The Ins And Outs Of Volatile Memories

by
Don Lancaster
Synergetics

Don Lancaster provides us with this discussion of some of the read/write memory techniques which are available to experimenters using readily available parts. In this background tutorial, Don discusses memory techniques from the simple gate flip flop to the bus-oriented RAM system using static memory circuits. For more detailed looks at the designs of circuits using some of the techniques in this article, readers should turn to Don's book, The TTL Cookbook, *available from Howard W. Sams, Indianapolis IN. The material in this article is abstracted from Chapter 3 of Don's forthcoming* TV Typewriter Cookbook, *also to be published by Sams.*

Unfortunately, there is no cheap and reasonable memory system available today that will both remember information forever *and* be able to read and write information rapidly, cheaply, and with reasonable timing signals. This is called the volatility problem.

A memory is non-volatile if it remembers forever. A volatile memory loses its contents if you remove supply power or fail to observe any timing restrictions it might have.

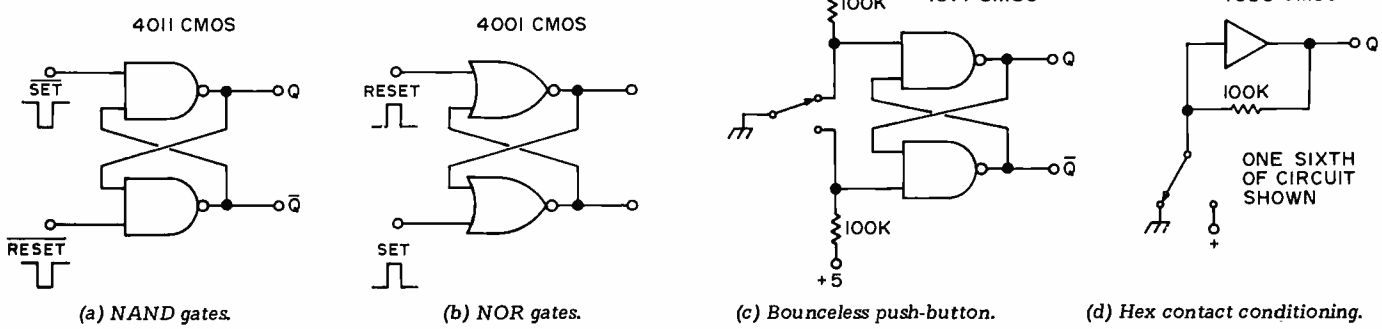
One older and obvious non-volatile memory system, of course, is magnetic core. The problem with core is that much in the way of support circuitry (including sense amplifiers, write after destructive read circuits, system timing generators, power down interrupts, etc.) makes core highly impractical for small scale TV typewriter and microcomputer systems.

Sometimes you can gain non-volatility with a volatile memory by some system level tricks, such as a power down technique that holds a low voltage from a battery applied to the memory when the main supply power goes away. Newer CMOS RAM memories consume almost negligible power in the standby mode and lend themselves well to this. You can also transfer data to some non-volatile outside storage such as a cassette recorder or a magnetic disc.

Read-write memory circuits can have their contents changed rapidly at system timing rates. This makes them useful for storing characters, computer programs, update commands, and anything else we want to temporarily store and recover. Most read-write memory circuits are volatile, holding their information only as long as supply power remains present and so long as any timing restrictions are not ignored.

Read-write memories for TVT and microcomputer use can range from single bit control and debouncing circuits up through thousand word character stores to 16k and 32k microcomputer memory stems. Some of the more important memory types include the set-reset flip flop, the word storage latch, the shift register, the Random Access Memory or static RAM, micropower RAMs, and dynamic RAMs. Let's look at these in turn.

Fig. 1. Set-reset flip flops are the simplest type of read-write memory.



