Freescale Semiconductor
Application Note

Document Number: AN3533
Rev. 0, 11/2007

Programming the User-Programmable Machine (UPM) for SDRAM Memory Devices

Networking and Multimedia Group
Freescale Semiconductor, Inc.
Austin, TX

Freescale devices such as the MPC8349E, MPC8540, and MPC8560 include an SDRAM machine in their local bus controller (LBC) that controls SDRAM memory devices. However, the MPC8313E, MPC8315E, MPC8572E, and MPC8610 feature an eLBC (enhanced local bus controller), which does not support an SDRAM machine. This application note explains how to program one of the eLBC's user-programmable machines (UPMs) to control an SDRAM memory device.

NOTE

These devices have different address and data bus widths, so the examples here arbitrarily assume a 32-bit data bus, a 26-bit address bus, 8 chip selects, and byte strobe signals.

Contents

freescale semiconductor

1 SDRAM Hardware Interfacing

An SDRAM's memory array is divided into two or more banks, which allows one bank to be precharged while the other is accessed. This is a process known as interleaving and eliminates precharge latency, which increases bandwidth. Interleaving is not supported on the eLBC, so it cannot be used when controlling memories through the UPM. Therefore, any access cycles are treated as individual cycles and the autoprecharge (AP) command is used.

Each SDRAM uses a burst counter to increment column addresses on each clock for burst cycles. The burst length and burst type (sequential or interleaved) is selected by programming an on-chip mode register. The user sets the length of the burst sequence to one, two, four, or eight transfers by programming a mode register of the SDRAM. The burst sequence takes advantage of a three-stage pipeline that allows new memory accesses to be initiated before the preceding access is complete. If the pipeline is full, data can be accessed on every clock cycle. After a read burst is complete, the outputs are placed into high impedance mode until a new access cycle is initiated. Because interleaving is not supported, a sequential burst of eight transfers should be programmed for a 32-bit bus width.

The internal SDRAM mode register settings determine when to present data-out information during a read burst. Data can be programmed to appear 1, 2, or 3 clock cycles after a READ command. This feature is called $\overline{\text{CAS}}$ latency and allows the system designer to delay the data's appearance onto the bus until the system is ready. No latencies exist for subsequent cycles in a burst read cycle. While a common $\overline{\text{CAS}}$ latency is 2, it may vary depending on the device and the considered speed for a specific design.

1.1 Hardware Interface

Figure 1 shows the suggested interface from a device using a UPM to an SDRAM. This interface requires glue logic as described in 1.2, "Address Multiplexing and Bank Select". The UPM on the device drives the control, so the $\overline{\text{CS}}$ on the SDRAM is interfaced to one of the $\overline{\text{LCS}n}$ lines on the device, excluding $\overline{\text{LCS0}}$. The SDRAM $\overline{\text{DQM}}$[3:0] signals select the byte lanes and are connected to the appropriate byte strobe signals ($\overline{\text{LBS}}$[0:3]) on the device. $A10_{SD}$ is connected to LGPL0, which has the functionality to drive an address either on the line or to a defined level; as a result, $A10_{SD}$ acts as both an address line and a control line. LGPL2 and LGPL3 generate $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$, and LGPL1 generates $\overline{\text{WE}}$. LGPL5 drives the address mux control line, and the device's LCLK0 signal drives CLK, which is a reference point for the eLBC controller.

