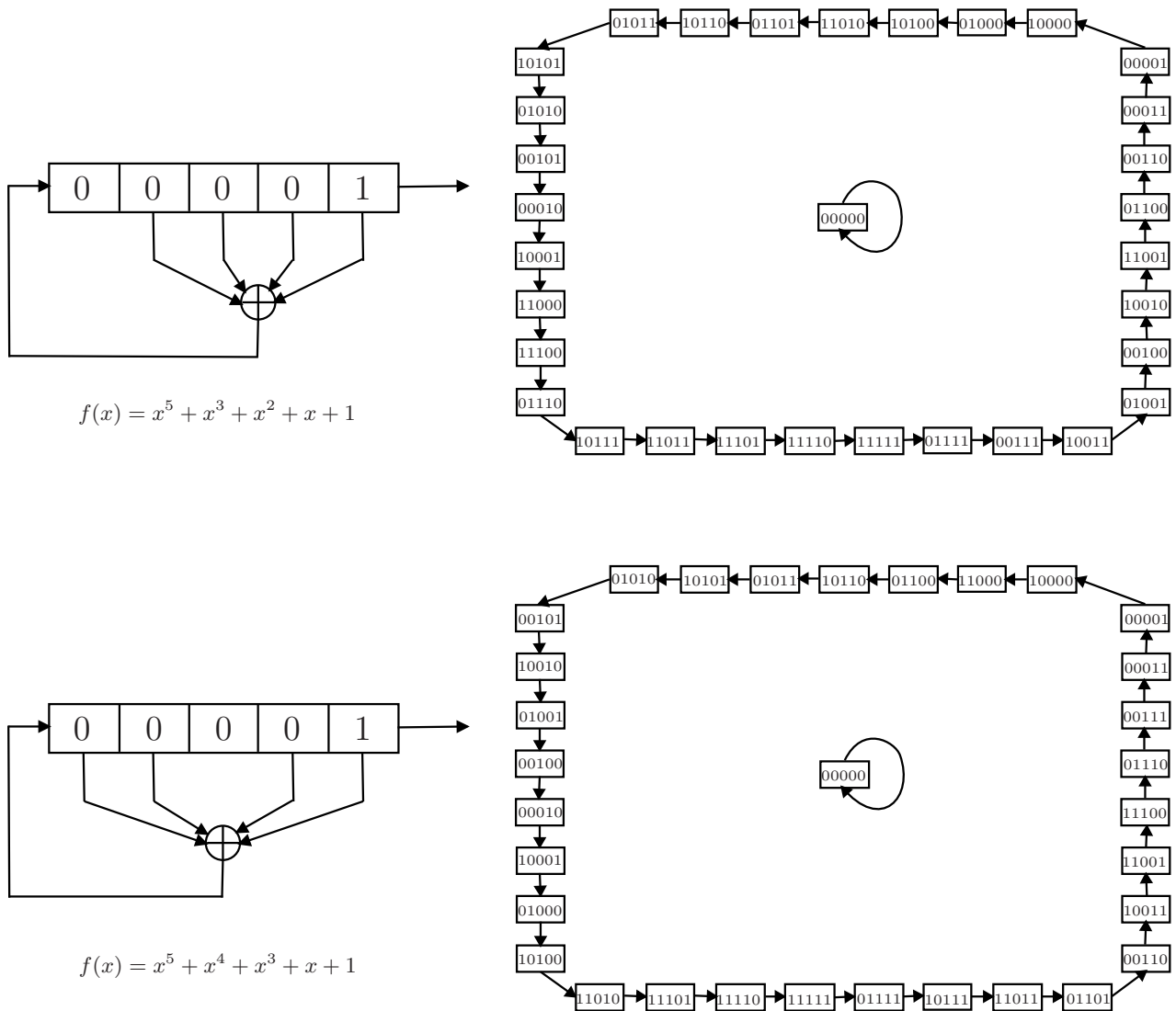


## Solutions to Assignment 1

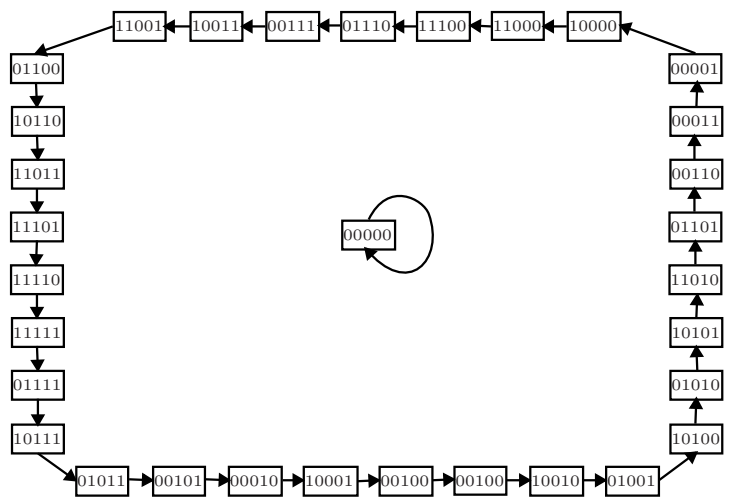
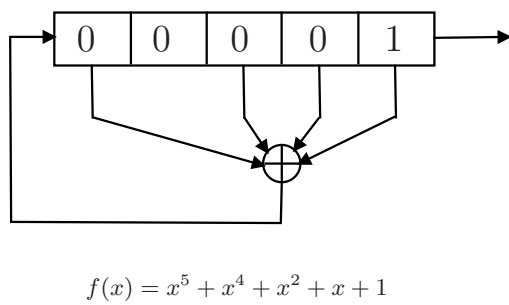
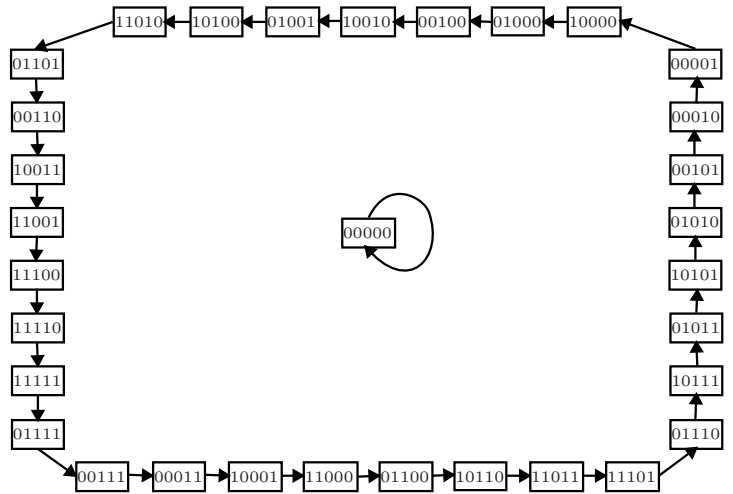
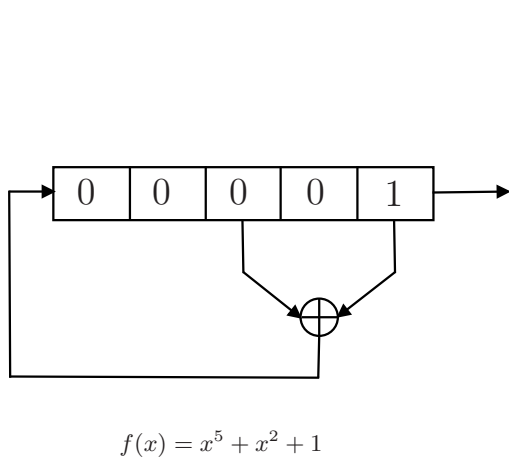
Xinxin Fan

### Part 1 Solutions

- (a) According to 6 primitive polynomials, we can obtain 6 LFSRs of degree 5 and the corresponding state transition diagrams (see Figure 1). Starting from any non-zero initial state, we can generate an  $m$ -sequence of period 31. (Here we list all of them. However, you only need to give one without the state transition diagram.)



The output sequences with an initial state  $(a_0, a_1, a_2, a_3, a_4) = (1, 0, 0, 0, 0)$  for 6 LFSRs are listed in Table 1.



