Designing Telecommunication Applications Using Digital Signal Processing Functions with FPGAs

Field programmable gate arrays (FPGA) can speed time to market for your designs of telecommunication applications because of their quick turnaround time. Actel's core HDL program offers third-party developed, high-level, language-based functions—intellectual property (IP) functions. Based on your requirements, you can customize and reuse the VHDL or Verilog HDL cores.

Further, when using high-level, language-based functions developed by third-party companies, you can complete the application much sooner since, you don't have to do the design yourself. This paper illustrates the use of a digital signal processing (DSP) core targeted to Actel's A32200DX field programmable gate array. Performance and utilization results are reported, along with the design flow. Techniques discussed in this paper can be applied to other similar applications.
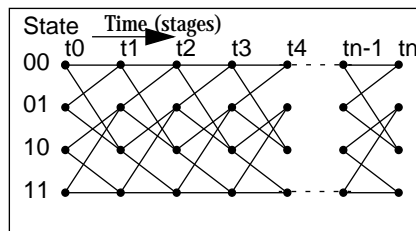
The Viterbi Algorithm

The Viterbi algorithm, one of the common digital signal processing functions, is used in the design of a decoder for convolutional codes. The Viterbi decoder performs maximum-likelihood decoding to achieve optimum performance. However, the Viterbi decoder requires extensive hardware for computation and storage.

The Viterbi algorithm is used not only to decode convolutional codes but also to produce the maximum-likelihood estimate of the transmitted sequence through a channel with intersymbol interference (ISI).

Among decoding techniques, the Viterbi maximum-likelihood algorithm is one of the best methods for decoding convolutional codes. The Viterbi decoder is especially good for short constraint length, and it makes decoding feasible at relatively high data rates. The Viterbi algorithm can be graphically represented by a state diagram called the *trellis diagram*, drawn as a function of time. A simple example of the trellis diagram is shown in Figure 1.

As shown in Figure 1, each node corresponds to a distinct state at a given time, and each branch represents a transition to other states at the next stage.

The Viterbi decoder receives a sequence of bits and attempts to find a path in the trellis diagram with an output digit sequence that mostly agrees with the received sequence.


Figure 1 • *The Trellis State Diagram*

The *branch metric* is a value associated with each branch shown in the trellis state diagram. If the branch metrics are viewed by weight, the goal of the Viterbi algorithm is to find the smallest weight path from the left-most column to the right-most column of the trellis diagram, and only a few path metrics (the smaller path metrics) are retained at each node. The Viterbi algorithm calculates a decision value and an input label for each node. The decision value is the smallest path from the previous stage. Finally, the output of the Viterbi algorithm is the smallest path metric in the last stage.

The Viterbi Decoder Architecture

The Viterbi decoder is divided into three functional parts. The first part is an add-compare-select (ACS) unit that is used to calculate the path metrics. The second one is the survivor memory control unit for survivor memory management. The survivor memory, used to store the survivor sequences, is the last part of the Viterbi decoder. Usually, the survivor memory can be implemented using the register exchange technique or the trace-back technique. Technical Data Freeway's (TDF) VHDL model of the Viterbi decoder is based on the trace-back technique and is described in more detail in the data sheet. The Viterbi decoder block diagram is shown in Figure 2.
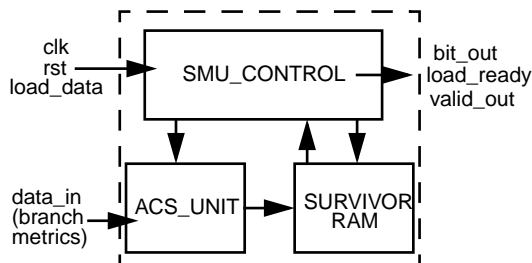
Figure 2 • *Viterbi Decoder Functional Block*

Parameters and Properties for the Viterbi Decoder

The Viterbi decoder VHDL model is fully parametrized. Depending on the design requirements, various parameters can be set to determine the general properties and bit accuracies of the Viterbi decoder. They are passed into the actual instantiation in the VHDL code.

