

Information Systems: Hardware Version of an Optimal Convolutional Decoder

W. Lushbaugh

Communications Systems Research Section

A hardware version of an optimal convolutional decoder is described. This decoder implements the Viterbi Algorithm for maximum-likelihood decoding of short constraint length convolutional codes. It is capable of decoding at data rates up to a megabit for codes of constraint-length 3, 4, or 5 at code rates $\frac{1}{2}$ or $\frac{1}{3}$.

I. Introduction

The simulation results obtained by Heller (Ref. 1) for the decoding of short constraint length convolutional codes using the Viterbi decoding algorithm created considerable interest in the practical application of the algorithm. Layland (Ref. 2) described an all software decoder of this type capable of a 5 kbits/s data rate for a $K = 4$ rate $\frac{1}{3}$ code. This article describes a hardware version of a Viterbi decoder capable of a megabit data rate for constraint lengths 3, 4, or 5, at code rates $\frac{1}{2}$ or $\frac{1}{3}$.

II. Convolutional Codes and the Optimum Convolutional Decoding Algorithm

A convolutional encoder for a code of constraint length K and rate $1/V$ consists, typically, of a shift register of length $K - 1$, coupled with V parity-check adders. For each new bit to be encoded, the V linear combinations

(Mod 2) of the $K - 1$ bits in the shift register, and the new data bit, are computed and transmitted. The new data bit is then shifted into the coder register and the oldest bit therein shifted out. Clearly, the encoder can be viewed as a finite-state machine with 2^{K-1} states. For each of the 2^{K-1} states that the encoder can assume, there are only two predecessor states possible e.g., if $K = 4$ and the present state is 001, the two possible predecessors are 010 and 011. Thus each state has a possible *zero* or *one* predecessor, the *zero* or *one* being associated with the least significant bit of the state number.

The optimum convolutional decoder contains information pertaining to all possible states of the encoder i.e., there are 2^{K-1} sets of likelihoods or metrics, and 2^{K-1} most-likely bit streams (called survivors) leading to each of the states. At each step in the algorithm, a computation element associated with each state examines the two possible predecessors, adds the correlation between the branch

symbols from each predecessor and the new V received symbols, and chooses the higher likelihood as the new metric for that state. The survivor associated with the winner is also copied as next state survivor and the *zero* or *one* associated with the winner is added as the newest bit in this survivor stream. It has been shown experimentally (see Ref. 2), that oldest bits of the 2^{K-1} survivors converge to each other and to the maximum likelihood decoded answer.

III. Design Philosophy

Two main systems designs were originally considered for the Viterbi decoder. These were the serial computation of the 2^{K-1} states which would use parallel operations on each of the state metrics, and the finally adopted design of parallel computations of the states with a computing unit associated with each of states using serial arithmetic. This design was adopted because the design aim of completing one branch (bit) calculation in $1\ \mu\text{s}$ could easily be met, and because the hardware for one state calculation could be put on a printed circuit board and enable the design to be expanded to larger constraint lengths. A picture of the state board is shown in Fig. 1. Sixteen of these boards were used in the present design, and in general 2^{K-1} are needed for a constraint length K machine.

