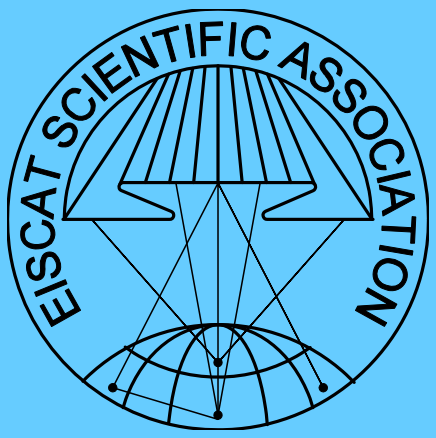
81/24



EISCAT
TECHNICAL
NOTES

SCIENTIFIC PROGRAMMING OF THE EISCAT DIGITAL CORRELATOR

(REVISED)

TERRANCE HO and HANS-JØRGEN ALKER

KIRUNA
Sweden

SCIENTIFIC PROGRAMMING OF THE EISCAT DIGITAL CORRELATOR (REVISED)

by

TERRANCE HO* AND HANS-JØRGEN ALKER**

*MAX PLANCK INSTITUT
POSTFACH 20,
D-3411 KATLENBURG-LINDAU 3
W. GERMANY

**ELAB/SINTEF
N-7034 TRONDHEIM-NTH
NORWAY

EISCAT Technical Note 81/24
Printed in Sweden
EISCAT Scientific Association
Kiruna, March 1981
ISSN 0349-2710

TABLE OF CONTENTS

ABSTRACT

This report describes the basic philosophy of how to create special-purpose microprograms for the EISCAT digital correlator. A structure of microprogram units (main programs/subroutines) are introduced and covers the main real-time data processing of the radar system. The program units have been tested on the prototype of the EISCAT correlator.

I. INTRODUCTION

During 1977/78 a prototype of the EISCAT correlator has been designed and built. The unit will be a part of the radar system on-line data handling. The correlator is a real-time, digital processor specially designed for the data reduction before the computer data analysis. The correlator is microprogrammable and can by computer control execute different processing schemes, either by separate programs or in mixed mode by subroutine processing.

The purpose of this report is to explain the philosophy of the programming structure of the digital correlator and to give a proposal for the special-purpose real-time programming required for the radar processing. Step by step instructions of how to design a microprogram from the flow chart with consideration to the selection of locations in the APB, APM processors and the flip flops (FF1, FF2) on the accumulators will be given. A breakdown of a simple Single Pulse Autocorrelation program (ie calculation of noise and calibration samples and transfer program are not given) will be used as an example. This report is primarly intended for the programmer and only basic knowledge of complex numbers and / notation will be assumed.

Descriptions of all the programs designed to date will be given. At a later date a complete library of program subroutines with descriptions will be documented and can be used as a reference when designing more complicated algorithms.

It is assumed that the user has some familiarity with the basic instruction-set of the correlator which can be found in the report: "Instruction Manual for EISCAT Digital correlator (Revised)", the report: A Programmable Correlator Module for the EISCAT Radar System" and also the report: "Programs CORRSIM , CORRTEST: System for Program Development and Software Simulation of EISCAT Digital Correlator. Users Manual"

II. PHILOSOPHY OF THE PROGRAMMING STRUCTURE.

The concept of the software system is such that a program for an experiment can be built up from a main program and a set of subroutines which are the algorithms for the different schemes that are available (ie. Power Profile, Single Pulse etc.) from the CORRSIM files PROG0, PROG1 etc. (read access only). Thus for a given experiment it is only necessary for the user to write the main program and then to link it with the subroutines on file directly. The CORRSIM program has the facility such that a subroutine can be read into the correlator memory to locations relative to those defined in the subroutine so there is no difficulty in placing a subroutine into a given part of the correlator memory. The PROMS(programmable read only memory) of the correlator will contain several fixed programs (ie. main programs and subroutines) and it is anticipated, at least in the beginning, that the necessary programs will be contained in the PROM to suit the needs of an experimenter. (see REF.2. Fig. 1)

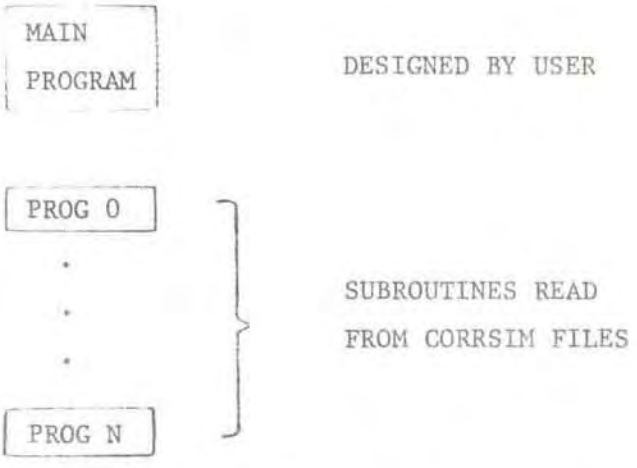


FIGURE 1. COMPOSITION OF PROGRAM STRUCTURE

With this proposed program structure a high degree of flexibility is maintained for covering changes in the radar processing. The main program (user defined) has the necessary link references to the subroutines for execution of a given radar experiment. By changing the link references only in the main program new processing schemes can be easily made. The proposed structure is optimum when considering the programming work and time involved for reloading the correlator when changing the radar experiment. An extension for multi-channel input processing can also be made by a rerun of the program on different data-sets in the buffer-memory.

III. NECESSARY FORMULAE FOR THE VARIOUS SUBROUTINES:

Here explanations of the formulae used for the different algorithms, when possible in a non-mathematical way, will be given.

In the radar processing, parameter estimations are based on signal-averaging (time averaging of data from a particular scattering volume) and the correlator has internal accumulators/memory for performing the data integration. In this chapter the necessary typical algorithms for the computation of a single contribution to this average are explained. For the explanation of the integration over these single contributions, see Chapter IV.3.

The definitions of the formulae are given with respect to the way in which the X,Y samples are read from the buffer memory.

III.1 POWER PROFILE (ZERO LAG ESTIMATION)

Assume that we have a set of N complex samples Z_0, Z_1, Z_2..... Z_{N-1} in a rangecell, where $Z_0 = X_0 + iY_0$ etc and there are r = 1 ... M rangecells Then the zero lag estimate for the rth rangecell K_r is defined as:

$$K_r = \sum_{j=0}^{N-1} Z_{j+(N+D-1)(r-1)} \; Z^*_{j+(N+D-1)(r-1)}$$

$$= \sum_{j=0}^{N-1} (X_{j+(N+D-1)(r-1)} + iY_{j+(N+D-1)(r-1)})(X_{j+(N+D-1)(r-1)} - iY_{j+(N+D-1)(r-1)})$$

$$= \sum_{j=0}^{N-1} (X^2_{j+(N+D-1)(r-1)} + Y^2_{j+(N+D-1)(r-1)})$$

$$+ i \sum_{j=0}^{N-1} (X_{j+(N+D-1)(r-1)} Y_{j+(N+D-1)(r-1)} - Y_{j+(N+D-1)(r-1)} X_{j+(N+D-1)(r-1)})$$

$$= \sum_{j=0}^{N-1} (X^2_{j+(N+D-1)(r-1)} + X^2_{j+(N+D-1)(r-1)})$$

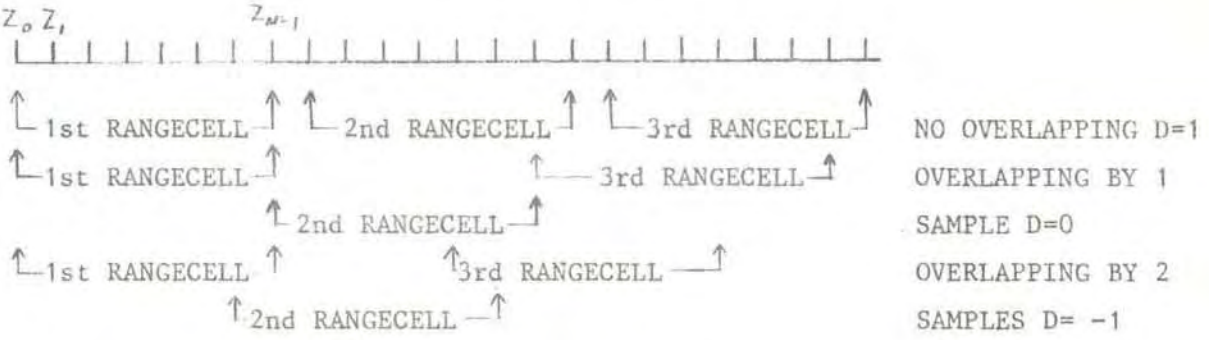
where * denotes the conjugate and D is the overlap factor between the rangecells such that for D = 1 there is no overlapping of rangecells and D ≦ 0 there is overlapping of rangecells. See figure below for illustration. Note that when having overlapping the last sample in the Mth rangecell which is used in the algorithm must always be equal to the last sample written in the buffer memory for that data set. eq. For 13 samples in a rangecell and 27 rangecells with an overlap factor of -4 the index for the last sample in the 27th rangecell would be:

Index for last sample = 12+(13-4-1)(26)
= 220

Therefore there must be exactly 221 samples in the buffer memory for this data set. (Index goes from 0-220)

These values are computed in DATA CHANNEL 1 of the correlator (see Ref. 3 ARI/ACC INSTRUCTIONS). As the imaginary part is zero we compute the MEAN VALUE ESTIMATION M_r in DATA CHANNEL 2.

where $$M_r = \sum_{j=0}^{N-1} \left(X_{j+(N+D-1)(r-1)} + Y_{j+(N+D-1)(r-1)}\right)$$



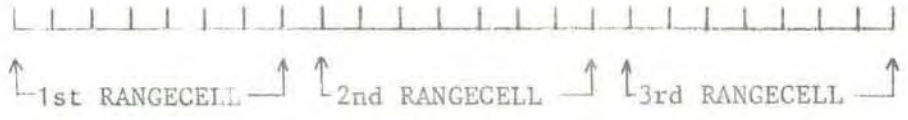
SCHEME FOR OVERLAPPING OF RANGECELLS.

Note that strictly speaking the formulae for K_r and M_r should be defined as:

$$K_r = \frac{1}{N_r} \sum_{j=0}^{N-1} Z_{j+(N+D-1)(r-1)} \; Z^*_{j+(N+D-1)(r-1)}$$

and $$M_r = \frac{1}{N_r} \sum_{j=0}^{N-1} X_{j+(N+D-1)(r-1)} + Y_{j+(N+D-1)(r-1)}$$

However the $1/N_r$ normalization factor is computed in the post processing of the data by the computer and therefore will not be included in any of the forthcoming formulae.



CONSIDER AS SAMPLES IN THE BUFFER MEMORY.

Looking at the formulae pictorially, with D=1 we have:

$$K_1 = Z_0^2 + Z_1^2 + Z_2^2 + \ldots + Z_{N-1}^2$$

$$K_2 = Z_N^2 + Z_{N+1}^2 + Z_{N+2}^2 + \ldots + Z_{2N-1}$$

etc.

$$M_1 = Z_o + Z_1 + \ldots . Z_{N-1}$$

$$M_2 = Z_N + Z_{N+1} \ldots . Z_{2N-1}$$

etc.

Here I have written Z_0^2 instead of $Z_0 Z_0^*$ however, what is important when considering the formulae pictorially is only to think of which locations from the buffer memory we take the X and Y samples. Therefore Z_0^2 would mean that we are using the same X and Y Sample twice from location 0 in the buffer memory. This applies in all forthcoming explanations.

III.2 AUTOCORRELATION FOR SINGLE PULSE TRANSMISSIONS (SINGLE PULSE) ALGORITHM

Assume that we have a set of N complex samples Z_0, Z_1 ... Z_{N-1} in a range-cell, where $Z_0 = X_0 + iY_0$ etc and there are r = 1 ... M rangecells,

then

$$K_{L,r} = \sum_{j=0}^{N-L-1} Z_{j+(N+D-1)(r-1)} \; Z^*_{j+L+(N+D-1)(r-1)}$$

$$= \sum_{j=0}^{N-L-1} (X_{j+(N+D-1)(r-1)} + iY_{j+(N+D-1)(r-1)})(X_{j+L+(N+D-1)(r-1)} - iY_{j+L+(N+D-1)(r-1)})$$

$$= \sum_{j=0}^{N-L-1} (X_{j+(N+D-1)(r-1)} \; X_{j+L+(N+D-1)(r-1)} + Y_{j+(N+D-1)(r-1)} Y_{j+L+(N+D-1)(r-1)})$$

$$+ i \sum_{j=0}^{N-L-1} (X_{j+L+(N+D-1)(r-1)} Y_{j+(N+D-1)(r-1)} - X_{j+(N+D-1)(r-1)} Y_{j+L+(N+D-1)(r-1)})$$

where * denotes the conjugate and D is the overlap factor between rangecells, such that for D = 1 there is no overlapping of rangecells and D ≤ 0 for overlapping of rangecells (see figure in III.1 for further details) and L is the lag parameter, such that L = 0,1 ... N-1. Note that when having overlapping the last sample in the Mth rangecell which is used in the algorithm must always be equal to the last sample written in the buffer memory for that data set. e.g. For 13 samples with 13 lags in a rangecell and 27 rangecells with an overlap. Factor of -4 the index for the last sample in the 27th rangecell would be:

For zero lag, last sample Index = 12+(13-4-1)(26)
= 220

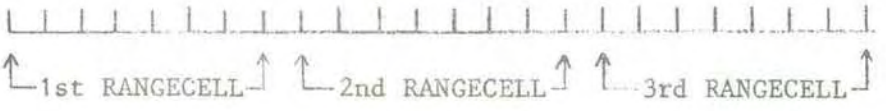
or for lag 12, last sample Index = 0+12+(13-4-1)(26)
= 220

Therefore there must be exactly 221 samples in the buffer memory for this data set. (Index goes from 0-220)

The Re $\{K_{L,r}\}$ is computed in DATA CHANNEL 1

The Im $\{K_{L,r}\}$ is computed in DATA CHANNEL 2.

Pictorially with D = 1 we have:



CONSIDER AS SAMPLES IN BUFFER MEMORY

$$K_{0,1} = Z_0^2 + Z_1^2 + Z_2^2 + \ldots Z_{N-1}^2$$

$$K_{1,1} = Z_0 Z_1 + Z_1 Z_2 + Z_2 Z_3 + \ldots$$

$$K_{2,1} = Z_0 Z_2 + Z_1 Z_1 Z_3 + \ldots$$

$$\vdots$$

$$K_{N-1,1} = Z_0 Z_{N-1}$$

$$K_{0,2} = Z_N^2 + Z_{N+1}^2 + Z_{N+2}^2 + \ldots + Z_{2N-1}^2$$

$$K_{1,2} = Z_N Z_{N+1} + Z_{N+1} Z_{N+2} + Z_{N+2} Z_{N+3}$$

$$K_{2,2} = Z_N Z_{N+2} + Z_{N+1} Z_{N+3} +$$

$$\vdots$$

$$K_{n-1,2} = Z_N Z_{2N-1}$$

etc.

There will also be a truncated version of this algorithm whereby the lag parameter L can be less than or equal to N-1

then $$K_{L,r} = \sum_{j=0}^{N-L-1} Z_{j+(N+D-1)(r-1)} \, Z^*_{j+L+(N+D-1)(r-1)}$$

where $L = 0, 1, \ldots P \qquad P \leq N-1$

and pictorially with D = 1 we have:

$$K_{0,1} = Z_0^2 + Z_1^2 + Z_2^2 + \ldots \quad + Z_{N-1}^2$$

$$K_{1,1} = Z_0Z_1 + Z_1Z_2 + Z_2Z_3 + ..$$

$$K_{2,1} = Z_0Z_2 + Z_1Z_3$$

$$\vdots$$

$$K_{P,1} = Z_0Z_p + Z_1Z_{P+1} + Z_2Z_{p+2} + \ldots$$

$$K_{0,2} = Z_N^2 + Z_{N+1}^2 + Z_{N+2}^2 + \ldots + Z_{2N-1}^2$$

$$K_{1,2} = Z_NZ_{N+1} + Z_{N+1}Z_{N+2} + Z_{N+2}Z_{N+3} + \ldots$$

$$K_{2,2} = Z_NZ_{N+2} + Z_{N+1}Z_{N+3} \ldots$$

$$\vdots$$

$$K_{P,2} = Z_NZ_{N+P} + Z_{N+1}Z_{N+P+1} + Z_{N+2}Z_{N+P+2} +$$

etc.

III.3 AUTOCORRELATION FOR MULTIPLE PULSE TRANSMISSIONS (MULTIPULSE ALGORITHM

Assume that we have a set of P complex samples Z_0, Z_1, Z_2, ... Z_{P-1} where $Z_0 = X_0 + iY_0$ etc, N pulses in the pulse group and r = 1,2, .. M range-cells. The various lag products from the same height can be expressed as:

$$K_{L,r} = Z_{S+r-1} Z^*_{S+L+r-1}$$

$$= (X_{S+r-1} + iY_{S+r-1})(X_{S+L+r-1} - iY_{S+L+r-1})$$

$$= (X_{S+r-1}X_{S+L+r-1} + Y_{S+r-1}Y_{S+L-r-1})$$

$$+i(X_{S+L+r-1}Y_{S+r-1} - X_{S+r-1}Y_{S+L+r-1})$$

where * denotes the conjugate.

Let J_0 = Position of 1st pulse

J_1 = Sample difference between 1st and 2nd pulse

J_2 = Sample difference between 1st and 3rd pulse

.

.

.

J_{N-1} = Sample difference between 1st and Nth pulse

then $S = J_0, J_1, J_2 \ldots J_{N-2}$

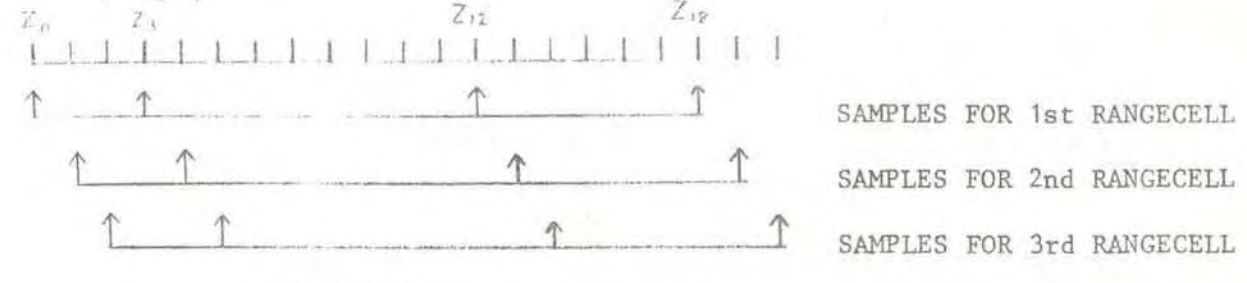
and $L = J_1 - S, \ldots J_{N-1} - S$ with the restriction $L > 0$

In this scheme the order of the lags computed are scrambled and would have to be unscrambled after the computation. The total number of lags computed for N pulses is given by:

No. of lags computed: $N(N-1)/2$

The Re $\{K_{L,r}\}$ is computed in DATA CHANNEL 1

The Im $\{K_{L,r}\}$ is computed in DATA CHANNEL 2.



CONSIDER AS SAMPLES IN THE BUFFER MEMORY

Assume that we have four pulses each having 3 samples in each pulse at Z_0, Z_3, Z_{12}, Z_{18} corresponds to the first sample in each pulse, then pictorially what is computed sequentially is:

$$K_{3,1} = Z_0 Z_3$$

$$K_{12,1} = Z_0 Z_{12}$$

$$K_{18,1} = Z_0 Z_{18}$$

$$K_{9,1} = Z_3 Z_{12}$$

$$K_{15,1} = Z_3 Z_{18}$$

$$K_{6,1} = Z_{12} Z_{18}$$

$$K_{3,2} = Z_1 Z_4$$

$$K_{12,2} = Z_1 Z_{13}$$

$$K_{18,2} = Z_1 Z_{19}$$

$$K_{9,2} = Z_4 Z_{13}$$

$$K_{15,2} = Z_4 Z_{19}$$

$$K_{6,2} = Z_{13} Z_{19}$$

etc.

