Very Compact FPGA Implementations of the AES Algorithm

Pawel Chodowiec and Kris Gaj

George Mason University

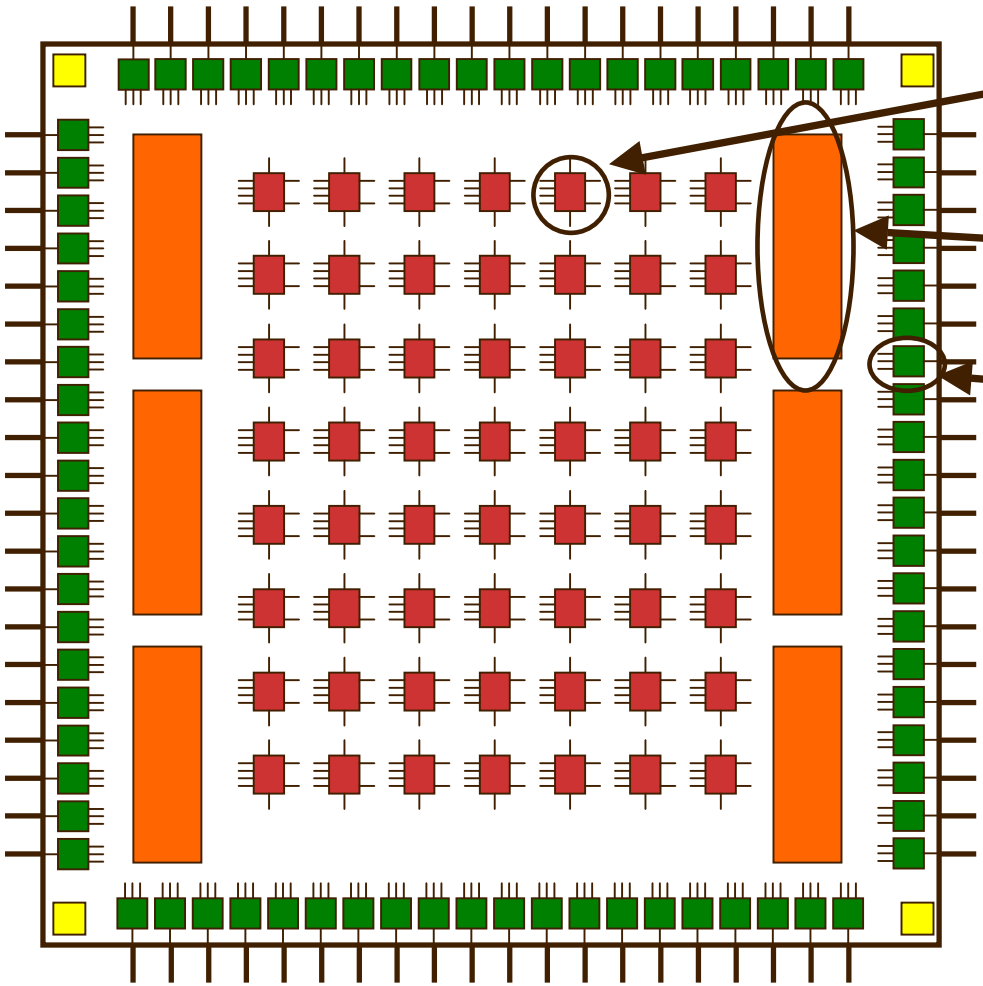
Why Yet Another AES Implementation?

- Most of the published FPGA implementations target only high-end products
  - Multi-Gigabit throughputs
  - FPGA device costs reach $1,000 per single unit
  - No regards for power consumption
- AES needs to be deployed in low-end products as well (ATM, pay TV, wireless communication, PDA etc.)
  - Rarely need more than 100Mbps
  - Low cost is a must
  - Low power consumption is very welcome

Low-end Design Objectives

- Support for encryption, decryption and key schedule in one circuit
- Feedback modes of operation supported with no penalty on performance
- Encryption/decryption speed not smaller than 100Mbps
- Minimum FPGA device cost
  - Target low-cost Xilinx Spartan-II family (less than $10 per unit)
- Minimum power consumption
  - Minimize circuit size
  - Minimize circuit activity

