

# Design of Advanced Encryption Standard Using VHDL

<sup>1</sup>Chaudhari R. Priyanka, <sup>2</sup>Pinky Brahmhatt

<sup>1</sup>P.G. Student, <sup>2</sup>Associate Professor

<sup>1</sup>Electronics and Communication Department

<sup>1</sup>L.D. College of Engineering, Ahmedabad, India

**Abstract:** Security is very important parameter of today's era. Today most concerned thing about communication includes looks into on security concern. At present most recognized cryptographic algorithm for encryption is Advance Encryption Standard. Advanced encryption standard is most important algorithm of cryptographic. AES is symmetric key algorithm which is used to encrypt (cipher text) and decrypt (plain text) the data. Previously Data Encryption Standard (DES) is used but it's proved inadequate because of smaller key length. An AES takes an input block size of 128 bit. AES has three size of cryptographic key 128,196 and 256 bit. Basically, AES uses substitution and permutation operation. AES can be implemented by both software simulation and hardware simulation but as compared to software simulation hardware simulation has more merits like low area consumption and less time requirement. A huge amount of hardware required for implementation of Advanced Encryption standard because if it's complex nature. The huge amount of hardware requirement makes AES very complex and burdensome. In this paper we implemented FPGA based AES algorithm. Xilinx \_ISE 14.7 software is being used for simulating and to synthesize the VHDL code. Xilinx's SPARTEN 6 FPGA is used for implementation. One application is developed for transferring the data from PC to FPGA.

**Index Terms**—AES, Encryption, Decryption, FPGA, Visual Basic, VHDL ,Different Key Length.

## I. INTRODUCTION

In this era of technology data is considered as very important thing. Security is must because millions of devices are sharing their files daily such as financial and legal files, medical support, and bank related services via web, telephone talk, and transactions. Important files can be accessed by intruder very easily if data is not encrypted. Cryptographic is one of the secure algorithm. Cryptographic is Greek origin word which means "secret-writing". Cryptography was synonyms with encryption it convert original data or secure message in completely non sense form. Cryptography is secure way for transferring the information without theft or change. Cryptography is used for both side to encrypt the information and to decrypt the information. The encryption's task is to change our main information or plain text (data) into a cipher text (non-sense text). The decryption process is convert cipher text (non-sense text) into a plain text or original information. There are two types of cryptographic algorithm secret key (symmetric) and public key (asymmetric) algorithm. Secret key (symmetric) algorithm are algorithm for cryptography that used same cryptographic key for encryption and decryption. Key is also called private key. Advantage of symmetric key algorithm that it executes and it's less complex; they are used for bulk data transmission. Private Key (Asymmetric key) algorithm is known as public key algorithm, uses public and private key to encrypt and decrypt the data. One key is shared with everyone, and other key is kept secret. Asymmetric key is complex in nature as compared to symmetric key. So mainly symmetric key algorithm is used for to transmit the secure information. There are numerous types of secret key (symmetric) key algorithm: DES, AES, triple DES, RSA.

Data Encryption Standard is used to secure communication in earlier 90's. DES is symmetric key algorithm that use 56 bit key. DES has 64 bit plain text. It will convert 64 bit plain text into a 64 bit chipper text with the help of 56 bit symmetric key. But as said the time requirement to decrypt the encryption algorithm is proportional to the length of the bit key used for the safe conversion. To show that DES was inadequate and less secure a challenges were sponsored to see how long it would take to decrypt a message. In 1997 challengers took 87 days to decrypt the message using brute force attack. In 1998 challengers took less than one month to decrypt the DES. In 1999 challengers took 22 hours and 15 minutes. So they proved DES was inadequate. Even triple DES was inadequate and slower because of using DES encryption three times it was proved in adequate.

