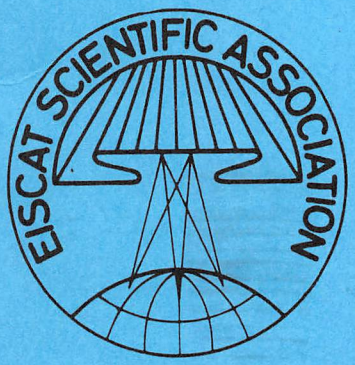
85/44



EISCAT
TECHNICAL
NOTE

THE GEN-SYSTEM FOR THE EISCAT INCOHERENT SCATTER RADARS

by
Tauno Turunen

KIRUNA
Sweden

EISCAT Scientific Association

EISCAT Technical Note 85/44

THE GEN-SYSTEM FOR THE EISCAT INCOHERENT SCATTER RADARS

by

Tauno Turunen

EISCAT Scientific Association
S-981 27 Kiruna, Sweden
September 1985
EISCAT Technical Note 85/44
Printed in Sweden
ISSN 0349-2710

CONTENTS

THE GEN-SYSTEM FOR THE EISCAT INCOHERENT SCATTER RADARS

by

Tauno Turunen
Eiscat Scientific Association
S-981 01 Kiruna 1, Sweden

ABSTRACT

A new set of real time algorithms for the EISCAT radars has been developed, together with an experimental philosophy which fully utilizes the new methods. The design goal has been to develop a method which allows effective use of the EISCAT hardware system, gives an easy way to design general purpose algorithms and makes the best use of the information obtainable from the target with standard incoherent scatter modulations. In some cases the spatial response of the new experiments is also better than in the earlier ones. This new system of designing incoherent scatter experiments using the EISCAT radars is called here the GEN-SYSTEM. It contains the real time algorithms, guidelines to use them effectively and also a program library containing ready experiments called GEN-PROGRAMS. The real time algorithms and the experimental philosophy are described in this report and two examples of the GEN-PROGRAMS are given.

1. INTRODUCTION

EISCAT (European Incoherent SCATter) Scientific Association is equipped with a tristatic UHF-radar installation located in northern Scandinavia, with a transmitter and receiver station at Tromso (Norway) and auxiliary receiving stations at Kiruna (Sweden) and Sodankyla (Finland),together with a monostatic VHF-radar system in Tromso. At the time of writing this (May. 1985) the UHF system is fully working and the VHF system is under the installation. EISCAT is financed by CNRS (France), SA (Finland), MPG (West Germany), NAVF (Norway), NFR (Sweden) and SERC (The United Kingdom). A short description of the EISCAT system is given by Baron (1984).

Since 1981, when the first EISCAT measurements were made, the system has been used extensively for auroral zone ionospheric research and it has been developed in many respects. In November 1984 the UHF radar was working in modes which, in all essential features, fulfil the original design goals. This means that all the 8 parallel receiving channels can be used and one is able to use the full modulation flexibility of the radar transmitter. One can have in a single transmission pattern long pulses, pulse codes and different power profiles in the general case. With some restrictions one is also able to use both phase coded and non coded modulations in the same pattern.

During 1984-1985 the EISCAT VHF-radar installation was proceeding steadily and the operation is expected to beging in 1986 . The receiving system of this radar is essentially ready and because the softaware needed for VHF experiments is in many respects the same or similar to that of fully working UHF-radar, one can in principle use the VHF-radar to its full capacity soon after the testing period is completed.

When EISCAT operations started in 1981, a few measuring schemes were developed and have since then been used extensively both in the EISCAT common programme experiments and in the special programme experiments of the associates. Some developments were done later which allow more extensive use of the radar, but, in general, the basic philosophy has remained the same. Unhappily this philosophy does not necessarily make the most effective use of the system capabilities. That this happened is relatively natural. The methods first used at EISCAT were based on earlier experience from other radars not having all the modulation, multichannel receiving, and processing capabilities of EISCAT. For example the EISCAT real time correlator is a unique device not similar existing at other radars.

The original experiment design philosophy, which was rather heavily dictated by the available correlator programs, was by no means the most effective one. The subroutines were written in the way that the use of the control registers was not optimized and the result memory was not used effectively. As a consequence experiments designed using a few available correlator programs are not flexible enough to take full advantage of the EISCAT hardware system. The experiments are not optimized and expensive investments remain unused.

The situation became much easier when the CORLAN language for the correlator programming became available (Torustad, 1982). This very powerful way of programming the correlator was, however, left almost unused.

This report describes a totally new experimental philosophy, which is called GEN-SYSTEM. In its development no consideration is given to keeping the algorithms "compatible" with the earlier ones. Instead, the only design criterium was to develop a system which allows the most effective use of the radar's inherent capabilities.

The GEN-SYSTEM contains two parts:

1) A set of correlator routines which permit almost any practical experiment using the classical modulations to be processed even in cases when all the 8 receiver channels are in use.

2) A program library which contains a few tested experiments covering the most common experimental needs. This library not only uses the general purpose versions of GEN-SYSTEM correlator programs but in some cases also special code oriented programs which are not developed in "user oriented" way.

To use the general purpose set of GEN-SYSTEM correlator routines the user must write the main correlator program. However, no microprocessor level programming is needed. The user need only to define the algorithms at the subroutine call level using CORLAN. For the users not willing to enter this level, the GEN-PROGRAMS library offers an easy way to have ready experimental algorithms developed for different purposes.

A complete understanding of this report requires knowledge about the EISCAT system and the general diffuse target radar philosophy. The EISCAT Experiment Preparation Manual describes the software system needed in the experiment design (Johansson et al, 1984). The CORLAN manual (Torustad, 1982) contains the syntax needed for the correlator programming. The hardware system in general is described in the extensive series of the EISCAT Technical Notes. Most of the references in this report belong to that series.

2. MULTICHANNEL MULTIMODULATION EXPERIMENTS AS A REAL TIME PROCESSING PROBLEM

An experiment in which one can measure from the lower border of the E-layer to the upper parts of the F-layer requires 2-5 different modulations in order to be effective. Especially the high resolution parts of the modulation pattern must be repeated several times on several channels to get reasonable statistical accuracy. These kind of experiments usually demand the use of all the available 8 receiving channels. A typical transmission pattern consists of 1-4 long pulses, 2-4 pulse codes and 1-5 powerprofiles.

Some parameters are needed to control the necessary real time computations. These control parameters are loaded into the microprocessor registers in the correlator. The minimum amount of input information needed to define the computation is given below for different standard modulations:

1) long pulse
   -the number of lags
   -the number of gates
   -a parameter defining the resolution

2) power profile
   -the number of gates

3) pulse code
   -the number of lags
   -the number of samples
   -lag increment in the sample space

The third parameter in the long pulse control is a new parameter used in the GEN-SYSTEM. It is needed to make the computation of the long pulse data more general than in the older methods. This is later explained in more detail in this report.

Control parameters for the pulse codes are those needed in the modulation free UNIPROG algorithm (Ho et al. 1983, Turunen, 1983).

If either the long pulse part or the pulse code part contains two modulations demanding different sets of control parameters, then altogether 10 registers must be used to control the computation. In addition, one needs also at least the following other control parameters:

1) -constant (usually 1)

2) -address counter

3) -one or more temporary storages

The total number of registers available in the EISCAT correlators to control the input data and computations is 16. For a waveform consisting of a combination of one long pulse, three power profiles and two different pulse codes all the 16 registers are needed. Thus, to use this modulation combination, the following can be immediately concluded:

1) -it is not possible to give independent buffer memory start addresses for all the eight channels. The address counting has to be arranged with a single address counting register.

2) -it is not possible to design a general purpose algorithm which demands the whole code structure as an input parameter set (positions of the pulses in the code, or the system of the code) and can still control multimodulation computations. This does not rule out the possibility that for certain very special cases that kind of programs can be written. Some versions of GEN-PROGRAMS library programs, in fact, are written using specially developed code dependent algorithms to produce a simple output data structure.

In the earlier methods the start addresses have been given for each channel. This is not necessary. No generality is lost if bufffer memory filling always starts from the address 0000, the number of sampled data points at every channel is exactly correct, and there are no gaps between the different data blocks in the buffer memory. In this way only one address counter is needed and it is programmed to point to the first data point belonging to a new data block, when the subroutine returns, and it points to 0000 in the first subroutine call.

The pulse code definition, as done in the most older programs, is not necessary. In the method, which defines the pulse code already in the real time computations, one has to give N-1 registers to define a code (numbers 3,1,5,2 for example define one 5-pulse code uniquely) and one register for the number of samples. This uses the available register stack too rapidly. The UNIPROG approach is a general way to compute the pulsecodes and no other way to do the computation in a general purpose program using 8 channels is known to the author. The UNIPROG approach can define a general pulse code computation problem with the same number of registers as needed for 3-pulse code in the older system (providing that the code definition is done in the most compact way using the "code system" as a starting point).

The general 8 channel problem has also another side, the output part. At present EISCAT has only 2k result memory available in the correlators. This provides an average of only 256 data points per channel, a very small amount except in some very simple power profile applications. It is thus clear that one must have:

1) -a way to compress the data as much as possible, and

2) -a possibility to add the results together from channels having identical modulations, sampling and computation.

Data compression is possible in connection with power profiles and pulse codes. In many cases the spatial resolution of the power profile pulse is much worse than the range separation of the samples. In this situation it is reasonable to add a few neighboring primary gates together without degrading the spatial resolution. Sometimes it is also reasonable to add the neighboring gates together in the pulse coded channels. This can be done in the UNIPROG approach within certain limitations. The method is often applicable to Barker coded multipulses when the high spatial resolution is a direct consequence of the short baud lenght needed to get lag increment short enough but the spatial resolution demand is not as high. This possibility makes also oversampling of the pulse codes possible and thus the integration time can be decreased somewhat. Gate addition can be used to compress the data also in cases when the calibrations (background power or noise injection power estimate) are done by using powerprofile computation.

The addition of neighboring gates together is called "gating" in this report and it is an inherent feature of the GEN-SYSTEM programs. The saving in result memory space is often quite considerable.

The addition of identical channels together is almost a necessity if one wants, for example, to have 4 pulse coded channels, 2 long pulses, and several powerprofiles in the same pattern. In the GEN-SYSTEM this is done by programming the start addresses of the result vectors to define whether the identical data vectors are to be summed at the correlator level or kept separate.

By using both the gating and the channel addition in effective way, the need for a 4k correlator result memory is limited to a rather special applications like certain pulse to pulse correlation experiments and high resolution general purpose experiments using long codes in the low altitude part in the EISCAT UHF system. On the other hand it will be needed in the EISCAT VHF system, in particular for two beam experiments.

The incoherent scatter real time algorithm must be fast enough. In practical cases the multiplication rate is from about 3.5 to about 4.5 MHz in GEN-SYSTEM programs depending on the modulation structure. The slowest algorithms are the UNIPROG and POWERPROFILE, where the mean multiplication rate is only 2.5 MHz. Usually, however, the pulse code part of the computation does not demand a very high number of multiplications. It is the long pulse part which typically dictates the computing time. This algorithm is fast. With commonly occurring control parameters the multiplication rate of the LONGPULSE routine is 4.2-4.7 MHz.The power profile part usually takes only a few percent of the total computing time. The remote station algorithms are very fast in the GEN-SYSTEM programs.

With careful design of the experiments all these factors can be taken into account and it turns out that very practical multimodulation multichannel programs can be written without serious limitations due to the hardware system.

3. THE UNIPROG MATRIX

Most of the algorithms and measurement schemes in this report are described in terms of the UNIPROG matrix. This method was selected because the UNIPROG matrix provides full desription of the information which can be obtained from a target illuminated by a radar pulse. The UNIPROG matrix is described in detail elsewhere (Turunen, 1983 , Turunen and Silen, 1984) and is briefly reviewed here.

The UNIPROG matrix is a very simple form of matrix. It is schematically shown in Fig.3.1. Its elements are of the form

$$E(n,m) = Z(n)*Z^{*}(n+m*LI) \quad \text{,where} \qquad (3.1)$$

Z(n) is an element of the input data vector and the supsscript * denotes complex conjugate. LI is in this report called lag increment in sample space and must be a positive integer, and m is a positive integer.

The UNIPROG matrix is formed from the expectation values of E(n,m) and the expectation values are obtained by taking time averages of the products making up the elements. As seen from equation (3.1) only elements in the upper right triangle are computed. If the input data vector is formed by sampling a radar signal and if the autocorrelation function of the transmitted modulation does not vanish at a delay

$$d = m*LI*t \text{ , where} \qquad (3.2)$$

t is the sampling interval

then E(n,m) is directly proportional to the target autocorrelation function value at the delay d.

For any given m the function E(n,m) forms a single diagonal of the matrix. Every element in this diagonal corresponds to certain range along the radar beam and moving downwards along the diagonal corresponds to moving to farther ranges. Every element has identical spatial resolution which is only a function of transmitted waveform used in the target illumination and the parameter LI. The spatial separation of the elements on the other hand is a function of the sampling interval only and every element has different range.The diagonals are often called "lagprofiles" in this report. A special case is the main diagonal of the matrix, the powerprofile.

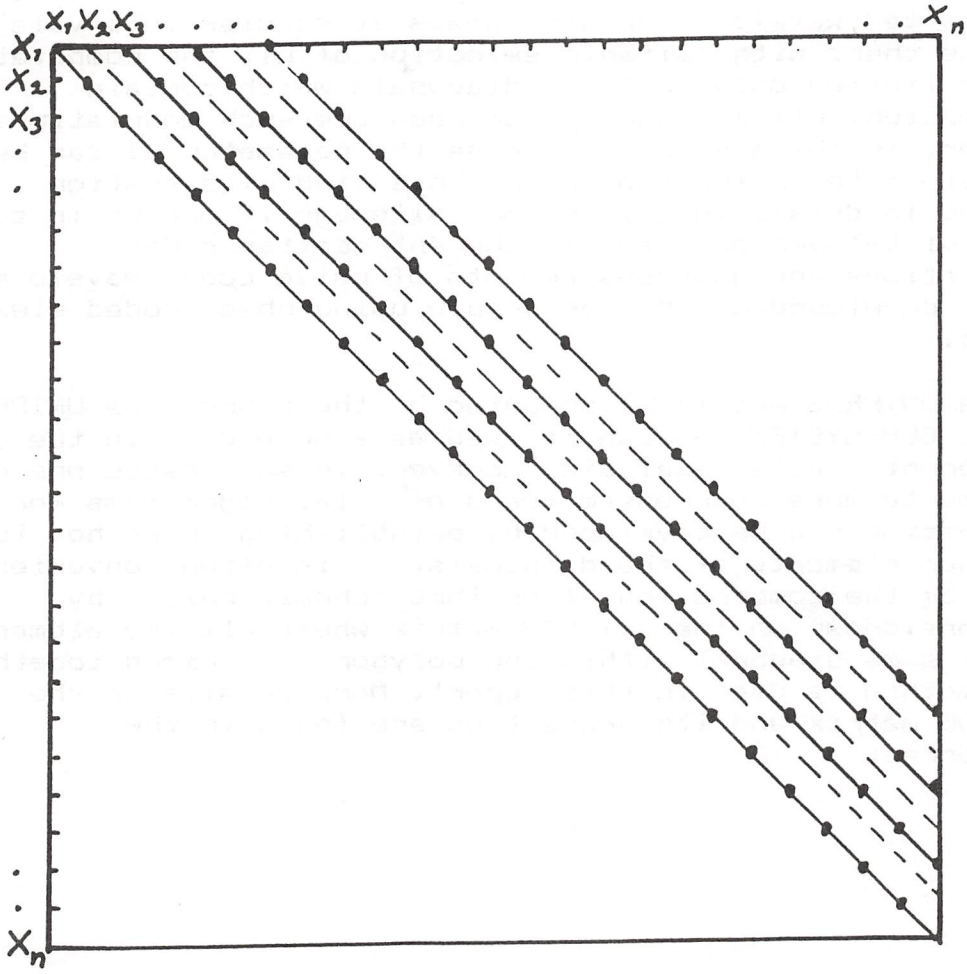


Fig. 3.1. The UNIPROG matrix. The only nonzero elements of this complex valued matrix are in the main diagonal or in some diagonals in the upper triangle. The nonzero elements are expectation values of the delayed complex conjugate products of the form $X(n)*X^{*}(n+m*LI)$, n = 1,...,nmax , m=0,..., mmax. In the example LI=2, nmax=25 and mmax = 3. Nonzero elements and the active diagonals are shown enhanced. In the correlator the first point is the upper left corner after which comes the rest of the main diagonal, then the next active diagonal etc. The last point is the lower right corner point of the last diagonal.

The formulation given in (3.1) fits well to the standard well known modulations used in incoherent scatter and is especially practical when the autocorrelation function of the transmitted waveform contains peaks at regular intervals because then, with suitable selection of LI, the computation can be limited only to those diagonals which contain information. All standard pulsecodes are such modulations. However, in the most general case the parameter LI can be a suitable integer function of m. This kind of situation is not handled in detail in this report although it occurs in some powerful D-layer oriented incoherent scatter radar applications and numerous amounts of pulse coded waveforms can be developed on this principle using phase coded element pulses.

The UNIPROG matrix is computed by the subroutine UNIPROG in the GEN-SYSTEM and can be used as source data in the incoherent scatter analysis. However, in some cases one needs to compute more compressed forms of data. Algorithms for compression can be developed by establishing rules how to sum together elements of the diagonals. It is often convenient to describe the compression algorithms schematically by polygons drawn on the UNIPROG matrix where all the elements on the same diagonal within the polygon are summed together. This method is used in this report. More details on the UNIPROG matrix and its properties are found in the references.

4. THE CORRELATOR PROGRAMS USED IN THE GEN-SYSTEM

Multimodulation multichannel experiments are severely constrained by the earlier correlator programs. Thus the GEN-SYSTEM correlator programs have been written in different way and they have the following properties:

1) -the programming language is CORLAN

2) -an individual main program must be written for every new experiment

3) -a set of computational algorithms are available as library programs. These programs are ready combinations of suitable subroutines and they are called "master programs" in this report.

4) -the starting point of the algorithms is the UNIPROG matrix

5) -the master programs have been designed to cover the most common modulation patterns

The GEN-SYSTEM correlator programs inteded for general purpose applications contain the following algorithms:

1) POWERPROFILE
2) UNIPROG
3) SPECIALUNI (modified UNIPROG)
4) LONGPULSE
5) REMOTE
6) IMPULSE
7) GEN-DUMP
8) DUMMY

Only the first six are called in the main programs written by the user. The subroutine GEN-DUMP is used only to transfer the data to the CAMAC system and the subroutine DUMMY is a do-nothing statement used to shorten the programs. The subroutine REMOTE is not a single subroutine but a combination of three subroutines which do the timing check, remote station signal ACF computation and boxcar weighted ACF computation of the background and noise injection data . It can be called in many different modes. One can directly enter to the last of these subroutines, subroutine IMPULSE. This is used in the receiver system calibration programs belonging to the GEN-PROGRAMS library.

More advanced routines have been written in the individual code oriented way for the special correlator programs used in some experiments in the GEN-PROGRAMS library. These routines are not handled in this report.

The computational algorithms were selected so that the information is used as effectively as possible. The UNIPROG and POWERPROFILE routines permit addition of receiving channels having identical modulations and the addition of neighboring gates. The largest difference with the earlier programs occurs in the single pulse autocorrelation function (ACF) routine. In the GEN-SYSTEM the single pulse ACF estimate is computed such that the absolute volume boundaries are the same at every lag and the spatial resolution is a programmable parameter.

The GEN-SYSTEM master programs are combinations of subroutines to which the user writes the main program. The main program is often only a list of subroutine calls, but it can be a more complicated program , too. Depending on the master program the number of different subroutine calls allowed is 2 to 6. The formats of the calls are described in the comment part of the master programs. The main program can contain one level subroutines (subroutines containing subroutine call lists) but it cannot contain loopcounter based looping.

It is possible to make the program basing on the comments given in the master programs, but if the subroutine call list is not enough to solve the problem, the user is advised to study the CORLAN Manual, the related report GEN-SYSTEM Correlator Programs (Turunen, 1985) and corresponding solutions in the GEN-PROGRAMS library for further details. The correlator master programs in the GEN-SYSTEM are the following (as in May, 1985):

```
GEN-A-L3P2:CLAN
GEN-B-L2U1P2:CLAN
GEN-C-L1U2P3:CLAN
GEN-D-11U2P2:CLAN
GEN-E-L1U1P3:CLAN
GEN-F-L1U1P2:CLAN
GEN-G-L1U1S1P2:CLAN
GEN-H-U3P2:CLAN
GEN-REMOTE:CLAN

    (L = LONGPULSE ROUTINE)
    (U = UNIPROG)
    (P = POWERPROFILE)
    (S = SPECIAL UNIPROG)
```

The last part of the code name reveals the subroutine structure of the program. Thus, for example, L2U1P2 means that the program contains two independent long pulse routines (L2), one UNIPROG routine (U1) and two power profile routines (P2). Remote station oriented master program GEN-REMOTE is designed for beamwidth limited applications and can handle two different long pulse modulations. It contains also suitable routines for the system checking applications.

The subroutine calls are not very complicated. The example given below could be a subroutine call in master programs GEN-B, GEN-E, GEN-F and GEN-G.

```
NEXT
RELOAD LCR1
RELOADVALUE=RSAPB(APBINCR)
RSAPM(RESTARTADD)=RSAPM(RESENTRY4)
CALL UNIPROG
```

The first line defines a new program step in CORLAN. The next two lines load the loop counter 1 register LCR1 to a value given in the APB register named APBINCR (=APB increment=1). This defines the gating parameter and if loaded to a value 1 it causes that two neighboring gates are added together in the correlator. The fourth line defines that the APM register named RESTARTADD is loaded to value given in a register named RESENTRY4. This defines the first result memory address going to be used by the subroutine which is called in the last line. The rest of the computation control for the UNIPROG routine is read from the APB registers loaded by the user. The routine used in this example reads three parameters: the number of the samples to be handled, the maximum lag and the lag increment before it starts processing the data.

The APB and APM stack registers have "symbolic" names in the CORLAN language. These names are given in the INDEX statements which relate the names to the addresses of the control registers. So in the example the RSAPM(RESTARTADD) is the APM stack register , which contains always the RESult memory START ADDress of the called subroutine. The corresponding GEN-E INDEX statement is

```
INDEX RESTARTADD=2 ,
```

which means that the register RSAPM(2) is used as a register RSAPM(RESTARTADD). The user has to know the INDEX table of the master program in order to correctly load the registers in the ELAN file. This table is always given in the beginning of the master program. The INDEX statements do not separate APB and APM registers. The INDEX feature and the general syntax of CORLAN result in programs that are nearly "symbolic". They are easy to read and easy to write once the principles of the CORLAN language are known. The subroutine design demands knowledge about the structure of the correlator and the properties of its microprosessors. Such knowledge of the correlator hardware is not necessary for writing the main programs.

In different master programs the subroutines and their calls differ to a certain extent.

The subroutine call sequence in the GEN-SYSTEM correlator programs is in the same order as the data vectors in the buffer memory. If the buffer memory contains data from

channels 1-4 in that order, and the first channel is a long pulse channel, the next two are pulsecodes and the last one is a power profile, then the subroutine call list of the main program has the following simplified form:

```
NEXT
----
----
CALL LONGPULSE

NEXT
----
----
CALL UNIPROG

NEXT
----
----
CALL UNIPROG

NEXT
----
----
CALL POWERPROFILE
```

The lines not specified in the example contain the gating and channel addition control for the UNIPROG and POWERPROFILE and the number of gates and channel addition control for the LONGPULSE. The rest of the information needed in the computation is read by the routines from the relevant APB and APM registers loaded in the ELAN program.

A little more advanced way to handle the subroutine calls is to make subroutines which contain only subroutine calls. One level of subroutines is allowed in the user written part of the GEN-SYSTEM correlator programs. Many of the GEN-PROGRAMS are written this way and the examples given in the appendixes show the needed structure in detail. This method makes possible to control computation of more than 20 datablocks with very limited number of correlator main program steps. Thus, for example, one can always program in a way that no "garbage" gates appear in the long pulse data and that calibrations in pulse code channels are done in an effective and memory saving way.

The decicion to have the main program be a part of the experiment design was needed to give the necessary flexibility to the system. The number and order of the subroutine calls is the same as the number and order of the data blocks in the buffer memory (note also the looping possibility). One data block is often, but not necessarily, all data of one channel. If free positions for the subroutine calls exist in the correlator, then it is always practical to handle the data and calibrations of a single channel as separate data blocks.

The above description gives the basic ideas of the GEN-SYSTEM correlator programs. The CORLAN language manual describes the syntax in detail. The examples in this report give further information about the use of the programs. After knowing the details (there are not too many of them) the correlator part of the experiment design is in fact an easy part of the programming. It is a description of the buffer memory layout in the form of subroutine calls. Finally the program is compiled by the CORLAN compiler, which only asks for the input and output files.

The total number of the subroutine calls depends on the amount of free positions in the correlator program memory and this depends on the subroutine structure. In the present versions of the programs, the number of free program steps is between 12 and 29. The program GEN-C is the most "crowded" one. Sometimes the number of available subroutine calls is a small limitation but an 8 channel multimodulation experiment seems to be always possible to control, at least by looping the subroutine calls. Further developments of the algorithms may give more space for the subroutine calls; but even in the present forms they are practical.

Correlator programs which have not been designed to have general purpose features and no importance is given to the easy use of the program can be commanded to do more computations than the standard general purpose GEN-SYSTEM correlator programs. In a few cases this kind of special programs have so much benefits that they are worth writing. One application is in experiments having 300 meters resolution. If one wants to have reasonable range coverage and high number of lags the UNIPROG matrix format enters to difficulties because too large number of matrix elements outside the decodable altitude range has to be computed. Similar difficulty arises if the pulse-to-pulse correlation experiment uses Barker coded pulses in complicated combination codes. Sometimes the modulation structure is such that a possibility exists for real time decoding of all the computed UNIPROG matrixes and it is tempting to make use of it. For those cases special programs have been developed in the way that they can be used exactly with the given measuring algorithm for optimized output data structure. They are not intended for other applications and are not handled in this report to any details. Some experiments in the GEN-PROGRAMS library make use of such special correlator programs.

5. THE GEN-SYSTEM POWERPROFILE

The power profile routine POWERPROFILE is essentially the same as in all EISCAT correlator programs but it is more general in the sense that in the subroutine call one can load the loop counter register 1 to a value which defines how many neighboring gates are to be added together. This is called "gating" in this report and it is defined in a similar way also in the UNIPROG routine. The value specified for the gating is the number of gates to be added together minus one. Thus a value of 0 as the gating parameter produces a "normal" powerprofile.

Note that in some master programs there are no APB stack registers reserved for the gating control. Thus in the gating command sequence:

```
NEXT
----
RELOAD LCR1          %gating
RELOADVALUE=.....    %command
CALL POWERPROFILE
```

the RELOADVALUE must be formed by arithmetical operations from the available register values in the APB stack. The values 0,1 and 2 are simple and always available because they can be formed in the following way:

```
RELOADVALUE=0
RELOADVALUE=RSAPB(APBINCR)
RELOADVALUE=RSAPB(APBINCR)+RSAPB(APBINCR) ,
```

where the content of RSAPB(APBINCR) =1 (APB increment).

Other values have to be formed either as a sum or difference of two (and only two) available APB register values or set to the available register values themselves. This sounds like a big limitation but since in general the APB stack registers contain more than 10 small numbers , in practice, a large number of values can be generated. Sometimes it is necessary to select some other parameters in the program in such a way that a suitable gating control value is also obtainable. There was no other way to arrange the gating control because free APB stack registers were not always available.

The power profile routine computes the main diagonal of the UNIPROG matrix and adds, under the control of the gating parameter, a few neighboring points together. This is shown schematically in the Fig. 5.1. The power profile routine counts the number of samples. Thus the number of gates is in the output data vector:

NOGATES=NOSA/(GATING+1) ,where (5.1)

NOSA= the number of samples in the input data
NOGATES=the number of gates going to be computed
GATING= the gating parameter as defined above

Only the parameter NOSAMPLES is given in an APB register and in the GEN-SYSTEM index tables the name PPNOSAMPLES is assigned to the register used for this purpose.

The power profile routines in the GEN-SYSTEM programs are not essentially different from the earlier programs. Some points are, however, worth mentioning. Because the gating parameter is given in the subroutine call and does not usually appear in the correlator APB register stack loaded in the ELAN file the information about the gating has to be put onto the tape in another way. One way is to describe it in the description file of the experiment. Another way is to write a symbolic file of the main correlator program to the data tape. There are also other reasons for doing that and thus this method is recommended.

The gating may differ in the signal part and calibration parts of the power profiles. In that case the mean values of the background noise and noise injection have to be scaled before applying them in the subtraction of the background noise and determination of the calibrated signal strenght. If the signal part is computed with gating factor GATING1 and the background noise with the gating factor GATING2, then before subtracting the noise mean value, it must be multiplied by:

M=(GATING1+1)/(GATING2+1) (5.2)

The powerprofile routine is very often used to compute the background noise and noise injection parts of the pulse coded channels in GEN-PROGRAMS. Almost invariably the gating for the calibration is different from the gating for the pulse coded data (computed by the UNIPROG). If the POWERPROFILE gating factor is GATINGP and the UNIPROG gating factor is GATINGU then the background noise and noise injection estimates must be multiplied by

M=(GATINGU+1)/(GATINGP+1). (5.3)

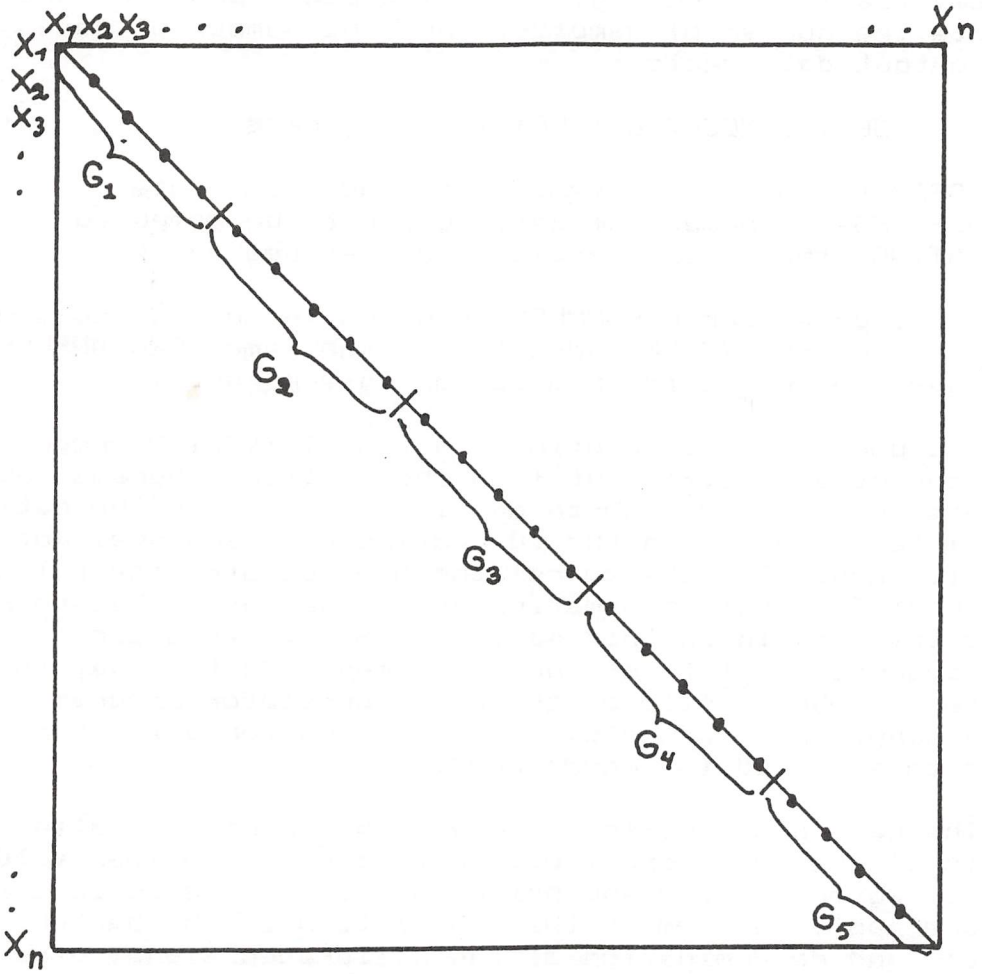


Fig. 5.1. Gated powerprofile computation. Only the main diagonal of the UNIPROG matrix is computed and the (GATING+1) neighboring elements added together in the way shown. In the example the gating parameter is 4 and 5 result memory points are computed.

Any individual sample in the main diagonal of the UNIPROG matrix contains signal power coming from a volume having lenght D along the beam:

$$D=(T+STEP)*c/2, \qquad (5.4)$$

where T is the lenght of the transmitted pulse and STEP is the receiver system step response time to the final level (in practice almost the same value as the filter group delay).

The parameter D, when expressed in this way, gives with good accuracy the separation of the upper and lower boundaries of the volume. The spatial weighting factor within the volume is obtained by convolving the transmitted pulse with the receiver impulse response function and squaring the result. In many practical cases, when T is considerably longer than STEP, the spatial weighting is nearly boxcar shaped. When applying gating, several volumes of this shape are summed together with the centers of the volumes separate by an amount

$$d=t*c/2 , \qquad (5.5)$$

where t is the sampling interval.

If T = t*(GATING+1), the resulting volume has almost a triangular spatial weighting function. In all the other cases the spatial weighting function is similar to "triangle without the top". The lenght of the volume in the case of gating is

$$DABS=(T+STEP+GATING*t)*c/2. \qquad (5.6)$$

The resulting volume center points have a separation which in this report is called gate separation

$$SEP=(GATING+1)*t*c/2 \qquad (5.7)$$

The middle point of the first gate is at a range RANGE1

$$RANGE1= DEL*c/2-(T+STEP-GATING*t)*c/4, \qquad (5.8)$$

where DEL is the delay between the transmission of the leading edge of the illumination and start sampling command.

In many cases it is convenient to use gating because it compresses the data and produces gates whose separation is not too small compared with the gate resolution. When triangular spatial weighting is programmed, the volume size at the 75% contribution level is DABS/2 ; in two neighboring gates 25% of the contribution arises from the common part of the volumes. The gate separation is the same as the gate resolution at about the 75% level. This parameter selection (GATING+1=T/t) is often a quite reasonable compromise.

The pulse lenght, receiver step response function, sampling frequency and the gating together dictate the spatial filtering properties of the powerprofile algorithm.

The power profile routine is fast enough in GEN-SYSTEM programs. The multiplication rate is 2.5 MHz undepending on the gating parameter.

The computation time COMP is obtained from the formula

COMP=(2*NOSAMPLES+7)*0.2 us (5.9)

The speed of the power profile routine is not critical. The total number of UNIPROG matrix elements computed by the power profile routine is usually very small compared with the number of elements computed by the other subroutines. In practical experiments only a few percent of the total computing time is used by the power profile routine.

6. THE GEN-SYSTEM UNIPROG AND PULSE CODED MODULATIONS

The UNIPROG routine is a development of the original "Universal Program" (Ho et al, 1983). This algorithm is used in the GEN-SYSTEM mainly to compute lag products for the pulse coded parts of the modulation pattern. One has to remember, however, that the UNIPROG routine is a general algorithm which works with any modulation although with some it is not practical because of high amount of the result memory space needed. In the case of standard pulse codes, however, it produces an output vector which is the most compact one possible that contains all the available information. The UNIPROG routine computes the main diagonal of the UNIPROG matrix and every LIth diagonal in the upper triangle up to a limit. The limit specified by the total number of diagonals is UNILAGMAX+1, where the parameter UNILAGMAX is given in the APB stack. LI is called the lag increment and the name UNILAGINCR is assigned to the corrsponding APB register in the GEN-SYSTEM index table.

The gating parameter for the UNIPROG routine is defined in the same way as for the power profiles. The gating parameter cannot, however, be set to any arbitrary value but it must be such that the resulting UNIPROG matrix can be decoded.

Before handling the gating one has to know how the pulse code is decoded from the output vector. Fig. 6.1 shows schematically the computations done by the UNIPROG routine. The gating is done so that the first point at every diagonal is also a first point of a new gate. Thus the algorithm requires that the number of points in every diagonal is the gating parameter+1 multiplied by an integer. The main diagonal in the UNIPROG matrix before gating has the same number of points as exist in the input data vector. Thus the number of points in the input data vector must necessarily be an integer times the gating parameter +1. Because the number of points in the other diagonals is the number of points in the main diagonal minus i*LI, where i=1,...,UNILAGMAX the necessary conditions for gating are:

1) the number of points in the input data vector must be an integer times the gating parameter+1, and

2) the LI divided by the gating parameter+1 must be an integer.

