International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 05 | May 2019 www.irjet.net p-ISSN: 2395-0072

Implementation of Radix-16 and Binary 64 Division VLSI Realizations for Energy Efficiency and Low Power Dissipation

SRIKANTH IMMAREDDY

Electronics and Communication Engineering Dept, Methodist College of Engineering and Technology, Hyderabad 500001, India.

Abstract - ***The chip design and its fabrication process use the VLSI realizations to reduce the complexities. This paper presents the VLSI realization process for the digit-recurrence binary division and, it uses the redundant representation of partial remainders and quotient digits. The fast carry-free computation for next partial remainder is allowed by the partial remainders and, the quotient digits help to reduce the required divisor multiples. A four-division architectural approach has proposed in this paper for exploring the design space and, this novel idea is based on binary CS or radix-16 signed digit (SD) representations of partial remainders. On the other end, the partial remainders use the full or partial pre-computation of divisor multiples and to maintain the consistency in the design operations an operand must be constant in all cycles. This operand is additionally added to the system by a small multiplexer at the cost of two adders. A radix-16 [−9, 9] SDs stage is used to represent the quotient digits and, the proposed method synthesis results achieve the better than the previous works. When compared with the reference work the power and the energy-delay of the product are 26%–35% less.***

Index Terms —VLSI realizations, Digit-recurrence binary division, Partial remainders, Quotient digits, radix-16, Signed digit (SD) representation

1. INTRODUCTION

Execution of complex problems using the basic arithmetic operations on digital processors needs millions of transistors and an innovative technology named as "Very Large Scale integration (VLSI)" known for its ability to execute the complex problems by using basic mathematical operations such as addition, subtraction, multiplication, and division. As mentioned in various research papers the division operation is the least used operation among all four. The execution of the typical complex problems is carried out smoothly by the division operation. It takes more time for the execution process and simultaneously the complexity levels are high because of high time consumption. The VLSI circuitry and its design process need realization approach for executing the algorithms within the prescribed timeline. The dividers used in the VLSI realization are classified as follows,

S.NO.	Digital recurrence	Functional
1	It is popularly known as subtractive operation.	It is popularly known as multiplicative operation.
2	It is a kind of realization divider acts based on two classes of algorithms.	It is a kind of realization divider acts based on two classes of algorithms.
3	It is cost effective.	It is a little bit expensive than the digit-recurrence.
4	It is slow such that it requires a separate recurrence cycle for obtaining each digit of the quotient.	The number of iterations is logarithmically proportional to the number of quotient digits.

Table 1: Digital recurrence vs Functional

1.1 Quotient digit selection (QDS)

The division schemes are classified into two types namely storing and non-storing based on the application. The traditional binary algorithms are simple in their design such as non-storing division scheme, where the subsequent quotient bit is obtained just by examining the sign of partial remainder. A radix-2h (e.g., h = 4) division scheme has been proposed to reduce the number of recurrences at the cost of more complex QDS and, this approach helps to select the one out of 2h possible digit values.

1.2 Signed Digit Number Systems

The usage of the high radix SD number systems for representing the partial remainders and which helps to carry the specified carry-free addition based on recurrence. A reference of radix-16 SD representation of partial remainders in two of the proposed designs in this paper and in one previous work [13]. Almost all previous works fail to indicate a specified digit set to represent the radix-16 stage but the proposed design comes up with new indication parameter known as MRSD. In the previous works that use radix-16 MRSD number system (see [4], [5]), each radix-16 MRSD digit is represented by a 5-bit two's complement encoding. In [5], the most significant bit is a negabit with arithmetic value −1(0) corresponding to logical status 0(1) [14]. The below figure 1 shows the two digit slice of the required carry-free addition, where white (black) dots represent negabits (posibits). One 4-bit adder is allocated to each radix-16 position and, it repeats for every radix-16 position. Once the allocation process is accomplished, the sum digits are produced using the input bits. The allocated 4-bit adders produce the necessary negabit and three posibitsfo the sum digit for each radix-16 position. The proposed four designs use the carry save (CS) adder is leading to less power dissipation due to the use of the aforementioned adder. The low power dissipation is due to usage of 5 bits compare of the 8 bits of the previous works.