III.4 CROSS CORRELATION BETWEEN TWO SETS OF DATA

Assume that we have two sets of N complex samples Z_0, Z_1, ... Z_{N-1} and Z_S, Z_{S+1}, ... Z_{S+N-1} in a rangecell, where $Z_0 = X_0 + iY_0$ and $Z_S = X_S + iY_S$ etc, S = sample difference between the two sets of data and there are r = 1 ... M rangecells.

Then

$$K_{L,r} = \sum_{j=0}^{N-L-1} Z_{j+(N+D-1)(r-1)} \, Z^*_{j+S+L+(N+D-1)(r-1)}$$

$$= \sum_{j=0}^{N-L-1} (X_{j+(N+D-1)(r-1)} + iY_{j+(N+D-1)(r-1)})(X_{j+S+L+(N+D-1)(r-1)} - iY_{j+S+L+(N+D-1)(r-1)})$$

$$= \sum_{j=0}^{N-L-1} (X_{j+(N+D-1)(r-1)} X_{j+S+L+(N+D-1)(r-1)} + Y_{j+(N+D-1)(r-1)} Y_{j+S+L+(N+D-1)(r-1)}$$

$$+ i \sum_{j=0}^{N-L-1} (X_{j+S+L+(N+D-1)(r-1)} Y_{j+(N+D-1)(r-1)} - X_{j+(N+D-1)(r-1)} Y_{j+S+L+(N+D-1)(r-1)})$$

where * denotes the conjugate and D is the overlap factor between rangecells, such that for D=1 there is no overlapping of rangecells and D ≤ 0 for overlapping of rangecells (see Figure in III.1 for further details) and L is the lag parameter, such that L=0,1 ... N-1. When having overlapping of rangecells the last sample in the Mth rangecell, for both data sets, which is used in the algorithm must always equal the last sample written in the buffer memory for both data sets. E.g. for 13 samples in a rangecell and 27 rangecells with an overlap factor of -4 the index for the last sample in the 27th rangecell would be:

For zero lag, last sample index of 1st Data Set = 12+(13-4-1)(26)
= 220

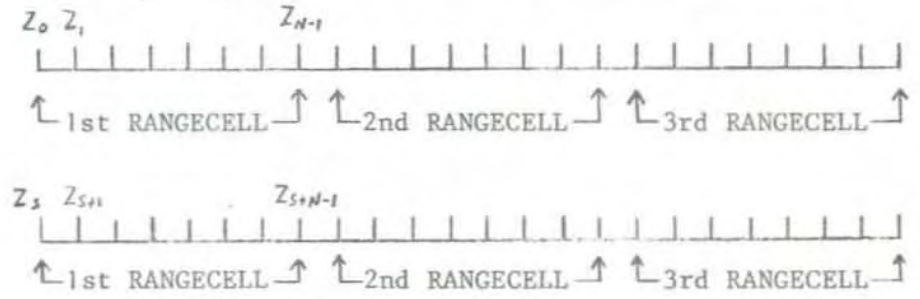
For zero lag, last sample index of 2nd Data Set = 12+S+(13-4-1)(26)
= 220+S

Therefore there must be exactly 221 samples in the buffer memory for both data sets (Index goes from 0-220 for 1st data set and S-(220+S) for 2nd data set). Thus for this example S ≥ 221.

It must be noted that with this algorithm only half of the correlation function is obtained and that if S=0 we have the Single Pulse autocorrelation algorithm.

The $\mathrm{Re}\left\{K_{L,r}\right\}$ is computed in DATA CHANNEL 1

The $\mathrm{Im}\left\{K_{L,r}\right\}$ is computed in DATA CHANNEL 2



CONSIDER AS SAMPLES IN THE BUFFER MEMORY

Pictorially with D=1 we have:

$$K_{0,1} = Z_0 Z_S + Z_1 Z_{S+1} + Z_2 Z_{S+2} + \cdots + Z_{N-1} Z_{S+N-1}$$

$$K_{1,1} = Z_0 Z_{S+1} + Z_1 Z_{S+2} + Z_2 K_{S+3} + \cdots$$

$$K_{2,1} = Z_0 Z_{S+2} + Z_1 Z_{S+3} + \cdots$$

.

.

.

$$K_{N-1,2} = Z_0 Z_{S+N-1}$$

$$K_{0,2} = Z_N Z_{S+N} + Z_{N+1} Z_{S+N+1} + Z_{N+2} Z_{S+N+2} + \cdots + Z_{2N-1} Z_{S+2N-1}$$

$$K_{1,2} = Z_N Z_{S+N+1} + Z_{N+1} Z_{S+N+2} + Z_{N+2} Z_{S+N+3} + \cdots$$

$$K_{2,2} = Z_N Z_{S+N+2} + Z_{N+1} Z_{S+N+3} + \cdots$$

.

.

.

$$K_{N-1,2} = Z_N Z_{S+2N-1}$$

etc.

There will also be a truncated version of this algorithm whereby the lag parameter L can be less than or equal to N-1

then

$$K_{L,r} = \sum_{j=0}^{N-L-1} Z_{j+(N+D-1)(r-1)} \; Z^*_{j+S+L+(N+D-1)(r-1)}$$

where $L = 0, 1 \ldots P_1 \quad P \leqslant N-1$

and pictorially with D=1 we have

$$K_{0,1} = Z_0 Z_S + Z_1 Z_{S+1} + Z_2 Z_{S+2} + \quad + Z_{N-1} Z_{S+N-1}$$

$$K_{1,1} = Z_0 Z_{S+1} + Z_1 Z_{S+2} + Z_2 Z_{S+3} + \cdots$$

$$K_{2,1} = Z_0 Z_{S+2} + Z_1 Z_{S+3} + \cdots$$

$$\vdots$$

$$K_{P,1} = Z_0 Z_{S+P} + Z_1 Z_{S+P+1} + Z_2 Z_{S+P+2} + \cdots$$

$$K_{0,2} = Z_N Z_{S+N} + Z_{N+1} Z_{S+N+1} + Z_{N+2} Z_{S+N+2} + \cdots + Z_{2N-1} Z_{S+2N-1}$$

$$K_{1,2} = Z_N Z_{S+N+1} + Z_{N+1} Z_{S+N+2} + Z_{N+2} Z_{S+N+3} + \cdots$$

$$K_{2,2} = Z_N Z_{S+N+2} + Z_{N+1} Z_{S+N+3} + \cdots$$

$$\vdots$$

$$K_{P,2} = Z_N Z_{S+N+P} + Z_{N+1} Z_{S+N+P+1} + Z_{N+2} Z_{S+N+P+2} + \cdots$$

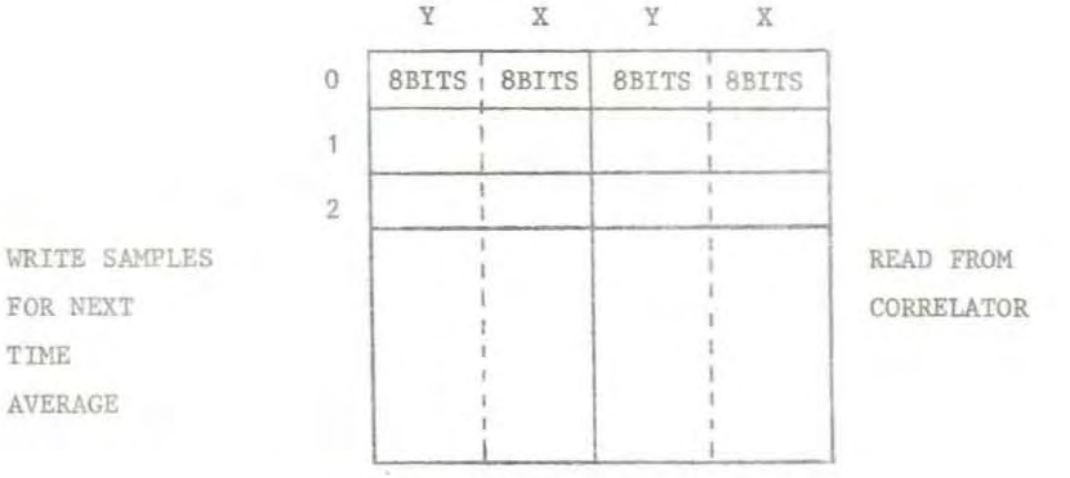
etc.

IV. PROGRAMMING CONSIDERATIONS

When designing a program it is essential that all values be written in decimal (integer coding). All program codes, Pc values and register stack values will be given in decimal. Ideally a program is optimally constructed when every microprogram instruction processes a data sample. However, this is not always possible and it should be borne in mind when designing a program that the number of dummy statements should be at a minimum otherwise the efficiency of the program will be reduced and the execution time unnecessarily increased.

IV. 1. BUFFER MEMORY AND RESULT MEMORY

The buffer memory is two ported which means that the memory can have READ/WRITE operations from the CORRELATOR/RADAR CONTROLLER performed on it simultaneously. The memory has 32 BIT words and is split into two sections each of 16 BITS; 8 BITS each for the X and Y samples (see Fig. 2)



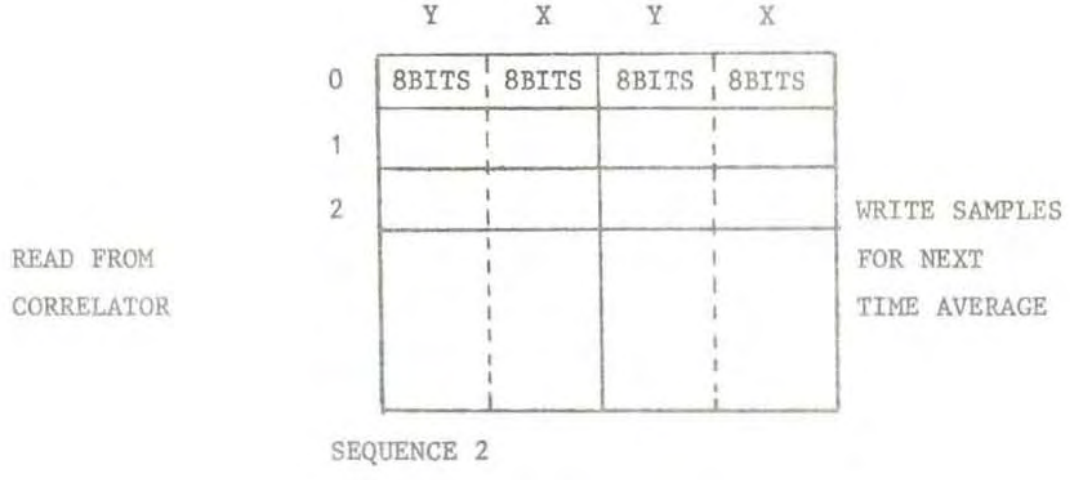


FIGURE 2. BUFFER MEMORY

When the correlator has computed one time average the READ/WRITE operations on the memory are then switched over, this is controlled by the radar controller, and the data for the next time average is then read by the correlator. (see Fig. 2).

The correlator has an internal result memory of 1K 1024 64 BIT words which can be expanded to 4 K 4096 64 BIT words. 32 BITS each are used for DATA CHANNEL1 and DATA CHANNEL2. This is equivalent to a double integer format in computer representation for each channel. (see Fig. 3)

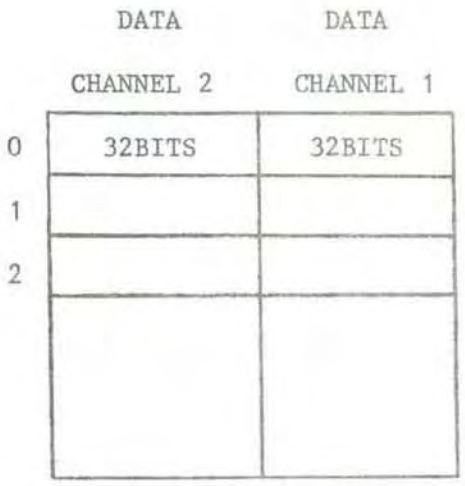


FIGURE 3. RESULT MEMORY

The buffer memory has a 16 BIT representation which means its numeric range is $0\text{-}177777_8$ = 0-65535 = 64K
The result memory has a 12 BIT representation which means its numeric range is $0\text{-}7777_8$ = 0-4095 = 4K
For further details see Ref1. page 30 and REF3, APB/APM INSTRUCTIONS.

The correlator modules delivered to each site will have internal result memories of 2K 2048 64 BIT words and at the present time no slaves will be built.

The buffer memories delivered to each site will have memory sizes of 4K 4096 32 BIT words (16 BITS for each half of the memory). For further details about the buffer memory see Ref.4.

IV. 2. SELECTION OF DATA VALUES FROM APB AND DATA COMPUTATION

In any one microprogram instruction only one location in the buffer memory can be selected and the X,Y sample be strobed into the A and B registers of the multipliers. Therefore taking the single pulse scheme as an illustration (see III.2)

$$K_0 = Z_0^2 + Z_1^2 + Z_2^2 + Z_3^2 + \ldots + Z_{N-1}^2$$

$$K_1 = Z_0Z_1 + Z_1Z_2 + Z_2Z_3 +$$

$$K_2 = Z_0Z_2 + Z_1Z_3 +$$

$$\vdots$$

$$K_{N-1} = Z_0Z_{N-1}$$

FIGURE 4. SCHEME FOR SINGLE PULSE ALGORITHM

In Fig. 4 the rangecell subscript r has been dropped in order to simplify the scheme. As can be seen from Fig. 4 lags K_1, K_2 ...K_{N-1} has two different samples multiplied together and it would be inefficient to program the correlator to compute lags K_0 first, K_2 second, ... K_{N-1} last, completely. Instead it is better to compute the lags in this way. First Z_0^2, Z_0Z_1, Z_0Z_2 ... Z_0Z_{N-1} then Z_1^2, Z_1Z_2, Z_1Z_3 ... Z_1Z_{N-1} etc. When following this scheme it can be seen that it is only necessary to strobe one new sample into the multipliers each time as the other sample is already contained in the multipliers and thus the method of computation becomes highly efficient. Programming the correlator by the first method would entail n(n+1) read operations from the buffer memory whereas by the second method it would only entail $\frac{n(n+1)}{2}$ read operations from the buffer memory.

IV. 3. FLIP FLOPS (FF1, FF2) ON ACCUMULATORS

At the beginning of an experiment the result memory is not cleared to zero. Therefore it is necessary to have a function (FF1) which has an independent control over the READ control of the result memory to the accumulators. Consider the scheme in Fig. 4 as an illustration where there are more than one rangecell to be computed. When the correlator receives the START COMPUTE signal from the radar controller FF1 is reset (i.e. $c_5 = 1$ $c_4 = 0$ in REF1 or CLEAR = 1 SET1 = 0 in REF3) and Z_0^2, Z_0Z_1, Z_0Z_2 ... Z_0Z_{N-1} is first computed. This means that the READ (=1) statement for the result memory is inhibited and the computed values are just written into the result memory (WRITE=1). Now when Z_1^2, Z_1Z_2, Z_1Z_3 ... Z_1Z_{N-1} etc. are computed it is now necessary to set FF1 (CLEAR1 = 0 SET1 = 1) and the READ statement is now enabled so that values from the result memory are then accumulated with the newly computed values and then written back into the result memory. When the next rangecell is to be computed FF1 has now to be reset as this is equivalent to a START COMPUTE for the next rangecell.

However, when the next time average is to be processed (i.e. correlator receives next START COMPUTE. signal from the radar controller) it is now necessary to have a higher priority flip flop (FF2) which controls FF1 such that FF1 control of READ is inhibited and all values from the result memory are read into the accumulators. Thus when FF2 is set ($C_7 = 0$ $C_6 = 1$) or CLEAR2 = 0 SET2 = 1) FF1 control is inhibited. The FF2 control can be set either through program control or by status word control. For program control see REF1, page 13 and REF3 ARI/ACC INSTRUCTIONS. For status word control see REF3, STATUS WORD, BIT 5. In the standard programs FF2 control will be set through program control. After a DMA transfer has been made it is necessary to reset FF2 ($C_7 = 1$ $C_6 = 0$ or CLEAR2 = 1 SET2 = 0) so that the program is again under FF1 control. This resetting is done in the Transfer program of the correlator.
Note that it is not allowed to have a READ/WRITE statement to the same memory location in the result memory in two consecutive clock cycles. This is due to the pipelining structure of the correlator.

V. DESCRIPTION OF SINGLE PULSE PROGRAM

Discussion: Typically there will be N samples in a rangecell, N lags to be computed, option for with or without overlapping rangecells. Let LCR1 = Parameter for the no. of samples which is also equal to the no. of lags in the rangecell and LCR2= Parameter for the no. of rangecells for time average. See III. 2. for the formula.

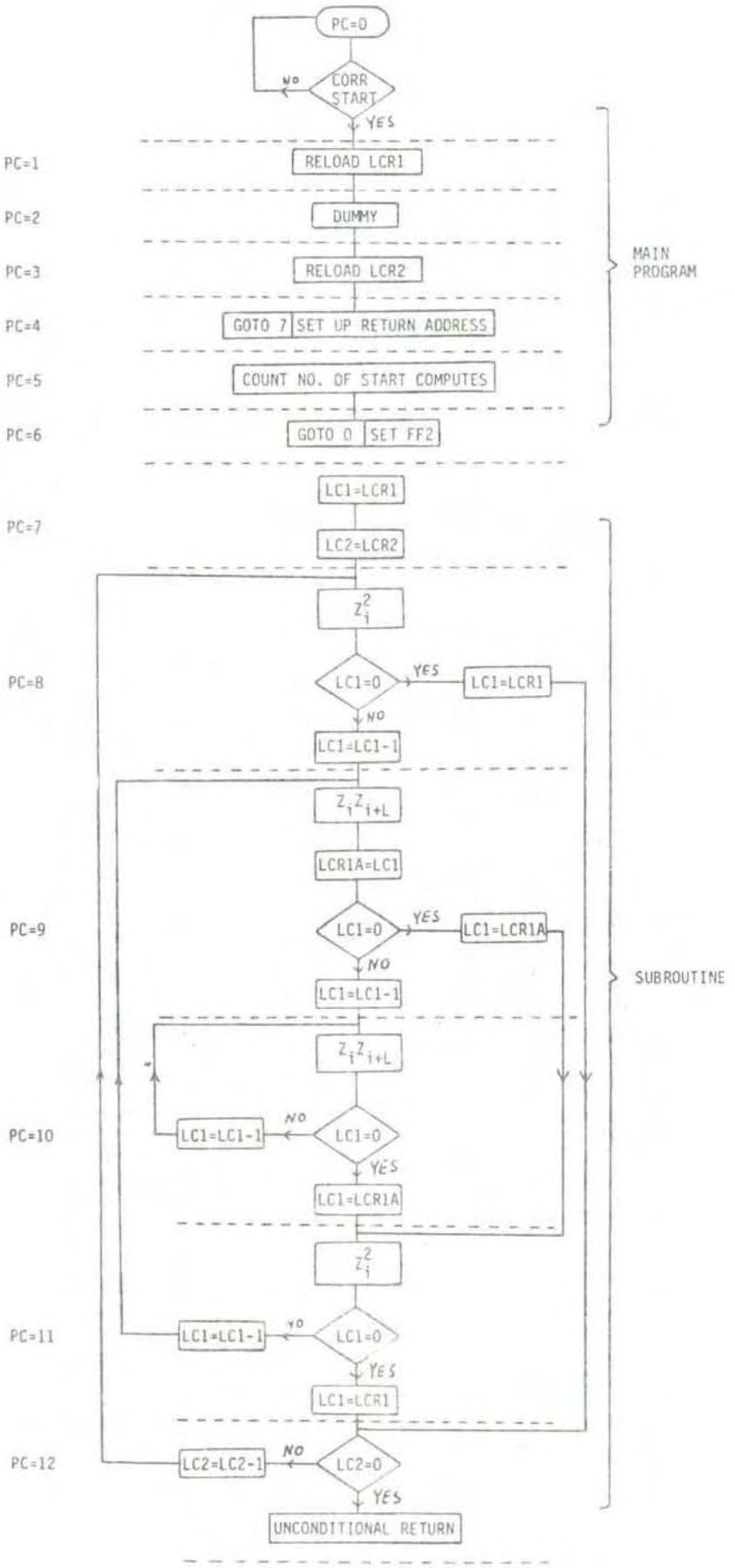


FIGURE 5. FLOW CHART FOR SINGLE PULSE PROGRAM

COMMENTS

PC = 0 Correlator is in idle mode until START COMPUTE signal is received from the radar controller, then jumps to PC value set in START ADDRESS REGISTER. (See Ref. 3 DATA-FIELD In DIGITAL CORRELATOR.)

