4.5.13 – 144 & 168 PIN PROCESSOR ENHANCED MEMORY MODULE (PEMM) FAMILIES WITH EDO–DRAM OR SDRAM

BACKGROUND INFORMATION: The Processor Enhanced Memory Module (PEMM) architecture is a modification to existing JEDEC DIMM standards which enables processing to take place directly on memory modules. This architecture is sometimes refered to as ”Basava Technology”.The applicable modules were modified in an earlier release to add signals that are needed to enable this new functionality. This Standard shows three possible block diagrams for PEMMs including these new additions/ changes.

PEMMs are intended to be designed using X64 interfaces only, on 144–pin or 168–pin DIMMs, containing either EDO–DRAM or SDRAM devices. These block diagrams are being supplied to JEDEC for reference only as a guideline for possible PEMM designs.

CAPACITY—As defined on the applicable module standards.
DATA CONFIGURATIONS—One DATA Word configuration is defined:
—64 BIT without PARITY
CONFIGURATION—3 Different Configurations are defined using various combinationa of X8 & X16 memory devices.
LOGIC FEATURES—The modules contain processor enhancement functionality.
PACKAGE—144 & 168 PIN JEDEC DIMM MEMORY MODULES
PIN ASSIGNMENTS AND PD TABLES—As defined in the applicable DIMM Standards.
PIN DEFINITIONS—As defined in the applicable DIMM Standards.
CONFIGURATION BLOCK DIAGRAM—Figs. 4.5.13–F through 4.5.13–M

The contents of this Standard are as follows:

Section 1 defines a series of terms and acronyms that are specific to the PEMM Architecture. The terms will not be added to the general terminology section in Chapter 2 of this Standard.

Section 2 contains descriptions of the PEMM Modes Of Operation.

Section 3 cintains block diagrams that are being supplied for reference only as a guideline for possible PEMM designs.

Section 4.5.13.1–PEMM specific term and acronym definitions

C0: Reference designator for the PEMM controller.

CK0L, CK0H, CK1L, CK1H: Clock signals to shared memory (low and high, 2 copies each) on PEMM. Selected from MUX M2 and M3.

CKEL, CKEH: Clock Enable signals to shared memory (low and high) on PEMM. Sourced by PEMM Controller.

CKP: PEMM processor shared memory clock source signal.

M0L–M1L, M0H–M1H: Reference designators for 24–to–12 bit FET based MUXes. Used on the PEMM for selecting control lines to shared memory.

M2–M3: Reference designators for 4x 2–to–1 bit FET based MUXes. Used on the PEMM for selecting the clock source to shared memory.

$\overline{\text{MIRQ}}$: New card edge signal standardized for the PEMM to request an interrupt from the host.

$\overline{\text{MWAIT}}$: New card edge signal standardized for the PEMM to alert host to stall current memory access.

P0: Reference designator for the PEMM processor.

PA[n..0]: PEMM processor shared memory address bus.

PBA[m..0]: PEMM processor shared memory bank select bus.

$\overline{\text{PCAS}}$: PEMM processor shared memory CAS signal.

PDQ[31:0]: PEMM processor shared memory data bus.

PDQM[3..0]: PEMM processor shared memory data byte mask bus.

PEMM: Abbreviation for Processor Enhanced Memory Module.

PEMM controller: A device on the PEMM that controls the PEMM's general functions, such as: shared memory control, program download control, program execution control, etc.

PEMM processor: The processing device that is present on the PEMM.

$\overline{\text{PRAS}}$: PEMM processor shared memory RAS signal.

Processor Enhanced Memory Module: A 168–pin DIMM or 144–pin SO–DIMM with an embedded processor and shared system memory. PEMMs can be designed using SDRAM, or EDO DRAM, using X64 interfaces.

$\overline{\text{PS}}$: PEMM processor shared memory chip select signal.

$\overline{\text{PWE}}$: PEMM processor shared memory write enable signal.

S0A–S7A, S0B–S7B: Reference designators for X32 FET based data switches. These switches are controllable in 8–bit words. Used on the PEMM for selecting data lines to shared memory.

Section 4.5.13.2

1 PEMM Modes of Operation

The PEMM has three basic modes of operation. These modes are:

1) **Standard Mode:** Default power–up mode. (The PEMM appears to the system as a standard DIMM)

2) **Configuration Mode:** The PEMM is active, and its control registers are mapped into the main memory space

3) **Smart Mode:** Same as the configuration mode; however the embedded processor on the PEMM is actually executing

The **standard mode** is the default power–up mode of the module. In the standard mode, the module is functionally equivalent to a standard DIMM.

Before the benefits of the processor on the PEMM can be realized, the host system must first recognize the presence of the PEMM, and then activate the superset features of the PEMM by switching it from the standard mode to the configuration mode. Superset information is provided in the PEMM's SPD area to aid in the recognition and location of the PEMM at system power–up time.

The **configuration mode** is entered only after the following two requirements are met, in order, while in the standard mode:

1) The proper toggling pattern takes place (driven by the host system) on the PEMM's MIRQ signal line. This pattern needs only to occur once after power–up.

2) A specific serial signature is written to the "power–up address" location as specified in the Superset SPD area for the PEMM.

(See the "Transitioning from Standard Mode to Configuration Mode" Application Note (section 2) for more information regarding these two requirements.)

