Memory Control

(CRAY T90™ Series)

HTM-059-0

Cray Research Proprietary

Memory Control
-059-

Cray Research, Inc.

Record of Revision

REVISION	DESCRIPTION
	March 1995. Original printing.

Any shipment to a country outside of the United States requires a letter of assurance from Cray Research, Inc.

Requests for copies of Cray Research, Inc. publications should be directed to:

CRAY RESEARCH, INC.
Logistics
6251 South Prairie View Road
Chippewa Falls, WI 54729

Comments about this publication should be directed to:

CRAY RESEARCH, INC.
Hardware Publications and Training
890 Industrial Blvd.
Chippewa Falls, WI 54729

MEMORY CONTROL

Figures

Tables

Tables (continued)

General Information

The memory scheme of the CRAY T90 series system is different than the memory scheme of past systems. Previously, Cray Research used a memory system referred to as a *parallel access system*. In this type of memory, all CPUs are involved in every memory reference.

The CRAY T90 series system uses a *network access system*. In this system, the requesting CPU has to arbitrate the access only in its own ports. Once that decision has been made, the reference leaves the CPU, and no other CPU ever gets involved in the process. It is the job of the network module and the memory module to resolve access conflicts to the memory bank.

The memory and network modules work in a *handshaking* fashion. There are first-in-first-out (FIFO) buffers and arbitrators on these modules. If a FIFO buffer for an arbitrator is full because of an unresolved conflict, that buffer will not send a Resume signal to its supplier. The supplier then starts filling up its FIFOs. This process continues all the way back to the CPU. When the CPU FIFOs are full, the system stops issue. The buffer sizes have been designed to minimize the number of hold issues caused by memory backup.

Options

There are ten option types used in the CRAY T90 series systems for memory control. Of these ten types, there are 51 total options per CPU that deal with memory control. Table 1 lists the option types, the number of options per CPU, and their main functions.

Table 1. Memory Control Options

Option Type	Number	Function
CA	2	Calculates address, checks LAT, and generates destination codes for ports A, A', B, and B'
CB	2	Calculates address, checks LAT, and generates destination codes for ports C and C'
CC	1	Calculates address, checks LAT, and generates destination codes for port D
CD	2	Calculates address, checks LAT, and generates destination codes for port E, cache hit or cache miss determination
CF	6	Checks write data conflicts and port-to-section priority and arbitration
CG	2	Performs the check-bit generation write data to memory (1 CG per pipe)
CH	16	Decodes destination code, data steering, and cache memory
CI	8	Section driver that transmits control and data to network or memory module; one CI for each memory section
CJ	8	Performs the SBCDBD (single-byte correction/double-byte detection) for read data path from memory; one CJ for each memory section
CK	4	Provides overall arbitration and steering of data through destination codes and cache control

Memory Ports

Each CPU has eight memory ports for access to memory. Refer to NO TAG for a list of the port functions.

Table 2. Port Functions

Port	Function
A	Block reads, vector even elements and B transfers
A'	Block reads, vector odd elements
B	Block reads, vector even elements and T transfers
B'	Block reads, vector odd elements

Table 2. Port Functions (continued)

Port	Function
C	Block writes, vector even elements and B/T transfers
C’	Block writes, vector odd elements
D	Exchange, fetch and I/O
E	Address and scalar data, cache

Depending on which port is available, vector load operations use either port A/A’ or B/B’. If both sets of ports are available, A/A’ is used first. The double-gather operation uses both sets of ports, A/A’ and B/B’.

Logical Address Translation (LAT) Table

The CRAY T90 series computer system uses a system of memory management called logical address translation (LAT) tables. The LAT tables replace the traditional base address registers and the limit address registers, which were too restrictive. LAT tables allow for multiple address spaces.

Each LAT has a 16 K-block size and is based on a 40-bit logical address space and a 38-bit physical address space. This means that the maximum logical address is 1 teraword, and that space is allocated and checked in 16K-blocks. Because of the 38-bit physical address space, the LAT implementation has a 256-Gword limit. However, the CRAY T90 series system supports 34 bits of address for a total of 16 Gwords of memory.

The LAT table can contain up to eight entries; there is space in the exchange package for eight LAT tables. There are hardware limitations in the memory control logic that allow for a maximum of three LAT entries. There can be three entries with read and/or write permission and three entries with execute permission. If the limits are violated, the LAT information could be lost or ignored. If a LAT table has no mode bits set, the rest of the entry is ignored.

The CRAY T90 series system does not return memory requests in a predetermined order. This means that the order of the LAT entries into the hardware cannot be determined from their order in the exchange package. Because of this, there is no way to determine which LAT table was lost if mode limits are exceeded.

The options that handle the port address calculations (CA, CB, CC, and CD) are responsible for determining a LAT compare (hit) or no compare (miss). The LAT tables with the execute (X) bit set are sent to the CC options for use in fetch, exchange, and I/O operations; in addition, the LAT tables with the execute (X) bit set use absolute addresses. The LAT tables with the read mode bit (R) set go to the CA and CD options for use by ports A, B, and E. The LAT tables with write (W) bit set go to the CB and CD options for use by ports C and E.

Each of these options has storage space for three LAT tables. In order for a LAT hit to occur, the generated request address must be greater than or equal to the LAT base field and less than the LAT limit field. If a given address fails to compare (misses) with any of the three stored LAT tables, a *no match fault* is generated. There are actually two types of no-match faults that can occur: address range faults and access violations. An address range fault occurs when a request is made outside any LAT range. An access violation occurs when a request matches a LAT without the proper modes set. In the CRAY T90 series system, there is no distinction between the two types; they are both reported as no-match faults.

If an address matches (hits) more than one of the stored LAT tables, a *multiple LAT hit fault* occurs. The hardware handles *no match faults* and *multiple LAT hit fault* errors the same way. On a read request, the memory reference is aborted; on a write request, the request is treated as a NO-OP. Depending on the conditions specified by the exchange parameters, an interrupt may or may not occur.

If a LAT address range fault occurs and the Interrupt on Operand Range Error (IOR) bit is set in the exchange package, the reference is aborted. The Operand Range Error (ORE) flag then sets in the exchange package and an exchange sequence occurs. If the IOR bit is clear, the reference aborts and program execution continues.

If a LAT fault occurs on a fetch request, the state of the Interrupt on Program Range Error (IPR) bit determines the next action that occurs. If this bit is set in the exchange package, the request is aborted. The Program Range Error (PRE) flag then sets in the exchange package and an exchange sequence occurs. If the IPR bit is clear, the request is aborted; this causes words of zeroes to be written to the instruction buffers. When these parcels execute, a zero instruction (000000) is decoded as an error exit.

If a single LAT hit occurs, the appropriate physical bias is multiplexed (MUXed) from the LAT entry and added to the generated request address to form a physical address that is sent to the CF option. Table 3 defines the LAT fields.

Table 3. LAT Fields

Field	Description
Logical Base	The first logical address of this LAT
Logical Limit	The last logical address +1 of this LAT
Physical Bias	A logical-to-physical mapping constant (Physical_Base_Address = Physical_Bias + Logical_Base Address)
Modes	Bits that define the use of this LAT (R)ead (W)rite (X)execute (C)achable

LAT Calculations

LAT calculations are based on three fields: Logical Base Address, Logical Limit Address, and Physical Bias. Unlike previous systems, there is no Physical Base Address.

Physical Bias maps logical addresses to a physical space in memory. The Physical Bias is calculated by the operating system and the user has no control over it. Physical Bias is calculated using the following equation:

Physical Bias = Physical Base Address – Logical Base Address

NOTE: Because there is no relation between physical space in memory and logical space, the Physical Bias could be a negative number.

The programmer sets the Logical Base Address and Logical Limit Address in the exchange package. Refer to Figure 1 for an example of the CRAY T90 series memory mapping. In this example, three *pages* or areas of memory are mapped.

Page 1 has a physical base address of 0 and a logical base of 30K. Using the formula to calculate physical bias yields a physical bias of –30K. If a requested address was 35K and you wanted to map that to a physical space in memory, you would add that address to the physical bias, and the results would be at address 5K in memory. This example shows that the physical bias can be a negative number.

The other two pages can be calculated the same way as the first page. Note that when the physical bias is 0, the request address always equals the physical address.

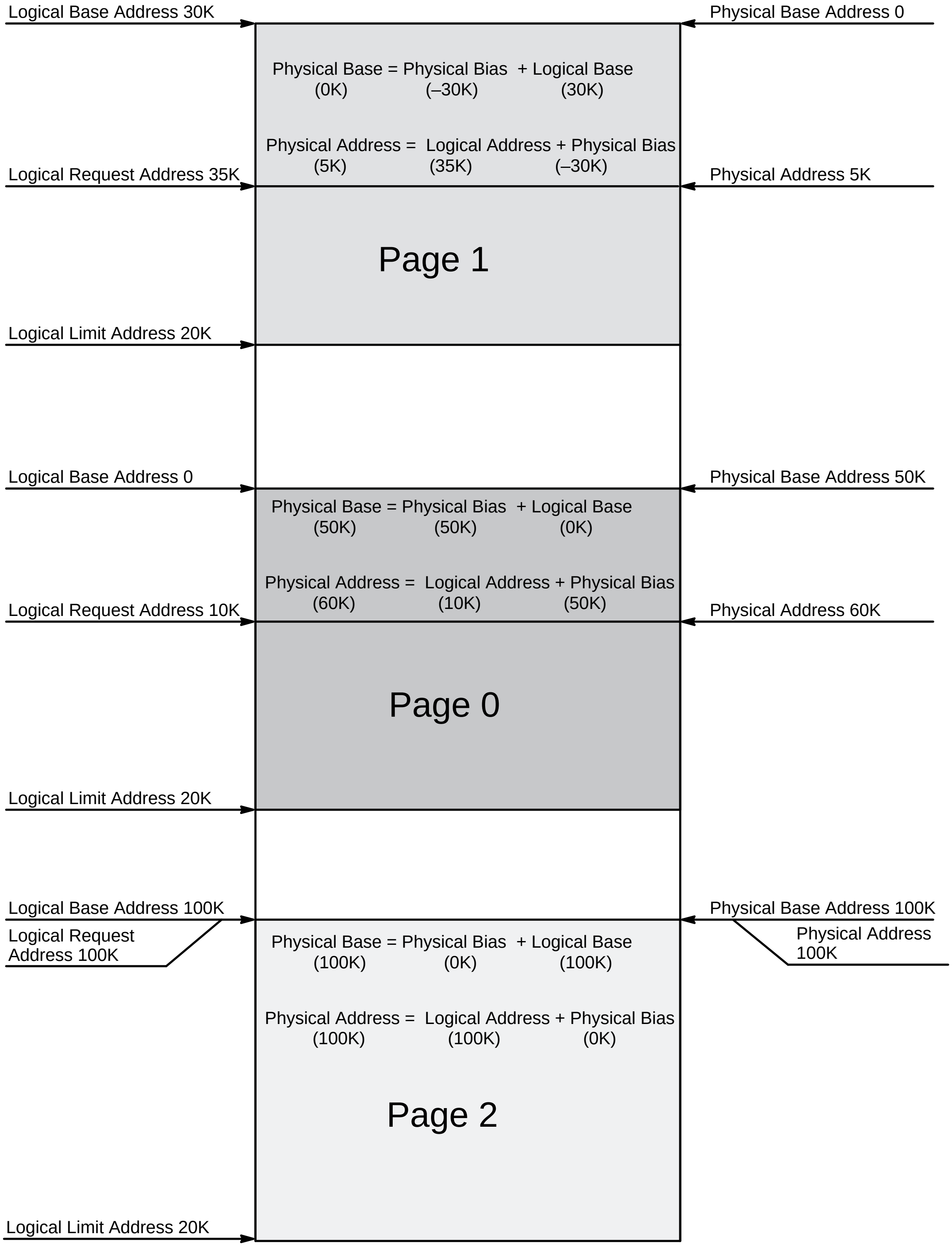

Logical Base Address 30K
Physical Base Address 0
Physical Base = Physical Bias + Logical Base
(0K) (–30K) (30K)
Physical Address = Logical Address + Physical Bias
(5K) (35K) (–30K)
Logical Request Address 35K
Physical Address 5K
Page 1
Logical Limit Address 20K
Logical Base Address 0
Physical Base Address 50K
Physical Base = Physical Bias + Logical Base
(50K) (50K) (0K)
Physical Address = Logical Address + Physical Bias
(60K) (10K) (50K)
Logical Request Address 10K
Physical Address 60K
Page 0
Logical Limit Address 20K
Logical Base Address 100K
Physical Base Address 100K
Logical Request Address 100K
Physical Address 100K
Physical Base = Physical Bias + Logical Base
(100K) (0K) (100K)
Physical Address = Logical Address + Physical Bias
(100K) (100K) (0K)
Page 2
Logical Limit Address 20K

Figure 1. CRAY T90 Series Memory Mapping

Modes

The mode bits (refer to the following list) are the controlling field within each LAT table; the mode fields are duplicated in the exchange package. The mode bits of each of these entries must be identical. For example, word 0 and word 10 are the exchange package entries for LAT 0. The mode bits for these exchange words must be the same; if the read bit is set in word 0, it must also be set in word 10.

