

Convolution Codes

♣ **The Origin and Design of Cyclic Codes.** One reason that *cyclic codes* are so useful is that they can be efficiently encoded by means of *shift registers*. There are also decoding schemes utilizing shift registers. Many important codes such as the Golay codes, the Hamming codes, and *BCH* codes can be represented as cyclic codes. Furthermore, much is known theoretically about cyclic codes, which enhances their practical applications.

We start with a device called a *linear shift register*. This is commonly used to encode cyclic codes. As we shall see, the encoding process is efficient because no storage is required as the codewords are generated by shifting and adding. These are two basic building blocks for shift registers. One is the storage element, which can be a *flip-flop*, in which a field element is stored, and which has one input and one output (Figure 1). The arrows indicate the input and output. The other building block is the *binary adder*, which has two inputs and one output (Figure 2), which is the binary sum of the inputs.



We give an example of a shift register, Figure 3, with four storage elements and two binary adders.

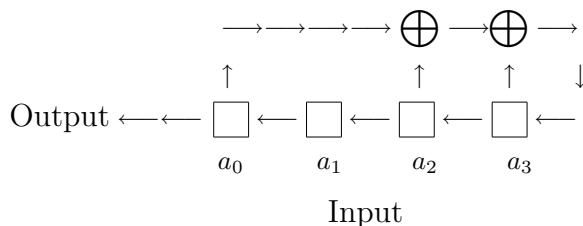


Figure 3

At time zero, four binary elements are placed in $a_0, a_1, a_2,$ and a_3 . After one time interval a_0 is output, a_1 is shifted into a_0 , a_2 into a_1 , and a_3 into a_2 ; and the new element is entered into a_3 . In our example this element is the sum $a_0 + a_2 + a_3$ corresponding to the polynomial $g(x) = 1 + x^2 + x^3 \text{ mod } (1 + x^7)$. We suppose that the digits 1101 are placed in $a_0, a_1, a_2,$ and a_3 and follow the outputs and inputs for seven time intervals.

Outputs	a_0	a_1	a_2	a_3	Time
	1	1	0	1	t_0
	1	1	0	1 0	t_1
	1	1	0	1 0 0	t_2
	1	1	0	1 0 0 0	t_3
	1	1	0	1 0 0 0 1	t_4
	1	1	0	1 0 0 0 1 1	t_5
	1	1	0	1 0 0 0 1 1 0	t_6
	1	1	0	1 0 0 0 1 1 0 1	t_7

If this process is continued, the vector (1, 1, 0, 1, 0, 0, 0) will be repeated. This shift register will repeat any vector of length 7 that it has generated from four initial entries $a_0, a_1, a_2,$

and a_3 . This code has the property that whenever $(a_0, a_1, a_2, a_3, a_4, a_5, a_6)$ is a codeword so is $(a_6, a_0, a_1, a_2, a_3, a_4, a_5)$, that is, whenever a word is in the code so are all of its cyclic shifts.

Consider now the 4-stage shift register in Figure 4.

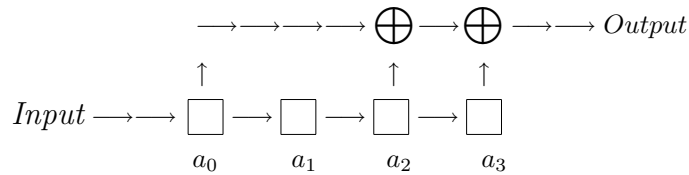


Figure4

Assume the contents of the registers is initially $(0, 0, 0, 0)$ and the input stream:

$$a_0, a_1, a_2, \dots a_6 \quad \text{is} \quad 1010000.$$

The contents of the registers and outputs are summarized in the following table.

$g(x) = 1 + x^2 + x^3$			
time	input	$a_0 a_1 a_2 a_3$	ouput
-1	-	0 0 0 0	-
0	1	1 0 0 0	1
1	0	0 1 0 0	1
2	1	1 0 1 0	1
3	0	0 1 0 1	1
4	0	0 0 1 0	1
5	0	1 0 0 1	1
6	0	1 0 0 1	1

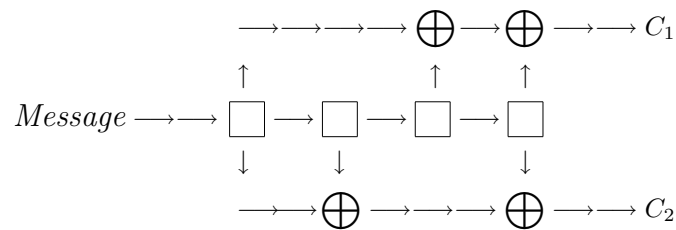


Figure5