In 2001 National Institute of Standard and Technology introduced AES which is public (symmetric) key algorithm to overcome the Data Encryption Standard failure. AES is also known as RIJNDAEL algorithm. Till now no one can break RIJNDAEL algorithm. So AES algorithm is most important algorithm for secure communication.

The further paper is organized as follow: Section II gives application of cryptographic. Section III gives introduction of AES algorithm Section IV gives encryption of AES Section V gives decryption of AES Section VI gives simulation results. Finally section VII gives Conclusion.

## II. APPLICATION

1. Automatic Teller Machines (ATM)
2. Smart Cards
3. WhatsApp
4. HTTP Source

5. Bluetooth Transfer
6. Digital Signature

### III. INTRODUCTION OF AES

In 2001 National Institute of Standard and Technology introduced Advanced Encryption Standard developed by Joan Daemen of Proton World International and VicentFijmen of Ketholieke University Leauen Advance Encryption Standard is Symmetric block chipper that uses Substitution and Permutation. AES use for both side Encryption and Decryption. Encryption convert original data into a completely non-sense form that called chipper text. Decryption covert chipper text into an original data. AES is more efficient cryptographic algorithm compared to Des but its most positive advantage is that it can use various key length. AES allow to select a 128,192, 256 bit key length, making it exponentially stronger. AES has different number of rounds depends on size of the key. AES has 10, 12 and 14 rounds which are dependent to size of the key. In this paper we have used 128-bit key for 128-bit data.

Table 1  
AES ROUND INFORMATION

	Key size	Block size	Rounds
Aes 128 bit	4 bit	4 block	10 rounds
Aes 192 bit	6 bit	4 block	12 rounds
Aes 256 bit	8 bit	4 block	14 rounds

Table 2  
Comparison between AES and DES [1]

BASIS FOR COMPARISON	DES (DATA ENCRYPTION STANDARD)	AES (ADVANCED ENCRYPTION STANDARD)
Basic	In DES the data block is divided into two halves.	In AES the entire data block is processed as a single matrix.
Principle	DES work on Feistel Cipher structure.	AES works on Substitution and Permutation Principle.
Plaintext	Plaintext is of 64 bits	Plaintext can be of 128,192, or 256 bits
Key size	DES in comparison to AES has smaller key size.	AES has larger key size as compared to DES.
Rounds	16 rounds	10 rounds for 128-bit algo 12 rounds for 192-bit algo 14 rounds for 256-bit algo
Rounds Names	Expansion Permutation, Xor, S-box, P-box, Xor and Swap.	Subbytes, Shiftrows, Mix columns, Addroundkeys.
Security	DES has a smaller key which is less secure.	AES has large secret key comparatively hence, more secure.
Speed	DES is comparatively slower.	AES is faster.

### IV AES ENCRYPTION

Encryption transforms original information or plain text into a completely non-sense form called chipper text. Value of the rounds in Encryption process is depends on key length of the AES. AES has three different key lengths 128,192,256 bit key. AES algorithm operates on 128 bit block data and similarly 10 rounds for algorithm. Encryption process has four round block sub-byte transformation, Mix column transformation, Shift row, and Key expansion. An additional Add Round Key transformation is added at the beginning of the round and no Mix Columns transformation are added in the last round. The first step of the round is to convert 128 bit into 4 x 4 state matrix.

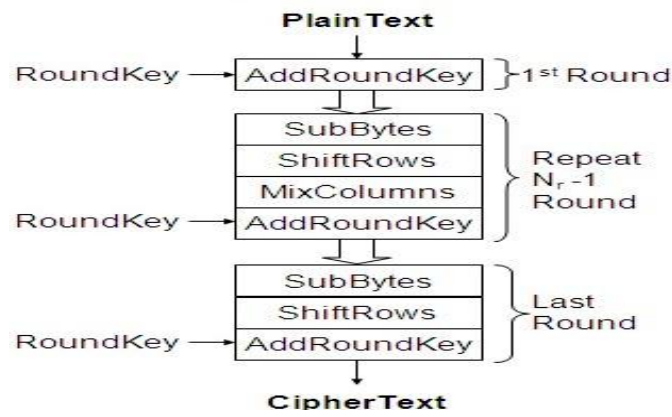


Figure 1 Block of Encryption

**1. Sub Byte Conversion:**

Sub Byte conversion is Non-linear transformation. In sub byte transformation each byte in substitute matrix is replaced by another byte using S-box. Using S-box first hexadecimal number correspond to row poisoning and second hexadecimal number correspond to column poisoning is replaced.

