Hardware-Efficient Index Mapping for Mixed Radix-2/3/4/5 FFTs

Tomasz Patyk
Dolby Poland
Wroclaw, Poland
Email: tomasz.patyk@dolby.com

Jarmo Takala
Tampere University of Technology
Tampere, Finland
Email: jarmo.takala@tut.fi

***Abstract*—Orthogonal frequency-division multiplexing modulators and demodulators for modern communication standards require efficient implementation of the fast Fourier transform (FFT). Traditionally, radix-2 and radix-4 FFT algorithms have been used. Over the last few years, support for non-power-of-two transform sizes, with the emphasis on the radix-3 and radix-5, started to become a standard. We have created a systematic approach for designing simple digital circuits that compute array access indices for the mixed radix-2/3/4/5 FFT computations. Proposed index mapping, allows for the use of a bit rotation instead of the add/modulo and multiply operations. Index generation circuits, implementing the proposed index mapping, have hardware complexity comparable to index generation circuits for power-of-two FFTs.**

I. Introduction

In the year 1965, Cooley and Tukey presented their seminal paper [1] and brought the fast Fourier transform (FFT) algorithm to scientific and engineering communities' attention. Their work, laid the groundwork for a fledgling discipline of the digital signal processing (DSP). Since then, the FFT along with digital filters constitute two most important classes of DSP algorithms [2].

The attractiveness of the FFT algorithm comes from the computational complexity reduction it offers. The quadratic O(N^2) complexity of the discrete Fourier transform (DFT) calculated by definition, is reduced to a linearithmic O($NlogN$) complexity. Cooley and Tukey [1] proved that for a given radix, complexity is proportional to $NlogN$. Compared to a direct calculation of the DFT, FFT algorithms exploit two techniques to avoid redundant operations: the divide and conquer technique, and a short length DFT optimisation. By employing the divide and conquer approach and reorganising data, a large sized DFT is recursively divided into smaller ones. This was extensively described by many authors [1], [3], [4], [5] and generalised to multidimensional index mapping by Burrus [6]. Two classes of factorisation divide FFT algorithms into prime factor algorithms (PFA) and common factor algorithms (CFA). Optimisation of a short length DFT is most commonly known for power-of-two DFTs of size 2 and 4 (radix-2 and radix-4 butterflies) which can be implemented without multiplication and with only additions. Winograd [4] presented optimised DFTs of non-power-of-two sizes of 3, 5, and 7.

Historic applications of the FFT include spectral analysis, filter banks, convolutions, and many more [2]. However, recent years have put a spotlight on the FFT use in communication applications. In particular, orthogonal frequency-division multiplexing (OFDM) which is used in wideband digital communication networks including 3G and 4G mobile communication networks. Efficient OFDM modulator and demodulator implementations are often based on the IFFT and FFT computations [7]. One of the challenges that the FFT has to face in the new field of application is an efficient implementation of non-power-of-two DFT sizes. As the uplink precoding [7] of Long-Term Evolution (LTE) requires transform sizes of $12-1296$, it is clear that efficient implementations of mixed radix algorithms, including radices 3 and 5, will gain popularity. Challenges are yet to overcome as non-power-of-two computations of odd radices are not trivial to implement on binary logic, which we use today.

Linear mapping of data indices from one-dimensional to multi-dimensional mapping reduces the numbers of multiplications and additions required to calculate a DFT. However, performing linear mapping itself can be non-trivial task taking up significant chunk of hardware resources or computational time [8]. Therefore, non-power-of-two DFT sizes and algorithms with regular access patterns have been preferred over the years. Efficient hardware implementations for single radix-2, radix-4, and mixed-radix-2/4 are reported in the literature. On the contrary, number of hardware implementations for non-power-of-two DFT sizes is limited. Typically those papers present extra hardware for add/subtract and modulo operations [9] or complex multiplications [10].

