FPGA implementation of PHOTON

Navaneeth Garakahalli
ECE464 – Fall 2013
Final Project Presentation

Contents

- Introduction
- PHOTON – An Overview
- FPGA Implementation
- Results

Introduction

Introduction

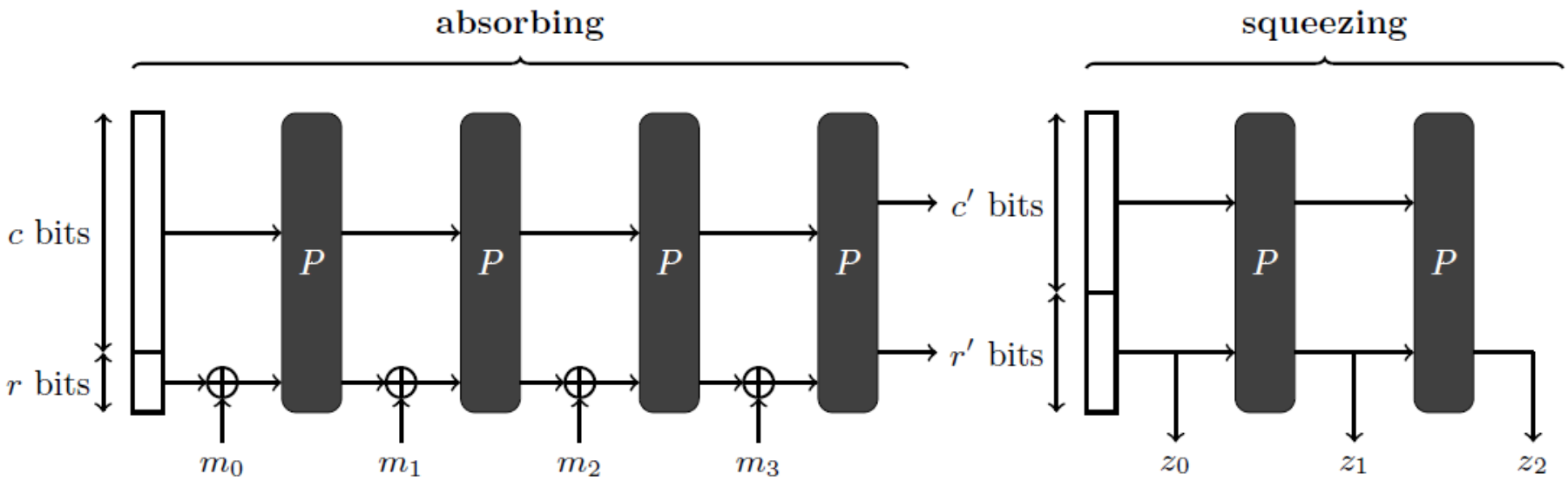
- Based on the paper entitled “The PHOTON Family of Lightweight Hash Functions” Authored by Jian Guo, Thomas Peyrin, Axel Poschmann
- Presented at CRYPTO 2011
- Lightweight hash function
- Structure based on AES
- The paper proposes 5 different flavors of the hash function; each with different lengths of input & output bit rates and hash value
- The hash function with input and output bit rates of 32bits (32 bits block is processed per iteration) is implemented here.
  The hash value generated is 256 bits long.

PHOTON – An overview

PHOTON – An overview

- Sponge Function
- Permutation
- State register
- Add Constants
- Sub Cells
- Shift Rows
- Mix Column Serial
- A PHOTON round

Sponge Function



Permutation

- All operations modify the contents of the State register
- Each permutation involves 12 rounds
- Each round consists of 4 operations:
  - -Add Constants
  - -Sub Cells
  - -Shift Rows
  - -Mix Column Serial

State register

- 288 bits wide; to be viewed as 6x6 matrix with cell size of 8 bits
- To be initilized to {240'b0,8'h40,8'h44,8'h44}
- Input message block is xor-ed to first 4 cells of the state matrix

6 Columns

6 Rows							
	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits
	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits
	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits
	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits
	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits
	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits	8 bits

Add Constants

- Round constants and Internal constants are generated by Linear Feedback Shift Registers
- The generated constants are added to first column of the state matrix
- Galois field addition is performed

Sub Cells

- Based on the flavor of the hash funciton, AES Sbox or PRESENT Sbox is used
- Each element in the state matrix is substituted by a value from the Sbox

