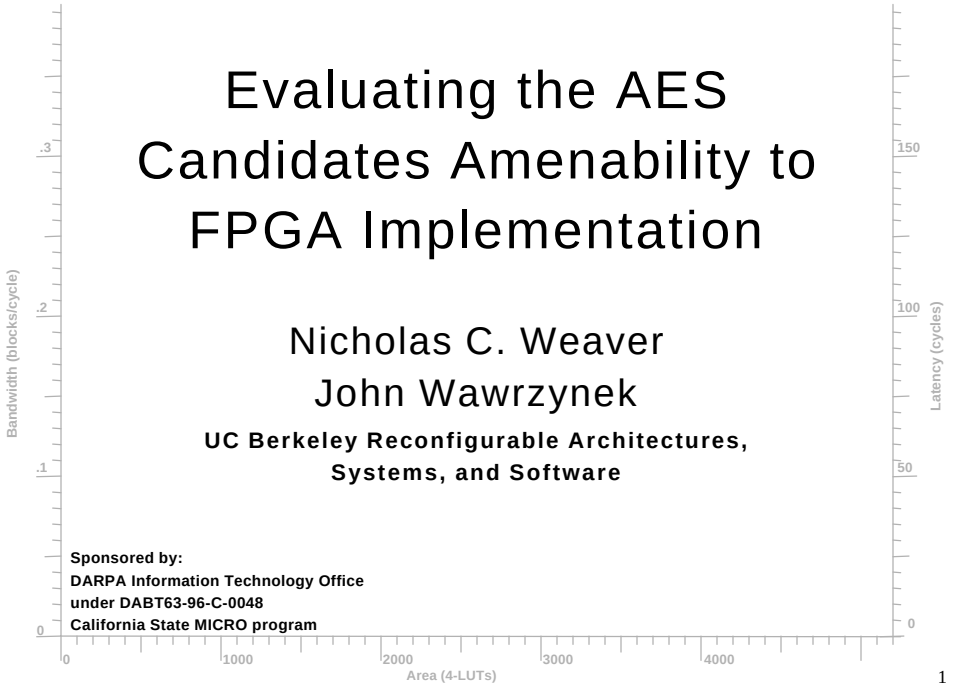
Evaluating the AES Candidates Amenability to FPGA Implementation

Nicholas C. Weaver
John Wawrzynek

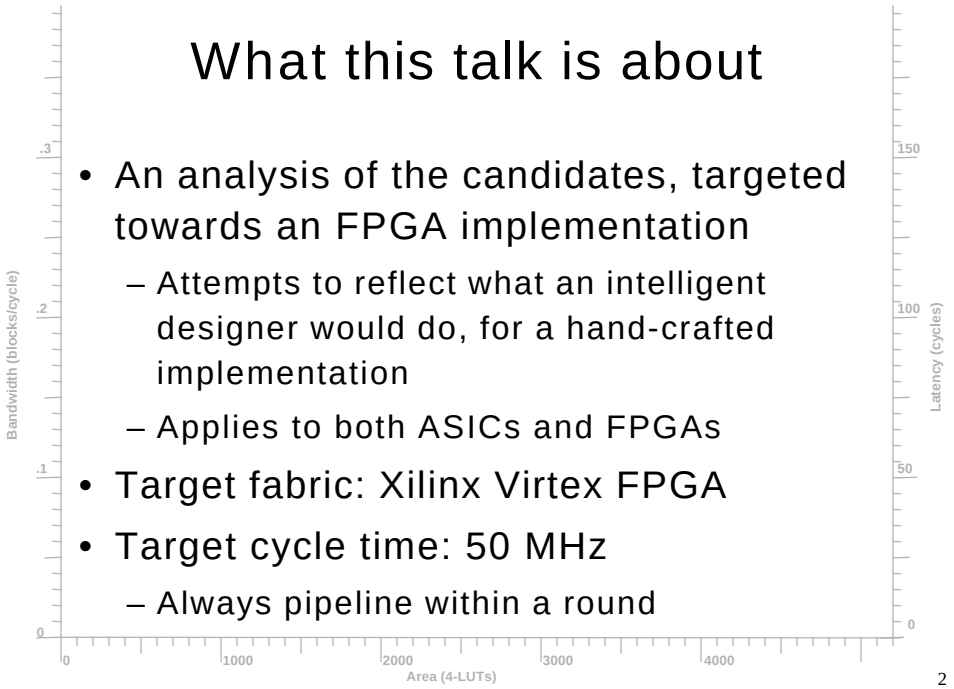
UC Berkeley Reconfigurable Architectures, Systems, and Software

Sponsored by:
DARPA Information Technology Office
under DABT63-96-C-0048
California State MICRO program



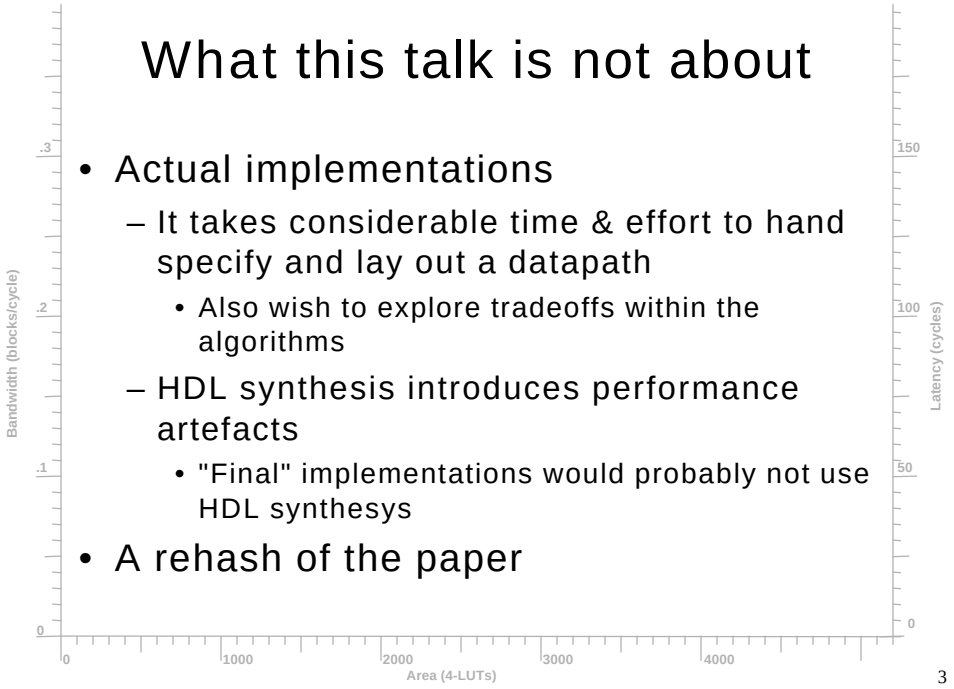
What this talk is about

- An analysis of the candidates, targeted towards an FPGA implementation
  - Attempts to reflect what an intelligent designer would do, for a hand-crafted implementation
  - Applies to both ASICs and FPGAs
- Target fabric: Xilinx Virtex FPGA
- Target cycle time: 50 MHz
  - Always pipeline within a round



