Implementing a Synchronous DRAM Controller in Cypress CPLDs

Abstract

This application note discusses the implementation of a synchronous DRAM (Dynamic Random Access Memory) controller for a Pentium® processor. Today's high-performance CPUs demand high-speed memory. Conventional DRAM technology cannot support the data rates that today's CPUs require. As the bus speed gets faster than 50 MHz, new memory devices are required. Experts believe synchronous DRAM (SDRAM) is the new memory for high-speed CPUs. A simple SDRAM controller can easily fit in a Cypress CPLD (CY7C375i). The circuit in this paper is described in VHDL and uses the Cypress *Warp2*® VHDL compiler. The VHDL code can be obtained from your local Cypress FAE.

Introduction

Of all DRAMs manufactured today, approximately 70% are used in desktop and notebook PCs, where they are used to provide two different functions: main storage and frame buffers. Most PCs are offered with L2 cache to bridge the processor/memory performance gap. This makes the speed of the DRAM memory used for main storage an important but secondary consideration to price. However as multitasking increases with large programs, the frequency of accesses to the L2 cache decreases. This degrades overall system performance, since the processor must wait for the DRAM to supply the requested data. To recover lost system performance, larger L2 cache or faster DRAM main memory are required. Therefore SDRAM is the right main memory choice to increase the system performance.

Some of the advantages of SDRAM are:

- Low-cost.
- Speed up to 100 MHz and beyond.
- State machine based operation (not pulse width driven).

What is SDRAM?

SDRAM is fast DRAM with a high-speed synchronous interface. Since SDRAM I/Os are synchronized with the clock, new options are available for the designer. These new options give designers a great opportunity to achieve high-bandwidth and greatly simplifies timing.

Some of the SDRAM features are:

- **Synchronous control**
  All inputs and outputs are synchronized with the clock.
- **Burst I/O**
  With burst I/O it's possible to give one address and read from or write to multiple (consecutive) addresses.
- **Programmable burst length**
  The programmable burst length allows the designer to use the same SDRAMs on different systems. The burst length can also be changed during normal operation.
- **Two-bank architecture**
  SDRAM allows two row addresses of the DRAM to be opened simultaneously. Memory accesses between two opened banks can be interleaved to hide the row precharge and first access delays. In doing so a seamless data rate of 100 MHz can be achieved to read or write the entire device. With a two-bank architecture it's possible to access one bank and to precharge the other bank.
- **Auto refresh (CBR) and self refresh**
  Auto refresh is referred to as CAS before RAS refresh on FPM and EDO DRAMs. Self refresh allows the device itself to generate the control signals necessary to refresh the storage cells within the allotted retention interval.

Conventional DRAM is controlled asynchronously. This means that the system must insert wait states (latency cycles to allow the DRAM to catch up with the CPU) to meet the specifications of the conventional DRAMs. Timing depends on the speed of DRAM device used, and is independent of the bus speed.

For example, conventional DRAM requires 80 ns total latency for a single read/write. When the DRAM has a burst cycle of 4 (i.e., a CPU access to 4 consecutive locations) the total latency of the DRAM would be 4 * 80 = 320 ns. With an SDRAM device, the total latency for a single read/write cycle is 85 ns (slower than a single access to asynchronous DRAM), but when the SDRAM has a burst cycle of 4, each subsequent word of data can be delivered in a single clock cycle (total latency of 85 ns + 3 x 15 ns = 130 ns)

Performance

With SDRAM, the performance can be increased by up to 2.7 times that of the fast page mode device (conventional, asynchronous) when using the two-bank capabilities.

System Specifications

The SDRAM controller in this application note was designed to work with a Pentium processor with a bus speed of 66 MHz (Pentium 100 or Pentium 133). The SDRAM used in this application note is based on NEC technical data and operates at 66 MHz. Any other memory device with a similar specification would be acceptable (e.g., IBM, Mitsubishi, Texas Instruments, etc.). The proposed system has the following specifications:

- Pentium Processor 100/133 MHz with a bus speed of 66 MHz
- Memory device SDRAM 16 Mbytes - 66 MHz
- Quad word, double word, word, and byte addressing
- Refresh timer
- Single and burst mode

Pentium Processor

Pentium processors have a 64-bit-wide data bus. The data can be accessed in 8, 16, 32 or 64 bits with the Byte Enable signals (BE).

