Journal of Engineering and Applied Science

RESEARCH

Open Access

Design of canonical signed digit multiplier using spurious power suppression technique adder

Jayalakshmi K. P.[1*], Priya Seema Miranda[1] and K. Aarya Shri[1]

*Correspondence: lakshmi.rag08@gmail.com

[1] Department of Electronics and Communication Engineering, St Joseph Engineering College, Mangaluru, India

Abstract

Reducing power consumption is a major challenge in developing integrated processors for smart portable devices. This is particularly important for extending battery life and ensuring extended usage of the device. However, some DSP processing applications involve complex algorithms that consume more power, which poses a significant challenge in designing DSP applications for VLSI circuits. To address this issue, low-power consumption methodologies are required. Although various strategies have been developed to reduce power consumption, they have not demonstrated a significant decrease in dynamic power consumption, which is the primary factor determining the total amount of power dissipation.

The focus of this research is to develop a low-power multiplier using the spurious power suppression technique (SPST), a method that divides the arithmetic unit into the most significant part (MSP) and least significant part (LSP) and turns off the MSP when it is not required for computation. This approach reduces dynamic power and overall power consumption of the VLSI combinational circuit. The proposed system also utilizes canonical signed digit (CSD) representation to further reduce power usage.

The system was designed using Cadence design suite, and the results showed a significant reduction of 35.8% in power consumption for a 32-bit SPST-enabled CSD multiplier. The proposed system's total power consumption is 0.561 mW. Additionally, the proposed system was used in a power and area-efficient 256-point FFT architecture, resulting in an 86.6% reduction in power consumption. This system is suitable for real-time applications such as systems that use orthogonal frequency division multiplexing.

Keywords: SPST, CSD, FFT, Canonical signed digit, Most significant part, Low power

Introduction

Multiplication is a fundamental arithmetic operation and plays a significant role in digital signal processing (DSP) applications. In the multiplier, partial products obtained through the multiplication process are added using adders. However, as the number of partial products increases, more adders are required, resulting in increased power consumption. Thus, with technological advancements, the need for low-power multipliers has arisen [10]. To achieve this, low-power techniques are employed to reduce the number of partial products effectively while also decreasing power consumption. Most low-power techniques aim to reduce dynamic power usage by dividing the arithmetic unit into two sections, most significant part (MSP) and least significant part (LSP), and turning off the noneffective component during partially guarded computation (PGC) to save power. Experimental findings have shown that the PGC approach reduces power consumption by 10–44% in array multipliers. Another low-power design approach involves checking and carrying out the functional unit addition operation, scaling the resultant total to equal the initial length. The simulation's findings demonstrate that low-power adders perform computations more effectively than traditional adders. A low-power multiplier design that reduces the switching activity of partial products by using the booth algorithm (Radix 4) results in low-power consumption with increased delay and area. Glitch power can be reduced by changing current gates to ones that have a control input, shortening the settling time to reduce minimum power downtime after reactivation. The proposed system employs two techniques: canonical signed digit and spurious power suppression technique (SPST). Canonical signed digit reduces the number of nonzero digits in a number, thereby reducing the number of partial products. SPST divides the two N-bit binary number into MSP and LSP, computing MSP results only when the computation's results are affected, lowering the dynamic power, and reducing the overall power consumption of the VLSI combinational circuit [1].

Related work

A novel technique selectively activates components within functional units based on the input data, effectively reducing unnecessary power consumption. The findings and experimental results highlight the potential of partially guarded computation in achieving significant power savings while maintaining system performance. This work serves as an important reference for researchers and practitioners in the field of low-power electronics design, guiding the development of energy-efficient solutions for modern electronic systems. In their work, Choi, Jeon, and Choi focused on power minimization of functional units through partially guarded computation. They presented their findings at the IEEE International Symposium on Low Power Electronics Design in 2000. The authors introduced a technique that aimed to reduce power consumption by selectively activating functional units only when necessary. By employing this approach, they achieved significant power savings in the design of functional units. The paper provides detailed insights into the methodology and experimental results obtained, showcasing the effectiveness of their proposed technique [2]. Chen, Sheen, and Wang presented a low-power adder design that operates on effective dynamic data ranges. Their work, published in the IEEE Transactions on Very Large Scale Integration (VLSI) Systems in 2002, aimed to reduce power consumption in adder circuits. The authors introduced an innovative approach that utilizes dynamic data ranges to optimize the adder's power efficiency. By adjusting the data range dynamically based on the input operands, the proposed adder achieved significant power savings while maintaining accuracy. The paper provides a comprehensive analysis of the design methodology, experimental results, and comparative evaluations with other adder architectures [3]. In the study titled "Minimization of switching activities of partial products for designing low-power multipliers," the authors developed a low-power design technique that focuses on minimizing the switching activities of partial products. They examined the switching activities of several partial product generation systems and identified the factors that influenced power usage. The proposed method is a modified partial product generation technique that lowers switching activities [4]. The authors of the study “Glitch power minimization by selective gate freezing” developed a method to reduce glitch power consumption in digital circuits by selectively freezing specific gates. Glitches are short-lived, undesired voltage changes that occur in combinational circuits when logic gates are switched. These glitches can waste a large amount of power, increasing the circuit and overall power usage. The proposed algorithms identify the gates that should be frozen based on their glitch-generating behavior [5]. In their work, Benini et al. proposed a technique called selective gate freezing for glitch power minimization. Published in the IEEE Transactions on Very Large Scale Integration (VLSI) Systems in 2000, the authors addressed the issue of power consumption in digital circuits due to glitches. They introduced a novel approach that selectively freezes certain gates in the circuit during inactive periods, reducing dynamic power dissipation. The paper presents a detailed analysis of the methodology, experimental results, and comparisons with existing glitch power reduction techniques. The findings highlight the effectiveness of selective gate freezing in achieving significant power savings while maintaining circuit functionality [6]. Henzler et al. presented a fast power-efficient circuit-block switch-off scheme in their publication in Electronics Letters in 2004. The authors addressed the power consumption issue in integrated circuits by proposing a technique that selectively switches off circuit blocks during periods of inactivity. By employing a fast switch-off mechanism, the proposed scheme achieved power savings by reducing leakage and dynamic power dissipation. The paper provides a concise explanation of the technique along with experimental results demonstrating its effectiveness in reducing power consumption. Comparative evaluations with other power reduction schemes further emphasize the advantages of the proposed circuit-block switch-off scheme [7]. Lakshmi et al. proposed a design technique for low-power multipliers by employing the spurious power suppression technique (SPST). The authors addressed the challenge of power consumption in multiplier circuits and introduced an innovative approach to mitigate spurious power dissipation. The SPST method effectively suppressed power consumption by identifying and reducing unnecessary power dissipation caused by various sources, such as glitching and leakage currents. The paper presents a comprehensive explanation of the design methodology, experimental results, and comparative evaluations with conventional multiplier architectures. The findings highlight the effectiveness of the SPST technique in achieving substantial power savings while maintaining the desired functionality of the multiplier circuit [1].

