Testing Interconnected VLSI Circuits in the Big Viterbi Decoder

I. M. Onyszchuk
Communications Systems Research Section

The Big Viterbi Decoder (BVD) is a powerful new error-correcting hardware device for the Deep Space Network (DSN), in support of the Galileo and CRAF/ Cassini missions. Recently, a prototype was completed and run successfully at 400,000 or more decoded bits per second. This prototype is a complex digital system whose core arithmetic unit consists of 256 identical very large scale integration (VLSI) gate-array chips, 16 on each of 16 identical boards which are connected through a 28-layer, printed-circuit backplane using 4416 wires. Special techniques were developed for debugging, testing, and locating faults inside individual chips, on boards, and within the entire decoder. The methods are based upon hierarchical structure in the decoder, and require that chips or boards be wired themselves as Viterbi decoders. The basic procedure consists of sending a small set of known, very noisy channel symbols though a decoder, and matching observables against values computed by a software simulation. Also, tests were devised for finding open and short-circuited wires which connect VLSI chips on the boards and through the backplane.

A new "single-board" BVD is being constructed now for implementation in DSN stations. The design is based upon a new VLSI chip which contains 65,536 bits of static RAM and circuitry now located on boards and in 4 gate arrays. Sixty-four of these new chips will be mounted on a single board and interconnected with 5120 wires. Experience gained from the prototype actually led to special circuitry designed inside the chips for simpler and better checking of these wires. Furthermore, each chip also operates independently as a complete, constraint length 9, rate 1/6—(9,1/6) Viterbi decoder, for testing within the single-board BVD. Like the prototype BVD, the new decoder can also be programmed to decode convolutional codes having constraint length 2 to 15 and rate 1/2, 1/3, 1/4, 1/5, or 1/6.

I. Introduction

A Viterbi decoder is a probabilistic device which may be ultimately verified by measuring bit error rates over a range of channel noise levels [3]. However, such a test requires considerable work for a simple pass/fail result, and no information about the source of error is obtained when the decoder fails. Therefore, other techniques were required to test the Big Viterbi Decoder (BVD), and to pinpoint sources of error when the decoder stops working. It was discovered that running the decoder with a small number of known, very noisy channel symbols was extremely effective for testing and isolating faults within the BVD. Furthermore, the modular construction of the BVD enabled independent checking of individual chips and boards as smaller Viterbi decoders prior to their installation in the BVD.

The problems addressed in this article are testing, fault location inside, and debugging of the BVD. The problems also include checking of "metric connections"—not control signals—between very large scale integration (VLSI) chips. A connection is defined as a chip output driver, the "metric" wire it drives to another chip, and the corresponding input driver on the other chip. There are 4416 such connections in the prototype BVD. Unique software programs were required to verify connections on boards and in backplanes. These procedures were used to debug the BVD prototype and are now part of the system software.

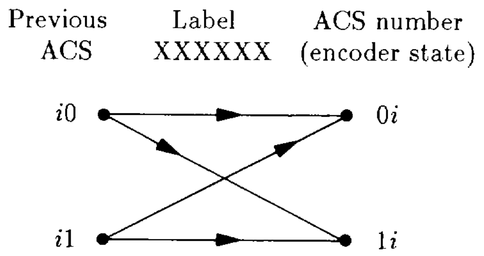
The basic idea is to connect one or more chips in order to implement the arithmetic core of a parallel Viterbi decoder. The hierarchical decomposition of the (15,1/6) BVD into identical sub-blocks [2] makes this approach possible. For example, one gate array may be wired as the arithmetic part of a (7,1/6) decoder. Similarly, each of the 16 boards in the prototype BVD, which contain 16 gate arrays, may be tested independently as the core of a (11,1/6) decoder. Thus, chips and boards may themselves be individually and effectively tested as smaller Viterbi decoders before they are integrated into the BVD.

This article is organized as follows. Basic arithmetic processors called "butterflies," and their interconnection in the BVD, are described in Section II. Procedures for testing the VLSI chips are explained in Section III. Techniques are described in Section IV for locating short-circuited and open-circuited metric wires on boards and in the backplane. The single-board BVD is described in Section V. Based upon experience with the 16-board prototype, a plan is outlined for independently testing new VLSI chips and connections in the single-board BVD.

