An NMOS Microprocessor for Analog Signal Processing

MATT TOWNSEND, MEMBER, IEEE, MARCIAN E. HOFF, JR., SENIOR MEMBER, IEEE, AND ROBERT E. HOLM

Abstract—A special purpose microprocessor for real time processing of analog signals is described. Design and implementation of architecture allowing a user programmable and erasable read only memory (EPROM), a 25 bit digital processor and a 9 bit analog acquisition system on the same substrate is discussed. The relationship between the device's resources and specific signal processing building blocks is discussed.

Introduction

DIGITAL processing of real time analog signals has been largely confined to specially configured high speed processors. These processors range in complexity from large array processor peripherals used primarily to perform the fast Fourier transform to bipolar large-scale integration (LSI) bit slice systems microcoded to perform signal processing functions (primarily multiplications and analog I/O control) efficiently.

High speed multiplication capability is key to performance standards for digital signal processors. Dedication of a system architecture to perform these multiplications, however, can be detrimental to overall flexibility when required to perform a signal processing block that needs no multiplies. Examples of such blocks may include limiters, rectifiers, and comparators.

Another advantage to digital processing of analog signals is flexibility through programmability. This allows analog problems to be solved using a series of program sequences. Many high speed signal processors lose their advantage, either in cost, performance, or both, when their instructions must be delivered by an ancillary processing system. Pipelined processors and microcode mapping for bit slice machines may also complicate firmware to the point where programming of a general purpose machine becomes a tedious task, at best.

A specially configured processor has been developed capable of realizing major analog systems. Included on the same substrate is a user programmable and erasable read only memory (EPROM), a digital processor with random access memory and a 9 bit analog to digital, digital to analog acquisition system. The chip is realized using an n-channel MOS technology. The processor is oriented toward real time processing of digitized analog signals. It is capable of realizing functions such as filters, oscillators, multipliers, limiters, rectifiers, and various nonlinear approximations.

The flexibility of the architecture allows the realization of complex analog modules such as modems, multifrequency tone receivers, equalizers, and frequency sources. The user may customize the chip to each application with a program sequence in the on-chip EPROM.

All operations performed by the analog and/or digital subsystems do so under program control. The EPROM has 192 words by 24 bits. The 24 bits are divided into 5 fields that control each element within the signal processor chip. The EPROM then, can be thought of as a microprogram controller.

The processor contains a wide data word (25 bits) for signal processing computations, and arithmetic for signal processing functions.

Although there are some digital signals available to the user, such as internal clocks and operational indicators, the analog subsystem provides the majority of the chip's external interface.

Manuscript received June 13, 1979; revised August 17, 1979 and September 13, 1979.

The authors are with the Applications Research Group, Intel Corporation, Santa Clara, CA 95001.

Choice of Technology

An NMOS EPROM technology was selected for a variety of reasons. In general, it was chosen because the process capabilities existed to implement the required signal processing subsystems and thus provide single chip solutions to complex analog problems.

By using EPROM as a storage element for the instruction sequence, the burden of ancillary central processing unit (CPU) interface is eliminated. Additionally, parameters such as center frequencies, limit levels, and such can be changed quite easily and without incurring costs and delays involved with mask ROM's or fuse-link PROM's.

The mix of EPROM and high density digital logic on the same substrate has been proven effective and reliable [3]. This relieves the potential burden of having to implement a multichip CPU + memory configuration and, in general, lower production costs while increasing system reliability.

The design of NMOS analog components is established and understood [4], [5]. By adapting the design techniques to the process, an analog acquisition system fully compatible with the digital processor was implemented.

Architecture

Fig. 1 shows the system architecture. The major subsystems implemented on the same substrate include the EPROM controller, the 25 bit digital processor, and the 9 bit analog to digital/digital to analog acquisition system. These three subsystems will be discussed separately.

