Journal of Computer and Communications, 2021, 9, 54-69
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

A Low-Area, Low-Power Dynamically Reconfigurable 64-Bit Media Signal Processing Adder

Priscilla Sharon Allwin*, Chien-In Henry Chen

Wright State University, Dayton, OH, USA
Email: *allwin.2@wright.edu

How to cite this paper: Allwin, P.S. and Chen, C.-I.H. (2021) A Low-Area, Low-Power Dynamically Reconfigurable 64-Bit Media Signal Processing Adder. *Journal of Computer and Communications*, **9**, 54-69.
https://doi.org/10.4236/jcc.2021.93004

Received: February 14, 2021
Accepted: March 19, 2021
Published: March 22, 2021

Abstract

Multimedia devices like cellphones, radios, televisions, and computers require low-area and energy-efficient dynamically reconfigurable data paths to process the greedy computation algorithms for real-time audio/video signal and image processing. In this paper, a novel low-area, energy-efficient 64-bit dynamically reconfigurable adder is presented. This adder can be run-time configured to different reconfigurable word lengths based on the partition signal commands provided. Moreover, the design is partitioned into sub-blocks based on functionality to save power, *i.e.*, configuring the computation only for the necessary data path, thus avoiding the unnecessary switching power from the data path computed values that do not get used. Only functions that are needed are powered on, and the rest of the functionality is powered off. The proposed 64-bit dynamically reconfigurable media signal processing (MSP) adder is implemented in the 180 nm CMOS technology at 1.8 V, requiring an area of 39,478 μm^2 and a power of 79.24 mW. The dynamic MSP adder achieves a 15.7% reduction in area and a 59.2% reduction in power than the 64-bit MSP adder.

Keywords

Media Signal Processing (MSP), Reconfigurable Adder, Dynamic Reconfiguration

1. Introduction

Multimedia systems play an essential part in our daily lives and have drastically improved the quality of life over time. Adders are the fundamental data path blocks in media signal processors present in electronic devices such as cellphones, radios, televisions, and computers. These devices require low-area,

low-power reconfigurable adders to process real-time greedy computation algorithms such as discrete cosine transform [1] [2] [3], inverse discrete cosine transform, fast Fourier transform, etc. Configuring an architecture on-the-go such as dynamic configuration, allows a system to be dynamically modified during its normal operation without the need for resetting the remaining circuitry or removing any reconfigurable blocks for programming. The reconfiguration also optimizes the necessary component count and power consumption, making it suitable for data path components in media signal processing, networking, and cryptography [4]-[18].

Several reconfigurable adder architectures have been proposed for MSP applications. A reconfigurable adder uses an 8-bit carry generation block as the basic unit and a prefix-based controlled carry grouping logic to reduce power consumption [9]. A reconfigurable carry-skip adder for minimal energy and high-speed performance was proposed in [11]. A reconfigurable SQRT-CSLA with a modified ripple carry adder for high speed is proposed in [12]. A hybrid reconfigurable adder architecture combined with variants like RCA, CLA, CSLA for reduced power and area is proposed in [13]. A reconfigurable adder targeted for Binary/BCD addition for high-speed performance was proposed in [19].

In media signal processing, the data path switching, such as in adders and multipliers, can substantially consume switching power. We aim to configure the computation only for the necessary data path, thus avoiding the unnecessary switching power from the data path computed values that do not get used. This paper focuses on a novel 64-bit reconfigurable adder architecture for MSP applications with reduced area and power consumption. Moreover, the reconfiguration of the proposed design is more dynamic than the existing design [11] and requires less area and power.

The proposed 64-bit dynamic reconfigurable adder includes the second stage of partition to increase the reconfigurability. This dynamic configuration is a unique feature that has not been explored in previous designs [9] [11] [12] [13] [19]. Multimedia Signal Processors perform millions of complex computations within seconds; therefore, they require a high-speed, low-power data-path to store and compute the signals that arrive at it. The high switching activity in the data-path components contributes to higher power consumption, which has been minimized in the proposed adder by using a new reconfiguration scheme. It gives more flexibility and control over the choice of partition, which minimizes data path hardware usage by turning off the unused components and routing the signals efficiently, leading to lower power consumption.

Section 2 describes the logic behind the 64-bit CSMT reconfigurable adder and then the 64-bit dynamic CSMT reconfigurable adder.

2. 64-Bit Reconfigurable Adder for Media Signal Processing

2.1. Reconfigurable Adder

The architecture explained in [11] is taken as a reference in our first 64-bit

CSMT reconfigurable adder design. Figure 1 shows the architecture of the proposed adder. It consists of a series of non-uniform linearly increasing blocks of the order such as 1-bit, 3-bit, 4-bit, 5-bit, 6-bit, 7-bit, 8-bit, 9-bit, 10-bit, and 11-bit, with 1-bit being the Least Significant Bit (LSB) block and 11-bit being the Most Significant Bit (MSB) block. The proposed design performs run-time reconfiguration for one 64-bit, two 32-bit, four 16-bit, or eight 8-bit additions based on the partition signal command provided to it. The on-demand reconfiguration is made possible with two control signals **P0** and **P1**. Since the 8-bit addition is the smallest precision block, the least significant blocks such as 1-bit, 3-bit, and 4-bit blocks do not require any partitioning.

Table 1 provides the partition configuration of the proposed adder design. Each block on the table has two sub-blocks of bits (x-bit, y-bit) where x is the least significant and y is the most significant. When partitioning is required, the control signals (**P0** and **P1**) ensure that no carry propagation occurs between these two separate sub-blocks, x-bit, and y-bit. Consider the case of eight 8-bit additions, where every MSB block (5-bit to 11-bit) requires partitioning. The 5-bit block is split in the 1st-bit position, and the 6-bit block is split in the 3rd-bit position, and there is no carry propagation between the two partitioned blocks. Table 1 shows how each block is divided concerning the configuration of addition.