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

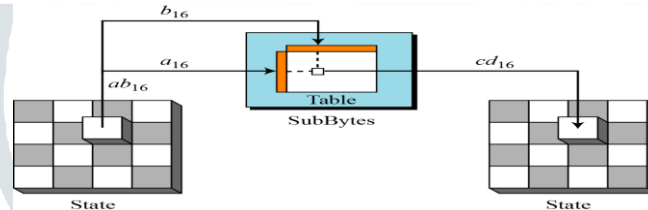


Figure 2: Sub byte transformation

**2. Shift Row Conversion:**

In Shift row conversion every row of the matrix is shifted by left. Rows are shifted left by 1 byte respectively, the first row of the matrix is does not change, second row is shifted by1 byte to the left, third row is shifted by 2 byte to the left and fourth row is shifted by 3 byte to the left.

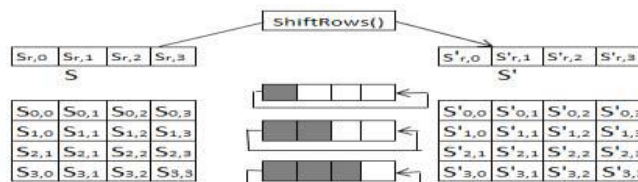


Figure 3: Shift Row transformation

**3. Mix Column Conversion:**

In mix column conversion each byte in column is replaced by another byte using Galois field ( $2^8$ ).

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 01 \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \dots \text{for } 0 \leq c \leq Nb$$

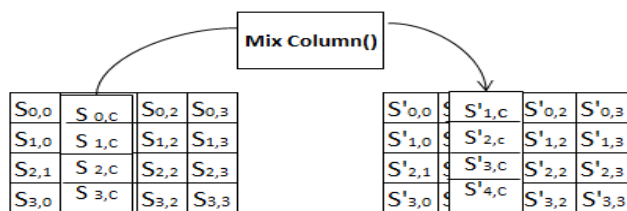


Figure 4: Mix column transformation

**4. Add round key:**

In add round key operation output of the mix column conversion is EX-OR with input key. The EX-OR operation is done bit by bit. Total 10 round key is obtain from this operation.

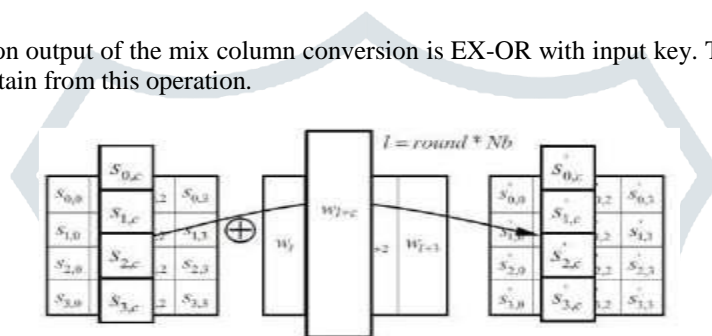


Figure 5: round key generation

**5. Key Expansion:**

From the first master key word total 10 key word matrix will be obtained.

Method:

- For the word column in any key matrix which is not a multiple of 4 it will take simply XOR (W<sub>i-1</sub>) and (W<sub>i-4</sub>).But for word column in any key matrix which is a multiple of 4 is filled in a special way.

