ECE 459/559
Secure & Trustworthy Computer Hardware Design

Modern Cryptography Standards
AES & SIMON

Garrett S. Rose

Spring 2017

THE UNIVERSITY OF TENNESSEE KNOXVILLE

Summary

- Advanced Encryption Standard (AES)
- AES vs. DES
- SIMON

The AES Contest

- 1997 – NIST calls for proposals
  Criteria:
  - Unclassified code
  - Publicly disclosed
  - Royalty-free worldwide
  - Symmetric block cipher for 128-bit blocks
  - Usable with keys of 128, 192, and 256 bits
- 1998 – 15 algorithms selected

The AES Contest

- 1999 – 5 finalists
  - MARS from IBM
  - RC6 by RSA Laboratories
  - Rijndael (RINE-dahl) by Joan Daeman and Vincent Rijmen
  - Serpent by Ross Anderson, Eli Biham and Lars Knudsen
  - Twofish by Bruce Schneier, John Kelsey, Doug Whiting, Dawid Wagner, Chris Hall and Niels Ferguson
- Evaluation of finalists
  - Public and private security
  - Key evaluation areas:

    security / cost or efficiency of operation / ease of software implementation

The AES Contest

- 2001 – and the winner is …

    Rijndael (RINE-dahl)
    Authors: Joan Daeman and Vincent Rijmen (both Belgian)

- Adopted by US government as Federal Information Processing Standard 197 (FIPS 197)

Overview of Rijndael/AES

- Similar to DES – cyclic type of approach
  - 128-bit blocks of plaintext P
  - # of iterations based on key length
    - 128-bit key => 9 rounds
    - 192-bit key => 11 rounds
    - 256-bit key => 13 rounds
- Basic operations for each round:
  - Substitution – byte level (confusion)
  - Shift row (transposition) – depends on key length (diffusion)
  - Mix columns – LSH and XOR (confusion + diffusion)
  - Add subkey – XOR used (confusion)

Overview of Rijndael/AES

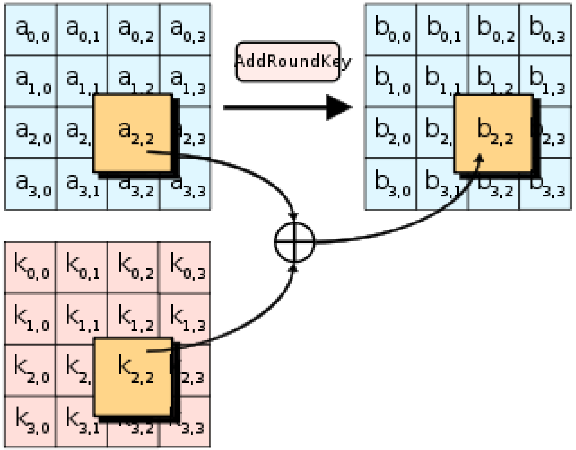
- Rounds – except for the last round, all rounds are identical
- Round processing based on arranging 128-bit block into 4x4 <u>**state array**</u> of bytes, as follows:

*byte*0	*byte*4	*byte*8	*byte*12
*byte*1	*byte*5	*byte*9	*byte*13
*byte*2	*byte*6	*byte*10	*byte*14
*byte*3	*byte*7	*byte*11	*byte*15

- For each round, 4x4 **input state array** is used to produce 4x4 **output state array**
- AES also uses the notion of a <u>**word**</u>, or 4 bytes (32 bits)
- Unlike DES, AES decryption algorithm *different* from encryption

