

## Design of Compact Modified Radix-4 8-Bit Booth Multiplier

Trio Adiono<sup>1</sup>, Hans Herdian<sup>2</sup>, Suksmandhira Harimurti<sup>1</sup>, and Tengku Ahmad Madya Putra<sup>2</sup>

<sup>1</sup>University Center of Excellence on Microelectronics Institut Teknologi Bandung, INDONESIA

<sup>2</sup>School of Electrical Engineering and Informatics Institut Teknologi Bandung, INDONESIA  
tadiono@stei.itb.ac.id

**Abstract:** The conventional serial booth multiplier generates full partial product  $(2n+1)$  and it require a  $2n$  bit adder to perform the operation. Because of that the area of the circuit will be inefficient. Modified booth algorithm is used to reduce the number of partial product calculation to only half of the serial-parallel multiplier implementation, while keeping the area is still in acceptable level. In order to optimize the area, we propose a design by modifying the architecture of radix-4 modified booth multiplier. Here, we customize the architecture by sharing the registers and minimizing the bit size of adder. For the adder block, we further optimize the design by combining the ripple carry and carry look-ahead adder in order to eliminate the need to create separate circuit for receiving many signal inputs. The carry look-ahead segment is implemented using static Manchester Carry Chain because of its high speed, small size, and consistent performance. The design is implemented using  $0.13\mu\text{m}$  CMOS technology. From the implementation and post-layout simulation result, the total number of transistor that is used is 1132 and the total combination delay is 340 ns.

**Keywords:** booth multiplier; radix-4; Manchester carry-chain adder

### 1. Introduction

Multiplier is an essential component used in a computer system, especially for digital signal processing. The requirement for a multiplier varies depends on what type of device or application it is used for. For low-power application, it requires a small and low-power multiplier [1-4], while for real-time signal processing unit, a fast multiplier is required [5-7].

There are several algorithms that can be used to implement a multiplier [8-9]. The simplest one is the straightforward long multiplication algorithm. This algorithm requires the same calculation of partial product with the number of operand digit. Hence, it requires bigger adder for parallel implementation and more iteration for serial implementation.

Another algorithm that can be used is Modified Booth Algorithm [10]. This algorithm can reduce the number of partial product that must be calculated. By implementing this algorithm in higher radix, Modified Booth Algorithm can reduce the number of calculation in exchange to the design complexity. For digital circuit implementation, radix-4 is considered the best trade-off between speed and complexity because it is able to halve the number of partial product needed to be calculated while only adding a bit-shift function as an additional operation [11-13].

Most of the designs of this multiplier architecture are implemented in FPGA [14-15]. While FPGA offers lesser development time and effort in multiplier IC design process. However, it limits the degree of optimization that can be applied to the design. Several groups has proposed several approaches to optimize a multiplier architecture using a full-custom design. A custom adder using complementary pass transistor and transmission gate logic has been proposed in order to minimize the delay as well as to lower the power consumption [16]. A parallel multiplier with optimized architecture, scaling the voltage supply to 0.5V, and reduced gates size has also been proposed to achieve an ultra-low power design [17]. Moreover, to achieve a significantly low power consumption and minimum delay, a replacement of several logical circuits with an optimized full-adder in (4;2) compressors was proposed [18]. In this paper, we propose a compact design of radix-4 booth multiplier by merging the result and multiplier register into one

single register and using an optimized number of bit of adder. In addition, the adder cell is designed using a modified Manchester carry chain topology to reduce circuit delay. As a result, the total propagation delay, power consumption, and total circuit size can be significantly reduced by implementing our proposed booth multiplier.

## 2. Conventional Radix-4 Booth Multiplier

### A. Radix-4 Booth Algorithm

Step-by-step of radix-4 booth algorithm to multiply two n-bits operands is as follows:

1. Initialize the A by multiplicand, B by multiplier and multiplication result (P) by 0. A and B is n-bits while P is 2n-bits.
2. For first calculation, append 'zero' to the right side of LSB of number B. Then, examine 3 adjacent LSB of number B.
3. Check the value of the examined 3 bits and perform operation to the MSB of number P based on Table 1. The operand must be sign extended by 1 bit.
4. Shift B and P 2 times to the left. Repeat this algorithm from step 1  $n/2$  times, then P is the multiplication results.

Graphically, the step-by-step of radix-4 booth algorithm to multiply two-bit operand is illustrated in Figure 1. In addition, Table 1 is also provided to give a more detail explanation on how the logic operations are done to P, based on 3 LSB of number B.

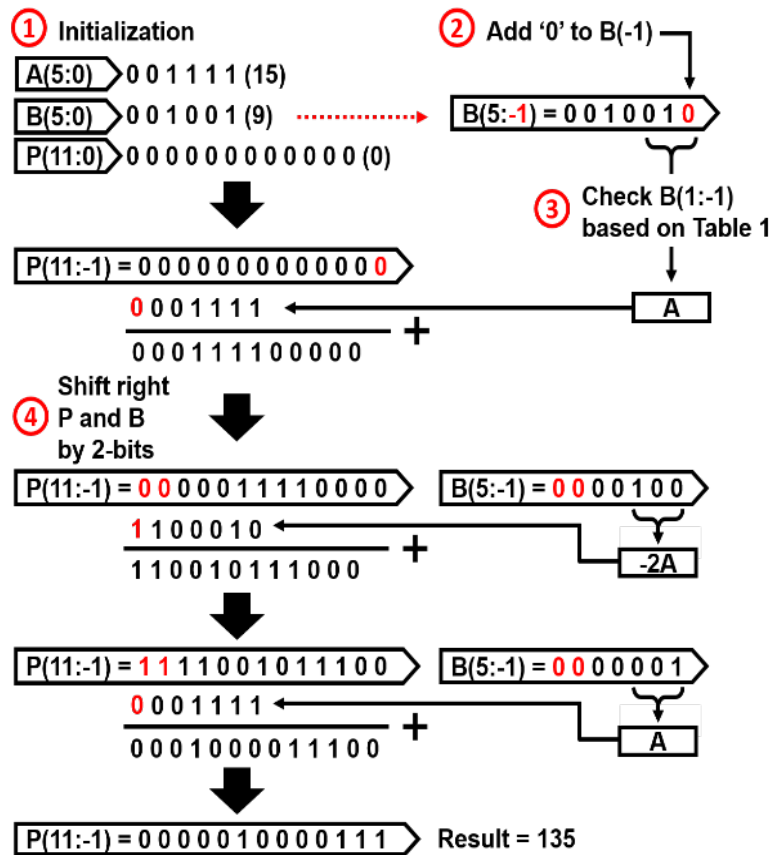


Figure 1. Visualization of Radix-4 Modified Booth Algorithm for 6-bit operand

Table 1. Logic Operation on P from All Possible 3 LSB of Number B

B(1)	B(0)	B(-1)	Logic Operation on P
0	0	0	0
0	0	1	A
0	1	0	A
0	1	1	2A
1	0	0	-2A
1	0	1	-A
1	1	0	-A
1	1	1	0

