REALIZATION OF DECIMAL MULTIPLICATION USING SIGN MAGNITUDE ENCODING

P. GAYATHRI[1], GOWTHAMI [2]
[1]PG Scholar, ECE Department, ALTS, Anantapur
[2] Assistant Professor, ECE Department, ALTS , Anantapur

ABSTRACT: Decimal X × Y augmentation is a mind boggling activity, where moderate incomplete items (IPPs) are regularly chosen from an arrangement of precomputed radix-10 X products. A few works require just [0 , 5]× X by means of recoding digits of Y to one-hot portrayal of marked digits in [− 5 , 5] . This decreases the choice rationale at the cost of one additional IPP. Two's supplement marked digit (TCSD) encoding is frequently used to speak to IPPs, where dynamic refutation (by means of one xor per bit of X products) is required for the recoded digits of Y in [− 5 , − 1] .In this paper, regardless of age of 17 IPPs, for 16-digit operands, we figure out how to begin the fractional item lessening (PPR) with 16 IPPs that improve the VLSI consistency. In addition, we spare 75% of nullifying xors through speaking to precomputed products by sign-size marked digit (SMSD) encoding. For the principal level PPR, we devise a productive snake, with two SMSD input numbers, whose entirety is spoken to with TCSD encoding. From that point, staggered TCSD 2:1 diminishment prompts two TCSD amassed incomplete items, which by and large experience an uncommon early started change plan to get at the last parallel coded decimal item. All things considered, a VLSI execution of 16 × 16-digit parallel decimal multiplier is combined, where assessments demonstrate some execution change over past applicable outlines.

INTRODUCTION

Decimal math equipment is exceptionally requested for quick preparing of decimal information in money related, Web-based, and human intelligent applications. Quick radix-10 increase, specifically, can be accomplished by means of parallel halfway item age (PPG) and incomplete item decrease (PPR), which is, in any case, profoundly territory expending in VLSI usage. Along these lines, it is wanted to bring down the silicon cost, while keeping the rapid of parallel acknowledgment. Let P = X × Y speak to a n × n decimal augmentation, where multiplicand X , multiplier Y , and item P are ordinary radix-10 numbers with digits in [0 , 9]. Such digits are regularly spoken to by means of double coded decimal (BCD) encoding. In any case, middle halfway items (IPPs) are spoken to by means of a decent variety of frequently repetitive decimal digit sets and encodings (e.g., [0 , 10] convey spare (CS), [0 , 15] over-burden decimal, [− 7 , 7] marked digit (SD), twofold 4, 2, 2, 1, and [− 8 , 8] SD).

The decision of option IPP portrayals is compelling on the PPG, which is of specific significance in decimal duplication from two perspectives: one is quick and minimal effort age of IPPs and the other is its effect on portrayal of IPPs, which is powerful on PPR proficiency. Clear PPG by means of BCD digit-by-digit augmentation is moderate, costly, and prompts n twofold BCD IPPs for n × n increase (i.e., 2 n BCD numbers to be included). Be that as it may, crafted by [10] recodes both the multiplier and multiplicand to sign-size marked digit (SMSD) portrayal and utilizations a more proficient 3-b by 3-b PPG. In any case, following a long standing practice, most PPG plans utilize precomputed products of multiplicand X (or X products). Precomputation of the entire set0 , 1 ,... 9 }× X , as ordinary BCD numbers, and the consequent determination are likewise moderate and exorbitant. A typical therapeutic strategy is to utilize a littler less expensive set that can be accomplished through quick convey free control (e.g., 0 , 1 , 2 , 4 , 5 }× X) at the cost of multiplying the tally of BCD numbers to be included PPR; that is, n twofold BCD IPPs are created, for example, 3 X = (2 X , X) ,7 X = (5 X , 2 X) ,or9 X = (5 X , 4 X) . We offer an outline of PPG and PPR attributes of a few past pertinent works.

The recoding of multiplier's digits, in some important works, prompts a convey bit other than the n recoded digits of the multiplier, which will create an additional halfway item. This is especially tricky for parallel duplication with n = 16 (i.e., number of significand's decimal digits as per IEEE standard size of single exactness radix-10 gliding point numbers), where the 17 created incomplete items require five PPR levels rather than four (i.e., log 2 16) . Besides, they progressively refute positive products in light of the indication of multiplier's recoded digits. This system lessens the region and postponement of rationale that chooses the X products at the cost of restrictively nullifying the chose products, which requires no less than 4 n 2 XOR entryways for n × n augmentation. In this paper, we intend to exploit [− 5 , 5] SMSD recoding of multiplier and dynamic invalidation of X products, while decreasing the

quantity of XOR doors by means of creating [− 6 , 6] SMSD precomputed X products (i.e., only one XOR entryway for each 4-b digit). Different commitments of this paper are featured beneath. 1) Starting the PPR With 16 Partial Products: A particular on the fly growth of two center SMSD digits prompts lessening the profundity of halfway item framework by 1, to such an extent that the PPR begins with 16 operands comfortable end of PPG, with no defer punishment for the last mentioned.