- The Read bit indicates that the LAT can be read from.
- The Write bit indicates that the LAT area can be written to.
- The Execute bit indicates that the LAT area contains executable code. One LAT must contain an executable mode bit.
- The Cachable bit indicates that read data can be cached. It is the programmer’s/compiler’s responsibility to set or clear the cache bit, depending on whether cache is valid.
- The Dirty status bit is set by the hardware. It is set anytime a LAT space is written, indicating that its associated physical memory space has also been written to. This bit is not cleared by the hardware. It is intended for use by the operating system.

CA Option

The CA option is responsible for all address calculations and LAT comparisons for the block read port to memory. There are two CA options per CRAY T90 series system CPU: CA000 and CA001. CA000 handles all references for port A and A’ (B and V reads), both even and odd elements. CA001 handles all references for port B and B’ (T and V reads), both even and odd elements.

The CA option also generates the destination code that is sent with the memory reference and is carried through memory. It is then returned to the CPU on a read so that the CPU’s in-bound memory port can determine the destination of the data.

A resume counter on the CA option counts the number of valid signals sent to the CF option versus how many resume signals are sent from the CF option. If the number of valid signals gets too high, a hold signal is generated, which stacks the references in three buffer ranks. This hold signal also stacks V*k* data (both even and odd elements) on a gather operation. Only V*k* rank 0 is on the CA; the other ranks, 1 through 6, are on the VF option.

A copy of the vector length register is sent to the CA options from the AN option for use during vector read operations. The CA options also capture all read LATs during an exchange and hold them until the next exchange.

The CA options also have a block reference counter, which ensures that all the references for a given instruction have left the CA before a port release signal is sent back to the JA option. For B and T transfers, the block counter is initialized from the A*k* path. This is the same path that V*k* data comes from: the VF002 and VF003 options. On a vector operation, the counter is initialized with a value from the VL register. The incoming vector length is divided by two and then rounded by the remainder (the incoming vector length is shifted down by 1 bit and then rounded up by the remainder). This is done because the port sends out two references per clock period, and the vector length needs to be rounded to handle vector lengths that are odd. This allows the block decrement counter to always decrement by 1 no matter which instruction is in progress.

If the CA option can ensure that all references left in the block counter will be gone, either to the CF ranks or in the CA itself, by the time the next instruction is ready to start, it will then send the Release Port A/B signal back to the JA options. This is done by checking the block counter plus the resume counter to ensure the total is less than four. (The block counter tells how many references are left, and the resume counter tells how many references the CF can take.) On a gather instruction, there are two more conditions that must be checked before a release can be generated. First, the CA option verifies that all V*k* elements have been sent from the registers. Second, the CA option verifies that the Gather Done signal is ready to send to the JA option.

Table 4 shows how request addresses are calculated for the different types of read transfers.

Table 4. Request Address Calculations

Read Type	First Address	Increment	Port Used
B	A0	1, accum	A
T	A0	1, accum	B
V normal V normal	A0 A0 + A*k*	2A*k*, accum 2A*k*,accum	A, B if A busy A’, B’ if A’ busy
V gather V gather	A0 + V*k*0 A0 + V*k*1	V*k* V*k*	A, B if A busy A’, B’ if A’ busy
Double gather	A0 + V*k*0	V*k*	A
Double gather	A0 + V*k*1	V*k*	A’
Double gather	A*k* + V*k*0	V*k*	B
Double gather	A*k* + V*k*1	V*k*	B’

CB Option

The CB option is responsible for all address calculations and LAT comparisons for the block write port to memory. There are two CB options per CRAY T90 series system CPU: CB000 and CB001.

CB000 handles all references for port C (B and T writes and V even element writes), and CB001 handles all references for port C’ (V odd element writes). The CB also generates the destination code for the CPU write reference. Because this is the only CPU write port to memory, there is only one possible code that can be generated. The code is actually generated from the master clear signal on the module, forcing all bits (except bit 8) to 0’s. Refer to NO TAG for descriptions of the destination codes.

Table 5. Destination Codes

Destination Code Bits															
13	12	11	10	9	8	7	6	5	4	3	2	1	0	Description	
1	1	1	–	w	w	w	w	w	w	w	w	w	w	Cache	Word
1	1	0	r	r	r	–	e	e	e	e	e	e	e	Vector	Number, Word
1	0	1	r	r	r	0	–	–	–	–	–	–	–	S Register	Number
1	0	1	r	r	r	1	–	–	–	–	–	–	–	A Register	Number
1	0	0	–	–	–	0	–	r	r	r	r	r	r	T Register	Number
1	0	0	–	–	–	1	–	r	r	r	r	r	r	B Register	Number
0	1	1	g	g	g	–	–	–	w	w	w	w	w	Fetch Read	Group, Word
0	1	0	t	t	t	–	–	w	w	w	w	w	w	I/O Read	Type, Word
0	0	1	–	–	–	–	–	–	w	w	w	w	w	Exchange Read	Word
0	0	0	t	t	t	1	–	–	–	–	–	–	–	I/O Write	Type
0	0	1	–	0	1	0	a/s	–	–	–	–	–	–	Processor Write	
0	0	1	1	1	–	–	–	–	e	e	e	e	e	Reconfigure	Word
0	0	0	0	0	0	–	–	–	e	e	e	e	e	Memory Error	

e = element, r = register number, w = word, t = type, and – equals an unused bit position returned as 0.

Each CB option also receives a copy of the A*h* field from the CD option. This copy of the A*h* field forms the vector length register on the CB option for use with vector operations.

CC Option

The CC option provides all the memory addresses for port D (instruction fetch, exchange and I/O.) The CC option also generates the destination code that is sent with the memory reference. It is then carried through memory and returned to the CPU on a read so that the CPU's in-bound memory port can determine the destination of the data.

A resume counter on the CC option counts the number of valid signals sent to the CF option and how many resume signals are sent from the CF option. If the number of valid signals gets too high, a hold signal is generated that stacks the references in buffer ranks until the CF starts sending more resume signals.

The CC option performs some arbitration when more than one type of reference tries to execute at the same time. The IC and JA options ensure that the Go Fetch and Go Exchange signals do not arrive at the CC option simultaneously. The CC option, however, must delay an exchange if a fetch is in progress, or delay the fetch until the exchange completes. Because I/O write data is merged with CPU write data on the CH option, the JA option waits until all memory activity is quiet before sending the Go I/O signal to the CC option. The JA option holds issue until the I/O Request signal is dropped and the I/O operation has completed on the CC option.

The CC option receives eight CPU Write Complete signals (one from each section), which notifies the CC option that an exchange has completed. If the exchange is still active, the CPU write completes are part of this process. The CC counts these signals until the count reaches 32_{10} (remember, the exchange package is 32 words long).

When the count reaches 32, the Exchange Write Complete signal sets, which causes the Exchange Active signal to drop. If the exchange is not active, these signals are added to the CPU Read Complete signals from the CK options and to the CPU Write Aborted signal from the CI option. The addition of these signals creates a 6-bit count of the number of CPU references, which is sent to the performance monitor.

CD Option

There are two CD options per CRAY T90 series system CP module. The CD option is responsible for the address and LAT calculations and generates destination codes for port E, address, and scalar and cache data. The CD options also determine cache hits or cache misses. Once the request address is calculated, a cache check is performed to determine one of three things: the CD checks for a cache hit, a cache miss, or no cache. The cache operates in *write through mode* so that memory is updated as cache is updated.

For a cache hit to occur, there must be a page compare, there must be a line valid, and the LAT must be set for cache. If there is a compare, the CD options form a cache address and send it to the CH options. CD000 controls the even CH options, CD001 controls the odd CH options, and both CDs monitor all cache operations.

The CD option also sends a Cache Register/Page Request signal to the CK options. This signal notifies the cache that a new line is due to arrive and that the cache should begin counting the references. The CK option then returns the Cache Line Complete signal immediately after 16 words (one cache line) have been transmitted from memory to this cache line. Refer to Figure 2 for an illustration of the cache control.

Port E is also used as a write port to memory for A and S registers. However, the CF option, which arbitrates the writes to memory, MUXes the inputs from ports C and E so that only one write port can be active at a time.

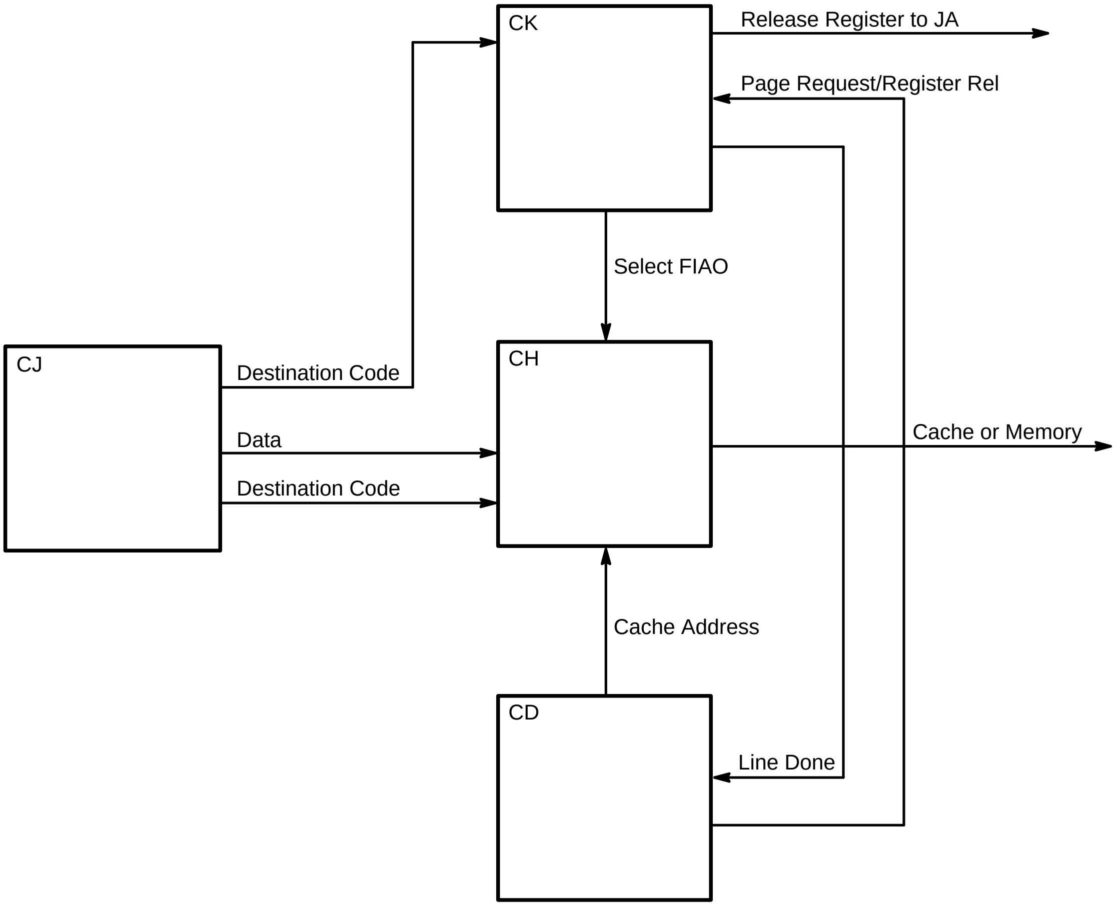


Figure 2. Cache Control

Refer to Table 6 and Table 7. These tables show the distribution of reference code and address bits from the CD options to the CF options. The request code bits are the destination code formed by the address bits and control bits. They specify the page and line during cache references, A and S register data transfers, and CPU writes.

NOTE: The destination code bits 0 and 1 are not sent by the CD options because the CD requests 4 or 8 words per clock period.

Table 6. Request Codes and Address Bits from CD000 to CF Options

CF Option	I Terms	Request Code Bits	I Term	Address Bits
CF000	IDN, IDO	6 12	IDP, IDT, IDV	6 12 14
CF001			IDT, IDV	16 18
CF002	IDM, IDN	2 8	IDO, IDP, IDT, IDV	2 8 20 22
CF003			IDT, IDV	24 26
CF004	IDM, IDN	4 10	IDO, IDP, IDT, IDV	4 10 28 30
CF005	IDP	Abort	IDT, IDV	32 34

Table 7. Request Codes and Address Bits from CD001 to CF Options

CF Option	I Terms	Request Code Bits	I Term	Address Bits
CF000			IDS, IDU	11 13
CF001	IDN IDO	7 13	IDP, IDS, IDU	7 15 17
CF002			IDS, IDU	19 21
CF003	IDM, IDN	3 9	IDO, IDP, IDS, IDU	3 9 23 25
CF004			IDS, IDU	27 29
CF005	IDM, IDN	5 11	IDO, IDS, IDU	5 31 33

CF Option

The CF option performs the write data conflict checking and port-to-section arbitration for all memory requests. The CF options also steer the destination code, bank, and address to the selected section.

