Defence Science Journal, Vol. 75, No. 2, March 2025, pp. 206-214, DOI : 10.14429/dsj.19961

Approximate Successive Cancellation Decoder for Polar Codes

Jali Nandini*, Muralidhar Pullakandam and Sreehari Rao Patri

National Institute of Technology, Warangal – 506 004, India
**E-mail: jalinandini@gmail.com*

ABSTRACT

Polar codes are the forward error correcting (FEC) codes renowned for achieving channel capacity for various codeword lengths. A low-complexity decoder, termed a Successive Cancellation (SC) decoder, is commonly employed to decode polar codes. However, the SC decoder's sequential nature leads to a drawback in terms of decoding speed. This paper proposes an approximate successive cancellation decoder (ASCD), which incorporates approximate computing techniques that are equivalent alternatives to the exact computational units. The comparator, adder-subtractor block, is replaced by approximate units in the merged processing unit, and an approximate two-bit processing unit is designed at the last stage of the decoder to reduce the hardware complexity and delay with negligible performance degradation. The overall design of the proposed ASCD is implemented targeting the Xilinx Virtex-6 FPGA platform. With the proposed approximate counterparts, the ASCD achieves an average throughput improvement of 68 % compared to the former decoders. In addition, the usage of overall hardware resources is reduced by 41 %, reducing the processing complexity. The proposed decoder proves beneficial for error-resilient applications in 5G wireless communications.

Keywords: Forward error correcting codes; Successive cancellation decoder; Approximate successive cancellation decoder; Merged processing unit; Approximate computing techniques

NOMENCLATURE

A2bPU : Approximate 2-bit processing unit
AMPU : Approximate merged processing unit
ASCD : Approximate successive cancellation decoder
BDMC : Binary discrete memoryless channels
FEC : Forward error correction codes
LLR : Logarithmic likelihood ratios
MPU : Merged processing unit
SC : Successive cancellation
SCL : Successive cancellation list
3GPP : 3rd generation partnership project

1. INTRODUCTION

Polar codes, acknowledged as the first rigorously proven capacity-achieving codes for Binary Input Discrete Memoryless Channels (BDMC)[1], have drawn significant attention within the domain of Forward Error Correction (FEC) codes. Polar codes have been integrated into the 3GPP standardisation process as the error-correcting code for the 5G New Radio[2] control channel. With their explicit structure[3-8] and streamlined encoding/decoding procedures featuring low complexity[9-12], polar codes have emerged as key contributors in the field of coding theory. Several efforts have been made to investigate different theoretical characteristics of polar codes[13-18].

The decoding techniques for polar codes can be broadly categorised into three types: the Successive Cancellation Decoder (SC), the Successive Cancellation List Decoder (SCL), and the Belief Propagation (BP) decoder. While the SC decoder is known for its low complexity, its decoding time escalates with longer code word lengths due to its serial nature. The line and tree architectures were proposed[19] to streamline hardware complexity and reduce the latency for high data rate applications. An enhanced SCL decoder, introduced with a list of L decoding paths[20-21] improves performance at the expense of increased complexity. On the other hand, the belief propagation decoder[22-23], operating in parallel and relying on message passing, offers high throughput and performance. However, a drawback emerges as the latency and power dissipation increase with more iterations.

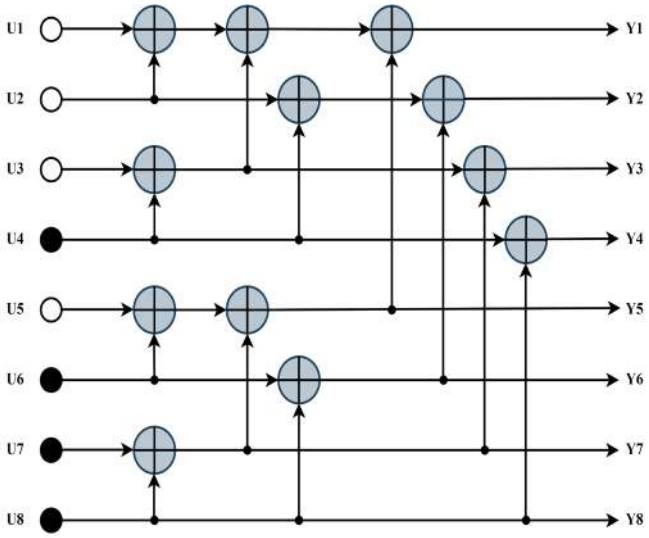
Low-complexity merging operations are introduced[24] to enhance tree-level parallelism and minimize node visitation. A parallel Partial-Sum Calculator (PSC) architecture is proposed, enabling single-cycle updates of partial-sum registers without increasing hardware complexity, significantly lowering latency and outperforming previous designs. Additionally, a reconfigurable hardware architecture[25] for SC decoders is designed using three techniques, namely Low-Area Quantization (LA-QS), High-Efficiency Frozen-Bit Control (HE-FBCS), and Grouping Storage Circuits (GSC), which deliver high performance with low latency. For 5G New Radio (NR), a detailed analysis of design parameters, focusing on hardware complexity, throughput, and latency for semi-parallel architectures, is presented[26], highlighting trade-offs between algorithmic and architectural parameters. Two multi-frame decoding architectures[27] are proposed to enhance throughput and processing unit utilisation. To address the serial nature of

Received : 09 March 2024, Revised : 04 February 2025
Accepted : 11 February 2025, Online published : 24 March 2025