In this paper, we propose a novel hardware efficient array index mapping for a decimation-in-time (DIT) mixed-radix-2/3/4/5 FFT algorithms to be used for in-place computations. The scheme is generic as it supports any DFT size which can be factorised to supported radices. The order of radices can be chosen arbitrarily. Additionally, we present an implementation of an index generator circuit (IG). The implementation is based on pseudo-linear counter and rotators, and it does not require any multipliers or extra adders. Small design corresponds to less silicon area, shorter critical path, and reduced power consumption. Latter is especially important for battery-powered portable devices where new communication

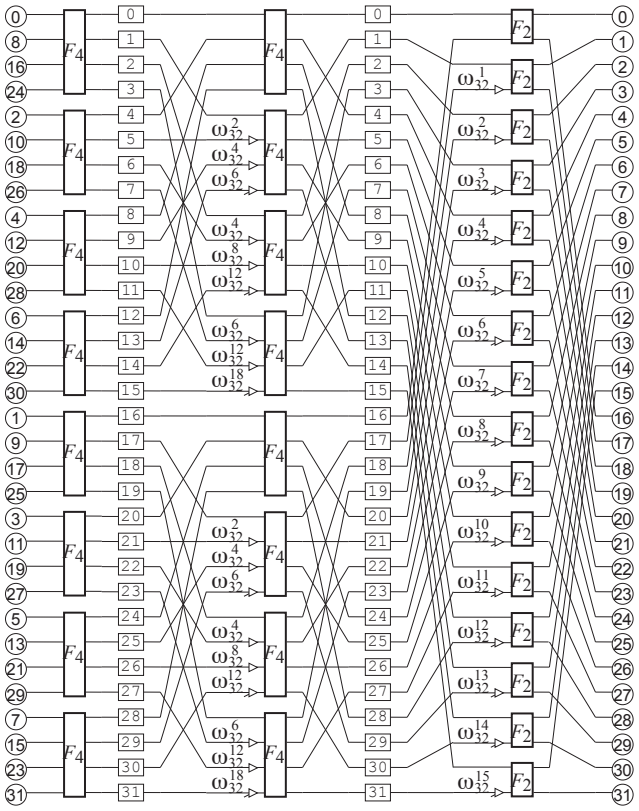

Fig. 1. Signal flow graph of 32-point FFT.

TABLE I
ACCESS INDICES OBTAINED FROM LINEAR INDICES IN BINARY NUMERAL SYSTEM FOR 3 STAGES OF 32-POINT FFT (ROTATED BITS IN BOLD).

Linear		Access	
R-10	R-2	R-2	R-10
Stage 1			
	$2^2 4^1 \mathbf{4^0}$	$2^2 4^1 \mathbf{4^0}$	
	$b_4 b_3 b_2 \boldsymbol{b_1 b_0}$	$b_4 b_3 b_2 \boldsymbol{b_1 b_0}$	
0	000000	000000	0
1	000001	000001	1
2	000010	000010	2
3	000011	000011	3
4	000100	000100	4
...	...	...	...
Stage 2			
	$2^2 \mathbf{4^1} 4^0$	$2^2 \mathbf{4^1} 4^0$	
	$\boldsymbol{b_4 b_1 b_0} b_3 b_2$	$\boldsymbol{b_4 b_3 b_2 b_1 b_0}$	
0	00000	00000	0
1	00001	00100	4
2	00010	01000	8
3	00101	01100	12
4	00100	00001	1
...	...	...	...
Stage 3			
	$\mathbf{4^2} 4^1 \mathbf{2^0}$	$\mathbf{2^2} 4^1 \mathbf{4^0}$	
	$\boldsymbol{b_3 b_2 b_1 b_0} b_4$	$\boldsymbol{b_4 b_3 b_2 b_1 b_0}$	
0	00000	00000	0
1	00001	10000	16
2	00010	00001	1
3	00011	10001	17
4	00100	00010	2
...	...	...	...

standards are applicable.

The rest of the paper is organised as follows. Section II presents current state of the research in the field. Section III explains the novelty of the proposed index mapping. Section IV proposes a hardware efficient implementation of the new mapping. Section V discusses advantages of the design as compared to existing solutions. Finally, section VI concludes the paper.

II. Related Work

There are several aspects of a FFT algorithm that translate to different index mapping. For a given DFT size, there are decimation-in-time [1] or decimation-in-frequency [3] algorithms; input sequence in order or permuted; for a mixed radix FFT, the order of radices can be chosen arbitrary, and finally, a computation stage can have regular or irregular (split-radix) geometry [5]. Demuth [11] proposed an unified set of equations, that allow FFT computations of an arbitrary DFT size. The implementation based on three nested loops can be implemented on programmable processor using general purpose arithmetic logic unit (ALU). Naturally, using ALU for index computations usually means that ALU does not compute FFT kernel (butterfly). This approach is flexible but far from optimal at the same time.