There are six CF options per CPU. All the code bits are bit-sliced across the six options, and all the options perform the arbitration in parallel.

The CF options receive port requests from A, A’, B, and B’, and arbitration is performed. If a hold condition occurs due to a No Section Resume signal coming in from memory or due to a port conflict that was generated by the CF option, the CF option stores the reference in a four-deep stack until the hold condition ends.

The same situation is true for requests coming from ports C, C’, D, and E, with the exception that the hold stack is five deep. The CF option also prevents references from occurring on both ports C and E at the same time. To prevent this, the two ports are MUXed at the input to the CF options and share the same priority position.

The CF000 through CF003 options also send a CPU Memory Request (OIQ) signal to the performance monitor to count the number of requests.

CF001 sends out the following status control signals to the JA options for issue control:

- Read Ports quiet (OIM)
- Write Ports quiet (OIN)
- CM All quiet (OIO)
- Port A/B quiet (OIP)

Hold Conditions

For a hold condition to exist, several events must occur. First, a conflict, which was caused of one of the following events, must exist:

- No section Resume signal was received from memory

- A write data conflict occurred between two ports

- Two ports were simultaneously requesting the same section in memory

Second, a second reference from the same port to the same section must arrive before the previously mentioned conflict is resolved. If this happens, the request would be held in rank 0 of the stack for that port, and the port Resume signal (OJA through OJE) is dropped. This instructs the port to stop sending requests. If the conflict is still not resolved, the requests from that port are held in the subsequent port rank, regardless of which section they are requesting. The Resume signal is blocked until the conflict is resolved. When this occurs, the requests begin unstacking.

Each section has a separate request register for each port. This means that up to 56 separate references (8 sections × 7 memory ports = 56 request registers) can be held before it becomes necessary to stack any references.

Priority and Arbitration

The CF option receives requests from eight ports and must prioritize arbitration of these requests and conflicts.

The CF option can receive up to seven requests at the same time. All port requests are sent to the section request register of the section that was requested. When two or more ports on the same CPU request the same section, the request is released or held according to a priority scheme. Each section can handle one release per clock period.

Each section request register on the CF has a 4-bit counter; the counter counts from 0 to 16 and back to 0. Each time a request is released, the counter increments by 1. At each counter value, the priority shifts. Refer to Table 8 for the port priority.

Table 8. Port Priority

Priority							
Count	Highest						Lowest
0	A	A’	B	B’	CE	C’	D
1	A’	A	B’	B	CE	C’	D
2	A	A’	B	B’	CE	C’	D
3	A’	A	B’	B	CE	C’	D
4	A	A’	B	B’	CE	C’	D
5	A’	A	B’	B	CE	C’	D
6	A	A’	B	B’	CE	C’	D
7	A’	A	B’	B	CE	C’	D
10	A	A’	B	B’	CE	C’	D
11	A’	A	B’	B	CE	C’	D
12	A	A’	B	B’	CE	C’	D
13	A’	A	B’	B	CE	C’	D
14	A	A’	B	B’	CE	C’	D
15	A’	A	B’	B	CE	C’	D
16	D	A	A’	B	B’	CE	C’
17	D	A’	A	B’	B	CE	C’

Ports C and C’ never swap priority because these ports are used for vector writes. Port C receives the even elements and port C’ receives the odd elements. When both of these ports are active, they operate in a *lock-step mode*. This mode ensures that they always enter the same ranks and shift through and hold at the same time. This design prevents problems on vector scatter operations.

During a vector scatter operation, a possibility exists that both ports could be sending the data to the same address in the section. The only way to ensure data integrity is to force ports C and C’ (that go to the same section) to release in element order. This is done by forcing port C over C’ and then blocking any subsequent port C requests from entering the request register until no C’ conflicts exist. The same is true for port C’; thus they are forced into *lockstep*.

Bit Slice

Refer to NO TAG to see how the reference code bits and address bits are sliced (divided) across the six CF options. The port control options (CA, CB, CC, CD) send these bits to the CF options. The CF options then send this control to the eight CI options for a memory reference.

Table 9. CF Option Bit Slicing to CI Options 0 through 7

CF Option	Reference Code Bits	Bank Bits	Address Bits
CF000	0 6 12	6, Bad Bit	11 12 13 14
CF001	1 7 13	7, Partition Bit	15 16 17 18
CF002	2 8	2 8, Partition Bit	19 20 21 22
CF003	3 9	3 9	23 24 25 26
CF004	4 10	4 10	27 28 29 30
CF005	5 11	5 Abort	31 32 33 34

Control Bit Slice

Control is also sliced across four of the CF options. The control terms are distributed as shown in Table 10.

Table 10. Control Bit Slicing

Boolean Term	CF000	CF001	CF002	CF003
IGJ		Port C Vector Request		
IGU	Port A Gather	Port A Gather		
IGV	Port B Gather	Port B Gather		
IGW	Port C Scatter	Port C Scatter	Port C Scatter	Port C Scatter
IGX	Double Gather	Double Gather		
OIA	Sect 0 Valid	Sect 1 Valid	Sect 2 Valid	Sect 3 Valid
OIB	Sect 0 Valid	Sect 1 Valid	Sect 2 Valid	Sect 3 Valid
OIC	Sect 4 Valid	Sect 5 Valid	Sect 6 Valid	Sect 7 Valid
OID	Sect 4 Valid	Sect 5 Valid	Sect 6 Valid	Sect 7 Valid
OIG	Even Route 0			
OIH	Even Route 1			
OII	Even Route 2			
OIJ	Even Route Valid	Even Route Valid	Even Route Valid	
OIM		Rd Ports Quiet		
OIN		Wr Ports Quiet		
OIO		CM Quiet		
OIQ	CPU Req Bit 0	CPU Req Bit 1	CPU Req Bit 2	CPU Req Bit 3
OIS		Odd Route 0		
OIT		Odd Route 1		
OIU		Odd Route 2		
OIV		Odd Route Valid		
OIW		Odd Route Valid		
OIX		Odd Route Vector		
OJK	Port C Hold VA	Port C Hold BT		
OJL	Port A Hold VF	Port A Hold VF		
OJM	Port B Hold VF	Port B Hold VF		
OJN	Port C Hold VF	Port C Hold VF	Port C Hold CB	
OJQ			I/O Route 1	I/O Route 1
OJR			I/O Route 2	I/O Route 2
OJS			I/O Route Go Even	I/O Route Go Even
OJT			I/O Route Go Odd	I/O Route Go Odd

Multiple Requests

Several signals cause the CF option to send multiple requests from only one input request. These signals control exchange, I/O, and scalar cache:

- Exchange Active (IHJ)
- Exchange Read (IHK)
- I/O Request (IHL)
- I/O, Fetch Single (IHM)
- Cache Miss (IHP)

The number of output requests varies, depending on system configuration (full-memory vs. half-memory configuration). Refer to Table 11 for a list of the multiple requests.

Table 11. Multiple Requests

		Number of References	
Reference Type	Control Signal	Full Memory	Half Memory
Exchange Write	IHJ ihj	1	1
Exchange Read	IHJ IHK	8	4
Fetch	ihj ihl ihm	8	4
I/O Single	IHL IHM	1	1
I/O Double	IHL ihm	2	2
Cache Miss	IHP	8	4

Destination Codes

Refer to Table 12 for descriptions of the destination codes received by the CF options. These codes, sometimes referred to as reference codes, are generated by the port address options and sent out with the address. This code is carried (along with the reference) through memory. The destination code comes back to the CPU with the read data so the CPU can determine to which register the data is to be sent.

Table 12. Destination Codes for CF Options

		Destination Code Bits													
13	12	11	10	9	8	7	6	5	4	3	2	1	0	Description	
1	1	1	–	w	w	w	w	w	w	w	w	w	w	Cache	Word
1	1	0	r	r	r	–	e	e	e	e	e	e	e	Vector	Number, Word
1	0	1	r	r	r	0	–	–	–	–	–	–	–	S Register	Number
1	0	1	r	r	r	1	–	–	–	–	–	–	–	A Register	Number
1	0	0	–	–	–	0	–	r	r	r	r	r	r	T Register	Number
1	0	0	–	–	–	1	–	r	r	r	r	r	r	B Register	Number
0	1	1	g	g	g	–	–	–	w	w	w	w	w	Fetch Read	Group, Word
0	1	0	t	t	t	–	–	w	w	w	w	w	w	I/O Read	Type, Word
0	0	1	–	–	–	–	–	–	w	w	w	w	w	Exchange Read	Word
0	0	0	t	t	t	1	–	–	–	–	–	–	–	I/O Write	Type
0	0	1	–	0	1	0	a/s	–	–	–	–	–	–	Processor Write	
0	0	1	1	1	–	–	–	–	e	e	e	e	e	Reconfigure	Word
0	0	0	0	0	0	–	–	–	e	e	e	e	e	Memory Error	

e = element, r = register number, w = word, t = type, and – equals an unused bit position returned as 0.

CG Option

Because there are two vector pipes, there are two CG options: CG000 and CG001. CG000 handles the even pipe; CG001 handles the odd pipe.

Refer to Figure 3 for an illustration of the CG option. There are two 64-bit data paths into each CG option. One path is for vector data and the other path is for A, S, B, and T register data and exchange data.

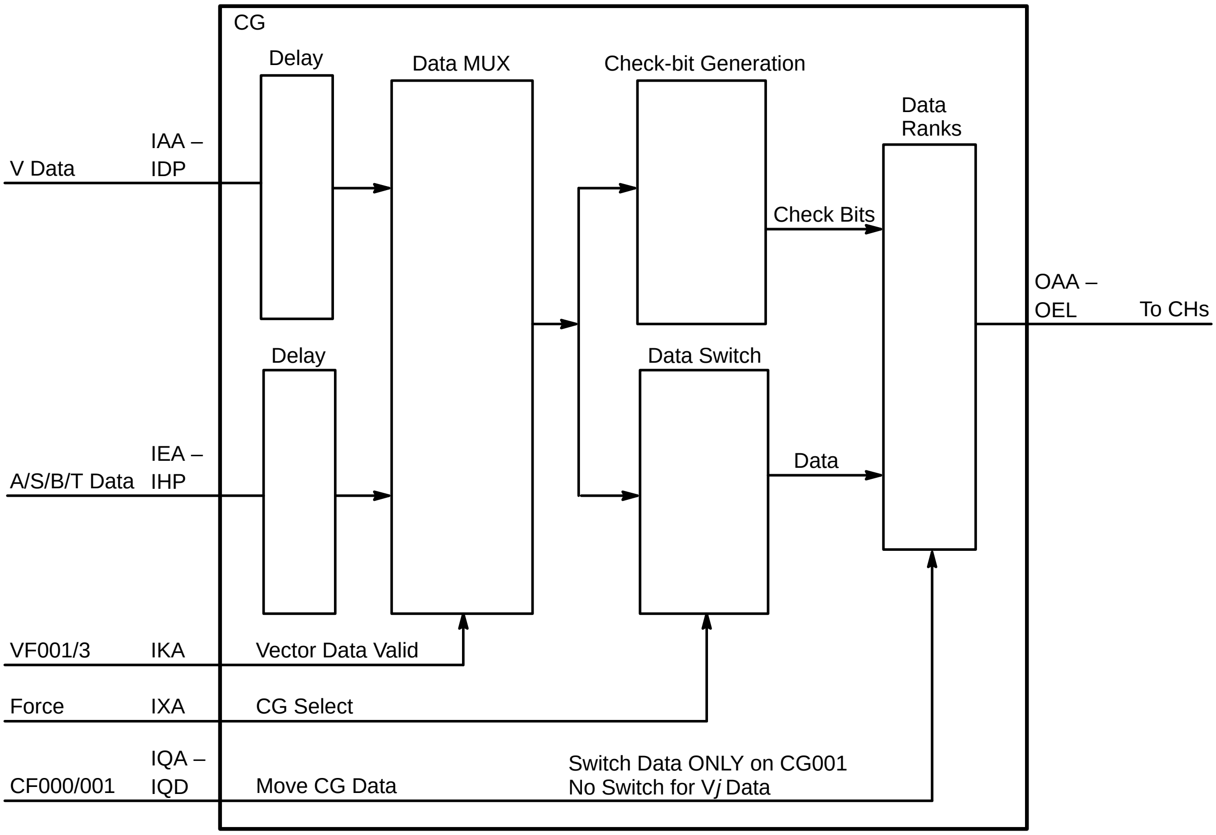


Figure 3. CG Option

The Vector Data Valid signal (from the VF option) selects the input path. The CG Select signal identifies the two CG options; these are forced terms. CG000 sends out A, S, B, and T register data and exchange data bits 0 through 63. CG001 sends the data out on bits 32 through 63, then on bits 0 through 31. The bit swap provides a means of maintaining speed in the cache even when the system is running in degraded mode or with a stride value of four.

