Parameterized Macrocells with Accurate Delay Models for Core-Based Designs

Makram M. Mansour, Mohammad M. Mansour, Amit Mehrotra
Coordinated Science Laboratory/ECE Department
University of Illinois at Urbana-Champaign
1308 W. Main St., Urbana, IL 61801
{mansour, mmansour, amehrotr}@uiuc.edu

Abstract

In this paper we propose a new design methodology targeted for core-based designs using parameterized macrocells (PMC's). This methodology provides the flexibility for instance-based cores to be easily customized for application requirements. By using few scaling parameters to characterize a PMC, a macrocell can be instantiated in virtually any size depending on the required performance. Moreover, a new first-order macro delay model is proposed which is a function of the scaling parameters of the PMC which enables accurate delay predictions at the subsystem/core level. The proposed delay model is suitable for use by a delay optimizer to determine the optimum scaling parameters of individual PMC's in a core. A PMC library has been developed and used to design cores for communications applications. To demonstrate the effectiveness of the proposed methodology, several subsystems used in a channel LDPC decoder were synthesized using this library where the individual PMC's were optimized for minimum delay. The resulting custom-quality layout have areas ranging from $40 \times 100\,\mu\text{m}^2$ *to* $380 \times 200\,\mu\text{m}^2$ *and delay in the range of* $1.6\,\text{ns}$ *to* $10\,\text{ns}$ *in* $0.18\,\mu\text{m}$, $1.8\,\text{V}$ *CMOS technology.*

1 Introduction

Predictable core-based design approach based on design reuse has been established as an effective way for harnessing the full potential of system-on-a-chip integration offered by the advanced IC process technologies [1,2]. In these technologies, physical design problems have a significant impact on the predictability, reliability, and efficiency of the designs. Furthermore, the increased complexity of these technologies requires designers to deal with numerous structural and electrical parameters that interact in a nonlinear manner. For example, timing and floor-planning must be addressed early in the design process, at the same time as the functional requirements.

By using pre-designed and pre-verified intellectual property (IP) cores, design trade-offs can be made at higher levels of abstraction. This, in turn, leads to shorter design cycles and more time to explore the abstracted architectural variations. However, the problem with these IP cores is that they are application specific and difficult to customize. This restricts the architectural design space to investigate. For example, at the macrocell level, some desirable macrocells may not exist because both the number and the variations in the driving capabilities of the cells are limited. Consequently, circuit performance has to be sacrificed. Furthermore, even if these cores are customizable, their performance, in general, can become unpredictable and re-characterization is inevitable.

In this paper, we propose an effective reuse methodology which is based on layout parameterization and delay characterization. Layout parameterization allows instance-based cores to be easily optimized to a specific application by tailoring the individual leaf cells inside the core. Delay characterization allows accurate delay prediction of the instantiated cores. This in turn results in cores with custom-quality compact layouts which are more energy-efficient than those produced by pure silicon compilation.

To demonstrate the effectiveness of the proposed reuse methodology, parameterized macrocells (PMC's) were developed together with parameterized delay models for accurate delay characterization of these PMC's. These PMC's include parameterized cells such as: D-latch, full adder, 2's complementor, 8-to-1 (4-to-1) (de-)multiplexer, shuffle-exchange network switch, super buffer for delay equalization, etc., which can be used to generate layouts for real designs and to steer the development of new architectures for communication applications. This PMC library has been used to design the main subsystems of a channel LDPC decoder widely used in communication receivers. These subsystems include [6, 7]: kernel function (Q-blocks) for message computations, message processing soft-input soft-output units, and dynamic networks for message transport.

The rest of the paper is organized as follows: In Section 2, we give a brief overview on the layout parameterization and discuss the details of the parameterized macrocells developed. The PMC's are implemented within the framework of Cadence [3] CAD design suite for automatic layout generation, using $0.18\,\mu\text{m}$, $1.8\,\text{V}$ TSMC CMOS process technology. Section 3 discusses the delay characterization of the PMC's. Delay equations are derived which are function of the individual cell size and its fanout sizes. This allows accurate delay prediction by only considering cells and fanouts and without the need for low-level transistor timing analysis. Section 4 provides simulation results and implementation examples featuring the PMC's. Finally, Section 5 summarizes this work and

0-7695-1881-8/03 $17.00 © 2003 IEEE

looks at future directions.