### B. Implementation of Conventional Radix-4 Booth Multiplier

Radix-4 booth multiplier can be implemented using one  $n$ -bits register for multiplicand, one  $n+1$  bits shift left 2 register for multiplier, and one  $2n$ -bits register to hold multiplication result. Partial product generation will be done by booth encoder and  $n+1$  bits multiplexer system to choose the right partial product. Addition process will be done by  $2n$ -bits adder. And finally, shifting process will be done by using wire arrangement, which consumes no additional component. For example, to create radix-4 booth multiplier with 8-bits input, the required component will be one 8-bits register, one 9-bit shift left 2 register, one 16-bits register, one booth encoder, two 9-bits multiplexer, and 16-bits adder. Block diagram for this implementation can be seen in Figure 2.

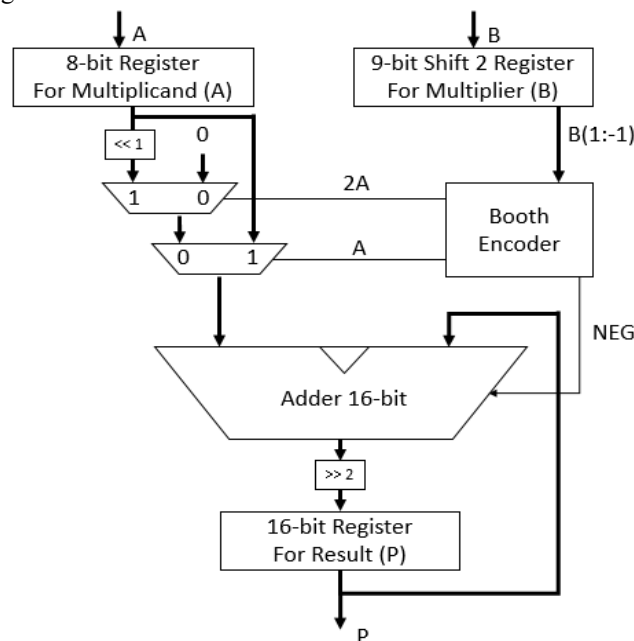


Figure 2. Block diagram for 8-bit Radix-4 Booth Multiplier

## 3. Proposed Radix-4 Booth Multiplier

### A. Radix-4 Modified Booth Multiplier Architecture

The proposed design is focused on the size reduction of basic radix-4 booth multiplier architecture by minimizing two components, i.e. adder and registers.

Register minimization is done by combining result register (P) and multiplier register (B) into a single register. The combination process, based on two observations in Figure 2, is:

- The last  $n$ -bits of P register is actually not used in calculation process and only shifted by 2 to the left until the final result is generated.
- B register is also shifted by 2 to the left for each iteration. Causing the register to gradually be empty relative from the direction of MSB

From this observation, it can be concluded that the last  $n$ -bits in P register can be combined with B register into a single register. Because the size of B register is  $n+1$  bits, the size of the combined register will be  $2n+1$  bits. By using this method, the total number of register bits will be reduced by  $n-1$  bits. The proposed radix-4 booth multiplication algorithm using combined P and B register is shown in Figure 3.

Adder minimization is done from the fact that only  $n+1$  MSB in the P+B register is actually needed to be calculated, while the rest of  $n$ -bits LSB is passed through only. It can be seen from Figure 3 that only the first 6-bits of + sign extension is calculated, while the rest is just passed through. Therefore, it can be concluded that the required adder size is only  $n+1$  bits, which is almost half of the size of the basic implementation, which is  $2n$  bits. The block diagram of our proposed architecture of Modified Radix-4 Booth Algorithm is shown in Figure 4.

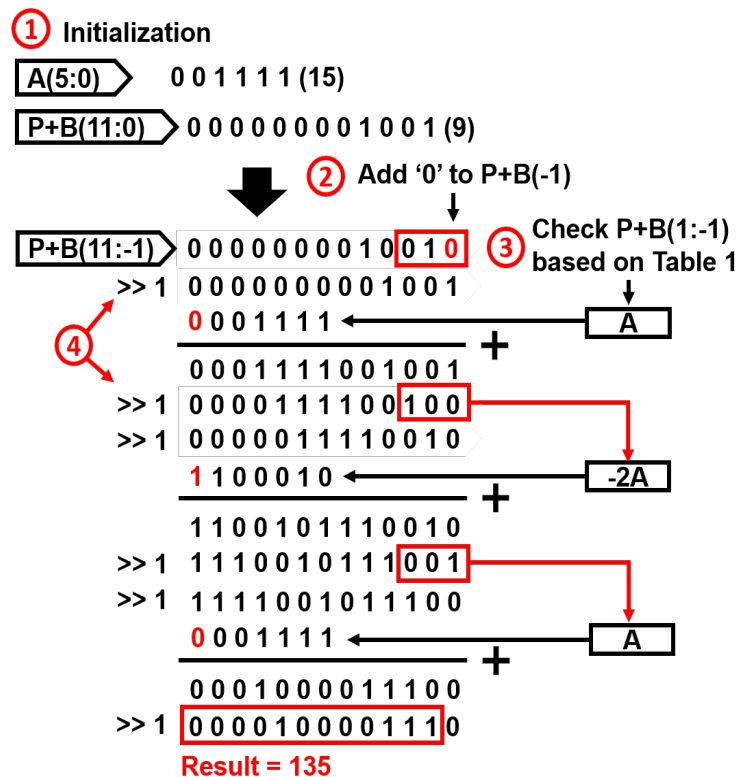


Figure 3. Radix-4 Booth Multiplier Algorithm using combined P and B register for 6-bit operand

Step-by-step of the proposed 8-bit radix-4 booth multiplier algorithm is as follows:

1. Initialize A register with multiplicand input, P+B register with sign-extended multiplier input.
2. Append 'zero' to LSB of P+B register for first calculation.
3. Examine last 3 bits of P+B register to select mode of operation, based on Table 1. Do the arithmetic operation with right-shifted P+B register.
4. Shift the result once to the right, then save it to P+B register. Repeat steps 1 to 3  $n/2$  times. The result is the last P+B register value, discarding its LSB.

