Journal of Theoretical and Applied Information Technology
31st December 2020. Vol.98. No 24
© 2005 – ongoing JATIT & LLS

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

LOW POWER DESIGN OF ADAPTIVE TURBO DECODER WITH FEEDFORWARD MESSAGE DECODING AND LOOK-UP-TABLE APPROACH

[1]GIRISH N, [2]VEENA M B

[1]Research Scholar, Department of Electronics and Communication, BMS College of Engineering, Visvesvaraya Technological University, India

[2]Associate Professor, Department of Electronics and Communication, BMS College of Engineering, Visvesvaraya Technological University, India

E-Mail: [1]giribms17@gmail.com, [2]veenamb.ece@bmsce.ac.in

ABSTRACT

Underwater acoustic communication networks (UWAN) have recently attracted much attention in the research community. Error control coding one of the primary building blocks in underwater MODEMs plays a vital role in achieving better Bit Error Rate (BER) performances. In this paper, Adaptive Turbo Decoder (ATD) algorithm is designed by considering Adaptive Viterbi Decoder (AVD) customized for underwater acoustic MODEM applications. The ATD module is designed with four parallel processing architectures to perform message decoding, multipath computing, branch metric unit (BMU), and Add-Compare-Select (ACS) operations. The designed ATD is modelled in MATLAB and evaluated for its BER performance considering Rayleigh and Rician channels demonstrating improvement in BER over 10^{-4}. The ATD module is further coded in Verilog HDL and verified for its functionality considering different test-vectors and implemented on Virtex FPGA optimizing Area, Power, and Timing requirements.

Keywords: *Turbo Decoder, Underwater Communication, Viterbi Decoder, Adaptive Algorithm, FPGA Implementation*

1. INTRODUCTION

Underwater communication in recent years is in greater demand with applications such as oil field monitoring, data collection from the ocean, maritime archaeology, disaster prevention, coastline security, and fishing activity. Deployment of wireless nodes that operates on acoustic signals in an underwater environment helps in data gathering and communication between them. Sensor nodes based on acoustic signal support an operating range of 5-10 kilometres with the trade-off being low data rates. The underwater environment is a complex channel having a multipath environment and limited bandwidth due to time-varying, space varying, and frequency selective characteristics [1]. Acoustic signals travel at slow speed (approximately less than 1500 m/s) in water medium limiting transmitting speed of data and delay in delivery. Acoustic signals get attenuated due to absorption in water and signals get transformed due to heterogeneous characteristics as well as Doppler effects that are caused due to movement between transmitter and receiver. Underwater channel exhibit random characteristics due to seasonal changes, geographical variation in temperature and water salinity, seabed relief, currents, tides, and internal waves caused due to movement of objects. The primary building blocks in the acoustic modem are the modulator and channel coder. The use of Orthogonal Frequency Division Multiplexing (OFDM) achieves high data rates, it is required to carefully analyze the appropriate modulation scheme and error coding scheme for establishing a successful communication link in underwater environments.

To achieve Bit Error Rate (BER) nearer to 10^{-4} suitable error coding techniques have been used that combines Reed Solomon (RS) Code with Block Turbo Code (BTC) [2-3] and is termed as

RS-BTC code. Decoding of information from the received signal was carried out based on Chase-Pyndiah [4-5] algorithm and error correction was carried out using the Berlekamp-Chen algorithm [6]. Comparison of convolutional codes, RS codes with RS-BTC is carried out by Goalic et al. [7]. To improve BER from 10^{-2} to 10^{-4} in convolution coding the number of iterations in the decoder was set to 16. BER of 10^{-4} was achieved by using three iterations in RS-BTC coding. The limitations of RS-BTC are that it occupies a long data packet and is difficult to set constraints during real-time analysis. Adaptive transmission schemes that use adaptive modulation techniques and coding techniques according to channel conditions are reported [8-10] achieving better BER and data rates. The improvement in the BER performance of the OFDM system using LDPC and Turbo codes for different channel models like AWGN, Rayleigh flat, and frequency selective fading were carried by Sameer A. Dawood et al. [11]. The results indicated that the use of channel coding in the OFDM system can offer a higher data rate. BER Performance using Rician and Rayleigh channel model of a precoded OFDM system were for varying Doppler shifts proposed by Padmanabhan et al. [12] show throughput can be increased by precoded OFDM system.

Adaptive OFDM based underwater communication modem is implemented on a hardware platform [13] and is evaluated for its performances considering the QPSK modulation scheme demonstrating improvement in BER. Weiwei Kan et al. [14] have developed an OFDM system with (2, 1, 7) convolutional code and Viterbi decoder to minimize BER and to overcome phase error. Interleaving of error coded data is carried out to minimize continuous errors to random errors. QPSK and 16 QAM modulation schemes are used in this system demonstrating BER of 10^{-3} at 15 dB SNR. OFDM algorithm is developed using Dynamic Coded Cooperation (DCC) scheme for underwater relay networks [15] with two different error coding schemes: Low-Density Parity Check (LDPC) and Layered Interblock Erasure Correction Codes. QPSK modulation and 1024 FFT processor is used for symbol mapping and subcarrier modulation, respectively. Packet error rate (PER) of 10^{-3} is observed in LDPC coding and Block Error Rate of 0.05 is observed with the use of Erasure correction scheme. LDPC code is demonstrated to be superior to turbo codes for speech data transmission over an underwater environment [16]. LDPC coding is modified further to match the properties of modulated symbols demonstrating improvement in BER in multicarrier underwater systems [17]. LDPC coding for underwater communication has also been demonstrated to achieve improvement in BER with lower SNR [18] which is achieved by suitably selecting encoding and decoding parameters matching the underwater acoustic channel requirements. An underwater communication system that can process the weak signal in a low-frequency acoustic band is designed by Michel Barbeau et al. [19]. Frequency shift keying modulation scheme, synchronization methods have been used along with convolutional codes for channel coding with rate ½ and constraint length of 32. At the receiver Fano [20] decoding algorithm is used which is a sequential decoding algorithm in place of Viterbi decoder to achieve a superior decoding rate of 0.22 bits per time unit as compared with 1 bit/time units of Viterbi decoder. Convolutional encoded error coding scheme with Viterbi decoder is used with Frequency shift keying modulation scheme [21] demonstrating PER less than 10^{-3} at 10dB Eb/No. The watermark model is used for channel modelling and the performances are compared with JANUS implementation. Studies on turbo coding methods for underwater communication is reported in the literature demonstrating advantages over RS encoding and Convolutional encoding techniques. With the underwater channel being dynamic in nature, most of the error coding methods are not able to achieve minimum BER results.

In this paper, adaptive turbo decoder is proposed and is evaluated to achieve better and consistent BER considering dynamic properties of the underwater channel, Section 2 discusses in detail the underwater channel and its properties, Section 3 presents turbo encoder and decoder algorithm, Section 4 presents the architecture of Viterbi decoder, Section 5 presents adaptive Viterbi decoder for turbo decoding and results are presented in section 6.

2. UNDERWATER CHANNELS

Absorption loss, signal spreading loss increases with distance and is also dependent on operating frequency [22]. The path loss in acoustic signal is given as in Eq. (1), is expressed in terms of absorption coefficient a(f), signal frequency f, transmission distance l, reference distance l_r, and path loss constant k_a. Absorption coefficient expressed in terms of dB/km increases with an increase in acoustic signal frequency.

$$A(l,f) = (l/l_r)^{k_a} a(f)^{l-l_r} \qquad (1)$$

Ambient noise and site-specific noise are the two prominent noise sources in underwater channel that effects acoustic signal propagation. Site-specific noise may be due to ice breaking or shrimp snapping and is modelled as non-Gaussian noise. Ambient noise is due to turbulence, rain, and ship movement and is modelled as Gaussian noise. Noise power spectral density decreases at a rate of 18 dB/decade with an increase in frequency [23]. The Signal to Noise Ratio (SNR) is a function of signal bandwidth in an underwater channel environment and is expressed as in Eq. (2).

$$SNR(l,f) = S_l(f)/A(l,f)N(f) \qquad (2)$$

S_l(f) is the power spectral density of signal travelling in the channel. The acoustic signal bandwidth is a function of transmission distance. The bandwidth and power that is required to achieve a specific SNR over a specific distance are computed by Eq. (3). The parameters b, p, β, and Ψ depend upon path loss and ambient noise.

$$\left.\begin{aligned} B(l) &= b.l^{-\beta} \\ P(l) &= p.l^{\Psi} \end{aligned}\right\} \qquad (3)$$

At higher acoustic signal frequency, the bandwidth is limited to less than 1 kHz, for the lower frequency of operation around 10 kHz the bandwidth is around 12 kHz. Another important factor to be considered in an underwater environment is the multi hopping concept that achieves higher data rates and lower power consumption by placing multiple Trans- receiver modules at short distance locations between source and destinations [24]. Multipath occurs in the underwater channel due to sound reflection at the surface, at the bottom of any other objects. Sound travels at different speeds and is dependent on temperature, pressure, and water salinity. The underwater channel can be modelled as an impulse filter considering the reflection and refraction properties of sound waves and the number of significant propagation paths and signal strengths. The simplest model for underwater channel is expressed as in Eq. (4) that represents the frequency response of the sound path [25]

$$H_p(f) \frac{\Gamma_p}{\sqrt{A(l_p,f)}} \qquad (4)$$

The parameters Γp and A represents the reflection coefficients and propagation loss along pth path respectively. The propagation loss is a function of lp called the length of the path and frequency of signal 'f' [26]. Time variability of an underwater channel is also to be considered during underwater communication model design. Time variability in this channel is observed due to changes in propagation medium or changes in surface waves that shift the reflection point resulting in scattering of signal and Doppler spreading. It is a very complex process to model acoustic channels considering time variability, and the simplest acoustic channel model is based on Rayleigh or Rician channel models [27]. Doppler effect that occurs due to the motion of transmitter and receiver causes frequency shift and signal spreading. Doppler shift is measured as a ratio of relative transmitter-receiver velocity and sound speed. Doppler effect is not negligible in an underwater environment as there will always move in the transmitter-receiver modules due to movement in waves, currents, and tides. Research studies are active in identifying accurate methods for modelling Doppler effects in an underwater environment.