Once the configuration mode is active, the host system may configure the PEMM for operation by accessing any or all memory–mapped configuration registers, and downloading the program for the embedded processor. These memory–mapped registers control all of the PEMM's superset functions including:

– Control of the onboard shared memory

– Direct communication with the embedded processor on the PEMM

– PEMM program download and execution control

– General status monitoring and configuration

The memory–mapped registers are placed "on top" of the standard memory, starting at the SPD specified power–up address as shown in the following figure:

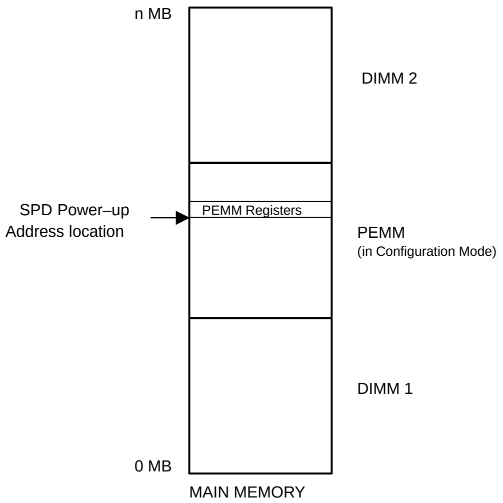


Figure 4.5.13–1: PEMM Registers shown on top of main memory

The **smart mode** can be entered from the configuration mode by simply accessing one of the memory–mapped configuration registers. Booting and execution of the embedded processor can be controlled directly via these registers.

Likewise, the standard mode can be re–entered from either the configuration mode or the smart mode at any time via control of the same memory–mapped configuration registers. It must be noted that whenever the smart mode is exited, that program execution on the PEMM will immediately halt.

The following figure shows the basic state diagram for the PEMM:

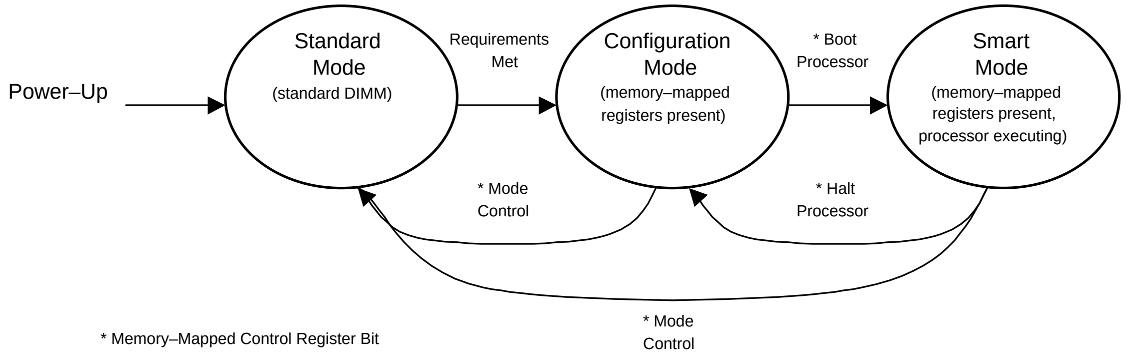

Power–Up
Standard Mode (standard DIMM)
Requirements Met
Configuration Mode (memory–mapped registers present)
* Boot Processor
Smart Mode (memory–mapped registers present, processor executing)
* Mode Control
* Halt Processor
* Mode Control
* Memory–Mapped Control Register Bit

Figure 4.5.13–2: PEMM State Diagram

2 Transitioning from Standard Mode to Configuration Mode – Application Note (information only)

The PEMM device driver is responsible for the discovery (via SPD), management, and all communication with the PEMM's memory–mapped registers. This communication includes the switching between the PEMM's operational modes.

The PEMM's default mode after power–up is the standard mode. While in the standard mode, the PEMM appears as a normal DIMM to the host system. It is important that the PEMM remain in the standard mode until specific events occur that transition it to the configuration mode.

The first event that must occur following system power–up is that the PEMM's MIRQ signal line must be toggled in the proper sequence, driven by the host system. In the standard mode, the MIRQ line on the PEMM is configured as an input. (MIRQ becomes a PEMM open–drain output when the PEMM is in the configuration mode or smart mode; thus, the host system should **never** drive the MIRQ signal when any installed PEMM is in these modes.) After the MIRQ sequence has been transmitted after power–up, the host should configure MIRQ as an input, monitoring this signal for PEMM–driven falling edges (interrupt condition).

Directly after power–up, the PEMM will enter the IDLE state and wait for the proper $\overline{\text{MIRQ}}$ sequence to be received. Once the proper sequence is received, the PEMM will transition to the IDLE_2 state. It is only from this state that the PEMM will begin to monitor the memory bus for the serial signature sequence. The MIRQ sequence only needs to be entered once after power–up as the IDLE_2 state is re–entered directly upon transitioning to the standard mode from the configuration or smart modes.

The second event (receipt of the serial signature) must occur after the IDLE_2 state has been reached. From the IDLE_2 state, the PEMM can be placed into the configuration mode by writing a "serial signature" to the power–up address specified in the PEMM's SPD. This serial signature is sent to the PEMM by the host system by writing several words to this address, in a particular order with NO reads in between each write. Reads and writes to other addresses on the PEMM, or to other DIMMs will be ignored by the PEMM Controller, thus having no effect on the validation of the Serial Signature sequence. Any read or invalid write to the PEMM's power–up address will be rejected and necessitate the Serial Signature sequence attempt to begin again.

