

A Parallel VLSI Architecture for a Digital Filter of Arbitrary Length Using Fermat Number Transforms¹

T. K. Truong

Communications Systems Research Section

I. S. Reed, C. S. Yeh, and H. M. Shao

University of Southern California

In this paper a parallel architecture is developed to compute the linear convolution of two sequences of arbitrary lengths using the Fermat number transform (FNT). In particular a pipeline structure is designed to compute a 128-point FNT. In this FNT, only additions and bit rotations are required. A standard barrel shifter circuit is modified so that it performs the required bit rotation operation.

The overlap-save method is generalized for the FNT to compute a linear convolution of arbitrary length. A parallel architecture is developed to realize this type of overlap-save method using one FNT and several inverse FNTs of 128 points. The generalized overlap-save method alleviates the usual dynamic range limitation in FNTs of long transform lengths. Its architecture is regular, simple, and expandable, and therefore naturally suitable for VLSI implementation.

I. Introduction

Fermat number transforms (FNTs) were developed to compute cyclic convolutions (Refs. 1-3). A cyclic convolution of two sequences can be obtained by taking the inverse FNT of the product of the FNTs of these two sequences.

FNTs over certain transform lengths have the advantage over most number-theoretic transforms in that no multiplications are required. McClelland (Ref. 4) designed a hardware system to realize a 64-point 17-bit FNT that used commercially available ECL IC chips. For this purpose he developed a

new binary number representation and the binary arithmetic operations modulo a Fermat number (Refs. 4, 5). The Fermat number transform can be applied to digital filtering (Refs. 2, 3), image processing (Refs. 6, 7), X-ray reconstruction (Ref. 8), and to the encoding and decoding of certain Reed-Solomon codes (Refs. 9, 10).

In this paper, a parallel architecture is designed to realize a digital filter of arbitrary length using the FNT. In Section II, a pipeline structure is used to compute a 128-point FNT. Only additions and bit rotations are required in this structure. The bit rotation operations are implemented by a modification of a standard barrel shifter circuit (Ref. 11). In Section III, the overlap-save method is generalized to compute the linear convolution of a digital filtering system. Then a parallel architecture is designed to realize the generalized overlap-save

¹This work was supported in part by the JPL Director's Discretionary Fund, FY82.

method using one FNT and several inverse FNTs of 128 points. The circuit design of an FNT butterfly is given in the Appendix.

II. A Parallel Structure for Computing a 128-Point FNT

Let $F_t = 2^{2^t} + 1$ be the t th Fermat number where $t \geq 0$. F_t is a prime number for $0 \leq t \leq 4$. Let $\{x_n\}$ be a N -point sequence of integer numbers, where $0 \leq x_n \leq F_t - 1$, $0 \leq n \leq N - 1$, and N is a power of 2. The Fermat number transform $\{X_k\}$ of $\{x_n\}$ over F_t is defined as follows:

$$X_k \equiv \sum_{n=0}^{N-1} x_n \alpha^{nk} \pmod{F_t}, \quad k = 0, 1, \dots, N-1 \quad (1)$$

where $0 \leq X_k \leq F_t - 1$ and α is an N th root of unity. That is, N is the least positive integer such that $\alpha^N \equiv 1 \pmod{F_t}$. The corresponding inverse FNT is the following:

$$x_n \equiv \left(\frac{1}{N}\right) \sum_{k=0}^{N-1} X_k \alpha^{nk} \pmod{F_t}, \quad n = 0, 1, \dots, N-1 \quad (2)$$

In order that a cyclic convolution can be computed by the FNT pair in Eqs. (1) and (2), N depends on the F_t and α chosen (Refs. 2, 3). More details of an FNT can be found in (Refs. 2 and 3).

In this paper F_t , α , and N are selected specifically to be $F_5 = 2^{32} + 1$, $\sqrt{2}$, and 128 respectively. That is, the data of this FNT are integers between 0 and 2^{32} . Hence 33 bits are required to represent a number. The transform length of this FNT is 128. In an FNT over F_t , the quantity $\sqrt{2}$ represents the integer $2^{2^{t-2}} (2^{2^{t-1}} - 1)$ (Refs 2, 3). For $t = 5$, since $2^{32} \equiv -1 \pmod{F_5}$, $\sqrt{2} = 2^{2^4} - 2^8 = 2^{2^4} + 2^{40}$. A conservative value of the dynamic range (Ref. 12) is $\sqrt{2^{32}/(2^8)} \cong 2^{12}$. This value is sufficiently large for a number of applications.

