RUHR-UNIVERSITÄT BOCHUM

Not available

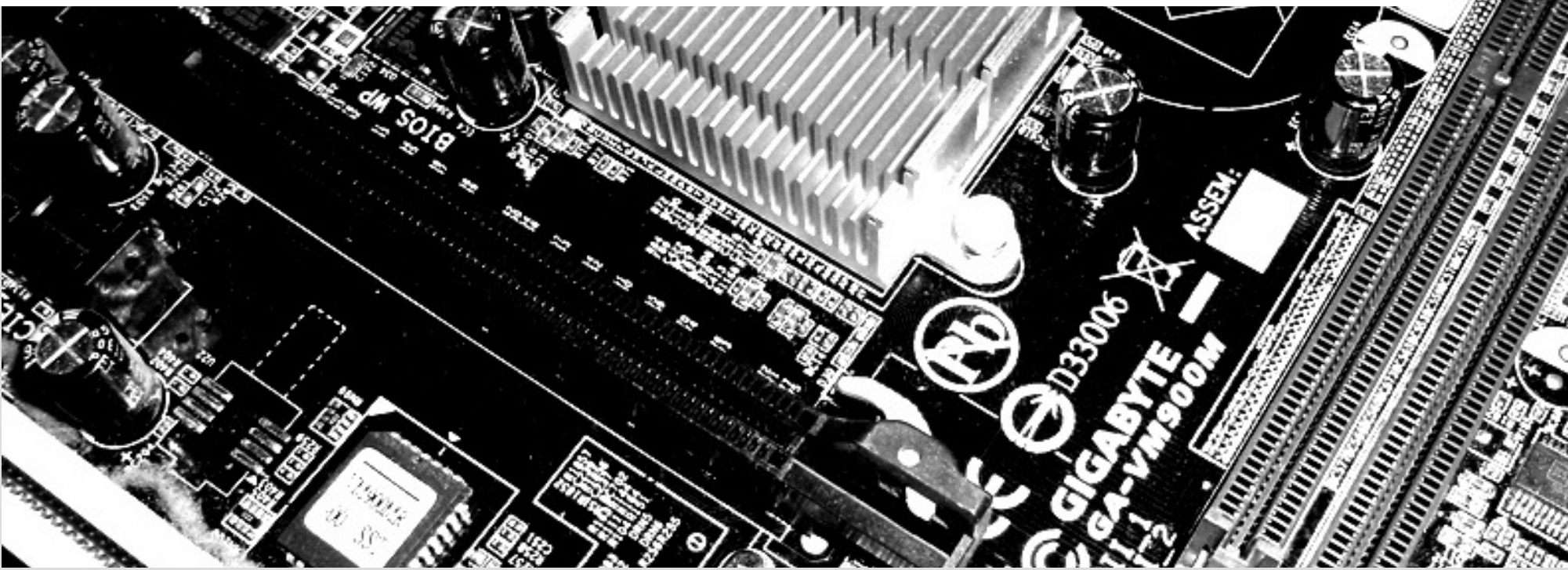
Corporate Design

High-Speed Implementation of bcrypt Password Search using Special-Purpose Hardware

10. December 2014

Horst Görtz Institute for IT-Security
Ruhr University Bochum

Friedrich Wiemer and Ralf Zimmermann



Outline

Motivation

Password Hashing Function?

Not available

Corporate Design

Can’t we just store passwords in plain?

[1] blog.ebay.com/ebay-inc-ask-ebay-users-change-passwords

[2] blogs.adobe.com/conversations/2013/10/important-customer-security-announcement.html

Motivation

Password Hashing Function?

Not available

Corporate Design

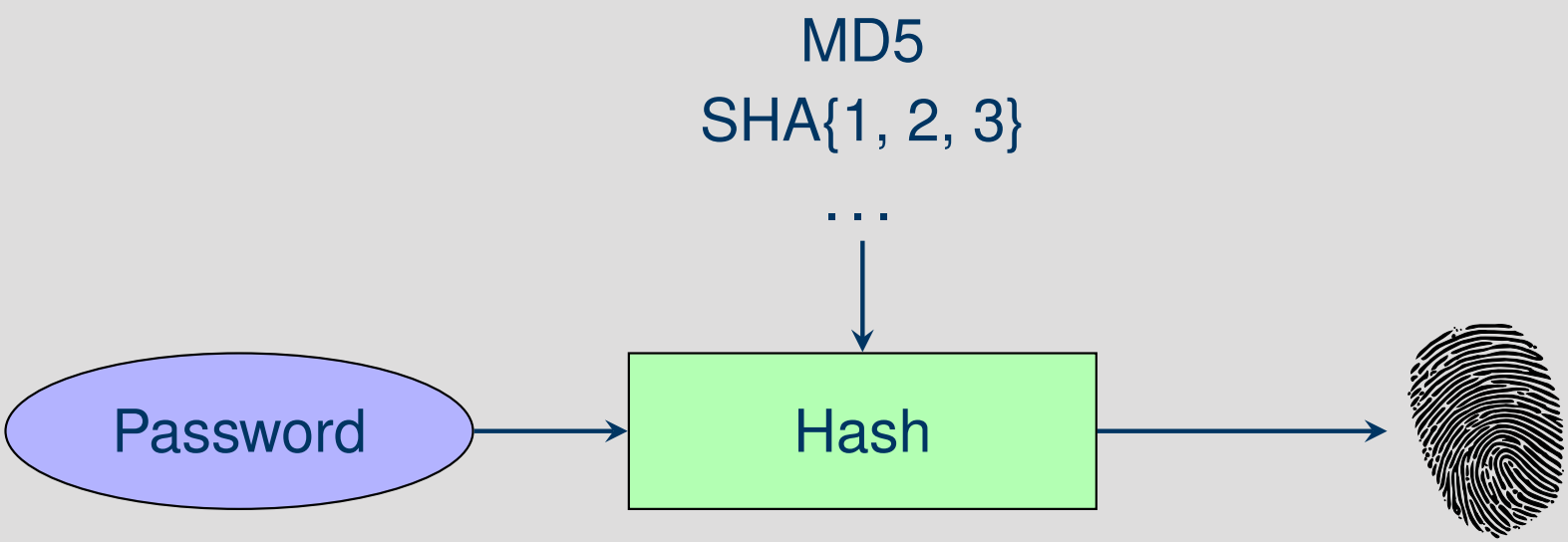
Can't we just store passwords in plain?[1][2]

