**Pseudo-Random Sequences**

A truly random sequence of binary symbols (ones and zeros, for simplicity) would be one for which a knowledge of the complete past history of the sequence would be of no assistance in predicting the next symbol, i.e. the probability that the next symbol would be a one or zero would still be one-half even if the complete sequence of previous output were available. Such a sequence would have to be produced by passing an analog noise source through a comparator sampled at regular clock intervals or some other scheme such as tossing a fair coin. Such a sequence is sometime referred to as digital noise.

![Figure 1: Pseudo-Random Sequence Generator](image)

A pseudo-random sequence is one that appears to be perfectly random for $K$ output symbols but then repeats, i.e. it is periodic with a cycle time of $K$ symbols. If $K$ can be made large enough, this is not a great limitation; any interval up to $K$ symbols will appear to be perfectly random. Such a sequence can be constructed using shift registers and EXCLUSIVE-OR gates. Shown in Fig. 1 is an $N$ stage shift register. Each time a clock pulse is applied the state of the shift register shifts one stage to the right; the state of the final stage $[N]$ is lost and the input to the first stage $[1]$ is obtained by EXCLUSIVE-ORing a set of the stages of the shift register. Typically, stage $N$ provides one of the EXCLUSIVE-OR inputs, while the other input is drawn from stage $M$; other possibilities involve the EXCLUSIVE-OR of the final and a number of lower stages. The output sequence is usually constructed by taking the consecutive values of any one stage; stage $N$ is most commonly used. These circuits or machines are known as feedback shift registers or pseudo noise generators.

If the feedback taps (stages chosen as inputs to the EXCLUSIVE-OR gate) are properly chosen, a maximal length shift register sequence can be obtained. For an $N$ stage shift register, $K = 2^N - 1$ for a maximal length
shift register sequence. Such a circuit goes by a number of names: pseudo-
random sequence generator or maximal-length shift register sequence gener-
ator, pseudo-random bit sequence generator, pseudo-random noise generator,
or a pseudo-noise generator. Not all settings produce a maximal-length se-
quence, viz. they have cycle periods less than $K$. The selection of the proper
taps to produce a maximal-length pseudo-random sequence for an $N$ stage
shift register is an arcane topic in higher mathematics.

Assuming that the taps have been selected to produce a maximal-length
pseudo-random sequence, the state machine or circuit shown in Fig. 1 would
produce an output sequence having one more 1 than zero—all the $N$ bit binary
numbers from 1 to $N$ ones would be present in the $N$ bit shift register for
different clock pulses (all zeros is excluded because it would cause the shift
register to latch-up in the all zero state). If the EXCLUSIVE-OR gate is
followed by an inverter, the complement of the previous output sequence will
be obtained which will have one more zero than one. The state of the shift
register for various clock cycles assumes all of the $N$ bit binary numbers with
the exception of the all zeroes state.

Because $K$ depends exponentially on $N$, this makes it possible to have
very long pseudo-random sequences. For instance, with a 33-bit (stage) reg-
ister clocked at 1 MHz the cycle time would be over 2 hours and with a
100-bit register clocked at 10 MHz the cycle time would be around 27,000
times the estimated age of the universe which is 15 billion years. Thus, even
at relatively high clock rates, it is possible to produce pseudo-random se-
quences that would take centuries to repeat. They can be used as random
binary sequences for any time interval less than the cycle time.

Pseudo-random sequences have numerous applications in electrical and
computer engineering—especially telecommunications. A pseudo-random ana-
log noise source can be obtained by low-pass filtering the output of the
pseudo-random sequence where the cutoff frequency of the filter should be
much less than the clock frequency of the sequence generator. Such a se-
quence may be used to scramble data for security reasons (EXCLUSIVE-
ORing the data stream with the pseudo-random sequence at the trans-
mitter and then reversing the procedure at the receiver with the identical
pseudo-random sequence which is known only to the valid users). Digital
pseudo-random sequences are used extensively in error-correcting and de-
tecting codes. Because they have very peaked autocorrelation properties,
they are widely used radar ranging and GPS systems. The almost random
but repeatable nature of pseudo-random sequences are widely used in spread
spectrum communications systems such as digital cell phones that use CDMA
technology to, among other things, give each digital cell phone a unique code.

Procedure

- Use the supplied Mathcad program to produce sequences for shift reg-
  isters having lengths of 3, 4, 5, and 6. Experimentally determine which
  feedback taps (values of M) result in maximal length pseudo random
  sequences.

- Experimentally determine the largest shift register that can be practi-
  cally analyzed using the laboratory pc which is a Pentium II with a 500
  MHz microprocessor. Do this by measuring the time duration taken
  to compute the period or cycle time of maximal-length sequences. Be
  careful. A large shift register will cause the blinking light bulb icon to
  appear and Mathcad will not stop until it has finished the calculation.
  The Windows Task Manager may be used to halt the program but
  when it is reopened it will initiate the computation with the previously
  selected values of M and N. So a fresh copy of the program should be
  used.

- Place each pseudo-random sequence into the function generator, ob-
  serve the display on the oscilloscope, and print the resulting display.
  This can be done by putting the pointer into the array on the Math-
  cad sheet for \( y(t_j, 1) \) and right clicking the mouse. Then select select
  all, copy selection, minimize Mathcad, open Waveform Editor, paste
  the waveform in, click on the icon to transfer the waveform into the
  function generator, and then ok.

- Listen to the sounds made by several pseudo-random sequences with a
  small speaker supplied by the laboratory instructor. Experiment with
  different frequencies of the function generator.