this board. The additional page-B memory does not need a new single-line memory, character generator, and output video register, as it can borrow the one in the page-A memory when the second page is in use. This is called *bus organization.* The character generator will respond to anything that is enabled on the bus, be it page-A memory, page-B memory, a calculator add-on, or whatever. Of course, we have to be careful to only enable *one* possible source of characters at a time, but this is easy. We can also use the bus optionally to *output* characters to the outside world.

The output of the memory board also contains a video combiner that assembles the character video, sync signals, and the internal test signal into one composite video output. This output may either be used directly or routed to the rf modulator for clip-on operation of an unmodified TV. It can be optionally flashed or blinked.

The TIMING board contains a crystal divider and TTL (Transistor Transistor Logic) countdown chain that generates all the needed signals to run the typewriter in proper sequence. It does not normally use interlace, but the timing chain is split so that the somewhat more complex TV full-interlace system can be added if you need this sort of thing for video titling. There are two principal areas to the timing board, the MAIN timing, and the DERIVED timing. The main timing is the continuous waveforms obtained off the crystal divider, while the derived outputs combine portions of the main timing signals into properly coded signals needed to run the rest of the typewriter. Two examples are the composite sync signal and a blinker used for flashing, cursor winking, and repeat functions.

The third essential board is a CURSOR board. Anyone who ever tried to design and debug a simple one of these will easily understand why it is called a cursor board. Anyway, the cursor keeps track of where the next character is to go; runs the winking line that shows the character position; controls entry of the character; and optionally sets up characters for output. It also contains an input conditioner, and debouncer and a detector for CTRL commands that tells the typewriter to carriage return rather than enter a character.

Many cursor systems are extremely complex. This one is relatively simple in that it uses a *phase-shift counting technique.* The cursor has a *continuously running* counter just like the main timing chain does. Its output drops suddenly in some *relative* position, indicating where the next character is to go.

To back the cursor up, we throw in another count pulse. To run it forward, we hold back one normal count pulse. Thus, the *relative* position or phase of the cursor counter advances or backs up with respect to the system timing. Actually, to go forward, we hold back *two* normal system timing pulses and throw in a new one. This buys us a simplification of circuitry, but still ends up with the same result.

An ADD-SUBTRACT switch on the mainframe controls the cursor direction for editing. LINEFEED is handled by adding and removing the proper number of counts in the proper position in the cursor counter so that the new position is reached. Just like most typewriters, the linefeed always returns to the lefthand side.

In normal operation, each character entry moves the cursor over one character. When it gets to the end of the line, it starts again on the next line. When it gets to the bottom of the page, it starts again at the top. A CLEAR or HOME override also moves the cursor to the upper lefthand position.

And, this is about all you need for a normal parallel entry type of TV typewriter. One possible optional board is a MODEM or frequency-shift-keying interface. This would use a MOS chip and some TTL to convert to or from a serial tone input, suitable for computer or telephone line communication. A cassette recorder will work just as well with the modem for electronic notebook use.

Another possible add-on makes the typewriter into a calculator. This is done by converting the scan from a complete frame to a single line of numerals and would use a surplus calculator chip to provide the familiar calculator functions. If you already have or need the TV typewriter for something else, this add-on is far cheaper than a conventional calculator would be, and its display would be obviously larger and more readible.

Or, you can add most anything else you want onto the machine by tieing into the bus-oriented lines (b1 through b6). For instance you can think of the memory as sixteen registers of 32 numbers each, and those numbers are decimal numbers plus, not bits! With six bits per word, you can store 10 possible numerals and 54 machine commands in any word! Or, you can split the registers into 32 registers of 16 decimal numbers each, building your own computer or programmable calculator.

Of course, this computer-add-on is very much an advanced experimenter project, but it really doesn't take much more than a double handful of TTL to pull it off. While such a computer will be relatively slow (around a 33-ms cycle time), it does provide an extremely accurate and very low cost computer approach, particularly when you are working directly with BCD numbers instead of binary.

Some basics

Before we turn to the actual circuitry, some basics of what a character is and how it can get on a TV screen is in order. Lets start with the characters:

If we had six possible binary bits of either 1 or 0, we would have sixty-four different possible combinations ranging from 000000, 000001, 000010, . . . through to 111110 and finally 111111. These 64 different states can represent all the capital letters, all the numbers, a blank, and most punctuation, following the standard ASCII code. In the TV Typewriter, all of the six bits of this code must be presented at once or in *parallel* form, and this is the *only* code the circuitry shown can use. Other codes can be converted to ASCII before going into the TV Typewriter. A seventh bit is used to separate characters from internal commands.

Our final presented character consists of an array of 5 × 7 dots. Since it only takes 6 bits to store a coded character and at least 35 bits to store a generated one, its obviously much better and cheaper to generate the characters *after* they are stored, rather than before.

For other keyboards and encoders, the

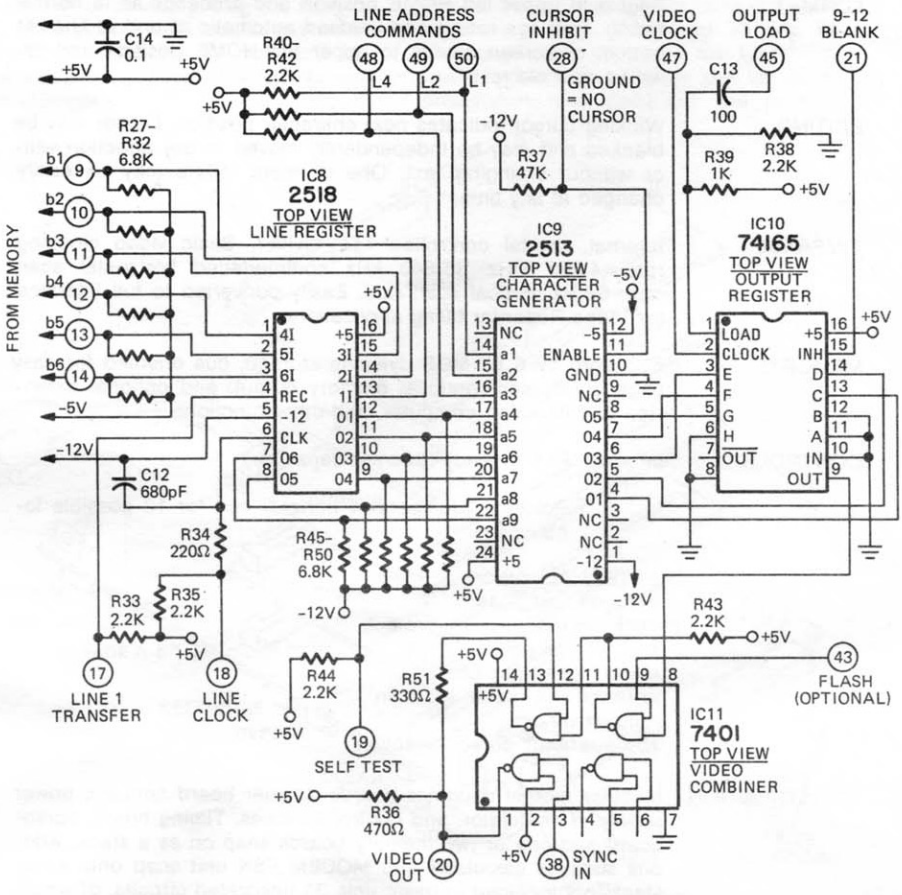

FIG. 2—SCHEMATIC OF THE CHARACTER GENERATOR. the use of integrated circuits greatly reduces circuit complexity and makes possible a very compact device.