St. Vincent Pallotti College of Engineering & Technology, Nagpur
Department of Computer Engineering
Session 2022-23

IANS Practical (7^{th} Sem A & B)
Practical 6

Problem Statement:

To create round key for each round using AES-128 bit key expansion process.

Aim:

WAP to implement the pseudo-code for AES-128 bit key expansion process.

Assume 128 bit cipher key as shown below:

Batch A: (24 75 A2 B3 34 75 56 88 31 E2 12 00 13 AA 54 87)

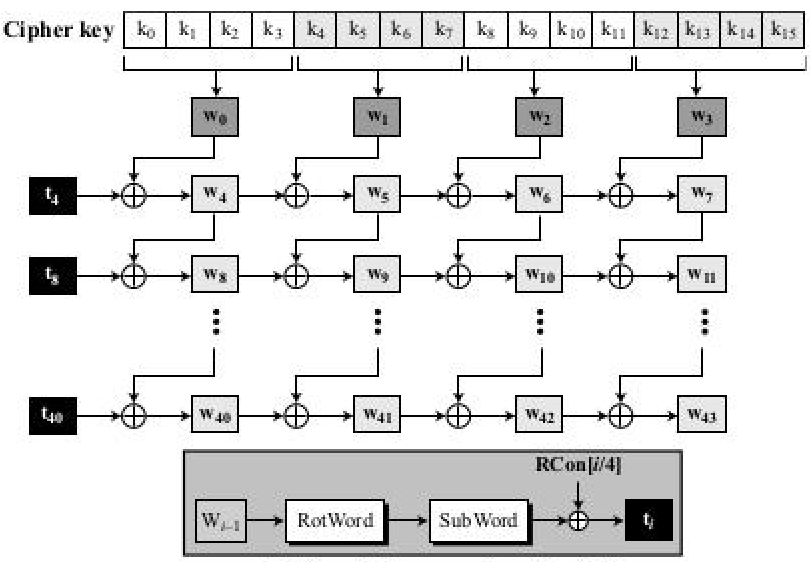
Batch B: (12 45 A2 A1 23 31 A4 A3 B2 CC AA 34 C2 BB 77 23)

Theory:

Key Expansion in AES-128

Let us show the creation of words for the AES-128 version; the processes for the other two versions are the same with some slight changes. Figure 7.16 shows how 44 words are made from the original key.

Figure 7.16 *Key expansion in AES*



Making of t_i (temporary) words $i = 4\ N_r$

The process is as follows:

1. The first four words ($\mathbf{w}_0$, $\mathbf{w}_1$, $\mathbf{w}_2$, $\mathbf{w}_3$) are made from the cipher key. The cipher key is thought of as an array of 16 bytes (k_0 to k_{15}). The first four bytes (k_0 to k_3) become $\mathbf{w}_0$; the next four bytes (k_4 to k_7) become $\mathbf{w}_1$; and so on. In other words, the concatenation of the words in this group replicates the cipher key.
2. The rest of the words ($\mathbf{w}_i$ for $i = 4$ to 43) are made as follows:
   a. If (i mod 4) $\neq 0$, $\mathbf{w}_i = \mathbf{w}_{i-1} \oplus \mathbf{w}_{i-4}$. Referring to Figure 7.16, this means each word is made from the one at the left and the one at the top.

b. If (i mod 4) = 0, $\mathbf{w}_i = \mathbf{t} \oplus \mathbf{w}_{i-4}$. Here **t**, a temporary word, is the result of applying two routines, SubWord and RotWord, on $\mathbf{w}_{i-1}$ and XORing the result with a round constants, RCon. In other words, we have,

$$\mathbf{t} = \text{SubWord}\ (\text{RotWord}\ (\mathbf{w}_{i-1})) \oplus \text{RCon}_{i/4}$$

RotWord

The **RotWord** (rotate word) routine is similar to the ShiftRows transformation, but it is applied to only one row. The routine takes a word as an array of four bytes and shifts each byte to the left with wrapping.

