Computer Applications in Electrical Engineering
Vol. 12 *2014*

On simplification of residue scaling process in pipelined Radix-4 MQRNS FFT processor

Robert Smyk, Maciej Czyżak
Gdańsk University of Technology
80-233 Gdańsk, ul. G. Narutowicza 11/12, e-mail: mczyzak@ely.pg.gda.pl

Residue scaling is needed in pipelined *FFT* radix-4 processors based on the Modified Quadratic Residue Number System (MQRNS) at the output of each butterfly. Such processor uses serial connection of radix-4 butterflies. Each butterfly comprises *n* subunits, one for each modulus of the *RNS* base and generates four complex residue numbers. In order to prevent arithmetic overflow intermediate results after each butterfly have to be scaled, *i.e.* divided by a certain constant. The number range of the processed signal increases due to transformation of coefficients of the FFT algorithm to integers and summation and multiplication within the butterfly. The direct approach would require eight residue scalers that would be highly ineffective regarding that such a set of scalers had to be placed after each butterfly. We show and analyze a structure which uses parallel-to-serial transformation of groups of numbers so that only two residue scalers are needed.

KEYWORDS: Fast Fourier Transform, residue number system, modified quadratic residue number system, pipelined FFT processor

1. Introduction

The calculation of the Discrete Fourier Transform (DFT) is usually performed with one of the Fast Fourier Transform algorithms [1-3]. A review of two decades of the FFT processor developement was given in [4]. As the computing environment general purpose computers, digital signal processors or specialised FFT processors are used. These processors are needed when a very quick calculation is required. This may be as short as single microseconds. Such requirements can be fulfilled by digital signal processors. As an example may serve TI's C66x series [5]. This processor may perform an 1K-point complex FFT (radix-4) within 5.47 μs using 6840 clock cycles at 1.25 GHz. The second example is Analog Devices (AD) TigerSHARC [6] that can perform at most 16-bit 256 point FFT (radix-2) in 0.975 μs and 1K-point complex FFT in 15.64 μs, both with 600 MHz clock. The family of ADSP-214xx processors contains an FFT accelerator, that may be used to increase the effectiveness of computation of the radix-2 FFT through dedicated hardware that supports the fundamental butterfly operation. The processing time per sample is about 10 us, when the core is running at 450 MHz with a single channel tap length of 512 [7]. Also the processing speed when using C66x can be increased by using 8

processors operating in parallel with high speed links between them. A way to increase the processing speed can be the use of pipelining. The pipelining mechanism is a common method used for pipelining instructions in general purpose processors as well as in digital signal processors within the processor structure for instruction pipelining. In the case of specialized FFT processors built as ASICs, the pipelining can be realized at the Full Adder (FA) level. Such approach provides the highest data throughput.

The FFT processors may use fixed-point arithmetic, block floating-point arithmetic [1] or Residue Number Systems (RNS)[8]. Theoretically operations in each type of arithmetic can be pipelined at low level, but in certain cases this may prove inefficient due to the high number of pipeline registers needed within the structure. Moreover, large multipliers and memories within the structure limit the pipelining rate. The use of RNS allows to attain pipelining at the FA level because the needed multipliers can be small as 5 x 5 bit and multipliers by a constant can be implemented using blocks of logic functions. Moreover, certain types of residue numbers systems facilitate complex multiplication. For example, the Modified Quadratic Residue Number System (MQRNS) [9] allows to implement complex multiplication using only three real multiplications. However, in addition to its advantages, the RNS FFT processor has also certain disadvantages because the number of processing channels is equal to the number of moduli of the RNS base and residue scalers are needed at the output at each butterfly. This is due to the fact that RNS is an integer number system but FFT algorithm coefficients are fractional numbers and they have to be transformed to integers by a multiplication by a suitable constant K and rounded off. Therefore when the RNS dynamic range is fully utilized, the product after each multiplication has to be scaled by K in order to avoid arithmetic overflow. The scaling operation is fairly complex with regard to the needed hardware amount [10-11]. At the output of the FFT processor butterfly in dependence of the type of algorithm, usually radix-2 or radix-4, there are two or four complex numbers which must be scaled. In the direct approach four or eight residue scalers would be required that seems to be highly ineffective because of the needed amount of hardware. In this paper we show that using an intermediary structure that delays and multiplexes the radix-4 butterfly output numbers, the number of scalers can be reduced to two. The structure of the radix-4 MQRNS butterfly was presented in [12]. In Section 2 we review typical FFT pipelined structures, in Section 3 we consider the general structure of the pipelined radix-4 MQRNS FFT processor and present a delay network that allows to reduce the number of residue scalers.

