Unit 25 Shift registers and pseudo-random generators.

- A shift register is a line of single bit memory cells which are connected so that the output of one provides the input to the next in line.
- An alternative name is a bucket brigade.
- A shift register moves the contents of each memory cell one bit to the left or right at each clock pulse.
- A shift register can operate with either a serial or parallel input and either a serial or parallel output.

A single flip flop acts as a memory for a single bit. A shift register is a set of single flip flops which are connected in series so that a clock pulse causes the contents of each flip flop to be shifted either to the left or to the right along the register. An alternative term is a bucket brigade—visualise a line of people passing buckets of water from one to the next in order to bring water to put out a fire.

One form of shift register is the serial in, serial out type of register in which the input stream of data bits is clocked into one end of the shift register and appears at the output end of the register after n clock pulses where n is the number of flip flops in the register. The structure of such a SISO type shift register is shown in Figure 25.1.

The simplest version of the SISO shift register uses D type flip flops connected in series. On the positive going edge of the clock the input is read into the first flip flop and the current value of the Q output of the nth flip flop is read into the $n+1$th flip flop D input.

In the example shown in Figure 25.1, the timing diagram shows a random input being applied to the serial input. At the next positive going edge of the clock this instantaneous value is read into the first stage of the shift register and appears at the output of the 4 stage shift register four clock cycles after this positive going clock edge as indicated by the arrow showing the time displacement. Because the input transitions are not synchronized with the clock edges, the duration of the input pulses may be changed by up to one

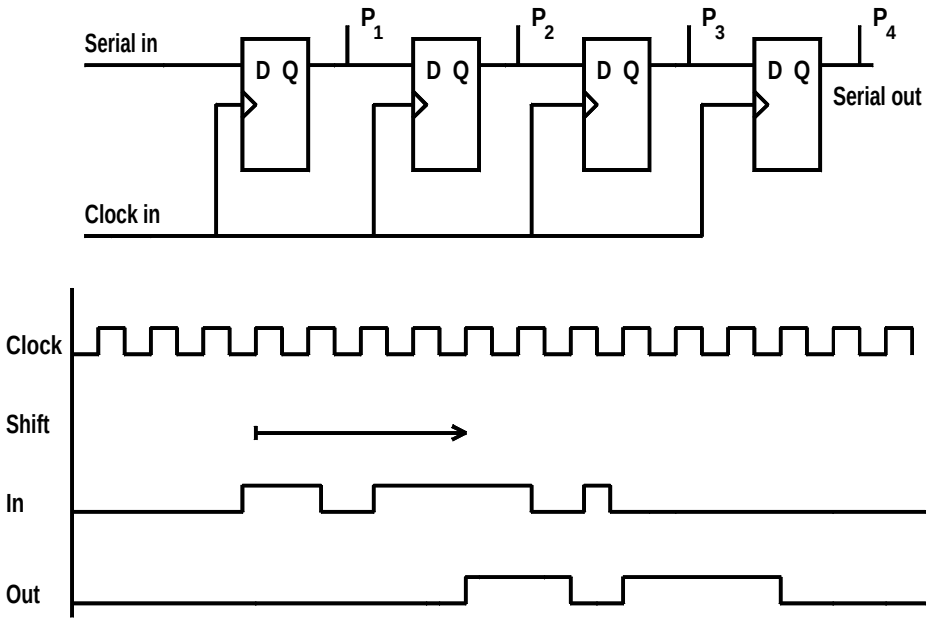

Figure 25.1: SISO shift register and input and output waveforms.

clock period as shown in some of the transitions. For input pulses which are significantly longer than a clock period the difference will be negligible but for short input pulses the pulse may fall completely between two positive transitions of the clock cycle and be missed entirely as is shown in the last input pulse in Figure 25.1.