Let us take one matrix which contain 128 bit data. Here W<sub>4</sub> is a multiple of 4 we follow following procedure:

32	88	31	e0
43	5a	31	37
f6	30	98	07
A8	8d	a2	34

**Step 1:**

Rotate by one byte column that is just previous to it. In case of previous column is W<sub>3</sub> so rotate it one byte. it can get following O/P.

37
07
34
e0

**Step 2:**

Now apply the S-box to all four bytes of above O/P.

9a
c5
18
e1

**Step 3**

Now it will XOR the above O/P by a round constant RCON according to a predefined value for each round as follows:

**Values of RCON in hexadecimal**

Rcon =

01	02	04	08	10	20	40	80	1b	36
00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00

As seen into a table for round-1 RCON value is 01.

$$\begin{pmatrix} 9a \\ c5 \\ 18 \\ e1 \end{pmatrix} + \begin{pmatrix} 01 \\ 00 \\ 00 \\ 00 \end{pmatrix} = \begin{pmatrix} 9b \\ c5 \\ 18 \\ e1 \end{pmatrix}$$

**Step 4:**

Finally XOR the value of above step with fourth previous column to i.e.  $W_{i-4}$  i.e.  $W_{4-4}$  i.e.  $W_0$  which is,

$$\begin{pmatrix} 32 \\ 43 \\ f6 \\ a8 \end{pmatrix} + \begin{pmatrix} 9b \\ c5 \\ 18 \\ e1 \end{pmatrix} = \begin{pmatrix} a6 \\ 86 \\ e6 \\ 59 \end{pmatrix}$$

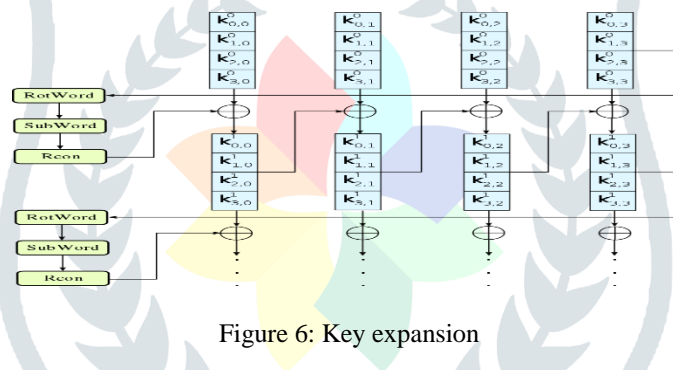


Figure 6: Key expansion

**V. AES DECRYPTION**

AES Decryption is inverse process of encryption. Decryption transforms cipher text or encrypted text into an original or plain text. The key length is same for decryption process. The decryption process has four rounds inv shift row, inv sub byte transformation, inv mix column transformation and inv key expansion.

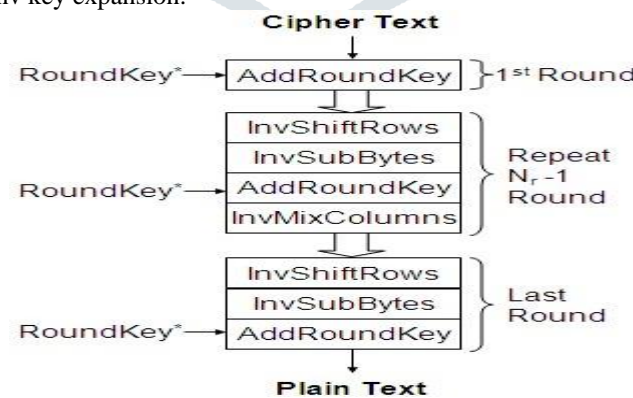


Figure 7: Block of Decryption

**1. Add round key:**

EX-OR is done between cipher text and round key.