2.2. Dynamic CSMT Based 64-Bit Adder

Figure 2 shows a new modified 64-bit reconfigurable CSMT adder, which is

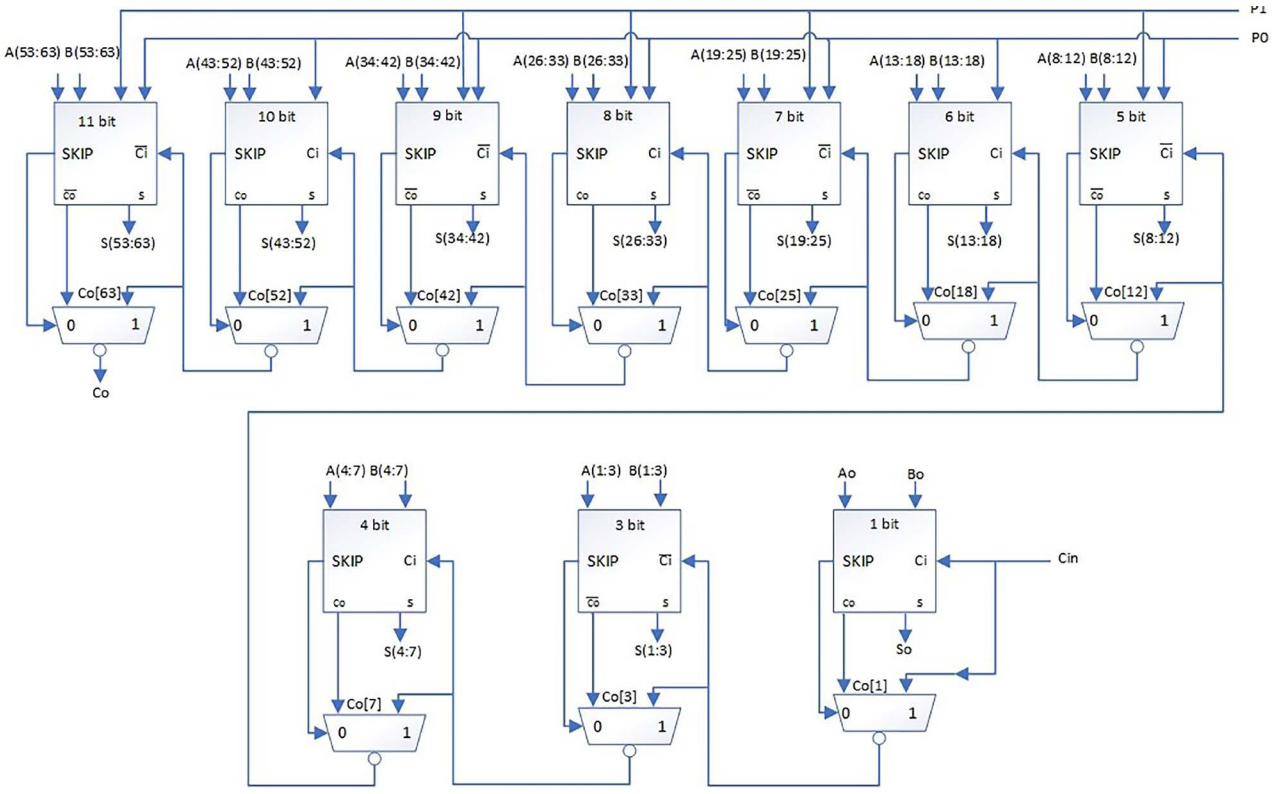


Figure 1. The first proposed 64-bit reconfigurable MSP adder.

Table 1. Partition commands for the first 64-bit reconfigurable MSP adder.

Sub-blocks	P0 = 0 P1 = 0	P0 = 0 P1 = 1	P0 = 1 P1 = 0	P0 = 1 P1 = 1
	64-bit	32-bit	16-bit	8-bit
5-bit	Previous block carry	Previous block carry	Previous block carry	Original Cin
6-bit	-	-	(3-b, 3-b)	(3-b, 3-b)
7-bit	-	-	-	(2-b, 5-b)
8-bit	-	(2-b, 6-b)	(2-b, 6-b)	(2-b, 6-b)
9-bit	-	-	-	(3-b, 6-b)
10-bit	-	-	(5-b, 5-b)	(5-b, 5-b)
11-bit	-	-	-	(8-b, 3-b)

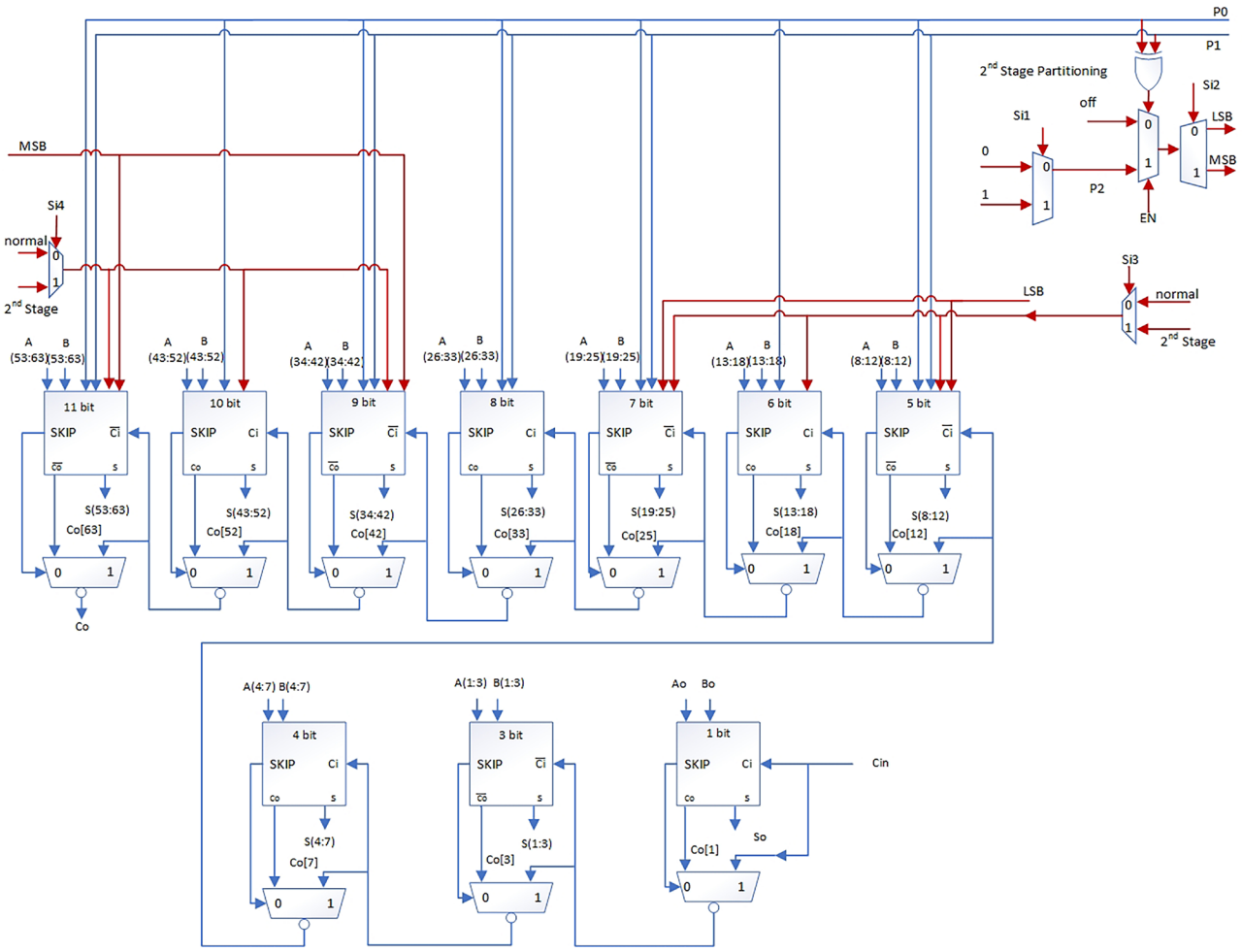


Figure 2. The proposed 64-bit dynamically reconfigurable MSP adder.

dynamic, meaning the adder can be further partitioned into several other configurations along with the existing one (Figure 1) to enhance on-demand media signal processing. The red lines indicate the additional signals that were added to the existing architecture to make it dynamic. The second stage of partitioning is

proposed for this purpose, which is explained in Table 2. The second stage becomes active for additional partitioning when the original configuration performs either two 32-bit additions or four 16-bit additions. The partition signals (**P0** and **P1**) and the control signals **Si1**, **Si2**, **Si3**, and **Si4** (enable signals of the control multiplexers) would allow users to configure the adder according to their requirements. The command signals and their functions are explained in Table 3.

Control signal **Si1** enables the user to decide between either a 16-bit partition or an 8-bit partition, based on the enable signal given to the multiplexer M1. The demultiplexer M3 gets an input (**P2** or **off**) from multiplexer M2 depending on the XOR output. This M3 directs the partition value signal (**P2**) to either the most significant 32-bit/16-bit block or the least significant 32-bit/16-bit block depending on the enable signal **Si2**. The multiplexers M4 and M5 select either the LSB or MSB side of the adder, respectively. The enable signals **Si3** (MUX M4) and **Si4** (MUX M5) choose between maintaining the original configuration or switching to the 2nd stage partitioning for each side individually.