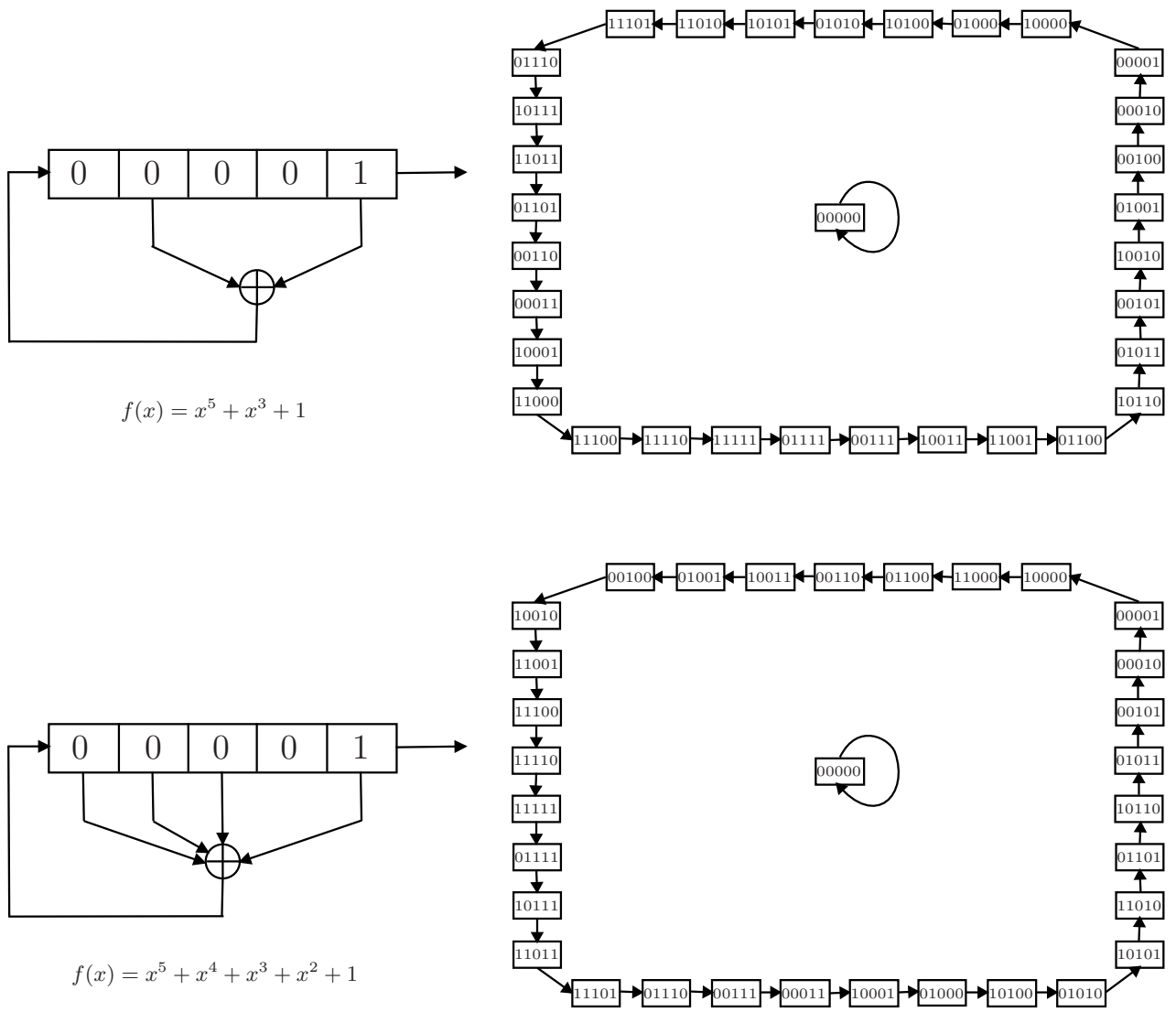


Figure 1: The 5-stage LFSRs and Their State Transition Diagrams

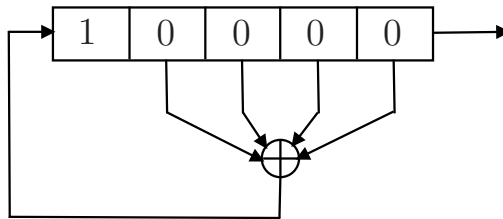
Table 1: Output Sequences with an initial state 10000 for 6 LFSRs