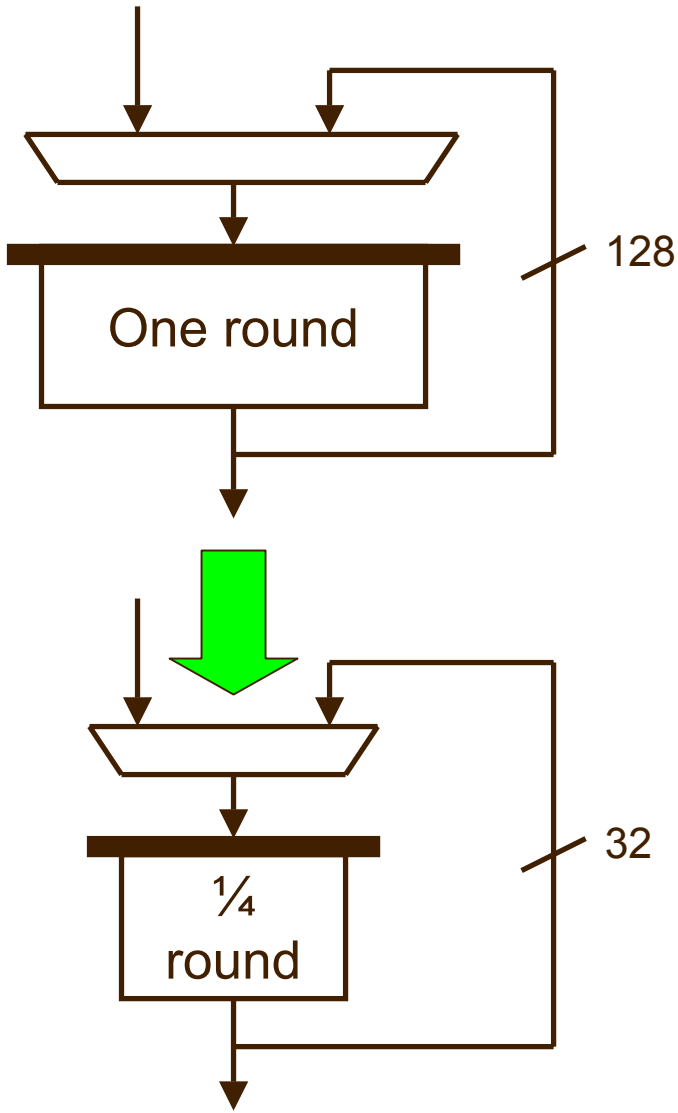
Spartan-II Family Architecture



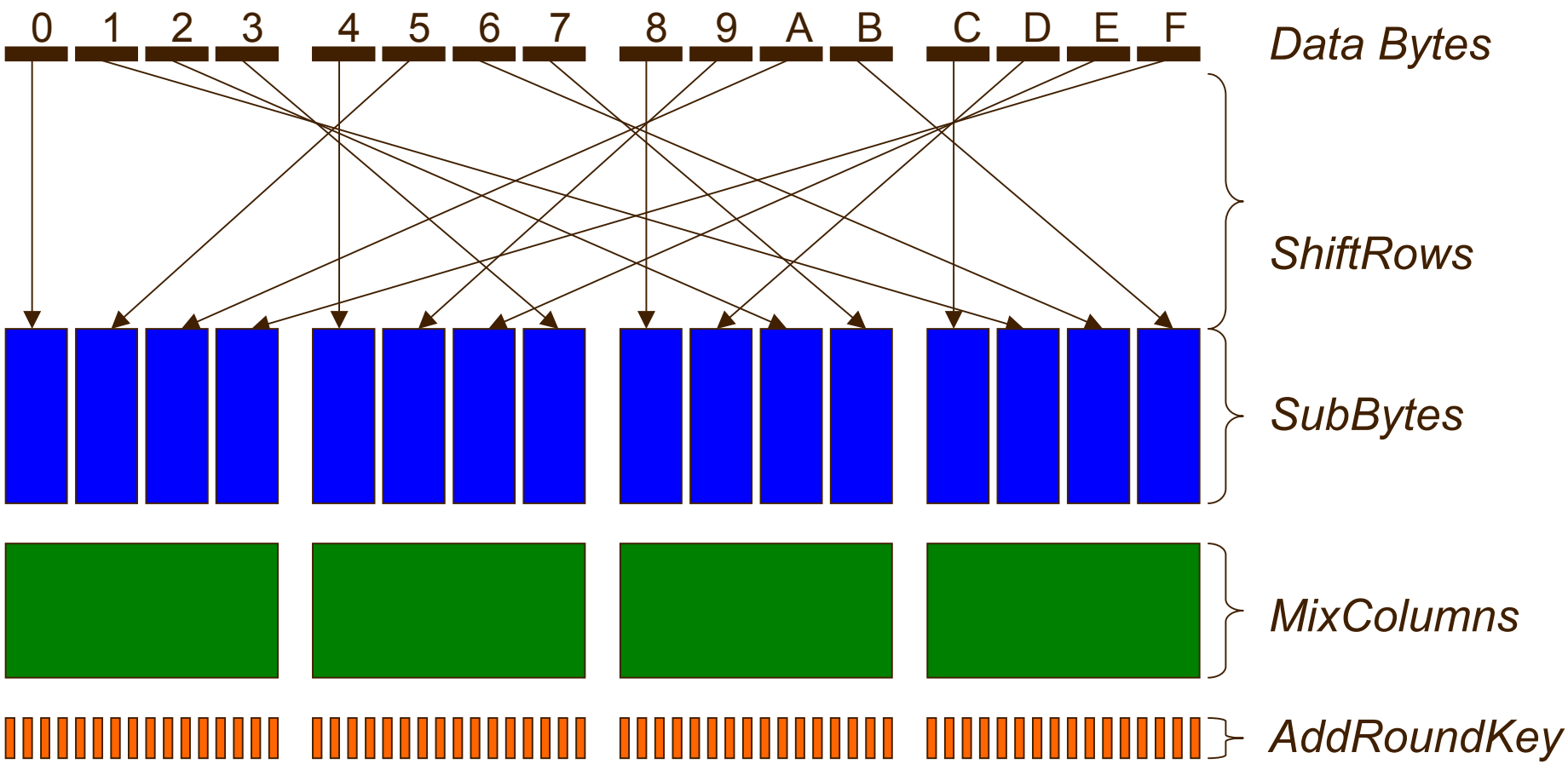
- 0.22µm CMOS process
- Up to 2352 Slices
- Up to 14 4kbit BlockRAMs
- Cost per unit < $10

