AN-730
Application Note

A HIGH SPEED FIFO MEMORY USING THE MECL MC10143 REGISTER FILE

Prepared by
Bill Blood
Computer Applications Engineering

First in/first out memories are commonly used to store information in digital processing systems. However, these memories can also be designed to interface subsystems operating asynchronously or at different data rates. This application note describes a high speed first in/first out memory design based on the MECL MC10143 8 x 2 Multiport Register File.

MOTOROLA *Semiconductor Products Inc.*

A HIGH SPEED FIFO MEMORY USING THE MECL MC10143 REGISTER FILE

INTRODUCTION

The first in/first out memory (FIFO) is an important building block for modern digital systems. In computing systems, the FIFO is used for stack registers where register outputs are sequentially read in the same order that data was entered. A FIFO also simplifies many information handling operations such as high speed compiling and code conversions.

Equally important to internal use in a system is the performance of a FIFO as an interface element between two subsystems. It is possible the two subsystems will not share a common clock and the FIFO must operate asynchronously. In this case, data input is controlled by the device supplying information and the read function is controlled by the system using the information. Also, the two systems can be operating at different data rates where several words are stored prior to being read. For example, a lower speed peripheral system may load data at a slow rate to be read by a CPU in a high speed burst.

The first in/first out memory has both advantages and disadvantages when compared to the conventional random access memory (RAM). The RAM allows a system to address any memory location, to write in new data, or to read existing stored data. Data is not lost to memory during a read operation and can be used as often as necessary. The random access memory gives a system maximum flexibility for reading stored data, but requires keeping knowledge of where in memory information is stored and addressing that location.

For applications where it is desirable to read out information in the same order that it was written, a FIFO greatly simplifies system operation. Data to be written in the memory is automatically stored in the next available location and a read operation advances the outputs to the next unread memory word. Once the read is advanced, the previous word normally cannot be used again. The big advantage of a FIFO is the absence of external memory addressing. It is automatically controlled within the memory, requiring only data inputs, read outputs, and the associated clock lines.

This application note describes some of the required features in a high speed FIFO design, especially when asynchronous data entry and read are required. The different approaches to FIFO design are compared with respect to system performance and a design is shown using the MC10143 Multiport Register File as a basic FIFO memory element.

IMPORTANT FEATURES IN A FIFO

As with any memory system, read access time is an important FIFO parameter. An excessive read time directly delays the response of a system to a FIFO input. In some communication systems, the rate at which information can be entered into or read from a FIFO is also important. This is especially true when different data rates are required between the input and output of the FIFO.

Another important feature to measure the performance of a FIFO is the ease with which data can be entered into the memory. Ideally, FIFO read and write operations should be completely independent of each other to system timing. This is especially true with asynchronous operation where the data inputs and write clock must always be ready to accept data. Any time the read cycle interferes with the ability to write in information, the memory cannot accept asynchronous data without additional buffer circuitry.

Status indicators should be the only limiting factor controlling the ability to perform a memory read or write. The EMPTY status indicator tells that the memory is empty and any read command would result in meaningless data. The FULL status indicator shows that the memory capacity is filled and any additional write command would result in a loss of some of the information stored in the FIFO. Also useful is a HALF-FULL indicator as an overall guide to memory status.

FIFO STORAGE ELEMENTS

One method of building a FIFO is with shift register circuits. By shifting data through parallel shift registers (one shift register for each bit and a separate register for status) the proper first in/first out order is assured. When data is entered into this type of FIFO, it is automatically shifted to the last register stage. The next entry is shifted into the second to last stage (last empty register stage) and so on until the register is full. The status shift register is used to identify full register locations so data is shifted only to the last empty stage. This requires that each flip-flop element in the shift registers have a gated clock so that new data is shifted down only the empty locations. After data has been read, a read clock shifts all registers one bit and the FIFO is ready for the next read or write operation.

Although the shift register approach minimizes addressing logic, it has a major drawback when high performance is required. The worst case read time after data is entered

is limited by having to clock data the full length of the shift register. Improvement can be gained by going to high speed shift registers, but the technique is still limited by the number of clock cycles from data in to data out.