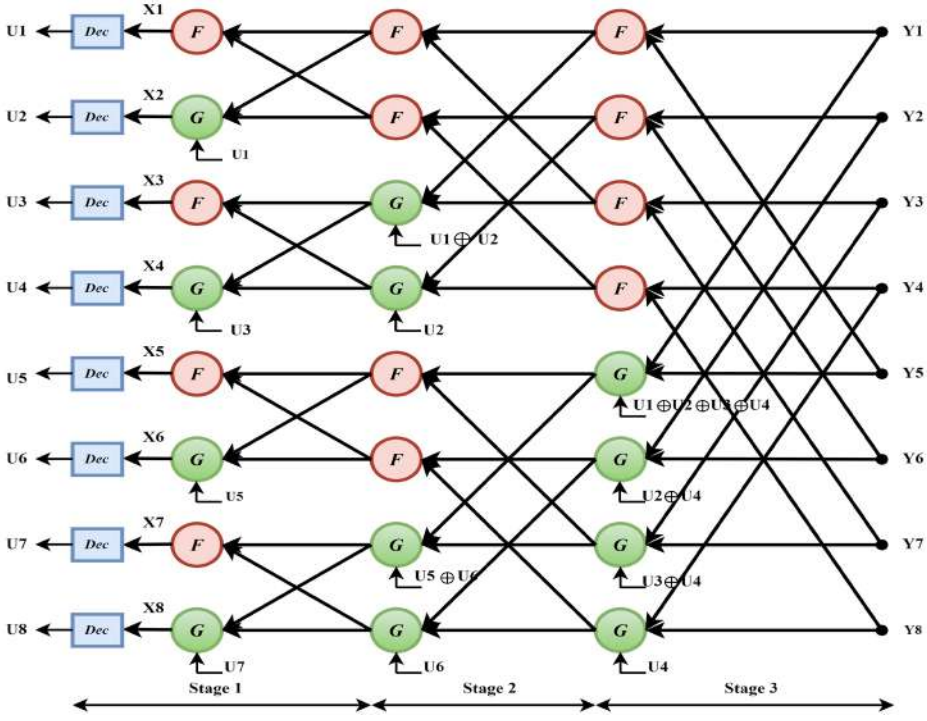
SC decoding, a fast SC decoder leveraging parallel decoders for special nodes in the binary decoding tree is introduced, reducing latency while maintaining error-correction performance and making it an efficient low-complexity solution for polar code decoders.

The motivation behind the proposed design is to develop an SC decoder that offers high throughput and low power consumption. Approximate computing has been a viable option in recent years for achieving high speed and/or low power[28]. Digital Signal Processing (DSP) systems typically need to compute with 100 % precision. Nonetheless, a certain level of computational error may not significantly impair the performance of some error-resilient systems[29-30], like high-speed video broadcasting systems.

This work introduces an architecture for the Approximate Successive Cancellation Decoder (ASCD), incorporating approximate techniques to mitigate latency. The ASCD leverages an approximate comparator, an approximate adder-subtractor unit, and an approximate 2-bit Processing Unit serving as replacements for the exact counterparts in the processing units to accelerate the performance of our decoder. The organisation of the paper is outlined below. Section II provides a comprehensive review of the fundamental concepts of polar codes; Section III explains the approximate counterparts for polar decoding; and Section IV gives the top-level architecture of the proposed approximate successive cancellation decoder. Section V gives the implementation results of the proposed approximate polar codes in the Xilinx



(a)



(b)

Figure 1. (8, 4) Polar encoder & decoder; (a) Encoder; and (b) Decoder.

Virtex-6 FPGA platform, along with the comparison to the prior architectures, and section VI concludes.

2. REVIEW OF POLAR CODES

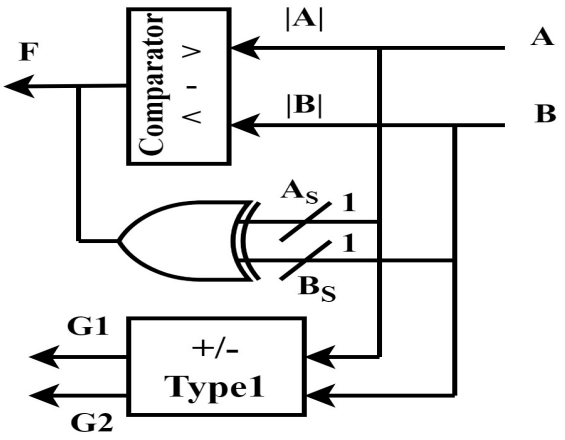
Polar codes are (n, k) linear block codes, where $n=2^N$ represents code length and k represents message bits, and are employed in binary discrete memoryless channels with various codeword lengths and code rates (k/n). Rooted in channel polarisation, polar codes effectively break down the wireless communication channel into numerous sub-channels, distinguishing them as reliable and unreliable channels as the code length 'n' approaches infinity. The transmitter and receiver use reliable channels to transmit information bits for low error rate communication. The unreliable channels transmit the frozen bits, i.e., '0'.

An (n, k) polar code is constructed using the generator matrix G, which is the Nth Kronecker product of matrix F, Where $F=\begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}$. The input vector $U=(U_0, U_1, U_2,\ldots, U_{n-1})$ is encoded to output vector $Y=(Y_0, Y_1, Y_2, Y_3,\ldots, Y_{n-1})$ using the generator matrix in Eqn. (1). The input vector U of the polar code is specified by the K element subset of $I \subset \{1, 2,\ldots, n\}$, which represents the information bits of U, and I^C represents the complementary set of I in $\{1, 2,\ldots, n\}$. If the ith bit of U belongs to I^C, then U_i is forced to Zero, a frozen bit. Eqn. (2) represents the generator matrix for N=3, and its equivalent encoding graph of the polar codes for a code length of (8, 4) is shown in Fig. 1.

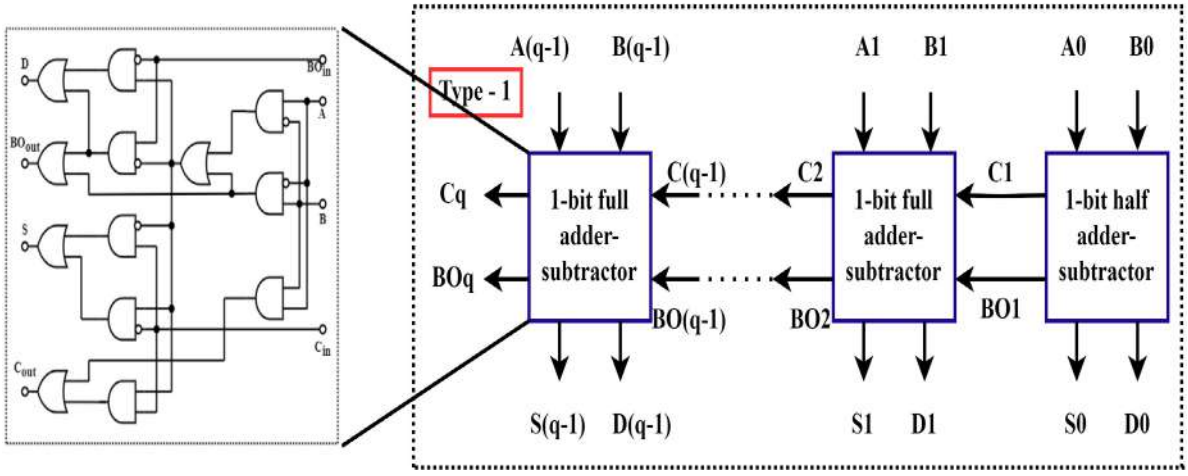
$$Y = U.\begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}^{\otimes N} \tag{1}$$

$$G = F^{\otimes 3} = \begin{bmatrix}1&0&0&0&0&0&0&0\\ 1&1&0&0&0&0&0&0\\ 1&0&1&0&0&0&0&0\\ 1&1&1&1&0&0&0&0\\ 1&0&0&0&1&0&0&0\\ 1&1&0&0&1&1&0&0\\ 1&0&1&0&1&0&1&0\\ 1&1&1&1&1&1&1&1\end{bmatrix} \tag{2}$$