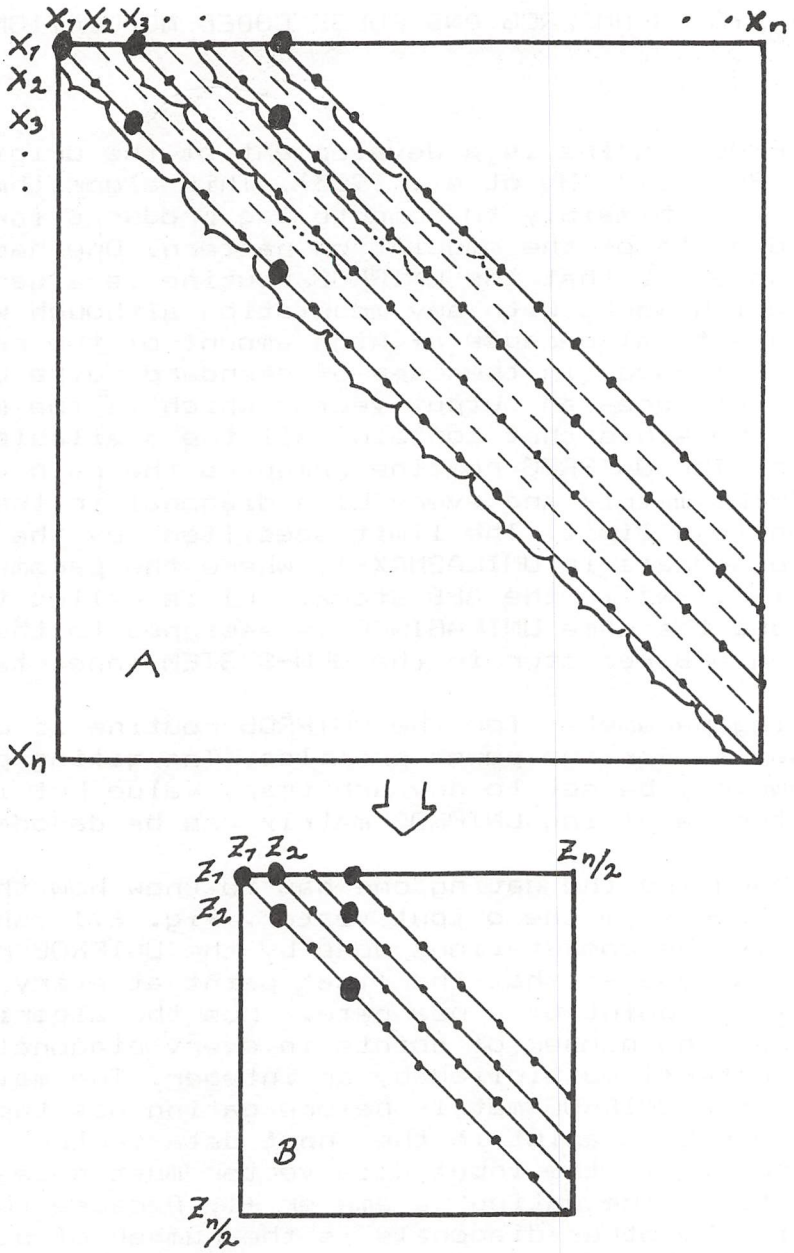


Fig. 6.1. Gated UNIPROG computation. The parameters in the example are nmax=20, LI=2, mmax=3, and GATING = 1. The figure shows how the gating scales the original UNIPROG matrix to a smaller matrix, which still has the properties of the UNIPROG matrix. The enhanced points are the elements related to the decoding of a 3-pulse code transmitted in system 1:2 and sampled for LI=2. In the smaller UNIPROG matrix , which appears in the correlator, LI = 1 because of gating. Two neighboring gates are thus added together in the example.

After gating the resulting compressed UNIPROG matrix has

1) a main diagonal containing the original number of points divided by the gating parameter+1, and

2) the same number of computed diagonals as before, and

3) a lagincrement equal to the original lagincrement divided by the gating parameter+1

Figure 6.1b shows schematically how the gating scales the original UNIPROG matrix to a smaller UNIPROG matrix having different parameters.

The next aspect to consider is the pulse code. Before discussing the actual decoding, we must have a system for describing the pulse code modulation pattern. We choose a system in which the pulse code is described by a set of numbers specifying the relative delay between neighboring pulses in the code in the order in which they appear. The unit delay used is the shortest delay between any two pulses (DELMIN) in the code divided by the product of sampling interval (t) times lagincrement LI. Examples given below illustrate this system of describing the code:

```
__-_-___-_                      3-pulse code, system 1:2
__-_-____-___-_                 4-pulse code, system 1:3:2
__-___-____-_-_                 4-pulse code, system 2:3:1
__-____-_-_________-___-_       5-pulse code, system 3:1:5:2
```

a) DELMIN/(t*LI)=1 , _-_ is a simple pulse.

```
_BB_B                           3-pulse code, system 1:2
_BB__B_B_                       4-pulse code, system 1:3:2
_B__BB____B_B_                  5-pulse code, system 3:1:5:2
```

b) DELMIN/(t*LI)=1, _B_ is a phase coded pulse with M bit code. Codes are transmitted so that in the lag 1 group no gap exists between the pulses. The transmission gaps between the other pulses are integer times M*t

```
_B.B._.B.                       3-pulse code, system 1:2
_.B.B._._.B._.B.                4-pulse code, system 1:3:2
```

c) DELMIN(t*LI)=1, _.B._. is a phase coded pulse with M bits followed by a gap of one bit lenght. All the other gaps are integer times M+1 bits

Fig. 6.2. Examples of the formation of the code description system. The pulse patterns show the amplitude modulation needed to form pulse codes. The leftmost pulse is assumed to be transmitted first. For details see the text.

The delays obtainable from the code are all the individual elements of the system and all the possible sums of two or more neighboring elements of the system (called the system sum and denoted by SSU). Thus the 4-pulse code having the system 2:3:1 has the following lags: 2, 3, 1, 2+3, 3+1, 2+3+1 i.e. all the lags from 1 to 6. When these values are multiplied by the unit delay of the lag 1 pair, then one obtains the actual time delays of the ACF estimate.
If the smallest element in the system of the code is 1 as in all the simple examples given in Fig. 6.2., then the lag 1 delay is t*LI. It is, however, not necessary that the smallest element is 1. Pulse codes can be built, using phase coded pulses, also so that the smallest element of the system is an integer obtained by dividing the number of bits of the elementary pulse phase code by an integer. In order to obtain a suitable element pulse one often has to add one or more "zero bits" into the elementary pulse (13 cannot be divided by any number but 13+1 is already dividable by 2 and 7). These kind of pulse codes could be called "compressed" pulse codes. They are about as free from the spatial ambiquities as the "normal" pulse codes using phase coded pulses.

EXAMPLE: A 3-pulse code has a 13 bit Barker coded pulse plus a gap of one baud lenght as an elementary pulse. A code having system 2:3 is formed by a sequence:pulse+pulse+ 7 bit gap +pulse. This code has lagincrement 7 and in UNIPROG computation the lag 1 is a 7 bit delay (note that lag 1 is a "missing lag")

In general case the first obtainable delay from the pulse code is t*LI*SMIN, where SMIN is the smallest number in the system of the code.

All "standard" pulse codes ,with or without phase coding, have 1 as a smallest element. "Compressed" pulse codes, however, have sometimes very useful properties.

Before analysis one has to decode the UNIPROG matrix, i.e. to define the correlator addresses of the matrix elements which form an autocorrelation function estimate for a given altitude.

The decoding of the UNIPROG matrix is done using the system described above. The system sums, SSU , give the obtainable lags. But the different lags which correspond to the ACF from a fixed altitude are not ordered in a straightforward way in the UNIPROG matrix. This is illustrated in Figure 6.3. for the 4-pulse code transmitted in the system 2:3:1 and sampled for lagincrement 2. Fig. 6.3. is a range-time diagram showing the timing of 4 transmitted pulses (assumed as delta functions) and the data samples Z1.

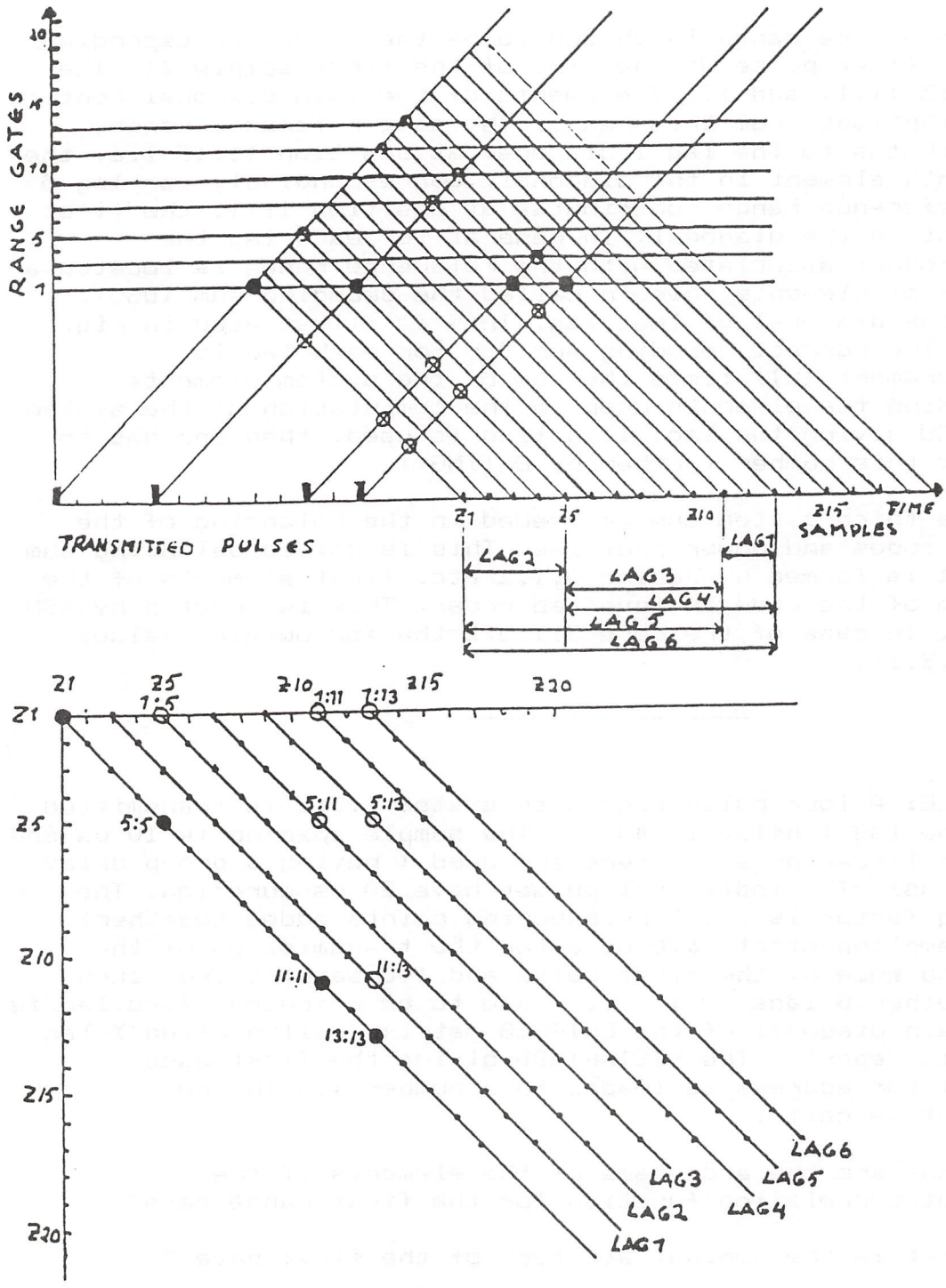


Fig. 6.3. The range-time diagram (upper part) for a 4-pulse code showing the formation of the ACF of the first gate. The cluttering altitudes have been marked with open circles. The lower part shows the elements of the UNIPROG matrix containing contribution from the first gate. The elements needed in the ACF have been marked with open circles. Note the similarities in the upper and lower parts of the figure after rotation of 45 degrees. For details see the text.

The reference range is chosen to be the range corresponding to the first pulse at the time of the first sample Z1. The 1:1,5:5,11:11 and 13:13 elements of the main diagonal contain contributions from this range. The same reference range contributes to the lag 1 diagonal at position 11:13 i.e. the eleventh element in the diagonal. Correspondingly for lag 5 the reference range contributes at position 1:11, the first element in the diagonal. In general for each lag the lag-product associated with the reference range is located a number of elements, herein called the decoding sum (DSU), down the diagonal of that lag. This is illustrated in Fig. 6.3b. The correct decoding sum DSU for each lag is lagincrement (LI) times the sum of the system elements prededing the elements used in the computation of the system sum SSU giving the lag. If gating is used, then one has to divide this number further by GATING+1.

The third system sum is needed in the balancing of the pulse codes and power profiles. This is called balancing sum and it is formed by adding 0,1,2 etc. first elements of the system of the code in inverted order. This is denoted by XSU later. In case of the code 3:1:5:2 the XSU obtains values 0,2,7,8,11.

EXAMPLE: A four pulse code with system 1:3:2 is transmitted and the lag 1 delay is 40 us. The sample spacing is 10 us and 25 kHz linear phase filters are used (having a group delay of 21 us). The individual pulses have 20 us duration. The gating factor is 1 (2 neighboring points added together). The sampling starts 620 us after the transmission of the leading edge of the first pulse and 100 samples are taken. Altogether 8 lags (0,1,...,7) are to be computed. Zero lag is the main diagonal of the UNIPROG matrix (called often X-lag in this report). The RESTARTADD giving the first used correlator address is loaded to a number 900 in the subroutine call .

-What are the addresses of the elements of the autocorrelation function for the first range gate?

-What is the nominal altitude of the first gate ?

-How many gates can be decoded?

-How many result memory points are needed for the data?

-How long time is needed for the computation in the EISCAT correlators?

SOLUTION: The lag 1 delay is 40 us and the sampling frequency is 10 us. Thus the UNILAGINCR in the GEN-SYSTEM UNIPROG is 40/10=4. The number of samples is 100 and thus the parameter UNINOSAMPLES is 100. The gating factor is 1 and thus in the subroutine call a sequence RELOAD LCR1 RELOADVALUE=RSAPB(APBINCR) must be given. Gating factor +1 = 2. Thus the final lag increment is 4/2=2 and the number of points in the main diagonal is 100/2=50. The number of the points in the other diagonals is 50-i*LI, where LI is the final lag increment. Thus the number of the points in all the diagonals and the addresses of the first points of the diagonals and the first gate at each lag are the following:

LAG	POINTS	SSU	DSU*LI	FIRST POINT	FIRST GATE
X	50	not defined		900	not defined
1	48	1	0	950	950+0=950
2	46	2	(1+3)*2	998	998+8=1006
3	44	3	1*2	1044	1044+2=1046
4	42	1+3	0	1088	1088+0=1088
5	40	3+2	1*2	1130	1130+2=1132
6	38	1+3+2	0	1170	1170+0=1170
7	36	not defined		1208	not defined

the address of the last point is 1208+36-1=1243

The total space taken by the data is 1243-900+1=344 result memory locations. The GEN-SYSTEM UNIPROG takes gating + 1 times = two computing cycles (one cycle is 200 ns) for one result memory point. The computing time taken by the correlator is approximately 2*344*2*0.2=275.2 us. In practice, when taking into account the overhead of the computing, a value 280 us could be used when estimating the time needed for the example computations. The total number of gates is always the number of points in the longest lag diagonal, the one formed by adding all the system elements together. In the example it is the lag 6 and thus the number of gates is 38. The addressing of the first gate is shown in the last column. The data of the ith gate is formed by adding i=0,...,37 to the value given, and i is the index of the gate (starting from 0). The lag 7 is not provided by the pulse code waveform. Thus there is no correlating signal and the mean value of all the lag 7 points can be used to estimate the receiver offset voltage. The lag 0 cannot be handled in the same way because it contains contribution from several altitudes. It is called here the lag X and it is important for balancing the pulse coded channels against the power profiles. It is also possible to make deconvolution and obtain a true zero lag estimate from the X-profile. This special application is not handled in this report.

Where is the first volume? The sampling started 620 us after the leading edge of the first pulse in the code and the group delay of the used filter was 21 us. This is also the time needed for the step response function to obtain the final value with great accuracy in the used filter conficuration. The middle point of the first primary volume is approximately at (620-21)*c/2, where c is the velocity of the light. This is at the range 89.85 km. The gating factor

was 1 and thus two neighboring gates where added. The two gates added together to form the final first gate have the middle points at ranges 89.85 and 91.35 km. The resulting middle point is 90.6 km. This can be considered to be the final middle point of the first volume after the gating. Because before the gating the range separation of the volumes is the same as the sampling frequency*c/2=1.5 km, then after the gating the gate separation is 3 km in this example. The total range is (38-1)*3=111 km and thus the total range coverage is 90.6-201.6 km. Note that the altitude of the first gate is not absolutely correct. Because the settling time of the step response function is slighltly longer than the pulse lenght, there is a tiny asymmetry in the spatial response. The calculation given above does not take into account the distance factor change within the volume either. This causes a bigger error than the step response function behaviour. The resolution of the gate is from the absolute lower boundary to the absolute higher boundary (20+21)*c/2=6.15 km before the gating. The spatial weighting factor is approximately "Gaussian squared" with 25% limits separated by 3 km. Two of these, separated by 1.5 km , are added together to form the final gate. The absolute boundaries are then 7.65 km apart and this is not infinitely small compared with the range of the nominal middle point, which was 90.6 km. Thus a nonzero distance factor correction exists. The main contribution arises , however, from a much smaller volume of the order of 3 km depending on how the user wants to define the volume. The tiny correction factors mentioned here are, in most practical cases, only of academic interest.

It is straightforward to develope formulae for the decoding of the UNIPROG matrix for any special application. It is, however, not very trivial to give general formula, because the way how one wants to calibrate the experiment may differ from case to case. The used method may also be dictated by the number of free subroutine calls available in the correlator program. If one has to combine two data vectors into the same UNIPROG matrix, then the decoding greatly differs from the one given in the example. In pulse coded data it is only necessary to calibrate the offset, gain, and system noise. The impulse response information has to be measured externally. The calibration data may be handled with the same subroutine call or it may be handled by the power profile routine, because only power profile computations are needed in the gain and system noise estimation. At long lags the expectation value of the target ACF gives a smaller contribution than the receiver offset voltage (and all low frequency noise components) in spite of the improvements to the receiver system. The easiest way to obtain the offset estimate is to compute a lag which does not contain contribution from the target.

Some general formulations, sufficient for most applications, are given in the next two chapters.

7. DECODING OF THE PULSE CODES FROM THE UNIPROG DATA AND CHANNEL BALANCING

If the GEN-SYSTEM UNIPROG routine is used in "normal"way, then the decoding can be expressed in terms of simple formulae. It is assumed that the UNIPROG is used with the philosophy described in the previous chapters and this is always so in the GEN-PROGRAMS library. The formulae are, however, also given in little more advanced versions which are not usually needed in connection with GEN-PROGRAMS but which make the analysis of the original UNI-PROG based data (like EFOR and ESLA data collected by many users) possible in the similar way.

In the decoding formulae the following symbols are used:

RES = the result memory start address for the data computed by the UNIPROG routine. It is either the RESENTRY given in the APM stack and used in connection with the CALL UNIPROG or the start address computed automatically by the GEN-SYSTEM correlator program.

SAM = The number of input data points handled by the uniprog section. It is given in the register UNINOSAMPLES.

LI = The lag increment used in the computation and given in the register UNILAGINCR. Note that this is the computation lagincrement and not the final output data lag increment.

MLA = The maximum lag given in the register UNILAGMAX. (the number of UNIPROG matrix diagonals to be computed + 1).

GAT = The gating parameter (the number of gates to be added-1) given in the subroutine call (RELOAD LCR1)

SSU = The system sum giving the index of the lag being computed. See previous chapter for details.

DSU = The decoding sum used to compute the address offset along the diagonal to get the address of the first gate at the lag specified by the SSU. See previous chapter for details.

XSU = The balancing sum needed for the pulse code and power profile channel balancing. See previous chapter for details.

MSU = The largest SSU which exists in the pulse code and is not greater than MLA. It is needed to compute the number of gates.

i = lag index 1, 2, ..., MLA. This is the same as the SSU if the SSU exists for the code in use for a given i. If the SSU does not exist, then the lag does not exist in the code and that lag can be used for calibration purposes.

g = gate index 1,2,...,gmax. The lowest gate is 1, the highest is gmax.

ADDR(g,i) = The result memory address of the target autocorrelation function for the gate g and lag i.

ACF(g,i) = The value of the data point at address ADDR(g,i). ACF(g,i) is the raw autocorrelation estimate.

DATA(k) = The value of the data at the address k.

XBACKGR = The mean value of the background points in the pulse coded channel.

XNOISE = The mean value of the noise injection points in the pulse coded channel.

PBACKGR = The mean value of the background points in the pulse coded channel.

PNOISE = The mean value of the noise injectin points in the power profile channel.

Note that the last four parameters have to be corrected for the possible difference in the gating factor between the scatter data and calibrations.

The parameters listed above are sufficient to decode the UNIPROG section data if only the data block containing the scattered signal is computed by the UNIPROG and the same UNIPROG matrix contains only one data block.

The first step is to check that SSU exists for the i under consideration. When the elements needed in the SSU are found, then the elements used in the DSU are known and DSU(i) can be computed. One must check the value of gmax.

$$gmax = (SAM-MSU*LI)/(GAT+1) \qquad (7.1)$$

$$ADDR(g,i)=RES+(i*SAM-((i-1)*i/2+DSU(i))*LI/(GAT+1)+g-1 \qquad (7.2)$$

where
i=1,..,LMA and SSU=i exists
g=1,..,gmax

If SSU does not exist, the lag is a "missing lag". Then the data points are proportional to the offset bias voltages and very low frequency (hum) internal interference. In the simplest case these can be assumed constant during the sampling (not always a valid assumption) and the mean value of the points can be taken. The complex mean value is called OFFSET and the corresponding diagonal in the UNIPROG matrix is called the offset profile. It is often wise to leave some of the first points in the offset profile unused, because they may contain receiver recovery and ground clutter effects. The number of unused points is called DEL and its value depends on the measuring conditions, hardware status and the modulation pattern. DEL has to be spesified by the user in the analysing phase of the data. Then the normalized OFFSET can be computed by using the formula

$$OFFSET= \sum_{k=ADDR(1,i)+DEL}^{ADDR(1,i)+(SAM-i*LI)/(GAT+1)} DATA(k)/((SAM-i*LI)/(GAT+1)-DEL) \qquad (7.3)$$

and SSU=i does not exist

If the data dump contains several offset profiles, a mean value of all of them can be used as the offset. The value for the offset corrected ACF(g,i) = CACF(g,i) is obtained from

$$CACF(g,i) = ACF(g,i)-OFFSET \quad , SSU=i \text{ exists} \qquad (7.4)$$

Two other elements are needed to obtain the target autocorrelation function, namely the zero lag estimate and the absolute power scale. These are obtained from the power profiles, which have to be measured using the same spatial response and the same receiving parameters and in the way that the power profile gates coincide with the pulse code gates.

The approach described here is based on the fact that if the signal to noise ratio is good enough for analysis, then it is also good enough for balancing using direct comparisons of the target signals. The main diagonal of the UNIPROG matrix (called the X-profile in this report) is a linear combination of as many power profiles as there are pulses in the pulse code. These power profiles have relative spatial

offsets which depend only on the used pulse code. To find them we define balancing sum XSU(n) as the sum of n last elements of the code and n = 0,1,...,nmax, where nmax is the total number of the elements in the system of the code. Thus, for example, the XSU(n) values for the code 1:3:2 are 0,2,5 and 6. The XSU(n) are the relative offsets to make when summing power profiles to obtain the profile to be compared (balanced) against the x-lag profile.

To get the absolute shift between the reconstructed power profile and the x-lag profile we must calculate the difference in range between the first gates of the two profiles. Power profiles must be programmed to start at lower range. The first decodable pulse code gate is the one to which the datapoints CACF(1,i) belong to. It is at the range RPC(g), g=1

$$RPC(1) = ((PCDEL-(T+STEP)/2)*c/2 +((GAT+1)/2)*t*c/2 \quad (7.5)$$

where

PCDEL = start sampling delay of the pulse coded channel
T = the pulse lenght used in the code
STEP = the step response time of the receiver system (in practice the same as the group delay)
t = sampling interval

In similar way one can define the first gate of the power profile channel RPP(1) by replacing PCDEL by the corresponding start sampling delay of the power profile channel (PPDEL) because the other parameters must be the same. The offset, in units of gates, between the power profile and pulse code is given by

$$DAL = (PCDEL-PPDEL)/((GAT+1)*t) \quad (7.6)$$

DAL must be an integer and it must also be a positive number. Sometimes the lowest gates are contaminated by the receiver recovery effects and ground clutter. Thus the user must provide a positive integer parameter to be the number of the first power profile gates not to be used in the balancing. This data quality dependent user input is here denoted by MAR.

The lowest altitude contributing to the X-profile is MSU*LI/(GAT+1) gates below the first pulse code gate and this number is usually greater than DAL. The first X-profile point which can be used in the balancing has the result memory address

$$XSTART=RES+MSU*LI/(GAT+1)-DAL+MAR \quad (7.7)$$

The address XSTART must be equal to or greater than RES. This can always be arranged by selecting a suitable MAR undepending how the sampling of the pulse codes and power profiles have been done.

The address of the first power profile point which can be used in the balancing is

$$PSTART=PRES+MAR \tag{7.8}$$

where PRES is the address of the first powerprofile point.

The balancing is based on the target contribution and thus the background must be subtracted both from the X-profile and from the powerprofile.

The target contribution in the X-profile can now be simulated from the address XSTART to some address XSTART+kmax but we need only the sum of the power of the target contribution in this profile. This sum is denoted here by SXSIMUL and obtained from

$$SXSIMUL = \sum_{n=0}^{nmax} \sum_{k=PSTART}^{PSTART+kmax} (DATA((k+XSU(n))-PBACKGR) \tag{7.9}$$

The expectation value of the SXSIMUL deviates from the corresponding power sum for the measured target contribution in the X-profile only by a constant. The sum of the X-profile power SXPROF is obtained from

$$SXPROF = \sum_{k=XSTART}^{XSTART+kmax} (DATA(k)-MXBACKGR) \tag{7.10}$$

The number of X-profile points used in the balancing is kmax. This parameter can either be given as a user input so that the balancing is limited only to the strong signals from the E-layer peak surrounding or it may be computed to be the maximum possible. The parameter kmax is either depending on the number of the available powerprofile points or on the number of the available X-profile points. Usually the first case is true. If the kmax is powerprofile limited, then the value is

$$Pkmax = PPNOSAMPLES/(GAT+1)-MAR-XSU(nmax)*LI/(GAT+1) \tag{7.11}$$

If the kmax is X-profile limited, then its value is

$$Xkmax = SAM/(GAT+1)-(XSTART-RES) \qquad (7.12)$$

The parameter kmax must necessarily be equal to or smaller than smaller of the values of Pkmax and Xkmax.

An estimate for the constant (GAINFACTOR) can be obtained from the power sums of these two profiles.

$$GAINFACTOR = SXSIMUL/SXPROF \qquad (7.13)$$

In these computations the mean values of the background estimates have to be computed for the same effective gating factor. The same number of channels added together in the UNIPROG and POWERPROFILE parts is not assumed.

The pulse coded channel can now be scaled to the same scale as the powerprofile by multiplying the target contribution part by the constant GAINFACT and the zero lag is directly the powerprofile value at the same altitude as the pulse coded gate. The gain calibration is obtained using the calibration values (background, noise injection) of the power profile data. The final autocorrelation function values FACF(g,i) are then

(7.14)

$$FACF(g,i) = DATA(PSTART+DAL+g-1)-PBACKGR \quad ;i=0$$

$$FACF(g,i) = GAINFACTOR*CACF(g,i) \quad ;i=1,\ldots,LMA \text{ and SSU=i exists}; g=1,\ldots,gmax$$

Note that when the balancing is done in this way, the noise injection value obtained from the power profile data gives the necessary gain parameter for the decoded ACFs. The noise injection of the pulse coded channel is not needed at all. It is, however, always measured in the GEN-PROGRAMS library programs to give a cross checking possibility.

In GEN-PROGRAMS the pulse code calibration is almost always done by using powerprofile routine (see examples in the appendixes) and one UNIPROG routine call handles only one data block. In this situation the formulation given above is sufficient for the decoding computations. The situation becomes much more complicated if two or more data blocks are computed into a single UNIPROG matrix. Because this kind of compromises in the computation are no more needed when using the GEN-SYSTEM correlator programs, the general formulation is not fully handled here. Large amount of EISCAT data, however, exist basing on the old Universal Program routine and in many experiments two or more datablocks were computed

into a single UNIPROG matrix. In many cases the decoding is done by writing a program which handles a single experiment only and it is often done by using externally designed start address tables for the pieces of UNIPROG matrix diagonals needed in the decoding. Because the calibration philosophy in those early experiments was not unique (one had to do what one could by using the Universal Program) a general formulation is too complicated to be practical. It is, however, helpful to have the formula 7.2 written into a more advanced form.

Let us assume that the data computed by the UNIPROG routine is a combination of three data blocks 1,2 and 3 having the number of samples DAT1, DAT2 and DAT3. So the total number of samples is DAT1+DAT2+DAT3. The expression for the ADDR(g,i) for the datablock 2 is then the following:

(7.15)

$$ADDR(g,i)=RES+((i+1)*DAT1+i*DAT3 +i*DAT2-((i-1)*i/2+DSU(i))*LI)/(GAT+1)+g-1$$

Note that in the old Universal Program based data the parameter GAT is always zero. The formula 7.15 is quite general because the data blocks 1 and 3 can be combined data blocks, too, or they can be empty. Note also, that the formula is valid for i=0 which gives the addressing of the X-profile points related to the data block 2.

If the data block 2 under consideration is eiher background or noise injection data one can obtain corresponding ACF estimate by putting DSU(i)=0 and averaging over all the available "gates" at every lag. The number of points available in the average for a given lag i is

$$NOCALPO(i) = (DAT2-i*LI)/(GAT+1) \qquad (7.16)$$

$$i=0,1,\ldots,MLA$$

Finally it can be mentioned that in certain special cases it is possible to design a special correlator program which does the described decoding in real time. It is also possible to design the measuring algorithm in the way that the power profiles and pulse codes are in gain balance by alternating the roles of the power profile and pulse code channels. A few ready measuring algorithms using this principle exist in the GEN-PROGRAMS library.

3. DECODING OF PULSE TO PULSE CORRELATION DATA COMPUTED BY THE GEN-SYSTEM UNIPROG ROUTINE

Pulse to pulse correlation can be handled with almost the same formalism as used for pulse codes. In a common situation one has separate data and calibration parts, because it is difficult to get clutter-free sky noise and noise injection estimates when transmitting and receiving a high repetition rate pulse to pulse correlation oriented modulation. The calibrations are only of the gain and offset voltage as in pulse codes. The receiver impulse response function has only small spatial effects. It does not affect the results in the frequency domain because the modulation and receiver bandwidths are always very much larger than the target bandwidth. Gain and offset calibrations can be arranged by applying the powerprofile routine to suitable set of calibration samples.

The system of the pulse to pulse correlation code can be defined in the same way as the system of the pulse code. If the code contains 5 pulses (suitable for 4 delays in the estimate) transmitted at equal intervals, then the system of the code is 1:1:1:1 and the lag increment must be the number of the samples in one receiving period after every transmission. The parameter UNINOSAMPLES is the total number of samples in the whole data vector, in this example five times the number of samples taken during one receiving period. One notices that in this example lag 2 appears in three system sums (SSU) each one having a different decoding sum (DSU). In this situation we denote DSU = DSU(i,k), where DSU(i,k) takes the values arising from all the SSU:s having the value i. One can index k from 1 to kmax(i), where kmax(i) is the number of different allowed combinations of the system elements giving the same system sum i. This number is dependent on the particular pulse to pulse correlation code. In the above simple example, kmax(2)=3 and DSU(2,k) obtains thus the values DSU(2,1)=0 , DSU(2,2)=1 and DSU(2,3)=2. When defining the DSU(i,k) in this way, one quarantees that the SSU=i exists for every k = 1,...kmax(i) and the formulation becomes simplified. Thus the same range gate appears in three places along the lag 2 diagonal in the UNIPROG matrix. In the case of pulse codes this were an error in the code. In the case of pulse to pulse correlation, it is allowed. The formula given below follows the notations given above and is, in fact, a relatively general formula. It gives the target autocorrelation function estimate PACF(g,i) normalized to boxcar weighting.

$$PACF(g,i) = \sum_{k=1}^{kmax} DATA(RES+g-1+(DSU(i,k)*LI)/(GAT+1)+DSTA)/kmax \qquad (8.1)$$

where

i= 1,2,...,LAGMAX and SSU=i exists for every k

g= 1,2,...,LI/(GAT+1)

DSTA=(i)*(UNINOSAMPLES-i*(i+1)*LI/2)/(GAT+1)

Figure 8.1. shows schematically the pulse to pulse correlation in terms of the UNIPROG matrix in a very simple case.

The rules for gating and lag increment are in principal the same as in case of the pulse codes.

The simple methods described above are not often effective enough. The main limitation is that the simple UNIPROG based data structure demands usually too much memory space in pulse to pulse correlation applications. The GEN-PROGRAMS library contains pulse to pulse correlation programs with special correlator programs written so that as much of the decoding and data compression as possible is done already at the correlator level. This is, in principle , done by programming the correlator to compute the SSU and DSU parameters for a given code and by programming then essentially the formula 8.1 into the correlator and limiting the computation only to the matrix elements which are needed. Those programs demand individual data handling procedure which is not handled here.

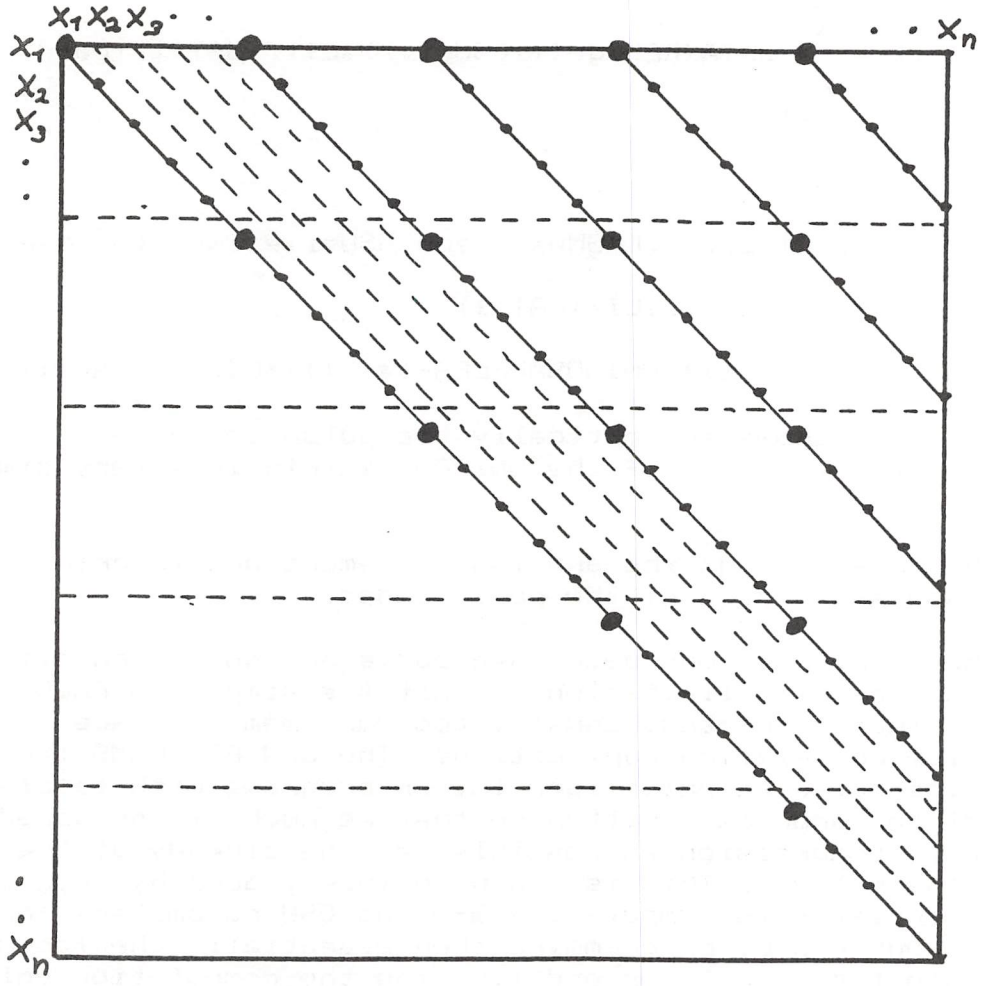


Fig. 8.1. UNIPROG matrix in pulse to pulse correlation. In the example, lags 0,1,...,4 are computed. Five range gates are handled. The enhanced points show the elements belonging to the first range gate. For details see the text.