The second type of shift register is the serial in parallel out type in which the data is inputted sequentially but read out in parallel with all of the bits being read (non-destructively) from all of the flip flops at the same time. The circuit shown in Figure 25.1 has extra outputs, labeled P_1, P_2 etc, where the parallel outputs are available. So, if the outputs are taken from P_1, P_2 etc then the device is being operated as a Serial In Parallel Out (SIPO) device.

The third version of the shift register has facilities for inputting data in parallel to all of the registers at the same time and then reading the data out sequentially. This gives a Parallel In Serial Out (PISO) shift register. Essentially this involves allowing the flip flops to be set to the data values of the parallel input instead of taking the data from the flip flop one step back in the sequence. Provision must then be made for reverting to shift mode for sequential transfer of data so that the data can be read out sequentially. The inputs to such a circuit will therefore be the data signal to each of the flip flops and a control signal to all of the flip flops which will determine whether the data is loaded in parallel or is shifted by one position on the next clock edge. This signal is denoted S/L for shift or load. Figure 25.2 shows the logic

gates which are used to determine the source of the data to a particular flip flop. This circuit pattern is repeated n times to form an n bit shift register.

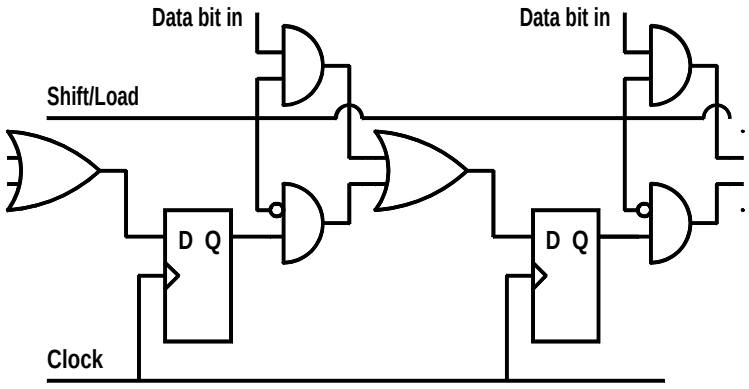

Figure 25.2: Single stage of a PISO shift register.

In this discussion we have distinguished between the three different types of shift register. In practice, the commercial integrated circuit implementations of the shift registers incorporate all of the functions described here into one single chip.

As an example of a typical shift register, the 7495 type TTL 4 bit shift register gives parallel access and shift functions. The parallel loading is accomplished by applying the four bits of data and taking the mode control input high. The data is loaded into the associated flip flops and appears at the outputs after the high-to-low transition of the clock 2 input. During loading the entry of serial data is inhibited.

Shift right is accomplished on the high-to-low transition of clock 1 when the mode control is low; shift left is accomplished on the high-to-low transition of clock 2 when the mode control is high by connecting the output of each flip flop to the parallel input to the previous flip flop (Q_D to input C etc) and serial data is entered at input D. The clock input may be applied commonly to clock 1 and clock 2 if both modes can be clocked from the same source. Changes at the mode control input should normally be made while both clock inputs are low; however, conditions described in the last three lines of the function table will also ensure that register contents are protected.

Inputs								Outputs			
Mode	Clocks		Serial	Parallel				Q_A	Q_B	Q_C	Q_D
Control	2(L)	1(R)		A	B	C	D				
H	H	X	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}
H	↓	X	X	a	b	c	d	a	b	c	d
H	↓	X	X	Q_B	Q_C	Q_D	a	Q_{Bn}	Q_{Cn}	Q_{Dn}	a
L	L	H	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}
L	X	↓	H	X	X	X	X	H	Q_{An}	Q_{Bn}	Q_{Cn}
L	X	↓	L	X	X	X	X	H	Q_{An}	Q_{Bn}	Q_{Cn}
↑	L	L	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}
↓	L	L	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}
↓	L	H	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}
↑	H	L	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}
↑	H	H	X	X	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}