AES Round Steps

- **AddRoundKey** – transformation step applies bitwise XOR between state array and elements of the RoundKey (Remember: RoundKey generated for each round from AES key)



$$b'(i,j) = a(i,j) \oplus k(i,j)$$

AES Round Steps

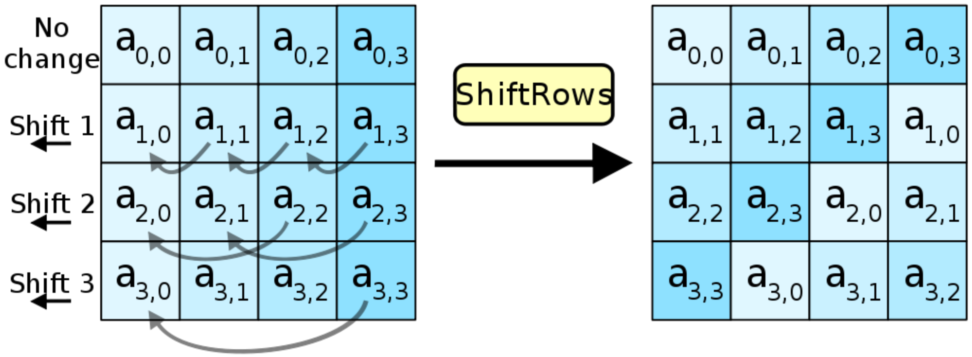
- **<u>SubBytes</u>** – provides nonlinear byte substitution by operating on each byte in state array using a look-up table called **S-Box** (If interested, S-Box based on Galois field operations)

- From S-Box look-up table (below), each byte 'XY' in state array replaced by byte at intersection of row X with column Y

	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	C8	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

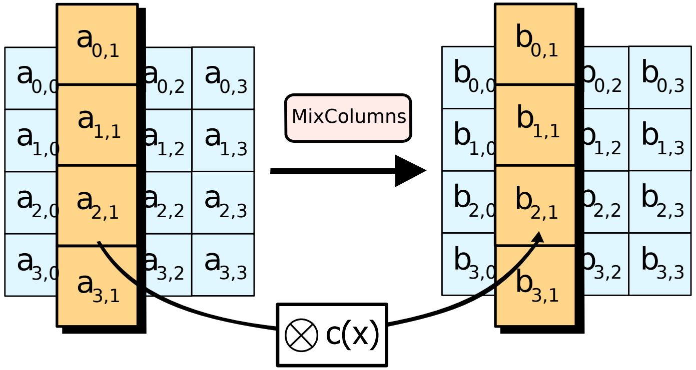
AES Round Steps

- **<u>ShiftRows</u>** – shifts each row in state array cyclically (performs a tranposition operation)



AES Round Steps

- **<u>MixColumns</u>** – performs column by column transformation (along with ShiftRows, primary source of diffusion)
- After 10 rounds of processing (for AES128), each bit in resulting ciphertext depends on every bit of the original plaintext – provides what's known as an "**avalanche effect**"



$$c(x) = 3x^3 + x^2 + x + 2$$

AES Key Expansion (KeySchedule)

- From AES cipher key, generates keys used for each AES round (for AES128, a total of 10 rounds requires 10 round keys)
- Round keys are an expansion of the cipher key