AES Compact Design Approach

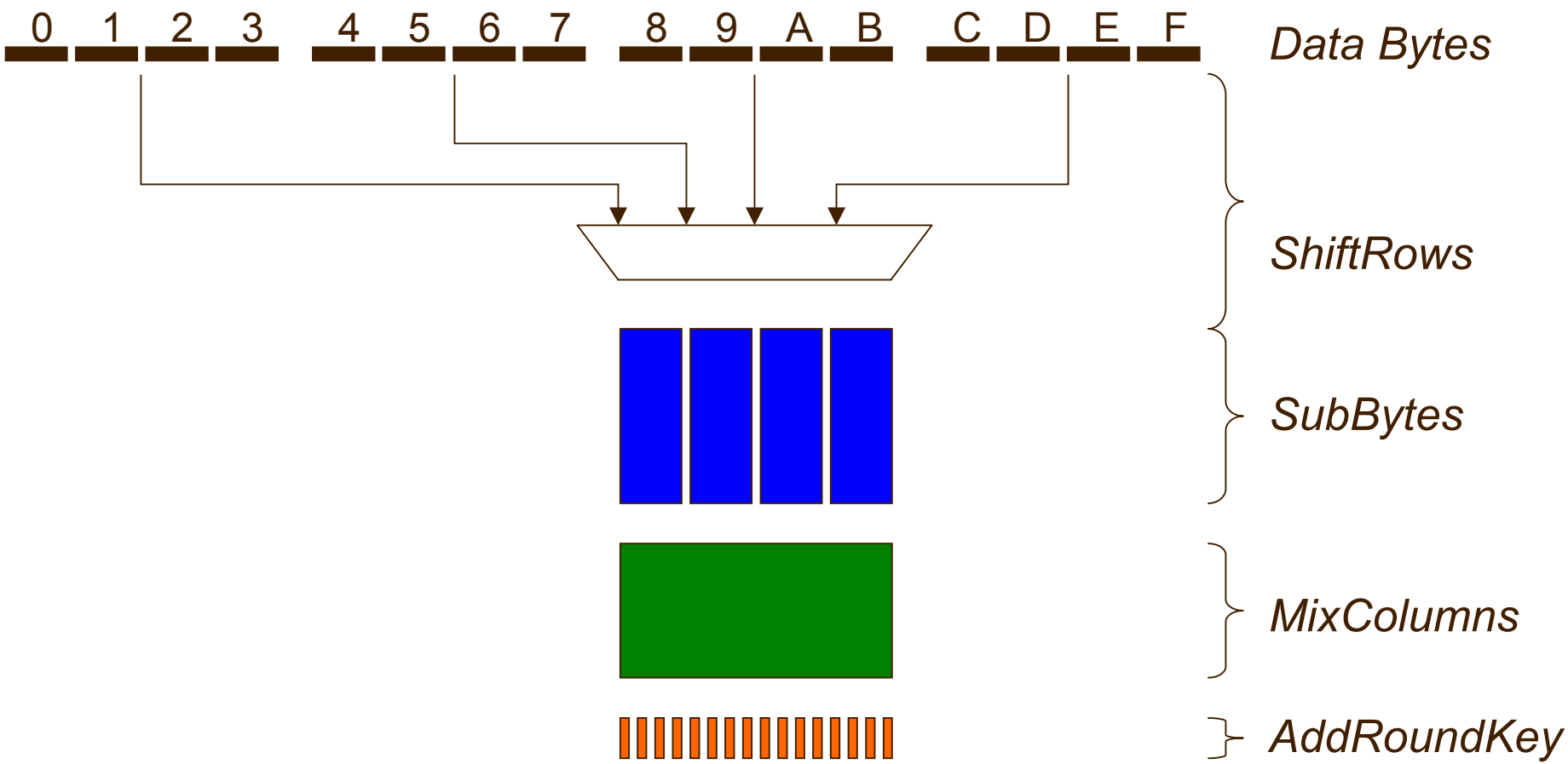
1. Start with iterative architecture
2. Fold the iterative architecture to minimize circuit area
   - In this case make folded architecture 4 times smaller