FFT implementations based on application-specific integrated circuits (ASIC) or instruction-set processors (ASIP) tend to use dedicated index (address) generation units (IG). Hardware-efficient IG units have been proposed for power-of-two FFTs with regular signal flow graphs, both single radix [12] and mixed radix [13]. These design transform a linear index generated with an accumulator to the actual array access index through a bit rotation, where the number of rotated bits depends on the computational stage in the FFT. The rotations can be implemented with a set of multiplexers making the design as hardware-optimal.

Efficiency of hardware implementation for power-of-two FFTs is related closely to binary logic; Indices are obtained with multiplications/divisions by power-of-two factors, which can be efficiently implemented with shifters. This is not the case for non-power-of-two radices. For instance single radix-3 FFT will require multiplication by power-of-three numbers. For mixed radix-3/4 FFT index calculations will require multiplication by integer multiples of 12.

Few authors [14], [15], [16], [17] proposed a LTE compliant designs, which support DFT sizes ranging power-of-two as 128–2048 and 1536. However, 1536 can be considered as a special case as it contains a single radix-3 stage, which can be left last in the computations, thus the ordinary power-of-two index mapping hardware can be used.

Hsiao [9] proposed a generalised, mixed radix algorithm and its hardware implementation. Address (index) generation hardware unit is a direct implementation of the index mapping proposed by Burrus [6]. All indices for a single butterfly are generated in parallel. This requires two accumulators, adder, and a modulo circuit per butterfly input. For each stage

TABLE II
ACCESS INDICES OBTAINED FROM LINEAR INDICES IN TERNARY NUMERAL SYSTEM FOR 3 STAGES OF 27-POINT FFT.

Linear		Access					
All stages		Stage 1		Stage 2		Stage 3	
R-10	R-3	R-3	R-10	R-3	R-10	R-3	R-10
0	000	000	0	000	0	000	0
1	001	001	1	010	3	100	9
2	002	002	2	020	6	200	18
3	010	010	3	001	1	001	1
4	011	011	4	011	4	101	10
5	012	012	5	021	7	201	19
...	...	...	...	...	...	...	...

different seed values need to be provided for the accumulators. In addition, modulo logic must be set to DFT size. Each index is obtained by adding together outputs of two accumulators and modulo truncated if needed. The authors did not explain, if the seed values required for accumulators, are calculated on the fly or stored in a ROM memory and fetched by the control logic. Chen [18] extends the previous method to support wider range of radices. The extended scheme also supported several DFT sizes using the same address (index) generation hardware.

Ma [10] proposed an approach, where pseudo-linear address from accumulator is first rotated. The number of rotated bits depends on the stage and decreases as computation proceeds. Certain chunks of the accumulator are then multiplied by constant values. The products of those multiplications are added together to obtain the final index. For longer DFT sizes, this approach might become extremely expensive as it requires $(s-1)$ multiplications and additions for s computational stages.

The proposed index mapping supports non-power-of-two radices and mixed radix FFTs. Unlike the previous solutions discussed earlier it requires neither add/modulo or multiplication to obtain memory access indices.

III. PROPOSED INDEX MAPPING

The DFT of an input vector $Y = [x_0, x_1, ..., x_{N-1}]^T$ is defined as the vector $Y = [y_0, y_1, ..., y_{N-1}]^T$ such that:

$$y_m = \sum_{n=0}^{N-1} \omega_N^{nm} x_n, \tag{1}$$

or equivalently $Y = F_N X$, where F_N is the $(N \times N)$-matrix of DFT with entries:

$$\omega_N^{nm} = \exp -\frac{i2\pi nm}{N}. \tag{2}$$

Many FFT algorithms were derived for efficient computation of the DFT. In this paper, we use the in-place, decimation-in-time (DIT), mixed radix algorithm with permuted input, and in-order output. The formula for this FFT algorithm is given by (3):

$$F_{p_1^m} = \prod_{s=m}^{1} \left\{ \left[I_{p_{s+1}^m} \otimes (F_{r_s} \otimes I_{p_1^{s-1}}) \right] (I_{p_{s+1}^m} \otimes T_{r_s, p_1^{s-1}}) \right\} R_{p_1^m}, \tag{3}$$

TABLE III
ACCESS INDICES OBTAINED FROM LINEAR INDICES IN MIXED RADIX NUMERAL SYSTEM FOR 3 STAGES OF 144-POINT FFT (FIRST STAGE OMITTED).