The long delays of a shift register FIFO can be reduced by designing the memory with RAM circuits as storage elements. This requires keeping track of the next location to be written into or read from with additional circuits, usually binary counters, within the memory system. Most random access memory circuits have a single set of address lines to select the memory location. This requires sharing the address lines for both read and write operations as shown on the simplified diagram in Figure 1. When address lines are shared for read and write, it is not possible to write into the FIFO while reading an output. This leads to complicated timing or additional storage circuitry for asynchronous data entry.

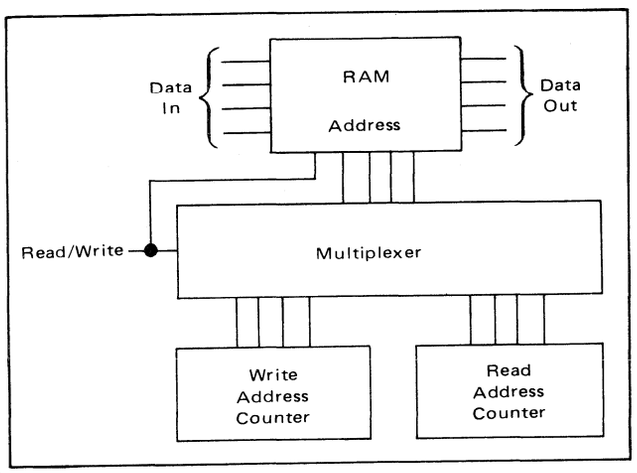

FIGURE 1 – Multiplexed Address Lines with a Single Port RAM

By designing with a multiport register file (RAM) the long access times of the shift register and the addressing problem of the single port RAM are both eliminated. The multiport register as illustrated by the MC10143 in Figure 2 has one set of address lines for writing data into the memory and two independent sets of read address inputs for reading any memory location. Although only one set of read address lines is required for a FIFO design, the other read address inputs can often be used to improve the total memory capability. Examples of uses for the third set of address lines include reading the preceeding or following address from the normal FIFO output, or scanning memory contents.

The read and write address ports of a multiport register are used independently in a FIFO design. The write address lines are always addressed to the next memory location into which data will be entered. In this manner, the write inputs are always ready to clock in data and there are no delays or address set-up times.

The read address lines are selected to the address currently being read. When the next memory location is required the read clock is pulsed and the next word can be read after the memory read access time. This is an ideal condition for asynchronous data entry since information can be entered into the memory at any time, regardless of status of the read outputs and read address.

A HIGH SPEED FIFO DESIGN

A high speed FIFO design based on the MC10143 register file is shown in Figure 3. Each MC10143 is 8 words by 2 bits, and this design incorporates the write and read enable inputs to cascade two register file packages to a memory depth of 16 words. The illustrated circuit is only 4 bits wide, but is easily extended by adding additional memory circuits (two MC10143s for every 2 bits). Since the additional circuits are driven in parallel with the 4 bits shown, no additional logic is required for addressing or status detection when extending the memory to larger bit widths.

The MC10143, Figure 2, features full master-slave flip-flop operation to allow simultaneous read and write. The write address lines A_0-A_2 select one of the eight flip-flop master stages available to each bit. While the clock is low, data is free to enter the selected master stage. When the clock is high the master flip-flops are inhibited and any change on the write address lines or data inputs has no effect on memory information. The read address lines B_0-B_2 and C_0-C_2 independently select one of the eight master flip-flops for each bit and transfer its contents to the associated slave flip-flops. Data transfer from master to slave occurs only while the clock is high. When the clock is low, data is latched in the slave and cannot be altered by changes in either written data or read address lines. This master-slave operation allows data to be entered into the memory at any time, regardless of read status, by applying a negative going pulse to the clock line. In addition, read address can be changed at any time, but output data updates to a new read address only when the clock is high.