PC = 1 See Ref. 3. PROGRAM INSTRUCTIONS

PC = 2 Correlator needs two clock cycles for a RELOAD, hence one dummy instruction.

PC = 4 Jump to the subroutine and push the register stack for return from subroutine (i.e. jump to PC = 5 from subroutine)

PC = 5 The START COMPUTE count gives the number of times the correlator has actually received a START COMPUTE Signal before a DMA transfer. (i.e. Actual integration time before DMA transfer). This value is placed after the data in the result memory in both channels as a negative number.

PC = 6 Jump to PC = 0 and wait for next START COMPUTE signal. The FF2 is now set for integration of next time average.

PC = 9 Note that LCR1A receives the value of LC1 before LC1 changes value. The loading of LCR1A occurs in the middle of the clock cycle and LC1 changes value at the end of the clock cycle.

APB PROCESSOR

RS(15) = No. of Samples - 1 in rangecell

RS(14) = No. of Rangecells -1 for time average

RS(13) = Rangecell Incr. (=1 for no overlapping of rangecells and $\leq$ 0 for overlapping of rangecells)

RS(12) = Sample Incr. (Normally = 1)

RS(11) = Temporary Storage.

PC = 0	Do nothing		
PC = 1			F → RS(15)
Pc = 2	Do nothing		
PC = 3			F → RS(14)
PC = 4		Q = F	F → $\overline{\text{DATAI}}$.AND.0
PC = 5		Q = F	F → RS(12)+Q
PC = 6	Do nothing		
PC = 7		Q = F	F → Q-RS(13)
PC = 8		Q = F	F → RS(13)+Q
PC = 9	RS(11)=F		F → RS(12)+Q
PC = 10	RS(11)=F		F → RS(12)+RS(11)
PC = 11		Q = F	F → RS(12)+Q
PC = 12	Do nothing		

APM PROCESSOR

RS(15) = Rangecell Incr. (= no. of Lags computed)

RS(14) = Incr. (Normally = 1)

RS(13) = Temp. storage

PC = 0	Do nothing		
PC = 1	Do nothing		
PC = 2	Do nothing		
PC = 3	Do nothing		
PC = 4		Q = F	F → $\overline{\text{DATAI}}$.AND.0
PC = 5		Q = F	F → RS(15)+Q
PC = 6	Do nothing		
PC = 7		Q = F	F → Q-RS(15)
PC = 8		Q = F	F → RS(15)+Q
PC = 9	RS(13)=F		F → RS(14)+Q
PC = 10	RS(13)=F		F → RS(14)+RS(13)
PC = 11			F → Q
PC = 12	Do nothing		

FLIP FLOPS (FF1, FF2) ON ACCUMULATORS

PC = 0	set strobe = 0	WRITE = 0	READ = 0
PC = 1	set strobe = 1	WRITE = 0	READ = 0
PC = 2	set strobe = 1	WRITE = 0	READ = 0
PC = 3	set strobe = 1	WRITE = 0	READ = 0
PC = 4	set strobe = 1	WRITE = 0	READ = 0
PC = 5	set strobe = 1	WRITE = 1	READ = 1
PC = 6	set strobe = 1	WRITE = 0	READ = 0
PC = 7	set strobe = 1	WRITE = 0	READ = 0
PC = 8	set strobe = 1	WRITE = 1	READ = 1
PC = 9	set strobe = 1	WRITE = 1	READ = 1
PC = 10	set strobe = 1	WRITE = 1	READ = 1
PC = 11	set strobe = 1	WRITE = 1	READ = 1
PC = 12	set strobe = 1	WRITE = 0	READ = 0

	FF1		FF2	
PC = 0	CLEAR1 = 0	SET1 = 0	CLEAR2 = 0	SET2 = 0
PC = 1	CLEAR1 = 0	SET1 = 0	CLEAR2 = 0	SET2 = 0
PC = 2	CLEAR1 = 0	SET1 = 0	CLEAR2 = 0	SET2 = 0
PC = 3	CLEAR1 = 0	SET1 = 0	CLEAR2 = 0	SET2 = 0
PC = 4	CLEAR1 = 0	SET1 = 0	CLEAR2 = 0	SET2 = 0
PC = 5	CLEAR1 = 1	SET1 = 0	CLEAR2 = 0	SET2 = 0
PC = 6	CLEAR1 = 0	SET1 = 0	CLEAR2 = 0	SET2 = 1
PC = 7	CLEAR1 = 0	SET1 = 0	CLEAR2 = 0	SET2 = 0
PC = 8	CLEAR1 = 1	SET1 = 0	CLEAR2 = 0	SET2 = 0
PC = 9	CLEAR1 = 0	SET1 = 0	CLEAR2 = 0	SET2 = 0
PC = 10	CLEAR1 = 0	SET1 = 0	CLEAR2 = 0	SET2 = 0
PC = 11	CLEAR1 = 0	SET1 = 1	CLEAR2 = 0	SET2 = 0
PC = 12	CLEAR1 = 0	SET1 = 0	CLEAR2 = 0	SET2 = 0

COMMENTS

PC = 5 — If FF2 has been reset write -1 into result memory. If FF2 has been set read from result memory, add -1, then write to result memory.

PC = 8 — Reset FF1 to read 0 into accumulators, then write to result memory.

PC = 9 — If FF1 has been previously reset read 0 into accumulators, if FF1 has been previously set read from result memory into accumulators, then write to result memory.

PC = 10 — If FF1 has been previously reset read 0 into accumulators, if FF1 has been previously set read from result memory into accumulators, then write to result memory.

PC = 11 — Set FF1 to read from result memory into accumulators, then write to result memory.

see Ref. 3 ARI/ACC INSTRUCTIONS. Refer to ACC Section.

ARITHMETICAL PART OF DATA PROCESSING

DATA SAMPLES TO REGISTERS

	MULTIPLIER4		MULTIPLIER3		MULTIPLIER2		MULTIPLIER1	
	B	A	B	A	B	A	B	A
PC = 0	X	X	X	X	X	X	X	X
PC = 1	X	X	X	X	X	X	X	X
PC = 2	X	X	X	X	X	X	X	X
PC = 3	X	X	X	X	X	X	X	X
PC = 4	X	X	X	X	X	X	X	X
PC = 5	X	X	X	X	X	X	X	X
PC = 6	X	X	X	X	X	X	X	X
PC = 7	X	X	X	X	X	X	X	X
PC = 8	Y	X	X	Y	Y	Y	X	X
PC = 9	Y	X	X	Y	Y	Y	X	X
PC = 10	Y	X	X	Y	Y	Y	X	X
PC = 11	Y	X	X	Y	Y	Y	X	X
PC = 12	X	X	X	X	X	X	X	X

STROBING INTO REGISTERS

	MULTIPLIER4		MULTIPLIER3		MULTIPLIER2		MULTIPLIER1	
	B	A	B	A	B	A	B	A
PC = 0	NS	NS	NS	NS	NS	NS	NS	NS
PC = 1	NS	NS	NS	NS	NS	NS	NS	NS
PC = 2	NS	NS	NS	NS	NS	NS	NS	NS
PC = 3	NS	NS	NS	NS	NS	NS	NS	NS
PC = 4	NS	NS	NS	NS	NS	NS	NS	NS
PC = 5	NS	NS	NS	NS	NS	NS	NS	NS
PC = 6	NS	NS	NS	NS	NS	NS	NS	NS
PC = 7	NS	NS	NS	NS	NS	NS	NS	NS
PC = 8	S	S	S	S	S	S	S	S
PC = 9	S	NS	S	NS	S	NS	S	NS
PC = 10	S	NS	S	NS	S	NS	S	NS
PC = 11	S	S	S	S	S	S	S	S
PC = 12	NS	NS	NS	NS	NS	NS	NS	NS

S = strobe value into register. NS = No strobe.
PC = 0 to PC = 4 and PC = 6, PC = 7, PC = 12 are dummy instructions. In PC = 5 - 1 is strobed into the ALU's and in PC = 8 to PC = 11 new samples are selected from the buffer memory and ALU12 computes MULTIPLIER1 + MULTIPLIER2 and ALU34 computes MULTIPLIER3 - MULTIPLIER4. See Ref. 3 ARI/ACC instructions. Refer to ARI instructions.

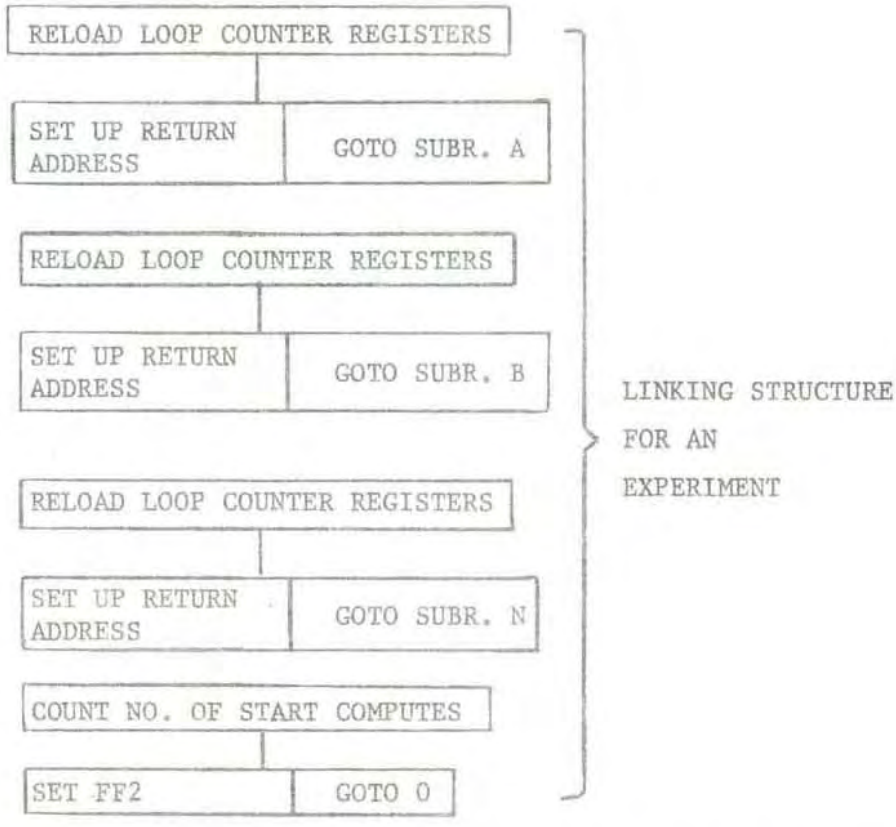
OUT AND I/O INSTRUCTIONS

These values are set to zero. The OUT instructions are used for a DMA transfer and the I/O instructions are only used in multicorrelator systems. See REF.3 for further details.

FURTHER COMMENTS

As can be seen from the flow chart the loop counters are decremented after sample processing so it is thus necessary to define LCR1 = No of samples in rangecell - 1 and LCR2 = No of rangecells - 1 for time average.
The only restriction in this subroutine is that No. of samples ≥ 1. The subroutine assames that the start addresses of both the buffer and result memories are already in the Q registers of the APB, APM processors which in this case are set to zero in the main program.

VI. MAIN PROGRAM AND LINKING STRUCTURE



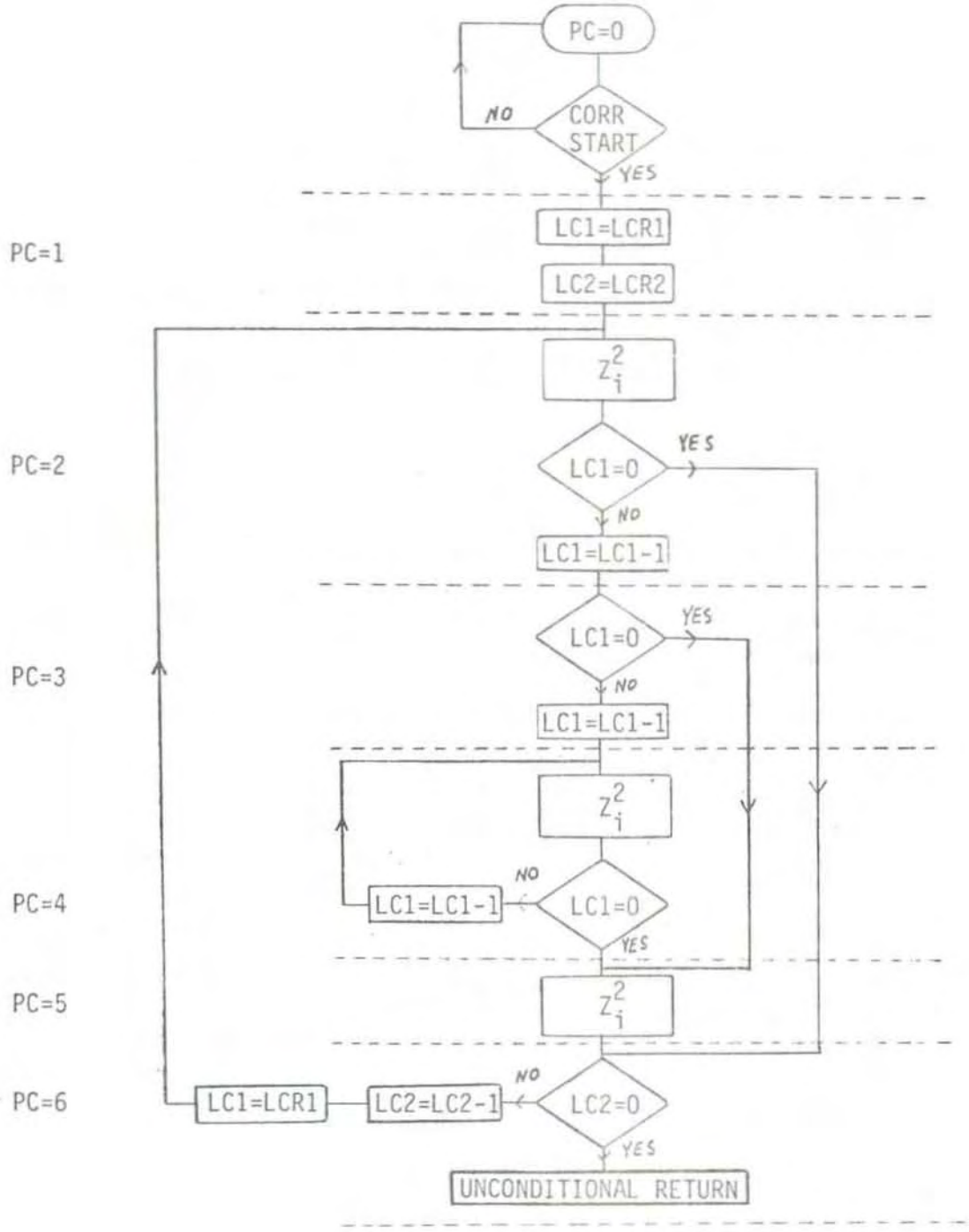
With this structure it is fairly easy to construct a complete program for a particular experiment. It is only necessary to create the main program, load the necessary subroutines from file changing only the registers in the APB and APM stacks if they are differently defined for the experiment and then saving it on a file for later use. It should be remembered that the transfer program must start at location 32 in the RAM of the correlator. An example of how to combine several subroutines with the corresponding main program will be given in chapter VIII.2.

VII. DESCRIPTIONS OF SUBROUTINES

Descriptions of all subroutines designed to date will be given in this section. Also a description of the transfer program will be included.

VII.1.

POWER PROFILE SUBROUTINE (VERSION 1)



APB

RS(15) = No. of Samples -1 in rangecell

RS(14) = No. of Rangecells -1 for time average

RS(13) = Rangecell Incr. (=1 for no overlapping of rangecells)

RS(12) = Sample Incr. (Normally = 1)

PC = 0	Do nothing		
PC = 1		Q = F	F → Q-RS(13)
PC = 2		Q = F	F → RS(13)+Q
PC = 3			F → Q
PC = 4		Q = F	F → RS(12)+Q
PC = 5		Q = F	F → RS(12)+Q
PC = 6	Do nothing		

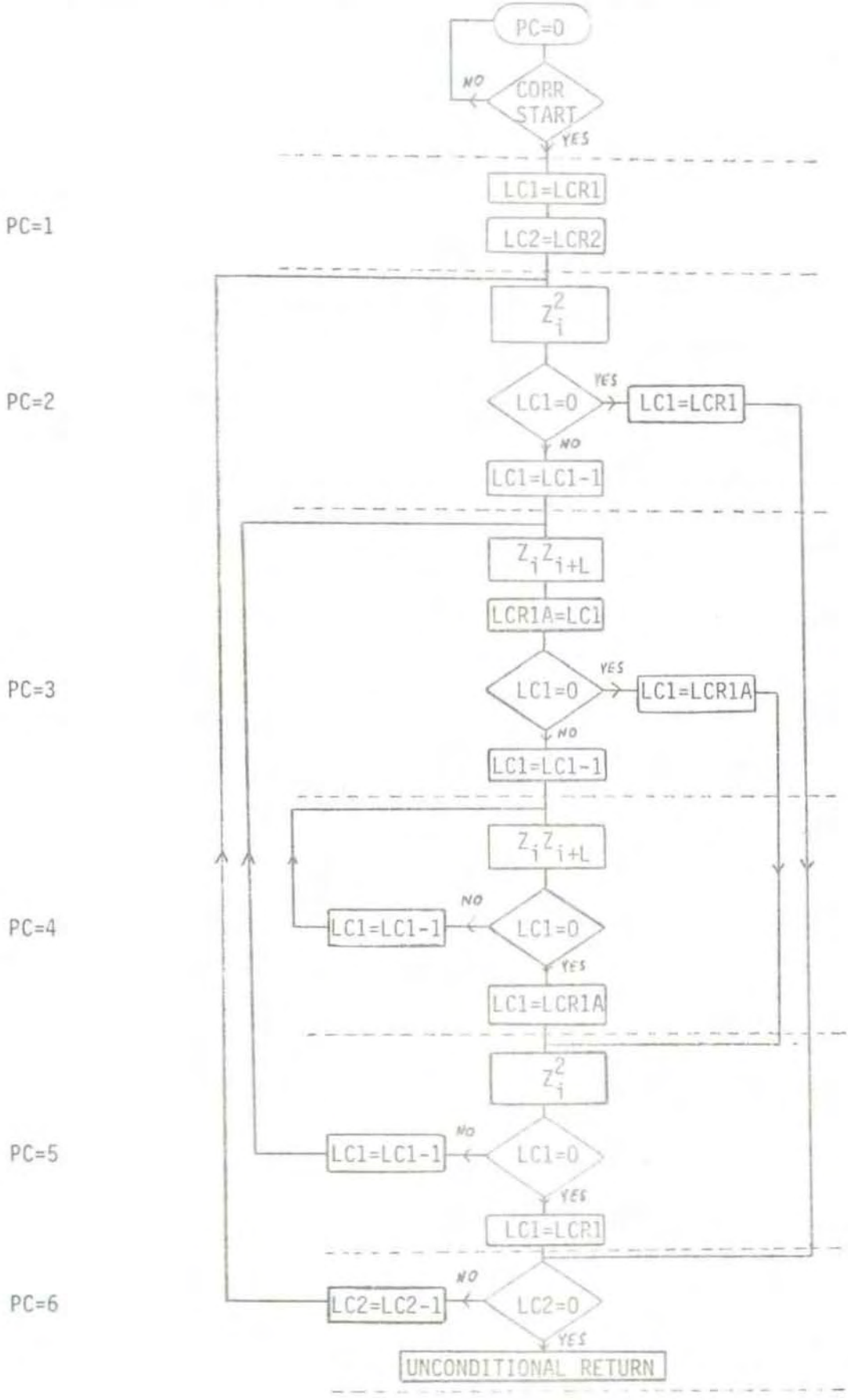
APM

RS(15) = Increment (= 1)

PC = 0	Do nothing		
PC = 1		Q = F	F → Q-RS(15)
PC = 2		Q = F	F → RS(15)+Q
PC = 3	Do nothing		
PC = 4	Do nothing		
PC = 5			F → Q
PC = 6	Do nothing		

VII.2.

