Communications
Research
Centre

REAL-TIME IMPLEMENTATION OF THE CHA ALGORITHM USING AN ARRAY PROCESSOR

by

Eloi Bossé

This work was sponsored by Department of National Defence, Research and Development Branch under Project No. 011LA13.

CRC REPORT NO. 1422
OTTAWA, FEBRUARY 1988

G[illegible]anada
D[illegible]mmunications

Gouvernement du Canada
Ministère des Communications

Canada

COMMUNICATIONS RESEARCH CENTRE

DEPARTMENT OF COMMUNICATIONS
CANADA

REAL-TIME IMPLEMENTATION OF THE CHA ALGORITHM USING AN ARRAY PROCESSOR

by

Eloi Bossé

(Radar and Communications Technology Branch)

CRC REPORT NO. 1422

February 1988
OTTAWA

This work was sponsored by the Department of National Defence,
Research and Development Branch under Project No.011LA13.

TK
5102.5
C673e
#1422
c.b

TABLE OF CONTENTS

ABSTRACT

Presented in this report, is a study of the implementation of the CHA algorithm on the AP-120B array processor. The study had two main objectives: (1) to investigate the AP-120B's real-time capabilities and (2) to determine the digital noise generated by the AP-120B due to round-off errors. The execution time that was sought for the array processor was 20 msec per data sample. It was found that a processing time of 13 msec could be achieved if no communications were required between the AP-120B and its host computer. This value quickly grew to 85 msec if interaction with the host computer was called for. To lessen interaction with the host, a GPIOP (General-purpose Programmable Input Output processor) was used to interact between the AP and external devices. Also, it was found that the digital noise generated by the AP-120B was negligible when compared to typical radar signal noise.

1 INTRODUCTION

The Correlation Height Analysis (CHA) technique has been developed to improve the performance of tracking radars against sea-skimming missile targets. It receives video data from the radar and uses signal processing to extract the target altitude. The CHA algorithm consists of 370 lines of Fortran code and requires a processing time of 6 sec per data point, when implemented on a PDP-11/34 computer. Real-time operation of the CHA algorithm requires that this processing time be reduced to 20 msec or less.

A number of studies and analyses have been carried on the implementation of the CHA algorithm as a real-time processor. Canadian Marconi [1] has studied a number of possible techniques, including: the Intel 8087 numeric processor in combination with the 8086 16-bit microprocessor, a technique using memory based precomputed values, and a fixed-point implementation on array processors model AP-400 (Analogics) and FPS-100 (Floating Point Systems). The AMD 29500 signal processor family and other bit slice approaches have been considered at C.R.C for implementing the CHA. The 29500 family represents an evolving set of LSI devices for high performance signal processors. Real-time speed requirements were met but it was necessary to include MSI devices to ensure that a complete and coherent system solution was provided. In the latter case, time, effort and development cost were judged to be too great.

This document reports the results of the real-time implementation of the CHA algorithm on a Floating Point System AP-120B array processor. A brief description is given of the physical phenomena under investigation and the mathematical algorithm used to model them. This is followed by a description of the AP-120B and a description of a GPIOP interacting with the AP-120B and an external device. An evaluation of the implemented CHA processing time is then presented.

2 CHA ALGORITHM

Accurate tracking of low-elevation targets is difficult when a target is within about a beamwidth of the horizon. Difficulty is caused by a portion of the electromagnetic (E-M) energy which is reflected from the surface of the sea or land and received by the tracking radar. This results in multipath interference between the direct and indirect E-M waves.

The multipath interference can be from either specular reflections or diffuse reflections. Specular reflection refers to the "coherent" part of the reflected signal where the phase varies with the position of the radar or the target. Diffuse reflection refers to the "incoherent" part of the reflected signal which varies randomly. These reflections modify both the amplitude and phase of the radar signals used for aligning the aperture of the radar with the incoming E-M. They cause the tracking radar to indicate an erroneous target position. The reflected signal appears as a target in the image position as in Fig. 1.

The CHA is a high resolution algorithm which takes advantage of the specular multipath information contained in the composite signal detected by the radar. Two main problems affect the composite signal: surface clutter or backscatter, and multipath or forward scatter of target energy.

Litva and Rook [2] developed the CHA algorithm at CRC based on a simplified transmission model. This model was appropriate for their experimental arrangement which used a cooperative beacon as the target in conjunction with a transponder that gave range information.

The advantage of this approach is the elimination of the necessity to search over numerous range cells for the target, leaving multipath as the only problem. This is the model considered in this report. When a radar is used in lieu of a beacon, a more complicated model must be developed [3].

2.1 The One-Way Propagation Model: (Cooperative Beacon)

The CHA technique models the interference pattern which is a function of the angular separation between the two primary signals arriving at the receiver. These signals are: a) the direct signal, b) the coherent component of the indirect signal.

The height of the target (H_R) (Fig. 2) is derived by cross-correlating theoretical data, which are simulated over a height window at equidistant increments (HGHTV), with the measured data. The correlator finds correlation maxima at heights of closest agreement between the measured and simulated data (VPKC vector) identifying a family of possible target tracks (H_n, H_{2n}, ..., H_R, ..., H_{Rn}). Only one of these target tracks is true (H_R). Its identification is carried out by means of a perturbation technique which relies on the fact that changes to the radar's transmitted frequency result in the perturbation of all target tracks except the true track.

The CHA algorithm is divided up into three major part :

- simulation of the amplitude and phase distribution across the antenna aperture for various postulated values of target height,
- cross-correlation of experimental data with simulated data,
- identification of the true target track.

2.1.1 Simulation of the Amplitude and Phase Distribution

The CHA algorithm uses a model of the electromagnetic field set up over a vertical array. As shown in Fig. 1, consider a source and the image of the source below the reflecting surface illuminating an 8-element vertical array. The signal at element n consists of the sum of two signals: the direct signal and the coherent part of the indirect signal. The CHA equation (details in [2]) models the interference pattern, which depends upon the location of the target. It has the following structure when normalized to a particular reference element:

$$T(n,rng,H_i) = 1+\rho SD\ EXP\{j((2\pi/\lambda)nd(\sin\theta_i-\sin\theta_d)-(\phi+2\pi\Delta r/\lambda))\} \qquad (1)$$

where

$T(n,rng,H_i)$:	Theoretical amplitude and phase for a given range (rng), a given height (H_i) at a particular element n
$\rho EXP\{-j\phi\}$:	Complex reflection coefficient
S:	Specular scattering coefficient
D:	Divergence factor
$(2\pi nd/\lambda)\sin\theta_d$:	Phase shift due to element spacing d
Δr:	Curved-earth path-length difference between direct and indirect signal.
θ_d:	Elevation of the direct signal (+ve if above antenna boresight)
θ_i:	Elevation of the indirect signal (-ve if below)

The evaluation of the equation 1 is the most time consuming part of the CHA algorithm and involves three steps:

- derivation of the point of reflection on a curved surface,
- determination of the path length difference and angle-of-arrival for the direct and indirect signals,
- calculation of divergence factor and reflection coefficient as modified by a rough surface.

The mathematical formulae accompanying all these steps are fully described in the Litva and Rook report [2]. At each incremental height, equation 1 must be calculated for each horn element giving a set of 8 complex values.

2.1.2 Cross-Correlation of Experimental with Simulated Data:

The general expression for the complex correlation coefficient denoted by C may be written as

$$C(rng,H_i) = \sum_{\frac{-N-1}{2}}^{\frac{N-1}{2}} \frac{M(n,rng)\ T^*(n,rng,H_i)}{\sigma(rng)\sigma(rng,H_i)} \qquad (2)$$

where

$$\sigma(rng) = \left[\sum_{\frac{-N-1}{2}}^{\frac{N-1}{2}} M(n,rng)\ M^*(n,rng) \right]^{\frac{1}{2}} \qquad (3)$$

$$\sigma(rng,H_i) = \left[\sum_{\frac{-N-1}{2}}^{\frac{N-1}{2}} T(n,rng,H_i)\ T^*(n,rng,H_i) \right]^{\frac{1}{2}} \qquad (4)$$

N:	Number of elements
T():	Theoretical values
M():	Measured values

For each set of experimental data obtained by the CHA, the correlation analysis is repeated NC times, where NC equals the number of postulated target heights. The sampled target heights ($H_1,H_2,\ldots,H_R,\ldots H_{NC}$) are arranged at equal intervals across a height window defined by either the radar's total low angle region or by prior information (fig.2). The CHA performs the correlation and peaks are observed at heights of closest agreement between the measured and simulated data. Multiple peaks are observed because of the periodicity of radar's interference pattern. The tracks are simply specified by the peaks.

2.1.3 Identification of the True Target Track:

If the radar frequency and/or antenna height were perturbed in a known manner, deviations exhibiting similar characteristics would appear on all tracks except on the true track [2]. Simulated results and mathematical development are presented in [2] to illustrate that radar frequency and/or antenna-height perturbation technique can be used for identifying the true track.

3 REAL-TIME IMPLEMENTATION USING AN ARRAY PROCESSOR

This section describes the implementation of the CHA algorithm on an FPS AP-120B array processor with a DEC LSI-11/73 MICRO PDP as the host computer. General considerations on array processor architecture and programming will be presented first, followed by a description of the AP-120B/GPIOP real-time CHA software. Finally, some considerations to reduce the number of floating point operations are discussed.

3.1 Array Processor Architecture

Array processors have been developed which have overcome the limitations of the Von Neumann architecture, which stores data and program instructions in the same memory. These machines are normally used to augment the processing speed capability for a variety of batch scientific processing jobs or to provide a more dedicated capability for real-time data processing and analysis.

The fundamental and dominant operation involved in digital signal processing systems is the following multiply-add operation:

$$\sum_i A_i X_i$$

The most important requirements for signal processor implementation of the above calculation are accuracy and speed. The accuracy determines dynamic range and signal-to-distortion ratio. The speed directly affects the amount of signal processing, and consequently the real-time capability. There are two major approaches to high-speed computation: the utilization of fast components and the utilization of parallel architecture.

Peripheral array processors use parallelism and pipelining to attain high data processing speeds. They are most powerful and effective in performing multiplication and addition operations, but leave much to be desired when performing general-purpose operations such as tests, bit, byte and string manipulation as well as a variety of data-dependent tasks.

The architecture of peripheral array processors supports the concurrent execution of arithmetic and memory access operations. This type of parallelism is applicable to all computational operations, not just to the elementary steps of multiplication and addition. This implies that the system should be able to simultaneously fetch both the instruction to be executed, and the data to be multiplied and added in every cycle. Paths for the movement of all of these data must be provided and the circuitry for address computation, loop counting and other operations must be supported in parallel.

In general, one can program sequential general-purpose computers without detailed knowledge of their architecture. However without detailed diagrams of the machine architecture, it is impossible to program peripheral array processors. It is necessary to know which operations share which buses because operations sharing the same bus cannot be executed in parallel, thus creating major programming difficulties. This is the reason for describing the FPS AP-120B peripheral array processor and the General-purpose Programmable Input/Output Processor (GPIOP) used to implement the CHA algorithm in real time.

3.1.1 The FPS AP-120B Array Processor:

The AP-120B [4-7] is a high-speed (167 ns cycle time) peripheral floating point arithmetic array processor (AP) which is intended to work in parallel with a host computer. Its use in applications such as CHA can significantly improve the throughput of a computer-based system. The AP-120B internal organization is particularly well suited to performing the large numbers of iterative multiplications and additions required in CHA processing. The architecture permits each logical unit of the machine to operate independently and at maximum speed.

The highly-parallel structure of the AP allows overhead of array indexing, loop counting and data fetching from memory to be performed simultaneously with arithmetic operations on the data. Since multiplication and addition require two different operands, it is desirable that both operations be performed in parallel. For this reason, the AP-120B has two distinct data pad memories (DPX,DPY) as well as dual arithmetic logic units: a floating point adder (FADD) and a floating point multiplier (FMUL). The size difference between data words and instruction words combined with the requirement for parallel access to them force the AP-120B to have several kinds of memory for storing program instruction (PS), fast-access table memory for precomputed constants (TM), and main data memory for mass storage (MD). These main system elements are interconnected by multiple buses which also operate independently. A general block diagram of the AP-120B functional units appears in Fig.3.

Because of the microcode nature of array processor programs, instruction words are typically wider than data words. Each instruction word has 64 bits so that many different operations may be performed concurrently during each cycle. It is divided into five general areas: program control, address control, arithmetic memory and I/O. In order to achieve high-speed capability, each instruction of a microcoded program should use as many fields as possible. Consequently, to attain maximum speed, all adders and multipliers should compute a useful output during each machine cycle keeping the following elements activited:

- FMUL, the floating point multiplier, a three-stage pipeline;
- FADD, the floating point adder, a two-stage pipeline;
- MD, the main memory, a three-stage pipeline, with a call possible every two cycles, if 'even' and 'odd' memory addresses are called alternately;
- TM, a writeable table memory in a two-stage pipeline;
- DPX, a 32 word scratch-pad memory;

- DPY, a 32 word scratch-pad memory;
- SP, the S-pad, a memory address calculator.