Shift Row

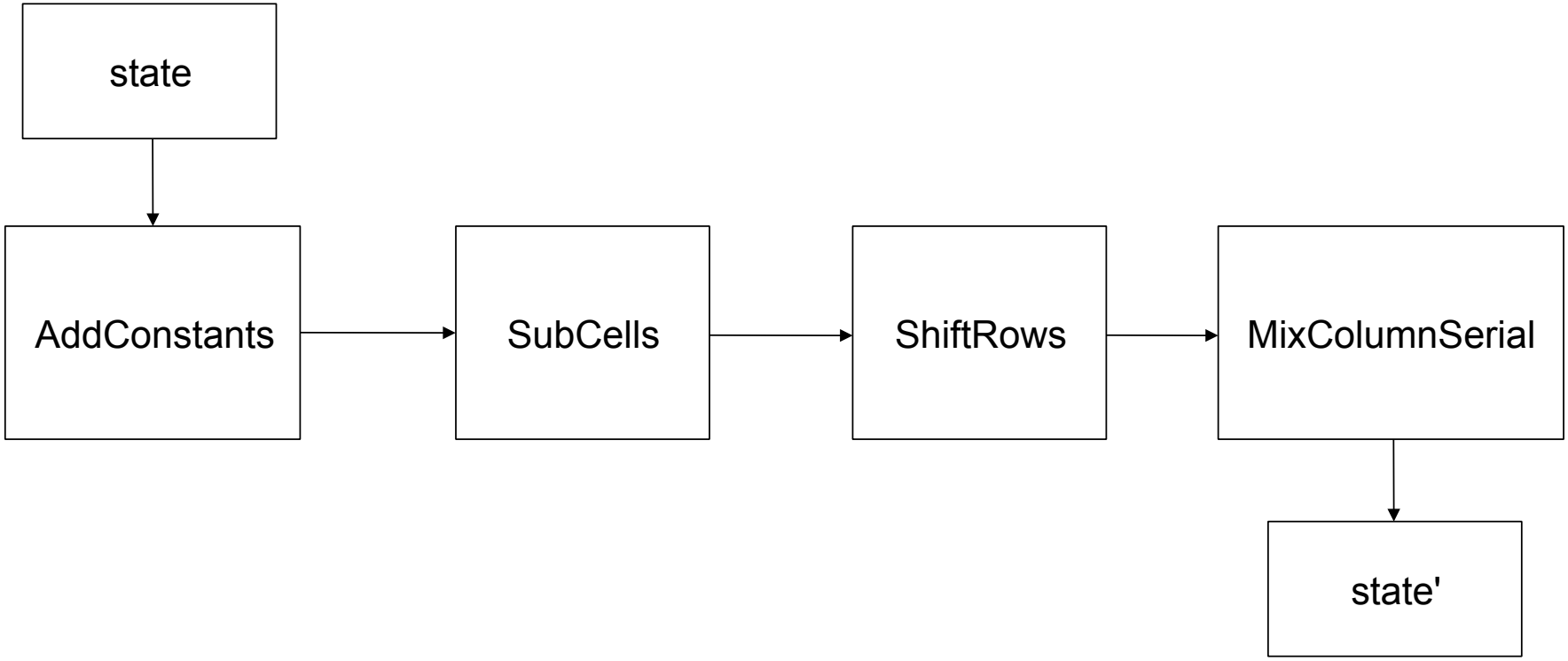
- Each cell in a row *i* of the state matrix is rotated to the left by *i* positions
- The rows are numbering starting from 0

Mix Column Serial

- The state matrix is multiplied by a MDS matrix (which is raised to a fixed power 6)
- The multiplication is carried out in Galois Field

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 3 & 1 & 2 & 1 & 4 \end{pmatrix}^6 = \begin{pmatrix} 2 & 3 & 1 & 2 & 1 & 4 \\ 8 & 14 & 7 & 9 & 6 & 17 \\ 34 & 59 & 31 & 37 & 24 & 66 \\ 132 & 228 & 121 & 155 & 103 & 11 \\ 22 & 153 & 239 & 111 & 144 & 75 \\ 150 & 203 & 210 & 121 & 36 & 167 \end{pmatrix}$$

