AT&T BELL LABORATORIES TECHNICAL JOURNAL
Vol. 63, No. 3, March 1984
Printed in U.S.A.

The DTWP: An LPC-Based Dynamic Time-Warping Processor for Isolated Word Recognition

By M. K. BROWN,* R. THORKILDSEN,* Y. H. OH,* and S. S. ALI*

(Manuscript received August 1, 1983)

Special-purpose hardware for calculating dynamic time-warp distances has been designed and tested utilizing technology. The Dynamic Time-Warp Processor (DTWP) performs all of the necessary arithmetic and decision-making operations for selecting a word from a given vocabulary based on log likelihood distance measurements. The speed limitation in previously designed hardware was due to programmed decision making (often referred to as combinatorics). The combinatorics have been implemented in hardware in such a way that the decisions are made in the time of several gate delays rather than the time of several program cycles. Thus, a dynamic time warp (DTW) is performed on typical 40-frame templates in less than one millisecond. The DTWP serves as a slave to a 16-bit microcomputer. It performs all of the computation and control necessary for pattern classification, and is now operating on the board level. The processor is now being implemented for very large-scale integration. All logic has been designed in 2.5 μm, Complementary Metal-Oxide Semiconductor polycells and has been simulated on the Metal-Oxide Semiconductor Timing Simulator (MOTIS). The timing simulations indicate that the DTW time of 1 ms implemented at the board level can also be met on the integrated circuit.

I. INTRODUCTION

The typical isolated speech recognition system attempts to recognize

* AT&T Bell Laboratories.

an unknown utterance by comparing the unknown with each of a number of previously stored reference templates. The recognition accuracy of speech is substantially increased when variations in the rate of speech production are taken into consideration. This is accomplished by dynamically warping the time axis of the reference utterance to the unknown utterance so that the minimum difference is found. In this way the majority of temporal variation in the speech is removed while the underlying spectral sequential structure is preserved. We have used the normalize-and-warp procedure of Myers, Rabiner, and Rosenberg,[1] with the global constraints proposed by Sakoe and Chiba.[2] The well-known Itakura algorithm has been implemented in a manner similar to that described by Ackenhusen and Rabiner.[3] The identification of the unknown is based on the minimum dynamic time-warp (DTW) distance obtained over the set of reference templates.

Let us assume that we are given a characterization of the isolated word, which consists of a set of N vectors of Linear Predictive Coding (LPC) coefficients. The test pattern, **T**, is represented as:

$$\mathbf{T} = \{T(1),\ T(2)\ \cdots\ T(N)\}, \tag{1}$$

where the vector $T(i)$ is a spectral (LPC) representation of the ith frame of the test word. In our system a set of nine autocorrelations constitutes the vector from which an eighth-order LPC model is derived. The duration of the test utterance is N frames, where each frame represents 45 mS of speech, and adjacent frames are spaced 15 mS apart.

For a given vocabulary of V words, the reference vector, $\mathbf{R}_v$, is represented as:

$$\mathbf{R}_v = \{R(1),\ R(2)\ \ldots,\ R(M_v)\} \tag{2}$$

where each vector, $R(j)$, is again a spectral representation of the corresponding jth frame within the reference utterance, and M_v is the number of frames in the vth reference.

To optimally align the time scale of the reference pattern (the dependent m index) to the test pattern (the independent n index), we must solve for a warping path function of the form:

$$m = w(n) \tag{3}$$

and thereby seek to minimize the total distance

$$D = \sum_{n=1}^{N} d[T(n),\ R(w(n))] \tag{4}$$

over all possible paths, $w(n)$, within the constraints, where $d(T(n), R(m))$ is the local distance between test frame n and reference frame $m = w(n)$. This operation must be performed for each reference vector,

$R(j)$, in the vocabulary. The test pattern is classified as belonging to the class for which the smallest accumulated DTW distance, D, is obtained. In addition to the standard DTW algorithm, we have employed linear time normalization of the normalize-and-warp procedure described by Myers et al.,[1] thus allowing the widest range of time alignment paths to be considered. This procedure linearly normalizes the test and reference utterances to a fixed length (in this case 40 frames) before the DTW is performed. This prenormalization greatly simplifies the processor design since the control logic can be fixed rather than have to respond to variables associated with warps of varying length.

Dynamic time warping has been implemented based on the Itakura[4] constraints as follows:

$$\omega(1) = 1 \tag{5}$$

$$\omega(40) = 40 \tag{6}$$

$$g(n, m) = \begin{cases} 1, & \text{if } \omega(n) \neq \omega(n-1) \\ \infty, & \text{if } \omega(n) = \omega(n-1), \end{cases} \tag{7}$$

where accumulated distance is given by

$$\begin{aligned} D(n, m) &= d(T(n), R(m)) + \\ &\min[D(n-1, m)\, g(n-1, m), \\ &D(n-1, m-1), \\ &D(n-1), m-2)] \end{aligned} \tag{8}$$

and

$$D(1, 1) = d(T(1), R(1)). \tag{9}$$

Constraints (5) and (6) require the endpoints of the test and reference to match. Time alignment is performed between the endpoints constrained locally by (7). Basically, constraint (7) and eq. (8) allow the reference utterance to be time compressed by skipping one frame for each test frame. Time stretching is accomplished by duplicating a reference frame. This constraint limits the number of times a reference frame may be repeated to one, as shown in Fig. 1a. The path marked "x" is not allowed.