Proposed method

Figure 1 shows the block diagram of an SPST-enabled CSD multiplier. The inputs, A and B, are both 16-bit numbers. The partial product candidate generator block takes in input A and generates three partial product candidates, with values of − A, 0, and A, each having 32 bits. The CSD recoder output provides magnitude and sign data, which are used to select one of these partial products. The 16-bit input B is recoded by the

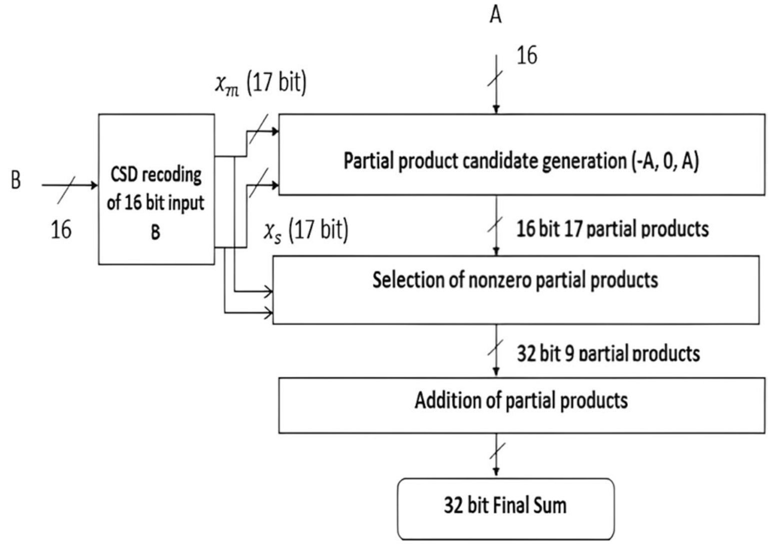

A
16
x_m (17 bit)
B
16
CSD recoding of 16 bit input B
Partial product candidate generation (-A, 0, A)
x_s (17 bit)
16 bit 17 partial products
Selection of nonzero partial products
32 bit 9 partial products
Addition of partial products
32 bit Final Sum

Fig. 1 Canonical signed digit multiplier block diagram with spurious power suppression enabled

CSD recoding block to produce 17-bit magnitude and sign values. Depending on these values, 17 partial products are created, and one is chosen for further processing. The selection of the partial product is based on the nonzero magnitude values from the CSD block. Seventeen partial products are then generated, of which only nine are chosen. The chosen partial products, except for the ninth one, are provided to the SPST adder, with two of them being supplied back to back. The shifting block left shifts the remaining 16 partial products. The conventional adder receives the outputs from the four SPST adders and provides the final result.

CSD recoding block

The canonical signed digit is one type of number representation used for arith metic operations. It is also known as a recoding technique since it will recode the original number to create a new one with a minimum amount of nonzero digits. This method guarantees that the average number of nonzero digits will never be greater than n/2. The canonical signed digit (CSD) representation is distinct because consecutive nonzero digits are one of the key properties of the CSD [8].

The CSD recoding circuit is constructed to take benefit of this property by converting three input bits into a single CSD digit, as shown in Fig. 2. The converter recodes three binary digits, i.e., b_{i+1}, b_i, and b_{i-1}, into a single CSD digit x_i which is represented in terms of magnitude bit $x_{i,m}$ and sign bit $x_{i,s}$. In the sign-magnitude encoding, 0.1 and -1 are represented as 00, 01, and 11, respectively. Additionally, two bypass signals are employed, namely input bypass signal p_i and output bypass signal p_{i+1}. The truth table for binary to CSD conversion is shown in Table 1. It can be seen that when $p_i = 0$, a single CSD digit x_i and the new bypass signal for the following procedure are formed from the three binary digits b_{i+1}, b_i, and b_{i-1}. The magnitude bit $x_{i,m}$ has the same value as the output bypass signal p_{i+1}. The magnitude bit is determined by the value of b_i and b_{i-1}, while the sign bit is impacted by b_{i+1} and $x_{i,m}$. Regardless of the inputs, all outputs

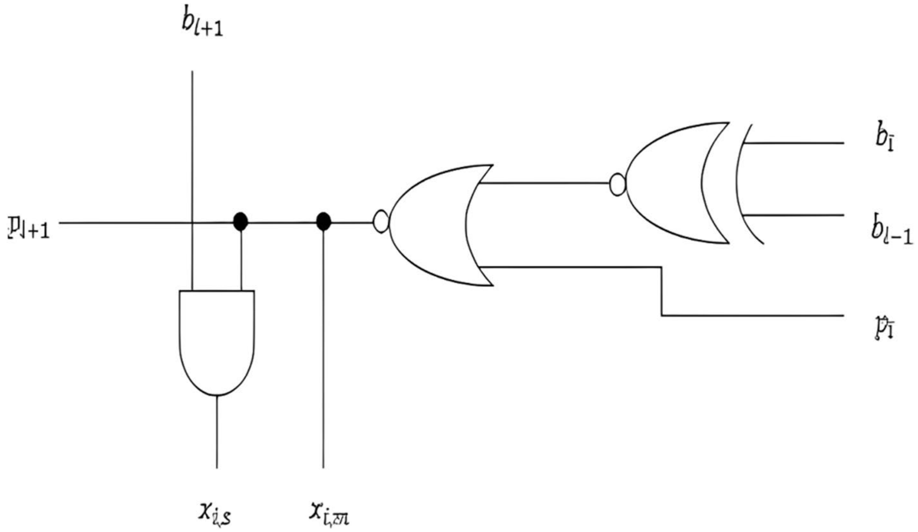


Fig. 2 Binary to CSD conversion block

Table 1 Truth table showing binary to CSD conversion

p_i	*bi* + 1	b_i	*bi* − 1	x_i	*xi,s*	*xi,m*	*pi* + 1
0	0	0	0	0	0	0	0
	0	0	1	1	0	1	1
	0	1	0	1	0	1	1
	0	1	1	0	0	0	0
	1	0	0	0	0	0	0
	1	0	1	− 1	1	1	1
	1	1	0	− 1	1	1	1
	1	1	1	0	0	0	0
1	d	d	d	0	0	0	0

become zeros when $p_\mathrm{i} = 1$. Therefore, in this instance, the converter's inputs are ignored or bypassed, generating the next process's bypass signal $p_{\mathrm{i}+1}$ with the value zero [9].