A PHOTON round

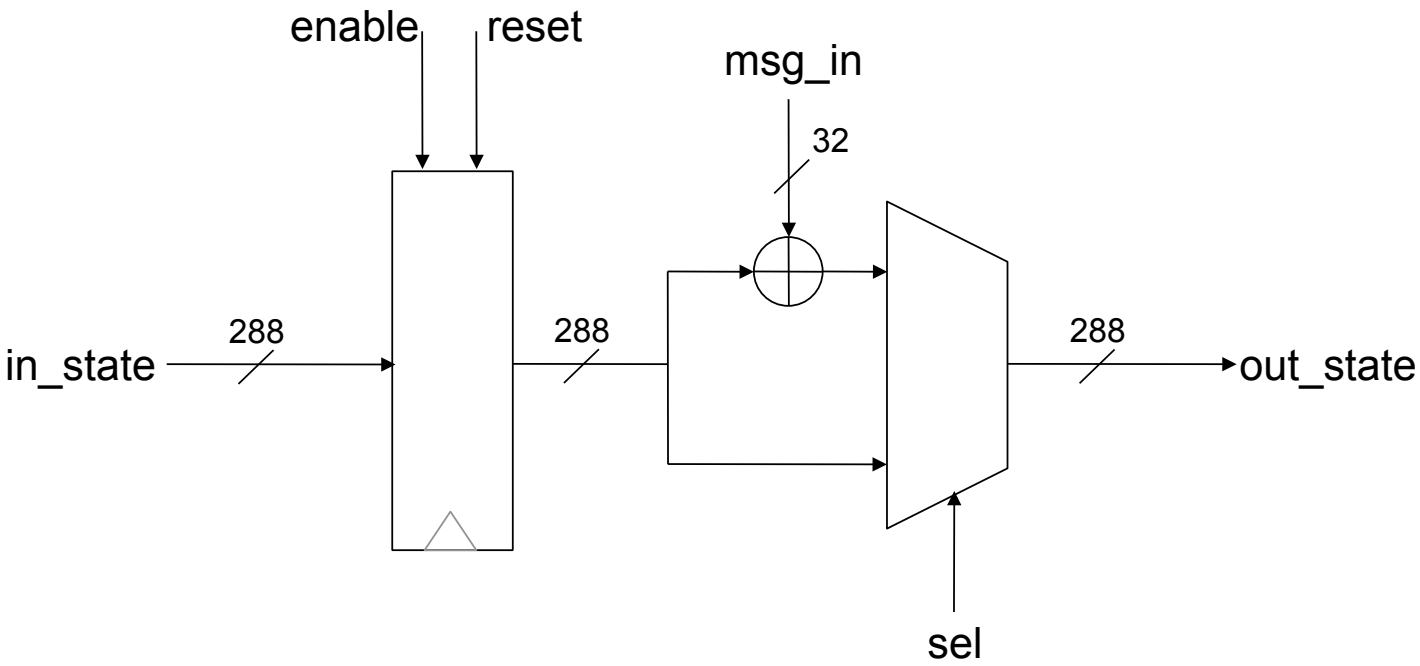


FPGA implementation

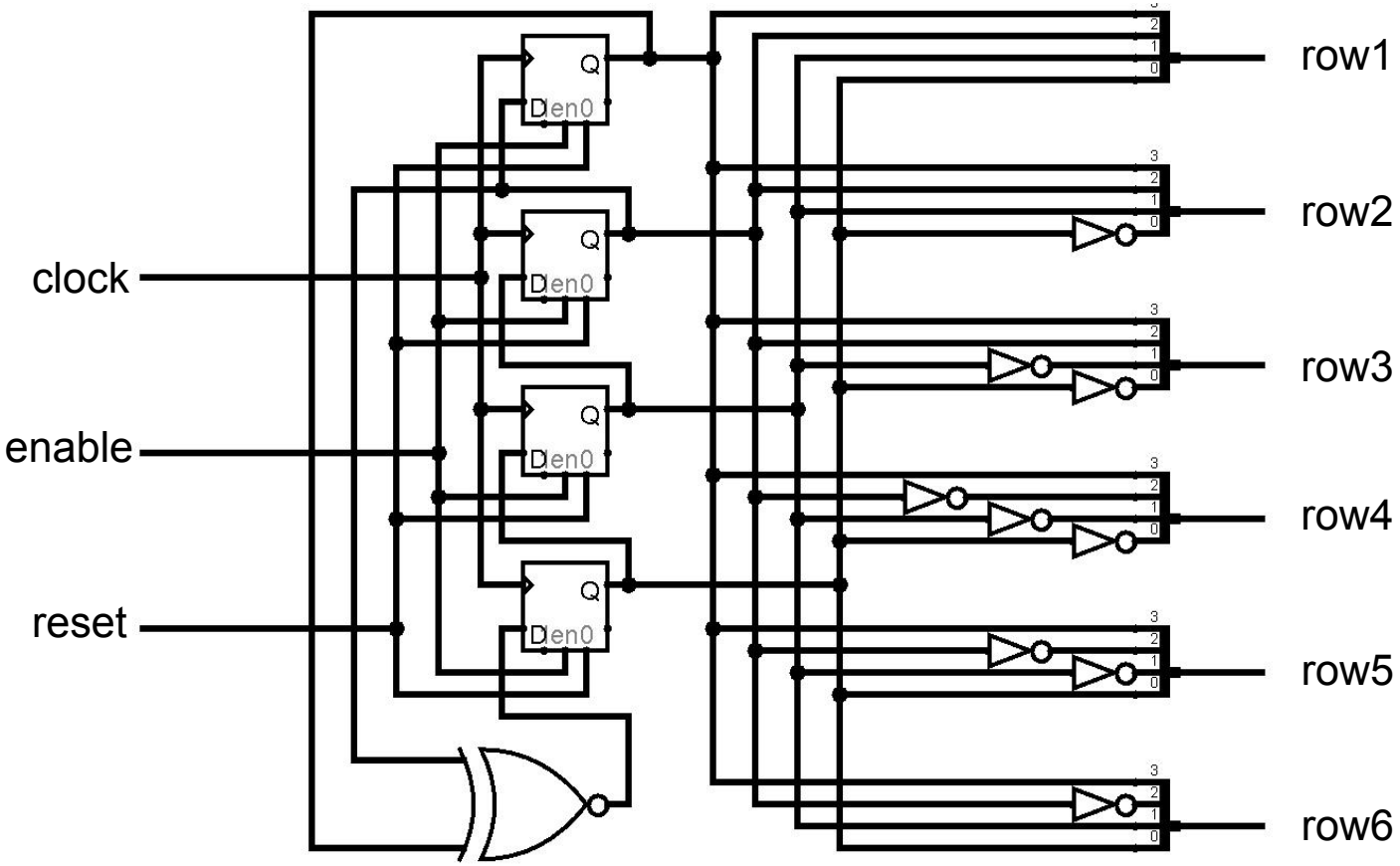
FPGA implementation

- Block diagram
- Interface
- Controller FSM
- Timing analysis
- Tools & Target hardware
- Coding & Testing