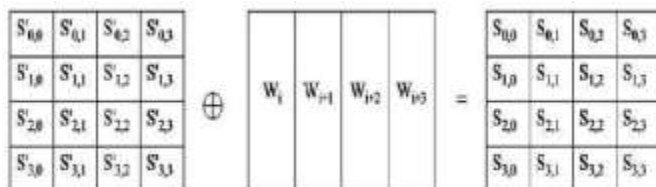


Figure 8: Add round key

**2. Inverse sub byte transformation:**

In Inverse sub byte every byte of the matrix is replaced by another byte using inverse S-box look up table.

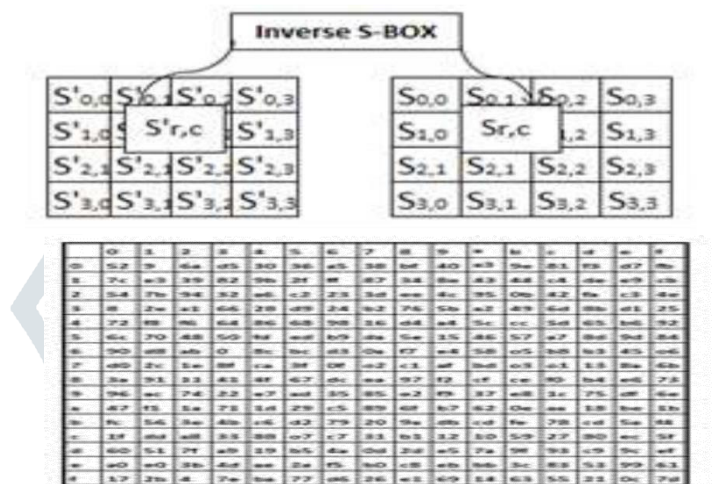


Figure 9: sub byte inverse conversion

**3. Inverse shift row transformation:**

In inverse Shift row transformation every row of the matrix is cyclically shifted by right. Rows are shifted right by 1 byte respectively, The first row of the matrix is does not change, second row is shifted by 1 byte to the right, third row is shifted by 2 byte to the right and fourth row is shifted by 3 byte to the right.

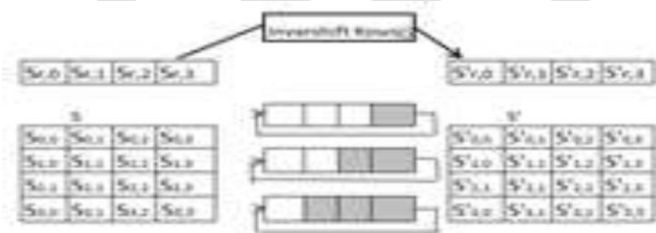


Figure 10: shift row inverse conversion

**4. Mix column inverse conversion :**

In mix column inverse conversion every byte of the column matrix is replaced by another byte using Galois field (2<sup>8</sup>).

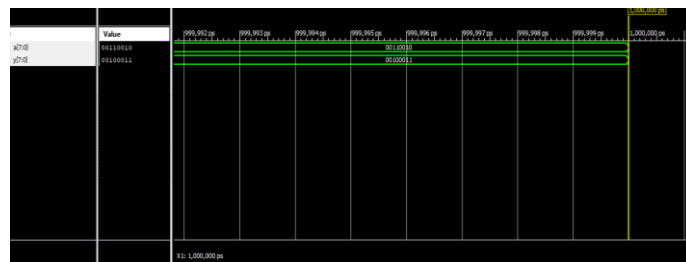
$$\begin{bmatrix} S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \\ S'_{4,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \dots \text{for } 0 \leq c \leq Nb$$

Figure 11: Inverse mix column transformation

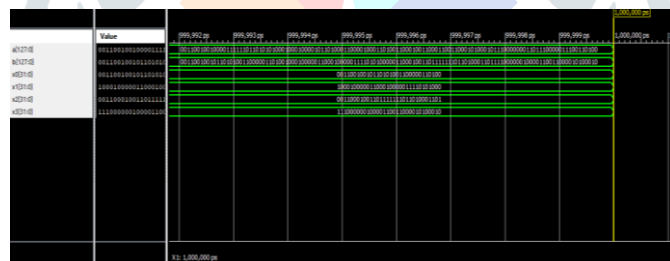
**VI. Simulation Result**

The purpose of our paper to design Advanced Encryption Algorithm using VHDL to improve performance. The Xilinx 14.7 environment was used for written the code.After coding behavioural Simulation was done using Xilinx ISM.