The 16-bit CSD recoding block is shown in Fig. 3. The 17-bit CSD representation of the input is generated by this circuit in the form of 17-bit magnitude values and 17-bit sign values. The single CSD digit binary to CSD conversion circuit used by the recoding block produces the sign bit and magnitude bit.

SPST adder

Figure 4 shows the cause for the spurious signal transitions. It is clear from the first and second cases that adding the two operands has no effect on the MSP outcome, whether or not there is a carry from LSP. Results for MSP can be expected from the sum achieved in both circumstances.

Eliminating the computation of MSP of operands can lead to a reduction in switching activities in related components, which subsequently lowers power consumption in the adder stage and minimizes glitching noises. In this study, an SPST adder has been developed that divides the adder into two sections and freezes the MSP input data if it does

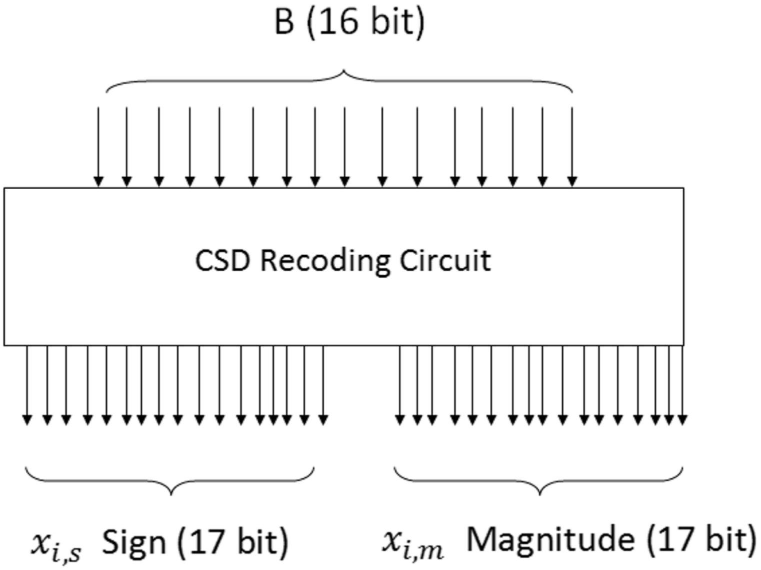


Fig. 3 16-bit CSD recoding circuit

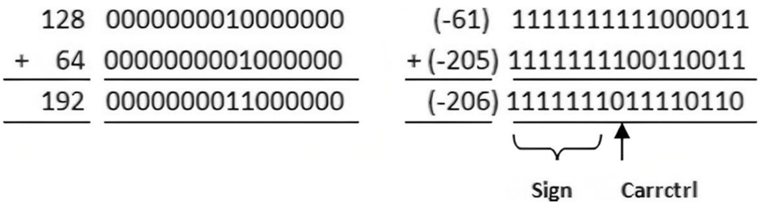


Fig. 4 Examples showing the spurious transitions

not affect the final sum. A detection logic circuit has been devised to identify the effective range of input and determine whether MSP results are influencing the calculation outcome. The Boolean expression used to design this logic circuit is displayed below:

$$A_{MSP} = A[31:16]B_{MSP} = B[31:16] \tag{1}$$

$$A_{and} = A[31]A[30]\ldots A[16] \tag{2}$$

$$B_{and} = B[31]B[30]\ldots B[16] \tag{3}$$

$$A_{NOR} = A\overline{[31] + A[30] + +A[16]} \tag{4}$$

$$B_{NOR} = \overline{B[31] + B[30] + +B[16]} \tag{5}$$

$$Close = \overline{(A_{and} + A_{nor})(B_{and} + B_{nor})} \tag{6}$$

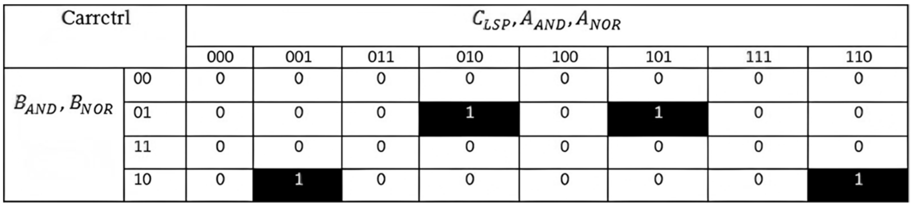
Where A[m] is the m^{th} bit of the operand A and B[n] is the n^{th} bit of the operand B. A_{MSP} and B_{MSP} are the MSP part of the inputs A and B. When all of the bits in A_{MSP} and B_{MSP} are zeros,

A_{and} and B_{and} have zero values. The value of A_{nor} and B_{nor} is zero when all of the bits in A_{MSP} and B_{MSP} are ones. The detection logic unit will produce three output signals: Close, Carrctrl, and Sign. The MSP section will either be disabled or not, depending on the Close value. The MSP component is disabled to reduce power consumption if the Close is zero. This reduces the switching operations in the MSP section, resulting in zero dynamic power usage. The zero inputs are then sent to the MSP part. The MSP result obtained will be computed in the detection logic unit, and MSP bits are compensated by the Sign and Carrctrl signals.

The Boolean expression for the Sign and Carrctrl signals is obtained from the Karnaugh map as shown in Figs. 5 and 6. Using the eight possible combinations of the inputs A and B, the Sign, Carrctrl, Close, A_{and}, B_{and}, A_{nor}, and B_{nor} are generated which is shown in Table 2.

