PRIME FACTOR DFT PARALLEL PROCESSOR USING WAFER SCALE INTEGRATION

Edward T. Chow
Dan I. Moldovan

University of Southern California
Department of Electrical Engineering - Systems
Los Angeles, CA 90089-0781

ABSTRACT

A high speed, flexible, simple and regular Discrete Fourier Transform (DFT) Array Processor architecture based on the Prime Factor Algorithm (PFA) is presented in this paper. The array processor is based only on one type of VLSI cell and can compute an N point DFT in N clock cycles throughput when N is a composit number of prime numbers. The high throughput rate is achieved with only a small number of cells. With a special indexing scheme presented in this paper, this processor can use shift registers as the system memory so that minimum global control and addressing is achieved. This array processor architecture is also highly tolerant to both semiconductor processing yield and processor defects during run time. Thus, it can be manufactured in large quantity with VLSI technology on a single wafer and used in hazardus environments With these advantages, it is very attractive to satellite, military and commercial applications.

1. INTRODUCTION

Since 1975, VLSI technology has allowed many complicated digital signal processing building blocks and single chip digital signal processors to become available to system engineers. Many low sampling rate real-time signal processing tasks have been implemented by designing special purpose processors. However, most of the high sampling rate signal processing tasks are still much too demanding in speed and much too complicated to be implemented on single chips with the current technology. The solution is to partition the function into multiple chips.

In order to avoid the cost and performance loss associated with individual packaging of chips, VLSI technologists are fast developing Wafer-Scale Integration (WSI). D. D. Gajski <Ga82> introduced six requirements for a WSI model:

a. a few type of simple processor memory modules replicated throughout the wafer;

b. regular communication network between modules with a constant number of crossovers;

c. I/O ports on the boundary of the wafer;

d. asynchronous communication among modules;

e. I/O rate independent of the size of the problem;

f. high level of fault-tolerance.

Fast computation of Discrete Fourier Transform (DFT) is very important in many areas of digital signal processing. There has been a continued interest in devising new algorithms and processor architectures for this purpose. The Fast Fourier Transform (FFT) and Prime Factor Algorithm (PFA) have been developed to speed up the computation time, primarily by transforming a long DFT into a set of low-order DFTs. These transforms achieve improved computational complexity at the expense of increasing interconnection complexity. Thus, a direct mapping of the FFT and PFA into hardware does not satisfy requirement b) of the WSI model.

The design goal of this paper is to find a parallel architecture for high speed DFT computation which can be implemented by using WSI technology. This architecture satisfies the basic requirements of the WSI model given by Gajski, that is, it must have a minimum number of different cells ,and interconnections must be simple and regular. Furthermore, this architecture has to be insensitive to the semiconductor processing yield, based on an assumption of 20% yield rate. The processor also has to be very flexible in that the length of the DFT computation can be changed by simply reprogramming the cells.

This paper proposes a DFT array architecture based on the Prime Factor Algorithm. This architecture is simple and regular because we found out a special indexing scheme. This indexing scheme is the main factor for simplifying the interconnection complexity of the PFA transform. Only one special VLSI cell is needed, and shift registers are intermixed with the cells to proform arbitrary order DFT on a single wafer with little or no global control. The complexity of this VLSI cell is within the current IC technology capability, so practical implementation is possible.

The organization of this paper is as follows: The PFA and the new indexing scheme is described in section 2. The parallel architecture is presented

in section 3, and a comparison of this processor with other DFT array processors is given in section 4. Some design considerations and conclusions are discussed in section 5.

2. PRIME FACTOR ALGORITHM

The one-dimensional Discrete Fourier Transform of order N has been defined as

$$X_n=\sum_{k=0}^{N-1} x_k W_N^{nk} \quad W_N=EXP(-j\frac{2\pi}{N}) \quad n=0,1,..,N-1 \quad ...(1)$$

The direct evaluation of X_n in equation (1) requires N^2 complex multiplications and additions (N^2 operations). Let N be a composite number decomposable into relatively prime factors. Then the computational burden can be reduced by employing the prime factor algorithm <Co67> as follows:

Assume $N=r_1 * r_2$ where r_1 and r_2 are mutually prime

then, indexes k and n in equation (1) can be expressed as

$$k=(k_1 r_1+k_2 r_2) \ (MOD\ N) \quad k_1=0,1,..,r_2-1 \quad k_2=0,1,..,r_1-1 \ ..... \ (2)$$

$$n=(s_1 r_2 n_1+s_2 r_1 n_2) \ (MOD\ N) \quad n=0,1,..,N-1 \ ... \ (3)$$

where

$$n_1=n(MOD\ r_1) \quad n_1=0,1,..,r_1-1$$
$$n_2=n(MOD\ r_2) \quad n_2=0,1,..,r_2-1$$

and

$$s_1 r_2=1(MOD\ r_1)$$
$$s_2 r_1=1(MOD\ r_2)$$

Using equation (2) and (3) , equation (1) can be rewritten in a two dimensional array format as follows:

$$X(n_1,n_2)=\sum_{k_2=0}^{r_1-1}\sum_{k_1=0}^{r_2-1} W_N^{(k_1 r_1+k_2 r_2)(s_1 r_2 n_1+s_2 r_1 n_2)} x(k_1,k_2) \quad .... \ (4)$$

Since

$$W_N^{r_1 r_2 k_1 n_1 s_1}=W_N^{r_1 r_2 k_2 s_2}=1$$

equation (4) can be simplified to :

$$X(n_1,n_2)=\sum_{k_2=0}^{r_1-1} W_{r_1}^{n_1 k_2} A(n_2,k_2) \quad .... \ (5)$$

Where

$$A(n_2,k_2)=\sum_{k_1=0}^{r_2-1} W_{r_2}^{k_1 n_2} x(k_1,k_2) \quad .... \ (6)$$

That is, the N-point DFT can be viewed as r_1 r_2-point-DFT's follow by r_2 r_1-point-DFT's. If r_1 and r_2 point DFT computations require M_1 and M_2 operations, respectively, the N-point DFT can be obtained in $N(M_1/r_1+M_2/r_2)$ operations. The flow diagram of a six point PFA computation is shown in Figure 1. Notice that the interconnections between 2 point transform stage and 3 point transform stage have crossovers. For an N point PFA which has many mutually primed factors, the number of interconnections between stages is large, and moreover, they are different from one stage to another. Thus, crossover pattern is very irregular. This is inadmissible for WSI implementation.

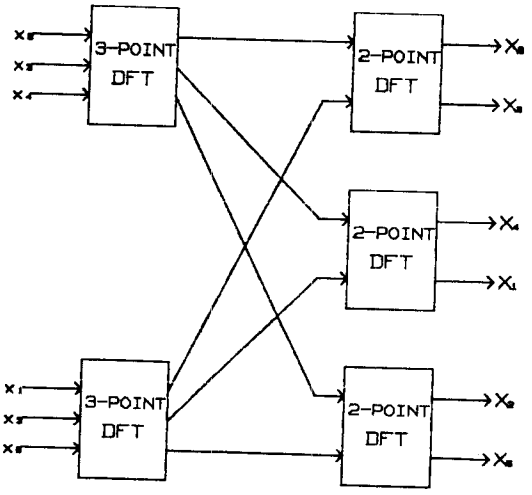

Fig. 1. flow-graph of 6-point DFT computation using PFA

In the following, we will show a method which can reduce the irregular interconnection burden in the PFA. The basic idea here is to first transform the DFT equation into a one dimensional linear equation with shorter length DFT transforms. This will still preserve the interconnection regularity of the one dimensional linear equation. We will then represent this equation into a hardware flow diagram form so that we can easily identifying the independent data paths. Each independent data path can be arbitary preordered without affecting the computation result. Finally, we will combine

the one-dimensional linear processor array with preordered data sequence into multi-dimensional array. We will use an N=42 (42=7*3*2) point DFT as an example from now on. Rewrite eqation (1) in the following:

$$X_n = \sum_{k=0}^{N-1} x_k W_N^{nk}$$

If $N=r_1*r_2$ and $GCD(r_1,r_2)=1$

$$k=(k_1 r_1+k_2 r_2) \text{ (MOD N)}$$

then

$$X_n = \sum_{k_2=0}^{r_1-1} \sum_{k_1=0}^{r_2-1} x(k_1,k_2)\, W_N^{(k_1 r_1+k_2 r_2)n}$$

$$= \sum_{k_2=0}^{r_1-1} W_N^{k_2 r_2 n} \sum_{k_1=0}^{r_2-1} x(k_1,k_2)\, W_N^{k_1 r_1 n}$$

$$= \sum_{k_2=0}^{r_1-1} W_{r_1}^{k_2 n} \left(\sum_{k_1=0}^{r_2-1} x(k_1,k_2)\, W_{r_2}^{k_1 n} \right) \quad \ldots (7)$$