2 Parameterized Macrocells

In this section, we present a parameterized macrocell design approach targeted for core-based designs. The main motivation behind this approach is to expose low-level transistor sizing capability to the architectural level through a *small* set of scaling parameters to enable power and delay optimization at the core-abstraction level. Scaling parameters of all constituent PMC's in the core are fed to a core-optimizer that determines the optimal individual scaling parameters of the PMC's. To keep the problem tractable, it is essential first to keep the number of scaling parameters per PMC as small as possible, by optimizing the transistors relatively within each macrocell starting from minimum sized transistors, and then assigning a scaling parameter for a group of transistors depending on the functionality and number of outputs of the PMC. This results in a high-performance, handcrafted density cores.

2.1 Layout structure of macrocells

A core is composed of a number of subsystems such as datapath operators (adder, multiplier), memory elements (SRAM, flip-flop), and control structures (state machines) [4]. A subsystem is defined in terms of macrocells; for example, an *n*-bit full adder is a subsystem made of *n* one-bit full adder macrocells and an *n*-bit buffer is made up of *n* one-bit D-flip-flop macrocells. Hence, a macrocell is a regularly recurring element in a subsystem.

To enable regularity in the design, the one-dimensional layout strategy [5] was adopted. This strategy consists of a single row of P-type and N-type transistors running parallel to single power and ground rails. Fixing the rail-to-rail height enables back-to-back stacking of macrocells which share common power and ground rails. Routing occurs within the power rails with signals connected via a combination of poly and metal-1 lines. Only metal-1 layer was used within a macrocell so that other metal layers could be used to connect macrocells together and for global routing. For macrocells with small number of transistors, the P- and N-type transistors are placed across a horizontal center line running parallel to the power rails. In this case, transistors are scaled vertically away from the center line while staying within the fixed cell height defined by the power rails as shown in Figure 1(a). The maximum transistor sizing in this case is 4 to 5 times as large as the minimum transistor size. For larger scaling, transistor folding techniques are used [5]. On the other hand, macrocells with a large number of transistors are rotated 90 degrees and the P- and N-type transistors are laid across one or more parallel vertical center lines as shown in Figure 1(b). Scaling in this case is done horizontally.

2.2 Macrocell parameterization

A parameterized macrocell is a macrocell which enables scaling of individual or groups of transistors, wires, and spacing to tailor the macrocell according to user specified scaling parameters. A PMC can therefore be instantiated with practically any size to within the technology's minimum feature size. This constitutes a considerable advantage over macrocells which are generally available in few discrete sizes (generally two) in a standard macrocell library. A PMC can have one or more scaling parameters depending on its functionality and the number of outputs.

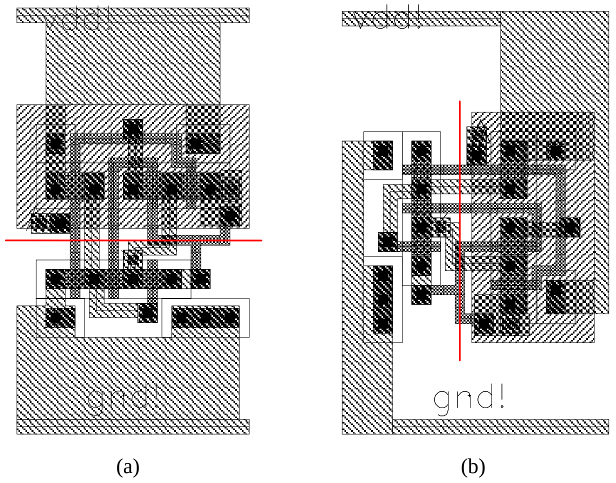


Figure 1. 1-D layout strategy: (a) Horizontal, and (b) vertical centerline approaches.