TABLE I
SUMMARY OF PPG AND PPR CHARACTERISTICS

	Reference	MR	Pre-computed multiples	ME	DN	#OP	PPDS	PPDE
1	[11]	None	$\{0,1,\ldots,9\} \times X$	BCD	No	2n	[0, 9]	BCD
2	[3], [29], [21]		$\{0,1,2,4,5\} \times X$				[0, 10]	Double BCD
3	[22]		$\{0,1,2,5,6,9\} \times X$					
4	[2]		$\{0, \pm1, \pm2, 5, 10\} \times X$	4,2,2,1				Double 4,2,2,1
5	[6] Radix-5							
6	[10]	[−5, 5]	None	[−5, 5]	Yes	n + 1	[−6,6]	Svoboda
7	[6] Radix-10		$\{0,1,2,3,4,5\} \times X$	4,2,2,1			[0, 9]	4,2,2,1
8	[7]			[−8,0]			[−8,0]	−8,4,2,1,1
9	[4]			[−3,12] Excess-3			[0, 15]	8,4,2,1

2) Special 4-in-1 SMSD Adder With TCSD Sum: To maintain a strategic distance from the testing expansion of SMSD IPPs, we outline a novel convey free snake that speaks to the total of two [− 6 , 6] SMSD operands in [− 7 , 7] two's supplement marked digit (TCSD) design, where one brought together viper is used for all the four conceivable sign blends.

3) Improved TCSD Addition: whatever remains of the diminishment procedure utilizes exceptional TCSD adders that are really an enhanced variant of the quick TCSD snake. Such 2:1 decrease advances the VLSI normality of the PPR circuit, particularly for n = 16 (i.e., the suggested operand size of).

4) Augmenting the Final Redundant to Nonredundant Conversion With the Last PPR Level: The last PPR level would regularly prompt TCSD item, which ought to be changed over to BCD. Be that as it may, to acquire speed and diminish costs, we gadget an exceptional half and half decimal snake with two TCSD inputs and a BCD yield.

EXISTING SYSTEM

In this segment, we quickly consider a few past pertinent works by aggregating their PPG and PPR qualities. The section acronyms MR, ME, DN, #OP, PPDS, and PPDE remain for multiplier recoding, various encoding, dynamic invalidation of IPPs, number of operands to be included (i.e., number of initially created nonredundant decimal numbers, or SD fractional items), halfway item digit set, and incomplete item digit encoding, individually.

We alludes to the encoding of a digit d ∈[0 , 6] ([− 6 , − 0]) by a 5-bit twofold number 3 d (31 − 3 d) .

Some different chips away at decimal increase with coasting point operands, particular plans for Field Programmable Gate Array (FPGA), or digit-by-digit iterative approach are not recorded, since they depend on one of the organized works, or utilize implanted FPGA segments, which are out of the extent of this paper.

Some more points of interest for those reference works that have been apparently executed, for n = 16, are as per the following.

Successive multipliers, with quick convey free X- various age, where during the time spent fractional item gathering, the incomplete item digit set (PPDS) and halfway item digit encoding (PPDE) change from [0 , 18] to [0 , 10] and twofold BCD to BCD CS, individually. Parallel multiplier, with quick convey free X - various age, 4:1 and 2:1 multiplexing, where PPDS and PPDE will later change to [0 , 10] and BCD CS, individually. The quantity of decrease levels is 6. Two plans are offered in this work, where both utilize 3:2 decrease and particularly × 2 rectification cells.

Radix-10: Parallel multiplier, [−5, 5] SMSD recoding of multiplier, moderate convey proliferating 3 X age, 5:1 multiplexing with dynamic invalidation. The quantity of lessening levels is 6.

Consecutive multiplier, ease back PPG through BCD-to-[−5,5] SMSD recoding of multiplier's and multiplicand's digits, trailed by digit-by-digit increase prompting [− 6 , 6] PPDS with moderate halfway item amassing. Parallel multiplier, [− 5 , 5] SMSD recoding of multiplier, quick convey free X - various age through repetitive portrayal of products including 3 X , 5:1 multiplexing with dynamic refutation.

Diminishment is refined in two phases. One is 17:8 that incorporates three levels of CS viper and a 4-b snake. The other (i.e., 8:2) utilizes two levels of (4; 2) blowers and a 5-b snake. The two phases finish up with some amendment rationale. As above, aside from lessening, where IPPs are spoken to as [0, 15] radix-10 numbers. Parallel 4:2 and 3:2 diminishments are utilized with due decimal rectifications.