Based on equation (7) on the last page, we can derive the 42 point DFT equation as follows:

$$\begin{aligned}
X_n = &[(x_0 + x_{21} W_2^n) W_3^0 + (x_{14} + x_{35} W_2^n) W_3^n + (x_{28} + x_7 W_2^n) W_3^{2n}] W_7^0 + \\
&[(x_6 + x_{27} W_2^n) W_3^0 + (x_{20} + x_{41} W_2^n) W_3^n + (x_{34} + x_{13} W_2^n) W_3^{2n}] W_7^n + \\
&[(x_{12} + x_{33} W_2^n) W_3^0 + (x_{26} + x_5 W_2^n) W_3^n + (x_{40} + x_{19} W_2^n) W_3^{2n}] W_7^{2n} + \\
&[(x_{18} + x_{39} W_2^n) W_3^0 + (x_{32} + x_{11} W_2^n) W_3^n + (x_4 + x_{25} W_2^n) W_3^{2n}] W_7^{3n} + \\
&[(x_{24} + x_3 W_2^n) W_3^0 + (x_{38} + x_{17} W_2^n) W_3^n + (x_{10} + x_{31} W_2^n) W_3^{2n}] W_7^{4n} + \\
&[(x_{30} + x_9 W_2^n) W_3^0 + (x_2 + x_{23} W_2^n) W_3^n + (x_{16} + x_{37} W_2^n) W_3^{2n}] W_7^{5n} + \\
&[(x_{36} + x_{15} W_2^n) W_3^0 + (x_8 + x_{29} W_2^n) W_3^n + (x_{22} + x_1 W_2^n) W_3^{2n}] W_7^{6n}
\end{aligned}$$

To equation (7) it corresponds a flow diagram as shown in Figure 2:

The data flow is as follows: Input data x_0 and x_{21} from input bus. Multiply the input data with different weighting factors W_2^0 and W_2^1 in each 2-point Complex Multiply-Accumulator (CMAC) box. The output from each 2-point CMAC box are stored in all the 3-point CMAC boxes below it. After three sets of input pairs x_0 and x_{21}, x_{14} and x_{35}, x_{28} and x_7 had been processed by the 2-point CMAC, we start to activate all 3-point CMAC boxes in the second row. At the same time, the 2-point CMAC boxes in the first row start to input x_6 and x_{27}, and then, repeat the above process. The 3-point CMAC boxes will put their results into the 7-point CMAC. After seven 3-point CMAC had been calculated, we can start to do 7-point CMAC. All the steps above continue again and again until all 42 output points come out from the seven point stage.

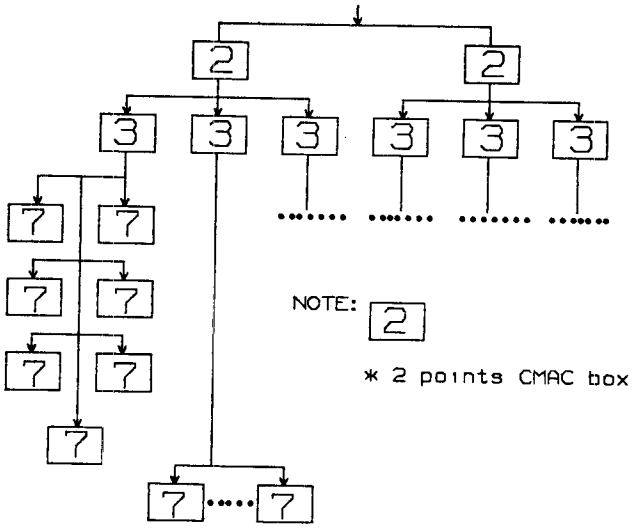

Fig. 2. 42 point PFA hardware flow diagram

We can see that the structure of the diagram looks like a tree. Each branch from the same parent can operate independently. The order of the branch calculation is not important. So, if we place a memory between two stages, we can schedule the computations of the next stage.

