Area and power efficient decimal carry-free adder

Liu Han, Hao Zhang and Seok-Bum Ko✉

As decimal floating-point (DFP) is better than binary floating-point in commercial and financial computing including billing systems, currency conversion, tax calculation and banking, many research activities have been focused on improving the performance of the DFP arithmetic unit recently. To achieve the high performance of the DFP arithmetic unit, a fast decimal fixed-point adder is the most important building block. The conventional three steps carry-free signed digit (SD) addition algorithm is first investigated. A new method for the decimal SD addition and subtraction based on the digit set [−9, 9] is proposed. Additionally, a digit-set converter which can directly generate the absolute value of the result is proposed. A model of the proposed decimal SD adder is implemented in VHDL. After exhaustive tests to ensure the correctness, the proposed design was synthesised in STM 90 nm technology. The results show that the proposed adder has a lower power and area consumption compared with previous designs.

Introduction: In the last decade, decimal floating-point (DFP) arithmetic has become more attractive in the financial and commercial computing sector which includes billing/banking systems, currency conversion and tax calculation. The reason is because most of the finite DFP numbers would need an infinitely recurring binary fraction (e.g. the decimal number 0.1). A detailed discussion about decimal and binary floating point arithmetic can be found in [1]. Owing to the importance seen in decimal arithmetic, in 2008, DFP format and operations were included in the latest version of the IEEE standard for floating-point arithmetic (IEEE 754-2008) [2].

In fixed-point decimal arithmetic, addition is the most important since it is the basis of all other operations. In [3], the authors showed that the logics related to the significand computation units occupy 51% timing delay and 41% area of the DFP adder. As an improvement of addition can benefit directly other decimal operations, many methods and algorithms have been proposed to enhance the performance of the decimal adder.

The signed digit (SD) number system could be applied to remove the carry chain in the carry-free addition. After the first decimal SD adder was published in 1969 [4], several papers have been presented in the last decade. In [5], authors provided a decimal SD adder in the digit set [−9, 9]. In [6], the authors contributed on fully redundant decimal addition based on stored unibit transfer encoding. In [7], a redundant binary coded decimal adder (RBCD) is proposed.

In this Letter, the conventional decimal carry-free algorithm and the possible improvement to make it more area and power efficient are discussed. Subsequently, a two-level decimal SD adder which performs addition and subtraction in digit set [−9, 9] is proposed. Since in DFP arithmetic, the significand represented in BCD (binary coded decimal) format is a non-negative number, we propose a digit-set conversion algorithm which calculates the absolute value of the SD result and converts it to the conventional encoding.

Proposed carry-free SD addition: In the traditional carry-free addition algorithm, in order to obtain the transfer digit T_{i+1}, the operands X_i and Y_i need to be added together and compared with the threshold value. The performance of the carry-free adder is limited by a carry chain in this process. A speculative method could be applied (e.g. [5, 8, 9]) to improve the performance. In this speculative method, all possible results which depend on different transfer digits T_i and T_{i+1} can be computed simultaneously and the correct one is chosen by the value of the transfer digits. The redundancy from hardware perspective in the aforementioned designs results in the bigger area and higher power consumption. However, the non-speculative method for maximally redundant SD addition (e.g. [10]) provided a faster design in binary world (i.e. radix -2^h).

In this Section, we propose a new non-speculative decimal SD addition which can directly calculate the result without the hardware redundancy. The proposed adder which works in digit set [−9, 9] has a simple range division logic. Moreover, the operands and results are encoded in 5-bit two's complement to reuse the binary circuit as much as possible.

In the conventional carry-free algorithm, the transfer digit T_{i+1} and the temporary sum W_i are generated from the position sum P_i. To parallelise the transfer digit generation with the position sum calculation, the temporary sum W_i and transfer digit T_{i+1} also can be directly expressed in terms of P_i and Y_{op_i}, where $Y_{\mathrm{op}_i} = Y_i$ for add operation and $Y_{\mathrm{op}_i} = -Y_i$ for subtraction.

In the decimal SD number system, ± 9 should be avoided in temporary sum, otherwise, an incoming transfer digit could generate a carry to the next digit. The position sum, which is equal to ± 9, is called an exception in the proposed design. Furthermore, detection of the exception will pull down the performance of the decimal SD adder compared with its binary counterpart. Hence, the fewer exception cases, the better. Besides the exception detecting logic, only the most significant bits for X_i and Y_{op_i} and simple logic are needed to determine the transfer digit.

Once the transfer digit is obtained, W_i is generated by adding a correction value to P_i, and then the result S_i is computed by performing addition of W_i with T_i. In this process, the speed is limited by these two serial computations. To generate the correction signal, the transfer digit T_i from the last digit is used. Therefore, the decimal correction signal can be decided directly, and then the further computation for adding T_i is removed.