Linear		Access	
R-10	R-3/4/3/4	R-3/4/3/4	R-10
Stage 2			
0	0000	0000	0
1	0001	0010	3
2	0002	0020	6
3	0003	0030	9
4	0010	0001	1
5	0011	0011	4
6	0012	0021	7
7	0013	0031	10
...	...	...	...
Stage 3			
0	0000	0000	0
1	0001	0100	12
2	0002	0200	24
3	0010	0001	1
4	0011	0101	13
5	0012	0201	25
...	...	...	...
Stage 4			
0	0000	0000	0
1	0001	1000	36
2	0002	2000	72
3	0003	3000	108
4	0010	0001	1
5	0011	1001	37
6	0012	2001	73
7	0013	3001	109
...	...	...	...

where I_N denotes an identity matrix of order N, $\otimes$ is a tensor product, m is number of stages, s is the stage index in $s \in (1, m)$, r_s is the radix of the s^{th} stage, and p_n^m is a product of radices of stages n to m given by:

$$p_n^m = \begin{cases} \prod_{i=n}^{m} r_i & \text{if } n \leq m, \\ 1 & \text{if } n > m. \end{cases} \tag{4}$$

T matrix of stage twiddle factors is given by:

$$T_{r,k} = \oplus_{i=0}^{r-1} D_N^{ki}, \tag{5}$$

$$D_N^{ki} = diag(\omega_N^{0i}, \omega_N^{1i}, \ldots, \omega_N^{(k-1)i}), \tag{6}$$

where $N = rk$ equals DFT size and $\oplus$ denotes the matrix direct sum. Finally, $R_{p_1^m}$ is an input permutation matrix:

$$R_{p_1^m} = \prod_{s=m}^{1} (I_{p_{s+1}^m} \otimes P_{p_1^{s-1}, r_s}) \tag{7}$$

based on the stride-by-K permutation matrix [19] of order of N defined as follows:

$$[P_{N,K}]_{m,k} = \begin{cases} 1 & \text{if } k = (mK \mod N) + \lfloor mK/N \rfloor, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

The signal flow graph of a 32-point mixed radix-4/4/2 FFT is illustrated in Figure 1. The algorithm uses in-place processing; input data is read through a stride permutation, processed in stages, and overwritten by the result data. Thus,

TABLE IV
PROPOSED INDEX MAPPING. ACCESS INDICES OBTAINED FROM LINEAR INDICES IN BINARY CODED MIXED RADIX NUMERAL SYSTEM FOR 3 STAGES OF 27-POINT FFT.

R-10	Linear BCMR	Access BCMR	R-10
	Stage 1		
	$3^2 3^1 \mathbf{3^0}$	$3^2 3^1 \mathbf{3^0}$	
	$b_5 b_4 b_3 b_2 \mathbf{b_1 b_0}$	$b_5 b_4 b_3 b_2 \mathbf{b_1 b_0}$	
0	000000	000000	0
1	000001	000001	1
2	000010	000010	2
4	000100	000100	4
5	000101	000101	5
6	000110	000110	6
...	...	...	...
	Stage 2		
	$3^2 \mathbf{3^0 3^1}$	$3^2 \mathbf{3^1 3^0}$	
	$\mathbf{b_5 b_4 b_1 b_0 b_3 b_2}$	$\mathbf{b_5 b_4 b_3 b_2 b_1 b_0}$	
0	000000	000000	0
1	000001	000100	4
2	000010	001000	8
4	000100	000001	1
5	000101	000101	5
6	000110	001001	9
...	...	...	...
	Stage 3		
	$\mathbf{3^1 3^0 3^2}$	$\mathbf{3^2 3^1 3^0}$	
	$\mathbf{b_3 b_2 b_1 b_0 b_5 b_4}$	$\mathbf{b_5 b_4 b_3 b_2 b_1 b_0}$	
0	000000	000000	0
1	000001	010000	16
2	000010	100000	32
4	000011	000001	1
5	000100	010001	17
6	000101	100001	33
...	...	...	...

TABLE V
ACCESS INDICES OBTAINED FROM LINEAR INDICES IN MIXED RADIX NUMERAL SYSTEM FOR 3 STAGES OF MIXED RADIX-3/4/3/4 144-POINT FFT (FIRST STAGE OMITTED).

