16비트 마이크로 컴퓨터를 사용한 FFT 연산속도 향상에 관한 연구*

金錫載・池石根・金千德
韓國海技研修院, 釜山水產大學
(1990년 1월 31일 접수)

A Study for Improving the Computing Speed of FFF Using 16bit Microcomputer

Suk-Jae KIM , Suk-Kum JEE and Chun-Duck KIM
Korea Marine Training and Research Institute, National Fisheries University of Pusan
(Received January 31, 1990)

The processing efficiency of the special purpose hardware which is designed and implemented for the FFT caculation was investigated in this paper. This hardware equipment was consisted of LSI chips of four high speed multiplier and adder/substractor, and was interfaced with the 16bit microcomputer(NEC PC-9801E). The FFT processing time by this hardware equipment was improved approximately 4.8 times by the co-processor(Intel C8087-3).

서 론

디지탈 신호처리는 외부에서 입력된 신호를 디지탈 연산처리하여 정보를 축출하거나 해석하기 쉬운 형태로 변환하는 것으로 음향학, 의용공학, 통신, 음성인식 및 합성 등의 연구에 활발히 이용되고 있다[1)~3)]. 특히 푸리에 변환은 입력 신호의 스펙트럼을 분석하는 것으로 오래전부터 널리 이용되고 있는 신호처리 방법중 하나이다.

푸리에 변환의 연산은 많은 데이타를 반복연산하므로 고속연산이 필요하다[2),3)]. N개의 데이타를 푸리에 변환할 경우 N^2회 승산과 N(N-1)회 가감산의 많은 연산이 필요하므로 이 연산수를 줄이는 여러가지 고속푸리에 변환 알고리즘이 개발되었다. 특히 N이 $N=r^m$혹은 $N=N_1\times N_2\times N_m$($N_1$, N_2,..., N_m은 서로 소)인 특수한 값일 경우 효과적인 고속 푸리에 변환 알고리즘들이 고안되었다. N이 m개의 서로 소의 곱인 후자의 경우 Winograd 푸리에 변환[4)]이나 Prime factor 푸리에 변환[5)] 등의 알고리즘이 개발되어 사용하고 있으나 프로그램이 복잡한 단점이 있다. $N=r^m$인 경우 Cooley와 Tukey의 알고리즘[6)]과 Preuss의 알고리즘[7)]으로 크게 2가지로 대별할 수 있다. 이 중 Preuss의 알고리즘은 대칭성과 비대칭성을 이용하여 고안된 것으로 복잡한 단점이 있으나, Cooley과 Tukey가 고안한 알고리즘은 일반적으로 알려진 FFT이며 프로그램작성이 용이하고 분석 가능한 데이타 수가 넓기 때문에 많이 사용되고 있다.

본 논문에서는 대중화되어 널리 사용되고 있는 16비트 마이크로 컴퓨터로 디지탈 신호처리를 하기위해 Cooly과 Tukey가 고안한 알고리즘을 이용한 FFT 전용 고속 연산장치를 설계 제작하였다. 이 장치는 16비트 마이크로 컴퓨터에 부가하

* 이 논문은 1986년도 한국과학재단 연구비지원에 의해 연구되었읍니다.

여 실험한 결과 보조프로세서(Intel C8087-3)만을 사용할 때보다 연산속도가 크게 향상되었다. 따라서, 디지탈 신호처리에 16비트 마이크로 컴퓨터로 연구할 때 FFT 전용 연산장치를 부가하므로서 경비절약이나 이동의 가변성이 쉬워 크게 이용될 것으로 기대된다.

FFT 연산을 위한 고속 연산장치의 구성

1. 고속 푸리에 변환(FFT) 알고리즘

N개의 이산적(Discrete)인 복소 시간신호 x(n)과 그 DFT X(k)의 함수는 (1), (2)식으로 표현된다[1].

