A High-Throughput, Low Power Architecture and Its VLSI Implementation for DFT/IDFT Computation

Shen-Fu Hsiao   Wei-Ren Shiue
Institute of Computer and Information Engineering
National Sun Yat-Sen University
Taiwan

Abstract

A recursive algorithm for computation of both forward and backward DFT has been proposed where the common entries in the decomposed matrices are factored out in order to reduce the number of multipliers needed during implementation. The derived algorithm is essentially the band-matrix-vector multiplication with matrix bandwidth of 3. By exploiting the heterogeneous dependency graphs for the matrix-vector multiplication and using an efficient mapping technique, only log_2N adders and log_2N-1 multipliers are needed to compute the DFT of size N, a great saving from a recently proposed systolic architecture which calls for $3log_2N$ adders and $3log_2N$ multipliers. Furthermore, due to the simplicity and regularity of the architectures, it is possible to design a low power processor by turning off the hardware components of no operation at proper time steps. VLSI implementation of the DFT/IDFT processor with distributed FSM for timing control is also presented.

1. Introduction

Discrete Fourier transform (DFT) has been one of the fundamental operations in DSP. Since the famous fast Fourier transform (FFT) algorithm by Cooley and Tukey, there have been a lot of papers on the fast implementation of DFT in order to achieve real-time processing speed [2]. As VLSI technology advances, new criteria of designing and evaluating the fast algorithms emphasize on regularity, modularity, and locality in the architecture realization. In this case, systolic arrays [3] become popular for VLSI implementation since they satisfy the above architecture requirement. However, most papers present systolic architectures with hardware complexity proportional to the transform length N, which is impractical for VLSI implementation of long-length transforms due to the high hardware cost. Recently, Boriokoff [1] proposed a new systolic DFT architecture which requires only $3log_2N$ complex adders and multipliers. The basic idea is to decompose the transform coefficient matrix into product of simpler band matrices with only a number of regularly spaced nonzero diagonals, and to implement the resultant band-matrix-vector multiplication by efficient systolic arrays.

In this paper, the number of complex multipliers in Boriakoff's systolic DFT architecture is further reduced to only log_2N-1 by exploiting the features of the entries in the decomposed coefficient matrices. A hardware-efficient linear systolic architecture based on the algorithms is generated based on the mapping of the corresponding *heterogeneous* dependency graphs (DGs). The new architecture has very regular structure and simple control and thus is easily scalable and pipelinable. Furthermore, due to the regularity of the multiplied coefficients, it is easy to reduce the power consumption of the DFT processor by turning off the temporarily unused multipliers. VLSI implementation of an 8-pt. DFT processor is presented which consists of the related coefficient ROMs and the distributed finite state machine (FSM) for the control unit.

2. Recursive Formula for DFT

The definition of order-N DFT $yk(N,x)=\sum_{n=0}^{N-1} xn\times W_N^{kn}$, $W_N=\exp(-j\frac{2\pi}{N})$ for a given input complex data sequence $\{xn:n=[0,N-1]\}$ and the output complex data sequence $\{yk:k=[0,N-1]\}$ could be expressed as the matrix-vector multiplication $y=E(N)\,x$ where $E(N)$ denotes the coefficient matrix of N-pt. DFT.

Splitting the time index n into two equal intervals, one with index n':$[0,N/2-1]$ and the other with index $n''=n'+N/2$, and splitting the frequency index k into two equal intervals according to the even and odd partitioning, we obtain the relationship among the four $N/2$-pt. DFTs as shown in Tab. 1 where $E(N/2)$ denotes the $N/2$-pt. DFT.

	$n=n'$, $n'=[0,N/2-1]$	$n=n''=n'+N/2$
$k=2i$, $i=[0,N/2-1]$	$E(N/2)$	$E(N/2)$
$k=2i+1$	$E(N/2)\,W_{N/2}^{n'/2}$	$-E(N/2)\,W_{N/2}^{n'/2}$

Tab. 1: The relationship between the four $N/2$-pt. DFTs after the index partitioning.