The Pentium has an 8-bit parity bus. The parity bus indicates the parity status of the data bus and is used to detect data errors. When the Pentium processor detects a parity error, the signal $\overline{\text{PCHK}}$ will be activated. The system then needs to generate an interrupt (not implemented in this application note, but may be added to improve data integrity).

The signal $\overline{\text{RW}}$ indicates whether a read or write cycle is taking place. A read is indicated by a low on $\overline{\text{RW}}$.

The memory devices used in this application note are 16-Mbyte SDRAM modules. The address bits [24:3] are used to determine which quad word of the memory is to be accessed. The address bits [2:0] are the byte selects.

The Pentium processor can handle two different length cycles (i.e., single or burst). The difference between a single access and a burst access is determined by the signal $\overline{\text{CACHE}}$. When $\overline{\text{CACHE}}$ is LOW during the correct clock edge of an access, burst cycles are selected. When $\overline{\text{CACHE}}$ is HIGH, single cycles are selected.

Single Transfer

A single transfer will occur any time the Pentium performs a non-cacheable memory access. The Pentium processor initiates an access cycle by activating the address strobe signal $\overline{\text{ADS}}$. The first clock edge in which $\overline{\text{ADS}}$ is activated is, by definition, the first clock in the bus cycle. The SDRAM controller returns $\overline{\text{BRDY}}$ (bus ready) when it has completed a read/write cycle. The Pentium processor drives data parity bits in the same clock cycle and with the same timing as the data (*Figure 1*).

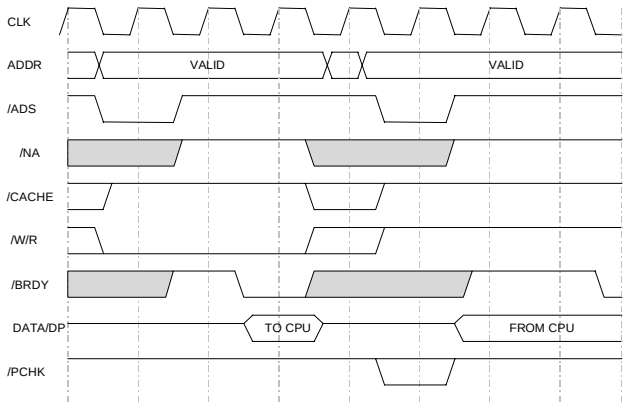


Figure 1. Single Read & Write Access

Burst Transfer

For a bus cycle that requires more than a single data transfer (cacheable cycles) the Pentium uses burst data transfers. Burst cycles are implemented by holding the $\overline{\text{BRDY}}$ pin active for four clock cycles. With conventional DRAM, wait states must be added for reads or writes because the DRAM is not fast enough to keep up with the CPU. Wait states are added by holding $\overline{\text{BRDY}}$ inactive until data is available from the DRAM. With SDRAM it's possible to get data in or out of the memory device on subsequent clock cycles.

For an overview of the Pentium processor's I/O control signals see Appendix 3. I/O From and To the Pentium.

Synchronous DRAM

In a system that uses asynchronous DRAM (i.e., Fast Page DRAM or EDO DRAM), $\overline{\text{RAS}}$ (Row Address Strobe) is used to lock the upper portion of the DRAM address and must be held active during the entire access. Consequently the DRAM cannot accept new addressing and control signaling during this period. This has a negative effect on the overall performance of this system.

The advantage of SDRAM is that all control signals are sampled on the rising edge of the clock. Consequently, control signals like $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ (Column Address Strobe) are only active for one clock cycle (*Figure 2*) and address changes may be overlapped with the previous cycle.

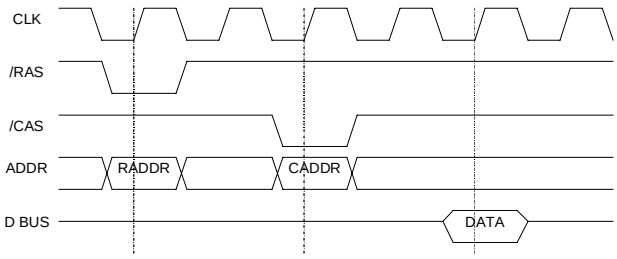


Figure 2. RAS/CAS Timing

Another advantage of SDRAM is that it can accept a new $\overline{\text{CAS}}$ every clock cycle. There is also something call $\overline{\text{CAS}}$ Latency, which is defined in a later section. The first time the processor accesses the memory, it takes a few clock cycles before the data is available. During this timeframe, the processor can perform other activities, which could not be done with the normal DRAM (as certain bus signals are held constant with asynchronous DRAMs). After this time period, the processor can collect new data by a burst read every clock cycle.

