IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p- ISSN: 2278-8735
PP 63-69
www.iosrjournals.org

Memory Efficient Fft Computation With Error Tolerance Sdc-Sdf Architecture

M. Suresh[1], J. Dhanapathi [2]
PG Student, VLSI Design, Department of ECE, Surya group of institutions, India[1]
Assistant professor (Sr.G.), Department of ECE, Surya group of institutions,, India[2]

Abstract: *The appearance of radix-2^2 was a milestone in the design of pipelined FFT hardware architectures. However, radix-2^2 was only proposed for single-path delay feedback (SDF) architectures. In this paper we presents the radix-2^2 DIF with combined single-path delay commutator-feedback (SDC-SDF) pipelined fast Fourier transform architecture. In feed forward architectures FFT radix can be used for any number of parallel samples which is a power of two. Furthermore, both decimation in frequency (DIF) and decimation in time (DIT) decompositions can be used. In this paper, an efficient VLSI architecture of a pipeline fast Fourier transform (FFT) processor with 100% hardware utilization and 50% normal output order sequence is presented. The proposed pipelined fast Fourier transform architecture, which includes log2 N − 1 SDC stages, and 1 SDF stage. The low complexity is achieved by sharing the common arithmetic for 100% resource hardware resource utilization in the time-multiplexed approach, including both adders and multipliers. Finally complexity reduction is proved and functionality is verified and compared with existing methods.*

.

I. Introduction

The Discrete Fourier transform (DFT) is obtained by decomposing a sequence of values into components of different frequencies. The Fast Fourier transforms (FFTs) are the efficient algorithms to compute the DFT [1]. The FFT algorithms are based on the principle of decomposing the computation of DFT into sequences of smaller DFTs. This operation is useful in many fields but computing it directly from the definition is often too slow to be practical. The FFT is used in various applications where the frequency-domain representation of a signal has to be analyzed. In the communications area, the FFT has gained attention because of its use in orthogonal frequency division multiplexing (OFDM) systems [1]. For OFDM receivers, a FFT processing block is required. Several communication systems require medium resolution (9–12 bits) analog-to-digital converters with bandwidths in the tens of MHz range. The applications that use the FFT impose challenging specifications for its processing, such as small silicon area, high throughput, short processing time and reduced power consumption. For these applications, pipeline FFT architectures are accurate. FFT has applications in mixed radix system, one of the popular numerical systems in which FFT numerical base or radix varies from one position to another position. FFT is the most popular digital spectrum analysis technique. The DFT is one of the fundamental operations in digital signal processing. The original computation of DFT with sample input requires complex multiplications. Cooley and Tukey first introduced the concept of FFT to demonstrate a significant computational reduction from to by making efficient use of symmetry and periodicity properties of the twiddle factors. The related algorithms for the computation of the DFT are generally known as the FFTs. DFT and FFT are very popular signal processing tools. An FFT computes the DFT and produces exactly the same result as evaluating the DFT definition. Computing the DFT of N points in the naive way using the definition takes O(N2) arithmetical operations while a FFT can compute the same DFT in only O(N log N) operations. FFT module can be designed for the receiver and can be used for the transmitter IFFT with external conjugation either in hardware or software. The discrete Hartley transform (DHT) is widely used in signal and image processing applications. The advantage of the DHT over the DFT is that it can be used to avoid complex operations when the input sequence is real. The forward and inverse DHTs [3] differ from each other in their form only in the scaling factor. The decimation in-time (DIT) and the decimation in- frequency (DIF) algorithms [4] are the typical forms of the FFT algorithm.

Let us consider a ring R with primitive nth root of unity ω where n =pk. Suppose that we wish to evaluate a polynomial f ∈ R[x] of degree less than n at n points and the particular set of points used for the multipoint evaluation is not important. In this case, the number of operations needed to compute the multipoint evaluation can be significantly reduced if f is evaluated at each of the powers of ω, i.e. {f(1), f(ω), f(ω2), f(ω3), . . . , f(ωn−1)}. Each of the n points used for this computation is a root of xn − 1. In [1], Gao calls this operation the “multiplicative FFT” to distinguish it from the operation that will be discussed in Chapter 3. For the remainder of this chapter, it is to be understood that “FFT” will refer to this multiplicative FFT. Nearly every presentation and shows how these computations can be completed using a “divide-and-conquer” approach involving the factorization of this matrix. In this chapter, we will instead view the FFT as a special case of the multipoint evaluation algorithm discussed. In this chapter, we will first present two types of algorithms that can compute an FFT when p = 2 using algebraic descriptions of these algorithms found in [2]. Next, we will give algorithms with lower operation counts in the case where multiplication by certain roots of unity can be computed more efficiently than others. Finally, we will present multiplicative FFT algorithms that can be used when p = 3. With this background, the reader can develop FFT algorithms for other values of p if desired.

