Designing a Page-Mode DRAM Controller Using MACH Devices

February 2002

Introduction

The three major parts of many digital systems consist of processor, memory and control logic including input/output functions. When implementing these systems, a well-designed memory controller usually determines overall system performance. Each system requires the proprietary memory control specification such as memory map allocation. There are many factors designers must consider when implementing a memory controller, i.e., reliability, fast operation, and ease-of-design migration. Therefore, designing with high-speed programmable logic for memory control has become a popular system level solution.

MACH® devices are the most suitable choice to implement control logic because they provide a lower cost, easy-to-use system solution without sacrificing reliability and performance. This application note describes the general DRAM controller design method using a high-performance, low-cost MACH device. Due to the processor-dependent nature of memory control design, a typical memory-access interface with an MC68340 embedded microprocessor is used to describe the whole design procedure.

This application note contains the fundamental memory controller design theory to give both new designers and skilled system designers the techniques to use programmable devices for the effective design of memory controller. The theory and logic designs described in this document include state machines and timing diagrams of each operation. Source files and test benches are also included in conjunction with this application note.

Design Goals

The goal of this design is to implement a high-performance, low-cost page mode DRAM controller using a single MACH device. While there are many design techniques for memory control such as EDC (Error Detection and Correction), parity generation, bank interleaving and various refresh methods, only the following essential issues are discussed in this document:

- Address decoding and multiplexing
- Bank selection and data sizing
- RAS, CAS timing generation
- Interfacing to processor
- Refresh request generation
- Random access timing
- Page mode access timing

Below is the brief memory controller specification to be implemented in this application note.

- 4Mbytes dual bank 16 bits wide memory interface
  - Bank0: 2Mbytes (000000H - 1FFFFFH)
  - Bank1: 2Mbytes (200000H - 3FFFFFH)
- Interface with MC68340-25MHz embedded processor
- Two 60ns 16Mbits DRAM (1M x 16)
- Support random and page mode read/write access
- Refresh
  - Built-in refresh timing generator
  - CAS-Before-RAS refresh mode

THEORY OF OPERATION

Overview

In order to design a DRAM controller successfully, the timing and functionality of the following four major outputs should be carefully considered:

- RAS (Row Address Strobe)
- CAS (Column Address Strobe)
- Data-ready
- Address multiplexer for DRAM

Using the MC68340-25, this application design generates RAS1B and RAS2B as the row address strobes, LCASB and UCASB as the column address strobes, DSACK1 and DSACK2 as the data-ready signals, and MA[9:0] as the multiplexed address for DRAM. Figure 1 illustrates the typical system-level implementation using this DRAM controller design. Since each I/O pin in a MACH device can drive up to 16mA, no external drivers are needed for the control signals and multiplexed address to DRAM.

The designs of DRAM controller for the Motorola 68000 family of CPUs usually require one to three wait states for an access cycle depending on the memory performance and configuration. The page mode access can eliminate these wait states because the RAS pre-charge time is not required during page-mode operation. As long as the CPU issues the addresses within the same DRAM page, the DRAM controller can accomplish the access cycle with minimum CPU clock cycles. In this design, the page-mode access requires 3 CPU clock cycles, while the page-miss or initial-access needs 5 to 6 clock cycles.

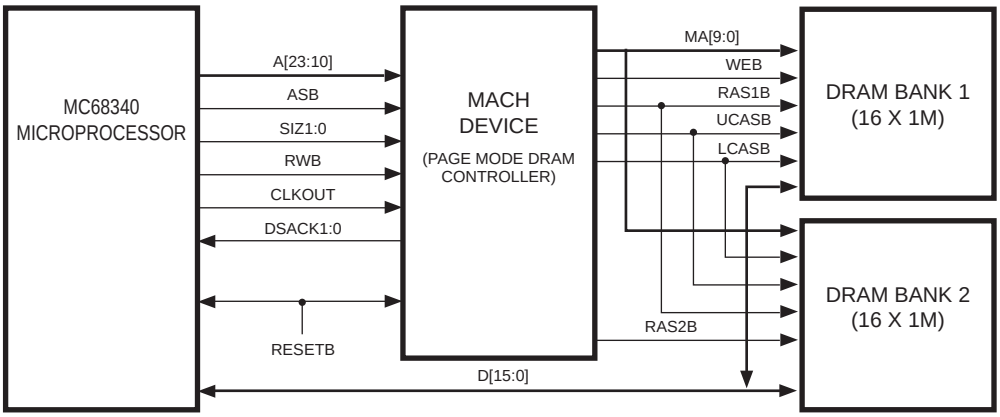

Figure 1. CPU - Memory System Block Diagram

Signal Descriptions

A[23:0]	Address bus	Input	

These signals are inputs from the CPU that define the address of the data to be transferred during a bus cycle.

