# HARDWARE IMPLEMENTATION OF SUCCESSIVE CANCELLATION DECODER FOR POLAR CODE

D.Dineshkumar
Asst. Professor, Department of ECE
Mepco Schlenk Engineering College
Sivakasi, India
dineshddk@mepcoeng.ac.in

E.Bhuvaneshwari
PG Student, Department of ECE
Mepce Schlenk Engineering College
Sivakasi, India
bhuvaneshwari951995@gmail.com

***Abstract*— Polar codes are the first error correcting codes to achieve the symmetric capacity in binary input discrete memory less channels. The successive cancellation algorithm enables very low complexity implementations in hardware, due to the regular structure exhibited by polar codes. This paper describes pipelined tree successive cancellation decoder architecture. The successive cancellation decoding can be implemented using addition and subtraction operations, thereby eliminating costly multiplication and division operations and reducing the complexity of each processing element greatly. The complexity of both encoding and SC decoding of polar codes are O(NlogN),where N is the code block length. This architecture is implemented in Spartan6 Field Programmable Gate Array (FPGA), in which the area is reduced.**

***Keywords—Polar code, Processing elment, Successive cancellation decoding, FPGA.***

## I. INTRODUCTION

Polar codes, as the first provable capacity achieving codes over binary-input discrete memoryless channel (B-DMC) [1], have received significant attention among various forward error correction (FEC) codes. Due to their explicit structure and low-complexity encoding and decoding scheme, polar codes have emerged as one of the most important codes in coding theory. Polar codes achieve channel capacity asymptotically in the code length N—when the underlying channel is memoryless and has a discrete input alphabet [2]. Moreover, in some information theoretic applications, such as achieving the secrecy capacity of the wiretap channel in the general case, polar codes are the only known solution which is both explicit and efficient [3]. From a practical point of view, however, polar codes come close to achieving the channel capacity only for very large code lengths. Compare to the prior best channel codes, polar codes can potentially outperform low density parity check (LDPC) codes in terms of error correcting performance with the similar code length [4]. The FPGA implementation of polar decoder based on the Belief propagation (BP) algorithm was reported [5]. Although BP decoder has particular advantages in parallel design, due to the requirement of large number of processing elements (PEs), the BP decoder is not attractive for practical applications. In [6], [7], a precomputation scheme was applied to the SC algorithm, which succeeded in reducing the overall latency from (2n-2) to (n-1). However, considering the penalty of increased hardware, the SC-precomputation decoder does not show significant improvement with respect to hardware efficiency.

The rest of the paper is organized as follows. Related works on polar encoding design have been discussed in section 2. Section 3 discusses the details of successive cancellation decoder architecture. The experimental result has been discussed in section 4. Section 5 concludes the paper with the summary of the carried out work.

## II. POLAR CODE

Polar codes are linear block codes of size $N = 2^n$, n being a positive integer. In [1], Arıkan defined a construction based on a 2x2 binary matrix, denoted as the kernel of the code: $F=\begin{bmatrix}1 & 0\\ 1 & 1\end{bmatrix}$. The generator matrix of the code is a submatrix of the $n^{th}$ kronecker power of F, denoted $F^{\otimes n}$, where $\otimes$ is kronecker product. The generator matrix G is given for code word N is eight is given (1).

$$\begin{pmatrix}X_0\\X_1\\X_2\\X_3\\X_4\\X_5\\X_6\\X_7\end{pmatrix}=\begin{bmatrix}1&0&0&0&0&0&0&0\\1&1&0&0&0&0&0&0\\1&0&1&0&0&0&0&0\\1&1&1&1&0&0&0&0\\1&0&0&0&1&0&0&0\\1&1&0&0&1&1&0&0\\1&0&1&0&1&0&1&0\\1&1&1&1&1&1&1&1\end{bmatrix}\begin{pmatrix}d_0\\d_1\\d_2\\d_3\\d_4\\d_5\\d_6\\d_7\end{pmatrix} \quad (1)$$

The matrix represents the polar encoding structure. The encoding is simple XOR operations. The polar encoding architecture is shown Fig.1. The encoding architecture is computed from generator matrix. The symbol $\oplus$ denotes the XOR operation. An efficient polar based transmitter can be constructed based on the following principles: 1) sending required information bits at "good" positions, which can strongly guarantee the reliability of transmission; and 2) sending fixed "0" at "bad" positions, since after the transmission any decoded bits at these "bad" positions are highly unreliable. Those "0" bits are called "frozen" bits since these are