A proper serial signature entry from any other state other than IDLE_2 has no effect. The inclusion of the $\overline{\text{MIRQ}}$ sequence as a necessity to enter the IDLE_2 state acts as an added hardware safeguard against unwanted PEMM mode transitions.

The following figure shows in detail the state diagram within the standard mode:

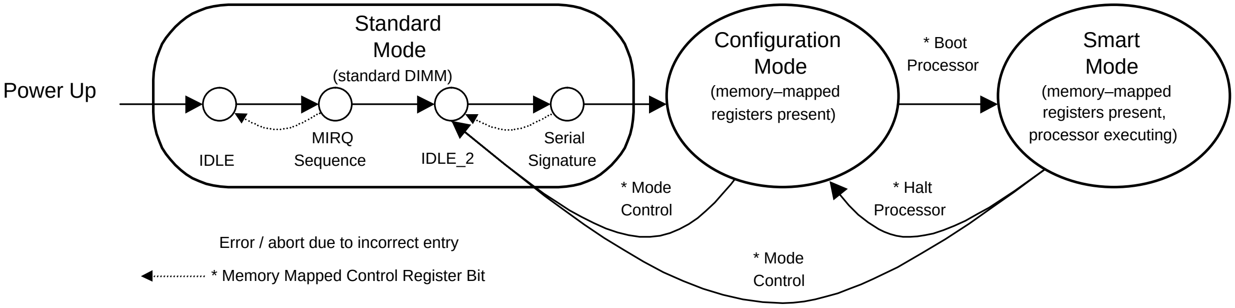


Figure 4.5.13–3: PEMM State Diagram with Standard Mode Substates Shown

2.1 $\overline{\text{MIRQ}}$ Sequence

The $\overline{\text{MIRQ}}$ sequence, driven by the host system to the PEMM, must consist of at least NEDGE falling edges and satisfy the timing requirements tSEQ, tHIGH, and tLOW. If the minimum NEDGE requirement is not satisfied within tSEQ, then the sequence is rejected and must begin again. Transition from the IDLE to the IDLE_2 state will not occur until all of these requirements are met. $\overline{\text{MIRQ}}$ need not be synchronous with any system clock.

The following figure shows an example $\overline{\text{MIRQ}}$ sequence:

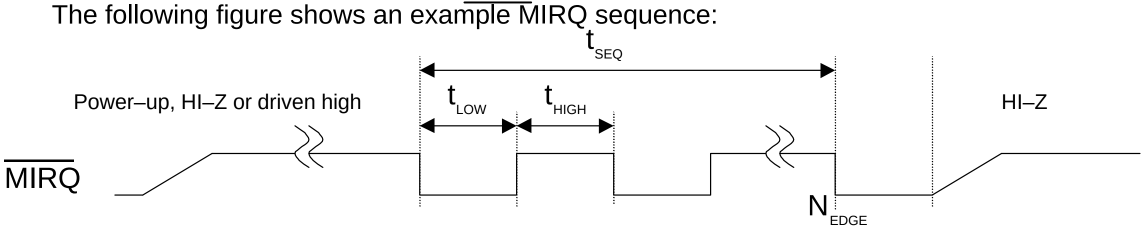


Figure 4: $\overline{\text{MIRQ}}$ Sequence

Parameter	Min	Max
tSEQ		5 us
tLOW	100 ns	
tHIGH	100 ns	
NEDGE	5	

Table 4.5.13–1: $\overline{\text{MIRQ}}$ Sequence Timing Requirements

2.2 Serial signature sequence

It is important that all data written to the PEMM during the serial signature sequence arrive in the proper intended order, and with no missing writes. For this reason, both L1 and L2 system cache must be bypassed at this power–up address (as a minimum). The PEMM device driver is responsible for all system cache management within the range of the PEMM's system address space.

Because some systems have memory data lines that are connected between the host chipset and the DIMM socket in improper order (i.e., the chipset's D0 line may not be connected to the DIMM socket's D0 line, D1 may not be connected to D1, and D2 to D2, etc...), the host will not always be able to send out a specific coherent signature in parallel to the PEMM. A signature comprising 64–bit parallel writes of only all 1's or 0's should then be used. The PEMM Controller will monitor a certain subset of the 64 available data lines, making sure that all monitored bits are equal to each other (i.e., all 1's or all 0's). Any other combination on the monitored data lines will be rejected and will necessitate the serial signature sequence attempt to begin again.

The multiple writes of all 1's or 0's to the power–up address must be completed in the proper order to match the expected code embedded in the PEMM's Controller. An incorrect sequence in the code will be rejected and will necessitate the serial signature sequence attempt to begin again.

The number of writes required to match the entire code can vary by PEMM design, as can the number of data lines monitored, but it must be noted that longer and wider codes will be more robust against unintended matches.

There is no "real time" limit for the serial sequence to complete.