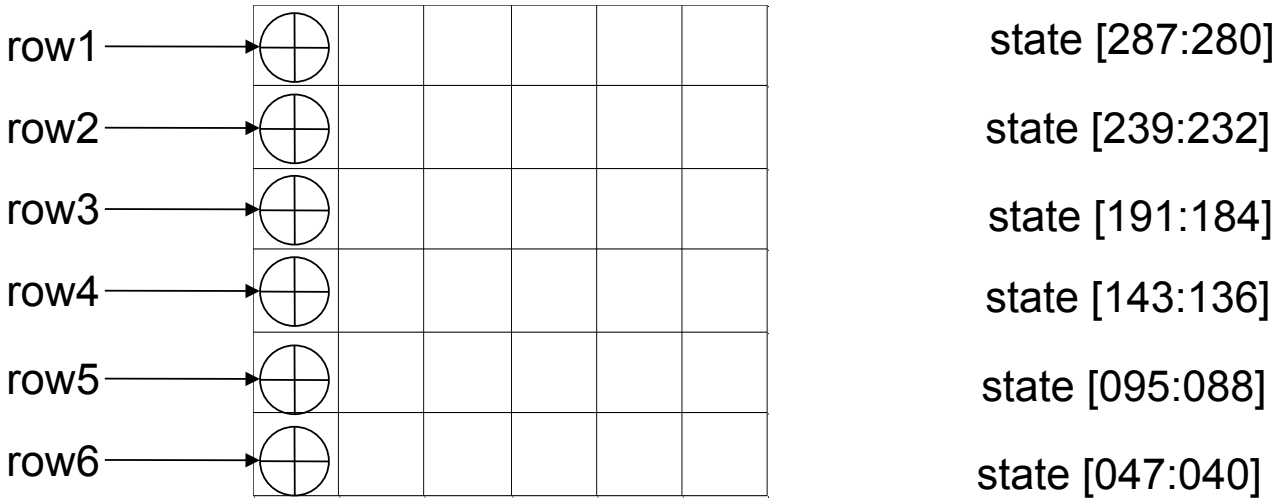
Block diagram



Block diagram

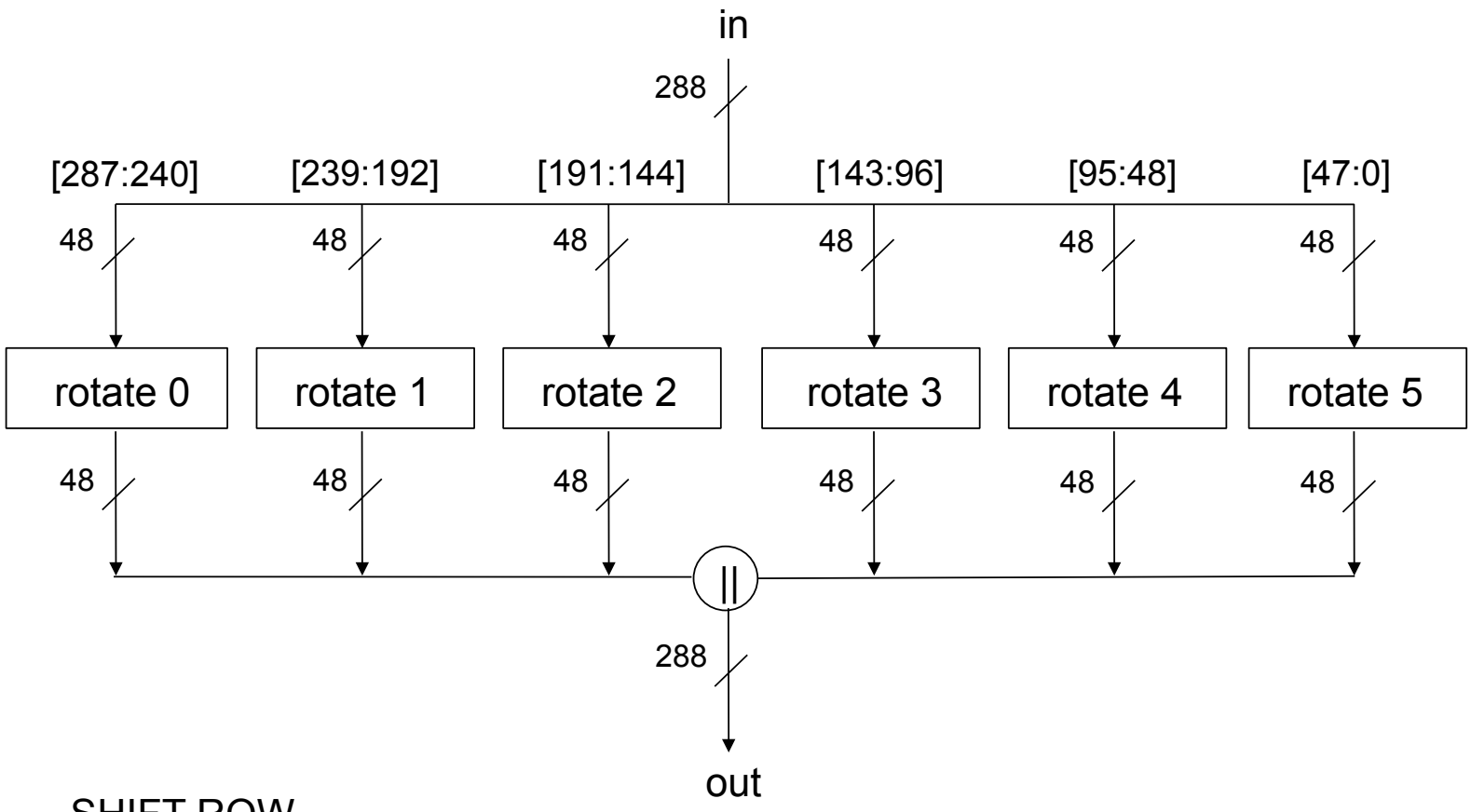


Block diagram



ADD CONSTANTS

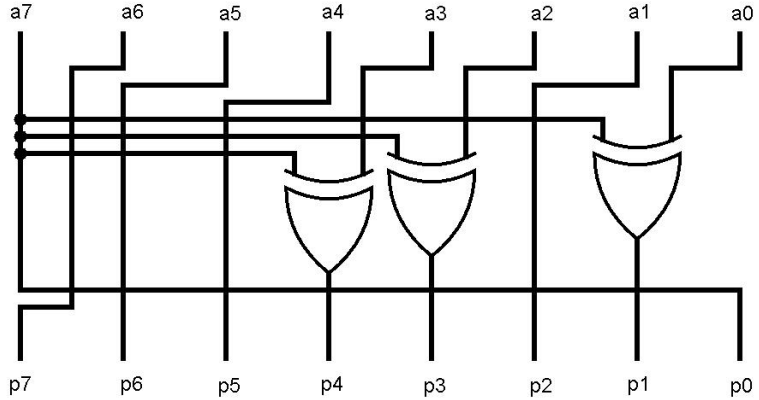
Block diagram

