

1-Mar-04 (1)

Tradeoffs in Parallel and Serial Implementations of the International Data Encryption Algorithm IDEA

1-Mar-04 (2)

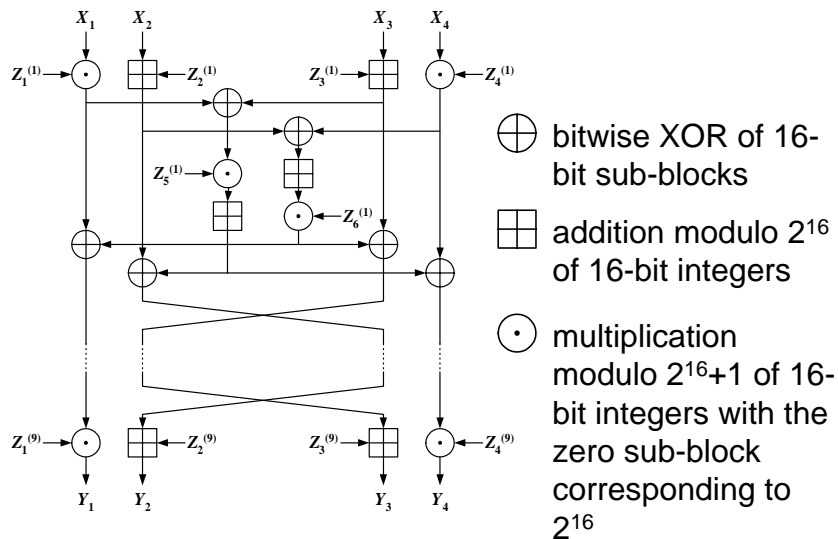
Introduction

- Implementations of the International Data Encryption Algorithm (IDEA)
- A bit-serial approach
 - Deeply pipelined
 - Minimal resource requirements
 - High clock rate
- A bit-parallel approach
 - Deeply pipelined
 - High throughput

FPGA-based Cryptosystems

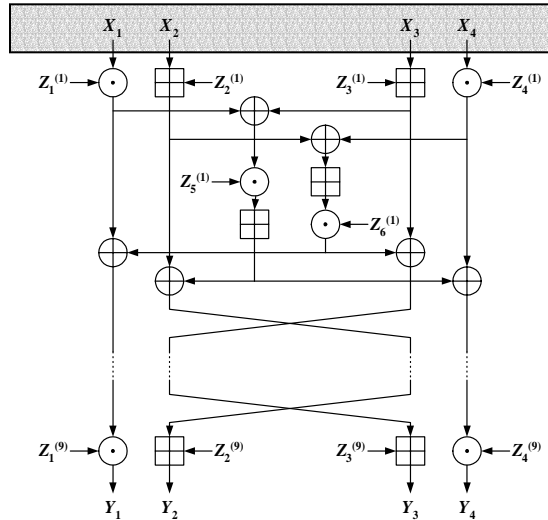
- Cryptography is an ideal application for FPGAs
- The same FPGA hardware can be used for many different cryptographic protocols
- Moore's law guarantees the availability of faster and larger FPGA devices
- Possibility of specializing the hardware
- Reconfigurable nature makes it feasible to attempt designs employing more sophisticated algorithms which leads to improved performance

International Data Encryption Algorithm



1-Mar-04 (5)

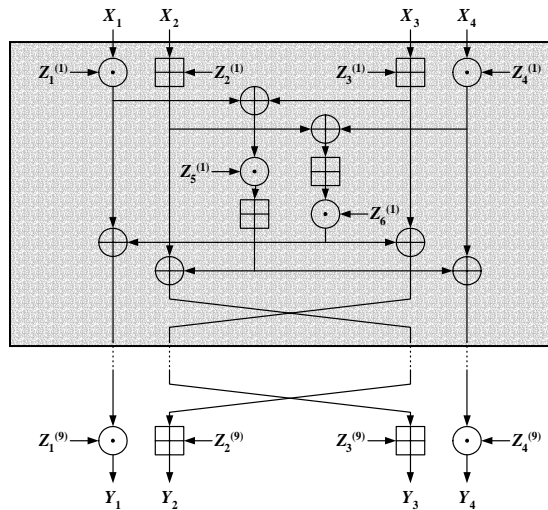
International Data Encryption Algorithm



A 64-bit plaintext is divided into 4 16-bit sub-blocks, X_1 to X_4

1-Mar-04 (6)