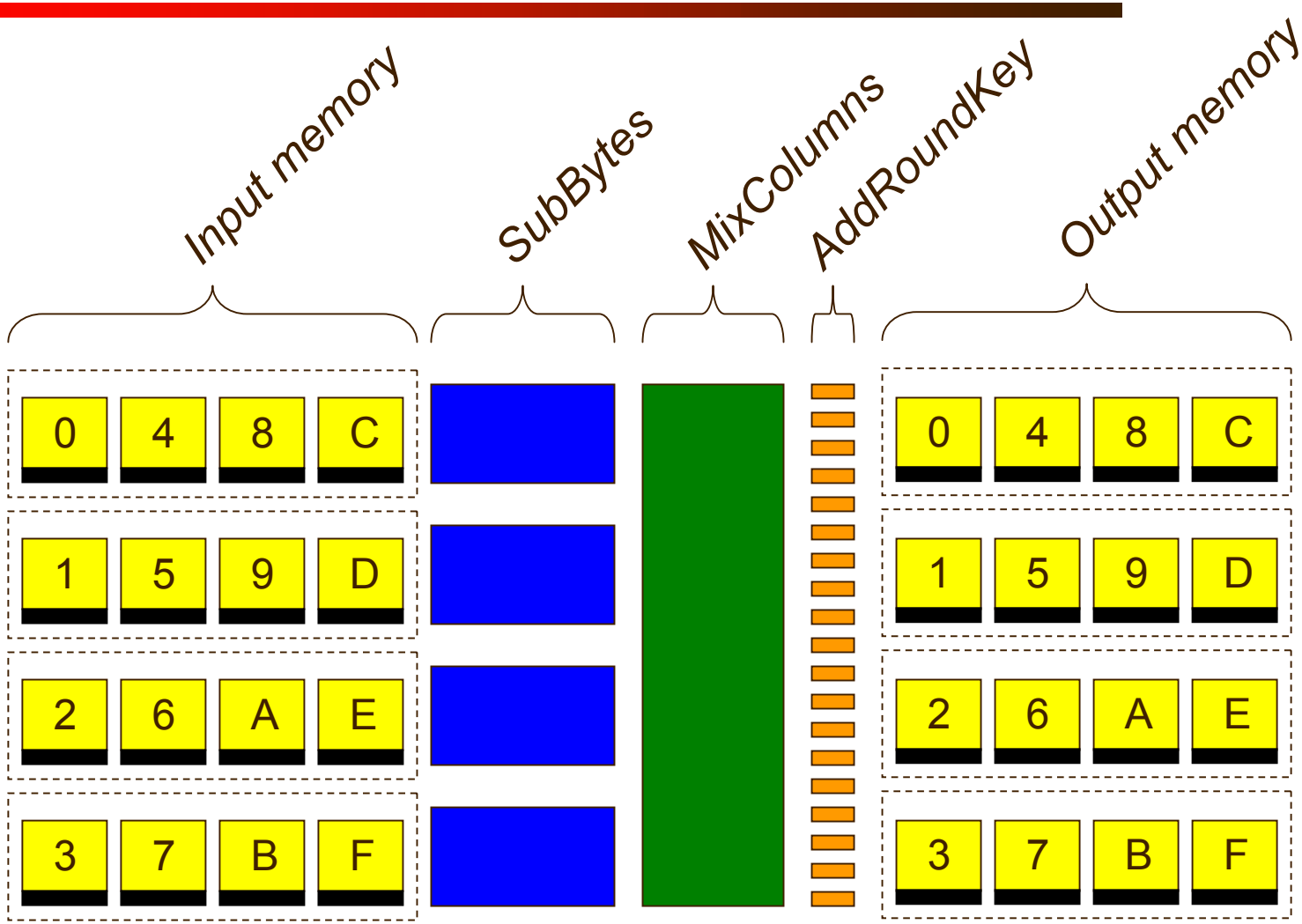
AES Encryption Round



What Makes Folding Non-Trivial?



Folding the Register (1)



Note that ShiftRows is executed with no additional routing!

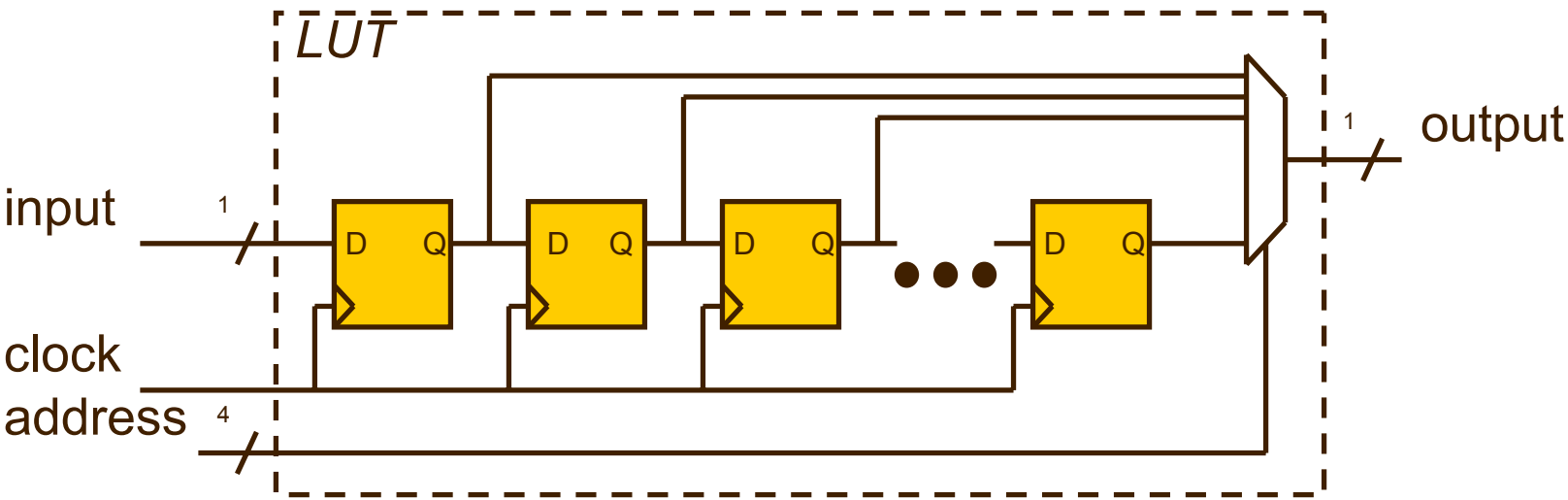
Folding the Register (2)