II. Related Work

A novel pipelined fast Fourier transform (FFT) architecture which is capable of producing the output sequence in normal order. A single-path delay commutator processing element (SDC PE) [5] has been proposed for the first time. It save a complex adder compared with the typical radix-2 butterfly unit. The new pipelined architecture can be built using the proposed processing element. The proposed architecture can lead to 100% hardware utilization and 50% reduction in the overall number of adders required in the conventional pipelined FFT design. In order to produce the output sequence in normal order, we also present a bit reverser, which can achieve a 50% reduction in memory stage. Multipath-path delay commutator [3] structures are utilized to improve the throughput rate of radix-2 and radix-4 FFT [2] computation by a factor of 2 to 4. Latency can be reduced by a factor of 2 to 3. Compared with previous radix-2 and radix-4 FFT structures, the proposed high-throughput FFT with doubled throughput rate requires similar or even less hardware cost. Although split radix FFT design is more hardware efficient, the regular structure of proposed FFT structure are attractive for high throughput FFT design. An efficient VLSI architecture of a pipeline fast Fourier transform (FFT) processor capable of producing the normal output order sequence is presented. A new FFT design based on the decimated dual-path delay feed-forward data commutator [2] unit by splitting the input stream into two half-word streams is first proposed. The resulting architecture can be achieve full hardware efficiency such that the required number of adders can be reduced by half. Next, in order to generate the normal output order sequence, this paper also presents a sequence conversion method by integrating the conversion function into the last data commutator module.

III. Low-Power Fft Design Technique

a) Radix–2 DIT FFT algorithm

With the introduction of field programmable gate arrays (FPGAs), it is feasible to provide hardware for application specific computation design. The changes in designs in FPGA’s 5] can be accomplished within a few hours, and thus result in significant savings in cost and design cycle. FPGAs offer speed comparable to dedicated and fixed hardware systems for parallel algorithm. The radix-2 decimation in time is applied recursively to the two lengths N/2 DFT’s to save computation time. The full radix-2 decimation-in time of length 8-signals is illustrated in fig-1, using the simplified butterflies. It involves M = log2N stages, each with N/2 butterflies per stage. Each butterfly requires 1 complex multiplier [3] and two adder per butterfly. The total cost of the algorithm is thus computational cost of radix-2 DIT FFT .- N/2log2N complex multipliers, Nlog2N complex adders.

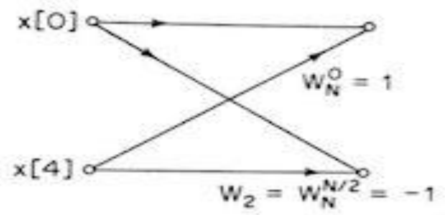


Fig 1 Butterfly for radix-2 DIT FFT

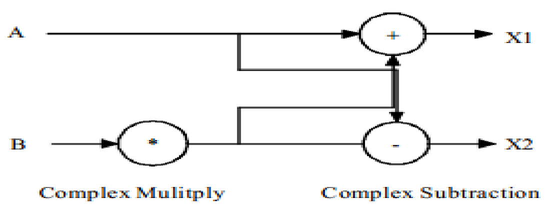


Fig 2 Block diagram of Radix-2 FFT

b) Radix -2^k algorithm

The N-point DFT is formulated as

$$X(k)=\sum_{n=0}^{N-1} x(n)W_N^{nk}, k=0,1,\ldots N-1 \quad (1)$$

Where the twiddle factors are defined as $W_N^{nk}=e^{-\frac{2\pi nk}{N}}$ .The n denotes the time index and the k denotes the frequency index. The radix 2^k algorithm can be derived by integrating twiddle factor decomposition through a divide and conquer approach.