2. Typical FFT Radix-4 pipelined structures

A typical pipelined FFT processor consists of a number of butterfly stages that implement butterfly operations. For $N = 1024$ and radix-2, ten stages are needed and five stages for radix-4. The form of butterflies and their

interconnections determine the structure of an FFT processor. The butterfly input data has to properly ordered before butterfly computations may start. This reordering is performed using commutators. Commutators provide for the proper connection between subsequent butterfly stages in the pipeline. The form of these commutators influences the structure of the FFT processor. There are three most typical commutator types used in the pipelined FFT processors: Multi-path Delay Commutator (MDC), Single-path Delay Feedback (SDF) and Single-path Delay Commutator (SDC) [13, 14]. The single-path FFT processor architectures process two segments of the given input signal, while multi-path architectures process N data segments at the same time.

The MDC is the most classical approach for pipelined implementation radix-4 FFTs. Its form termed the R4MDC for radix-4 FFT is shown in Figure 1. Here the input sequence is broken into four parallel data streams flowing forward. Input signal samples flowing to the radix-4 butterfly (BF4) must be in a proper order and scheduled by proper delays. The disadvantage of this realization is 25% utilization of butterflies and just 50% utilization of memory blocks. This can be compensated only in some special applications where four FFTs are being processed simultaneously.

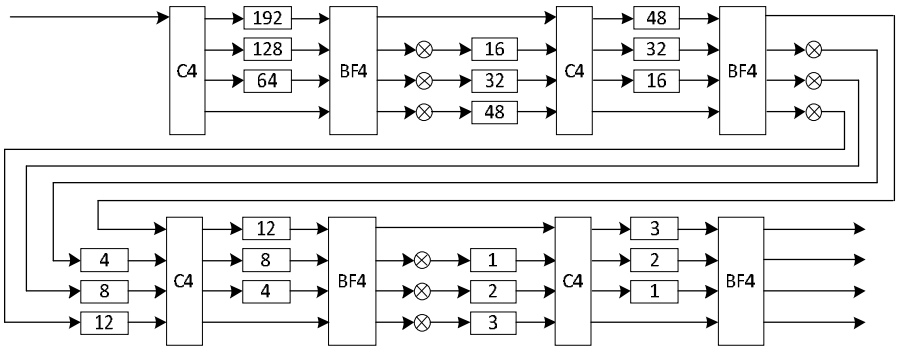

Fig. 1. The block diagram of R4MDC for $N = 256$

An economical variant, R4SDC, with respect to smaller hardware amount is given in Figure 2. Similarly as for the case of R4MDC, it allows only for 25% utilization of multipliers, butterfly elements and memory blocks. A pipelined N-point radix-4 FFT processor based on this architecture, has $\log_4 N$ stages. Each stage produces one output number within each cycle. Each stage contains a commutator (C4), a butterfly (BF4) element and a complex multiplier. The sequential outputs at each stage should be ordered. Here the butterfly element performs only the summation that can be realized by six programmable adder/subtractors with an additional control circuit. Three complex adder/subtractors (each comprising a real and an imaginary part) are used instead

of eight complex adders. Control signals can be stored in LUTs (Look-Up Table), that select data fed into add/subs modules. This butterfly architecture generates N outputs consecutively in N cycles, compared to the R4MDC butterfly which generates N outputs in $N/4$ cycles, with $N/3$ idle cycles.