ASB	Address Strobe	Input	Active low

This signal indicates that a valid address from the CPU is on the address bus.

SIZ1:0	Size	Input	

These signals indicate the number of bytes remaining to be transferred for this cycle.

RWB	Read/ Write	Input	Active low

This signal indicates the direction of data transfer on the bus. It goes to low when a write operation from CPU is initiated.

CLKOUT	System Clock	Input	

System clock out from CPU.

DSACK1:0	Data & Size Acknowledge	Output

During DRAM access cycle, the controller asserts DSACK1:0 to indicate the data ready. These signals also indicate to the CPU the size of the DRAM port (16 bits).

RESETB	System Reset	Input	Active low

This signal resets the internal state machine and other registered functions in this design.

MA[9:0]	Memory Address	Output	

These signals are the multiplexed outputs for the DRAM address lines from the CPU address and provide the row and column addresses for access of the DRAM.

WEB	DRAM Write Enable	Output	Active low

This signal controls the write operation and provides the Early Write cycle timing to the DRAM.

RAS1:0B	Row Address Strobe	Output	Active low

These signals provide the strobe for a row address on the DRAM address lines. The RAS1B selects the memory bank 1, while the RAS2B enables the memory bank 2. In this application note, these signals are represented as RASxB.

UCASB	Column Address Strobe	Output	Active low

This signal provides the strobe for a column address on the DRAM address lines. The UCASB selects the upper byte of the 16-bit data lines on the DRAM.

LCASB	Column Address Strobe	Output	Active low

This signal provides the strobe for a column address on the DRAM address lines. The LCASB selects the lower byte of the 16-bit data lines on the DRAM. In this application note, UCASB and LCASB signals are represented as xCASB.

Transfer Modes

This design is implemented to access two separate memory banks, and RAS1B is connected to the RAS signal of the first bank of 16Mbit DRAM, while RAS2B is connected to the RAS signal for the second bank. Each DRAM bank has two 8-bit wide data ports, with the UCASB signal from the MACH device controlling the upper port, and the LCASB controlling the lower port. The memory control interface for the MC68340 supports the following transfer modes:

- Even Byte-to-Word transfer
  - Valid data on the upper byte (D[15:8])
- Odd Byte-to-Word transfer
  - Valid data on the lower byte (D[7:0])
- Word-to-Word aligned transfer
  - Valid data on the both bytes (D[15:0])
- Long Word-to-Word aligned transfer
  - Valid data on the both bytes (D[15:0])

When Even Byte-to-Word transfer occurs (A0=0), only the UCASB signal can be active to access the data on the upper 8-bit data bus and the lower byte on the data bus will be ignored. In case of Odd Byte-to-Word transfer mode (A0=1), the lower 8 bit data lines are used for the transfer by the LCASB signal and the upper byte on the data bus is discarded. Except for the A1 signal, when the MC68340 performs Long Word-to-Word transfer, it holds the same address as the current one and adjusts the data size from 32 bits to 16 bits on SIZ1 and SIZ0 on the next access, so that the 32-bit wide access can be completed. Table 1 shows the transfer modes supported in this design and the relationship among the signals.

Transfer Mode	SIZ1	SIZ2	A0	DSACK1	DSACK0	D[15:8]	D[7:0]
Even Byte-to-Word	0	1	0	0	X	Valid	Invalid
Odd Byte-to-Word	0		1	0	X	Invalid	Valid
Word-to-Word	1	0	0	0	X	Valid	Valid
Long Word-to-Word	0	0	0	0	X	Valid	Valid

Table 1. Supported Transfer Modes

Functional Description

For this memory controller design, two 60ns fast-page-mode 1Mbit x 16 DRAMs are used in order to obtain good performance throughput while maintaining low cost. The page size for this DRAM is 1024 bits (1Kbyte) and the bus size interfacing with the CPU is 16-bits wide without any interleaved accesses. Therefore, this memory controller has 2K bytes of address range as one page in which the controller keeps on performing the fast accesses for DRAMs in the page-mode. When the addresses issued by the CPU are in the same page range, the first 16-bit access is done by random mode. The memory state machine then immediately begins the next accesses in page-mode until the CPU issues the address indicating out-of-the-page range. Hence, the page-mode DRAMs appear to the CPU as if they are cache memory. The MC68340 CPU has 32-bit address lines and the upper 8 lines (A31-A24) can be configured as a parallel I/O port, or interrupt-acknowledge lines. This design assumes that the address lines A31:24 of the CPU were configured as either Port A[7:0] or IACK[7:0] to eliminate the number of input signals for the decoder block.

The following is the description of each access mode:

Random Access

This is a typical access between MC68340 CPU and memory. Although this design doesn't include the simple random-access-only-mode, it shows conventional access timing. It is important to understand the signal interface and timing of this access mode. The CPU always initiates the memory accesses. Figure 2 gives the random-access timing diagram of the random access mode.