The following table shows the 48–write long serial signature pattern that the host must write to properly activate a PEMM requiring the signature “42 41 53 41 56 41” (hex):

Serial Write Order	Signature		64–bit Data Value
1st Write	4	0	All 0’s (0000000000000000h)
2nd Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
3rd Write		0	All 0’s (0000000000000000h)
4th Write		0	All 0’s (0000000000000000h)
5th Write	1	0	All 0’s (0000000000000000h)
6th Write		0	All 0’s (0000000000000000h)
7th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
8th Write		0	All 1’s (FFFFFFFFFFFFFFFFh)
9th Write	4	0	All 0’s (0000000000000000h)
10th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
11th Write		0	All 0’s (0000000000000000h)
12th Write		0	All 0’s (0000000000000000h)
13th Write	1	0	All 0’s (0000000000000000h)
14th Write		0	All 0’s (0000000000000000h)
15th Write		0	All 0’s (0000000000000000h)
16th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
17th Write	5	0	All 0’s (0000000000000000h)
18th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
19th Write		0	All 0’s (0000000000000000h)
20th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
21st Write	3	0	All 0’s (0000000000000000h)
22nd Write		0	All 0’s (0000000000000000h)
23rd Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
24th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
25th Write	4	0	All 0’s (0000000000000000h)
26th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
27th Write		0	All 0’s (0000000000000000h)
28th Write		0	All 0’s (0000000000000000h)
29th Write	1	0	All 0’s (0000000000000000h)
30th Write		0	All 0’s (0000000000000000h)
31st Write		0	All 0’s (0000000000000000h)
32nd Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
33rd Write	5	0	All 0’s (0000000000000000h)
34th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
35th Write		0	All 0’s (0000000000000000h)
36th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
37th Write	6	0	All 0’s (0000000000000000h)
38th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
39th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
40th Write		0	All 0’s (0000000000000000h)
41st Write	4	0	All 0’s (0000000000000000h)
42nd Write		1	All 1’s (FFFFFFFFFFFFFFFFh)
43rd Write		0	All 0’s (0000000000000000h)
44th Write		0	All 0’s (0000000000000000h)
45th Write	1	0	All 0’s (0000000000000000h)
46th Write		0	All 0’s (0000000000000000h)
47th Write		0	All 0’s (0000000000000000h)
48th Write		1	All 1’s (FFFFFFFFFFFFFFFFh)

Table 4.5.13–2: A Typical 48–write long Serial Signature Sequence For Signature = “42 41 53 41 56 41” (hex)

Once the entire serial signature is properly received, the PEMM will immediately enter the configuration mode. After this point, the next read from the power–up address will return the inverse of the last write (the last write of the serial sequence). This provides verification to the host system that the configuration mode has been entered properly. The PEMM's memory–mapped registers will then appear atop the main memory space, starting at the power–up address, and reading and writing to these registers, as well as to the rest of surrounding memory, may proceed as normal.

After the Configuration Mode has been entered, a two–way protocol can then be executed to determine the mapping of the data lines between the host chipset and the DIMM socket. A correction for the mapping, if any, can be employed from then on by the device driver or implemented via programmable setup registers in the PEMM hardware.

The following figure shows an N–length Serial Signature state diagram for entering the Configuration Mode:

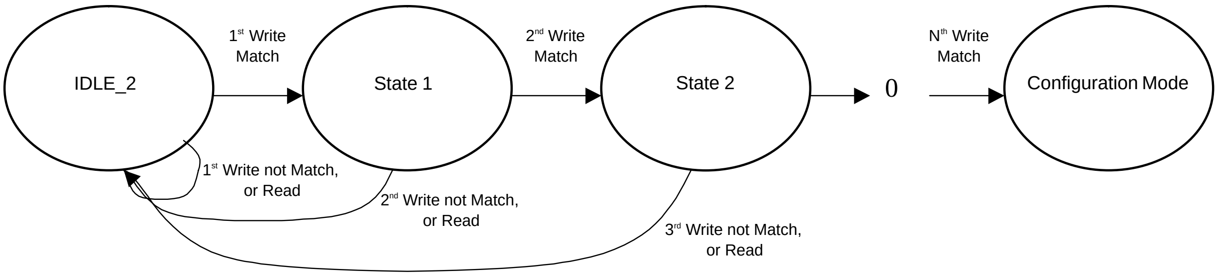

IDLE_2
1st Write Match
State 1
2nd Write Match
State 2
0
Nth Write Match
Configuration Mode
1st Write not Match, or Read
2nd Write not Match, or Read
3rd Write not Match, or Read

Figure 4.5.13–5: Serial Signature Sequence State Diagram

3 Serial Signature Sequence Robustness – Application Note (information only)

“Bullet proofing” the sequence for the PEMM to enter the configuration mode is very important. Involuntary entering into the configuration mode, not controlled by the PEMM device driver, can have adverse effects on the operating system and other programs and data residing in memory. It is important that the PEMM stay in standard mode until specifically asked to exit that mode under control of the device driver. Safeguarding against such an event is achieved by three means:

1) The OS and device driver shall know before hand (early on, at boot up time) if a PEMM is present in the system by detection through the SPD.

2) The MIRQ sequence shall serve as a hardware validation for the PEMM. The serial signature sequence will be ignored by the PEMM until the proper MIRQ sequence has been received.

