UNIVERSITY OF TEXAS AT ARLINGTON

Interfacing SDRAM with Intel® 80386DX™

Microprocessor Systems Design (Memory Interfacing)

Project Implementation and design report on designing an SDRAM controller for supporting MT48LC8M8A2 memory device and interfacing memory with an Intel® 80386DX™ Microprocessor.

NABEEL NAYYAR • 9th August 2023 • UTA College of Engineering

Table of Contents

[This page is intentionally left blank]

Abstract

The goal of this term project is to design an SDRAM controller that enables the interfacing of SDRAM memory with a microprocessor lacking synchronous memory support. The project focuses on creating a functional design without the need for hardware implementation containing schematics and the theory of operation.

The designed SDRAM controller interfaces with one SDRAM memory device, with support for the Micron Synchronous DRAM module MT48LC8M8A2 2 Meg x 8 x 4 Banks memory device and Intel 80386DX (386DX™) microprocessor with 32-bit asynchronous memory data interface.

The project report contains a detailed design for the Microprocessor, SDRAM controller, and memory device following reference designs that show the internal working of the SDRAM controller in Finite state machine diagrams and System Verilog (where needed) necessitating a controller solution. The SDRAM controller encompasses various critical functionalities, represented in state machine designs, and signal generation for row, column, and bank addressing, data masking, data flow management, READY logic, and refresh support to maintain memory integrity.

To meet the project requirements, the report contains a detailed theory of operation in Initializing, Reading, Writing, and Refreshing the memory device and timing analysis of these top-level states.

NOTES:

Find the second Attachment along with this document for the Verilog implementation of the SDRAM Controller.

Design Assumptions/Specifications

1. **SDRAM Device:** The design assumes the use of Synchronous Dynamic Random Access Memory (SDRAM) as memory technology. Specifically, the MT48LC8M8A2 SDRAM memory device is considered as the minimum requirement for the controller interface. The SDRAM parameters, such as CAS latency, burst length, and refresh rate, will be configured according to the datasheet of the selected MT48LC8M8A2 device of speed grade -75.
2. **Processor:** The SDRAM controller is designed to interface with a microprocessor lacking native SDRAM support. The target processor for this project is the 80386DX or a similar 32-bit asynchronous memory data interface processor. The controller's operation and timing are tailored to meet the requirements of this specific microprocessor.
3. **Control Signal Interface:** The controller will interface with the microprocessor using standard control signals such as ~ADS (Address Strobe), W/~R (Write/Read), and M/~IO (Memory/Input-Output). These signals will be assumed to be provided by the microprocessor for memory access operations.
4. **Hardware Implementation:** The scope of this project focuses solely on the design and theoretical implementation of the SDRAM controller. It does not include the hardware implementation of the controller, nor does it involve using any specific northbridge/MCH/IMCH chipsets or existing SDRAM controller reference designs.
5. **Burst Length:** The controller will be designed to support burst length 4 transfers.
6. **Refresh Support:** The SDRAM controller will incorporate an internal refresh counter to generate AUTO REFRESH commands at a rate sufficient to maintain memory integrity, as per the SDRAM's requirements.
7. **Ready Logic and Microprocessor Waiting:** The controller will be designed to generate the READY signal for the microprocessor, indicating when data is available to read or when the controller is ready to accept new data write. The design will aim to minimize microprocessor waiting due to refresh cycles, unless a bus locking protocol is used.
8. **Race/Pipeline Condition:** It is assumed that the microprocessor will never issue a write command when there is a read command in-progress and vice versa.
9. **Device Precharge:** The memory device will Precharge after every READ or WRITE command.

It is important to note that these design assumptions are based on the project requirements and constraints. The actual implementation may vary based on the designer's choices and the specific microprocessor and SDRAM memory devices used. Any additional design decisions made during the development process will be thoroughly documented and justified in the final project report.

Top-Level Hardware Design

The Microprocessor, SDRAM Controller, and the memory device interface with another as shown in Figure 1.1. The SDRAM controls block interfaces with the memory device and the Microprocessor.

