ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Implementation of Parallel Encoder and Decoder for Long Polar Codes

Ms.S.Anusha
M.Tech (VLSI),
CMR Institute of Technology,
Hyderabad, India.

Mr.P.Nagaraja Kumar
Assistant Professor,
CMR Institute of Technology,
Hyderabad, India.

Ms.V.Sumathi
Assistant Professor,
CMR Institute of Technology,
Hyderabad, India.

Abstract:

Due to the channel accomplishing property, the polar code has turned out to be a standout amongst the most positive blunder amending codes. As the polar code accomplishes the property asymptotically, in any case, it ought to be sufficiently long to have a decent mistake amending performance. In spite of the fact that the past completely parallel encoder is instinctive what's more, simple to actualize, it is not reasonable for long polar codes be- reason for the enormous equipment many-sided quality required. In this brief, we examine the encoding procedure in the perspective of extensive scale reconciliation execution and propose another proficient encoder engineering that is sufficient for long polar codes and successful in mitigating the equipment multifaceted nature. As the proposed encoder permits high-throughput encoding with little equipment many-sided quality, it can be efficiently connected to the plan of any polar code and to any level of parallelism.

I.INTRODUCTION:

Polar Code is another class of blunder redressing codes that provably accomplishes the limit of the fundamental channels. What's more, solid calculations for building, encoding, and deciphering the code are altogether created [1]–[5]. Because of the channel limit accomplishing property, the polar code is currently considered as a noteworthy achievement in coding hypothesis, and the pertinence of the polar code is being explored in numerous applications, including information stockpiling devices[6],[7]. Despite the fact that the polar code accomplishes the fundamental channel limit, the property is asymptotical since a decent mistake amending execution is gotten when the code length is adequately long. To be near the channel limit, the code length ought to be no less than 220 bits, and numerous writing works [7] – [9] acquainted polar codes going from210 with 215 to accomplish great mistake remedying exhibitions practically speaking. What's more, the span of a message secured by a blunder redressing code away frameworks is regularly 4096 bytes, i.e., 32768 bits, and is required to be protracted to 8192 bytes or 16384 bytes sooner rather than later. In spite of the fact that the polar code has been viewed as being connected with low multifaceted nature, such a long polar code experiences serious equipment many-sided quality and long inactivity. Along these lines, a design that can proficiently manage long polar codes is important to make the expansive scale mix (VLSI) usage achievable.

Different theoretic parts of the polarcode, including code development and unraveling calculations, have been explored in past works [1]–[5], and effective deciphering structures have been examined. Progressive cancelation (SC) deciphering has been generally utilized in[9]–[11],and propelled unraveling calculations, for example, conviction engendering decoding[12], listed e-coding[13],and disentangled SC[7],[14] have been as of late utilized. Then again, equipment models for polar encoding have once in a while been talked about. Among a couple of compositions managing equipment implementation, [1] presented a straight-forward encoding design that procedures all the message bits in a completely parallel manner. The completely parallel engineering is natural and simple to implement, but it is not reasonable for long polar codes because of over the top equipment multifaceted nature.

What's more, the incomplete total network (PSN) for a SC decoder [7], [8], [11] is viewed as a polar encoder. Because of the way of progressive decoding, however, the number of data sources is seriously limited in the PSN, 1or2 bits at once. Since a polar encoder typically takes the contributions from a buffer or memory of which bit width is much larger, the PSN is not proper for outlining a general polar encoding engineering. For the primary time, this brief breaks down the encoding procedure in the view purpose of VLSI usage and proposes an incompletely parallel engineering. The proposed encoder is profoundly alluring in actualizing a long polar encoder a sit can achieve a high throughput with little equipment many-sided quality.

II. POLARENCODING:

The polar code uses the channel polarization wonder that every channel approaches either a splendidly solid or a totally uproarious channel as the code length goes to boundlessness over a consolidated channel developed with an arrangement of N indistinguishable sub channels [1].As there risk of every sub channel is known a priori, K most dependable sub channels are used to transmit information, and the rest of the sub channels are set to foreordained qualities to build a polar (N, K) code. Since the polar code has a place with the class of direct square codes, the encoding procedure can be portrayed by the generator framework. The generator network GN for code length N or 2n is gotten by applying the nth energy to the given the generator lattice, the code word is figured by x = uGu, where u and x speak to data and code word vectors, individually. All through this brief, we expect that data vector u is organized Na regular request, while code word vector x is masterminded in somewhat turned around request to rearrange the clarification on the encoding procedure. A clear completely parallel encoding design was displayed in [1], which has encoding multifaceted nature of O (N log N) for a polar code of length N and takes n stages when N =2. For instance, a polar code with a length of 16 is executed with 32 XOR entryways and handled with four phases, as delineated in Fig. 1. In the completely parallel encoder, the entire encoding procedure is finished in a cycle.