1. The MC68340 drives a valid address onto the bus and a read/write signal.
2. Address strobe signal (ASB) is then asserted by MC68340.
3. The decoder in the DRAM controller decodes the address lines and drives DRAMSEL signal high, if the current address is in DRAM area.
4. The state machine goes to RW1 and asserts row-address strobe signal (RASxB) and DSACK active. Then the CPU has one wait state and reads DSACK1 as low at the end of RW1.
5. In RW2 state, the column address strobe (xCASB) is asserted.
6. At the falling edge of S4 (the end of RW2), the CPU reads the data from DRAM. In case of a write operation, data from the CPU is read to DRAM when xCASB is asserted.

For the second or following random accesses, two wait states are included in each access cycle.

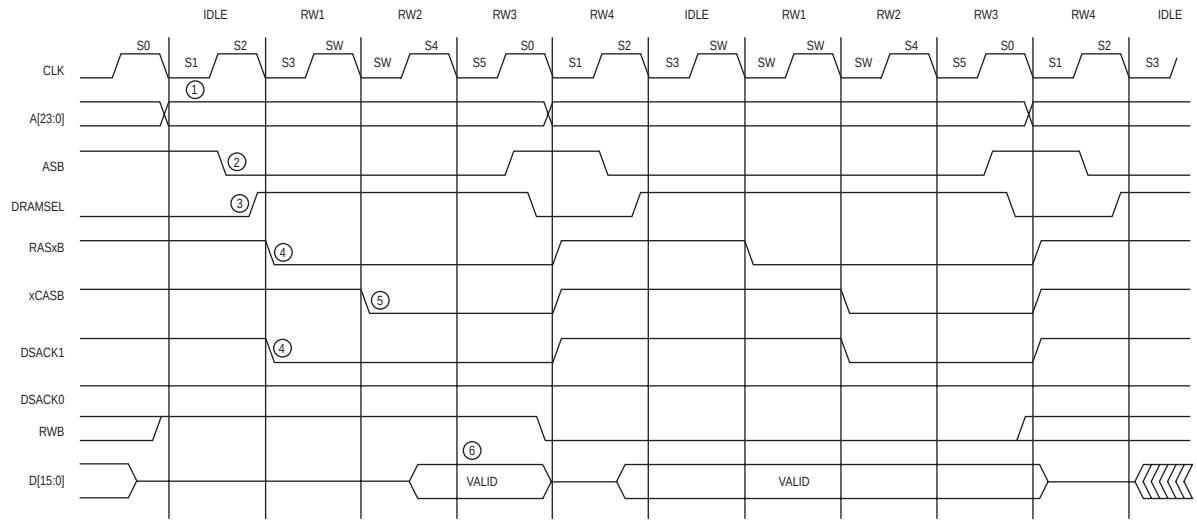

Figure 2. Random Access Mode Timing Diagram

Page Mode Access

The page-mode access cycle can be started when the current address driven by the CPU is the same as the previous one. Figure 3 is the timing diagram for the case of page-hit.

The initial access timing is the same as the random access except that the RASxB signal is still active during RW4 state in order to stay in a page.

7. During the initial access, the address lines are stored by the ASB signal.
8. When the address comes out from the CPU, the current address lines are compared with the stored value.
9. If the current address indicates the same page as the last address, then page-hit signal (PAGE_HIT) is asserted.
10. The DTACK signal is generated asynchronously during PAGE_HIT to avoid the insertion of a wait state in the page mode.
11. The state machine goes to the PG1 state and asserts the xCASB. During write operation, the data from the CPU is captured when the xCASB is falling.
12. In case of the read operation, the data from the DRAM is transferred to the CPU at the falling edge of S4. At the end of PG2 state, the xCASB and DTACK signals are inactive and the xCASB has a pre-charge time during PG3.