3. TURBO ENCODER AND DECODER

Turbo coding uses two convolutional coders with recursive structure [28-29] operated in parallel with an interleaver between encoders that generates two-bit streams is as shown in Figure 1. Two recursive systematic convolutional (RSC) codes are used to generate turbo codes. The advantage of the RSC encoder is that BER performances are better than any other systematic coding methods at a higher coding rate. The output of each of the storage units in RSC is recursively computed as in Eq. (5), where the input data bit is represented as D_k and g'_i is the generator code words.

$$a_k = d_k + \sum_{i=1}^{k-1} g'_i \, a_{k-1} \qquad (5)$$

generation of low-weight codewords [30]. The Channel model for underwater applications is considered as the Rician and Rayleigh model with three multipath signals and a doppler frequency shift of 50 Hz. The turbo encoder with a constraint length of K=3 and rate ½ is considered for the channel encoding scheme. The encoded data Y_k is transmitted over the underwater channel and at the receiver, the data with error Y'_k will be decoded by the Turbo decoder. At the receiver, a decoding

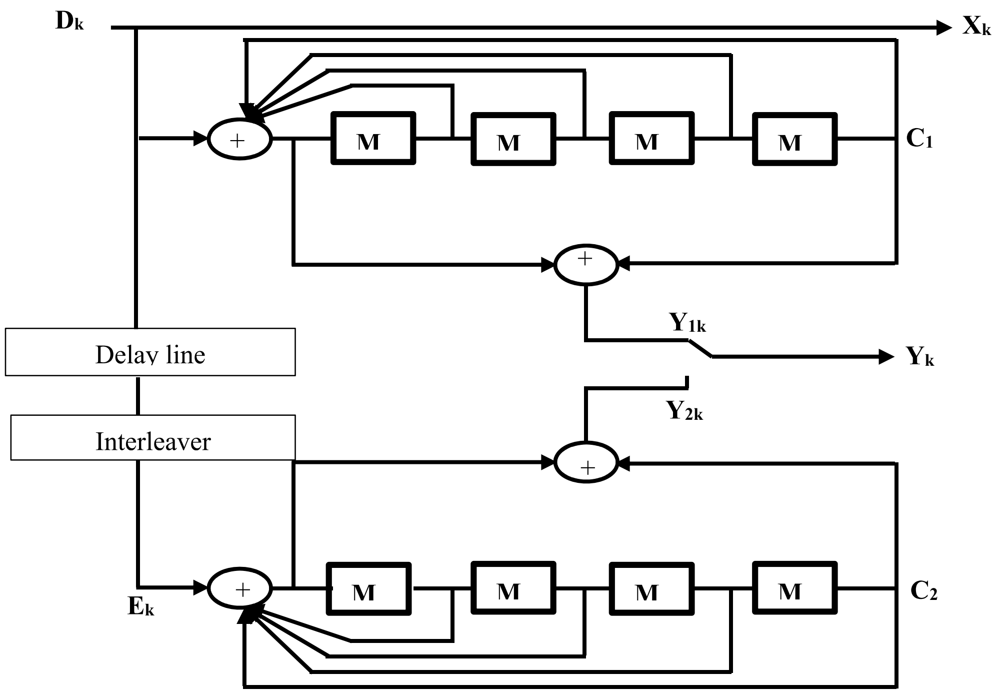


Figure 1: Turbo Encoder Using Convolutional Encoders And Interleaver

The input data D_k is processed by the recursive convolutional encoder C_1 to generate the encoded data sequence Y_{1K}. The data stream D_k is interleaved by random interleaver to generate data stream E_k which is encoded by the second encoder C_2 to generate data stream Y_{2k}. The encoded data stream X_k, Y_{1k}, and Y_{2k} are multiplexed to get encoder output denoted as Y_k. The interleaved logic can be designed to suit the channel noise errors. The data rate of convolutional encoders is computed as R = (n1 +n2)/(2n1 + n2), where n1 and n2 indicate the number of iterations required for encoding the data stream D_k and E_k respectively. The IIR property of RSC and the use of random interleaves protect the turbo encoder in the

algorithm [31] is selected to generate soft decision output. The soft decision output is derived considering joint probability $\lambda_k^{i,m}$ which is defined as in Eq. (6),

$$\lambda_k^{i,m} = P\{d_k = i, S_k = m | R_1^N\} \qquad (6)$$

The parameters S_k is the state of the encoder at any given time k, R_1^N is the binary data received during the time k=1 to N. The decoded data is represented as binary data and is expressed as in Eq. (7),

$$P\{d_k = i | R_1^N\} = \sum_m \lambda_k^{i,m} \; i = 0,1 \qquad (7)$$

Considering Eq. (7) the log-likelihood ration (LLR) is expressed as in Eq. (8),

$$L(\hat{d}_k) = log\left[\frac{\sum_m \lambda_k^{1,m}}{\sum_m \lambda_k^{0,m}}\right] \qquad (8)$$

RSC decoder decides the data bit based on the maximum a posteriori (MAP) decision rule by comparing Eq. (8) to a set threshold (threshold is basically set to zero), which is mathematically represented as in Eq. (9),

$$\left.\begin{aligned} \hat{d}_k &= 1 \; if \; L(\hat{d}k) > 0 \\ \hat{d}_k &= 0 \; if \; L(\hat{d}k) < 0 \end{aligned}\right\} \qquad (9)$$

Figure 2 presents the Turbo Decoder block diagram that decodes a set of input R_k those are represented as random variables X_k and Y_k. The decoder input Y_k which is a turbo encoded data sequence is redundant which is obtained from two RSC encoders at the transmitter. The data sequence Y_k is de-multiplexed and one set of data sequence (Y_{1k}) is processed by Decoder 1. Similarly, the de-multiplexed data Y_{2k} is processed by decoder 2. Decoder 1 decodes the information bit from the noisy data sequence which is interleaved by a similar interleaver that was used at the transmitter. The interleaved data output of decoder 1 is time-ordered as Y_{2k} data and both data sequences are processed by decoder 2 to generate the correct decoder output.