The terminal endpoint conditions must be satisfied by applying global constraints, as illustrated in Fig. 1b. The parallelogram constraint shown forms the basic constraint requirement such that local constraint (7) is satisfied from either endpoint. Sakoe and Chiba[2] have found that a further global constraint limiting the deviation of the solution path from the diagonal to ±R substantially reduces the

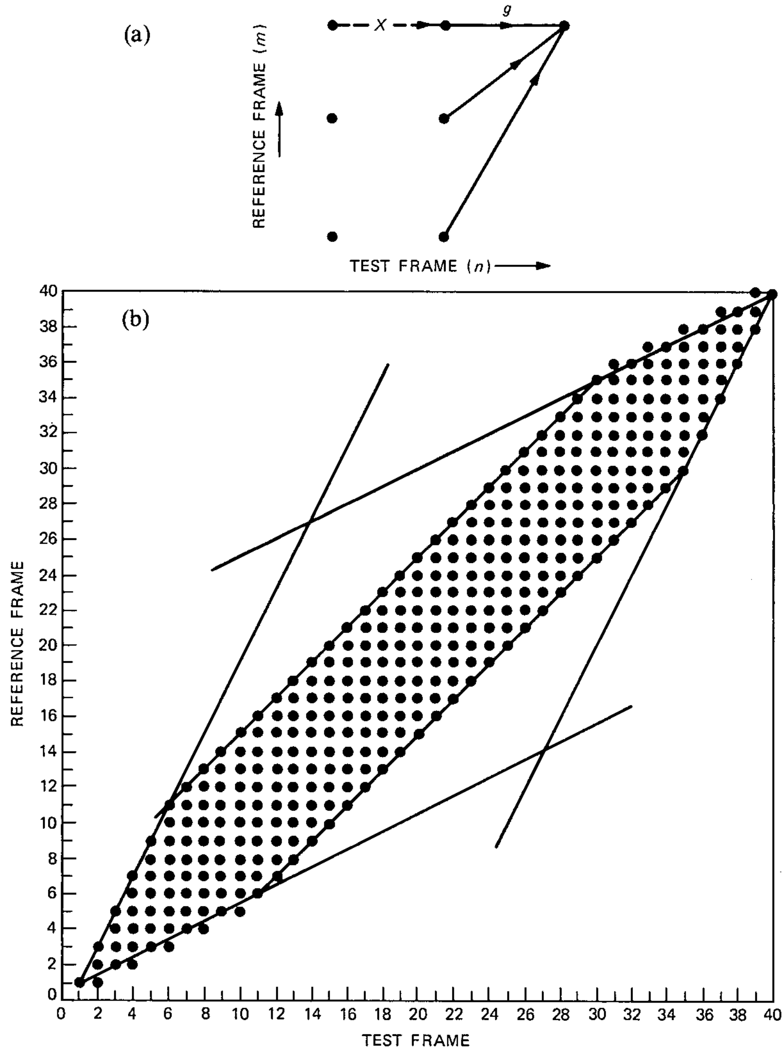

Fig. 1—Dynamic time warp for (a) local constraints, and (b) global constraints.

number of distance calculations necessary without suffering a significant loss in accuracy. In this implementation R is five.

Comparison of a test frame to a reference frame requires a measure of closeness in some sense. Several distance measures have been investigated and used for utterance comparison purposes. A method developed by Itakura[4] has been found to yield high recognition accuracy and costs relatively little in computation. This distance function, often referred to as the log likelihood ratio (LLR), yields numerical values that are indicative of the spectral energy difference between the two frames of speech. The form of the function is

$$d(x, y) = \ln[(a_r \mathbf{V}_t a_r')/(a_t \mathbf{V}_t a_t')], \tag{10}$$

where t refers to a test frame and r refers to a reference frame, a is a $(1 \times p + 1)$ vector consisting of a 1 followed by p LPC coefficients of a pth-order LPC model of the speech and $\mathbf{V}$ is the $(p + 1 \times p + 1)$ autocorrelation matrix of the test frame. By appropriately computing a set of reference coefficients and test coefficients, $d(x, y)$ can be obtained as the result of a $(p + 1)$-point dot product as described by Itakura.[4]

The DTW algorithm, which has traditionally been performed by a general-purpose processor or microprocessor, requires 360 distance calculations, 630 two-way comparisons, and 359 additions for eighth-order LPC. We have developed a finite-state machine specifically for DTW calculations which performs these calculations in 902.5 microseconds. The processor currently operates with a 4-MHz clock, but it is estimated from design analysis that the clock rate could be increased to 6 MHz for the board-level implementation.