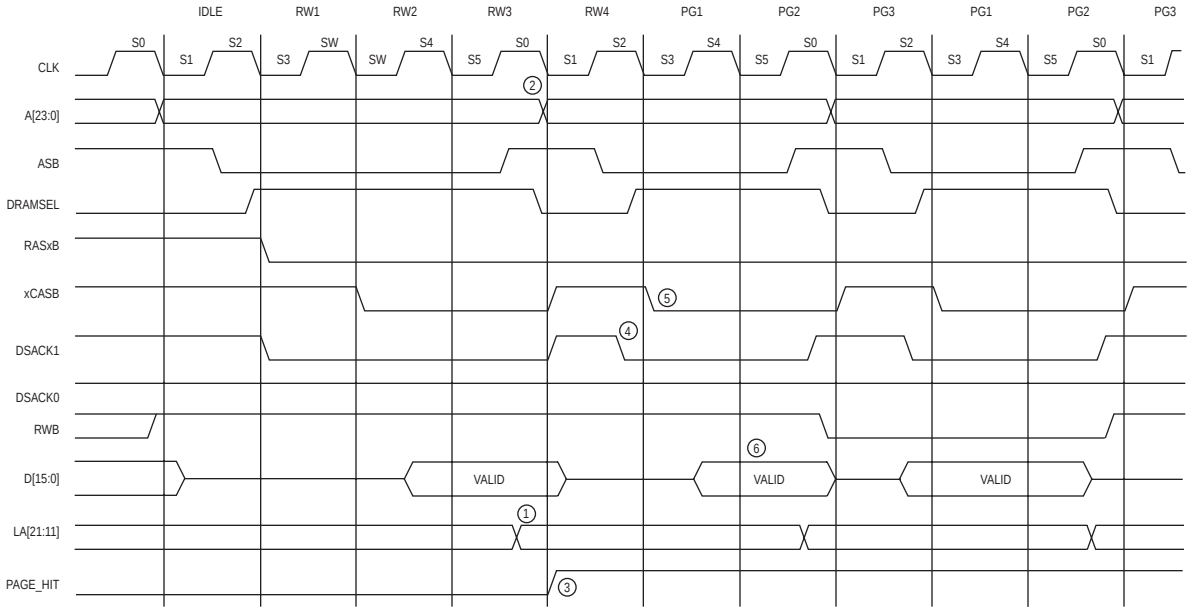

Figure 3. Page Mode Access Timing Diagram (Page Hit)

Once the sequential accesses are completed within a page, the next access will be off the page and the state machine will go into the initial access cycle. Figure 4 illustrates the timing diagram for the case of page-miss.

13. The last address lines are stored by the ASB signal.

14. When the address comes out of the CPU, the current address lines are compared with the stored value.

15. If the current address is out-of the page range, then the page-hit signal (PAGE_HIT) is deasserted.

16. The state machine goes to a RAS pre-charge state (PRECHG) after the detection of low level on the PAGE_HIT, then goes to the IDLE state.

17. The state machine starts an initial access again, if the current cycle is a DRAM read/write access. If not, the state machine will wait until DRAMSEL = 1.

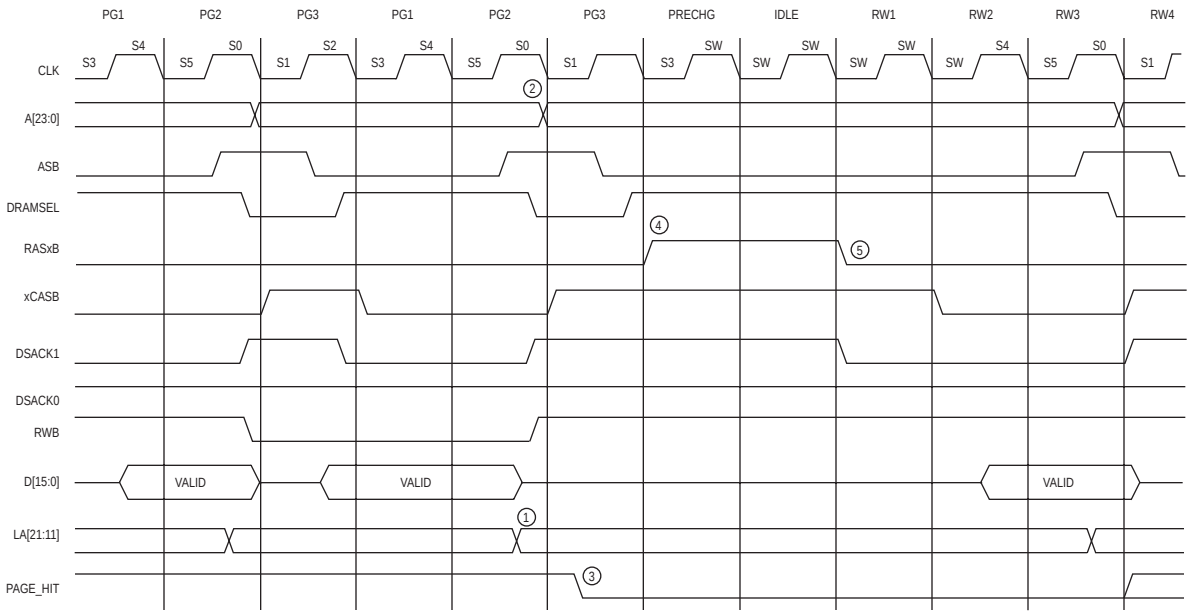

Figure 4. Page Mode Access Timing Diagram (Page Miss)

Refresh Mode

This DRAM controller is designed to support the CAS-Before-RAS (CBR) refresh mode. Figure 5 shows the timing of the CBR.