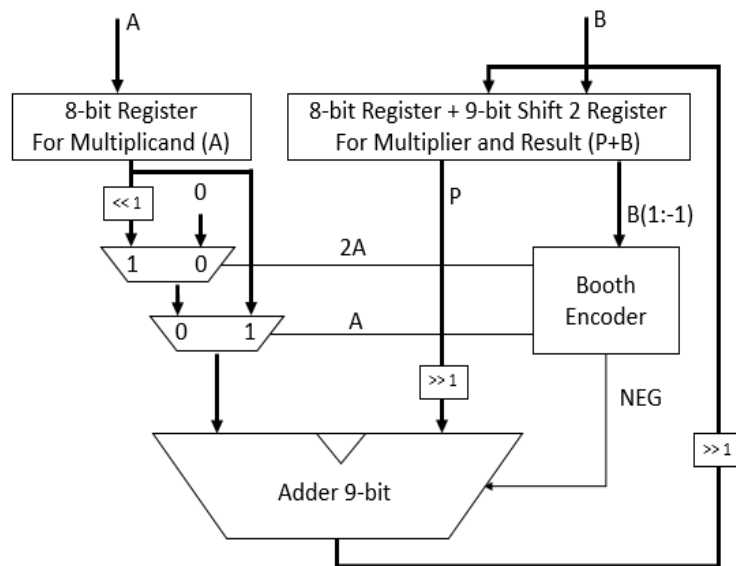


Figure 4. Proposed 8-bit Radix-4 Booth Multiplier design with combined P+B register and reduced adder size

Step-by-step of the proposed 8-bit radix-4 booth multiplier algorithm is as follows:

5. Initialize A register with multiplicand input, P+B register with sign-extended multiplier input.
6. Append 'zero' to LSB of P+B register for first calculation.
7. Examine last 3 bits of P+B register to select mode of operation, based on Table 1. Do the arithmetic operation with right-shifted P+B register.
8. Shift the result once to the right, then save it to P+B register. Repeat steps 1 to 3  $n/2$  times. The result is the last P+B register value, discarding its LSB.

### B. Circuit and Layout Design

All the circuits are implemented using full-custom design approach with the optimization is mainly done in registers, multiplexers, and adder. In conventional approach, register is usually implemented using NAND based D Flip-flop [19]. A NAND gate uses four transistors, while D Flip-flop uses at least four NAND gates. This means D Flip-flop uses at least 16 transistors.

Moreover, one D Flip-flop only handles one-bit data. If the system handles many bits of data, then the number of transistors used in the circuit will be significantly increased. In our approach, to reduce the number of transistors, pass transistor logic was used. Pass transistor logic has been known of its notable reduction in transistor count for various of circuits by elimination of redundant branches in principle [20]. Priya *et al* demonstrated a full adder circuit using only 8 transistors with power consumption of only 5.87 nW [21]. A more compact compressor circuit, using 28 transistors instead of 40 transistors, has also been achieved by implementing pass transistor logic architecture [22]. In our circuit design, the registers and multiplexers were created using pass transistor, as shown in Figure 5. As the result, 1/8 of transistors in conventional register circuit were able to be reduced. This is a significant reduction of the number of transistors, which also expectedly will produce less power consumption.

In a multiplier circuit, one of the most resource consuming circuit is adder circuit since it normally receives various signals/data from different circuits. Therefore, it is essential to optimize adder circuit. There are several types of full adder circuit, i.e. ripple carry, carry look ahead, carry select, kogges stone, etc [23-24]. Each of them has their own advantageous and drawbacks. Ripple carry adder has the simplest circuit implementation architecture amongst all. However, in ripple carry, the output can only be calculated if the carry from the previous stage

has been produced. This in return produces a considerable delay and it cannot be avoided. Carry look ahead eliminates this issue by calculating the carry signals from the input signals. This means additional circuits are needed. It is considered a good solution when the system has no more than 4-bits. Otherwise, the circuit implementation will be much more complicated. In carry select adder, it basically utilizes two ripple carry adders. One is given with constant input of 0 carry-in, while the another one is with 1 carry-in. To select the correct pre-calculated partial sums, a multiplexer is used [25]. Carry select adder is generally considered as the best merit of high speed and less circuit complexity. Kogge-stone adder is actually a variant of carry look-ahead adder, but is constructed in a parallel prefix architecture [26]. One drawback of kogge-stone adder is the wiring complexity, and thus current leakage and/or parasitic capacitance may become a problem when the circuit is large.

