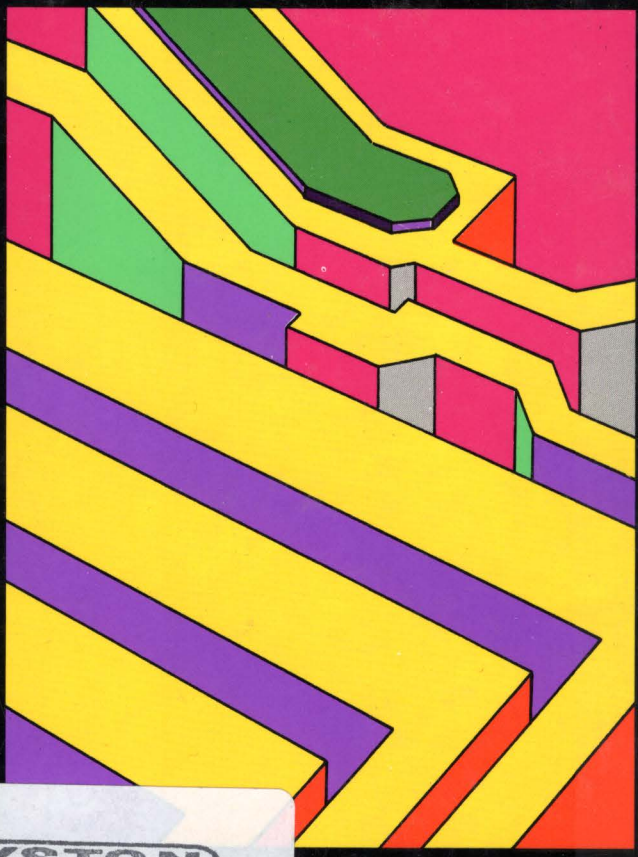
FIFO & CACHE TAG MEMORIES

APPLICATION NOTES



SGS-THOMSON
MICROELECTRONICS

FIFO & CACHE TAG MEMORIES

APPLICATION NOTES

FEBRUARY 1989

USE IN LIFE SUPPORT MUST BE EXPRESSLY AUTHORIZED

SGS-THOMSON' PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF SGS-THOMSON Microelectronics. As used herein:

1. Life support devices to systems are devices or systems which, are intended for surgical implant into the body to support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

TABLE OF CONTENTS

FIFO

Part Number	Org.	Access Time	I_{CC} Max		V_{CC}		Temperature Range	Pin Count
			Act	St.by				
MK4501K10	512 x 9	100ns	80mA	8.0mA	5V	±10%	0 to +70°C	32
MK4501K12	512 x 9	120ns	80mA	8.0mA	5V		0 to +70°C	32
MK4501K15	512 x 9	150ns	80mA	8.0mA	5V		0 to +70°C	32
MK4501K20	512 x 9	200ns	80mA	8.0mA	5V		0 to +70°C	32
MK4501K65	512 x 9	65ns	80mA	8.0mA	5V		0 to +70°C	32
MK4501K80	512 x 9	80ns	80mA	8.0mA	5V		0 to +70°C	32
MK4501N10	512 x 9	100ns	80mA	8.0mA	5V	±10%	0 to +70°C	32
MK4501N12	512 x 9	120ns	80mA	8.0mA	5V		0 to +70°C	32
MK4501N15	512 x 9	150ns	80mA	8.0mA	5V		0 to +70°C	32
MK4501N20	512 x 9	200ns	80mA	8.0mA	5V		0 to +70°C	32
MK4501N65	512 x 9	65ns	80mA	8.0mA	5V		0 to +70°C	32
MK4501N80	512 x 9	80ns	80mA	8.0mA	5V		0 to +70°C	32
MKI4501N10	512 x 9	100ns	80mA	8.0mA	5V	±10%	−40 to +80°C	28
MKI4501N12	512 x 9	120ns	80mA	8.0mA	5V		−40 to +85°C	28
MKI4501N15	512 x 9	150ns	80mA	8.0mA	5V		−40 to +85°C	28
MKI4501N20	512 x 9	200ns	80mA	8.0mA	5V		−40 to +85°C	28
MKI4501N65	512 x 9	65ns	80mA	8.0mA	5V		−40 to +85°C	28
MKI4501N80	512 x 9	80ns	80mA	8.0mA	5V		−40 to +85°C	28
MK45H01N25*	512 x 9	25ns	120mA	12mA	5V	±10%	0 to +70°C	28
MK45H01N35*	512 x 9	35ns	120mA	12mA	5V		0 to +70°C	28
MK45HO1N50*	512 x 9	50ns	120mA	12mA	5V		0 to +70°C	28
MK45H01N65*	512 x 9	65ns	120mA	12mA	5V		0 to +70°C	28
MK45H01N80*	512 x 9	80ns	120mA	12mA	5V		0 to +70°C	28
MK45H01N12*	512 x 9	120ns	120mA	12mA	5V		0 to +70°C	28
MK45H01K25*	512 x 9	25ns	120mA	12mA	5V	±10%	0 to +70°C	28
MK45H01K35*	512 x 9	35ns	120mA	12mA	5V		0 to +70°C	28
MK45H01K50*	512 x 9	50ns	120mA	12mA	5V		0 to +70°C	28
MK45H01K65*	512 x 9	65ns	120mA	12mA	5V		0 to +70°C	28
MK45H01K80*	512 x 9	80ns	120mA	12mA	5V		0 to +70°C	28
MK45H01K12*	512 x 9	120ns	120mA	12mA	5V		0 to +70°C	28
MK45H02N25*	1024 x 9	25ns	120mA	12mA	5V	±10%	0 to +70°C	28
MK45H02N35*	1024 x 9	35ns	120mA	12mA	5V		0 to +70°C	28
MK45H02N50*	1024 x 9	50ns	120mA	12mA	5V		0 to +70°C	28
MK45H02N65*	1024 x 9	65ns	120mA	12mA	5V		0 to +70°C	28
MK45H02N80*	1024 x 9	80ns	120mA	12mA	5V		0 to +70°C	28
MK45H02N12*	1024 x 9	120ns	120mA	12mA	5V		0 to +70°C	28
MK45H02K25*	1024 x 9	25ns	120mA	12mA	5V	±10%	0 to +70°C	32
MK45H02K35*	1024 x 9	35ns	120mA	12mA	5V		0 to +70°C	32
MK45H02K50*	1024 x 9	50ns	120mA	12mA	5V		0 to +70°C	32
MK45H02K65*	1024 x 9	65ns	120mA	12mA	5V		0 to +70°C	32
MK45H02K80*	1024 x 9	80ns	120mA	12mA	5V		0 to +70°C	32
MK45H02K12*	1024 x 9	120ns	120mA	12mA	5V		0 to +70°C	32
MK4503N10	2048 x 9	100ns	120mA	12mA	5V	±10%	0 to +70°C	28
MK4503N12	2048 x 9	120ns	120mA	12mA	5V		0 to +70°C	28
MK4503N15	2048 x 9	150ns	120mA	12mA	5V		0 to +70°C	28
MK4503N20	2048 x 9	200ns	120mA	12mA	5V		0 to +70°C	28
MK4503N65	2048 x 9	65ns	120mA	12mA	5V		0 to +70°C	28
MK4503N80	2048 x 9	80ns	120mA	12mA	5V		0 to +70°C	28
MK45H03N25*	2048 x 9	25ns	120mA	12mA	5V	±10%	0 to +70°C	28
MK45H03N35*	2048 x 9	35ns	120mA	12mA	5V		0 to +70°C	28
MK45H03N50*	2048 x 9	50ns	120mA	12mA	5V		0 to +70°C	28
MK45H03N65*	2048 x 9	65ns	120mA	12mA	5V		0 to +70°C	28
MK45H03N80*	2048 x 9	80ns	120mA	12mA	5V		0 to +70°C	28
MK45H03N12*	2048 x 9	120ns	120mA	12mA	5V		0 to +70°C	28

Note: *: Product to be introduced this year.
#: 300-MIL Package

FIFO (Cont'd)

Part Number	Org.	Access Time	I_{CC} Max		V_{CC}		Temperature Range	Pin Count
			Act	St.by				
MK45H03K25*	2048 x 9	25ns	120mA	12mA	5V	±10%	0 to +70°C	32
MK45H03K35*	2048 x 9	35ns	120mA	12mA	5V		0 to +70°C	32
MK45H03K50*	2048 x 9	50ns	120mA	12mA	5V		0 to +70°C	32
MK45H03K65*	2048 x 9	65ns	120mA	12mA	5V		0 to +70°C	32
MK45H03K80*	2048 x 9	80ns	120mA	12mA	5V		0 to +70°C	32
MK45H03K12*	2048 x 9	120ns	120mA	12mA	5V		0 to +70°C	32
MK45H13N25*	2048 x 9	25ns	120mA	12mA	5V	±10%	0 to +70°C	28 #
MK45H13N35*	2048 x 9	35ns	120mA	12mA	5V		0 to +70°C	28 #
MK45H13N50*	2048 x 9	50ns	120mA	12mA	5V		0 to +70°C	28 #
MK45H13N65*	2048 x 9	65ns	120mA	12mA	5V		0 to +70°C	28 #
MK45H13N80*	2048 x 9	80ns	120mA	12mA	5V		0 to +70°C	28 #
MK45H13N12*	2048 x 9	120ns	120mA	12mA	5V		0 to +70°C	28 #
MK45H04N25*	4096 x 9	25ns	120mA	12mA	5V	±10%	0 to +70°C	28
MK45H04N35*	4096 x 9	35ns	120mA	12mA	5V		0 to +70°C	28
MK45H04N50*	4096 x 9	50ns	120mA	12mA	5V		0 to +70°C	28
MK45H04N65*	4096 x 9	65ns	120mA	12mA	5V		0 to +70°C	28
MK45H04N80*	4096 x 9	80ns	120mA	12mA	5V		0 to +70°C	28
MK45H04N12*	4096 x 9	120ns	120mA	12mA	5V		0 to +70°C	28
MK45H14N25*	4096 x 9	25ns	120mA	12mA	5V	±10%	0 to +70°C	28 #
MK45H14N35*	4096 x 9	35ns	120mA	12mA	5V		0 to +70°C	28 #
MK45H14N50*	4096 x 9	50ns	120mA	12mA	5V		0 to +70°C	28 #
MK45H14N65*	4096 x 9	65ns	120mA	12mA	5V		0 to +70°C	28 #
MK45H14N80*	4096 x 9	80ns	120mA	12mA	5V		0 to +70°C	28 #
MK45H14N12*	4096 x 9	120ns	120mA	12mA	5V		0 to +70°C	28 #
MK45H08N25*	8192 x 9	25ns	120mA	12mA	5V	±10%	0 to +70°C	28
MK45H08N35*	8192 x 9	35ns	120mA	12mA	5V		0 to +70°C	28
MK45H08N50*	8192 x 9	50ns	120mA	12mA	5V		0 to +70°C	28
MK45H08N65*	8192 x 9	65ns	120mA	12mA	5V		0 to +70°C	28
MK45H08N80*	8192 x 9	80ns	120mA	12mA	5V		0 to +70°C	28
MK45H08N12*	8192 x 9	120ns	120mA	12mA	5V		0 to +70°C	28
MK4505MN25	1024 x 5	15ns	100mA		5V	±10%	0 to +70°C	24
MK4504MN33	1024 x 5	20ns	100mA		5V		0 to +70°C	24
MK4505MN50	1024 x 5	25ns	100mA		5V		0 to +70°C	24
MK4504SN25	1024 x 5	15ns	100mA		5V	±10%	0 to +70°C	20
MK4505SN33	1024 x 5	20ns	100mA		5V		0 to +70°C	20
MK4505SN50	1024 x 5	25ns	100mA		5V		0 to +70°C	20
MK45264N55	65 x 5 x 2	55ns	100mA		5V	±10%	0 to +70°C	24
MK45264N + 70	65 x 5 x 2	+ 70ns	100mA		5V		0 to +70°C	24
MK45265N55	65 x 5 x 2	55ns	100mA		5V		0 to +70°C	24
MK45265N + 70	65 x 5 x 2	+ 70ns	100mA		5V		0 to +70°C	24

CACHE TAGRAM

MK41H80N20	4096 x 4	20ns	120mA		5V	±10%	0 to +70°C	22
MK41H80N22	4096 x 4	22ns	120mA		5V		0 to +70°C	22
MK41H80N25	4096 x 4	25ns	120mA		5V		0 to +70°C	22
MK41H80N35	4096 x 4	35ns	120mA		5V		0 to +70°C	22
MK4202Q20*	2048 x 20	20ns	225mA	70mA	5V	±10%	0 to +70°C	68
MK4202Q22*	2048 x 20	22ns	225mA	70mA	5V		0 to +70°C	68
MK48H74N35*	8192 x 8	35ns	125mA		5V	±10%	0 to +70°C	28
MK48H74N45*	8192 x 8	45ns	125mA		5V		0 to +70°C	28
MK48H74N55*	8192 x 8	55ns	125mA		5V		0 to +70°C	28

Note: *: Product to be introduced this year.
#: 300-MIL Package

APPLICATION NOTES

BiPORT™ RAMs AND FIFOs MAKE GOING FAST TWICE AS EASY

INTRODUCTION

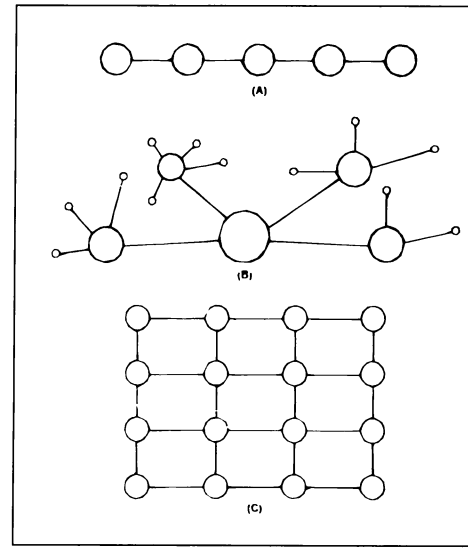
To anyone who grew up as I did, making pocket money mowing lawns in the summer time, some general things about going fast are abundantly clear. One approach is using a larger lawnmower, in fact, most of the lawns I worked on could have been done in three passes with a tractor and a gang mower. Unfortunately, houses, trees, flower beds, dogs and small children would have been in the way ; besides, I couldn't afford one. Another approach is to just go faster with the mower you've got. That works quite well within some limits. Eventually, though, you find your mower choking on the load. The most practical approach seems to be getting more mowers. Each one runs at a reasonable rate, working on a different part of the lawn until it is done.

Systems designers have found themselves faced with much the same dilemma ; how do you get things done quickly. The oldest approach has been to apply bigger and bigger systems to the job. But as with the tractor and gang mower, both the capital investment in the machine and the intrinsic operating obstacles of such a system often make that an impractical choice. Small systems have also been tuned up to run faster. Unfortunately those speed increases are usually very hard won, and typically of relatively small proportions. The performance increases that come from simply running faster are often just not up to the tasks at hand.

So system designers have done the same thing you and I did as kids, they've gotten more mowers ; multiple processors working on the same task until it is done. Of course, the practicality of this solution hinges upon the practicality of breaking the task into independent sections. Yards were easy for us to divide. Dividing processing tasks into independent tasks has been the key to what is referred to as "multi-processing".

Details of the various approaches used in multi-processing certainly go beyond the scope of this paper, but they can be broadly divided into three main groups of approaches (as shown in figure 1) ; a "linear" or "pipelined" approach, a "clustered" approach or a "matrix" approach. In the linear setting, data moves through the system from one point to the next, with more data coming in behind it. The flow is undirectional and often continuous. Digital signal processing often times fits this model.

Figure 1 : Multi-processing Models
(A) Linear, (B) Clustered,
(C) Matrix.



The clustered and matrix approaches are similar to the extent that data flow may not be unidirectional in nature. Bi-directional links between the processing elements are required. Of course these architectures each lend themselves to solving different kinds of problems. The clustered approach fits well into a master CPU - smart peripheral situation, the oldest of the multi-processing applications ; while the matrix architecture is finding its way into complex calculation applications.

In any event, what all of these multi-processing approaches have in common is the requirement that processors talk to one another ; that, in itself, turns out to be a thing easier said than done, which is why dual port RAMs and FIFOs have become so popular in the last few years.

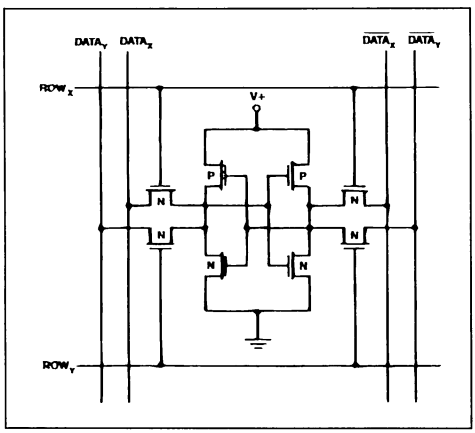
SGS-THOMSON BiPORT MEMORY DEVICES

Make no mistake, systems have been using FIFOs and dual port RAM for a long time. But with the exception of some rather low density devices, monolithic dual port RAMs and FIFOs are only beginning to become common. While others have wrestled conventional architectures into monolithic devices and come up with shift register based FIFOs and time-multiplexed dual port RAMs, SGS-THOMSON decided that a new memory cell design would make the whole affair much simpler.

The schematic of a BiPORT memory cell is shown in figure 2. The use of a BiPORT cell array allows simultaneous access to the memory array from two ports. A completely separate set of lines from the row decoders and data lines to and from the column I/O circuits, wired in parallel with the original set, guarantees that there cannot be any row or column contention, even if both ports are writing at once.

But even the use of BiPORT memory cells does not allow simultaneous read/write access to a given cell without a chance of error. SGS-THOMSON BiPORT FIFO prevents such an error by halting on full or empty conditions. The BiPORT RAM provides the user handshake logic ; but more on that later.

Figure 2 : A Full CMOS BiPORT RAM Cell.



FIFO BASED PROCESSOR LINKS

One of the beauties of many multi-processing applications is that addresses often become superfluous. There is data to be moved from one place to another. The data is contained in a defined, finite packet suitable, at least for the time it is in transit, for treatment as a sequential data stream. In short, an application just begging for a FIFO.

The MK4501 BiPORT FIFO

The MK4501 is a large, high performance FIFO that utilizes a memory based architecture and a memory matrix constructed with BiPORT memory cells. Refer to the device block diagram in figure 3.

The MK4501's 512 x 9 organization and memory-like interface (see figure 4) make it a good fit for many applications. Nevertheless, for those applications requiring more width or depth, the device is designed for easy expansion, with no performance penalty (such as an extended "fall-through" delay). Two status flags are available, Empty and Full, each of which remain active (low) whenever the stated condition exists.

Interfacing the MK4501

Although a multitude of techniques have been used to interface with the MK4501, the following three approaches cover a great many of the existing requirements.

A RAM-Like FIFO Interface

First of all, the MK4501 can be mapped into a processor's address space with conventional address decoding just like anything else. The user has the option to map it into one or many locations ; that is to say, the FIFO could be reached at any of a range of addresses.

Mapping the FIFO into a range of addresses allows the use of block-move transfers normally used with ordinary RAM. Users seem evenly split between whether or not to use the flags. Some applications that assure the FIFO will never empty or fill, others check the flags before every operation. Both approaches are useful depending upon the constraints placed upon the design (reference figure 5).

Peripheral Interface

Parallel ports are also used to control FIFOs. One reason is that ports generally have the ability to manage an access failure gracefully. If the user attempts illogical operations, such as read when empty or write while full, the MK4501 ignores the command. In refusing to write while full, the device protects the data that has been stored but not yet read. This sort of aborted attempt cannot occur when talking with memory. Memories always do what they are told ; peripherals, however, do not. Consequently parallel ports must be prepared to make adjustments whenever they encounter a "not ready" peripheral.

The MK4501 can be adapted to the common two-wire handshake favored by the low density FIFO manufacturers and many parallel ports with a little bit of logic, as shown in the schematic in figure 6.

Figure 3 : MK4501 Block Diagram.

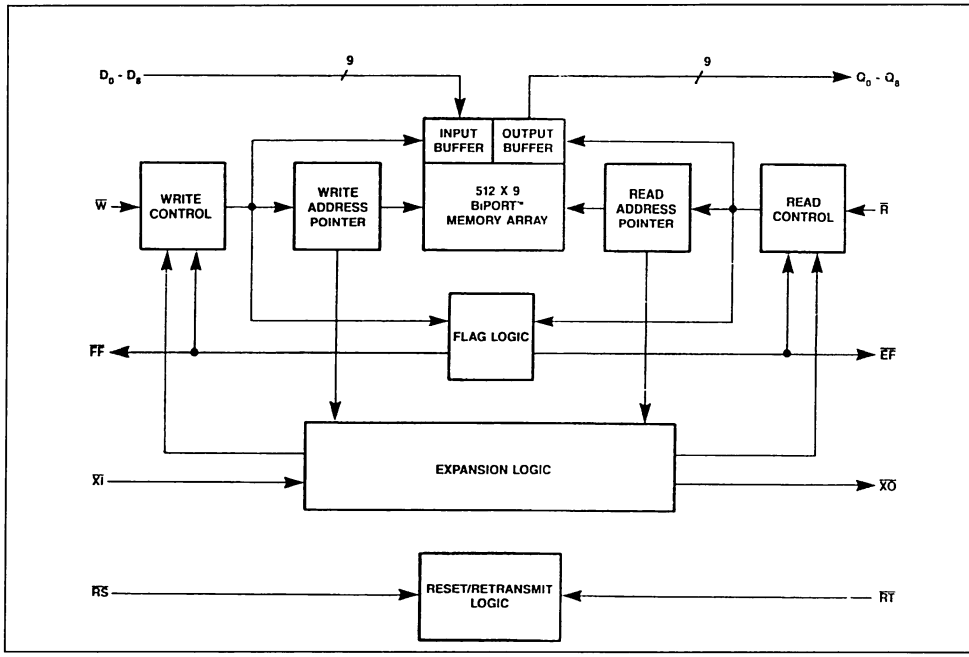
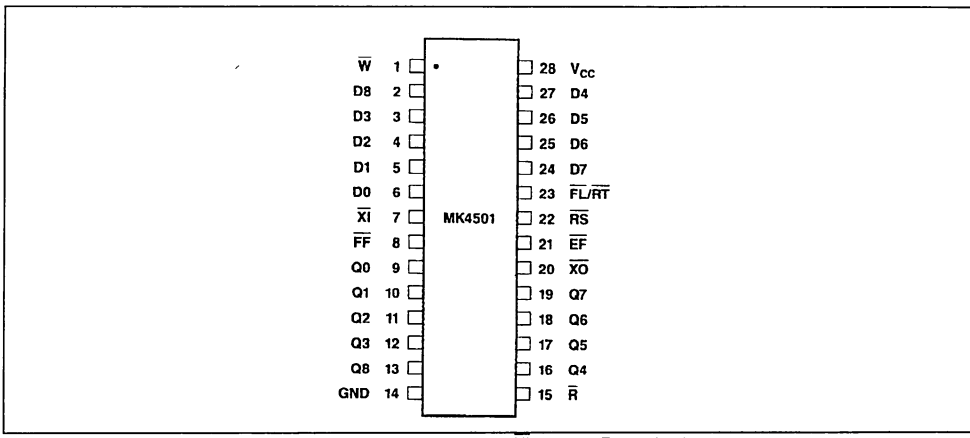


Figure 4 : MK4501 Pinout.