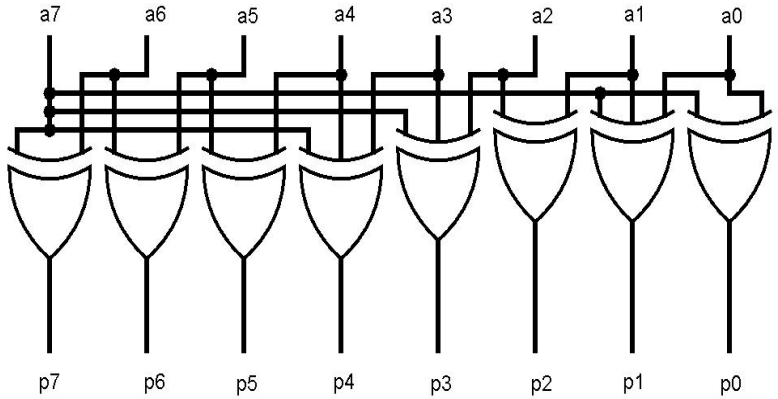


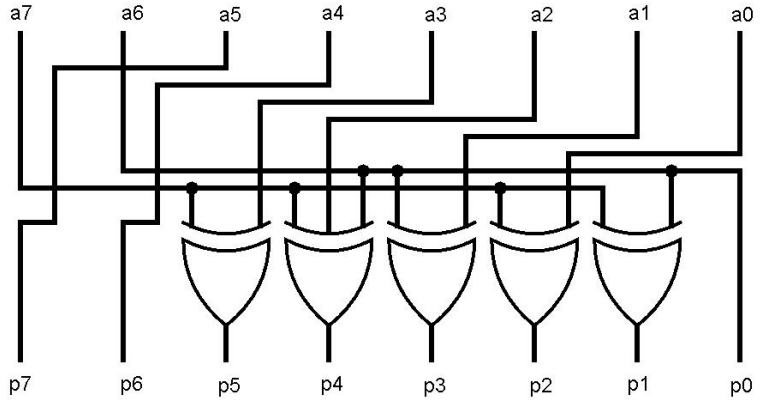
SHIFT ROW

Block diagram

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 3 & 1 & 2 & 1 & 4 \end{pmatrix}$$