Writing Tab. 1 in the matrix form, we have

$$\begin{bmatrix} ye \\ yo \end{bmatrix} = \begin{bmatrix} E(N/2) & E(N/2) \\ E(N/2)C(N/2) & -E(N/2)\,C(N/2) \end{bmatrix} \begin{bmatrix} xp \\ xr \end{bmatrix}, \tag{1}$$

where x_p is the preceding half of the input data; x_r is the rear half of the input data; y_e is the even-numbered transformed data; y_o is the odd-numbered transformed data; $C(N/2)=diag(W_{N/2}^{n'/2})$ is an $(N/2)\times(N/2)$ diagonal matrix. Repeating the above partition for the DFT of progressively smaller size, the frequency index will be eventually in bit-reverse order while the time index is still in the normal order. Let $R(N)$ denote the permutation matrix for the bit-reverse operation of an N-pt. vector and let $\hat{y}=R(N)\,y$ be the bit-reversed output vector of size N. Eqn. (1) becomes

$$\begin{aligned}
\hat{y} = \begin{bmatrix} \hat{y}e \\ \hat{y}o \end{bmatrix} &= \begin{bmatrix} R(N/2) & \\ & R(N/2) \end{bmatrix} \begin{bmatrix} y_e \\ y_o \end{bmatrix} \\
&= \begin{bmatrix} R(N/2) & \\ & R(N/2) \end{bmatrix} \begin{bmatrix} E(N/2) & E(N/2) \\ E(N/2)C(N/2) & -E(N/2)C(N/2) \end{bmatrix} \begin{bmatrix} xp \\ xr \end{bmatrix} \\
&= \begin{bmatrix} \hat{E}(N/2) & \hat{E}(N/2) \\ \hat{E}(N/2)C(N/2) & \hat{E}(N/2)C(N/2) \end{bmatrix} x \\
&= \begin{bmatrix} \hat{E}(N/2) & \\ & \hat{E}(N/2) \end{bmatrix} \begin{bmatrix} I(N/2) & I(N/2) \\ C(N/2) & -C(N/2) \end{bmatrix} \begin{bmatrix} xp \\ xr \end{bmatrix} \\
&= \begin{bmatrix} \hat{E}(N/2) & \\ & \hat{E}(N/2) \end{bmatrix} \underbrace{\underbrace{\begin{bmatrix} I(N/2) & \\ & C(N/2) \end{bmatrix}}_{BD_2(N)} \underbrace{\begin{bmatrix} I(N/2) & I(N/2) \\ I(N/2) & -I(N/2) \end{bmatrix}}_{BB_2(N)}}_{B_2(N)} \begin{bmatrix} xp \\ xr \end{bmatrix}
\end{aligned} \tag{2}$$

where $R(N/2)$ is the permutation matrix for the bit-reversed operation of $N/2$ data samples. $\hat{E}(N/2) = R(N/2)E(N/2)$ is generated by rearranging the rows of $E(N/2)$ in bit-reversed order. $I(N/2)$ is the $(N/2)\times(N/2)$ identity matrix. $B_2(N) = BD_2(N)BB_2(N)$ is the butterfly operation matrix and can be split into a diagonal matrix $BD_2(N)$ and a band matrix $BB_2(N)$ consisting of only three nonzero diagonals. Eqn. (2) is in fact the general decimation-in-frequency (DIF) recursive formula for order-*N* DFT [5]. In the next section, we will show how to realize the butterfly operation $B2(N)$ using an efficient linear systolic array containing only one multiplier and one adder.