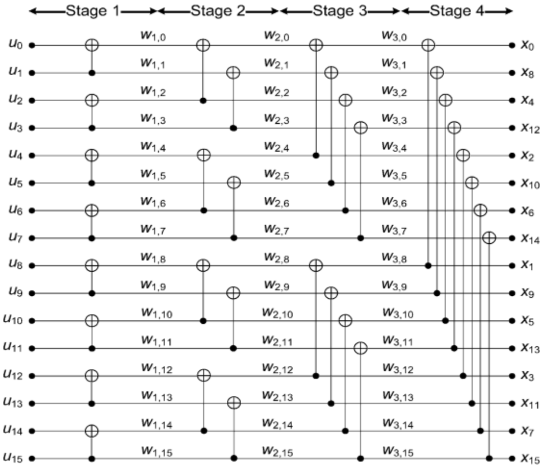


Fig. 1.Fully parallel architecture for encoding a 16-bit polar code

The completely parallel encoder is naturally composed in light of the generator grid; yet actualizing such an encoder turns into a significant load when a long polar code is utilized to accomplish a decent mistake redressing execution. In down to earth usage, the memory estimate and the quantity of XOR doors increment as the code length increments. None of the past works has profoundly considered how to encode the polar code efficiently, albeit different tradeoffs are conceivable between the inactivity and the equipment many-sided quality.

III. PROPOSED POLAR ENCODER:

In this segment, we propose a halfway parallel structure to encode long polar codes efficiently. To obviously demonstrate to the proposed approach and proper methodologies to change the design, a 4-parallel encoding engineering for the 16-bit polar code is exemplified top to bottom. The completely parallel encoding design is first changed to a collapsed shape [15], [18], and after that the lifetime examination [16] and enroll allotment [17] are connected to the collapsed engineering. Ultimately, the proposed parallel engineering for long polar codes is portrayed.

A. Collapsing Transformation:

The collapsing change [15], [18] is generally used to save hardware assets by time-multiplexing a few operations one useful unit. An information flow diagram (DFG) comparing to the fully parallel encoding process for 16-bit polar codes is shown in Fig. 2, where a hub speaks to the portion network operation, and wdenotes the jth edge at the ith arrange. Take note of that theDFG of the completely parallel polar encoder is like that of the fast Fourier change [18], [19] aside from that the polar encoder employs the part lattice rather than the butterfly operation. Given the 16-bit DFG, the 4-parallel collapsed engineering that processes 4 bits at once can be acknowledged with putting two functional units in every phase since the utilitarian unit computes2 bits at once. In the collapsing change, determining a collapsing set, which speaks to the request of operations to be executed in a practical unit, is the most imperative plan factor [15]. To build efficient collapsing sets, all operations in the fully parallel encoding are rest classified as particular folding sets. Since the information is in a characteristic request, it is sensible to alternatively disseminate the operations in the sequential order.

Thus, every stage comprises of two collapsing sets, each of which contains just odd or even operations to be performed by separate unit. Considering the four-parallel info grouping in a characteristic request, organize 1 has two collapsing sets of {A0, A2, A4, and A6} and {A1, A3, A5, A7}. Every collapsing set contains four components, and the position of a component speaks to the operational request in the corresponding practical unit. Two practical units for stage 1execute A0 and A1 at the same time toward the start and A2 and A3 at the following cycle, et cetera. The collapsing sets of stage 2 Have an indistinguishable request from those of stage 1, i.e., {B0, B2, B4, B6} and {B1, B3, B5, B7}, since the four-parallel info grouping of stage 2 is equivalent to that of stage 1. Besides, to decide the collapsing sets of another stage s, the property that the utilitarian unit forms a couple of data sources whose records contrast by 2s-1 is misused. On account of stage 3, two information whose files vary by 4 are handled together, which infers that the operational separation of the comparing information is two as the piece practical unit figures two information at once. For instance, w2, 0 and w that originate from B0 and B2 are utilized as the contributions to C0. Since both information sources ought to be legitimate to be handled in a useful unit, the operations in stage 3 are adjusted to the2, 4 late info information. Cyclic moving the collapsing sets appropriate by one, which can be acknowledged by embeddings a postponement of one time unit, is to empower full use of the utilitarian units by covering nearby emphases. Subsequently, the collapsing sets of stage 3 are resolved to {C6, C0, C2, C4} and {C7, C1, C3, C5}, where C6 in the present emphasis is covered with A0 and B0 in the following cycle. In a similar way, the property that the practical unit forms a couple of data sources whose files vary by 8 is abused in stage 4.

