Objective:

The objective of this report is to design SDRAM controller to interface SDRAM memory i.e., MT48LC16M4A2 with 80386DX microprocessor having only asynchronous memory support. This report shows the complete design and theory of operations.

A 132MHZ clock source is used to clock the SDRAM controller and SDRAM at speed grade -7E.

Specifications:

- MT48LC16M4A2 memory (4Meg * 4* 4banks)
- 4096 ROWS * 1024 Columns
- Clock Cycle Time (T_{clcl}) =7.5ns
- Speed Grade -7E
- Burst Length BL=8
- CAS Latency CL=2
- Refresh Time (4096 rows) = 64ms
- Frequency of Operation 132MHZ

INTERFACING DIAGRAM:

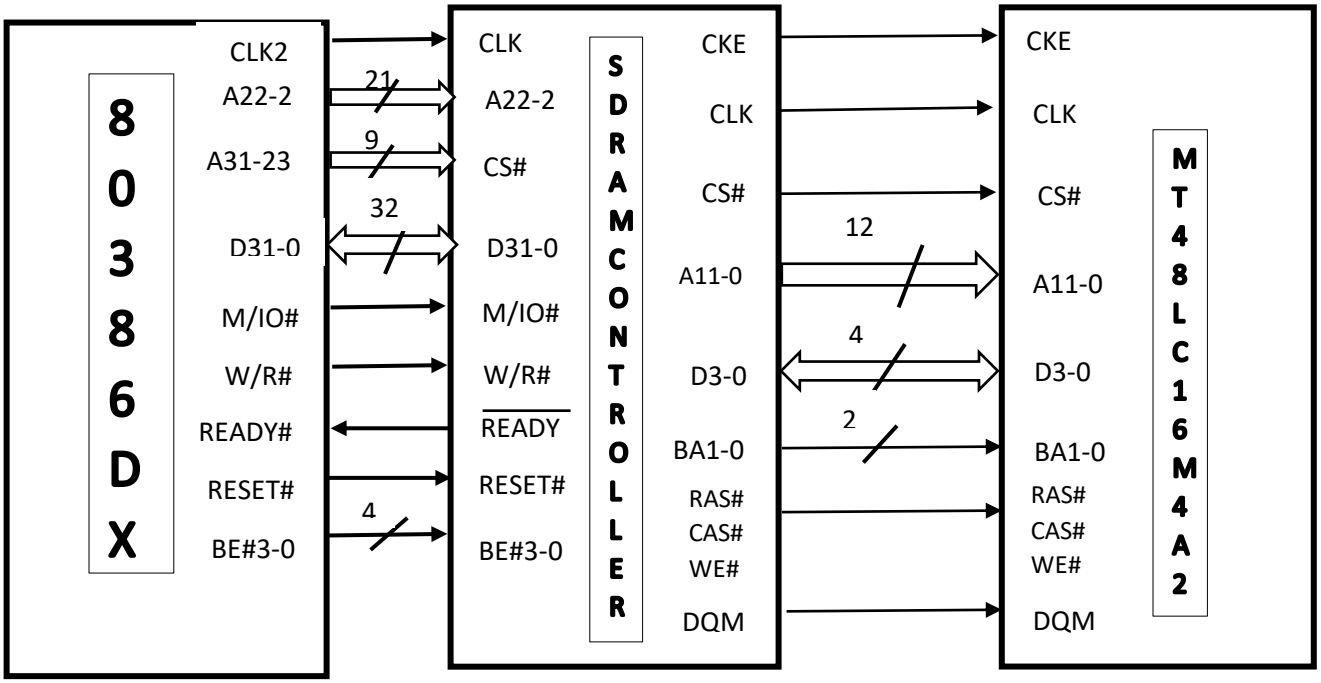
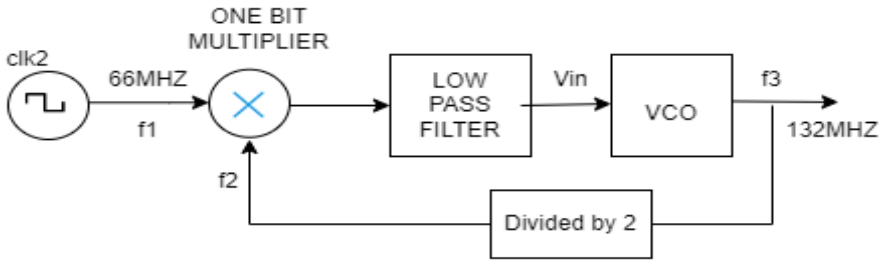
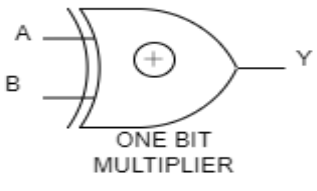


Figure 1: SDRAM controller interfaced with the 80386DX and SDRAM - MT48LC16M4A2

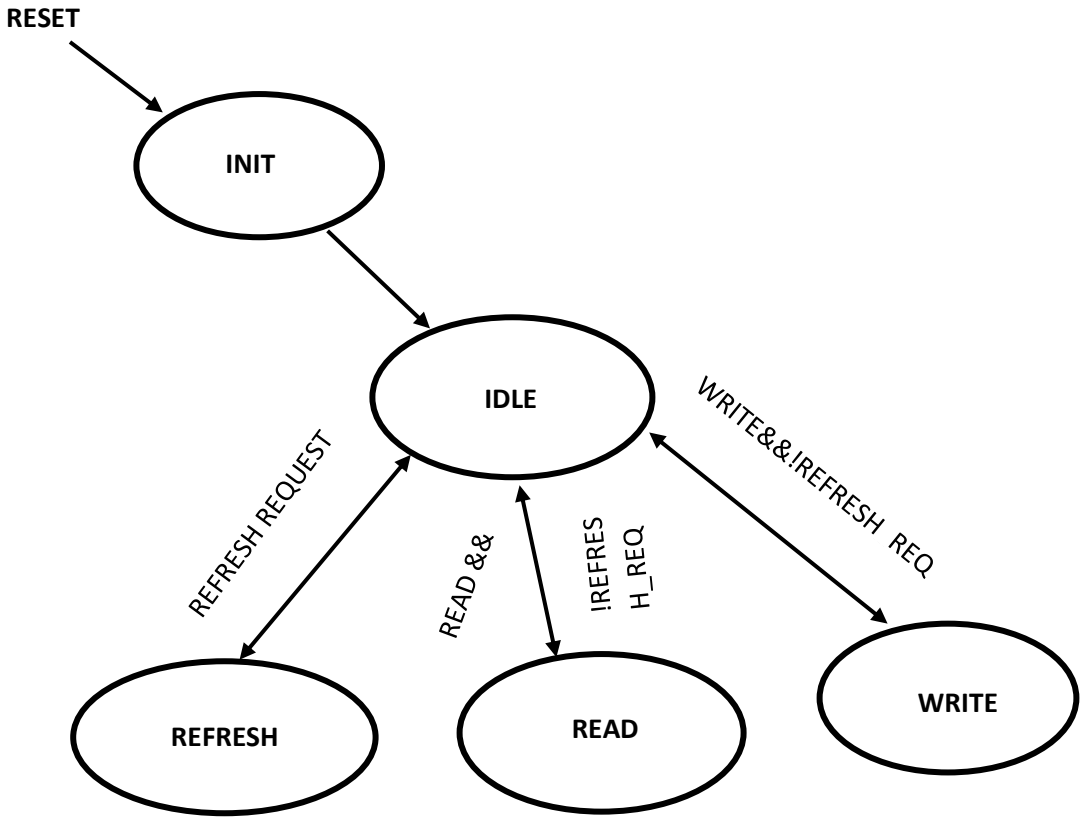
PHASE LOCKED LOOP:





one bit multiplier		
0	0	0
0	1	1
1	0	1
1	1	0

MAIN STATE DIAGRAM:



State Transition Table:

Current state	Condition	Next state
NA	Reset# = 0	INITIALIZATION
Initialization	INIT_Done	Idle
Idle	Refresh_Request	Refresh
Refresh	Refresh_Request_done	Idle
Idle	Read && (!Refresh_Req)	Read
Read	Read_Done	Idle
Idle	Write && (!Refresh_Req)	Write
Write	Write_Done	Idle

Memory Address Mapping:

DCD_SDRAM	BA [1:0]	ROW	COL	⛌ ⛌
A31-23	A22-21	A20-9	A8-2	⛌ ⛌
9bits	2bits	12bits	7bits	

