Frequenz
60 (2006)
7-8

Faster than the FFT: The chirp-z RAG-n Discrete Fast Fourier Transform*

By Uwe Meyer-Bäse, Hariharan Natarajan, Encarnación Castillo, Antonio García

Abstract – DFT and FFTs are important but resource intensive building blocks and have found many application in communication systems ranging from fast convolution to coding of OFDM signals. It has recently be shown that the n-Dimensional Reduced Adder Graph (RAG-n) technique is beneficially in many applications such as FIR or IIR filters, where multiplier can be grouped in multiplier blocks. This paper explores how the RAG-n technique can be applied to DFT algorithms. A RAG-n fast discrete Fourier transform will be shown to be of low latency and complexity and posses a VLSI attractive regular data flow when implemented with the Bluestein chirp-z algorithm. VHDL code synthesis results for Xilinx Virtex II FPGAs are provided and demonstrate the superior properties when compared with Xilinx FFT IP cores.

Index Terms – Fast Fourier Transform, OFDM, FPGA, n-Dimensional Reduced Adder Graph

1. Introduction

The recently introduced new algorithms like JPEG2000, MPEG, or DVB, require high performance, programmability, and low power in embedded systems. Figure 1 gives an overview of possible implementation options for such systems [1]. The most universal is a pure microprocessor, but usually lacks a sufficient power/performance ratio for embedded applications. The other extreme, a dedicated design, has the highest power/performance ratio, but is very specialized and not universal enough in most cases. An array of microprocessors is another attractive alternative that has been implemented in many variations [2-6], but usually has a long development time and requires sophisticated scheduling and compiler. A powerful microprocessor augmented by a collection of dedicated co-processors seems to be the most universal solution that allows rapid prototyping and provides still enough computing power to fulfill these demanding multimedia processing needs.

Discrete Fourier Transform (DFT) and its fast implementation, the Fast Fourier Transform (FFT), have played a central role in many digital signal processing applications. A possible FFT co-processor configuration of the Xilinx LogiCore FFT IP [7-11] is shown in Fig. 2.

The processor + co-processors system configuration requires that first the (complex) data are transferred to the co-processor, followed by the transform computations, and finally the "unload" phase to apply a bit reverse sort and transfer data back to the main processor memory. Although this Butterfly based co-processors are small in area, (Software code see [12]; HDL code see for instance [13]) the downside of the Butterfly co-processor is a substantial latency, i.e. the time that it takes until the first DFT coefficient is available. There are however other system configurations that do not have the high latency associated with the Butterfly processing scheme for the processor + co-processors system configuration.

A much shorter latency can be achieved when the Rader or Bluestein chirp-z Transform (CZT) are used. Both algorithms translate the DFT equation:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \qquad k = 0,1,.....N-1 \tag{1}$$

$$with \;\; W_N^{kn} = e^{-j2\pi kn/N} \tag{2}$$

into a convolution, i.e. FIR filter operation. The Rader transform has the limitation that only prime length transform can be computed and both, input and output sequences, appear in permutated order. The disadvantage of the CZT algorithms is that compared to the Rader algorithms two additional complex I/O multiplication are required. But any transform length can be built with the CZT algorithm and the following discussing will focus therefore on the CZT transform.

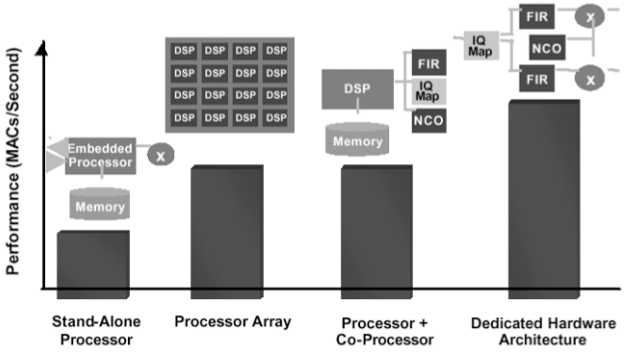

Fig. 1: Data path processing architecture options [1]

