Level-1 Regional Calorimeter Trigger Testing, Setup, and Operation Manual

Pamela Klabbers
pamc@hep.wisc.edu
August 31, 2006

Contents

Acronyms and Abbreviations

Acronym	Definition
ASIC	Application Specific Integrated Circuit
BC0	Bunch Crossing Zero (On schematics this is usually BX0)
BSCAN	Boundary Scan
CCC (CC)	Clock and Control Card (Clock and Control)
ECL	Emitter Coupled Logic
EDC	Bit Error Detection Code, i.e. Hamming Code
EIC (EI)	Electron Identification Card (Electron Identification)
EISO	Electron Isolation
EM	Electromagnetic Calorimeter data
HD	Hadronic Calorimeter data
hex	Hexadecimal
J/S (JSC)	Jet Summary (Jet Summary Card)
LUT	Look Up Table (memory)
RC	Receiver Card
RCT	Regional Calorimeter Trigger
RMC	Receiver Mezzanie Card (Vitesse link)
STC	Serial Test Card
TTL	Transistor-Transistor Logic
Vcc	TTL High = +5.0 VDC (In this use, however in ECL this is ground)
Vtt	ECL low, –2 VDC in RCT circuits

Card Slots and Addresses

Card Location and Type	Base VME Address (hexadecimal)
Clock and Control Card*	10000000
Slot 0 Electron Identification Card	11000000
Slot 0 Receiver Card	12000000
Slot 1 Electron Identification Card	13000000
Slot 1 Receiver Card	14000000
Slot 2 Electron Identification Card	15000000
Slot 2 Receiver Card	16000000
Slot 3 Electron Identification Card	17000000
Slot 3 Receiver Card	18000000
Jet Summary Card*	19000000
Slot 4 Electron Identification Card	1a000000
Slot 4 Receiver Card	1b000000
Slot 5 Electron Identification Card	1c000000
Slot 5 Receiver Card	1d000000
Slot 6 Electron Identification Card	1e000000
Slot 6 Receiver Card	1f000000

*Clock and Control Card and the Jet/Summary may only be plugged into their specific slots during normal RCT crate operation.

The RCT crates use geographical addressing. Addresses for the individual cards are assigned based on their slot in the crate. For each type of card, the sub-addresses are represented as $00 to $08 (hexadecimal) where the $ represents the first 6 hexadecimal digits of a card's address. For example, if the receiver card in slot 3 is programmed and its address is hex 18000000, the address for the card ID should be 18000006. In the case of the memories (LUTs), only the first two digits are dropped, so $440000 would translate to 18440000 for the receiver card example above.

In the following tables, all data written to registers and LUTs is in hexadecimal format.

Card Descriptions, Command Registers and Control

Clock and Control Card

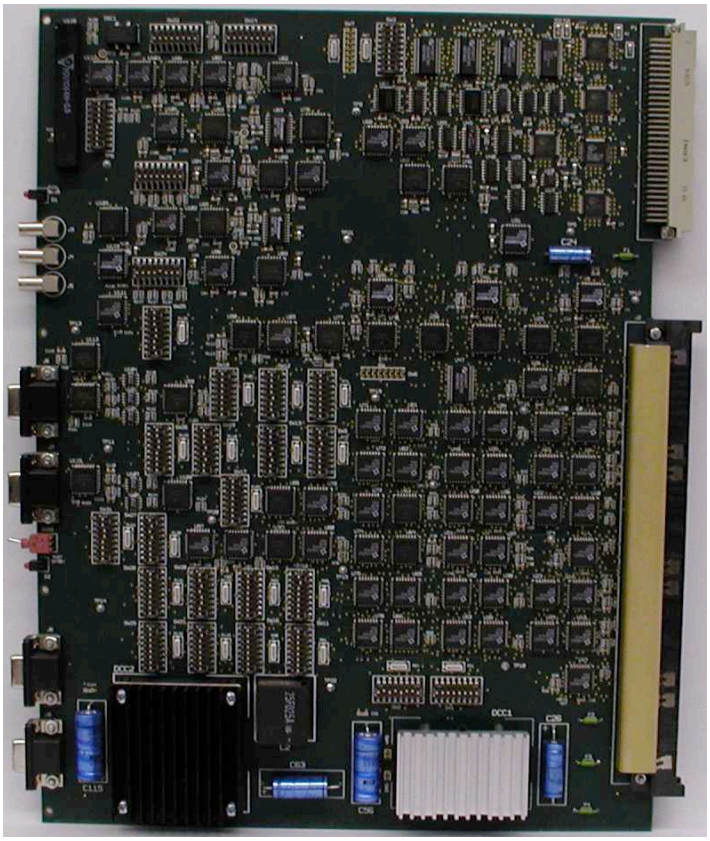


Figure 1: A clock and control card (front side).

The clock and control card distributes the clocks, reset, and bunch crossing zero (BC0) for each RCT crate. It can provide the signals locally, or via two input connectors (fifteen pin VGA) on the front panel. A toggle switch on the front panel switches between the two signal sources. There is also a differential pair of 120 MHz clock outputs (Lemo connectors – ECL signals) for the STC control, and a 40 MHz output for

use with modified STC's when testing the links (again Lemo – ECL). The additional VGA connectors are for the Jet Capture Card, described in a separate document.

The CCC base address is always 10000000. It must always be in the first RCT slot (a 4 column VME connector above a special backplane connector) at the front of the crate (3rd slot from the left).

Card Control $00

Bit	R/W	Name	Description
0	R/W	Card Control	0= Setup – Normal CCC operation
1	R/W	Card Control	5=/STP_CLK
2	R/W	Card Control	6=/SSTP_CLK, 7=/STRT_CLK
3	R/W	CLR_ERR	Set to 1, then to 0 to clear error status
4	R/W	Not Used	
5	R/W	Not Used	
6	R/W	Not Used	
7	R/W	ENB_INT	For VME programming
8	R	ERR	0= No errors, 1=Errors exist
9	--	0	Tied to Ground
10	--	0	Tied to Ground
11	--	0	Tied to Ground
12	--	0	Tied to Ground
13	--	0	Tied to Ground
14	--	0	Tied to Ground
15	--	0	Tied to Ground

Normal Operation: The $00 address should equal 0 for normal operation of the Clock and Control Card.
Clearing Errors: To clear the error status bit (bit 8), first set bit 3 of $00 to 1 and then back to 0.
Single Step Mode: To use the single step mode, first set the number of clock cycles using CCC address $04, and then stop the clock by setting address $00 to 5. Setting address $00 to 6 activates the Single Step Clock for the number of cycles set by address $04. To go back to normal operation, start the clock by setting address $00 to 7 and the resetting the value back to 0.
Errors: Bit 8 is an OR of the individual status bits coming from the cards. The status bits cannot be cleared at the CCC, but must be cleared at the individual cards, however they can be read out at address $1000. RC's and the JSC are the only cards with errors, the EIC does not have any. Only the Link and EDC (Hamming Code) errors are sent on, the Compare Phase is not.

