

ADVANCED SINGLE-CHIP SIGNAL PROCESSOR

T. Nishitani, I. Kuroda, Y. Kawakami,
H. Tanaka and T. Nukiyama

NEC Corporation
1-1, Miyazaki 4-chome, Miyamae-ku, Kawasaki 213 Japan

ABSTRACT

A CMOS single chip signal processor, which has 32-bit floating point arithmetic units and large capacity on-chip memories, has been developed. The processor, having a floating point parallel multiplier, a floating point accumulator, two 512-word data RAMs, a 1024-word data ROM and a 2048-word program ROM, is implemented within a 15.4 x 8.4 mm chip area, containing 370,000 elements. As the processor is designed to perform highly accurate multiply-accumulate operations for digital filtering and to attain complex addressing capability for FFT computation, this processor can execute FIR computation at the 150 nsec per tap rate, as well as achieve 1024 point complex FFT computation in 12.3 msec.

1. INTRODUCTION

First generation signal processors, featured by fixed point arithmetic operations, have been successfully introduced to signal processing applications, such as speech coders, data modems, echo cancellers and so on. However, ever increasing demands for advanced digital signal processing applications have run over the first generation signal processors capability in these days. Wider dynamic range operations and higher precision operations, as well as high speed processing, are fundamental requirements for complex real time applications, such as multi-dimensional adaptive filtering and iterative matrix-based processing. Floating point arithmetic employment is desired for coming second generation signal processors.

Thanks to the up-to-date LSI technology, a 32-bit floating point signal processor has been developed for opening the second generation signal processor era. The processor architecture design concept has been based on a first generation signal processor 7720[1]. However, every function block in the developed processor is enriched, in order to expand its signal processing capability. In addition to 150 nsec floating point multiply-add operation capability, the developed processor contains large capacity on-chip memories with enriched memory management units. Complex address generations, required for FFT processing, can be successfully performed in these memory management

units without any ALU support. External program/data memory handling capability is also added to the processor. Due to enriched function blocks and improved processor cycle time, several bench mark tests for digital signal processing jobs show that the developed processor has about 10 times higher throughput than its predecessor.

In this paper, after describing the design concept and overall architecture, attention is focused on arithmetic units, memory addressing and I/O functions. The paper also presents the instruction sets and application examples.

2. DESIGN CONCEPT AND OVERALL ARCHITECTURE

The architecture has been designed with special consideration applied to the following points. First is the employment of high performance floating point multiply-accumulate arithmetic units. Second is parallel address computation, by multi-address units, which can efficiently generate an address sequence for FFT processing. The third point is the employment of a high speed parallel I/O interface.

The developed signal processor block diagram is illustrated in Fig. 1. The processor consists of 3 main blocks, the arithmetic block, the memory block and the control & I/O block. Each block is connected to a 32-bit main bus. The arithmetic block includes a floating point multiplier and a floating point accumulator, combined in a pipelined multiplier-accumulator form. The memory block includes three data memories and a program memory, each having its own addressing units. The control & I/O block includes an instruction decoder, a serial I/O unit, a parallel I/O with external address port and a status register. The following describes more precise features in these 3 blocks.

Arithmetic block

The arithmetic block includes a 32-bit floating point multiplier (FMPY), a 55-bit floating point arithmetic logic unit (FALU), 55-bit working registers (WR) and other control or scratch pad registers (FLAG, TR).

