UDC 534.87:621.395.38.037.3

Digital Signal Processor

● Toshitaka Tsuda ● Hirohisa Gambe ● Ryusuke Hoshikawa

(Manuscript received July 21, 1989)

This paper reviews Fujitsu's DSP development and gives application examples. In 1988, Fujitsu developed a DSP LSI (MB86232), which has a floating-point multiplier meeting the IEEE standard. This is Fujitsu's 6th DSP LSI and is equipped with several special features. This paper reviews the development history, focusing on the evolution of the architecture, support tools, and process technology. Some DSP application examples are also given.

1. Introduction

The basic theory of digital signal processing (DSP) has been around for a long time. Recently, the technology has been spreading rapidly to new applications. This is due in part to the mature environment which has caused digitization to proceed rapidly and the price of components to drop. Once digital signal processing became feasible, the introduction of the technology has been accerelated and has expanded to several fields. This is because of the advantages of handling analog signals in digital form.

Fujitsu is recognized as one of the leaders in this field, and has developed and marketed a wide variety of digital signal processing components and equipment. The first product developed was the 2 400/4 800 bit/s digital modem in 1977[1]. The DSP (MB8833) was the main component. The announcement of this product as the first Japanese LSI modem had a considerable impact and helped establish a good position in the modem market.

The success of Fujitsu's first DSP (MB8833) promoted the development of other DSP LSIs[2], the latest one being the floating point DSP LSI, called the FDSP-4 (Fujitsu DSP version 4: MB86232).

Each new DSP developed by Fujitsu has established a break-through in LSI processes and DSP architecture.

This paper reviews Fujitsu's DSP LSI development. Section 2 describes the evolution of Fujitsu's DSP in terms of architecture, LSI process and software development tools. Section 3 introduces the features of the latest DSP LSI. Section 4 describes some DSP application in different fields. Section 5 discusses the direction for DSP development and applications in the future.

2. DSP evolution in Fujitsu

2.1 DSP development at Fujitsu

Starting with the development of the first DSP (MB8833[1]) in 1977, Fujitsu developed a series of DSPs, shown in Fig. 1. Table 1 lists the main features of each DSP.

The first two DSPs (MB8833 and MB60502) were for digital modem applications and contained only arithmetic/logic unit (ALU) functions on the chip. The third DSP (MB8760[3]) was for a speech synthesizer and was the first single chip DSP to contain on-chip memory. This was the beginning of a series of single-chip DSP developments. The suceeding DSPs were named FDSP-n (Fujitsu DSP version n).

The first real general-purpose DSP (FDSP-3: MB8764[4)-8]) was developed in 1983, and included many new features. A full parallel pipeline multiplier was used which resulted in a speed of 10 MOPS (Mega Operations Per Second). The chip had powerful logic and data

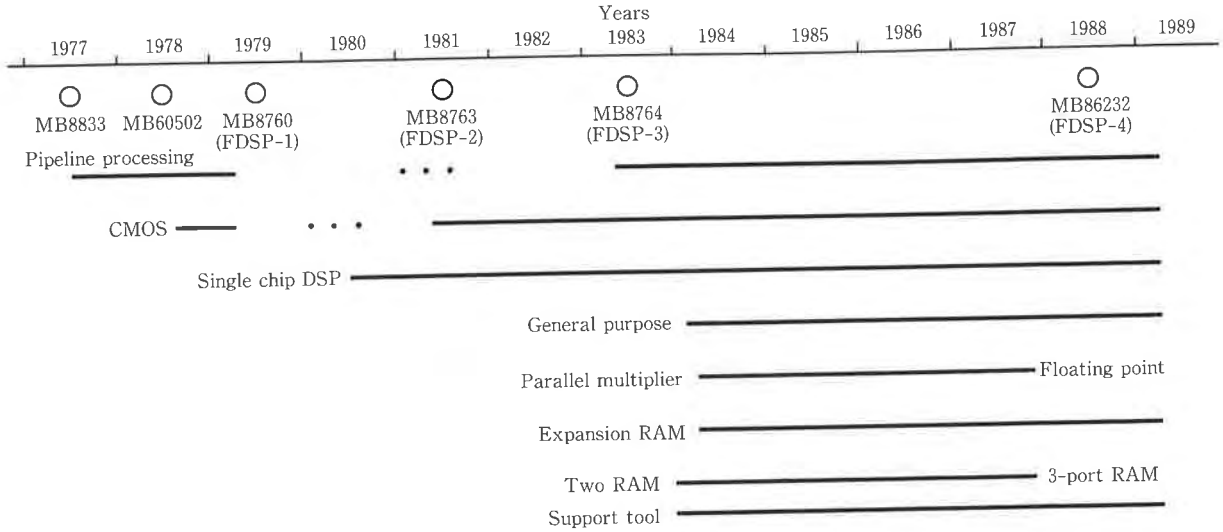


Fig. 1—DSP development at Fujitsu.

Table 1. Main features of Fujitsu's DSPs

Parameter / Type	Multiply and add		Logic operations	On chip memory (kbit)	Process
	Speed (MOPS)	Bit precision			
MB8833	0.144	8 x 12 + 6	–	–	nMOS
MB60502	0.345	10 x 13 + 14	–	–	CMOS
MB8760 (FDSP-1)	0.7	16 x 16 + 16	○	RAM 1 ROM 1	nMOS
MB8763 (FDSP-2)	1.9	14 x 12 + 16	○	RAM 3.5 ROM 16	CMOS
MB8764 (FDSP-3)	10.0	16 x 16 + 26	○	RAM 4 ROM 16	CMOS
MB86232 (FDSP-4)	6.6 13.3	24E8 (Floating) 24 x 24 + 36	○	RAM 16 ROM 32	CMOS

MOPS: Mega Operations Per Second

handling functions.

Previous efforts to improve DSPs focused on faster multiplier operations. Fujitsu focused on both faster multiplier operations and improved data handling capability. In designing the FDSP-3, a considerable part of the effort was spent on improving the data handling capability. This led to the independent dual RAM architecture and the expandable memory capacity by using an external RAM.

Another important point was the software development tool prepared for FDSP-3. This provided users with an easy-to-use program development tool.

The latest DSP (FDSP-4: MB86232) inherited the architecture of the FDSP-3. It has a floating point arithmetic capability which meets the IEEE standard, and the operation speed was increased to 30 percent more than the FDSP-3. This can be considered as the final-stage DSP of the second generation.

2.2 DSP architecture evolution

2.2.1 Characteristic features of DSP

The software and hardware architecture of DSP is similar to that of a microcomputer in that:

1) operation is controlled by software
2) arithmetic/logic unit is included
3) some interface functions to peripherals are

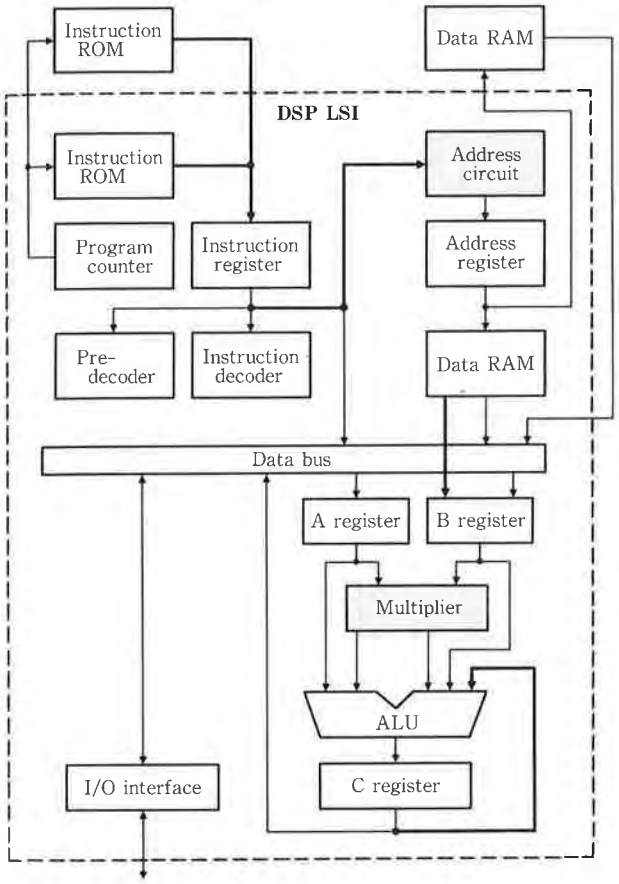


Fig. 2—DSP functional block configuration.

provided.

DSP, however, has some specific architecture which is tailored for efficient and high-speed execution of digital signal processing functions such as multiply and add. Figure 2 shows a typical functional block configuration of an advanced DSP.

In this figure, DSP specific features are shown in thick lines. These are:

1) high-speed hardware multiplier
2) dedicated hardware for memory address calculation
3) dedicated paths for instruction fetch and data transfer.

Article 1 is for the high throughput of multiply and add operations which forms the basis of digital signal processing. Articles 2 and 3 are for concurrent execution of certain functions, and enable pipeline operation for a DSP. Figure 3 shows the difference in operation modes between an advanced DSP and

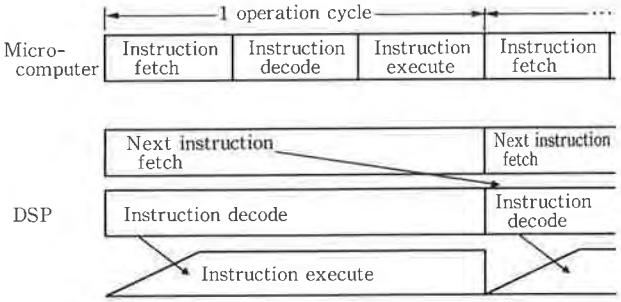


Fig. 3—Program execution of a minicomputer and a DSP.