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

$$x(n)=\frac{1}{N}\sum_{n=0}^{N-1}X(k)\,W_N^{kn},\quad n=0,1,\cdots,N-1 \quad (2)$$

여기서 $W_N=\exp(-j(2/N))$이다.

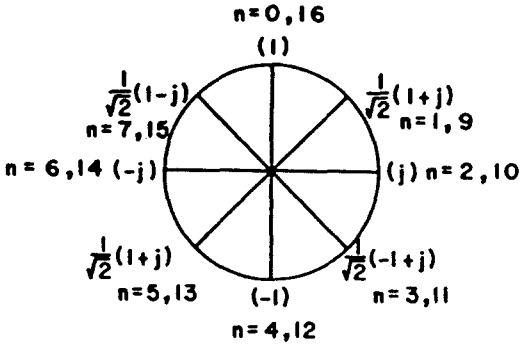

Fig.1. Twiddle factor of complex number(W_N, N=8).

식(1)에서 k번째의 주파수에 대한 푸리에 변환을 계산하기 위해서는 x(n)과 W_N의 복소수 곱을 N회, 복소수 덧셈을 N-1회 해야 한다. 그래서 N개의 스펙트럼을 계산하기 위해서 N^2회의 복소수 곱셈과 N(N-1)회의 복소수 덧셈이 필요하나 복소수 회전인자(Twiddle Factor)W_N^{kn}의 주기성과 대칭성을 이용하면 효과적으로 계산할 수 있다[1),3),8)]. N=8일 경우의 W_N의 주기성과 대칭성을 Fig. 1에 나타냈다. 여기서 kn이 N보다 크면 kn-N일 때의 값과 같은 주기성과 kn일 때의 값과 180°의 위상차가 있는 kn+N/2일 때의 값이 절대값은 같고, 부호만 다른 대칭성을 볼 수 있다. 따라서 특수한 N값에 대해 회전인자의 주기성과 대칭성을 이용한 효과적인 알고리즘이 많이 개발되었다. 특히 $N=r^m$ 즉 멱급수 형태인 경우에 개발된 Cooley와 Tukey의 알고리즘은 프로그램의 작성이 용이하고 계산속도가 빠르다. 이 경우 기수(Radix) r이 2인 2의 멱급수의 데이타 수를 가질 때 FFT의 연산이 규칙적이고 유용성이 있어 전용 연산장치 구성이 쉽다[3].

이 Cooley, Tukey의 FFT 알고리즘은 크게 두 가지로서 시분할(Decimation-In-Time ; DIT)법과 주파수분할(Decimation-In-Frequency ; DIF)법이 제안되고 있다[1),8)]. 이 중 데이타가 주파수 영역에서 순서적인 DIT 법을 사용하여 (1)식을 나타내면 (3)식과 같이 된다.

$$X(k)=\sum_{n\ even}x(n)\,W_N^{kn}+\sum_{n\ odd}x(n)\,W_N^{kn} \quad \cdots(3)$$

여기서 n=2r(even), n=2r+1(odd)를 대입하면 (4)식이 된다.

$$\begin{aligned}X(k)&=\sum_{r=0}^{(N/2)-1}x(2r)\,W_N^{2rk}+\sum_{r=0}^{(N/2)-1}x(2r+1)\,W_N^{(2r+1)k}\\&=\sum_{r=0}^{(N/2)-1}x(2r)\,(W_N^2)^{rk}+W_N^k\sum_{r=0}^{(N/2)-1}x(2r+1)\,(W_N^2)^{rk}\end{aligned} \quad \cdots\cdots(4)$$

여기서 $W_N^2=W_{N/2}$이므로 대입하면 (5)식이 된다.

$$\begin{aligned}X(k)&=\sum_{r=0}^{(N/2)-1}x(2r)\,W_{N/2}^{rk}+W_N\sum_{r=0}^{(N/2)-1}x(2r+1)\,W_{N/2}^{rk}\\&=G(k)+W_N^kH(k)\end{aligned} \quad \cdots\cdots(5)$$

(단, $G(k)=\sum_{r=0}^{(N/2)-1}x(2r)\,W_{N/2}^{rk}$,

$H(k)=\sum_{r=0}^{(N/2)-1}x(2r+1)\,W_{N/2}^{rk}$ 이다.