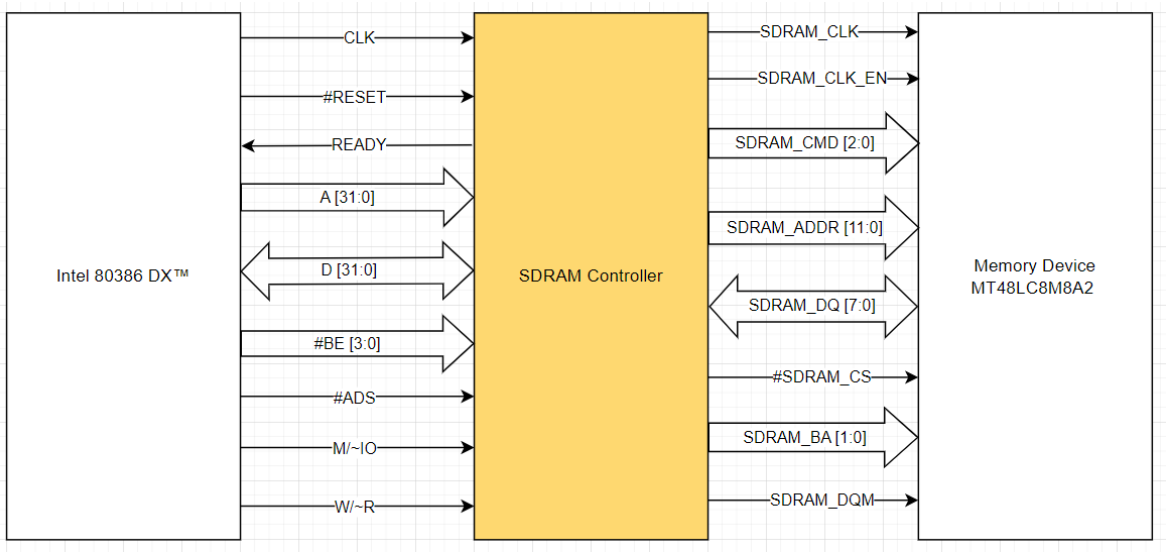


Figure 1.1 General Block diagram of the Top-Level SDRAM Unit

The Top-level design consists of the Microprocessor, the SDRAM Controller (which is detailed in the next section). We choose a Memory device with specifications shown in Table 1.1 with the Microprocessor clock configurations realized in Table 1.2.

Configuration	8 Meg × 8 (2 Meg × 8 × 4 banks)		
Schematic Marking	8M8		
Refresh Count	4K		
Bank Addressing	4 (BA0, BA1)		
Column Addressing	512 (A0-A8)		
Refresh Rate	64 ms (4096-cycle refresh) (15.6 µs/row) Commercial Grade		
Speed Grade	-75 (100 MHz)		
Access Time	CL = 2 (6 ns)	Setup time 1.5 ns	Hold time 0.8 ns
Burst configurations	BL = 4	Sequential Bursts	-

Table 1.1 MT48LC8M8A2 memory device configuration [1]

Microprocessor Clock Settings	CLK: 25 MHz	CLK2: 50 MHz	SDRAM CLK (After PLL/VCO): 100 MHz

Table 1.2 Intel 386™ DX microprocessor Clock configuration [2]

As detailed in Table 1.2, we are provided the Clock values of CLK (25 MHz) and CLK2 (50MHz) that are obtained from the microprocessor's internal functioning however the SDRAM clock is obtained from a PLL/VCO unit where CLK2 is multiplied in Phase giving double the frequency of 100 MHz for the SDRAM controller. This clock multiplier logic is as shown in Figure 1.2 below.

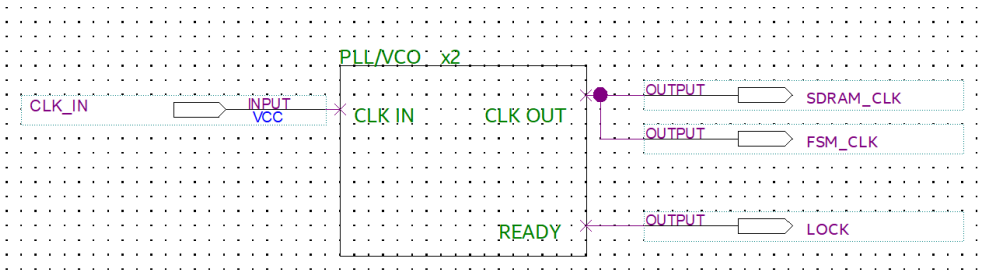


Figure 1.2 PLL/VCO Logic for clock multiplication of 2 to generate 100MHz from 50 MHz source.

To complete the top-level design, we then realize these modules in Verilog and an RTL Netlist view of the Top-level diagram in Figure 1.1 is as shown in figure 1.3.

