THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 58, No. 9, November 1979
Printed in U.S.A.

Sequencer Designs for Scanning-Beam Satellites

By W. L. ARANGUREN, R. E. LANGSETH, and C. B. WOODWORTH

(Manuscript received April 24, 1979)

We present conceptual designs for the controlling sequencer that would be required on board a communication satellite employing a scanning phased-array antenna. Two basic forms are discussed: one providing separate pointing control for each time slot in the TDMA *frame and the other providing a compact storage arrangement when the typical beam dwell time is several time slots. The update interface to the telemetry link is discussed briefly. The impending availability of radiation-resistant* CMOS *memories makes the control part of a large (100-element) scanning array feasible in the 1980 time frame, with an estimated power consumption under 50 watts.*

I. INTRODUCTION

Recently D. O. Reudink and Y. S. Yeh[1] proposed a scanning spot-beam antenna to provide high-gain, area coverage in a satellite communication system. The most efficient method of implementing the scanning function appears to be by way of a phased array together with electronically-controlled phase shifters (separate sets for transmit and receive). These phase shifters are scanned in a cyclic fashion to provide the necessary interconnection between ground stations.

If the scanning were sufficiently slow, it would be possible to transmit the ongoing phase shifter information to the satellite from a control earth station. However, the downlink beam may shift position every microsecond or so.[1] With a 100-element array and 3-bit phase shifters per element, about 300 megabits per second would be required for real-time phase control. Obviously, then, the satellite must store the necessary information to control the basic scanning with only *changes* transmitted by the control station. In this paper we describe efficient methods for organizing this storage on board the satellite, with consideration given to power consumption by some representative memory

devices currently available. Additionally, we indicate how selective phaser stepping (or dithering), which may be required for initial beam forming, can be accomplished.

II. BASIC CONTROLLER

Figure 1 shows the basic N element phased-array beam former and phase controller; separate shifters and control memories will be necessary for transmit and receive functions. As described in Ref. 1, there might be M ($\approx$100) distinct beams, each requiring a distinct set of N phases, θ_n.

Let us assume that each phase, θ_n, must be stored using b bits, to provide for acceptable sidelobe degradation as compared to purely analog phases; three bits may be adequate for this purpose.[1] Thus, $N \cdot M \cdot b$ bits of onboard storage must be provided to define the basic beam-number-to-phase mapping. For reasons of reliability, this storage module should be arranged as N subunits of Mb bits each, where each subunit is used to control the phase shifters of one element.

The control function is thus reduced to selecting, in some sequence, the Nb bits which form a given beam, maintaining these bits on the phase shifters for the desired duration of the given up- or downlink beam, moving onto the next beam by selecting a new set of Nb bits, and so on. Thus, a basic controller has the general form sketched in Fig. 2, wherein a sequencer, to be described shortly, controls $\log_2 M$ binary address lines to permit selection among the M possible beams. Note that because uplinks and downlinks scan separately over the M

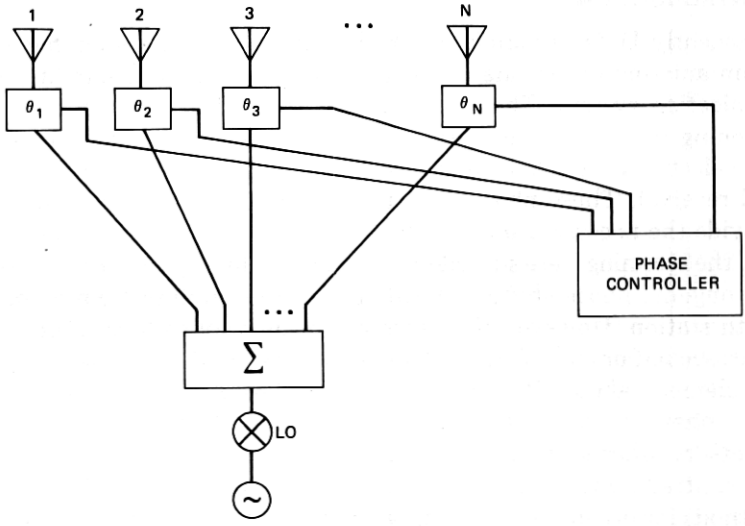

Fig. 1—A phased array with digital phase shifters.