Load A: RAM (x_1)	$x_1 \rightarrow$ A reg.
Load B: RAM (y_1)	$y_1 \rightarrow$ B reg.
MPL	$x_1 \times y_1 \rightarrow$ C reg.
Load A: RAM (x_2)	$x_2 \rightarrow$ A reg.
Load B: RAM (y_2)	$y_2 \rightarrow$ B reg.
MPL & Add	$x_2 \times y_2 + C \rightarrow$ C reg.
Load A: RAM (x_3)	$x_3 \rightarrow$ A reg.
Load B: RAM (y_3)	$y_3 \rightarrow$ B reg.
MPL & Add	$x_3 \times y_3 + C \rightarrow$ C reg.

a) Microcomputer

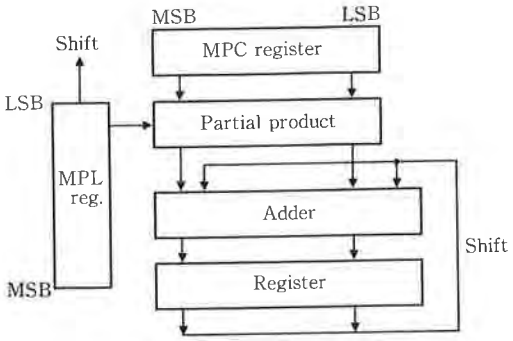
Load A, B: RAM (x_1), RAM (y_1)	$x_1 \rightarrow$ A reg., $y_1 \rightarrow$ B reg.
MPL, Load A, B:	$x_1 \times y_1 \rightarrow$ C reg.,
RAM (x_2), RAM (y_2)	$x_2 \rightarrow$ A reg., $y_2 \rightarrow$ B reg.
MPL & Add, Load A, B:	$x_2 \times y_2 + C \rightarrow$ C reg.,
RAM (x_3), RAM (y_3)	$x_3 \rightarrow$ A reg., $y_3 \rightarrow$ B reg.
MPL & Add, Load A, B:	$x_3 \times y_3 + C \rightarrow$ C reg.,
RAM (x_4), RAM (y_4)	$x_4 \rightarrow$ A reg., $y_4 \rightarrow$ B reg.

b) DSP

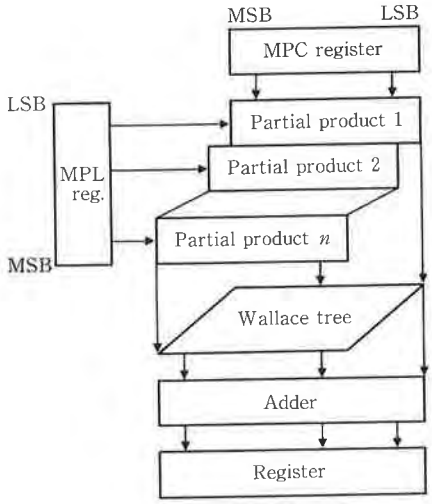
Fig. 4—Typical instruction sequence of a microcomputer and a DSP.

a microcomputer. In a microcomputer, one operation cycle is divided into several phases, and certain resources such as the ALU and system bus are shared. The necessary functions for an instruction execution are carried out serially. However, DSP executes all functions at one time, thus requiring dedicated hardware for each function.

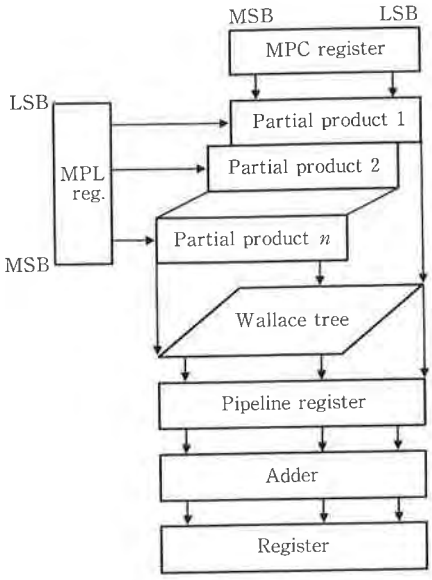
This concurrent execution capability is reflected in the instruction structure. Figure 4 illustrates a typical instruction sequence for a microcomputer and a DSP for the same signal processing. While a microcomputer writes down the operations one by one, the DSP instruction indicates many operations at the same time.



a) Shift and add type



b) Parallel type



c) Parallel pipeline type

MPC: Multiplicand
MPL: Multiplier
LSB : Least significant bit
MSB: Most significant bit

Fig. 5—Multiplier structures.

This instruction structure is called a horizontal structure. This suits pipeline operation of a DSP quite well.

The next section describes the architecture evolution of Fujitsu's DSP in terms of the high-speed multiplier, data/instruction memory and addressing, and instruction execution.

2.2.2 High-speed multiplier

Figure 5 shows the typical configuration of a multiplier.

Type a is called a "shift and add" type and performs a multiply operation by making a partial product of a multiplicand on a bit-by-bit basis of a multiplier and adding it to a right-shifted partial sum, making a new partial sum as the result. This type requires fewer logic gates but the multiplication capability is low since it takes n clock cycles to obtain the final result, where n is the number of bits of the multiplier.

Compared to this, parallel multiplier (type b) and parallel pipeline multiplier (type c) calculate all the partial products and add them by providing the necessary number of partial product circuits. Some circuits are required between the partial product stage and the final adder stage to reduce the number of the equally weighted signals to two. This allows the adder to carry out the final addition. The Wallace

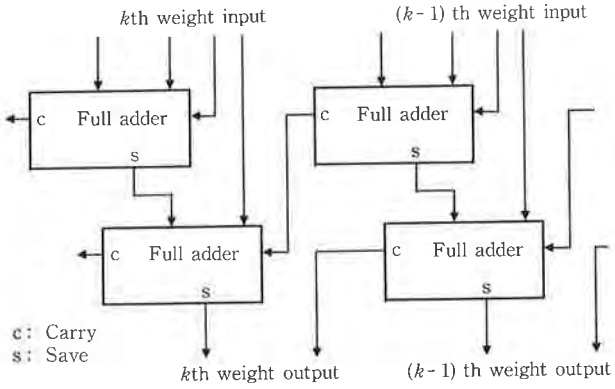


Fig. 6—Wallace tree structure for four equal weight inputs.

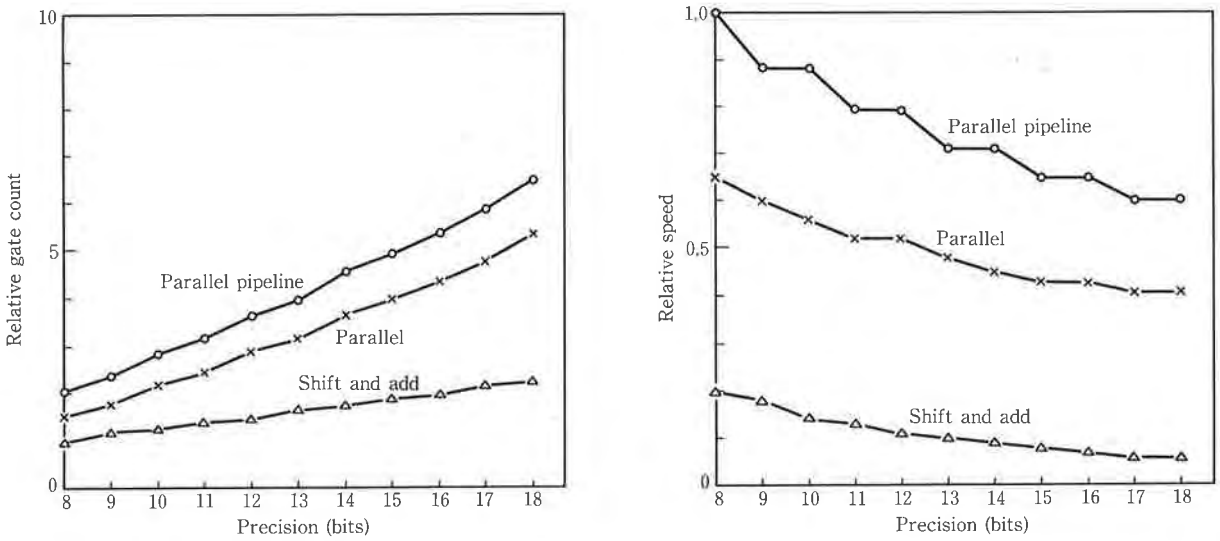


Fig. 7—Relative performance and complexity of each type of multiplier.

tree is a widely used circuit to provide such a function. It has a multi-stage structure whose number depends on n. Figure 6 shows the function of the Wallace tree when n is equal to 4. Parallel architecture leads to a superior multiplication capability, but more logic gates are needed. The differences between the pipeline and non-pipeline is that in a pipeline structure the partial sum is latched at a certain stage of calculation before obtaining the final result. By this, the critical path of signal propagation is cut short, and higher performance is possible if consecutive multiplication is performed. Figure 7 shows a rough estimation of the relative performance and required gate counts for each type of multiplier. Here, the second order Booth's algorithm for partial product, Wallace tree for partial sum, and the 4-bit full adder with carry look ahead is assumed. Note that this estimation excludes functions such as accumulation, selector functions, and logic functions.

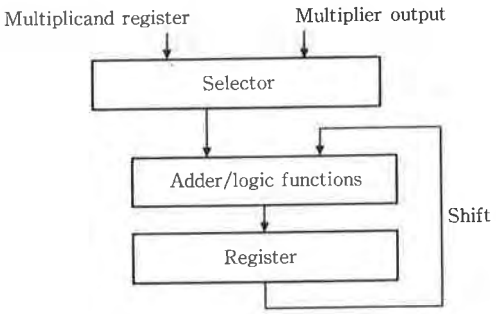
Table 2 shows the multiplier architecture adopted for each DSP. The choice of architecture was mainly influenced by the possible integration scale of the time. In the first DSP (MB8833), a bit-by-bit shift and add architecture was used. From the second DSP (MB60502) to the fourth DSP (FDSP-2), a certain degree of parallelism was introduced, thus reducing

Table 2. Multiplier in each DSP

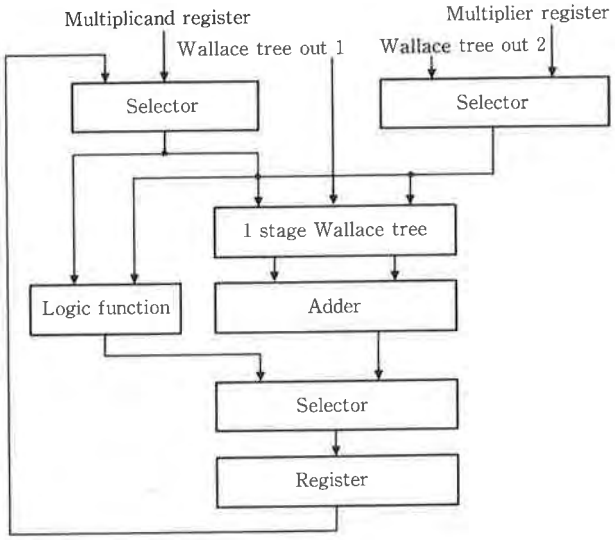
Type \ Item	Multiplier type
MB8833	Serial (12-bit x 1-bit) x 8 cycles
MB60502	Serial (13-bit x 2-bit) x 5 cycles
MB8760 (FDSP-1)	Serial (16-bit x 2-bit) x 8 cycles
MB8763 (FDSP-2)	Serial (14-bit x 4-bit) x 3 cycles
MB8764 (FDSP-3)	Parallel pipeline (16-bit x 16-bit)
MB86232 (FDSP-4)	Parallel pipeline (Floating) (24-bit x 24-bit)

the required number of clock cycles. In the FDSP-3, a full parallel pipeline architecture was adopted. The FDSP-4 expanded the FDSP-3 architecture and added the capability of the IEEE standard compatible floating point arithmetic.