Crate Wide Signal Control $02

Bit	R/W	Name	Description
0	R/W	LCL_ACPT	1=Send an Accept
1	R/W	SEL_LCL_ACPT	1=Accept source is local to crate
2	R/W	LCL_BX0	1=Send a Bunch Crossing Zero
3	R/W	SEL_LCL_BX0	1=Bunch Crossing Zero is local to crate
4	R/W	LCL_RESET	1=Send a reset for the clock
5	R/W	SEL_LCL_RST	1=Reset source is local to crate
6	R/W	SYSTEM_RST	1=Send a system reset
7	R/W	ENB_VECT	For VME Programming
8	--	0	Tied to Ground
9	--	0	Tied to Ground
10	--	0	Tied to Ground
11	--	0	Tied to Ground
12	--	0	Tied to Ground
13	--	0	Tied to Ground
14	--	0	Tied to Ground
15	--	0	Tied to Ground

To send a local reset, first set address $02 to 20, then to 60, and back to 20. A similar approach is necessary for the Bunch Crossing Zero and the Level-1 Accept.

Crate Wide Signal Control $04

Bit	R/W	Name	Description
0	R/W	STP_CNT0	Sets the count, in clock cycles, for the Single Step Clock, controlled via address $00.
1	R/W	STP_CNT1	
2	R/W	STP_CNT2	
3	R/W	STP_CNT3	
4	R/W	STP_CNT4	
5	R/W	STP_CNT5	
6	R/W	STP_CNT6	
7	R/W	STP_CNT7	
8	--	0	Tied to Ground
9	--	0	Tied to Ground
10	--	0	Tied to Ground
11	--	0	Tied to Ground
12	--	0	Tied to Ground
13	--	0	Tied to Ground
14	--	0	Tied to Ground
15	--	0	Tied to Ground

Card Identification and VME Vector $06

Bit	R/W	Name	Description
0	R	VECT0	Read value of the Vector – set with SW5 (dip). Necessary for VME programming.
1	R	VECT1	
2	R	VECT2	
3	R	VECT3	
4	R	VECT4	
5	R	VECT5	
6	R	VECT6	
7	R	VECT7	
8	R	Card Id	Eight bits for Card identification, tied to Vcc, Can be strapped to ground. Value is complement of serial number of card – to Ease visual inspection of straps on SW7
9	R	Card Id	
10	R	Card Id	
11	R	Card Id	
12	R	Card Id	
13	R	Card Id	
14	R	Card Id	
15	R	Card Id	

Crate Card Information $08

Bit	R/W	Name	Description
0	R	EISO0_STAT	0= EISO Slot 0 empty, 1= Card in Slot 0
1	R	RCVR0_STAT	0= RCVR Slot 0 empty, 1= Card in Slot 0
2	R	EISO1_STAT	0= EISO Slot 1 empty, 1= Card in Slot 1
3	R	RCVR1_STAT	0= RCVR Slot 1 empty, 1= Card in Slot 1
4	R	EISO2_STAT	0= EISO Slot 2 empty, 1= Card in Slot 2
5	R	RCVR2_STAT	0= RCVR Slot 2 empty, 1= Card in Slot 2
6	R	EISO3_STAT	0= EISO Slot 3 empty, 1= Card in Slot 3
7	R	RCVR3_STAT	0= RCVR Slot 3 empty, 1= Card in Slot 3
8	R	JETSUM_STAT	0= JETSUM Slot empty, 1= Card in Slot
9	R	EISO4_STAT	0= EISO Slot 4 empty, 1= Card in Slot 4
10	R	RCVR4_STAT	0= RCVR Slot 4 empty, 1= Card in Slot 4
11	R	EISO5_STAT	0= EISO Slot 5 empty, 1= Card in Slot 5
12	R	RCVR5_STAT	0= RCVR Slot 5 empty, 1= Card in Slot 5
13	R	EISO6_STAT	0= EISO Slot 6 empty, 1= Card in Slot 6
14	R	RCVR6_STAT	0= RCVR Slot 6 empty, 1= Card in Slot 6
15	--	0	Tied to Ground

Dump this address to find out what cards are inserted. This will work whether or not a card is programmed. The circuitry to handle this is on the backplane.

Crate Card Status Registers

Reading these addresses provides information about the card status:

$10000: This address is for the card status bits, in the order shown for the Crate Card Information address - $08. If a card is in error, its status bit is one. Status bits must be cleared at the individual cards.

$10002: Time of Death Bits 0-15
$10004: Time of Death Bits 16-31
$10006: Time of Death Bits 32-39

These three addresses provide the count, in 40 MHz clock cycles, since the last reset occurred, until an error has occurred. One must clear the error to see a change in the count, not clear the count. Write a 1 to address $00, and then write a 0. Then dump the three addresses to get a count.

Receiver Card

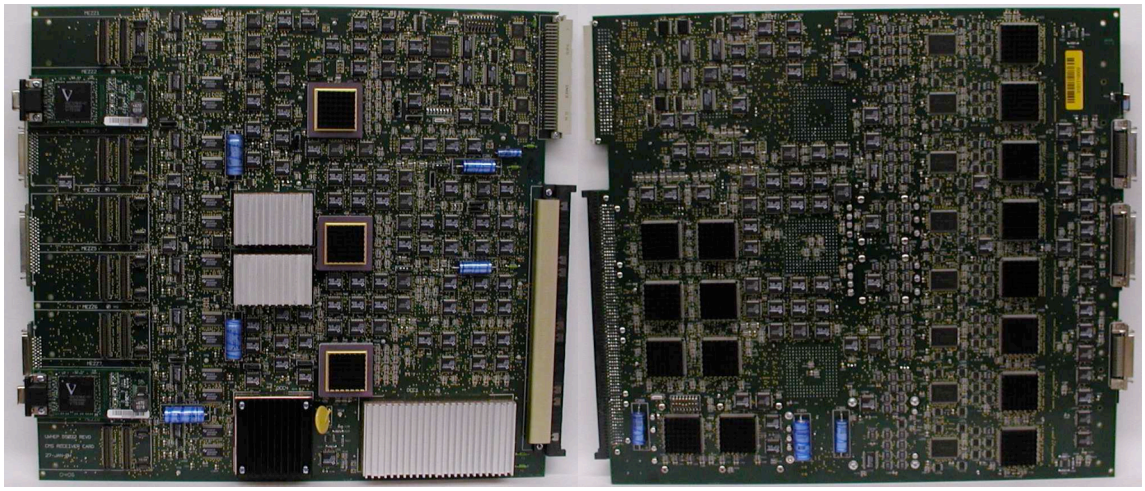


Figure 2: A Receiver Card (front side and back side).

The receiver card receives the trigger primitives via the eight V7216 copper links and handles energy lookup for ECAL and HCAL E_T, region summing, τ veto, and sharing for the e/γ algorithm.

Card Control $00

Bit	R/W	Name	Description
0	R/W	Card mode	0=Read LUTs, 1=Write LUTs,
1	R/W	Card mode	2=Test mode, 4=Run mode,
2	R/W	Card mode	5=Boundary Scan mode
3	R/W	Not Used	
4	R/W	Not Used	
5	R/W	Not Used	
6	R/W	Not Used	
7	R/W	RST_BX0	Raise to 1 and lower to 0 to reset the BX0 Error
8	R	/RDY	BSCAN status
9	R	/BSCAN_INT	BSCAN status
10	--	0	Tied to Ground
11	--	0	Tied to Ground
12	--	0	Tied to Ground
13	--	0	Tied to Ground
14	--	0	Tied to Ground
15	--	0	Tied to Ground

This RC address is important for the overall operation of the card. Setting its value to 0 (hex) will allow one to read out the values written to the LUTs described later in this section. A value of 1, allows writing to the LUTs. Running in test mode, 2, cycles through the first 256 locations in memory using the test vector of the Phase ASIC. Run mode, 4, is for basic operation of the card, i.e. as a regular part of the trigger system, and data will propagate as designed. A value of 5 will put the card in BSCAN mode, used only for running the JTAG self-tests on the cards.