The problem with the flow diagram shown in Figure 2 is that each block includes a complicated Complex Multiply-Accumulator (CMAC) and a certain amount of memory. The CMAC boxes from the same parent on the same stage are weighted just like the DFT calculation. There are many existing special DFT computation architecture, so it is better to replace the CMAC blocks with such a DFT processor.

Based on the above two modifications of the flow diagram, we can come up with the architecture shown in Figure 3, which is a processor pipeline with shorter length DFT blocks, and memory between every two DFT blocks. The addressor is basically composed of a few variable length Modular N counters and it will take care of all the necessary sequencing of data. The block diagram of this is shown in Figure 3:

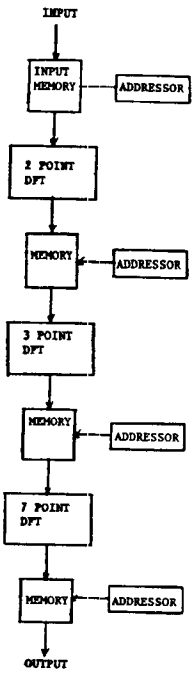

Fig. 3. 42 point PFA pipeline processor

The addressor in Figure 3 is not desirable in WSI implementation because it introduces a few problems. First, because of semiconductor process yield consideration, we need to put in many redundant processor, memory and addressor cells. Also, we need a source selection method to select and connect good cells. Second, because we need to put in many redundant cells, the horizontal address line accross the wafer will introduce a large amount of crossover. Third, memory defect can occur, and this disrupts the flow of data.

Because independent data sequences can be calculated in arbitrary order, we can prearrange the input data sequence in a sequential order to eliminate the need for the addressor. However, a change in the addressing scheme for the memory between two stages influences the addressing scheme for the memory at subsequent stages. We will end up with a scrambled output sequence, but the output addressing can be taken care by precalculating the address and storing it in a ROM.

To slove the memory defect probelm, we propose to break the memory into smaller modules. Because of the defect density in the silicon, the probability of a defective large memory is higher than a small memory. Also, because of the redundancy requirement of the processor during run time, we need less area to provide redundancy for smaller memory modules than larger memory modules.

Based on the above discussion, a multi-dimensional indexing scheme is proposed. This scheme will enable us to use only first-in-first-out shift register(FIFO) instead of random access memory, thus no addressing is needed. It also enables us to break a large shift register into smaller shift registers to avoid the memory defect problem. By imposing the constraint of FIFO data flow between stages, it will result in some input and output addressing sequences, stored into input and output tables respectively. The input table will address the input buffer memory to load input data into the array processor. The output table is used to unload output data in the proper order. The sequence of operation corresponding to the scheme is shown in figure 4.

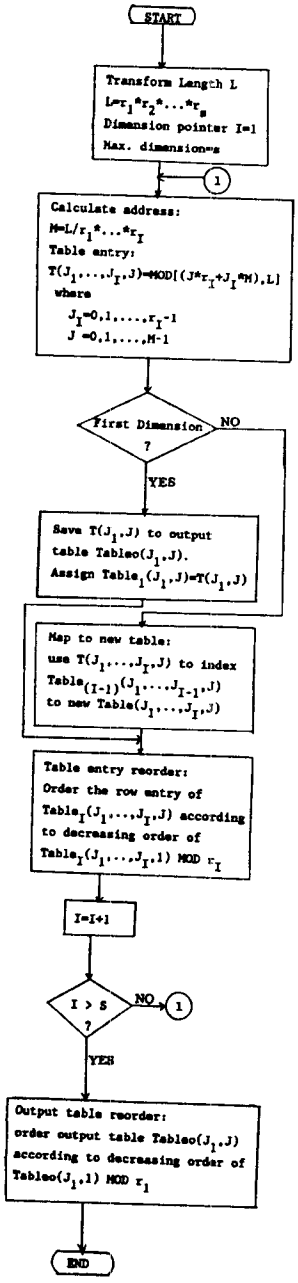

Fig. 4. Flow Chart of Multidimensional Indexing Scheme for PFA processor

3. PROCESSOR DESIGN

Based on the addressing scheme described in section 2, we can improved the architecture shown in Figure 3 by replacing the memory modules with shift registers. The new architecture is shown in Figure 5.