II. Butterflies

The fundamental arithmetic unit in the BVD, a "butterfly," is described in this section from a purely operational viewpoint. The following explanation should be sufficient for understanding the remainder of this article. The basic theory underlying the use of butterflies in Viterbi decoders appears in previous articles [1,2].

The "butterfly" shown below is the basic arithmetic unit in the BVD [1,2]:



$B = 2^{K-2}$ butterflies may be connected to form the arithmetic core of a parallel, rate $1/n$, constraint-length K, Viterbi decoder [2]. For example, 32 butterflies are wired in Fig. 3 to form the arithmetic part of a $K = 7$ decoder. Each butterfly in the decoder is identified by a number, i, in the range 0 to B–1 inclusive. Butterfly i has two processors, called Add-Compare-Select (ACS) units for reasons that will become evident when the ACS function is described later. The ACS units are marked $0i$ and $1i$ for the encoder states to which they correspond [1]. An ACS unit in the decoder and the corresponding encoder state will be referred to interchangeably. After the hardware comprising a butterfly is reset, but before the butterfly starts operating, a 6-bit "label" is loaded into a register inside the butterfly. This label is computed using the butterfly's number (i) and 6 encoder generator polynomials [1]. If the code has less than 6 generator polynomials, because the code rate is not 1/6, then zero polynomials are added.

A decoder cycle will denote the basic unit of processing time for a Viterbi decoder. During each decoder cycle, butterfly i receives two 16-bit numbers, called state metrics, bit-serially (least significant bit first): one from the output of ACS $i0$ in butterfly $2i$ (mod B) and the other from ACS $i1$ in butterfly $2i + 1$ (mod B). The state metrics, stored inside ACS units, are all set to a common value such as 0 when the butterflies are reset. During each decoder cycle, all butterflies also receive a common set of six 8-bit quantized, demodulated channel symbols. These are used with the label inside each butterfly to compute a number (branch metric) for each of the 4 branches. Now for each of the two incoming branches, each ACS unit adds the branch metrics to the corresponding state metric. Each ACS unit keeps only the smaller of the two resulting numbers, for transmission as a state metric input to a butterfly during the next decoder cycle, and writes into a random-access memory (RAM) the least significant bit of the ACS/state whose metric led to the smallest number. The bits written to RAM are called traceback bits because during later decoder cycles, a process traces backwards through the RAM to extract the decoded bits to be output by the Viterbi decoder. During every decoder cycle, each one of the $2B$ ACS units writes a 0 or 1 into a particular location in traceback memory (Fig. 1). The location is indexed by two numbers: a state number which is the ACS unit number, and a column number which is the same for all ACS units and depends only on the decoder cycle. After each decoder cycle, one column of traceback RAM is written. Starting from reset, columns 0 to 509 are written sequentially, but, thereafter, a column is overwritten immediately after a traceback process has used it.

In this article, "testing" a chip, board, or decoder means verifying that no manufacturing or assembly defects exist. "Fault isolation" means precisely locating sources of error when the hardware stops operating correctly. "Debugging" means finding and fixing errors in logic design and timing. Techniques based upon software simulations using known channel symbols, corresponding to an information bit signal-to-noise ratio E_b/N_0 of 0 dB, were instrumental during all phases of BVD testing, debugging, and fault isolation.

For purposes of testing a butterfly, the controllables are a 6-bit label which is loaded once after reset, and then groups of six 8-bit symbols. The only observables are traceback bits written from the butterflies into RAM. However, during the debugging of chips and boards, a logic analyzer was required to observe state metrics.

III. Testing VLSI Chips

Before encapsulation inside packages with pins, VLSI chips are tested by applying a sequence of binary patterns called "test vectors" to input pads during successive clock cycles. The test vectors are designed to exercise all circuitry inside the chip and expose any faults created during fabrication. The resulting vectors of bits at chip output pads are compared with expected, correct vectors obtained from a software simulation, like one used to verify logic and timing within the chip. Two different VLSI chips were designed for the BVD: a semi-custom, military-specified chip containing 16 identical butterflies, circuitry for processing control signals, and circuits for organizing traceback bits before they are written to memory; and a commercial gate array which implements two semi-custom chips, for a total of 32 butterflies.

The first VLSI testing for the BVD involved the semi-custom chips. Every flip-flop inside these chips, except those in the middle of serial registers, has a two-input multiplexor at the data input. All of these multiplexors are controlled by a common TEST signal. When TEST is active, each multiplexor selects the output of another flip-flop, instead of a combinational circuit output as it would during normal operating mode. Thus, all flip-flops inside the chip may be chained together in order to load or read bit patterns. In order to test a chip, the procedure described below is repeated once for each pattern in a very carefully designed set.