Since $DFT(N)$ is symmetric and the inverse of the permutation matrix $R^{-1}(N) = R^T(N)$ is equal to its transpose, the decimation-in-time (DIT) DFT algorithm could be simply obtained from the DIF algorithm. Indeed,

$$\hat{y} = \hat{E}(N)x \quad \Rightarrow R(N)\hat{y} = R(N)\hat{E}(N)R^T(N)R(N)x \quad \Rightarrow$$
$$y = E(N)R^T(N)\hat{x} = E^T(N)R^T(N)\hat{x} = \left(R(N)E(N)\right)^T\hat{x} = \hat{E}^T(N)\hat{x}$$

Thus the recursive algorithm for the order-*N* DIT DFT can be expressed as

$$\begin{aligned} y &= \begin{bmatrix} \hat{E}^T(N/2) & C(N/2)\,\hat{E}^T(N/2) \\ \hat{E}^T(N/2) & -C(N/2)\,\hat{E}^T(N/2) \end{bmatrix}\hat{x} \\ &= \underbrace{\begin{bmatrix} I(N/2) & I(N/2) \\ I(N/2) & -I(N/2) \end{bmatrix}\begin{bmatrix} I(N/2) & \\ & C(N/2) \end{bmatrix}}_{B_2^T(N)}\begin{bmatrix} \hat{E}^T(N/2) & \\ & \hat{E}^T(N/2) \end{bmatrix}\hat{x} \\ &= B_2^T(N)\begin{bmatrix} \hat{E}^T(N/2) & \\ & \hat{E}^T(N/2) \end{bmatrix}\hat{x} \end{aligned} \quad (3)$$

Note the difference of the *post*-multiplication in $B2(N)$ for the DIF DFT and the *pre*-multiplication in $B_2^T(N)$ for the DIT DFT.

Using the same index partition method as in the forward DFT, we could obtain general recursive formula for inverse DFT (IDFT) in DIT or DIF form. The butterfly matrix of the DIT IDFT is $IB_2(N) = B_2^{-1}(N)$ while the butterfly matrix for the DIF IDFT is $IB_2^T(N) = B_2^{-T}(N)$.

We observe that the recursive algorithm for the DIF DFT and that for the DIT IDFT lead to the same architecture with different post-multiplication coefficients in the butterfly matrices. Thus, we can merge the DIF forward DFT and the DIT backward DFT into the same architecture by changing multiplication coefficients in BD_2 and BD_2^{-1}. In fact, as will be seen in Sec. 4, we don't need to prepare two different coefficient ROMs (for the input operand of the multipliers) for both the forward and backward DFTs. Instead, both DFT and IDFT can share the same coefficient ROMs.

3. New DFT Architecture

We would consider a special case for *N*=8 and then extend to the order-*N* DFT architecture. According to Eqn. (2), we could obtain

$$\hat{y} = \underbrace{\begin{bmatrix} B_2(2) & & & \\ & B_2(2) & & \\ & & B_2(2) & \\ & & & B_2(2) \end{bmatrix}}_{\text{stage 3}} \underbrace{\begin{bmatrix} B_2(4) & \\ & B_2(4) \end{bmatrix}}_{\text{stage 2}} \underbrace{B_2(8)}_{\text{stage 1}}\; x \quad (4)$$

where three stages of butterfly operations are

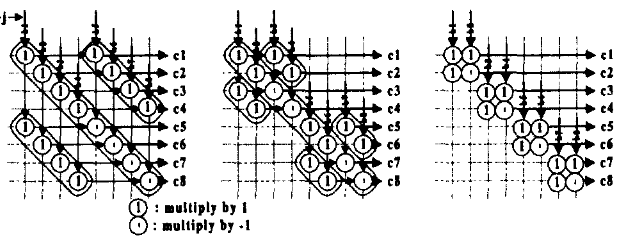
$$B2(2) = \begin{bmatrix} I(1) & \\ & C(1) \end{bmatrix}\begin{bmatrix} I(1) & I(1) \\ I(1) & -I(1) \end{bmatrix} = \begin{bmatrix} 1 & \\ & 1 \end{bmatrix}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$B2(4) = \begin{bmatrix} I(2) & \\ & C(2) \end{bmatrix}\begin{bmatrix} I(2) & I(2) \\ I(2) & -I(2) \end{bmatrix} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & W_2^{1/2} \end{bmatrix}\begin{bmatrix} 1 & & 1 & \\ & 1 & & 1 \\ 1 & & -1 & \\ & 1 & & -1 \end{bmatrix}$$