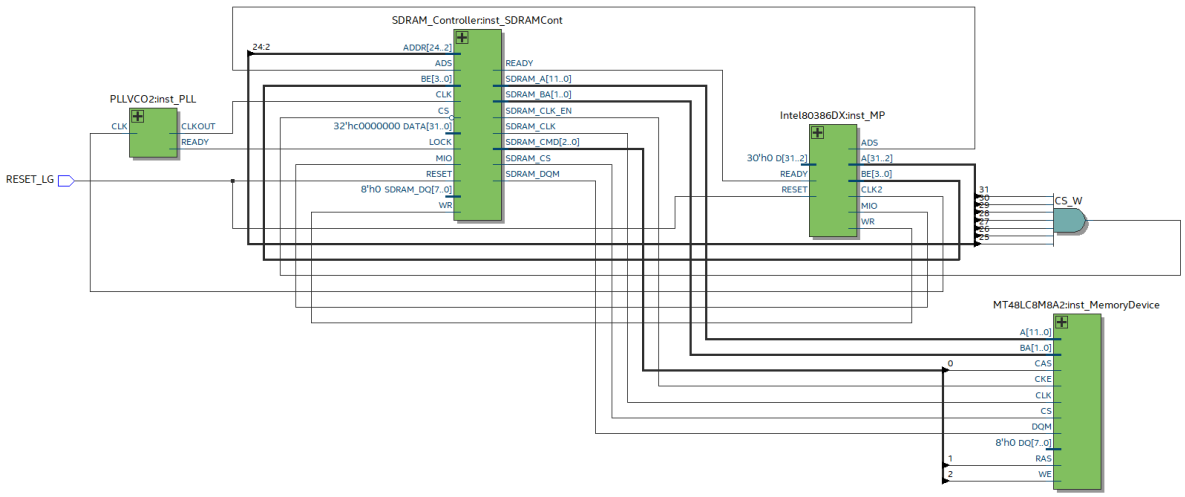


Figure 1.3 Top-level Design of the interfacing between Intel 386™, SDRAM Controller, and Memory Device

Using the PLL/VCO we assume that the clock is multiplied by 2 giving us 100 MHz at the CLKOUT pin of the PLL instantiation (in figure 1.2) and the READY signal (from inst_PLL) identifies that the clock is stable and ready to be used by the SDRAM controller. It is to note here that the Bi-directional Data buses are not connected explicitly in the figure since the SDRAM Controller Data output at DQ must be latched to the Microprocessor which is as shown in figure 2.2. The Top-Level Design considers an input RESET_LG connected to the microprocessor and the SDRAM controller at input pin. RESET_LG serves a purpose of reset both the FSM state of the SDRAM controller to RESET state and the Microprocessor to reset (reboot) anticipating resetting both units imitating a ‘boot-up sequence’.

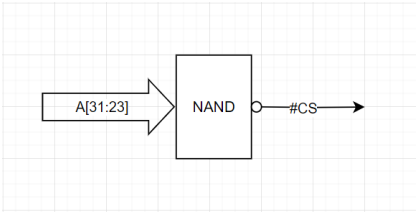


Figure 1.4 Memory Decoder Block (DCD) Logic implemented for the SDRAM Controller

SDRAM Controller Design

The SDRAM functions on portioning the incoming address bus from the Microprocessor and assigned properties of the address bus are used for later described functions in the SDRAM interface. The SDRAM Controller works on basis of a Finite state machine which (in top level) is as shown in Figure 2.1:

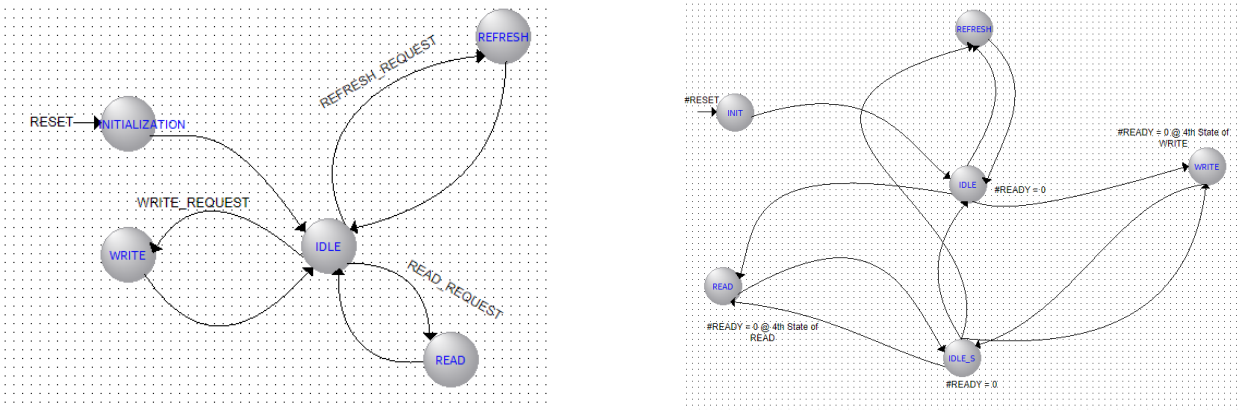


Figure 2.1 SDRAM Controller General Top-Level FSM detailing operation states (left) and detailed FSM for #READY signal generation (right)

Further expanding the operation states, we await a RESET command, which for our design will be expected as the Boot-up command for the SDRAM controller. After which the SDRAM controller is placed in the INITIALIZATION state which is further broken down as shown in Figure 2.2 and the state transition table in Table 2.2.

For our design the Following were the Address bus partitions from the microprocessor address bus to the SDRAM Address bus.