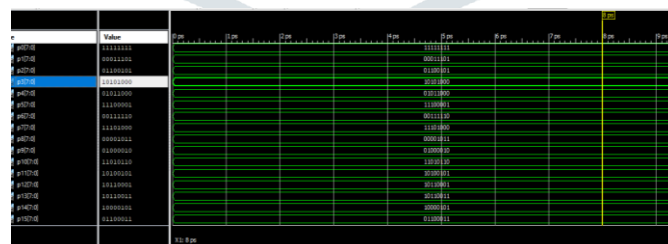
**1. S-box and Sub byte transformation**



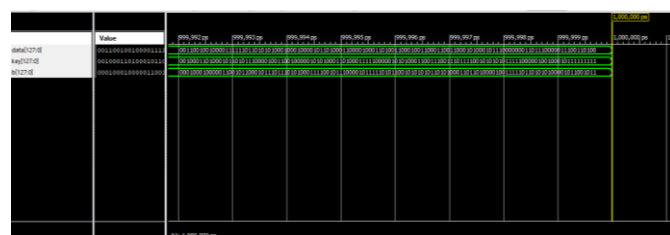
**2. Shift row transformation:**



**3. Mix column transformation:**



**4. Add round key:**



5. Key expansion:

Value	key_0	key_1	key_2	key_3	key_4	key_5	key_6	key_7	key_8	key_9	key_10	key_11	key_12
key_0	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
key_1	00000001	00000002	00000003	00000004	00000005	00000006	00000007	00000008	00000009	0000000A	0000000B	0000000C	0000000D
key_2	0000000E	0000000F	00000010	00000011	00000012	00000013	00000014	00000015	00000016	00000017	00000018	00000019	0000001A
key_3	0000001B	0000001C	0000001D	0000001E	0000001F	00000020	00000021	00000022	00000023	00000024	00000025	00000026	00000027
key_4	00000028	00000029	0000002A	0000002B	0000002C	0000002D	0000002E	0000002F	00000030	00000031	00000032	00000033	00000034
key_5	00000035	00000036	00000037	00000038	00000039	0000003A	0000003B	0000003C	0000003D	0000003E	0000003F	00000040	00000041
key_6	00000042	00000043	00000044	00000045	00000046	00000047	00000048	00000049	0000004A	0000004B	0000004C	0000004D	0000004E
key_7	0000004F	00000050	00000051	00000052	00000053	00000054	00000055	00000056	00000057	00000058	00000059	0000005A	0000005B
key_8	0000005C	0000005D	0000005E	0000005F	00000060	00000061	00000062	00000063	00000064	00000065	00000066	00000067	00000068
key_9	00000069	0000006A	0000006B	0000006C	0000006D	0000006E	0000006F	00000070	00000071	00000072	00000073	00000074	00000075
key_10	00000076	00000077	00000078	00000079	0000007A	0000007B	0000007C	0000007D	0000007E	0000007F	00000080	00000081	00000082
key_11	00000083	00000084	00000085	00000086	00000087	00000088	00000089	0000008A	0000008B	0000008C	0000008D	0000008E	0000008F
key_12	00000090	00000091	00000092	00000093	00000094	00000095	00000096	00000097	00000098	00000099	0000009A	0000009B	0000009C
key_13	0000009D	0000009E	0000009F	000000A0	000000A1	000000A2	000000A3	000000A4	000000A5	000000A6	000000A7	000000A8	000000A9
key_14	000000AA	000000AB	000000AC	000000AD	000000AE	000000AF	000000B0	000000B1	000000B2	000000B3	000000B4	000000B5	000000B6
key_15	000000B7	000000B8	000000B9	000000BA	000000BB	000000BC	000000BD	000000BE	000000BF	000000C0	000000C1	000000C2	000000C3
key_16	000000C4	000000C5	000000C6	000000C7	000000C8	000000C9	000000CA	000000CB	000000CC	000000CD	000000CE	000000CF	000000D0
key_17	000000D1	000000D2	000000D3	000000D4	000000D5	000000D6	000000D7	000000D8	000000D9	000000DA	000000DB	000000DC	000000DD
key_18	000000DE	000000DF	000000E0	000000E1	000000E2	000000E3	000000E4	000000E5	000000E6	000000E7	000000E8	000000E9	000000EA