Parity

The memory device is 9 bits wide (8 bits for data and one for parity). The most significant bit (msb) is the parity bit. Parity is used to detect data errors due to transmission problems or memory corruption.

Two-Bank Architecture

The internal memory array of an SDRAM consists of two separate memory banks. Which bank is selected depends on the highest order address bit. In this application note, it is the A11 (BS: Bank Select) address line. If A11 is activated, Bank B is selected otherwise the access is to Bank A. If one bank is selected, the other bank can be precharged at the same time.

Programmability of the Burst Length

Different burst lengths can be programmed in the SDRAM. The SDRAM has an internal counter which is used to generate the addresses for burst cycles. Five burst lengths are supported: 1, 2, 4, 8 and full-page (512). The processor generates the first address, the following addresses are generated by the burst counter in the SDRAM. These values are pro-

grammed in the Mode Set Register of the SDRAM. For further information see the section “Programming the Mode Register”. In a Pentium, only two burst lengths are used (1 for single transfers, and 4 for bursts). The CAS latency is changed based upon bus loading and clock speed. For a lightly loaded system, a CAS latency of 1 will yield the highest performance.

CAS Latency

The CAS latency is used to define the delay between the read command and the clock cycle where the data is available at the outputs. The latency of it is expressed in clock cycles 1, 2 or 3. These values are programmed in the Mode Set Register of the SDRAM. For further information, see “Programming the Mode Register”.

Pre-Charge Command

DRAM reads are destructive. A read will cause the stored value to discharge. Pre-charge rewrites the value after a read. The pre-charge command is used to pre-charge a bank that is going to be activated. The pre-charge command is triggered when $\overline{CS}$, $\overline{RAS}$, and $\overline{WE}$ are activated and $\overline{CAS}$ is deactivated at the rising edge of the clock. Two address bits are used to define which bank will be precharged (A10 & A11).

Table 1. Address Bits Pre-Charge

A10	A11	Precharge bank(s)
LOW	LOW	Bank A
LOW	HIGH	Bank B
HIGH	Don't Care	Bank A & B

If one bank is accessed, the other bank can be pre-charged at the same time. This is useful for the Pentium processor because it can read/write data in an “interleaved” fashion. This means that the Pentium has a burst order of 1 3 2 4 (odd, odd, even, even). The advantage of this interleave is that when bank A is accessed, bank B is being pre-charged. This is faster than sequential, because a bank must to be pre-charged after it is accessed. In interleaved operation, the pre-charge may be overlapped with the access to the other bank. The other mode is sequential, which is a burst order of 1 2 3 4 (odd, even, odd, even).

Refresh Command

DRAM technology uses capacitors to store the data. Over time, the capacitors will lose their charge. Therefore, the DRAM must periodically rewrite (refresh) the data, which restores the capacitors’ charge. The SDRAM has an ‘automatic refresh’ command and a ‘self refresh’ command. In the proposed design the automatic refresh command is used. The refresh command is controlled by a counter for controlling the time when the SDRAM has to be refreshed (every 4096 cycles).

Programming the Mode Register

The CAS latency, burst length, and burst sequence are user-defined variables and can be programmed into the SDRAM Mode Register with a single Mode Register Set command. All variables can be updated by re-executing the Mode Register Set command. If the user chooses to modify only a subset of the Mode Register variables, all of the variables have to be redefined when the Mode Register Set command is issued. To program the MR the following signals have to be activated: $\overline{CS}$, $\overline{RAS}$, $\overline{CAS}$, and $\overline{WE}$. When these signals are active, at the next rising edge of the clock, the op-code at (A0...A11) programs the MRS.

For an overview of all the inputs and outputs of the SDRAM circuit see Appendix 4. I/O to the SDRAM Circuits.