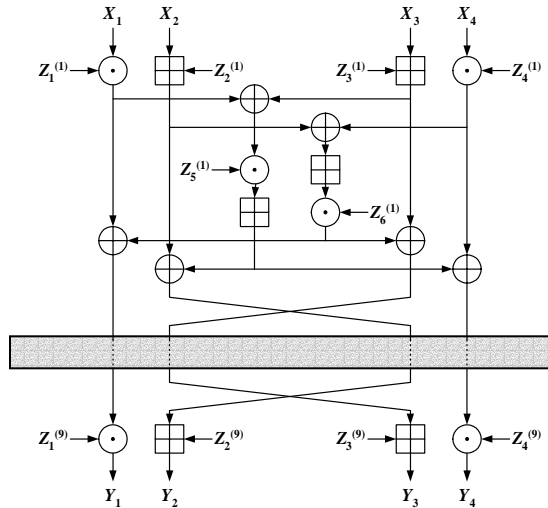
International Data Encryption Algorithm



The algorithm consists of iterative blocks known as rounds, each round takes 6 16-bit subkeys, $Z_1^{(r)}$

1-Mar-04 (7)

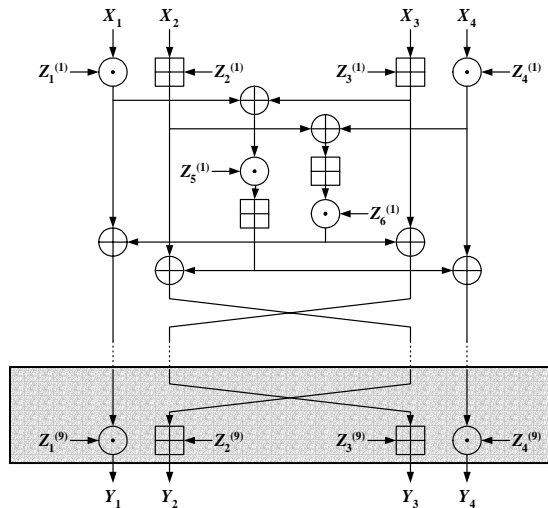
International Data Encryption Algorithm



There are 8 cascaded identical (full) rounds in the algorithm

1-Mar-04 (8)

International Data Encryption Algorithm

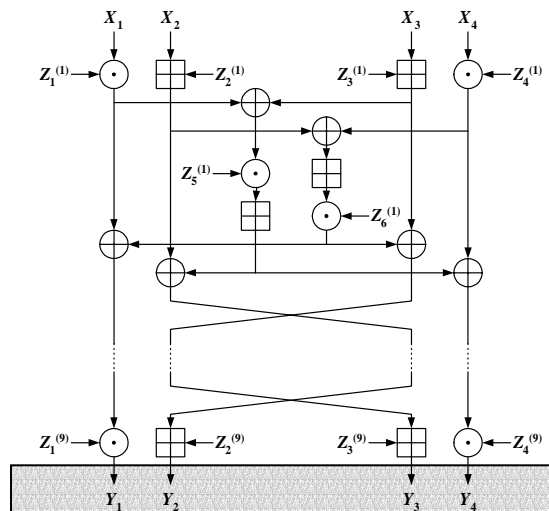


After the 8 full rounds there is 1 half round (or output transformation) which takes 4 16-bit subkeys

All the $8 \times 6 + 4 = 52$ subkeys are derived from a 128-bit key

1-Mar-04 (9)

International Data Encryption Algorithm



The 4 16-bit output sub-blocks, Y_1 to Y_4 , form the 64-bit ciphertext

Decryption process is the same as the encryption process but with a different key schedule (also derived from the 128-bit key)

1-Mar-04 (10)