c) Radix -2^2 algorithm

Consider the first two steps of decomposition in radix-2 DIF FFT together. Applying a 3-dimensional linear index map as follows

$$n=\frac{N}{2}n_1+\frac{N}{4}n_2+n_3\{n_1,n_2=0,1n_3=0\sim\frac{N}{4}-1\}$$

$$k=k_1+2k_2+4k_3\{k_1,k_2=0,1k_3=0\sim\frac{N}{4}-1\} \quad (2)$$

The DFT has the form of

$$X(k_1+2k_2+4k_3)=\sum_{n_3=0}^{\frac{N}{4}-1}\sum_{n_2=0}^{1}\sum_{n_1=0}^{1}x(\frac{N}{2}n_1+\frac{N}{4}n_2+n_3)W_N^{nk}$$

$$=\sum_{n_3=0}^{\frac{N}{4}-1}\sum_{n_2=0}^{1}\{B_{\frac{N}{2}}^{k_1}(\frac{N}{4}n_2+n_3)\}W_N^{(\frac{N}{4}n_2+n_3)(k_1+2k_2+4k_3)} \quad (3)$$

where the first butterfly structure has the form of

$$B_{\frac{N}{2}}^{k_1}(\frac{N}{4}n_2+n_3)=x(\frac{N}{4}n_2+n_3)+(-1)^{k_1}x(\frac{N}{4}n_2+n_3+\frac{N}{2}) \quad (4)$$

Decomposing the composite twiddle factor,it can be expressed in Eq.(5).

$$W_N^{(\frac{N}{4}n_2+n_3)(k_1+2k_2+4k_3)}=(-j)^{n_2(k_1+2k_2)}W_N^{n_3(k_1+2k_2)}W_{N/4}^{n_3k_3} \quad (5)$$

Substituting the Eq.(5) into Eq.(3) and expanding the summation with regard to index n_2 ,we have a set of 4 DFT of length $N/4$.

$$X(k_1+2k_2+4k_3)=\sum_{n_3=0}^{\frac{N}{4}-1}[H_{N/4}^{k_1k_2}(n_3)W_N^{n_3(k_1+2k_2)}]W_{N/4}^{n_3k_3} \quad (6)$$

where a secondary butterfly structure $H_{N/4}^{k_1k_2}(n_3)$ is expressed as

$$H_{N/4}^{k_1k_2}(n_3)=B_{\frac{N}{2}}^{k_1}(n_3)+(-1)^{k_2}(-j)^{k_1}B_{\frac{N}{2}}^{k_1}(n_3+\frac{N}{4}) \quad (7)$$

After these two columns, full multiplications are used to apply the decomposed twiddle factor $W_N^{n_3(k_1+2k_2)}$ in Eq.(6).Applying this cascade decomposition recursively to the remaining DFTs of length $N/4$ in Eq.(6), the complete radix -2^2 FFT algorithm is obtained. Equation (7) represents the first two columns of butterflies with only trivial multiplication of (-j) which can be implemented using only real-imaginary swapping and sign inversion.The radix-2^2 algorithm is characterized according to the merit that it has the same multiplicative complexity and as the radix-4 algorithm, but still retains simple structures of the radix-2 butterfly.

d) Single-path delay commuator processing The SDC PE, consists of a data commutator, a real add/sub unit, and an optimum complex multiplier unit In order to minimize the arithmetic resource of the SDC PE, the most significant factor is to maximize the arithmetic resource utilization via reordering the data sequences of the above three units. In the stage t, the data commutator shuffles its input data (Node−A) to generate a new data sequence (Node−B), whose index difference is N/2t, where t is the index of stage. The new data sequence (Node−B) is critical to the real add/sub unit, where one real adder and one real subtracted can both operate on two elements for each input data. The sum and difference results (Node−C) overlap the places of the two input elements. Therefore, it preserves the data sequence, requires only one real adder and one real subtracted. For the optimum complex multiplier unit[5], its output data sequence (Node−E) should be the same as its input data sequence (Node−C). If so, its output sequence (Node−E), which is also the output sequence of the SDC stage t, can become the direct input data sequence (Node−A) of the SDC stage t+1.