Options of the SDRAM

The options that the SDRAM has to offer exceed the requirements of the Pentium processor.

The features of the SDRAM are:

- Fully synchronous
- Dual banks
- Programmable $\overline{CAS}$ latency: 1, 2, 3 clock cycles
- Programmable burst length: 1, 2, 4, 8, full-page (512)
- Programmable wrap sequence: sequential or interleave
- Multiple burst read with single write option
- Automatic and controlled precharge command
- Data mask for read-write control
- Auto refresh and self-refresh
- Suspend mode and power down mode
- 4096 refresh cycles
- Random column address every CLK (1-N rule)

Note that not all of these features will be covered in this paper because the Pentium processor cannot take advantage of them all. However, if the designer is implementing a non-Pentium system, these features may be useful.

Top-Level Design

The architecture of the design is shown in Appendix 1. Memory Architecture.

The circuit is split into seven blocks (*Figure 3*). Each block has it’s own functionality. The *ADDRESS DECODER* (block U1) determines if the memory system is accessed. The address signals do not have to be latched, because they are kept valid during the entire access cycle. The System *CONTROL* block determines the action that has to be taken when the SDRAM controller gets a match (adr_dec) from the Pentium. With the “command” bus the controller tells the *COMMAND DECODER* block and the *CHIP SELECTOR* block what kind of action has to be performed. Byte enables are allowed to change during the cycle. *BYTE ENABLE LATCH* is a block that stores the ‘byte enables’ to keep them valid during the process. The *OUTPUT LATCH* block has the same function as the *BYTE ENABLE LATCH* block, but in this device the outputs are kept valid. The last block is the *REFRESH COUNTER*, which gives a refresh every 64 ms.

Each block is described in more detail in the following sections.

Note: The pin diagram is shown in Appendix 2.