Intel 386™ Address Bus	A31-A23	A24-A23	A22-A11	A10-A2	A1-A0
Pin Designation	DCD	BA1, BA0	Row address	Column Address	XX

Table 2.1 Microcontroller Address Bus pin partitions with respective pin designation/property

INITIALIZATION

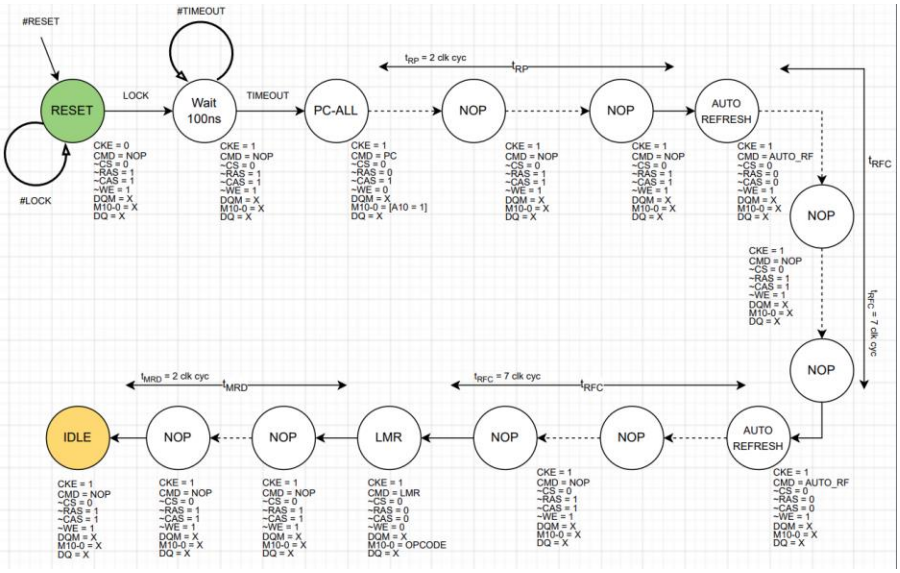


Figure 2.2 SDRAM Initialization FSM.

STATE	CONDITION	NEXT STATE	Outputs of Current State (#CS, #RAS, #CAS, #WE) (CKE)
-	#RESET = 0	RESET	(1, x, x, x) (0)
RESET	LOCK = 1	*WAIT 100 µs*	(0, 1, 1, 1) (1)
WAIT 100 US	TIMEOUT = 1	PC_ALL	(0, 1, 1, 1) (1)
PC ALL	-	NOP (till t_{RP})	(0, 0, 1, 0) (1)
NOP (till t_{RP})	-	AUTO REFRESH	(0, 1, 1, 1) (1)
AUTO REFRESH	-	NOP (till t_{RFC})	(0, 0, 0, 1) (1)
NOP (till t_{RFC})	-	AUTO REFRESH	(0, 1, 1, 1) (1)
AUTO REFRESH	-	NOP (till t_{RFC})	(0, 0, 1, 0) (1)
NOP (till t_{RFC})	-	LMR	(0, 1, 1, 1) (1)
LMR	-	NOP (till t_{MRD})	(0, 0, 0, 0) (1)
NOP (till t_{MRD})	-	IDLE	(0, 1, 1, 1) (1)
IDLE	-	-	(0, 1, 1, 1) (1)

Table 2.2 SDRAM Initialization FSM state transition table

As shown in the Initialization phase, we activate the memory device upon the startup "~RESET" signal reaches LOW. The Initialization requires us to set the parameters to the memory device where we configure it to work with our defined parameters in Table 1.1, we load the LMR with an opcode:

(x, x, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0)

As shown in Figure 2.2 and Table 2.1, we have a Wait state which is run by a counter which is designed as shown in Figure 2.4 which generates the TIMEOUT signal to go to the next state.

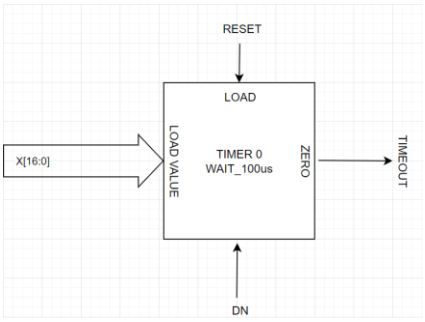


Figure 2.4 TIMEOUT Control Signal Generator for waiting 100µs (counts down to 0 from x load value)

$$t = \frac{1}{100MHz} = 10ns$$

$$for\ delay\ at\ 100\mu s\ (100{,}000\ ns), \frac{100{,}000ns}{10ns} = 10{,}000, Thus\ x = (10{,}000)_{10} = (0010011100010000)_2$$

After the memory device is initialized, we are in an IDLE state where the memory device is awaiting command from the Microprocessor. The next states (from IDLE) as shown in the Top-Level design are dependent on 2 control signals from the microprocessor which are elaborated in Table 2.3 and logic shown in Figure 2.5.