3.1.2 The General-Purpose Programmable I/O Processor (GPIOP):

The AP-120B is dedicated to perform intensive calculations while the host computer usually handles all programming input, file manipulation and I/O operations. However, for CHA real-time applications greater speed can be obtained from AP-120B, if I/O access is handled directly via a dedicated I/O handler (GPIOP) [8-10].

The GPIOP consists of two major elements, a control processor (CPROC) and a data processor (Fig.4). CPROC is a microcode processor programmed to control data transfers between the AP and an external device (e.g. radar acquisition system [11]). The data processor consists of two separate elements, the format processor (FPROC) and the FIFO memory. FPROC is dedicated to data format conversion and data transfer. The FIFO memory holds the I/O data, as it passes through the GPIOP, to synchronize the speed of the external device with the speed of the AP.

3.2 Array Processor Programming

As mentioned previously, optimal coding of an algorithm on an array processor is substantially more difficult than on a sequential machine. The programmer must understand and control the many parallel data paths and take into account the relative delays between each operation in order to achieve efficient pipelining. On a sequential machine one instruction means one operation while with array architecture one instruction means multiple concurrent operations. The array architecture performs multiplications and additions in every cycle while concurrently performing all the necessary data access and address calculations required to feed the arithmetic units.

A fundamental operation in a sequential machine, for indicating that a set of operations is to be translated to vector form, is the LOOP. The LOOP determines the number of vector elements to process for all operations. A sequential execution of a LOOP begins by fetching inputs, computing results and then storing them, and a test is performed to recognize the loop end. On a parallel and pipeline processor it is drastically different. Fig. 5 depicts the flow of tasks when the LOOP is executed on a pipeline machine. The strategy is to look ahead far enough to provide all data at the required time and to ensure that each of the computing elements has a minimum idle time. It means that while multiplications and additions are performed on a set of data, all necessary data access and address calculations are concurrently performed on the next set of data and so on depending on the pipeline level.

A number of items affect the rate at which a given task runs on the array processor (AP). Significant factors are:

- the cycle rate of the main data memory,
- the placement of vectors in the main data memory,
- the host system overhead,
- the timing of data transfer between the host and the AP,
- the AP execution time of the routine.

Prior to the execution of a routine by the AP, the host system must load the routine into the AP program source memory (if it has not been previously loaded) and must load the parameters into the S-pad registers. This time interval, called the host overhead, varies from system to system depending on the complexity of the host operating system. Host overhead is typically 100 to 1000 microseconds. The time required for program initiation, communication between the two processors, and data transfer can be the same or greater than the array processor execution time.

The most effective method of minimizing the effects of host overhead is to reduce the number of calls to the AP from the HOST. Clearly, a program that runs many short tasks in the AP or spends most of its time transferring data to and from the AP is not well designed. Consequently, it is highly desirable to organize array processor application codes such that they minimize requests for CPU service and system resources by chaining together multiple AP calls in order that they may be sent to the array processor program memory in one data transfer. The result will be to lessen, but not to eliminate, the program initiation time penalty.

Another item which affects the real-time use of an array processor is the I/O process. Performing I/O access directly rather than indirectly through the host is more efficient for the AP. Because large amounts of data enter and leave the AP, direct memory access (DMA) is the preferred method of transfer. The data format conversion is also an important aspect of the I/O process. Conversion of the data from one floating point format to another must be fast because it can have an impact on the effective transfer rate.

Simultaneous transfer and processing of data can also be helpful to reduce the overhead for applications such as the CHA. The AP and GPIOP are built so that their parallel operation allows the programmer to write tasks which process and transfer data simultaneously. The advantage is that it can speed up program run time without loss of synchronization.

When real-time processing is considered, the pre-loading of constants and tables to prime the task to be performed is important. Also the microcode instructions necessary to execute the task must be loaded in the appropriate memory, preferably in contiguous memory locations, to eliminate overlays. In the same way, the AP is most efficiently used when a sequence of operations is performed on one or more sets of data which reside in an internal data memory. This reduces data transfer overhead and retains maximum numerical precision.

Finally, after consideration of all of these items, the AP throughput rate for the CHA algorithm will be maximized by careful and efficient programming of all available parallel functional units (multiplier, adder, GPIOP, memory fetches, and display of results) .

3.3 CHA Real-Time Software

The original CHA algorithm was coded in Fortran IV on a PDP-11/34. It was converted to array processor syntax to achieve real-time operation. The difficulties involved in conversion of the CHA program were due to microprogramming, which is necessary for efficient operation of the AP. Because of the parallelism and the many data paths in the array processor, it was difficult for a higher-level language to make full use of the capabilities that the architecture can support. Consequently, using sophisticated programming tools sacrifices execution efficiency, the primary motivation for using an array processor. Microprogramming a task on an AP is obviously more difficult than machine language programming on sequential machines. Speed and simplicity are two conflicting parameters; increasing speed means using a low level of programming language, while increasing simplicity requires use of a high level programming language.

3.3.1 Vector Translation of the Original CHA Program

The CHA program which solves the mathematical equations of section 2 takes up approximately 370 lines of Fortran code divided up into 10 subroutines namely:

- FINDP - Determination of peaks of the correlation function.
- CORMH - Determination of the corresponding correlated heights.
- THEOC - Determination of simulated complex phase and amplitude distribution across aperture for a given range.
- CURVE - Determination of reflection coefficient for a curved earth geometry.
- REFCO - Determination of reflection coefficient taking into account the surface roughness factor.
- RLCOR - Determination of cross-correlation coefficients.
- HGHTF - Determination of the true track
- SQUEZ - Subroutine to squeeze heights.
- SETHV - Determination of height vector.
- SUBAR - Determination of the antenna horn configuration.

The first logical step was to identify the subroutines that could best be done in the AP. It was found that all of the CHA algorithm could be efficiently programmed on an array processor. The first six subroutines have a vector struçture suitable to be executed on an array processor. The first five routines calculate the theoretical values to be correlated with measured data to obtain the target height. The RLCOR routine determines the correlation coefficient. The last four routines perform track initiation and formation.

The fundamental method in Fortran for indicating that a set of operations is to be performed across whole arrays of data is the DO loop. Since the DO statement determines the number of vector elements to process for all operations in the loop, it is this construct on which vector translations are based. Within a DO loop, the occurence of any of the following will interfere with vector translation:

- a subroutine CALL;
- an I/O statement;
- branches to other parts of the program;
- nonlinear indexing of an array;
- recursive program segments (feedback of results from one pass as input to the next).

In the CHA program, no I/O statements are used within any of the DO loops. The restructuring needed to make array optimization possible, consisted of putting all the subroutines into a loop or putting a loop into the subroutines. Branches were eliminated and recursive relations, which occur only once in the CURVE subroutine, were microprogrammed to simplify vector translation.

Programming the CHA algorithm on a complete HOST-AP-GPIOP system (Fig.4) is a multilevel task. The system used in this study contained 4 processors, each of which is a programmable device:

- HOST (DEC MICRO-PDP-11/73),
- AP-120B Array Processor,
- GPIOP (I/O handler): CPROC and FPROC processors.

Each processor functions independently although the CPROC controls the FPROC, the AP controls the CPROC and finally the HOST controls the AP in a master-slave mode. The AP-120B performs all intensive CHA calculations while the host displays the results: the range versus the target height. The GPIOP receives commands from the AP, and transmits data between AP-120B Main Data Memory and the external device which is the radar system [11].

3.3.1.1 Host Development Software

The HOST is the source of all software needed to program the CHA application on the AP-120B/GPIOP. The programmer has to be familiar with firmware software options supplied by FPS in order to obtain the performance expected for the particular application. The software packages [12-20] supplied by FPS help the user in

- writing programs;
- downloading programs in AP and GPIOP memories;
- running programs;
- and diagnosing hardware faults.

The Host CHA software flowchart is given in appendix A.

3.3.1.2 AP-120B CHA Software

The AP-120B CHA program relies heavily on routines contained in a mathematical library of microcode subroutines available from the manufacturer [12]. These microcode units can be linked with user developed routines to implement the CHA algorithm with a reasonable software development effort. The tracking process is not very suitable for execution on the AP since many logical decisions and non-vector arithmetic operations have to be performed sequentially. Those user developed routines are carefully designed to describe the complete tracking process. The flowchart describing all the CHA processing inside the AP-120B is presented in Appendix A. The AP program requires approximately 3K of AP program memory. It can reside entirely in the AP program memory and can be invoked by using only one CALL statement.

3.3.1.3 GPIOP CHA Software

The GPIOP contains 2 programmable processors. Each processor uses a programming language designed specifically to fulfil the functions of that processor. The assemblers for these languages run entirely on the host processor. The GPIOP can be programmed using either its own assemby language (GPAL) or a higher level interpreted language (IOCAL). GPAL provides a full set of instruction memory for the CPROC. IOCAL is an interpretive language using higher level instructions than GPAL microcode instructions. This eliminates much of the complexity involved in microcode programming. The IOCAL instructions permit the GPIOP to function as a channel processor dedicated to perform I/Os for a specific device (e.g. radar system).

The user can take any of the two approaches to program the GPIOP. However, the use of IOCAL requires a significant amount of overhead, both in processing time (5 to 10 usec) and CPROC memory space. Therefore, programming the CPROC directly in GPAL is more appropriate for the CHA application in which the speed of I/O is critical. The flowchart for the CPROC program is presented in Appendix A. CPROC program is closely related to hardware.

The programs that operate within the GPIOP-FPROC are available in a format conversion library supplied by the manufacturer. The GPIOP format library has a subroutine called SP11AP [10] running on the FPROC which converts a PDP11 floating point number to an AP floating point number . Only that subroutine is needed for the FPROC software. Note that the FPROC cycle time is three times faster than the CPROC cycle. Its cycle time of 55.6 ns meets the requirement for faster data format conversion, hence transfer rate is not affected.

3.4 Considerations to Reduce Calculations:

The CHA linear array consists of N equally-spaced elements separated by a distance d. The reference element is assigned the index n=5. The other elements are assigned indices depending on the following distribution relative to the reference element:

$$D(n) = (2\pi d/\lambda)(Ref-n) \quad \text{for } n=1,8 \quad (Ref=5) \tag{5}$$

Considering the distribution D(n), all calculations involving this distribution could be reduced substantially. Around the reference element, Ref=5, the following is observed:

$$D(5) = 0.0 \tag{6}$$

$$D(6) = -D(4) \tag{7}$$

$$D(7) = -D(3) \tag{8}$$

$$D(8) = -D(2) \tag{9}$$

This result is important for the real-time processing since it can save a lot of calculations in the subroutine THEOC, which is the most time consuming step of the CHA algorithm (see Table 4.1). Rearranging equation (1) to determine the simulated phase and amplitude distribution across the aperture for a given range at a given height gives the following equation:

$$T(n,rng,H_i) = \exp\{j(\theta d*D(n))\} + ZR*\exp\{j(\theta_i*D(n))\} \quad \text{for } n=1,8 \tag{10}$$

where ZR: Complex reflection coefficient as modified by a surface roughness factor.

Half of the calculations can be saved when evaluating $\exp\{j(\theta d*D(n))\}$ and $\exp\{j(\theta_i*D(n))\}$ because

$$\exp\{j(\theta d*D(5))\} = \{1,0\} \tag{11}$$

$$\exp\{j(\theta d*D(6))\} = \exp\{-j(\theta d*D(4))\} \tag{12}$$

$$\exp\{j(\theta d*D(7))\} = \exp\{-j(\theta d*D(3))\} \tag{13}$$

$$\exp\{j(\theta d*D(8))\} = \exp\{-j(\theta d*D(2))\} \tag{14}$$

(similarly with θ_i)

With this new set of equations (11-14) we can evaluate (10) at n=5 the reference element giving:

$$T(5,rng,H_i) = ZR + 1.0 \tag{15}$$

The final step to get the complete set of 8 complex amplitude and phase values across the aperture is the normalization process described by the following equation:

$$T(n,rng,H_i) = T(n,rng,H_i)/T(5,rng,H_i) \tag{16}$$

Several multiplications can be eliminated by use of the following equations. Assume that

$$\{A2(n),\ B2(n)\} = \exp\{j(\theta d * D(n)\} \tag{17}$$

$$\{A1(n),\ B1(n)\} = \exp\{j(\theta_i * D(n)\} \tag{18}$$

$$\{C1,D1\} = ZR \tag{19}$$

$$\{C2,D2\} = 1/T(5,rng,H_i) \tag{20}$$

then for n 5

$$\begin{aligned} T(n,rng,H_i) = \{&C1A1(n)C2-D1B1(n)C2+A2(n)C2-C1B1(n)D2-\\ &D1D2A1(n)-B2(n)D2\} + j\{C1B1(n)C2+D1A1(n)\\ &C2+B2(n)C2+C1A1(n)D2-D1D2B1(n)+A2(n)D2\} \end{aligned} \tag{21}$$

for n 5

$$\begin{aligned} T(n,rng,H_i) = \{&C1A1(n)C2+D1B1(n)C2+A2(n)C2+C1B1(n)D2-\\ &D1D2A1(n)+B2(n)D2\} + j\{-C1B1(n)C2+D1A1(n)C2-\\ &B2(n)C2+C1A1(n)D2+d1D2B1(n)+A2(n)D2\} \end{aligned} \tag{22}$$

for n = 5

$$T(5,rng,H_i) = \{1,0\} \tag{23}$$

It is obvious that half of the multiplication operations are eliminated in the most time consuming steps of the CHA processing.