Number of states in the trellis:StatesC

Number of bits to represent transition values:MetricBitsC

Add-Compare-Select (ACS) cells:AcsUnitsC

The length of the trace-back:TbLengthC

The length of the received burst:BurstLengthC

The initial path metric for state 0:InitValueC

The survivor memory (RAM) word length:MemWlC

The clock periods per input:acs_period

The size of each of the memory banks:mem_size

The burst length:burst_length

For more information about these parameters and properties, refer to TDF's Macrofunction data sheet.

The ACS UNIT

The functional diagram of the ACS cell is shown in Figure 3. The function of the ACS cells is to add the branch metrics to the corresponding path metrics to select the smaller path. The ACS cells also generate the pointer to the previous state (the decision value) and place it into the survivor memory.
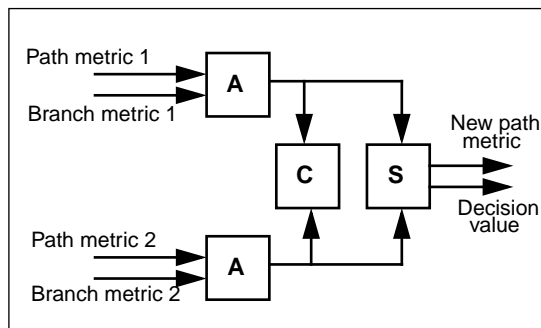

Figure 3 •   *Add-Compare-Select Cell*

The ACS unit contains more than one ACS cell. The detailed block diagram of the ACS unit is shown in Figure 4. The number of ACS cells is determined by the parameters of ACS_UNITs and path metrics, which are stored by the registers and temporary registers while processing. The number of the input branch metrics is also determined by the ACS unit.

The computation of the path metrics and the decision values are controlled by the signal clock counter. This signal clock counter also drives the SMU control unit. There are two parts to the calculations. They calculate the path metrics of the states in the upper part of the trellis diagram, parallel with the path metrics in the lower part of the trellis diagram. The incoming clock counter counts from 0 to ACS_period-1. When the clock counter is greater than zero, the computed path metrics are stored in the temporary registers. When the clock counter is equal to zero, the last computed path metrics and the contents of the temporary registers are stored in the path metrics registers.

