S-DES DECRYPTION SAMPLE

The ciphertext output produced by the encryption sample was 01000110. We therefore adopt that as input here. As a check, we expect the output that will emerge here to produce the encryption sample's original plaintext input, which was 01101101.

As input to this decryption stage, use ciphertext obtained from having applied S-DES to some plaintext. To the input (ciphertext), apply initial permutation IP: IP 2 | 6 | 3 | 1 | 4 | 8 | 5 | 7 In the next steps, we will develop 4 bits with which to replace the left half of this "blue" result.	Input: 0 1 0 0 0 1 1 0 1 1 0 0 0 0 0 1
To right 4 bits of above result, apply expansion/permutation E/P (generating 8 bits from 4). The bit numbering is that of the 4-bit right-nibble, not of the 8-bit byte (e.g., indicated bit 2 refers to byte's bit 6). E/P 4 | 1 | 2 | 3 | 2 | 3 | 4 | 1	1 0 0 0 0 0 1 0
Upon above result, perform binary XOR operation with subkey K2 (when encrypting we applied K1 first then K2; decrypting now, we reverse that sequence): K2 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1	1 1 0 0 0 0 0 1
Determine a row and a column from above XOR result. For the row, combine bits 1 and 4 and convert to decimal. For the column, combine bits 2 and 3 and convert to decimal. Determine another row and column. For this second row, combine bits 5 and 8; for this second column, bits 6 and 7.	

Identify the entry in s-box S0 at the first row/first column you determined. S0 shows it in decimal; convert it to binary (two bits). Enter those bits as the first half of the 4-bit number at right. Identify the entry in s-box S1 at the second row/second column you determined. Convert it to binary; enter those two bits as the second half of the number at right.

S0 =		c0	c1	c2	c3
	r0	1	0	3	2
	r1	3	2	1	0
	r2	0	2	1	3
	r3	3	1	3	2

S1 =		c0	c1	c2	c3
	r0	0	1	2	3
	r1	2	0	1	3
	r2	3	0	1	0
	r3	2	1	0	3

left nibble:
bits 1 & 4 -> 10 -> 2
bits 2 & 3 -> 10 -> 2
therefore, get from S0 row 2 col 2
result is 1 -> 01

right nibble:
bits 1 & 4 -> 01 -> 1
bits 2 & 3 -> 00 -> 0
therefore, get from S1 row 1 col 0
result is 2 -> 10

0 1 1 0

To above result, apply permutation P4:

P4			
2	4	3	1

1 0 1 0

Upon the above P4 result, perform binary XOR operation, combining it with the left 4-bits of our first result (application of IP to original ciphertext input, blue cell above).

We are trying to replace the left half of that first result. These XOR result bits are the replacement bits for it.

XOR with 1100

0 1 1 0

Rewrite that "blue" first result with its left half replaced. (Look it up, keep/copy its right half, use the preceding result as the new left half.)	0 1 1 0 0 0 0 1
Swap the two 4-bit halves of the above (previous) result. In the next steps, we will again develop 4 replacement bits, and with them replace the left half of this "green" swap result. The steps will be the same ones used for that purpose already.	0 0 0 1 0 1 1 0
To right 4 bits of above swap result, apply expansion/permutation E/P (generating 8 bits from 4): E/P 4 1 2 3 2 3 4 1	0 0 1 1 1 1 0 0
Upon above result, perform binary XOR operation with subkey K1: K1 1 0 1 0 0 1 0 0	1 0 0 1 1 0 0 0
Determine a row and a column from above result. For the row, combine bits 1 and 4 and convert to decimal. For the column, combine bits 2 and 3 and convert to decimal. Determine another row and column. For this second row, combine bits 5 and 8; for this second column, bits 6 and 7. Identify the entry in s-box S0 at the first row/first column you determined. It's given in decimal; convert it to binary	left nibble: bits 1 & 4 -> 11 -> 3 bits 2 & 3 -> 00 -> 0 therefore, get from S0 row 3 col 0

(two bits). Enter those bits as the first half of the 4-bit number at right. Identify the entry in s-box S1 at the second row/second column you determined. Convert it to binary; enter those two bits as the second half of the number at right.	result is 3 -> 11 right nibble: bits 1 & 4 -> 10 -> 2 bits 2 & 3 -> 00 -> 0 therefore, get from S1 row 2 col 0 result is 3 -> 11 1 1 1 1

S0 =

	c0	c1	c2	c3
r0	1	0	3	2
r1	3	2	1	0
r2	0	2	1	3
r3	3	1	3	2

S1 =

	c0	c1	c2	c3
r0	0	1	2	3
r1	2	0	1	3
r2	3	0	1	0
r3	2	1	0	3

To above result, apply permutation P4:	1 1 1 1

P4			
2	4	3	1

Upon the above P4 result, perform binary XOR operation, combining it with the left 4-bits of the earlier swap result (green cell above). We are trying to replace the left half of that swap result. These XOR result bits are the replacement bits for it.	XOR with 0001 1 1 1 0
Rewrite that "green" swap result with its left half replaced. (Look it up,	

keep/copy its right half, use the preceding result as the new left half.)	1 1 1 0 0 1 1 0
To above result, apply reverse of initial permutation IP, which is IP^{-1}: IP^{-1}: 4 1 3 5 7 2 8 6 This result is plaintext. It is the S-DES decryption of the ciphertext input. It should identically match the original plaintext, from which the ciphertext input to this process was first derived.	0 1 1 0 1 1 0 1

This decryption output duplicates the encryption input, so satisfies us that probably the algorithm works and our calculations are good.