[1] blog.ebay.com/ebay-inc-ask-ebay-users-change-passwords

[2] blogs.adobe.com/conversations/2013/10/important-customer-security-announcement.html

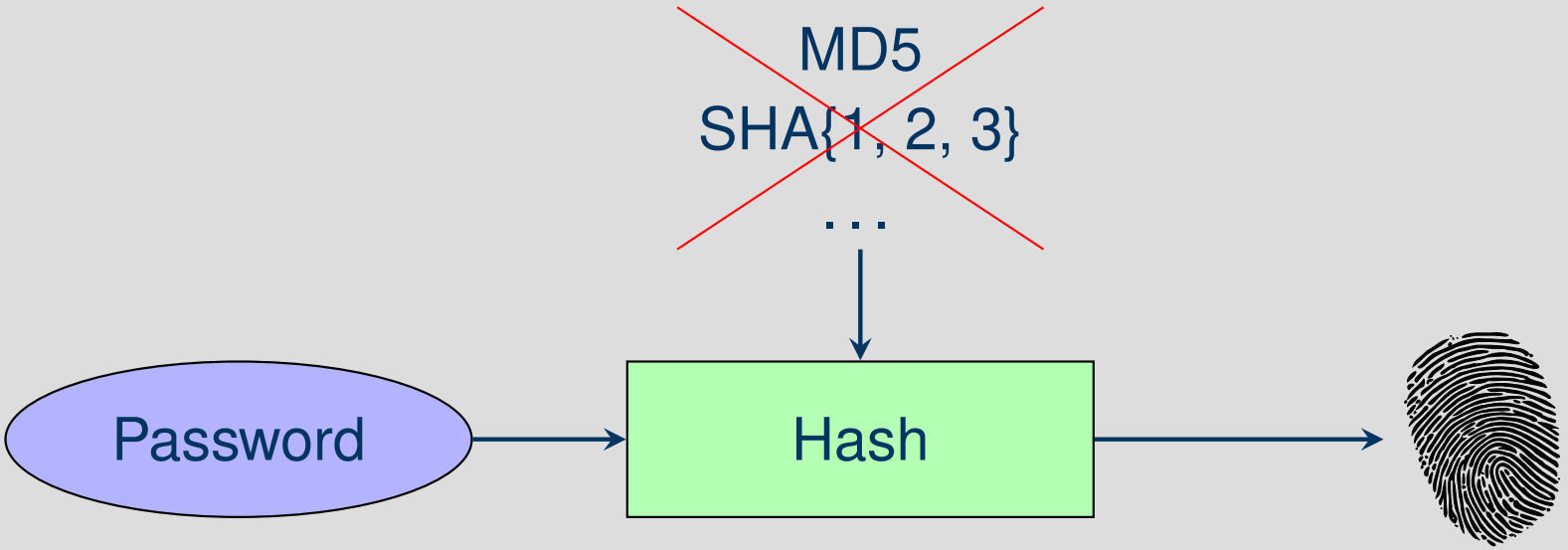
Motivation

Secure Storage?