Table 2. The additional second stage partition configuration in the proposed 64-bit dynamically reconfigurable MSP adder.

	P0 = 0 **P1 = 0**	**P0 = 0** **P1 = 1**			**P0 = 1** **P1 = 0**		**P0 = 1** **P1 = 1**
Sub-blocks	**64-bit**	**32-bit**			**16-bit**		**8-bit**
		32-bit **Normal**	**P2 = 0** **16-bit**	**P2 = 1** **8-bit**	**16-bit** **Normal**	**P2 = 1** **8-bit**	
5-bit	Previous block carry	Previous block carry	Previous block carry	Original Cin	Previous block carry	Original Cin	Original Cin
6-bit	--	--	(3-b,3-b)	(3-b,3-b)	(3-b, 3-b)	(3-b, 3-b)	(3-b, 3-b)
7-bit	--	--	--	(2-b,5-b)	--	(2-b, 5-b)	(2-b, 5-b)
8-bit	--	(2-b, 6-b)	(2-b, 6-b)	(2-b, 6-b)	(2-b, 6-b)	(2-b, 6-b)	(2-b, 6-b)
9-bit	--	--	--	(3-b, 6-b)	--	(3-b, 6-b)	(3-b, 6-b)
10-bit	--	--	(5-b,5-b)	(5-b,5-b)	(5-b, 5-b)	(5-b, 5-b)	(5-b, 5-b)
11-bit	--	--	--	(8-b,3-b)	--	(8-b, 3-b)	(8-b, 3-b)

Table 3. Control signals and their functions.

Si1	**Partition**
0	16-bit
1	8-bit
Si2	**Function**
0	LSB
1	MSB
Si3/Si4	**Function**
0	Original Partitioning
1	2nd stage partitioning

Section 3 describes the design process of the proposed 64-bit reconfigurable MSP adder and the 64-bit dynamically reconfigurable MSP adder. Subsection 3.1 further explains the design and operation of the internal sub-blocks for the regular partition and the second stage partition in detail.

3. Design Implementation of the 64-Bit Dynamically Reconfigurable MSP Adder

The Carry Select Modified Tree adder is a multiplexer-based design with low latency and low energy consumption [20] [21]. The proposed 64-bit dynamic reconfigurable adder adopts the CSMT based adder to build the basic reconfigurable block. Figure 3(a) shows a bit slice of the multiplexer-based adder that is designed using the CSMT principle proposed in [21]. Consider a simple addition function, say

$$Y = A + B$$

where $Y = Y_{w-1}....Y_0$, $A = A_{w-1}....A_0$ and $B = B_{w-1}....B_0$ represents W-bit binary numbers. A multiplexer-based design is implemented by defining a new recoding where,

$$A_{ir} = A_i \cdot B_i$$

$$B_{ir} = A_i + B_i$$

This creates a don't care condition, thereby reducing the complexity of the circuit. The equations for sum and carry can be expressed as,

$$S_{i+1} = C_i \left(B_{ir} + A_{ir} \right) + C_i \left(A_{ir} + B_{ir} \right)$$

$$C_{i+1} = C_i \cdot B_{ir} + C_i \cdot A_i$$

3.1. Sub-Block Operation

The sub-blocks fall into two categories, the one that requires partitioning and the one that does not need it. The partitioning decision is determined by the partition commands in Table 2 and the control signals in Table 3. The reconfiguration is designed in such a way as to avoid unnecessary circuitry in the carry propagation path between the configured blocks.

3.1.1. Design of the Least Significant Bit Blocks

The 1-, 3- and 4-bit blocks are the LSB blocks of the design, and they do not require any configuration since the lowest bit operation is an 8-bit addition (1 + 3 + 4 = 8). The design is simple, and straight forward. Figure 3(a) shows the 1-bit block design having inputs **A**, **B**, and $\mathbf{C_i}$, where C_i becomes the select line of the multiplexers to generate the sum (**S**) and Carry out ($\mathbf{C_o}$). The SKIP signal (SKIP = $\overline{\mathbf{A} \cdot \mathbf{B}}$) is used as a select line for the Inverted multiplexer of the carry skip circuit present for each block. This block is then extended to design the 3-bit and 4-bit adder blocks, where the $\mathbf{C_o}$ of the 1st-bit becomes the $\mathbf{C_i}$ value for the next bit. The design of the 4-bit adder block is shown in Figure 3(b).

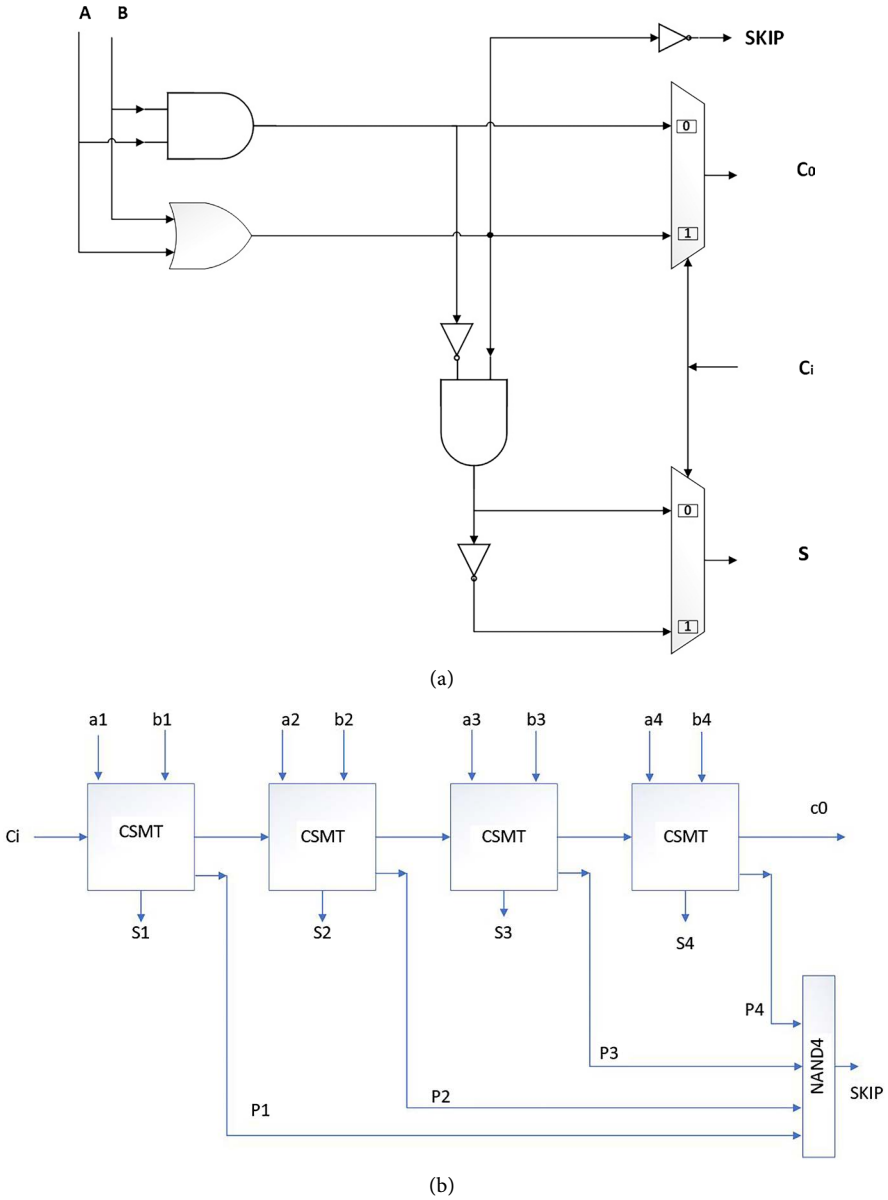


Figure 3. (a) 1-Bit CSMT based adder; (b) 4-Bit adder sub-block extended using the 1-bit block

3.1.2. Design of the Most Significant Bit Blocks

The design of the MSB blocks is essential since it involves the partition process. These partitions are sub-grouped into three categories for a better understanding of the configuration. Based on the partition commands given to the blocks, they are:

1) 6-bit and 10-bit blocks—require partition only for 8- and 16-bit additions

2) 5-bit, 7-bit, 9-bit, 11-bit blocks—require partition only for 8-bit addition

3) 8-bit block—require partition only for 8-, 16- and 32-bit addition.