To reuse the well-optimised circuits in binary world as much as possible, the input operands are encoded in 2's complement. Therefore, Y_{op_i} is obtained by inverting Y_i with *XOR* gates controlled by the operation signal *op* which is the penalty of the subtraction. The increment on the least significant bit could be added as an incoming carry to the right most full adder.

In this proposed design, the exception handling logic directly uses the input operands and is simplified to only four pairs of operands detection which improves the implementation speed. To further improve the performance of the proposed design, compared with our previous work in [11], a new carry chain as shown in Fig. 1 is applied to the second-stage full adders. This high-speed carry logic allows for more efficient area and power optimisations and, therefore, a lower area and power consumption are achieved.

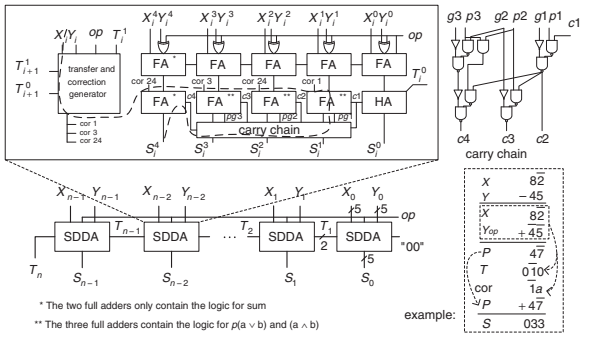

Fig. 1 *Proposed signed-digit decimal adder (n-digit)*

The hardware implementation of the proposed decimal SD adder is given in Fig. 1. The bold dash line is the critical path which passes through the transfer and correction logic and an optimised carry chain. The full adders with the asterisk only contain the logics for sum. Furthermore, in the second level of full adders, the critical path only pass through one *XOR* gate in the left most full adder.

Absolute value digit-set conversion: In the proposed design, the 5-bit 2's complement encoding format is used internally. However, in memory system, the BCD format is commonly used to encode the decimal data. As a result, the operands from the memory must be converted to the internally used format before being sent to the input ports of the proposed decimal adder. In a similar way, after the calculation, the result should be converted again to BCD format for the memory.

In the IEEE 754-2008 standard, the absolute value of the mantissa is represented in the significant digits section. However, for the SD subtraction, the result could be less than zero. Hence, before sending to the memory, the result which is less than zero must be converted to its absolute value.

In [3], a negation unit and prefix network are applied to make sure the calculation result is in BCD format. In [7, 6], to convert the BCD encoding to the internal encoding, nine-level and one-level of gates are used, respectively. Further, the authors proposed two algorithms to convert from internal encoding to BCD encoding with a carry propagation

chain. The negation unit for the redundant number system can be implemented digit by digit.

In this Letter, a merged algorithm is proposed which can directly convert the negative result to its absolute value in BCD encoding with a low penalty on delay.

In our design, since the digit set is encoded in 5-bit 2's complement, and the input operands in BCD encoding are always in digit set [0, 9], the front conversion which is only a 1-bit sign extension does not cost any logic. For the backward conversion, a borrow carry which is negative may be propagated through the whole conversion logic. For the negative SD result, before converting to the BCD encoding, the absolute value of it is obtained by inverting all signs on each digit.

To merge the negation with the digit-set conversion and improve the performance of the converter, we propose an absolute value digit-set conversion method which includes a prefix network and a correction unit. It leads to a logarithmical timing delay which is more suitable for high precision computation. The architecture of the proposed convertor for a p-digit input is given in Fig. 2.

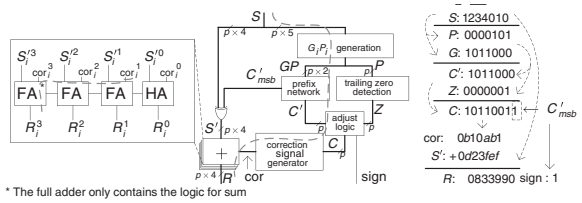

Fig. 2 *Proposed absolute value digit-set converter*

Results and discussions: The proposed design is implemented using VHDL. An exhaustive verification is done to make sure the design functions correctly. Thereafter, the proposed design is synthesised using Synopsys Design Compiler with STM 90 nm technology with normal case parameters.

To compare with other designs, we also implemented the designs in [5–7, 12], and our previous work [11] in the same technology. The implementation results, including timing delay, hardware area and power consumption are listed in Table 1. As this Letter is focused on low area and power cost, as shown in Table 1, it provides the minimum area and power cost while the latency is minimum.