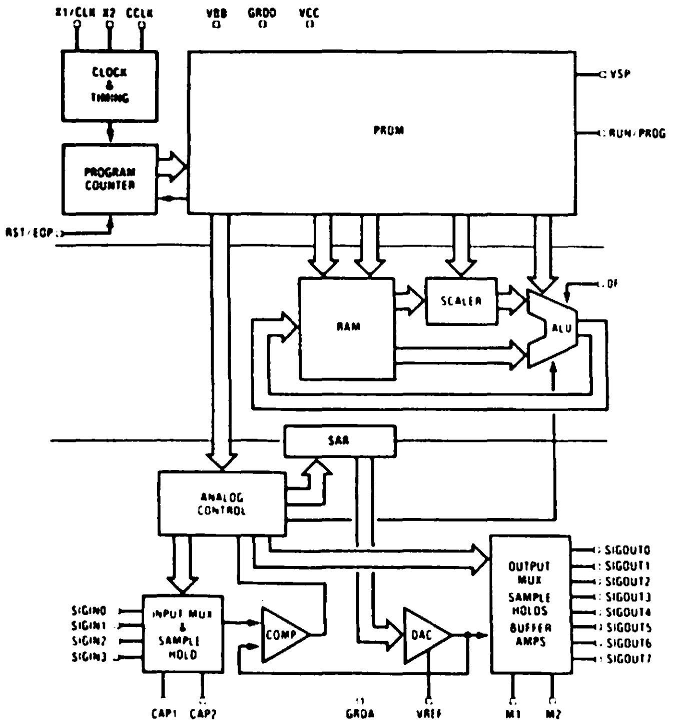

Fig. 1. Block diagram of chip.

The EPROM Controller

The controller section contains 4608 bits of EPROM, and is arranged as 192 words of 24 bits each. Each word corresponds to one instruction.

The PROM section acts as the system controller. Each 24 bit control word contains bit patterns that determine the operations to be performed by the analog and arithmetic sections. The control word is divided into five fields of which one controls the analog section and the remaining four control the arithmetic section. The four arithmetic section control fields include two 6 bit fields which perform RAM operand selection, a 4 bit scaler control field, and a 3 bit arithmetic and logic unit (ALU) control field.

EPROM word addresses are numbered from 0 to 191. In normal operation all locations are accessed in sequence and no program jumps are allowed. The EPROM returns to location 0 upon completion of execution of the command in word 191, or if an EOP instruction is located in the analog control instruction field. The EOP feature allows the program to be terminated at the end of a user's program that does not use the full complement of 192 words.

The PROM may be thought of as a crystal or clock controlled cycle generator as it determines the sampling frequency of the analog signals. If an input is sampled once per program pass, the sampling frequency f_s is given by

$$f_s = \frac{1}{NT} \tag{1}$$

where N is the number of words (instructions) in the program and T the time required to execute one instruction.

The EPROM fetch/execute cycle is pipelined, with the next four instructions being fetched while the previously fetched instructions are being executed. Although otherwise invisible to the user, this technique makes it necessary to insert the EOP instruction (if used) in a word with an address divisible by four e.g., 0, 4, $\cdots$ 188. The term N in (1) is altered in four word sections. If a finer adjustment of the sample rate is desired, it must be accomplished by adjusting T, the cycle speed.

The Analog Subsystem

The analog subsystem blocks are shown at the bottom of Fig. 1. All analog operations are controlled by a 5 bit PROM field that designates a specific operation as well as selects a channel number or bit position to be converted. All input and output sampling is done under program control. Because of this, program jumps are restricted to jumps back to the first location. This preserves the sample rate of the incoming signal, an important consideration for a digital signal processing environment.

Analog elements include a 4 to 1 input multiplexer which uses an external sample and hold capacitor, in conjunction with an on-chip bipolar to unipolar signal conversion circuit. A high speed chopper stabilized voltage comparator [8] drives the successive approximation analog to digital conversion logic. The registered value is sent to an 8 bit digital to analog converter, thus closing the input conversion loop. The successive approximation register provides a link between the digital processor and the analog system. The processor may feed data to the digital to analog converter for subsequent outputting. In this case, an operational amplifier [5] buffers the digital to analog converter to a one to eight demultiplexer with on-chip sample and hold circuitry.

The acceptable input or output signal range is established by an externally supplied voltage reference. For detector applications where outputs will be two-state, provisions are made on-chip for output levels to be translated to TTL compatible drive capability.

Table I shows some typical performance data for the analog system.