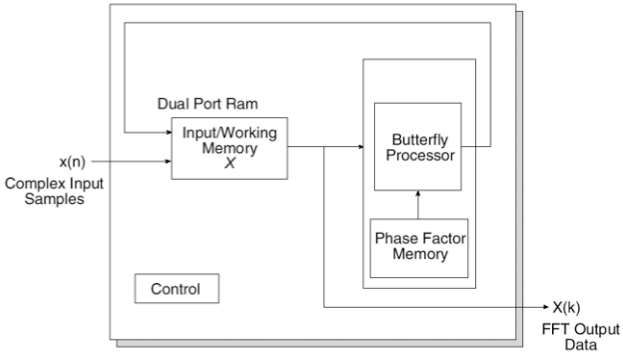

Fig. 2: FFT co-processor core [7]

The FIR part of the Rader and CZT algorithms are very resource demanding, and in the past such implementations were not feasible. Burrus and Parks [2, p.37] for instance claim: "If implemented on digital hardware, the chirp-z transform does not seem advantageous for calculating the normal DFT." With the advent of the multiplier block coding with the reduced adder graph technique, however we can lower the complexity of the CZT that an implementation becomes feasible and advantageous.

The remainder of the paper is organized as follows. We will give first a brief overview of multiplier block coding using CSD, MAG, and RAG-n coding. Section 3 presents the CZT and de-

* FAMU-FSU College of Engineering, Department of Electrical and Computer Engineering, Florida State University, Tallahassee, USA

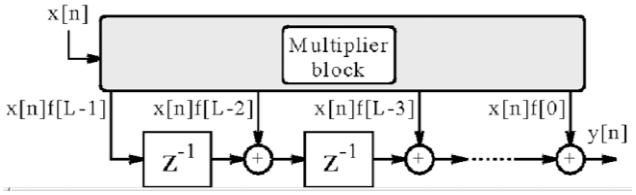

Fig. 3a: FIR filter in the transposed structure

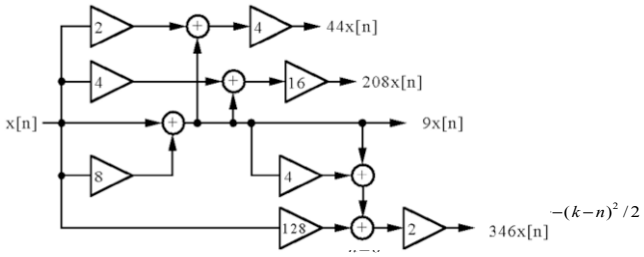

Fig. 3b: Realization of the Goodman and Carey halfband filter F6 using the RAG-n algorithm

scribe then how to implement the CZT DFTs on FPGAs. Finally we present synthesis results for Xilinx FPGA.

2. Multiplier Block Coding

There are only a few applications (e.g., adaptive filters) where a general programmable filter architecture is needed. In many applications, the filters are linear time-invariant (LTI) systems and the coefficients do not change over time. In this case, the hardware effort can essentially be reduced by exploiting the constant coefficient multiplier coding and adder (trees) used to implement the transposed FIR filter multiplier block, see Fig. 3a.

Statistically, half the digits in the two's complement coding of a number are zero. As a result, if a constant coefficient is realized with an array multiplier on average 50% of the partial products are zero. In case the canonic signed digit (CSD) system, i.e., digits can have ternary values $\{-1,0,1\} = \{\bar{1},0,1\}$, the density of the non zero elements becomes 33%. Consider for instance the coding the decimal number $15_{10}=1111_2$ using a 5-bit binary CSD code reduced to $1111_2 = 1000\,\bar{1}_{\mathrm{CSD}}$ which reduces the implementation effort for an array multiplier from 3 adders to one subtraction.

We have just seen that the cost of multiplication is a direct function of the number of nonzero elements in the coefficient coding. The CSD system reduces this cost on average to 33%. It can, however, sometimes be more efficient first to factor the coefficient into several factors, and realize the individual factors in an optimal CSD sense. For instance the direct CSD codes for a multiplier 45 requiring three adders, while realized as multiplier adder graph (MAG), i.e., as factors (3*15=(2+1)*(16-1)) requires one addition and one subtraction. The complexity for the factor implementation is reduced by about 30%.

