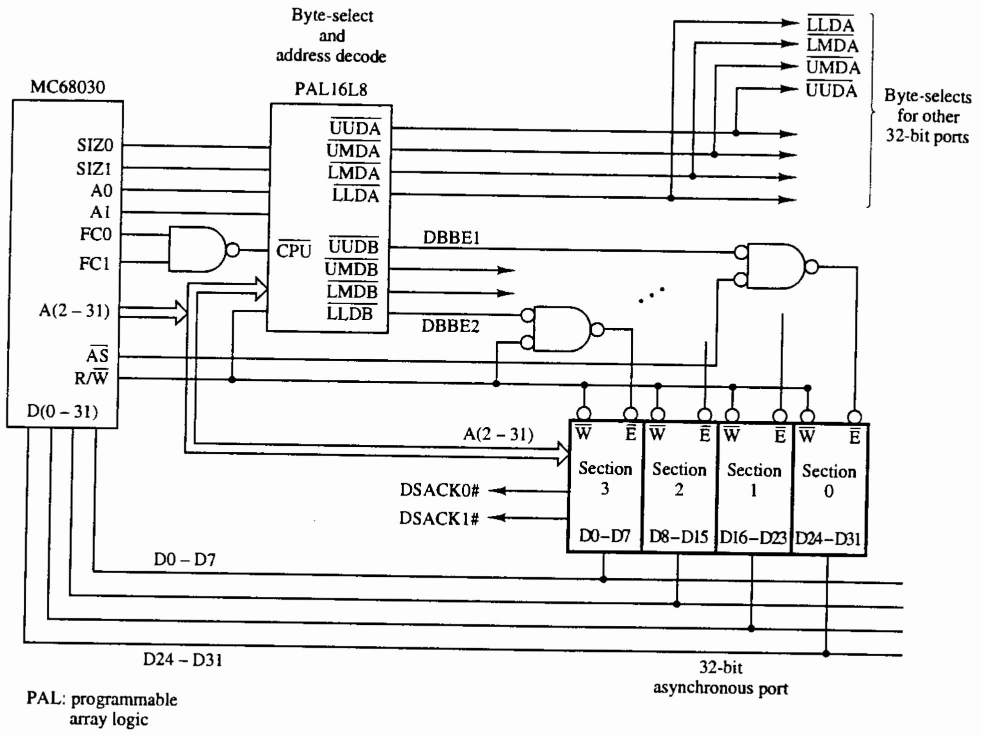
Chip pair 0: FFF00000, FFF00004, FFF00008, FFF0000C, ..., FFF3FFFC
Chip pair 1: FFF00001, FFF00005, FFF00009, FFF0000D, ..., FFF3FFFD
Chip pair 2: FFF00002, FFF00006, FFF0000A, FFF0000E, ..., FFF3FFFE
Chip pair 3: FFF00003, FFF00007, FFF0000B, FFF0000F, ..., FFF3FFFF

.

.

.

(b) Physical addresses (in hex) assigned to some of the memory chips



(c) Motorola 68030 interfaced to an asynchronous 32-bit memory port (Adapted from [1])

Figure 3.11 (concluded)

In both cases, it is the selected external slave device that controls the actual length of the bus cycle by sending the above signals to the processor to terminate the cycle. (The bus error signal BERR# is also used to terminate the bus cycle in the absence of DSACKs or STERM# to indicate a bus error condition.) If Figure 3.11a shows the interface on the memory side, Figure 3.11c shows the interface on the Motorola 68030 microprocessor side. The PAL (programmable array logic) device[20] (or an equivalent) is used to provide the byte-enables for the asynchronous 32-bit port.

[20]The PALs are devices used to provide reliable, high-performance substitutes for conventional TTL logic. Their easy programmability allows for quick design of "custom" chips for microprocessor interface and typically results in a more compact board.

Example 3.9: Interfacing a 32-Bit Memory to Intel 80x86

Figure 3.12 shows the simplified diagram of interfacing a 32-bit memory to an Intel 80x86 microprocessor. In this example, each of the four byte sections is made up of one 2K × 8 SRAM chip. The interface bus controller (not shown) provides the read and write commands to memory (MRDC# and MWTC#), the control signals for the address latches (ALE#) and data transceivers (DEN#), and sends back to the processor the READY# active signal to end the bus cycle. Eleven address bits (A12–A2) are routed as "word address" to memory; the four byte enables BE0#–BE3# are also sent to the respective four byte sections of memory, conditioned with the memory write command MWTC#. The input pins OE# of the memory chips are triggered by the MRDC# command, and therefore *all four sections drive the data bus during read operations*. For write operations, the MWTC# is gated with the byte-enable signals to specify which byte sections will store incoming data bytes.