- Input and output memories are exchanged every fourth clock cycle
  - In practice both memories are the same memory with ability to read and write simultaneously (Dual-Port Memory)
  - Lookup Tables (LUT) inside each CLB can be configured as Dual-Port Memories!

- Note: data get written into consecutive locations in the output memory
  - A shift-register could be used instead…

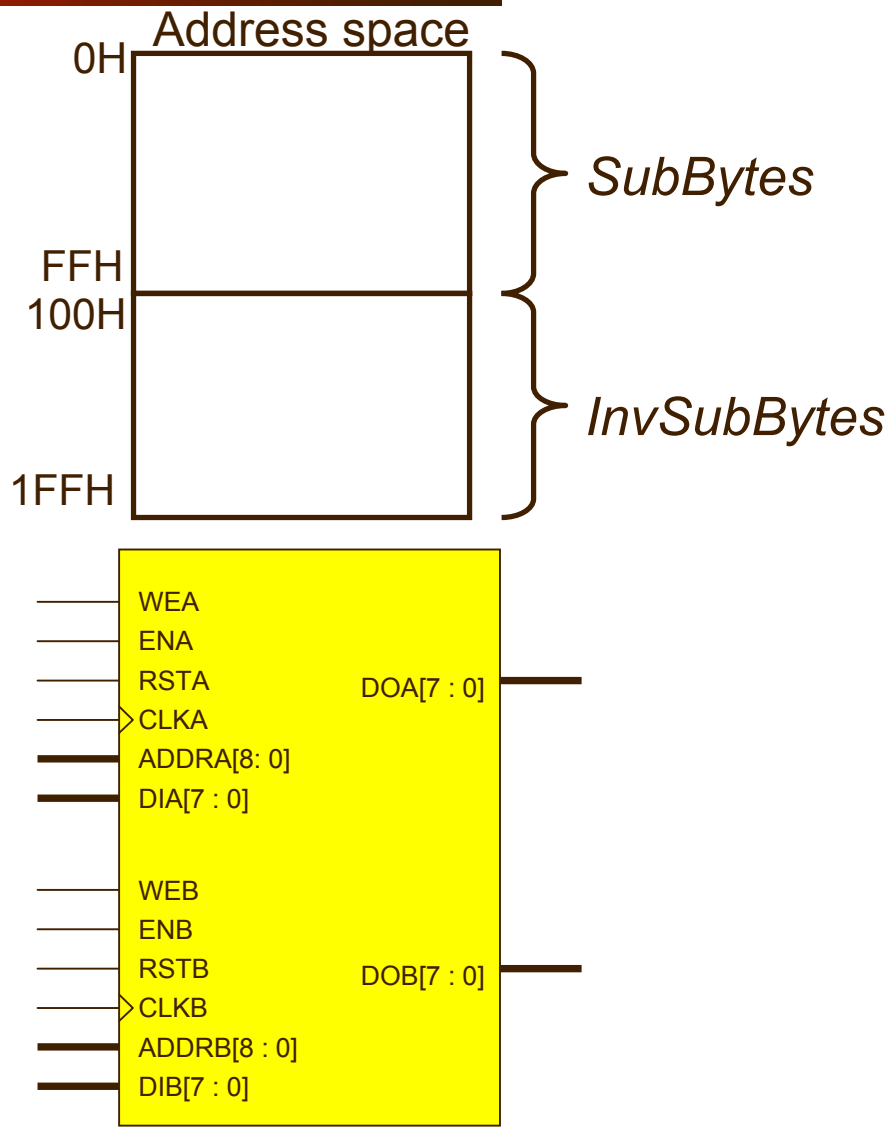
Folding the Register (3)

- A LUT can be configured as a 16-deep shift-register!