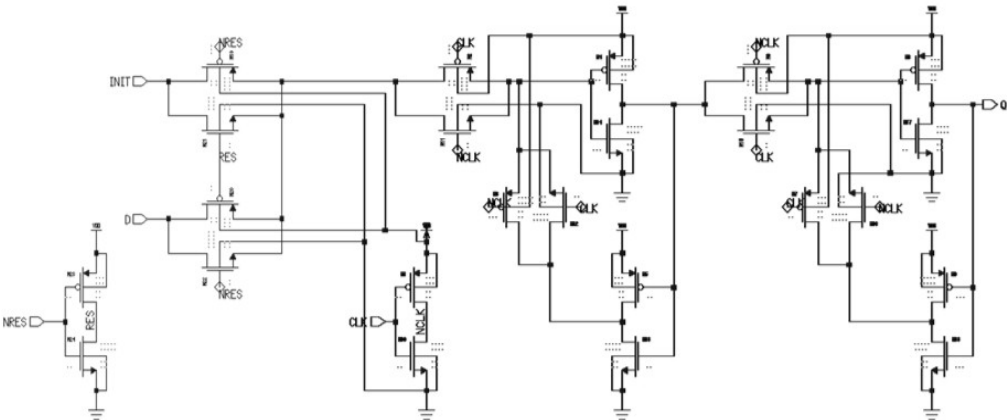


Figure 5. Register circuit with pass transistor on input

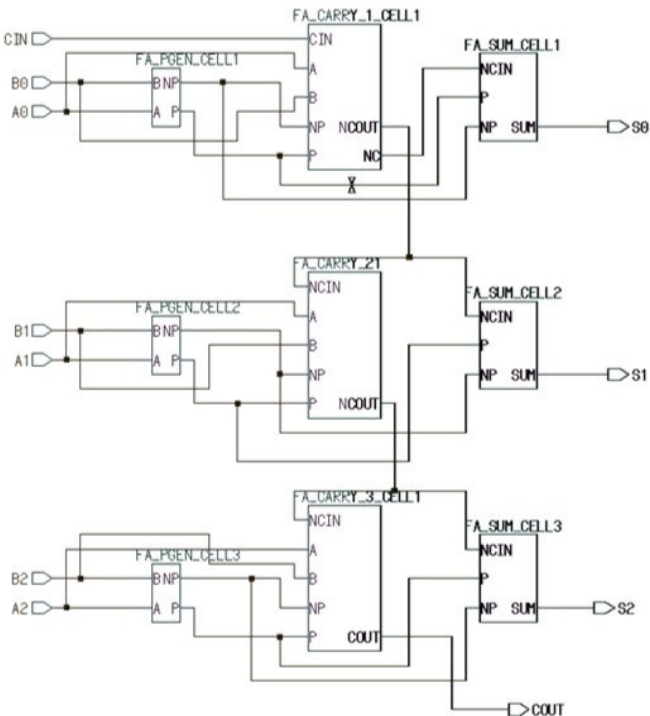


Figure 6. 3-Bit Carry Look Ahead Adder

Figure 6 shows our adder design. We use a combination of ripple carry and carry look ahead adder. In detail, our designed adder is a 3-segment ripple carry adder, in which each segment is a 3-bit carry look ahead adder. The carry look-ahead segment is implemented using static Manchester Carry Chain because of its high speed, small size, and consistent performance [27]. Normal Static Manchester Carry Chain cannot be directly driven by the direct input signal, instead it had to be driven by additional G and N signal to operate [28]. Thus, it requires an additional separate circuit to implement. Basically, additional 1 NAND, 2 NOT, and 1 NOR circuits are needed to generate G and N signal. This will significantly increase the number of transistors in use since these additional circuits is proportional with the number of bits it processes. Therefore, to realize a compact circuit design, it is necessary to overcome this issue. In our circuit design, to eliminate the need to create this additional circuit, the normal Manchester Carry Chain circuit is modified, so that it can directly accept adder input signal A and B without any conversion. The schematic of the modified static Manchester Carry Chain is shown in Figure 7. It can be seen that a PMOS and a NMOS transistor is added to the circuit. These additional transistors will change the logic computation, and thus the adder input signal can be directly used to drive the transistor to generate the output signal. Moreover, by using our modified static Manchester Carry Chain, 12 transistors can be eliminated. With more system bit, our modified circuit will adequately save a huge transistor in use.

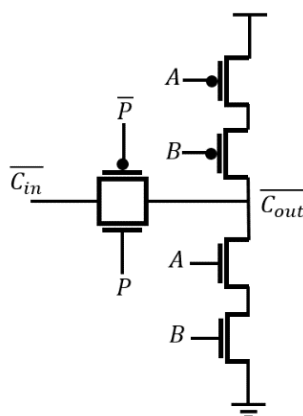


Figure 7. Modified Static Manchester Carry Chain

The circuit layout is designed using Pyxis Mentor Graphics on 0.13 $\mu\text{m}$  CMOS technology. The channel width of PMOS is set to be  $14\lambda$ , while the PMOS is  $6\lambda$ . The layout design is shown in Figure 8. From Figure 8, it can be seen that the total layout area is  $(123 \times 102) \mu\text{m}^2$ . The layout area still consists of some space area, and thus contributes to the enlargement of the layout area. However, the layout area of individual circuit module is considerably small. This shows that our approach contributes to a more efficient layout area.

From Figure 8, it can be seen that the multiplier register along with shift register takes up the biggest layout area amongst all. This is not only because it is a combination of two circuits, but it also consists of a buffer circuit and some inseparable spaces. Moreover, it can also be seen that an additional circuit, Sign Changer, appears in the layout. This additional circuit is actually needed by the booth encoder and the system to correctly compute the bit value of the input data. In terms of layout area, it takes only a small portion of the total layout area-approximately the same area with full adder circuit. This layout area has even already included with a buffer circuit. The other circuit blocks show a very small area consumption as compared to multiplier-shift register circuit. This strengthens the evidence that our proposed circuit design leads to a compact and efficient layout area consumption of radix-4 booth multiplier circuit.

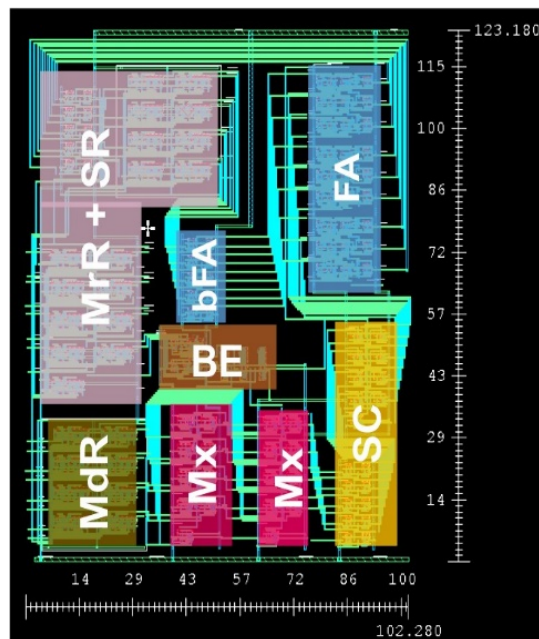


Figure 8. Layout Design of Full-Custom Design of Serial Radix-4 Modified Booth Multiplier.  
 MrR = Multiplier Register, SR = Shift Register, MdR = Multiplicand Register, BE = Booth Encoder, Mx = Multiplexer, SC = Sign Changer, bFA = buffer before Full Adder, FA = Full Adder.

#### 4. Simulation Result

For circuit simulation, the system multiplication functionality is tested by giving 15 and 9 as the input. The system should return 135 as the result. First of all, input A, the 8-bit register, is given a data with a value of 15. It samples the input at the rising-edge of the clock signal. The simulation is shown in Figure 9. As shown in Figure 9, the input data is able to be sampled correctly and with the output delay is below 56 ns.