(a) NAND gates. *(b) NOR gates.* *(c) Bounceless push-button.* *(d) Hex contact conditioning.*

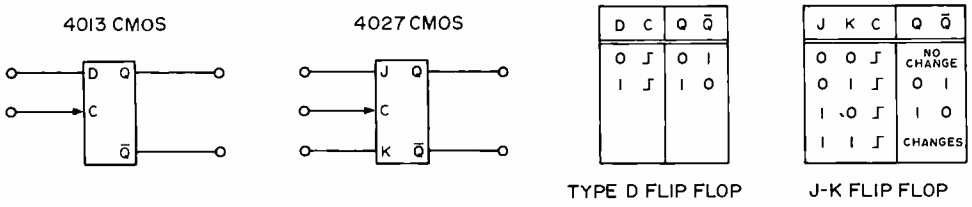


Fig. 2. Clocked flip flops only change in response to a control signal called the "clock".

Fig. 3. Word storage latches provide a handy way to keep track of multiple bits of data.

The Set-Reset Flip Flop

One of the simplest read-write storage systems is the set-reset flip flop. It can store only one bit of information, and may be built using the NOR gates of Fig. 1(a) or the NAND gates of Fig. 1(b). When set, the Q output goes and stays in the "1" condition. When reset, the Q output goes and stays in the "0" state. A complementary or $\overline{Q}$ output is also supplied – it's a "1" when Q is a "0" and vice versa. Simple set-reset flip flops have a hangup in that if you simultaneously try to set and reset them, they go into a disallowed state, and the final way they end up depends on the last input to be released.

Fig. 1(c) shows us how to use a set-reset flip flop to eliminate the mechanical noise and bouncing of a SPDT push-button or mechanical contact. The first instant the contact makes, the flip flop jumps to the "1" state and stays there till the first instant after the switch is completely released, eliminating any bounce, noise or chatter. Circuits of this type are absolutely essential anytime you want to enter data from a switch or mechanical contact into any digital system. Fig. 1(d) shows us a simple hex contact conditioner using one CMOS non-inverting buffer.

A major use of set-reset flips flops is to debounce switches.

The Storage Latch

Operation of a set-reset flip flop is nearly instantaneous. If we tried to cascade a bunch of them, we'd get an unchecked wild race, for after changing the first one, the rest may follow domino style. It's much better to have digital circuits change only when you want them to and then do so on a one-stage-at-a-time orderly basis. To do this, we go to a clocked flip flop, such as the type D or JK flip flops of Fig. 2.

With these devices, the D input or the JK inputs set up what the flip flop is to do, but the actual change isn't carried out till a certain edge or level of the clock input happens. With a "D" flip flop we can clock in a "1" or clock in a "0", most often on the positive edge of the clock. We can also divide by two with a D flop by cross coupling the $\overline{Q}$ output to the D input, making the logic block change states every clocking. With the JK flip flop, we have the options of clocking in a "1", a "0", doing absolutely nothing, or changing each and every state as a binary frequency divider. For convenience in use, most JK and D flip flops also have extra direct set and reset inputs; these operate immediately and are useful for clearing or initializing memory states.

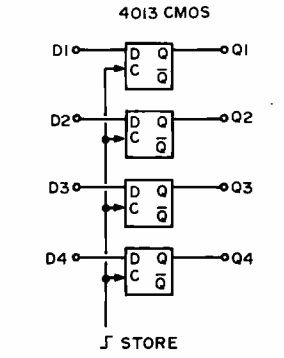
Word Storage

One flip flop can only store one bit at a time. If we use flip flops in groups, we can store a multiple bit word at once. In Fig. 3(a), we use four D flops to store a 4-bit word. Data is entered on the leading (positive) edge of the clock. Figs. 3(b), 3(c) and 3(d) show how we can use larger MSI logic blocks to store words of four, six and eight bits in length.

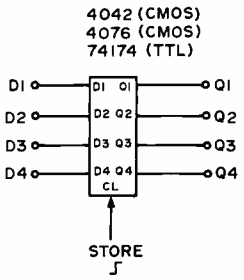
A word storage latch of this type is often handy to "catch" an input signal on the way by and hold it till you can use it. For instance, a microcomputer may output a word for only a microsecond or two, but your TVT circuit may not get around to using the word for milliseconds or even seconds later. In this case, you catch the microcomputer's output word with a storage latch and then keep it till you are certain you have used it. You then release the latch and ask the microcomputer for a new word through a handshaking signal.

