

further automation for typewriter-type electronic keys

Some time ago I developed an automatic typewriter-type keyer that features self-completing characters, automatic character spacing, wide speed range and includes special characters used by hams.* In response to several queries the possibility of adding another bank of memory storage within the same keyer case was investigated; this would permit the operator to type ahead of the actual transmission—or to slow down momentarily while hunting for a key—and still produce faultless Morse code.

With an added "buffer storage module" you type at the same average speed as the keyer (as normal), but if you hit two keys close together and a third much later, the keyer sends perfect code. The buffer storage results in several subtle changes. In the basic keyer, for example, a key is held down to repeat a character. With the buffer storage, you tap the key twice just as when using a typewriter to repeat a character. If you press a key before the logic circuits are ready for it, with or without the storage module, you cannot affect the stored

characters; simply hold the key down, and when the memory circuit is clear, the new character is stored automatically, and the key can be released.

With the added buffer storage, two keys can be tapped rapidly and a third key held down until the first character is complete, a fourth key held down until the second character is complete, etc. Your average typing speed is not increased, but you have a backlog of two characters that gives you a much wider operational flexibility.

basic keyer

The basic keyer consists of two circuit boards (matrix and logic board) and a keyboard. The matrix board contains the information on each character and mounts on the keyboard. The logic board stores the information on the selected character, scans this storage at the selected speed, provides the proper spacing between characters and contains the output relay and side-tone circuitry. A simplified block diagram of the basic keyer is shown in fig. 1.

Thirteen wires carry the information from the matrix board to the logic board - one trigger wire to signal the logic board that a key has been pressed and six pairs of dot/dash wires that carry character information. This permits the formation of

Robert L. Kurtz, W6PRO, Micro-Z Company, Box 2426, Rolling Hills, California 90274

*This keyer is marketed as the Pro-Key Kit and is available from Micro-Z Company, Box 2426 Rolling Hills, California 90274 for \$149.50 postpaid. The buffer storage unit described in the article is available as a kit for \$34.50 additional.

characters with up to six dot/dash combinations such as SK, period and question mark. If the "A" button is pressed, a positive voltage appears on the "dot" line of the first pair of wires, on the "dash" line of the second pair of wires, and on the "trigger" wire.

The trigger signal starts the keyer clock and opens the insert gates momentarily to store the selected character in the main storage elements on the logic board. After the gates close, the stored character cannot be disturbed. The logic circuits then scan the main storage to create the Morse character. If a new key is held down, the new character is automatically inserted only after the first character is complete and the proper spacing time has elapsed.

buffer storage module

The additional storage module is inserted between the keyboard and the main logic board (fig. 2). When a key is pressed, the trigger signal is differentiated, stored in the module, and is used to open the module signal input gates so the selected character can enter the buffer memory.

As soon as the selected character has been entered into secondary storage, the trigger memory is erased, and the input gates close. The presence of a stored character in the module generates a trigger signal which is sent to the basic keyer. This starts the clock and inserts the character in the main storage, just as if the signals were coming directly from the

the storage module (after a slight delay). The total process takes only a few microseconds, and the storage module is ready to accept another character from the keyboard. Meanwhile, the logic in the basic keyer is leisurely scanning the main storage to create the first character at the selected sending speed.

Another key can be tapped immediately and the buffer storage process is repeated. However, since the first character is still being transmitted, the second character remains in storage until the main storage in the basic keyer is ready to accept it. The presence of a stored signal

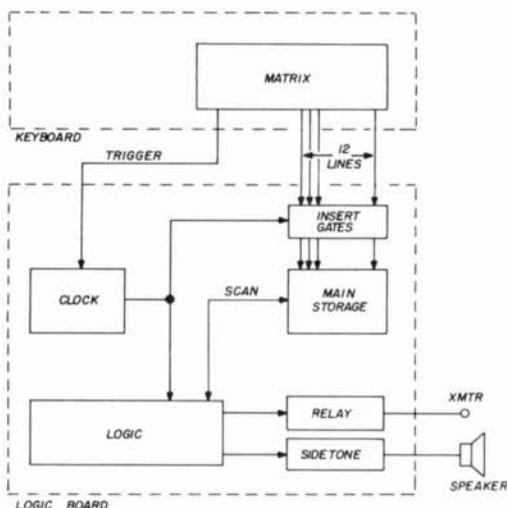
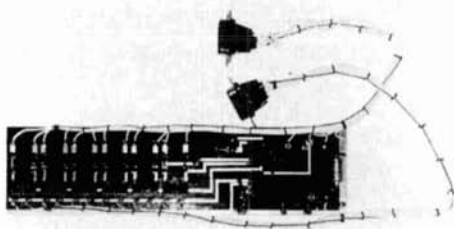


fig. 1. Block diagram of the basic pro-key type-writer-type electronic keyer.