The 6-bit and 10-bit design uses a simple inverter and a multiplexer to decode the partition command $\overline{\mathbf{P0}}$. The 6-bit block is partitioned as (3-b, 3-b) sub-block and the 10-bit block is partitioned as (5-b, 5-b) sub-block. Depending on the

decoded value, the previous block's carry out is given to the current block or it by-passes to the next block. Figure 4(a) shows the 6-bit design without the second stage partition. The second stage partition is introduced into the design by adding a multiplexer. Figure 4(b) shows the signal flow for both the original partition and the second stage partition, where the red lines indicate the 2nd stage partition flow.

The 5-, 7-, 9- and 11-bit designs require partition only when performing eight 8-bit additions; therefore, a NAND gate along with a multiplexer is used to decode the partition command $\overline{\mathbf{P0}\cdot\mathbf{P1}}$. The blocks are partitioned as described in Table 2. The design of a regular 5-bit and a 9-bit block is presented in Figure 5(a) and Figure 5(b). By adding two multiplexers and an inverter, the second

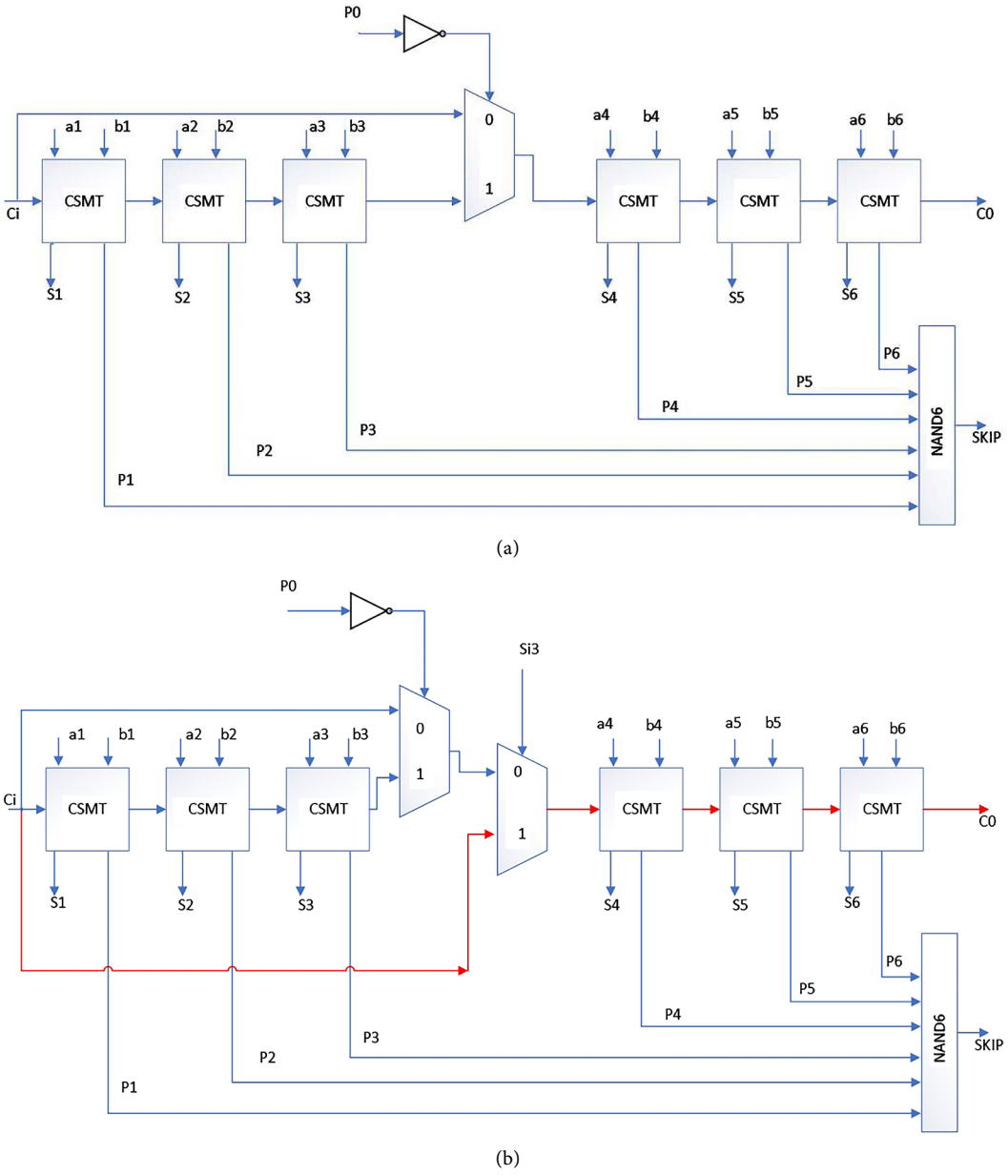
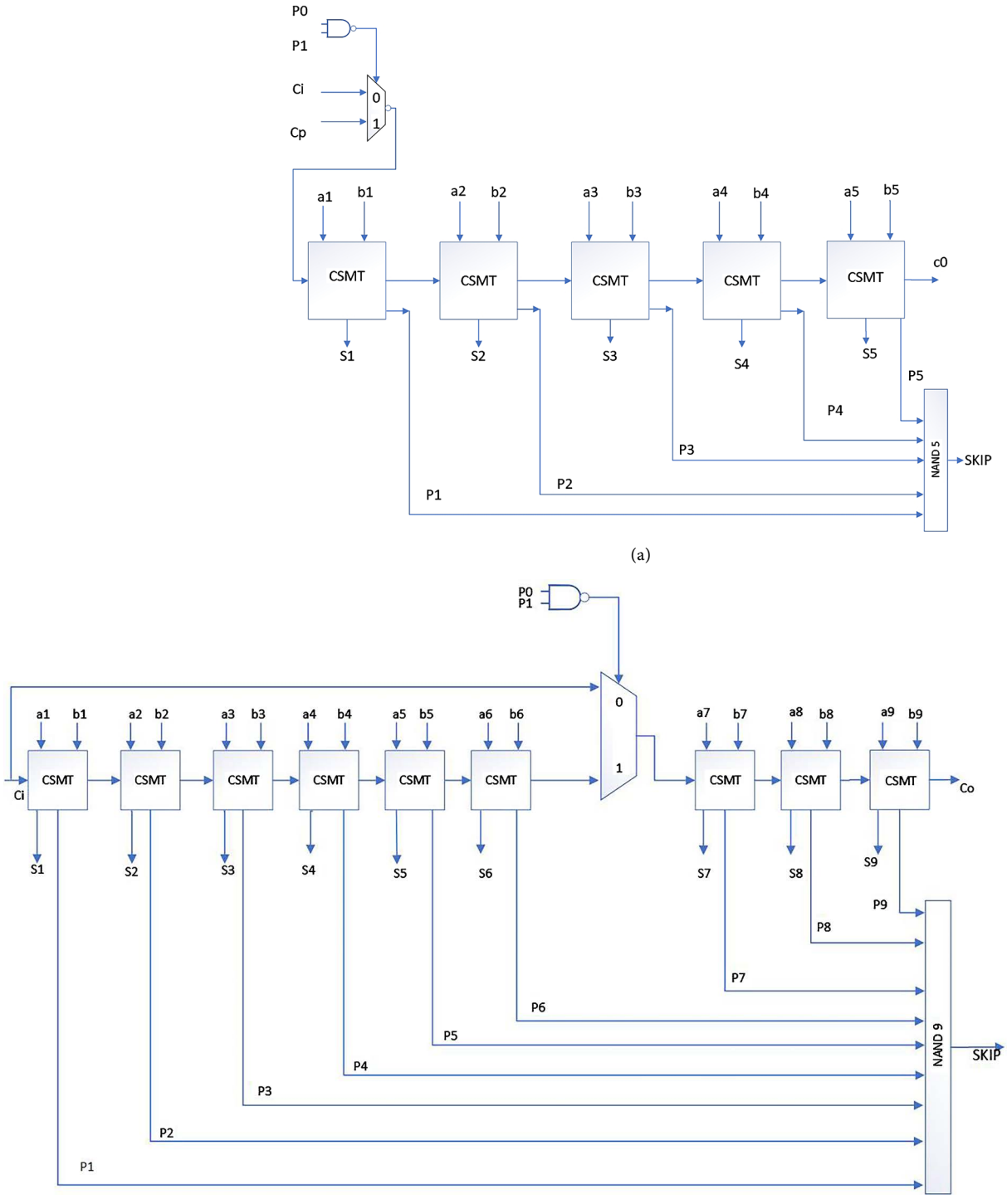
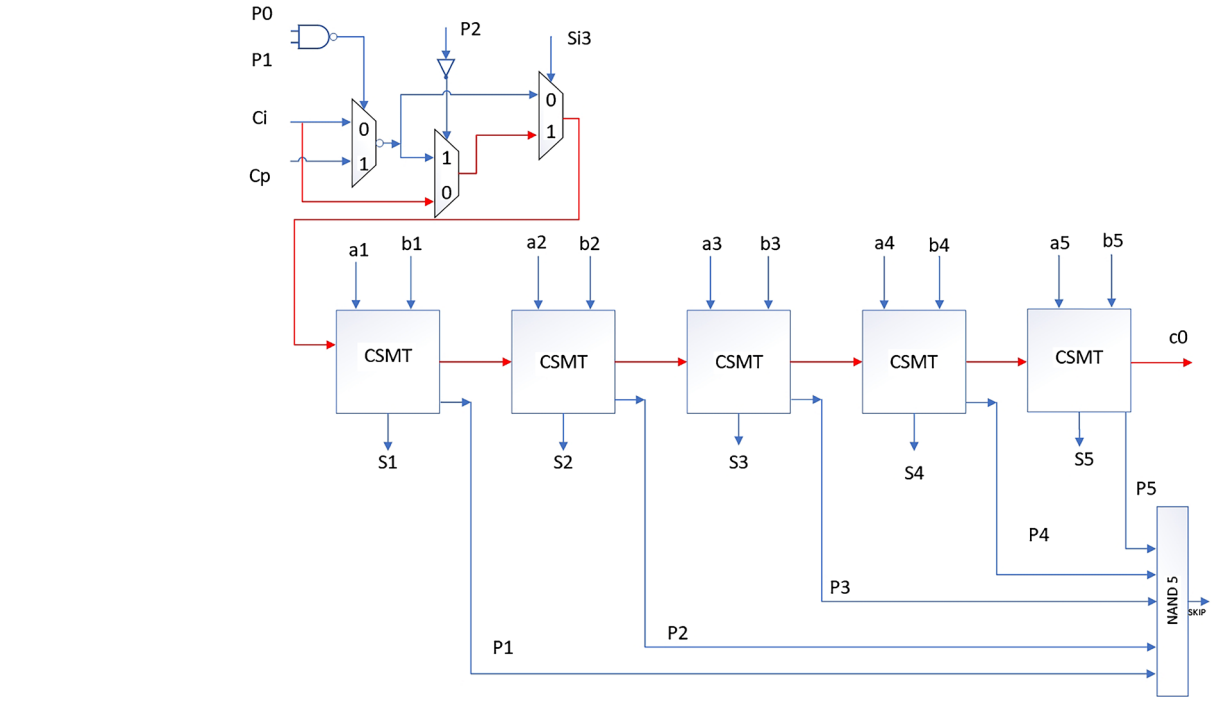


