Stream cipher (AES S-box based)

Referent

Email luca.crocetti@phd.unipi.it
Teams l.crocetti@studenti.unipi.it

Project

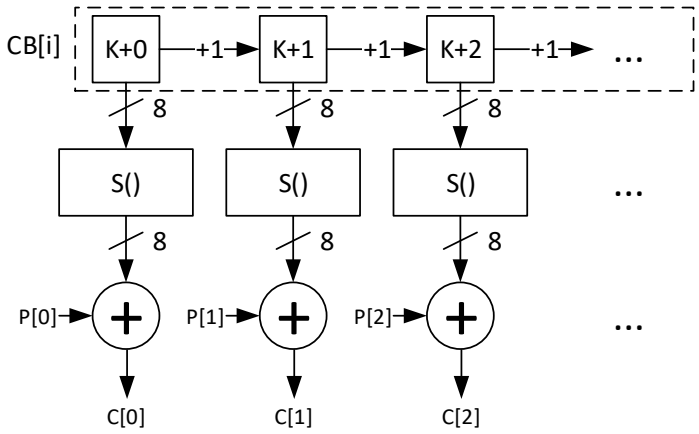
Design and implement an AES S-box based stream cipher supporting both encryption and decryption. The encryption algorithm of cipher is performed by XORing each character of the plaintext (an 8-bit ASCII coded character) with an 8-bit value obtained by substituting an 8-bit counter value with the S-box transformation of the AES algorithm. The 8-bit counter value (Counter Block, CB) has to be initialized with the value of an 8-bit symmetric key, K. The encryption law of the stream cipher can be expressed as it follows:

$$C[i] = P[i] \oplus S(CB[i])$$

where

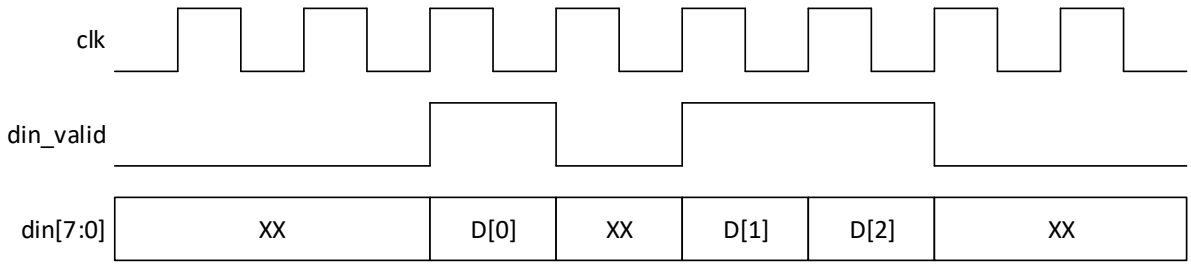
$C[i]$	is the 8-bit ASCII code of the ith character of ciphertext.
$P[i]$	is the 8-bit ASCII code of the ith character of plaintext.
$CB[i]$	is the 8-bit value of the ith counter (counter block), for $i = 0,1,2, \ldots$, and it can be represented by the formula $CB[i] = K + \ i \ mod \ 256$, being K the 8-bit symmetric key. Note mod is the modulo operation.
$S(\ \ )$	is the S-box transformation of AES algorithm, that works over a byte.
$\oplus$	is the XOR operator.

Following figure represents the encryption scheme of the stream cipher.

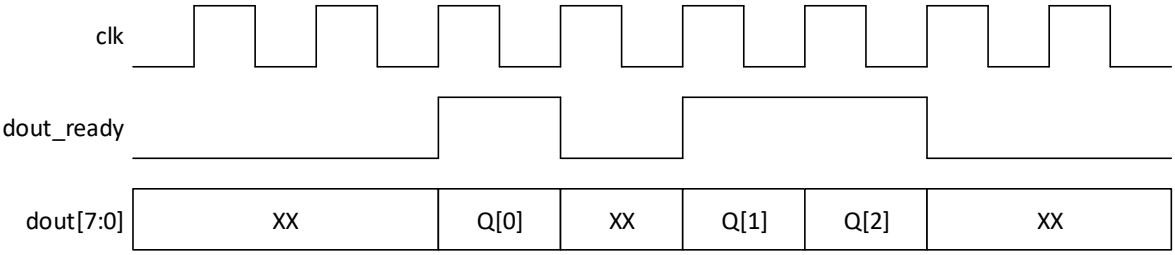


Additional design specifications

- The stream cipher shall have an asynchronous active-low reset port;
- The stream cipher shall process one plaintext/ciphertext character per clock cycle;
- The input data character (plaintext or ciphertext) can be any 8-bit ASCII character;
- The stream cipher shall feature an input port which has to be asserted when providing the input data (plaintext or ciphertext) character (*din_valid* port): 1'b1, when input character is valid and stable, 1'b0, otherwise; the following waveform is expected at input interface of stream cipher



- The stream cipher shall feature an output port which is asserted when the generated output character (ciphertext or plaintext) is available at the corresponding output port (*dout_ready* port): 1'b1, when output character is valid and stable, 1'b0, otherwise; this flag shall be kept to logic 1 at most for one clock cycle; the following waveform is expected at the output interface of stream cipher



- For each arbitrary length plaintext/ciphertext message, the counter block shall start from initialization value K (the 8-bit key)

Hints

- The AES S-box function is largely documented online: implement the LUT version (for faster developing).
  Below it is reported the S-box of AES algorithm, in hexadecimal format: it works on a byte, using the 4 MSb and the 4 LSb of input byte, respectively, as row and column coordinates to substitute it.

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

For instance, assuming to apply the S-box transformation to the byte (hex) 8'hd3, the result (hex) is 8'h66, i.e. the cross between row d0 and column 03: thus $S(8'hd3) = 8'h66$.

- Mind that before encrypting/decrypting the first character of a plaintext/ciphertext message the counter block shall be initialized with the key value, and that the counter block shall not be reinitialized with another key value (or the same) until the encryption/decryption of the whole message has been performed.
- Develop a testbench that encrypts two plaintext messages (from file or from string): for each plaintext message P, store the corresponding ciphertext C into proper object (file or string), then decrypt C (using the stream cipher) and store the corresponding plaintext P' into proper object (file or string); compare P' with P and check they match (character by character or using string methods). At least one of the two plaintext messages should be greater than 256 characters (around 300, for instance).
  For decryption focus on following XOR property:
  $$S \oplus S = 0$$
  thus, if $C = P \oplus S$, then $C \oplus S = (P \oplus S) \oplus S = P \oplus S \oplus S = P \oplus (S \oplus S) = P$.