Successive Cancellation (SC) decoder serially processes the logarithmic likelihood ratios (LLRs) received through the channel at the receiver. The decoding graph of the SC decoder is shown in Fig. 1. The decoder operates in $m=log_2 n$ stages and decodes one bit per cycle. The output information bit is decoded based on the 'X' value in Fig. 1, which depends on the LLR computed in the previous stage. If X is ≥ 0, then the information bit is to be decoded as '0'; if X is <0, then the information bit is decoded as '1'. Independent of the X value, if the bit position is frozen, then the bit is decoded as '0', as represented in Eqn. (3).



(a)



(b)

Figure 2. (a) Merged processing unit; and (b) Type -1 processing unit.

$$U_i = \begin{cases} 0, & if \quad X \geq 0 \\ 0, & if \quad Frozen \\ 1, & if \quad X < 0 \end{cases} \tag{3}$$

SC decoder mainly consists of two functional elements, namely F and G. Both functional units operate on the logarithmic likelihood ratio (LLR) domain, and they are formulated as given in Eqn. (4) and Eqn. (5), respectively.

$$F(LLR(Y_1), LLR(Y_2)) = \{Sign(LLR(Y_1)).Sign(LLR(Y_2)), \min(|LLR(Y_1)|, |LLR(Y_2)|)\} \tag{4}$$

$$G(LLR(Y_1), LLR(Y_2)) = LLR(Y_1)(-1)^{U_{Sum}} + (LLR(Y_2) \tag{5}$$

The merged processing unit (MPU) is the amalgamation of the functional units F and G, used to calculate the Eqn. (4) and Eqn. (5) Y1 and Y2 are two Q-bit inputs. Y_s and |Y| represent the sign bit and magnitude, respectively. Output F is computed as the minimum of the two LLRs received from the channel or previous stages and concatenates with the XOR of signs of both the LLRs, representing the Eqn. (4). Eqn. (5) can be summarized as $G_1=Y_1+Y_2$ if $U_{Sum}=0$, $G_1=Y_2-Y_1$ if $U_{Sum}=1$. U_{Sum} is the resultant XOR operation of all the earlier decoded bits. The architecture representation of the MPU is depicted in Fig. 2(a), and the Type 1 processing unit used to calculate G_1 and G_2 is shown in Fig. 2(b). The critical path delay for the exact 1-bit full adder subtractor unit from Fig. 2(b) is given in Eqn. (6).

$$T_{Critical\ path\_EASU} = 2.T_{AND} + 2.T_{OR} + 2.T_{NOT} \tag{6}$$

3. PROPOSED APPROXIMATE COUNTERPARTS FOR POLAR DECODING

3.1 Approximate Comparator

In the context of two magnitudes, each consisting of n bits, A and B, under certain conditions, it is possible to determine the smaller magnitude by exclusively examining their binary representations from the MSB (Most Significant Bits) to the LSB (Least Significant Bits). It is important to note that many comparisons can be efficiently conducted by focusing only on a few bits near the MSB. This observation inspires the introduction of an approximate comparator replacing the exact comparator in (m-1) stages. For Q-bit quantization, the approximator comparator disregards the least significant k bits of both A and B during the comparison process.

The MSB bits of both inputs, $A(A_{q-1}, A_{q-2}, \ldots, A_k)$ and $B(B_{q-1}, B_{q-2}, \ldots, B_k)$, represented in Fig. 3(a) are compared, and then, from the result, the min of both inputs is chosen. That is, if $A(A_{q-1}, A_{q-2}, \ldots, A_k) \geq B(B_{q-1}, B_{q-2}, \ldots, B_k)$, then the comparator generates a high value, then the minimum value is chosen as B, and if the comparator value is low then the minimum value is chosen as A.

However, the error occurs in the worst-case condition when the MSB bits of both inputs are the same. That is in condition of if $A(A_{q-1}, A_{q-2}, \ldots, A_k) = B(B_{q-1}, B_{q-2}, \ldots, B_k)$, and $A(A_{k-1}, A_{k-2}, \ldots, A_0) < (B_{k-1}, B_{k-2}, \ldots, B_0)$, then the approximate comparator anticipates as $A \geq B$ but actually it is $A < B$. The approximate comparator is efficient in all conditions except the one presented above. To interpret the accuracy of the approximate comparator, the relation between the error rate and the number of bits discarded (k) in the worst-case condition is presented below.

Assuming two random numbers with uniform distribution, if $P(A_i=B_i)=1/2$, for $i=0, 1,\ldots,q-2, q-1$. Then $P(A(A_{q-1},A_{q-2},\ldots,A_k)=B(B_{q-1},B_{q-2},\ldots,B_k))=(1/2)^{q-k}$, and $P(A(A_{k-1},A_{k-2},\ldots,A_0)< B(B_{k-1},B_{k-2},\ldots,B_0))=(1-(1/2)^k)/2=(2^k-1)/(2^{k+1})$. In this, the maximum number of errors introduced by discarding k bits is 2^k-1. Finally, the accuracy of the approximate comparator in terms of error rate is given in Eqn. (7) as follows.

$$Error\ Rate = \left(\frac{1}{2}\right)^{q-k} \cdot \frac{2^k-1}{2^{k+1}} = \frac{2^k-1}{2^{q+1}} \tag{7}$$

Discarding k LSB bits speeds up the comparison because the system processes fewer bits, hence increasing the efficiency by a factor of $\left(\frac{q}{q-k}\right)$. As the k increases, accuracy decreases, efficiency increases, and vice versa.

A comparator comparing two n-bit inputs has an overall computational complexity of $O(q)$. In the worst-case scenario, the approximate comparator reduces the computational complexity to $O(q\text{-}k)$.