Function table for the 7495 4 bit shift register

There are a number of important applications of shift registers and we will briefly discuss a few applications. In a computer data bytes are stored as set of 8 bits forming a byte. When information is to be transferred to another computer over a serial medium such as a phone line the bytes are read out from the computer in parallel into a register, parity check bits are added and the register contents are then read out in serial form at the appropriate data rate for the medium which is determined by the clock rate. At the other end of the communications line the incoming bit stream is read in serially into the shift register at the prearranged data rate or baud rate and when all the bits have been clocked in the parity is checked and the data then transferred in parallel onto the computer data bus. This serial transfer of data through a serial computer port is so common that specialist serial interface chips are available for all of the computer systems which will manage the transfer and interface correctly with the computer control signals. However, the core function of all of these serial ports is a shift register.

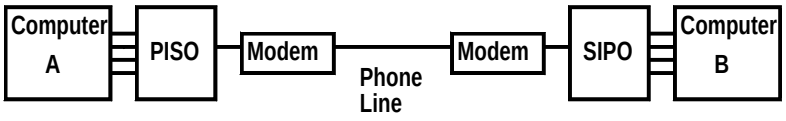

Figure 25.3: Block diagram for serial data transfer between computers.

Another application of shift registers is to generate a pseudo random digital noise signal. A true random digital noise signal will change from 0 to

1 and back to 0 at random times. The first restriction is to require that the transitions only occur at the positive edge of a regular clock. These random changes will then have certain statistical properties when sampled. The bit stream of 0's and 1's will be as random as the stream of Heads and Tails from successive tosses of a coin. It is possible to generate a stream of 0's and 1's which have the same statistical properties as a true random sequence but which repeats exactly after some give number of cycles. This means that information can be broadcast on an open communication channel, which can be monitored by an eavesdropper, and this information can be hidden so as to appear to be random noise. However, if the modulation sequence or key is known, the signal can be retrieved from this pseudo random noise by the legitimate recipient of the signal.

Now let us consider how a pseudo random noise or bit stream signal can be generated.

If we take a shift register of n bits, as shown in Figure 25.4, and feed the output signal back to the input, the sequence of bits appearing at the output rotates and will repeat exactly after n clock pulses. There are 2^n possible different sequences depending on the initial values stored in the shift register but each of these sequences is only n bits long.

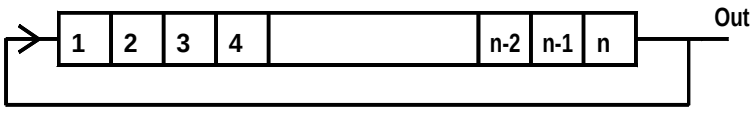

Figure 25.4: Feedback shift register.

One method used to generate pseudo random signals having a greater degree of randomness and longer periodicity is to use a tapped shift register, shown in Figure 25.4, in which an intermediate signal is tapped off from the shift register, Exclusive-ORed with the output and fed back to the input.

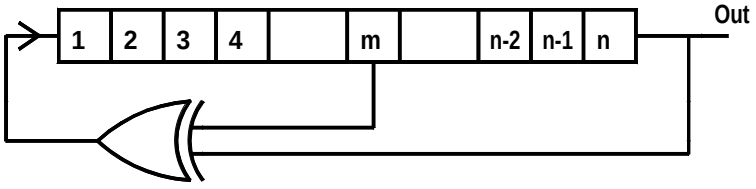

Figure 25.5: Pseudo Random Noise generator.

The shift register is m bits long and the tapping point is at bit position n. The introduction of this feedback tab now causes the system to have

up to $2^n - 1$ bits in the output sequence before the stream repeats again. This is called a Maximal Length stream. In simplest terms, it is necessary that n and m have no common factors if a maximal length sequence is to be obtained. But the essential point is that the introduction of the feedback tap increases the number of cycles before repetition occurs from n clock cycles to $2^n - 1$ clock cycles.