3.2.3 DRAM Memories

8207 DRAM controller. The design and interface of DRAM memories presents more interest because it is the type of main memory most commonly accessed by the processor since it contains the program, data, and information used in processing. The logic used for interfacing to DRAM memory must primarily time-multiplex the two parts of the supplied address, generate the row address (RAS#) and column access (CAS#) strobes, and provide the memory-refresh logic. While earlier designs used a number of different discrete components to implement this interface, the use of the "programmable DRAM controller" chip simplifies this task considerably. DRAM controllers can be used in either noninterleaved or interleaved memory configurations. A number of such DRAM controller chips exist, such as the Intel 8207 [6], which can drive up to 4 banks composed of 256-Kbit DRAM chips each; the TI 74ALS6301 [7], which can drive up to 4 banks composed of 1-Mbit DRAM chips each; the Integrated Devices Technology 79R3721 [8], which supports memories composed of 1-Mbit to 16-Mbit DRAM chips in both noninterleaved and two-way interleaved configurations, in cache burst fill cycles, and in "page mode" operation using on-chip page detection (to be explained at the end of the chapter); etc. All have similar structures, input and output signals, and they function in more or less the same way. In most of our discussion and examples we will be using the Intel 8207. An example using ALS6301 is given at the end of the chapter.

Figure 3.13 shows the detailed internal structure of the Intel 8207 DRAM controller. The 8207 DRAM controller is designed to interface easily with a wide variety of microprocessors to 16K, 64K, and 256K DRAM devices. The 8207 can be used in both single- and dual-port configurations; two internal port interfaces (each havings its own "port enable" input, PEA# and PEB#) allow two independent processors to access a single main memory controlled by one 8207 DRAM controller. Each port can be programmed separately to interface to either synchronous or asynchronous environments; there exist output "data transfer acknowledge" signals (XACKA#/ACKA#, XACKB#/ACKB#, etc.) which can be used for that. Internally the controller has the logic needed to refresh the DRAM chips connected to it. The 9-bit multiplexed address is placed on the output pins AO0–AO8. The controller's LEN output can be used as another "address latch enable" signal, and the chip can be programmed for either internal or external refresh option.[21] Finally, as we

[21]Earlier DRAM controller chips required an external "memory timing controller" chip to generate the control signals needed by the DRAM controller (to access and refresh the dynamic memory, and to arbitrate between refresh and access cycles).

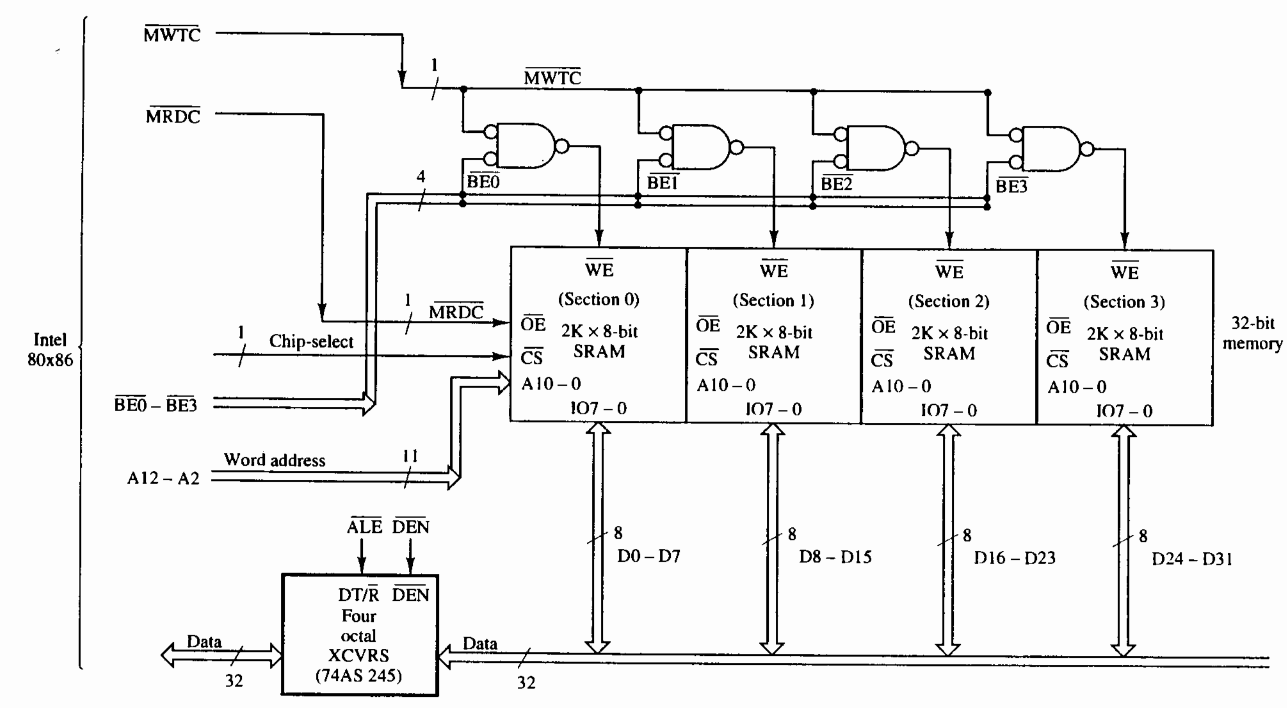


Figure 3.12 Thirty-two-bit SRAM memory interfaced to an Intel 80x86 micro processor. (Adapted from [5]. Reprinted by permission of Intel Corporation, Copyright/Intel Corporation 1986.)