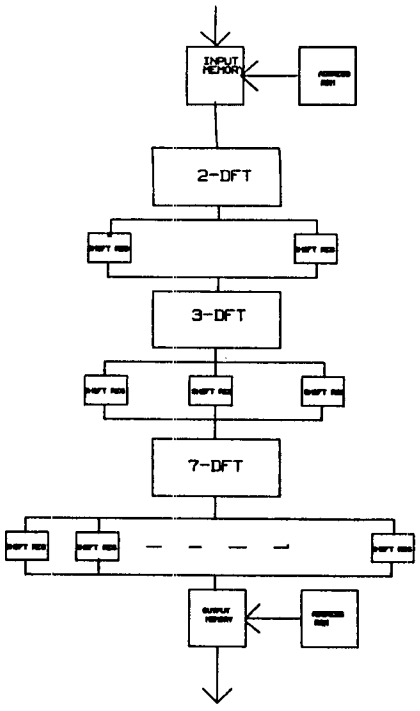

Fig. 5. 42 point PFA processor with shifter register

To design each computation cell, we want to find a way to compute short length DFT as fast as possible. By taking advantage of the symmetry of SIN and COS in the coefficient, the Winograd Short Length DFT (WSLDFT) and it's architecture has been derived <Pe81>. The architecture of the VLSI cell used in this processor is shown in Figure 6. Table 1 shows the number of cycles needed for each short length DFT.

According to Table 1, an N point WSLDFT take about 4N to 6N cycles to be processed. For the purpose of balancing the number of cycles needed in each stage so that all the processors are used most of the time, we can increase the number of processors on each stage by four to six times. A block diagram of the proposed PFA ARRAY PROCESSOR is shown in Figure 7.

To initialize the transform, 42-point sampled data is loaded into the input memory. The input addressing ROM with the address shown in Figure 8 will be used to address the input memory to load data into array processor according to the order of P.11, P.12, P.13 and P.14. While P.11 processor is calculating SLDFT, P.12 takes in data from

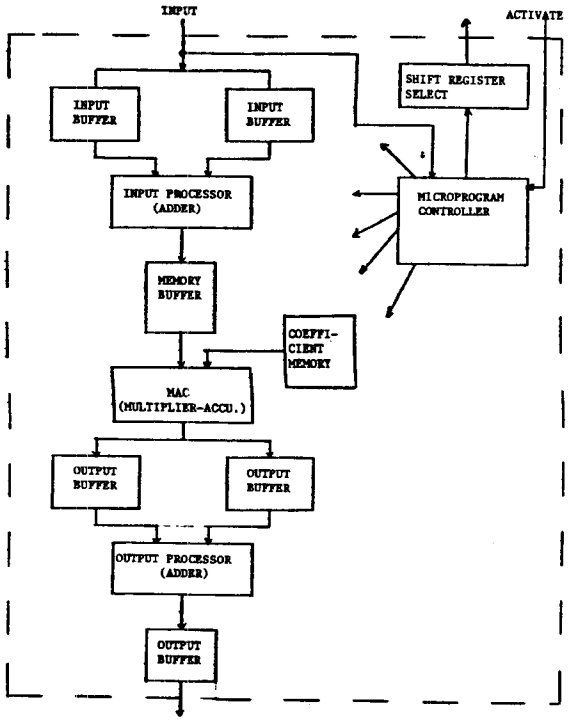

Fig. 6. Winograd Short Length DFT Processor architecture

Transform Size (N)	No. of cycles (C_N)	No. of cycles per point
2	8	4.00
3	12	4.00
4	16	4.00
5	20	4.00
6	24	4.00
7	28	4.00
8	32	4.00
9	36	4.00
10	40	4.00
11	62	5.64
12	48	4.00
13	68	5.23
14	72	5.15
15	60	4.00
16	64	4.00