For input B, the 17-bit-shift-by-2 register, a data with a value of 9 is given to the circuit. In this register circuit, the input is sampled at the rising-edge of the clock signal as well as when the reset signal is ON. As shown in Figure 10, the input data is able to be sampled correctly and the delay of the register is below 45 ns.

The analysis for booth encoder functionality is based on last three bits of the shift-by-2 register. The output of the register should reflect the logic operation described in Table 1. As shown in Figure 11, the output shows three control signals, A, 2A, and NEG. These signals which then control the multiplexers and 9-bit full adder. These control signals also show a perfect match with the functions described in Table 1 which proofs that the module functions correctly. The delay of this booth encoder is less than 125 ns.

For multiplexer, the functionality analysis is based on booth encoder outputs. The multiplexer should have its output be either 0, the same as input A, or twice of input A, depending on booth encoder outputs. As shown in Figure 12, the multiplexer gives the correct output and the delay is less than 141 ns. This delay is a propagation delay which initially comes from registers, then through the booth encoder. Thus, the delay is higher than the delay of booth encoder.

The 9-bit full adder adds or subtracts values from shift-by-2 register and multiplexer output. The operation is controlled by booth encoder. As shown in Figure 13, the full adder correctly works and the delay is less than 323 ns. This delay is also a propagation delay which is accumulated from all the previous modules, i.e. multiplexer, shift-by-2 register, and booth encoder.