Figure 4. (a) Design of a 6-bit CSMT adder; (b) Second stage partition design of the 6-bit block.

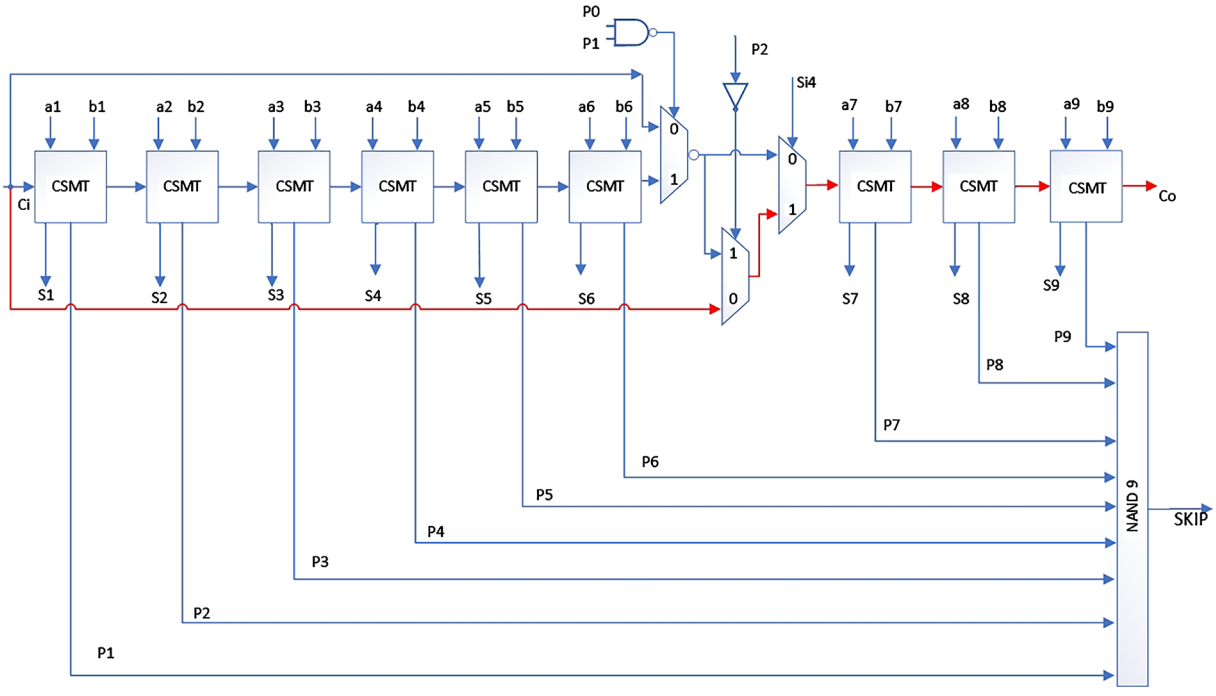
stage partition can decode the partition command. Figure 5(c) and Figure 5(d) show the 5-bit and 9-bit design signal flow for the second stage partition. The red lines show the flow of the second stage partition signals.