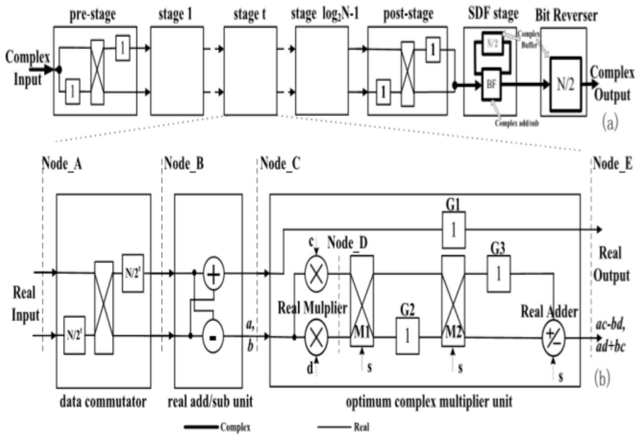


Fig 3 Block diagram of the proposed FFT architecture.

As it contains 2 multiplexers (M1 and M2), 1.5-word memory (G1, G2, andG3), 2 Real Multipliers and 1 Real Adder. The signal s controls the behavior of two multiplexers (M1 and M2): through or swap. The signal s also controls the behavior of the Real Adder, which supports both addition and subtraction operations. For the input couple (0_r,8 _r) and (0 _i,8 _i) at the Node−C in Table II the sum part data 0 _r and 0_i will directly pass to the delay memory G1 to generate 0_r* and 0_i* with one cycle delay in consecutive two cycles, while the difference part 8_r and 8_i will directly enter the Real Multipliers (Node−D) to generate (c × 8_r d × 8_r) and (c × 8_i, d × 8_i) before reordering. The reordering process is performed as follows.

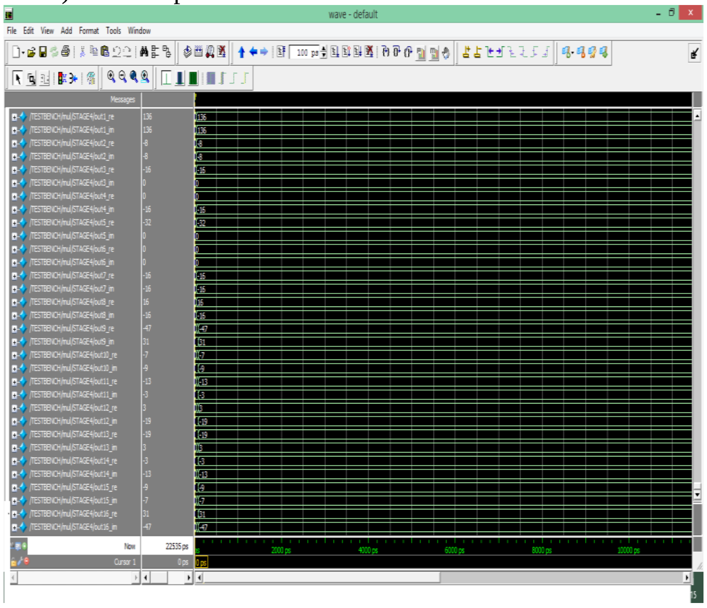
1) In the first cycle, when 8_r comes, the signal s (s =1) selects “through”; that is, the up (down) input of the multiplexer (M1 or M2) connects to the up (down) output. Then, the G2 (or G3) would be d × 8_r (or c × 8_r) in the second cycle.

2) In the second cycle, when 8_i comes, the signal s (s = 0) selects “swap”; that is, the up (down) input of the multiplexer (M1 or M2) connects to the down (up) output. Then, the G2 (or G3) would be c ×8_i (or d × 8_r) in the third cycle. The s will make the Real Adder perform subtraction operation and then c× 8_ r_d × 8_i (8_r*) would appear at the Node−E.

3) In the third cycle, the signal s (s = 1) selects “through” for M1 and M2, and chooses addition operation for Real Adder. Then, d × 8_r+c × 8_i (8_i*) would appear at the Node−E.