The collapsing sets of stage 4 are {D2, D4, D6, D0} and {D3, D5, D7, D1}, which are gotten by cyclic moving the past collapsing sets of stage 3by two. As a rule, a phase whose record s is not exactly or equivalent to log P, where P is the level of parallelism, has a similar collapsing sets controlled by equally interleaving the operations in the sequential request, and another phase whose list s is bigger than log2P has the collapsing sets got by cyclic moving the past collapsing sets of stage s - 1 appropriate by s - log2P.
Presently, let us consider the postpone components required in the collapsed engineering all the more decisively. At the point when an edge wij from useful unit S to practical unit T has a deferral of d, the postpone necessities for w ij ascertained dole out the F-collapsed engineering can be
D(wij)=Fd+ t - s (1)

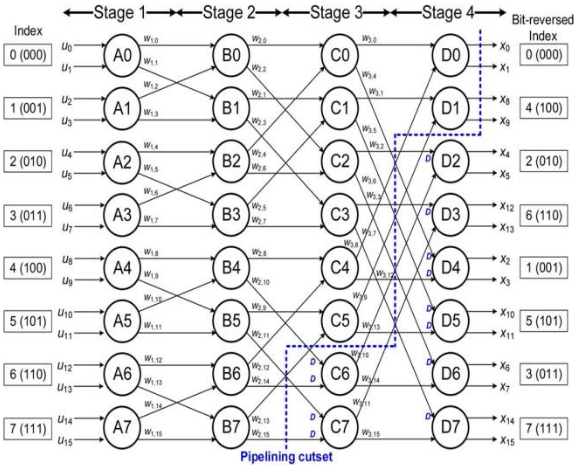
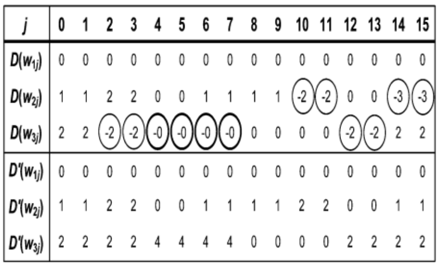


Fig. 2.DFG of 16-bit polar encoding

Where t and s indicate the position in the collapsing set comparing to T and S, individually. Take note of that (1) is a simplified postpone condition [15] determined with accepting that the part practical unit is not pipe lined. The defer necessities of the 4-collapsed design, i.e., D (w) for 1 = i = 3 and 0 = j = 15, are outlined in Fig. 3. For example, wij from B0 to C0 requests one postponement since d =0, t =1, and s =0. Take note of that a few edges demonstrated by circles have negative postponements. For the 2, 0 collapsed design to be attainable, the defer necessities must be bigger than or equivalent to zero for every one of the edges. Pipe covering or re timing methods can be connected to the completely parallel DFG so as to guarantee that its collapsed equipment has no negative delays. Every edge with a negative postponement ought to be repaid by embeddings no less than one defer component to make the estimation of (1) not negative. We need to ensure that the two contributions of an operation go through a similar number of defer components from the beginning stages. In the event that they are distinctive, extra postpone components are embedded to make the ways have a similar defer components. In Fig. 3, for instance, four edges with zero postponements are uncommonly set apart with negative zeros since extra deferrals are vital because of the confound of the quantity of postpone components. The DFG is pipe lined by embeddings defer components, as appeared in Fig. 2, where the dashed line shows the pipeline cut set connected with 12 postpone components. The postpone prerequisites of the pipe lined DFG D_) are recalculated in view of (1) and appeared at the base of Fig. 3. Thus, 8 practical units and 48 postpone components altogether are sufficient to actualize the 4-parallel 4-collapsed encoding engineering in light of the collapsing sets.