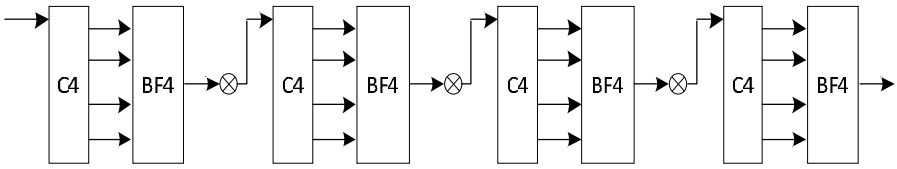

Fig. 2. The block diagram of R4SDC for $N = 256$

However, the higher utilization of components is desirable. This can be achieved in the R4SDF (Fig. 3), by increased to 100% utilization of memory blocks, in which both the input and the output data stream are stored. In effect the utilization of multipliers is 75% by storing three BF4 outputs [15]. In general, SDF architectures have the most efficient memory utilization for pipelined FFT processors.

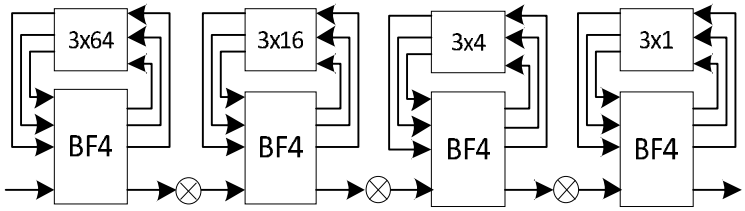

Fig. 3. The block diagram of R4SDF for $N = 256$

An effective example of better memory utilization and in consequence lower power needed is the R2^{2}SDF DIF architecture (Fig. 4). It is based on radix-2 FFT algorithm with utilization of pseudo radix-4 butterfly. The input samples should be in natural order and the output is generated in a bit-reversed order.