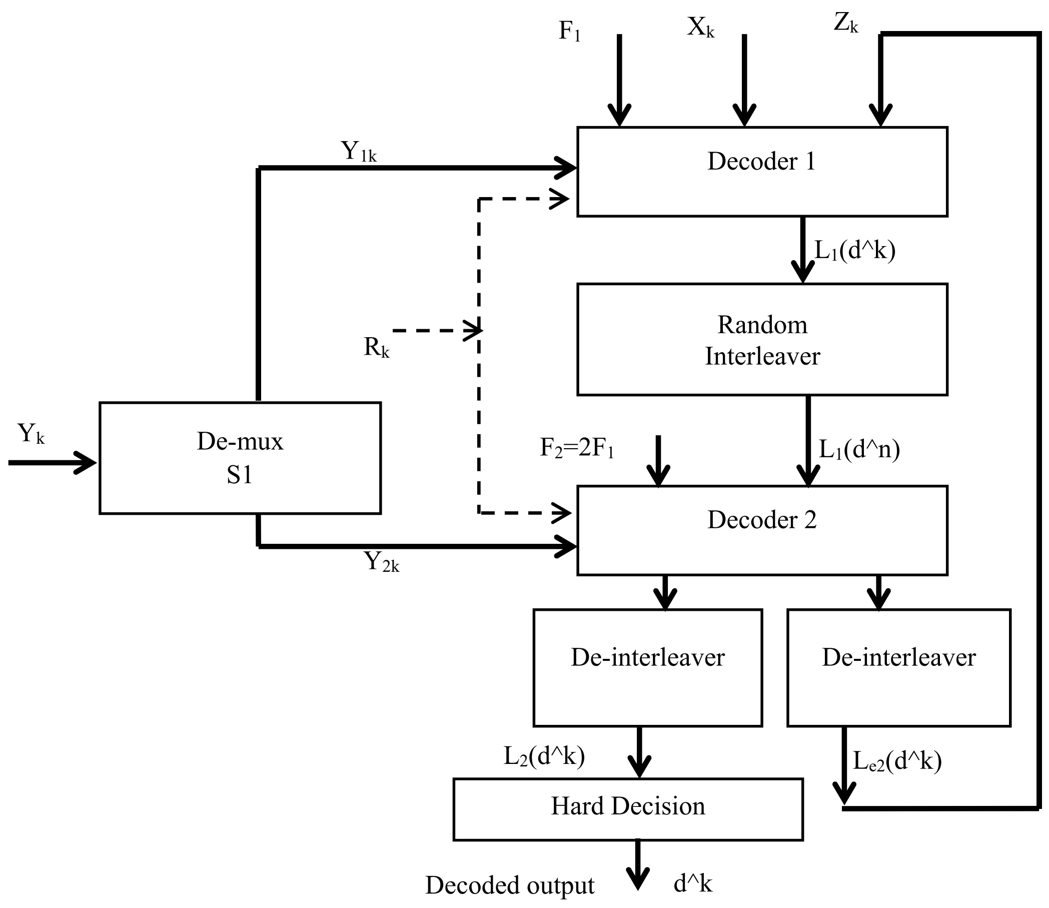


Figure 2: Turbo Decoder

3.1 Turbo Decoder Iterative Algorithm

Considering the log-likelihood ratio(LLR) the output of RSC decoder $L(\hat{d})$ is expressed considering three LLRs which are statistically independent such as channel measurement LLR ($L_c(x_k)$), a priori knowledge of data (L(d)) and extrinsic LLR ($L_e(\hat{d}_k)$)as in Eq. (10)

$$L(\hat{d}) = L_c(x_k) + L(d) + L_e(\hat{d}_k) \qquad (10)$$

Considering the data bits are equally likely i.e. $L_e(\hat{d}_k) = L(\hat{d}_k)|_{xk=0}$ Eq. (10) is rewritten as Eq. (11) for the soft decision output at any given time k. The terms $L_c(x_k)$ and $L_e(\hat{d}_k)$ are data sequences assumed to be corrupted with uncorrelated noise.

$$L_2(\hat{d}_k) = f[L_1(\hat{d}_k)] + L_{e2}(\hat{d}_k) \qquad (11)$$

The data bits d_k being decoded by the decoder can be observed using the information in the data sequence $L_e(\hat{d}_k)$. Decoder 1 processes inputs Y_{1k} and feedback input zk ($L_{e2}(\hat{d}_k)$) to generate the data sequence $L_1(\hat{d}_k)$. Similarly, decoder 2 processes two inputs y_{2k} and L_1(d^n) to generate Le2(d^k) and L_2(d^k). The output L_{e2}(d^k) is fed back in the loop and the output L_2(d^k) is processed by the hard decision logic to generate binary data. The output of decoder 2 is expressed as in Eq. (11), considering that both inputs are statistically independent. Considering the underwater channel as Gaussian channel, the term L_1(d^k) is expressed as in Eq. (12),

$$L_1(\hat{d}_k) = \frac{2}{\sigma_0^2} x_k + L_{e1}(\hat{d}_k) \qquad (12)$$

The extrinsic data L_{e2}(d^k) and the observation vectors that are used by the two decoders X_k and Y_{1k} are weakly correlated due to the presence of interleaver between decoder 1 and decoder 2. As they are weakly correlated, using these data sequences together helps in decoding of the binary sequence dk. Decoder performance in decoding binary bits with good BER is achieved by an increase in the number of iterations. With the feedback loop and complexity in the operation of each of the decoders, there is a large delay in the turbo decoding process. Literature studies on turbo decoder design reducing computation complexity have been reported in [32]. Another approach is to use the Viterbi decoding algorithm in place of turbo decoders and it is estimated that turbo decoders are twice complex as Viterbi decoders are less complex than turbo decoders both in terms of decoding time and circuit complexity. An improved method reducing decoding time and power dissipation in the Viterbi decoding algorithm is presented by self ref [35]. Adaptive decoding logic with a feedforward approach is presented for the Viterbi decoder algorithm. To achieve good BER performances and improving hardware efficiency in terms of delay and power metrics Adaptive Turbo Decoder (ATD) is proposed.

4. ARCHITECTURE OF VITERBI DECODER

The architecture of the design is as shown in Figure 3. There are 5 main units in the Viterbi decoder: Branch Metric Unit (BMU), Add Compare Select Unit (ACS), Traceback Unit (TBU), Memory Management Unit (MMU), its Survivor Memory (SMU), and Control Unit. The key challenge in designing low-power Viterbi decoder architecture is the realization of the suitable structures for the three major blocks in the decoder i.e. the BMU, ACS, and the SMU.

4.1 Control unit, clock, BMU and ACS

The Control unit and the clock unit is used to provide control and clock signals for all blocks. The control unit generates signals such as CompareStart, TB_EN, and ACSpage. From the trellis diagram we see that for the first (K–1) stages on each node, there is only one incoming branch entering a node. The CompareStart will signal ACS not to do a compare operation during the first (K-1) process since there is only one survivor. The traceback process will begin only after there were 32 sets of survivor data in the survivor memory when the TB_EN is high. The ACS page signal indicates the processed state in the trellis.

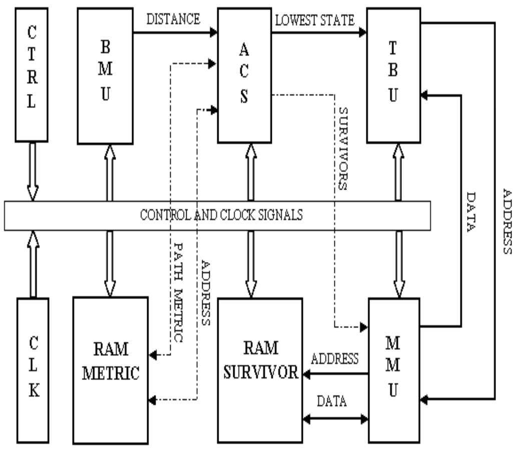
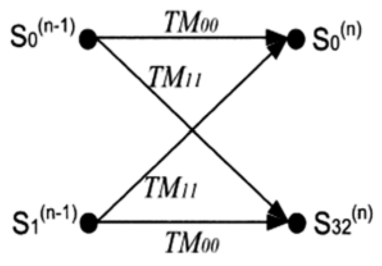
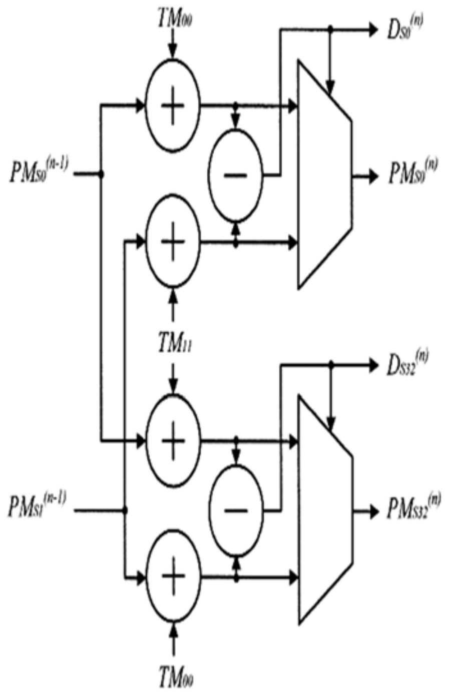


Figure 3: Architecture of Viterbi decoder

The Branch Metric Unit (BMU) calculates the distance between the received symbols and codeword on the branches. Hamming distance is used to calculate the minimum distances. The BMU is internally composed of an encoder which computes the codeword and some additional circuitry to calculate the hamming distance between the computed codeword and the received codeword. The input to this block is a 2-bit codeword which forms the received data. The expected data which is also 2-bit wide is compared with the received bits to obtain the hamming distance. The BMU calculates its distance with all possibility of branch metric, giving the distances of the code word. In this way, all possible values of branch metrics are



(a) Trellis Diagram



(b) ACS architecture

Figure 4: State transition and corresponding ACS structure

generated by the BMU. These distances are later used by the ACS unit to select the survivors among the incoming branches. The second block in the decoder is the Add-Compare-Select unit (ACS) which is computationally the most demanding part in the Turbo decoder (TD). The ACS unit consumes the most power and area in the TD due to the large no. of ACS operations which is equal to 2K-1 per trellis stage. This block selects the optimal path to each state in the Viterbi trellis. The required ACS operations per trellis stage are equal to 2K-1. A path metric unit summarizes branch metrics to get metrics for the 2K−1 path, one of which can eventually be chosen as optimal. The results of these decisions i.e. the survivors are written to the memory of a traceback unit. ACS architecture is shown in Figure 4.