Signal Name	Hardware Designation	Description
WRITE_REQUEST	MWTC#	LOW determines CPU Write Request
READ_REQUEST	MRDC#	LOW determines CPU Memory Read Request
REFRESH_REQUEST	RRQ	HIGH determines DRAM refresh Request

Table 2.3 SDRAM Internal Control Signals

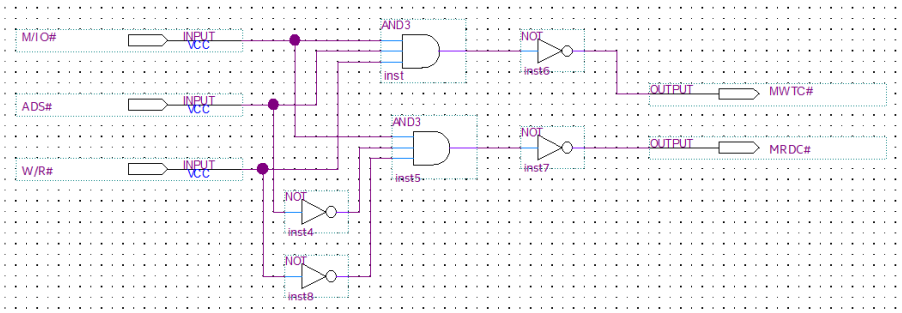


Figure 2.5 SDRAM Internal Control Signals generation logic

Before Proceeding with the Write or Read states of the Microprocessor, we need to generate a READY# signal for the CPU to acknowledge that the device is available for carrying out transfers. For that Figure 2.1 (right) shows a Top-Level FSM that is a bit more specific to the READY# signal generation. We can observe we load READY for almost 3 SDRAM clock cycles which will be close to a complete clock period of the Microprocessor.

READ

If, for example the Microprocessor has a Valid address on the Address bus and wants to read the memory (ADS# = 0, M/IO# = 1, W/R# = 0) the internal logic block in Figure 2.5 will issue an MRDC# command where we begin to read to Memory. The read operation is as shown in Figure 2.6. It is to note that #READY is issued at Precharge State followed till Shadow Idle state (IDLE_S) equaling 4 SDRAM Clock cycles matching 1 processor cycle.

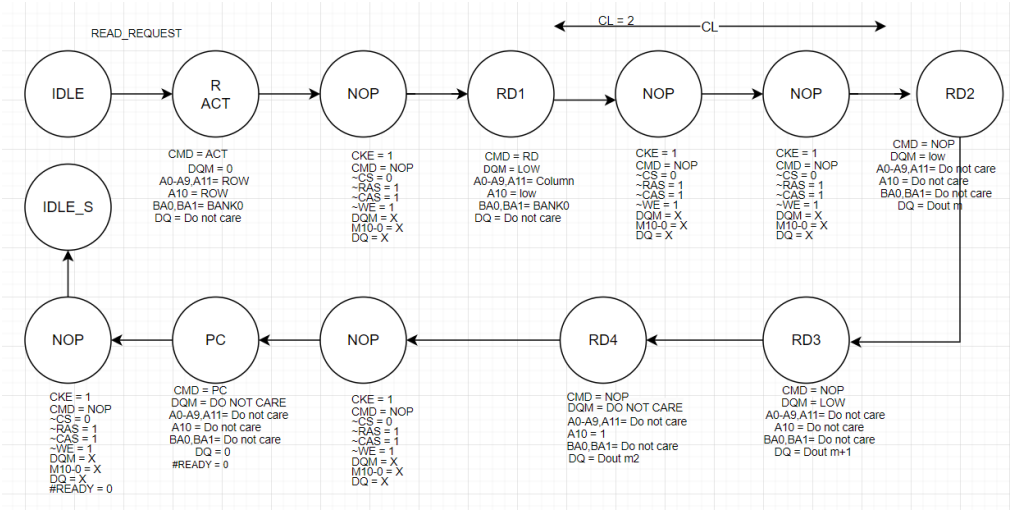


Figure 2.6 SDRAM Memory READ FSM

STATE	CONDITION	NEXT STATE	Outputs of Current State
IDLE	READ_REQUEST	READ_ACTIVATE	NOP
READ ACTIVATE	-	NOP	READ_ACTIVATE
NOP	-	RD	NOP
RD	-	NOP	READ
NOP	-	NOP	NOP
NOP	-	RD1	NOP
RD1	-	RD2	RD1 Signal -> Latch & NOP
RD2	-	RD3	RD2 Signal -> Latch & NOP
RD3	-	RD4	RD3_Signal -> Latch & NOP
RD4	-	NOP	RD4_Signal -> Latch & NOP
NOP	-	PRECHARGE	NOP
PRECHARGE	-	NOP	#READY = 0
NOP	-	S_IDLE	#READY = 0

Table 2.4 SDRAM READ FSM State Transition Table

The Data Bus for the output (during READ) operation done by the SDRAM controller is latched to Interface with the 32-bit data bus of the Microprocessor since the memory unit only has 8-bit Data Bus. Thus, using RD_x ($x \in 1\text{-}4$) for all the Read states committed by the Memory, we assert the data to the Microprocessor Data bus. The output of each Latch is activated when the READY# signal is triggered so the data for all the read states is only sent to the Microprocessor data bus when a READ state is completed. It is to note that DQM is to be set at low for valid read requests from the memory.

