

Ultra Low Power Consumption Military Communication Systems

Sagara Pandu
Assistant Professor,
Department of ECE,
Gayatri College of Engineering
Visakhapatnam-530048.

ABSTRACT

New military communications require increased versatility for the transmission of a wide range of data, voice and images. There are many algorithms needed for communication like FFT (Fast Fourier Transform), DCT (Discrete Cosine Transforms), Viterbi coders and decoders, trellis techniques etc. Advances in VLSI technology continue to improve the quality of algorithm that can be implemented on a single die. But even though all these algorithms implementable on a single chip, they require large amount of computation and consequently require large die area and high levels of power dissipation.

In wireless communication, Viterbi decoder which consumes more power plays an important role in communication applications. Viterbi decoder is used to decode the received data which is encoded using convolution codes, which has less probability of error compared to other coding techniques. In many digital systems like Viterbi decoder, multiplexers and XOR gates are the major building blocks, which are often designed using transmission gate logic. So, to reduce the power consumption and number of transistors count in the decoder design, the transmission gates are replaced with the pass transistors which contain only one NMOS transistor. Since the eliminated PMOS transistor is large in size, the capacitance in the circuit is reduced thereby reducing the power, size and increase in the speed.

This technique is simulated using TANNER tool. The improved performance of Viterbi decoder using pass transistor logic is compared with that of the transmission gate logic and results were compared using TANNER tools.

General Terms

Viterbi decoder algorithm, The branch metric (BM), The path metric, Tanner Tools.

1. INTRODUCTION

Reduction of the power dissipation is obtained by reducing the capacitance in the circuit. The first step is to use minimum size transistors whenever possible. The second step is to use circuits, which heavily utilize transmission gates, and to implement these gates with single n-type transistors (Pass Transistor logic) instead of the two transistors generally used. Since the eliminated p-type transistor is usually larger than the n-type one, the input capacitance is reduced by a factor of more than 2. This reduction of input capacitance results in decrease in the delay of the circuit and 2) reduction of the dynamic power dissipation

Transmission Gate(TG) which consist of one n-MOSFET and one p-MOSFET acts as a switch where the NMOS passes strong '0' and the PMOS passes strong '1'. The transmission

gate logic is often used for the design of multiplexers and exclusive OR gates. Since the multiplexers and EXOR gates are the major blocks in many digital systems, the number of transistors and the power consumed is reduced by designing those circuits using pass transistor logic which consists of one NMOS gate.

A sample example to understand the technique is given below. In the Fig.1.1 a multiplexer design, which is heavily used in communication is shown using transmission gate and pass transistor logic.

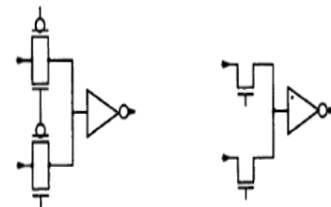


Fig.1.1 Multiplexer designed using transmission gate and pass transistor logic

Application of proposed technique

Error correction is an integral part of any communication system and for this purpose, the convolution codes are widely used as forward error correction codes. For decoding of convolution codes, at the receiver end Viterbi Decoder is being employed. The high speed and small area are two important design parameters in today's wireless technology. Viterbi decoders employed in digital wireless communications are complex and dissipate large power. The Viterbi decoding algorithm, proposed in 1967 by Viterbi, is a decoding process for convolutional codes in memory-less noise. The algorithm can be applied to a host of problems encountered in the design of communication systems. The Viterbi Algorithm (VA) finds the most-likely state transition sequence in a state diagram, given a sequence of symbols and is used to find the most likely noiseless finite-state sequence, given a sequence of finite-state signals that are corrupted by noise. This technique on widely used Viterbi decoder.

VITERBI DECODER ALGORITHM:

The decoding algorithm uses two metrics:

- The branch metric (BM) and
- The path metric (PM).