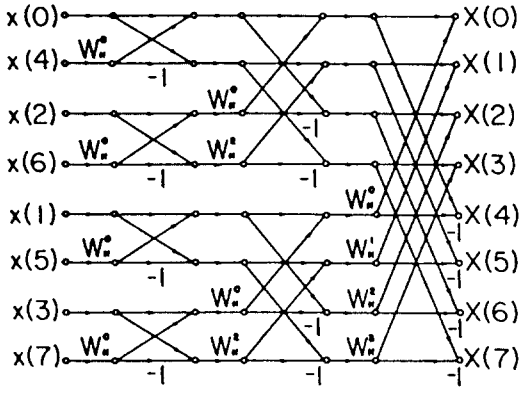

Fig.2. Flow graph of eight-point DFT using algorithm of decimation-in-time.

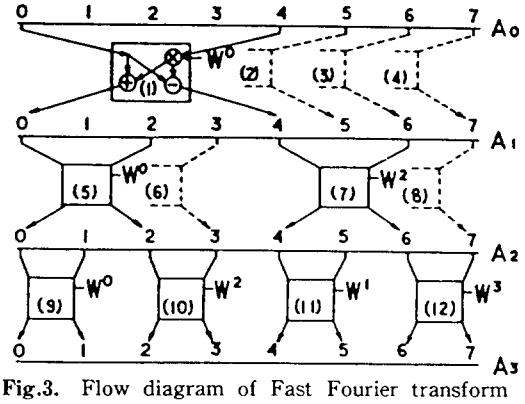

Fig.3. Flow diagram of Fast Fourier transform for N=8.

이 G(k)와 H(k)를 (3)식에서 (5)식의 과정으로 계속 반복하면 결국 2점 DFT 까지 나누워진다. Fig. 2에 DIT 알고리즘을 이용한 8점 DFT 계산의 신호선도(Flow Graph)를 나타냈다.

2. 고속 연산 장치의 구성

특수목적의 FFT 고속 연산장치에 대한 논의는 디지탈 신호처리에서 실시간 신호처리라는 문제로 부터 기인한 것이다[9)~11)]. 더 빠른 고속 연산처리는 Cooley과 Tukey의 FFT 알고리즘의 사용함과 동시에 이 알고리즘을 실행하는 하드웨어인 전용 연산장치로 실현할 수 있다. 이 하드웨어의 설계는 성능, 가격, 기술적 요건등 여러가지 조건들에 의해 정해진다[9)]. 우선 데이타 수가 2의 멱수인가 즉 N이 어떤값인가에 의해 연산장치의 수행속도 및 구조적 복잡성 등이 결정된다. 데이타의 기수가 8인 경우 알고리즘 실현에 최적인 반면 기수가 2인 경우 알고리즘이 간단하고 규칙적이고 때문에 하드웨어 실현에 있어서 큰 장점을 가진다. 그리고 입력신호를 실수 혹은 수정된 복소수 중 어느 것으로 할 것인가도 결정해야 한다.

본 연구에서는 여러가지 하드웨어 구성에 장점이 있고 다른 알고리즘에도 사용할 수 있는 조건들을 선택했다. 기수는 2로 하고 입력 데이타는 복소수로 했으며 구조는 순차적 처리기(Sequential Processor) 형태로 결정했다. 이를 이용하여

