
THE UNIVERSITY OF ARIZONA
University Libraries
UA CAMPUS
REPOSITORY

Development of a digital pseudorandom noise generator

Item type	text; Thesis-Reproduction (electronic)
Authors	Stemple, Eugene Powers, 1939-
Publisher	The University of Arizona.
Rights	Copyright © is held by the author. Digital access to this material is made possible by the University Libraries, University of Arizona. Further transmission, reproduction or presentation (such as public display or performance) of protected items is prohibited except with permission of the author.
Downloaded	15-Sep-2016 22:45:08
Link to item	http://hdl.handle.net/10150/554733

DEVELOPMENT OF A DIGITAL PSEUDORANDOM NOISE GENERATOR

by

Eugene Powers Stemple

A Thesis Submitted to the Faculty of the

DEPARTMENT OF COMPUTER SCIENCE

In Partial Fulfillment of the Requirements
For the Degree of

MASTER OF SCIENCE

In the Graduate College

THE UNIVERSITY OF ARIZONA

1 9 7 4

STATEMENT BY AUTHOR

This thesis has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: Eugene P. Stemple

APPROVAL BY THESIS DIRECTOR

This thesis has been approved on the date shown below:

G. A. KORN — 8/24/77
Date

Professor of Electrical Engineering

ACKNOWLEDGMENTS

The writer is grateful to the Department of Electrical Engineering, Dr. Roy H. Mattson, Head, for providing the laboratory facilities and for making available the digital computer for this project.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

ABSTRACT

A hardware pseudo-random number generator was designed and implemented as an inexpensive peripheral device for a minicomputer. To obtain acceptable pseudo-noise statistics, the design was based on the linear interconnection of multiple shift registers. A 36-bit configuration of six registers having six bits each was evaluated and a list is given of all possible interconnection combinations which result in maximal-length sequences.

From the full list of maximal-length sequence combinations several were selected for statistical testing of the resulting pseudo-random numbers. The results of these tests are summarized, and a final hardware implementation was chosen which produces statistically satisfactory, uniformly distributed, 18-bit, pseudo-random numbers.

A complete logic design is given for interfacing the pseudo-random number generator to a PDP-9 minicomputer.

CHAPTER 1

INTRODUCTION

Many of the applications for modern digital computers depend upon the ready availability of random numbers. The response of very complex physical or social systems to the alteration of various parameters may be studied by application of the Monte Carlo method to a suitable computer model. In some simulation and optimization studies, the search for an optimal set of parameter values may be carried out by simply trying a very large number of random combinations and successively saving those sets which are superior to the previous best set. The number of possible combinations of parameter values rises dramatically as the number of parameters increases, and the Monte Carlo technique depends upon a large number of trials, so that it is only the computational speed of the electronic digital computer that allows such methods to be applied. Furthermore, since the trial of all possible combinations is not being considered, the statistical interpretation of the results will depend upon the statistical properties of the values which are actually tried. If a very large number of trial values are to be used, it will be sufficient to know the statistical properties of the source set of which the trial values

constitute a sample. From a practical standpoint for the mathematical analysis, a uniform random variable (having identical probability of taking any possible value between two limits) is a suitable choice. Other more specialized applications of digital computers, such as flight simulators, may utilize random numbers directly to simulate natural phenomena, such as wind gusts, which vary in an unpredictable manner. In the event that uniformly distributed values are not desired, there are algorithms available to transform the values to a Gaussian distribution, or to other forms as needed.

Even though nature provides random processes, such as thermal molecular motion and radioactive decay, which might be tapped as a source of random values, it is quite difficult to control their statistical properties. Two such random number generators have utilized this approach specifically for digital computers (Belt 1969, RAND Corporation 1955) and no doubt other examples exist. Analog "noise" generators to develop a randomly varying voltage, for use in analog computation and in communications industry applications, are also available. Most digital computation employs software to generate deterministic sequences of numbers satisfying the essential statistical tests for randomness (Shreider 1964). The use of such pseudo-random numbers, instead of truly random ones, has one redeeming feature--during the development and debugging of programs,

and for comparison purposes, identical data may be provided for each computer run.

The statistical quality of such pseudo-random numbers has been shown to be limited with regard to the uniformity of distribution of strings of output numbers, and the non-uniformity is more severe for shorter-word-length computers (Coveyou and MacPherson 1967). The sequence length is also shorter for the shorter-word-length computers and may be as short as 65,535 numbers for the 16-bit minicomputers. The sequence length may be extended and the statistical quality improved by performing the generating algorithm in multiprecision arithmetic. This technique is particularly unattractive for many minicomputer applications in real-time interaction with laboratory experiments, or in real-time data or signal processing, for the extra time required by the multiprecision operations will inevitably degrade the effectiveness of the computer as an element in the control loop.

A hardware device for minicomputers which would generate a very long sequence of pseudo-random numbers without any software overhead is thus a logical development. The remainder of this paper will describe a project to implement and evaluate such a device.

CHAPTER 2

BACKGROUND

An excellent choice for a hardware device to generate pseudo-random numbers employs a linear sequential circuit of the type often called recursive shift-register. In this method, a binary shift register is supplied with an input (0 or 1) produced from the sum, modulo 2, of certain selected bits within the register. Of particular interest are the maximal-length sequences, of period 2^{n-1} for an n-bit register, which result when the bits selected for the feedback are properly chosen. Such maximal-length sequences are usually referred to as m-sequences. The analytical study of m-sequences (Tausworthe 1965, Golomb 1967) has established three "randomness" properties which are directly applicable to the problem of pseudo-random number generation. These three "randomness" properties apply to all m-sequences: (1) in each period of an m-sequence there will be 2^{n-1} ones and $2^{n-1}-1$ zeros; (2) in each period every possible combination of n consecutive bits, except n zeros, occurs exactly once, and as a consequence half the runs (strings of identical bits) have length 1, one quarter have length 2, etc., until there is one run of zeros of length n-1 and one run of ones of length n; (3) the auto-correlation function over the full

sequence period is two-valued, as for example one formulation (Korn 1966) gives the discrete auto-correlation function as

$$\psi(k) = \frac{1}{2^n-1} \sum_{i=1}^{2^n-1} a_i a_{i+k}$$

which will be either $\frac{2^{n-1}}{2^n-1}$ (for k = integer multiples of the period) or $\frac{2^{n-2}}{2^n-1}$ (for all other values of k). Tausworthe (1965) has further shown that binary numbers formed from consecutive bits of an m-sequence have essentially the same mean and variance as a uniform distribution with the deviation from the uniform decreasing as the sequence length increases. These theoretical results apply to the entire period of an m-sequence while the usefulness of such a sequence as a pseudo-random number source for a computer depends upon <u>not</u> using its full length.

An additional property of a pseudo-random number generator that is desirable is that it should produce M-tuples having a binomially distributed weight, i.e., a binomial distribution in the number of ones in a sequence of M consecutive bits. If this property can be achieved, then a random variable having very nearly a Gaussian distribution may be produced by summing the bits in a suitably sized M-tuple. Random variables having a Gaussian distribution are frequently used in simulation studies. Some recent

research (Lindholm 1968, Jordan and Wood 1973) has indicated that the M-tuple weight distribution is dependent upon the structure of the recurrence relation for the sequence and not just the fact that it is maximal-length. In particular, it is shown that a skewed weight distribution results when only a few bits are involved in the feedback sum. The theoretical tests derived by Lindholm will be applied below to ensure a nearly binomial M-tuple weight distribution for the implementation of this project.

Theoretical Discussion

A general expression for the recursive definition of a shift register of the type being discussed is given by the value of the kth bit

$$X_k = \sum_{i=1}^{n} C_i X_{k-i} \qquad \text{(modulo 2)}$$

where n is the number of bits in the register, the X_{k-i} are the individual bit values, and the n values of the C_i are either 0 or 1. This recursive relation may be discussed in terms of a polynomial function of the form

$$f(X) = \sum_{i=0}^{n} C_i X^i \qquad \text{(modulo 2)}$$

where $C_o = C_n = 1$ and the rest of the C_i are 0 or 1. With the proper values of the coefficients, C, filled in and the limit, n, determined by the number of bits in the shift

register this expression becomes the characteristic polynomial. It is a necessary and sufficient condition for the sequence generated by the corresponding shift register arrangement to be a maximal-length sequence if the characteristic polynomial is primitive, i.e., the polynomial has no factors of degree less than n and the smallest positive integer, p, for which f(X) divides $1-X^p$ is 2^n-1 (Golomb 1967).

When several shift registers are interconnected it will be necessary to derive the characteristic polynomial from the parameters of the implementation, i.e., the individual register lengths and the specific interconnections that have been made. If, for example, the recursion relation for each register is

$$X_k^{(i)} = X_{k-q}^{(i)} + X_{k-d}^{(j)}$$

i.e., the input to the ith register is the sum (modulo 2) of the the qth bit of that register and the dth bit of another register, the delay operator, D, defined by $D^d X_u = X_{u-d}$, may be introduced and the expression becomes

$$X_k^{(i)} = D^q X_k^{(i)} + D^d X_k^{(j)} .$$

Thus, a set of simultaneous equations (one equation for each shift register) are defined whose system determinant is the characteristic polynomial in the delay operator which then

leads directly to the characteristic polynomial representing the combined shift registers.

One of the statistical tests which can be applied to pseudo-random numbers (or the bits of an m-sequence) is the measurement of the moments of the distribution of weights of M-tuples from the given sequence. The weights are defined as the number of ones within M consecutive bits of the sequence. The distribution is essentially binomial with only a slight aberration introduced by the absence of the string of zeros of length equal to, or greater than, n (the length of the shift register). Analytical work has shown (Lindholm 1968) that the first and second moments of this distribution depend only upon M and N (the subsequence length and the m-sequence length, respectively, where N is 2^n-1) but that the third and higher moments are affected by the characteristic polynomial. Thus, skewed distributions in this test can be avoided by discarding those characteristic polynomials which theory indicates will have an excessive third moment. The general expression for the third central moment is

$$S_c^3 = -\frac{M^3}{N} + 3!\,\frac{N+1}{N}\left\{MB_3^* - \sum_{i=1}^{B_3^*} d_i\right\} + \frac{3M^2(N+1-M)}{N^2}$$

where M is the length of the subsequence, N is the length of the m-sequence, and B_3^* is the number of trinomials of degree less than M that are divisible by the characteristic

polynomial, f(X), and d_i are their degrees. For practical cases, where M is greater than the length of the shift register yet much shorter than the full m-sequence, the first and last terms of the equation are negligible and the third central moment is determined by the low degree trinomials.