$f(x)$	Initial State	Output Sequence
$x^5 + x^3 + x^2 + x + 1$	10000	1000010110101000111011111001001
$x^5 + x^4 + x^3 + x + 1$		1000011010100100010111110110011
$x^5 + x^2 + 1$		1000010010110011111000110111010
$x^5 + x^4 + x^2 + x + 1$		1000011100110111110100010010101
$x^5 + x^3 + 1$		1000010101110110001111100110100
$x^5 + x^4 + x^3 + x^2 + 1$		1000011001001111101110001010110

(b) From the given sequence **a**, we obtain the following randomness measurements:

- 0 distribution = 15/31,
- 1 distribution = 16/31,
- runs of 0's of length 1 = 4,
- runs of 0's of length 2 = 2,
- runs of 0's of length 3 = 1,
- runs of 0's of length 4 = 1,
- runs of 1's of length 1 = 4,
- runs of 1's of length 2 = 2,
- runs of 1's of length 3 = 1,
- runs of 1's of length 4 = 0,
- runs of 1's of length 5 = 1.

The initial state of the LFSR which generates the sequence **a** is 10000 and the implementation is shown in the following Figure 2.



$$f(x) = x^5 + x^3 + x^2 + x + 1$$

Figure 2: The 5-Stage LFSR Generating the Sequence **a**

(c) Note that the  $m$ -sequence generated with the LFSR in (b) has the period 31. If you start from any non-zero initial state, you can obtain 31 different pseudorandom numbers of 5 bits. If you start from all-zero state, you get the number 00000. Therefore, the LFSR can totally generate 32 different numbers.

2. (a) Bob can generate a DSS signature of the message  $m$  as follows:

- Randomly pick a state generated by PRSG:  $k = 10, 0 < k < 32$ .
- Compute  $r = (g^k \bmod p) \bmod q = (2^{10} \bmod 47) \bmod 23 = 37 \bmod 23 = 14$ .
- Compute  $h(m) = 2m \bmod p = 2 \times 101 \bmod 47 = 14$ .
- Solve for  $s$  in the equation:  $h(m) \equiv xr + ks \pmod q \Rightarrow s = k^{-1}(h(m) - xr) \pmod q = 10^{-1}(14 - 5 \times 14) \pmod{23} = 22$ .

Therefore, the pair  $(14, 22) = (01110, 10110)$  is a DSS signature of the message  $m = 101 = 1100101$ .

(b) If the random number  $k$  used in the DSS signature is compromised, then an attacker can recover the signer's private key when he intercepted the corresponding message  $m$  and signature  $(r, s)$ . More specifically, the attacker can find the signer's private key by solving the following linear congruence equation:

$$x \equiv r^{-1}(h(m) - ks) \pmod q.$$

(c) Since the signature verification requires the computation of  $s^{-1} \pmod q$ . If  $s = 0$ , then  $s^{-1}$  does not exist. To avoid this situation, the signer needs to check that  $s \neq 0$ . Another reason is that if  $s = 0$ , an attacker can obtain the signer's private key  $x$  by computing  $x \equiv r^{-1}h(m) \pmod q$ .

(d) If the hash function is not secure in *DSS*, then an attacker can easily find another message  $m'$  such that  $h(m) = h(m')$ . Therefore, the attack can replace the message  $m$  by  $m'$  and the resulting message and signature tuple  $\langle m', (r, s) \rangle$  will also pass the verification of the verifier.