For example, as shown in Figure 2 a four-bit 2-to-1 multiplexer needs a single scaling parameter, while a transmission-gate-based one-bit full adder (FADDXG) needs two scaling parameters as shown in Figure 3. Moreover, an FADDXG implementing saturation arithmetic (FADDXG-SAT) needs three scaling parameters, two for the FADDXG and one for the output correction logic as shown in Figure 4. Through simulations, one can identify groups of transistors to be scaled together, in addition to relative scaling within each of these groups (see Figure 2 for example). Thus, by only manipulating a small number of scaling parameters one can effectively achieve the same performance as that achieved by macrocells designed by sizing the transistors individually.

3 Delay Characterization

In order for the core-based design methodology employing parameterized macrocells to be successful, accurate delay models are needed. Gate level modeling techniques [8–10] are impractical to apply to characterize macrocells. Moreover, macromodeling techniques for characterizing standard macrocells is also not applicable since it is infeasible to characterize each possible instance of a PMC. These macromodeling techniques employ look-up tables and hence are not continuous, posing problems during the optimization phase. What is required is a simple macromodeling technique that takes into account the scaling parameters of the macrocells and have continuous first derivatives with respect to these parameters.

We propose to use a first-order macromodeling delay technique targeted for PMC's which is based on the techniques of [4] for standard macrocells. The delay model is given by the following equation

$$\tau(\underline{\sigma}, \underline{\sigma}_L) = \tau_0(\underline{\sigma}) + m(\underline{\sigma})C(\underline{\sigma}_L), \tag{1}$$

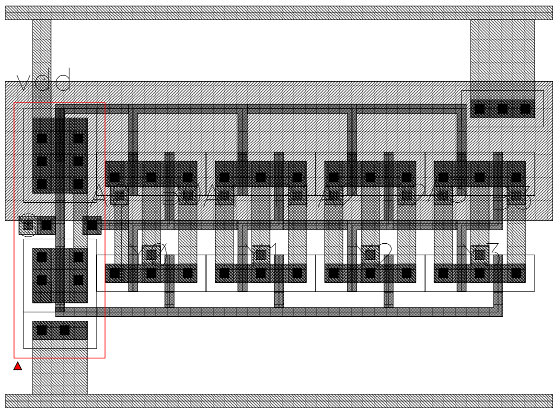


Figure 2. Layout of a four-bit 2-to-1 multiplexer with a single scaling parameter.

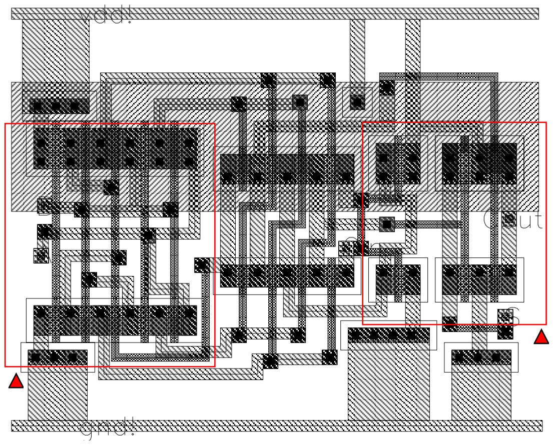


Figure 3. Layout of a one-bit full adder with two scaling parameters.

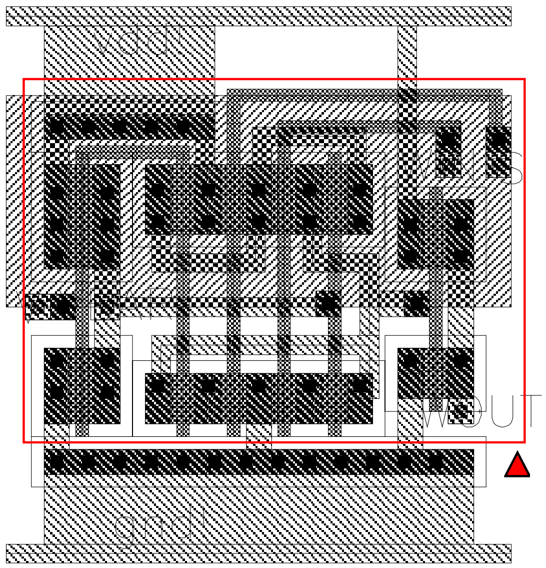


Figure 4. Layout of an output correction logic for saturation arithmetic used in full adders and subtractors, requiring one scaling parameter.

where $\underline{\sigma}$ is the scaling vector of the PMC, $\underline{\sigma}_L$ is the (combined) scaling vector of the loading PMC(s), τ_0 is the intrinsic delay of the PMC, m is an empirically determined parameter from the slope of the delay vs. load capacitance plots of the PMC, and C is the (combined) input capacitance of the loading PMC(s).