The BX0 error is reset at this address. Set the seventh bit (hex 80) to one and then to zero to reset the BX0 error. Be careful to not disable the function of the card if that is not your intention by resetting the lower bits. The BX0 error may be completely disabled by raising this location to 1 and not returning it to zero.

Error Handling $02

Bit	R/W	Name	Description
0	R/W	RST_ERR0	Set to FF and then to 00 to reset the errors
1	R/W	RST_ERR1	from Phase ASICs 0-7.
2	R/W	RST_ERR2	Readout is at address $04.
3	R/W	RST_ERR3	Keeping a bit value high disables errors.
4	R/W	RST_ERR4	
5	R/W	RST_ERR5	
6	R/W	RST_ERR6	
7	R/W	RST_ERR7	
8	R	CMP_PHS0	1=Error in Compare Phase of Phase ASIC 0-7
9	R	CMP_PHS1	0=No Error in Comp. Phase of Phase ASIC 0-7
10	R	CMP_PHS2	(Compares 120 and 160 MHz Clock Phases -
11	R	CMP_PHS3	Independent of EDC and link errors
12	R	CMP_PHS4	Does not zero data)
13	R	CMP_PHS5	
14	R	CMP_PHS6	
15	R	CMP_PHS7	

Error Handling $04

Bit	R/W	Name	Description
0	R	ERR0	1=error in Phase ASIC 0-7
1	R	ERR1	0=No error in Phase ASIC 0-7
2	R	ERR2	Reset is in address $02
3	R	ERR3	Link or EDC errors possible
4	R	ERR4	(These are controlled via address $06)
5	R	ERR5	Values float if there is no RMC plugged in.
6	R	ERR6	
7	R	ERR7	
8	R	ERR_PHS0	0=No Error on phase of reset and 160 MHz
9	R	ERR_PHS1	1, 2, 3=Error on phase of reset and 160 MHz
10	R	0	Tied to Ground
11	R	0	Tied to Ground
12	R	0	Tied to Ground
13	R	0	Tied to Ground
14	R	BX0_E	0/1=No Error/Error in BX0 for EM Receive
15	R	BX0_H	0/1=No Error/Error in BX0 for HD Receive

Card Position, Error modes, and Card Identification $06

Bit	R/W	Name	Description
0	R/W	CARD_01	1=RC in Slot 0 or 1, 0= RC in slots 2-6
1	R/W	CARD_ODD	NOT USED
2	R/W	CARD_SHR_LR	Enable edge sharing on cables (RC 0 or 1 only)
3	R/W	EDC_ERR	0/1=EDC Errors Disabled/Enabled
4	R/W	LINK_ERR	0/1=Link Errors Disabled/Enabled
5	R/W	R0_UP/DN	Changes order of R0 ϕ's: 0=default, 1=reversed
6	R/W	R1_UP/DN	Changes order of R1 ϕ's: 0=default, 1=reversed
7	R/W	ZERO_DATA	0/1=Disable/Enable zeroing of data for any error
8	R	Card Id	Eight bits for Card identification, tied to Vcc,
9	R	Card Id	Can be strapped to ground.
10	R	Card Id	Value is complement of serial number of card – to
11	R	Card Id	Ease visual inspection of straps on SW4
12	R	Card Id	
13	R	Card Id	
14	R	Card Id	
15	R	Card Id	

Addressing Memory on the Receiver Card

The schematic for block ADRS_DIST of the receiver card provides the address locations for the EM and HD data streams summarized here. The schematic for block TRNSLTRS shows how the addressing is done for the individual LUTs.

Bit	Description
0	Reserved
1	EM E_T Bit 0
2	EM E_T Bit 1
3	EM E_T Bit 2
4	EM E_T Bit 3
5	EM E_T Bit 4
6	EM E_T Bit 5
7	EM E_T Bit 6
8	EM E_T Bit 7
9	EM E_T Bit 8/Fine Grain
10	HD E_T Bit 0
11	HD E_T Bit 1

Bit	Description
12	HD E_T Bit 2
13	HD E_T Bit 3
14	HD E_T Bit 4
15	HD E_T Bit 5
16	HD E_T Bit 6
17	HD E_T Bit 7
18	Sums (0) or e/γ (1) half of mem
19	Bit 0 for specific physical LUT
20	Bit 0 for specific physical LUT
21	Bit 0 for specific physical LUT
22	1 (One) for LUTS
23	Not used for LUTS - BSCAN

The base addresses for the memories are therefore (from LUT 0 to LUT 7):
HD: $400000, $480000, $500000, $580000, $600000, $680000, $700000, $780000
EM: $440000, $4c0000, $540000, $5c0000, $640000, $6c0000, $740000, $7c0000

The full 17 bits (addresses 1-18 above) of LUT address effectively accesses two memory locations in each LUT simultaneously. When writing to the memory, it is necessary to

use the specific base addresses given above to address the different locations. The 18th bit specifies whether you are writing to the hadron (regional sum) location or the e/γ location. The hadron (regional sum) path writes/outputs 10 bits, with 9 bits of E_T and the top bit (200 in hex) the tau bit for use by the tau veto circuit. The e/γ path writes/outputs 8 bits, with 7 bits of E_T and the top bit (80) to be used for the fine grain and H/E "OR".

When in test mode (Address $00, value hex 2) only the first eight bits of each address are cycled. The cycle repeats itself after 256 clocks (equivalent to 64 bunch crossings). Writing patterns to the first 256 locations is all that is necessary for testing. Some programs exist to aid the writing the full memory (>32000 locations). They are described later in the section about booting the crate and initializing the cards since they are system dependent.

Hadron Memory locations for first test "bunch crossing": $XX00YY

Below is the correct orientation for cards 1, 3, 5 in a crate on the positive side of the η mapping. For cards 0,2, and 4 a mirror image along the upper edge defines the memory locations on the positive side of the η mapping. Reflect along the left edge for cards 0, 2, and 4 on the negative η side, and do both reflections for cards 1, 3, 5. Card 6 maps Region 0 as shown below, Region 1 is directly below and the order of the memories changes with respect to η (η_0: XX=78, η_1: XX=70, η_2: XX=68, η_3: XX=60).

Cards 0 to 5

	Region 0				Region 1			
	η_0	η_1	η_2	η_3	η_0	η_1	η_2	η_3
ϕ_0	XX=40 YY=00	XX=48 YY=00	XX=50 YY=00	XX=58 YY=00	XX=60 YY=00	XX=68 YY=00	XX=70 YY=00	XX=78 YY=00
ϕ_1	XX=40 YY=02	XX=48 YY=02	XX=50 YY=02	XX=58 YY=02	XX=60 YY=02	XX=68 YY=02	XX=70 YY=02	XX=78 YY=02
ϕ_2	XX=40 YY=04	XX=48 YY=04	XX=50 YY=04	XX=58 YY=04	XX=60 YY=04	XX=68 YY=04	XX=70 YY=04	XX=78 YY=04
ϕ_3	XX=40 YY=06	XX=48 YY=06	XX=50 YY=06	XX=58 YY=06	XX=60 YY=06	XX=68 YY=06	XX=70 YY=06	XX=78 YY=06