In the following section we will discuss the architecture of the DTWP and explain its operation. Modules of particular interest are the combinatoric logic and the logarithm calculation logic, which will be described in Sections III and IV. Some preliminary specifications on the VLSI implementation will be discussed in Section V. Options of system integration are covered in Section VI with an example of a multiprocessor system implementation.

II. PROCESSOR ARCHITECTURE

The DTWP architecture is specifically designed to implement a particular DTW algorithm used in the speech recognizer described by Ackenhusen and Rabiner.[3] Both the test utterance and reference template contain 40 frames each of LPC features. The Itakura constraints are applied so that expansion and compression ratios of the reference time axis with respect to the test time axis are limited to two to one or one half to one, respectively. Furthermore, the additional constraints described by Sakoe and Chiba[2] are applied (with $R = 5$) so that the number of distance calculations per column is limited to 11. In part, the high effective processing speed obtained with the DTWP is due to this specialization of the design. Changes in the number of frames to be warped, the constraints, and the Sakoe constraint (R) will require redesign of the DTWP.

The DTWP is interfaced to the control microprocessor via the data and address buses and four control lines from an I/O port. The microprocessor can load LPC values into both the test memory and reference memory. The test and reference templates are stored as 12-

bit values, 9 values per frame. Thus, each template requires 4320 bits of memory. The number of warps to be performed by the DTWP depends on the number of reference patterns stored in reference memory. An 8-bit register is provided in the DTWP for telling the processor the number of patterns to warp. After this value is loaded, the DTWGO signal is asserted by the microprocessor and the DTWP takes over control of its memories. These memories are removed from the microprocessor bus and split so that test and reference memories are individually accessible. In this way, data are obtained from the two memories in parallel at the rate of 24 bits per clock cycle or 96 million bits per second.

Warps are performed every 902.5 microseconds. The DTWP may be interrupted between warps so that the microprocessor can obtain the accumulated warp distances for each reference. The DTWP keeps track of the best distance and the corresponding reference index, which can be read back by the microprocessor between warps. If the DTWP is not interrupted between warps, it can perform 256 warps in 231 milliseconds.

An LPC distance is computed by multiplying the nine values of a test frame with the nine values of a reference frame, accumulating the sum of the products, and taking the logarithm (in this case a base two logarithm). These operations are performed by a 12 × 12 Wallace tree multiplier with a 24-bit accumulator in a pipelined fashion. Referring to Fig. 2, which shows a block diagram of the distance calculator, the sequence of operation is:

1. Apply addresses to the test and reference memories
2. Clock data into the multiplier input registers
3. Clock the product into the accumulator
4. Repeat steps 1, 2, and 3 eight times
5. Clock the logarithm result into the combinatoric logic.

Thus, it takes 13 clock cycles to obtain the first distance, and 10 clock cycles for each succeeding distance. The first three operations require nine clock cycles to compute the nine-point dot product. Operation (5) is performed on the tenth clock cycle.

A diagram of a DTW calculation is shown in Fig. 1. Each point on the diagram indicates a distance calculation consisting of a 9-point dot product and its logarithm, which is the LPC distance between one test frame and one reference frame. After the distance is computed, all calculations in the processor are performed in 16-bit unsigned arithmetic. An accumulated distance is calculated at each point, which consists of the local distance (or dot product) and the minimum of three possible predecessor distances, as shown in Fig. 1a. Thus, if each point is calculated sequentially (from bottom to top and from left to right on the diagram), only the previous column of points needs to be

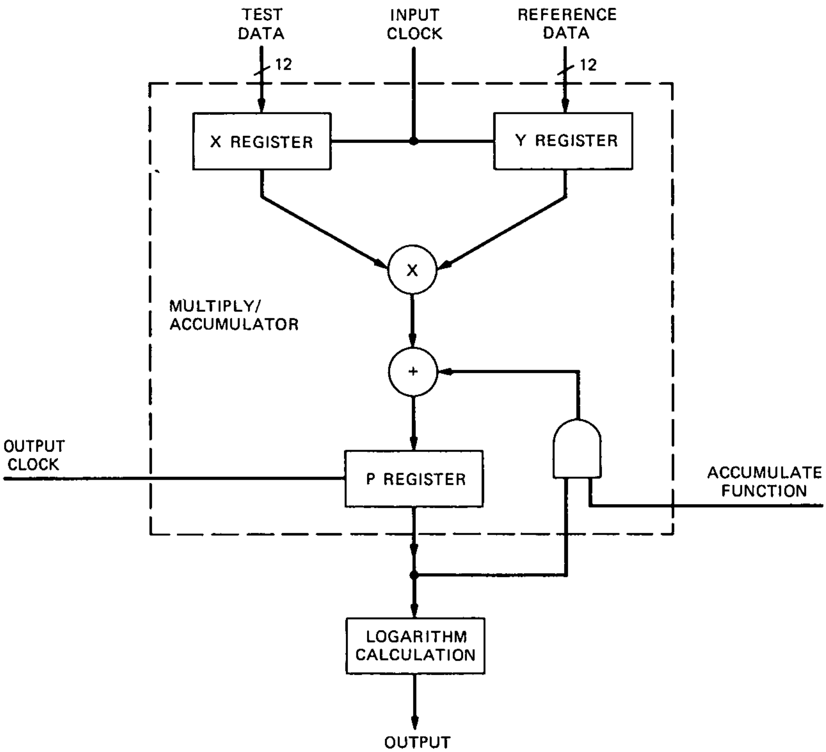

Fig. 2—Distance calculator.

available at any one time.