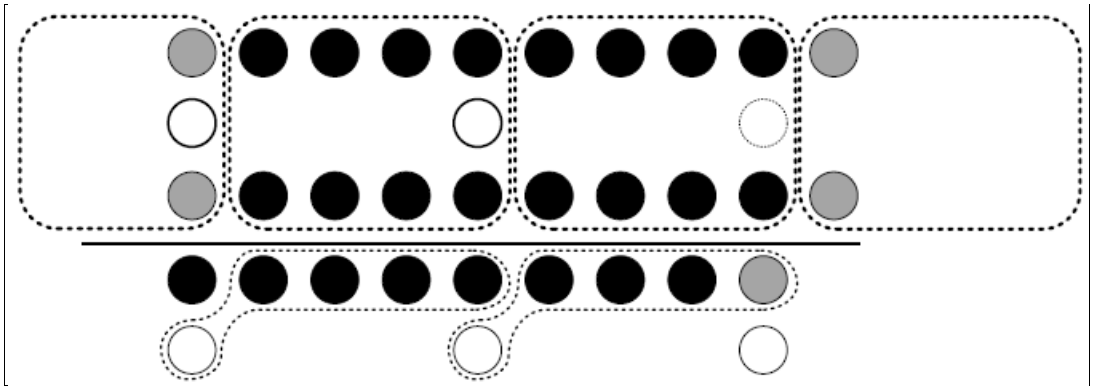

Figure 1: Two digit slices of the carry-free addition (CFA) for partial remainder computation (PRC)

2. BACKGROUND

2.1 VLSI

The very-large-scale integration (VLSI) has brought revolutionary changes in both design and testing of micro-chip. According to Moore's law the usage of transistors in chip design increases dual times for a margin of 18 months and standard research organizations defined very-large-scale integration (VLSI) as combination of millions transistors which results in micro chip of size (tens of nanometers). The millions of FETS integration through very-large-scale integration (VLSI) to perform operations of predominant research areas like medical content processing, digital signal processing and advance robotics for artificial intelligence creation.

2.2 Power Efficient Division

Although division and square root are not frequent operations, most processors implement them in hardware to not compromise the overall performance. Two classes of algorithms implement division or square root: digit-recurrence and multiplicative (e.g., Newton-Raphson) algorithms. Previous work shows that division and

square root units based on the digitrecurrence algorithm offer the best tradeoff delay-areapower. Moreover, the two operations can be combined in a single unit. Here, we present a radix-16 combined division and square root unit obtained by overlapping two radix-4 stages. The proposed unit is compared to similar solutions based on the digit-recurrence algorithm and it is compared to a unit based on the multiplicative Newton-Raphson algorithm.

3. LITERATURE SURVEY

The knowledge abundance in the digital arithmetic is not enough to create the new generation processors free of problems. The researchers, graduates, and designers will find the appropriate solutions. In the past, the design of the processors based on digital arithmetic is to be technology dependent. In 2003, Ercegovac and Lang, the two prolific experts of the digital arithmetic design a unified treatment of digital arithmetic, to encourage the researchers towards technology independent and, they conclude that this concept helps in reducing the problems in a diplomatic way [1].

The division operation has a negative impact on the area and the performance of the processor. Then three noted researchers namely Sweeney, Robertson, and Tocher (1958) diagnose that the usage of the traditional floating points is a reason behind the low division performance. They proposed a new concept called SRT [Sweeney, Robertson, and Tocher] dividers and these dividers attain the higher performance and low complexity by retiring more quotient bits in each cycle. The previous research works reveal that the realistic stages are limited to radix-2 and radix-4 and, these low-radix stages are effectively combined to form higher radix dividers. The effect of the radix-2 and radix-4 SRT divider's architecture on divider area and performance is analysed. This idea significantly improved the divider performance by aggressive circuit techniques [2].