Refer to Table 13 for the CG option data fanouts to the CH options.

Table 13. CG Option Data Fanouts

CG000 / CG001							
O terms from CG	Bits	Receiving Option	I Terms	O term from CG	Bits	Receiving Option	I Terms
OAA	0	CH000	ICA	OCA	32	CH001	ICA
OAC	1		ICB	OCC	33		ICB
OAE	2		ICC	OCE	34		ICC
OAG	3		ICD	OCG	35		ICD
OEA	CB 0		ICE	OEK	CB 8		ICE
OAI	4	CH002	ICA	OCI	36	CH003	ICA
OAK	5		ICB	OCK	37		ICB
OAM	6		ICC	OCM	38		ICC
OAO	7		ICD	OCO	39		ICD
OEC	CB 1		ICE	OEH	CB 9		ICE
OBA	8	CH004	ICA	ODA	40	CH005	ICA
OBC	9		ICB	ODC	41		ICB
OBE	10		ICC	ODE	42		ICC
OBG	11		ICD	ODG	43		ICD
OEE	CB 2		ICE	OEJ	CB 10		ICE
OBI	12	CH006	ICA	ODI	44	CH007	ICA
OBK	13		ICB	ODK	45		ICB
OBM	14		ICC	ODM	46		ICC
OBO	15		ICD	ODO	47		ICD
OEB	CB 3		ICE	OEL	CB 11		ICE
OAB	16	CH008	ICA	OCB	48	CH009	ICA
OAD	17		ICB	OCD	49		ICB
OAF	18		ICC	OCF	50		ICC
OAG	19		ICD	OCG	51		ICD
OED	CB 4		ICE				
OAJ	20	CH010	ICA	OCJ	52	CH011	ICA
OAL	21		ICB	OCL	53		ICB
OAN	22		ICC	OCN	54		ICC
OAP	23		ICD	OCP	55		ICD
OEF	CB 5		ICE				
OBB	24	CH012	ICA	ODB	56	CH013	ICA
OBD	25		ICB	ODD	57		ICB
OBF	26		ICC	ODF	58		ICC
OBH	27		ICD	ODH	59		ICD
OEG	CB 6		ICE				
OBJ	28	CH014	ICA	ODJ	60	CH015	ICA
OBL	29		ICB	ODL	61		ICB
OBN	30		ICC	ODN	62		ICC
OBP	31		ICD	ODP	63		ICD
OEI	CB 7						

CG000 provides the IC terms to the CH options for pipe 0.
CG001 provides the ID terms to the CH options for pipe 1.

The CG option performs check-bit generation for the write data path to memory. The check-bit algorithm provides 12 check-bits for the detection and correction of adjacent errors on 2-bit nibbles and double-byte detection. For more information on the error-correction scheme, refer to the *CM02 Memory Module* document, publication number HTM-004-0.

CH Option

There are 16 CH options per CP module. The CH options receive data from several sources. The CH options receive corrected data from memory through the CJ options and direct the data to a first-in-any-out (FIAO) buffer specified by the accompanying destination code. The CH options also receive CPU write data from the CG options and I/O data from the HA options.

Write Data

The CH options receive 4 bits of memory write data from each of the two CG options. CG000 sends the data bits from pipe 0, and CG001 sends the data bits from pipe 1. Table 14 shows the bits received from the CH options. This table also shows how the bits are distributed on the CH options for the cache memory.

Table 14. Memory Write Data to CH Options from CG Options

Option	Bits	Check Bit
CH000	0 – 3	0
CH002	4 – 7	1
CH004	8 – 11	2
CH006	12 – 15	3
CH008	16 – 19	4
CH010	20 – 23	5
CH012	24 – 27	6
CH014	28 – 31	7
CH001	32 – 35	8
CH003	36 – 39	9
CH005	40 – 43	10

The pipe 0 I terms are ICA – ICE.
The pipe 1 I terms are IDA – IDE.

Table 14. Memory Write Data to CH Options from CG Options (continued)

Option	Bits	Check Bit
CH007	44 – 47	11
CH009	48 – 51	
CH011	52 – 55	
CH013	56 – 59	
CH015	60 – 63	

The pipe 0 I terms are ICA – ICE.
The pipe 1 I terms are IDA – IDE.

Read Data

On a read data request, each of the 16 CH options receives 8 bits of data from memory and 12 bits of destination code. The CH option then initially determines to which of the following areas the data should be sent:

- Vector even data
- Vector odd data
- B/T data
- Cache data
- A/S data
- Fetch/IO data
- Exchange

Table 15 and Table 16 define the bit layout for CH read data. The I terms apply only to read data.

Table 15. CH Option Read Data from CJ Options (Sections 0, 1, 6, and 7)

Options	Bits	SEC 0	SEC 1	SEC 6	SEC 7
CH000	0 – 3 32 –35	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH002	4 – 7 36 – 39	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH004	8 – 11 40 – 43	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH006	12 – 15 44 – 47	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH008	16 – 19 48 – 51	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH010	20 – 23 52 – 55	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH012	24 – 27 56 – 59	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH014	28 – 31 60 – 63	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH

Table 16. CH Option Read Data from CJ Options (Sections 2, 3, 4, and 5)

Option	Bits	SEC 2	SEC 3	SEC 4	SEC 5
CH001	0 – 3 32 – 35	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH003	4 – 7 36 – 39	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH005	8 – 11 40 – 43	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH007	12 – 15 44 – 47	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH009	16 – 19 48 – 51	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH011	20 – 23 52 – 55	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH013	24 – 27 56 – 59	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH
CH015	28 – 31 60 – 63	IGA–IGD	IIA–IID	IKA–IKD	IMA–IMD
		IGE–IGH	IIE–IIH	IKE–IKH	IME–IMH

The data, with the exception of cache data, is then routed to a FIAO or FIFO buffer where it is stored until a code sent from the CK option notifies the CH options which buffer to read. The output of the CH option then signals the CH option’s FIAO or FIFO to close and wait for the next code, as illustrated in Figure 4.

NOTE: There is one FIAO per section. The even-numbered CH options handle data from sections 0, 1, 6, and 7; the odd-numbered CH options handle data from sections 2, 3, 4, and 5.

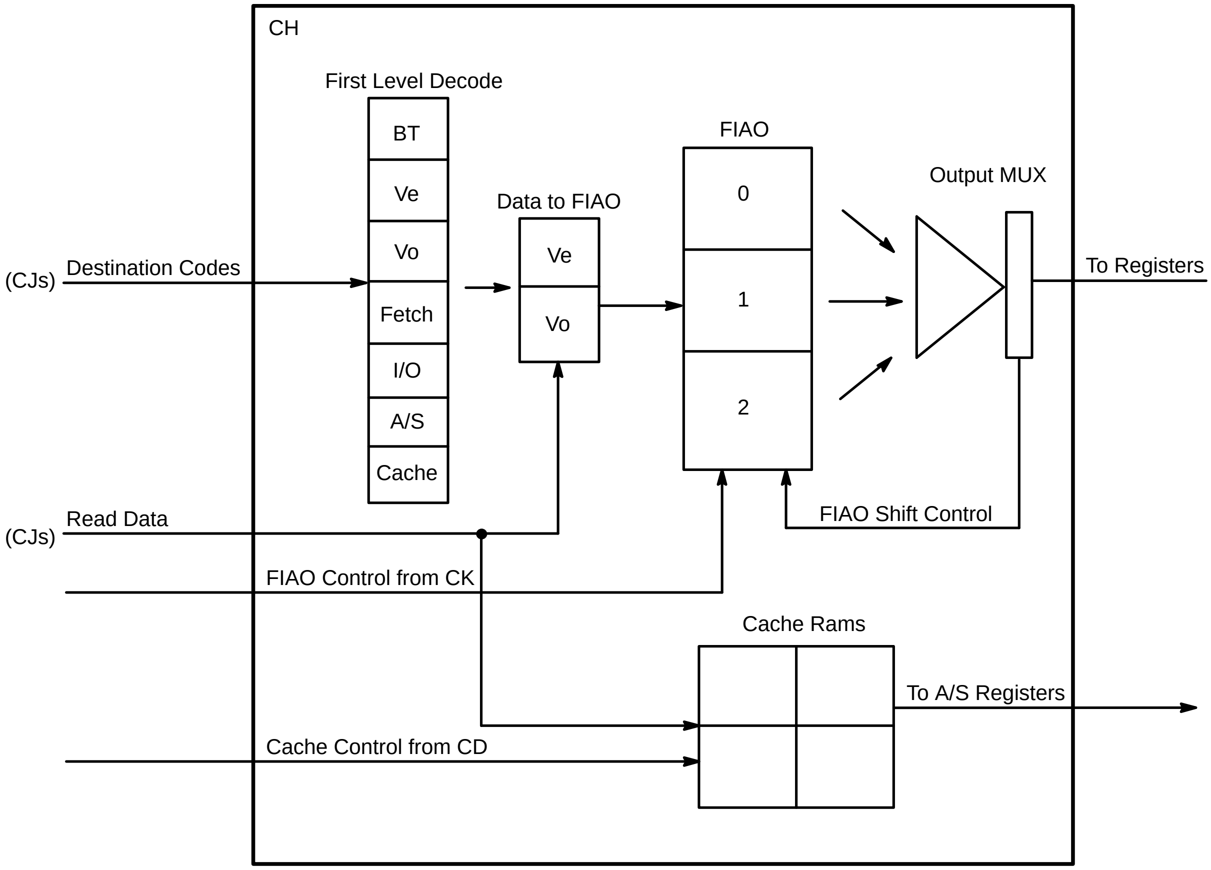


Figure 4. CH Option FIAO

The CH options also contain the random access memory (RAM) for the cache memory. When memory write data arrives from the CG options, the destination code is decoded to determine whether the data is for the cache (the CD options have already determined that the address requested generated a cache hit). The CD options then send 11 bits of cache control to address the cache; refer to Table 17 to see how the 11 bits are decoded.

Table 17. Cache Request Address

Bit	Function
0 – 1	Array (0 – 3)
2	Side
3	Upper / Lower
4 – 9	Array Address
10	Write
11	Read

Refer to Figure 5 to see how cache data is written to the arrays. This scheme keeps the full cache bandwidth available even when the system is in a degraded mode of operation, such as half-section mode.

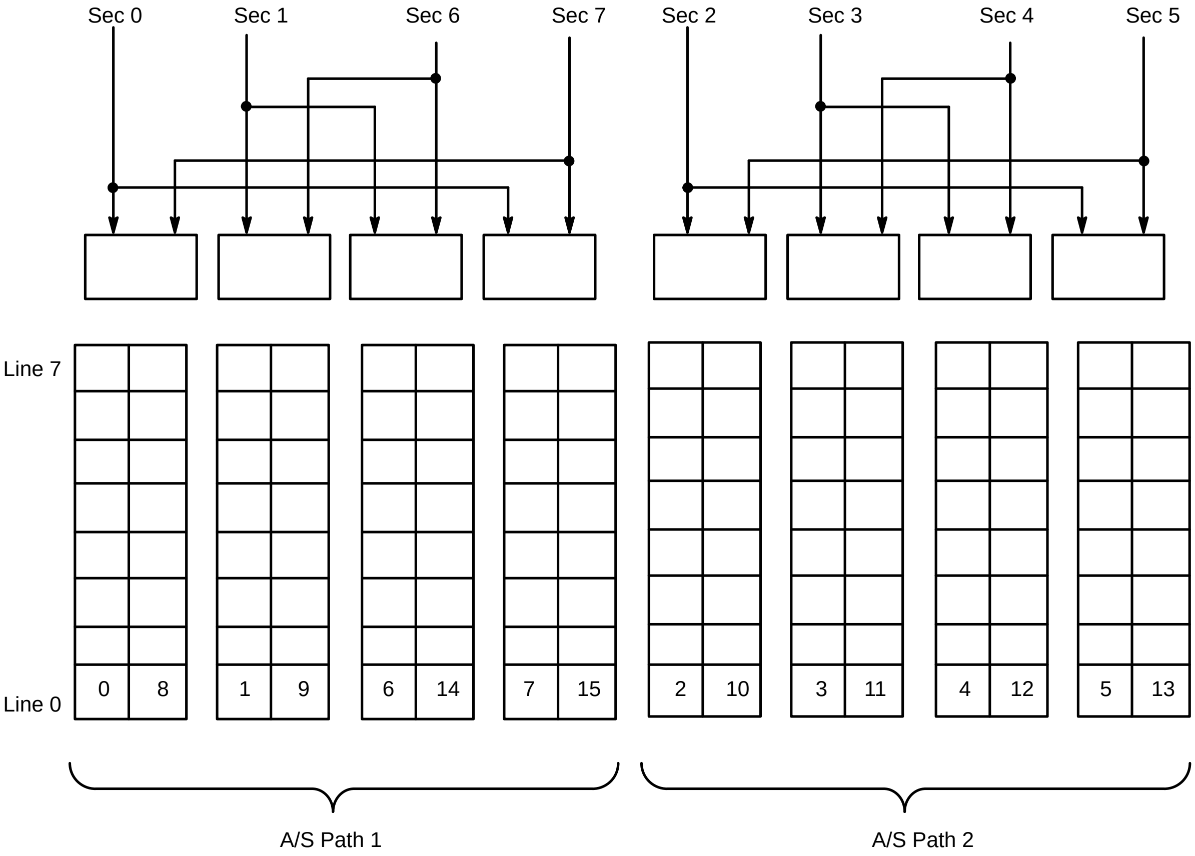


Figure 5. Data-written-to-cache Arrays

CI Option

There are eight CI options (one per memory section). The CI option is the section driver for common memory; it transmits the control and data from the CPU to the memory modules. The CI option also receives the 14 bits of destination code from the CF option and passes them to memory.

