Arun Avudainayagam

Nandkishore Muralidharan

Ramakrishna Venigalla

EEL 6323: Advanced VLSI design project

# The AMV 6323: A programmable, variable rate Turbo encoder IC for next-generation wireless devices

*UF's Best Bet to Dominate the Wireless IC Market in the 21st Century.*

# Abstract


## *AMV 6323: A versatile IC to combat channel errors.*

Our design project[1] addresses the design of a programmable, variable rate turbo encoder. Turbo codes are currently the most powerful forward error correction (FEC) codes. They are also specified in the 3GPP (WCDMA) and 3GPP2 (CDMA 2000) standards for the next-generation wireless networks. Because of the number of different turbo codes used in different standards, making an IC that is custom built for each of the standards is not a sound business strategy. Not only does this increase design time, but it also increases various other costs associated with fabricating the IC and other logistic expenses. Further, this also reduces the profit margin for the vendor because of the reduced customer base.

Our IC, the AMV 6323, solves this problem by integrating a number of different turbo codes into the same chip in a programmable manner. This increases the customer base many fold with a negligible increase in design time. Different vendors can buy the same chip and program it according to their standards by making use of the select lines provided on the chip.

With the integration of voice, data and multimedia in wireless networks, quality of service becomes an important issue. Different services have different loss requirements and hence a chip that can provide different levels of protection for different types of data is required. The AMV 6323 is capable of providing two different levels of protection depending on the application.

The AMV 6323 also has an on-chip test module. This has the potential to result in tremendous savings to the manufacturer both in terms of time and money. Moreover, since mobile devices are often constrained by their power resources, the ability to save power whenever possible becomes an important design issue. The AMV 6323 can operate with the same efficiency using less battery power while not sacrificing the high data rate.

Our design project also includes a cost estimate of the chip together with simulation results and application notes.


[1] Computer path to final layout: /home/users/0/venigalr/Rama_tests/vlsi_test/

# TABLE OF CONTENTS


ABSTRACT ..... 2
TABLE OF CONTENTS ..... 3
1.0 INTRODUCTION ..... 4
2.0 THE TURBO ENCODER ..... 6
2.1 The Constituent Convolutional Encoders ..... 7
2.2 The Interleaver ..... 8
2.3 The Puncturer ..... 8
3.0 CHIP DESCRIPTION ..... 9
3.1 Shift Register ..... 10
3.2 'Interleaver Clock' Generator ..... 10
3.3 Encoder ..... 11
3.4 Puncturer ..... 12
3.5 Chip Floor Plan ..... 14
4.0 APPLICATION NOTES ..... 15
4.1 Turbo Code Selection ..... 16
4.2 Design Specifications ..... 20
5.0 TEST CAPABILITY ..... 21
6.0 FUTURE DIRECTIONS ..... 22
7.0 SUMMARY ..... 23
REFERENCES ..... 26
APPENDIX A: YIELD CALCULATIONS ..... 27
APPENDIX B: LVS OUTPUTS ..... 28
APPENDIX C: CHIP LAYOUT ..... 30
APPENDIX D: SIMULATION RESULTS ..... 31

# 1.0 Introduction

## *Turbo codes are capable of achieving capacities within fractions of dB of the Shannon's channel capacities!*

With the concept of ubiquitous computing becoming more and more of a reality, the need for an IC that can be used in any wireless system, anywhere in the world increases. Also with the advent of wireless internet, WAP enabled cell phones, PDAs and other wireless devices, data services along with voice services become an integral part of a wireless network. Hence, wireless devices should be capable of handling both data and voice traffic efficiently. However, voice and data services should not be handled the same way because of their different quality of service (QoS) requirements. Voice services can tolerate some loss but data services are very stringent in their loss requirements. A simple way to combat loss or erroneous data packets is to request a retransmission. However, this is not an efficient strategy as it reduces system bandwidth. An efficient way to combat errors in a packet is to use powerful forward error correction (FEC) codes.

The FEC codes like any other code have their own advantages and disadvantages. These codes add redundancy to the data transmitted and use this redundancy to correct errors in the packet. Adding more redundancy increases the error correction capability of a code but it also increases the overhead and hence decreases the throughput of the system. Hence, the amount of redundancy to be added to a message should be carefully chosen based on the application, in order to maximize the throughput of the system. For example, since data packets have more stringent loss requirements, more redundancy should be added. On the other hand, since the quality of voice traffic is acceptable even when a couple of packets are in error, lesser redundancy should be added. This technique increases the overall throughput of the system. The amount of redundancy added to a message is specified by a performance measure called the code rate. The code rate for an error correcting code is defined as

$$\text{code rate} = \frac{\#\ of\ information\ bits}{total\ \#\ of\ bits} = \frac{\#\ of\ information\ bits}{\#\ of\ information\ bits\ +\ \#\ of\ redundant\ bits}$$

Hence, it is seen that the lower the code rate, the more the redundancy and the more powerful the code. Similarly a higher code rate implies a less powerful error correction code. Hence, wireless devices require an IC that implements an FEC algorithm, and that is capable of producing different code rates (i.e. a lower code rate for data applications and a higher code rate for voice applications) for different applications.

Further, with advanced techniques like adaptive modulation, coding and turbo-coded ARQ schemes becoming popular, the use of turbo codes in the wireless networks of tomorrow is inevitable. Different turbo codes are used by different countries and by various protocols and standards. Hence, an IC that can be used in any mobile device anywhere in the world and that can be used to exchange data over any protocol is required.

The AMV 6323 is such an IC. The AMV 6323 supports two code rates (one half and one third) and is programmable in the sense that it can generate 48 different symmetric turbo

codes including the ones used in the 3GPP and 3GPP2 standards. In addition, the AMV 6323 can also be programmed to produce $48^2$ asymmetric turbo codes[2]. We envision that this IC is versatile enough to be used in any network that uses turbo codes for error correction, anywhere in the world.

The rest of the report is organized as follows: Section 2 describes the turbo encoder and the constituents of the turbo encoder. Section 3 describes the different modules used in the chip. Section 4 describes the operation of the chip in a typical wireless device. Section 5 describes the testing capability of the chip.

[2] See section 2 for a description of symmetric and asymmetric turbo codes.

# 2.0 The Turbo Encoder

## *Turbo codes are the most powerful error-correcting codes!*

The turbo encoder consists of two rate ½ recursive, systematic convolutional encoders and an ***interleaver*** (also called ***permuter***). The constituent encoders are arranged in a parallel concatenation as shown in figure 1. A ***puncturer*** is used to control code rates.

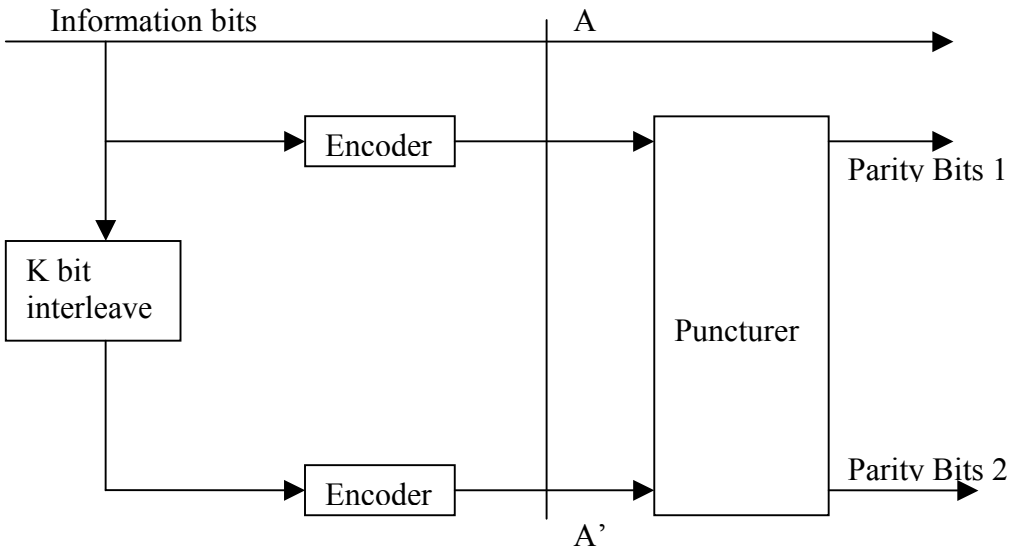



FIGURE 1: Block diagram of the turbo encoder

At AA' there are three output bits for every input information bit. Hence, the rate of the code, which is defined as the ratio of the number of input bits to the number of outputs bits, is 1/3 at AA'. As the code rate decreases (there are a greater number of output bits for the same number of input bits), the error correcting capability of the code increases. However, there is rate penalty associated with this because of the overhead involved in transmitting the parity bits. Hence, the effective throughput (rate of transmission of information bits) decreases. Therefore the common practice is to have rate 1/3 as the lowest code rate and achieve higher code rates by puncturing some of the parity bits. See section 2.3 for more details on puncturing.

The next couple of sections describe the different components of the turbo encoder.

## 2.1 The Constituent Convolutional Encoders

The block diagram of the constituent convolutional encoders is shown in figure 2.