Previous Works			
Reference Author(s)	**Focus**	**Objective**	**Main Findings and Conclusions**
Morgan Kaufmann, **2003**	Digital Arithmetic	The digital arithmetic has been an essential research area in the processor's design. The prominent application-oriented research fields such as digital image processing, digital signal processing, graphical designing systems, and communications use the digital arithmetic for designing the processors for various applications based on the complexity.	A definitive reference and a consistent teaching tool for developing a deep understanding of the "arithmetic style" of algorithms and designs.
Tony M. Carter & James E. Robertson**1990**	Radix-16 signed-digit division	The algorithm requires a two-digit estimate of the (initial) partial remainder and a three-digit estimate of the divisor to correctly select each successive quotient digit. The normalization of redundant signed-digit numbers requires accommodation of some fuzziness at one end of the range of numeric values that are considered normalized. A set of general equations for determining the ranges of normalized signed-digit numbers is derived.	The staged division algorithm used can be extended to other radices as long as the signed-digital number representation used has certain properties.

Previous Works			
Reference Author(s)	**Focus**	**Objective**	**Main Findings and Conclusions**
Lee et al. (2005)	A Modified Booth‟s algorithm (MBA)	The proposed multiplier inherits the advantage of the MBA and then reduces both space and time complexities. A multiplexer-based structure is proposed for realization of the proposed algorithm. The authors have shown that their multiplier saves about 9% space complexity as compared to former multipliers, if the generating polynomial is trinomial or all one polynomial	The proposed multiplier inherits the advantage of the MBA and then reduces both space and time complexities.

4. PROPOSED METHODOLOGY

4.1 Intended Architectures

The schematic diagram of the general digit recurrence division architecture is shown in Fig. 2. The relevancies of the proposed designs are provided as appropriate. All proposed designs are set to be double precision operands (i.e., binary64 floating point) and, it has set as per the main reference [6].

The above figure 2 states the Two different representations namely static and semidynamic DMGs and, it helps to provide the design space based on the following four options.

(A) Radix-16 Quotient Digit Set

- The radix-n stages are more prevalent in many of the previous works and as well as in the proposed method.
- It helps to reduce the number of cycles.
- Radix-16 Quotient Digit Set happened as the direct generation of quotient bits vs the reduced number of cycles.

(B) SD Representation of Quotient Digits

- Once the direct generation of the quotient bits happened as stated in the previous step and here a [−9, 9] radix-16 SD set is used to represent the intermediate representation of quotient digits.
- As discussed in the introduction chapter, many of the previous works tend to use the 8 bits which in turn results in the high power consumption and whereas in the proposed methodology 5-bit representation of MSRD is used as shown in figure 1.
- The initial redundant choice is [−8, 8] digit set which needs a minimum number of divisor multiples.
- The maximum choice is [−15 15] digit set which needs a maximum number of divisor multiples.
- Another interesting set of bits that taken into consideration is fractional digits, which are sufficient for truncated comparison of partial remainders with divisor multiples. This is later shown to be 2 in thecase of digit sets [−α, α] (α ∈ [9, 15]), and 3 for α = 8.

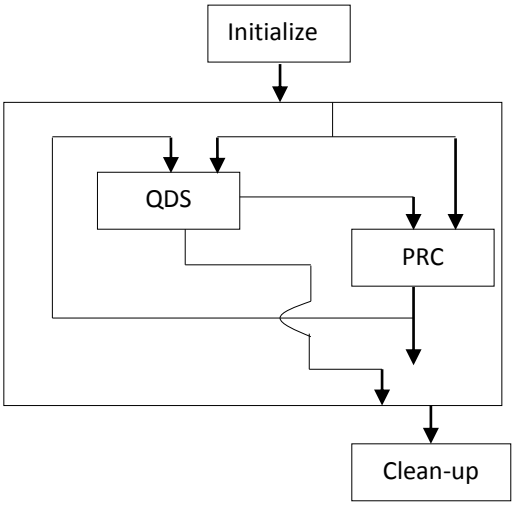

Figure 2: General division architecture.