In several DSP systems we find that many multipliers share the same input and we combine these multipliers in a multiplier block. The transposed FIR filter shown in Figure 3(a) is a typical example for a multiplier block. Dempster and Macleod [15,16] have introduced a systematic algorithms, which produces a n-Dimensional Reduced Adder Graph (RAG-n) of a block multiplier. In general, however, finding the optimal RAG-n is an NP-hard problem. RAG-n determines in the first steps an optimal coding and for the suboptimal part heuristics are applied. The full 10 step RAG-n algorithms can be found in [15]. To illustrate the RAG-n algorithm, consider coding the coefficients defining the F6 halfband FIR filter of Goodman and Carey [13]. The halfband filter F6 has four nonzero coefficients, namely which are 346, 208, -44, and 9. For the direct CSD code realization, 9 adders are required. If the RAG-n algorithms is used the number of adders is reduced from 9 to 5. The adder path delay is also reduced from 4 to 3. Fig. 3b shows the resulting reduced-adder graph.

3. The Bluestein Chirp-z Transform

In the Bluestein chirp-z transform (CZT) algorithm, the DFT exponent *nk* is quadratic expanded to:

$$nk = -(k-n)^2/2 + n^2/2 + k^2/2 \tag{3}$$

The DFT therefore becomes

$$X[k] \;=\; W^{k^2/2}\sum_{n=0}^{N-1}(x[n]W^{n^2/2})\;W^{-(k-n)^2/2} \tag{4}$$

This algorithm is graphically interpreted in Fig. 4. The Bluestein Chirp-z algorithm is computed using the following thee steps:

1. N multiplication of $x[n]$ with $W^{n^2/2}$
2. Linear convolution of $x[n]W^{n^2/2} * W^{-n^2/2}$
3. N multiplications with $W^{k^2/2}$.

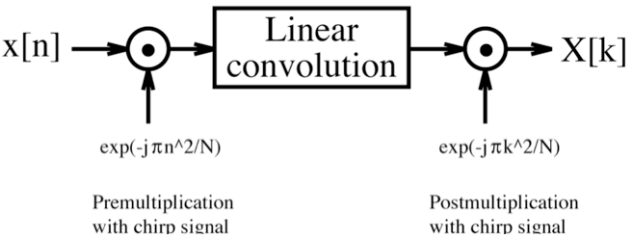

Fig. 4: The Bluestein chirp-z algorithm

For a complete transform, a length N convolution and $2N$ complex multiplications are required. The advantage, compared with the Rader algorithm, is that there is no restriction to primes in the transform length N. CZT can be defined for every length. Narasimha et al. [14] have noticed that in the CZT algorithm many coefficients of the FIR filter part are trivial or identical. For instance, the length-8 CZT has an FIR filter of length 15, i.e.,

$$C(n) = e^{j2\pi\frac{n^2/2 \bmod 8}{8}} \qquad n = 1,2,...,14 \tag{5}$$

but there are only four different complex coefficients as graphically interpreted in Fig. 5. These four coefficients are 1, j, and $\pm e^{j22.5^\circ}$, i.e., we have only two nontrivial real coefficients (cos(22.5°) and sin(22.5°)) to implement.

The maximum DFT length N for a fixed number C_N of (complex) constant coefficients may be of general interest. This is shown in following table.

DFT length	8	12	16	24	40	48
C_N	4	6	7	8	12	14
DFT length	72	80	120	144	168	180
C_N	16	21	24	28	32	36

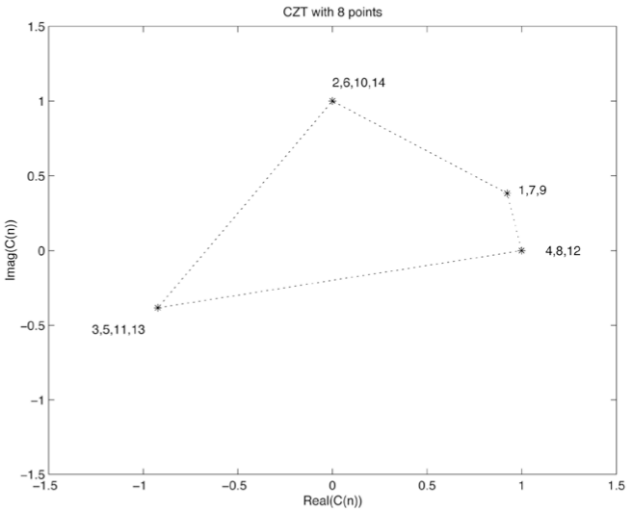

Fig. 5: The 14 complex coefficients C(n) for a length 8 CZT DFT

For comparison purpose consider the number of complex multiplier necessary to build a parallel radix-2 FFT architecture. A column processor [18] will need N/2 variable (complex) multiplier, i.e. no optimization for constant coefficient can be applied. A full pipelined architecture will require $\log_2(N)N/2$ constant multiplier.