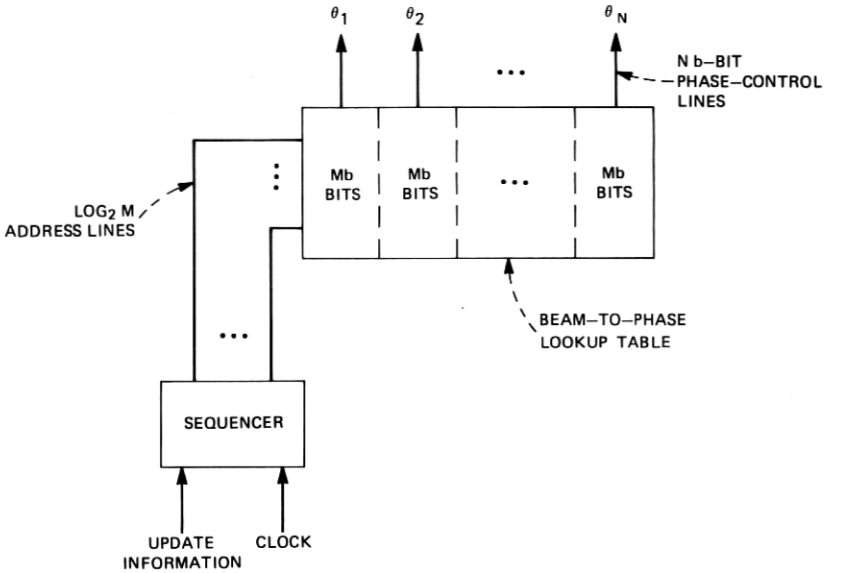

Fig. 2—Generic phase controller.

beams, two such controllers will be on the satellite, each with its own beam-to-phase lookup memory and sequencer. The update input provides for reconfiguration, which may provide for demand assignment, if desired.

The heart of the controller is thus the sequencer which contains the information to control the actual duration of each beam and the sequence with which they occur (the up- and downlink sequences being distinct). By updating the sequencers, variations in traffic patterns can be accommodated. In the next section, three basic forms of the sequencer are presented. The first form is designed to permit fine-grain timing control, with a resolution of one basic time slot. The second form is convenient when each beam position is maintained for several time slots, which permits a compact storage arrangement for the sequence of durations. This will probably be primarily true of the uplink, which remains fixed while the downlink is distributing data among as many as $M - 1$ other beams. The complexity of the two methods are similar for downlink control. The third design provides improved demand-assignment flexibility with limited memory size.

III. SEQUENCERS

The first form of the sequencer provides a from-to assignment for each individual satellite circuit. That is, both the uplink and downlink are capable of being reconfigured on a per-time-slot basis. This allows

the system to be used in a fixed assignment, a demand assignment, or a mixed mode of operation.

Figure 3 presents a block diagram of this sequencer. A single $\div K$ counter drives a look-up table which provides the beam number for each time slot. Thus, each step of the counter defines a single time slot. For an M-beam system, the memory requirement for each table would be a block of K words of $\text{Log}_2 M$ bits each. For example, seven memory ICs of 16,384 × 1 bits each would be capable of distributing 16,384 circuits among 128 beams.

This configuration of using one memory IC per bit of output word lends itself to block error-correcting codes. That is, we are assuming the failure mode would involve a single bit at a time (not entire words), which would be the case of a single memory IC failure. Four additional bits, and thus four additional memories, would be required to use a Hamming single error-correcting code, for example.

In the second form of sequencer implementation, we observe that the sequence of beams is uniquely determined by (*i*) how long to stay with the current beam and (*ii*) which beam is next. In the downlink case, these numbers are also a function of which beam is currently active on the uplink. Consider first the uplink sequencer. With M beams, there can be $M - 1$ "next beams," each of which can be coded by $\log_2 (M - 1) \approx \log_2 M$ bits. Thus, a memory of M words, each of $\log_2 M$ bits suffices to define the sequence of "next beams." For a satellite system with up to $M = 128$ beams, a 128 × 7 memory suffices. This memory can be implemented as one integrated circuit 128 × 8 RAM (random access memory). This same memory organization can be used for the duration of each beam, providing for a 128:1 range of duration.