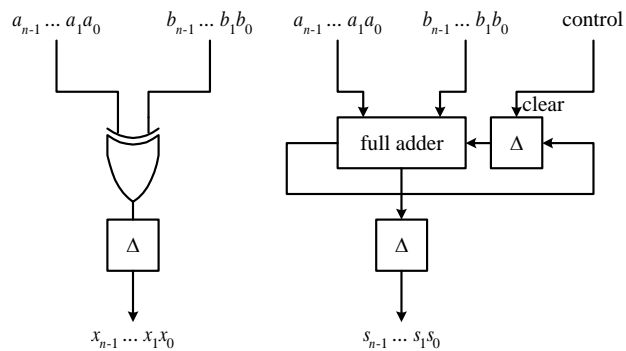
Bit-Serial Architectures

- Characterized by the property that
 - operators perform their computations in a bitwise fashion
 - communications between operators are multiplexed in time over a single wire
- Minimal hardware requirements
- High clock rate at the expense of increased latency

1-Mar-04 (11)

Bit-Serial Architectures

- Two of the primitive operators, XOR and addition modulo 2^{16} , can be implemented trivially



1-Mar-04 (12)

Multiplication Modulo $2^{16}+1$

```
uint16 mulmod(uint16 x, uint16 y)
{
    uint32 t;
    x = (x - 1) & 0xFFFF;
    y = (y - 1) & 0xFFFF;
    t = (uint32) x * y + x + y + 1;
    x = t & 0xFFFF;
    y = t >> 16;
    x = (x - y) + (x <= y);
    return x;
}
```

32-bit intermediate result
may bottleneck the
throughput of the operator

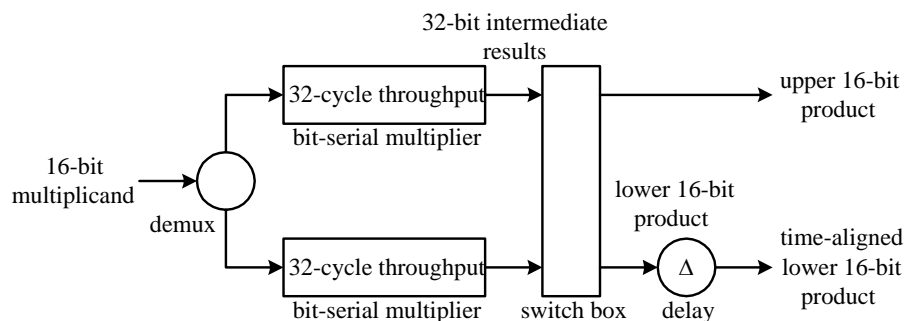
1-Mar-04 (13)

Multiplication Modulo $2^{16}+1$

- Solution – use multiple multiplication pipelines
- 2 inputs a and b – multiplicand and multiplier, each 16-bit
- 2 outputs p and q – alternately output 32-bit products
- Takes advantage that for a given key, one of the operands is always a constant (a subkey) in IDEA
- Fully pipelined – a new bit can enter every cycle

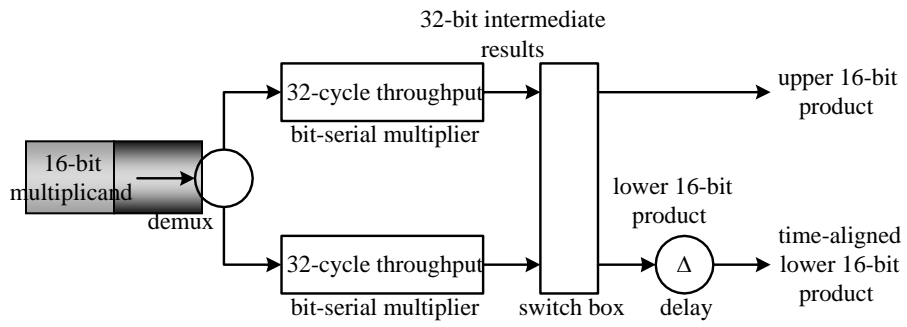
1-Mar-04 (14)