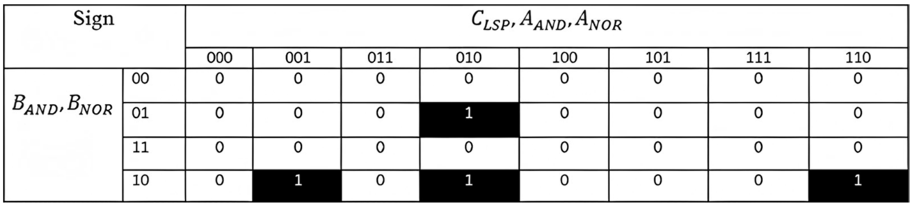
The expression of Carrctrl and Sign are derived from the Karnaugh map is given in Eqs. 7 and 8.

$$\begin{aligned}Carrctrl = {} & \left(\overline{C_{LSP}}.\overline{A_{AND}}.A_{NOR}.B_{AND}.\overline{B_{NOR}}\right)+\\ & \left(\overline{C_{LSP}}.A_{AND}.\overline{A_{NOR}}.\overline{B_{AND}}.B_{NOR}\right)+\\ & \left(C_{LSP}.\overline{A_{AND}}.A_{NOR}.\overline{B_{AND}}.B_{NOR}\right)+\\ & \left(C_{LSP}.A_{AND}.\overline{A_{NOR}}.B_{AND}.\overline{B_{NOR}}\right)\end{aligned} \tag{7}$$



Carrctrl		$C_{LSP}, A_{AND}, A_{NOR}$							
		000	001	011	010	100	101	111	110
B_{AND}, B_{NOR}	00	0	0	0	0	0	0	0	0
	01	0	0	0	1	0	1	0	0
	11	0	0	0	0	0	0	0	0
	10	0	1	0	0	0	0	0	1

Fig. 5 Karnaugh map of Carrctrl expression



Sign		$C_{LSP}, A_{AND}, A_{NOR}$							
		000	001	011	010	100	101	111	110
B_{AND}, B_{NOR}	00	0	0	0	0	0	0	0	0
	01	0	0	0	1	0	0	0	0
	11	0	0	0	0	0	0	0	0
	10	0	1	0	1	0	0	0	1

Fig. 6 Karnaugh map of Sign expression

Table 2 Sign, Carrctrl, and Close computation for eight combinations of inputs A and B

*A*MSP	*B*MSP	CLSP	Close	Carrctrl	Sign
0000000000000000	0000000000000000	0	0	0	0000000000000000
0000000000000000	0000000000000000	1	0	1	0000000000000000
0000000000000000	1111111111111111	0	0	1	1111111111111111
0000000000000000	1111111111111111	1	0	0	0000000000000000
1111111111111111	0000000000000000	0	0	1	1111111111111111
1111111111111111	0000000000000000	1	0	0	0000000000000000
1111111111111111	1111111111111111	0	0	0	1111111111111111
1111111111111111	1111111111111111	1	0	1	1111111111111111

$$\begin{aligned} sign = {} & C_{LSP}.(A_{AND}.A_{NOR}.B_{AND}.B_{NOR}+ \\ & A_{AND}.\overline{A_{NOR}}.\overline{B_{AND}}.B_{NOR}+ \\ & A_{AND}.\overline{A_{NOR}}B_{AND}.\overline{B_{NOR}})+ \\ & C_{LSP}.A_{AND}.\overline{A_{NOR}}.B_{AND}\overline{B_{NOR}} \end{aligned} \tag{8}$$

The detection logic circuit is shown in Fig. 7, and Fig. 8 shows the 32-bit SPST adder design. The two 32-bit inputs A and B in this design are split into the MSP and LSP.

The LSP adder computes the LSP independently. Latches are employed in the MSP component to control the inputs to the MSP adder designed with AND gates. If an MSP computation is required, the latches allow two MSP inputs to enter the adder; otherwise, they freeze the MSP inputs and permit the MSP adder to receive zero inputs. Moreover, the detection logic circuit receives these MSP inputs and employs them to determine whether to activate or deactivate the MSP.

The detected logic circuit enables the latches to provide MSP inputs to the MSP adder only if MSP computation is needed. On the other hand, if MSP computation is not needed, the detection logic circuit will disable the latch and provide zero inputs to the MSP adder. The resulting MSP sum will then be compensated by the sign extension circuit. The sign extension circuit receives three signals from the detecting logic circuit as inputs.

Partial product candidate generation based on CSD magnitude and sign values

The block diagram of this architecture features a partial product generator that takes in a 16-bit input A and generates three possible partial products: A, 0, and – A. The sign and magnitude values produced by the CSD recoder block are utilized to select which of these 16-bit partial products to use. Additionally, a second 16-bit input (B) is fed into this block, which generates a 17-bit recoded output represented in terms of 17-bit magnitude and sign values. The output from the partial product generator block is 17, 16-bit length partial products, as depicted in Fig. 9.