This sequencer is sketched in Fig. 4. Initially, we may latch beam "zero" into the next beam latch, and its duration into the down

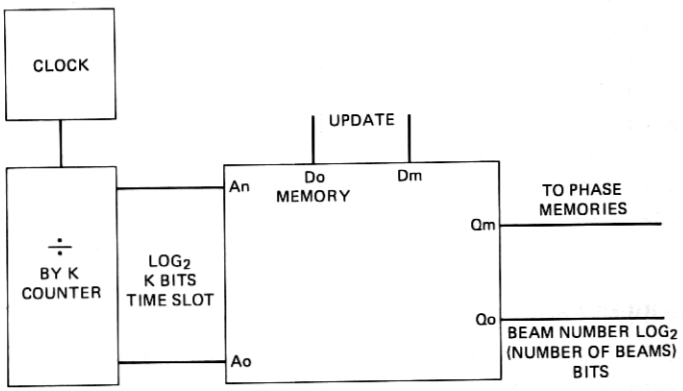

Fig. 3—A sequencer that maps individual time slots to desired beam number.

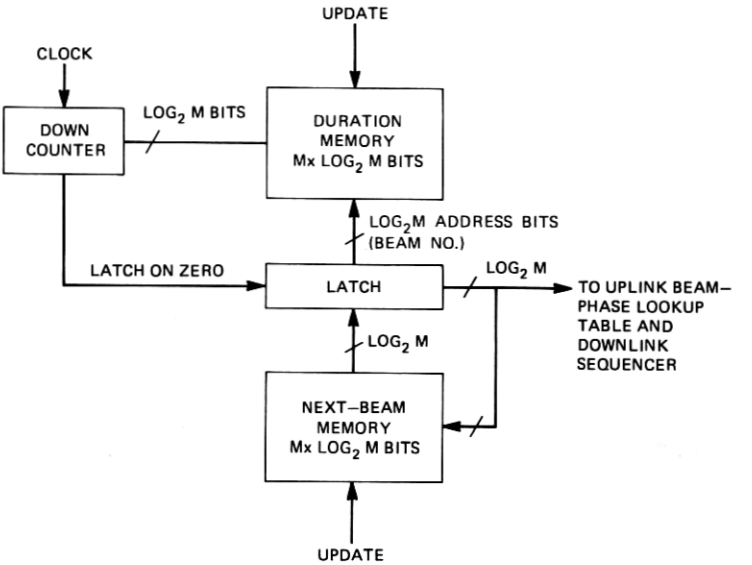

Fig. 4—Uplink sequencer.

counter. The contents of the next beam latch are used to address the beam-phase lookup table. As the counter decreases, the same beam is maintained until the counter reaches zero. At this time, the binary-coded address of the next beam number is latched, the new duration is loaded into the counter, and the process repeats.

A similar sequencer may be used to form the downlink beams, but it will have to be M times larger, since the sequence of downlink "next-beams" and durations will probably depend on which uplink beam is active. Thus, instead of $\log_2 M$ address lines for these two memories, $2 \log_2 M$ lines are required, where $\log_2 M$ of them came from the uplink sequencer. This configuration is sketched in Fig. 5. Note that the output of this sequencer is still only $\log_2 M$ bits to address the M sets of Nb bits defining the M downlink beams. With $M = 128$, each memory of Fig. 5 would be $16{,}384 \times 7$ bits, which can be implemented with seven chips. In practice, one would probably use the method of Fig. 3 for the downlink, since it could be implemented with a single set of seven 16K × 1 memories.

In a demand assignment situation, it may be desirable to be able to return to a beam number several times within a superframe. For example, there may be a fixed block of time slots assigned at the beginning of the superframe followed by a pool for demand assignment. As a result, a ground station may be bursting up several times within a superframe.