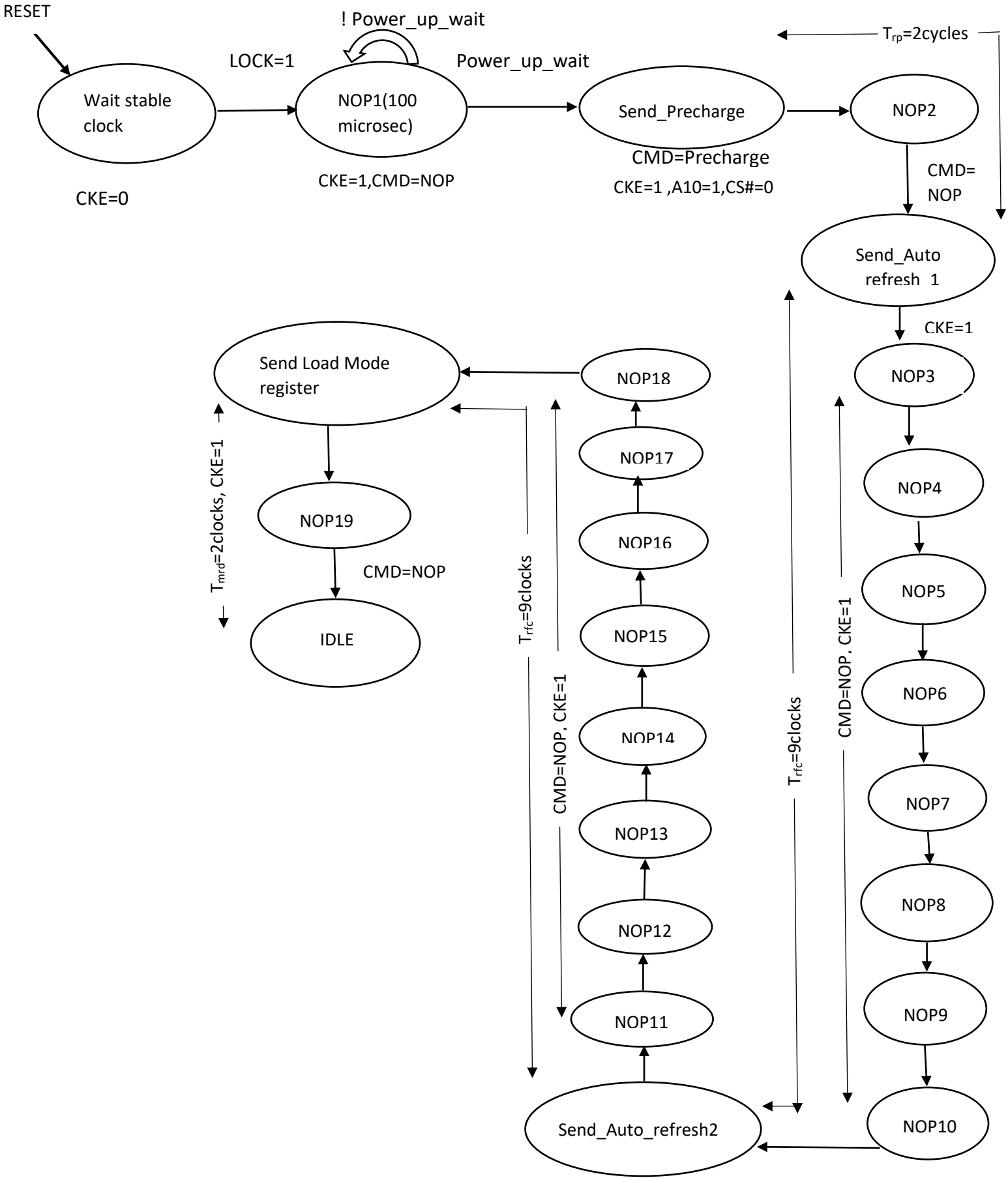
INITIALIZATION STEPS:

SDRAMs must be powered up and initialized in a predefined manner. Operational procedures other than those specified may result in undefined operation. After power is applied to VDD and VDDQ (simultaneously) and the clock is stable (stable clock is defined as a signal cycling within timing constraints specified for the clock pin), the SDRAM requires a 100µs delay prior to issuing any command other than a NOP. Starting at some point during this 100µs period and continuing at least through the end of this period, NOP commands must be applied. Once the 100µs delay has been satisfied with at least one NOP command having been applied, a PRECHARGE command should be applied.

Wait up to t_{RP} time during this time NOP2 command is given. All banks will complete their precharge and it will move to send_auto_refresh command. It waits at least t_{RFC} time i.e., 9 clocks during this time NOP3-10 commands are given. After precharge at least two Auto refresh cycles must be performed during this time we had given NOP3-NOP18 commands.

The SDRAM is now ready for mode register programming. Because the mode register will power up in an unknown state, it should be loaded with desired bit values prior to applying any operational command. Using the LOAD MODE REGISTER command, program the mode register. Wait at least t_{MRD} time, during this time we applied NOP19 command as t_{MRD} is 2 clocks. At this point the DRAM is ready for any valid command.

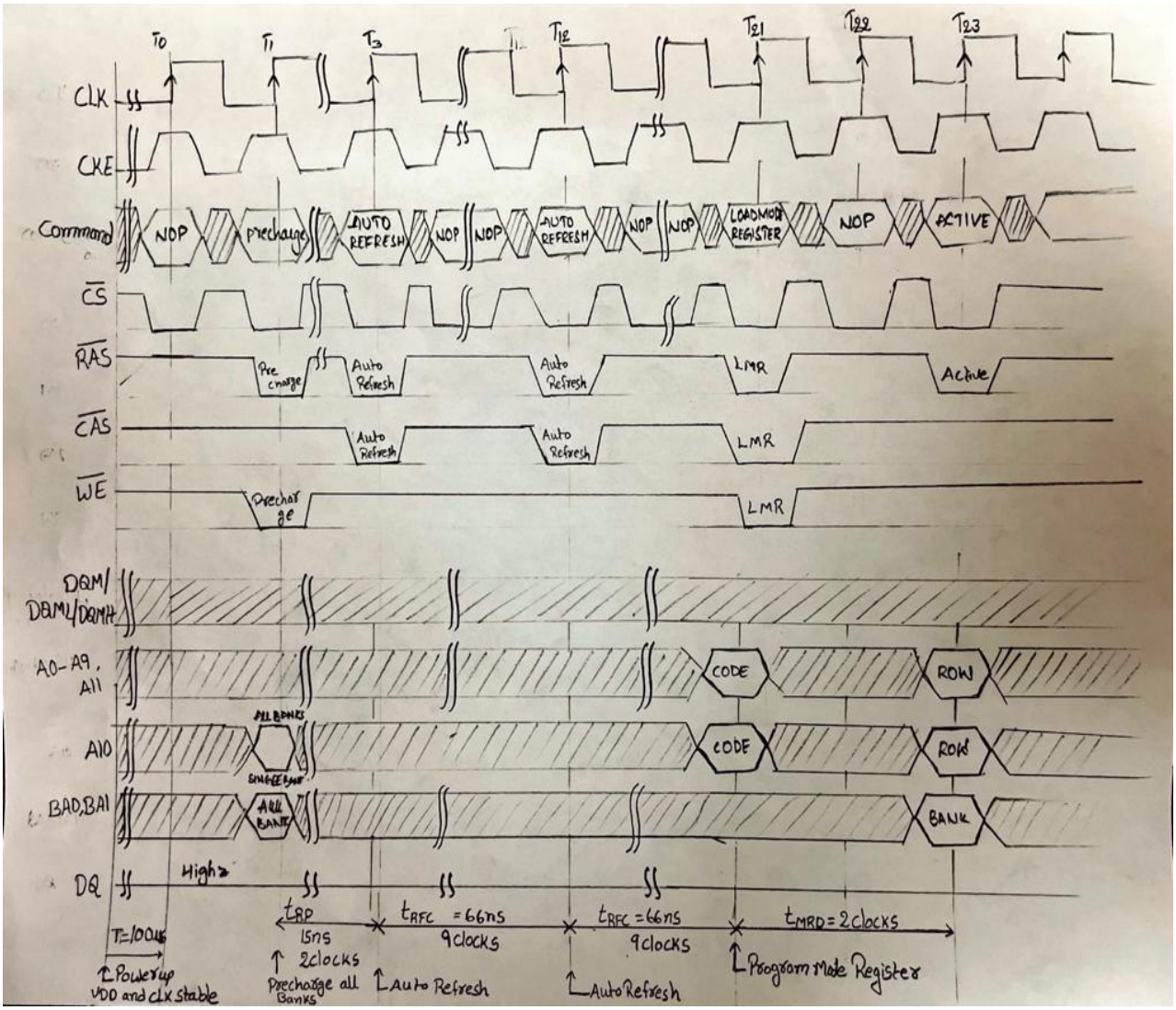
State Machine for Initialization:



State Transition Table for Initialization:

Current State	Condition	Next State
NA	Reset#=0	Reset
Reset	Next Clock, CKE=0	Wait Stable Clock
Wait Stable Clock	Lock=1	NOP1(100microsec)
NOP1	Power_up_wait! =0	NOP1
NOP1	Power_up_wait=0, CKE=1	Send_precharge
Send_precharge	⊠	NOP2
NOP2	⊠	Send_Auto refresh 1
Send-Auto refresh 1	⊠	NOP3
NOP3	⊠	NOP4
NOP4	⊠	NOP5
NOP5	⊠	NOP6
NOP6	⊠	NOP7
NOP8	⊠	NOP9
NOP9	⊠	NOP10
NOP10	⊠	Send_Auto refresh2
Send-Auto refresh2	⊠	NOP11
NOP11	⊠	NOP12
NOP12	⊠	NOP13
NOP13	⊠	NOP14
NOP14	⊠	NOP15
NOP15	⊠	NOP16
NOP16	⊠	NOP17
NOP17	⊠	NOP18
NOP18	⊠	Send Load Mode register
Send Load Mode register	⊠	NOP19
NOP19	⊠	IDLE

CLOCK CALCULATIONS:	Value (ns)	no.of clocks
NOP1(100microsec)	100microsec	13,333
Precharge period (T_{rp})	15ns	2clocks
Auto Refresh period (T_{rfc})	66ns	9clocks
Load Mode Register period (T_{mrd})	15ns	2clocks



INITIALIZATION OUTPUT TABLE:

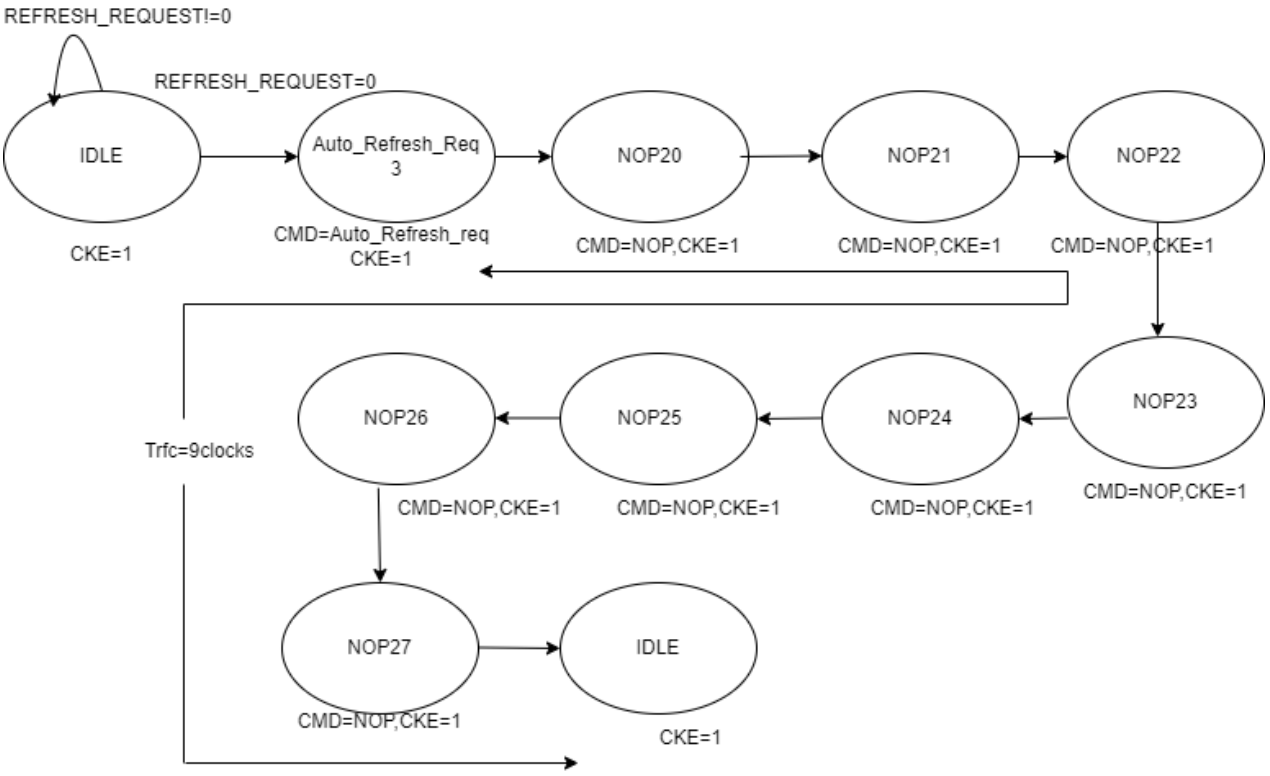
Output	Active Condition
CKE	State==((NOP_n)||(Precharge)||($Auto\_refresh_m$) || (Load mode register) ||(Idle)) (Here n=1-19, m=1,2)
CS#	State==((NOP_n)||(Precharge)||($Auto\_refresh_m$) || (Load mode register) ||(Idle)) (Here n=1-19, m=1,2)
RAS#	State==((Precharge)||($Auto\_refresh_m$) || (Load mode register)) (Here m=1,2)
CAS#	State==(($Auto\_refresh_m$) || (Load mode register)) (Here m=1,2)
WE#	State== (Precharge ||Load Mode Register)
A0-9, A11	State== (Load Mode Register || Idle)
A10	State== (Precharge || Load Mode Register || Idle)
DQ	State=High z
BA0, BA1	State== (precharge|| Idle)

LOAD MODE REGISTER:

Function	Function Value	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Burst length (BL)	8										0	1	1
Burst Type	Sequential									0			
CAS Latency	2						0	1	0				
Op mode	Standard operation				0	0							
Write Burst Mode	Programmed Burst Mode			0									
Reserved	--	0	0										

LMR: 00 0 00 010 0 011

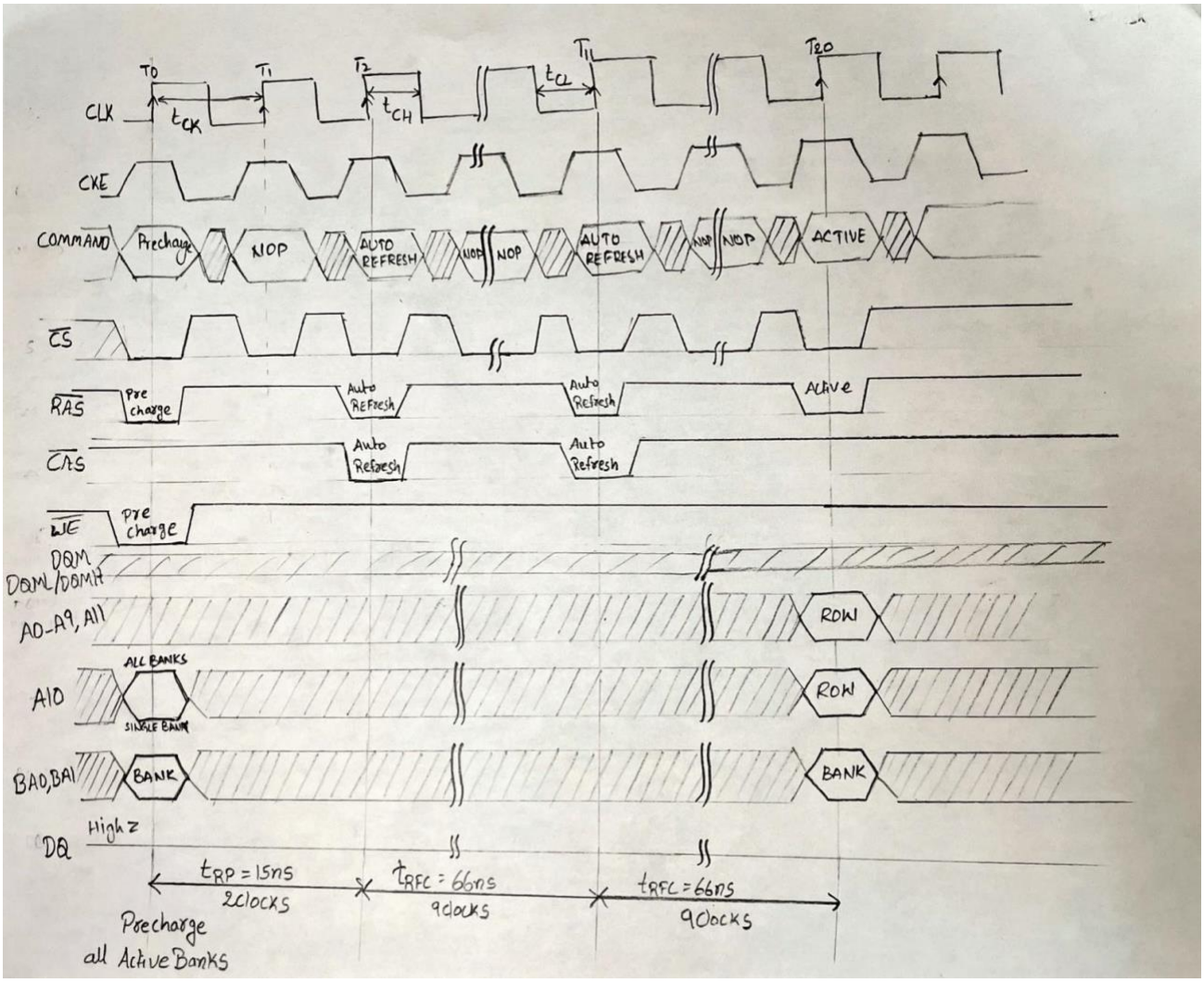
State Machine for Auto Refresh:



CLOCK CALCULATIONS:	Value (ns)	no.of clocks
Precharge period (T_{rp})	15ns	2clocks
Auto Refresh period (T_{rfc})	66ns	9clocks

Auto Refresh STEPS:

This ⛒ 4-bit wide memory has 4096 rows and 1024 columns in each of the 4 banks. So, for 64Mb SDRAM 4096 Auto Refresh cycles require after every 64ms.A row must be refreshed after 64ms/4096=15.6micro sec. SDRAM has internally refresh controller which automatically goes to auto refresh state after every 15.6 micro sec. Which is done by signal named REFRESH_REQUEST#, when it goes to zero then it enters Auto_Refresh_Req state. After that it moves to NOP20-27 states to cover t_{RFC} time i.e., 9clocks and after that it goes back to idle state.