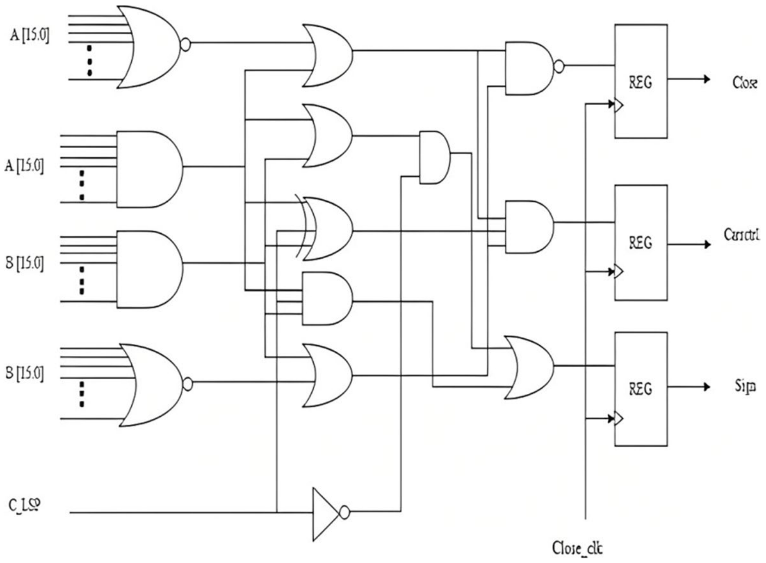


Fig. 7 Detection logic circuit design

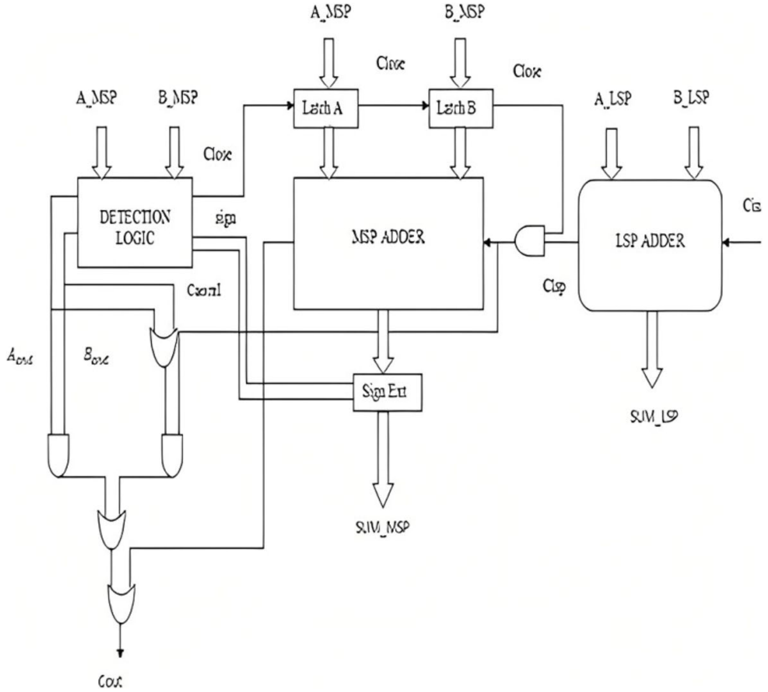


Fig. 8 SPST adder

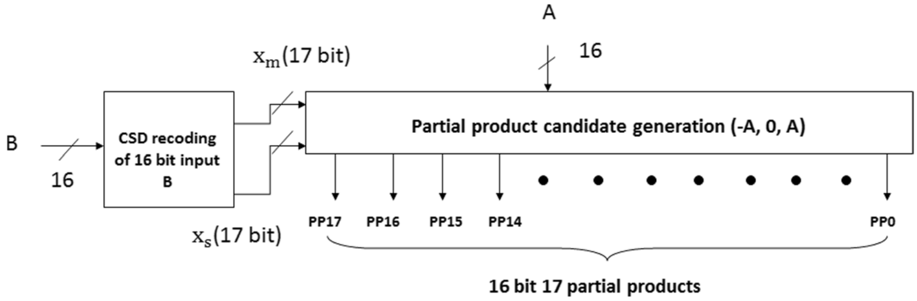

A
16
x_m(17 bit)
B
16
CSD recoding of 16 bit input B
Partial product candidate generation (-A, 0, A)
x_s(17 bit)
PP17
PP16
PP15
PP14
PP0
16 bit 17 partial products

Fig. 9 Partial product candidate generation

Selection of partial products block

In the previous design, 17 partial products of 16 bits each were generated and passed to the sign extension and shifting block. In this block, each partial product is extended to 32 bits using the sign extension method, which involves making the theoretical calculation of the multiplication process 32 bits long regardless of the actual length of the partial product obtained. During the multiplication process, all partial products except the first one are shifted by 1 bit. After sign extension, shifting operation is performed. The resulting partial products are then passed to the selection of partial products block, where only nine partial products out of the 17 are selected. This is because the CSD recoded output will have only n/2 nonzero values. Therefore, only those partial products whose sign values match the recoder output's 01 and 11 are selected, as shown in Fig. 10.

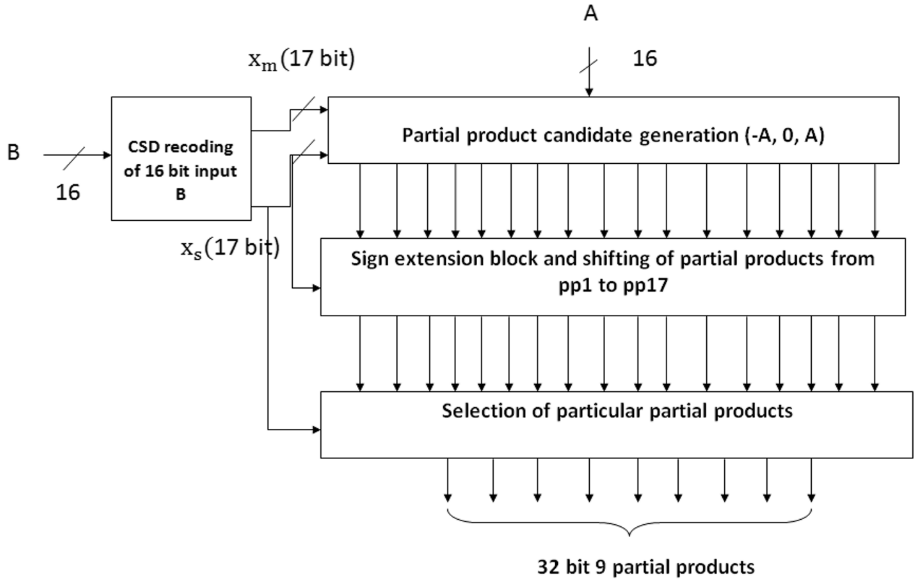

A
16
x_m(17 bit)
B
16
CSD recoding of 16 bit input B
x_s(17 bit)
Partial product candidate generation (-A, 0, A)
Sign extension block and shifting of partial products from pp1 to pp17
Selection of particular partial products
32 bit 9 partial products

Fig. 10 Selection of partial products

Adding of partial products

After extracting the nine selected partial products, all except for the ninth one are fed into the SPST adder in pairs of two. The four resulting totals from the SPST adders are then passed into the two conventional adders, as illustrated in Fig. 11. To obtain the final result, the sum from the conventional adders is added to the ninth partial product using a traditional adder.