Branch metric

The branch metric is a measure of the "distance" between what was transmitted and what was received, and is defined for each arc in the trellis. In hard decision decoding, where we are given a sequence of digitized parity bits, the branch metric

is the Hamming distance between the expected parity bits and the received ones.

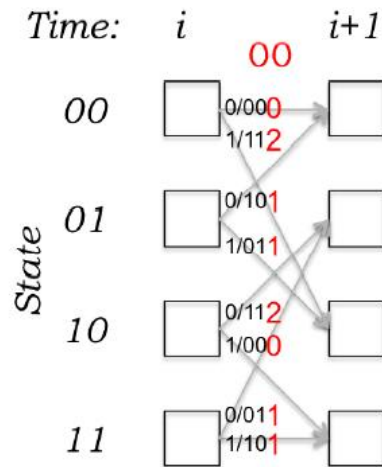


Fig.2.3 Branch metric

In this example, the receiver gets the parity bits 00. For each state transition, the number on the arrow shows the branch metric for that transition. The non-zero branch metrics correspond to cases when there are bit errors.

Path metric

The path metric is a value associated with a state in the trellis (i.e., a value associated with each node), it corresponds to the Hamming distance over the most likely path from the initial state to the current state in the trellis. By “most likely”, we mean the path with smallest Hamming distance between the initial state and the current state, measured over all possible paths between the two states that minimize the total number of bit errors.

The key insight in the Viterbi algorithm is that the receiver can compute the path metric for a (state, time) pair incrementally using the path metrics of previously computed states and the branch metrics.

