Unified Hardware Architecture for 128-bit Block Ciphers AES and Camellia

A. Satoh and S. Morioka
Tokyo Research Laboratory
IBM Japan Ltd.

Contents

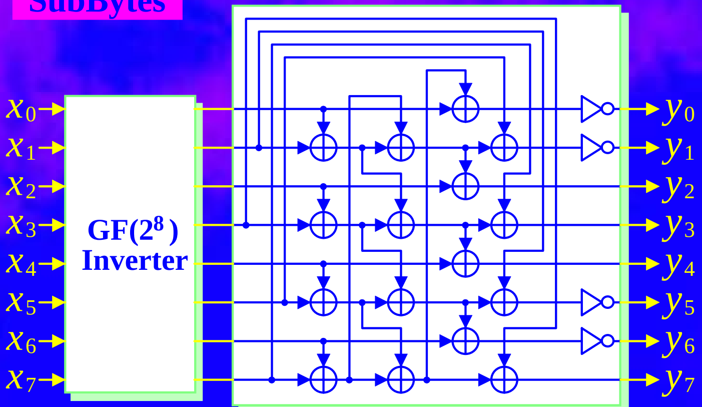
- Unified S-Box
- Unified Permutation Layer
- Unified Data Path Architecture
- ASIC Implementation Results
- Conclusion

Unified S-Box

AES S-Box

- Combinations of GF(2^8) inverter and affine transformations
- Inverter followed by affine transformation for encryption and inverter follows affine transformation for decryption

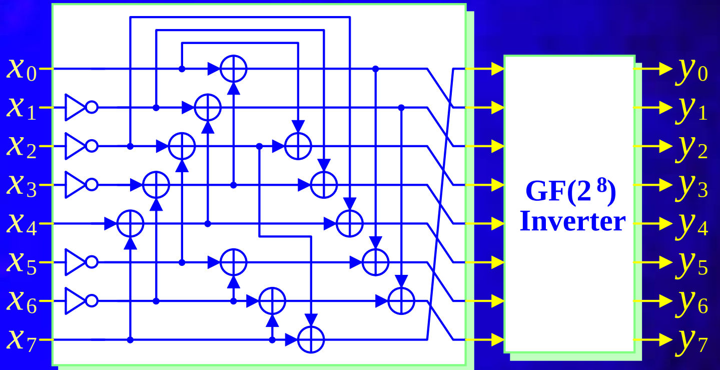
SubBytes



Affine Transformation *A*

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

InvSubBytes

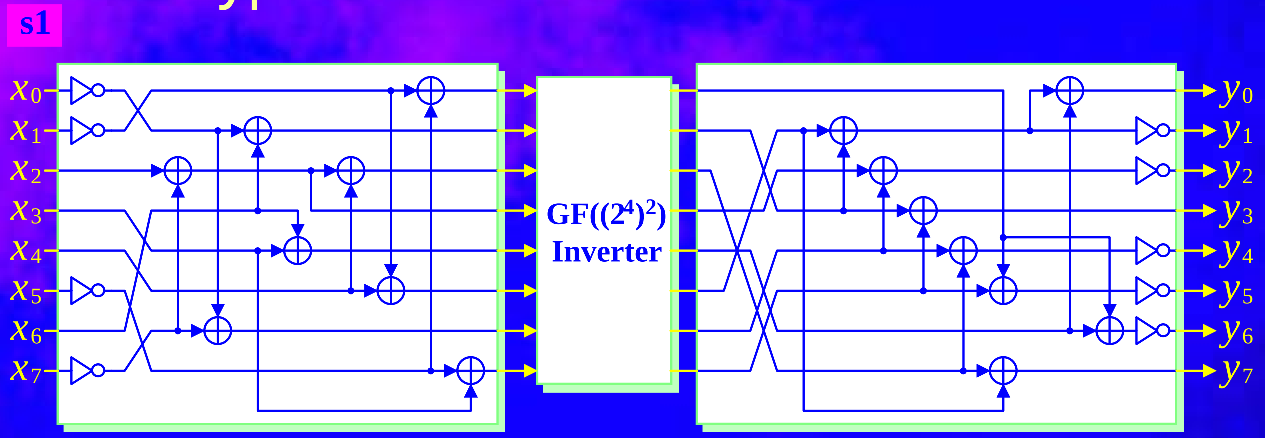


Affine Transformation A^{-1}

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} a_0 \oplus 1 \\ a_1 \oplus 1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \oplus 1 \\ a_6 \oplus 1 \\ a_7 \end{pmatrix}$$

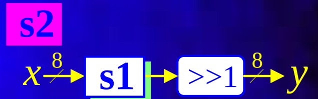
Camellia S-Box

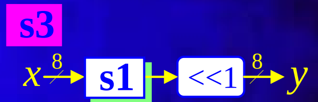
- GF((2^4)^2) inverter is placed between two affine transformations
- Four S-Boxes S1~S4 (I/O ordering is differed) are used
- Feistel-type cipher Camellia uses same S-Box in encryption and decryption

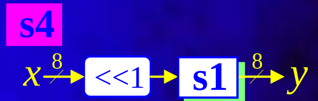


Affine Transformation *F*

Affine Transformation *H*





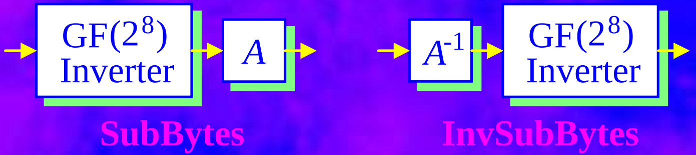


$$\begin{pmatrix} 0&1&0&0&0&1&0&0\\ 1&0&0&0&0&0&1&0\\ 0&0&1&0&1&0&0&1\\ 0&0&1&0&0&0&0&1\\ 0&0&0&1&0&0&1&0\\ 0&1&0&0&1&0&0&0\\ 1&0&0&0&0&0&0&1\\ 0&0&0&1&0&1&0&0 \end{pmatrix}\begin{pmatrix} a_0\oplus 1\\ a_1\oplus 1\\ a_2\\ a_3\\ a_4\\ a_5\oplus 1\\ a_6\\ a_7\oplus 1 \end{pmatrix}$$