The 8-bit design requires a constant partition unless the 64-bit addition is performed. A multiplexer and a NOR gate ($\overline{\mathbf{P0}\cdot\mathbf{P1}}$) are used to decode the partitioned signal coming into the block. Figure 6 describes the 8-bit block partitioning.





(c)



(d)

Figure 5. (a) Design of a 5-bit CSMT adder block; (b) Design of a 9-bit CSMT adder sub-block; (c) Second stage partition of the 5-bit block; (d) Second stage partition of the 9-bit block.

This block does not require any additional partition for the 2nd stage partition.

The operation of the 64-bit dynamic MSP adder which is built using these sub-blocks and the significance of the second stage partition is explained in detail in the following section.

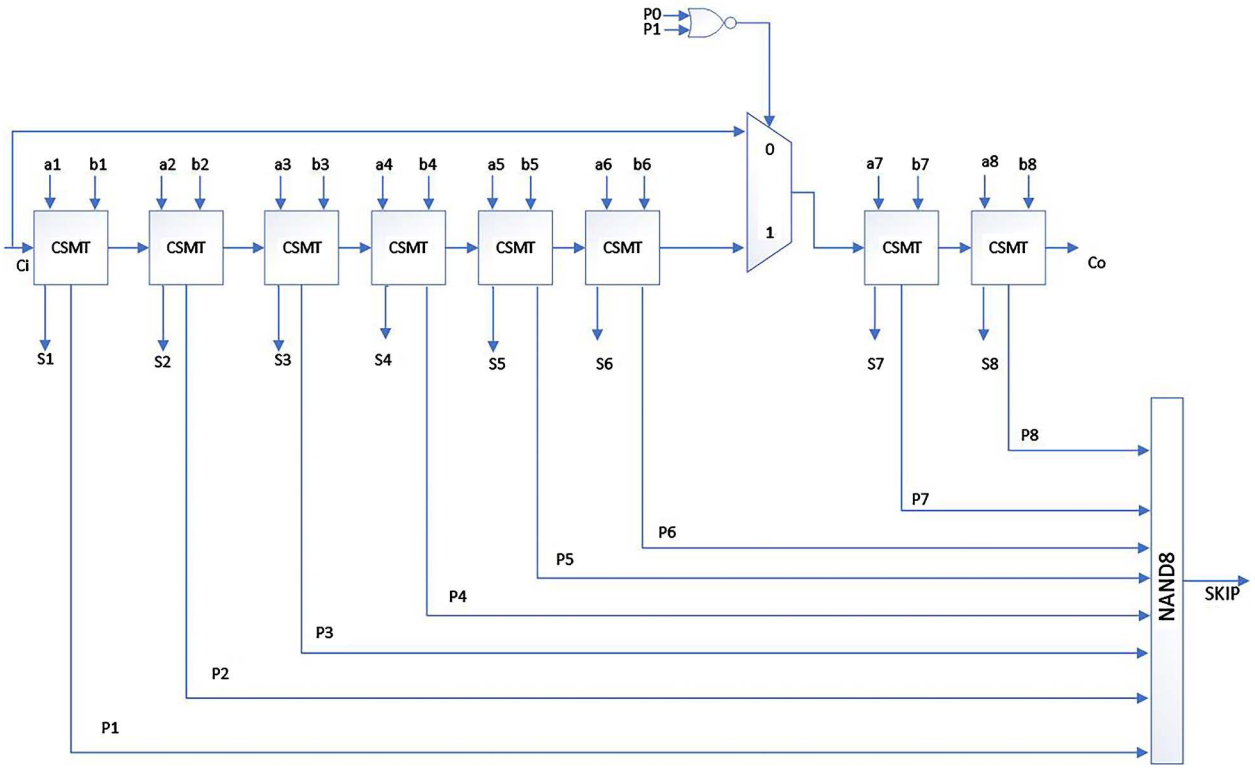


Figure 6. Design of the 8-bit CSMT adder block.

4. The Dynamically Reconfigurable CSMT Based 64-Bit MSP Adder Operation

Figure 7 shows how the adder is configured to perform two 32-bit additions using the CSMT based 64-bit MSP adder. The highlighted green boxes indicate each of the individual 32-bit adders. The control signals **P0** and **P1** are set to **0** and **1** respectively to perform the 32-bit additions concerning the partition commands provided in Table 1. The control signals **P0** and **P1** activate the 8-bit block (highlighted in black) to be partitioned as (2-b, 6-b) sub-blocks and ensures that no carry bit is propagated among these sub-blocks. The drawback of this adder is that it can have only a single configuration model at a given time.

A second stage configuration is introduced in the proposed 64-bit dynamically reconfigurable adder to solve this problem. Figure 8 shows the proposed CSMT based dynamically reconfigurable adder that has additional configuration options. Let us take the two 32-bit addition case. As a supplement to the first stage 32-bit partition, the second stage control signals **Si1**, **Si2**, **Si3/Si4** allow the user to configure additional partitions to the adder if needed. Among the two 32-bit adders, either of the adders can be further partitioned to perform two 16-bit or four 8-bit additions.

In Figure 8, we take the LSB 32-bit adder for partitioning. The MSB still performs the original 32-bit addition. The green highlighted sections indicate the first stage 32-bit adders obtained by setting **P0 = 0** and **P1 = 1**, therefore splitting the 8-bit block (highlighted in black) into (2-b, 6-b) sub-blocks. The red