Lindholm (1968) has also provided an algorithm to search for trinomials that are divisible by f(X). It is shown that any trinomial of the form $g(X) = 1 + X^{d-c} + X^{d}$ will have a factor, f(X), if, and only if, the (n-1)-tuples $(a_{c+1}, a_{c+2}, \ldots, a_{c+n-1})$ and $(a_{d+1}, a_{d+2}, \ldots, a_{d+n-1})$ are identical, assuming that $a_o = 1$ and a_1 through $a_{n-1} = 0$. This search can be carried out to any desired degree to provide an estimate of the third moment value.

Implementation Decisions

A recent report (Hurd 1972) describes a novel method of interconnecting several shift registers to simultaneously produce widely spaced phase shifts of a maximal-length sequence. It seemed appropriate as a development project to apply this method to a specific realization, tailored to minicomputer characteristics and requirements, and to determine whether the pseudo-random numbers thus generated were indeed statistically acceptable. Other constraints were imposed (see Table 1) mostly by practical considerations.

Table 1. Design Constraints

1. Digital computer	- DEC PDP-9
2. Word size	- 18 bits
3. I/O method	- single word parallel transfer via standard I/O bus on computer command
4. Controls	- separate reset, advance, and read commands
5. Sequence length	- at least 2^{30} words
6. Logic family	- TTL 7400 series
7. Speed	- next word within 5 microseconds

There are a relatively small number of distinct shift register types available in TTL integrated circuit form which may be considered for use in this device. The most commonly used ones have lengths of 4, 5, or 8 bits, but of course, other lengths may be readily made by using these shift eegisters and individual flip-flops in various combinations. A preliminary logic design was worked out which provided 36 bits total in 6 separate registers, each of which consists of one 5 bit standard shift register and one flip-flop. The individual flip-flop is not used simply to extend the shift register by one bit, but is a necessary part of the design since the implementation requires the last bit of each register to toggle, i.e., to complement

itself whenever a logical one is "shifted" into it. The resulting 6 by 6 array of bits has no particular significance but resulted naturally from the inclination to avoid the extremes (either too many registers of short length or too few registers of long length) and for the computer word size (18 bits) to be a multiple of the number of registers. The option to change this design decision was retained through several phases of simulation and the application of theoretical tests but after passing all tests this configuration was adopted for the actual hardware construction.

A somewhat generalized representation of the logic of one shift register (6 bits long) is shown in Figure 1 and it can be seen that the parameter, d, is yet to be chosen for each of the six shift registers. The recursion relation for a register in terms of the delay operator, D, may be written as

$$X_i = \frac{D^6}{1+D} X_{i-1} + D^{d_{i-2}} X_{i-2}$$

where the subscripts, i-2 and i-1, are taken modulo 6. The set of six simultaneous equations is shown in matrix form in Figure 2. The resulting characteristic equation (from which the characteristic polynomial for the corresponding shift register is taken directly) is shown in Figure 3. The circular symmetry of the shift register interconnections makes the cyclical permutations indistinguishable for a

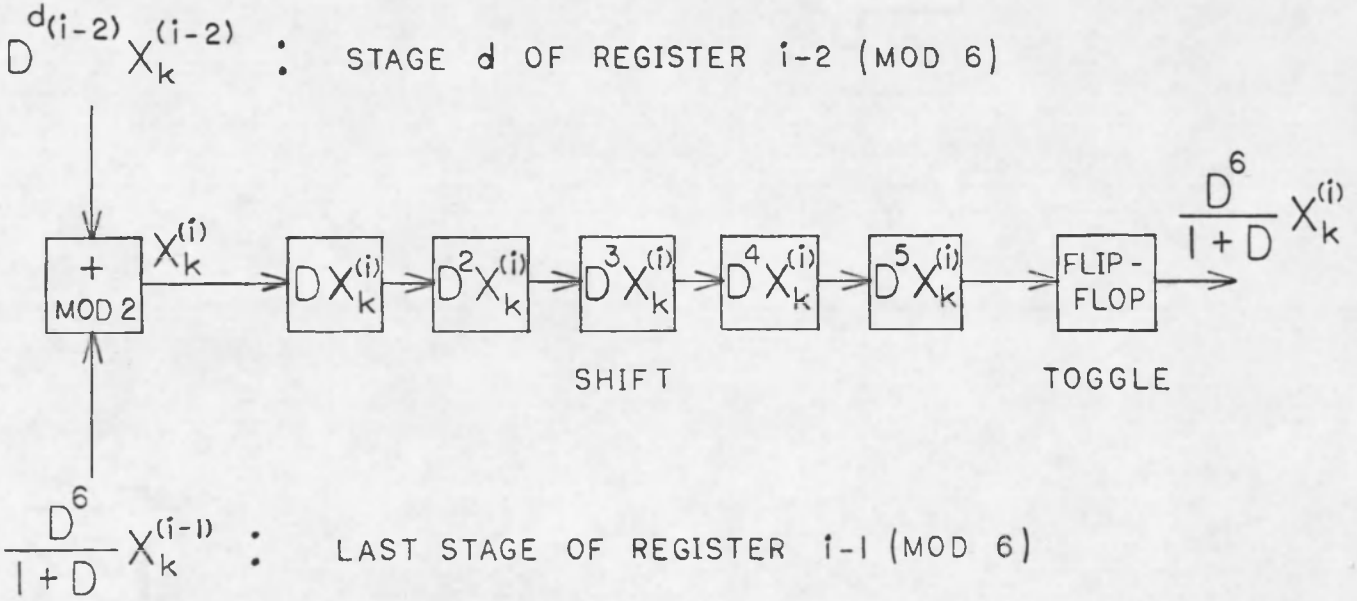


Figure 1. General 6-bit Register

$$\begin{bmatrix} (1+D) & 0 & 0 & 0 & (1+D)D^{d_4} & D^6 \\ D^6 & (1+D) & 0 & 0 & 0 & (1+D)D^{d_5} \\ (1+D)D^{d_0} & D^6 & (1+D) & 0 & 0 & 0 \\ 0 & (1+D)D^{d_1} & D^6 & (1+D) & 0 & 0 \\ 0 & 0 & (1+D)D^{d_2} & D^6 & (1+D) & 0 \\ 0 & 0 & 0 & (1+D)D^{d_3} & D^6 & (1+D) \end{bmatrix} \cdot \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \end{bmatrix} = 0$$

Figure 2. Delay Operator Equations

$$D^{36} + (1+D)^6 + (1+D)^2 D^{24} \sum_{i=0}^{5} D^{d_i} + (1+D)^6 \prod_{i=0}^{5} D^{d_i}$$

$$+ (1+D)^6 (D^{d_0} D^{d_2} D^{d_4} + D^{d_1} D^{d_3} D^{d_5})$$

$$+ (1+D)^4 D^{12} [D^{d_0}(D^{d_2}+D^{d_3}+D^{d_4}) + D^{d_1}(D^{d_3}+D^{d_4}+D^{d_5}) + D^{d_2}(D^{d_4}+D^{d_5}) + D^{d_3} D^{d_5}]$$

WHERE d_i (FOR $i=0,1,2,3,4,5$) ARE THE INTERCONNECTION PARAMETERS

Figure 3. Characteristic Equation

given set of "d" parameters; thus, there are only about 2600 distinct sets of "d" parameters possible with the six-register arrangement. Figure 4 shows the interconnection pattern, and the resulting symmetry, along with the assignment of the "d" parameters. A computer program was written to search exhaustively through all permutations of these parameters to produce a list of those combinations having primitive characteristic polynomials, i.e., those combinations which would produce a maximal-length sequence. The program iterates through three phases: (1) generates the next set of parameters, skipping over cyclical permutations of previously tested sets; (2) calculates the characteristic polynomial for the parameter set; and (3) tests the polynomial to determine whether it is primitive. Initially, the entire program was written in FORTRAN, but subsequently the heavily used portions of the program were replaced with assembly language subroutines to improve the computation speed. The complete list of 85 parameter sets, and the corresponding characteristic polynomials, are given in Appendix A. Close inspection of that list will reveal that any set of parameters taken in reverse order will produce the same polynomial. For example, the parameter set 223343 taken in reverse order is 343322, which cyclically permutes to 223433, and both produce the same polynomial. A few cases, such as 111353, are themselves symmetrical forward and backward and so do not appear in both forms, thus the

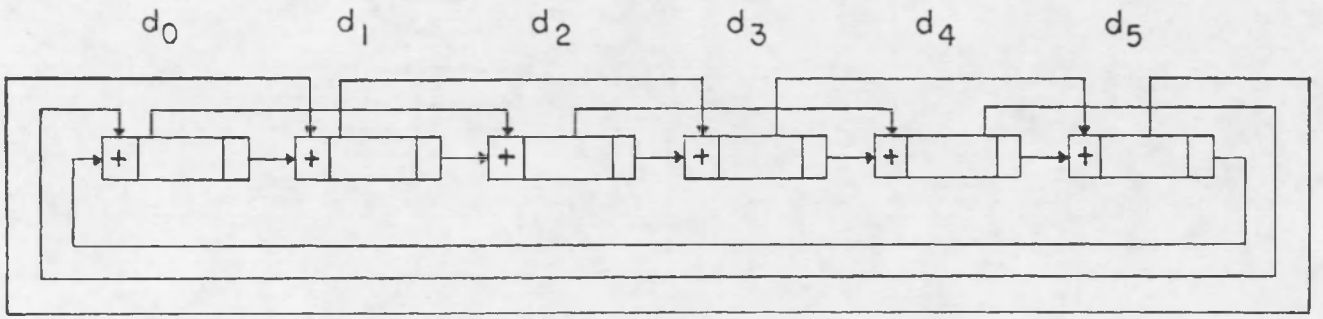


Figure 4. Six-register Interconnection Pattern

listing actually contains 45 different primitive polynomials (each giving a maximal-length sequence) which may be implemented in the proposed six register arrangement by an appropriate choice of the register interconnections. The greatest number of non-zero terms in any of the polynomials is 23, and there are five parameter sets giving that number. These five were selected for further study, with the intention of evaluating all of them in the actual hardware before making a final implementation decision.

