EECS 150 Spring 2004

Checkpoint1 Part1

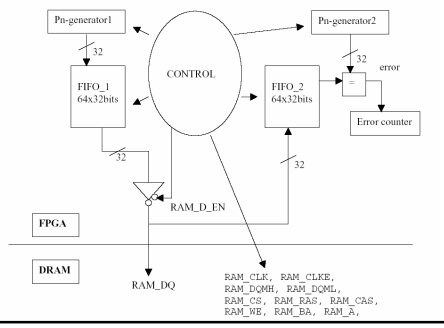
2/27/2003

Greg Gibeling
(Adapted From Sandro Pintz)

Motivation

- Learn to configure external SDRAM
- Write and read from external SDRAM
- To use FIFOs as buffers
- Design a memory controller
  - You will need this for the project
- Analyze retention time of SDRAMs

Block Diagram



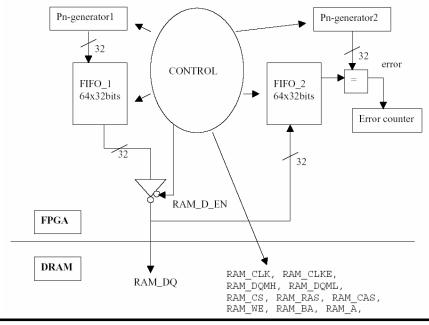
Methodology (1)

- Initialize and configure SDRAM
- Generate 32-bit pseudo random numbers into FIFO1
- After first 4 words start reading FIFO1 and writing to SDRAM and start programmable timer…
- Fill SDRAM and stop write operation

Methodology (2)

- … after programmable delay expires, start reading SDRAM and filling FIFO2
- Use identical pn-generator to compare data from FIFO2 with written data
- Count the errors and display on LEDs
- Writing and Reading SDRAM may overlap!!

Block Diagram



Theory of SDRAM (1)

- SDRAM: Synchronous Dynamic RAM
- Dynamic RAM is large but slow
- Synchronous interface allows more bandwidth
- SDRAM Control can be tricky

Theory of SDRAM (2)

- DRAM is BIG so we time mux address
  - Row Address
  - Column Address
- Steps to Read/Write
  - Send Row Address
  - Send Column Address
  - Send/Get Data

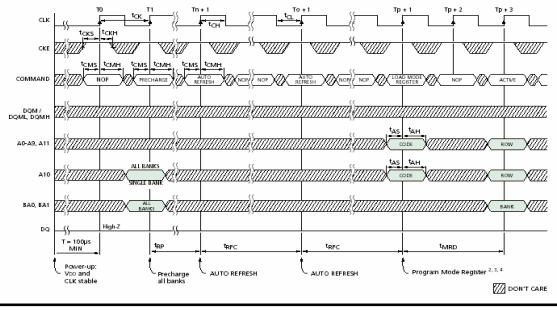
Theory of SDRAM (3)

- SDRAM Steps to Read/Write
  - Send Row Address
  - Send Start Column Address/Send Data
  - Send/Get Data
  - Send/Get Data
  - Send/Get Data
  - Get Data

Theory of SDRAM (4)

- SDRAM is a large FSM
  - Send it a command
  - Get a response
- SDRAM Controller's Job
  - Send the right command signals
  - Ensure command sequences are timed correctly
  - Another large FSM

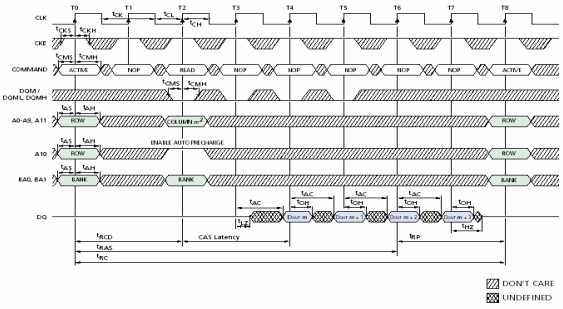
SDRAM Initialization



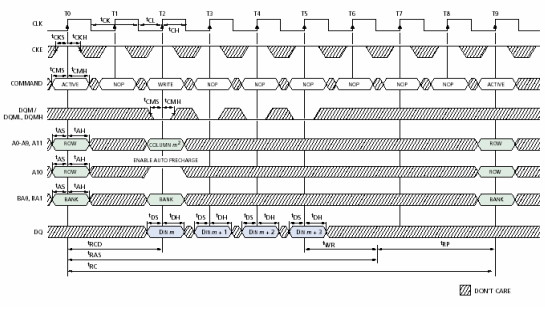
SDRAM Commands

NAME (FUNCTION)	CS#	RAS#	CAS#	WE#	DQM	ADDR	DQs	NOTES
COMMAND INHIBIT (NOP)	H	X	X	X	X	X	X	
NO OPERATION (NOP)	L	H	H	H	X	X	X	
ACTIVE (Select bank and activate row)	L	L	H	H	X	Bank/Row	X	3
READ (Select bank and column, and start READ burst)	L	H	L	H	L/H[8]	Bank/Col	X	4
WRITE (Select bank and column, and start WRITE burst)	L	H	L	L	L/H[8]	Bank/Col	Valid	4
BURST TERMINATE	L	H	H	L	X	X	Active	
PRECHARGE (Deactivate row in bank or banks)	L	L	H	L	X	Code	X	5
AUTO REFRESH or SELF REFRESH (Enter self refresh mode)	L	L	L	H	X	X	X	6, 7
LOAD MODE REGISTER	L	L	L	L	X	Op-Code	X	2
Write Enable/Output Enable	–	–	–	–	L	–	Active	8
Write Inhibit/Output High-Z	–	–	–	–	H	–	High-Z	8