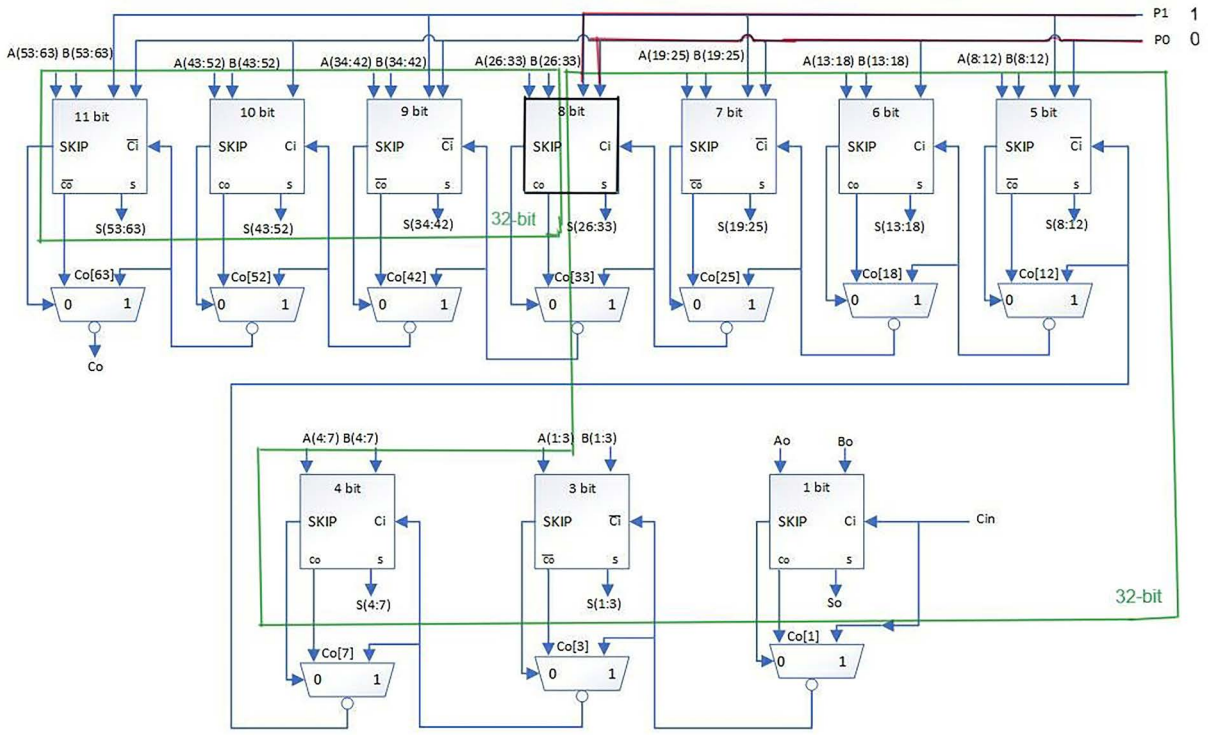


Figure 7. Data path partition for the CSMT based 64-bit MSP adder.

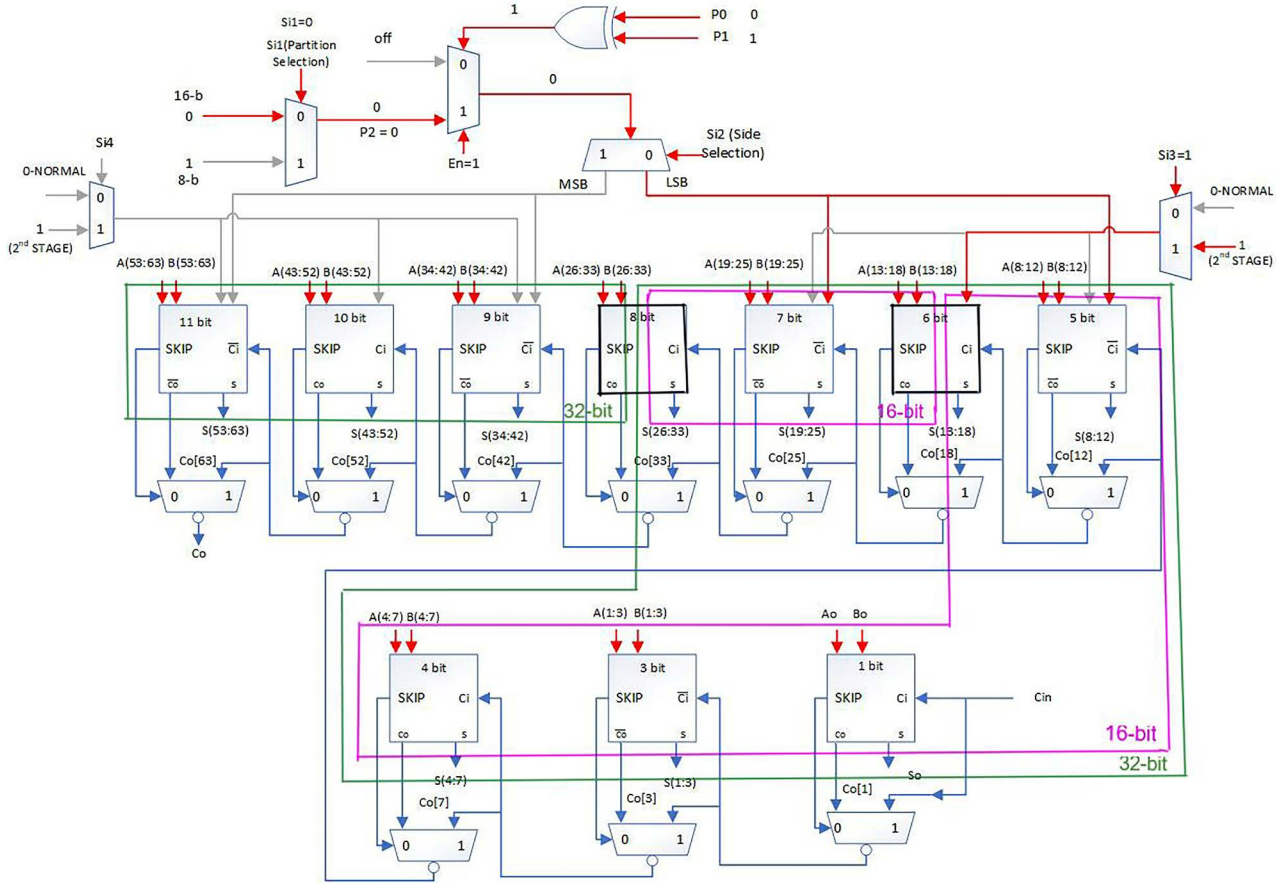


Figure 8. Data path partition for the CSMT based dynamically reconfigurable MSP adder.

arrows indicate the active data path signals, and the grey arrows indicate the inactive data path signals. The second stage of partition is activated when the first stage of configuration is either 32-bit or 16-bit.

In Figure 8 the XOR output is 1 for the 32-bit configuration (**P0 = 0**, **P1 = 1**). The second stage control signal **Si1** is set to **0**, **Si2** is set to **0**, and **Si3** is set to **1**, according to Table 3. The enable signal **En = 1**. The corresponding partition selection is then sent to **P2**, either selecting the LSB or the MSB based on the **Si2** value. For an example of two 16-bit adder configurations, the 6-bit block (highlighted in black) is split as (3-b, 3-b) sub-blocks and no carry propagation occurs among the sub-blocks. The 16-bit blocks are highlighted in pink within the 32-bit green block.

The dynamically reconfigurable adder has two advantages:

1) Internal partitions can be run-time configured without having to replace the adder blocks.

2) The MSP adder's power consumption is significantly reduced since the MSP adder computes only for the necessary data path, thus avoiding the unnecessary switching power from the data path computed values that do not get used. The performance evaluation of the CSMT based 64-bit MSP adder, the CSMT based dynamically reconfigurable 64-bit MSP adder and the 64-bit MSP adder [11] in terms of area, power and delay are explained in the next section.

5. Performance Evaluation

Table 4 shows the area, power, and delay comparison for the three designs. They are 1) the 64-bit MSP adder [11], 2) the proposed CSMT based 64-bit MSP adder, and 3) the proposed dynamically reconfigurable CSMT based 64-bit MSP adder. All three designs were implemented, verified, synthesized, and optimized in CMOS 180 nm technology.

In comparison with the 64-bit MSP adder [11]:

1) the proposed CSMT based 64-bit MSP adder has a 23% reduction in area and a 53.1% reduction in power, and

2) the proposed dynamically configurable CSMT based 64-bit MSP adder has a 15.7% reduction in area and a 59.2% reduction in power.

When compared with the proposed CSMT based 64-bit MSP adder, the dynamically configurable CSMT based 64-bit MSP adder has a 13% reduction in power. The area is increased by 9.4% because of the second stage of partition

Table 4. Area, power and delay comparison results.

Parameters	64-bit MSP adder [11]	The proposed CSMT based 64-bit MSP adder	The proposed Dynamically configurable CSMT based 64-bit MSP adder
Area (μm^2)	46,851	36,085	39,478
Power (mW)	194.19	91.02	79.24
Delay (ns)	19.94	20.67	21.53

that configures the MSP computation only for the necessary data path, thus avoiding the unnecessary switching power from the data path computed values that do not get used.