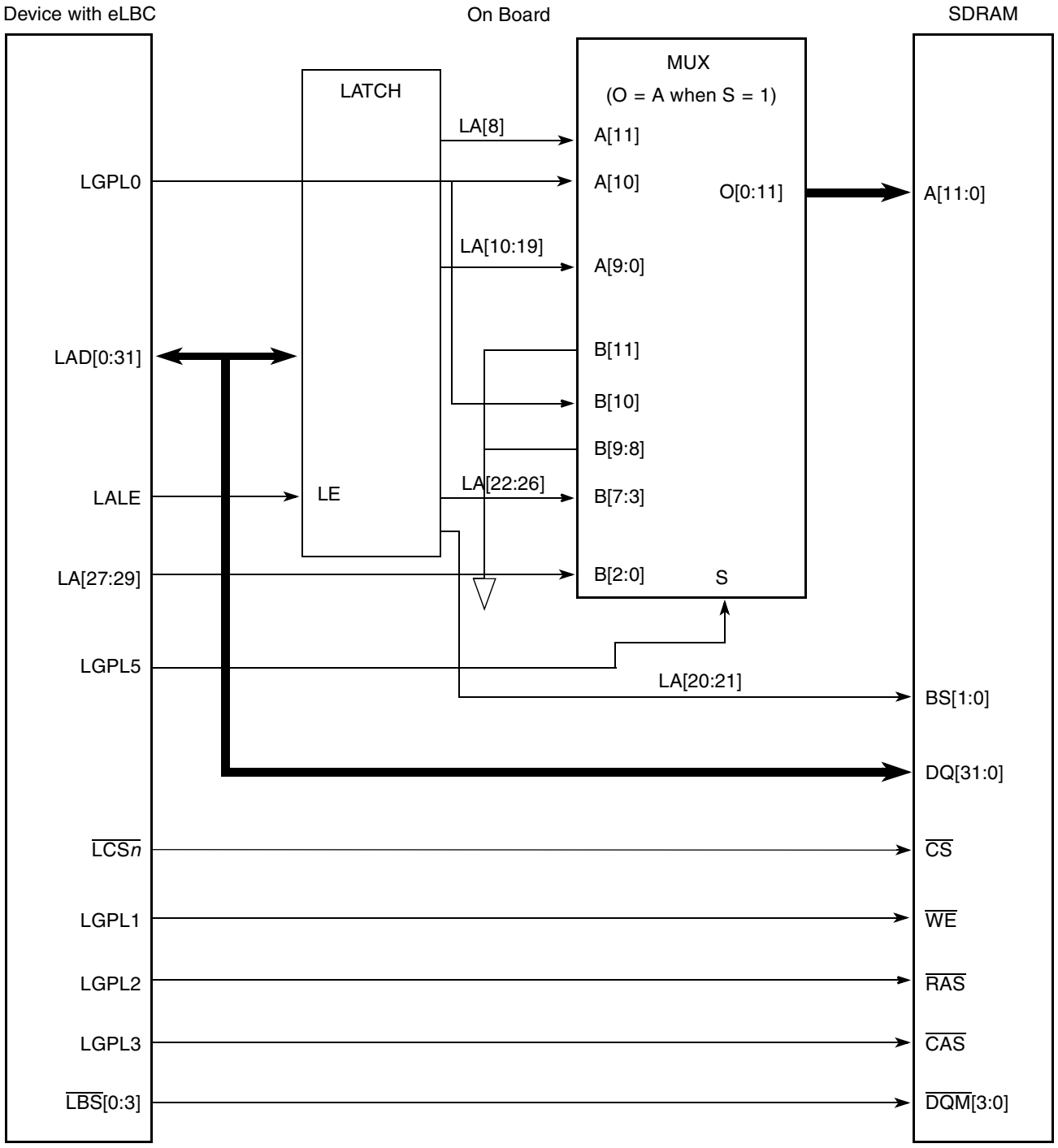

Figure 1. eLBC Device SDRAM Hardware Interface

NOTE

Figure 1 shows that addressing requires glue logic on the board. Section 1.2, "Address Multiplexing and Bank Select," explains the need for the glue logic and how to implement it.

Table 1 shows an example using SDRAM that has 4096 rows and 256 columns, and therefore requires 12 row address lines and eight column address lines. BS[1:0] are connected to $A[20:21]_{MPC}$.

Table 1. SDRAM Addressing Example

Device with eLBC	SDRAM
A[8:19]	12 bits to cover 4096 rows
A[20:21]	BS[1:0]
A[22:29]	8 bits to cover 256 columns

1.2 Address Multiplexing and Bank Select

An SDRAM memory device requires a mixed addressing method. Because of the way the row and column addresses are multiplexed, during an ACTIVATE command the row address is driven on the address bus, and during the READ/WRITE command the column address is driven on the address bus. However, the bank address is not multiplexed and is driven on BS[1:0] (for a four-bank device) regardless of the current command. The UPM on the eLBC cannot support this method of operation and can drive either a linear or a multiplexed address. It cannot use some of the address line in a linear mode (as required for bank address) and some in a multiplexed mode (as required for row/column address). Glue logic on a host board can solve this problem.

In the suggested solution, the UPM is programmed to drive the linear address on LAD[0:31] signals of the device by programming the AMX field to 0b00 in all UPM RAM words. The row and column addresses are multiplexed on the board using a multiplexer. A control signal must be driven to control the mux. Because only the user can know that the glue logic exists, a general purpose line to control the mux should be implemented. LGPL5 can be used to control the address multiplexing on the board because it is unused in the UPM command patterns for the SDRAM memory device. 2 bits of LAD[0:31] are used as bank select lines and are not multiplexed. Refer to Figure 2 for a description of glue logic.

NOTE

The suggested solution is relevant only to an SDRAM memory device with 12 row address bits, 8 column address bits, and 4 banks. Adjustments must be made for devices with a different size or addressing.

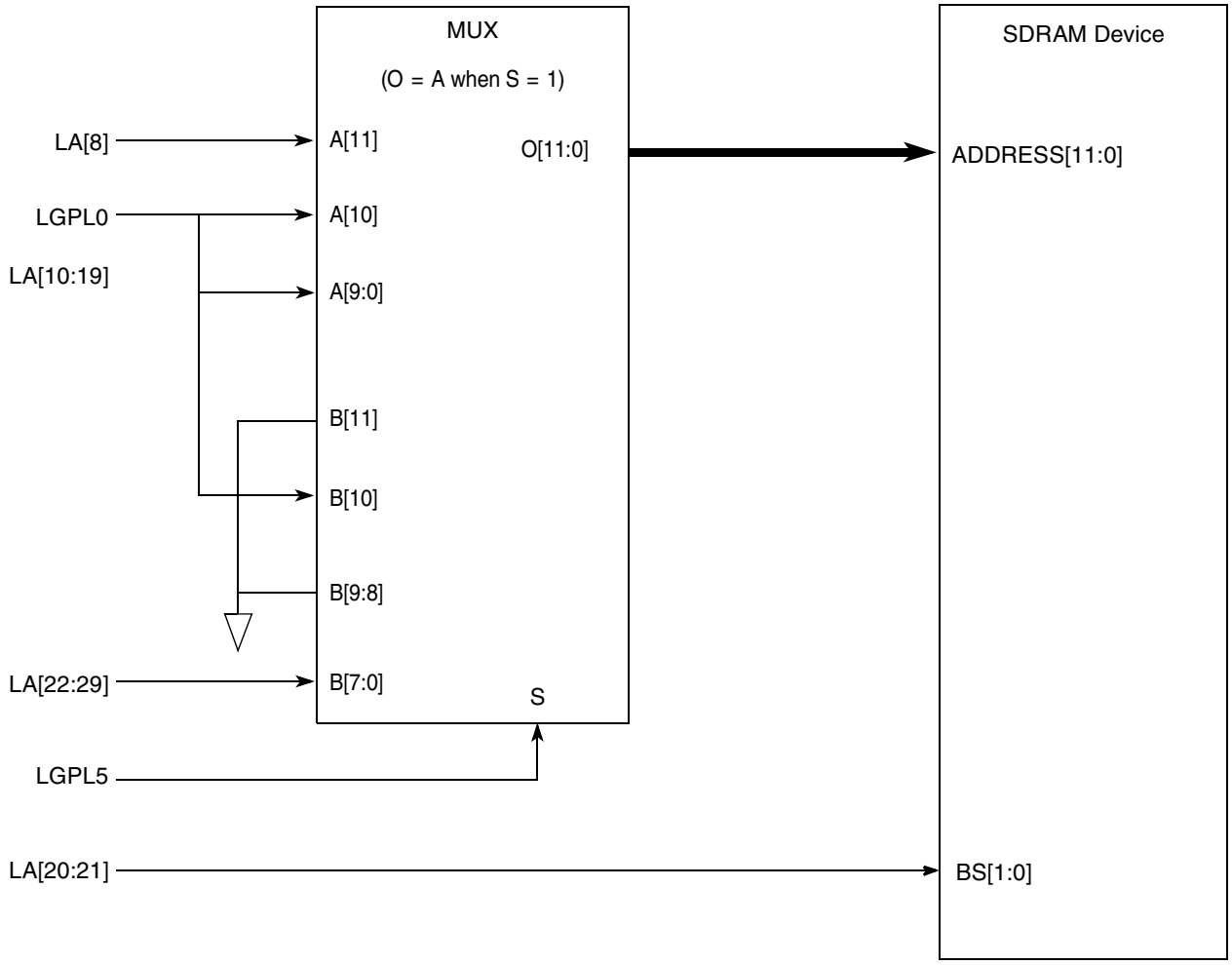

Figure 2. Suggested Glue Logic for SDRAM Addressing

2 SDRAM Usage

This section includes the following subsections:

- Section 2.1, “SDRAM Commands”
- Section 2.2, “SDRAM Access Cycles”

2.1 SDRAM Commands

Table 2 describes the SDRAM commands used with eLBC devices.