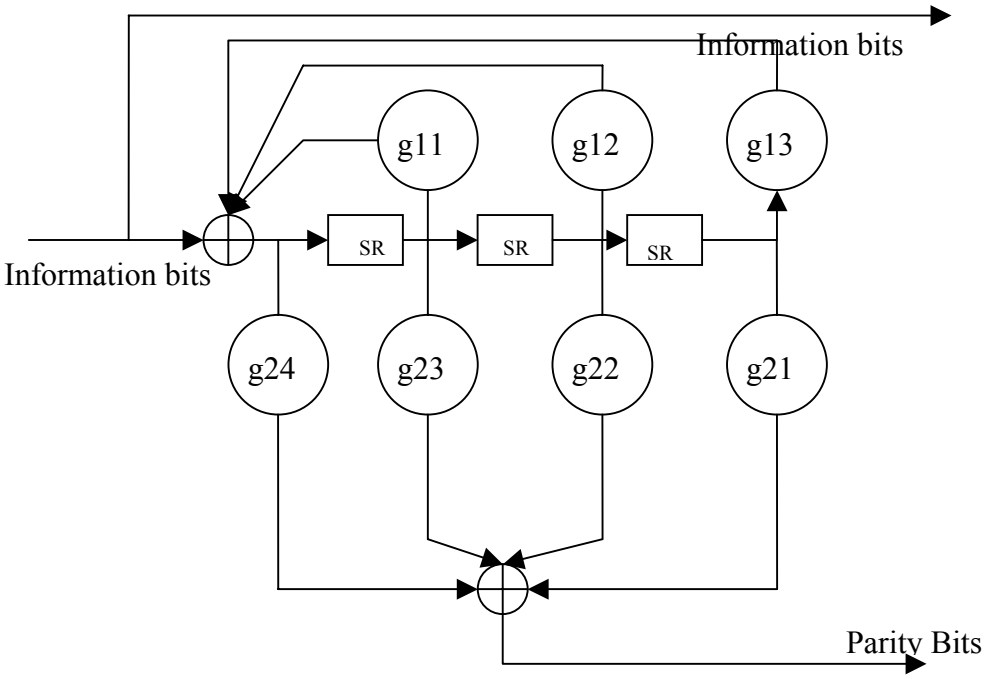



FIGURE 2: Recursive Systematic Convolutional Encoder

The encoder shown above is called ***recursive*** due to the presence of the feedback paths. It is a ***systematic encoder*** because the information bits appear without any change at the output. Since for every information bit, there are 2 output bits, the code rate of this encoder is ½.

The basic structure is formed by the shift registers. The memory of the code (the number of shift registers) is 3 for the constituent codes used in most turbo encoders. The performance of the convolutional codes increases with the number of memory elements but in the case of turbo codes, the decoder complexity increases tremendously with an increase in the memory elements. Hence, the number of shift register elements is restricted to 3 in the constituent encoders.

The weights marked in the circles in FIGURE 2 can either be 0 or 1. A 1 indicates a connection from the shift register to the summing block and a 0 indicates no connection to the summing block. The feedback and feed-forward connections are usually specified in the octal format. For example, a (13, 15) turbo code refers to a code with feedback connections 1011 (13) and feed-forward connection defined by 1101(15). Thus, referring to figure 2, g1=[0,1,1] and g2=[1,1,0,1] (g2 is of the form [g24,g23,g22,g21] ). Thus, by changing the connection coefficients, different turbo codes can be obtained. For example, setting g11 and g22 equal

to 0 is equivalent to having a turbo code of memory 2. The (13,15) code is used in the CDMA2000 (3GPP2) and WCDMA (3GPP) standards.

Most turbo encoders have identical constituent convolutional codes for the 2 encoders. Such turbo codes are called ***symmetric*** or regular turbo codes.

## 2.2 The Interleaver

The function of the ***interleaver*** is to permute the input to encoder 1 in a pre-determined fashion and feed it to encoder 2. The performance of conventional convolutional codes is limited by certain input sequences that create output codewords that are easily corrupted*. These output codewords are called low-weight codewords. If the input to encoder 1 is a sequence that causes a low-weight codeword, then by permuting it before feeding it to encoder 2 ( which is identical to encoder 1), we are avoiding a low-weight codeword at the output of encoder 2. Hence, we are ensuring that at least one of the encoders produces a good codeword (one that is not easily corrupted). This is one of the reasons for the good error correction capability of the turbo code

There a number of different types interleavers to choose from depending on the application. Some typical interleavers include block interleavers, random interleavers, S-random interleavers, circular-shift interleavers and algorithmic interleavers. Each of these interleavers has its own advantages and disadvantages). The AMV 6323 is targeted towards mobile wireless devices including cell-phones. It should therefore be able to handle voice packets that are relatively small (around 300 bits). Since the random interleaver provides a very good error performance, it was implemented in the AMV 6323.

## 2.3 The Puncturer

The rate of the code produced by a turbo code alone is 1/3. To get higher rate codewords, a puncturer is used to delete some of the parity bits. For example, to get a rate 1/2 , only one of the parities is sent for each bit i.e., for the even information bits, parity 1 is sent and for odd bits parity 2 is sent. Hence, for every information bit, we have two bits at the output giving a rate 1/2 code. Higher rate codewords can be achieved by puncturing more parity bits. The AMV 6323 is capable of producing codewords with rate 1/2 and 1/3.

See [1] for more details on the components of the turbo encoder.

* By a codeword getting corrupted it is implied that the codeword can be mistaken for another codeword and cause errors while decoding.

# 3.0 Chip Description

## *The whole is more than the sum of its parts*

The schematic of the turbo encoder used in our project is shown below. The operation of the turbo encoder begins with 64 information bits being stored sequentially in a buffer of 64 shift registers. These 64 bits will be the input to the first encoder when the 'Load' clock is ON. Once the 64 bits are loaded into the shift registers, the 'Interleaver Clock' signal is now turned ON and the 64 bits are interleaved and stored in another buffer of 64 shift registers which will be input to the second encoder. The 'Interleaver Clock' signal is now turned OFF. Both the sets of 64 bits are simultaneously loaded into the two encoders. The systematic bit and the output of the two encoders are fed into the puncturer in a parallel fashion and the resulting output is a serial stream. The required code rate is achieved by dropping selected bits. The serial output is then sent for transmission to the wireless channel.

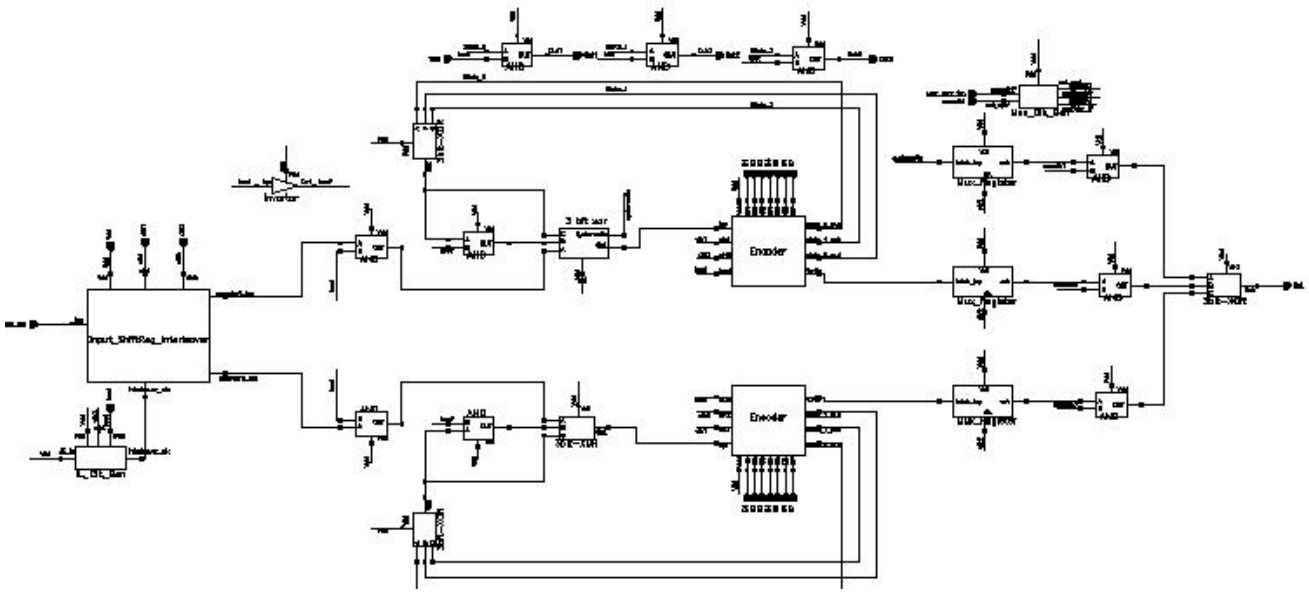



FIGURE 3: Schematic of the Turbo Encoder IC

The schematic of the various components used and a description of their operation follows.

## 3.1 Shift Register

The schematic of the shift register is shown in figure 4.

The basic components used in the shift register are the

- Transmission gate
- AND gate
- Inverter

When the 'Load' and 'Clock1' signals are ON, data enters the loop within the shift register. This value is stored until the next bit enters the shift register in the next 'Clock1' cycle. When 'Clock2' turns ON the data stored inside the shift register is transferred out. When 'Clock1' turns ON at the next clock cycle, a new value enters the shift register and the previous value is overwritten. It should be noted that 'Clock1' and 'Clock2' are two non-overlapping signals.

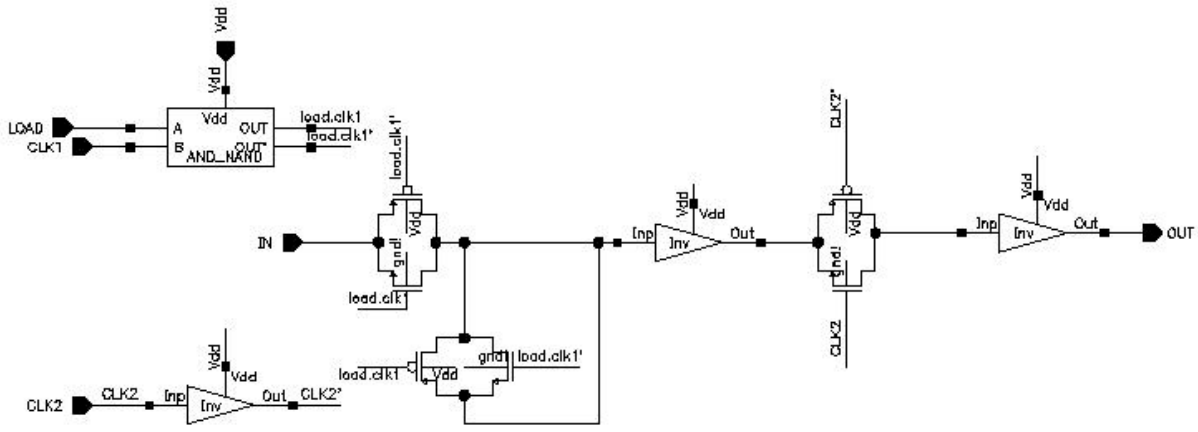



FIGURE 4: Schematic of the shift register

## 3.2 'Interleaver Clock' Generator

The interleaver requires all 64 bits to be available for successful operation. Hence, 64 bits are first input and stored in the first buffer. Then the bits are interleaved and stored in the second buffer. The 'Interleaver clock' is the signal that controls this operation. It has to turn ON after 64 clock cycles and turn OFF after the values have been transferred to the second buffer.