Card 6 Region 0

	η_0	η_1	η_2	η_3
ϕ_3	XX=40 YY=06	XX=48 YY=06	XX=50 YY=06	XX=58 YY=06
ϕ_2	XX=40 YY=04	XX=48 YY=04	XX=50 YY=04	XX=58 YY=04
ϕ_1	XX=40 YY=02	XX=48 YY=02	XX=50 YY=02	XX=58 YY=02
ϕ_0	XX=40 YY=00	XX=48 YY=00	XX=50 YY=00	XX=58 YY=00
ϕ_0	XX=60 YY=00	XX=68 YY=00	XX=70 YY=00	XX=78 YY=00
ϕ_1	XX=60 YY=02	XX=68 YY=02	XX=70 YY=02	XX=78 YY=02
ϕ_2	XX=60 YY=04	XX=68 YY=04	XX=70 YY=04	XX=78 YY=04
ϕ_3	XX=60 YY=06	XX=68 YY=06	XX=70 YY=06	XX=78 YY=06

Card 6 Region 1

Electron Memory locations for first test "bunch crossing": $XX00YY

Below is the correct orientation for cards 1, 3, 5 in a crate on the positive side of the η mapping. For cards 0,2, and 4 a mirror image along the upper edge defines the memory locations on the positive side of the η mapping. Reflect along the left edge for cards 0, 2, and 4 on the negative η side, and do both reflections for cards 1, 3, 5. Card 6 maps Region 0 as shown below, Region 1 is directly below and the order of the memories changes with respect to η (η_0: XX=7c, η_1: XX=74, η_2: XX=6c, η_3: XX=64).

Cards 0 to 5

	Region 0				Region 1			
	η_0	η_1	η_2	η_3	η_0	η_1	η_2	η_3
ϕ_0	XX=44 YY=00	XX=4c YY=00	XX=54 YY=00	XX=5c YY=00	XX=64 YY=00	XX=6c YY=00	XX=74 YY=00	XX=7c YY=00
ϕ_1	XX=44 YY=02	XX=4c YY=02	XX=54 YY=02	XX=5c YY=02	XX=64 YY=02	XX=6c YY=02	XX=74 YY=02	XX=7c YY=02
ϕ_2	XX=44 YY=04	XX=4c YY=04	XX=54 YY=04	XX=5c YY=04	XX=64 YY=04	XX=6c YY=04	XX=74 YY=04	XX=7c YY=04
ϕ_3	XX=44 YY=06	XX=4c YY=06	XX=54 YY=06	XX=5c YY=06	XX=64 YY=06	XX=6c YY=06	XX=74 YY=06	XX=7c YY=06

Card 6 Region 0

	η_0	η_1	η_2	η_3
ϕ_3	XX=44 YY=06	XX=4c YY=06	XX=54 YY=06	XX=5c YY=06
ϕ_2	XX=44 YY=04	XX=4c YY=04	XX=54 YY=04	XX=5c YY=04
ϕ_1	XX=44 YY=02	XX=4c YY=02	XX=54 YY=02	XX=5c YY=02
ϕ_0	XX=44 YY=00	XX=4c YY=00	XX=54 YY=00	XX=5c YY=00
ϕ_0	XX=64 YY=00	XX=6c YY=00	XX=74 YY=00	XX=7c YY=00
ϕ_1	XX=64 YY=02	XX=6c YY=02	XX=74 YY=02	XX=7c YY=02
ϕ_2	XX=64 YY=04	XX=6c YY=04	XX=74 YY=04	XX=7c YY=04
ϕ_3	XX=64 YY=06	XX=6c YY=06	XX=74 YY=06	XX=7c YY=06

Card 6 Region 1

Electron Identification Card

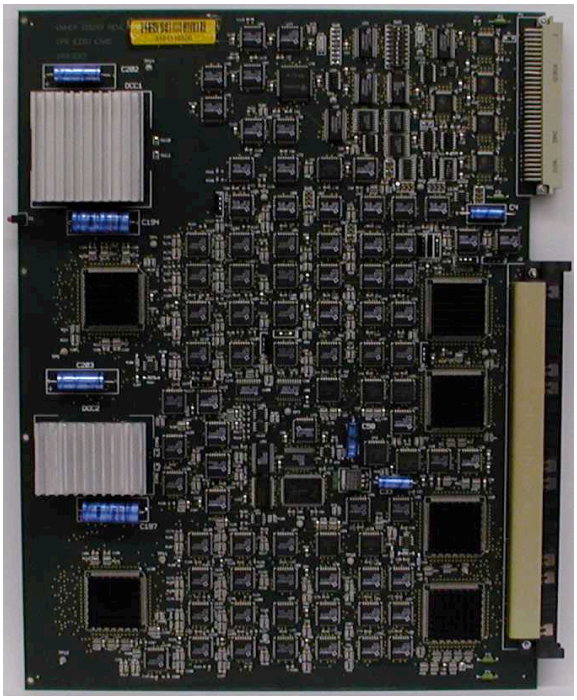


Figure 2: An Electron Identification Card (front side).

The electron identification card receives the 32 direct and 28 shared tower E_T's and fine-grain/H over E vetoes via the backplane from the receiver cards. The two EISO ASICs use these to produce four e/γ candidates per card: one 7-bit isolated and one 7-bit non-isolated e/γ candidates per region of the calorimeter. A memory look up converts these to 6 bit energies and forwards the results via the backplane to the Jet/Summary card.

Card Control $00

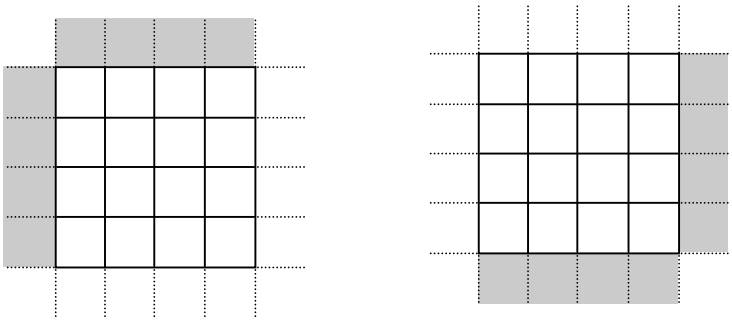
Bit	R/W	Name	Description
0	R/W	Card mode	0=Read memories, 1=Write Memories,
1	R/W	Card mode	4=Run mode,
2	R/W	Card mode	5=Boundary Scan mode
3	R/W	Not Used	
4	R/W	Not Used	
5	R/W	Not Used	
6	R/W	TP_LF_R1	**Card 6 only**
7	R/W	ODD	Card is in slot 1, 3, or 5
8	R	/RDY	BSCAN status
9	R	/BSCAN_INT	BSCAN status
10	--	0	Tied to Ground
11	--	0	Tied to Ground
12	--	0	Tied to Ground
13	--	0	Tied to Ground
14	--	0	Tied to Ground
15	--	0	Tied to Ground

Thresholds, EISO ASIC control, Card Identification $02

Bit	R/W	Name	Description
0	R/W	THRSR0_0	Bit 0 of threshold for Region 0 (EISO ASIC)
1	R/W	THRSR0_1	Bit 1 of threshold for Region 0 (EISO ASIC)
2	R/W	THRSR0_2	Bit 2 of threshold for Region 0 (EISO ASIC)
3	R/W	THRSR1_0	Bit 0 of threshold for Region 1 (EISO ASIC)
4	R/W	THRSR1_1	Bit 1 of threshold for Region 1 (EISO ASIC)
5	R/W	THRSR1_2	Bit 2 of threshold for Region 1 (EISO ASIC)
6	R/W	TP_LF	EISO ASIC (See description below)
7	R/W	SLOT6	Set to 1 when EISO Card is in Slot 6
8	R	Card ID	Eight bits for Card identification, tied to Vcc,
9	R	Card ID	Can be strapped to ground.
10	R	Card ID	Value is complement of serial number of card – to
11	R	Card ID	Ease visual inspection of straps on SW8
12	R	Card ID	
13	R	Card ID	
14	R	Card ID	
15	R	Card ID	

The threshold bits (0-6) set the three-bit threshold energy for the Neighbor E_T veto of the isolation cuts for the Electron/Photon Algorithm. Please note that the EISO ASIC treats the threshold with a less-than or equal to (≤) and setting these to zero causes all e/γ candidates to be non-isolated.