Another important use for word storage is to resynchronize data and make sure it is valid. As an example, suppose your system changes words every microsecond, but that the

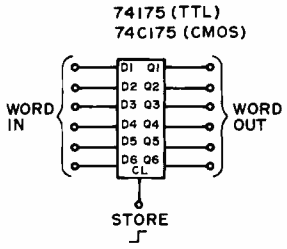
One bit of memory is one set-reset flip flop.



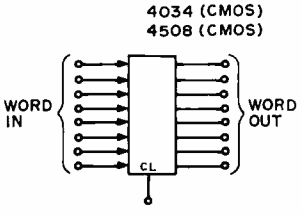
(a) 4-bit word using D-flops.



(b) 4-bit word using MSI.



(c) 6-bit word.



(d) 8-bit word.

previous block that's giving you inputs takes 900 nanoseconds or so to get around to giving you a valid output after *its* inputs change. This only gives you 100 nanoseconds or so of valid data and may change with temperature or supply voltage. Add a storage latch and you can catch this output and hold it for the entire next microsecond. The output data will always be one microsecond late, but it will always be valid and always be locked to your system timing. A word storage latch of this type may be needed between the memory and the character generator of a TV typewriter if very long line lengths are in use.

Shift Registers

We can cascade a stack of type D flip flops so that they pass on their contents one stage to the right each clocking. This is called a shift register, and Fig. 4 shows several examples.

In Fig. 4(a), we've built a four stage register out of type D flip flops. Each clocking passes data one stage to the right. In Fig. 4(b), we've added some enter-recirculate logic to let us either send the data round and round or else change selected bits at once. We can make the register any length we like by adding extra internal storage flip flops.

A shift register is the digital equivalent of a "delay line" - - your bits go in now and come out later. An "electronic disk memory" is an extremely large memory made out of long shift registers - and is programmed by the computer almost exactly like an old-fashioned fixed head magnetic disk.

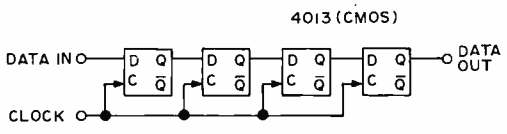
In Fig. 4(c), we use a MSI integrated circuit arranged as an 8-bit parallel-in-serial-out shift register. Shift registers are useful to convert dot matrix dots to serial video in a TVT; they are also handy for converting data from parallel to serial form. Fig. 4(d) is the opposite; this is a serial-in-parallel-out shift register useful to convert serial data into parallel form. Finally, Fig. 4(e) shows us a 1024-bit serial-in-serial-out register, useful for storing bulk data. Use several of these side by side if larger words are to be stored. For instance, to store 1024 ASCII characters, six of these could be used side by side. They can also be cascaded end to end for 1024, 2048, 4096 and more bits per word.

There are lots of long MOS shift registers available. Other popular bit arrangements include the hex 32-bit and hex 40-bit shift registers, the 2518 and 2519. On the surface, shift registers would appear to be ideal for character and program storage in TV typewriter circuits. One of the earliest TVTs (see September 73 *Radio Electronics*) made extensive use of shift register storage, and similar registers are still used in many premium computer terminals.

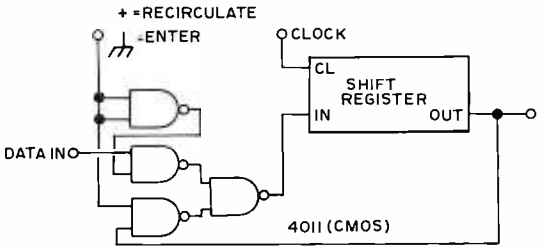
Today, we usually have a far better approach to data storage in the static random access memories of the next section. While these long shift registers were the first truly low cost semiconductor storage and are still useful for certain applications, they do have problems and the RAM techniques are often better.

Many of the early shift registers were dynamic devices in which you had to keep the data moving above some critical rate. Most early clocking circuits required a waveform that had to come from a fast, high current, noisy clock driver, swinging 17 volts or more with very strict pulsewidth and spacing

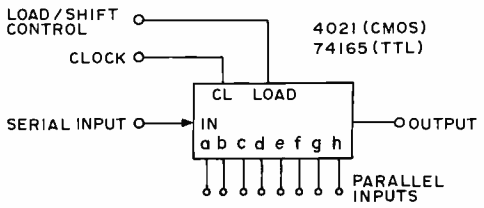
Fig. 4. Some shift register memories. Modern RAM memory devices tend to make shift registers obsolete except for special applications.