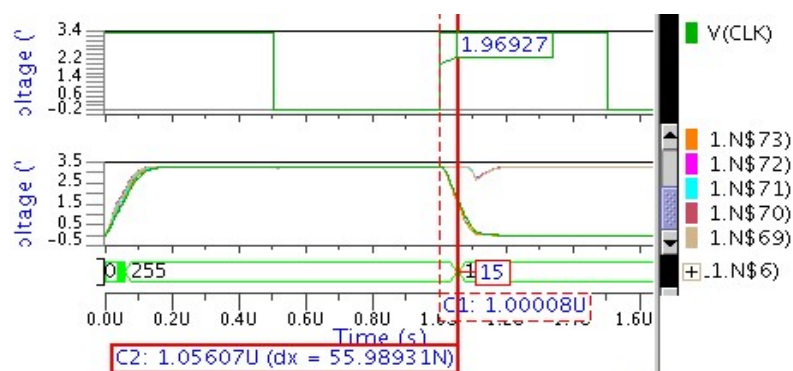


Figure 9. Simulation Result of 8-bit Register Circuit

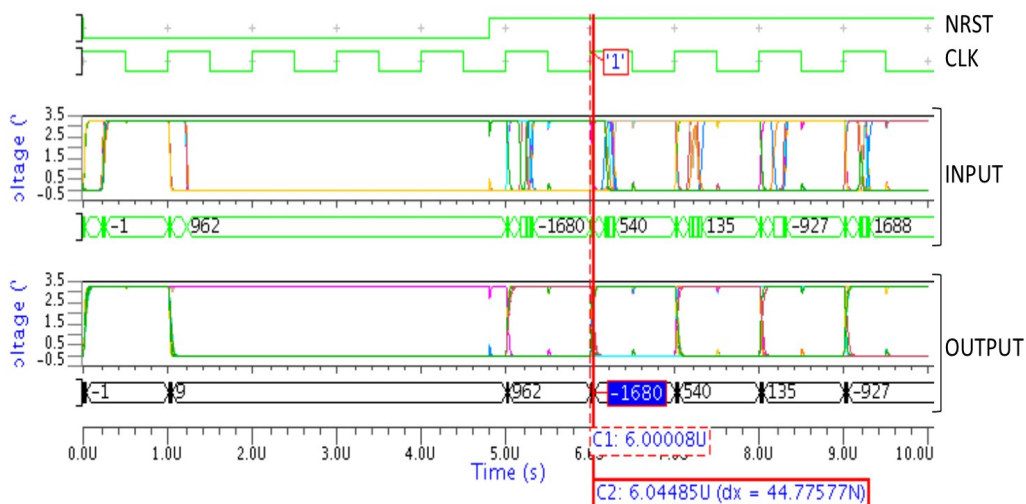


Figure 10. Simulation Result of 17-bit-shift-by-2 Register Circuit

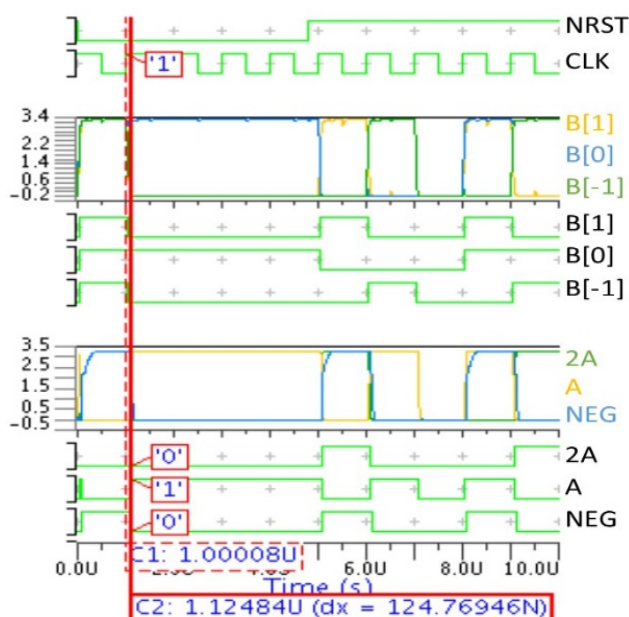


Figure 11. Simulation Result of Booth Encoder Circuit

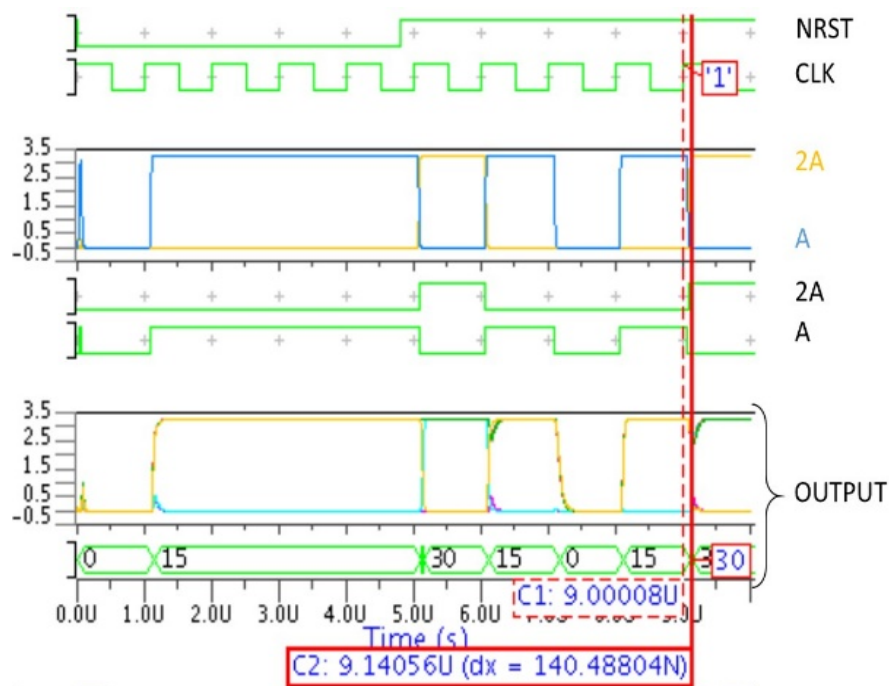


Figure 12. Simulation Result of Multiplexer Circuit

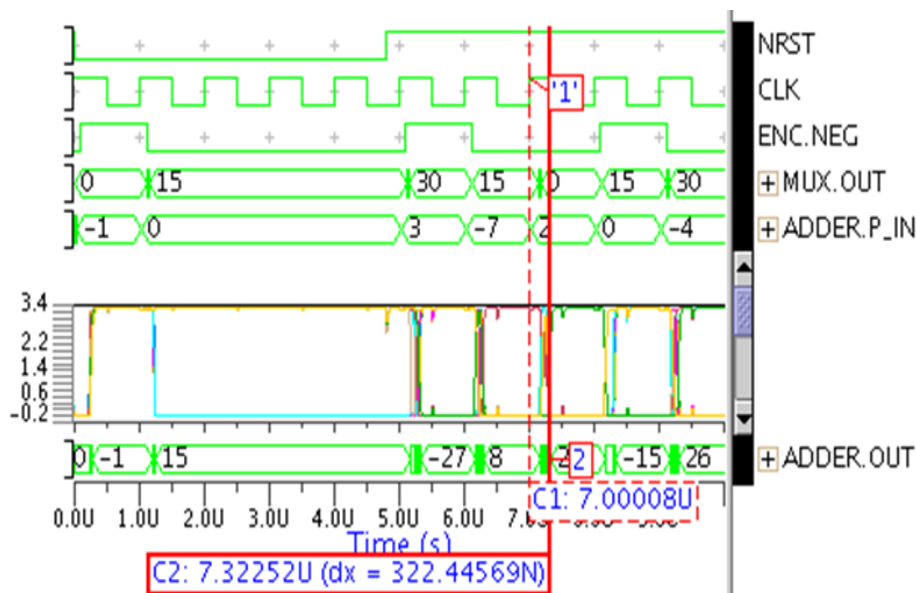


Figure 13. Simulation Result of 9-bit Full Adder Circuit

Figure 14 shows the full circuit simulation with 15 (input A) and 9 (input B) as the input data. The simulation result shows that the multiplication result is 135, which appears at the third rising-edge clock after the reset signal is ON. This result shows that the proposed design successfully calculates the result in half of the total clock needed compare to Serial-Parallel Multiplier design.

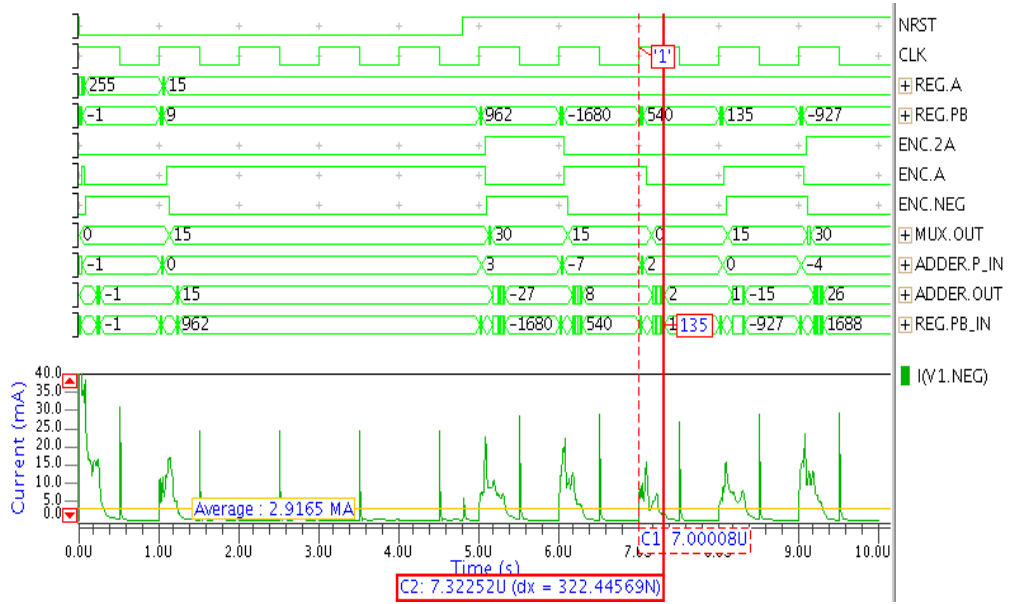


Figure 14. Simulation Result of Full Modified Radix-4 Booth Multiplier Circuit

Table 2. Post-layout simulation result at  $V_{dd} = 3.3V$  and  $T = 25^{\circ}C$ 

Circuit Block	Number of Transistor	Delay (ns)
Adder 9-bit	210	135
Booth Encoder	28	12.5
Multiplexer 9-bit	54	4
Shift Register 9-bit	216	22.5
Full circuit	1132	340

Moreover, a post-layout simulation is also conducted to observe the parasitic effect to the circuit performance. From the post-layout simulation result, shown in Table 2, it can be seen that the adder block produces the longest delay. Moreover, several additional components that are not mentioned in Table 2, such as buffers and parasitic capacitive load between blocks, also contribute to the inter-block delay, and thus worsening the full circuit propagation delay. In details, the 9-bit adder presented in this design requires 210 transistors, or 23.3 transistors per adder bit, while the smallest ripple carry adder, built by using logic gates, requires 24 transistors per adder bit. Additionally, since our adder is a cascaded carry look ahead adder, it has a better performance compare to the pure ripple carry adder.