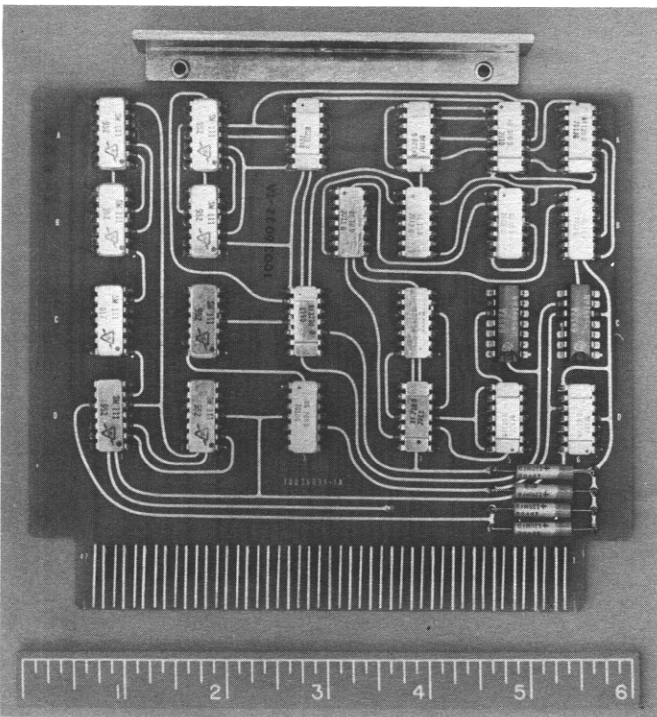


Fig. 1. Viterbi decoder state board

IV. Operation of the State Board

The logic diagram of the state board is shown in Fig. 2. The state board consists mainly of shift register memory, 10 bits for the metric of that state, 32 bits for the survivor stream associated with that state and some serial adders. To increase the speed of the system, two 10-bit registers are actually used to hold metrics, one for each of the two candidates for the new state metric. The metrics are shifted at a 10-MHz rate to be consistent with the $1\ \mu\text{s}$ per node. Since the two candidates are serially introduced to a state board, and the correlation with the received symbols is opposite for each, the received metric from a *zero* predecessor state has the branch metric (for that state) added to it while the metric from the *one* predecessor state has it subtracted. The larger of these two values then becomes the new metric for the state in question. Both of these values have to be saved, since the larger of the two is not known until the sign bits of each arrive at the end of the cycle. A comparison of the two possibilities is made serially as the branch metric is added and the result is stored into the x flipflop at the end of the cycle. This x decision then steers the correct metric out of the board on the next cycle and is used to copy the correct survivor stream into the state board in question. This x value also becomes the most recent member of the survivor stream.

The length of the survivor stream is 32 bits broken into two 16-bit portions, so that a 16-MHz shift rate is necessary to copy the survivors in $1\ \mu\text{s}$. The decoded bit is taken from the oldest survivor of the all *one's* state board. As is shown in Ref. 2, little degradation is suffered by choosing this answer rather than the survivor associated with the maximum likelihood state at each step.

A. Special Boards

In addition to the 2^{K-1} state boards in the system, four special-purpose boards were needed to complete the system. These were the timing board, which develops the various timing signals needed, the metric calculator, a metric select board, and a node sync board.

1. Metric calculator. For a rate $\frac{1}{2}$ code there are only eight possible received sets of 3 symbols for each transmitted bit. Moreover, each state unit of the decoder always expects to see the same set of received symbols. Thus the eight possible values of sums and differences of the three received symbols can be computed in one central location and fed to the appropriate state boards. The states that receive each branch metric are determined by the code, and to change codes, only these eight signals