A computer program was written to determine whether a given polynomial, f(X), is a factor of any higher degree polynomials of trinomial form. The motivation for this search and the method used were discussed in greater detail above. The main program is in FORTRAN and one assembly language subroutine is provided to perform the bit-by-bit comparison of two strings stored in array form. The five characteristic polynomials being considered were tested by this program and in all cases there were no higher degree trinomials found of any degree less than 4000 (which was the limit of the program search).

CHAPTER 3

LOGIC DESIGN

Shift Registers

The hardware implementation is based on the interconnection of six individual shift registers of six bits each. In order to produce a pseudo-random number of 18-bit length the six shift registers will be clocked (shifted) three times and the sequentially output bits saved in flip-flops for subsequent transfer to the computer. A schematic representation of one shift register is shown in Figure 5 and its operation will be described here. It may also be compared with Figure 1 of the previous chapter. A full set of schematic diagrams are included in Appendix C.

The input to the shift register is produced by the XOR (exclusive or) gate which has two inputs, one from X_6 of the preceding shift register, and the other from a selected bit, X_1 through X_5, of the second preceding shift register, as determined by the corresponding interconnection parameter. With each pulse on the "clock" line the bits of the shift register move one position (X_4 to X_5, etc.) and the input bit is set into X_1. A separate signal clock, but synchronized with this one, is supplied to the inverter, I2, and in turn clocks the three flip-flops, X_6, X_7, and X_8.

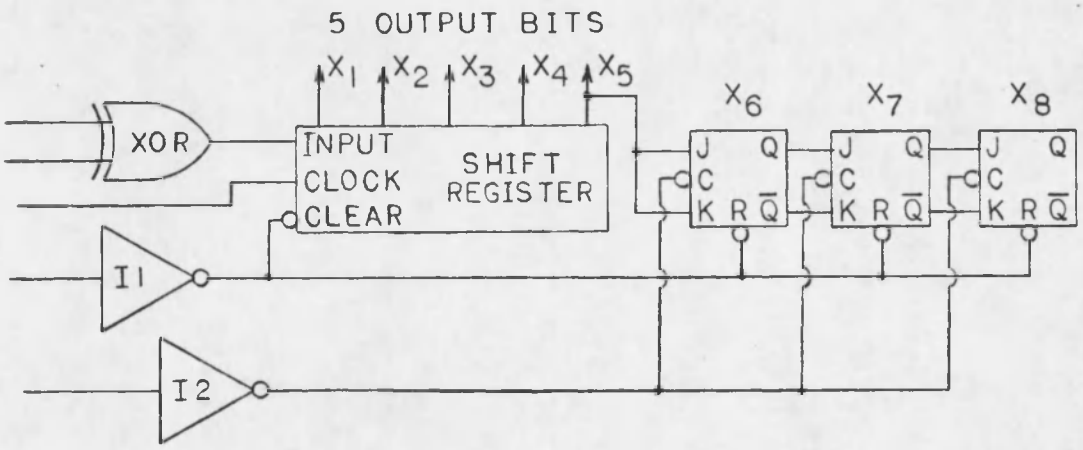


Figure 5. Individual Shift Register

Flip-flop X_6, with both J and K inputs driven by the output from X_5 of the shift register, performs the toggle function described previously and is the logic point at which a binary m-sequence will appear. X_7 and X_8 are connected as a shift register to save the two previous bits of the m-sequence produced at X_6. A signal into the inverter, I1, will clear the shift register and the three flip-flops. The five-bit shift register is provided with a preset input which would allow initialization of the contents of the shift register to any other state besides the zero condition, however, there was no need to use this feature in the present design. The logic signals to clear and clock these circuits are generated in a control logic section and distributed to all six of the shift register sections. The inverters I1 and I2 are necessary to comply with the fan-out and fan-in requirements of the TTL logic gates being used (Texas Instruments, Inc. 1972).

Control Logic

The major function of the control logic section is the generation of clock pulses to be used for shifting the six individual registers. Three clock pulses are required; thus, a four-state sequential circuit is provided to count the clock pulses. The sequential circuit remains in an idle, or reset, state until operation is initiated by a

computer command, and it then generates three clock pulses and reverts back to the idle state.

The oscillator frequency used to generate the shift clock pulses is not a critical parameter. It must be less than the specified maximum clock frequency for the integrated circuit shift registers (10 MHz in this design) and it must be fast enough to produce four cycles within the desired maximum time interval between successive 18-bit pseudo-random numbers. This project design utilized an 8.625 MHz crystal oscillator which makes successive 18-bit numbers available in less than 500 nanoseconds. At this speed there are no restrictions in programming the PDP-9, and other minicomputers would be similarly unrestricted in program controlled I/O operations.

The following description of the operation of the control logic section refers to the schematic designations appearing in Appendix C and conforms to the notational conventions described there.

Three control signals from the digital computer (through the interface logic described below) initiate all operations of the shift registers. A "clear" signal is produced at N1.11 by the presence of PWR. CLR. = 0 at input N1.13 or by the presence of IOP1 = 1 at input I4.5 and this "clear" signal is distributed to one gate and six inverters to provide sufficient fan-out to clear all the shift registers and all flip-flops in the pseudo-random

number generator. The control signal IOP4 at input I4.1 initiates the generation of the next pseudo-random number by setting flip-flop F11.3. This is an unconditional operation, since the J input is "1" and the K input is "0." The resulting "1" at F10.8 enables the two-bit binary counter, F10, and it advances through the four possible states sequentially with each clock pulse at input F10.9. (The clock pulses for this counter come from a continuously running oscillator formed with three inverters of the I3 circuit, the final output being taken from I3.6.) When both bits of the counter, F10, are "1" the gate N1.8 produces a signal which clears the F11.3 flip-flop and the counter is then disabled after it reaches the cleared state. The gate N1.3 decodes the state of the counter and produces a "1" for the three non-cleared states and a "0" for the cleared state. This signal is applied to input N1.4 where it is "ANDed" with the continuous clock to selectively allow three clock pulses to pass through the gate to N1.6 for each full cycle of the counter. These three clock pulses are buffered through inverter I3.12, and then six individual inverters for the clocking of each shift register to produce the next pseudo-random number. Of course, after three shift clocks the four-state counter is back in the cleared state and is disabled from further counting by the "0" state of flip-flop F11.3, so that the control logic remains in this idle condition until the next command from the computer is received.

Computer Interface

The remaining portion of the logic required for this pseudo-random number generator provides the interface to the digital computer. The discussion here is applicable to the DEC PDP-9 specific characteristics (Digital Equipment Corporation 1968) while the preceding paragraphs concerned only the TTL portion of the generator which would remain unchanged and could be adapted to any minicomputer, with a word size of 18 bits or less, through an appropriate interface. The function of the interface is to translate logic voltages and circuit impedances and to provide decoding of computer commands directed to the device "on the other side" of the interface. For this purpose DEC series M logic modules were selected, which have been designed specifically for this purpose (Digital Equipment Corporation 1971). The basic PDP-9 I/O structure consists of an 18 bit bi-directional data bus and a set of 17 uni-directional address and control lines.

From the PDP-9 a set of six address bits (and two auxiliary device-select bits not used in this case) are distributed to all peripheral devices. Each peripheral device will be assigned a unique device number and when this number appears on the address lines a gate is enabled in the interface and the attached device is said to be "selected." Three timing signals (IOP1, IOP2, and IOP4) are also distributed to all peripheral devices but normally they are

"ANDed" in the interface with the "device selected" signal so that only one device at a time receives these signals. When combinations of these three timing signals are insufficient to convey the set of commands required by the peripheral device, the bi-directional data bus is used to provide the additional bits of information. In the present instance, the pseudo-random number generator needs only three commands so that the data bus is not used for output from the computer. The three commands implemented in this interface are: (1) IOP1--Reset--to clear all control logic and to set the shift registers to their initial condition, i.e., all zero except one bit; (2) IOP2--Read--transfers the present contents of the pseudo-random number generator output registers to the PDP-9 I/O bus and signals the computer to read these data into the accumulator; (3) IOP4--Advance--initiates the generation of the next number. The separation of the "read" and "advance" commands is not absolutely necessary and in most programs the two commands are issued by the same program instruction; however, some diagnostic (trouble-shooting) procedures are simplified if the data being transferred from the peripheral device can be held constant.

A DEC M103 logic module (Device Selector) provides the address decoding and the voltage level conversion and the same module also contains level converters for the IOP timing pulses. Two DEC M632 and one M633 logic modules

(Bus Drivers) provide the voltage level conversion for the 18 data bits and one control line from the TTL logic to the computer.

The TTL integrated circuit shift registers and control logic were fabricated on a DEC type W941 wire wrappable module which will accommodate 25 integrated circuit packages. The design implemented in this project required 24 I.C. packages and the 25th space was used for mounting the crystal and resistors used in the clock oscillator circuit.

The power requirements for this device are very modest. A +5 volt supply at 1.1 amps is required for the TTL logic and the four interface cards. A -15 volt supply at 0.5 amps is required for the four interface cards and for the I/O cable terminators.

CHAPTER 4

STATISTICAL TESTS

The only practical way to establish the acceptability of a pseudo-random number generator is to perform a series of statistical tests on samples produced by the generator. These tests are based on a comparison of observed results with theoretically estimated values which would be expected to occur if the samples had been produced by a truly random source. In most cases a hypothesis test is performed, based on a calculated test statistic and a given significance level. Every result is, of course, a random event and even a successful test only tells us that the pseudo-random numbers are "probably" indistinguishable from true random numbers. In the end, the user must be responsible for establishing the criteria for randomness and determining the suitability of a pseudo-random number generator for a specific application; obviously some applications will be more demanding than others. The tests to be described in the following paragraphs were intended to establish the acceptability of the hardware pseudo-random number generator for general-purpose use. The results of these tests are summarized in Table 2. Useful lists of statistical tests have been published (Knuth 1969, Jansson 1966) intended for

Table 2. Statistical Test Summary

Test	Theory	A	B	C	D	E
Frequency above 0.05	20	20	16	21	14	22
Frequency below 0.95	20	18	28	17	21	28
Independence above 0.05	112.5	133	119	116	104	114
Independence below 0.95	112.5	125	100	118	96	140[a]
Weight above 0.05	20	17	21	16	24	24
Weight below 0.95	20	24	10[a]	18	18	18
Runs above 0.05	5	5	3	5	7	3
Sum of |deviation| from theory		42	42	19	38	45

Shift-register interconnect parameters:

A - 113145
B - 113415
C - 113453
D - 133244
E - 234334

[a]Indicates excessive deviation from expected value.

pseudo-random number testing, and the tests applied here were drawn from those works and from the comments of other authors cited in the references.