Table 2. SDRAM Commands

Command	Description	$A[0:9]_{SD}$, $A11_{SD}$	$A10_{SD}$	$\overline{CS}$	$\overline{RAS}$	$\overline{CAS}$	$\overline{WE}$
ACTIVATE	Bank activate command. Row addresses are latched on $A[0:10]_{SD}$. A further command cannot be issued until t_{RCD} is met.	V [1]	V	L	L	H	H
PRECHARGE ALL	Precharge all. Precharges both banks simultaneously, then switches to the idle state. A further command cannot be issued until t_{RP} is met.	X [2]	H	L	L	H	L
WRITE	Write. Performs a write access to the bank selected by bank select (BS_{SD}). The data is latched on the positive edge of CLK. The burst length and the addressing mode are programmed in the mode register at power-up before the write operation.	V[1]	L	L	H	L	L
WRITEA	Write with autoprecharge (AP). Performs a write command with a precharge operation automatically after the write operation. This command cannot be interrupted by any other command. A further command cannot be issued until t_{RP} is met.	V[1]	H	L	H	L	L
READ	Read. Performs a read access to the bank selected by BS. The data is issued on the positive edge of CLK at a time defined by the $\overline{CAS}$ latency.	V[1]	L	L	H	L	H
READA	Read with autoprecharge (AP). Performs a READ command with a precharge operation automatically after the write operation. This command cannot be interrupted by any other command. A further command cannot be issued until t_{RP} is met.	V[1]	H	L	H	L	H
MRS	Mode register set. Programs $\overline{CAS}$ latency, addressing mode, and burst length in the mode register, which must be configured after power-up because after reset, the mode register is undefined. A further command cannot be issued until t_{RSC} is met.	V[1]	V[1]	L	L	L	L
NOP	No-operation. Performs no operation (the same as device deselect).	X[2]	X[2]	L	H	H	H
AUTOREFRESH	Autorefresh. Refreshes the row address provided by the internal refresh counter. A further command cannot be issued until t_{RC} is met.	X[2]	X[2]	L	L	L	H

1 Valid address

2 No effect

2.2 SDRAM Access Cycles

This section includes the following subsections:

- Section 2.2.1, “Initialization”
- Section 2.2.2, “Mode Register Setting”

2.2.1 Initialization

The first task is to program the power-up sequence. The following explains the JEDEC standard for power-up:

1. Maintain NOP inputs for 200 µs, which can be maintained after power-up by driving an inactive state on the memory controller signals.
2. Issue precharge command (PRCG) to both banks and wait the precharge time, which is $t_{RP.}$
3. Issue eight AUTOREFRESH commands. The refresh command period t_{RC} must be met between refreshes.
4. To initialize the mode register, issue a MODE REGISTER SET (MRS), which programs the burst length, $\overline{\text{CAS}}$ latency, and write mode. After time t_{RSC} the first access can occur. For t_{RSC} = 2 clocks, one trailing wait clock is required.

This complete start-up operation should be programmed into the UPM to configure the SDRAM after start-up. Note that the refresh portion of the initialization can also be executed by simply waiting the required time after initialization for 8 refresh cycles; it is more efficient for a loop in the UPM pattern to generate those 8 refresh cycles.

An MRS command is initiated by driving $\overline{\text{CS}}$, $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and $\overline{\text{WE}}$ low on the rising edge of the clock. The address is then sampled and its value written into the mode register. Note that the next command cannot be given to the memory until t_{RSC} elapses. JEDEC recommends two wait clocks before the next command.

2.2.2 Mode Register Setting

The following settings in the mode register are required:

- Burst length ($A[2:0]_{SD}$). The burst length can be initialized to 2, 4, or 8 words. Because this design is implemented using a 32-bit wide port, the burst length should be set to 8. With a 32-bit wide port, the UPM must do eight accesses in a burst to handle the normal burst size of 32 bytes. Therefore, $A[2:0]_{SD}$ must be 0b011 during the MRS command.
- Address modeselect ($A3_{SD}$). Selects between interleave mode and sequential mode. Sequential mode is selected because the MPC837x has no support for interleaved memories. This requires $A3_{SD}$ = 0 during the MRS command.
- $\overline{\text{CAS}}$ latency ($A[6:4]_{SD}$). The $\overline{\text{CAS}}$ latency is the number of clocks from the READ command to data valid, which depends on two factors: the system clock frequency and the speed of part used. Note that not every SDRAM device supports a $\overline{\text{CAS}}$ latency of 1. You may have to continually use a $\overline{\text{CAS}}$ latency of 2, depending on the SDRAM device.
- Write mode ($A9_{SD}$). Write mode should be configured for burst read and burst write, so $A9_{SD}$ is set to 0 during the MRS command.

All remaining address bits should be set to zero.

2.2.2.1 Mode Register by the UPM Sequence

To interface the Freescale device with an SDRAM memory device using a UPM machine, the user must set the mode register before sending any READ/WRITE command to the SDRAM. The sequence for setting the SDRAM mode register is as follows:

1. Write the value 0x00A0_3C31 to the desired address in the UPM RAM array. The address can be any address between 0x00 and 0x63. However, it is not recommended to write to a RAM address that is dedicated to a specific routine (for example, read burst). For a detailed description of what this value means and how to write into the UPM's RAM array, please refer to the UPM sections within the eLBC chapter in the reference manual.
2. Write MAR with the desired value for the mode register. As explained in the aforementioned reference manual section, this value is 0x23. However, because the two lsb's of the address are not connected to the address lines of a 32-bit port SDRAM memory device, 0x8c must be written into MAR.
3. Program M*x*MR register to run a special routine on the next transaction that hits a UPM assigned bank by setting M*x*MR[OP] = 0b11. On the next access to a UPM assigned bank, the UPM runs a routine starting at RAM Array Address set in M*x*MR[MAD]. To load the mode register, program M*x*MR[MAD] to the same address in which the value 0x00A0_3C31 was written.
4. Initiate a dummy transaction, that is, send a WRITE command on the XLB bus with an address assigned to the UPM, with garbage data. The eLBC knows to ignore the data because M*x*MR[OP] = 0b11, and instead it runs the routine written in the address specified in M*x*MR[MAD]. If steps 1–3 are followed correctly, it should be an MRS command.