As mentioned before, the number of different complex coefficients does not directly correspond to the implementation effort, because some coefficients may be trivial (i.e., +/-1 or +/-j) or may show symmetry. In particular, the power-of-two length transforms enjoy many symmetries. If we compute the maximum DFT length for a specific number of nontrivial real coefficients, we find as maximum length transforms to be:

DFT length	10	16	20	32	40	48
R_N	2	3	5	6	8	9
DFT length	50	80	96	160	192	
R_N	10	11	14	20	25	

Length 16 and 32 are therefore the maximum length DFTs with only 3 and 6 real multipliers, respectively. In general, power-of-two lengths are popular building blocks for Cooley-Tukey FFTs, therefore we implemented CZT transforms of length $N=2^n$. The number of adders required for the CSD, MAG and RAG-n methods for the CZT DFT up to length 70 is graphically interpreted in Figure 6(a). For each DFT length the three different implementation efforts symbolized via (o, x, and *) are shows. It can be seen that the effort depends strongly on the symmetry of the coefficients and we may therefore refrain from implementing certain DFT length and use a little larger DFT. These more efficient DFTs with maximum transform length are connected through an additional solid line .

3.1 Complex Bluestein Chirp-z DFT

So far we have discussed CZT DFTs of a real input sequences and the complex twiddle factor multiplication can then be realized with two real multiplications. For complex input DFTs we have two choices how to implement the complex multiplication. We may use the straight forward approach with 4 real multiplications and 2 real additions, i.e.,

$$(a+jb)(c+js) = ac - bc + j(as + bc) \quad (6)$$

or we may use a different factorization:

$$s[1] = a - b \quad s[2] = c - s \quad s[3] = c + s$$

$$m[1] = s[1] \times s \quad m[2] = s[2] \times a \quad m[3] = s[3] \times b$$

$$s[4] = m[1] + m[2] \quad s[5] = m[1] + m[3]$$

$$(a+jb)(c+js) = s[4] + js[5] \quad (7)$$

which requires 3 real multiplications and 5 real additions.

Figure 6 (b) shows for the complex FIR part of the chirp-z transform the number of required adder for transforms up to length 70. Although the results vary for different transform length, the connection of the maximum length transform through the solid line shows, that the 4*2+ algorithm is superior when compared with the 3*5+ algorithms. This is due to the fact that with the 4*2+ algorithms for a complex coefficient filter of length L two multiplier blocks with size $2L$ are designed, while for the 3*5+ algorithms three multiplier block filters with block size L have to be used, where $L=2N-1$. In a larger multiplier block RAG-n optimizes much better than in a shorter filter. We used therefore the 4*2+ algorithm to implement the FIR filter part of the CZT (see Fig. 4), while the (3*5+) algorithm was used to implement the two complex input and output multiplication of the CZT.

4. Synthesis Results

The CZT architecture shown in Fig. 4 was modified such that only one complex multiplier is shared by the I/O multiplications. This does not reduce the registered performance because after all input values are received, input complex multiplications are no longer required and the multiplier can be used for the output multiplications. This modification saves many gates because the 3x5+ complex multiplier requires already 12963 gates, or over 50% for a length 4 transform.

The circuits for length 4, 8, 16, 32, and 64 point CZT DFTs have been developed using generic VHDL coding. The VHDL code of these CZT designs [17] are available online via the FSU