Motivation

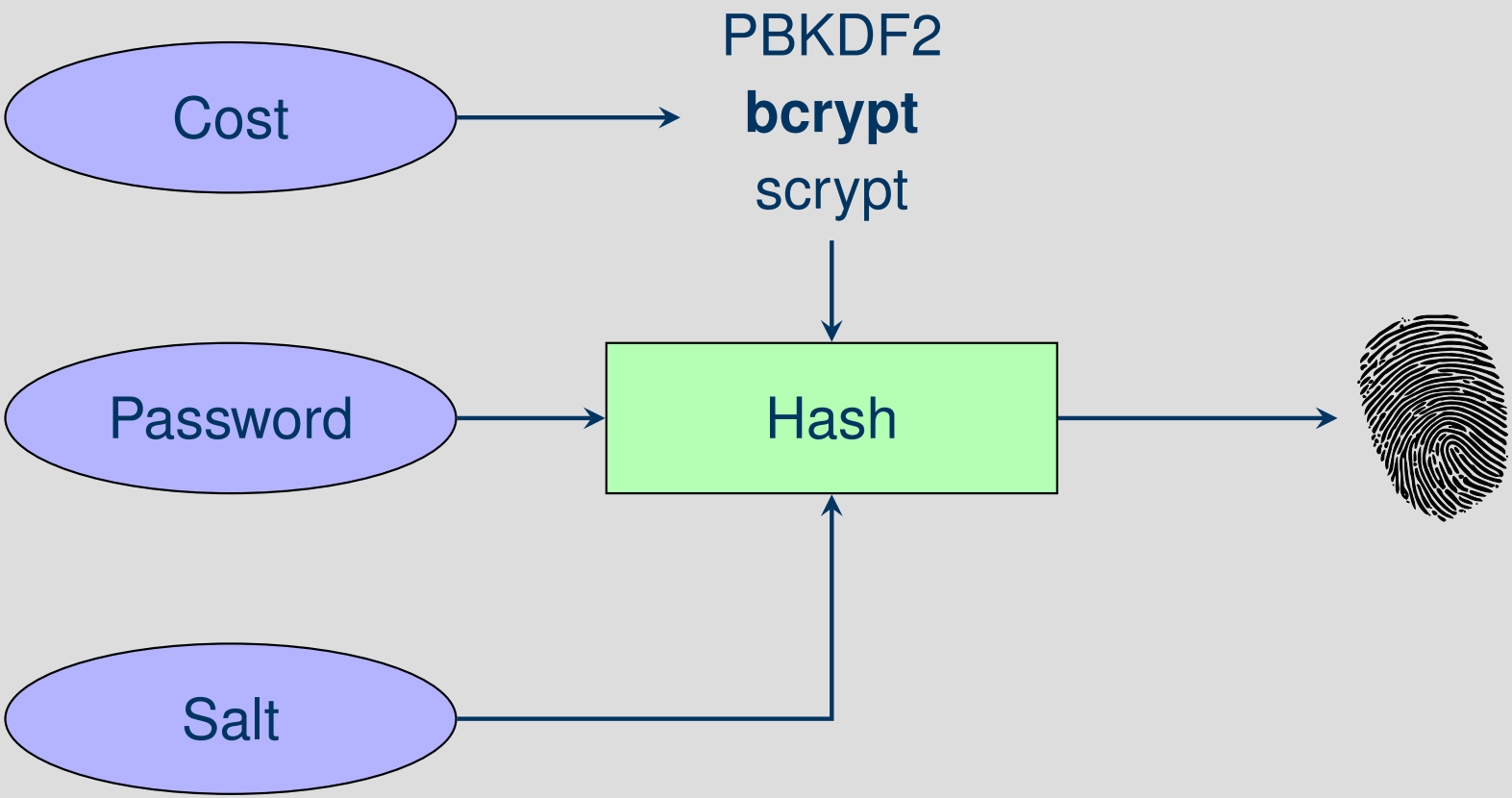
Secure Storage?



don't use standard hash functions

Motivation

Secure Storage!



Motivation

Why do we care?

- password cracking has an inherent parallel structure
- FPGAs enable to exploit this parallelism

- bcrypt claims to resist hardware optimizations
- currently available implementations[3] suffer from interface bottlenecks and instable operations

[3] K. Malvoni et al. Are Your Passwords Safe: Energy-Efficient Bcrypt Cracking with Low-Cost Parallel Hardware *8th USENIX Workshop on Offensive Technologies (WOOT 14), 2014*

What is bcrypt?

Introduced in 1999 by Provos and Mazières.[4] Implemented in OpenBSD 2.1, Ruby on Rails, and PHP as standard password hash.

bcrypt

- cost-parameterized
- based on modified Blowfish

[4] www.usenix.org/events/usenix99/full_papers/provos/provos.pdf

What is bcrypt?

Introduced in 1999 by Provos and Mazières.[4] Implemented in OpenBSD 2.1, Ruby on Rails, and PHP as standard password hash.

bcrypt

- cost-parameterized
- based on modified Blowfish

Blowfish

- symmetric blockcipher
- Feistel network

[4] www.usenix.org/events/usenix99/full_papers/provos/provos.pdf

bcrypt

Structure

- setup state, using the *password* and *salt* as key with modified Blowfish key schedule
- encrypt magic value
- output ciphertext as hash

bcrypt

Structure

- setup state, using the *password* and *salt* as key with modified Blowfish key schedule
- encrypt magic value
- output ciphertext as hash

Work

- needs $(2^{\text{cost}+1} + 1) \cdot 521$ Blowfish encryptions (roughly $2^{\text{cost}+10}$)
- needs $3 \cdot 64$ Blowfish encryptions