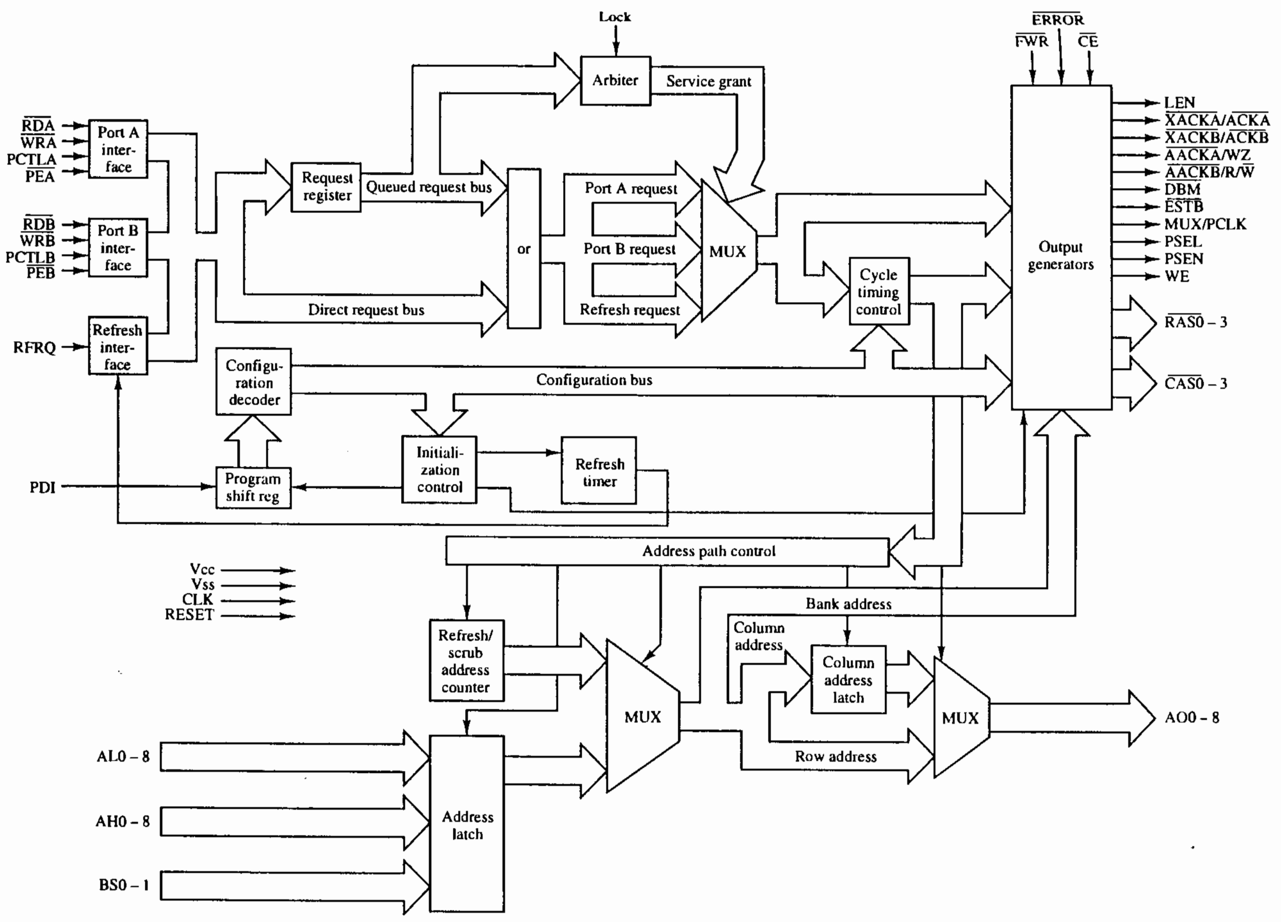


Figure 3.13 DRAM controller (internal structure of the Intel 8207) [6]. Reprinted by permission of Intel Corporation, Copyright/Intel Corporation 1983.

will explain in more detail below, the controller issues four different pairs of RAS# and CAS# signals.

The 8207 DRAM controller is interfaced to 16K, 64K, and 256K DRAMs using the address connections of Figure 3.14. If, for example, a 16-bit memory is used, then A0 will be used to effectively generate the DRAM "byte-enable" signal and A1 and A2 will be applied to the 8207's "bank-select" input pins BS0 and BS1 to select one of up to four memory banks that may be connected to the controller (explained below in more detail). The size of the DRAM determines the number of the 8207 input address pins to be used (unassigned address input pins may be strapped to high or low).

