The iRAM—an innovative approach to microprocessor memory solutions

by JOHN J. FALLIN

Intel Corporation
Aloha, Oregon

ABSTRACT

When designing the local read/write memory for a microprocessor system, several specific needs should be addressed. These needs include flexibility, ease of use, and low cost. The iRAM, a new product, effectively addresses all of these needs. The iRAM is a complete memory system that combines a high-density 8K x 8 DRAM array with complete memory and refresh control on a single chip. The iRAM combines the advantages of both the DRAM and SRAM, making it the ideal choice for microprocessor memory. The iRAM's concept and features will be discussed as well as its use in local memory systems for both 8-bit and 16-bit microprocessors.
MICROPROCESSOR MEMORY NEEDS

With the advent of microprocessors, more system designers are involved in the design of microprocessor local memories. Analysis of the memory system highlights several important memory characteristics: ease of use, low cost, and flexibility. In the past, the designer has had to decide on the relative importance of each of these characteristics and then make a choice between DRAMs and SRAMs, neither of which satisfies all of the requirements in all cases.

Ease of Use

To understand how a new memory, the iRAM, meets all of these system criteria, it is necessary to examine why these three characteristics are so important. Ease of use simplifies the interface between the memory and the microprocessor. This translates into an easier, shorter design cycle while minimizing the component count. The advantages to the manufacturer include quicker time to market, lower cost, and less board-area requirements.

Cost

If ease of use were the only criterion used in selecting a memory component for the microprocessor local bus, then the SRAM would be the obvious choice. The SRAM is the simplest memory to use. The SRAM, however, has a complex storage cell, making it more difficult to manufacture at the higher densities and thus more expensive. Excessive cost limits SRAMs to small memory systems (those typically less than 8K bytes).

Higher density and corresponding lower cost allow DRAMs to be used in larger memories. The refresh requirement of DRAMs, however, makes the system more complex and difficult to design. The cost of the refresh control, however, can be amortized in large systems (those greater than 64K), making DRAMs very cost-effective in this memory segment.

Flexibility

Flexibility is another characteristic that simplifies the design cycle. Rather than design separate systems for different models of the end product or redesign for an upgrade, which only requires more memory, the designer can design around the universal site. This is a JEDEC standard site (Figure 1), which is capable of accommodating other JEDEC standard memory devices, both volatile and nonvolatile RAMs, EPROMs, PROMs, and E²PROMs.

Many advantages are gained through the use of the universal site. In the early design phase, the mix between ROM, EPROM, and RAM may not be known. Using the universal site can allow the hardware design to be completed at this preliminary stage. This permits software to be debugged on the final product and the RAM/ROM mix to be easily varied later. The universal site also allows for easy upgrades because memory devices from 2K to 16K fit the site.

As mentioned earlier, the SRAM, with its easy-to-use bus structure, fits nicely in small systems (those 8K and smaller). For large systems (those greater than 64K), where the cost of using SRAMs becomes prohibitive, DRAMs are used. Between 8K and 64K a transition region exists in which neither SRAMs or DRAMs are a good fit. In this region low cost, ease of use, and flexibility are all important. In the past, compromises had to be made between these features. Now, because of an innovative new development in RAM technology, these compromises are eliminated. The iRAM, which is the first in a generation of intelligent memories, is a complete memory system on a single silicon chip. The iRAM achieves low cost through the use of a high-density DRAM array. It is easy to use because of its onboard refresh control and is manufactured in a JEDEC compatible 28-pin package, making it very flexible.

A SOLUTION: THE iRAM

The Intel iRAM comes in two different varieties: the 2186 and the 2187. Both iRAMs are 5-volt only DRAM subsystems organized as 8192 8-bit words, but they operate differently. In most microprocessor systems, the asynchronously refreshed iRAM, the 2186, is used. This iRAM incorporates a refresh timer to maintain refresh. To synchronize refresh and access cycles, an onboard arbiter queues cycles and provides a ready handshake signal (RDY) to indicate a pushout in the cycle. (The RDY output is commonly used to request the insertion of WAIT states when required). The 2187 replaces the RDY output with a refresh enable (REFEN) input. During 2187 operation, the user strobes REFEN to maintain refresh. The 2187 is ideal for use in systems where WAIT states cannot be tolerated, or in systems where accesses occur in a manner adequate to maintain refresh (such as in some graphics systems).

Internal Structure

A block diagram of the iRAM is shown in Figure 2. For ease of discussion, the iRAM is divided into five basic sections; a 65,536-bit DRAM array, a refresh-request time, a refresh
row-address counter, a high-speed cycle arbiter, and full control circuitry.