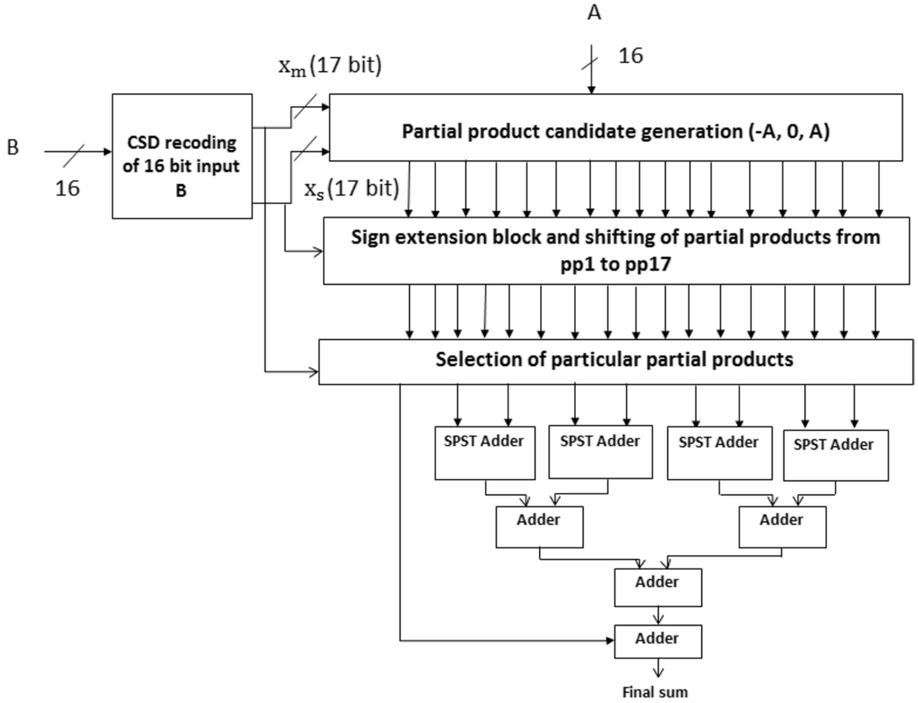

A
16
x_m(17 bit)
B
16
CSD recoding of 16 bit input B
x_s(17 bit)
Partial product candidate generation (-A, 0, A)
Sign extension block and shifting of partial products from pp1 to pp17
Selection of particular partial products
SPST Adder
SPST Adder
SPST Adder
SPST Adder
Adder
Adder
Adder
Adder
Final sum

Fig. 11 Adding of partial products block

Power and area-efficient 256 FFT architecture

The fast Fourier transform (FFT) technique is a popular DSP method used to convert signals from the time domain to the frequency domain and vice versa. However, FFT involves a large number of multiplication operations, which can consume significant power. To address this issue, a power and area-efficient architecture for a 256-point FFT can be developed by employing the SPST-enabled CSD multiplier technique, as illustrated in Fig. 12 [10]. In this architecture, the complex multiplier block utilizes an SPST-enabled CSD multiplier to perform the multiplication of the twiddle factors.

Results and discussion

The Verilog code for the SPST-based CSD multiplier was implemented using Cadence with 90-nm technology. Figures 13 and 14 depict the RTL design and the output waveform of the SPST adder, respectively. It can be observed from the waveform that the MSP computation is not performed during the negative cycle, even though the LSP output is present. Instead, MSP operations are carried out on the positive edge of the clock. As a result, the final sum is determined during the positive edge of the clock.