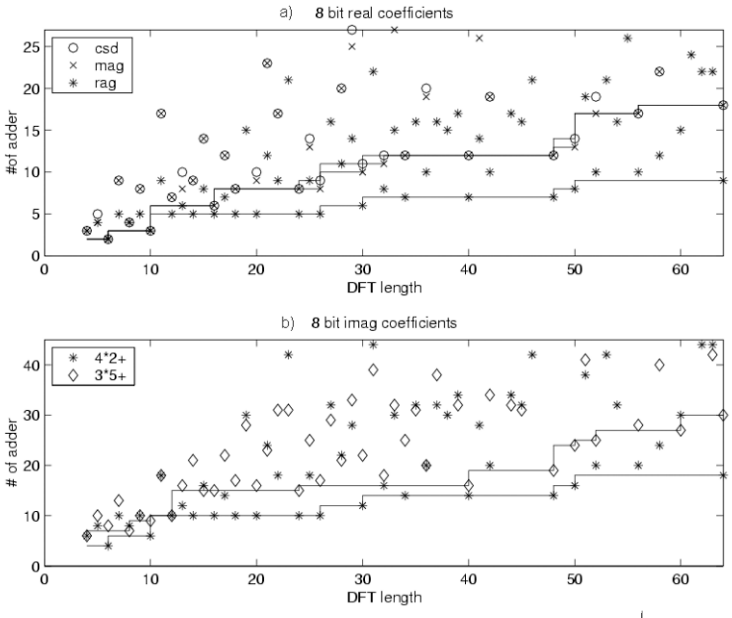

Fig. 6: Bluestein chip-z transform implementation effort. (a) Number of adders for all real coefficient of DFT length N. (b) Comparision of two different complex multiplier designs

library at www.lib.fsu.edu. The results are reported in Table 1. Circuits have then been synthesized from their VHDL descriptions and optimized for speed using ISE2 synthesis tools from Xilinx. We found that with optimization goal "size" the registered performance and area do not change significant for this adder intensive circuit type [17]. The device for all designs was a 2v1000fg256-6 from Xilinx Virtex II device family. Because the Xilinx synthesis tools used embedded array multiplier, logic cells with two 4-input and one 3-input tables, and block RAMs, the best way to measurement the design area is the equivalent number of gates (called Gates in Table 1+2) from the Xilinx "Mapping Report File." To have reliable timing data we use the "Post Place & Route Static Timing Report" rather than the map time estimations.

Table 1: CZT DFT synthesis results for 2v1000fg256-6 Xilinx Virtex II device.

DFT length	4	8	16	32	64
Gates	21K	32K	52K	90K	154K
#4 LUT (10240)	644 (6%)	1257 (12%)	2663 (26%)	4864 (47%)	8809 (86%)
Embedded multiplier (40)	3 (15%)	3 (15%)	3 (15%)	3 (15%)	3 (15%)
Number I/O (172)	119	121	121	127	122
Peak memory usage (MB)	87	94	112	139	163
Registered Perform-ance (ns)	14.44	15.263	15.303	15.702	20.279

We have used four additional pipeline stages in our design: two for the complex multiplier and two for the RAG-n FIR. The total latency, i.e. the time between all input vectors are transferred and the first output DFT values is available, is therefore 4 clock cycles. In contrast the netlist optimized Xilinx LogiCore IP block that uses the Butterfly architecture shown in Fig. 2 requires from 66 to 3121 clock cycles until the first valid DFT value is available. Different results and less optimal resource utilization is expected if we use the pameterizable FFT cores available also from Xilinx.

Table 2: Comparison of CZT and Xilinx FFT IP core latency and gate count.

DFT length	16	32	64	256	1024
CZT latency	4	4	4	4	4
Xilinx latency	66	132	82	561	3121
Slices	1386	908	1161	1616	1869
Gates	429K	159K	167K	181K	189K
FFT radix	4	2	4	4	4

It can also be seen from Table 2 that the latency depends on the radix of the Butterfly. A larger radix allows the computation in less clock cycles, but is still essentially larger than the CZT solution. In addition a large radix reduces the number of possible transforms. A radix-R FFT can only implement length R^k FFTs, i.e., for a radix-4 FFT possible transform length are 4, 16, 64, etc. but 32, 128, 512, etc. point FFTs can not be build with a radix-4 FFT.

We notice also from Table 2 that the 16 point Xilinx LogicCore FFT is implemented with logic cells only, and the slice count is therefore larger as expected. For length larger than 16 two RAM block are required as working memory. Each of the block RAMs requires about 65K equivalent gates.

It can also be seen from Table 1 that the registered performance of the 64-point DFT decreases due to the fact that the device is 86% full.

5. Conclusion

The paper shows that Bluestein Chirp-z DFT has become a viable implementation path for fast discrete Fourier Transform designs when the multiplier block is implemented with reduced adder graph technique. The paper has demonstrated the dramatic reduction in effort the RAG-n algorithm has, when implementing this DFT real or complex constant coefficient filter. For instance the length 63 TAP complex filter in 8 bit (plus sign) precision used in the 32 point Bluestein Chirp-z DFT needs only 12 real multipliers that can be implemented with 24 adders compared with the worst case of $(2\times32-1)\times8\times4 = 2016$ adders of a direct realizations.