In overall, the design and performance comparison between our proposed and conventional radix-4 booth multiplier is shown in Table 3. From the comparison data, it can be concluded that our proposed multiplier provides much better performance and design efficiency than the conventional one.

Our proposed multiplier design is optimized for serial multiplier architecture, and thus there is a probability for delay optimization limit. Therefore, other researchers also exploring parallel multiplier architecture [29-31]. However, our design features are at cell level as well, especially the adder circuit. Hence, our design features can also be implemented in any multiplier architectures as long as it uses modified booth multiplier algorithm.

Table 3. Circuit Design and Performance Comparison between Proposed and Conventional Booth Multiplier

Parameter	Proposed Multiplier	Conventional Multiplier	Improvement
Adder block size	Area: 83 $\mu\text{m}^2$ #Transistor: 24	Area: 7 x 83 $\mu\text{m}^2$ #Transistor: 7 x 24	Area: 86% #Transistor: 86%
Register block size	Area: 684 $\mu\text{m}^2$ #Transistor: 24	Area: 8 x 684 $\mu\text{m}^2$ #Transistor: 8 x 24	Area: 87% #Transistor: 87%
Total size	Area: 12599 $\mu\text{m}^2$ #Transistor: 1132	Area: 13870 $\mu\text{m}^2$ #Transistor: 1492	Area: 9% #Transistor: 24%
Delay (ns)	340	405	16%
Power dissipation (mW) at $f = 1$ MHz	10	36	72%

## 5. Conclusions

The implementation of Serial Radix-4 8-bit Modified Booth Multiplier using full-custom approach has been presented. By merging the registers and reducing the adder size, the overall multiplier size can be significantly reduced. From post-layout simulation result and comparing with the conventional radix-4 booth multiplier architecture, the total number of transistors in the proposed multiplier is reduced by 24%, while the total combination delay is reduced by 16%, and the power dissipation is greatly reduced by 72%. Our proposed multiplier design feature is also at cell level, and thus our design feature can be adopted in other multiplier architectures as well. For future works, we would like to implement our design feature for different multiplier algorithms, such as Wallace tree and Vedic multiplier.

## 6. References

- [1]. K. Y. Kyaw, W. L. Goh and K. S. Yeo, "Low-power high-speed multiplier for error-tolerant application", *IEEE International Conference of Electron Devices and Solid-state Circuits*, Hong Kong, China, pp. 1-4, Dec. 15-17, 2010.
- [2]. M. A. Rahman, H. B. Mohamed, M. B. I. Reaz, S. Hamid, Md Ali and W. M. D. W. Zaki, "Design of Low Power 6-bit Digitally-Controlled Oscillator (DCO)", *International Journal on Electrical Engineering and Informatics*, Vol. 6, No. 2, pp. 297-305, 2014.
- [3]. I. A. Khan and M. T. Beg, "Clock Gated Single-Edge-Triggered Flip-Flop Design with Improved Power for Low Data Activity Applications", *International Journal on Electrical Engineering and Informatics*, Vol. 6, No. 3, pp. 562-576, 2014.
- [4]. S. Balamurugan and P. S. Mallick, "Fixed-Width Multiplier Circuits Using Column Bypassing and Decomposition Logic Techniques", *International Journal on Electrical Engineering and Informatics*, Vol. 7, No. 4, pp. 655-664, 2015.
- [5]. Y. C. Tsao and K. Choi, "Area-Efficient Parallel FIR Digital Filter Structures for Symmetric Convolutions Based on Fast FIR Algorithm," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 20, No. 2, pp. 366-371, 2012.
- [6]. S. Wang, W. Lin, J. Ye and M. Shieh, "Fast scalable radix-4 Montgomery modular multiplier," 2012 *IEEE International Symposium on Circuits and Systems (ISCAS)*, Seoul, South Korea, pp. 3049-3052, May 20-23, 2012.
- [7]. E. P. A. Akbar, M. Haghparast and K. Navi, "Novel design of a fast reversible Wallace sign multiplier circuit in nanotechnology", *Microelectronics Journal*, Vol. 42, No. 8, pp. 973-981, 2011.
- [8]. A. Akbar, T. Adiono, S. Harimurti and T. A. M. Putra, "Design of Neuron Net Function using Modified Radix-4 Booth Multiplier with a Flipped Logic Parallel Prefix Adder," 2018 *International Symposium on Electronics and Smart Devices (ISESD)*, Bandung, Indonesia, pp. 1-6, Oct. 23-24, 2018.