The TP_LF sets which region will pick up the e/γ candidate if there is a pair of towers with equal energy along a region boundary. **For region zero of an EISO card in slot 3**, the grey towers will be included in the candidate energy inside of the region if the energies are equal. A modification was made to card 6 to make TP_LF compatible with its geometry. It is explained in another document. For Region 0 use TP_LF of address $02 bit 0x40. For Region 1 set TP_LF use address $00 bit 0x40.



TP_LF=0 TP_LF=1

Addressing Memory on the Electron Identification Card

The schematic for block ADRS_DIST of the electron identification card provides the address locations for the EM data streams summarized here. The schematic for block TRNSLTRS shows how the addressing is done to the individual LUT.

Bit	Description
0	Reserved
1	EM E_T Bit 0
2	EM E_T Bit 1
3	EM E_T Bit 2
4	EM E_T Bit 3
5	EM E_T Bit 4
6	EM E_T Bit 5
7	EM E_T Bit 6
8	Not Used
9	Not Used
10	Not Used
11	Not Used

Bit	Description
12	Not Used
13	Not Used
14	Not Used
15	Not Used
16	Not Used
17	Not Used
18	Not Used
19	Not Used
20	Not Used
21	Not Used
22	LUT base address
23	Not used for LUTs - BSCAN

The base address for the memory is $400000. Eight bits can be read/written but only the lower six bits are used for the EM E_T forwarded to the Jet/Summary Card. See schematic LUT for the EIC. Some programs exist to aid the writing the full memory (128 locations). They are described later in the section about booting the crate and initializing the cards since they are system dependent.

Jet Summary Card

Card Control $00

Bit	R/W	Name	Description
0	R/W	Card mode	0=Read memories, 1=Write Memories,
1	R/W	Card mode	84=Run mode, 5=Boundary Scan mode,
2	R/W	Card mode	4=Test mode (Rev. B Cards have 4=Run mode,
3	R/W	Not Used	84=Test mode)
4	R/W	Not Used	
5	R/W	Not Used	
6	R/W	Not Used	
7	R/W	HF test mode	
8	R	/RDY	BSCAN status
9	R	/BSCAN_INT	BSCAN status
10	--	0	Tied to Ground
11	--	0	Tied to Ground
12	--	0	Tied to Ground
13	--	0	Tied to Ground
14	--	0	Tied to Ground
15	--	0	Tied to Ground

HF Errors and Card Identification $02

Bit	R/W	Name	Description
0	R	HF_PHS0	0=No Error on phase of reset and 160 MHz
1	R	HF_PHS1	1, 2, 3=Error on phase of reset and 160 MHz
2	R	HF_BX0	0/1=No Error/Error in BX0
3	R	HF_ERR	0/1=No Error/Error Link or EDC
4	R	HF_PHS_CMP	0/1=No Error/Error in Compare Phase of ASIC
5	--	0	Tied to Ground
6	R/W	RST_BX0	Raise to 1 and lower to 0 to reset BX0 Error
7	R/W	RST_ERR	Raise to 1 and lower to 0 to reset Link/EDC Error
8	R	Card ID	Eight bits for Card identification, tied to Vcc,
9	R	Card ID	Can be strapped to ground.
10	R	Card ID	Value is complement of serial number of card – to
11	R	Card ID	Ease visual inspection of straps on SW4
12	R	Card ID	
13	R	Card ID	
14	R	Card ID	
15	R	Card ID	

Threshold for Region Sums and Phase ASIC Errors and Control $06

Bit	R/W	Name	Description
0	R/W	THRSH_SCK	"Clock" for serial input of threshold data
1	R/W	THRSH_SDX	Threshold serial data in
2	R	THRSH_SDR	Threshold serial data out
3	--	0	Tied to Ground
4	--	0	Tied to Ground
5	--	0	Tied to Ground
6	--	0	Tied to Ground
7	--	0	Tied to Ground
8	--	0	Tied to Ground
9	R/W	ENB_EDC_ERR	0=EDC Errors Off, 1= EDC Errors On
10	R/W	ENB_LNK_ERR	0=Link Errors Off, 1= Link Errors On
11	R/W	ENB_ZERO	0=No zeroing of data, 1=Zero data for any error
12	--	0	Tied to Ground
13	--	0	Tied to Ground
14	--	0	Tied to Ground
15	--	0	Tied to Ground

Setting the quiet thresholds on the JSC

Programming the Quiet Thresholds on the Jet Summary Card

The Jet Summary Card has two nine bit programmable thresholds to set the quiet bit for the barrel (corresponding to Receiver cards 0,1,2,3) and endcap (RC's 4,5,6) region sums. The quiet bit will be high (one) for region $E_T \leq$ threshold. It can be tricky to program since it is done in a serial loop. Eighteen bits of information (bit 1) and "clock" (bit 0) must be entered from the Endcap MSB to the Barrel LSB using the register $06 on the JSC. For data to enter the serial loop correctly, the clock bit must start at zero, and then be set to one (data goes in on the rising edge). The LSB must go in first. For example, in vmedia, to zero the thresholds execute:

```
vmedia> poke 1 19000006 0
vmedia> poke 1 19000006 1
```

18 times. One must be careful with this address, since the bits for the control of the phase ASIC are included in address $06. During normal running, these should probably be:

```
vmedia> poke 1 19000006 e0
vmedia> poke 1 19000006 e1
```

also executed eighteen times. To read back what was in the thresholds, dumping address $06 before the two pokes is necessary. Some example scripts for vmedia are available:

jsc_threshold.txt	Sets both thresholds to hex 0AA and then 155 (for checking all bits by hand).
jsc_threshold_eb.txt	Sets hex 0AA for the endcap and 155 for the barrel calorimeter.
jsc_threshold_zero_and_read.txt	Sets thresholds to zero and reads back.
jsc_threshold_output_test.txt	Sets nine different thresholds necessary to for validation of the JSC, using the ET's from the **output_test** regional sum pattern generation program.

Care must be taken to not disable the phase ASIC settings. When running the values should have a hex e0 added to them.

Addressing Memory on the Jet Summary Card

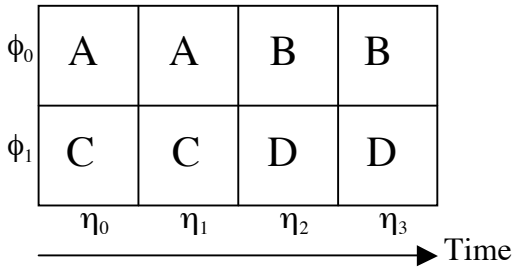
The schematic for block ADRS_DIST of the J/S card provides the address locations for the HF data streams summarized here. The schematic for block TRNSLTRS shows how the addressing is done to the individual LUTs.