W = Write
R = Read
RS = Reset
FI/RT = First Load/ Retransmit
D = Data In
O = Data Out

XI = Expansion In
XO = Expansion Out
FF = Full Flag
EF = Empty Flag
Vcc = 5 Volts
GND = Ground

Figure 5 : Memory Mapped FIFO Interface Schematic.

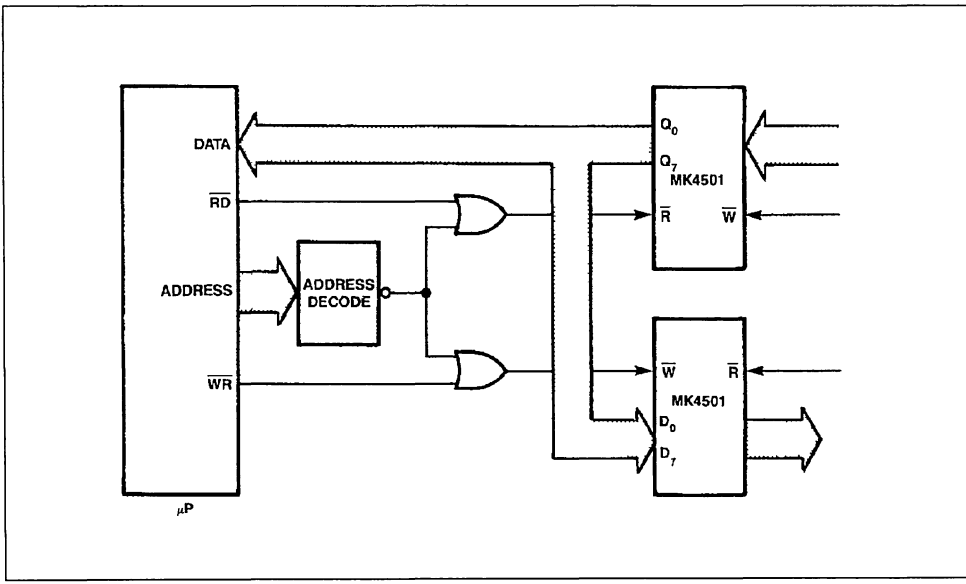
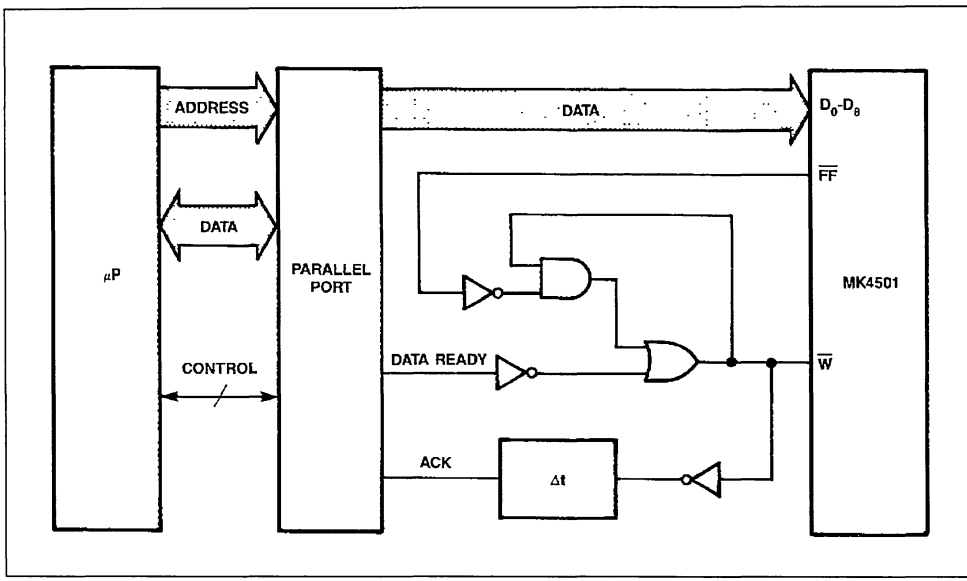


Figure 6 : Parallel Port Interface Schematic.



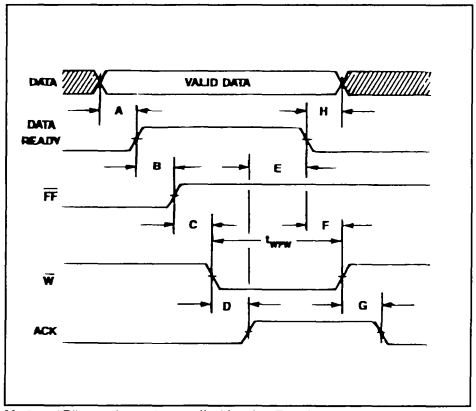
The logic performs three key functions. First, it blocks the port's attempt to dump data into the FIFO until the FIFO is ready, that is, until it becomes not full. Because the clearing of the Full Flag is under the control of the FIFO's read port, and because that port may remain inactive for an extended period of time, the delay between the parallel port signaling that valid data is available and the FIFO becoming ready, may be quite long.

Second, because the Full Flag will go active (fall) during the last write, feedback must be provided in the gating circuit to prevent it from "pinching off" the end of the last valid write pulse.

Third, the circuit provides the Acknowledge (ACK) or "shift-out" signal to the parallel port, indicating that data has been received. The delay block shown can be implemented any number of ways, but it must be selected to provide enough delay for the write pulse to the FIFO to meet specification.

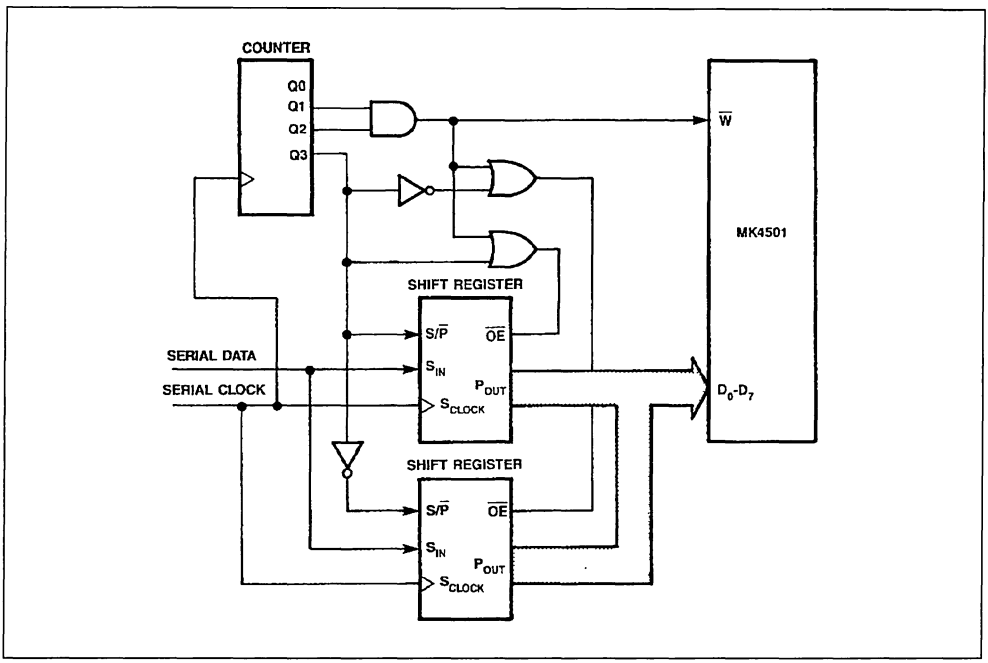
The schematic and timing shown illustrates transfers from a parallel port to the FIFO. The same circuit can be used on the read side of the FIFO to transfer data out of the FIFO to a parallel port. Naturally, multiplex/demultiplex chips can be used to implement bidirectional simplex links, or dual FIFOs can be specified for full duplex links.

Figure 7 : Parallel Port Interface Timing Diagram.



Note : "B" may be ∞ (controlled by the Read Port).
"D" equals Δt in schematic selected to make D + E + F Greater than or equal to the FIFOs minimum Write Pulse Width (t_{WPW}).

Figure 8 : Serial-to-parallel Interface Schematic.



Serial Bit-Stream Interface

Thus far this note has only alluded to the speed of the MK4501. At this point in time, catalog parts are available with cycle times as short as 80ns (12.5MHz) ; that is not, however, the device's maximum data rate. Remember that the MK4501 is a nine bit wide device. Doing a serial to parallel conversion allows the device to operate with serial bit streams running up to nine times faster ; at 112.5MHz. Of course that is pretty tricky.

A much slower and simpler implementation is shown in figure 8. Even running at 30MHz, maximum frequency for ALS-logic, the 33ns serial cycles driving a 8-bit shift register allow a 266ns parallel load cycle to the FIFO. It may interest you to know that the slowest MK4501 we sell accesses in 200ns and cycles in 235ns. Performance like that makes the MK4501 an excellent candidate for all kinds of serial applications, particularly disk controllers and LAN interfaces.

Figure 9 : Serial-to-parallel Timing Diagram.

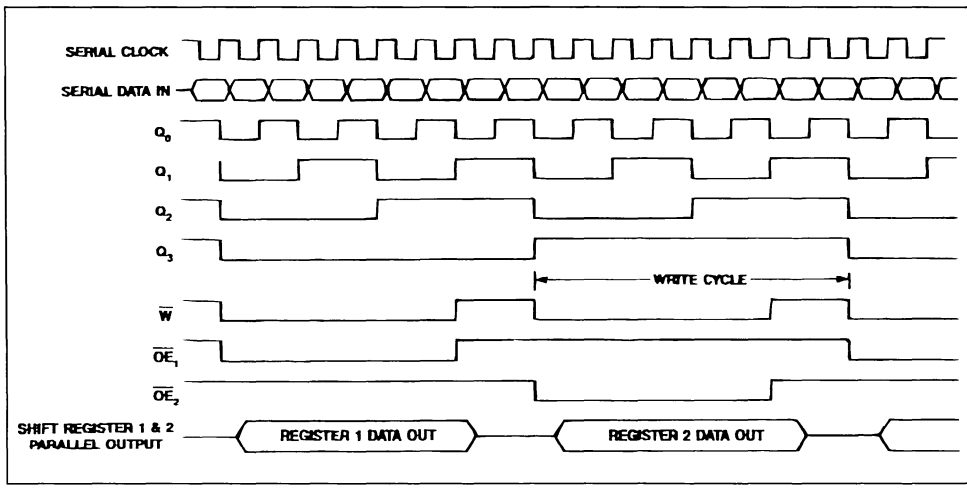


Figure 8 is simplified representation of a circuit that employs conventional TTL logic, a divide by 16 counter and two serial-to-parallel shift registers with three-state outputs. The counter is working as a state machine, both managing the two shift registers and generating the proper write pulse width and write pulse recovery timing for the FIFO, as shown in the timing diagram.

Using two shift registers and alternating between the two of them eliminates the need to unload a single shift register into a latch within a single serial clock cycle. Other functions that would need to be implemented in an actual design would include the reset logic and appropriate preload-at-reset circuitry to get the whole thing kicked off in sync.

Again, only the Write Port circuit and timing are shown because they apply equally to the Read Port.

DUAL PORT RAM PROCESSOR LINKS

Though they may be close, FIFOs are not the be all and end all of processor to processor links. Dual port RAMs are also a very popular choice. While the FIFOs offer many advantages in high performance links, dual port RAMs offer good speed and simultaneous bidirectionality in a single package. When those characteristics are combined with the right features, a sophisticated single chip bi-directional link can become a reality. The MK4511 is designed for just such a link.

The MK4511 BiPORT RAM

The MK4511 dual port RAM contains a single 512 x 9 CMOS memory matrix that can be accessed simultaneously from both of the input/output ports (reference figure 10). Dual port operation is achieved through the use of a memory array composed of BiPORT memory cells. Each memory cell is accessible from both ports at all times.

Pin count is kept low through the use of address/data multiplexing. This technique is being used on advanced microprocessors and other devices to keep pin counts and package sizes down.

The MK4511 incorporates all functions required for dual port operations, including software controlled

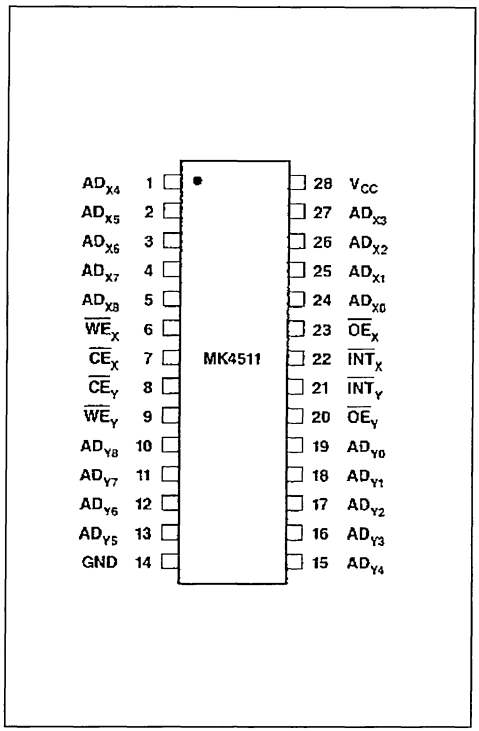
interrupt outputs. Use of the interrupt outputs is optional, allowing both polled and interrupt controlled applications.

Arbitration vs Handshake Routines

Unlike other dual port RAMs, the MK4511 allows unrestricted access to all locations at all times. There is no internal arbiter to push out an access when a collision occurs. Some have suggested that the lack of an arbiter is problematic in a dual port RAM design. However, that is only true in a limited group of specialized applications.

Arbiters are useful in applications where any given memory location may be written by one port and written or read by the other simultaneously. Such applications require either the old data or the new data, but they are unique in that they, by definition, cannot care which. There is no mechanism within the RAM to signal the processor which piece of data was stored or read. The arbiter simply assures the user will get one or the other, instead of a scramble of both.

Figure 10 : MK4511 Pinout .



Relatively few processor to processor links can operate under those assumptions. Most need to know when a valid message is in the mailbox and when it has been picked up. So, in most cases, not only is an internal arbitration circuit of no real benefit, extra logic must be provided by the user to accomplish the required handshake. The MK4511 is, to date, unique in that it provides the logic required to implement a variety of polled or interrupt driven handsake routines under software control.

The first byte of the memory array serves as the interrupt register for the X port. Addressing the first byte from Port X additionally accesses Port X interrupt logic. The last byte serves as the interrupt register for Port Y. Addressing the last byte from Port Y additionally accesses Port Y interrupt logic.

The lower three bits of each byte written to the interrupt registers are the ones routed simultaneously to the interrupt logic. The interrupt logic consists of three flip-flops per port that serve as the interrupt Request/Cancel flag, interrupt output Enable/Disable flag, and interrupt Acknowledge/Ready flag.

Message Passing With a MK4511

The following narrative description of a mailbox transaction should illustrate how the MK4511's onboard interrupt logic can be used :

Allocate pre-defined blocks of memory to each port. Each port may write only to its assigned memory block, preventing Port X and Port Y attempting to load their messages into the same area.

Write the message to be passed into the Port X message area. When finished, read the Y port Acknowledge/Ready flag (see figure 11). If ready, request an interrupt on port Y by writing a 1 to the X port Request/Cancel flag.

Next, acknowledge the interrupt to Port Y by writing a 1 to the Acknowledge/Ready flag on Port Y. Begin reading the message via Port Y. The Acknowledge should not be cleared to Ready until after the message has been read.

The X port will need to check to see that the message was received. The X port can poll the Y port Acknowledge/Ready flag or, the Y port can signal ready by making an interrupt request of its own.

Interfacing the MK4511

Use of the MK4511 is not restricted to Address/Data multiplexed processors. A simple state decoding circuit, together with a three-state driver and a transceiver, allow the MK4511 to be used with any non-multiplexed processor, as shown in figure 12.

Figure 11 : The MK4511 Handshake Logic.

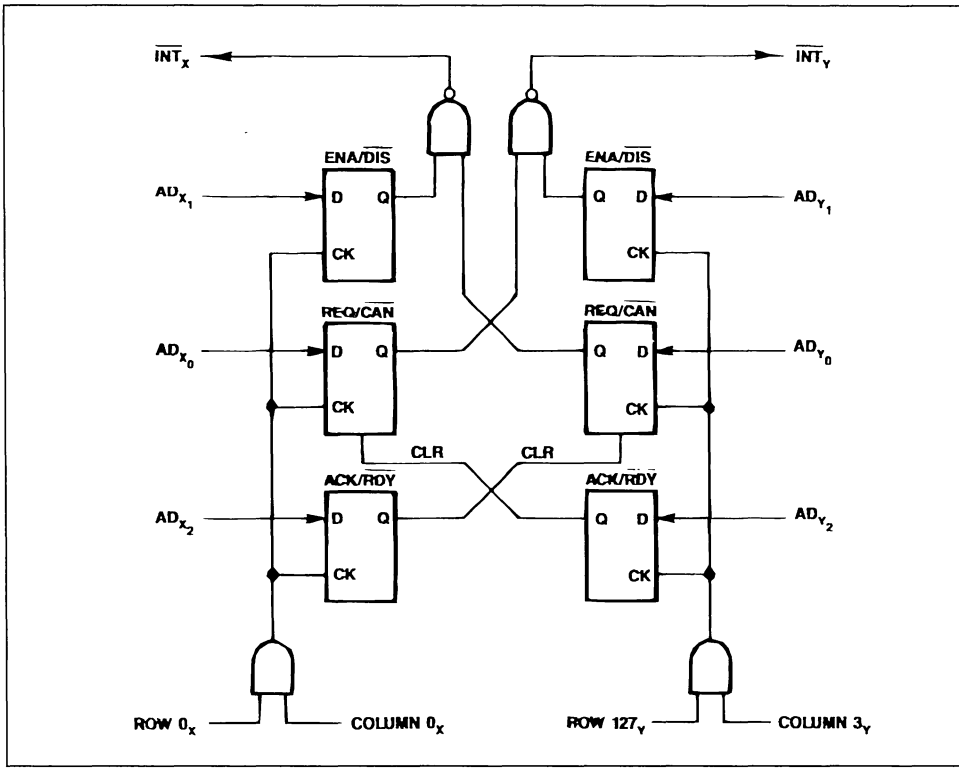
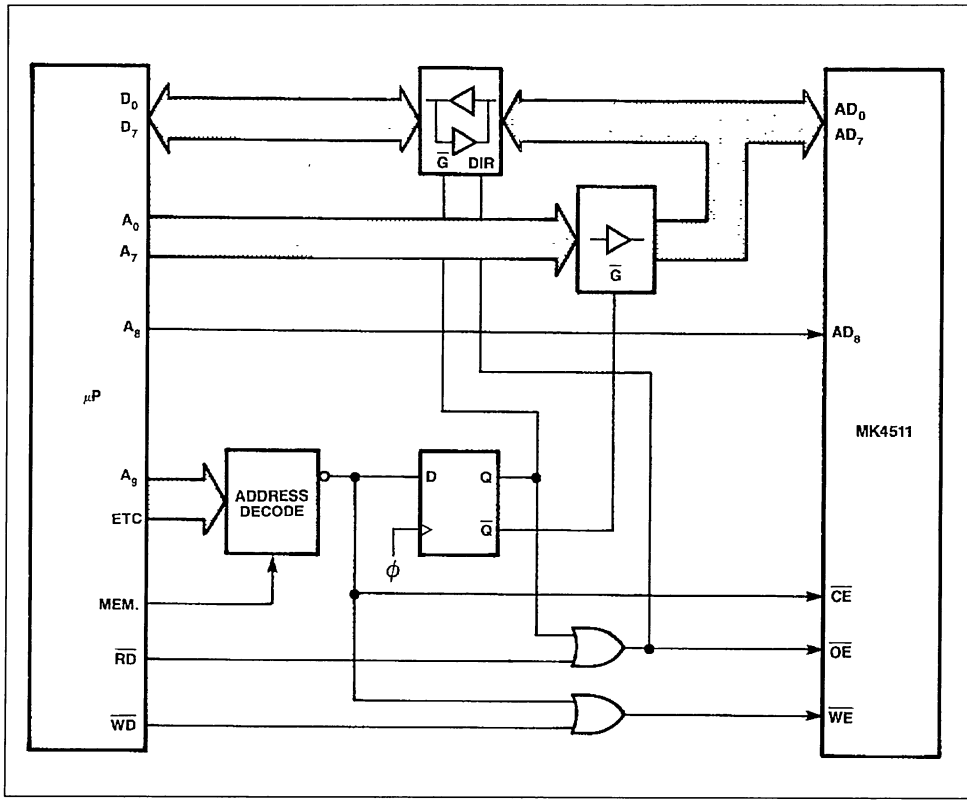


Figure 12 : The MK4511 and a Non-multiplexed Processor.



SUMMARY

Well, this paper has certainly run the gambit ; from serial transfers to address/data multiplexed busses ; from high speed burst transfers by a FIFO sitting in local memory to mid-performance bi-directional transfers managed by a single chip ; and that, is the central message of this Application Note. BiPORT RAMs and FIFOs are finding their way into a vast array of applications. They are being used effectively across a broad range of performance requirements. High speed users are designing with these devices because of their speed. Low speed users, often with performance requirements well below the slowest FIFO or RAM, are designing in the MK4501 and MK4511 for their density. All of this suggests that in the multi-processing arena, one should not dive into a data link design without at least considering the benefits offered by BiPORT RAMs and FIFOs.