8. 7. 1

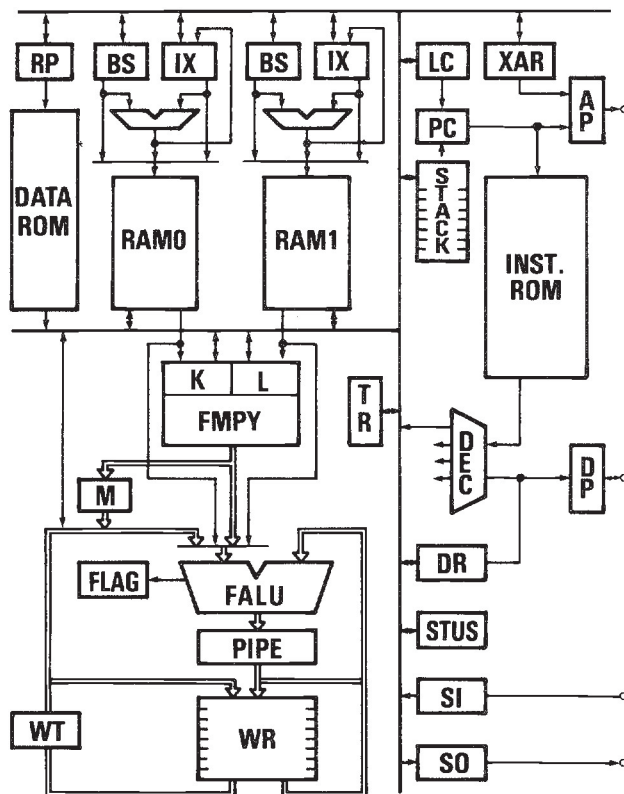


Fig. 1 ARCHITECTURE

FMPY and FALU are combined to perform multiply-accumulate operation in every instruction cycle. FMPY has two input data registers (K,L), which are directly combined to a respective 32-bit data RAM. Figure 2 illustrates the 32-bit floating point format and the 55-bit floating point format, employed in the processor. The 32-bit format consists of an 8-bit exponent part and a 24-bit mantissa part, both of which are represented in two's complement format. The 55-bit format corresponds to the FMPY output format, which consists of an 8-bit exponent part and a 47-bit mantissa part. In this format, FALU executes the floating point operations. One input to FALU is selected from eight registers in WR, while the other input is selected from FMPY output, outputs from data RAMs, and a main bus. Although FALU performs pre-denormalization and mantissa addition/subtraction in one instruction, the result is not normalized in order to speed up the instruction cycle. When a normalized result is required, the unnormalized result can be normalized by a normalization instruction. FALU has a format conversion function to and from the IEEE standard 32-bit floating point format, in order to allow communication with general purpose microprocessors.

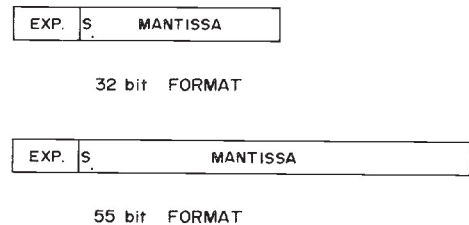


Fig. 2 DATA FORMAT

FALU and WR are connected to the 55-bit main bus, whose upper 32 bits are connected to the 32-bit main bus. Data transfer from WR to the 32-bit memory (RAM0, RAM1) can be controlled by working register transformer (WT). In order to store the 55-bit data into the memory, the upper 32-bit and lower 23-bit data can be transferred to the memory in two steps, using the 32-bit main bus. By using the above function, double precision operations can be easily executed.

The processor also has 24-bit fixed point arithmetic capability. For fixed point arithmetic operations, the exponent part is set to zero. Therefore, FMPY executes 24 x 24-bit multiplication, and the FALU has 47-bit arithmetic and logical function.

Memory addressing

The processor includes two 32-bit 512-word data RAMs (RAM0, RAM1), a 32-bit 1024-word data ROM (DATA ROM) and a 32-bit 2048-word instruction ROM (INST. ROM). As they have their own address calculation units, data preparations for operations can be performed independently. The addressing unit for the data RAM consists of an index register (IX) and a base register (BS), both of which have increment, decrement and clear to zero functions. The data RAM can be addressed by IX, or BS, or a sum of these two. Another RAM address function is to store the sum of the two registers into IX. Register BS has a modulo counter function, whose counter length is selected from 1 to 7 bits.

The data ROM is addressed by an ROM pointer (RP), which has increment, decrement and power of two increment functions. The last function increments the ROM pointer from 2 to the 9th power of 2.