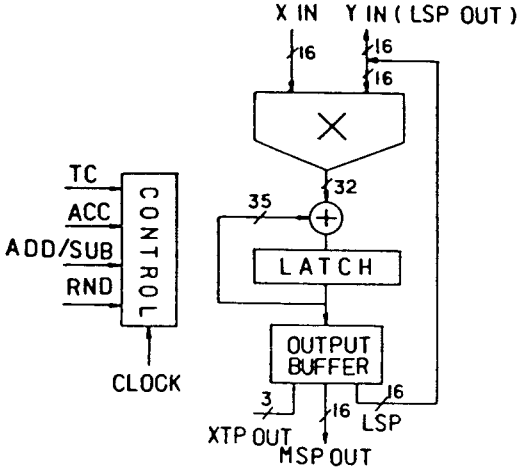

Fig.4. Functional diagram of high speed multiplier(TRW LSI Products TDC1010 J).

8점 DFT 계산을 할 경우의 과정을 Fig. 3에 나타냈다. Fig. 3과 같이 FFT 알고리즘은 2점 DFT로 구성되므로 이 연산을 하드웨어로 구현했다. Fig. 4는 하드웨어 구성에 사용된 고속 승산기(TRW사 1010J)의 내부구조이다[12)]. 승산한 후 가감산하는 고속 승산기의 구조때문에 DIT 알고리즘을 사용하게 된 동기가 되었고 Fig. 5에 DIT 알고리즘의 2점 DFT 연산 흐름도를 나타냈다. 이 연산을 복소수 입력신호에 대해 표현하면 (8)식과 같이 된다.

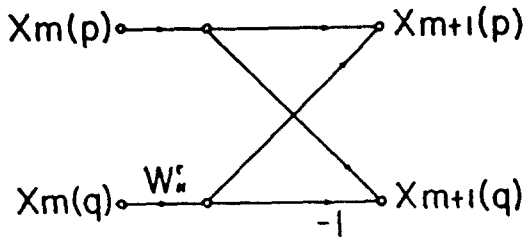

Fig.5. Flow graph of 2 point DFT butterfly computation.

$$A \pm B \times W_N = (Re[A] + jIm[A]) \pm (Re[B] + jIm[B]) \times (Re[W_N] + jIm[W_N])$$
$$= Re[A] \pm (Re[B] \times Re[W_N]) \mp (Im[B] \times Im[W_N]) \pm jIm[A] \pm j(Re[B] \times Im[W_N]) \pm j(Im[B] \times Re[W_N])$$
$$= \{Re[A] \pm (Re[B] \times Re[W_N] - Im[B] \times Im[W_N])\} \pm j\{Im[A] + (Re[B] \times Im[W_N] + Im[B] \times Re[W_N])\} \cdots\cdots (8)$$

따라서 2점 버터플라이 계산 결과는 (9), (10) 식과 같다.

$$Re[A'] + jIm[A'] = \{Re[A] + (Re[B] \times Re[W_N] - Im[B] \times Im[W_N])\} + j\{Im[A] + (Re[B] \times Im[W_N] + Im[B] \times Re[W_N])\} \cdots\cdots (9)$$

$$Re[B'] + jIm[B'] = \{Re[A] - (Re[B] \times Re[W_N] - Im[B] \times Im[W_N])\} + j\{Im[A] - (Re[B] \times Im[W_N] + Im[B] \times Re[W_N])\} \cdots\cdots (10)$$

위의 (9), (10)식에서 공통적인 계산항을 이용하면 4개의 승산기와 6개 가감산기로 2점 DFT 전용연산장치를 구성할 수 있다. 여기에 6개의 입력 레치와 4개의 출력 레치를 사용한 전체 구성도를 Fig. 6에 나타냈다. Table 1과 같이 각 레치 및 제어부분을 각 어드레스선에 연결하여 구성했다. 이 고속연산장치를 인터페이스한 16비트 마이크로 컴퓨터(NEC PC-9801E)에 &HD0-&HDF 어드레스 I/O포트를 사용했다. 가감산