3 UPM Programming

This section describes the stages of programming the UPM to interface with an SDRAM memory device, taking into consideration the additional glue logic. It includes the following subsections:

- Section 3.1, "eLBC UPMs"
- Section 3.2, "Recommended UPM Programming for SDRAM memory"
- Section 3.3, "Single Write"

3.1 eLBC UPMs

UPMs are flexible interfaces that connect to a wide range of memory devices. At the heart of each UPM is an internal RAM array that specifies the logical value driven on the external memory control signals (such as $\overline{\text{LCS}}n$, $\overline{\text{LBS}}$[0:3], and LGPL[0:5]) for a given clock cycle. Each word in the RAM array provides bits that allow a memory access to be controlled with a resolution of up to one quarter of the external bus clock period on the byte- and chip-select lines. This subsection elaborates on the programming sequence of the UPMs and the definition of the UPM's RAM words.

For information on the eLBC UPM's other functions and features, refer to the applicable reference manual.

3.1.1 Programming the UPMs

The UPM is a microsequencer that requires microinstructions (or RAM words) to generate signal timings for different memory cycles. Follow these steps to program the UPMs:

1. Set up BR*n* and OR*n* registers.
2. Write patterns into the RAM array.
3. If refresh is required, program MRTPR, LURT, and MAMR[RFEN].
4. Program M*x*MR.

Patterns are written to the RAM array by setting M*x*MR[OP] = 01 and accessing the UPM with any write transaction that hits the relevant chip select. The entire array is thus programmed by an alternating series of writes: to MDR (RAM word to be written), each time followed by a read from MDR and then followed by a dummy write transaction to the relevant UPM assigned bank. A read from MDR is required to ensure that the MDR update has occurred before the write transaction. RAM array contents can also be read for debug purposes, for example, by alternating dummy read transactions, each time followed by reads of MDR (when M*x*MR[OP] = 10).

NOTE

M*x*MR/MDR registers should not be updated while a dummy READ/WRITE access is in progress. If the M*x*MR[MAD] is incremented, the previous dummy transaction is complete.

To properly order updates to the M*x*MR/MDR register and dummy accesses to the UPM memory region, two rules must be followed:

- Because the result of any update to M*x*MR/MDR must be in effect before the dummy READ/WRITE to the UPM region, a write to M*x*MR/MDR should be immediately followed by a read of M*x*MR/MDR.
- The UPM memory region and the memory region containing the M*x*MR should have the same MMU settings; both should be mapped by the MMU as cache-inhibited and guarded. This prevents the core from reordering a read of the UPM memory around the read of M*x*MR. After the UPM array is programmed, the MMU setting for the associated address range can be set to the proper mode for normal operation, such as cacheable and copyback.

3.1.1.1 UPM Programming Example (2 Sequential Writes to the RAM Array)

The following example further illustrates the steps required to perform two writes to the RAM array at nonsequential addresses, assuming that the relevant BR*n* and OR*n* registers have been previously set up:

1. Program M*x*MR for the first write (with the desired RAM Array Address).
2. Write pattern/data to MDR to ensure that the M*x*MR has already been updated with the desired configuration.
3. Read MDR to ensure that it is updated with the desired pattern. If step 2 is not performed, read M*x*MR register.

4. Perform a dummy write transaction.
5. Read/check M*x*MR[MAD]. Repeat step 5 until it is incremented. Once incremented, the previous dummy write transaction is complete; proceed to step 6.
6. Program M*x*MR for the second write with the desired RAM Array Address.
7. Write pattern/data to MDR to ensure that the M*x*MR is properly updated.
8. Read MDR to ensure that it is properly updated.
9. Perform a dummy write transaction.
10. Read/check M*x*MR[MAD]. Once it is incremented, the previous dummy write transaction is complete.

Note that if step 1 or 6 and 2 or 7 are reversed, step 3 or 8 is replaced by the following: Read M*x*MR to ensure that the M*x*MR has already been updated with the desired configuration.

3.1.1.2 UPM Programming Example (2 Sequential Reads from the RAM Array)

RAM array contents can also be read for debug purposes, for example, by alternating dummy read transactions, each time followed by reads of MDR (M*x*MR[OP] = 0b10). The following example further illustrates the steps required to perform two reads from the RAM array at nonsequential addresses assuming that the relevant BR*n* and OR*n* registers have been previously set up:

1. Program M*x*MR for the first read with the desired RAM Array Address.
2. Read M*x*MR to ensure it has already been updated with the desired configuration, such as RAM Array Address.
3. Perform a dummy read transaction.
4. Read/check M*x*MR[MAD]. Repeat step 4 until it is incremented. Once incremented, the previous dummy read transaction is complete; proceed to step 5.
5. Read MDR.
6. Program M*x*MR for the second read with the desired RAM Array Address.
7. Read M*x*MR to ensure that it has been updated with the desired configuration, such as RAM Array Address.
8. Perform a dummy read transaction.
9. Read/check M*x*MR[MAD]. Repeat step 9 until it is incremented. Once incremented, the previous dummy read transaction is complete; proceed to step 10.
10. Read MDR.

3.1.2 RAM Words

The RAM word is a 32-bit microinstruction stored in one of 64 locations in the RAM array. It specifies timing for external signals controlled by the UPM. Figure 3 shows the RAM word fields. When LCRR[CLKDIV] = 4 or 8, CST*n* and BST*n* determine the state of UPM signals $\overline{\text{LCS}}n$ and $\overline{\text{LBS}}$[0:3] at each quarter phase of the bus clock. If LCRR[CLKDIV] = 2, CST2 and CST4 are ignored and the external has the values defined by CST1 and CST3 but extended to half the clock cycle. The same interpretation occurs for the BST*n* bits when LCRR[CLKDIV] = 2.