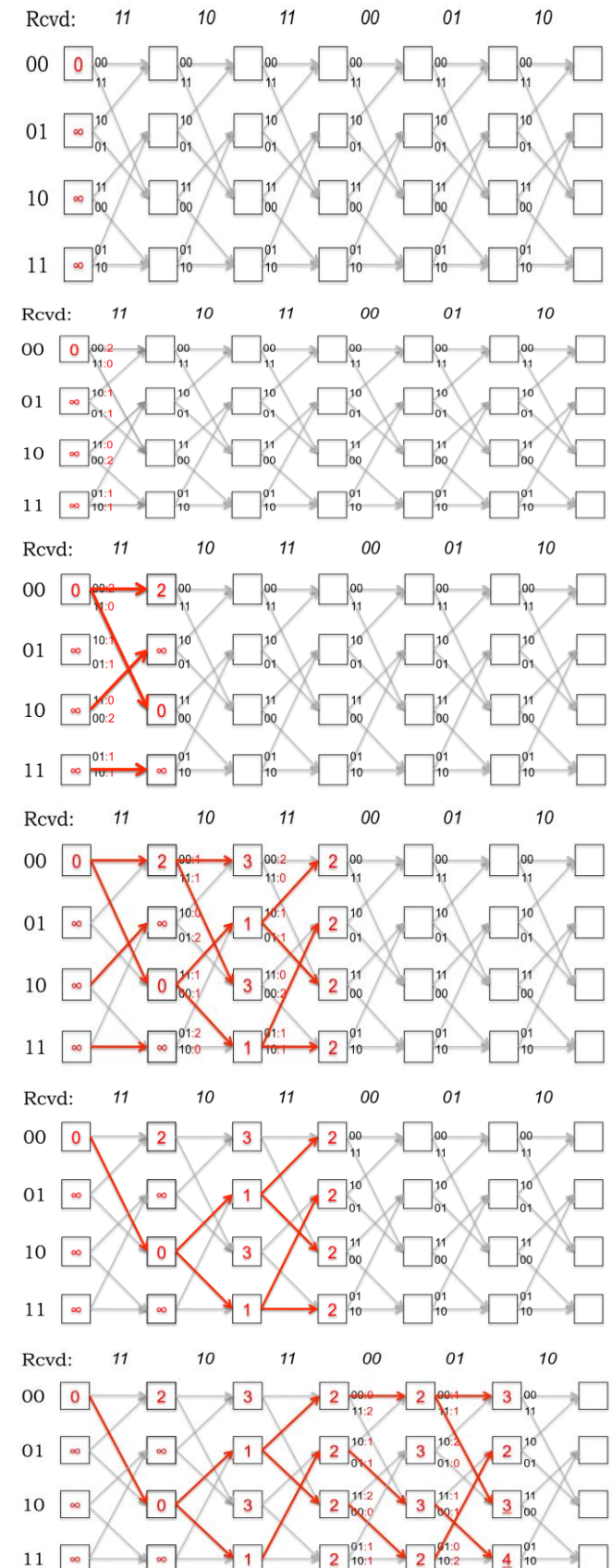
Finding the Most Likely Path

We can now describe how the decoder finds the most likely path. Initially, state ‘00’ is taken as the starting state by default if not mentioned. The main loop of the algorithm consists of two main steps: calculating the branch metric for the next set of parity bits, and computing the path metric for the next column. The path metric computation may be thought of as an add-compare-select procedure:

1. Add the branch metric to the path metric for the old state.
2. Compare the sums for paths arriving at the new state (there are only two such paths to compare at each new state because there are only two incoming arcs from the previous column).
3. Select the path with the smallest value, breaking ties arbitrarily. This path corresponds to the one with fewest errors.

Figure 2.4 shows the algorithm in action from one time step to the next. This example shows a received bit sequence of 11 10 11 00 01 10 and how the receiver processes it. The fourth picture from the top shows all four states with the same path metric. At this stage, any of these four states and the paths leading up to them is most likely transmitted bit sequences (they all have a Hamming distance of 2). The bottom-most

picture shows the same situation with only the survivor paths shown. A survivor path is one that has a chance of being the most likely path. There are many other paths that can be removed because there is no way in which they can be most likely.



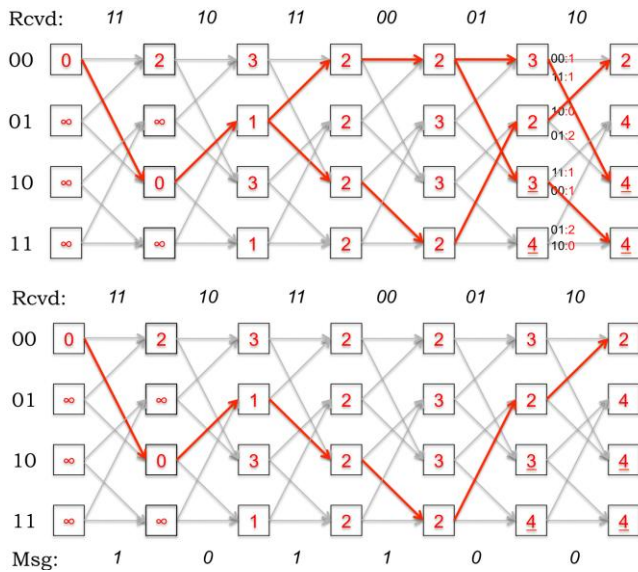
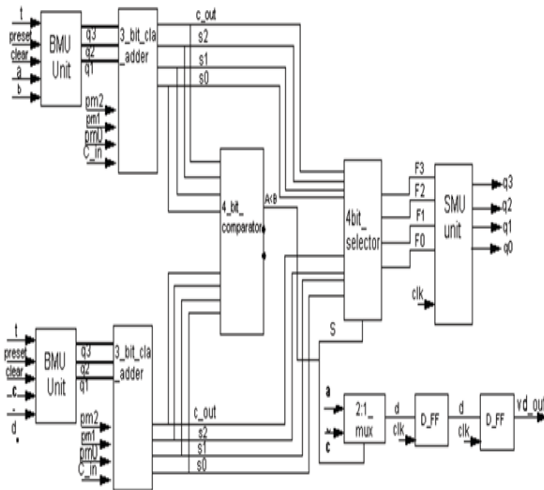


Fig.2.4 Viterbi decoding in action with decoding message

The reason why the Viterbi decoder is practical is that the number of survivor paths is much, much smaller than the total number of paths in the trellis. Another important point about the Viterbi decoder is that future knowledge will help it break any ties, and in fact may even cause paths that were considered “most likely” at a certain time step to change. Figure 9-4 shows decoding all the received parity bits to produce the most likely transmitted message, which has two bit errors.