R-10	Linear BCMR	Access BCMR	R-10
	Stage 2		
	$4^3 3^2 \mathbf{3^1 4^0}$	$4^3 3^2 \mathbf{4^1 3^0}$	
	$b_7 b_6 b_5 b_4 \mathbf{b_1 b_0 b_3 b_2}$	$b_7 b_6 b_5 b_4 \mathbf{b_3 b_2 b_1 b_0}$	
0	00000000	00000000	0
1	00000001	00000100	4
2	00000010	00001000	8
3	00000011	00001100	12
4	00000001	00000001	1
5	00000101	00000101	5
6	00000110	00001001	9
7	00000111	00001101	13
...	...	...	...
	Stage 3		
	$4^3 \mathbf{4^2 3^1 3^0}$	$4^3 \mathbf{3^2 4^1 3^0}$	
	$b_7 b_6 \mathbf{b_3 b_2 b_1 b_0 b_5 b_4}$	$b_7 b_6 \mathbf{b_5 b_4 b_3 b_2 b_1 b_0}$	
0	00000000	00000000	0
1	00000001	00010000	16
2	00000010	00100000	32
4	00000100	00000001	1
5	00000101	00010001	17
6	00000110	00100001	33
...	...	...	...
	Stage 4		
	$\mathbf{3^3 4^2 3^1 4^0}$	$\mathbf{4^3 3^2 4^1 3^0}$	
	$\mathbf{b_5 b_4 b_3 b_2 b_1 b_0 b_7 b_6}$	$\mathbf{b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0}$	
0	00000000	00000000	0
1	00000001	01000000	64
2	00000010	10000000	128
3	00000011	11000000	192
4	00000100	00000001	1
5	00000101	01000001	65
6	00000110	10000001	129
7	00000111	11000001	193
...	...	...	...

the same index mapping can be used for reading and writing. Data fed to the first stage is shuffled with (7) while the output data is in order. The array access indices for s^{th} radix stage, defined in (3) by tensor products, can be represented with a permutation matrix as:

$$I_{p_{s+1}^m} \otimes P_{r_s p_1^{s-1}, p_1^{s-1}}. \tag{9}$$

The equation (9) implies that index computations require multiplication, division, modulo operation, and addition. However, for stride-by-K matrix of order N, where both K and N are power-of-two values, the access address can be obtained with a rotation of linear binary index [12], [13]. For stage s, $\sum_{i=1}^{s} log_2 r_i$-least significant bits are rotated right by $log_2 r_s$ bits. Linear indices rotated to get access indices of the 32-point FFT are presented in Table I. Bits $b_1 b_0$ and $b_3 b_2$, correspond to radix-4 stage 1 and 2, respectively. Bit b_4 represents stage 3 of radix-2. The bits of the linear index indicated in bold are rotated two bits to the right in the first two stages, and rotated one bit to the right in the last stage. The decimal values of the indices are shown in the radix-10 (R-10) columns of the table.

It can be observed that this principle applies to any radix, including mixed radix, as long as the index is represented in positional numeral system based on the radices used in the FFT computations. For instance, access indices for 27-point FFT with three radix-3 stages, can be obtained through a rotation of a linear index represented in the ternary system as shown in Table II. Likewise, Table III illustrates the relation between linear and access indices of 144-point mixed radix FFT. Indices are represented in the radix-3/4/3/4 numeral system corresponding to order of radices in the FFT computations. Unfortunately, in the binary representation the rotation alone does not produce expected access indices anymore.

In this paper, we propose to use linear index in binary-coded mixed radix (BCMR) representation. BCMR encoding, alike binary-coded decimal (BCD), uses $n = \lceil log_2 r_s \rceil$-bits to represent a numerical digit. Contrary to BCD, radix, and therefore number of bits representing digits in the BCMR, might be different for each numerical position. For the linear index, radices assigned to numerical position change with the stage. The radix of the least significant digit r_s is the radix of current stage s. Other digits have radices assigned according to the order of radices in the FFT computations. Rotating linear indices produces an unambiguous mapping to access indices by rearranging all radices to the order in which their are computed. Tables IV and V presents BCMR-based index mappings for the 27- and 144-point FFTs, respectively.