SGS-THOMSON MICROELECTRONICS

APPLICATION NOTE

MK4501/03 BiPORT™ FIFO WIDTH EXPANSION TO AVOID WORD SKEW

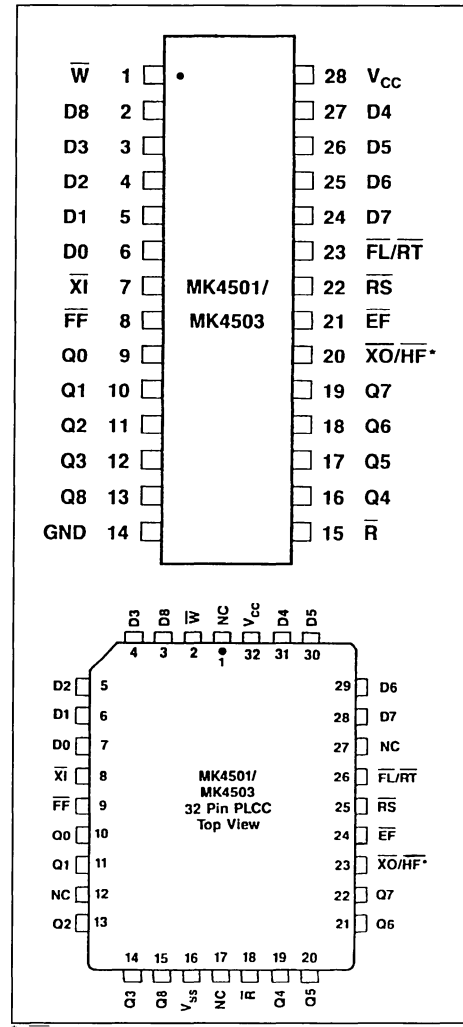
By Rick TUTTLE

INTRODUCTION

The MK4501 and MK4503 are high density BiPORT CMOS FIFOs (First-In-First-Out Memories) using SGS-THOMSON's industry standard pinout. The MK4501 consists of a 512 x 9-bit organization, with two status flags : EMPTY ($\overline{EF}$), and FULL ($\overline{FF}$). The MK4503 is pin-for-pin compatible with the MK4501, but has a 2K x 9-bit organization, and includes an additional Half-Full status flag ($\overline{HF}$) (see figure 1). Both devices utilize a BiPORT™ RAM cell design which allows simultaneous and asynchronous writes/reads, thus avoiding the ripple-through delay times of conventional shift register based FIFOs. The MK4501 and MK4503 are expandable to any width and depth, as well as offering a retransmit capability. Retransmit is allowed in single device or word width mode. Depth expansion is achieved by simply connecting the $\overline{XO}$ pin of one device to the $\overline{XI}$ pin of the next device in a daisy chain fashion (refer to the data sheet for example diagrams). Width expansion is accomplished by providing the necessary data bus width (usually 16 to 32 bits), and connecting control inputs of the FIFOs together for synchronized operation.

The design of MK4501 and MK4503 incorporates internal logic such that when the $\overline{FF}$ is set ($\overline{FF}$ = low), further writes are inhibited. The same is true when the $\overline{EF}$ is set ($\overline{EF}$ = low), further reads are inhibited. There is an arbitration condition where either status flag may be set while receiving an asynchronous operation on the other port that would clear the status flag (i.e. read when full, or write when empty), immediately followed by another set operation (write-when-full, or read-when-empty), where the set-before-clear operation would be indeterminant. Referring to the MK4501 or MK4503 data sheet, we find when doing a write within t_{WPI} before the $\overline{FF}$ is cleared after a valid read, that the write is considered indeterminant. Once the $\overline{FF}$ is cleared, t_{FFW} must be satisfied to assure a successful write cycle. The same is true when the $\overline{EF}$ is set where a read cycle, proceeded by a valid write to clear the $\overline{EF}$, is considered interminant within t_{RPI} ; but will be valid if t_{RFF} is satisfied once the EF is cleaed ($\overline{EF}$ = High).

Figure 1 : Pin Connections



* HF available only with the MK4503

CIRCUIT DESCRIPTION

In figures 2 and 3 we have presented an application concept showing an example of using either the MK4501 or MK4503 for width expansion of 16 to 32 bits. (Any unused Data-Inputs pins should be tied low). These diagrams also depict that once the FIFO array is full ($\overline{FF}$ = low), further writes will be synchronously disabled. The same is true when empty ($\overline{EF}$ = low), further reads will be synchronously disabled. This type of application is only required where system designs allow write-when-full, or read-when-empty, where complete read/write control cannot be immediately obtained for a defined condition set, i.e. when the FIFOs become full ($\overline{FF}$ = low). From the previous discussion regarding indeterminant operations, one can see that in word width expansion, work skew generation is possible where the validity of the operation is dependent upon the flag-to-clear access time. Actual device flag access times could vary tens of nano-seconds producing an arbitration condition, and thus allowing valid operations to some of the devices in the array, while being indeterminant to others. This is especially true where cycle times of 120ns or greater are being used. Therefore, we have added a small amount of logic to our circuit in order to avoid word skew generation (see figure 3), and thus ensure that indeterminant reads or writes for the FIFO array never occur. This will allow all devices to remain in a synchronized operational mode.

CIRCUIT OPERATION

The schematic diagram in figure 3 displays an example for width expansion of 16-bits using either the MK4501 or MK4503. (Of course wider word widths are just as easy). In this schematic, the D-type transparent latch on either side provides a latched status flag with true logic to the OR gate. The transparent latch (7475 series) is recommended versus a clocked flip-flop to provide smoother logic transistions to avoid glitching on the OR gate inputs during control line transistions. If either status flag is latched (LFF = high, LEF = high) the corresponding OR gate is disabled providing a high level to $\overline{R}$ or $\overline{W}$. The external control signals, $\overline{W_X}$ or $\overline{R_X}$, are prevented from passing through the OR gate, thus further read or write operations are inhibited. The AND gates provides a composite clear flag status (logic one), and any FIFO in the array can produce a set condition for $\overline{FF_X}$ or $\overline{EF_X}$ (logic zero). For simplicity, the $\overline{FF}$ and $\overline{EF}$ have been labeled "U" or "L" to denote upper or lower device.

Further explanation is given in figures 4 and 5. Figure 4 displays the timing for write-to-full then read, followed by successive read/write cycles. Reset is presumed, followed by consecutive write cycles until the FIFO array is full. Once $\overline{FF}$ is asserted low, $\overline{FF_X}$ will also go low after the AND gate propagation delay time. This is turn will cause LFF to go high via $\overline{QW}$ when CW is high through the D-type transparent latch, resulting in a synchronized write inhibit as $\overline{W}$ will remain high through the OR gate until $\overline{FF_X}$ is cleared ($\overline{FF_X}$ = high) after a valid read. Figure 5 displays read-to-empty then write, followed by successive write/read cycles. The same exercise for figure 4 can be performed by the reader in figure 5 by simply reversing roles of write for read, and full for empty.

There is a special case that could be encountered when the composite status flags are in a set ($\overline{FF_X}$ or $\overline{EF_X}$ = low) to clear transistion. For this condition, the set-up time to the D-type latch may be violated in reference to data in and CW going high via external control lines ($\overline{R_X}$ or $\overline{W_X}$). The arbitration results would be : (1) the existing operation would be valid to the FIFO array - this is a don't care situation since the FIFOs' status flags are cleared and further operations are valid, or (2) the existing operation will be ignored since set-up times were no met, resulting in a dummy cycle which is expected.

CONCLUSION

Even though the MK4501 and MK4503 will inhibit reads while empty, and writes while full, the simple logic application concept would avoid any possible word skew probabilities in the word width expansion mode. This approach could be extremely helpful for designs where exclusive read/write processor control is not allowed within the time frame needed for a defined set of conditions. It should also be noted that while this circuit will successfully ignore write-when-full and read-when-empty in a synchronized fashion, expected data during these operations will be lost if not stored in some type of temporary buffer or memory medium.

Figure 2 : Block Diagram/Equivalent Circuit.

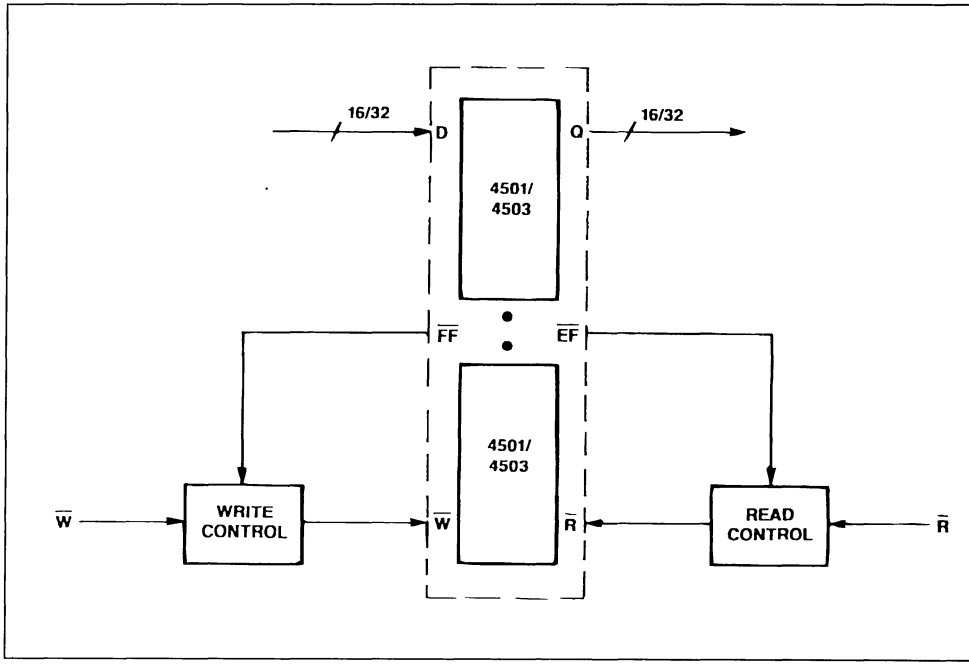


Figure 3 : Schematic Diagram.

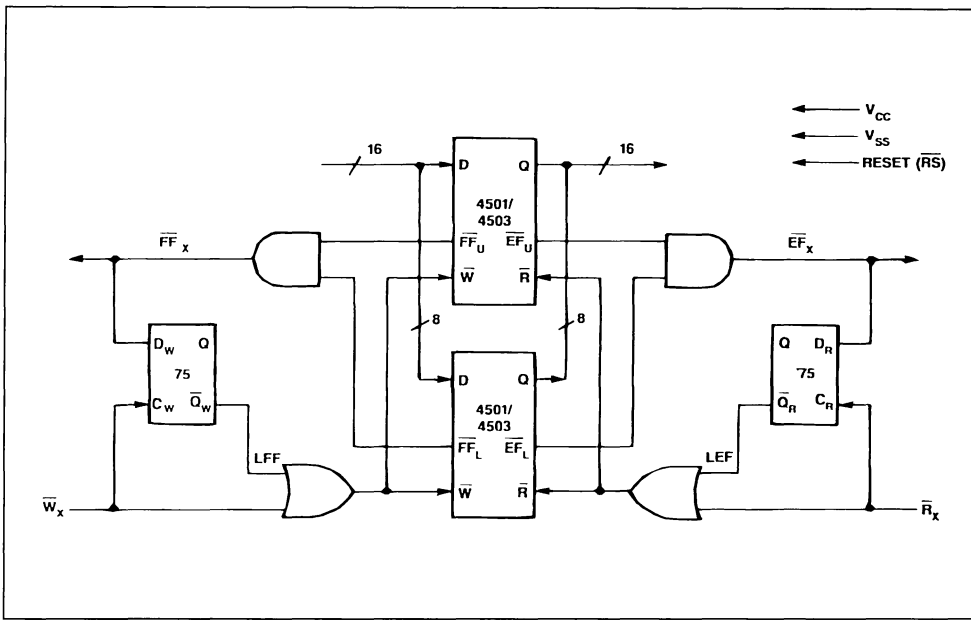


Figure 4 : Write-to-Full/Read-Write.

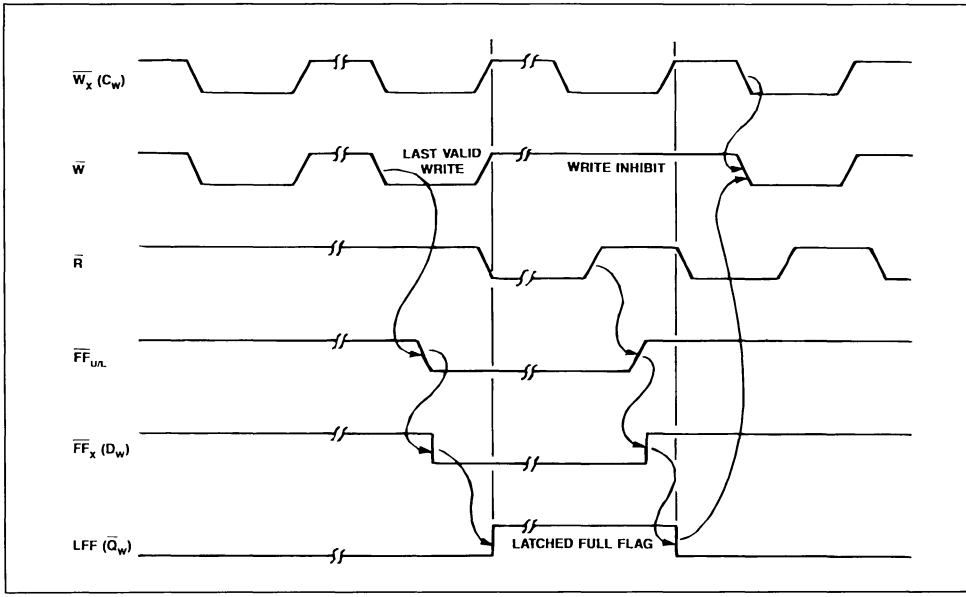
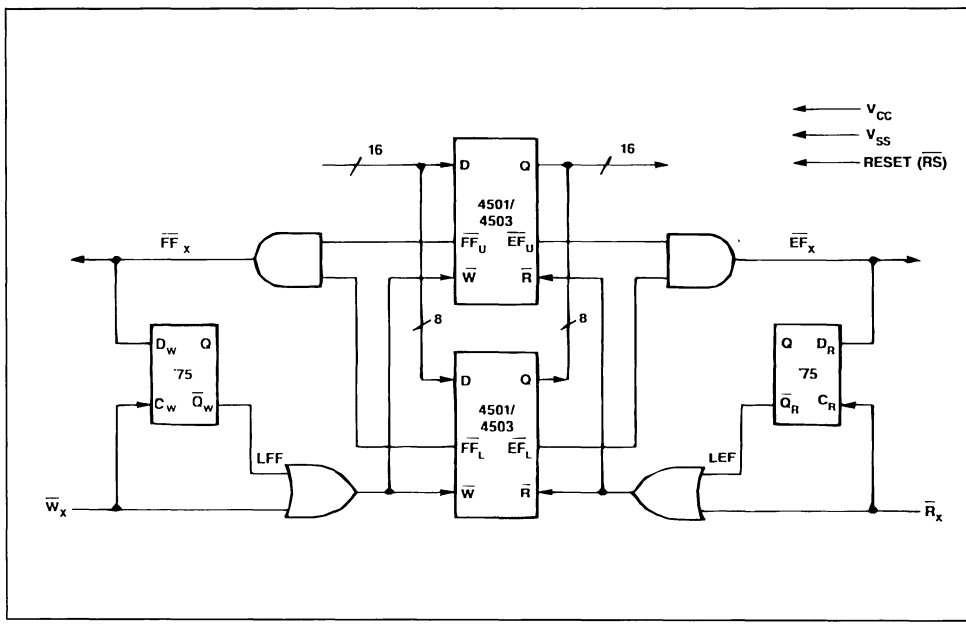


Figure 5 : Read-to-Empty/Write-Read.



USING THE MK4501 BiPORT™ FIFO AS A DIGITAL DELAY

By D. CHAPMAM AND R. TUTTLE

INTRODUCTION

One of the most significant features of the MK4501 FIFO (FIRST-IN-FIRST-OUT Memory) is that it does not suffer from the "ripple-through" delay that has been the common denominator between most other FIFOs. To avoid excessive delays, earlier FIFOs were made no larger than 128 bytes deep, in contrast to the MK4501 which is 512 bytes deep. Furthermore, the MK4501 can be depth expanded in 512 byte increments to any depth desired without any speed penalty. There is, however, a group of folks who have been using the ripple-through delay of conventional FIFOs to their advantage. They have been building data delay devices. The requirement for data delays most often surfaces in digital signal processing applications ; video, audio, etc...

CIRCUIT DESCRIPTION

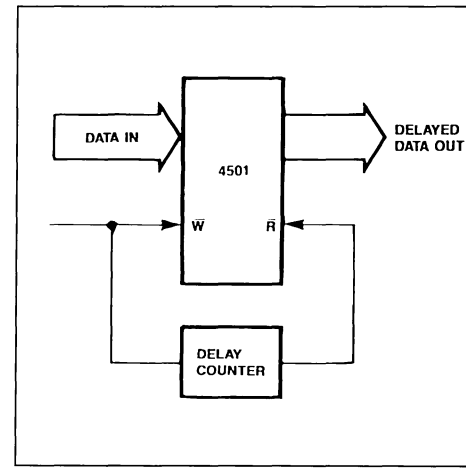
Even through the MK4501 does not exhibit a ripple-through delay characteristic, it is a very attractive device for delay applications because of its depth. With the addition of a counter, the MK4501 can serve as such a delay. (See figure 1 for a block diagram). After reset, the counter counts the desired number of write pulses (Delay time = Write cycle time x count), and then gates the write pulses into the MK4501's read ($\overline{R}$) input. From then on, the data is read out of the FIFO at the same rate it comes in (see figure 4).

In such an application the MK4501 should never be allowed to fill, that is the Full Flag should not be allowed to become active LOW. To prevent the Full Flag from being asserted LOW, the first read must occur before or simultaneously with the next-to-last write (#511). The circuit shown in figure 3 allows the maximum delay that a single MK4501 can provide without going full. An example of the maximum read delay timing before $\overline{FF}$ is asserted can be referenced in figure 2.

CIRCUIT OPERATION

In figure 3 the flip-flop to the far left is the LSB of a 9-bit counter formed by the flip-flop and the 8-bit '592 counter. Wiring the Q output to the active low counter enable ($\overline{CCKEN}$) has the effect of presetting the 9-bit counter to 1 at reset. Inasmuch as maximum count on a 9-bit counter is 511, presetting the 1 causes the counter output (RCO) to assert on write #510. The RE' (Read Enable Set) signal goes HIGH as soon as the counter output (RCO) and the inverted write signal ($\overline{W}$) both go LOW. This will be during the write recovery time between write #510 and #511. RE (read enable) is then available at the beginning of write #511, which when NORed with the write pulse produces read #1 in sync simultaneously as the result of the MK4501's BiPORT™ cell design. This action ensures that the $\overline{FF}$ will not be asserted active LOW. Each additional write results in a read producing a constant data path. Holding the $\overline{W}$ input HIGH will prevent further data flow.

Figure 1 : Block Diagram - 4501 Data Delay.



CONCLUSION

In this application example, the counter can be preset to higher values through the '592's parallel inputs (A-H) to shorten the delay. In fact, programmatic control of the preset would allow a variable delay function to be implemented with this circuit. This would allow data delays between 2 and 510.

Figure 2 : Write/Read Timing - Latest Possible First Read to Prevent Full Flag Assertion.

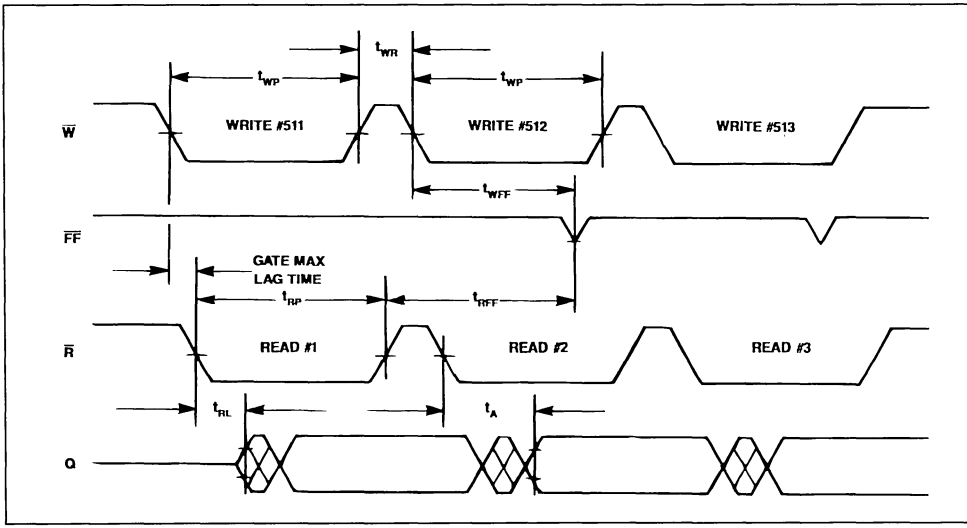


Figure 3 : Delay Counter Schematic.

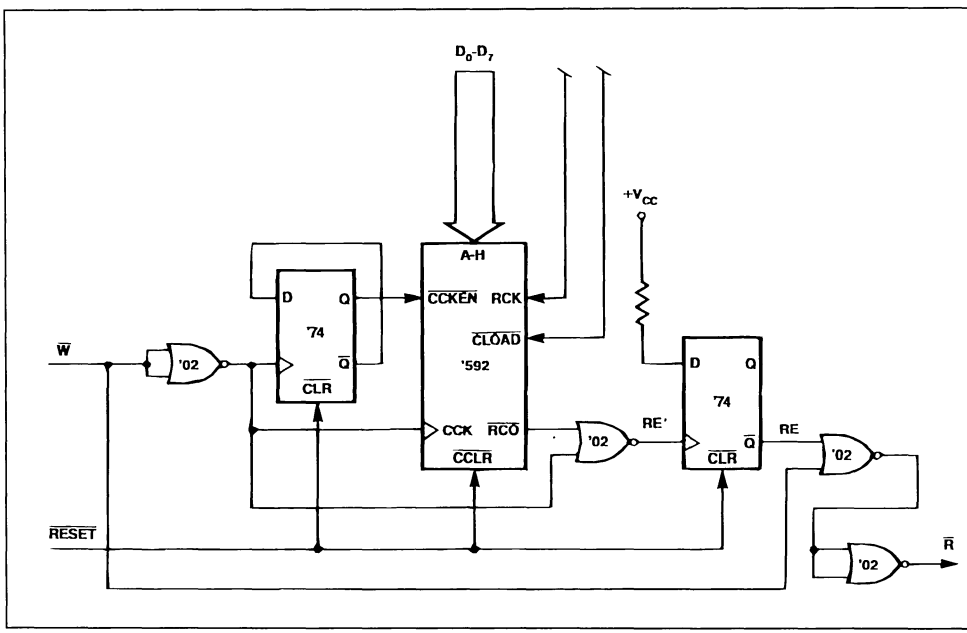
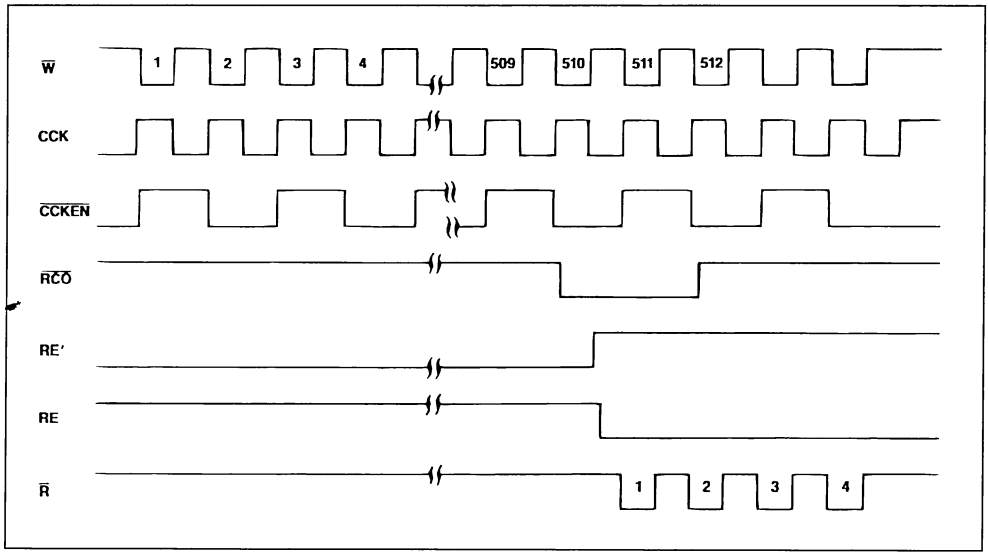


Figure 4 : Timing Diagram for Digital Delay Circuit.