4 EVALUATION OF THE REAL TIME PERFORMANCE OF THE CHA PROCESSOR

This section presents the results of a study concerning the processing speed of the AP-120B when running the CHA algorithm. A complete breakdown of the number of basic operations contained in the CHA will be presented in order to identify the most time consuming steps. Then, a brief discussion concerning the arithmetic accuracy and speed of these operations on the AP-120B is also included. Finally, the results of the timing study are presented.

4.1 Number of Basic Operations in the CHA Algorithm

An assembly listing of all subroutines was obtained to determine a complete breakdown of the number of CHA basic operations. The results are tabulated in table 4.1 and table 4.2.

TABLE 4.1

NUMBER OF BASIC OPERATIONS TO CALCULATE THE CORRELATION FUNCTION
(for one height sample)

SUBROUTINE NAME	MEMORY REFERENCE GROUP looping indexing	EXTENDED ARITHMETIC GROUP load,store	FLOATING POINT GROUP							TOTAL
			+/-				SIN	COS	EXP	
FINDP	74	8	1	2	1	-	-	-	-	86
CORMH	88	5	2	-	-	-	-	-	-	95
THEOC	453	88	15	100	16	-	17	17	-	706
CURVE	150	81	22	28	12	1	-	-	-	293
REFCO	59	18	19	24	5	-	1	-	3	125
RLCOR	377	1	79	118	2	1	-	-	-	577
TOTAL	1201	201	138	272	36	2	18	17	3	1882

TABLE 4.2

NUMBER OF BASIC OPERATIONS TO IDENTIFY AND FOLLOW THE TRUE TRACK

SUBROUTINE NAME	MEMORY REFERENCE GROUP looping indexing	EXTENDED ARITHMETIC GROUP load,store	FLOATING POINT GROUP			TOTAL	MULTIPLICATIVE FACTOR
			+/-				
HGHTF	70	14	-	2	2	88	1
	130	24	7	-	2	163	NPEAKS
	41	6	3	-	-	50	NPEAKS**2
SETHV	45	11	3	2	-	61	1
	31	3	2	-	-	36	NC
SQUEZ	7	6	-	-	-	13	1
	55	-	1	-	1	57	NPEAKS
SUBAR	11	6	1	-	2	20	1

4.2 Processing Speed of AP-120B Basic CHA Operations

Table 4.3 presents all the basic floating point operations used to program the CHA algorithm on an AP-120B.

TABLE 4.3
AP-120B FLOATING-POINT PROCESSING SPEED

OPERATION	TIME/LOOP (usec)	SETUP TIME (usec)
ADD/SUB	1.0	2.67
MUL	1.0	2.67
DIV	1.67	4.17
EXP	2.33	4.17
SQRT	1.83	4.17
SIN	1.33	7.00
COS	1.33	7.00

The execution time is given for a 167 nsec memory access. Note that the execution time is specified on a per loop basis. Thus, if we add two 1000 element vectors, the execution time with a 167 nsec memory is 1000 * 1.0 = 1000 usec, plus an additional 2.67 usec of setup time needed to initially fill the AP pipeline. Therefore, the execution time using a 167 nsec memory is 1002.67 usec.

4.3 AP-120B CHA Digital Processing Noise

This section presents a brief discussion on the arithmetic error due to internal floating-point format [6] of the AP-120B when processing the CHA algorithm and a comparison is made with typical radar receiver noise.

AP-120B FLOATING-POINT FORMAT

0 EXPONENT 9	10 MANTISSA 37

EXPONENT E0-E9	10 bits	two's complement fraction
MANTISSA M0-M27	28 bits	binary exponent biased by 512

The value of a floating point number in this format is defined as

$$(\text{MANTISSA}) * 2^{-512}$$

The positive dynamic range of this format is from:

$$3.7 * 10^{-155} \quad \text{to} \quad 6.7 * 10^{153}$$

The negative dynamic range is

$$-1.8 * 10^{-155} \quad \text{to} \quad -6.77 * 10^{153}$$

The 28-bit fraction, combined with the convergent rounding algorithm used in the floating point adder and multiplier, gives a maximum relative error of

$$7.5 * 10^{-9} \qquad (2^{-27})$$

per arithmetic operation. This is a precision of 8.1 decimal digits. As a comparison, unrounded IBM 360 format gives only 6.0 decimal digits of arithmetic accuracy. Referring to Table 4.1, when calculating the correlation function we have approximately 2000 basic operations per pass resulting in an arithmetic error of

$$1.5 * 10^{-5}$$

This is a precision of 4.8 decimal digits.

A study has been carried out to examine the design alternatives for a radar front-end to be used for the CHA [21]. Using typical parameters a receiver power budget was worked out and the S/N ratio found was 17 dB which represents a precision of 1.7 digits. Considering all radar signal noise sources, the digital noise introduced by AP-120B, when processing the CHA, becomes negligible.

4.4 Results of the CHA Timing Study

The system consists of a DEC MICRO-PDP 11/73 with an attached FPS AP-120B array processor. The original host system was a PDP-11/34. Some results in this study are based on the original system. It was necessary to upgrade the PDP-11/34 to a MICRO-PDP because a better reliability and a physical size reduction were required for sea trials. The major disadvantage in the migration from the UNIBUS system (PDP-11) to the Q-BUS (MICRO-PDP) is the reduced speed. A special converter (QNIVERTER [22]), permitting a Q-BUS computer system to access a UNIBUS compatible device such as the AP-120B, was used. There is a timing difference between the two buses. The time sharing of the 16 data lines with the 18 address lines on the Q-BUS slows down the Q-BUS as compared to the UNIBUS which has separate data and address lines. The reduction of 16 lines from the UNIBUS to Q-BUS and the delay for the QNIVERTER to match the two buses reduce the data transfer speed performance by 28%.

4.4.1 CHA Processing Time with Experimental Data

The objective of this section is to compare the CHA processing time on the AP-120B with the processing obtained with a sequential program on the PDP-11/34. The time required to process and display a complete file of 577 records by the PDP-11/34 was over 1 hour (6.25 sec/sample). The same file processed and displayed with PDP-11/34 and AP-120B took approximately 50 sec (86.6 msec/sample).

In order to separate the host time used to display the results, and the AP processing time, the experiment was repeated once with only half of the results displayed, and once with no display at all, as shown in the following table.

TABLE 4.4

CHA PROCESSING TIME WITH EXPERIMENTAL DATA
(PDP-11/34/AP-120B)

	FULL DISPLAY	HALF DISPLAY	NO DISPLAY
TIME/FILE (sec)	50	30	10
TIME/SAMPLE (msec)	86.6	52	17.33

The trial with experimental data was an interim step to measure the full capability of the AP-120B to process CHA. The process of retrieving the disk resident data, normalizing and adding calibration coefficients reduced the data transfer rate. It took 195 sec to retrieve and prepare 17507 records for CHA processing, which corresponds to an overhead of 11.2 msec/CHA sample. Thus, to measure the optimal processing speed of the Host/AP-120B system, simulated data was used.

4.4.2 CHA Processing Time with Simulated Data

A target was simulated by moving from 11 km to 1 km and taking samples at each one meter height interval. This represents 10000 samples to be processed. The height of the target was fixed at 20 m. Also the number of simulated heights was kept constant at 100. An attempt was made to obtain a breakdown of the total processing time as follows:

- host overhead;
- display (graphic);
- AP-120B CHA processing time.

The results are presented in Table 4.5

TABLE 4.5a

RESULTS OF THE AP-120B CHA TIMING STUDY
(NC:number of heights=100)
(Host:DEC MICRO-PDP 11/73)

DISPLAY DENSITY	LOOP INSIDE HOST	LOOP INSIDE AP-120B	WAIT ON AP RUNNING (APWR)	GET DATA (APGET)	WAIT ON DATA (APWD)	TIME msec/loop
FULL	10000	1	YES	YES	YES	87.5
1/2	5000	2	YES	YES	YES	43.7
1/4	2500	4	YES	YES	YES	21.9
1/10	1000	10	YES	YES	YES	14.0
NO	10000	1	YES	YES	YES	16.1
NO	1	10000	YES	YES	YES	12.6
NO	10000	1	YES	YES	NO	15.5
NO	10000	1	YES	NO	NO	14.3
NO	10000	1	NO	YES	NO	13.1
FULL	10000	1	YES	YES	NO	87.5
FULL	10000	1	NO	YES	NO	87.5
NO	10000	NO	NO	YES	YES	1.9
NO	10000	NO	NO	YES	NO	1.3

TABLE 4.5b

RESULTS OF THE AP-120B CHA TIMING STUDY
(NC:number of heights was dynamically adjusted)
(Host:DEC MICRO-PDP 11/73)

DISPLAY DENSITY	LOOP INSIDE HOST	LOOP INSIDE AP-120B	WAIT ON AP RUNNING (APWR)	GET DATA (APGET)	WAIT ON DATA (APWD)	TIME msec/loop
FULL	1000	1	YES	YES	YES	85.0
1/2	500	2	YES	YES	YES	42.0
1/4	250	4	YES	YES	YES	21.0
1/10	100	100	YES	YES	YES	2.4
NO	10000	1	YES	YES	YES	7.0
NO	1	10000	YES	YES	YES	1.8

4.4.3 Acquisition Time for GPIOP/Consumer Interface No.1

According to table 4.6, the data acquisition and transfer time, from the radar system to the AP-120B, via GPIOP is 2.8 msec/block data. The host supervisory software overhead would be .7 msec. The data block size is 72 PDP-11 FPN real words. The acquisition time and the format conversion time of the AP-120B/GPIOP/Consumer interface no.1 is 34 Kwords/sec.

TABLE 4.6

ACQUISITION TIME FOR GPIOP/CONSUMER INTERFACE NO.1
(Host:DEC MICRO-PDP 11/73)

NLOOP	NUMBER OF TRANSFERS	TIME (msec)
1	10,000	2.8
10,000	10,000	2.1

4.5 Discussion on the Timing Study

Tables 4.4, 4.5a and 4.5b display various aspects of the CHA timing study, which consist of:

- Host time to display the CHA results graphically;
- Host supervisory software overhead:
  * remove CHA results from AP memory,
  * synchronize HOST and AP.
- AP-120B CHA processing time with
  * experimental data (real noisy data) and
  * simulated data (noise free).

Tables 4.4 and 4.5b present results obtained with both experimental data and simulated data in order to compare the processing time when CHA is used in a real environment. The total number of processed heights is not known precisely. The first pass through the CHA filter consists of approximately 100 postulated heights, each at a 1 metre interval. In subsequent passes the number of correlated heights is reduced to around 20 to 30 and declines to 10 heights when the track is fully established. As expected, experimental and simulated data require the same CHA processing time. The experimental data were stored on a disk and it took 17.33 msec/CHA sample as compare to 7 msec when the data are simulated inside the AP eliminating the ~10 msec disk overhead, which is included in Table 4.4 (see previous section).

In the simulation case (Table 4.5a) the number of postulated heights is kept constant at 100 heights for all passes through the CHA. When the CHA starts, it scans a 100 metre window with the spacing between the heights in the scanned window dynamically varied using the average accumulated deviation between possible tracks in preceding passes. Thus, the 12.6 msec given in Table 4.5a corresponds to the time for the AP-120B to process the CHA filter on 8 complex values at 100 different heights.

The bottleneck limitation of the system is the display of the results on the HP-2648 graphic display terminal. It consumes around 70 msec per range sample which is too slow when compared to the complete CHA processing time (around 13 msec). Viewing the results decimated by 4 or even 10 could be a short term solution to this problem giving a processing time capacity up to 14 msec (Table 4.5a).

Another time factor depicted clearly in Table 4.5a is that of the host supervisory software to move the CHA results (Range, Target height) from the AP main data memory to the host memory and to synchronize their operation. Two wait commands, APWR and APWD are available to obtain synchronization. APWD (wait on data) causes the host program to wait until a data transfer between the host and the AP has been completed. APWR (wait on running) causes the host to wait until the AP has finished running the CHA.