What this talk is not about

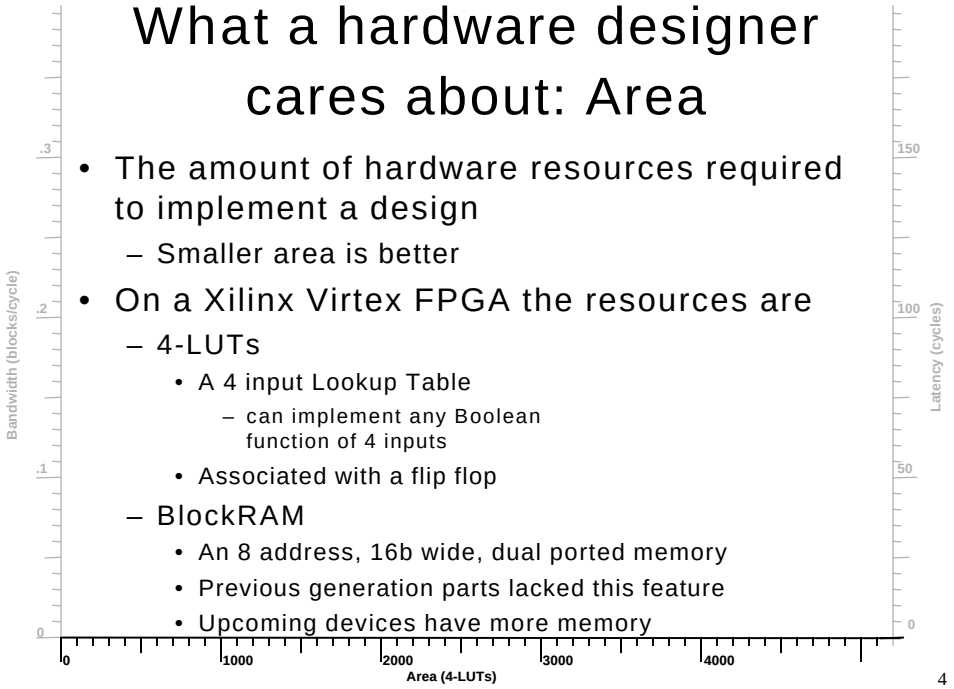
- Actual implementations
  - It takes considerable time & effort to hand specify and lay out a datapath
    - Also wish to explore tradeoffs within the algorithms
  - HDL synthesis introduces performance artefacts
    - "Final" implementations would probably not use HDL synthesys
- A rehash of the paper



3

What a hardware designer cares about: Area

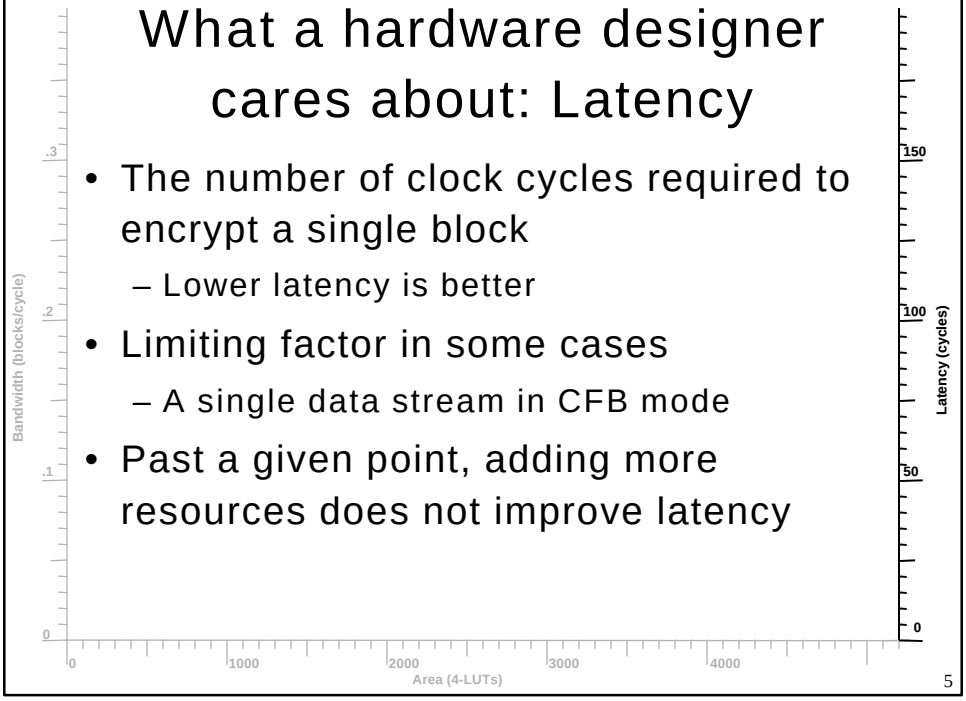
- The amount of hardware resources required to implement a design
  - Smaller area is better
- On a Xilinx Virtex FPGA the resources are
  - 4-LUTs
    - A 4 input Lookup Table
      - can implement any Boolean function of 4 inputs
    - Associated with a flip flop
  - BlockRAM
    - An 8 address, 16b wide, dual ported memory
    - Previous generation parts lacked this feature
    - Upcoming devices have more memory



4

What a hardware designer cares about: Latency

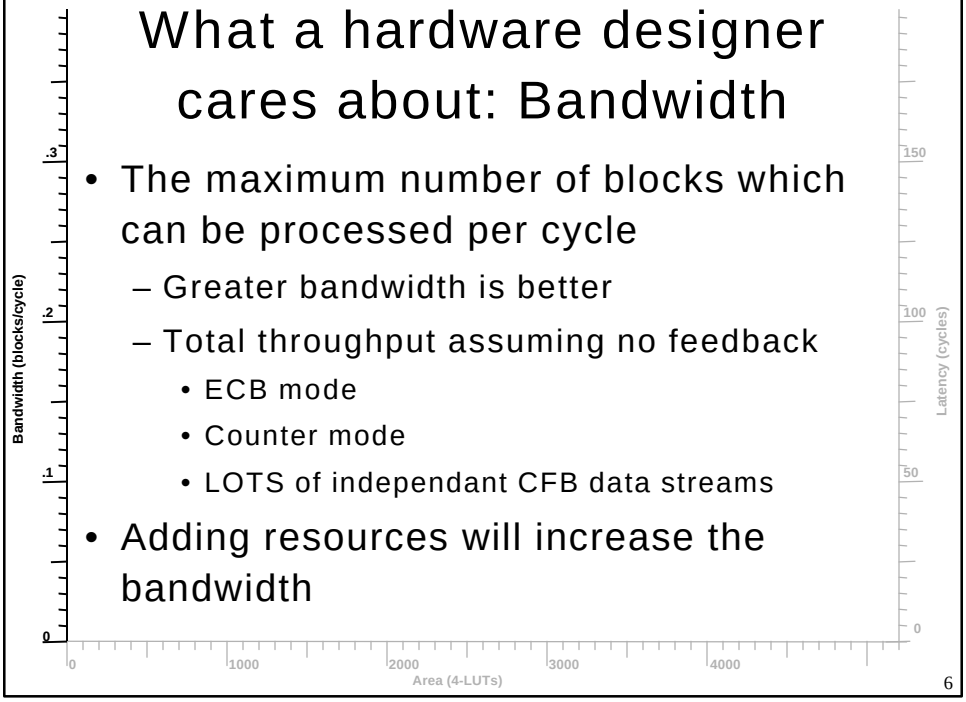
- The number of clock cycles required to encrypt a single block
  - Lower latency is better
- Limiting factor in some cases
  - A single data stream in CFB mode
- Past a given point, adding more resources does not improve latency



5

What a hardware designer cares about: Bandwidth

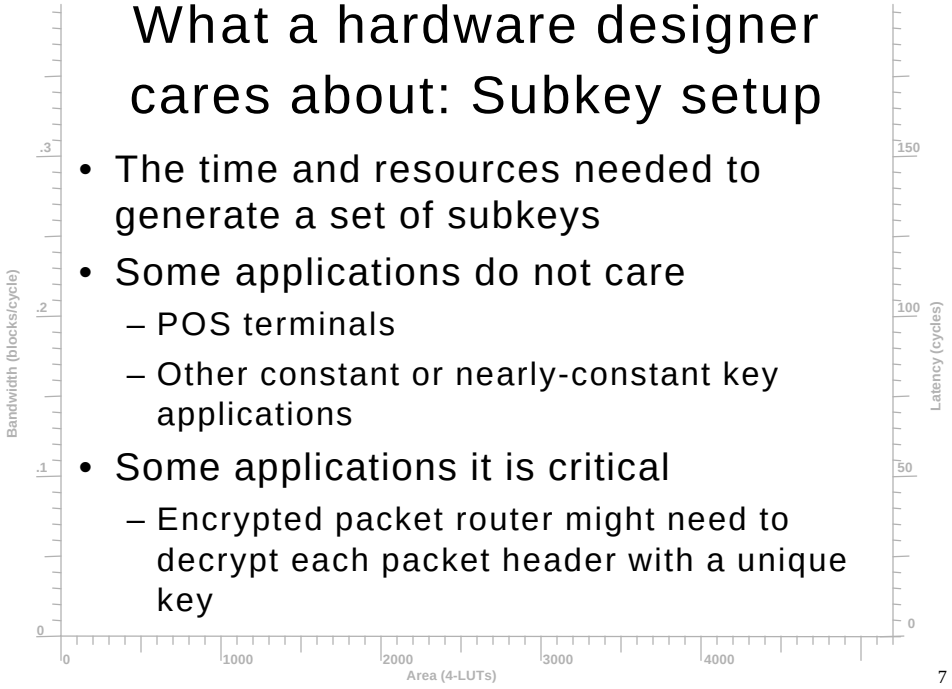
- The maximum number of blocks which can be processed per cycle
  - Greater bandwidth is better
  - Total throughput assuming no feedback
    - ECB mode
    - Counter mode
    - LOTS of independant CFB data streams
- Adding resources will increase the bandwidth