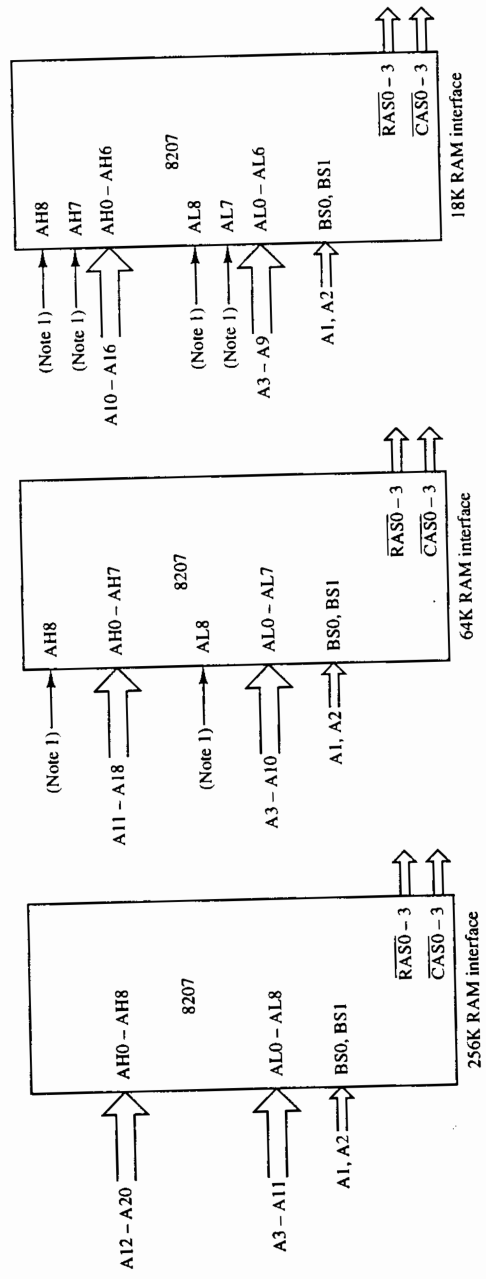
We said earlier that the DRAM controller can be programmed to control up to four memory banks, each bank having its own RAS#/CAS# pair. The default condition has the controller control four occupied DRAM banks; in this case, RAS0#/CAS0# will be routed to bank 0, RAS1#/CAS1# to bank 1, RAS2#/CAS2# to bank 2, and RAS3#/CAS3# to bank 3, as shown in Figure 3.15. As another example, we can see from Figure 3.15 that when the controller is programmed for a two-bank configuration, then a pair of RAS# signals and a pair of CAS# signals are activated per bank.

Interfaces using the 8207 DRAM controller. Figure 3.16 shows the simplified block diagram of the 8207 DRAM controller and the way it receives the address, multiplexes it, and supplies it to a 256K × 1-bit DRAM chip along with the RAS# and CAS# signals. First of all, the controller is assumed here as configured to control 256K locations and receive an 18-bit input address (see Figure 3.14). Each 9-bit half of the address is connected to the respective row address (AL0–AL8) and column address (AH0–AH8) input pins of the DRAM controller. As Note 1 in this figure explains, which 18 address bits one should connect to the DRAM controller input pins depends on the particular design. A multiplexer inside the controller, along with other timing circuitry, will properly sequence in time the two 9-bit halves of the address to the controller output lines AO0–AO8 along with their respective RAS# and CAS# signals. The design here shows applying the RAS0# and CAS0# to the memory chip.

Example 3.10: DRAM Controller Interfaced to a 16-Bit Memory

Figure 3.17 shows the details of an example configuration in which the DRAM controller is used to control a 16-bit DRAM subsystem of total capacity 0.5 Mbytes constructed using 256K × 1-bit DRAM chips, eight per byte section. Each 256K DRAM chip has nine input address lines requiring multiplexing to produce the 18 address bits needed to access all its 256K cell locations. At the beginning of every bus cycle, the respective RAS# is asserted by the DRAM controller along with the lower 9 address bits to address a row; then CAS# is asserted along with the upper 9 address bits to specify a column.

The figure shows the use of external logic (the "section-select logic") to decode Intel's A0 and BHE# outputs to condition the write-enable signals for the even and odd byte sections. For a read cycle, both byte sections of the 16-bit memory are triggered to drive their respective byte lanes of the data bus. In this example, the next 18 address bits (A1–A18) needed to access a location inside this 0.5M-byte memory are submitted to the DRAM controller. Finally, the remaining 5 most significant bits are used to assign/allocate actual physical addresses to this memory subsystem (via the address decode logic) that triggers the controller's PE# (enable) input pin. Since only one memory bank is connected, the DRAM controller has been programmed (initialized) to operate with only "1 bank occupied"; furthermore, since the bank-



Notes:
1. Unassigned address input pins should be strapped high or low.
2. A0 along with BHE are used to select a byte within a processor word.
3. Low-order address bits are used as bank select inputs so that consecutive memory access requests are to alternate banks allowing bank interleaving of memory cycles.

Figure 3.14 Address connections to the 8207 DRAM controller for 256K, 64K, and 16K DRAM devices [6]. Reprinted by permission of Intel Corporation, Copyright/Intel Corporation 1983.