To avoid overflow, the path metric register word length should be as small as possible. When the most likely path metric is selected, the decision value is also chosen. When the whole burst is calculated in the ACS unit, the freeze signal will rise, and the ACS unit will stop processing. This puts the Viterbi decoder in the trace-back mode.
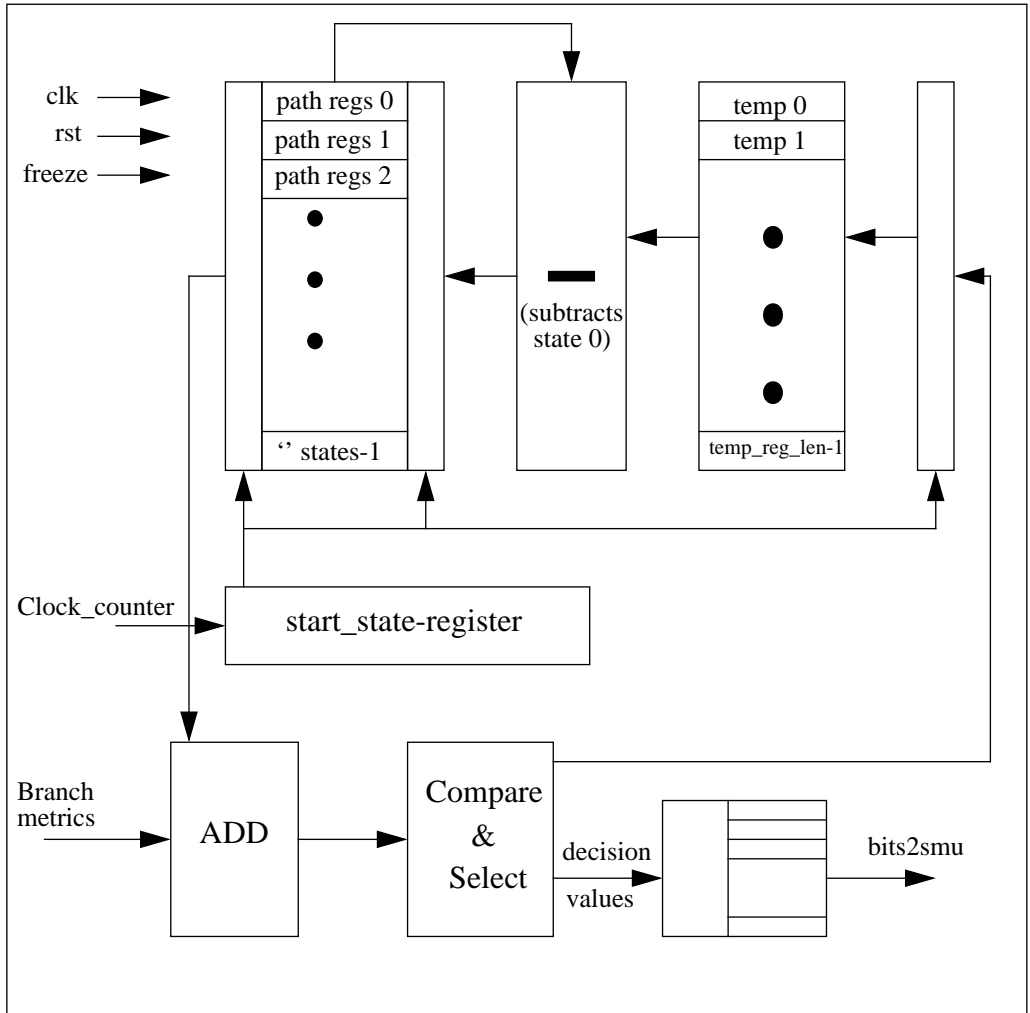
Figure 4 • *ACS Unit Block Diagram*

The SMU CONTROL Unit

The SMU control is used to manage the survivor memory. Based on the trace-back method, the ACS's computing and tracing back are carried out alternately. In other words, the alternation for reading the new branch metrics and the trace-back is controlled by the state machine. During the trace-back mode, the SMU control unit reads the decision values from the memory and outputs the decoded bits. When the branch metrics are inputted in bursts, the decision values are written in bursts into the survivor memory. The SMU control unit contains four counters. The first one is the ACS counter, counting from 0 to ACS_period-1 during the ACS operation. It