- [9]. T. Adiono, H. Ega, H. Kasan, S. Fuada and S. Harimurti, "Full custom design of adaptable montgomery modular multiplier for asymmetric RSA cryptosystem," 2017 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), Xiamen, China, pp. 910-914, Nov. 6-9, 2017.
- [10]. S. Vaidya and D. Dandekar, "Delay-power performance comparison of multipliers in VLSI circuit design," *International Journal of Computer Networks & Communications*, Vol. 2, No. 4, pp. 47-56, 2010.
- [11]. K. L. S. Swee and L. H. Hiung, "Performance comparison review of Radix-based multiplier designs", 2012 4th *International Conference on Intelligent and Advanced Systems (ICIAS)*, Kuala Lumpur, Malaysia, pp. 854-859, June 12-14, 2012.
- [12]. W. Liu, L. Qian, C. Wang, H. Jiang, J. Han and F. Lombardi, "Design of Approximate Radix-4 Booth Multipliers for Error-Tolerant Computing," *IEEE Transactions on Computers*, Vol. 66, No. 8, pp. 1435-1441, 2017.
- [13]. N. Nagamani, R. Nikhil, M. Nagaraj and V. K. Agrawal, "Reversible Radix-4 booth multiplier for DSP applications," 2016 *International Conference on Signal Processing and Communications (SPCOM)*, Bangalore, India, pp. 1-5, June 12-15, 2016.
- [14]. V. R. Raut and P. R. Loya, "FPGA Implementation of Low Power Booth Multiplier Using Radix-4 Algorithm," *International Journal of Advanced Research in Electrical, Electronics, and Instrumentation Engineering*, Vol. 3, No. 8, pp. 11479-11486, 2014.
- [15]. Yosr Bchir, Soufien Gdaim, and Abdellatif Mtibaa, "A Fuzzy Direct Torque Control of Induction Motor for FPGA Implementation", *International Journal on Electrical Engineering and Informatics*, Vol. 8, No. 4, pp. 851-864, 2016.
- [16]. R. Lathal and P. T. Vanathi, "Design of Digital Filters for Multi-standard Transceivers", *International Journal on Electrical Engineering and Informatics*, Vol. 7, No. 3, pp. 517-530, 2015.
- [17]. D. Baran, M. Aktan and V. G. Oklobdzija, "Multiplier structures for low power applications in deep-CMOS," 2011 *IEEE International Symposium of Circuits and Systems (ISCAS)*, Rio de Janeiro, Brazil, pp. 1061-1064, May 15-18, 2011.
- [18]. Pishvaie, G. Jaberipur and A. Jahanian, "Improved CMOS (4; 2) compressor designs for parallel multipliers", *Computers and Electrical Engineering*, Vol. 38, No.6, pp. 1703–1716, 2012.
- [19]. Lin, T. Kuo and Y. Chen, "Implementation of a genetic logic circuit: bio-register", *Systems and Synthetic Biology*, Vol. 9, pp. 43–48, 2015.
- [20]. S. P Philip, S. P. Prakash, and S. Valarmathi, "Design of Efficient Complementary Pass Transistor based Modified Booth Encoder Array Multiplier", *International Journal of Computer Applications*, Vol. 64, No. 5, pp. 25-31, 2013.
- [21]. M. G. Priya and K. Baskaran, "Low Power Full Adder With Reduced Transistor Count", *International Journal of Engineering Trends and Technology*, Vol. 4, No. 5, pp. 1755-1759, 2013.
- [22]. D. Radhakrishnan and A.P. Preethy, "Low Power CMOS Pass Logic 4-2 Compressor for High-Speed Multiplication", *Proceedings of the 43rd IEEE Midwest Symposium on Circuits and Systems*, Lansing, MI, USA, Vol.3, pp. 1296-1298, Aug. 8-11, 2000.
- [23]. J. Saini, S. Agarwal and A. Kansal, "Performance, analysis and comparison of digital adders," 2015 *International Conference on Advances in Computer Engineering and Applications*, Ghaziabad, India, pp. 80-83, March 19-20, 2015.
- [24]. M. D. Ercegovic and T. Lang, "Chapter 2 - Two-Operand Addition" In *Digital Arithmetic*, The Morgan Kaufmann Series in Computer Architecture and Design, San Francisco, CA: Morgan Kaufmann, 2004, pp. 50-135.
- [25]. B. Likhari and Sakshi, "Design and Comparison of Regularize Modified Booth Multiplier Using Different Adders", 2013 *International Conference on Machine Intelligence Research and Advancement*, Katra, India, pp. 387-391, Dec. 21-23, 2013.
- [26]. Y. Chen, M. Zhang, P. Wei, S. Sui, Y. Zhao and L. Xie, "Implementation of a Parallel Prefix Adder Based on Kogge-Stone Tree", 2016 *International Conference on*

- Communications, Information Management and Network Security*, Shanghai, China, pp. 234-237, Sep. 25-26, 2016.
- [27]. W. L. Pang and M. B. I. Reaz, "Performance Evaluation of Manchester Carry Chain Adder for VLSI designer Library", *the 5th WSEAS International Conference on Simulation, Modelling, and Optimization (SMO)*, Corfu, Greece. pp. 685-689, Aug. 17-19, 2005.
- [28]. G. A. Ruiz, "New Static Multi-Output Carry Lookahead CMOS Adders," *IEEE Proceedings - Circuits, Devices and Systems*, Vol. 144, No. 6, pp. 350-354, 1997.
- [29]. P. Amberg, N. Pinckney, D. M. Harris, "Parallel High-radix Montgomery Multipliers," 2008 42nd Asimolar Conference on Signals, Systems and Computers, CA, USA, pp. 772-776, Oct. 26-29, 2008.
- [30]. M. Thomas, "Design and Simulation of Radix-8 Booth Encoder Multiplier for Signed and Unsigned Numbers," *International Journal for Innovative Research in Science & Technology*, Vol. 1, No. 1, pp. 1-10, 2014.
- [31]. B. Sharma, A. Bakshi, "Comparison of 24x24 Bit Multipliers for Various Performance Parameters," *1st International Conference on Advent Trends in Engineering, Science and Technology (ICATEST 2015)*, Amravati, India, pp. 146-149, March 1, 2015.



**Trio Adiono** received the B.Eng. degree in electrical engineering and M.Eng. degree in microelectronics from Institut Teknologi Bandung, Indonesia, in 1994 and 1996, respectively. He obtained his Ph.D. degree in VLSI Design from Tokyo Institute of Technology, Japan, in 2002. He holds a Japanese Patent on "High Quality Video Compression System". He is now a lecturer at the School of Electrical Engineering and Informatics and also serves as the Head of the Microelectronics Center and IC Design Laboratory, Institut Teknologi Bandung. He currently serves as the chair of the IEEE SSCS Indonesia Chapter. His research interests include VLSI design, signal and image processing, smart card, electronics solution design and integration.



**Hans Herdian** received the B.Eng. degree in electrical engineering from Institut Teknologi Bandung, Indonesia, in 2016. Currently, he is a doctoral student at Tokyo Institute of Technology, Japan, where he received his M. Eng. degree. He joins the Matsuzawa and Okada Laboratory where he conducts a research on GHz RF circuit design. His research interests include VLSI/IC design for high speed communication and digital signal processing.



**Suksmandhira Harimurti** received the B.S. degree in engineering physics from Institut Teknologi Bandung, Indonesia, in 2012, and the M.Eng. degree in electrical engineering from The University of Tokyo, Japan, in 2015. Currently, he is a researcher at the Microelectronics Center, Institut Teknologi Bandung. His research interests include VLSI and RF analog integrated circuits design for smart card and biomedical application.



**Tengku Ahmad Madya Putra** received bachelor degree in electrical engineering from Institut Teknologi Bandung, Indonesia. He has been working in the fields of silicon-level analog front-end for NFC since 2016 and also involved in the development of System-on-Chip design since 2019. He is currently a master student in Microelectronics, Department of Electrical Engineering Institut Teknologi Bandung. His research interests include chip design, near-field communication, and silicon-level mixed-signal circuitry.