Read Operation



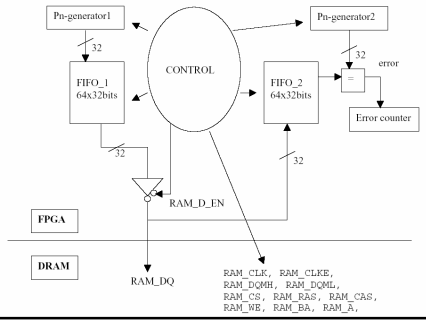
Write Operation



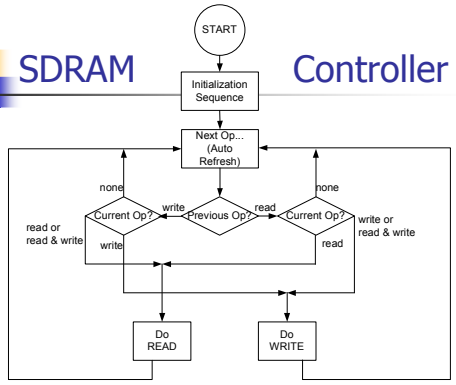
Write Timing

SYMBOL*	-7E MIN	-7E MAX	-75 MIN	-75 MAX	-8E MIN	-8E MAX	UNITS
tCMS	1.5		1.5		2		ns
tDH	0.8		0.8		1		ns
tDS	1.5		1.5		2		ns
tRAS	37	120,000	44	120,000	50	120,000	ns
tRC	60		66		70		ns
tRCD	15		20		20		ns
tRP	15		20		20		ns
tWR	1 CLK + 7ns		1 CLK + 7.5ns		1 CLK + 7ns		–

Block Diagram



SDRAM Controller (1)



SDRAM Controller (2)

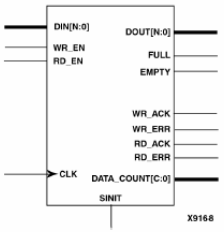
- Design it so that it is good for the project
- Ping-pong between reads and writes
  - Keep track of what the last operation was and try to do the other one
  - The project will have 5R/5W ports
- Tristate data line when reading!!

SDRAM Controller (3)

- Can make one large FSM
  - One state per command perhaps
- Can make two FSMs
  - One has states for each sequence
  - One has a state for each command
- This is totally free-form…
- …just make sure it works

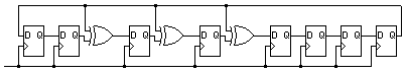
FIFOs

- Buffer to match two data rates
- Great for data path clock domain crossings (we'll talk about it later this semester)



LFSR

- Pseudo Random Sequences
- Signature Generation/Checking
- Built in Self Test (BIST)
- Pattern can be exactly repeated



The Project (1)

- Define your interface clearly
  - SDRAM Top contains this checkpoint
  - SDRAM Control should be reusable
  - Don't rely on special signals
- A good interface is worth a LOT
  - Will make this checkpoint easier
  - Thinking for 20min now will save a week later in the semester

The Project (2)

- A Suggested Interface
  - Write Request/Done
  - Read Request/Done
  - Read (Word Valid)
  - Write (Need a word)
- Just one set of ideas
- We're not explaining
- We want you to invent something

Tips (1)

- Manage your files well
  - Have a backup copy on U:\Checkpoint1\...
  - Work with c:\users\cs150-*\Checkpoint1\...
  - Save only the verilog and blackboxes
  - Create a new Xilinx project every time you start over, they're bit and pointless to save
  - Name your subdirectories well, and use them consistently!
  - Poor file management can cost days of work

Tips (2)

- Draw Diagrams
  - Bubble and Arc for FSMs
  - Block Diagrams for EVERYTHING
- Why?
  - Helps build it faster
  - Reuse it more easily
  - TAs can help you more with diagram than a directory full of verilog

Tips (3)

- Verilog Style!
  - Develop a Style (Or use ours)
  - STICK TO IT (Both Partners)
  - Use reasonable naming schemes
  - Indentation is your friend
  - Look at our verilog! See how pretty it is?

Tips (4)

- Build some basic circuit elements
  - We've given you a bunch
  - What else might you want? (Register, …)
- Registers
  - Don't think of them as memory
  - Think of them as a delay of 1 clock cycle

Tips (5)

- Group similar wires into a single bus
  - assign {wire1, wire2} = bus;
  - Then you can set all the wires with a single assignment
- Name your constants
  - Use "parameter"
  - ALWAYS name your states
  - You can also name other constants
  - (Hint: SDRAM Commands)
  - Don't name "on" and "off" or integers…

Tips (6)

- Manage your files and directories
- Draw diagrams (keep them updated)
- Use good verilog style!
- Build up a library of basic circuits
- Think of registers as a delay of 1 clock
- Group similar wires
- Name your constants

The Checkpoint

- You have two weeks
  - First Week: Simulation
  - Second Week: Demo circuit on board
- START EARLY (Do the prelab)
  - It requires more design work than labs
  - Don't get behind on the checkpoints
  - We're happier helping you early on
- Partner Problems?
  - See Greg after lab lecture
  - Post to the newsgroup