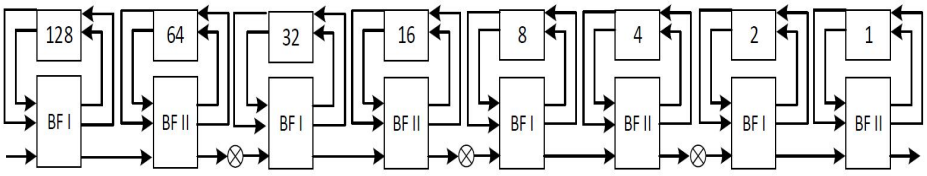

Fig. 4. The block diagram of R2^{2}SDF (DIF) for $N = 256$

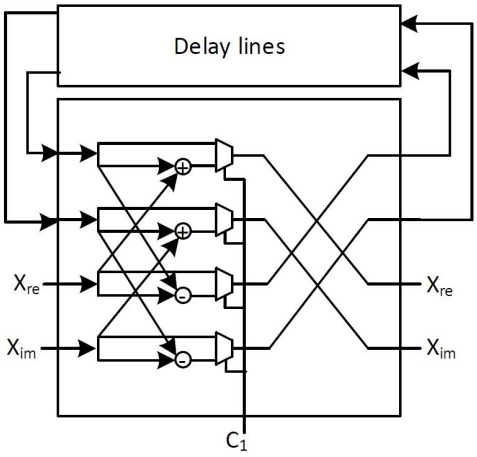

Fig. 5. The block diagram of R22SDF BF I

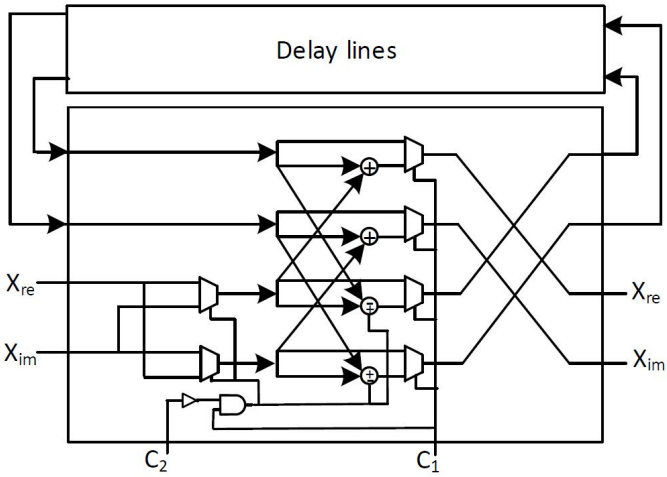

Fig. 6. The block diagram of R22SDF BF II

In the R2^{2}SDF architecture two distinct butterfly structures are used: BF I (Fig. 5) and BF II (Fig. 6). The BF I computes two-point DFT, for k cycles, where $k = N/2^i$ is the length of delay lines at every stage i. During the first k cycles, c_1 should be low and the input stream is directed straight into delay line registers. In the subsequent k cycles, when c_1 is high, the addition of input and feedback data is performed. Similarly, the BF II for the first k cycles loads the input stream to the registers in delay lines. For the next $k/2$ cycles, when c_1 is high and c_2 is low, the input stream and the output from delay lines are added. Finally, for the last $k/2$ cycles, when c_1 and c_2 are high, the input samples are swapped, that corresponds to multiplication by $-j$. This process is repeated every N samples.

In Table 1 a comparison of hardware requirements for the individual pipelined FFT processors is given. The weakness of presented architectures is non-effective utilization of memory blocks. The more efficient in this respect are architectures based on the delay feedback rather than on delay commutators. Additionally, the radix-4 SDF requires less multipliers than radix-4 MDC. The R2^{2}SDF can be treated s a trade-off between R2SDF and R4SDF structures, where the multiplication complexity is reduced. Here still 4 real multiplications and 2 real additions make up one multiplier [15].

Table 1. Comparison of required hardware for commutator schemes in pipelined *FFT*

The pipeline scheme	multipliers	butterflies	registers
R2MDC	$\log_2(N-2)$	$\log_2 N$	$3/2N$
R2SDF	$\log_2(N-1)$	$\log_2 N$	$N-1$
R4SDF	$\log_4(N-1)$	$\log_4 N$	$N-1$
R4MDC	$3\log_4 N$	$\log_4 N$	$5/2N-4$
R4SDC	$\log_4(N-1)$	$\log_4 N$	$2(N-1)$
R2^{2}SDF	$\log_4(N-1)$	$\log_4 N$	$N-1$

3. General structure of radix-4 fft pipelined processor based on MQRNS

The radix-4 FFT processor that uses the MQRNS consists of $\log_4 N$ stages, where each stage comprises *n* butterflies, one for each modulus of the RNS base. The flow of MQRNS calculations has been given previously in [5]. As stated above after each stage the residue scaling is needed. The scaling is performed in a block of residue scalers [8]. As each butterfly generates four residue numbers in the Complex Residue Number System (CRNS) form, in total, eight real residue numbers have to be scaled. The direct approach would require eight residue scalers after each stage of butterflies (Fig. 7). This would lead to an excessive number of scalers (in this case 40 scalers), that would make the implementation of the processor impractical. This is due to the fact that a single scaler is a fairly complex circuit. The number of scalers can be reduced by applying the special switching matrix (Fig. 8) in which the butterfly outputs are suitably delayed, so that only two scalers instead of eight can be used.

At an instatnt t_0 eight real residue numbers come up at the output of the each butterfly. The numbers are stored in the first stage of registers. In the next cycle they are written to the next level. After four cycles two numbers from the registers Re_0 and Im_0 are directed using the multiplexers MUX1 and MUX2 to

the respective inputs of the scalers SCALER1 and SCALER2. During the next cycles the properly delayed data enters the scalers.