6

What a hardware designer cares about: Subkey setup

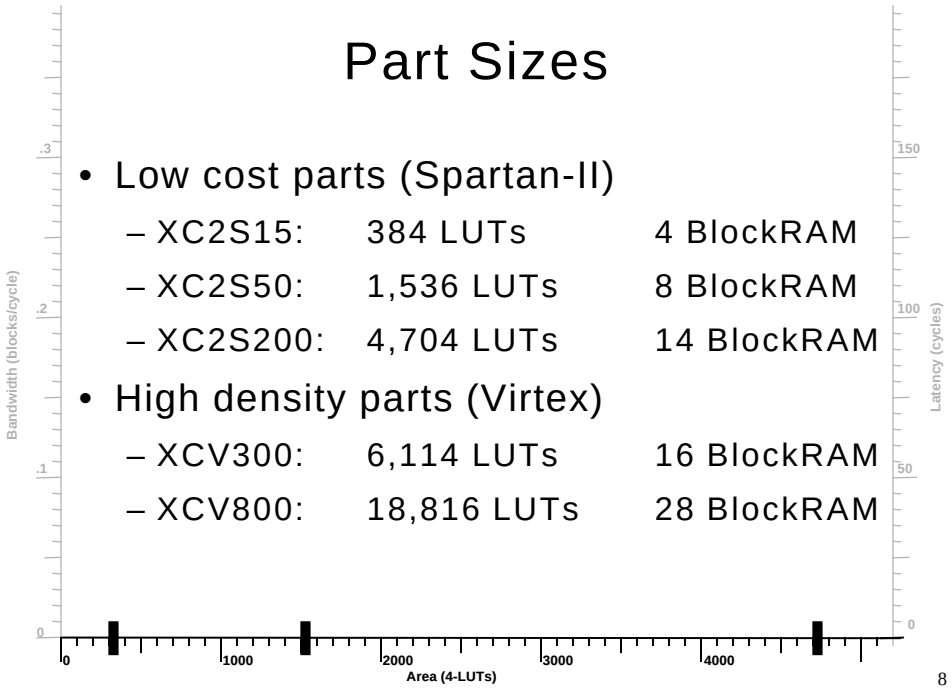
- The time and resources needed to generate a set of subkeys
- Some applications do not care
  - POS terminals
  - Other constant or nearly-constant key applications
- Some applications it is critical
  - Encrypted packet router might need to decrypt each packet header with a unique key



7

Part Sizes

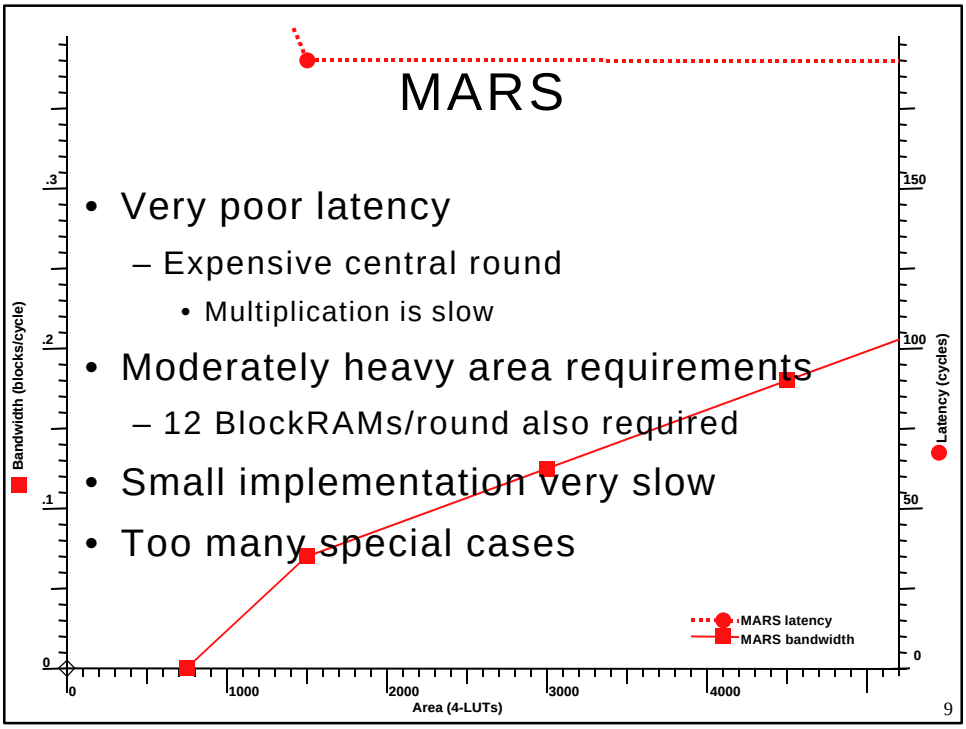
- Low cost parts (Spartan-II)
  - XC2S15: 384 LUTs 4 BlockRAM
  - XC2S50: 1,536 LUTs 8 BlockRAM
  - XC2S200: 4,704 LUTs 14 BlockRAM
- High density parts (Virtex)
  - XCV300: 6,114 LUTs 16 BlockRAM
  - XCV800: 18,816 LUTs 28 BlockRAM



8

MARS

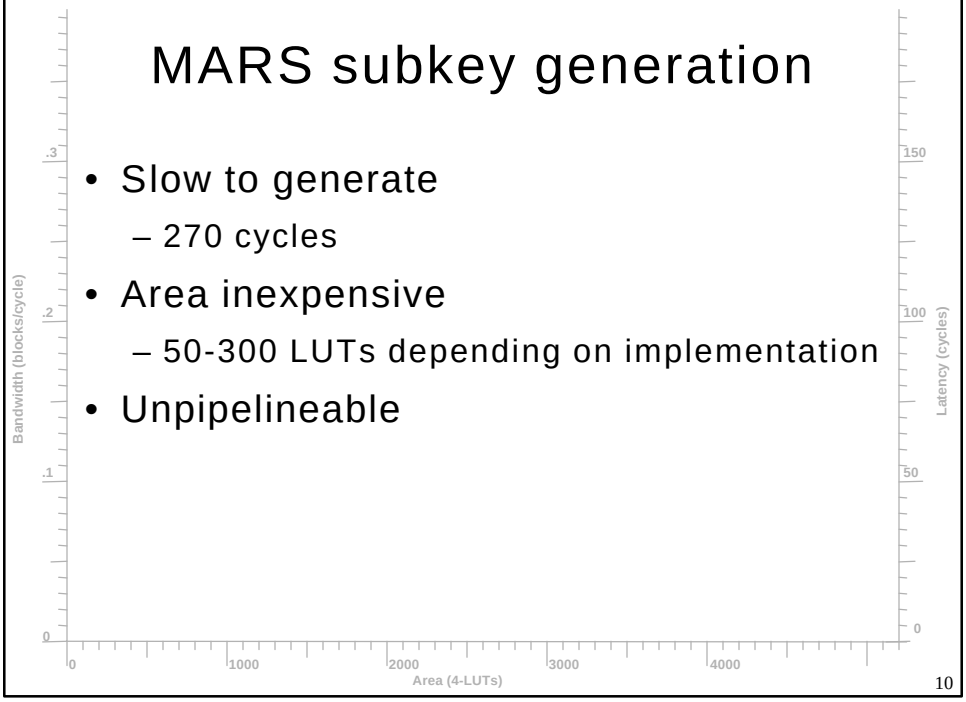
- Very poor latency
  - Expensive central round
    - Multiplication is slow
- Moderately heavy area requirements
  - 12 BlockRAMs/round also required
- Small implementation very slow
- Too many special cases