Frequency Distribution

The 18-bit pseudo-random numbers were first tested for uniform distribution. The full range (all possible combinations of 18 bits) was divided into 32 equal class intervals, and frequency counts for each interval were determined for a sample of 8192 numbers. The best number of intervals for this sample size would be 35 (Mann and Wald 1942) but choosing a power of 2 makes the sorting program more efficient and no great harm should result as long as the number of actual intervals is reasonably close to the theoretical best value. A chi-square statistic was then computed as the sum

$$\chi^2 = \sum_{i=1}^{32} \frac{(R_i - X)^2}{X}$$

where each R_i is the observed frequency count in each class interval and X is the expected value for a uniformly distributed variable, i.e., the number of samples divided by the number of intervals. The result of this summation was compared to the tabulated limits corresponding to the 5% and 95% significance levels with 31 degrees of freedom. This test was repeated 100 times and the results, along with the results of the other statistical tests, are given in

Appendix B. The frequency distributions in two, three, and four dimensions were determined by similar tests performed for pairs, triples, and quadruples of numbers. The 2-dimensional test used an 8 x 8 array of regions; the 3-dimensional test used a 4 x 4 x 4 array; and the 4-dimensional test used a 4 x 4 x 4 x 4 array. In the 4-dimensional case, there are a total of 256 regions (4^4 class intervals) in which frequency counts are made, and this substantially exceeds the "best" number indicated above; however, the alternative of using 3^4 regions is unattractive (2^{18} is not divisible by 3; thus the resulting regions are not quite of equal size), and 2^4 regions is probably too few, since only the most significant bit of each sample is then required to sort it into 2 regions. The results of the 2- and 3-dimensional tests are compared to chi-square tabulated values for 63 degrees of freedom, and the 4-dimensional test results are compared to values for 255 degrees of freedom. None of the five implementations being evaluated showed any significant deviation from the expected uniform distribution.

Statistical Independence of Bit Pairs

The individual bits of the pseudo-random number were next tested for statistical independence. This test is based on the calculation of a contingency-table statistic (Korn and Korn 1968)

$$nf^2 = \frac{N \cdot (N_{00}N_{11} - N_{01}N_{10})^2}{(N_{00}+N_{01})(N_{10}+N_{11})(N_{00}+N_{10})(N_{01}+N_{11})}$$

where N is the number of samples and the N_{ij} are the number of occurrences of the combination i and j in the pair of bits being tested (i and j, of course, must be 0 or 1). The hypothesis of statistical independence may then be accepted if nf^2 does not exceed the chi-square value for one degree of freedom at the desired level of significance. The comparisons were made at the 5% and 95% levels. All the 153 possible bit pairs within the 18-bit number were evaluated for statistical independence, using a sample size of 10,000 numbers. Similar tests were performed for each bit of a word and the corresponding bit of a subsequent word over an interval of 1, 2, 3, and 4 words. These tests were repeated 10 times. One of the shift-register-interconnection implementations being evaluated deviated below the 95% significance level by an excessive amount and is, therefore, considered not acceptable (see Table 2).

Weight Distribution

The weight distribution of the pseudo-random numbers was compared with a binomial distribution (weight meaning the number of ones in the binary representation of the number). For applications involving the generation of a random variable with Gaussian distribution, through addition

or filtering of pseudo-random bits, it is important to have a binomial distribution. The distribution of weights in a sample of 10,000 numbers was determined and a chi-square statistic was calculated as the sum

$$\chi^2 = \sum_{i=0}^{18} \frac{(R_i - X_i)^2}{X_i}$$

where R_i are the observed occurrences of each weight and X_i is the expected value, assuming a binomial distribution. The low and high ends of the distribution categories were lumped together, as required, so that no category had an expected occurrence of fewer than 5. This minimum value has been suggested as a general rule of thumb when applying the chi-square test. The chi-square value was compared with the tabulated values at the 5% and 95% significance level for the appropriate number of degrees of freedom, viz. with 10,000 samples the weights of 0, 1, and 2 will be combined (and also weights of 16, 17, and 18) giving 14 degrees of freedom. This test was performed 100 times and similar tests were performed over M-tuples of 36, 54, and 72 bits length. One of the shift-register implementations had an excessive deviation in this test.

An additional calculation was made on the observed weight distribution to check the third central moment, which indicates skewness. For this purpose, the first three values of v_k were calculated from the formula for the kth

moment about A

$$v_k = \sum_{i=1}^{n} (X_i - A)^k / n$$

where A was taken as the mid-point of the distribution range. The third central moment is then directly calculated from

$$\mu_3 = v_3 - 3v_1v_2 + 2v_1^3 .$$

The highest (magnitude) value of μ_3 was printed by the program and the relatively small values found by this test confirm the indication of the chi-square test that there is no significant deviation from a binomial weight distribution.

Run Analysis

Successive pseudo-random numbers interpreted as a continuing string of binary bits were analyzed for runs. (A "run" of length m is a series of m identical bits.) A sample length of 180,000 bits was used, and the test procedure (Miller and Freund 1965) was carried out as follows: First, a μ value is calculated as

$$\mu = \frac{2n_0n_1}{n_0+n_1} + 1$$

where n_0 and n_1 are the number of bits equal to 0 and 1, respectively. Second, a σ value is calculated as

$$\sigma = \sqrt{\frac{2n_0n_1(2n_0n_1-n_0-n_1)}{(n_0+n_1)^2(n_0+n_1-1)}}$$

with n_0 and n_1 as defined above. Third, the test statistic

$$Z = \frac{U-\mu}{\sigma}$$

where U is the total number of runs, is calculated and compared to the tabulated value for a normal distribution function at the 5% level of significance. This test was performed 100 times. Three of the implementations had a marginal deviation but not sufficient to make them unacceptable on the basis of this test alone.

The results of the above tests for each of the five different shift register configurations are assembled in Appendix B. The total number of each test which exceeded the 5% and 95% significance level are shown in Table 2.

Visual CRT Displays

Three additional tests were performed which do not have quite as formal a statistical interpretation but are, nevertheless, very powerful in the detection of non-random properties in pseudo-random number sequences. In each case the essential feature is the display of pictorial information on a graphical CRT and the presence of an intelligent observer who is "programmed" to look for patterns.

In the first test, a frequency distribution was calculated with 256 class intervals and the cumulative sums displayed throughout the program execution. This procedure may reveal significant deviations from the expected uniform distribution which are either too localized or too short in duration to be detected by the more conventional statistical test described previously.

In the second test, an auto correlation function for a sequence of 18-bit pseudo-random numbers over delay intervals ranging from zero to 255 numbers was calculated and displayed. The zero delay correlation, of course, has the maximum possible value and serves as a reference for comparisons to the rest of the points. This procedure should reveal any periodicities of the pseudo-random number generator which are shorter than 255 numbers in length.

In the third test, successive pairs of pseudo-random numbers are plotted with one number providing the X-axis value and the other providing the Y-axis value. As an added option the program may be directed to display only those pairs which immediately follow a "keyword," i.e., a number occurring in sequence which has a value between specified limits. This procedure may reveal non-random patterns and, in fact, this test was the basis for the rejection of two of the five shift register configurations being evaluated.

The analysis of the statistical tests has resulted in the rejection of three of the five implementations.

Configuration B was rejected on the basis of the weight distribution test. Configurations D and E were rejected on the basis of a distinct pattern (definitely non-random) in the third CRT visual test and E also gave unacceptable results in the statistical independence test. The two remaining configurations passed all of these tests but configuration C was selected for the final implementation in this project on the basis of a smaller total deviation (for all tests) from the theoretically expected values.

CHAPTER 5

CONCLUSIONS

The pseudo-random number generator developed in this project has been shown to produce numbers which are statistically acceptable for general purpose use on the basis of the listed randomness tests. Tested against the hypothesis that the numbers are drawn from a set of uniformly distributed random numbers there was no evidence to reject the hypothesis at the 5% significance level. The present development of the theory of shift register generated binary m-sequences does not provide sufficient guidance to allow the selection of shift-register configurations which will guarantee statistically satisfactory performance. Even in the five cases evaluated in this project there were some which were satisfactory and some which were not.

A natural extension of the work of this project would be the empirical evaluation of all the 45 primitive polynomials available in this hardware configuration to see if there is any distinction between those that produce acceptable pseudo-random sequences and those that do not. Further empirical studies of this type may provide clues to assist the development of theory in this area, and the

accumulation of empirical data provides the basis for testing new theoretical hypotheses.

APPENDIX A

LISTING OF M-SEQUENCE INTERCONNECTIONS

The tabulation on the following pages contains all of the interconnection configurations that will produce m-sequences. In column one is the set of six "d" parameters which specify the interconnections between shift registers. In column two is the resulting characteristic polynomial with the least significant coefficients at the left. In column three is the number of non-zero coefficients in the polynomial.