## Part 2 Solutions

1. The encryption and decryption procedure of DES is shown in Figure 3.

The decryption algorithm of DES consists of the encryption algorithm with the same key but reversed key schedule. More specifically (see Figure 3), the effect of  $IP^{-1}$  is cancelled by  $IP$  in decryption, leaving  $(R_{16}, L_{16})$ . If we consider applying the first round encryption to this input, then the operation on the left half yields  $R_{16} \oplus f(L_{16}, K_{16})$  which is equal to  $L_{15} \oplus f(R_{15}, K_{16}) \oplus f(R_{15}, K_{16}) = L_{15}$  (Note that  $L_{16} = R_{15}$  and  $R_{16} = L_{15} \oplus f(R_{15}, K_{16})$ ). Therefore, the round 1 decryption yields  $(R_{15}, L_{15})$ . Similarly, the remaining 15 rounds are likewise cancelled one by one in reverse order, due to the reversed key schedule.

2. We first divide the plain text  $m = 1100010111110011$  into four blocks  $m_1 = 1100, m_2 = 0101, m_3 = 1111$  and  $m_4 = 0011$ . Then we can find the corresponding cipher text as follows:

$$c_1 = E(0011 \oplus 1100) = E(1111) = 1000,$$

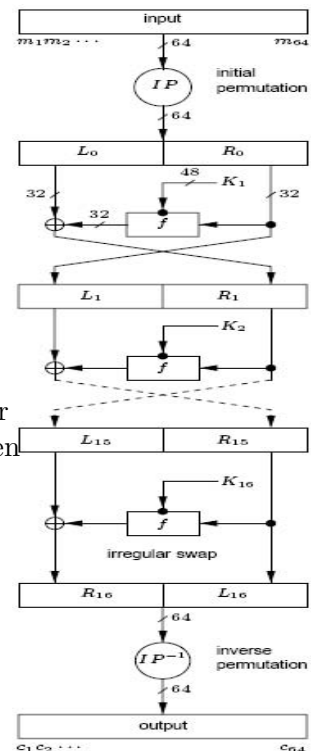


Figure 3: DES Encryption and Decryption Algorithm

$$\begin{aligned} c_2 &= E(0011 \oplus 0101) = E(0110) = 0111, \\ c_3 &= E(0011 \oplus 1111) = E(1100) = 1010, \\ c_4 &= E(0011 \oplus 0011) = E(0000) = 0000. \end{aligned}$$

Therefore, we get the cipher text which is 1000011110100000.

3. Alice's public key:  $Pk_A = g^{x_A} \pmod{p} = 5^3 \pmod{47} = 31$ .  
 Bob's public key:  $Pk_B = g^{x_B} \pmod{p} = 5^7 \pmod{47} = 11$ .  
 For performing the Diffie-Hellman key agreement, Alice and Bob will do the following computations:

- Alice computes:  $k_{AB} = Pk_B^{x_A} = 11^3 \pmod{47} = 15$ .
- Bob computes:  $k_{AB} = Pk_A^{x_B} = 31^7 \pmod{47} = 15$ .

Therefore, Alice and Bob obtained the shared key  $k_{AB} = 15$  as a result.

4. Bob can generate a digital signature using the RSA scheme as follows:
  - Bob creates his public key and private key pair, for example,  $(e, d) = (3, 147)$ .
  - Bob computes  $h(m) = 2m \pmod{n} = 2 \times 2 \pmod{253} = 4$ .
  - Bob computes  $r = h(m)^d \pmod{n} = 4^{147} \pmod{253} = 49$ , which is Bob's RSA signature on the message  $m$ .
5. (a) After Bob submits his public key to CA, the CA will generate a certificate for Bob which binds between Bob's public key and his identity. Let  $Pk_{Bob}$  be Bob's public key,  $ID_{Bob}$  be Bob's identity,  $T_{Bob}$  be the expiration date of Bob's certificate,  $S$  be the scope (enc or sign). Then Bob's certificate has the following format:

$$\text{certificate} = \langle Pk_{Bob}, ID_{Bob}, T_{Bob}, S, \text{RSASign}_{CA}(Pk_{Bob}, ID_{Bob}, T_{Bob}, S) \rangle.$$

(b) Alice should first obtain Bob's certificate and check whether it is valid by verifying the CA's signature. Furthermore, Alice also needs to communicate with the CA to ensure that Bob's certificate has not been revoked before the expiration date.

(c) A certificate authority (CA), which issues digital certificates for use by other parties, is the trust root of a public-key system. Users establish the trust relationship among themselves through verifying certificates issued by a CA.