Implementation of SubBytes

- Spartan-II devices contain 4kbit BlockSelect RAMs
  - One SubBytes could be implemented using 2kbit memory
- Each BlockSelect RAM is a Dual-Port Memory with independent access
- Only 2 BlockSelect RAMs are required to implement 4 SubBytes and 4 InvSubBytes



Decomposition of InvMixColumns (1)

Let:

$$c(x) = \text{MixColumns}$$
$$d(x) = \text{InvMixColumns}$$

Following property holds:

$$\{01\} = c(x) \bullet d(x)$$

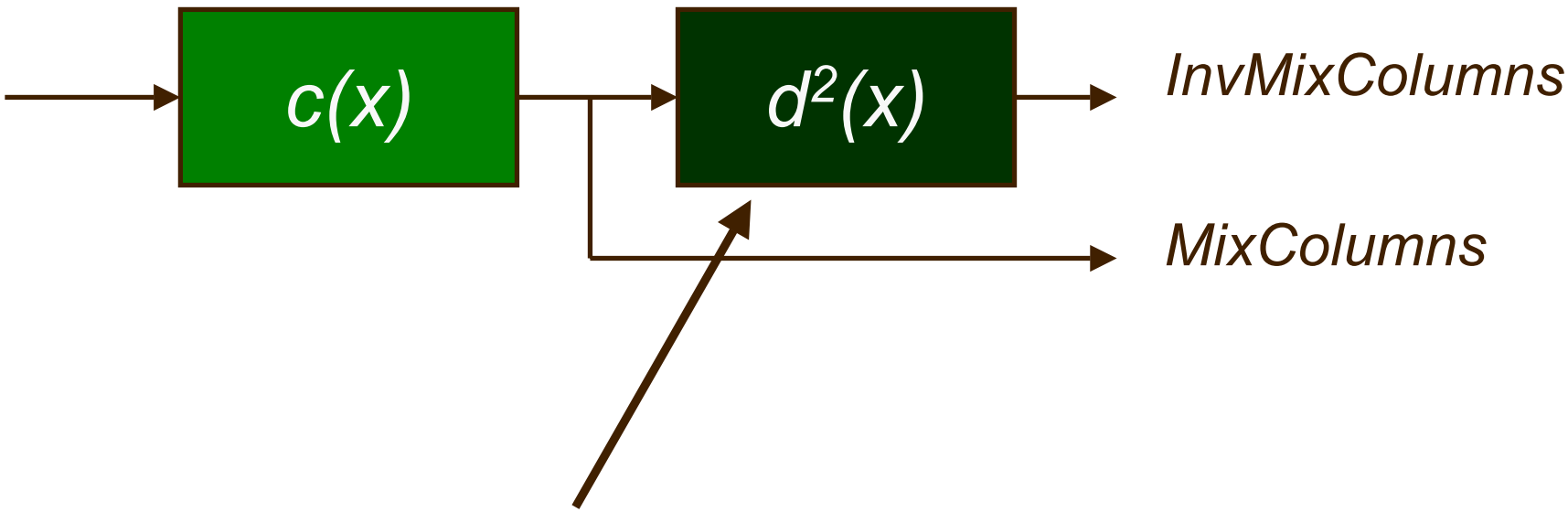
Multiply both sides by d(x):