The AP host interface is capable of transferring data to and from the host while it is processing data. APWR and APWD can be omitted because in the CHA application, it is certain that the data being transferred and the data being processed are not the same. The measured host overhead of (16.1-12.6)msec = 3.5 msec can be reduced to .5 msec by using this technique. However it should be used with caution because it has the potential to cause errors in computations.

4.6 CHA Processing Using Block Queuing Technique

The AP-120B is capable of processing 100 CHA heights in 12.6 msec without interaction with the host. The required time increases to 16.1 msec when the CHA data is transferred to the host memory but not displayed. This section proposes an approach to distribute the CHA processing over a complete CHA tracking run.

As mentioned in section 4.4, during the initialization process the correlation function is calculated over 100 postulated heights. The number of heights declines progressively to 10 when the track is established, hence the data rate is reduced by a factor of 10. If a system is designed using 100 heights per pass it will be an over-designed system. It would be more efficient to reserve a portion of the AP-120B main data memory in order to accumulate the incoming CHA data blocks when the AP-120B is unable to process them on a real time basis during track initialization. When the track is established the number of postulated heights is reduced and the AP can regain the real-time rate. This is referred to as block queuing mode, where the processing load is distributed over a complete run leaving the AP with an evenly distributed workload. This technique can be implemented in software by using a block level FIFO with a CHA block status flag on each data block. The first block in would be the first processed block out of the AP-120B (FBIFBO). A CHA processing time approaching 7 msec is possible by using the FBIFBO technique and reducing the display density during the initialization process.

5 CONCLUSION

In this report, an array processor approach has been considered to implement the CHA algorithm as a real-time process. A review of the CHA low-angle tracking technique was given and an implementation on an array processor has been proposed. The corresponding processing time study was also presented.

A brief description of the FPS AP-120B array processor was provided and its real-time processing of the CHA algorithm was investigated.

The displaying of results slows the processing speed. When graphic interactions take place a processing time of 86.6 msec can be achieved, compared with 16.1 msec when the output is not displayed. A processing time of 12.6 msec is possible when all the results reside in the AP-120B. (calculated over 100 heights)

To reduce interactions between AP and the host, the GPIOP, which is a programmable I/O processor that acquires the CHA block data at 34 Kwords/sec, was considered. A block queuing technique, which would allow a CHA processing time of 7 msec, is proposed.

Future work will be required to modify the real-time one-way beacon model described in this report, such that a two-way propagation CHA algorithm including meteorological data and more sophisticated tracking processes can be implemented.

For the CHA computing system, the general points presented in this report will still be very useful. In the present state-of-the-art, the CHA computing system can be upgraded using self-contained array processor boards (20 MFLOPS). This will satisfy the CHA's requirement for high speed throughput. These boards have almost the same general architecture [23] as the AP-120B (12 MFLOPS) and offer increased speed and reduced physical size. The purchase of a new graphics system must be considered seriously. Efforts should be concentrated on a system that can reside on the host main bus, so that it can be accessed at memory speed. This will provide a graphics output system that is concurrent with CHA processing.

6 ACKNOWLEDGEMENTS

This work was undertaken by the Communications Research Centre under the Department of National Defence Project 011LA13. The author wishes to thank Dr. J. Litva and Mr. B.J. Rook for their helpful discussions.

REFERENCES

[1] Marconi Company, " Final Report On Low Angle Tracking ", DSS Contract 2ST81-00113, Ottawa, February 1982.

[2] Rook, B.J., Litva, J., " An Improved CHA Algorithm For Tracking Low Angle Targets ", CRC Report No. 1356, January 1982

[3] Chan, H.C, Litva, J., " Performance of the CHA Algorithm on the UK Low Angle Target Data ", CRC Report No. 1406, October 1986.

[4] FPS Technical Publication Staff, " Processor Handbook ", Publication No. 860-7284-004C, March 1982.

[5] FPS Technical Publication Staff, " Maintenance Manual ", Publication No. 860-727-000C, January 1978.

[6] FPS Technical Publication Staff, " AP Programmer's Reference Manual ", Vol.1,2, Publication No. 860-7319-00IB, June 1981.

[7] FPS Technical Publication Staff, " AP-120B/190L/180V, Software User's Guide ", Publication No. 860-7448-00IB, April 1982.

[8] FPS Technical Publication Staff, " GPIOP Hardware Reference Manual ", Publication No. 860-7425-0008, October 1982.

[9] FPS Technical Publication Staff, " GPIOP Software User's Guide ", Publication No. 860-7497-001B, April 1982.

[10] FPS Technical Publication Staff, " GPIOP Reference Manual ", Publication No. 860-7430-002C, January 1983.

[11] Bosse, E., Caseault, J., Fines, N.R , " A Description of the ELAT Radar Distributed Computing System ", CRC Report No 1425, 1987.

[12] FPS Technical Publication Staff, " APMATH38 Volumes 1 to 4 ", Publication No. FPS-860-7288-008C, July 1982.

[13] FPS Technical Publication Staff, " APAL Reference Manual ", Publication No. FPS-860-7412-003B, April 1982.

[14] FPS Technical Publication Staff, " APLOAD Reference Manual ", Publication No. FPS-860-7410-003B, April 1982.

[15] FPS Technical Publication Staff, " APLINK Reference Manual ", Publication No. FPS-860-7420-001B, March 1982.

[16] FPS Technical Publication Staff, " APSIM/APDBUG Reference Manual ", Publication No. FPS-860-7364-005A, April 1982.

[17] FPS Technical Publication Staff, " Vector Function Chainer Reference Manual ", Publication No. FPS-860-7351-005B, April 1982.

[18] FPS Technical Publication Staff, " AP Test And Verification Reference Manual ", Publication No. FPS-860-7284-004C, March 1982.

[19] FPS Technical Publication Staff, " GPIOP Diagnostic Software ", Publication No. FPS-860-7395-001, Novembre 1981.

[20] FPS Technical Publication Staff, " APEX Reference Manual ", Publication No. FPS-860-7371-004C, March 1982.

[21] Norris, Wong et al., " CRC Radar Front-End Study ", Final Technical Report, Contract No UK-83-0331/1, STL LTD, London, 1984.

[22] ABLE Computer, " QNIVERTER User's Guide ", April 1981.

[23] MERCURY Computer Systems, " ZIP User's Manual", Publication No. 3232-P-U, 1986.

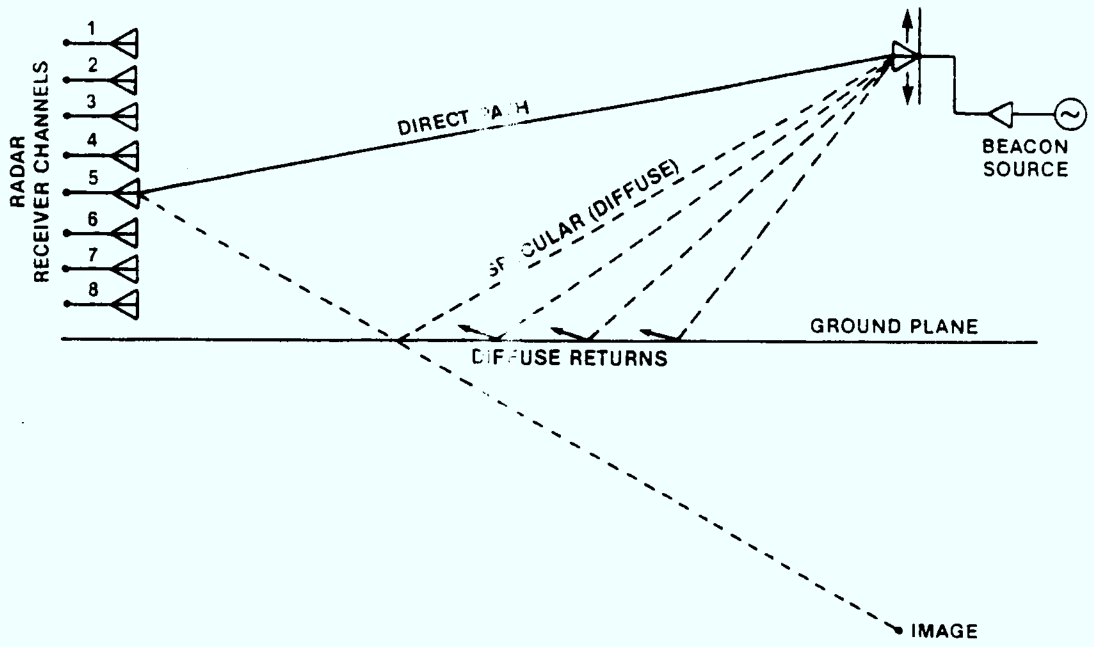


Fig. 1 - Low-Angle Multipath Phenomena

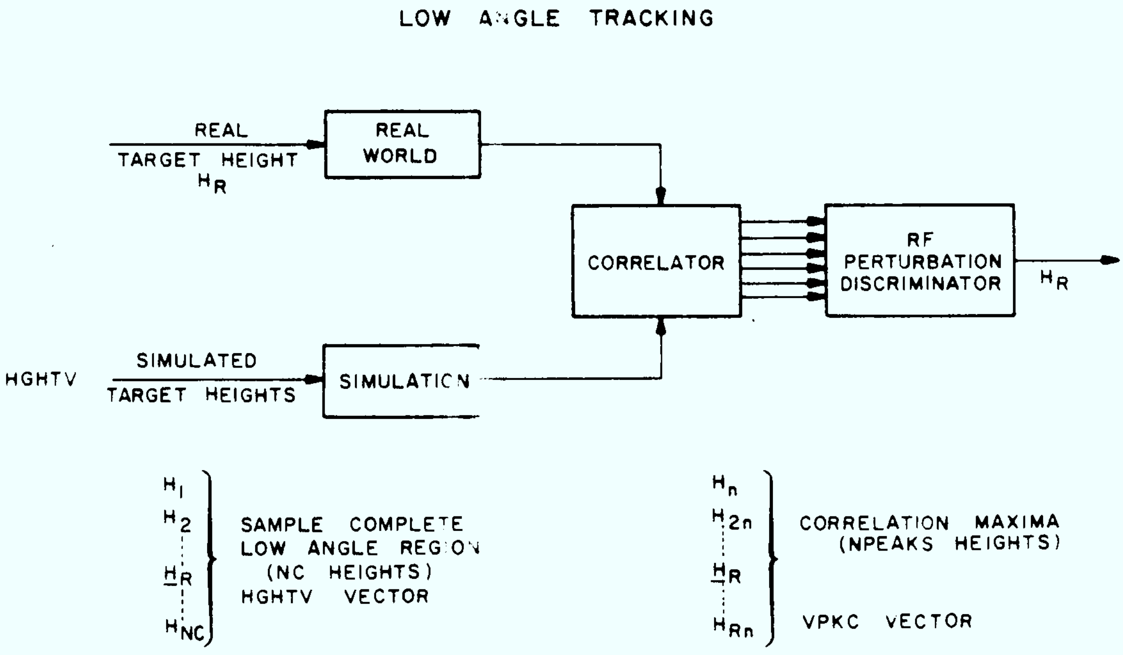

LOW ANGLE TRACKING
REAL TARGET HEIGHT H_R
REAL WORLD
CORRELATOR
RF PERTURBATION DISCRIMINATOR
H_R
HGHTV
SIMULATED TARGET HEIGHTS
SIMULATION
H_1 H_2 H_R H_{NC}
SAMPLE COMPLETE LOW ANGLE REGION (NC HEIGHTS) HGHTV VECTOR
H_n H_{2n} H_R H_{Rn}
CORRELATION MAXIMA (NPEAKS HEIGHTS)
VPKC VECTOR