$$B2(8) = \begin{bmatrix} I(4) & \\ & C(4) \end{bmatrix}\begin{bmatrix} I(4) & I(4) \\ I(4) & -I(4) \end{bmatrix} \quad (5)$$

$$= \begin{bmatrix} 1 &&&&&&& \\ & 1 &&&&&& \\ && 1 &&&&& \\ &&& 1 &&&& \\ &&&& 1 &&& \\ &&&&& W_4^{1/2} && \\ &&&&&& W_4^{1} & \\ &&&&&&& W_4^{3/2} \end{bmatrix}\begin{bmatrix} 1 &&&& 1 &&& \\ & 1 &&&& 1 && \\ && 1 &&&& 1 & \\ &&& 1 &&&& 1 \\ 1 &&&& -1 &&& \\ & 1 &&&& -1 && \\ && 1 &&&& -1 & \\ &&& 1 &&&& -1 \end{bmatrix}$$

Each butterfly operation could be divided into one diagonal matrix and a 3-band matrix. The multiplication of a diagonal matrix can be realized by a single multiplier. Thus, we focus on the structural implementation of the band-matrix-vector multiplication where the band matrix consists of elements of only 0, 1, and −1. The dependence graphs (DGs) of the 3-band matrices in the three stages are shown in Fig. 1. Numbers in nodes represent matrix elements in the corresponding 3-band matrices.



(a) (b) (c)

Fig. 1: Homogeneous DGs of 3-band matrixes in (a) stage 1, (b) stage 2 and (c) stage 3 of the order-8 DIF DFT.

Direct mapping of the homogeneous DG in each butterfly stage leads to three PEs. However, it is possible to reduce the hardware cost by some manipulation of the DGs. We move the lower band in each DG rightward and cascade it with the upper band as illustrated in Fig. 2. After the movement, the original homogeneous DGs become heterogeneous, and thus some multiplexers are required to select the desirable input operands. After projecting and scheduling along the direction of (i, j)=(1, 1), only two PEs are required, as shown in Fig. 3 for the first two stages of the DIF DFT. We use a different schedule vector for stage 3, since the schedule vector of (1, 1) leads to poor utilization efficiency of PEs and lower throughput rate (only 50%). In order to derive better architecture for stage 3, the direction of the inputs is changed upward, seen in Fig. 4(a). The schedule vector is selected as $(-1,2)$, and the result architecture is shown in Fig. 4(b).

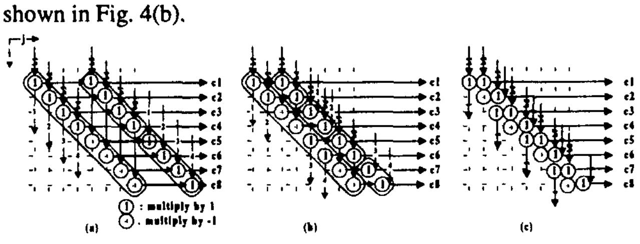
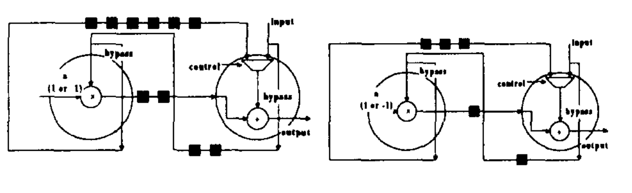
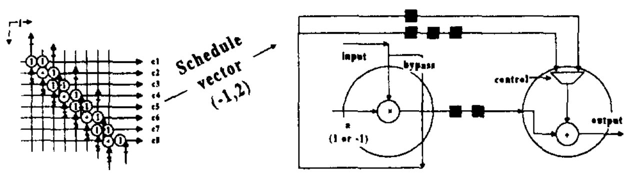


Fig. 2: Heterogeneous DGs of 3-band matrixes in (a) stage 1, (b) stage 2 and (c) stage 3 after the movement of the order-8 DIF DFT.