Note : This is a test circuit, and is intended only to illustrate an application concept.

THE MK4501/03 BiPORT™ FIFO USING BIDIRECTIONAL RESET

By RICK TUTTLE

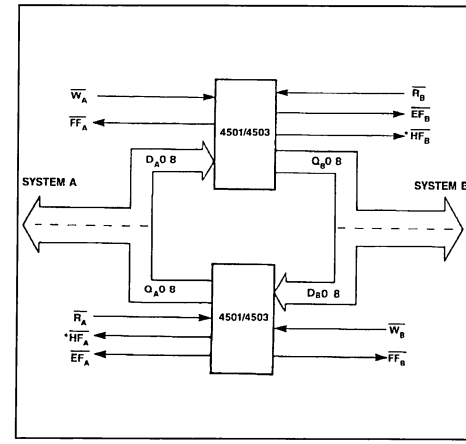
INTRODUCTION

The MK4501 and MK4503 are high density CMOS FIFO (First-In-First-Out) BiPORT memories from SGS-THOMSON Microelectronics. Its family of BiPORT memory products has gained recognition world-wide, and already claims the industry standard in high density FIFO BiPORT memories with the introduction of the MK4501 in 1983. As the family of BiPORT memory products increases, so do the design applications. This produces more demand and uses for the unique static BiPORT memory cell arrangement, and thus forces more functions, higher speed, and more density with each generation of memory devices. With the complexity of today's multi-processor designs, comes the need for high density and high speed shared memory. Some applications call for passing data from one processor to the other processor in a mailbox type fashion. This can be accomplished by using a Dual-Port SRAM, such as SGS-THOMSON MK4532A/MK4542A or MK45DP8. These are CMOS low power, high speed 2k x 8 Dual Port SRAMs with different data path organizations. The MK4532A/MK4542A allows passing data between two 8-bit processors, while the MK45DP8 allows data to flow from a 16-bit processor on the left port to an 8-bit processor on the right port. High density FIFOs, such as the MK4501 and MK4503 which have 512 x 9-bit and 2k x 9-bit organizations respectively, can also pass data from one system processor to the other, but do not allow random access Read/Write functions on the same port as do the Dual-Port SRAMs. First-In-First-Out memories are typically used as data rate buffers for writing data in one side or port at a given rate, and reading or retrieving that same data at a different rate on the other port in a first-in-first-out fashion.

BIDIRECTIONAL BUFFERS

There is a FIFO application that serves somewhat like a Dual-Port SRAM function when using a bidirectional FIFO arrangement. This type of arrangement, however, has no worries of address match arbitration as with the Dual-Port SRAMs. This is because the design of the FIFO memory does not produce contention arbitration between Read/Write functions and physical address locations. Thus bidirectional FIFO arrangements allow asynchronous

Figure 1 : Bidirectional FIFO Mode.



* Available for the MK4503 only.

and simultaneous Read/Writes operations from either processor. Referring to figure 1, designs requiring bidirectional buffering between two systems (each system capable of Read and Write functions) can be achieved by pairing FIFOs in a suitable manner as shown (this applies for both the MK4501 and MK4503). Status flags must be monitored by each system such that the $\overline{FF}$ status is viewed where $\overline{W}$ is used, and the $\overline{EF}$ status is viewed where $\overline{R}$ is used. When using the MK4503, the $\overline{HF}$ function may be viewed from either control side depending on its reference of intention ; that is, whether sensing Half Full or Half Empty.

CIRCUIT DESCRIPTION

High density FIFOs are well suited for bidirectional applications to provide a unique handshake between two systems. Data can be manipulated between the two systems easily and efficiently. This type of arrangement can allow the systems to share common data in a pipeline fashion, or allow passing data from one processor to another for data manipulation, and having those results relayed back to the originating processor in a FIFO sequence. However, one difficulty may arise where a Master Reset to the bidirectional FIFO arrangement may not

be determined by both processors at the same designated time. For example, if one processor initiates the Reset signal, then it can also determine the state of its Read and Write controls. The other processor, however, would continue Read or Write functions until an interrupt had been acknowledged. This would result in probable invalid Reset set-up times (t_{RSS}), or invalid Read (t_{RPW}) or Write (t_{WPW}) pulse widths to the FIFO device matrix. (As a brief reference, see figure 2. Further timing details are provided in the MK4501 or MK4503 data sheet).

In order to assure a proper Reset to the system's FIFOs, the Reset signal needs to be latched for at least the duration of its pulse width (t_{RS}), and be able to satisfy t_{RSS} for the $\overline{W}$ and $\overline{R}$ inputs. The system reset circuit must also assure proper enable pulse widths as t_{RPW} and t_{WPW} for either or both system processors. The functional schematics in figures 3 and 4 reveal two options to latch Reset, and ensure correct timing sequences. These circuits can be applied to all Read and Write controls, or only to the processor where the Read/Write controls are unpredictable during Reset. Whatever the case, each Reset is assured the correct timing on the Read/Write control signals to satisfy t_{RSS}, and Read or Write pulse widths.

CIRCUIT OPERATION

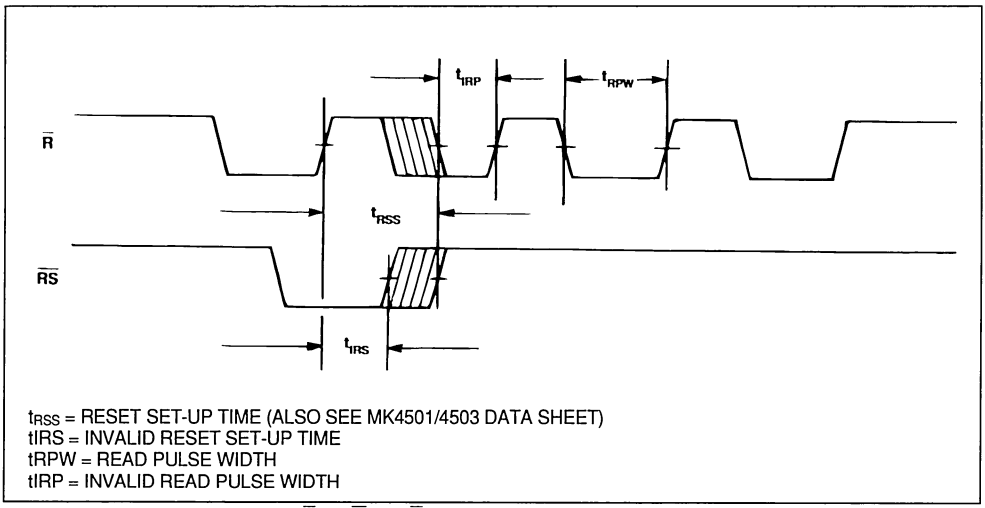
Referring again to figure 3, simply stated the inverted output of the NOR gate $\overline{R}$, will follow the Rx input as long as it is enabled, that is, when all other inputs to the NOR gate are LOW. The transparent D-type latch allows the D-input to propagate through determining the Q and $\overline{Q}$ outputs whenever Rx is HIGH. During Reset, when $\overline{RS}$ goes LOW, RS' immediately forces $\overline{R}$ HIGH, and thus disables further Reads to the FIFO during the Reset pulse duration. Meanwhile, when Rx goes HIGH at the end of the normal Read operation, Q will latch $\overline{R}$ HIGH until the next valid Read cycle. When a Reset occurs between Reads, then $\overline{Q}$ is latched HIGH keeping the NOR gate disabled after Reset until the next occurring valid Read cycle. This type of design allows the first Read after Reset to be enabled, as well as ensuring proper set-up times and Read/write pulse widths.

The circuit in figure 4 shows another method of latching Reset. Here the Q output of the clocked D-type flip-flop acts as an enable to the two input OR gate. During normal operation the Q output will remain LOW for each successive rising edge of the external Rx clock. This will keep the OR gate enabled, and thus allow $\overline{R}$ to follow Rx delayed by the OR gate propagation time. When a Reset occurs, strobing Preset LOW, then the Q output of the flip-flop would go HIGH, disabling the OR gate, thus forcing $\overline{R}$ HIGH. Further Reads from Rx would be ignored. This ensures that t_{RSS} is satisfied in reference to Reset and $\overline{R}$, even though the processor may attempt further reads before an interrupt is acknowledged. The main difference between this method and the previously discussed example, is that this method demands that the processor allow for a "dummy" Read or Write cycle after an initial Reset. In other words, the first Read to the circuit in figure 4 cannot be guaranteed due to possible arbitration in set-up times unless one dummy Read cycle occurs after Reset. (This is best viewed from the waveform example in figure 5). Now for instances where an interrupt after Reset produces a wait cycle, this condition has no consequence. The problem would be upon the initial start-up Reset where the first valid Read cycle may or may not be enabled, depending upon the rising edge of Reset in relation to the Rx clock input to the D-type flip-flop. This could result in set-up conditions as previously displayed in figure 2. Should a Reset always occur between Reads, as depicted in figure 6, then a dummy cycle would not be needed after Reset ; however, this is not the probable consistent scenario.

CONCLUSION

Bidirectional applications using high density FIFOs are easily and effectively implemented using either SGS-THOMSON's MK4501 or MK4503. By using a D-type latch coupled with a logic gate for Read or Write enabled functions, two examples have been presented to incorporate proper set-up times and Read/Write pulse widths for a Master Reset to the bidirectional arrangement. Individual system processors are not affected, and other control signal timing to the FIFOs can be ignored during Reset and at least one cycle after Reset. These example circuits could be used for each Read/Write control line, or only for the control lines considered unpredictable during a Reset operation.

Figure 2 : Valid Reset Timing.



t_{RSS} = RESET SET-UP TIME (ALSO SEE MK4501/4503 DATA SHEET)
tIRS = INVALID RESET SET-UP TIME
tRPW = READ PULSE WIDTH
tIRP = INVALID READ PULSE WIDTH

Note : Correct timing is required for both $\overline{R}$ and $\overline{W}$, only $\overline{R}$ is shown.

Figure 3 : Bidirectional Reset Using A Transparent Latch.

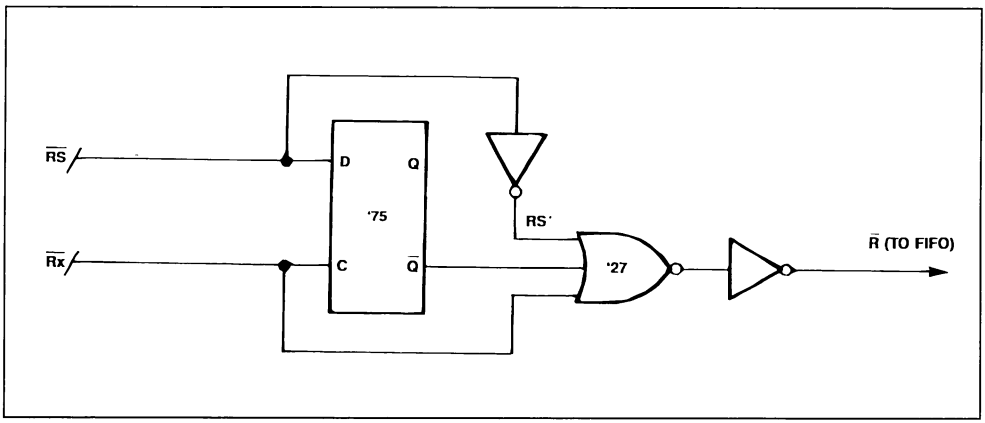


Figure 4 : Bidirectional Reset Using An Edge Trigger Latch.

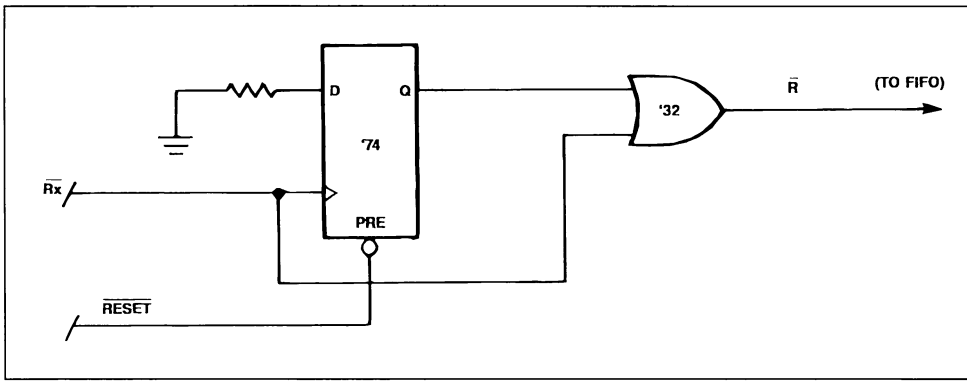


Figure 5 : Reset During Read.

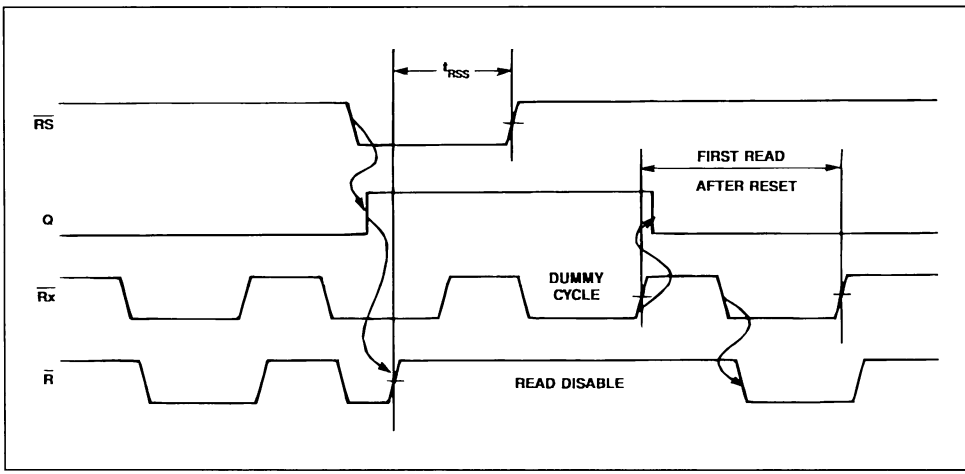
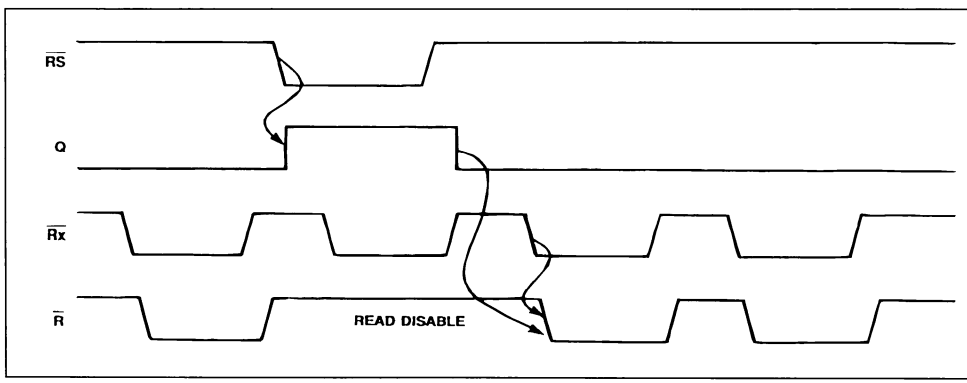


Figure 6 : Reset before Read.



THE MK4503 BiPORT™ FIFO 16-BIT TO 8-BIT CONVERSION

INTRODUCTION

When SGS-THOMSON Microelectronics introduced the MK4501 in 1983, it was the first high density FIFO with a BiPORT™ memory cell architecture. The MK4501 quickly became the industry standard with an organization of 512 x 9 bits, and included both an Empty and Full status flag. SGS-THOMSON has extended this technology to develop a device with four times the density - the MK4503. The MK4503 has a 2K x 9-bit organization, and includes the addition of a Half Full (HF) status flag, as well as the Empty and Full status flags. Its BiPORT™ RAM cell design allows simultaneous and asynchronous Write/Reads, and avoids the added ripple-through delay times of conventional shift register based FIFOs. As with the MK4501, word with and depth expansion is easily achieved by using the XI and XO pins. The MK4503 is pin-for-pin compatible with the MK4501, and thus can be used as a density upgrade in many applications.

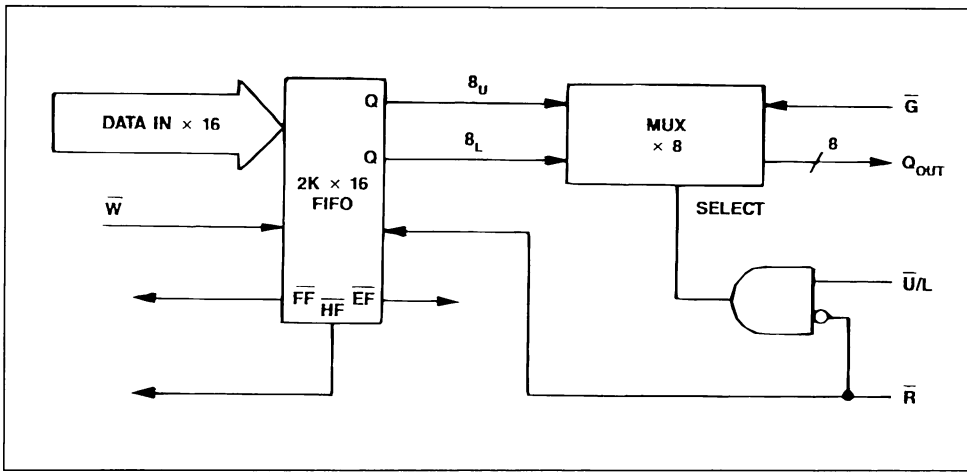
CIRCUIT DESCRIPTION

As shown in figure 1, the block diagram for this application concept shows an example of how to interface a 16-bit Microprocessor to an 8-bit peripheral. Due to the MK4503's architecture, which provides easy width expansion, we can interface two MK4503 FIFOs to a 16-bit Microprocessor for collecting and holding our data dumps. By using a small amount of additional logic, we can retrieve this data in consecutive 8-bit bytes. (The diagram in figure 2 shows the basic idea in an equivalent circuit). Using the two MK4503 FIFOs, we start with a 2K x 16 memory buffer as our input, and convert it to a 4K x 8 memory buffer output. The status flags will keep us updated to let us know when we are Full ($\overline{FF}$ = low), or Empty ($\overline{EF}$ = low). The Half-Full status flag ($\overline{HF}$ = low) will help to avoid those sudden unpredicted halts, for example, should there not be adequate block memory space available. We will also take advantage of the asynchronous and simultaneous Write/Read capability of the MK4503's BiPORT™ design.

CIRCUIT OPERATION

The schematic diagram in figure 3, shows that we will be writing 16-bit words designated as UPPER and LOWER 8-bit bytes, and then reading first the UPPER 8-bit byte, and then the LOWER 8-bit byte. As with all FIFO applications, we must start with RESET ($\overline{RS}$) to initialize the circuit. Remembering that upon reset our $\overline{EF}$ output goes active low, all Read cycles are ignored until the first Write cycle has been

Figure 1 : Block Diagram.



completed. Once a succesful Write has been performed, the $\overline{EF}$ output will go inactive high. The ideal operation would allow the Write count to remain at least one cycle ahead of the Read, thus avoiding $\overline{EF}$ from being asserted active. Referring again to figure 3, the A/B select to the data multiplexers via Q of the D-type flip-flop, alternates at the **end** (rising edge) of each Read to get ready for the next Read cycle. This avoids additional gate delay time, as well as providing a READ strobe during the full $\overline{R}$ pulse width. Therefore, after Write, and upon the first Read, we access the UPPER byte ($Q_{A\,0\text{-}7}$), and alternate thereafter between the LOWER ($Q_{B\,0\text{-}7}$) and UPPER byte with each consecutive Read. The flag status needs only to be taken from the LOWER byte FIFO since it will be Read last, and both are written simultaneously. Of course an (Empty - 1) function could be implemented on the 8-bit side by using the Empty Flag output from the UPPER FIFO.

In this application we have included data bus control with a fast external Output Enable ($\overline{G}$) on the multiplexer. The waveform timing diagram is referenced in figure 4. Since there will be a specific Q-valid access time from the enabled FIFO to the data inputs of the multiplexer, the $\overline{G}$ input can be tied to the $\overline{R}$ system input without any penalty. The timing diagram in figure 5 displays the typical access times to be considered. For example, t_{A1} is the combined access time of the OR gate plus Read access (t_A) of the MK4503. Further definitions are : t_{A2} = MUX Q-Valid access time, t_{A3} = MUX Q-Valid access time from $\overline{R}$ asserted low, t_{GLQV} is MUX $\overline{G}$ (Output Enable) access to Q-Valid, t_{OH1} = Q-Hold time of the FIFO, t_{OH2} = Q-Hold time of the MUX, and t_{GHQZ} is output Enable to High Z. It should be noted that t_{A3} is equal to ($t_{A1} + t_{A2}$), where the limiting factor is t_{A1} not t_{GLQV}, when $\overline{G}$ = R (see figure 5).

CONCLUSION

This implementation presumes that Read strobes will be halted when the lower MK4503 indicates itself empty. In this example, should the lower FIFO become empty, additional Read strobes will continue to toggle the select D-type flip-flop even though the FIFOs will not respond. Should Read-while-Empty compatibility be required, then additional logic will be needed to disable the select flip-flop when the lower FIFO is empty.

Figure 2 : Equivalent Circuit.