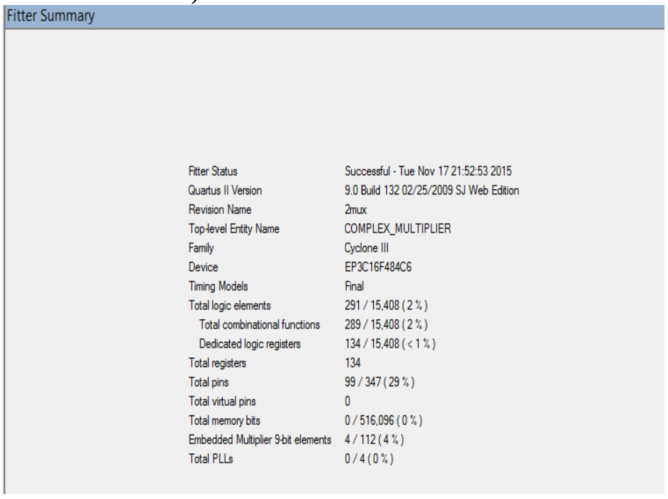
Iv. Software Implementation Results

a) 2 multiplier functional verification in Modelsim



```
/TESTBENCH/mul/STAGE4/out1_re    136
/TESTBENCH/mul/STAGE4/out1_im    136
/TESTBENCH/mul/STAGE4/out2_re    -8
/TESTBENCH/mul/STAGE4/out2_im    -8
/TESTBENCH/mul/STAGE4/out3_re    -16
/TESTBENCH/mul/STAGE4/out3_im    0
/TESTBENCH/mul/STAGE4/out4_re    0
/TESTBENCH/mul/STAGE4/out4_im    -16
/TESTBENCH/mul/STAGE4/out5_re    -32
/TESTBENCH/mul/STAGE4/out5_im    0
/TESTBENCH/mul/STAGE4/out6_re    0
/TESTBENCH/mul/STAGE4/out6_im    0
/TESTBENCH/mul/STAGE4/out7_re    -16
/TESTBENCH/mul/STAGE4/out7_im    -16
/TESTBENCH/mul/STAGE4/out8_re    16
/TESTBENCH/mul/STAGE4/out8_im    -16
/TESTBENCH/mul/STAGE4/out9_re    -47
/TESTBENCH/mul/STAGE4/out9_im    31
/TESTBENCH/mul/STAGE4/out10_re   -7
/TESTBENCH/mul/STAGE4/out10_im   -9
/TESTBENCH/mul/STAGE4/out11_re   -13
/TESTBENCH/mul/STAGE4/out11_im   -3
/TESTBENCH/mul/STAGE4/out12_re   3
/TESTBENCH/mul/STAGE4/out12_im   -19
/TESTBENCH/mul/STAGE4/out13_re   -19
/TESTBENCH/mul/STAGE4/out13_im   3
/TESTBENCH/mul/STAGE4/out14_re   -3
/TESTBENCH/mul/STAGE4/out14_im   -13
/TESTBENCH/mul/STAGE4/out15_re   -9
/TESTBENCH/mul/STAGE4/out15_im   -7
/TESTBENCH/mul/STAGE4/out16_re   31
/TESTBENCH/mul/STAGE4/out16_im   -47
Now                              22535 ps
Cursor 1                         0 ps
```

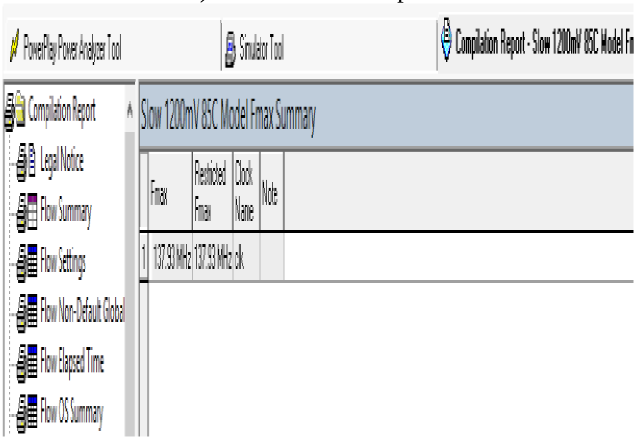
b) Performance of area



Fitter Summary

Fitter Status	Successful - Tue Nov 17 21:52:53 2015
Quartus II Version	9.0 Build 132 02/25/2009 SJ Web Edition
Revision Name	2mux
Top-level Entity Name	COMPLEX_MULTIPLIER
Family	Cyclone III
Device	EP3C16F484C6
Timing Models	Final
Total logic elements	291 / 15,408 (2 %)
Total combinational functions	289 / 15,408 (2 %)
Dedicated logic registers	134 / 15,408 (< 1 %)
Total registers	134
Total pins	99 / 347 (29 %)
Total virtual pins	0
Total memory bits	0 / 516,096 (0 %)
Embedded Multiplier 9-bit elements	4 / 112 (4 %)
Total PLLs	0 / 4 (0 %)