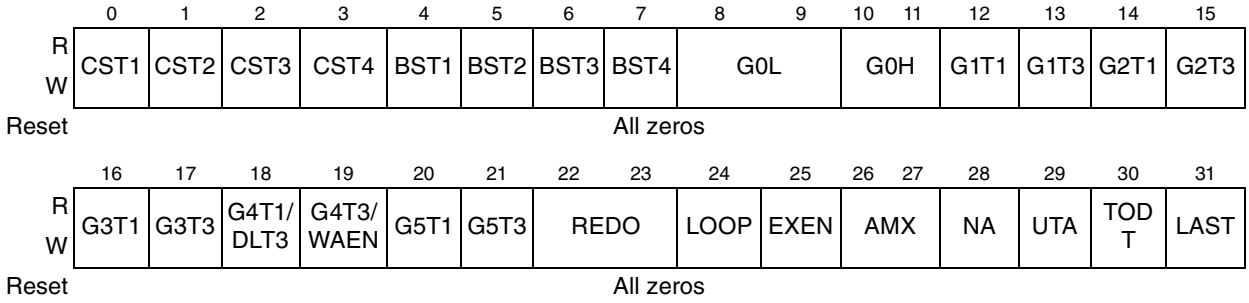

Figure 3. RAM Word Fields

Table 3. RAM Word Field Descriptions

Bits	Name	Description
0	CST1	Chip select timing 1. Defines the state (0 or 1) of $\overline{\text{LCS}}n$ during bus clock quarter phase 1.
1	CST2	Chip select timing 2. Defines the state (0 or 1) of $\overline{\text{LCS}}n$ during bus clock quarter phase 2.
2	CST3	Chip select timing 3. Defines the state (0 or 1) of $\overline{\text{LCS}}n$ during bus clock quarter phase 3.
3	CST4	Chip select timing 4. Defines the state (0 or 1) of $\overline{\text{LCS}}n$ during bus clock quarter phase 4.
4	BST1	Byte select timing1. Defines the state (0 or 1) of $\overline{\text{LBS}}$ during bus clock quarter phase 1.
5	BST2	Byte select timing 2. Defines the state (0 or 1) of $\overline{\text{LBS}}$ during bus clock quarter phase 2.
6	BST3	Byte select timing 3. Defines the state (0 or 1) of $\overline{\text{LBS}}$ during bus clock quarter phase 3.
7	BST4	Byte select timing 4. Defines the state (0 or 1) of $\overline{\text{LBS}}$ during bus clock quarter phase 4.
8–9	G0L	General purpose line 0 lower. Defines the state of LGPL0 during the bus clock quarter phases 1 and 2 (first half phase). 00 Value defined by M*x*MR[G0CL] 01 Reserved 10 0 11 1
10–11	G0H	General purpose line 0 higher. Defines the state of LGPL0 during the bus clock quarter phases 3 and 4 (second half phase). 00 Value defined by M*x*MR[G0CL] 01 Reserved 10 0 11 1
12	G1T1	General purpose line 1 timing 1. Defines the state (0 or 1) of LGPL1 during bus clock quarter phases 1 and 2 (first half phase).

Table 3. RAM Word Field Descriptions (continued)

Bits	Name	Description
13	G1T3	General purpose line 1 timing 3. Defines the state (0 or 1) of LGPL1 during bus clock quarter phases 3 and 4 (second half phase)
14	G2T1	General purpose line 2 timing 1. Defines state (0 or 1) of LGPL2 during bus clock quarter phases 1 and 2 (first half phase).
15	G2T3	General purpose line 2 timing 3. Defines the state (0 or 1) of LGPL2 during bus clock quarter phases 3 and 4 (second half phase).
16	G3T1	General purpose line 3 timing 1. Defines the state (0 or 1) of LGPL3 during bus clock quarter phases 1 and 2 (first half phase).
17	G3T3	General purpose line 3 timing 3. Defines the state (0 or 1) of LGPL3 during bus clock quarter phases 3 and 4 (second half phase).
18	G4T1/DLT3	General purpose line 4 timing 1/delay time 3. The function of this bit is determined by M*x*MR[GPL4]. If M*x*MR[GPL4] = 0 and LGPL4/LUPWAIT pin functions as an output (LGPL4), G4T1/DLT3 defines the state (0 or 1) of LGPL4 during bus clock quarter phases 1 and 2 (first half phase). If M*x*MR[GPL4] = 1 and LGPL4/LUPWAIT functions as an input (LUPWAIT), if a read burst or single read is executed, G4T1/DLT3 defines the sampling of the data bus as follows: 0 In the current word, the data bus should be sampled at the start of bus clock quarter phase 1 of the next bus clock cycle. 1 In the current word, the data bus should be sampled at the start of bus clock quarter phase 3 of the current bus clock cycle.
19	G4T3/WAEN	General purpose line 4 timing 3/wait enable. Bit function is determined by M*x*MR[GPL4]. If M*x*MR[GPL4] = 0 and LGPL4/LUPWAIT pin functions as an output (LGPL4), G4T3/WAEN defines the state (0 or 1) of LGPL4 during bus clock quarter phases 3 and 4 (second half phase). If M*x*MR[GPL4] = 1 and LGPL4/LUPWAIT functions as an input (LUPWAIT), G4T3/WAEN is used to enable the wait mechanism: 0 LUPWAIT detection is disabled. 1 LUPWAIT is enabled. If LUPWAIT is detected as being asserted, a freeze occurs in the external signals' logical values until LUPWAIT is detected as being negated.
20	G5T1	General purpose line 5 timing 1. Defines the state (0 or 1) of LGPL5 during bus clock quarter phases 1 and 2 (first half phase).
21	G5T3	General purpose line 5 timing 3. Defines the state (0 or 1) of LGPL5 during bus clock quarter phases 3 and 4 (second half phase).
22–23	REDO	Redo current RAM word. Defines the number of times to execute the current RAM word. 00 Once (normal operation) 01 Twice 10 Three times 11 Four times
24	LOOP	Loop start/end. The first RAM word in the RAM array where LOOP is 1 is recognized as the loop start word. The next RAM word where LOOP is 1 is the loop end word. RAM words between, and including the start and end words, are defined as part of the loop. The number of times the UPM executes this loop is defined in the corresponding loop fields of the M*x*MR. 0 The current RAM word is not the loop start word or loop end word. 1 The current RAM word is the start or end of a loop.