Implementation

Cracker

Not available

Corporate Design

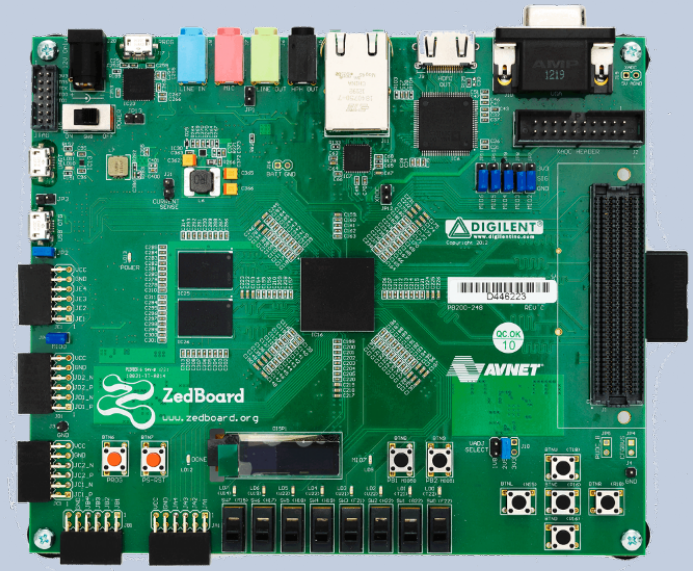
Target Platforms

Not available

Corporate Design

Low cost, low power FPGA

Zedboard



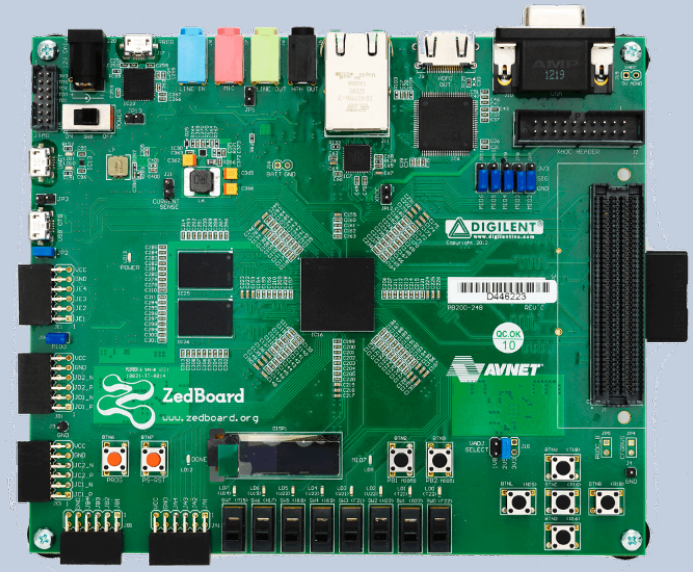
Target Platforms

Not available

Corporate Design

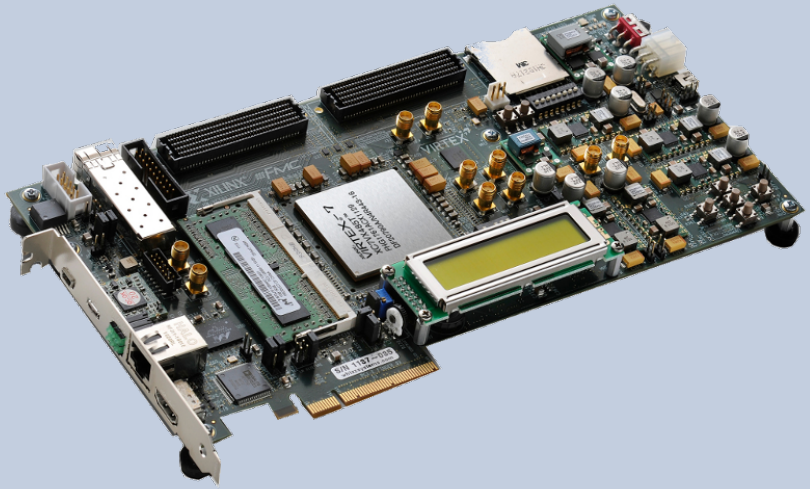
Low cost, low power FPGA

Zedboard



High Performance FPGA

Virtex-7



Optimization

Optimization Goal?

Low Area Footprint (bcrypt)