Fig. 2 - CHA Technique

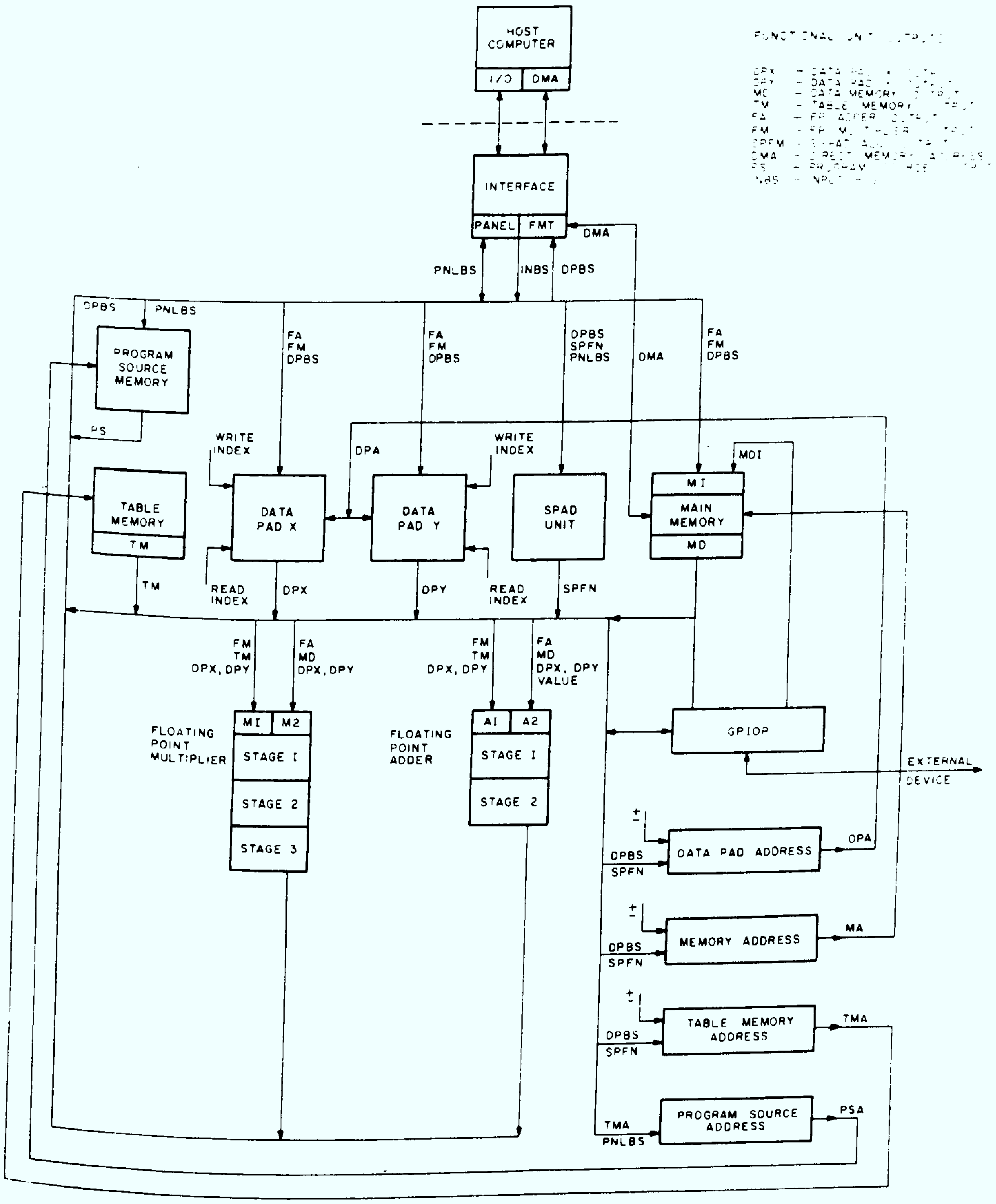


Fig. 3 – AP-120B Functional Block Diagram

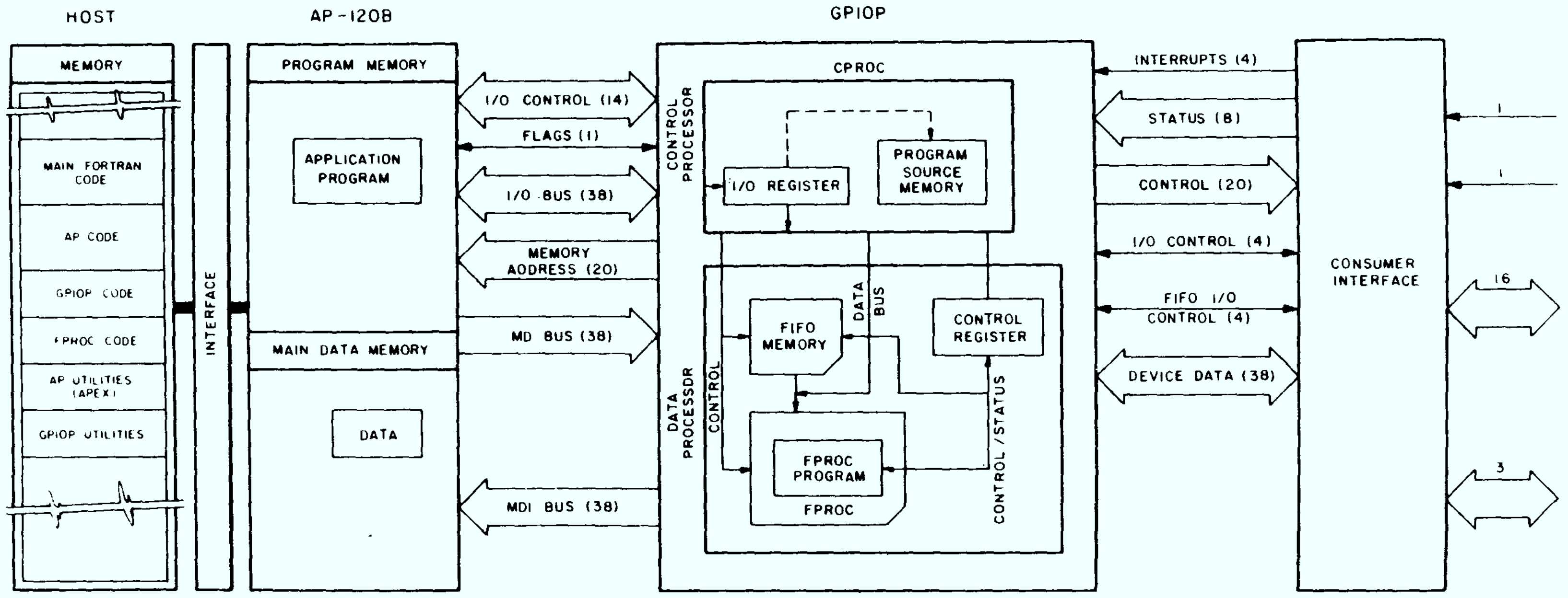


Fig. 4 - CHA Processor Architecture

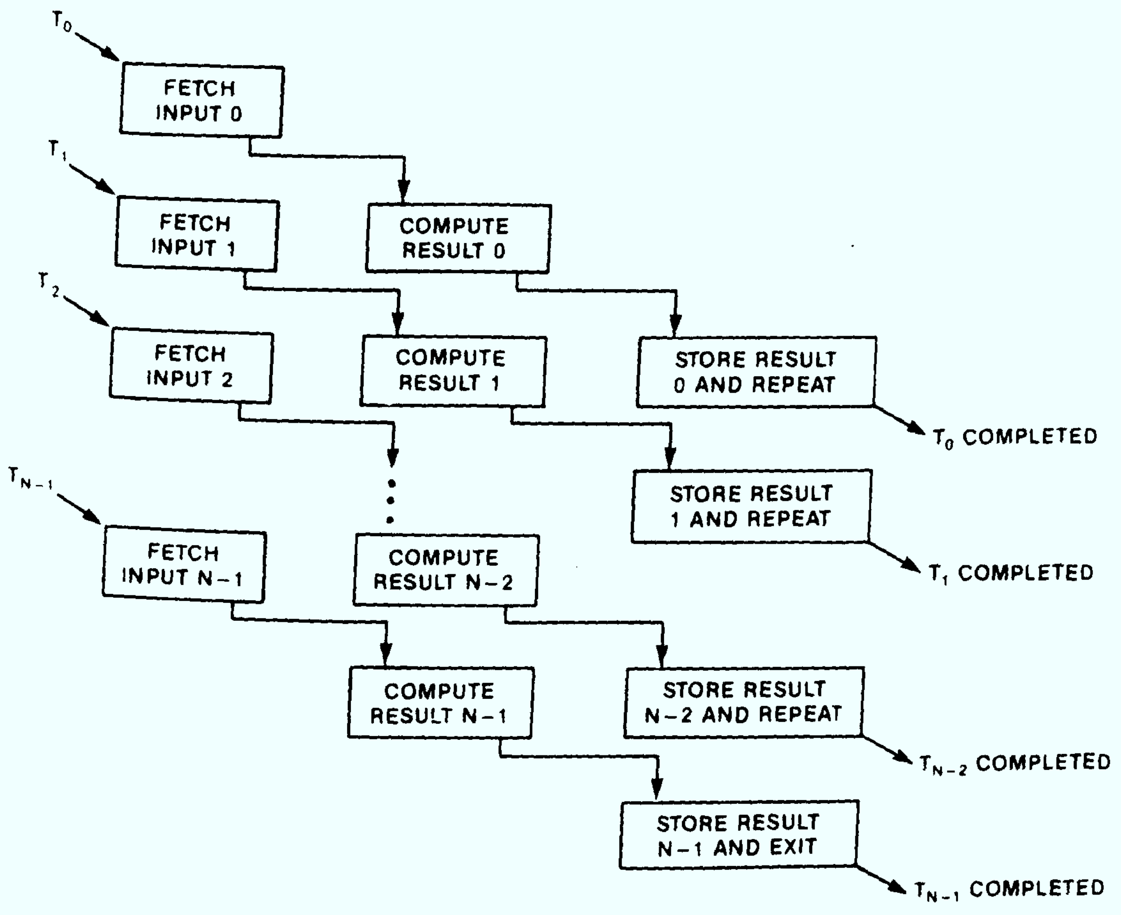
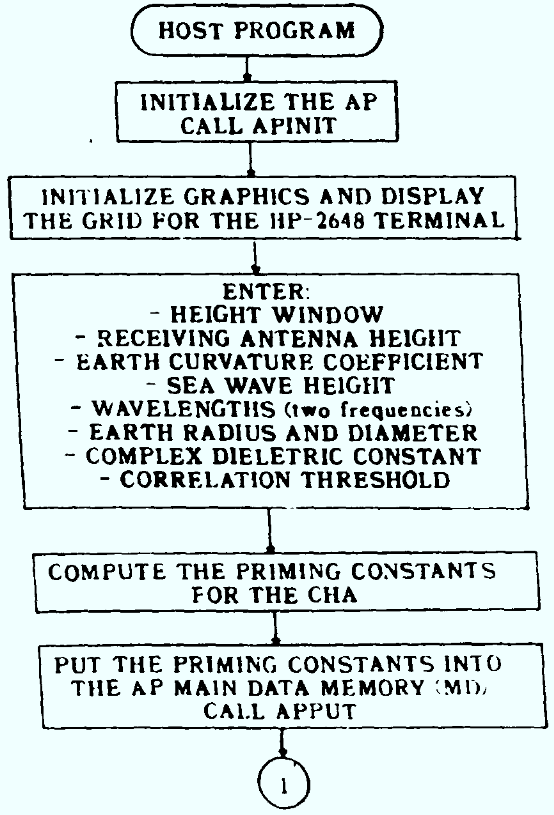


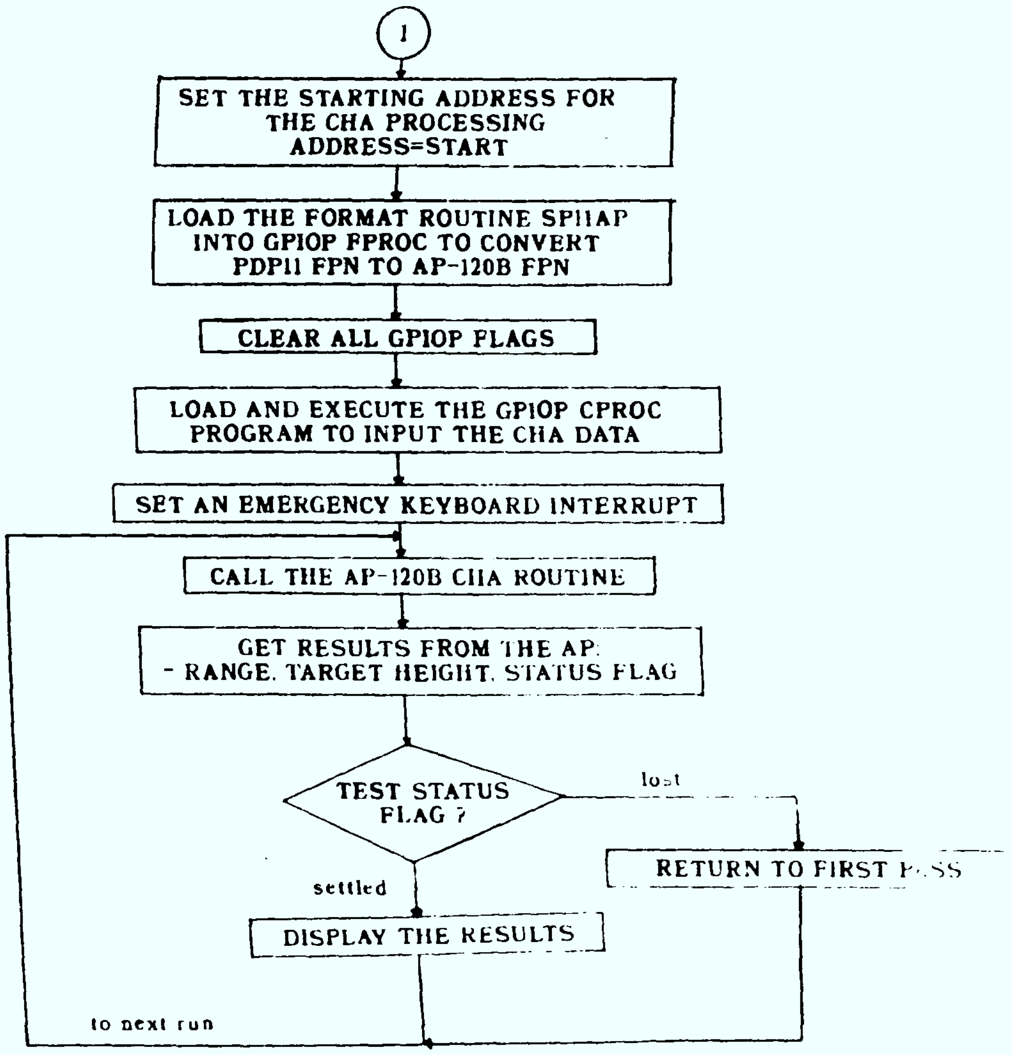
Fig. 5 - Flow of Tasks on a Pipeline Machine