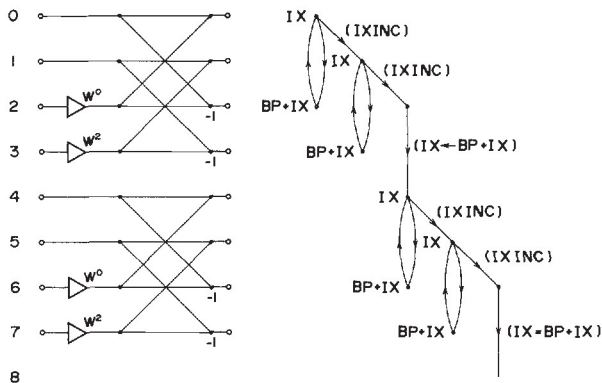


Fig. 3 FFT ADDRESS TRANSITION SCHEME

One major application of the enriched address generation units is for the Fast Fourier Transform computation. Since the developed processor employs floating point arithmetic, FFT can be computed without considerations on dynamic range limitation. When the real part of the complex signal is stored in RAM0, and the imaginary part is stored in RAM1, FMPY and FALU efficiently computes the complex multiplication between a complex signal sample and a twiddle factor, using subbuses connecting memories and multiplier input registers. The addressing unit for each data RAM and ROM can be configured to generate FFT addresses, efficiently. Figure 3 illustrates a part of the decimation-in-time FFT signal flow and an address generation scheme. At one stage in the FFT computation, consecutively addressed butterflies appear in several separate groups. There are two butterfly groups in Fig. 3. When the distance between the butterfly dual node pair is stored in BS (which is 2 in Fig. 3), addresses for the first nodes in the first butterfly group (nodes 0 and 1) are generated by incrementing IX from zero (IXINC). The other node in the dual node pair is accessed by the addition of BS and IX ($BP+IX$). The first node address transition from the last butterfly operation in the first group to the second butterfly group (nodes 4 and 5) is performed by storing the sum of two pointers into IX ($IX \leftarrow BP+IX$). In this way, the address sequence for FFT can be generated. Moreover, hardware bit reverse function is supported in FALU.

The ring memory technique, which emulates a FIFO memory or a delay line in an FIR filter, can be realized using the modulo counter function in BS. By setting the modulo counter word length, 2 to the 7th power of 2 word, a ring memory can be realized. Thus, datashift functions in a FIFO memory or an FIR delay line can be virtually realized by modifying address pointers.

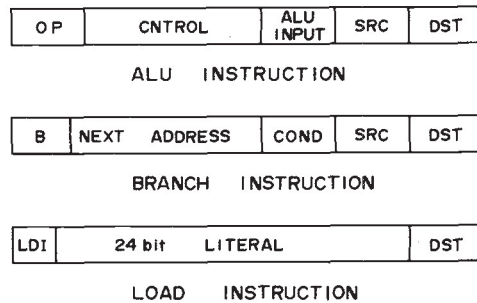


Fig. 4 INSTRUCTION FORMAT

I/O Units

The I/O units consist of a 32-bit parallel I/O port (DP), a 12-bit address port (AP) and a serial input (SI) / output (SO) port. The processor has two parallel I/O modes, the master mode and the slave mode. In the master mode, the processor can use a directly addressable 32-bit wide 8192-word deep off chip memory. This 8192-word external memory is divided into 2 areas, namely a 4096-word high speed access area, and a 4096-word low speed access area. The high speed access area can be used as the external instruction memory. External program fetch and execute can be performed within normal processing cycle of 150 nsec. On the other hand, when the area is used as the data memory, read/write operation, whose address is specified by the address register (AR), is performed in two instruction steps (300 nsec).

The serial input/output port enables the processor to interface with such other devices as PCM codecs or A/D converters. The data width is selected from 8, 16, 24 and 32 bits. The data width and the data format (MSB first or LSB first) can be selected with the status register (STUS). The serial clock can be provided by either the internal master clock or an external clock, up to 5 MHz.

3. INSTRUCTION SETS

The developed signal processor has 3 individual 32-bit instruction formats, ALU, branch and load immediate, as shown in Fig. 4.