Table 3. RAM Word Field Descriptions (continued)

Bits	Name	Description
25	EXEN	Exception enable. Allows branching to an exception pattern at the exception start address (EXS). When an internal bus monitor time-out exception is recognized and EXEN in the RAM word is set, the UPM branches to the special exception start address (EXS) and begins operating as the pattern defined there specifies. The user should provide an exception pattern to negate signals controlled by the UPM in a controlled fashion. For DRAM control, a handler should negate $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ to prevent data corruption. If EXEN = 0, exceptions are ignored by UPM (but not by Local Bus) and execution continues. After the UPM branches to the exception start address, it continues reading until the LAST bit is set in the RAM word. 0 The UPM continues executing the remaining RAM words, ignoring any internal bus monitor time-out. 1 The current RAM word allows a branch to the exception pattern after the current cycle if an exception condition is detected.
26–27	AMX	Address multiplexing. Determines the source of LAD[0:31] during a LALE phase. Any change in the AMX field initiates a new LALE (address) phase. 00]LAD[0:31] (and in conjunction with LA[16:25]) is the non-multiplexed address. For example, column address. 01 Reserved 10 LAD[0:31] (and in conjunction with LA[16:25]) is the address multiplexed according to M*x*MR[AM]. For example, row address. 11 LAD[0:31] (and in conjunction with LA[16:25]) is the contents of MAR. Used, for example, to initialize a mode. Note that Source ID debug mode is only supported for the AMX = 00 setting.
28	NA	Next burst address. Determines when the address is incremented during a burst access. 0 The address increment function is disabled. 1 The address is incremented in the next cycle. In conjunction with the BR*n*[PS], the increment value of LA[21:25] is 1, 2, or 4 for port sizes of 8 bits, 16 bits and 32 bits.
29	UTA	UPM transfer acknowledge (TA). Indicates assertion of TA in the current cycle. 0 TA is not asserted in the current cycle. 1 TA is asserted in the current cycle.
30	TODT	Turn-on disable timer. The disable timer associated with each UPM guarantees a minimum time between 2 successive accesses to the same memory bank. This feature is critical when DRAM requires a $\overline{\text{RAS}}$ precharge time. TODT turns the timer on to prevent another UPM access to the same bank until the timer expires. The disable timer period is determined in M*x*MR[DS*n*] and does not affect memory accesses to different banks. Note that TODT must be set together with LAST, otherwise it is ignored. 0 The disable timer is turned off. 1 The disable timer for the current bank is activated preventing a new access to the same bank (when controlled by the UPMs) until the disable timer expires. For example, precharge time.
31	LAST	Last word. When LAST is read in a RAM word, the current UPM pattern terminates and control signal timing set in the RAM word is applied to the current and last cycle. However, if the disable timer is activated and the next access is to the same bank, execution of the next UPM pattern is held off and the control signal values specified in the last word are extended in duration for the number of clock cycles specified in M*x*MR[DS*n*]. 0 The UPM continues executing RAM words. 1 Indicates the last RAM word in the program. The service to the UPM request is done after this cycle concludes.

3.2 Recommended UPM Programming for SDRAM memory

Because an SDRAM interface uses a device with an eLBC controller, glue logic should be considered when programming the UPM. This section provides a recommended UPM RAM array programming for initiating different SDRAM commands.

3.2.1 Single Read

Table 4 describes the recommended UPM RAM array programming for a single READ command. Each row specifies a bit or bits in the RAM word, and the column specifies the RAM Array Address where that word should be programmed. The specified address for the start of the single read routine in the UPM RAM array is 0x00, and the total available RAM words for this routine are 8. The recommended programming requires the use of only 5 RAM words for a single read routine. For a detailed description on each action, refer to Table 5.

Table 4. UPM Programming for Single Read Operation

Address	0x00	0x01	0x02	0x03	0x04
Action	ACTIVATE	NOP	READ with no AP	Precharge	Transfer acknowledge (TA)
RAM Word Bit	Value	Value	Value	Value	Value
CST1	0	1	0	0	1
CST2	0	1	0	0	1
CST3	0	1	0	0	1
CST4	0	1	0	0	1
BST1	1	1	0	0	0
BST2	1	1	0	0	0
BST3	1	1	0	0	0
BST4	1	1	0	0	0
G0L	00	11	10	11	11
G0H	00	11	10	11	11
G1T1	1	1	1	0	1
G1T3	1	1	1	0	1
G2T1	0	1	1	0	1
G2T3	0	1	1	0	1
G3T1	1	1	0	1	1
G3T3	1	1	0	1	1
G4T1/DLT3	1	1	1	1	1
G4T3/WAEN	1	1	1	1	1

Table 4. UPM Programming for Single Read Operation (continued)

Address	0x00	0x01	0x02	0x03	0x04
Action	ACTIVATE	NOP	READ with no AP	Precharge	Transfer acknowledge (TA)
RAM Word Bit	Value	Value	Value	Value	Value
G5T1	0	0	1	0	1
G5T3	0	0	1	0	1
REDO	00	01	00	0	00
LOOP	0	0	0	0	0
EXEN	0	0	0	0	0
AMX	00	00	00	00	00
NA	0	0	0	0	0
UTA	0	0	0	0	1
TODT	0	0	0	0	0
LAST	0	0	0	0	1

Table 5. Description of Access Cycles for Single Read Operation

RAM Array Address	Value	Description
0x00	0x0F0C_F000	Activation
0x01	0xFFFF_F100	NOP. Required to meet t_{RCD}. More NOP cycles can be added by increasing REDO field in the RAM word.
0x02	0x00AF_3C00	READ with no AP. Precharge is initiated by the UPM.
0x03	0x00F0_F000	NOP. Required to meet the $\overline{\text{CAS}}$ latency. More NOP cycles can be added by increasing REDO field in the RAM word.
0x04	0xF0FF_FC05	TA and a LAST bit in the RAM word.

3.3 Single Write

Table 6 describes the recommended UPM RAM array programming for a single WRITE command. Each row specifies a bit or bits in the RAM word, and the column specifies the RAM Array Address where that word should be programmed. The specified address for the start of the single write routine in the UPM RAM array is 0x18. 8 RAM words are available but the recommended programming requires the use of only 4 RAM words for a single write routine. For a detailed description of each action, refer to Table 7.