SubWord

The **SubWord** (substitute word) routine is similar to the SubBytes transformation, but it is applied only to four bytes. The routine takes each byte in the word and substitutes another byte for it.

Round Constants

Each round constant, RCon, is a 4-byte value in which the rightmost three bytes are always zero. Table 7.4 shows the values for AES-128 version (with 10 rounds).

Table 7.4 *RCon constants*

Round	*Constant (RCon)*	*Round*	*Constant (RCon)*
1	$(\underline{\mathbf{01}}\ 00\ 00\ 00)_{16}$	6	$(\underline{\mathbf{20}}\ 00\ 00\ 00)_{16}$
2	$(\underline{\mathbf{02}}\ 00\ 00\ 00)_{16}$	7	$(\underline{\mathbf{40}}\ 00\ 00\ 00)_{16}$
3	$(\underline{\mathbf{04}}\ 00\ 00\ 00)_{16}$	8	$(\underline{\mathbf{80}}\ 00\ 00\ 00)_{16}$
4	$(\underline{\mathbf{08}}\ 00\ 00\ 00)_{16}$	9	$(\underline{\mathbf{1B}}\ 00\ 00\ 00)_{16}$
5	$(\underline{\mathbf{10}}\ 00\ 00\ 00)_{16}$	10	$(\underline{\mathbf{36}}\ 00\ 00\ 00)_{16}$

The SubBytes transformation table is shown below:

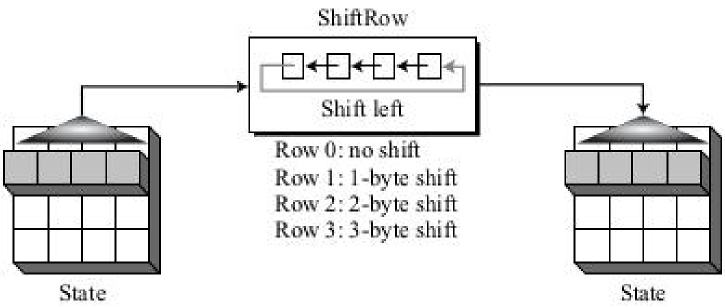
Table 7.1 *SubBytes transformation table*

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*A*	*B*	*C*	*D*	*E*	*F*
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	CB	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

ShiftRows

In the encryption, the transformation is called **ShiftRows** and the shifting is to the left. The number of shifts depends on the row number (0, 1, 2, or 3) of the state matrix. This means the row 0 is not shifted at all and the last row is shifted three bytes. Figure 7.9 shows the shifting transformation.

Figure 7.9 *ShiftRows transformation*



Note that the ShiftRows transformation operates one row at a time.

The key-expansion routine can either use the above table when calculating the words or use the GF(2^8) field to calculate the leftmost byte dynamically, as shown below (*prime* is the irreducible polynomial):

RC_1	$\rightarrow x^{1-1}$	$=x^0$	mod *prime*	$=1$	$\rightarrow$ 00000001	$\rightarrow 01_{16}$
RC_2	$\rightarrow x^{2-1}$	$=x^1$	mod *prime*	$=x$	$\rightarrow$ 00000010	$\rightarrow 02_{16}$
RC_3	$\rightarrow x^{3-1}$	$=x^2$	mod *prime*	$=x^2$	$\rightarrow$ 00000100	$\rightarrow 04_{16}$
RC_4	$\rightarrow x^{4-1}$	$=x^3$	mod *prime*	$=x^3$	$\rightarrow$ 00001000	$\rightarrow 08_{16}$
RC_5	$\rightarrow x^{5-1}$	$=x^4$	mod *prime*	$=x^4$	$\rightarrow$ 00010000	$\rightarrow 10_{16}$
RC_6	$\rightarrow x^{6-1}$	$=x^5$	mod *prime*	$=x^5$	$\rightarrow$ 00100000	$\rightarrow 20_{16}$
RC_7	$\rightarrow x^{7-1}$	$=x^6$	mod *prime*	$=x^6$	$\rightarrow$ 01000000	$\rightarrow 40_{16}$
RC_8	$\rightarrow x^{8-1}$	$=x^7$	mod *prime*	$=x^7$	$\rightarrow$ 10000000	$\rightarrow 80_{16}$
RC_9	$\rightarrow x^{9-1}$	$=x^8$	mod *prime*	$=x^4+x^3+x+1$	$\rightarrow$ 00011011	$\rightarrow 1B_{16}$
RC_{10}	$\rightarrow x^{10-1}$	$=x^9$	mod *prime*	$=x^5+x^4+x^2+x$	$\rightarrow$ 00110110	$\rightarrow 36_{16}$