This is accomplished in the DTWP by a 14-stage shift register, as shown in the block diagram of Fig. 3. The boundary conditions of the warp are established by inserting large values, which we will call infinities, into the shift register at the top and bottom of each column of distances. These infinities will never be chosen as a minimum by the minimum selector for the accumulated distance calculation. Thus, when columns of 11 distances are processed (the middle portion of the warping function), two infinities must be inserted into the shift register after each column to separate one column from another. The number of stages in the shift register (14) comes from an analysis of the central portion of the warp. The 11 distances and two infinities occupy 13 stages. The 14th stage is necessary to position the accumulated distances of the previous column at the correct inputs of the three-way minimum selector. In this way, accumulated distances of the preceding column and the local distances currently being calculated are properly associated.

The number of distances in a column is fewer than 11 and increases

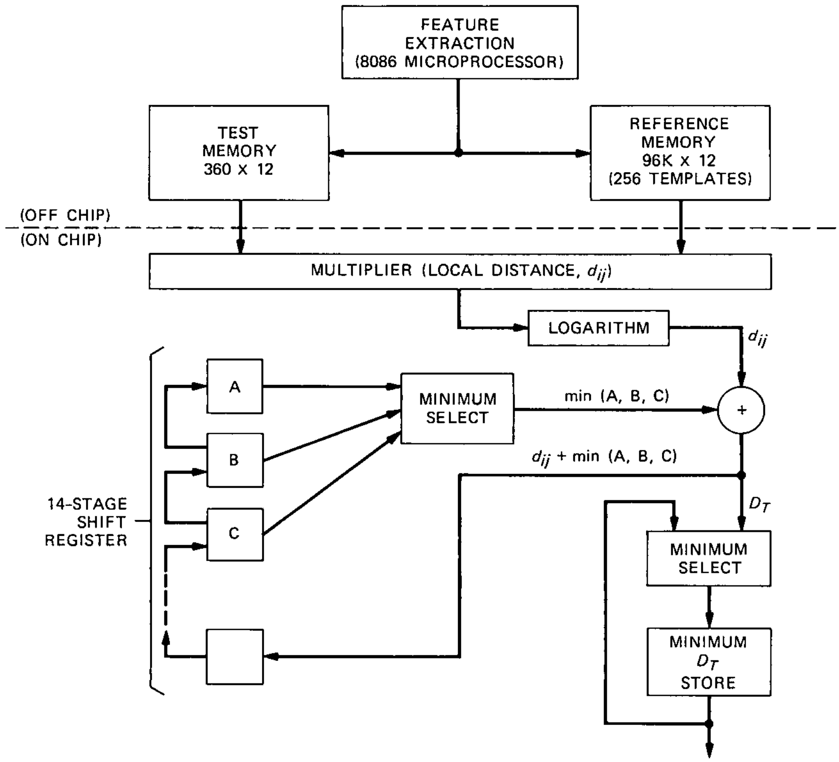

Fig. 3—Architecture of the dynamic time-warping processor.

at the beginning of the warp and decreases at the end of the warp. To properly align the columns of distances an appropriate number of infinities must be inserted into the shift register after each column. The number of distances and the number of infinities needed for each column are stored in Read-Only Memory (ROM). The sequence used in the DTWP is shown in Table I.

These values are critical to proper operation of the circuitry and cannot be usefully modified without corresponding changes in the surrounding hardware. Since one processor clock cycle is required to insert each infinity into the shift register, distance calculations are performed in parallel with the shifting operation. Normally, the number of infinities to be inserted is fewer than 10, so the shifting is normally completed before the next distance calculation is finished. There are four columns of distances, however, that require 10 or more infinity shifts. In these cases the distance calculator enters a wait state after the 10th clock cycle until the shifts are completed. During a warp, six wait-state clock cycles are used.

The last three stages of the 14-stage shift register deliver 16-bit

Table I—Sequence of distances and infinities for DTWP

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

values to the three inputs of a three-way comparator shown in the block diagram of Fig. 3. The first of these three comparator inputs corresponds to a horizontal path segment, as shown in Fig. 1a. This comparator input can be disabled by a control bit stored in an auxiliary shift register, which indicates if a horizontal path was taken when the previous column of distances was calculated. This auxiliary shift register is 12 stages long and is clocked at the same time as the 14-stage shift register. If the horizontal path input is not disabled, and it is found that this input gives the lowest accumulated distance, then the path disable bit is set in the first stage of the auxiliary shift register in preparation for the following column of distances. Further details on the three-way comparator are given in the next section.