This nullification cost is reproduced n times for parallel n × n augmentation. In addition, the n embedded 1s for 10's complementation in and n × (n + 1) 1s for digit insightful two's complementation in negatively affect territory and power sparing. The same is valid for the rectification consistent, and more mind boggling recoding because of zero taking care of, for [0 , 15] fractional items. One approach to spare these cost, is to create the SD precomputed X

products with sign size arrangement, in order to diminish the XOR entryways to one for each digit (about 75% funds in the quantity of refuting XOR doors) and expel the previously mentioned negative effects.

Notwithstanding, other than backing off the PPG to some degree (e.g., in examination with radix-5 usage), new issues are presented in PPR, which are clarified and unraveled in the following segment, where we additionally decrease the profundity of IPP network to n = 16, adequately before end of PPG.

Drawbacks in existing system

- Existing like radix-10 BCD multiplier is slower in generating and selecting partial products.
- Hardware resource utilization is more.

PROPOSED SYSTEM

DECIMAL IPPS WITH SIGN-MAGNITUDE REPRESENTATION OF SIGNED DIGITS

Decimal SDs in [− α, α] (α ≤ 7) are typically encoded with insignificant 4-b marked numbers. For instance, consider α = 5 and α = 7 with sign size and two's supplement portrayals, separately. The last is reasonable for fundamental math activities, aside from refutation, which is best performed on sign extent organize. In this segment, we propose a decimal augmentation plot with the accompanying attributes, which are in an indistinguishable line from those of the outlines recorded in Table I:

1) [− 5, 5] SMSD recoding of multiplier's digits;

2){ 0 , 1 , 2 , 3 , 4 , 5 }× X precomputed products;

3) 4-b [− 6 , 6] SMSD encoding of precomputed products;

4) Dynamic invalidation of products with just a single XOR per digit (i.e., per 4 b);

5) Instead of (n + 1) operands to be included for n × n augmentation;

6) Unified SMSD + SMSD −→ TCSD viper for every one of the four information sign blends;

7) [− 7 , 7] TCSD portrayal for collected incomplete items;

8) Early beginning of excess to BCD transformation;

9) Augmenting last PPR level With definite transformation to BCD.

Fig.3.1 portrays the general engineering of the proposed 16 × 16 augmentation P = X × Y ; the points of interest of each building square will be clarified later. Specifically, in the main three obstructs, the multiplier's digits are recoded to n one-hot [− 5 , 5] SMSDs (i.e., one sign and five extent bits), increased with a 10 n - weighted convey bit. The products [0 , 5]× X are precomputed as n [− 6 , 6] SMSDs and a 10 n - weighted [− 5 , 4] SMSD. Each SMSD contains a sign piece s and 3-b size. The negative products [1 , 5]× (− X) are accomplished by means of dynamic sign reversal of products [1 , 5]× X at the cost of just a single XOR entryway for every digit.

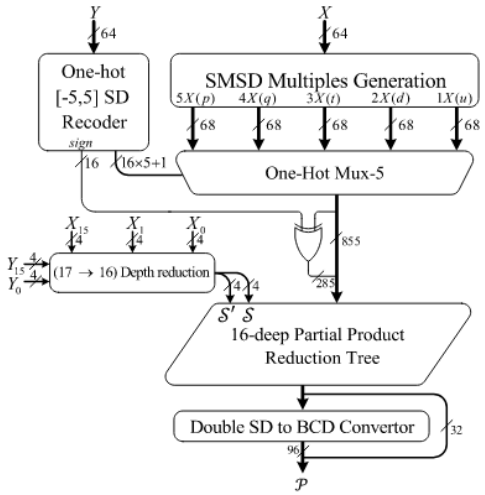


Fig. 3.1: Block diagram of the proposed multiplier.

Recoding of Multiplier's Digits

Unique BCD digits of multiplier require [0 , 9]× X precomputed products, which incorporate hard products { 3 , 6 , 7 , 9 }× X that dissimilar to { 2 , 4 , 5 , 8 }× X are not resultant without convey engendering. Then again, BCD-to-excess [− 5 , 5] SMSD recoding of multiplier's digits with dynamic refutation of IPPs decreases the required X products to [0 , 5]× X that incorporate just a single hard various (i.e., 3 X) .

Be that as it may, this recoding produces a convey as the (n + 1) th digit of multiplier, which builds the quantity of IPPs by 1. This is particularly not attractive for n = 16 (i.e., the prescribed IEEE754-2008 word measure for decimal operands). The reason is that it might expand the quantity of 2:1 PPR levels by 1, which can be maintained a strategic distance from as will be managed in Section III-C.