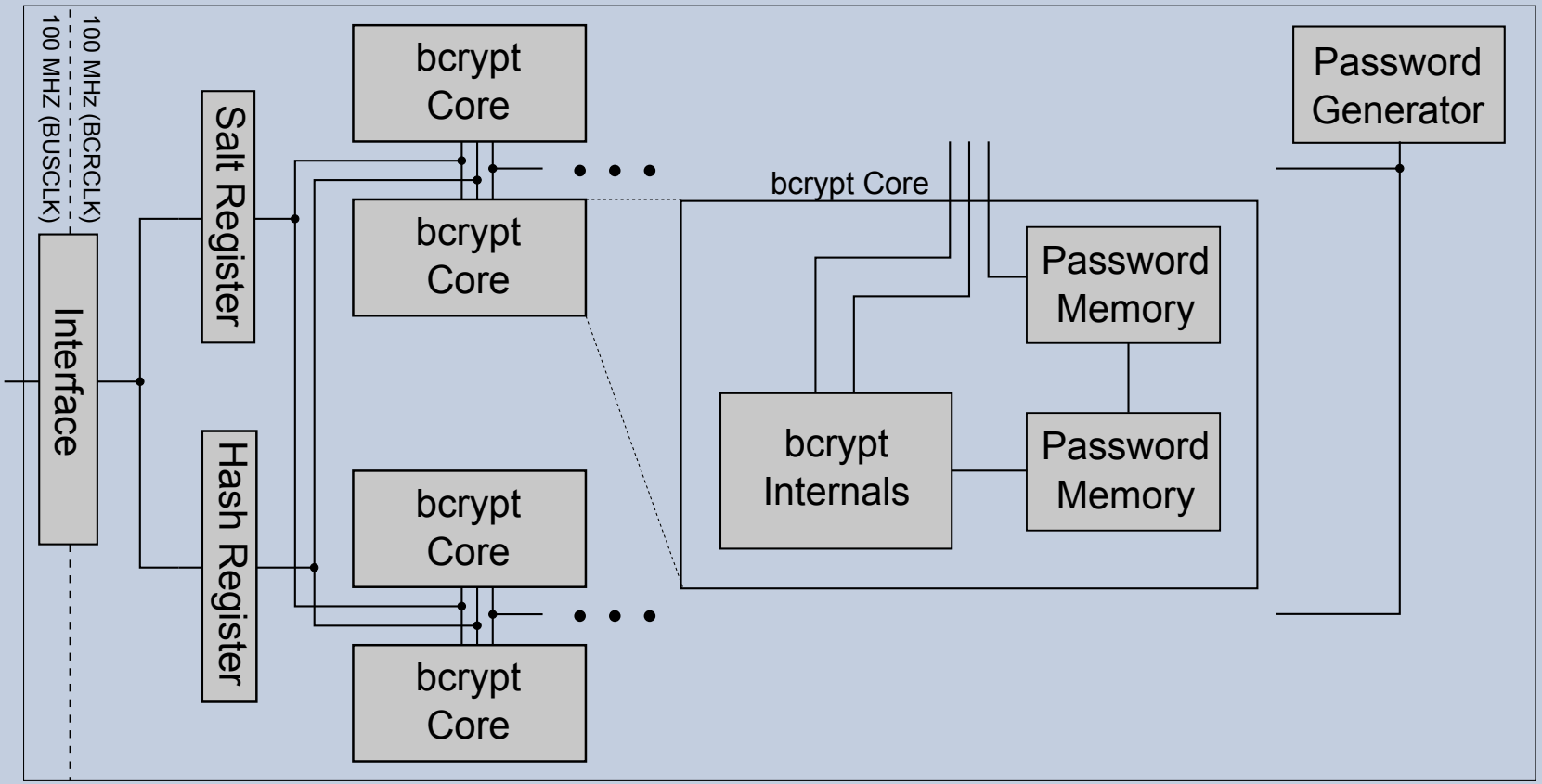
Low Area Footprint (bcrypt)
High-Speed (Blowfish)