9. THE LONG PULSE ALGORITHM IN THE GEN-SYSTEM

The LONGPULSE subroutine in the GEN-SYSTEM library differs from the earlier algorithms in a very essential way. This routine, together with the routine used to compute the calibration part in the program GEN-REMOTE, are the only GEN-SYSTEM routines which really differ from the earlier approaches to the incoherent scatter problem. The starting point was to develope an algorithm which does not waste information and which simultaneously has a well defined spatial response. The information is fully utilized if, and only if, every UNIPROG matrix element which can have contribution from the target within the wanted range is computed at least once. The spatial response of the algorithm must be such that the resolution is about the same at every lag. It is also clear that at every lag the nominal middle point of the volume must be at the same range and that the spatial weighting function must be symmetric around this middle point.

There are two possible solutions to the problem. The first one is an algorithm, which has the same absolute volume boundaries at every lag. The second possibility is to have about the same number of crossproducts at every lag (it is impossible to have exactly the same number of crossproducts at every lag and at the same time have the volume middle point at constant range). The first possibility was selected for the GEN-SYSTEM long pulse algorithm because it is statistically good and can be programmed in EISCAT using a straighforward and fast algorithm. The second possibility, although demanding less multiplications, is slower and demands more program steps. In certain applications, however, it can be better from the target ACF spatial filtering point of view.

The selected algorithm has the following properties:

1) The absolute boundaries of the volume are the same at every lag of the target ACF estimate.

2) The spatial resolution is programmable within the limits dictated by the modulation.

3) The statitistical significance of the estimate is of the same order of magnitude at every lag, when using practical parameters.

4) The spatial overlapping of the volumes has about the same value at every lag, not however exactly the same, because no such algorithm can exist for long pulse modulation.

5) The resolution and overlapping are not independent parameters but coupled together. These cannot be made independent without violating the demand of making use of all the possible information obtainable from the target.

6) The algorithm is fast enough for most practical applications.

The algorithm is schematically shown in the Fig. 9.1. The elements of the UNIPROG matrix belonging to the same volume are bounded by a polygon. All the elements of the same diagonal within the polygon are added together. The figure shows how the first two volumes are computed and it also shows that some of the elements are used several times in the computation and in the implemented algoritm they are, in fact, recomputed each time they are used. Although it is possible to develop an algorithm such that the multiplications are not duplicated, the resulting program is longer and , in fact, slower. The reason is that the accumulation has to be done twice in any case and it takes exactly as much time as multiplication and accumulation. Thus no saving is obtained but, instead, more overhead is needed.

From Fig. 9.1. one can also easily see the following properties of the algorithm:

1) The parameters which fully define the computation are the number of lags (called MAXLAG in the GEN-SYSTEM programs) and the number of elements added together in the main diagonal (called VOLUMEINDEX in the GEN-SYSTEM programs). The only additional parameter needed is the number of gates (called NOGATES in the GEN-SYSTEM programs).

2) The first sample of the data vector is used only once in connection with the computation of the longest delay of the target ACF estimate for the first volume. This has a consequence, that the middle point definition of the first volume demands both the MAXLAG and VOLUMEINDEX together with the instant, when the sampling starts and it demands also the sampling interval.

3) The gate separation is only a function of VOLUMEINDEX and the sampling interval.

The spatial response of the algorithm is shown schematically in Figure 9.2. and is explained below. First, we show that the algorithm really has the same absolute volume boundaries at every lag. The volume size is defined by VOLUMEINDEX denoted by V. Assume that the first sample used

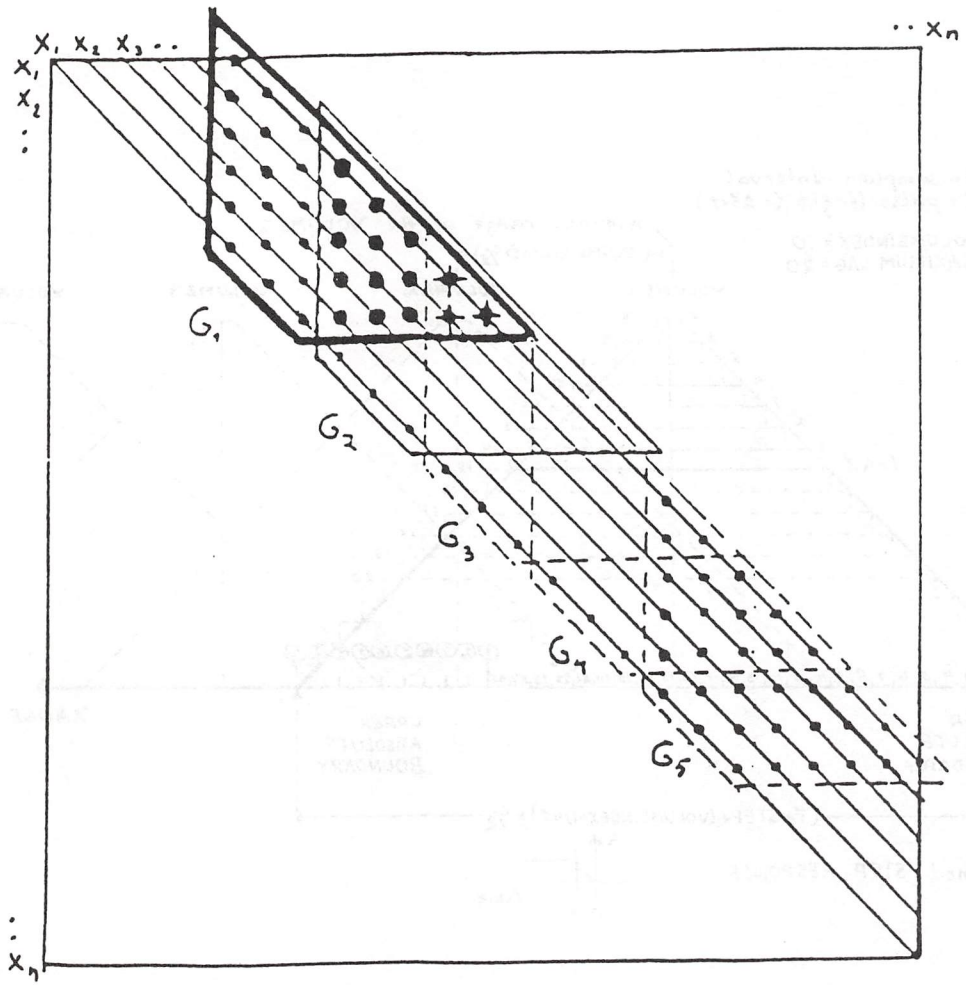


Fig. 9.1. GEN-SYSTEM longpulse algorithm. The example shows the case when 4 volumes are computed with parameters VOLUMEINDEX = 3 and MAXLAG = 5. The elements along the same diagonal within the same polygon are added together. Special marks used for the elements of the first gate , which are also used in the neighboring gates. For details see the text.

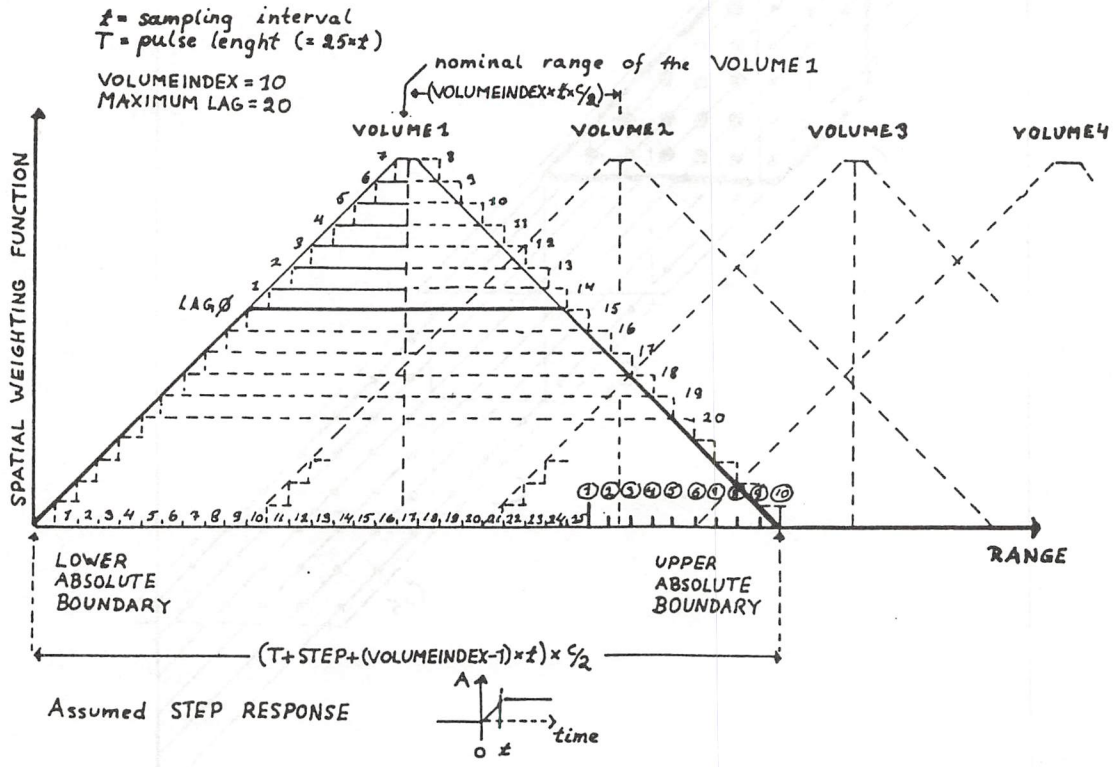


Fig. 9.2. Spatial weighting function formation in the GEN-SYSTEM longpulse algorithm. for details see the text. In the example the pulse lenght T=25*t, where t is the sampling interval, VOLUMEINDEX is 10 and MAXLAG = 20. The given parameters are very practical in real experiments.

for the zero lag estimate is Xm. The first crossproduct is then Xm*X^*m and the last one X(m+V-1)*X^*(m+V-1). The lowest border of the volume is defined by the lowest border of the illumination related to the sample Xm. The highest border of the illumination is in similar way dictated by the highest border of the illumination related to the sample X(m+V-1). In every diagonal the spatial response is the same along the diagonal and the ranges of the contributing volumes change by the amount t*c/2, where t is the sampling interval. At every element Xn*X^*(n+i) the lower border of the volume contributing to the element is defined by the illumination related to the sample X(n+i) and the upper border of the volume in similar way by the sample Xn. At every lag i the first element used from the UNIPROG matrix is of the form X(m-i)*X^*m and thus the lower border defined by the sample Xm is the same as it is in the zero lag. The last element used for the estimate is X(m+V-1)*X^*(m+V+i-1) and the upper border is again the same as it is in the zero lag estimate. The spatial response of all the other crossproducts used in the estimate is such that both the lower and upper boundaries are within the absolute boundaries of the volume.

The algorithm can be briefly defined as follows:

1) The volume size for a given long pulse is defined by the VOLUMEINDEX which shows how many neighboring elements Xm*X^*m, X(m+1)*X^*(m+1), ... , X(m+V-1)*X^*(m+V-1) are summed together to obtain the zero lag estimate.

2) In all the other lags, each crossproduct used in the summation contains some of the samples Xm, X(m+1), ..., X(m+V) as a multiplicant.

3) Every element of the UNIPROG matrix having spatial contribution within the lowest border of the first possible volume and the upper border of the last volume is used at least once and the elements of the main diagonal are never used twice.

The rules given above describe the principles of the algorithm. The algorithm is formally described by:

$$A(g,i)=\sum_{k=1}^{V+i} X(M+g*V-i+k)*X^*(M+g*V+k) \qquad (9.1)$$

A(g,i)= the ith lag of the target ACF estimate for the gth gate

M = MAXLAG , the index of the maximum delay to be computed

V = VOLUMEINDEX, which is the number of points added together along the main diagonal to to obtain the ACF estimate at zero lag

g = gate index 0,1,....,NOGATES-1

i = lag index 0,1,...,MAXLAG

The number of UNIPROG matrix elements used for a given lag i is:

$$N = V+i \quad \text{,where} \tag{9.2}$$

N is the number of elements,

V is the VOLUMEINDEX, and

i is the lag index.

The target contribution for a single element of the UNIPROG matrix is directly proportional to P given below if the modulation is a simple long pulse.

$$P = (T-i*t) \quad \text{,where} \tag{9.3}$$

T is the total length of the pulse, and

t is the sampling interval

The total contribution from a target volume at a lag i is proportional to

$$B = N*P = (V+i)*(T-i*t) = t*(V+i)*(T/t-i), \tag{9.4}$$

which is a downwards opening parabola in B,i coordinate system.

It is practical to normalize the parameter B to unity at zero lag.

$$W = B/(V*T) \quad \text{, or using earlier notation} \tag{9.5}$$

$$W(i) = (t/V*T)*(V+i)*(T/t-i)=(1+i/V)*(1-i*t/T)$$

The parameter W(i) is the weighting factor appearing in the long pulse data handled by GEN-SYSTEM algorithm. Figure 9.3. shows an example with certain computing parameters.The target autocorrelation function is finally obtained as follows:

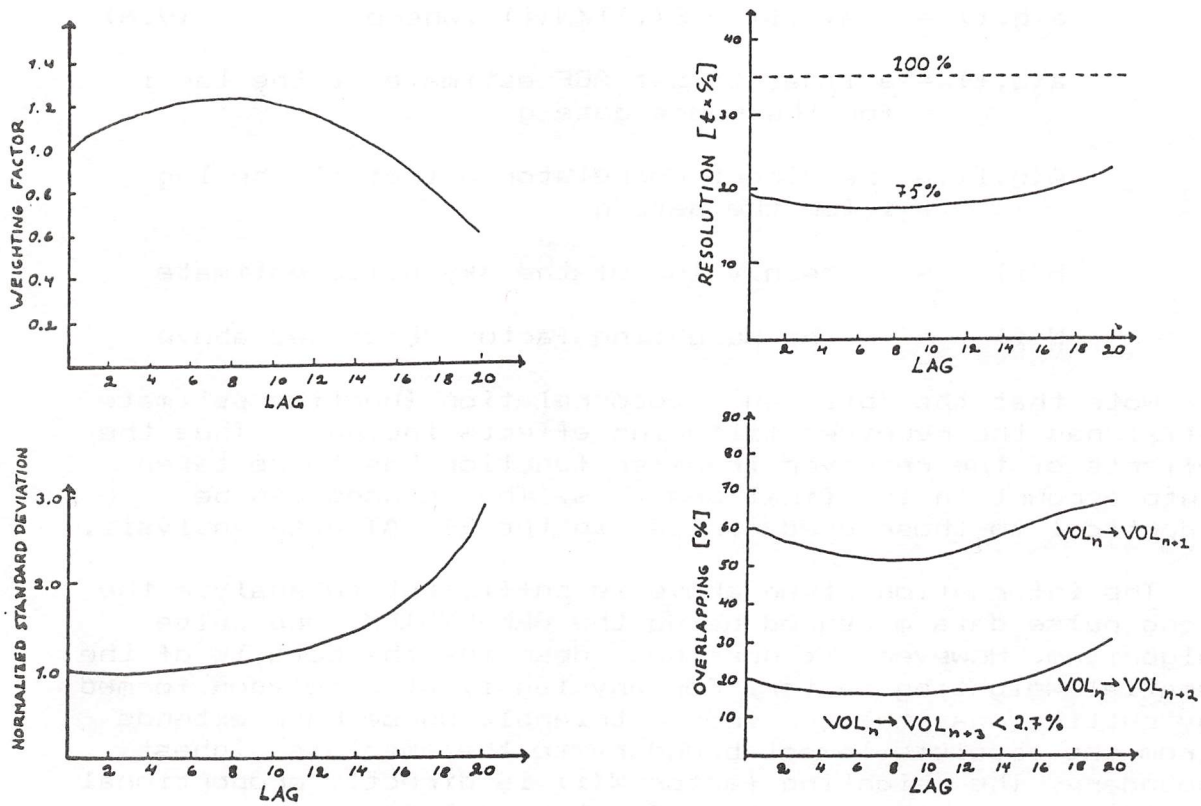


Fig. 9.3. Example on weighting factor, standard deviation, resolution and overlapping in the GEN-SYSTEM longpulse algorithm. The computing parameters are the same as used in the Fig. 9.2. For details see the text.

$$a(g,i) = (A(g,i) - S(i))/W(i) \text{ ,where} \qquad (9.6)$$

a(g,i) = a final target ACF estimate at the lag i for the range gate g

A(g,i) = the direct correlator output at the lag i for the gate g

S(i) = mean value of the sky noise estimate

W(i) = the weighting factor described above

Note that the obtained autocorrelation function estimate still has the receiver filtering effects included. Thus the effects of the receiver transfer function has to be taken into account in the final analysis. The methods can be identical to those used earlier in the EISCAT data analysis.

The information given above is sufficient to analyze the long pulse data measured using the GEN-SYSTEM long pulse algorithm. However, it does not describe the details of the spatial weighting factor. For any lag i, at a polygon formed by cutting away the top from a triangle whose base extends from the absolute lowest boundary to the absolute highest boundary. The weighting factor W(i) is directly proportional to the area of the weighting function polygon.

Fig. 9.2. illustrates the details of the weighting function. The staircase behaviour is for the case when the receiver bandwidth is very high. The straight line, on the other hand, closely approximates the case when the sampling interval is matched with the receiver bandwidth. It has to be remembered that the spatial weighting function is never exactly as clean as shown. However, it is a valid approximation if the step response function of the receiver system is well behaved. Some warning has to be given if the highest possible lags are computed. In such a case, the exact behaviour of the receiver step response has to be considered when estimating the spatial weighting and the exact value of the weighting factor.

The spatial weighting function must be symmetric and it is, as can be seen. However, at this point the distance factor effects within the volume are not taken into account. The distance factor deformation of the volume does not distort the target ACF estimate if at every lag the spatial weighting function is the same. It is, however, impossible to have an algorithm having exactly the weighting function at every lag in case of long pulse modulation. The GEN-SYSTEM weighting function is a rather well behaving one and it is propably relatively safe to assume that the differences in the weighting function shape do not cause severe distortions. If the weighting function is too different at different lags, then the distance factor weighting withing the volume causes the target autocorrelation function estimate to have a different nominal altitude at different lags. The reason for

this is that the dimension of the volume is usally not small compared with the range to the nominal middle point of the volume. This is relatively easy to show and not handled here in more detail. It can be mentioned that the GEN-SYSTEM spatial weighting function is much better behaved than the weighting function of the algorithm used earlier, where often the total dimension of the volume changes by more than one order of magnitude and the absolute weighting factor sometimes by three orders of magnitude within the delays of the ACF estimate. The GEN-SYSTEM volume has constant boundaries and in practical cases the weighting factor is the same within factor of 1.5-3. Also the spatial weighting function is not very different at different lags.

The weighting factor (the area under the spatial weighting function) as a function of lag is a parabola , which maximises at the delay, when the spatial weighting function obtains a form of a triangle or almost a triangle. The dimension of the volume along the beam is

$$D = (T+t*V+STEP)*c/2 \quad \text{,where} \tag{9.7}$$

D = the dimension of the volume along the beam,
T = the pulse length, and
t = the sampling frequency
V = the VOLUMEINDEX
STEP = the receiver step response time

The lowest boundary of the volume is at a position

$$L = (d-STEP)*c/2-T*c/2 = (d-STEP-T)*c/2 \tag{9.8}$$

L = the range to the lowest border of the first volume

d = sampling delay from the leading edge of the pulse to the first sample used in the zero lag computation of the first gate.

The middle point of the volume is at position

$$R = L+D/2 \quad \text{,where} \tag{9.9}$$

R = the nominal range of the volume

Because the algorithm can not use the first elements of the UNIPROG matrix for the zero lag computation of the first gate, but skips over the same number of points as the maximum lag (MAXLAG) , the first sample has to be taken at delay

$$F = d-t*MAXLAG \tag{9.10}$$

F = the delay to the first sample

This gives the necessary information to calculate the instant of start sampling for a given nominal range of the first gate. Note that the formula given above do not take into account the distance factor changes within the volume.

The range separation between the gates is simply

$$S = V*t*c/2 \qquad (9.11)$$

S = the range difference between the nominal middle points of the gates.

The properties of the spatial weighting function affect also two important parameters, the "overlapping" of the gates and the spatial filtering properties of the modulation-algorithm combination.

The overlapping is here defined as correlation of the results between the neighboring gates. This definition differs from the one used earlier in EISCAT. The "overlapping" defined in those algorithms is only the percentage of the common input data samples used in two neighboring gates and the "gate" is defined by truncating the sampled data vector into "blocks", where the block size is usually the same as the pulse length divided by the sampling interval. Note that practically everything is "wrong" in this old way:

1) The "gate" is not a well behaving gate in space as shown earlier.

2) The overlapping does not define the real overlapping of the volumes. It is instead a heavy function of the lag, being very high at zero lag and always going to zero at the longest possible lag.

3) Finally the habit to define the "data block" in the given way to define a gate does not have any other reasons but the historical ones.

In the GEN-SYSTEM the overlapping is defined in the following way. At any lag the spatial weighting functions cover a common part invariably. It is, in fact, impossible to define a reasonable algorithm were the neighboring gates do not have a common volume. The ratio of the common part to the total volume is defined as overlapping. The definition does not take into account the dimension of the common part, it only counts the percentage of the contribution arising from the common volume relative to the total signal at a given lag. The overlapping is a function of lag but not too heavily. The overlapping factor has a minimum at the same lag

where the weighting factor has the maximum. Around the overlapping minimum a rather accurate expression is relatively easily defined. The overlapping minimum, spatial resolution maximum and the weighting factor maximum occur for the lag index

$$i = \mathrm{NINT}((T/t-V)/2) \quad , T/t > V, \qquad (9.12)$$
$$i = 0 \quad , T/t < V$$

NINT defines a function "nearest integer"

For that value of i the overlapping G in percent is quite accurately

$$G = 100*((V/2+i)*T/t)/((V+i)*(T/t+V))) \qquad (9.13)$$

At all the other lags the overlapping factor is higher but not usually by essential amounts.

The overlapping shows the independence of the estimates related to different volumes. It is clear that if the volumes have a large common part, the results must necessarily correlate. This correlation is not, however, a measure of the correlation of errors. In fact the parameter called overlapping does not say very much about the quality of the data. The only thing it says that if the overlapping is very high, say around 70 %, then necessarily the results cannot differ very much from gate to gate and if they differ, then the question may be about the inaccuracies in the measurement - the difference is often difficult to justify from empirical point of view. High overlapping is not an indication of badly designed algorithm. The GEN-SYSTEM longpulse routine is such that if one computes the ACF estimate in the way that the spatial resolution is only slightly worse than the maximum theoretical resolution obtainable with the modulation (i.e. small value in the VOLUMEINDEX) then necessarily the overlapping factor becomes high. This arises from the demand to use all the possible information. In many cases this is exactly the right choice of the parameters, especially if one studies phenomena below the F-layer peak. High overlapping is not economical from computation time point of view and some care is needed.

In the older systems one reason for defining the overlapping was to save information, because high overlapping caused that greater amount of UNIPORG matrix elements were computed. In the GEN-SYSTEM this is not the case. The algorithm uses always all the available information. The resolution and gate separation are the primary parameters the user has to consider and depend on the application of the experiment. The overlapping is an "automatic byproduct", which necessarily enters into the data at some level.

The overlapping defines some kind of "spatial correlation" of the results between two or even more neighboring gates. The correlation of errors must, however, be handled in different way. The correlation of errors within a single ACF estimate between different lags arises if the data is oversampled by essential amounts. This is a well known fact and handled in litterature in all details. Another type of correlation between errors is inherent in the diffuse target radar analysis and appears in several algorithms. It arises from the fact that the same UNIPROG matrix elements are used in ACF estimates for different gates. By assuming that the original samples in the data are "relatively independent" this last kind of correlation is easily handled in the GEN-SYSTEM programs. The correlation of errors between two neighboring gates is simply the number of common elements divided by the total number of elements used in estimating the given lag.

$$C = i/(V+i) \text{ , where} \qquad (9.14)$$

C is the correlation of errors between two neighboring gates at lag i.

It can be seen that in the GEN-SYSTEM programs the estimates at lag zero are statistically independent undepending on the overlapping assuming that the individual datapoints are relatively independent. At long delays of the ACF estimate the errors start to correlate heavily. The behaviour of these parameters is shown schematically in the Fig. 9.3. given earlier.

The result given above has the consequence, that if the VOLUMEINDEX is small, then there is high "spatial correlation" because of high overlapping and at the same time at long delays there is high correlation of errors, too. This means that in the analysis both the parameters and the errors correlate and thus the appearance of a similar pecualirity in two or more neighboring gates does not necessarily prove that the peculiarity is really a physical fact.

When considering the response of the long pulse modulation, one of the most important points is the filtering properties of the modulation-algorithm combination. There are two kinds of filtering one has to take into account. The target ACF is formed necessarily by using signal, which has passed through the receiver system. Thus the original time series is filtered and this filtering is not dependent on the algorithm used in the computations. It has to be taken into account in the final analysis and in the GEN-SYSTEM analysis it appears exactly in the same way as in all the other methods. The other form of filtering is spatial in nature and its effects are perhaps not always remembered. The function, which is filtered is the "lag profile". The lag profile describes the behaviour of a given delay of the target ACF as

a function of range along the beam of the radar. It is a function of normalized target ACF, cross section, distance factor and the receiver system transfer function. The lag profile as seen by the radar is necessarily filtered spatially in the way how the modulation filters it. The least filtered lag profiles appear as diagonals in the UNIPROG matrix. Every diagonal is a lag profile having only the spatial filtering, which necessarily arises from the modulation. This spatial filtering is generally a function of lag. In case of the long pulse modulation the elements in the diagonals of the UNIPROG matrix represent the mean values of the lag profiles and the mean value is spatially taken over a volume having along the beam dimension

$$S = (T-i*t)*c/2 \qquad (9.15)$$

This represent a boxcar weighting over the volume defined by S. The receiver effects to the spatial filtering can be taken into account by applying the reveiver step response function to the pulse of duration T-i*t. The resulting weighting function is no more exactly a boxcar function but somewhat distorted. The same result can be obtained by applying the step response function to the original modulation pattern. The modified pattern forms then the "effective illumination pattern".

When elements of the UNIPROG diagonals are added together in order to get suitable spatial response, the filtering becomes more effective. Finally the spatial filtering is a weighted mean of the lag profile over the volume and the weighting function has the form , which was shown earlier and called the "spatial weighting function". In GEN-SYSTEM algorithm this is to a good approximation "a triangle without the top". Thus the final estimates of the target ACF are formed from spatially low pass filtered lag profiles and the filter characteristics are generally different at different lags.

In the analysis the obtained ACF is connected to a certain altitude, the nominal altitude of the gate. This is exactly right if, and only if, at every used delay of the ACF the weighted mean value of the corresponding lag profile obtains the true value of the lag profile at the nominal altitude of the gate. This is valid at least in the following cases:

1) The lag profile is constant within the volume

2) The lag profile is only a linear function of range

3) The lag profile is an odd function of range around the nominal middle point of the volume.

The three conditions given above are not exactly necessary. None of the conditions is in realistic case exactly valid. In practice one has to work hoping that the condition 2 is valid to a needed accuracy.

If it happens, that the given conditions are valid for the lag profiles constructed from the normalized target ACF but they are not valid for the true lag profiles, which have also the distance factor and cross section dendence, then in cases that the spatial weighting function is not too different at different delays, the obtained ACF is still a correct target ACF within the volume but not exactly at the nominal altitude of the gate. This kind of situation may easily arise at short ranges and in case of heavy electron density gradients within the volume.

Some care is always needed when working with a long pulse. The new possibilities given in the GEN-SYSTEM allow higher spatial resolution with fully utilized information. This possibility should be used when needed . In some cases relatively short long pulses, 150 -250 us for example , used in connection with small VOLUMEINDEX can open new possibilities because of much better resolution than in earlier methods. On the other hand , if high resolution is not needed - it should not be used.

The total number of samples needed for N gates with a given VOLUMEINDEX and MAXLAG is

NOSAMPLES = N*VOLUMEINDEX + 2*LAGMAX , where (9.16)

N is the number of gates

NOSAMPLES is the number of samples needed in the input data vector.

If possible, the calibrations should be arranged with a separate subroutine call in order to avoid garbage ACFs in the data. This is not always possible.

EXAMPLE: MAXLAG is 15, VOLUMEINDEX is 10 and the number of clean gates for signal is to be 20, for sky noise is to be 5, and for noise injection gates is to be 2. How many gates must be computed? How many garbage gates appear? How many samples are needed etc. It is assumed that no possibility to arrange an individual subroutine call for the calibration gates exist and thus the data vector has to be handled as a single block.

SOLUTION: 20 signal gates demand 20*10 + 2*15 = 230 points
5 sky noise gates demand 5*10 + 2*15 = 80 points
2 noise injection gates 2*10 + 2*15 = 50 points

The total input data vector is thus 360 points. From formula 17 it can be seen that altogether 33 gates appear. The number of usable gates is ofcourse 20+5+2 = 27 and thus the algorithm makes 6 garbage gates with the given parameters. The number of result memory points is 33*(15+1) = 528. If one can arrange an individual subroutine call for all

the three different data types (signal, sky, noise injection) then only 27 gates appear and the result memory need is 432. Note also that in this last case one can also arrange the order of output data blocks as one wants by defining RESENTRY points in suitable way. This possibility is also faster from computational point of view.

In many of the GEN-SYSTEM programs more than one set of parameters can be defined and thus two or three different long pulses are possible in the same pattern. This is useful in some very long range applications and can be useful also in some VHF applications.

It is not necessary to have different modulation in order to utilize the possibility of different control parameteres. In fact it is sometimes practical to divide the range in two or three blocks and compute them with different parameters even if the illumination of the target is done with the same modulation. One application is in the UHF very long range measurements. One can measure in the short ranges all the parameters with relatively high spatial resolution but in the long range part decrease the resolution and number of lags and measure only doppler. In VHF system some opposite behaviour may be useful. One can measure the oxygen dominated part with one set of control parameters and the the upper part having helium and hydrogen contribution with another set of parameters. In the latter case one needs more lags because of the demand of more complicated analysis. Note also that the same illumination can be received by two or more channels having different receiver parameters. The calibration philosophy differs a little in that kind of applications because usually only one set of control parameters can be calibrated directly. The other set must be handled in a little more complicated way by modifying the sky noise estimate to match the used parameters. This is done by defining a set of weighting factors for the calibration data and it is a relatively straightforward task.

The last comment on the subroutine LONGPULSE is related to the computing time which depends on the parameters. In practical cases, the multiplication rate is between 3.5 and 4.5 MHz. The higher the VOLUMEINDEX the faster the algorithm is. Approximate time needed to compute a gate is:

$$T=((V-1)*(M+2)+M*(M+1)/2+4*M+3)*0.2 \text{ , where} \quad (9.17)$$

T is the computing time per gate in microseconds
V is VOLUMEINDEX
M is MAXLAG

In the earlier example with V=10, M=15, the computing time is thus quite exactly 67.2 us. Thus 33 gates take 33*67.2=2217.6 us. Thus it can be seen that with the given parameters the mean multiplication rate is 67.2/280 = 4.1666... MHz. (280 is the number of UNIPROG matrix elements

per gate using the given parameters). The formula given above does not take into account all the overhead needed in the program, however, the amount by which it underestimates the time can be corrected by adding 5 us per subroutine call. The example given above is quite practical. If the sampling interval is 15 us, the gate separation is 22.5 km. The 20 gates cover 427.5 km in range, for example 200 - 627.5 km. The GEN-SYSTEM can compute four such datablocks in 8888 us and thus within quite usual 10 ms repetition period one could compute easily quite many powerprofiles, too. The GEN-SYSTEM long pulse routine can be considered fast enough.

The GEN-SYSTEM long pulse part is much more advanced than the corresponding part in any other existing method. To what extent this flexibility is going to be used and what are the real benefits obtainable are seen in the future.

10. THE ALGORITHM GEN-REMOTE

The remote station algorithm, GEN-REMOTE, has been developed to carry out the real time computations at remote sites when the volume is illuminated by a long pulse. It cannot handle any other modulations. In applications where the modulation pattern does not consist of long pulse, the remote station computations can be handled by the UNIPROG part of some other GEN-SYSTEM program. This very special situation is not handled in this report because it does not appear in general purpose algorithms.

The GEN-REMOTE algorithm has four parts:

1) -timing check based on powerprofile computation or

2) -timing check based on ACF computation

3) -signal ACF

4) -background and noise injection calibration ACF (boxcar weighted)

The timing check is done by computing either a power profile or a set of overlapping ACFs (signal ACF algorithm) from the sampled data expected to contain the signal. The sampling can be commanded to start before the leading edge of the pulse illuminates the volume and it usually ends some time after the trailing edge has left the volume. Thus the data appears to have three different parts, separated by short transition intervals:

1) -samples before the illumination

2) -samples during the illumination

3) -samples after the illumination

In the powerprofile based timing check it is assumed that the number of samples which do not contain signal (or contain signal only from partly illuminated volume) is sampled to be the same both before and after the illumination. This number of samples is called MARGIN in the GEN-REMOTE. The number of samples taken during the illumination is called SIGSAMPLES. The power profile is computed over MARGIN + SIGSAMPLES + MARGIN points and after that the address counter returns to point to the first sample in the part of the data where the illumination is total. In the next phase an autocorrelation function estimate is computed with triangular weighting and the number of samples is "SIGSAMPLES" and the maximum lag is given in the APB register named "MAXLAG". Thus one does not have to compute all the possible lags. This part of the algorithm is very similar to the earlier ones. After computing the signal ACF the algorithm handles the calibration data in the way described later.

If one uses ACF based timing check (as done in the programs belonging to the GEN-PROGRAMS library) one defines the number of gates to be computed to be greater than one and gate overlapping in the sample space by MARGIN. In this application the parameter MARGIN has to be a negative number. One has to sample the data in the way that one of the computed gates uses signals when the remote station volume is fully illuminated.

One cannot easily use both powerprofile and ACF based timing simultaneously.

After the timing check and signal ACF computation the algorithm computes the calibration ACF:s.

In the beamwidth limited case, the algorithm used to compute the signal ACF is the only reasonable one. One cannot find any way to improve the measurement by new computational algorithms. The only way to get some improvement is to increase the accuracy of the background estimate. This cannot be done effectively by using the same algorithm as used in the computation of the signal ACF because that algorithm necessarily leaves part of the UNIPROG elements unused. GEN-REMOTE has been programmed to have a calibration part which computes the ACF estimate by using boxcar weighting. The maximum lag is the same as in the signal ACF, but the number of crossproducts added together at a given delay is a programmable parameter called CALNOPRODUCTS. The number of the calibration ACF gates is also a programmable parameter CALNOGATES.

The whole algorithm is shown on the UNIPROG diagonals in Fig. 10.1a and 10.1b. The ACF estimates used in the calibration part are totally independent without a single common element. It is often enough to compute two sky noise estimates and one noise injection estimate.

As in all the other GEN-SYSTEM programs, different receiver channels can either be added together or left separate. Altogether, two independent control parameter sets can be handled. Thus only two different pulselengths can be received but the number of pulses of a given length is not limited. The subroutine calls CALL REMOTE1 or CALL REMOTE2 do the calculations shown above; including the calibarations.

Also available in the GEN-REMOTE package is the subroutine call, CALL IMPULSE. This causes only the calibration part to be computed using the parameters CALNOPRODUCTS, MAXLAG and CALNOGATES to control the computation. This algorithm is not suitable for any applications with illuminated target. Simultaneous use of subroutine calls REMOTE1 or REMOTE2 and IMPULSE is difficult because the routine IMPULSE uses some of the temporary storages of REMOTE as control registers. The subroutine IMPULSE is a very powerful way to compute the estimate needed for the impulse response function determination and is used at all sites in the calibration programs.