REG. TAPS	PRIMITIVE POLYNOMIAL (LSB FIRST)	T
111125	1010000111100010210100100010110100001	15
111152	1010000111100010010100100010110100001	15
111353	1010101000000000010001000010101010001	11
111354	1010101111111110101000111010010110001	21
111453	1010101111111110101000111010010110001	21
112334	1010100000000001100010110010001000001	11
112354	1010101000000000111010110000110011000001	15
112525	1010111101000100010000011001000000000001	13
113145	1010101111111111111000101111001010110001	23
113245	1010101000000000000001000101100110000101	11
113254	1010101100000000101011101101100110000101	17
113335	1010101100000000100100010000100010000101	11
113415	1010111101111101111111101010001010110001	23
113453	1010101011111111101101110100011111000001	23
113525	1010100010111101111100021100111100000001	19
113543	1010101011111111101101110100011111000001	23
114332	1010100000000000110001011001000210000001	11
114344	1010101011111111101000100000111100000001	19
114434	1010101011111111101000100000111100000001	19
114523	1010101100000000101011101101100110000101	17
114532	1010101000000000111011100001100110000101	15
115143	1010111101111101111111101010000101110000101	23
115252	1010111101000100010000211001000000000001	13
115253	1010100010111101111100021100111100000001	19
115333	1010101100000000100100010000100010000101	11
115413	1010101111111111111000101111001010110000101	23
115423	1010101000000000000000100010110011000101	11
121243	1010100111111101001000010000111100000001	17
121342	1010100111111101001000010000111100000001	17
121515	1011111101001000110001100101100000000001	17
122415	1010000010111011100010011001001111000001	17
122533	1010100010000001110001001110101010000101	15
123525	1010100010100011111110110011001000000001	17
124535	1010101010100000000010111101001100000001	15
124542	1010101000000000010001000010101010000101	11
125343	1010101010000000010010000011011110000101	13
125554	1010101000002200000000000111111111000001	13
131425	1010000010101000200010101111100110000101	15
131444	1010100010111111100101100100111100000001	19
131524	1010000010101000000010101111100110000101	15
132353	1010101011111111101001000011101001000001	19
132514	1010000010101000101111111111001100000101	19
133244	1010101011111111011101111101111000000001	23
133522	1010100010000001110001001110101010000101	15
134344	1010101001111111100101110000110100000001	19
134352	1010101010000000001001000001101110000101	13
135323	1010101011111111101001000011101001000001	19
135535	1010101001010000011110000021111101000001	17

```
141523    10100000101010001011111111111001100001    19
142215    10100000101101110100011001001111100001    17
143225    10101000101111101110001001000101101000 1    17
144233    10101010111111110111011111011110000000 1    23
144343    10101010011111111001011110001101000001    19
144535    10101010101010000100001010001100000001    13
145552    10101010000000000000000011111111100001    13
152234    10101000101111101110001001000101101000 1    17
152532    10101000101000111111101100110010000001    17
153542    10101010101000000000101111010011000001    15
153544    10101010101010000010000101000110000001    13
153553    10101010010100000111100000111110100001    17
154545    10101010010101010001111100010111110000 1    19
223343    10101010111111111101000100002111100000 1    19
223433    10101010111111111101000100000111100000 1    19
224354    10101010010000000101011101110001000001    15
224534    10101010010000000101011101110001000001    15
233324    10101011101111110011011100200111100000 1    21
234255    10101010001111111001010110100111000001    19
234334    10101010011111111101011101111110000001    23
234455    10101010000011111111100101001010000000 1    17
243255    10101010000111111100101011010011100000 1    19
243343    10101010011111111101011101111110000001    23
243425    10101011012100011101001010110001000 1    17
243455    10101010001010111121010101010100000001    17
245355    10101010000000000101110100011001000 1    13
245554    10101010000011111111010111101100000001    19
255354    10101010000000000101110100011001000 1    13
255434    10101010001010111101010101010100000001    17
255443    10101010000011111111100101001010000000 1    17
343455    10101010000010000001001111111010100000 1    15
343554    10101010000021000000100111111010100000 1    15
344345    10101010000211111112110121101101100000 1    21
344354    10101010000211111112101011011011000001    21
345555    10101010000000211111110112111010110100 1    21
353545    10101010001010111111000002111010000001    17
355554    10101010000000111111110110111010110100 1    21
```

SEARCH COMPLETE.

2635 PATTERNS CHECKED.

APPENDIX B

TEST PROGRAM COMPUTER OUTPUT

The following five pages are copies of the computer output for each of the interconnection configurations that were evaluated in hardware form.

The bit correlation program (actually a hypothesis test for statistical independence) has printed, under zero word interval, the results of all the 153 bit pairs within one word; in the non-zero intervals 18 bit pairs were evaluated.

The weight distribution program has output the highest (magnitude) value of the third central moment. The relatively small values found were as expected from theoretical considerations.

SHIFT REGISTER STATISTICAL TEST SUMMARY

FEEDBACK TAPS AT POSITIONS 1,1,3,1,4,5

N-DIMENSIONAL CLASS INTERVAL FREQUENCY DISTRIBUTION:
100 TESTS PERFORMED WITH 8192 SAMPLES PER TEST.

DIMENSIONS	BELOW 0.95	ABOVE 0.05
1	4	3
2	7	6
3	3	6
4	4	5

BIT CORRELATIONS:
10 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

WORD INTERVAL	BELOW 0.95	ABOVE 0.05
0	71	86
1	8	8
2	3	10
3	5	10
4	13	5

N-TUPLE BINOMIAL WEIGHT DISTRIBUTION:
100 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

LENGTH	BELOW 0.95	ABOVE 0.05	MAX. 3RD MOMENT
18	4	3	0.4849
36	8	4	1.3159
54	7	7	-2.7279
72	5	3	4.2373

RUN TEST:
100 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

5 OUTSIDE 0.05 LIMIT

SHIFT REGISTER STATISTICAL TEST SUMMARY

FEEDBACK TAPS AT POSITIONS 1,1,3,4,1,5

N-DIMENSIONAL CLASS INTERVAL FREQUENCY DISTRIBUTION:
100 TESTS PERFORMED WITH 8192 SAMPLES PER TEST.

DIMENSIONS	BELOW 0.95	ABOVE 0.05
1	8	5
2	8	6
3	7	3
4	5	2

BIT CORRELATIONS:
10 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

WORD INTERVAL	BELOW 0.95	ABOVE 0.05
0	93	86
1	7	9
2	10	12
3	4	11
4	11	15

N-TUPLE BINOMIAL WEIGHT DISTRIBUTION:
100 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

LENGTH	BELOW 0.95	ABOVE 0.05	MAX. 3RD MOMENT
18	3	4	-0.5186
36	2	4	-1.8744
54	3	4	3.4322
72	2	9	-4.4470

RUN TEST:
100 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

3 OUTSIDE 0.05 LIMIT

SHIFT REGISTER STATISTICAL TEST SUMMARY

FEEDBACK TAPS AT POSITIONS 1,1,3,4,5,3

N-DIMENSIONAL CLASS INTERVAL FREQUENCY DISTRIBUTION:
100 TESTS PERFORMED WITH 8192 SAMPLES PER TEST.

DIMENSIONS	BELOW 0.95	ABOVE 0.05
1	8	5
2	2	4
3	3	4
4	4	8

BIT CORRELATIONS:
10 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

WORD INTERVAL	BELOW 0.95	ABOVE 0.05
0	73	81
1	13	7
2	14	8
3	10	8
4	8	12

N-TUPLE BINOMIAL WEIGHT DISTRIBUTION:
100 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

LENGTH	BELOW 0.95	ABOVE 0.05	MAX. 3RD MOMENT
18	4	3	-0.5484
36	7	4	-1.8138
54	5	5	-3.5992
72	2	4	3.7278

RUN TEST:
100 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

5 OUTSIDE 0.05 LIMIT

SHIFT REGISTER STATISTICAL TEST SUMMARY

FEEDBACK TAPS AT POSITIONS 1,3,3,2,4,4

N-DIMENSIONAL CLASS INTERVAL FREQUENCY DISTRIBUTION:
100 TESTS PERFORMED WITH 8192 SAMPLES PER TEST.

DIMENSIONS	BELOW 0.95	ABOVE 0.05
1	7	3
2	7	4
3	4	5
4	3	2

BIT CORRELATIONS:
10 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

WORD INTERVAL	BELOW 0.95	ABOVE 0.05
0	74	73
1	2	7
2	2	8
3	14	7
4	4	9

N-TUPLE BINOMIAL WEIGHT DISTRIBUTION:
100 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

LENGTH	BELOW 0.95	ABOVE 0.05	MAX. 3RD MOMENT
18	4	5	-0.6044
36	4	4	1.7211
54	6	8	3.1071
72	4	7	-5.1351

RUN TEST:
100 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

7 OUTSIDE 0.05 LIMIT

SHIFT REGISTER STATISTICAL TEST SUMMARY

FEEDBACK TAPS AT POSITIONS 2,3,4,3,3,4

N-DIMENSIONAL CLASS INTERVAL FREQUENCY DISTRIBUTION:
100 TESTS PERFORMED WITH 8192 SAMPLES PER TEST.

DIMENSIONS	BELOW 0.95	ABOVE 0.05
1	8	5
2	4	8
3	3	4
4	13	5

BIT CORRELATIONS:
10 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

WORD INTERVAL	BELOW 0.95	ABOVE 0.05
0	103	77
1	7	6
2	17	11
3	7	8
4	6	12

N-TUPLE BINOMIAL WEIGHT DISTRIBUTION:
100 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

LENGTH	BELOW 0.95	ABOVE 0.05	MAX. 3RD MOMENT
18	5	6	-0.5426
36	3	4	1.7117
54	5	8	3.2022
72	5	6	5.4842

RUN TEST:
100 TESTS PERFORMED WITH 10000 SAMPLES PER TEST.

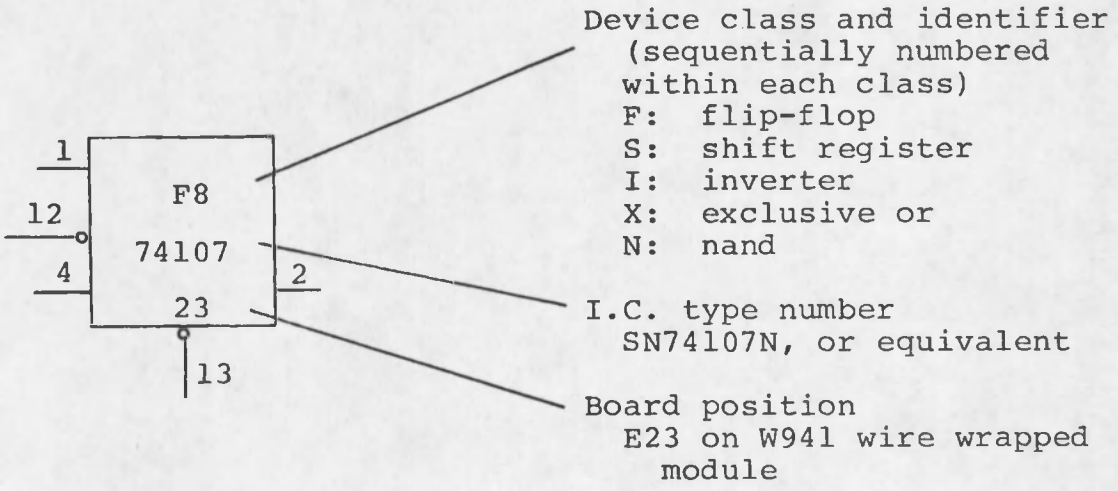
3 OUTSIDE 0.05 LIMIT

APPENDIX C

LOGIC DIAGRAMS

Five types of integrated circuits are used in the TTL portion of this device. The symbols used in these diagrams conform to those of the Texas Instruments Logic Handbook (1972) and that reference may be consulted for detailed specifications.