3.2 Approximate Adder Subtractor Unit

In this paper, the AASU is designed by modifying certain output bits of the exact adder-subtractor unit. Altering certain bits from '1' to '0' and '0' to '1' minimizes the logic expression of the exact adder-subtractor unit. The carry and borrow bits of the AASU are not changed and are the same as the exact adder-subtractor unit. A few sum and difference bits are changed in the AASU compared to the conventional adder subtractor unit. The modified bits from the exact adder-subtractor unit are shown in the shaded part of Table 1. The logic "sum of product" expressions for the sum and difference from the table is given in Eqn. (8) and Eqn. (9), respectively.

Table 1. Truth table for approximate sum and approximate difference

A	B	C	Approximate sum	Approximate difference
0	0	0	**1**	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	**1**
1	0	0	1	**0**
1	0	1	0	0
1	1	0	**1**	0
1	1	1	**0**	1

$$Approximate\ Sum = \overline{A}\overline{B} + \overline{C} \tag{8}$$

$$Approximate\ Difference = Borrow \tag{9}$$

The approximate sum and difference unit reduces the number of gates needed compared to conventional sum difference units. The critical path delay of the approximate sum and difference circuits is shortened, increasing the speed of the operations. The critical path delay for the approximate 1-bit full adder subtractor unit from Fig. 3(b) is given in Eqn. (10).

$$T_{Critical\ Path\_AASU} = T_{AND} + 2.T_{OR} + T_{NOT} \tag{10}$$

The critical path delay and the number of logic gates for the approximate 1-bit full adder subtractor unit compared to

the exact 1-bit adder subtractor unit are decreased, resulting in increased throughput, and decrease in the area.

In the design, the performance and accuracy of the approximate adder subtractor unit depend significantly on the number and combination of sum/difference bits that are flipped to achieve a simplified Boolean expression. As we flip more bits, the complexity of the Boolean logic is reduced, leading to improved performance (e.g., lower latency and resource usage). However, this comes at the expense of accuracy, that is as the larger number of flipped bits increases the error increases. Conversely, when fewer bits are flipped, the accuracy of the adder subtractor unit is better preserved, but the performance gains are diminished. This trade-off requires careful consideration to achieve an optimal balance based on the application's specific requirements.

3.3 Approximate Merged Processing Unit

The Approximate Merged Processing Unit (AMPU) is designed using the approximate comparator and the approximate adder-subtractor unit, as depicted in Fig. 3(a). The type 2 processing unit showing the approximate addition subtraction is depicted in Fig. 3(b). The function unit F in MPU has an XOR gate for sign bits and a comparator for magnitude comparison. The design strategy of the function unit F in AMPU stems from the understanding that the output bit is primarily influenced by the sign bits of the logarithmic likelihood ratios, so meticulous attention is given to flushing the LSB bits of the comparator and preserving the XOR gate. The "approximate comparator" is designed as defined above by considering the MSB bits while ignoring the LSB bits.

In this work, an Approximate Adder-Subtractor Unit (AASU) is designed for the (m-1) stages of the Merged Processing unit. The addition and subtraction operations are performed in the MPU, and depending on the exclusive-or operation of the earlier decoded bits, only one of the LLRs of the adder or subtractor is chosen. So, the error occurs due to the computation of only the addition or subtraction processes, not both. Moreover, the error in the intermediate stages does not necessarily imply an error in the last stage, where the information bits are decoded. The final output vector decoded depends mainly on the sign bits of the LLRs obtained. AASU does not create any error in the sign bit because the carry or borrow bits of the adder or subtractor are not modified and are the same as the exact adder and subtractor. So, the error through the sign bit is not propagating to the further stages during the computations in the merged processing unit. To eliminate the errors further, only the most significant bit of the two inputs undergoes addition or subtraction using an exact