9

MARS subkey generation

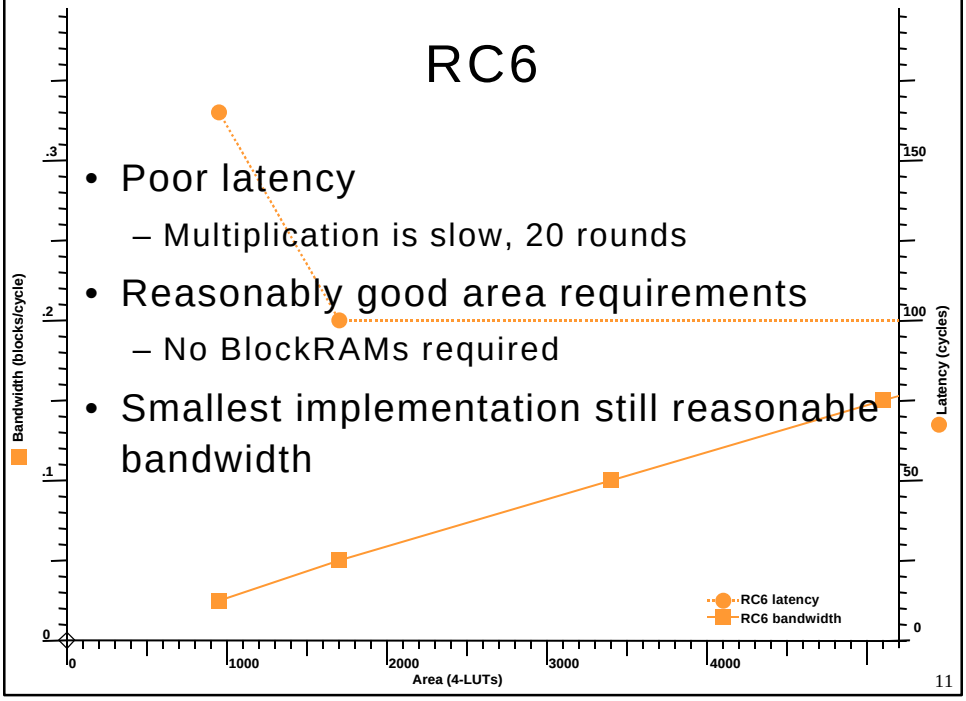
- Slow to generate
  - 270 cycles
- Area inexpensive
  - 50-300 LUTs depending on implementation
- Unpipelineable



10

RC6

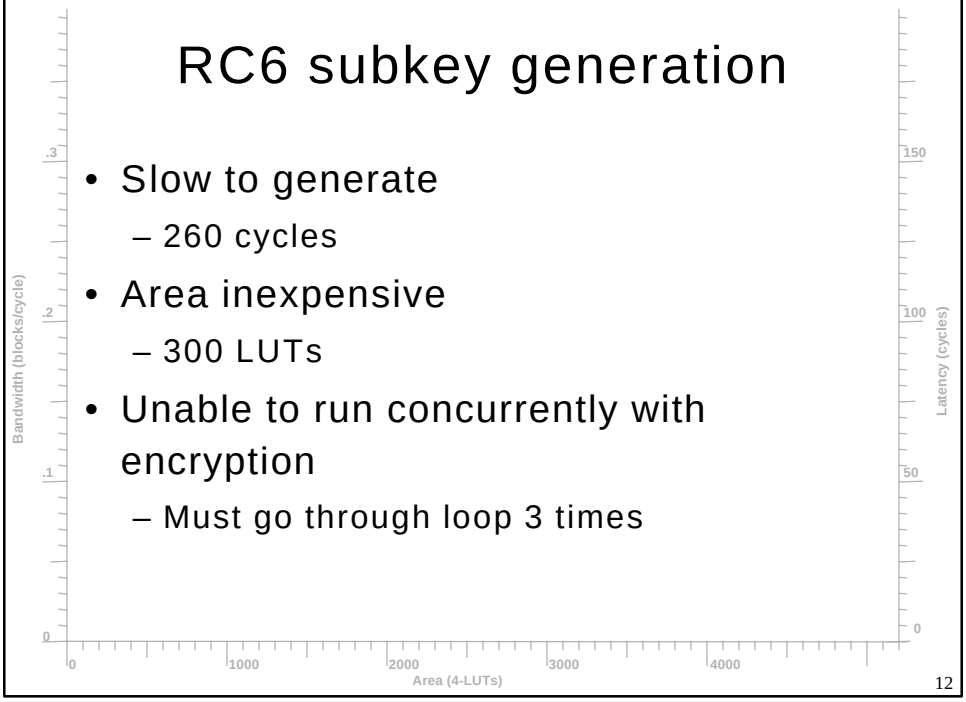
- Poor latency
  - Multiplication is slow, 20 rounds
- Reasonably good area requirements
  - No BlockRAMs required
- Smallest implementation still reasonable bandwidth



11

RC6 subkey generation

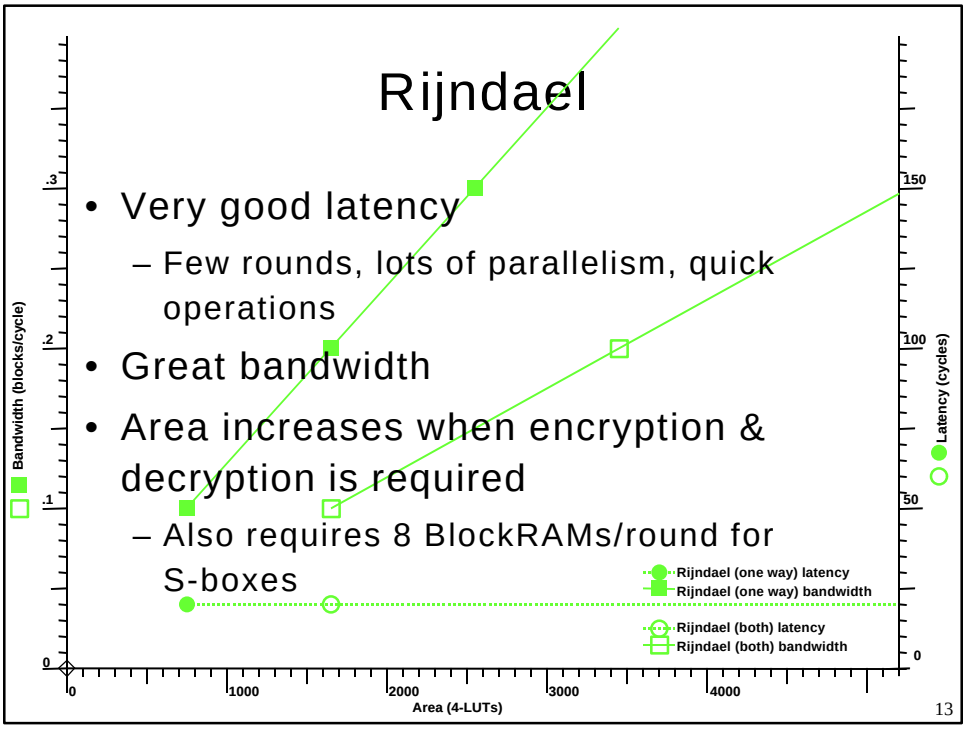
- Slow to generate
  - 260 cycles
- Area inexpensive
  - 300 LUTs
- Unable to run concurrently with encryption
  - Must go through loop 3 times