generates the write-enable and load-enable signals. The second counter is the load counter, counting from 0 to burst_length during ACS operation. It counts the input data and generates the internal control signals. The third counter is the TB counter, counting from 0 to mem_size-1 during the trace-back mode. It generates the internal control signals and the valid output signal. The fourth counter is the addr pointer, which serves as a memory address pointer, counting from 0 to mem_size-1.
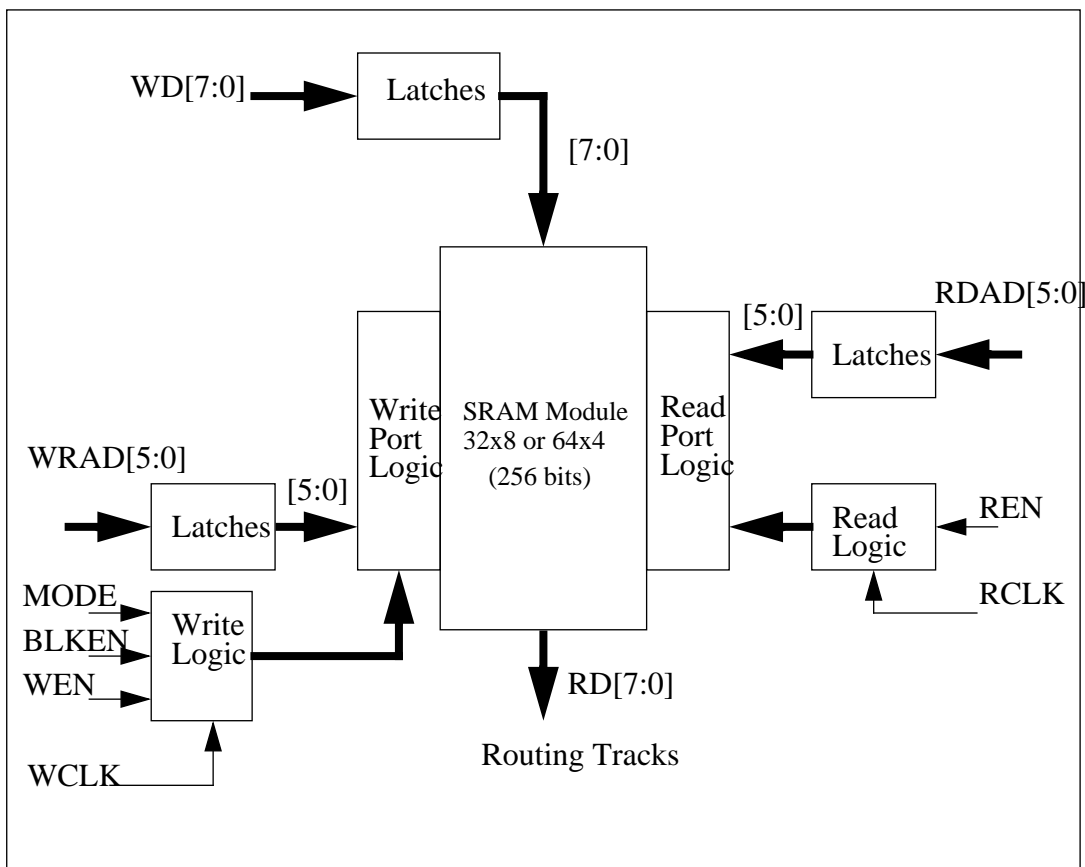
Figure 5 • *3200DX Dual-Port SRAM Block*

The Survivor RAM

The survivor RAM is a simple RAM model with read and write signals. While the write enable (Web) is active, it performs the write operation and stores the data in the appropriate location. While the read enable is active, it performs the read operation and outputs data via the output ports.