The one-hot recoding input/yield articulations are given by (1), where Y I = v 3 v 2 v 1 v 0 ,and Y I − 1 = w 3 w 2 w 1 w 0 speak to two successive digits of BCD multiplier, ω shows whether Y I − 1 ≥ 5, s v is the indication of target code, and v 1 – v 5 are one-hot signs relating to outright estimations of recoded multiplier's digit Y I (i.e., 1– 5), whose decimal weight is equivalent to that of Y I .

$$\omega = w_3 \vee w_2(w_1 \vee w_0)$$
$$v_1' = \overline{v_2 \vee v_1}(\omega \oplus v_0)$$
$$v_2' = \omega v_0(\overline{v_3 \vee v_2 \vee v_1} \vee v_2 v_1) \vee \overline{\omega \vee v_0}(v_3 \vee \bar{v}_2 v_1)$$

Decimal position		$i+1$	i	$i-1$
X	$[0,9]$	X_{i+1}	X_i	X_{i-1}
$3X$	$[0,9]$		L_i	L_{i-1}
	$[0,2]$	H_i	H_{i-1}	
$3X$	$[-6,3]$		L_i'	L_{i-1}'
	$[0,3]$	H_i'	H_{i-1}'	
$3X$	$[-6,6]$		T_i	

Fig. 3.2: Two consecutive digits of X ,3 X (BCD), and 3 X ([− 6 , 6] SMSD)

$$v_3' = v_1(\omega \oplus v_0)$$
$$v_4' = \overline{\omega \vee v_0} v_2 \vee \omega v_0(v_2 \oplus v_1)$$
$$v_5' = v_2 \bar{v}_1(\omega \oplus v_0)$$
$$s_{v'} = v_3 \overline{\omega v_0} \vee v_2(v_1 \vee v_0).$$

It is anything but difficult to confirm that comparable recoding can be connected to different products to be spoken to with a similar digit set ([− 6 , 6] SMSD). The relating sensible expr essions for SMSD products of the multiplicand X (i.e., 1 X (u) ,2 X (d) ,3 X (t) ,4 X (q) , and 5 X (p)) can be determined regarding bits of BCD digits an (in position I) and b (in position I − 1) . For instance, the coherent articulation of 3 X (t) is given by (3), and the rest can be found in the Appendix.

Note that such products are spoken to with at most one additional digit (i.e., aggregate of 68 bits), since the most noteworthy digit of the created different is at most 4 (because of 5 × 9 = 45) , which remains 4 inside the BCD-to-SMSD transformation.

$$s_t = \bar{a}_3\bar{a}_2(\bar{a}_1\bar{b}_2 b_1 \vee b_1\bar{b}_0) \vee b_1\bar{b}_0(\bar{b}_2 \vee \bar{a}_3\bar{a}_1\bar{a}_0) \vee b_3(\bar{a}_3 \vee \bar{b}_0) \vee b_2\bar{b}_1 b_0$$

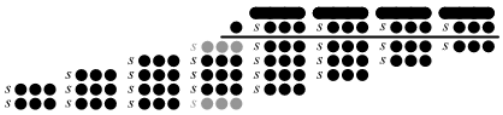


Fig 3.3: Normal organization of IPPs.

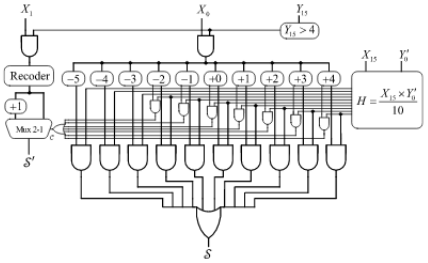
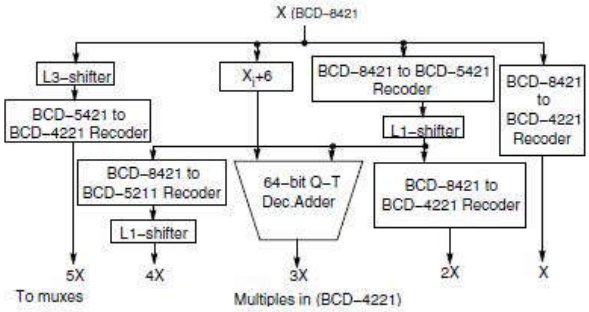


Fig. 3.4: Required circuit for (17 → 16) depth reduction

$$t_2 = \bar{b}_1 b_0(\bar{b}_3\bar{b}_2(a_3 \vee a_2) \vee \bar{a}_3 b_2(\bar{a}_2 \vee \bar{a}_1\bar{a}_0) \vee \bar{a}_2 a_1 \bar{b}_3) \vee b_2\bar{b}_1\bar{b}_0(a_3 \vee a_2 a_1 \vee a_2 a_0) \vee \bar{a}_3\bar{b}_0(b_3 \vee \bar{a}_2\bar{a}_1\bar{b}_2 b_1) a_3 b_2 b_1 b_0$$
$$t_1 = \bar{b}_2\bar{b}_1(a_2\bar{b}_3(a_1 \vee a_0) \vee b_1(a_2 \vee a_1) \vee a_3\bar{b}_1) \vee \bar{a}_3\left(\begin{matrix}\bar{a}_2(b_3 b_0 \vee b_2\bar{b}_1\bar{b}_0) \vee \bar{a}_2\bar{a}_1(b_2\bar{b}_0 \vee \bar{b}_2\bar{b}_1 b_0) \vee \\ \bar{a}_1\bar{a}_0(b_3 b_0 \vee b_2\bar{b}_1\bar{b}_0)\end{matrix}\right) \vee b_2 b_0(a_2(a_1 \vee a_0) \vee b_1(a_2 \vee a_1)) \vee a_3\bar{b}_3 b_0(\bar{b}_2 \vee \bar{b}_1) \vee \bar{a}_2\bar{a}_1 b_3\bar{b}_0$$
$$t_0 = b_0(a_2(a_1 \vee a_0) \vee \bar{a}_3\bar{a}_2\bar{a}_1) \vee \bar{b}_0(a_3 \vee \bar{a}_2 a_1 \vee a_2\bar{a}_1\bar{a}_0). \quad (3)$$