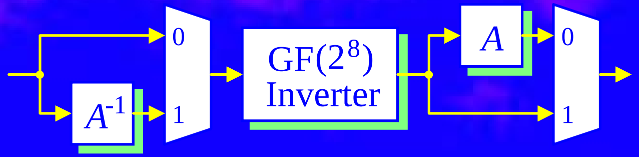
$$\begin{pmatrix} 0&1&0&0&1&1&0&0\\ 0&1&0&0&0&1&0&0\\ 0&0&0&1&0&0&1&0\\ 0&1&0&0&0&0&0&1\\ 0&0&1&0&0&0&1&0\\ 1&0&0&0&0&0&0&1\\ 1&0&0&0&1&0&0&0\\ 0&0&1&0&0&1&0&0 \end{pmatrix}\begin{pmatrix} a_0\\ a_1\\ a_2\\ a_3\\ a_4\\ a_5\\ a_6\\ a_7 \end{pmatrix}\oplus\begin{pmatrix} 0\\ 1\\ 1\\ 0\\ 1\\ 1\\ 1\\ 0 \end{pmatrix}$$

Unified S-Box

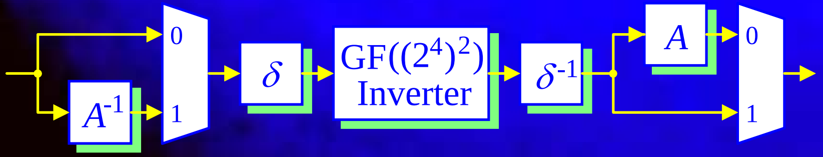
(1) AES S-Boxes



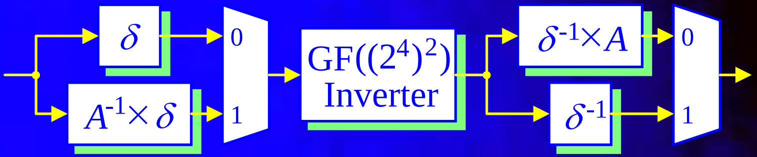
(2) Share GF(2^8) inverter



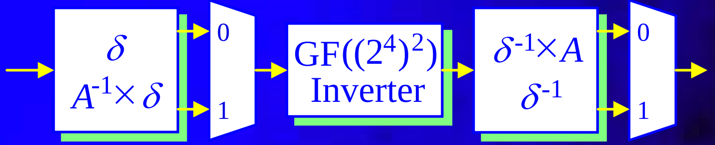
(3) Use GF($(2^4)^2$) inverter



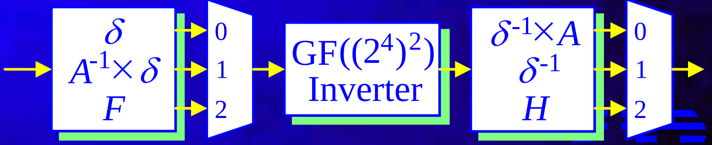
(4) Combine affine and isomorphism



(5) Share common terms

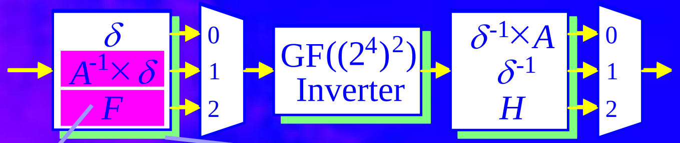


(6) Merge Camellia S-Box



Modifying affine transformations

- In order to share common terms, bit inverse operations on input are converted to the operations on output



$A^{-1}\times\delta$

$$\begin{pmatrix}1&1&0&0&0&1&1&0\\0&1&1&1&0&0&0&1\\0&1&1&1&1&0&0&0\\1&1&1&1&0&1&1&1\\1&0&1&0&0&0&0&0\\0&0&1&0&1&0&1&0\\0&1&1&0&1&1&0&0\\0&0&1&0&0&0&1&0\end{pmatrix}\begin{pmatrix}a_0\\a_1\oplus 1\\a_2\oplus 1\\a_3\\a_4\\a_5\\a_6\oplus 1\\a_7\oplus 1\end{pmatrix}$$

$$\begin{pmatrix}1&1&0&0&0&1&1&0\\0&1&1&1&0&0&0&1\\0&1&1&1&1&0&0&0\\1&1&1&1&0&1&1&1\\1&0&1&0&0&0&0&0\\0&0&1&0&1&0&1&0\\0&1&1&0&1&1&0&0\\0&0&1&0&0&0&1&0\end{pmatrix}\begin{pmatrix}a_0\\a_1\\a_2\\a_3\\a_4\\a_5\\a_6\\a_7\end{pmatrix}\oplus\begin{pmatrix}0\\1\\0\\0\\1\\0\\0\\0\end{pmatrix}$$