With the previous implementation of the sequencer, it is impossible

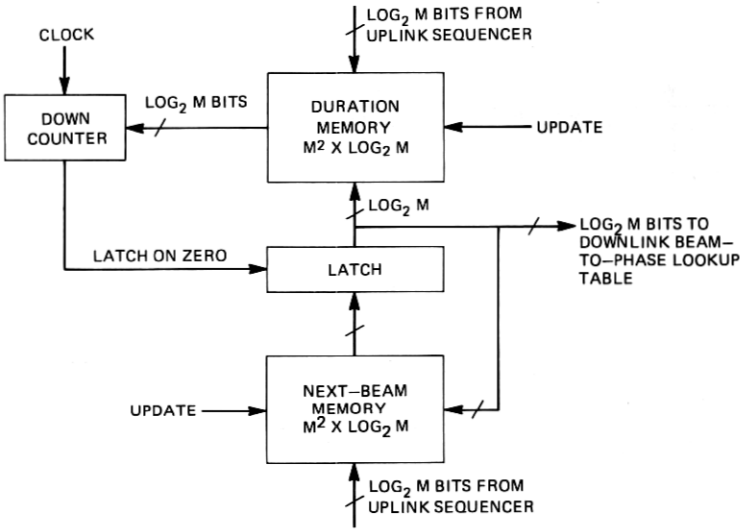

Fig. 5—Downlink sequencer.

to return to any beam number within a superframe without becoming caught in a loop. Figure 6 represents an impossible sequence. When beam number 6 is latched the second time, the next beam number should be 2 to continue in the sequence. This is impossible, since the next beam memory is addressed by the current beam number which is 6, and this location already has a next beam of 1. Thus, the remaining sequence would be beam 1 and beam 6 repeated until the end of the superframe. Also, with the previous proposal, the number of bits in the data word of the duration memory sets the maximum number of time slots per beam per superframe, since any beam number can only be entered once in the next beam memory.

A sequencer that can have repeated beam numbers in the beam memory, thus allowing demand assignment and unlimited time for each beam, is shown in Fig. 7. Instead of addressing the next beam memory and duration memories by the current beam number, a sequence number addresses a current beam memory and a duration memory. At the beginning of a superframe, the sequence counter is reset. This addresses the first beam number which is latched, and the duration number corresponding to that beam number is loaded into a counter which acts as a timer. On timeout, the sequence number is incremented and the next beam number, which may be the same as the first, is latched for a new duration. Figure 8 displays a typical sequence in operation. Here 50 units represent the maximum duration; beams requiring more time can be repeated, as in the case of beams 0 and 6 in the figure.

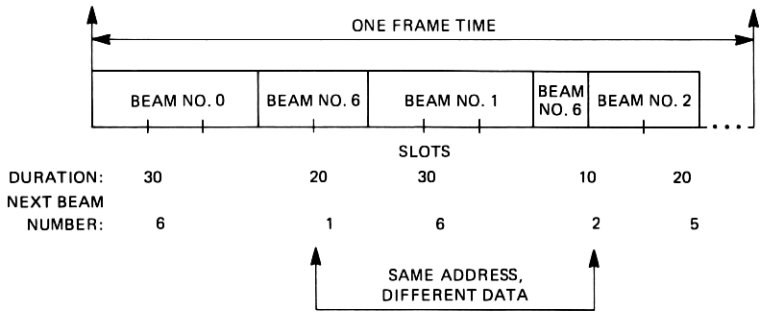

Fig. 6—Impossible frame picture from sequencer in Fig. 4. Next beam number and duration for beam 6 are not unique.