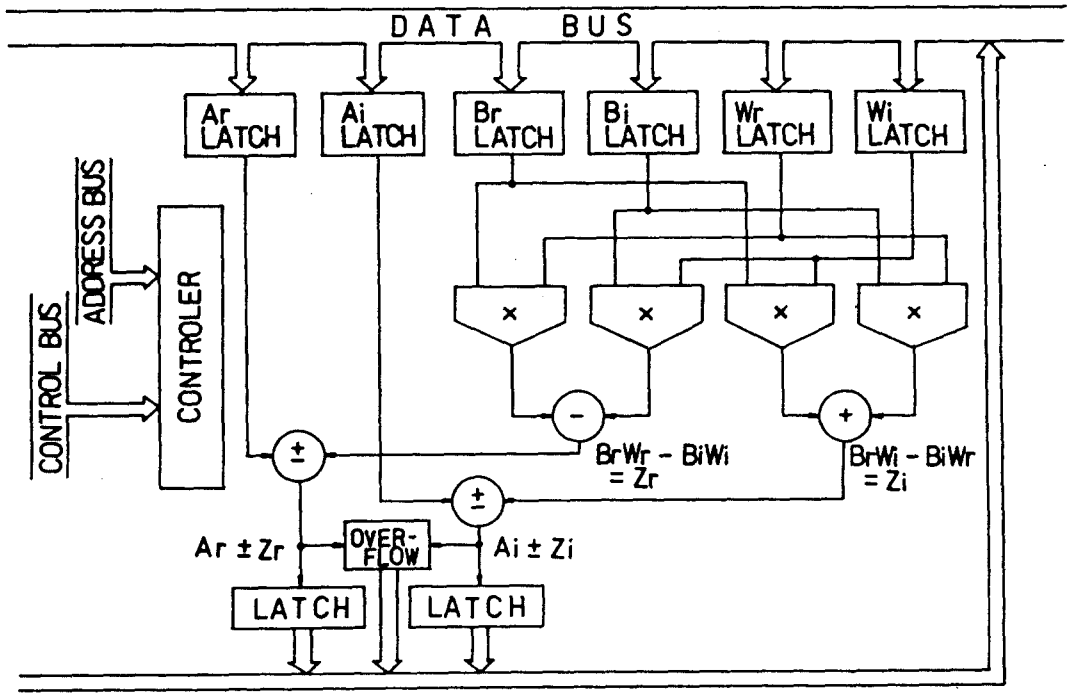

Fig.6. Overall diagram of the implemented hardware.

Table 1. Function of the address lines

Address			function			
A3	A2	A1	Write		Read	
0	0	0	Wr	register	Ar	register
0	0	1	Wi	register	Ai	register
0	1	0	Ar	register	Br	register
0	1	1	Ai	register	Bi	register
1	0	0	Br	register	overflow detection	
1	0	1	Bi	register	·	
1	1	0	reset		·	
1	1	1	·		·	

기는 고속 로직게이트 74S381, 4개와 74S182, 1개로 구성하여 빠른 연산을 하도록 했고, 오버플러워(Overflow)가 발생되면 바로 입력데이타를 수정하도록 구성했다. 6개의 A, B 및 W_N의 실수 데이타가 주어지고 A', B'의 결과가 계산되는 소요시간은 약 230 *n sec* 걸렸다.

3. 고속연산장치 운용프로그램 및 평가

DIT 알고리즘을 사용한 8점 DFT를 나타낸 Fig. 2에서 Fig. 3과 같이 2점 DFT 연산을 하드웨어로 구현한 경우 이를 프로그램으로 제어하기 위해 블럭(Block)과 스테이지(Stage)로 구분하여 나타낸 것이 Fig. 7이다. N개의 데이타 메모리에서 2점 DFT 연산을 할 데이타쌍 A, B와 그 회전인자 W_N를 고속 연산장치로 보내어 연산한 후 원장소에 연산결과 A', B'를 가져온다. 한