Notation in, and around, a logic symbol is interpreted as follows:



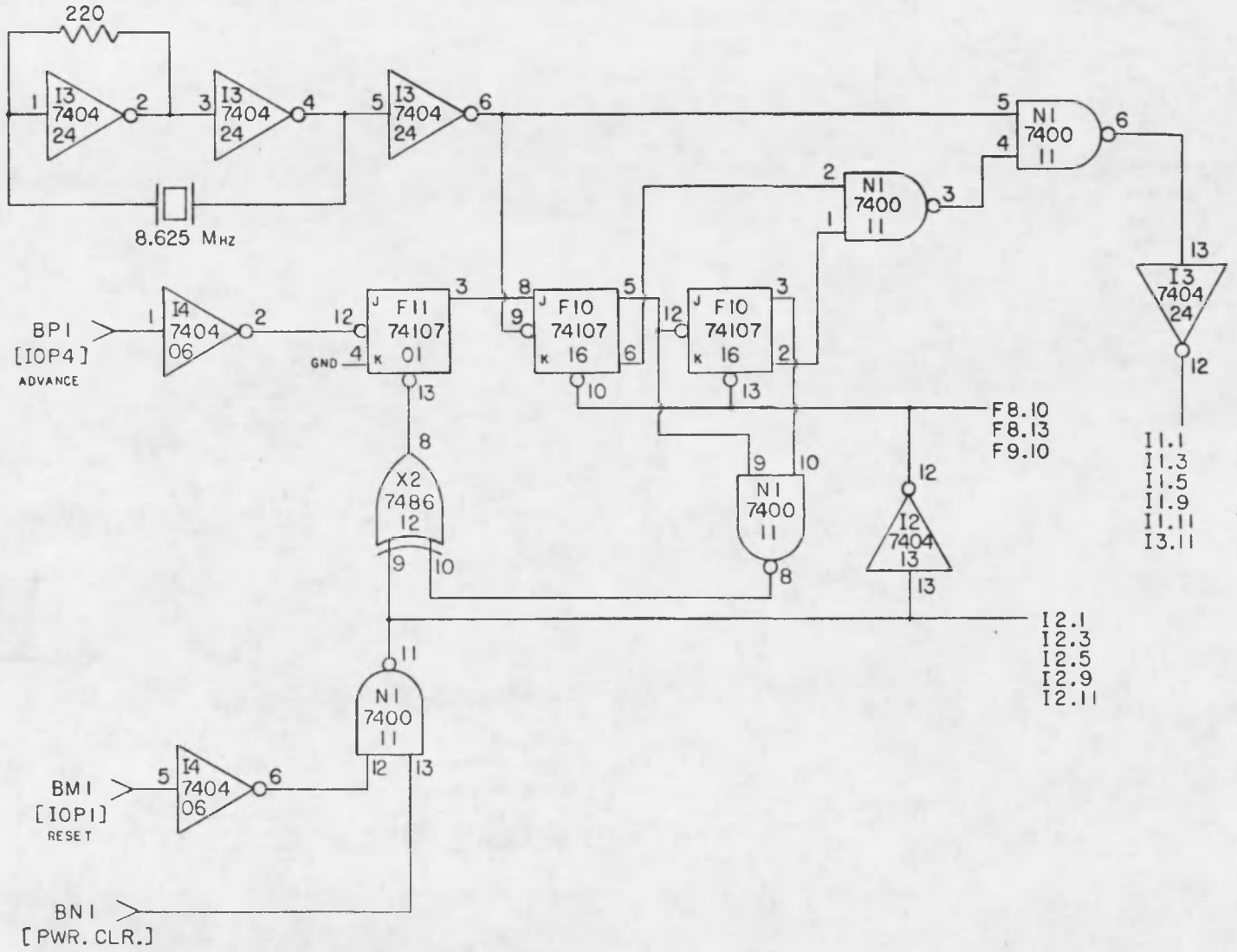
Outside numbers are I.C. pin numbers. (Pin numbers not shown are assumed at a logical "1" level.)

Destination references: additional off-page points to which the signal is connected are listed as device identifier and pin number, e.g.. X1.10.

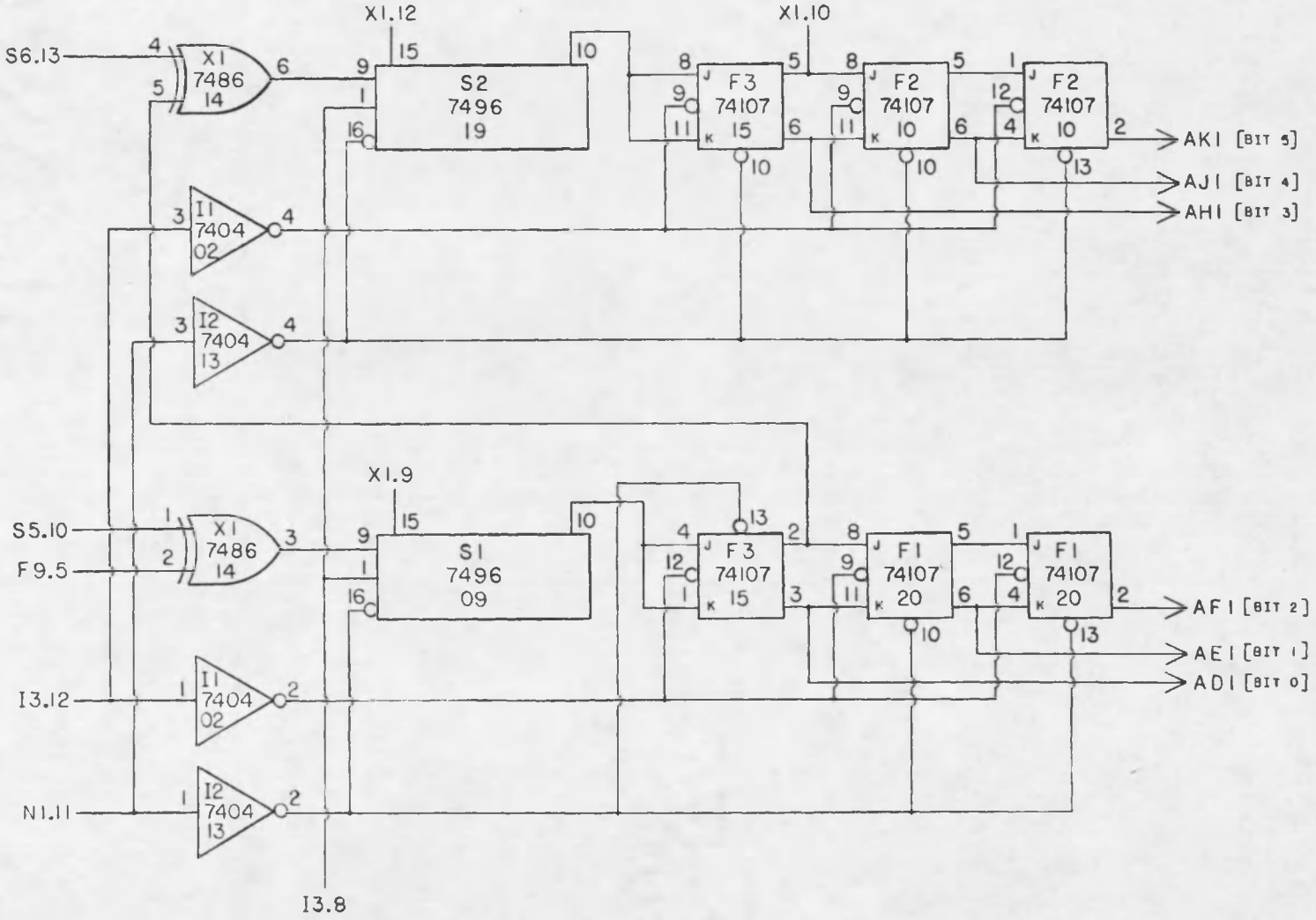
Source references: off-page signal sources are indicated by device identifier and pin number, e.g., N1.11.

Signals which are routed off of the W941 module are indicated by connector symbol (⟶ or ⟶) and module pin identification, e.g., ⟶ AD1.