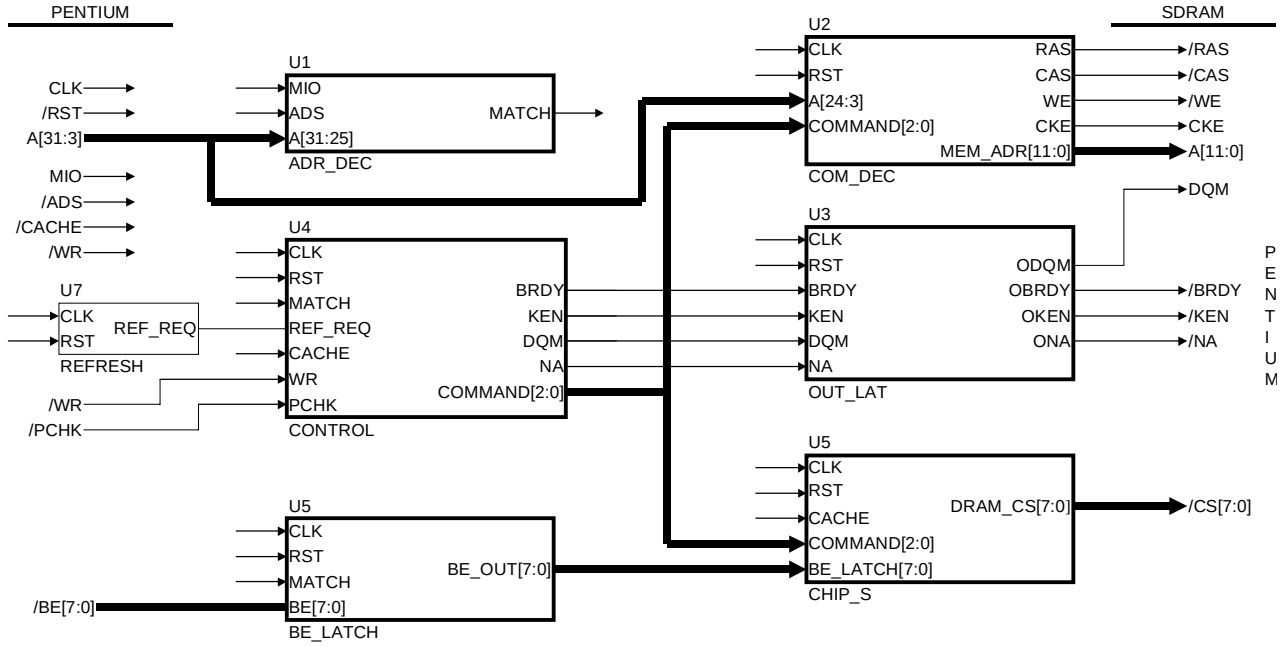


Figure 3. Top-level Design

Controller

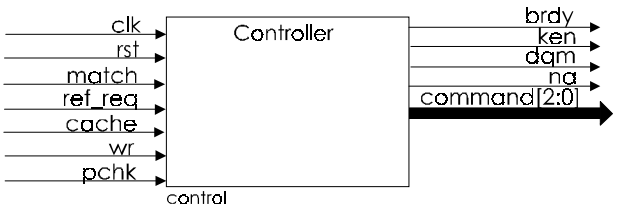


Figure 4. Control.vhd

The Controller block manages all other devices. When the inputs change, the controller determines what action should be taken. This flow of actions is drawn in a simplified state in *Figure 5*.

If the Pentium processor generates a single read (or any command), the controller will detect this and generate the appropriate commands for the SDRAM. When the SDRAM is ready to present the data, the controller notifies the Pentium by activating the $\overline{\text{BRDY}}$ signal.

Every 4096 cycles the SDRAM needs to be refreshed. This is done by the refresh command. When the refresh timer activates the refresh request signal, this signal will be stored until the controller is in the IDLE State. When the controller is in IDLE and detects a refresh request, the command “refresh” will be generated and the stored signal will be reset.

Control Final State Machine (simplified)

The following describes each state of the state machine:

INITIALIZATION: The device enters this state after a global reset is given to the system. In this state the SDRAM will be precharged, refreshed and set for a burst length of 1. The final state machine always returns to the IDLE state.

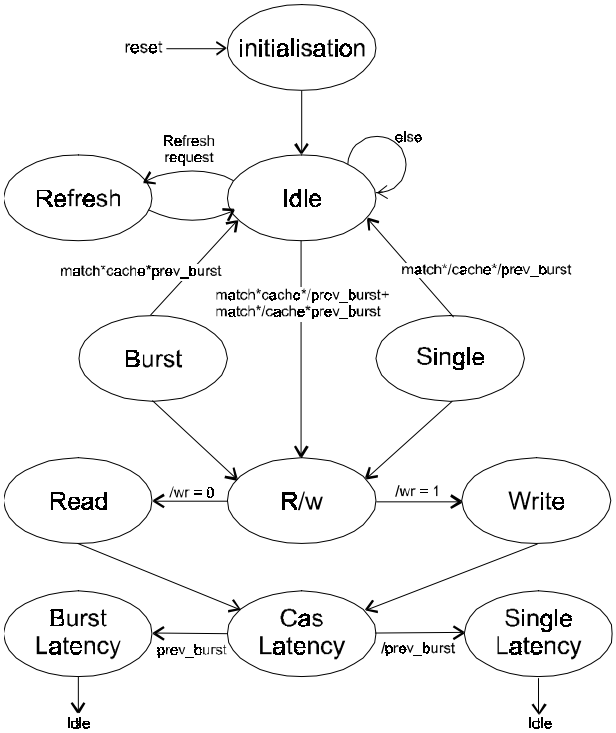


Figure 5. Simplified Control FSM

IDLE: In this state the system will wait until a memory access is initiated. No command is given to the SDRAM. If there is a matched R/W, the FSM moves to the matching state. If there is a refresh request, the next state will be REFRESH.

REFRESH: The system will refresh the SDRAM in this state, and then returns to the IDLE state.

BURST: In this state the SDRAM Mode Register will be set to a burst length of 4, if the previous burst length was 1. The next state will always be R/W.

SINGLE: In this state the SDRAM Mode Register will be set to a burst length of 1, if the previous burst length was 4. The next state will always be R/W.

R/W: n this state the RAS command (Bank Active) must be sent to the SDRAM for the appropriate burst length. Depending on the input $\overline{RW}$, the FSM will move to the READ or WRITE state.

READ: In this state the CAS command will be sent to the SDRAM. From here, CAS LATENCY is the next state.

WRITE: In this state the CAS command will be sent to the SDRAM. From here, CAS LATENCY is the next state.