(C) Semidynamic DMG

- The initialization cycle is used by the partial remainder computation (PRC) [−9, 9] multiples of the divisor, where the ten-way multiplexer is required for selecting q j+1D within the initialization cycle.
- Although the proposed use the conventional method for implementation and it also uses a four-way multiplexer generate only {2, 3, 6}D, and dynamically obtain ±{4, 5, 7, 8, 9}D, as ±{6D − 2D, 6D ∓ D, 6D + 2D, 6D + 3D}, respectively, within each recurrence.

(D) Use of Redundant Number Systems for PRC

- A significant amount of work has been carried out in the past and most of the conventional works opted for CS representation of partial remainders.
- To show the advantage of the aforementioned semidynamic DMG, the CS use as one option and it doubles the representation storage to yield the desirable lower power dissipation.
- Apart from the carry save, the proposed methodology uses the higher radix redundant number systems as another choice for the partial remainder representation. For example, radix-16 maximally redundant number system (MRSD) [5] is a viable choice, with only 25% extra representation storage.

  The above mentioned architectural description makes it sure that the proposed methodology will have the following attributes to its credit.
  - CS-10, CS-4,
  - MRSD-10, and MRSD-4, where CS and MRSD regard the partial remainder representation, and 4 and 10 refer to the sizes of utilized multiplexers,

4.2 Quotient Digit Selection

The quotient digit set has high-level prominence as stated in the previous works and the proposed choice of quotient digit set is [−9, 9]. As discussed in the previous sections, the next quotient digit q_{j+1} should be selected from [−α, α] based on the convergence condition and in the proposed approach it is taken as (i.e., α = 9), ρ = (a/(16 − 1)) = (9/15) = 0.6.

The algorithmic approach of the quotient digit is as follows

a) The comparison of the partial remainder with a set of divisor multiples are done based on the convergence condition as displayed in figure 2 and it is stated as (e.g., −1.6D and −0.4D, for q j+1 = −1) and finally, it helps to decide the value of the next quotient digit.

b) However, for some W[j] values (e.g., 16W[j] = −.5 D), there may be more than one valid q j+1. For example, see the overlapped zone between the dashed lines for q j+1 = −1 and q j+1 = 0.
c) A pre-computation of a set of comparison constants helps to ease the process of QDS at one end and, at the other end; it helps to reduce the number of comparisons which eventually helps to reduce the complexity. Therefore, an ease to compute choice is Mk = (k + 0.5)D, which leads to the case that the exact interval of 16W[j] (for a particular value of q j+1 = −1) falls between M−2 = −1.5D and M−1 = −.5D (see the corresponding bold arrows in Fig. 3)
$$D(k+0.4) \le M_k \le D(k+0.6) \quad (4)$$
d) The comparison of an SD or CS number cannot be performed by digit by digit comparator as it won't be trustworthy anymore. The digit comparator but at the cost of with respect to losing the carry-free advantage of SD or CS number systems
e) The proposed methodology opts for significant fractional bits over the conventional comparator and, he digit by digit comparator and the fractional bits simply cannot lose the advantage of having the carry-free advantage of SD or CS number systems.

For example, consider the first two digits of a hexadecimal MRSD operand as X = x1x0, and nonredundant number Y = y1 y0, and assume x1 = 8, x0 = −2, y1 = 7, and y0 = 15. A normal comparator rules X > Y, since x1 > y1. However, it turns out that X can be equivalently represented as x1 = 7 and x0 = 14. Therefore, the same trivial comparison scheme leads to X <Y , since x0 = 14 < 15 = y0.