Multiplication Modulo $2^{16}+1$



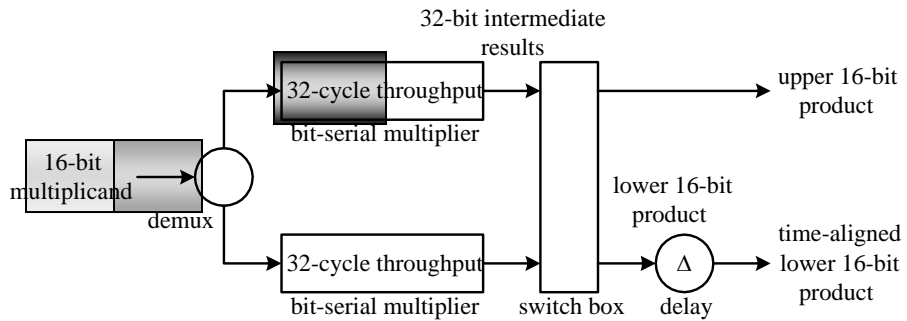
1-Mar-04 (15)

Multiplication Modulo $2^{16}+1$



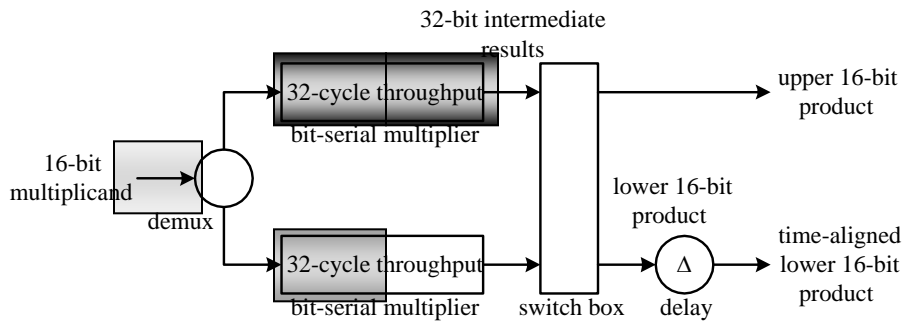
1-Mar-04 (16)

Multiplication Modulo $2^{16}+1$



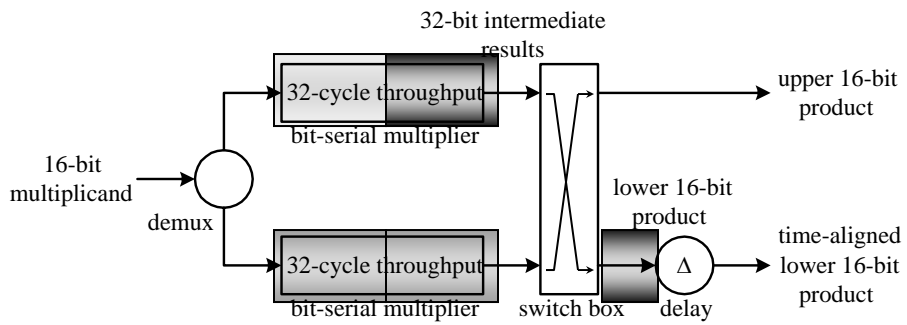
1-Mar-04 (17)

Multiplication Modulo $2^{16}+1$



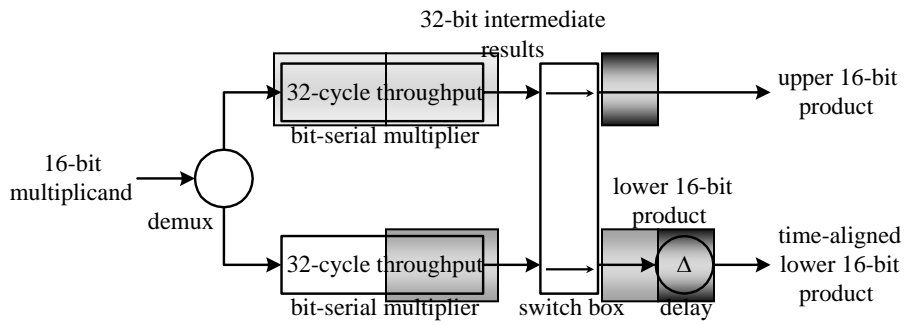
1-Mar-04 (18)