The leftmost byte, which is called RC_i is actually x^{i-1}, where i is the round number. AES uses the irreducible polynomial $(x^8 + x^4 + x^3 + x + 1)$.

Algorithm

Algorithm 7.5 is a simple algorithm for the key-expansion routine (version AES-128).

Algorithm 7.5 *Pseudocode for key expansion in AES-128*

```
KeyExpansion ([key_0 to key_15], [w_0 to w_43])
{
    for (i = 0 to 3)
        w_i ← key_4i + key_4i+1 + key_4i+2 + key_4i+3

    for (i = 4 to 43)
    {
        if (i mod 4 ≠ 0)    w_i ← w_i−1 + w_i−4
        else
        {
            t ← SubWord (RotWord (w_i−1)) ⊕ RCon_i/4          // t is a temporary word
            w_i ← t + w_i−4
        }
    }
}
```

Example 7.6

Table 7.5 shows how the keys for each round are calculated assuming that the 128-bit cipher key agreed upon by Alice and Bob is $(24\ 75\ A2\ B3\ 34\ 75\ 56\ 88\ 31\ E2\ 12\ 00\ 13\ AA\ 54\ 87)_{16}$.

Table 7.5 *Key expansion example*

Round	*Values of t's*	*First word in the round*	*Second word in the round*	*Third word in the round*	*Fourth word in the round*
—		w_{00} = 2475A2B3	w_{01} = 34755688	w_{02} = 31E21200	w_{03} = 13AA5487
1	AD20177D	w_{04} = 8955B5CE	w_{05} = BD20E346	w_{06} = 8CC2F146	w_{07} = 9F68A5C1
2	470678DB	w_{08} = CE53CD15	w_{09} = 73732E53	w_{10} = FFB1DF15	w_{11} = 60D97AD4
3	31DA48D0	w_{12} = FF8985C5	w_{13} = 8CFAAB96	w_{14} = 734B7483	w_{15} = 2475A2B3
4	47AB5B7D	w_{16} = B822deb8	w_{17} = 34D8752E	w_{18} = 479301AD	w_{19} = 54010FFA
5	6C762D20	w_{20} = D454F398	w_{21} = E08C86B6	w_{22} = A71F871B	w_{23} = F31E88E1
6	52C4F80D	w_{24} = 86900B95	w_{25} = 661C8D23	w_{26} = C1030A38	w_{27} = 321D82D9
7	E4133523	w_{28} = 62833EB6	w_{29} = 049FB395	w_{30} = C59CB9AD	w_{31} = F7813B74
8	8CE29268	w_{32} = EE61ACDE	w_{33} = EAFE1F4B	w_{34} = 2F62A6E6	w_{35} = D8E39D92
9	0A5E4F61	w_{36} = E43FE3BF	w_{37} = 0EC1FCF4	w_{38} = 21A35A12	w_{39} = F940C780
10	3FC6CD99	w_{40} = DBF92E26	w_{41} = D538D2D2	w_{42} = F49B88C0	w_{43} = 0DDB4F40

In each round, the calculation of the last three words are very simple. For the calculation of the first word we need to first calculate the value of temporary word (**t**). For example, the first **t** (for round 1) is calculated as

RotWord (13AA5487) = AA548713 $\rightarrow$ **SubWord** (AA548713) = AC20177D

t = AC20177D $\oplus$ **RCon$_1$** = AC20 17 7D $\oplus$ 01000000_{16} = AD20177D

Conclusion:

Students need to write the AES Key expansion analysis for the same.