4.2.1 Proof of t = 2

Let's assume that "t" fractional digits is for Mkand W, and truncated operands be denoted as Mk − 16−t < (Mk)t ≤ Mk and 16W[j] − 16−t < (16W[j])t < 16W[j] + 16−t , respectively. In the latter case, the discarded bits are assumed as the positive as well as negative due to SD nature of the partial remainders. A similar procedure can be adapted to the decimal values by adapting it to the radix-16 division. Replacing the operands of (2) with the corresponding truncated operands, we get at (5) and (6), respectively

$$(M_k)_t \le (16W[j])_t, for q_{j+1} = k (\mathbf{5})$$

$$16W[j]_t < (M_k)_t, for q_{j+1} = k+1 \quad (\mathbf{6})$$

Applying these results on the convergence condition in (3) (i.e., −ρD<W[j + 1] and W[j + 1] <ρD) leads to (7) and (8), respectively. After doing the much needed changes in the above two equations, they can be represented as follows,

$$(5) \Longrightarrow M_k - 16^{-t} < (M_k)_t \le 16W[j]_t < 16W[j] + 16^{-t} \Longrightarrow M_k - 2X16^{-t} - KD < 16W[j] - kD = W[j+1]$$

$$(6) \Longrightarrow 16W[j] - 16^{-t} < 16W[j]_t < (M_k)_t \le M_k \Longrightarrow M_k + 16^{-t} - (k-1)D > 16W[j] - (k-1)D = W[j+1] - \rho D < M_k - 2X16^{-t} - KD \Longrightarrow 2X16^{-t} + KD\rho D < M_k (\mathbf{7})$$

$$M_k + 16^{-t} - (k-1)D < \rho D - 16^{-t} - (k-1)D (\mathbf{8})$$

Recalling that ρ = 0.6 and 1/16 ≤ D < 1, and combining the inequalities (7) and (8), we get at the following:

$$\rho D + 2X - 16^{-t} + KD < \rho D - 16^{-t} - (k-1)D \Longrightarrow 3X16^{-t} < (2\rho - 1)D = 0.2XD \Longrightarrow 16^t > \frac{15}{D}$$

However, given the existence of a non-fractional integer digit of the operands, three-digit comparisons are required, for which the cost of converting into two's complement can be afforded.

4.3 Partial Remainder Computation

The Partial Remainder Computation is the one aspect which is mainly involved in all the four architectural designs and, it is described in this section.

4.3.1 CS-10 and MRSD-10 Designs

Here a straightforward PRC is taken into account and which use a unified carry-free adder/subtractor (CFA/S) and a 10:1 multiplexer, for the quotient digit set [−9, 9]. a digit is meant to denote either an MRSD or four CS digits, and QDS box represents Fig. 4, whose output control signals are shown as "MUX Selector" and "ADD/SUB."

The ten-way multiplexing is expected to be highly influential in prolonging the latency. Recalling the PRC, as W[j + 1] = 16W[j] − q j+1D, note that W[j] is maintained either in radix-2 CS or radix-16 MRSD, and the multiples of D are represented in binary. Therefore, the PRC is simply done via a CS

addition (for the CS-10 case), and by a 4-bit

adder per radix-16 digit (for the MRSD-10 case), as was described in Section II-B. The nonredundant binary dividend X must be converted into redundant representation to provide for W[0].

