Advanced Research and Development Journal
ISSN: **3139-1281**| Volume 2 Issue 2 | April-June 2026 | ARD-046
www.ardjournal.com | DOI: https://doi.org/10.5281/zenodo.21257903

Design and Optimization of a Quaternary Signed-Digit Adder for VLSI Applications

[*1]**RAYANCHU PAVANI,** *M.Tech Student, Department of ECE,*
[*2]**D. RAVIKIRAN BABU,** *Associate Professor, Department of ECE,*
[*1,*2]*Jyothishmathi Institute of Technology and Science (AUTONOMOUS), Karimnagar, TG.*

ABSTRACT— The time required for addition is constrained by the carry propagation from the least significant bit to the most significant bit in the context of conventional binary adders. This lower bound increases in proportion to the operand's width. The quaternary signed-digit (QSD) number system, a redundant representation of radix-4 using {-3,...,3}, is used to overcome this constraint. Because each digit can be represented in several ways, addition can be done in a consistent amount of steps regardless of the number of digits. A carry-free QSD adder for VLSI applications is designed and optimized in this study. A compact three-bit two's complement digit encoding and two-step implementation are used in the adder. From each pair of digits, an intermediate carry and sum are generated. This prevents subsequent digit additions from generating a carry. All digit slices are reduced to tiny two-level networks, and the adder is a parallel array of identical slices. Due to generating logic enhancement, it has been decreased to this level. The design is compared to ripple-carry, carry-lookahead, carry-select, and redundant-binary adders in power, transistor count, and latency using a transistor-level approximation in a 45 nm CMOS flow. The normal data shown here show that worst-case addition latency is consistent from 8 to 64 digits. In contrast to binary adders' growing delay, the transistor's cost is linearly related to a ripple-carry adder. The numerical values provided are for illustration only; the reader should duplicate them using SPICE data and synthesis.

Index Terms—*Quaternary signed digit, carry-free addition, redundant number system, high-speed adder, VLSI arithmetic, low-power design.*

I. INTRODUCTION

A. Background and Motivation

Avizienis proposed redundancy to eliminate carry propagation. He used signed-digit representations to limit carry propagation to one place to make addition time independent of word length. Parhami created the generalized signed-digit framework to broaden these principles. This framework classifies redundant systems by radix and digit range and determines carry-free or limited-carry addition conditions. Koren, Parhami, and Hwang discuss computer arithmetic generally in their texts.

This family's quaternary signed-digit system has been studied for VLSI efficiency. However, Dakhole and Wakde built multi-digit QSD adders on programmable devices and demonstrated the two-step approach in hardware. In their carry-free QSD adder demonstration, Dubey and Rani found that the constant-delay characteristic, which is beneficial for operand lengths longer than a few tens of bits, becomes useful. Additionally, current mode and multi-valued circuit designs have used QSD and other signed-digit segments to increase speed and density. Canonical QSD arithmetic has been studied for

Advanced Research and Development Journal
ISSN: 3139-1281| Volume 2 Issue 2 | April-June 2026 | ARD-046
www.ardjournal.com | DOI: https://doi.org/10.5281/zenodo.21257903

optoelectronic symbolic substitution computer applications. To speed up multiplication, redundant signed-digit addition trees are often used with adders. Carry-free signed-digit processes are used in decimal arithmetic, and parity-based approaches improve signed-digit adder fault tolerance. This work adds an efficient single-digit slice shared at the encoding level and compares it to common binary adder families to this collection.

B. Contributions

This paper discusses carry-free QSD adder design and optimization. This adder is for VLSI implementation. We present a compact encoding of a two's-complement digit to 3 bits that allows simple two-level logic to generate the intermediate carry and sum, an optimized single-digit slice that shares and minimizes the generation and second-stage addition, a fully parallel array construction whose worst-case delay is independent of operand width, and a comparative evaluation against ripple-carry, carry-lookahead, and ca. The reader can use SPICE flows and synthesis to re-deduce the numbers in this article, which are representative values.

II. RELATED WORK

Avizienis proposed redundancy to eliminate carry propagation. He used signed-digit representations to limit carry propagation to one place to make addition time independent of word length. Parhami created the generalized signed-digit framework to broaden these principles. This framework classifies redundant systems by radix and digit range and determines carry-free or limited-carry addition conditions. Koren, Parhami, and Hwang discuss computer arithmetic generally in their texts.