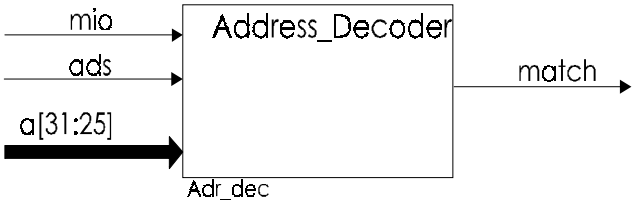
CAS LATENCY: Depending on the burst mode, the CAS latency will be 2 or 4, so the appropriate state will be the next. 2 is the minimum CAS latency for 66-MHz SDRAM, specified in the data sheets.

BURST LATENCY: This state will activate $\overline{BRDY}$ until the last bytes, of a burst, are written to or read from the SDRAM. After this action is complete, the FSM will return to its IDLE State.

SINGLE LATENCY: This state will activate $\overline{BRDY}$ until the data is written to, or read from the SDRAM. After this action is done, the FSM will return in its IDLE State.

The functional description of the Control FSM is described in VHDL. This VHDL code can be obtained from your local Cypress FAE.

Address Decoder



The address decoder determines if the Pentium processor is addressing the memory. The Pentium processor can perform two different access types, memory or I/O. The Pentium indicates the difference with the signal $\overline{M}$/IO. The SDRAM controller only responds to memory accesses.

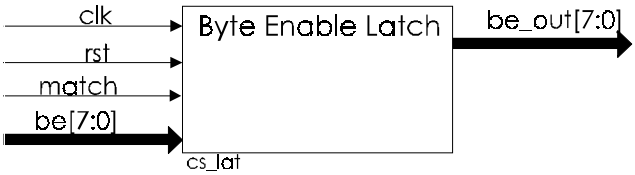
With the signal $\overline{ADS}$ the Pentium indicates that a new valid bus cycle is currently being started. In this implementation the lowest 16 MB of the processor's address space is used for the SDRAM. When one or more of bits A[31:25] are activated, the selected address is above the 16 Mbytes address space and is ignored by the SDRAM controller.

$2^{(25-1)}$ = 16.384 KB= 16 MB

If the status of $\overline{M}$/IO, $\overline{ADS}$ and A[31:25] indicates an SDRAM memory access, a match is generated.

The functional description of the address decoder is described in VHDL. This code can be obtained from your local Cypress FAE.

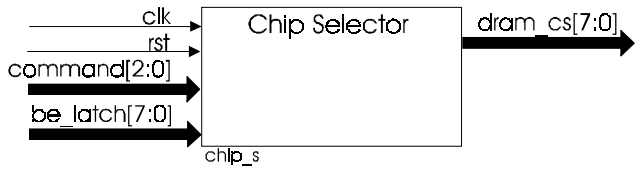
Byte Enable Latch



The Byte Enable Latch is responsible for storing the BE[7:0] (Byte Enable) signals from the Pentium processor. It makes sure that the BE[7:0] values of the Pentium processor are held during the entire access. The signal MATCH is the trigger for storing new 'Byte Enables'.

The functional description of the address latch is described in VHDL. This code can be obtained from your local Cypress FAE.

Chip Selector



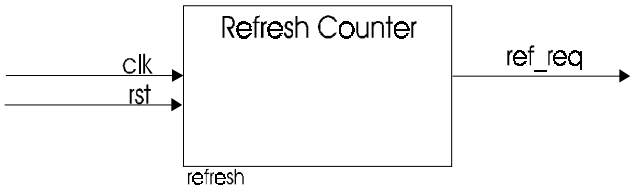
The Chip Selector is responsible for selecting the right chip(s) of the SDRAM. The SDRAM consists of 8 2-Mbyte chips which together have a total range of 16 Mbytes. Each chip has 8 databits and one bit for parity. When the Pentium processor wants to read/write a byte, word or longword, the right chip(s) must be selected using the 'Byte Enables'. If the Pentium processor wants to read/write a burst, all the chips of the SDRAM have to be selected, because a burst is written or read using all 64 databits. The Chip Selector uses the Byte Enable signals and a few control signals for selecting the correct SDRAM device. *Table 2* is the truth table for the SDRAM Chip Select decoding.