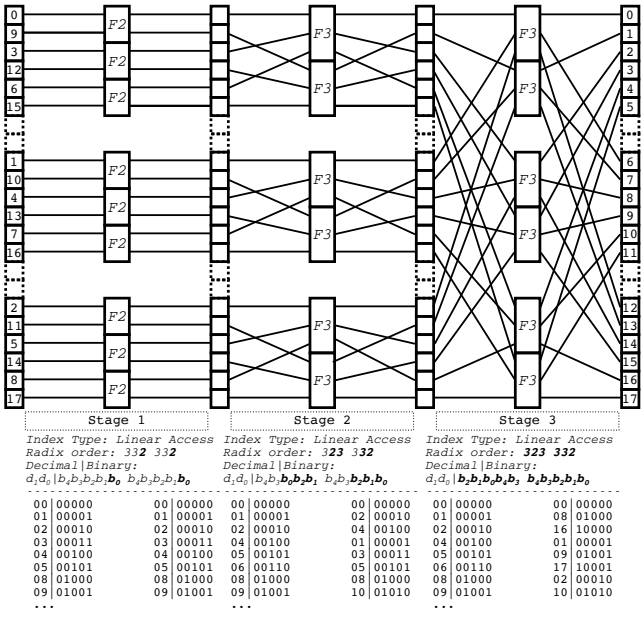

Fig. 2. Signal flow graph of 18-point FFT (multiplications by twiddle factors omitted for clarity).

Comparison of the BCMR-based indices with corresponding decimal indices in Tables II and III reveals that the proposed mapping, nonetheless unambiguous, is discontinuous in the address space. As a result, a larger memory space is needed than the DFT size indicates. In practice, however, FFT processing units support many DFT sizes; if the largest supported power-of-two DFT uses n bits for access indices, all non-power-of-two DFTs that use n bits for addressing, are supported too. Figure 2 gives an example of a signal flow graphs for 18-point FFT with $n = 5$, which requires 22 memory cells, covered by 2^5-point DFT. Similarly, signal flow graph of the 30-point, given in Figure 3, requires 42 memory cells supported by 2^6-point DFT. It must be noted that we carry out in-place computations, i.e., the same memory cells are used through all computational stages. The discontinuities should only be considered on the input and output accesses to the FFT. All in all, practical systems need to support several FFT sizes, thus the proposed method exploits only the memory, which is already available in the system.

IV. Hardware Implementation

The index mapping proposed in section III can be efficiently implement in hardware. The unit is comprised of two blocks: the mixed radix accumulator and the rotator. The rotator converts the linear index produced by the accumulator to obtain the array access index.

Figure 4 depicts an example circuit designed for the 30-point FFT presented in Figure 3. The unit has two inputs and a 6-bit output to read access indices from. The 1-bit *trig* input, if set to 1, triggers unit to produce consecutive access indices on each clock cycle. The *stage* input sets internal multiplexers and demultiplexers through a simple logic (not shown on the figure). This input could be omitted if the

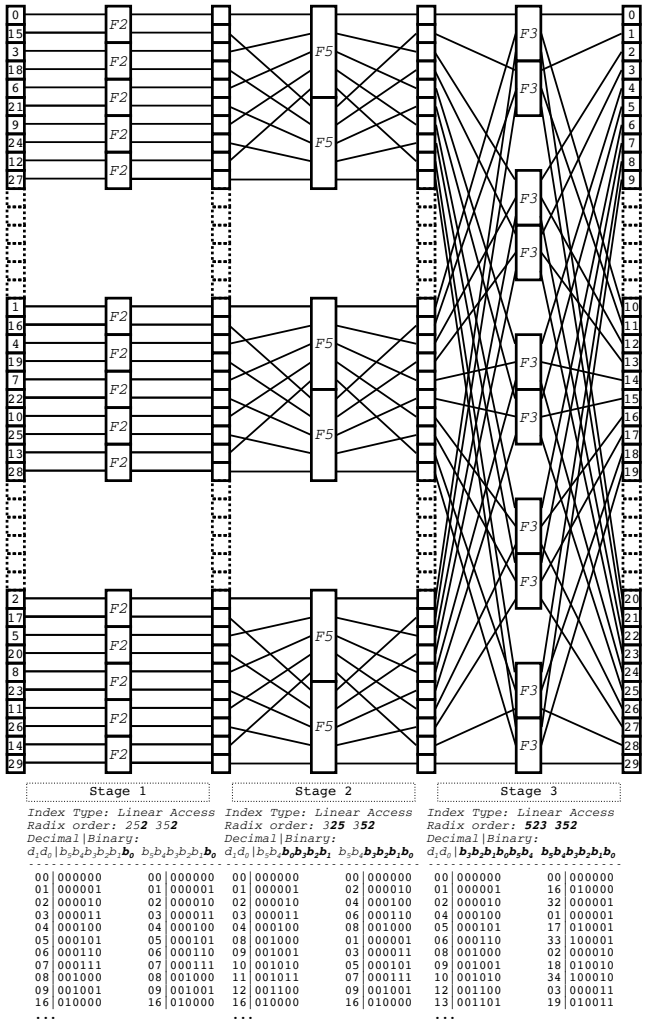

Fig. 3. Signal flow graph of 30-point FFT (multiplications by twiddle factors omitted for clarity).