Since the FNT has a mathematical algorithm similar to the FFT, an FFT-type structure can be applied to perform a fast FNT. Figure 1 shows a pipeline structure (Ref. 13) for computing a 128-point FNT over F_5 . The radix-2 decimation-in-time (DIT) technique is used in this structure. The structure for performing an inverse FNT is the mirror image of the circuit shown in Fig. 1 if the radix-2 decimation-in-frequency (DIF) technique is used.

In Fig. 1 z^{-j} denotes a j -step delay element, which can be realized by a set of j first-in-first-out (FIFO) registers. The

symbolic diagram and operations of a DIT FNT butterfly are shown in Fig. 2. The design of a DIT FNT butterfly is given in the Appendix. A similar DIF FNT butterfly was designed in Ref. 4.

In Fig. 1, SW_i is a shuffle-exchange switch controlled by the control signal S_i for $1 \leq i \leq 6$. The operations of the SW_i are shown in Fig. 3. The S_i 's can be implemented simply by a 6-stage up-counter if no buffer registers are used in the FNT butterflies (Ref. 13). With the buffer registers in the butterflies, delay elements are needed at the outputs of the counter, as shown in Fig. 4, for the purpose of synchronization.

In the next section the overlap-save method (Ref. 13) is generalized to implement a digital filter of arbitrary length using one FNT and several inverse FNTs of 128 points over F_5 . Then a parallel VLSI architecture is designed to realize this overlap-save method using the FNT structure designed above.

III. A Digital Filter Architecture of Arbitrary Length Using the FNT

In the previous section F_t , α , and N are chosen to be F_5 , $\sqrt{2}$, and 128 respectively. $N = 128$ is the maximum transform length over F_5 (Refs. 2, 3), and 2^{12} is the dynamic range. One could increase the transform length by choosing F_t for $t \geq 6$. In so doing, however, at least $2^6 + 1 = 65$ bits are required to represent a number. Alternatively, one could use a specific α , where α is not a power of $\sqrt{2}$, over F_3 or F_4 to increase the transform length. In such a case a complete multiplication is required. In addition, the dynamic range is used up readily. To remedy this difficulty, the overlap-save method is generalized to compute the linear convolution of a digital filter of arbitrary input data and filter lengths. A parallel architecture is developed to realize this generalized overlap-save method using the 128-point FNT structure designed in the previous section.

Let $\{x_n\}$ and $\{h_m\}$ be the input and filter sequences of a digital filter, respectively, where $0 \leq n \leq N - 1$ and $0 \leq m \leq M - 1$. The output sequence $\{y_k\}$ of the filter is the linear convolution of $\{x_n\}$ and $\{h_m\}$, where $0 \leq k \leq N + M - 1$ (Ref. 13). It is shown (Ref. 13) that such a linear convolution can be obtained by computing a cyclic convolution. For purposes of exposition it is assumed that $N = 1024$ and $M = 256$ in the following argument.

In order to use 128-point FNTs to compute $\{y_k\}$, four 128-point subfilters $\{h_m^1\}$, $\{h_m^2\}$, $\{h_m^3\}$ and $\{h_m^4\}$ are formed by partitioning $\{h_m\}$ as follows:

$$h_m^i = \begin{cases} h_{m+64(i-1)} & \text{for } 0 \leq m \leq 63 \\ 0 & \text{for } 64 \leq m \leq 127 \end{cases} \quad (3)$$

for $1 \leq i \leq 4$. Next the overlap-save method (Ref. 13) is used to compute the linear convolution $\{y_k^i\}$ of $\{x_n\}$ and $\{h_m^i\}$ by using the cyclic convolution technique, where $1 \leq i \leq 4$ and $0 \leq k \leq 1087$. To accomplish this $\{x_n\}$ is sectioned into 128-point subsequences with 64 points of $\{x_n\}$ overlapped between two consecutive subsequences. That is $\{x_n\}$ is sectioned into $\{x_m^1\} = \{x_m\}$, $\{x_m^2\} = \{x_{m+64}\}$, \dots , $\{x_m^{15}\} = \{x_{m+896}\}$, where $0 \leq n \leq 1023$ and $0 \leq m \leq 127$. Then $\{y_k^i\}$, for $1 \leq i \leq 4$, is computed by overlapping the cyclic convolution of $\{h_m^i\}$ and $\{x_m^j\}$ for $1 \leq j \leq 15$ using 128-point FNTs. Finally the output sequence $\{y_k\}$, for $0 \leq k \leq 1024 + 256 - 1 = 1279$, results evidently from $\{y_k^i\}$ for $1 \leq i \leq 4$ by the following equation:

$$\begin{aligned} y_k &= y_k^1 + y_k^2 z^{-64} + y_k^3 z^{-128} + y_k^4 z^{-192} \\ &= (y_k^1 + y_k^2 z^{-64}) + (y_k^3 + y_k^4 z^{-64}) z^{-128} \end{aligned} \quad (4)$$

The relationship between $\{y_k\}$ and $\{y_k^i\}$ for $1 \leq i \leq 4$ is illustrated in Fig. 5. Other cases of the generalized overlap-save method are constructed in a similar manner.

In Fig. 6 is shown the block diagram of an architecture for the generalized overlap-save method of a digital filter using one FNT and four inverse FNTs of 128 points. In this system the DIT and DIF techniques are used for the FNT and inverse FNTs, respectively. In the generalized overlap-save method, one of the two outputs of the inverse FNT butterfly in the last stage is not needed. Hence, the inverse FNT butterfly in the

last stage is a degenerative butterfly circuit, and the delay elements associated with this butterfly circuit are not needed. The H_k^i 's in Fig. 6 are the FNTs of $\{h_m^i\}$. The $(1/N)$ factor in Eq. (2) is incorporated into the H_k^i 's. These H_k^i 's can be precomputed and stored in the system. The adders in Fig. 6 perform normal binary additions, not additions modulo F_r .

The advantage of the generalized overlap-save method for implementing a digital filter using FNT transforms are the following: (1) It requires no multiplications. Only additions and bit rotations are needed. (2) It alleviates the usual dynamic range limitation for long sequence FNTs. (3) It utilizes the FNT and inverse FNT circuits 100% of the time. (4) The lengths of the input data and filter sequences can be arbitrary and different.

IV. Conclusion

A pipeline structure is developed to compute a 128-point Fermat number transform. In this 128-point FNT, only additions and bit rotations are required. A barrel shifter circuit is modified to perform the multiplication of an integer by a power of 2 modulo a Fermat number. The overlap-save method is generalized to compute the linear convolution of a digital filter with arbitrary input data and filter lengths. An architecture is developed to realize this generalized overlap-save method by a simple combination of one 128-point FNT and several inverse FNT structures. This realization alleviates the dynamic range limitations of the FNT with a long transform length. The architecture is simple and regular, and hence suitable for VLSI implementation.

References

1. Rader, C.M., "Discrete Convolutions Via Mersenne Transforms," *IEEE Trans. Computers*, Vol. C-21, No. 12, pp. 1269-1273, Dec. 1972.
2. Agarwal, R. C., and Burrus, C. S., "Fast Convolution Using Fermat Number Transforms with Applications to Digital Filtering," *IEEE Trans. Acoustic, Speed, and Signal Processing*, Vol. ASSP-22, No. 2, pp. 87-97, April 1974.
3. Agarwal, R. C., and Burrus, C. S., "Number Theoretical Transforms to Implement Fast Digital Convolution," *Proc. IEEE*, Vol. 63, No. 4, pp. 550-560, April 1975.
4. McClellan, J. H., "Hardware Realization of A Fermat Number Transform," *IEEE Trans. Acoustic, Speed, and Signal Processing*, Vol. ASSP-24, No. 3, pp. 216-225, June 1976.
5. Leibowitz, L. M., "A Simplified Binary Arithmetic For The Fermat Number Transform," *IEEE Trans. Acoustic, Speed, and Signal Processing*, Vol. ASSP-24, No. 5, pp. 356-359, Oct. 1976.
6. Reed, I. S., Truong, T. K., Kwoh, Y. S., and Hall, E. L., "Image Processing by Transforms Over A Finite Field," *IEEE Trans. Computers*, Vol. C-26, No. 9, pp. 874-881, Sep. 1977.
7. Rader, C. M., "On the Application of the Number Theoretic Methods of High Speed Convolution to Two-Dimensional Filtering," *IEEE Trans. Circuits and Systems*, Vol. CAS-22, No. 6, pp. 575, June 1975.
8. Reed, I. S., Kwoh, Y. S., Truong, T. K., and Hall, E. L., "X-Ray Reconstruction by Finite Field Transforms," *IEEE Trans. on Nuclear Science*, Vol. NS-24, No. 1, Feb. 1977.
9. Reed, I. S., Truong, T. K., and Welch, L. R., "The Fast Decoding of Reed-Solomon Codes Using Fermat Number Transforms," *IEEE Trans. Information Theory*, Vol. IT-24, No. 4, pp. 497-499, July 1978.
10. Roots, H. F. A., and Best, M. R., "Concatenated Coding on a Spacecraft-to-Ground Telemetry Channel Performance," Processing ICC 81, Denver, CO. 1981.
11. Mead, C. A., and Conway, L. A., *Introduction to VLSI Systems*, Addison-Welsey, Reading, Mass., 1980.
12. Reed, I. S., and Truong, T. K., "The Use of Finite Field to Compute Convolutions," *IEEE Trans. Information Theory*, Vol. IT-21, No. 2, pp. 208-213, March 1975.
13. Rabiner, L. R., and Gold, B., *Theory and Application of Digital Signal Processing*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.