The output of the three-way minimum selector is added to the local distance from the distance calculator and the result is inserted into the first stage of the shift register between columns, until all 360 distances have been calculated. At this point a 4-clock cycle sequence is entered, which compares the current warp distance with the best previously stored warp distance and updates the stored value if necessary. Also, during this 4-cycle sequence, internal registers are cleared, the reference index is updated, and the control logic is initialized for the next warp. Thus, a warp is performed in 3600 clock cycles for distance calculations, plus 6 clock cycles for wait states, and 4 clock cycles for classification and reinitialization for a total of 3610 clock cycles.

III. COMBINATORIC LOGIC

The three-way minimum selector performs the equivalent of three memory reads, two comparisons, and two memory writes in slightly more than 50 ns in Transistor-Transistor Logic (TTL). MOTIS simulation indicates that the three-way comparison can be done in about 75 ns in Complementary Metal-Oxide Semiconductor (CMOS) logic. This very fast selection logic, coupled with the shift register architecture, allows the combinatorics of the DTW to be performed as fast as distances can be calculated. A hardware adder combines the minimum values selected with the LPC distance from the distance calculator and delivers the result to the first stage of the shift register. This combinatoric procedure occurs within one 250-ns clock cycle.

An expanded block diagram of the three-way minimum selector is shown in Fig. 4. The three-way selector is implemented as a pair of two-way selectors. The first selector determines the minimum of inputs A and B, and the second selector chooses between the A or B output of the first selector and input C. If input A is selected, the path disable bit is set in the auxiliary shift register. This bit is used to disable input C of the three-way minimum selector so that a horizontal path (in the sense of Fig. 1) can be inhibited.

The three-way selector is designed so that it will favor input A or B in the event that all three inputs are presented with equal values. This characteristic is considered to be the best since it assumes no expansion or compression of the test utterance with respect to the reference template and does not constrain the horizontal path unnecessarily.

The adder is a standard logic circuit. It performs a 16-bit unsigned addition in about 35 ns. The output of this adder is made available to the shift register at the next processor clock time.

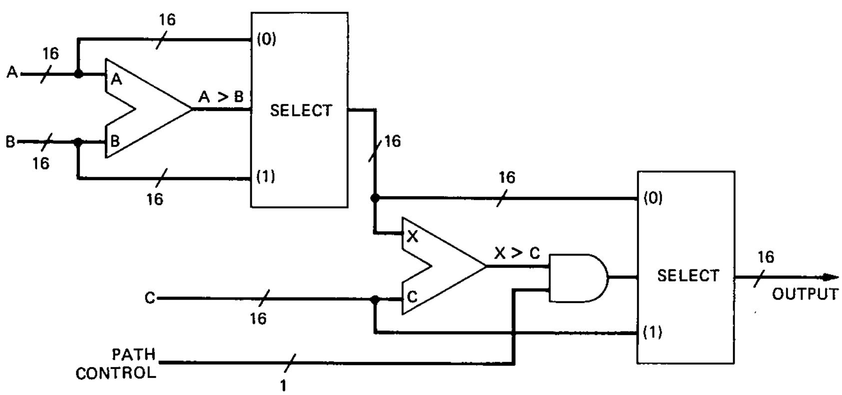

Fig. 4—Minimum selector.

IV. LOGARITHM FUNCTION LOGIC

It is well known that the logarithm of a number can be estimated from a geometric series. Since we are applying the logarithm as a multiplicative function (and performing a relative comparison of the results), the base of the log is unimportant. The logarithm function circuitry makes use of a first-order approximation of the log base two in the range of one to two as follows:

$$\log(x) = x - 1 \quad \text{for} \quad 1 < x < 2. \tag{11}$$

The maximum amount of error between the straight line approximation and the base two logarithm function is less than 0.09 in magnitude. Relative error approaches about 30 percent near $x = 0$; however, this has little impact on the relative comparison of accumulated distances. A plot of logarithm calculation error versus x is shown in Fig. 5a.

The logarithm for values of x greater than two can be approximated by dividing x by the largest value of 2^n that results in a quotient that is greater than or equal to one. Thus, the logarithm base two may be approximated with a piecewise linear function as:

$$\log(x) = n + (x/2^n) - 1 \qquad \begin{array}{c} \text{for } x \geqslant 1 \\ \text{and} \\ 1 \leqslant (x/2^n) \leqslant 2. \end{array} \tag{12}$$

This function is implemented with a priority encoder and multiplexor as shown in Fig. 5b. The priority encoder determines the value of n and the multiplexor performs the division by 2^n. This logic functions accurately only for values of x greater than or equal to 1.0. The TTL design requires four dip packages and yields a propagation delay of about 25 ns. The implementation of Very Large-Scale Integration (VLSI) has been simulated with MOTIS and requires 35 ns for the equivalent operation using 2.5-μm CMOS. This is a substantial improvement over an earlier design that used an Erasable Programmable Read-Only Memory (EPROM) with an access time of about 350 ns. For VLSI design, this approach is much more desirable since it only requires about 200 gates as compared to about 25,000 bits of ROM.