3) The serial signature sequence shall be complicated enough to guard against inadvertent entries into the configuration mode

Specifically for the serial sequence, if a PEMM design is to monitor 32 of the 64 data lines for the serial signature pattern, and the matching code requirement is 48 writes long, then the probability that the PEMM will be switched to the configuration mode when it is not intended would be:

$$P = \frac{1}{(2^{32})^{48}}$$

If this number is evaluated, it can be seen that the probability of the system inadvertently writing the required sequence to match a specific signature code is infinitesimal, 1 in 2.4X10^{462}.

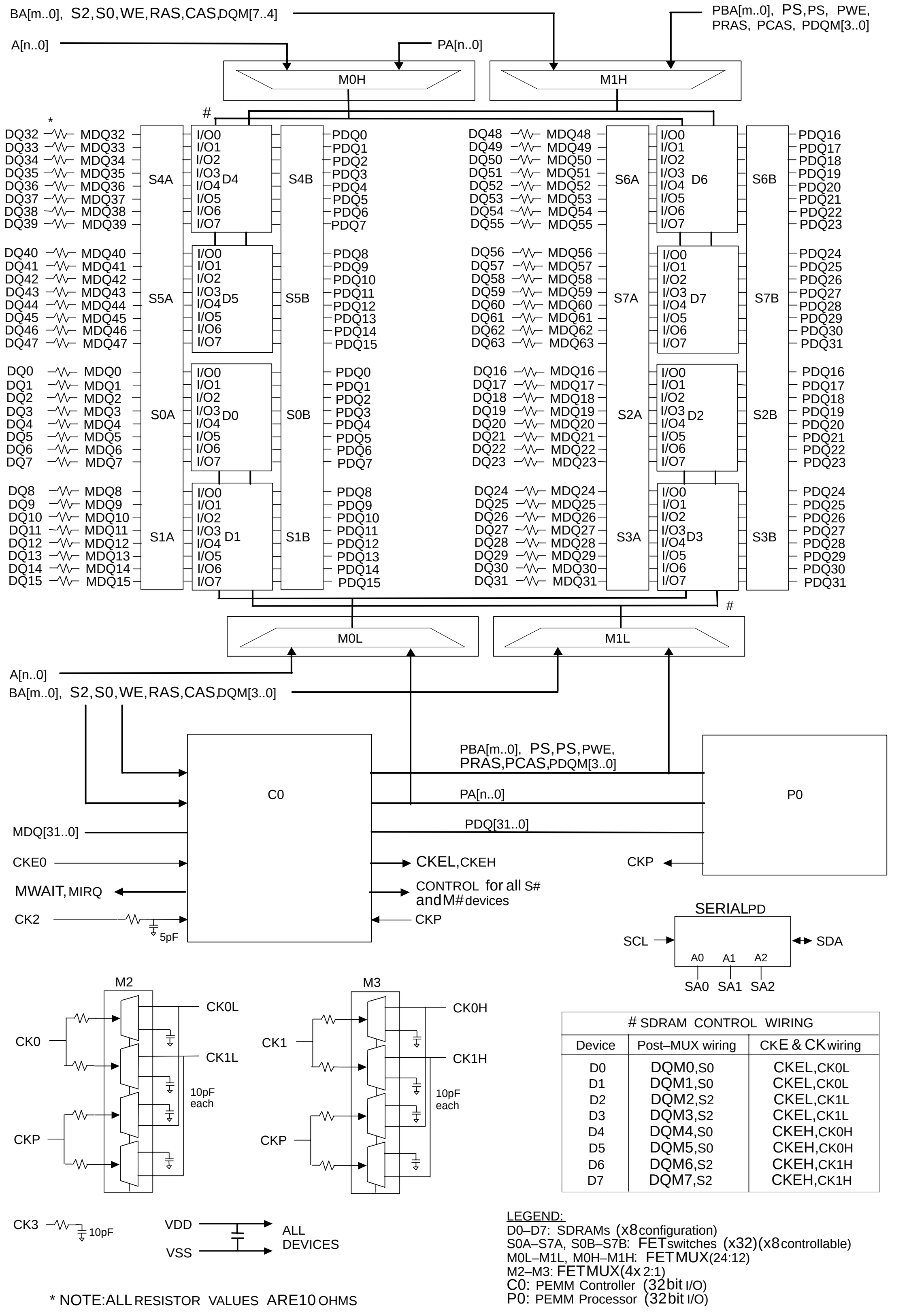
Section 4.5.13.3 – Representative Block Diagrams

The following figures are supplied for reference only, as a guideline for possible PEMM designs. Adherence to these figures is not required for designs to meet this Standard.

FIGURE 4.5.13–6: X64 SDRAM PEMM, 1 Bank with X8 SDRAMs
(1 Bank (2 x 32–bit buffers) of SDRAM is shared with a single 32–bit processor)

FIGURE 4.5.13–7: X64 SDRAM PEMM, 1 Bank with X16 SDRAMs
(1 Bank (2 x 32–bit buffers) of SDRAM is shared with a single 32–bit processor)

FIGURE 4.5.13–8: X64 SDRAM PEMM, 2 Banks with X16 SDRAMs
(1 Bank (2 x 32–bit buffers) of SDRAM is shared with a single 32–bit processor;
1 Bank of SDRAM is dedicated “host only” memory)