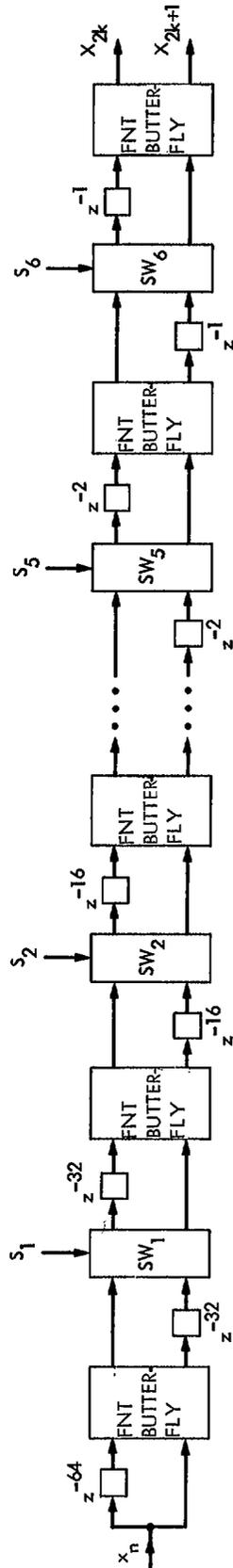


Fig. 1. A pipelined structure for computing a 128-point Fermat number transform

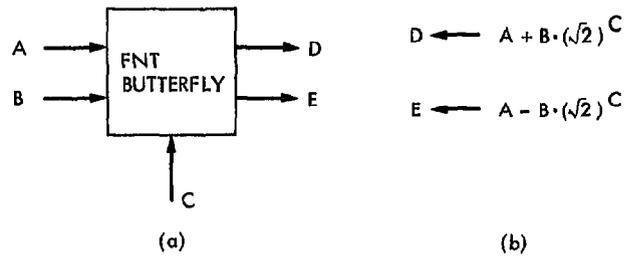


Fig. 2. (a) The symbolic diagram and (b) operations of a DIT FNT butterfly

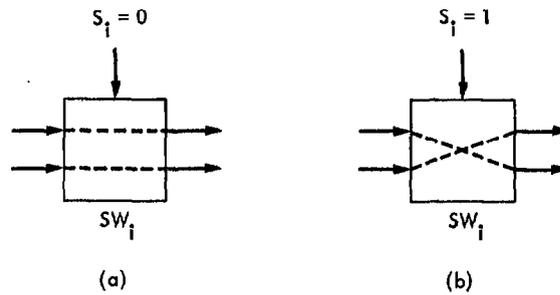