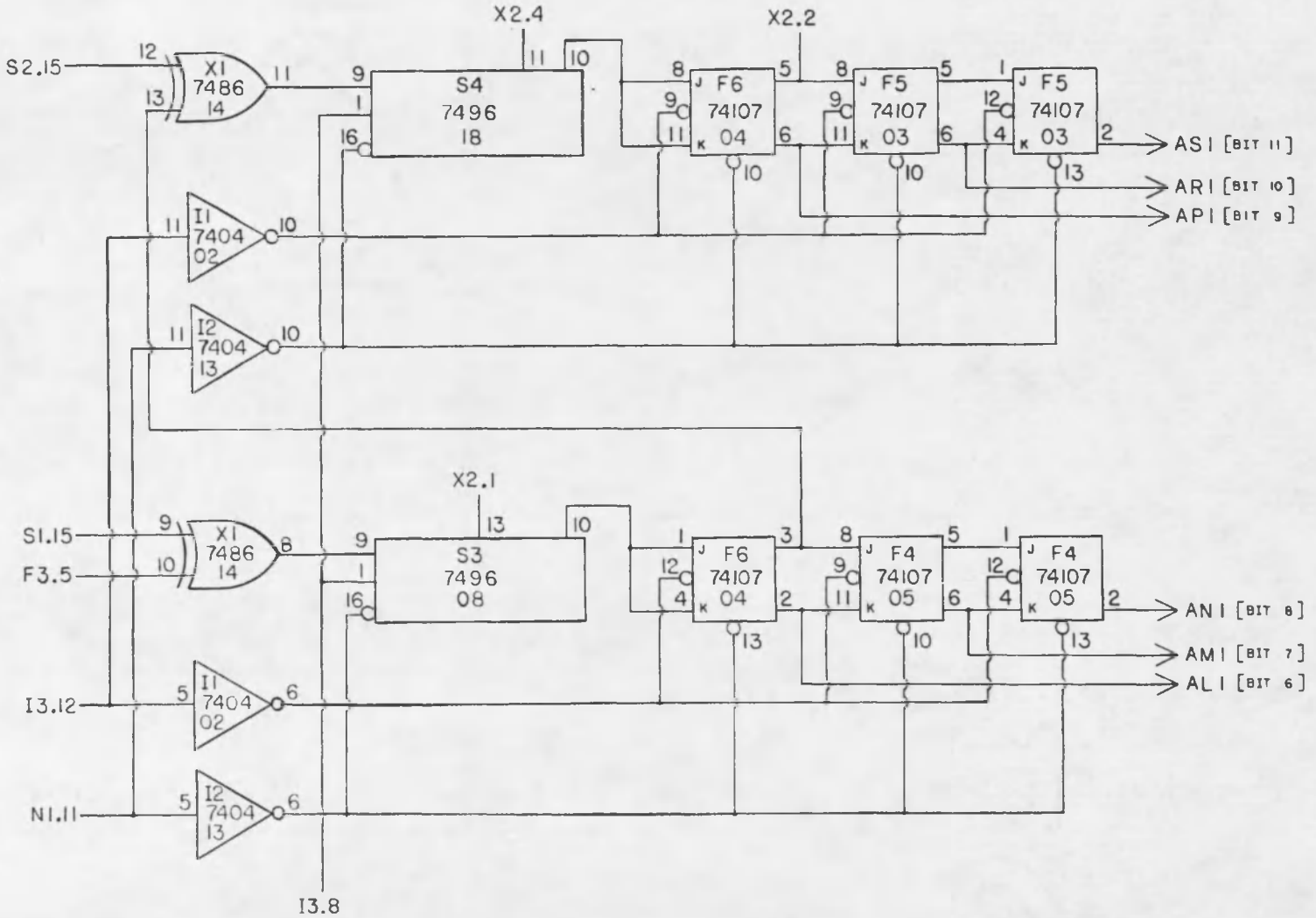
In the diagram of the DEC series M modules the use of connector symbols indicates a connection to the PDP-9 I/O bus cable and the signal name is shown.