Input Signal Conversion

Fig. 2 shows the circuitry used to convert a bipolar input signal to positive only values. During input sampling, Devices 1 and 2 are turned on and the capacitor is charged to the input value. At the same time, Devices 3 and 4 are off and the reference input to the comparator is set to 0 V by the digital processor. After an input instruction, the capacitor is allowed to float. For analog to digital conversion of the sign position, Devices 3 are turned on and a voltage comparison is made between the bipolar input signal and 0 V. Based on the results of that conversion, Devices 3 will remain on if the input was positive, Devices 4 if negative. This, in effect, rectifies the incoming signal. The conversion of the magnitude is then done with positive only values at the comparator input. A similar technique is used for unipolar to bipolar conversion when outputting. This technique, then, implies only one positive reference voltage is required.

The resulting full scale number system, using this technique,

TABLE I
ANALOG CHARACTERISTICS

Parameter	Typical	Comments
A to D linearity	±0.3 LSB	LSB = 3.87 mV
A to D offset	-0.2 LSB	Corrected for number system offset
D to A linearity	±0.3 LSB	LSB = 3.87 mV
D to A offset	-10 mV	
D to A gain	0.93	
Output impedance	500 Ω	

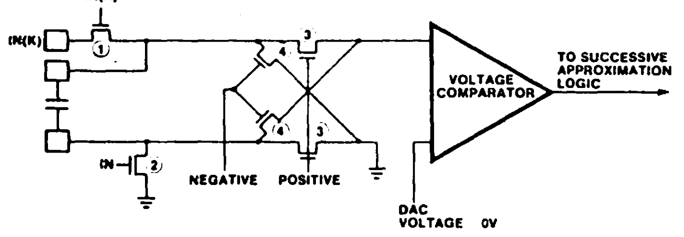

Fig. 2. Sign bit conversion.

is 1's complement, that is, there is a plus and minus 0 represented. The successive approximation register (SAR)/digital to analog register (DAR) value needs to be 2's complement in order to operate correctly in the processor section. To accomplish the number system conversion, the A-D sign bit is used to conditionally complement the magnitude. The results of this conversion, however, will yield an A-D offest of - 1 LSB.

THE DAC

The digital to analog converter (DAC) (Fig. 3) [4] uses an array of 256 diffused resistors in a series string. The resistors are folded back on each other every 16 providing a compact layout and ease in incorporation of 2 level decoding similar to memory circuits. An added advantage is relative insensitivity to process and temperature gradients.

The resistor edges are defined on one side by the gate of the column transfer devices and on the other side by a dummy polysilicon gate connected to a voltage potential sufficient to allow isolation from the adjacent resistor. The DAC output is developed using a 16 to 1 multiplexer selected by the Row decoders. Thus, 1 of 16 column gate outputs (i.e., resistor taps) is transferred to the DAC output.

Since the DAC step sizes are developed as a 256 resistor divider between ground and a reference voltage, the step voltages can be calculated as

$$\text{step size} = \frac{\text{reference voltage}}{256}$$

where the reference can be externally set between 1 and 2 V.

ARITHMETIC UNIT AND MEMORY

A block diagram of this subsystem is shown in Fig. 4. This subsystem consists of three major elements: a RAM storage array, a scaler, and an ALU.

Data within this structure are processed using 25 bit 2's complement arithmetic, although at certain locations larger or smaller words may be found.

Each of the elements making up this portion receives command or address information from the PROM. The storage array receives two 6 bit address fields, the scaler receives a 4 bit control field, and the ALU receives a 3 bit control field.

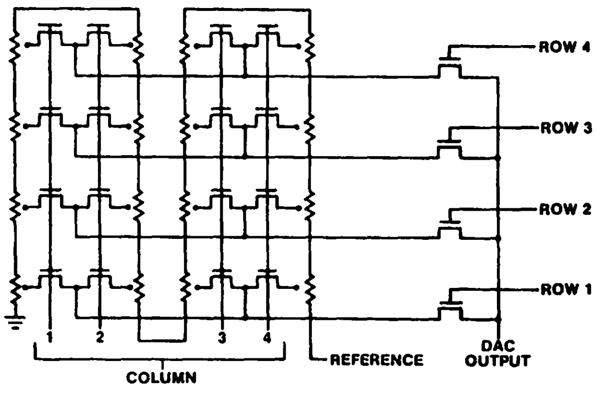

Fig. 3. Folded resistor array DAC.