Table 1. Number of cycle needed by N cycle DFT on Fig. 6's processor

input and starts to calculate SLDFT. Processor P.13 follows P.12 to input data and processor P.14 follows P.13. After P.14 finished loading, P.11 will start input and this sequence will be repreated until all the data has been processed by the first stage. When processor P.11 finishes a SLDFT computation, the output data will be output to shift registers S.11 and S.12. The kth transformed point will be put into the S.1k shift register block. All the other processors will follow the same output order. When all the length 2 DFT computations have been done, the two point DFT transformed output data will be in the S.11 and S.12 shift registers. Stage two can start to process 3 point DFT by accessing data from the shift register block S.11 and S.12. All the stage n access data from shift registers according to the equation :

$$Count=Factor(1)*Factor(2)*...*Factor(n)$$

The processors will access data from S.11 until counter in the SLDFT processor equal to Count-1, then processor will start to access from S.12. The output of the 3 point DFT will be put into the shift register below it and the 7 point DFT will take input data from that shift register. The output from the 7 point DFT will be put into a Random Access memory, and the precalculated output ROM which contains the output address in Figure 8 will be used to address the output sequence.

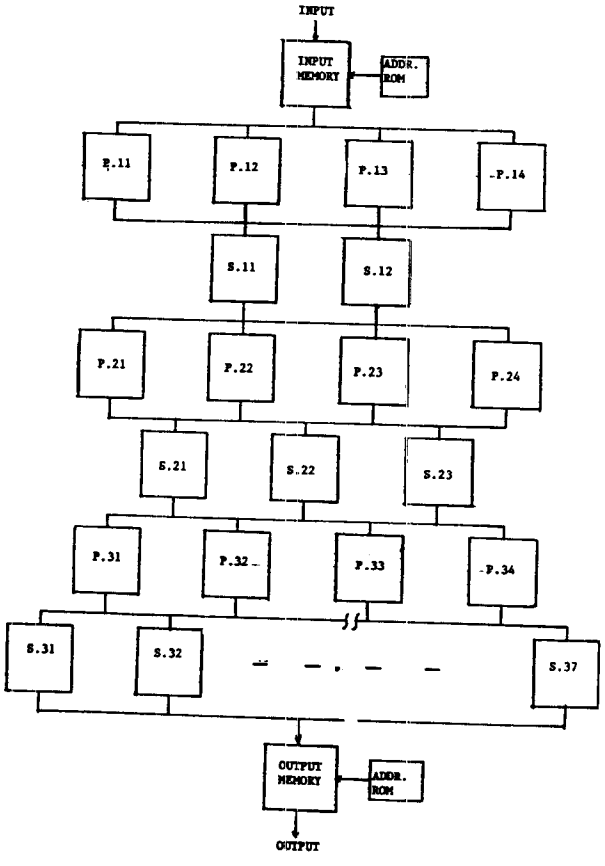

Fig. 7. 42 Point PFA array processor

An example of a 4 stage PFA array processor is shown in Figure 9. It can cover DFT length up to 32000 points with a maximum 16 points SLDFT. The shift register has to be designed to be variable length so that variable length DFTs can be computed on the same processor.

OUTPUT REFERENCE TABLE:

0	14	28	21	35	7
6	20	34	27	41	13
12	26	40	33	5	19
18	32	4	39	11	25
24	38	10	3	17	31
30	2	16	9	23	37
36	8	22	15	29	1

INPUT REFERENCE TABLE:

ARRAY		0
0	21	
28	1	
14	35	
ARRAY		1
36	15	
22	1	
8	29	
ARRAY		2
30	9	
16	37	
2	23	
ARRAY		3
24	3	
10	31	
38	17	
ARRAY		4
18	39	
4	25	
32	11	
ARRAY		5
12	33	
40	19	
26	5	
ARRAY		6
6	27	
34	13	
20	41	

Fig. 8. 42 Point PFA Array Processor I/O Address Table

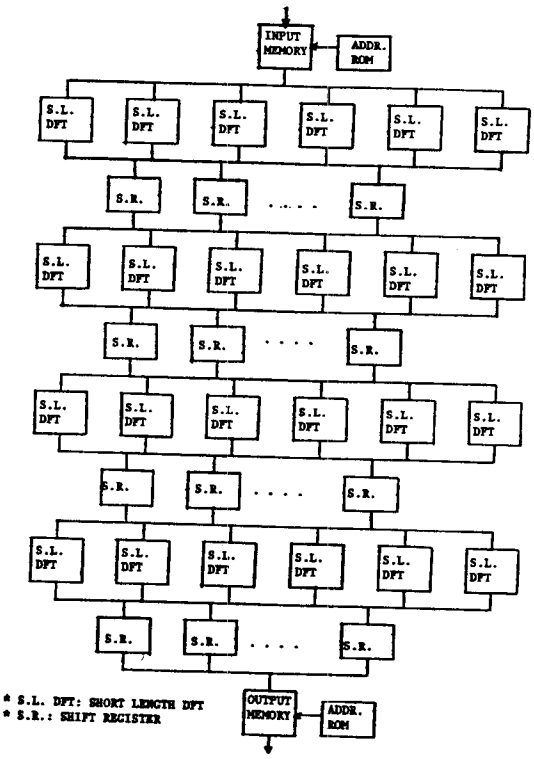

Fig. 9. PFA array processor

4. COMPARISON

The advantage of this processor are:

a. Highly pipeline structure

Can compute an N point DFT in N clock cycles throughput. Further increase in throughput of the DFT system, we can just use as many PFA array processor wafers as needed. To increase the system real-time response time can be achieved by using crossbar switch to overcome the single bus bottleneck, but the addressing scheme will have to be modified.

b. Independent computation cell

Each cell in the same stage is running independently. Any time should one cell fail, another build-in redundant cell can take over its job of this processor without modifying the interconnection.

c. Insensitive to IC process yield and highly fault tolerable

If enough redundant cells are put on each stage, we can be sure that each stage can have enough working cells to make the system function correctly after the IC processing steps. Some laser welding techniques may be used after a good die has been found. Even after it has been used in the field, we can design some built in test capability to detect bad cell and switch between dead and good cells automatically without changing the function of the processor.

d. Very few global control lines

Because each processing cell has it's own internal control, and all the cells hand shake with asynchronous flag signals. Except for the ring signal passed to flag neighbor cell to start input, few global controls are needed. Reduced global control means better system performance and less break down bottleneck.

e. Self contained

The processor has all the control signal necessary to do the DFT computation, so it can be connected to an A/D converter directly. Because this processor is fast enough to take input from an A/D converter, no other control circuit is necessary.

The processor we presented in this paper shows many advantages over other types of DFT array processors (<Gr70>,<Tr82>,etc) when built with the WSI technology. First, the required number of processor is very little compared to the one suggested in <Gr70>. Second, the architecture is highly structured and regular, and the interconnections are just one directional complex bus compare to the exchange network needed in <Gr70>. These features make it possible to put this array on a single wafer with enough room to put extra redundant processors. Third, minimum global control and addressing compare to <Tr82>. Since each processing cell is independent, the array is highly fault tolerable. So, the processor can be manufactured in large quantity, and the processed wafer is used as the final product.

5. CONCLUSION

The problem with this processor is that the bus between stages may be too long when many redundant processors are put on the same bus , and because of this, the delay on a silicon wafer may be too long. This problem is a violation of the basic systolic principle. By using only metal layer for busing, high driving power precharging and insulating substrate (CMOS-SOS, etc.) technology could be solutions to this problem.

It's throughput rate of one point per cycle means that we can process sampled data from A/D converter as fast as it comes in. The only limitation in here is the physical device delay. We desinged our system based on the assumption that the clock rate is 10 MHz, so the sampling data rate is 100 nsec per point. Currently, the throughput of the processor is fast enough for applications like: radar imaging, IR imaging, communication coding and others. With some modifications, The throughput can be increased to accomerdate applications such as navigation and guidance. As the technology growth, the system will become very attractive to many applications. When GaAs VLSI become available, this type of tightly packed WSI system will produce a GHz DFT processor.

6. REFERENCE

<Co67>
Cooley J., Lewis P., Welch P.," Historical Notes on the Fast Fourier Transform", Audio Electroacoust., Vol. AU-15, June 1967

<Pe81>
Peterson A., Private Communication, Standford University, 1981.

<Ga82>
D. D. Gajski, A. H. Saueh, and J. A. Wisniewski, "Iterative Algorithms for Tridiagonal Matrices on A WSI-Multiprocessor", 1982

<Gr70>
H. L. Groginsky and G. A. Works, "A Pipline Fast Fourier Transform" IEEE Trans. on Computer, Vol. C-19, No. 11, November 1970

<Tr82>
B. L. Troutman, G. W. Mclver, W. E. Kingsley, B. H. Whaien, " A 2um CMOS/LSI 32-point Fast Fourier Transform Processor", IEEE Inter. Solid-State Circuits Conference, 1982