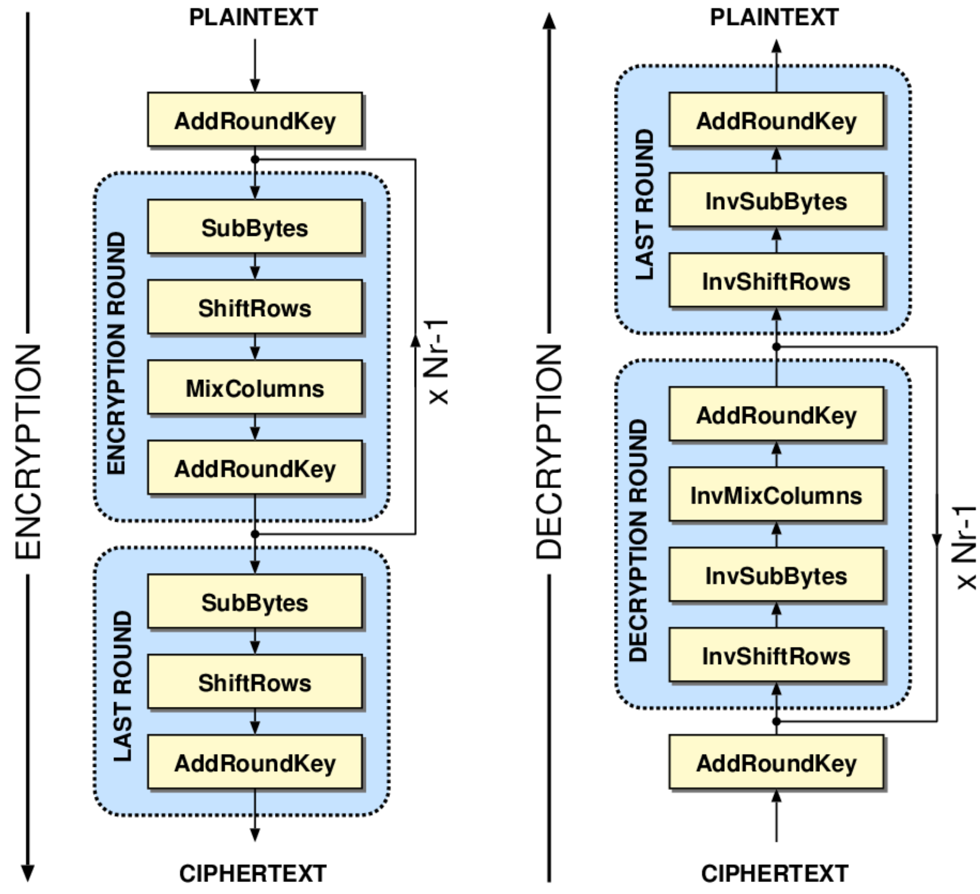
W_0	W_1	W_2	W_3																	W_{43}
2b	28	aB	9	a0	88	23	2a	f2	7a	59	73	3d	47	1e	6d	.....	d0	c9	e1	b6
7e	Ae	f7	cf	fa	54	a3	6c	c2	96	35	59	80	16	23	7a	.....	14	ee	3f	63
15	a2	15	4f	fe	2c	39	76	95	b9	80	f6	47	fe	7e	88		f9	25	0c	0c
16	a6	88	3c	17	b1	39	5	f2	43	7a	7f	7d	3e	44	3b		a8	89	c8	a6
CipherKey				RoundKey 1				RoundKey 2				RoundKey 3					RoundKey 10			

- Compute each new column W(i) (after cipher key columns) by:
  - Apply rotation and SubBytes transformation to previous column W(i-1)
  - Add this result to column 4 positions back W(i-4) plus a provided constant “Rcon”

THE UNIVERSITY OF TENNESSEE KNOXVILLE

AES Encrytion & Decryption

- Encryption functions have inverse counterparts:
  - InvShiftRows
  - InvSubBytes
  - InvMixColumns
- Decryption not same as encryption
  - **Non-Feistel Cipher**
- DES is **Feistel** Cipher
  - Encryption and decryption algorithms same



Strengths of AES

- Extensive cryptanalysis by government and independent experts
- Belgian inventors have no ties to the NSA or other US government bodies (less suspicion of built-in trapdoors)
- Solid mathematical foundation
  - Despite seemingly simple steps within the rounds

Comparing DES & AES

	DES	AES
Date (standardized)	1976	2001
Block size (bits)	64	128
Key length (bits)	56 (effectively)	128, 192, 256, or more
Encryption Primitives	Substitution, Permutation	Substitution, Shift, Bit Mixing
Cryptographic Primitives	Confusion, Diffusion	Confusion, Diffusion
Design	Open	Open
Design Rationale	Open	Open
Selection Process	Secret	Secret, accepted public comments
Source	IBM, NSA enhanced	Dutch researchers

Comparing DES & AES

- Weaknesses in AES?
  - 20+ years experience with DES eliminated fears of weakness (intentional or not) – likely naive
  - Experts pored over AES through 2-year review period
- Longevity of AES?
  - DES is 40 years old now (1976) – can be cracked in days
  - Longevity of AES more difficult to answer
    - Can extend key length over 256 bits (DES: 56)
    - Can extend the number of rounds (DES: 16)
  - Extensibility of AES seems significantly better than DES, but…
    - Human ingenuity is impressive!
    - Need to incessantly search for better and better algorithms