F

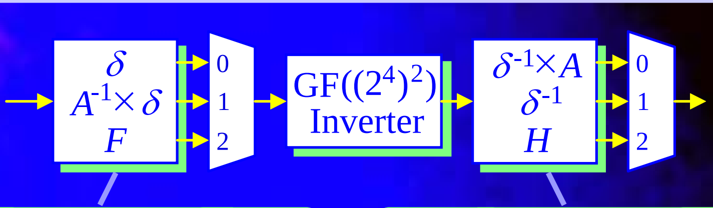
$$\begin{pmatrix}0&1&0&0&0&1&0&0\\1&0&0&0&0&0&1&0\\0&0&1&0&1&0&0&1\\0&0&1&0&0&0&0&1\\0&0&0&1&0&0&1&0\\0&1&0&0&1&0&0&0\\1&0&0&0&0&0&0&1\\0&0&0&1&0&1&0&0\end{pmatrix}\begin{pmatrix}a_0\oplus 1\\a_1\oplus 1\\a_2\\a_3\\a_4\\a_5\oplus 1\\a_6\\a_7\oplus 1\end{pmatrix}$$

$$\begin{pmatrix}0&1&0&0&0&1&0&0\\1&0&0&0&0&0&1&0\\0&0&1&0&1&0&0&1\\0&0&1&0&0&0&0&1\\0&0&0&1&0&0&1&0\\0&1&0&0&1&0&0&0\\1&0&0&0&0&0&0&1\\0&0&0&1&0&1&0&0\end{pmatrix}\begin{pmatrix}a_0\\a_1\\a_2\\a_3\\a_4\\a_5\\a_6\\a_7\end{pmatrix}\oplus\begin{pmatrix}0\\1\\1\\1\\0\\1\\0\\1\end{pmatrix}$$

Common term sharing

- Merging 6 matrices into 2 matrices
- XORs are reduced by 40%

Original	δ	20 XORs
	δ^{-1}	21 XORs
	$A^{-1}\times\delta$	22 XORs
	$\delta^{-1}\times A$	21 XORs
	F	9 XORs
	H	9 XORs
	Total	102 XORs
Merged	AES+ S1~S4	56~60 XORs



$$\begin{pmatrix} 1&0&1&0&0&0&0&0\\ 1&0&1&0&1&1&0&0\\ 1&1&0&1&0&0&1&0\\ 0&1&&\delta&&&0&0\\ 1&1&&&&&1&0\\ 0&1&0&1&0&0&1&0\\ 0&0&0&0&1&0&1&0\\ 1&1&0&1&1&1&0&1 \end{pmatrix}\begin{pmatrix} a_0\\a_1\\a_2\\a_3\\a_4\\a_5\\a_6\\a_7 \end{pmatrix}$$

$$\begin{pmatrix} 1&1&0&0&0&1&1&0\\ 0&1&1&1&0&0&0&1\\ 0&1&1&1&1&0&0&0\\ 1&1&&A^{-1}\times\delta&&&1&1\\ 1&0&&&&&0&0\\ 0&0&1&0&1&0&1&0\\ 0&1&1&0&1&1&0&0\\ 0&0&1&0&0&0&1&0 \end{pmatrix}\begin{pmatrix} a_0\\a_1\\a_2\\a_3\\a_4\\a_5\\a_6\\a_7 \end{pmatrix}\oplus\begin{pmatrix} 0\\1\\0\\0\\1\\0\\0\\0 \end{pmatrix}$$

$$\begin{pmatrix} 0&1&0&0&0&1&0&0\\ 1&0&0&0&0&0&1&0\\ 0&0&1&0&1&0&0&1\\ 0&0&&F&&&0&1\\ 0&0&&&&&1&0\\ 0&1&0&0&1&0&0&0\\ 1&0&0&0&0&0&0&1\\ 0&0&0&1&0&1&0&0 \end{pmatrix}\begin{pmatrix} a_0\\a_1\\a_2\\a_3\\a_4\\a_5\\a_6\\a_7 \end{pmatrix}\oplus\begin{pmatrix} 0\\1\\1\\1\\0\\1\\0\\1 \end{pmatrix}$$

$$\begin{pmatrix} 1&0&0&0&0&1&1&0\\ 1&1&0&1&0&0&0&0\\ 1&0&0&0&1&1&1&0\\ 0&1&&\delta^{-1}\times A&&&1&1\\ 0&0&&&&&0&1\\ 0&1&0&1&1&0&0&1\\ 1&0&0&0&1&1&1&1\\ 0&1&1&0&0&1&0&1 \end{pmatrix}\begin{pmatrix} a_0\\a_1\\a_2\\a_3\\a_4\\a_5\\a_6\\a_7 \end{pmatrix}\oplus\begin{pmatrix} 0\\1\\1\\0\\0\\0\\1\\1 \end{pmatrix}$$

$$\begin{pmatrix} 0&0&1&0&0&1&0&0\\ 1&1&1&0&1&1&1&0\\ 1&0&1&0&0&1&0&0\\ 0&1&&\delta^{-1}&&&1&0\\ 1&0&&&&&1&0\\ 0&1&1&1&0&0&1&0\\ 1&0&1&1&0&0&0&0\\ 0&1&0&1&0&0&0&1 \end{pmatrix}\begin{pmatrix} a_0\\a_1\\a_2\\a_3\\a_4\\a_5\\a_6\\a_7 \end{pmatrix}$$

$$\begin{pmatrix} 0&1&0&0&1&1&0&0\\ 0&1&0&0&0&1&0&0\\ 0&0&0&1&0&0&1&0\\ 0&1&&H&&&0&1\\ 0&0&&&&&1&0\\ 1&0&0&0&0&0&0&1\\ 1&0&0&0&1&0&0&0\\ 0&0&1&0&0&1&0&0 \end{pmatrix}\begin{pmatrix} a_0\\a_1\\a_2\\a_3\\a_4\\a_5\\a_6\\a_7 \end{pmatrix}\oplus\begin{pmatrix} 0\\1\\1\\0\\1\\1\\1\\0 \end{pmatrix}$$