This family's quaternary signed-digit system has been studied for VLSI efficiency. However, Dakhole and Wakde built multi-digit QSD adders on programmable devices and demonstrated the two-step approach in hardware. In their carry-free QSD adder demonstration, Dubey and Rani found that the constant-delay characteristic, which is beneficial for operand lengths longer than a few tens of bits, becomes useful. Additionally, current mode and multi-valued circuit designs have used QSD and other signed-digit segments to increase speed and density. Canonical QSD arithmetic has been studied for optoelectronic symbolic substitution computer applications. To speed up multiplication, redundant signed-digit addition trees are often used with adders. Carry-free signed-digit processes are used in decimal arithmetic, and parity-based approaches improve signed-digit adder fault tolerance. This work adds an efficient single-digit slice shared at the encoding level and compares it to common binary adder families to this collection.

III. THE QSD NUMBER SYSTEM

The quaternary signed-digit system uses a radix-4 representation with a seven-valued set of digits (D = { -3, -2, -1, 0, 1, 2, 3 }). Weights are being raised by four, one, two, etc. These digits form a weighted sum that represents a number. With seven-digit values for a four-digit radix, the approach is redundant. A single number can be represented by many digit sequences from different combinations. For example, the number of five can be expressed as (1, 1) (1.4 + 1) or (2, -3) (2.4 - 3). This independence allows the adder to avoid a carry chain.

Advanced Research and Development Journal
ISSN: 3139-1281| Volume 2 Issue 2 | April-June 2026 | ARD-046
www.ardjournal.com | DOI: https://doi.org/10.5281/zenodo.21257903

Assigning bits to each QSD digit simplifies hardware implementation. The three-bit two's complement is convenient because it covers all seven needed digit values with one duplicate code, -4 to 3. A four-bit two's complement word may hold the sum of two digits, which is an integer from -6 to 6. The carry and sum of the intermediate are in the sets {-1, -1, 1} and {-2, -1, -1, 1, 2}, respectively. All of these small marked fields map straight to ordinary cells. Each field is signed. Negating the digits you subtract from the total subtracts. Both positive and negative values are available. One adder structure can add and subtract.

IV. CARRY-FREE ADDITION ALGORITHM

The inclusion of QSD requires two steps. Initially, a digit sum like an i + b i is calculated by adding each pair of operand digits between -6 and 6. An intermediate carry c i+1 ∈ {−1, 0, 1} is allocated to this value and transmitted to the next higher position. Additionally, a local intermediate total w i ∈ {−2, −1, 0, 1, 2} is recorded. Both numbers are moderate. The values are chosen so that an i + b i = 4x c i+1 + w i. Table I shows the fixed mapping.

TABLE I
QSD INTERMEDIATE CARRY AND SUM GENERATION

Digit sum $a_i + b_i$	Carry c_{i+1}	Interm. sum w_i
−6	−1	−2
−5	−1	−1
−4	−1	0
−3	−1	1
−2	0	−2
−1	0	−1
0	0	0
1	0	1
2	0	2
3	1	−1
4	1	0
5	1	1
6	1	2

The most essential feature is that the intermediate total and inbound carry can only be -2 to 2 and -1, 0, or 1. This is because the sum of the second phase, $S_i = w_i + c_i$, is always between -3 and 3, making it a valid QSD digit. The second step is local, and the overall addition time is the time needed to add two digits, regardless of the operands' digit count. Thus, no second-level carry occurs. With carry-free computing, the QSD adder maintains a constant delay.

This flow is shown with a worked example. Assigning numbers 9 and 9 to variables A = (2, 1) and B = (3, −3) yields the expected sum of 18. Since the least significant position is 1 plus (-3), the intermediate total is -2 and the carry is 0. In the next slot, a carry of one and an intermediate sum of one total five. The second phase produces the output values S0 = -2 + 0 = -2, S1 = 1 + 0 = 1, and S2 = 0 + 1 = 1. The result (1,1,−2)=16+4−2=18 meets the conditions. The output of each digit was parallel.

Advanced Research and Development Journal
ISSN: 3139-1281| Volume 2 Issue 2 | April-June 2026 | ARD-046
www.ardjournal.com | DOI: https://doi.org/10.5281/zenodo.21257903

V. PROPOSED ADDER DESIGN

Figure shows the flat array of identical digit slices that make up the adder. a. 1. Since both calculation processes are local, the entire array is evaluated simultaneously. All first-step generators and second-step adders only know their own operands and the single carry from their right neighbor. Since there is no word-wide path, increasing the operand width does not expand the essential path.

