Dynamic RAM

Larger microcomputer systems use Dynamic RAM (DRAM) rather than Static RAM (SRAM) because of its lower cost per bit. DRAMs require more complex interface circuitry because of their multiplexed address buses and because of the need to refresh each memory cell periodically.

After this lecture you should be able to: (1) describe basic DRAM structure and terminology, (2) interface DRAMs to a CPU bus by multiplexing row and column address lines and forcing refresh cycles, and (3) justify the choice of DRAM or SRAM for a particular application.

DRAM Structure

A description of the structure of a DRAM helps explain some of the unique features of DRAMs.

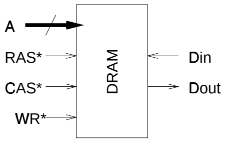
A typical DRAM memory is laid out as a square array of memory cells with an equal number of rows and columns. Each memory cell stores one bit. The bits are addressed by using half of the bits (the most significant half) to select a row (the *row address*) and the other half to select a column (the *column address*).

Exercise: How many rows and columns would there be in the memory cell array in a 16 M by 1 (16 Mbit) DRAM?

Address Multiplexing

In order to reduce the number of pins on the DRAM chip and thus reduce the size of the DRAM chip, most DRAM chips multiplex the row and column addresses onto the same set of pins.

Two strobes, RAS* (row address strobe) and CAS* (column address strobe) tell the chip which half of the address (row or column) is currently on the address pins. To further reduce the chip count the CAS* signal typically acts as an output enable when R/W* line is high (read) and the falling edge acts as a write strobe when R/W* is low (write).

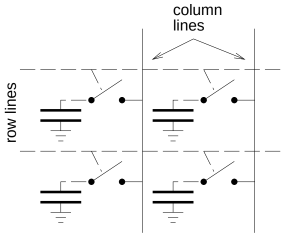


Exercise: How many address pins would be found on a typical 16 M x 1 DRAM?

DRAM Access and Refresh

To maximize the capacity of the DRAM, each memory cell is very simple – it consists of a capacitor and a FET switch. A DRAM memory cell is therefore much smaller than an SRAM cell which needs at least two gates to implement a flip-flop.

The following diagram shows the structure of DRAM array using switches in place of transistors:

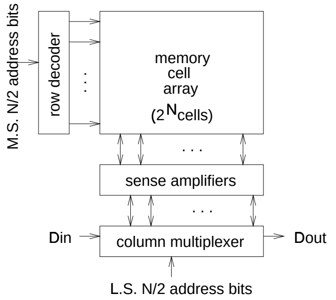


The *row* address drives a decoder which enables the appropriate *row-select* signal. This row-select line turns on all of the FET switches in that row and connects each the capacitors in the selected row to its *column* line. Note that each vertical column line is shared by all the memory cells in a column although only one capacitor is connected to the column at any time.

The charge stored in each memory cell capacitor is relatively small so each column line is connected to a "sense amplifier" which amplifies the voltage present on the line while RAS* is asserted.

When CAS* is asserted the output of the sense amplifiers is driven back onto the column lines and recharges the capacitors. Thus each memory access *refreshes* the contents of that row.

The column address also drives a multiplexer which selects one of the column lines and connects it to the output, thus reading out the value of a single bit. During a write, the value on the input over-rides the sense amplifier value for the addressed column and this stores new data into the desired memory cell.



DRAM Timing

In addition to the DRAM timing requirements of setup and hold times for the row and column addresses, DRAMs also require a minimum “pre-charge” time between the end of RAS* or CAS* and the start of the next cycle. This “recovery” time is needed to re-charge the storage capacitors. This pre-charge time extends the minimum cycle duration to considerably more than the access time. For example, a DRAM with a 60 ns access time may have a minimum cycle time of 100 ns.

DRAM Timing

Figure shows timing diagrams and specifications for a typical DRAM. The sequence of operations required to read or write from a DRAM both start in the same way:

- set R/W* to the appropriate value and place the row address (the MS half of the address) on the address pins,
- wait the RAS* setup time, bring RAS* low, and wait for the RAS* hold time

In order to read from the DRAM:

- place the column address on the address pins, wait the CAS* setup time, bring CAS* low, and wait for the CAS* hold time
- wait until the access times from both CAS* and RAS* are met and then read the data from the data out pin

In order to write to a DRAM the sequence is similar except that during a write cycle the data is latched on the falling edge of CAS*:

- place the column address on the address pins, data on the data input pin, wait for the CAS* and data setup times, bring CAS* low, wait for the CAS* and data hold times.

At the end of either cycle we must then bring RAS* and CAS* high and wait the pre-charge (recovery) time before starting another cycle.

Exercise: Draw a timing diagram for a DRAM read cycle showing the address lines, RAS*, CAS*, WR* and the data pins. Show on the timing diagram the following specifications: address setup and hold times from RAS* and CAS* active, access times from RAS* and CAS* active, minimum “pre-charge” times (from RAS* or CAS* inactive), and the minimum cycle time (from RAS* to RAS*).