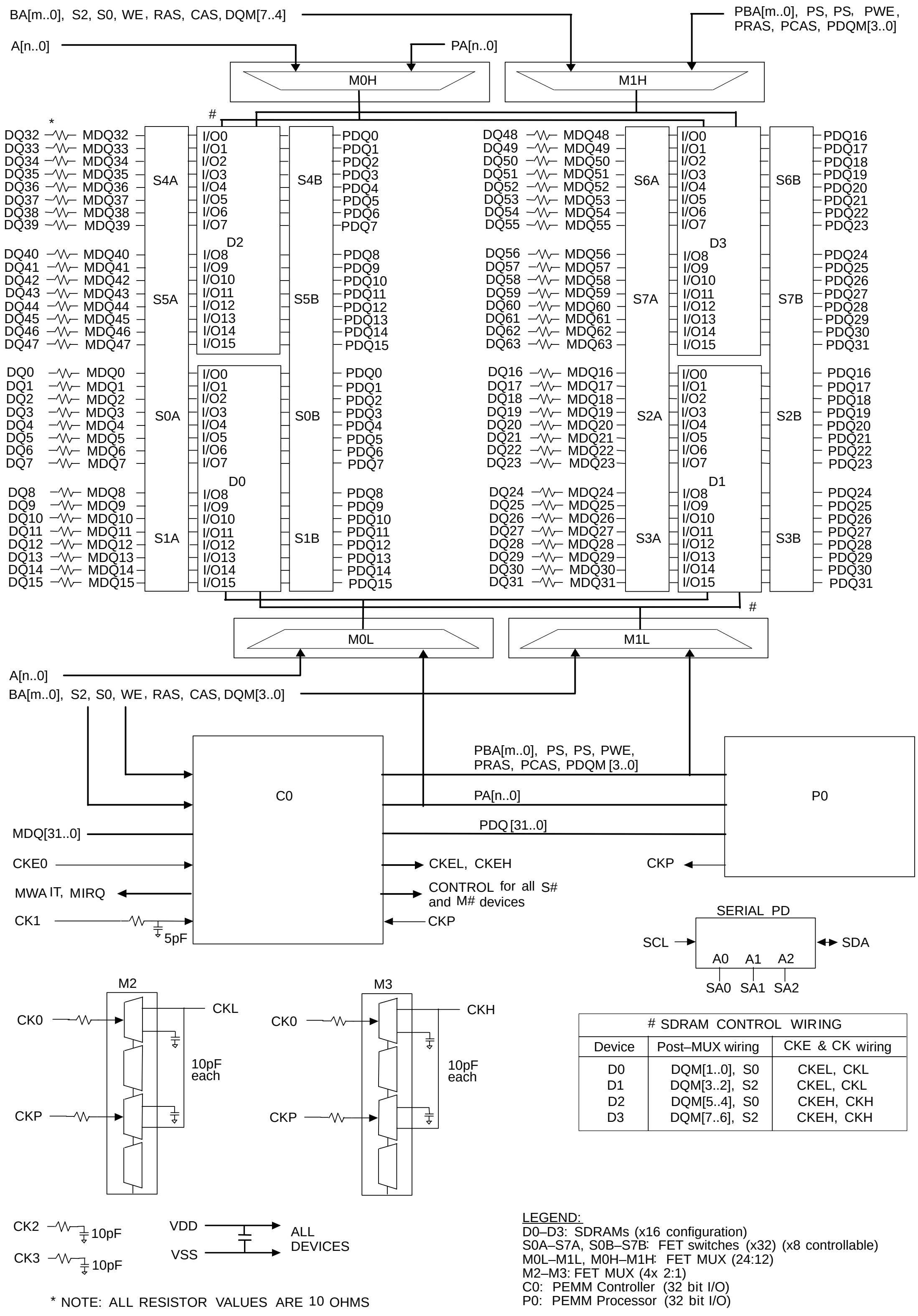


# SDRAM CONTROL WIRING		
Device	Post–MUX wiring	CKE & CK wiring
D0	DQM0,S0	CKEL,CK0L
D1	DQM1,S0	CKEL,CK0L
D2	DQM2,S2	CKEL,CK1L
D3	DQM3,S2	CKEL,CK1L
D4	DQM4,S0	CKEH,CK0H
D5	DQM5,S0	CKEH,CK0H
D6	DQM6,S2	CKEH,CK1H
D7	DQM7,S2	CKEH,CK1H

LEGEND:
D0–D7: SDRAMs (x8 configuration)
S0A–S7A, S0B–S7B: FET switches (x32)(x8 controllable)
M0L–M1L, M0H–M1H: FET MUX (24:12)
M2–M3: FET MUX (4x 2:1)
C0: PEMM Controller (32 bit I/O)
P0: PEMM Processor (32 bit I/O)