j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$D(w_{1j})$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$D(w_{2j})$	1	1	2	2	0	0	1	1	1	1	-2	-2	0	0	-3	-3
$D(w_{3j})$	2	2	-2	-2	-0	-0	-0	-0	0	0	0	0	-2	-2	2	2
$D'(w_{1j})$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$D'(w_{2j})$	1	1	2	2	0	0	1	1	1	1	2	2	0	0	1	1
$D'(w_{3j})$	2	2	2	2	4	4	4	4	0	0	0	0	2	2	2	2

Fig.3.Original delay requirements D(wij) and recalculated delay requirements D (wij)

B. Lifetime Analysis and Register Allocation:

Despite the fact that a collapsed design for 16-bit polar encoding is exhibited in the past area, there is space for minimizing the quantity of defers components. The lifetime investigation [16] is utilized to find the base number of defer components required in executing the collapsed engineering. The life time of each factor is graphically spoken to in the straight life time diagram delineated in Fig. 4. Since every one of the edges beginning from stage 1 request no postpone components, just w2j and product exhibited in Fig. 4. For example, w3, 03jis alive for two cycles as it is created at cycle 1 and devoured at cycle 3. The quantity of factors alive in every cycle is displayed at the correct side of the chart. Note that the quantity of live factors at the fourth or later clock cycles considers the following emphasis covered with the present emphasis. Thusly, the most extreme number of live factors is 12, which implies that the collapsed engineering can be actualized with 12 defer components rather than 48.

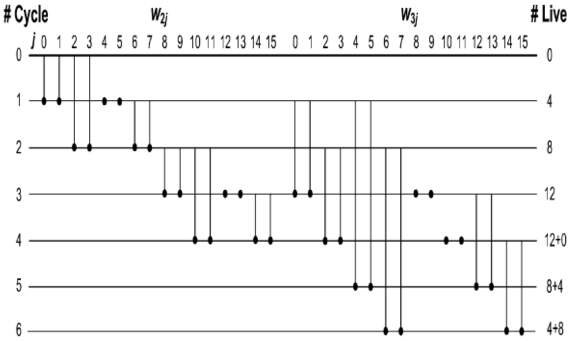
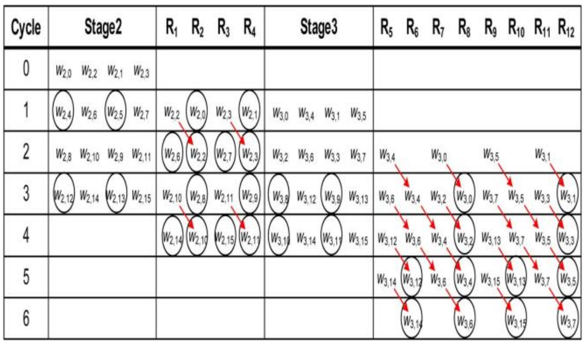


Fig.4.Linearlifetimechartforw2ȷandw3ȷ

Once the base number of defer components has been resolved, every variable is assigned to an enlist. For the above illustration, the enroll distribution is arranged in Fig. 5. In the enroll portion table [17], all the 12 registers are appeared at the first push, and each line depicts how the registers are designated at the comparing cycle. With considering the 4-parallel handling, factors are precisely designated to registers in a forward way.