The ALU of DSP is designed to have a multiply and add function as its core. To expand the versatility, it also has an add function and some logic functions such as "shift", "inverse", "and", and "or". Figure 8 shows the ALU configuration of the shift and add types (FDSP-1, -2) and parallel pipeline type multiplier for FDSP-4. The ALU for the shift



a) ALU for shift and add multiplier



b) ALU for parallel pipeline multiplier

Fig. 8—ALU structures.

and add type constitute of a multiplier followed by an integrated add and logic unit. Compared to this, that of a parallel pipeline multiplier has an adder as a part of a second stage multiplier. To enable a multiply and add operation, a one-stage Wallace tree is inserted which converts three equally weighted input signals into two output signals. The logic function block is separated from the adder function. This is due to shortening the critical path of the second stage multiplier, thus improving the operation capability.

2.2.3 Data/instruction memory and addressing

Table 3 summarizes the features of each DSP in terms of data/instruction memory structure and the addressing capability. One point to be noted is the location of the memory. It started with external memory due to the lack of integration at that time. They were integrated on the chip for FDSP-1 and afterwards, gradually expanding the capacity. The external instruction ROM was made available most of the time because field programmability is essential for the DSP to be used widely. The external RAM

Table 3. Memory and addressing of each DSP

Parameter / Type	Instruction memory			Data memory		Data interface	Data addressing	Inner bus
	Internal	External	Interface	Internal	External			
MB8833	—	ROM 2-bit × 512	2-bit serial	—	RAM 2-bit × 512	2-bit serial	1 address Direct	—
MB60502	—	ROM 2-bit × 1 024	2-bit serial	—	RAM 2-bit × 1 024	2-bit serial	1 address Direct	—
MB8760 (FDSP-1)	ROM 16-bit × 256	—	—	RAM 16-bit × 64	—	8-bit parallel	1 address Direct, Index modify	16-bit × 1
MB8763 (FDSP-2)	ROM 16-bit × 1 024	ROM 16-bit × 2 048	16-bit parallel	RAM 14-bit × 256	—	8-bit parallel	1 address Direct, Index modity DMA write	14-bit × 1
MB8764 (FDSP-3)	ROM 24-bit × 1 024	ROM 24-bit × 1 024	24-bit parallel	RAM 16-bit × 128 ×2	RAM 16-bit × 1 024	16-bit parallel	2 addresses Direct, Index modify DMA write/read Vértual shift	16-bit × 1
MB86232 (FDSP-4)	ROM 32-bit × 1 024	ROM 32-bit × 64-kbit	32-bit parallel	RAM 32-bit × 512 3-port	RAM 32-bit × 64-kbit × 16	16-bit or 32-bit parallel 1-bit serial	3 addresses Direct, Index modify DMA write Vertual shift	32-bit × 2

Table 4. Pipeline operation of each DSP

Parameter / Type	Concurrency of operation	Pipeline operation		
		Arithmetic operation	Instruction decode	Address calculation
MB8833	ALU operation + 2 data transfer	2 stages	Non	Non
MB60502	ALU operation + 2 data transfer	2 stages	Non	Non
MB8760 (FDSP-1)	Non	Non	Non	Non
MB8763 (FDSP-2)	Partly PLU operation + 1 data transfer	Non	2 stages	2 stages
MB8764 (FDSP-3)	ALU operation + 2 data transfer	2 stages	2 stages	2 stages
MB86232 (FDSP-4)	ALU operation + 3 data transfer	2 stages	2 stages	2 stages

disappeared for FDSP-1 and FDSP-2 when on-chip integration became possible. Then the arithmetic operation capability was increased by a parallel pipeline multiplier and a limitation of memory capacity rather than arithmetic operation capability was foreseen in many applications. This led to the introduction of an architecture which allowed memory to be expanded by an external RAM in the design of the FDSP-3.

Another point to be noted is the data RAM structure. In achieving a pipeline operation, it is necessary to transfer two data items at a time to be processed in the next operation cycle. For the FDSP-3, an independent two-data RAM structure was proposed to solve this problem. A three-port RAM structure was adopted for the FDSP-4, enabling concurrent two-data-item read and one-data-item write. Since each port can access the entire memory area, the memory location management required in FDSP-3 software design was reduced. This architecture together with a memory capacity expandable structure is commonly used in advanced DSPs.

Only simple direct addressing was available for early DSPs. An index modification was adopted for the FDSP-1, and DMA write was introduced for the FDSP-2, providing an easy data input mechanism. Virtual shift addressing, which simulates a shift register operation without moving the data itself, was adopted for the FDSP-3. In digital signal procesing, a shift register operation is a basic function and is frequently used in transversal filters. Virtual shift addressing achieves this function efficiently, and is used in some advanced DSPs.

2.2.4 Instruction execution

Table 4 shows the instruction execution method of each DSP. It is interesting to note that the concurrent execution of data transfer and ALU operation was adopted from the beginning, but not for FDSP-1 and FDSP-2. This was because of the productivity of application programs. In a pipeline operation, the programmer must keep in mind a precise data flow timing diagram, which requires much effort. Concurrent execution, however, is essential to obtain a high throughput, and was again adopted for the FDSP-3 and FDSP-4. This was possible because of the development of support tools for programming which kept program productivity high even for the pipeline operation. Section 2.4 describes this in more detail.

Another important point is the pipeline operation of instruction decoding and address calculation we proposed. To shorten the critical path of signal propagation, and thus increase the operation speed, pipelining using pre-fetch

and precalculation is very effective. This architecture is commonly used these days. Figure 9 shows the instruction execution mode of the FDSP-4.

2.3 LSI process evolution

DSPs were fabricated using the latest LSI technology.

Table 5 summarizes the process technology used for each DSP. Both nMOS and CMOS were used, but COMOS was mostly used for later DSPs. The design rule became finer and the available chip size became larger. Accordingly, the gate count kept increasing until it was 40 times larger than the first DSP. This improvement in integration scale was reflected by the DSP architecture, achieving a parallel pipeline multiplier with floating point arithmetic and dual inner bus structure.

The increase of the pin count must also be noted. Owing to this increase, parallel I/O interfaces became available and led to the adoption of the expanded external memory architecture.

The sophisticated wiring layer structure used for the FDSP-4 enable high integration density and reduced the propagation delay time.

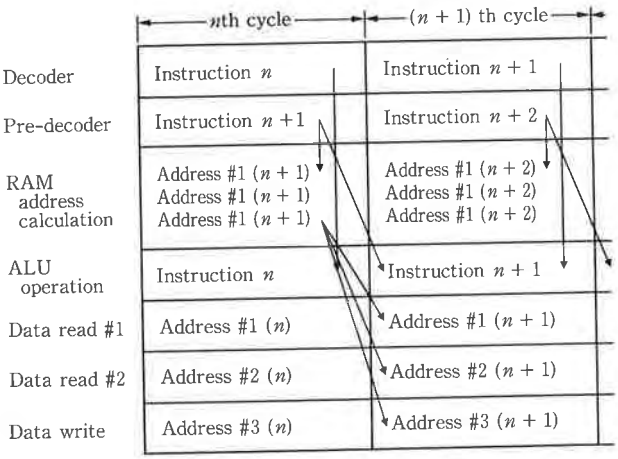


Fig. 9—Instruction execution of FDSP-4.

2.4 Support tools[9)]

When a programmable device is introduced, we need to have efficient firmware development for the device.

This is also the case of DSPs.

From the time development of the FDSP-1 first started, the most efficient firmware development tools were always considered.

The following aspects for support tools must be considered:

1) To provide efficient programming languages for a processor
2) To provide a simulation program by which program execution can be traced
3) To provide a hardware emulator by which a DSP program can be run on a real-time basis.

Figure 10 shows the structure of DSP support tools.

Table 5. Process technology of each DSP

Parameter / Type	Process	Design rule	Gate count	Chip size	Pin count	Wiring	Power consump.
MB8833	nMOS	6.0 μm	1 500	3.4 x 4.2 mm	40	Single metal	1 W
MB60502	CMOS	3.6 μm	1 900	7.2 x 7.2 mm	64	Double metal	500 mW
MB8760 (FDSP-1)	nMOS	4.0 μm	8 000	5.9 x 6.0 mm	28	Single metal	1 W
MB8763 (FDSP-2)	CMOS	3.0 μm	13 000	8.2 x 7.3 mm	64	Double metal	150 mW
MB8764 (FDSP-3)	CMOS	2.3 μm	23 000	9.5 x 9.6 mm	88	Double metal	300 mW
MB86232 (FDSP-4)	CMOS	1.2 μm	65 000	13.5 x 13.5 mm	208	Triple metal	1 W

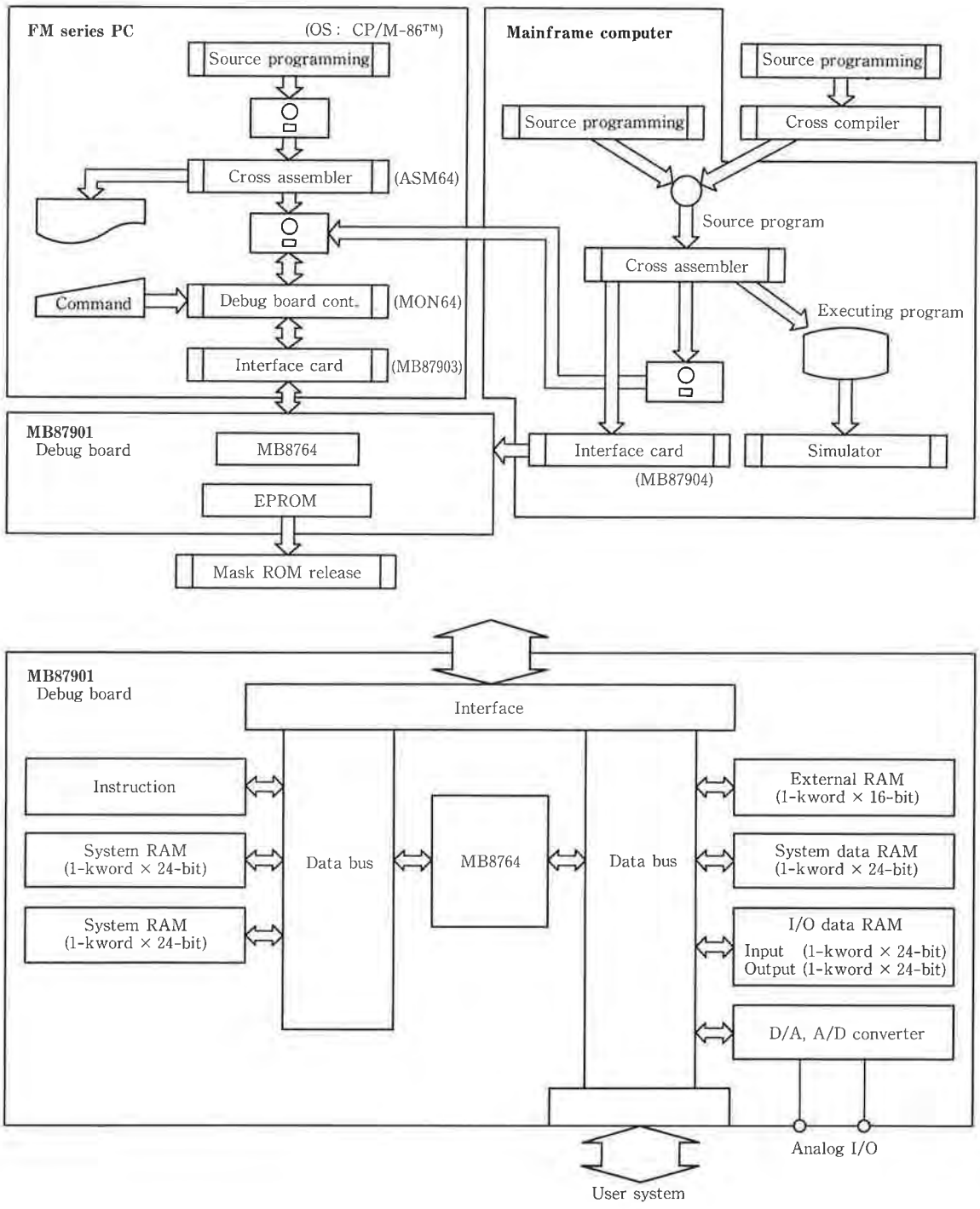


Fig. 10—DSP support tool structure.