$$d(x) = c(x) \bullet d^2(x)$$

InvMixColumns can be expressed as a product of MixColumns and $d^2(x)$, where

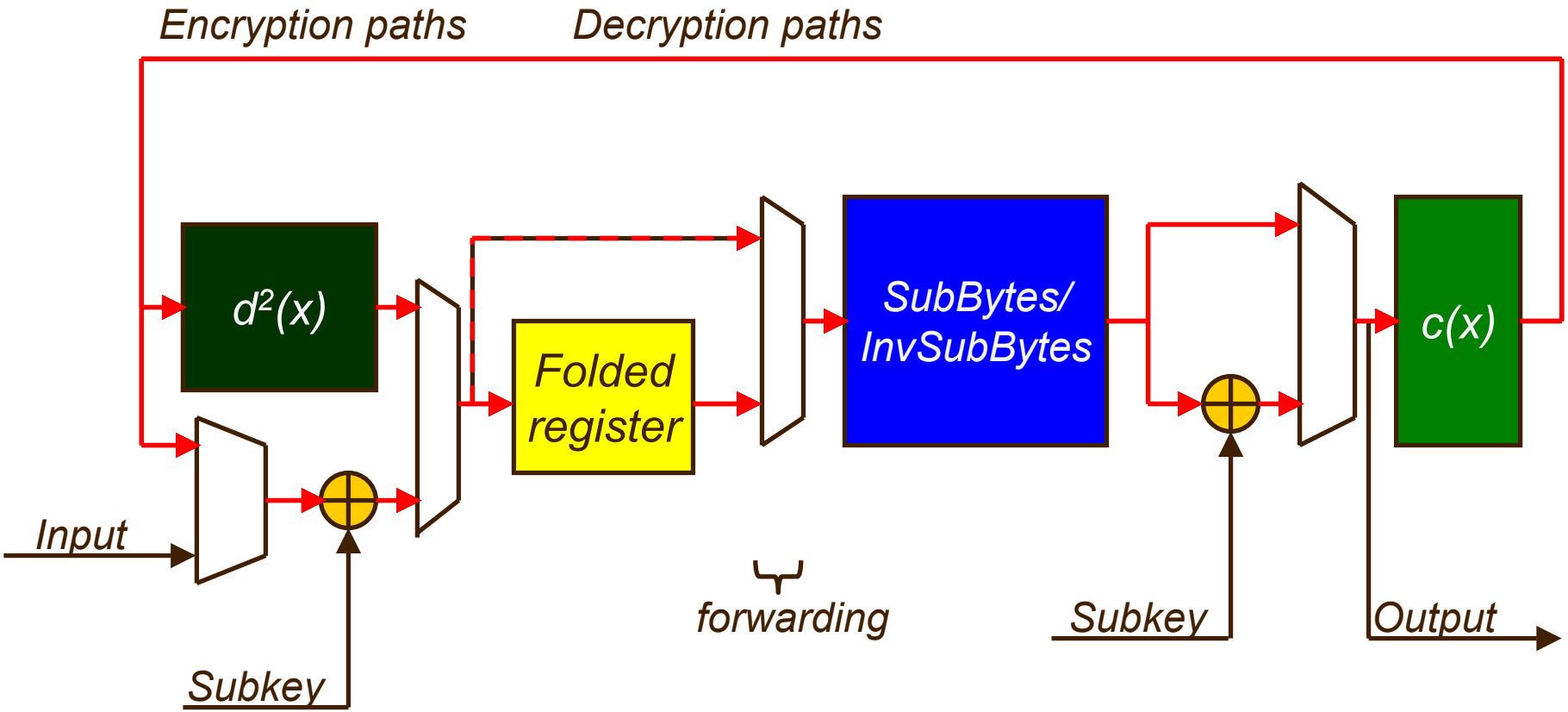
$$d^2(x) = \{04\}x^2 + \{05\}$$

Decomposition of InvMixColumns (2)

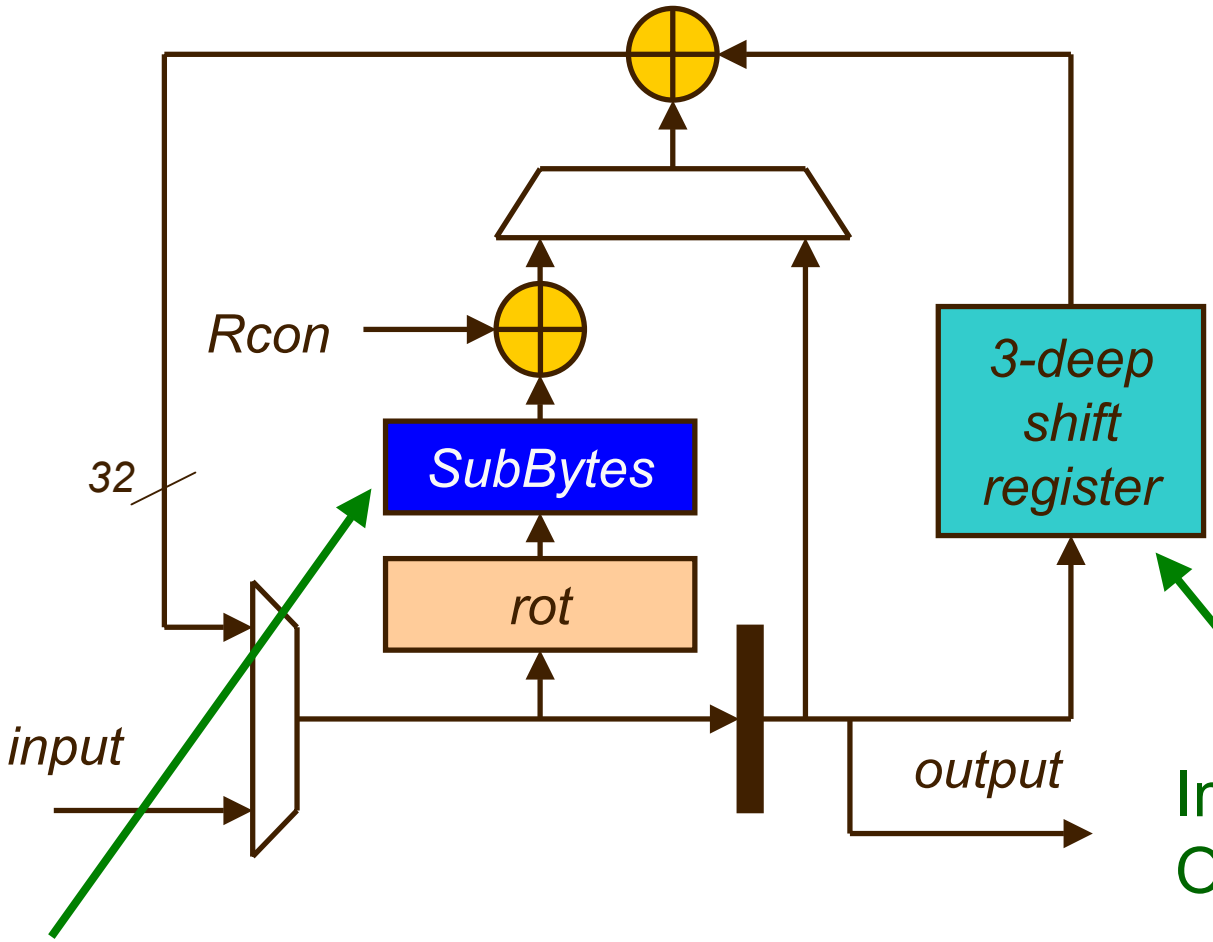


Implementation of $d^2(x)$ requires even smaller area than c(x)!