Multiplication Modulo $2^{16}+1$



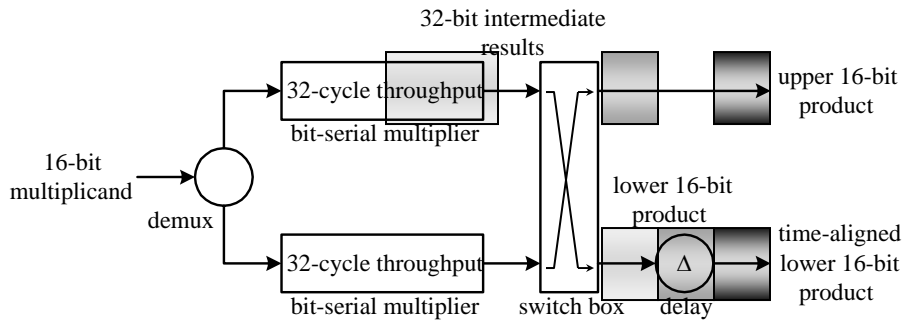
1-Mar-04 (19)

Multiplication Modulo $2^{16}+1$



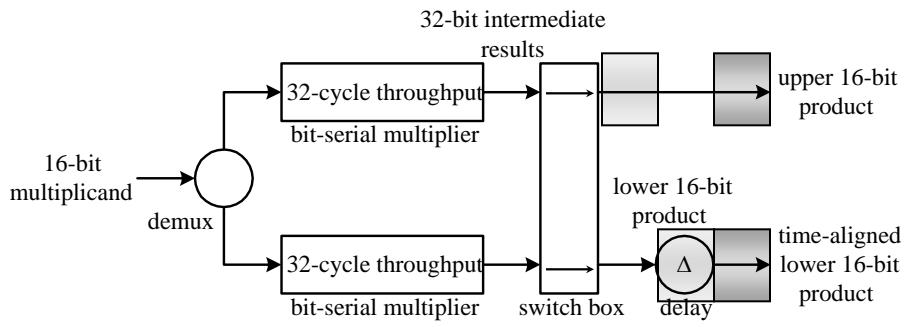
1-Mar-04 (20)

Multiplication Modulo $2^{16}+1$



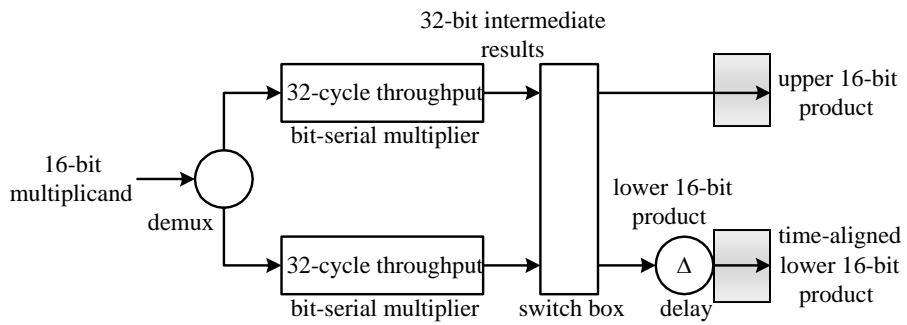
1-Mar-04 (21)

Multiplication Modulo $2^{16}+1$



1-Mar-04 (22)