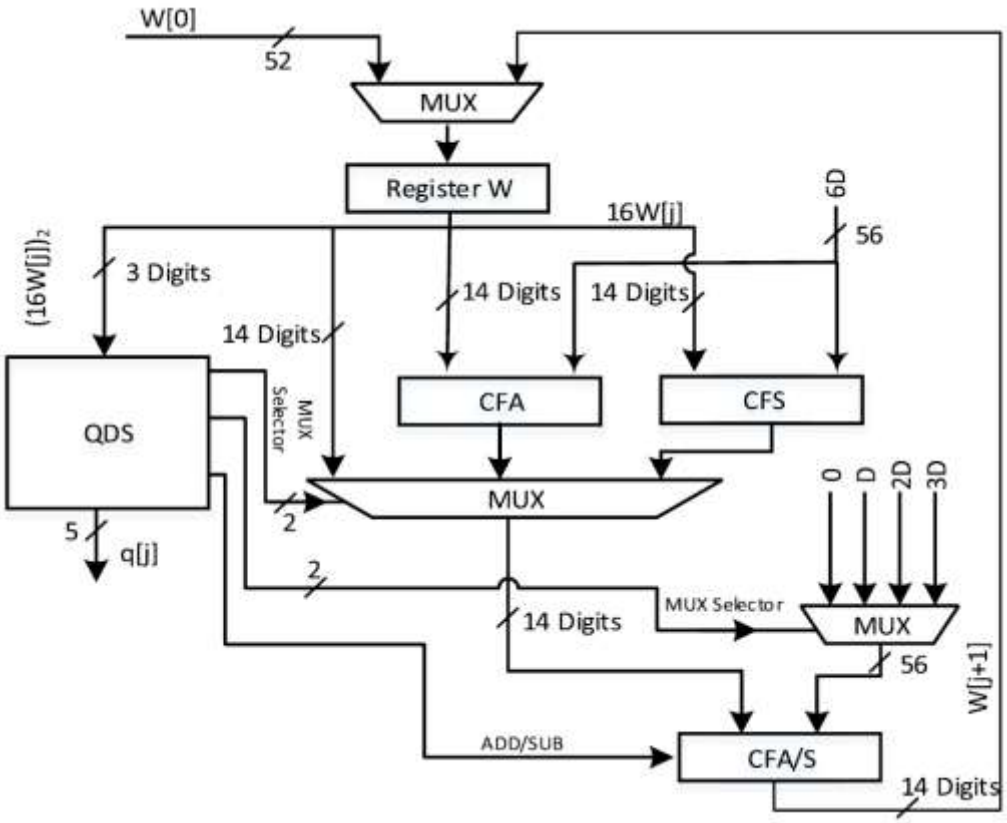

Figure 3: PRC for designs CS-4 and MRSD-4

4.3.2 CS-4 and MRSD-4 Designs

In order to utilize a smaller selector of divisor multiples, we propose the architecture of Fig. 6, where again QDS box represents Fig. 4, whose output control signals are shown as "MUX Selector" and "ADD/SUB." The operation of this part overlaps with the QDS. The rest of PRC is carried out after QDS, where βD is selected via the four-input multiplexer, followed by computing W[j + 1] = W[j + 1]′ + βD, via another CFA.

4.4 Initialization and Clean-Up

The initialization cycle is responsible for the following tasks.

1) Converting X to W[0]: This is a cost- and delay free operation. In case of CS designs, zero-valued bits are inserted as the second bits of CS representation. However, in MRSD cases, a zero-valued negabit (with logical status 1) is added per each four bits of X to lead to the radix-16 MRSD representation of W[0].

2) Parallel Computation of the Divisor Multiples: a) CS-10 and MRSD-10 Designs: There are four shifters (for 2D,4D,6D, and 8D) and four parallel adders (for 3D = 2D + D,5 D = 4D + D,7 D =8D−D, and 9D =8D+D).

3) Parallel Precomputation of the Truncated Comparison Constants: This is done, as in the following expressions for M0 to M8.

CS-4 and MRSD-4 Designs:

M1 = D/2, M2 =3D/2, M3 =2D+M1

M4 = 3D+M1, M5 =4D+M1

M6 = 4D+M2, M7 =6D+M1

M8 = 6D+M2, M9 =8D+M1.

The terminal cycle is responsible for the conversion of the obtained quotient and remainder to nonredundant binary format. Such conversionentails word wide binary addition that is undertaken via a parallel prefix binary adder in both CS and MRSD cases, which also takes care of the required rounding addition.

5 RESULTS AND ANALYSIS

5.1 Proposed Results

1. The improvement in power and EDP measures also follow the following.

1. The power dissipation in comparison to the average of the power measures of the four proposed designs is about 62% more.

2. The area consumption is, on the average, 35% less than that of the proposed designs. However, that of the former is more when excluding the area measure of the initialization stages.