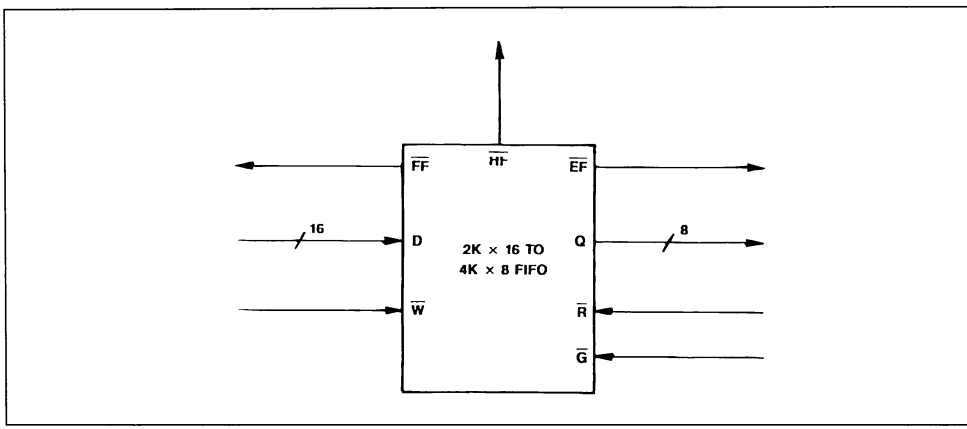


Figure 3 : Asynchronous 16-Bit to Asynchronous 8-Bit Schematic.

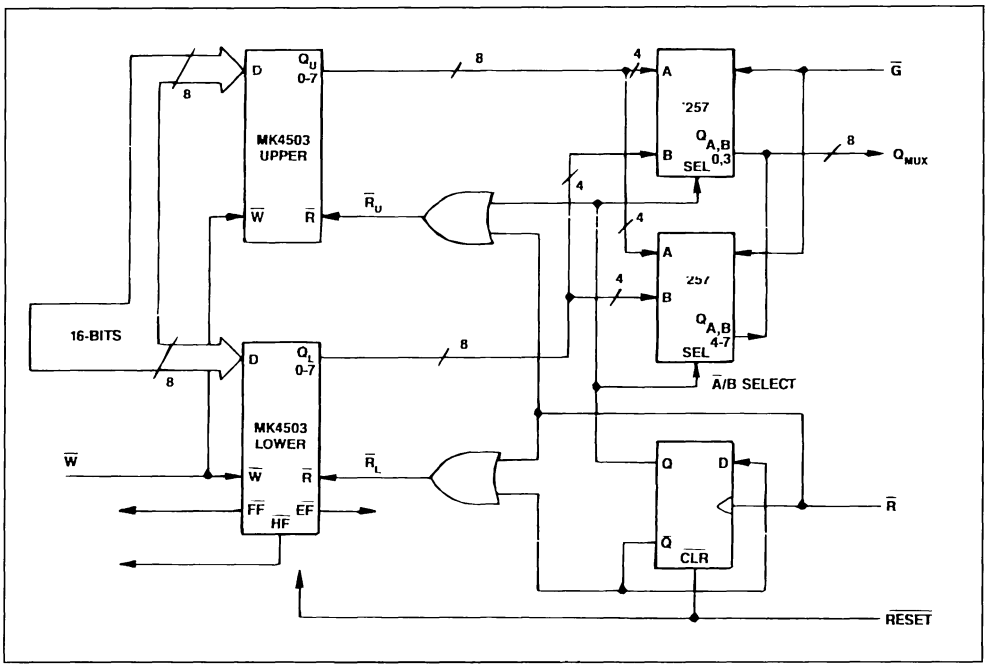


Figure 4 : Timing Diagram.

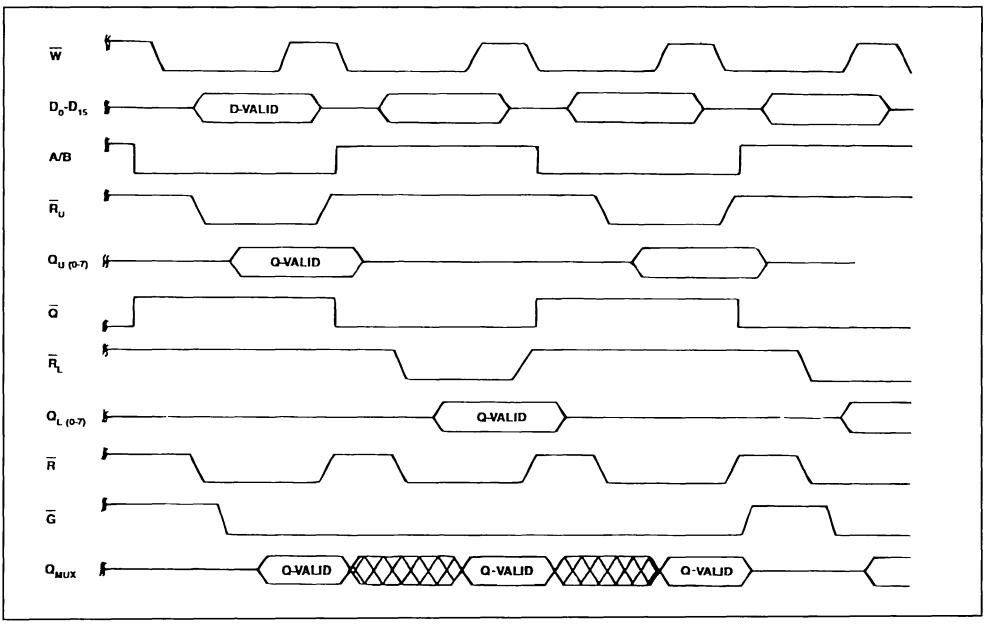
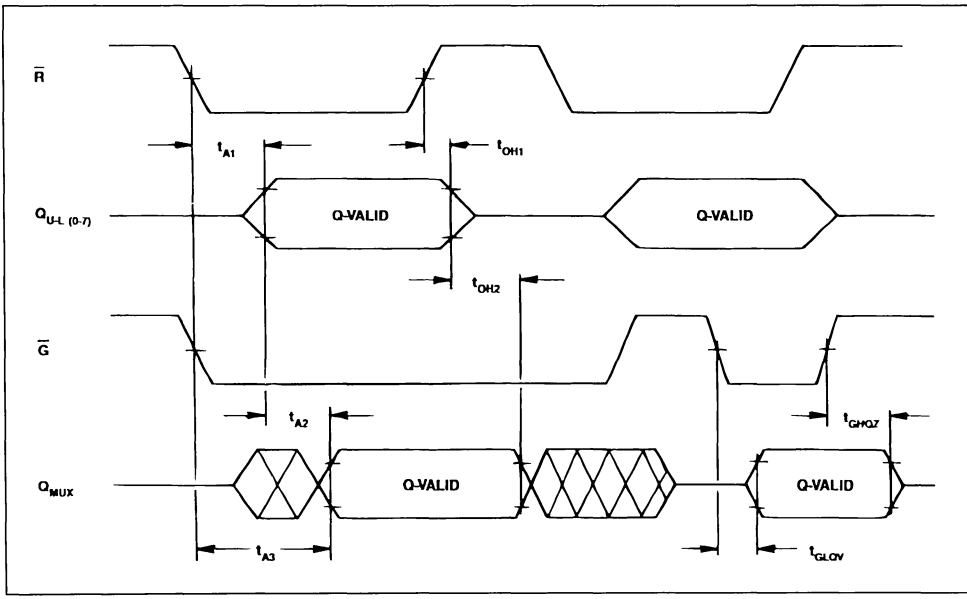


Figure 5 : Access Times.



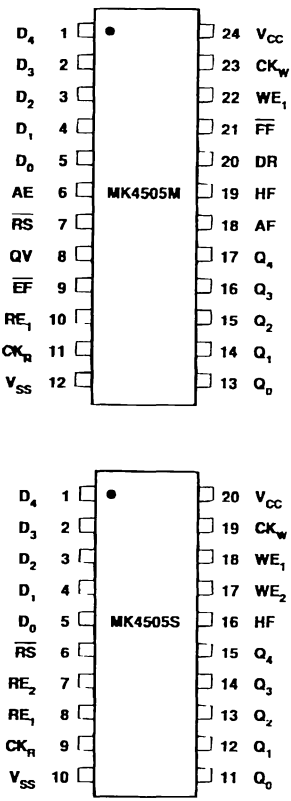
THE MK4505 CLOCKED FIFO INTRODUCTORY CONCEPTS

by Rick TITTLE

DEVICE DESCRIPTION

The MK4505 from SGS-THOMSON Microelectronics is the industry's first asynchronous clocked First-In-First-Out (5FIFO) memory. This very fast clocked FIFO supports two independent, asynchronous, free running clock inputs, and has an organization of 1k x 5-bits with FULL, EMPTY, HALF FULL, ALMOST FULL, ALMOST EMPTY, DATA READY, and OUTPUT VALID status flags. As shown in the pin-out diagram in figure 1, the MK4505 has two configurations : the MK4505M (Master) which features the full compliment of status flags, and the MK4505S (Slave). Either device, however, may be used in a stand-alone mode of operation. The Master features an automatic write protect when FULL, and an automatic read protect when EMPTY. In contrast, the Slave offers no protect circuitry, but can read and write continuously. Other MK4505 features are separate read and write enable inputs with the ability to enable or disable

Figure 1 : Pin Configuration, 300 mil DIP.



PIN NAMES

D_0 - D_4	DATA INPUT
Q_0 - Q_4	DATA OUTPUT
CK_W , CK_R	WRITE AND READ CLOCK
WE_1	WRITE ENABLE INPUT 1
$\overline{RE}_1$	READ ENABLE INPUT 1
RS	RESET (active low)
HF	HALF FULL FLAG
V_{CC}, GND	+ 5 VOLT, GROUND

(4505M only)

$\overline{FF}$, $\overline{EF}$	FULL AND EMPTY FLAG (active low)
AF, AE	ALMOST FULL, ALMOST EMPTY FLAG
DR, QV	IMPUT READY, OUTPUT VALID

(4505S only)

WE_2	WRITE ENABLE INPUT 2
RE_2	READ ENABLE INPUT 2 (rising edge triggered 3 state control)

DEVICE DESCRIPTION (continued)

read/write operations upon command in the presence of a continuous periodic clock train. The MK4505 is available with a cycle rate of 40MHz, offering a 15ns access time.

MK4505M/S CIRCUIT OPERATION

The MK4505 appears to the outside world as a clocked D-type flip-flop with separate enable inputs. For example, each time a D-type flip-flop receives a clock, the Q output responds with the D input after a certain propagation time, provided data and enable set-up times are met prior to the rising edge of the clock. The MK4505 responds in somewhat the same manner, but offers a First-In-First-Out, 1024 x 5-bit sized buffer.

Referring to the block diagram schematic in figure 2, the input stage looks like a rising edge triggered (CK_W) D-type flop with a separate enable comprised of a two input AND gate. Obviously, if either input is LOW, then the write operation is disabled. The output stage is similar, clocked by CK_R with a two input AND gate providing the read enable. If either input is LOW, the read operation is disabled. This provides the automatic read and write protection logic for the Master. The MK4505 FIFO uses the BIPORT™ RAM architecture, and appears as a dual port SRAM between the two I/O stages (see figure 3). Internal write and read address pointers or counters automatically provide the RAM with the correct addresses. Figure 4 displays the combined logic symbol derived from the diagrams shown in figures 2 and 3.

The MK4505 moves data only with each clock edge, as well as updating all status flags with the same clock edges. Each write function latches the input data with the rising edge of the write clock CK_W. The latched data is then transferred to a dual port RAM array. The predetermined first-in-first-out read sequence allows the read address counter to reach the stored data and present it to the input of the output latches before it is actually read. The output latches are then clocked on the rising edge of CK_R, and valid data is available t_A after the rising edge of the clock. Should a clock be turned off, as being locked LOW or HIGH, then the previous cycle is latched regardless of any input changes. This is true for the clocked D-type flip-flop, and the MK4505. Since the MK4505 supports asynchronous read/write clock functions, then either the read or write clock can discontinue without affecting the other port's activities.

Even though the MK4505 is designed for systems potentially operating from two asynchronous free running square wave clocks with cycle rates up to 40MHz, the device can be operated at much slower clock rates. A perfect square wave clock is not essential for proper device functionality. Parameters that must be satisfied are the specified minimum clock high and low times per the data sheet, and the set-up and hold times for enable inputs to the rising edge of the operational clocks. Clock duty cycle is not critical as shown in the example of figure 5. Remember that the FIFO triggers from the rising edge of the clock, and clock transitions meeting V_{IL} and V_{IH} specifications must be obtained.

The MK4505M (Master) cannot be written while FULL, nor read while EMPTY. A FULL condition results only from writing all 1024 bit locations, but an EMPTY condition results from either a RESET or by reading all bit locations previously written. Referring to the MK4505 data sheet, First Read Latency (t_{FRL}) describes the delay from the first write clock after empty, to the first valid rising edge of CK_R that is guaranteed to produce the first write data at the Q outputs. This is the time required from Data In to be latched, presented to the BiPORT RAM array, and flow through to the output latches where it will be clocked to the outside world.

CONCLUSION

The MK4505M/S from SGS-THOMSON is the industry's first high performance clocked FIFO. The MK4505 supports two independent, asynchronous, free running clock inputs. It has a 1k x 5-bit architecture, and offers a 15ns access time with a 40 MHz clock rate. The device features both read and write enable controls in the presence of continuous read and write periodic clock trains.

The MK4505 is available in two versions as a Master or Slave. This provides for easy width and depth expansion FIFO array configurations. The MK4505M Master provides a full compliment of status flags, as well as providing all necessary controls for width and depth expansion. A MK4505M is required for each 1k of depth, and a MK4505 Slave for each additional 5 bits of width. Either device may be used separately with certain system applications. The MK4505M cannot be written while FULL, or read when EMPTY ; whereas, the MK4505S can be forced to read and write continuously regardless of device status. Both devices use SGS-THOMSON BiPORT RAM based memory cell allowing simultaneous read/write operations, and offer full TTL compatibility.

Figure 2 : MK4505 Block Diagram Schematic.

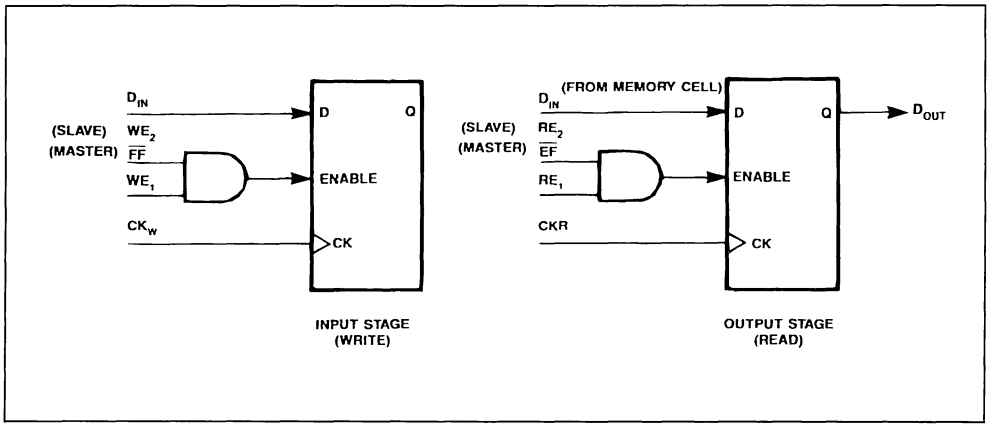


Figure 3 : MK4505 Representative Block Diagram.

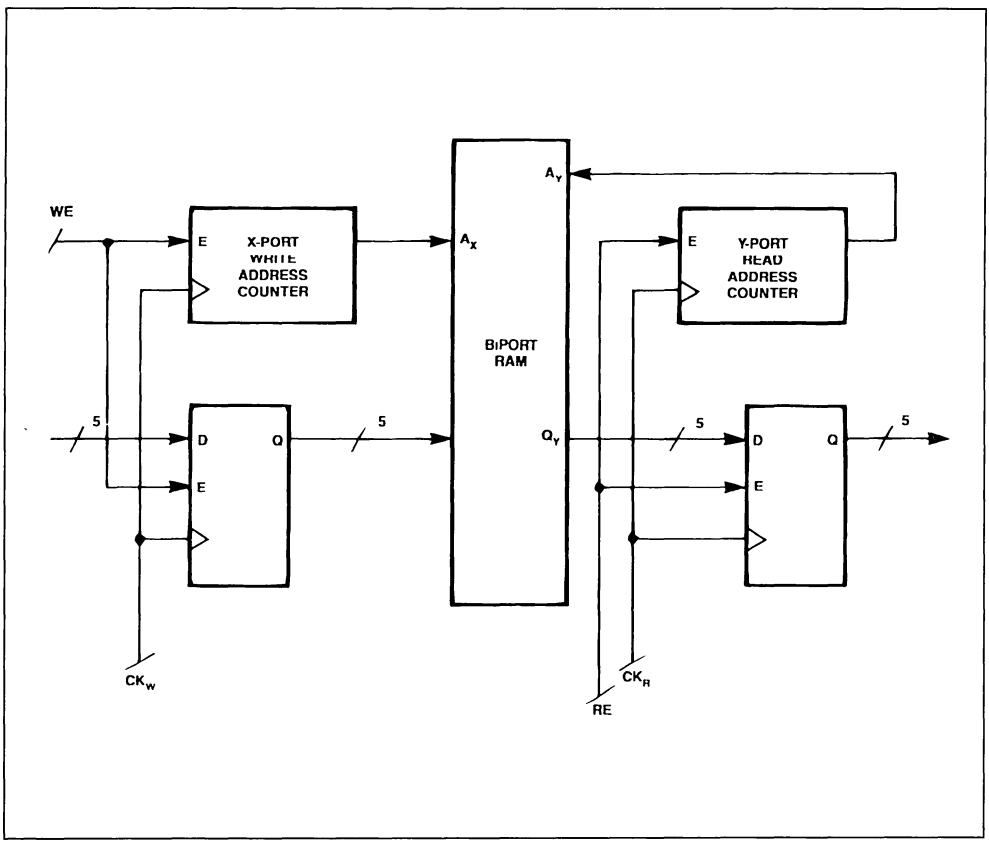
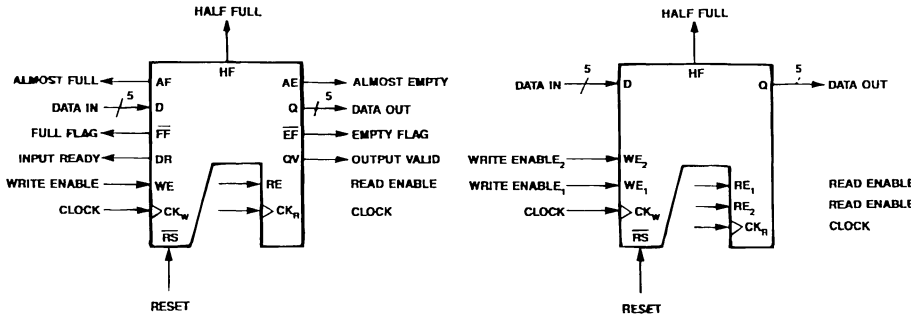


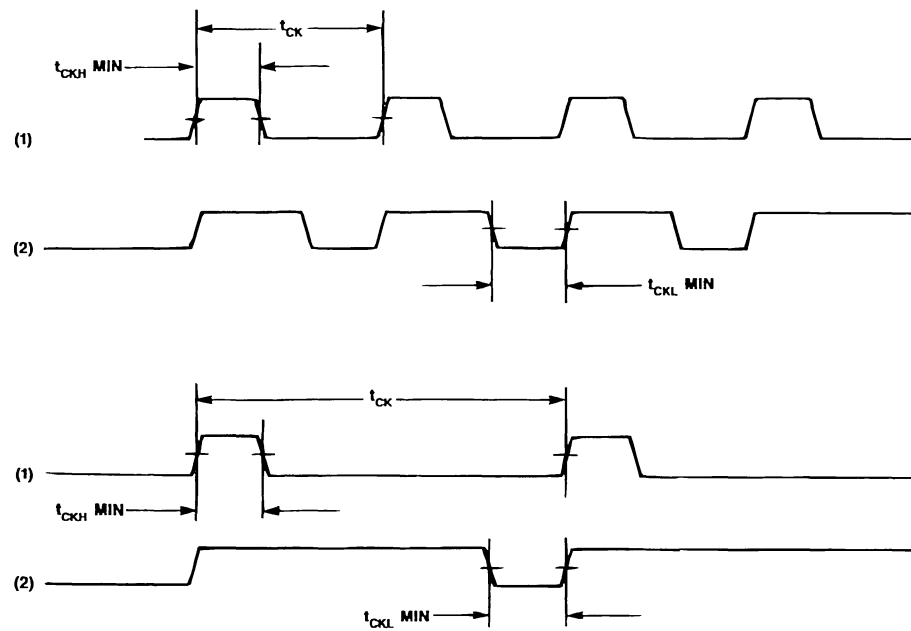
Figure 4 : MK4505 Logic Symbols.



MK4505M LOGIC SYMBOL DIAGRAM (master)

MK4505S LOGIC SYMBOL DIAGRAM (slave)

Figure 5 : Possible Valid Clock Examples.



USING THE MK4505 CLOCKED FIFO AS A DIGITAL DELAY

INTRODUCTION

The MK4505 is the first FIFO to support two independent, asynchronous, free running clock inputs. The MK4505 clocked interface makes high performance designs much simpler, eliminating the need for external registers, buffers, or pulse shaping circuits. The MK4505 is available in two configurations : the MK4505M (Master) and the MK4505S (Slave). This allows very simple width and depth expansion using one MK4505M for each 1024 bits of depth, and one MK4505M with one MK4505S for each additional 5-bits of width. Because it is Bi-PORT™ RAM based, it is not subject to the ripple-through delay times of conventional shift register type FIFOs. The MK4505 is suitable for applications with cycle rates up to 40MHz.

CIRCUIT DESCRIPTION

In the block diagram shown in figure 1, we have displayed a 1k by 10 configuration that would be typical of a high performance digital video delay application. Of course narrower or wider bit fields can be accomplished just as easily. The MK4505M provides all additional enables to the MK4505S, with WE1 going to both devices, into the counter circuit, and thus generating the Read Port Enable (RE1) once the desired count is reached.

CIRCUIT OPERATION

For the following application, we have shown an example for digital data delay in a system using one free running clock (ø clock) for CKW and CKR, and one Write/Read (WE1) control clock, with a programmable counter for determining the desired delay from 2 to 1022 cycles. In this particular example, we have shown the maximum delay allowed to avoid the FF (Full Flag) from being asserted low (see figure 2). The output of the counter circuit serves a two-fold purpose. First it provides the enable logic to the ENT pin for the first stage of the counter. Presetting the counter to 402HEX, assures that QC of the counter's third stage is set high, which sets the ENT enable of the first stage at the beginning of the count sequence. Secondly, the counter circuit provids an active high RE1 to the FIFOs enabling the delayed READ for output data once the count of 1022 (Full -2) has been reached.

After power-up and $\overline{\text{RESET}}$ ($\overline{\text{CLR}}$), preset the counter circuit by presenting 402 HEX (010000000010) through the ABCD inputs while strobing $\overline{\text{LOAD}}$ active low. The 74161 4-bit counter to the far left is the first stage. The system clock (Øclock) goes to each of the clock inputs of the counter circuit, WE1 is ENP enable, and RC0 goes to each successive device ENT enable ; with the exception of the first stage which is enabled via the QC output of the final stage (loaded high during preset). Once the counter circuit has reached the count of 800 HEX, which is the delay count of 1022 as (Full -2) for the MK4505, the QD output of the third stage is set which enables the delayed Read Port (RE1 = high) to the MK4505 1K x 10 configuration. At the same time that QD is set, the QC output of the third stage is cleared, inhibiting count of the counter circuit. Since the Read Port Enable (RE1) is latched active high, continuous Write/Reads are allowed on each successive rising edge of Øclock (see figures 3 and 4). The circuit will maintain a constant 1022 cycle delay until the next asynchronous $\overline{\text{RESET}}$ occurs.