3.1 2's complementor example

This example illustrates how to derive the delay characteristic equations of a 2's complementor PMC. 2's compelmentors are commonly employed in subsystems where it is necessary to determine the absolute value of 2's complement numbers. Figure 5 shows a circuit schematic of a 2's complementor. There are three inputs for the 2's complementor: the data input X, an input propagate signal R_{in}, and an enable E. The outputs are the 2's complement Y and an output propagate signal R_{out}. As shown, the two outputs have two different loading requirements: R_{out} will drive R_{in} of the following 2's complement stage, while Y is the primary output of the macrocell and can drive any arbitrary macrocell (or load). It is therefore natural to define the scaling vector $\underline{\sigma} = (\sigma_R, \sigma_Y)$ for this macrocell corresponding to the two outputs R and Y.

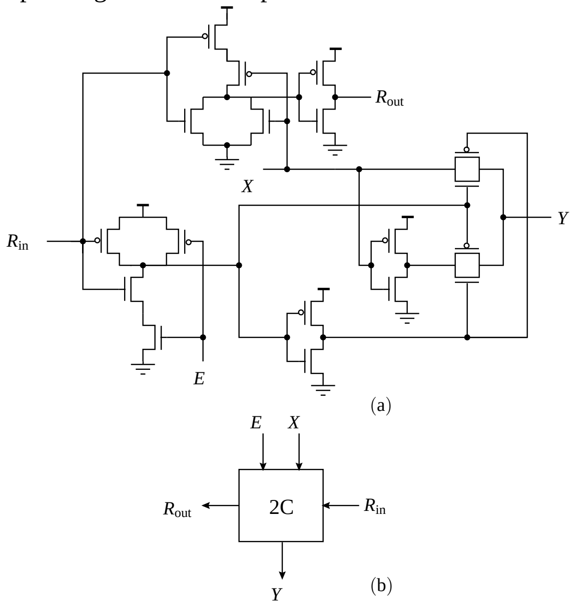


Figure 5. (a) Circuit schematic of a 2's complementor, and (b) block symbol.

Next, according to (1) there are three quantities that need to be determined: $\tau_0(\underline{\sigma})$, $m(\underline{\sigma})$, and $C(\underline{\sigma}_L)$. The 2's complement PMC is instantiated for scaling vectors $\underline{\sigma} = (i, j),\ i, j = 1, \cdots, 4$. First, the intrinsic delay is determined by simulating each of the instances with open loads, resulting in

$$\tau_0(\underline{\sigma}) = (0.65 - 0.22\sigma_R + 0.03\sigma_R^2, 0.79 - 0.28\sigma_Y + 0.04\sigma_Y^2)\ \text{ns}.$$

Next, the slope parameter $m(\underline{\sigma})$ is determined by repeating the first step for different load capacitances at the outputs R_{out} and Y, resulting in

$$m(\underline{\sigma}) = (0.05 - 0.02\sigma_R, 0.07 - 0.03\sigma_Y)\ \text{ns/fF}.$$

Finally, the X and R_{in} input gate capacitances of the PMC are obtained by loading the PMC with itself. This results in

$$C(\underline{\sigma}_L) = (0.9+5\sigma_{LY}, 1.7+7.9\sigma_{LR}, 1.4+6.2\sigma_{LE})\text{ fF}.$$

To demonstrate the accuracy of the proposed model, Figure 6 shows the delay of a 2's complement PMC driving similar cells at both outputs versus actual delay determined using HSPICE simulations. As shown in the figure, the proposed model accurately approximates the delay of the PMC for different scaling parameters of the cell to within an error of 6% when loaded with another PMC with scaling vectors $\underline{\sigma}_{L1} = (1,1)$ (bottom) and $\underline{\sigma}_{L2} = (6,6)$ (top).