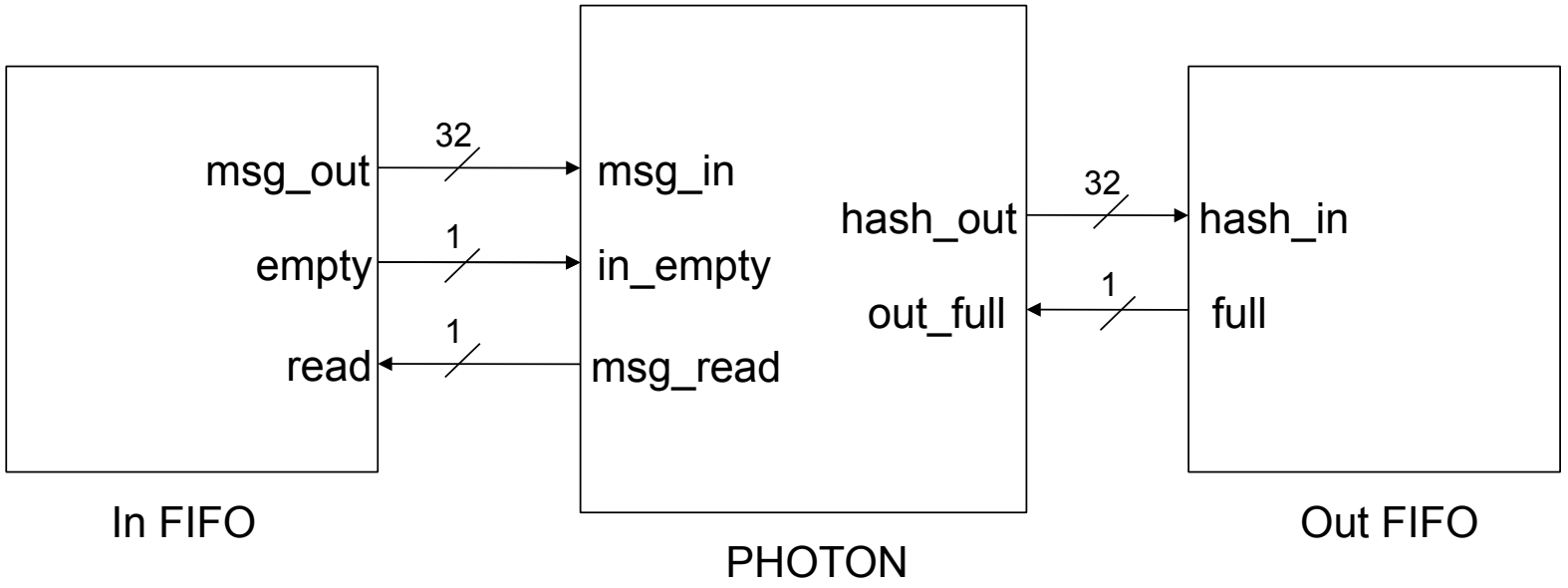




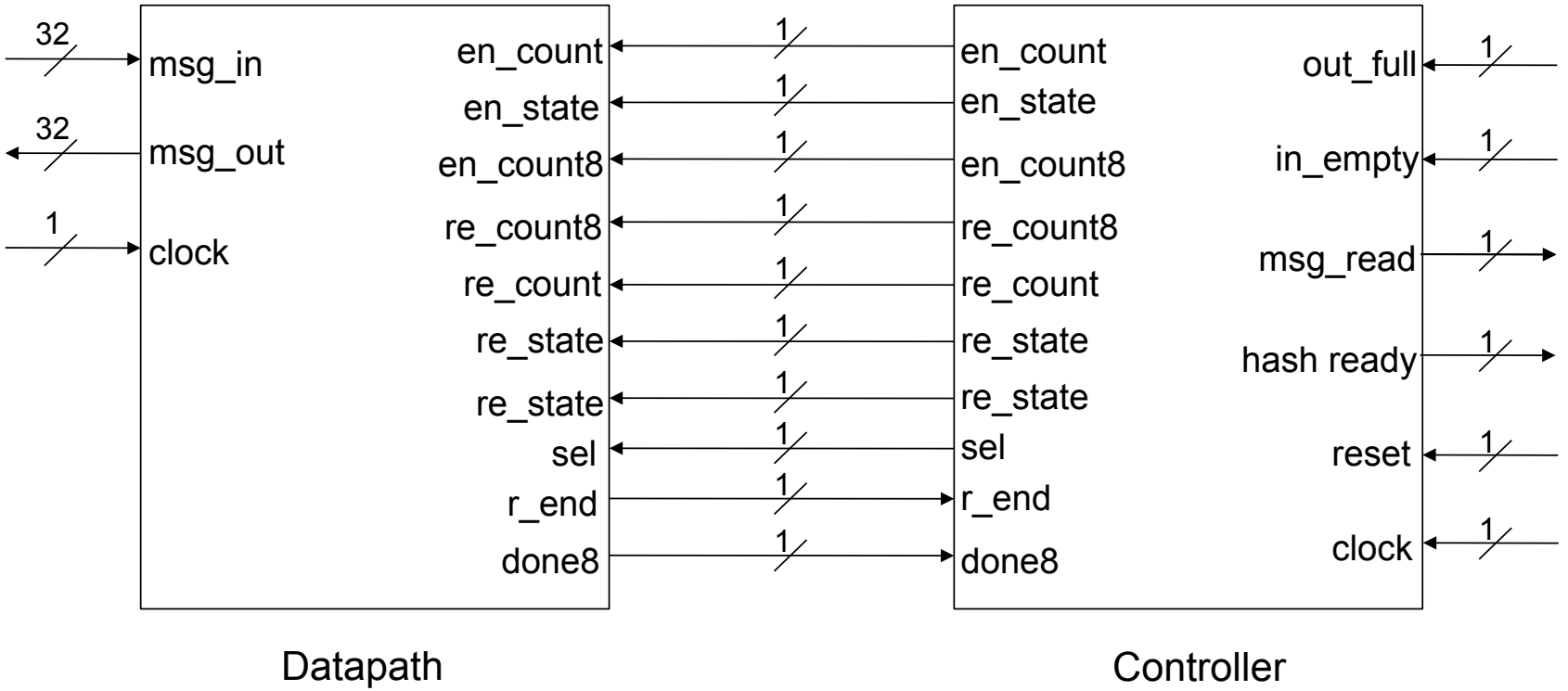


MIX COLUMN SERIAL

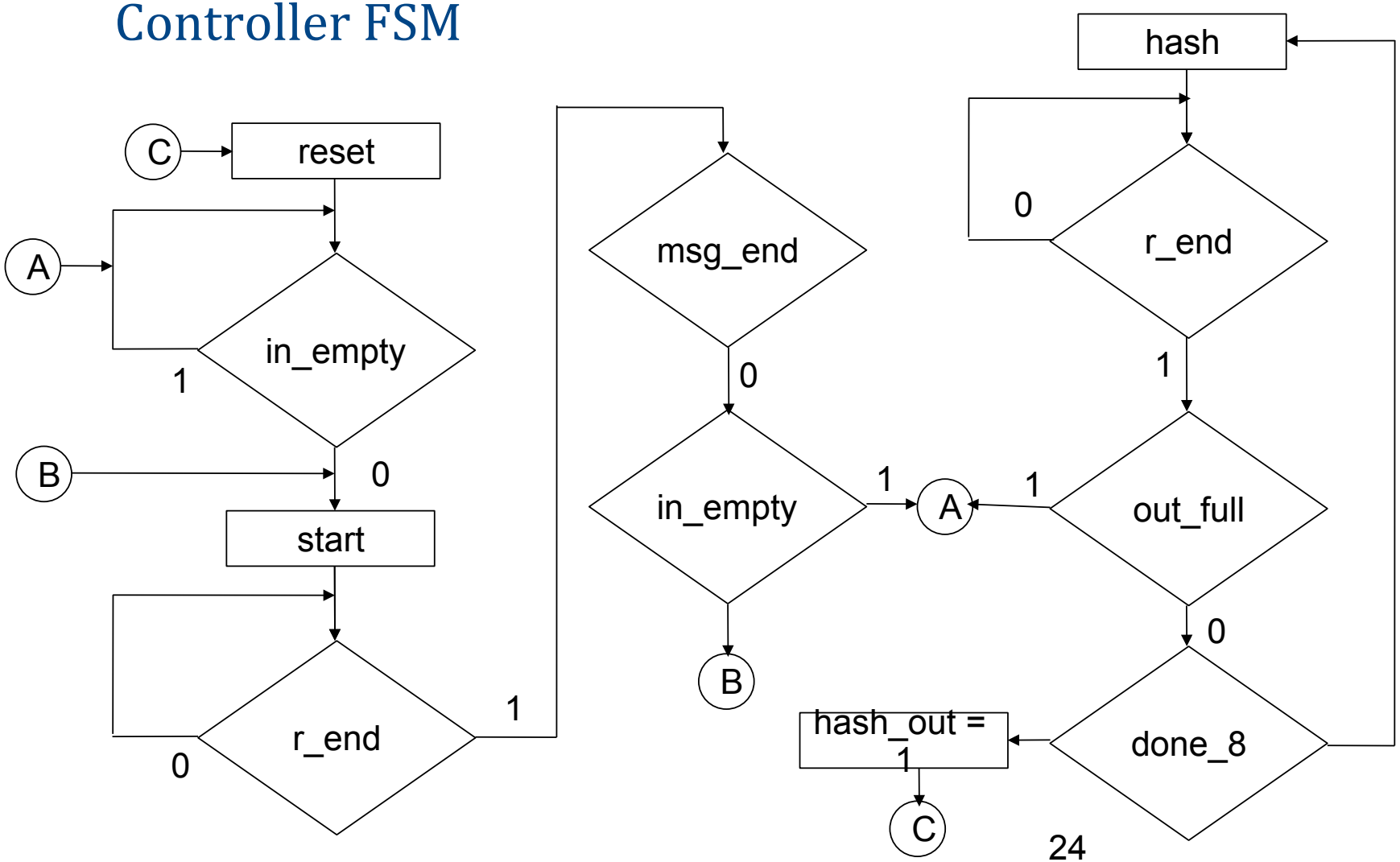
Interface


msg_out
empty
read
32
1
1
msg_in
in_empty
msg_read
hash_out
out_full
32
1
hash_in
full
In FIFO
PHOTON
Out FIFO

Interface


32
msg_in
32
msg_out
1
clock
en_count
en_state
en_count8
re_count8
re_count
re_state
re_state
sel
r_end
done8
1
1
1
1
1
1
1
1
1
1
en_count
en_state
en_count8
re_count8
re_count
re_state
re_state
sel
r_end
done8
out_full
in_empty
msg_read
hash ready
reset
clock
1
1
1
1
1
1
Datapath
Controller

Controller FSM