For a write cycle, show the setup and hold times for D_{in} from CAS* active.

Refresh

Since the DRAM storage capacitor discharges over time it must be refreshed periodically. The DRAM’s structure ensures that all the memory cells in a row are refreshed every time that row is read. Therefore it is only necessary to periodically cycle through all of the *row* addresses to refresh all of the bits in the memory array.

The simplest technique to provide DRAM refresh is to include a device (such as a DMA controller or video display circuit) that accesses the RAM in such a way that all of the rows are accessed at least once during the minimum refresh time (typically every few tens of milliseconds). This is called *RAS*-only refresh* because it’s not necessary to assert CAS* in order for the refresh operation to take place.

Another technique is to add a circuit that periodically forces a cycle in which CAS* is asserted before RAS*. This is called *CAS* before RAS* (CBR) refresh*. Modern DRAMs have an an internal refresh counter that cycles through the possible row values. On these DRAMs the CAS* before RAS* operation causes an internal row-refresh operation. The advantage of this type of refresh is that the refresh controller need only control RAS* and CAS*, it need not generate the refresh row addresses.

Exercise: Assume a microprocessor with a 200 ns memory cycle time is using a 1 MByte DRAM with a maximum refresh time of 10 ms. How many row addresses will have to be refreshed every 10ms? What is the approximate time between each refresh cycle? How many memory cycles are there per refresh cycle? What percentage of the memory accesses are “wasted” on refresh cycles?

DRAM versus SRAM

Since the SRAM devices require more circuitry per memory element they are more expensive (per bit) to produce and have lower density per chip. The typical ratio between DRAM and SRAM for the same size chip is about 4 to 1.

The disadvantages of DRAMs are that they require additional control circuits to multiplex address lines and to handle refresh. If DRAMs are organized as bit-wide devices it is necessary to use a number of devices that is a multiple of the the data bus width (8, 16 or 32) in a system.

The use of large DRAM arrays in which the CPU address and data buses must drive many chips usually requires buffers for the address and data lines. Because of these reasons DRAMs are mainly used in systems that require large memories and SRAMs are mainly used in smaller systems.

The fastest RAM designs are static, so SRAMs are often used for high-speed memories such as cache or address translation tables.

CMOS SRAMs consume very little power when not being accessed so they are often used in low-power designs. With the use of battery (or a large capacitor) backup they can retain their contents for months. On the other hand, since DRAMs must be continuously refreshed, their power consumption cannot be reduced to very low values.

Due to the larger number of bits per chip and wider bus sizes, DRAMs are now being offered in nybble (4-bit) and larger organizations. The number of extra pins required to provide the additional data bits is less of a concern with modern high density packages such as QFP and SOIC.

Exercise: Consider a system using 16 Mbit X1 memories to design a memory array for a microprocessor system with a 32-bit data bus. What is the minimum amount of memory that could be provided using these devices?

EDO DRAM

Many DRAMS include an extended data out (EDO) feature, in which the output data is latched and held past the end of a read cycle and into the start of the next cycle in order to help satisfy CPU hold times.

Fast Page Mode and Synchronous DRAM

Due to the need to charge the DRAM capacitors, the shortest practical cycle times are currently about 50 ns. If the CPU cycle time is less than this then wait states must be inserted for each memory access. This would limit the performance of processors with cycles times less than about 50 ns.

Instead of accessing the DRAM memory directly, these faster computers use fast SRAM memories that record (“cache”) values as they are read from the main memory. When the CPU accesses a value that is already in the cache it can retrieve it much faster than it could from DRAM.

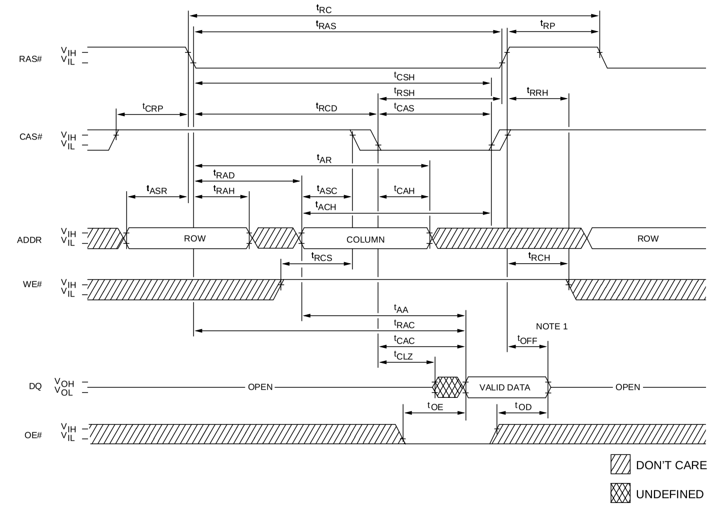
Instead of retrieving a single value from the main memory, cache controllers are designed to retrieve the values from several (e.g. 4) consecutive memory locations in a burst. FPM and SDRAM memories provide fast access to consecutive memory locations and minimize the total time required for a burst read.