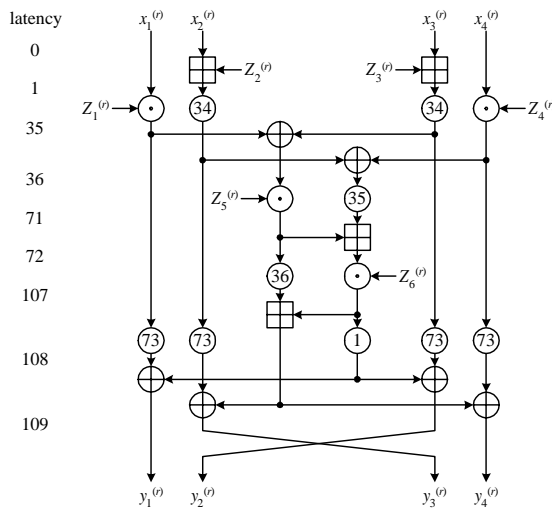
Multiplication Modulo $2^{16}+1$



Multiplication Modulo $2^{16}+1$

- Extract the lower and upper 16 bits from the 32-bit intermediate value using a switch box
- Flips the two inputs every 16 cycles
- Overall latency is 35 cycles
- Overall throughput is 16 cycles – the same as the other 2 operators therefore not a bottleneck

IDEA Core – One Round



Stage latches (SRL16E) are inserted for time-alignment

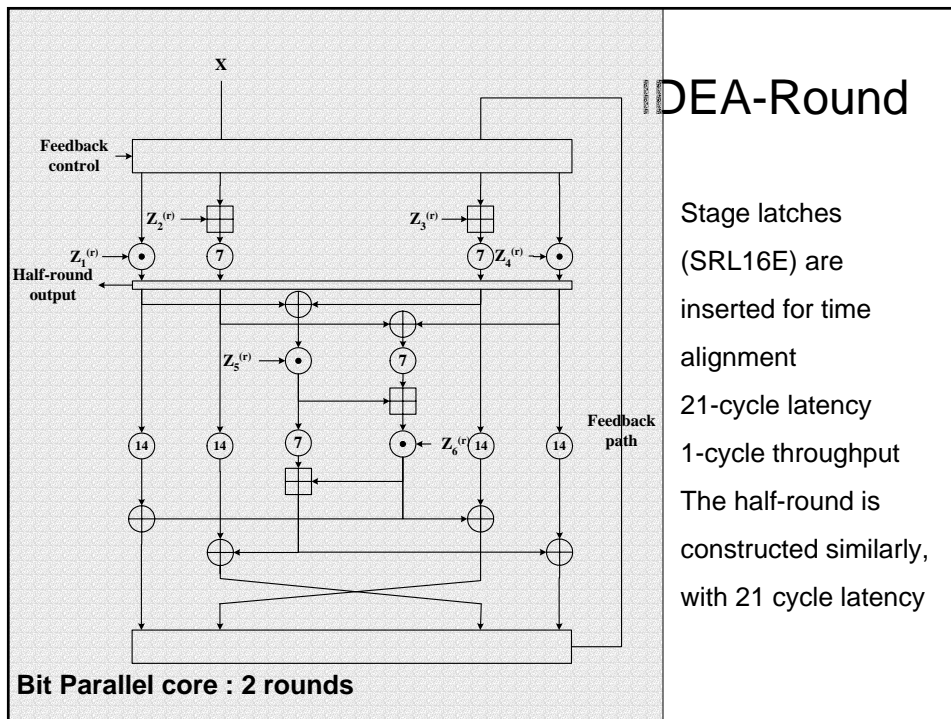
109-cycle latency

16-cycle throughput

The half-round is constructed similarly, with 35-cycle latency

Bit-Parallel Architectures

- 16 bit Multiplier
 - 16 bit multiplier has 4 stage pipeline
- Compared with bit-serial approach
 - high throughput at the expense of hardware requirement
 - In an XCV300-6, only 2 IDEA rounds can be contained
- Modular multiplier
 - Built from 16 bit multiplier
 - Latency is 7 cycles
 - Throughput is 1 cycle



IDEA-Round

Stage latches (SRL16E) are inserted for time alignment
 21-cycle latency
 1-cycle throughput
 The half-round is constructed similarly, with 21 cycle latency

1-Mar-04 (27)

Runtime Reconfigurability

- Key-schedule stored inside ROM implemented as a lookup table
- FPGA design stored as a bitstream
- This design changes bitstream directly to modify the key
 - Saves interface logic normally required to change the key

1-Mar-04 (28)