APPENDIX A

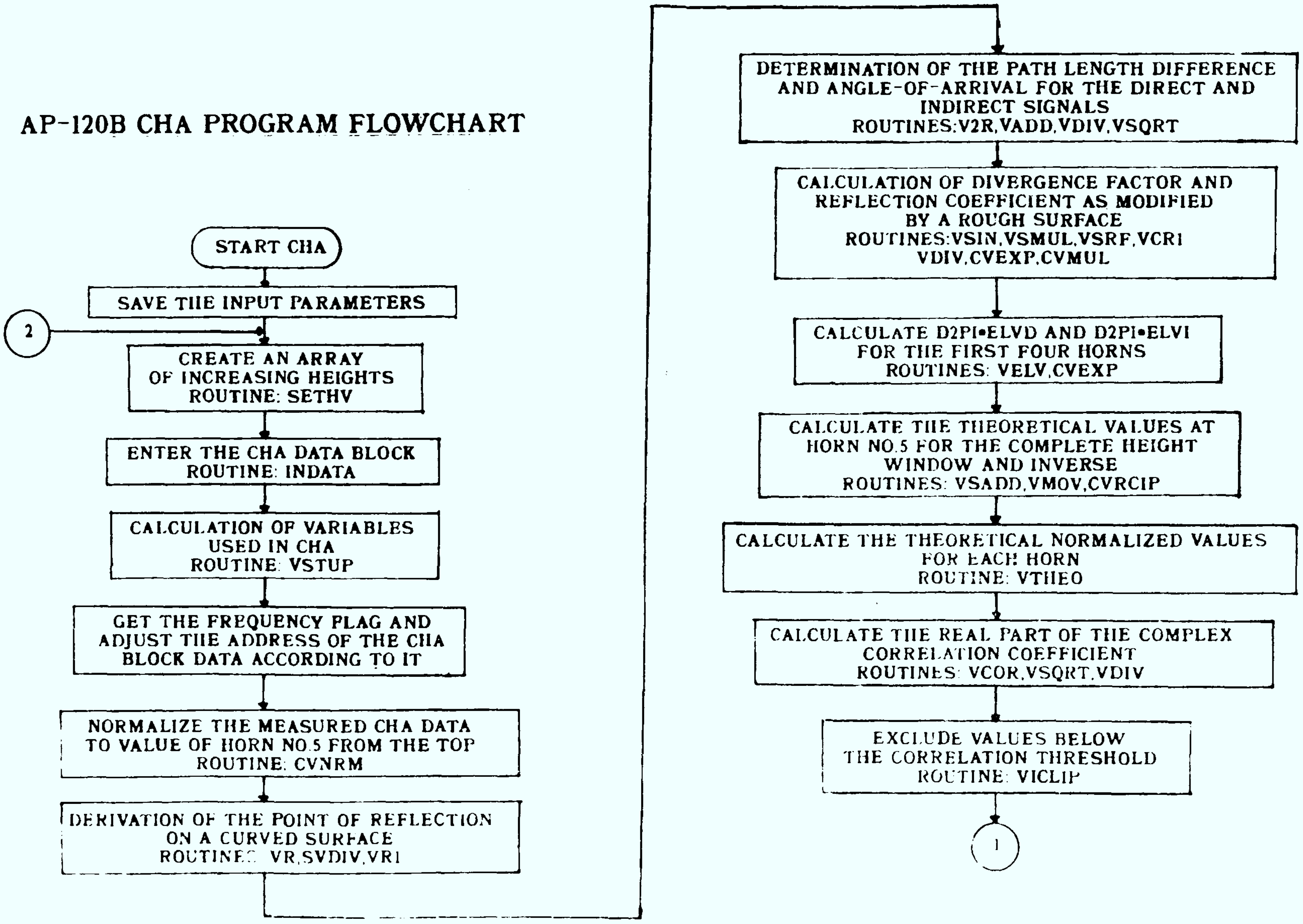
* HOST CHA PROGRAM FLOWCHART
* AP-120B CHA PROGRAM FLOWCHART
* GPIOP PROGRAM FLOWCHART

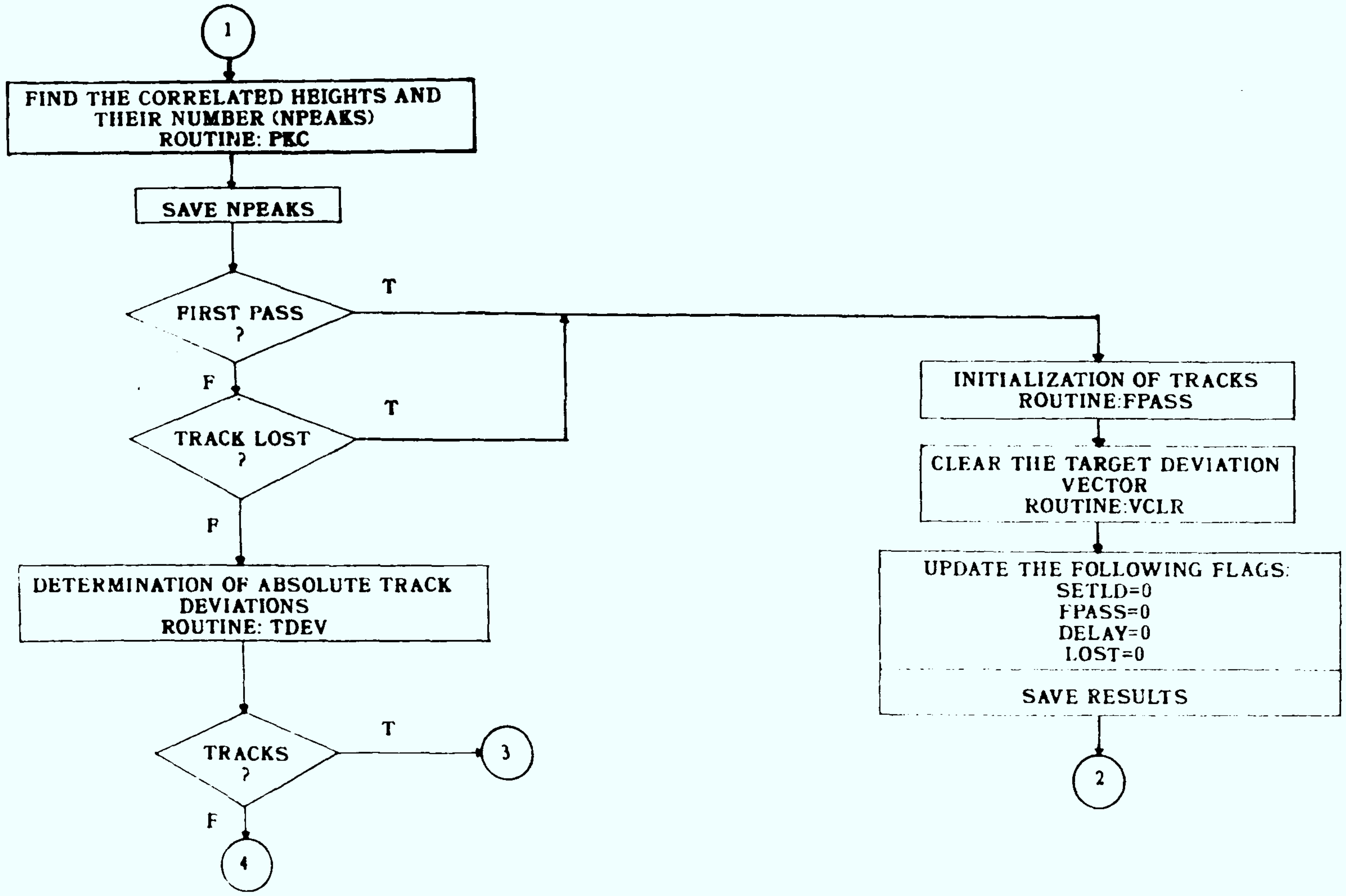
HOST CHA PROGRAM FLOWCHART

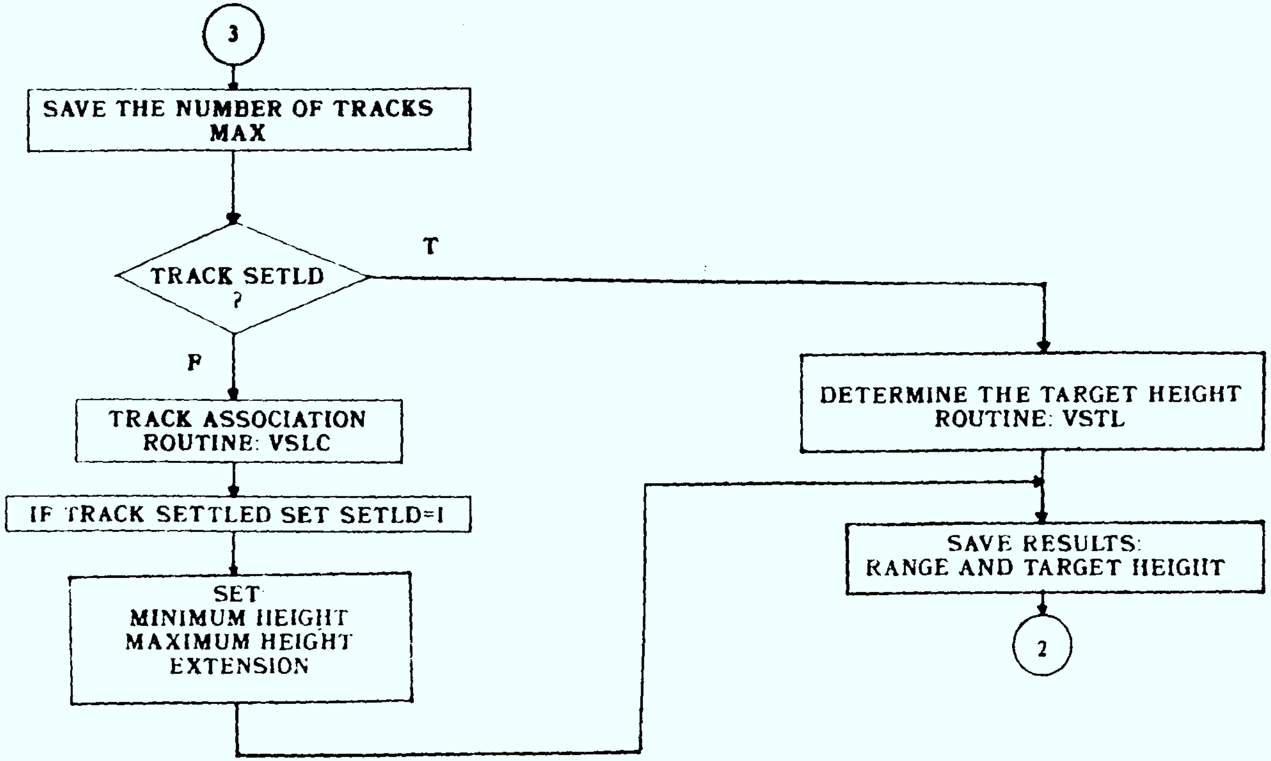

HOST PROGRAM
INITIALIZE THE AP CALL APINIT
INITIALIZE GRAPHICS AND DISPLAY THE GRID FOR THE HP-2648 TERMINAL
ENTER:
- HEIGHT WINDOW
- RECEIVING ANTENNA HEIGHT
- EARTH CURVATURE COEFFICIENT
- SEA WAVE HEIGHT
- WAVELENGTHS (two frequencies)
- EARTH RADIUS AND DIAMETER
- COMPLEX DIELETRIC CONSTANT
- CORRELATION THRESHOLD
COMPUTE THE PRIMING CONSTANTS FOR THE CHA
PUT THE PRIMING CONSTANTS INTO THE AP MAIN DATA MEMORY (MD) CALL APPUT
1