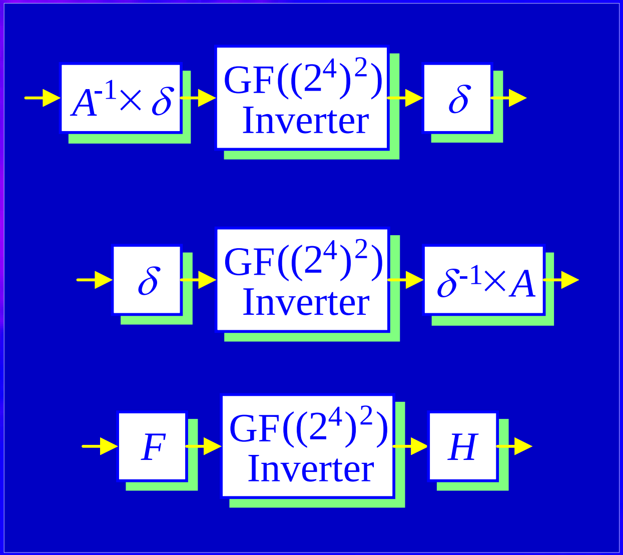
Performance of unified S-Box

- 1/2 size in comparison with discretely implemented S-Boxes
- Speed degraded by 20% due to selectors for component sharing

InvSubBytes
(AES)

SubBytes
(AES)

S1~S4
(Camellia)



280 gates
3.65 ns

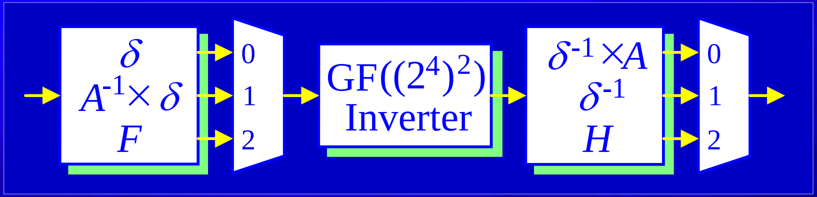
280 gates
3.56 ns

256 gates
3.45 ns

Total
816 gates

SubBytes+
InvSubBytes+
S1~S4

(AES+Camellia)



411~414 gates
4.29~4.65 ns

Unified Permutation Layer

Merging MixColumns and InvMixColumns

- Break permutation layers of AES (MixColumns and InvMixColumns) into {01, 02, 04, 08} elements
- All elements of MixColumns are included in InvMixColumns

MixColumns

$$\begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

$$\boxed{= \begin{pmatrix} 00 & 01 & 01 & 01 \\ 01 & 00 & 01 & 01 \\ 01 & 01 & 00 & 01 \\ 01 & 01 & 01 & 00 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 02 & 02 & 00 & 00 \\ 00 & 02 & 02 & 00 \\ 00 & 00 & 02 & 02 \\ 02 & 00 & 00 & 02 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}}$$

Same Matrices

InvMixColumns

$$\begin{pmatrix} \\ \\ \\ y_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

$$\boxed{= \begin{pmatrix} 00 & 01 & 01 & 01 \\ 01 & 00 & 01 & 01 \\ 01 & 01 & 00 & 01 \\ 01 & 01 & 01 & 00 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 02 & 02 & 00 & 00 \\ 00 & 02 & 02 & 00 \\ 00 & 00 & 02 & 02 \\ 02 & 00 & 00 & 02 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}}$$

$$+ \begin{pmatrix} 04 & 00 & 04 & 00 \\ 00 & 04 & 00 & 04 \\ 04 & 00 & 04 & 00 \\ 00 & 04 & 00 & 04 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 08 & 08 & 08 & 08 \\ 08 & 08 & 08 & 08 \\ 08 & 08 & 08 & 08 \\ 08 & 08 & 08 & 08 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

Merging P-function and InvMixColumns

- Generate 64-bit matrix contains two InvMixColumns
- Factor common terms between P-function and 64-bit InvMixColumns

P - function

$$\begin{pmatrix} w_0\\ w_1\\ w_2\\ w_3\\ w_4\\ w_5\\ w_6\\ w_7 \end{pmatrix} = \begin{pmatrix} 1&0&1&1&0&1&1&1\\ 1&1&0&1&1&0&1&1\\ 1&1&1&0&1&1&0&1\\ 0&1&1&1&1&1&1&0\\ 1&1&0&0&0&1&1&1\\ 0&1&1&0&1&0&1&1\\ 0&0&1&1&1&1&0&1\\ 1&0&0&1&1&1&1&0 \end{pmatrix} \cdot \begin{pmatrix} x_0\\ x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6\\ x_7 \end{pmatrix}$$

$$= \begin{pmatrix} 1&0&0&0&0&1&1&1\\ 0&1&0&0&1&0&1&1\\ 0&0&1&0&1&1&0&1\\ 0&0&0&1&1&1&1&0\\ 1&1&0&0&&&&\\ 0&1&1&0&&&0&\\ 0&0&1&1&&&&\\ 1&0&0&1&&&& \end{pmatrix} \cdot \begin{pmatrix} x_0\\ x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6\\ x_7 \end{pmatrix} + \begin{pmatrix} 0&0&1&1&&&&\\ 1&0&0&1&&&0&\\ 1&1&0&0&&&&\\ 0&1&1&0&&&&\\ &&&&0&1&1&1\\ &0&&&1&0&1&1\\ &&&&1&1&0&1\\ &&&&1&1&1&0 \end{pmatrix} \cdot \begin{pmatrix} x_0\\ x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6\\ x_7 \end{pmatrix}$$