Partial Product Generation

All the required decimal multiplicand products, aside from the 3X various, are gotten in a couple of levels of combinational rationale utilizing distinctive digit recoders and performing diverse settled m-bit left moves (Lmshift) in the bit-vector portrayal of operands. The structure of these digit recoderS. Fig. 3 demonstrates the square graph for the age of the positive multiplicand products {X, 2X, 3X, 4X, 5X} for the SD radix-10 recoding.



Every one of these products are coded in (4221). The X BCD multiplicand is effectively recoded to (4221) utilizing the sensible articulations. where xi,j and wi,j are information and yield, separately,

the bits of the BCD and (4221) portrayals of X. The age of products is as per the following:

Numerous 2X. Each BCD digit is first recoded to the (5421) decimal coding appeared in Table 1 (the

mapping is one of a kind). A L1shift is performed to the recoded multiplicand, acquiring the 2X different in BCD. At that point, the 2X BCD different is recoded to (4221) utilizing Expressions (2).

Various 4X. It is acquired as 2X × 2, where the 2X different is coded in (4221). The second ×2 task is actualized as a digit recoding from (4221) to code (5211), trailed by a L1shift. The plan of the (4221) to (5211) digit recoders. The ×2 task, with input operands coded in (4221) or (5211), is likewise executed in the decimal CSA trees utilized for fractional item decrease.

Numerous 5X. It is acquired by a basic L3shift of the (4221) recoded multiplicand, with resultant digits coded in (5211). At that point, a digit recoding from (5211) to (4221) is performed (see Section 4.3). Fig. 4 demonstrates a case of this activity.

Different 3X. It is assessed by a convey engender expansion of BCD products X and 2X of every a d-digit BCD snake. The BCD aggregate digits are recoded to (4221) as demonstrated by Expression (2). The inertness of the fractional item age for the SD radix-10 conspire is compelled by the age of 3X.

Fig. 3.3 portrays the PPG, and the typical association of IPPs of such n × (n + 1) duplication for n = 4. The tick bars speak to BCD digits of the multiplicand. Profundity of the most profound segment of IPP lattice (i.e., 10 n - weighted position) is (n + 1) , where all digits have a place with [− 6 , 6] , aside from the best and base ones (in dark) that have a place with [− 5 , 4] and [− 6 , 3] , separately.

We lessen the grid profundity to n (e.g., 5 → 4for n = 4, and 17 → 16 for n = 16) , with no postponement between the end of PPG and beginning of PPR. Here is the manner by which it works: we process entirety of the two dark digits (see Fig.3.3) free of (and in parallel with) typical PPG as takes after. In the event that Y n − 1 ≤ 4, the estimation of 10 n - weighted convey of recoded multiplier is zero, so the base dim digit must be zero.

Along these lines, no expansion is required. For Y n − 1 > 4, let H signify the most noteworthy digit of X n − 1 × Y 0 (e.g., the best dim digit in Fig. 3), where X n − 1 and Y 0 speak to the most critical BCD digit of multiplicand and the minimum huge recoded digit of multiplier, separately. We extricate H as ten one-hot signs by means of an eight-input rationale (see the furthest right box in Fig. 3.4).

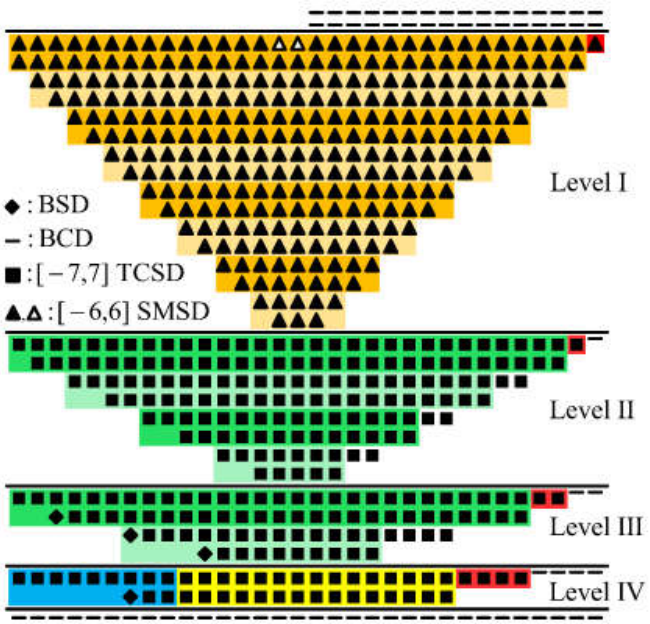


Fig. 3.5: Overall view of 16 × 16 digit multiplier.