18. When the refresh counter reaches the terminal-count, a refresh request signal (REFREQ) is asserted.
19. At the last state of each access cycle, the state machine samples the level on the REFREQ signal.
20. When the REFREQ is detected, the state machine goes to the first state of CBR mode and generates a refresh acknowledge signal (REFACK).
21. The REFACK clears the REFREQ signal and the internal refresh counter starts the count over again.
22. In the CBR2 state, both of the xCASB signals are asserted prior to RASxB.
23. In the CBR3 state, both of the RASxB signals are asserted while xCASB signals are still in active state.
24. The RASxB goes to an inactive state and the state machine completes the refresh cycle. Then a RAS pre-charge state is inserted before an access cycle starts.

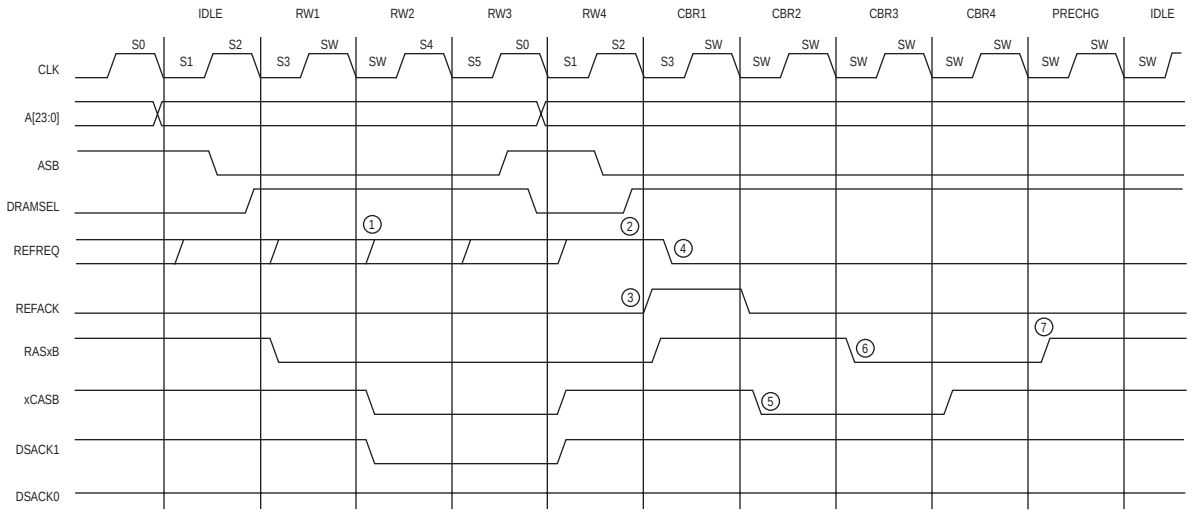

Figure 5. CAS-Before-RAS (CBR) Refresh Timing

IMPLEMENTATION OF THE DRAM CONTROLLER

The page-mode DRAM controller is designed using Vantis' DesignDirect™-CPLD software tool or Synario tool environment. The designs of all functional blocks are implemented by ABEL language, and a top schematic interconnects each block. Figure 6 shows the diagram of the internal blocks in the DRAM controller.

The page mode DRAM controller consists of the following four functional blocks:

- Decoder block
- Page detector
- Page mode timing generator
- DRAM address multiplexer

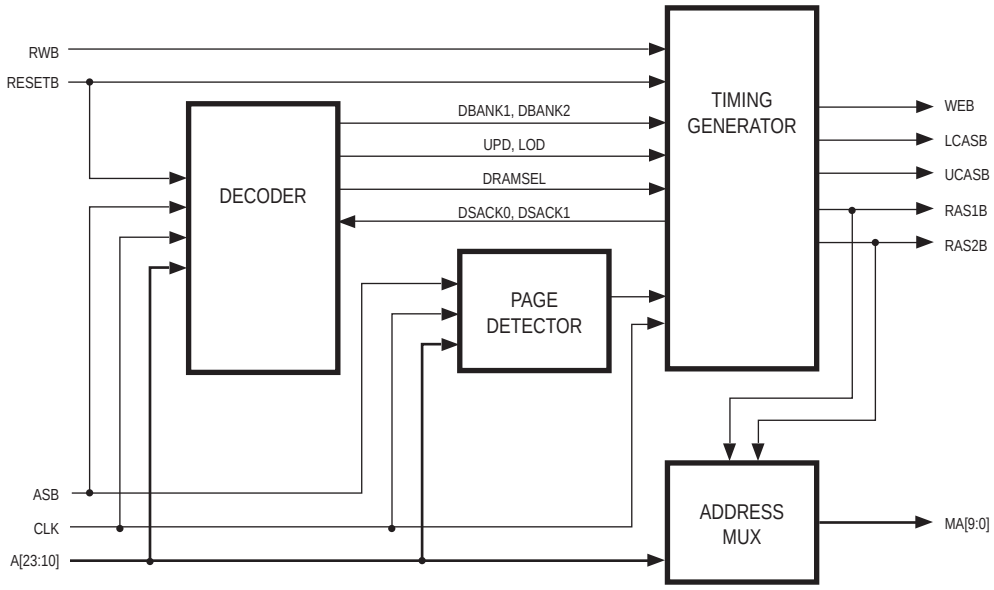

Figure 6. DRAM Controller Block Diagram

Decoder Block

The decoder block decodes the following output signals:

- DRAMSEL DRAM Memory Area (0000000h ~ 03FFFFFh)
- BANK1 DRAM Bank1 Select (0000000h ~ 01FFFFFh)
- BANK2 DRAM Bank2 Select (0200000h ~ 03FFFFFh)
- UPD Upper Byte Access Enable
- LOD Lower Byte Access Enable

Since the DRAMSEL signal initiates the state machine to perform DRAM memory access, the ASB signal should be included in the decoding equation in order to make sure that the CPU starts a memory access cycle. Since the upper 8 bit address lines (A31-A22) are assumed to be configured as a parallel port or interrupt acknowledge signals, the controller decodes only A23 through A0. If the system requires the upper address area, the controller should include A31-A22 lines in the decoding equation.

The BANKx signals are used for the decoding of RASxB signals. When the DRAM controller supports page-mode access, either the RAS1B or RAS2B signal needs to be active until the sequential page-mode accesses are completed. Since the BANKx signals can be asserted or deasserted by the DRAMSEL condition and there is one DRAMSEL inactive clock cycle between page-mode accesses, the BANKx signals should be decoded with extended time to cover the window for the whole RASxB active period. Figure 7 illustrates this operation.

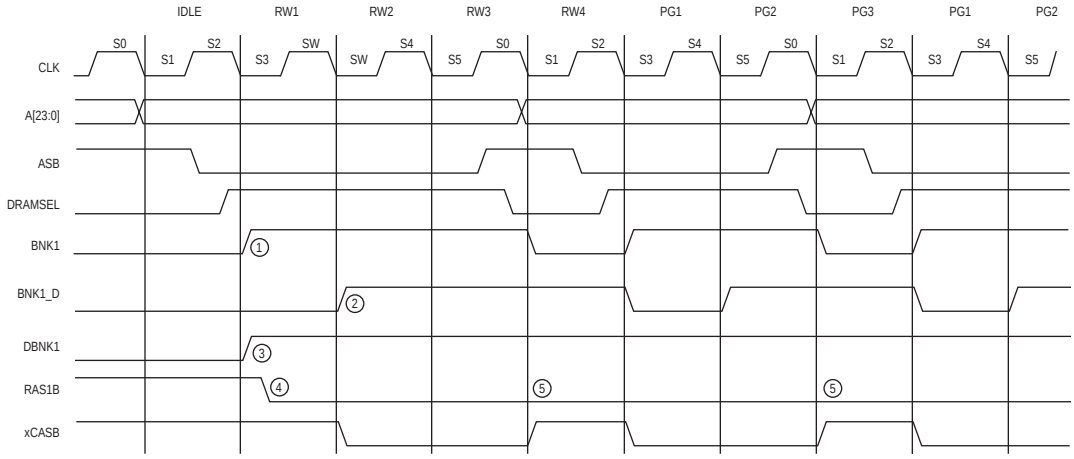

Figure 7. DRAM Bank Select Timing Diagram

25. When the CPU drives a DRAM bank1 area address and ASB, the BNK1 signal is generated at the next falling edge of the CPU clock.

26. The BNK1_D signal is the one-clock delayed signal of BNK1.

27. The final bank select signal DBNK1 is generated during the start of BNK1 active period to the end of BNK1_D active period. This results in the DBNK1 being active until the end of page-mode accesses.

28. The RAS1B signal is selected while the DBNK1 is active.

29. Even for the period that the DRAMSEL is not active, the RAS1B signal can be in an active state.

As Table 1 shows, this DRAM controller has a 16-bit wide data bus and responds to the CPU as only word-wide transfer. In other words, even though the CPU initiates a Byte, Word, or Long-Word access, the DRAM controller always tells the CPU that the bus size to be transferred is 16 bits. In case of the Byte-to-Word transfer, therefore, when the CPU accesses an odd byte address (SIZ1 = 0, SIZ0=1, A0 = 1), the DRAM controller drives DSACKx signals to inform the CPU of the 16-bit bus size and enables only the LCASB to access the lower byte for a DRAM. On the other hand, the Even Byte-to-Word access enables only the UCASB. For Word or Long-Word transfer, both the LCASB and UCASB are enabled.

DRAM Address Multiplexer Block

The DRAM address multiplexer block generates the following output signals.