The Viterbi Decoder in the A32200DX

The Viterbi decoder can be efficiently implemented using the A32200DX device, because this device offers a fine-grained, register-rich architecture with the industry's fastest embedded dual-port SRAM and wide decode circuitry. An A32200DX device contains 10 blocks of the built-in SRAM (32x8). An A32200DX dual-port SRAM block is shown in Figure 5.
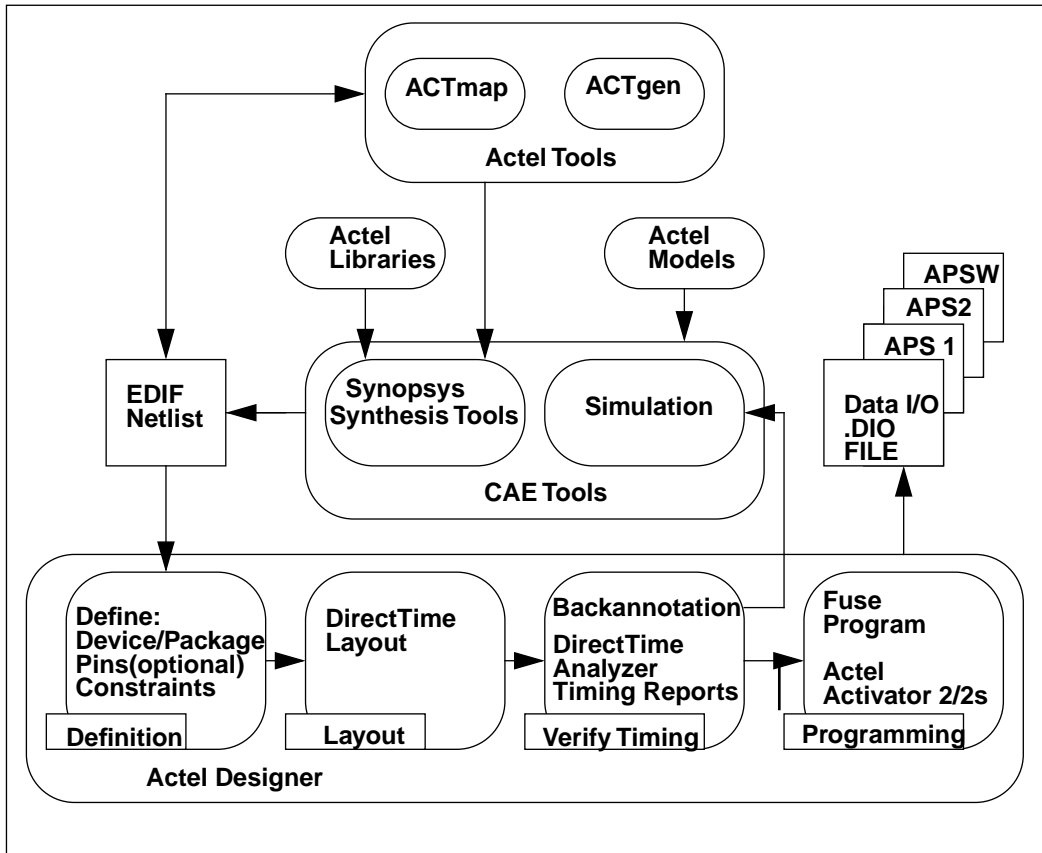
Figure 6 • *Actel-Synopsys Design Flow*

A complete Viterbi decoder is implemented in a A32200DX with the following parameter settings:

- Number of states in the trellis is 16.
- Number of bits to represent transition values is 8.
- Number of ACS cells is 4.
- The length of trace-back is 30.
- The length of received burst is 5.
- The initial path metric of the state 0 is -4.
- Survivor memory (RAM) word length is 8.

Using the Synopsys VHDL synthesis tool, the Viterbi decoder's VHDL code can be synthesized by Actel's 3200DX library and can produce an EDIF netlist. Figure 6 illustrates the complete Actel/Synopsys synthesis design flow. The netlist is imported into Actel's Designer Development tool and is placed and is routed using the DirectTime Layout tool in the A32200DX device. Timing requirements can be specified in the Direct-Time Layout tool to achieve your goal. As expected, the Viterbi decoder can achieve a speed of 10 MHz, using only 65% of the device's 20K logic gates. So, you still have more than adequate room to add logic to this design.

Conclusion

The Viterbi decoder, one of the best methods for decoding convolutional codes, can be achieved in an FPGA using third-party-developed high-level, language-based functions—intellectual property cores with a combination of high-performance A32200DX devices. Using TDF's IP model libraries is one of the fast paths to FPGA system design. You can easily access over 90 system-level FPGA models in the following fields: communications, DSP, multimedia, bus library, microcontrollers, and common function library. Particularly, you can customize, reuse, and synthesize these IP models through various computer-aided synthesis tools, such as Synopsys, ACTmap, Exemplar, Autologic, and Synplicity tools. Therefore, using IP models to design with FPGAs is an extremely efficient approach.

References

DSP-MACRO: Macrofunction data sheets: Viterbi decoder, version 1.0 from TDF.

B.P. Lathi, Modern Digital and Analog Communication Systems, 1989.

Actel and the Actel logo are registered trademarks of Actel Corporation.

All other trademarks are the property of their owners.