V. CHIP DESCRIPTION

The DTWP chip will be packaged in a 68-pin chip carrier. The allocation of pins is given in Table II. The 68-pin chip carrier is a standard of AT&T Technologies, Inc. The six additional pins will be used for testing purposes.

The data pins will be multiplexed so that the DTWP can access the reference and test memories while it is calculating a warp path, and

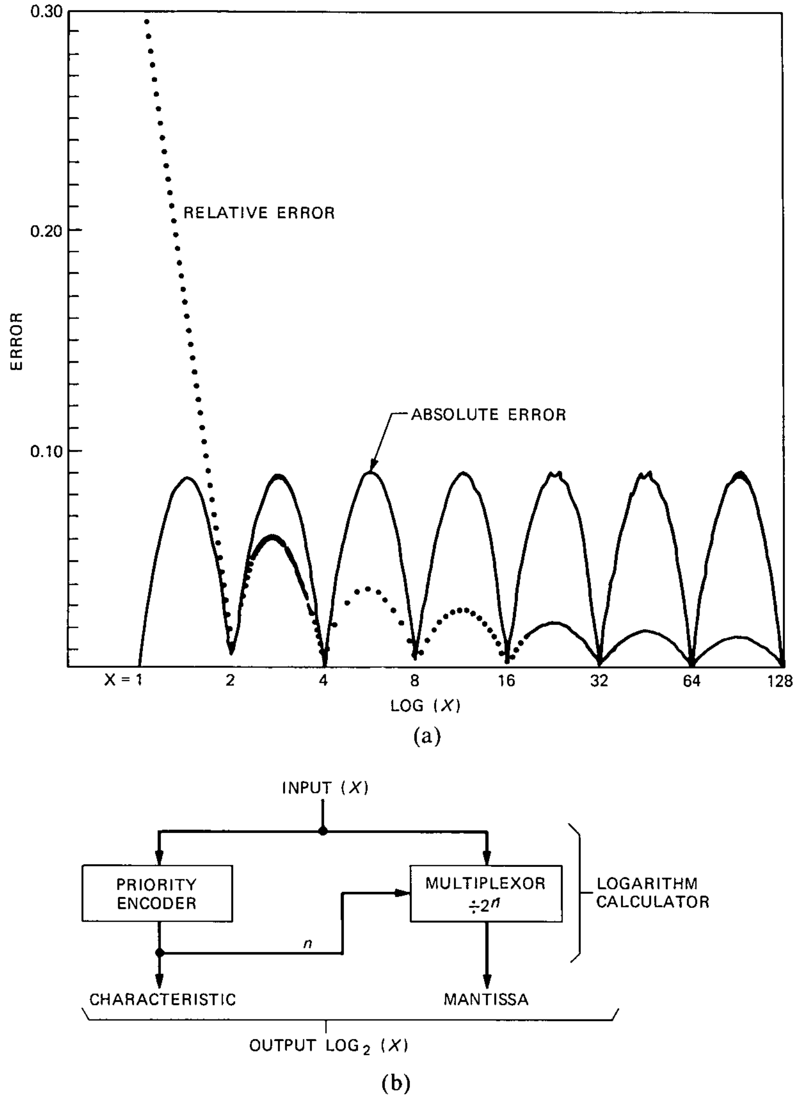

Fig. 5—(a) Logarithm calculation error. (b) Logarithm calculator.

the microprocessor can access internal DTWP registers while the DTWP is idle. Access to the internal DTWP registers is controlled by three Read/Write Control lines. The Interrupt Enable and Interrupt Flag will allow the microprocessor to halt the DTWP temporarily while the internal registers are being read. DTWP Busy is provided to aid in interfacing the test and reference memories, which must appear to be dual-port memories to the DTWP. The Hard Reset allows the micro-

Table II—DTWP chip pin allocation

Reference addressing	17
Reference data and DTWP low byte for distance, index, number of warps	12
Test addressing	9
Test data and DTWP high byte	12
Read/write control:	
*Load register	1
†DTWP go	1
*Read distance	1
*Read best distance	1
*Read index	1
†Interrupt enable	1
†Interrupt flag	1
†DTWP busy	1
*Hard reset	1
Clock	1
Power	1
Ground	2
Total pins currently (others used for testing)	63

* Address decoded.
† I/O control.

processor to reset internal registers and clear all pending DTWP operations.

As shown in Table II, several registers are available to the microprocessor. The Distance Register will allow the microprocessor to read the accumulated warp distance of each warp, if desired, under control of the Interrupt function. This is accomplished by enabling the DTWP interrupt via Interrupt Enable and waiting for the Interrupt Flag to be asserted. At this time the DTWP is idle and the microprocessor can read the internal distance register, the best distance found so far, and the index of the best reference template. When the read is complete, the Interrupt Enable is lowered by the microprocessor, the DTWP then clears the Interrupt Flag and proceeds with the next warp. If the distance for the next warp is required by the microprocessor, then it must enable the interrupt within 900 microseconds in order to catch the next interrupt.