Programmed mode	Bank-select input		RAS#/CAS# pair allocation	
	BS1	BS0		
Four banks occupied	0	0	RAS0#, CAS0# to bank 0	Default
	0	1	RAS1#, CAS1# to bank 1	
	1	0	RAS2#, CAS2# to bank 2	
	1	1	RAS3#, CAS3# to bank 3	
Three banks occupied	0	0	RAS0#, CAS0# to bank 0	
	0	1	RAS1#, CAS1# to bank 1	
	1	0	RAS2#, CAS2# to bank 2	
	1	1	Bank 3 unoccupied	
Two banks occupied	0	0	RAS0,1#, CAS0,1# to bank 0	
	0	1	RAS2,3#, CAS2,3# to bank 1	
	1	0	Bank 2 unoccupied	
	1	1	Bank 3 unoccupied	
One bank occupied	0	0	RAS0–3#, CAS0–3# to bank 0	
	0	1	Bank 1 unoccupied	
	1	0	Bank 2 unoccupied	
	1	1	Bank 3 unoccupied	

Figure 3.15 Bank selection decoding and word expansion on the 8207 DRAM controller [6]. Reprinted by permission of Intel Corporation, Copyright/Intel Corporation 1983.

selects BS0 and BS1 pins of the DRAM controller are connected to logic 0, then from Figure 3.15 it is concluded that all four pairs of RAS# and CAS# strobes will be assigned[22] to this bank 0.

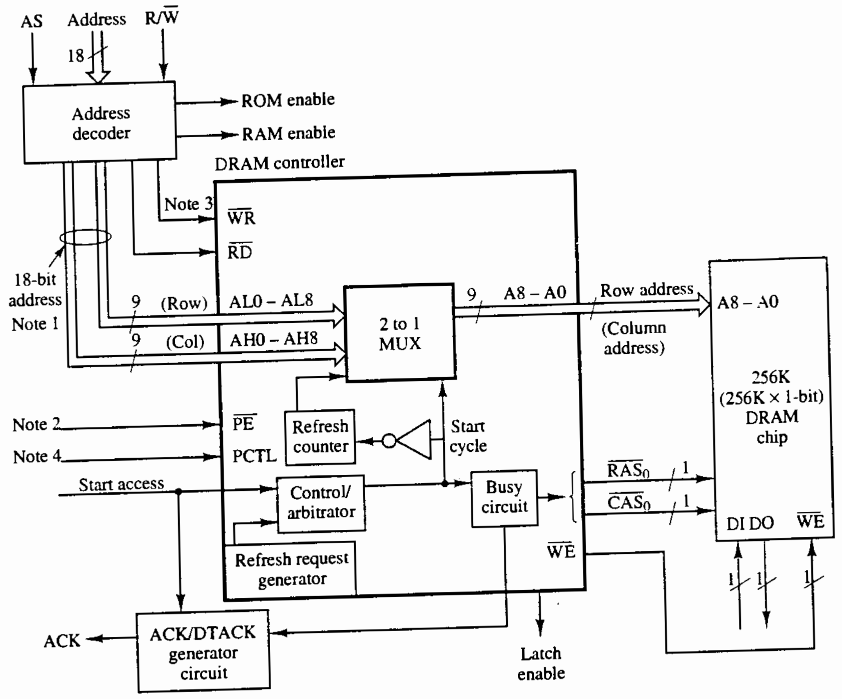
To implement a 1-Mbyte memory, we would need one more memory bank of the type shown on Figure 3.17, for a total of 32 chips, 16 chips per section. Instead of 19 address bits, 20 processor address bits must now be used, and the 8207 programmed to realize that two banks are connected to it. To differentiate between these two banks, an unused address bit must now be connected to pin BS0, while BS1 must be low. (If we connect address bit A19, we will make these two memory banks occupy two contiguous 512-Kbyte memory areas; if we connect address bit A1, we will implement "2-way word interleaving" in which Bank 0 will have the even word locations and Bank 1 the odd word locations, as we will see later.)

Example 3.11: Interface to Four 256-Kbyte Noninterleaved DRAM Banks

Use the 8207 DRAM controller to control a 16-bit DRAM memory of total capacity 16M bytes constructed out of the TMS 4256 (a 256K × 1-bit) DRAM chips. Assume a little-endian microprocessor with a 24-bit address bus and a 16-bit data bus. Show the memory's interface to these bus lines.

Solution: Figure 3.17 showed the interface of the 8207 DRAM controller to one 16-bit memory bank of capacity 512 Kbytes. The 8207 can control up to four such 256-Kbyte memory banks as shown in Figure 3.18a. The 8207 DRAM controller has been programmed to operate with "4 banks occupied" and, therefore, will supply each bank with its own pair of RAS#/CAS# strobes. The memory subsystem's wordlength is 16 bits and address bit A0 is again used (along with the BHE#) to identify the byte. Since this memory has 2 Mbytes capacity, it requires a

[22]If not all banks are occupied, the RAS# and CAS# strobes can be used to allow wider memories ("word expansion") to transfer wider data words without increasing the loading on the RAS# and CAS# drivers.



Notes:
1. Which address bits are supplied to the DRAM controller depends on the particular design. (For example : A0 may not be applied to the controller if used to distinguish between even and odd memory sections: or low-order address bits may be used as bank-select inputs in interleaved memory designs.)
2. Usually external logic combines the remaining unassigned address bits to generate the chip-enable (or chip-select) signal.
3. Directly from the microprocessor's output pins or from the bus controller.
4. Port control for configuring one of the two ports to accept command signal (PCTL).