2.4.1 Programming languages

A DSP usually has its own instruction set. The instruction set is usually similar to that of a microprocessor. As DSP instructions must support parallel execution and a pipeline structure, the set may become quite complex. This assembly language type instruction set is directly related to the instruction code. For ease of programming, every effort should be made to implement an efficient instruction set. The DSP must be able to execute signal processing in real time. In the DSPs it is quite important to make program steps as short as possible. A programmer may require a long time to

write efficient programs using this assembler level instruction set. To make a DSP more popular as a general-purpose processor, more convenient programming functions may be needed. For such demand, the authors studied high-level language support for some DSPs. For the FDSP-2 and FDSP-3, the authors developed a PASCAL compiler. The most important issue of the compiler was how efficient the object programs were produced. Fujitsu usually measures its efficiency by comparing its object programs with assembled programs written in assembler by a DSP programming expert. Fujitsu marked a step size efficiency of 1.34 and a memory size efficiency of 1.21 for an FDSP-3 test program.

Future DSP systems may consist of many DSP chips to construct a parallel processing system. In these systems, a high-level language which supports parallel processing will be a future subject for DSP compilers.

2.4.2 Simulator

To test the performance of newly developed programs, an off-line base simulator is helpful. It is required to carry out the same computations as the real chip. With such a simulator, any value in any register or memory location at any program step can be monitored. For the FDSP-3, a simulator which runs under CP/M is provided. Fujitsu also provided a simulator for the FDSP-4 which runs under MS-DOS.

2.4.3 Real-time emulator

In the final stage of DSP program development, a real-time base test is important. For this purpose, Fujitsu provides a real-time emulator. With this, all the programs can be tested on a real-time basis. If needed, intermediate results of any register or memory location can be monitored as is done in an off-line simulator.

The FDSP-3 emulator is controlled by an FM series personal computer. We also have an emulator for the FDSP-4. Its DSP ASICs are discussed later.

The FDSP-4 emulator has an in-circuit emulator connector. The DSP board may thus be tested on a real-time basis.

By connecting emulators, a multi-DSP system may also be evaluated.

3. FDSP-4 (MB86232[10),11)])

Fujitsu's most advanced general-purpose DSP FDSP-4 is fabricated by 1.2-μm CMOS technology. It has a 32-bit floating point multiplier and has a 75-ns machine cycle. The multiplier performs the IEEE 24E8 'normal number' multiplication specified by the IEEE Draft 10 once every two machine cycles. It also handles 24-by-24-bit integer multiplication with a 48-bit accumulator and 24-by-24-bit fixed point multiplication with a 36-bit accumulator.

Fujitsu developed this processor to cover most speech applications. At the same time, Fujitsu took care to make its architecture as general as possible so that future version chips would be able to run using the same software resources and development tools. As is discussed later, the FDSP-4 is also used as a base processor for future ASIC DSPs.

3.1 Architecture

The block diagram of the FDSP-4 is shown in Fig. 11.

3.1.1 Memory space architecture

The Harvard architecture is adopted to achieve efficient program execution. RAM and ROM are located at different memory spaces. This allows high speed operation and also an efficient pipeline architecture.

3.1.2 Data formats

The three types of data format shown in Fig. 12 are supported. It is basically a kind of 32-bit processor.

However, 24-bit fixed-point computation can also be done. The device may be used both for quick algorithm checking with floating-point arithmetic functions and for cost efficient computation with fixed-point arithmetic functions.

3.1.3 Bus architecture

There are two pairs of 32-bit main data buses and each register/memory is connected to these two buses.

Therefore, two independent data transfer operations can be executed simultaneously. A sub-bus between RAM and register A is also

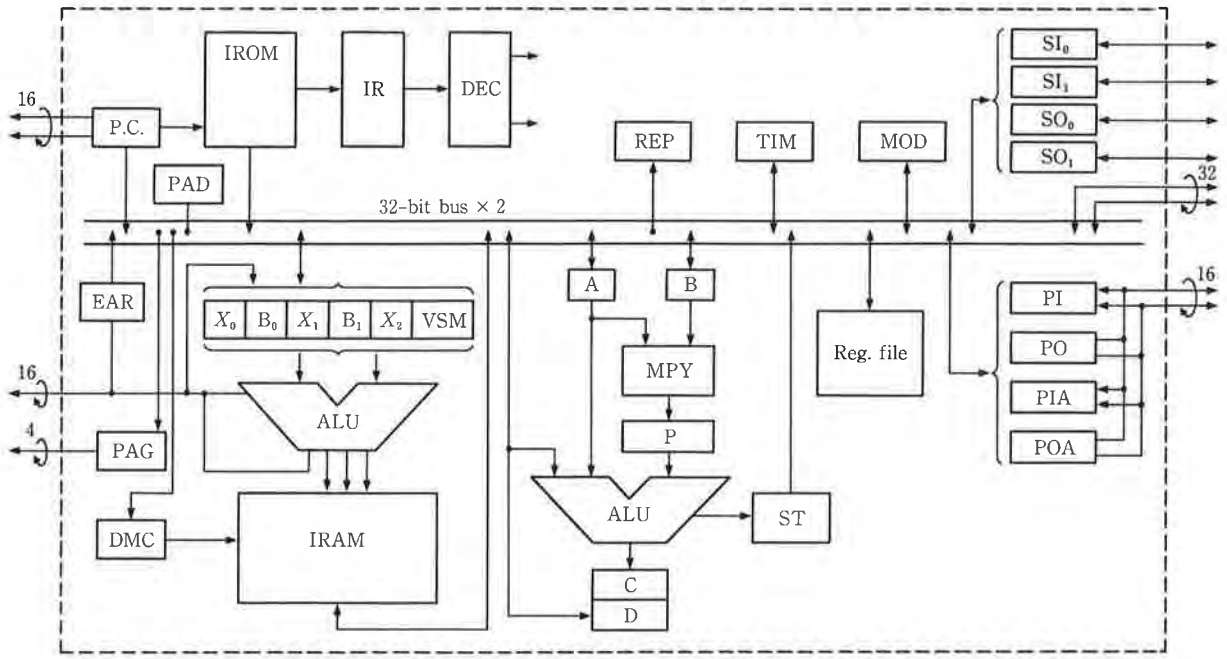


Fig. 11—Block diagram of the FDSP-4.

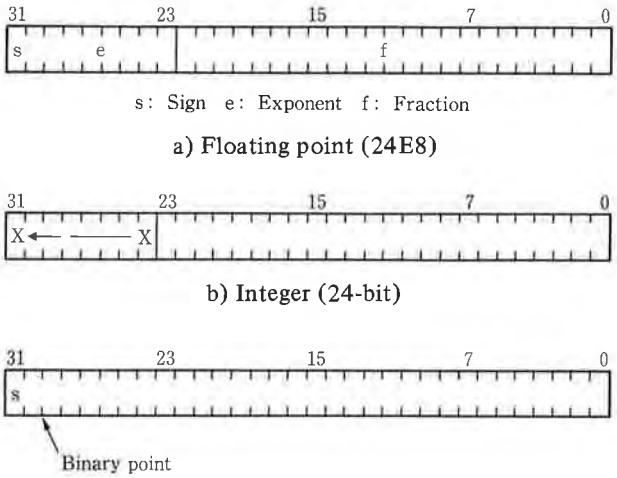


Fig. 12—Data format.

supplied to allow triple data transfers between the three-port RAM and the registers. This flexible data transfer function eliminates bottlenecks from data transfer.

3.1.4 ALU operation

The multiplier and adder are in the pipeline structure. The computation of an inner product can thus be carried out once per two machine cycles for 32-bit floating point computation. The same computation can be done once every machine cycle with the fixed-point arithmetic functions.

3.1.5 Memory

The chip contains 512-word by 32-bit three-port RAM. Two read operations and one write operation can be done by a single instruction. This means that three simultaneous data transfers can be executed. The chip also contains a two-port register file of 16-word by 32-bit RAM. RAM can be extended off-chip if a larger memory is needed.

3.1.6 Data input/output

Dual serial input and output ports are provided. These ports may be used to build a multi-DSP system. A 32-bit parallel I/O port for external RAM and a 16-bit general purpose parallel I/O port are also provided. In the slave mode, the chip acts as a RAM and data read/write can be done through the 32-bit parallel I/O port.

3.1.7 Interrupt

For single-bit interrupt terminals, two flags and a timer interrupt function are provided. These interrupt functions may be useful for microprocessor-oriented applications.

All interrupts are maskable by a mode register.

There are some other interrupt functions for program debugging.

3.2 Software structure

The FDSP-4 has a highly functional instruction set. Although it is mainly a digital signal processor, it may be used as a high-speed microprocessor. The instruction set was carefully designed to make it the standard set for future versions so that high-level language compilers could be developed for it.

3.2.1 Instruction forms and codes

The instruction code consists of 32 bits. All instructions are single word. In most of the instructions, six bits are reserved for multiplier and/or ALU instructions. Therefore, any arithmetic operation can be arbitrarily combined with non-arithmetic operations, although there are some instructions which cannot include arithmetic computations.

Each instruction is expressed by a combination of arithmetic and non-arithmetic mnemonics with some operands such as

ADDC: LAB $5, $6.