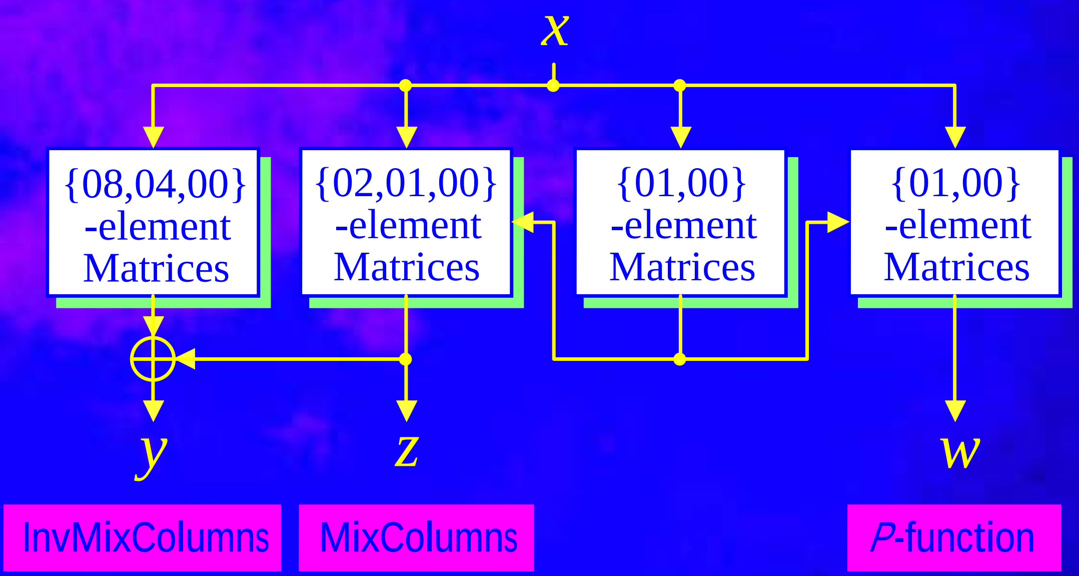
InvMixColumns

$$\begin{pmatrix} y_0\\ y_1\\ y_2\\ y_3\\ y_4\\ y_5\\ y_6\\ y_7 \end{pmatrix} = \begin{pmatrix} E&B&D&9&&&&\\ 9&E&B&D&&&0&\\ D&9&E&B&&&&\\ B&D&9&E&&&&\\ &&&&E&B&D&9\\ &0&&&9&E&B&D\\ &&&&D&9&E&B\\ &&&&B&D&9&E \end{pmatrix} \cdot \begin{pmatrix} x_0\\ x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6\\ x_7 \end{pmatrix}$$

$$= \begin{pmatrix} E&A&E&8&&&&\\ 8&E&A&E&&&0&\\ E&8&E&A&&&&\\ A&E&8&E&&&&\\ &&&&E&A&E&8\\ &0&&&8&E&A&E\\ &&&&E&8&E&A\\ &&&&A&E&8&E \end{pmatrix} \cdot \begin{pmatrix} x_0\\ x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6\\ x_7 \end{pmatrix} + \begin{pmatrix} 0&1&1&1&&&&\\ 1&0&1&1&&&0&\\ 1&1&0&1&&&&\\ 1&1&1&0&&&&\\ &&&&0&1&1&1\\ &0&&&1&0&1&1\\ &&&&1&1&0&1\\ &&&&1&1&1&0 \end{pmatrix} \cdot \begin{pmatrix} x_0\\ x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6\\ x_7 \end{pmatrix}$$

Performance of unified permutation layer

- Number of XOR terms is reduced to 1/3 of the original matrices with two additional XOR-gates of delay