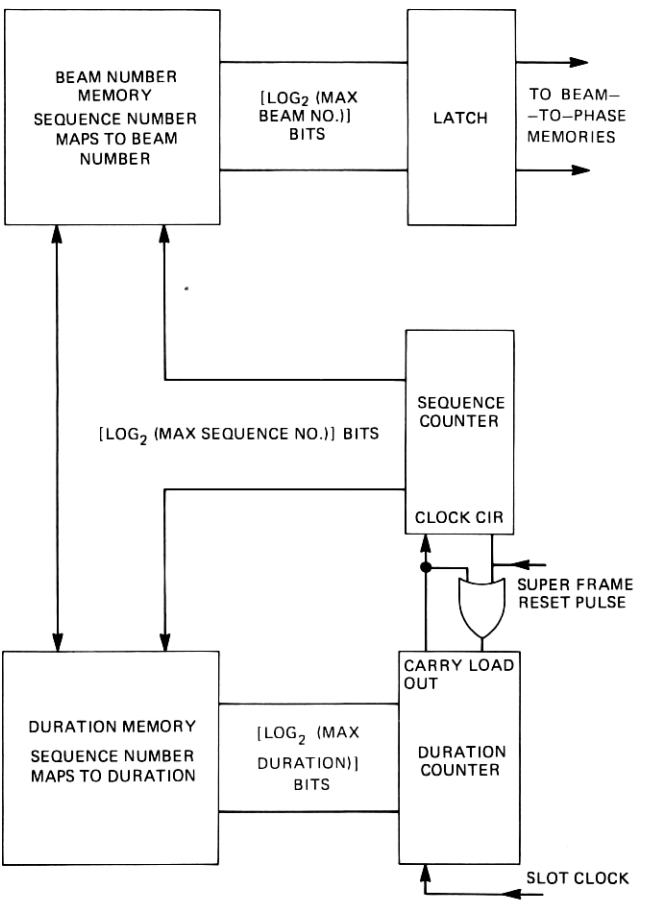

Fig. 7—New uplink sequencer which allows repeated beam numbers.

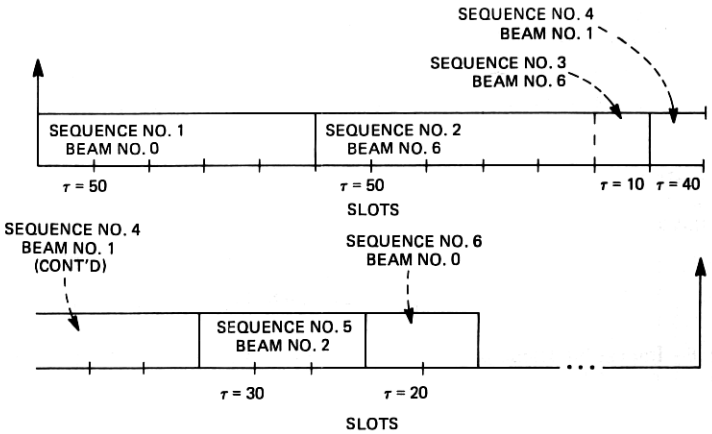

Fig. 8—A typical frame picture of the new uplink sequencer in Fig. 7.

The bus size of the beam memory is $\log_2$ of the maximum sequence number by $\log_2$ of the maximum number of beams. The duration memory must be $\log_2$ of the maximum sequence number by $\log_2$ of the maximum duration per sequence number.

Just as the "uplink" sequencer of Fig. 4 was generalized to the downlink sequencer of Fig. 5, so the sequencer of Fig. 7 can be generalized to a downlink counterpart, as shown in Fig. 9. Here the contents of the uplink sequence counter are used to form a portion of the address for the downlink duration memory and sequence-to-beam number memory. A separate duration counter is incremented by the basic slot clock; its carry-out is used to increment the downlink sequence counter. Each time the uplink sequence counter is incremented by the carry from the uplink duration counter, the downlink sequence counter is reset to begin counting the downlink sequence for the new uplink beam number, using a new set of duration and beam numbers.

Given several possible sequencer designs, it is natural to ask when it is reasonable to use each design. The answer basically depends on two things: the number of slots per TDMA frame, and the number of beams in the satellite system. Consider a TDMA frame length of 25 ms, such as is discussed in Ref. 2 and 3. A basic slot may be around 2.5 μs in length, permitting capacity to be assigned in units of from one voice circuit (a slot repeated once per frame) on up. Thus, there are 10,000 slots per frame. Let us consider two exemplary systems, one with 100 distinct beam positions and one with 20.