SINGLE PULSE SUBROUTINE (NO. OF LAGS .EQ. NO. OF SAMPLES)



APB

RS(15) = No. of Samples -1 in rangecell

RS(14) = No. of Rangecells -1 for time average

RS(13) = Rangecell Incr. (= 1 for no overlapping of rangecells)

RS(12) = Sample Incr. (Normally = 1)

RS(11) = Temporary Storage

PC = 0	Do nothing		
PC = 1		Q = F	F → Q-RS(13)
PC = 2		Q = F	F → RS(13)+Q
PC = 3	RS(11) = F		F → RS(12)+Q
PC = 4	RS(11) = F		F → RS(12)+RS(11)
PC = 5		Q = F	F → RS(12)+Q
PC = 6	Do nothing		

APM

RS(15) = Rangecell Incr. (= No. of Lags computed)

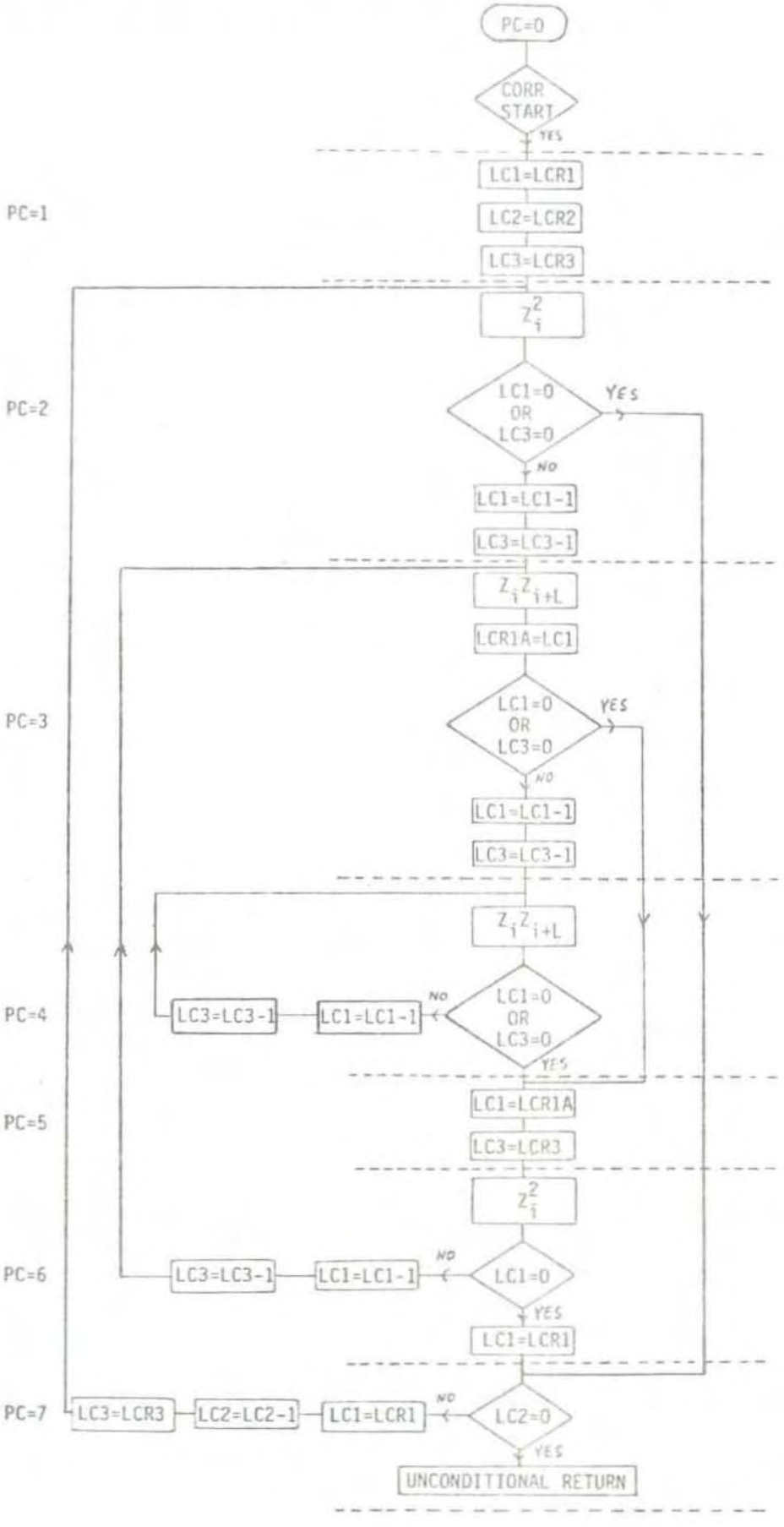
RS(14) = Increment (= 1)

RS(13) = Temporary storage

PC = 0	Do nothing		
PC = 1		Q = F	F → Q-RS(15)
PC = 2		Q = F	F → RS(15)+Q
PC = 3	RS(13) = F		F → RS(14)+Q
PC = 4	RS(13) = F		F → RS(14)+RS(13)
PC = 5			F → Q
PC = 6	Do nothing		

VII.3.

SINGLE PULSE SUBROUTINE (NO. OF LAGS .LE. NO. OF SAMPLES)



APB

RS(15) = No. of Samples -1 in rangecell

RS(14) = No. of Rangecells -1 for time average

RS(13) = No. of Lags -1 in rangecell

RS(12) = Rangecell Incr. (= 1 for no overlapping of Rangecells)

RS(11) = Sample Incr. (Normally = 1)

RS(10) = Temporary Storage

PC = 0	Do nothing		
PC = 1		Q = F	F → Q-RS(12)
PC = 2		Q = F	F → RS(12)+Q
PC = 3	RS(10) = F		F → RS(11)+Q
PC = 4	RS(10) = F		F → RS(11)+RS(10)
PC = 5	Do nothing		
PC = 6		Q = F	F → RS(11)+Q
PC = 7	Do nothing		

APM

RS(15) = Rangecell Incr. (= No. of Lags computed)

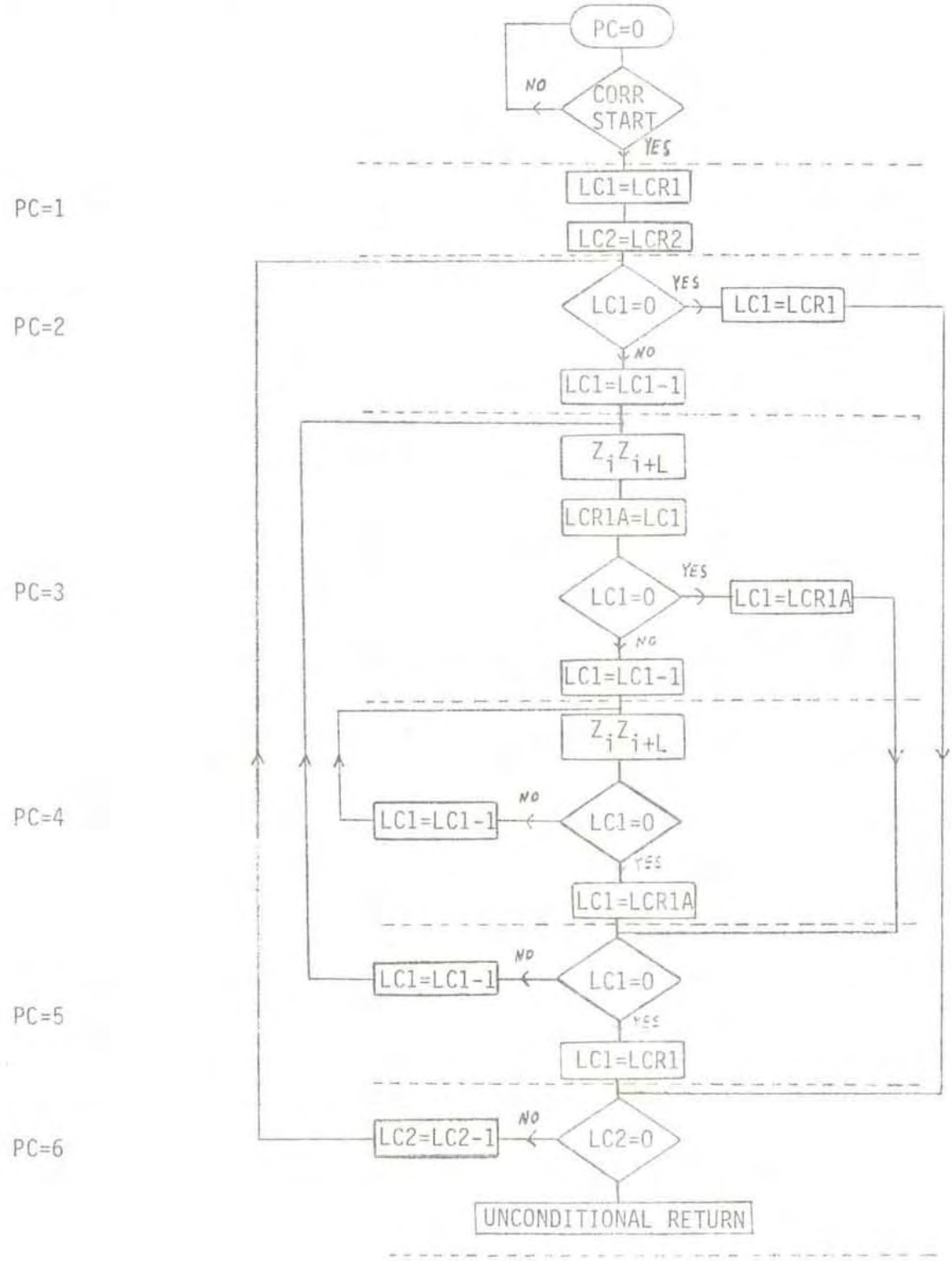
RS(14) = Increment (= 1)

RS(13) = Temporary Storage

PC = 0	Do nothing		
PC = 1		Q = F	F → Q-RS(15)
PC = 2		Q = F	F → RS(15)+Q
PC = 3	RS(13) = F		F → RS(14)+Q
PC = 4	RS(13) = F		F → RS(14)+RS(13)
PC = 5	Do nothing		
PC = 6			F → Q
PC = 7	Do nothing		

VII.4.

MULTI PULSE SUBROUTINE



APB

RS(15) = No. of Pulses -1 in Rangecell

RS(14) = No. of Rangecells -1 for time average

RS(13) = Rangecell Increment (= 1 for no overlapping of rangecells)

RS(12) = Temporary Storage

RS(1) = Sample Distance between 2nd last and 1st Pulse

RS(0) = Sample Distance between last and 1st Pulse

PC = 0	Do nothing		
PC = 1	RS(12) = F		F → Q-RS(13)
PC = 2	RS(12) = F		F → RS(13)+RS(12)
PC = 3			F → RS(12)+RS(LC1)
PC = 4			F → RS(12)+RS(LC1)
PC = 5		Q = F	F → RS(12)+RS(LC1)
PC = 6	Do nothing		

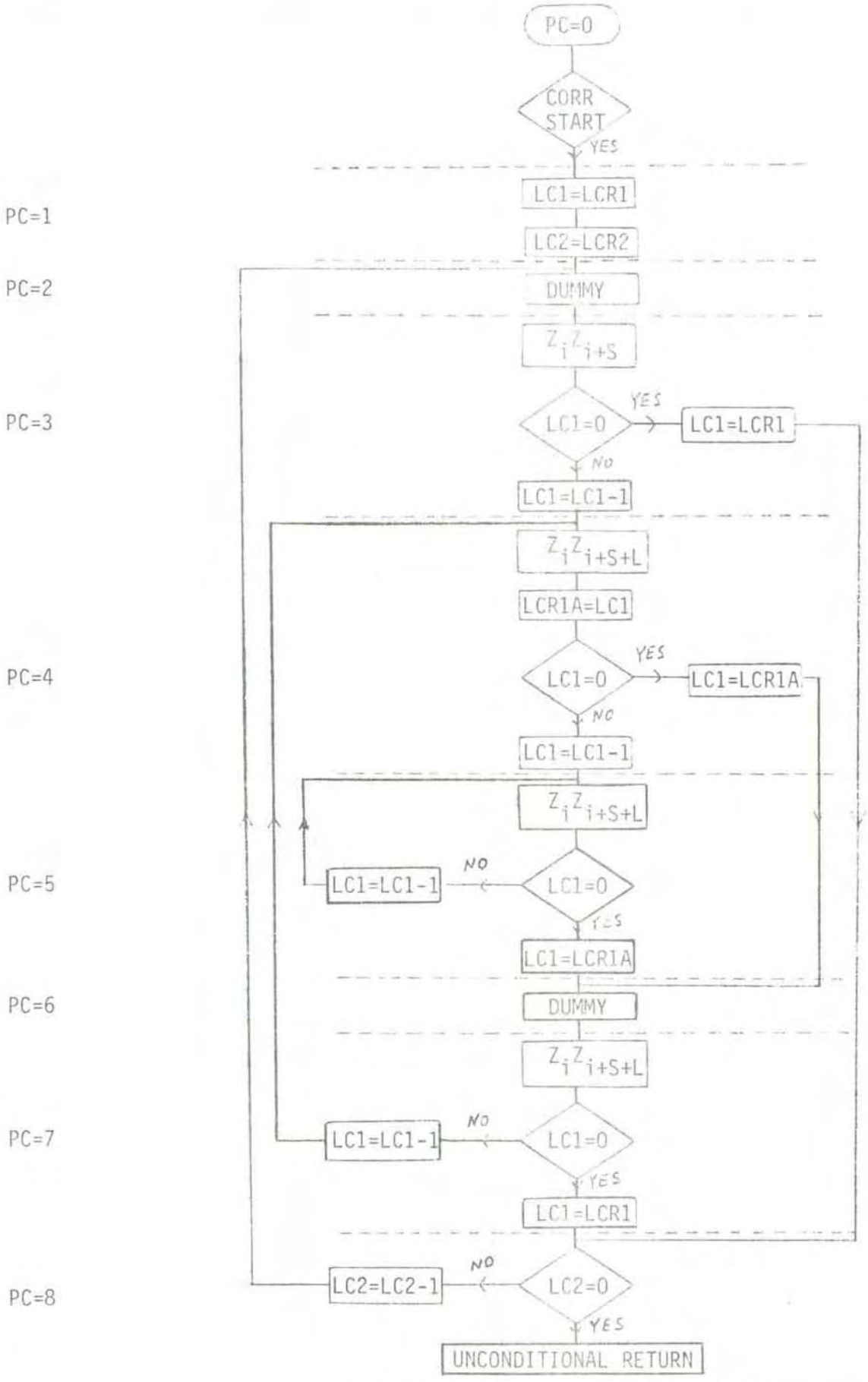
APM

RS(15) = Increment (= 1)

PC = 0	Do nothing		
PC = 1		Q = F	F → Q-RS(15)
PC = 2	Do nothing		
PC = 3		Q = F	F → RS(15)+Q
PC = 4		Q = F	F → RS(15)+Q
PC = 5	Do nothing		
PC = 6	Do nothing		

VII.5.

CROSS CORRELATION SUBROUTINE (NO. OF LAGS .EQ. NO. OF SAMPLES)



APB

RS(15) = No. of Samples -1 in Rangecell
RS(14) = No. of Rangecells -1 for time average
RS(13) = Rangecell Incr. (= 1 for no overlapping of rangecells)
RS(12) = Start Address of 2nd Field - Start Address of 1st Field
RS(11) = Sample Incr. (= 1)
RS(10) = Temporary Storage

PC = 0	Do nothing		
PC = 1		Q = F	F → Q-RS(13)
PC = 2		Q = F	F → RS(13)+Q
PC = 3	RS(10) = F		F → RS(12)+Q
PC = 4	RS(10) = F		F → RS(11)+RS(10)
PC = 5	RS(10) = F		F → RS(11)+RS(10)
PC = 6		Q = F	F → RS(11)+Q
PC = 7	RS(10) = F		F → RS(12)+Q
PC = 8	Do nothing		

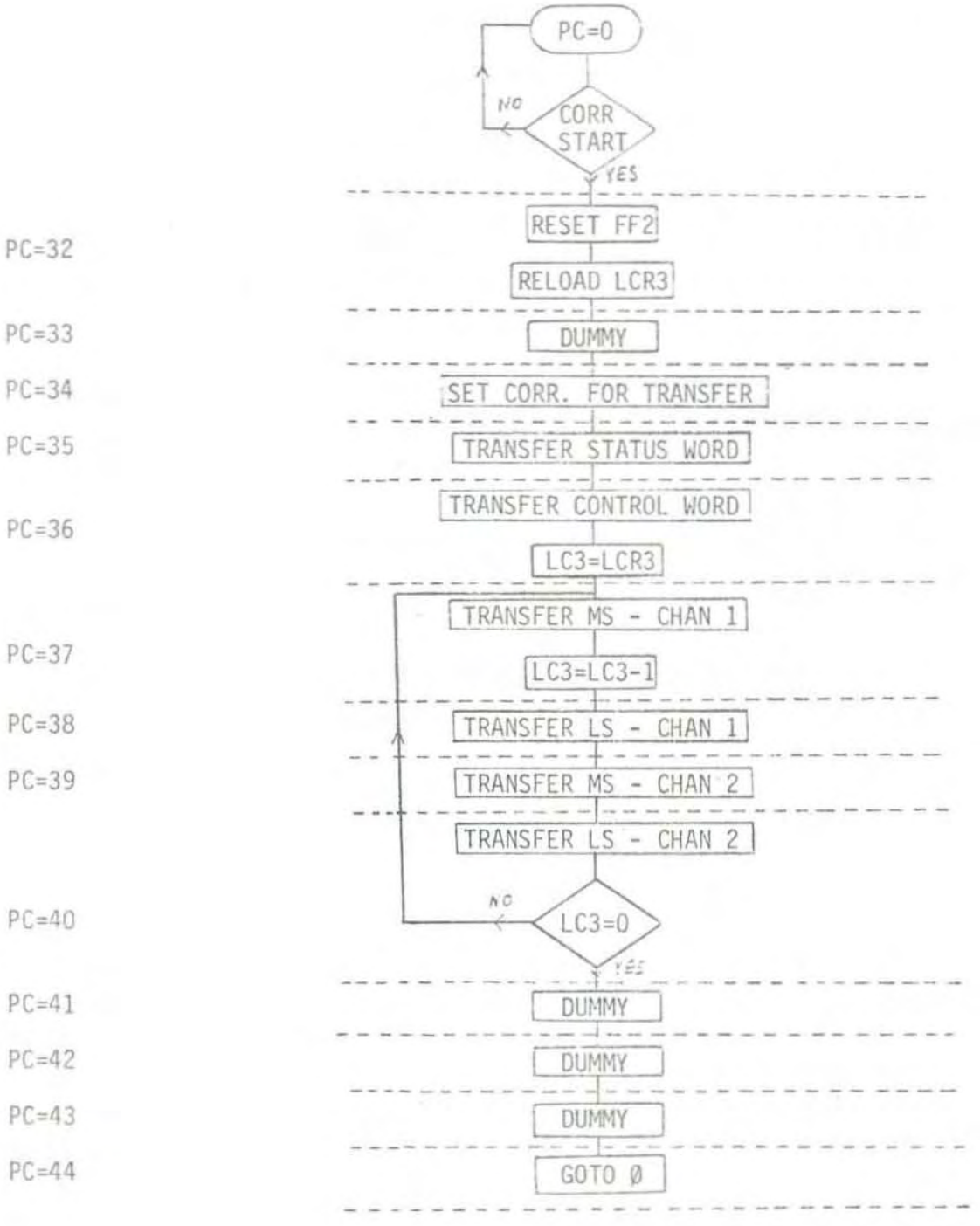
APM

RS(15) = Rangecell Incr. (= No. of Lags computed)
RS(14) = Incr. (= 1)
RS(13) = Temporary Storage

PC = 0	Do nothing		
PC = 1		Q = F	F → Q-RS(15)
PC = 2	Do nothing		
PC = 3	RS(13) = F		F → RS(15)+Q
PC = 4	RS(13) = F		F → RS(14)+Q
PC = 5	RS(13) = F		F → RS(14)+RS(13)
PC = 6	Do nothing		
PC = 7			F → Q
PC = 8	Do nothing		

VII.6.