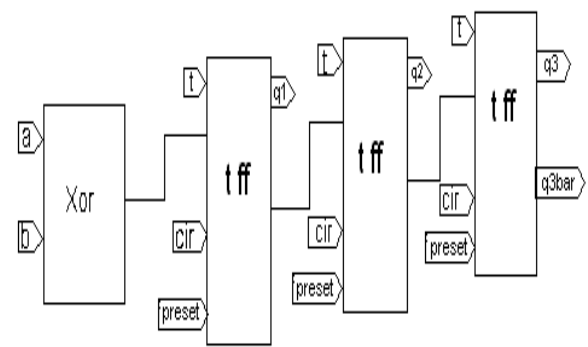
DESIGN:

We implemented the Viterbi decoder by transmission gate and pass transistor logic. The entire block diagram of the Viterbi decoder is shown in fig



The Viterbi decoder consists of the Branch Metric Unit, Add compare and Select Unit, and the Survivor Path Memory Unit.

The Branch Metric Unit (BMU) is used to measure the Hamming distance between the received sequences with the expected code sequence. The BMU consists of a two input EXOR gate and three bit asynchronous counter. The counter is designed using T flip flops, which are designed using d flip flops.



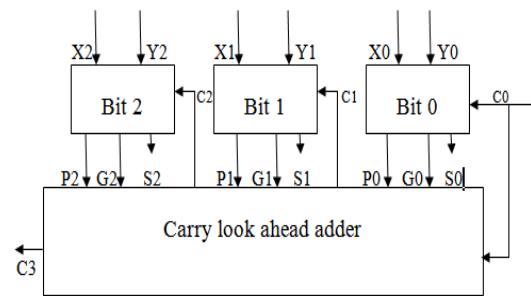
BMU Unit

Add Compare and Select Unit:

Carry look ahead adder

In the design we used a 3 bit carry look ahead adder. In the case of parallel adder, the speed with which an addition can be performed is governed by the time required for the carries to propagate through all of the stages of the adder. The carry look ahead adder speeds up the process by eliminating this ripple carry delay.

This is based on two additional functions of full adder called carry generate and carry propagate functions.



Block diagram of Carry Look ahead Adder

Comparator

A comparator is a logic circuit used to compare the magnitudes of two binary numbers. It gives the greater than, less than and equal to outputs depending on the input bits. An XNOR is a simple comparator.

In our design we used a 4 bit comparator which is designed by 2 bit comparators. The expressions for the outputs of two bit comparator with 2 inputs as $A=A_1A_0$ and $B=B_1B_0$ are:

$$A > B : G = A_1 \cdot \bar{B}_1 + (A_1 \odot B_1) \cdot A_0 \cdot \bar{B}_0$$

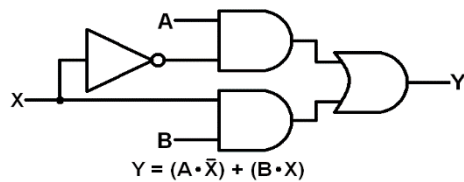
$$A < B : L = \bar{A}_1 \cdot B_1 + (A_1 \odot B_1) \cdot \bar{A}_0 \cdot B_0$$

$$A = B : E = (A_1 \odot B_1) \cdot (A_0 \odot B_0)$$

Selector

The multiplexer is the major block in the selector. The selector selects the appropriate branch. The selector unit consists of four 2:1 multiplexers.

2:1 multiplexer has two input lines, one data select line and one output line. The logic level applied to the select line determines which data input is passed to the output. If A and B are the inputs with select line X, the logic diagram and expression for output Y are given below.



Logic diagram for Selector

The truth table is below.