Figure 4(a) shows an intermediate stage of the trellis in the 64-stage trellis diagram. The TM represents the Transition Metric or the Branch Metric value. S stands for an intermediate state such as S_0 and S_{32} as shown in figure 4(b). Figure 4(b) forms the ACS unit for trellis shown in Figure 4(a). The PM_{S0} $^{(n-1)}$ and $PM_{S32}^{(n-1)}$ is Path Metric of the previously calculated path which is retrieved from the memory for the ACS operation. The design uses four parallel simplified ACS units which perform comparison among candidate paths to determine survivors and compute the corresponding path metrics. In figure 4(b) shown, D_{S0} and D_{S32} are the survivors obtained after the ACS operations. The four-state ACS unit updates path metrics for a single iteration of the trellis. On each clock cycle, four path metrics from the previous stage are input and four updated path metrics are output. The PM_{S0} and PM_{S32} are the current state path metric or the updated path metric obtained after the ACS operation which is temporarily stored in the RAM and retrieved again for calculating the next state ACS operations. Each updating also generates a vector of four 1-b decisions that are output to the trace-back unit. All outputs are registered in flip-flops.

4.2 SMU, MMU and TBU

The survivor memory (SMU) is used to save the survivor's data calculated by the ACS Unit. The survivor memory size is chosen according to the constraint length. The survivor data will be later used by the Traceback unit to find decoded data. The Memory Management Unit (MMU) block will control the operation of survivor memory. The memory management unit generates signals such as TBPage, AddressTB, DataRAM, Readclock, and Writeclock. The TBPage holds the value of the current stage in the trellis. For a read operation, the address is generated from the TBPage signal from the MMU, and the AddressTB signal coming from the Traceback unit. The resulting Address, the combination of TBpage and AddressTB, will then be used to read the survivor data. For a write operation, the DataRAM signal will be sent into the Traceback Unit. The Traceback Unit will provide the next AddressTB signal. By decreasing the TBPage, we will be able to read the survivor data from the memory. The Traceback unit (TBU) is the final stage in the decoder which retrieves the survivors from the memory and decodes the data. The ACS block assigns the measurement functions to each state, but the actual Viterbi decisions on encoder states are based on the traceback operation to find the path of the states. Using the traceback operation, every state from a current time is followed backwards through its maximum likelihood path. All the paths converge at a point somewhere previous point in time. The point at which the corrected bitstreams starts is called the traceback depth. By finding this point, the traceback operation decisively determines the state of the encoder at a given time, by showing that, at this point, there is no choice for an encoder state given the global maximum likelihood path.

The performance of the Viterbi decoder largely depends upon the traceback depth. The increase in traceback depth increases the complexity and hardware exponentially so one has to trade-off between the performance level and the complexity and hardware. Normally for decoders using non-punctured codes, the traceback depth equals 5- times constraint length, which is enough to decode the correct output in the presence of noise. But for decoders using punctured codes and a 2/3 data rate, and 8-times constraint length is optimal. If the data rate is 3/4, then a 10-times constraint length should be used. The main bottleneck in implementing the Viterbi algorithm for convolution codes is that, for every newly received symbol information, the algorithm must trace back over multiple states. The summary of the decoding process is as shown in the flow chart of Figure 5.

5. ADAPTIVE VITERBI DECODER (AVD) FOR TURBO DECODING

In [33] adaptive Viterbi decoder architecture based on systolic array architecture with time multiplexing and pipelining is implemented on Virtex-II FPGA operating at a maximum frequency of 40 MHz and consuming power of less than 800 MW. In [34] reconfigurable hardware that takes full advantage of adaptive Viterbi algorithm is implemented with parallelism scheme on Xilinx XC4036-based PCI board, with 20%

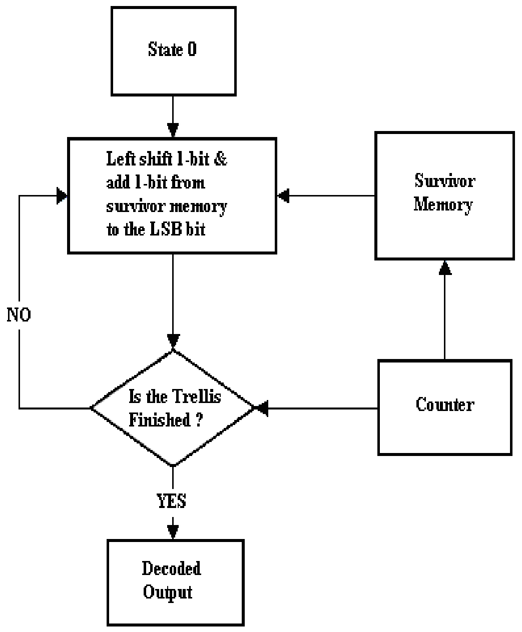

State 0
Left shift 1-bit & add 1-bit from survivor memory to the LSB bit
Survivor Memory
NO
Is the Trellis Finished ?
Counter
YES
Decoded Output

Figure 5: Decoding process

performance improvement with a maximum operating frequency of 40 MHz. Very few studies have been reported on Adaptive Turbo Decoder algorithm implementation. In this section, a detailed discussion is presented on ATD architecture design and VLSI implementation. To improve the operating speed and reduce the power dissipation of adaptive Turbo decoder architecture on the FPGA platform, it is required to design the subsystems as compatible with FPGA resources. FPGA consists of slices, Look Up Table (LUTs), multiplexers, flip flops, adders, multipliers, and memory elements. In this work, an adaptive decoding algorithm is realized on FPGA by utilizing logic elements in the configurable logic block.

In the Viterbi decoder, there are four nodes in every trellis. Each of these nodes (S0, S1, S2, S3) performs branch metric operation and accumulate, compare, select operation. In Figure 6 the adaptive logic that is proposed in [36-37] is presented that forms the sub-block for adaptive Viterbi decoder architecture for one node. In the branch metric unit C_1 and C_2 which are expected outputs at every state is XOR with received code R for computing branch metric and the path metric is updated with branch metric. In the adaptive logic with feed-forward data processing, the lowest path is computed and is retained simultaneously. The path selected is stored in the register along with the corresponding decoded message bit. The path metric computed in every clock is fed back into the ACS unit for computation of next state path metrics and the decoded message for a new set of the received codeword.

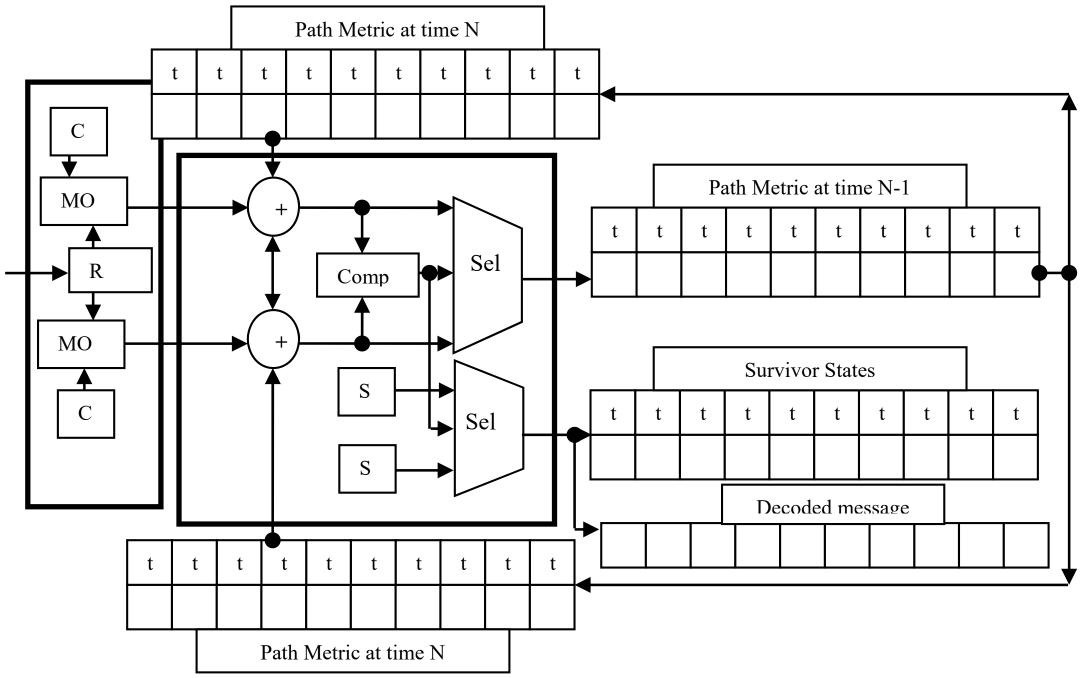


Figure 6: Decoder Architecture for every node.