Design

Not available

Corporate Design

First Attempt

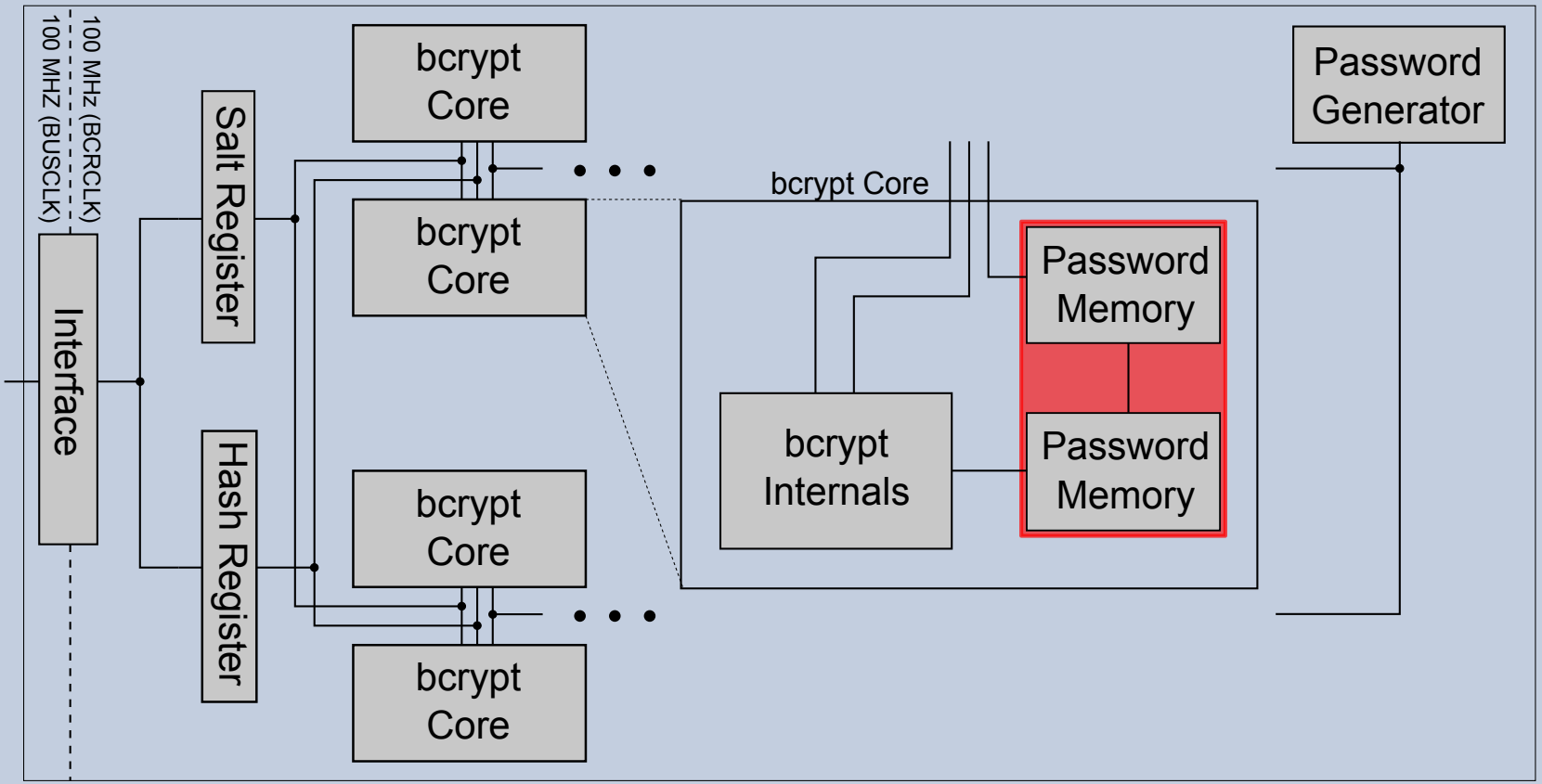


Design

Not available

Corporate Design

First Attempt

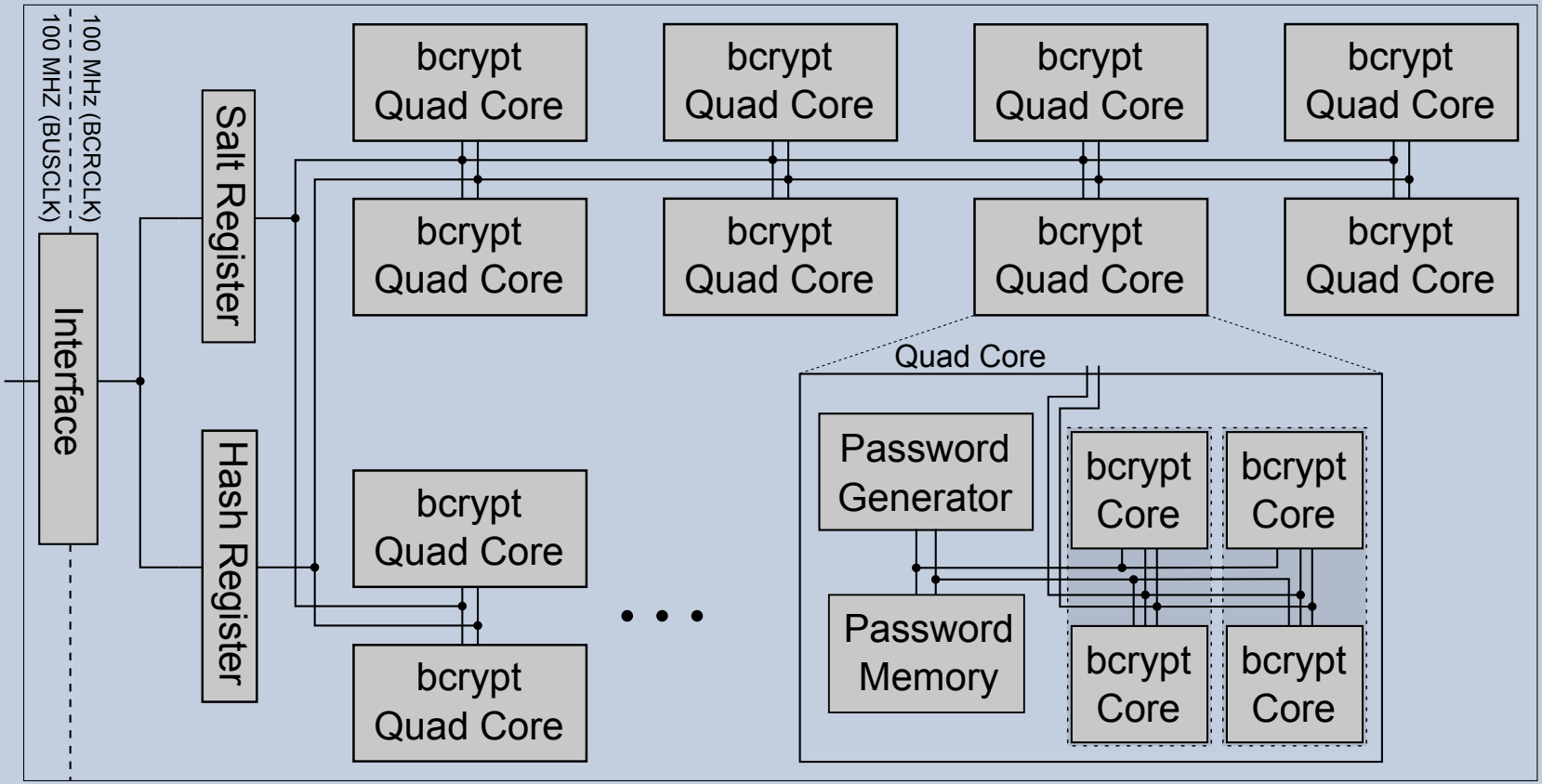


Design

Not available

Corporate Design

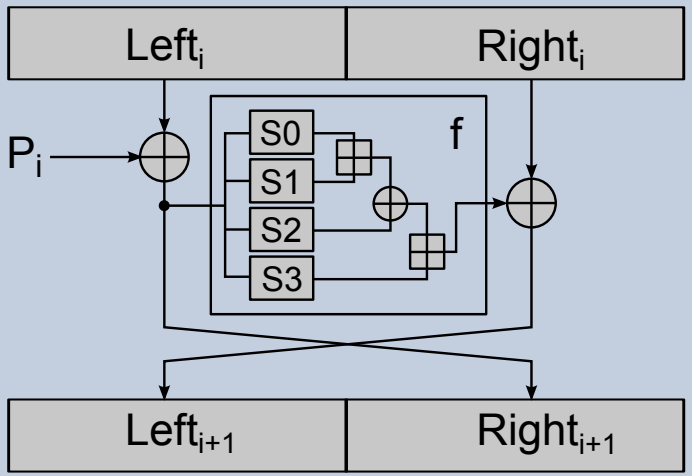
Quad Core



Design

Blowfish Core

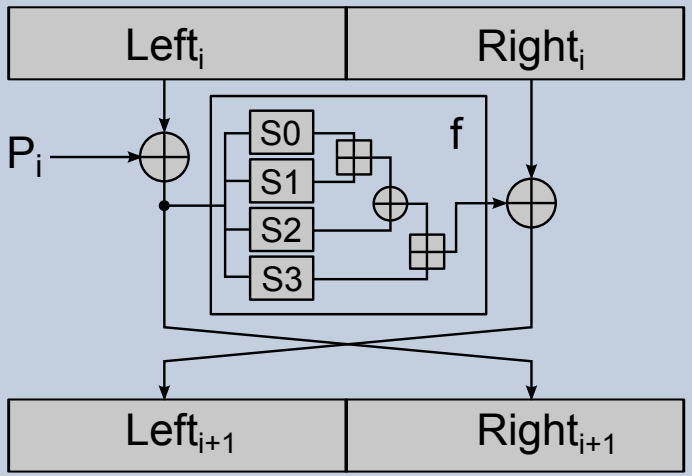
One Round



Design

Blowfish Core

One Round