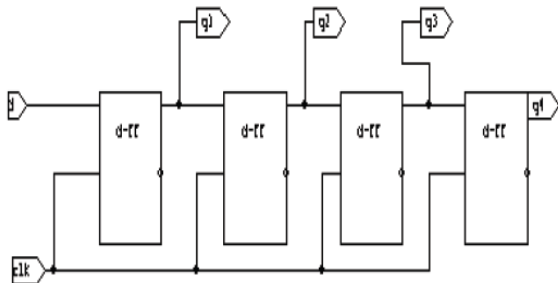
Truth Table for Selector

X	Y
0	A
1	B

The outputs of adder unit are given as inputs to the selector unit. The output (less than bit) of the comparator is given as selection line for the selector.

Survivor Memory Unit (SMU):

The important step in the decoding process is finding the survivor path. The output of the selector is the survivor path and that path is stored in the survivor memory unit. The survivor memory unit (SMU) is designed by cascading serial in serial out (SISO) shift registers. The length of the shift registers depends on the length of the encoder. The SMU unit consists of four SISO shift registers. When the positive clock pulse is applied, the data D is transferred to the output of the flip-flop and for each positive clock cycle the value stored in one register is shifted to another register.



Survivor Memory Unit

SOFTWARE USED:

The viterbi decoder is implemented in tanner tools in transmission gate and pass transistor logics.

Tanner Tools Consists of following Machines

It offers various types of design and verification tasks which include

1. S-EDIT (Schematic Edit)
2. T-EDIT (Simulation Edit)
3. W-EDIT (Waveforms Edit)
4. L-EDIT (Layout Edit)

3 Steps in Tanner Tools

To implement a circuit in Tanner EDA software, S-edit platform was used. This implementation is done by following steps:

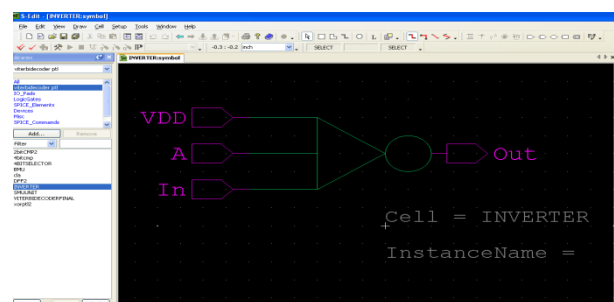
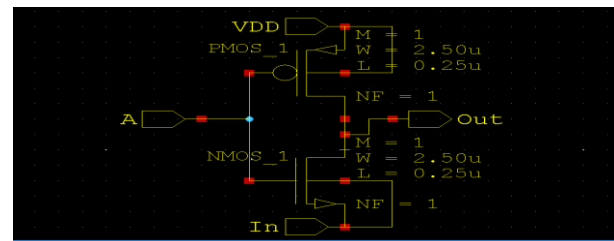
Steps to Launch S-EDIT

1. First of all double click on the icon of s-edit on the desktop, and then a new window will open.
2. Go to >>file >> New >> New Design Select New Design
3. Design Name: Give the name your design as you wish create a Folder: Give the path where you want to save the S-Edit Files. Then Click on 'OK'.
4. Now to add libraries in your work click on **Add** left on the library window. Give the path where Libraries are stored. As for example C:\Documentsand Settings\My Documents\Tanner EDA\Tanner Tools v13.0\Libraries\All\All. Tanner.
5. Now to create new cell
Go to cell menu >> New view
Select 'New view' the new cell will appear like below:

Design = your design name. Cell = cell no. Cell no you can change but your design name **XXX** will be same for different cells.

6. Design name should be changed only when you are going to design another new circuit design.
7. After schematic window opens, we connect the circuit and then create a symbol for that circuit. Like for example, we make the connections for an Inverter, XOR-gate etc., and create a symbol for this so that the symbol can be used in any other circuit.
8. This way, different symbols are created and an example is shown below.

If we want to move any gate or any device we must click on that and by pressing ALT+MOUSE and then move it to the desired place.