Operation of the FIFO in Figure 3 is as follows: Reset is applied following a power up to reset both address counters to the empty state. Reset need not be used again in normal memory operation.

Data is presented on the A through D inputs. Any time it is desired to enter data, the write clock input is enabled with a negative going pulse. The positive edge of this write clock pulse latches the data into memory. The only time data should not be changed is during setup and hold times around the positive going clock edge. With the MC10143 register file, this time is typically less than 3 ns.

Write addressing is controlled by a MC10178 binary counter. The first word entered after power up is stored in address 000. Since the positive edge of the write clock also toggles the write address counter each data entry will automatically increment the counter to the next memory location. Both the Q_3 and $\overline{Q}_3$ are pinned out on the MC10178 and this allows the register files to be enabled without additional decoding logic. The maximum rate of data entry is limited by the propagation delay of the write counter and the address setup time of the register

files. For the MECL circuits shown, this is typically under 14 ns, allowing a data entry rate in excess of 70 MHz.

Following a power-up reset, the read address is selected to read the first memory location. Until data is initially entered, the outputs Q_A through Q_D will be random depending on the state of the MC10143 internal latches after power up. This does not present a problem as the EMPTY indicator can be used to show there is no valid information in the memory. Once the first data entry is made, the Q outputs will automatically display this data and the EMPTY condition no longer exists.

This data will continue to be read, regardless of additional information being placed in the memory, until the read clock is pulsed. Following read clock the memory advances to the next read location. The delay time from read clock to valid data out is the memory read access time, and would typically be under 20 ns for the circuits shown (MC10178 propagation delay plus MC10143 access time). An exception to the 20 ns access time would happen if the write clock is low while the next read location was addressed. Under worst case timing, possible with asynchronous data entry, this could increase read access time by the write clock pulse width. With a 5 ns write clock pulse width, the longest typical read time would be under 25 ns and this occurs only with worst case timing between the write and read clocks.

As a minimum system, a FIFO could be made only with the MC10143 register files as storage elements and two binary counters for address control. However, memory status indicators are normally desired to insure memory capacity is not exceeded. The key indicators are memory empty and full, although a half-full indication is also sometimes used.

FIFO designs often incorporate up/down counters to detect status. The write clock initiates a count up and the read clock a count down. When the counter reads zero, the memory is empty; if the counter reaches maximum word depth, the memory is full. There are two major problems with this approach. First, the up/down counter cannot receive simultaneous read and write commands as could occur with asynchronous data entry. Therefore, a read command interferes with the ability to write data

