An Industry Standard Benchmark Consortium

TeleBench™ Version 1.1

Benchmark Name: Viterbi Decoder

Highlights

- **Benchmarks ability to process a forward error corrected stream**
- **Algorithm handles IS-136 channel**
- **Input is packet of 344 6-bit values**
- **Implements add-compare-select**
- **Includes four distinct data sets**

Application

The Viterbi Decoder benchmark exploits redundancy in a received data stream to be able to recover the originally transmitted data. **The benchmark provides an indication of the potential performance of a microprocessor to be able to process a forward error corrected (FEC) stream using the Viterbi algorithm for decode.**

A communication channel that is corrupted by noise typically uses FEC to maintain transmission quality and efficiency. One such FEC mechanism is the use of Convolutional encoding (see the Convolutional Encoding EEMBC benchmark datasheet) at the transmitter and the use of Viterbi decoding at the receiver. The Viterbi decode process is an "asymptotically optimum" approach to the decoding of Convolutional codes in a memory-less noise environment. This benchmark implements a Viterbi decoder that would be used to handle an embedded IS-136 channel.

Benchmark Description

The benchmark implements a soft decision Viterbi decoder. The input is a packet of 344 6-bit values each of which represents a pair of encoded bits (i.e. the input bit stream was produced by a ½ rate Convolutional encoder which generates a pair of output bits for each input bit). The 3-bit value of each bit represents a soft decision value in the range 0 to 7. The value 0 indicates a strong indication that a "1" has been received whilst a value 7 indicates a strong indication that a "0" has been received. The generator polynomials used for the Convolutional encode process are:

$$\mathbf{1 + x + x^3 + x^5}$$
$$\mathbf{1 + x^2 + x^3 + x^4 + x^5}$$

The Viterbi decoding algorithm is best viewed from the perspective of the trellis, for which the reader is referred to the relevant literature. The trellis describes the state diagram of the convolutional encoder as it evolves through time.

The decode process consists of a number of processes which are described below:

Compute Branch Metrics

This process progresses forwards through the trellis and attempts to calculate at each stage the distance between the received code word and all of the possible channel code words that could have been received.

Add Compare And Select (ACS)
Takes the branch metrics and computes the partial path metrics at each node in the trellis. The surviving path at each node is identified and the state history table updated accordingly.

Select Minimum Path Metric
Once the computation of branch metrics and ACS is the complete the state with the minimum path metric from the last stage of state history table is selected. This is the starting point for the trace back.

Trace Back And Recover Data
Using the starting point at the end of the state history table with the minimum path metric iterate back through the state history table, compute and then store the bit that would cause each state transition.

Analysis of Computing Resources

Viterbi decode is a computationally expensive process. The benchmark explores the target CPU’s ability to perform loops, bit-wise operations, table-lookups, comparisons and basic arithmetic operations.