* NOTE: ALL RESISTOR VALUES ARE 10 OHMS

Figure 4.5.13–6
X64 SDRAM PEMM, 1 Bank with X8 SDRAMs

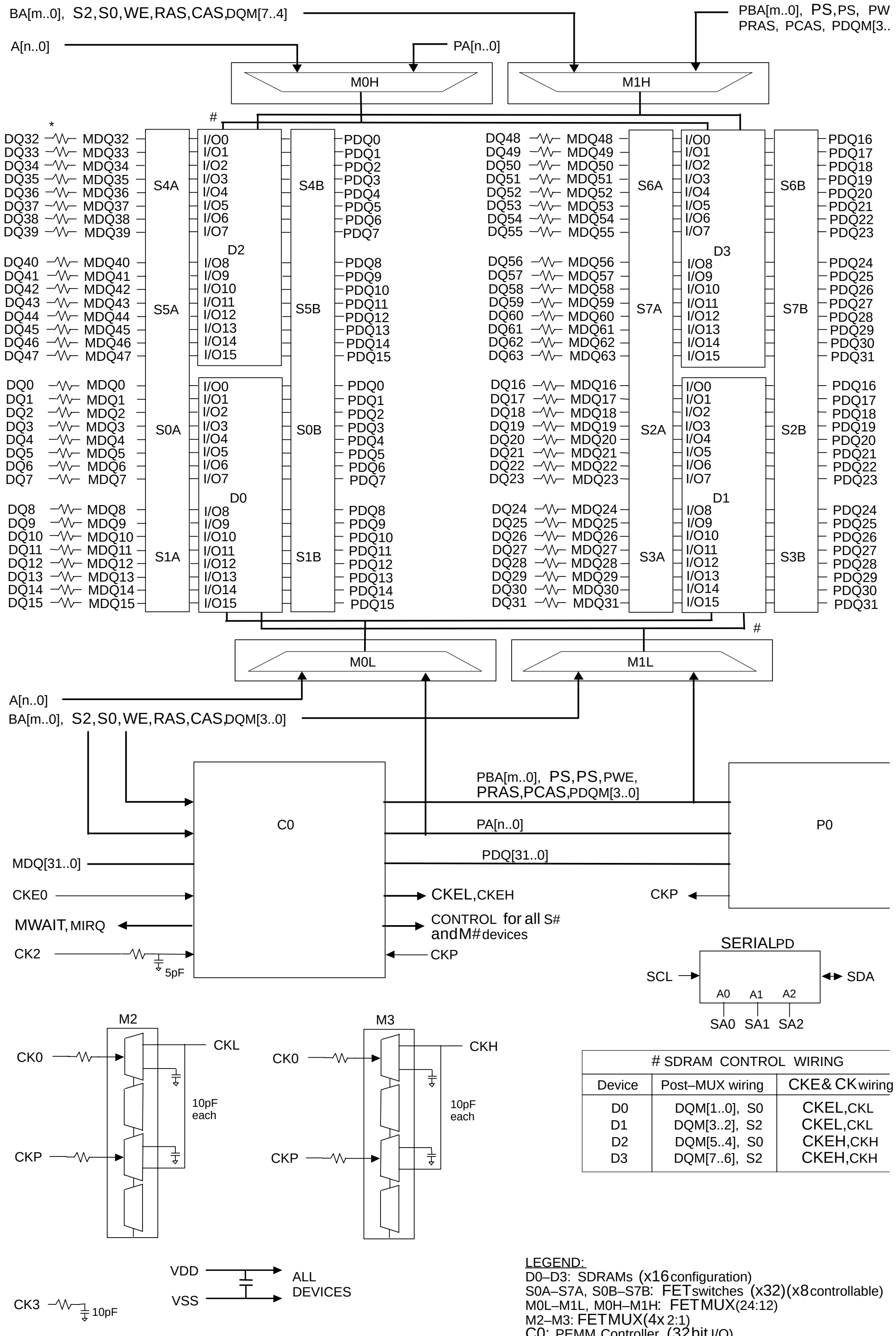


# SDRAM CONTROL WIRING		
Device	Post–MUX wiring	CKE & CK wiring
D0	DQM[1..0], S0	CKEL, CKL
D1	DQM[3..2], S2	CKEL, CKL
D2	DQM[5..4], S0	CKEH, CKH
D3	DQM[7..6], S2	CKEH, CKH

LEGEND:
D0–D3: SDRAMs (x16 configuration)
S0A–S7A, S0B–S7B: FET switches (x32) (x8 controllable)
M0L–M1L, M0H–M1H: FET MUX (24:12)
M2–M3: FET MUX (4x 2:1)
C0: PEMM Controller (32 bit I/O)
P0: PEMM Processor (32 bit I/O)