The refresh timer provides refresh requests to the arbiter, which synchronizes the refresh cycles with access cycles. Refresh addresses are generated by the refresh row-address counter, multiplexed with external row addresses and then routed to the DRAM array. This DRAM array is divided into four quadrants, each containing 128 rows and 128 columns. Internal control circuitry synchronizes all internal events. The 2187 differs from the 2186 in that it has no arbiter and the refresh timer is user-controlled, instead of being free running.

**iRAM Operation**

The pinouts of the 2186 and the 2187 are shown in Figure 1. Note that, except for pin 1, the iRAMs have identical pinouts to the 8K x 8 SRAM and EPROM, providing flexibility for the RAM/ROM mix in the system. The 2186 operates much
like an SRAM during read and write cycles. The only differences involve the CE and WE inputs and the RDY output (2186) or the REFEN input (2187). Because the iRAM is an edge-triggered device, its control inputs must be clean transitioning. Also, because of the dynamic nature of the device, its addresses are latched on the leading (falling) edge of LE, eliminating latches in some systems. Both of these requirements are very easy to implement at the system level, as will be shown later. Another difference of the iRAM is that during a write cycle, the iRAM latches in data on the leading (falling) edge of WE, as opposed to the SRAM, which latches data on the trailing edge of WE.

During operation, when the 2186 is accessed, the RDY output will occasionally be pulled low, indicating a deferred cycle. This RDY output is normally routed back to the microprocessor READY or WAIT input. WAIT states are injected only when required during an access/refresh conflict.

Another type of cycle that exists for the iRAM is the false memory cycle, in which the iRAM receives an active CE but then neither OE or WE comes low. The false memory cycle is somewhat like a RAS-only refresh in that the row selected by the seven external row addresses is refreshed.

The 2187 operates slightly differently than 2186 in that the user now strobes the REFEN input to maintain refresh. The REFEN strobe must be timed such that access cycles are not attempted during refresh cycles.

APPLICATION EXAMPLES

In the section that follows, several microprocessor/iRAM interfaces will be discussed to demonstrate the iRAMs flexibility and ease of use. These will include both 8-bit and 16-bit systems, as well as a synchronous 2187 design. Also included is an iRAM/microcontroller system design.

8088/2186

Figure 3 shows the simplicity of system design in an 8088/2186 max mode system that runs at 5 MHz without WAIT states. Operating like a clocked static RAM, the iRAM requires stable addresses on the falling edge of a CE, which should be transient-free to prevent multiple selection of the iRAM. Only one TTL device is needed to perform this function.

To guarantee a clean transitioning CE, the CE decoder is only enabled when its address inputs are stable. This is accomplished by using a cross-coupled NAND gate to generate a decoder-enable signal. Also used as a decoder enable is status

Figure 4—8086/2186 system
bit 2 (M/I0). This guarantees that a CE will only occur if the ALE output from the processor is associated with a memory read or write cycle. This eliminates false memory cycles, which have an extended CE high-time requirement not allowed for in this design. Note that the address inputs to the decoder are latched, because addresses can transition between bus cycles even if ALE is not high. Because the decoder is enabled during this period, transitions could propagate through the decoder and cause invalid CEs to occur, jeopardizing data integrity.

The WEs for the iRAM array are generated directly from the 8288 bus controller. The 8288 output MWTC has a delayed falling edge, which allows for the leading-edge write requirement of the iRAM, while providing compatibility with SRAMs.

8086/2186

The 8086/2186 system shown in Figure 4 is similar to the previously described 8088 system and also runs at 5 MHz with zero WAIT states using a 2186-30. The major difference is in the CE generation circuitry. Because this is a 16-bit system, a 2186 can receive a CE but no WE or OE, resulting in a false memory cycle. This condition occurs in the unwritten byte of the 16-bit word during a byte-write operation. The longer CE high-time requirement of the false memory cycle can be accommodated by using the CE generation circuitry shown in Figure 4. The circuitry is basically a 2-bit counter that initiates

![Figure 5—CE circuit timings](image)

![Figure 6—80186/2186 system](image)
its count on the rising edge of ALE. On the next clock (falling edge) after this, CE is activated and remains so for two clocks before returning high. This provides a CE high time of approximately two clock periods, or, for a 5 MHz 8086, approximately 400 ns, sufficient to meet the CE high-time requirement. Timings for this circuit are shown in Figure 5.

Because the 8086 is configured in the min mode, its WR output falls too early in the cycle to satisfy the leading-edge write requirement of the iRAM. To meet this requirement, a cross-coupled NAND gate circuit delays WE's falling edge. The WE rising edge, however, is not delayed, which allows for SRAM compatibility.