Bit	Description
0	Reserved
1	HF E_T ϕ_0 Bit 0
2	HF E_T ϕ_0 Bit 1
3	HF E_T ϕ_0 Bit 2
4	HF E_T ϕ_0 Bit 3
5	HF E_T ϕ_0 Bit 4
6	HF E_T ϕ_0 Bit 5
7	HF E_T ϕ_0 Bit 6
8	HF E_T ϕ_0 Bit 7
9	HF E_T ϕ_1 Bit 0
10	HF E_T ϕ_1 Bit 1
11	HF E_T ϕ_1 Bit 2

Bit	Description
12	HF E_T ϕ_1 Bit 3
13	HF E_T ϕ_1 Bit 4
14	HF E_T ϕ_1 Bit 5
15	HF E_T ϕ_1 Bit 6
16	HF E_T ϕ_1 Bit 7
17	ECL16 Bit
18	Chooses U93 or U225 LUT
19	
20	
21	
22	1 (One) for LUTS
23	Not used for LUTS - BSCAN

The ECL16 bit is used like an additional memory location, so the base addresses for the memories are therefore (LUT 0): η_0: \$400000, η_1: \$420000 and η_2: \$440000, η_3: \$460000. Each memory location is 16 bits wide – therefore when writing to a location for lookup, the lower 8 bits are for the ϕ_0 and the upper 8 bits are for the ϕ_1. Some programs exist to aid the writing the full memory (>32000 locations) for testing. They are described later in the section about booting the crate and initializing the cards since they are system dependent.

It can be tricky to program these LUTs correctly. The test mode accesses memory differently (described later) and sending patterns over the STC can be confusing. The order of the V7216 channels for the HF is as follows:



Below are some tables with examples for the first 25 ns to render the LUT "invisible", i.e. data in from the STC is the same as data out of the JSC (HF):

	STC Channel	**Value ϕ_0**	**STC Channel**	**Value ϕ_1**	**Address**	**Data**
η_0	A	1	C	1	19400202	101
η_1	A	2	C	2	19420404	202
η_2	B	3	D	3	19440606	303
η_3	B	4	D	4	19460808	404
STC vmedia Script		**channelaabbccdd_x0_fg01.txt** QB with 3 & 4 (or **channelaabbccdd_x0_fg10.txt**) QB with 1 & 2				
RCT vmedia Script		**jsc_hf_test_qb.txt**				

	STC Channel	**Value ϕ_0**	**STC Channel**	**Value ϕ_1**	**Address**	**Data**
η_0	A	55	C	0F	19401EAA	0F55
η_1	A	55	C	0F	19421EAA	0F55
η_2	B	AA	D	F0	1945E154	F0AA
η_3	B	AA	D	F0	1947E154	F0AA
STC vmedia Script			**jsc_hf_allbits.txt**			
RCT vmedia Script			**hf_all_bits.txt**			

Running the HF test mode of the JSC

Like the receiver card, the HF can be cycled so that data comes out from the LUTs. In this case the memory is addressed through the Phase ASIC. The Phase ASIC ADDRS0 to ADDRS7 inputs increment from hex 0 to hex FF via a binary counter elsewhere on the board. A reset starts counting at hex 0. This, along with the chip "output enable" and the ECL 16 bit control the output of DATA_ZERO and DATA_ONE. To send a simple increment pattern out via the memory, one must increment the lookup data and the addresses as follows:

JSC Address	**Data**
19440000	000
19460202	101
19400404	202
19420606	303
19440808	404
19460a0a	505
19400c0c	606
19420e0e	707
etc.	etc.

The first cycle actually arrives 12.5 ns earlier than the first cycle for the test mode of

the receiver card. Because of this, the program **output_test_thresholds** used to test the HF circuitry has shifted the order of the data used for the regional sums so that the output for the HF matches that for the regional sums.

Booting the Crate and Initializing the Cards

OS 9 and the MVME167 Crate Controller (Obsolete)

1. First turn on the hard drive and terminal, then power on the crate at the power strip and the 48 V (breaker number 3) at the rack. Turn on the switch for the fans and the VME power supply on the power strip below the crate.
2. Type: `cl fff40084 0.g`
3. Boot from the hard-drive: `hs`
4. Login as super, password poohbah
5. Set the execution directory: `chx /h0/users/lackey/progs/src`
6. Change the directory: `chd /h0/users/lackey/progs/src`
7. Type: `vme_setup&` for any combination of cards. Addresses will be in order by slot. Clock and control card is always 10000000, next slot is the EISO number 0, it will be 11000000, the J/S card will be 19000000, etc. down the line – Receiver Card 3 will be 18000000
8. Change the directory again: `chd ../reva_test`
9. Start the poke and peek program: `vmedia`
10. In vmedia: `read aa3.txt` (do this always when restarting vmedia) to set up a series of commands for the boards and the STC.
11. Now use poke and peek to look at your cards, read the memories, etc.
    a. Scripts Available:
        i. random_long.txt – program random data into the STC memory with correct Hamming Code (EDC)
12. Or exit and use an existing program
    (do `chx /h0/users/lackey/progs/write_mem` before running)
    a. zero_mem – zero a single memory
    b. inc_mem – write an increment pattern to a single memory
    c. dec_mem – write a decrement pattern to a single memory
    d. rand_mem – write a random pattern to a single memory
    e. zero_all – zero all memories in all cards
    f. rand_all – write a random pattern to all memories on a card, one will be prompted for programming each card. This runs very slowly.
    g. All programs have a version for the Electron Identification Card LUT, except for zero_all and rand_all. They are called zero_mem_eiso, inc_mem_eiso, dec_mem_eiso, and rand_mem_eiso.
13. Common Problems:
    a. When programming a new card the vme_setup program crashes. This usually means the vector hasn't been set on SW1 or is reading out incorrectly. Check SW1 and check the orientation of the resistor pack next to it.
    b. After running BSCAN/JTAG the memory doesn't program. Try a: `poke $800014 7001` in VMEDIA for the offending card to reset BSCAN.

Linux and the SBS 620 PCI Crate Controller

1. First turn on the system at the power strip switch. If this doesn't work, check the 5V supply and the breaker panel for the AC 400V to the DC 48V.
2. At the computer that the SBS is connected to try:
   a. `rctBoot` and then type a "b". `rctBoot` may need to be exited and entered again to get it to work.
   b. If it doesn't work, check for the btp module:
   `lsmod`
   c. If it doesn't exist, insert the module (one must have sudo to do this):
   `sudo insmod /cms/SBS/SBSDeviceDriver/1003/v2.0/dd/btp.o`
   you will have to have type your password.
3. Log into the cmslab account.
4. One can now start the peek and poke program `vmedia`. Several scripts are available to help with debugging tasks. One may also start the testing GUI, currently under development.
5. A specialized program, rctCrateTest currently handles memory read/write operations. To run the program and change the test options type
   `rctCrateTest –t XXXX –c YYYY`
   a. `XXXX` is a hexadecimal map, each bit representing a different test:
      i. `XXXX=0:` perform no tests
      ii. `XXXX=1:` Zero Pattern Test only
      iii. `XXXX=2:` Increasing Pattern Test only
      iv. `XXXX=4:` Decreasing Pattern Test only
      v. `XXXX=8:` Random Pattern Test only
      vi. `XXXX=400:` Increasing Hadron Pattern Test
      vii. `XXXX=800:` Decreasing Hadron Pattern Test
   b. To do multiple tests set the values to the appropriate hexadecimal value. For example, `rctCrateTest –t 9` will do the zero and random pattern tests to all cards in the crate. The default value will perform all of the tests described above.
   c. `YYYY` is also a hexadecimal map, each bit representing one of the 15 different card locations:
   $$YYYY=R_6E_6R_5E_5R_4E_4J_0R_3E_3R_2E_2R_1E_1R_0E_0$$
   To access multiple cards set the values to the appropriate hexadecimal value. For example, `rctCrateTest –t 9 –c c0` will do the zero and random pattern tests for RC 3 and EIC 3 only. The default will test all cards in the crate.
6. JTAG for the receiver cards can be run with the program `fault_library`. It will perform all checks for all receiver cards and give a message if there is an error.
7. Output tests can be done with two programs. First zero all the memories using `rctCrateTest –t 1`. Then run output_test to send and predict patterns of sums to the sum cable outputs on the JSC, or `output_electron` to send and `oe_electron` to predict patterns to the JSC electron cable connectors (mezzanine card).