unit had internal stage counter. The *rotator* is build from six, 6-input multiplexers. The *mixed radix accumulator* has three radix accumulator blocks, one per stage. Each block is connected to the *trig* input and other blocks through a set of multiplexers and demultiplexers. Those are configured for s^{th} stage such that r_s accumulator block is set to be the first, and remaining blocks are connected in the order of computations. The $radix - 1$ values must be set to blocks on initialisation and remain unchanged during the FFT computations.

An example of a $radix - 3/4$ *accumulator block (R3/4)*, a building block of the *mixed radix accumulator*, is given in Figure 5. When input c_{in} is set to 1, the *incrementer* increases its value by 1 on each clock cycle. When *comparator* detects $radix - 1$ value in the registers, $0s$ are written to them in the next clock cycle.

This design can be easily extended to any DFT size and radix configuration if corresponding *radix accumulator blocks* are present. Multiple DFT sizes can be supported when simple logic controlling multiplexers and demultiplexers is used. A design with s radix accumulator blocks requires: s-input

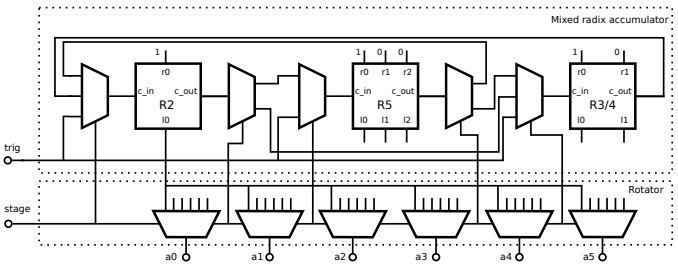

Fig. 4. Example of index generator for 30-point FFT shown in Figure 3.

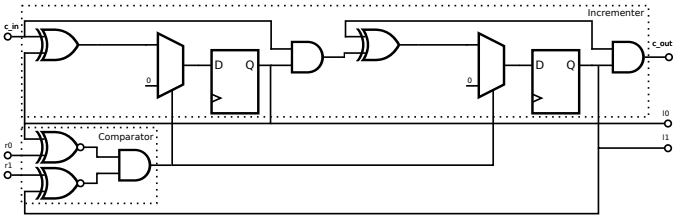

Fig. 5. Exemplary R3/4 accumulator block.

multiplexer and 2-output demultiplexer for the first block; 3-input multiplexer for the last block; 2 or 3-input multiplexer and 2-output demultiplexer for the next to last block; and 3-input multiplexer and 3-output demultiplexer for all the remaining blocks.

V. Comparison

As explained in section IV, the implementation of the proposed index mapping for n-bit addressing space requires n flip-flops, a set of multiplexers connecting them, simple comparators, and n multiplexers for bit rotation. Compared to implementations [12] and [13], which only support power-of-two DFT sizes, connecting multiplexers and comparators were added. When the number of supported radices was extended beyond power-of-two. Designs presented in [9] and [18] support odd radices but at the cost of two accumulators, adder, and modulo operation implemented with a subtractor and a multiplexer. They can obtain several addresses in parallel by duplicating hardware resources while this can be done also with the proposed method. Finally, the design in [10], alike the proposed method, uses bit rotation, but partial results require costly multiplications and additional addition.

VI. Conclusion

In this paper, we proposed a novel array index mapping for mixed radix-2/3/4/5 FFTs. The mapping allows for the use of a bit rotation, instead of add, modulo and multiply operations, to obtain indices for memory accesses in the FFT computations. A systematic method for designing a hardware-efficient implementation was discussed. Address generation routine is simple and requires small silicon footprint. The proposed method can be used as a memory addressing unit, in a application-specific fixed-function FFT processor or a address generator unit of a application-specific instruction-set processor (ASIP), for low-power and fast FFT computations.

References

[1] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comp.*, vol. 19, pp. 297–301, Apr. 1965.