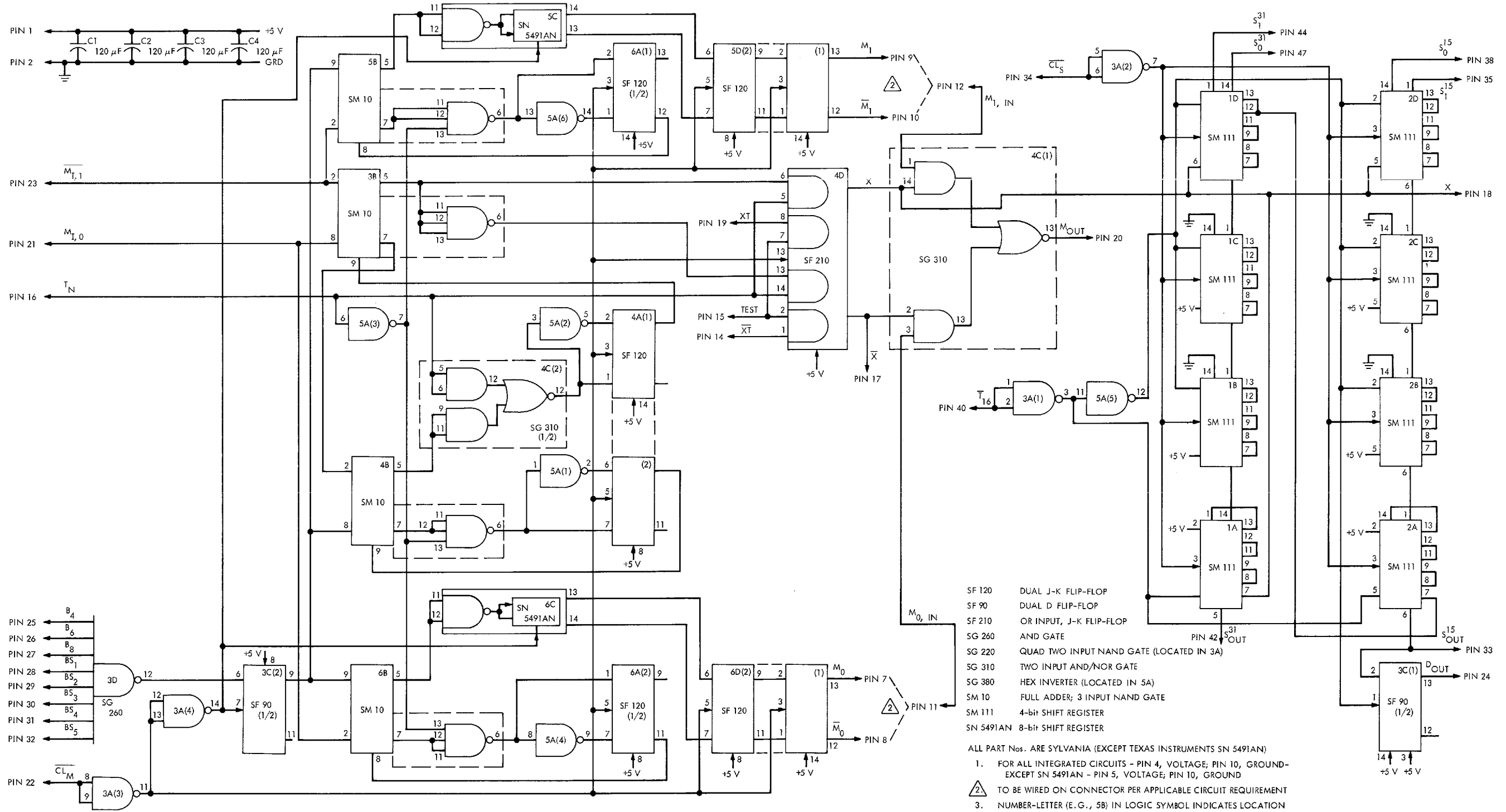


Fig. 2. Logic diagram Viterbi decoder main board

need to be rerouted. This selection and distribution of the branch metrics is performed by the code select board which consists of 16 four-way select gates, one for each state board in the system, allowing up to four codes to be hard-wired into the system and selectable with a four-position switch.

The metric calculator has been designed to accept symbols quantized up to four bits each. When these symbols are properly scaled and three are added together the range of branch metrics for a rate $\frac{1}{3}$ code is -45 to $+45$.

Changing the code rate to rate $\frac{1}{2}$ is accomplished by setting one of the symbols to zero on the metric calculator and using the four significant results that remain as the branch metrics.

To change the constraint length of the code, state boards are associated in groups of $16/2^{K'-1}$ where K' is one of the other possible values of constraint length i.e., either 4 or 3. Each of the groups of states now corresponds to a superstate with each state unit within it receiving the same branch metrics. It can be shown that all states within a superstate perform identically after a transient of $16/2^{K'-1}$ steps in the algorithm.

2. Node sync. Layland has shown in Ref. 3 that node synchronization of a Viterbi decoder can be obtained using the decoder state likelihoods integrated over several hundred code branches assuming only that the data source is sufficiently random. The design finally adopted, however, does not use integration over a fixed number of nodes but instead compares the uphill climb of the metrics to the sum of the absolute values of the received symbols

$$\sum_{j=1}^v |r_j|$$

Figures 3 and 4 show the experimental densities of the state likelihoods minus

$$\sum_{j=1}^v |r_j|$$

for a $K = 4$ and $K = 5$ code. It was observed from these curves that if $+4$ is added to both curves, then the in-sync version would be shifted positive while the out-of-sync curve would remain negative. Thus an in-sync indication is the positive drift of the quantity