Figure 3.16 Simplified block diagram of a DRAM controller and its interface to a 256K × 1-bit DRAM memory chip.

21-bit address to access any internal byte location; address bits A1–A18 are supplied to the controller's input pins AL0–AL8 and AH0–AH8, while A19 and A20 are connected to the BS0, BS1 inputs. Since the four banks occupy *contiguous* (noninterleaved) 512-Kbyte memory areas, the BS0 and BS1 pins must be driven by bits A19 and A20.[23] Because the DRAM controller is enabled when address bits A21–A23 are zeros (PE receives the number zero output from the decoder), this 2-Mbyte memory is assigned to the lowest 2 Mbytes of the 16-Mbyte maximum main memory space.

[23]Using the high-order bits to select the bank is also sometimes called "high-order interleaving" or "bank switching."

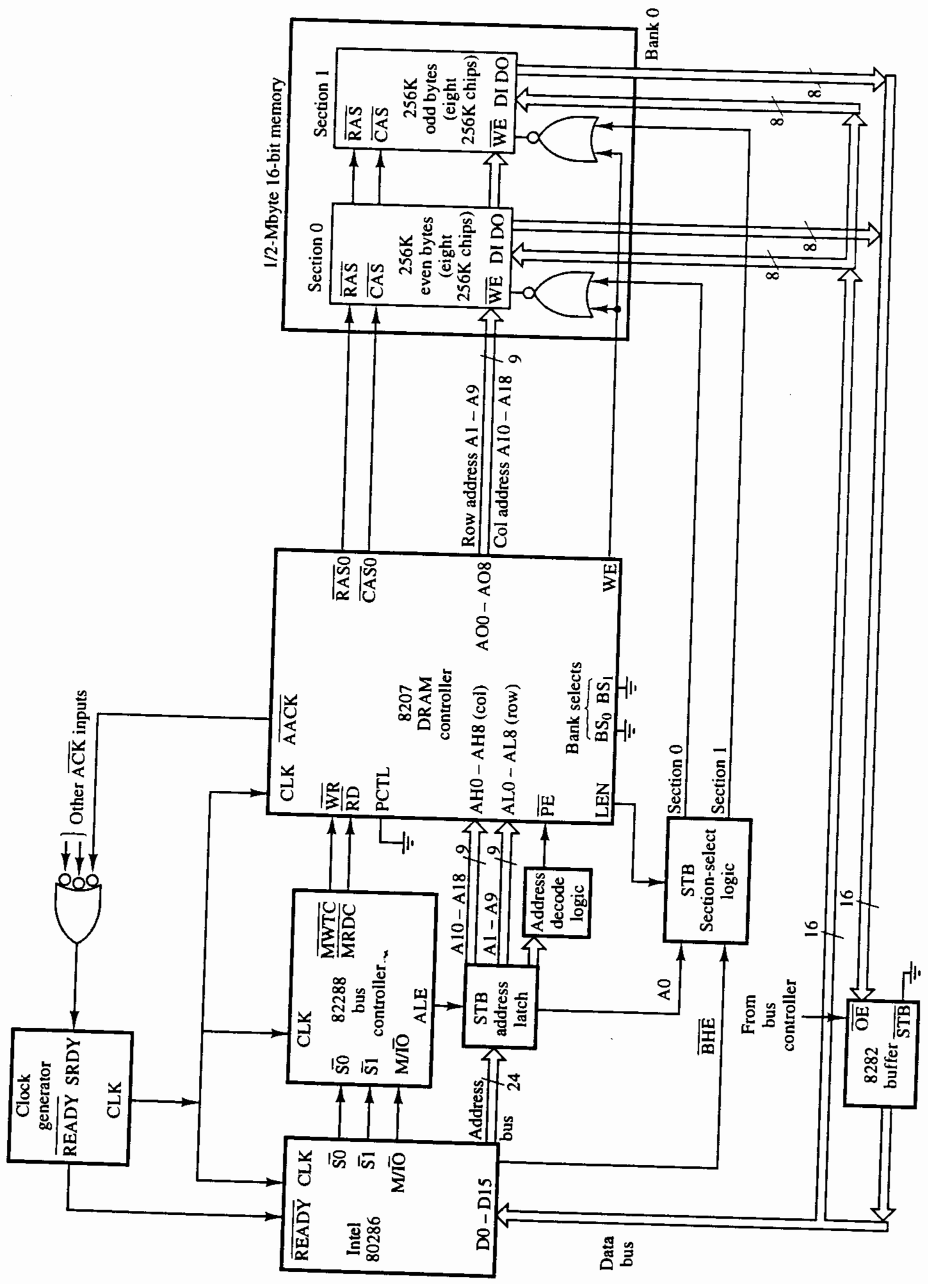
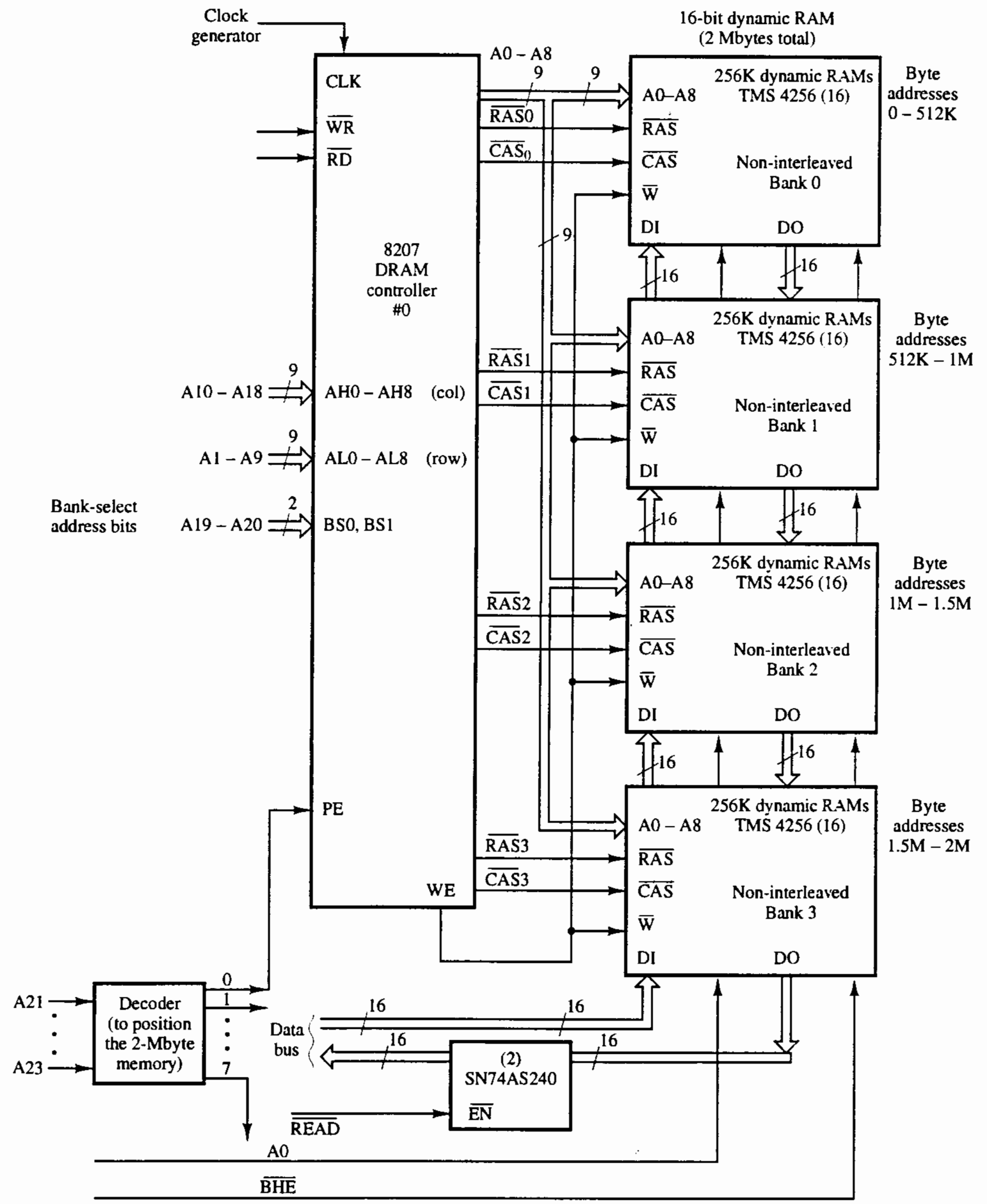
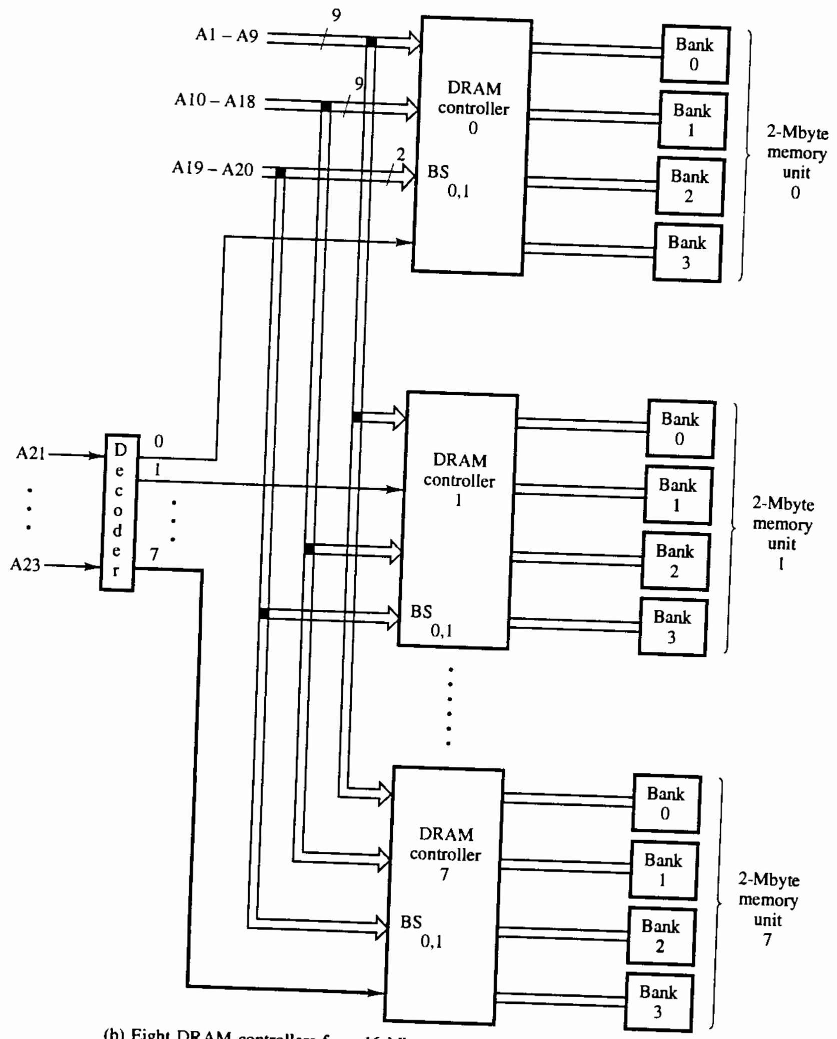