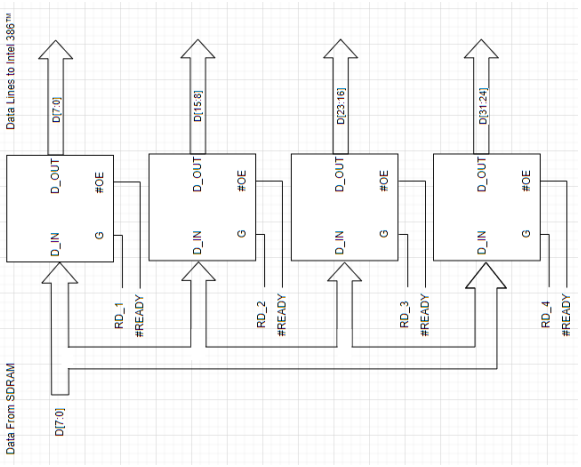


Figure 2.7 Latching of Data OUT from 8-bit SDRAM memory to Microprocessor of 32-bit Data Bus

WRITE

If, for example the Microprocessor has a Valid address on the Address bus and wants to Write the memory (ADS# = 0, M/IO# = 1, W/R# = 1) the internal logic block in Figure 2.5 will issue an MWTC# command where we begin to write to Memory. The operation of writing to the memory is shown in Figure 2.7.

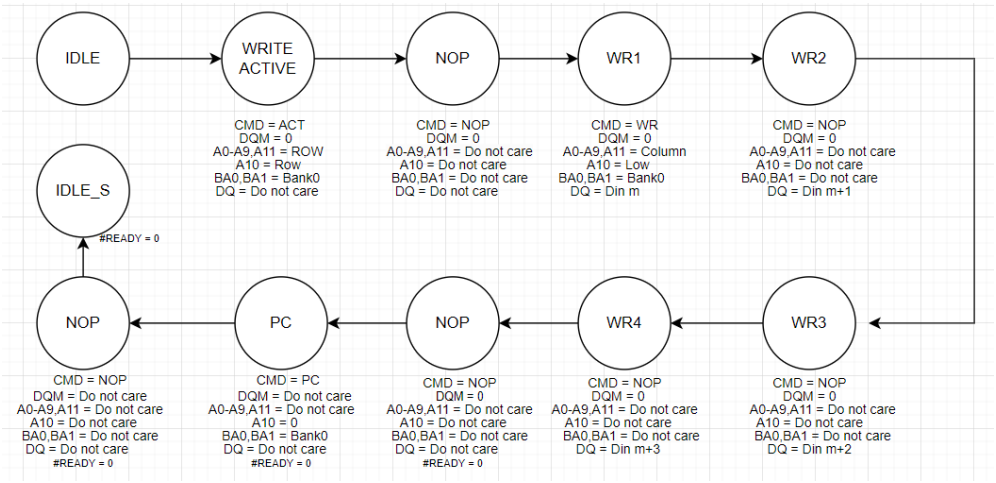


Figure 2.7 SDRAM Write FSM

A Write is initiated from the WRITE_REQUEST signal and the controller proceeds to Activate the address specified followed by activating a sequential write on the memory provided that DQM is Valid (0) for Write states.

STATE	CONDITION	NEXT STATE	Outputs of Current State
IDLE	WRITE COMMAND	WRITE ACTIVATE	-
WRITE ACTIVATE	-	NOP	NOP
NOP	-	WR1	WR CMD
WR1	-	WR2	NOP
WR2	-	WR3	NOP
WR3	-	WR4	NOP
WR4	-	NOP	NOP
NOP	-	PRECHARGE	#READY = 0
PRECHARGE	-	NOP	#READY = 0
NOP	-	S_IDLE	#READY = 0

Table 2.5 SDRAM WRITE FSM State Transition Table

REFRESH

As specified in Table 1.1, we are to refresh the SDRAM memory device on specific intervals, this is treated as an interrupt since a counter (TIMER 1) is deployed when the SDRAM is initialized. Figure 2.7 shows the SDRAM refresh operation.

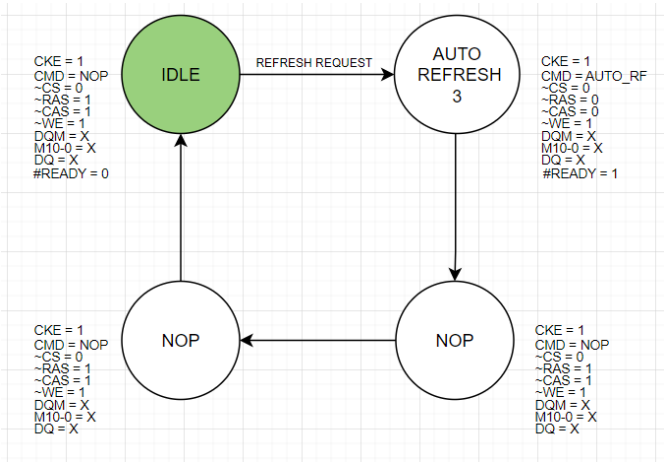


Figure 2.7 SDRAM Refresh FSM (REFRESH_REQUEST is generated by TIMER 1)

STATE	CONDITION	NEXT STATE
IDLE	REFRESH_REQUEST = 1	AUTO REFRESH
AUTO REFRESH	CLR_RR = 1 && REFRESH REQUEST = 0	NOP
NOP	tRFC (7 CYCLES)	IDLE

Table 2.6 SDRAM Refresh FSM State Transition Table

TIMER1, that generates the REFRESH_REQUEST signal is as shown in Figure 2.9 with calculated Load Value ‘Y’ for Triggering the interrupt. The Request is also cleared using ‘CLR_R_R’ input to the register.