This instruction specifies the following operations:

1) Add data in registers A and B and place the results in register C
2) Load register A with data from memory location 5

Table 6. ALU operations of the FDSP-4

Code	ALU operation		Cycle	F	I
00	NOP	No operation	1	○	○
01	ADD	a + A → a	1	○	○
02	SUB	a − A → a	1	○	○
03	ADX	a + A + ca → a	1		○
04	SBX	a − A − ca → a	1		○
05	MLS	A × B → P (with sign)	1	○	○
06	MLU	A × B → P (without sign)	1		○
07	MSM	a + PL → a, A × B → P	1	○	○
08	MSP	PL → a, A × B → P	1	○	○
09	SMP	a + PL → a	1	○	○
0A	MRD	a + PL → a, A × B → P	1	○	○
0B	AND	a ∩ A → a	1	○	○
0C	ORA	a ∪ A → a	1	○	○
0D	EOR	a ⊕ A → a	1	○	○
0E	NOT	$\overline{a}$ → a	1	○	○
0F	ABS	|a| → a	1	○	○

Code	ALU operation		Cycle	F	I
10	ADL	a + PL → a	1		○
11	ALC	a + PL + ca → a	1		○
12	ADH	a + PH → a	1		○
13	AHC	a + PH + ca → a	1		○
14	NEG	−a → a	1	○	○
15	NEX	−a − ca → a	1		○
16	CMP	a − A	1	○	○
17	FCP	a − A	2	○	○
18	FAD	a + A → a	2	○	○
19	FSB	a − A → a	2	○	○
1A	FML	A × B → P	2	○	○
1B	FMS	a + P → a, A × B → P	2	○	○
1C	FMR	a − P → a, A × B → P	2	○	○
1D	FAM	|a| → a	2	○	⊙
1E	FSM	P + a → a	2	○	○
1F	FSP	P → a, A × B → P	2	○	○

Code	ALU operation		Cycle	F	I
20	LSR	logic SR	1		○
21	ASR	arith SR	2	○	○
22	LSL	logic SL	1		○
23	ASL	arith SL	1	○	○
24	ROR	rotate R	1		○
25	ROL	rotate L	1		○
26	CIF	int → float	2		○
27	CFI	float → int	2		○
28	CXF	fix → float	3	○	
29	CFX	float → fix	3	○	
2A	FDV	a/A → a	27	○	○
2B	FNE	−a → a	2	○	○
2C					
2D					
2E					
2F					

Code	ALU operation		Cycle	F	I
30	REV	bit reverse	1	○	○
31	RND	round off	1	○	
32	INC	a + 1 → a	1	○	○
33	DEC	a − 1 → a	1	○	○
34	TRC	C → a	1	○	○
35	TRD	D → a	1	○	○
36	TRA	A → a	1	○	○
37	TNA	−A → a	1	○	○
38	TPH	PH → a	1		○
39	TPL	PL → a	1		○
3A	ACZ	a + ca → a	1		○
3B	TRP	P → a	1	○	
3C	TST	a A	1	○	○
3D					
3E					
3F					

F: Fixed point mode I: Integer mode

3) Load register B with data from memory location 6.

Many instructions require only a single machine cycle. Some take more than two machine cycles.

3.2.2 Arithmetic operations

Six bits are used to define arithmetic operations. One of these bits designates the destination accumulator, A or B. The other five bits define the type of operation. The arithmetic operations are listed in Table 6. The sets are slightly different between the integer computation mode and the fixed-point computation mode.

3.2.3 Non-arithmetic instructions

Due to the two pairs of 32-bit data buses, the hardware architecture allow two items of data to be transferred simultaneously. Therefore, many types of double transfer instructions are provided. Some instructions even allow triple data transfer.

3.2.4 Other instructions

Conditional branches and branches to subroutines are available.

These conditions can be used for other purposes such as:

: LDIF LTC, $8, B,

which means

IF C < 0 THEN MOV data from address 8 of RAM to B ELSE no operation

There are some special operations such as:

RPT #8

which means repeat the next operation eight times.

4. DSP application examples

This section describes some application examples of DSP collected from different fields.

4.1 Digital modem

The digital modem was the first successful application of DSPs. The first modem had a speed of 4800 bit/s[1]. Current DSP modems have a speed of 19 200 bit/s. The core function is the automatic line equalizer to compensate for the transmission line loss and group delay characteristics. Figure 13 shows the block diagram of the automatic line equalizer.

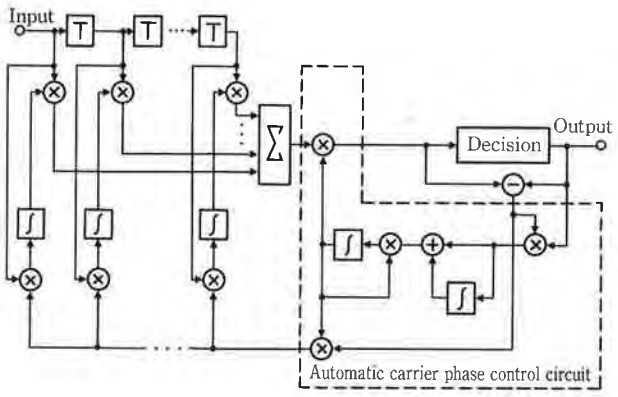


Fig. 13—Block diagram of automatic equalizer.

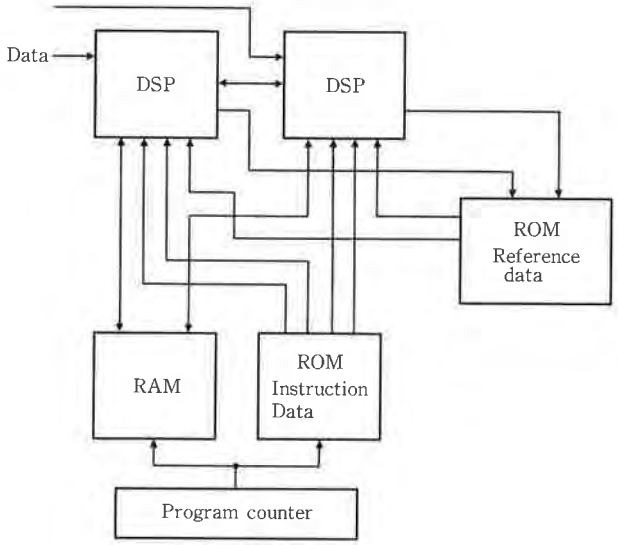


Fig. 14—Block diagram of first digital modem.

To see the arithmetic operation, Equations (1) and (2) describe the transverse filter and its tap coefficient adaptation operations.

$$y = \Sigma a_i \times x_i , \quad \ldots\ldots\ldots\ldots (1)$$

$$a_i(t) = a_i(t-1) + \alpha \times x_i \times e , \quad \ldots (2)$$

where a_i: tap coefficient,
x_i: received signal,
y: output,
e: equalization error,
α: decay factor.

As can be seen, these calculations can be done by repeatedly executing multiply and add operations ($a \times b + c = c$). This can be done efficiently by a DSP. Figure 14 shows the

Fig. 15—Fujitsu's first LSI modem ($450^{l} \times 210^{b} \times 172^{h}$: mm).

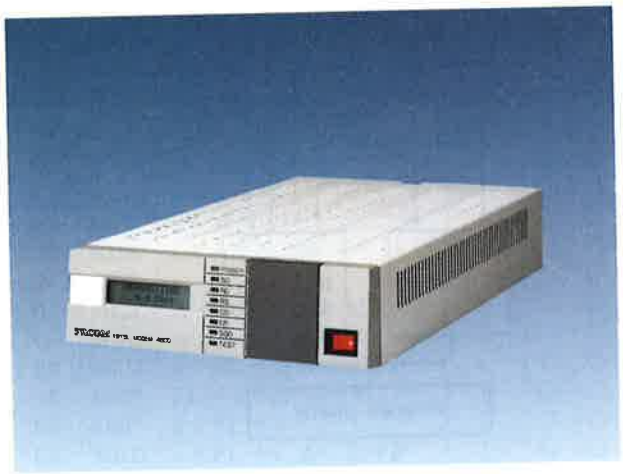
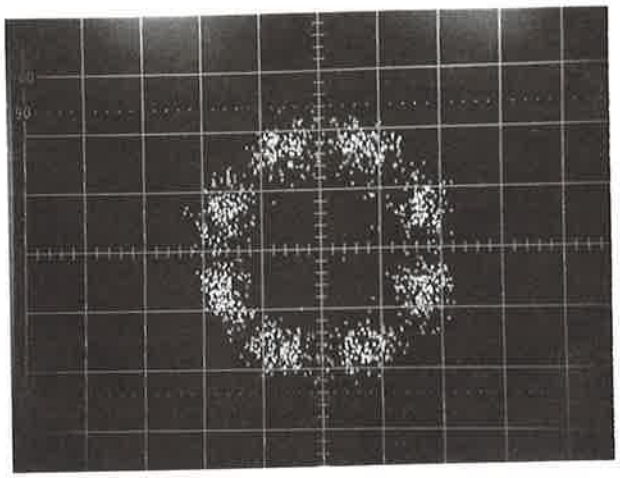
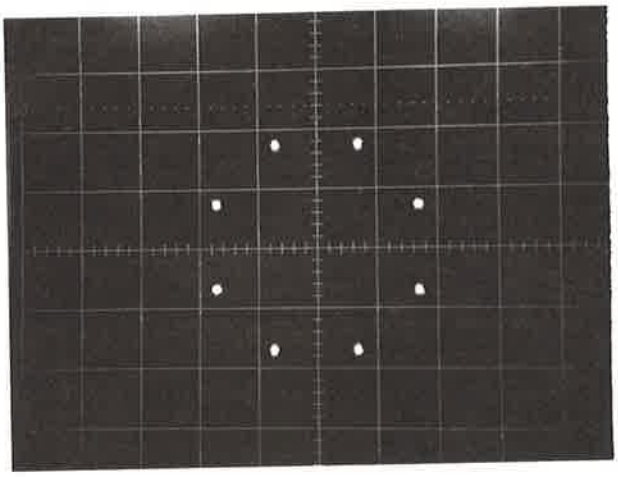


Fig. 16—New version of LSI modem ($350^{l} \times 210^{b} \times 70^{h}$: mm).



a) Before line equalization



b) After line equalization

Fig. 17—Eye openings.

block diagram of the automatic line equalizer for the first 4 800 bit/s modem. Two DSP chips (MB8833) and the ROM and RAM made up the digital signal processing part of the modem. Figures 15 and 16 show the first modem and a recent one based on the FDSP-2[12]. With the progress is DSP, the size of the PC board has been reduced in spite of the increased complexity. Figure 17 shows the eye diagram of signals before and after line equalization.

4.2 Voice synthesizer[3]

The FDSP-1 was dedicated to voice synthesis which outputs voice signals created by an extremely reduced number of vocal tract parameters. For this purpose, the FDSP-1 is equipped with a 10-bit precision D/A converter for analog output signals. Figure 18 shows the block diagram of the voice synthesis system. The FDSP-1 is programmed to perform the voice synthesis based on the PARCOR (PARtial CORelation) algorithm. Parameters are stored in ROM in variable-length data format. Data is read into the microcomputer and converted to 16-bit fixed length format, then transferred to the FDSP-2 for synthesis. Functions such as timing signal generation, addressing of data ROM and bit-by-bit operations are all performed by the microcomputer. This can be considered a good example of role partition between a general-purpose microcomputer and a DSP which is tailored to a high-speed arithmetic operation.