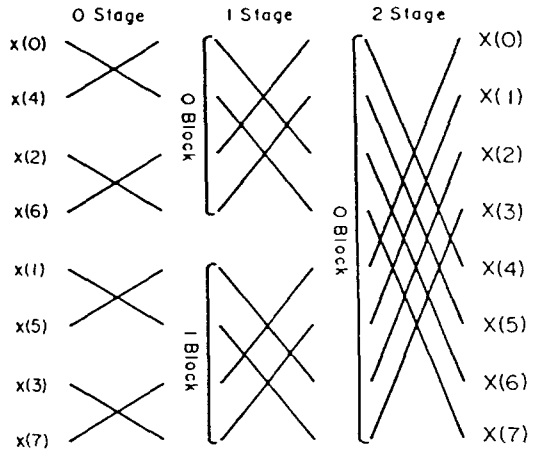

Fig.7. Flow diagram of butterfly computation using algorithm of decimation-in time.

쌍의 데이타 A, B의 어드레스와 회전인자 W_N의 값을 결정하기 위해 스테이지 카운터, 블럭 카운터 및 2점 DFT 연산반복 카운터로 제어하도록 프로그램한다. 예를 들어 $N=2^3$의 경우 블럭 및 스테이지와 블럭내 2점 DFT연산 반복수의 관계를 Table 2에 나타냈다. 이 DIT 알고리즘을 알

Table 2. Number of blocks and stages in eight-point DFT

Stage	1 Stage	2 Stage	3 Stage
Number of Blocks	4	2	1
Number of Iterations	1	2	4

```
Procedure  FFT;
  N = number of data;
  M = log value of data number; // N = 2^M//
  augmentation = 512;
  Shuffling( ); // bit-reverse-order program //
  stage counter = M;
  number of block = n / 2;
  number of iteration = 1;
  repeat
    block counter = number of block;
    data address one = 0;
      repeat
        factor address = 0;
        iteration counter = number of iteration;
          repeat
            data address two = data address one + number of iteration;
            read data of address one and address two;
            butterfly computation;
            write data of address one and address two;
          until ((stage counter = 0); increment data address one;
                  factor address = factor address + augmentation;); // block end ? //
```

```
      until ((block counter = 0); data address one = data address two + 1;);// stage end ? //
    Overflow (); // if detect overflow then processing program //
  until ((stage counter = 0); number of block = number of block/2;
         augmentation = augmentation/2; number of iteration = number of iteration * 2;);
end FFT;

Procedure Suffling;
  for (data address = 0) to (N-1) by increment do
    for (number of bit = 0) to (M-1) by increment do
      if (data address AND (2**number of bit)) > 0
      then bit reverse order address = (bit reverse order address OR (2**(M-number of bit));
    end;
    data  exchange  data address  and  bit reverse order address;
  end;
  return;
end.

Procedure Overflow;
  if (read overflow flag > 0) then
     for ( data address = 0 ) to (N-1) by increment do all
        data one bit shift down;
     end;
  return;
end.
```

고리즘 기술언어로 표현하면 다음과 같다.

FFT가 시작되면 데이타수 N과 M을 세트하고 입력 데이타 N개를 역순 비트법(Bit Reverse Order)으로 재배치한다. 스테이지 카운터에 M ($N=2^M$)을 저장하여 각 스테이지가 마칠 때마다 하나씩 감산하고 이것이 0이 되면 FFT가 끝난다. 각 스테이지의 블럭수는 스테이지가 S인 경우 $N\times2^{-s}$이므로 스테이지가 증가할 때마다 반분됨을 알 수 있다. 그리고 블럭내 2점 DFT연산 반복수는 스테이지가 증가하면 배가된다. 테이타 A와 쌍인 B의 데이타 주소는 A의 어드레스에 이 2점 DFT 연산 반복횟수를 가산하여 만든다. 회전인자의 어드레스는 각 2점 버터플라이 연산때 마다 ΔW를 더하여 만들며, 이 ΔW는 스테이지 내에서는 일정하고 스테이지가 증가하면 반분된다.