A pattern is loaded bit-serially into flip-flops while the TEST signal is active. The chip is then clocked for one or more cycles while TEST is inactive. Next, the contents of all flip-flops are read out as a serial bit stream while TEST is active, and compared with a known correct pattern in order to detect faults. The total number of clock cycles needed for each bit pattern is a few more than twice the number of flip-flops in the chip. However, the number of test vectors required to implement all bit patterns in the designed set can be very large, particularly if there is only one serial chain path which connects all flip-flops in the chip, instead of many parallel chains—for example, one for each butterfly. There is only one serial chain path in the semi-custom chips, and it joins all 1776 flip-flops in the chip. Furthermore, the chip requires two clock cycles per test vector. Hence, at least 7106 test vectors are required for each bit pattern. However, integrated circuit test equipment for both the semi-custom and gate array BVD chips could process at most 16,384 different test vectors. Furthermore, any testing of chips individually will not expose potential problems when several identical chips are wired together. When 512 semi-custom chips are installed in the BVD, fault isolation using bit patterns which are serially loaded and unloaded may take a long time and require complex software for analyzing output bits.

The semi-custom chips were tested by the manufacturer using a set of test vectors created by a software simulation of a single butterfly. The 4 metric inputs and outputs for each butterfly inside these chips are wired directly to pins. All 16 butterflies were tested in parallel, using the same label, noisy symbols, and input metrics. However, it was not certain, for reasons explained below, that the chips could be connected as the ACS part of a Viterbi decoder. Hence, the 32 butterflies in two semi-custom chips were interconnected as shown in Fig. 3.

A software simulation was run for the two interconnected chips. The program evolved from one used to generate test vectors for the gate arrays. Pseudo-random data were sent through a (7,1/6) convolutional encoder and the 0/1 bits output were mapped to ±21, which represent quantized, demodulated channel symbols when there is no channel noise. Then 8-bit quantized, zero-mean, white Gaussian noise was added to these symbols. The noise variance was set for a bit signal-to-noise ratio of 0 dB, an extremely difficult operating point for a (7,1/6) Viterbi decoder. Groups of 6 noisy symbols were processed by the software simulator, and test vectors were generated. The pair of semi-custom chips passed the test vectors (herein called the zero-dB check) only after the test vectors were modified as described below. Due to timing of signals at chip pins, every state metric became hex 4000 instead of 0000 after reset. This anomaly does not affect traceback bits, so it merely resulted in a simple change to the software simulation. A few other problems found were compensated for outside the chips. These small discrepancies could not be exposed by the original set of test vectors, but understanding the anomalies turned out to be crucial during later BVD testing.

The zero-dB check demonstrated that two semi-custom chips could work as the ACS part of a (7,1/6) Viterbi decoder. The results in [2] showed that 2^{K-6} properly interconnected chips would form a (K,1/6) decoder, for $K \geq 6$. These facts inspired confidence that 32 semi-custom chips on a board could be tested as the ACS part of an (11,1/6) decoder, and that 16 boards connected through a backplane would implement the ACS section of the (15,1/6) BVD.

A batch of gate-array chips was tested at 25 MHz by the manufacturer using test vectors generated by a simulation like that used for the zero-dB check, except that extremely noisy symbols (corresponding to E_b/N_0 near -8 dB) were used to better exercise butterfly logic. Twenty chips that passed tests were packaged as engineering prototypes. The 32 butterflies inside each of these gate arrays were then connected using a test fixture with wires connecting chip pins as the ACS part of a (7,1/6) Viterbi decoder. All 20 gate arrays passed the zero-dB check. As a result, 600 chips were ordered to build two prototype BVDs, each one with 256 gate arrays, 16 gate arrays on each of two spare boards, and 12 spare chips. The zero-dB check was also used to test individual boards, each with 16 gate arrays, when the boards were wired using a special edge connector as the ACS part of an (11,1/6) Viterbi decoder.