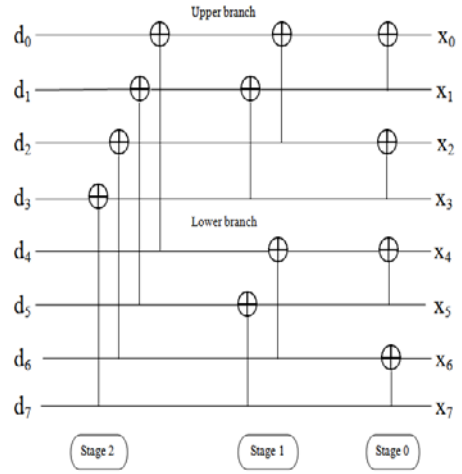


**Fig.1 Encoding architecture for polar code**

fixed and their positions are known at both the encoder and the decoder. The encoder output is bit reversal order.

## III. Successive Cancellation Decoder Architecture

### A. Processing element

The processing element is the main arithmetic component of the successive cancellation decoder [8]. The processing element architecture is shown in Fig.2.The processing element consists of f unit and g unit. The f unit and g unit are used to calculate the propagated LLR values. Here S to C is the block that performs the conversion from sign-magnitude form to 2's complement form, while C to S unit carries out the inverse conversion. Additionally, adder and subtractor are employed to carry out addition and subtraction between the two inputs. Finally, at the output end of the processing element, control signal is used to determine the outputs, which is propagated to the next stage. The control input is generated by partial sum product unit.

### B. Successive cancellation decoder

The Successive Cancellation decoder (SCD) is basic decoder for polar codes. The SCD is more advanced and efficient decoder. For a code of length ($N = 2^n$), after being sent over the transmission channel, the noisy version Y of the codeword X is received [9]. These LLRs are denoted $\lambda_i$, with $0 \leq i \leq N-1$. The decoder successively estimates every bit $u_i$ based on the channel observation vector ($\lambda_0^{N-1}$) and the previously estimated bits ($\widehat{u_0}^{i-1}$). The decoding process of a N = 2 polar code can be summarized as follows

$$\begin{pmatrix} f(L_1,L_2) \\ g(L_1,L_2) \end{pmatrix} = \begin{pmatrix} \frac{L_1 L_2 + 1}{L_1 + L_2} \\ L_1 L_2 or L_2 / L_1 \end{pmatrix} \quad (2)$$

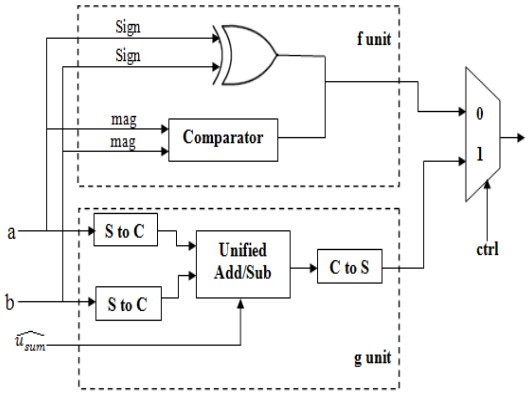


**Fig.2 Processing element architecture**

$$\begin{pmatrix} f(L_1,L_2) \\ g(L_1,L_2) \end{pmatrix} = \begin{pmatrix} \ln f(e^{l_1},e^{l_2}) \\ \ln g(e^{l_1},e^{l_2}) \end{pmatrix} \approx \begin{pmatrix} \ln(1+\exp(l_1+l_2)/\exp(l_1)+\exp(l_2)) \\ l_2+l_1 \; or \; l_2-l_1 \end{pmatrix} \quad (3)$$

$$\begin{pmatrix} f(L_1,L_2) \\ g(L_1,L_2) \end{pmatrix} = \begin{pmatrix} sign(L_1)sign(L_2)min\{|L_1|,|L_2|\} \\ L_2+(-1)^u L_1 \end{pmatrix} \quad (4)$$

The control input is determined according to (5).

$$\widehat{u_i} = \begin{cases} 0 & if \lambda_{i,o} > 0 \\ 1 & otherwise \end{cases} \quad (5)$$

### C. Pipelined Successive Cancellation decoder

The pipelined SC decoder has three stages. The first stage has four processing element. The second stage has two processing element. The third stage has one processing element. A processing element is a configurable element that can perform either function f or g. It also includes the computation block that updates the value with the last decoded bit. Compared to the butterfly-based structure, the pipelined tree architecture performs the same amount of computation with the same scheduling (see Table 1) but with a smaller number of processing elements and registers. The pipelined SC decoder complexity is,

$C_{pipe} = (n-1)(C_{PE}+C_r)+nC_r$ (6)

where $C_{PE}$ is processing element complexity, $C_r$ is complexity of the memory register.