9. Once the connections are made, we have to give the supply and ground connections to the circuit properly.
10. The next step is simulation. This is done by following the steps given below.

So now we use the T-Edit.

For simulation go to>> tools>> T-spice>> 'ok'

```

INVERTER
* SPICE export by: SEDIT 13.00
* Export time: Thu Apr 24 23:45:56 2014
* Design: viterbidecoder.pul
* Cell: INVERTER
* View: Main
* Export as: top-level cell
* Export mode: hierarchical
* Exclude .model: no
* Exclude .end: no
* Expand parts: yes
* Wrap lines: no
* Root path: C:\Documents and Settings\Administrator\Desktop\tanner mine\viterbidecoder.pul
* Exclude global pins: no
* Control property name: SPICE

***** Simulation Settings - General section *****
.model nmos1 nmos level=1
.model pmos1 pmos level=1
Va & GND PULSE 0 2.5 1ps 1ps 4ns 8ns
VIN In gnd 0
VDD VDD In 2.5V
.tran 1ps 100n
.print V(A) V(OUT)
***** Simulation Settings - Parameters and SPICE Options *****

*----- Devices: SPICE.ORDER > 0 -----
NMOS1_1 Out In In NMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u
PMOS1_1 Out In VDD PMOS W=2.5u L=250n AS=2.25p PS=6.8u AD=2.25p PD=6.8u

Status InputFile Out... StartDate/Time Exp...

```

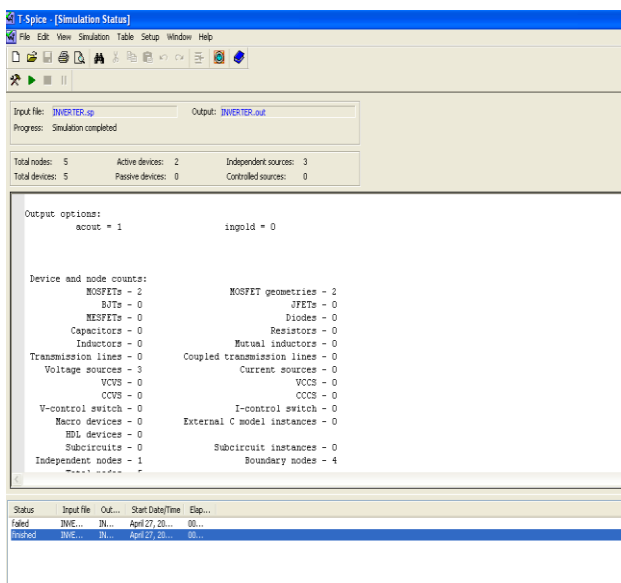
T-SPICE Command Window

A window will be opened as shown above. In this we have to give the commands anywhere before the end statement.

Start time, stop time, CLK signal everything is declared here as command statements.

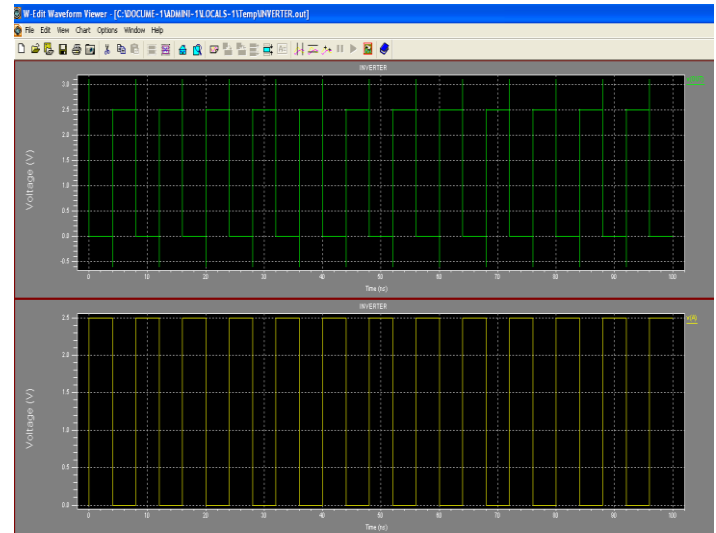
Finally .print, .power, etc. command statements are used here to get the output required. Click on the simulation button or F5 key to get the results.