Problematic

- SBox addresses can not be computed in the same clock as the look up is used
- needs 2 clock cycles per round

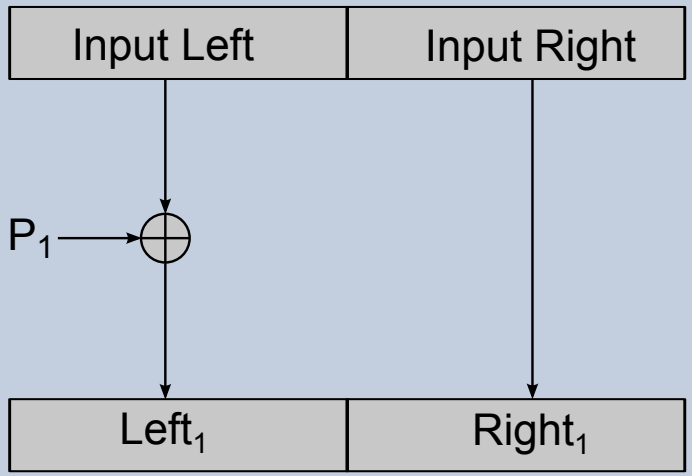
Design

Blowfish Core Retimed

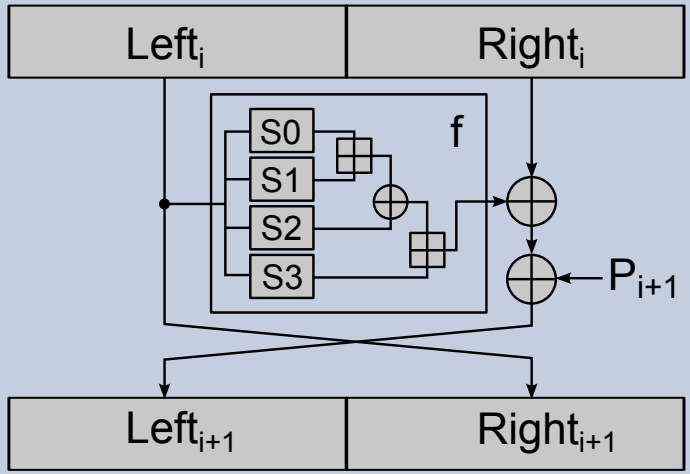
Not available

Corporate Design

Prefetch



Retimed Round



Advantages

- needs only 1 clock per round

Resulting Resources

Zedboard

- estimations for one zedboard:

  40 cores as upper bound, BRAMs as limiting resource
- first design attempt (password in registers):

  12 cores fit, LUT utilization way to high
- Quad Core Design:

  40 cores fit, while using “big” interface

Virtex-7

- Quad Core Design:

  316 cores per FPGA

Resulting Resources

Resource utilization of design and submodules

	LUT	FF	Slice	BRAM
Overall	64.8%	13.06%	93.29%	95.71%
Quad Core	2,777	720	801	13
Single Core	617	132	197	3
Blowfish Core	354	64	71	0
Password Generator	216	205	81	0

Resulting Hashrates

Not available

Corporate Design

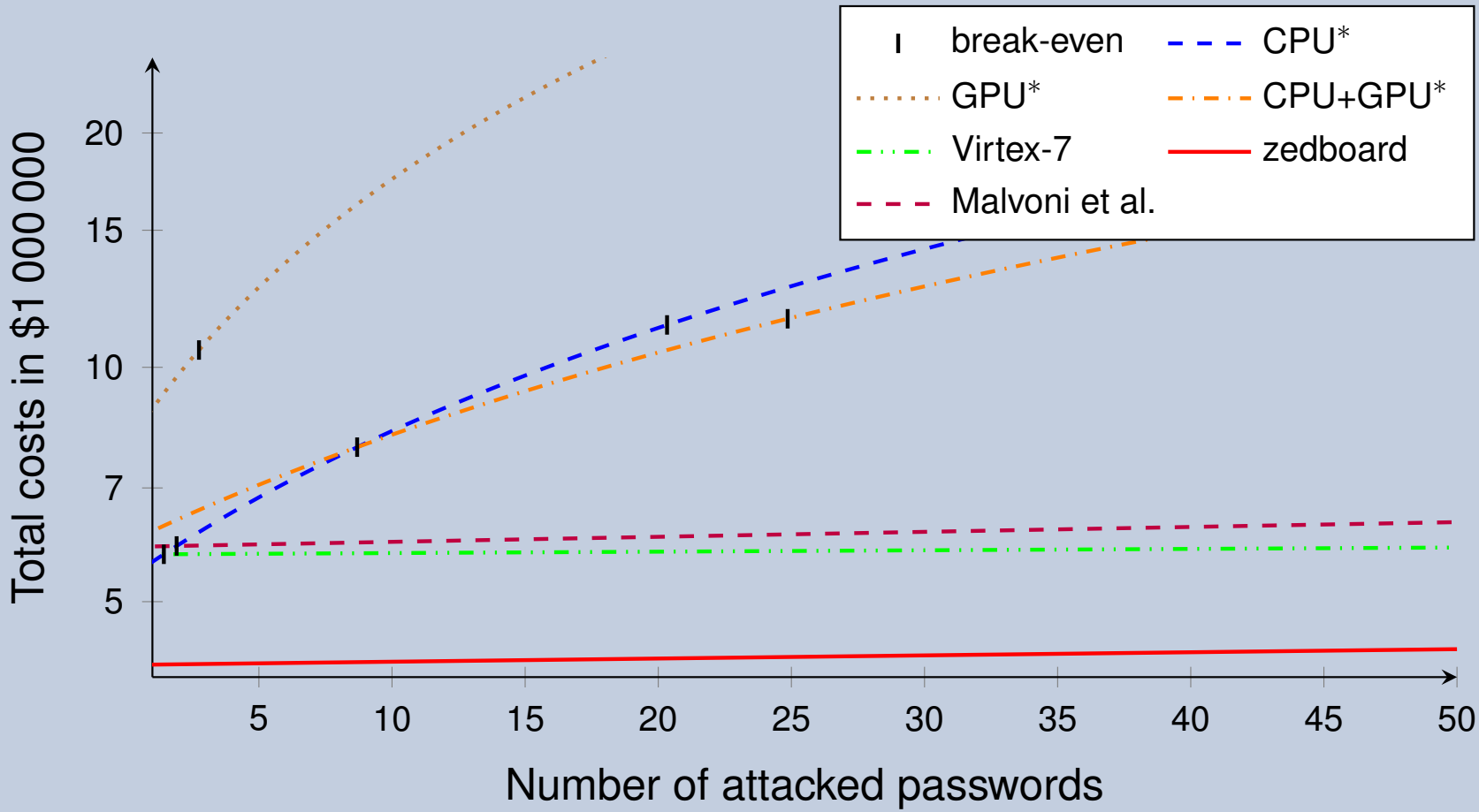
Compared to

	cost factor 5		cost factor 12	
	Hashes / Second	Hashes / Watt Second	Hashes / Second	Hashes / Watt Second
Zedboard	**6,511**	**1,550**	**51.95**	**12.37**
Malvoni (GSoC)	780			
Malvoni et al.	4,571	682.24	64.83	9.68
Virtex-7	51,437	2,572	410.4	20.52
Xeon E3-1240	6,210	20.7	50	0.17
GTX 750 Ti	1,920	6.4	15	0.05

Brute Force Attack

Not available Corporate Design

Cost 5



Questions?

Thank you for your attention!

Not available

Corporate Design

Images: Wikimedia Commons, flickr