[2] A. V. Oppenheim and Schafer, *Discrete-time signal processing*, 3rd ed., ser. Prentice Hall signal processing series. Upper Saddle River, NJ, USA: Prentice Hall, 2009.

[3] W. M. Gentleman and G. Sande, "Fast Fourier transforms: For fun and profit," in *Proc. of the AFIPS Fall Joint Comp. Conf.*, 1966, pp. 563–578.

[4] S. Winograd, "On computing the discrete Fourier transform," *Math. Comp.*, vol. 32, no. 141, pp. 175–199, Jan. 1978.

[5] P. Duhamel, "Implementation of "Split-radix" FFT algorithms for complex, real, and real-symmetric data," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 34, no. 2, pp. 285–295, Apr. 1986.

[6] C. S. Burrus, "Index mappings for multidimensional formulation of the DFT and convolution," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 25, no. 3, pp. 239–242, Jun. 1977.

[7] G. (2008), "Evolved universal terrestrial radio access (E-UTRA); physical channels and modulation," Tech. Rep. ETSI TS 136 211. [Online]. Available: http://www.3gpp.org/lte/

[8] K.-L. Wong, R. Chan, D. P.-K. Lun, and W.-C. Siu, "Efficient address generation for prime factor algorithms," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 38, no. 9, pp. 1518–1528, Sep. 1990.

[9] C.-F. Hsiao, Y. Chen, and C.-Y. Lee, "A generalized mixed-radix algorithm for memory-based FFT processors," *IEEE Trans. Circuits Syst. II*, vol. 57, no. 1, pp. 26–30, Jan. 2010.

[10] C. Ma, Y. Xie, H. Chen, Y. Deng, and W. Yan, "Simplified addressing scheme for mixed radix fft algorithms," in *Proc. IEEE Acoust., Speech, Signal Process. (ICASSP)*, Florence, Italy, May 2014, pp. 8355–8359.

[11] G. L.Demuth, "Algorithms for defining mixed radix FFT flow graphs," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 37, no. 9, pp. 1349–1358, Sep. 1989.

[12] E. Chu and A. George, *Indised the FFT black box*, ser. Computational Mathematics. Boca Raton, FL, USA: CRC Press, 1999.

[13] T. Pitkänen, R. Mäkinen, J. Heikkinen, T. Partunen, and J. Takala, "Transport triggered architecture processor for mixed-radix FFT," in *Proc. IEEE Fortieth Asilomar Conf. on Signals, Syst. Comput.*, Pacific Grove, CA, Oct. 2006, pp. 84–88.

[14] S.-Y. Peng, K.-T. Shrt, C.-M. Chen, and Y.-H. Huang, "Energy efficientt $128 \sim 2048/1536-$ point FFT processor with resource block mapping for 3GPP-LTE system," in *Proc. of IEEE Int. Conf. on Green Circuits and Systems (ICGCS)*, Shanghai, 2010, pp. 14–17.

[15] T. Patyk, D. Guevorkian, T. Pitkäanen, P. Jääskeläinen, and J. Takala, "Low-power application-specific FFT processor for LTE applications," in *Proc. of IEEE Int. Conf. on Embedded Comp. Syst.: Arch. Mod. Simul. (SAMOS)*, Agios Konstantinos, Grece, 2013, pp. 28–32.

[16] I. Cho, T. Patyk, D. Guevorkian, J. Takala, and S. Bhattacharyya, "Pipelined FFT for wireless communications supporting 128–2048/1536 -point transforms," in *Proc. of IEEE Global Conf. on Sign. Inf. Proc. (GlobalSIP)*, Austin, TX, 2013, pp. 1242–1245.

[17] C. Yu and M.-H. Yen, "Area-efficient 128- to 2048/1536-point pipeline FFT processor for LTE and mobile WiMAX systems," *IEEE Trans. VLSI Syst.*, vol. 23, no. 9, pp. 1793–1800, Sep. 2015.

[18] J. Chen, J. Hu, S. Lee, and G. E. Sobelman, "Hardware efficient mixed radix-25/16/9 FFT for lte systems," *IEEE Trans. VLSI Syst.*, vol. 23, no. 2, pp. 221–229, Feb. 2015.

[19] J. Takala, D. Akopian, J. Astola, and J. Saarinen, "Constant geometry algorithm for discrete cosine transform," *IEEE Trans. Signal Process.*, vol. 48, no. 6, pp. 1840–1843, Jun. 2000.