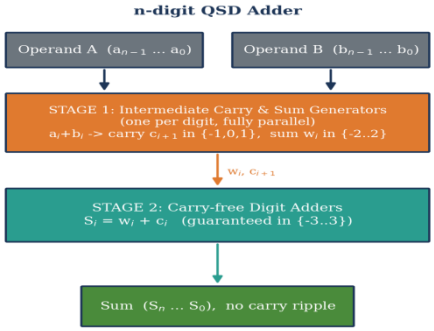
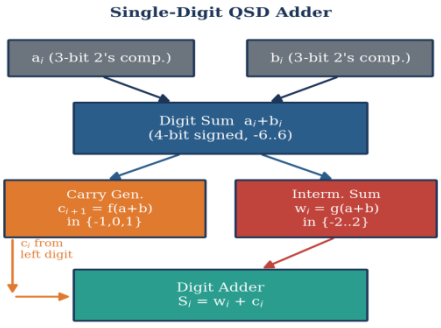


Fig. a. 1. A two-stage carry-free QSD adder has an intermediate carry/sum generation step and a carry-free digit-addition stage. Two levels are completely parallel on digits.

A. Single-Digit Slice

The figure shows part of a number. The number two has three little parts. Add the two 3-bit operand digits to get the 4-bit signed digit total. Second, two combinational generators turn the digit sum into the intermediate carry and intermediate sum, as shown in Table I. Third, a small signed adder combines the local intermediate sum with the surrounding lower slice carry to produce the final output digit. Doing this produces the final output digit. With a guaranteed output range of {-3, -3}, this terminal adder does not need carry-out logic to perform properly.



See Figure 2. Adder segment with one QSD digit. The second adder adds the intermediate sum to the input carry, and the digit sum powers the discrete carry and intermediate sum generators.

B. Logic Optimization

Advanced Research and Development Journal
ISSN: 3139-1281| Volume 2 Issue 2 | April-June 2026 | ARD-046
www.ardjournal.com | DOI: https://doi.org/10.5281/zenodo.21257903

Most optimization is done by generation logic. Instead of assuming the two generators are independent lookup functions of the 4-bit digit sum, the sign and magnitude are shared once. Intended to save time. The carry is a thresholded function that returns +1 if the digit total is three or more, -1 if it is three or fewer, and 0 otherwise. Given this, it may be reduced to a comparison using +3 and -3 and a two-level network. The intermediate sum is the digit sum after the carry weight of four is subtracted, and the same sign signal can be used again. Slice delay, a small constant, is about two gate levels. By exchanging these words, duplicate gates are eliminated and each slice's two-level depth is kept tiny.

Before the first step, an add/subtract control conditionally inverts the B-operand digits. Negating in two's complement is bit-inversion plus one, and subtracting digits means negating them. A slice can work as an adder–subtractor with little added cost, which is useful when the block is repeated in MAC units and multipliers.

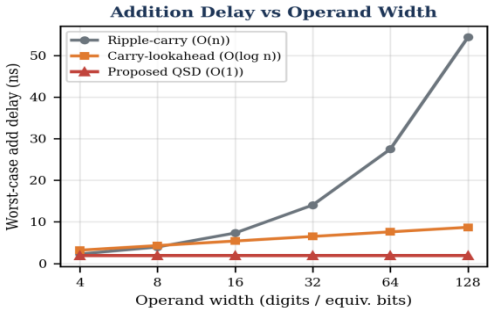
VI. RESULTS AND DISCUSSION

The architecture is compared to four popular adder families in Table II. The binary adders differ mostly in how much they attack the carry chain, although they all have a delay that grows with operand width. The proposed QSD adder achieves constant latency at radix four and packs more range into each slice. The redundant-binary adder uses radix-2 signed digits for a constant delay but a limited range per digit.

TABLE II
COMPARISON OF ADDER ARCHITECTURES

Adder	Radix	Redundant	Carry handling	Delay
Ripple-carry	2	No	Full ripple	O(n)
Carry-lookahead	2	No	Look-ahead	O(log n)
Carry-select	2	No	Speculative	O(√n)
Redundant binary	2	Yes	Limited to 1 pos.	O(1)
Proposed QSD	**4**	**Yes**	**Carry-free**	**O(1)**

Figure summarizes delay behavior. 3. The ripple-carry delay is proportional to width, but the carry-lookahead delay is logarithmic. Both delays match breadth. In contrast, QSD delay is straight. The two-step technique takes the same time with eight- or sixty-four-digit operands. The QSD adder is faster than the lookahead adder above the crossing, which is easy to notice at intermediate widths. This matches QSD adder literature, which shows that the advantage steadily increases after that.