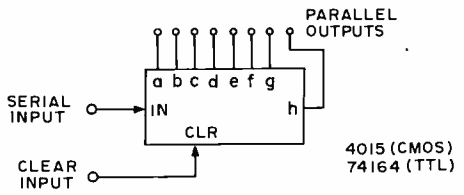
(a) Four stage using type D flip flops.



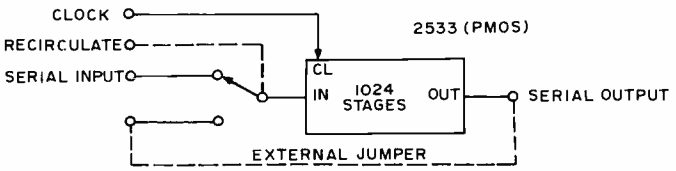
(b) Recirculate logic.



(c) Parallel in, serial out, 8-bit.



(d) Serial in, parallel out, 1024-bit.



(e) Serial in, serial out, 1024-bit.

Set-reset flip flops are asynchronous—in systems where outputs affect inputs, use of an SR latch can send you off to the races—an uncontrolled oscillation.

restrictions and any pulse overshoot strictly forbidden. Some earlier multiplexed shift registers, particularly the 1402, 1403 and 1404 exhibited selective dropouts called bit pattern sensitivity that would change data if the particular IC didn't happen to like the combination of clocking waveforms, supply voltage, temperature and internal data that it happened to have on hand.

Newer static N channel shift registers have eliminated most of these problems, but they still have one key drawback: This is simply that you can't immediately get at the data you want. The memory must be clocked around once and exactly once to get back any bit. All the other bits are usually between you and the bit you are after. With the RAMs of the next section you can selectively pick off any bit at any time, rather than waiting till it comes around. More importantly, you can be extremely sloppy about your timing and addressing between the times you are actually using the data, and the RAM doesn't care. With a shift register, one missed timing pulse is a disaster. Additional limitations of shift registers as bulk storage devices are that they often cost more at system levels than do RAMs and that they are not directly microprocessor compatible.

A bit is a bit. A word is "n" bits contemplated simultaneously.

Two areas where we can continue to see shift registers as important TVT and microprocessor parts are in First In First Out or FIFO buffer memories such as the Fairchild 3341 (64 x 4) and the 33512 (40 x 9) and similar devices by AMI and Western Digital, and in the newly emerging charge coupled device bulk storage systems such as the Fairchild CCD450 (1024 x 9) and CCD460 (128 x 32 x 4) devices. A FIFO gives us a way to interface two systems having different data rates, while the CCD devices promise to eventually provide very low cost and dense bulk storage.

Random Access Memory (RAM)

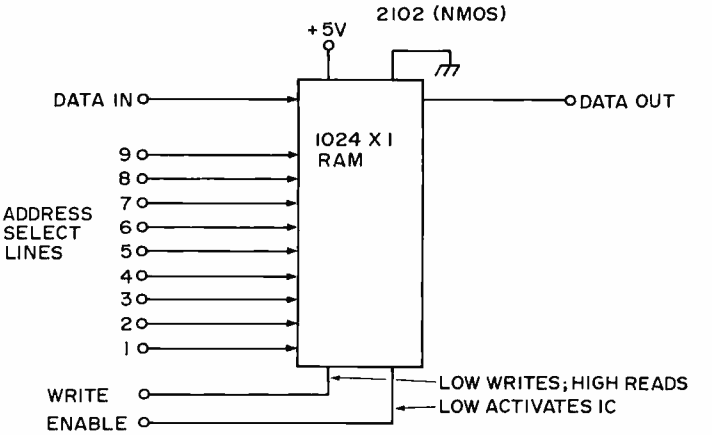
A random access memory or RAM differs from a shift register in that we can get to any memory location any time we want. If we like, we can address our RAM shift register style, working with the storage cells in sequential order. But we don't have to — we can get at any cell at any time in any order.