Table 2. CS Decoding

Cache	command[2:0]	be_latch[7:0]	dram_cs[7:0]
x	000	x	11111111
x	001	x	00000000
x	010	x	00000000
x	011	x	00000000
x	100	x	00000000
0	101	data	00000000
0	110	data	00000000
0	111	data	00000000
1	101	data	be_latch
1	110	data	be_latch
1	111	data	be_latch

The functional description of the Chip Selector is described with VHDL. This code can be obtained from your local Cypress FAE.

Refresh Counter



The SDRAM has an automatic and self refresh command. In this design the automatic refresh command is used to simplify the overall design.

The SDRAM circuit has to be refreshed completely within 64 ms. This means that a refresh command has to be generated every 4096 cycles. An independent 12-bit counter is used to request refresh. When the counter reaches 4096, a REFRESH REQUEST is generated.

The functional description of the Refresh Counter is described in VHDL. This code can be obtained from your local Cypress FAE.

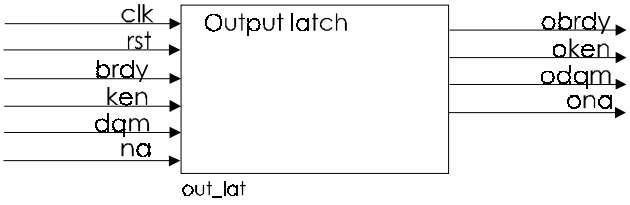
Command Decoder

The Command Decoder is responsible for generating the commands to the SDRAM. These commands are displayed in Appendix 5. Encoding of the Command Bits.

The outputs of the Command Decoder are registered to improve the timing. If they were combinatorial, the “clock to output delay” would be 16 ns. When registered outputs are used, this delay is 7 ns. Note that this also requires an additional clock cycle. The outputs of the chip select decoder are also registered for the same reason.

The functional description of the COMMAND DECODER is described in VHDL. This code can be obtained from your local Cypress FAE.

Output Latch



Due to the problem described above (clock to output delay), all outputs are registered. This includes the Mealy outputs from the FSM Control.

The functional description of the OUTPUT LATCH is described in VHDL. This code can be obtained from your local Cypress FAE.

Simulation

Simulations have been done with the ViewSim™ simulator from Viewlogic.

Conclusions

This paper describes one of many possible implementations of an SDRAM controller using a CYPRESS CPLD device. The design may be modified for many similar Pentium applications. It can also be modified to add new features and options (such as more memory, pipelining and Multi processor architectures).

References

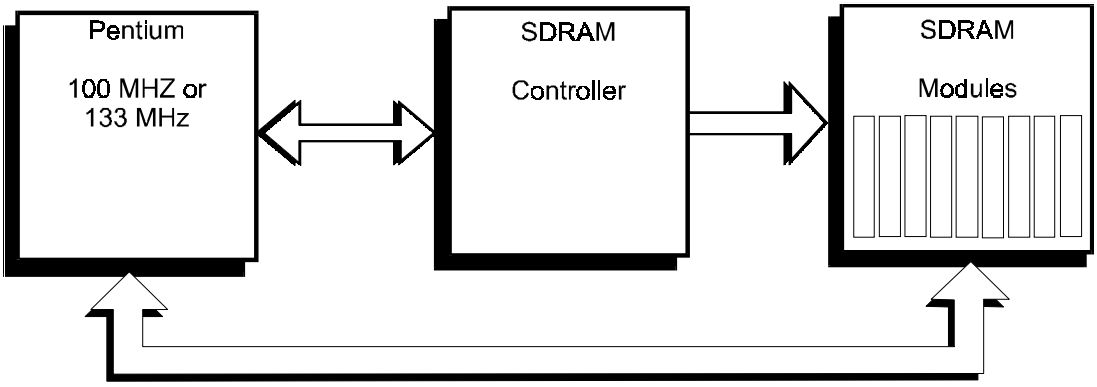
1. Intel, Pentium processor specification’s chapter 5 and 6.
2. IBM, 16 MB Synchronous DRAM Datasheets.
3. *VHDL for Programmable Logic*, Kevin Skahill, Cypress Semiconductor.
4. *Cypress Applications Handbook*, Cypress Semiconductor.
5. *Cypress Programmable Logic Data Book*, Cypress Semiconductor.