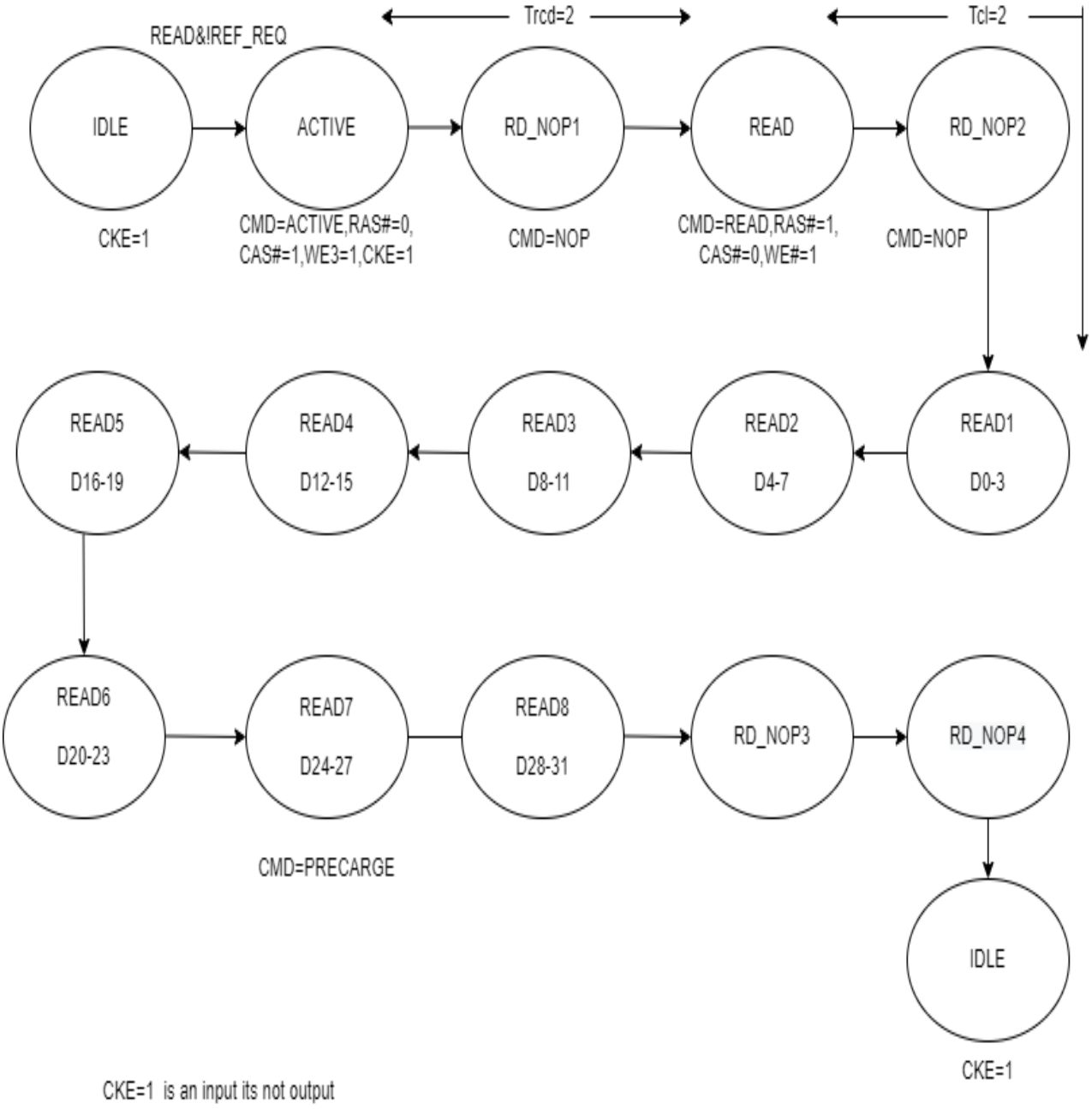
State Transition Table for Auto Refresh:

Current State	Condition	Next State
IDLE	Refresh_Request! = 0	IDLE
IDLE	Refresh_Request = 0	Auto_Refresh_Req
Auto_Refresh_Req	⤫	NOP20
NOP20	⤫	NOP21
NOP21	⤫	NOP22
NOP22	⤫	NOP23
NOP23	⤫	NOP24
NOP24	⤫	NOP25
NOP25	⤫	NOP26
NOP26	⤫	NOP27
NOP27	⤫	IDLE

AUTO REFRESH OUTPUT TABLE:

OUTPUT	ACTIVE CONDITION
CKE	State==((NOP$_n$)||(Precharge)||(Auto_Refresh_Req) || (idle)) (Here n=20-27)
CS#	State==((NOP$_n$)||(Precharge)||(Auto_Refresh_Req) || (idle)) (Here n=20-27)
RAS#	State==((Precharge)||(Auto_Refresh_Req))
CAS#	State==Auto_Refresh_Req
WE#	State==Precharge
A0-9, A11	State==Active
A10	State== (precharge|| Active)
BA0, BA1	State== (precharge|| Active)
DQ	High Z

State Machine for Read:



CKE=1 is an input its not output

Calculation for Read:

CAS Latency CL=2clocks

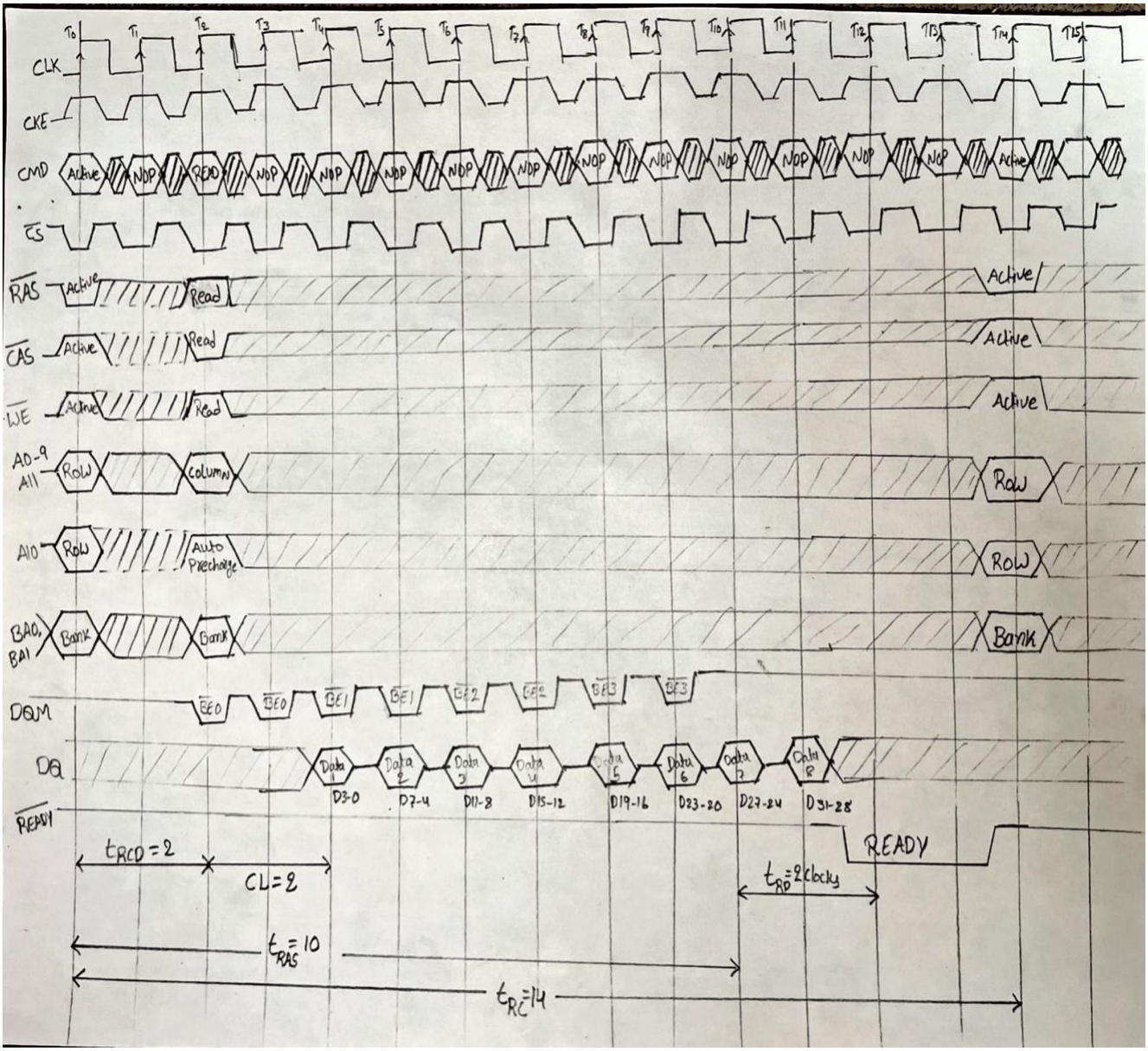
Active to Read/write delay T_{rcd}=15ns=2clocks

Precharge command period T_{rp}=15ns = 2clocks

Active to precharge command T_{RAS}=37ns = 5clocks

Read STEPS:

The READ command is used to initiate a read access to an active row. When READ&! REF_REQ goes to zero then it moves to Active state where RAS#=0, CAS#=1, WE#=1 by activating row and bank and this state followed by RD_NOP1 state and it moves to READ state in this state we are giving column address and the value of input A10 which determines whether auto precharge is used or not, in this case I have taken auto precharge so A10=high. After READ state based on CAS Latency it must wait. In this case CL=2 so it moves to RD_NOP2 state and next cycle it moves to READ1 it will continues up to READ8 in each read we will get 4bits of data so 4*8=32 bits of data we will get. After completion of READ then READY signal is generated and wait for 2cycles minimum because SDRAM clock is twice the faster as microprocessor so, for that purpose we are giving 2 wait cycles.