IV. Locating Bad Metric Wires (Connections)

During BVD debugging, procedures were needed to verify that VLSI chips already installed in the system were still working, and that all connections between chips were correct. Thus, a "traceback read" procedure was implemented, based upon the zero-dB check, to match a group of traceback bits written to the entire traceback memory, against values computed during a software simulation. Also, new procedures were devised for locating broken or missing metric wires (open circuits) and short-circuited metric wires in the BVD. These tests (1) examine traceback bits computed inside butterflies and stored in memory, (2) were required to debug backplanes, and thus (3) are now part of the BVD software. All of the above tests start by resetting the decoder and then loading a particular set of butterfly labels into the chips.

In the traceback read procedure, 510 sets of 6 symbols are processed so that every traceback memory location is written to and later read from. However, only the first wrong traceback bit can be reported, even though state metrics could be incorrect during decoder cycles prior to the one for which the first wrong bit is computed. Also, reading traceback memory cannot always be used to determine which of the two metrics input to the butterfly, for which a traceback bit error is reported, is incorrect. This problem could be reduced by repeating the procedure using many different sets of symbols, which also helps to detect intermittent faults. A traceback bit error could be caused by one of the chips driving metric wires, by connections between chips, or by the chip receiving the two metrics (Fig. 2). Without a tester for chips, it is impossible to isolate the fault, so the receiving and then driving chips are replaced in that order. If this does not solve the problem, the connection between chips is suspected.

The procedures described next require that the VLSI chips, memory chips, and interface circuitry on every board operate perfectly. This is verified in the prototype (16-board) BVD by testing each board independently as an (11,1/6) decoder.

A. Detecting Open-Circuited/Grounded Metric Wires

In the prototype BVD, or any subset of it wired as a $(K,1/6)$ Viterbi decoder, if a metric wire is open-circuited or grounded, the metric value will remain zero if the wire floats low, which is usually the case because there is nothing to drive the wire. This zero metric will propagate, with branch metrics added, to all state metrics after $K-1$ sets of 6 symbols are processed. The zero metric will continue to corrupt every other state metric.

Detecting open-circuit metric wires is straightforward: the label 010101 is loaded into each butterfly, the decoder is reset, and six particular symbols are processed to force every ACS unit to select the "horizontal" branch input. Then for each state, a "traceback" bit is read from column 0 of the memory and an error reported when the bit differs from the most significant bit of the state. In the case of an error, a diagonal branch was chosen so the state having a wrong traceback bit is reported, as well as the butterfly input 0 or 1 which caused the error. A similar procedure detects metric wires connected to power.[1]

B. Detecting Short-Circuited Metric Wires

When two or more metric wires are connected together, the resulting signal has an intermediate voltage (between 0 and 5 volts) during arithmetic clock cycles in which the metric values output by VLSI chips drive the line differently (recall that metrics are processed bit-serially). The resulting metric value may be lower, equal to, or higher than the correct value. However, when the wire is being driven high and low by two different metric output drivers, a logic low is usually interpreted by VLSI chip inputs. The following example demonstrates a possible resulting metric C when two state metrics, labelled A and B, are shorted together:

$$A: \quad 0\,1\,0\,0\,0\,0\,0\,1\,1\,1\,0\,0\,0\,1\,0\,1$$

$$B: \quad 0\,1\,0\,0\,0\,0\,1\,0\,0\,1\,1\,0\,0\,0\,1\,1$$

$$C: \quad 0\,1\,0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,0\,1\,0\,1$$

While an open-circuit metric may be detected during the first set of traceback reads, which occurs from column 0 of memory, shorts are detected only after the second set of symbols has been processed because all metrics are the same constant after reset, so all column 0 traceback bits are correct if no wires are open and all circuitry works properly. Hence, no errors must be reported during an open-circuit metric test prior to running the short-circuit test described below.

The algorithm described below locates with very high probability all metric wires having multi-level signals, but it cannot determine which wires are shorted together. Consider the set of 64 groups of six symbols, formed by taking a given group of 6 noisy symbols and inserting all 64 possible 6-bit vectors in place of the 6 sign bits. For each group of 6 symbols in this set, the decoder is reset, 6 fixed noisy symbols are processed, and then the group is sent to the decoder. Every bit in column 1 of traceback memory is read to determine which metrics are corrupted by shorts (recall that column 0 is correct because the decoder passed the open-circuit test). When an incorrect traceback bit is read from memory, the butterfly input corresponding to this bit is assumed shorted because a corrupted metric usually has a lower value, so the decoder selects it instead of the correct path. After all 64 iterations are complete, one for each group of 6 symbols in the set of 64, the number of errors found at each of the two metric inputs is reported for every decoder state.