$$M_k - M_0 - \sum_{n=1}^k \sum_{j=1}^v |r_{j,n}| + 4k$$

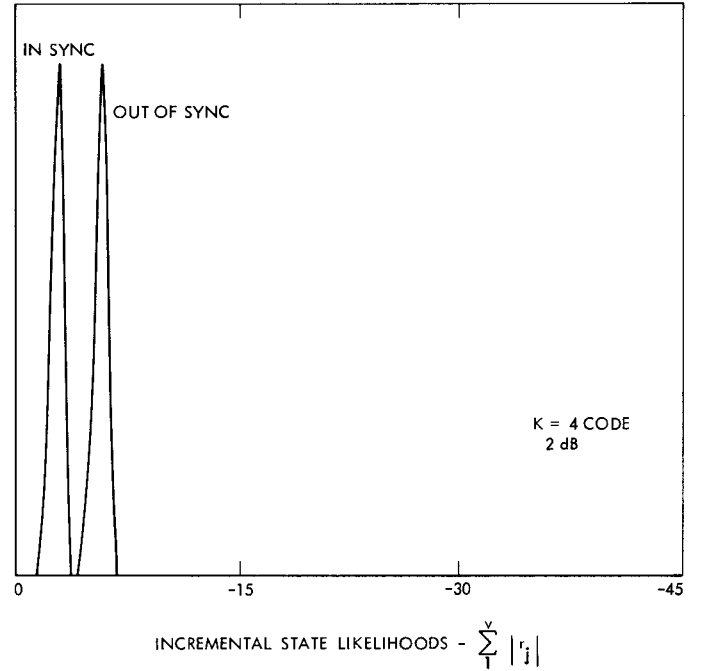


Fig. 3. Experimental densities of incremental state likelihoods minus $\sum_{j=1}^v |r_j|$ for a $k = 4$, rate $\frac{1}{3}$ code

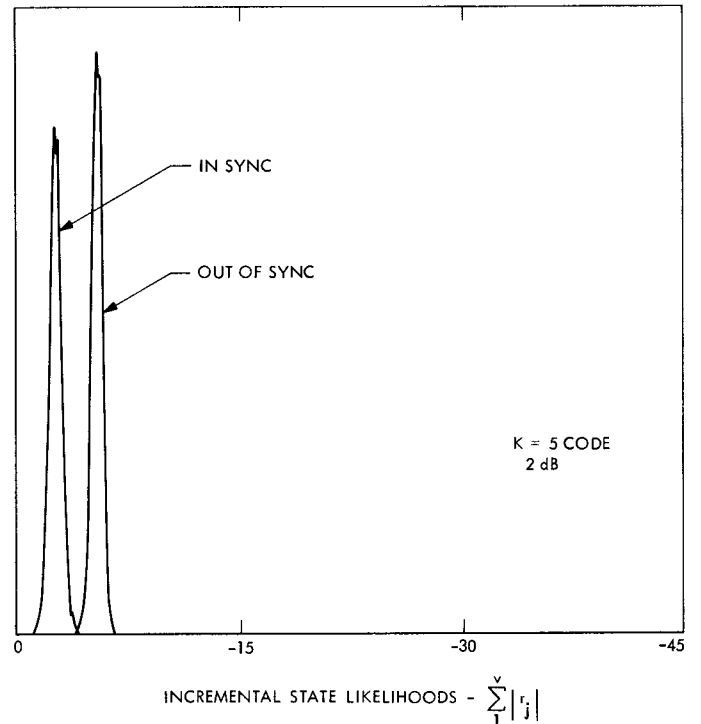


Fig. 4. Experimental densities of incremental state likelihoods minus $\sum_{j=1}^v |r_j|$ for a $k = 5$, rate $\frac{1}{2}$ code

where M_k is the k th metric and M_0 is the metric at some starting point. In the actual design, 10 bit numbers are used for this quantity which is initially loaded with some $M_0 + 256$ and the sign bit is monitored for positive or negative overflow. These overflows are in turn divided by three to increase the integration time to the order of magnitude suggested by Layland.