1. This program identifies which slots contain cards using CCC address 1000008 and what kind of card should be inserted in the slot.

2. Zero, incremented, decremented, and random data is written to the card's LUT(s) based on the card type (details for the card types are in the table below) and it is read back:
    a. If the data agrees, nothing happens, and the program moves to the next memory location on the card or the next card in the sequence, finally exiting when all cards are complete. No messages are generated except for identifying which card is being tested at that moment.
    b. If the data returned doesn't equal data programmed – it logs the errors, giving the first 10 cases of data expected, the data returned, and the memory address.

Card	Memory Locations	Address	Word Size	Depth
EIC	1X400000 (X=1,3,5,7,A,C,E)	7 bits	8 bits (6 bits to JSC)	128 locations
RC (Hadron)	1X400000, 1X480000, 1X500000, 1X580000, 1X600000, 1X680000, 1X700000, 1X780000 (X=2,4,6,8,B,D,F)	17 bits	10 bits	131072 locations
RC (e/γ)	1X440000, 1X4c0000, 1X540000, 1X5c0000, 1X640000, 1X6c0000, 1X740000, 1X7c0000 (X=2,4,6,8,B,D,F)	17 bits	8 bits	131072 locations
JSC	19400000, 19440000	17 bits	16 bits	131072 locations

Note: Addresses count by twos – the LA0/LD0 isn't used here. So the first location is 19400000, the second and third location 19400002, 19400004, etc…

Linux and the CAEN PCI Crate Controller

8. First turn on the system at the power strip switch. If this doesn't work, check the 5V supply and the breaker panel for the AC 400V to the DC 48V.
9. Log into the cmslab account at the computer that the CAEN is connected to.
10. In a terminal window do:
    a. `caen-init` to load the driver for the CAEN controller.
    b. `caenRCTBoot` and you should see all the cards that are plugged in by their address.
    c. If it doesn't work, try to reload the module and reboot: `caen-init` and `caenRCTBoot`
    d. If this fails try powering the crate on and off, and repeating it. If this fails,the computer may need to be rebooted.
11. Set the environment variable for the VME controller:
    `export VME_CONTROLLER=CAEN`

12. One can now start the peek and poke program `vmedia`. Several scripts are available to help with debugging tasks. One may also start the testing GUI, currently under development.
13. A specialized program, rctCrateTest currently handles memory read/write operations. To run the program and change the test options type `rctCrateTest –t XXXX –c YYYY`
    a. `XXXX` is a hexadecimal map, each bit representing a different test:
        i. `XXXX=0:` perform no tests
        ii. `XXXX=1:` Zero Pattern Test only
        iii. `XXXX=2:` Increasing Pattern Test only
        iv. `XXXX=4:` Decreasing Pattern Test only
        v. `XXXX=8:` Random Pattern Test only
        vi. `XXXX=400:` Increasing Hadron Pattern Test
        vii. `XXXX=800:` Decreasing Hadron Pattern Test
    b. To do multiple tests set the values to the appropriate hexadecimal value. For example, `rctCrateTest –t 9` will do the zero and random pattern tests to all cards in the crate. The default value will perform all of the tests described above.
    c. `YYYY` is also a hexadecimal map, each bit representing one of the 15 different card locations:
    $$\texttt{YYYY}=R_6E_6R_5\,E_5R_4E_4J_0\,R_3E_3R_2E_2\,R_1E_1R_0E_0$$
    To access multiple cards set the values to the appropriate hexadecimal value. For example, `rctCrateTest –t 9 –c c0` will do the zero and random pattern tests for RC 3 and EIC 3 only. The default will test all cards in the crate.
14. JTAG for the receiver cards can be run with the program `fault_library`. It will perform all checks for all receiver cards and give a message if there is an error.
15. Output tests can be done with two programs. First zero all the memories using `rctCrateTest –t 1`. Then run output_test to send and predict patterns of sums to the sum cable outputs on the JSC, or `output_electron` to send and `oe_electron` to predict patterns to the JSC electron cable connectors (mezzanine card).

3. This program identifies which slots contain cards using CCC address 1000008 and what kind of card should be inserted in the slot.

4. Zero, incremented, decremented, and random data is written to the card's LUT(s) based on the card type (details for the card types are in the table below) and it is read back:
    a. If the data agrees, nothing happens, and the program moves to the next memory location on the card or the next card in the sequence, finally exiting when all cards are complete. No messages are generated except for identifying which card is being tested at that moment.
    b. If the data returned doesn't equal data programmed – it logs the errors, giving the first 10 cases of data expected, the data returned, and the memory address.

Card	**Memory Locations**	**Address**	**Word Size**	**Depth**
EIC	1X400000 (X=1,3,5,7,A,C,E)	7 bits	8 bits (6 bits to JSC)	128 locations
RC (Hadron)	1X400000, 1X480000, 1X500000, 1X580000, 1X600000, 1X680000, 1X700000, 1X780000 (X=2,4,6,8,B,D,F)	17 bits	10 bits	131072 locations
RC (e/γ)	1X440000, 1X4c0000, 1X540000, 1X5c0000, 1X640000, 1X6c0000, 1X740000, 1X7c0000 (X=2,4,6,8,B,D,F)	17 bits	8 bits	131072 locations
JSC	19400000, 19440000	17 bits	16 bits	131072 locations

Note: Addresses count by twos – the LA0/LD0 isn't used here. So the first location is 19400000, the second and third location 19400002, 19400004, etc…

Appendix

The VMEDIA VME Access Program

Used extensively with the OS9 system, VMEDIA has become a powerful command-line debugging tool. Updated for use with Linux and the SBS crate controller, a few more commands have been added to simplify scripting and speed up routine board validation procedures. A summary of the available commands is listed below.

Command	Description
a16	Change to a16 address space
a24	Change to a24 address space
a32	Change to a32 address space
alias *name command*	*name* is now a user defined command
bdump *n address*	Dump *n* bytes starting with *address*
bdump *n address values*	Dump *n* bytes starting with *address* into a symbol *values*
bpeek *n address*	Peek *n* bytes starting with *address*
bpoke *n address values*	Poke *n* bytes starting with *address*
closedumpfile	Close an open dumpfile
dump *n address*	Dump *n* words starting with *address*
dump *n address values*	Dump *n* words starting with *address* into a symbol *values*
exit	Exit program
help !needs to be implemented!	Print help information
ldump *n address*	Dump *n* long words starting with *address*
ldump *n address values*	Dump *n* long words starting with *address* into a symbol *values*
list(*command, command2, …*)	Execute *commands* in the list
loop(*n, command*)	Execute *command n* times, *n*=0 is 1032 times!
lpeek *n address*	Peek *n* long words starting with *address*
lpoke *n address values*	Poke *n* long words starting with *address*
opendumpfile *filename*	Open a file *filenam*e for output of dump command
peek *n address*	Peek *n* words starting with *address*
permit *n address*	Get permission to read/write to *n* bytes starting with *address*
poke *n address values*	Poke *n* words starting with *address*
print	Print a line of text to the screen (useful with text scripts), output will be all lowercase.
read *filename*	Read and execute commands from file *filename*
repeat	
symbol *name count initvalues*	Name can be substituted for numeric values initialized using *initvalues*. *initvalues* can be: array (s1,s2, …), file (filename), series (start, increment)
symbol *name value*	*Name* can be substituted for *value*
wait	Wait for the user to hit return (useful with text scripts)
wdump *n address*	Dump *n* words starting with *address*
wdump *n address values*	Dump *n* words starting with *address* into a symbol *values*
wpeek *n address*	Peek *n* words starting with *address*
wpoke *n address values*	Poke *n* words starting with *address*