* NOTE: ALL RESISTOR VALUES ARE 10 OHMS

Figure 4.5.13–7
X64 SDRAM PEMM, 1 Bank with X16 SDRAMs

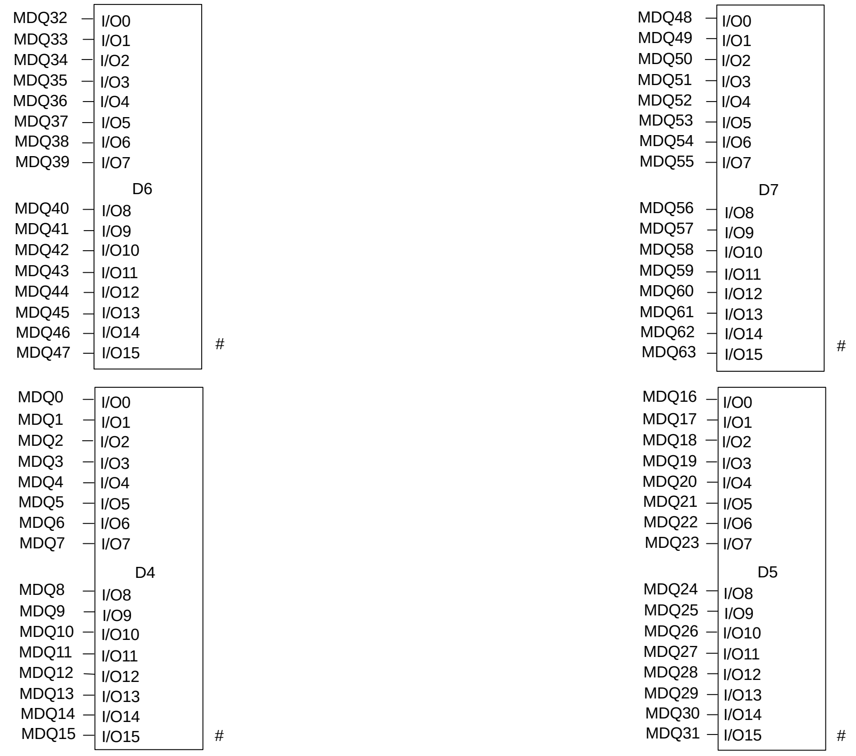


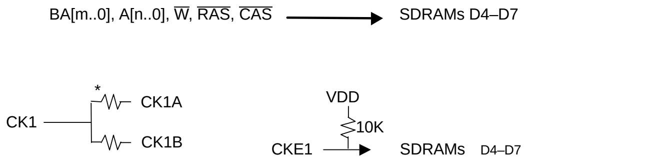
# SDRAM CONTROL WIRING		
Device	Post–MUX wiring	CKE & CK wiring
D0	DQM[1..0], S0	CKEL, CKL
D1	DQM[3..2], S2	CKEL, CKL
D2	DQM[5..4], S0	CKEH, CKH
D3	DQM[7..6], S2	CKEH, CKH

LEGEND:
D0–D3: SDRAMs (x16 configuration)
S0A–S7A, S0B–S7B: FET switches (x32)(x8 controllable)
M0L–M1L, M0H–M1H: FET MUX (24:12)
M2–M3: FET MUX (4x 2:1)
C0: PEMM Controller (32 bit I/O)
P0: PEMM Processor (32 bit I/O)

* NOTE: ALL RESISTOR VALUES ARE 10 OHMS

Figure 4.5.13–8
X64 SDRAM PEMM, 2 Banks with X16 SDRAMs


MDQ32 MDQ33 MDQ34 MDQ35 MDQ36 MDQ37 MDQ38 MDQ39
I/O0 I/O1 I/O2 I/O3 I/O4 I/O5 I/O6 I/O7
D6
MDQ40 MDQ41 MDQ42 MDQ43 MDQ44 MDQ45 MDQ46 MDQ47
I/O8 I/O9 I/O10 I/O11 I/O12 I/O13 I/O14 I/O15
#
MDQ48 MDQ49 MDQ50 MDQ51 MDQ52 MDQ53 MDQ54 MDQ55
I/O0 I/O1 I/O2 I/O3 I/O4 I/O5 I/O6 I/O7
D7
MDQ56 MDQ57 MDQ58 MDQ59 MDQ60 MDQ61 MDQ62 MDQ63
I/O8 I/O9 I/O10 I/O11 I/O12 I/O13 I/O14 I/O15
#
MDQ0 MDQ1 MDQ2 MDQ3 MDQ4 MDQ5 MDQ6 MDQ7
I/O0 I/O1 I/O2 I/O3 I/O4 I/O5 I/O6 I/O7
D4
MDQ8 MDQ9 MDQ10 MDQ11 MDQ12 MDQ13 MDQ14 MDQ15
I/O8 I/O9 I/O10 I/O11 I/O12 I/O13 I/O14 I/O15
#
MDQ16 MDQ17 MDQ18 MDQ19 MDQ20 MDQ21 MDQ22 MDQ23
I/O0 I/O1 I/O2 I/O3 I/O4 I/O5 I/O6 I/O7
D5
MDQ24 MDQ25 MDQ26 MDQ27 MDQ28 MDQ29 MDQ30 MDQ31
I/O8 I/O9 I/O10 I/O11 I/O12 I/O13 I/O14 I/O15
#

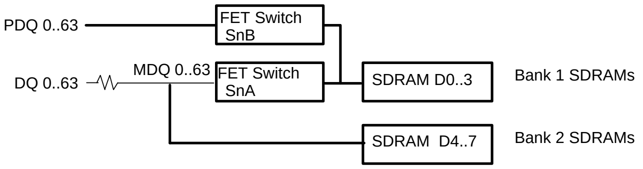


* NOTE: ALL RESISTOR VALUES ARE 10 OHMS UNLESS OTHERWISE STATED.

# SDRAM CONTROL WIRING		
Device	DQM & S wiring	CKE & CK wiring
D4	DQM[1..0], S1	CKE1, CK1A
D5	DQM[3..2], S3	CKE1, CK1A
D6	DQM[5..4], S1	CKE1, CK1B
D7	DQM[7..6], S3	CKE1, CK1B

LEGEND:
D4–D7: SDRAMs (X16 configuration)

BANK 2 DEVICES



2 Bank PEMM Data Bus Connections

Figure 4.5.13–8 (continued)
X64 SDRAM PEMM, 2 Banks with X16 SDRAMs