TRANSFER PROGRAM



APB

DATAI REG = No of 64 BIT Words for DMA Transfer.

PC = 0	Do nothing		
PC = 32			F → DATAI
PC = 33	Do nothing		
PC = 34	Do nothing		
PC = 35	Do nothing		
PC = 36	Do nothing		
PC = 37	Do nothing		
PC = 38	Do nothing		
PC = 39	Do nothing		
PC = 40	Do nothing		
PC = 41	Do nothing		
PC = 42	Do nothing		
PC = 43	Do nothing		
PC = 44	Do nothing		

APM

RS(0) = Increment (= 1)

PC = 0	Do nothing		
PC = 32	Do nothing		
PC = 33	Do nothing		
PC = 34	Do nothing		
PC = 35		Q = F	F → $\overline{\text{DATAI}}$.AND.0
PC = 36		Q = F	F → Q-RS(0)
PC = 37		Q = F	F → RS(0)+Q
PC = 38			F → Q
PC = 39			F → Q
PC = 40			F → Q
PC = 41	Do nothing		
PC = 42	Do nothing		
PC = 43	Do nothing		
PC = 44	Do nothing		

VIII USE OF CORRSIM, CORRTEST. PROGRAM DEVELOPMENT AND CORRELATOR CONTROL

The program CORRSIM is used for designing new programs and interactive communication with the correlator. The user file can be of any name on which the correlator program can be READ/WRITTEN from/to. The CORRSIM files PROG0... PROGN (read access only) are fixed files of the various subroutines. There will also be a library of fixed programs from which the experimenter may use, if suitable, for a particular experiment. It will only be necessary to define the parameters for that experiment. These files can be found under the names GP1, GP2 ... GPN and descriptions can be found in the manual "STANDARD SUBROUTINES AND PROGRAMS FOR EISCAT DIGITAL CORRELATOR" by T. Ho. EISCAT Technical Notes. No. 81/27.

The program CORRTEST is used for testing of programs off-line by simulating the correlator hardware.

For furthers details about these programs see Ref. 2.

VIII.1 STRUCTURE OF THE PROM IN THE CORRELATOR.

There will be several PROMs which will contain different fixed programs. It is only necessary to insert the necessary PROMs into the correlator in order to use them. These programs will be given in a later report. Note that the transfer program for transferring data from the result memory through the DMA channel will start at PC = 32 in all PROMs. When the correlator receives the START TRANSFER signal from the radar controller it automatically executes a jump to this value.

Although Fig. 11 only shows one PROM there are in fact eight PROMs which have to be inserted into the correlator.

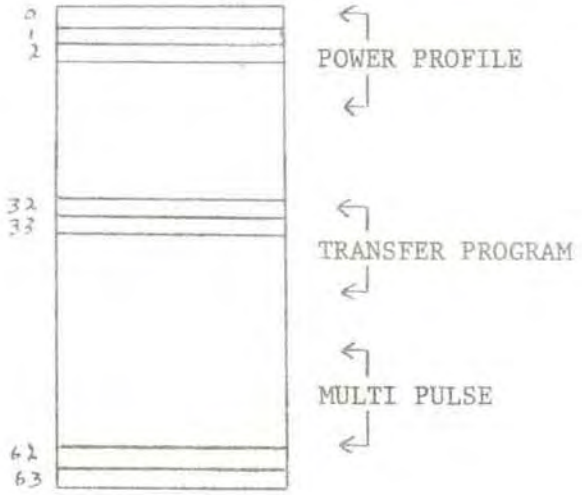


FIGURE 11 STRUCTURE OF THE PROM

These eight PROMs correspond to the eight RAM modules in the correlator. Figure 11 is only intended as an illustration and the order of the programs should be ignored except for the position of the transfer program.

In order to use the PROM BIT 3 in the STATUS word is set to 1, the SAR is set to the location at which the particular program starts and all the other necessary parameters are set through defining the DATA field in the CORRSIM program. The data field is then loaded into the correlator and the correlator is then ready for START COMPUTE from the radar controller.

VIII. 2 CREATION OF A NEW PROGRAM

If a new program is to be created containing a combination of the sub-routines stored in the files PROG 0 - PROG 9, then it is only necessary to create a main program and link it to whichever subroutines are neccessary from PROG0 - PROGN (see Fig. 1).

As an example consider the following scheme. We would like to have a program whereby we could select either the Power Profile, single Pulse or Cross Correlation scheme such that the data, noise and calibration samples may be computed in any order. We could have the following flow chart.

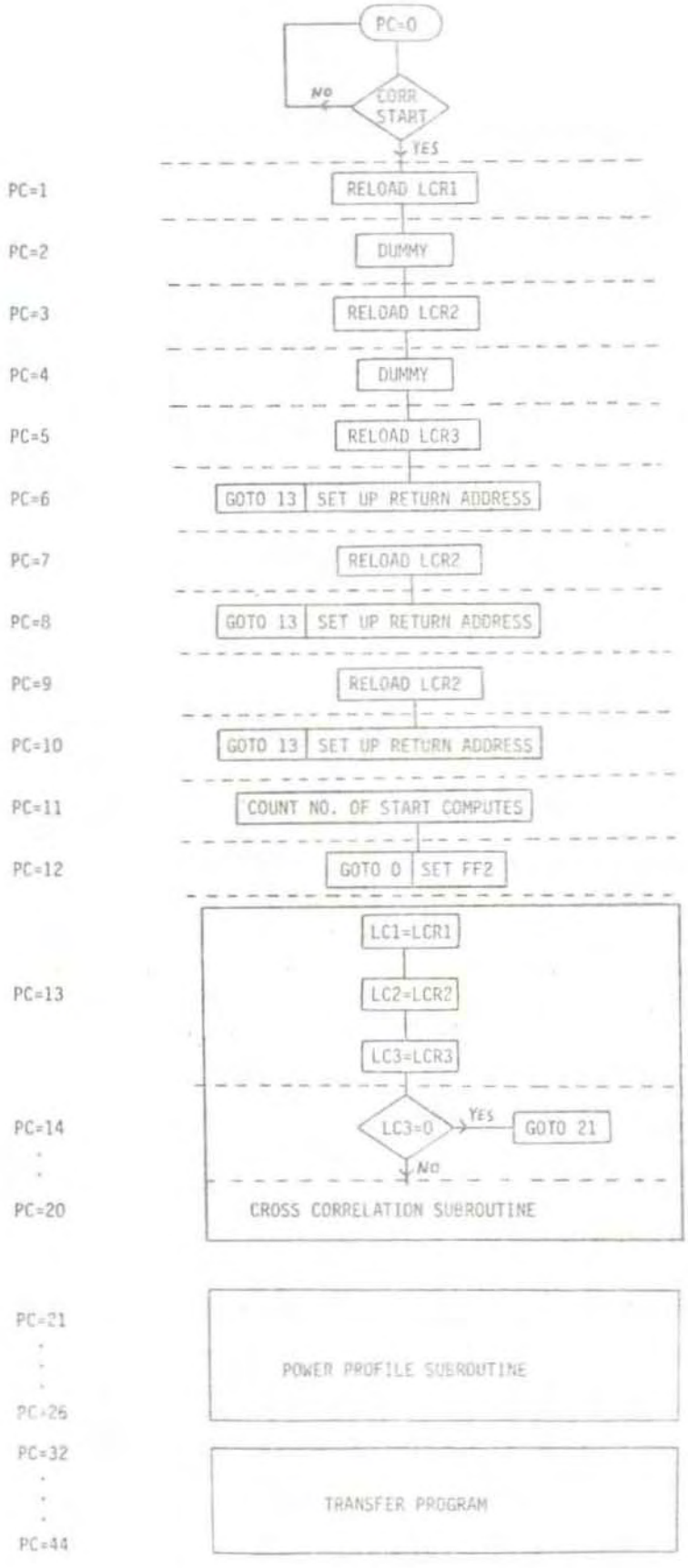


FIGURE 12. LINKING STRUCTURE WITH SUBROUTINES

APB PROCESSOR

DATAI REG = No. of words for DMA TRANSFER

RS(15) = No. of Samples -1 in Rangecell (Data and Noise and Calibration)

RS(14) = No. of Rangecells -1 for time average (Data or Noise or Calibration) Computed first.

RS(13) = No of Lags -1 in Rangecell (Data and Noise and Calibration)

RS(12) = Rangecell Increment (=1 for no overlapping of rangecells)

RS(11) = Start Address 2nd Field-Start Address 1st Field

RS(10) = Sample Increment (NORMALLY = 1)

RS(9) = Temporary Storage.

RS(8) = No. of Rangecells -1 for time average (Data or Noise or Calibration) computed 2nd

RS(7) = No. of Rangecells -1 for Time average (Data or Noise or Calibration) computed 3rd.

FOR MAIN PROGRAM

PC = 0	Do nothing	
PC = 1		F → RS(15)
PC = 2	Do nothing	
PC = 3		F → RS(14)
PC = 4	Do nothing	
PC = 5		F → RS(13)
PC = 6	Q = F	F → $\overline{\text{DATAI}}$.AND.0
PC = 7		F → RS(8)
PC = 8	Q = F	F → RS(10)+Q
PC = 9		F → RS(7)
PC = 10	Q = F	F → RS(10)+Q
PC = 11	Q = F	F → RS(10)+Q
PC = 12	Do nothing	

APM PROCESSOR

RS(15) = Rangecell Increment (= No. of Lags computed)

RS(14) = Increment (= 1)

RS(13) = Temporary storage

RS(0) = Increment for Transfer Program (= 1)

FOR MAIN PROGRAM

PC = 0	Do nothing		
PC = 1	Do nothing		
PC = 2	Do nothing		
PC = 3	Do nothing		
PC = 4	Do nothing		
PC = 5	Do nothing		
PC = 6		Q = F	F → $\overline{\text{DATAI}}$.AND.0
PC = 7	Do nothing		
PC = 8		Q = F	F → RS(15)+Q
PC = 9	Do nothing		
PC = 10		Q = F	F → RS(15)+Q
PC = 11		Q = F	F → RS(15)+Q
PC = 12	Do nothing		

In this program the LCR1 register is loaded with the No. of samples -1 in Rangecell (Data and Noise and Calibration) parameter, the LCR2 register is loaded with the No. of Rangecells -1 for time average (Data or Noise or Calibration), the LCR3 register is loaded with No. of Lags -1 in Rangecell (Data and Noise and Calibration). This will be so for all subroutines except for the Multi Pulse where LCR1 register is loaded with No. of Pulses -1 in Rangecell. The Data and Noise and Calibration in parenthesis means that the same value is used when computing the three different sets of data. The Data or Noise or Calibration in parenthesis means that three values have to be defined for the different data sets and the computed 1st, 2nd, 3rd says in which order they are used in the computation.

In the program it can be seen that the APB, APM register stacks are differently defined from those of Power Profile subroutine and are consequently changed in the program. A modification has also been made in the Cross correlation subroutine at PC = 14 (PRO instruction) and in the Power Profile subroutine at PC = 21 (APM instruction) of the program.

To get the Power Profile routine define APBRS(13) = 0
To get the Single Pulse routine define APBRS(11) = 0
To get the Cross Correlation routine define APBRS(13) $\neq$ 0 and APBRS(11) $\neq$ 0

In order to place a subroutine in a different location than that defined in the subroutine into the correlator it is necessary to give the difference between the wanted location and that of the defined location when the display shows "GIVE ENTRY-POINT IN CORRELATOR MEMORY:" For example, for this program the cross correlation subroutine starts at location (PC =) 1 on the file and it is necessary to place it in the correlator memory starting at location 13 then a value of 12 has to be given when "GIVE ENTRY-POINT IN CORRELATOR MEMORY:" is displayed. If there is a direct jump address in the subroutine then it will be automatically adjusted. Also location 0 will remain location 0 when a value is given (see Ref 2 for further details).

IX CONCLUSIONS

There are several restrictions which have to be taken into account when designing a program. First, the size of RAM of the Correlator. This memory has 64 locations and as location 0 is used for idle running of the correlator, location 63 for the storage of a service routine and locations 32-44 for the transfer program there are in effect only 41 locations free for programming (1 - 31 and 45 - 62). Hence the number of subroutines that can be loaded into the correlator at any one time is restricted by the lengths of the main program and subroutines. Second, the size of the register stacks for the APB and APM processors. These stacks have 16 available locations for storage of the parameters. When using the Multi Pulse subroutine in a particular combination it can be seen that there will be necessarily a restriction on the number of pulses that can be used for that scheme in that particular experiment depending on how the register stack for the APB processor is defined. Third, there are only 4 return addresses that can be used at any one time in a program. Fourth, there are only 3 loop counters that can be used at any one time.

In spite of these programming restrictions the correlator has enough programming flexibility to cater for the needs of an experimenter.

ACKNOWLEDGEMENT

We would like to thank the EISCAT Scientific Association for sponsoring this work and to all of the staff in Tromsö for their help.

X. REFERENCES

1: Hans-Jørgen Alker: "A PROGRAMMABLE CORRELATOR MODULE FOR THE EISCAT RADAR SYSTEM". Reprint no. 51-78 from the Auroral Observatory, Tromsø, Norway (Febr. 1978)

2: Terrance Ho: "PROGRAMS CORRSIM, CORRTEST: SYSTEM FOR PROGRAM DEVELOPMENT AND SOFTWARE SIMULATION OF EISCAT DIGITAL CORRELATOR. USER'S MANUAL". EISCAT TECHNICAL NOTE 81/25, 1981

3: Terrance Ho: "INSTRUCTION MANUAL FOR EISCAT DIGITAL CORRELATOR". (Revised). EISCAT TECHNICAL NOTE 81/26, 1981

4: SVEIN A. KVALVIK "CORRELATOR BUFFER-MEMORY FOR THE EISCAT RADAR SYSTEM." EISCAT TECHNICAL NOTE 80/20, 1980

APPENDIX A

Here listings of the basic subroutines, Transfer program and a combination program are given.

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: POWER PROFILE SUBROUTINE (VERSION 1)
FILE-NAME (NORD 10): PROGØ:DATA
PROGRAM DESCRIPTION:

DATA CHANNEL 1 ZERO LAG ESTIMATION $K_r = \sum_{i=0}^{N-1} (X^2_{i+(N+D-1)(r-1)} + Y^2_{i+(N+D-1)(r-1)})$

DATA CHANNEL 2 MEAN VALUE ESTIMATION $M_r = \sum_{i=0}^{N-1} (X_{i+(N+D-1)(r-1)} + Y_{i+(N+D-1)(r-1)})$

WHERE N=NO. OF SAMPLES IN RANGECELL
D=OVERLAP FACTOR (=1 FOR NO OVERLAPPING AND $\leqslant$0 FOR OVERLAPPING)
r=1,2,...,M RANGECELLS FOR TIME AVERAGE

RESTRICTIONS

MINIMUM NO. OF SAMPLES IN RANGEDATA: 1

NOTES

1. The formulae above are given with respect to the way in which the X, Y samples are read from the buffer memory.
2. The Q registers of the APB, APM processors must be defined with the start addresses of the buffer, result memories in the main program.
3. The LCR1 register must be reloaded with the APBRS(15) register and the LCR2 register must be reloaded with the APBRS(14) register in the main program.

START ADDRESS FOR PROGRAM: 1
PROGRAM-MEMORY LOCATIONS USED: 1 - 6

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO

DATE: 6/8/80

PROGRAM NAME: POWER PROFILE SUBROUTINE (VERSION 1)
FILE-NAME (NORD 10): PROGØ:DATA

REGISTER NAME	REGISTER ADDRESS	PARAMETER
SAR	4	START ADDRESS OF SUBROUTINE
APB RS(15)	16,15	NO. OF SAMPLES-1 IN RANGECELL
APB RS(14)	16,14	NO. OF RANGECELLS-1 FOR TIME AVERAGE
APB RS(13)	16,13	RANGECELL INCREMENT (=1 FOR NO OVER-LAPPING OF RANGECELLS)
APB RS(12)	16,12	SAMPLE INCREMENT (NORMALLY=1)
APM RS(15)	17,15	INCREMENT (=1)

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: POWER PROFILE SUBROUTINE (VERSION 1)
FILE-NAME (NORD 10): PROGØ:DATA
PROGRAM DESCRIPTION:

	APB PROCESSOR		APM PROCESSOR	
PC=1	Q=F	F→Q-RS(13)	Q=F	F→Q-RS(15)
PC=2	Q=F	F→RS(13)+Q	Q=F	F→RS(15)+Q
PC=3		F→Q	DO NOTHING	
PC=4	Q=F	F→RS(12)+Q	DO NOTHING	
PC=5	Q=F	F→RS(12)+Q		F→Q
PC=6	DO NOTHING		DO NOTHING	

START ADDRESS FOR PROGRAM:
PROGRAM-MEMORY LOCATIONS USED:

*** CORRELATOR SIMULATOR ***

PROGRAM NAME : POWER PROFILE SUBROUTINE (VERSION 1)

CODING (INTEGER) OF SEPARATE CORRELATOR FUNCTIONS

PROGRAM-INSTRUCTIONS DEFINED

MEM-LOC.	COND.CODE	CODE-B	CODE-A	ADDR.	LC1	LCR1A	LC2	LC3	RELOAD	R-ADDR.
0	56	0	4	0	0	0	0	0	0	0
1	56	0	4	0	2	0	3	0	0	0
2	57	6	4	6	1	0	0	0	0	0
3	57	6	4	5	1	0	0	0	0	0
4	57	4	6	4	1	0	0	0	0	0
5	56	0	4	0	0	0	0	0	0	0
6	58	1	6	2	2	0	1	0	0	0

APB-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.	SELECT
0	7	5	1	0	0	0
1	0	1	0	13	0	0
2	0	0	0	13	0	0
3	2	0	1	0	0	0
4	0	0	0	12	0	0
5	0	0	0	12	0	0
6	7	5	1	0	0	0

APM-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.
0	7	5	1	0	0
1	0	1	0	15	0
2	0	0	0	15	0
3	7	5	1	0	0
4	7	5	1	0	0
5	2	0	1	0	0
6	7	5	1	0	0

ARITHMETICAL INSTRUCTIONS DEFINED

MEM-LOC.	M1A	M1B	M2A	M2B	M3A	M3B	M4A	M4B	SM1	SM2	SM3	SM4	M12	M34
0	0	0	0	0	0	0	0	0	0	0	0	0	15	15
1	0	0	0	0	0	0	0	0	0	0	0	0	15	15
2	0	0	1	1	4	0	4	1	3	3	3	3	9	9
3	0	0	0	0	0	0	0	0	3	3	3	3	6	6
4	0	0	1	1	4	0	4	1	3	3	3	3	9	9
5	0	0	1	1	4	0	4	1	3	3	3	3	9	9
6	0	0	0	0	0	0	0	0	0	0	0	0	15	15

ACCUMULATOR INSTRUCTIONS DEFINED

MEM-LOC.	STROBE	I/O WRITE	READ	CLEAR1	SET1	CLEAR2	SET2
0	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0
2	1	1	1	0	0	0	0
3	1	0	0	0	0	0	0
4	1	0	0	0	0	0	0
5	1	1	0	0	0	0	0
6	1	0	0	0	0	0	0

I/O INSTRUCTIONS DEFINED

MEM-LOC.	SETF	CLEARF	SELECT	BUF.ADDR.	STROBE	I-REG.	ENABLE EDB	ENABLE EA
0	0	0	0	0	0		0	
1	0	0	0	0	0		0	
2	0	0	0	0	0		0	
3	0	0	0	0	0		0	
4	0	0	0	0	0		0	
5	0	0	0	0	0		0	
6	0	0	0	0	0		0	

OUTPUT-TRANSFER INSTRUCTIONS DEFINED

MEM-LOC.	TRANSFER	INHIBIT	CLOCK	DATA-READY	TRANSFER-CODE	SOURCE
0	0	0	0	0		0
1	0	0	0	0		0
2	0	0	0	0		0
3	0	0	0	0		0
4	0	0	0	0		0
5	0	0	0	0		0
6	0	0	0	0		0

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: SINGLE PULSE SUBROUTINE (NO. OF LAGS .EQ. NO. OF SAMPLES)
FILE-NAME (NORD 10): PROG2:DATA
PROGRAM DESCRIPTION:

DATA CHANNEL 1

$$\mathrm{Re}\left\{K_{L,r}\right\} = \sum_{i=0}^{N-L-1} \left(X_{i+(N+D-1)(r-1)} X_{i+L+(N+D-1)(r-1)} + Y_{i+(N+D-1)(r-1)} Y_{i+L+(N+D-1)(r-1)}\right)$$

DATA CHANNEL 2

$$\mathrm{Im}\left\{K_{L,r}\right\} = \sum_{i=0}^{N-L-1} \left(X_{i+L+(N+D-1)(r-1)} Y_{i+(N+D-1)(r-1)} - X_{i+(N+D-1)(r-1)} Y_{i+L+(N+D-1)(r-1)}\right)$$

WHERE N=NO. OF SAMPLES IN RANGECELL
L=0,1,2,...,N-1
D=OVERLAP FACTOR (=1 FOR NO OVERLAPPING AND $\leq$0 FOR OVERLAPPING)
r=1,2,...,M RANGECELLS FOR TIME AVERAGE

RESTRICTIONS

MINIMUM NO. OF SAMPLES IN RANGEDATA: 1

NOTES

1. The formulae above are given with respect to the way in which the X, Y samples are read from the buffer memory.
2. The Q registers of the APB, APM processors must be defined with the start addresses of the buffer, result memories in the main program.
3. The LCR1 register must be reloaded with the APBRS(15) register and the LCR2 register must be reloaded with the APBRS(14) register in the main program.