In the former case, there would be, on average, 100 slots assigned to each uplink beam with one slot assigned per downlink for each uplink. In this case, the compact memory designs of Fig. 4 or Fig. 7 would be

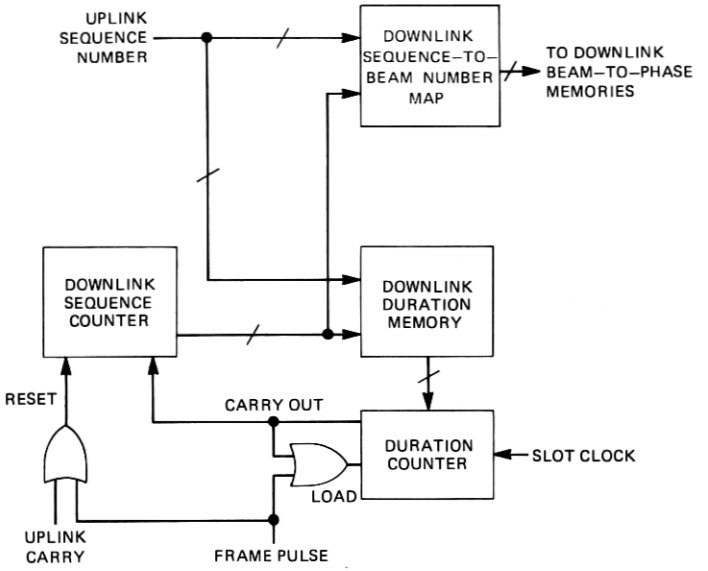

Fig. 9—Downlink sequencer allowing repeated beam numbers.

used for the uplink sequencer using a 7-bit sequence counter while the expanded design of Fig. 3 could be used on the downlink using a 14-bit slot counter. In the latter situation of a system using 20-beam positions, around 500 slots would be assigned to each uplink beam, with about 25 slots assigned to each downlink beam for each uplink beam. In this case, both up- and downlink sequencers could be of the compact designs, either Fig. 4 or Fig. 7 for the uplink, and Fig. 5 or Fig. 9 on the downlink. Here we would need, perhaps, a 5-bit sequence counter together with a 9-bit uplink duration memory, along with a 5-bit downlink sequence counter and a 5-bit downlink duration memory and counter in Fig. 9.

No matter what sequencer design is chosen, it must permit modification of the memory contents to permit changes in the scanning sequence to accommodate changes in traffic patterns; these changes may be seasonal, daily, hourly, or perhaps more frequent. In any case, the sequencers must be able to receive update information without interrupting the sequence in progress. This involves writing update information into memories that may be accessed each time slot. One way of accomplishing this would be to provide two copies of each memory, one of which is active at any time. Update information would be written into the second copy, with the second taking over upon command, probably at the beginning of some frame.

Alternatively, update can be accomplished by either using a read-modify-write cycle or by separate read-and-write cycles for each time

slot. A read-modify-write cycle has the advantage of being faster, thus allowing smaller slot times, but it requires that the update address match the current sequencer or phase memory address. State-of-the-art memories are sufficiently fast for separate read-and-write cycles per slot at slot times presently proposed. Dual access allows any location to be updated anytime and has simpler timing requirements.

Since the beam-to-phase memories also must be updated without interruption, the same technique can also be used to update both the sequencers and the beam-to-phase memories. Figure 10 shows an interlaced update scheme using the dual access technique. While the sequencer memories are doing a beam lookup, the beam-to-phase memories are in an update memory cycle. Once the lookup is complete, the beam number is now available to access the beam-to-phase memories. At this time, the sequencer can receive update information. Using this method, the sequencer could be completely updated in a single frame if necessary. With this flexibility, demand assignment