• MA[9:0] Multiplexed DRAM Row and Column Address Lines

This block generates the address input lines for both DRAMs. According to the DRAM timing specification, the 10-bit row address lines are latched at the falling edge of RASxB signal. Similarly, the falling edge of the xCASB signal latches the 10-bit column address lines. Therefore, the 20-bit address lines from the CPU should be properly multiplexed in order to meet the timing requirements - address setup and hold time. The address multiplexer changes the MA output address from the DRAM row address to the column address after the RASxB is active (low). To meet the timing requirement, the multiplexer selector signal (DD_RASB) is generated from the RASxB signals passing through two internal feedback paths inside the MACH device. The internal node, DD_RASB, is asserted 2 x tPD after the RASxB is active and the final output MA[9:0] will have transition after one more tPD. Thus, it can meet the RAS address hold time requirement. See the ABEL source in Appendix B for the detail implementation.

Page Detector Block

The page detector generates the following output signal.

u PAGE_HIT Page -Hit Detection Output

The page detection block performs a simple operation, which compares the previous DRAM row address with the current row address. Since A[21:11] lines are assigned to the DRAM row address, these address lines are latched as LA[21:11] at the low to high transition of the ASB signal. As soon as the next address lines come out, the comparison with the previous address is performed and the PAGE_HIT will be high if both of the addresses are the same, i.e. representing the accesses within the same page. Figure 3 gives the timing diagram for PAGE_HIT generation.

Page Mode Timing Generator Block

The following output signals are generated from the timing generator.

• RAS1B Row Address Strobe for the Bank1 (to DRAM)
• RAS2B Row Address Strobe for the Bank1 (to DRAM)
• UCASB Column Address Strobe for the Upper Byte (to DRAM)
• LCASB Column Address Strobe for the Upper Byte (to DRAM)
• WEB DRAM Write Enable (to DRAM)
• DSACK0 Data Size Acknowledge and Data Ready 0 (to CPU)
• DSACK1 Data Size Acknowledge and Data Ready 1 (to CPU)

The timing generator block consists of three parts: a state machine, a DSACK timing generator and a refresh counter.

Figure 8 shows the state diagram for the DRAM memory control state machine.

The state machine arbitrates between memory accesses and refresh cycles. There are 13 states to perform this function.

• IDLE Idle State
• RW1 Initial or Random Access State 1

- RW2 Initial or Random Access State 2
- RW3 Initial or Random Access State 3
- RW4 Initial or Random Access State 4
- PAGE1 Page Access State 1
- PAGE2 Page Access State 2
- PAGE3 Page Access State 3
- CBR1 CAS-Before-RAS Refresh State 1
- CBR2 CAS-Before-RAS Refresh State 2
- CBR3 CAS-Before-RAS Refresh State 3
- CBR4 CAS-Before-RAS Refresh State 4
- PRECHG RAS Pre-Charge State

Description of the Timing Generator State Machine

The state machine uses four state variables (S3..S0) with which a total of 16 states can be handled. The remaining 3 states are redundant and, therefore, are not used in this state machine.

The DRAM memory access always starts in the IDLE state. If the CPU drives the address for the DRAM area, then the DRAMSEL is active and the state machine starts an initial memory access. In this case, the state machine goes to RW1 state. When the refresh counter issues the refresh request signal, REFREQ, the state machine immediately goes from the IDLE to CBR1 state. In the RW1 state, one of the RASxB signals is asserted depending on the decoded address range. The DSACK0 is asserted in the RW1 state to inform the CPU that the data is ready to transfer and the transfer is 16 bit wide. The one or both xCASB signals are asserted in the RW2 state depending on the CPU transfer size requirement. If the SIZ1=0 and SIZ0=1, then either the LCASB or UCASB is asserted. If the CPU requests the Word transfer, both the LCASB and UCASB are asserted. The xCASB and DSACK0 signals are deasserted in the RW4 state, while the xRASB signal is still active in order to support the page mode access.

If the page-hit condition (PAGE_HIT & DRAMSEL) is met in the RW4 state, the state machine goes to the page-mode access cycle. In the page-mode access, only the xCASB signals are asserted and deasserted while the page-miss condition occurs. If the page-miss occurs, then the state machine goes to the PRECHG state to give the DRAM the required pre-charge time for the next access. Then it goes to the IDLE state to start another initial random access cycle.

At the end of each access cycle, the state machine checks if there is a refresh request. If the REFREQ exists at those states, the state machine goes to the CBR1 state immediately. During the CAS-Before-RAS refresh cycle, the DRAMSEL holds the DRAM access request of the CPU until the CBR cycle is done no matter what the current access mode is - page or random.