Cycle	Stage2	R_1	R_2	R_3	R_4	Stage3	R_5	R_6	R_7	R_8	R_9	R_{10}	R_{11}	R_{12}
0	$w_{2,0}$ $w_{2,2}$ $w_{2,1}$ $w_{2,3}$													
1	$w_{2,4}$ $w_{2,6}$ $w_{2,5}$ $w_{2,7}$	$w_{2,2}$	$w_{2,0}$	$w_{2,3}$	$w_{2,1}$	$w_{3,0}$ $w_{3,4}$ $w_{3,1}$ $w_{3,5}$								
2	$w_{2,8}$ $w_{2,10}$ $w_{2,9}$ $w_{2,11}$	$w_{2,6}$	$w_{2,2}$	$w_{2,7}$	$w_{2,3}$	$w_{3,2}$ $w_{3,6}$ $w_{3,3}$ $w_{3,7}$	$w_{3,4}$		$w_{3,0}$		$w_{3,5}$		$w_{3,1}$	
3	$w_{2,12}$ $w_{2,14}$ $w_{2,13}$ $w_{2,15}$	$w_{2,10}$	$w_{2,8}$	$w_{2,11}$	$w_{2,9}$	$w_{3,8}$ $w_{3,12}$ $w_{3,9}$ $w_{3,13}$	$w_{3,6}$	$w_{3,4}$	$w_{3,2}$	$w_{3,0}$	$w_{3,7}$	$w_{3,5}$	$w_{3,3}$	$w_{3,1}$
4		$w_{2,14}$	$w_{2,10}$	$w_{2,15}$	$w_{2,11}$	$w_{3,10}$ $w_{3,14}$ $w_{3,11}$ $w_{3,15}$	$w_{3,12}$	$w_{3,6}$	$w_{3,4}$	$w_{3,2}$	$w_{3,13}$	$w_{3,7}$	$w_{3,5}$	$w_{3,3}$
5							$w_{3,14}$	$w_{3,12}$	$w_{3,6}$	$w_{3,4}$	$w_{3,15}$	$w_{3,13}$	$w_{3,7}$	$w_{3,5}$
6								$w_{3,14}$		$w_{3,6}$		$w_{3,15}$		$w_{3,7}$

Fig.5.Registerallocationtableforw2ȷandw3ȷ

In Fig. 5, a bolt directs that a variable put away in an enroll is relocated to another enlist, and a circle shows that the variable is devoured at the cycle. For case, w2, 0 and w2, 5 are expended in an utilitarian unit to execute operation C0 that produces w2, 4 and w3, 0.At a similar time, w2, 1and w3, 4 are devoured in another useful unit to execute operation C1 that produces w3, 1 and w3, 5.The movement of alternate factors can be followed by taking after the enroll designation table. At last, the subsequent 4-parallel pipe lined structure proposed to encode the 16-bit polar code is delineated in Fig. 6, which comprises of 8 useful units and 12 postpone components. A couple of two utilitarian units takes accountable for one phase, and the defer components are to store factors as per the enlist designation table. The equipment structures for stages 1 and 2 can be clearly acknowledged as no postpone components are essential in those stages, while for stages 3 and 4, a few multiplexers are put before some utilitarian units to configure the contributions of the practical units. The proposed engineering constantly forms four specimens for each cycle as per the collapsing sets and the enlist designation table. Take note of that the proposed encoder takes a couple of contributions to a characteristic request and creates a couple of yields in somewhat turned around request, as appeared in Fig. 2. As the practical unit in the proposed engineering forms a couple of 2 bits at once, the proposed design keeps up the back to back request at the information side and the bit turned around request at the yield side if a couple of continuous bits is viewed as a solitary element.

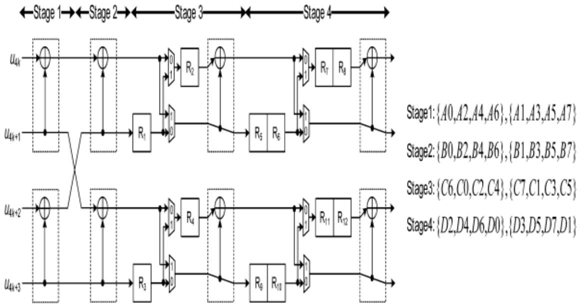
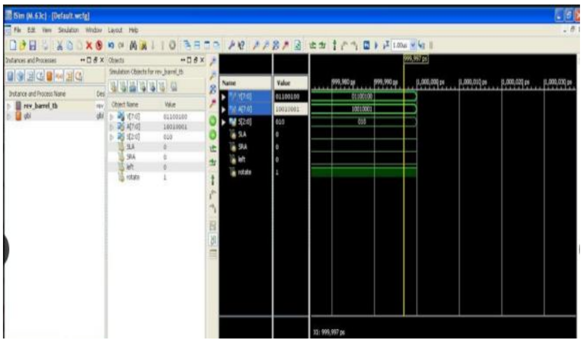
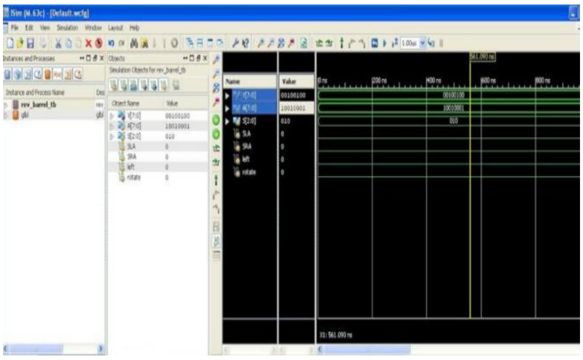