Fig. 3. A shuffle-exchange switch. (a) Direct connection. (b) Crossed connection

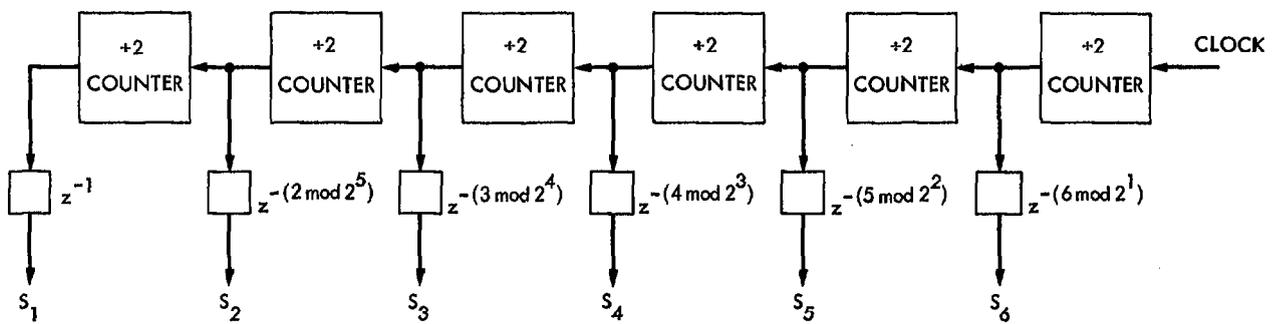


Fig. 4. A 6-stage up-counter used to generate the control signals S_i 's in Fig. 1

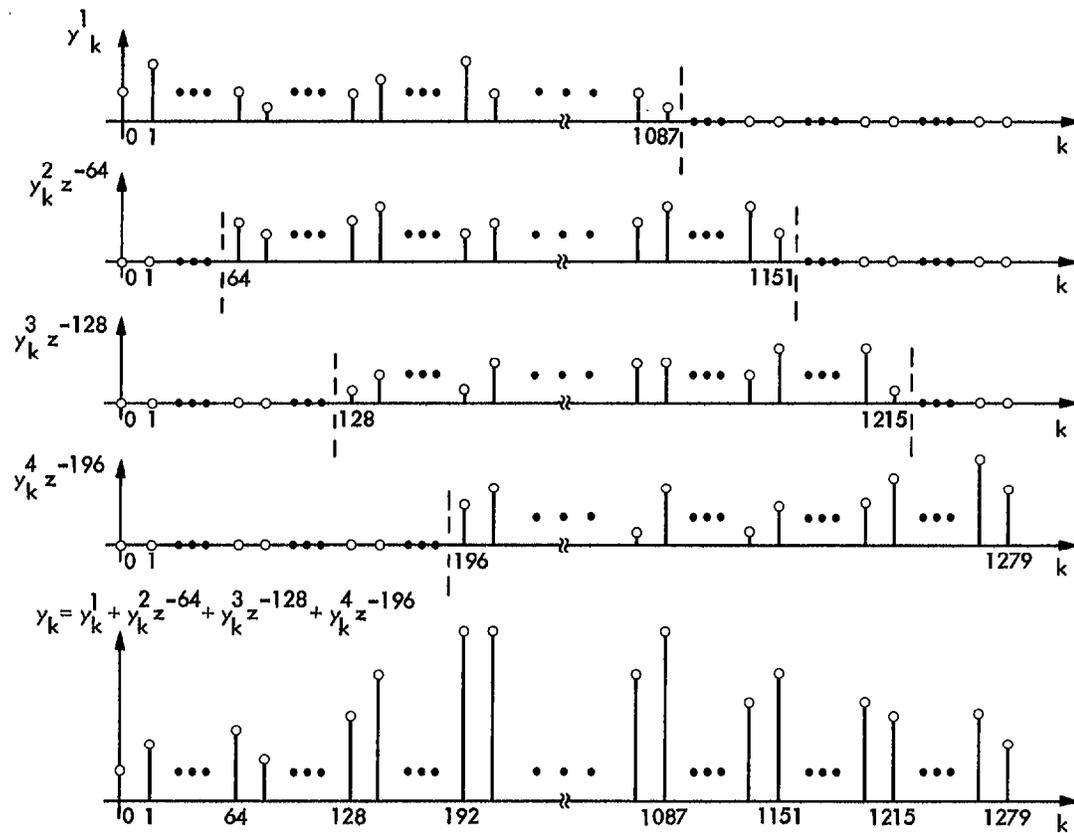


Fig. 5. The example of the generalized overlap-save method in Eq. (4)