Advanced Research and Development Journal
ISSN: 3139-1281| Volume 2 Issue 2 | April-June 2026 | ARD-046
www.ardjournal.com | DOI: https://doi.org/10.5281/zenodo.21257903

Fig. 3. Worst-case addition delay versus operand width (illustrative). The QSD delay is independent of width, while binary adders grow with n.

Prolonged delays stress the region. Each slice has a small digit sum, two generators, and a second adder, increasing its transistor count compared to a binary full adder. This is because of slice building. Table III's final column shows the total transistor count scaling linearly with board width. The area penalty is moderate because the slices have short wiring and this linear scaling is similar to a ripple-carry adder. Thus, the area penalty is low. Fig. Figure 4 compares the representative per-slice transistor count and normalized power for the five layouts. The QSD slice sits between the ripple-carry cell and the larger lookahead and carry-select cells because there is no high-activity carry chain switching throughout the word. This arrangement keeps the QSD slice low-power.

TABLE III
REPRESENTATIVE DELAY AND AREA VERSUS OPERAND WIDTH

Width (digits)	RCA (ns)	CLA (ns)	QSD (ns)	QSD txr.
8	4.0	4.3	1.9	272
16	7.3	5.4	1.9	544
32	14.0	6.5	1.9	1088
64	27.5	7.6	1.9	2176

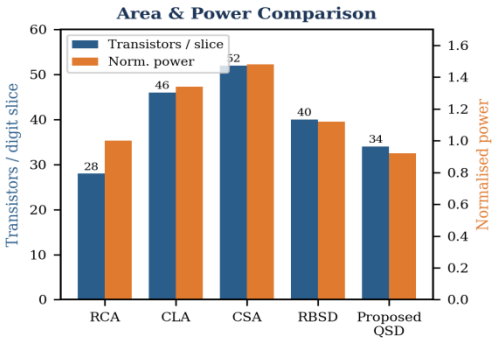


Fig. 4. Representative per-slice transistor count and normalized power for the compared adders (illustrative values).

The findings indicate that the QSD adder chooses between a delay that is no longer dependent on operand width and a well-bounded and relatively tiny area gain. The QSD adder is used to produce carry-free multipliers and MAC units because it is an attractive exchange for long datapaths like DSP and cryptographic hardware accumulators and partial-product reductions. Its main use as a building block is this. Note that the data above are indicative illustrative figures chosen to illustrate the patterns stated in the related article. Designers interested in exploring a technology should renew simulation data at the synthesis and transistor levels before drawing quantifiable findings.

A. Applications in VLSI

The QSD adder is useful in addition-heavy datapaths due to its constant delay function. This component mainly reduces carry-free multipliers and multiply–accumulate units. These units use a tree of QSD adders to add partial products, with the delay defined by the depth rather than the breadth. The slice also supports FIR and IIR digital filter accumulators and FFT engine butterfly enhancements. Since word expansion would be needed, each binary carry

Advanced Research and Development Journal
ISSN: **3139-1281**| Volume 2 Issue 2 | April-June 2026 | ARD-046
www.ardjournal.com | DOI: https://doi.org/10.5281/zenodo.21257903

chain would be longer without this functionality. QSD arithmetic is signed and matches residue and modular arithmetic, which are used in cryptographic hardware, hence multiple-valued logic and optical computing have often targeted it. Characteristics can also convert the digit set into more than two logic levels. In all cases, the adder is inside a QSD arithmetic island, and binary to QSD conversion occurs only at its boundaries. This amortizes conversion costs across several processes.

B. Limitations

The QSD approach has limitations. Because each digit requires three bits, rather than the minimum necessary by information theory, storage and connectivity are larger than a packed binary word. Conversion logic added to the edges of a QSD island when converting binary to QSD may limit the benefits of isolated short additions. The carry chain is deleted due to duplication, therefore there are multiple value encodings, making comparison and equality testing harder. The architecture is most useful when integrated into large addition-dominated arithmetic pipelines rather than sequentially replacing each adder.