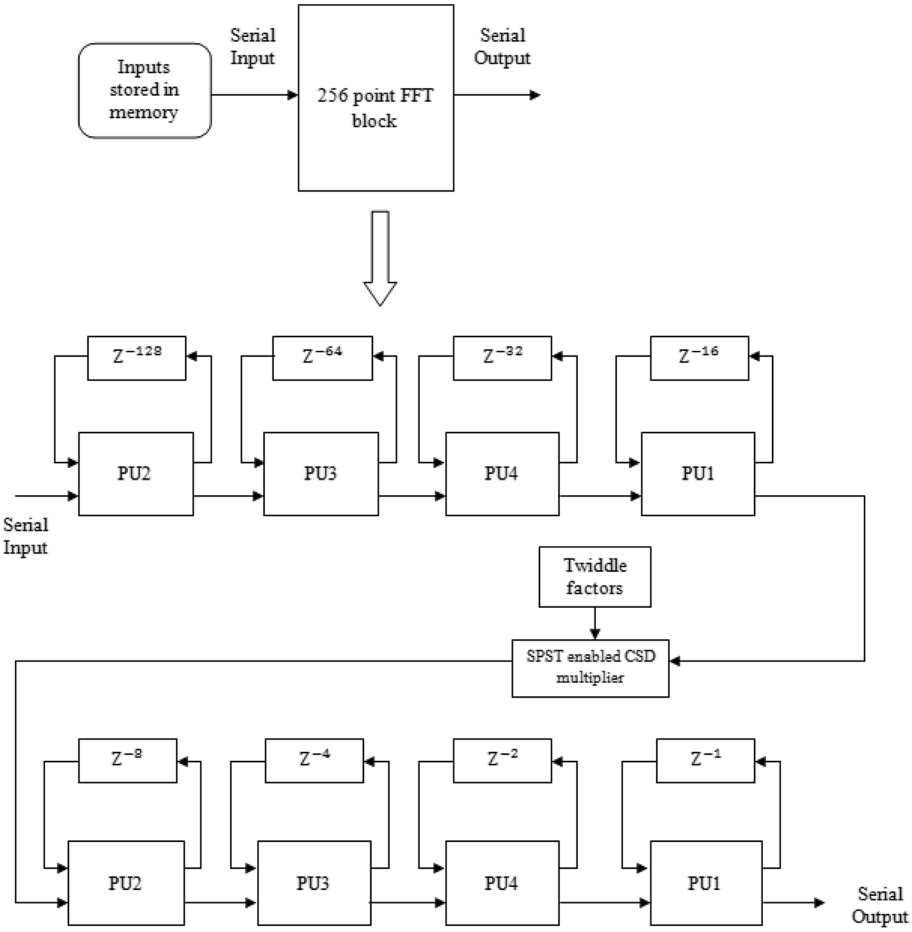


Fig. 12 FFT architecture — 256 points

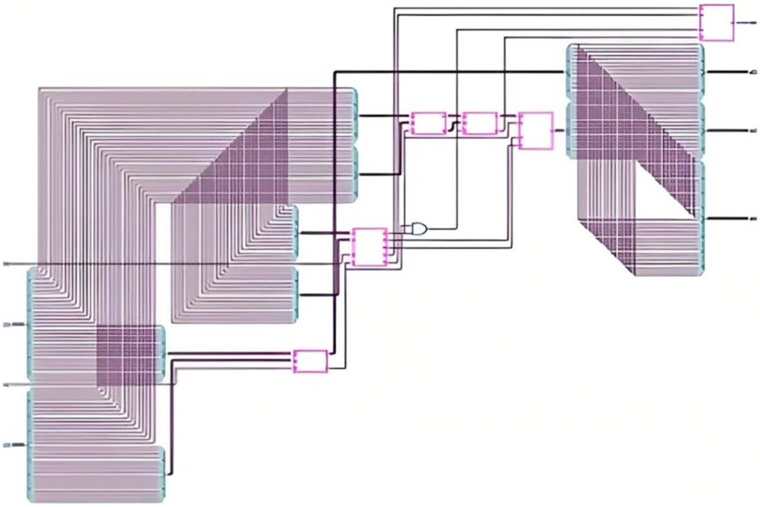


Fig. 13 SPST adder-RTL schematic

Table 3 presents the performance metrics of the SPST adder. On the other hand, Table 4 illustrates the dynamic power consumption of the MSP portion of the SPST adder and ripple carry adder. Notably, Table 4 reveals a remarkable reduction in dynamic power consumption for the MSP adder. This decrease is attributed to the exclusion of two ineffective input computations that the MSP adder does not add. Instead, the detecting logic unit of the SPST adder compensates for any obtained sum outcomes. As a result, unnecessary switching activity in the MSP is reduced, resulting in lower dynamic power consumption.

Figure 15 shows the output waveform for the signed multiplication for all signed input combinations. The performance parameters of the SPST-based CSD multiplier are shown in Table 5. A total of 256-point FFT architecture is implemented using SPST-enabled CSD multiplier. The FFT architecture is also implemented using Baugh Wooley-multiplier. The results obtained were compared, and it has been observed that SPST-enabled CSD multiplier consumes less power and area compared to Baugh-Wooley multiplier. The results are shown in Table 6.

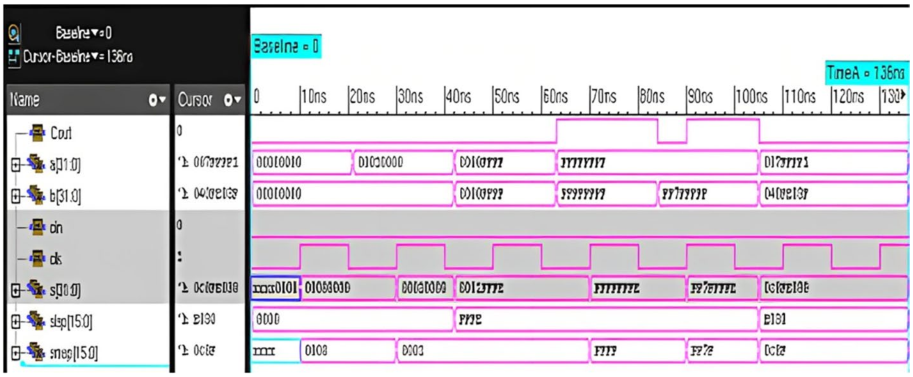


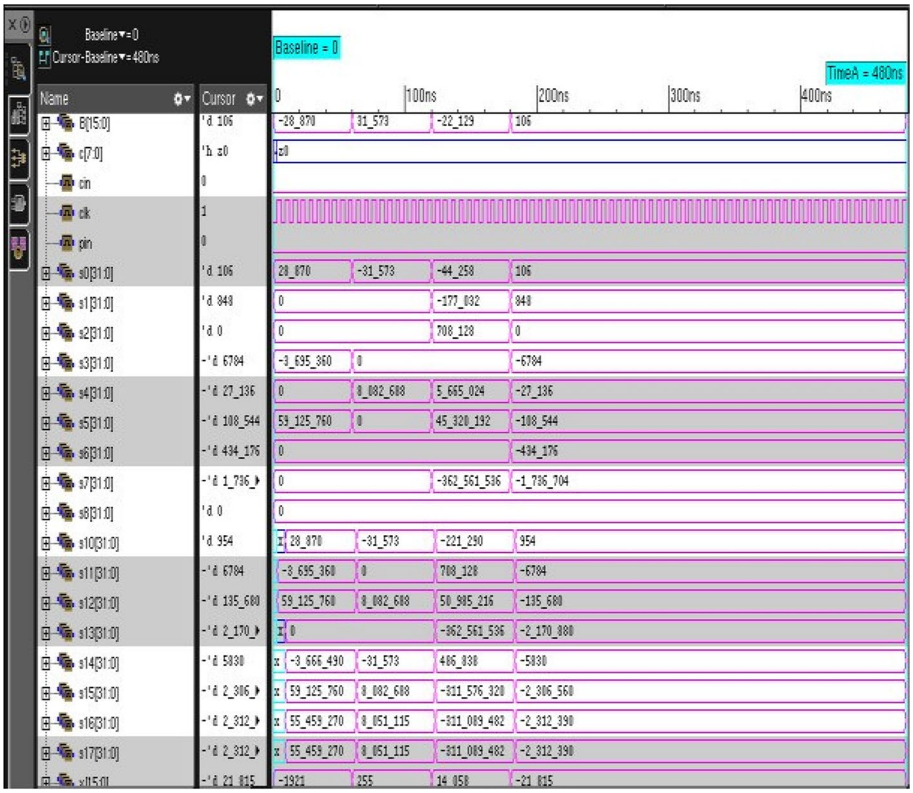
Fig. 14 Output waveform of SPST adder

Table 3 Performance parameters of the SPST adder

Performance parameters	SPST adder
Area (μm^2)	2624
Power (nW)	20,033.024
Delay (ps)	6867