Stage 1
Stage 2
Stage 3
Stage 4
u_{4k}
u_{4k+1}
u_{4k+2}
u_{4k+3}
R_1
R_2
R_3
R_4
R_5
R_6
R_7
R_8
R_9
R_{10}
R_{11}
R_{12}
0
1
Stage1: {A0,A2,A4,A6},{A1,A3,A5,A7}
Stage2: {B0,B2,B4,B6},{B1,B3,B5,B7}
Stage3: {C6,C0,C2,C4},{C7,C1,C3,C5}
Stage4: {D2,D4,D6,D0},{D3,D5,D7,D1}

Fig. 6. Proposed 4-parallel folded architecture for encoding the polar (16, K) codes.

IV.SIMULATION RESULTS:




ISim (M.63c) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Instances and Processes
Objects
Simulation Objects for rev_barrel_tb
Instance and Process Name
rev_barrel_tb
glbl
0 ns
200 ns
400 ns
600 ns
800 ns
561.093 ns
X1: 561.093 ns

Object Name	Value
Y[7:0]	00100100
A[7:0]	10010001
S[2:0]	010
SLA	0
SRA	0
left	0
rotate	0

V. CONCLUSION:

This brief has introduced another somewhat parallel encoder design created for long polar codes. Numerous advancement systems have been connected to infer the proposed engineering. Trial comes about demonstrate that the proposed design can spare the equipment by up to 73% contrasted and that of the completely parallel engineering. At long last, the relationship between the equipment many-sided quality and the through puts is examined to choose the most reasonable engineering for a given application. Along these lines, the proposed design gives a useful answer for encoding a long polar code.

REFERENCES:

[1] E. Arikan, “Channel polarization: A method for constructing capacity achieving codes for symmetric binary-input memory less channels,” IEEE Trans. Inf. Theory, vol. 55, no. 7, pp. 3051–3073, Jul. 2009.

[2] R. Mori and T. Tanaka, “Performance of polar codes with the construction using density evolution,” IEEE Commun. Lett., vol. 13, no. 7, pp. 519–521, Jul. 2009.

[3] S. B. Korada, E. Sasoglu, and R. Urbanke, “Polar codes: Characterization of exponent, bounds, constructions,” IEEE Trans. Inf. Theory, vol. 56,no. 12, pp. 6253–6264, Dec. 2010.

[4] I. Tal and A. Vardy, “List decoding of polar codes,” in Proc. IEEE ISIT,2011, pp. 1–5.

[5] K. Chen, K. Niu, and J. Lin, “Improved successive cancellation decoding of polar codes,” IEEE Trans. Commun., vol. 61, no. 8, pp. 3100–3107,Aug. 2013.

[6] G. Sarkis and W. J. Gross, “Polar codes for data storage applications,” in Proc. ICNC, 2013, pp. 840–844.

[7] G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross, “Fast polar decoders: Algorithm and implementation,” IEEE J. Sel. Areas Commun., vol. 32, no. 5, pp. 946–957, May 2014.

[8] G. Berhault, C. Leroux, C. Jego, and D. Dallet, “Partial sums generation architecture for successive cancellation decoding of polar codes,” in Proc.IEEE Workshop SiPS, Oct. 2013, pp. 407–412.

[9] B. Yuan and K. K. Parhi, “Low-latency successive-cancellation polar decoder architectures using 2-bit decoding,” IEEE Trans. Circuits Syst.
I, Reg. Papers, vol. 61, no. 4, pp. 1241–1254, Apr. 2014.

[10] C. Leroux, A. J. Raymond, G. Sarkis, and W. J. Gross, “A semi-parallel successive-cancellation decoder for polar codes,” IEEE Trans. Signal Process., vol. 61, no. 2, pp. 289–299, Jan. 2013.