A Minicomputer Vector Generator

R. F. Emerson
Communications Systems Research Section

A vector generator was designed and built as a peripheral for a minicomputer. The vector generator is a device which accepts two endpoints and draws a straight line between them on some display device. While this could be done point-by-point by a minicomputer, it would use valuable computing power in an inefficient way. This generator is part of the Precision Signal Power Measurement (PSPM) System and is used to graph the power spectra of the signal measured to determine performance of spacecraft and ground telemetry transmitters and receivers. This hardware increases the graphing efficiency by an average of 4000 times over the point-by-point plotting method, requiring less than 15 μs of computer time per endpoint. The vectors are composed of points spaced 0.004 mm apart, providing excellent resolution and linearity.

I. Introduction

A vector generator was designed and built as a peripheral for the Lockheed Electronics Company MAC 16 minicomputer. The MAC 16 is being used as the control computer for the development of the Precision Signal Power Measurement (PSPM) System, a system which measures the downlink power of a spacecraft signal (Refs. 1 and 2). Measurement of the signal power is accomplished by computing the power spectrum of the signal and from this the signal-to-noise ratio. The signal-to-noise ratio is but one of many parameters available from the spectrum. The continuing demands on the PSPM System call for the utilization of other parameters contained within the spectrum. One of the techniques that can be used to measure other parameters is the graphing of the spectrum. In the past the graphing of the spectra has been done by hand, line printer, and slow computer-controlled mechanical plotting. These methods are no longer satisfactory because of the increasing data rates of the PSPM System. Replacing the mechanical plotter with a cathode-ray tube (CRT) display can improve the situation, but point-by-point plotting on a CRT has inherent speed limitations requiring that the line be broken into small segments and plotted incrementally. For each step in this process, the computer must calculate the increment and send it to the plotter. To maintain linearity and provide sufficient resolution, these calculations must be done hundreds of times even for lines of 1 to 2 cm. It will require, typically, 50 to 60 μs of computer processing time for each step during which time no other calculations can be done. A major portion of these calculations is now done by the hardware described in this report.

The vector generator, built as a peripheral to the MAC 16 computer, performs the computing necessary for plotting vectors and requires less than 30 μs of computer time for each vector regardless of length. This results in increasing the efficiency of plotting by an average of 4000 times with respect to point-by-point plotting. There is also an increase in plotting speed, averaging 60 times. The vector is drawn as a series of steps 0.004 mm in size, insuring a linearity of better than 0.02%. This is the theoretical limit and does not include the effects of the CRT, which limits the linearity to 1%.

Figure 1 presents the structure of the display system for which the vector generator was designed. Provisions were made to include a character generator within the display system by placing an analog interface multiplexer before the display unit. The display unit is a Tektronix Storage Display Unit, Type 611 (Ref. 3). Storage display units do not require refreshing. Both input and output interfaces handle data and control signals to and from the computer via the Programmed Data Channel (Ref. 4). The clock and control circuitry regulate the sequence of events within the vector generator. X and Y vector channels form the heart of the vector generator. It is here that the endpoints are stored and the calculations made to produce the straight line path between these endpoints. It is also necessary to position the beam without writing on the CRT; therefore, Z-axis control logic is provided. This logic interprets commands from the computer and turns the beam on or off.

II. Vector Generator Implementation

The vector generator was designed around the properties of a binary rate multiplier (BRM). The BRM circuit produces an output frequency that is a selectable rational fraction of the input frequency. The waveforms and logic diagram of Fig. 2 show this process. The binary counter divides the input pulse train of frequency f_i by two for each stage. AND gates (G_i) select unique transitions in this counting sequence, gating them with the input rate selection (X_i). The outputs of the AND gates are then ORed to produce the final pulse train (f_0). The average frequency of the output pulse train is

$$f_0 = \frac{\text{rate}}{2^n} f_i$$

where the rate is represented in binary by the X_i and n is the number of stages in the counter. To generate a vector, it is necessary to step from the initial point to the endpoint uniformly, or nearly so, in both axes. This can be done by counting pulses whose average rate is proportional to the difference between endpoints. For example, if it is desired to draw a vector from point $(-1, -1)$ to point $(0, +1)$, the pulse rate in the Y channel must be twice the pulse rate in the X channel.

