FPGA security

Nele Mentens
nele.mentens@kuleuven.be

Summer school on real-world crypto and privacy

May 31 – June 5, 2015, Šibenik, Croatia

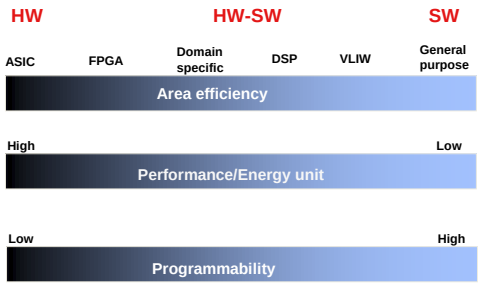
Outline

- Introduction
- FPGA technology
- FPGA security

Outline

- Introduction
  - FPGA vs. other platforms
  - FPGA vs. ASIC
  - FPGA vs. general purpose microprocessor
  - FPGA application
- FPGA technology
- FPGA security

Introduction
FPGA vs. other platforms



Introduction
FPGA vs. ASIC

- FPGA = Field-Programmable Gate Array
  - Programmable hardware
- ASIC = Application-Specific Integrated Circuit
  - Fixed hardware
- FPGA advantages over ASIC
  - faster time-to-market
  - smaller Non-Recurring Engineering (NRE) cost
  - programmable in the field
- ASIC advantages over FPGA
  - lower cost for high volumes
  - better performance

Introduction
FPGA vs. general purpose microprocessor

- FPGA
  - Programmable hardware
  - Code (= hardware description language, e.g. VHDL) describes the hardware that we need
- General purpose microprocessor
  - Fixed hardware
  - Code (= programming language, e.g. C) describes what should be executed on the fixed hardware

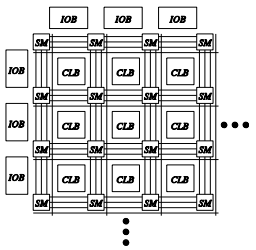
Introduction
FPGA application

- Prototype for ASIC design
- End product
  - Recently developed FPGAs are heterogeneous systems with dedicated building blocks.
  - FPGAs closely follow technology scaling because they are manufactured in high volumes.
- Application domains:
  - space
  - telecommunication
  - signal processing
  - …
- Many applications require data security on FPGA.

Outline

- Introduction
- FPGA technology
  - Architecture
  - Configuration
  - Design flow
  - Example code
  - Vendors
  - Performance comparison
- FPGA security

FPGA technology
Architecture – regular cells

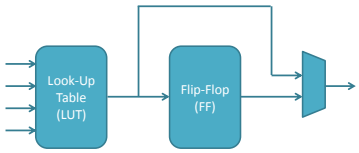


Basic FPGA architecture:

- CLB = Configurable Logic Block
  - CLBs consist of slices.
  - Slices consist of
    - Look-Up Tables (LUTs),
    - Multiplexers,
    - Flip-Flops (FFs),
    - Carry logic.
- SM = Switch Matrix
- IOB = Input/Output Block

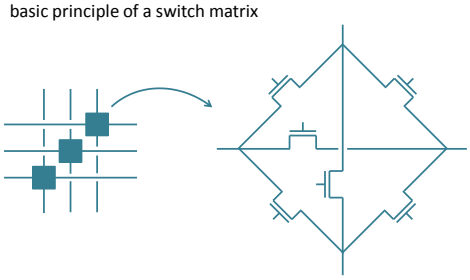
FPGA technology
Architecture – regular cells

basic content of a slice (excluding carry-logic)