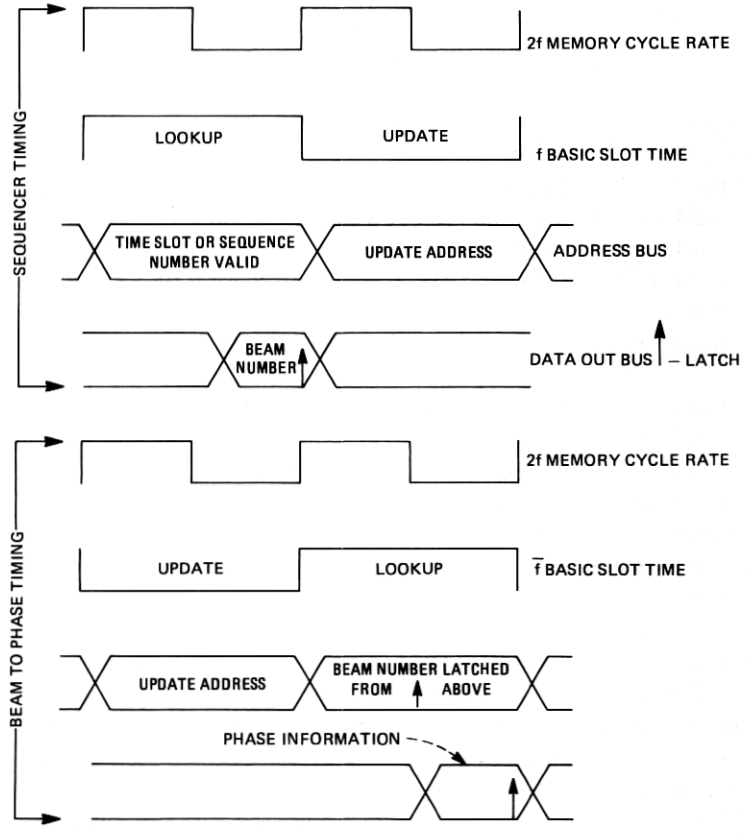

Fig. 10—Interlaced update timing diagram.

could be done on a call-by-call basis if desired, provided the uplink telemetry channel had adequate capacity.

IV. POWER REQUIREMENTS

Power on board a satellite is a precious commodity. We are thus concerned about the total power required by the scanning controller. From previous discussion, it is apparent that the beam-to-phase lookup table will probably be the largest power consumer, since it could be constituted of up to 100 separate memories. Typical devices that are useful here are 256 × 4 bit CMOS RAMs. These devices have access times under 0.5 μs, which is adequate, since a typical slot-time in the satellite system's transmission frame would be around 1 to 2 μs. Typical power requirements for these devices appears to be in the 50- to 100-mW range. Thus, the basic lookup table for 100 elements should require between 5 and 10 W, not including a few miscellaneous MSI and SSI circuits, whose power consumption would amount to another watt or so.

The second major component is the sequencer, for which several designs have been discussed. Consider Fig. 3 first. The slot to beam-number mapping sequencer might be implemented with seven 16K × 1 RAMs, providing for up to 128 beams. The actual satellite transponder capacity may be around 10,000 slots (voice circuits), assuming 64 kb/s per voice circuit and a transponder bit rate of around 600 Mb/s. Available 16K × 1 static RAMs have maximum power dissipations of around 500 mW. Considering that such memories require some additional SSI circuitry for address decoding, etc., we may allocate around 5 W for this form of sequencer, which would most likely be used in the downlink. The design of Fig. 9 would be similar.

Moving on to Fig. 4 or Fig. 7, the major components here are the duration and next beam memories. For up to 128 beams, and 256:1 range of uplink beam durations, each of these could be implemented as single 128 × 8 MOS static RAM, with a maximum power requirement of around 600 mW. Adding the usual SSI peripheral chips suggests that this sequencer would consume between 1.5 and 2 W.

We may, thus, construct the following estimated power budget for both up- and down-link phase controllers:

Beam-to-phase Lookup (2)	20 watts
Uplink sequencer (Fig. 4 or Fig. 7)	2
Downlink sequencer (Fig. 3)	5
	27 watts

Reliability considerations suggest that redundancy must be added to the sequencer control memories. Although this has not been inves-

tigated in detail, we know, for example, that by adding eight additional 16K × 1 RAMs to the seven already proposed for the downlink sequencer, the sequencer can be made immune to the failure of any two of the memories.[4] (This assumes, of course, that a highly reliable error-correcting decoder converts the 15 redundant bits to the desired seven control bits.) Thus, we may estimate that a phase controller with redundancy will require more like 30 to 35 W of dc power. This amount of power would appear to place no major additional requirement on the overall satellite power generation.