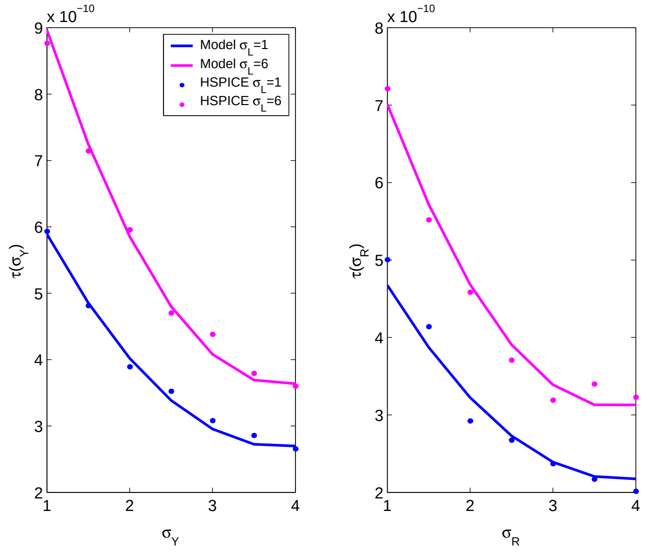


Figure 6. Comparison between HSPICE and model delay for a 2's complementor PMC.

3.2 PMC library

Table 1 lists the PMC's developed together with the number of parameters assigned to each cell and their corresponding areas. All cells were characterized for delay in a similar fashion to the example given in subsection 3.1.

4 Subsystem Implementation Examples

In this section, we implement several subsystems for a communication channel decoder employing low-density parity check (LDPC) codes for forward error correction. The subsystems include the compare-select plus correction kernel operation for the well known BCJR algorithm in differential form [6, 7], the soft-input soft-output message process unit (SISO-MPU), and the Omega network for message transport between memory and the SISO-MPU's.

4.1 Kernel function *Q(x,y)*

The bivariate symmetric function $Q(x,y)$ implements the compare-select plus correction operation given by

$$Q(x,y) = \max(x,y) - \max(x+y,0) + \max\left(\frac{5}{8} - \frac{|x-y|}{4}, 0\right) - \max\left(\frac{5}{8} - \frac{|x+y|}{4}, 0\right). \quad (2)$$

Table 1. PMC library listing.

Cell Name	Area (μm^2)	No. of Parameters
AND2	47.2	1
BUFX1	37.8	2
BUFX2	94.4	2
DEMUX-2X1	63.6	2
DEMUX-3X1	169.4	3
DFF	134.4	2
DFFE	179.4	2
DFFR	151.7	2
DFFRE	198.2	2
DFFS	153.6	2
DFFSE	198.2	2
DLATCH	102.2	1
FADDXG	158.7	2
FSUBXG	204.4	2
INV	40.0	1
MUX-2X1	58.1	2
MUX-4X1	124.8	3
NAND2	37.8	1
NOR2	31.9	1
OR2	48.6	1
SATURATE-LSB	98.8	1
SATURATE-MSB	147.5	1
SHUFFTLE-EXCHANGE	76.8	2
TWOSCOMP	132.3	2
XGATE	66.6	1
XNOR	53.7	1
XOR	58.1	1
DELTA	367.4	2
DELTA1-DELTA2	235.2	1

The logic circuit implementing (2) using 2's complement saturation arithmetic is shown in Figure 7. Optimizing the Q-block for delay is critical for the operation of the LDPC decoder since it constitutes a computation bottleneck. (For further details, the reader is referred to [6, 7].) Figure 8a shows a PMC optimized for high speed operation (delay of 1.6 ns), while the PMC of Figure 8b is optimized for a delay of 10 ns.

Table 2. Delay and area results for the optimized 4-bit PMC's used in the Q-block.