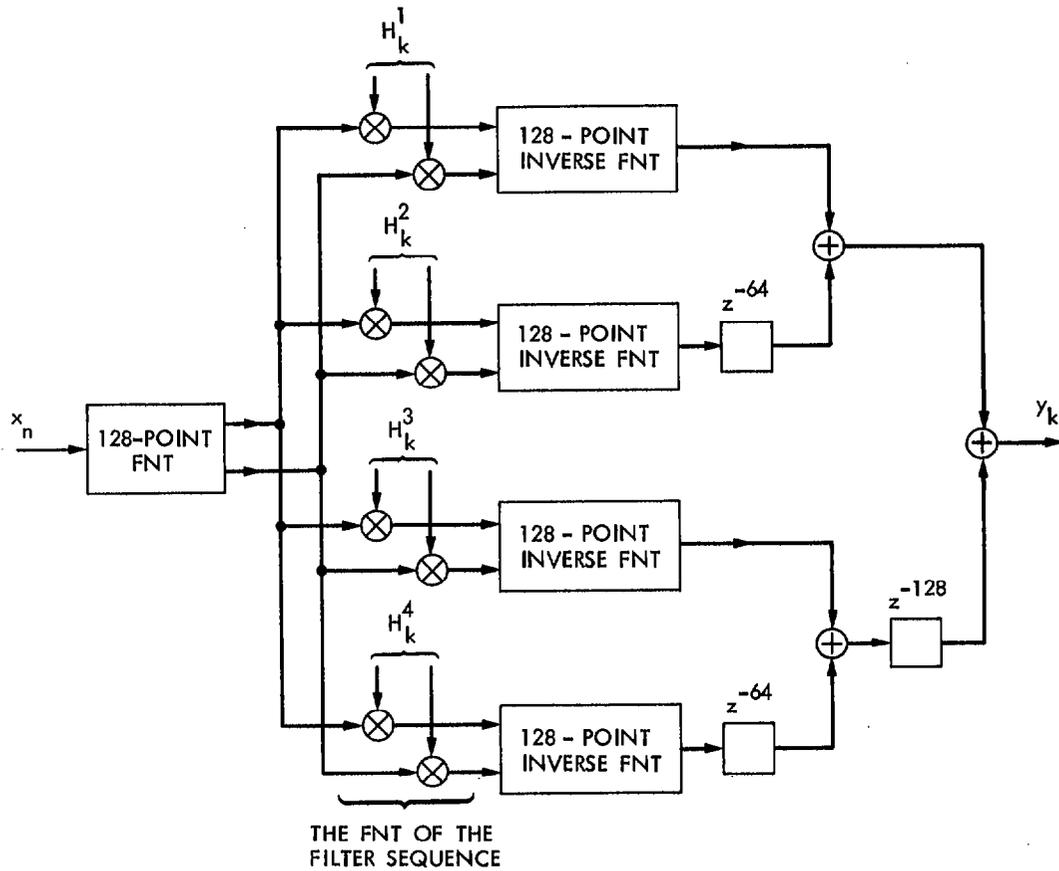


Fig. 6. A realization of a digital filter with a filter sequence of 256 points by using the generalized overlap-save method and the FNT technique

Appendix

In this appendix a circuit is designed to implement a DIT FNT butterfly shown in Fig. 2. A similar DIF FNT butterfly was designed in Ref. 4. To efficiently perform the FNT, number representations have been proposed (Refs. 4, 5) for binary arithmetic operations modulo F_7 . The diminished-1 representation proposed by Liebowitz (Ref. 5) is used in the following design. Let A be represented by $[a_{32} a_{31} \dots a_1 a_0]$, where $0 \leq A \leq 2^{32}$ and a_i is the i th bit of A . Table A-1 shows the correspondence between decimal numbers in a normal binary representation and their values in the diminished-1 representation. The most significant bit (MSB) a_{32} can be viewed as the zero-detection bit in the diminished-1 representation.

Two basic binary arithmetic operations modulo F_7 with $\alpha = \sqrt{2}$ are addition and multiplication by a power of 2. Other operations can be expressed in terms of these two operations. In the following, some details of these operations are described briefly. More specifics can be found in Ref. 5.

- (1) Addition: Let $S = A + B$. If $A = 0$, then $S = B$. If $B = 0$, then $S = A$. If neither A nor B equals 0, add $[a_{31} a_{30} \dots a_1 a_0]$ and $[b_{31} b_{30} \dots b_1 b_0]$. Then complement the carry and add it to the previous sum. This yields S .
- (2) Multiplication by a power of 2: Let $B = A \cdot 2^C$. If $A = 0$, then $B = 0$. If $A \neq 0$, left rotate $[a_{31} a_{30} \dots a_1 a_0]$ C bit positions, but complement the value of bit 31 when it is rotated to bit position 0, and set $b_{32} = 0$.