Encryption/Decryption Unit



Implementation of Key Schedule

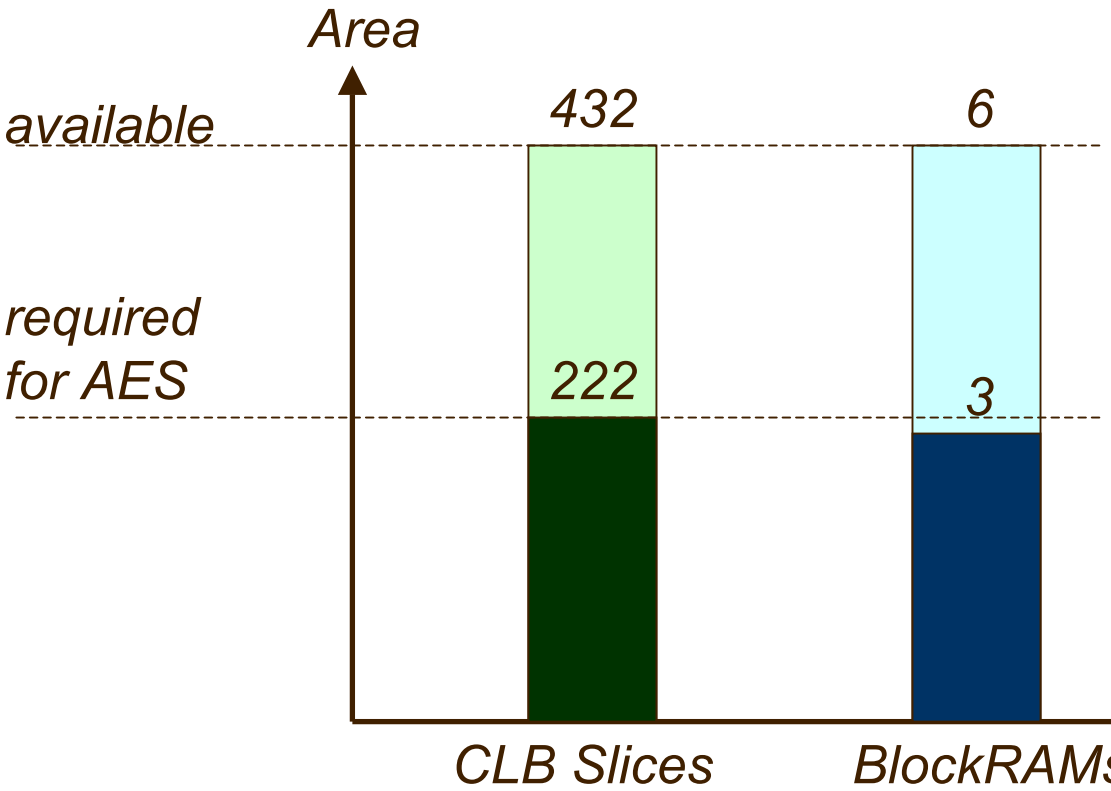


Implemented
On LUTs

SubBytes shares BlockSelect RAM
with encryption/decryption unit

Implementation Results

- The entire design fits in a single Spartan-II XC2S30, second smallest in the Spartan-II family



- Nearly 50% of the device available for other logic
- Throughput: **174Mbps** at 60MHz clock frequency

Comparison with Other Compact Arch.

	Area		Throughput [Mbps]
	CLB Slices	Block RAMs	
0.22µm			
Our	**222**	**3**	**174**
S. McMillan	240	8	250
0.18µm			
Amphion CS5220	421	4	294
Helion compact	392 LUTs	3	223
0.15µm			
Amphion CS5220	403	4	350

Comparison with Iterative Arch.

	Area		Throughput [Mbps]
	CLB Slces	Block RAMs	
0.22µm			
Our	**222**	**3**	**174**
P. Chodowiec	1230	18	577
A. Dandalis	5673	0	353
A. Elbirt	3528	0	294.2
V. Fisher - FLEX	2530 LE	24 EAB	451
V. Fisher - ACEX	2923 LE	12 EAB	212
K. Gaj	2507	0	414
0.18µm			
Amphion CS5230	**573**	**10**	**1061**
V. Fisher - APEX	2493 LE	50 ESB	612
Helion fast	2259 LUT	18	1001
0.15µm			
Amphion CS5230	573	10	1323
Helion fast	2259 LUT	18	1408

Conclusions

- We presented a successful method for a compact AES implementation in FPGAs
- The area requirements of the compact design are smaller than the ¼ of the area of the smallest iterative architecture in the same technology
- The speed of our design is higher than the ¼ of the speed of the fastest iterative architecture in the same technology
- The design avoids complicated routing associated with implementation of ShiftRows operation