The CI options receive the Valid Request signal from the CF options and send the Valid Resume signal to the CF options. The Resume signal is sent to inform the CF option that a valid request has been honored and a Valid Reference signal is also sent to the MF options on the memory module. If the MF option buffers are full and a backup condition exists, a Not Resume signal is sent.

The CI options have six FIFO buffers that buffer input requests. This means that the CI options can hold up to six valid references without being honored by the MF options before the CI option stops sending Resume signals caused by the section conflict. All the data and control transmitted to the CI option (from the CH and CF options before the CF can be shut down) is buffered in the six FIFO buffers on the CI.

Depending on the type of data, the CH options send the write data along with either the CPU Write Data Valid signal or I/O Write Data Valid signal. These signals can be sent independently of the Valid Request signal. Refer to Table 18 and Table 19 for a list of the data bits received from the CH options and a list of the I/O and CPU valid signals.

Table 18. Write Data Bits to CI Options

Option	CI I Terms	Write Data Bits to CI000 – CI007
CH000	IAA – IAD, IDH	0 – 3, CB 0
CH001	IBN – IBQ, IDP	32 – 35, CB8
CH002	IAE – IAH, IDI	4 – 7, CB1
CH003	IBR – IBS, IDQ	36 – 39, CB9
CH004	IAI – IAL, IDJ	8 – 11, CB2
CH005	ICC – ICF, IDR	40 – 43, CB10
CH006	IAM – IAP, IDK	12 – 15, CB3
CH007	ICG – ICJ, IDS	44 – 47, CB11
CH008	IAQ – IAS, IBA, IDL	16 – 18, 19, CB4
CH009	ICK – ICN	47 – 51
CH010	IBB – IBE, IDM	20 – 23, CB5
CH011	ICO – ICR	52 – 55
CH012	IBF – IBI, IDN	24 – 27, CB6
CH013	ICS, IDA – IDC	56, 57 – 59
CH014	IBJ – IBM, IDO	28 – 31, CB7
CH015	IDD – IDG	60 – 63

Table 19. Valid Signals to CI Options

Option	CI I Terms	CPU / I/O Wdata Valid
CH000	IEM, IEN	I/O wdata valid to CI000, CI001, CI004, CI005
CH001	IEA, IEB	CPU wdata valid to CI000, CI001, CI004, CI005
CH002	IEO, IEP	I/O wdata valid to CI000, CI001, CI004, CI005
CH003	IEC, IED	CPU wdata valid to CI000, CI001, CI004, CI005
CH008	IEM, IEN	I/O wdata valid to CI002, CI003, CI006, CI007
CH009	IEA, IEB	CPU wdata valid to CI002, CI003, CI006, CI007
CH010	IEO, IEP	I/O wdata valid to CI002, CI003, CI006, CI007
CH011	IEC, IED	CPU wdata valid to CI002, CI003, CI006, CI007

Software commands configure the CI option to send the correct address to memory. Table 20 lists the commands the CI options receive.

Table 20. Configuration Codes

Code	Description
00	Set 4 section
01	Set 8 section
02	Force section bit 2
10	Force subsection bit 0
11	Force subsection bit 1
12	Force subsection bit 2
13	Force bank bit 0
14	Force bank bit 1
15	Force bank bit 2
16	Force bank bit 3
17	Clear configuration codes 10 – 16
20	Set subsection bit 0
21	Set subsection bit 1
22	Set subsection bit 2
23	Set bank bit 0
24	Set bank bit 1
25	Set bank bit 2
26	Set bank bit 3
27	Clear configuration codes 20 – 26
30	Force group bit 0
31	Force group bit 1
32	Set group bit 0
33	Set group bit 1
34	Clear group bit 0
35	Clear group bit 1
37	Clear configuration codes 30 – 33
40	Force CPU upper 256K
41	Clear force CPU upper 256K
42	Force I/O upper 256K
43	Clear force I/O upper 256K
77	Reset configuration codes

CJ Option

The CJ option performs single-byte correction/double-byte detection (SBCDBD) for the read data path from memory. There are eight CJ options (one per memory section). The CJ option is sometimes referred to as a section receiver because it is the first option to receive data from memory. The CJ options are located next to the connectors and receive the read data directly from the connectors.

There are eight holding ranks called *stacks* on the input to the CJ option. These stacks store the input data and destination codes when memory backup occurs. Backup starts when the CJ option stops sending resume signals back to memory.

The destination codes that accompany the memory references are sent to the CK option, which then sends Return Resume signals to the CJ option by reference type. The CJ option can hold the following number of references for a particular type:

- 3 Vector even element references
- 3 Vector odd element references
- 2 B/T references
- 1 A/S reference
- 1 Fetch/exchange reference
- 1 I/O reference

If the resume counter on the CJ option has reached a maximum count for a particular type of reference but the reference being held in stack 0 is of a different type, it is sent to the CK options. Two clock periods later, the corrected data is sent to the CH options along with an abbreviated copy of the destination code (bits 0 through 10 only.)

The CJ options can be configured in half mode and upper/lower configurations. This is done using configuration codes and sanity codes. If a CJ option does not receive sanity code, it ignores all inputs to it. The CJ option also receives the following maintenance mode functions that affect SBCDBD:

- Code 172 Disable Error Correction
- Code 173 Transmit Check Bits to V*i*
- Code 174 Transmit Check Bits to I/O
- Code 175 Disable Error Log

NOTE: To invoke the function executed by a 173 or 174 instruction, error correction must first be disabled.

There are other maintenance codes that the CJ option acts on; the different rates are listed below. These codes change the rate at which Resume signals are sent back to memory. By changing the rate, the outbound memory port is forced into backup for testing. More than one mode can be selected; if this is the case, the delays are added together.

- Code 164 4-CP Resume Delay
- Code 165 16-CP Resume Delay
- Code 166 63-CP Resume Delay

Error Correction

The CJ options are also responsible for error detection and correction. The CJ options receive 64 read-data bits plus 12 read check bits from memory. The check bits are regenerated from the 64 data bits coming in from memory and then combined by an exclusive OR function with the check bits from memory to produce a syndrome. If the syndrome is zero, no error occurred. If any of the bits of the syndrome are set, an error has occurred and the syndrome must be decoded to determine the error type and the bits in error.

For a detailed explanation of how the error-correction algorithm works, refer to the *CM02 Memory Module* document.

CK Option

The CK options monitor everything that comes into the CPU from memory; the CK options monitor both the left and right sections of memory. CK000 and CK001 control the operation of the cache. The memory traffic is reviewed and sorted according to the following types:

- Vector even data
- Vector odd data
- B/T data
- Cache data
- A/S data
- Fetch and IO data
- Exchange

Each type of data has its own type of arbitration rules in order to move as much data as possible.

NOTE: The terms *left* and *right* refer to a particular set of memory sections. For example:

- Left refers to memory sections 0, 1, 6, and 7
- Right refers to memory sections 2, 3, 4, and 5

B and T Register Arbitration

The B and T register rules of arbitration are as follows:

- B left and T right (both at the same time)
- T left and B right (both at the same time)
- B left or B right (one at a time)
- T left or T right (one at a time)

Fetch and I/O Arbitration

The rules for fetch and I/O arbitration are as follows:

- Left even element and right odd element
- Left odd element and right even element
- Left even or right even
- Left odd or right odd

Exchange Arbitration

The rules for exchange arbitration are as follows:

- Left low or right low
- Left high and right high
- Left low and right high
- Left high and right low

NOTE: *Low* refers to words 0 through 17 in the exchange package or the parameter words. *High* refers to words 20 through 37 in the exchange package or the A and S register values.

Vector Arbitration

The CK options handle the following vector data:

- CK000 Vector even data for sections 0, 1, 6, and 7
- CK001 Vector even data for sections 2, 3, 4, and 5
- CK002 Vector odd data for sections 0, 1, 6, and 7
- CK003 Vector odd data for sections 2, 3, 4, and 5

The rules for vector arbitration are as follows:

- One operand from left side (sections 0, 1, 6, and 7)
- One operand from right side (sections 2, 3, 4, and 5)
- The contents of the destination registers must not be equal.

A and S Arbitration

The CK options A and S register arbitration rules are as follows:

- One left and/or one right
- Cache register will override

FIAO Operation

Figure 6, Figure 7, Figure 8, and Figure 9 show how the FIAOs work. This scheme would not be possible without the use of the destination code that accompanies all data coming from memory. This code notifies the CPU of the data destination and allows the FIAO buffers to write data to the vector register out of order. There are four entries in the FIAOs; however, there is enough logic to evaluate only three positions. The FIFOs feed directly into the priority logic, and from there the destination code fills the first available space in the FIAO.

The FIAOs have destination codes from Vector 0 and Vector 1. The CK options try to control the transfer of as much data as possible, using the above-mentioned arbitration rules. In Figure 6, the priority pointer is pointing to the left, so the CK options try to read out V0 from the left and one operand from the right. If a conflict exists, the CK option then checks through the FIAO to find the first location that can be read out. In this case, it is the element in position 1, so the CK options read out L (left) 0 and R (right) 1.

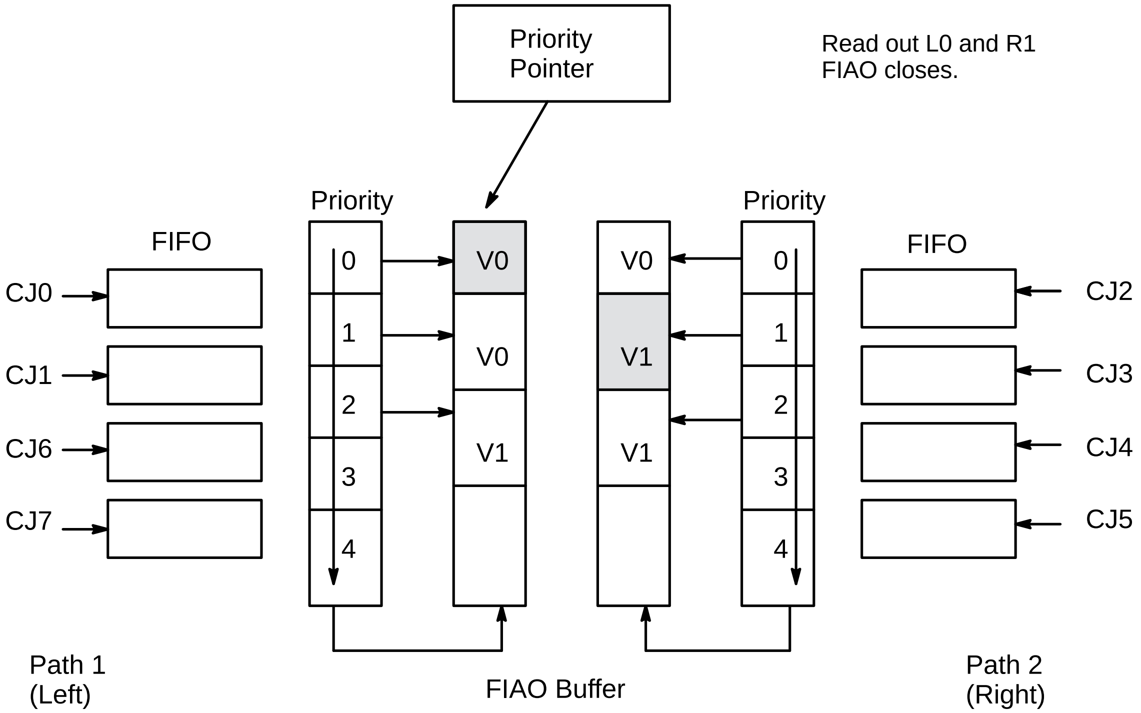


Figure 6. CK Option FIAO Operation (Part 1 of 4)

In Figure 7, because of the previous conflict, the priority pointer has moved to the right. The FIAO ranks have been closed, and another conflict exists. The readout will now be R0 and L1.