START ADDRESS FOR PROGRAM: 1
PROGRAM-MEMORY LOCATIONS USED: 1 - 6

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO

DATE: 6/8/80

PROGRAM NAME: SINGLE PULSE SUBROUTINE (NO. OF LAGS .EQ. NO. OF SAMPLES)
FILE-NAME (NORD 10): PROG2:DATA

REGISTER NAME	REGISTER ADDRESS	PARAMETER
SAR	4	START ADDRESS OF SUBROUTINE
APB RS(15)	16,15	NO. OF SAMPLES-1 IN RANGECELL
APB RS(14)	16,14	NO. OF RANGECELL-1 FOR TIME AVERAGE
APB RS(13)	16,13	RANGECELL INCREMENT (=1 FOR NO OVER-LAPPING OF RANGECELLS)
APB RS(12)	16,12	SAMPLE INCREMENT (NORMALLY=1)
APB RS(11)	16,11	TEMPORARY STORAGE
APM RS(15)	17,15	RANGECELL INCREMENT (=NO. OF LAGS COMPUTED)
APM RS(14)	17,14	INCREMENT (=1)
APM RS(13)	17,13	TEMPORARY STORAGE

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: SINGLE PULSE SUBROUTINE (NO. OF LAGS .EQ. NO. OF SAMPLES)
FILE-NAME (NORD 10): PROG2:DATA
PROGRAM DESCRIPTION:

	APB PROCESSOR			APM PROCESSOR		
PC=1		Q=F	F→Q-RS(13)		Q=F	F→Q-RS(15)
PC=2		Q=F	F→RS(13)+Q		Q=F	F→RS(15)+Q
PC=3	RS(11)=F		F→RS(12)+Q	RS(13)=F		F→RS(14)+Q
PC=4	RS(11)=F		F→RS(12)+RS(11)	RS(13)=F		F→RS(14)+RS(13)
PC=5		Q=F	F→RS(12)+Q			F→Q
PC=6	DO NOTHING			DO NOTHING		

START ADDRESS FOR PROGRAM:
PROGRAM-MEMORY LOCATIONS USED:

*** CORRELATOR SIMULATOR ***

PROGRAM NAME : SINGLE PULSE SUBROUTINE (NO. OF LAGS .EQ. NO. OF SAMPLES)

CODING (INTEGER) OF SEPARATE CORRELATOR FUNCTIONS

PROGRAM-INSTRUCTIONS DEFINED

MEM-LOC.	COND.CODE	CODE-B	CODE-A	ADDR.	LC1	LCR1A	LC2	LC3	RELOAD	R-ADDR.
0	56	0	4	0	0	0	0	0	0	0
1	56	0	4	0	2	0	3	0	0	0
2	57	6	4	6	6	0	0	0	0	0
3	57	6	4	5	7	1	0	0	0	0
4	57	4	6	4	7	0	0	0	0	0
5	57	4	6	3	6	0	0	0	0	0
6	58	1	6	2	0	0	1	0	0	0

APB-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.	SELECT
0	7	5	1	0	0	0
1	0	1	0	13	0	0
2	0	0	0	13	0	0
3	0	0	3	12	11	0
4	1	0	3	12	11	0
5	0	0	0	12	0	0
6	7	5	1	0	0	0

APM-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.
0	7	5	1	0	0
1	0	1	0	15	0
2	0	0	0	15	0
3	0	0	3	14	13
4	1	0	3	14	13
5	2	0	1	0	0
6	7	5	1	0	0

ARITHMETICAL INSTRUCTIONS DEFINED

MEM-LOC.	M1A	M1B	M2A	M2B	M3A	M3B	M4A	M4B	SM1	SM2	SM3	SM4	M12	M34
0	0	0	0	0	0	0	0	0	0	0	0	0	15	15
1	0	0	0	0	0	0	0	0	0	0	0	0	15	15
2	0	0	1	1	1	0	0	1	3	3	3	3	9	6
3	0	0	1	1	1	0	0	1	2	2	2	2	9	6
4	0	0	1	1	1	0	0	1	2	2	2	2	9	6
5	0	0	1	1	1	0	0	1	3	3	3	3	9	6
6	0	0	0	0	0	0	0	0	0	0	0	0	15	15

ACCUMULATOR INSTRUCTIONS DEFINED

MEM-LOC.	STROBE	I/O WRITE	READ	CLEAR1	SET1	CLEAR2	SET2
0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0
2	1	1	1	1	0	0	0
3	1	1	1	0	0	0	0
4	1	1	1	0	0	0	0
5	1	1	1	0	1	0	0
6	1	0	0	0	0	0	0

I/O INSTRUCTIONS DEFINED

MEM-LOC.	SETF	CLEARF	SELECT BUF.ADDR.	STROBE I-REG.	ENABLE EDB	ENABLE EAB
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0

OUTPUT-TRANSFER INSTRUCTIONS DEFINED

MEM-LOC.	TRANSFER	INHIBIT CLOCK	DATA-READY	TRANSFER-CODE	SOURCE
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: SINGLE PULSE SUBROUTINE (NO. OF LAGS .LE. NO. OF SAMPLES)
FILE-NAME (NORD 10): PROG3:DATA
PROGRAM DESCRIPTION:

DATA CHANNEL 1

$$Re\left\{K_{L,r}\right\} = \sum_{i=0}^{N-L-1} \left(X_{i+(N+D-1)(r-1)} X_{i+L+(N+D-1)(r-1)} + Y_{i+(N+D-1)(r-1)} Y_{i+L+(N+D-1)(r-1)}\right)$$

DATA CHANNEL 2

$$Im\left\{K_{L,r}\right\} = \sum_{i=0}^{N-L-1} \left(X_{i+L+(N+D-1)(r-1)} Y_{i+(N+D-1)(r-1)} - X_{i+(N+D-1)(r-1)} Y_{i+L+(N+D-1)(r-1)}\right)$$

WHERE N=NO. OF SAMPLES IN RANGECELL
L=0,1,2,...,P P≤N-1
D=OVERLAP FACTOR (=1 FOR NO OVERLAPPING AND ≤0 FOR OVERLAPPING)
r=1,2,...,M RANGECELLS FOR TIME AVERAGE

RESTRICTIONS

MINIMUM NO. OF SAMPLES IN RANGEDATA: 1
MINIMUM NO. OF LAGS IN RANGEDATA: 2

NOTES

1. The formulae above are given with respect to the way in which the X, Y samples are read from the buffer memory.
2. The Q registers of the APB, APM processors must be defined with the start addresses of the buffer, result memories in the main program.
3. The LCR1 register must be reloaded with the APBRS(15) register, the LCR2 register must be reloaded with the APBRS(14) register and the LCR3 register must be reloaded with the the APBRS(13) register in the main program.

START ADDRESS FOR PROGRAM: 1
PROGRAM-MEMORY LOCATIONS USED: 1 - 7

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: SINGLE PULSE SUBROUTINE (NO. OF LAGS .LE. NO. OF SAMPLES)
FILE-NAME (NORD 10): PROG3:DATA

REGISTER NAME	REGISTER ADDRESS	PARAMETER
SAR	4	START ADDRESS OF SUBROUTINE
APB RS(15)	16,15	NO. OF SAMPLES-1 IN RANGECELL
APB RS(14)	16,14	NO. OF RANGECELLS-1 FOR TIME AVERAGE
APB RS(13)	16,13	NO. OF LAGS-1 IN RANGECELL
APB RS(12)	16,12	RANGECELL INCREMENT (=1 FOR NO OVER-LAPPING OF RANGECELLS)
APB RS(11)	16,11	SAMPLE INCREMENT (NORMALLY=1)
APB RS(10)	16,10	TEMPORARY STORAGE
APM RS(15)	17,15	RANGECELL INCREMENT (=NO. OF LAGS COMPUTED)
APM RS(14)	17,14	INCREMENT (=1)
APM RS(13)	17,13	TEMPORARY STORAGE

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: SINGLE PULSE SUBROUTINE (NO. OF LAGS .LE. NO. OF SAMPLES)
FILE-NAME (NORD 10): PROG3:DATA
PROGRAM DESCRIPTION:

	APB PROCESSOR		APM PROCESSOR	
PC=1	Q=F	F→Q-RS(12)	Q=F	F→Q-RS(15)
PC=2	Q=F	F→RS(12)+Q	Q=F	F→RS(15)+Q
PC=3	RS(10)=F	F→RS(11)+Q	RS(13)=F	F→RS(14)+Q
PC=4	RS(10)=F	F→RS(11)+RS(10)	RS(13)=F	F→RS(14)+RS(13)
PC=5	DO NOTHING		DO NOTHING	
PC=6	Q=F	F→RS(11)+Q		F→Q
PC=7	DO NOTHING		DO NOTHING	

START ADDRESS FOR PROGRAM:
PROGRAM-MEMORY LOCATIONS USED:

*** CORRELATOR SIMULATOR ***

PROGRAM NAME : SINGLE PULSE SUBROUTINE (NO. OF LAGS .LE. NO. OF SAMPLES)

CODING (INTEGER) OF SEPARATE CORRELATOR FUNCTIONS

PROGRAM-INSTRUCTIONS DEFINED

MEM-LOC.	COND.CODE	CODE-B	CODE-A	ADDR.	LC1	LCR1A	LC2	LC3	RELOAD	R-ADDR.
0	56	0	4	0	0	0	0	0	0	0
1	56	0	4	0	2	0	3	2	0	0
2	61	6	4	7	1	0	0	1	0	0
3	61	6	4	5	1	1	0	1	0	0
4	61	4	6	4	1	0	0	1	0	0
5	56	0	4	0	3	0	0	2	0	0
6	57	4	6	3	6	0	0	1	0	0
7	58	1	6	2	2	0	1	2	0	0

APB-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.	SELECT
0	7	5	1	0	0	0
1	0	1	0	12	0	0
2	0	0	0	12	0	0
3	0	0	3	11	10	0
4	1	0	3	11	10	0
5	7	5	1	0	0	0
6	0	0	0	11	0	0
7	7	5	1	0	0	0

APM-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.
0	7	5	1	0	0
1	0	1	0	15	0
2	0	0	0	15	0
3	0	0	3	14	13
4	1	0	3	14	13
5	7	5	1	0	0
6	2	0	1	0	0
7	7	5	1	0	0

ARITHMETICAL INSTRUCTIONS DEFINED

MEM-LOC.	M1A	M1B	M2A	M2B	M3A	M3B	M4A	M4B	SM1	SM2	SM3	SM4	M12	M34
0	0	0	0	0	0	0	0	0	0	0	0	0	15	15
1	0	0	0	0	0	0	0	0	0	0	0	0	15	15
2	0	0	1	1	1	0	0	1	3	3	3	3	9	6
3	0	0	1	1	1	0	0	1	2	2	2	2	9	6
4	0	0	1	1	1	0	0	1	2	2	2	2	9	6
5	0	0	0	0	0	0	0	0	0	0	0	0	15	15
6	0	0	1	1	1	0	0	1	3	3	3	3	9	6
7	0	0	0	0	0	0	0	0	0	0	0	0	15	15

ACCUMULATOR INSTRUCTIONS DEFINED

MEM-LOC.	STROBE I/O	WRITE	READ	CLEAR1	SET1	CLEAR2	SET2
0	0	0	0	0	0	0	
1	1	0	0	0	0	0	
2	1	1	1	1	0	0	
3	1	1	1	0	0	0	
4	1	1	1	0	0	0	
5	1	0	0	0	0	0	
6	1	1	1	0	1	0	
7	1	0	0	0	0	0	

I/O INSTRUCTIONS DEFINED

MEM-LOC.	SETF	CLEARF	SELECT BUF.ADDR.	STROBE I-REG.	ENABLE EDB	ENABLE EAF
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0
7	0	0	0	0	0	0

OUTPUT-TRANSFER INSTRUCTIONS DEFINED

MEM-LOC.	TRANSFER	INHIBIT CLOCK	DATA-READY	TRANSFER-CODE	SOURCE
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: MULTI PULSE SUBROUTINE
FILE-NAME (NORD 10): PROG4:DATA
PROGRAM DESCRIPTION:

DATA CHANNEL 1 $\mathrm{Re}\{K_{L,r}\} = X_{S+r-1}X_{S+L+r-1} + Y_{S+r-1}Y_{S+L+r-1}$

DATA CHANNEL 2 $\mathrm{Im}\{K_{L,r}\} = X_{S+L+r-1}Y_{S+r-1} - X_{S+r-1}Y_{S+L+r-1}$

LET J_0=POSITION OF 1st PULSE
J_1=SAMPLE DIFFERENCE BETWEEN 1st AND 2nd PULSE
:
J_{N-1}=SAMPLE DIFFERENCE BETWEEN 1st AND Nth PULSE

WHERE $S=J_0,J_1,J_2,\ldots,J_{N-2}$
$L=J_1-S,J_2-S,\ldots,J_{N-1}-S$ WITH THE RESTRICTION $L>0$
$r=1,2,\ldots,M$ RANGECELLS FOR TIME AVERAGE

RESTRICTIONS

MINIMUM NO. OF ELEMENT PULSES IN PULSE GROUP: 2
MAXIMUM NO. OF ELEMENT PULSES IN PULSE GROUP: 13

NOTES

1. The formulae above are given with respect to the way in which the X, Y samples are read from the buffer memory.
2. The Q registers of the APB, APM processors must be defined with the start addresses of the buffer, result memories in the main program.
3. The LCR1 register must be reloaded with the APBRS(15) register and the LCR2 register must be reloaded with the APBRS(14) register in the main program.
4. The zero lag is not computed in this algorithm, therefore to calculate the number of lags computed use the formula: $N(N-1)/2$ where N is the number of element pulses in the pulse group.

START ADDRESS FOR PROGRAM: 1
PROGRAM-MEMORY LOCATIONS USED: 1 - 6

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: MULTI PULSE SUBROUTINE
FILE-NAME (NORD 10): PROG4:DATA

REGISTER NAME	REGISTER ADDRESS	PARAMETER
SAR	4	START ADDRESS OF SUBROUTINE
APB RS(15)	16,15	NO. OF PULSES-1 IN RANGECELL
APB RS(14)	16,14	NO. OF RANGECELLS-1 FOR TIME AVERAGE
APB RS(13)	16,13	RANGECELL INCREMENT (NORMALLY=1)
APB RS(12)	16,12	TEMPORARY STORAGE
⋮	⋮	⋮
APB RS(1)	16,1	SAMPLE DISTANCE BETWEEN 2nd LAST AND 1st PULSE
APB RS(0)	16,0	SAMPLE DISTANCE BETWEEN LAST AND 1st PULSE
APM RS(15)	17,15	INCREMENT (=1)

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: MULTI PULSE SUBROUTINE
FILE-NAME (NORD 10): PROG4:DATA
PROGRAM DESCRIPTION:

	APB PROCESSOR		APM PROCESSOR	
PC=1	RS(12)=F	F→Q-RS(13)	Q=F	F→Q-RS(15)
PC=2	RS(12)=F	F→RS(13)+RS(12)	DO NOTHING	
PC=3		F→RS(12)+RS(LC1)	Q=F	F→RS(15)+Q
PC=4		F→RS(12)+RS(LC1)	Q=F	F→RS(15)+Q
PC=5	Q=F	F→RS(12)+RS(LC1)	DO NOTHING	
PC=6	DO NOTHING		DO NOTHING	

START ADDRESS FOR PROGRAM:
PROGRAM-MEMORY LOCATIONS USED:

*** CORRELATOR SIMULATOR ***

PROGRAM NAME : MULTI PULSE SUBROUTINE

CODING (INTEGER) OF SEPARATE CORRELATOR FUNCTIONS

PROGRAM-INSTRUCTIONS DEFINED

MEM-LOC.	COND.CODE	CODE-B	CODE-A	ADDR.	LC1	LCR1A	LC2	LC3	RELOAD	R-ADDR.
0	56	0	4	0	0	0	0	0	0	0
1	56	0	4	0	2	0	3	0	0	0
2	57	6	4	6	6	0	0	0	0	0
3	57	6	4	5	7	1	0	0	0	0
4	57	4	6	4	7	0	0	0	0	0
5	57	4	6	3	6	0	0	0	0	0
6	58	1	6	2	0	0	1	0	0	0

APB-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.	SELECT
0	7	5	1	0	0	0
1	0	1	3	13	12	0
2	1	0	3	13	12	0
3	1	0	1	12	0	1
4	1	0	1	12	0	1
5	1	0	0	12	0	1
6	7	5	1	0	0	0

APM-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.
0	7	5	1	0	0
1	0	1	0	15	0
2	7	5	1	0	0
3	0	0	0	15	0
4	0	0	0	15	0
5	7	5	1	0	0
6	7	5	1	0	0

ARITHMETICAL INSTRUCTIONS DEFINED

MEM-LOC.	M1A	M1B	M2A	M2B	M3A	M3B	M4A	M4B	SM1	SM2	SM3	SM4	M12	M34
0	0	0	0	0	0	0	0	0	0	0	0	0	15	15
1	0	0	0	0	0	0	0	0	0	0	0	0	15	15
2	0	0	1	1	1	0	0	1	3	3	3	3	9	6
3	0	0	1	1	1	0	0	1	2	2	2	2	9	6
4	0	0	1	1	1	0	0	1	2	2	2	2	9	6
5	0	0	1	1	1	0	0	1	3	3	3	3	9	6
6	0	0	0	0	0	0	0	0	0	0	0	0	15	15

ACCUMULATOR INSTRUCTIONS DEFINED

MEM-LOC.	STROBE I/O WRITE	READ	CLEAR1	SET1	CLEAR2	SET2
0	0	0	0	0	0	0
1	1	0	0	1	0	0
2	1	0	0	0	0	0
3	1	1	1	0	0	0
4	1	1	1	0	0	0
5	1	0	0	0	0	0
6	1	0	0	0	0	0

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: CROSS CORRELATION SUBROUTINE (NO. OF LAGS .EQ. NO. OF SAMPLES)
FILE-NAME (NORD 10): PROG5:DATA
PROGRAM DESCRIPTION:

DATA CHANNEL 1

$$\mathrm{Re}\left\{K_{L,r}\right\} = \sum_{i=0}^{N-L-1} \left(X_{i+(N+D-1)(r-1)} X_{i+S+L+(N+D-1)(r-1)} + Y_{i+(N+D-1)(r-1)} Y_{i+S+L+(N+D-1)(r-1)}\right)$$

DATA CHANNEL 2

$$\mathrm{Im}\left\{K_{L,r}\right\} = \sum_{i=0}^{N-L-1} \left(X_{i+S+L+(N+D-1)(r-1)} Y_{i+(N+D-1)(r-1)} - X_{i+(N+D-1)(r-1)} Y_{i+S+L+(N+D-1)(r-1)}\right)$$

WHERE N=NO. OF SAMPLES IN RANGECELL
L=0,1,2,...,N-1
S=SAMPLE DIFFERENCE BETWEEN THE TWO SETS OF DATA
D=OVERLAP FACTOR (=1 FOR NO OVERLAPPING AND ≤0 FOR OVERLAPPING)
r=1,2,...,M RANGECELLS FOR TIME AVERAGE

RESTRICTIONS

MINIMUM NO. OF SAMPLES IN RANGEDATA: 1

NOTES

1. The formulae above are given with respect to the way in which the X, Y samples are read from the buffer memory.
2. The Q registers of the APB, APM processors must be defined with the start addresses of the buffer, result memories in the main program.
3. The LCR1 register must be reloaded with the APBRS(15) register and the LCR2 register must be reloaded with the APBRS(14) register in the main program.
4. Only half of the correlation function can be obtained with this scheme.
5. The Single Pulse autocorrelation scheme (see PROG2:DATA) can be obtained by setting APBRS(12)=0.