PMC	Delay (ns)	Area (μm^2)
FADDXG	0.34	918.3
FADDXG-SAT	0.42	1058.9
FSUBXG-SAT	0.39	1291.0
MUX-2X1	0.19	114.0
TWOSCOMP	0.26	299.4
DELTA	0.41	293.1
INV	0.25	30.9
XOR	0.35	45.6
DFF	0.29	608.3

The 4-bit Q-block subsystem of Figure 8 was implemented using the proposed PMC design methodology, and the proposed macro delay model was used for delay computations. The Q-block includes the following PMC's: full adder (FA), FA with saturation arithmetic, full subtractor with saturation arithmetic, 2-to-1 multiplexer, 2's complementor, a PMC im-

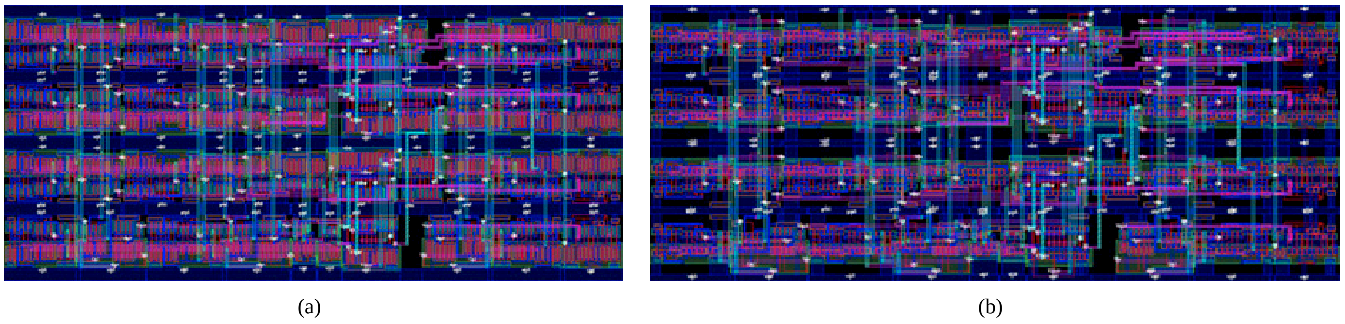


Figure 8. PMC for the "max-quartet" function: (a) high speed (1.6 ns), and (b) delay-optimized for 10 ns.

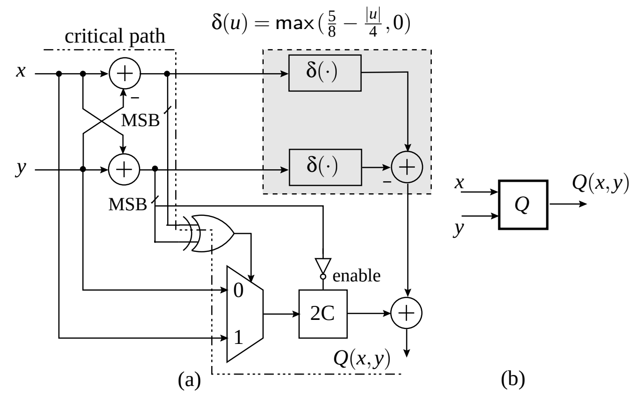


Figure 7. (a) Logic circuit implementing the compare-select plus correction function $Q(x,y)$, and (b) logic symbol of the Q-block.

plementing the function

$$\delta(u) = \max\left(\frac{5}{8} - \frac{|u|}{4}, 0\right),$$

as well as an inverter and an XOR gate. The scaling vectors of each of the PMC's used in the Q-block were determined using a delay optimizer which minimizes the delay of the critical path shown in Figure 7. Table 2 shows the delay and area characteristics of the optimized PMC's. The overall area of the Q-block is $4,864.1\,\mu\text{m}^2$ with a critical path delay of 1.6 ns.

4.2 SISO-MPU

Messages in an LDPC decoder are updated to the key equation of the BCJR algorithm which can be written in terms of the function $Q(x,y)$ using the following recursions:

$$\Delta\alpha' = Q(\Delta\alpha, \gamma - \lambda), \qquad \Delta\beta' = Q(\Delta\beta, \gamma - \lambda),$$
$$\Lambda = Q(\Delta\alpha, \Delta\beta), \qquad \Gamma = \Lambda + (\gamma - \lambda), \tag{3}$$

where λ, γ are soft-input messages and Λ, Γ are soft-output messages. Figures 9-10 show a parallel and a serial MPU implementation of (3) and Figures 11a and 11b show their respective PMC's optimized for a stage delay of 10 ns.