3. The least achieved delay is, on the average, 26% less than that the proposed designs

4. The energy figures of the proposed designs are, on the average, 48.5% less

5. The EDP of the proposed designs is, on the average, about 30% less

6. The EDP of the MRSD-4 design is, on the average, 15.5% less than that of our other three designs. The energy-per division of the MRSD designs are, on the average, 10% less than those of the CS designs.

7. The E × Area of all the proposed designs and the EDP × Area of MRSD-4 are less.

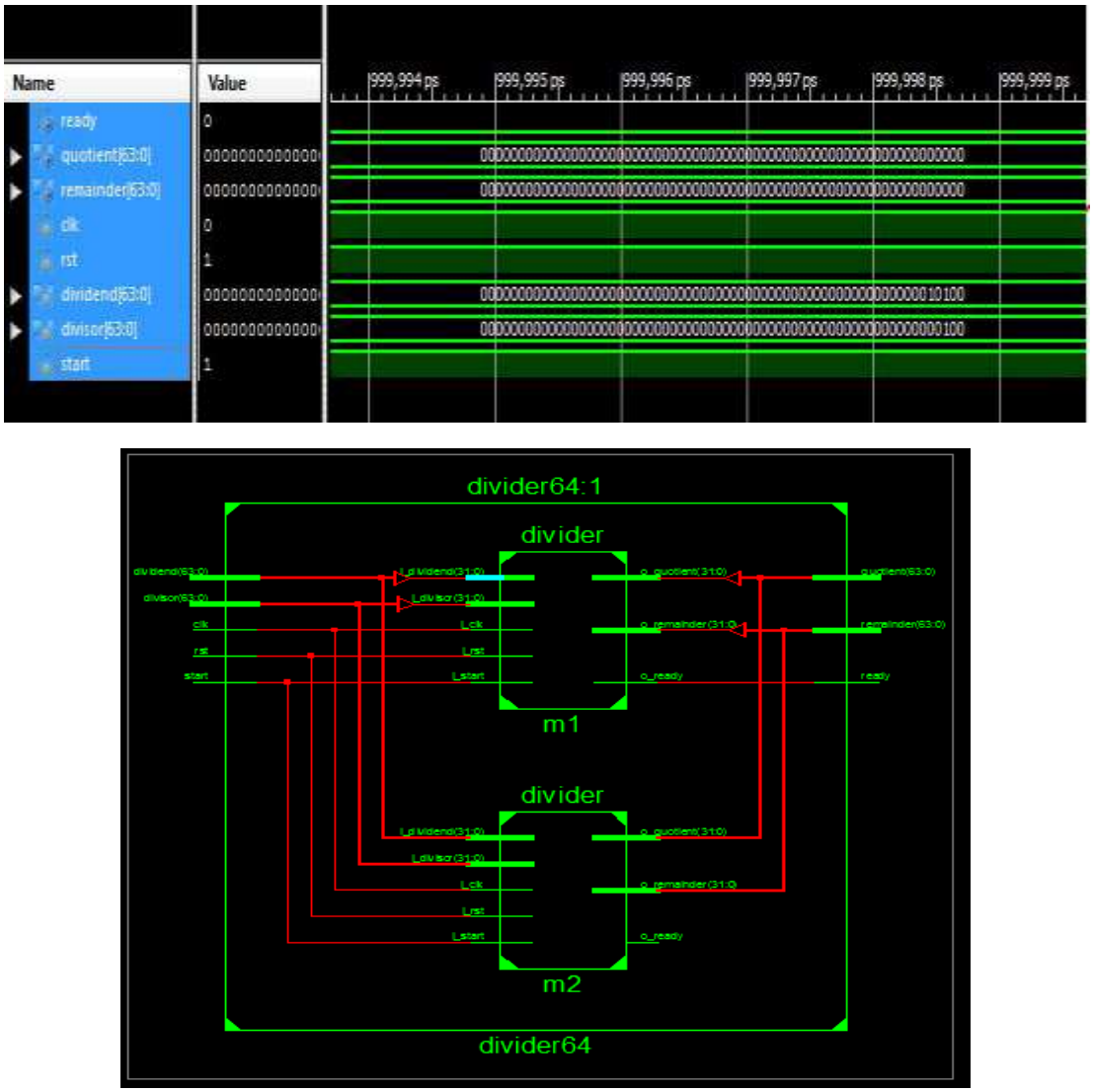

Figure 4: RTL Schematic

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	1793	4656	38%
Number of Slice Flip Flops	666	9312	7%
Number of 4 input LUTs	3427	9312	36%
Number of bonded IOBs	260	232	112%
Number of GCLKs	1	24	4%

Figure 5: Design and Summary

```
Timing Summary:
---------------
Speed Grade: -5

   Minimum period: 9.991ns (Maximum Frequency: 100.092MHz)
   Minimum input arrival time before clock: 7.606ns
   Maximum output required time after clock: 5.373ns
   Maximum combinational path delay: 6.106ns
```

Figure 6: Timing Report

6. CONCLUSION

We studied the impact of the following two design options on the figures of merit of binary digit-recurrence division hardware. 1) Representation of partial remainders via high radix redundant number systems. Our representation choice is maximally redundant radix-16 SD number system with digit set [−15, 15]. 2) Dynamic generation of some divisor multiples in [−9, 9] ×D around the pre-computed multiple 6D. We also studied the relevant previous designs, which have opted for binary CS representation of partial remainders and representation of radix-16 quotient digits via minimally redundant radix-4 [−2,2] digits, which leads to partial dynamic generation of divisor multiples. For better evaluation of the above options, we explored a design space containing four architectures based on