To do this, some external binary address lines are routed to internal decoder and selector circuits. When a memory cell is addressed, it is available either for reading as an output or for writing new information into it. Most RAMs tend to be only a single bit wide to save on package pins, but 4- and 8-bit words are sometimes offered.

There are lots of RAMs available. Bipolar types using TTL technology are usually fast and expensive. They generally provide less dense storage and fewer bits per package. Two examples are the 7489 arranged as 16 words of four bits each and the 74200 arranged as 256 x 1, or 256 words of one bit per word. Both cycle in under 50 nanoseconds.

MOS memories using PMOS, NMOS and CMOS are also widely available. These are often slower and cheaper per bit. They usually offer more bits per package, with up to 4096 bits being common and 16384 just being talked about at this writing. Two early and essentially obsolete examples of MOS memory were the 1101 that was a fully static 256 x 1 device and the 1103 that was a dynamic RAM organized 1024 x 1 that singlehandedly toppled "king core" from the computer world. The early 1101s ran extremely hot with weird supply voltages, while the early 1103s had incredibly complex clocking, refresh, and timing restrictions, besides being bit pattern sensitive.

Fig. 5. The 2102 is the ideal RAM for many TVT and microcomputer uses.



After several generations of much improved MOS memories, a device called the 2102 arrived and dropped enough in price to often be the best all around choice for TVT character storage and most smaller microcomputer memory tasks as well. The 2102 is arranged as 1024 x 1. It is N channel MOS and works on a single +5 volt supply and is fully compatible with TTL and CMOS logic on all pins. It is fully static, needing no clocks, refresh, charge pumps, memory busy interlocks, sense amplifiers, or similar garbage. Economy versions of the 2102 cycle in one microsecond, while premium jobs are available with 200 nanoseconds or less cycle time. Even the slowest 2102s are usually more than adequate for TVT use, although a bit slow for the newer microprocessors.

Best of all, at this writing, 2102s cost under $5 in singles and as low as $2 in large quantities. While designed by Intel, sources today include just about everybody — TI, Intersil, AMD, National, Signetics, Fairchild and Synertek.

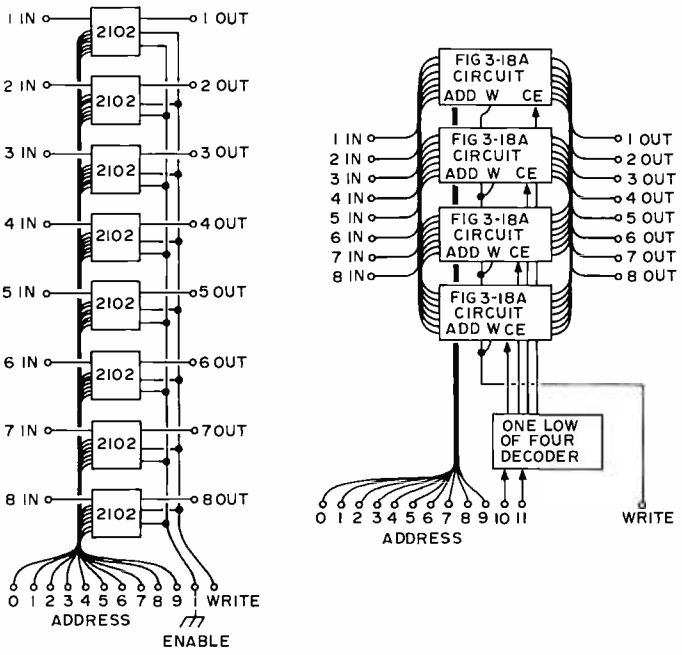
Fig. 5 shows us the connections to a 2102 memory. We see a data input pin, a data output pin, 10 address lines, a write line, and an enable line. The data in

and data out lines are the same sense, meaning that a "1" input is stored as a "1" and appears as a "1" at the output. Our 10 address lines select one of the 1024 storage cells by providing binary addresses ranging from 00000-00000, 00000-00001 . . . through 11111-11110 and 11111-11111. We can mix up input address lines any way we want so long as all packages and all input address timing circuits agree on what address combination goes with what storage cell. Jumbling the memory address inputs sometimes helps the circuit layout and makes such things as efficient BCD (Binary Coded Decimal) rather than binary addressing feasible.