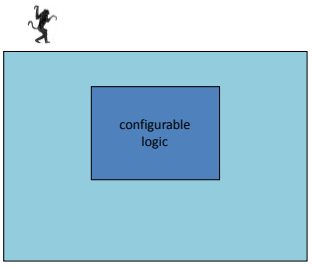
FPGA technology
Architecture – regular cells

basic principle of a switch matrix

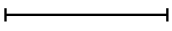


FPGA technology
Architecture – evolution of Xilinx FPGAs

1991: XC4000



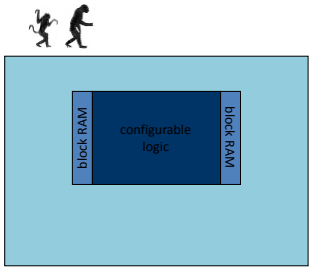
technology node:



0.25 µm

FPGA technology
Architecture – evolution of Xilinx FPGAs

1991: XC4000
1998: Virtex

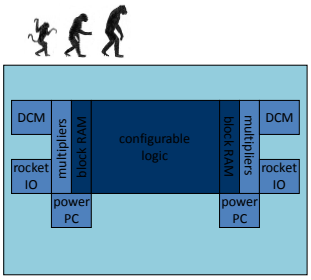


technology node:

0.22 µm

FPGA technology
Architecture – evolution of Xilinx FPGAs

1991: XC4000
1998: Virtex
2002: Virtex-II Pro

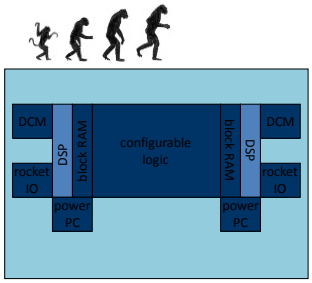


technology node:

0.13 µm

FPGA technology
Architecture – evolution of Xilinx FPGAs

1991: XC4000
1998: Virtex
2002: Virtex-II Pro
2004: Virtex-4

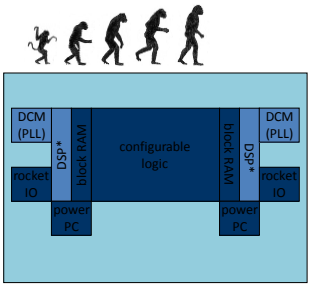


technology node:

90 nm

FPGA technology
Architecture – evolution of Xilinx FPGAs

1991: XC4000
1998: Virtex
2002: Virtex-II Pro
2004: Virtex-4
2006: Virtex-5

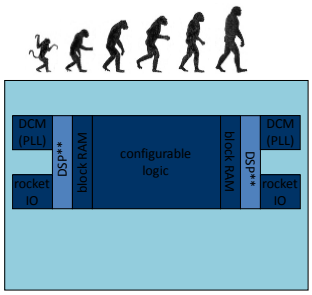


technology node:

65 nm

FPGA technology
Architecture – evolution of Xilinx FPGAs

1991: XC4000
1998: Virtex
2002: Virtex-II Pro
2004: Virtex-4
2006: Virtex-5
2009: Virtex-6

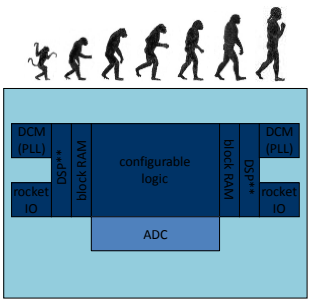


technology node:

45 nm

FPGA technology
Architecture – evolution of Xilinx FPGAs

1991: XC4000
1998: Virtex
2002: Virtex-II Pro
2004: Virtex-4
2006: Virtex-5
2009: Virtex-6
2010: Virtex-7



technology node:

28 nm

FPGA technology
Architecture – evolution of Xilinx FPGAs

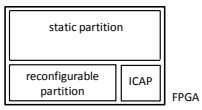
- Recent development of Xilinx FPGAs:
  - Zynq-7000 series
  - ARM + FPGA
  - Processor-centered architecture

FPGA technology
Configuration

- Configuration data: bitstream
- Configuration technology:
  - (anti-)fuse: one-time programmable
  - flash: non-volatile configuration memory
  - SRAM: volatile configuration memory
- SRAM (vs. flash) configuration memory
  - Higher density
  - Higher power consumption
  - On-board or on-chip non-volatile memory needed to store the bitstream during power-off
  - Higher configuration speed