Buffer storage module

keyboard. The pulse that opens and closes the main insert gates on the basic keyer is used to clear the secondary storage in

in buffer storage "locks out" — or inhibits — the storage module input gates so the second character cannot be disturbed until it moves into main storage and the buffer storage is cleared.

A third key can be pressed immediately after the second. When the keyer has completed the first character, the character in buffer storage automatically moves into main storage, and the buffer is cleared. The buffer storage will then accept the third character, the third key can be released, and the operator can move on to the fourth character while

the second is being transmitted.

An entire CQ or DE is stored by simply tapping two keys, just as you would on a typewriter, and the Morse characters are

des are raised above ground. This is accomplished by a positive trigger signal from the keyboard operating through the trigger storage flip-flop and inhibit

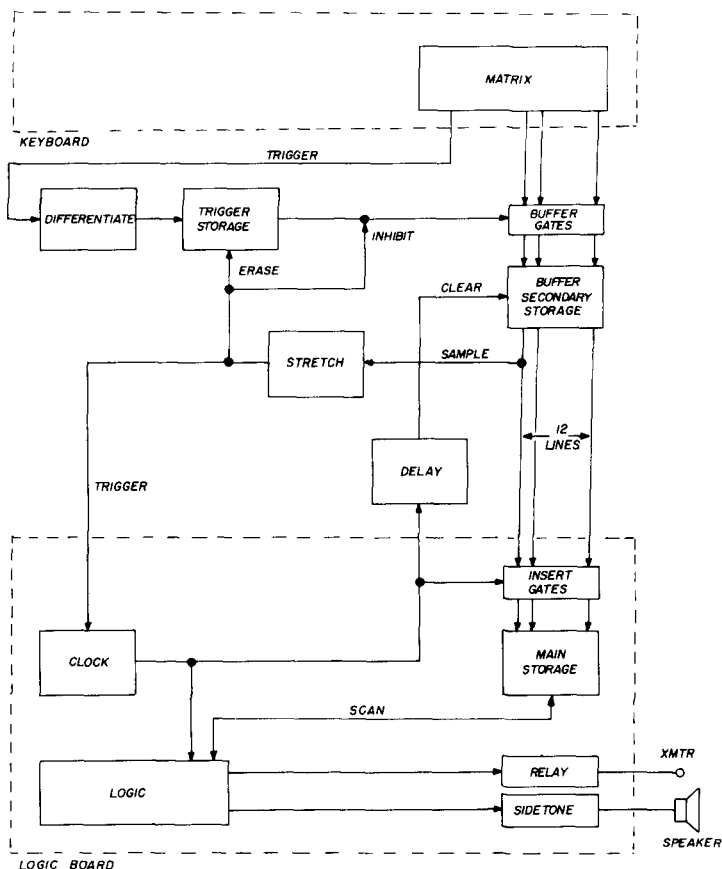


fig. 2. Block diagram of the keyer with the added buffer storage module.

transmitted at the speed set by the operator. The operator can speed up or slow down his typing but the keyer transmits smooth, even code. This is particularly useful for hunt-and-peck typists who use a typewriter-type keyer at 20 to 25 wpm.

circuit

In the logic diagram of the buffer shown in fig. 3, only three of the twelve storage RS flip-flops are shown for simplicity. Keyboard signals set the proper storage flip-flops to provide a positive output only if the cathode ends of the gate di-

gate to apply a positive signal to all gate diode cathodes. The first two storage flip-flops are sampled by a two-input NOR gate which resets the trigger flip-flop and closes, or grounds, the input gate diodes through the inhibit gate when either of the two storage flip-flops have been set. The output of this NOR gate is stretched by the series 5k resistor and .01 μ F capacitor to give all of the appropriate storage flip-flops a chance to set before closing the input gates.