pre-computation of all or part of divisor multiples and CS or MRSD representation of partial remainders. The HDL simulations and synthesis showed low-power and low-energy advantages of all the four designs as compared with the best previous work. However, while our designs do not operate as fast as the reference one, the EDP of the proposed designs is 26%–35% less than that of the reference work.

7. REFERENCES

[1] M. Ercegovac and T. Lang, Digital Arithmetic. San Mateo, CA, USA: Morgan Kaufmann, 2003.

[2] D. L. Harris, S. F. Oberman, and M. A. Horowitz, “SRT division architectures and implementations,” in Proc. 13th IEEE Symp. Comput. Arithmetic, Jul. 1997, pp. 18–25.

[3] A. Avizienis, “Signed-digit number representations for fast parallel arithmetic,” IRE Trans. Electron.Comput., vol. 10, no. 3, pp. 389–400, Sep. 1961.

[4] H. A. H. Fahmy and M. J. Flynn, “The case for a redundant format in floating point arithmetic,” in Proc. 16th IEEE Symp. Comput.Arithmetic, Santiago de Compostela, Spain, Jun. 2003, pp. 95–102.

[5] S. Gorgin and G. Jaberipur, “A family of high radix signed digit adders,” in Proc. 20th IEEE Symp.Comput. Arithmetic, Tübingen, Germany, Jul. 2011, pp. 112–120.

[6] W. Liu and A. Nannarelli, “Power efficient division and square root unit,” IEEE Trans. Comput., vol. 61, no. 8, pp. 1059–1070, Aug. 2012.

[7] J. Ebergen and N. Jamadagni, “Radix-2 division algorithms with an overredundant digit set,” IEEE Trans. Comput., vol. 64, no. 9, pp. 2652–2663, Sep. 2015.

[8] A. Nannarelli and T. Lang, “Low-power divider,” IEEE Trans. Comput., vol. 48, no. 1, pp. 2–14, Jan. 1999.

[9] A. Nannarelli and T. Lang, “Low-power division: Comparison among implementations of radix 4, 8 and 16,” in Proc. 14th IEEE Symp. Comput. Arithmetic, Apr. 1999, pp. 60–67.

[10] T. M. Carter and J. E. Robertson, “Radix-16 signed-digit division,” IEEE Trans. Comput., vol. 39, no. 12, pp. 1424–1433, Dec. 1990.