12

Rijndael

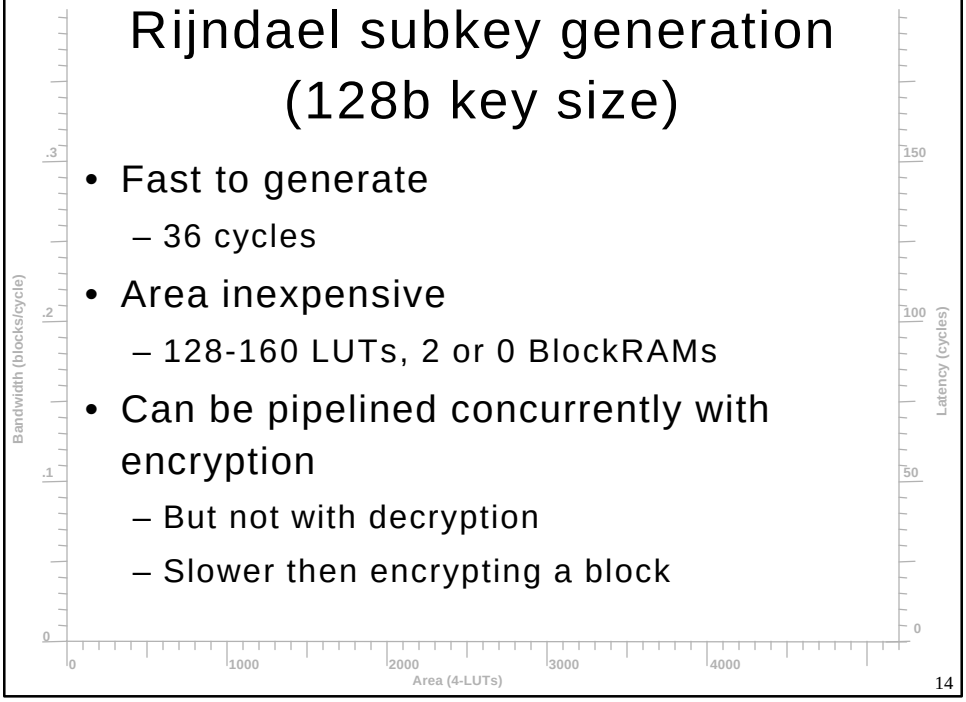
- Very good latency
  - Few rounds, lots of parallelism, quick operations
- Great bandwidth
- Area increases when encryption & decryption is required
  - Also requires 8 BlockRAMs/round for S-boxes



13

Rijndael subkey generation (128b key size)

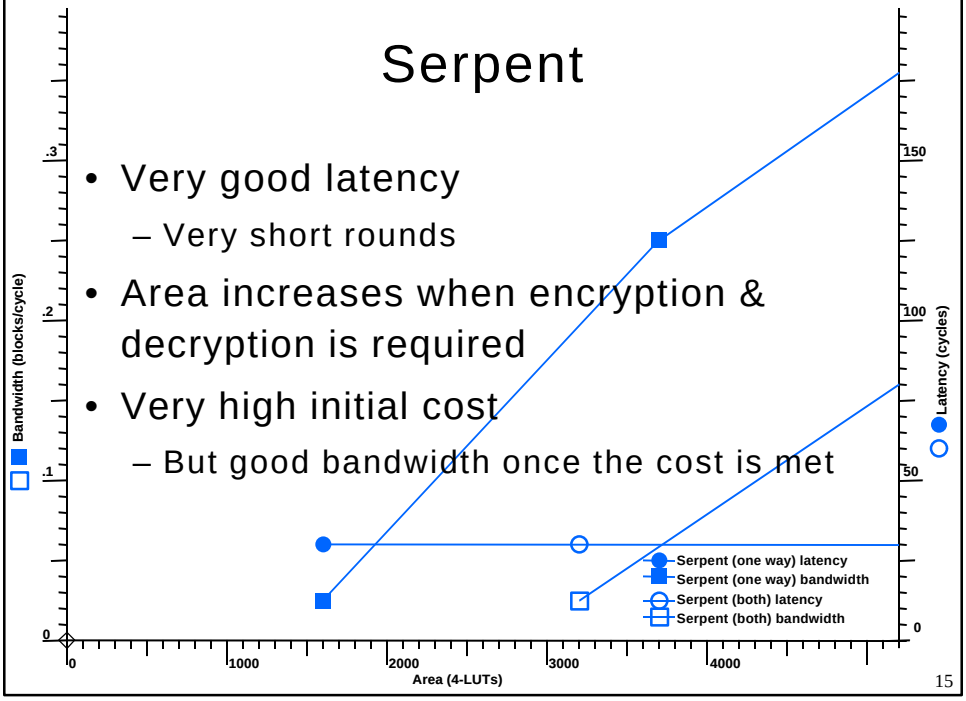
- Fast to generate
  - 36 cycles
- Area inexpensive
  - 128-160 LUTs, 2 or 0 BlockRAMs
- Can be pipelined concurrently with encryption
  - But not with decryption
  - Slower then encrypting a block



14

Serpent

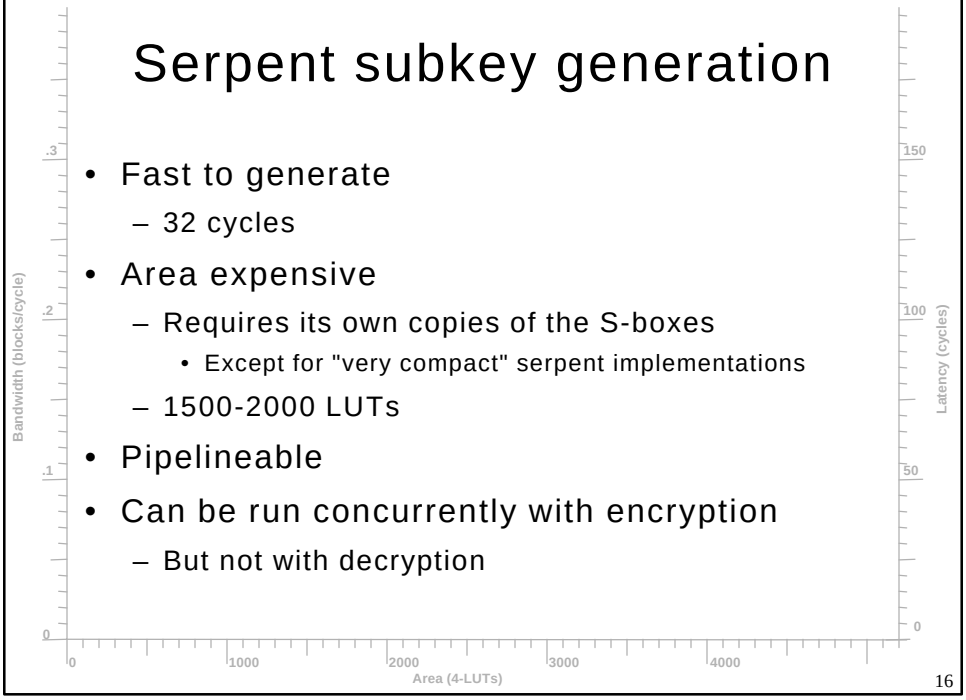
- Very good latency
  - Very short rounds
- Area increases when encryption & decryption is required
- Very high initial cost
  - But good bandwidth once the cost is met