VII. CONCLUSION

A carrier-free quaternary signed digit adder tailored for VLSI applications was built for this project. Adders use narrower input windows to compute each output digit. A radix-4 redundant representation is used in addition to the two-step generating method. Thus, the worst-case delay of this operation is independent of operand breadth. The compact three-bit two's complement encoding and sign and threshold term sharing compress each digit slice to a short two-level network. Same slice might be adder and subtractor. Using a 45 nm transistor cost of the same linear order as a ripple-carry adder, the delay is nearly constant from 8 to 64 digits. However, ripple-carry and carry-lookahead adders delay more with more digits. The numbers are shown for demonstration, but the reader should verify them using synthesis and SPICE data. Expanding the slice into a carry-free QSD multiplier and quantifying the binary-to-QSD conversion overhead in a whole datapath are the next steps. Both steps are obvious.

REFERENCES

[1] *A. Avizienis, "Signed-digit number representations for fast parallel arithmetic," IRE Trans. Electron. Comput., vol. EC-10, no. 3, pp. 389–400, Sep. 1961, doi: 10.1109/TEC.1961.5219227.*

[2] B. Parhami, "Generalized signed-digit number systems: a unifying framework for redundant number representations," IEEE Trans. Comput., vol. 39, no. 1, pp. 89–98, Jan. 1990, doi: 10.1109/12.46283.

[3] B. Parhami, "Carry-free addition of recoded binary signed-digit numbers," IEEE Trans. Comput., vol. 37, no. 11, pp. 1470–1476, Nov. 1988.

[4] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs, 2nd ed. New York, NY, USA: Oxford Univ. Press, 2010.

[5] K. Hwang, Computer Arithmetic: Principles, Architecture and Design. New York, NY, USA: Wiley, 1979.

Advanced Research and Development Journal
ISSN: **3139-1281**| Volume 2 Issue 2 | April-June 2026 | ARD-046
www.ardjournal.com | DOI: https://doi.org/10.5281/zenodo.21257903

[6] P. K. Dubey and R. Rani, "VLSI implementation of fast addition using quaternary signed digit number system," in Proc. IEEE Int. Conf. Emerging Trends Comput. Commun. Technol., 2013.

[7] P. K. Dakhole and D. G. Wakde, "Multi-digit quaternary adder on a programmable device: design and verification," in Proc. Int. Conf. Electron. Design (ICED), Dec. 2008, pp. 1–4.

[8] J. U. Ahmed and A. A. S. Awwal, "Multiplier design using RBSD number system," in Proc. IEEE Nat. Aerosp. Electron. Conf. (NAECON), vol. 1, 1993, pp. 180–184.

[9] N. Takagi, H. Yasuura, and S. Yajima, "High-speed VLSI multiplication algorithm with a redundant binary addition tree," IEEE Trans. Comput., vol. C-34, no. 9, pp. 789–796, Sep. 1985.

[10] O. Ishizuka, A. Ohta, K. Tanno, Z. Tang, and D. Handoko, "VLSI design of a quaternary multiplier with direct generation of partial products," in Proc. IEEE Int. Symp. Multiple-Valued Logic (ISMVL), 1997, pp. 169–174.

[11] J. Moskal, E. Oruklu, and J. Saniie, "Design and synthesis of a carry-free signed-digit decimal adder," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), 2007, pp. 1089–1092.

[12] G. C. Cardarilli, M. Ottavi, S. Pontarelli, M. Re, and A. Salsano, "Error detection in signed-digit arithmetic circuits with parity checkers," in Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst. (DFT), 2003, pp. 401–408.

[13] G. C. Cardarilli, M. Ottavi, S. Pontarelli, M. Re, and A. Salsano, "Localization of faults in radix-n signed-digit adders," in Proc. IEEE Int. On-Line Testing Symp. (IOLTS), 2006.

[14] S. Kaza, M. B. Srinivas, and others, "High-speed and low-power quaternary signed-digit adder design," in Proc. IEEE Int. Conf. VLSI Design, 2012.

[15] A. K. Cherri, "Canonical quaternary signed-digit arithmetic using optoelectronics symbolic substitution," Opt. Laser Technol., vol. 30, no. 8, pp. 483–493, 1998.

[16] I. Koren, Computer Arithmetic Algorithms, 2nd ed. Natick, MA, USA: A. K. Peters, 2002.

[17] T. Hanyu and M. Kameyama, "A 200 MHz pipelined multiplier using 1.5 V-supply multiple-valued MOS current-mode circuits," IEEE J. Solid-State Circuits, vol. 30, no. 11, pp. 1239–1245, Nov. 1995.

[18] S. Wei and K. Shimizu, "Residue arithmetic circuits using a signed-digit number representation," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), 2000.