TECHNICAL NOTE

A DIGITAL SEQUENTIAL GENERATOR FOR IMPLEMENTING REINFORCEMENT SCHEDULES

Traditionally, reinforcement schedules have been implemented using a motor-driven punched tape programmer. These devices are often inconvenient to use, especially when interfaced with the digital logic circuits used in behavioral experimentation. A digital generator has been developed that replaces the conventional tape drive while improving accuracy and versatility.

A block diagram of the digital sequential generator is shown in Figure 1. The blocks constructed from solid lines represent the functional organization of the device while the blocks constructed from dotted lines represent an actual integrated circuit used to implement that function in a practical circuit. The memory's output, in the form of a digital word, is connected directly

to the digital comparator. A particular word that has been programmed into the memory is selected by the contents of the address register. Most operations are performed using only the start and clock inputs and one output, timeout enable (TOE). When a pulse is applied to the start input, TOE changes state, indicating that the digital sequential generator is counting. The clock register counts each input clock pulse and feeds its output to the digital comparator. When the count in the clock register equals the output of the memory, the digital comparator produces a timeout signal, which is then sent to the control logic. The timeout condition changes the state of TOE, resets the clock register, and sequences the address register. These changes have the

TOE - TIME OUT ENABLE

$\overline{AD}$ - INHIBIT SEQUENCE OF ADDRESS REGISTER

* FOR FR AND VR SCHEDULES REMOVE JUMPER AND APPLY INPUT HERE.

Fig. 1. Block diagram of digital sequential generator.

VISDESIGN
DESIGN CALCULATION OF A SOLID STATE VARIABLE INTERVAL SCHEDULE TIMER

INPUT CONSTANTS
COEFFICIENT CONVERSION FACTOR 60
INPUT PULSE PERIOD (SEC) 0.948

PROGRESSION TABLE

COEFFICIENT	DESIRED INTERVAL (SEC)	COUNTS	ACTUAL INTERVAL (SEC)	PERCENT ERROR
0.03190	1.91400	2	1.89600	0.940
0.09868	5.92080	6	5.68800	3.932
0.17010	10.20600	11	10.42800	2.175
0.24716	14.82960	16	15.16800	2.282
0.33057	19.83420	21	19.90800	0.372
0.42159	25.29540	27	25.59600	1.188
0.52167	31.30020	33	31.28400	0.052
0.63313	37.98780	40	37.92000	0.178
0.75844	45.50640	48	45.50400	0.005
0.90178	54.10680	57	54.03600	0.131
1.06923	64.15380	68	64.46400	0.484
1.27055	76.23300	80	75.84000	0.516
1.52326	91.39560	96	91.00800	0.424
1.86306	111.78360	118	111.86400	0.072
2.38629	143.17740	151	143.14800	0.021
3.77259	226.35540	239	226.57200	0.096

DESIRED INTERVAL SEQUENCE MEAN (SEC) 60
ACTUAL INTERVAL SEQUENCE MEAN (SEC) 60.02025

AVERAGE ERROR 0.804
AVERAGE ERROR OF THE FIVE LARGEST ERRORS 2.104

LIST OF THE FIVE LARGEST ERRORS

ELEMENT	ERROR
2	3.931901094
4	2.281922641
3	2.175191064
6	1.188358358
1	0.9404388715

MEMORY PROGRAM

RANDOM SEQUENCE	COUNT	MEMORY LOCATION	MEMORY VALUE
1 16	239	0 0 0 0	1 1 1 0 1 1 1 1
2 4	16	0 0 0 1	0 0 0 1 0 0 0 0
3 10	57	0 0 1 0	0 0 1 1 1 0 0 1
4 5	21	0 0 1 1	0 0 0 1 0 1 0 1
5 3	11	0 1 0 0	0 0 0 0 1 0 1 1
6 1	2	0 1 0 1	0 0 0 0 0 0 0 1
7 7	33	0 1 1 0	0 0 1 0 0 0 0 1
8 14	118	0 1 1 1	0 1 1 1 0 1 1 0
9 11	68	1 0 0 0	0 1 0 0 0 1 0 0
10 2	6	1 0 0 1	0 0 0 0 0 0 1 1
11 9	48	1 0 1 0	0 0 1 1 0 0 0 0
12 6	27	1 0 1 1	0 0 0 1 1 0 1 1
13 15	151	1 1 0 0	1 0 0 1 0 1 1 1
14 13	96	1 1 0 1	0 1 1 0 0 0 0 0
15 8	40	1 1 1 0	0 0 1 0 1 0 0 0
16 12	80	1 1 1 1	0 1 0 1 0 0 0 0

Fig. 2. Computer printout for a progression in which the probability of reinforcement as a function of time is constant.

