMdata_i (7:0)	in	CRAM data in bus
CRAMwea_o	out	CRAM write enable
VME_DTACK_OE_o	out	Signal for driving external buffers (described in ch. 4)
VME_DATA_DIR_o	out	Signal for driving external buffers (described in ch. 4)
VME_DATA_OE_o	out	Signal for driving external buffers (described in ch. 4)
VME_ADDR_DIR_o	out	Signal for driving external buffers (described in ch. 4)
VME_ADDR_OE_o	out	Signal for driving external buffers (described in ch. 4)
RST_i	in	WB bus signal, for additional info see [5]
DAT_i (63:0)	in	WB bus signal, for additional info see [5]
DAT_o (63:0)	out	WB bus signal, for additional info see [5]
ADR_o (63:0)	out	WB bus signal, for additional info see [5]
CYC_o	out	WB bus signal, for additional info see [5]
ERR_i	in	WB bus signal, for additional info see [5]
LOCK_o	out	WB bus signal, for additional info see [5]
RTY_i	in	WB bus signal, for additional info see [5]
SEL_o (7:0)	out	WB bus signal, for additional info see [5]
STB_o	out	WB bus signal, for additional info see [5]
ACK_i	in	WB bus signal, for additional info see [5]
WE_o	out	WB bus signal, for additional info see [5]
STALL_i	in	Addition to the standard WB signals (described in ch. 5.2)
IRQ_i	in	Interrupt request input

Table 5: VME64x core I/O signals

8 References

[1] ANSI/VITA, American National Standard for VME64, April 1995.
[2] ANSI/VITA, American National Standards for VME64 Extensions, October 1998.
[3] ANSI/VITA, American National Standards for 2eSST, December 1999.
[4] VME64x in CMS, Design rules for custom VME modules in CMS, C.Schwick, CMS IN 2004/005, January 2004. (http://cmsdoc.cern.ch/~cschwick/VME/resources/VMEGuidelines.pdf)
[5] WISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores, Revision: B.3, September 2002