Appendix List

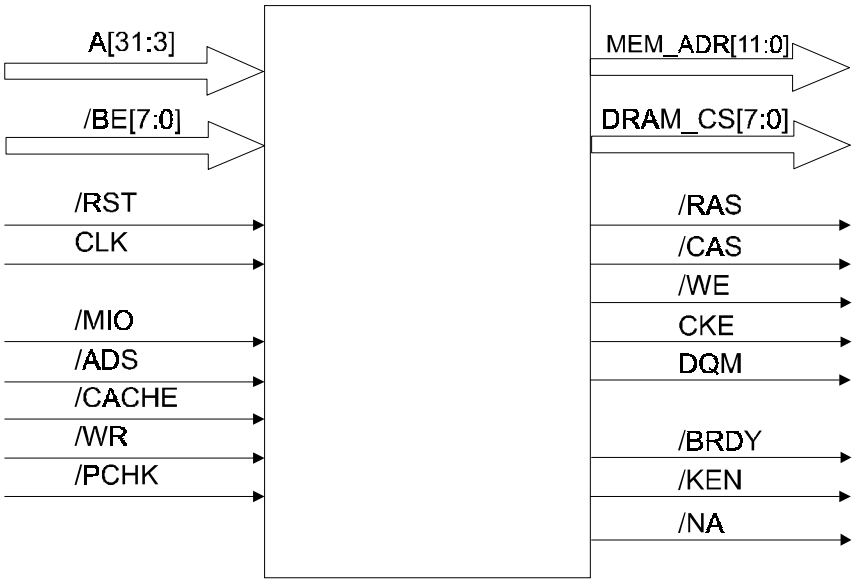
Table 3. List of Appendices

App. 1	Memory Architecture Block
App. 2	Pin Diagram
App. 3	I/O From and To the Pentium
App. 4	I/O to the SDRAM Circuits
App. 5	Encoding of the Command Bits

Appendix 1. Memory Architecture



Appendix 2. Pin Diagram



Appendix 3. I/O From and To the Pentium

Pin Name	Type	Description
A[31:3]	in	29-bit bidirectional address bus
$\overline{BE}$[7:0]	in	Byte enables for the 64-bit Data bus
$\overline{M}$/IO	in	Distinguishes a memory access from an I/O access
$\overline{ADS}$	in	Indicates that a new bus cycle is currently being driven by the Pentium processor
$\overline{CACHE}$	in	External indication of internal cacheability
$\overline{WR}$	in	Distinguishes a read cycle from a write cycle
$\overline{PCHK}$	in	Indicates the result of a parity check on a data read
$\overline{BRDY}$	out	Transfer complete indication
$\overline{KEN}$	out	Indicates whether or not the system can support a cache line fill
$\overline{NA}$	out	Indicates that the external memory is prepared for a pipelined cycle
$\overline{RST}$	in	Reset
CLK	in	Clock

Appendix 4. I/O to the SDRAM Circuits

Pin Name	Type	Description
Mem_adr[11:0]	out	Address output • Row address A0–A10 • Column address A0–A8 • Back select address A11
$\overline{RAS}$	out	Row address strobe command
$\overline{CAS}$	out	Column address strobe command
$\overline{WE}$	out	Write enable command
CKE	out	Clock enable
DQM	out	Input/Output mask
Dram_cs[7:0]	out	Chip selects

Appendix 5. Encoding of the Command Bits

Function	Commando	CKE	$\overline{CS}$	$\overline{RAS}$	$\overline{CAS}$	$\overline{WE}$	A11...A0
Nop	000	1	1	1	1	1	111111111111
MSR (bl=1)	001	1	0	0	0	0	010000101000
MSR (bl=4)	011	1	0	0	0	0	010000101010
Auto refresh	010	1	0	0	0	1	xxxxxxxxxxx
Precharge	100	1	0	0	1	0	x1xxxxxxxxx
Bank active	101	1	0	0	1	1	a[24:13]
AWRITE	110	1	0	1	0	0	a24,1,a[12:3]
AREAD	111	1	0	1	0	1	a24,1,a[12:3]

Warp2 is a registered trademark of Cypress Semiconductor Corporation.
Pentium is a registered trademark of Intel Corporation.