Our chip enable controls whether the memory will do anything. If the chip enable is high, the data out line assumes a floating, high impedance, tri-state mode, and the write input logic is disabled. If the chip enable is low, the IC operates normally. *Chip enable should stay grounded except in very special uses.* The memory will read if the write input is high and will write if the write input is low.

There is one important timing restriction on the write input – *input addresses must be stable when the write input is low.* To write into memory, apply input data and an input address. After the inputs are stable, bring the write input low for at least the minimum write time. This time varies from 100 to 700 nanoseconds depending upon the device. Then release the write input, letting it go high before you change the address inputs or the data inputs. If you try to change addresses during the write process, certain memory locations may get "flashed" by in the address decoding and could lose or alter data. Incidentally this write-only-when-stable rule applies not only to 2102s – it should be observed with practically all RAM circuits. For normal 2102 read operation, make the write input *high* and the chip enable *low.*

Even the slowest 2101s are usually more than adequate for TVT use – although a bit slow for the newer microprocessors.



(a) 1024 x 8 character store memory. *(b) 4096 x 8 microcomputer memory.*

Fig. 6. Larger 2102 memories are created by repeating the same circuit over and over.

Fig. 6 shows how we build a 1024 x 8 memory good for TVT character storage. Eight 2102s are simply put on the same PC card with their write, address and enable lines in parallel, the latter usually grounded. All inputs and outputs go to separate pins, and this way we can read or write 8-bit words. In Fig. 6(b), we've combined this circuit four times over to get a 4096 x 8 static memory suitable for a microprocessor main memory and big enough to hold a small compiler for an elementary higher level language. Two new address lines are one-of-four decoded and routed to the chip selects of each quarter of the memory. One chip select is made low and the other three remain high, and the tri-state outputs are shorted together as shown. A total of 32 ICs is needed.

When building either type of memory card, lots of 0.1 microfarad bypassing capacitors are recommended, along with wide supply and ground runs. PC layout is usually simplest with a double sided board and through-the-pins lead routing. Use plated through boards and keep the through-the-pins routing on the component side if possible.

Reorganized 2102s

Sometimes shorter words of more bits per word may be desirable. For these applications, some manufacturers have reworked the 2102 into different organizations including 128 8-bit words and 256 4-bit words. The Motorola and AMI 6810 are typical 128 x 8 units in a 24-pin package. The Signetics 2606 is a 256 x 4 version in a 16-pin package. Since there aren't enough pins to go around, the 2606 shares common input and output lines and must be used with a bidirectional data bus system. The Intel 2101 is a 22-pin version of the same thing with separate input and output pins, while the 2111 is an 18-pin memory that needs a bidirectional input/output system. At this writing, costs of these devices are considerably higher than conventional 2102s and are likely to stay that way since they have larger packages and less availability. Nevertheless, they often save you enough packages to be worthwhile in smaller systems. Having only 8 address bits makes the 256 x 4 memories directly compatible with 8-bit microprocessors as well.

Micropower Static RAMs

We can also build CMOS random access memories similar to the 2102. CMOS has one major advantage – its standby power needed when the memory is not cycling is almost zero. This makes CMOS memories ideal for "non-volatile" storage where a small battery can fill in for extremely long term data holding, as well as safely handling routine power outages. CMOS memory cells tend to be physically larger than NMOS ones and more process steps are often involved. So, CMOS memories will probably remain a more expensive route, but a very attractive one where micropower memory is essential. Obvious applications include electronic checkbooks and remote data acquisition systems.

Typical devices are the 64-bit Motorola 4505; 256-bit devices including the RCA 4061, Intersil 6523, and Motorola 4532 (the latter is arranged 64 x 4); 512-bit versions including the Nortec and AMI S2222, and the Intel 5105, arranged as 1024 x 1.

Dynamic RAMs

A static RAM takes a full memory cell for data storage. We can get by with nothing but a capacitor as a storage device if we are willing to reshuffle, move around, or refresh the stored data more or less continuously. This is the principle behind the dynamic RAM. In exchange for cheap and dense storage, system level restrictions in the way of memory busy times, refresh cycles, clock lines, and clocking restrictions are needed, often combined with analog output sense amplifiers. Traditionally, any particular size RAM starts out as an impossible to use dynamic device, upgrades itself into a very difficult to use "quasi-static" device, and then gets replaced with a static no-hassle IC the third time around.