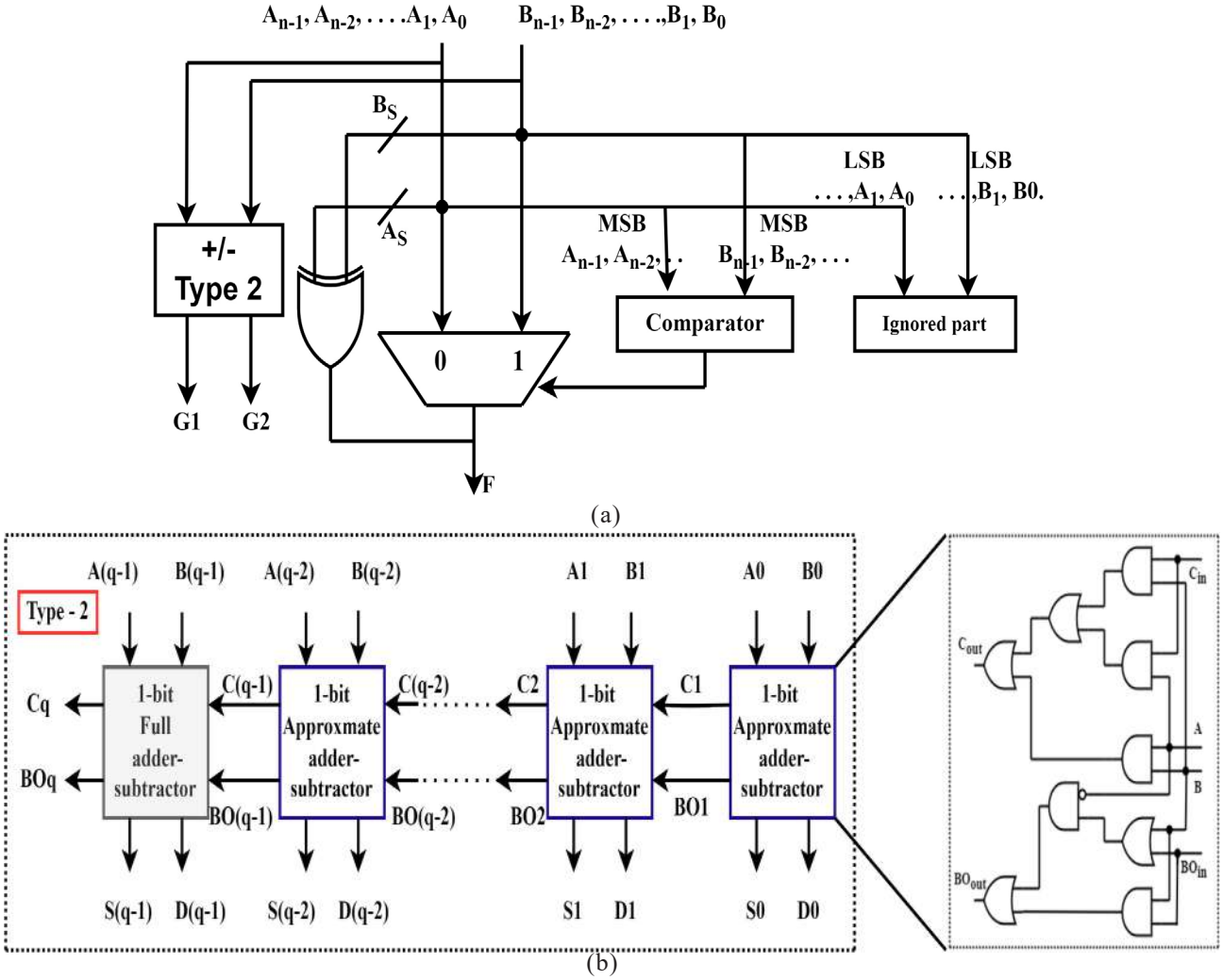


Figure 3. (a) Approximate merged processing unit; and (b) Type -2 processing unit.

adder-subtractor unit. This strategic approach enhances the accuracy of the overall system.

3.4 Approximate 2-bit Processing unit

The MPUs in the last stage of the decoder are replaced with an approximate 2-bit Processing Unit (A2bPU) to decode two output bits (U_{2i-1}, U_{2i}) simultaneously, per clock cycle. The approximate 2bPU is designed to mitigate the delay caused by the comparator block by excluding it, thereby inducing a negligible error in the output bit U_{2i}.

The working of A2bPU is initially associated with the frozen condition of the output bits (U_{2i-1}, U_{2i}). The signals designated as frozen1(F1) and frozen2(F2) are included to denote whether the particular bits are frozen or information bits. If the output bit U_{2i-1} is frozen, then F1 is set to one else, which will be zero. Similarly, depending on whether the output bit U_{2i} is frozen or not, F2 will be Zero or one. Additionally, the A2bPU relies on the sign bits of the input LLRs, represented as S1 and S2. These sign bits signify whether the corresponding LLR is positive (0) or negative (1). The comparator block, used in the exact processing unit, is eliminated to reduce the hardware complexity and critical path delay. Table 2 represents the output bits of A2bPU for all the combinations, and the corresponding Boolean expression for both the outputs is given in Eqn. (11) and Eqn. (12). The architecture of A2bPU is shown in Fig. 4 and its critical path delay is given in Eqn. (13).

Table 2. Truth table of Approximate 2-bit Processing Unit

F1	F2	S1	S2	U_{2i-1}	U_{2i}
0	0	0	0	0	0
		0	1	1	1
		1	0	1	0
		1	1	0	1
0	1	0	0	0	0
		0	1	1	
		1	0	1	
		1	1	0	
1	0	0	0	0	0
		0	1		1
		1	0		1
		1	1		1
1	1	0	0	0	0
		0	1		
		1	0		
		1	1		

$$U_{2i-1} = \overline{F1}(S1 \oplus S2) \tag{11}$$

$$U_{2i} = \overline{F2}(S2 + F1S1) \tag{12}$$

$$T_{Critical\ Path\_A2bPU} = T_{XOR} + T_{AND} + T_{OR} \tag{13}$$

The algorithmic representation of the proposed approximate successive cancellation decoder is given as follows.

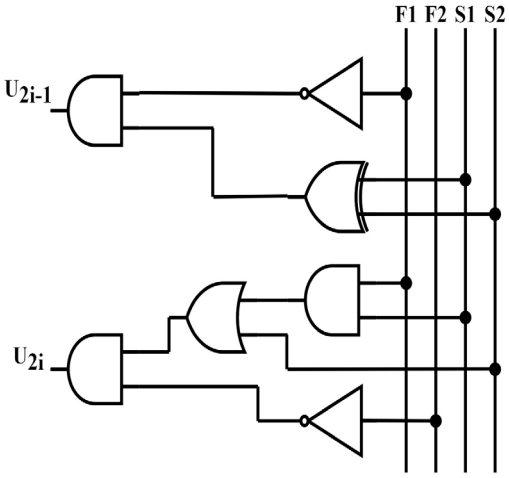


Figure 4. Approximate 2-bit processing unit.

Algorithm: Decoding algorithm for Approximate Successive Cancellation Decoder (ASCD)

1. **Inputs:** Logarithmic likelihood ratios $\{Y_1, Y_2, Y_3, \ldots, Y_n\}$, Frozen bits $F1, F2, \ldots, Fn$
2. $(m-1)$ Stages of Approximate Merged Processing Units which compute F and G functions
3. Last stage Approximate 2-bit Processing Unit
   Here F1 and F2 are frozen positions, S1 and S2 are sign bits of input LLRs to A2bPU
   a. *If* $F1=0$ *and* $F2=0$
      $U_{2i-1} = S1 \oplus S2,\ U_{2i} = S2$
   b. *If* $F1=0$ *and* $F2=1$
      $U_{2i-1} = S1 \oplus S2,\ U_{2i} = 0$
   c. *If* $F1=1$ *and* $F2=0$
      $U_{2i-1} = 0,\ U_{2i} = F1S1 + S2$
   d. *If* $F1=1$ *and* $F2=1$
      $U_{2i-1} = 0,\ U_{2i} = 0$
4. Selecting the G output based on the partial sum of two previously generated outputs
5. Recursive AMPU and A2bPU operations till all the outputs are decoded
6. **Outputs:** $U_1, U_2, \ldots, U_n$

The proposed approximate 2-bit processing unit gives erroneous output in the second decoding output U_{2i} of the A2bPU for only two combinations. For the frozen bits $F1$=1, $F2$=0, and sign bits $S1$=0, $S2$=1, or vice versa, the output depends on the comparator block, which is removed in the architecture of A2bPU, which causes errors in output bits. The combinations for which error occurs is as given: {$F1$, $F2$, $S1$, $S2$, C}=10100 and {$F1$, $F2$, $S1$, $S2$, C}=10011.