x

{08,04,00} -element Matrices

{02,01,00} -element Matrices

{01,00} -element Matrices

{01,00} -element Matrices

y

z

w

InvMixColumns

MixColumns

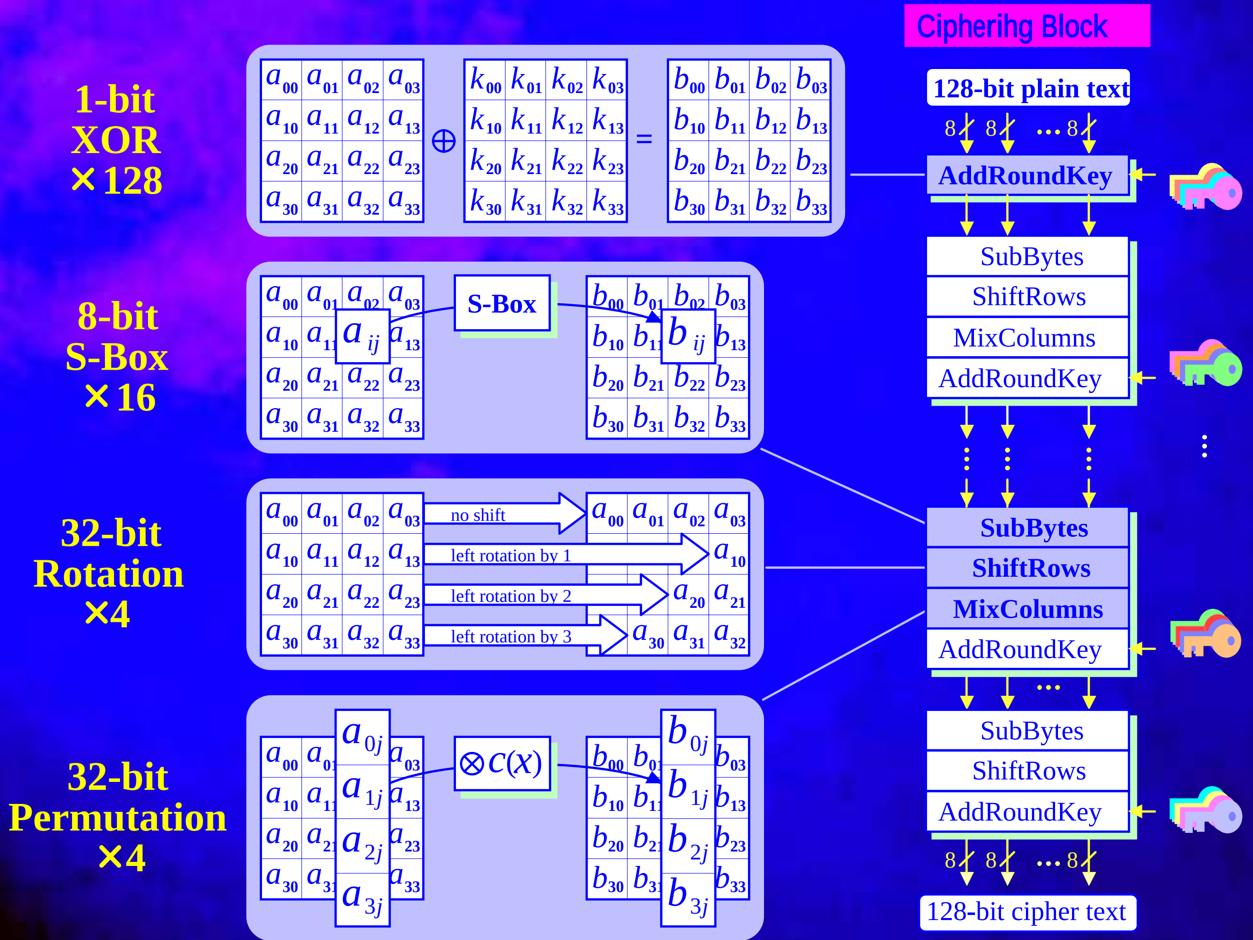
P-function

	Original Matrices				Shared Matrix
	MixCol	InvMixCol	P-funk	Total	
XORs	304	880	288	1,482	476
Delay (gates)	3	5	3	5	7

Unified Data Path Architecture

AES Algorithm

- SPN structure using 4 primitive functions takes 11 rounds



1-bit XOR ×128

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} \oplus \begin{bmatrix} k_{00} & k_{01} & k_{02} & k_{03} \\ k_{10} & k_{11} & k_{12} & k_{13} \\ k_{20} & k_{21} & k_{22} & k_{23} \\ k_{30} & k_{31} & k_{32} & k_{33} \end{bmatrix} = \begin{bmatrix} b_{00} & b_{01} & b_{02} & b_{03} \\ b_{10} & b_{11} & b_{12} & b_{13} \\ b_{20} & b_{21} & b_{22} & b_{23} \\ b_{30} & b_{31} & b_{32} & b_{33} \end{bmatrix}$$

8-bit S-Box ×16

a_{ij} → S-Box → b_{ij}

32-bit Rotation ×4

no shift

left rotation by 1

left rotation by 2

left rotation by 3

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} \rightarrow \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ & & & a_{10} \\ & & a_{20} & a_{21} \\ & a_{30} & a_{31} & a_{32} \end{bmatrix}$$

32-bit Permutation ×4

$\begin{bmatrix} a_{0j} \\ a_{1j} \\ a_{2j} \\ a_{3j} \end{bmatrix} \otimes c(x) \rightarrow \begin{bmatrix} b_{0j} \\ b_{1j} \\ b_{2j} \\ b_{3j} \end{bmatrix}$