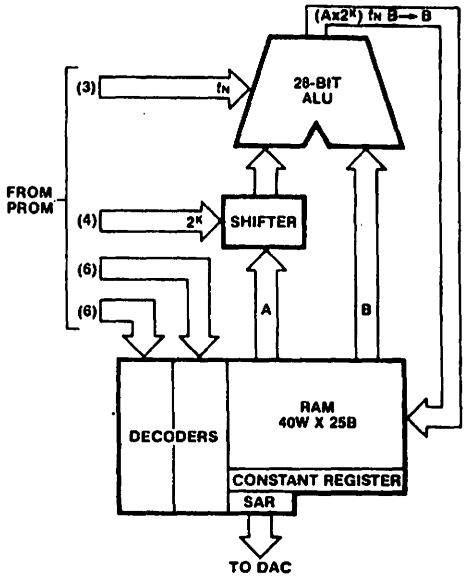

Fig. 4. Processor subsystem.

The Storage Array

The storage array consists of a random access two port static read/write memory organized as 40 words of 25 bits each. Each port is independently controlled by a 6 bit control field from the EPROM. The address range (64 possible locations) of this memory is extended to include an array of constants and an input/output register (SAR) which serves to link the arithmetic and analog/digital conversion sections. The two ports of the memory are designated *A* and *B*. The *A* port is read-only, and data read from it is passed through the scaler to one input of the ALU. The *B* port passes data to the second ALU input and receives the ALU results.

The constant array consists of 16 "pseudolocations" in the address field, and may be accessed only from the *A* port. Four bits of the address are directly translated to the high 4 bits of the data field, with the remaining data bits equivalent to zeroes.

The memory-mapped analog to digital SAR is 9 bits wide. As a memory location, the SAR occupies the 9 most significant bit positions of the 25 bit word and can be accessed via both A and/or B ports. Thus the user/programmer may think of the analog subsystem as a memory mapped input/output device. The remaining 16 bit positions, when read to the ALU as an operand, are all 1's, or $(2^{-16}-1)$, to correct for the A to D conversion offset.

Scaler

The scaler is an arithmetic barrel shifter located between the A port RAM data output and the ALU. Any value read from the A port can be shifted from 2 positions left to 13 positions right. Shifts to the left fill with zeroes at the right, shifts to the right fill with the sign bit at the left. These arithmetic shifts are equivalent to multiplication of the A port data value by a power of two, i.e., it is equivalent to a multiplication by 2^n, where $2 \geq n \geq -13$. The scaler is controlled by a 4 bit wide control field from the EPROM.

The ALU

The ALU calculates a 25 bit result from its A and B operands based on an operation code from the EPROM. The 25 bit result is written back into the B memory location at the end of the instruction cycle.

The ALU is 28 bits in precision and employs carry look-ahead techniques external to 4 bit ripple carry blocks. The final carry out is stable at about the same time as the final ripple carry is in each 4 bit block. The EPROM's bit field is decoded into eight control functions. In addition to basic operations such as add, subtract, and load, other operations include absolute values, limits, and complementing.

The ALU uses extended precision to allow calculation of the correct result even when receiving left-shifted operands from the scaler. If the computed result x exceeds the bounds $-1.0 \leq x \leq 1.0$, an overflow condition is indicated, and the result is replaced with the legal value closest to the desired result, i.e., with -1 if the computed value was negative, and $+1.0$ if the result was positive. This overflow saturation characteristic is useful for realizing certain nonlinear functions such as limiters, and is beneficial to the stability of filters.

Conditional Arithmetic Operations

In addition to the basic operations described, some ALU functions may execute conditionally. Certain codes in the analog/digital control field cause the arithmetic operation to be executed conditionally. For ADD and LOAD, the control field code selects a bit of the SAR, which is tested to determine how the instruction is to be executed. For a conditional subtract the carry from the previous result determines the arithmetic operation, and the selected bit of the SAR is altered by being set equal to the carry from the current instruction.