The multiplier consists of three stages. The first latches the multiplicand and multiplier and generates six partial products in parallel using a two bit at a time algorithm.[5] The partial products are added together in the last two stages, which are a bit slice adder and a full-carry look-ahead adder. Since the output of the multiplier is latched and then gated back to the top of the bit slice adder, accumulation of products is done along with partial product addition when required. This operation is selected or inhibited by the gate.

As we mentioned previously, the multiplier accumulator performs a 12- × 12-bit multiply and 24-bit accumulate in less than one clock cycle (250 ns). Thus, each clock pulse latches new data into the first

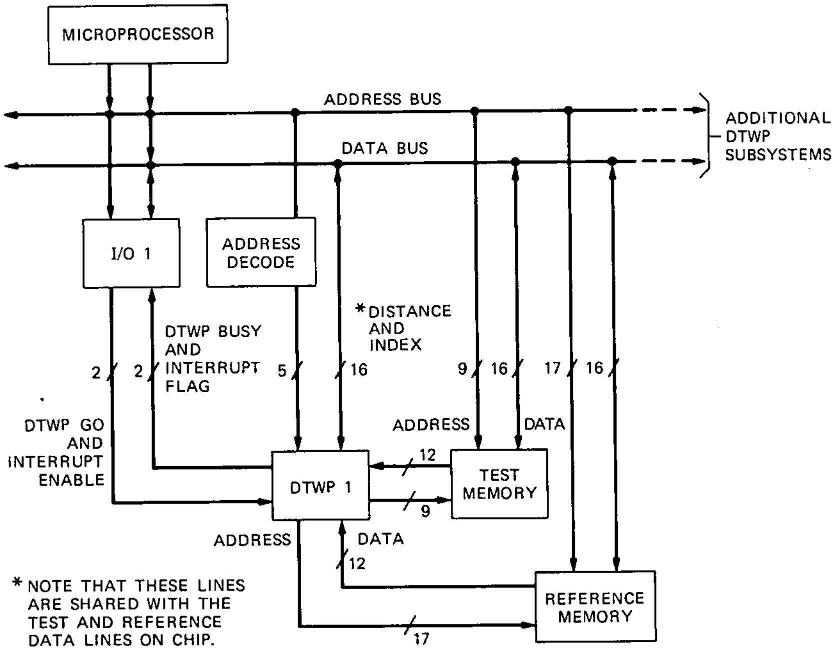

Fig. 6—DTWP interfacing.

stage and simultaneously latches the previous result at the multiplier output in a pipeline manner.

Current estimates indicate that the chip will contain about 6,600 gates, or 40,000 transistors. Of these, approximately 11,000 transistors comprise the multiplier-accumulator. This should result in a chip size of about 9 mm by 9 mm when implemented by polycell design techniques in 2.5-μm CMOS technology.

VI. SYSTEM IMPLEMENTATION

A block diagram of a typical DTWP configuration is shown in Fig. 6. Both DTWP test and reference template memories are accessible to the microprocessor via the bus by appropriate addressing. The number of DTWP's that can be handled by the microprocessor depends on the amount of memory address space for reference memory since each reference memory requires 192K bytes of address space. The 8086 microprocessor, for example, can handle only about four DTWPs. If the reference memory is stored in ROM, then this addressing limitation is removed.

It should be noted that the test and reference memories here appear to be dual port. This is accomplished by use of tri-state buffers that are controlled by the DTWP Busy signal. When this signal indicates a warp is in progress the memories are floated off the microprocessor address/data/control buses and connected to the DTWP address/data bus. This allows the microprocessor to have access to the system bus so that it may execute another program during DTWs.

Each DTWP is controlled by the microprocessor through either I/O control signals or memory address lines. The memory address must be decoded by outside logic to provide the desired reset, load, or read command. The Interrupt Flag and DTWP Busy can be handled by the microprocessor through polling or as a true interrupt.

The DTWP Reset should be triggered on power up and should also be able to be triggered by software. This clears all DTWP registers and resets internal control logic to standby. In this mode the test and reference memories are available to the microprocessor.

The reference memories are loaded by the microprocessor in the training mode with the appropriate number of reference templates (up to 512 templates in this case). At this point the recognition mode is entered. The two test memories are loaded by the microprocessor with an unknown test utterance. Each DTWP must have its own copy of the test word. The Load Register line is enabled and a microprocessor write cycle is performed to enter into each DTWP the number of reference templates to compare. This can be performed simultaneously for all DTWPs if the same number is to be written to each DTWP.

When the DTWP Go is triggered for each DTWP, the DTWP Busy flag for each processor will be asserted. Each DTWP will then sequentially compare the test word against each reference in its reference memory until the designated number of references have been checked. Optionally, within 900 μs after the DTWP Go, the user may assert the Interrupt Enable and thus cause the DTWP to enter an idle state after completing the first warp. At this point the microprocessor may read the distance score for the first comparison. Clearing the Interrupt Enable causes the DTWP to resume operation. If the Interrupt Enable is asserted again within 900 μs, the DTWP will stop after the next warp. In this way the distances for each comparison may be obtained by the user. When the specified number of warps is complete, the DTWP Busy flag will be cleared and the DTWP enters the idle state. The microprocessor can read the Best Distance and Index by sending the appropriate Read Enable signal to the DTWP and performing a read cycle. Each DTWP must be handled separately by the microprocessor during a read cycle so that data from the two DTWPs are not mixed together. In this configuration, 512 warps can be performed in about 231 ms if intermediate distance readings are not taken.