SIMON & SPECK Ciphers

- Developed by the NSA in 2013
- Lightweight block ciphers
  - 10 block/key sizes range from 32/64 to 128/256
- High performance on ASICs, FPGAs, microcontrollers and microprocessors
- Support range of implementations
  - From very compact to very high throughput / low latency

Block Size	Key Sizes
32	64
48	72, 96
64	96, 128
96	96, 144
128	128, 192, 256

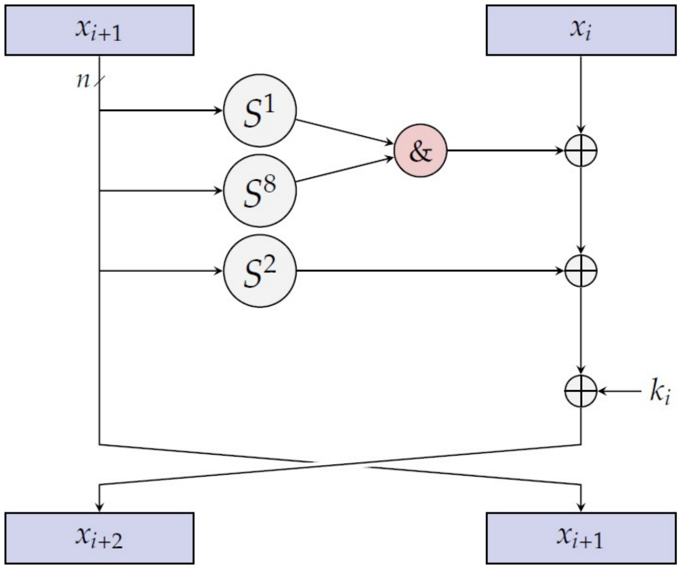
SIMON

- Goal is versatility in both hardware and software
- Optimized more for hardware implementation
- SIMON is a Feistel cipher – symmetric structure, in that same logic used for decryption as for encryption (like DES)
- Record breaking performance on ASICs and FPGAs
- Creators of SIMON claim 50% less area than AES on FPGA
- Research shows SIMON can be as much as 86% more area efficient on an FPGA!

Ref.: Aysu, et al., “SIMON Says: Break Area Records of Block Ciphers on FPGAs,” IEEE ESL, 2014.

SIMON Round Functions

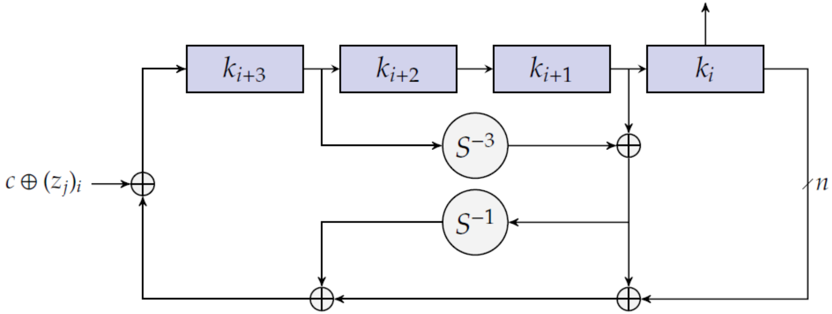
- At input to round, block data broken into 2 halves
- Functions used for each round:
  - Bitwise XOR, ⊕
  - Bitwise AND, &
  - Left circular shift, S^j (for j bits)
- At end of each round, left and right halves of block data swapped



THE UNIVERSITY OF TENNESSEE KNOXVILLE

SIMON Key Expansion

- Options for key expansion:
  - Two, three, and four-word expansion
- SIMON rounds are perfectly symmetric w.r.t. circular shift
- At first round (i=0), registers, k_{i+3}, k_{i+2}, k_{i+1}, and k_i loaded with key
  - For each new round i, key registers updated accordingly
- Constants eliminate slide properties and circular shift symmetries
  - E.g., u = $u_0 u_1 u_2$ … = 11111010001001010110000111001110
  - * **<u>For Lab 1</u>**, constant u and corresponding operation $c \oplus (z_j)i$ can be determined using provided u_bit(…) and xor16bit_triple(…) components