The ACSU operation with path metric update (PMU) for the j and j+2 nodes is given in Eq. (13). From the path metric the selection of minimum path is carried out based on the decision $D_{Sj}^{(n)}$ and $D_{Sj+2}^{(n)}$ as in Eq. (14)

$$PM_{Sj}^{(n)} = min\left\{PM_{S2j}^{(n-1)} + C_{2j},\ PM_{S2j+1}^{(n-1)} + C_{2j+1,j}\right\}$$

$$PM_{Sj+2}^{(n)} = min\left\{PM_{S2j}^{(n-1)} + C_{2j,j+2},\ PM_{S2j+1}^{(n-1)} + C_{2j+1,j+2}\right\} \quad (13)$$

$$D_{Sj}^{(n)} = sign\left\{PM_{S2j}^{(n-1)} + C_{2j} - PM_{S2j+1}^{(n-1)} + C_{2j+1,j}\right\}$$

$$D_{Sj+2}^{(n)} = sign\left\{PM_{S2j}^{(n-1)} + C_{2j,j+2} - PM_{S2j+1}^{(n-1)} + C_{2j+1,j+2}\right\} \quad (14)$$

The path metric is computed for multiple clocks and the minimum path metric is computed by identifying the most-likely message data and the truncation length. The truncation length TL for constraint K is given as in Eq. (15), where R is the coding rate, C is the channel capacity.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$\frac{T_L}{K} \geq \begin{cases} \dfrac{1}{1-\dfrac{2R}{C}}, & 0 \leq R \leq \dfrac{C}{4} \\ \dfrac{1}{2\left(1-\sqrt{\dfrac{R}{C}}\right)^2}, & \dfrac{C}{4} \leq R \leq \dfrac{C}{2} \\ \dfrac{\left(1+\sqrt{\dfrac{R}{C}}\right)}{\left(1-\sqrt{\dfrac{R}{C}}\right)}, & \dfrac{C}{2} \leq R < C \end{cases} \quad < \qquad (15)$$

The trellis paths for the decoding algorithm with input sequence [11 01 00 10 10 01 01 11 10 00 01] and error observed at 5th, 9th, and 14th-bit positions are shown in Figure 7. It is observed

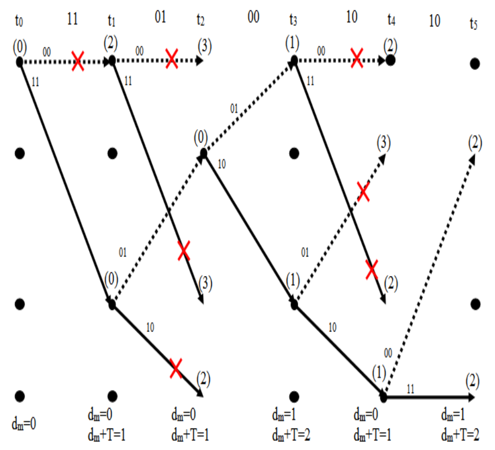


Figure 8: Adaptive Viterbi algorithm (trellis states $t_0 - t_5$)

$(T + d_m)$ and survivor paths that do not exceed (N_{max}). Figure 8 shows the trellis diagram for the Adaptive Viterbi Decoder for trellis states from time t_0 to t_5. Comparing Figure 7 and Figure 8 the number of survivor paths in time t_5 are eight and two, respectively.

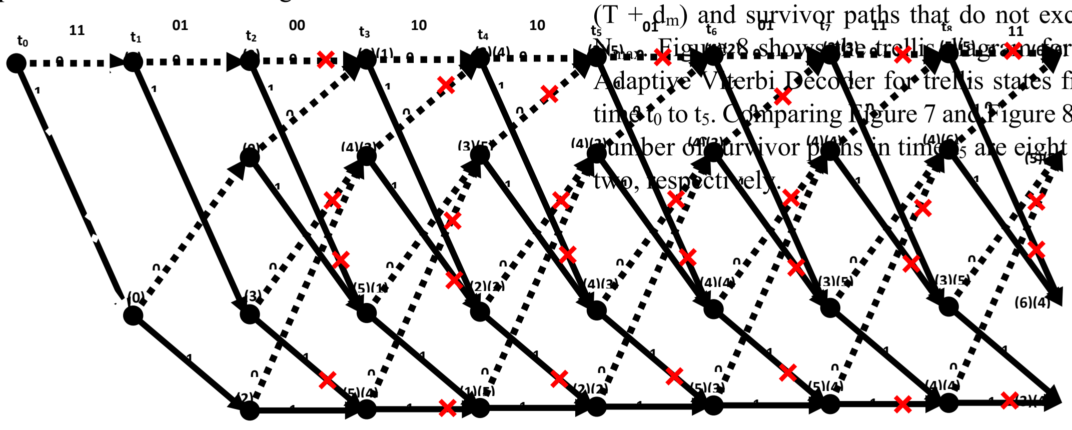


Figure 7: Viterbi decoding algorithm

that as at every stage of decoding there are eight paths that need to be verified and the path with minimum error to be retained thus increasing decoding complexity. The path metric after decoding is traced back to identify the message bits which further delays the decoding time. In AVD instead of retaining all 2K-1 paths, the computation complexity and memory requirement is reduced by retaining only paths that satisfy predetermined criteria. The three parameters that decide the number of paths to be retained are the threshold (T), Hamming distance (d_m), and a total number of survivor paths (N_{max}). The decoding algorithm retains only the path metrics that are less than

5.1 Adaptive Algorithm

At the time i, the trellis would have four states with path metrics. The selection of path metrics before progressing to the next time interval is based on T and N_{max}. With the threshold set to T = 1, and Nmax = 3, the path metrics with less than the threshold are considered for next stage decoding, all the other paths are discarded. The threshold T is adaptively varied as the trellis progress from the current state to the next state. Initially to begin with T is set to 1, as the trellis progress to time interval 4, T is dynamically changed to 2, and so on. If the number of iterations is less than truncation length the algorithm moves to step 2, else the survivor paths stored at every stage are used in decoding the message bits.

To achieve BER on par with Viterbi decoding algorithm selection of T the threshold and N_{max} the number of paths plays a critical role. Selection of T and N_{max} to a large value increases the area and memory size, with additional computation complexity. Smaller values of T and Nmax reduce BER, hence optimal values of T and N_{max} need to be chosen. For hardware implementation with reduced complexity in terms of area and power T = 1 and N_{max} = 3 is recommended in [38]. At every stage of decoding the threshold (dm + T) is kept constant and hence may lead to a decoding error. To overcome this limitation, it is required to adaptively change the threshold values. If the truncation length is t_N, consisting of N-1 trellis states, requiring N-1 time slots for decoding, the threshold values are set as in Eq. (16),

$$threshold\ T = \begin{cases} dm + T,\ 0 \leq t < t + i \\ dm + 2T, t + i \leq t < t + \ 2i \\ dm + 3T, t + 2i \leq t < tN \end{cases} \tag{16}$$

Where i is an integer (i = 7 is selected in this work), the selection of i depends on the number of input bits being encoded, convolutional encoder, and truncation length. In this work the threshold value T is set with truncation length, the architecture dynamically reconfigures with several decoding stages and trellis states. In figure 2 presented, two changes were carried out to design from turbo decoder to adaptive turbo decoder. The threshold T is defined for dynamic adaption of Viterbi decoder and Viterbi decoder is replaced with dynamically reconfigurable adaptive Viterbi decoder. The proposed adaptive Turbo decoder architecture is modeled in MATLAB and is evaluated for its performances considering different modulation schemes and underwater channel models. HDL model for ATD is developed and implemented on Xilinx FPGA device further optimizing for the area, power, and timing requirements.

6. RESULTS AND DISCUSSION

The Turbo encoder and adaptive turbo decoder model proposed in this work are evaluated for their BER performances considering different modulation schemes. Random data source with a frame length of 2048 bits is symbol mapped at the transmitter and its being turbo encoded. Considering three different channels such as AWGN, Rayleigh and Rician models with multiple paths and Doppler shifts the turbo encoded data is filtered by the channel model.

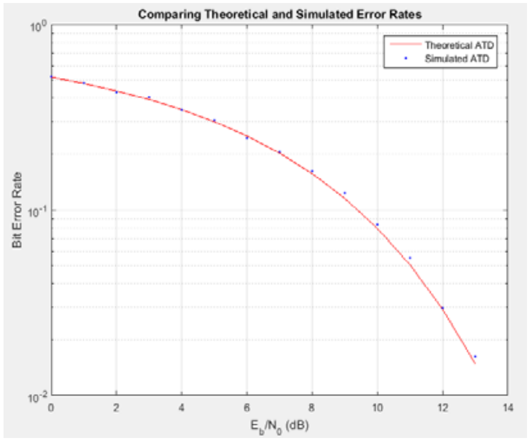


Figure 9: Validation of ATD model

At the receiver, adaptive turbo decoder and demodulation scheme decode the transmitted data, and BER is computed. The simulation results of BER for various SNR is presented in Figure 9. For a known set of data bits of length 256 bits that are encoded using turbo encoder, the error is introduced at 90 different positions and the information is decoded using ATD. The theoretical and simulated ATD results for various Eb/No obtained demonstrates the logic correctness of the developed model in MATLAB.

To evaluate BER performances of ATD and compare with performances of Adaptive Viterbi De-coder and Turbo decoder, MATLAB model for all de-coding methods are developed. It is observed that by increasing the number of iterations (k), BER performances can be improved in all three decoding methods. Figure 10 and Figure 11 presents a comparison of all three decoding methods for k=7, 8, and 12. It is observed that ATD achieves better BER results as compared with TD and AVD.