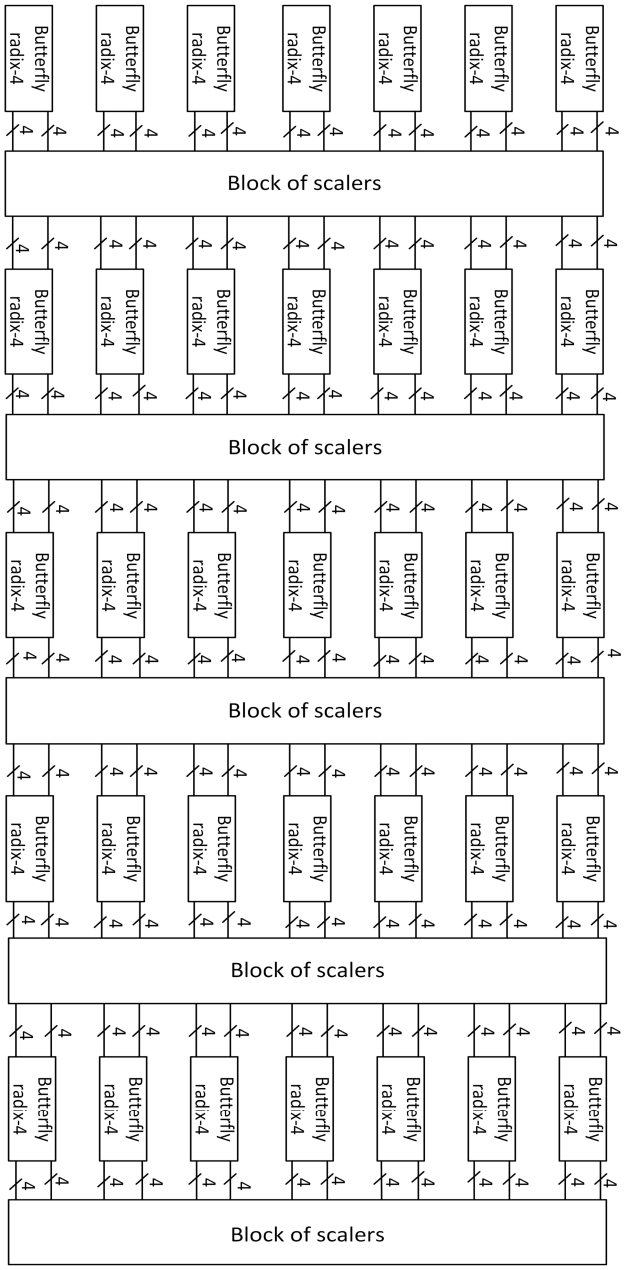

Fig. 7. The block diagram of radix-4 MQRNS FFT processor

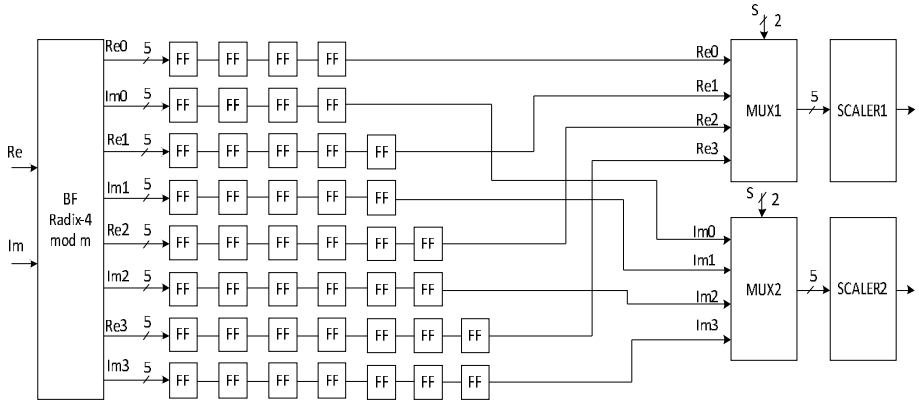

Fig. 8. The structure of the parallel-to-serial converter at the butterfly output