FPGA technology
Configuration

- On recent FPGAs, configuration can be partial/dynamic:
  - Partial: only a part of the FPGA is reconfigured
  - Dynamic: the FPGA is reconfigured at run-time
- For dynamic reconfiguration, the configuration memory is accessible from inside the FPGA through an ICAP (Internal Configuration Access Port)*



- On processor-centered architectures, the (partial or full) configuration is handled by the processor through a PCAP (Processor Configuration Access Port)*

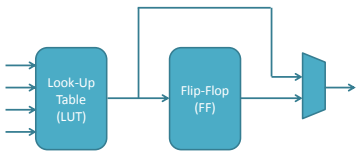
*terminology used for Xilinx FPGAs

FPGA technology
Configuration

- Design parameters for partial reconfiguration:
  - bitstream size
    - proportional to the size of the reconfigurable partition
  - reconfiguration time
    - proportional to the size of the reconfigurable partition
  - availability of on-chip decryption
    - not available for partial bitstreams in older FPGA families

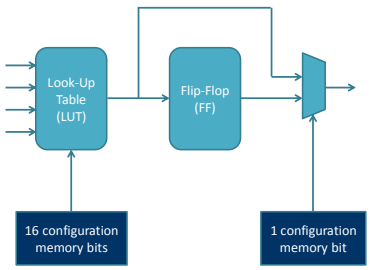
FPGA technology
Configuration

basic content of a slice (excluding carry logic)



FPGA technology
Configuration

basic content of a slice (excluding carry logic) + configuration



FPGA technology
Configuration

A	B	C	D	Z_0	Z_1	Z_2	Z_3	...	Z_{65280}	...	Z_{65535}
0	0	0	0	0	1	0	1		0		1
0	0	0	1	0	0	1	1		0		1
0	0	1	0	0	0	0	0		0		1
0	0	1	1	0	0	0	0		0		1
0	1	0	0	0	0	0	0		0		1
0	1	0	1	0	0	0	0		0		1
0	1	1	0	0	0	0	0		0		1
0	1	1	1	0	0	0	0		0		1
1	0	0	0	0	0	0	0		1		1
1	0	0	1	0	0	0	0		1		1
1	0	1	0	0	0	0	0		1		1
1	0	1	1	0	0	0	0		1		1
1	1	0	0	0	0	0	0		1		1
1	1	0	1	0	0	0	0		1		1
1	1	1	0	0	0	0	0		1		1
1	1	1	1	0	0	0	0		1		1

Why 16 configuration bits for a 4-to-1 LUT?

2^{16} possible output functions:

$Z_0 = 0$

$Z_1 = A'.B'.C'.D'$

$Z_2 = A'.B'.C'.D$

$Z_3 = A'.B'.C'$

...

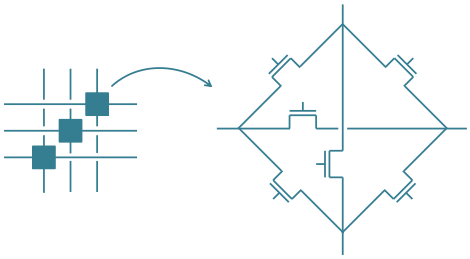
$Z_{65280} = A$

...

$Z_{65535} = 1$

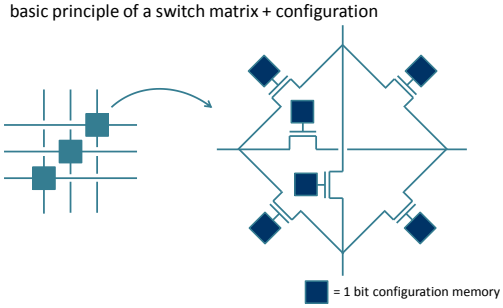
FPGA technology
Configuration

basic principle of a switch matrix

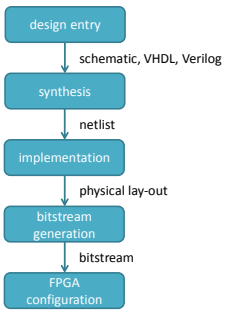


FPGA technology
Configuration

basic principle of a switch matrix + configuration



FPGA technology
Design flow



FPGA technology
Example code

- Two code snippets that describe the same function:
  - One in VHDL (to be processed by a synthesis tool)
  - One in C (to be processed by a compiler)