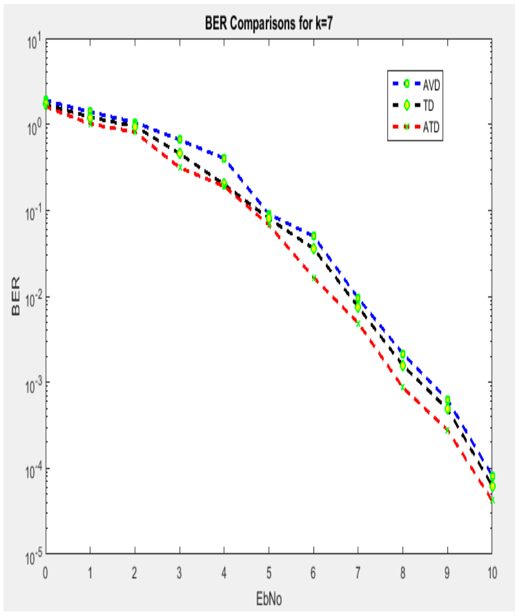


Figure 10: BER comparison at k=7

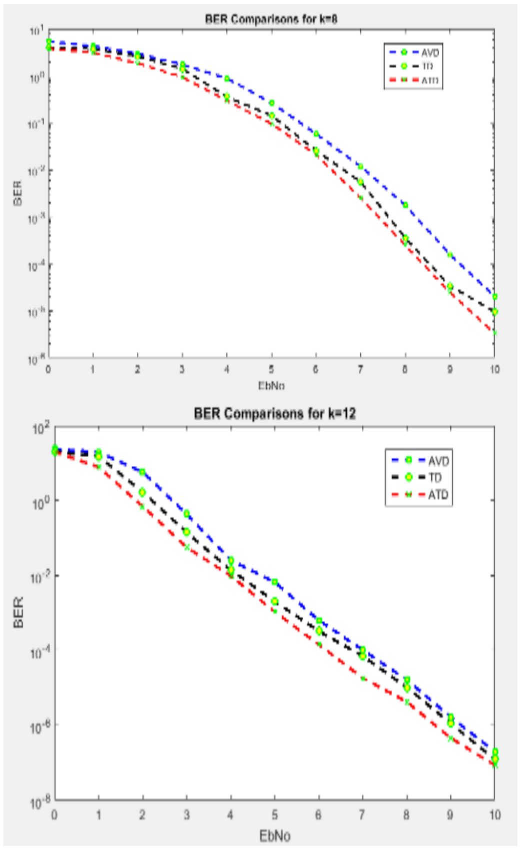


Figure 11: BER comparison (a) k=8 (b) k=12

As the number of iterations is more ATD decoder can achieve BER in the order of 10^{-4} and above. For Eb/No less than 5 dB all three decoders achieve similar BER but with Eb/No above 5 dB ATD achieves BER better than TD and AVD. The adaptive algorithm in the Viterbi decoding process can distinguish between the path metric with minimum error and false path metric. The decoded message from the first adaptive

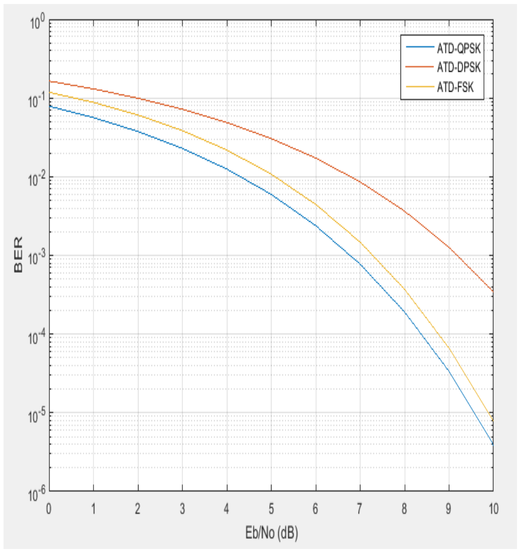


Figure 12: Modulation scheme with ATD

decoder and the interleaved input that is used for second level decoding improves the message decoding process. The feedback mechanism in both decoders helps in observing the message decoding process and with the number of iterations improves BER performances.

Figure 12 presents BER performances of three modulation schemes with ATD used for error decoding. QPSK modulation scheme with ATD can achieve BER in the range of 10^{-4} and above as compared with DPSK and FSK modulation schemes. Rayleigh channel and Rician channel with input signal sampled at 100 kHz, maximum Doppler shift of 130 Hz with SNR of channel varied between 0 to 20 dB is modeled. BPSK modulation scheme is used for symbol mapping for a test input data of 5000 bits. Turbo decoding algorithm and Adaptive Turbo Decoding algorithm is incorporated in the MATLAB model with Rayleigh and Rician channel.

Figure 13 presents the BER comparisons of both models. It is observed that for the corresponding random data considered ATD can achieve better BER in the order of 10^{-4} in Rician

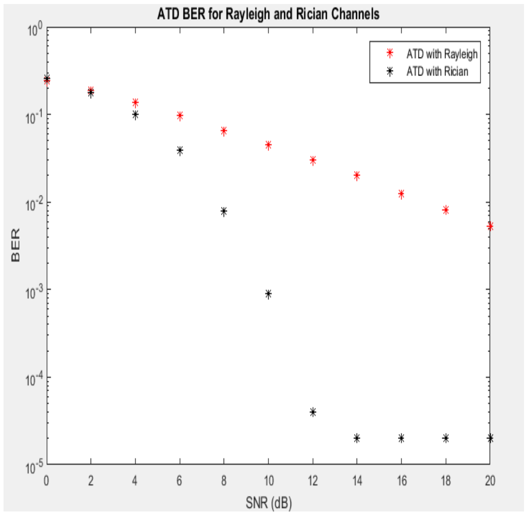


Figure 13: BER comparison of underwater channel with ATD

channel environment compared with BER performances of ATD in Rayleigh channel model. To evaluate the performances of ATD in an underwater environment with multiple paths, MATLAB model with an underwater channel with multipath

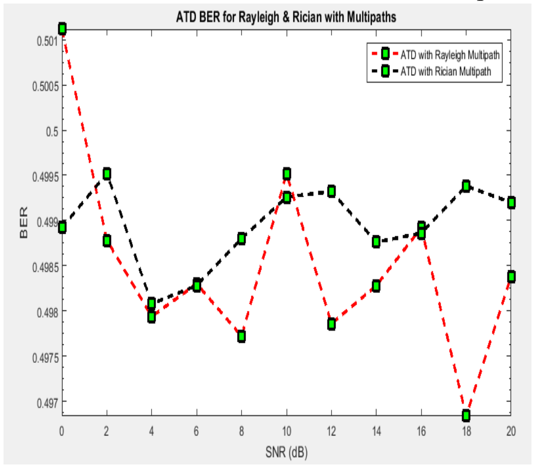


Figure 14: Comparison of ATD in multipath environment

fading and signal power is developed. Path delays varying from 200ns to 4000ns and path gains varying from -0.9 dB to -30 dB are considered for analysis. For the test input considered, it is observed that ATD performs decoding process better in Rayleigh channel environment com-pared with Rician channel. In a multipath environment, ATD is recommended for error correction in an underwater environment as observed from the results presented in Figure 14.

6.1 FPGA Implementation

Computation complexity of ATD for hardware implementation is addressed by design of Adaptive Viterbi Decoder using feedforward message decoding logic and LUT based sub-block designs. The top-level block diagram presented in Figure 2 is modeled using the hierarchical design method. The AVD is developed considering rate ½ and constraint length of 4. Interleaver and de-interleaver with random logic are modeled considering 12 different random logics that are stored in LUT and interleaver is configured based on a 4-bit input pattern. Multiple intermediate registers are introduced for feedback and data synchronization.