The block diagram of Fig. 3 illustrates the functional units needed for a vector channel. A digital to analog (D-A) converter converts the digital representation of position to a voltage which deflects the writing beam of electrons in the CRT. The output register holds the instantaneous position for the D-A converter. This register is loaded from the interface with the initial point of the vector. It counts the pulse train from the BRM, thus stepping from the initial to the final point. A comparator controls the stepping process by indicating the direction to count, up or down, and by stopping the counting when the final point has been reached. The equality signal is also used by the control logic to determine when the vector is complete. The endpoint of the vector is stored in the temporary register, where it is available to both the comparator and the subtractor. Using the initial and final values stored in the registers, the subtractor computes the difference in 2's complement form. Since a BRM can accept only positive rates, the absolute value of the difference is formed using a conditional complementor.

The rate formed above could be used directly to control the BRM, but all vectors would take the same time to complete, thus making shorter vectors brighter than longer ones. To eliminate this problem, the rates for both X and Y channels are shifted left until the larger of the two reaches the most significant bit of its respective register. Since it is possible for both rates to be zero, a 1 is inserted in the least bit of each rate register to insure that the shifting process stops. The signals labeled "Load," "Shift," and "Max" control the operation of the rate register. After shifting stops, the clock is turned on, and the BRM begins producing output pulses at the proper rate. The clock is coherent with the computer clock and runs at 1 MHz. The time to draw a vector is proportional to its length on the axis plus the shifting time. Thus the minimum time will be 13 μs and the maximum will be 4099 μs.

The circuitry used for the above functions is of the medium scale integration type, packaged in dual-in-line form. These are inserted in a logic board where connections are made with wire-wrap (Fig. 4). Cascadeable BRM packages containing 6 stages, register packages of 4 bits each, and a computer word of 16 bits lead to the selection of 12 bits for register length. This was consistent with the required resolution and linearity.

III. Using the Vector Generator

The following section will assume that the reader is familiar with the MAC 16 computer and the MAC 16 assembly language, LEAP 8 (Refs. 4 and 5).

Commands to the display can be separated into two types: those that control the general display, such as "Erase," and those that refer to the vector generator itself. All of the commands are used in the following sequence:

```
LDA   DATA
EXO   UNIT,FUNCTION
JMP   *-1
```

Here EXO stands for either External Command Out (ECO), or External Data Out (EDO). The instruction following the EXO is skipped only when the device addressed has accepted the data and has acknowledged receipt of them, thus providing a ready test.

While general display commands use only the least three bits of the data word, all 16 bits are used to command the vector generator. Table 1 lists these two formats, enumerates the options included in each data type, and shows the instructions that can be used with each. The control bits, the least three for the general commands and the least four for the vector commands, operate independently and their functions will be merged. For the display commands it is necessary to specify all of the conditions required in one command as subsequent instructions will change all of these conditions. The nature of the vector commands does not place such a constraint upon their use. For these instructions the merging is for the convenience of the programmer. Bit 13 of the general display command data selects the input to the analog multiplexer. If it is a 1, the vector generator is connected to the display. A zero connects the character generator. Bit 14, when a 1, interchanges the roles of the X and Y channels producing a 90-deg rotation in the display. Bit 15 enables the interrupt pulse generated at the completion of vector generation to be sent to the computer. Bits 12 and 13 of the vector commands specify the destination for the point location data within the vector generator. This may be combined with the write/do not write (bit 14) and initiate vector generation (bit 15) subfunctions. The point location data have no meaning for these two subfunctions.

The expression field of the external command instruction is composed of two parts: the unit address or device number, which for the display system is a hexidecimal "D"; and the function or sub-address. Only 3 of the 16 possible function addresses are used with the vector generator: "3" calls for the display to be erased, "E" causes the display control bits to be accepted by the display interface, and "F" transfers the 12-bit position data to the vector generator as specified by the control bits of the word. When the data are transmitted with an EDO instruction, both the output and the temporary registers are loaded with the data. An ECO instruction only loads the data into the temporary register. Viewed in another way, the EDO instruction sets the initial point for the vector while the ECO sets the final point. It should be noted that the initial point must be loaded first as the temporary register is also loaded by the EDO command. The data formats of Table 1 and the summary of instructions in Table 2 provide a reference for the use of the vector generator. Further, it is important to note that the final point of the first vector can be used as the initial point of the next vector without reloading it, as it is remembered by the output register.