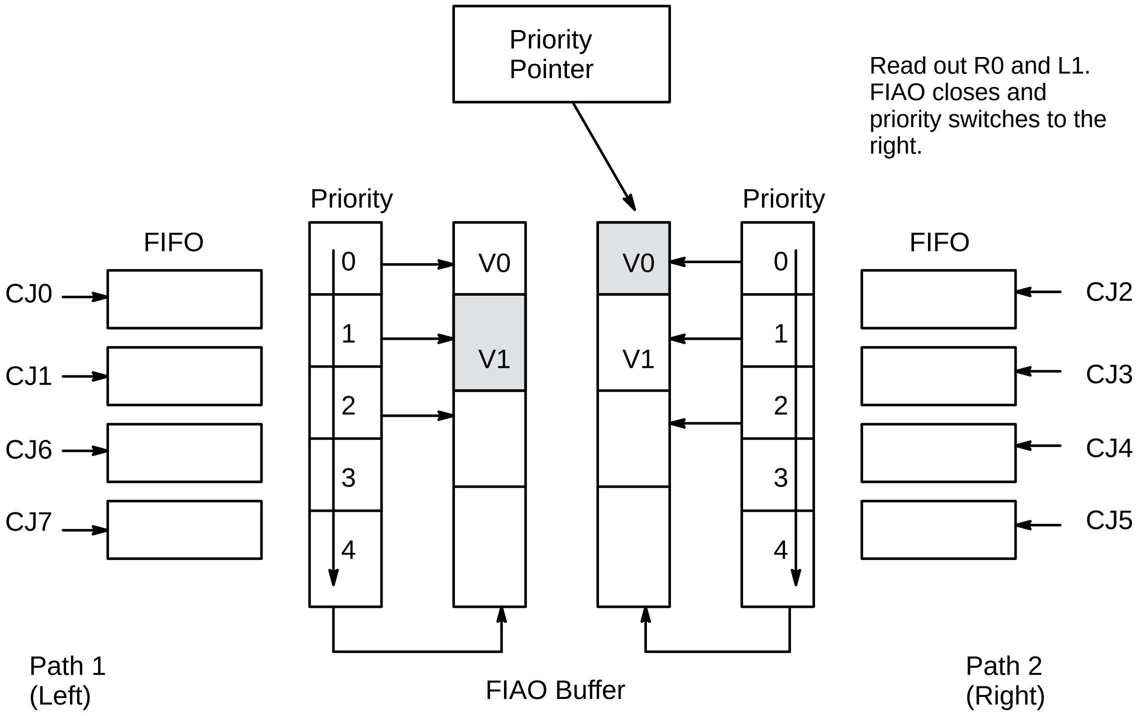


Figure 7. CK Option FIAO Operation (Part 2 of 4)

In Figure 8, because of the previous conflict, the priority pointer has again moved to the left. The FIAO ranks have been closed, and this time, no conflict exists. The readout will now be L0 and R0.

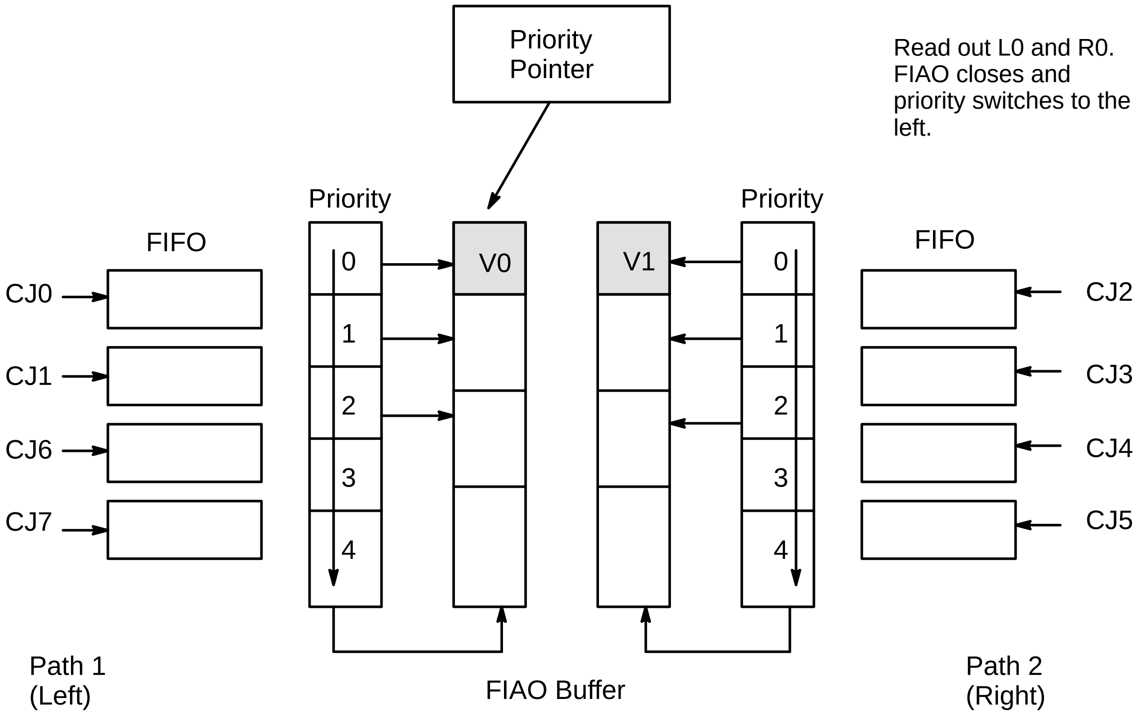


Figure 8. CK Option FIAO Operation (Part 3 of 4)

In Figure 9, the priority pointer is pointing left and a conflict exists with all elements in the right FIAO. In this case, the CK option checks the right FIAO and tries to arbitrate the conflict. As shown, the readout will be R0 and L1.

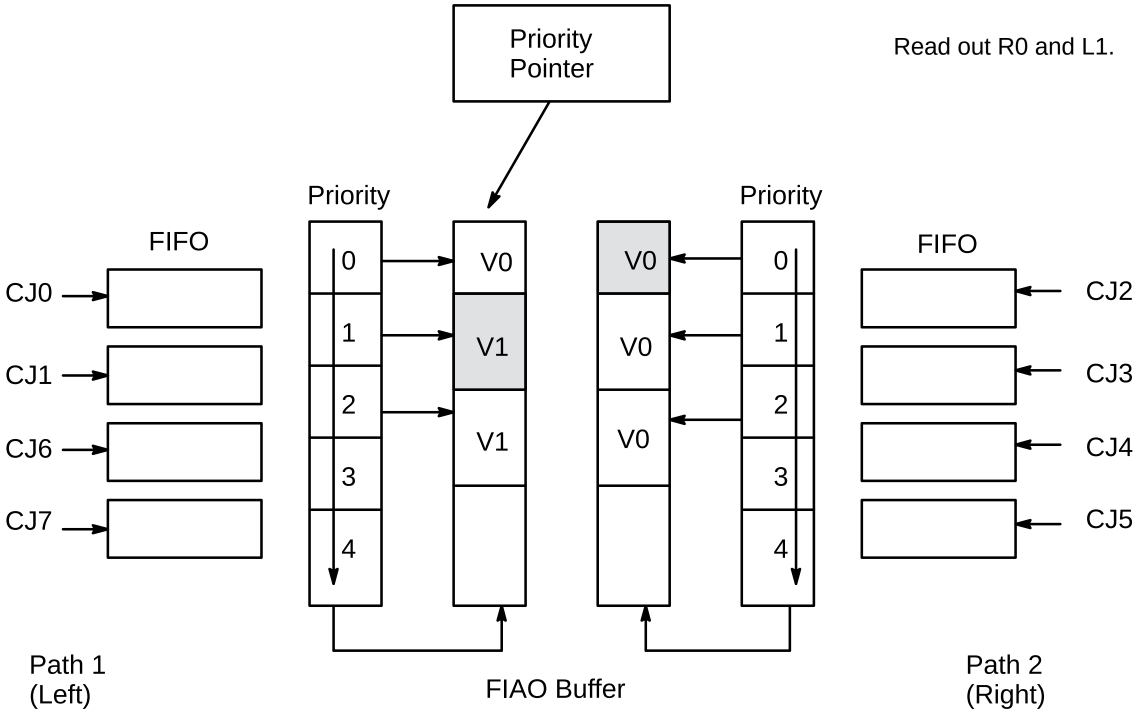
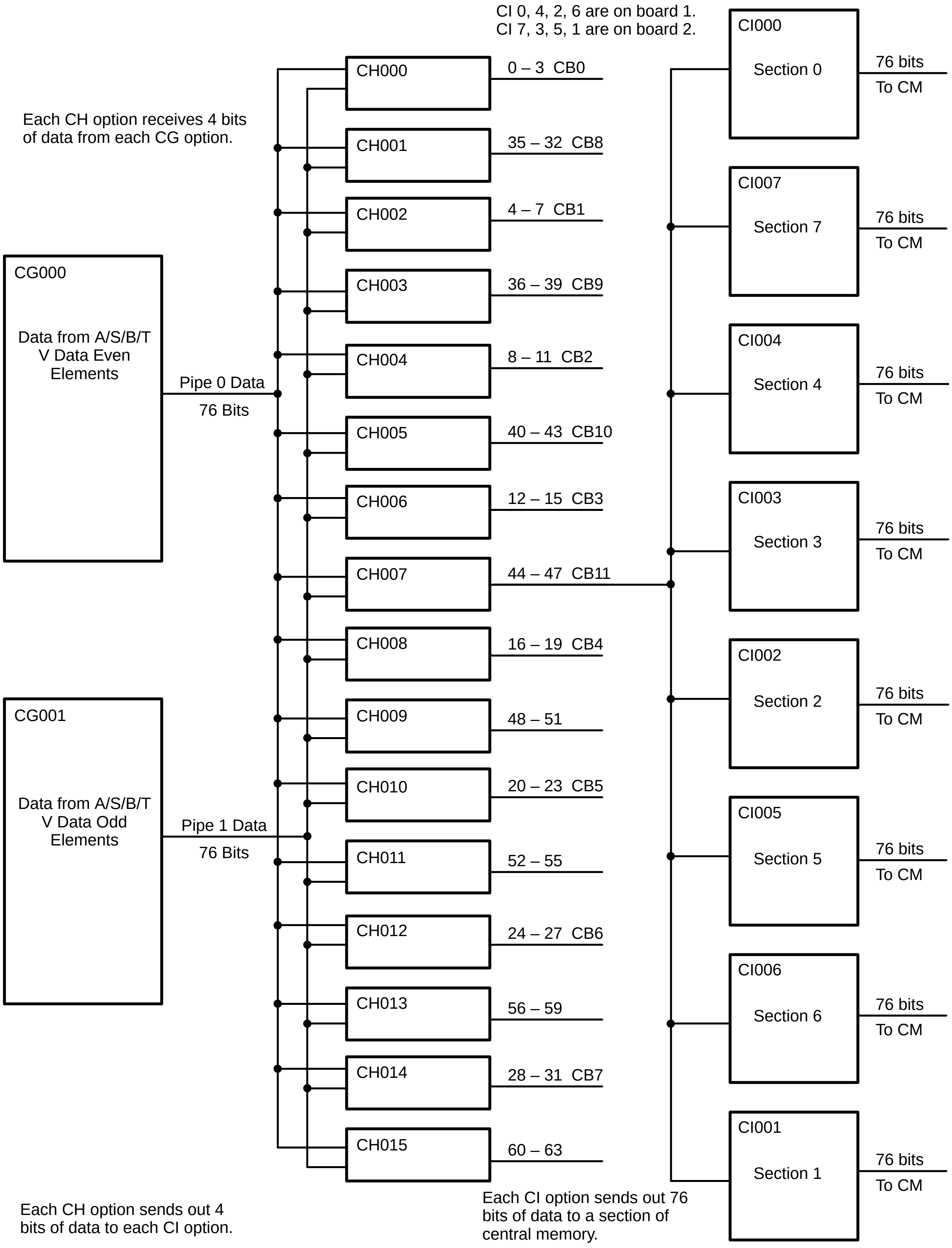


Figure 9. CK Option FIAO Operation (Part 4 of 4)

Scalar Memory Write Data Path

Refer to Figure 10 for an illustration of the write data path to memory. The CG options receive data from A/S/B/T and V registers. Both options receive all the A/S/B/T data. CG000 data bits 0 through 3 go to memory and cache, while data bits 32 through 35 go only to cache. CG001 data bits 32 through 35 go to memory only, while data bits 0 through 3 and 32 through 35 go to cache.

Each CH option receives 4 bits of data plus a check bit. One copy of the A/S/B/T data goes to memory; the other copy is written to the cache RAMs on the CH options. The CH options output 4 data bits plus a check bit to all eight CI options, which are the section drivers to memory. Each CI option receives 76 bits of data from 16 CH options and sends the data to common memory.



CI 0, 4, 2, 6 are on board 1.
CI 7, 3, 5, 1 are on board 2.