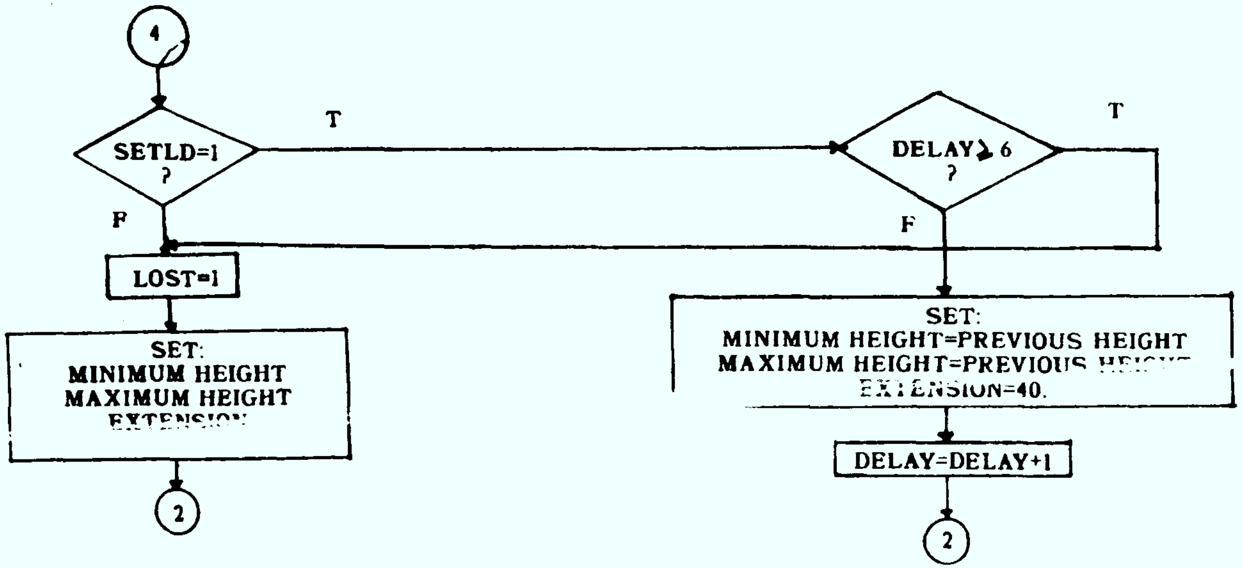

1
SET THE STARTING ADDRESS FOR THE CHA PROCESSING ADDRESS=START
LOAD THE FORMAT ROUTINE SPHAP INTO GPIOP FPROC TO CONVERT PDP11 FPN TO AP-120B FPN
CLEAR ALL GPIOP FLAGS
LOAD AND EXECUTE THE GPIOP CPROC PROGRAM TO INPUT THE CHA DATA
SET AN EMERGENCY KEYBOARD INTERRUPT
CALL THE AP-120B CHA ROUTINE
GET RESULTS FROM THE AP:
- RANGE, TARGET HEIGHT, STATUS FLAG
TEST STATUS FLAG ?
lost
RETURN TO FIRST PASS
settled
DISPLAY THE RESULTS
to next run

AP-120B CHA PROGRAM FLOWCHART

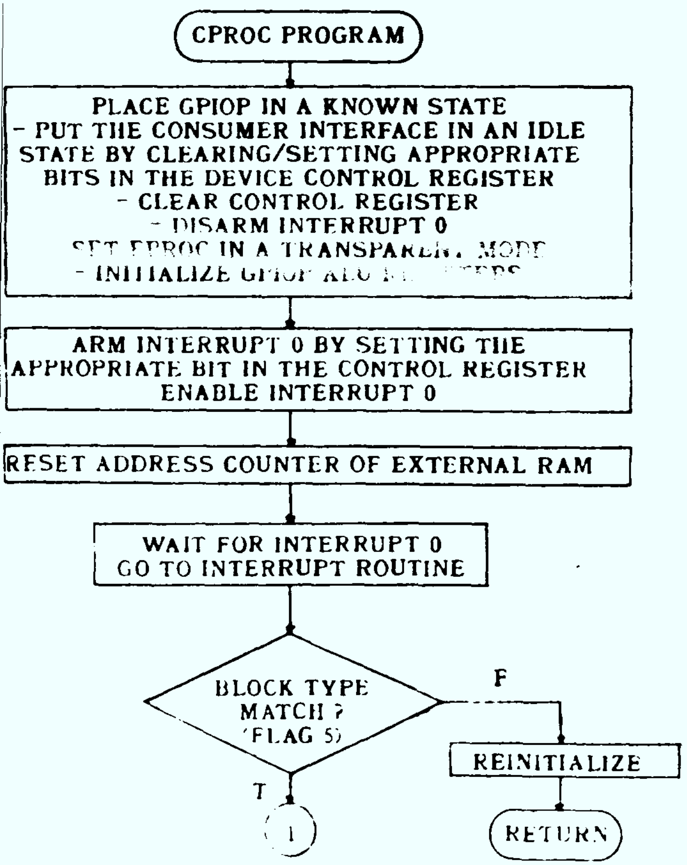

START CHA
SAVE THE INPUT PARAMETERS
2
CREATE AN ARRAY OF INCREASING HEIGHTS ROUTINE: SETHV
ENTER THE CHA DATA BLOCK ROUTINE: INDATA
CALCULATION OF VARIABLES USED IN CHA ROUTINE: VSTUP
GET THE FREQUENCY FLAG AND ADJUST THE ADDRESS OF THE CHA BLOCK DATA ACCORDING TO IT
NORMALIZE THE MEASURED CHA DATA TO VALUE OF HORN NO.5 FROM THE TOP ROUTINE: CVNRM
DERIVATION OF THE POINT OF REFLECTION ON A CURVED SURFACE ROUTINES: VR,SVDIV,VR1
DETERMINATION OF THE PATH LENGTH DIFFERENCE AND ANGLE-OF-ARRIVAL FOR THE DIRECT AND INDIRECT SIGNALS ROUTINES:V2R,VADD,VDIV,VSQRT
CALCULATION OF DIVERGENCE FACTOR AND REFLECTION COEFFICIENT AS MODIFIED BY A ROUGH SURFACE ROUTINES:VSIN,VSMUL,VSRF,VCR1 VDIV,CVEXP,CVMUL
CALCULATE D2PI•ELVD AND D2PI•ELVI FOR THE FIRST FOUR HORNS ROUTINES: VELV,CVEXP
CALCULATE THE THEORETICAL VALUES AT HORN NO.5 FOR THE COMPLETE HEIGHT WINDOW AND INVERSE ROUTINES: VSADD,VMOV,CVRCIP
CALCULATE THE THEORETICAL NORMALIZED VALUES FOR EACH HORN ROUTINE: VTHEO
CALCULATE THE REAL PART OF THE COMPLEX CORRELATION COEFFICIENT ROUTINES: VCOR,VSQRT,VDIV
EXCLUDE VALUES BELOW THE CORRELATION THRESHOLD ROUTINE: VICLIP
1

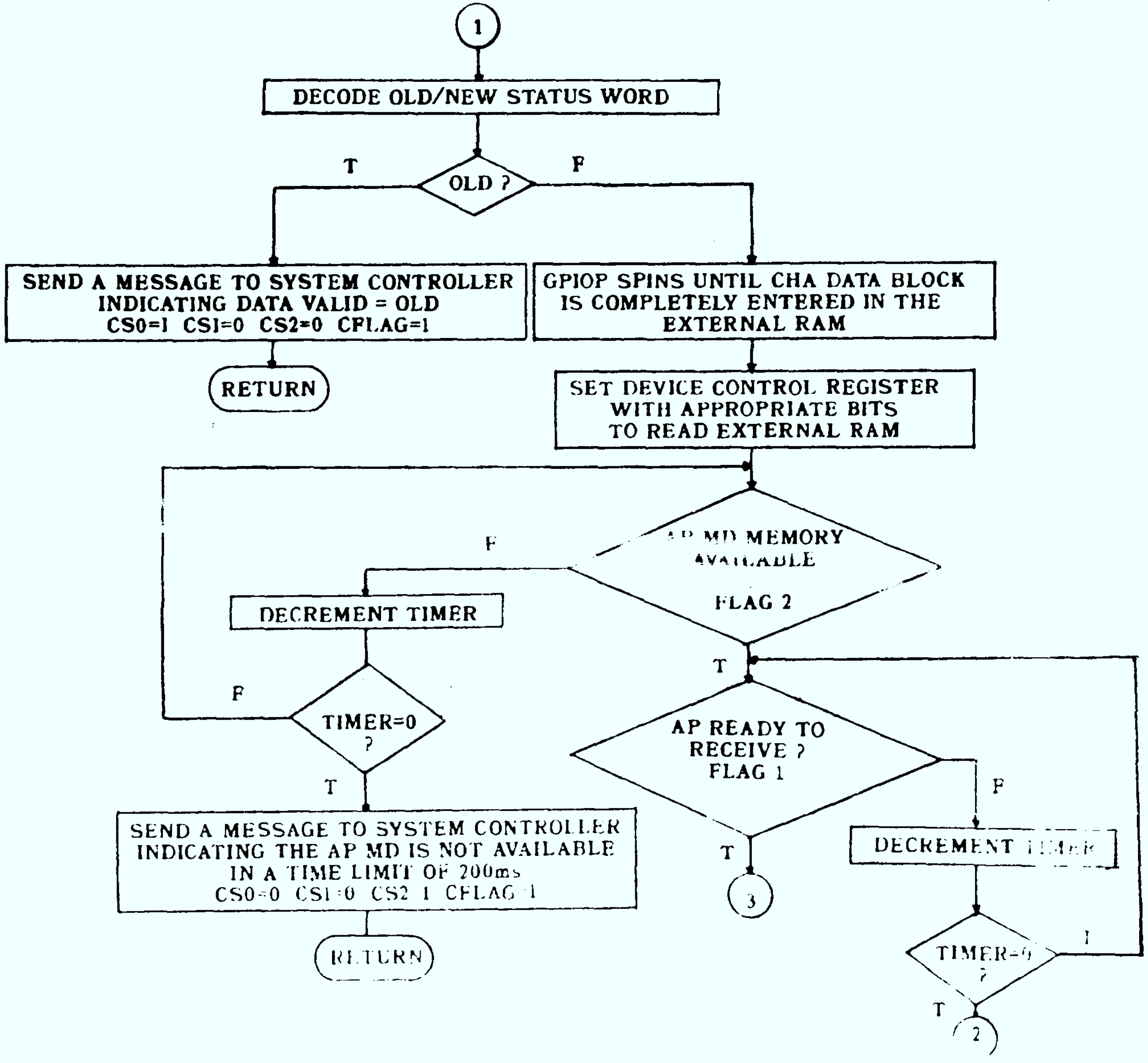

1
FIND THE CORRELATED HEIGHTS AND THEIR NUMBER (NPEAKS) ROUTINE: PKC
SAVE NPEAKS
FIRST PASS ?
T
F
TRACK LOST ?
T
F
DETERMINATION OF ABSOLUTE TRACK DEVIATIONS ROUTINE: TDEV
TRACKS ?
T
3
F
4
INITIALIZATION OF TRACKS ROUTINE:FPASS
CLEAR THE TARGET DEVIATION VECTOR ROUTINE:VCLR
UPDATE THE FOLLOWING FLAGS:
SETLD=0
FPASS=0
DELAY=0
LOST=0
SAVE RESULTS
2

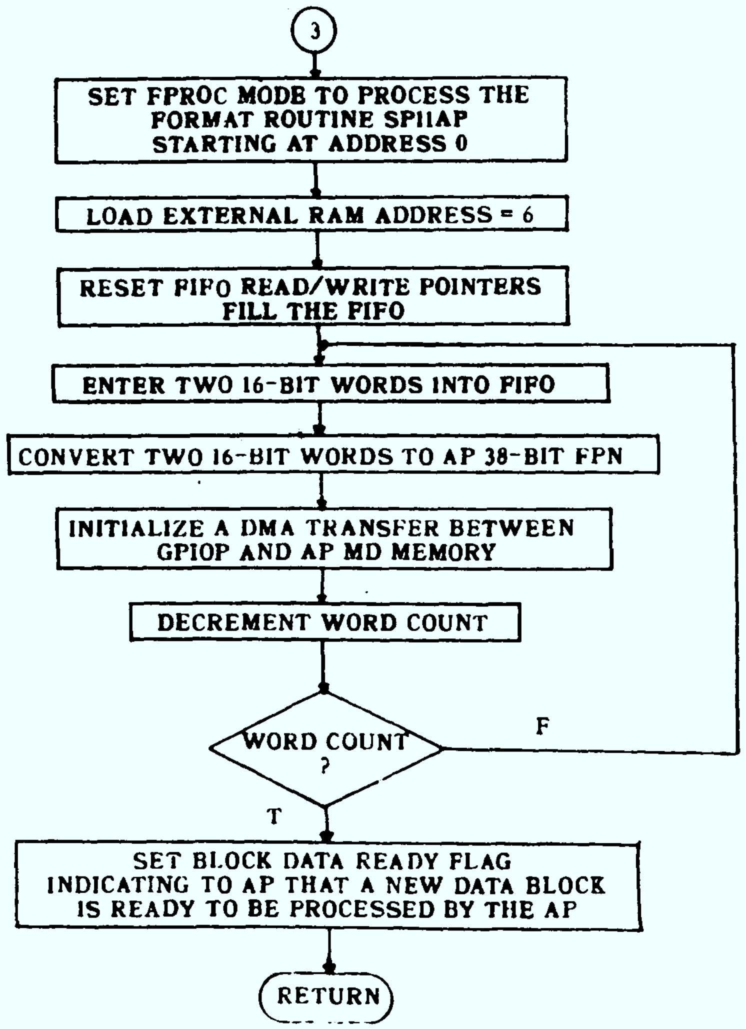

3
SAVE THE NUMBER OF TRACKS MAX
TRACK SETLD ?
T
F
TRACK ASSOCIATION ROUTINE: VSLC
IF TRACK SETTLED SET SETLD=1
SET MINIMUM HEIGHT MAXIMUM HEIGHT EXTENSION
DETERMINE THE TARGET HEIGHT ROUTINE: VSTL
SAVE RESULTS: RANGE AND TARGET HEIGHT
2