Conditional additions are used for shift and add multiplications of one variable by a second. (Multiplication of variables by constants is treated differently.) The multiplier is loaded

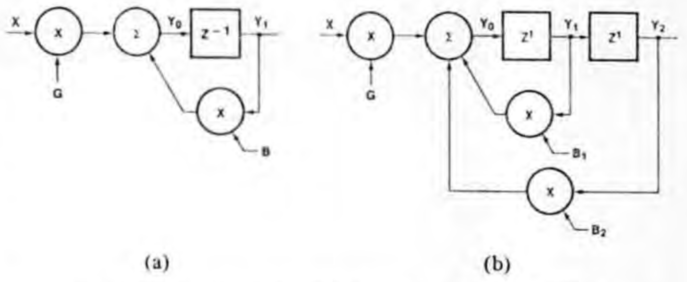

Fig. 5. (a) First-order and (b) second-order digital filters.

into the SAR, and the multiplicand is added conditionally to the partial product.

Conditional subtraction is used for division of one positive variable by another using a nonrestoring division algorithm. The divisor is conditionally subtracted from the dividend, and quotient bits are assembled in the SAR.

Multiplication by a Constant

Since many signal processing coefficients may be treated as fixed constants, their multiplication by a delay tap value can be reduced to a series of additions and subtractions. In general, multiplication by this method will be faster than shift and add techniques and more cost effective than parallel multipliers of equivalent precision [6]. This statement especially holds true for the architecture described in this paper, since the processor executes the operation.

$$A \times 2^N f_N B \rightarrow B$$

where f_N is the selected ALU operation.

The number of processor steps required to perform an operation of the type

$$Y = Y + C * X$$

is determined by the value of the constant C [7].

Processor Architecture as it Relates to Signal Processing Functions

Fig. 5 depicts the block structure of a typical first- and second-order digital filter. The described processor architecture is especially efficient at performing this operation. Unit delays are performed by transferring an A port memory location to a B port location each program pass, multiply/accumulation by the constant multiply algorithm described. The device, however, is not limited to digital filters.

By adjusting Fig. 5's coefficients to make the structure unstable, an oscillator can be developed. Other types of oscillators may be realized. An example is a sine wave approximation using a clipped triangle wave. This is generated by taking the absolute value of a sawtooth oscillator (generated by incrementing a register). The triangle can be scaled up, where the overflow saturation function clips the peaks. These oscillators may require subsequent filtering to eliminate unwanted harmonics. Any design should include consideration for these harmonics.

There are many other signal processing functions performable by the processor. Modulators can be realized using

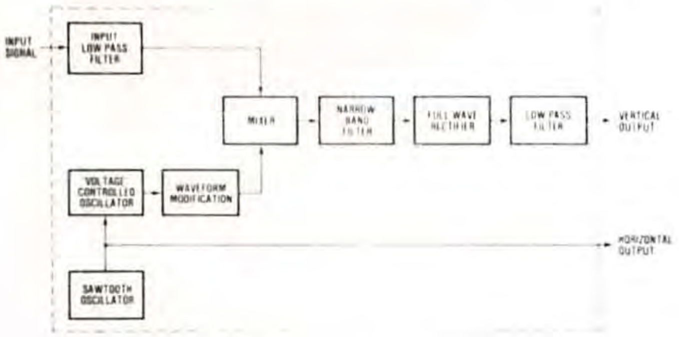

Fig. 6. Block diagram of a spectrum analyzer.

multiplication of a variable representing the carrier by a variable representing the modulating waveform. Automatic gain control (AGC) can be realized by dividing the signal by a level derived from the signal magnitude.

Correlation functions involve delays, products, and filtering. The delay achievable is limited by the number of RAM words provided, but two or more samples may be packed in a word to increase the achievable delay.

Logical operations can be performed using logical functions in the ALU's repertoir, or by conditional arithmetic, or by using threshold logic, i.e., summation combined with the saturation function. In some cases, several logical variables can be stored in one RAM word. Since A-D conversion is under EPROM control, logic inputs do not require full conversion. In general, the logic threshold value can be written into the SAR. A single conversion instruction can then be used to set one of the low order SAR bits to show whether the input falls above or below the threshold. By setting a low order bit the threshold value may not be significantly altered, so it may still be available for additional logic input operations. Thus, several inputs may be read if they all have the same logic threshold.