6. Conclusion

It is nearly impossible to achieve a design that takes up less area, consumes less power, and runs at high speed simultaneously since there is always a trade-off. This paper has presented a low-area, low-power CSMT based 64-bit MSP adder, which can be run-time reconfigured to perform either eight 8-bits, four 16-bits, two 32-bits, or one 64-bit addition based on the partition command. This circuit is further optimized in power by adding a second stage of partition to make the design more dynamically reconfigurable. It gives the user flexibility and control over the choice of partitioning for media signal processing. The proposed CSMT based 64-bit MSP adder achieves a 23% reduction in area, 53% reduction in power with a slight 4% increase in delay than the 64-bit MSP adder. Next, the proposed dynamically configurable CSMT based 64-bit MSP adder consumes less power by 59.2% and 13% compared to the 64-bit MSP adder and the proposed CSMT based 64-bit MSP adder, respectively. The proposed dynamically configurable CSMT based 64-bit MSP adder requires less area by 15.7% than the 64-bit MSP adder. All three designs have close critical path delays; however, the low-area and low-power feature of the proposed dynamically configurable CSMT based 64-bit MSP adder makes it practical for media signal processing applications.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Gupta, V., Mohapatra, D., Raghunathan, A. and Roy, K. (2013) Low-Power Digital Signal Processing Using Approximate Adders. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **32**, 124-137. https://doi.org/10.1109/TCAD.2012.2217962

[2] Pashaeifar, M., Kamal, M., Afzali-Kusha, A. and Pedram, M. (2018) Approximate Reverse Carry Propagate Adder for Energy-Efficient DSP Applications. *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, **26**, 2530-2541. https://doi.org/10.1109/TVLSI.2018.2859939

[3] Kiran, K.R., Kumar, C.A. and Suresh Kumar, M. (2015) Design and Analysis of a Novel High-Speed Adder Based Hardware Efficient Discrete Cosine Transform (DCT). 2015 5*th International Conference on Advances in Computing and Communications* (*ICACC*), Kochi, 2-4 September 2015, 169-173. https://doi.org/10.1109/ICACC.2015.88

[4] Yang, H.Q., Zhang, F.J., Lai, J.M. and Wang, Y. (2010) Image Filtering Using Partially and Dynamically Reconfiguration. 2010 10*th IEEE International Conference on Solid-State and Integrated Circuit Technology*, Shanghai, 1-4 November 2010,

2067-2073. https://doi.org/10.1109/ICSICT.2010.5667280

[5] Bhandari, S., Subbaraman, S., Pujari, S., Cancare, F., Bruschi, F., Santambrogio, M.D. and Grassi, P.R. (2012) High-Speed Dynamic Partial Reconfiguration for Real-Time Multimedia Signal Processing. *Proceedings of the* 2012 15*th Euromicro Conference on Digital System Design*, 5-8 September 2012, 319-326. https://doi.org/10.1109/DSD.2012.74

[6] Vipin, K. and Fahmy, S.A. (2018) FPGA Dynamic and Partial Reconfiguration: A Survey of Architectures, Methods, and Applications. *ACM Computing Surveys*, **51**, Article No. 72, 39 p. https://doi.org/10.1145/3193827

[7] Hubner, M., Tradowsky, C., Gohringer, D., Braun, L., Thoma, F., Henkel, J. and Becker, J. (2011) Dynamic Processor Reconfiguration. *Proceedings of the* 2011 *International Conference on Reconfigurable Computing and FPGAs*, Cancun, 30 November-2 December 2011, 123-128. https://doi.org/10.1109/ReConFig.2011.30

[8] Kojima, T. and Amano, H. (2018) A Configuration Data Multicasting Method for Coarse-Grained Reconfigurable Architectures. 2018 28*th International Conference on Field Programmable Logic and Applications*, Dublin, 27-31 August 2018, 239-2393. https://doi.org/10.1109/FPL.2018.00048

[9] Chetan Kumar, V., Sai Phaneendra, P., Ahmed, S.E., Veeramachaneni, S., Muthukrishnan, N.M. and Srinivas, M.B. (2011) A Prefix Based Reconfigurable Adder. 2011 *IEEE Computer Society Annual Symposium on VLSI*, Chennai, 4-6 July 2011, 349-350. https://doi.org/10.1109/ISVLSI.2011.69

[10] Perri, S., Corsonello, P. and Cocorullo, G. (2002) 64-Bit Reconfigurable Adder for Low Power Media Processing. *Electronics Letters*, **38**, 397-399. https://doi.org/10.1049/el:20020295

[11] Stefania, P., Pasquale, C. and Giuseppe, C. (2003) A High-Speed Energy Efficient 64-Bit Reconfigurable Binary Adder. *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, **11**, 939-943. https://doi.org/10.1109/TVLSI.2003.817109

[12] Kumar, G.K. and Balaji, N. (2017) Reconfigurable Delay Optimized Carry Select Adder. 2017 *International Conference on Innovations in Electrical*, *Electronics*, *Instrumentation and Media Technology* (*ICEEIMT*), Coimbatore, 3-4 February 2017, 123-127. https://doi.org/10.1109/ICIEEIMT.2017.8116819

[13] Karthick, S., Valarmathy, S. and Prabhu, E. (2014) Reconfigurable Adder Architectures for Low Power Applications. *International Journal of Computer Applications*, **96**, 31-36. https://doi.org/10.5120/16783-6367

[14] Vijay, K.C.N. and Pandu, S. (2013) Design and Implementation of CVNS Based Low Power 64-Bit Adder.

[15] Mohanty, B.K. and Patel, S.K. (2014) Area-Delay-Power Efficient Carry-Select Adder. *IEEE Transactions on Circuits and Systems II*: *Express Briefs*, **61**, 418-422. https://doi.org/10.1109/TCSII.2014.2319695

[16] Parmar, S. and Singh, K.P. (2013) Design of High-Speed Hybrid Carry Select Adder. 2013 3*rd IEEE International Advance Computing Conference* (*IACC*), Ghaziabad, 22-23 February 2013, 1656-1663. https://doi.org/10.1109/IAdCC.2013.6514477

[17] Benara, V. and Purini, S. (2016) Accurus: A Fast Convergence Technique for Accuracy Configurable Approximate Adder Circuits. 2016 *IEEE Computer Society Annual Symposium on VLSI* (*ISVLSI*), Pittsburgh, 11-13 July 2016, 577-582. https://doi.org/10.1109/ISVLSI.2016.58

[18] Abhilash, R., Dubey, S. and Chinnaiah, M. (2016) ASIC Design of Signed and Unsigned Multipliers Using Compressors. 2016 *International Conference on Microelectronics*, *Computing and Communications* (*MicroCom*), Durgapur, Durgapur,

1-6. https://doi.org/10.1109/MicroCom.2016.7522523

[19] Ahmed, S.E., Veeramanchaneni, S., Muthukrishnan, N.M. and Srinivas, M.B. (2011) Reconfigurable Adders for Binary/BCD Addition/Subtraction. 2011 *Asia Pacific Conference on Postgraduate Research in Microelectronics & Electronics*, Macau, 6-7 October 2011, 106-109. https://doi.org/10.1109/PrimeAsia.2011.6075082

[20] Parhi, K.K. (1999) Low-Energy CSMT Carry Generators and Binary Adders. *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, **7**, 450-462. https://doi.org/10.1109/92.805752

[21] Parhi, K.K. (2013) Comments on "Low-Energy CSMT Carry Generators and Binary Adders". *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, **21**, 791. https://doi.org/10.1109/TVLSI.2012.2190771