4. PROPOSED ASCD TOP – LEVEL ARCHITECTURE

The precomputation tree-based architecture achieves high throughput with low latency among the previous SC decoders[19]. This paper presents an approximate SC decoder (1024,512) based on the 2b-tree architecture using the approximate counterparts implemented in the AMPU till (m-1) stages and A2PU in the last stage. For the code length of 1024, there are (m-1)=9 stages of AMPU units, one last stage of A2bPU. The proposed architecture is for the quantization of 5-bits. The top-level architecture of our proposed approximate successive cancellation decoder with an AMPU and A2bPU is depicted in Fig. 5. The LLRs are input to the decoder, the bits indicating the frozen pattern, and the decoded output bits are stored in the memory. The AMPU combines the functionality of the functional units F and G. The approximate 2bPU decodes 2 bits per clock cycle. The partial sum generator[27-28] performs the exclusive-or operation of the earlier decoded output bits and is given to the internal stages.

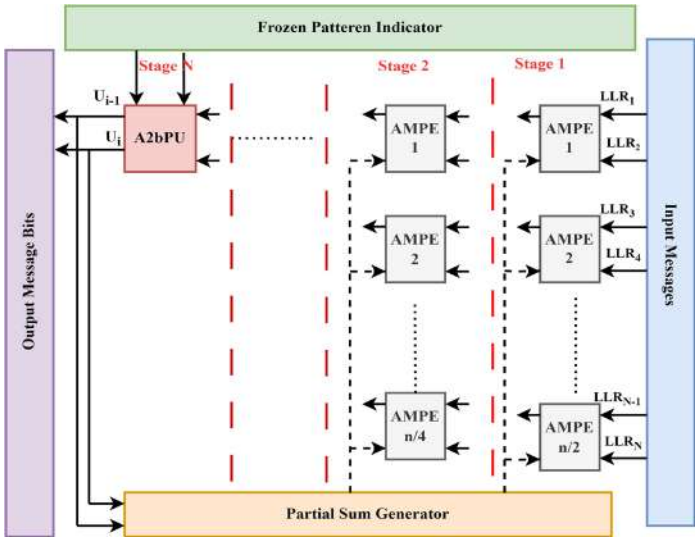


Figure 5. Top-level architecture of approximate SC decoder with A2bPU.

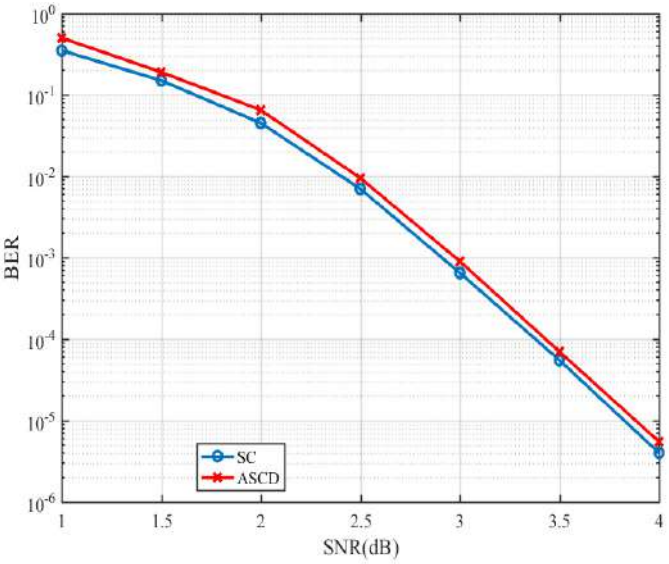
The function F in the AMPU is approximated using the approximate comparator only for the LSB. The MSB and the sign bits (XOR operation) that are the primary source for the correct output are not approximated. This results in minimal error due to functional unit F in AMPU. Functional unit G is approximated using the approximate adder-subtractor for all the quantised bits except the one most significant bit, which is the primary source for correct output. Moreover, the carry and borrow bits are not changed in the AASU, which implies that the error is not propagated to the higher-order bits.

So, the error incurred by the AASU in AMPU is also negligible, which is tolerable. Using this approximate comparator, AASU in the AMPU reduces the path delay compared to the conventional MPU, thus reducing the latency. The number of gates is also reduced using these approximate units in AMPU. The designed A2bPU reduces the hardware eliminating the comparator which is used only for two combinations of input bits. This approximation causes errors in the 2nd decoding output of A2BPU which is negligible. We can get information from negligible error outputs in error-resilient applications like multimedia. In such applications of 5G, we use this approximate SC decoder for polar codes.

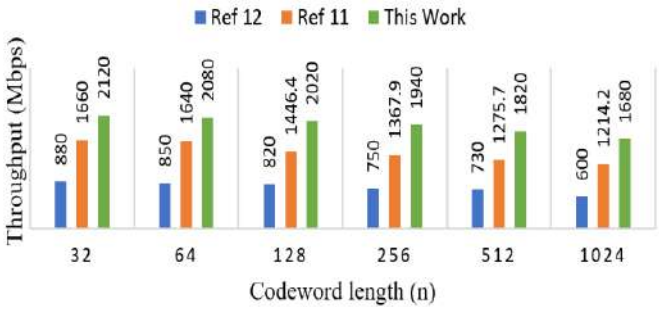
5. RESULTS AND DISCUSSION

The approximate polar codes described in this paper are implemented using Verilog HDL in the Xilinx Vivado 2020.1 environment. The FPGA implementation results for the proposed decoders are evaluated for various code lengths on the Xilinx Virtex-6 FPGA platform. The proposed decoders are designed using tree architecture with the precomputation technique and are synthesized for a code length of (1024, 512) with a code rate of 0.5.

The Quantisation scheme used in the proposed approximate successive cancellation decoder is represented as Q (Q_c, Q_i), where Q_c represents bits employed for channel logarithmic likelihood ratios, and Q_i represents the LLR values for the internal stages. In this work, we select the Quantization scheme as Q (5, 5), meaning a total of 5 bits represent the LLR values for both internal stages and channel values. Fig. 6(a) represents the BER performance of the ASCD compared with the conventional SC decoder. The figure shows clearly that the error incurred in the proposed approximate decoder is minimal and tolerable in the error-resilient application of 5G-like video broadcasting.



(a)



(b)

Figure 6. Results: (a) The error correction performance of ASCD, conventional SCD; and (b) Throughput comparison.