TABLE 1 Schedule for the pipelined SCD

| CC | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $S_2$ | f | | | | | | | g | | | | | | | |
| $S_1$ | | f | | | g | | | | f | | | | g | | |
| $S_0$ | | | f | G | | f | g | | | f | f | G | | f | G |
| $\widehat{u_i}$ | | | $\widehat{u_0}$ | $\widehat{u_1}$ | | $\widehat{u_2}$ | $\widehat{u_3}$ | | | | $\widehat{u_4}$ | $\widehat{u_5}$ | | $\widehat{u_6}$ | $\widehat{u_7}$ |

In addition to the lower complexity, one can notice that the routing network in the decoder is much simpler in the tree architecture than in the butterfly-based structure. Connections

between processing elements are also local. The g unit is computed after f

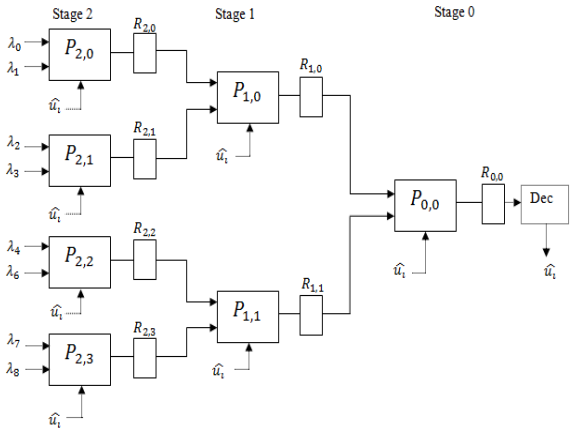


**Fig.3 Pipelined Successive Cancellation Decoder**

unit. The pipelined successive cancellation decoder architecture is shown in Fig.3. When stage *l* is enabled, to indicate which function (f or g) is applied to the $2^l$ activated nodes at stage $S_l$ during each clock cycle(CC). Every generated variable is used twice during the decoding. For example, the four variables generated in stage 2 at CC#1 are consumed on CC#2 and CC#5 in stage 1. This means that, in stage 2, the four registers associated with the f function can be reused at CC #8 to store the four data values generated by the g function.

## IV. Results and Discussion

The designed decoder architecture is implemented in FPGA. The pipelined decoder has processing element, matrix generation unit and partial sum computation unit. The matrix generator is to generate the generator matrix for to compute the partial sum product. The matrix generator circuit has XOR gate and D flip flops. The partial sum product unit is to generate the control input for processing element. The simulation output of processing element is shown in Fig.4.The simulation output of pipelined successive cancellation decoder is shown in Fig.5. The synthesis report gives the efficient usage of number of flip-flops used and bonded IOBs, number of slices used. The Xilinx summary report of decoder architecture is shown in Table 2.

TABLE 2 Device Utilization Summary of Pipelined SC Decoder

| Slice logic utilization | Used | Available | Utilization |
|---|---|---|---|
| Number of Slice Registers | 46 | 18,224 | 1% |
| Number of Slice LUTs | 149 | 9,112 | 1% |
| Number used as logic | 149 | 9,112 | 1% |
| Number of occupied Slices | 50 | 2,278 | 2% |
| Number with an unused Flip Flop | 105 | 149 | 70% |
| Number of fully used LUT-FF pairs | 44 | 149 | 29% |