For all runs of this algorithm on the BVD hardware, a multi-level signal has always been found on the metric input for which the largest number of errors was reported at a given state. In fact, one of the two metric inputs to a state having wrong traceback bits nearly always had 0 errors reported. The 64 groups of symbols have the effect of creating all 64 possible sets of 4 branch metrics inside a butterfly for 6 given input symbols. These different branch metrics help expose erroneous state metrics—sent along wires between chips—by propagating errors to the traceback bits, which are the only observables. Of course there is no guarantee that the algorithm will expose every short, but it has always worked in practice. In fact, the algorithm located touching pins on the backplane and in extender cards, caused by misalignment when boards were installed. In order to increase the probability that a short-circuited metric wire is detected, the above algorithm could be repeated with one or more additional, completely different symbol sets and the errors accumulated before final reporting.

V. Testing the Single-Board BVD

For implementation in DSN stations, the 16 boards with 256 identical 32-butterfly gate arrays in the prototype BVD are being compacted into 64 identical new VLSI chips to be mounted on a single backplane (Fig. 4). Each new chip contains 128 butterflies, 65,536 bits of static RAM for traceback, and circuitry for traceback, decoding, and testing. These will be interconnected with 5120 wires on a single board. Key system design objectives include automatic checking of all metric wires between chips and testing chips individually, perhaps using on-chip self-test circuitry not yet designed, while the chips are installed in the decoder. Due to the new VLSI chips, the arithmetic core of the single-board BVD will be more reliable and about 4 times less expensive than the arithmetic core in the current BVD prototype. Furthermore, the maximum rate of the single-board BVD will be at least 1.5 million bits/sec, at a 20-MHz system clock.

[1] I. M. Onyszchuk, "Finding Open and Short-Circuited Metric Wires in the BVD," Interoffice Memorandum 331-91.2-014 (internal document), Jet Propulsion Laboratory, Pasadena, California, March 18, 1991.

Each of the new VLSI chips can operate as a complete (9,1/6) Viterbi decoder. Test circuitry has been designed to internally connect all 128 butterflies to form the ACS section. Every cell in the traceback memory can be checked individually. Thus, even while installed in the BVD, each chip may be tested separately. Of course, on-chip self-test circuitry, not currently part of the new chips, would also enable such testing; each chip could be commanded using a special signal to test its internal circuitry and memory, and then provide a pass/fail signal. Chip self-testing has the advantage of verifying correct operation using only 2 signals. Testing a chip using external symbols and signals requires perfectly working interface circuitry. However, none of the input or output drivers for metrics would be tested by either of these techniques. The method suggested below will expose a bad driver by using multiplexors inside ACS units in order to check metric wires between chips.

Reading traceback memory is a difficult and time-consuming operation in the current prototype BVD. Although the tests have been successful and instrumental in fault isolation, they are not foolproof and require manual interpretation. Therefore, a better method is required to test the decoder and to locate faults. Also, 160 of the 196 signal pins on the new VLSI chips are used for metric wires. Thus, connecting a 16-bit parallel address bus to each chip might eliminate pins needed for chip testing either by the manufacturer or by the BVD software. In addition, there may not be enough space on the single-board backplane for address drivers and wires to all chip memories.

The suggestion outlined below has been made in order to simply but thoroughly test every connection in the single-board BVD. Recall that a connection in the decoder consists of a VLSI chip metric output driver, the wire it drives to another chip, and the corresponding input driver on the other chip. There are 5120 such connections in the single-board BVD. By placing a two-input multiplexor at the input to each one of the four 12-bit state-metric registers in each butterfly, the registers may be chained together during a test mode. The four multiplexors together require only about the same number of transistors as one of the 10 full-adders or one of the 60 flip-flops in a butterfly. The test mode could be used to check metric wires between chips after all chips have been successfully tested. Using the above multiplexors, a bit pattern may be loaded serially into state metric registers. If the decoder then processes 6 all-zero symbols, the pattern will be sent along metric wires between chips and into butterflies. The contents of all state-metric registers may then be read out bit-serially and used to identify bad metric wires directly. Traceback read operations are avoided, thereby reducing circuitry on the board, software, chip pins, and wiring of memory addresses to all VLSI chips.