Because of this, dynamic RAMs should be avoided entirely for all TVT and microcomputer usage. While there are a wide variety of yet unstandardized 4096 x 1 dynamic RAMs on the market, including the Electronic Arrays 1504, Intel 2107, Standard Microsystems 4412, TI 4030, Mostek 4096, and the Motorola 6605, they presently cost much more than the equivalent storage using 2102s and are harder to get and harder to use. They do have the potential advantage of reducing package count 4:1 in very large memory systems where the 4096 x 1 format can be used, and are ideal for larger computer memories.

Traditionally, any particular size RAM starts out as an impossible to use dynamic device, upgrades itself into a difficult to use "quasi-static" device, and then gets replaced with a static no-hassle IC the third time around. Home brew computer people for the most part have to wait for the third stage or have a talent for wrestling with difficult hardware problems. At this writing, 1k memories are in stage 3, 4k memories are in stage 2, and 16k memories are at the beginning of stage 1.

Bus Organization

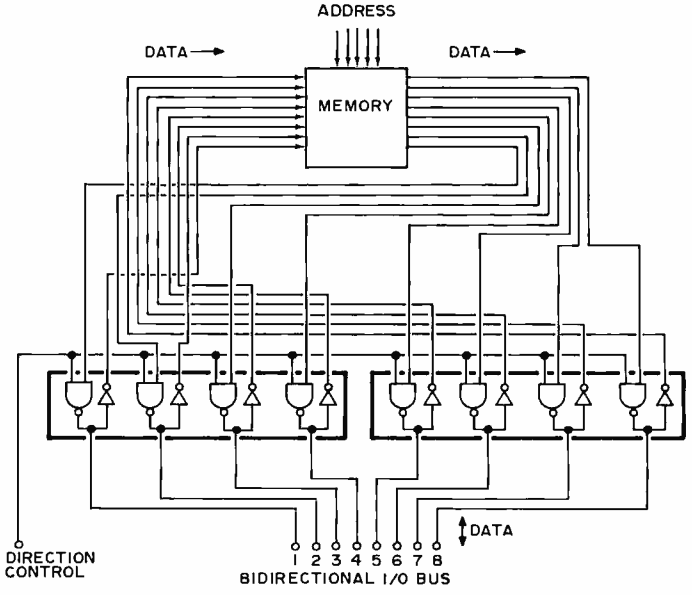
Any memory system has input data lines, output data lines, and address data lines. It's usually simplest to keep these lines completely separate, for this way there are no timing commands needed to separate input, address and output signals, and no times can occur when one would interfere with the other. This is called an isolated bus or separate I/O system.

Many microcomputers instead use bidirectional data bus arrangements to save on pins and interconnections. The data bus just sits there. If something wants to transmit, its tri-state output is enabled. If something wants to receive, its input is enabled. The signals can go either way on the bus, but system timing has to make certain that only one source is transmitting and only the receivers for that source are responding to the transmitted information.

We can convert an isolated bus system into a bidirectional bus system using bus transceiver integrated circuits as shown in Fig. 7. Transceivers are built into the 2606 and 2111 (they don't have enough pins to do otherwise), and may be externally added to regular 2102 type memories. In the case of a TV typewriter, it's usually desirable to keep the display and its memory source (either its own or the memory of a microcomputer) connected together; this eliminates dropouts when the bus is going downstream. Thus, the display electronics is best placed between the memory read outputs and the bus transceiver.

Address lines can also share the same data bus as the input and output data lines, but this leads to extremely difficult timing, particularly in 8-bit systems. More often, the address bus will remain separate but will be able to accept address commands from several sources. These sources could include the TV typewriter's live scan timing, a cursor address for update during retrace, and an optional external dominant microcomputer control for rapid and wholesale screen changes.

Fig. 7. Connecting a memory subsystem to a bidirectional input output bus is required in most microcomputer applications.



QUAD BUS TRANSCEIVERS 3440, 26S10, 26S11, 8838, ETC.
NON-INVERTING TYPES ARE ALSO AVAILABLE, SUCH AS THE 8833, ETC.