Table 3. FPGA implementation results

Block length	This work			Reference 11			Reference 12		
	LUT	Registers	RAM (Bits)	LUT	Registers	RAM (Bits)	LUT	Registers	RAM (Bits)
32	1026	384	224	1611	742	0	2266	568	416
64	2524	782	448	3921	1480	0	5724	1166	832
128	6635	1584	896	8926	2985	0	13882	2211	1664
256	13870	3468	1792	20259	6058	0	31678	5144	3328
512	32685	7436	3584	45986	12334	0	77948	9367	6656
1024	73797	15472	7168	102740	25139	0	190127	22928	13312

The throughputs in the FPGA drop gradually as the code lengths increase, as represented in Fig. 6(b). The throughput of the ASCD is improved by 27 % and doubles compared to Ref[11] and Ref[12], respectively. Using the approximate techniques in the ASCD improves the hardware architecture and throughput with minimal degradation in the bit error rate performance.

As the code length increases routing becomes more intricate in a combinational decoder, primarily due to the use of more logic blocks. Consequently, interconnect delay escalates with the increased code length. Combinational decoders built with logic blocks devoid of storage elements and control circuits. This design choice effectively reduces the delay associated with the decoder's maximum path by eliminating delays inherent in read/write operations, setup/hold periods, and complex processing units. The absence of these operations in the internal stages of the decoder leads to low power consumption. RAM blocks in the proposed decoder store the input LLRs, frozen bits, and output bits of the decoder. Registers in this decoder are used for the storage of partial sums and retrieving RAM outputs.

As Table 3 outlines, comparing the proposed combinational decoder for N = 1024 with state-of-the-art decoders provides insights into decoder performance. The LUT count of the proposed approximate SC decoder is reduced by 28 % and 61 % compared to Reference 11 and Reference 12, respectively. RAM count is reduced by 46 % compared to Refence 12, and the number of registers used is reduced by 38 % and 32 %, respectively.

6. CONCLUSIONS

This work presents a novel hardware-efficient ASCD architecture using approximate computing techniques. The proposed ASC decoder uses an Approximate Merged Processing Unit and an Approximate 2-bit Processing Unit, replacing the functional units F and G in the conventional decoder architecture. The functional unit F in AMPU is designed using the approximate comparator for lower significant bits without replacing the most significant bits and the XOR gate, which are crucial for accurate output determination. Functional unit G in AMPU is modified by using the approximate adder-subtractor unit for all the quantized bits except for one last MSB bit, and the last stage of the decoder architecture is replaced with A2bPU, which reduces the delay by decoding 2 bits per clock cycle. ASCD is implemented targeting the Xilinx Virtex-6 FPGA platform. Compared with the existing successive cancellation decoders, the ASCD achieves an average of 44 % reduction in LUT count. The RAM bits are reduced by 46 %, and registers are reduced by 35 % on average. The BER performance of the ASCD indicates a minimal performance degradation and can be tolerable in error-resilient 5G applications.

REFERENCES

1. Arikan, E. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transact. Inf. Theory.*, 2009, **55**(7), 3051-3073. doi: 10.1109/TIT.2009.2021379.
2. Bioglio, V.; Condo, C. & Land, I. Design of polar codes in 5G new radio. *IEEE Commun. Surveys & Tutorials.,* 2020, **23**(1), 29-40. doi:10.1109/COMST.2020.2967127.
3. Giard, P.; Balatsoukas, S.A.; Sarkis, G.; Thibeault, C. & Gross, W.J. Fast low-complexity decoders for low-rate polar codes. *J. Signal Process. Syst.*, 2018, **90**, 675-685. doi: 10.1007/s11265-016-1173-y.
4. Giard, P.; Sarkis, G.; Thibeault, C. & Gross, W.J. Multi-mode unrolled architectures for polar decoders. *IEEE Transact. Circuits and Syst. I: Regular Papers.*, 2016, **63**(9), 1443-1453. doi:10.1109/TCSI.2016.2586218.
5. Krasser, F.G.; Liberatori, M.C.; Coppolillo, L.; Arnone, L. & Moreira, J.C. Fast and efficient FPGA implementation of polar codes and SoC test bench. *Microprocessors and Microsyst.,* 2021, **84**, 104264. doi: 10.1016/j.micpro.2021.104264.
6. Sarkis, G.; Giard, P.; Vardy, A.; Thibeault, C. & Gross, W.J. Fast polar decoders: Algorithm and implementation. *IEEE J. Selected Areas in Commun.*, 2014, **32**(5), 946-957. doi: 10.1109/JSAC.2014.140514.
7. Shrestha, R. & Sahoo, A. High-speed and hardware-efficient successive cancellation polar-decoder. *IEEE Transact. Circuits and Syst. II: Express Briefs.*, 2018, **66**(7), 1144-1148. doi: 10.1109/TCSII.2018.2877140.
8. Zhang, C. & Parhi, K.K. Latency analysis and architecture design of simplified SC polar decoders. *IEEE Transact. Circuits and Syst. II: Express Briefs.*, 2013, **61**(2), 115-119. doi: 10.1109/TCSII.2013.2291065.
9. Yoon, H.Y & Kim, T.H. Efficient successive-cancellation polar decoder based on redundant LLR representation. *IEEE Transact. Circuits and Syst. II: Express Briefs.*, 2018, **65**(12), 1944-1948. doi: 10.1109/TCSII.2018.2811378.
10. Yun, H.R. & Lee, H. Simplified merged processing element for successive-cancellation polar decoder. *Electron. Letters.*, 2016, **52**(4), 270-272. doi:10.1109/LCOMM.2011.101811.111480.
11. Rezaei, H.; Rajatheva, N. & Latva-aho, M. High-throughput rate-flexible combinational decoders for multi-kernel polar codes. *IEEE Transact. Circuits and Syst. I: Regular Papers.*, 2023, **70**(11), 4492-4504. doi: 10.1109/TCSI.2023.3311514.
12. Dizdar, O. & Arıkan, E. A high-throughput energy-efficient implementation of successive cancellation decoder for polar codes using combinational logic. *IEEE Transact. Circuits and Syst. I: Regular Papers.*, 2016, **63**(3), 436-447. doi: 10.1109/TCSI.2016.2525020.
13. Korada, S.B.; Şaşoglu, E. & Urbanke, R. Polar Codes: Characterisation of exponent, bounds, and constructions. *IEEE Transact. Inf. Theory.*, 2010, **56**(12), 6253-6264. doi: 10.1109/TIT.2010.2080990.
14. Mori, R. & Tanaka, T. Performance of polar codes with the construction using density evolution. *IEEE Commun.*

Letters., 2009, **13**(7), 519-521.
doi: 10.1109/LCOMM.2009.090428.

