FLEXIBLE FORMAT GENERATOR DESIGN TECHNIQUES

W. M. RATHBONE
Lockheed Electronics Co.
NASA Manned Spacecraft Center
Houston, Texas.

Summary A Flexible Format Generator is described for use in a Manned Spacecraft Data Management System. The design uses a memory for storing and controlling a wide variety of sampling formats. Design techniques and their benefits are discussed. Particular emphasis is placed upon functional organization. Techniques are described for combining truncated relative addresses with Mode Codes and Programming Instructions. Software counters are stressed and close attention is given to Power Strobing. These techniques should find application in a wide variety of data management systems.

Introduction As Manned Space Missions, become more ambitious the requirements for quantity, quality and variety of data increases. This places greater demands on available power and bandwidth and overloads ground receiving networks and data storage archives. This has prompted development of highly flexible sampled data systems that can adapt data management to mission needs.

Conventional systems are limited in flexibility due to the relatively fixed counting sequence of their programmers. Although some variations in format can be accomplished by changes to the Decode Matrix, these are usually quite limited and often involve hardware change.

These limitations can be overcome by the addition of a FFG (Flexible Format Generator), using a programmable memory for storing multiple formats that can be accessed as required and can be reprogrammed with only software change.

This paper proposes some FFG design techniques for minimizing power, size, and weight consistent with reliability and ease of programming. These techniques resulted from the design and development of a Laboratory Model Stored Program Data Acquisition System at the NASA Manned Spacecraft Center. The system is an outgrowth of some preliminary design concepts formulated in NASA Study Contract NAS 9-4326 by TRW Systems and presented in NTC ’67 proceedings (References 1 & 2). Some of the capabilities described were not built into the Laboratory Model but are presented as suggestions for future design.

System Description/Capability The DCCU (Digital Control & Combiner Unit) is the Nucleus of the system and manages all peripheral units. It addresses and accepts analog and Bi-level data serially through party-line RAU's (Remote Acquisition Units) and controls Serial Digital data inputs. An External Control interface allows an Onboard Computer or Data Compressor, on command, to randomly address and accept data via the DCCU. An FFG is added to provide the DCCU a wide variety of stored formats. The DCCU contains a conventional fixed format Counter/Decoder; a channel Address Register; a variable Bit Rate Generator (1-100 Kb/s); a CSR (Command Storage Register) to control operating mode, bit rate, and modulation code; and a Command Processor to decode and process incoming instructions (FFG memory programming or CSR up-dating). The CSR can also be controlled by Discrete (switch) inputs.

System operating mode determines DCCU Address Register control. In the Fixed Format Mode, the Counter/Decoder controls the register in a fixed sequence. In the Flexible Format Mode, the FFG controls the register in any sequence desired. In External Mode, the register is slaved to an Onboard Computer or Data Compressor. Selected time slots can be allocated for CA (Computer Access) via the external control interface. This is especially useful for on-board status checks and for monitoring time-critical crew or equipment parameters when contact with receiving stations is delayed or obscured. A FID (Format I. D.) channel monitors the CSR to notify the ground station prior to format changes.

Each RAU can accommodate 64 analog and 64 bi-level channels. Thus a 10 bit channel address code can address any of 8 RAU's, with 3 bits, and specify within an RAU any of 64 analog/bi-level channels, with 7 bits. This gives a total system capacity of 512 analog and 512 bi-level channels. An 11 bit channel address code would double channel capacity with 16 RAU's. All system coding is binary.

FFG Description The FFG, shown in figure 1, consists of a 12 bit X 2048 location random access NDRO (non destruct readout) memory and associated control logic for storing and generating formats for the DCCU.

The FFG has two basic modes of operation: Standby and Flex. The Standby (Store/Verify) mode is used for loading and/or verifying the contents of memory. The Flex mode is used for supplying the DCCU with stored Address Codes (Analog, Bi-level, Serial Digital, FID, CA) and stored Data Codes (Frame Sync, Frame Count, Subframe Count) in any format desired. The number and length of formats that can be stored at any given time is limited only by memory capacity. Memory can be reprogrammed, in flight, during a mission.

In implementing such a design, lowpower integrated circuits should be used wherever practical. Considerable design effort should be devoted to minimizing power

consumption by use of power strobing, with close attention to functional organization and interface coordination with DCCU and memory designers.

Interface lines should be held to a minimum. For example; in figure 5, mode control is established with four lines. Codes 1 through 12 specify a Flexible mode and also form the four active bits of a truncated address for the format Start Cell (the missing bits are automatically zero). The remaining four codes indicate Standby modes to the FFG; one for Fixed Format, one for External Control, and two spare codes. Programming and verification is interfaced serially through the Store/Verify shift register. Address and Data codes to the DCCU are interfaced in parallel (10 lines) between the FFG I/O register and the DCCU Address/Data register. Other control lines and power bring the total to 30.

The Timing Generator is a counter driven from the DCCU clock. It is powered-up and started with the DCCU word rate sync pulse. The sequence length is established by the shortest process time; longer processes are accomplished by allowing the counter to recycle. The longest process (3 cycles) is used in restoring initial sub-subcom Jump Addresses. By borrowing DCCU Clock and using a recycle counter, Timing Generator power and components are minimized.

The 16 bit Shift Register serves several needs - In the Standby mode it is used for Up-link Programming and Down-link Verification. In the Flex mode it is used for momentarily holding subcom Jump Addresses. Multifunctional use minimizes power and logic.

In the Standby mode, cell location Zero is used in conjunction with the Address Counter to store its contents between DCCU programming and verification commands. This allows the Address Counter to be powered-down between commands. Cell Zero is thus designated as the “Address Count Save Cell”, behaving as an alter-ego for the Address Counter. Since this feature requires one Read/Write cycle, some duty cycle trade-offs would be required to justify its use in the Flex mode --- especially at high word rates.

Addresses from four sources and data from two sources are controlled through address and data Selection Gates. These gates are power gated on only during Read/Write cycles (approximately 1.5 μs). All control logic for loading or verifying memory is powered-down during Flex mode while memory is in use. Most of the process steps in all modes are sequentially power gated from the Timing Generator at 3.0 μs intervals.

Memory Organization Cell locations Zero through Twelve are “Absolute Locations” that are normally accessed with a truncated (4 bit) address from the DCCU Mode Code lines or the message instruction bits in the Shift Register. Cell Zero is an Address Count Save Cell and also the transfer point for random access to memory (explained under

programming). Cell locations One through Twelve are called Format Start Cells and form the index of available format starting addresses. The remaining locations in memory are allocated to completely unrestricted format loading.

Mainframe, subcom, and sub-subcom lists may be loaded anywhere in any order; this simplifies programming. “House keeping” lists or others requiring multiple use, need be stored only once; subsequently they can be used by any format and can also be repeated within any format. This increases storage efficiency.

An example of a format containing two Subcoms is shown in figure 2. To enter the format the DCCU Mode Code 0011, accompanied by a word rate sync pulse, addresses Start Cell #3. This cell contains the format starting address (S), prestored, which is then transferred to the Address Counter. The Address Counter increments to S + 1 and reads the first word in the Main Frame list. Successive word sync commands step through the list to S + N EOF (End-of-Frame) which refers to location #3 whereupon the starting address is restored to the Address Counter and thus the format repeats. Subcom “b” is addressed twice during a frame and subcom “a” once. The sub-subcom list is addressed from subcom “all once per subframe. The subcoms in this example were stored in the low ordered cell locations; however, they could be stored anywhere and also accessed by other main frame lists. In the same manner, the sub-subcom list could be shared by other subcoms.

If a format is to be entered manually (discrete command), a Manual Select signal accompanies the Mode Code which allows the format starting location (S) to be Read (before incrementing) and its contents transferred to the DCCU Command Register. (S) contains the Command Word (bit rate, mode, etc.) for the manually selected format.

The foregoing illustrates the simplicity, flexibility, and. storage efficiency that can be achieved.

Memory Programming/Verification In Fixed Format mode, memory can be stored and verified. Figure 3 shows the Command Message Structure and the Process Flow for the store sequence. Immediately upon detecting an FFG message, the DCCU Command Processor sends a Store Gate signal to the FFG that interrupts any verification that may be in progress; powers-up the Shift Register; and inhibits all other commands from entering the FFG until that message has been stored. The message, accompanied by a clock, is entered into the Shift Register. With proper Parity, an Execute pulse powers-up the remaining logic after which the message
is processed and all logic powered-down.

The instruction repertoire is limited to four basic operations: Prestore, Reload, Sequential Load, and Readout. The Prestore instruction loads the format Starting

Address (SR data bits) into the format Start Cell (loc. 1- -12) and also into location Zero. Thus the format Starting Address is already accessed for subsequent loading. A Reload instruction fetches the address that was stored in Zero, places it in the Address Counter, and loads that address with the data bits held in SR. Sequential Load increments the address stored in Zero before loading SR data bits. Subsequent verification commands will verify the contents of only the location whose address was last stored in Zero. To resume Sequential Verification, a Readout instruction is sent that resets the Post-Store Flip-Flop.

Random access to any location in memory is accomplished with a Prestore instruction. This is usually done through cell Zero to prevent disturbing Start Cell data. As soon as a location is accessed, Down-link verification words reveal its content. A Reload instruction will correct or up-date the contents.

Three consecutive time slots are allocated periodically to transmit memory cell content and its associated address for verification purposes. This process is shown in figure 4.

Up-link/Down-link interfaces were simulated for the Laboratory Model System with a PDP-8 Computer. Programs were developed for Automatic Loading, Echo Checking, and Status Monitoring.

Mode Control Mode Decode logic in the FFG monitors the DCCU Mode Code lines shown in figure 5. In Flex mode the FFG looks to the Mode Code during every EOF branch process for the proper Start Cell address.

The format's Starting Address is always restored during the last portion of an EOF cycle. If a Flex format is manually selected, the discrete for the new format is decoded (top of figure 5) which generates the new Mode Code and sets the Manual Select latch. This momentarily enables the Manual Select gates and disables the Command gates (both DOT OR'ed with the initialization gates). The new Mode Code, along with the Manual Command signal, is processed during the first portion of an EOF cycle. The new Starting Address location is read into I/O and its contents (the command word for the new format) transferred to the DCCU CSR. The Command Word transfer signal also resets the Manual Select latch, allowing the existing format Starting Address to be restored during the last portion of the EOF branch and operation continued until the CSR Output Register is up-dated with the New Command Word --- normally at EODC (end-of-data-cycle). Manual select from a Standby condition momentarily places the FFG into a Flex mode EOF condition (for approximately 12 As) while the Command Word is fetched from memory. The FFG then powers-down and waits until the CSR output is up-dated.

Change over to a new format normally occurs at EODC; however, especially at low bit rates, the FFG may be requested to change formats immediately. This leaves the previous

format uncompleted and its subcoms and sub-subcoms may be out-of-sync. To minimize data loss upon re-entering that format, provisions are made for initializing it. Initialization is commanded (bottom of figure 5) from a register analogous to Manual Select. The Initialization Command forces the FFG Timing Generator to recycle continuously, thereby processing the selected format at the highest possible rate. At this rate, the maximum (worst case) time for initializing a 128 word X 50 frame format is approximately 50 milliseconds. The Initialization Complete signal resets the initialize latch and normal operation is resumed. Formats that get out-of-sync during Flex mode (from power faults, glitches, etc.), are automatically commanded into initialization, internally, by the FFG Power Detect or Sync Detect logic. This helps to keep data loss to a minimum and eliminates the necessity for outside intervention; this is especially convenient if the up-data link is occupied with other tasks.

Mode Control was purposely illustrated and described in considerable detail to show its relative simplicity and efficiency, with a minimum of logic and interfaces.

Flexible Format Processing The Process Flow shown in figure 6 is straightforward; however, the power strobing should be emphasized. The only logic continuously powered-on in this mode is the Address Counter, the process latches for establishing branch paths, and the subcom and subsubcom sync detection/restoration latches. The word rate pulse powers-up memory, control logic, and timing. If a main frame word is processed, it is transferred to the DCCU within 9 μs and logic powered-down until the next word rate pulse arrives. When a subcom jump address is recognized, the subcom branch is established and the subcom Address Register powered-up. The subcom word is transferred to the DCCU within 21 μs. Sub-subcom words are processed in a similar manner within 33 μs. Process times were included to give the unfamiliar reader an appreciation for the efficiencies that can be achieved with power strobing.

A sample format is shown in figure 7. The Main Frame is commutated into 21 channels of which 4 and 14 are subcommutated into 4 frames and 8 frames respectively. Channel 14 is further sub-subcommutated into three subframes at frame 6. Word 1 is a stored Frame Sync code. Word 2 is a Frame Count code. During the first frame, when cell 2 is read into I/O, the I/O contents are cleared except for Op code and a zero count restored to MF cell 2. On subsequent frames, the I/O contents are incremented by one before restoring, thus producing a binary frame count. Word 3, 5 thru 13 and 15 thru 21 are Main Frame channel addresses. Cell 4 is the Jump Address cell for subcom list #1. This Jump Address, read into I/O, is also transferred into and held by the Subcom Address Register. One is added to the address bits in I/O (making them the Jump Address for the next subcom word) which are now restored to cell 4 (still addressed by the Address Counter). The original Jump Address, held in the Subcom Address Register, is now used to address the first cell in the subcom list. Thus, in each successive frame, the contents of cell 4 are incremented for use in addressing each successive word in the subcom list.

When subcom cell 4/5 is read, its Op code identifies it not as a subcom word but as the Initial Jump Address which must be restored to main frame cell 4. Timing is recycled and the first word in the subcom list is processed again as above. Subcom #2 is processed in the same manner. In a like manner, sub-subcom lists are addressed from an incrementing Jump Address stored in a subcom cell. Subframe Count (14/1) is handled similar to Frame Count; however, its contents are incremented at the subframe rate and restored to a subcom cell. Extra sub-subcoms were excluded from this example for the sake of clarity. There are several methods of detecting and restoring sub-subcom sync; however, these impose some restrictions on their placement within the format. These restrictions can be eliminated at the sacrifice of automatic sync detection and restoration, i.e. by requiring outside intervention.

Subcom synchronization is accomplished by making all subcoms some multiple of the longest subcom. Thus, all subcoms will end in the same frame in which the longest subcom ends. Sub-subcom sync is accomplished in a similar manner on a subframe basis.

A 12 bit memory Bit Assignment chart is shown in figure 8 for controlling 8 RAU's (10 bit address code). This chart was included to show that Op codes can be stored in the same cell location as their associated data which allows efficient processing in terms of speed, logic, and memory. However, a 16 RAU system using the same memory and requiring 11 address bits necessitates placing the Op codes in separate cell locations immediately ahead of their respective data. Increasing memory word length would overcome this difficulty.

Conclusions FFG design techniques have been described that minimize power, logic, and interfaces while enhancing reliability and ease of programming.

Memory has been organized to allow up to a dozen formats to be accessed either manually or by command with no further programming. A single prestore instruction alters the format index to allow access to other formats.

The use of an Address Count Save Cell was described that conserves Standby power. Software counters for frame and subframe identification were described that simplify design and conserve both logic and power. A highly efficient mode control scheme was described that minimizes both logic and interfaces.

Although automatic synchronization can easily be provided, future efforts should be applied to minimizing restrictions on sub-subcom placement and elimination of any necessity for outside intervention.

Future requirements will place an increasing premium on design efficiency. It is hoped that some of these techniques will help fill that need.

Acknowledgements. -The author wishes to thank his Lockheed associates; especially Mr. D. E. Noble who implemented the DCCU and RAU designs and Mr. G. R. Smith who developed the Computer Simulation programs.

Thanks is also due Messrs. C. D. Levy, W. E. Mallory and A. W. Young within the NASA Digital Techniques Section for their helpful suggestions and system concepts.

References

1) Spacecraft Digital Data Processor Final Report - November 1966 prepared for NASA under Contract NAS 9-4326 by TRW Systems.

2) C. Levy, A. Egger, J. Stevens - Concepts for a Stored Program Data Acquisition System; National Telemetry Conference Proceedings 1967.

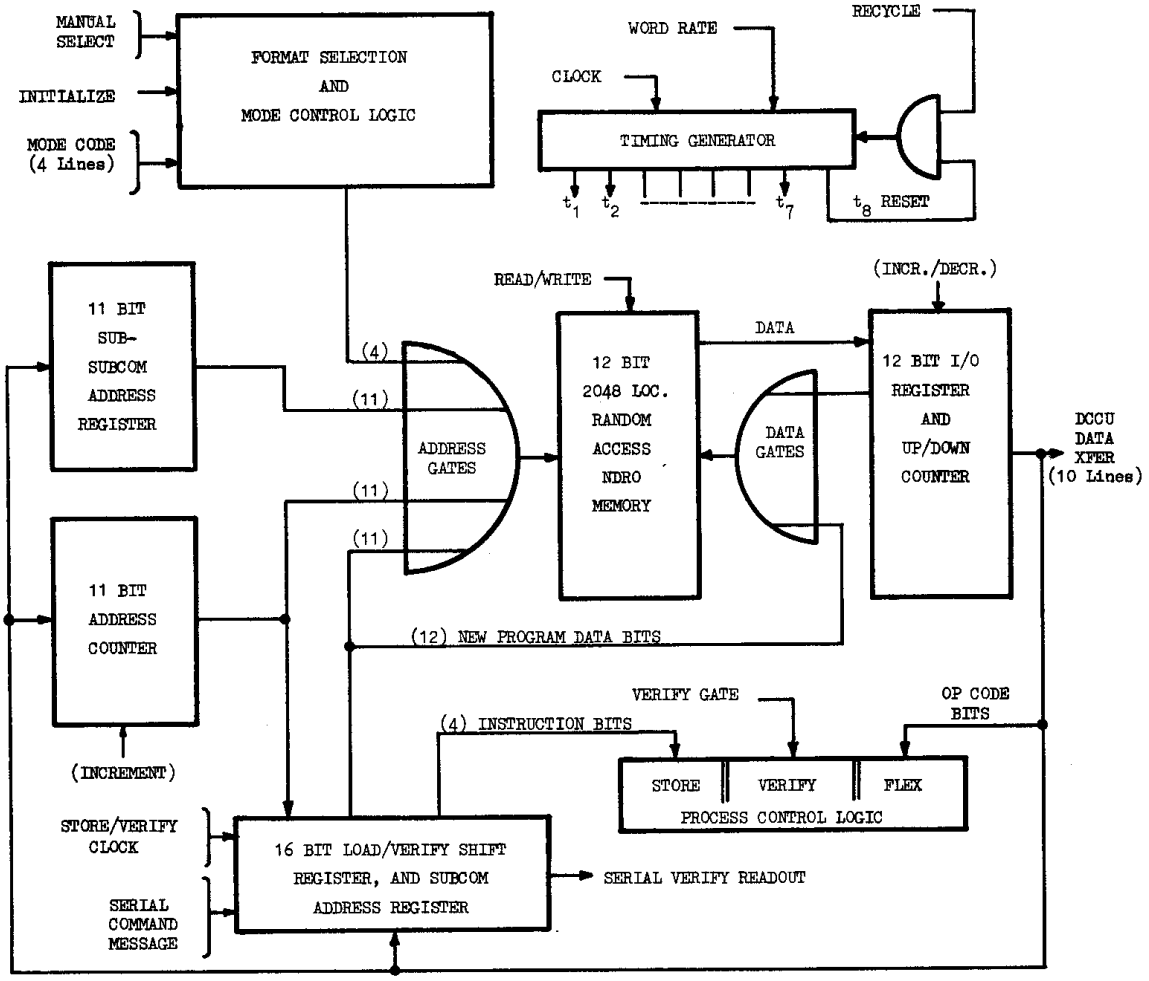

FIGURE 1 - FFG BLOCK DIAGRAM

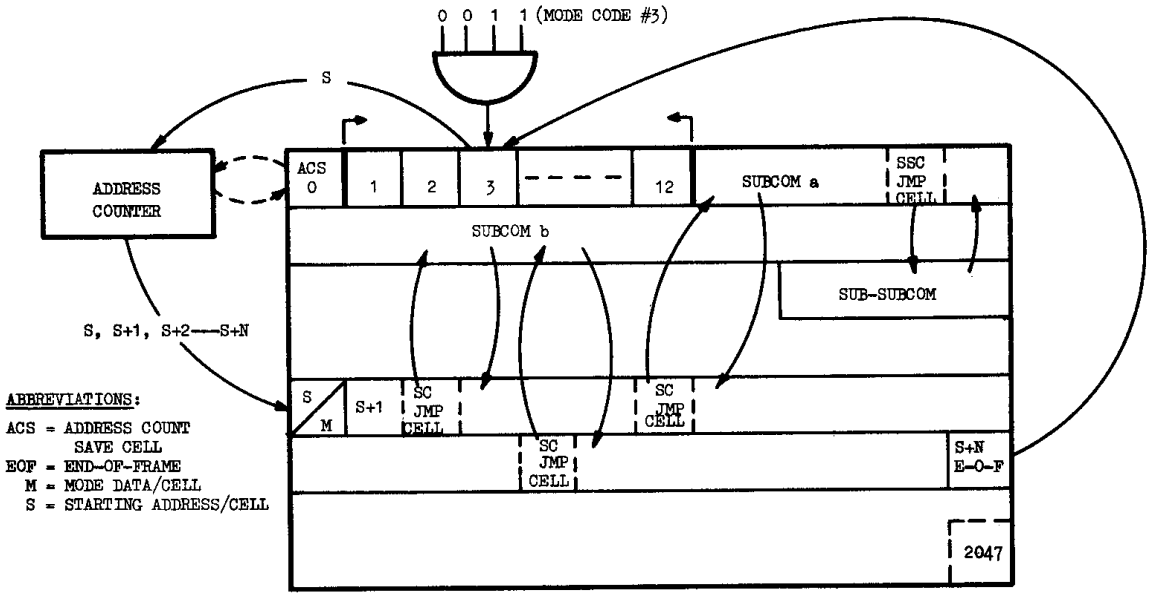

FIGURE 2 - MEMORY ORGANIZATION

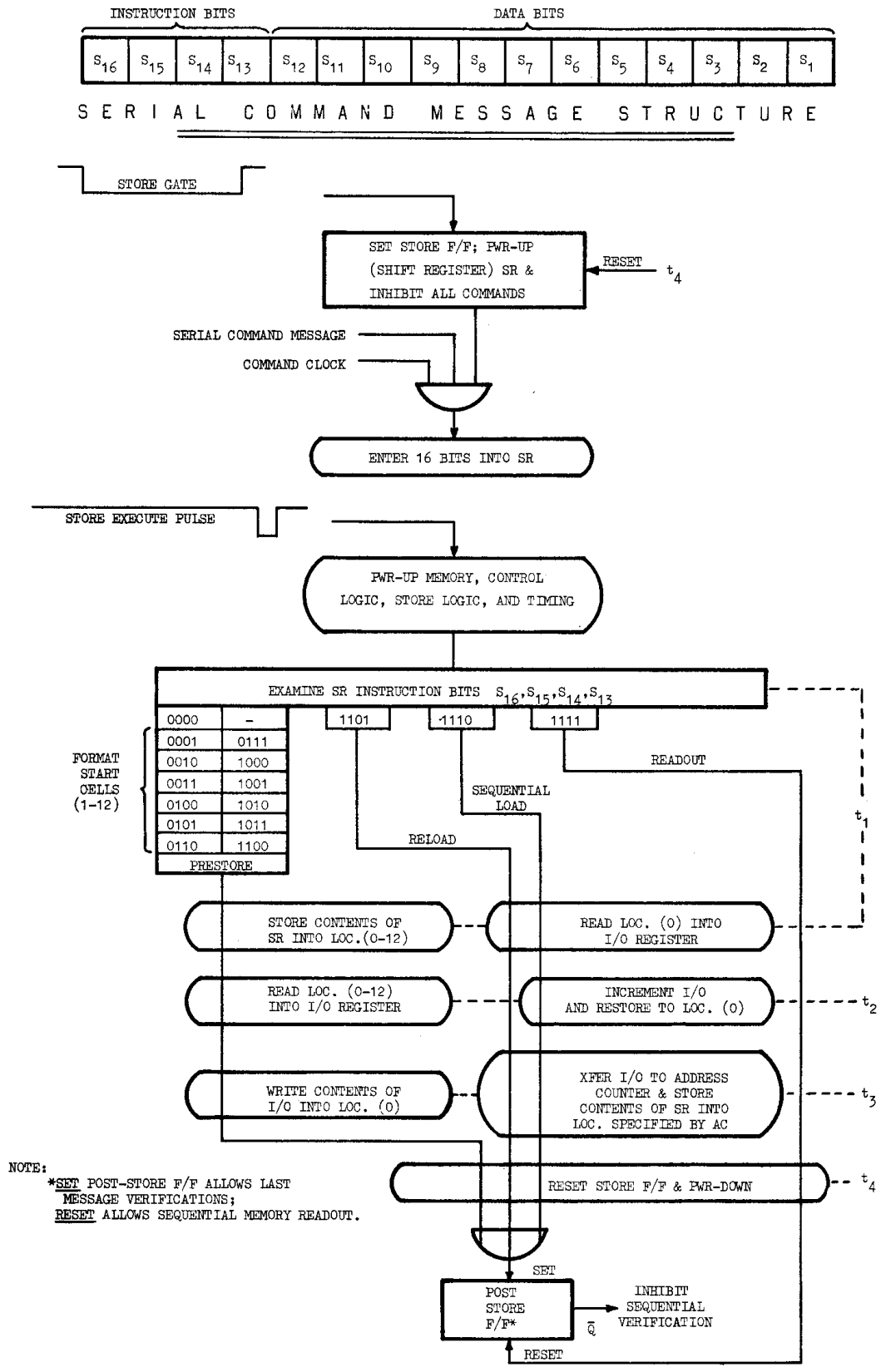

FIGURE 3 - STORE MODE PROCESS FLOW

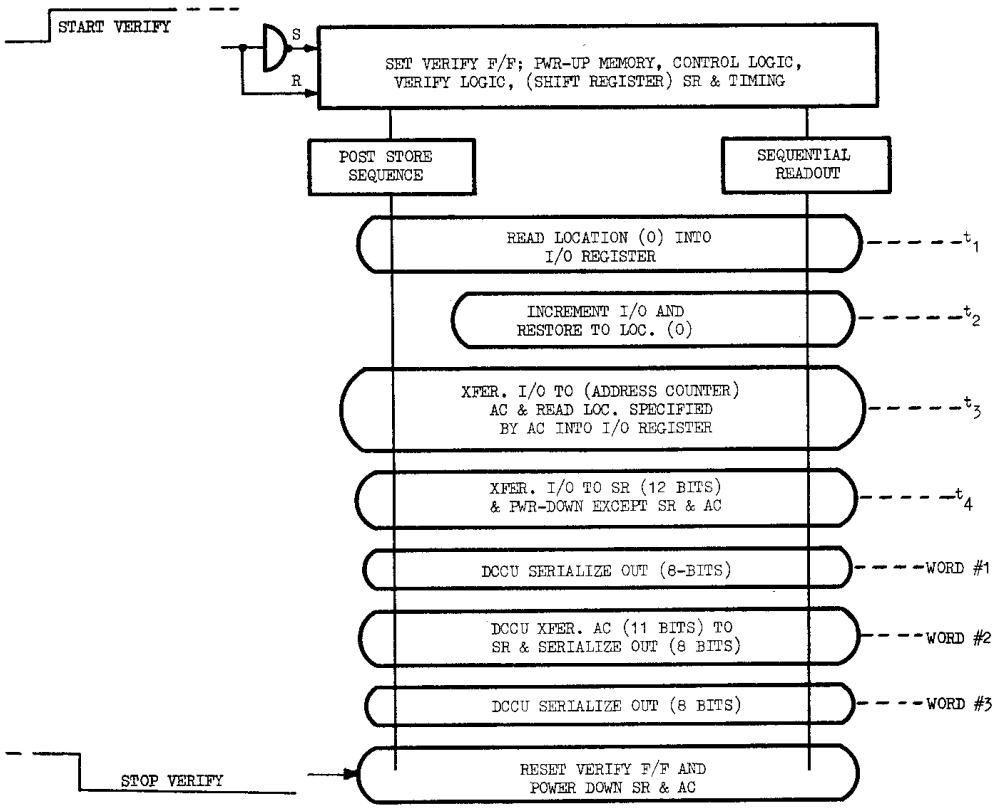

FIGURE 4 - VERIFY MODE PROCESS FLOW

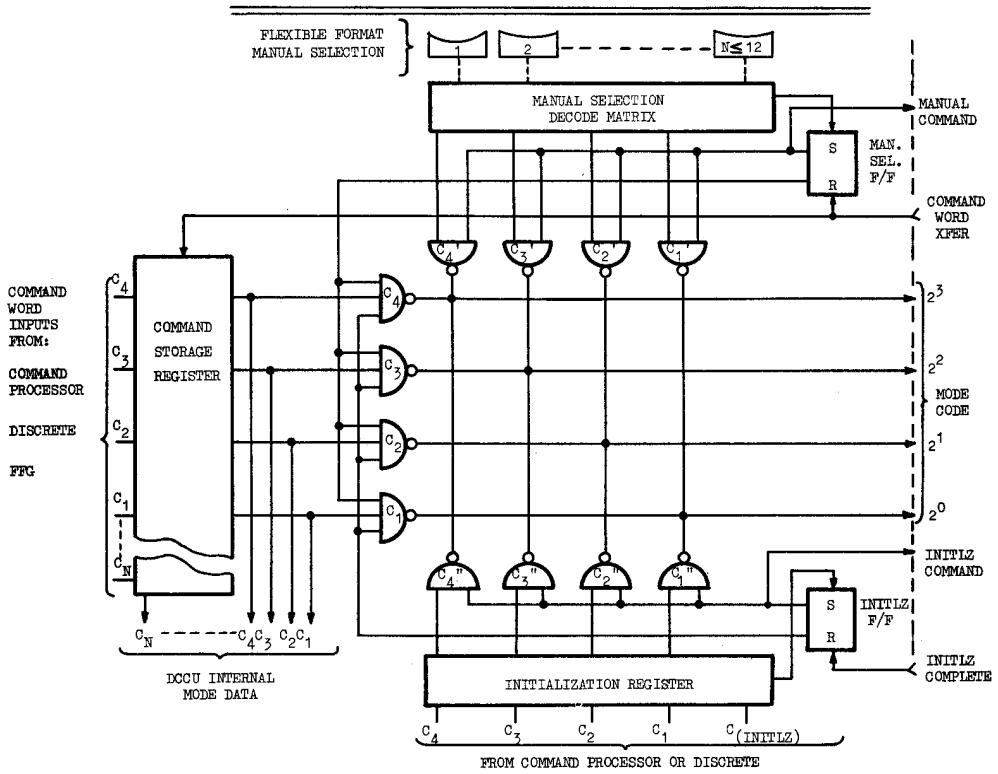

FIGURE 5 - MODE CONTROL LOGIC

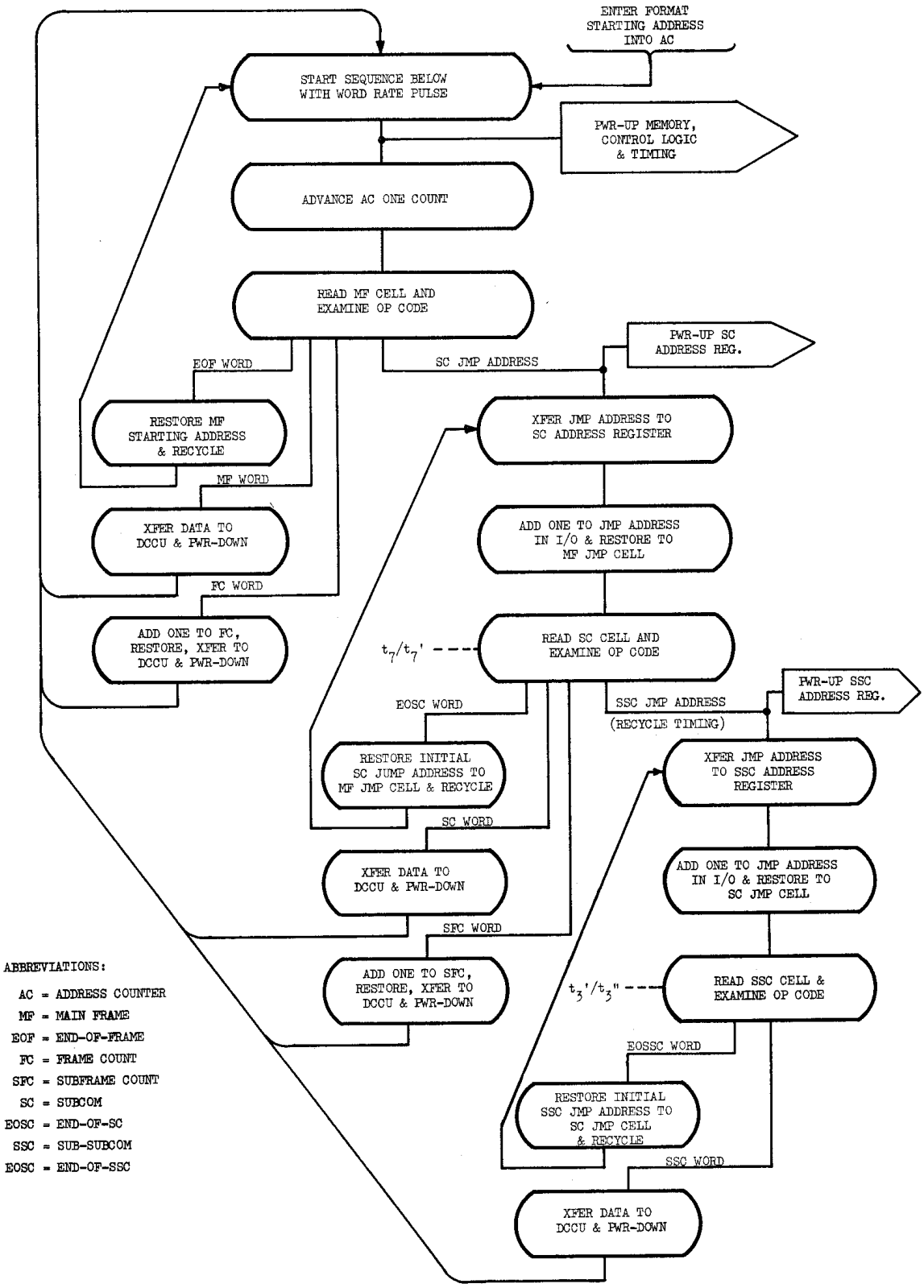

FIGURE 6 - FORMAT PROCESS FLOW

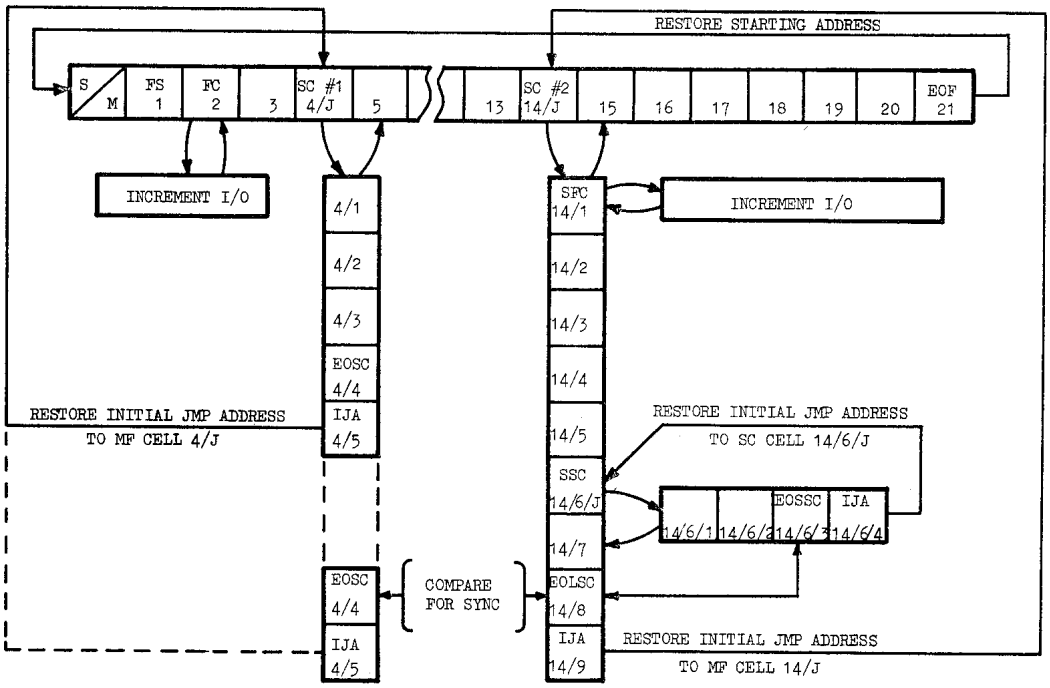

FIGURE 7 - SAMPLE FORMAT

MEMORY CELL BIT ASSIGNMENTS APPEARING IN THE I/O REGISTER DURING:

		12	11	10	9	8	7	6	5	4	3	2	1
		OP CODE		DATA BITS									
MAIN FRAME BRANCH PROCESSING	MF WORD	0	0	A	A	A	A	A	A	A	A	A	A
	EOF WORD	1	1	A	A	A	A	A	A	A	A	A	A
	SC JUMP ADDRESS	0	1	J	J	J	J	J	J	J	J	J	J
	FS WORD	1	0	1	1	S	S	S	S	S	S	S	S
	FC WORD	0	0	1	1	C	C	C	C	C	C	C	C
		FS, FC & SFC OP CODES USE BITS 9 - 12 BITS 1 - 8 ARE SYNC/COUNT CODES											
SUBCOM BRANCH PROCESSING	SFC WORD	0	0	1	1	C	C	C	C	C	C	C	C
	SC WORD	0	0	A	A	A	A	A	A	A	A	A	A
	EOSC WORD	1	0	A	A	A	A	A	A	A	A	A	A
	EOLSC WORD	1	1	A	A	A	A	A	A	A	A	A	A
	SSC JMP ADDRESS	0	1	J	J	J	J	J	J	J	J	J	J
SUB-SUBCOM BRANCH PROCESSING	SSC WORD	0	0	A	A	A	A	A	A	A	A	A	A
	EOSSC WORD	1	0	A	A	A	A	A	A	A	A	A	A
	EOLSSC WORD	1	1	A	A	A	A	A	A	A	A	A	A
EOF BRANCH DURING MANUAL ENTRY	COMMAND WORD	X	X	X	T	B	B	B	B	M	M	M	M
		SPARES			TLM	BIT RATE				MODE			

ABBREVIATIONS:

A = ADDRESS CODE BIT
B = BIT RATE CODE BIT
C = COUNT CODE BIT
J = JMP ADDRESS BIT
M = MODE CODE BIT
S = SYNC CODE BIT
T = TLM MODULATION BIT
X = UNASSIGNED BIT

MF = MAIN FRAME
EOF = END-OF-FRAME
FS = FRAME SYNC
FC = FRAME COUNT
SFC = SUBFRAME COUNT
SC = SUBCOM
EOSC = END-OF-SC
EOLSC = END-OF-LONGEST SC
SSC = SUB-SUBCOM
EOSSC = END-OF-SSC
EOLSSC = END-OF-LONGEST SSC

FIGURE 8 - MEMORY CELL BIT ASSIGNMENTS