If another key is pressed while the buffer storage is full, the trigger flip-flop is

again "set" and tries to open the input gates, but the inhibit gate prevents this until the storage flip-flops are cleared by the main logic board. The NOR gate that

fer. This provides the delay necessary to assure sufficient time to insert the stored signal in the keyer prior to clearing the buffer.

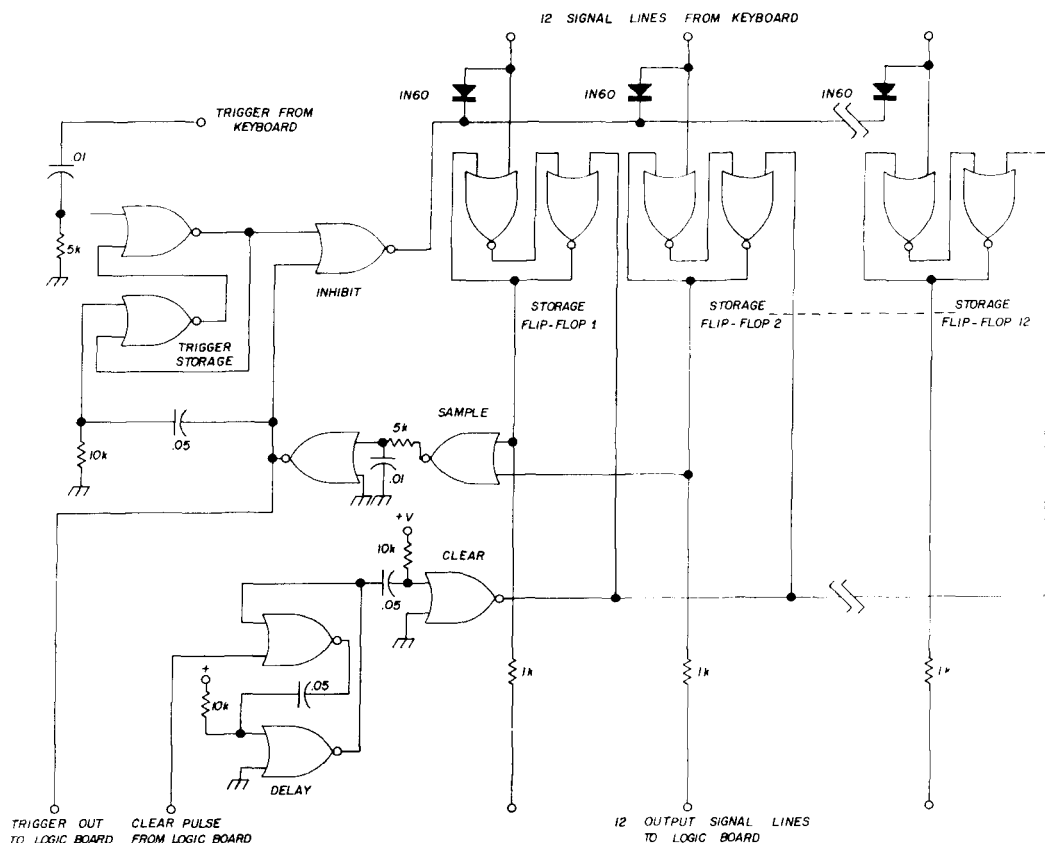


fig. 3. Logic diagram of the buffer storage unit. The author used eight Motorola MC724P quad two-input gates in his model.

detects the presence of a stored signal and inhibits the opening of the input gates also supplies a positive trigger to the basic keyer, just as if the signals in buffer storage were coming directly from the keyboard. With this logic, if a key is pressed while the storage is empty, the character will be inserted only once, and the key must be pressed again for a repeat.

When the main insert gates are opened, a positive pulse triggers a one-shot circuit. The negative-going trailing edge of the output pulse of this stage is inverted and used to reset all flip-flops in the buf-

adding more storage

Since the output of the buffer storage appears as a virtual keyboard, and is compatible to input and output signal logic polarity, buffer modules can be stacked to provide banks of four, five, six or more character storage. I have connected four of these together to store any five-letter word (such as my call) but the utility is questionable since the first character is generally complete by the time you are typing the third or fourth character, except at very low code speeds.

ham radio