15. Tal, I. & Vardy, A. How to construct polar codes. *IEEE Transact. Inf. Theory.,* 2013, **59**(10), 6562-6582.
doi: 10.1109/TIT.2013.2272694.
16. Leroux, C.; Raymond, A.J.; Sarkis, G. & Gross, W.J. A semi-parallel successive-cancellation decoder for polar codes. *IEEE Transact. Signal Process.*, 2012. **61**(2), 289-299.
doi: 10.1109/TSP.2012.2223693.
17. Roy, J.S.; Lakshminarayanan, G. & Ko, S.B. High-speed architecture for successive cancellation decoder with split-g node block. *IEEE Embedded Syst. Lett.*, 2020, **13**(3), 118-121.
doi: 10.1109/LES.2020.3021144.
18. Hassan, H.G.; Hussien, A.M & Fahmy, H.A. A simplified radix-4 successive cancellation decoder with partial sum lookahead. *AEU-Int. J. Electron. Commun.*, 2018, **96**, 267-272.
doi: 10.1016/j.aeue.2018.09.027.
19. Yuan, B. & Parhi, K. Low-latency successive-cancellation polar decoder architectures using 2-Bit decoding. *IEEE Transact. Circuits and Syst. I: Regular Papers.*, 2014, **61**(4), 1241-1254.
doi: 10.1109/TCSI.2013.2283779.
20. Fan, Y.; Xia, C.; Chen, J.; Tsui, C.Y.; Jin, J.; Shen, H. & Li, B. A low-latency list successive-cancellation decoding implementation for polar codes. *IEEE J. Selected Areas in Commun.*, 2015, **34**(2), 303-317.
doi: 10.1109/JSAC.2015.2504318.
21. Xia, C.; Chen, J.; Fan, Y.; Tsui, C.Y.; Jin, J.; Shen, H. & Li, B. A high-throughput architecture of list successive cancellation polar codes decoder with large list size. *IEEE Transact. Signal Process.*, 2018, **66**(14), 3859-3874.
doi: 10.1109/TSP.2018.2838554.
22. Abbas, S.M.; Fan, Y.; Chen, J. & Tsui, C.Y. High-throughput and energy-efficient belief propagation polar code decoder. *IEEE Transact. Very Large-Scale Integration (VLSI) Syst.*, 2016, **25**(3), 1098-1111.
doi: 10.1109/TVLSI.2016.2620998.
23. Yuan, B. & Parhi, K.K. Early stopping criteria for energy-efficient low-latency belief-propagation polar code decoders. *IEEE Transact. Signal Process.*, 2014, **62**(24), 6496-6506.
doi: 10.1109/TSP.2014.2366712.
24. Kam, D.; Yoo, H. & Lee,Y. Ultralow-latency successive cancellation polar decoding architecture using tree-level parallelism. *IEEE Transact. Very Large Scale Integration (VLSI) Syst.*, 2021. **29**(6), 1083-1094.
doi: 10.1109/TVLSI.2021.3068965.
25. Shih, X.Y.; Tsai, J.H.; Li, B.X. & Huang, C.P. Reconfigurable hardware architecture of area-efficient multimode Successive Cancellation (SC) decoder. *IEEE Transact. Circuits and Syst. II: Express Briefs.*, 2022. **69**(4), 2291-2295.
doi: 10.1109/TCSII.2022.3144435.
26. Mouhoubi, O.; Nour, C. A. & Baghdadi, A. Latency and complexity analysis of flexible semi-parallel decoding architectures for 5G NR polar codes. *IEEE Access.*, 2022. **10**, 113980-113994.
doi: 10.1109/ACCESS.2022.3216292.
27. Zhou, Y.; Zheng, Y. & Wang, Z. Fast successive-cancellation decoding of 5G parity-check polar codes. *IEEE Commun. Lett.*, 2023. **27**(1), 37-40.
doi: 10.1109/LCOMM.2022.3217163.
28. Zhou, Y.; Chen, Z.; Lin, J. & Wang, Z. A high-speed successive-cancellation decoder for polar codes using approximate computing. *IEEE Transact. Circuits and Syst. II: Express Briefs.*, 2018, **66**(2), 227-231.
doi: 10.1109/TCSII.2018.2847441.
29. Xu, M.; Jing, S.; Lin, J.; Qian, W.; Zhang, Z.; You, X. & Zhang, C. Approximate belief propagation decoder for polar codes. *In* 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Calgary, AB, Canada, 2018.
doi: 10.1109/ICASSP.2018.8462478.
30. Yuan, B. & Parhi, K.K. Successive cancellation decoding of polar codes using stochastic computing. 2015 IEEE International Symposium on Circuits And Systems (ISCAS): Lisbon, Portugal, 2015.
doi: 10.1109/ISCAS.2015.7169328.
31. Fan, Y. & Tsui, C.Y. An efficient partial-sum network architecture for semi-parallel polar codes decoder implementation. *IEEE Transact. Signal Process.*, 2014, **62**(12), 3165-3179.
doi: 10.1109/TSP.2014.2319773.
32. Berhault, G.; Leroux, C.; Jego, C. & Dallet, D. Partial sums generation architecture for successive cancellation decoding of polar codes. In SiPS 2013 Proceedings: Taipei, Taiwan, 2013.
doi: 10.1109/SiPS.2013.6674541.

CONTRIBUTORS

Mrs Jali Nandini obtained MTech in VLSI and Embedded Systems from Kakatiya Institute of Technology and Science, Warangal, India. Her research interests include: VLSI architectures for forward error correction coding for wireless communication systems.
In the current study she designed the overall architecture of the proposed approximate successive cancellation decoder and is responsible for writing and drafting the manuscript.

Prof P. Muralidhar obtained his PhD degree from NIT Warangal and currently he is working in the ECE Department at NIT Warangal. His research interests include: Design of embedded systems and VLSI architectures for video processing systems. In the current study he contributed towards the selection of approximate adder-subtractor units.

Prof. Sreehari Rao Patri obtained his PhD in Mixed-signal IC design from the NIT, Warangal and currently working as a Professor at the Department of Electronics and Communications Engineering at NIT Warangal. His research interests include: Analog/mixed signal IC design, power management IC design, RF IC design, On-chip smart sensors, and communication systems. In the current study he contributed towards designing the merged processing unit.