(3) Negation: Since $2^{32} \equiv -1 \pmod{F_5}$, $-A = A \cdot 2^{32}$. Hence if $A \neq 0$, $-A = [a_{32} \bar{a}_{31} \bar{a}_{30} \dots \bar{a}_1 \bar{a}_0]$ where \bar{a}_i denotes the complement of a_i . If $A = 0$, then $-A = 0$.

(4) Multiplication by $\sqrt{2}$: Since $\sqrt{2} = 2^{24} + 2^{40}$, $A \cdot \sqrt{2} = A \cdot 2^{24} + A \cdot 2^{40}$.

(5) Multiplication by a power of $\sqrt{2}$: Let $B = A \cdot (\sqrt{2})^C$. If C is even, then $B = A \cdot (2)^{C/2}$. If C is odd, then $B = (A \cdot \sqrt{2}) \cdot 2^{(C-1)/2}$.

In Fig. A-1 is shown a block diagram of an FNT butterfly shown in Fig. 2. In this design, A, B, D , and E are 33-bit data, and C is the 7-bit exponent nk in Eq. (1). Two realizations of an FNT adder can be found in Ref. 4. Figure A-2 shows a pass-transistor full-adder, which requires less silicon area. The multiplier in Fig. A-1 is used to multiply a number by a power of 2 modulo F_5 . Figure A-3 shows a block diagram of this multiplier. The shifter in Fig. A-3 is a modification of a barrel shifter (Ref. 11) for performing bit rotation operations.

For purposes of illustration, consider the simple FNT over $F_0 = 2 + 1$. In such an FNT butterfly the functional table and circuit of a modified barrel shifter are shown in Fig. A-4, where the inputs are $[b_1 b_0]$ and $[s_3 s_2 s_1 s_0]$, and the outputs are $[b_1^* b_0^*]$.

Table A-1. The correspondence among decimal numbers, their values in the normal binary representation, and in the diminished-1 representation

Decimal number	Normal binary representation							Diminished-1 representation						
	a_{32}	a_{31}	a_{30}	...	a_2	a_1	a_0	a_{32}	a_{31}	a_{30}	...	a_2	a_1	a_0
0	0	0	0	...	0	0	0	1	0	0	...	0	0	0
1	0	0	0	...	0	0	1	0	0	0	...	0	0	0
2	0	0	0	...	0	1	0	0	0	0	...	0	0	1
.				.							.			
.				.							.			
.				.							.			
2^{32-2}	0	1	1	...	1	1	0	0	1	1	...	1	0	1
2^{32-1}	0	1	1	...	1	1	1	0	1	1	...	1	1	0
2^{32}	1	0	0	...	0	0	0	0	1	1	...	1	1	1