Now another window will be opened showing the results as no. of MOSFETS used, time taken for the entire simulation, no. of nodes, no. of interconnections etc. will be displayed as shown in below fig.



T-SPICE Results Window

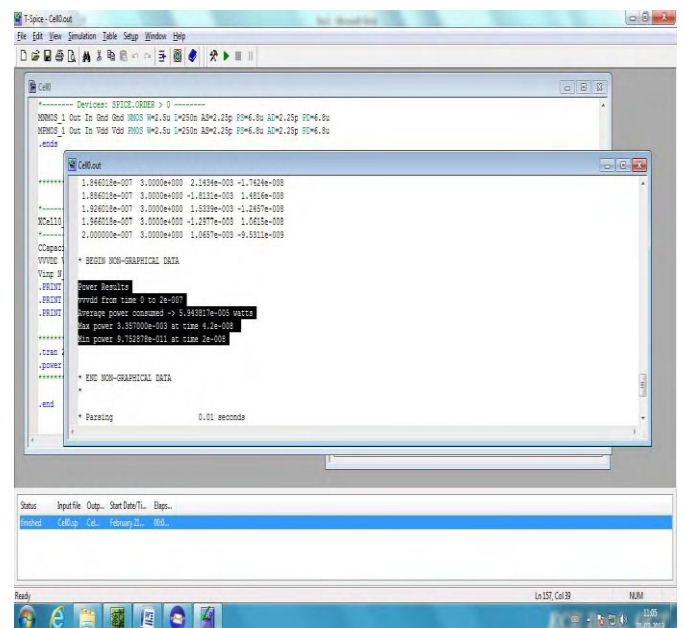
Output waveforms for inverter are obtained now directly on W-EDIT platform as given below.



W-EDIT Waveform Window

In order to calculate power, Save the net list in TSPICE window in required location with .SP extension before simulating in t-spice.

Now in the TSPICE command window .power VDD command is entered and the design is simulated. In the window open the saved file previously, with .out extension. The power analysis is displayed at the end of the out file opened as shown in the below figure.