Runtime Reconfigurability

- First step (only performed once for a given design)
 - Information about physical location of individual LUTs for key-schedule are extracted from bitstream
 - Those information written to a location file (locfile)

1-Mar-04 (29)

Runtime Reconfigurability

- Second step (performed every time key changed)

```
Changekeys(locfile,bitsteam)
```

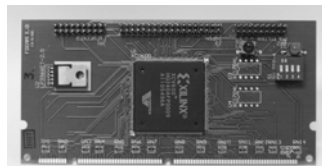
```
{
  locdb = read(locfile);
  for each bit in the key
  {
    Find location of the LUT using locdb;
    Modify the value of the bit in the LUT;
  }
  Recompute CRC for the bitstream;
  Write bitstream;
}
```

- Time to change keys : 0.12sec

1-Mar-04 (30)

Hardware Platform

- CUHK Pilchard board
 - Memory slot interface (100 Mhz 64bit)
- Annapolis Micro Systems Wildcard Reconfigurable Computing Engine
 - CardBus interface (33MHz 32bit)



Both use Xilinx Virtex XCV300 FPGA

1-Mar-04 (31)

Results – IDEA Core

	Bit-Parallel			Bit-Serial		
Device (speed grade –6)	XCV300	XCV600	XCV1000	XCV300	XCV600	XCV1000
Scaling	2x	5x	9x	1x	2x	4x
Number of slices	2444	6368	11602	2878	5756	11512
Device Utilization	79.56%	92.13%	94.42%	93.68%	83.28%	93.68%
Encryptions per second ($\times 10^6$)	18.220	45.551	81.991	9.375	18.750	37.500
Encryptions rate (Mb/sec)	1166.1	2915.3	5247.4	600.0	1200.0	2400.0

1-Mar-04 (32)

Results – Comparison of Performance (ECB Mode)

Implementation	Performance
Ascom's implementation on Intel Pentium II 450MHz	23Mb/sec
Optimized implementation on Sun Enterprise E4500 (12 Ultra-IIi 400MHz processors)	147Mb/sec
VLSI implementation by Bonnenberg et. al. (1.5 μ m CMOS technology, 1992)	44Mb/sec
VINCI (1.2 μ m CMOS technology, 1994)	177Mb/sec
VLSI implementation by Wolter et. al. (0.8 μ m CMOS technology, 1995)	355Mb/sec
VLSI implementation by Salomao et. al. (0.7 μ m CMOS technology, 1998)	424Mb/sec
Ascom IDEACrypt Coprocessor	300Mb/sec
Proposed bit-parallel IDEA core (0.22 μ m FPGA-based implementation, 2001)	1166Mb/sec (XCV300) 5247Mb/sec (XCV1000)

1-Mar-04 (33)

Results – Implementations

- Wildcard
 - Measured encryption rate 39Mb/sec
- Pilchard
 - Measured encryption rate 146Mb/sec
- Bottleneck of the implementation was in the CPU to FPGA transfers
- Pilchard much faster since it has much lower latency and higher bandwidth transfers

1-Mar-04 (34)

Conclusion

- Two different designs of the IDEA cipher were presented
- 600Mbit/sec encryption rate at 150MHz on an XCV300-6 FPGA for bit-serial approach
- 1166Mbit/sec encryption rate at 82MHz on an XCV300-6 FPGA for bit-parallel approach
- Verified on Wildcard (39Mb/sec) and Pilchard (146Mb/sec)
- For IDEA, the bit-parallel implementation has higher throughput and lower latency for the same area
- Bitstream reconfiguration used to save key-schedule logic

1-Mar-04 (35)

Tradeoffs in Parallel and Serial Implementations of the International Data Encryption Algorithm IDEA

1-Mar-04 (36)

Runtime Reconfigurability

- 6 X 16 SRL16E to implement our key schedule
 - = 96 slices (3.125% of XCV300)
- With bitstream reconfiguration we use 1/2 the number of slices

Encryption modes

- Our implementations only works for ECB mode
- Previous highly pipelined designs have same problem
- One solution: Feedback delayed output