START ADDRESS FOR PROGRAM: 1
PROGRAM-MEMORY LOCATIONS USED: 1 - 8

I/O INSTRUCTIONS DEFINED

MEM-LOC.	SETF	CLEARF	SELECT BUF.ADDR.	STROBE I-REG.	ENABLE EDB	ENABLE EA
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0

OUTPUT-TRANSFER INSTRUCTIONS DEFINED

MEM-LOC.	TRANSFER	INHIBIT CLOCK	DATA-READY	TRANSFER-CODE	SOURCE
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: CROSS CORRELATION SUBROUTINE (NO. OF LAGS .EQ. NO. OF SAMPLES)
FILE-NAME (NORD 10): PROG5:DATA

REGISTER NAME	REGISTER ADDRESS	PARAMETER
SAR	4	START ADDRESS OF SUBROUTINE
APB RS(15)	16,15	NO. OF SAMPLES-1 IN RANGECELL
APB RS(14)	16,14	NO. OF RANGECELLS-1 FOR TIME AVERAGE
APB RS(13)	16,13	RANGECELL INCREMENT (=1 FOR NO OVER-LAPPING OF RANGECELLS)
APB RS(12)	16,12	START ADDRESS OF 2nd FIELD-START ADDRESS OF 1st FIELD
APB RS(11)	16,11	SAMPLE INCREMENT (NORMALLY=1)
APB RS(10)	16,10	TEMPORARY STORAGE
APM RS(15)	17,15	RANGECELL INCREMENT (=NO. OF LAGS COMPUTED)
APM RS(14)	17,14	INCREMENT (=1)
APM RS(13)	17,13	TEMPORARY STORAGE

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: CROSS CORRELATION SUBROUTINE (NO. OF LAGS .EQ. NO. OF SAMPLES)
FILE-NAME (NORD 10): PROG5:DATA
PROGRAM DESCRIPTION:

	APB PROCESSOR		APM PROCESSOR	
PC=1	Q=F	F→Q-RS(13)	Q=F	F→Q-RS(15)
PC=2	Q=F	F→RS(13)+Q	DO NOTHING	
PC=3	RS(10)=F	F→RS(12)+Q	Q=F	F→RS(15)+Q
PC=4	RS(10)=F	F→RS(11)+RS(10)	RS(13)=F	F→RS(14)+Q
PC=5	RS(10)=F	F→RS(11)+RS(10)	RS(13)=F	F→RS(14)+RS(13)
PC=6	Q=F	F→RS(11)+Q	DO NOTHING	
PC=7	RS(10)=F	F→RS(12)+Q		F→Q
PC=8	DO NOTHING		DO NOTHING	

START ADDRESS FOR PROGRAM:
PROGRAM-MEMORY LOCATIONS USED:

*** CORRELATOR SIMULATOR ***

PROGRAM NAME : CROSS CORRELATION SUBROUTINE (NO. OF LAGS .EQ. NO OF SAMPLES.

CODING (INTEGER) OF SEPARATE CORRELATOR FUNCTIONS

PROGRAM-INSTRUCTIONS DEFINED

MEM-LOC.	COND.CODE	CODE-B	CODE-A	ADDR.	LC1	LCR1A	LC2	LC3	RELOAD	R-ADDR.
0	56	0	4	0	0	0	0	0	0	0
1	56	0	4	0	2	0	3	0	0	0
2	56	0	4	0	0	0	0	0	0	0
3	57	6	4	8	6	0	0	0	0	0
4	57	6	4	6	7	1	0	0	0	0
5	57	4	6	5	7	0	0	0	0	0
6	56	0	4	0	0	0	0	0	0	0
7	57	4	6	4	6	0	0	0	0	0
8	58	1	6	2	0	0	1	0	0	0

APB-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.	SELECT
0	7	5	1	0	0	0
1	0	1	0	13	0	0
2	0	0	0	13	0	0
3	0	0	3	12	10	0
4	1	0	3	11	10	0
5	1	0	3	11	10	0
6	0	0	0	11	0	0
7	0	0	3	12	10	0
8	7	5	1	0	0	0

APM-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.
0	7	5	1	0	0
1	0	1	0	15	0
2	7	5	1	0	0
3	0	0	0	15	0
4	0	0	3	14	13
5	1	0	3	14	13
6	7	5	1	0	0
7	2	0	1	0	0
8	7	5	1	0	0

ARITHMETICAL INSTRUCTIONS DEFINED

MEM-LOC.	M1A	M1B	M2A	M2B	M3A	M3B	M4A	M4B	SM1	SM2	SM3	SM4	M12	M34
0	0	0	0	0	0	0	0	0	0	0	0	0	15	15
1	0	0	0	0	0	0	0	0	0	0	0	0	15	15
2	0	0	1	1	1	0	0	1	1	1	1	1	9	6
3	0	0	1	1	1	0	0	1	2	2	2	2	9	6
4	0	0	1	1	1	0	0	1	2	2	2	2	9	6
5	0	0	1	1	1	0	0	1	2	2	2	2	9	6
6	0	0	1	1	1	0	0	1	1	1	1	1	9	6
7	0	0	1	1	1	0	0	1	2	2	2	2	9	6
8	0	0	0	0	0	0	0	0	0	0	0	0	15	15

ACCUMULATOR INSTRUCTIONS DEFINED

MEM-LOC.	STROBE I/O WRITE	READ	CLEAR1	SET1	CLEAR2	SET2
0	0	0	0	0	0	0
1	1	0	1	0	0	0
2	1	0	0	0	0	0
3	1	1	1	0	0	0
4	1	1	0	0	0	0
5	1	1	0	0	0	0
6	1	0	0	0	0	0
7	1	1	0	1	0	0
8	1	0	0	0	0	0

I/O INSTRUCTIONS DEFINED

MEM-LOC.	SETF	CLEARF	SELECT BUF.ADDR.	STROBE I-REG.	ENABLE EDB	ENABLE EAB
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0
7	0	0	0	0	0	0
8	0	0	0	0	0	0

OUTPUT-TRANSFER INSTRUCTIONS DEFINED

MEM-LOC.	TRANSFER	INHIBIT CLOCK	DATA-READY	TRANSFER-CODE	SOURCE
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: TRANSFER PROGRAM
FILE-NAME (NORD 10): PROG27:DATA
PROGRAM DESCRIPTION:

The Transfer program enables the transfer of the 64 BIT words of the correlator result memory as 4x16 BIT words to the NORD 10 computer. The way in which the 64 BIT words are split up and the order in which they are transferred are:

TRANSFERRED 1st	MOST SIGNIFICANT BITS	DATA CHANNEL 1
TRANSFERRED 2nd	LEAST SIGNIFICANT BITS	DATA CHANNEL 1
TRANSFERRED 3rd	MOST SIGNIFICANT BITS	DATA CHANNEL 2
TRANSFERRED 4th	LEAST SIGNIFICANT BITS	DATA CHANNEL 2

The STATUS and CONTROL words of the correlator are transferred out before the data to the NORD 10 computer. The transfer of these two words are set in the program and must not be included in the parameter for the number of 64 BIT words for DMA transfer.

START ADDRESS FOR PROGRAM: 32
PROGRAM-MEMORY LOCATIONS USED: 32 - 44

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: TRANSFER PROGRAM
FILE-NAME (NORD 10): PROG27:DATA

REGISTER NAME	REGISTER ADDRESS	PARAMETER
DATAI REGISTER, APB	6	NO. OF 64 BIT WORDS FOR DMA TRANSFER
APM RS(0)	17,0	INCREMENT (=1)

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: TRANSFER PROGRAM
FILE-NAME (NORD 10): PROG27:DATA
PROGRAM DESCRIPTION:

	APB PROCESSOR	APM PROCESSOR	
PC=32	F→DATAI	DO NOTHING	
PC=33	DO NOTHING	DO NOTHING	
PC=34	DO NOTHING	DO NOTHING	
PC=35	DO NOTHING	Q=F	F→$\overline{\text{DATAI}}$ .AND. 0
PC=36	DO NOTHING	Q=F	F→Q-RS(0)
PC=37	DO NOTHING	Q=F	F→RS(0)+Q
PC=38	DO NOTHING		F→Q
PC=39	DO NOTHING		F→Q
PC=40	DO NOTHING		F→Q
PC=41	DO NOTHING	DO NOTHING	
PC=42	DO NOTHING	DO NOTHING	
PC=43	DO NOTHING	DO NOTHING	
PC=44	DO NOTHING	DO NOTHING	

START ADDRESS FOR PROGRAM:
PROGRAM-MEMORY LOCATIONS USED:

*** CORRELATOR SIMULATOR ***

PROGRAM NAME : TRANSFER PROGRAM WITH STATUS AND CONTROL WORDS X-FERRED

CODING (INTEGER) OF SEPARATE CORRELATOR FUNCTIONS

PROGRAM-INSTRUCTIONS DEFINED

MEM-LOC.	COND.CODE	CODE-B	CODE-A	ADDR.	LC1	LCR1A	LC2	LC3	RELOAD	R-ADDR.
0	56	0	4	0	0	0	0	0	0	0
32	56	0	4	0	0	0	0	0	1	20
33	56	0	4	0	0	0	0	0	0	0
34	56	0	4	0	0	0	0	0	0	0
35	56	0	4	0	0	0	0	0	0	0
36	56	0	8	0	0	0	0	2	0	0
37	56	0	4	0	0	0	0	1	0	0
38	56	0	4	0	0	0	0	0	0	0
39	56	0	4	0	0	0	0	0	0	0
40	60	0	5	0	0	0	0	0	0	0
41	56	0	4	0	0	0	0	0	0	0
42	56	0	4	0	0	0	0	0	0	0
43	56	0	4	0	0	0	0	0	0	0
44	56	0	6	0	0	0	0	0	0	0

APB-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.	SELECT
0	7	5	1	0	0	0
32	7	0	1	0	0	0
33	7	5	1	0	0	0
34	7	5	1	0	0	0
35	7	5	1	0	0	0
36	7	5	1	0	0	0
37	7	5	1	0	0	0
38	7	5	1	0	0	0
39	7	5	1	0	0	0
40	7	5	1	0	0	0
41	7	5	1	0	0	0
42	7	5	1	0	0	0
43	7	5	1	0	0	0
44	7	5	1	0	0	0

APM-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.
0	7	5	1	0	0
32	7	5	1	0	0
33	7	5	1	0	0
34	7	5	1	0	0
35	7	5	0	0	0
36	0	1	0	0	0
37	0	0	0	0	0
38	2	0	1	0	0
39	2	0	1	0	0
40	2	0	1	0	0
41	7	5	1	0	0
42	7	5	1	0	0
43	7	5	1	0	0
44	7	5	1	0	0

ARITHMETICAL INSTRUCTIONS DEFINED

MEM-LOC.	M1A	M1B	M2A	M2B	M3A	M3B	M4A	M4B	SM1	SM2	SM3	SM4	M12	M34
0	0	0	0	0	0	0	0	0	0	0	0	0	15	15
32	0	0	0	0	0	0	0	0	0	0	0	0	15	15
33	0	0	0	0	0	0	0	0	0	0	0	0	15	15
34	0	0	0	0	0	0	0	0	0	0	0	0	15	15
35	0	0	0	0	0	0	0	0	0	0	0	0	15	15
36	0	0	0	0	0	0	0	0	0	0	0	0	15	15
37	0	0	0	0	0	0	0	0	0	0	0	0	15	15
38	0	0	0	0	0	0	0	0	0	0	0	0	15	15
39	0	0	0	0	0	0	0	0	0	0	0	0	15	15
40	0	0	0	0	0	0	0	0	0	0	0	0	15	15
41	0	0	0	0	0	0	0	0	0	0	0	0	15	15
42	0	0	0	0	0	0	0	0	0	0	0	0	15	15
43	0	0	0	0	0	0	0	0	0	0	0	0	15	15
44	0	0	0	0	0	0	0	0	0	0	0	0	15	15

ACCUMULATOR INSTRUCTIONS DEFINED

MEM-LOC.	STROBE I/O	WRITE	READ	CLEAR1	SET1	CLEAR2	SET2
0	0	0	0	0	0	0	0
32	0	0	0	0	0	1	0
33	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0

I/O INSTRUCTIONS DEFINED

MEM-LOC.	SETF	CLEARF	SELECT	BUF.ADDR.	STROBE I-REG.	ENABLE EDB	ENABLE EA
0	0	0	0		0	0	0
32	0	0	0		0	0	0
33	0	0	0		0	0	0
34	0	0	0		0	0	0
35	0	0	0		0	0	0
36	0	0	0		0	0	0
37	0	0	0		0	0	0
38	0	0	0		0	0	0
39	0	0	0		0	0	0
40	0	0	0		0	0	0
41	0	0	0		0	0	0
42	0	0	0		0	0	0
43	0	0	0		0	0	0
44	0	0	0		0	0	0

OUTPUT-TRANSFER INSTRUCTIONS DEFINED

MEM-LOC.	TRANSFER	INHIBIT CLOCK	DATA-READY	TRANSFER-CODE	SOURCE
0	0	0	0	0	0
32	0	0	0	0	0
33	0	0	0	0	0
34	1	0	0	0	0
35	1	1	1	0	0
36	1	1	1	1	0
37	1	1	1	3	0
38	1	1	1	2	0
39	1	1	1	5	0
40	1	1	1	4	0
41	1	0	0	0	0
42	1	0	0	0	0
43	1	0	0	0	0
44	1	0	0	0	0

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: POWER PROFILE (VER.1) OR SINGLE PULSE OR CROSS CORRELATION
FILE-NAME (NORD 10): GP1:DATA
PROGRAM DESCRIPTION:

POWER PROFILE

DATA CHANNEL 1 ZERO LAG ESTIMATION $K_r = \sum_{i=0}^{N-1} (X^2_{i+(N+D-1)(r-1)} + Y^2_{i+(N+D-1)(r-1)})$

DATA CHANNEL 2 MEAN VALUE ESTIMATION $M_r = \sum_{i=0}^{N-1} (X_{i+(N+D-1)(r-1)} + Y_{i+(N+D-1)(r-1)})$

WHERE N=NO. OF SAMPLES IN RANGECELL
D=OVERLAP FACTOR (=1 FOR NO OVERLAPPING AND $\leq$0 FOR OVERLAPPING)
r=1,2,...,M RANGECELLS FOR TIME AVERAGE

SINGLE PULSE

DATA CHANNEL 1

$$Re\{K_{L,r}\} = \sum_{i=0}^{N-L-1} (X_{i+(N+D-1)(r-1)} X_{i+L+(N+D-1)(r-1)} + Y_{i+(N+D-1)(r-1)} Y_{i+L+(N+D-1)(r-1)})$$

DATA CHANNEL 2

$$Im\{K_{L,r}\} = \sum_{i=0}^{N-L-1} (X_{i+L+(N+D-1)(r-1)} Y_{i+(N+D-1)(r-1)} - X_{i+(N+D-1)(r-1)} Y_{i+L+(N+D-1)(r-1)})$$

WHERE N=NO. OF SAMPLES IN RANGECELL
L=0,1,2,...,P P$\leq$N-1
D=OVERLAP FACTOR (=1 FOR NO OVERLAPPING AND $\leq$0 FOR OVERLAPPING)
r=1,2,...,M RANGECELLS FOR TIME AVERAGE

START ADDRESS FOR PROGRAM: 1
PROGRAM-MEMORY LOCATIONS USED: 1 - 26, 32 - 44

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: POWER PROFILE (VER.1) OR SINGLE PULSE OR CROSS CORRELATION
FILE-NAME (NORD 10): GP1:DATA
PROGRAM DESCRIPTION:

CROSS CORRELATION

DATA CHANNEL 1

$$\mathrm{Re}\left\{K_{L,r}\right\} = \sum_{i=0}^{N-L-1} \left(X_{i+(N+D-1)(r-1)} X_{i+S+L+(N+D-1)(r-1)} + Y_{i+(N+D-1)(r-1)} Y_{i+S+L+(N+D-1)(r-1)}\right)$$

DATA CHANNEL 2

$$\mathrm{Im}\left\{K_{L,r}\right\} = \sum_{i=0}^{N-L-1} \left(X_{i+S+L+(N+D-1)(r-1)} Y_{i+(N+D-1)(r-1)} - X_{i+(N+D-1)(r-1)} Y_{i+S+L+(N+D-1)(r-1)}\right)$$

WHERE N=NO. OF SAMPLES IN RANGECELL
L=0,1,2,...,P P≤N-1
S=SAMPLE DIFFERENCE BETWEEN THE TWO SETS OF DATA
D=OVERLAP FACTOR (=1 FOR NO OVERLAPPING AND≤0 FOR OVERLAPPING)
r=1,2,...,M RANGECELLS FOR TIME AVERAGE

RESTRICTIONS

POWER PROFILE: MINIMUM NO. OF SAMPLES IN RANGEDATA: 1

SINGLE PULSE: MINIMUM NO. OF SAMPLES IN RANGEDATA: 1
MINIMUM NO. OF LAGS IN RANGEDATA: 2

CROSS CORRELATION: MINIMUM NO. OF SAMPLES IN RANGEDATA: 1
MINIMUM NO. OF LAGS IN RANGEDATA: 2

START ADDRESS FOR PROGRAM:
PROGRAM-MEMORY LOCATIONS USED:

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: POWER PROFILE (VER.1) OR SINGLE PULSE OR CROSS CORRELATION
FILE-NAME (NORD 10): GP1:DATA
PROGRAM DESCRIPTION:

NOTES

1. This program assumes that the start addresses of the buffer and result memories are zero.
2. The number of start computes counted is placed after the last data sample in the result memory. Therefore the real and imaginary parts of this location will contain the number of start computes counted as a negative number.
3. The location for the number of start computes counted must be included in the number of 64 BIT words for the DMA transfer, ie total number samples computed+1 (for number of start computes counted).
4. The Status and Control words are transferred out before the data to the computer.
5. With the Cross Correlation program only half of the correlation function can be obtained.
6. The data, noise and calibration samples may be computed in any order, eg for a given experiment data may be computed 1st, then noise 2nd and then calibration 3rd or noise 1st, data 2nd and calibration 3rd etc.
7. All X, Y samples must be placed contiguously in the buffer memory and all samples are written contiguously in the result memory.
8. In this scheme only one routine may be chosen for a particular experiment.