This prompts the coveted entirety digit S in 10 n - weighted position (instead of two dim digits of Fig. 3.3) and a convey bit c to be added to the 10 n + 1 - weighted digit beside the base dim digit to bring

about S (S and S are likewise recognized by white triangles in Fig. 3.5).

This digit, as appeared in the furthest left piece of Fig. 3.4, is acquired by straightforwardly recoding the 10-weighted digit of multiplicand (i.e., X 1).

Partial Product Reduction (PPR)

The general PPR for n = 16 is represented by Fig. 3.5, where a bar, triangle, square, and precious stone speak to a BCD, [− 6 , 6] SMSD, [− 7 , 7] TCSD, and paired marked digit (BSD), separately. The decision of SMSD portrayal for the primary level IPPs, while encouraging the PPG, bears no additional many-sided quality for PPR, since all lessening levels utilize TCSD adders, with the exception of the first that requires an extraordinary SMSD+SMSD-to-TCSD viper. Nonetheless, as will be appeared, this snake isn't more mind boggling than a straightforward TCSD viper.

SMSD Adder: A digit cut of the in advance of said SMSD+SMSD-to-TCSD snake for four unique cases comparing to every single conceivable blend of the information signs is portrayed by Fig. 3.6(a)– (d) in speck documentation portrayal. The highly contrasting specks speak to posibits and negabits. (A posibit is an ordinary piece whose number juggling esteem squares with its consistent status, and the number-crunching estimation of a negabit with coherent status x measures up to x − 1) .

In the stage I, the sign bits are connected to the sizes, to such an extent that a negative sign changes the extremity of size posibits to negabits and alters their intelligent states. In this way, in a similar stage, the bit gathering U is decayed, and the bit accumulation V is recoded. In the second stage, notwithstanding, as will be clarified in the blink of an eye, just a single 4-b viper deals with all the four cases, which clarifies the reason for assignment of the snake.

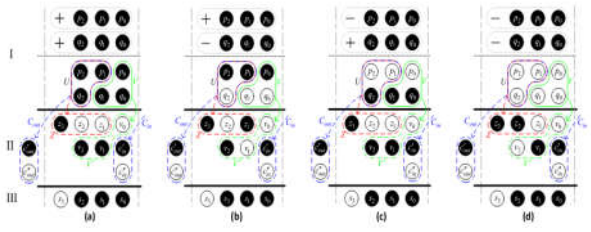


Fig. 3.6: Digit slice of SMSD+ SMSD-to-TCSD adder for four sign combinations.

(a) $s_p = +, s_q = +$. (b) $s_p = +, s_q = -$. (c) $s_p = -, s_q = +$ (d) $s_p = -, s_q = -$.

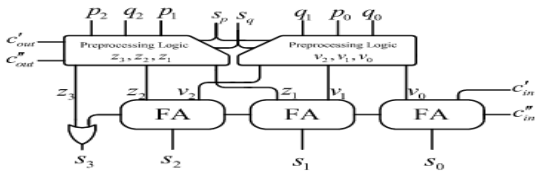


Fig. 3.7: Digit slice of the 4-in-1 SMSD+SMSD → TCSD adder.

The (c in , c in) combine speaks to the approaching marked convey C in from the less noteworthy position. Portrayals of Z , V , and C in are so decided as to prompt two's supplement portrayal for S in all the four cases (see beneath for more clarifications, and the accompanying numerical case)

Case 1: (Fig. 3.6 by numerical qualities): Fig. 8 portrays a numerical case, where two SMSDs P = s p 101 (| P |= 5) and Q = s q 100 (| Q |= 4) are included. Fig. 8 copies Fig. 6 with numerical qualities, where signs (i.e., s p and s q) are expressly appeared just like the case in Fig. 6, and negabits are conversely encoded as 1 − (0 −) , which speak to the number juggling esteem 0 (− 1) . The approaching marked convey C in = 0 is spoken to by the posibit c in = 0 and conversely encoded negabit c in = 1 − .

Accordingly, the Full Adder in position 0 gets two negabits and one posibit and produces a posibit aggregate 1 and a negabit convey 0 − , with the end goal that 2 × (− 1) + 1 =− 1, as there was just a single mathematically nonzero input 0 − (i.e., − 1) .

The 4-in-1 viper is somewhat more productive than [− 7 , 7] TCSD snake (i.e., less dormancy with no territory overhead), as can be confirmed by investigating (4) and (5) for the preprocessing rationale confines 4-in-1 snake and that of TCSD viper.

2) TCSD Adder: The TCSD snake, which is required for the rest of the (log 2 n − 2) ensuing diminishment levels. The required engineering is the same as in Fig. 3.7, aside from the preprocessing boxes, where the required legitimate articulations are portrayed in (6). Likewise, Fig. 3.9 delineates one digit cut of this snake

$$c'_{out} = \overline{p}_3\overline{q}_3, \quad c''_{out} = p_3q_3\overline{p_2q_2p_1} \vee \overline{p}_2\overline{q}_2(p_1 \vee \overline{p}_3\overline{q}_3)$$
$$v_0 = \overline{p_0 \oplus q_0}, \quad v_1 = q_1 \oplus (p_0 \vee q_0), \quad v_2 = q_1(p_0 \vee q_0)$$
$$z_1 = (p_2 \vee q_2)(p_3\overline{q}_3\overline{p}_1 \vee \overline{p}_3(q_3 \oplus p_1)) \vee \overline{p}_3\overline{q}_3\overline{p}_2\overline{q}_2\overline{p}_1 \vee p_3q_3p_1\overline{p_2q_2}$$

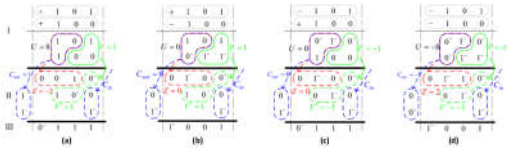


Fig. 8. Numerical example with $|p| = 5$ and $|q| = 4$. (a) $s_P = +, s_Q = +$. (b) $s_P = +, s_Q = -$. (c) $s_P = -, s_Q = +$. (d) $s_P = -, s_Q = -$.

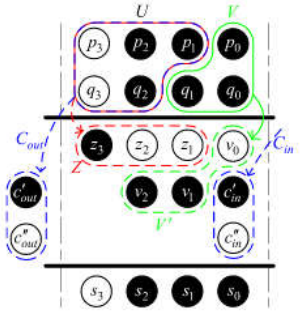


Fig. 9: TCSD adder.

$$z_2 = (p_2 \oplus q_2)(p_3(q_3 \vee p_1) \vee q_3p_1) \vee p_2q_2\overline{p}_1\overline{p}_3\overline{q}_3 \vee \overline{p}_3\overline{q}_3(p_2q_2 \vee \overline{p}_2\overline{q}_2\overline{p}_1)$$
$$z_3 = \overline{p}_3\overline{q}_3\overline{p}_2\overline{q}_2p_1 \vee p_3q_3p_2q_2\overline{p}_1 \vee (p_3 \oplus q_3)(p_2q_2p_1 \vee \overline{p}_2\overline{q}_2\overline{p}_1). \quad (6)$$

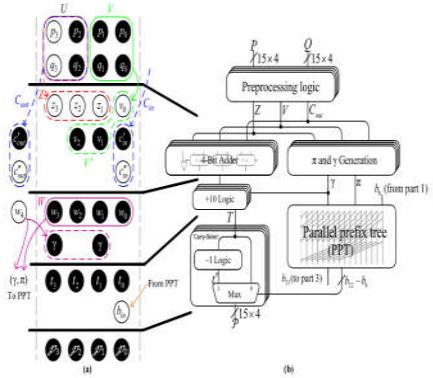


Fig. 3.10: Final conversion in part 2. (a) Digit slice of TCSD+TCSD-to- BCD adder. (b) Architecture for fin al product generation in positions 8–22.

The − 6 w 4 task can be supplanted by + 10 w 4 .However, on account of w 4 = 0, a decimal obtain is persisted to the more critical decimal position that causes acquire proliferation. To evade such moderate acquire spread, we utilize a parallel prefix obtain generator that utilizations decimal get proliferate and create signals π = (W = 0) and γ = (W < 0) = w 4 , separately. These obtain signals are produced through a four-level Kogge– Stone (KS) [26] parallel prefix connect with 15 input sets (π , γ) , and acquire in b 8 from section 1 (i.e., out of position 7)

Advantages of proposed system

- Partial product generation and selection circuitry is reduced.
- We are implementing multiplier for negative BCD numbers efficiently.
- Less hardware resource utilization.
- High speed BCD multiplier for signed opereands.

CONCLUSION

We propose a parallel 16 × 16 radix-10 BCD multiplier, where 17 fractional items are produced with [−6, 6] SMSD portrayal. A few advancements of this paper and utilization of past systems, as recorded underneath, have prompted negligible 1.5% less territory utilization and 10% less power scattering, at 4.8-ns inertness, as for the quickest past work [4]. The minimum conceivable deferral for the last is 4.8 ns, while the proposed configuration drives the blend instrument to meet the 4.4-ns time requirement (i.e., 9% quicker). As it were, the favorable position is that the proposed configuration can work in 9% higher recurrence and disseminate up to 13% less power with no claim in zone change.