VHDL

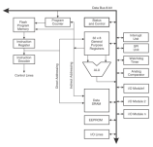
```
if sel = '1' then
        z <= a sll 1;
else
        z <= a srl 1;
end if;
```

C

```
if((sel & 0x01) == 1)    {
        z = a << 1;
} else {
        z = a >> 1;
}
```

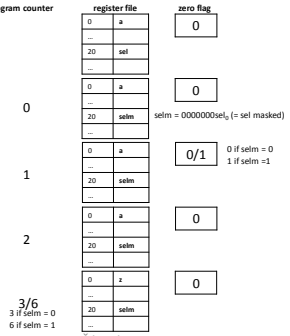

FPGA technology
Example code

C → microprocessor

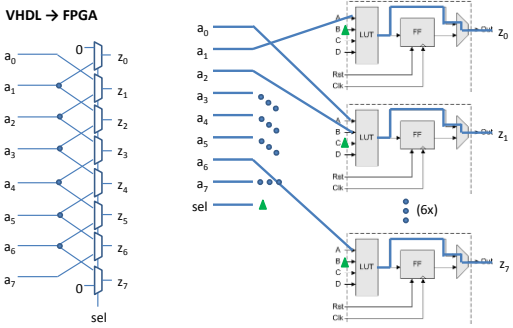


program memory

0	0111000001000001	ANDI r20, 1
1	0011000001000001	CPI r20, 1
2	1111000000011001	BREQ 3
3	1001010000000110	LSR r0
...		
6	0000110000000000	LSL r0



FPGA technology
Example code



FPGA technology
Example code

upper LUT = upper multiplexer

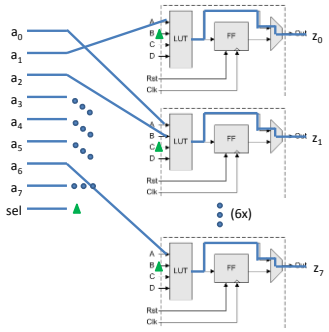
A	B	C	D	LUT-out
0	0	x	x	0
0	1	x	x	0
1	0	x	x	1
1	1	x	x	0

middle LUT (6x) = middle multiplexer (6x)

A	B	C	D	LUT-out
0	0	0	x	0
0	0	1	x	0
0	1	0	x	1
0	1	1	x	0
1	0	0	x	0
1	0	1	x	1
1	1	0	x	1
1	1	1	x	1

lower LUT = lower multiplexer

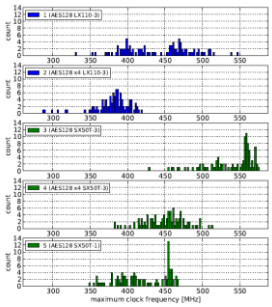
A	B	C	D	LUT-out
0	0	x	x	0
0	1	x	x	0
1	0	x	x	0
1	1	x	x	1



FPGA technology
Vendors

- Xilinx and Altera:
  - SRAM-based FPGAs
  - Over 2/3 of the FPGA market
- MicroSemi:
  - Flash-based FPGAs
  - Specific market sector for applications that require high reliability, high security, low power
- Lattice and Quicklogic:
  - Low power

FPGA technology
Performance comparison



- Be careful not to compare apples to oranges.
- Performance depends on:
  - the place & route seed,
  - the degree of occupation,
  - the speed grade of the device.
- Results and picture from Saar Drimer's Ph.D. dissertation
- Use the ATHENa tool of George Mason University for a fair comparison (https://cryptography.gmu.edu/athena)