Table 6. UPM Programming for Single Write Operation

Address	0x18	0x19	0x1A	0x1B
Action	ACTIVATE	NOP	WRITE with no AP + TA	Precharge
RAM Word Bit	Value	Value	Value	Value
CST1	0	1	0	0
CST2	0	1	0	0
CST3	0	1	0	0
CST4	0	1	0	0
BST1	1	1	0	0
BST2	1	1	0	0
BST3	1	1	0	0
BST4	1	1	0	0
G0L	00	11	10	11
G0H	00	11	10	11
G1T1	1	1	0	0
G1T3	1	1	0	0
G2T1	0	1	1	0
G2T3	0	1	1	0
G3T1	1	1	0	1
G3T3	1	1	0	1
G4T1/DLT3	1	1	1	1
G4T3/WAEN	1	1	1	1
G5T1	0	0	1	0
G5T3	0	0	1	0
REDO	00	01	00	0
LOOP	0	0	0	0
EXEN	0	0	0	0
AMX	00	00	00	00
NA	0	0	0	0
UTA	0	0	1	0
TODT	0	0	0	0
LAST	0	0	0	1

Table 7. Description of Access Cycles for Single Write Operation

RAM Array Address	Value	Description
0x18	0x0F0C_F000	Activation
0x19	0xFFFF_F100	NOP. Required to meet t_{RCD}. More NOP cycles can be added by increasing REDO field in the RAM word.
0x1A	0x00A3_3C04	WRITE + TA with no AP, which is initiated by the UPM.
0x1B	0x00F0_F001	AP+ LAST bit in the RAM word.

3.3.1 Burst Read

Table 8 describes the recommended UPM RAM array programming for a burst read command. Each row specifies a bit or bits in the RAM word, and the column specifies the RAM Array Address where that word should be programmed. The specified address for the start of the burst read routine in the UPM RAM array is 0x08. 8 RAM words are available for this routine and the recommended programming requires the use of all 8 RAM words. For a detailed description on each action, refer to Table 9.

Table 8. UPM Programming for Burst Read Operation

Address	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
Action	ACTIVATE	NOP	READA	NOP	TA four times	TA two times	TA	TA + LAST
RAM Word Bit	Value	Value	Value	Value	Value	Value	Value	Value
CST1	0	1	0	0	1	1	1	1
CST2	0	1	0	0	1	1	1	1
CST3	0	1	0	0	1	1	1	1
CST4	0	1	0	0	1	1	1	1
BST1	1	1	0	0	0	0	1	1
BST2	1	1	0	0	0	0	1	1
BST3	1	1	0	0	0	0	1	1
BST4	1	1	0	0	0	0	1	1
G0L	00	11	11	11	11	11	11	11
G0H	00	11	11	11	11	11	11	11
G1T1	1	1	1	0	1	1	1	1
G1T3	1	1	1	0	1	1	1	1
G2T1	0	1	1	0	1	1	1	1
G2T3	0	1	1	0	1	1	1	1
G3T1	1	1	0	1	1	1	1	1
G3T3	1	1	0	1	1	1	1	1

Table 8. UPM Programming for Burst Read Operation (continued)

Address	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
Action	ACTIVATE	NOP	READA	NOP	TA four times	TA two times	TA	TA + LAST
RAM Word Bit	Value	Value	Value	Value	Value	Value	Value	Value
G4T1/DLT3	1	1	1	1	1	1	1	1
G4T3/WAEN	1	1	1	1	1	1	1	1
G5T1	0	0	1	0	1	1	1	1
G5T3	0	0	1	0	1	1	1	1
REDO	00	01	00	00	11	01	00	00
LOOP	0	0	0	0	0	0	0	0
EXEN	0	0	0	0	0	0	0	0
AMX	00	00	00	00	00	00	00	00
NA	0	0	0	0	0	0	0	0
UTA	0	0	0	0	1	1	1	1
TODT	0	0	0	0	0	0	0	0
LAST	0	0	0	1	0	0	0	1

Table 9. Description of Access Cycles for Burst Read Operation

RAM Array Address	Value	Description
0x08	0x0F0C_F000	Activation
0x09	0xFFFF_F100	NOP. Required to meet t_{RCD}. More NOP cycles can be added by increasing REDO field in the RAM word.
0x0A	0x00FF_3C00	READA
0x0B	0xF0FF_F000	NOP. Required to meet the $\overline{CAS}$ latency. More NOP cycles can be added by increasing REDO field in the RAM word
0x0C	0xF0FF_FF04	TA four times. The total number of TA is eight but the max value of REDO filed in the RAM word is four.
0x0D	0xF0FF_FD04	TA two times only because in the 7th TA $\overline{BST}$ is negated so a different RAM word is needed.
0x0E	0xFFFF_FC04	TA. This is the 7th TA so in this RAM word $\overline{BST}$ is negated. The last TA requires a different RAM word because it has the LAST bit.
0x0F	0xFFFF_FC05	TA with LAST bit of the RAM array.

3.3.2 Burst Write

Table 10 describes the recommended UPM RAM array programming for a burst write command. Each row specifies a bit or bits in the RAM word, and the column specifies the RAM Array Address where that word should be programmed. The specified address for the start of the burst write routine in the UPM RAM array is 0x20. The total available RAM words for this routine are 8, but the recommended programming requires the use of only 6 RAM words. For a detailed description of each action, refer to Table 11.

Table 10. UPM Programming for Burst Write Operation

Address	0x20	0x21	0x22	0x23	0x24	0x25
Action	ACTIVATE	NOP	WRITEA + TA	TA four times	TA two times	TA + LAST
RAM Word Bit	Value	Value	Value	Value	Value	Value
CST1	0	1	0	1	1	1
CST2	0	1	0	1	1	1
CST3	0	1	0	1	1	1
CST4	0	1	0	1	1	1
BST1	1	1	0	0	0	0
BST2	1	1	0	0	0	0
BST3	1	1	0	0	0	0
BST4	1	1	0	0	0	0
G0L	00	11	11	11	11	11
G0H	00	11	11	11	11	11
G1T1	1	1	0	1	1	1
G1T3	1	1	0	1	1	1
G2T1	0	1	1	1	1	1
G2T3	0	1	1	1	1	1
G3T1	1	1	0	1	1	1
G3T3	1	1	0	1	1	1
G4T1/DLT3	1	1	1	1	1	1
G4T3/WAEN	1	1	1	1	1	1
G5T1	0	0	1	1	1	1
G5T3	0	0	1	1	1	1
REDO	00	01	00	11	01	00
LOOP	0	0	0	0	0	0
EXEN	0	0	0	0	0	0