PROGRAMMING DELAY INTERVAL

In conclusion, we have calculated a digital data delay of 1022 cycles. This is accomplished by realizing that setting QD of the third stage allows our maximum count to be 800 HEX (2048). Preset is determined by subtracting the desired delay nnn HEX from 800. Thus for this example it is (800 - 3FE HEX), which is 402. The decimal equivalent can also be used where ddd is the desired delay as (2048 - ddd) = DDDD. Preset is DDDD. Convert DDDD to nnn HEX. The final step is converting nnn HEX to its BINARY equivalent for the ABCD inputs during preset.

Figure 1 : Block Diagram - 4505 Data Delay 1K x 10-bits.

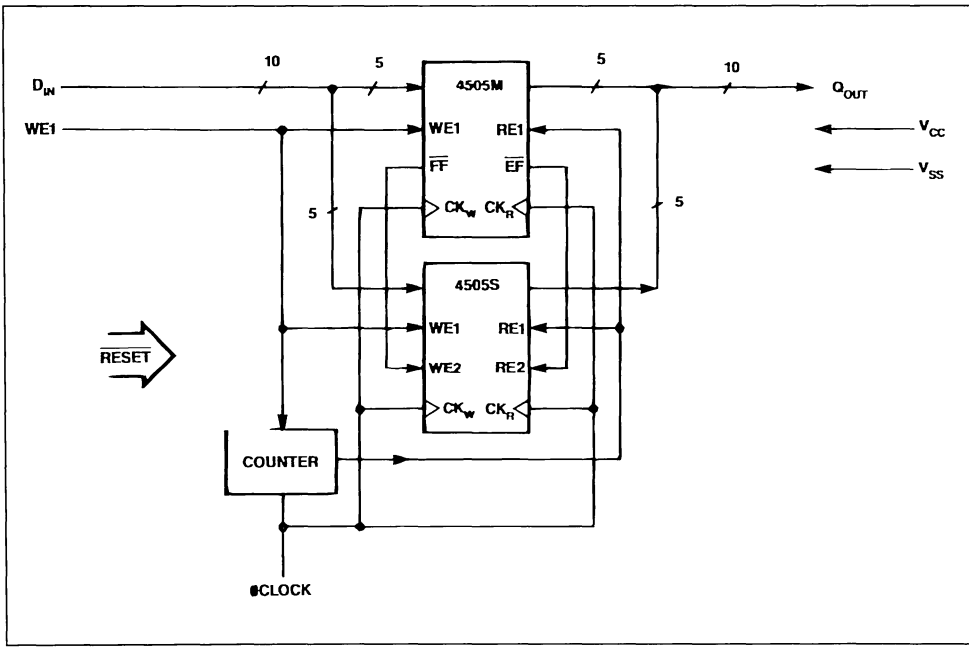
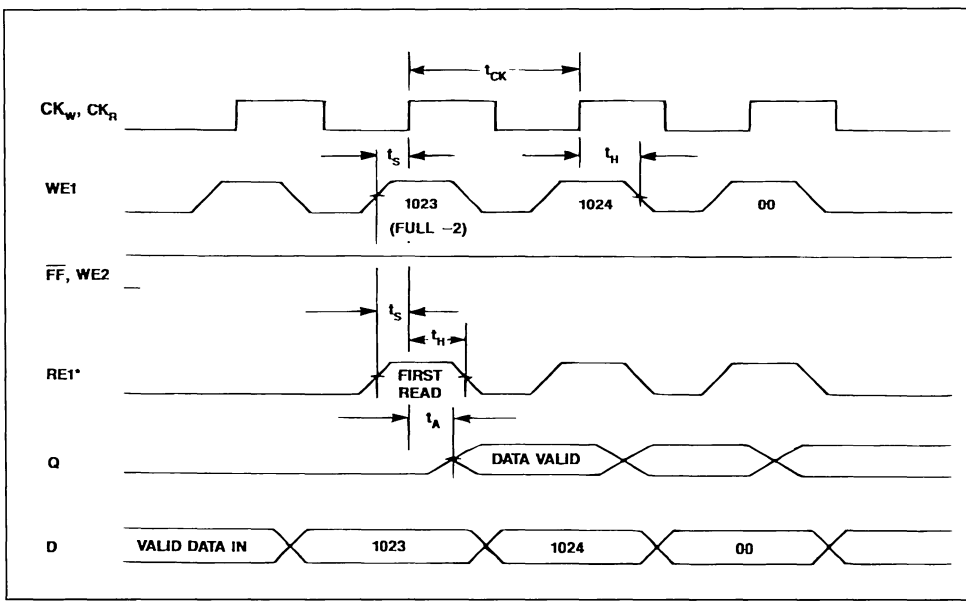


Figure 2 : Write/Read Timing - Latest Possible First Read to Prevent Full Flag Assertion.



* First read occurs at (full -2) to avoid full flag ($\overline{FF}$) assertion.

Figure 3 : Counter Delay Schematic.

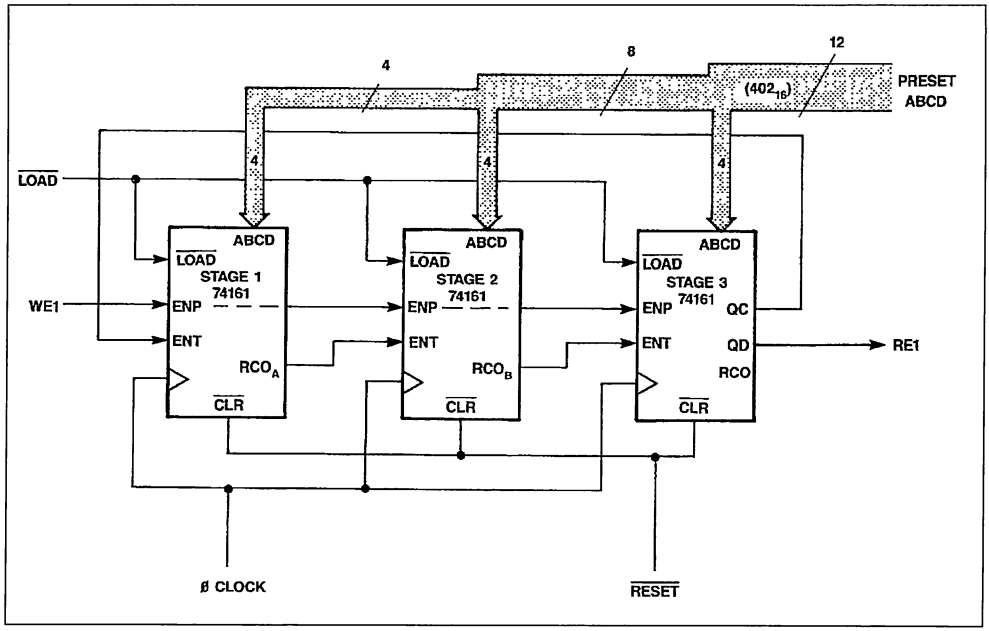
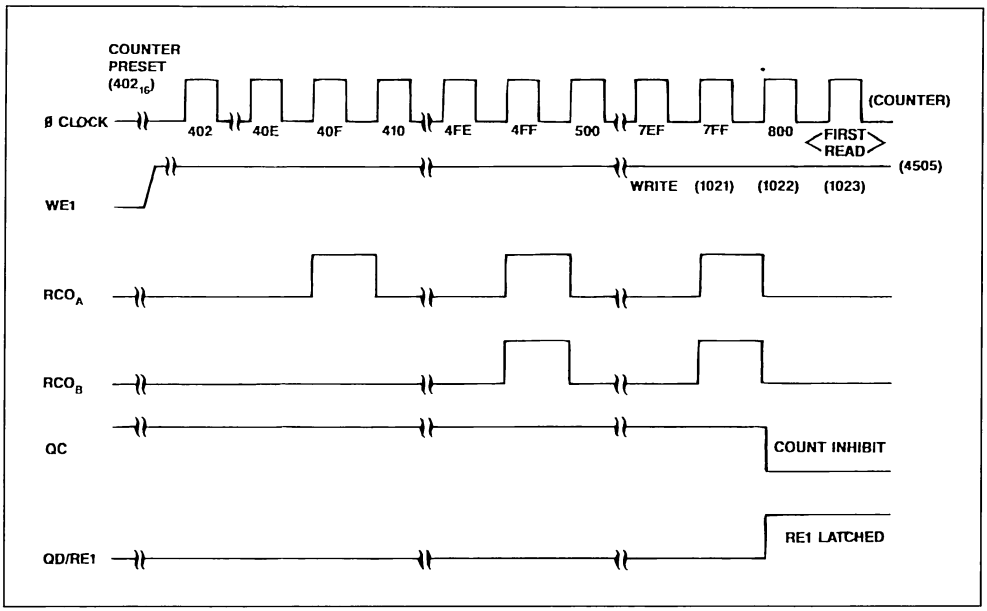


Figure 4 : Delay Timing Diagram.



* Timing diagram presumes preset of 402_{16} (1026) after reset, and full count of 1022 to latch re1 high when reaching 800_{16}(2048)

THE MK4505 LATCHED FLAG DESIGN CONSIDERATIONS

INTRODUCTION

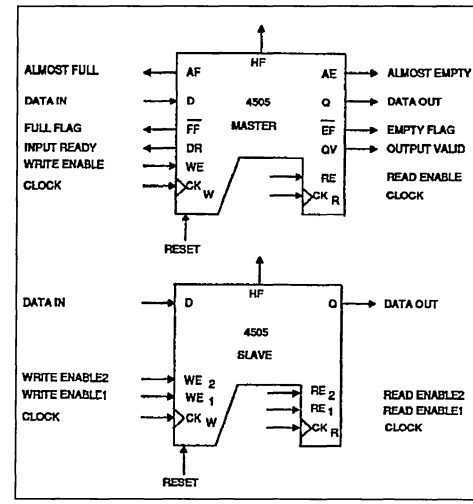
The MK4505M/S (Master/Slave) is a high speed clocked FIFO from SGS-THOMSON Microelectronics. The MK4505 offers high performance, high density First-In-First-Out operation, and supports two asynchronous periodic clock inputs for write/read operations. Using the master/slave concept, the MK4505's 1K x 5 architecture is easily width and/or depth expandable, with all control and flag signals provided by the MK4505 Master. This type of organization allows ease of use to the system designer, avoids word skew in width expansion, and provides write protection when Full, and read protection when Empty. In addition to Full and Empty detection, the MK4505 provides three additional status flags for early warning as : AE (Almost Empty), AF (Almost Full), and HF (Half Full). The MK4505 also provides separate write and read enable inputs which allows the write/read operations to be enabled or disabled upon command in the presence of a continuous periodic clock.

The purpose of this application brief to discuss the latch function of the MK4505's $\overline{EF}$ (Empty Flag) and $\overline{FF}$ (Full Flag), proper operation, and design considerations. The latched flag function can cause subtle unexpected flag status operations when compared to previous ripple-through FIFO devices. This applies to both asynchronous and simultaneous write/read functions. Understanding how the flags operate and why, will enable the system designer to correctly manipulate the flag logic for proper system operation.

DEVICE OPERATION

As a brief review, the MK4505 appears to the outside world as a clocked D-type flip-flop with enable inputs. (Refer to the MK4505 application brief Introductory Concepts). Data on the D input is clocked through the flip-flop, and presented to the Q output at some propagation time (access time), and remains stable (data is held valid) until the next rising clock edge. The MK4505 functions in much the same manner, allowing 1024 bits of information to be "clocked through" in a First-In-First-Out fashion. The MK4505 also provides a full compliment of status flags dependent upont the rising edges of the write and read clocks. Figure 1 displays the logic symbols for the MK4505 Master and Slave.

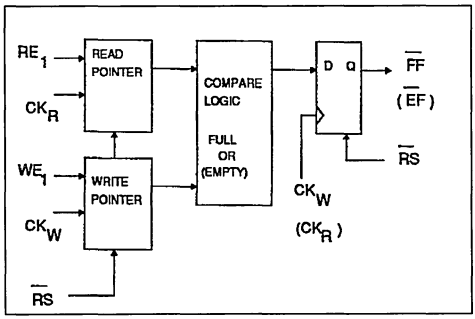
Figure 1 : MK4505 Logic Symbols.



LATCHED FLAGS

The MK4505 can only perform a valid write operation at the rising edge of CK_W, or a valid read operation at the rising edge of CK_R. Updates to all status flags are also dependent upon either CK_W or CK_R, but not both clocks together. The $\overline{EF}$ (Empty Flag) status can only be updated with the rising edge of CK_R, while the $\overline{FF}$ (Full Flag) status can only be updated with the rising edge of CK_W. Therefore, using the D-type flip-flop analogy, we conclude that these status flags are latched on a rising clock edge, and cannot change logic states until another clock edge occurs (refer to figure 2). This is an important concept for design considerations when using status flag logic states to signal a processor or PAL, for example, to initiate or interrupt predefined system operations. It makes the system designer's job easier because it ensures proper set-up and hold times referenced to the rising edge. It also promises that the flag's logic state will remain valid for at least one complete cycle.

Figure 2 : Latched Flag Logic Diagram.



EDGE TRIGGERED STATUS

Let's presume that in our application we want to write the FIFO until Full, then disable write operations and read until Empty. In addition, we find that it would take fewer logic gates to turn CK_R and CK_W on or off than gating read (RE_1) and write (WE_1) enable inputs. (figures 3A and 3B are simplified block examples for enable or clock control for a write-to-full, read-to-empty sequence). However, this can set the stage for unexpected flag status if both clocks are not active. The MK4505 can be operated in this manner, and deliver correct data in a first-in-first-out manner, but the designer must remember the latched flag philosophy. Specifically, one can write the MK4505 until Full where $\overline{FF}$ is latched low, and still maintain a valid empty condition with $\overline{EF}$ active low. This can happen by turning CK_W on and CK_R off, as CK_R updates the $\overline{EF}$ status. On the other hand, let's presume that we allow both clocks to run until full, and then disable the write clock (CK_W). We will find the same type of dilemma with $\overline{FF}$ remaining active low while clocking CK_R until the FIFO is empty. This would result again in both flags being set active low. Therefore, to ensure proper flag operation the system designer should use the write and read enable inputs when gating write/read operations.

Previous ripple-through FIFOs, such as the MK4501 and MK4503, allow $\overline{FF}$ and $\overline{EF}$ status updates as soon as an operation to clear or set the flag is accomplished. The MK4505, however, must determine the operation that clears or sets the flag, and then receive a rising edge clock (CK_W or CK_R) to update the flag's status. This is necessary for a clocked system environment, and again ensures set-up and hold times into the next cycle. Referring to the MK4505 data sheet, one will notice that DR (Data Ready) follows $\overline{FF}$ directly, and QV (Output Valid) follows $\overline{EF}$ by one cycle. All to say that these flags are latched with $\overline{EF}$ and $\overline{FF}$ as well. Their design function constitutes the handshake control for depth expansion.

TRUE FLAG OPERATION

As with other FIFOs, the MK4505 requires a reset pulse to initialize all internal counters before normal operation begins. Since the $\overline{EF}$ is latched active low after reset, two read clocks are needed to gate valid data out of the FIFO. The first rising edge of CK_R to occur t_{FFL} (First Flag Latency) after the first write will clear the Empty Flag ($\overline{EF}$ = high) within t_{F1A}. Read clocks less than t_{FFL} after the first write may clear the $\overline{EF}$, but are not guaranteed. The second rising edge of CK_R to ocur t_{FRL} (First Read Latency) after the first write will gate valid data onto the outputs provided RE_1 is active high. This is the expected "fall through" delay time, and is calculated as : $t_S + t_{FRL} + t_A$. This is true for asynchronous or si-

Figure 3A : Clock Control Block Diagram (not suggested).

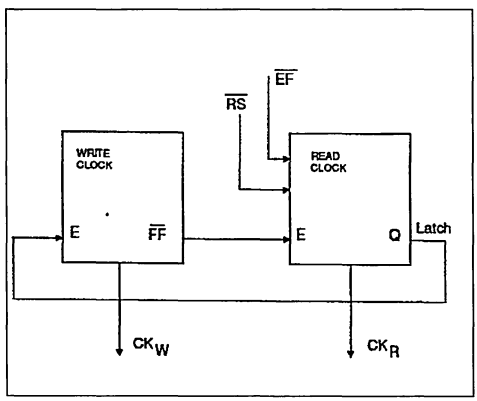


Figure 3B : Enable Control Block Diagram (suggested).

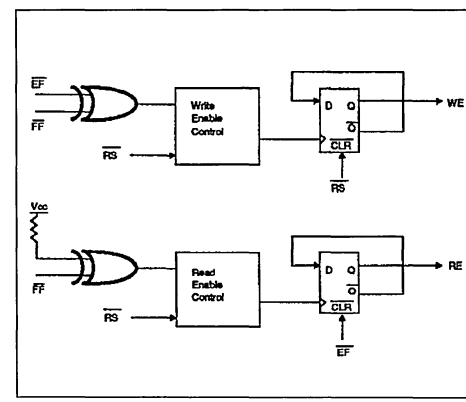
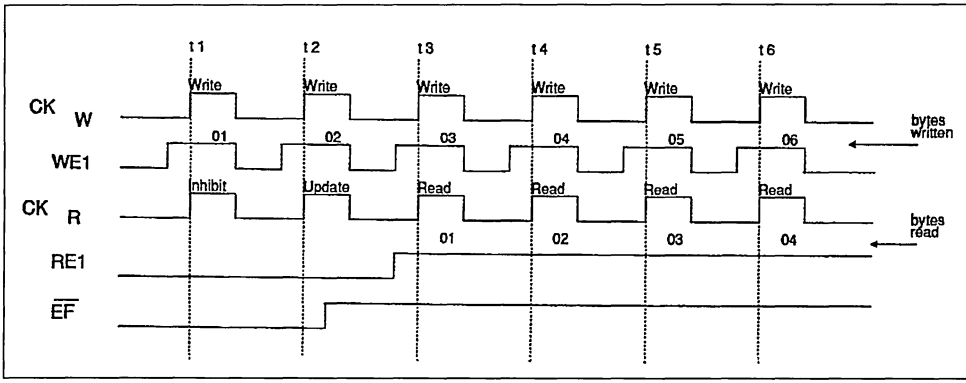


Figure 4 : True Empty Flag Operation.



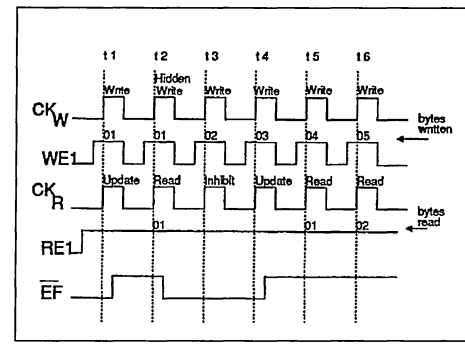
multaneous write/read operations. It should be noted that the RE_1 input should be held inactive until t_{FFL} has been satisfied, thereby observing the t_{FRL} parameter. This will ensure true $\overline{EF}$ operation ; see figure 4. (Although WE_1 is clocked in this example, it is not mandatory for proper device operation noted by RE_1).

If the MK4505 is written to full where $\overline{FF}$ is latched low, two write clocks are required before more data is written into the FIFO. The rising edge of CK_W on the last available byte of data latches the $\overline{FF}$ low within t_{F1A}. Once $\overline{FF}$ is latched low, the first rising edge of CK_W to occur after the first valid read will clear the $\overline{FF}$ status ($\overline{FF}$ = high). The second rising edge of CK_W will write data into the FIFO provided WE_1 is active high. These latched flag functions are a benefit of the MK4505's clocked design, and pipeline architecture. They provide write protection when full, and read protection when empty.

FALSE FLAG AMBIGUITY

There are certain situations and conditions that can result in improper device operation and probable unexpected status flag results. These conditions can occur with the $\overline{EF}$ or $\overline{FF}$ during asynchronous or simultaneous write/read operations. The worse case scenario being simultaneous write/read operations when near or at Empty or Full. Remembering that this is an edge triggered device that latches logic states by using a pipeline architecture, the internal arbiter will give priority to the read operation when a simultaneous write/read cycle occurs on the **last byte** of data. This results in the $\overline{EF}$ being latched low even though a valid write occurred at the same time. This is the design definition for the $\overline{EF}$ flag in a pipeline architecture being updated on the rising edge of CK_R. The actual last byte of data must be read without a simultaneous write, as this can "trap" one byte of data in the FIFO until a subsequent valid write is detected, the t_{FFL} parameter is again satisfied to CK_R, and $\overline{EF}$ is cleared ($\overline{EF}$ = high).

Figure 5 : False Empty Flag.

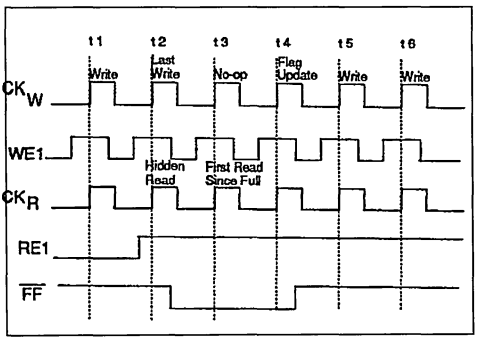


This can be best explained by the diagram in figure 5. If all operations were truly simultaneous, the false $\overline{EF}$ would not occur. In an actual system, however, the rising edges of CK_W and CK_R can become slightly skewed by a few nanoseconds in a simultaneous write/read operation. In the False Flag Diagram, the rising edge of CK_R is presumed to occur slightly **after** CK_W at t1, and slightly **before** CK_W at t3. The $\overline{EF}$ update circuitry is annulled at t2 because $\overline{EF}$ is high, and waits to determine a valid write while $\overline{EF}$ **is low**. This example results in improper device operation as t_{FFL} and t_{FRL} are not satisfied, and an unexpected $\overline{EF}$ status is displayed. Therefore, if RE_1 is kept disabled t_S prior to t3 as in the True Flag diagram, or if CK_R occurs just before or exactly with CK_W at t1, then t_{FFL} is satisfied at t2, t_{FRL} is observed at t3, and the unexpected empty condition will not occur. The $\overline{EF}$ flag will function as anticipated without "trapping" data by allowing the

writes to stay one ahead of the reads. This allows successful data transmission with asynchronous or simultaneous write/read cycles.