Each CH option receives 4 bits of data from each CG option.

Each CH option sends out 4 bits of data to each CI option.

Each CI option sends out 76 bits of data to a section of central memory.

Figure 10. Write Data to Memory

Scalar Memory Read Control Path

When a scalar or address register makes a request for data from memory, the following actions occur:

- The JA issues the scalar load instruction, which makes a register reservation, obtains an address from an A register, and obtains the constant data.
- The CD obtains this information and creates a destination tag.
- The CD calculates the address and performs a LAT compare and a cache check.
- At the same time that the request is being checked for a cache hit or miss, a copy of the address is sent to memory. This is necessary in case a miss sequence is generated.
- If a hit sequence occurs, the valid signal that is being sent to memory is stopped, and the rest of the request is recirculated.
- On a noncache request, the CD sends the request to the CF option, which is the CPU arbitrator.
- The CF arbitrates the request and sends it to the CI option, which is the interface to the memory module.

Refer to Figure 11 through Figure 15 for related read and write information.

Vector Store Operation: Chunking

All vector load operations (memory reads) use a process called chunking. A chunk is a group of 8 words (or elements) that is delivered as a unit to the vector register.

The CK options control the chunking process. Each of the four CK options receives 6 bits of chunk information from all of the CK options. These 6 bits are the vector register value and the element number. Each CK option is responsible for two vector registers.

- CK0 handles V0 and V4
- CK1 handles V1 and V5
- CK2 handles V2 and V6
- CK3 handles V3 and V7

The element bits that arrive at each CK option from the CK options are bits 3 through 6. These bits allow the CK option to keep track of eight word chunks.

The counters on the CK option count elements from memory; the counter counts from 0 to 7 and then goes to overflow. When overflow is detected, the CK option sends the VN Chunk Done signal to the VA options, which then start the write sequence to the selected vector register. If the vector length designates an odd number of words are being transferred, the VA option sends the Release V cm Read signal to the CK options after it has received all the elements specified by the vector length. This signal clears all the counters on the CK options and enables them for the next operation.