The AVD architecture is designed for high speed and low power requirements. Suitable test cases are considered for functional verification of ATD. The test bench is generated for the adaptive turbo decoder architecture designed in this work is of two sequences 'a' and 'b' each of 18-bit. The originally transmitted data is 111000 which is processed by the turbo encoder and generates the encoded data, which is of 12 bits, which is padded with 6-bits. Figure 15 presents the simulation results of the test bench for ATD. The two inputs are de-multiplexed and are processed by the two AVDs to generate a decoded message. The first input data is the data with predefined error is applied to the first ATD and the second data with error input is interleaved and applied to the second AVD. ATD operation is performed and output obtained from Finrout is observed. For the test case considered and presented in Figure 15, the ATD decodes the message bit as "111000" which is like the input message

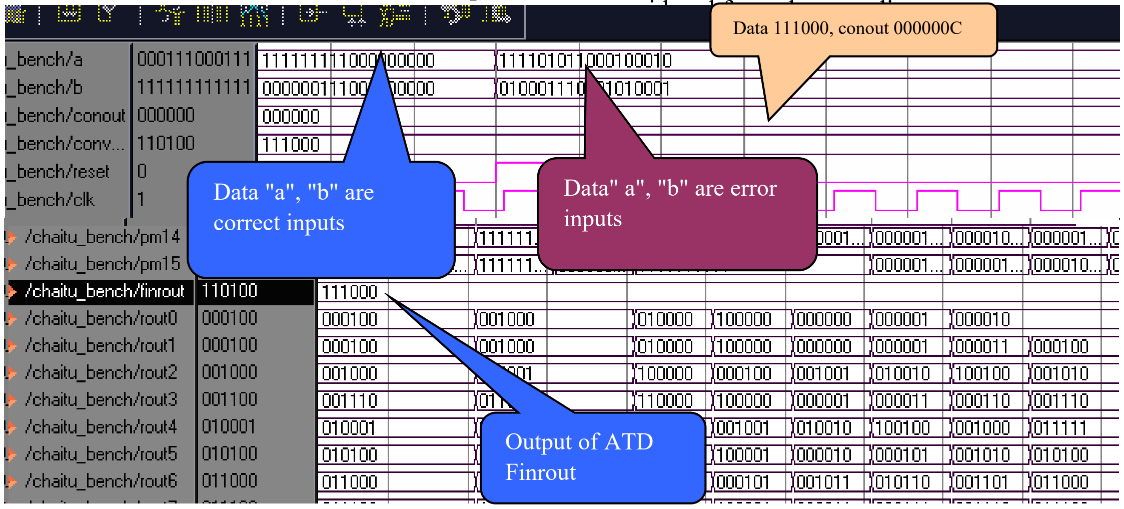
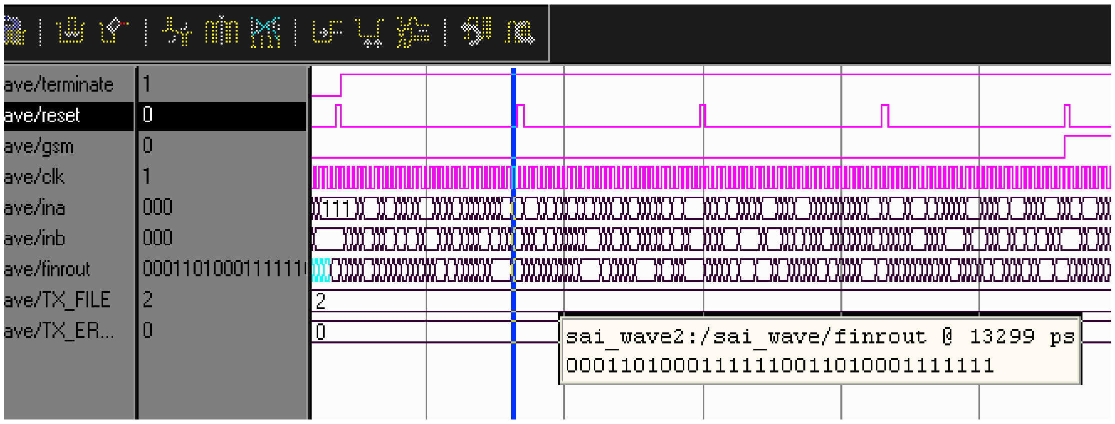


Figure 15: Simulation results of ATD

Figure 16 presents the simulation results of ATD for 378 input bits. It is observed that decoding



Fiaure 16: Simulation result of ATD

378 input bits requires multiple iterations. Every iteration requires 189 clock cycles and 7 iterations have been considered to decode the correct message without errors.

The HDL code for ATD is evaluated for decoding 2048 length input data with a maximum of 60 errors and it is observed that the ATD can decode the correct message in all test cases. The functionally verified HDL model is synthesized using Xilinx ISE and its hardware performances in terms of area, timing, and power are estimated from the synthesis report. Table 1 and Table 2 presents the comparison of three different decoders considering slice LUT utilization, maximum operating frequency, and total power required for logic implementation. Two different input data with variation in error (SNR of 3dB and SNR of 4dB) are considered for analysis.

From the synthesis results, it is observed that ATD occupies 1296 LUTs and operates at a maximum frequency of 301 MHz. The power dissipation is estimated to be 34.63 mW at 200 Mbps for ATD. Increasing the number of iterations from k=7 to k=10, the number of slices LUTs and power dissipation for ATD implementation on FPGA is increased. Maximum operating frequency is reduced for ATD. From the results observed the power dissipation for generating the output for both AVD and ATD is less compared with the power dissipation of turbo decoder. The feedforward message decoding unit designed in this work minimizes power dissipation for output generation. Adaptive logic in turbo decoding improves operating frequency by a factor of 25% as compared with turbo decoding without adaptive logic. The area complexity measured in terms of the number of LUTs is reduced by 55%. The advantages of ATD are demonstrated in terms of reduced power dissipation and area utilization.

To further optimize the area and power requirement, the advanced synthesis

Table 1: Comparison decoding algorithm for k=7

Parameter	AVD		Turbo		Adaptive Turbo	
Slice LUTs - Max. no. 69120	1011	941	3128	2884	1484	129 6
SNR (dB)	3	4	3	4	3	4
Max. Operating Freq. (MHz)	322	341	214	225	289	301
Power(mW) @200 Mbps						
Decoding Computation	25.74	24.87	33.52	31.99	28.71	26.1 1
Output Generation	9.56	7.91	36.42	16.21	10.69	8.52
Total	35.3	32.78	69.94	48.2	39.4	34.6 3

Table 2: Comparison of decoding algorithm for k=10

Parameter	AVD		Turbo		Adaptive Turbo	
Slice LUTs - Max. no. 69120	1894	1254	5622	4334	2264	1456
SNR (dB)	3	4	3	4	3	4
Max. Operating Freq. (MHz)	311	326	189	212	231	278
Power (mW) @200 Mbps						
Decoding Computation	32.11	28.45	42.12	36.13	34.11	30.41
Output Generation	10.16	8.13	38.45	18.29	11.19	9.29
Total	42.27	36.58	80.57	54.42	45.30	39.70

optimization process is used in the Xilinx ISE environment. In the advanced synthesis optimization process, priority for optimization is set to memory usage and improvement in speed by using grey code FSM coding style. Results of advanced synthesis are presented in Table 3. The maximum operating frequency of ATD is increased from 231 MHz to 308 MHz. The frequency increased by a factor of 25% with power dissipation increasing by 4% and LUTs by 20%.

Table 4 compares the FPGA implementation performances of different error decoders in terms of operating frequency, area requirement, and power dissipation. Adaptive turbo decoders reported in the literature are based on software models and very few have attempted in FPGA implementation. In this work, adaptive logic for turbo decoding is designed with feedforward mechanism and LUT based sub-block design optimizing both power and area requirement. The ATD proposed in this work operates at a maximum frequency of 308 MHz and requires less than 4% of FPGA resources. With reduced power dissipation and higher operating frequency, the proposed turbo decoder can be integrated with the OFDM system for underwater communication.

7. CONCLUSION

ATD module developed in the work is demonstrated to achieve BER in order of 10^{-4} in both Rayleigh and Rician underwater channel environments for different constrain lengths (K=7 to 12). BER performance of ATD in the Rician channel is superior compared to the Rayleigh channel. The decoding complexity of ATD is addressed by designing multiplexer based sub-systems that are compatible with FPGA implementations. The processing speed of 308 MHz with a power dissipation of 47.23 MW and area optimization of 95 % is achieved for underwater communication applications. Power requirement of less than 50 MW improves endurance time of MODEMs by 3X times as compared with existing modules. The multiplexer based mechanism in designing ATD sub-systems achieves adaptability of error coding techniques for underwater applications.

Table 3: Comparison of decoding algorithm with advanced synthesis of FPGA

Memory and speed optimization with grey code FSM style		
Parameter	**Turbo Encoding**	**Adaptive Turbo Encoding**
Slice LUTs - Max. no. 69120	5622	2864
Number LUT-FF pairs (Max. 1014)	364	212
Power dissipation (mW)	80.57	47.23
Max. operating freq. (MHz)	189	308

Table 4: Comparison of Turbo decoder architectures

	This work (ATD)	**Gan & Jun 2010 [37]**	**Wonsun et al. 2012 [38]**	**Girish et al. 2017 [35]**	**Sujatha Cyril et al. 2014 [36]**
Device	Virtex-5	EP1S20F484/ APEX EP20K200	XC2V1000-4FG256	Virtex-5	XCV5110t ff1136
Data rate	308 MHz	102MHz/ 40MHz, 588 Kbps	40 MHz 78 Kbps	188.91MHz 180 Mbps	153.759MHz 160 Mbps
Resource utilization	2854 out of 69120 (LUTs)	14226/ 2200 (Logic elements)	2934 Slices	5622 out of 69120 (LUTs)	941 out of 69120 (LUTs)
Power dissipation	47.23 mW	---	473 mW	80.57 mW	1.192 W

ACKNOWLEDGEMENT

The authors gratefully acknowledge the department of electronics and communication, BMS College of engineering for extending all the facilities for carrying out this work. The authors also like to thank Dr. Cyril Prasanna Raj P, Dean Research, M S Engineering College for his valuable comments, suggestions, and technical assistance during the underwater experiments.