The sample program of Fig. 5 illustrates the use of all these commands. While the comments included in the program listing explain most of the points of the program, a description of the way in which these parts form a whole will aid in understanding the program. The "START" routine initializes the display system in the vector mode. It also sets it to operate under interrupt control, initializes the program for interrupt control, and forces the first interrupt from the display. This first interrupt shifts the computations to the display control level. "VECT" plots the first point on the display (as this is a unique point), and "VECTC" plots the remainder of the points. After all of the points have been plotted, the "VECTE" routine restores the computer to the normal level and disconnects the display from the vector generator. The points to be plotted are contained in a table, "TABLE," and have the control bits in the lower portion of the word as stored. The results of running this program are shown, simulated, with the points labeled, in Fig. 6. The numbers associated with each point refer to the data values in "TABLE" with the same number. A solid line indicates that a line was drawn and a dashed line indicates the path taken by the beam without writing on the scope.

IV. Conclusion

The equipment described in this report has been built, tested, and demonstrated in the laboratory in a form that can be installed in the field. The vector generator reduces the demand for computer processing time to less than 15 μs per endpoint. The high-speed, on-line display of graphic information will permit the real-time display of spectra used for signal power measurement and similar systems.

Further, this technique, with only minor hardware modifications, can be applied to a vector generator for the slower mechanical X–Y plotters, relieving a computer controlling such a device of vast amounts of processing time.

References

1. Winkelstein, R., "Precision Signal Power Measurement," in *JPL Quarterly Technical Review*, Vol. 2, No. 2, pp. 18–24. Jet Propulsion Laboratory, Pasadena, Calif., July 1972.
2. Newton, J. W., "Digital Device Development: Precise Measurement of Spacecraft Signal Power," in *The Deep Space Network*, Space Programs Summary 37-58, Vol. II, pp. 42–50. Jet Propulsion Laboratory, Pasadena, Calif., July 31, 1969.
3. *Instruction Manual for Type 611 Storage Display Unit*, Tektronix, Inc., Beverton, Oregon, 1968.
4. *MAC-16 Computer Reference Manual*, Lockheed Electronics Company, Los Angeles, Calif., Jan. 1970.
5. *LEAP Assembly Manual*, Lockheed Electronics Company, Los Angeles, Calif., Jan. 1970.

Table 1. Data format

Used with	Function	Data words 0 1 2 3	4 5 6 7	8 9 10 11	12	13	14	15
		0 1 2 3	4 5 6 7	8 9 10 11	12	13	14	15
ECO \$D,\$E	General display commands							
	Select vector generator	←	Not used	→		1	x	x
	Select character generator					0	x	x
	Exchange X and Y outputs to display					x	1	x
	Enable interrupt from vector generator					x	x	1
ECO \$D,\$F or EDO \$D,\$F	Vector generator commands							
	Select X channel	S ←		Point location →	1	x	x	x
	Select Y channel			Point location	x	1	x	x
	Z on (write)			Point location	x	x	1	x
	Z off (do not write)			Point location	x	x	0	x
	Initiate vector generation			Point location	x	x	x	1

Notes: 1. S = sign bit (2's complement form)
2. x = don't care
3. Functions may be merged
4. For vector commands: EDO loads initial point
ECO loads final point

Table 2. Display instructions

Instruction	Function
ECO \$D,3	Erase the display
ECO \$D,\$E	General display command
ECO \$D,\$F	Vector generator command load final point
EDO \$D,\$F	Vector generator command load initial point