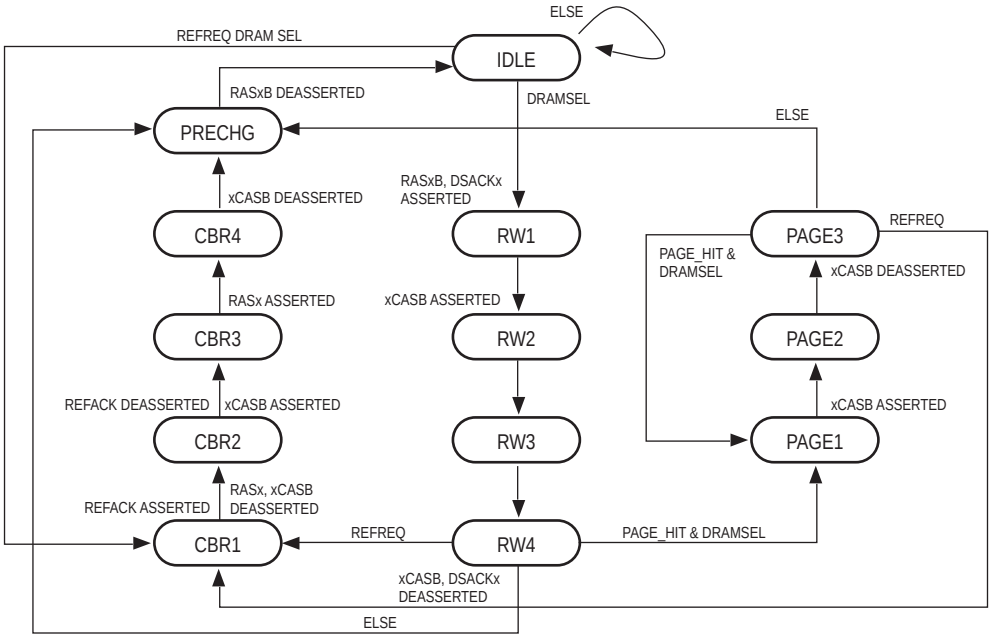

Figure 8. State Diagram

DSACK Timing Generator

In this design, the xCASB signals are generated if the DSACK signals are in the active state. According to the MC68340 timing requirement, however, the DSACK signals should be available at least 5ns prior to the falling edge of S2. In the initial random access cycle (the first access in Figure 3), the xCASB signals can be available one clock after the DSACK assertion. In the page mode access cycle, however, there is no way to have the DSACK assertion synchronously before the xCASB signals assertion at the falling edge of S2 (the second access of Figure 3). Therefore, the asynchronous DSACK generation is adopted for the page mode access. Figure 3 shows that the DSACK1 is generated synchronously in the initial access, while it is generated asynchronously and has the same timing as the DRAMSEL signal for the page mode access to meet the timing requirement.

Refresh Counter

Recent DRAM devices accept slow refresh rates such as a 32ms period for a 1024- row refresh. In this design, however, the standard 16ms-refresh rate is used for more reliable operation. This invokes the refresh operation at two times the specification value, but the actual performance impact is negligible.

The refresh counter is an 8-bit synchronous counter that generates a terminal-count signal when the count reaches 186-hexadecimal. This means that the terminal count occurs every 15.6(s using a 40ns (25MHz) system clock cycle. The terminal count acts as a clock for the REFREQ signal. Once the state machine recognizes the REFREQ, it goes to the CBR1 state and asserts REFACK signal. The counter is cleared by the REFACK, and it counts again to generate the next refresh request.

IMPLEMENTATION USING MACH DEVICES

This design requires about 40 macrocells and less than 150 product-terms to implement into a MACH device. Since this design uses 39 I/O pins, the M4-96/48-10YC (100 PQFP, 96 Macrocells, 48 I/Os) device can be selected as the best device for implementation. When the design was implemented in this device, the static timing analyzer reports the following maximum operating frequency:

- Device:Lattice M4-96/48-10YC (template: M4-96/48) Package QFP
- Device maximum operating frequency : 41.667 MHz
- Device operating period (1/fmax) : 24.000 ns

Based on this report, a 12 or 15ns MACH device can be used for this design because the system clock is in the 25 MHz range. As there are some critical timing requirements among the CPU, the DRAM controller and the DRAM due to the fast page-mode access, the implementation with a 10ns MACH device is a recommended solution.

CONCLUSION

In conclusion, this application note describes a page mode DRAM controller design using an M4-96/48-10ns device. Although this design was not built and tested on the physical board, the functional and timing verifications were performed for validation purposes. All the functional block designs were created in ABEL language and tested using Lattice tools. Appendix A shows the top schematic for the whole DRAM controller system and Appendix B includes the entire ABEL source files.

The most updated source codes and testbenches may be downloaded from the Lattice web site at www.lattices-emi.com.

Appendix A. Page-Mode DRAM Controller Schematic Diagram