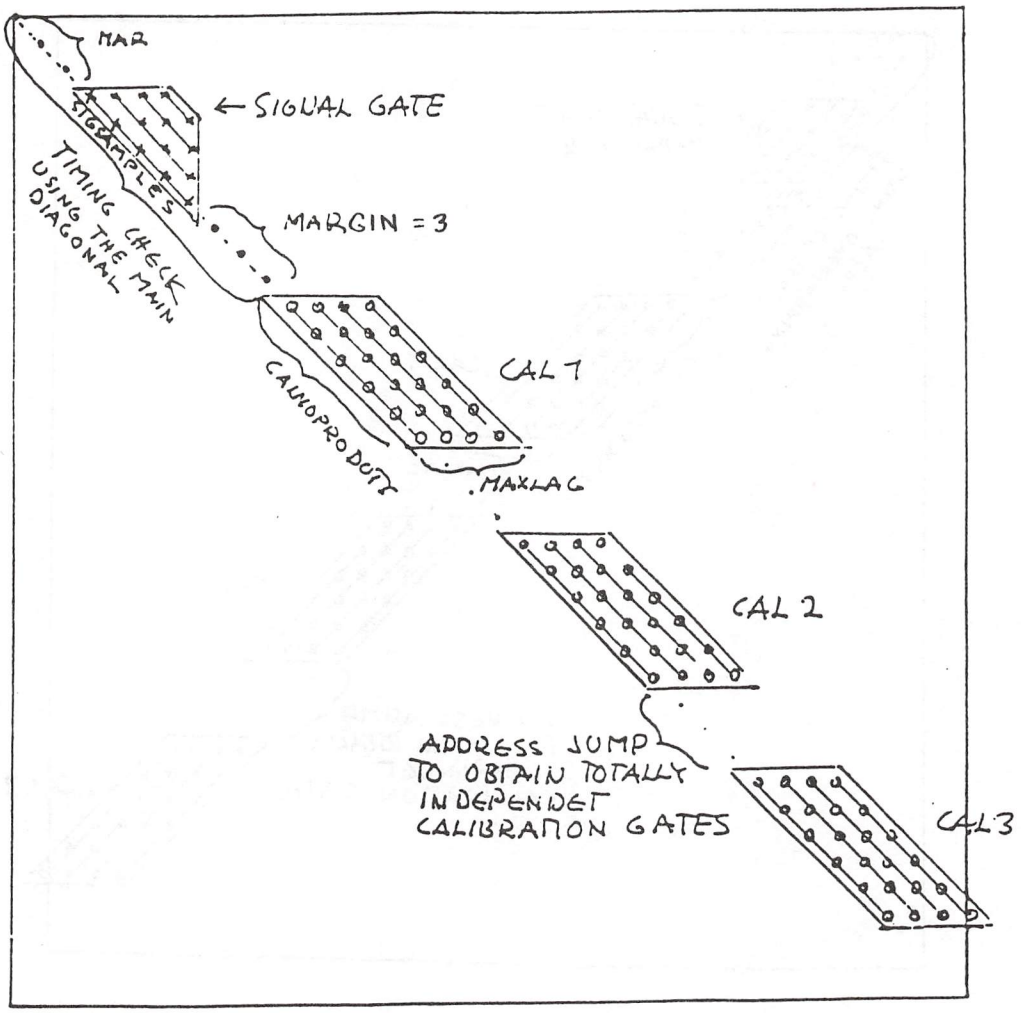


Fig. 10.1a. The algorithm GEN-REMOTE shown schematically using the UNIPROG matrix in the mode using the power profile mode of timing check. The parameters are the following in the example: MARGIN=3, SIGSAMPLES=5, MAXLAG = 3, CALNOPRODUCTS = 6, CALNOGATES =3. For details see the text. In the correlator the order is: TIMING CHECK, SIGNAL GATE, CAL1, CAL2, CAL3. The order of the lags in the autocorrelation function estimates is from 0 to MAXLAG. Note the different weighting factors in the SIGNAL GATE and the CAL gates.

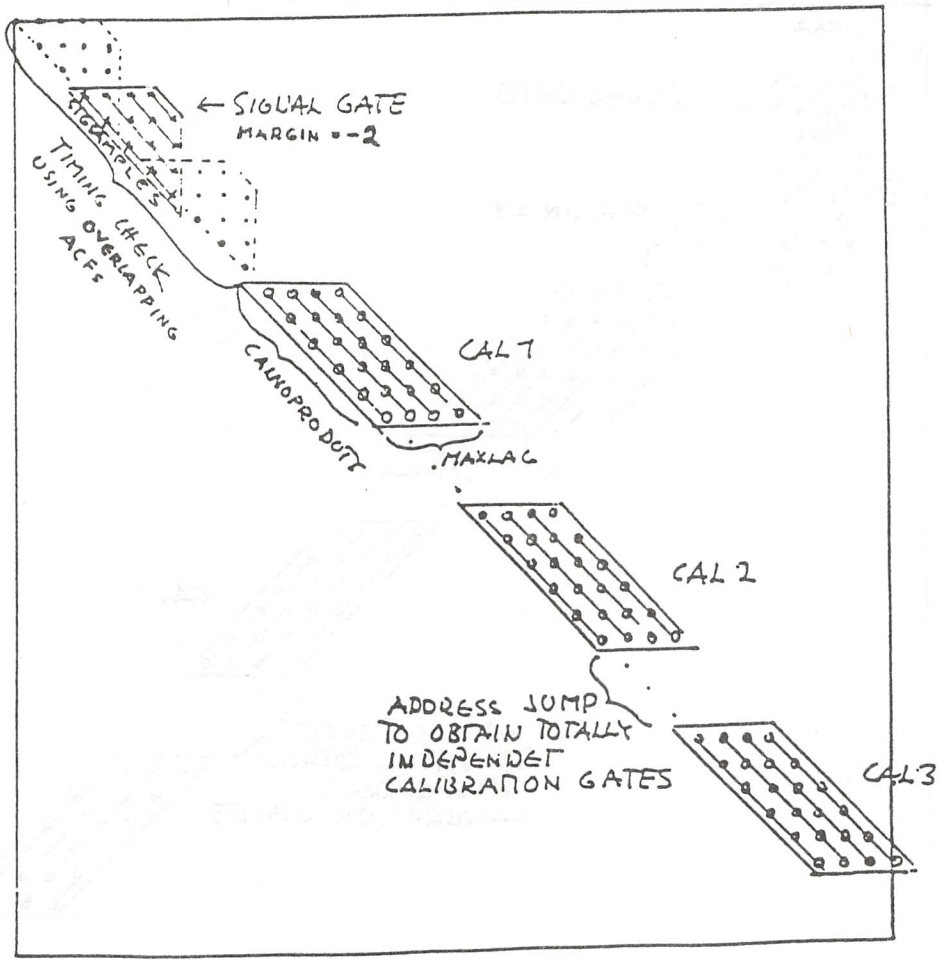


Fig. 10.1b. As Fig. 10.1a bun now GEN-REMOTE is commanded into the mode having autocorrelation functions for the timing check. The parameters are the following in the example: Number of signal gates=3 (this is loaded in the subroutine call), MARGIN=-2, SIGSAMPLES=5, MAXLAG = 3, CALNOPRODUCTS = 6, CALNOGATES =3. For details see the text. In the correlator the order is: TIMING GATE, SIGNAL GATE, TIMING GATE, CAL1, CAL2, and CAL3.

The GEN-SYSTEM correlator master program GEN-REMOTE can be used in several different modes and all possibilities cannot be handled here. The detailed information is given in the comment part of the file GEN-REMOTE:CLAN. (See Appendix 1).

The algorithms used in the GEN-REMOTE can be expressed with the following formulae. The powerprofile timing check is simply

$$P(i) = X(i)*X^{*}(i) \quad , \; i = s, s+1, \ldots, s+2*M+S \qquad (10.1)$$

M = MARGIN
S = SIGSAMPLES
Xs is the first sample of the channel
i.e. s is the start address of the data

The signal ACF estimate is computed by:

$$SACF(j) = \sum_{k=s+M}^{s+M+S-1-n} X(k)*X^{*}(k+j) \quad , j=0,1,\ldots,G \text{ and} \qquad (10.2)$$

SACF(j) is the ACF estimate at lag(j)

G is the maximum lag MAXLAG

Finally the calibration gates are computed from the formula:

$$CACF(j) = \sum_{k=i+2*M+S+g(C+G)+s}^{i+2*M+S+C-1+g(C+G)+s} X(k)*X^{*}(k+j) \quad , \; j=0,1,\ldots,G \qquad (10.3)$$

and CACF(j) is the calibration ACF estimate at lag(j),
C = CALNOPRODUCTS
g = the index of the calibration gate,
g=0,1..CALNOGATES-1
s = the start address of the data

The subroutine IMPULSE computes as shown in 10.3 but the parameters M and S are zero.

In practice the GEN-REMOTE structure makes possible to fully utilize all the long pulses transmitted for the monostatic purposes regardless of how many such pulses exist. At the same time it is possible to sample the background or noise injection all the time when it is possible and compute almost all the relevant crossproducts from the data. Especially at long delays and with relatively short long pulses, the obtained background estimate may be more than one order of magnitude more accurate than in the earlier

algorithms. In practical applications the experiment can be arranged so that in commonly used 10 second dump the background estimate obtaines the accuracy of 0.1% or better. The data dumps also become short. If channel addition is utilized, a normal datadump is of the order 100-150 complex words. More specifically it is, using powerprofile timing,

$$D = 2*M+S+G+1+(K+J)*(G+1) \qquad (10.4)$$

or

$$D = 2*M+S+(1+K+J)*(G+1)$$

D = the number of output points per channel
K = the number of sky noise gates
J = the number of noise injection gates
(K+J = CALGATES)

The only drawback of having different weighting for the calibrations and signal gate is that certain manipulations have to be made before subtracting the sky noise estimate from the signal gate.

If all the sky noise gates are added together, then the ACF estimate containing only the scatter contribution is obtained from:

$$ACF(j) = SACF(j)-CACF(j)*(S-j)/(C*K),\ j=0,1,\ldots,G \qquad (10.5)$$

ACF(j) is the ACF of the scattered signal

Note that the formula 10.5 gives the signal ACF with a triangular weighting factor. The ACF with boxcar weighting is obtained from

$$TACF(j) = ACF(j)/(S-j)\ ,\ \text{where} \qquad (10.6)$$

TACF(j) is the signal ACF estimate with the same weighting factor at every lag.

Note that after the receiver impulse response function correction TACF(j) is the true target ACF.

When using the GEN-REMOTE in system calibration mode (CALL IMPULSE calls only), the ACF estimates are boxcar weighted. If the number of lags is high enough, and the sampling is free from aliasing, then the fourier transform gives directly the system transfer function.

The algorithm GEN-REMOTE is rather fast. Most of the time is used to compute the calibration part in a practical experiment. The time needed for a single calibration gate is

approximately given by

$$D = (C*(G+3)+5)*0.2 \quad \text{, where} \qquad (10.7)$$

D = the computing time per calibration gate in microseconds.
C = CALNOPRODUCTS
G = MAXLAG

The program GEN-REMOTE is an essential part of the GEN-SYSTEM. Its applicability to multichannel applications and possibility for two different long pulses makes the use of a special pulse for the remote stations unneccessary. The benefits obtained in the use of the transmitter duty cycle are considerable. The possibility for a very compressed data output block causes considerable savings if effectively used.

EXAMPLE: The intersection volume is illuminated by a 350 us pulse.
One wants to make use of 300 us of the illumination. The sampling interval is 10 us. The filter delay is 21 us. The wanted MARGIN is 15 samples. The pulse is transmitted at instant 100. The user wants to start sampling at 100 (in the remote station TARLAN file). The maximum lag is 20. The user wants to have two sky noise gates and one noise injection gate with maximally good statistics. The pulse repetition period is 10 ms. What is the pratical solution?. What is the computing time needed for that channel?

SOLUTION: One leaves 25 us unused both in the beginning and at the end of the illumination. The leading edge of the pulse arrives at the middle point of the volume at instant 100. The first signal sample is wanted at 125 and to that one has to add filter delay 21 us . Thus the first signal sample has to be taken at 146. One wanted to have 15 sample margin. Thus the real start sampling command has to be at an 146-15*10=61. The TLAN file start sampling is given at 100. Thus the OFFSET-PPD given in the ELAN file must be -104. The total number of samples related to the illumination is 15+31+15. Thus one has to give channel off command at 100 + 61*10 - 10/2 = 705 in order to guarantee the right number of samples.

Before starting the sky noise samples one has to wait long enough so that the illumination has also passed through the near sidelobes. This time depends slightly on the geometry of the experiment. Here we assume that 500 us gap is enough. Turning on the noise injection demands about 100 us and at the end one usually needs some extra time also. We reserve another 100 us. One can thus start the sky noise sampling at 1205 and for calibration type sampling one has 10000-1205-100-100=8795 us available. Thus the maximum number of samples one can take with this timing is 880 for sky noise and noise injection. The nearest number , which can be divided by 3 is 879 and the division gives 293 samples. The number of samples needed for a single calibration gate is

CALNOPRODUCTS+MAXLAG and thus CALNOPRODUCTS is 273. The computing time for a single sky noise gate is (273*(20+3)+5)*0.2=1256.8 and all the 3 gates demand about 3770 us. One could, in fact, compute two channels with these parameters in 10 ms. The computing time is not a problem and our single channel solution works in every respect. One has to sample 2*293=586 sky noise samples and 293 noise injection samples for the parameters given. Altogether 940 samples are taken and 145 result memory points used.

The repetition frequency is 100 Hz. One 10 s cycle has thus in practice a little less than 1000 receiving periods. The sky noise estimate at every delay is based on about 1000*273*2 estimates of the delayed product. This is 546000 and because the experiment in fact can be carried out at two channels with the given parameters the background accuracy of 0.1% can be obtained at every lag in the boxcar weighted sky noise estimate.

11. PRINCIPLES OF MEASURING ALGORITHMS WRITTEN USING GEN-SYSTEM

The purpose of this report is not to give a comprehensive description of GEN-PROGRAM library or a detailed programming description. The general principles are described only and two examples given in the appendixes. When more experience is obtained with the programs and the program library is more thoroughly tested, then a report related to measuring algorithms will be published.

The measuring programs used at EISCAT in 1984 and before did not always make efficient use of the transmitter duty cycle. Those programs containing no high resolution E-layer part usually consist of three modulations; a power profile pulse, a monostatic long pulse, and a remote station pulse. The RF duty cycle is divided so that the remote station pulse takes about 70% and the monostatic part about 30% on the available RF power. This kind of approach is used in CP-3 and CP-2. The programs containing a high resolution low altitude part have 2 long pulses for remote station use, one of which is used for monostatic purposes, two pulse coded channels, and two different power profiles. About 60 percent of the RF dutycycle is transmitted for monostatic purposes and the rest for the remote sites. The monostatic part usually contains many gaps and thus only about 60% of beam on time is utilized during transmission. All the versions of the CP-1, the field aligned common mode experiment have been roughly as described above.

Some special programs like EFOR and ESLA series of programs developed by the author in 1982-83 can carry out the high resolution E-layer part much more effectively than any of the general purpose programs but do not contain a monostatic F-layer part. In those programs about 30-50% of the power is transmitted for the remote station purposes only. These programs are based on the original Universal program principle and it is very difficult to program other than high resolution part for the monostatic mode.

The GEN-SYSTEM is developed to allow effective general purpose programs to be written. The high resolution part can be programmed to be as good or better than in the earlier Universal program based special programs, the monostatic F-layer part can be programmed to be much better than in any of the earlier programs, and the remote station part as good or better than in the earlier developed programs. This is achieved by following the principles listed below:

1) -the part of the RF duty cycle needed for the remote sites is divided into several long pulses all of which are utilized for the monostatic measurement. Thus the pulse length is selected to fit the monostatic measurement and the same pulse length is repeated as many times as necessary to achieve good statistics at the remote sites. In this way, 100% of the RF duty cycle can be used for monostatic measurements.

2) -the high resolution E-layer part is written so that the gaps existing in the pattern are as few as possible. This usually demands 4 pulse-coded channels if the modulation is not phase-coded and 2 channels if it is. Two power profile channels with similar resolution are included to fill the gaps in the pulse coded pattern. Alternatively a separate powerprofile group (usually 4 pulses) is transmitted having the same spatial response as the pulsecodes sharing the same channel as the pulsecode.

3) -together with the E-layer power profiles, separate F-layer power profiles with coarser resolution are included. The E-layer powerprofiles share the channel with pulsecodes or with F-layer powerprofiles.

4) -if the memories and the computation time allow, the high resolution part is oversampled and a few of the resulting heavily overlapping gates are added together using the "gating " described earlier.

5) -phase coding is used in the high resolution part if the measuring problem can be more effectively solved in this way.

6) -calibrations are done effectively

7) -the receiver parameters are carefully considered.

8) -the available transmitter duty cycle is fully used.

The examples found in the appendixes show the programming philopsophy and also how the GEN-SYSTEM correlator programs are used. The first example resembles CP-3 and the second example resembles CP-1. There are many other programs in the GEN-PROGRAMS library written along the same lines but having a little different bias in the optimization.

The program GEN-5 is a "CP-3 looking" experiment. It does not have as wide a range coverage as CP-3. Experiments having wider coverage cannot be optimized as well as GEN-5. The general structure of the GEN-5 transmitter pattern is shown below

The long pulses are of duration 350 us and the power profile pulses 70 us. The interval between TX-groups is 8777 us and the total repetition period is 17256 us. A gap of 5 us is left between every pulse for the settling time of the exiter. Delay between BEAMON and RFON is 50 us. The BEAM ON duty cycle is 12.05% and the RF duty cycle 11.36%. The sampling interval is 14 us on all channels and 25 kHz linear phase filters are used in the library version. The filter characteristics must be determined externally by using a suitable version of the GEN-CAL program.

The first gate of the power profiles is at 76.5 km range. The gating factor is 4 and thus the gates are at 10.5 km intervals. The spatial resolution is also 10.5 km measured at 75% contribution level. Altogether 30 power profile gates are measured and thus the last gate is at a range of 381.0 km. The first long pulse gate is centered at 200 km range. The parameter VOLUMEINDEX is 15 and MAXLAG is 15. Thus the long pulse gate separation at nominal middle points is 31.5 km. The number of gates is 25 and thus the last gate is centered at 956 km range.

The peak to peak resolution of the long pulse gate is about 84.0 km. The 75% level resolution at most lags is less than 42 km.

The maximum lag is 210 us and 16 lags are computed with a lag increment of 14 us.

The details of the experiment can be found in the Appendix 1 including all the necessary files of the experiment.

The second example resembles CP-1. It is programmed in a way that is very practical for many general purpose experiments covering both the E- and F-layer altitudes. In these experiments the E-layer oriented modulation and the F-layer oriented modulations are transmitted separately. The modulation blocks are:

1) - a multipulse group MPG for the E-layer

2) - an E-layer power profile group EPG, which has the same spatial response as the MPG

3) - an F-layer power profile group FPG. This can use the same channels and frequencies as EPG

4) - a long pulse group LPG

The MPG uses 4 channels, EPG 2 channels, FPG 2 channels and LPG 2 channels. The LPG is also the remote station modulation.

For the transmissions the MPG and EPG are combined to form the TXE group, and the FPG and LPG are combined to form the TXF group. The sequence is shown below and forms the "10-block" modulation used in several experiments in the GEN-PROGRAMS library.

Phase 1: transmit TXE
Phase 2: receive MPG and EPG ,calibrate LP channels
Phase 3: transmit TXF
Phase 4: receive FPG and LPG,calibrate pulse code and powerprofile channels

The TXE phase cannot last more than about 400 us maximum because all those modulations have been intended for low altitude sampling. The modulations have also the property that they become almost useless somewhere around 250 km or below, Thus the Phase 2 receiving period lasts only a short time, usually less than 2000 us. The transmitter is not ready for a new transmission until the duty cycle permits after the TXE phase. This takes about 3500-4000 us. In that period one can complete the signal reception , sample sky noise for the channels to be used as F-layer long pulse reception and also do the noise injection for the F-layer channels.

The F-layer transmission contains the LPG and FPG. The LPG channels have already been calibrated and FPG channels have sampled the E-layer powerprofiles. The transmission of the F-layer power profile pulses at the same frequency is not very dangerous. The E-layer powerprofile pulses have in practice little effect on the F power profile because they are short and they are at long ranges (about 600-700 km range when the F-layer power profile sampling starts. The F-layer power profiles do not disturb the E-layer power profiles because the period from FTX to ETX is quite long, 6-10 ms depending on the modulations in FTX.

The calibration of the power profile channels is done at the end of the "F-cycle" either in the F-layer gating mode or in the more compressed calibration gating mode. In most cases it is practical to do also the pulse code channel calibrations at the end of F-cycle, but not always. If done so, then the cycle starts directly with the ETX group transmission. The example algorithm GEN-4 is programmed in this way.

The structure given earlier has several benefits. If the E-layer measurement is phase coded, then the number of channels needed in the E-layer part is not 6 but usually 4. Then there is no need (and no possibility) to give two different tasks for the power profile channels. By giving the long pulse measurements the channels 1 and 2 which have fast analog-to-digital converters, one can sample the sky noise samples into their own buffers at the same time the E-layer channels receive in Barker coded mode (matched filter on). In this way it is possible to separate the Barker coded and noncoded parts of the experiment in two almost independent blocks and the channel division is 4 Barker coded and 4 noncoded channels. The E- and F-layer parts can be designed independently. Without this structure it is a little tricky to design effective experiments with both phase coded and noncoded elements.

When following the philosophy of almost independent E- and F-layer parts, one can design them to look almost as "subroutines". Thus it is easy to develope a large set of experiments having different combinations of the E-layer parts and F-layer parts.

The third advantage is that the structure allows E-layer modulation part designed in the way that the number of RFOFF gaps in the pattern is the smallest possible. This makes effective use of the transmitter duty cycle. The duty cycle balance between the E-layer, F-layer and remote station illumitations is also quite reasonable.

Two disadvantages can be mentioned, too. One necessarily has to do complicated balancing computations between the powerprofiles and pulse codes for obtaining the zero lag. The second disadvantage is that the number of frequencies used for target illumination in the ETX phase is usually 6. This may cause difficulties due to the plasmalines in certain physical conditions.

Both these disadvantages can be avoided by using "12-block" modulation pattern. In this solution the E-layer powerprofile group EPG is transmitted separately (EPTX) using the same channels as in the pulse coded group MPG in the next transmitting phase MPTX. The third transmitting phase FTX contains LPG and FPG. EPG and MPG use four channels, LPG and FPG the remaining four channels. The

"12-block" sequence is thus as follows.

1) transmit EPTX
2) receive E-layer powerprofiles
3) transmit MPTX
4) receive pulse coded data and calibrate LP channels
5) transmit FTX
6) receive LP and FPP data and calibrate pulse code and FPP channels

FPP channel calibration can be programmed also during the phase 4. EPG and MPG use the same channels and have thus a common calibration.

The phase 2 is very short, often less than 2000 us. Thus some clutter arises from the EPG to MPG receiving. This clutter is, however, much smaller than the "self clutter" of the pulse codes themselves, but it is strong enough to make the X-profile based channel balancing doubtful. The channel balancing is, however, not needed because the E-layer power profiles (EPG) and pulse codes (MPG) use the same channels and same frequencies. The power profiles transmitted shortly before the MPTX phase raise only the noise level a little, they do not contribute to the expectation values of higher order lags for which the pulse codes are intended.

In 12-block mode there are only 4 frequencies simultaneously illuminating the critical E-layer altitude and this is quite safe solution from the plasmaline contamination point of view if adjecent frequencies are used. The only practical plasmaline difficulty may exist in the first background gates in the long pulse channels because that data is taken when the pulse coded modulations illuminate the lower ionosphere.

The 12-block mode allows a simple form of plasmaline experiment to be programmed. One can receive using LP channels a powerprofile type of data when the EPG is illuminating the target (during the phase 2). These profiles (called plasmaline profiles in GEN-PROGRAMS library) may be useful in some plasmaline studies and they, in any case, can give warning for the possibility that some LP background gates contain plasmaline contamination.

There is one drawback compared with the 10-block mode. The transmitter duty cycle cannot be as effectively utilized in the 12-block mode as in the 10-block mode.

Finally it can be mentioned that at least in one case, when the MPG is formed by using 4-pulse codes only, one can write a special correlator program which decodes the 12-block mode algorithm and combines all the E-layer data into form of autocorrelation functions with zero lag included (library program GEN-8).

The example selected for the "CP-1 looking" GEN-SYSTEM experiment has the code name GEN-4. It is one of the simplest and at the same time quite effective general purpose E- and F-layer experiment when the E-layer part is not phase coded. It is written in "10-block mode".

The MPG part of the experiment is formed by combining two 4-pulse codes with system 2:1:4 and lag 1 delay of 40 us with two 3-pulse code with system 1:2 and lag1 delay of 60 us. The element pulses are 19 us long. Two power profiles using 19 us pulses form the EPG part. The sampling interval is 10 us and thus the 4-pulse code lag increment in the input data is 4 and the 3-pulse code lag increment is 6. A gating factor of one is used and thus the final lag increments in the output data are 2 and 3. The transmission pattern looks as follow:

The lags obtained after balancing and combining the channels are the following :

0, 40, 60, 80, 120, 160, 180, 200, 280 us

Thus this simple group produces 9 points of the ACF estimate. The points are not at equal intervals, which eliminates certain aliasing difficulties. CP-1 versions having modulation resembling the one shown above have given promising results.

The lag at 120 us delay is common to both codes and is thus statistically the best one. The channels can be balanced in the ways described earlier in this report.

The F-layer part uses the following pattern of 350 and 70 us pulses.

The pulses at F6 and F7 form also the remote station pattern.

The programming details and the exact parameters can be found in the files related to this experiment (Appendix 2). A "12-block mode" solution using the same pattern has the code name GEN-7 in the GEN-PROGRAMS library.

12. CALIBRATION PROGRAMS

System calibration programs have, to a certain extent, been a weak point in the EISCAT system. In the GEN-SYSTEM, the calibration program has the name GEN-CAL. It is a relatively simple program written in the following way. The antenna is pointed to the pole star, CYG-X and CAS-A. The antenna stays in each position for 5 minutes and samples the noise at a relatively high sampling frequency. In the pole star position, the doppler calibrator is turned on for part of the time. Measurements are done with both normal and very high attenuator setting. For every channel the correlator computes two ACF estimates: one for background and one for the noise injection. This is computed in the boxcar weighted mode using the last routine in the GEN-REMOTE correlator program.

If desired, one can also scan through the different polarizer settings at the remote sites.

This calibration program checks the following:

1) cabling and frequencies (DOPPLER calibration)
2) system noise and
3) noise injection (pointing cycle with pole, CYG-X)
4) linearity (CAS-A related saturation effects)
5) impulse response functions of the used channels
6) low frequency system noise and offsets

Different versions of the GEN-CAL are included in the GEN-PROGRAMS library. In critical applications, the use of a suitably modified version tailored to the actual measuring configuration is worth the time it takes.

13. COMPARISON BETWEEN THE GEN-SYSTEM AND THE OTHER METHODS

Comparison between the GEN-SYSTEM and the earlier methods is not very straightforward to make. At the time when this report was written, only very few tests had been done with transmitter and only preliminary analyses are available. The comparison is also difficult in a formal sense because the methods have different responses in the spatial domain. A rough analysis is, however, possible and gives order of magnitude estimates.

In this comparison, we omit the possible differences of measuring bandwidths and other similar parameters which are freely selectable in any algorithm. Then one can immediately conclude that the power profiles are the same except for the fact that more channels can be used in the GEN-SYSTEM. The gating done at correlator level produces data in which the gate separation and gate resolution have a reasonable relation. Thus the real time display is clearer even though there is not increase in the information. The power profiles in the example program GEN-5 have about 6 times shorter integration time than in the original CP-3 because of the greater number of transmitted pulses per unit time. Oversampling can also be used to decrease the integration time, sometimes by considerable amount. Thus in the "CP-1" looking programs, the E-layer power profiles have slightly shorter integration time than in similar older programs, but on the other hand the response is also different. In the GEN-SYSTEM one can easily have different power profiles in the same program which is a benefit in general purpose programs. This was difficult in earlier methods. The value of this feature depends on the importance of having different power profiles.

In the pulse coded part of the CP-1 looking GEN-PROGRAMS, the general improvemed is about a factor of 4 in integration time. This comes from programming twice as many channels as earlier and from oversampling by a factor of two or more. It is also possible to program much better calibration. The factor of 4 is quite considerable. In weak signal to noise conditions, it is equivalent to doubling the transmitter power. The GEN-SYSTEM gives also many more degrees of freedom in designing multichannel patterns, which make the algorithms more effective in special applications. The programming philosophy permits multiplexing easily the Barker coded and noncoded modulations. Although the philosphy is not unique to the GEN-SYSTEM, it is easiest to use with GEN-SYSTEM programs.

The monostatic long pulse has a different response and thus the comparison is not straightforward. The example program GEN-5 has about 3 times more illumination of target by monostatic long pulses than the original CP-3. At short delays, however, the integration time is not shorter by that amount but is about the same. On the other hand the spatial resolution is better and is almost constant with lag. (If the same spatial resolution were programmmed in the GEN-SYSTEM, for the zero lag, then the integration time improvement were about 3). At long delays the GEN-SYSTEM longpulse routine always wins over the earlier algorithms. At longest delays the integration time improvement is higher than the duty cycle benefit done. In the example program, the integration time at the longest programmed GEN-5 delay delay is more than 5 times shorter for a given accuracy of the lag estimate than in CP-3-C at corresponding delay. Tests with analysis will be needed to verify the final importance of the improved accuracy at long delays as well as the benefits of the spatial filtering properties of the GEN-SYSTEM long pulse routine. In the CP-1 looking experiment, the number of monostatic long pulses is 2 instead of one. The integration time at short delays is about the same as earlier and at long delays it is about 3 times shorter for a given accuracy. These results are also shown in Fig. 13.1. For a given filter bandwidth and target properties, the integration time is inversely proportional to the number of UNIPROG matrix elements times the square of the volume contributing to the expectation value of the element. This is valid during weak signal to noise conditions.

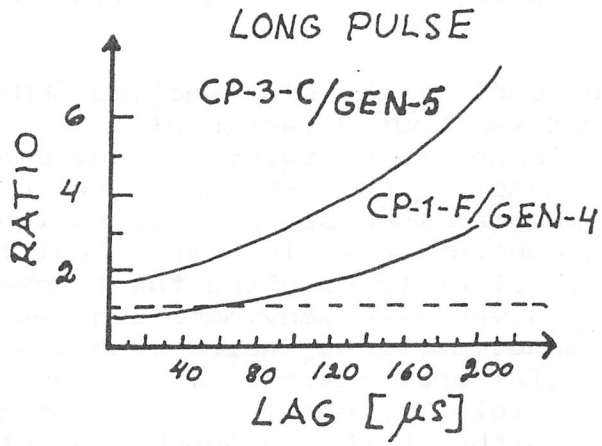


Fig. 13.1 A rough integration time comparison between GEN-4 with CP-1-F and GEN-5 with the CP-3-C for lags common in both algorithms. The same filter bandwidth is assumed. Note that because of differences in the spatial responses the exact integration time difference cannot be correctly done.

The remote station accuracy is about the same in GEN-SYSTEM programs as in the earlier common mode programs. The first few lags tend to have a slighly shorter integration time while the long lags are worse, especially in CP-3 resembling experiments. However the calibration accuracy is much better.

The calibration philosophy is much better in the GEN-SYSTEM than in most of the earlier programs. More than 1/3 of the input data is often taken for calibration purposes at the monostatic site and sometimes more than 95% of the remote station samples are only for calibration purposes. This together with algorithm improvements makes considerable increase in the calibration accuracy at all sites and especially at remote stations.

The GEN-SYSTEM allows also some practical and even economical benefits. Much more effective use of the transmitter and receiver system can be done without increasing the tape comsumption. Instead the monostatic data can be handled by a little smaller amount of tapes in some experiments (like GEN-5) and at remote stations the tape consumption is smaller by about a factor of 3-5 compared with the older algorithms.

14. GEN-PROGRAMS LIBRARY

During the development of the GEN-SYSTEM the following twelwe library programs were written to test the different features of the algorithms and the EISCAT hardware performace. All the test programs were designed by keeping the practical applications in mind. The library is certainly going to grow in the future. The basic idea is that the measuring algorithm is designed ready, tested in the system and documented. The user applications can be developed by programming suitable measuring geometry to the algorithm. Below a list of the currectly available programs is shown. The details are described in the description files of the library programs.

GEN-1

-a monostatic experiment intended for Es studies. Produces 125 gates of 300 meters resolution around the E-layer peak altitudes. Lagincrement is 28 us and the number of lags is 12. GEN-01 uses a specially designed correlator program for Barker coded 5-pulse code applications. Output format is a set of boxcar weighted autocorrelation functions including zero lag but lag 10 missing. Only powerprofile type calibration is included.

GEN-2

- a general purpose auroral experiment of "CP-1 type". Written in "12-block mode" using standard GEN-SYSTEM approach. Pulse coded part is formed by using 5-pulse codes. Low altitude part is sampled and gated to 4.5 km resolution. Lagincrement is 30 us. 350 us long pulses are sampled at 14 us intervals and computed using VOLUMEINDEX 15 and MAXLAG 15 in the present version.

GEN-3

-an algorithm where all the transmitter power is used for 5-pulse codes and powerprofiles to produce ready autocorrelation functions for the lower part of the ionosphere. Resolution is 4.5 km and the lagincrement is 30 us. The number of lags is 10. The algorithm is "self decoding and self balancing" one for which a specially designed correlator program is written. Also the calibrations appear in the form of autocorrelation functions.

GEN-4

-a "CP-1 looking" experiment written using "10-block" mode. See the Appendix 2.

GEN-5
-a "CP-3 looking" experiment. See the Appendix 1. The algorithm CP-3-D is based on GEN-5.

GEN-6
-a general purpose auroral experiment using Barker coded 4-pulse code for the E-layer and written using the standard GEN-SYSTEM approach. Low altitude part gated to 1.2 km resolution. Lagincrement is 56 us. Long pulses are 240 us long, sampled at 10 us intervals and computed using VOLUMEINDEX 5 and LAGMAX 17 (high resolution long pulse mode). The library version is written to "small scale" latitudinal scan mode.

GEN-7
-as GEN-4 but written into "12-block mode". Plasmaline profiles are included.

GEN-8
-a "CP-1 looking" experiment using 4-pulse codes and written in "12-block mode". A special correlator program compresses the E-layer data into a form of boxcar weighted autocorrelation functions with 7 lags (zero lag included). F-layer part as in GEN-4. The output contains only three data blocks: E-layer autocorrelation functions, F-layer autocorrelation functions and low resolution powerprofile with necessary calibrations.

GEN-9
-as GEN-8 but written using standard GEN-SYSTEM approach. Plasmaline profiles are included.

GEN-10
-development algorithm to study the programming difficulties in long range VHF algorithms. Final version intended for VHF applications. Not usable in UHF system.

GEN-11
-a very special pulse-to-pulse correlation program intended mainly for VHF applications

GEN-12
-as GEN-6 but gated for 600 meters resolution in the low altitude part. The library version contains three direction scanning with possibility to have real time mode control. Intended for sporadic E-layer, gravity wave and general purpose auroral research when high resolution is of importance.

The listed programs have been developed but are not necessarily in the very final form (August 1985 situation). Tiny modifications are expected during testings with transmitter to some of them. A report describing the available library is going to be published later.

15. SUMMARY

The GEN-SYSTEM has been developed for the EISCAT radar system. Its principles could also be utilized by other radars working against distributed targets. In the EISCAT system, it gives the user the possibility to design both general purpose and special purpose programs in such a way that the hardware system is fully used and that the information flow from the target is optimized for the modulations which can be handled. The GEN-SYSTEM is not the absolutely optimum way to do multichannel incoherent scatter measurements but it is nearly optimum for the EISCAT system in the originally designed hardware conficuration. Some potentially more powerful methods are beyond the capabilities of the EISCAT real-time processing system (correlator). Although the GEN-SYSTEM is originally written mainly for UHF applications, it lends itself to effective VHF algorithms, too, because the data compression permits long range two beam experiments to fit within the available correlator result memory. More VHF programs will soon be included in the GEN-PROGRAMS library.

The GEN-SYSTEM is not very simple to use but an experienced EISCAT user can learn it in relatively short time . It requires a quite good general understanding of the EISCAT system. The correlator programming requires a familiarity with the syntax of CORLAN. One has to remember, also, that the data structure is not compatible with the earlier formats and thus modifications to the tape handling and analysing routines are needed. The work which has to be done before being able to develop new programs with the GEN-SYSTEM pays itself back in shorter integration times and better spatial response. In many applications the user can make use of the existing library programs.

REFERENCES

Baron, M. (1984) The EISCAT facility. J. Atmos. Terr. Phys. 46, 469-472

Ho T., and H.-J. Alker (1981) Scientific programming of the EISCAT digital correlator (revised). EISCAT technical note 81/24, EISCAT Scientific Association, S-981 27 Kiruna, Sweden

Ho. T.(1981) Standard subroutines and programs for EISCAT digital correlator. EISCAT technical note 81/27. EISCAT Scientific Association, S-981 27 Kiruna, Sweden

Ho T, T. Turunen, J. Silen and M. Lehtinen (1983) The lag profile routine and the Universal Program for the EISCAT digital correlators. EISCAT technical note 83/37. EISCAT Scientific Association, S-981 27 Kiruna, Sweden

Johansson, K.-O., K. Persson, W. Schmidt , and A.L. Turunen (1984), EISCAT Experiment Preparation Manual. Eiscat Technical Note 84/41, EISCAT Scientific Association, S-981 27 Kiruna, Sweden

Turunen, T. (1983) The Universal Correlator Program - a versatile tool for calculating the target autocorrelation function in incoherent scatter radar measurements. EISCAT meetings 83/8, EISCAT Scientific Association, S-981 27 Kiruna, Sweden

Turunen, T. and J. Silen (1984) Modulation patterns for the EISCAT incoherent scatter radar. J. Atmos. Terr. Phys. 46. 593-600

Törustad, B. W. (1982) CORLAN (CORrelator LANguage). EISCAT technical note 82/36. EISCAT Scientific Association, S-981 27 Kiruna, Sweden

ACKNOWLEDGEMENT

The author is very greatful to the Director of EISCAT, Dr.Murray Baron, for his great interest in this report. He followed the work carefully and did valuable comments.

I also thank Dr. Walter Schmidt for fruitful discussions and Dr. Peter Collis for reading and commenting the manuscript.

EISCAT is supported by: Centre National de la Recherche Scientifique (CNRS), France; Suomen Akatemia (SA), Finland; Max-Planck-Gesellschaft (MPG), West Germany; Norges Almenvitenskapelige Forskningsråd (NAVF), Norway; Naturvetenskapliga Forskningsrådet (NFR), Sweden; Science and Engineering Research Council (SERC), United Kingdom.

APPENDIXES

Appendix 1: GEN-5

Appendix 2: GEN-4

GEN-4 and GEN-5 are two example programs from the program library GEN-PROGRAMS. The remote station correlator program of GEN-4 is not included because GEN-5 shows the way how to use the master program GEN-REMOTE and the only differences are seen in the file GEN-4-R-SUBR:LIST. Some GEN-5 files are in the form used in CP-3-D (differs only by name).

A1/1

APPENDIX 1

THE EXPERIMENT GEN-5 (USED ALSO AS CP-3-D)

ALL FILES

54 PAGES

VERSION MARCH 1987

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%       GEN-5:DESC
%
%       CP-3 LOOKING EXPERIMENT
%
%       TAUNO TURUNEN
%       EISCAT HQ, MARCH 1985
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

THE GEOMETRICAL PROPERTIES OF THIS EXPERIMENT HAVE BEEN
TAKEN FROM THE CP-3-C

MODULATION

REPETITION PERIOD IS 17256 US. THE PERIOD CONTAINS TWO TRASMISSION PHASES, BOTH OF WHICH HAVE 2 LONG PULSES OF 350 US DURATION AND 4 POWER PROFILE PULSES OF 70 US DURATION.

IN THE FIRST TRANSMISSION PHASE THE LONG PULSES ARE AT FREQUENCIES F0 AND F1.
IN THE SECOND TRANSMISSION PHASE THE LONG PULSES ARE AT FREQUENCIES F2 AND F3.
THE POWER PROFILES ARE ALWAYS AT FREQUENCIES F4,F5,F6 AND F7.
THE LONG PULSES FORM ALSO THE REMOTE STATION PART OF THE MODULATION.
THE LONG PULSES ARE CALIBRATED JUST BEFORE THE CORRESPONDING FREQUENCIES ARE TRANSMITTED. THE POWER PROFILES ARE CALIBRATED AT THE END OF BOTH "SUBCYCLES".

THE LONG PULSES ARE RECEIVED USING CHANNELS 1,2,3 AND 4.
THE POWER PROFILES USE CHANNELS 5,6,7 AND 8.

FOR DETAILS THE USER IS ADVISED TO STUDY THE CORRESPONDING ELAN AND TLAN FILES.

PHASE 1

PHASE 2

THE CORRELATOR PROGRAM GEN-5-T:CCOD IS WRITTEN USING THE PROGRAM GEN-F-L1U1P2 AS A MASTER PROGRAM. THE NEEDED MAIN PROGRAM IS IN THE FILE GEN-5-T-SUBR:LIST. THE REMOTE STATIONS USE CORRELATOR PROGRAM GEN-5-R:CCOD BASING ON GEN-REMOTE.

THE MONOSTATIC PART USES A NEW "CONSTANT VOLUME SIZE" ALGORITHM IN THE LONG PULSES (GEN-SYSTEM LONG PULSE ROUTINE).

A1/2

ALL CHANNELS PRODUCING IDENTICAL DATA ARE ADDED TOGETHER ALREADY AT THE CORRELATOR LEVEL BOTH IN THE MONOSTATIC AND REMOTE RECEIVING. THUS IN THE MONOSTATIC RECEIVING THE CHANNELS 1,2,3 AND 4 MUST HAVE AS EQUAL GAIN AS POSSIBLE (LON PULSES) AND 5,6,7 AND 8 MUST HAVE AS EQUAL GAIN AS POSSIBLE (POWERPROFILES). AT THE REMOTE SITES THE CHANNELS 3,4,5 AND 6 ARE USED FOR LONG PULSE RECEIVING AND MUST HAVE AS EQUAL GAIN AS POSSIBLE.

IF THE GAIN BALANCE IS NOT GOOD, THE DATA IS NOT LOST BUT THE STATISTICAL ACCURACY IS NOT THE BEST POSSIBLE ONE AND THE VARIANCE COMPUTATIONS LEAD TO TOO SMALL ESTIMATE.

THE CHANNELS CAN BE BALANCED TO WITHIN +-0.5 DB, WHICH IS GOOD ENOUGH AND THE ASSUMPTION THAT THE CHANNELS HAVE THE SAME GAIN WORKS REASONABLY WELL IN THE VARIANCE ESTIMATE.

THE GAIN BALANCE CAN BE TESTED BY RUNNING A SUITABLE VERSION OF GEN-CAL AT THE SITE BEFORE THE EXPERIMENT. WITH THIS CALIBRATION EXPERIMENT ONE ALSO PRODUCES DATA TO ESTIMATE THE SYSTEM TRANSFER FUNCTIONS VERY ACCURATELY FOR EVERY INDIVIDUAL CHANNEL.

MONOSTATIC PART USES 25 KHZ LINEAR PHASE FILTERS, WHICH IS A FILTER HAVING MAXIMALLY WELL BEHAVING TIME DOMAIN RESPONSE. AT THE REMOTE SITES THE FILTERS ARE 25 KHZ BUTTERWORTH FILTERS.

GENERAL

A) POWER PROFILE PART STARTS FROM ADDRESS 0000 IN THE RESULT MEMORY OF THE CORRELATOR AND HAS 30+10+2 GATES (SCATTER, SKY, NOISE INJECTION)

   THE FIRST POWER PROFILE GATE IS CENTERED AT A NOMINAL RANGE OF 76.5 KM AND THE GATE SEPARATION IS 10.5 KM,TOTAL RANGE COVERAGE IS 76.5-381.0 KM. P-P RESOLUTION IS 21.0 KM, 75% CONTRIBUTION RESOLUTION IS 10.5 KM, AND THE GATE CORRELATION IS 25% (DEFINES THE POWERPROFILE GATE OVERLAPPING). THESE PROPERTIES ARISE FROM 70 US PULSE, 14 US SAMPLING AND GATING FACTOR 4. THUS 5 NEIGHBORING POINTS ARE SUMMED TOGETHER CAUSING ALMOST EXACTLY TRIANGULAR SPATIAL WEIGHTING FUNCTION.

B) ACF:S START FROM FROM THE CORRELATOR ADDRESS 42 HAVING 25+7+2 GATES WITH 16 LAGS IN THE ORDER: SCATTER,SKY,NOISE INJECTION. LAG INCREMENT IS 14 US.

   IN THE LONG PULSE (350 US) DATA THE FIRST ACF GATE IS CENTERED AT NOMINAL RANGE OF 200 KM, THE GATE SEPARATION IS 31.5 KM AND THE RANGE COVERAGE IS 200-956 KM. THE RESOLUTION OF THE VOLUME IS 84.0 KM MEASURED BETWEEN THE ABSOLUTE BOUNDARIES OF THE VOLUME. THIS IS THE SAME AT EVERY LAG. MORE MEANINGFUL FIGURE FOR THE RESOLUTION IS OBTAINED BY CONSIDERING A VOLUME FROM WHICH ABOUT 70 % OF THE CONTRIBUTION TO THE TARGET ACF ESTIMATE ARISES. THE SMALL TABLE BELOW GIVES THE RESOLUTION AT SOME LAGS.

A1/3

CONTRIBUTION OF THE 42 KM VOLUME TO THE ACF ESTIMAT AT DIFFERNT LAGS:

LAG0: 73%
LAG5: 75%
LAG10: 73%
LAG15: 67%

CORRELATION BETWEEN THE GATES ARISES FROM THE FACT THAT THE NEIGHBORING GATES HAVE CERTAIN AMOUNT OF CONTRIBUTION FROM THE SAME PART OF THE TARGET. THIS CAN BE USED AS AN ESTIMATE OF THE "OVERLAPPING". NOTE THAT OVERLAPPING IS DEFINED HERE AS A TRUE TARGET VOLUME OVERLAPPING GIVEN IN PERCENT OF CONTRIBUTION FROM A COMMON VOLUME. IN EARLIER CP-3 VERSIONS THE OVERLAPPING HAS DIFFERENT MEANING (SAMPLE SPACE OVERLAPPING USING GATING IN THE SAMPLE SPACE).

CORRELATION BETWEEN THE NEIGHBORING GATES:

LAG0: 42%
LAG5: 39%
LAG10: 42%
LAG15: 50%

LONG PULSE DATA PROPERTIES

AFTER SUBTRACTING THE SKY NOISE ONE HAS TO DIVIDE THE OBTAINED TARGET ACF ESTIMATE BY WEIGHTING FACTOR W(I), WHICH CAN BE OBTAINED FROM THE FORMULA

W(I) =(15+I)*(25-I)/(15*25) ; WHERE
I=0,1,...,MAXLAG (=15) IS THE LAG INDEX
15 = VOLUMEINDEX
25 = 350/14 =PULSELENGHT/SAMPLING INTERVAL

IN THE GIVEN FORM OF W(I) THE VALUE AT ZERO LAG IS 1.000.
THE TABLE BELOW GIVES THE NORMALIZED WEIGHTING FACTORS FOR THE MEASURED LAGS.

LAG0 :1.000	LAG6 :1.064	LAG12:0.936
LAG1 :1.024	LAG7 :1.056	LAG13:0.896
LAG2 :1.043	LAG8 :1.043	LAG14:0.851
LAG3 :1.056	LAG9 :1.024	LAG15:0.800
LAG4 :1.064	LAG10:1.000	
LAG5 :1.067	LAG11:0.975	

NOTE THAT THIS TABLE IS VALID ONLY FOR THE TARGET CONTRIBUTION PART OF THE ACF ESTIMATE (SKY NOISE SUBTRACTED). THE RAW TARGET ACF ESTIMATE AFTER THE SUBTRACTION MUST BE DIVIDED BY THE GIVEN NUMBERS IN ORDER TO OBTAIN THE BOXCAR WEIGHTING.

A1/4

WHEN THE SYSTEM TRANSFER FUNCTION IS ESTIMATED BY TAKING THE FOURIER TRANSFORM OF THE SKY NOISE ACF (OR NOISE INJECTION ACF- SKY NOISE ACF), THEN THE WEIGHTING FACTOR WITH WHICH A BOXCAR WEIGHTED ACF IS OBTAINED IS

NOISEWEIGHT(I) = (15+I)/15

LONG PULSE STATISTICS

THE NUMBER OF CROSSPRODUCTS PER VOLUME AT A GIVEN LAG I IN A SINGLE DUMP IS

NOPRODUCTS(I) = (SCANCOUNT*4-3)*(15+I) ; I IS THE LAGINDEX
4 IS THE NUMBER OF CHANNELS ADDED TOGETHER

NOTE THAT THE LAST FORMULA IS NOT MEANINGFUL IF THE CHANNELS 1,2,3 AND 4 DO NOT HAVE REASONABLE GAIN BALANCE.

POWERPROFILE STATISTICS

THE NUMBER OF CROSSPRODUCTS PER GATE IN THE POWERPROFILES IS

NOPRODUCTS(POWERPROFILE) = (SCANCOUNT*2*4-7)*5 ; WHERE

2 NUMBER OF CYCLES /REP
4 IS THE NUMBER OF ADDED CHANNELS
5 COMES FROM THE GATING FACTOR

*** REMOTE STATION DATA***

REMOTE STATIONS ADD FOUR CHANNELS TOGETHER. THE DATA STRUCTURE CONSIST OF FIVE ACF:S WITH 21 LAGS WITH TRIANGLULAR WEIGHTING AND 3 ACF:S WITH 21 LAGS HAVING BOXCAR WEIGHTING.

THE SIGNAL SHOULD BE IN THE 3RD AFC (TRIANGLULAR WEIGHTING).

THE FIRST FIVE ACF:S HAVE BEEN COMPUTED FROM THE DATA VECTOR CONTAINING 100 SAMPLES. GATING IS 30 SAMPLES AND OVERLAPPING 10. THE SIGNAL IS FULLY ILLUMINATING THE VOLUME IN THE 30 SAMPLES IN THE MIDDLE OF THE DATA VECTOR. SIGNAL LEKAGE SHOULD BE CLEARLY SEEN IN THE ACF:S 2 AND 4. ACF:S 1 AND 5 SHOULD BE SIGNAL FREE. BECAUSE OF OVERLAPPING COMPUTATION THE SIGNAL LEAKAGE SHOULD BE QUITE STRONG (ABOUT 30%) IN ACF:S 2 AND 4 AND ALMOST SYMMETRIC IF THE TIMING IS CORRECT.

THE PROPERTIES OF THE SIGNAL ACF ARE THE FOLLOWING:

UNNORMALIZED WEIGHTING FACTOR RSIGWEIGHT(I)

RSIGWEIGHT(I) = 30-I ; I = 0,1,...,20

THE NUMBER OF CROSSPRODUCTS AT A GIVEN LAG I IN ONE DUMP IS

NOPRODUCTS=(4*SCANCOUNT-3)*(30-I)

SKY NOISE IS IN THE ACF:S 6 AND 7 AND NOISE INJECTION IN 8.
THESE ACF:S HAVE BOXCAR WEIGHTING AND THE WEIGHTING FACTOR IS 260
AT EVERY LAG!!!!!
AS A CONSEQUENCE ONE CANNOT SUBTRACT THE SKY NOISE DIRECTLY FROM THE
SIGNAL. IN REAL TIME MONITORING ONE HAS TO USE "BO" STATEMENT OF RTGRAPH.
ONE HAS TO ANSWER "SIGSAMPLES =30" AND "PRODUCTS =260"

WHEN THE TWO SKY NOISE ACF:S ARE ADDED TOGETHER, THE NUMBER OF CROSSPRODUCTS
ADDED TOGETHER AT EVERY LAG IS

CALNOPRODUCTS = (SCANCOUNT*4*4-14)*130

THIS NUMBER IS MORE THAN 1.100.000 AT EVERY LAG IN ONE 10 SECOND DUMP.
THIS GIVES GOOD ACCURACY FOR THE SKY NOISE ESTIMATE.

THE CORRESPONDING NUMBER OF THE CROSSPRODUCTS IN THE NOISE INJECTION ACF IS

CALPRODUCTS(NOISE)=(SCANCOUNT*4*2-7)*130

AS A CONSEQUENCE THE NUMBER BY WHICH THE SKY NOISE ESTIMATE (BOTH TWO ADDED)
MUST BE MULTIPLIED BEFORE DOING THE SUBTRACTION FROM THE SIGNAL GATE IS THE
FOLLOWING:

SKYFACTOR = ((SCANCOUNT*4-3)*(30-I))/((SCANCOUNT*4*4-14)*130)

****

GEOMETRICAL DETAILS ARE IDENTICAL TO THOSE IN CP-3-C

THE COMMON PROGRAM THREE EXPERIMENT IS DESIGNED TO PROVIDE BROAD
LATITUDINAL COVERAGE OF THE AURORAL IONOSPHERE.

THE TROMSO ANTENNA SCANS IN 15 DISCRETE STEPS ALONG (APPROXIMATELY)
A MERIDIAN. THE PLANE OF THE SCAN PASSES DIRECTLY OVERHEAD OF BOTH
TROMSO AND KIRUNA. THE KIRUNA AND SODANKYLA ANTENNAS TRACK THIS
SCAN, POINTING AT AN INTERSECTION VOLUME AT 325 KM ALTITUDE.

A1/6

ANTENNA POINTING

POSITION	TIME (SEC)	TROMSO AZ	TROMSO EL	KIRUNA AZ	KIRUNA EL	SODANKLYLA AZ	SODANKLYLA EL	LAT	LONG
				HEIGHT = 325 KM					
1	100	344.9	28.0	346.0	19.9	335.4	15.9	74.2	14.6
MOVE	20			822 KM		957 KM		(RANGE)	
2	90	344.9	35.1	346.0	24.3	332.9	19.0	73.2	15.8
MOVE	20			712 KM		849 KM			
3	90	344.9	43.7	346.0	29.3	329.9	22.4	72.3	16.8
MOVE	20			619 KM		756 KM			
4	90	344.9	54.0	346.0	35.1	326.5	26.0	71.5	17.6
MOVE	20			539 KM		678 KM			
5	90	344.9	65.4	346.0	41.5	322.6	29.6	70.8	18.2
MOVE	20			476 KM		614 KM			
6	80	344.9	77.5	346.0	48.7	318.0	33.3	70.2	18.7
MOVE	20			424 KM		562 KM			
7	120	MOVING	90.0	346.0	57.1	312.5	37.2	69.6	19.2
MOVE	20			382 KM		517 KM			
8	80	164.9	77.9	346.0	66.8	305.4	41.2	69.0	19.6
MOVE	20			352 KM		479 KM			
9	70	164.9	66.7	346.0	78.0	296.3	45.1	68.4	20.0
MOVE	30			332 KM		448 KM			
10	120	164.9	57.1	MOVING	90.0	284.7	48.4	67.9	20.4
MOVE	30			325 KM		426 KM			
11	70	164.9	49.0	166.0	77.9	270.7	50.4	67.3	20.8
MOVE	20			332 KM		415 KM			
12	80	164.9	42.1	166.0	66.4	255.0	50.6	66.7	21.2
MOVE	20			353 KM		414 KM			
13	90	164.9	35.6	166.0	54.8	237.8	48.2	66.0	21.6
MOVE	20			393 KM		428 KM			
14	90	164.9	29.7	166.0	44.3	222.3	43.1	65.2	22.0
MOVE	20			454 KM		463 KM			
15	90	164.9	24.6	166.0	35.5	210.5	36.8	64.2	22.5
RECYCLE	150			534 KM		520 KM			

CYCLE = 1800 SEC = 30 MIN
OF WHICH: 1350 SEC (75%) IS DATA, 450 SEC IS ANTENNA MOTION

THE POSITIONS AND SYNC TIMES HAVE BEEN SET SUCH THAT ALL ANTENNAS SHOULD BE IN POSITION FOR THE PLANNED INTERVALS; ALL MOTIONS SHOULD BE COMPLETED IN THE ALLOWED TIMES.

A1/7

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  ***  E R O S   P R O G R A M M E   F O R   T R O M S O   ***
%%
%%         EXPERIMENT GEN-5,FILE GEN-5-T:ELAN
%%
%%         A CP-3 LOOKING PROGRAM WRITTEN BY TAUNO TURUNEN
%%
%%         EISCAT HQ, JAN.1985
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
WRI-EXP-HEAD ,,
%
WRITE-FILE (EXP)GEN-5-T:ELAN
WRITE-FILE (EXP)GEN-5-T:TLAN
WRITE-FILE (EXP)GEN-5:DESC
WRITE-FILE (EXP)GEN-5-T-SUBR:LIST
%
POINT-REF-HE 344.9,28.0,325.0
%
SYNC -120
%
LOAD-RADAR (EXP)GEN-5-T
%
TRANSF-NOISE-CONT RAD
%
DEF-FLUKE-POS 3,Y
DEF-FLUKE-POS 4,Y
%
SET-LO1 1053.5
SET-SIG-PAT ALLX
SET-SIG-ATT X,5
%
SET-CH-ATT 1,4
SET-CH-ATT 2,4
SET-CH-ATT 3,4
SET-CH-ATT 4,4
SET-CH-ATT 5,4
SET-CH-ATT 6,4
SET-CH-ATT 7,4
SET-CH-ATT 8,4
%              NOTE:BALANCE THE CHANNELS FOR GROUPS 1,2,3,4 AND 5,6,7,8
%                   BY CHECKING THE ATTENUATOR VALUES BEFORE THE EXPERIMENT.
SET-LO2 1,94.0
SET-LO2 2,93.5
SET-LO2 3,93.0
SET-LO2 4,92.5
SET-LO2 5,92.0
SET-LO2 6,91.5
SET-LO2 7,91.0
SET-LO2 8,90.5
%
```

A1/8

```
SET-FILTER 1,LIN,25.0
SET-FILTER 2,LIN,25.0
SET-FILTER 3,LIN,25.0
SET-FILTER 4,LIN,25.0
SET-FILTER 5,LIN,25.0
SET-FILTER 6,LIN,25.0
SET-FILTER 7,LIN,25.0
SET-FILTER 8,LIN,25.0
%
%
LOAD-CORR (EXP)GEN-5-T:CCOD      %MASTER PROGRAM GEN-F-L1U1P2:CLAN
%
SET-CO-APB 15,15        %VOLUMEINDEX
SET-CO-APB 14,15        %LAGMAX
SET-CO-APB 13,25        %LPNOGATES, SCATTER PART
SET-CO-APB  9,210       %PPNOSAMPLES1 (GATING+1=5)
SET-CO-APB  7,7         %USERREG1,LP SKY NOISE GATE NUMBER
SET-CO-APB  6,2         %USERREG2,LP NOISE INJECTION GATE NUMBER
SET-CO-APB  5,4         %USERREG3, PP GATING CONTROL (5-1)
SET-CO-APB  0,1         %APBINCR
%
%
SET-CO-APM 15,0         %RESENTRY1, POWERPROFILES (30+10+2;DATA,SKY,NOISE INJ
.)
SET-CO-APM 14,42        %RESENTRY2, LP SCATTER GATES
SET-CO-APM 13,442       %RESENTRY3, LP SKY NOISE
SET-CO-APM 12,554       %RESENTRY4, LP NOISE INJECTION
SET-CO-APM 11,586       %RESENTRY5, SCANCOUNTADDRESS
%
SET-CO-APM  3,15        %LPMAXLAG
%
SET-CO-APM  0,1         %APMINCR
%
%
SET-CO-DATAIO 587
%
%          INFORMATION FOR DISPLAY PROGRAMS:
%          CORRELATOR DUMP CONSISTS OF TWO PARTS
%          A) POWER PROFILE PART STARTING FROM ADDRESS 0000 IN
%             THE CORRELATOR HAVING 30+10+2 GATES
%          B) ACF:S STARTING FROM ADDRESS 42 HAVING 25+7+2 GATES WITH
%             16 LAGS IN ORDER SCATTER,SKY,NOISE INJECTION
%
%          THE FIRST POWER PROFILE GATE IS CENTERED AT RANGE 76.5 KM
%          AND THE GATE SEPARATION IS 10.5 KM , TOTAL COVERAGE 76.5-381.0 KM
.
%          P-P RESOLUTION 21.0 KM, 75% RESOLUTION 10.5 KM, GATE
%          CORRELATION 25%
%
```

A1/9

```
%           IN LONG PULSE DATA THE FIRST ACF GATE IS CENTERED AT RANGE 200.0
KM.
%           GATE SEPARATION IS 31.5 KM. TOTAL RANGE 200.0-956.0 KM.
%           P-P RESOLUTION IS 84.0 KM. WEIGHTED RESOLUTION IS THE
%           FOLLOWING:
%           LAG0: 42 KM/73%
%           LAG5: 42 KM/75%
%           LAG10:42 KM/73%
%           LAG15:42 KM/67%
%           THIS DEFINES RESOLUTION
%
%           CORRELATION BETWEEN THE NEIGHBORING GATES:
%           LAG0:42%, LAG5:39%, LAG10:42%, LAG15:50%
%           THIS DEFINES OVERLAPPING
%
%           FOR DETAILS SEE THE DESCRIPTION FILE
%
%
SYNC 60
%
SET-BUF-M 2,1020
SET-BUF-M 3,2040
SET-BUF-M 4,3060
SET-BUF-M 5,600
SET-BUF-M 6,1620
SET-BUF-M 7,2640
SET-BUF-M 8,3660
SET-BUF-M 1,0000            % MUST BE THE LAST COMMAND!!!!!!!
%                           % THE FIRST UNUSED BUFFER ADDRESS 4080
%
SET-ADC-INT 1,14
SET-ADC-INT 2,14
SET-ADC-INT 3,14
SET-ADC-INT 4,14
SET-ADC-INT 5,14
SET-ADC-INT 6,14
SET-ADC-INT 7,14
SET-ADC-INT 8,14
%
%
START-CORR
SYNC 50
START-RADAR 10,1,10
%
ENABLE-DMA
%
%
SYNC 11
DO -1
START-REC-DATA
SYNC 100              % T1          100    SEC
POINT-REF-H 344.9,35.1,325.0  % T2     20+ 90    SEC
SYNC 110
POINT-REF-H 344.9,43.7,325.0  % T3     20+ 90    SEC
```

A1/10

```
SYNC 110
POINT-REF-H 344.9,54.0,325.0  % T4      20+ 90   SEC
SYNC 110
POINT-REF-H 344.9,65.4,325.0  % T5      20+ 90   SEC
SYNC 110
POINT-REF-H 344.9,77.5,325.0  % T6      20+ 80   SEC
SYNC 100
POINT-REF-H 12.8,89.99,325.0  % T7
SYNC 20            %          20+120   SEC
POINT-REF-H 155.0,89.99,325.0  % T8
SYNC 120
POINT-REF-H 164.9,77.9,325.0  % T9      20+ 80  SEC
SYNC 100
POINT-REF-H 164.9,66.7,325.0  % T10     20+ 70   SEC
SYNC 90
POINT-REF-HE 164.9,57.1,325.0 % T11     30+120   SEC    KIRUNA K10&K11
SYNC 150
POINT-REF-H 164.9,49.0,325.0  % T12     30+ 70   SEC
SYNC 100
POINT-REF-H 164.9,42.1,325.0  % T13     20+ 80   SEC
SYNC 100
POINT-REF-H 164.9,35.6,325.0  % T14     20+ 90   SEC
SYNC 110
POINT-REF-H 164.9,29.7,325.0  % T15     20+ 90   SEC
SYNC 110
POINT-REF-H 164.9,24.6,325.0  % T16     20+ 90   SEC
SYNC 110
STOP-REC-DATA
POINT-REF-H 344.9,28.0,325.0  % T1        150   SEC
SYNC 150
ENDDO
RETURN
```

A1/11

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%       GEN-5-T:TLAN
%
%       TAUNO TURUNEN,EISCAT HQ, JAN.1985
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
AT 1 TRANS,CHOPULS
AT 10 RXPROT,LOPROT
AT 50 BEAMON
AT 80 F0
AT 100 RFON
AT 450 RFOFF,F1
AT 452 RFON
AT 802 RFOFF,F4
AT 804 RFON
AT 874 RFOFF,F5
AT 876 RFON
AT 946 RFOFF,F6
AT 948 RFON
AT 1018 RFOFF,F7
AT 1020 RFON
AT 1090 RFOFF,BEAMOFF
AT 1150 RXPOFF
AT 1200 LOPOFF
AT 1300 RECEV
%
%
AT 1310 CH1
AT 1325 CH5
AT 1396 CH6
AT 1468 CH7
AT 1540 CH8
AT 1662 CH2     % FIRST LP-VOLUME AT 200.0 KM RANGE
%               % VOLUME SEPARATION 31.5 KM
%               % 75% RESOLUTION 41 KM (ABOUT)
%               % FIRST POWERPROFILE GATE AT 76.5 KM
%               % RESOLUTION 21 KM P-P, 10.5 KM AT 75% CONTRIBUTION
%               % POWERPROFILE GATE SEPARATION 10.5  KM
%
%
AT 3414 CH5OFF
AT 3486 CH6OFF
AT 3558 CH7OFF
AT 3630 CH8OFF
%               % 150 SCATTER POINTS IN POWERPROFILE CHANNELS
%
```

A1/12

```
AT 5457 CH3,CH4
AT 6647 CH5,CH6,CH7,CH8
AT 6970 CH1OFF
AT 7322 CH2OFF
AT 7337 ALLOFF,CAL100                    %135 SKY SAMPLES,L-P
%                                        %405 LP-DATA SAMPLES,L-P
%                                        %50 SKY, POWERPROFILES
AT 7400 CH3,CH4,CH5,CH6,CH7,CH8
AT 7533 CH5OFF,CH6OFF,CH7OFF,CH8OFF      %10 NOISE INJECTION SAMPLES
AT 8230 ALLOFF,CAL0                      %60 NOISE INJECTION SAMPLES
%
SETTCR 8500
%
AT 1 TRANS,CHQPULS
AT 10 RXPROT,LOPROT
AT 50 BEAMON
AT 80 F2
AT 100 RFON
AT 450 RFOFF,F3
AT 452 RFON
AT 802 RFOFF,F4
AT 804 RFON
AT 874 RFOFF,F5
AT 876 RFON
AT 946 RFOFF,F6
AT 948 RFON
AT 1018 RFOFF,F7
AT 1020 RFON
AT 1090 RFOFF,BEAMOFF
AT 1150 RXPOFF
AT 1200 LOPOFF
AT 1300 RECEV
%
%
%GEN-5   RECEIVING PART
AT 1310 CH3
AT 1325 CH5
AT 1396 CH6
AT 1468 CH7
AT 1540 CH8
AT 1662 CH4     % FIRST LP-VOLUME AT 200.0 KM  RANGE
%               % VOLUME SEPARATION 31.5 KM
%               % 75% RESOLUTION 41 KM (ABOUT)
%               % FIRST POWERPROFILE GATE AT 76.5 KM
%               % RESOLUTION 21 KM P-P, 10.5 KM AT 75% CONTRIBUTION
%               % POWERPROFILE GATE SEPARATION 10.5  KM
%
```

A1/13

```
%
AT 3414 CH5OFF
AT 3486 CH6OFF
AT 3558 CH7OFF
AT 3630 CH8OFF
%              % 150 SCATTER POINTS IN POWERPROFILE CHANNELS
%
AT 5457 CH1,CH2
AT 6647 CH5,CH6,CH7,CH8
AT 6970 CH3OFF
AT 7322 CH4OFF
AT 7337 ALLOFF,CAL100                    %135 SKY SAMPLES,L-P
%                                        %405 LP-DATA SAMPLES,L-P
%                                        %50 SKY, POWERPROFILES
AT 7400 CH1,CH2,CH5,CH6,CH7,CH8
AT 7533 CH5OFF,CH6OFF,CH7OFF,CH8OFF      %10 NOISE INJECTION SAMPLES
AT 8230 ALLOFF,CAL0                      %60 NOISE INJECTION SAMPLES
%
%
AT 8400 CH8
AT 8410 STC
AT 8420 ALLOFF
AT 8460 BUFLIP
%
SETTCR 0000
%
AT 17256 REP
```

A1/14

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%          GEN-5-T-SUBR:LIST
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NEXT
CALL SUBRLIST1

NEXT
CALL SUBRLIST1

NEXT
CALL SUBRLIST2

NEXT
CALL SUBRLIST2

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY5)
GOTO SCANCOUNT

NEXT
SUBROUTINE SUBRLIST1
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
RELOAD LCR3
RELOADVALUE=RSAPB(LPNOGATES)
CALL LONGPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY3)
RELOAD LCR3
RELOADVALUE=RSAPB(USERREG1)
CALL LONGPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY4)
RELOAD LCR3
RELOADVALUE=RSAPB(USERREG2)
CALL LONGPULSE


NEXT
RSAPM(RESTARTADD)=0
RELOAD LCR1
RELOADVALUE=RSAPB(USERREG3)
CALL POWERPROFILE1

NEXT
RSAPM(RESTARTADD)=0
RELOAD LCR1
RELOADVALUE=RSAPB(USERREG3)
CALL POWERPROFILE1
```

A1/15

```
NEXT
RETURN

NEXT
SUBROUTINE SUBRLIST2
RSAPM(RESTARTADD)=RSAPM(RESENTRY3)
RELOAD LCR3
RELOADVALUE=RSAPB(USERREG1)
CALL LONGPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY4)
RELOAD LCR3
RELOADVALUE=RSAPB(USERREG2)
CALL LONGPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
RELOAD LCR3
RELOADVALUE=RSAPB(LPNOGATES)
CALL LONGPULSE

NEXT
RSAPM(RESTARTADD)=0
RELOAD LCR1
RELOADVALUE=RSAPB(USERREG3)
CALL POWERPROFILE1

NEXT
RSAPM(RESTARTADD)=0
RELOAD LCR1
RELOADVALUE=RSAPB(USERREG3)
CALL POWERPROFILE1

NEXT
RETURN
```

A1/16

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%          CORRELATOR PROGRAM GEN-F-L1U1P2:CLAN                                %
%         ************************************                                 %
%                                                                              %
%     THIS PROGRAM CONTAINS THE FOLLOWING SUBROUTINES:                         %
%                                                                              %
%                     LONGPULSE                                                %
%                     UNIPROG                                                  %
%                     POWERPROFILE1                                            %
%                     POWERPROFILE2                                            %
%                                                                              %
%            GEN-SERIES OF CORRELATOR PROGRAMS                                 %
%                                                                              %
%                     GEN-A-L3P2                                               %
%                     GEN-B-L2U1P3                                             %
%                     GEN-C-L1U2P3                                             %
%                     GEN-D-U3P2                                               %
%                     GEN-E-L1U1P3                                             %
%                     GEN-F-L1U1P2                                             %
%                     GEN-G-L1U1S1P2                                           %
%                     GEN-REMOTE                                               %
%                                                                              %
%      THE PROGRAM NAME SHOWS THE SUBROUTINE STRUCTURE,                        %
%      GEN-REMOTE IS A REMOTE  STATION ORIENTED PROGRAM.                       %
%                                                                              %
%                                                                              %
%      PROGRAM DEVELOPMENT:  TAUNO TURUNEN                                     %
%                            EISCAT HQ , MARCH 1985                            %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%              TABLE OF INDEXES
%             **********************
%
%                APB STACK REGISTERS
%                ___________________
%
%        ***LONGPULSE APB STACK***
INDEX VOLUMEINDEX=15,LPLAGMAX=14,LPNOGATES=13
%
%        ***UNIPROG APB STACK***
INDEX UNILAGINCR=12,UNILAGMAX=11,UNINOSAMPLES=10
%
%        ***POWERPROFILE1 APB STACK***
INDEX PPNOSAMPLES1=9
%
%        ***POWERPROFILE2 APB STACK***
INDEX PPNOSAMPLES2=8
%
```

A1/17

```
%
%       ***FREE REGISTERS FOR USER***
INDEX USERREG1=7
INDEX USERREG2=6
INDEX USERREG3=5
INDEX USERREG4=4
%
%       ***TEMPORARY STORAGES USED BY PROGRAM***
INDEX LAGINDEXCOUNTER=3
INDEX BUFFERJUMPER=2       % USED ALSO AS UNILAGMAX REGISTER
INDEX BUFSTARTADD=1
%
%       ***CONSTANT (=1)***
INDEX APBINCR=0
%
%
%                 APM STACK REGISTERS
%                 ___________________
%
%        ***PROGRAMMABLE RESULT MEMORY START ADDRESSES***
INDEX RESENTRY1=15,RESENTRY2=14,RESENTRY3=13,RESENTRY4=12
INDEX RESENTRY5=11,RESENTRY6=10,RESENTRY7=9,RESENTRY8=8
INDEX RESENTRY9=7,RESENTRY10=6,RESENTRY11=5,RESENTRY12=4
%
%       *** LONGPULSE (SAME VALUE AS IN LPLAGMAX)***
INDEX LPMAXLAG=3
%
%       *** TEMPORARY STORAGES USED BY PROGRAM***
INDEX RESULTJUMPER=2
INDEX RESTARTADD=1
%
%       ***CONSTANT (=1)***
INDEX APMINCR=0
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                            %
%    COMPILER SUBSTITUTES THE NAMES WITH THE GIVEN NUMBERS                   %
%                                                                            %
%    EXAMPLES: RSAPB(LAGMAX)=RSAPB(14)                                       %
%              RSAPM(APMINCR)=RSAPM(0)                                       %
%                                                                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%    MAIN PROGRAMS CONTAIN USUALLY A LIST OF SUBROUTINE CALLS.
%    THE NUMBER OF FREE PROGRAM STEPS IS 17.
%
%    THE FOLLOWING FOUR DIFFERENT SUBROUTINE CALLS ARE ALLOWED:
```

A7/18

```
%
%     NEXT
%     RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%     RELOAD LCR3
%     RELOADVALUE=(ANY SUITABLE REGISTER COMBINATION)
%     CALL LONGPULSE
%
%
%     NEXT
%     RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%     RELOAD LCR1
%     RELOADVALUE= ( ANY SUITABLE REGISTER COMBINATION)
%     CALL UNIPROG
%
%
%     NEXT
%     RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%     RELOAD LCR1
%     RELOADVALUE= (ANY SUITABLE REGISTER COMBINATION)
%     CALL POWERPROFILE1
%
%
%     NEXT
%     RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%     RELOAD LCR1
%     RELOADVALUE= (ANY SUITABLE REGISTER COMBINATION)
%     CALL POWERPROFILE2
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%     NOTE: IN THE EXAMPLE COMMANDS THE RESENTRYN MUST BE SOME
%           OF THE POSSIBILITIES RESENTRY1,RESENTRY2...RESENTRY11.
%
%     NOTE: THE RSAPM(RESTARTADD)=.... CAN ALSO BE IN THE FORM:
%
%                 RSAPM(RESTARTADD)=0
%
%     NOTE: THE RSAPM(RESTARTADD)=.. SPECIFIES THE FIRST RESULT
%           MEMORY ADDRESS  USED BY THE CALLED SUBROUTINE. THIS
%           ADDRESS IS GIVEN BY THE USER IN THE CORRELATOR APM
%           STACK LOADING. REGISTERS APM(15)-APM(4) HAVE BEEN
%           RESERVED FOR RESENTRIES RESENTRY1....RESENTRY12.
%
%     NOTE: IF THE LINE RSAPM(RESTARTADD)=... IS OMITTED ,THE
%           DEFAULT VALUE IS THE FIRST HIGHER ADDRESS WHICH WAS NOT
%           USED BY THE PREVIOUS SUBROUTINE AND IN CASE OF THE FIRST
%           SUBROUTINE CALL THE DEFAULT VALUE IS  ZERO.
%
%     NOTE: IN LONGPULSE ROUTINE ONE HAS TO LOAD LCR3 TO A VALUE
%           WHICH GIVES THE NUMBER OF GATES. ONE REGISTER IS
%           RESERVED FOR THE PURPOSE AND THE OTHER VALUES HAS TO BE
%           FORMED BY SUITABLE ARITHMETICAL OPERATIONS IN SIMILAR
%           WAY AS IN THE UNIPROG GATING EXPLAINED BELOW.
```

A1/19

```
%
%     NOTE: IN UNIPROG AND POWERPROFILE CALLS ONE HAS TO
%           LOAD LCR1. THIS COMMANDS THE GATING.
%
%           THE VALUE GIVEN IS THE NUMBER OF RANGES ADDED
%           TOGETHER -1. THUS A COMMAND
%
%                  RELOAD LCR1
%                  RELOADVALUE=0
%
%           CAUSES "NORMAL" POWERPROFILE AND UNIPROG
%           BEHAVIOUR. ONLY ONE REGISTER IS FREE
%           AND THUS ONE HAS TO USE REGISTERS IN
%           "CLEVER" WAY.
%
%                   EXAMPLES:
%
%                   RELOADVALUE=0
%                   RELOADVALUE=RSAPB(USERREG3)
%                   RELOADVALUE=RSAPB(APBINCR)
%                   RELOADVALUE=RSAPB(UNILAGINCR)-RSAPB(APBINCR)
%
%
%     NOTE: THE LAST EXAMPLE, WHEN USED IN
%           CONNECTION WITH UNIPROG CALL, CAUSES THAT
%           THE OUTPUT IS ALWAYS EQUIVALENT TO LAGINCREMENT=1
%           DATA. SO THE ROUTINE ADDS THEN ALTITUDE
%           GATES TOGETHER AND LAGINCREMENT GIVES THE NUMBER
%           OF ADDED GATES. WHEN USED IN CONNECTION WITH
%           UNIPROG AND THE MODULATION IS PULSE CODE,
%           THE LAGINCREMENT DIVIDED BY THE NUMBER OF
%           RANGEGATES ADDED TOGETHER MUST BE AN INTEGER.
%           OTHERWISE THE DATA CANNOT BE DECODED.
%
%
%
%     NOTE: THE SUBROUTINES ARE WRITTEN SO THAT THE
%           FIRST SAMPLE TAKEN FROM THE BUFFER MEMORY
%           IS IMMEDIATELY AFTER THE LAST USED SAMPLE
%           OF THE PREVIOUS SUBROUTINE. THE FIRST
%           SUBROUTINE ASSUMES BUFFER MEMORY START
%           ADDRESS TO BE ZERO.
%
%     NOTE: THE LAST CALLED SUBROUTINE MUST USE THE HIGHEST
%           ADDRESSES OF THE RESULT MEMORY. IF THIS RULE IS
%           VIOLATED, SPECIAL ARRANGEMENTS HAS TO BE DONE FOR
%           GETTING SCANCOUNT INTO A CORRECT ADDRESS.
%
```

A1/20

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%    THE FIRST COMMAND IS THE IDLE LOOP    %%%%%%%%%%%%%%%%%%%%
%%%%       DO NOT TOUCH THIS COMMAND          %%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
LOCATION=0
LABEL ZERO
RSAPB(BUFSTARTADD)=0
RSAPM(RESTARTADD)=0
CONTINUE                                          %THE IDLE LOOP ENDS HERE

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%  THE MAIN PROGRAM DEFINED BY THE USER BEGINS HERE  %%%%%%
%%%%%%%%%%  GET-STRING LABEL MAINPROG  %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
CALL SUBRLIST1

NEXT
CALL SUBRLIST1

NEXT
CALL SUBRLIST2

NEXT
CALL SUBRLIST2

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY5)
GOTO SCANCOUNT

NEXT
SUBROUTINE SUBRLIST1
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
RELOAD LCR3
RELOADVALUE=RSAPB(LPNOGATES)
CALL LONGPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY3)
RELOAD LCR3
RELOADVALUE=RSAPB(USERREG1)
CALL LONGPULSE
```

A1/21

```
NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY4)
RELOAD LCR3
RELOADVALUE=RSAPB(USERREG2)
CALL LONGPULSE


NEXT
RSAPM(RESTARTADD)=0
RELOAD LCR1
RELOADVALUE=RSAPB(USERREG3)
CALL POWERPROFILE1

NEXT
RSAPM(RESTARTADD)=0
RELOAD LCR1
RELOADVALUE=RSAPB(USERREG3)
CALL POWERPROFILE1

NEXT
RETURN

NEXT
SUBROUTINE SUBRLIST2
RSAPM(RESTARTADD)=RSAPM(RESENTRY3)
RELOAD LCR3
RELOADVALUE=RSAPB(USERREG1)
CALL LONGPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY4)
RELOAD LCR3
RELOADVALUE=RSAPB(USERREG2)
CALL LONGPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
RELOAD LCR3
RELOADVALUE=RSAPB(LPNOGATES)
CALL LONGPULSE

NEXT
RSAPM(RESTARTADD)=0
RELOAD LCR1
RELOADVALUE=RSAPB(USERREG3)
CALL POWERPROFILE1

NEXT
RSAPM(RESTARTADD)=0
RELOAD LCR1
RELOADVALUE=RSAPB(USERREG3)
CALL POWERPROFILE1
```

A2/22

```
NEXT
RETURN
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% THE USER'S MAIN PROGRAM ENDS HERE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% POSSIBLE USER'S SUBROUTINES MUST  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% START AFTER THE SCANCOUNT COMMAND %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%    SCANCOUNT TERMINATES THE COMPUTATION AND                                  %
%    MUST BE THE FIRST COMMAND AFTER THE USERS                                 %
%    MAIN PROGRAM                                                              %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NEXT
LABEL SCANCOUNT
RESMEMADDRESS=RSAPM(RESTARTADD)
CHANNEL1=-1 CHANNEL2=-1
INITIALIZE ACCUMULATOR
LOAD IREG
STORE OREG
STROBE IREG
GOTO LASTCOMMAND
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%      THE SUBROUTINES WRITTEN BY THE USER MUST START HERE                     %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%      EVERYTHING WRITTEN BY THE USER MUST END HERE                            %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%                      SUBROUTINE UNIPROG                                      %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
NEXT
SUBROUTINE UNIPROG
RSAPB(LAGINDEXCOUNTER)=0
CONTINUE

NEXT
LC1=LCR1
RELOAD LCR3
RELOADVALUE=RSAPB(UNILAGMAX)
CALL DUMMY

NEXT
LC3=LCR3
LCR1A=LC1
CONTINUEANDPUSH

NEXT
LABEL UNICOMPUTE
RELOAD LCR2
RELOADVALUE=RSAPB(UNINOSAMPLES)-RSAPB(LAGINDEXCOUNTER)
GOTO UNIENTRY

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
%      PROGRAM UNIPROG                                      %
%      LC1=GATINGCOUNTER                                    %
%      LC2=PRODUCT COUNTER                                  %
%      LC3=LAG COUNTER                                      %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

NEXT
LABEL UNIENTRY
CONTINUE

NEXT
LC2=LCR2
QAPB=RSAPB(BUFSTARTADD)-RSAPB(APBINCR)
QAPM=RSAPM(RESTARTADD)
CONTINUE

NEXT
LABEL SAMERESADDRESS
BUFFERADDRESS=QAPB+RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=QAPM
QAPM=RESMEMADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=Y*X-X*Y
IF (LC2=0) THEN GOTO LAGTEST ELSE CONTINUE
```

```
NEXT
LABEL PROFILECOMPUTE
BUFFERADDRESS=QAPB+RSAPB(LAGINDEXCOUNTER)
RESMEMADDRESS=QAPM
RSAPM(RESTARTADD)=RESMEMADDRESS
CHANNEL1=OLDVALUE*X+OLDVALUE*Y
CHANNEL2=OLDVALUE*X-OLDVALUE*Y
INITIALIZE ACCUMULATOR
STROBE IREG
LOAD IREG
STORE OREG
LC2=LC2-1
IF (LC1=0) THEN LC1=LCR1A ELSE LC1=LC1-1
IF (LC1=0) THEN CONTINUE ELSE GOTO SAMERESADDRESS

NEXT
LABEL NEXTRESADDRESS
BUFFERADDRESS=QAPB+RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=QAPM+RSAPM(APMINCR)
QAPM=RESMEMADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=Y*X-X*Y
IF (LC2=0) THEN CONTINUE ELSE GOTO PROFILECOMPUTE

NEXT
LABEL LAGTEST
QAPB=QAPB+RSAPB(LAGINDEXCOUNTER)
CONTINUE

NEXT
LC3=LC3-1
RSAPM(RESTARTADD)=RSAPM(RESTARTADD)+RSAPM(APMINCR)
RSAPB(LAGINDEXCOUNTER)=RSAPB(LAGINDEXCOUNTER)+RSAPB(UNILAGINCR)
IF (LC3=0) THEN GOTOANDPOP UNIFINISH ELSE GOBACK


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                            %
%   PROGRAM GEN-DUMP                                                         %
%                                                                            %
%   THIS PROGRAM CONTROLS THE DATA TRANSFER                                  %
%   AND MUST START FROM THE LOCATION 32                                      %
%                                                                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
LOCATION=32
RELOAD LCR1
RELOADVALUE=DATAI
PREPARETRANSFER
CONTINUE
```

A1/25

```
NEXT
PREPARETRANSFER
CONTINUE

NEXT
TRANSFER STATUSWORD
QAPM=-RSAPM(APMINCR)
LC1=LCR1
CONTINUE

NEXT
TRANSFER CONTROLWORD
CONTINUE

NEXT
LABEL TRANSFERLOOP
LC1=LC1-1
RESMEMADDRESS=QAPM+RSAPM(APMINCR)
QAPM=RESMEMADDRESS
QAPB=RSAPB(APBINCR)+RSAPB(APBINCR)
TRANSFER CHANNEL1 MSPART
CONTINUE

NEXT
RESMEMADDRESS=QAPM
TRANSFER CHANNEL1 LSPART
RELOAD LCR2
RELOADVALUE=QAPB+RSAPB(APBINCR)
CONTINUE

NEXT
RESMEMADDRESS=QAPM
TRANSFER CHANNEL2 MSPART
CONTINUE

NEXT
RESMEMADDRESS=QAPM
TRANSFER CHANNEL2 LSPART
LC2=LCR2
IF (LC1=0) THEN CONTINUEANDPUSH ELSE GOTO TRANSFERLOOP

NEXT
LABEL FINISHLOOP
LC2=LC2-1
FINISHTRANSFER
SET START-EXPERIMENT MODE
IF (LC2=0) THEN GOTO ZERO ELSE GOBACK
```

```
%
%   THE TRANSFER ROUTINE ENDS HERE

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                             %
%    THE LAST COMMAND OF UNIPROG                                              %
%                                                                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

NEXT
LABEL UNIFINISH
RSAPB(BUFSTARTADD)=QAPB
RETURN
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                             %
%                    SUBROUTINE DUMMY                                         %
%                                                                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE DUMMY
RETURN
%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                             %
%                                                                             %
%     SUBROUTINE POWERPROFILE1                                                %
%                                                                             %
%     LC1=GATING COUNTER                                                      %
%     LC2=SAMPLE COUNTER                                                      %
%                                                                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE POWERPROFILE1
RSAPB(LAGINDEXCOUNTER)=0
CONTINUE

NEXT
LC1=LCR1
RELOAD LCR3
RELOADVALUE=0
CALL DUMMY

NEXT
LCR1A=LC1
LC3=LCR3
RELOAD LCR2
RELOADVALUE=RSAPB(PPNOSAMPLES1)
GOTOANDPUSH UNIENTRY
```

```
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
%                                                           %
%      SUBROUTINE POWERPROFILE2                             %
%                                                           %
%      LC1=GATING COUNTER                                   %
%      LC2=SAMPLE COUNTER                                   %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE POWERPROFILE2
RSAPB(LAGINDEXCOUNTER)=0
CONTINUE

NEXT
LC1=LCR1
RELOAD LCR3
RELOADVALUE=0
CALL DUMMY

NEXT
LCR1A=LC1
LC3=LCR3
RELOAD LCR2
RELOADVALUE=RSAPB(PPNOSAMPLES2)
GOTOANDPUSH UNIENTRY

%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
%      SUBROUTINE LONGPULSE                                 %
%                                                           %
%      THIS SUBROUTINE COMPUTES THE TARGET ACF ESTIMATE     %
%      IN THE WAY THAT THE ABSOLUTE VOLUME BOUNDARIES ARE   %
%      THE SAME AT ALL LAGS                                 %
%                                                           %
%      LC1=LAGCOUNTER                                       %
%      LC2=VOLUME INDEX CONTROL COUNTER                     %
%      LC3=GATE COUNTER                                     %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE LONGPULSE
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(LPLAGMAX)
CONTINUE
```

A7/28

```
NEXT
LC3=LCR3
RELOAD LCR1
RELOADVALUE=RSAPB(LPLAGMAX)
CALL DUMMY

NEXT
LC1=LCR1
RSAPM(RESTARTADD)=RSAPM(RESTARTADD)-RSAPM(APMINCR)
RELOAD LCR2
RELOADVALUE=RSAPB(VOLUMEINDEX)
LC3=LC3-1
IF (LC3=0) THEN RETURN ELSE CALL DUMMY

NEXT
LABEL LONGGATE
LC2=LCR2
LCR1A=LC1
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(VOLUMEINDEX)
RSAPM(RESTARTADD)=RSAPM(RESTARTADD)+RSAPM(APMINCR)
CONTINUE

NEXT
RSAPB(BUFFERJUMPER)=-RSAPB(APBINCR)
RSAPM(RESULTJUMPER)=-RSAPM(APMINCR)
CONTINUE

NEXT
LABEL FULLROW
BUFFERADDRESS=RSAPB(BUFSTARTADD)-RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=RSAPM(RESTARTADD)
QAPM=RESMEMADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=Y*X-X*Y
LCR1A=LC1
LC1=LC1-1
LC2=LC2-1
INITIALIZE ACCUMULATOR
STROBE IREG
LOAD IREG
STORE OREG
IF (LC1=0) THEN GOTO TESTWHATTODO ELSE CONTINUE
```

A1/29

```
NEXT
LABEL COLUMNLAG
BUFFERADDRESS=QAPB+RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=QAPM+RSAPM(APMINCR)
QAPM=RESMEMADDRESS
LC1=LC1-1
CHANNEL1=OLDVALUE*X+OLDVALUE*Y
CHANNEL2=OLDVALUE*X-OLDVALUE*Y
STROBE IREG
LOAD IREG
STORE OREG
IF (LC1=0) THEN CONTINUE ELSE GOTO COLUMNLAG

NEXT
LABEL TESTWHATTODO
LC1=LCR1A
RESMEMADDRESS=QAPM-RSAPM(LPMAXLAG)
QAPM=RESMEMADDRESS
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)-RSAPB(APBINCR)
IF (LC2=0) THEN CONTINUE ELSE GOTO FULLROW

NEXT
LABEL ROWPIECE
LC1=LC1-1
RSAPM(RESULTJUMPER)=RSAPM(RESULTJUMPER)+RSAPM(APMINCR)
RSAPB(BUFFERJUMPER)=RSAPB(BUFFERJUMPER)+RSAPB(APBINCR)
IF (LC1=0) THEN GOTO NEXTGATE ELSE CONTINUE

NEXT
LCR1A=LC1
BUFFERADDRESS=RSAPB(BUFSTARTADD)-RSAPB(APBINCR)
QAPB=BUFFERADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=X*Y-Y*X
CONTINUE

NEXT
BUFFERADDRESS=QAPB+RSAPB(BUFFERJUMPER)
QAPB=BUFFERADDRESS
RESMEMADDRESS=RSAPM(RESTARTADD)+RSAPM(RESULTJUMPER)
QAPM=RESMEMADDRESS
CHANNEL1=OLDVALUE*X+OLDVALUE*Y
CHANNEL2=OLDVALUE*X-OLDVALUE*Y
GOTO COLUMNLAG
```

A1/30

```
NEXT
LABEL NEXTGATE
LC1=LCR1
LC2=LCR2
RSAPB(BUFSTARTADD)=QAPB+RSAPB(VOLUMEINDEX)
RSAPM(RESTARTADD)=RSAPM(RESTARTADD)+RSAPM(LPMAXLAG)
LC3=LC3-1
IF (LC3=0) THEN CONTINUE ELSE GOTO LONGGATE

NEXT
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(LPLAGMAX)
RSAPM(RESTARTADD)=RSAPM(RESTARTADD)+RSAPM(APMINCR)
RETURN
%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                               %
% THE LAST COMMAND TERMINATES THE COMPUTATION AND                               %
% THE CORRELATOR GOES TO THE IDLE LOOP                                          %
%                                                                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
NEXT
LABEL LASTCOMMAND
STROBE IREG
SET CONTINUE-EXPERIMENT MODE
GOTO ZERO

END
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

NOTE: GEN-5 IS IDENTICAL TO CP-3-D

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%
%%          EXPERIMENT CP-3-D,FILE CP-3-D-R:ELAN
%%
%%          BASED ON ALGORITHM GEN-5
%%
%%          WRITTEN BY TAUNO TURUNEN
%%          EISCAT HQ, APRIL.1985
%%
%%          GEOMETRICAL PROPERTIES IDENTICAL TO THOSE IN  CP-3-C
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
WRI-EXP-HEAD
%
WRITE-FILE (EXP)CP-3-D-R:ELAN
WRITE-FILE (EXP)CP-3-D-R:TLAN
WRITE-FILE (EXP)CP-3-D:DESC
WRITE-FILE (EXP)CP-3-D-R-SUBR:LIST
%
OFFSET-PPD -376             %+35-390+21 = DELAY TO ILLUMINATE THE VOLUME
%                                        -START EARLIER TO TAKE 40 SAMPLES
%                                         +FILTER DELAY
%                           %NOTE:IN TLAN FILE THE START SAMPLING IS AT
%                                 AN INSTANT OF RFON FOR THE PULSES
%
SET-SIG-PATH ALLX
SET-SIG-ATT X,5
%
POINT-REF-HE 344.9,28.0,325.0
%
&K SET-POL 90,0
&S SET-POL 87,0
%
%
LOAD-RADAR (EXP)CP-3-D-R
%
TRANSF-NOISE-CONT RAD
%
DEF-FLUKE-POS 3,Y
DEF-FLUKE-POS 4,Y
%
SET-LO1 1053.5
%
%
%
SET-CH-ATT 3,4
SET-CH-ATT 4,4
SET-CH-ATT 5,4
SET-CH-ATT 6,4
%
```

```
SET-LO2 3,94.0
SET-LO2 4,93.5
SET-LO2 5,93.0
SET-LO2 6,92.5
%
SET-FILTER 3,BU,25.0
SET-FILTER 4,BU,25.0
SET-FILTER 5,BU,25.0
SET-FILTER 6,BU,25.0
%
SYNC -120
%
LOAD-CORR (EXP)CP-3-D-R:CCOD        %MASTER PROGRAM GEN-REMOTE:CLAN
%
SYNC 40
%
SET-CO-APB 15,-10       %MARGIN1 (OVERLAPPING)
SET-CO-APB 14,30        %SIGSAMPLES1
SET-CO-APB 13,20        %LAGMAX1
SET-CO-APB 12,3         %CALGATES1
SET-CO-APB 11,130       %CALPRODUCTS1
SET-CO-APB 10,0         %MARGIN2
SET-CO-APB  9,0         %SIGSAMPLES2
SET-CO-APB  8,20        %LAGMAX2
SET-CO-APB  7,3         %CALGATES2
SET-CO-APB  6,130       %CALPRODUCTS2
SET-CO-APB  2,5         %USERREG (NUMBER OF SIGGATES)
SET-CO-APB  0,1         %APBINCR
%
%
SET-CO-APM 15,0         %RESENTRY1, SIGNAL PART 2+1+2 ACF:S
SET-CO-APM 14,105       %RESENTRY2, CALIBRATION ACF:S (3)
SET-CO-APM 13,168       %SCANCOUNTADDRESS
%
SET-CO-APM  3,20        %LPMAXLAG1
SET-CO-APM  2,20        %LPMAXLAG2
%
SET-CO-APM  0,1         %APMINCR
%
%
SET-CO-DATAID 169
%
%
SYNC 40
%
SET-BUF-M 3,0000
SET-BUF-M 4,2020
SET-BUF-M 5,1010
SET-BUF-M 6,3030
%
```

```
%
SET-ADC-INT 3,10
SET-ADC-INT 4,10
SET-ADC-INT 5,10
SET-ADC-INT 6,10
%
%
START-CORR
SYNC 30
START-RADAR 10,1,10
%
ENABLE-DMA
%
%
START-RADAR 10,1,10
%
ENABLE-DMA
%
SYNC 10
%
%
%            ************ ANTENNA MOTION LOOP***************
%                          FROM CP-3-C
%
%
DO -1
  START-REC-DATA
  SYNC 100                          % T1         100   SEC
%
  POINT-REF-H 344.9,35.1,325.0  % T2      20+ 90  SEC
%  &K SET-POLAR 90,0                    POLAR NOT CHANGED
&S SET-POLAR 86,0
  SYNC 20                           %         ANTENNA MOTION
  SYNC 90
%
  POINT-REF-H 344.9,43.7,325.0  % T3      20+ 90   SEC
% &KSET-POLAR 90,0                      POLAR NOT CHANGED
&S  SET-POLAR 85,-1
  SYNC 20                           %         ANTENNA MOTION
  SYNC 90
%
  POINT-REF-H 344.9,54.0,325.0  % T4      20+ 90   SEC
&K  SET-POLAR 90,-1
&S  SET-POLAR 83,-3
  SYNC 20                           %         ANTENNA MOTION
  SYNC 90
%
  POINT-REF-H 344.9,65.4,325.0  % T5      20+ 90   SEC
&K  SET-POLAR 90,-2
&S  SET-POLAR 82,-6
  SYNC 20                           %         ANTENNA MOTION
  SYNC 90
%
```

A1/34

```
  POINT-REF-H 344.9,77.5,325.0  % T6      20+ 80   SEC
&K  SET-POLAR 90,-4
&S  SET-POLAR 84,-10
  SYNC 20                       %         ANTENNA MOTION
  SYNC 80
%
  POINT-REF-H 344.9,90.0,325.0  % K7      20+120   SEC   TROMSO T7&T8
&K  SET-POLAR 90,-5
&S  SET-POLAR 90,-15
  SYNC 20                       %         ANTENNA MOTION
  SYNC 120
%
  POINT-REF-H 164.9,77.9,325.0  % T9      20+80    SEC
&K  SET-POLAR 90,-6
&S  SET-POLAR 101,-17
  SYNC 20                       %         ANTENNA MOTION
  SYNC 80
%
  POINT-REF-H 164.9,66.7,325.0  % T10     20+ 70   SEC
% &KSET-POLAR 90,-6                       POLAR NOT CHANGED
&S  SET-POLAR 114,-14
  SYNC 20                       %         ANTENNA MOTION
  SYNC 70
%
&K  CHANGE-REF-SITE KIRUNA,,,,
&K  POINT-REF-H 3.9,89.99,325.0  % K10
&S  POINT-REF-H 164.9,57.1,325.0 % T11    30+120   SEC
&K  SET-POLAR 90,0                %       SET FOR CIRCULAR
&S  SET-POLAR 122,-7
  SYNC 30                        %        ANTENNA MOTION
&S  SYNC 120
%
&K  POINT-REF-H 131.1,89.99,325.0 %K11
&K  SET-POLAR 90,0                 %      SET FOR CIRCULAR
&K  SYNC 120
%
&K  CHANGE-REF-SITE TROMSO,,,,
  POINT-REF-H 164.9,49.0,325.0  % T12     30+ 70   SEC
&K  SET-POLAR 90,-4
&S  SET-POLAR 121,2
  SYNC 30                       %         ANTENNA MOTION
  SYNC 70
%
  POINT-REF-H 164.9,42.1,325.0  % T13     20+ 80   SEC
&K  SET-POLAR 90,-1
&S  SET-POLAR 112,9
  SYNC 20                       %         ANTENNA MOTION
  SYNC 80
%
  POINT-REF-H 164.9,35.6,325.0  % T14     20+ 90   SEC
% &KSET-POLAR 90,-1                       POLAR NOT CHANGED
&S  SET-POLAR 99,12
  SYNC 20                       %         ANTENNA MOTION
  SYNC 90
%
```

A1/3T

```
  POINT-REF-H 164.9,29.7,325.0  % T15     20+ 90   SEC
&K  SET-POLAR 90,0
&S  SET-POLAR 90,9
  SYNC 20                       %         ANTENNA MOTION
  SYNC 90
%
  POINT-REF-H 164.9,24.6,325.0  % T16     20+ 90   SEC
% &KSET-POLAR 90,0                       POLAR NOT CHANGED
&S  SET-POLAR 88,6
  SYNC 20                       %         ANTENNA MOTION
  SYNC 90
%
  STOP-REC-DATA
  POINT-REF-H 344.9,28.0,325.0  % T1         150   SEC
% &KSET-POLAR 90,0
&S  SET-POLAR 87,0
  SYNC 150                      %         ANTENNA MOTION
ENDDO
%
STOP-EXPERIMENT,,
```

A1/36

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%       GEN-5-R:TLAN
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
AT 10 RECEV
AT 100 CH3
AT 452 CH4
AT 1195 CH3OFF
AT 1547 CH4OFF
%
%           110 SAMPLES TAKEN FOR 5 30 SAMPLE GATES
%           WITH 10 SAMPLES OVERLAPPING, SIGNAL SHOULD BE IN THE
%           MIDDLE GATE
%
AT 3005 CH3,CH4,CH5,CH6
AT 6000 ALLOFF,CAL30
%
%    2*(130+20) SAMPLES AT ALL CHANNELS, SKY NOISE
%
AT 6505 CH3,CH4,CH5,CH6
AT 8000 ALLOFF,CAL0
%
%       (130+20)SAMPLES AT EVERY CHANNEL, NOISE INJECTION)
%
AT 8600 CH5
AT 8952 CH6
AT 9695 CH5OFF
AT 10047 CH6OFF
%
%           110 SAMPLES OF SIGNAL FOR 5 GATES (30 SAMPLES, OVERLAPPING 10)
%
%
AT 11005 CH3,CH4,CH5,CH6
AT 14000 ALLOFF,CAL30
%    2*(130+20) SAMPLES TAKEN AT EVERY CHANNEL,SKY NOISE
%
AT 14505 CH3,CH4,CH5,CH6
AT 15000 STC
AT 16000 ALLOFF,CAL0
%
%    150 SAMPLES AT EVERY CHANNEL,NOISE INJECTION
%
AT 16040 BUFLIP
%
AT 17256 REP
```

A7/37

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%           GEN-5-R-SUBR:LIST
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NEXT
CALL SUBRLIST

NEXT
CALL SUBRLIST

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY3)
GOTO SCANCOUNT

NEXT
SUBROUTINE SUBRLIST
RSAPM(RESTARTADD)=RSAPM(RESENTRY1)
RELOAD LCR2
RELOADVALUE=RSAPB(USERREG)
CALL REMOTE1

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
CALL IMPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
CALL IMPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY1)
RELOAD LCR2
RELOADVALUE=RSAPB(USERREG)
CALL REMOTE1

NEXT
RETURN
```

A1/38

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%          CORRELATOR PROGRAM GEN-REMOTE:CLAN
%
%          THIS CORRELATOR PROGRAM IS MAINLY INTENDED FOR REMOTE
%          STATION SIGNAL HANDLING AND FOR SYSTEM CALIBRATION PURPOSES.
%
%          SUBROUTINES:  REMOTE1
%                        REMOTE2
%                        IMPULSE
%
%          THIS PROGRAM CAN BE USED IN SEVERAL WAYS DESCRIBED IN THE
%          REPORT : CORRELATOR PROGRAMS FOR THE GEN-SYSTEM
%
%          PROGRAM DEVELOPMENT:  TAUNO TURUNEN, EISCAT HQ
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%         TABLE OF INDEXES
%
%         THE INDEX TABLE RELATES THE GIVEN NAMES TO NUMBERS, WHICH
%         ARE THEN USED AS REGISTER NUMBERS. THERE IS NO DISTINCTION
%         BETWEEN THE APB AND APM STACKS
%
%         EXAMPLE: RSAPB(MARGIN1) IS RSAPB(15) AFTER COMPILATION
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
INDEX MARGIN1=15              % SUBROUTINE
INDEX SIGSAMPLES1=14          % REMOTE1
INDEX LAGMAX1=13              % APB
INDEX CALGATES1=12            % STACK
INDEX CALPRODUCTS1=11         % REGISTERS
%
INDEX MARGIN2=10              % SUBROUTINE
INDEX SIGSAMPLES2=9           % REMOTE2
INDEX LAGMAX2=8               % APB
INDEX CALGATES2=7             % STACK
INDEX CALPRODUCTS2=6          % REGISTERS
%
INDEX IMPULSEPRODUCTS=5       % SUBROUTINE IMPULSE APB STACK REGISTERS
INDEX IMPULSEGATES=4          % AND TEMPORARY STORAGES FOR
INDEX LAGMAX=3                % SUBROUTINES REMOTE1 AND REMOTE2.
%                             % 5 IS THE TEMPORARY STORAGE FOR MARGIN AND
%                             % CALPRODUCTS, 4 IS THE TEMPORARY STORAGE FOR
%                             % SIGSAMPLES AND CALGATES.
%
INDEX USERREG=2               % FREE REGISTER FOR USERS PARAMETERS
%
INDEX BUFSTARTADD=1 % APB ADDRESS COUNTER
INDEX APBINCR=0     % APB INCREMENT (=1)
```

A1/39

```
INDEX RESENTRY1=15,RESENTRY2=14,RESENTRY3=13,RESENTRY4=12
INDEX RESENTRY5=11,RESENTRY6=10,RESENTRY7=9,RESENTRY8=8
INDEX RESENTRY9=7,RESENTRY10=6
%
INDEX RESJUMPER=5       % TEMPORARY STORAGE FOR MAXLAG+1
%
INDEX MAXLAG=4          % SUBROUTINE IMPULSE REGISTER AND THE TEMPORARY
%                       % STORAGE FOR MAXLAG1 AND MAXLAG2.
%
INDEX MAXLAG1=3         % REMOTE1 APM STACK REGISTER
%
INDEX MAXLAG2=2         % REMOTE2 APM STACK REGISTER
%
INDEX RESTARTADD=1      % APM ADDRESS COUNTER
INDEX APMINCR=0         % APM INCREMENT  (=1)
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%           THE FOLLOWING THREE SUBROUTINE CALLS ARE ALLOWED:
%
%           NEXT
%           RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%           RELOAD LCR2
%           RELOADVALUE=...(ANY SUITABLE REGISTER COMBINATION)
%           CALL REMOTE1
%
%           NEXT
%           RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%           RELOAD LCR2
%           RELOADVALUE=...(ANY SUITABLE REGISTER COMBINATION)
%           CALL REMOTE2
%
%           NEXT
%           RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%           RSAPB(IMPULSEGATES)=...(ANY SUITABLE REGISTER COMBINATION)
%           CALL IMPULSE
%
%           THE THIRD LINE IS NOT ALWAYS NEEDED IN CALL IMPULSE CALL,
%           SEE ALSO THE NOTES BELOW.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%           THE MAXIMUM NUMBER OF USER DEFINED PROGRAM STEPS IS 13
%
%           NOTE:  IN THE EXAMPLE COMMANDS THE RESENTRYN MUST BE SOME
%                  OF THE POSSIBILITIES RESENTRY1,RESENTRY2...RESENTRY10.
%
%                  THE RSAPM(RESTARTADD)=.... CAN ALSO BE IN THE FORM:
%                  RSAPM(RESTARTADD)=0
%
```

A1/40

```
%
%               THE RSAPM(RESTARTADD)=.. SPECIFIES THE FIRST RESULT
%               MEMORY ADDRESS  USED BY THE CALLED SUBROUTINE. THIS
%               ADDRESS IS GIVEN BY THE USER IN THE CORRELATOR APM
%               STACK LOADING. REGISTERS APM(15)-APM(5) HAVE BEEN
%               RESERVED FOR RESENTRIES RESENTRY1....RESENTRY11.
%
%               IF THE LINE RSAPM(RESTARTADD)=... IS OMITTED ,THE
%               DEFAULT VALUE IS THE FIRST HIGHER ADDRESS WHICH WAS NOT
%               USED BY THE PREVIOUS SUBROUTINE AND IN CASE OF THE FIRST
%               SUBROUTINE CALL THE DEFAULT VALUE IS  ZERO.
%
%         NOTE: "...(ANY SUITABLE REGISTER COMBINATION)" IS OFTEN SOME
%               OF THE FOLLOWING:
%
%               Ø
%               RSAPB(APBINCR)                  %=1
%               RSAPB(APBINCR)+RSAPB(APB(INCR)  %=2
%               RSAPB(USERREG)
%
%
%         NOTE: IN CONNECTION WITH "CALL IMPULSE" THE APB REGISTER
%               HAVING LOADING IN THE SUBROUTINE CALL CAN ALSO BE THE
%               IMPULSEPRODUCTS BUT NOT LAGMAX WITHOUT CORRESPONDING
%               APM REGISTER LOADING (MAXLAG) WHICH IS POSSIBLE ONLY IF
%               THERE IS NO NEED TO LOAD RESTARTADD.
%
%         NOTE: THE LAST SUBROUTINE MUST BE THE ONE , WHICH
%               USES THE HIGHEST ADDRESSES IN THE RESULT MEMORY.
%               THE SCANCOUNT IS WRITTEN TO THE NEXT UNUSED
%               RESULT MEMORY POSITION. IF THIS RULE IS VIOLATED,
%               ONE HAS TO LOAD RSAPM(RESTARTADD) TO SCANCOUNTADDRESS
%               AS A LAST COMMAND OF THE MAIN PROGRAM. ONE POSSIBLE FORM
%               TO DO THIS IS THE FOLLOWING COMMAND:
%
%                 NEXT
%                 RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%                 GOTO SCANCOUNT
%                 %RESENTRYN CONTAINS THE SCANCOUNTADDRESS (DATAIO-1)
%
%         NOTE: THE SUBROUTINES ARE WRITTEN SO THAT THE
%               FIRST SAMPLE TAKEN FROM THE BUFFER MEMORY
%               IS IMMEDIATELY AFTER THE LAST USED SAMPLE
%               OF THE PREVIOUS SUBROUTINE. THE FIRST
%               SUBROUTINE ASSUMES BUFFER MEMORY START
%               ADDRESS TO BE ZERO.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

A7/41

```
%
%          RULES FOR THE SUBROUTINES REMOTE1 AND REMOTE2
%        ***************************************************
%
%
%       1) ***LCR2 IS LOADED TO 1 (APBINCR) IN THE SUBROUTINE CALL***
%
%       LCR2 LOADING GIVES THE NUMBER OF GATES FOR THE SCATTER ACF
%       COMPUTATION (TRIANGULAR WEIGHTED ACF WITH PROGRAMMABLE LAGMAX).
%
%       IF MARGIN=0 THEN POWERPROFILE (TIMINGCKECK) IS NOT COMPUTED
%       AND THE COMPUTATION STARTS FROM THE SCATTERACF.
%
%       IF ALSO SIGSAMPLES=0 THEN  THE COMPUTATION STARTS FROM THE
%       BOXCAR WEIGHTED (IMPULSE COMPUTATION) ACF WITH THE LAGMAX, CALGATES
%       AND CALPRODUCTS GIVEN IN THE APB STACK FOR THE CALLED SUBROUTINE.
%       THIS IS ONE OF THE POSSIBILITIES TO COMPUTE "IMPULSE".
%
%       IF EITHER CALPRODUCTS OR CALGATES IS ZERO THEN THE COMPUTATION
%       TERMINATES AFTER SCATTER ACF.
%
%       2) ***LCR2 IS LOADED TO ZERO IN THE SUBROUTINE CALL***
%
%       IF THE NUMBER OF GATES GIVEN FOR THE SCATTER ACF (LCR2 LOADING
%       IN THE SUBROUTINE CALL) IS ZERO,  THE COMPUTATION
%       STARTS FROM THE BOXCAR WEIGHTED CALIBRATION ACF (IMPULSE).
%
%
%       3) *** LCR2 LOADED TO A VALUE GREATER THAN 1***
%
%       IF THE NUMBER OF GATES GIVEN FOR THE SCATTER ACF IS
%       NEITHER 1 NOR ZERO THEN THE POWERPROFILE TIMING CHECK
%       IS SKIPPED AND THE SCATTER GATES ARE COMPUTED WITH SAMPLE SPACE
%       OVERLAPPING = - MARGIN, (MARGIN LOADED AS A NEGATIVE NUMBER IN
%       THIS CASE). AFTER SCATTER GATES PROGRAM BRANCHES TO BOXCAR ACF
%       IF THAT PART IS NOT SKIPPED.
%
%       THIS GIVES POSSIBILITY TO MAKE TIMING CHECK USING ACF:S INSTEAD
%       OF POWERPROFILES. SIMULTANEOUS USE OF BOTH POWERPROFILE AND ACF TYPES
%       OF TIMING (AND POWER STABILITY) CHECKING DEMANDS THREE STEP
%       SUBROUTINE CALLS AND IS NOT EXPLAINED HERE.
%
%       WITH THE EXPLAINED WAY ONE CAN ALSO SIMULATE MANY OLDER CORRELATOR
%       PROGRAMS (PROGRAMMABLE NUMBER OF LAGS AND OVERLAPPING)
%            ------------------------------------------
%
%
```

A1/42

```
%
%                RULES FOR THE SUBROUTINE IMPULSE:
%            ******************************************
%
%       1) *** IMPULSE COMPUTATION USING REMOTE1 AND/OR REMOTE2***
%
%       ONE WAY TO GET THE "CALL IMPULSE" COMPUTATION IS TO SKIP THE TIMING
%       CHECK AND SCATTER ACF IN THE REMOTE1 AND/OR REMOTE2. THEN THE
%       NUMBER OF GATES, CALPRODUCTS, LAGMAX AND MAXLAG ARE THOSE LOADED
%       INTO THE CORRESPONDING REGISTERS OF REMOTE1 AND/OR REMOTE2.
%
%       2) *** CALL IMPULSE WHEN NEITHER REMOTE1 NOR REMOTE2 ARE USED***
%
%       ONE OF THE APB REGISTERS CAN BE RELOADED (E.G. "IMPULSEGATES")
%       TO A WANTED VALUE BUT THIS IS NOT NEEDED IF THE REGISTER
%       IS ALREADY CONSTANTLY LOADED TO A WANTED VALUE IN THE ELAN FILE
%       OR HAS THE PROPER VALUE BECAUSE OF PREVIOUS COMPUTATION.
%       THE NUMBER OF GATES IS IN THE "IMPULSEGATES", THE NUMBER OF
%       CROSSPRODUCTS IN "IMPULSEPRODUCTS", THE PARAMETER "LAGMAX"
%       IN THE REGISTER "LAGMAX", AND "MAXLAG" IN THE REGISTER "MAXLAG".
%
%       IF THE SUBROUTINE IMPULSE IS THE ONLY SUBROUTINE USED IN THE
%       PROGRAM, THEN THE REGISTERS MENTIONED ABOVE HAVE TO BE LOADED
%       TO WANTED THE VALUES IN THE ELAN FILE. NOTE THE POSSIBILITY TO
%       RELOAD ONE OF THE APB REGISTERS IN THE SUBROUTINE CALL.
%
%       3)  ***CALL IMPULSE WHEN REMOTE1 AND/OR REMOTE2 ARE ALSO USED***
%
%       THE REGISTERS CONTROLLING THE COMPUTATION HAVE A VALUE
%       LOADED TO THESE REGISTERS BY THE PREVIOUSLY CALLED SUBROUTINE
%       (REMOTE1 OR REMOTE2). IMPULSEPRODUCTS IS CALPRODUCTS1 OR CALPRODUCTS2
,
%       LAGMAX IS LAGMAX1 OR LAGMAX2, AND MAXLAG IS MAXLAG1 OR MAXLAG2 ETC.
%       DEPENDING ON WHICH ONE OF THE ROUTINES REMOTE1 OR REMOTE2 WAS
%       CALLED IN THE PREVIOUS MAIN PROGRAM STEP. ONE OF THE APB REGISTERS
%       CAN BE RELOADED IN THE SUBROUTINE CALL (OFTEN NUMBER OF GATES) AND
%       RESULT MEMORY START ADDRESS PROGRAMMED.
%
%       IF THE "CALL IMPULSE" IS THE FIRST SUBROUTINE CALL IN THE MAIN
%       PROGRAM, THEN THE STARTING VALUES MUST BE LOADED IN THE ELAN FILE
%       TO PREVENT POSSIBLE CORRELATOR "HANGING".
%
%       SUBROUTINE IMPULSE DOES NOT CHANGE THE CONTENTS OF THE CONTROL
%       REGISTERS IT USES.
%                       ------------------------------
%
%
```

A1/43

```
%
%
%              POWER PROFILE COMPUTATION USING GEN-REMOTE
%            ************************************************
%
%
%       THE POWER PROFILE TYPE OF TIMING CHECKING IS ARRANGED SO THAT
%       IT IS NOT VERY SUITABLE FOR INDEPENDENT POWER PROFILE COMPUTATION.
%       ON THE DIFFERENT POSSIBILITIES THE FOLLOWING IS RECOMMENDED.
%
%       LOAD CALPRODUCTS1 TO NUMBER OF SAMPLES ADDED TOGETHER AND CALGATES1
%       TO NUMBER OF GATES. LOAD LAGMAX1 AND MAXLAG1 REGISTERS TO ZERO.
%       CALL REMOTE1  WITH LCR2 LOADING=0. THIS PRODUCES GATED POWERPROFILE
%       IF CALPRODUCTS1#0 AND NORMAL POWERPROFILE IF CALPRODUCTS1=0.
%       SIMILAR COMPUTATION IS POSSIBLE WITH REMOTE2 AND ALSO WITH IMPULSE.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%    THE FIRST COMMAND IS THE IDLE LOOP    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%    DO NOT TOUCH THIS COMMAND             %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
LOCATION=0
LABEL ZERO
RSAPB(BUFSTARTADD)=0
RSAPM(RESTARTADD)=0
CONTINUE                        %THE IDLE LOOP ENDS HERE

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%   THE MAIN PROGRAM WRITTEN BY THE USER BEGINS HERE  %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%         LABEL FOR GET-STRING : MAINPROG                %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%           GEN-5-R-SUBR:LIST
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NEXT
CALL SUBRLIST

NEXT
CALL SUBRLIST

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY3)
GOTO SCANCOUNT
```

A1/44

```
NEXT
SUBROUTINE SUBRLIST
RSAPM(RESTARTADD)=RSAPM(RESENTRY1)
RELOAD LCR2
RELOADVALUE=RSAPB(USERREG)
CALL REMOTE1

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
CALL IMPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
CALL IMPULSE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY1)
RELOAD LCR2
RELOADVALUE=RSAPB(USERREG)
CALL REMOTE1

NEXT
RETURN


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   THE USER DEFINED MAIN PROGRAM ENDS HERE   %%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%   POSSIBLE USER'S SUBROUTINES CAN  BE APPENDED  %%%%%%%%%
%%%%%   AFTER THE SCANCOUNT COMMAND  GIVEN BELOW      %%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%    SCANCOUNT TERMINATES THE COMPUTATION AND                                  %
%    MUST BE THE FIRST COMMAND AFTER THE USER'S                                %
%    MAIN PROGRAM                                                              %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
LABEL SCANCOUNT
RESMEMADDRESS=RSAPM(RESTARTADD)
CHANNEL1=-1 CHANNEL2=-1
INITIALIZE ACCUMULATOR
LOAD IREG
STORE OREG
STROBE IREG
GOTO LASTCOMMAND
%
```

A1/45

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                  %
%    POSSIBLE USER'S SUBROUTINES CAN  START HERE. THE MOST COMMON  %
%    FORM IS A SUBROUTINE CONTAINING A LIST OF SUBROUTINE CALLS    %
%                                                                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
%     EVERYTHING WRITTEN BY THE USER MUST END HERE          %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
%     SUBROUTINE REMOTE1                                    %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE REMOTE1
RSAPB(IMPULSEPRODUCTS)=RSAPB(MARGIN1)
RSAPM(RESJUMPER)=RSAPM(MAXLAG1)
CONTINUE

NEXT
RSAPB(LAGMAX)=RSAPB(LAGMAX1)
RSAPM(MAXLAG)=RSAPM(MAXLAG1)
LC2=LCR2
CONTINUE

NEXT
RSAPB(IMPULSEGATES)=RSAPB(SIGSAMPLES1)
RSAPM(RESJUMPER)=RSAPM(RESJUMPER)+RSAPM(APMINCR)
IF (LC2=0) THEN CONTINUE ELSE CALL SCATTERCOMPUTE

NEXT
RSAPB(IMPULSEPRODUCTS)=RSAPB(CALPRODUCTS1)
CONTINUE

NEXT
RSAPB(IMPULSEGATES)=RSAPB(CALGATES1)
GOTO BOXCARACF
%
```

A1/46

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%                                                              %
%     SUBROUTINE REMOTE2                                       %
%                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%
NEXT
SUBROUTINE REMOTE2
RSAPB(IMPULSEPRODUCTS)=RSAPB(MARGIN2)
RSAPM(RESJUMPER)=RSAPM(MAXLAG2)
CONTINUE

NEXT
RSAPB(LAGMAX)=RSAPB(LAGMAX2)
RSAPM(MAXLAG)=RSAPM(MAXLAG2)
LC2=LCR2
CONTINUE

NEXT
RSAPB(IMPULSEGATES)=RSAPB(SIGSAMPLES2)
RSAPM(RESJUMPER)=RSAPM(RESJUMPER)+RSAPM(APMINCR)
IF (LC2=Ø) THEN CONTINUE ELSE CALL SCATTERCOMPUTE

NEXT
RSAPB(IMPULSEPRODUCTS)=RSAPB(CALPRODUCTS2)
CONTINUE

NEXT
RSAPB(IMPULSEGATES)=RSAPB(CALGATES2)
GOTO BOXCARACF

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                              %
%       SUBROUTINE SCATTERCOMPUTE                              %
%                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE SCATTERCOMPUTE
RELOAD LCR1
RELOADVALUE=RSAPB(IMPULSEPRODUCTS)
LC2=LC2-1         % IMPULSEPRODUCTS = MARGIN
CALL DUMMY

NEXT
LC1=LCR1
IF (LC2=Ø) THEN CONTINUE ELSE GOTO SIGNALGATE

NEXT
IF (LC1=Ø) THEN GOTO SIGNALGATE ELSE CONTINUE
%
%
```

A1/47

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                            %
%                                                            %
%      PROGRAM BLOCK TIMINGCHECK                             %
%                                                            %
%      LC3=SAMPLE COUNTER                                    %
%                                                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
LABEL TIMINGCHECK
BUFFERADDRESS=RSAPB(IMPULSEPRODUCTS)+RSAPB(IMPULSEPRODUCTS)
QAPB=BUFFERADDRESS                  % IMPULSEPRODUCTS = MARGIN
CONTINUE

NEXT
RELOAD LCR3
RELOADVALUE=QAPB+RSAPB(IMPULSEGATES)
CALL DUMMY                          % IMPULSEGATES = SIGSAMPLES

NEXT
LC3=LCR3
QAPM=RSAPM(RESTARTADD)-RSAPM(APMINCR)
QAPB=RSAPB(BUFSTARTADD)-RSAPB(APBINCR)
CONTINUE

NEXT
LC3=LC3-1
IF (LC3=0) THEN RETURN ELSE GOTO ENTERPOWER1

%CONTINUES AFTER THE GEN-DUMP ROUTINE

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
%   PROGRAM GEN-DUMP                                        %
%                                                           %
%   THIS PROGRAM CONTROLS THE DATA TRANSFER   AND           %
%   MUST START FROM THE LOCATION 32                         %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%
LOCATION=32
RELOAD LCR1
RELOADVALUE=DATAI
PREPARETRANSFER
CONTINUE

NEXT
PREPARETRANSFER
CONTINUE
```

A1/48

```
NEXT
TRANSFER STATUSWORD
QAPM=-RSAPM(APMINCR)
LC1=LCR1
CONTINUE

NEXT
TRANSFER CONTROLWORD
CONTINUE

NEXT
LABEL TRANSFERLOOP
LC1=LC1-1
RESMEMADDRESS=QAPM+RSAPM(APMINCR)
QAPM=RESMEMADDRESS
QAPB=RSAPB(APBINCR)+RSAPB(APBINCR)
TRANSFER CHANNEL1 MSPART
CONTINUE

NEXT
RESMEMADDRESS=QAPM
TRANSFER CHANNEL1 LSPART
RELOAD LCR2
RELOADVALUE=QAPB+RSAPB(APBINCR)
CONTINUE

NEXT
RESMEMADDRESS=QAPM
TRANSFER CHANNEL2 MSPART
CONTINUE

NEXT
RESMEMADDRESS=QAPM
TRANSFER CHANNEL2 LSPART
LC2=LCR2
IF (LC1=0) THEN CONTINUEANDPUSH ELSE GOTO TRANSFERLOOP

NEXT
LABEL FINISHLOOP
LC2=LC2-1
FINISHTRANSFER
SET START-EXPERIMENT MODE
IF (LC2=0) THEN GOTO ZERO ELSE GOBACK
```

A1/49

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%     THE LAST TWO COMMANDS OF THE TIMINGCHECK                                 %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

NEXT
LABEL ENTERPOWER1
BUFFERADDRESS=QAPB+RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=QAPM+RSAPM(APMINCR)
QAPM=RESMEMADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=Y*X-X*Y
INITIALIZE ACCUMULATOR
STROBE IREG
LOAD IREG
STORE OREG
LC3=LC3-1
IF (LC3=0) THEN CONTINUE ELSE GOTO ENTERPOWER1

NEXT
RSAPM(RESTARTADD)=QAPM+RSAPM(APMINCR)
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(IMPULSEPRODUCTS)
GOTO SIGNALGATE
%                    %IMPULSEPRODUCTS = MARGIN



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%       SUBROUTINE DUMMY   (MODIFIED)                                          %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE DUMMY
QAPB=RSAPB(BUFSTARTADD)-RSAPB(APBINCR)
RETURN
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%     PROGRAM BLOCK SIGNALGATE                                                 %
%                                                                              %
%     LC1=LAGCOUNTER                                                           %
%     LC2=GATE COUNTER                                                         %
%     LC3=SAMPLECOUNTER    (COUNTS SIGSAMPLES - LAGMAX)                        %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
```

A1/50

```
%
NEXT
LABEL SIGNALGATE
RELOAD LCR1
RELOADVALUE=RSAPB(LAGMAX)
CALL DUMMY

NEXT
LC1=LCR1
RELOAD LCR3
RELOADVALUE=RSAPB(IMPULSEGATES)-RSAPB(LAGMAX)
CALL DUMMY          % IMPULSEGATES = SIGSAMPLES

NEXT
RSAPB(IMPULSEPRODUCTS)=RSAPB(IMPULSEPRODUCTS)+RSAPB(APBINCR)
LCR1A=LC1           % IMPULSEPRODUCTS = MARGIN+1 FOR COMING USE
LC3=LCR3
LC2=LCR2
IF (LC2=0) THEN GOTO LAGDECREASE  ELSE CONTINUEANDPUSH

NEXT
RSAPB(BUFSTARTADD)=QAPB+RSAPB(APBINCR)
IF (LC2=0) THEN RETURN ELSE GOTO LAGDECREASE
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                               %
%     BROGRAM BLOCK SCATTERACF                                                  %
%                                                                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
LABEL SCATTERACF
BUFFERADDRESS=RSAPB(BUFSTARTADD)
QAPB=BUFFERADDRESS
RESMEMADDRESS=RSAPM(RESTARTADD)
QAPM=RESMEMADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=Y*X-X*Y
INITIALIZE ACCUMULATOR
STROBE IREG
LOAD IREG
STORE OREG
IF (LC1=0) THEN LC1=LCR1  LC2=LC2-1 ELSE LC1=LC1-1
IF (LC1=0) THEN GOTO LASTLONG ELSE CONTINUE
```

A1/67

```
NEXT
LABEL SIGLOOP
BUFFERADDRESS=QAPB+RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=QAPM+RSAPM(APMINCR)
QAPM=RESMEMADDRESS
CHANNEL1=OLDVALUE*X+OLDVALUE*Y
CHANNEL2=OLDVALUE*X-OLDVALUE*Y
INITIALIZE ACCUMULATOR
STROBE IREG
LOAD IREG
STORE OREG
IF (LC1=0) THEN LC1=LCR1A ELSE LC1=LC1-1
IF (LC1=0) THEN CONTINUE ELSE GOTO SIGLOOP

NEXT
LC1=LC1-1
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(APBINCR)
IF (LC3=0) THEN CONTINUE ELSE GOTO LAGDECREASE

NEXT
LCR1A=LC1
GOTO SCATTERACF

NEXT
LABEL LAGDECREASE
LC1=LCR1A
LC3=LC3-1
GOTO SCATTERACF

NEXT
LABEL LASTLONG
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(IMPULSEPRODUCTS)
RSAPM(RESTARTADD)=RSAPM(RESTARTADD)+RSAPM(RESJUMPER)
LCR1A=LC1                          %IMPULSEPRODUCTS = MARGIN+1
LC3=LCR3                           %RESJUMPER=MAXLAG+1
IF (LC2=0) THEN RETURN ELSE GOTO LAGDECREASE
```

```
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                            %
%     SUBROUTINE IMPULSE                                     %
%                                                            %
%     PROGRAM BLOCK BOXCARACF                                %
%                                                            %
%     LC1=PRODUCT COUNTER                                    %
%     LC2=LAG COUNTER                                        %
%     LC3=GATE COUNTER                                       %
%                                                            %
%                                                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
NEXT
SUBROUTINE IMPULSE
LABEL BOXCARACF
RELOAD LCR1
RELOADVALUE=RSAPB(IMPULSEPRODUCTS)
CALL DUMMY

NEXT
LC1=LCR1
RELOAD LCR3
RELOADVALUE=RSAPB(IMPULSEGATES)
CALL DUMMY

NEXT
LC3=LCR3
RELOAD LCR2
RELOADVALUE=RSAPB(LAGMAX)
LC1=LC1-1
IF (LC1=0) THEN RETURN ELSE CALL DUMMY

NEXT
LC2=LCR2
LCR1A=LC1
LC3=LC3-1
IF (LC3=0) THEN RETURN ELSE CONTINUE
```

A1/53

```
NEXT
LABEL CALCOMPUTE
BUFFERADDRESS=RSAPB(BUFSTARTADD)
QAPB=BUFFERADDRESS
RESMEMADDRESS=RSAPM(RESTARTADD)
QAPM=RESMEMADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=Y*X-X*Y
INITIALIZE ACCUMULATOR
STROBE IREG
LOAD IREG
STORE OREG
LC2=LC2-1
IF (LC2=0) THEN GOTO PRODUCTTEST ELSE CONTINUEANDPUSH

NEXT
LABEL LAGLOOP
BUFFERADDRESS=QAPB+RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=QAPM+RSAPM(APMINCR)
QAPM=RESMEMADDRESS
CHANNEL1=OLDVALUE*X+OLDVALUE*Y
CHANNEL2=OLDVALUE*X-OLDVALUE*Y
STROBE IREG
LOAD IREG
STORE OREG
LC2=LC2-1
IF (LC2=0) THEN CONTINUEANDPOP ELSE GOBACK

NEXT
LABEL PRODUCTTEST
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(APBINCR)
LC2=LCR2
IF (LC1=0) THEN LC1=LCR1A ELSE LC1=LC1-1
IF (LC1=0) THEN CONTINUE ELSE GOTO CALCOMPUTE

NEXT
LABEL FINISHCALGATE
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(LAGMAX)
RSAPM(RESTARTADD)=QAPM+RSAPM(APMINCR)
LC3=LC3-1
IF (LC3=0) THEN RETURN ELSE GOTO CALCOMPUTE
```

A1/54

```
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
% THE LAST COMMAND TERMINATES THE COMPUTATION AND           %
% THE CORRELATOR GOES TO THE IDLE LOOP                      %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
NEXT
LABEL LASTCOMMAND
STROBE IREG
SET CONTINUE-EXPERIMENT MODE
GOTO ZERO
%
END
```

APPENDIX 2

THE EXPERIMENT GEN-4

ALL FILES EXCEPT GEN-4-R:CLAN

31 PAGES

VERSION: MARCH 1985

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                             %
%      GEN-4:DESC                                             %
%                                                             %
%      TAUNO TURUNEN, EISCAT HQ , MARCH 1985                  %
%                                                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

GEN-4-T IS AN EXPERIMENT WHICH IN MANY RESPECTS RESEMBLES CP-1-C EXPERIMENT USED IN EISCAT SINCE THE AUTUMN 1984 AS A COMMON MODE EXPERIMENT BUT IS PROGRAMMED MORE EFFECTIVELY.

GEOMETRY:
EXPERIMENT IS FIELD ALIGNED AT ABOUT 312 KM. THE COMMON VOLUME IS AT 309 KM ALTITUDE. THE ANTENNA IS KEPT FIXED.

MODULATIONS: GEN-4 IS A "10-CHANNEL" EXPERIMENT:

E-LAYER POWERPROFILES
****

NUMBER OF GATES	:	113
RANGE OF THE FIRST GATE AT CENTER POINT	:	48 KM
GATE SEPARATION	:	3 KM
GATE RESOLUTION (APPROXIMATELY)	:	3 KM
TOTAL RANGE COVERED	:	48-384 KM
GATE COINCIDING WITH THE FIRST PULSECODE GATE	:	15TH
CORRELATOR ADDRESSES (TWO CHANNELS ADDED)	:	0000-0112
PULSE LENGHT	:	19 US
SAMPLING INTERVAL	:	10 US
GATING FACTOR	:	1 (2 GATES ADDED)
RECEIVER FILTERS	:	25 KHZ, LIN
PROGRAMMED STEP RESPONSE TO ASYMPTOTIC LEVEL	:	20 US
CHANNELS USED	:	CH7,CH8
FREQUENCIES USED	:	F4,F5

THE NUMBER OF CROSSPRODUCTS (SQUARES) SUMMED IN A SINGLE GATE IN ONE DUMP IS :

SCANCOUNT*4-3

PULSECODES
****

TWO DIFFERENT PULSECODES ARE USED
PC1, SYSTEM 2:1:4, LAGINCREMENT=4, LAG1=40 US
PC2, SYSTEM 2:1 , LAGINCREMENT=6, LAG1=60 US

ELEMENT PULSES, FILTERING, SAMPLING, GATING, GATE SEPARATION AND GATE RESOLUTION AS IN THE E-LAYER POWERPROFILES

A2/2

NUMBER OF GATES : 60
RANGE TO THE FIRST GATE : 90 KM
TOTAL RANGE COVERED : 90-267 KM

PULSECODE 1 (2:1:4)

CHANNELS USED (ADDED TOGETHER) : CH3,CH4
FREQUENCIES USED : F0,F1
CORRELATOR ADDRESSES : 133-668

DECODING:

LAG	FIRST ADDRESS	FIRST GATE (60 GATES)
X	0133	0133-0206,USED IN BALANCING
1	0207	0211
2	0279	0279
3	0349	0349
4	0417	0423
5	0483	0487
6	0547	0547-0608,OFFSETT (62 POINTS)
7	0609-668	0609-0668

NOISE INJECTION:0113-0117 (GATING=7, 5 POINTS)
SKY NOISE :0118-0132 (GATING=7, 15 POINTS)

NOTE:NOISE INJECTION AND SKY NOISE VALUES HAVE TO BE DIVIDED BY 4 IN ORDER TO OBTAIN THE SAME SCALE AS IN X-PROFILE.

PULSECODE2 (2:1)

CHANNELS USED : CH5,CH6
FREQUENCIES USED : F2,F3
CORRELATOR ADDRESSES : 0689-1003

DECODING:

LAG	FIRST ADDRESS	FIRST GATE (60 GATES)
X	0689	0689-0757,USED IN BALANCING
1	0758	0764
2	0824	0824
3	0887	0887
4	0947-1003	0947-1003, OFFSETT (57 POINTS)

NOISE INJECTION:0669-0673 (GATING=7, 5 POINTS)
SKY NOISE :0674-0688 (GATING=7, 15 POINTS)

NOTE:NOISE INJECTION AND SKY NOISE VALUES HAVE TO BE DIVIDED BY 4 IN ORDER TO OBTAIN THE SAME SCALE AS IN X-PROFILE.

IN BOTH CODES THE NUMBER OF CROSSPRODUCTS SUMMED IN A SINGLE GATE IN ONE DUMP IS:

SCANCOUNT*4-3

A:2/3

F-LAYER POWERPROFILES
****

NUMBER OF GATES(DATA,NOISE INJECTION,BACKGR.)	:	43,5,15
RANGE OF THE FIRST GATE AT CENTER POINT	:	78 KM
GATE SEPARATION	:	12 KM
GATE RESOLUTION (75% CONTRIBUTION)	:	12 KM
SPATIAL WEIGHTING FUNCTION	:	TRIANGLE WITH 24 KM BASE
TOTAL RANGE COVERED	:	78-582 KM
GATE COINCIDING WITH THE FIRST LONGPULSE GATE	:	9TH
CORRELATOR ADDRESSES (TWO CHANNELS ADDED)	:	1004-1046,1047-1051,1052-1066
PULSE LENGHT	:	80 US
SAMPLING INTERVAL	:	10 US
GATING FACTOR	:	7 (8 GATES ADDED)
RECEIVER FILTERS	:	25 KHZ, LIN
PROGRAMMED STEP RESPONSE TO ASYMPTOTIC LEVEL	:	20 US
CHANNELS USED (COMMON WITH THE E-POWERPROFILES)	:	CH7,CH8
FREQUENCIES USED	:	F4,F5

NOTE: THE CALIBRATION PART (NOISE INJECTION, SKY NOISE) IS COMMON WITH THE E-LAYER POWERPROFILE BUT BEFORE USING IT IN CONNECTION WITH THE E-POWERPROFILES , THE VALUES HAVE TO BE DIVIDED BY 4 BECAUSE OF DIFFERENCES IN THE GATING FACTOR.

THE NUMBER OF CROSSPRODUCTS SUMMED IN A SINGLE GATE IN ONE DUMP IS

SCANCOUNT*16-15

LONG PULSES
****

NUMBER OF GATES(DATA,NOISE INJECTION,SKY NOISE)	:	25,2,10
RANGE OF THE FIRST GATE AT CENTER POINT	:	174 KM
GATE SEPARATION	:	24 KM
GATE RESOLUTION (ABOUT 75% CONTRIBUTION)	:	40 KM
SPATIAL WEIGHTING FUNCTION AT ALL LAGS	:	CUTTED TRIANGLE, 78 KM BASE
TOTAL RANGE COVERED	:	174-750 KM
NUMBER OF LAGS	:	21
LAG INCREMENT	:	10 US
CORRELATOR ADDRESSES (TWO CHANNELS ADDED)	:	1067-1591,1592-1633,1634-1840
PULSE LENGHT (USED ALSO AS REMOTE PULSES)	:	350 US
SAMPLING INTERVAL	:	10 US
RECEIVER FILTERS	:	25 KHZ, LIN
PROGRAMMED STEP RESPONSE TO ASYMPTOTIC LEVEL	:	20 US
CHANNELS USED (ADDED TOGETHER)	:	CH1,CH2
FREQUENCIES USED	:	F6,F7

A2/4

WEIGHTING FACTORS:

LAG	WEIGHTING FACTOR
0	1.000
1	1.032
2	1.061
3	1.086
4	1.107
5	1.125
6	1.139
7	1.150
8	1.157
9	1.161
10	1.161
11	1.157
12	1.150
13	1.139
14	1.125
15	1.107
16	1.086
17	1.061
18	1.032
19	1.000
20	0.964

THE LAGS OF THE OBTAINED TARGET ACF (AFTER SKY NOISE SUBTRACTION) HAVE TO BE DIVIDED BY THE GIVEN NUMBERS TO OBTAIN A BOXCAR WEIGHTED TARGET ACF ESTIMATE. THE NUMBERS CAN BE OBTAINED FROM THE FORMULA

WEIGHTING FACTOR= (35-I)*(16+I)/35*16, WHERE

I IS THE LAG INDEX, 35=PULSE DURATION DIVIDED BY SAMPLING INTERVAL AND 16 IS THE VOLUMEINDEX. FOR VARIANCE COMPUTATION ONE NEEDS THE NUMBER OF CROSSPRODUCTS USED AT A GIVEN LAG. IF THE CHANNELS ARE REASONABLY WELL BALANCED, THEN ONE CAN USE AS A NUMBER OF CROSSPRODUCTS AT A GIVEN LAG I IN ONE RANGE GATE IN ONE DUMP

(SCANCOUNT*2-1)*(16+I)

THIS FORMULA APPLIES ALSO FOR THE SKY NOISE AND NOISE INJECTION GATES. NOTE, HOWEVER, THAT WHEN ADDING THE SKY NOISE AND NOISE INJECTION GATES TO OBTAIN THE FINAL ESTIMATE, ONE IS NOT ADDING INDEPENDENT GATES EXCEPT AT ZERO LAG.

REMOTE STATION DATA
****

THE REMOTE STATION VOLUME IS AT 309 KM ALTITUDE AND IS KEPT FIXED.
THE CORRELATOR DUMP CONSISTS OF 5 OVERLAPPING TRIANGULAR WEIGHTED ACF:S WITH 21 LAGS AND THREE BOXCAR WEIGHTED CALIBRATION ACF:S (2 SKY NOISE ESTIMATE AND 1 NOISE INJECTION) IN THIS ORDER. THE 3RD TRIANGULAR WEIGHTED ACF HAS TH E
SIGNAL, TWO NEIGHBORING ONES HAVE BEEN PROGRAMMED TO HAVE ABOUT 1/3 OF THE SIGNAL CONTRIBUTION AT ZERO LAG (USING OVERLAPPING) AND THE FIRST AND FIFTH

A2/5

ACF:S SHOULD BE TOTALLY FREE FROM THE SIGNAL CONTRIBUTION IF THE TIMING IS CORRECT.

THE WEIGHTING FACTOR IN THE SIGNAL ACF IS

WSIG(I)=30-I , WHERE I IS THE LAG INDEX.

THE NUMBER OF THE CROSSPRODUCTS IN EVERY DUMP IS

(SCANCOUNT*2-1)*(30-I)

THE WEIGHTING FACTOR OF THE SKY NOISE ACF IS 300. IF THE TWO SKY NOISE ESTIMATES ARE ADDED TOGETHER, THEN THE WEIGHTING FACTOR IS 600 IN THE RESULTING SUM. THE FACTOR WITH WHICH THIS SUMMED SKY NOISE ESTIMATE HAS TO BE MULTIPLIED BEFORE SUBTRACTING IT FROM THE SIGNAL ACF IS

(SCANCOUNT*2-1)*(30-I)/((SCANCOUNT*4-2)*300), WHICH CAN BE SIMPLIFIED TO

SKYFACTOR = (30-I)/600

THE WEIGHTING FACTOR IN THE NOISE INJECTION GATE IS 300 AND THE NUMBER OF CROSSPRODUCTS SUMMED TOGETHER IN A DUMP IS

(SCANCOUNT*2-1)*300

A2/6

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%       PROGRAM GEN-4
%
%       BY TAUNO TURUNEN
%       EISCAT HQ
%
%       A PROGRAM RESEMBLING CP-1-F
%
%       JAN.1985
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
WRITE-EXP-HEAD,,
WRITE-FILE (TAUNO)GEN-4:DESC
WRITE-FILE (TAUNO)GEN-4-T:ELAN
WRITE-FILE (TAUNO)GEN-4-T:TLAN
WRITE-FILE (TAUNO)GEN-4-T-SUBR:LIST
%
SYNC -120
%
POINT-REF-H 181.6,76.5,309.0
%
SET-ADC-INT 1,10
SET-ADC-INT 2,10
SET-ADC-INT 3,10
SET-ADC-INT 4,10
SET-ADC-INT 5,10
SET-ADC-INT 6,10
SET-ADC-INT 7,10
SET-ADC-INT 8,10
%
SET-LO1 1053.5
%
SET-LO2 1,94.0
SET-LO2 2,93.5
SET-LO2 3,93.0
SET-LO2 4,92.5
SET-LO2 5,92.0
SET-LO2 6,91.5
SET-LO2 7,91.0
SET-LO2 8,90.5
%
SET-BUF-MEM 2,2048     %LP2
SET-BUF-MEM 3,712      %PC1(4-PULSE)
SET-BUF-MEM 4,2760     %PC1(4-PULSE)
SET-BUF-MEM 5,1020     %PC2(3-PULSE)
SET-BUF-MEM 6,3068     %PC2(3-PULSE)
SET-BUF-MEM 7,1318     %E- AND F-LAYER POWERPROFILES
SET-BUF-MEM 8,3366     %E- AND F-LAYER POWERPROFILES
SET-BUF-MEM 1,0000     %LP1, MUST BE THE LAST SET-BUF-MEM COMMAND
%
SYNC 30
```

A2/7

```
%
SET-SIG-ATT X,5
SET-CH-ATT 1,10
SET-CH-ATT 2,10
SET-CH-ATT 3,3
SET-CH-ATT 4,1
SET-CH-ATT 5,1
SET-CH-ATT 6,1
SET-CH-ATT 7,2
SET-CH-ATT 8,1
%
SET-FILTER 1,LIN,25
SET-FILTER 2,LIN,25
SET-FILTER 3,LIN,25
SET-FILTER 4,LIN,25
SET-FILTER 5,LIN,25
SET-FILTER 6,LIN,25
SET-FILTER 7,LIN,25
SET-FILTER 8,LIN,25
%
SET-SIG-PATH ALLX
%
%
LOAD-RA-CON (EXP)GEN-4-T
%
SYNC 20
%
LOAD-CORR (EXP)GEN-4-T:CCOD
%
SYNC 20
%
SET-COR-APB 15,16     %VOLUMEINDEX
SET-COR-APB 14,20     %LPLAGMAX
SET-COR-APB 13,25     %LPNOGATES ,SCATTER DATA
SET-COR-APB 12,4      %UNILAGINCR1
SET-COR-APB 11,7      %UNILAGMAX1
SET-COR-APB 10,148    %UNINOSAMPLES1   (CODE1 =4-PULSECODE)
SET-COR-APB  9,6      %UNILAGINCR2
SET-COR-APB  8,4      %UNILAGMAX2
SET-COR-APB  7,138    %UNINOSAMPLES2   (CODE2=3-PULSECODE)
SET-COR-APB  6,226    %PPNOSAMPLES1    ,E-POWERPROFILE
SET-COR-APB  5,504    %PPNOSAMPLES2    ,F-POWERPROFILE
SET-COR-APB  4,160    %PPNOSAMPLES3    ,PC-CALIBRATIONS
%
SET-COR-APB  0,1      %APBINCR
%
%
SET-COR-APM 15,0      %RESENTRY1, E-POWERPROFILE
SET-COR-APM 14,113    %RESENTRY2, PC1CAL
SET-COR-APM 13,133    %RESENTRY3, PC1
SET-COR-APM 12,669    %RESENTRY4, PC2CAL
SET-COR-APM 11,689    %RESENTRY5, PC2
SET-COR-APM 10,1004   %RESENTRY6, F-POWERPROFILES + POWER CALIBRATIONS
SET-COR-APM  9,1067   %RESENTRY7, LP SCATTER GATES (25)
SET-COR-APM  8,1592   %RESENTRY8, LP NOISE INJECTION (2)
SET-COR-APM  7,1634   %RESENTRY9, LP SKY NOISE (10)
SET-COR-APM  6,1844   %RESENTRY10, SCANCOUNTADDRESS
%
```

A2/8

```
%
SET-COR-APM  3,20     %LPMAXLAG
%
SET-COR-APM  0,1      %APMINCR
%
%
SET-COR-DATAIO 1845
%
%
%                        FOR THE STRUCTURE OF DATA DUMP, SEE THE GEN-4:DESC
%
SYNC 20
%
START-CORR
SYNC 10
%
ENABLE-DMA
%
SYNC 10
%
START-RA-CON 10,1,10
%
SYNC 10
%
DO -1
START-REC
SYNC 1200
STOP-REC
ENDDO
```

A2/9

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%      GEN-4, FILE GEN-4-T:TLAN
%
%      AN ALGORITHM RESEMBLING THE CP-1
%
%      BY TAUNO TURUNEN , EISCAT HQ, NOV. 1984
%
%      GEOMETRICAL PROPERTIES FROM CP-1
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
AT 1 TRANS,CHQPULS
AT 10 RXPROT,LOPROT
AT 50 BEAMON
AT 60 F0
AT 100 RFON              %CODE1,CH3,F0  4-PULSECODE
AT 119 RFOFF,F1
AT 120 RFON              %CODE2,F1,CH4  4-PULSECODE
AT 139 RFOFF,F2
AT 140 RFON              %CODE3,CH5,F2, 3-PULSECODE
AT 159 RFOFF,F3
AT 160 RFON              %CODE4,CH6,F3  3-PULSECODE
AT 179 RFOFF,F0
AT 180 RFON              %4-PULSECODE 2:1:4, LAGINCREMENT 4, LAG1=40 US
AT 199 RFOFF,F1          %3-PULSECODE 2:1  , LAGINCREMENT 6, LAG1=60 US
AT 200 RFON              %ELEMENT PULSES 19 US, RFOFF GAPS 1 US
AT 219 RFOFF,F0
AT 220 RFON
AT 239 RFOFF,F1
AT 240 RFON
AT 259 RFOFF,F2
AT 260 RFON
AT 279 RFOFF,F3
AT 280 RFON
AT 299 RFOFF,F4
AT 300 RFON                %E-POWERPROFILE1,CH7,F4
AT 319 RFOFF,F2
AT 320 RFON
AT 339 RFOFF,F3
AT 340 RFON
AT 359 RFOFF,F5
AT 360 RFON                %E-POWERPROFILE2,CH8,F5
AT 379 RFOFF,F0
AT 380 RFON
AT 399 RFOFF,F1
AT 400 RFON
AT 419 RFOFF,BEAMOFF
AT 470 RXPOFF
AT 520 LOPOFF
AT 570 RECEV
%
SETTCR 615         %615
%
```

A2/10

```
%
AT 20 CH7
AT 80 CH8
AT 100 CH3
AT 120 CH4
AT 140 CH5
AT 160 CH6
%
% THE FIRST PULSEECODEGATE AT 90 KM
% THE FIRST E-POWERGATE AT 48 KM
%
AT 340 CH1,CH2                    % SKY NOISE SAMPLING FOR LONG PULSE CHANNELS
%
% NUMBER OF SAMPLES NEEDED:       PULSECODES 1,2     148
%                                 PULSECODES 3,4     138
%                                 E-POWERPROFILES    226
%                                 CH1,CH2 SKY NOISE 200 (10 GATES)
AT 1515 CH5OFF
AT 1535 CH6OFF
AT 1575 CH3OFF
AT 1595 CH4OFF
AT 2275 CH7OFF
AT 2335 CH1OFF,CH2OFF,CH8OFF,CAL100
%
AT 2400 CH1,CH2
AT 3115 CH1OFF,CH2OFF,CAL0        %72 NOISE INJECTION SAMPLES ,CH1,CH2
%                                 % (2 GATES)
%
% PHASE 2
%
%
SETTCR 3750
AT 1 TRANS
AT 10 RXPROT,LOPROT
AT 50 BEAMON
```

A2/11

```
%
AT 615 CH1
AT 825 CH7
AT 910 CH8
AT 970 CH2              % F-POWERPROFILE GATING 8 SAMPLES
%                       % LP VOLUMEINDEX 16, MAXLAG 20, SAMPLING 10US
%                       % FIRST FPP-GATE AT 78 KM, GATE SEPARATION 12 KM
%                       % P-P RESOLUTION 24 KM, 75% RESOLUTION 12.0 KM, 43 GAT
ES.
%                       % F-POWERPROFILE RANGE COVERAGE 78-594 KM
%                       % FIRST LP-GATE AT 174 KM RANGE, GATE SEPARATION
%                       % 24 KM, 25 GATES, RANGE COVERAGE 174-750 KM.
%                       % 75% RESOLUTION APPROXIMATELY 40 KM.
%                       % PP-SEPARATION 78 KM (VOLUME BOUNDARY SEPARATION)
AT 4260 CH7OFF
AT 4345 CH8OFF          %344 SAMPLES FOR 43 F-POWERPROFILE GATES
%
%
%
AT 5010 CH1OFF
AT 5365 CH2OFF,CAL100
%                       %440 LP-POINTS SAMPLED FOR 25 SCATTER GATES
%
%
AT 6005 CH3,CH4,CH5,CH6,CH7,CH8
AT 6400 ALLOFF,CAL0
%                       %40 SAMPLES OF NOISE INJECTION, CH3,...,CH8
%
AT 6450 CH3,CH4,CH5,CH6,CH7,CH8
AT 7600 STC
AT 7645 ALLOFF          %120 SAMPLES OF SKY NOISE CH3,...,CH8
AT 7685 BUFLIP
%
%
SETTCR 000
AT 12500 REP
```

A2/12

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
%         GEN-4-T-SUBR:LIST                                 %
%                                                           %
%         FOR MASTER PROGRAM GEN-C-L1U2P3:CLAN              %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NEXT
CALL SUBRLIST

NEXT
CALL SUBRLIST

%
%    NOTE: APPEND  THE SUBROUTINE SUBRLIST AFTER THE SCANCOUNT
%          COMMAND IN THE MASTER PROGRAM

NEXT
SUBROUTINE SUBRLIST
RSAPM(RESTARTADD)=RSAPM(RESENTRY9)
RELOAD LCR3
RELOADVALUE=RSAPB(UNILAGMAX2)+RSAPB(UNILAGINCR2)
CALL LONGPULSE               %LP SKY NOISE, 10 GATES

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY8)
RELOAD LCR3
RELOADVALUE=RSAPB(APBINCR)+RSAPB(APBINCR)
CALL LONGPULSE               %LP NOISE INJECTION , 2 GATES

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY7)
RELOAD LCR3
RELOADVALUE=RSAPB(LPNOGATES)
CALL LONGPULSE               %LP SCATTER GATES, 25 GATES

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY3)
RELOAD LCR1
RELOADVALUE=RSAPB(APBINCR)
CALL UNIPROG1                 %PC-1,4-PULSECODE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
RELOAD LCR1
RELOADVALUE=RSAPB(UNILAGMAX1)
CALL POWERPROFILE3           %PC-1 CALIBRATION

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY5)
RELOAD LCR1
RELOADVALUE=RSAPB(APBINCR)
CALL UNIPROG2                %PC-2 ,3-PULSECODE
```

A2/13

```
NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY4)
RELOAD LCR1
RELOADVALUE=RSAPB(UNILAGMAX1)
CALL POWERPROFILE3          %PC-2 CALIBRATION

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY1)
RELOAD LCR1
RELOADVALUE=RSAPB(APBINCR)
CALL POWERPROFILE1          %E-POWERPROFILE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY6)
RELOAD LCR1
RELOADVALUE=RSAPB(UNILAGMAX1)
CALL POWERPROFILE2          %F-POWERPROFILE+CALIBRATION

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY10)   %SCANCOUNTADDRESS
RETURN
%
%   NOTE THE SPECIAL ARRANGEMENT FOR THE SCANCOUNT ADDRESS
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                            %
%        CORRELATOR PROGRAM GEN-C-L1U2P3:CLAN                %
%       *************************************                %
%                                                            %
%    THIS PROGRAM CONTAINS THE FOLLOWING SUBROUTINES:        %
%                                                            %
%                 LONGPULSE                                  %
%                 UNIPROG1                                   %
%                 UNIPROG2                                   %
%                 POWERPROFILE1                              %
%                 POWERPROFILE2                              %
%                 POWERPROFILE3                              %
%                                                            %
%          GEN-SERIES OF CORRELATOR PROGRAMS                 %
%                                                            %
%                 GEN-A-L3P2                                 %
%                 GEN-B-L2U1P2                               %
%                 GEN-C-L1U2P3                               %
%                 GEN-D-U3P2                                 %
%                 GEN-E-L1U1P3                               %
%                 GEN-R-R2I1                                 %
%                                                            %
%     THE PROGRAM NAME SHOWS THE SUBROUTINE STRUCTURE,       %
%     GEN-REMOTE IS A REMOTE  STATION ORIENTED PROGRAM.      %
%                                                            %
%                                                            %
%     PROGRAM DEVELOPMENT: TAUNO TURUNEN                     %
%                          EISCAT HQ , MARCH 1985            %
%                                                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%            TABLE OF INDEXES
%          **********************
%
%            APB STACK REGISTERS
%            ___________________
%
%       ***LONGPULSE APB STACK***
INDEX VOLUMEINDEX=15,LPLAGMAX=14,LPNOGATES=13
%
%       ***UNIPROG1 APB STACK***
INDEX UNILAGINCR1=12,UNILAGMAX1=11,UNINOSAMPLES1=10
%
%       ***UNIPROG2 APB STACK***
INDEX UNILAGINCR2=9,UNILAGMAX2=8,UNINOSAMPLES2=7
%
%       ***POWERPROFILE1 APB STACK***
INDEX PPNOSAMPLES1=6
%
%       ***POWERPROFILE2 APB STACK***
INDEX PPNOSAMPLES2=5
%
%       ***POWERPROFILE3 APB STACK***
INDEX PPNOSAMPLES3=4
%
%       ***TEMPORARY STORAGES USED BY PROGRAM***
INDEX LAGINDEXCOUNTER=3
INDEX BUFFERJUMPER=2       % USED ALSO AS UNILAGINCR REGISTER
INDEX BUFSTARTADD=1
%
%       ***CONSTANT (=1)***
INDEX APBINCR=0
%
```

A2/15

```
%
%                 APM STACK REGISTERS
%                 ___________________
%
%       ***PROGRAMMABLE RESULT MEMORY START ADDRESSES***
INDEX RESENTRY1=15,RESENTRY2=14,RESENTRY3=13,RESENTRY4=12
INDEX RESENTRY5=11,RESENTRY6=10,RESENTRY7=9,RESENTRY8=8
INDEX RESENTRY9=7,RESENTRY10=6,RESENTRY11=5,RESENTRY12=4
%
%       *** LONGPULSE (SAME VALUE AS IN LPLAGMAX)***
INDEX LPMAXLAG=3
%
%       *** TEMPORARY STORAGES USED BY PROGRAM***
INDEX RESULTJUMPER=2
INDEX RESTARTADD=1
%
%       ***CONSTANT (=1)***
INDEX APMINCR=0
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                             %
%    COMPILER SUBSTITUTES THE NAMES WITH THE GIVEN NUMBERS                    %
%                                                                             %
%    EXAMPLES: RSAPB(LAGMAX)=RSAPB(14)                                        %
%              RSAPM(APMINCR)=RSAPM(0)                                        %
%                                                                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%    MAIN PROGRAMS CONTAIN USUALLY A LIST OF SUBROUTINE CALLS.
%    THE MAXIMUM NUMBER OF SUBROUTINE CALLS IS 14.
%
%    THE FOLLOWING SIX DIFFERENT SUBROUTINE CALLS ARE ALLOWED:
%
%    NEXT
%    RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%    RELOAD LCR3
%    RELOADVALUE=(ANY SUITABLE REGISTER COMBINATION)
%    CALL LONGPULSE
%
%
%    NEXT
%    RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%    RELOAD LCR1
%    RELOADVALUE= ( ANY SUITABLE REGISTER COMBINATION)
%    CALL UNIPROG1
%
%
%    NEXT
%    RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%    RELOAD LCR1
%    RELOADVALUE= (ANY SUITABLE REGISTER COMBINATION)
%    CALL UNIPROG2
%
%
%    NEXT
%    RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%    RELOAD LCR1
%    RELOADVALUE= (ANY SUITABLE REGISTER COMBINATION)
%    CALL POWERPROFILE1
%
```

A2/16

```
%
%     NEXT
%     RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%     RELOAD LCR1
%     RELOADVALUE= (ANY SUITABLE REGISTER COMBINATION)
%     CALL POWERPROFILE2
%
%     NEXT
%     RSAPM(RESTARTADD)=RSAPM(RESENTRYN)
%     RELOAD LCR1
%     RELOADVALUE= (ANY SUITABLE REGISTER COMBINATION)
%     CALL POWERPROFILE3
%
%
%     NOTE: IN THE EXAMPLE COMMANDS THE RESENTRYN MUST BE SOME
%           OF THE POSSIBILITIES RESENTRY1,RESENTRY2...RESENTRY11.
%
%     NOTE: THE RSAPM(RESTARTADD)=.... CAN ALSO BE IN THE FORM:
%
%                 RSAPM(RESTARTADD)=0
%
%     NOTE: THE RSAPM(RESTARTADD)=.. SPECIFIES THE FIRST RESULT
%           MEMORY ADDRESS  USED BY THE CALLED SUBROUTINE. THIS
%           ADDRESS IS GIVEN BY THE USER IN THE CORRELATOR APM
%           STACK LOADING. REGISTERS APM(15)-APM(4) HAVE BEEN
%           RESERVED FOR RESENTRIES RESENTRY1....RESENTRY12.
%
%     NOTE: IF THE LINE RSAPM(RESTARTADD)=... IS OMITTED ,THE
%           DEFAULT VALUE IS THE FIRST HIGHER ADDRESS WHICH WAS NOT
%           USED BY THE PREVIOUS SUBROUTINE AND IN CASE OF THE FIRST
%           SUBROUTINE CALL THE DEFAULT VALUE IS  ZERO.
%
%     NOTE: IN LONGPULSE ROUTINE ONE HAS TO LOAD LCR3 TO A VALUE
%           WHICH GIVES THE NUMBER OF GATES. ONE REGISTER IS
%           RESERVED FOR THE PURPOSE AND THE OTHER VALUES HAS TO BE
%           FORMED BY SUITABLE ARITHMETICAL OPERATIONS IN SIMILAR
%           WAY AS IN THE UNIPROG GATING EXPLAINED BELOW.
%
%     NOTE: IN UNIPROG AND POWERPROFILE CALLS ONE HAS TO
%           LOAD LCR1. THIS COMMANDS THE GATING.
%
%           THE VALUE GIVEN IS THE NUMBER OF RANGES ADDED
%           TOGETHER -1. THUS A COMMAND
%
%                  RELOAD LCR1
%                  RELOADVALUE=0
%
%           CAUSES "NORMAL" POWERPROFILE AND UNIPROG
%           BEHAVIOUR. ONLY ONE REGISTER IS FREE
%           AND THUS ONE HAS TO USE REGISTERS IN
%           "CLEVER" WAY.
%
%                   EXAMPLES:
%
%                   RELOADVALUE=0
%                   RELOADVALUE=RSAPB(USERREG)
%                   RELOADVALUE=RSAPB(APBINCR)
%                   RELOADVALUE=RSAPB(UNILAGINCR1)-RSAPB(APBINCR)
```

```
%
%
%     NOTE: THE LAST EXAMPLE, WHEN USED IN
%           CONNECTION WITH UNIPROG CALL, CAUSES THAT
%           THE OUTPUT IS ALWAYS EQUIVALENT TO LAGINCREMENT=1
%           DATA. SO THE ROUTINE ADDS THEN ALTITUDE
%           GATES TOGETHER AND LAGINCREMENT GIVES THE NUMBER
%           OF ADDED GATES. WHEN USED IN CONNECTION WITH
%           UNIPROG AND THE MODULATION IS PULSE CODE,
%           THE LAGINCREMENT DIVIDED BY THE NUMBER OF
%           RANGEGATES ADDED TOGETHER MUST BE AN INTEGER.
%           OTHERWISE THE DATA CANNOT BE DECODED.
%
%
%
%     NOTE: THE SUBROUTINES ARE WRITTEN SO THAT THE
%           FIRST SAMPLE TAKEN FROM THE BUFFER MEMORY
%           IS IMMEDIATELY AFTER THE LAST USED SAMPLE
%           OF THE PREVIOUS SUBROUTINE. THE FIRST
%           SUBROUTINE ASSUMES BUFFER MEMORY START
%           ADDRESS TO BE ZERO.
%
%     NOTE: THE LAST CALLED SUBROUTINE MUST USE THE HIGHEST
%           ADDRESSES OF THE RESULT MEMORY. IF THIS RULE IS
%           VIOLATED, SPECIAL ARRANGEMENTS HAS TO BE DONE FOR
%           GETTING SCANCOUNT INTO A CORRECT ADDRESS.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%    THE FIRST COMMAND IS THE IDLE LOOP    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%       DO NOT TOUCH THIS COMMAND          %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
LOCATION=0
LABEL ZERO
RSAPB(BUFSTARTADD)=0
RSAPM(RESTARTADD)=0
CONTINUE  %THE IDLE LOOP ENDS HERE

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%  THE MAIN PROGRAM DEFINED BY THE USER BEGINS HERE  %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%          GEN-4-T-SUBR:LIST                                                   %
%                                                                              %
%          FOR MASTER PROGRAM GEN-C-L1U2P3:CLAN                                %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NEXT
CALL SUBRLIST
```

```
NEXT
CALL SUBRLIST

%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% THE USER'S MAIN PROGRAM ENDS HERE %%%%%%%%%%%%%%%%%%%%%%
%%%%% POSSIBLE USER'S SUBROUTINES CAN   %%%%%%%%%%%%%%%%%%%%%%
%%%%% START AFTER THE SCANCOUNT COMMAND %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                            %
%     SCANCOUNT TERMINATES THE COMPUTATION AND               %
%     MUST BE THE FIRST COMMAND AFTER THE USERS              %
%     MAIN PROGRAM                                           %
%                                                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NEXT
LABEL SCANCOUNT
RESMEMADDRESS=RSAPM(RESTARTADD)
CHANNEL1=-1 CHANNEL2=-1
INITIALIZE ACCUMULATOR
LOAD IREG
STORE OREG
STROBE IREG
GOTO LASTCOMMAND
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                            %
%      THE SUBROUTINES WRITTEN BY THE USER CAN START HERE    %
%                                                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NEXT
SUBROUTINE SUBRLIST
RSAPM(RESTARTADD)=RSAPM(RESENTRY9)
RELOAD LCR3
RELOADVALUE=RSAPB(UNILAGMAX2)+RSAPB(UNILAGINCR2)
CALL LONGPULSE              %LP SKY NOISE, 10 GATES

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY8)
RELOAD LCR3
RELOADVALUE=RSAPB(APBINCR)+RSAPB(APBINCR)
CALL LONGPULSE              %LP NOISE INJECTION , 2 GATES

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY7)
RELOAD LCR3
RELOADVALUE=RSAPB(LPNOGATES)
CALL LONGPULSE              %LP SCATTER GATES, 25 GATES
```

A2/19

```
NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY3)
RELOAD LCR1
RELOADVALUE=RSAPB(APBINCR)
CALL UNIPROG1                    %PC-1,4-PULSECODE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
RELOAD LCR1
RELOADVALUE=RSAPB(UNILAGMAX1)
CALL POWERPROFILE3               %PC-1 CALIBRATION

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY5)
RELOAD LCR1
RELOADVALUE=RSAPB(APBINCR)
CALL UNIPROG2                    %PC-2 ,3-PULSECODE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY4)
RELOAD LCR1
RELOADVALUE=RSAPB(UNILAGMAX1)
CALL POWERPROFILE3               %PC-2 CALIBRATION

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY1)
RELOAD LCR1
RELOADVALUE=RSAPB(APBINCR)
CALL POWERPROFILE1               %E-POWERPROFILE

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY6)
RELOAD LCR1
RELOADVALUE=RSAPB(UNILAGMAX1)
CALL POWERPROFILE2               %F-POWERPROFILE+CALIBRATION

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY10)   %SCANCOUNTADDRESS
RETURN
%

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%      EVERYTHING WRITTEN BY THE USER MUST END HERE                            %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%                     SUBROUTINE UNIPROG1                                      %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

A2/20

```
NEXT
SUBROUTINE UNIPROG1
RSAPB(LAGINDEXCOUNTER)=0
CONTINUE

NEXT
LC1=LCR1
RELOAD LCR3
RELOADVALUE=RSAPB(UNILAGMAX1)
CALL DUMMY

NEXT
LC3=LCR3
RSAPB(BUFFERJUMPER)=RSAPB(UNILAGINCR1)
CONTINUEANDPUSH

NEXT
LABEL UNICOMPUTE1
RELOAD LCR2
RELOADVALUE=RSAPB(UNINOSAMPLES1)-RSAPB(LAGINDEXCOUNTER)
GOTO UNIENTRY

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                          %
%                      SUBROUTINE UNIPROG2                                 %
%                                                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

NEXT
SUBROUTINE UNIPROG2
RSAPB(LAGINDEXCOUNTER)=0
CONTINUE

NEXT
LC1=LCR1
RELOAD LCR3
RELOADVALUE=RSAPB(UNILAGMAX2)
CALL DUMMY

NEXT
LC3=LCR3
RSAPB(BUFFERJUMPER)=RSAPB(UNILAGINCR2)
CONTINUEANDPUSH

NEXT
LABEL UNICOMPUTE2
RELOAD LCR2
RELOADVALUE=RSAPB(UNINOSAMPLES2)-RSAPB(LAGINDEXCOUNTER)
GOTO UNIENTRY
%
```

A2/21

```
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
%      PROGRAM UNIPROG                                      %
%      LC1=GATINGCOUNTER                                    %
%      LC2=PRODUCT COUNTER                                  %
%      LC3=LAG COUNTER                                      %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

NEXT
LABEL UNIENTRY
CONTINUE

NEXT
LC2=LCR2
QAPB=RSAPB(BUFSTARTADD)-RSAPB(APBINCR)
QAPM=RSAPM(RESTARTADD)
CONTINUE

NEXT
LABEL SAMERESADDRESS
BUFFERADDRESS=QAPB+RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=QAPM
QAPM=RESMEMADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=Y*X-X*Y
IF (LC2=0) THEN GOTO LAGTEST ELSE CONTINUE

NEXT
LABEL PROFILECOMPUTE
BUFFERADDRESS=QAPB+RSAPB(LAGINDEXCOUNTER)
RESMEMADDRESS=QAPM
RSAPM(RESTARTADD)=RESMEMADDRESS
CHANNEL1=OLDVALUE*X+OLDVALUE*Y
CHANNEL2=OLDVALUE*X-OLDVALUE*Y
INITIALIZE ACCUMULATOR
STROBE IREG
LOAD IREG
STORE OREG
LC2=LC2-1
IF (LC1=0) THEN LC1=LCR1 ELSE LC1=LC1-1
IF (LC1=0) THEN CONTINUE ELSE GOTO SAMERESADDRESS

NEXT
BUFFERADDRESS=QAPB+RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=QAPM+RSAPM(APMINCR)
QAPM=RESMEMADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=Y*X-X*Y
IF (LC2=0) THEN CONTINUE ELSE GOTO PROFILECOMPUTE
```

A2/22

```
NEXT
LABEL LAGTEST
QAPB=QAPB+RSAPB(LAGINDEXCOUNTER)
RSAPM(RESTARTADD)=QAPM
IF (LC3=0) THEN GOTOANDPOP UNIFINISH ELSE CONTINUE

NEXT
RSAPB(LAGINDEXCOUNTER)=RSAPB(LAGINDEXCOUNTER)+RSAPB(BUFFERJUMPER)
LC3=LC3-1
GOBACK

% THE PROGRAM STEP "UNIFINISH" IS AFTER THE GEN-DUMP ROUTINE

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                             %
%      SUBROUTINE POWERPROFILE3                                               %
%                                                                             %
%      LC1=GATING COUNTER                                                     %
%      LC2=SAMPLE COUNTER                                                     %
%                                                                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE POWERPROFILE3
RSAPB(LAGINDEXCOUNTER)=0
CONTINUE

NEXT
LC1=LCR1
RELOAD LCR3
RELOADVALUE=0
CALL DUMMY

NEXT
LC3=LCR3
RELOAD LCR2
RELOADVALUE=RSAPB(PPNOSAMPLES3)
GOTOANDPUSH UNIENTRY

%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                             %
%    PROGRAM GEN-DUMP                                                         %
%                                                                             %
%    THIS PROGRAM CONTROLS THE DATA TRANSFER   AND                            %
%    MUST START FROM THE LOCATION 32                                          %
%                                                                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
```

A2/23

```
%
%
LOCATION=32
RELOAD LCR1
RELOADVALUE=DATAI
PREPARETRANSFER
CONTINUE

NEXT
PREPARETRANSFER
CONTINUE

NEXT
TRANSFER STATUSWORD
QAPM=-RSAPM(APMINCR)
LC1=LCR1
CONTINUE

NEXT
TRANSFER CONTROLWORD
CONTINUE

NEXT
LABEL TRANSFERLOOP
LC1=LC1-1
RESMEMADDRESS=QAPM+RSAPM(APMINCR)
QAPM=RESMEMADDRESS
QAPB=RSAPB(APBINCR)+RSAPB(APBINCR)
TRANSFER CHANNEL1 MSPART
CONTINUE

NEXT
RESMEMADDRESS=QAPM
TRANSFER CHANNEL1 LSPART
RELOAD LCR2
RELOADVALUE=QAPB+RSAPB(APBINCR)
CONTINUE

NEXT
RESMEMADDRESS=QAPM
TRANSFER CHANNEL2 MSPART
CONTINUE

NEXT
RESMEMADDRESS=QAPM
TRANSFER CHANNEL2 LSPART
LC2=LCR2
IF (LC1=0) THEN CONTINUEANDPUSH ELSE GOTO TRANSFERLOOP

NEXT
LABEL FINISHLOOP
LC2=LC2-1
FINISHTRANSFER
SET START-EXPERIMENT MODE
IF (LC2=0) THEN GOTO ZERO ELSE GOBACK
```

A2/24

```
%
%   THE TRANSFER ROUTINE ENDS HERE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%   THE LAST COMMAND OF THE UNIPROG ROUTINE                                    %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NEXT
LABEL UNIFINISH
RSAPB(BUFSTARTADD)=QAPB
RETURN
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%                     SUBROUTINE DUMMY                                         %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE DUMMY
RETURN
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%                                                                              %
%      SUBROUTINE POWERPROFILE1                                                %
%                                                                              %
%      LC1=GATING COUNTER                                                      %
%      LC2=SAMPLE COUNTER                                                      %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE POWERPROFILE1
RSAPB(LAGINDEXCOUNTER)=0
CONTINUE

NEXT
LC1=LCR1
RELOAD LCR3
RELOADVALUE=0
CALL DUMMY

NEXT
LC3=LCR3
RELOAD LCR2
RELOADVALUE=RSAPB(PPNOSAMPLES1)
GOTOANDPUSH UNIENTRY
```

```
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                           %
%                                                           %
%      SUBROUTINE POWERPROFILE2                             %
%                                                           %
%      LC1=GATING COUNTER                                   %
%      LC2=SAMPLE COUNTER                                   %
%                                                           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE POWERPROFILE2
RSAPB(LAGINDEXCOUNTER)=0
CONTINUE

NEXT
LC1=LCR1
RELOAD LCR3
RELOADVALUE=0
CALL DUMMY

NEXT
LC3=LCR3
RELOAD LCR2
RELOADVALUE=RSAPB(PPNOSAMPLES2)
GOTOANDPUSH UNIENTRY

%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                          %
%      SUBROUTINE LONGPULSE                                %
%                                                          %
%      THIS SUBROUTINE COMPUTES THE TARGET ACF ESTIMATE    %
%      IN THE WAY THAT THE ABSOLUTE VOLUME BOUNDARIES ARE  %
%      THE SAME AT ALL LAGS                                %
%                                                          %
%      LC1=LAGCOUNTER                                      %
%      LC2=VOLUME INDEX CONTROL COUNTER                    %
%      LC3=GATE COUNTER                                    %
%                                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
NEXT
SUBROUTINE LONGPULSE
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(LPLAGMAX)
CONTINUE

NEXT
LC3=LCR3
RELOAD LCR1
RELOADVALUE=RSAPB(LPLAGMAX)
CALL DUMMY
```

```
NEXT
LC1=LCR1
RSAPM(RESTARTADD)=RSAPM(RESTARTADD)-RSAPM(APMINCR)
RELOAD LCR2
RELOADVALUE=RSAPB(VOLUMEINDEX)
LC3=LC3-1
IF (LC3=0) THEN RETURN ELSE CALL DUMMY

NEXT
LABEL LONGGATE
LC2=LCR2
LCR1A=LC1
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(VOLUMEINDEX)
RSAPM(RESTARTADD)=RSAPM(RESTARTADD)+RSAPM(APMINCR)
CONTINUE

NEXT
RSAPB(BUFFERJUMPER)=-RSAPB(APBINCR)
RSAPM(RESULTJUMPER)=-RSAPM(APMINCR)
CONTINUE

NEXT
LABEL FULLROW
BUFFERADDRESS=RSAPB(BUFSTARTADD)-RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=RSAPM(RESTARTADD)
QAPM=RESMEMADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=Y*X-X*Y
LCR1A=LC1
LC1=LC1-1
LC2=LC2-1
INITIALIZE ACCUMULATOR
STROBE IREG
LOAD IREG
STORE OREG
IF (LC1=0) THEN GOTO TESTWHATTODO ELSE CONTINUE

NEXT
LABEL COLUMNLAG
BUFFERADDRESS=QAPB+RSAPB(APBINCR)
QAPB=BUFFERADDRESS
RESMEMADDRESS=QAPM+RSAPM(APMINCR)
QAPM=RESMEMADDRESS
LC1=LC1-1
CHANNEL1=OLDVALUE*X+OLDVALUE*Y
CHANNEL2=OLDVALUE*X-OLDVALUE*Y
STROBE IREG
LOAD IREG
STORE OREG
IF (LC1=0) THEN CONTINUE ELSE GOTO COLUMNLAG
```

A2/27

```
NEXT
LABEL TESTWHATTODO
LC1=LCR1A
RESMEMADDRESS=QAPM-RSAPM(LPMAXLAG)
QAPM=RESMEMADDRESS
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)-RSAPB(APBINCR)
IF (LC2=0) THEN CONTINUE ELSE GOTO FULLROW

NEXT
LABEL ROWPIECE
LC1=LC1-1
RSAPM(RESULTJUMPER)=RSAPM(RESULTJUMPER)+RSAPM(APMINCR)
RSAPB(BUFFERJUMPER)=RSAPB(BUFFERJUMPER)+RSAPB(APBINCR)
IF (LC1=0) THEN GOTO NEXTGATE ELSE CONTINUE

NEXT
LCR1A=LC1
BUFFERADDRESS=RSAPB(BUFSTARTADD)-RSAPB(APBINCR)
QAPB=BUFFERADDRESS
CHANNEL1=X*X+Y*Y
CHANNEL2=X*Y-Y*X
CONTINUE

NEXT
BUFFERADDRESS=QAPB+RSAPB(BUFFERJUMPER)
QAPB=BUFFERADDRESS
RESMEMADDRESS=RSAPM(RESTARTADD)+RSAPM(RESULTJUMPER)
QAPM=RESMEMADDRESS
CHANNEL1=OLDVALUE*X+OLDVALUE*Y
CHANNEL2=OLDVALUE*X-OLDVALUE*Y
GOTO COLUMNLAG

NEXT
LABEL NEXTGATE
LC1=LCR1
LC2=LCR2
RSAPB(BUFSTARTADD)=QAPB+RSAPB(VOLUMEINDEX)
RSAPM(RESTARTADD)=RSAPM(RESTARTADD)+RSAPM(LPMAXLAG)
LC3=LC3-1
IF (LC3=0) THEN CONTINUE ELSE GOTO LONGGATE

NEXT
RSAPB(BUFSTARTADD)=RSAPB(BUFSTARTADD)+RSAPB(LPLAGMAX)
RSAPM(RESTARTADD)=RSAPM(RESTARTADD)+RSAPM(APMINCR)
RETURN
%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                          %
% THE LAST COMMAND TERMINATES THE COMPUTATION AND          %
% THE CORRELATOR GOES TO THE IDLE LOOP                     %
%                                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
NEXT
LABEL LASTCOMMAND
STROBE IREG
SET CONTINUE-EXPERIMENT MODE
GOTO ZERO

END
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%      GEN-4-R:ELAN
%%
%%      A PROGRAM RESEMBLING  CP-1
%%
%%      TAUNO TURUNEN, EISCAT HQ, MARCH 1985
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
WRITE-EXP-HEAD,,
WRITE-FILE (EXP)GEN-4-R:ELAN
WRITE-FILE (EXP)GEN-4-R:TLAN
WRITE-FILE (EXP)GEN-4-R-SUBR:LIST
WRITE-FILE (EXP)GEN-4:DESC
%
OFFSET-PPD 000
%
POINT-REF-H 181.7,76.5,309,0
%
%
LOAD-RA (EXP)GEN-4-R
%
%
SET-LO1 1053.5
%
SET-SIG-PAT ALLX
%
&KSET-POLAR 94,-6
&SSET-POLAR 106,-18
%
&KSET-SIG-ATT X,3
&SSET-SIG-ATT X,3
%
&KSET-CHAN-ATT 3,3
&SSET-CHAN-ATT 3,7
&KSET-CHAN-ATT 4,3
&SSET-CHAN-ATT 4,4
%
SET-LO2 3,91.0
SET-LO2 4,90.5
%
SET-FILT 3,BU,25
SET-FILT 4,BU,25
%
TRANSFER-NOISE RAD
%
%
```

A2/29

```
LOAD-CORR (EXP)GEN-4-R:CCOD  %MASTER PROGRAM GEN-REMOTE:CLAN
%
SET-CORR-APB 15,-10     % MARGIN1 (OVERLAPPING)
SET-CORR-APB 14,30      % SIGSAMPLES1
SET-CORR-APB 13,20      % LAGMAX1
SET-CORR-APB 12,0       % CALGATES1
SET-CORR-APB 11,0       % CALPRODUCTS1
SET-CORR-APB 10,0       % MARGIN2
SET-CORR-APB 9,0        % SIGSAMPLES2
SET-CORR-APB 8,20       % LAGMAX2
SET-CORR-APB 7,1        % CALGATES2
SET-CORR-APB 6,300      % CALPRODUCTS2
SET-CORR-APB 2,5        % USERREG, NUMBER OF "SIGNAL GATES"
SET-CORR-APB 0,1        % INCREMENT

SET-CORR-APM 15,000     % RESENTRY1, START OF "SIGNAL GATES"
SET-CORR-APM 14,105     % RESENTRY2, THE FIRST SKY NOISE ACF
SET-CORR-APM 13,126     % RESENTRY3, THE SECOND SKY NOISE ACF
SET-CORR-APM 12,147     % RESENTRY4, THE NOISE INJECTION ACF
SET-CORR-APM 3,20       % MAXLAG1
SET-CORR-APM 2,20       % MAXLAG2
SET-CORR-APM 0,1        % INCREMENT
%
SET-CORR-DATAIO 169     % (5+1+1+1)*21+1
%
%
SET-BUF-MEM 3,0
SET-BUF-MEM 4,1070
%
SET-ADC-INT 3,10.       % 10 US SAMPLES
SET-ADC-INT 4,10.
%
%
SYNC -120
%
SYNC 20
START-CORR
%
SYNC  90
START-R-C 10,1,10
%
ENABLE-DMA
SYNC 10
%
DO -1
START-RECORDING-DATA
SYNC 1200
STOP-RECORDING-DATA
ENDDO
```

A2/30

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                              %
%         GEN-4-R:TLAN                                                         %
%                                                                              %
%         TAUNO TURUNEN                                                        %
%         EISCAT HQ, MARCH 1985                                                %
%                                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
AT 1 RECEV
AT 105 CH3,CH4
AT 3300 ALLOFF
%
%  320 SAMPLES TAKEN FOR SKY NOISE
%
AT 3496 CH3       %LEADING EDGES ARRIVE AT 3850 AND 4205
AT 3851 CH4       %SAMPLING STARTS AT 3850-375+21 AND 4205-375+21
AT 4591 CH3OFF
AT 4946 CH4OFF
%
% 110 SAMPLES TAKEN, SIGNAL ADJUSTED TO 30 SAMPLES IN THE MIDDLE
% OFFSETT-PPD SHOULD BE 0000 IN THIS PROGRAM
%
AT 5150 CH3,CH4
AT 8345 ALLOFF,CAL30
%
% 320 SKY SAMPLES TAKEN
%
AT 8400 CH3,CH4
AT 10000 STC
AT 11595 ALLOFF,CAL0
AT 11635 BUFLIP
%
%  320 NOISE INJECTION SAMPLES TAKEN
%
AT 11655 REP
```

A2/31

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%     GEN-4-R-SUBR:LIST
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
NEXT
CALL SUBRLIST

NEXT
CALL SUBRLIST

NEXT
GOTO SCANCOUNT

NEXT
SUBROUTINE SUBRLIST
RSAPM(RESTARTADD)=RSAPM(RESENTRY2)
RELOAD LCR2
RELOADVALUE=0
CALL REMOTE2

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY1)
RELOAD LCR2
RELOADVALUE=RSAPB(USERREG)
CALL REMOTE1

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY3)
RELOAD LCR2
RELOADVALUE=0
CALL REMOTE2

NEXT
RSAPM(RESTARTADD)=RSAPM(RESENTRY4)
RELOAD LCR2
RELOADVALUE=0
CALL REMOTE2

NEXT
RETURN
```