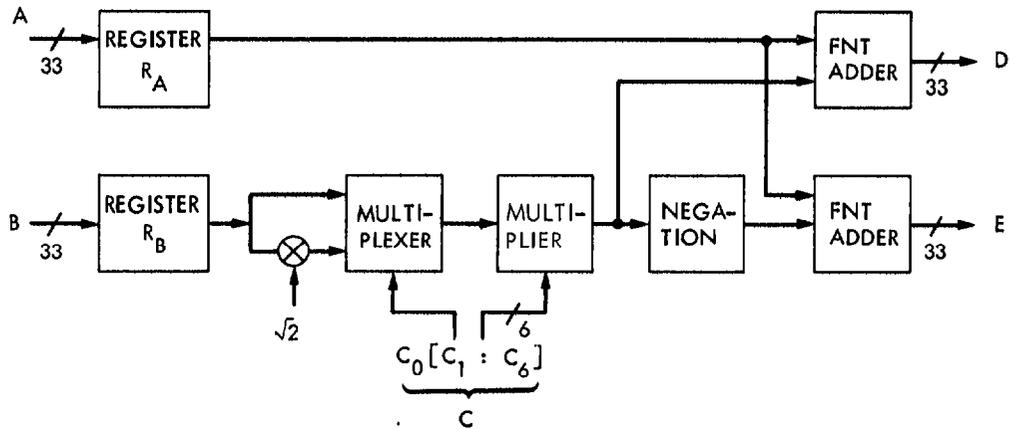


Fig. A-1. A block diagram of a DIT FNT butterfly

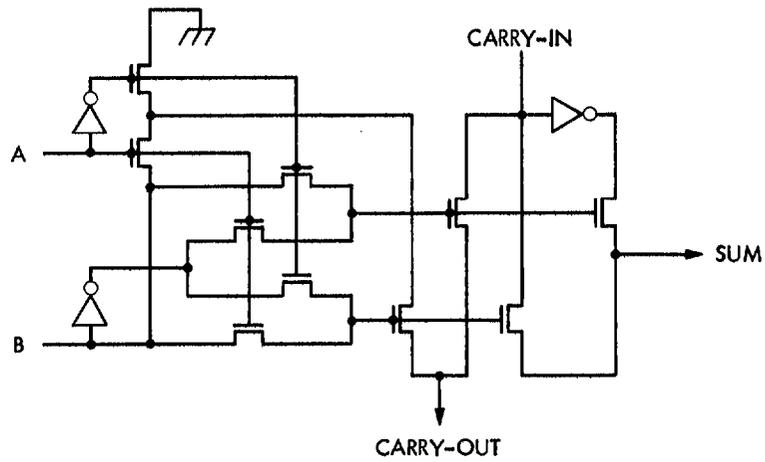


Fig. A-2. A pass-transistor full-adder

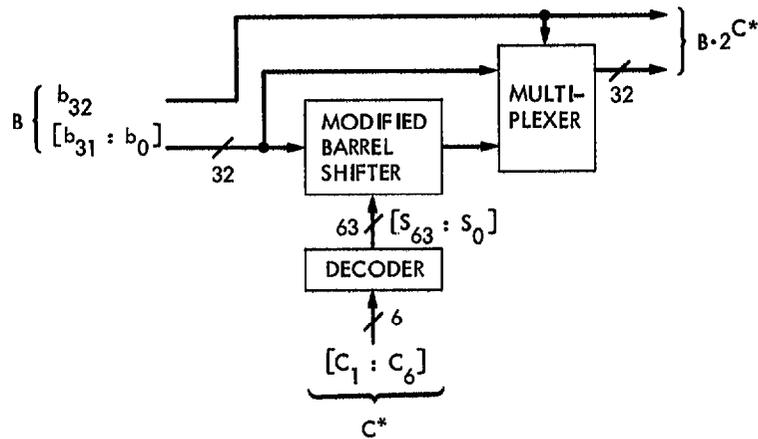
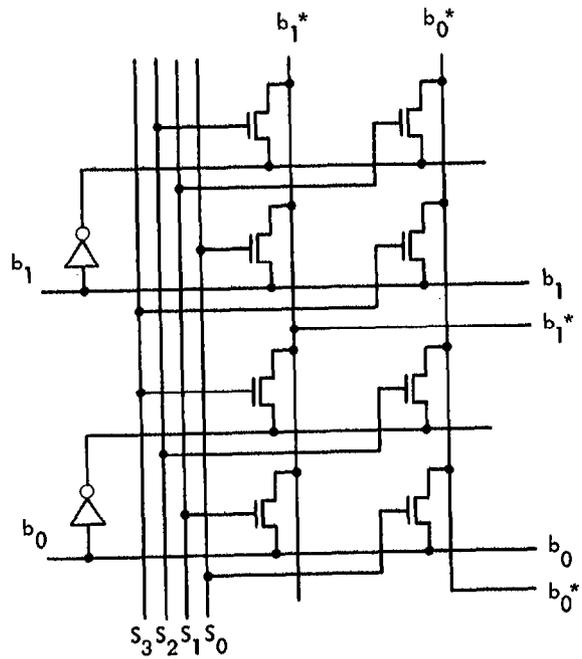


Fig. A-3. A circuit to perform $B \cdot 2^{C^*}$

INPUTS				OUTPUTS	
s_3	s_2	s_1	s_0	b_1^*	b_0^*
0	0	0	1	b_1	b_0
0	0	1	0	b_0	\bar{b}_1
0	1	0	0	\bar{b}_1	\bar{b}_0
1	0	0	0	\bar{b}_0	b_1



(a)

(b)

Fig. A-4. (a) The functional table and (b) circuit of a modified barrel shifter