REFERENCES

[1] M. F. Cowlishaw, “Decimal floating-point: Algorism for computers,” in Proc. 16th IEEE Symp. Comput. Arithmetic, Jun. 2003, pp. 104–111.

[2] T. Lang and A. Nannarelli, “A radix-10 combinational multiplier,” in Proc. 40th Asilomar Conf. Signals, Syst., Comput., Oct./Nov. 2006, pp. 313–317.

[3] R. D. Kenney, M. J. Schulte, and M. A. Erle, "A high-frequency decimal multiplier," in Proc. IEEE Int. Conf. Comput. Design (ICCD), Oct. 2004, pp. 26–29.

[4] A. Vazquez, E. Antelo, and J. D. Bruguera, "Fast radix-10 multiplication using redundant BCD codes," IEEE Trans. Comput., vol. 63, no. 8, pp. 1902–1914, Aug. 2014.

[5] S. Gorgin and G. Jaberipur, "Fully redundant decimal arithmetic," in Proc. 19th IEEE Symp. Comput. Arithmetic, Jun. 2009, pp. 145–152.

[6] A. Vazquez, E. Antelo, and P. Montuschi, "Improved design of highperformance parallel decimal multipliers," IEEE Trans. Comput., vol. 59, no. 5, pp. 679–693, May 2010.

[7] L. Han and S.-B. Ko, "High-speed parallel decimal multiplication with redundant internal encodings," IEEE Trans. Comput., vol. 62, no. 5, pp. 956–968, May 2013, doi: 10.1109/TC.2012.35.

[8] G. Jaberipur and A. Kaivani, "Binary-coded decimal digit multipliers," IET Comput. Digit. Techn., vol. 1, no. 4, pp. 377–381, 2007.

[9] R. K. James, T. K. Shahana, K. P. Jacob, and S. Sasi, "Decimal multiplication using compact BCD multiplier," in Proc. Int. Conf. Electron. Design, 2008, pp. 1–6.

[10] M. A. Erle, E. M. Schwarz, and M. J. Schulte, "Decimal multiplication with efficient partial product generation," in Proc. 17th IEEE Symp. Comput. Arithmetic, Jun. 2005, pp. 21–28.

[11] R. K. Richards, Arithmetic Operations in Digital Computers. New York, NY, USA: Van Nostrand, 1955.

[12] IEEE Standard for Floating-Point Arithmetic, IEEE Standard 754-2008, IEEE Standards Committee, 2008, doi: 10.1109/IEEESTD. 2008.4610935.

[13] A. Svoboda, "Decimal adder with signed digit arithmetic," IEEE Trans. Comput., vol. C-18, no. 3, pp. 212–215, Mar. 1969.

[14] B. J. Hickmann, A. Krioukov, M. Schulte, and M. Erle, "A parallel IEEE P754 decimal floating-point multiplier," in Proc. IEEE 25th Int. Conf. Comput. Design (ICCD), Oct. 2007, pp. 296–303.

[15] R. Raafat et al., "A decimal fully parallel and pipelined floating point multiplier," in Proc. 42th Asilomar Conf. Signals, Syst., Comput., Oct. 2008, pp. 1800–1804.

[16] M. A. Erle, B. J. Hickmann, and M. J. Schulte, "Decimal floatingpoint multiplication," IEEE Trans. Comput., vol. 58, no. 7, pp. 902–916, Jul. 2009.

[17] C. Tsen, S. González-Navarro, M. Schulte, B. J. Hickmann, and K. Compton, "A combined decimal and binary floating-point multiplier," in Proc. 20th IEEE Int. Conf. Appl.-Specific Syst., Archit. Process. (ASAP), Jul. 2009, pp. 8–15.

[18] C. Minchola and G. Sutter, "A FPGA IEEE-754-2008 decimal64 floating-point multiplier," in Proc. Int. Conf. Reconfigurable Comput. FPGAs (ReConFig), Dec. 2009, pp. 59–64.

[19] A. Vázquez and F. de Dinechin, "Efficient implementation of parallel BCD multiplication in LUT-6 FPGAs," in Proc. Int. Conf. FieldProgram. Technol. (FPT), Dec. 2010, pp. 126–133.

[20] M. A. Erle and M. J. Schulte, "Decimal multiplication via carrysave addition," in Proc. IEEE Int. Conf. Appl.-Specific Syst., Archit., Process. (ASAP), Jun. 2003, pp. 348–358.

[21] S. Gorgin and G. Jaberipur, "A fully redundant decimal adder and its application in parallel decimal multipliers," Microelectron. J., vol. 40, no. 10, pp. 1471–1481, Oct. 2009.