Table 4 Comparison of SPST adder and ripple carry adder in terms of dynamic power

Dynamic power results			
	Total dynamic power (nW)	**Dynamic power of LSP section of the adder (nW)**	**Dynamic power of MSP section of the adder**
SPST adder	16,731.97	10,796.795	700.321
Carry ripple adder	27,215.149	10,796.795	13,495.817



Name	Cursor				
B[15:0]	'd 106	-28_870	31_573	-22_129	106
c[7:0]	'h z0	z0			
cin	0				
clk	1				
pin	0				
s0[31:0]	'd 106	28_870	-31_573	-44_258	106
s1[31:0]	'd 848	0		-177_032	848
s2[31:0]	'd 0	0		708_128	0
s3[31:0]	-'d 6784	-3_695_360	0		-6784
s4[31:0]	-'d 27_136	0	8_082_688	5_665_024	-27_136
s5[31:0]	-'d 108_544	59_125_760	0	45_320_192	-108_544
s6[31:0]	-'d 434_176	0			-434_176
s7[31:0]	-'d 1_736_▸	0		-362_561_536	-1_736_704
s8[31:0]	'd 0	0			
s10[31:0]	'd 954	I 28_870	-31_573	-221_290	954
s11[31:0]	-'d 6784	-3_695_360	0	708_128	-6784
s12[31:0]	-'d 135_680	59_125_760	8_082_688	50_985_216	-135_680
s13[31:0]	-'d 2_170_▸	I 0		-362_561_536	-2_170_880
s14[31:0]	-'d 5830	x -3_666_490	-31_573	486_838	-5830
s15[31:0]	-'d 2_306_▸	x 59_125_760	8_082_688	-311_576_320	-2_306_560
s16[31:0]	-'d 2_312_▸	x 55_459_270	8_051_115	-311_089_482	-2_312_390
s17[31:0]	-'d 2_312_▸	x 55_459_270	8_051_115	-311_089_482	-2_312_390
x[15:0]	-'d 21_815	-1921	255	14_058	-21_815

Fig. 15 Output waveform for signed multiplication

Table 5 Performance parameters of the SPST-based CSD multiplier

Performance parameters	SPST enables CSD multiplier
Area (μm^2)	10,879
Power (nW)	561,606.766
Delay (ps)	12,277

Table 6 Comparison of power and area of modified system for signed multiplication

Comparison of power and area of signed multiplier results			
Power and area-efficient 256 FFT archi tecture	**Using Baugh-Wooley multiplier**	**Using SPST-enabled CSD multiplier**	**Percentage reduction in number of cells, area, and power**
Numbers of cells	51,990	10,994	78.8%
Power (mW)	23.228	3.1	86.6%
Area (μm^2)	367,019	70,070	80.9%

Conclusions

The canonical signed digit is a number representation commonly used in arithmetic operations, where the original number is recoded to produce a new one with the least possible number of nonzero digits. This method has the advantage of ensuring that the average number of nonzero digits is always less than or equal to n/2. The SPST adder incorporates both LSP and MSP adders in its design, with the MSP component turned off depending on the input's dynamic range. This feature contributes to a reduction in dynamic power dissipation.

After deployment, the SPST adder demonstrated a significant reduction in dynamic power consumption. For an input combination consisting of 50% data and a dynamic range of 16 bits out of 32 bits, the power dissipation decreased by 38.5% compared to the carry ripple adder (with MSP adder enabled for 50% of input combinations). The proposed multiplier consumed 0.561 mW of power. Furthermore, the proposed system was applied to the power and area-efficient 256-point FFT architecture, leading to an 86.6% reduction in overall power consumption compared to the same application using the Baugh-Wooley multiplier.

Abbreviations

SPST	Spurious power suppression technique
CSD	Canonical signed digit
FFT	Fast Fourier transform
MSP	Most significant part

Acknowledgements
First and foremost, I would like to thank my college who give me a comfortable working and researching atmosphere. I am also greatly indebted to all my teams who have helped me to develop the fundamental and essential academic competence. My sincere appreciation also goes to all my colleagues, who are my proud of my life. Last but not least, I want to thank all my friends, for their encouragement and support.

Authors' contributions

The manuscript was written through contributions of all authors. And all authors have read and approved the manuscript, which is the case.

Funding
Not applicable.

Availability of data and materials
Not applicable.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 4 March 2023 Accepted: 6 July 2023
Published online: 25 July 2023

References

1. Lakshmi A, Kyung Tae Kim, Tanmayi M, Pavani K (2020) Design of low-power multiplier using spurious power suppression technique (SPST). J Crit Rev 7(4)
2. Choi J, Jeon J, Choi K (2000) Power minimization of functional units by partially guarded computation. In: Proc IEEE Int Symp Low Power Electron Des. pp 131–36
3. Chen O, Sheen R, Wang S (2002) A low-power adder operating on effective dynamic data ranges. IEEE Trans Very Large Scale Integr (VLSI) Syst 10(4):435–53
4. Chen O, Wang S, Wu YW (2003) Minimization of switching activities of partial products for designing low-power multipliers. IEEE Trans Very Large Scale Integr (VLSI) Syst 11(3):418–33
5. Benini L, Micheli GD, Macii A, Macii E, Poncino M, Scarsi R (2000) Glitch power minimization by selective gate freezing. IEEE Trans Very Large Scale Integr (VLSI) Syst 8(3):287–98
6. Henzler S, Georgakos G, Berthold J, SchmittLandsiedel D (2004) Fast power-efficient circuit-block switch-off scheme. Electron Lett 40(2):103–104
7. Chen KH, Chu YS, Member (2009) A spurious power suppression technique for multimedia/DSP applications. IEEE Trans Circuits Syst Reg Pap 56(1):132
8. Vishwanath BR, Theerthesha TS (2015) Multiplier using canonical signed digit code. Int J Res Appl Sci Eng Technol (IJRASET) 3(V)
9. Faust F, Gustafsson O, Chang CH (2011) Fast and VLSI efficient binary-to-CSD en- coder using bypass signal. Electron Lett 47(1):18
10. Sagar M, Sayed Saber Ali, Sharath, Shashidhara NJ, Sharon Thomas, Vijay Ganesh PC (2016) Power and area efficient 256 FFT architecture

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.