The 'Interleaver Clock' is generated internally using the 'Load' and the 'Clk2' signals. The circuit is implemented using a 6-bit counter and its schematic is shown in figure 5. The basic components of this module are:

- JK Flip-flop (NAND gate implementation)
- AND gate
- NOR gate
- Inverter

Once the 6-bit counter counts 64 clock cycles, the 'Interleaver Clock' is turned ON for one clock cycle and then turned OFF again. The bits in the first 64-bit buffer are interleaved onto the second 64-bit buffer in this one cycle.

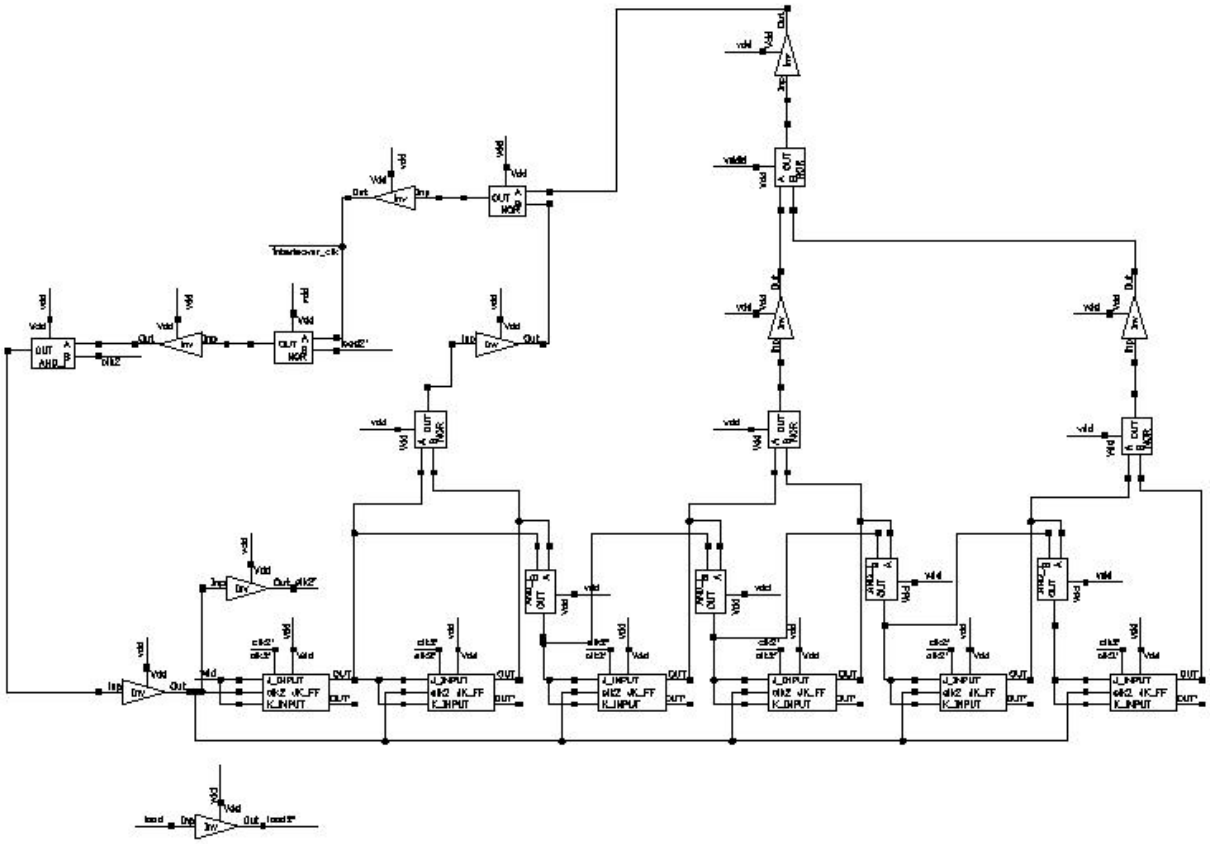


FIGURE 5: Schematic of the interleaver clock generator

## 3.3 Encoder

The schematic of the encoder used is shown in figure 6.

The basic components used in the encoder are :

- Shift register
- AND gate
- XOR gate and the
- Inverter

The feedback and feed-forward connections are implemented by performing the AND operation on the signal and the corresponding select signal (S1-S7). The encoder operates by first computing the 64 (block length) parity bits for 64 information bits. After 64 bits are evaluated, trellis-termination bits should be added to drive the encoder to the all zero state before the next 64 parity bits can be computed. This is implemented by adjusting the 'Load' signal to be ON for 64 clock cycles and OFF for the next 3 clock cycles. The encoder is terminated by performing the XOR operation on the feedback signal (the output of the XOR gate at the end of the feedback path) with itself for 3 consecutive clock cycles.

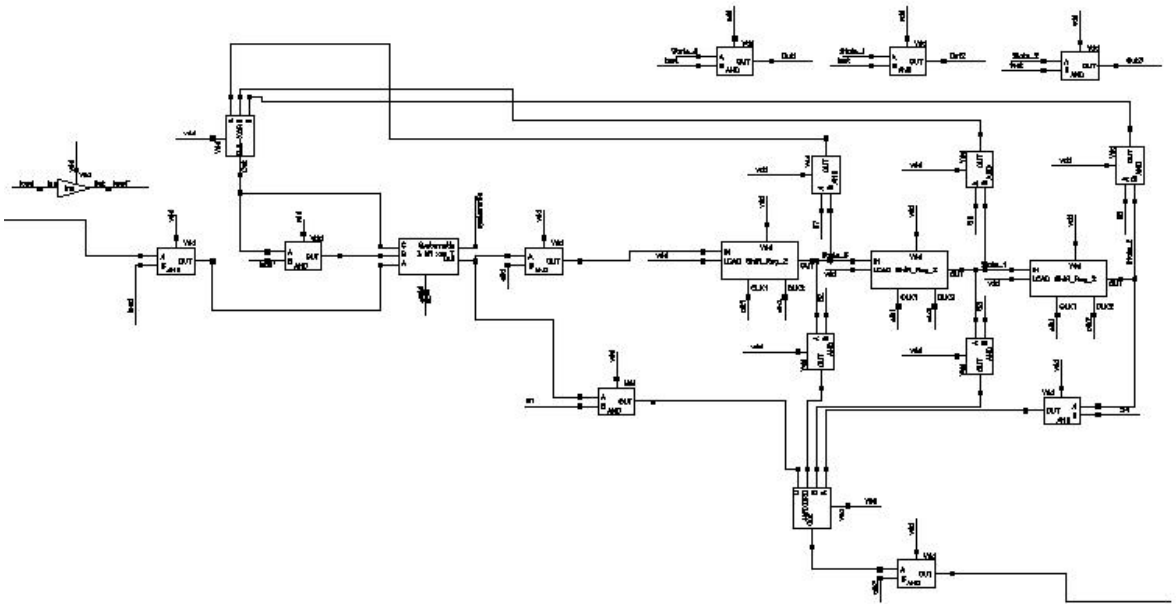


FIGURE 6: Schematic of the Recursive Convolutional Encoder

## 3.4 Puncturer

The figure below shows the schematic of the puncturer used in the AMV 6323.

The basic components used in the puncturer are

- AND gate

- XOR gate
- Inverter

This circuit performs two simple operations. Firstly, it takes the three output bits (Systematic bit, Parity bit 1 and Parity bit 2) and converts them to a serial form. Secondly, one of the two possible code rates is obtained by either selecting or dropping the parity bits. The code rate is determined by the user by providing the appropriate clock signals to the Muxclk1 and Muxclk2 input pins. The Muxclk3 signal is internally generated from the signals muxclk1 and muxclk2 using the circuit shown in figure 8. Muxclk1, Muxclk2 and Muxclk3 select the bits from the streams systematic bits, Parity 1 and Parity 2 respectively.

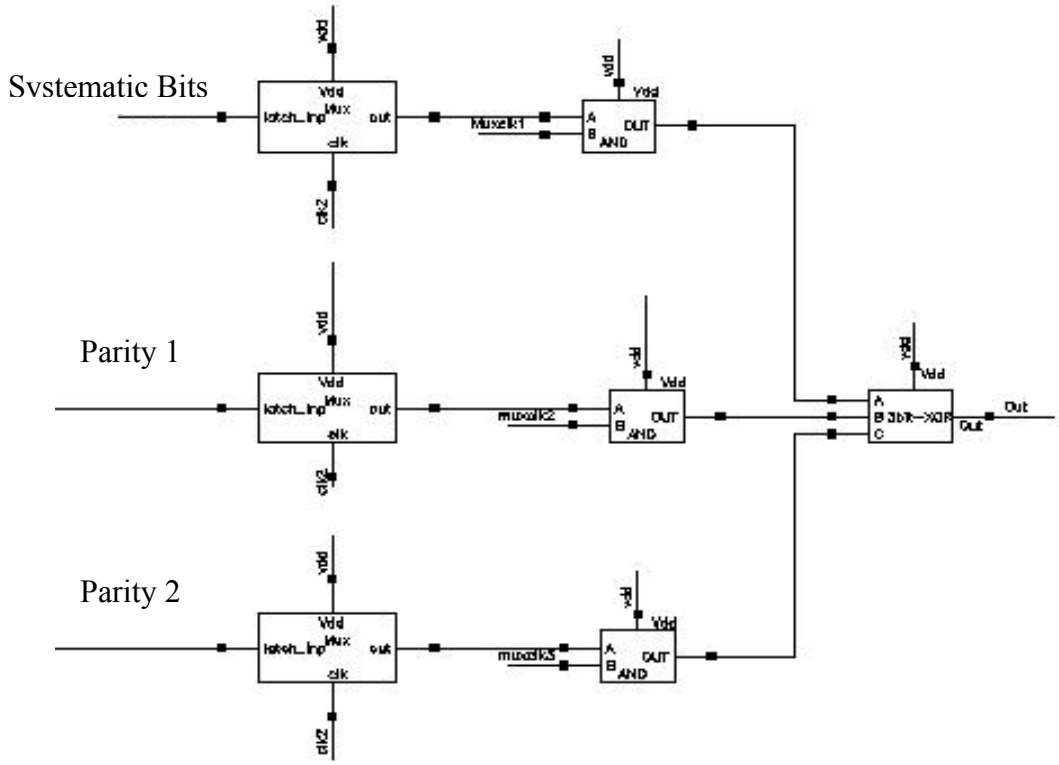



FIGURE 7: Schematic of the puncturer

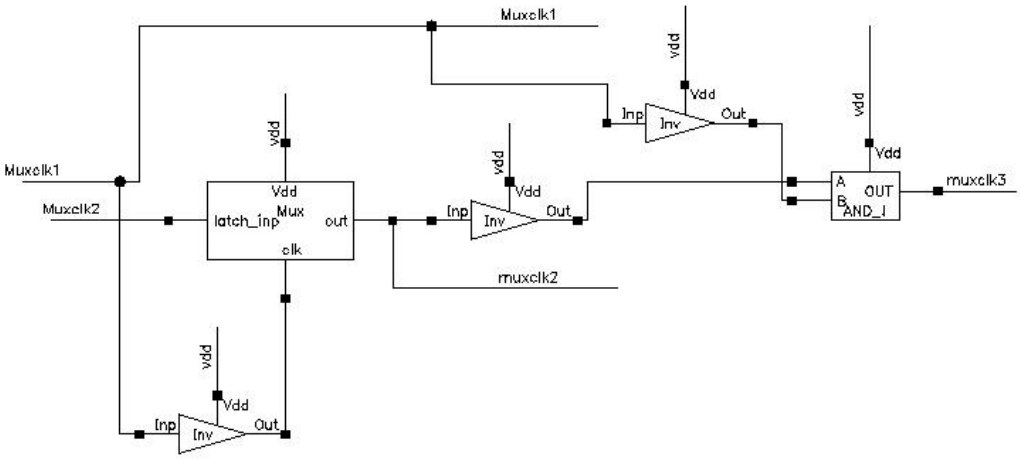



FIGURE 8: Circuitry to internally generate MUXCLK3 from MUXCLK1 and MUXCLK2

## 3.5 Chip Floor Plan

The modules were put together with an objective to minimize the chip area. The orientations of the two 32-bit input buffers were flipped to share the clocks. The chip floor plan is shown in figure 9 and the chip layout is attached as appendix C.

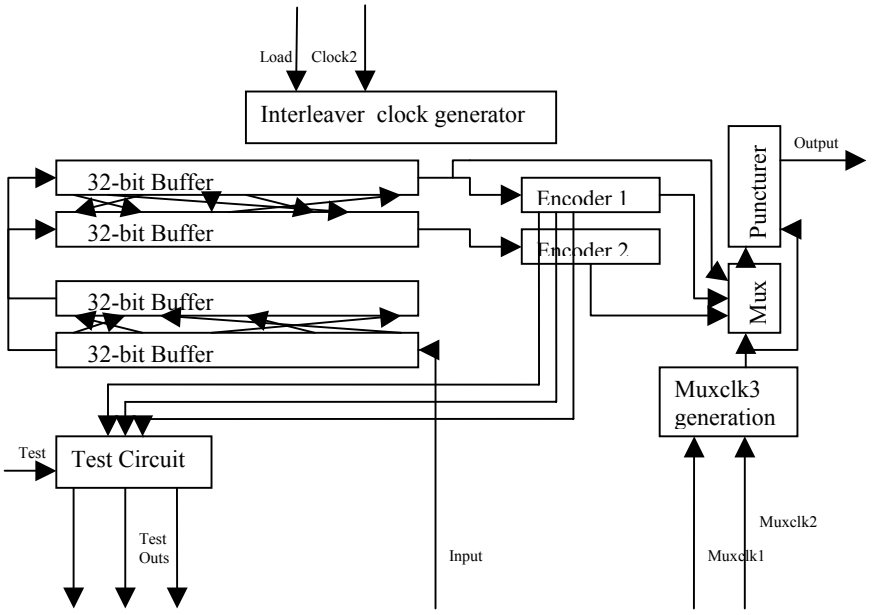



FIGURE 9: Chip floor plan

# 4.0 Application Notes

The AMV 6323 can be used in any typical wireless device. At the input, it takes a digital bit stream (1s and 0s) and outputs an encoded digital bit stream. A typical application in a digital cell-phone is shown in figure 10. The voice codec converts the voice signals to a digital bit stream. The DSP takes the output of the codec and processes it to reduce echo and other noise. Depending on the QoS requirements of the data, the DSP sends a signal to the code rate selector specifying the code rate to be used. It also splits the data into the specified block size (64 bits in this project) and sends it to the turbo encoder. The turbo encoder encodes this stream and punctures it depending on the signal (puncturing clock) it receives from the code rate selector. This encoded bit stream is sent to the RF codec for RF processing. The QPSK modulator converts the RF output to sine and cosine signals which are then sent to the RF section for up-conversion and transmission.

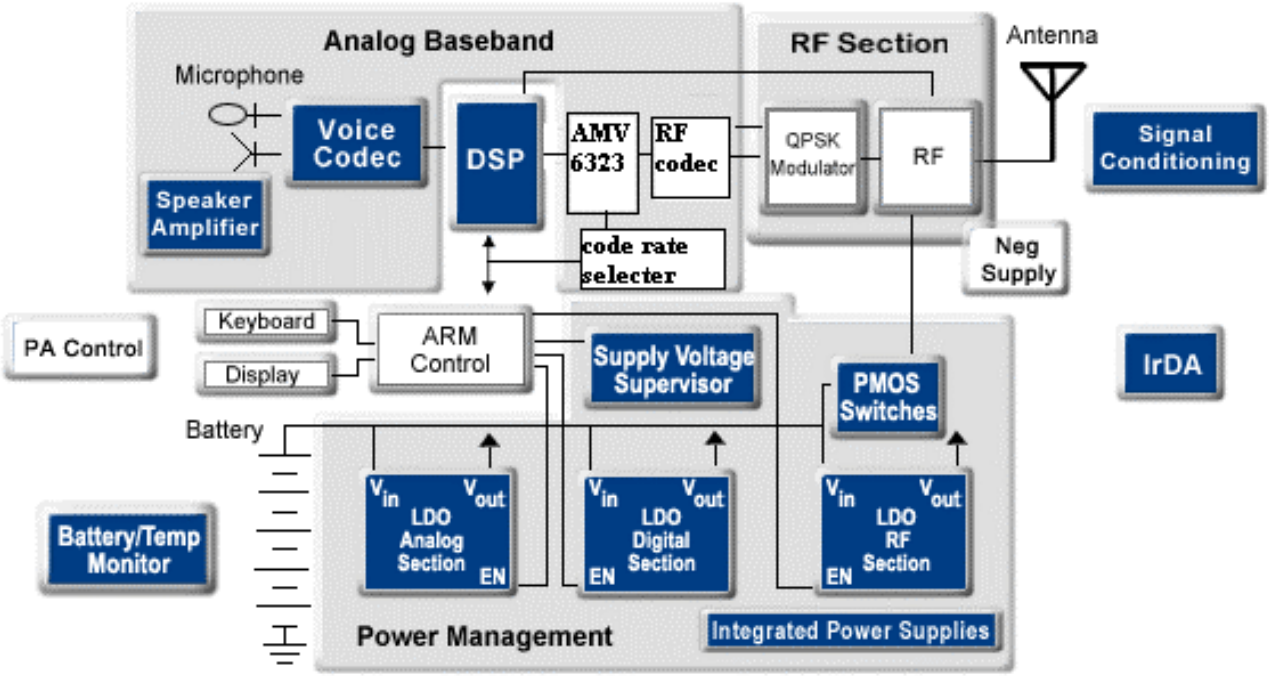



FIGURE 10: A typical application of the AMV 6323 in a digital cell phone transmitter

Since the AMV 6323 takes purely digital inputs it is compatible with any device that produces a digital bit stream. Since there are a wide variety of ICs that are used to process and output digital signals in wireless devices, the AMV 6323 can be interfaced with all these devices, thus increasing the versatility of the device.

## 4.1 Turbo Code Selection

The feedback and feed-forward polynomials for the constituent codes can be chosen using the select lines S1-S7 (for encoder 1) and S12-S72 (for encoder 2). The select pins are shown in the figure 3. The block diagram of encoder 1 with the select pins identified is shown in figure 11.

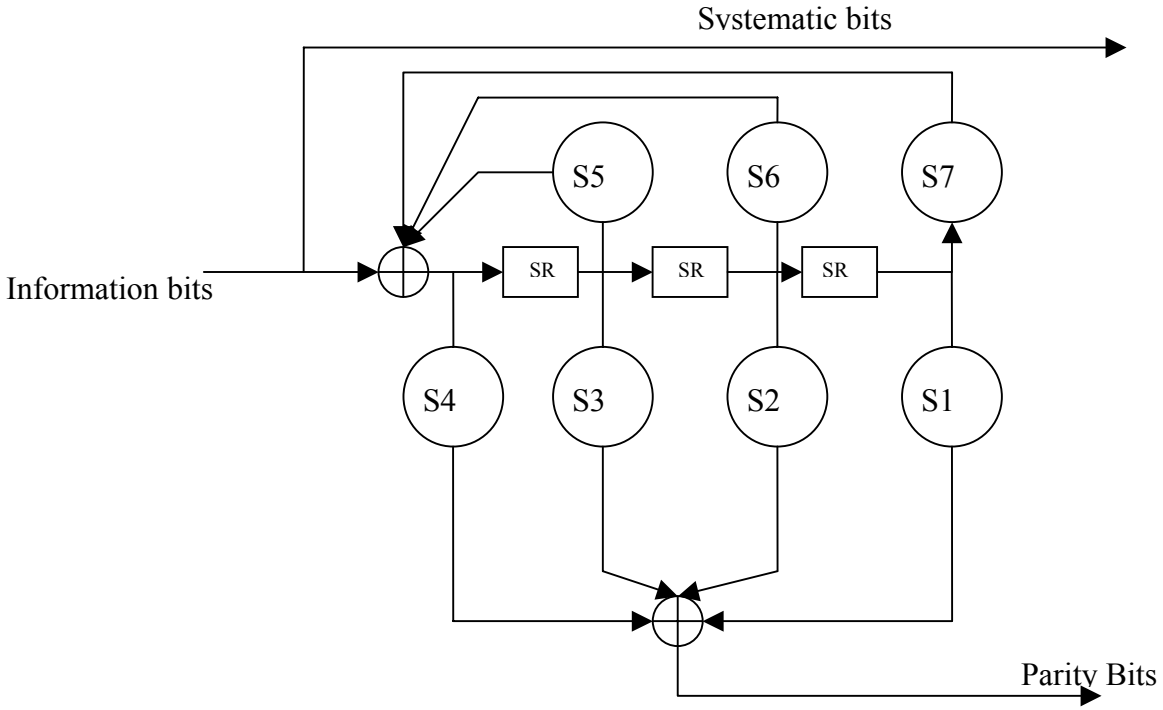



FIGURE 11: Encoder 1 shown with the feedback and feed-forward select lines

Encoder 2 is identical to encoder 1. The pin 'Si' in encoder 1 is labeled 'Si2' in encoder 2. As shown in figure 11, pins S1-S4 and S12-S42 select the feed-forward polynomials for encoder 1 and encoder 2 respectively. Similarly, pins S5-S7 and S52- S72 select the feedback polynomials for both the encoders. Table 1 shows the logic levels of the select pins for a few generator (feedback and feed-forward) polynomials. Table 2 shows the pin configuration of the IC.

| Generator polynomial (feedback, feed-forward) | Feed-forward Select Pins | | | | Feedback Select Pins | | |
|---|---|---|---|---|---|---|---|
| | S4/S42 | S3/S32 | S2/S22 | S1/S12 | S5/S52 | S6/S62 | S7/S72 |
| (13,15) | 1 | 1 | 0 | 1 | 0 | 1 | 1 |
| (17,12) | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| (3,12) -(4 state) | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| (16,11) | 1 | 0 | 0 | 1 | 1 | 1 | 0 |

TABLE 1: Generator polynomial selector

| PIN NO. | PIN | FUNCTION |
|---|---|---|
| 1 | NC | No connection |
| 2 | TEST | Test Enable |
| 3 | OUT 3 | Contents of Shift register 2 of encoder 1 during testing |
| 4 | OUT2 | Contents of Shift register 1 of encoder 1 during testing |
| 5 | OUT1 | Contents of Shift register 0 of encoder 1 during testing |
| 6 | GND | Ground pin |
| 7 | MUXCLK2 | Clock selection parity bit from encoder 1 at output |
| 8 | MUXCLK1 | Clock selecting systematic bits at output |
| 9 | INPUT | Input bit stream |
| 10 | S72 | Feedback polynomial selector for encoder 2 |
| 11 | S62 | Feedback polynomial selector for encoder 2 |
| 12 | S52 | Feedback polynomial selector for encoder 2 |
| 13 | S42 | Feed-forward polynomial selector for encoder 2 |
| 14 | S32 | Feed-forward polynomial selector for encoder 2 |
| 15 | S22 | Feed-forward polynomial selector for encoder 2 |

TABLE 2: Pin Configuration of the AMV 6323

| | | |
|---|---|---|
| 16 | S12 | Feed-forward polynomial selector for encoder 2 |
| 17 | OUTPUT | Serial output of encoded bits |
| 18 | S1 | Feed-forward polynomial selector for encoder 1 |
| 19 | S2 | Feed-forward polynomial selector for encoder 1 |
| 20 | S3 | Feed-forward polynomial selector for encoder 1 |
| 21 | S4 | Feed-forward polynomial selector for encoder 1 |
| 22 | S5 | Feedback polynomial selector for encoder 1 |
| 23 | S6 | Feedback polynomial selector for encoder 1 |
| 24 | S7 | Feedback polynomial selector for encoder 1 |
| 25 | LOAD | Load clock to load contents of input buffer into encoder |
| 26 | NC | No Connection |
| 27 | NC | No Connection |
| 28 | CLK1 | Clock input to the chip |
| 29 | CLK2 | Clock input to the chip |
| 30 | Vdd | Supply Voltage |

TABLE 2 (contd) : Pin Configuration of the AMV 6323

A sample configuration for the (13,15) turbo code using the AMV 6323 is shown below.

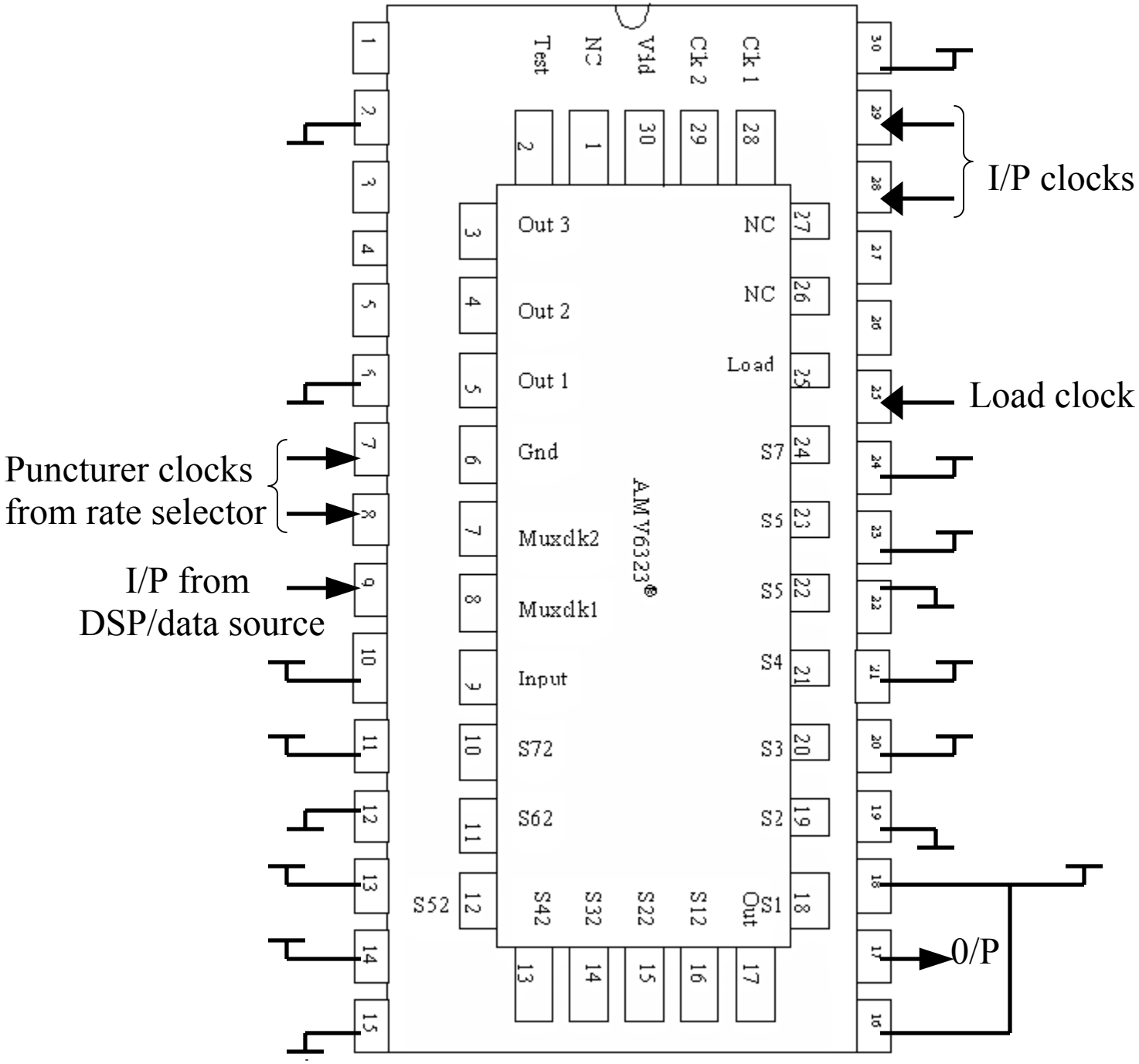



FIGURE 12: Chip configuration to implement the (13, 15) 3GPP turbo code.

As observed, most turbo codes have identical constituent encoders. But recently researchers have started investigating irregular/asymmetric turbo codes which have different encoders. This is the reason why both the encoders' select lines were made programmable. Thus, the AMV 6323 supports both symmetric and asymmetric turbo codes.

## 4.2 Design Specifications

- AMV 6323: 64-bit variable-rate, programmable Turbo encoder.
- Design Process: Hewlett Packard (HP) AMOS14TD process - 0.5µ feature size
- Number of pins: 30
- Cost per Chip: 95.71 cents
- Area of Chip (without pads): 895 µm x 445 µm
- Area of Chip (with pads): 1.52 mm x 1.068 mm
- Number of transistors (not including pads): 3868
- Number of transistors (including pads): 5052
- Input/Output bus width: 1 bit (bit stream input and output)
- Select (lines) bus width: 7 bits per encoder
- Maximum clock frequency: 12.8 MHz
- Supply Voltage: 5V
- Operating temperature range: 0-80 $^{o}$C

FEATURES:

- Capable of operating at 2 code rates (one-half and one-thirds)
- Programmable constituent encoders
- Capable of on-the-fly reconfiguration of the constituent encoders
- Capable of operating at reduced Vdd =3V
- Capable of achieving throughput of 12.8Mbps @ 12.8MHz clock.
- Low cost of 95.71 cents per chip

# 5.0 Test capability

## *Built in self-tests provide a significant reduction in testing costs!*

The AMV 6323 has a built-in test capability. The performance of a turbo code depends on the proper termination (force back to all zero state) of one of the constituent encoders. Terminating both the encoders provides a negligible performance improvement and is generally not done. The AMV 6323 has a test capability to check the contents of the shift registers of encoder 1. This enables the user to test if the states of the shift register are changing according to the turbo encoder he has programmed into the chip and if the encoder is being terminated properly.

Pin 2 of the AMV 6323 is a test enable pin. This pin should be grounded when the IC is being used in an application and should be connected to $V_{dd}$ only during testing. During testing, the contents of the shift registers 0, 1 and 2 of encoder 1 are shorted to pins 5, 4 and 3 respectively. This enables the tester to feed in a known information sequence and check if the states are changing according to the input sequence. Ensuring that the states are changing properly is enough to ensure the parities are being generated correctly. Also observing the states enables the tester to check if the code is being terminated properly i.e. returning to the all zero state, three clock cycles after the last bit has been fed in.

The test configuration is shown in the diagram below. Having an in-built test capability decreases the time on the tester and hence goes towards decreasing the overall cost of the chip.

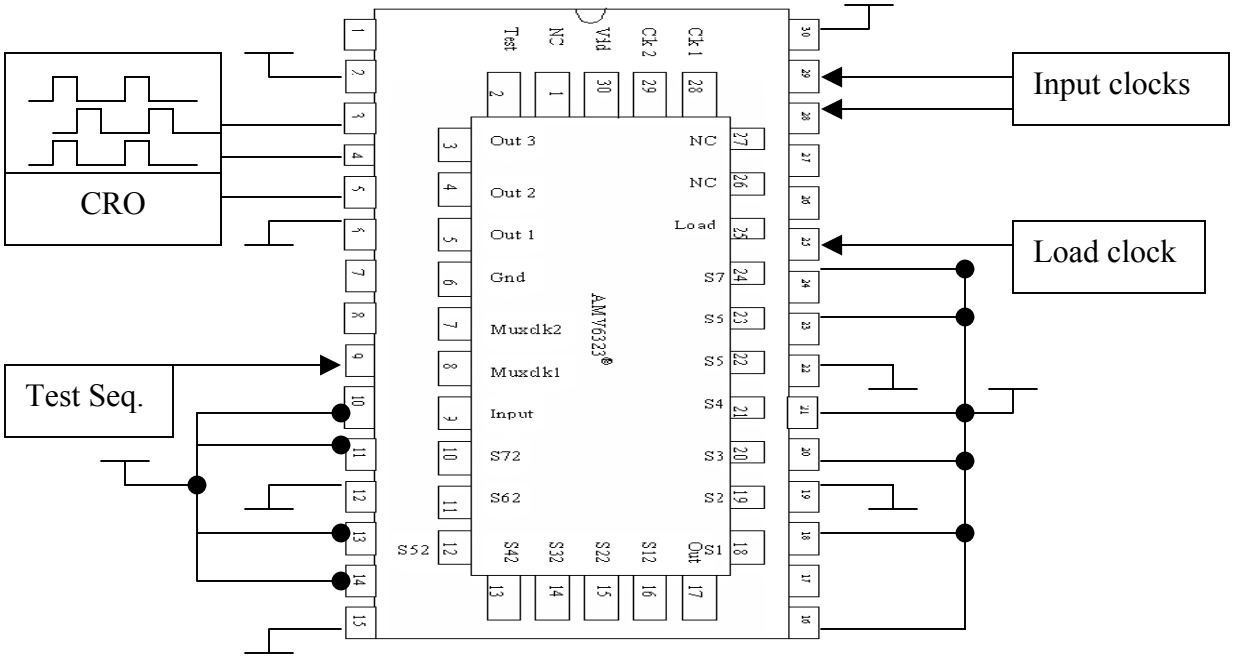



FIGURE 13: Test configuration for the (13,15) turbo code

# 6.0 Future Directions

The Turbo Encoder implemented in this project was implemented in its simplest form. Given another semester we would improve our project by considering some of the following modifications:

- Turbo codes offer best performance when the block size is of the order of 1000 bits. The current implementation can be improved by increasing the size of the block from 64 to 1024 bits. Supporting multiple block sizes could be considered to increase the versatility of the AMV 6323.

- The interleaver used in this project is a Random (pseudo-random) interleaver. The interleaver design and efficiency could be improved by implementing other types of interleavers like Circular-Shifting interleavers or Odd-Even interleavers. Algorithmic interleavers could also be considered

  Circular-Shifting interleavers offer very good performance for short block lengths (about 300 bits). Hence, they can be used for voice applications over a wireless environment.

- A PLL could be added to minimize clock skew.

- Implementation with a smaller number of clocks could be considered by implementing logic to generate some of the other clocks on chip.

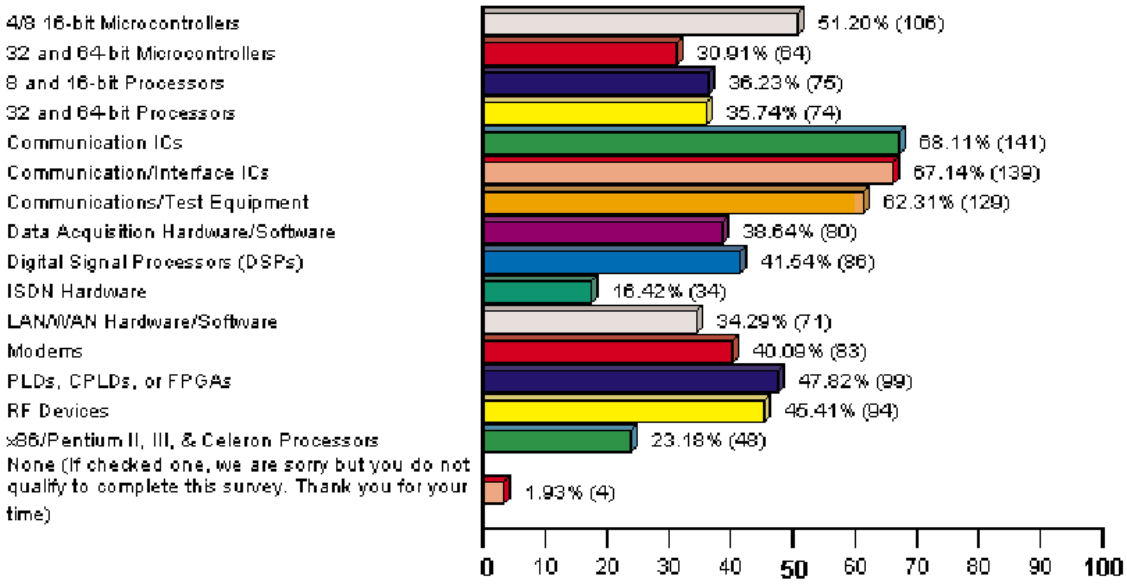
- Test time could be reduced by providing a separate test input to feed the shift register directly. This would get rid of the latency involved in the flow of data through the 64 bit input buffer.

- Circuit could be optimized to achieve higher throughput.

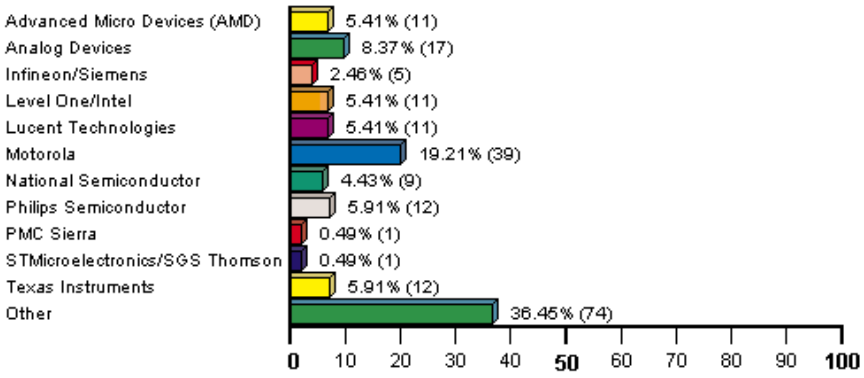
# 7.0 Summary

In order to evaluate the marketability of this IC [2], the following results of a 2000 survey would be in order.

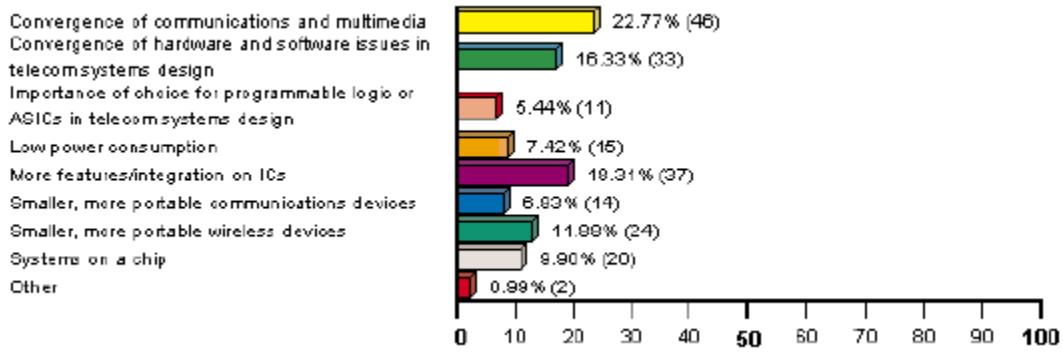
The response of managers for the communication ICs they would recommend, specify, evaluate or authorize the purchase of is shown below. It is seen that communication and communication interface ICs were suggested most often.



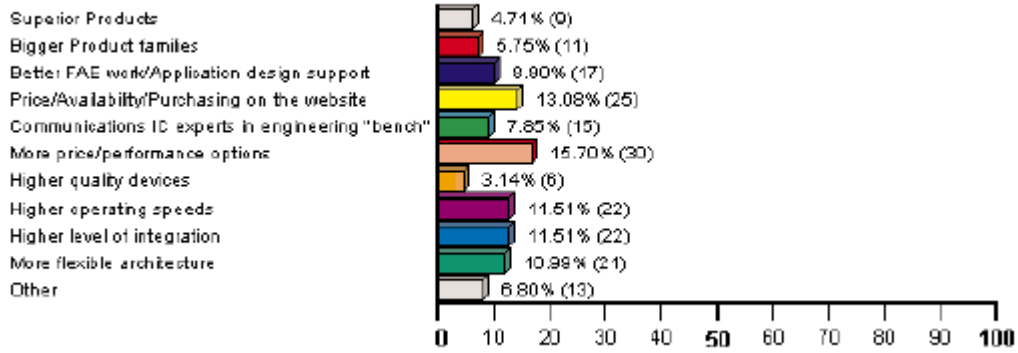

The response for preferred primary vendor of communication ICs for their current design is given below. It is seen that there is no strong favorite and that the major share is taken by other smaller companies and not by any of the IC giants. Hence, there is tremendous potential for a small startup in this area.




When asked about the most important technological trend in the communications market today, the majority of the votes went to the integration of communications and multimedia. Hence, an IC that would address this integration directly would be very marketable.




When asked about which unmet demand the vendors of today fail to meet, the most common response was that of price/performance options. Since the AMV 6323 is marketed over a wide customer base, there is an increase in profit and hence a decrease in cost.




In short, a successful communications IC that is introduced should have the following key characteristics:

- It should address the integration of voice and multimedia.

- It should have good price/performance options.

The AMV 6323 tackles the first issue by providing different rates (levels of protection) for different classes of traffic. Making the chip programmable attacks the second issue of cost. This ensures a wider market base and hence delivers an IC that implements the most powerful FEC technique known today at a very low cost. The AMV is also capable of operating at a lower power supply of 3V leading to lower power consumption.

Though the AMV 6323 is targeted towards mobiles, its application is not restricted to wireless systems. It can be used in any system that uses turbo coding. This includes both optical and powerline communication systems.

Thus, the AMV 6323 is a chip that directly addresses the integration of voice and multimedia and its versatility leads to the chip being very marketable. The chip also offers a good price-performance ratio without compromising functionality.

# References

[1] " A Turbo code tutorial", William E.Ryan. Available online.
http://www.ece.arizona.edu/~ryan/

[2] Electronic design media's survey on communications ICs. Available online.
http://www.edmediafile.com/docs/communications.pdf

[3] http://www.3gpp.org, The Official 3gpp website

[4] "Low-Power CMOS VLSI Circuit Design", K. Roy and S. C. Prasad. Wiley-Interscience.

[5] "Principles of CMOS VLSI Design", N. H. E. Weste and K. Eshraghain, 2nd Edition, Addison-Wesley.

[6] " EEL 6323 – Class Notes", William R. Eisenstadt. Spring 2002.

# Appendix A: Yield Calculations

Number of Pins = **30 pins**

$A_d$ = 1517.4 x $10^{-6}$ +1067.701 x $10^{-6}$ $m^2$ = **1.62 x $10^{-6}$ $m^2$**

Given, Wafer Radius r = 8 inches = 8 * 0.0254 = **0.2032 m**

$\therefore$ Wafer Area = $\pi r^2 = \pi (0.0762)^2$ = **0.1297 $m^2$**

$$\text{Number of dies} = \frac{\text{Wafer Area}}{\text{Die Area}} = \frac{0.1297}{1.62\text{x}10^{-6}} = 80{,}072$$

The defect density was assumed to be, $D_o$=1.5x$10^{-4}$ $m^{-2}$

Using the Murphy's model for yield,

$$\text{Yield} = \left[\frac{1-e^{-D_0 A_d}}{D_0 A_d}\right]^2 = 97.6\%$$

Assuming 95% of the dies to be good,

Number of working chips = 0.976 x 0.95 x 80072 = **74242 chips**

Wafer cost = $800

$$\therefore \text{cost per chip} = \frac{\text{Wafer cost}}{\#\text{ of working chips}} = \frac{800}{74242} = 0.011\,\textbf{cents}$$

Additional cost, Testing cost per die = $0.06

Cost of pins = $0.02 x 30 = $0.60

$\therefore$ Total cost per chip =0.011+6+60=**66.01 cents**

Adding a 45% profit, We get the total selling cost per chip = 66.01 x 1.45= **95.71 cents**

**SELLING COST /CHIP = 95.71 cents**

# Appendix B: LVS Output

## With Pads

```
@(#)$CDS: LVS version 4.4.3 05/11/2000 21:10 (cds230) $

Like matching is enabled.
Net swapping is enabled.
Using terminal names as correspondence points.
Compiling Diva LVS rules...

    Net-list summary for /home/users/0/venigalr/LVS/layout/netlist
       count
      1806        nets
      0           terminals
      2526        pmos
      2526        nmos
    Net-list summary for /home/users/0/venigalr/LVS/schematic/netlist
       count
      1806        nets
      11          terminals
      2030        pmos
      2030        nmos

The net-lists match.
                        layout  schematic
                         instances
      un-matched         0     0
      rewired            0     0
      size errors        0     0
      pruned             0     0
      active             5052  4060
      total              5052  4060

                           nets
      un-matched         0     0
      merged             0     0
      pruned             0     0
      active             1806  1806
      total              1806  1806

                         terminals
      un-matched         0     0
      matched but
      different type     0     0
      total              0     11
```

## Without Pads

```
@(#)$CDS: LVS version 4.4.3 05/11/2000 21:10 (cds230) $

Like matching is enabled.
Net swapping is enabled.
Using terminal names as correspondence points.
Compiling Diva LVS rules...

    Net-list summary for /home/users/0/venigalr/LVS/layout/netlist
       count
      1743        nets
      0           terminals
      1934        pmos
      1934        nmos

    Net-list summary for /home/users/0/venigalr/LVS/schematic/netlist
       count
      1743        nets
      27          terminals
      1934        pmos
      1934        nmos

The net-lists match.

                       layout  schematic
                        instances
      un-matched        0     0
      rewired           0     0
      size errors       0     0
      pruned            0     0
      active            3868  3868
      total             3868  3868

                          nets
      un-matched        0     0
      merged            0     0
      pruned            0     0
      active            1743  1743
      total             1743  1743

                        terminals
      un-matched        0     0
      matched but
      different type    0     0
      total             0     27
```

# Appendix C: Chip Layout

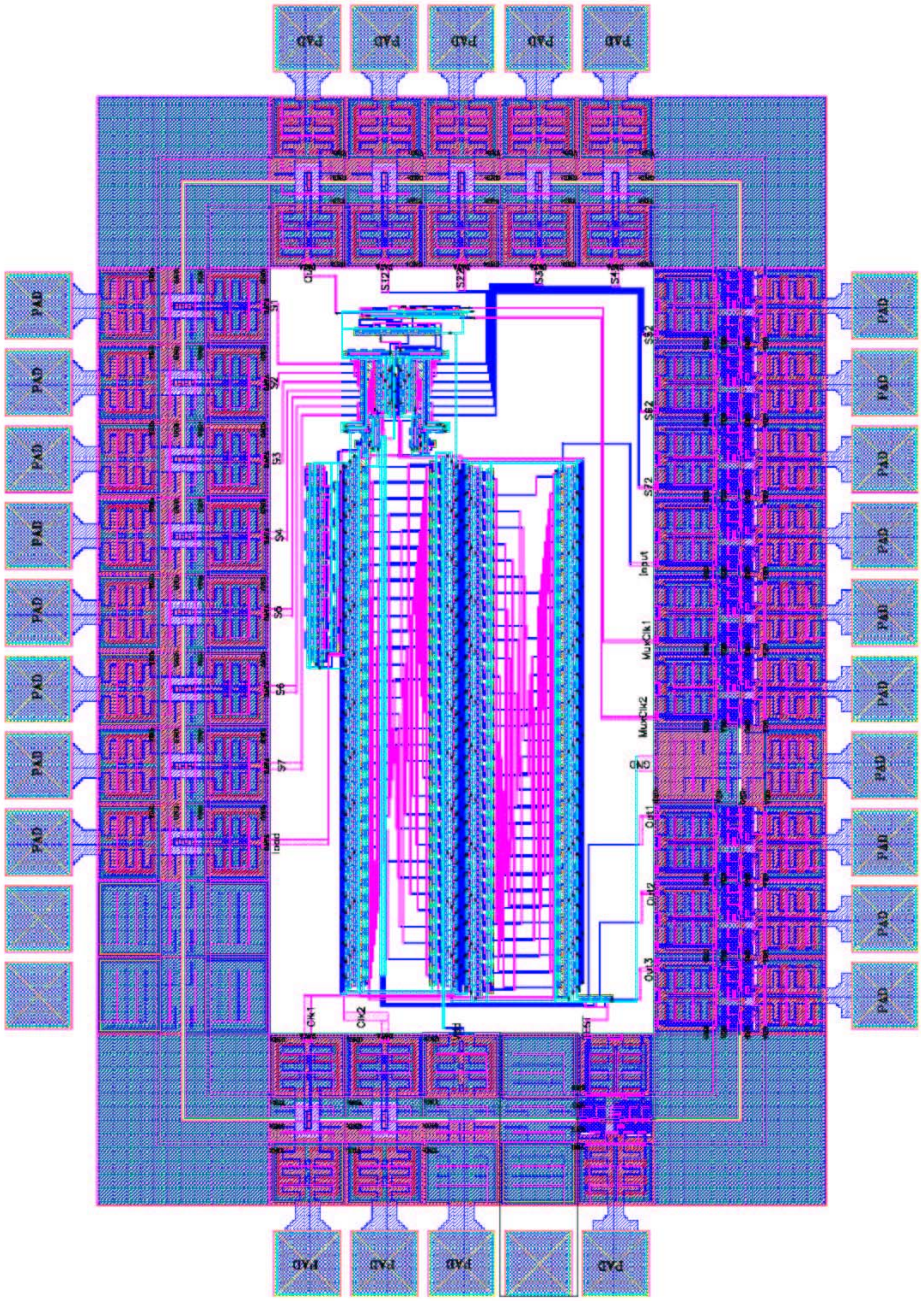


FIGURE 14: Chip Layout

# Appendix D: Simulation Results

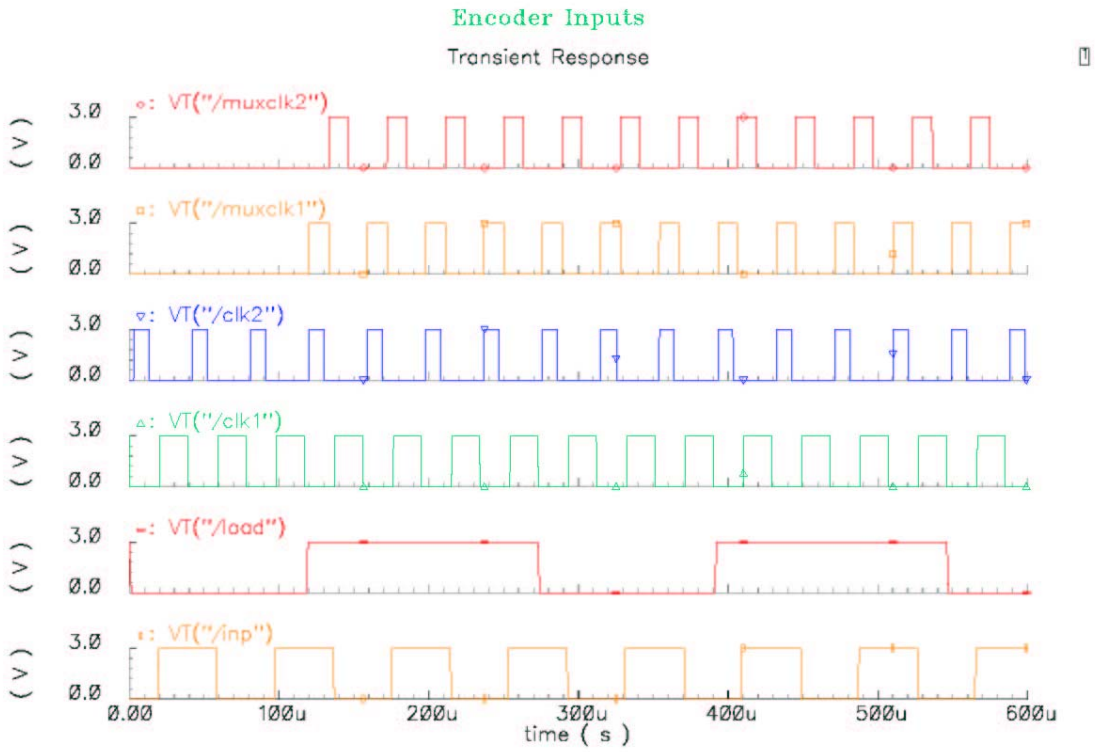



FIGURE 15: Clock inputs to the Turbo Encoder Chip

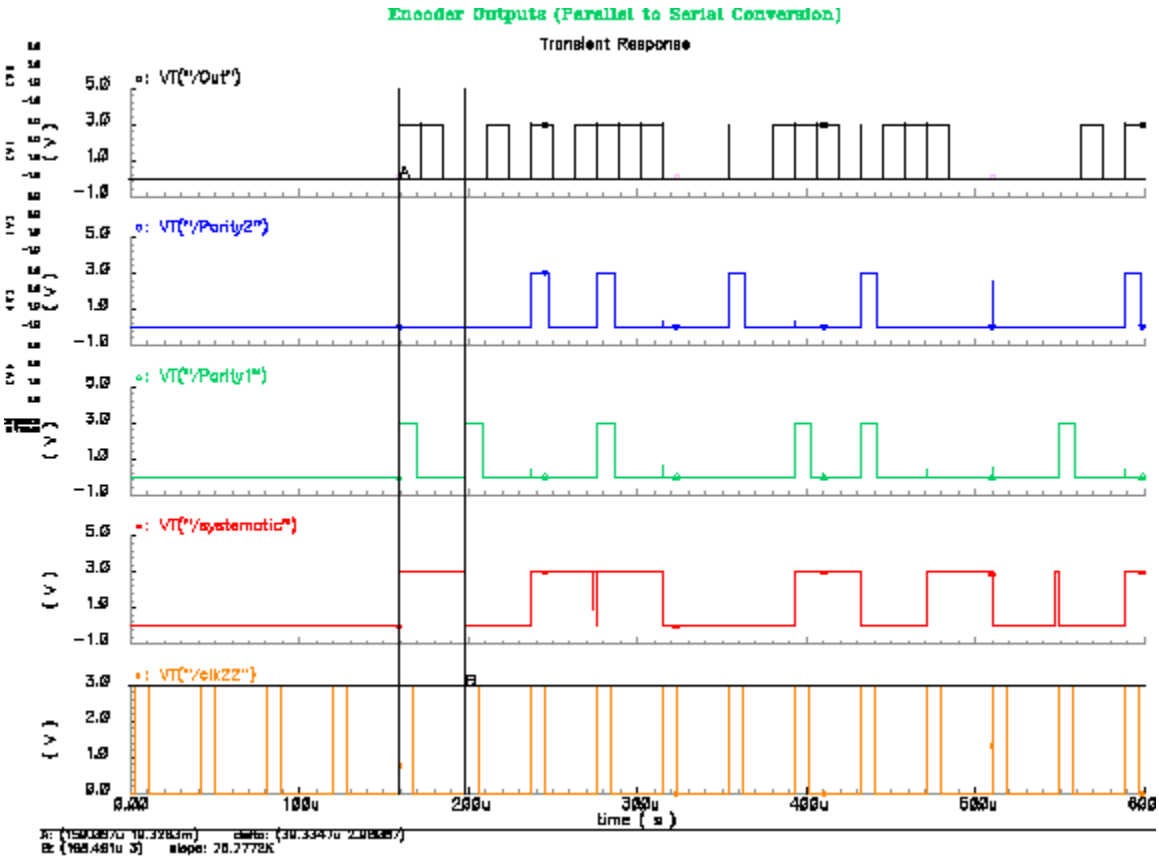



FIGURE 16: Output of the Turbo Encoder Chip

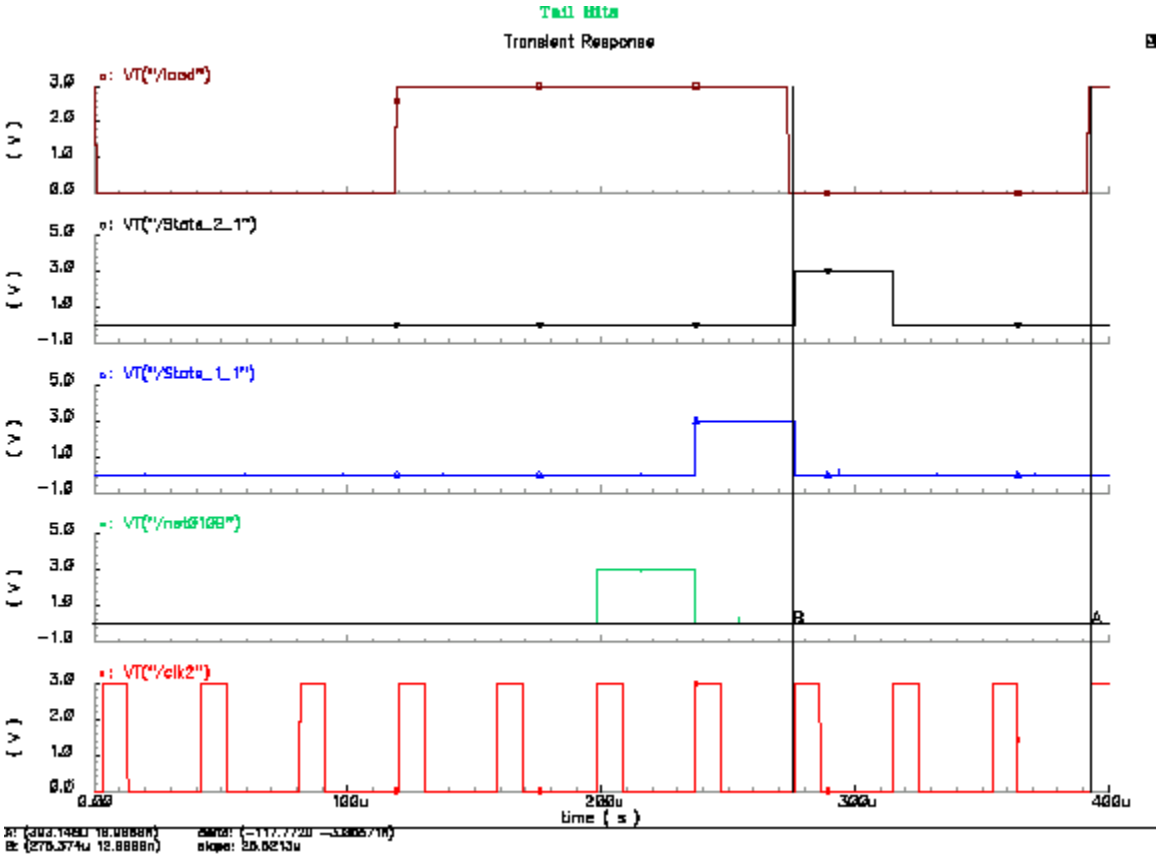



FIGURE 17: Trellis Termination of the constituent encoder 1