본 논문의 고속연산장치로 N개 데이타를 FFT 연산할때 각 명령문들의 실행 빈도수를 N의 다항식으로 표현할 수 있고 이를 이용하여 프로그램의 실행속도를 비교 검토할 수 있다. 역순 비트법의 실행으로 log N회 실행수를 첨가시키고 오버플로워는 발생했을 경우만 실행되므로 약2회로 정하여 N다항식을 유도했다. 전체 프로그램에서 log N회 반복되는 것이 9N+5이므로 전체 다항식은 (N log N+2)+(2+log N)×(9N+5)+5=10N log N+18N+5log N+17이 된다.

결과 및 고찰

본 논문에서 구성한 고속연산장치로 6개의 실수 데이타를 연산하여 4개의 데이타를 출력하는데, 즉 복소수 데이타의 2점 DFT 계산에 소요되는 시간은 약250 *nsec*로 메인 클럭이 5*MHz*인 NEC PC-9801 E 컴퓨터에서는 거의 한 클럭에 해당한다. 그래서 다항식 평가에서 승산시간을 거의 고려하지 않아도 가능하나 보조 프로세서로 승산을 실행할 경우 수십 클럭의 시간이 소요되므로 고려해야 한다. 그리고 보조 프로세서의 승산시간은 컴퓨터의 ROM 루틴 승산시간 보다 약 4.7배 정도 빠르다. 보조 프로세서로써 복소수의 2점 DFT 연산을 할 경우 두 실수 데이타를 이동시켜 연산하고 결과를 가져오는 연산이 10회이므로 30회의 데이타 이동이 있고, 10회의 연산중 4회가 승산이고 나머지 6회는 가산 또는 감산으로 되어 있다. 결과적으로 2점 DFT 계산에 소요되는 시간의 전체 N 다항식은 48N log N+6 log N+94 N+15가 된다. 두 다항식의 비 즉 전용 연산장치로 FFT를 실행할때 시간이득비는 약 4.8배가 된다. Fig.8에 전용 연산장치의 1024점 FFT 연산시간을 1로 하여 나머지 연산시간

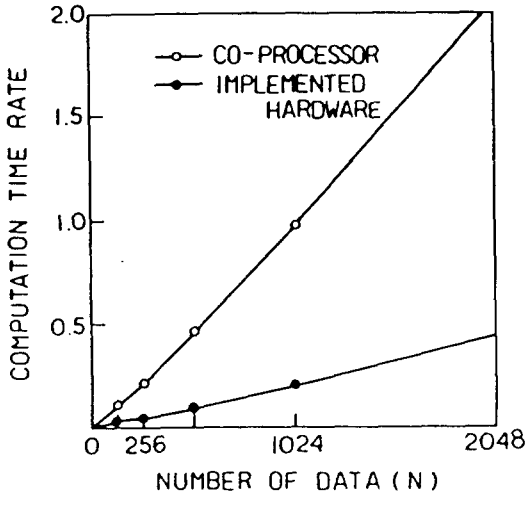

Fig.8. Comparison of FFT processing time for number of data

과 비교하여 표시하였다.

1024점 데이타를 실험적으로 FFT한 결과 전용 연산장치를 사용한 경우는 약 1.05 *sec*, 보조 프로세서 (Intel C8087-3)의 경우 약 3.5 *sec* 걸렸다. 실험의 시간이득비가 적은 원인은 전용 연산장치용 FFT프로그램의 내부에 불필요한 명령이 반복된 것으로 분석되었다. 특히 16비트 마이크로 컴퓨터 메모리 폭은 8비트이므로 2번의 명령을 거쳐 16비트 데이타를 전송하게 되므로 전용 연산장치의 연산시간의 거의 배로 늘게 되었다.