T0
T1
T2
T3
T4
T5
T6
T7
T8
T9
T10
T11
T12
T13
T14
T15
CLK
CKE
CMD
CS
RAS
CAS
WE
A0-9 A11
A10
BA0, BA1
DQM
DQ
READY
Active
NOP
Read
Row
Column
Auto Precharge
Bank
Data 1
Data 2
Data 3
Data 4
Data 5
Data 6
Data 7
Data 8
D3-0
D7-4
D11-8
D15-12
D19-16
D23-20
D27-24
D31-28
READY
tRCD = 2
CL = 2
tRP = 2 clocks
tRAS = 10
tRC = 14

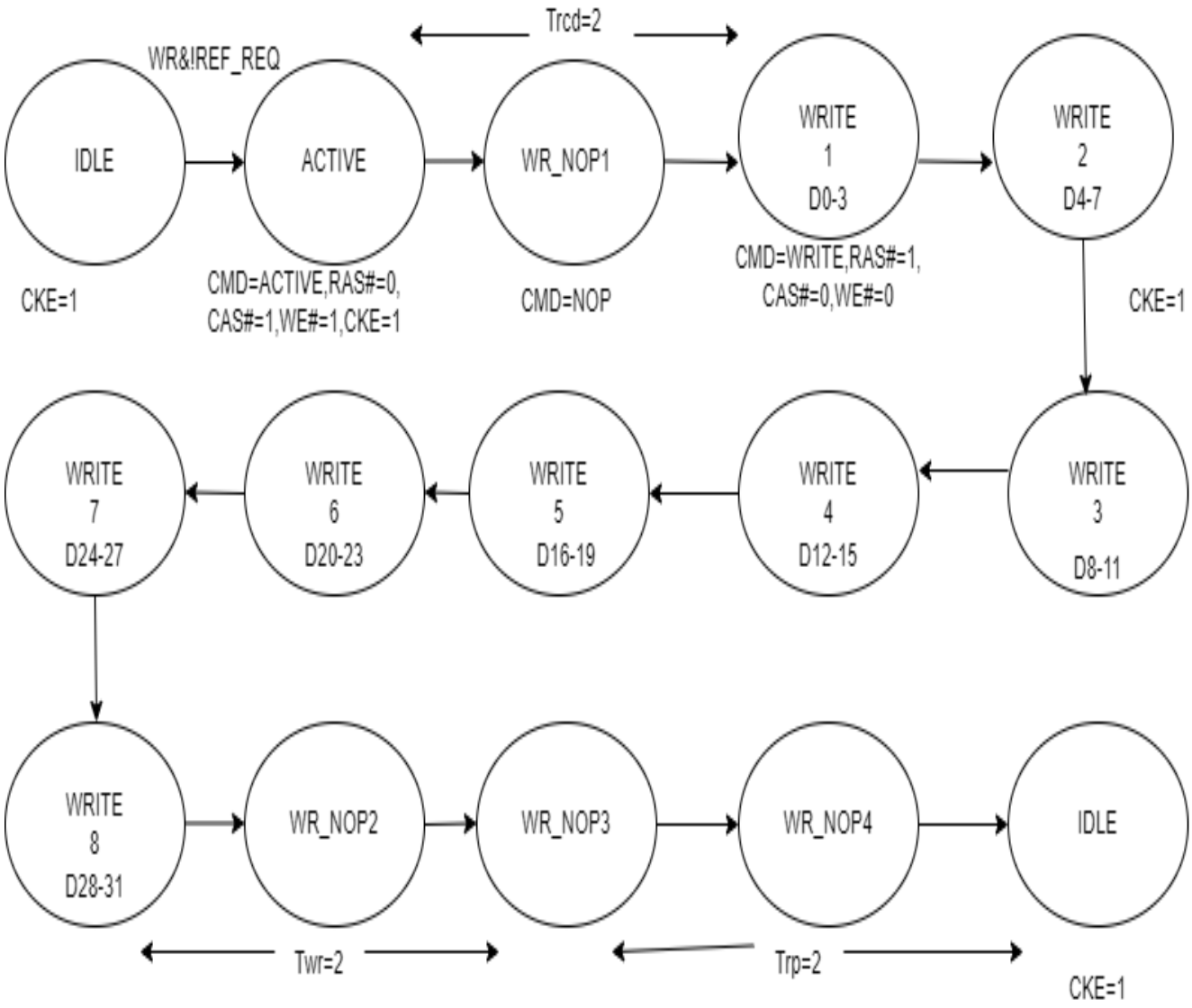
State Transition Table for Read:

Current State	Condition	Next State
IDLE	RD&! REF_REQ=0	ACTIVE
ACTIVE	⤫	RD_NOP1
RD_NOP1	⤫	READ
READ	⤫	RD_NOP2
RD_NOP2	⤫	READ1
READ1	⤫	READ2
READ2	⤫	READ3
READ3	⤫	READ4
READ4	⤫	READ5
READ5	⤫	READ6
READ6	⤫	READ7
READ7	⤫	READ8
READ8	⤫	RD_NOP3
RD_NOP3	⤫	RD_NOP4
RD_NOP4	⤫	IDLE

READ OUTPUT TABLE:

COMMAND	ACTIVE CONDITION
CKE, CS#	STATE= (ACTIVE||READ||READ1||READ2||READ3||READ4 ||READ5 ||READ6||READ7||READ8||RD_NOPm) Here m=1,2,3,4
CAS#	STATE=READ
RAS#	STATE=ACTIVE
DQ	STATE=(READ1||READ2||READ3||READ4||READ5||READ 6||READ7||READ8)
DQM	STATE=(READ||RD_NOP2||$READ_{1\text{-}6}$)
A0-9, A11	STATE=(READ||ACTIVE)
A10	STATE=(READ||ACTIVE||PRECHARGE)
BA0, BA1	STATE=(ACTIVE||READ)

State Machine for WRITE:



CLOCK CALCULATION FOR WRITE:

T_{clcl}=7.5ns

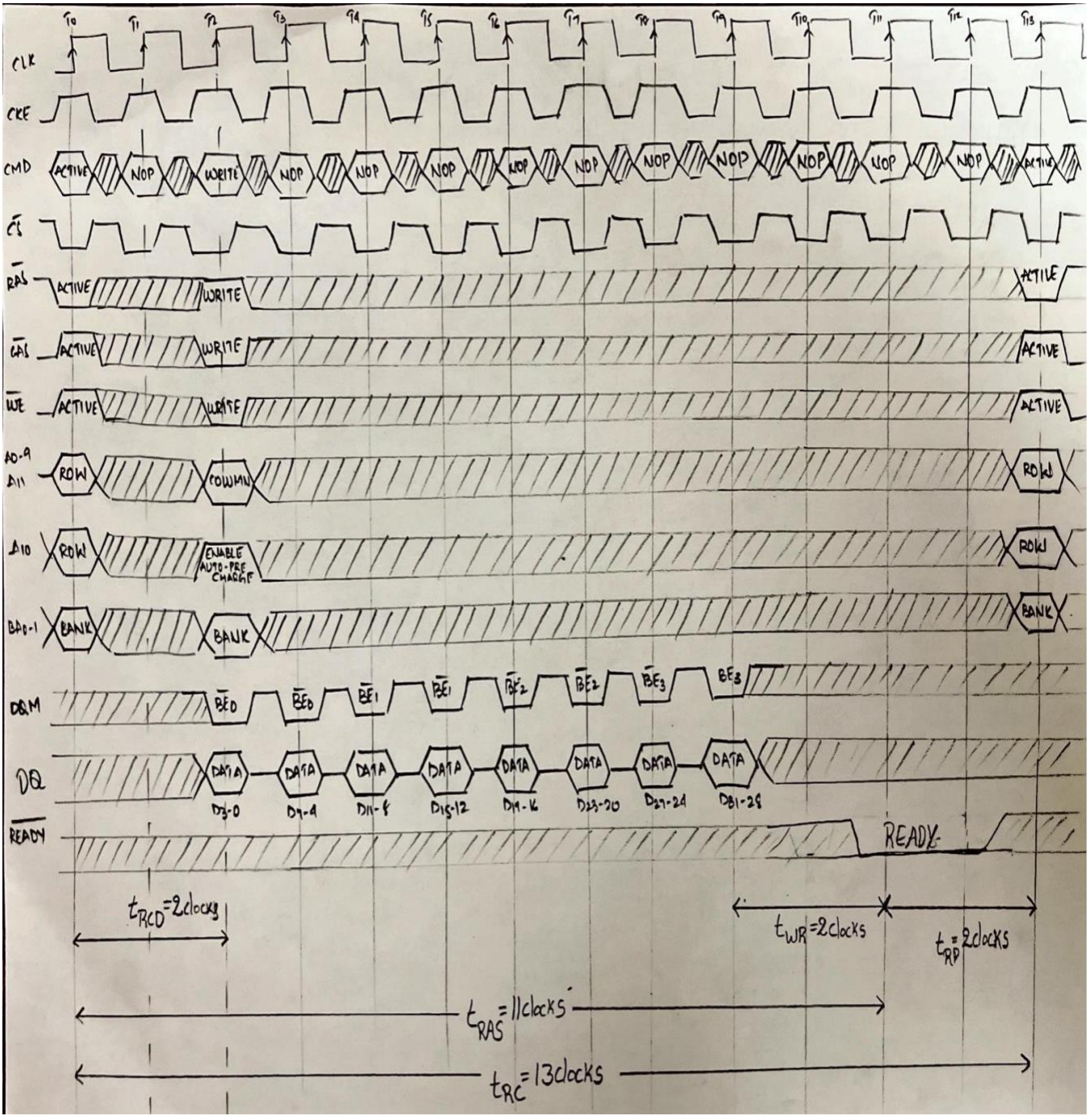
Active to Read /Write delay t_{rcd}=15ns=2clocks

Write Recovery time t_{wr}=1clk+7ns=2clocks

Precharge Command period t_{rp}=15ns=2clocks

WRITE STEPS:

The WRITE command is used to initiate a WRITE access to an active row. When WRITE&! REF_REQ goes to zero then it moves to Active state where RAS#=0, CAS#=1, WE#=1 by activating row and bank and this state followed by WR_NOP1 state and it moves to WRITE state in this state we are giving column address and WRITE1 will happen in WRITE state only and it will continue up to WRITE8 in each write we will get 4bits of data so 4*8=32 bits of data we will get. After completion of WRITE then it moves to WR_NOP2 state as t_{WR}=2, It takes 2cycles from WRITE8 to WR_NOP3 state. Next t_{rp}=2 so it takes 2 cycles from WR_NOP3 to IDLE.