15

Serpent subkey generation

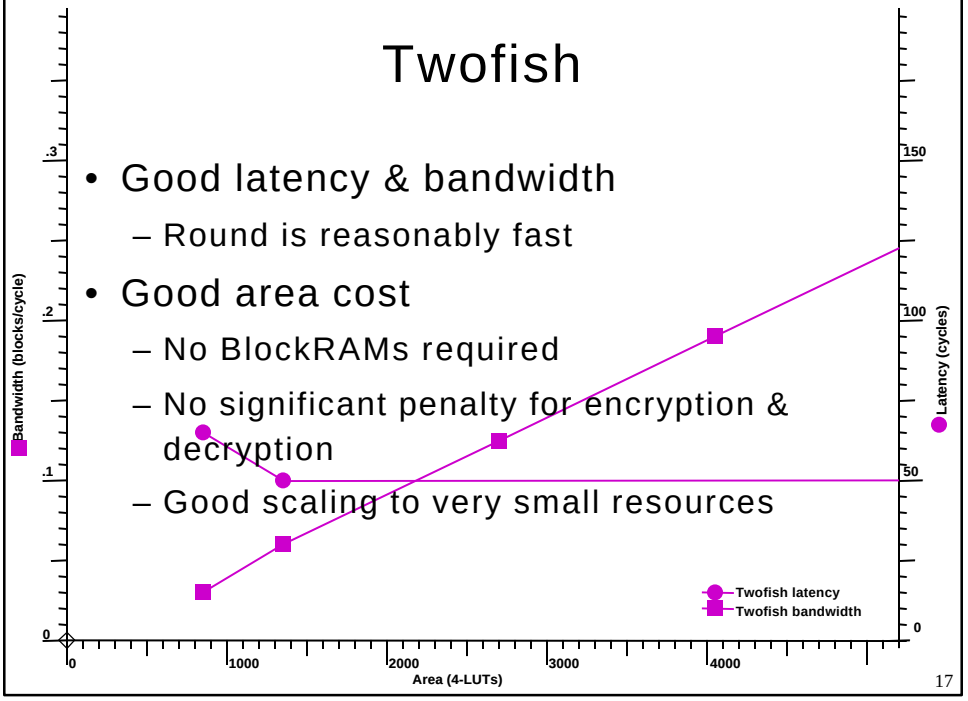
- Fast to generate
  - 32 cycles
- Area expensive
  - Requires its own copies of the S-boxes
    - Except for "very compact" serpent implementations
  - 1500-2000 LUTs
- Pipelineable
- Can be run concurrently with encryption
  - But not with decryption



16

Twofish

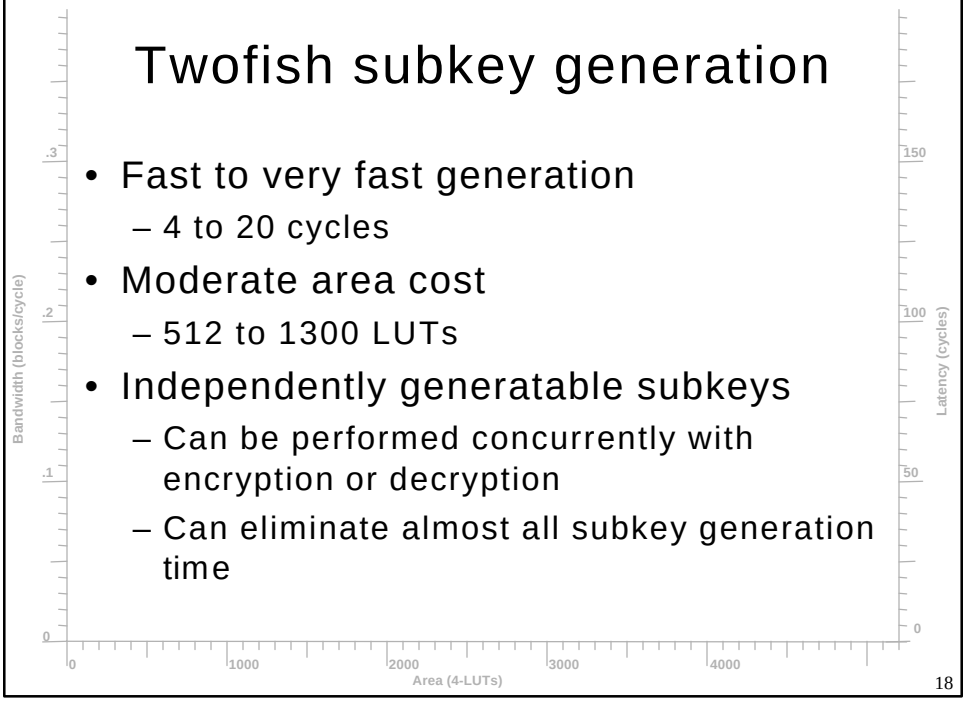
- Good latency & bandwidth
  - Round is reasonably fast
- Good area cost
  - No BlockRAMs required
  - No significant penalty for encryption & decryption
  - Good scaling to very small resources



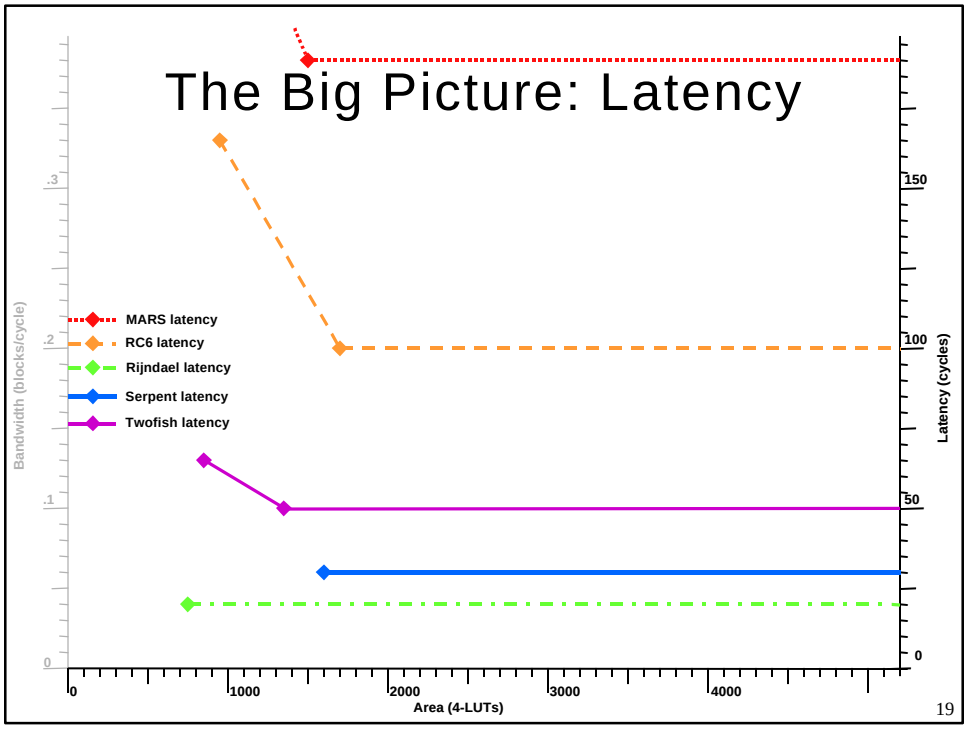
17

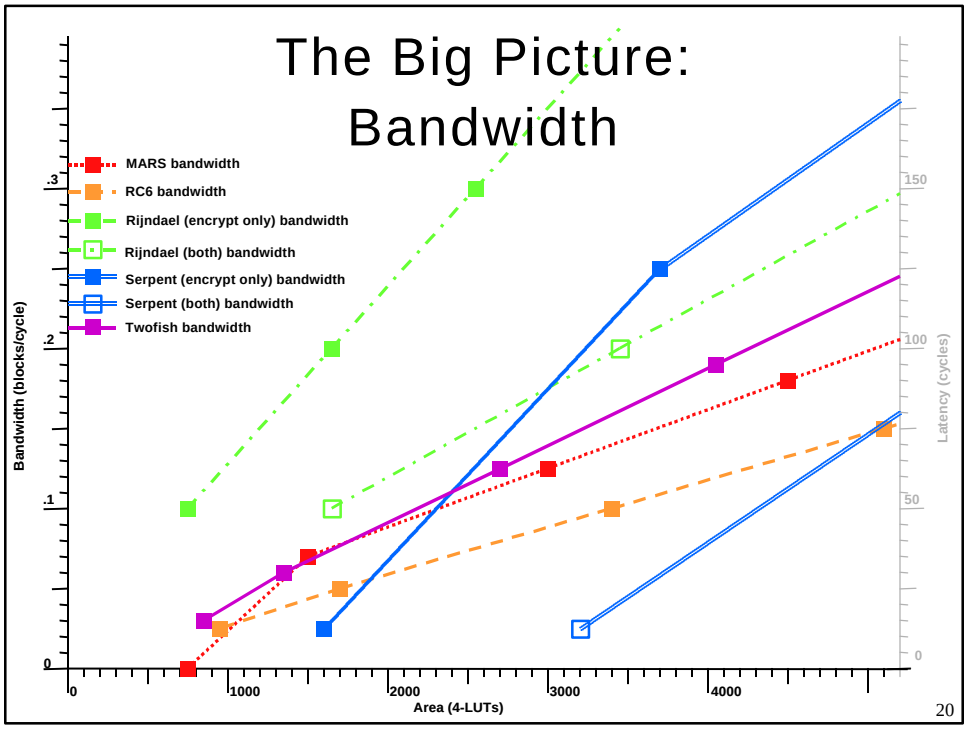
Twofish subkey generation

- Fast to very fast generation
  - 4 to 20 cycles
- Moderate area cost
  - 512 to 1300 LUTs
- Independently generatable subkeys
  - Can be performed concurrently with encryption or decryption
  - Can eliminate almost all subkey generation time