key_19	000000EB	000000EC	000000ED	000000EE	000000EF	000000F0	000000F1	000000F2	000000F3	000000F4	000000F5	000000F6	000000F7
key_20	000000F8	000000F9	000000FA	000000FB	000000FC	000000FD	000000FE	000000FF	00000100	00000101	00000102	00000103	00000104
key_21	00000105	00000106	00000107	00000108	00000109	0000010A	0000010B	0000010C	0000010D	0000010E	0000010F	00000110	00000111
key_22	00000112	00000113	00000114	00000115	00000116	00000117	00000118	00000119	0000011A	0000011B	0000011C	0000011D	0000011E
key_23	0000011F	00000120	00000121	00000122	00000123	00000124	00000125	00000126	00000127	00000128	00000129	0000012A	0000012B
key_24	0000012C	0000012D	0000012E	0000012F	00000130	00000131	00000132	00000133	00000134	00000135	00000136	00000137	00000138
key_25	00000139	0000013A	0000013B	0000013C	0000013D	0000013E	0000013F	00000140	00000141	00000142	00000143	00000144	00000145
key_26	00000146	00000147	00000148	00000149	0000014A	0000014B	0000014C	0000014D	0000014E	0000014F	00000150	00000151	00000152
key_27	00000153	00000154	00000155	00000156	00000157	00000158	00000159	0000015A	0000015B	0000015C	0000015D	0000015E	0000015F
key_28	00000160	00000161	00000162	00000163	00000164	00000165	00000166	00000167	00000168	00000169	0000016A	0000016B	0000016C
key_29	0000016D	0000016E	0000016F	00000170	00000171	00000172	00000173	00000174	00000175	00000176	00000177	00000178	00000179
key_30	0000017A	0000017B	0000017C	0000017D	0000017E	0000017F	00000180	00000181	00000182	00000183	00000184	00000185	00000186
key_31	00000187	00000188	00000189	0000018A	0000018B	0000018C	0000018D	0000018E	0000018F	00000190	00000191	00000192	00000193
key_32	00000194	00000195	00000196	00000197	00000198	00000199	0000019A	0000019B	0000019C	0000019D	0000019E	0000019F	000001A0
key_33	000001A1	000001A2	000001A3	000001A4	000001A5	000001A6	000001A7	000001A8	000001A9	000001AA	000001AB	000001AC	000001AD
key_34	000001AE	000001AF	000001B0	000001B1	000001B2	000001B3	000001B4	000001B5	000001B6	000001B7	000001B8	000001B9	000001BA
key_35	000001BB	000001BC	000001BD	000001BE	000001BF	000001C0	000001C1	000001C2	000001C3	000001C4	000001C5	000001C6	000001C7
key_36	000001C8	000001C9	000001CA	000001CB	000001CC	000001CD	000001CE	000001CF	000001D0	000001D1	000001D2	000001D3	000001D4
key_37	000001D5	000001D6	000001D7	000001D8	000001D9	000001DA	000001DB	000001DC	000001DD	000001DE	000001DF	000001E0	000001E1
key_38	000001E2	000001E3	000001E4	000001E5	000001E6	000001E7	000001E8	000001E9	000001EA	000001EB	000001EC	000001ED	000001EE
key_39	000001EF	000001F0	000001F1	000001F2	000001F3	000001F4	000001F5	000001F6	000001F7	000001F8	000001F9	000001FA	000001FB