Fig. 3. Schematic diagram of digital sequential generator.

effect of initializing the device for the next start pulse and selecting the next count in the sequence that has been programmed in the memory.

Many possible operating modes exist to implement several different types of reinforcement schedules. A variable-interval (VI) schedule can be achieved by programming the memory with a desired sequence, apply-

ing the appropriate clock frequency and interfacing the experiment to the device. A fixed-interval (FI) reinforcement schedule can be achieved by inhibiting the sequencing of the address register. A fixed-ratio (FR) schedule can also be implemented with this device by inhibiting the address register as previously explained and by connecting a signal representing a correct re-

sponse directly to the clock register. Since this allows the correct responses to be counted by the clock register, the device will stop as indicated by a change in state of timeout enable, after the number of correct responses received equals the number programmed in memory. By allowing the address register to sequence, the device will implement a variable-ratio (VR) reinforcement schedule. Because the read-only memory contains sixty-four possible reinforcement intervals, several different schedules can be stored simultaneously, making it possible to implement chained reinforcement schedules.

There are two errors inherent in the operation of this device, both of which are due to its digital design. The first is synchronization error that results because the system clock is not synchronized to the start pulse. By dividing the clock by 10 before applying it to the clock register, this error will produce an uncertainty of only 0.1 clock register pulses. The second type of error, digitizing error, results when the time interval desired cannot be represented by an integer number of clock pulses. A computer program written in APL, an interactive computer language, was used to optimize the choice of clock period to minimize digitizing error.

A sample printout using a progression in which the probability of reinforcement as a function of time is constant is shown in Figure 2. The program accepts terms of a progression that are normalized to produce a progression mean of one and multiplies each term by the desired progression mean to calculate the desired interval in seconds. Then, the program calculates the integer number of input pulses needed to best approximate a particular interval. The actual interval produced by the digital sequential generator is the product of input pulse period and the integer number of pulses used to produce that interval. The digitizing error is computed by dividing the absolute value of the difference between the desired interval and the actual interval by the desired interval. All printout values needed to calculate digitizing error are listed on the printout in the progression table. Statistics compiled from the digitizing error are tabulated by the program and are used to optimize the choice of clock pulse period. The criteria of optimization for the sample printout was to minimize the mean of the digitizing errors while ensuring the digitizing error for a particular interval did not exceed 4.0%. The program was run several times using different input pulse periods. Figure 2 represents the best solution that resulted in a maximum digitizing error of 3.9% and a mean of digitizing errors for all 16 intervals of 0.8%.

The APL printout includes a table that is used as an aid in programming the digital read-only memory. The integer count of input pulses is randomized and expressed as a binary number for each memory location. The constant probability VI reinforcement schedule was 16 intervals, shown in Figure 2, was pro-

grammed with the digital memory and was used in a successive discrimination task with rhesus monkeys as subjects. Experience has shown that the digital sequential generator is far more accurate than its tape drive counterpart.

A schematic diagram of the digital sequential generator is shown in Figure 3. The integrated circuit designation numbers and schematic diagram symbol table are listed in Table 1. All of these integrated circuits are standard products available at most electronic distributors, except for Q3, the programmable read-only memory. These devices have been introduced only recently, although many types are now commercially available. A monolithic bipolar memory, which was used to implement the digital sequential generator, is available from Motorola Semiconductors. It was programmed by applying appropriate voltages to burn away internal metallic fuses. This operation takes about 15 min for a particular schedule, but since the process is destructive, the memory cannot be reprogrammed.

Table 1
Integrated Circuit List

Q1	SN 74L93	Binary Counter
Q2	SN 7485	Comparator
Q3	MCM 5004AL	Prom (Motorola)
Q4	SN 74L93	Binary Counter
Q5	SN 7485	Comparator
Q6	SN 7490	Decade Counter
Q7	SN 7472	Flip Flop
Q8	SN 7493	Binary Counter
Q9	SN 7400	Nand Gate

Schematic Diagram	Symbol Table
Input/Output	Terminals
CL	Clock Pulse
ST	Start Pulse
AD	Address Register
TOE	Time Out Enable
MA	Memory Address

A new erasable and electrically reprogrammable read-only memory, now available from Intel Corporation, can be erased with an ultraviolet light for later programming, although its cost is much higher than the monolithic bipolar memory. The entire digital sequential generator can be built on a standard 4 by 5 in. (10 by 12.5 cm) printed circuit card for less than one hundred dollars when using the monolithic bipolar memory.

C. L. CHRISTMAN
U.S. Public Health Service
Rockville, Maryland