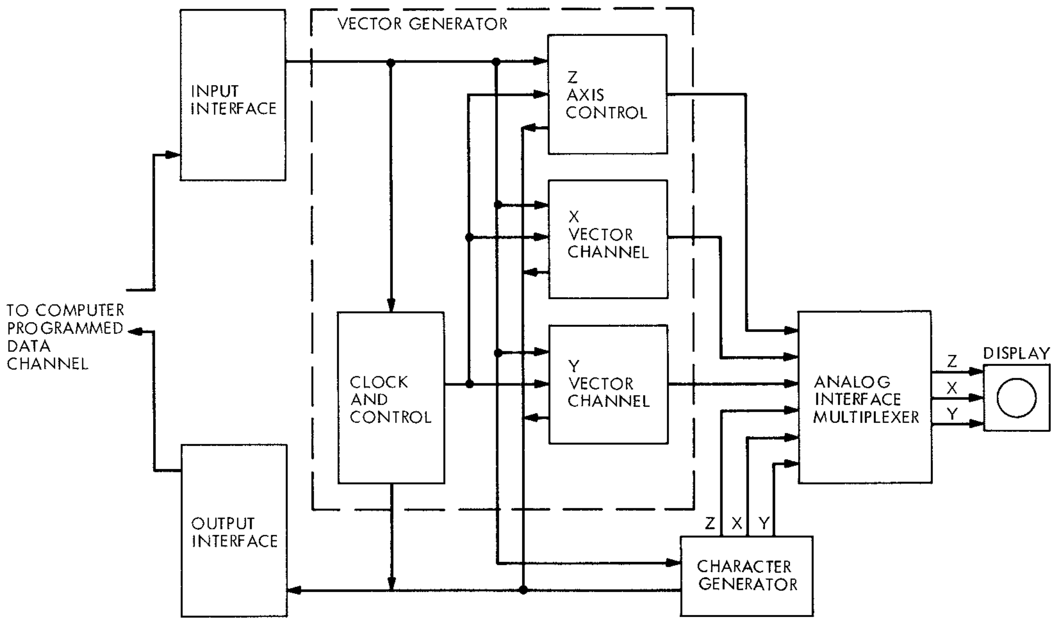

Fig. 1. Display system block diagram

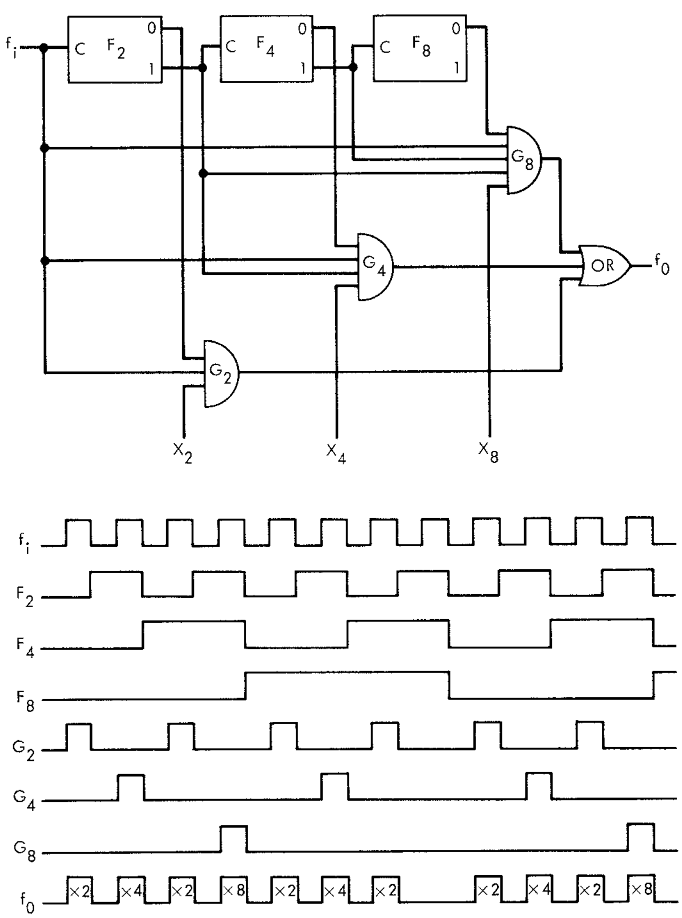

Fig. 2. Binary rate multiplier

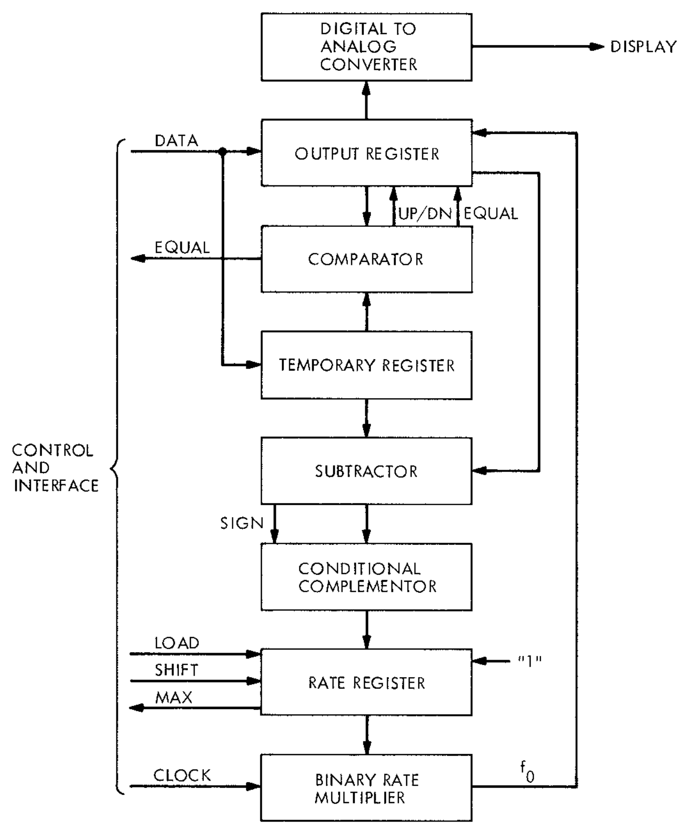

Fig. 3. Block diagram of vector generator channel

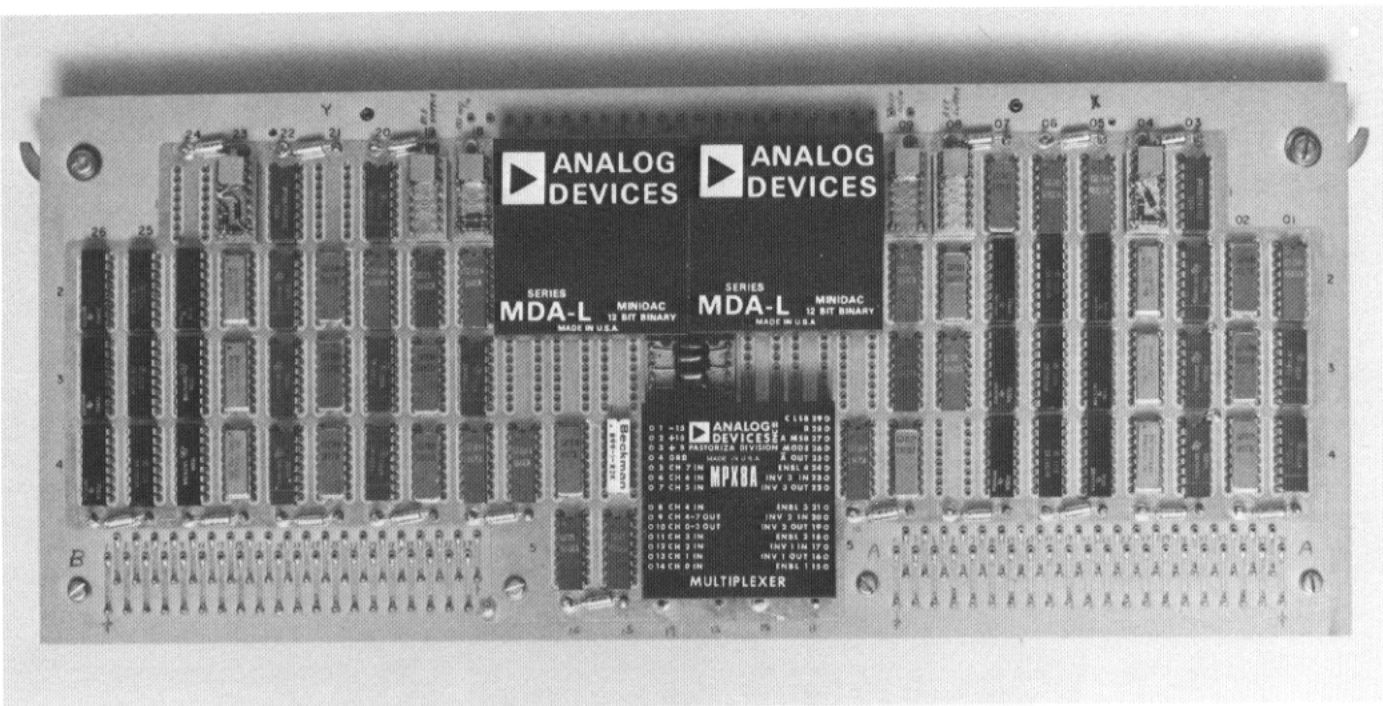

Fig. 4. Vector generator logic board

```
         BOOT
         ORG    $400
*SAMPLE PROGRAM FOR VECTOR GENERATOR
*RUN UNDER INTERRUPT
*
*INITIALIZE VECTOR GENERATOR
*
START    LDA    =VECT           ADDRESS OF VECTOR ROUTINE
         STA    *PILOC          INTERRUPT LOCATION POINTER
         ECO    $D,3            ERASE SCREEN
         JMP    *-1
         LDI    5               CONNECT VECTOR GENERATOR
         ECO    $D,$E           AND ENABLE INTERRUPT
         JMP    *-1
         LDI    1               FORCE AN INTERRUPT FROM THE
         ECO    $D,$F           VECTOR GENERATOR
         JMP    *-1
         JRL    *+1             PAUSE-HIT "SOP" TO CONTINUE
         JMP    START           GO AGAIN
*
*THE COMPUTER AND VECTOR GENERATOR ARE NOW WORKING INCONSORT
*
VECT     LDX    MN2             (X) = -TWICE THE NO. OF POINTS
         LDA    ETABLE,1        GET THE INITIAL X POINT
         EDO    $D,$F
         JMP    *-1
         LDA    ETABLE+1,1      GET THE INITIAL Y POINT
         EDO    $D,$F
         JMP    *-1
QUIT     INX    2
         JRL    VECTC           CONTINUE PLOTTING
         JRL    VECTE           END PLOTTING
VECTE    CLA
         ECO    $D,$E           CLEAR DISPLAY FROM VECTOR
         JMP    *-1
         JRL    VECT            SETUP FOR NEXT VECTOR SET
*
VECTC    LDA    ETABLE,1        GET THE NEXT X POINT
         ECO    $D,$F
         JMP    *-1
         LDA    ETABLE+1,1      GET THE NEXT Y POINT
         ECO    $D,$F
         JMP    *-1
         JMP    QUIT
*TABLE OF PLOTTED POINTS
*
TABLE    DC     $8008,$8004     1 INITIAL POINT -- LL
         DC     $7FF8,$7FF7     2 DIAGONAL FROM LL TO UR
         DC     $0,$5           3 POSITION AT MR
         DC     $8008,$3        4 DRAW X AXIS
         DC     $0,$7FF5        5 POSITION AT UL
         DC     $7FF8,$8007     6 DIAGONAL FROM UL TO LR
         DC     $8,$1           7 POSITION AT LM
         DC     $0,$7FF7        8 DRAW Y AXIS
ETABLE   EQU    *
MN2      DC     TABLE-ETABLE
         END    START
```

Fig. 5. Sample program

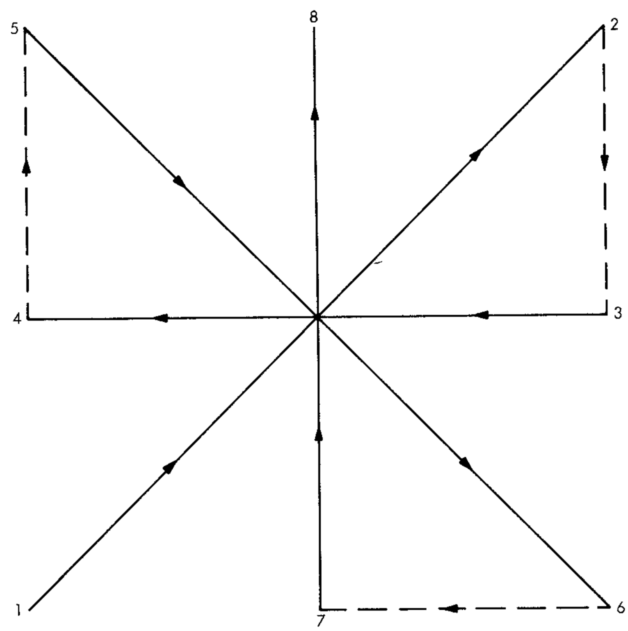

Fig. 6. Sample program output plot