Naturally, a progressive testing method for the decoder should be used in which successive tests depend upon the decoder's passing all previous tests. The datapath for symbols should be checked first. Then simple checks could be run on one chip to verify that control signals are working. Next, each chip, including the internal traceback memory, would be tested individually. This might be accomplished by using on-chip self-test circuitry. Each chip could be verified as a complete (9,1/6) Viterbi decoder. Then metric wires between chips could be checked using the procedure described above. Finally, the entire single-board decoder would be tested by comparing measured bit error rates for various noise levels against precomputed values from software simulations [3].

References

[1] J. Statman, G. Zimmerman, F. Pollara, and O. Collins, "A Long Constraint Length VLSI Viterbi Decoder for the DSN," *TDA Progress Report 42-95*, vol. July–September 1988, Jet Propulsion Laboratory, Pasadena, California, pp. 134–142, November 15, 1988.

[2] O. Collins, F. Pollara, S. Dolinar, and J. Statman, "Wiring Viterbi Decoders (Splitting deBruijn Graphs)," *TDA Progress Report 42-96*, vol. October–December 1988, Jet Propulsion Laboratory, Pasadena, California, pp. 93–103, February 15, 1989.

[3] I. M. Onyszchuk, "Coding Gains and Error Rates from the Big Viterbi Decoder," *TDA Progress Report 42-106*, vol. April–June 1991, Jet Propulsion Laboratory, Pasadena, California, pp. 170–174, August 15, 1991.

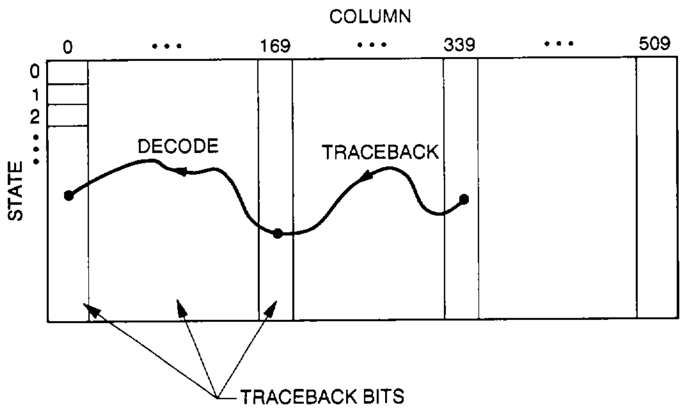


Fig. 1. Viterbi decoder traceback memory.

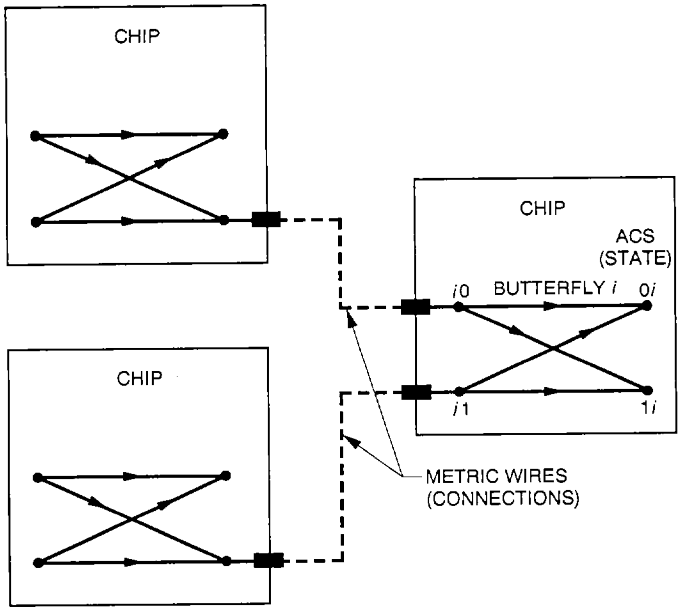


Fig. 2. Uncertainty of error location.