T0 T1 T2 T3 T4 T5 T6 T7 T8 T9 T10 T11 T12 T13
CLK
CKE
CMD
ACTIVE NOP WRITE NOP NOP NOP NOP NOP NOP NOP NOP NOP NOP ACTIVE
CS
RAS
CAS
WE
ACTIVE WRITE ACTIVE
A0-9 A11
ROW COLUMN ROW
A10
ENABLE AUTO-PRECHARGE
BA0-1
BANK BANK BANK
DQM
DQ
DATA D3-0 DATA D7-4 DATA D11-8 DATA D15-12 DATA D19-16 DATA D23-20 DATA D27-24 DATA D31-28
READY
tRCD=2clocks
tWR=2clocks
tRP=2clocks
tRAS=11clocks
tRC=13clocks

State Transition Table for WRITE:

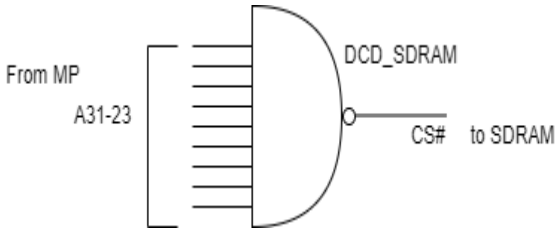
Current State	Condition	Next State
IDLE	WR&! REF_REQ =0	ACTIVE
ACTIVE	🗙	WR_NOP1
WR_NOP1	🗙	WRITE1
WRITE1	🗙	WRITE2
WRITE2	🗙	WRITE3
WRITE3	🗙	WRITE4
WRITE4	🗙	WRITE5
WRITE5	🗙	WRITE6
WRITE6	🗙	WRITE7
WRITE7	🗙	WRITE8
WRITE8	🗙	WR_NOP2
WR_NOP2	🗙	WR_NOP3
WR_NOP3	🗙	WR_NOP4
WR_NOP4		IDLE

WRITE OUTPUT TABLE:

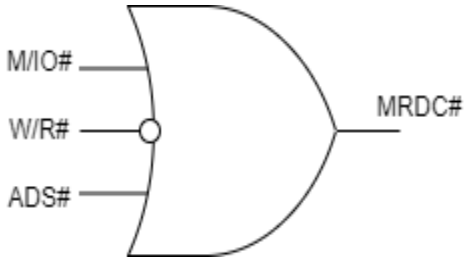
SIGNALS	ACTIVE CONDITION
CKE	STATE=(ACTIVE||$WRITE_n$||WR_NOPm) (Here n=1-8, m=1-4)
CS#	STATE=(ACTIVE||$WRITE_n$||WR_NOPm) (Here n=1-8, m=1-4)
CAS#	STATE=WRITE
RAS#	STATE=ACTIVE
WE#	STATE=WRITE
DQ	STATE=($WRITE_n$) (Here n=1-8)
DQM	STATE=($WRITE_n$) (Here n=1-8)
A0-9, A11	STATE= ACTIVE||WRITE
A10	STATE= ACTIVE||WRITE
BA0, BA1	STATE= ACTIVE||WRITE

Generation of Chip Signals

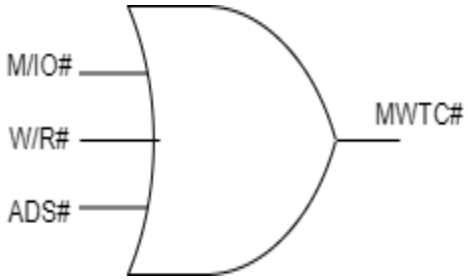
Generation of Chip Select:



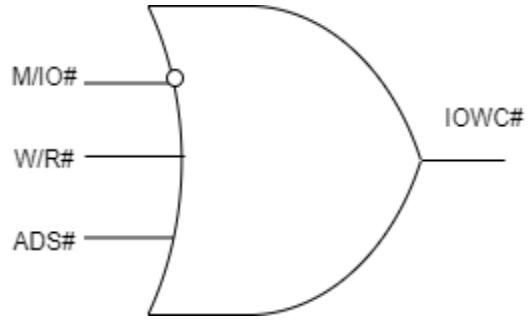
Generation of MRDC#



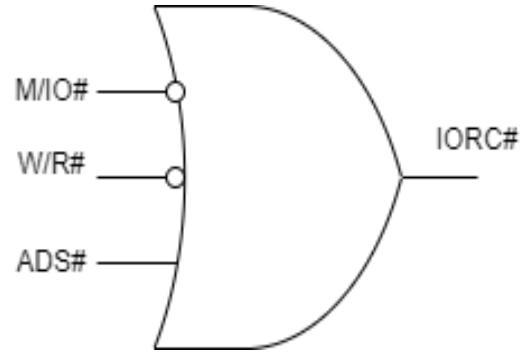
Generation of MWTC:



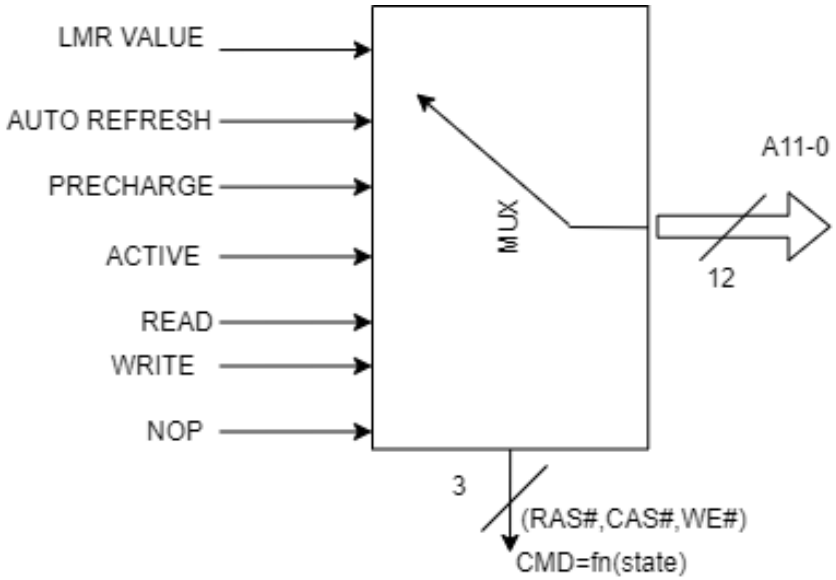
Generation of IOWC:



Generation of IORC:

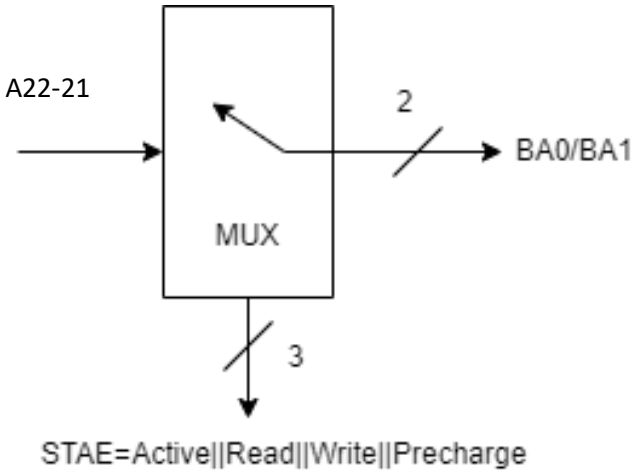


Generation of Functional Command Signal:

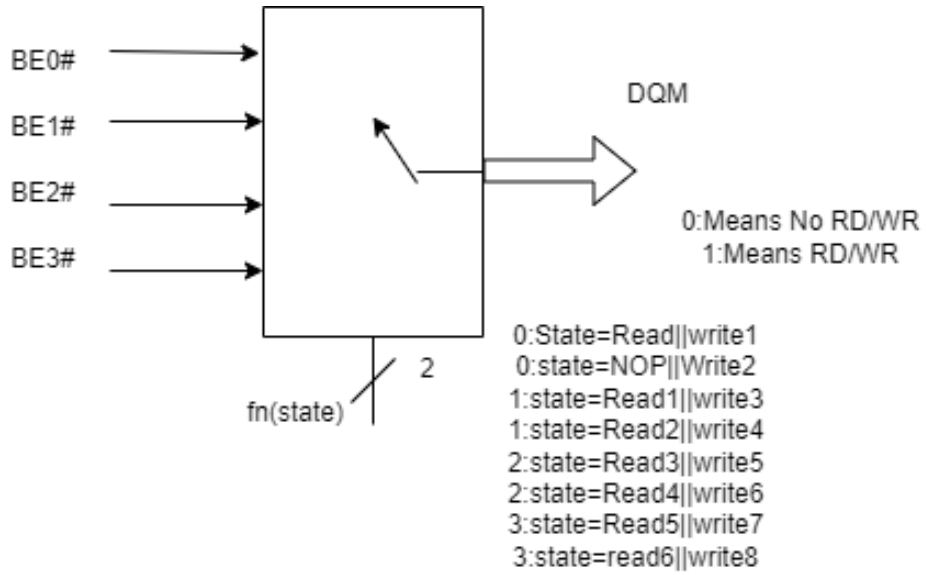


COMMAND	RAS#	CAS#	WE#
LMR	0	0	0
Auto Refresh	0	0	1
precharge	0	1	0
Active	0	1	1
write	1	0	0
Read	1	0	1
NOP	1	1	0