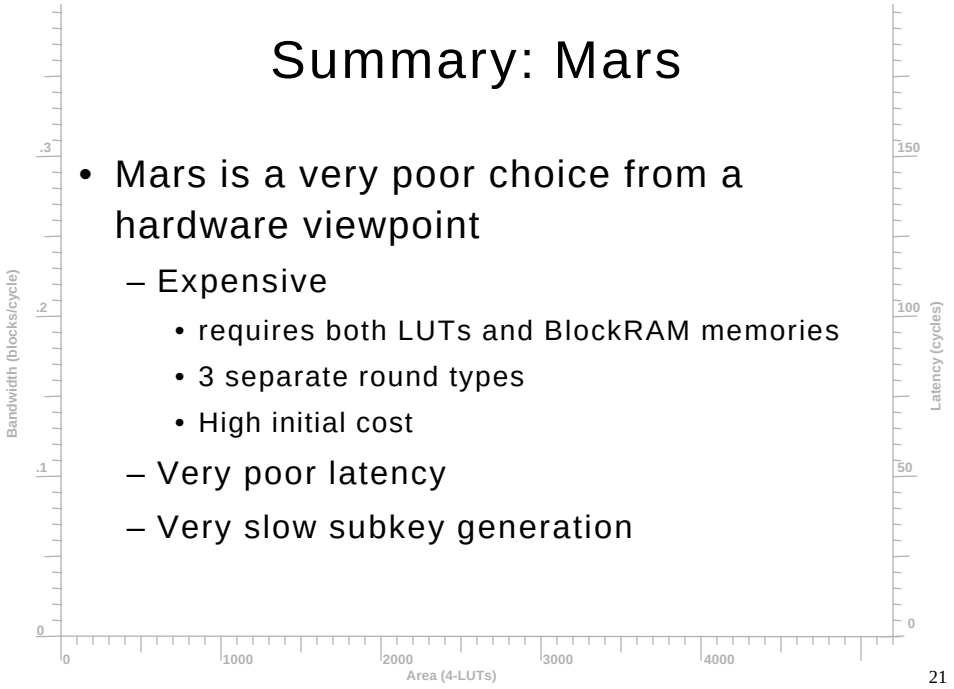
18


The Big Picture: Latency
MARS latency
RC6 latency
Rijndael latency
Serpent latency
Twofish latency
Bandwidth (blocks/cycle)
Latency (cycles)
Area (4-LUTs)
0
1000
2000
3000
4000
.1
.2
.3
50
100
150
19


The Big Picture: Bandwidth
MARS bandwidth
RC6 bandwidth
Rijndael (encrypt only) bandwidth
Rijndael (both) bandwidth
Serpent (encrypt only) bandwidth
Serpent (both) bandwidth
Twofish bandwidth
Bandwidth (blocks/cycle)
Latency (cycles)
Area (4-LUTs)
0
1000
2000
3000
4000
.1
.2
.3
50
100
150
20

Summary: Mars

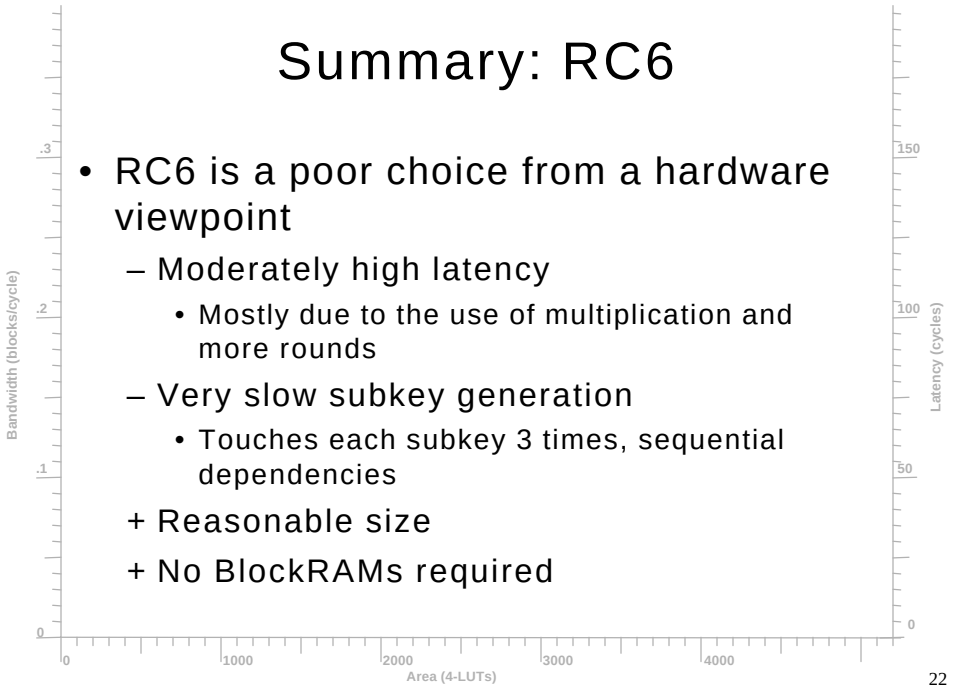
- Mars is a very poor choice from a hardware viewpoint
  - Expensive
    - requires both LUTs and BlockRAM memories
    - 3 separate round types
    - High initial cost
  - Very poor latency
  - Very slow subkey generation



21

Summary: RC6

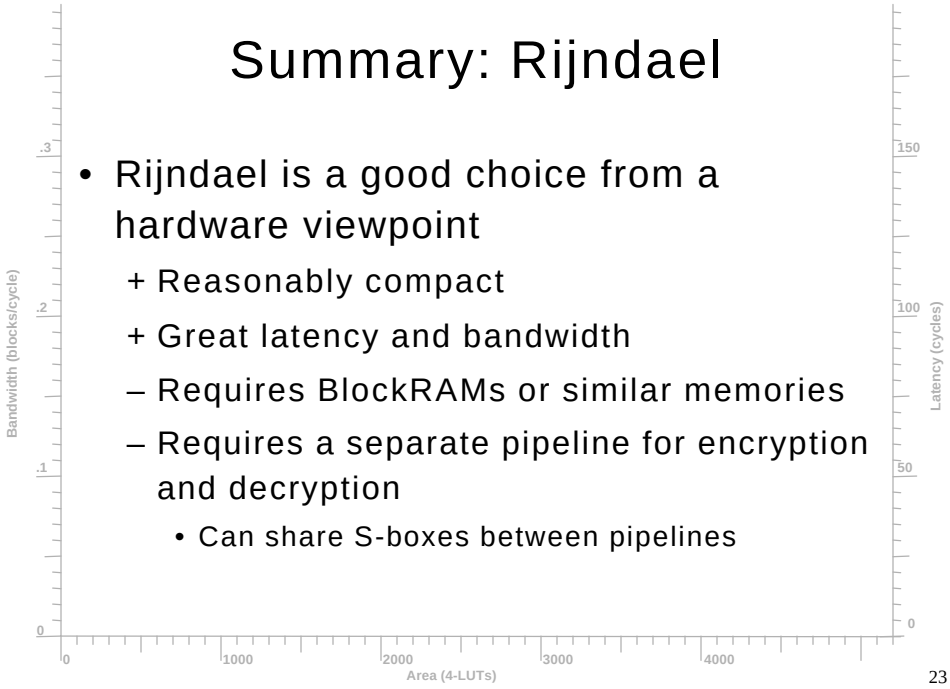
- RC6 is a poor choice from a hardware viewpoint
  - Moderately high latency
    - Mostly due to the use of multiplication and more rounds
  - Very slow subkey generation
    - Touches each subkey 3 times, sequential dependencies
  - \+ Reasonable size
  - \+ No BlockRAMs required