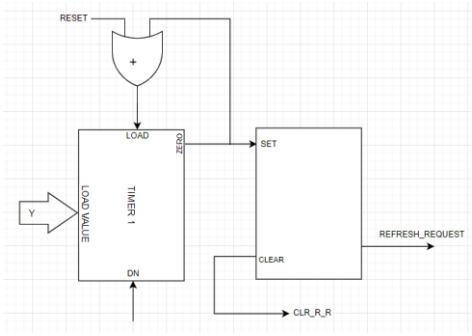
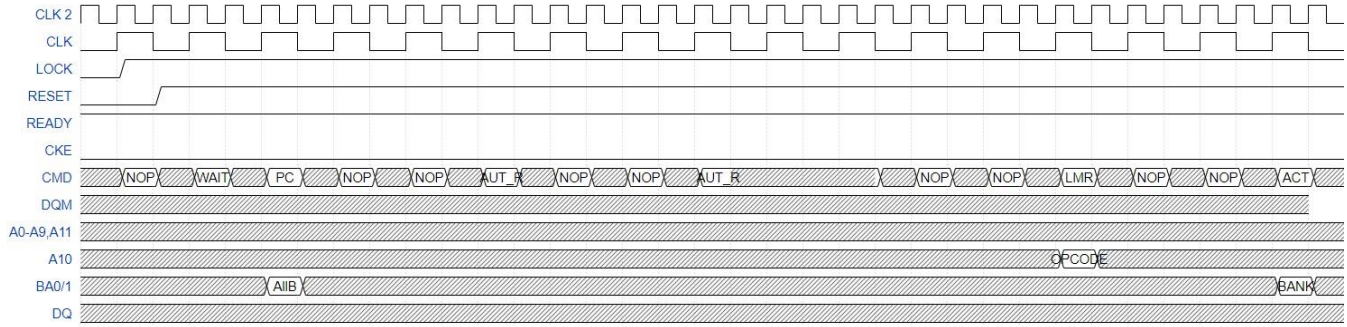


Figure 2.8 TIMER1 to generate REFRESH_REQUEST signal.

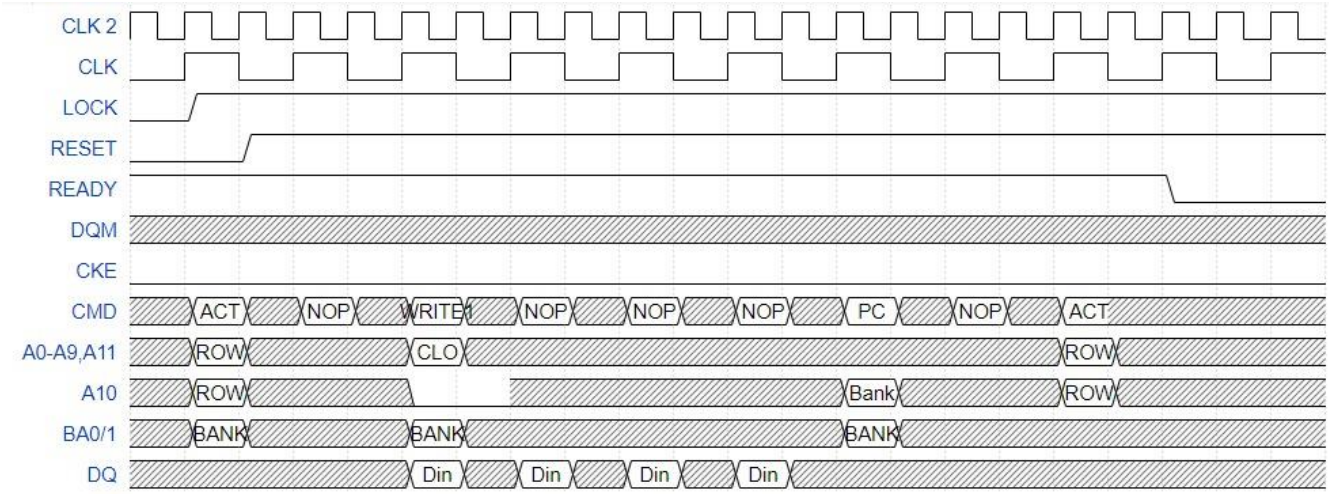
$$\frac{64\ ms}{\#\ of\ rows} = \frac{64\ ms}{4096} = 15.6\ us, thus\ \frac{15600ns/row}{10ns} = Y = 1560\ rows$$