Since $\overline{EF}$ and $\overline{FF}$ are both latched flags, the same type of false flag can be encountered with the $\overline{FF}$. Referring to figure 6, one can see that a simultaneous write/read on the **last available write** results in $\overline{FF}$ being latched low even though a valid read occurred at the same time. Due to the pipeline design, the arbiter gives priority to the write, thus latching the $\overline{FF}$ low. The $\overline{FF}$ will not be cleared until a subsequent rising edge of CK_W **occurs after the first valid read.** The $\overline{FF}$ update circuitry is annulled at t2 because $\overline{FF}$ is high, and waits to determine a read while $\overline{FF}$ **is low**. This is the design definition for the latched Full Flag in a pipeline architecture being updated by CK_W.

Figure 6 : False Full Flag.



SUMMARY

The MK4505 clocked FIFO provides latched Empty and Full flags guaranteeing a stable logic state for at least one complete cycle. This allows ease of use in a clocked system environment by ensuring that set-up and hold times are met for a subsequent cycle in reference to the system clock. Since $\overline{FF}$ and $\overline{EF}$ provide the MK4505 with automatic write protection when Full, and read protection when Empty, the designer needs to understand their design definition for proper device operation. When compared to previous ripple-through FIFOs, the latched $\overline{EF}$ and $\overline{FF}$ can result in an unexpected or false flag status. In reference to the latched Empty Flag, should the $\overline{EF}$ logic present a problem of trapping data in a given clocked system as previously described, the AE signal can be used as an option. By using the Almost Empty condition (Almost Empty by 8 bytes) for Empty, one can detect AE for system logic commands, perform a valid simultaneous write/read operation, and still read the **last byte** of data written. This will avoid a false flag situation, and allow the system to retrieve all expected data.

CONCLUSION

In this application brief, we have highlighted some subtle benefits and operations of the MK4505 clocked FIFO in regards to latched status flags. Obviously, the MK4505 demands certain design considerations when compared to previous ripple-through type FIFOs like the MK4501. By providing latched $\overline{EF}$ and $\overline{FF}$ flags, the MK4505 ensures proper operation for set-up and hold times in a clocked system environment. Previous ripple-through FIFOs cannot within themselves ensure proper set-up and hold times in a clocked system design. Therefore, the MK4505 makes design easier, and takes the worry out of interfacing a FIFO with a system operating on a periodic or free-running clock.

THE MK4505 MASTER/SLAVE WIDTH EXPANSION

INTRODUCTION

The MK4505 from SGS-THOMSON Microelectronics offers high performance, high density First-In-First-Out operation, and supports two asynchronous periodic clock inputs for read/write operations. The device is designed for applications where data is moving through a system on the rising edge of a free running clock, and allows simultaneous or asynchronous read/write operations. The basic concepts of the MK4505 clocked FIFO are previously described in an application brief entitled : "The MK4505 Clocked FIFO : Introductory Concepts", publication number 4430314. The ideas described in this brief will help first time users in understanding the different functional concepts with the MK4505 from previous low and high density FIFOs.

The main difference between the MK4505 and other FIFOs is that all operations are initiated on the rising edge of the read or write clocks, CK_R and CK_W respectively (see figure 1 for device pinout). This particular application brief will discuss read and write cycles for the MK4505M (Master), and the MK4505S (Slave) as separate devices, as well as their cascadable application. Width expansion will be reviewed as Master-to-Slave and Slave-to-Slave. Master-to-Master width expansion is not allowed. Each device has a 1K x 5 bit organization, and is width expandable to at least 40 bits with no additional logic. Cycle times range from 20 to 40MHz with access times of 15 to 25ns.

MK4505M/S (MASTER/SLAVE) CONCEPT

The MK4505M/S (Master/Slave) concept provides for easy width and depth expansion capability by using one MK4505M for each 1024 bits of depth, and one MK4505S for each additional 5 bits of width. For most applications, the Master-to-Slave width expansion configuration offers the best solution. A full set of status flags is offered by the MK4505M, and thereby providing all of the necessary controls signals to the Slave. The Master-Slave width expansion arrangement gives the system designer worry free operation from word skew ambiguities that can be realized due to status flag access times during asynchronous read/write operations. The MK4505M is supplied in a 300mil, 24 pin plastic DIP, and the MK4505S comes in a 300mil, 20 pin plastic DIP which gives additional savings in printed circuit board space. An example of some simple read and write timing diagrams are shown in figures 2 and 3 showing the differences of the MK4505M and MK4505S as stand-alone devices. These figures presume that the Master is neither full or empty.

SLAVE WIDTH EXPANSION

The key difference between the MK4505M and MK4505S is that the Master has on-chip write protection when full, and read protection when empty, whereas the Slave offers no overflow or underflow protect circuitry. The MK4505S can be used as a stand-alone device or in width expansion without the Master, but this forfeits the automatic read/write protect circuitry offered in the MK4505M. Even if no read cycles occur, previously written data will be over-written within 1025 write cycles. Of course data can also be re-read if no write cycles occur within 1025 read operations. A reset operation will also allow the user to re-read data with the MK4505S. However, during reset both the read and write pointers are reset to location (address) zero. Additionally, during reset the half full (HF) counters will be initialized.

An example schematic of Slave-to-Slave width expansion is shown in figure 4. The previous read/write diagrams in figures 2 and 3 are also examples of the MK4505S (Slave) width expansion timing. Since the MK4505S allows continual read and write cycles, the MK4505S can be used in applications where data underflow and overflow to the FIFO cannot occur, or where data underflow and/or overflow is desired or doesn't matter. Here the Slave will offer high performance with additional board space reductions. The MK4505S also offers a rising edge triggered three state output bus control. Whenever RE2 is LOW at the rising edge of CK_R, the Qs will be high-Z at t_{QZ} from the rising edge of CK_R.

MASTER/SLAVE WIDTH EXPANSION

As previously mentioned, the Master/Slave width expansion offers a full compliment of status flags and all necessary control signals for proper operation. The MK4505M provides a high impedance data bus after reset, and whenever the FIFO is empty. An example schematic of MK4505M/S width expansion is shown

in figure 5. This example is presuming a data bus width of 32 bits with an additional 3 bits. The additional bits could be used to pass decode information or other control status information. Master/Slave width expansion timing is displayed in figure 6. This figure shows an initial reset with flag status, and First Read Latency (t_{FRL}) after First Write, and finally write until full with read disabled (RE1 = LOW). Additionally, the read and write clocks are asynchronous at different cycle rates.

The logic state of the DR status flag follows the $\overline{FF}$ status flag, and is updated with each rising edge of the write clock (CK_W). The QV status flag follows $\overline{EF}$ by one cycle respective to the rising edge of CK_R. The DR and QV outputs are latched status flags, with their main function being to incorporate Master-to-Master depth expansion, or Master-to-Slave with and depth expansion (refer to the MK4505M/S data sheet). Therefore, DR (Data Ready) and QV (Output Valid) are offered as optional status flags to be monitored by the user in Master-Slave with expansion mode. They are totem-pole outputs, and can be left disconnected in this configuration.

DESIGN CONSIDERATIONS

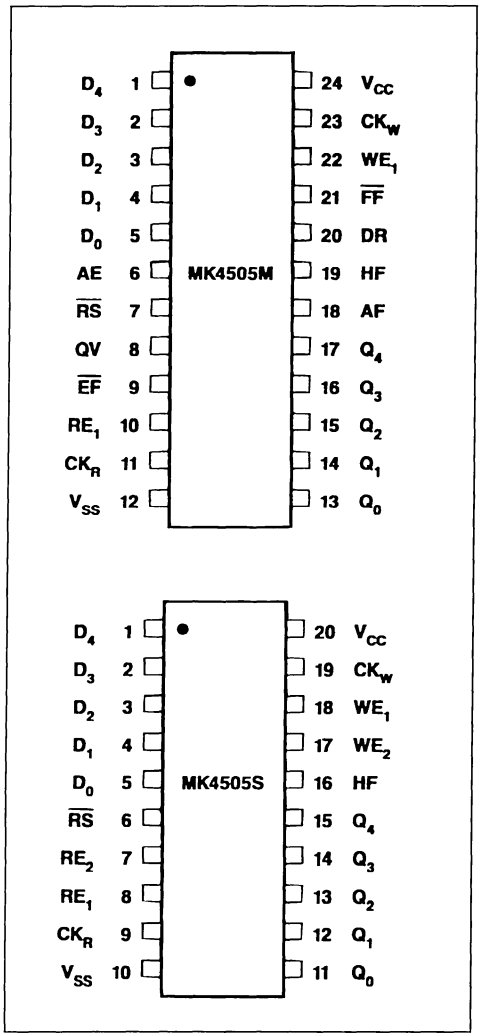
As with all FIFO applications, a reset is required to initialize all counters before normal operation begins. Referring again to the MK4505M/S data sheet, the user must take into account the first valid read delay access time from the first valid write operation after reset. This is the expected "fall-through delay time" calculated as : $t_S + t_{FRL} + t_A$. The t_S parameter being the set-up time to the First Write operation ; t_{FRL} is the First Read Latency as the First Read clock delay after the First Write ; and t_A is the Q output access time from the First Valid Read Clock. The MK4505M provides read protect circuitry via the $\overline{EF}$ status flag, therefore read operations will not be allowed until the $\overline{EF}$ is cleared. Thus first valid data after reset has a maximum of 75ns for a 40MHz cycle.

When using the MK4505S separately, the user must observe the t_{FRL} parameter to ensure First-Write-to-First Read valid data. Once t_{FRL} is satisfied, valid data is guaranteed on the Q outputs at t_A from the rising edge of CK_R. Read operations attempted before t_{FRL} is satisfied may result in reading RAM locations not yet written. Since the Slave offers no read or write protect circuitry, the user must observe t_{FRL}, especially when using free running asynchronous read/write clocks on the MK4505S.

CONCLUSION

The MK4505M/S offers width expansion to any line or word size, and at least 40 bits of width without additional support circuitry. Master-to-Slave width expansion has a full compliment of status flags as well as Slave control logic for proper operation. The MK4505S is also width expandable with certain restrictions. The MK4505M cannot be written while FULL, or read when EMPTY ; whereas, the MK4505S allows continuous read/write operations. Master-to-Master width expansion is not allowed. Both devices use the SGS-THOMSON Microelectronics BiPORT™ RAM based memory cell allowing simultaneous read/write operations, and a 1k x 5 pipeline architecture giving the MK4505 an unsurpassed 15ns access time with a 25ns cycle time.

Figure 1 : Pin Connection, 300 MIL DIP.



PIN NAMES

D_0 - D_4	- DATA INPUT
Q_0 - Q_4	- DATA OUTPUT
CK_W, CK_R	- WRITE AND READ CLOCK
WE_1	- WRITE ENABLE INPUT 1
RE_1	- READ ENABLE INPUT 1
$\overline{RS}$	- RESET (active low)
HF	- HALF FULL FLAG
V_{CC}, V_{SS}	- + 5VOLT, GROUND

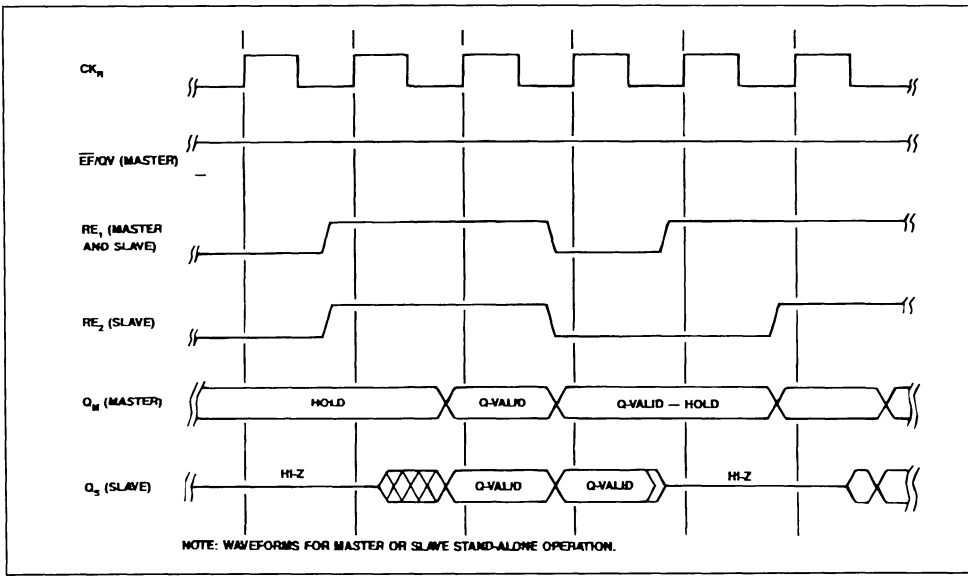
(4505M only)

$\overline{FF}$, $\overline{EF}$	- FULL AND EMPTY FLAG (active low)
AF,AE	- ALMOST FULL, ALMOST EMPTY FLAG
DR, QV	- INPUT READY, OUTPUT VALID

(4505S only)

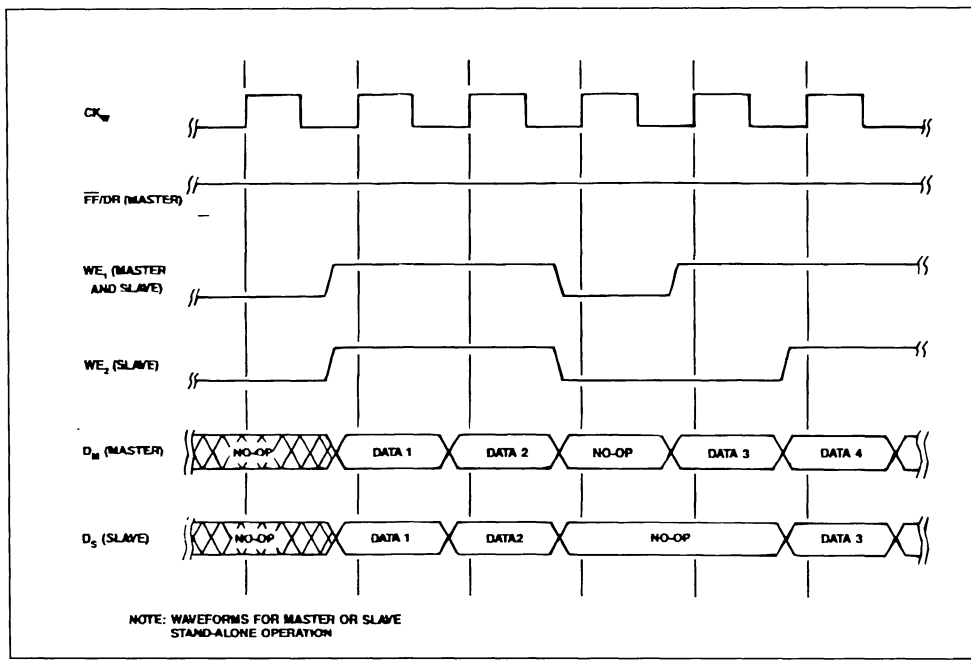
WE_2	- WRITE ENABLE INPUT 2
RE_2	- READ ENABLE INPUT 2 (rising edge triggered 3 state control)

Figure 2 : MK4505M/S Read Timing.



Note : Waveforms for master or slave stand-alone operation.

Figure 3 : MK4505M/S Write Timing.



Note : Waveforms for master or slave stand-alone operation

Figure 4 : Slave width Expansion.

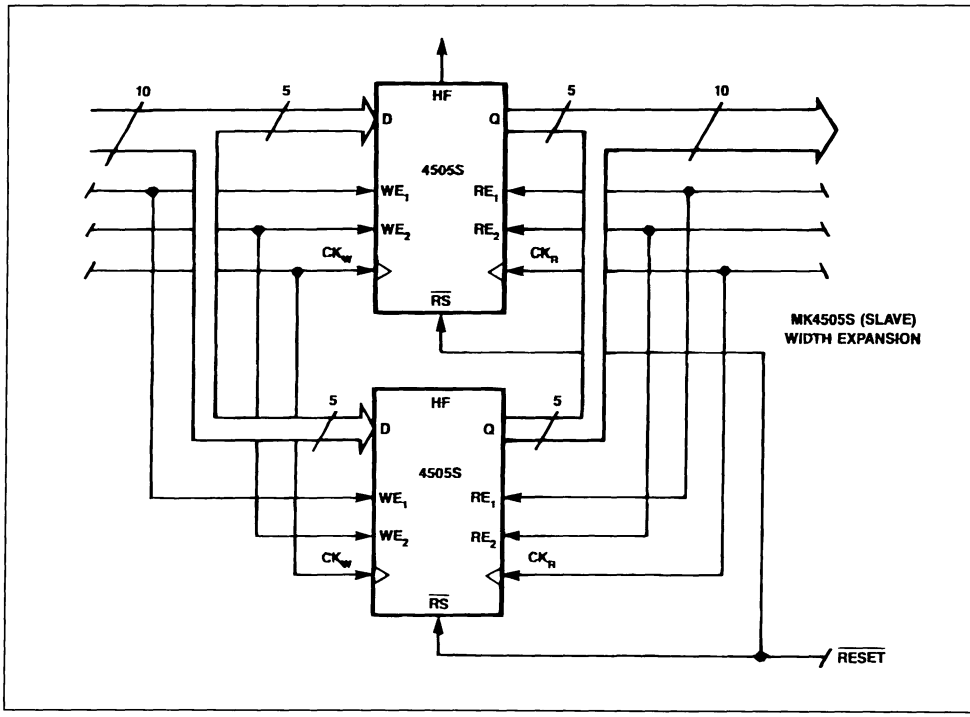


Figure 5 : Master/Slave width Expansion.

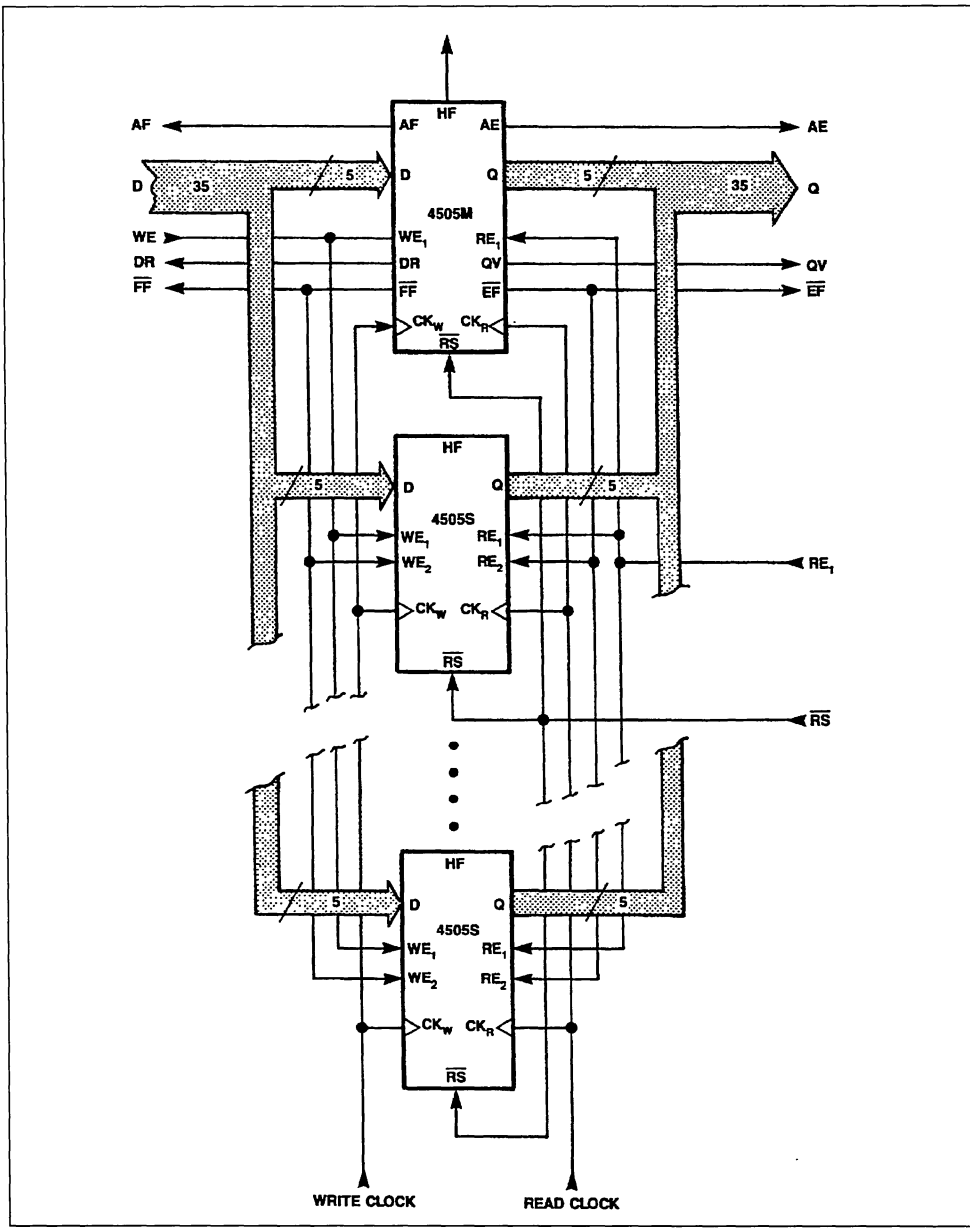
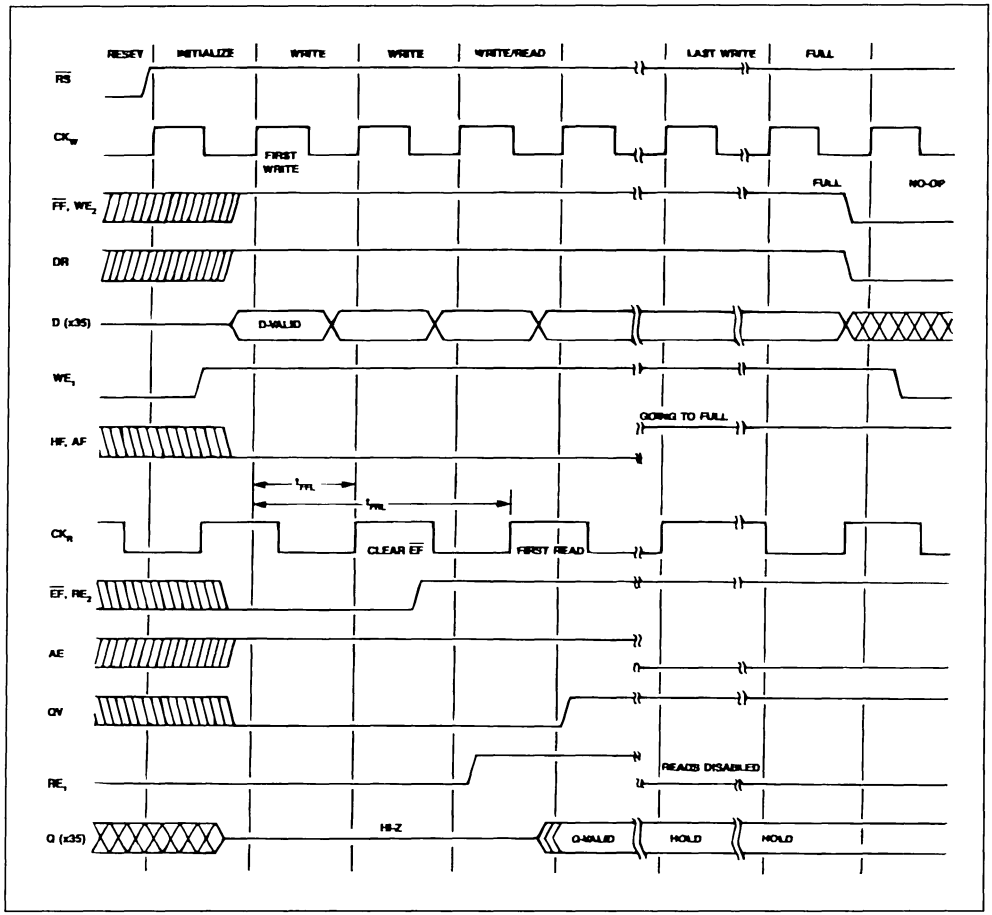


Figure 6 : Master-to-slave widthExpansion Timing Write/Read/Write-to-full.