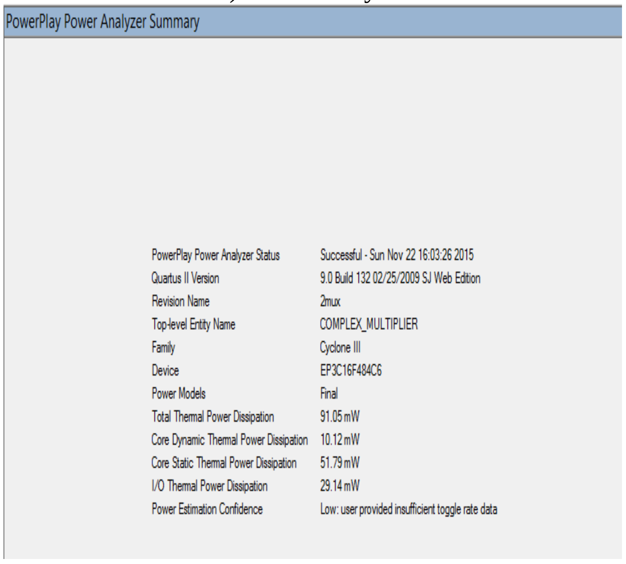
c) Performance of speed



Slow 1200mV 85C Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	137.93 MHz	137.93 MHz	clk	

d) Power analyzer



PowerPlay Power Analyzer Status	Successful - Sun Nov 22 16:03:26 2015
Quartus II Version	9.0 Build 132 02/25/2009 SJ Web Edition
Revision Name	2mux
Top-level Entity Name	COMPLEX_MULTIPLIER
Family	Cyclone III
Device	EP3C16F484C6
Power Models	Final
Total Thermal Power Dissipation	91.05 mW
Core Dynamic Thermal Power Dissipation	10.12 mW
Core Static Thermal Power Dissipation	51.79 mW
I/O Thermal Power Dissipation	29.14 mW
Power Estimation Confidence	Low: user provided insufficient toggle rate data

IV. TABLE

Comparison table for Multiplier, Area, Power, Speed**.**
Table 5.1

TYPE	MULTIPLIER	AREA	POWER	SPEED
4MUL BASED	8	453	98.16mW	131.94MHz
3MUL BASED	6	459	96.66mW	134.1MHz
PROPOSED	4	291	91.05mW	137.91MHz

IV. Conclusion

In this paper, a radix -2^2 algorithm and 16 point combined SDC-SDF pipelined FFT architecture radix -2^2 FFT architecture have been proposed for OFDM-based WPAN applications. The number of complex multipliers and twiddle factor LUTs are reduced using pre-shuffling units in the radix -2^2 algorithm. The proposed radix -2^2 FFT processor is extended into accurate FFT architecture for the 16-point SDC-SDF FFT processors. The proposed architecture has potential applications in high-rate OFDM-based WPAN systems.

References

[1]. M. Ayinala ,M. Brown , and K. Parhi , “Pipelined parallel FFT architectures via folding transformation,” IEEE Trans. Very Large Scale Inegr.(VLSI) Syst., vol. 20, no. 6, pp. 1068–1081, Jun. 2012.

[2]. P.Brent, Richard , G. Fred. Gustavson, and Y.David ,Y. Yun. Fast Solution of To eplitz Systems of Equations and Computation of Pade Approximants. Journal of Algorithms, 1: 259-295, 1980.

[3]. Brigham, E. Oran. The Fast Fourier Transform and its Applications, Prentice Hall (1988).

[4]. Buneman, Oscar. Inversion of the Helmholtz (or Laplace-Poisson) Operator for Slab Geometry, Journal of Computational Physics, 12: 124-30, 1973

[5]. Burden, L. Richard and J. Douglas Faires. Numerical Analysis, Fifth Edition, PWS Publishing Company (1993).

[6]. Cantor, G. David . On arithmetical algorithms over finite fields, J. Combinatorial Theory, Series A, 50(2): 285-300, 1989.