The ALU instruction consists of an ALU operation field (OP), a control field (CNTROL), an ALU input selection field (ALU INPUT), and a data transfer field (SRC, DST). In the control field, there are various kinds of processor resource control operations, such as memory address modification, auxiliary ALU operation and external

I/O function. A non-conditional branch operation and a direct ROM address set operation are also included in this control field. Conditional and non-conditional branch operations in branch instruction can be performed with data transfer operations. The load instruction is used to set the 24-bit immediate data into the mantissa part. For the 32-bit data set, the ROM direct access using an ALU instruction can be used.

4. LSI IMPLEMENTATION

The developed processor is implemented within a 15.4 x 8.4 mm chip area containing 370,000 elements, using advanced CMOS technology. Figure 5 shows a chip photomicrograph. The chip includes 12K-bytes of ROM, 4K-bytes of SRAM, a 32-bit floating point multiplier, a 55-bit floating point ALU and other control blocks. The parallel multiplier adopts a combination of third order modified Booth's algorithm and Wallace tree configuration in order to realize 150 nsec cycle time.

5. APPLICATIONS

The developed processor provides high speed and highly accurate calculation of multiply-accumulate operation by floating-point hardware. For an FIR filter, the processing speed is 150 nsec per tap, using the ring memory technique. A second order recursive filter can be implemented in 6 steps, which requires only 0.9 μ sec with normalized 32-bit floating point format. This multiply-accumulate structure is also efficient for computing an auto-correlation function, used in AR modelling or pitch detection.

In order to realize complex algorithms, such as Kalman filtering or Levinson-Durbin algorithm, highly accurate division is required, as well as multiply-accumulate operation. The inverse or the square root of a 32-bit floating point data can be computed in 4.5 μ sec or 9.0 μ sec, by using the Newton-Raphson iteration technique.

Another important application is the Fast Fourier Transform computation. An 512-point complex FFT can be computed in 4.6 msec using only on-chip memories. If the 2048-word of 32-bit external memory is attached, a 1024-point complex FFT can be computed in 12.3 msec with full 32-bit floating point accuracy. This computation process includes the data input process in bit reverse order, two 512-point complex FFT computations, last stage 512 butterfly computations and data output process.

This 32-bit floating point signal processor will expand the advanced application area which requires wide dynamic range and high precision computation, such as array signal processing and robotics, and also in the area of image/vision processing.

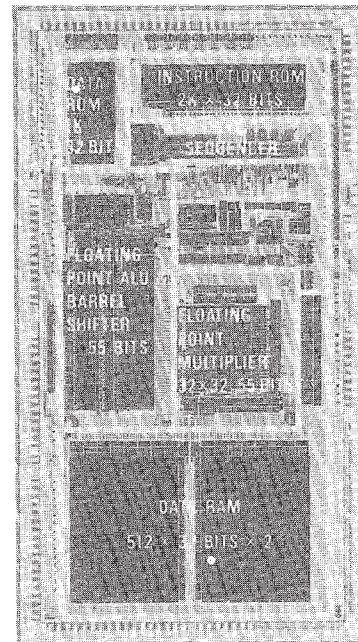


Fig. 5 PHOTOMICROGRAPH

6. CONCLUSION

Design concept and architecture for a 32-bit floating point signal processor has been described. The high performance multiply-accumulate architecture enables high speed and high accuracy filtering. Memories and their addressing units are configured so as easily to perform efficient FFT computations. A 512-point complex FFT can be computed in 4.6 msec. If a external memory is attached, a 1024-point complex FFT can be computed in 12.3 msec. This 32-bit floating point signal processor will expand signal processors' application area to include high precision and wide dynamic range fields such as robotics and vision processing.

ACKNOWLEDGMENT

The authors wish to thank H. Goto and R. Maruta at C&C Systems Research Labs. and M. Suzuki and T. Furuhashi at Microcomputer Product Div. for their constant support in the present project.

REFERENCE

- [1] T. Nishitani, Y. Kawakami, R. Maruta and A. Sawai, "LSI SIGNAL PROCESSOR DEVELOPMENT FOR COMMUNICATIONS EQUIPMENT", Proceedings of ICASSP'80, vol. 2 pp.386-389, 1980 Denver