(a) (b)

Fig. 3: Architectures for the (a) stage 1 and (b) stage 2 of the order-8 DIF DFT.



(a) (b)

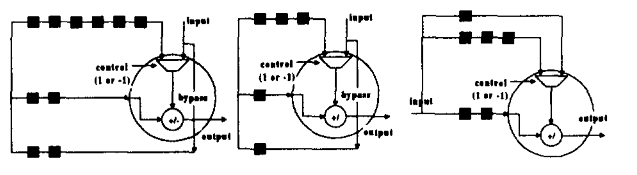
Fig. 4: (a) The new DG, and (b) the corresponding architecture in the last stage (stage 3) of the order-8 DIF DFT.

Further improvement for the architectures in Figs. 3 and 4 is possible. Since the left PE in each stage performs the multiplication of 1 or -1, the two PEs in each stage of Figs. 3 and 4 can be merged, as shown in Fig. 5 for the three stages. The control signal of each multiplexer selects the left or right input at the same time when it selects the operations of addition or subtraction in the add/sub module. A complete systolic architecture for order-8 radix-2 DIF DFT is shown in Fig. 6 where the number of registers in each stage is slightly different from that in Fig. 5 since we further reduce the number of registers through re-timing.

The two multipliers between stages 1-2 and stages 2-3 realize respectively the multiplication of the diagonal elements in $BD_2(8)$ and $BD_2(4)$ of Eqn. (5). Note that no multiplier is required for $BD_2(2)$ since all the diagonal elements are 1. In fact, the efficiency of the multiplier is only 50 percentage since half of the multiplied coefficients in each multiplier are 1's, as can be found in the diagonal elements of $BD_2(8)$ and $BD_2(4)$. Thus, we could add some control to bypass the input while the multiplied value is 1 in order to save power consumption. Thus, the DIF DFT architecture can easily save not only area but also power while maintaining the high throughput rate, a favorable design in consideration of area, power, and speed performance. VLSI implementation of a low-power high-throughput 8-pt. DFT/IDFT processor will be given in Sec. 4.

The 8-pt. DIF DFT (or DIT IDFT) architecture can be easily scaleable to general order of N. Fig. 7 shows the architecture of an N-pt. DIF DFT (or DIT IDFT) for any stage i ($i \neq \log_2 N$) except the last stage. The architecture of the last stage is the same as that in Fig. 6. Thus, only log_2N adders and log_2N-1 multipliers are required to compute the DFT/IDFT of size N with N power of 2. Compared with the approach in [1] that calls for 3 log_2N adders and $3log_2N$ multipliers, our algorithm leads to much less hardware since the common factors in each row of $B2(N)$ are collected into the diagonal entry of $BD2(N)$. Architecture for order-N DIT DFT (or the DIF IDFT) is the mirror of that for order-N DIF DFT (or DIT IDFT) and could be derived by reversing the stage order in Fig. 6.

Tab. 2 compares our radix-2 DFT architectures with another recently proposed $O(log_2N)$ architecture by Boriakof in [1] where the butterfly operation in each stage is implemented by three multiplication-addition units. The reduction of the hardware cost in our DFT processor is due to the factoring out of the common multiplication coefficients of $BD_2(N)$ in Eqn. (2) and the efficient mapping of the heterogeneous DGs in Fig. 2 for the 3-band matrix BB_2. Most other systolic DFT architectures, for example that in [4], requires $O(N)$ PEs making them impractical for VLSI implementation when N is large.



(a) (b) (c)

Fig. 5: The single-PE architecture for (a) stage 1, (b) stage 2, and (c) stage 3 of the order-8 DIF DFT.