Timing analysis

- Latency : 20 clock cycles
- Time between 2 inputs: 12 clock cycles
- Throughput:
  - Short messages – 2.5 bits/clock cycle
  - Long messages – 2.67 bits/clock cycle

Tools & Target Hardware

- Target hardware
  - Altera Cyclone IV - EP4CE15F17C6
  - Altera Cyclone V – 5CGXFC7C7F23C8

- Tools used
  - Quartus II
  - ModelSIM
  - Powerplay Power analyzer
  - Eclipse

Coding & Testing

- Verilog HDL
- About 750 lines of code
- Software implementation in C was used to generate test vectors
- Testbenches written to test each module sepearted and also for testing the entire unit

Results

Results

- The design was implemented on Cyclone IV & Cyclone V FPGAs from Altera
- Cyclone IV:
  - The implementation resulted in a unit that could run at a maximum frequecy of 128 MHz
  - It occupied an area of 8793 Logic elements
  - A vectorless power estimation resulted in maximum power dissipation of 167.55 mW
- Cyclone V:
  - The implementation resulted in a unit that could run at a maximum frequecy of 109 MHz
  - It occupied an area of 2129 Adaptive Logic Modules (ALM)
  - A vectorless power estimation resulted in maximum power dissipation of 145.81 mW

Thank you