HOW TO GET THE ROUTINES

POWER PROFILE: DEFINE APBRS(13)=0
SINGLE PULSE: DEFINE APBRS(11)=0
CROSS CORRELATION: DEFINE APBRS(13)≠0 AND APBRS(11)≠0

START ADDRESS FOR PROGRAM:
PROGRAM-MEMORY LOCATIONS USED:

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO DATE: 6/8/80

PROGRAM NAME: POWER PROFILE (VER.1) OR SINGLE PULSE OR CROSS CORRELATION
FILE-NAME (NORD 10): GP1:DATA
PROGRAM DESCRIPTION:

EXECUTION TIMES FOR PROGRAMS

POWER PROFILE: $(28+2(R_1+R_2+R_3))\times 0.1666667$ usec N=1

$(28+(N+2)(R_1+R_2+R_3))\times 0.1666667$ usec $N \geq 2$

SINGLE PULSE AND CROSS CORRELATION:

$(22+(2N+(L(1-L)/2)+N(L-1)+1)(R_1+R_2+R_3))\times 0.1666667$ usec $N \geq 2$, $L \geq 2$

WHERE N=NO. OF SAMPLES IN RANGECELL
L=NO. OF LAGS IN RANGECELL
R_1, R_2, R_3 =NO. OF RANGECELLS FOR DATA, NOISE AND CALIBRATION

START ADDRESS FOR PROGRAM:
PROGRAM-MEMORY LOCATIONS USED:

MICRO-PROGRAM FOR DIGITAL CORRELATOR

AUTOR: TERRANCE HO

DATE: 6/8/80

PROGRAM NAME: POWER PROFILE (VER.1) OR SINGLE PULSE OR CROSS CORRELATION
FILE-NAME (NORD 10): GP1:DATA

REGISTER NAME	REGISTER ADDRESS	PARAMETER
SAR	4	START ADDRESS OF PROGRAM
DATAI REGISTER, APB	6	NO. OF 64 BIT WORDS FOR DMA TRANSFER
APB RS(15)	16,15	NO. OF SAMPLES-1 IN RANGECELL (DATA AND NOISE AND CALIBRATION)
APB RS(14)	16,14	NO. OF RANGECELLS-1 FOR TIME AVERAGE (DATA OR NOISE OR CALIBRATION) COMPUTED 1st
APB RS(13)	16,13	NO. OF LAGS-1 IN RANGECELL (DATA AND NOISE AND CALIBRATION)
APB RS(12)	16,12	RANGECELL INCREMENT (=1 FOR NO OVER-LAPPING OF RANGECELLS)
APB RS(11)	16,11	START ADDRESS OF 2nd FIELD- START ADDRESS OF 1st FIELD
APB RS(10)	16,10	SAMPLE INCREMENT (NORMALLY=1)
APB RS(9)	16,9	TEMPORARY STORAGE
APB RS(8)	16,8	NO. OF RANGECELLS-1 FOR TIME AVERAGE (DATA OR NOISE OR CALIBRATION) COMPUTED 2nd
APB RS(7)	16,7	NO. OF RANGECELLS-1 FOR TIME AVERAGE (DATA OR NOISE OR CALIBRATION) COMPUTED 3rd
APM RS(15)	17,15	RANGECELL INCREMENT (=NO. OF LAGS COMPUTED)
APM RS(14)	17,14	INCREMENT (=1)
APM RS(13)	17,13	TEMPORARY STORAGE
APM RS(0)	17,0	INCREMENT FOR TRANSFER PROGRAM (=1)

*** CORRELATOR SIMULATOR ***

PROGRAM NAME : POWER PROFILE (VER.1) OR SINGLE PULSE OR CROSS CORRELATION PROGRAM

CODING (INTEGER) OF SEPARATE CORRELATOR FUNCTIONS

PROGRAM-INSTRUCTIONS DEFINED

MEM-LOC.	COND.CODE	CODE-B	CODE-A	ADDR.	LC1	LCR1A	LC2	LC3	RELOAD	R-ADDR.
0	56	0	4	0	0	0	0	0	0	0
1	56	0	4	0	0	0	0	0	1	18
2	56	0	4	0	0	0	0	0	0	0
3	56	0	4	0	0	0	0	0	1	19
4	56	0	4	0	0	0	0	0	0	0
5	56	0	4	0	0	0	0	0	1	20
6	56	0	10	13	0	0	0	0	0	0
7	56	0	4	0	0	0	0	0	1	19
8	56	0	10	13	0	0	0	0	0	0
9	56	0	4	0	0	0	0	0	1	19
10	56	0	10	13	0	0	0	0	0	0
11	56	0	4	0	0	0	0	0	0	0
12	56	0	6	0	0	0	0	0	0	0
13	56	0	4	0	2	0	3	2	0	0
14	60	6	4	21	0	0	0	0	0	0
15	61	6	4	20	1	0	0	1	0	0
16	61	6	4	18	1	1	0	1	0	0
17	61	4	6	17	1	0	0	1	0	0
18	56	0	4	0	3	0	0	2	0	0
19	57	4	6	16	6	0	0	1	0	0
20	58	1	6	14	2	0	1	2	0	0
21	56	0	4	0	2	0	3	0	0	0
22	57	0	4	26	1	0	0	0	0	0
23	57	6	4	25	1	0	0	0	0	0
24	57	4	6	24	1	0	0	0	0	0
25	56	0	4	0	0	0	0	0	0	0
26	58	1	6	22	2	0	1	0	0	0
32	56	0	4	0	0	0	0	0	1	20
33	56	0	4	0	0	0	0	0	0	0
34	56	0	4	0	0	0	0	0	0	0
35	56	0	4	0	0	0	0	0	0	0
36	56	0	8	0	0	0	0	2	0	0
37	56	0	4	0	0	0	0	1	0	0
38	56	0	4	0	0	0	0	0	0	0
39	56	0	4	0	0	0	0	0	0	0
40	60	0	5	0	0	0	0	0	0	0
41	56	0	4	0	0	0	0	0	0	0
42	56	0	4	0	0	0	0	0	0	0
43	56	0	4	0	0	0	0	0	0	0
44	56	0	6	0	0	0	0	0	0	0

APB-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.	SELECT
0	7	5	1	0	0	0
1	4	0	1	15	0	0
2	7	5	1	0	0	0
3	4	0	1	14	0	0
4	7	5	1	0	0	0
5	4	0	1	13	0	0
6	7	5	0	0	0	0
7	4	0	1	8	0	0
8	0	0	0	10	0	0
9	4	0	1	7	0	0
10	0	0	0	10	0	0
11	0	0	0	10	0	0
12	7	5	1	0	0	0
13	0	1	0	12	0	0
14	0	0	0	12	0	0
15	0	0	3	11	9	0
16	1	0	3	10	9	0
17	1	0	3	10	9	0
18	0	0	0	10	0	0
19	0	0	3	11	9	0
20	7	5	1	0	0	0
21	0	1	0	12	0	0
22	0	0	0	12	0	0
23	2	0	1	0	0	0
24	0	0	0	10	0	0
25	0	0	0	10	0	0
26	7	5	1	0	0	0
32	7	0	1	0	0	0
33	7	5	1	0	0	0
34	7	5	1	0	0	0
35	7	5	1	0	0	0
36	7	5	1	0	0	0
37	7	5	1	0	0	0
38	7	5	1	0	0	0
39	7	5	1	0	0	0
40	7	5	1	0	0	0
41	7	5	1	0	0	0
42	7	5	1	0	0	0
43	7	5	1	0	0	0
44	7	5	1	0	0	0

APM-INSTRUCTIONS DEFINED

MEM-LOC.	ALU-SOURCE	ALU-FUNCTION	ALU-DESTIN.	A-ADDR.	B-ADDR.
0	7	5	1	0	0
1	7	5	1	0	0
2	7	5	1	0	0
3	7	5	1	0	0
4	7	5	1	0	0
5	7	5	1	0	0
6	7	5	0	0	0
7	7	5	1	0	0
8	0	0	0	15	0
9	7	5	1	0	0
10	0	0	0	15	0
11	0	0	0	15	0
12	7	5	1	0	0
13	0	1	0	15	0
14	7	5	1	0	0
15	0	0	0	15	0
16	0	0	3	14	13
17	1	0	3	14	13
18	7	5	1	0	0
19	2	0	1	0	0
20	7	5	1	0	0
21	7	5	1	0	0
22	0	0	0	15	0
23	7	5	1	0	0
24	7	5	1	0	0
25	2	0	1	0	0
26	7	5	1	0	0
32	7	5	1	0	0
33	7	5	1	0	0
34	7	5	1	0	0
35	7	5	0	0	0
36	0	1	0	0	0
37	0	0	0	0	0
38	2	0	1	0	0
39	2	0	1	0	0
40	2	0	1	0	0
41	7	5	1	0	0
42	7	5	1	0	0
43	7	5	1	0	0
44	7	5	1	0	0

	ARITHMETICAL INSTRUCTIONS DEFINED													
MEM-LOC.	M1A	M1B	M2A	M2B	M3A	M3B	M4A	M4B	SM1	SM2	SM3	SM4	M12	M34
0	0	0	0	0	0	0	0	0	0	0	0	0	15	15
1	0	0	0	0	0	0	0	0	0	0	0	0	15	15
2	0	0	0	0	0	0	0	0	0	0	0	0	15	15
3	0	0	0	0	0	0	0	0	0	0	0	0	15	15
4	0	0	0	0	0	0	0	0	0	0	0	0	15	15
5	0	0	0	0	0	0	0	0	0	0	0	0	15	15
6	0	0	0	0	0	0	0	0	0	0	0	0	15	15
7	0	0	0	0	0	0	0	0	0	0	0	0	15	15
8	0	0	0	0	0	0	0	0	0	0	0	0	15	15
9	0	0	0	0	0	0	0	0	0	0	0	0	15	15
10	0	0	0	0	0	0	0	0	0	0	0	0	15	15
11	0	0	0	0	0	0	0	0	0	0	0	0	12	12
12	0	0	0	0	0	0	0	0	0	0	0	0	15	15
13	0	0	0	0	0	0	0	0	0	0	0	0	15	15
14	0	0	1	1	1	0	0	1	1	1	1	1	9	6
15	0	0	1	1	1	0	0	1	2	2	2	2	9	6
16	0	0	1	1	1	0	0	1	2	2	2	2	9	6
17	0	0	1	1	1	0	0	1	2	2	2	2	9	6
18	0	0	1	1	1	0	0	1	1	1	1	1	9	6
19	0	0	1	1	1	0	0	1	2	2	2	2	9	6
20	0	0	0	0	0	0	0	0	0	0	0	0	15	15
21	0	0	0	0	0	0	0	0	0	0	0	0	15	15
22	0	0	1	1	4	0	4	1	3	3	3	3	9	9
23	0	0	0	0	0	0	0	0	3	3	3	3	6	6
24	0	0	1	1	4	0	4	1	3	3	3	3	9	9
25	0	0	1	1	4	0	4	1	3	3	3	3	9	9
26	0	0	0	0	0	0	0	0	0	0	0	0	15	15
32	0	0	0	0	0	0	0	0	0	0	0	0	15	15
33	0	0	0	0	0	0	0	0	0	0	0	0	15	15
34	0	0	0	0	0	0	0	0	0	0	0	0	15	15
35	0	0	0	0	0	0	0	0	0	0	0	0	15	15
36	0	0	0	0	0	0	0	0	0	0	0	0	15	15
37	0	0	0	0	0	0	0	0	0	0	0	0	15	15
38	0	0	0	0	0	0	0	0	0	0	0	0	15	15
39	0	0	0	0	0	0	0	0	0	0	0	0	15	15
40	0	0	0	0	0	0	0	0	0	0	0	0	15	15
41	0	0	0	0	0	0	0	0	0	0	0	0	15	15
42	0	0	0	0	0	0	0	0	0	0	0	0	15	15
43	0	0	0	0	0	0	0	0	0	0	0	0	15	15
44	0	0	0	0	0	0	0	0	0	0	0	0	15	15

ACCUMULATOR INSTRUCTIONS DEFINED

MEM-LOC.	STROBE I/O	WRITE	READ	CLEAR1	SET1	CLEAR2	SET2
0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0
2	1	0	0	0	0	0	0
3	1	0	0	0	0	0	0
4	1	0	0	0	0	0	0
5	1	0	0	0	0	0	0
6	1	0	0	0	0	0	0
7	1	0	0	0	0	0	0
8	1	0	0	0	0	0	0
9	1	0	0	0	0	0	0
10	1	0	0	0	0	0	0
11	1	1	1	1	0	0	0
12	1	0	0	0	0	0	1
13	1	0	0	1	0	0	0
14	1	0	0	0	0	0	0
15	1	1	1	1	0	0	0
16	1	1	1	0	0	0	0
17	1	1	1	0	0	0	0
18	1	0	0	0	0	0	0
19	1	1	1	0	1	0	0
20	1	0	0	0	0	0	0
21	1	0	0	1	0	0	0
22	1	1	1	0	0	0	0
23	1	0	0	0	0	0	0
24	1	0	0	0	0	0	0
25	1	1	0	0	0	0	0
26	1	0	0	0	0	0	0
32	0	0	0	0	0	1	0
33	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0
38	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0

	I/O INSTRUCTIONS DEFINED						
MEM-LOC.	SETF	CLEARF	SELECT	BUF.ADDR.	STROBE	I-REG. ENABLE	EDB ENABLE EAB
0	0	0	0	0	0	0	
1	0	0	0	0	0	0	
2	0	0	0	0	0	0	
3	0	0	0	0	0	0	
4	0	0	0	0	0	0	
5	0	0	0	0	0	0	
6	0	0	0	0	0	0	
7	0	0	0	0	0	0	
8	0	0	0	0	0	0	
9	0	0	0	0	0	0	
10	0	0	0	0	0	0	
11	0	0	0	0	0	0	
12	0	0	0	0	0	0	
13	0	0	0	0	0	0	
14	0	0	0	0	0	0	
15	0	0	0	0	0	0	
16	0	0	0	0	0	0	
17	0	0	0	0	0	0	
18	0	0	0	0	0	0	
19	0	0	0	0	0	0	
20	0	0	0	0	0	0	
21	0	0	0	0	0	0	
22	0	0	0	0	0	0	
23	0	0	0	0	0	0	
24	0	0	0	0	0	0	
25	0	0	0	0	0	0	
26	0	0	0	0	0	0	
32	0	0	0	0	0	0	
33	0	0	0	0	0	0	
34	0	0	0	0	0	0	
35	0	0	0	0	0	0	
36	0	0	0	0	0	0	
37	0	0	0	0	0	0	
38	0	0	0	0	0	0	
39	0	0	0	0	0	0	
40	0	0	0	0	0	0	
41	0	0	0	0	0	0	
42	0	0	0	0	0	0	
43	0	0	0	0	0	0	
44	0	0	0	0	0	0	

OUTPUT-TRANSFER INSTRUCTIONS DEFINED

MEM-LOC.	TRANSFER	INHIBIT	CLOCK	DATA-READY	TRANSFER-CODE	SOURCE
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0
7	0	0	0	0	0	0
8	0	0	0	0	0	0
9	0	0	0	0	0	0
10	0	0	0	0	0	0
11	0	0	0	0	0	0
12	0	0	0	0	0	0
13	0	0	0	0	0	0
14	0	0	0	0	0	0
15	0	0	0	0	0	0
16	0	0	0	0	0	0
17	0	0	0	0	0	0
18	0	0	0	0	0	0
19	0	0	0	0	0	0
20	0	0	0	0	0	0
21	0	0	0	0	0	0
22	0	0	0	0	0	0
23	0	0	0	0	0	0
24	0	0	0	0	0	0
25	0	0	0	0	0	0
26	0	0	0	0	0	0
32	0	0	0	0	0	0
33	0	0	0	0	0	0
34	1	0	0	0	0	0
35	1	1	1	0	0	0
36	1	1	1	1	0	0
37	1	1	1	3	0	0
38	1	1	1	2	0	0
39	1	1	1	5	0	0
40	1	1	1	4	0	0
41	1	0	0	0	0	0
42	1	0	0	0	0	0
43	1	0	0	0	0	0
44	1	0	0	0	0	0

EISCAT publications

F. du Castel, O. Holt, B. Hultqvist, H. Kohl and M. Tiuri:
A European Incoherent Scatter Facility in the Auroral Zone (EISCAT).
A Feasibility Study ("The Green Report") June 1971. (Out of print).

O. Bratteng and A. Haug:
Model Ionosphere at High Latitude, EISCAT Feasibility Study, Report No. 9.
The Auroral Observatory, Tromsö July 1971. (Out of print).

A European Incoherent Scatter Facility in the Auroral Zone, UHF System and Organization ("The Yellow Report"), June 1974.

EISCAT Annual Report 1976.(Out of print).

P.S. Kildal and T. Hagfors:
Balance between investment in reflector and feed in the VHF cylindrical antenna.
EISCAT Technical Notes No. 77/1, 1977.

T. Hagfors:
Least mean square fitting of data to physical models.
EISCAT Technical Notes No. 78/2, 1978.

T. Hagfors:
The effect of ice on an antenna reflector.
EISCAT Technical Notes No. 78/3, 1978.

T. Hagfors:
The bandwidth of a linear phased array with stepped delay corrections.
EISCAT Technical Notes No. 78/4, 1978.

Data Group meeting in Kiruna, Sweden, 18-20 Jan. 1978
EISCAT Meetings No. 78/1, 1978

EISCAT Annual Report 1977

H-J. Alker:
Measurement principles in the EISCAT system
EISCAT Technical Notes No. 78/5, 1978

EISCAT Data Group meeting in Tromsö, Norway 30-31 May, 1978
EISCAT Meetings No. 78/2, 1978.

P-S. Kildal:
Discrete phase steering by permuting precut phase cables.
EISCAT Technical Notes No. 78/6, 1978

EISCAT UHF antenna acceptance test.
EISCAT Technical Notes No. 78/7, 1978.

P-S. Kildal:
Feeder elements for the EISCAT VHF parabolic cylinder antenna.
EISCAT Technical Notes No. 78/8, 1978.

H-J. Alker:
Program CORRSIM: System for program development and software simulation of EISCAT digital correlator, User's Manual.
EISCAT Technical Notes No. 79/9, 1979.

H-J. Alker:
Instruction manual for EISCAT digital correlator.
EISCAT Technical Notes No. 79/10, 1979

H-J. Alker:
A programmable correlator module for the EISCAT radar system.
EISCAT Technical Notes No. 79/11, 1979.

T. Ho and H-J. Alker:
Scientific programming of the EISCAT digital correlator.
EISCAT Technical Notes No. 79/12, 1979.

S. Westerlund (editor):
Proceedings EISCAT Annual Review Meeting 1969. Part I and II,
Abisko, Sweden, 12-16 March 1979.
EISCAT Meetings No. 79/3, 1979.

J. Murdin:
EISCAT UHF Geometry.
EISCAT Technical Notes No. 79/13, 1979.

T. Hagfors:
Transmitter Polarization Control in the EISCAT UHF System.
EISCAT Technical Notes No. 79/14, 1979.

B. Törustad:
A description of the assembly language for the EISCAT digital correlator.
EISCAT Technical Notes No. 79/15, 1979.

J. Murdin:
Errors in incoherent scatter radar measurements.
EISCAT Technical Notes No. 79/16, 1979.

EISCAT Digital Correlator. TEST MANUAL.
EISCAT Technical Notes No. 79/17, 1979.

G. Lejeune:
A program library for incoherent scatter calculation.
EISCAT Technical Notes No. 79/18, 1979.

K. Folkestad:
Lectures for EISCAT Personnel, Volume I
EISCAT Technical Notes No. 79/19, 1979.

Svein A. Kvalvik:
Correlator Buffer-Memory for the EISCAT Radar system
EISCAT Technical Notes. No. 80/20.

P-S. Kildal:
EISCAT VHF Antenna Tests
EISCAT Technical Notes No. 80/21

J. Armstrong
EISCAT Experiment Preparation Manual
EISCAT Technical Notes No. 80/22

A. Farmer
EISCAT Data Gathering and Dissemination
EISCAT Technical Note 80/23