REFERENCES

[1] Milica Stojanovic, "Recent Advances in High-speed Underwater Acoustic Communications" *IEEE Journal of Oceanıc Engıneerıng,* Vol. 21, No. 2, pp. 125-136,1996.
[2] A. Goalic, J. Trubuil, and N. Beuzelin, "Channel coding for underwater acoustic communication system" *OCEANS, September 18-21*, Boston, Ma, USA, 2006.
[3] J. Tribuil, A. Goalic, and N. Beuzelin, "Synchronization and channel coding in shallow water acoustic communication" *OCEANS,* September 15-18, Quebec, Canada, 2008.
[4] R. Pyndiah, A. Glavieux, A. Picart and S. Jacq, "Near optimum decoding of product codes" *in Proc of IEEE Globecom '94*, vol. 1/3, Nov-Dec, pp. 339-343,1994.
[5] D. Chase, "A class of algorithms for decoding block codes with channel measurement information" *IEEE trans. Intor. Theory,* vol. IT-8, Jan. 1972
[6] E. R. Berlekamp, "Algebraic coding theory" *McGraw-Hill Book Company*, New York, 1968.
[7] A. Goalic, J. Trubuil, and N. Beuzelin, "An Overview of Channel coding for underwater acoustic communications" *MILCOM 2012*, 29 Oct.-1 Nov, Orlando, FL, USA, 2012.
[8] A. Radosevic, R. Ahmed, T. M. Duman, J. G. Proakis, and M. Stojanovic, "Adaptive OFDM modulation for underwater acoustic communications: Design considerations and experimental results" *IEEE Journal of Oceanic Engineering*, vol. 39, No. 2, pp. 357–370, 2014.
[9] L. Wan, H. Zhou, X. Xu, Y. Huang, S. Zhou, Z. Shi, and J. H. Cui, "Adaptive modulation and coding for underwater acoustic OFDM" *IEEE Journal of Oceanic Engineering,* vol. 40, No. 2, pp. 327–336, 2015.
[10] E. Demirors, G. Sklivanitis, G. E. Santagati, T. Melodia, and S. N. Batalama, "Design of a software-defined underwater acoustic modem with real-time physical layer adaptation capabilities" *in ACM International Conference on Underwater Networks Systems* (WUWNet), pp.1–8,2014.
[11] Sameer A. Dawood, F. Malek, M S. Anuar, Abadal-Salam T. Hussain, "Performance

Evaluation of LDPC and Turbo-Coded OFDM Based on DMWCST" *Journal of Theoretical and Applied Information Technology,* Vol.79, No.1, Sept. 2015.

[12] S.K.Padmanabhan, T.Jayachandra Prasad "Design and Performance Analysis of Precoded OFDM Transceivers for Cognitive Radio" *Journal of Theoretical and Applied Information Technology,* Vol. 56, No.2, Oct. 2013.

[13] Mohammad Sadeghi, Mohammed Elamassie and Murat Uysal, "Adaptive OFDM-Based Acoustic Underwater Transmission: System Design and Experimental Verification" *IEEE International Black Sea Conference on Communications and Networking,* 2017.

[14] Weiwei Kan, Jue Wang, Haifeng Zhou, and Chen Xu, "Design of Underwater Acoustic Communication System Based on OFDM" *2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC),* 2018.

[15] Yougan Chen, Zhaohui Wang, LeiWan, Hao Zhou, Shengli Zhou, and Xiaomei Xu, "OFDM-Modulated Dynamic Coded Cooperation in Underwater Acoustic Channels" *IEEE Journal of Oceanic Engineering,* Vol. 40, No.1, JAN 2015.

[16] W. Han, J. Huang, and M. Jiang, "Performance analysis of underwater digital speech communication system based on LDPC codes" *in 4th IEEE Conference on Industrial Electronics and Applications*, pp. 567-570, 2009.

[17] J. Huang, S. Zhou, and P. Willett, "Nonbinary LDPC coding for multicarrier underwater acoustic communication" *IEEE Journal on Selected Areas in Communications*, vol. 26, pp. 1684-1696, 2008.

[18] X. Xiaomei, C. Yougan, Z. Lan, and F. Wei, "Comparison of the performance of LDPC codes over different underwater acoustic channels" *in 12th IEEE International Conference on Communication Technology*, pp. 155-158, 2010.

[19] Michel Barbeau, "Weak Signal Underwater Communications in the Ultra Low-Frequency Band" *Proceedings of the 7th GNU Radio Conference*. Vol. 2, No. 1, P. 8, SEP. 2017.

[20] Fano R. "A heuristic discussion of probabilistic decoding" *IEEE Transactions on Information Theory,* 9(2):64–74, April 1963.

[21] Lars M. Wolff and Sabah Badri-Hoeher, "Convolutionally Coded Hopping Pattern for MFSK Modulation in Underwater Acoustic Communication" *IEEE Access*, Vol. 7, 2019.

[22] L. Berkhovskikh and Y. Lysanov, "Fundamentals of Ocean Acoustics" Springer, 1982.

[23] M. Stojanovic, "On the Relationship between Capacity and Distance in an Underwater Acoustic Channel" *ACM SIGMOBILE Mobile Comp.* Commun. Rev., vol. 11, no. 4, Oct., pp. 34–43, 2007.

[24] F. Jensen et al., Computational Ocean Acoustics, *Springer Verlag*, 1994.

[25] M. Chitre, "A High-Frequency Warm Shallow Water Acoustic Communications Channel Model and Measurements" *J. Acoust. Soc. America,* vol. 122, No. 5, Nov, pp. 2580–86, 2007.

[26] W. B. Yang and T. C. Yang, "High-Frequency Channel Characterization for M-ary Frequency-Shift-Keying Underwater Acoustic Communications" *J. Acoust.l Soc. America,* vol. 120, No. 5, Nov., pp. 2615–26,2006.

[27] J. Preisig, "Acoustic Propagation Considerations for Underwater Acoustic Communications Network Development" *ACM SIGMOBILE Mobile Comp. Commun. Rev.*, vol. 11, No. 4, Oct. pp. 2–10, 2007.

[28] Berrou, C., Glavieux, A., and Thitimajshima, P., "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes" *IEEE Proceedings of the Int. Conf. on Communications*, Geneva, Switzerland, pp. 1064-1070, May 1993.

[29] Ali Calhan, Celal Ceken, and Ismail Erturk, "Comparative Performance Analysis of Forward Error Correction Techniques in Wireless Communications" *Proceedings of the Third International Conference on Wireless and Mobile Communications*

(ICWMC), IEEE Computer Society, page 63, Washington DC, USA, 2007.

[30] Dolinar, S. and Divsalar, D., "Weight Distributions for Turbo Codes Using Random and Nonrandom Permutations" *TDA Progress Report 42-122, Jet Propulsion Laboratory*, Pasadena, California, August 15, pp. 56-65,1995.

[31] Robertson, P., Villebrun, E., and Hoeher, P., "A Comparison of Optimal and Sub-Optimal MAP Decoding Algorithms Operating in the Log Domain," *Proc. of ICC '95,* Seattle, Washington, pp. 1009-1013, June 1995.

[32] Benedetto, S., et al., "A Soft-Input Soft-Output APP Module for Iterative Decoding of Concatenated Codes," *IEEE Communications Letters,* vol. 1, No. 1, pp. 22-24, Jan 1997.

[33] Man Guo, Omair Ahmad, M, and Chunyan Wang, "FPGA Design and Implementation of a Low-Power Systolic Array-Based Adaptive Viterbi Decoder" *IEEE Transactions on circuits and systems—I*: regular papers, vol. 52, No. 2, pp. 350-365, 2005.

[34] Sriram Swaminathan, Russell Tessier, and Dennis Goeckel, "A dynamically reconfigurable adaptive Viterbi decoder" *Proceedings of the International Symposium on Field Programmable Gate Arrays Archive*, pp. 227-236,2002.

[35] Girish N and Veena M B, "High-Speed Low Power Adaptive Viterbi Decoder Architecture for Underwater Acoustic Communication with Turbo Codes" *International Journal of Applied Engineering Research*, Vol. 12, No. 18, pp. 7982-7989, 2017

[36] Sujatha Cyril and Dharmistan K Varughese, "Hybrid Architecture for OFDM with Optimized Design of Analog Viterbi Decoder" *International Journal of Computer Applications*, Vol.55, No.2, Oct 2012.

[37] Gan Ouyang, Jun Yan Ren,"A High Throughput Low Power Soft-Output Viterbi Decoder" 2010.

[38] Wonsun Yoo, Yunho Jung, Moo Young Kim & Seongjoo Lee, "A Pipelined 8-bit Soft Decision Viterbi Decoder for IEEE802.11ac WLAN Systems" *IEEE Transactions on Consumer Electronics*, vol. 58, No. 4, pp. 1162-1168,2012.