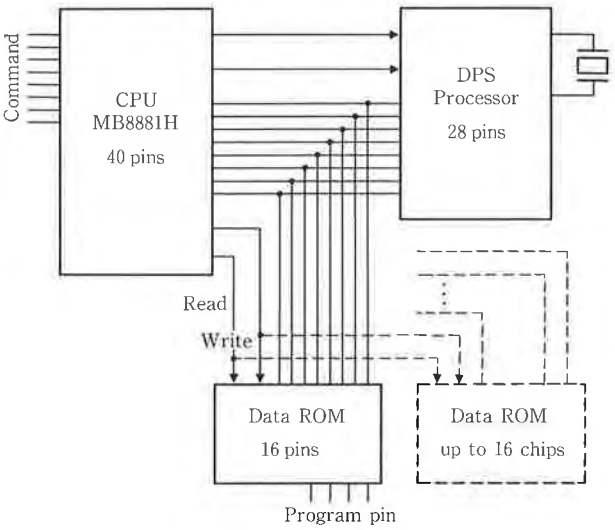


Fig. 18—Voice synthesizer.

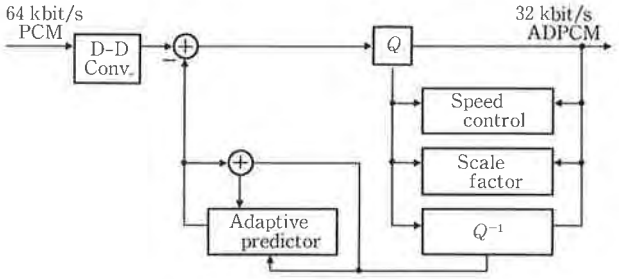


Fig. 19—32 kbit/s ADPCM voice codec.

4.3 Speech codec

Speech codecs, especially those for low bit rate voice encoding/decoding, are the most popular of the DSP applications. In present digital voice communication systems, 64-kbit/s PCM coding specified by the CCITT Recommendation G.711 has been widely used. However, there is much demand for a more efficient voice communication method which can be applied to the restricted capacity of digital communication lines.

There are two points in these demands:

1) To reduce the transmission bits by keeping almost the same quality of PCM.
2) To improve the quality or error-free capability by keeping the same transmission bit capacity compared with normal PCM coding.

4.3.1 32-kbit/s ADPCM (Adaptive Differential PCM[13)-16)])

This codec can provide almost the same quality as a PCM with just half the information rate. The algorithm of this codec has been standardized as specified by CCITT recommendation G.721.

Figure 19 shows the block diagram of ADPCM. In the encoder, an input signal is first compared with the predicted value provided from a prediction circuit. After the difference is calculated, its quantized value is sent out, while in the decoder the received signal is added with the predicted value provided from a prediction circuit, which acts in the same way as that of the encoder. Both in the encoder and decoder, the quantized values are reconstructed and fed into the prediction filter to produce the next prediction value. As this kind of differential coding is adopted, the encoder and decoder must employ exactly the same computation. Even a small mismatch of data bit length may cause a large degradation. The codec must therefore trace every detail of the computation algorithm specified by G.721. We developed this codec using the FDSP-3. A multi-channel codec (up to 32 channels) by ASIC is also available. This CCITT codec provides a voice transmission quality equivalent the 7-bit PCM. However, 9 600-bit/s modem signal which is widely used for facsimile transmission service cannot be sent successfully using the CCITT methods. Fujitsu provided additional capability to support facsimile service with additional FDSP-3 chips.

4.3.2 8- and 16-kbit/s codec[17),18)]

There are codecs for 8- or 16-kbit/s transmission rates with small degradation compared to PCM. There are many algorithms which have been developed for such requirements.

The CCITT standardization is still under study. Among the proposed methods, APC-AB (Adaptive Predictive Coding with Adaptive Bit Allocation) is the most popular. The basic configuration is shown in Fig. 20. This method combines sub-band coding and adaptive predictive coding. Input signals are divided into three bands and predictive coding is carried out independently in each band. The FDSP-4 base ASIC DSP was developed for this codec and

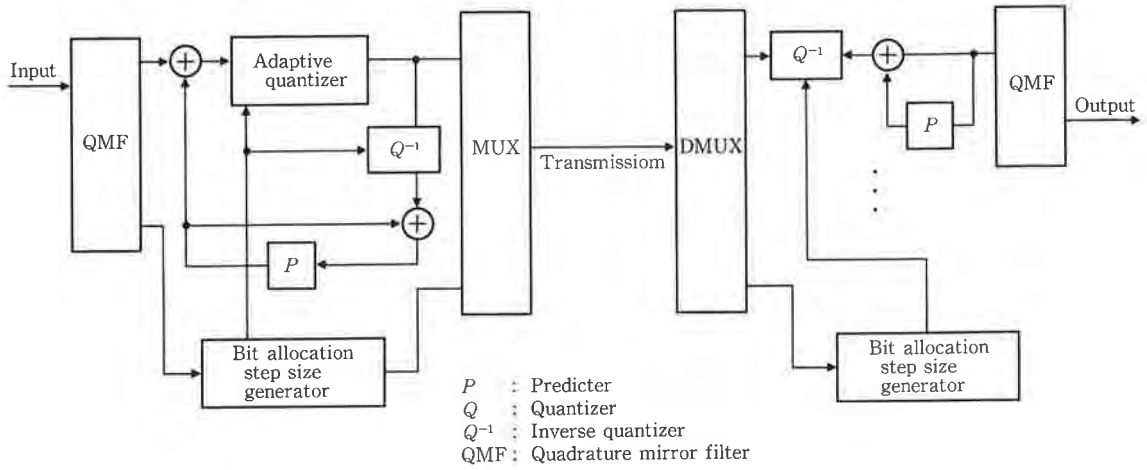


Fig. 20—APC-AB voice codec.

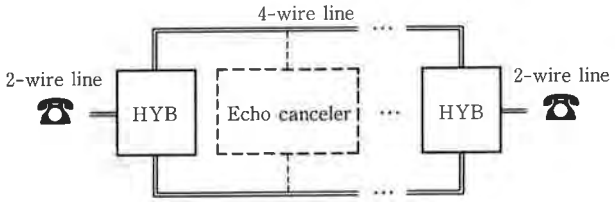


Fig. 21—Basic idea of an echo canceler.

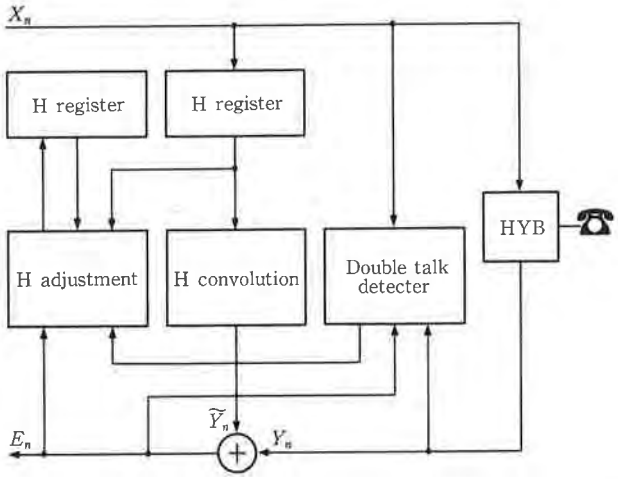


Fig. 22—Echo canceler configuration.

will be described later.

4.3.3 64 kbit/s ADPCM[19)]

This codec aims at high AM broadcast quality voice signal transmission with the same transmission rate of the PCM. The input signal is sampled at 16 kHz and divided into two subbands. ADPCM is applied in each band. This method has also been standardized as CCITT G.722. The FDSP-3 is used for this codec.

4.3.4 Burst-mode codec

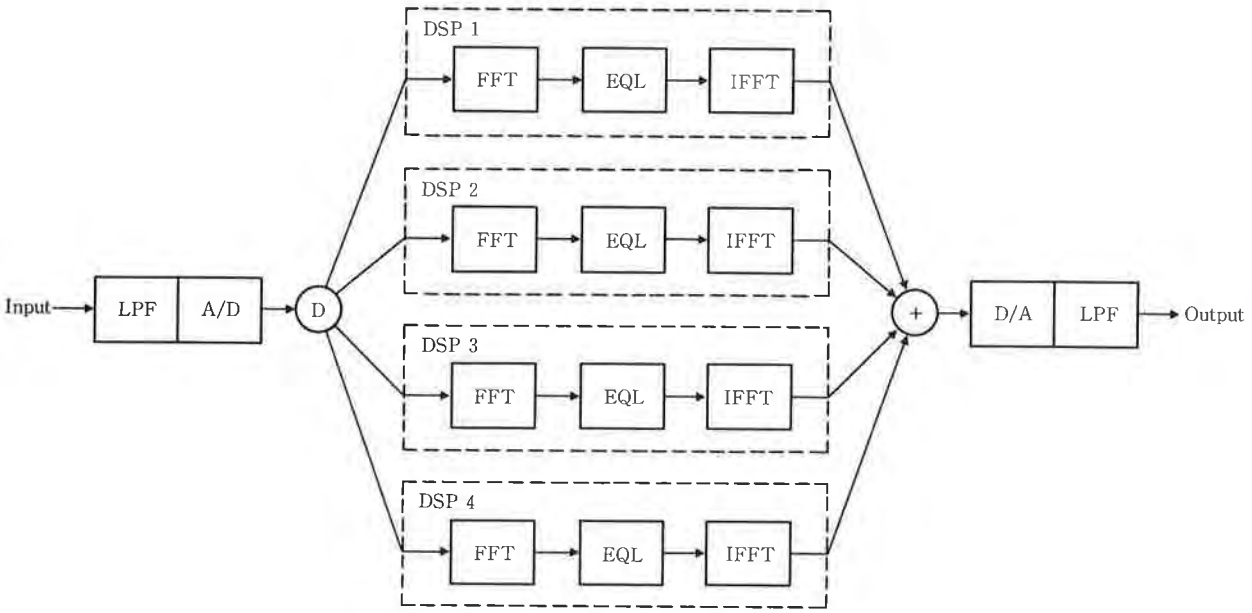
Speech codecs for ATM transmission systems are also being studied. The output of this encoder may be sent to a packet transmission network. The codec must thus be able to handle the possibility of losing a signal bit due to the loss of packet cells. The modified algorithm of ADPCM is now under study.

4.4 Echo canceler[20),21)]

Another popular application of using DSPs is the echo canceler. The basic idea of an echo canceler is shown in Fig. 21. In this network, voice signals sent out from the telephone to a wire transmission line may be looped back to the speaker side through a four-wire-to-two-wire conversion transformer at the other side of the transmission network. If the four-wire line becomes too long, the distance of the original signal and returned leak signal becomes large and the speaker may feel irritated to hear this leak signal as the echo. If one hop of the satellite transmission path is inserted in the transmission path, the echo delay may become as large as 300 ms.