Since the contents of each row are read with each RAS* operation, it is possible to obtain the values of more than one column within a given row without having to re-read the row. Some DRAMs support such a “fast page mode” in which multiple CAS* cycles may be used to access several addresses within one row (“page”) after one RAS* cycle. These accesses within one row are much faster than accesses to separate rows (e.g. 10 ns versus 50 ns).

It is also possible to design DRAMs to use multiple internal memory arrays which are read in parallel. Each bank is assigned to consecutive memory locations and, in the same was as FPM, this allows faster access to consecutive memory locations.

High-speed DRAMs use a synchronous (clocked) interface. The cache/DRAM controller writes the desired staring memory location and word count into registers in the DRAM, waits a fixed number of clock cycles, and then reads one word per clock cycle.

READ CYCLE



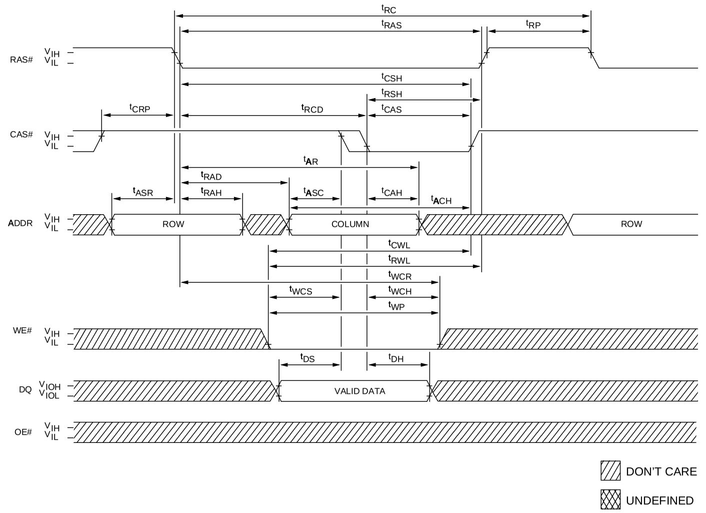
TIMING PARAMETERS

SYMBOL	-5 MIN	-5 MAX	-6 MIN	-6 MAX	UNITS
^{t}AA		25		30	ns
^{t}ACH	12		15		ns
^{t}AR	38		45		ns
^{t}ASC	0		0		ns
^{t}ASR	0		0		ns
^{t}CAC		13		15	ns
^{t}CAH	8		10		ns
^{t}CAS	8	10,000	10	10,000	ns
^{t}CLZ	0		0		ns
^{t}CRP	5		5		ns
^{t}CSH	38		45		ns
^{t}OD	0	12	0	15	ns
^{t}OE		12		15	ns

SYMBOL	-5 MIN	-5 MAX	-6 MIN	-6 MAX	UNITS
^{t}OFF	0	12	0	15	ns
^{t}RAC		50		60	ns
^{t}RAD	9		12		ns
^{t}RAH	9		10		ns
^{t}RAS	50	10,000	60	10,000	ns
^{t}RC	84		104		ns
^{t}RCD	11		14		ns
^{t}RCH	0		0		ns
^{t}RCS	0		0		ns
^{t}RP	30		40		ns
^{t}RRH	0		0		ns
^{t}RSH	13		15		ns

NOTE: 1. ^{t}OFF is referenced from rising edge of RAS# or CAS#, whichever occurs last.

EARLY WRITE CYCLE



TIMING PARAMETERS

SYMBOL	-5 MIN	-5 MAX	-6 MIN	-6 MAX	UNITS
^{t}ACH	12		15		ns
^{t}AR	38		45		ns
^{t}ASC	0		0		ns
^{t}ASR	0		0		ns
^{t}CAH	8		10		ns
^{t}CAS	8	10,000	10	10,000	ns
^{t}CRP	5		5		ns
^{t}CSH	38		45		ns
^{t}CWL	8		15		ns
^{t}DH	8		10		ns
^{t}DS	0		0		ns
^{t}RAD	9		12		ns

SYMBOL	-5 MIN	-5 MAX	-6 MIN	-6 MAX	UNITS
^{t}RAH	9		10		ns
^{t}RAS	50	10,000	60	10,000	ns
^{t}RC	84		104		ns
^{t}RCD	11		14		ns
^{t}RP	30		40		ns
^{t}RSH	13		15		ns
^{t}RWL	13		15		ns
^{t}WCH	8		10		ns
^{t}WCR	38		45		ns
^{t}WCS	0		0		ns
^{t}WP	5		5		ns