22

Summary: Rijndael

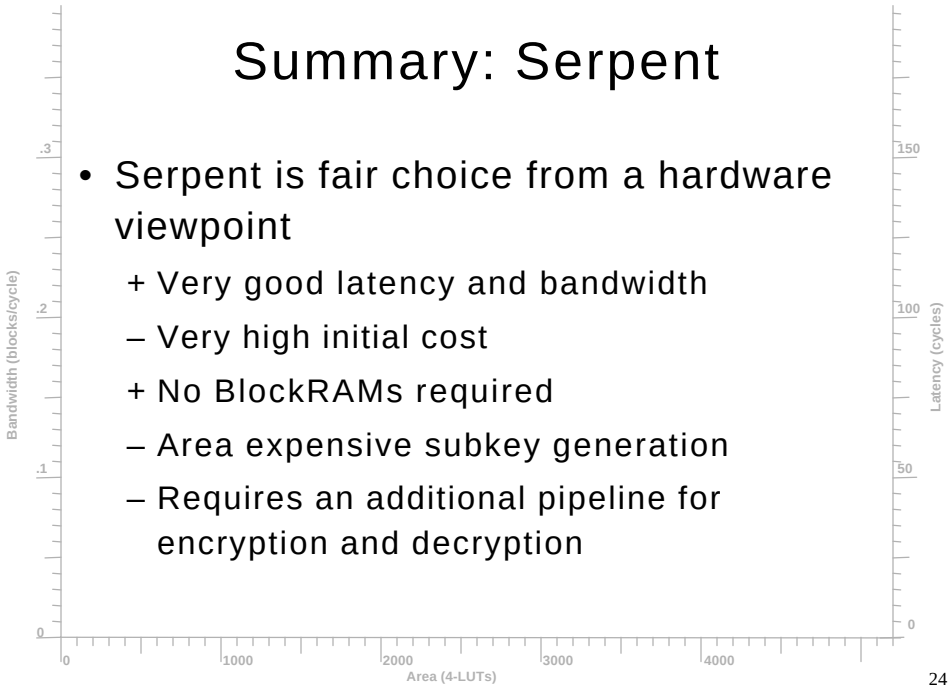
- Rijndael is a good choice from a hardware viewpoint
  - + Reasonably compact
  - + Great latency and bandwidth
  - – Requires BlockRAMs or similar memories
  - – Requires a separate pipeline for encryption and decryption
    - Can share S-boxes between pipelines



23

Summary: Serpent

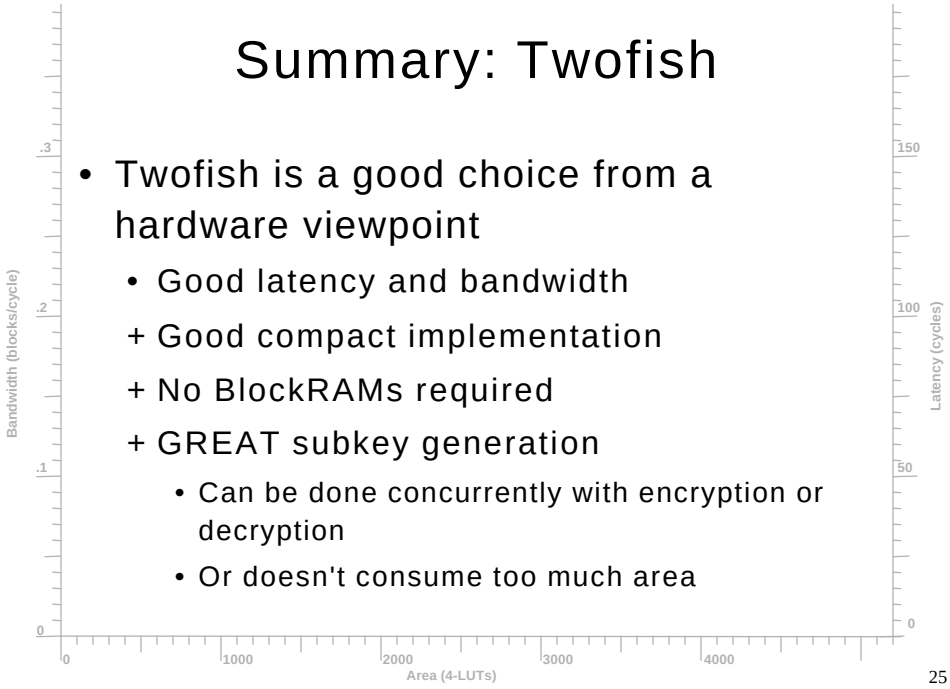
- Serpent is fair choice from a hardware viewpoint
  - + Very good latency and bandwidth
  - – Very high initial cost
  - + No BlockRAMs required
  - – Area expensive subkey generation
  - – Requires an additional pipeline for encryption and decryption



24

Summary: Twofish

- Twofish is a good choice from a hardware viewpoint
  - Good latency and bandwidth
  - \+ Good compact implementation
  - \+ No BlockRAMs required
  - \+ GREAT subkey generation
    - Can be done concurrently with encryption or decryption
    - Or doesn't consume too much area

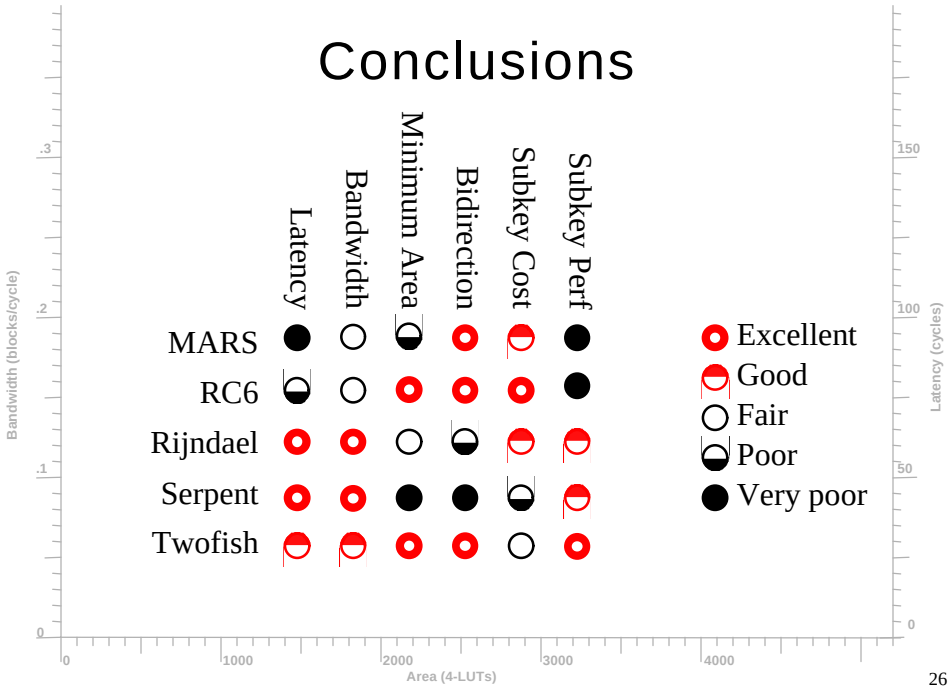


25

Conclusions

	Latency	Bandwidth	Minimum Area	Bidirection	Subkey Cost	Subkey Perf
MARS	●	○	◔	◎	◔	●
RC6	◔	○	◎	◎	◎	●
Rijndael	◎	◎	○	◔	◔	◔
Serpent	◎	◎	●	●	◔	◔
Twofish	◔	◔	◎	◎	○	◎

◎ Excellent
◔ Good
○ Fair
◔ Poor
● Very poor



26