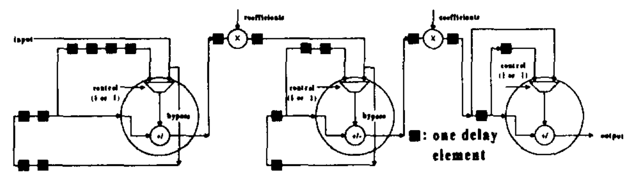


Fig. 6: A complete systolic architecture for the radix-2 order-8 DIF DFT (or DIT IDFT).

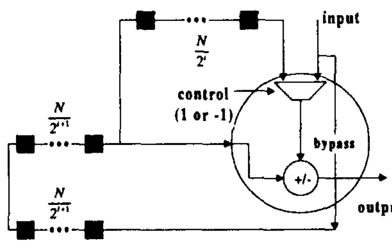


Fig. 7: General architecture of stage i (except the last stage) for order-N DIF DFT (or DIT IDFT).

	[4]	[1]	ours
# of complex multipliers	$> N$	$3log_2N$	log_2N-1
# of complex adders	$> N$	$3log_2N$	log_2N
operation per cycle	one complex multiplication-addition	one complex multiplication-addition	one complex multiplication
throughput	*1 sample/cycle*	*1 sample/cylce*	*1 sample/cycle*

Tab. 2 : Comparison of several systolic DFT architectures.

4. Implementation

A practical VLSI implementation of an order-8 DIF DFT and DIT IDFT processor with the architecture shown in Fig. 6 is given in this section using cell-based design methodology with Compass 0.6um cell library. The major components include complex adders, complex multipliers, coefficient ROM, and a finite state machine (FSM) for the control unit.

A. Complex Multipliers

A complex multiplication $(s1+js2)\times(c1+jc2)$ of an input signal $(s1+js2)$ and a coefficient $(c1+j\,c2)$ requires four real multipliers and two real adders/subtractors. Each butterfly stage (except the last stage) in the order-8 DFT processor also needs two ROMs to store the real (*c1*) and the imaginary (*c2*) parts of the coefficients to be multiplied. Another representation of the complex multiplication is

$$(s1 + js2) \times (c1 + jc2) =$$
$$[c1(s11 + s2) - s2(c1 + c2)] + j[c1(s1 + s2) - s1(c1 - c2)]$$

which calls for only three real multipliers, three adders/subtractors plus three ROMs storing *(c1+c2), (c1-c2),* and *c1*. Compared with the first approach, the second method requires one real multiplier less, but one more real adder/subtractor and one more ROM. Since the ROM size depends on the order *N*, while the area of a real multiplier and an adder depends on the bit accuracy, we perform some experiment and find that with *N<256* and for 8-bit accuracy, the second complex-multiplication approach takes less area.

We also observe that the coefficients in the diagonal matrix $BD_2^{-1}(N)$ of DIT IDFT are the reciprocals of those in the diagonal matrix $BD_2(N)$ of DIF DFT. In other words, the coefficients for IDFT have the same real parts but opposite imaginary pats as those for DFT. Thus, by adding a multiplexer to switch the data from the two ROMs of *(c1+c2)* and *(c1-c2)*, both the DIF DFT and DIT IDFT can share the same ROM.

B. Control

The control signals of 1 or –1 for the DIF DFT architecture are very regular in each butterfly stage and can be generated from counters. The signal of 1 selects the direct input of the multiplexer, and also controls the addition of the adders/subtractor while the control signal of -1 selects the delayed input and specifies the subtraction operation of the adder/subtracter.

After generation of the control signals, we have to design a finite state machine (FSM) to incorporate the operations of PEs in all the butterfly stages. This includes the generation of the start and stop signals in each stage, the request signal for the address counter of the ROM, and the data-ready signal for the output port. Fig. 8 shows the FSM for the control of the *i*-th ($i \neq \log_2 N$) stage in the DIT DFT (or DIF IDFT) processor.