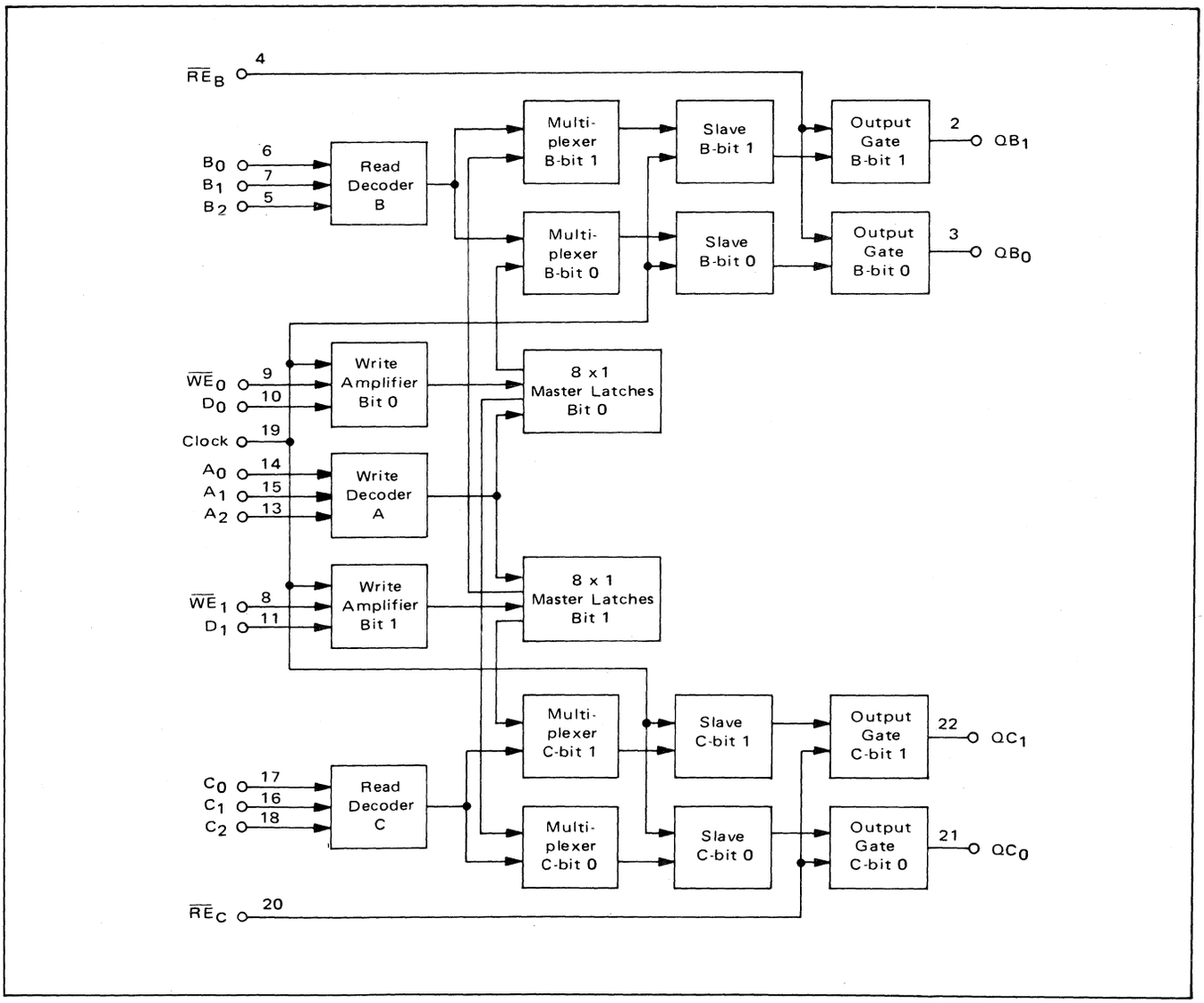

FIGURE 2 – Block Diagram of MC10143 Register File Circuit

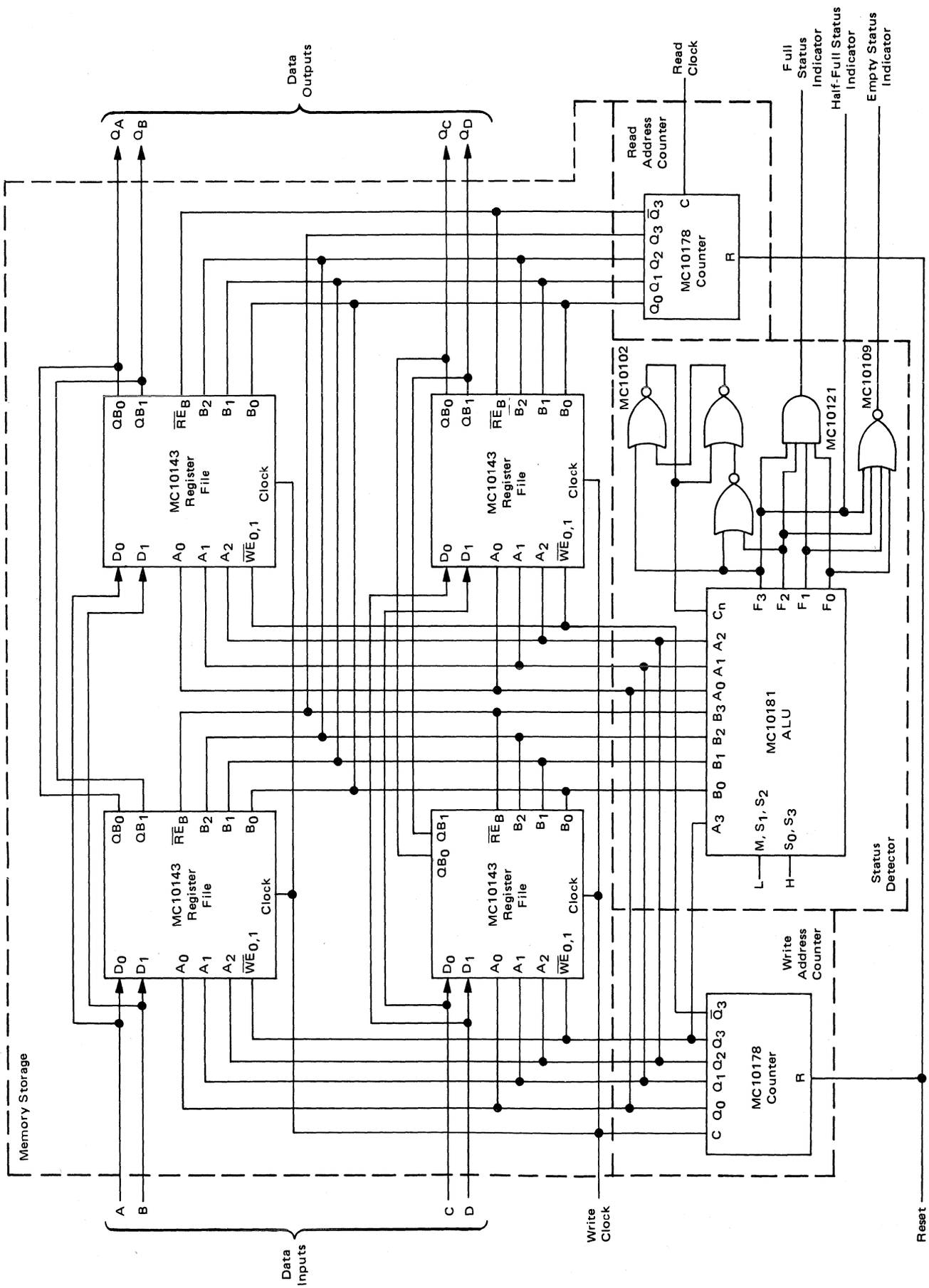

FIGURE 3 – High Speed MECL FIFO

while the counter is decrementing. A second problem is that if the up/down counter misses a count because of this, the status detection is not self correcting and the circuit could repeatedly indicate an incorrect empty or full.

These problems are overcome by replacing the up/down counter with a binary subtractor such as the MC10181ALU in Figure 3. This circuit reads the count difference between the two address counters. When the address counters are at the same binary number, the FIFO is either empty or full. A high logic level on C_n causes the MC10181 to perform the A-B function, and equal address inputs result in a binary 0000 on the function outputs. This count 0 is detected by the MC10109 NOR gate for an EMPTY status. A low logic level on C_n causes the MC10181 to perform the A-B-1 function, and equal address inputs result in a binary 1111 on the subtractor outputs. This count fifteen is detected by the MC10121 AND gate for the FULL status indicator. The only additional circuitry required is a control for the subtractor C_n input. The FIFO design in Figure 3 uses three gates from an MC10102 interconnected as an R-S flip-flop for this control. An MC10181 output of count 8 or greater sets C_n low and a count of 3 or less sets C_n high. HALF-FULL status is easily obtained from the F_3 output of the MC10181.

CONCLUSION

Multiport memories as storage elements open up a new level of performance in FIFO designs. These circuits are easily extended to larger memory sizes and operate with a minimum of addressing control logic. The separate write and read address lines of these register file circuits eliminate any critical timing between writing new data and reading existing stored information. In many computer and communication systems, shift register storage circuits can be converted to multiport registers with an improvement in both performance and part count.

MOTOROLA Semiconductor Products Inc.

Printed in Switzerland

AN-730