

GENERAL RESEARCH CORP SANTA BARBARA CALIF
SPIRE - A COMPUTER MODEL OF ELECTRO-OPTICAL SENSORS.(U)
NOV 79 H F GILMORE DAA001-
GRC-RM-2212 DNA-3964F-21

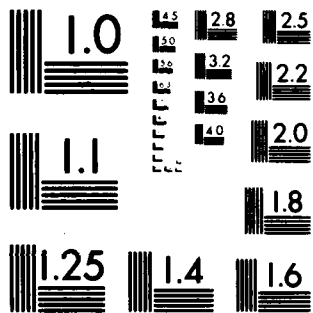
DNA001-78-C-0002

DNA-3964F-21

NL

1 : 1
1 : 1

END
DATE
FILMED
4 80
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

LEVEL III

AD-E 300 694

DNA 3964F-21

SPIRE—A COMPUTER MODEL OF ELECTRO-OPTICAL SENSORS

AD A 082364

General Research Corporation
P.O. Box 6770
Santa Barbara, California 93111

30 November 1979

Final Report for Period 9 November 1978—30 November 1979

CONTRACT No. DNA 001-78-C-0002

APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED.

THIS WORK SPONSORED BY THE DEFENSE NUCLEAR AGENCY
UNDER RDT&E RMSS CODE B322078464 S99QAXHC30101 H2590D.

Prepared for
Director
DEFENSE NUCLEAR AGENCY
Washington, D. C. 20305

DTIC
ELECTE
S MAR 25 1980 D
B

80 3 05 030

DDC FILE COPY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER DNA 3964F-21 ✓	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) SPIRE—A COMPUTER MODEL OF ELECTRO-OPTICAL SENSORS		5. TYPE OF REPORT & PERIOD COVERED Final Report for Period 9 Nov 78—30 Nov 79
		6. PERFORMING ORG. REPORT NUMBER RM-2212 ✓
7. AUTHOR(s) H. F. Gilmore		8. CONTRACT OR GRANT NUMBER(s) DNA 001-78-C-0002 <i>new</i>
9. PERFORMING ORGANIZATION NAME AND ADDRESS General Research Corporation P.O. Box 6770 Santa Barbara, California 93111		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Subtask S99QAXHC301-01
11. CONTROLLING OFFICE NAME AND ADDRESS Director Defense Nuclear Agency Washington, D.C. 20305		12. REPORT DATE 30 November 1979
		13. NUMBER OF PAGES 92
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES This work sponsored by the Defense Nuclear Agency under RDT&E RMSS Code B322078464 S99QAXHC30101 H2590D.		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Infrared Sensors Sensor Computer Models Electro-Optical Sensors		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) SPIRE (Simulation of Passive Infrared Equipment) is a computer program designed to simulate a variety of electro-optical sensors by representing their various elements. These elements include the optics, the detector array, amplifiers, and other linear and non-linear signal processing stages. This report is a user's introduction to the program; SPIRE is described, its use is discussed, and examples of applications are given. <i>✓</i>		

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

CONTENTS

<u>SECTION</u>		<u>PAGE</u>
1	GENERAL DESCRIPTION OF SPIRE	5
	1.1 Introduction	5
	1.2 Characteristics of Electro-Optical System	5
	1.3 SPIRE Representation of Sensors	6
	1.4 Model Characteristics	9
2	STRUCTURE OF SPIRE	11
	2.1 Introduction	11
	2.2 SPIRE Subroutines	12
	2.3 SPIRE Processing Modules	13
	2.4 Subroutine Flow Diagrams	14
3	USE OF SPIRE	24
	3.1 Example of Modeling	24
	3.2 Output Subroutines	32
	3.3 Modifying Block Diagrams	33
	3.4 Units in SPIRE	36
	3.5 Executive Requirements	38
4	FURTHER MODELING EXAMPLES	41
	4.1 Introduction	41
	4.2 Simplifying Block Diagrams	41
	4.3 Measurement Error Analysis	44
	4.4 Modeling Detector Non-Uniformities	46

<u>SECTION</u>		<u>PAGE</u>
5	MODULES AND VARIABLES	52
5.1	SPIRE Processing Modules	52
5.2	Variables in Common Storage	80

ACCESSION for		
NTIS	White Section	☒
DOC	Buff Section	☐
UNANNOUNCED		☐
JUSTIFICATION _____		
BY _____		
DISTRIBUTION/AVAILABILITY CODES		
Dist. AVAIL. and/or SPECIAL		
A		

FIGURES

<u>NO.</u>		<u>PAGE</u>
2.1	Flow Diagram for SPIDER	16
2.2	Flow Diagram for ABLUR	17
2.3	Flow Diagram for a Typical Data Subroutine	18
2.4	Flow Diagram for EXSMO	19
2.5	Flow Diagram for INS	19
2.6	Flow Diagram for a Typical Output Subroutine	20
2.7	Flow Diagram for SPIRE	21
2.8	Flow Diagram for SPSUB1	22
2.9	Flow Diagram for SPSUB2	23
2.10	Flow Diagram for SYSCAL	23
3.1	Example Focal Plane	25
3.2	Example System Block Diagram	25
3.3	Example Block Diagram Using SPIRE Modules	26
3.4	Coordinate System, Detector Array, and Field of View	30
3.5	Listing of a Simple Scan Function	32
3.6	Listing of General-Purpose Output Subroutine	34
3.7	Flow Diagram for General-Purpose Output Subroutine	35
3.8	Listing of Typical Executive Routine	39
4.1	Block Diagram of Optical Sensor (One Channel)	42
4.2	Simplified Representation of Sensor of Fig. 4.1	43
4.3	Sensor Selected for Modeling	45
4.4	Measured Point Target Positions for Sensor of Fig. 4.3	47

<u>NO.</u>		<u>PAGE</u>
4.5	The Effects of Vibration and Extended Targets	48
4.6	Model Detector Array	49
4.7	Block Diagram of Sensor Model	50
5.1	Coordinate Systems and Notation	83

1. GENERAL DESCRIPTION OF SPIRE

1.1 INTRODUCTION

SPIRE (Simulation of Passive Infrared Equipment) is a computer program designed for representing a variety of electro-optical sensors. It may be used alone or imbedded in a larger simulation such as ROSCOE. The purpose of SPIRE is to determine the output of the electro-optical sensor when aimed at a particular field of view, using a detailed model of the electro-optical system. SPIRE is designed so that detailed models of a variety of electro-optical systems can be easily produced.

1.2 CHARACTERISTICS OF ELECTRO-OPTICAL SYSTEMS

Electro-optical sensors to be represented with SPIRE consist of a number of elements. Infrared energy is collected by an optical system. In the image plane of this optical system is an array of radiation detectors. When the optical system scans, energy from different parts of the object field falls on the detectors. (In the case of "staring" sensors employing mosaic arrays, only changes in the object field result in changes in detector outputs.)

As this energy varies, the electrical outputs of the detectors vary. These varying electrical outputs are amplified and processed in various ways by electronic circuits. For example, band-pass filtering may be used to reduce noise, and thresholding may be used to prevent the sensor from responding to background structures fainter than desired targets. The result of this signal processing is a set of electrical signals at the output of the signal processing electronics. These signals contain information about the location of targets (or target detections) in object space, and usually also the amplitudes of the signals associated with each target, which can be used to deduce the aperture irradiance produced by the target. Because these sensors do not measure range, the location information usually consists of the angular coordinates of targets with respect to a reference direction

(the boresight) fixed in the sensor field of view, sometimes together with the time when the target coordinates were measured.

This target location may be used in various ways. For example, it can be used to direct the sensor boresight to track a particular target, or it may be transmitted to a central location for surveillance.

1.3 SPIRE REPRESENTATION OF SENSORS

SPIRE represents such electro-optical sensors, starting with the point where the optical image is formed, through the point where the angular target location information is derived. In considering what has to be represented by SPIRE we must consider the various effects in electro-optical sensors of the type to be modeled which cause the measured target locations and intensities to differ from the "true" values. These errors result from the simultaneous effects of a number of factors.

The optics produce an extended image of a point target, due to aberrations and diffraction. This in itself makes it difficult to measure target location and intensity. The detectors occupy non-zero areas in the image plane, which adds to target location uncertainty. Energy from a particular point target may fall on only one detector element or may be shared by two or more. In the latter case it is important that it be recognized that the detector outputs are all due to just one target. The signals out of the detectors are perturbed by noise generated by the detectors themselves or in preamplifier stages, or noise which is part of the photon stream. The scanning process may be non-linear or have irregularities. If the analog detector outputs are converted to sampled digital form, quantization errors occur. Finally, the data processing stages which have the purpose of rejecting false targets and accurately measuring the location and intensities of true targets may not be optimal; that is, they may not function perfectly due to factors not taken into account by the designer, or imposed by the targets themselves, their backgrounds, and noise in the system.

2 SPIRE is an effort to model electro-optical systems taking these and similar effects into account. In general terms, the computer program represents each of the effects discussed above by creating within the computer an analog of the actual system being modeled and an analog of the optical and electrical signals as they pass through the electro-optical sensor.

We start with a representation of object space, and describe this by projecting it on a primary image plane. In the program this plane is represented by a 100 x 100 array of intensity values. If a point target were seen against a uniform background, all the points but one would have the same value, representing the background, while one point would have a value representing the target.

Since in the actual system the energy from object space first passes through the sensor optics, the first step in the program is to call an optics subroutine (program module) which represents the effect of the optics blur and the detector area. This effect is to spread the energy from each point in the primary image plane among adjacent points. The results of this can be called the blurred image plane.

The next thing that happens in the system is the scan of the array of detectors through the image plane and the conversion of space-varying image plane irradiances into time-varying electrical detector outputs. In the program a module called SCAN takes each detector element in sequence, and moves it in small steps through the blurred image plane. At each step the detector output at that time is computed. This output depends on which points of the blurred focal plane are encompassed by the area of the detector at that time step, as well as the responsivity of the detector.

Note that as a result of this step the detector outputs -- which in the actual sensor being modeled are time-varying electrical wave-

forms -- are represented in the program as a sequence of numbers corresponding to samples of the waveform at closely spaced times.

In the actual sensor the detector-preamplifier output waveforms are distorted because of photon noise and detector noise. Therefore in the program a photon-noise module is used next to introduce the effects of photon noise, by randomly varying detector output sample values in the correct way, and then a detector noise module is used to introduce in a similar way the effects of detector noise.

After this sequence of operations the computer program contains representations (in the form of sequences of numbers) of the electrical output of each detector for a period of time. This time, which includes many sample periods, is usually the frame time of the sensor, that is, the time required for the actual sensor to cover the field of view represented by the primary image plane.

In the actual sensor, the detector outputs are processed as they occur. In the computer program, which must operate serially, all the detector outputs are generated first, then each output in turn is contaminated with noise, and then run through the representation of the signal processing electronics.

For example, suppose the sensor contained, following each detector, an amplifier, a threshold device, and a peak detector as electronics, with each peak (local absolute maximum) considered a target and its location and amplitude provided as an output. In the computer model, the detector outputs are treated in sequence; the program first calls a module to represent the amplifier, and the first detector's output is smoothed in accordance with the impulse response function of the amplifier. Then the threshold device is represented; the program calls a module which sets all waveform samples equal to zero unless they are larger than the appropriate threshold. The peak detector is represented

by a module which then goes through the samples and finds those which are greater than their nearest neighbors. Finally, the locations and amplitudes of these local maxima, after they have been calculated for each detector, are provided as simulated sensor outputs.

1.4 MODEL CHARACTERISTICS

This model has several characteristics of interest. The first is that it calculates the response of the sensor to a particular object-space configuration and thus it is best suited to run in deterministic simulations. Obtaining statistical data on the effects of various backgrounds, for example, may require a number of runs.

A corresponding aspect is that because the sensor is represented in considerable detail, the effects of signal-processing procedures on target detection and measurement can be examined directly, without any extensive analytical work.

Another aspect which is very important is the generality of such a model. In most electro-optical sensors the same type of functions are performed. For example, optical systems differ in the amount of blur they introduce, but all produce some image degradation. Detector arrays differ in detector size and arrangement, and scan patterns may be linear or circular, or stationary for mosaic systems, but the scanning process itself is the same for any of them. Amplifiers differ in bandwidth, but otherwise their effects on an electrical signal are the same. Thus the form of the computer model discussed here lends itself to writing general modules representing the optics, the detectors and scan process, amplifiers, threshold devices, and so on. These general modules make use of user-supplied inputs which give their specific characteristics. For example, input data may contain the diameter of the optical blur, the coordinates of the detectors in the array, the scan pattern, the bandwidths of amplifiers, threshold levels, and so forth, and when the appropriate module is called the input data is used

to determine the exact way the signals are processed.

This possibility means that a program like SPIRE can represent a wide variety of different sensors. All the user has to do is provide inputs telling what modules are used in the sensor being represented, and what the proper parameter values are. This gives the program an unusual combination of generality and explicitness.

2 STRUCTURE OF SPIRE

2.1 INTRODUCTION

To summarize the process described in Sec. 1, SPIRE determines the response of an optical sensor to a given image-plane irradiance distribution. The image-plane irradiance is specified at the points in a 100 x 100 array, which provides an input to the model of the sensor. The units used in specifying the entries in the array are watts per square centimeter per steradian ($\text{W}/\text{cm}^2 \cdot \text{sr}$) at the aperture of the system; these values are proportional to the actual image-plane irradiance in W/cm^2 . The first step in the sensor model is to blur this image-plane irradiance distribution to account for the effects of the optical blur and the elemental detector area. Then a set of detector outputs is constructed, corresponding to the detectors in the array. The set of outputs consists of samples of the waveforms generated by scanning the image-plane irradiance distribution with the detector array. To these waveforms are added randomly generated noise due to detector internal noise and, if desired, variations in average photon flux. These waveforms (represented by digital pulse trains) are then processed by digital analogs of the signal processing which follows the detector in the actual sensor being modeled. Target detections and false alarms are calculated for these sample waveforms.

A considerable simplification in the model results from using what might be termed a "frozen frame" concept. In this approach it is assumed that the time required for one frame (the scan by the sensor of its field of view) is short enough that during this time relative motions between the sensor and the external world can be neglected. Hence the position in space of the sensor and all targets and environmental structures is fixed at the beginning of each frame, and the sensor output is that which would result from these fixed spatial relations. This concept should not affect the capability of the model to evaluate sensor

performance, but it does have to be taken into account when sensor outputs are introduced into tracking filters.

2.2 SPIRE SUBROUTINES

The computations in SPIRE are accomplished in a number of subroutines, described briefly here. Flow diagrams for the subroutines are given in subsequent figures (Sec. 2.4). The principal variables used are stored in labeled common; a list of these variables appears in Sec. 5.2.

SPIDER	The initial executive subroutine which sets up the image plane description and the data, scan, and output flags, and calls SPIRE.
ABLUR	Blurs the input image plane to account for the effects of optical blur and elemental detector area.
DATA1 through DATA10	Provide for storage of the input data for specific sensors. Proper subroutine is called depending on the value of the data flag.
EXSMO	Does exponential smoothing for subroutine ABLUR.
INS	Used to insert new signal processing blocks in a system description; called from DATA1 through DATA10 if necessary.
OUT1 through OUT10	Provide for different types of output depending on the sensors being modeled.
SCAN1 through SCAN10	These are function subroutines which provide for the various types of scan associated with different detectors.
SPIRE	The executive subroutine which processes the blocks which describe the sensor in the correct order.

SPSUB1	Contains part of the signal processing modules.
SPSUB2	Contains the remainder of the signal processing modules.
SYSCAL	Calculates the spacing required in the array describing the image plane, based on the characteristics of the optical system.

2.3 SPIRE PROCESSING MODULES

The response of the sensor to the focal-plane image is represented by various modules in subroutines SPSUB1 and SPSUB2. Each of these modules performs an operation which would be done by one block in the block diagram of a sensor. Modeling a sensor consists in drawing a block diagram of it, choosing a module to represent each block, and assigning values to the descriptors of the modules. The processing modules which are available are listed in Table 2.1 and are described in Sec. 5.

The descriptors of the submodules used in the block diagram are stored in array BLK(10,990). For the array element BLK(J,K), K is the number of the block in the block diagram, and J is the index of a descriptor for that particular block. For example, if the fifth block represented a threshold device the following array elements might be specified:

BLK(1,5) = 9. This gives the type number of the block (this and the following input requirements are given in Sec. 5).

BLK(2,5) = 2. This gives the signal processing channel (or branch) upon which this block occurs.

BLK(3,5) = 3.8 This is the threshold value; crossings of this level are to be identified.

BLK(4,5) = +1. This indicates that positive threshold crossings are to be identified.

The assignment of values to the array BLK(J,K) is done in a DATA subroutine. The process of selecting values and preparing a DATA subroutine is done for a particular example in Sec. 3.1.

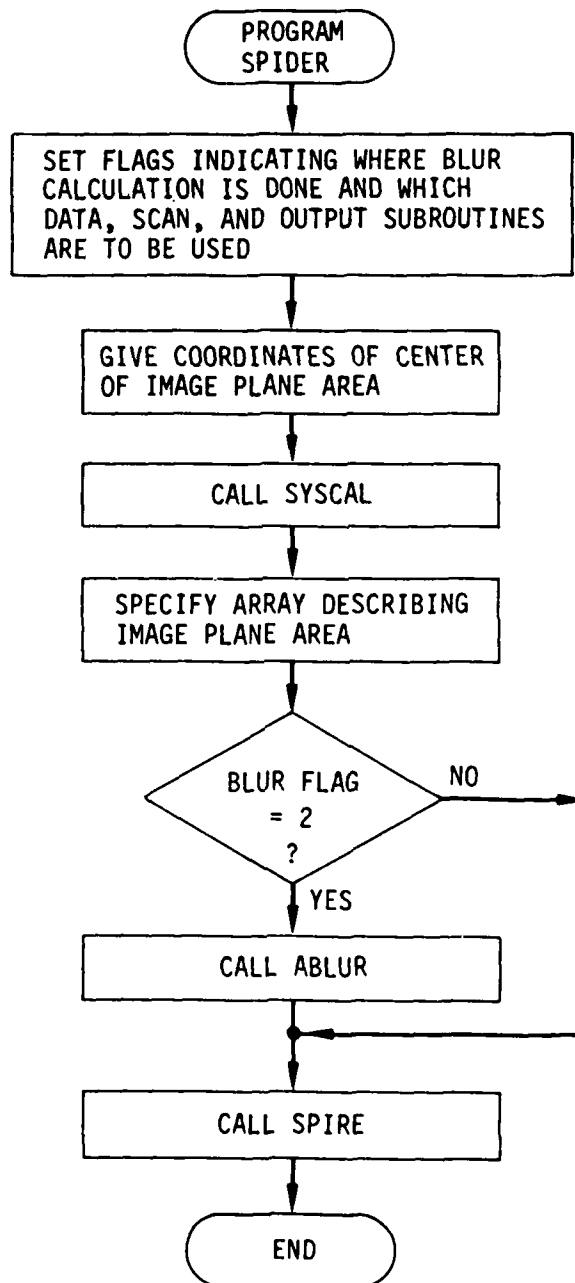
2.4 SUBROUTINE FLOW DIAGRAMS

Flow diagrams for SPIRE subroutines are given in Figs. 2.1 through 2.10.

TABLE 2.1
SPIRE PROCESSING MODULES

<u>TYPE NUMBER</u>	<u>FUNCTION REPRESENTED</u>
1	Optical System
3*	Optics and Detector
4	Scan
5	Detector Noise
6	Gaussian Smoothing
7	Bandpass Smoothing
8	Clipping
9	Threshold Crossing
10	Time Delay
11	Level Adjustment
12	Time Difference Between Threshold Crossings
13	Local Maximum Detection
14	Signal Branching
15	Signal Comparison
16	Average and RMS Calculation
17	Amplification
18	Signal Gating
19	Differentiation
20	Duplication of Blocks
21	General Non-Linear Processing
23*	Lateral Weighting
24	Two-Dimensional Maximum Detection
25	Photon Noise Generation
26	Output Selection
27	General Smoothing
28	Exponential smoothing
29	Absolute Maximum Computation
30	Binary Quantization
31	Display of Signal Waveform
32	High-Pass Filtering
33	Quantization

*Note that types 2 and 22 are not used.



AN-53784

Figure 2.1. Flow Diagram for SPIDER

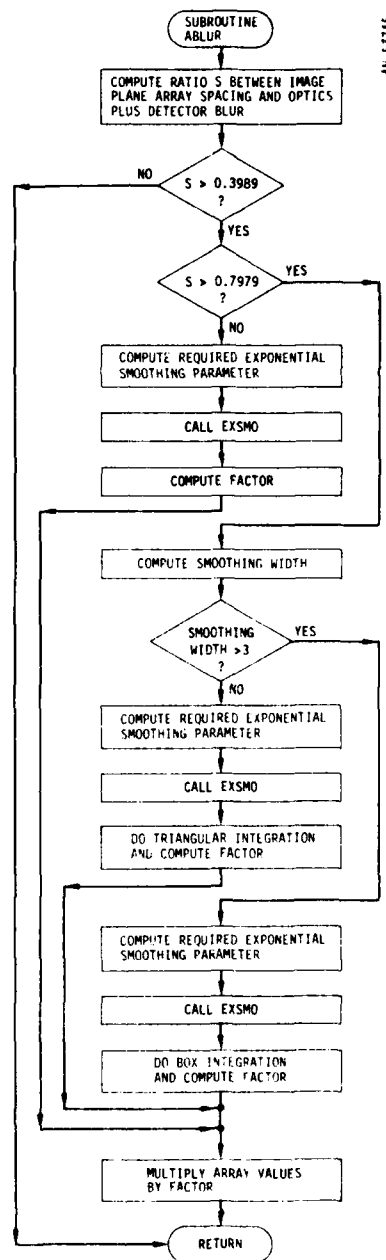


Figure 2.2. Flow Diagram For ABLUR

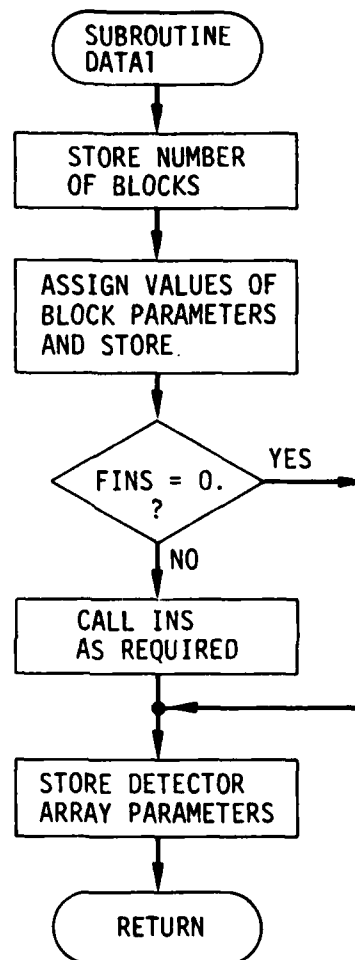
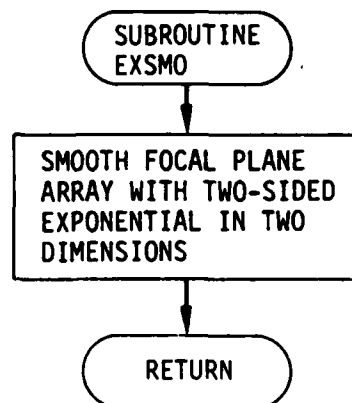
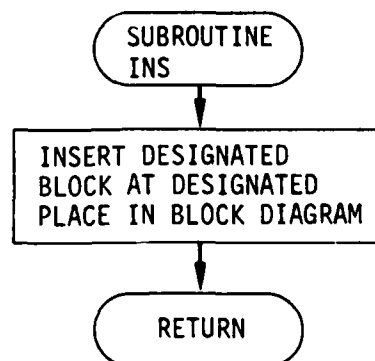


Figure 2.3. Flow Diagram For A Typical Data Subroutine



AN-53787

Figure 2.4. Flow Diagram For EXSMO



AN-53788

Figure 2.5. Flow Diagram For INS

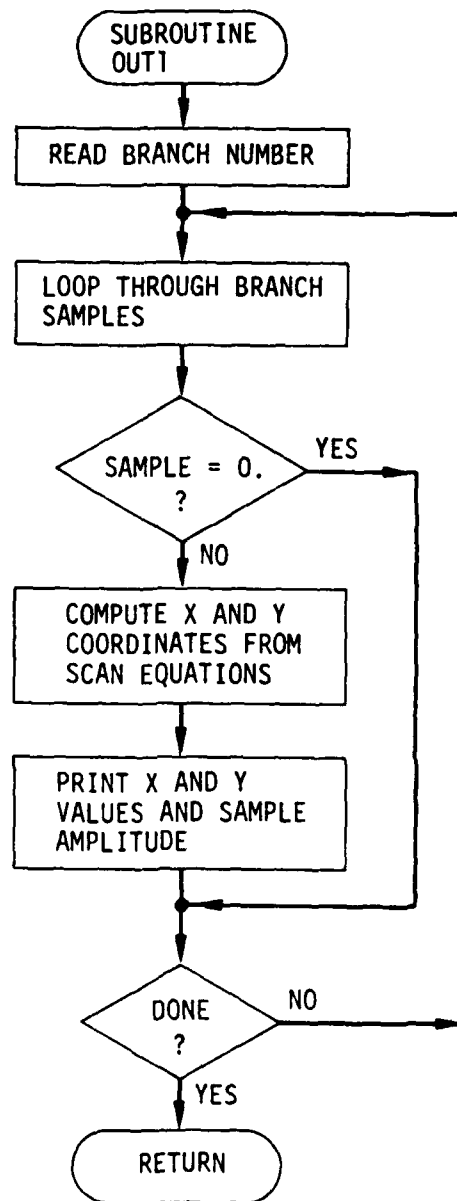


Figure 2.6. Flow Diagram For A Typical Output Subroutine

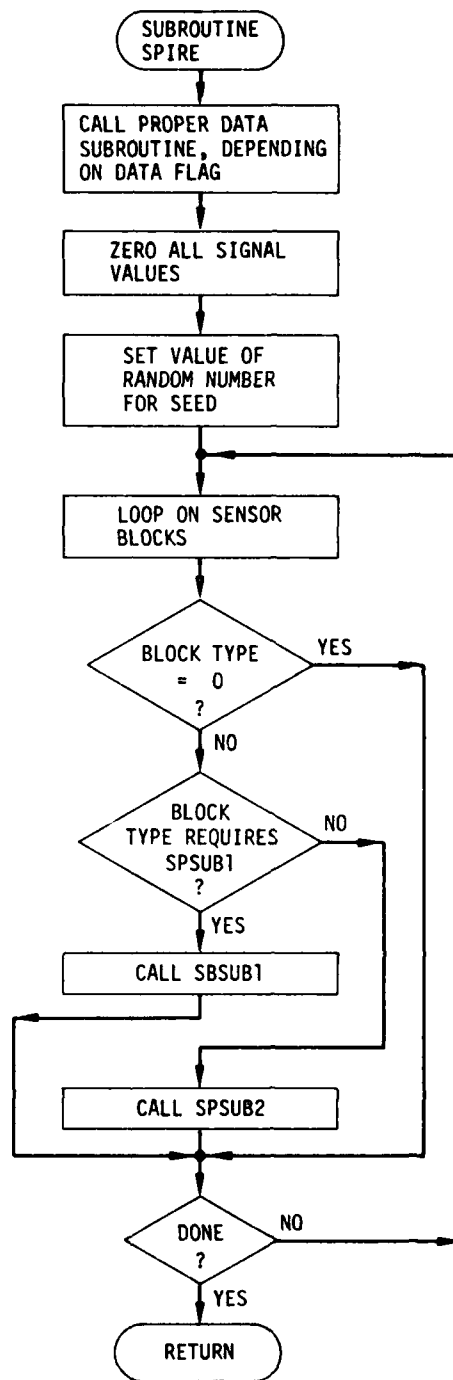
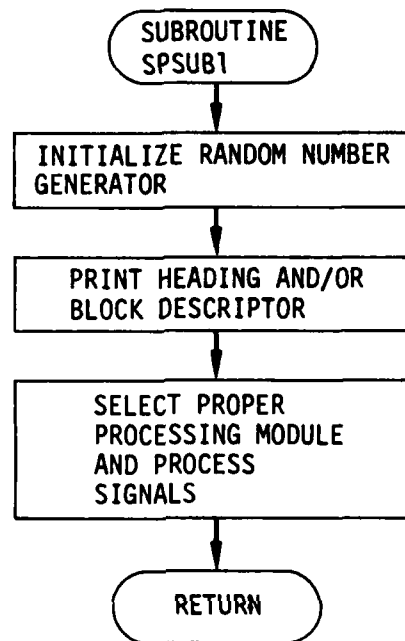


Figure 2.7. Flow Diagram For SPIRE



AN-53791

Figure 2.8. Flow Diagram For SPSUB1

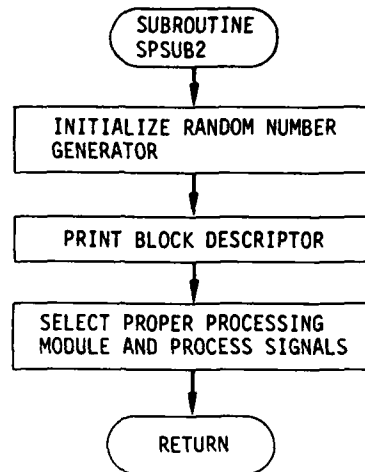


Figure 2.9. Flow Diagram For SPSUB2

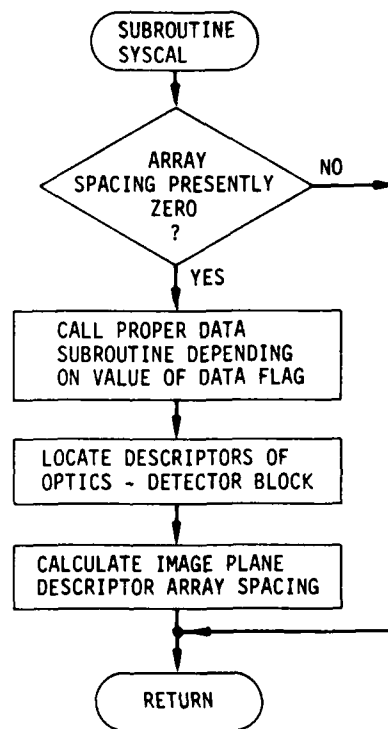


Fig. 2.10. Flow Diagram for SYSCAL

3 USE OF SPIRE

3.1 EXAMPLE OF MODELING

As described earlier, the purpose of SPIRE is to model specific electro-optical sensors. In general, the purpose of the inputs prepared for the program is to provide to the program a description of the sensor being modeled. The nature of the required inputs can best be indicated with a specific example. The following example has been chosen for simplicity; however, much more complex systems are modeled in similar ways. Figure 3.1 suggests a simple sensor in which an array of five infrared detectors, each subtending 1 mrad by 2 mrad in the optical system's focal plane, scans a 10 mrad by 10 mrad field of view, the frame time (time to scan the field) being 0.01 second.

The functional block diagram of the signal processing element is shown in Fig. 3.2. Each of the five detectors is followed by a pre-amplifier, a bandpass filter, and a threshold device, and a target coordinate generator. In operation the detector output signals are amplified in the preamplifier stage, the signal-to-noise ratio is improved in the bandpass filter, and signals exceeding the threshold are passed on to the coordinate generator, which provides as output the focal plane coordinates corresponding to signals exceeding the threshold; these are assumed to be targets.

In modeling this system with SPIRE the block diagram is redrawn as shown in Fig. 3.3. (The names of the blocks and the type numbers appear in Sec. 2, Table 2.1.) The first two blocks will appear in any system; they represent the optics and detector (OPTDET) and the scan (SCAN). The output of the SCAN block will be five trains of pulses representing the detector outputs of the five detectors as the 10 mrad x 10 mrad frame is scanned once by the array.

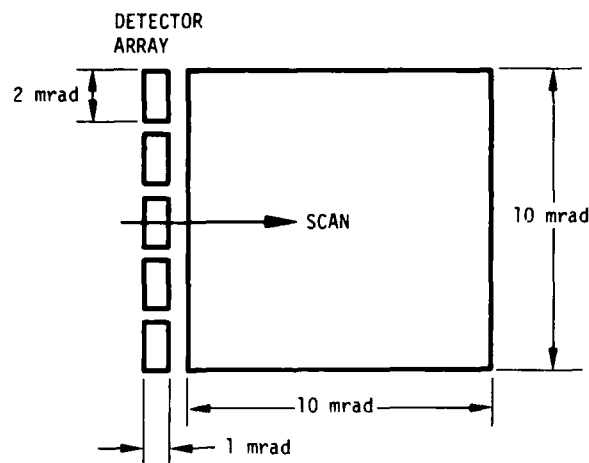


Figure 3.1. Example Focal Plane

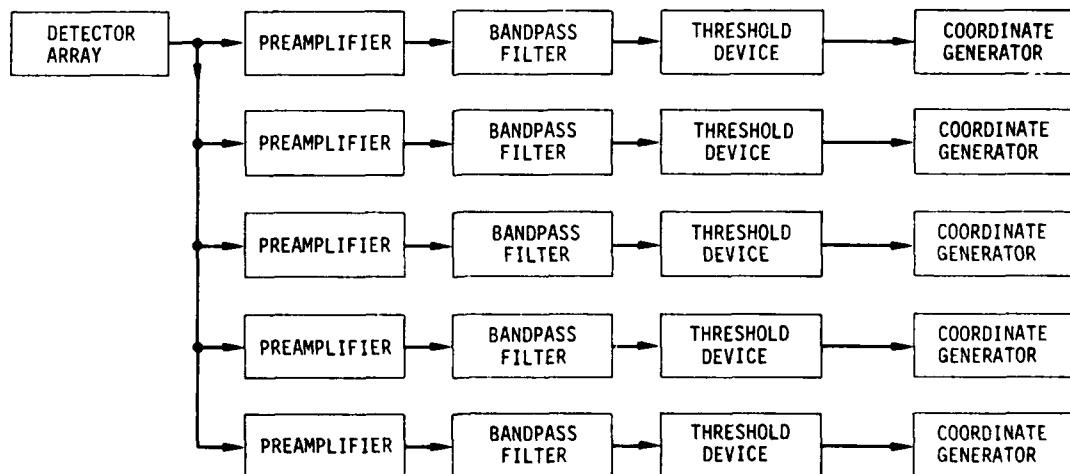
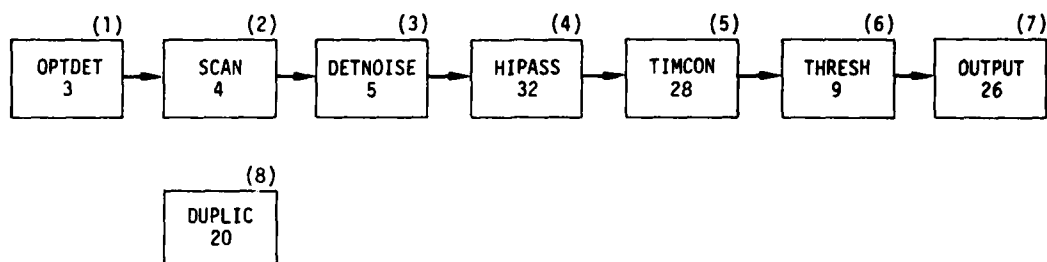


Figure 3.2. Example System Block Diagram



AN-53780

Figure 3.3 Example Block Diagram
Using SPIRE Modules

Following the SCAN block there is, for each detector,¹ a block to generate detector noise. Next come two blocks to represent the bandpass amplifier; these are a high-pass filter (HIPASS) and a low-pass filter (TIMCON). The next block represents the threshold (THRESH), and the coordinate computer is represented in an output block (OUTPUT).

Data declarations giving the characteristics of the block diagram are as follows:

DATA BLKS/28/

```

DATA(BLK(J), J = 011, 019)/3., 1.E-3, 1.E-3, 5., 1.E-12, 1000./
DATA(BLK(J), J = 021, 029)/4., 1., 1., .01, .01/
DATA(BLK(J), J = 031, 039)/5., 1./
DATA(BLK(J), J = 041, 049)/32., 1., .1, 0./
DATA(BLK(J), J = 051, 059)/28., 1., 1.6E-4, 0./
DATA(BLK(J), J = 061, 069)/9., 1., 5., 1./
DATA(BLK(J), J = 071, 079)/26., 1., 0., 0., 0./
DATA(BLK(J), J = 081, 089)/20., 4., 3., 7./

```

These data declarations are provided by the user in the form of a DATA subroutine following a standard format. The desired DATA subroutine

¹The parallel chains of blocks for the five detectors are represented by the duplicator block (DUPLIC), which will be described shortly.

will be called with a data flag in the executive program. The first declaration gives the number of blocks. There are 28 of these: the OPTDET and SCAN blocks, five processing blocks for each of the five detectors, and one for the duplicator block. Note that this is the total number of blocks to be processed in doing the system; the program stops when this number has been processed. The following declarations give the characteristics of each block. The block descriptions in Sec. 5.1 list the characteristics that must be input for each block type. In this example, they are (in the order given in the declarations):¹

Optics and detector (OPTDET)

Type	3.
Detector diameter, radians	$1. \times 10^{-3}$
Optical blur diameter, radians	$1. \times 10^{-3}$
Number of detectors	5.
Noise Equivalent Flux Density, watt/cm ²	$1. \times 10^{-12}$
NEFD Bandwidth, Hz	1000.

NOTE: NEFD Bandwidth is the noise bandwidth used in measuring the NEFD.

Scan (SCAN)

Type	4.
First branch number	1.
Interval between branch numbers	1.
Scan period, seconds	0.01
Scan length, radians	0.01

NOTE: The scan period is the time to scan one frame. The scan length is used in the program to calculate an average scan rate in radians per second.

¹

Note that all values are real, not integer. Units must be as given here and in Sec. 5.1.

Detector noise (DETNOISE)

Type	5.
Branch	1.

NOTE: In duplicating this and future blocks the branch numbers will be changed in the blocks for the other branches.

High-pass filter (HIPASS)

Type	32.
Branch	1.
Time constant, seconds	0.1
Space constant, radians	Not used

NOTE: The transfer function of this filter is $(2\pi jfT)/(1 + 2\pi jfT)$ where f is frequency and T is the time constant. For a time constant of 0.1 second, the 3 dB point is at about 1.6 Hz. The space constant is used only if the time constant value is zero. Conversion from space constant to time constant is made through the scan rate calculated from the data in the SCAN block.

Low-pass filter (TIMCON)

Type	28.
Branch	1.
Time constant, seconds	1.6×10^{-4}
Space constant, radians	Not used

NOTE: The transfer function of this filter is $(1 + 2\pi jfT)^{-1}$ where f is frequency and T is the time constant. For a time constant of 1.6×10^{-4} seconds, the 3 dB point is at about 995 Hz.

Threshold (THRESH)

Type	9.
Branch	1.
Threshold level, volts	5.
Sign	+ 1.

NOTE: The sign should be either +1. or -1. The former indicates that the output is unity when the threshold is crossed from below; -1. indicates that the output is unity when the threshold is crossed from above.

Output (OUTPUT)

Type	26.
Branch	1.

Duplicator (DUPLIC)

Type	20.
Number of duplications	4.
First block to be duplicated	3.
Last block to be duplicated	7.

It is now necessary to describe the detector array and the scan equations. The detector array is described by assigning values to $ARRX(J)$, $ARRY(J)$, $ARRA(J)$, $ARRL(J)$, and $ARRW(J)$. These assignments are made in the same DATA subroutine containing the BLK data statements. $ARRX(J)$ and $ARRY(J)$ are the X and Y coordinates, in radians, of the Jth detector at the time the scan starts. $ARRA(J)$ is the angle of the detector coordinate system with respect to the focal plane coordinate system. (See Sec. 5, Fig. 5.1.) $ARRL(J)$ and $ARRW(J)$ are the length (Y-dimension) and width (X-dimension) of the Jth detector. If either

of these is zero, the corresponding detector dimension is given by the value of Detector Diameter in the Optics and Detector (OPTDET) block.

In order to specify $ARRX(J)$ and $ARRY(J)$ a coordinate system must be set up in the focal plane. Let the coordinate of the lower left-hand corner of the field shown in Fig. 3.4 be (0,0). Then the following values describe the detector array:

$$ARRX(J) = 0 \quad \text{for all } J$$

$$ARRY(1) = 1 \times 10^{-3}$$

$$ARRY(2) = 3 \times 10^{-3}$$

$$ARRY(3) = 5 \times 10^{-3}$$

$$ARRY(4) = 7 \times 10^{-3}$$

$$ARRY(5) = 9 \times 10^{-3}$$

$$ARRA(J) = 0 \quad \text{for all } J$$

$$ARRL(J) = 2 \times 10^{-3} \quad \text{for all } J$$

$$(ARRW(J) = 0 \quad \text{for all } J)$$

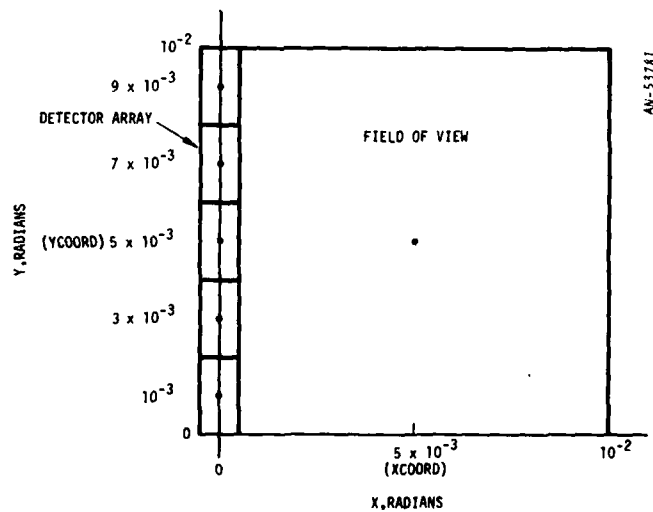


Figure 3.4 Coordinate System, Detector Array, and Field of View.

In addition to this detector array specification it is necessary to relate this focal plane coordinate system to the array SC(I,J) which represents the primary image. This is done by giving the coordinates of the center (SC(50,50)) of the SC array. These values, called XCOORD and YCOORD, must be specified in the driving routine (in the version described here, this is program SPIDER). In this example, $XCOORD = YCOORD = 5 \times 10^{-3}$ radians.

It should be noted that in this particular simulation the field of view does not include all the points in the array SC(I,J). In fact the latter covers an area of 35.4 x 35.4 milliradians; the 10 x 10 milliradian field of view is centered in this area.

The next requirement is to specify the scan equations. The scan equations are given in the FUNCTION SCAN_(X,Y,T,F) subroutines. As used in the program, parameter F indicates whether the function subprogram returns values of the x-coordinate, the y-coordinate, or angle. The parameter T is the time in the scan, and X and Y are the x- and y-coordinates of the detector at the beginning of the scan; they will be specific values of the ARRX(J) and ARRY(J) arrays.

For the particular system being simulated, the function subprogram is shown in Figure 3.5. FUNCTION SCAN4(X,Y,T,F) represents the equations

$$x = T$$

$$y = Y_0$$

$$\alpha = 0$$

where the first equation results because the scan rate is one radian per second, Y_0 represents the initial y-coordinate of the detector, and α represents the angle of the detector with respect to the focal plane coordinate system.

3.2 OUTPUT SUBROUTINES

The output desired from a SPIRE simulation may take many forms: for example, a representation of the electrical signal at one or several points in the circuit, or the location of targets in focal-plane coordinates. The output might in some cases be printed out, or might be transmitted to some other portion of a larger program and used for closing a control loop, for example. In some cases, rather than the target location, the error in target location--i.e. the difference between the true target location and that obtained by the sensor--is desired.

Because of these wide variations, it has not been possible to include all possibilities within the SPIRE program, and it is often necessary for the user to do some programming in order to provide the necessary outputs.

These are two ways of providing outputs in the program. The simplest, used principally for debugging, is the DISPA block, Type 31 (see Sec. 5.1). When the signal passes through this block, J and A(J,BR) are printed out for all values of A(J,BR) greater than the threshold.

The other output method uses the OUTPUT block, Type 26. This block calls one of the OUT subroutines, depending on the value of the output flag. Although there are a few such subroutines in SPIRE, the user may need to write his own and insert it in place of one of the dummy subroutines in the program.

```
Function SCAN4 (X,Y,T,F)
  If (F.EQ.1.) SCAN4 = T
  If (F.EQ.2.) SCAN4 = Y
  If (F.EQ.3.) SCAN4 = 0.
  RETURN
END
```

FIGURE 3.5. Listing of a Simple Scan Function

A general-purpose output subroutine, OUT4, is listed in Fig. 3.6 and flow-charted in Fig. 3.7. This can be used to provide the output for the example scanning system. As shown in the flow diagram, this subroutine loops through the samples on a particular branch. For each non-zero sample¹ the time corresponding to that sample is determined, and the proper scan equations are used to compute the X, Y coordinates of the point. This is actually the location of the center of the detector at that time, and will as a general rule not be the true target location; this of course represents an inherent error in the type of scanning sensor being modeled. Also, because of the quantization of time in the program, there is a quantization error in the scan direction (the X direction). The maximum value of this error is about 0.18 mrad in this particular case.

3.3 MODIFYING BLOCK DIAGRAMS

After a system has been block-diagrammed and represented in a DATA subroutine, it is often required to make changes in the block diagram. Changing some parameters, or substituting a block of one type for a block of another, can be done in a straightforward way by changes in one of the DATA statements.

However, if a block is to be removed or a block is to be inserted between two blocks already in the block diagram, it is desirable not to have to change all the indices in array BLK for all the following blocks. The easiest way to remove a block is to substitute for the type number of the original block the value zero. This causes this block to be counted during execution, but no processing is done. For example, if

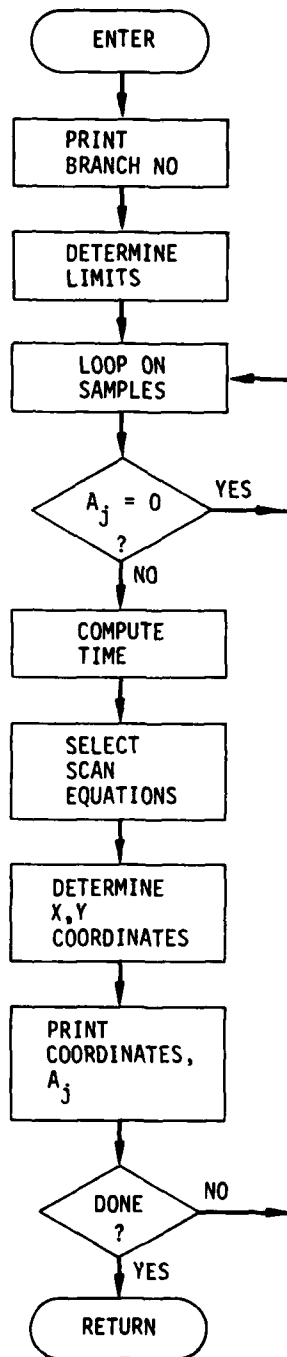
¹ In the system discussed earlier, non-zero samples result when the threshold is crossed from below.

```

SUBROUTINE OUT4
  INTEGER BR,DEL,BR1,BR2,BR3,BKMAX,ARRSR,BLKS
  COMMON/A/A(200,50),B(300),LA(50),L3(50)
  COMMON/B/BLK(10,990),SH(50),SS(50)
  COMMON/C/JBR(50),XOBR(50),YOBR(50)
  COMMON/D/ARRX(50),ARRY(50),ARRL(50),ARRA(50),ARRSR(50),ARRN(50)
  COMMON/H/BR,DEL,BR1,BR2,BR3,BLKS,PI,RANDM,NTYPE
  COMMON/I/SCRA,SAMP,SAMPT,NO,BLUR,BLUR0,TSTART,OSIZE
  COMMON/M/ DATFLG,SCNFLG,OUTFLG,BLRFLG
  COMMON/S/SC(100,100),SCS,XCOORD,YCOORD
26  RR=INT(BLK(2,NO))
  PRINT 2691, BR
2691 FORMAT(/5X*BRANCH*I12)
  L1=LA(BR)
  L2=LB(BR)
  DO 2601 K=L1,L2
  IF(A(K,BR).EQ.0.) GO TO 2601
  T=(K-1)*SAMP+TSTART
  NN=INT(SCNFLG+.1)
  GO TO (1,2,3,4,5,6,7,8,9,10) NN
1  X=SCAN1(ARRX(BR),ARRY(BR),T,1.)
  Y=SCAN1(ARRX(BR),ARRY(BR),T,2.)
  GO TO 2602
2  X=SCAN2(ARRX(BR),ARRY(BR),T,1.)
  Y=SCAN2(ARRX(BR),ARRY(BR),T,2.)
  GO TO 2602
3  X=SCAN3(ARRX(BR),ARRY(BR),T,1.)
  Y=SCAN3(ARRX(BR),ARRY(BR),T,2.)
  GO TO 2602
4  X=SCAN4(ARRX(BR),ARRY(BR),T,1.)
  Y=SCAN4(ARRX(BR),ARRY(BR),T,2.)
  GO TO 2602
5  X=SCAN5(ARRX(BR),ARRY(BR),T,1.)
  Y=SCAN5(ARRX(BR),ARRY(BR),T,2.)
  GO TO 2602
6  X=SCAN6(ARRX(BR),ARRY(BR),T,1.)
  Y=SCAN6(ARRX(BR),ARRY(BR),T,2.)
  GO TO 2602
7  X=SCAN7(ARRX(BR),ARRY(BR),T,1.)
  Y=SCAN7(ARRX(BR),ARRY(BR),T,2.)
  GO TO 2602
8  X=SCAN8(ARRX(BR),ARRY(BR),T,1.)
  Y=SCAN8(ARRX(BR),ARRY(BR),T,2.)
  GO TO 2602
9  X=SCAN9(ARRX(BR),ARRY(BR),T,1.)
  Y=SCAN9(ARRX(BR),ARRY(BR),T,2.)
  GO TO 2602
10 X=SCAN10(ARRX(BR),ARRY(BR),T,1.)
  Y=SCAN10(ARRX(BR),ARRY(BR),T,2.)
2602 CONTINUE
  PRINT 2692, X,Y,A(K,BR)
2692 FORMAT (/5X,3E12.4)
2601 CONTINUE
  RETURN
  END

```

Figure 3.6. Listing of General-Purpose Output Subroutine



AN-53782

Figure 3.7. Flow Diagram for General-Purpose Output Subroutine

(in the example) the declaration

```
DATA (BLK(J), J = 041, 049)/32., 1., .1, 0./
```

were replaced by

```
DATA(BLK(J), J = 041, 049)/0., 1., 1., 0./
```

the effect would be to remove the high-pass filters (block no. 4, type 32) from the system block diagram (Fig. 3.3).

If a block is to be inserted between two blocks in the diagram, the INS subroutine may be used. The first step in doing this is to add a DATA statement containing the characteristics of the block to be inserted; these characteristics should appear in the BLK array after the previously entered data. In the example, if a local maximum block (type 13) were to be inserted before the threshold block on each channel, the data statement could be DATA(BLK(J), J = 091, 099)/13.,1./. Since there is to be another block in each branch, the number of blocks and the duplication block must also be changed: DATA BLKS/28/ becomes DATA BLKS /33/, and DATA(BLK(J), J = 081,089)/20.,4.,3.,7.,/ becomes DATA(BLK(J), J = 081,089)/20.,4.,3.,8./. (Note that no renumbering of this entry is necessary.) A statement CALL INS(6,9) calls the INS subroutine which rearranges the BLK array so that the 9th block is inserted and becomes the 6th block. This call must follow the identification do-loop 111.

3.4 UNITS IN SPIRE

Within SPIRE, nearly all quantities representing distances are measured in radians. This includes detector dimensions, optical blur diameter, coordinates of detectors, scan equations, scan lengths, space constants, and so on. The only quantities involving other measures of length are the collector area ARCOL in Block Type 1, OPTSYS, the units for which are cm^2 , and the noise equivalent flux density NEFD in Block Type 3, OPTDET, the units for which are watt/cm^2 . Collector area is a required input only when photon noise is to be computed using Block Type 25, PHOTN. The NEFD input must be calculated for a square

detector having sides equal to the detector diameter (also input in Block Type 3) taking into account the effect of the optical blur which may cause some of the energy from a point target to fall off the detector.

Signals are carried through SPIRE from the input specification of the focal plane in array SC(100,100) to the output. The spacing between the points in array SC is SCS, in radians. The entries in array SC are optical system aperture irradiance values in $\text{watt/cm}^2 \text{sr}$. When a point object producing an aperture irradiance $H \text{ watt/cm}^2$ is to be represented by specifying a value for a single point in array SC, the value required is $H/(\text{SCS})^2$, since the point object is considered to correspond to an extended target producing an aperture irradiance $H/(\text{SCS})^2 \text{ watt/cm}^2$ over a solid angle of $(\text{SCS})^2 \text{ sr}$.

Within the data processing blocks of SPIRE, amplitudes in the A(J,BR) array represent voltage-type signals on an arbitrary scale. The initial detector outputs are assigned in such a way that when detector noise represented by a random variable having an RMS value of unity is added to each sample, the correct signal-to-noise ratio is obtained for a point target. (This addition of noise occurs in Block Type 5, DETNOISE). Processing subsequent to this will usually change the values of RMS noise and signal amplitudes differently. Therefore in specifying thresholds, clipping levels, and similar quantities it is often necessary to make a few calibration runs with the program. It has been found easier to do this, using inputs containing targets of various irradiances, than to calculate by hand what the thresholds or clipping levels should be. In the same way, to determine the target aperture irradiances corresponding to signals obtained in the output block, one or more calibration runs may be made.

3.5 EXECUTIVE REQUIREMENTS

Although the overall program for the simulation of passive electro-optical systems is called SPIRE, SPIRE is also the name of a subroutine which functions as an executive routine within the program. The overall program SPIRE is designed to be a part of larger simulations, and certain data and instructions must be provided by that simulation or, in the stand-alone mode described in this report, by an initial executive routine. In the version described in this report this initial executive subroutine is SPIDER. This section discusses the data and instructions which must be provided by the simulation which uses SPIRE or by the initial executive subroutine. SPIRE has been written for an operating system in which all storage is zeroed automatically at the beginning of a run.

Figure 3.8 is a listing of SPIDER for the example of Section 3.1. We will discuss the statements in this program in order.

- BLRFLG = 1. BLRFLG has value 1. or 2. When it is 1., subroutine ABLUR, which blurs the focal plane to account for the effects of optics and the elemental detector area, is called before subroutine SPIRE is called (in this case in SPIDER). When BLRFLG = 2., this blurring is done in subroutine SPSUB1. (See the discussion under SCS.)
- DATFLG = 4. DATFLG has values from 1. through 10. Its value determines which DATA subroutine is called for sensor descriptors.
- SCNFLG = 4. SCNFLG has values from 1. through 10. Its value determines which SCAN function is called for the scan equations.

```

PROGRAM SPIDER(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
COMMON/M/ DATFLG,SCNFLG,OUTFLG,BLRFLG
COMMON/S/SC(100,100),SCS,XCOORD,YCOORD
BLRFLG=1.
* IF BLRFLG=1., BLURRING DONE IN SPSUB1
* IF BLRFLG=2., BLURRING DONE IN SPIDER
DATFLG=4.
OUTFLG=4.
SCNFLG=4.
SCS=0.
XCOORD=5.E-3
YCOORD=5.E-3
CALL SYSCAL
SC(50,50) = 2.E-4
IF(BLRFLG.EQ.2.) CALL ABLUR
CALL SPIRE
END

```

Figure 3.8. Listing of a Typical
Executive Routine

OUTFLG = 4.	OUTFLG has values from 1. through 10. Its value determines which OUT subroutine is called to provide outputs.
SCS = 0.	SCS is the interval (in radians) between the samples representing the primary image plane. It can be specified in the initial executive subroutine, but it is normally calculated from the optical blur and the detector area either in subroutine SPSUB1 or in subroutine SYSCAL. Once SCS has been calculated it serves as a flag; when SCS is non-zero, the calculations in either SPSUB1 or SYSCAL are aborted.
XCOORD = 5.E-3 YCOORD = 5.E-3	XCOORD and YCOORD are the X and Y coordinates of the point SC(50,50) in the coordinate system used to specify the detector array and the scan equations.

CALL SYSCAL

When SYSCAL is called it locates the correct block (the OPTDET block, Type 3), reads the optical blur and the elemental detector size, and computes SCS.

$SCS = 1/4(\alpha^2 + \beta^2)^{1/2}$, where α is the diameter of the optical blur and β is the diameter of the elemental detector area. If SCS is non-zero when SYSCAL is called, the computation is not made.

SC(50,50) = 2.E-4 This is the portion of initial executive which specifies the SC array which describes the focal plane. In this case, only the central point is non-zero.

IF (BLRFLG.EQ.2) This statement calls subroutine ABLUR, which blurs the
CALL ABLUR focal plane to account for the effects of the optics and the elemental detector area.

CALL SPIRE This calls the SPIRE executive subroutine which causes the necessary calculations to represent a scan of the focal plane to be made.

4. FURTHER MODELING EXAMPLES

4.1 INTRODUCTION

Although in theory a sensor model such as SPIRE is capable of representing most signal processing networks in complete detail, in practice such networks are usually represented with various simplifications. This is due to two things. The actual signal processing circuitry is often quite complex, and the labor involved in representing it in detail is too great. Also, complex units of the circuitry may perform relatively simple functions, and the questions a program such as SPIRE is required to answer can often be tested by a simple block representing the function being performed rather than the circuitry used to perform it.

4.2 SIMPLIFYING BLOCK DIAGRAMS

An example of this is shown in Figs. 4.1 and 4.2. Figure 4.1 is a block diagram, in an already somewhat simplified form, of the electronics following one detector of the HOST optical sensor. It will be observed that the three blocks in the detector package and the two blocks in the preamplifier circuit have the function of maintaining the linearity and frequency response of the detector as well as providing gain and impedance matching. The sensor blocks in the circumvention circuit have the function of reducing noise due to hard radiation; this network operates in conjunction with the five blocks in the threshold/DC restorer circuit.

In a system simulation employing SPIRE, this optical tracking sensor would typically be represented as shown in Fig. 4.2. This representation is based on the idea that the questions to be answered are not concerned with the effectiveness of the detector and preamplifier circuitry, and that the compensation and threshold/DC restorer circuitry has the effect of introducing noise whose character can be represented as a function of the hard radiation event rate.

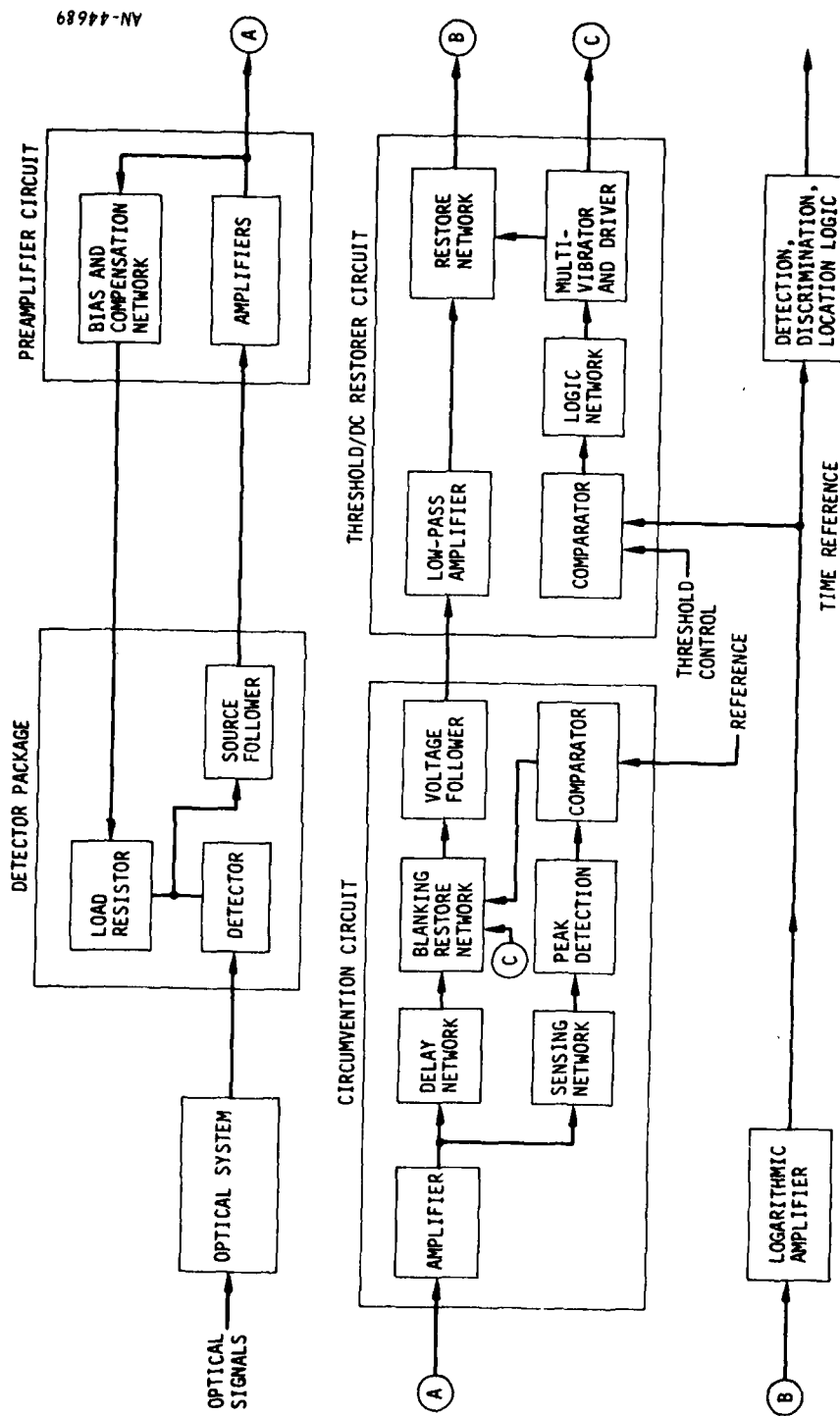


Figure 4.1. Block Diagram of Optical Sensor (One Channel)

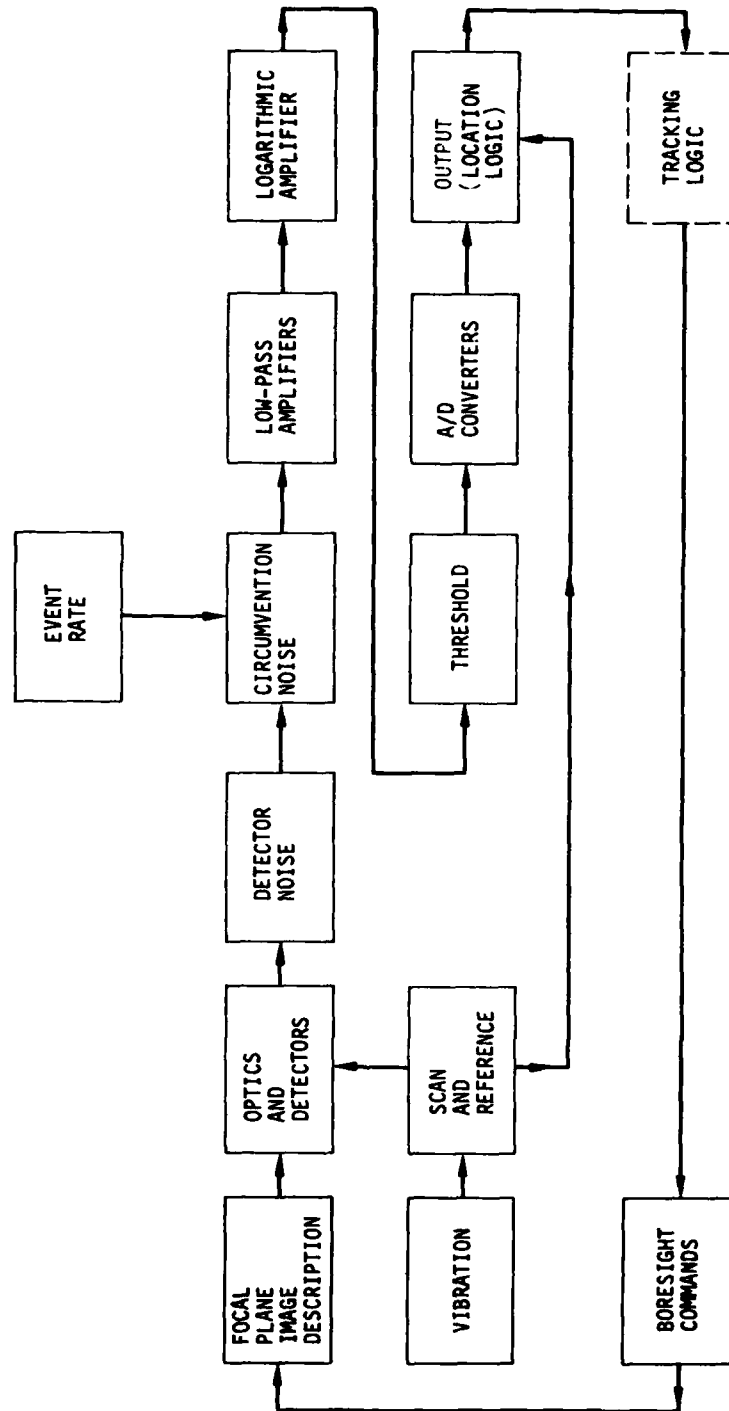


Figure 4.2. Simplified Representation of Sensor of Fig. 4.1.

Also, since this sensor may be used during powered flight, the effect of vibration on the accuracy of the output signals is of interest.

In Fig. 4.2, all of the blocks except EVENT RATE, CIRCUMVENTION NOISE, TRACKING LOGIC, BORESIGHT COMMANDS, and VIBRATION are represented by standard SPIRE modules. EVENT RATE simply represents the input for CIRCUMVENTION NOISE. However, if hard radiation events are to be included, the latter must be provided by the user. TRACKING LOGIC, shown in a dashed box, is external to SPIRE; this function is performed in a larger simulation in which SPIRE is imbedded. TRACKING LOGIC generates BORESIGHT COMMANDS, which determine the distribution of objects in the FOCAL PLANE IMAGE DESCRIPTION. VIBRATION is naturally included by adding random variables in the scan equations to represent the variations in boresight direction caused by vehicle vibration.

4.3 MEASUREMENT ERROR ANALYSIS EXAMPLES

Another example of the use of SPIRE is shown in Fig. 4.3. Here we show a linear array of 10 detectors covering a 22 milliradian field of view, with scanning accomplished by rotating the array with respect to the image of the field of view. (This might be accomplished in various ways, for example, by rotating the body of a simple missile.) The SPIRE modules necessary for representing one electronics channel, which processes the signal from one detector, are also shown. All of the modules shown are available in SPIRE except for the interpolation block. In the simulation this was included in the OUTPUT subroutine; the operation consisted in comparing (one-dimensional) local maxima with the signals on adjacent branches; when the local maximum in question was larger than the two adjacent ones, an interpolation procedure selected a coordinate value in the direction along the array. This process was designed to permit finer resolution along the array than would otherwise be possible with just 10 detectors in the radial direction.

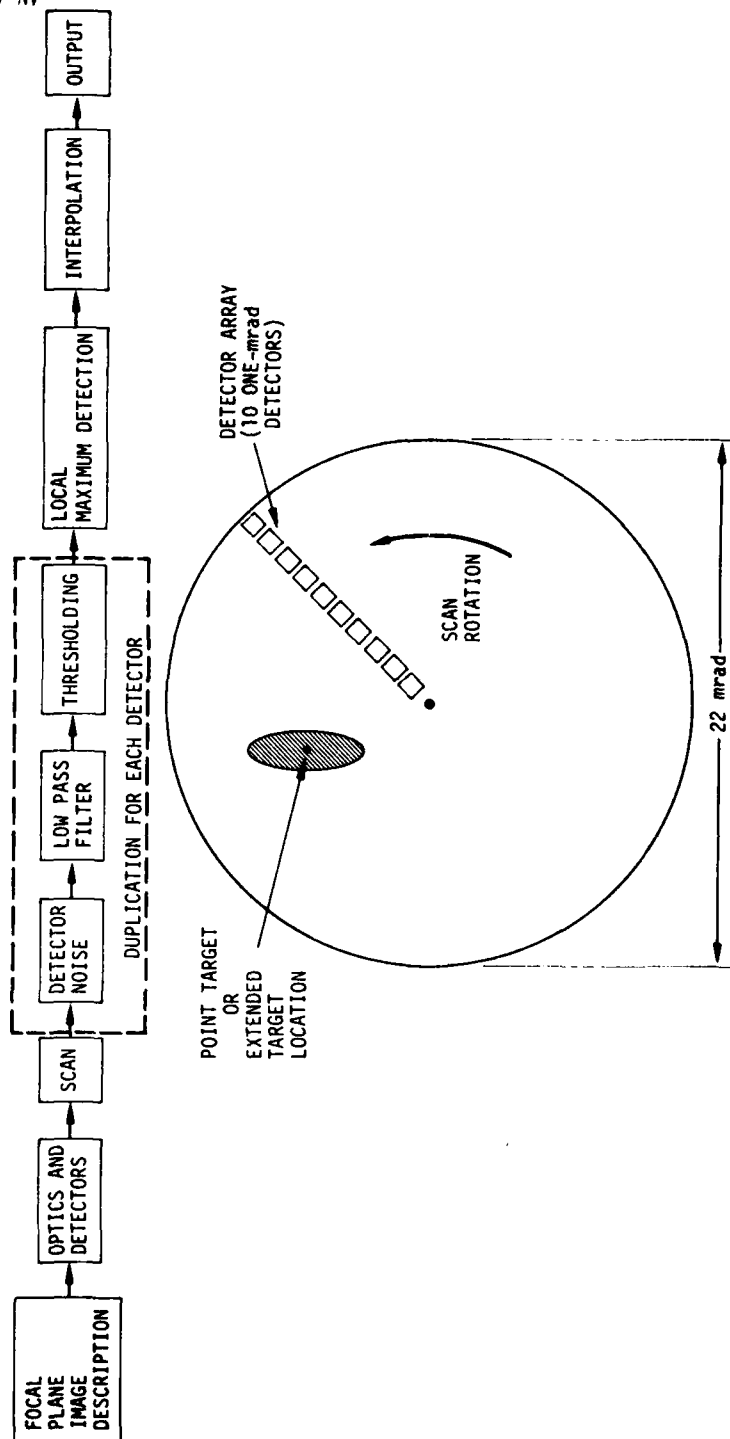


Figure 4.3. Sensor Selected for Modeling

Results of running 10 frames under different conditions are shown in Figs. 4.4 and 4.5. Figure 4.4 shows the effect of detector noise in 10 scans. In this figure a coordinate system with the target at the origin is shown. During the ten scans the target was calculated to be at the ten plotted points. Note that the interpolation procedure apparently results in a biased estimate of the target position.

Figure 4.5 shows similar results when vibration was added and when an elliptical extended target was in the field of view. For the extended target the algorithm used resulted in two targets being detected on most frames. This suggests the necessity of algorithm modification if the sensor of Fig. 4.3 must deal with extended targets.

4.4 MODELING DETECTOR NONUNIFORMITIES

As another example, Figs. 4.6 and 4.7 show some of the aspects of modeling a specific chevron tracker sensor. In a chevron tracker there are two rectangular detectors set at an angle to each other, as shown in Fig. 4.6. The scan is back and forth in the direction of the line joining the centers of the detectors. Target locations are determined along the direction of scan by the time of crossing one of the detectors (usually the one perpendicular to the scan direction) and in the cross direction by the time difference between the crossing of the two detectors.

Among the factors which contribute to the measurement errors of such a system are the optical blur and detector nonuniformities, nonlinearities in the scan, detector noise, and the effects of bandpass amplifiers on the pulse shapes. Detector nonuniformities are not treated explicitly in SPIRE, but they can be modeled, as suggested in Fig. 4.6, through the use of multiple detectors. In this simulation each of the two detectors of the actual system was represented by three superimposed detectors having different responsivities; the signals from the three were added to give the model detector output.

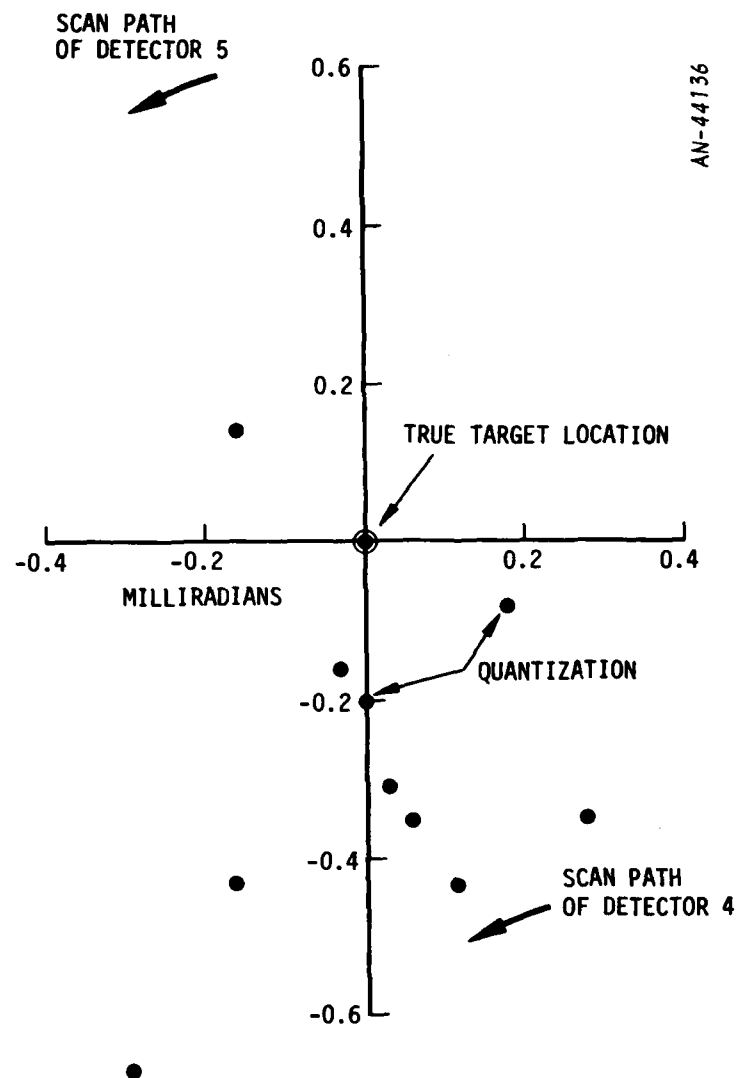


Figure 4.4. Measured Point Target Positions
For Sensor of Fig. 4.3.

VIBRATION

- ANGULAR VIBRATION OF OPTICAL AXIS
WIDE BAND - CORRELATION TIME < 0.5 ms
RMS VALUE 0.1 mrad
- OTHER CONDITIONS AS BEFORE

EXTENDED TARGET

- PEAK SIGNAL-TO-NOISE RATIO AFTER SMOOTHING ~ 5
- ON MOST FRAMES TWO MAXIMA WERE FOUND

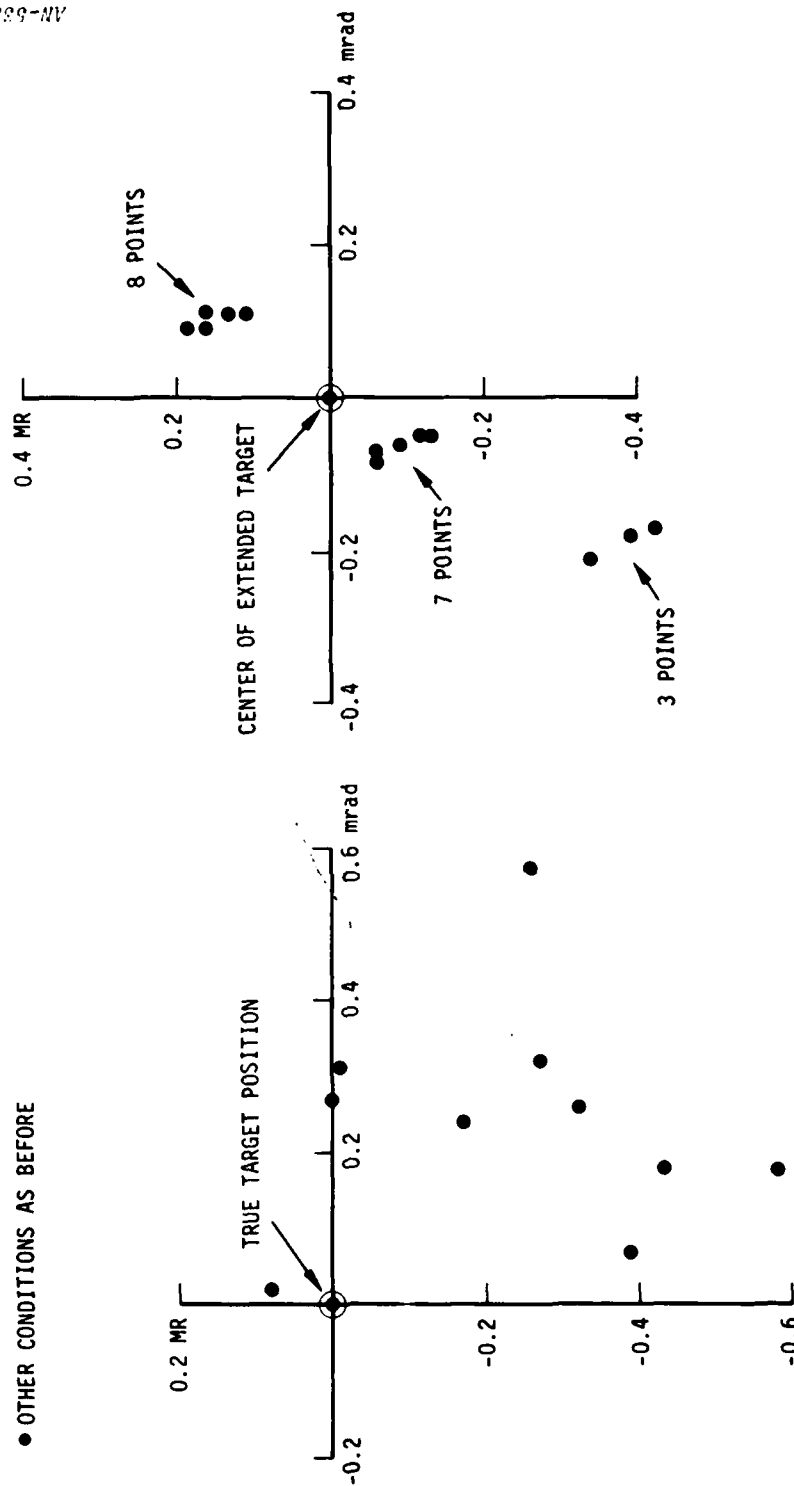


Figure 4.5. The Effects of Vibration and Extended Targets

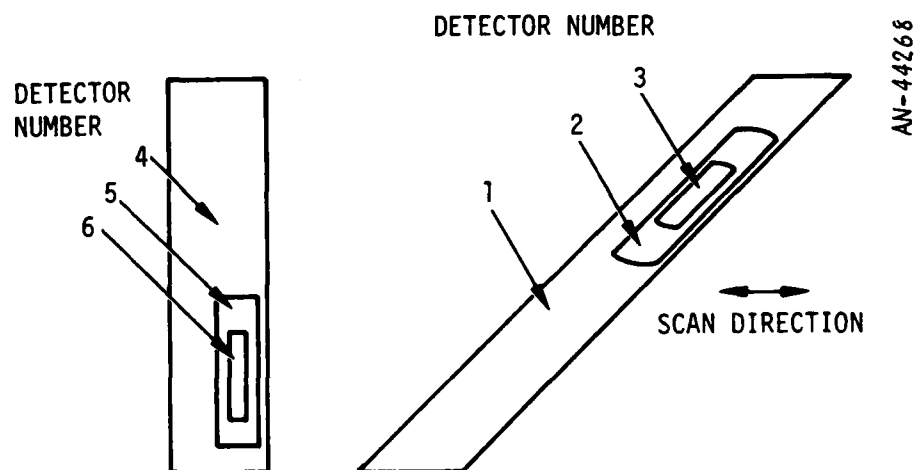


Figure 4.6. Model Detector Array

This process is shown in the block diagram of Fig. 4.7. The GAIN blocks (numbers 4, 5, 11, and 12) account for the responsivities and are adjusted so that the detector responsivity varies by the desired percentage over its surface.

As far as other sources of error are concerned, scan nonlinearities were introduced by added terms in the scan equations¹; detector noise was added in the usual way, and the time constants of the bandpass amplifiers were made slightly different to investigate tolerance to such errors.

The signal processing involves the use of an ABSOLUTE MAXIMUM module (Type 29), a GATE module (Type 18) and a TIME AND DELAY module (Type 12). In modeling this sensor, one frame consisted of a scan up

¹

The scan equations without these added terms were used to determine target locations.

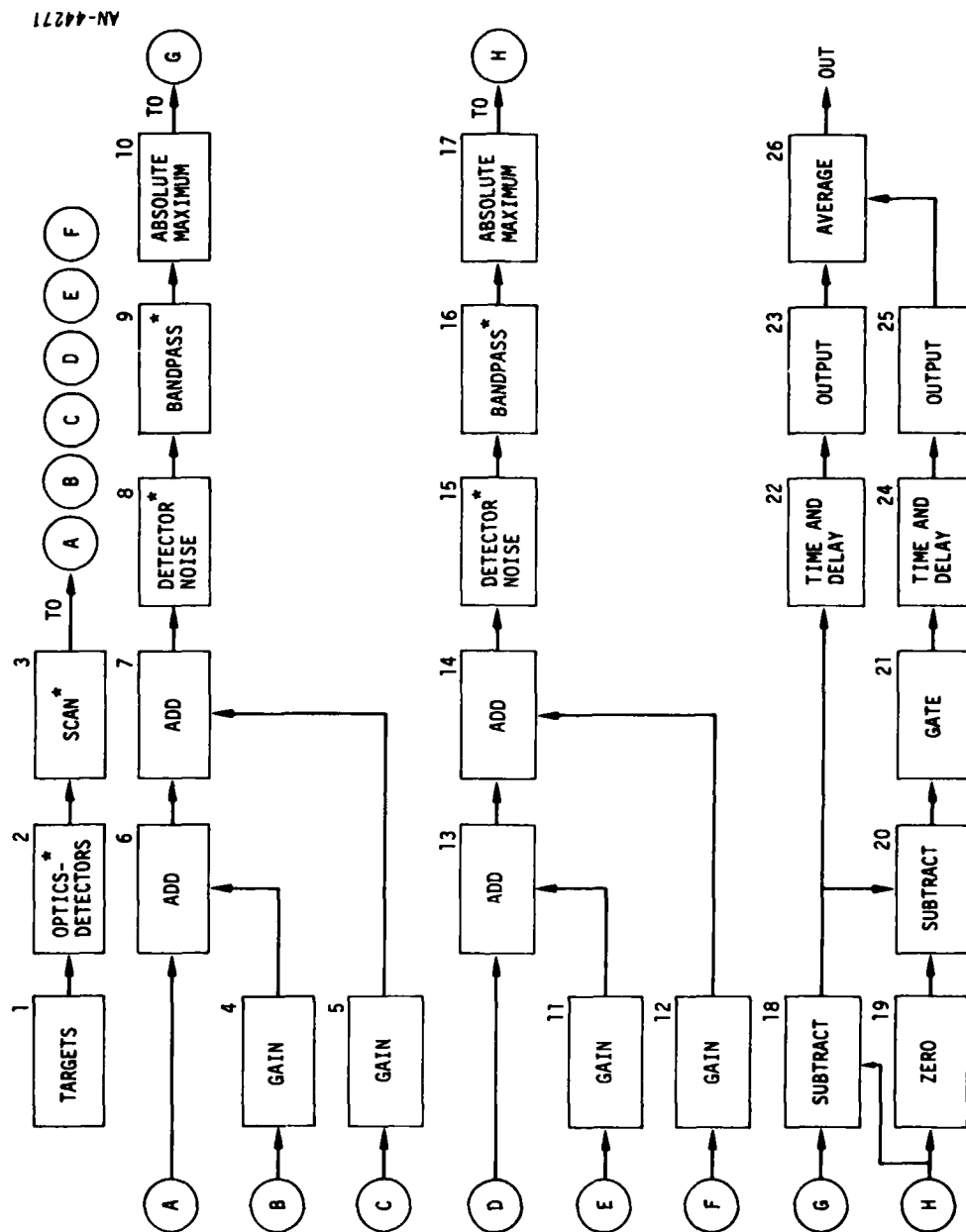


Figure 4.7. Block Diagram of Sensor Model

and back, and thus the target produced two signals out of each detector each frame. The peak signals in each half frame were detected in the ABSOLUTE MAXIMUM block. Subtraction of the signals in blocks 18 and 20 of Fig. 4.7 gave a positive spike followed by a negative spike for each half frame, and the time of the positive spike and the delay between the positive and negative spike were determined in the TIME AND DELAY block, and used in the OUTPUT blocks (numbers 23 and 25) to determine target coordinates, which were averaged in the AVERAGE block (number 26). The GATE block (Number 21) prevented the lower channel (in Figure 4.7) from responding to the upper channel signals.

5 MODULES AND VARIABLES

5.1 SPIRE PROCESSING MODULES

In this section the individual modules available in the program are described. The individual signal channels (usually one corresponding to each detector in the array) are referred to as branches in these descriptions, and the samples which represent the waveform on a particular branch are called a sample train. The inputs are the descriptors which appear in the BLK array; an abbreviated list appears at the beginning of the listing of the SPIRE subroutine. The short code-words following the type name are used in the dictionary of block types which appears at the beginning of the SPIRE subroutine in the listing. In the processing descriptions, previous values of the signal pulse train on the branch being processed are denoted by A_j ; new values generated in the module are denoted by A'_j .

Type 1: Optical System (OPTSYS)

Function

This block supplies values for the collector area and the detector quantum efficiency which are used when computing photon noise.

Inputs

Type number (1.)
Optical system collector area, cm^2
Detector quantum efficiency

Output

Values of collector area and quantum efficiency for photon noise modules.

Processing

None.

Type 2:

Not used

Type 3: Optics and Detector (OPTDET)

Function

This block supplies values for the detector and optical characteristics and computes an overall blur diameter; these values are used in other modules.

Inputs

Type number (3.)

Diameter of basic detector element (DETDIAM), radians¹

Optical blur diameter (BLRDIAM), radians

Number of detectors

Noise equivalent flux density (NEFD) at the optical aperture, W/cm²

Corresponding electrical bandwidth (NEFD Bandwidth), Hz

Output

Blur standard deviation; the number of detectors, the NEFD, and the NEFD bandwidth are stored.

Processing

Optics and detector diameters are divided by 4 to approximate their blur standard deviations, and these are RMS-added to obtain the blur standard deviation:

1. Calculate total blur:

$$BLURO = \frac{1}{4} (DETDIAM^2 + BLRDIAM^2)^{1/2}$$

2. SAMP = BLURO sample spacing, radians

¹Detectors of other sizes can be defined in DATA1, DATA2, ... Their performance will be scaled from that of the basic detector defined here.

- | | |
|----------------|--|
| 3. SCS = BLURO | if not previously assigned;
array spacing |
| 4. Call ABLUR | if not previously done
(BLRFLG = 1) |

Type 4: Scan (SCAN)¹

Function

This block generates a sampled waveform for each detector which represents the output of the detector during the specified scan period (which may be a part of one frame or many frames). Sample rates are determined from the scan rate and optics and detector blur sizes. The position (and orientation) of each detector in the array is calculated at corresponding sample times, and the energy received by the detector is determined relative to the sensor NEFD. Note that the scan block generates wave forms for all the detectors; it is not invoked separately for each detector.

Inputs

Type number (4.)
First branch number
Branch interval
Scan period, seconds
Scan length, radians

NOTE: Usually one branch corresponds to each detector output. Branch interval is specified so that the output of each detector can later be split into more than one branch for signal processing.

Output

A train of samples for each detector representing the waveform

¹Do not confuse with subroutines SCAN1, SCAN2, ...

resulting from target and background energy striking it as it scans during the scan period.

Processing

Scan rate and sample times are calculated. At each sample time the coordinates of the detector and its angle (if not square) with respect to a horizontal axis in the field-of-view are determined from equations which are input in SCAN1, SCAN2...subroutines. The sensor characteristics (NEFD, NEFD bandwidth) are used to convert target and background amplitudes to waveform amplitudes. (Detector noise is added as a random variable having a RMS value of unity in the detector noise module, Type 5.) The convolution of the detector and each background shape and target is computed at each sample time using a set of coordinate conversions, and the background and target contributions to that detector output are summed to yield the sample waveform amplitude at that time.

Type 5: Detector Noise (DETNOISE)

Function

Adds detector noise to sampled waveforms generated by the scan module. The detector outputs have been adjusted so that proper signal-to-noise ratios result when random noise having a unity RMS value is added to the waveforms.

Inputs

Type number (5.)
Branch number

Output

A train for each detector with representation of random detector noise added.

Processing

For each detector a quasi-Gaussian unity-RMS random variable is generated and added to the waveform.

1. Generate $x = \left(\sum_{j=1}^{12} y_j \right) - 6$ where the y_j are randomly chosen in the interval 0 - 1.
2. Add this number to the samples.

Type 6: Gaussian Smoothing (GAUSSMO)

Function

Represents a low-pass filter having Gaussian impulse response. (This approximates a filter matched to the detector output pulse shape.) The convolution of a truncated Gaussian and the sampled wave train yields a smoothed output wave train.

Inputs

Type number (6.)
Branch number
Pass band, Hz¹
(or) Impulse response 4-σ width, seconds

Output

A new smoothed sampled wave train for this branch number.

Processing

The convolution of the smoothing impulse response and the wave train is calculated directly.

1. Read the Pass-band or Impulse response width.

¹Input either pass band or impulse response width, set the other to zero

2. Compute Sigma:

$$\text{Sigma} = \frac{1}{4 \times \text{Pass-band}} \times \frac{\text{Scan Rate}}{\text{Samp}}$$

$$\text{or Sigma} = \frac{\text{Impulse response width}}{4 \times \text{Samp}}$$

Samp = Sample spacing, radians

3. Compute new A'_j values

$$A'_j = \sum_k \frac{A_k}{\sqrt{2\pi} \text{Sigma}} \exp \left[-\frac{(j - k)^2}{2(\text{Sigma})^2} \right]$$

where the summation extends $\pm 3 \text{Sigma}$
around j .

Type 7: Bandpass Smoothing (ACSMO)

Function

Represents an AC-coupled bandpass amplifier: convolution with the sum of a truncated Gaussian and a wider inverted truncated Gaussian of the same area suppresses DC signal components

Inputs

Type number (7.)

Pass band, Hz¹

(or) Impulse response 4- σ width, seconds

Ratio between lengths of direct and inverted Gaussians

Output

A new smoothed sampled wave train for this branch number.

¹Input either pass band or impulse response width; set the other to zero.

Processing

The required convolution is calculated directly.

1. Read time constant or space constant
2. Compute Sigma (high-frequency cutoff)
3. Loop on samples
4. Compute new A'_i values

$$A'_j = \sum_k \left\{ \frac{1}{\sqrt{2\pi} \text{Sigma}} \exp \left[\frac{(j-k)^2}{2(\text{Sigma})^2} \right] - \frac{\text{MULT}}{2\pi \text{Sigma}} \exp \left[- \frac{(j-k)^2 \text{MULT}^2}{2(\text{Sigma})^2} \right] \right\}$$

MULT = Ratio of low-frequency cutoff to high-frequency cutoff

The summation extends from -3Sigma to $+3 (\text{Sigma}) \times (\text{MULT})$

Type 8: Clipping (CLIP)

Function

Represents clipping or limiting of an electrical waveform. All sample values above (or below) a given clipping level are set equal to that level.

Inputs

Type number (8.)

Branch number

Clipping level

Sign (+1. if samples having values above the clipping level are set equal to the level, -1. if those below are to be set equal to the clipping level)

Output

New sampled wave train representing clipped waveform.

Processing

Direct test of each sample in the wave train.

1. Read clipping level
2. Loop on samples
3. If $A_j \geq CL$, $A'_j = CL$
CL = clipping level

Type 9: Threshold (THRESH)

Function

Finds the sample points at which threshold crossing occurs.
All sample points above (or, if desired, below) a fixed threshold value are set equal to 1, if the previous value was below (or above) threshold. All other values are set equal to zero.

Inputs

Type number (9.)
Branch number
Threshold value
Sign (if +1., positive threshold crossings
are identified, if -1., negative threshold crossings
are identified)

Output

New sampled wave train consisting of zeros and ones.

Processing

Direct test of each sample in wave train:

1. Read threshold
2. Loop on samples

3. If $A_j > \text{threshold}$ and

$A_{j-1} \leq \text{threshold}$ then $A'_j = 1.$;

otherwise $A'_j = 0.$

Type 10: Time Delay (DELAY)

Function

Delay trains so that outputs of detectors can be compared. This consists of renumbering the train pulses.

Inputs

Type number (10.)

Branch number

Spatial delay, radians

(or) Time delay, seconds

Outputs

Identical train with samples reindexed

Processing

The proper change of index is determined from delay input data:

1. Read delay time or delay space
2. Compute delay sample interval del
3. Loop on samples
4. $A'_{j+\text{del}} = A_j$

Type 11: Level Adjust (LEVEL)

Function

An input quantity is added to or subtracted from each pulse in a train.

Inputs

Type number (11.)

Branch number

Level to be added

Output

New train with changed DC level.

Processing

Input level is added to each sample of designated train.

1. Read level to be added
2. Loop on samples
3. $A'_j = A_j + \text{Level}$

Type 12: Threshold Crossing Time Differences (TIMAD)

Function

Determines the time between each positive and the succeeding negative crossing of a specified threshold. Each positive and subsequent negative threshold crossing is detected; the time in seconds between them is determined from the number of samples between them, and the time difference is assigned as a new value at the negative crossing sample. All other train pulses are set equal to zero.

Inputs

Type number (12.)
Branch number
Threshold (relative amplitude)

Output

Train having non-zero values only at negative threshold crossings; at these points value is time in seconds since last positive crossing.

Processing

Positive followed by negative crossings are detected, differences are measured and assigned as new train values:

1. Read threshold
2. Loop on samples (positive-going)
3. Obtain time of first (positive-going) threshold crossing
4. At time of next negative-going threshold crossing $A'_j = \text{time difference}$
5. $A'_j = 0$ at other sample times
6. Locate next (positive-going) threshold crossing and repeat

Type 13: Local Maxima (LOCMAX)

Function

Locates train local maxima; this corresponds exactly to differentiation followed by detection of zero crossing. Train samples are compared to determine either local maxima or locations of two equal samples preceded by one of lower amplitude.

Inputs

Type number (13.)

Branch number

Output

Train with samples equal to original train at local maxima;
otherwise equal to zero.

Processing

Desired samples are determined directly.

1. Loop on samples
2. If $A_{j-1} < A_j > A_{j+1}$, $A'_j = A_j$
3. Otherwise, $A'_j = 0$

Type 14: Branch Point (BRANCH)

Function:

Develops a new train equal to a previous one for subsequent parallel processing of waveforms. The previous train is duplicated with a new branch number.

Inputs

Type number (14.)

Old branch number

New branch number 1

New branch number 2

Output

Two additional trains.

Processing

Two trains are created with values equal to the old branch number train, assigned to new branch number 1 and new branch number 2.

1. Read two new branch numbers (Sample values B_j , C_j)
2. Loop on samples
3. $B_j = C_j = A_j$

Type 15: Comparison (COMP)

Function

To perform arithmetic operations on corresponding samples of two trains, corresponding (simultaneous) samples of specified trains are added, subtracted, multiplied, or divided.

Inputs

Type number (15.)
Branch number 1
Branch number 2
Type of operation (1. = addition, 2. = subtraction,
3. = multiplication, 4. = division)

Output

New train on branch 1.

Processing

Trains on branches 1 and 2 are combined according to the type of operation specified, and assigned as new values on branch 1.

1. Read branches to be compared (sample values A_j , B_j)
2. Read operation (sum, subtract, multiply, divide - symbol $\emptyset$)

3. Loop on samples
4. $A_j = A_j \oslash B_j$

Type 16: Average and RMS Calculation (RMS-AV)

Function

The mean and RMS (standard deviation from mean) values of a train over the entire period are calculated and stored.

Inputs

Type number (16.)
Branch number

Output

Stored values of pulse mean and standard deviation.

Processing

Direct computation of mean and standard deviation of pulses on the specified branch.

1. $\text{Mean} = \frac{1}{N} \sum A_j$ (N=number of samples)
2. $\text{RMS} = \left(\frac{1}{N} \sum A_j^2 - \text{Mean}^2 \right)^{1/2}$

Type 17: Gain (GAIN)

Function

Increase amplitude of a train by multiplying each pulse by a constant.

Inputs

Type number (17.)
Branch number
Gain (or amplification factor)

Output

New amplified train on that branch.

Processing

Each pulse amplitude on that branch is multiplied by the gain:

1. Read gain
2. Loop on samples
3. $A'_j = (\text{GAIN}) \cdot A_j$

Type 18: Gate (GATE)

Function

To pass only signals whose occurrence corresponds to specific spatial locations. The given train is set equal to zero except within a spatial region where original values are retained.

Inputs

Type number (18.)
Branch
Gate leading edge, radians
Gate trailing edge, radians

Processing

Gate locations are converted to pulse indices and all pulses outside this range are set equal to zero:

1. Read gate location limits
2. $N_1 = \text{INT} (\text{initial location}/\text{SAMP})$
 $N_2 = \text{INT} (\text{final location}/\text{SAMP})$
3. Loop on samples
4. If $N_1 < j < N_2$, $A'_j = A_j$
otherwise $A'_j = 0$

Type 19: Differentiator (DERIV)

Function

To simulate the performance of a differentiating circuit. Differences between successive pulses are obtained and multiplied by a specified constant.

Inputs

Type number (19.)
Branch number
Gain

Output

Differentiated train on same branch.

Processing

Direct differencing of successive pulses and multiplication by gain:

1. Read gain
2. Loop on samples
3. $A'_j = (\text{GAIN}) \cdot (A_{j+1} - A_j)$

Type 20: Duplicator (DUPLIC)

Function

To expand block diagrams by reproducing sections of an input diagram, in order to reduce the labor of generating inputs. A new set of blocks and branches are generated having properties identical to a specified set of already entered blocks and branches.

Inputs

Type number (20.)
Number of branches to be duplicated
Initial block to be duplicated
Final block to be duplicated

Output

Expanded set of blocks and branches

Processing

The set of blocks to be duplicated must be numbered consecutively. A set of new branches is generated, numbered starting with one greater than the branch number of the blocks to be duplicated; to each branch is assigned a set of blocks having characteristics identical to those of the blocks to be duplicated. (The train for these branches must be generated elsewhere as in a scan block.)

1. Read number of branches (Symbol N), first block to be duplicated (Symbol B1), last block to be duplicated (Symbol B2)
2. Create new set of blocks on branches
 $N1 + 1, N1 + 2, \dots, N1 + N$
identical to blocks B1 . . . B2, except for branch numbers. N1 is the branch number of blocks B1...B2.

Type 21: General Nonlinear Processing (NONLIN)

Function

To simplify insertion of any nonlinear amplitude-proportional signal processing. User enters a statement function representing output pulse amplitude as a function of input pulse amplitude.

Inputs

Type number (21.)

Branch number

Output

A new train on the given branch with adjusted amplitudes.

Processing

Amplitude of each pulse of specified train is converted using the amplitude function:

1. Loop on samples
2. $A'_j = \text{AMPFN}(A_j)$ (AMPFN is any function inserted by the user).

Type 23: Lateral Weighting (LATWT)

Function

To represent the combination of signals from nearby detectors in an array as required for signal-to-noise ratio improvement, correct amplitude measurement, and spatial filtering. A lateral weighting function is introduced by the user; this is convolved with the sets of simultaneous pulses occurring on a set of branches.

Inputs

Type number (23.)

First branch number

Last branch number

Space between branches (radians)

Output

A new set of trains on the specified branches representing the effects of lateral smoothing.

Processing

For each branch and at each pulse interval a new pulse is generated by adding the simultaneous pulses which occur on the other branches, each weighted by the lateral weighting function:

1. Read branches across which weighting is to be done (symbols B1...BN), spacing.
2. Loop on N samples
3. $A'_j = \sum_{k=1}^N B_j \cdot \text{WEIGHT}[(k-j) \cdot \text{Spacing}]$

(WEIGHT is a weighting function inserted by the user.
Spacing is the spacing between branches as required by the function WEIGHT.)

Type 24: Two Dimensional Maximum Detection (TWODMAX)

Function

To simulate the detection of targets by locating local maxima in the output of an array of detectors. Each pulse on a set of branches is examined to see if it is greater than the pulses before and after it in its own train and the simultaneous pulses on the two adjacent trains.

Inputs

Type number (24.)
First branch to be compared
Last branch to be compared

Output

New set of trains having original values at location of local maxima and approximately zero values at all other pulse locations.

Processing

Direct comparison of each pulse with four pulses (two from same train, two on adjacent branches):

1. Read range of branches
2. Loop on samples and branches
3. If $A'_j >$ adjacent samples on same branch and on adjacent branches, $A'_j = A_j$
4. Otherwise $A'_j = A_j * 10^{-200}$ (This has the effect of reducing the amplitude of all samples which are not local maxima, but preserving their relative values.)

Type 25: Photon Noise Generation (PHOTN)

Function

To introduce detector output variations due to photon statistics. The energy represented by each detector output pulse is determined from the pulse amplitude and the sample spacing in time. The corresponding mean number of photons is determined, and to the original pulse a random variable is added chosen from a quasi-Poisson distribution to represent the fluctuations expected in specific photon samples.

Inputs

Type number (25.)
Branch number
Detector number
Wavelength, micrometers

Output

A pulse train representing the detector output with photon fluctuations.

Processing

The pulse amplitudes in the detector output train are determined relative to the sensor NEFD prior to bandpass filtering. Thus, these amplitudes allow the number of photons to be determined from the collector area, the average wavelength of radiation, and the sample time, taking into account the fraction of energy from a point target which is actually collected by the individual detector. A random number representing fluctuations in the photon sample is added to the detector output pulse:

1. Find optical system block (OPTSYS, NTYPE = 1.)
2. Read quantum efficiency, collector area
3. If quantum efficiency = 0., set equal to 0.5
4. Find optics and detector block (OPTDET, NTYPE = 3)
5. Read optical blur, NEFD, NEFD bandwidth
6. Compute normalization factor
7. Loop on samples
8. Compute random number GAUS with approximate gaussian amplitude distribution

$$9. A'_j = A_j + \text{GAUS} \cdot [A_j \cdot \text{FACT}]^{1/2}$$

$$A_j = \text{sample}$$

$$\text{GAUS} = \text{random number}$$

$$\text{FACT} = \frac{2 \times 10^{-19}}{\Delta t \cdot \lambda \cdot A \cdot \text{NEFD} \cdot Q \cdot \text{FA} \cdot \text{FB} \cdot \text{FC}}$$

$$\Delta t = \text{SAMPT} = \text{sample interval, s}$$

$$\lambda = \text{wavelength, } \mu\text{m}$$

$$A = \text{ARCOL} = \text{collector area, cm}^2$$

$$\text{NEFD} = \text{noise equivalent flux density, watt/cm}^2$$

This must be provided as an input; it is

computed for a detector of area DSIZE x DSIZE)

$$Q = \text{Quantum efficiency}$$

$$FA = 1 - \exp \left\{ - \left(\frac{DSIZE^2}{\pi} \right) / \left[2 \left(\frac{OPB^2}{4} \right) \right] \right\}$$

DSIZE = detector element diameter, radians

OPB = optical blur diameter, radians

$$FB = (A_D)^{1/2} / DSIZE$$

A_D = actual detector area

$$FC = \left\{ 1 / (2 \cdot \Delta t \cdot DNEB) \right\}^{1/2}$$

DNEB = Electrical bandwidth in which NEFD is measured
or calculated (an input)

Type 26: Output (OUTPUT)

Function

To call the proper OUTPUT subroutine corresponding
to the output flag.

Inputs

Type number (26.)

Branch number

Output

Calls proper OUTPUT subroutine

Processing

Whenever block type 26 is encountered the output subroutine
is called. Any output differences due to branch number must
be taken account of in the output subroutine:

1. Call output subroutine (OUT1, OUT2,...OUT10)
corresponding to the flag OUTFLG.

Type 27: General Smoothing (GENSMO)

Function

To provide for linear processing of trains. The impulse response of an electrical filter is given by the user. The convolution of this with the train is determined.

Inputs

Type number (27.)

Branch number

First truncation limit

Second truncation limit

Sign (+1 indicates that impulse response is a function of time; -1 indicates a function of space, radians)

Output

A new train representing linear processing of the original.

Processing

The impulse response is converted into a space-dependent form if given as a time-dependent function, and truncated according to the number of pulses specified by the truncation limits. This modified response is convolved with the train:

1. Read limits (L1, L2) on convolution range, flag indicating whether smoothing function is space or time function.
2. Compute space.
3. Loop on samples.

$$4. A_j = \sum_{k=L1}^{j+L2} A_k \cdot \text{SMOOTH} [(j - k) \cdot \text{SPACE}]$$

(SPACE = SAMP or SAMPT, depending on the function SMOOTH.

SMOOTH is an arbitrary smoothing function provided by the user.)

Type 28: Exponential Smoothing (TIMCON)

Function

Accomplishes exponential smoothing (single time constant low-pass filtering) of pulse trains.

Inputs

Type number (28.)
Branch number
Time constant, seconds
(or) space constant, radians

Output

A new train representing exponential smoothing of the original.

Processing

Each sample of the new train is produced from the sum of that sample of the original train and the appropriate fraction of the previous sample of the new train:

1. Read time constant or space constant
2. Compute QK: $QK = \exp(-\text{SAMPT}/\text{time constant})$
or $= \exp(-\text{SAMP}/\text{space constant})$
3. Loop on samples
4. $A'_j = A_j \cdot (1 - QK) + A_{j-1} \cdot QK$

Type 29: Absolute Maximum Computation (ABSMAX)

Function

To obtain the largest signal sample in specified fractions of the train corresponding to one frame.

Inputs

Type number (29.)
Branch number
Number of subperiods

Output

Pulse train with all but the largest sample in each subperiod set equal to zero.

Processing

The train is divided into the specified number of subperiods, and the largest sample in each is found by sequential comparison:

1. Read number of subintervals
2. Loop on subintervals
3. Loop on samples
4. If A_j is the largest positive sample in the subinterval,
 $A'_j = A_j$. Otherwise $A'_j = 0$

Type 30: Binary Quantization (QUANT)

Function

To quantize the samples in a particular train in a binary format.

Inputs

Type number (30.)

Branch number

Number of bits

Output

A new train of samples with the amplitudes rounded to an accuracy given by the input number of bits.

Processing

Each sample amplitude is rounded off to the proper number of bits:

1. Read number of bits (Symbol NU)
2. Loop on samples
3. Compute $KK = \text{INT} \left(\frac{\text{Ln } (A_j)}{\text{Ln } (2)} \right) - \text{NU}$
4. $A'_j = \text{INT} [A_j / 2^{KK} + .5] \cdot 2^{KK}$

Type 31: Display Train (DISPA)

Function

To provide for printing of all samples in the train corresponding to a particular branch, which are above a given threshold.

Inputs

Type number (31.)

Branch number

Threshold

Output

Printout of the indices and values of all samples greater than the threshold.

Processing

Each sample is compared with the threshold and printed if it is above threshold:

1. Loop on samples
2. If sample is greater than input threshold value, print sample value

Type 32: High-pass Filtering (HIPASS)

Function

To provide for single time constant high-pass filtering of the pulse train on one branch

Inputs

Type number (32.)
Branch number
Time constant, seconds
(or) space constant, radians

Output

A new train representing single time constant high-pass filtering of the original train.

Processing

Each sample of the new train is produced from the proper combination of the corresponding original sample and the preceding sample:

1. Compute multiplier QK from time constant:

$$QK = \exp (-\text{SAMPT}/(\text{Time constant}))$$

$$\text{or } QK = \exp (-\text{SAMP}/(\text{Space constant}))$$

SAMPT = Sample period

SAMP = Sample spacing

Time constant or space constant is an input

2. New second sample value A'_2 :

$$A'_2 = A_1 \cdot (1. - QK)$$

3. Loop on samples

4. New succeeding sample values:

$$A'_j = A_j \cdot QK - (A'_j - A_{j-1}) \cdot QK$$

Type 33: Quantization (QUANT)

Function

To assign the amplitudes of the pulses on a particular train to various quantization levels on either a linear or logarithmic basis.

Inputs

Type number (33.)

Branch number

Number of levels

Peak signal

Minimum signal

Sign (1. implies linear quantization,

2. implies logarithmic quantization.)

Output

A new train of samples with the amplitudes expressed as one of the input number of quantization levels.

Processing

Each sample amplitude is assigned to one of the specified number of levels, on a linear or logarithmic basis.

1. For linear quantization read Number of levels and Peak

2. Quantization interval $DEL = \text{Peak} / (\text{Number of Levels})$
 2. Loop on samples
 3. $A'_j = DEL \cdot \text{INT} (A_j / DEL)$
 4. For logarithmic quantization read Peak, Valley, and Number of levels
 5. Quantization interval SZ:

$$SZ = \frac{\text{Ln} (\text{PEAK}/\text{VALLEY})}{(\text{number of levels})}$$
 6. Loop on samples
 7. $A'_j = \text{Valley} \cdot \exp (SZ \cdot \text{INT} (\text{Ln} (\frac{A_j}{\text{Valley}}) / SZ))$
- NOTE: If A_j is less or greater than the minimum or maximum values (zero and peak for linear; valley and peak for logarithmic) it is set equal to the minimum or maximum.

5.2 VARIABLES IN COMMON STORAGE

The principal variables in SPIRE are stored in labeled common storage and used in various subroutines. These variables are defined in the following list.

COMMON/A/

- | | |
|-----------|---|
| A(200,50) | Array A contains the sample values on each branch; the first index is the sample number, the second is the branch number. |
| B(300) | Array B is used in various subroutines for storage of samples on one branch. |
| LA(50) | LA is the number of the first sample of interest in A(200,50) for each branch. |
| LB(50) | LB is the number of the last sample of interest in A(200,50) for each branch. LA and LB are used to |

avoid processing values of A outside the range of interest.

COMMON/B/

BLK (10,990) Contains the descriptors for the blocks in the system. First index is the descriptor number, second is the block number.

SM(50) Provides storage for mean value of samples in a branch, computed by block type 16.

SS(50) Provides storage for standard deviation of samples in branch, computed by block type 16.

COMMON/C/

JBR(50) Provides storage for any integer index connected with various branches (not used in this version of program).

XOBR(50) Provides storage for any coordinate connected with various branches (not used in this version of program).

YOBR(50) Provides storage for any coordinate connected with various branches (not used in this version of program).

COMMON/D/

ARRX(50) The initial X-coordinates of the detector array

ARRY(50) The initial Y-coordinates of the detector array

ARRL(50)	The lengths of the detectors in the array
ARRA(50)	The initial angles of the detectors in the array. ¹
ARRSR(50)	The spectral regions of the detectors in the array (not used in this version of the program).
ARRW(50)	The widths of the detectors in the array.
 <u>COMMON/H/</u>	
BR	The number of the branch being processed.
DEL	The size of the quantization step when the samples are quantized linearly.
BR1	The number of the branch being processed (in blocks of type 14 or 15)
BR2	The number of the branch being output (in blocks of type 14) or combined (in blocks of type 15).
BR3	The number of the branch being output (in blocks of type 14).
BLKS	The number of blocks to be processed.
PI	3.14159

¹This is the angle from the scan coordinate system X-axis to the detector coordinate system X-axis. ARRW(J) is measured along the detector coordinate system X-axis, ARRL(J) is measured along the detector coordinate system Y-axis. See Figure 5.1.

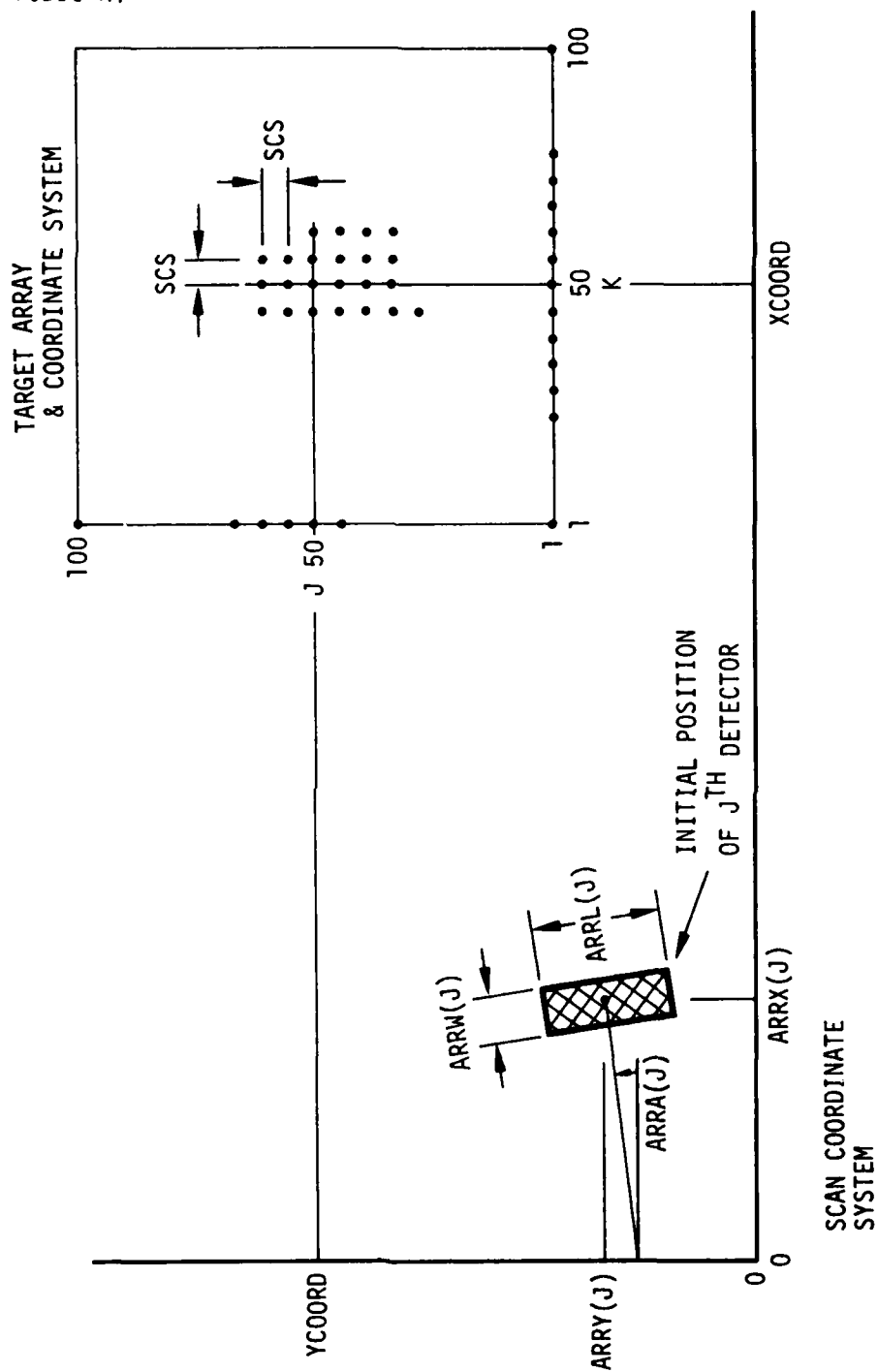


Figure 5.1. Coordinate Systems & Notation

RANDM	A random number generated in the program and used as a seed; this allows photon noise and detector noise to be different from run to run if desired.
NTYPE	The type of block being processed.
<u>COMMON/I/</u>	
SCRA	The scan rate in radian/second
SAMP	The sample distance in radians, equal to $SCRA * SAMPT$
SAMPT	The sample time in seconds. This is the interval at which the samples representing the waveforms on each branch are generated.
NO	The (sequence) number of the block being processed.
BLUR	The image spread due to the optics and the detector width
BLURO	The image spread due to the optics and the detector elemental area.
TSTART	The time the scan starts.
DSIZE	The linear dimensions of the elemental detector area in radians. For square detectors about the size of the optical blur or smaller, this is the size of the detector in radians.

COMMON/M/

DATFLG	The flag indicating which data subroutine should be used.
SCNFLG	The flag indicating which scan function should be used.
OUTFLG	The flag indicating which output subroutine should be used.
BLRFLG	The flag indicating whether blurring of the image is done in the executive (SPIDER, for example) or in subroutine SPSUB1.

COMMON/S/

SC(100,000)	The array containing the image plane description
SCS	The spacing (in radians) between points in array SC
XCOORD	The X-coordinate of the center of array SC
YCOORD	The Y-coordinate of the center of array SC

DISTRIBUTION LIST

DEPARTMENT OF DEFENSE

Assistant to the Secretary of Defense
Atomic Energy
ATTN: Executive Assistant

Defense Advanced Rsch. Proj. Agency
ATTN: STO, S. Zakanycz
ATTN: TIO

Defense Communications Engineer Center
ATTN: Code R410, J. McLean

Defense Nuclear Agency
ATTN: RAAE
ATTN: DDST
4 cy ATTN: TITL

Defense Technical Information Center
12 cy ATTN: DD

Field Command
Defense Nuclear Agency
ATTN: FCPR

Field Command
Defense Nuclear Agency
Livermore Division
ATTN: FCPRL

Interservice Nuclear Weapons School
ATTN: TTV

National Security Agency
ATTN: R52, J. Skillman

Undersecretary of Defense for Rsch. & Engrg.
ATTN: Strategic and Space Systems (OS)

WWMCCS System Engineering Org.
ATTN: R. Crawford

DEPARTMENT OF THE ARMY

Atmospheric Sciences Laboratory
U.S. Army Research and Development Command
ATTN: DELAS-EO, F. Niles

BMD Advanced Technology Center
ATTN: ATC-O, W. Davies
ATTN: ATC-T, M. Capps

BMD Systems Command
Department of the Army
ATTN: BMDSC-HW, R. Dekalb

Harry Diamond Laboratories
Department of the Army
ATTN: DELHD-N-P, F. Wimenitz
ATTN: DELHD-I-TL, Technical Library

U.S. Army Foreign Science & Technical Center
ATTN: DRXST-SD

U.S. Army Missile Intelligence Agency
ATTN: J. Gamble

DEPARTMENT OF THE ARMY (Continued)

U.S. Army Missile R&D Command
ATTN: DRDMI-XS
ATTN: RSIC

U.S. Army Nuclear & Chemical Agency
ATTN: Library

U.S. Army Satellite Comm. Agency
ATTN: Document Control

U.S. Army TRADOC Systems Analysis Activity
ATTN: ATAA-PL

DEPARTMENT OF THE NAVY

Naval Electronic Systems Command
ATTN: PME 117-20
ATTN: Code 501A

Naval Intelligence Support Ctr.
ATTN: Document Control

Naval Ocean Systems Center
ATTN: Code 532

Naval Postgraduate School
ATTN: Code 1424, Library

Naval Research Laboratory
ATTN: Code 6700, T. Coffey
ATTN: Code 6780, S. Ossakow
ATTN: Code 6709, W. Ali
ATTN: Code 6701, J. Brown
ATTN: Code 6780, P. Palmadesso
ATTN: Code 2627, Technical Library

Naval Surface Weapons Center
ATTN: X211, Technical Library

Naval Surface Weapons Center
ATTN: Code F14, R. Butler

Strategic Systems Project Office
Department of the Navy
ATTN: Nssp-2722, M. Meserole
ATTN: Nssp-2722, F. Wimberly

DEPARTMENT OF THE AIR FORCE

Air Force Geophysics Laboratory
ATTN: SULL
ATTN: LKB, K. Champion
ATTN: OPR, A. Stair
ATTN: OPR, H. Gardiner

Air Force Systems Command
ATTN: Technical Library

Air Force Technical Applications Center
ATTN: TFR, C. Meneely
ATTN: Technical Library

Air Force Weapons Laboratory
ATTN: SUL
ATTN: DYC

DEPARTMENT OF THE AIR FORCE (Continued)

Ballistic Missile Office
Air Force Systems Command
ATTN: MNX

Deputy Chief of Staff
Research, Development, & Acq.
Department of the Air Force
ATTN: AFRDQ

Headquarters Space Division
Air Force Systems Command
ATTN: DYS Space Defense Systems

Headquarters Space Division
Air Force Systems Command
ATTN: RSP

Headquarters Space Division
Air Force Systems Command
ATTN: SKX
ATTN: SKA, M. Clavin

Headquarters Space Division
Air Force Systems Command
ATTN: SZJ, P. Kelley

Rome Air Development Center
ATTN: OCSA, J. Simons
ATTN: TSLD
ATTN: OCS, V. Coyne

Strategic Air Command
Department of the Air Force
ATTN: NRT
ATTN: SPFS, B. Stephan

DEPARTMENT OF ENERGY

Department of Energy
ATTN: Document Control for Oma

OTHER GOVERNMENT AGENCIES

Department of Commerce
National Oceanic & Atmospheric Admin
Environmental Research Laboratories
ATTN: F. Fehsenfeld

Institute for Telecommunications Sciences
National Telecommunications & Info. Admin.
ATTN: W. Utlaut
ATTN: G. Falcon

DEPARTMENT OF DEFENSE CONTRACTORS

Aerojet Electro-Systems Co.
ATTN: J. Graham

Aerospace Corp.
ATTN: J. Reinheimer
ATTN: N. Stockwell
ATTN: I. Garfunkel
ATTN: R. Rawcliffe
ATTN: V. Josephson
ATTN: N. Cohen

Berkeley Research Associates, Inc.
ATTN: J. Workman

DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

ESL, Inc.
ATTN: J. Marshall

General Electric Co.
Space Division
ATTN: M. Bortner
ATTN: T. Baurer

General Electric Company-TEMPO
ATTN: DASIAC
ATTN: W. Knapp
ATTN: T. Stevens
ATTN: J. Jordano
ATTN: M. Stanton

General Research Corp.
ATTN: J. Ise, Jr.
ATTN: J. Garbarino
ATTN: H. Gilmore

Jamieson Science & Engineering
ATTN: J. Jamieson

Kaman Sciences Corp.
ATTN: D. Perio
ATTN: N. Beauchamp
ATTN: P. Tracy

Lockheed Missiles and Space Co., Inc.
ATTN: M. Walt

M.I.T., Lincoln Lab.
ATTN: D. Towle

McDonnell Douglas Corp.
ATTN: R. Halprin

Mission Research Corp.
ATTN: M. Scheibe
ATTN: D. Sappenfield
ATTN: D. Archer
ATTN: R. Bogusch
ATTN: F. Fajen
ATTN: R. Kilb
ATTN: R. Hendrick

Photometrics, Inc.
ATTN: I. Kofsky

University of Pittsburgh
Cathedral of Learning
ATTN: F. Kaufman

R & D Associates
ATTN: C. MacDonald
ATTN: R. Turco
ATTN: R. Lelevier
ATTN: B. Gabbard
ATTN: F. Gilmore

R & D Associates
ATTN: H. Mitchell

Rand Corp.
ATTN: C. Crain

Science Applications, Inc.
ATTN: D. Hamlin

DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

Science Applications, Inc.

ATTN: W. Mendes

ATTN: D. Divis

ATTN: R. Deliberis

SRI International

ATTN: W. Chesnut

ATTN: W. Jaye

DEPARTMENT OF DEFENSE CONTRACTORS (Continued)

Teledyne Brown Engineering

ATTN: J. Beaupre

ATTN: J. Cato

ATTN: G. Harney

Visidyne, Inc.

ATTN: J. Carpenter

ATTN: C. Humphrey