Hysteresis may be introduced to logic inputs by making the threshold value a function of the previous input value.

An application that uses a wide representation of processor capabilities is a spectrum analyzer (Fig. 6).

The purpose of this spectrum analyzer is to determine the long term spectral characteristics of a signal in the 200 Hz to 3.3 kHz frequency band. The approach used is to sweep the input signal through a high resolution (narrow-band) bandpass filter and observe the filter response as a function of the frequency sweep. First, the spectrum analyzer block diagram and parameters are determined. Then sampled data considerations are taken into account, and finally the signal processor code is developed.

Block Diagram Description

Ideally, a scanning spectrum analyzer could be implemented by simply scanning a tunable narrow-band bandpass filter across the input signal frequency to determine the signal energy at any frequency. Practically speaking, it is nearly impossible to design a complex tunable analog filter which can cover a 10 to 1 range of frequencies, especially near dc. Even digital implementation becomes very complex and hardware inefficient when tuning is required. It is therefore easier to realize the equivalent of the scanning filter by sweeping the signal past a fixed tuned narrow-band bandpass filter.

Additional Functions

The block diagram shows the basic functions of subsystems which must be implemented to operate the spectrum analyzer. In the digital implementation there must also be an input anti-aliasing filter, sample and hold, A-D converter, and the corresponding output D-A converter and reconstruction filter. The functions in Fig. 6 are implemented in the signal processor program.

CONCLUSION

An N-channel MOS signal processor for real time processing of analog signals has been described. A die size of 47 000 square mils establishes the economy of a multifaceted analog subsystem and microprocessor on a single substrate. The analog functions include four inputs, eight outputs with on-chip sample on hold, and 9 bit conversion system resolution; digital functions include 192 words by 24 bits of EPROM, and 40 words by 25 bits of two port RAM.

It has been shown that the device just described has relegated complex analog system problems, such as modems, tone detectors, or generators and process controllers to software solutions within a single chip. An example was given as a spectrum analyzer.

ACKNOWLEDGMENT

The authors wish to thank J. Huggins, B. Leung, B. Warren, H. Blume, and K. Mapes for their technical assistance in the development of this device. Special thanks to W. Li and C. F. Chan for their applications assistance.

REFERENCES

[1] M. E. Hoff and M. Townsend, "Single-chip NMOS microcomputer processes signals in real time," *Electronics*, p. 105, Mar. 1, 1979.
[2] —, "An analog input/output microprocessor for signal processing," in *ISSCC Dig. Tech. Papers*, Feb. 1979, p. 220.
[3] *8748 Users' Manual*, Intel Corporation, Santa Clara, CA, 9800270D.
[4] J. Huggins, M. E. Hoff, and B. Warren, "A single-chip NMOS PCM voice CODEC," in *ISSCC Dig. Tech. Papers*, Feb. 1978, p. 178.
[5] D. Senderowicz, D. Hodges, and P. Gray, "High performance NMOS operational amplifier," *IEEE J. Solid-State Circuits*, vol. SC-13, pp. 760-766, Dec. 1978.
[6] A. Peled, "On the hardware implementation of digital signal processors," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 76-86, Feb. 1976.
[7] A. D. Booth and K. H. U. Booth, *Automatic Digital Calculators*, 2nd ed. New York: Academic, 1956.
[8] M. E. Hoff, J. Huggins, Jr., and B. M. Warren, "An NMOS telephone codec for transmission and switching applications," *IEEE J. Solid-State Circuits*, vol. SC-14, pp. 47-53, Feb. 1979.

Matt Townsend (M'78) was born in Long Beach, CA, on September 4, 1947. He attended Arizona State University, Tempe, from 1969 to 1971.

From 1969 to 1971 he was also employed by Motorola Semiconductor, Phoenix, AZ. He joined the Applications Research Group at Intel Corporation, Santa Clara, CA, in 1971, where he became involved in a series of architectural studies and implementations of microcomputer systems. One such system, a 256 line PABX,