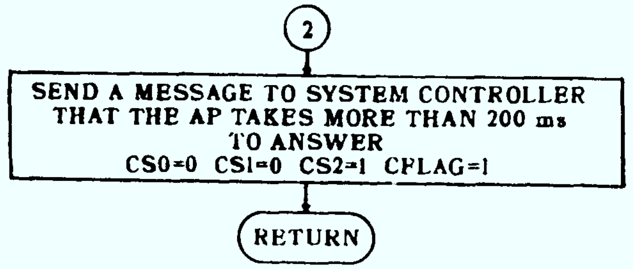

4
SETLD=1 ?
T
F
DELAY≥6 ?
T
F
LOST=1
SET:
MINIMUM HEIGHT
MAXIMUM HEIGHT
EXTENSION
2
SET:
MINIMUM HEIGHT=PREVIOUS HEIGHT
MAXIMUM HEIGHT=PREVIOUS HEIGHT
EXTENSION=40.
DELAY=DELAY+1
2

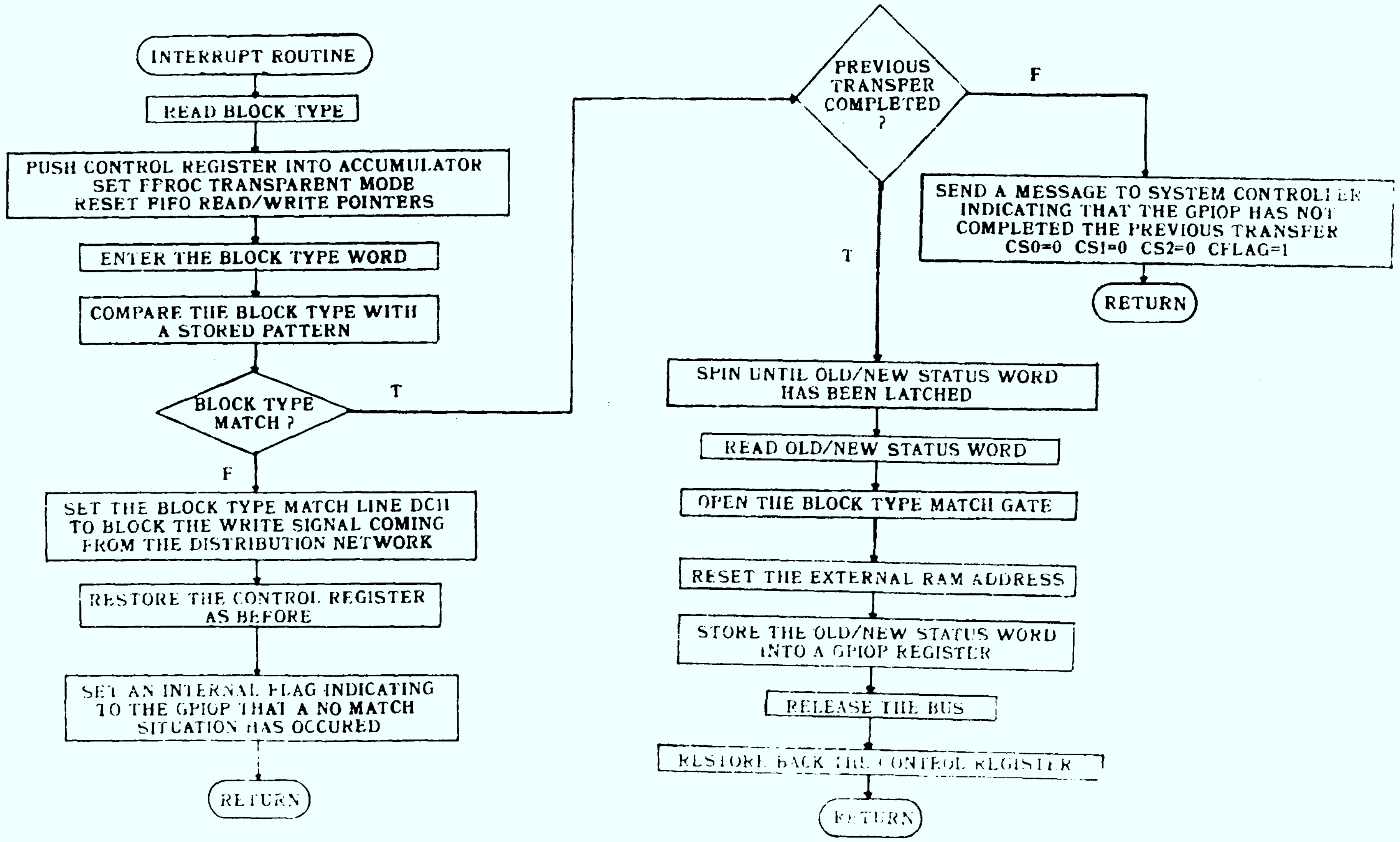
GPIOP PROGRAM FLOWCHART


CPROC PROGRAM
PLACE GPIOP IN A KNOWN STATE
- PUT THE CONSUMER INTERFACE IN AN IDLE STATE BY CLEARING/SETTING APPROPRIATE BITS IN THE DEVICE CONTROL REGISTER
- CLEAR CONTROL REGISTER
- DISARM INTERRUPT 0
SET EPROC IN A TRANSPARENT MODE
- INITIALIZE GPIOP ALU REGISTERS
ARM INTERRUPT 0 BY SETTING THE APPROPRIATE BIT IN THE CONTROL REGISTER ENABLE INTERRUPT 0
RESET ADDRESS COUNTER OF EXTERNAL RAM
WAIT FOR INTERRUPT 0 GO TO INTERRUPT ROUTINE
BLOCK TYPE MATCH ? (FLAG 5)
F
REINITIALIZE
RETURN
T
1


1
DECODE OLD/NEW STATUS WORD
OLD ?
T
F
SEND A MESSAGE TO SYSTEM CONTROLLER INDICATING DATA VALID = OLD CS0=1 CS1=0 CS2=0 CFLAG=1
RETURN
GPIOP SPINS UNTIL CHA DATA BLOCK IS COMPLETELY ENTERED IN THE EXTERNAL RAM
SET DEVICE CONTROL REGISTER WITH APPROPRIATE BITS TO READ EXTERNAL RAM
AP MD MEMORY AVAILABLE FLAG 2
F
DECREMENT TIMER
TIMER=0 ?
F
T
SEND A MESSAGE TO SYSTEM CONTROLLER INDICATING THE AP MD IS NOT AVAILABLE IN A TIME LIMIT OF 200ms CS0=0 CS1=0 CS2=1 CFLAG=1
RETURN
T
AP READY TO RECEIVE ? FLAG 1
F
DECREMENT TIMER
TIMER=0 ?
F
T
2
T
3


3
SET FPROC MODE TO PROCESS THE FORMAT ROUTINE SP11AP STARTING AT ADDRESS 0
LOAD EXTERNAL RAM ADDRESS = 6
RESET FIFO READ/WRITE POINTERS FILL THE FIFO
ENTER TWO 16-BIT WORDS INTO FIFO
CONVERT TWO 16-BIT WORDS TO AP 38-BIT FPN
INITIALIZE A DMA TRANSFER BETWEEN GPIOP AND AP MD MEMORY
DECREMENT WORD COUNT
WORD COUNT ?
F
T
SET BLOCK DATA READY FLAG INDICATING TO AP THAT A NEW DATA BLOCK IS READY TO BE PROCESSED BY THE AP
RETURN


2
SEND A MESSAGE TO SYSTEM CONTROLLER THAT THE AP TAKES MORE THAN 200 ms TO ANSWER
CS0=0 CS1=0 CS2=1 CFLAG=1
RETURN


INTERRUPT ROUTINE
READ BLOCK TYPE
PUSH CONTROL REGISTER INTO ACCUMULATOR SET FPROC TRANSPARENT MODE RESET FIFO READ/WRITE POINTERS
ENTER THE BLOCK TYPE WORD
COMPARE THE BLOCK TYPE WITH A STORED PATTERN
BLOCK TYPE MATCH ?
T
F
SET THE BLOCK TYPE MATCH LINE DCH TO BLOCK THE WRITE SIGNAL COMING FROM THE DISTRIBUTION NETWORK
RESTORE THE CONTROL REGISTER AS BEFORE
SET AN INTERNAL FLAG INDICATING TO THE GPIOP THAT A NO MATCH SITUATION HAS OCCURED
RETURN
PREVIOUS TRANSFER COMPLETED ?
F
T
SEND A MESSAGE TO SYSTEM CONTROLLER INDICATING THAT THE GPIOP HAS NOT COMPLETED THE PREVIOUS TRANSFER CS0=0 CS1=0 CS2=0 CFLAG=1
RETURN
SPIN UNTIL OLD/NEW STATUS WORD HAS BEEN LATCHED
READ OLD/NEW STATUS WORD
OPEN THE BLOCK TYPE MATCH GATE
RESET THE EXTERNAL RAM ADDRESS
STORE THE OLD/NEW STATUS WORD INTO A GPIOP REGISTER
RELEASE THE BUS
RESTORE BACK THE CONTROL REGISTER
RETURN

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM
(highest classification of Title, Abstract, Keywords)

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Establishment sponsoring a contractor's report, or tasking agency, are entered in section 8.)
Communications Research Centre
3701 Carling Ave., P.O. Box 11490, Station H
Ottawa, Ontario, K2H 8S2

2. SECURITY CLASSIFICATION (overall security classification of the document, including special warning terms if applicable)
UNCLASSIFIED

3. TITLE (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S,C,R or U) in parentheses after the title.)
REAL-TIME IMPLEMENTATION OF THE CHA ALGORITHM USING AN ARRAY PROCESSOR

4. AUTHORS (Last name, first name, middle initial. If military, show rank, e.g. Doe, Maj. John E.)
BOSSE, E

5. DATE OF PUBLICATION (month and year of publication of document)
July 1987

6a. NO. OF PAGES (total containing information. Include Annexes, Appendices, etc.)
36

6b. NO. OF REFS (total cited in document)
23

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)
CRC Report 1422

8. SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development. Include the address.)
Defence Research Establishment Ottawa
Department of National Defence
Ottawa, Ontario, K1A 0Z4

9a. PROJECT OR GRANT NO. (if appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant)
011LA13

9b. CONTRACT NO. (if appropriate, the applicable number under which the document was written)

10a. ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity. This number must be unique to this document.)
CRC Report 1422

10b. OTHER DOCUMENT NOS. (Any other numbers which may be assigned this document either by the originator or by the sponsor)

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

- (X) Unlimited distribution
- ( ) Distribution limited to defence departments and defence contractors; further distribution only as approved
- ( ) Distribution limited to defence departments and Canadian defence contractors; further distribution only as approved
- ( ) Distribution limited to government departments and agencies; further distribution only as approved
- ( ) Distribution limited to defence departments; further distribution only as approved
- ( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availabilty (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.)

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

Presented in this report, is a study of the implementation of the CHA algorithm on the AP-120B array processor. The study had two main objectives: (1) to investigate the AP-120B's real-time capabilities and (2) to determine the digital noise generated by the AP-120B due to round-off errors. The execution time that was sought for the array processor was 20 msec per data sample. It was found that a processing time of 13 msec could be achieved if no communications were required between the AP-120B and its host computer. This value quickly grew to 85 msec if interaction with the host computer was called for. To lessen interaction with the host, a GPIOP (General-purpose Programmable Input Output processor) was used to interact between the AP and external devices. Also, it was found that the digital noise generated by the AP-120B was negligible when compared to typical radar signal noise.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible, keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

CHA Algorithm
Array Processor
Real-Time

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM