A DIGITAL DATA INTERLEAVER

JOHN E. GOODWIN
Martin Marietta
Denver, Colorado

Summary. Many future space programs such as Space Shuttle contain a number of payloads, each generating its own digital data stream. It is frequently desirable to combine these data streams in a composite serial stream for the telemetry down link. The circuitry for combining these data streams should be as transparent as possible to the design and operation of individual data source subsystems. A concept is described for interleaving the data of several sources without any subsystem synchronization, few limitations on data rates, a no restrictions on formats. All data are accepted without loss and the composite stream is formatted in accordance with IRIG standards. The interleaver requires the use of artificial fill data to assure the possibility of accepting and formatting asynchronous data symmetrically; therefore, methods of error detection and correction of fill words are discussed to ensure nonambiguity of data and fill work.

Introduction. Multiple payload space flights are common. The payloads are frequently supplied as subsystems from several vendors. Usually each payload (experiment, etc.) provides performance and status information that ideally can be furnished as a unique payload data stream. Ideally these data streams can be supplied by the contractors of each payload independently of the other payload designs. This minimizes the Project Integrator's problems in correlating payloads.

Later sections of this paper show that a payload data interleaver can be implemented practically so that few constraints are imposed on each subsystem designer. Requirements common to each payload designer are as follows:

1) The experiment data rate shall be specified to the Integrator with a reasonable tolerance, e.g. ±1%;

2) The data shall be autputed at compatible signal levels, typically TTL levels;

3) The Integrator must be given a description of the individual data formats.

Although there are no constraints on the individual formats necessary for the interleaver, it is possible that a total system restriction will exist that specifies an IRIG or other standard.

No subsystem-subsystem data synchronization is required and no interdependent frequency specifications are necessary, except that the Project Integrator must specify and coordinate permissable maximum data rates.

Because the interleaver concept does not require synchronization of experiment data streams, the time occurrence of data transitions is not known. Data, therefore, will not always be ready for interleaving at the appropriate time slot unless there is excess time always available to allow for uncertainties. This delay would cause inefficient use of the data system. There are other approaches to the problem and are discussed here. One efficient approach inserts artificial fill words when real data are not ready.

There is some possibility that fill words can be confused with data and there are several ways to minimize this. If the fill word is selected from available tables of optimum length words with low autocorrelation, the probability of ambiguity is law. But an even lower probability may be desirable, especially in the presence of noise. In this case some type of error detection and correction may be desirable. An approach to detection and correction is presented here.

General Technical Considerations. The data interleaver is designed to accept any combination of one to seven NRZ-L digital data channels whose total data rates are in the range of 200 b/s to 256 kb/s. The design is one that imposes a minimum of constraints to the design and to the configuration of the data sources. In this respect, it is assumed that the individual data channels are totally asynchronous and don't necessarily have any frequency or harmonic frequency relationships. No restriction is imposed on the ratio of frequencies of the lowest to highest data rate channels; however, a reasonable tolerance on the stability of source data and clocks must be assumed, e.g. ±1%. For the purpose of later discussions, data and a clock will be assumed from each data source. However, if certain design requirements specify that source data are presented in biphase-L form with no clock, the design could be modified for internal input clock generation.

The data interleaver combines and formats incoming data into a single serial output data channel. Combining and formatting of data are done without the loss of any data. Overhead necessary to synchronize and identify all data, irrespective of data pattern, is incorporated in the serial data stream. Any format configuration compatible with incoming data and IRIG standards can be accepted by changing an inexpensive ROM stored program. Data are outputed in a Manchester biphase-L format via a balanced line driver with an output impedance matched to the modulator or other destination as required. For the purposes of this discussion a TTL-compatible output will be assummed to drive about 75 ohms.

Design Approaches. There are several ways that a data interleaver can be implemented, but all designs must be implemented so that input buffers don't overflow causing a loss of data. Any design selected should also be adaptable to any general format in lieu of specific source data definitions. Designs should be made to use a minimum amount of overhead for reliable performance to keep data handling efficiency high. The designs should also permit flexibility in format selections so that hardware can be easily adjusted to a variety of missions. A few possible tradeoffs for these criteria are as follows

1) Avoiding input buffer overflow - If the input buffers are never to overflow, the interleaver output data rate must always be slightly higher than, or precisely equal to, the total input rates of data and format overhead. The amount of buffering then must be adequate to handling the storage of several words in each channel to accommodate the maximum number of words that must be stored while interrogating the other channels. The buffering also must accommodate the condition where asynchronous data are not quite ready when they are requested (thus adding almost an extra word to the storage before the sample time comes around again). If the data source rates are known and constrained to reasonable values, the interleaver clock can be programmed to a slightly higher (1%) rate than the total input rate plus overhead. A voltage-controlled oscillator (VCO) can be referenced to a precise crystal reference oscillator and programmable digital dividers can be used in the control loops and VCO output to obtain any desired frequency. Clock circuits can be preprogrammed with replaceable read-only memories (ROM) or inflight programming of random access memories (RAM) if a processor control system is used. Flexible input buffeting can be easily implemented bus using "First In-First out" (FIFO) memories. The FIFO receives data at the source frequency, ripples it through to the output, and clocks it out at any design compatible rate. Although all inputs are asynchronous, only the FIFO outputs will be synchronous. The length of the FIFO is selected for worst case conditions that are determined during the design phase; however, data ripples to the output as it is received and they are in effect variable length buffers.

   An interleaver is well on its way to operating when the clock and buffer designs already discussed are used but a problem of data timing still exists. Although input data may average a total rate, individual channels will not be harmonically related and frequencies may run both faster or slower than nominal. This condition means data may not be ready when requested in the format. There are several ways to handle this situation. One way is to continuously compare input data rates to the interleaver clock rate and periodically correct the interleaver clock to provide a "channel adjusted" input/output data ratio. Another way is to operate the interleaver with fixed dead time intervals of sufficient length between channels so that "late data" can be awaited. The first approach is relatively complicated and the second may be inefficient and leave dead times in the output data stream that may be adverse to clock recovery or

synchronization. A third approach, which appears to be a compromise to solving problems, is to insert a fill word when data are not ready. An overhead bit is used to distinguish fill and valid data. This bit may be either appended to each word or these "status" bits in the discussion of the baseline design.

2) Flexible formatting - IRIG standard 106-73 is rather broad in selection of formats, especially as related to as small a number of input data sources as seven. It appears feasible to implement any combination of one to seven data channels in a symmetrical format of IRIG compatibility, including all overhead. It is assumed that all data will be assembled in minor frames and major frames as necessary. A optimum sync word can be used to identify each frame and special words as are permitted can be used to identify fill/ valid data where necessary. If supervisory control is desirable, the formats can be stored in electronic programmable RAMs or preflight interchangeable PROMs can be used, tailored to the mission.

   It is important that sync words and fill words be accurately identified because the loss of sync or ambiguous recognition of fill words will certainly impair data recovery. An optimum length word is probably suitable for sync because the location of sync is repetitive. If sync is once achieved in a 2-,or 3-bit window, there is a high probability of correct sync being maintained even with a number of errors. However, the location of fill words may almost be random and not predictable. The probability of recognizing random data, such as a fill or masking a fill by errors, is relatively high and the intervals between mistakes in a high rate system might be only a few minutes. If a status word, however, is inserted in the data stream periodically with a bit to identify the validity of each word of a block of words, the location of this status word will be known. The system will, therefore, know when to look for valid/fill identifiers. Furthermore, the status word can contain error detection and correction bits. Thus there will be a high probability that the status word can be recognized and interpreted correctly. Operation for several hours without errors is feasible if an error detection and correction method is used. The exact error-free time depends on the signal-to-noise ratio of the system and the numbers of bits to be corrected. The probability of error-free performance increases with better signal-to-noise environments and increased numbers of bits corrected. Increased error correction adds to circuit complexity and tradeoffs need to be made that are beyond the scope of this paper.

Baseline Design. The selected baseline design has circuits similar to some of those that have been implemented in the fabrication of a High Rate Multiplexer. This design is diagramed in Figure 1. The input of the data interleaver is shown at the left of the diagram. Two of seven identical input channels are shown. Input data are assumed to be single-ended digital NRZ-L and a clock is assumed to be available for each data stream. No data rate restrictions are imposed on any channel except that the total data rate will not exceed

256 kb/s nor be less the 200 b/s. Any number of input channels from one to all seven can be used.

All data and clocks are received by single-ended TTL-compatible line receivers. If final design considerations dictate, the receivers can be modified if necessary, and biphase-L to NR-L conversion circuitry will be added between the input receiver and first buffer register. If no clock is available with the data, provisions for internal clock generation, synchronous to incoming data, will be provided.

As data are received they are clocked in serial-to-parallel buffer registers. Data are then transferred in 24-bit bytes or words to FIFO registers. The FIFO registers shown in the diagram are typically three-paralleled 32x8 registers that appear more than adequate for all data configurations. Data are clocked into each FIFO at the input rate for each channel regardless of the output rate of the FIFO. As the data are received the byte counters count 24 clock periods of received data and a data transfer pulse is generated to load a 24-bit word into the FIFOs.

The outputs of FIFOs from all channels are connected through tristate multiplexer switches to a common 24-bit parallel bus that feeds a parallel-to-serial output register. Data are selected from each channel for transfer to the serial register by a PROM-stored format shown at the right side of the block diagram. A format for a major frame of data is stored in the PROM. A word counter sequentially addresses the PROM address locations. As an address is selected, the corresponding output of a demultiplexer is enabled to generate the channel address signals 1SL through 11SL. The channel addresses 1SL through 7SL and ANDED with output-ready signals of the FIFOs in the addressed channel to generate signals S1 through S7 respectively. Signals S1 through S7 enable the output multiplexers as data are requested. If the data in a FIFO are not ready when a channel is addressed, the ready signal will not be true and the S-signal won't be generated. A bit will then be shifted into the status register saying that data weren't available and the $\bar{S}$ (not S1, etc.) will enable transfer of a fill word to the serial register instead. The OR function of all SL signals enables the parallel loading of the serial output register. Status and sync words are selected directly by SL signals because no decision is required as to their readiness.

Timing of all operations of the interleaver is controlled by a central clock consisting of a programmable 4-phase VCO clock that is referenced to a fixed-crystal oscillator. The VCO is programmed by a selectable PROM to a frequency compatible with the total source data rate and the word counter for the format program is also programed by the frequency select PROM to correspond with the selected format PROM.

If a set of input data is selected, operation of the interleaver and related timing can be shown. Assume that there is data on all input channels whose total rate is 200 kb/s, ±1%.

Assume that the related format will have about 10% overhead, which includes all fills. The VCO clock will be programmed to provide a serial bit clock (SBCK) about 1% faster than the total data rate, or [200+0.01(200) + 0.1(200)] 1.01 = 224 kb/s. With this arrangement all data will be accepted without loss and there will be a minimum of fill words if the input rate stays at about 200 kb/s. If the input data are reduced by data losses the output data stream will have more fill data and the system will continue to function normally at a lower efficiency.

Designate channels A through G and assume the following set of data rates in kb/s: A = 16, B = 16, C = 32, D = 96, E - 64, F = 39.5, and G = 0.5. A minor frame can be set so that one word of A, one word of B, two words of C, six words of D, four words of E, 2.47 words of F, and 0.5 word of G are received during the period. A sync word(s) will occur at the beginning of every frame and fills (x) will occur when data aren't ready and a status word (w) will occur at the end. Minor frames must always be symmetrical in the occurrence of events and subcommutation is permissible. A time relationship for the preceding illustration is shown in Figure 2. The diagram shows input data as synchronous for simplicity and a random start is assumed with no data in the FIFOs. The first frame can be seen used for initialization and there are many fills (x) because data are not ready. After the first frame passes, normal operation occurs, and G-data can be seen not ready for output until the third frame. The F-data, which has no harmonic relationship to the other frequencies, will have periodic fills. After initialization the percentage overhead data, including fill words, are only 5/60 or 8.3%. Studies show that even complex formats from more data sources would average about 90% efficient. The length of words affect achievable efficiency - the longer the word, the higher the efficiency. All IRIG standards allow up to 33 bits but 24 bits are used for the selected design because less hardware is required. If efficiency is a premium, hardware can be increased to provide 32-bit formats. Because computers are most generally configured to 8-bit bytes, a word length that is a multiple of eight is always suggested. After data are serialized, they are converted from NRZ-L to biphase L and outputed through a balanced line driver capable of driving modulator or other loads at about 75 ohms.

Another example shows a situation where subcommutation is used. Assume the following data set of data in kb/s: A = 55, B = 33, C, D, E, each = 17, F = 40, and G = 2. A minor frame can be set so that there are five words of A, three words of B, 1-7/11 words each of C, D, and E, 3-7/11 words of E, and 2/11 words of G. If time slots were set in symmetrical frames with no subcommutation, there would be five words for A, three words for B, four words for E, two words each for C, D, F, and one word for G. Because C, D, E, F, and G wouldn't always be ready there would be a relatively large number of fill words. The total words per minor frame occupied by C, D, E, and G are: $3(17/11) + 2/11 = 53/11 \cong 5$; therefore, subcommutation of these into five time slots will improve the efficiency of the data transmission. If a frame is assumed to have a sync word and a status word, there

would be 21 words in a frame without subcommutation and 19-words with subcommutation. Efficiency improvement would be about 10%. Figure 3 illustrates the timing for subcommutation.

The ground station receiving the data stream looks at the data stream and can demultiplex all data (with the correct software or hardware) into its separate channels, omitting all fill data as identified by the status word. Software modifications to existing IRIG ground stations to handle the composite data streams are believed to be simple, although minor hardware changes to discard fill data may be even simpler and cheaper.

In the upper right corner of Figure 1 there is an optional circuit referred to as a check bit generator. The inclusion of this circuit depends on final design criteria. The purpose of the circuit would be to provide a BCH sequence to detect and correct errors in the status word when received at the ground station. If long missions are projected (i.e., longer than 24 hours and/or noise levels in the transmission link will create errors at a rate of more than about 1 in 10^7 bits), an error correction would still ensure a high probability of correct status as to whether data are valid or fill information. However, there is a possibility that data signatures and time are adequate to resolve ambiguities without error correction even in the presence of high. A study of all conditions, including the complexity of implementing software in IRIG stations for BCH error correction, is recommended during the design phase. A 31, 16 BCH code was implemented for this purpose in the High Rate Multiplexer already mentioned.

Conclusions. It is feasible and practical to implement a digital data interleaver to combine several asynchronous digital data sources into one serial data stream. These data may be formatted in symmetrical IRIG-compatible sequences without the loss of data. There is no requirement for the incoming signals to be harmonically related, nor is it necessary to impose any frequency constraints within the capacity of the system. One of the best ways to keep the formats symmetrical when data are to be acquired but are not ready, is to use fill words. Potential ambiguities between fill words and data are resolved by the use of optimum codes for fill words or a status word that identifies the validity of each word. If extremely small probabilities of ambiguities between fill and data words are required in a noisy system, error correcting techniques may be applied to coded fill or status words.

Bibliography

1. Berlekamp, Elwyn R.: Algebraic Coding Theory, McGraw Hill, 1968.

2. Massey, James L.: Step-by-Step Decoding of the Bose-Chandhuri, Hooquenghem Codes, IEEE Transactions on Information Theory. Vol II-11, No. 4, Oct. 1965.

3. Williard, M. W.: Refinements on Analysis of PCM Synchronization, Proceedings of the 1966 National Telemetry conference.

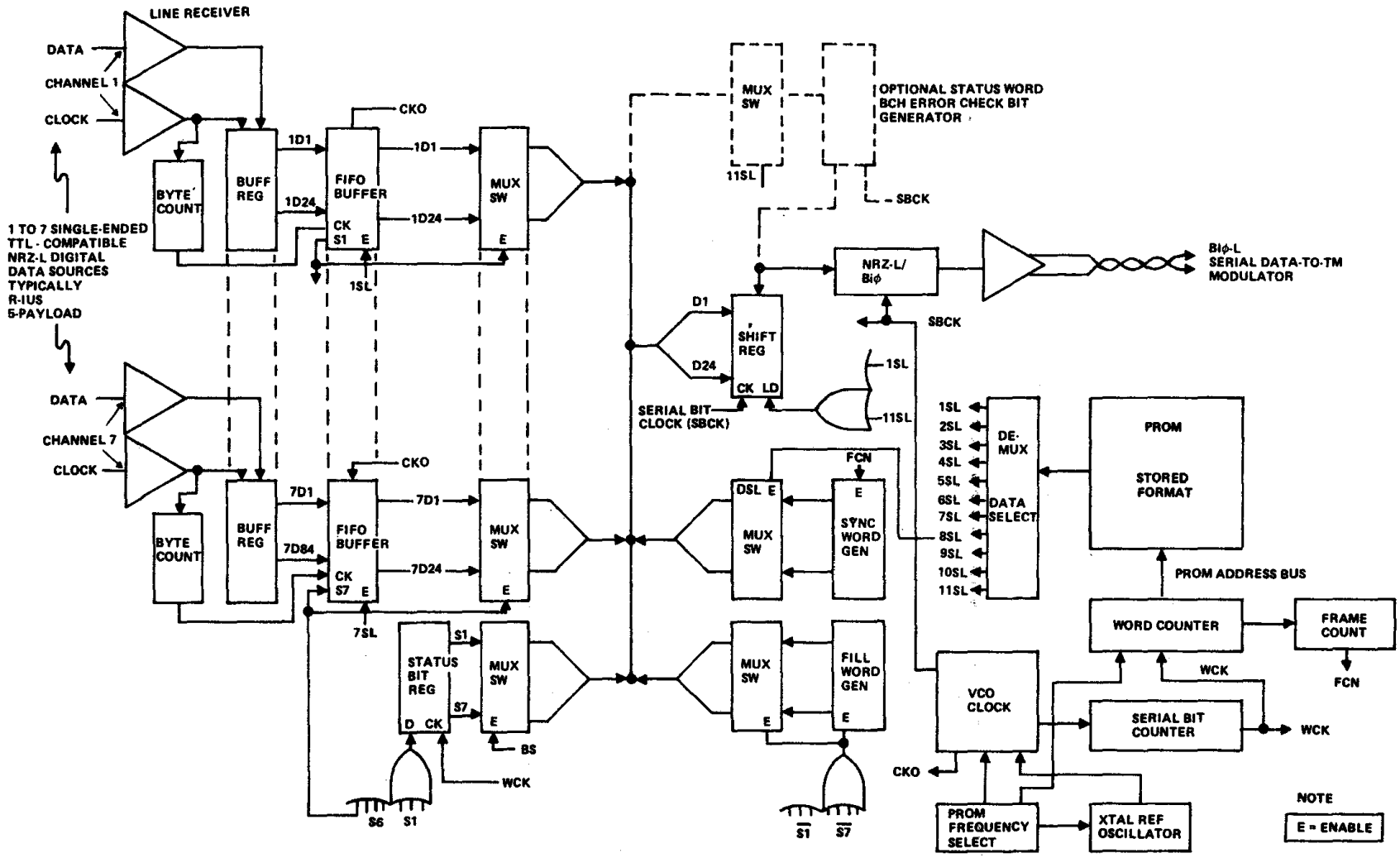

Figure 1 Data Interleaver Block Diagram

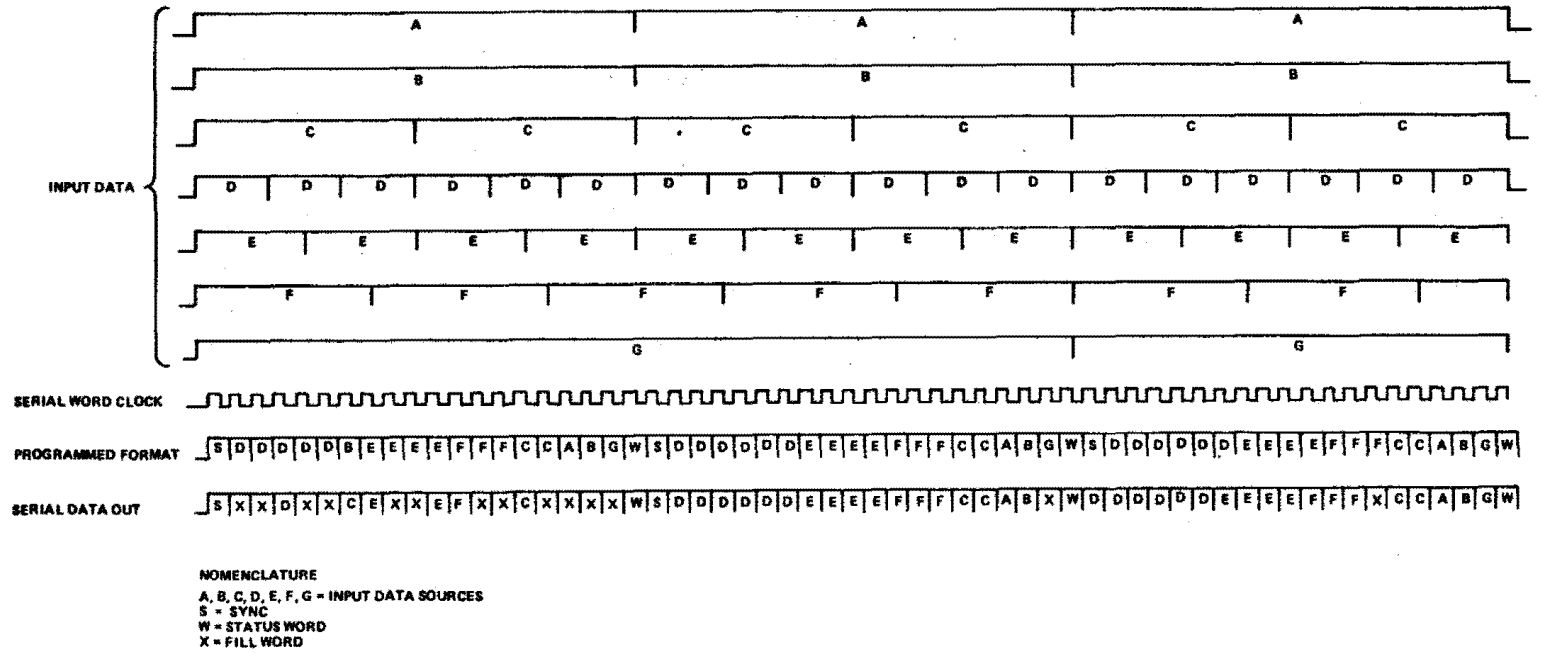

Figure 2 Typical Data Interleaver Word Timing Diagram

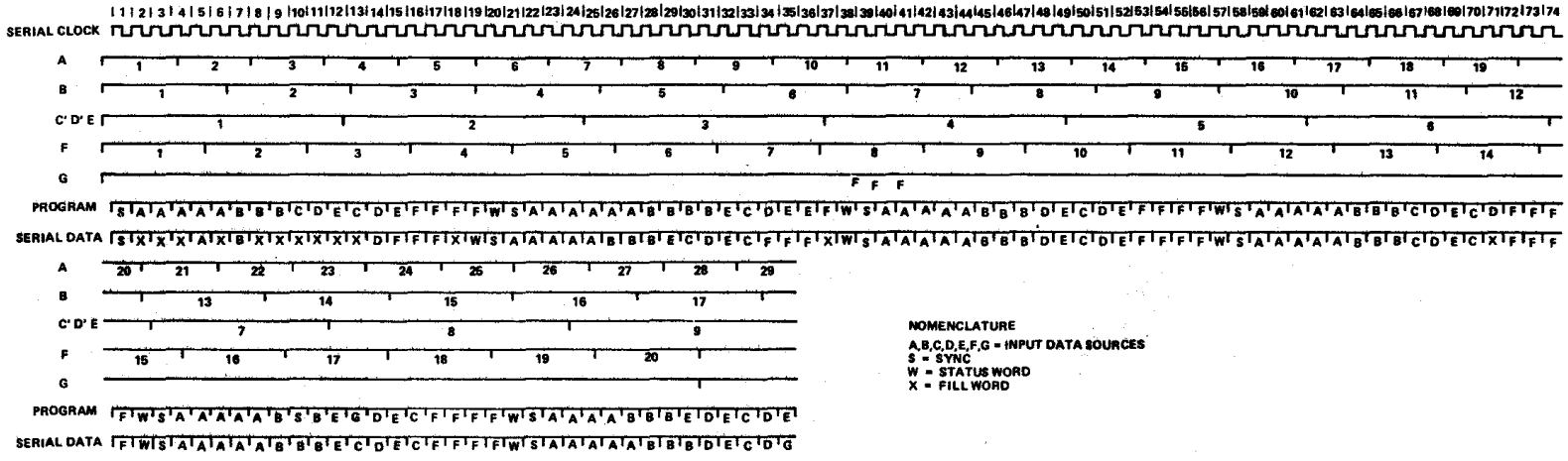

Figure 3. Interleaver Format with Subcommutation