Outline

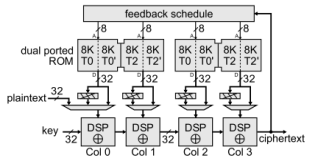
- Introduction
- FPGA technology
- FPGA security
  - Crypto on FPGA
  - Secure remote configuration
  - Secure IP core licensing

FPGA security
Crypto on FPGA

- Maximize the use of dedicated building blocks
  - Multipliers (in older FPGAs)
    - A*B
    - with or without registers
  - DSP slices (in more recently developed FPGAs)
    - version 1: A * B + C
    - version 2: (A + B) * C + D
    - many options for including or excluding pipeline registers
  - Block RAM
    - single-port or dual-port
  - Shift registers
    - a LUT can also be used as an addressable shift register

FPGA security
Crypto on FPGA

Example: AES T-box implementation on FPGA [1]
- Block RAM for SubBytes and MixColumns
- DSP slices for AddRoundKey



$$T_0[x] = \begin{bmatrix} S[x] \times 02 \\ S[x] \\ S[x] \\ S[x] \times 03 \end{bmatrix} \quad T_1[x] = \begin{bmatrix} S[x] \times 03 \\ S[x] \times 02 \\ S[x] \\ S[x] \end{bmatrix}$$

$$T_2[x] = \begin{bmatrix} S[x] \\ S[x] \times 03 \\ S[x] \times 02 \\ S[x] \end{bmatrix} \quad T_3[x] = \begin{bmatrix} S[x] \\ S[x] \\ S[x] \times 03 \\ S[x] \times 02 \end{bmatrix}$$

[1] S. Drimer, T. Güneysu, and C. Paar, "DSPs, BRAMs and a pinch of logic: extended recipes for AES on FPGAs", ACM Transactions on Reconfigurable Technology and Systems (TRETS), 3(1), 2010.

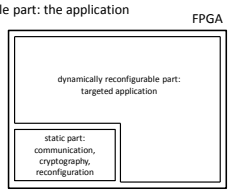
FPGA security
Secure remote configuration

- Technology support:
  - Confidentiality:
    - Built-in decryption core (AES), encryption/decryption key stored in internal ROM
  - Authentication:
    - Built-in HMAC core, authentication key included in the encryption bitstream
- Issues:
  - Vulnerability to replay attacks:
    - Changing the encryption/decryption key can't be done remotely without erasing the whole configuration.
  - Vulnerability to side-channel attacks:
    - The built-in decryption core was proven to be susceptible to side-channel attacks, allowing the attacker to recover the bitstream [2]

[2] A. Moradi, M. Kasper, and C. Paar. Black-Box Side-Channel Attacks Highlight the Importance of Countermeasures - An Analysis of the Xilinx Virtex-4 and Virtex-5 Bitstream Encryption Mechanism. In Proceedings of the Cryptographers' Track at the RSA Conference (CT-RSA), pp 1–18. Springer, 2012.

FPGA security
Secure remote configuration

- How do we avoid using the existing technology support?
- Architectures that allow "any" cryptographic protocol to be executed, use partial reconfiguration
  - Static part: communication + cryptography + reconfiguration control
  - Reconfigurable part: the application



- Challenge: store the bitstream in a secure manner

FPGA security
Secure remote configuration

- Store the bitstream in a secure manner
  - FPGA is trusted: bitstream storage in Block RAM
    - Use bitstream compression [3,4]
    - Re-encryption of the bitstream before storing it in external memory [5]
  - Board is trusted: bitstream storage in external memory
- Open challenge:
  - Vulnerability of the built-in decryption core that always performs the first decryption

[3] J. Vliegen, N. Mentens, and I. Verbauwhede, "Secure, remote, dynamic reconfiguration of FPGAs," ACM Transactions on Reconfigurable Technology and Systems 7(4), pp. 8:1-8:19, 2014.