SIMON – VHDL Ports & Signals

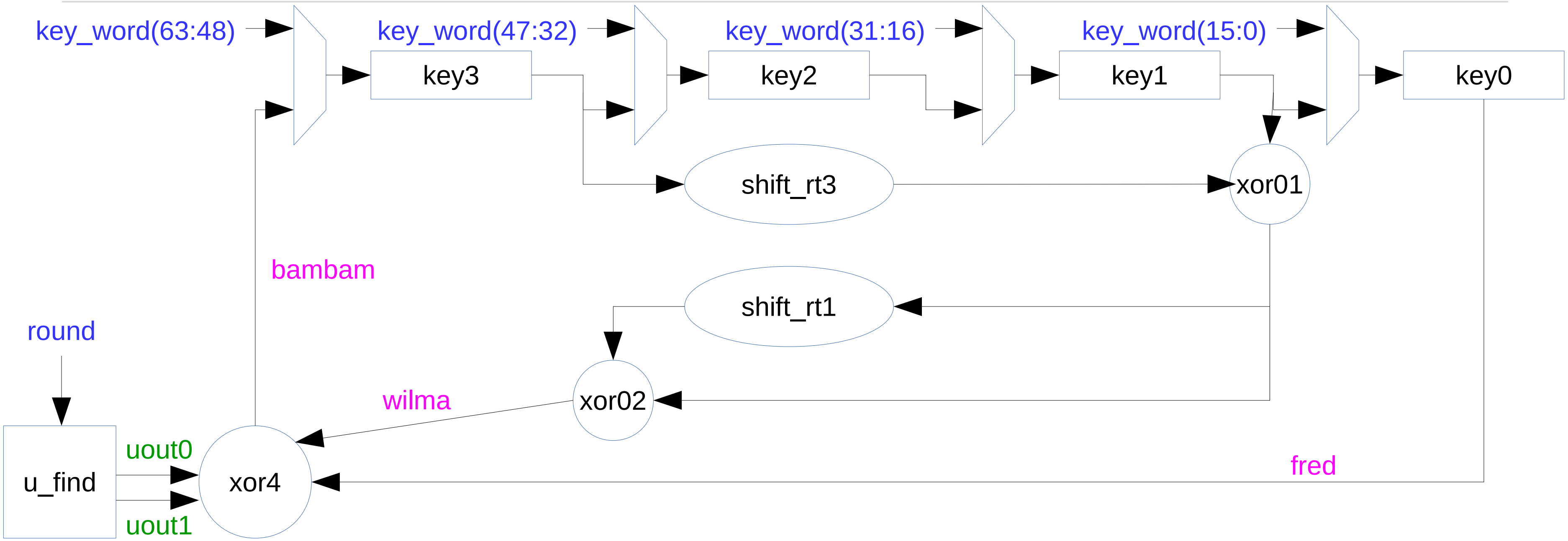
- Some inputs to key_expansion component (KEY_EXPANSION.vhd):

```
port(
  ...
  round    : in  std_logic_vector(7 downto 0);   -- indicates round number i
  key_word : in  std_logic_vector(63 downto 0);  -- 64-bit supplied cipher key
  ...
);
```

- Declare some signals (between “architecture” and “begin”):

```
...
signal uout0   : std_logic_vector(15 downto 0);
signal uout1   : std_logic_vector(15 downto 0);
signal wilma   : std_logic_vector(15 downto 0);
signal fred    : std_logic_vector(15 downto 0);
signal bambam  : std_logic_vector(15 downto 0);
...
```

SIMON – Some VHDL



- Connect up components for useful function:

```
u_find  : u_bit            PORT MAP (round, uout0, uout1);

xor4    : xor16bit_triple PORT MAP (wilma, fred, uout0, uout1, bambam);
```