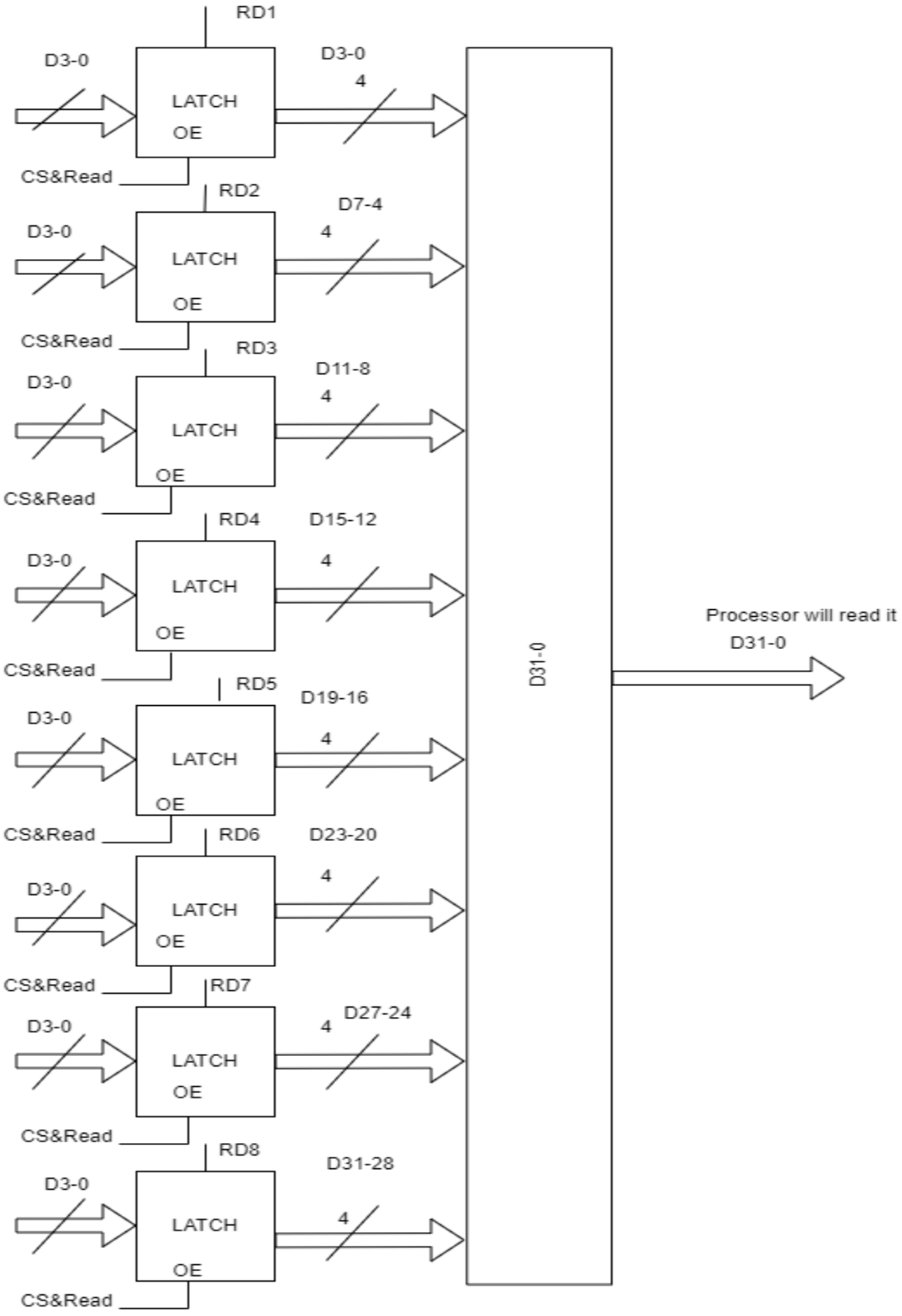
Generation of Bank Address:



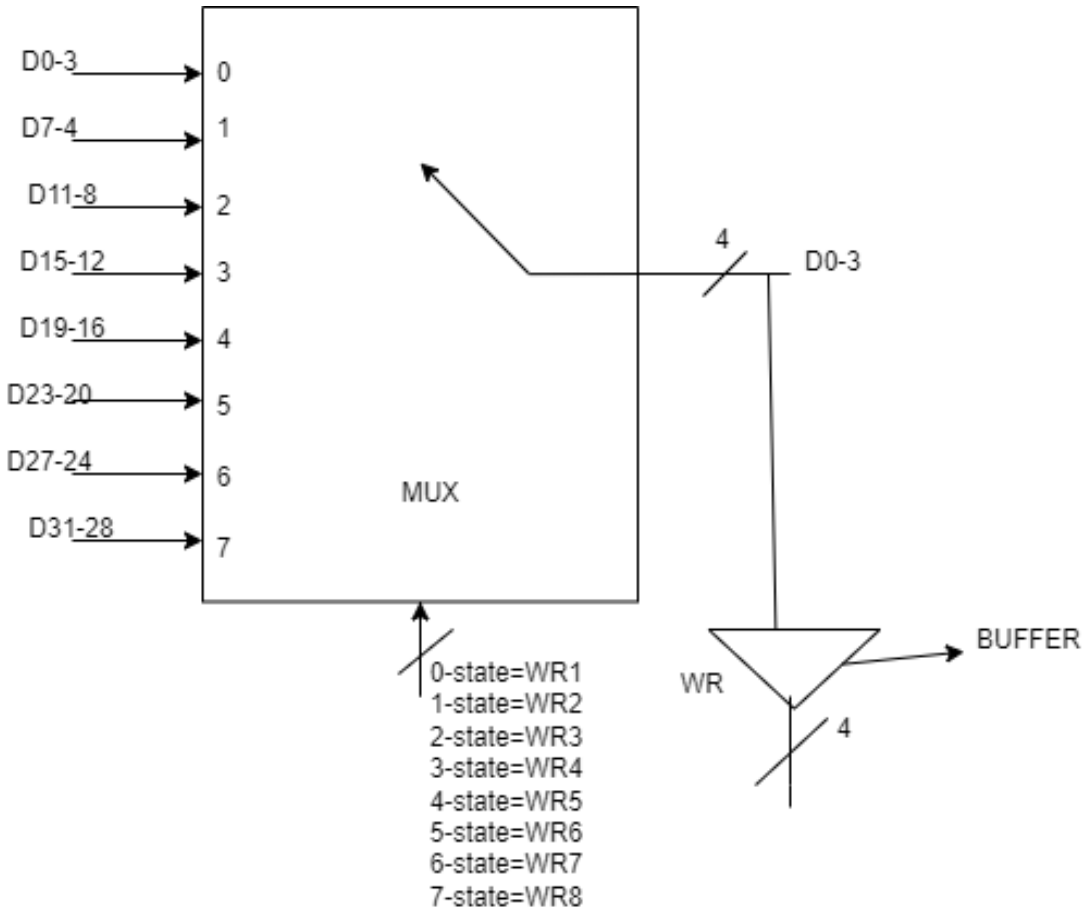
Generation of DQM:



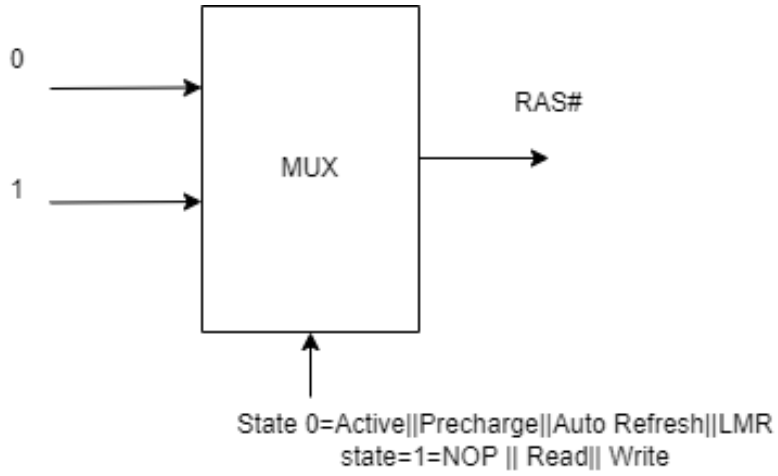
Data Latch for Read:



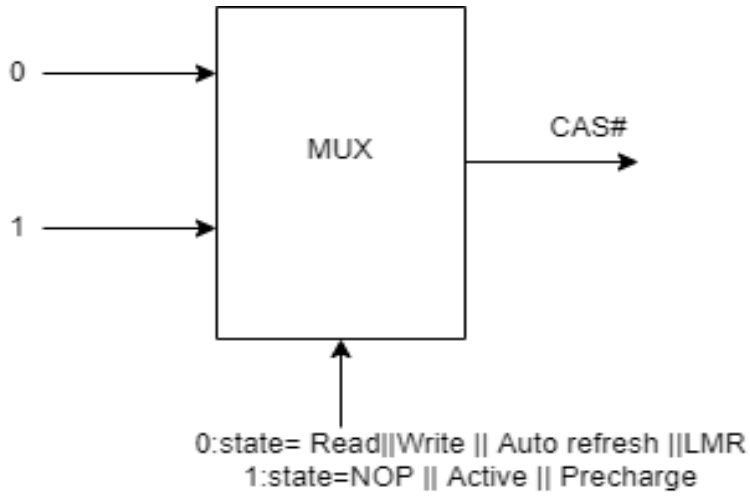
Data Mux for Write:



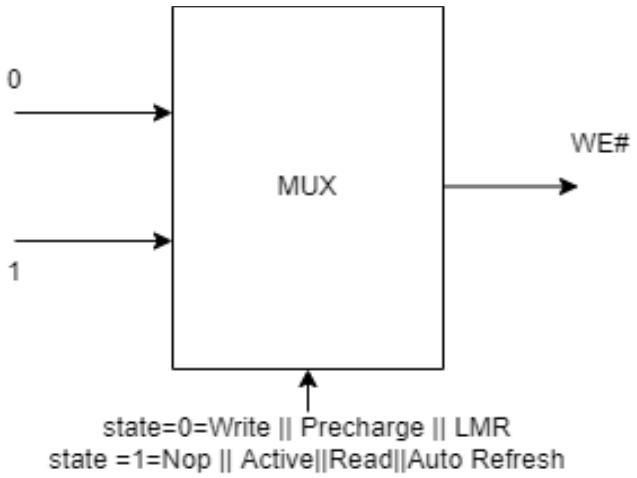
Generation of RAS#:



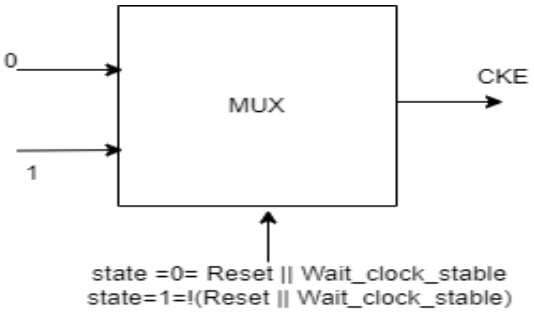
Generation of CAS#:



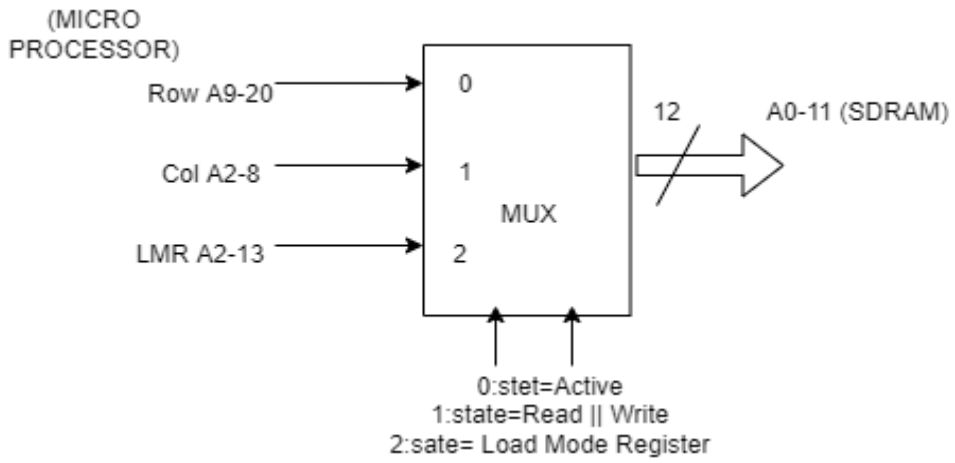
Generation of WE#:



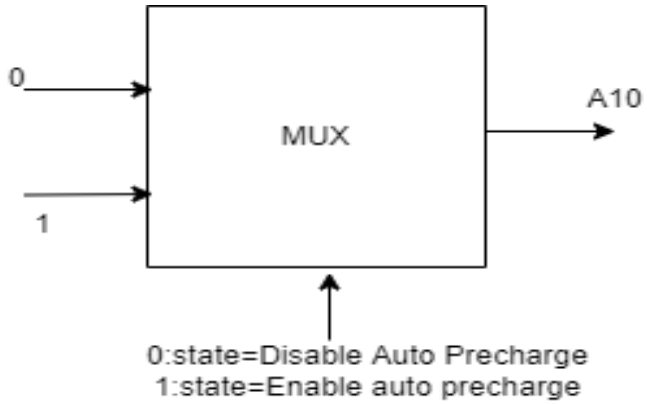
Generation of CKE Signal:



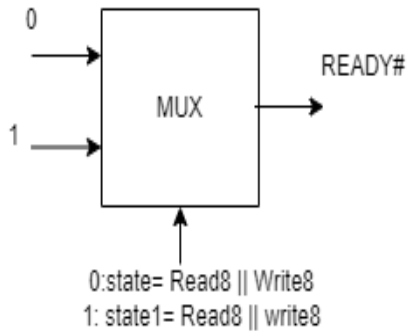
Generation of Row and Column Address:

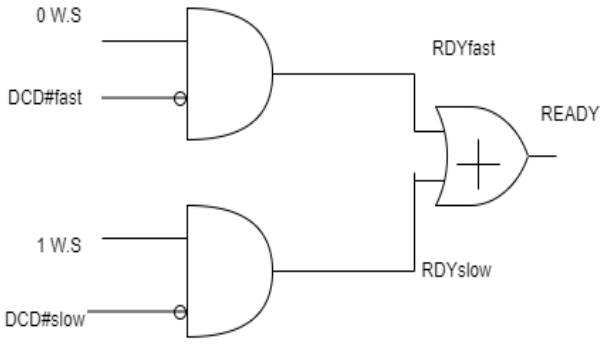


Generation of Auto Precharge(A10):

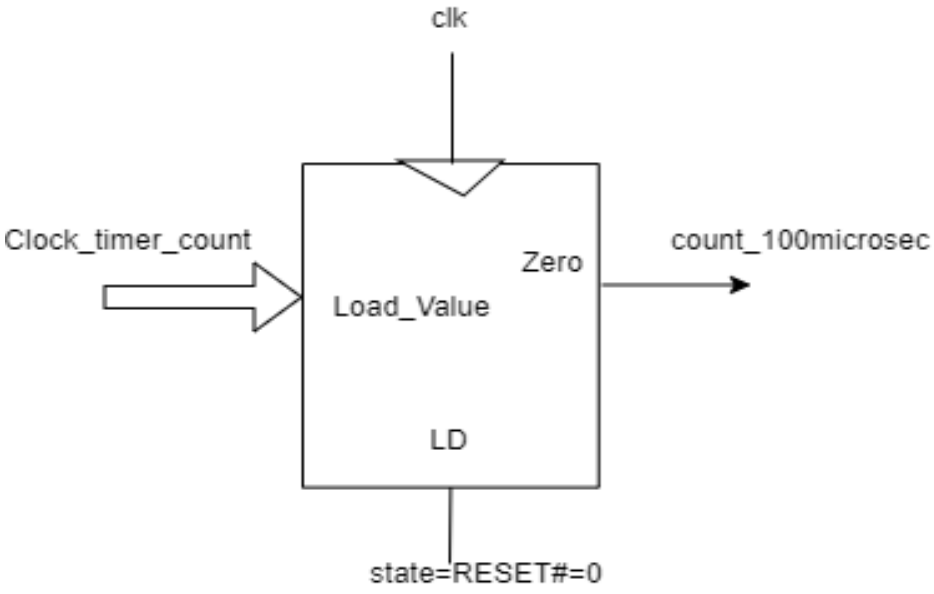


Generation of READY Signal:



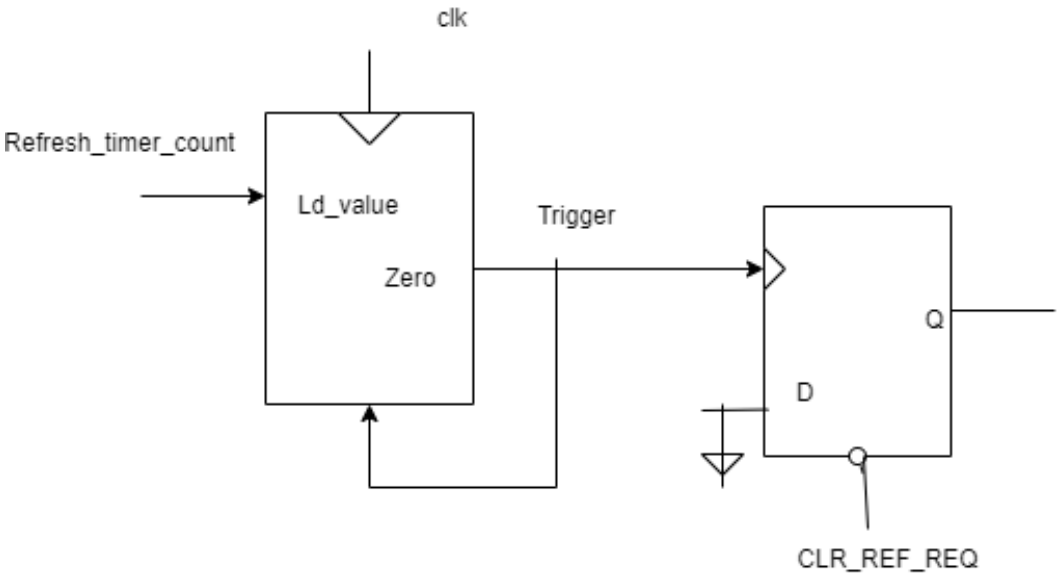


Generation of Clock Timer:



Clock_timer_count=100microsec*132MHZ=13,200counts

Generation of Refresh Timer:



Refresh_timer_count=64ms*132MHZ=8448