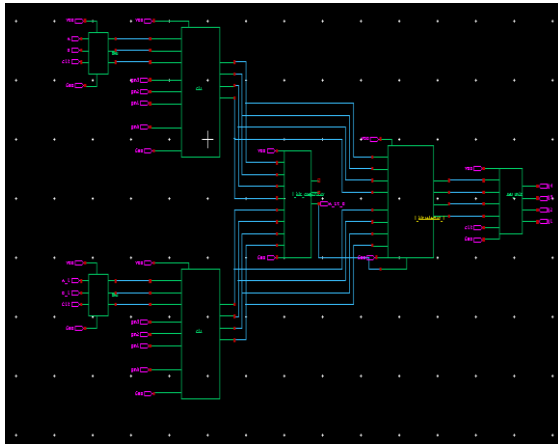


Power analysis displayed in the TSPICE window

2. RESULTS AND CONCLUSION

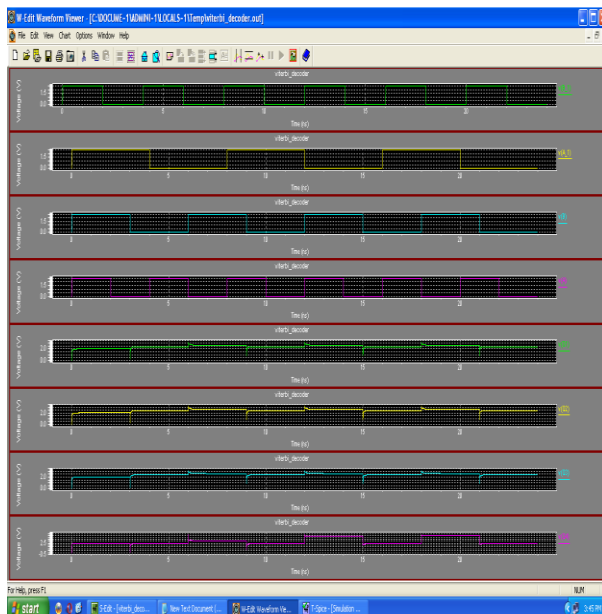
The goal of this project is to design VITERBI DECODER. For this design of complete viterbi decoder we have designed the individual blocks for XOR gate, D-flipflop, Inverter, 3-bit carry look ahead adder, 4-bit comparator, 2:1 multiplexer for selector unit. These all are used to build different block units for our entire design. These blocks were simulated and the corresponding outputs were verified. The viterbi decoder was implemented and its functionality was verified by simulating the circuit for different input combinations.

Viterbi decoder designed in tanner tools



Schematic of VITERBI DECODER

Simulation Results



Performance Comparison :

The Viterbi decoder designed using Transmission gate and Pass transistor logic is simulated using TANNER tools and the performance of the design obtained in both logics regarding the power, transistor count and simulation time is given in the following table.

Performance analysis

S. no.	Viterbi Decoder	Power (mw)	Transistor Count	Simulation Time(sec)
1	Transmission gate	0.1386	654	67.56
2	Pass transistor	0.0789	556	14.36

3. FUTURE SCOPE

We have designed the Viterbi Decoder using Transmission Gate and Pass Transistor logic where it was observed that Pass Transistor logic requires less number of transistors and consumes less power.

However, the circuits designed using pass transistor logic may not operate satisfactorily when the input V_{in} is high because NMOS transistors does not transmit full input voltage instead transmit reduced voltage ($V_{in} - V_{tn}$) there by reducing the output voltage swing. These problems can be eliminated by reducing the threshold voltage of the NMOS transistor not below 0.5V due to charge retention problems. So the Viterbi Decoder can be designed using pass transistor logic with modified threshold voltages for best output voltage levels.

4. CONCLUSION

The Viterbi decoder is designed in circuit level using Transmission gate logic and pass transistor logic. The design is simulated using TANNER tool in 250nm technology 2.5V VDD. The simulation results shows the power consumption of the Viterbi Decoder using Pass transistor logic provides 43% of lower power consumption comparing to that of the design by using Transmission gate logic. The transistor count is also reduced by 15% by Pass transistor logic comparing to Transmission gate logic.

Obtained results indicate that Pass transistor logic consumes less power and area compared to transmission gate logic.

5. REFERENCES

- [1] LOW POWER CONSUMPTION SYSTEMS Menahem Lowy, Chi-Yuan Chin, Jerome J. Tiemann General Electric / Corporate Research and Development Schenectady, NY 12301 July 1990
- [2] M.S. Adler, "GE High Density Interconnect: a solution to "The system interconnect problem," SPIE International Symposium on Advances in Interconnects and Packaging, Conf. Proceedings, Nov. 1990.
- [3] K.M. Chu and D.L. Pulfrey, "A Comparison of CMOS Circuit Techniques: Differential Cascade Voltage Switch Logic versus Conventional Logic", IEEE J. Solid- State Circuits, vol. SC-22, pp.528- 532, August 1987.
- [4] Jinjin He, Huaping Liu, Zhongfeng Wang, Xinming Huang, and Kai Zhang "High-Speed Low-Power Viterbi Decoder Design for TCM Decoders ", IEEE transactions on very large scale integration (vlsi) systems, VOL. 20, NO. 4, APRIL 2012.
- [5] Lang L, Tsui C.Y and Cheng R.S.1997. "Low power soft output Viterbi decoder scheme for turbo code decoding", IEEE Conference-Paper, ISCAS ,97,New York, USA, 24, 1369-1372.
- [6] WWW.Wikipedia.org
- [7] Slide Share.net