[4] J. Vliegen, N. Mentens, and I. Verbauwhede, "A Single-chip Solution for the Secure Remote Configuration of FPGAs using Bitstream Compression," In 2013 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2013, R. Cumplido, E. De la Torre, and M. Wirthlin (eds.), IEEE Computer Society, pp. 1-6, 2013.

[5] H. Kashyap, and R. Chaves, "Secure partial dynamic reconfiguration with unsecured external memory," In the 24th International Conference on Field Programmable Logic and Applications, FPL 2014, pp. 1-7, 2014.

FPGA security
Secure IP core licensing

Design re-use
- Growing market of selling and buying efficient software and hardware cores
- Design re-use on FPGA: easy and dynamic integration
- Pitfall: loss of IP if no security measures are taken
- Solution: IP core licensing scheme
  - Online/offline schemes
  - Pay-per-use/pay-once schemes

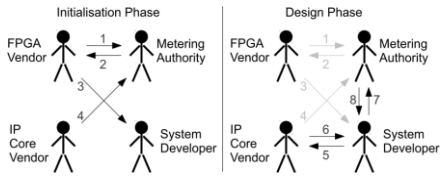
FPGA security
Secure IP core licensing

- Existing work on IP core licensing on FPGA:
  - Software cores on a processor embedded in the FPGA
  - Solutions for the implementation of only one hardware IP core
- Our work:
  - Practical pay-per-use solution for the implementation of many IP cores on one FPGA [6,7]

[6] R. Maes, D. Schellekens, and I. Verbauwhede. "A Pay-per-Use Licensing Scheme for Hardware IP Cores in Recent SRAM based FPGAs," IEEE Transactions on Information Forensics and Security, 7(1):98–108, 2012.

[7] J. Vliegen, D. Koch, N. Mentens, D. Schellekens, and I. Verbauwhede. "Practical feasibility evaluation and improvement of a pay-per-use licensing scheme for hardware IP cores in Xilinx FPGAs," Journal of Cryptographic Engineering 2014(1), pp. 1-10, 2014.

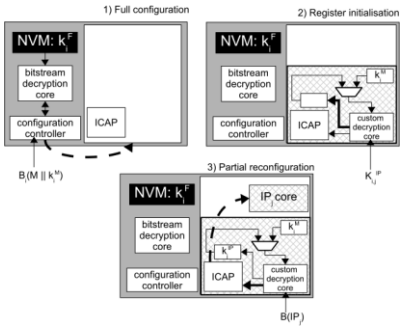
FPGA security
Secure IP core licensing



2 phases:
1. Initialization phase
2. Design phase

4 entities:
1. FPGA Vendor (FV)
2. Metering Authority (MA)
3. IP core vendor (IV)
4. System developer (SD)

FPGA security
Secure IP core licensing



FPGA security
Secure IP core licensing

- Issues in the original architecture:
  - Nested partial reconfiguration
    - The metering design is in the static part
    - The system developer's design is in the reconfigurable part
    - The IP core is a partial module in the system developer's design
    - This means we need nested partial reconfiguration, which is not provided by commercial design tools
  - Flexible placement of IP cores
    - The bitstream contains placement information
    - This means the IP core can only be placed at a predetermined position
    - Flexible placement is also not provided by commercial design tools
- Solution: use the academic tool GoAhead [8]
- Additional improvement: key storage in the configuration memory instead of in the flip-flops in the reconfigurable fabric of the FPGA

[8] C. Beckhoff, D. Koch, and J. Tørresen. Go Ahead: A Partial Reconfiguration Framework. In IEEE 20th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), pages 37–44. IEEE, 2012.

Conclusion

- Many FPGA-specific challenges in security
- Partial/dynamic reconfiguration is important for security applications
- Continuous effort of FPGA vendors to provide security features
- New FPGA architectures pose new problems, but also new opportunities