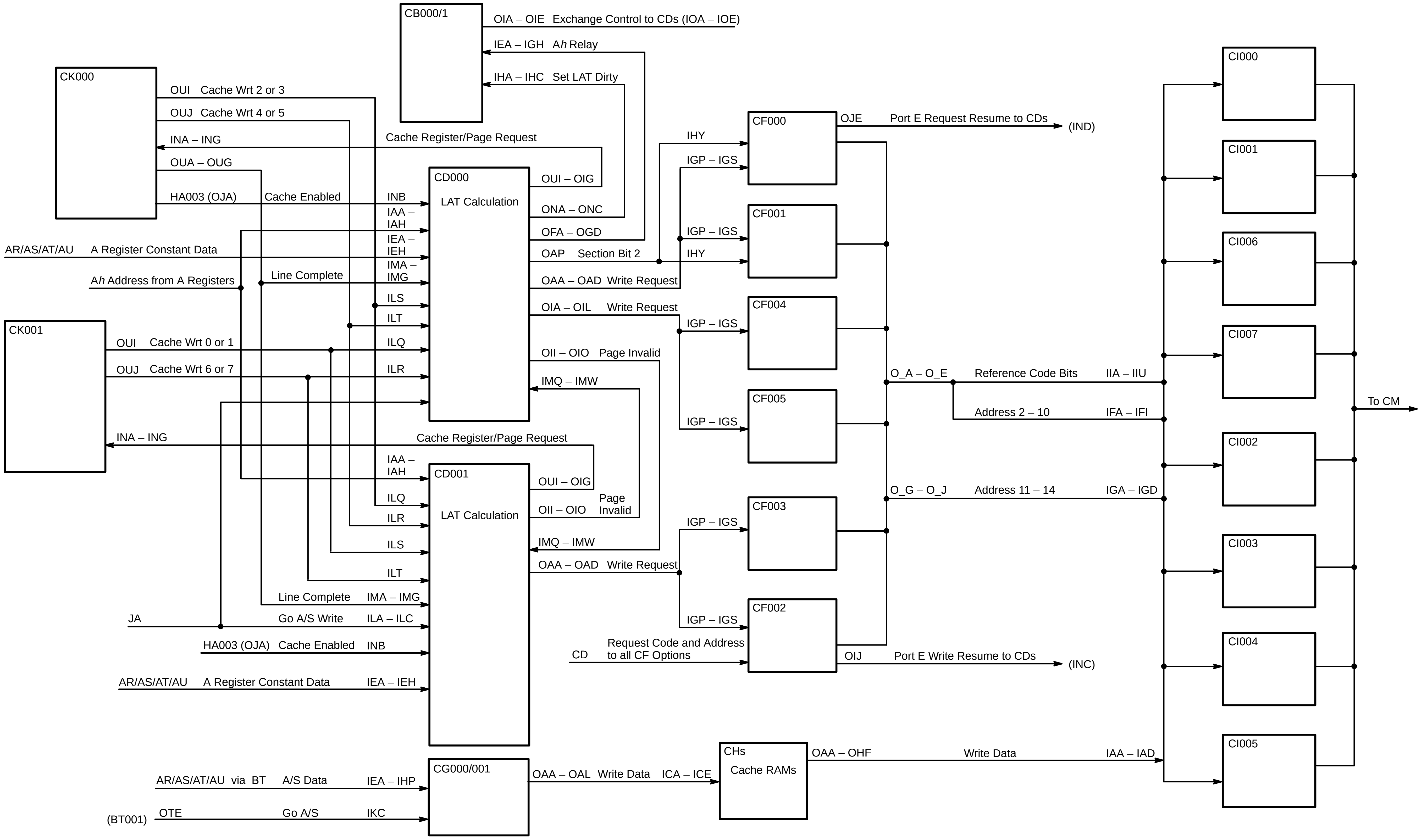


Figure 11. A/S Register Memory Write and Read Request Path

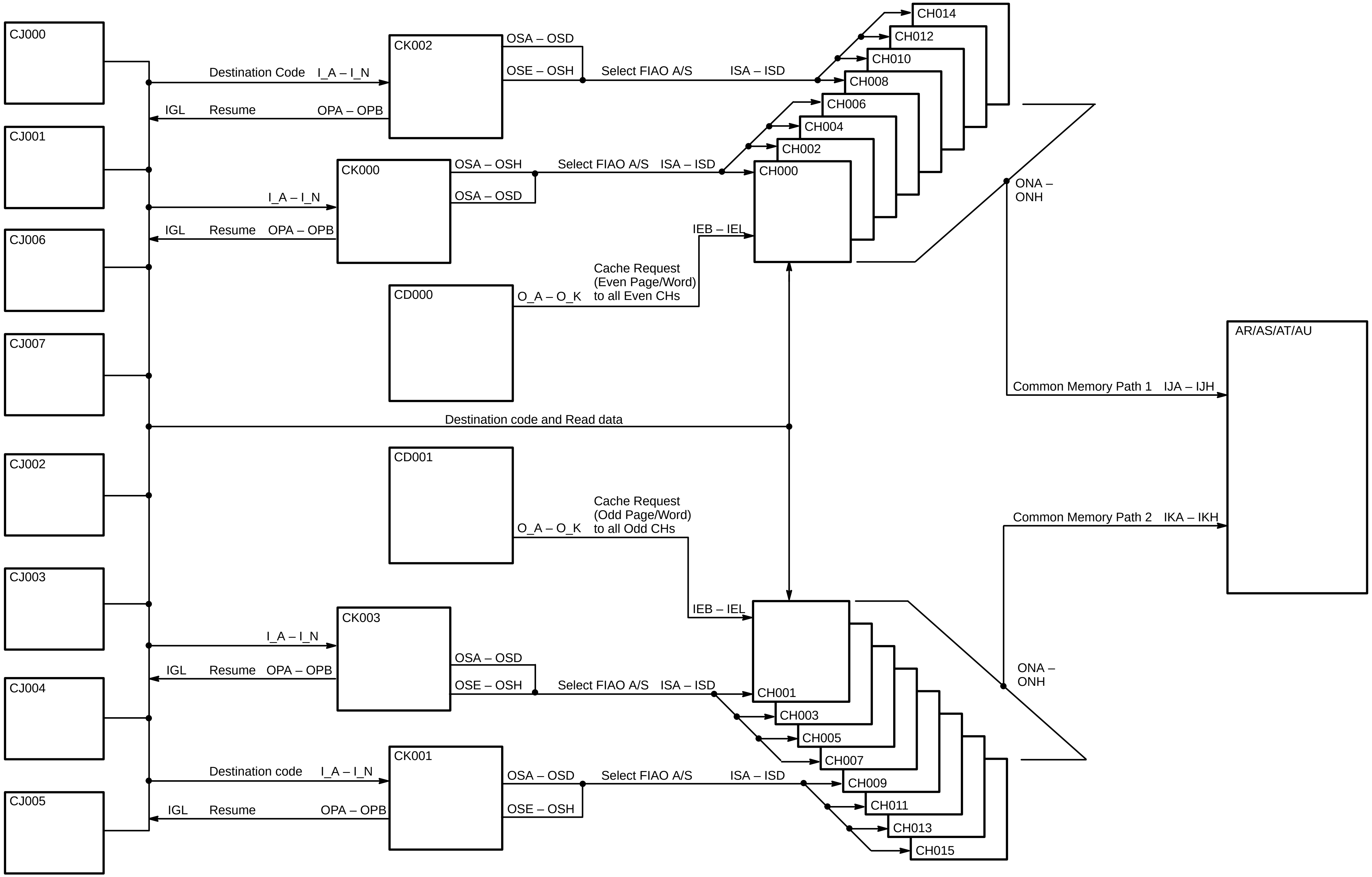


Figure 12. A/S Register Memory Read

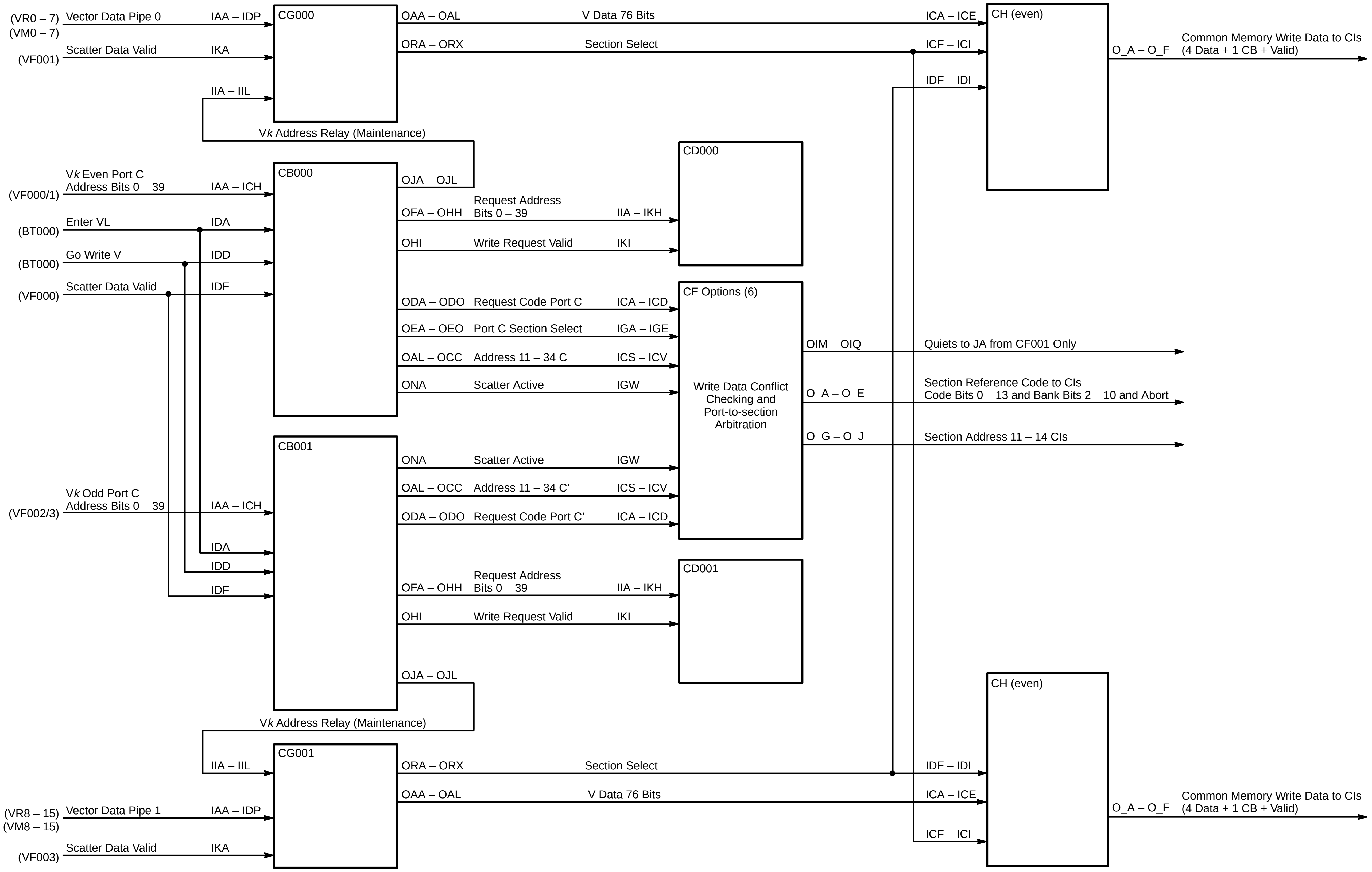


Figure 13. Vector Write

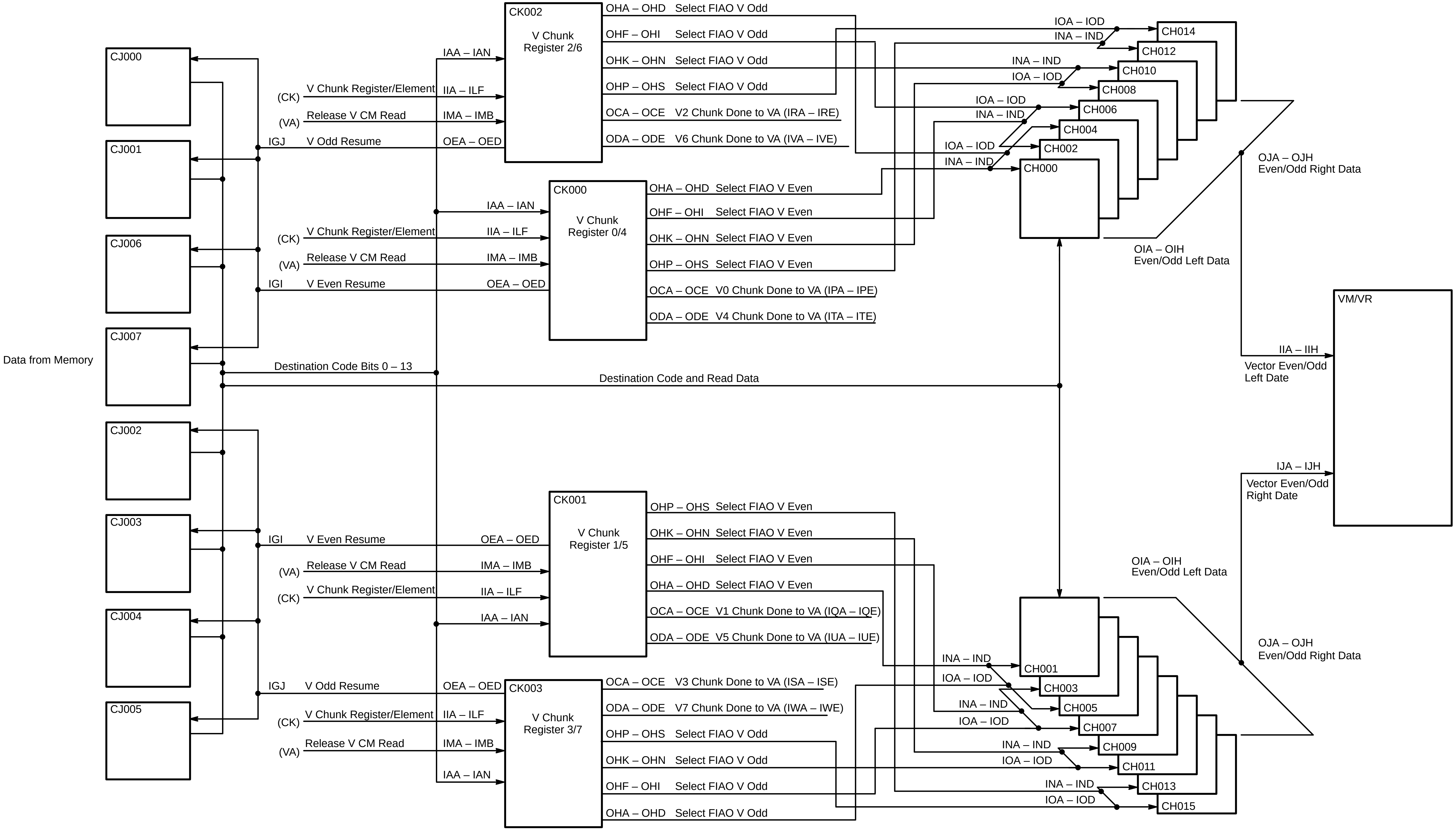

CK002
V Chunk Register 2/6
OHA – OHD Select FIAO V Odd
OHF – OHI Select FIAO V Odd
OHK – OHN Select FIAO V Odd
OHP – OHS Select FIAO V Odd
OCA – OCE V2 Chunk Done to VA (IRA – IRE)
ODA – ODE V6 Chunk Done to VA (IVA – IVE)
CJ000
CJ001
CJ006
CJ007
CJ002
CJ003
CJ004
CJ005
IAA – IAN
(CK)
V Chunk Register/Element
IIA – ILF
(VA)
Release V CM Read
IMA – IMB
IGJ
V Odd Resume
OEA – OED
CK000
V Chunk Register 0/4
OHA – OHD Select FIAO V Even
OHF – OHI Select FIAO V Even
OHK – OHN Select FIAO V Even
OHP – OHS Select FIAO V Even
OCA – OCE V0 Chunk Done to VA (IPA – IPE)
ODA – ODE V4 Chunk Done to VA (ITA – ITE)
IGI
V Even Resume
Data from Memory
Destination Code Bits 0 – 13
Destination Code and Read Data
IOA – IOD
INA – IND
CH014
CH012
CH010
CH008
CH006
CH004
CH002
CH000
OJA – OJH
Even/Odd Right Data
OIA – OIH
Even/Odd Left Data
VM/VR
IIA – IIH
Vector Even/Odd
Left Date
IJA – IJH
Vector Even/Odd
Right Date
CK001
V Chunk Register 1/5
OHP – OHS Select FIAO V Even
OHK – OHN Select FIAO V Even
OHF – OHI Select FIAO V Even
OHA – OHD Select FIAO V Even
OCA – OCE V1 Chunk Done to VA (IQA – IQE)
ODA – ODE V5 Chunk Done to VA (IUA – IUE)
CK003
V Chunk Register 3/7
OCA – OCE V3 Chunk Done to VA (ISA – ISE)
ODA – ODE V7 Chunk Done to VA (IWA – IWE)
OHP – OHS Select FIAO V Odd
OHK – OHN Select FIAO V Odd
OHF – OHI Select FIAO V Odd
OHA – OHD Select FIAO V Odd
CH001
CH003
CH005
CH007
CH009
CH011
CH013
CH015

Figure 14. Vector Load Operation (Memory to Vector Register)

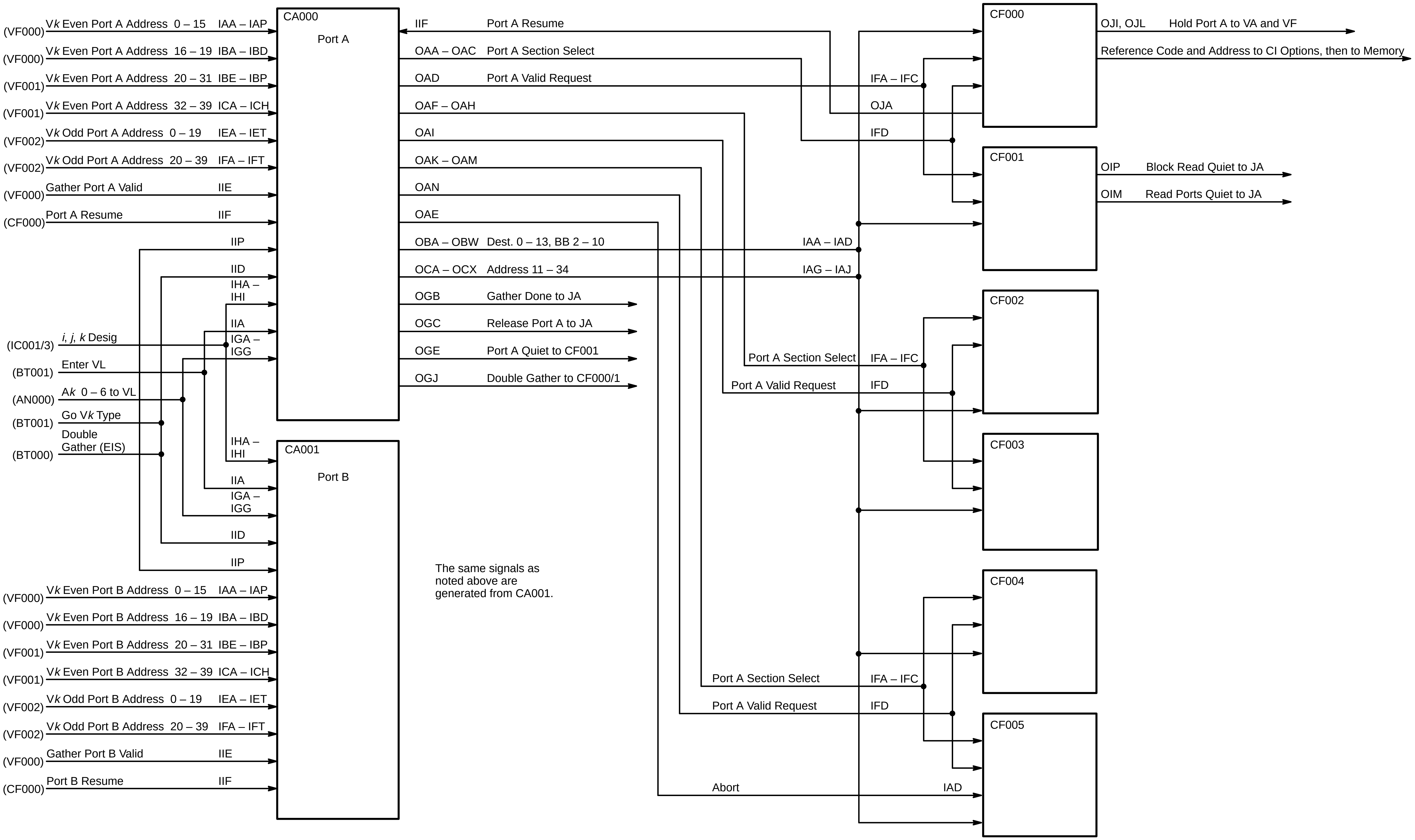

CA000
Port A
(VF000) Vk Even Port A Address 0 – 15 IAA – IAP
(VF000) Vk Even Port A Address 16 – 19 IBA – IBD
(VF001) Vk Even Port A Address 20 – 31 IBE – IBP
(VF001) Vk Even Port A Address 32 – 39 ICA – ICH
(VF002) Vk Odd Port A Address 0 – 19 IEA – IET
(VF002) Vk Odd Port A Address 20 – 39 IFA – IFT
(VF000) Gather Port A Valid IIE
(CF000) Port A Resume IIF
IIP
IID
IHA – IHI
IIA
IGA – IGG
(IC001/3) i, j, k Desig
(BT001) Enter VL
(AN000) Ak 0 – 6 to VL
(BT001) Go Vk Type
(BT000) Double Gather (EIS)
IIF Port A Resume
OAA – OAC Port A Section Select
OAD Port A Valid Request
OAF – OAH
OAI
OAK – OAM
OAN
OAE
OBA – OBW Dest. 0 – 13, BB 2 – 10
OCA – OCX Address 11 – 34
OGB Gather Done to JA
OGC Release Port A to JA
OGE Port A Quiet to CF001
OGJ Double Gather to CF000/1
CA001
Port B
IHA – IHI
IIA
IGA – IGG
IID
IIP
(VF000) Vk Even Port B Address 0 – 15 IAA – IAP
(VF000) Vk Even Port B Address 16 – 19 IBA – IBD
(VF001) Vk Even Port B Address 20 – 31 IBE – IBP
(VF001) Vk Even Port B Address 32 – 39 ICA – ICH
(VF002) Vk Odd Port B Address 0 – 19 IEA – IET
(VF002) Vk Odd Port B Address 20 – 39 IFA – IFT
(VF000) Gather Port B Valid IIE
(CF000) Port B Resume IIF
The same signals as noted above are generated from CA001.
IFA – IFC
OJA
IFD
IAA – IAD
IAG – IAJ
Port A Section Select IFA – IFC
Port A Valid Request IFD
Port A Section Select IFA – IFC
Port A Valid Request IFD
Abort IAD
CF000
CF001
CF002
CF003
CF004
CF005
OJI, OJL Hold Port A to VA and VF
Reference Code and Address to CI Options, then to Memory
OIP Block Read Quiet to JA
OIM Read Ports Quiet to JA

Figure 15. Vector Load Operation (Memory Request Path)