B. Code Conventions

For each $V - 3$ code there are 6 different permutations of sending the three symbols as well as a twofold symmetric way of interpreting the order of the taps on the encoder. For this reason a standard way of interpreting each code is desirable. The present decoder has been wired according to the conventions defined by the following algorithm:

- (1) Write the code in binary on V lines and write the weight of each term below, e.g., for a $K = 6$ and $V = 3$ code

$$\begin{array}{r} 1\ 0\ 1\ 1\ 1\ 1 \\ 1\ 0\ 1\ 1\ 0\ 1 \\ 1\ 1\ 0\ 0\ 1\ 1 \\ \text{Weight of each tap: } 3\ 1\ 2\ 2\ 2\ 3 \end{array}$$

- (2) Take the K digit number that results and its reverse e.g., in this case,

$$3\ 1\ 2\ 2\ 2\ 3 \quad \text{and} \quad 3\ 2\ 2\ 2\ 1\ 3$$

and choose the larger of the two as the standard order for writing the code. The newest bit of the encoder will now be associated with the most significant bit of this number.

- (3) Rewrite the code as in step 1 but in the standard order e.g.,

$$\begin{array}{r} 1\ 1\ 1\ 1\ 0\ 1 \\ 1\ 0\ 1\ 1\ 0\ 1 \\ 1\ 1\ 0\ 0\ 1\ 1 \end{array}$$

- (4) Now interpret each row as a binary number, most significant bit on the left, and reorder them by mag-

nitude. The smallest of these will be S_1 , the next S_2 , etc., e.g.,

$$S_1 = 1\ 0\ 1\ 1\ 0\ 1$$

$$S_2 = 1\ 1\ 0\ 0\ 1\ 1$$

$$S_3 = 1\ 1\ 1\ 1\ 0\ 1$$

This is the uniquely defined standard form for the code and the encoder is to be wired to transmit these 3 symbols in the order S_1, S_2, S_3 . Table 1 shows the standard form of the three codes presently wired into the machine.

Table 1. Standard form of present three codes

Code	S_1	S_2	S_3
$K = 3, V = 2$	101	111	—
$K = 4, V = 3$	1011	1101	1111
$K = 5, V = 3$	10101	11011	11111

V. Conclusions

Figure 5 shows a picture of the completed system. The 16-state board and four special boards, which occupy two slots each, just fills a standard chassis with 25 slots. A single 5-V power supply is used and the system draws 12 A. System debugging has been completed and simulation has shown that the system operates in agreement with error rate predictions from previous software simulation.

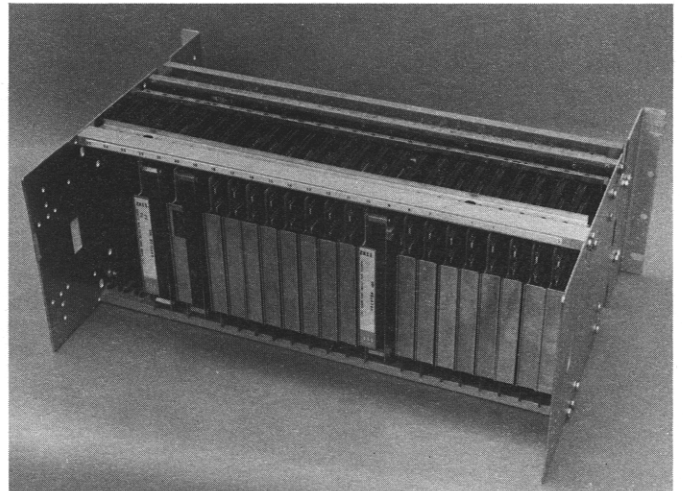


Fig. 5. Completed decoder

References

1. Heller, J. A., "Sequential Decoding: Short Constraint Length Convolutional Codes," in *Supporting Research and Development*, Space Programs Summary 37-54, Vol. III, pp. 171-177. Jet Propulsion Laboratory, Pasadena, Calif., Dec. 31, 1968.
2. Layland, J. W., "Information Systems: Capabilities of an All-Software Optimum Convolutional Decoder," in *The Deep Space Network*, Space Programs Summaries 37-62, Vol. II, pp. 64-66. Jet Propulsion Laboratory, Pasadena, Calif., March 31, 1970.
3. Layland, J. W., "Information Systems: Synchronizability of Convolutional Codes," in *The Deep Space Network*, Space Programs Summary 37-64, Vol. II, pp. 44-50. Jet Propulsion Laboratory, Pasadena, Calif., August 31, 1970.