Table 1: Evaluation results and comparisons (16-digit adders)

	Delay (ns)	Ratio	Area (μm^2)	Ratio	Power (mW)	Ratio
Proposed	0.28	1	11,781	1	3.53	1
[11]	0.28	1	14,581	1.23	4.68	1.32
[6]	0.39	1.39	12,746	1.08	7.33	2.07
[7]	0.45	1.60	11,907	1.01	5.71	1.61
[5]	0.49	1.75	39,913	3.38	27.46	7.78
[12]	0.51	1.82	25,192	2.13	15.09	4.27

Since the proposed adder is designed for the digit set [−9, 9], and 2's complement encoding format is used for operands, there is no need to add a forward conversion logic. In [6], the authors use the digit set [−8, 9], so there is an *OR* gate in the forward conversion logic. In [7], a combinational logic to generate the correction signal and a 4-bit adder are proposed to convert BCD to RBCD encoding.

For the backward converter which converts the digit format in the proposed design to the conventional BCD encoding, in [6, 7], the authors proposed two algorithms processed in linear timing delay which is proportional to the digit width of the input. Furthermore, to generate the absolute value, the aforementioned designs need more logics to check the sign of the result and invert the result digit by digit. The proposed converter in this Letter could generate the absolute value of the result in BCD encoding with a timing delay logarithmically proportional to the digit width.

Conclusion: In this Letter, a new area and power efficient non-speculative decimal SD adder is proposed which calculates the operands in digit set [−9, 9] with 2's complement encoding. This design determines the transfer digit directly on the input operands instead of using the position sum P_i as in the traditional carry-free addition algorithm in SD number system. Also the digit range of the operands is analysed and the range division method with the least cost of exception handling is used. Moreover, to further reduce the area of the proposed adder, a new circuitry which calculates the result digit S_i without the temporary result W_i is proposed. The synthesised results demonstrate the superiority of the proposed design in terms of area and power.

The proposed digit-set conversion architecture could directly convert the absolute value of the SD result to the BCD encoding. The new converter which has a timing delay logarithmically proportional to the digit width is more suitable for high precision computation.

Acknowledgment: This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

Submitted: *20 March 2015* E-first: *16 October 2015*
doi: 10.1049/el.2015.0786

Liu Han, Hao Zhang and Seok-Bum Ko (*Department of Electrical and Computer Engineering, University of Saskatchewan, Saskatoon, Saskatchewan, Canada*)

✉ E-mail: seokbum.ko@usask.ca

References

1 Cowlishaw, M.: 'Decimal floating-point algorism for computers'. Proc. of the 16th IEEE Symp. on Computer Arithmetic, June 2003, pp. 104–111
2 'IEEE Standard for Floating-Point Arithmetic', IEEE Working Group of the Microprocessor Standards Subcommittee, IEEE, 2008
3 Wang, L.-K., Schulte, M.J., Thompson, J.D., *et al.*: 'Hardware designs for decimal floating-point addition and related operations', *IEEE Trans. Comput.*, 2009, **58**, (3), pp. 322–335
4 Svoboda, A.: 'Decimal adder with signed digit arithmetic', *IEEE Trans. Comput.*, 1969, **c-18**, (3), pp. 212–215
5 Nikmehr, H., Phillips, B., and Lim, C.C.: 'A decimal carry-free adder'. Proc. of SPIE Conf. on Smart Materials, Nano-, Micro-Smart Systems, 2004, pp. 786–797
6 Kaivani, A., and Jaberipur, G.: 'Fully redundant decimal addition and subtraction using stored-unibit encoding', *Integr. VLSI J.*, 2009, pp. 34–41, doi: 10.1016/j.vlsi.2009.04.001
7 Shirazi, B., Yun, D., and Zhang, C.N.: 'RBCD: redundant binary coded decimal adder', *IEE Proc. E, Comput. Digit. Tech.* , 1989, **136**, (2), pp. 156–160
8 Fahmy, H., and Flynn, M.J.: 'The case for a redundant format in floating-point arithmetic'. Proc. of the 16th IEEE Symp. on Computer Arithmetic, June 2003, pp. 95–102
9 Jaberipur, G., and Ghodsi, M.: 'High radix signed digit number systems: representation paradigms', *Scientia Iranica*, 2003, **10**, pp. 383–391
10 Jaberipur, G., and Gorgin, S.: 'A nonspeculative maximally redundant signed digit adder'. The 13th Int. CSI Computer Conf., 2008, pp. 235–242
11 Han, L., Chen, D., Wahid, K.A., *et al.*: 'Nonspeculative decimal signed digit adder'. Proc. of IEEE Int. Symp. on Circuits and Systems (ISCAS), May 2011, pp. 1053–1056
12 Moskal, J., Oruklu, E., and Saniie, J.: 'Design and synthesis of a carry-free signed-digit decimal adder'. IEEE Int. Symp. on Circuits and Systems, 2007, pp. 1089–1092