4. Summary

We presented a solution to the problem of reduction of the number of scalers within the radix-4 FFT MQRNS processor. The use of the MQRNS allows to attain high pipelining rates limited by the delay of a pipeline register and a Full Adder delay. However, the price paid is the need to scale residue numbers after each stage of butterflies. As there are eight residue numbers to be scaled, in the direct approach eight scalers after each stage would be needed. This would render this solution impractical. We propose an intermediary structure between butterfly stages that by a suitable redirection and multiplexing allows to use only two scalers after each stage. This is at the cost of introducing certain delay, which is, however, not meaningful as compared to the overall delay of the pipeline.

References

[1] Rabiner L.R., Gold B.: Theory and Application of Digital Signal Processing. Prentice-Hall 1975.

[2] Oppenheim A.V., Schafer R.W.: Discrete-Time Signal Processing, Third Edition. Prentice-Hall, 2009.

[3] Brigham E.O.: The Fast Fourier Transform and Its Applications. Prentice-Hall, Upper Saddle River, NJ, 1988.

[4] Swartzlander Jr. E.E.: Systolic FFT processors: A Personal Perspective. Journal of Signal Processing Systems, Number 53, 2008, Pages 3 – 14.

[5] Texas Instruments, TI C66x, http://www.ti.com .

[6] Analog Devices: TigerSHARC processor benchmarks. http://www.analog.com/en/content/tigersharc_benchmarks/fca.html, October 2014.

[7] Analog Devices, ADSP-214xx SHARC® Processor Hardware Reference, http://www.analog.com/static/imported-files/processor_manuals/ADSP-214xx_hwr_rev1.1.pdf, Apil 2013.
[8] Soderstrand M.A. *et al*.: Residue Number System Arithmetic: Modern Applications in Digital Signal Processing. IEEE Press, Piscataway, NJ, 1986.
[9] Krishnan R., Jullien G.A., Miller W.C.: The modified quadratic residue number system (MQRNS) for complex high-speed signal processing. IEEE Transactions on Circuits and Systems, Volume 33, Number 3, Pages 325 – 327, 1986.
[10] Ulman Z., Czyżak M.: Highly parallel, fast scaling of numbers in nonredundant residue arithmetic. IEEE Trans. on Signal Processing, Volume 45, Number 2, 1998, Pages 487 – 496.
[11] Czyżak M., Smyk R., Ulman Z.: Pipelined scaling of signed residue numbers with the mixed-radix conversion in the programmable gate array. Poznan University of Technology Academic Journals. Electrical Engineering, Number 76, Pages 89 – 99, 2013.
[12] Czyżak M., Smyk R.: Radix-4 DFT butterfly realization with the use of the modified quadratic residue number system. Poznan University of Technology Academic Journals, Electrical Engineering, Number 63, Pages 39 – 51, 2010.
[13] Shousheng H., Torkelson M.: A new approach to pipeline FFT processor. Proceedings of the 10th International Parallel Processing Symposium IPPS '96, Pages 766 – 770, 1996.
[14] Wold E., Despain A.: Pipeline and parallel-pipeline FFT processors for VLSI implementations, IEEE Transactions on Computers, Volume C-33, Number 5, Pages 414 – 426, May 1984.
[15] Shousheng H., Torkelson M., Shousheng H., Torkelson M., Designing Pipeline FFT processor for OFDM (de)modulation, Signals, Systems, and Electronics, 1998. ISSSE 98, 1998 URSI International Symposium on, Pages 257 – 262, 29 Sep – 2 Oct 1998.
[16] Wenqi L., Xuan W., Xiangran S.: Design of Fixed-Point High-Performance FFT Processor, 2010 2nd International Conference on Education Technology and Computer (ICETC), Volume 5, Pages V5-139 – V5-143, June 22-24, 2010.