Cipherihg Block

128-bit plain text

8 8 … 8

AddRoundKey

SubBytes

ShiftRows

MixColumns

AddRoundKey

⋮

SubBytes

ShiftRows

MixColumns

AddRoundKey

…

SubBytes

ShiftRows

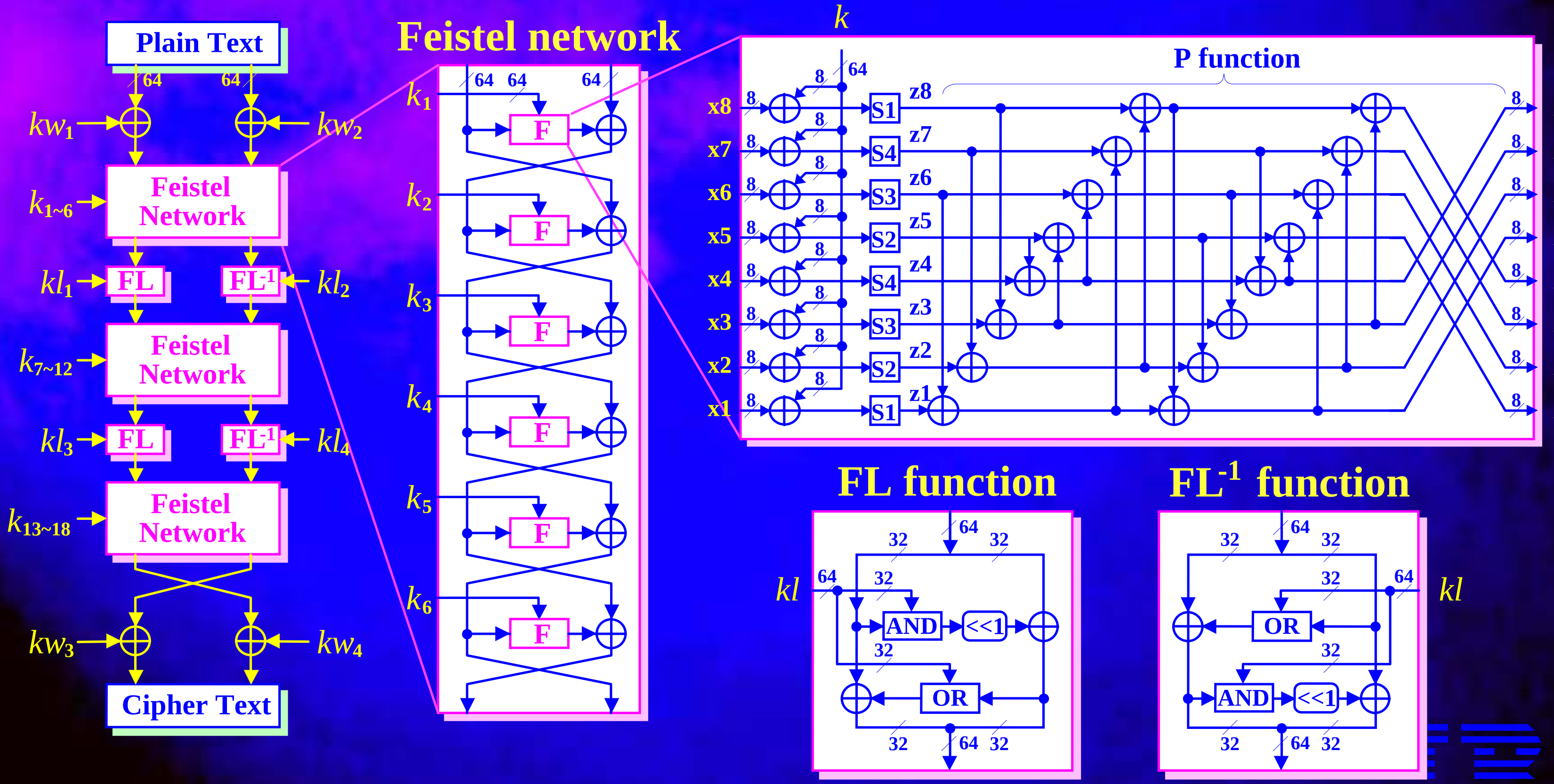
AddRoundKey

8 8 … 8

128-bit cipher text

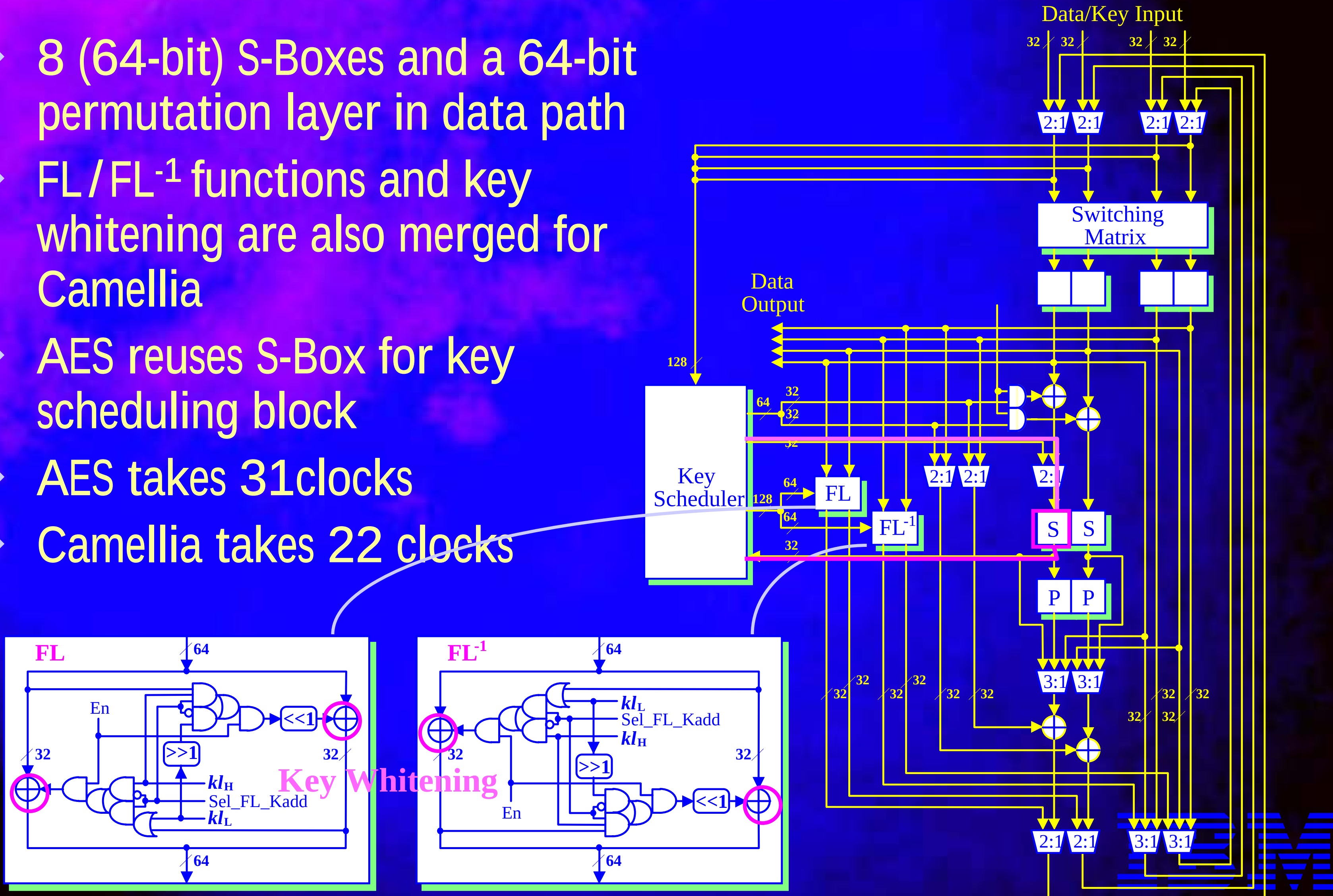
Camellia Algorithm

- 2 FL / FL^{-1} functions are inserted between 3 Feistel network blocks
- It takes 22 rounds by processing 64 bits in each round