If a data stream is to be transmitted with some degree of confidentiality it can be Exclusive ORed with the output from a pseudo random generator which is preloaded with a prearranged code sequence. The transmitted data is then secure and there is no slowing of the transmission rate as would be required if the data is encrypted. The only delay is in the initial setting up of the connection when the preload value for the pseudo noise generator is exchanged using a full encryption system.

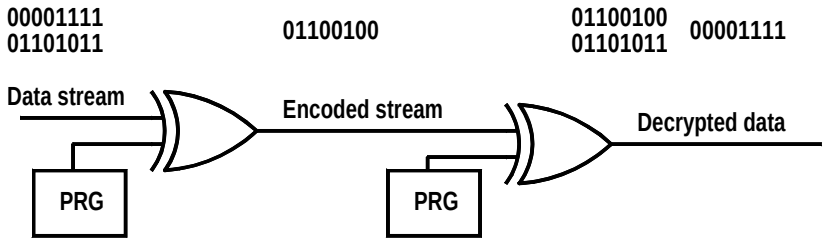

Figure 25.6: Use of Pseudo Random Noise generator for encryption.

This is shown in Figure 25.6 in which a data stream of 00001111 is eXclusive ORed with the stream from a pseudo random noise generator. The encrypted result is 01100100 which is transmitted over a phone line or radio link or email. The encrypted message is decoded at the receiving end by using another synchronised pseudo random noise generator which has been loaded with the same settings. When the encrypted message is eXclusive ORed with the bit stream from the pseudo random noise generator it is found that the original data, 00001111, is recovered.

In order to have a secure system it is necessary to use strong encryption when exchanging the settings for the pseudo random generators. It is also necessary to use different settings for each message. It is also good practice to use the full available bit length for encryption. In the early days of GSM phones it was discovered that only 30 bit length encryption keys were used even though 56 bits were available. The leading 16 bits were set to 0 by default. This meant that it was possible to discover the encryption key after about a minute and eavesdrop on GSM phone conversations by using a computer and a sample which is tested with all possible tap settings until intelligible conversation is obtained. It has not been determined whether

this security lapse was was due to stupidity or due to a deliberate attempt to weaken the system so that the government security agencies could listen in to phone conversations. Also it should be remembered that the encryption only applies to the radio link between the mobile user and the base station where the signal is decrypted. For government agencies which have covert access to the computers controlling the phone system it is a simple operation to branch all calls from a specific phone to a recording system in a distant location or to arrange that specific phones are supplied with prearranged nonrandom security settings. In any case, the most interesting surveillance information is probably having records of which phone called which phone at what time and for how long. The location of a mobile phone is also recorded at all times even when it is not being used to make a call so if your GSM phone is turned on, your travels around the city or country are known and recorded and the records are readily available.

Another application of pseudo random noise generators is when data is transmitted from a number of sources on the same radio frequency. This occurs when the orbital data is transmitted from Global Positioning System satellites. Each satellite uses a different pseudo random generator and the signals can be pulled out of noise by using the identification code for that particular satellite. There are other techniques such as spread spectrum transmissions used but the net effect is that a useful signal can be abstracted even when it is buried in 20dB of noise.

The reader is referred to the references for a fuller discussion of these applications.

25.1 References

Redl, S H
GSM and Personal Communications Handbook
Artech House 1998

Tsui James Bao-yen
Fundamentals of Global Positioning Systems Receivers
Wiley, 2000

Parkinson and Spilker
Global positioning System: Theory and Applications
Progress in Astronautics and Aeronautics, Volume 164

25.2 Problems

25.1 For the feedback loop similar to Figure 25.4 having $n = 4$ and with an initial value of 1001 stored in the registers, enumerate the successive contents of the registers and give the output bit stream values.

25.2 For the feedback loop similar to Figure 25.5 having $n = 4$ and $m = 3$ and with an initial value of 1001 stored in the registers, enumerate the successive contents of the registers and give the output bit stream.