Clock and Control Card Switch Settings

Front of CCC

CCC SW	Switch Type	1	2	3	4	5	6	7	8	Total Delay (ps)	Signal	To Board(s)
1	CDELAY	0	0	0	0	0	0	0	0	0	BX0	JSC
2	CDELAY	0	1	0	1	0	1	1	0	1505	ACPT (120)	JSC
3	VME	1	1	1	0	0	0	0	0	n/a	VECTOR	
8	DELAY	0	0	0	0	0	0	0	0	0	160 CLK	RC 6
9	DELAY	0	0	1	0	0	0	0	0	560	160 CLK	RC 4 & 5
10	DELAY	0	1	0	0	0	0	0	0	1120	BX0	RC 2 & 3
11	DELAY	0	1	0	1	1	1	0	0	1610	BX0	RC 0 & 1
12	DELAY	0	0	0	0	1	0	1	0	175	160 CLK	EIC 6
13	DELAY	0	0	0	1	0	1	0	0	350	160 CLK	EIC 4 & 5
14	OSC	0	0	0	0	0	0	0	0	0	OSC DELAY	CCC
15	DELAY	0	0	0	0	0	0	0	0	0	BX0	EIC 2 & 3
16	DELAY	0	0	0	0	0	0	0	0	0	BX0	EIC 0 & 1
17	OFFSET	0	0	0	0	0	0	0	0	0	BX0	EIC 0 to 6
18	DELAY	0	0	0	0	0	0	0	0	0	RESET	RC 6
19	DELAY	0	0	0	1	0	1	0	0	350	RESET	RC 4 & 5
20	DELAY	0	0	1	0	1	1	0	0	770	ACPT (120)	RC 2 & 3
21	DELAY	0	1	0	0	0	0	0	0	1120	ACPT (120)	RC 0 & 1
22	OSC	0	0	0	0	0	0	0	0	n/a	OSC DELAY	CCC
23	OSC	0	0	0	0	0	0	0	0	0	OSC DELAY	CCC
24	OSC	0	0	0	0	1	0	0	0	140	OSC DELAY	CCC
25	DELAY	0	0	0	1	1	0	1	0	455	RESET	EIC 6
26	DELAY	0	0	1	1	0	0	0	0	840	RESET	EIC 4 & 5
27	PHASE	0	1	0	1	0	0	0	0	1400	SET_PHSE	RC & JSC
28	DELAY	0	0	0	0	0	0	0	0	0	ACPT (120)	EIC 2 & 3
29	DELAY	0	0	0	0	0	0	0	0	0	ACPT (120)	EIC 0 & 1
30	OSC	1	1	1	1	0	0	0	0	30	OSC DELAY	CCC
31	OFFSET	0	0	1	1	0	0	0	0	840	160 CLK	EIC 0 to 6

Almost all of these switches control MC10E195's (or SY10E195). The data sheet can be obtained via the web, or from http://www.hep.wisc.edu/~pamc/datasheets. Any switch type labeled "DELAY" is a single MC10E195 with the delay values given in table 1. A switch type labeled "CDELAY" or "OFFSET" is a cascade of two MC10E195's with delay values given in Table 1. If the delay is labeled with "OSC" it is part of the local clock system of the CCC. It should not be changed. Only switches that are hand controlled by dip-switches are shown. Those that must be soldered with jumper wires are not shown.

Switch 3 controls the VME vector, used during card programming, and not used for any delay. It must be set such that it has a hex value E0 for the card to program. This requires that the last 3 switches must be in the off (zero) position.

Switch 14 controls a cascade of 4 MC10E195's, and switch 22 controls the maximum value of 3 of the chips. See the schematic OSC of the CCC, D0708.

Switch 27 controls the phase of the 120 and 160 MHz clocks. This was set up to center the reset in the 160 MHz clock. Please do not change it!

Switch 30 controls a PDU1064H-.5 delay. This delay is in 0.5 ns increments, and only the first 6 switches are used, with switch 6 the LSB, and switch 1 the MSB, i.e. 0.5 ns and 16 ns, respectively. This adjusts the 120 MHz clock with respect to the 160 MHz clock to control the Compare Phase bit for incoming data. For different configurations, cable lengths, etc., this may need to be changed.

Rear of CCC

CCC SW	Switch Type	1	2	3	4	5	6	7	8	Total Delay (ps)	Signal	To Board(s)
33	CDELAY	1	1	1	0	0	1	1	1	4042.5	160 CLK	JSC
34	CDELAY	0	1	0	0	0	0	0	1	1137.5	RESET	JSC
35	DELAY	0	0	1	0	0	1	0	0	630	BX0	RC 6
36	DELAY	0	1	0	0	0	1	1	0	1225	BX0	RC 4 & 5
37	DELAY	0	0	1	0	1	1	1	1	822.5	160 CLK	RC 2 & 3
38	DELAY	0	0	1	1	1	1	1	0	1085	160 CLK	RC 0 & 1
39	DELAY	0	0	0	0	0	0	0	0	0	BX0	EIC 6
40	DELAY	0	0	0	0	0	0	0	0	0	BX0	EIC 4 & 5
41	DELAY	0	0	1	0	1	0	0	0	700	160 CLK	EIC 2 & 3
42	DELAY	0	0	1	1	1	0	1	0	1015	160 CLK	EIC 0 & 1
43	OFFSET	1	1	1	1	1	1	1	1	4462.5	RESET	EIC 0 to 6
44	DELAY	0	1	0	0	0	0	0	0	1120	ACPT (120)	RC 6
45	CDELAY	0	0	0	0	0	0	0	0	0	STAT_CLK	
46	DELAY	0	0	0	1	0	0	0	0	280	ACPT (120)	RC 4 & 5
47	DELAY	0	0	0	1	0	0	0	0	280	RESET	RC 2 & 3
48	DELAY	0	0	0	1	0	1	1	0	385	RESET	RC 0 & 1
49	DELAY	0	0	0	0	1	0	1	1	192.5	ACPT (120)	EIC 6
50	DELAY	0	0	0	0	0	0	0	0	0	ACPT (120)	EIC 4 & 5
51	DELAY	0	0	1	0	1	0	1	0	735	RESET	EIC 2 & 3
52	DELAY	0	0	1	1	1	1	0	0	1050	RESET	EIC 0 & 1
53	OFFSET	0	0	0	0	0	0	0	0	0	ACPT (120)	EIC 0 to 6

All of these switches control MC10E195's (or SY10E195).

Parameter/Pin	D0/27	D1/26	D2/25	D3/24	D4/23	D5/22	D6/21	D7/20
Switch	8	7	6	5	4	3	2	1
Delay Value	17.5	35	70	140	280	560	1120	17.5

Table 1: Delay values for the 10E195