4.3 Omega network

The input messages λ and γ must be permuted before being processed by the SISO-MPU's, and the output messages Λ and

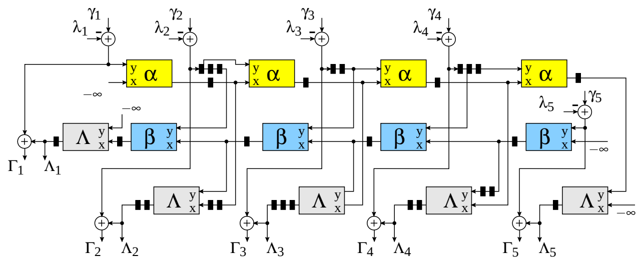


Figure 9. Parallel MPU implementation.

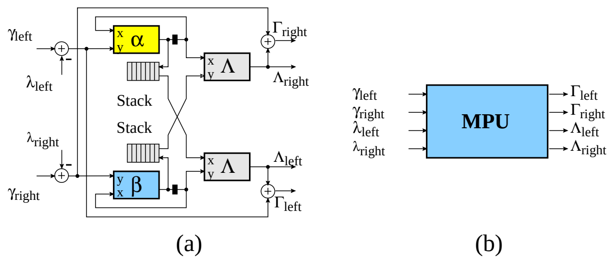


Figure 10. (a) Serial MPU, and (b) block symbol.

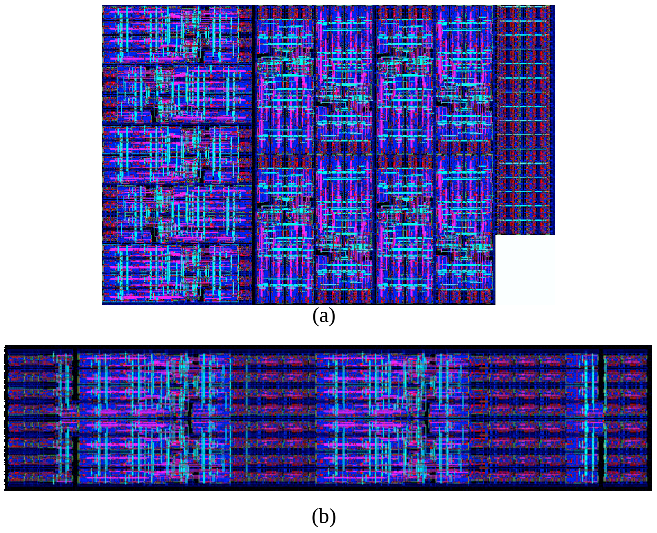


Figure 11. PMC for the (a) parallel MPU, and (b) serial MPU optimized for a stage delay of 10 ns.

Γ must be inverse-permuted after exiting the SISO-MPU's. A shuffle-exchange multi-stage interconnection network (Omega network) is used for message permutation. The architecture of an Omega network that permutes $S = 8$ messages is shown in Figure 12, where the blocks shown in the figure are 2×2 switches. Figure 13a shows the PMC of a 1-bit Omega network for routing $S = 64$ bits, and Figure 13b shows the PMC of a 4-bit Omega network for routing $S = 64$ 4-bit messages.

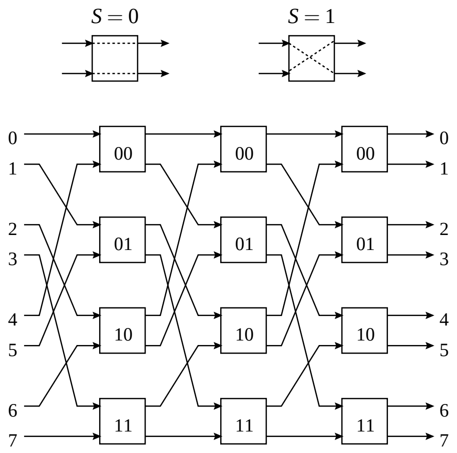


Figure 12. Omega Network.