Synthesis results for 5 VHDL design examples show that up to length 64 CZT DFT transforms fit in a 2v1000fg256-6 device and all design have a latency of only 4 clock cycles compared with 84 clock cycles of the equivalent 64 point Xilinx IP core. Gates count for Butterfly and CZT designs are similar, which CZT gate count design a little smaller.

Further investigations will focus on cell-based VLSI design of the Bluestein Chirp-z DFT algorithms and building large length FFTs (N>1024) with Bluestein DFT as basic building block. Another investigation may also include a size/speed comparison of the Bluestein DFTs with the Winograd DFT blocks.

The authors would like to thank Xilinx for their support under the University programs. A. García and E. Castillo were supported by the Ministerio de Ciencia y Tecnología (MCyT, Spain) under project TIC2002-02227.

References

[1] Altera Corporation, "DSP Netseminar series: Video & Image Processing with FPGAs," *http://www.altera.com*, 2004.

[2] F. Barat, R. Lauwereins, G. Deconinck, "Reconfigurable Instruction Set Processors from a Hardware/Software Perspective", *IEEE Transactions on Software Engineering*, vol. 28(9), pp. 847-862, 2002.

[3] T. Miyamori, K. Olukotun, "REMARC: Reconfigurable Multimedia Array Coprocessor", *Proc. Sixth Int'l Symp. Field-Programmable Gate Arrays*, 1998.

[4] S. Hauck, T.W. Fry, M.M. Hosler, J.P. Kao, "The Chimaera Reconfigurable Functional Unit", *Proc. Workshop FPGAs and Custom Computing Machines* 1997.

[5] P. Kievits, E. Lambers, C. Moerman, R. Woudsma, "R.E.A.L. DSP Technology for Telecom Baseband Processing", *ICSPAT* 1998.

[6] G. Cichon, P. Robelly, H. Seidel, T. Limberg, G. Fettweis, "SAMIRA: A SIMD-DSP architecture targeted to the Matlab™ source language", *Proceedings GSPX*, 2004.

[7] Xilinx Corporation, "High-Performance 16-Point Complex FFT/IFFT", *Product Specification*, 2000.

[8] Xilinx Corporation, "High-Performance 32-Point Complex FFT/IFFT", Product Specification, 2002.

[9] Xilinx Corporation, "High-Performance 64-Point Complex FFT/IFFT", Product Specification, 2000.

[10] Xilinx Corporation, "High-Performance 256-Point Complex FFT/IFFT", Product Specification, 2000.

[11] Xilinx Corporation, "High-Performance 1024-Point Complex FFT/IFFT", Product Specification, 2000.

[12] C. Burrus, T. Parks, *DFT/FFT and Convolution Algorithms*, John Wiley & Sons, New York, 1985.
[13] U. Meyer-Baese, *Digital Signal Processing with Field Programmable Gate Arrays.* Springer-Verlag, Berlin Heidelberg New York, 2nd ed., 2004.
[14] M. Narasimha, K. Shenoi, A. Peterson, "Quadratic residues: Application to chirp filters and discrete Fourier transforms", *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 12-24, 1976.
[15] A. Dempster, M. Macleod, "Use of minimum-adder multiplier blocks in fir digital filters," *IEEE Transactions on Circuits and Systems II*, vol. 42, pp. 569-577, 1995.
[16] A. Dempster, M. Macleod, "Comments on 'minimum number of adders for implementing a multiplier and its application to the design of multiplierless digital filters", *IEEE Transactions on Circuits and Systems II*, pp. 242-243, 1998.
[17] H. Natarajan, "Implementation of Chip-z Discrete Fourier Transform on Virtex II FPGA," Master's Thesis, Florida State University, Tallahassee, 2004.
[18] S. Gorman and J. Wills, "Partial Column FFT Pipelines," IEEE Transactions on Circuits and Systems-II. vol. 42, pp. 414-423, 1995

Assistant Professor Dr. Uwe Meyer-Bäse
and H. Natarajan
Department of Electrical and Computer Engineering
Tallahassee, Florida 32310-6046,
USA
Fax: +1(850)410-6479
E-mail: umb@eng.fsu.edu; natara@eng.fsu.edu

Encarnacion Castillo and Prof. Dr. Antonio García
Dept. of Electronics and Computer Technology
University of Granada
18071 Granada
Spain
Fax: +34-958243230
E-mail: encas@ditec.ugr.es; agarcia@ditec.ugr.es

(Received on December 27, 2005)
(Revised on March 20, 2006)