Since we are placing rather heavy reliance on CMOS memories, care must be taken in their design to ensure survivability in the radiation environment of geostationary orbit. At the present time, CMOS radiation tolerance is insufficient for a 7-year on-orbit lifetime. However, considerable effort is under way in the aerospace industry to improve this situation,[5] so that it is highly likely that adequate CMOS memories will be available in a short time.

V. PHASE DITHER

It may be necessary to provide some closed-loop means by which it can be ascertained that a given beam in fact points where it is supposed to. Reasons for this concern include the slow, uncorrected satellite motion that occurs between station-keeping maneuvers, which may cause small but significant pointing error, particularly for beamwidths under one degree. To track this motion, the basic beam-to-phase mapping will require occasional updating. As suggested by Y. S. Yeh, the update information can be obtained by sequentially tagging the array elements with a phase dither, by which means some (or perhaps all) the earth stations can measure the relative element phases. By transmitting the information to a control station, corrective updating of the satellite downlink beam-phase lookup can be effected as necessary. (A similar updating of the uplink lookup table can be done with measurements made on the satellite.)

The dither approach basically requires that, during a preassigned subframe, the phase shifter of one of the array elements be periodically stepped, providing a stepwise linear phase progression, essentially equivalent to a frequency offset of that element. At the same time, a fixed reference element is stepped in the opposite direction. By proper processing at an earth station, the relative element phase can be estimated.

A simple method of adding this capability to the phase controller of Fig. 1 is to follow each of the N array-element phase memories with a so-called programmable up/down counter, as shown in Fig. 11 for a typical element. During normal operation, the counting is inhibited, and the normal data from the memory are "programmed" through the

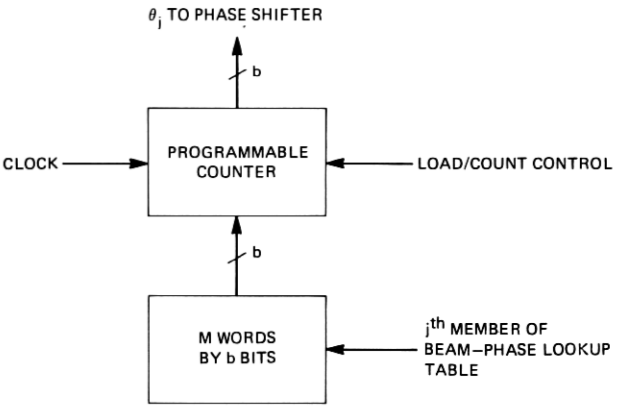

Fig. 11—How to provide phase dither.

counter and simply appear on the output for normal operation. By selectively enabling the counter mode, the phase on a given element will be stepped by the clock, implementing the desired tagging.

VI. CONCLUSIONS

We have discussed several designs for the controller sequencer that would be required on board a communication satellite employing a scanning spot beam. By making use of LSI memory technology, compact and reasonably low power controllers are feasible. Two different memory organizations were described, one providing for a different beam position in each time slot of the TDMA frame, while the other provides a more compact memory arrangement for those situations wherein typical beam dwell times are many slots. For large arrays (about 100 elements), it is necessary to use a low-power technology such as CMOS to store the individual element phases; it appears that a radiation-resistant version of this technology should be available by the late 1980s.

REFERENCES

1. D. O. Reudink and Y. S. Yeh, "A Rapid Scan Spot-Beam Satellite System," B.S.T.J., *56*, No. 8 (October 1977), pp. 1549–60.
2. D. O. Reudink and Y. S. Yeh, "The Organization and Synchronization of a Switched Spot-Beam System," Conf. Record of 4th Internat'l Conf. on Dig. Sat. Comm., Montreal, October 1978, pp. 191–6.
3. A. S. Acampora and R. E. Langseth, "Baseband Processing in a High-speed Burst Modem for a Satellite Switched TDMA System," Conf. Record of 4th Internat'l Conf. on Dig. Sat. Comm., Montreal, October 1978, pp. 131–8.
4. S. Lin, *An Introduction to Error-Correcting Codes*, Englewood Cliffs, N.J.: Prentice-Hall, 1970, Section 6.1.
5. P. J. Klass, "New Microcircuits Resist Radiation," Aviation Week and Space Technology, October 10, 1977, pp. 58–61.