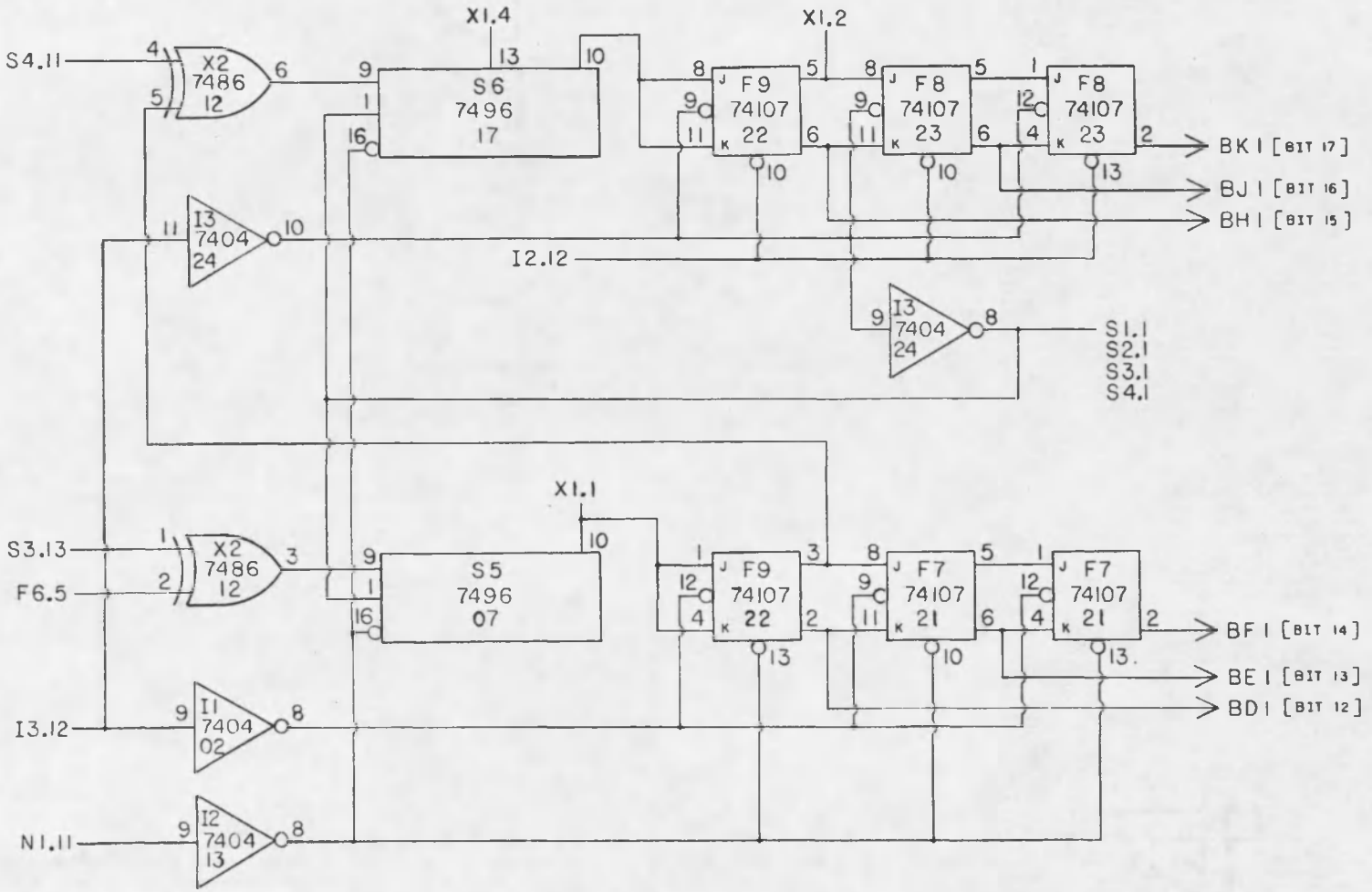
TTL Control Logic - W941



TTL Shift Registers 1 and 2 - W941

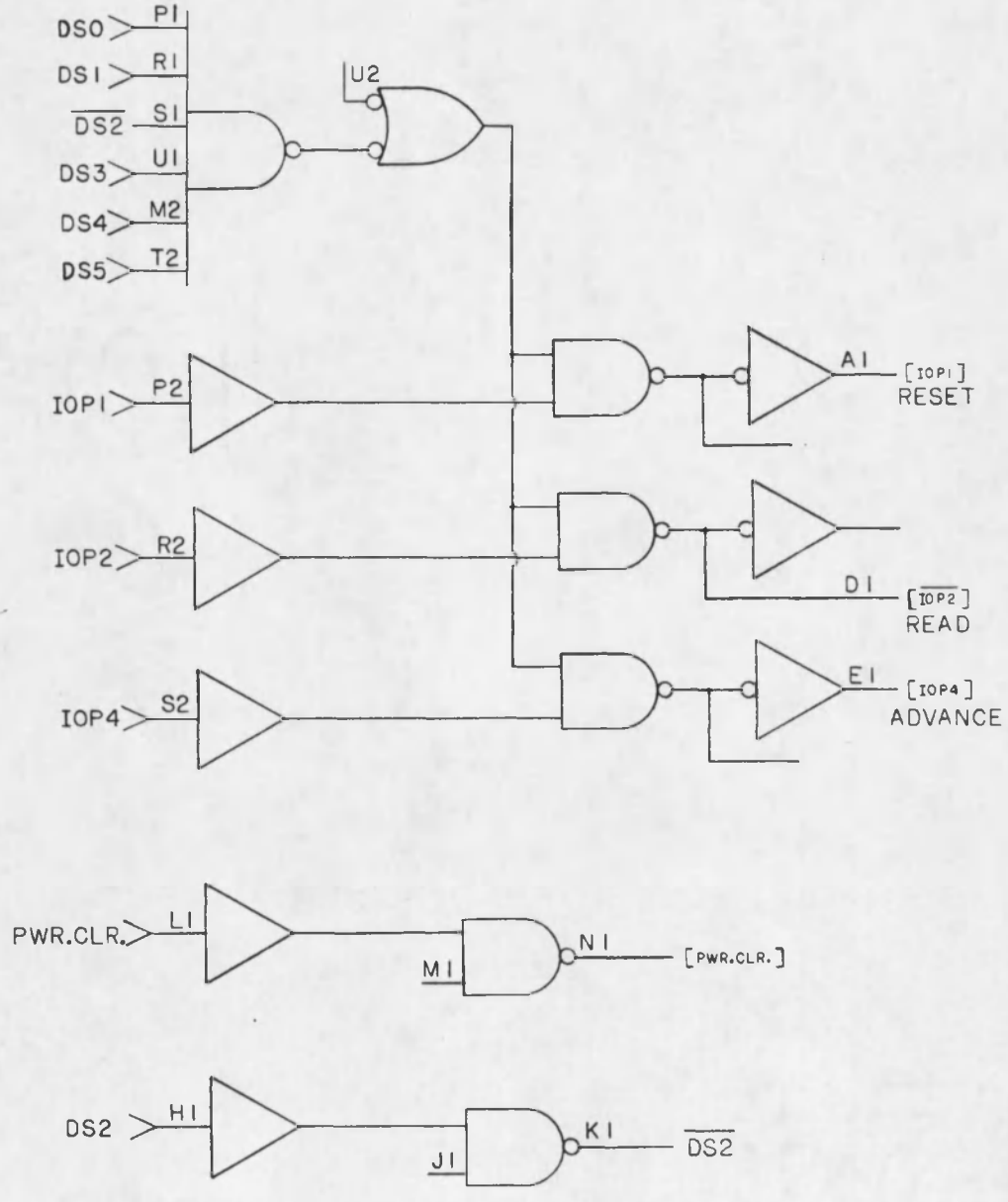


TTL Shift Registers 3 and 4 - W941

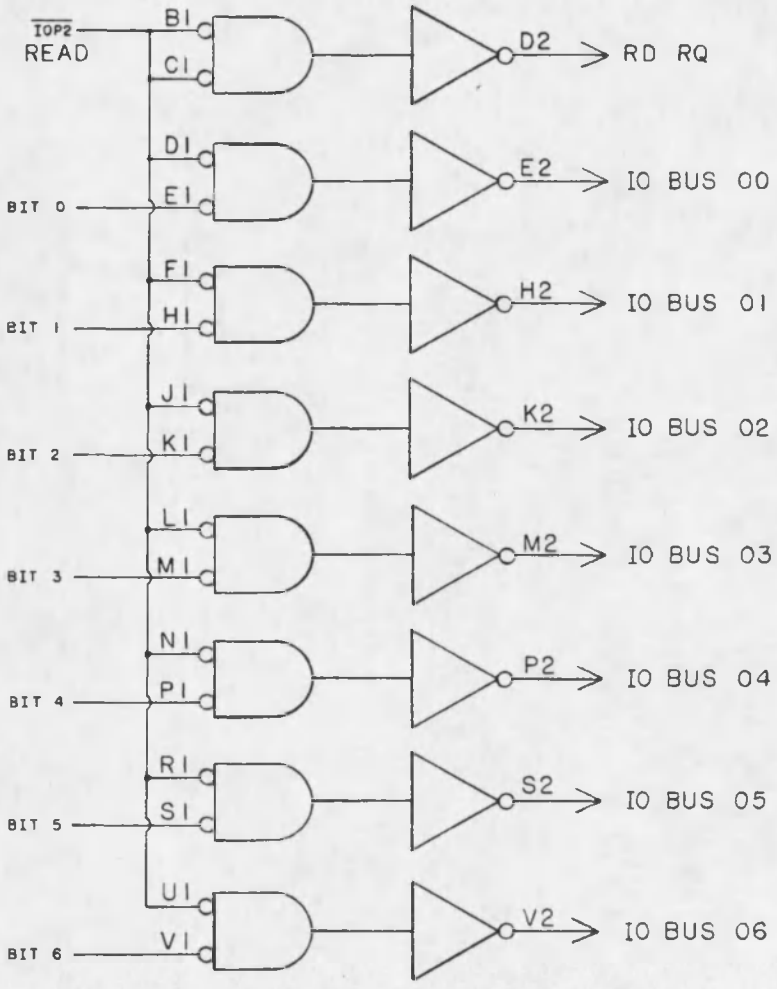


TTL Shift Registers 5 and 6 - W941

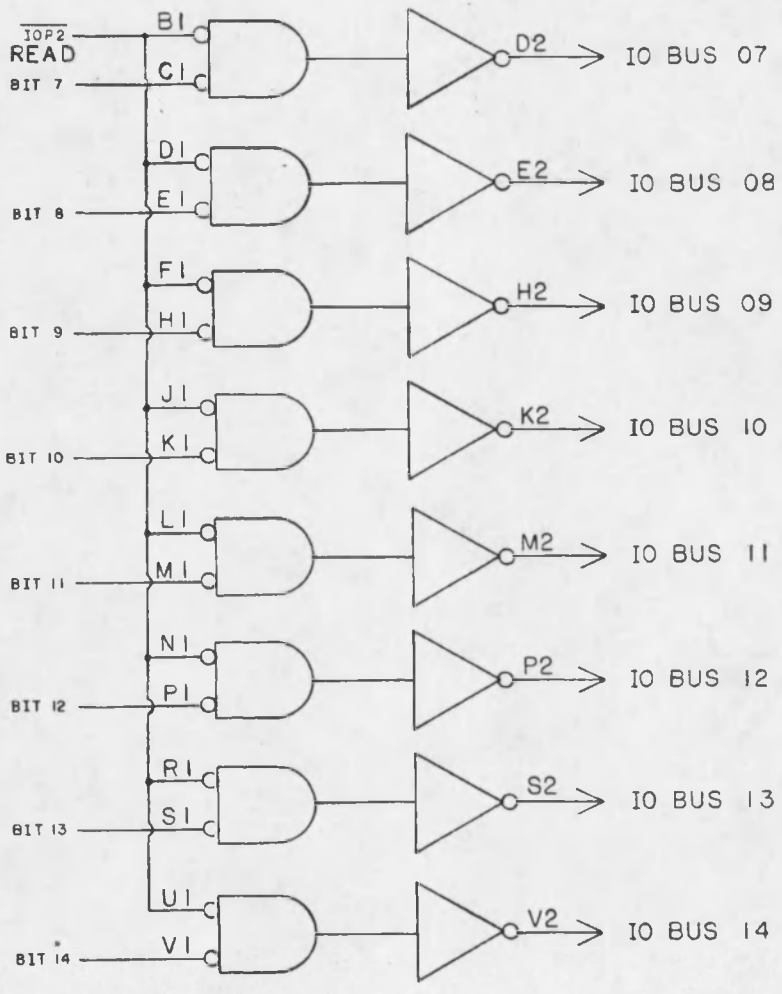
DEVICE CODE 67



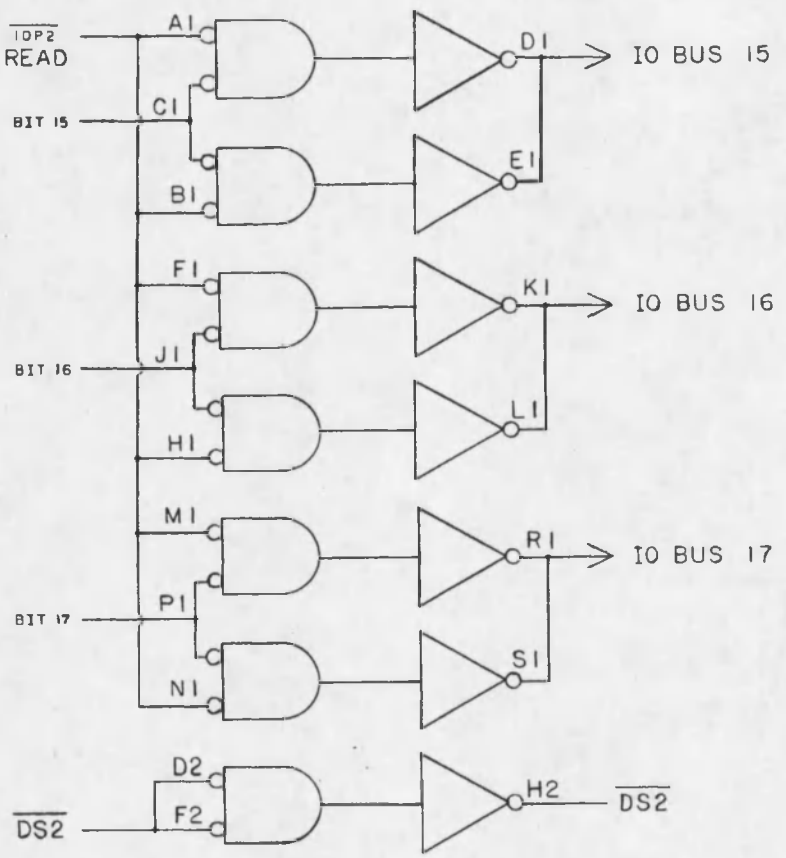
DEC Device Selector - M103



DEC Bus Driver (A) - M632



DEC Bus Driver (B) - M632



DEC Bus Driver - M633

LIST OF REFERENCES

Belt, John Edward, A Random Noise Generator for a Digital Computer, M.S. Thesis, Dept. of Electrical Engineering, University of Arizona, 1969.

Coveyou, R. R., and R. D. MacPherson, Fourier Analysis of Random Number Generators, Journal of the ACM, Vol. 14, No. 1, January 1967.

Digital Equipment Corp., PDP-9, PDP-9/L Interface Manual, Maynard, Mass., 1968.

Digital Equipment Corp., Logic Handbook, Seventh Edition, Maynard, Mass., 1971.

Golomb, Solomon W., Shift Register Sequences, Holden-Day, San Francisco, Calif., 1967.

Hurd, William J., Efficient Generation of Statistically Good Pseudonoise by Linearly Interconnected Shift Registers, Jet Propulsion Laboratories Technical Report 32-1526, Vol. XI, 1972.

Jansson, Birger, Random Number Generators, Victor Pettersons Bokindustri Artiebolag, Stockholm, Sweden, 1966.

Jordan, Harry F., and David C. M. Wood, On the Distribution of Sums of Successive Bits of Shift-Register Sequences, IEEE Transactions on Computers, Vol. C-22, No. 4, April 1973.

Knuth, Donald E., The Art of Computer Programming; Volume 2--Seminumerical Algorithms, Addison-Wesley Publishing Co., Reading, Mass., 1969.

Korn, Granino A., Random-Process Simulation and Measurements, McGraw-Hill Book Co., New York, 1966.

Korn, G. A., and T. M. Korn, Mathematical Handbook for Scientists and Engineers, McGraw-Hill Book Co., New York, 1968.

Lindholm, James H., An Analysis of the Pseudo-Randomness Properties of Subsequences of Long m-Sequences, IEEE Transactions on Information Theory, Vol. IT-14, No. 4, July 1968.

Mann, H. B., and A. Wald, On the Choice of the Number of Class Intervals in the Application of the Chi-Square Test, Annals of Mathematical Statistics, Vol. 13, 1942.

Miller, Irwin, and John E. Freund, Probability and Statistics for Engineers, Prentice-Hall, Englewood Cliffs, N. J., 1965.

RAND Corp., A Million Random Digits with 100,000 Normal Deviates, Free Press, Glencoe, Ill., 1955.

Shreider, Yu. A., Method of Statistical Testing, Monte Carlo Method, Elsevier Publishing Co., Amsterdam, Netherlands, 1964.

Tausworthe, Robert C., Random Numbers Generated by Linear Recurrence Modulo Two, Mathematics of Computation, Vol. 19, No. 90, April 1965.

Texas Instruments Inc., The Integrated Circuits Catalog for Design Engineers, First Edition, Dallas, Texas, 1972.

15

0051 8