key_40	000001FC	000001FD	000001FE	000001FF	00000200	00000201	00000202	00000203	00000204	00000205	00000206	00000207	00000208
key_41	00000209	0000020A	0000020B	0000020C	0000020D	0000020E	0000020F	00000210	00000211	00000212	00000213	00000214	00000215
key_42	00000216	00000217	00000218	00000219	0000021A	0000021B	0000021C	0000021D	0000021E	0000021F	00000220	00000221	00000222
key_43	00000223	00000224	00000225	00000226	00000227	00000228	00000229	0000022A	0000022B	0000022C	0000022D	0000022E	0000022F
key_44	00000230	00000231	00000232	00000233	00000234	00000235	00000236	00000237	00000238	00000239	0000023A	0000023B	0000023C
key_45	0000023D	0000023E	0000023F	00000240	00000241	00000242	00000243	00000244	00000245	00000246	00000247	00000248	00000249
key_46	0000024A	0000024B	0000024C	0000024D	0000024E	0000024F	00000250	00000251	00000252	00000253	00000254	00000255	00000256
key_47	00000257	00000258	00000259	0000025A	0000025B	0000025C	0000025D	0000025E	0000025F	00000260	00000261	00000262	00000263
key_48	00000264	00000265	00000266	00000267	00000268	00000269	0000026A	0000026B	0000026C	0000026D	0000026E	0000026F	00000270
key_49	00000271	00000272	00000273	00000274	00000275	00000276	00000277	00000278	00000279	0000027A	0000027B	0000027C	0000027D
key_50	0000027E	0000027F	00000280	00000281	00000282	00000283	00000284	00000285	00000286	00000287	00000288	00000289	0000028A
key_51	0000028B	0000028C	0000028D	0000028E	0000028F	00000290	00000291	00000292	00000293	00000294	00000295	00000296	00000297
key_52	00000298	00000299	0000029A	0000029B	0000029C	0000029D	0000029E	0000029F	000002A0	000002A1	000002A2	000002A3	000002A4
key_53	000002A5	000002A6	000002A7	000002A8	000002A9	000002AA	000002AB	000002AC	000002AD	000002AE	000002AF	000002B0	000002B1
key_54	000002B2	000002B3	000002B4	000002B5	000002B6	000002B7	000002B8	000002B9	000002BA	000002BB	000002BC	000002BD	000002BE
key_55	000002BF	000002C0	000002C1	000002C2	000002C3	000002C4	000002C5	000002C6	000002C7	000002C8	000002C9	000002CA	000002CB
key_56	000002CC	000002CD	000002CE	000002CF	000002D0	000002D1	000002D2	000002D3	000002D4	000002D5	000002D6	000002D7	000002D8
key_57	000002D9	000002DA	000002DB	000002DC	000002DD	000002DE	000002DF	000002E0	000002E1	000002E2	000002E3	000002E4	000002E5
key_58	000002E6	000002E7	000002E8	000002E9	000002EA	000002EB	000002EC	000002ED	000002EE	000002EF	000002F0	000002F1	000002F2
key_59	000002F3	000002F4	000002F5	000002F6	000002F7	000002F8	000002F9	000002FA	000002FB	000002FC	000002FD	000002FE	000002FF
key_60	00000300	00000301	00000302	00000303	00000304	00000305	00000306	00000307	00000308	00000309	0000030A	0000030B	0000030C
key_61	0000030D												