전용 연산장치를 사용하여 FFT 연산을 실행할때 소요된 시간을 분석하여보면 6개의 실수 데이타로 이루어진 복소수 2점 DFT 계산시간은 약 230 *n sec* 이나 6개의 실수 데이타를 I/O포트를 이용하여 전송하려면 약 800 n sec×6개 만큼 시간이 소요된다. 그러고 마이크로 컴퓨터의 I/O 포트를 이용한 데이타 전송은 프로세서 내부의 AX(Accumulator Hexadecimal)레지스터를 거쳐야 하므로 일반적인 메모리 데이타 전송보다 시간 결손이 발생한다. 그러나 전용 연산장치와 마이크로 컴퓨터에 공유 메모리를 두고 어드레스 지정에 의한 데이타의 공유와 FFT의 반복적인 명령들을 파이프 라인 처리나 전용 연산장치의 연산을 병렬처리하므로서 고속처리할 수 있을 것이다.

결 론

본 연구에서는 널리 사용되고 있는 16비트 마이크로 컴퓨터에서 FFT와 같은 디지탈 신호처리를 하기위해 전용연산장치를 구성하여 NEC PC-9801 E에 연결하였다. 특히 FFT 연산은 승산 및 가감산의 2점 DFT로 구성되어 있고 이것이 반복연산된다. 그리고 마이크로 컴퓨터에서 승산은 많은 시간이 소요되므로 외부에 고속 승산기를 이용한 전용연산장치로 연산하므로서 연산속도를 향상시킬 수 있었다.

설계 제작한 전용연산장치를 이용하여 FFT를 실행한 시간은 보조 프로세서 (Intel C8087-3)로 실행한 것 보다 약4.8배 빠르게 나타났다. 따라서 마이크로 컴퓨터에 FFT을 위한 전용연산장치를 구성하여 연결하므로서 연산속도가 크게 향상되어 디지탈 신호처리에 많이 이용될 수 있을 것이다.

참고문헌

1) Oppenheim, A.V. and R.W. Schafer(1975): Digital Signal Processing. Prentice-Hall, 88-121.
2) Oppenheim, A.V. (1978): Application of Digital Signal Processing. Prentice-Hall, 117-161.
3) 松澤英明(1980)：ディジタル信號處理用汎用高速プロセッサの設計にる研究. 日本 東京大學碩士學位論文.
4) Silverman, H.F.(1977): An Introduction to Programing the Winograd Fourier Transform Algorithm (WFTA). IEEE Trans Acoust. Speech Signal Process. ASSP-25(2), 152-165.
5) Kolba, D.P. and T.W. Parks(1977): A Prime Factor FFT Algorithm Using High Speed Convolution. ibid.ASSP-25(4), 281-294.
6) Cooley, J.W. and J.W. Tukey(1965): An Algorithm for the Machine Cacluation of Complex Fouier Series. Math. of Computation 19, 297-301.

7) Preuss, R.D.(1982): Vary Fast Computation of the Radix-2 Discrete Fourier Transform IEEE Trans. Acoust. Speech Signal Process ASSP-30(1), 595-607.
8) Rabinar, L.R. and B. Gold(1975): Theory and Application of Digital Signal Processing. Prentice-Hall, 573-626.
9) Bergland, G.D.(1969): Fast Fourier Transform Hardware Implementation-An Overview. IEEE Trans. Audio Electroacoust Au-17, 104-108.
10) 安倍正人, 城戶健一(1985): パーソナルコンピュータの信號處理ソフトウェア. 日本音樂學會誌 41(9), 630-636.
11) Pomerleau, A., M. Fourier and H.L Buijs(1976): On the Design of a Real Time Modular FFT Processor. IEEE Trans Circuits and Syst. CAS-23. 630-633.
12) TRW LSI PRODUCTS(1979): TRW Multiplier accumulator parallel 16-bit.
13) 淺野泰之 外 3人(1981): PC-9801 System 解析(上, 下). アスキ-出版國.
14) Digital Singnal Processing Committee IEEE(1979): Programs for Digital Signal Processing. IEEE Press.