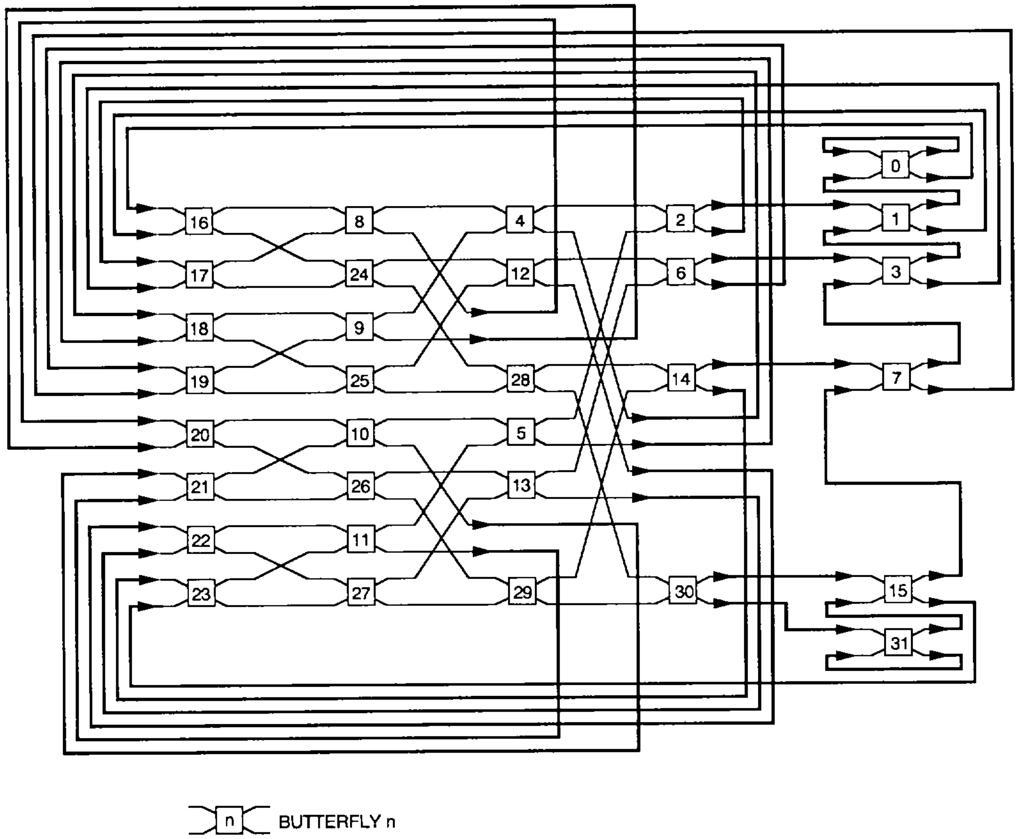


Fig. 3. Thirty-two butterflies connected to form the Add-Compare-Select (ACS) section of a constraint-length 7 Viterbi decoder.

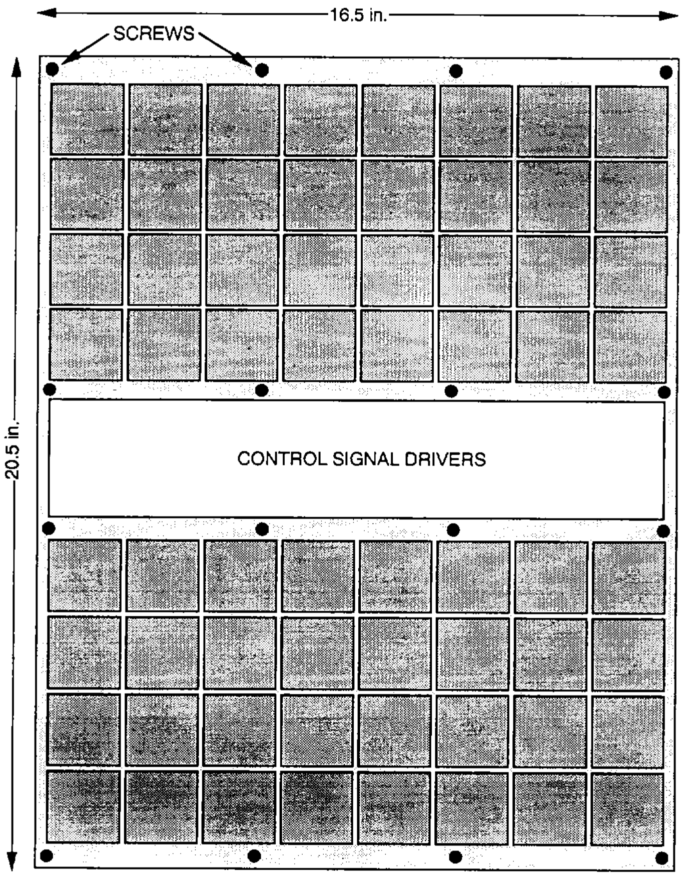


Fig. 4. A possible layout for the single-board Big Viterbi Decoder.