Figure 22 shows the basic configuration of the echo canceler. The echo signal is canceled by subtracting the predicted echo from the transmission signal.

Echo Y_n at sampling time n and the output $\widetilde{Y}_n$ provided from echo canceler block are



FFT: Fast fourier transform IFFT: Inverse FFT LPF: Low-pass filter

Fig. 23—Delay-time equalizer for MF radio transmission.

written as:

$$Y_n = \Sigma_i (W_i X_{n-1}) \quad \tilde{Y}_n = \Sigma_i (H_i X_{n-i}).$$

Then the error E_n of echo canceler is written as

$$E_n = Y_n - \tilde{Y}_n$$
$$= \Sigma_i \{ (W_i - H_i) X_{n-i} \},$$

and H_i is computed as

$$H_i \leftarrow H_i + \frac{\alpha E_n X_{n-i}}{X_i}.$$

This type of computation is very simple to handle by DSP. FDSP-3 is used for small delay applications. For longer delays, an ASIC chip has been developed.

An audio echo canceler is also under study. Delays longer than 300 ms must be covered in this application because the echo signal is transmitted through air.

FDSP-4 chips are being used to build a trial system.

4.5 Delay-time equalizer for the MF radio transmission

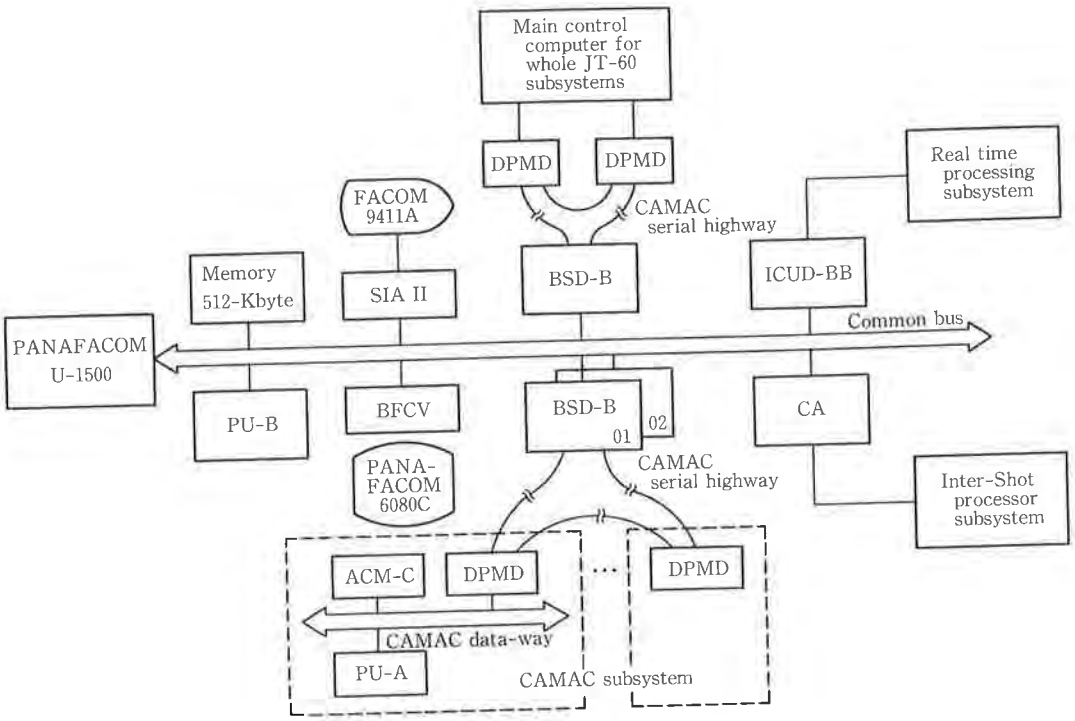
When an AM broadcasting signal is sent from one station to another, an NTT F-1 transmission line is often used. However, due to the transmission line characteristics of the F-1 line, the signal sent through this line has group delay degradations. This group delay will lose sound clarity or may induce significant problems for the transmission of emergency signals. A delay equalizer is inserted to cancel these group delay characteristics.

We developed such an equalizer system by using FDSP-3 chips.

In this system, digitized input signals are first converted into frequency domain signals by DFT. The group delay characteristics are canceled in this frequency domain and then converted back to the time domain signal by IDFT.

Figure 23 shows the system configuration. A 512-point FFT is adopted due to the limit of memory size of FDSP-3 and signal speed. It has a structure of four-phase FFT blocks as shown in the figure. Each block has independent DFTs.

Each period of 512 sampling points are divided into four 128-point blocks and the 128-point effective signals is followed by 384 successive "0"s. This structure allows the spread-



ACM-C : Auxilialy controller with microcomputer type-C
BSD-A, -B : Byte serial driver type-A, -B
CA : Channel adapter
CAMAC : Computer automated measurement and control
DPMD : Dual port memory with D-port
ICUD-BB : Digital interface control unit type-B
PU-A, -B : Pri-processor unit type-A, -B

Fig. 24—Data processing system for plasma experiment.

ing power of the frequency domain signal to remain in the location where "0"s are originally inserted so that the recovered signal can be reconstructed.

To implement this kind of system with a 16-bit fixed point DSP FDSP-3, some computation techniques are introduced. One is the shifting of intermediate results to an efficient bit length. The second is that a 128 successive input signal series constructs a group and each group is regarded as big if the amplitude of the biggest signal is more than 0 dBm. Otherwise it is regarded as a small group. If the series is small, all the input signals are multiplied by A and applied by DFT conversion and divided by A when the time domain signals are reconstructed. The dynamic range of the fixed point DSP FDSP-3 is thus efficiently used.

There are a total of 98 962 machine cycles per 512-point signals.

4.6 Diagnostic data processing system for plasma experiments

Many studies have been made of nuclear fusion as a new energy source. JAERI (Japan Atomic Energy Research Institute) has been carrying out a large-scale project called JT-60 (JAERI Tokamak-60) for this purpose.

When the temperature and density of the plasma become high, nuclei collide with each other and high-energy and neutrons are produced. When the plasma emits the same amount of energy as it receives from the outside, it is called critical plasma and parameters such as temperature and density are called the critical plasma conditions. In order to use nuclear fusion as an energy source, it is necessary to know how to artificially create the critical plasma conditions. To have such conditions, unstable plasma must be closed in a certain area and heated to a very high tempera-

ture. In the JT-60 experiment, a doughnut shaped vacuum tube is prepared. The tube is surrounded by various types of coils which produce magnetic fields. The plasma is then locked in a "magnetic cage". This method is widely known as Tokamak.

The purpose of the JT-60 system in the current stage is to collect various data when the critical plasma conditions are achieved. In the JT-60 Tokamak system, there are some subsystems to carry out various types of experiments. As one subsystem, the real-time processing subsystem[22] is prepared to collect the various types of measurement data under critical plasma conditions.

Figure 24[22] shows the configuration of this subsystem. There are four main functions: the command generating function, the real-time processing function, the subsystem control/communication function and a communication function to another subsystem named Inter-Shot processor subsystem[23] in which the main analysis of the experiment is performed.

In the experiment, the high-temperature plasma is generated once every 10 minutes and last for 5-10 seconds. In the middle of this discharge period, measurement data is collected every millisecond and is processed in real time.

The arithmetic processing performed by the ACM-C, PU-A, PU-B and FFTP (Fast Fourier Transformation Processor) is the conversion of the measurement data into parameters needed to control the discharge.

The FFTP performs the FFT. The ACM-C, PU-A and PU-B perform the inverse Abel transform and other matrix operations and linear interpolation. The ACM-C, FFTP, and PU-A perform independent data calculations in each measurement section. The PU-B performs common data operations. The calculations are optimized and changed according to the progress of the experiments. Facilities are thus provided for developing firmware programs for the PU-A and PU-B. FDSP-3 is used as the main processor for these blocks.

Fig. 25—Blood flow analysis system ($920^{l} \times 560^{b} \times 1410^{h}$: mm).

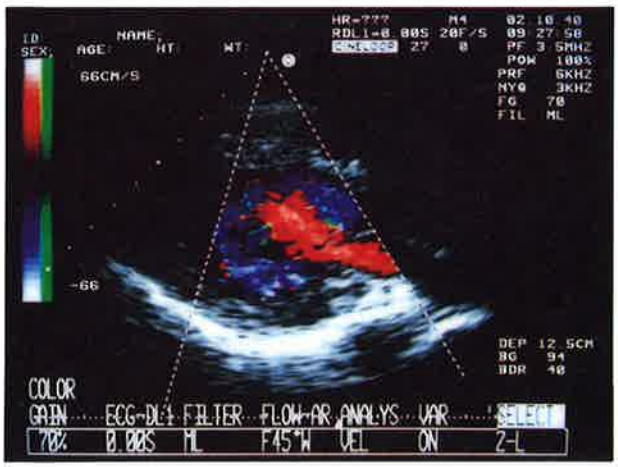


Fig. 26—Blood flow display example.

4.7 Blood flow analysis

FDSP-3 is also applied to medical engineering. Figure 25 shows the ultrasonic blood flow analysis system for heart disease testing. This system displays the two dimensional blood flow velocity distribution in a heart by using colored images with a video rate of 20 frames/s. This enables the analysis of blood flow.

Digital signal processing in this system is the high-pass filtering and center frequency

detection by the auto-correlation method. The pointwise blood velocity is detected by analyzing the Doppler signal of the ultrasonic wave reflected by a moving blood cell. The high-pass filter removes noise caused by muscle movement. To accomplish all the processing within one frame, 17 FDSP-3s are used in parallel. This achieves 512 points/scan line analysis with a 2 ms/scan line processing speed. Figure 26 shows one scene of blood velocity display in color.

5. Future topics

The trend of general-purpose DSPs which are targeted towards speech signal applications has been described above. Today, the most popular DSPs in the commercial field are 16-bit fixed-point processors. However, the development target is being shifted to higher level processors and 32-bit floating point chips are replacing the 16-bit ones. However, to make digital signal processing technology more popular in various areas of application, Fujitsu is now considering future trends.

There are two aspects to consider about future DSPs.

The first is economics.

To meet the final cost target, there will be many cases in which a DSP specific to a new application with the same architecture as a general-purpose device will be quite effective. The authors introduced Fujitsu's unique way to develop such devices under the concept of DSP ASIC.

The second is the diversification of application areas of the DSP type approach.

As already described, digital signal processing technology originated from speech signal processing. With the advancement of semiconductor technologies, high-speed devices will be available and DSPs for different areas will become popular. ar.

5.1 DSP ASIC

Figure 27 shows the diversifying application area of DSPs including the first and second generation DSPs and DSP ASICs.

DSP ASICs can be classified as follows:

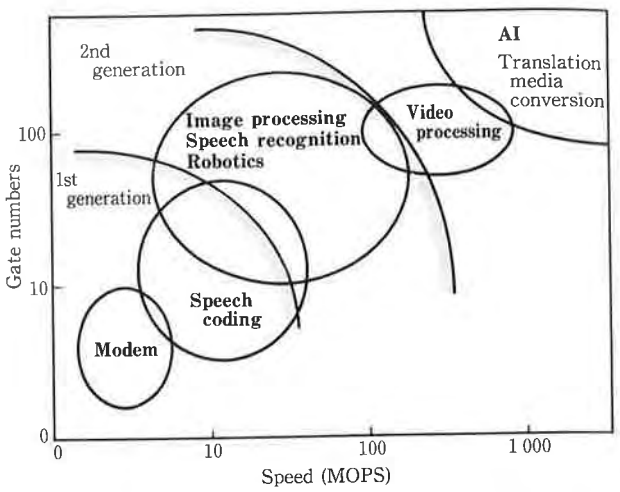


Fig. 27—DSP application diversification.

1) To achieve a large processing unit which has functions that cannot be implemented on a single general-purpose DSP.
2) To achieve a small processing unit for which a general-purpose DSP is too big.

It is not easy to develop many ASIC type chips for each application. It would be nice to have some good methodology to develop such devices easily.

The authors have made the following method to develop new devices by using our past experience in developing general-purpose DSPs.

1) The authors have settled the restriction on the architecture of the processor. That is, the FDSP-4 is defined as a base processor.
2) To develop relatively small DSPs, a subset of FDSP-4 is taken in the sense of both hardware and software to design ASIC chips.
3) For large DSPs, the authors use the main part of the base processor as the core, and these add dedicated circuitry around it.

As a small DSP ASIC, Fujitsu developed MB87528 for a 8- or 16-kbit/s speech codec. For this, the authors added the following changes to the FDSP-4 base processor.

1) Change in memory area

To implement both encoder and decoder functions on the same chip, Fujitsu increased the instruction ROM area and made the RAM area as small as possible.

2) Decrease in data length

To optimize data length, we reduced the number of bits for the mantissa of each floating data from 16 to 8.

3) Eliminate useless functions for codecs

We eliminate register files and parallel I/O.

To develop a big DSP ASIC, we developed the DSP core by modifying a FDSP-4 base processor as follows:

1) To optimize the data length, Fujitsu modified the data form to 18E6 which can be used for many kinds of speech signal applications.
2) We eliminated the I/O block and left the I/O design to each user.
3) To reduce the ROM area, we restricted the instruction code.

This approach is open to DSP customers and they can design their own chip with this core. The core is also available as a new general-purpose DSP MB86220.

In the future, it will be possible to use more than one core on a single DSP chip. At the laboratory level, Fujitsu has developed a test device by 0.8-μm process on which three cores are mounted.

5.2 DSPs for faster applications

As is shown in Fig. 27, the area of DSP applications is spreading. Although DSP technology originated from various speech signal processing, DSPs for different applications are becoming popular with the advancement of semiconductor technologies. Among them, DSPs for video signal processing is most promising. At present, some low-bit-rate video codecs such as that at 64 kbit/s will become popular. For this system, vector quantization and/or DCT (Discrete Cosine Transform) will be used. The processor must thus have a function to cope with this type of computation. For a video processor, the key point will be how efficiently memory access functions can be implemented on a chip.

Ultra-high-speed processors using advanced semiconductor technology such as GaAs and HEMT are also expected.

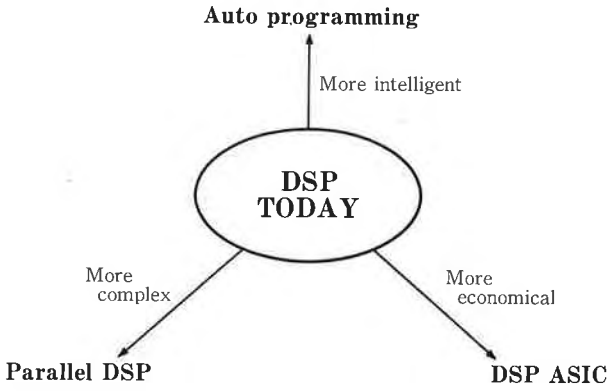


Fig. 28—Direction of DSP advancement.

5.3 Future trends

Beyond these new aspects of DSP development, further advancement of DSP is expected. As shown in Fig. 28, it is necessary to consider three directions for future DSP advancement.

From the economical aspect, Fujitsu is starting to enter DSP ASIC as discussed here. In this aspect, it is necessary to study the design methodologies more to implement DSP ASIC in a simple way. A total design system from high-level circuit description to fully automatic layout is needed.

For the complexity of the circuit, it will be possible to apply a wafer scale level device in the near future. In this aspect, a parallel DSP will be a promising approach.

For such DSP systems, a high-level language which can cope with the highly parallel signal processing performance is needed. System debugging in real-time will also be a critical issue.

For the final aspect, it is necessary to consider how to improve the DSP system. The most important issue for today's DSP is basically high performance with the sacrifice of programming complexity. However, the more complex the system becomes, a simpler method for algorithm implementation will be needed. The DSP technology will be closely related to other processing technologies. A new method such as the neural network method will be introduced for DSP systems. It is also expected that the applications of DSP themselves will have more intelligent aspects. DSP

applications related to human perception will increase the popularity of DSP technologies.

6. Conclusion

This paper has reviewed Fujitsu's DSP development history. Details of the latest DSP (FDSP-4) were introduced as the completed version of the second generation DSP.

Applications from different fields were also described.

DSP LSI development is continuing and the authors are planning to describe their results in a future paper.

References

1) Murano, K., Unagami, S., and Tsuda, T.: LSI processor for digital signal processing and its application to 4 800 bit/s modem. *IEEE Trans. Commun.*, **COM-26**, 5, pp. 499-506 (1978).
2) Moriya, T., Murano, K., and Gambe, H.: Digital Signal Processing, *FUJITSU Sci. Tech. J.*, **22**, 4 (Special Issue of Communications), pp. 343-354 (1986).
3) Tsuda, T. et al.: LSI digital signal processor tailored to voice synthesis. Proc. Int. Conf. Commun. 1981, p. 72. 1.
4) Kikuchi, H. et al.: A 23 K gate CMOS DSP with 100 ns multiplication. Proc. Int. Solid-State Circuits Conf., 1983, pp. 128-129.
5) Ikezawa, T. et al.: A general purpose digital signal processor. Proc. ECCTD 1183, p.S.1.1.
6) Mochida, Y. et al.: A High Performance LSI Digital Signal Processor for Communication. *IEEE J. Select. Areas Commun.*, **SAC-3**, 2, pp. 347-356 (1985).
7) Gambe, H. et al.: On the Design of a High Performance LSI Circuit Digital Signal Processor for Communication. *IEEE J. Select. Areas Commun.*, **SAC-3**, 2, pp. 357-368 (1985).
8) Gambe, H. et al.: A high speed DSP LSI and its application. Proc. globcom 1983, p. 45.5.1.
9) Ikesaka, M. et al.: Evaluation of experimental compiler for digital signal processor. Proc. Nat. Conv. Inform. Proc., Jpn., 1982, p. 20-5.
10) Gambe, H. et al.: A 32 bit floating point digital signal processor FDSP-4 and its application to the communication systems. Proc. GLOBCOM 1987, pp. 12.1.
11) Gambe, H. et al.: Development of 32 bit floating point digital signal processor FDSP-4. Tech. Meet. Inst. Elect. Commun. Engrs., Jpn., 1987, CAS87-135, pp. 91-96.
12) Tsuda, T. et al.: CMOS LSI DSP and its application to voice-band signals. Proc. ICASSP 1983, p. 20.10: see also Mochida, Y.: VLSI high speed data modem. Proc. GLOBECOM 1983, p. 45.8.
13) Maeda, Y., and Taka, M.: 32 kbps Speech Encoding Equipment. Proc. Nat. Conv. Inst. Elect. Commun. Engrs., Jpn., 1985, p. 9-1.
14) Maruyama, I. et al.: Implementation of CCITT G.721 ADPCM CODEC using a general purpose digital signal processor. Proc. Nat. Conv. Inst. Elect. Commun. Engrs., Jpn., 1985, p. 9-5.
15) Matsumura, T. et al.: Implementation of 32 kbit/s ADPCM CODEC using a general purpose digital signal processor. Proc. GLOBCOM '85, 1985, p. 37.1.1.
16) Takebayashi, T. et al.: A 32 kbit/s ADPCM with improvement in coding characteristics for 9 600 bit/s modem signal. Proc. ICASSP., 1986, pp. 2187-2190.
17) Tomita, Y. et al.: A Digital Signal Processing of a 16 kbit/s APC-AB CODEC by Fixed Point Digital Signal Processor (FDSP-3). Proc. ICASSP 1986, p. 6.11.1.
18) Taniguchi, T. et al.: A 16 kbit/s ADPCM with Multi-quantizer (ADPCM-MQ) CODEC. Proc. Inst. Elect. Commun. Engrs., Jpn., CAS85-196, 1986.
19) Taka, M. et al.: 64 kbit/s High Quality Speech CODEC for Video Conference. Proc. Inst. Elect. Electrons Commun. Engrs., CS83-196, 1984, pp. 77-84.
20) Yasukawa, H., and Shimada, S.: Voice-band Bothway Repeater using Echo Cancellers for Automatic Call-Transfer Services. Proc. ISCAS 1985, pp. 1489-1492.
21) Takahashi, H. et al.: A sub-micron CMOS echo canceller using a DSP cell. Proc. Inst. Solid-State Circuits Conf. 1987, pp. 148-149.
22) Iida, S., Yahiro, K., and Kumada, M.: JT-60 Real Time Processing Subsystem. *FUJITSU*, **37**, 1 (Special Issue: Nuclear Fusion Research and Diagnostic Processing System), (in Japanese), pp. 31-36 (1986).
23) Mochizuki, O., Yanai, Y., and Kambe, T.: JT-60 Inter-Shot Processor subsystem. *FUJITSU*, **37**, 1 (Special Issue: Nuclear Fusion Research and Diagnostic Processing System), (in Japanese), pp. 14-22 (1986).

Toshitaka Tsuda

Computer Network and Visual Communications Laboratory
FUJITSU LABORATORIES, KAWASAKI
Bachelor of Electrical Eng.
The University of Tokyo 1970
Dr. of Philosophy in Electrical Eng.
The University of Tokyo 1975
Specializing in Computer Communication and Digital Signal Processing

Hirohisa Gambe

Communication Device Technology Dept.
Technology and Production Eng. Div.
FUJITSU LIMITED
Bachelor of Electronics Eng.
Tokyo Institute of Technology 1973
Master of Science in Electronics Eng.
Polytechnic Institute of New York 1978
Specializing in Communication VLSI and Digital Signal Processing

Ryusuke Hoshikawa

MOS LSI Div.
FUJITSU LIMITED
Bachelor of Electrical Eng.
Hokkaido University 1966
Master of Electrical Eng.
Hokkaido University 1968
Specializing in MOS LSI Development