Timing Analysis

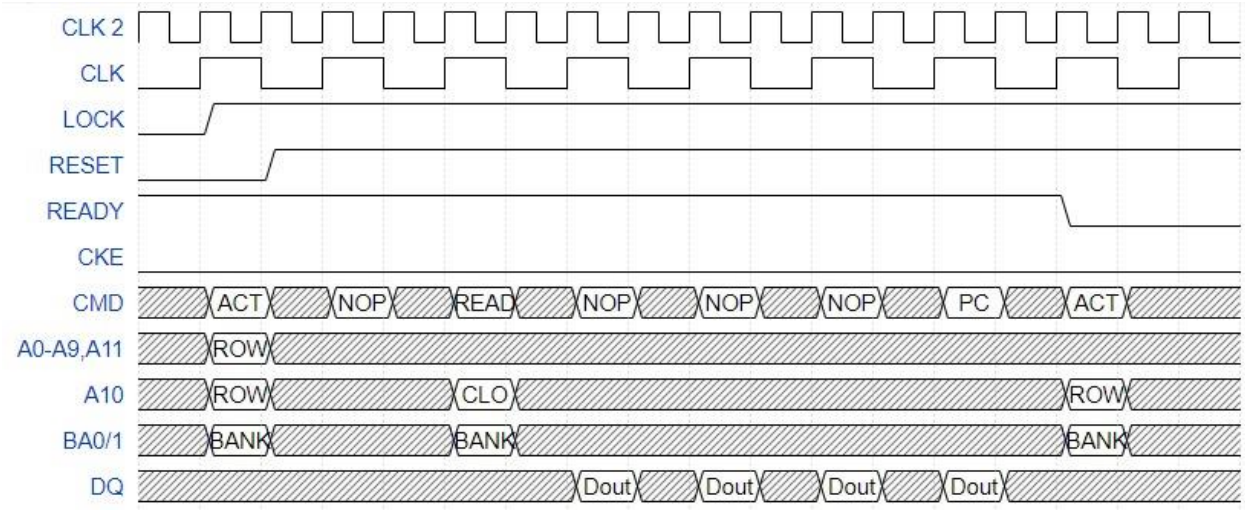
Initialization Timing Diagram



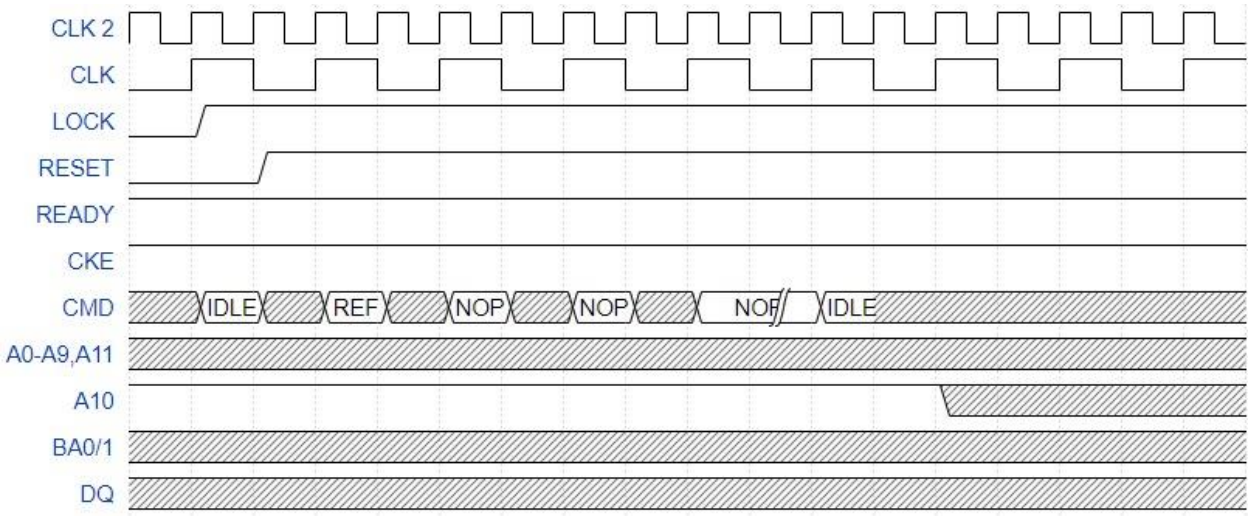
Write Timing Diagram



Read Timing Diagram



Refresh Timing Diagram



References

[1] Micron, Staff. (2000). *64MB: X4, X8, X16 SDRAM - Micron Technology*. Synchronous DRAM MT48LC16M4A2 – 4 Meg x 4 x 4 banks. https://media-www.micron.com/-/media/client/global/documents/products/data-sheet/dram/64mb_x4x8x16_sdram.pdf?rev=c8fbbb9fc6534202a79725709cb0486f

[2] Intel Corporation, S. (1995). *Intel386 DX datasheet*. INTEL386 DX Datasheet pdf - 32-BIT CHMOS MICROPROCESSOR WITH INTEGRATED MEMORY MANAGEMENT - Intel. https://www.datasheetcatalog.com/datasheets_pdf/I/N/T/E/INTEL386_DX.shtml