80186/2186

Figure 6 depicts an 80186/2186 interface. The 80186 is a highly integrated device that combines an enhanced 8086
CPU with 10 commonly used system components, including a
bus driver, clock generator, interrupt controllers, and pro­
grammable memory chip selects.

The circuitry used to generate the CE is similar to that for
the 8086 system in the last example. Again, a 2-bit counter is
used, although its count is initiated differently. A pro­
grammable memory chip select is used to toggle the first flip­
 flop from a zero to a one on the first clock. This action causes
a CE to be generated. On the next clock, the first flip-flop will
remain set and the second flip-flop will also be set. On the
next clock after this, the first flip-flop will toggle to zero
(clear), causing CE to return high. The next clock clears the
second flip-flop, which completes the sequence.

A timing diagram of this circuit is shown in Figure 7. Note
that a CE high time of approximately two clocks is provided,
allowing for false memory-cycle operation.

The WR output of the 80186 becomes active too early to
allow for a leading-edge write. This situation is remedied by
gating WR with the Q output of the second flip-flop, which
delays it long enough to meet the 2186 write requirements.
Note that the rising edge of WE is timed such that a trailing­
edge write is also allowed for full SRAM compatibility.

8088/2187

The circuit in Figure 8 interfaces the synchronous 2187 to an
8088 in such a way as to make the iRAM refreshes invisible.
This method, which is commonly used in many micro­
processor/DRAM systems, allows the iRAM to be refreshed
every time an OP Code fetch is performed. OP Code fetches
are restricted to some memory other than the iRAM (typically
EPROM or ROM) because the 2187 cannot be accessed dur­
ing a refresh cycle. To meet the 2187 iRAM refresh require­
ments, at least 128 OP Code fetches need to be performed in
every 2-ms period, a condition easily met. Certain conditions
could, however, jeopardize iRAM refresh. These conditions
would include extended-hold (DMA) and WAIT (single-step)
states, in which OP Code fetches do not occur. If either of
these conditions were allowed to persist for an extended
period of time, the contents of the iRAM could be lost.

To synchronize the 2187 refresh with the processor OP
Code fetches, the signal M1 is decoded from the processor
status bits. This signal will go low at the beginning of any OP
Code fetch, thus initiating a refresh in the 2187. M1 then
returns high before the OP Code fetch is complete, allowing
the 2187 to be accessed on the next cycle if necessary.

Figure 9 shows a two-chip microcomputer system employ­
ing the 2186 with an 8051 microcontroller, and Figure 10

---

Figure 9—8051/2186 system

Figure 10—8051/2186 timings

From the collection of the Computer History Museum (www.computerhistory.org)
shows timings for this system. The microcontroller includes 4K bytes of program memory on board (EPROM), and the 2186 provides 8K bytes of data memory.

The interface to the multiplexed 8751 bus is very simple. Because the 2186 latches address on the falling edge of \( \overline{CE} \), no address latches are necessary in the system. \( \overline{CE} \) is generated on the falling edge of ALE only for accesses to the lower 32K of external data memory. This is assured by gating ALE with P2.7, which acts as the highest-order address during external memory operations, allowing ALE to generate a \( \overline{CE} \) only when P2.7 is low. To ensure that \( \overline{CEs} \) are not generated at other times, P2.7 is initially set to a 1, which it will continue to output until an external memory operation is done. After the external memory operation, P2.7 will return to its preset 1. Because the ports are configured as open drain outputs, a pull-up resistor is included.

The reason for generating \( \overline{CEs} \) during external-data-memory operations only is less than obvious. During external-data-memory operations, the 8751 outputs an ALE once every 12 clock cycles, resulting in a \( \overline{CE} \) cycle time compatible with the 2186. The 8751, however, generates external ALEs at all other times also. These ALEs, which occur once every six clock cycles, must be inhibited from generating \( \overline{CEs} \) to the iRAM because the 2186 cycle time with wait specification would be violated.

The 8051 does not have a READY input; this does not, however, preclude the 2186 from being used with it. After examining the access-time requirements of 8051 data memory, it can be concluded that for speeds up to 8 MHz, a 2186-25 is fast enough to meet the 8051 memory speed requirements—even in its worst case not-ready condition. This access time requirement would be 850 ns for the 8051 at 8 MHz; the 2186-25 worse case access time with refresh is 675 ns.

CONCLUSIONS

The iRAM fulfills all three requirements of microprocessor memory. Using a dynamic RAM memory cell, it meets the density and low-cost criteria. Also incorporating complete on-chip control, the iRAM satisfies the ease-of-use and flexibility requirements. Several examples of microprocessor systems have been shown to demonstrate the ease of use. For memory sizes of 8K to 64K bytes, no other memory device simultaneously satisfies all microprocessor memory requirements as does the iRAM.