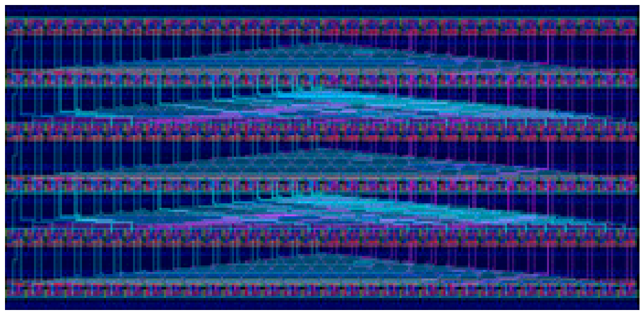
Table 3 lists the decoder subsystems generated together with their corresponding areas.

Table 3. Generated decoder subsystems and their areas.

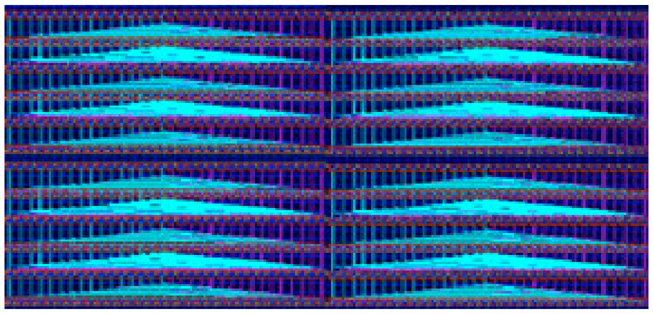
Cell Name	Area (μm^2)
QBlock-slow	4,474.9
QBlock-fast	4,864.1
Ω-Network 1-bit	20,187.6
Ω-Network 4-bit	79,016.1
SISO-MPU parallel	76,141.1
SISO-MPU serial	37,201.0

5 Conclusion

A new design methodology based on parameterized macrocells and macro delay models targeted for core-based designs have been presented. Development of accurate macro power models for the PMC's and making them technology portable to further increase reusability and flexibility will be taken in a future work. In addition, a new current macromodel for computing the maximum current drawn by a PMC in terms of its characteristic parameters will be developed to size-up the power rails to address power grid noise and reliability issues.



(a)



(b)

Figure 13. PMC for the Ω-network of Figure 12.

References

[1] H. Chang, L. Cooke, M. Hunt, G. Martin, A. McNelly, and L. Todd, *Surviving the SOC Revolution: A Guide to Platform-Based Design.* Boston: Kluwer Academic Publishers, 1999.

[2] M. Keating and P. Bricaud, *Reuse Methodology Manual: For System-on-a-Chip Designs.* Boston: Kluwer Academic Publishers, 1999.

[3] Cadence Design Systems, Inc., *Design Framework II, Virtuoso Parameterized Cell Reference, Release IC 4.4.5*, 1999.

[4] N. Weste and K. Eshraghian, *Principles of CMOS VLSI design: a systems perspective*, New York: Addison-Wesley Longman Publisher, 1994.

[5] R. Baker, H. Li, and D. Boyce, *CMOS: Circuit Design, Layout, and Simulation.* New Yorl: Wiley-IEEE Press, 1997.

[6] Mohammad M. Mansour and N. Shanbhag, "High-throughput memory efficient decoder architectures for LDPC codes," *submitted to IEEE Transactions on VLSI Systems*, 2002.

[7] Mohammad M. Mansour and N. Shanbhag, "Low-power VLSI decoder architectures for LDPC codes," in *ISLPED 2002, Monterey, CA, Aug. 2002, pp. 284-289*.

[8] Makram M. Mansour, Mohammad M. Mansour, and A. Mehrotra, "Modified Sakurai-Newton current model and its applications to CMOS digital circuit design," in *IEEE Computer Society Annual Syposium on VLSI (to appear)*, Feb. 2003.

[9] Makram M. Mansour and N. Shanbhag, "Simplified current and delay models for deep submicron CMOS digital circuits," in *IEEE Interenational Conference on Circuits and Systems*, vol. 5, pp.109-112, May 2002.

[10] Makram M. Mansour, *Layout generation for deep submicron CMOS circuits*, MS thesis, University of Illinois at Urbana-Champaign, 2002.