The FSM is divided into three phases: the filling pipeline phase, the full pipeline phase, and the freeing pipeline phase, depending on whether the input signal is going into the stage, completely in the stage, or leaving the stage. The control signal of #start_PE0=1 starts the control counter to determine the function of addition/subtraction and the input selection of the multiplex in the PE. The beginning of the multiplier operation is controlled by the signal #start_MUL=1 which also starts the addressing counter to fetch the coefficients from the ROM. The signal of #ready=1 tells the next stage to begin the computation since the data is available. This type of distributed control scheme, with one FSM for each stage, makes the processor easily scalable to compute DFT of any size.

The final physical layout of the radix-2 order-8 DFT/IDFT processor is shown in Fig. 9 with a core area of 3012x3422 um^2 and maximum operation frequency of 40M HZ under 0.6um, 5-V CMOS technology. The verification of the whole processor, including the coefficient ROMs and the control unit, is done in the RTL level. The Synopsys Design Analyzer is used to generate gate-level code which is then fed into Cadence for automatic place-and-route to general the final physical layout.

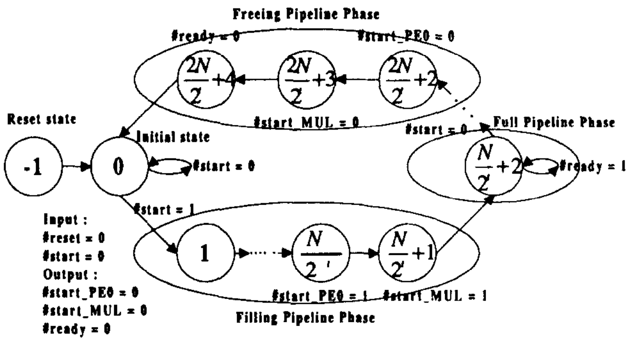


Fig. 8: The finite state machines (FSMs) for the control of the *i*-th stage (except the last stage) of DIF DFT processor.

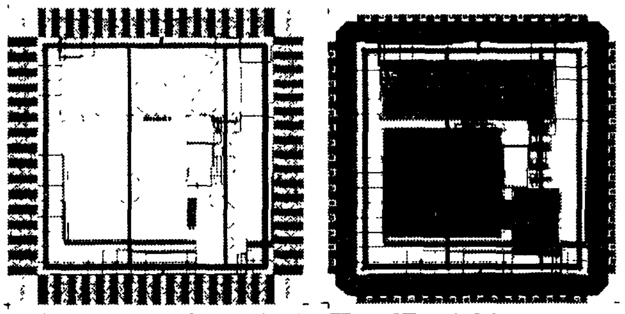


Fig. 9: The layout of the order-8 DFT/IDFT with 8-bit accuracy.

5. Conclusion

A new high-through low-power architecture has been presented which require much less hardware resource compared to other similar approaches. The saving of the hardware cost is achieved by factoring out the common entries in each row of the decomposed matrix and by using the efficient mapping for the heterogeneous dependency graphs. The power saving is obtained by turning off the unused multiplier when the multiplication coefficients are 1. Due to the regularity of the structure and the related control, the architecture is easily pipelinable and scalable to computing DFT of any size. An 8-pt. DFT processor with distributed FSM control and coefficient ROMs is practically implemented onto a single chip.

References:

[1] V. Boriakoff, "*FFT Computation with Systolic Arrays, A New Architecture*", IEEE Trans. Circuits and Systems-II, Vol. 41, No. 4, pp. 278-284, Apr. 1995.

[2] P Duhamel and M. Vetterli, "*Fast Fourier Transform: A Tutorial Review and a State of the Art*", Signal Processing, Vol. 19, No. 4, pp. 259-300, Apr.1990.

[3] S. Y. Kung, *VLSI Array Processors*, Englewood Cliffs, NJ: Prentice-Hall, 1988.

[4] C.-M. Liu and C.-W. Jen, "*On the Design of VLSI Arrays for Discrete Fourier Transform*", IEE Proc.-G, Vol. 139, No. 4, pp. 541-552, Apr.1992.

[5] A. V. Oppenheim and R. W Schafer, "*Discrete-Time Signal Processing*", Prentice-Hall, 1989.