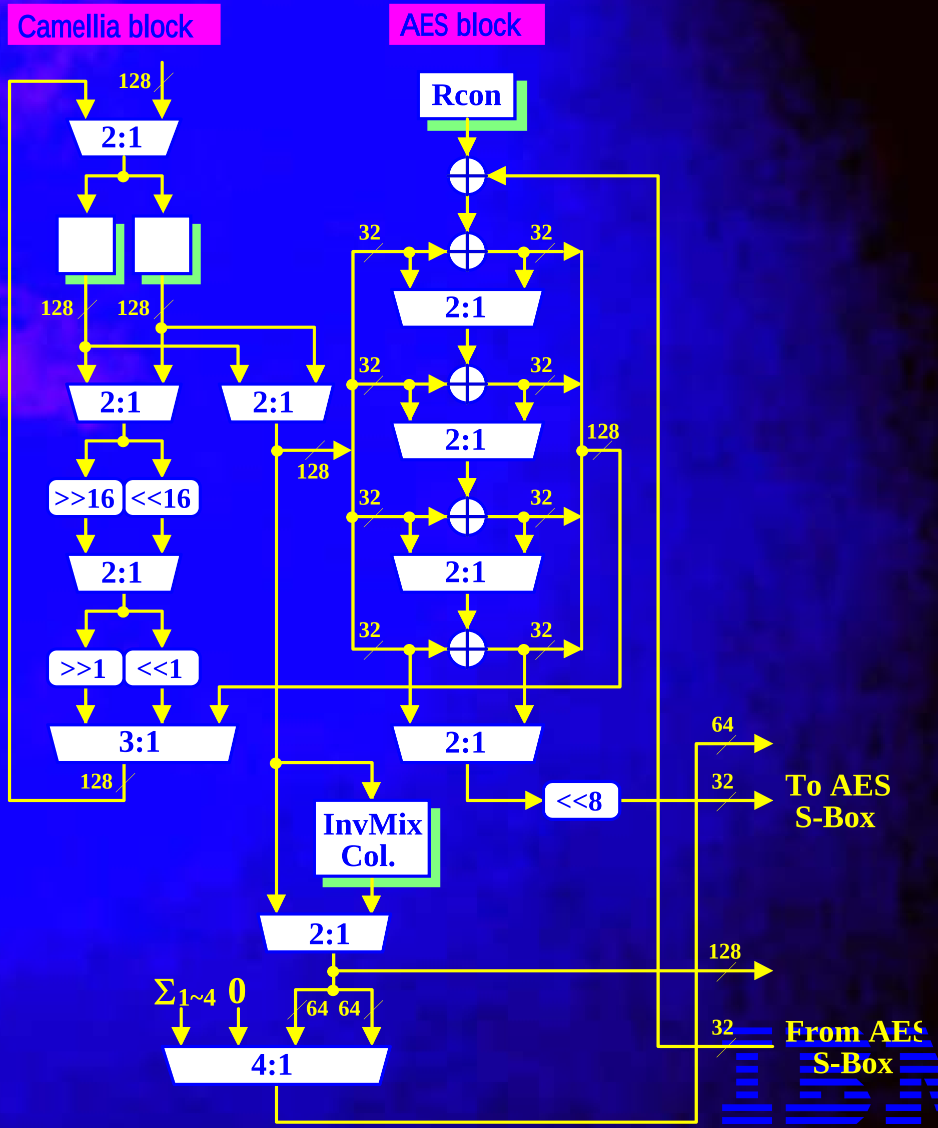
Ciphering block

- 8 (64-bit) S-Boxes and a 64-bit permutation layer in data path
- FL / FL^{-1} functions and key whitening are also merged for Camellia
- AES reuses S-Box for key scheduling block
- AES takes 31clocks
- Camellia takes 22 clocks



Key scheduler

- AES repeats XOR operations and Camellia repeats bit rotations
- Only registers can be shared because key scheduling methods are completely different



ASIC Implementation Results

ASIC Implementation Results

- Two circuits (area optimized and speed optimized) were synthesized by using 0.13um CMOS ASIC library
- 15~25 Kgates of unified hardware are smaller by 30% compared with 22~35 Kgates of discrete designs
- Throughputs are lower by 9~14% for AES, and by 31~40% for Camellia

	Algorithm	Cycle	Gate Counts	Max. Freq. (MHz)	Throughput (Mbps)	Synthesis Optimization
This Work	AES	31	14,918	113.64	469.22	Area
	Camellia	22			661.18	
	AES	31	24,730	192.31	794.05	Speed
	Camellia	22			1,118.89	
ASIACRYPT 2001	AES	32	7,998	153.85	548.68	Area
			14,777	218.82	875.28	Speed
3rd NESSIE	Camellia	18	13,557	153.85	1,094.04	Area
			19,783	227.27	1,616.14	Speed

Conclusions

Conclusions

- Unified hardware architecture for AES and Camellia
  - Share GF($(2^4)^2$) inverters in all S-Boxes
  - Merge affine transformations and isomorphism functions
  - Factorize common terms in permutation layers
  - Combine FL / FL^{-1} functions and key whitening
  - Reuse S-Boxes in ciphering block for key scheduler
  - Share key registers
- Implemented by using 0.13-um CMOS ASIC library
  - 15 Kgates with 459 Mbps (AES) and 647 Mbps (Camellia)
  - Smaller than discrete implementations by 30%