MK4202 TAGRAM™ 32-BIT CACHE DESIGN CONCEPTS

INTRODUCTION

The MK4202 cache TAGRAM from SGS-THOMSON Microelectronics is a very fast CMOS SRAM based Cache Directory Comparator. The MK4202 offers high performance with a 25ns cycle time, and a 20ns address to tag compare access time. It is configured for the new generation 32-bit microprocessors, and implements a 2K x 20 architecture (see figure 1). The device contains a 20-bit on-board comparator with dual compare or match outputs for easy interface to various 32-bit processors. An on-board chip enable decoder is also included to allow both width and/or depth expansion. Depth expansion is allowed up to a 32K density without speed (propagation time) penalties.

The MK4202 can enhance both system performance and cost. The high speed compare access time enhances the zero wait state logic to the processor for better system performance. Secondly, the MK4202 has high impedance data bus control inputs ($\overline{S}$) ($\overline{G}$), and eliminates separate latch, transceiver, and comparator components. This reduces the required real estate in PC board area, and results in lower system cost with no added gate delays through separate components.

TAGRAM OPERATION

TAGRAM. What is a TAGRAM ? A TAGRAM is that part of a *cache subsystem* that determines if data or instructions is retained in the cache memory (data cache). While this may be obvious to some, it should be noted that typical cache subsystems are designed in three sections : the **data cache** which stores the data to be used by the microprocessor, the **cache tag buffer** or **cache directory**, which stores the upper order address of each cache entry, and the **cache control** logic for processor interface, and cache read/write operations (see figure 2). The data is identified by address location, and is stored in the cache directory (the MK4202 serves as the cache directory). If the TAGRAM "sees" a match, meaning that the necessary data is resident in the cache, **it** determines a **hit** and initiates a zero wait state operation to the processor. Conversely, **a miss** (nomatch) condition results in longer cycles for program execution since the cache does not contain the requested information.

During a compare cycle, the MK4202's on-board 20-bit comparator compares the upper address inputs (DQ_0 - DQ_{19}) with internal RAM data at the specified index address (A_0 - A_{10}). (The TAGRAM's architecture is shown in figure 3). If all bits are equal (match), a hit is determined, and both Compare Outputs (C_0 and C_1) will go HIGH. If at least one bit is different, a miss condition exists, and both C_0 and C_1 will go LOW. The user can optionally force a MISS or HIT on either or both compare outputs by asserting $\overline{M}_X$ or H_X active LOW. A forced MISS overrides a forced HIT input (note the data sheet Truth Table). The Compare Output Enable ($\overline{CG}_X$) has no affect during a Force Hit or Force Miss operation.

CIRCUIT DESCRIPTION

A Direct Mapped Cache subsystem application using the MK4202 is shown in the schematic block diagram in figure 4. In this example, we are serving a 32-bit microprocessor with a 32-bit address bus and data bus. The 12 lower significant address lines (A_0 - A_{11}) are connected to the address inputs of the MK4202, with A_{11} connected to E_0 of both TAGRAMs as shown. This input will implement a 4K address index for the Cache Directory in this example, and a 4K address for the Cache Data RAM. The upper address bits of the processor (A_{16} - A_{31}), function as the tag data bits for the MK4202 TAGRAM, with CDQ_0 pulled up to V_{CC} (+ 5 volts) as a **valid bit.** Therefore, the cache tag buffer, or Cache Directory, consists of 4K x 16 bits, or 8K bytes. The Data Cache consists of eight very fast MK41H68 4K x 4 SRAMs with Chip Enable ($\overline{E}$) for a 32-bit line width, resulting in a total of 16K bytes of memory (refer to figure 5). Cache subsystem studies have shown that a cache design using this mapping scheme and combined memory size can yield a hit rate of better than 80%.

The MK4202 TAGRAM uniquely identifies each cache entry with an *address scheme consisting of the index and tag.* The upper addresses of a 32-bit microprocessor are considered as the tag, and are connected to the DQ pins (DQ_1 - DQ_{16}) of the MK4202 (see figure 4). The CDQ_0 pin is designated to serve as the valid bit, as it allows the internal RAM to be reset or cleared to all zeros at all locations for the CDQ_0 output. The 11 address inputs (A_0 - A_{10}) of the MK4202 function as the **address index**, and

are connected to the lower addresses of the processor. Extra lower order address lines can be connected to the Enable Inputs (E_0 - E_3) for on-board chip enable decode for depth expansion. (The E_0 - E_3 inputs correspond directly to the P_0 - P_3 inputs as described in the MK4202 data sheet). In this example, A_{11} is connected to E_0 for a depth expansion of 4K. The C_0 and C_1 Compare Outputs incorporate a CMOS totem-pole 3-state design. This high impedance state allows each output to be tied together in a wired-OR configuration for multiple device applications, as shown in figures 4 and 6. A pull-up resistor is suggested to enhance proper operation.

All processor address lines do not have to be connected to the TAGRAM, as most systems require designated *non-cached* addresses. Some address lines may define peripherals from an address vector for various I/O devices, or define DMA operations. Thus, a certain amount of address space is often required for various non-cacheable operations. Therefore, extra address lines from the processor can be used to decode these system operations. Unused addresses from the processor can be left open ; however, unused control and address inputs or tag data inputs to the MK4202 should be tied to be determined as a logic one or logic zero.

CIRCUIT OPERATION

The basic purpose of our Direct Mapped Cache subsystem is to provide the microprocessor (henceforth CPU) with a quick response for requested data from the **data cache** to avoid long memory cycles to main memory. Since typical programs spend most of their execution time in loops or nested procedures from a few localized areas of the program, the cache subsystem can maintain a copy of this localized data for faster execution. This will make our system more efficient, and provide optimum performance.

Operation begins by initializing the MK4202 TAGRAM by asserting RS to clear all internal SRAM bits for the CDQ_0 output to a logic zero. This invalidates all entries in the cache directory. When the CPU begins its first read command cycle, the lower address bits select a location in the MK4202. This location in the TAGRAM is compared against the 16 bits of tag along with the single valid bit ($CDQ_0 = V_{CC}$). The valid bit will determine a miss condition causing C_X to go LOW since data ones is being compared against data zeros. This is a **cache miss**, and the CPU will wait for a response from main memory.

In our Direct Mapped Cache subsystem example, we will assume the *write-through* method for main memory and cache coherency. This means that while the CPU is waiting for data during a **read/miss** operation, that the cache write/read control logic is designed to write data from main memory into the MK41H68 data cache SRAMs at the address defined by the index. When this happens, the 16 bits of tag, along with the valid bit, will be written simultaneously into the MK4202 cache directory. This way both the main memory and the **cache subsystem** contain the same information. (* The MK41H79 is a FSRAM option with Output Enable ($\overline{G}$), and implements a RAM flash clear function).

Now the logic one valid bit on the MK4202 TAGRAM can verify that the data in the data cache is a valid copy of main memory, since both the internal data and CDQ_0 (valid) are data ones. Of course a cache miss will not only occur after a reset operation, but every time the lower 12 address index bits select a location where the 17 tag bits (16 bits plus valid bit) do not match. However, each time a miss occurs, the most recent data will be written into the cache memory. This maintains cache coherency with main memory by constantly updating the cache with the most recent data.

After the cache has been written with valid data, and the CPU executes a read command from the same location that has been written into the cache subsystem, then the data tag bits will match, and a **cache hit** results. During a hit condition, both compare outputs, C_0 and C_1, will go HIGH. This will cause the CPU to operate in a no-wait state, and data will be gated onto the data bus from the data cache memory. The match/hit operation takes the requested data from the data cache, avoids wait state cycles to main memory, and terminates the cycle with a data acknowledge to the CPU.

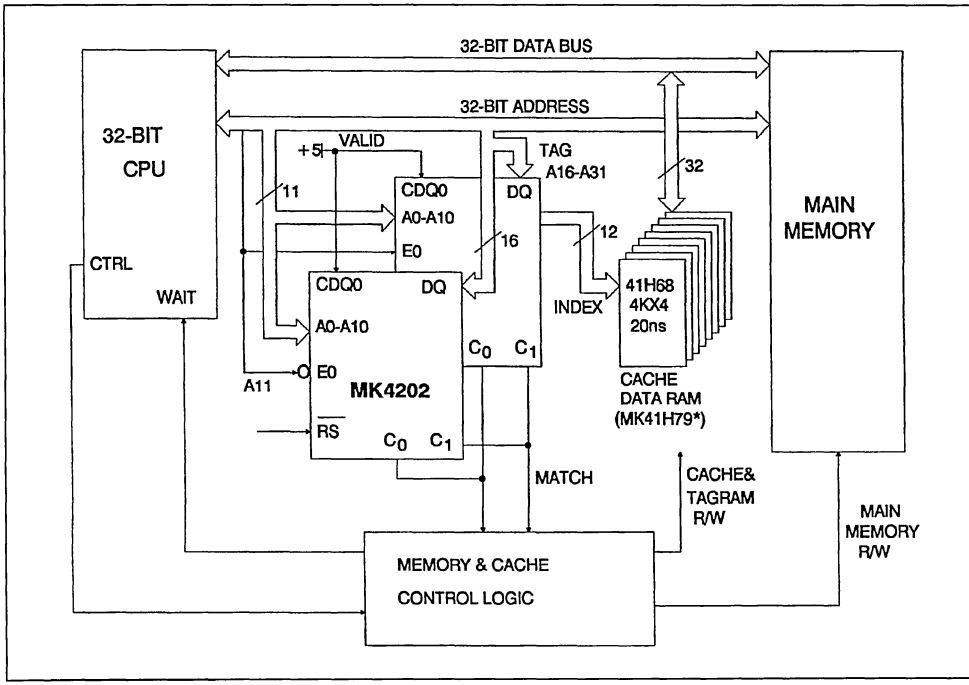
It should be noted that even though both C_0 and C_1 go HIGH during a hit, that both compare/match outputs may not be required for the cache hit/miss control logic. The user could use one match output for determining a miss or hit condition, and use the other compare output to pass inputs through the MK4202 to the CPU. This is illustrated in figure 6, where C_0 is normally HIGH except in an error condition. In this manner, a system error or halt will alert the CPU and invalidate cache execution at the same time. This is another benefit using the MK4202, and can reduce the number of required logic gates, and therefore system components.

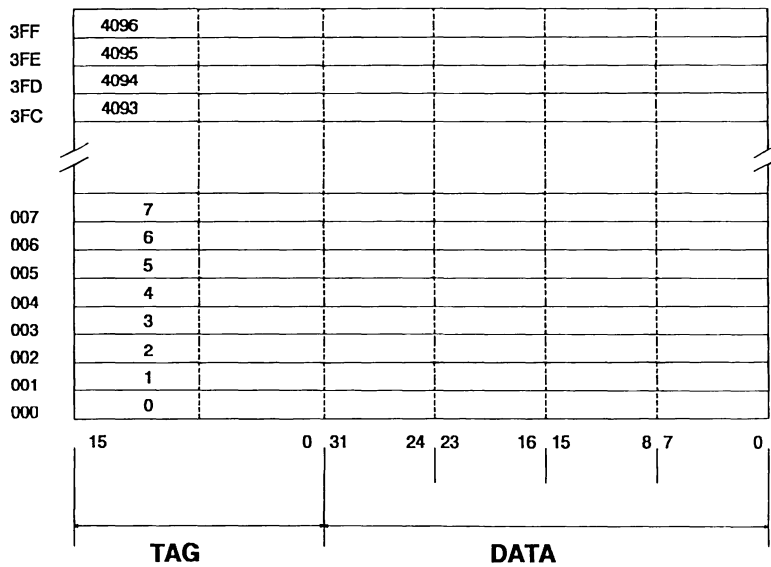
CONCLUSION

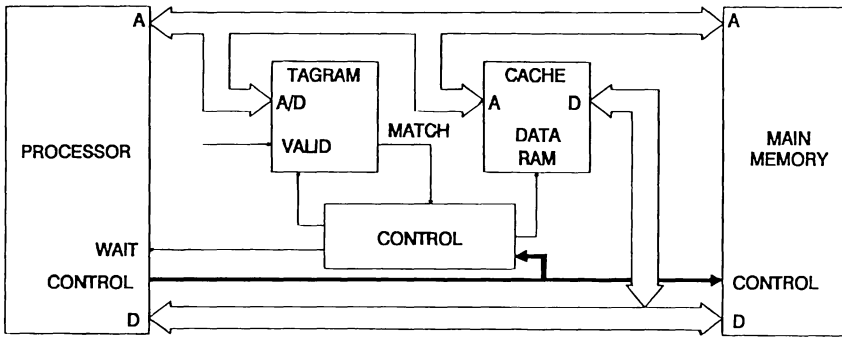
We have given a general application and overview of the MK4202 cache TAGRAM in conjunction with a 32-bit data bus Direct Mapped Cache subsystem concept. System operation allows the cache subsystem to be updated with the most recent data for each access to main memory. The MK4202 allows easier implementation, and reduces the number of devices or components. Of course the MK4202 can also be used to implement a two or more set-associative cache design. In fact, by using the enable and chip select inputs on the MK4202, one can easily implement several cache design arrangements.

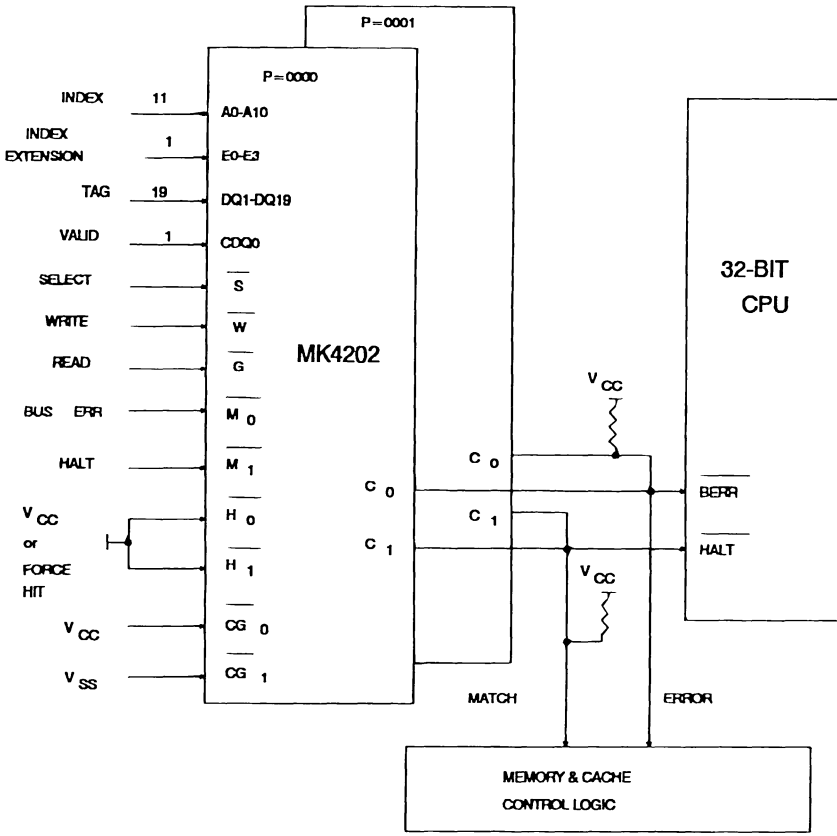
There are several features and organizational benefits when using the MK4202 for cache design. The device provides a flash clear ($\overline{RS}$) function to invalidate cache entries, which is useful for single processor systems, and for multi-processor systems sharing a cache subsystem. Additionally, the MK4202 is wide enough to provide extra bits for storing other information besides address tag data. One example might be "dirty bit" storage in a multiprocessor application using a *copy back* method for cache coherency. Another example could be identification of non-cached addresses.

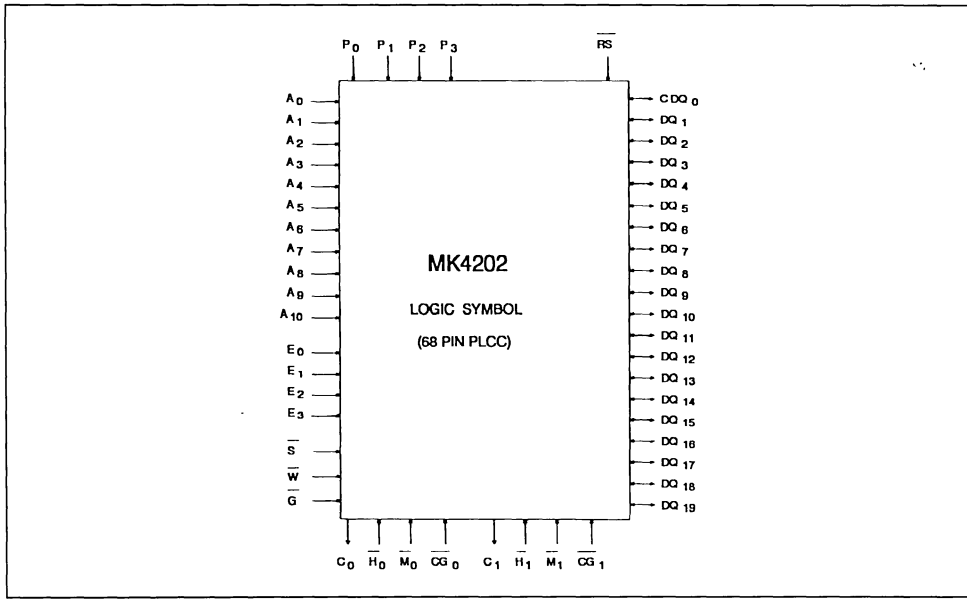
With the introduction of SGS-THOMSON's MK4202 TAGRAM, implementing cache subsystem designs for a 32-bit processor has been made easier. Cache subsystem implementation can now be provided for small-to-medium microcomputer systems without a large investment. The MK4202 TAGRAM provides for cost effective, high speed cache memory designs for today's 32-bit microprocessors, and is fully TTL compatible on its inputs and outputs.

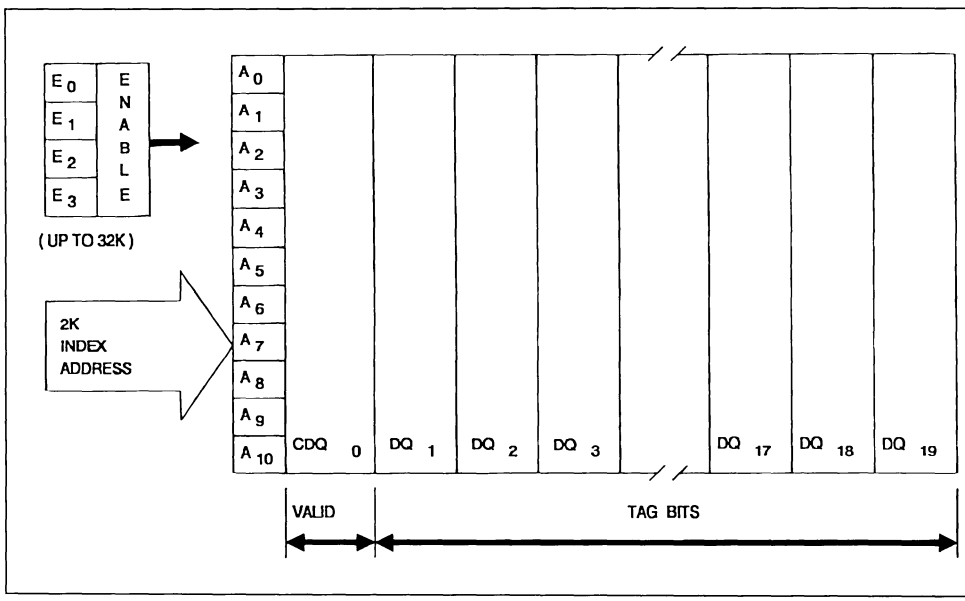

32-BIT DATA BUS
32-BIT ADDRESS
32-BIT CPU
CTRL
WAIT
+5
VALID
11
CDQ0
A0-A10
E0
DQ
16
TAG A16-A31
32
MAIN MEMORY
12
INDEX
41H68 4KX4 20ns
CACHE DATA RAM (MK41H79*)
A11
MK4202
RS
C0 C1
MATCH
CACHE& TAGRAM R/W
MAIN MEMORY R/W
MEMORY & CACHE CONTROL LOGIC


3FF 4096
3FE 4095
3FD 4094
3FC 4093
007 7
006 6
005 5
004 4
003 3
002 2
001 1
000 0
15 0 31 24 23 16 15 8 7 0
TAG
DATA


A
A
TAGRAM
A/D
VALID
MATCH
CACHE
A
D
DATA
RAM
PROCESSOR
MAIN
MEMORY
CONTROL
WAIT
CONTROL
CONTROL
D
D


P=0001
P=0000
INDEX
11
A0-A10
INDEX
EXTENSION
1
E0-E3
TAG
19
DQ1-DQ19
VALID
1
CDQ0
SELECT
S
WRITE
W
READ
G
MK4202
BUS ERR
M 0
HALT
M 1
V CC
or
FORCE
HIT
H 0
H 1
V CC
V SS
CG 0
CG 1
C 0
C 1
C 0
C 1
V CC
V CC
32-BIT
CPU
BERR
HALT
MATCH
ERROR
MEMORY & CACHE
CONTROL LOGIC


P0 P1 P2 P3
RS
A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10
E0 E1 E2 E3
S
W
G
MK4202
LOGIC SYMBOL
(68 PIN PLCC)
CDQ0 DQ1 DQ2 DQ3 DQ4 DQ5 DQ6 DQ7 DQ8 DQ9 DQ10 DQ11 DQ12 DQ13 DQ14 DQ15 DQ16 DQ17 DQ18 DQ19
C0 H0 M0 CG0 C1 H1 M1 CG1


E0 E1 E2 E3
ENABLE
(UP TO 32K)
2K
INDEX
ADDRESS
A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 A10
CDQ 0
DQ 1
DQ 2
DQ 3
DQ 17
DQ 18
DQ 19
VALID
TAG BITS