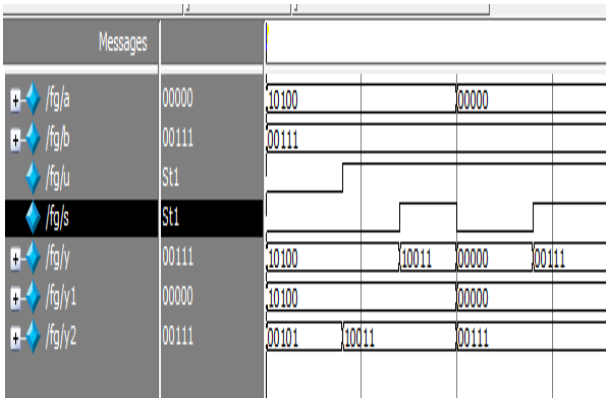


**Fig.4 Simulation output of processing element**

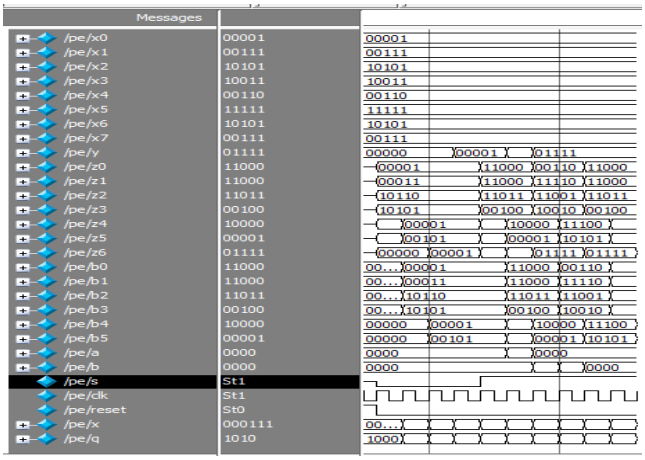


**Fig.5 Simulation output of pipelined successive cancellation decoder**

## V. Conclusion and Future Work

In this paper, the pipelined SC decoding architecture is designed. The designed architecture has significant advantage to reduce the hardware complexity. The pipelined SC decoder is designed utilizing the designed processing element, matrix generation unit and partial sum computation unit. The designed pipelined SC decoder is to reduce the processing element. The simulation of polar encoder and decoder is coded by Verilog HDL and is verified using modelsim. The designed system was implemented using Spartan 6. The results are obtained and analyzed with respect to the number of look up table (LUT). The future work includes implementing the Successive Cancellation List decoder for polar decoder. The SC list decoder is to reduce the area and latency.

## *Acknowledgment*

The authors wish to thank the Management and Principal of Mepco Schlenk Engineering College, for their support in carrying out this research work.

## *References*

[1] E. Arikan, "Channel polarization: A method for constructing capacityachieving codes for symmetric binary-input memoryless channels," IEEE Trans. on Inform. Theory, vol. 55, no. 7, pp. 3051 – 3073, Jul. 2009.

[2] I. Tal and A. Vardy, "How to construct polar codes," submitted to IEEE Trans. Inform. Theory, available online as arXiv:1105.6164v2,2011.

[3] ]E. Sasoglu, E. Telatar, and E. Arikan, "Polarization for arbitrardiscrete memoryless channels," in Proc. IEEE Information Theory Workshop ITW 2009, 2009, pp. 144–148.

[4] H. Mahdavifar and A. Vardy, "Achieving the secrecy capacity of wiretap channels using polar codes," in IEEE ISIT 2010, Jun.2010, pp. 913 –917.

[5] I.Tal and A.Vardy, "List decoding of polar codes," arXiv:1307.7154v2,2013.

[6] A. Pamuk, "An FPGA implementation architecture for decoding of polar codes," in Proc. 8th Int. Symp. on Wireless Commun.Syst. (ICWCS), Nov. 2011, pp. 437–441.

[7] C. Zhang, B. Yuan, and K. K. Parhi, "Reduced-latency SC polar decoder architectures," in Proc. Int. Conf. Commun., June 2012, pp. 3471–3475.

[8] C. Zhang and K. K. Parhi, "Low-latency sequential and overlapped architectures for successive cancellation polar decoder," IEEE Trans. Signal Processing, vol. 61, no. 10, pp. 2429–2441, May 2013.

[9] G. Berhault, C. Leroux, C. Jego, D.Dallet ,"Partial Sums Generation Architecture for Successive Cancellation Decoding Of Polar Codes,"arXiv:1309.7818v1,2013

[10] C.Leroux, J. Raymond, G.Sarkis, "Hardware Implementation of Successive Cancellation Decoders for Polar codes", Journal of Signal Processing System, 2012.