Figure 3.17 Interfacing to a 0.5M-byte memory (wordlength 16 bits) through the 8207 DRAM controller. (Adapted from [6].) Reprinted by permission of Intel Corporation. Copyright/Intel Corporation 1983.



(a) Details of interfacing DRAM controller #0 to four, noninterleaved, 256-Kbyte memory banks. (Adapted from [6] and [7].) Reprinted by permission of Intel Corporation, Copyright/Intel Corporation 1983 and Texas Instruments 1988.

Figure 3.18 DRAM controller chip controlling a 16-Mbyte dynamic memory. (Memory is constructed using 256K × 1-bit DRAM devices.) No interleaving is used.



(b) Eight DRAM controllers for a 16-Mbyte noninterleaved dynamic memory

Figure 3.18 (concluded)

Figure 3.18b shows the design of a 16-Mbyte dynamic memory (accessed with 24-bit addresses), which uses 8 DRAM controllers, each one controlling a 2-Mbyte memory of the kind shown in Figure 3.18a. Again, these eight 2-Mbyte memory units are not interleaved; they are contiguous because the selection of the 2-Mbyte memory unit (actually, the selection of its DRAM controller) is done by decoding the most significant address bits A21–A23.

Dual-ported memories. Dual-ported memories allow two independent units (such as one processor and one graphics controller, or two processors) to access the same memory, for example in situations when the microprocessor is updating (writing to) memory while the graphics controller accesses memory to read data and send them to the graphics display. A DRAM controller that provides two ports can do that by internally queueing each unit's requests, arbitrating between these requests, and directing data to or from the appropriate port. Furthermore, one of the two units can be interfaced synchronously and the other interfaced asynchronously.

Error detection and correction units (EDCU). Memory error detection and correction can be incorporated in the design by using the DRAM controller and an error detection and correction unit (EDCU). The simplified block diagram of such a design is shown in Figure 3.19. Additional memory is provided to the 32-bit DRAM memory to hold the 8 parity bits. The 8 parity bits are formed here by using a Hamming code (or modified Hamming code) scheme, which allows for single-error correction and double-error detection (SEC-DED). The EDCU works on both read and write cycles, providing corrected data to the rest of the system and to the memory cells.

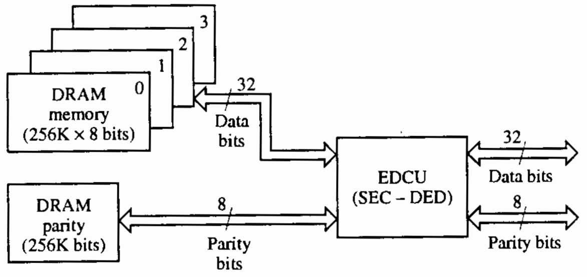


Figure 3.19 DRAM memory with error-detection and error-correction circuitry (EDCU).

On read cycles from memory, the EDCU executes a read-modify-write operation. It uses the 8 parity bits to check and if necessary correct the 32-bit doubleword read out from memory. The corrected 32-bit doubleword and the 8-bit parity are then written back into the DRAM cells and all 40 bits are sent to the rest of the system. On write cycles to memory, the EDCU receives and checks the 32 data bits and the 8 parity bits, corrects any errors, a new 8-bit parity is generated from the new 32-bit doubleword, and all 40 bits are stored into the DRAM cells.

3.3 ADVANCED DRAM ACCESS MODES

In Chapter 2 we covered the *burst transfer mode*, for which, once the initial access to memory occurs, subsequent data can be retrieved more quickly since the overhead of transmitting their addresses is eliminated. Lately, additional circuitry has been added on recent