VII. SUMMARY

The dynamic time-warping processor described here is a key element in the single-board isolated word recognizer. The DTWP chip will perform dynamic time warps at about 50 times the speed of the currently used microprocessor-based hardware. A single DTW requires 902.5 μs (3610 clock cycles at 4 MHz). The number of warps performed is controllable by the microprocessor and may currently be set for up to 256 warps. The DTWP also performs a single nearest neighbor rule classification. The index of the best reference candidate, warp distance to the best reference checked, and warp distance to the current reference are available to the control microprocessor after each warp by raising an Interrupt Enable Input to the DTWP. This will cause the DTWP to interrupt its processing at the end of the current warp and wait for the control microprocessor to read data from the DTWP registers. If the DTWP is not interrupted between warps, it can perform 256 warps in 231 ms (a rate of about 1108 warps per second).

The architecture and logic design have been tested with a TTL board-level implementation. A VLSI multiplier chip is the most complicated device used. Except for ROM memories, all of the other logic is of Medium-Scale Integration (MSI) complexity requiring about 100 packages. A 96K by 12-bit reference memory and a 1K by 12-bit test memory are used for template storage. The processor consumes about 1.8 *INTERPAC** 13-inch wirewrap boards, including reference memories. The DTWP chip, which will replace these 100 packages, is currently projected to contain about 6600 gates or 40,000 transistors.

REFERENCES

1. C. S. Myers, L. R. Rabiner, and A. E. Rosenberg, "Performance Trade-offs in Dynamic Time Warping Algorithms for Isolated Word Recognition," IEEE Trans. on Acoustics, Speech, and Signal Processing, *ASSP-28*, No. 6 (December 1980), pp. 622–33.
2. H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," IEEE Trans. on Acoustics, Speech, and Signal Processing, *ASSP-26*, No. 1 (February 1978), pp. 43–9.
3. J. G. Ackenhusen and L. R. Rabiner, "Microprocessor Implementation of an LPC-Based Isolated Word Recognizer," IEEE Conf. Acoustics, Speech, and Signal Processing (March 1980), pp. 746–9.
4. F. Itakura, "Minimum Prediction Residual Principle Applied to Speech Recognition," IEEE Trans. Acoustics, Speech, and Signal Processing, *ASSP-23*, No. 1 (February 1975), pp. 67–72.
5. R. M. M. Oberman, *Digital Circuits for Binary Arithmetic*, New York: John Wiley and Sons, 1979, pp. 151–62.

AUTHORS

Syed S. Ali, M.S. (Physics), 1973, University of Punjab, Pakistan; AT&T Bell Laboratories, 1980—. Since joining AT&T Bell Laboratories Mr. Ali has been involved in the development of software and hardware for speech recognition subsystems.

---

* Trademark of AT&T Technologies, Inc.

Michael K. Brown, B.S.E.E., 1973, M.S., 1977, and Ph.D., 1981 (Electrical Engineering), all from the University of Michigan, Ann Arbor; Burrough's Corporation, 1973–1980; AT&T Bell Laboratories, 1980—. From 1973 to 1976 Mr. Brown was employed with the Burrough's Corporation and was involved in the development of ink jet printer systems. From 1976 to 1980 he continued his work with Burrough's Corporation as a consultant while pursuing his Ph.D. Degree at the University of Michigan. His thesis was in the area of image processing and pattern recognition. In 1980 he joined the Speech Processing Group at AT&T Bell Laboratories, Murray Hill, New Jersey, where he was involved in research in speech recognition and synthesis techniques. Since 1983, he has been with the Robotics Principles Research Department.

Young Hwan Oh, B.S., 1971, M.S., 1974, Ph.D., 1978 (Electrical Engineering), University of New Mexico; M.B.A. Program, 1971–1972; GTE Automatic Electric Laboratories, 1978–1981; AT&T Bell Laboratories, 1981—. While at GTE Automatic Electric Laboratories, he was involved in hardware design, debugging, and testing for an I/O module for GTD5-EAX, Class 5 Digital End Office (DEO) Switching System. Also, he was a responsible engineer for converting the analog DTMF receiver to the digital DTMF receiver for No. 5 application. At AT&T Bell Laboratories, Murray Hill, he has been with the Speech Processing Group, where he has been involved in speech synthesis and recognition projects. His dissertation was in the area of digital filtering and performance analysis. Member, IEEE, Tau Beta Pi.

Reed Thorkildsen, B.S. (Physics), 1976, East Tennessee State University; Ph.D. (Physics), 1981, University of Virginia; AT&T Bell Laboratories, 1981—. Mr. Thorkildsen is engaged in development of voice recognition integrated circuits and subsystems. Member, Phi Kappa Phi, Sigma Xi.