Table 10. UPM Programming for Burst Write Operation (continued)

Address	0x20	0x21	0x22	0x23	0x24	0x25
Action	ACTIVATE	NOP	WRITEA + TA	TA four times	TA two times	TA + LAST
RAM Word Bit	Value	Value	Value	Value	Value	Value
AMX	00	00	00	00	00	00
NA	0	0	0	0	0	0
UTA	0	0	1	1	1	1
TODT	0	0	0	0	0	0
LAST	0	0	0	0	0	1

Table 11. Description of Access Cycles for Burst Write Operation

RAM Array Address	Value	Description
0x20	0x0F0C_F000	Activation
0x21	0xFFFF_F100	NOP. Required to meet t_{RCD}. More NOP cycles can be added by increasing REDO field in the RAM ward.
0x22	0x00F3_3C04	WRITE with TA and AP.
0x23	0xF0FF_FF04	TA four times. The total number of TA is eight but the max value of REDO filed in the RAM word is 4.
0x24	0xF0FF_FD04	TA two times only because the 8th TA requires the LAST bit as shown in the next word (ox25).
0x25	0xF0FF_FC05	TA with LAST bit in the RAM word.

3.3.3 Refresh Services

A user may want to use a periodic refresh service to an SDRAM memory device, and the UPMs in the eLBC support this kind of service. The refresh pattern of each UPM is stored in address 0x30 of its RAM array.

Each UPM contains a refresh timer that can be programmed to generate refresh service requests of a particular pattern in the RAM array. Figure 4 shows the clock division hardware associated with memory refresh timer request generation. The UPM refresh timer register (LURT) defines the period for the timers associated with all three UPMs.

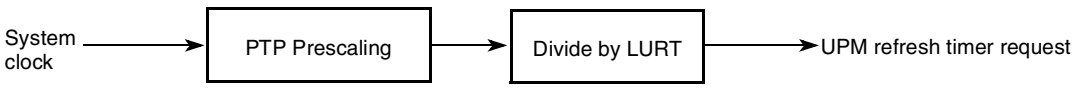

Figure 4. Clock Division for UPM Refresh Timer

By default, all local bus refreshes are performed using the UPMA refresh pattern; if refresh is required, MAMR[RFEN] must be set. Only one refresh routine should be programmed and placed in UPMA, which serves as the refresh executor. If the RFEN bit of the corresponding UPM is set, any banks assigned to a UPM are provided with the refresh pattern. Therefore, UPMA assigned banks always receive refresh services when MAMR[RFEN] is set, and if the corresponding M*x*MR[RFEN] bits are set, UPMB and UPMC assigned banks also receive the same refresh services.

Table 12 shows the recommended RAM array programming for a periodic refresh service. For a detailed description of the action, refer to Table 13.

Table 12. UPM Programming for Refresh Operation, Address = 0x30

RAM Word Bit	Value
CST1	0
CST2	0
CST3	0
CST4	0
BST1	1
BST2	1
BST3	1
BST4	1
G0L	11
G0H	11
G1T1	1
G1T3	1
G2T1	0
G2T3	0
G3T1	0
G3T3	0
G4T1/DLT3	1
G4T3/WAEN	1
G5T1	0
G5T3	0
REDO	00
LOOP	0
EXEN	0
AMX	00
NA	0
UTA	0

Table 12. UPM Programming for Refresh Operation, Address = 0x30 (continued)

RAM Word Bit	Value
TODT	0
LAST	1

NOTE

If performing a refresh command, t_{RC} must be met before executing another command. The user may want to add a few NOP cycles to prevent any chance of a timing violation.

Table 13 shows a detailed description of the action.

Table 13. Description of Access Cycles for Refresh Operation

RAM Array Address	Value	Description
0x30	0x0FFC_3001	Refresh. Note that t_{RC} must be met before any other command is initiated.

4 Revision History

Table 14 provides a revision history for this application note.

Table 14. Document Revision History

Rev. Number	Date	Substantive Change(s)
0	11/08/2007	Initial release.

Appendix A Programming Examples

This section demonstrates a few programming examples that show the programming flow, specifically of directly relevant registers.

NOTE

All examples use UPMB (which is associated with BR1 and OR1) with refresh service enabled and an address window starting at address 0x30000000.

Example A-1. SDRAM Mode Register Setting

This example shows how to set the SDRAM's mode register.

```
Start:
BR1 = 0x3000_18A1;
MBMR[OP] = 0b01;
MBMR[MAD] = 0x36; // any address works
MDR = 0x00A0_3C31;
ADDRESS 0x3000_0000 <= 0x12345678 // dummy transaction
MBMR[OP]=0b11;
MAR[A] = 0x0000_008c;
ADDRESS 0x3000_0000 <= 0x12345678 // dummy transaction
end
```

Example A-2. Single Read UPM Programming

This example demonstrates the programming pseudo code for a single read.

```
Start:
BR1 = 0x3000_18A1;
MBMR[OP] = 0b01;

for i = 0 to 4
      case i
           0: MDR = 0x0F0C_F000;
           1: MDR = 0xFFFF_F100;
           2: MDR = 0x00AF_3C00;
           3: MDR = 0x00F0_F000;
           4: MDR = 0xF0FF_FC05;
      end case
      MBMR[MAD] = i;
      ADDRESS 0x3000_0000 <= 0x12345678 // dummy transaction
```

```
end for
end
```

Example A-3. Burst Write UPM Programming

This example demonstrates the programming pseudo code for a burst write.

```
Start:
BR1 = 0x3000_18A1;
MBMR[OP] = 0b01;

for i = 0 to 5
      case i
           0: MAD = 0x0F0C_F000;
           1: MAD = 0xFFFF_F100;
           2: MAD = 0x00F3_3C04;
           3: MAD = 0xF0FF_FF04;
           4: MAD = 0xF0FF_FD04;
           5: MAD = 0xF0FF_FC05;
      end case
MBMR[MAD] = i + 0x20;
ADDRESS 0x3000_0000 <= 0x12345678 // dummy transaction
end for
end
```

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or
+1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064
Japan
0120 191014 or
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
+1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: AN3533
Rev. 0
11/2007