Design and Development of an FPGA Based Intelligent Stereo Vision System

Bastian Nordmeyer
Olaf Lüke

5. Mai 2010

Agenda

GET Lab

Motivation

Basics

System Design

Stereo Vision System

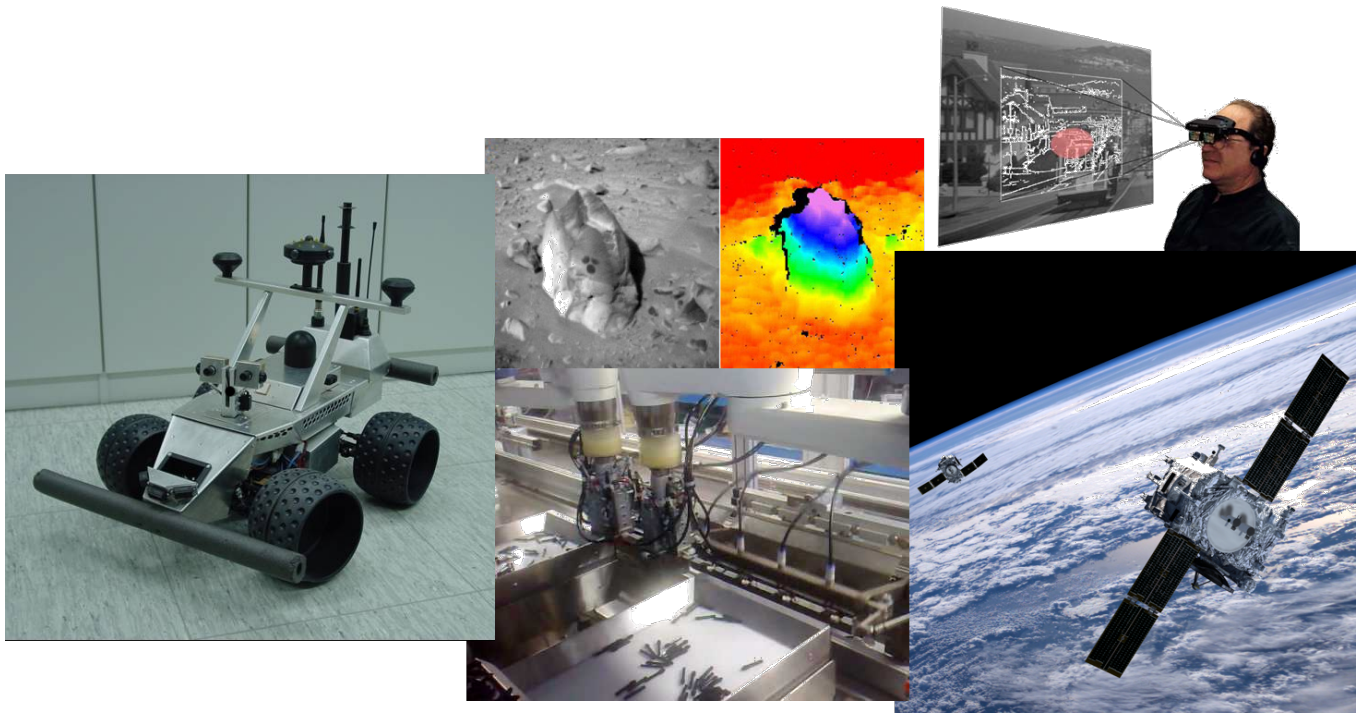
Results

Summary and Outlook

Motivation: Stereo Vision

GET Lab

- **Definition:** Stereo vision is the extraction of 3D information from images obtained by two cameras.
- Stereo vision is important in different fields:
  - e. g. aid for visually impaired, automated systems, robotics, space exploration



Motivation: FPGA

GET Lab

How does a stereo vision algorithm can be computed?

	Processor	FPGA
Single computation performance	●●	○●
Parallel computation performance	○●	●●
Costs	●●	●●
Usage simplicity	●●	○●

Decision

- Stereo vision algorithms are mostly parallalizable
- $\Rightarrow$ FPGAs are well suitable

Motivation:Tasks

GET Lab

Tasks

- Development of an prototyping stereo vision system
  - Generic and modular design
  - Cost-efficient system
  - Hardware choosing and construction

Agenda

GET Lab

Motivation

Basics
- FPGA
- Impulse C

System Design

Stereo Vision System

Results

Summary and Outlook

Basics: FPGA

What is an FPGA?

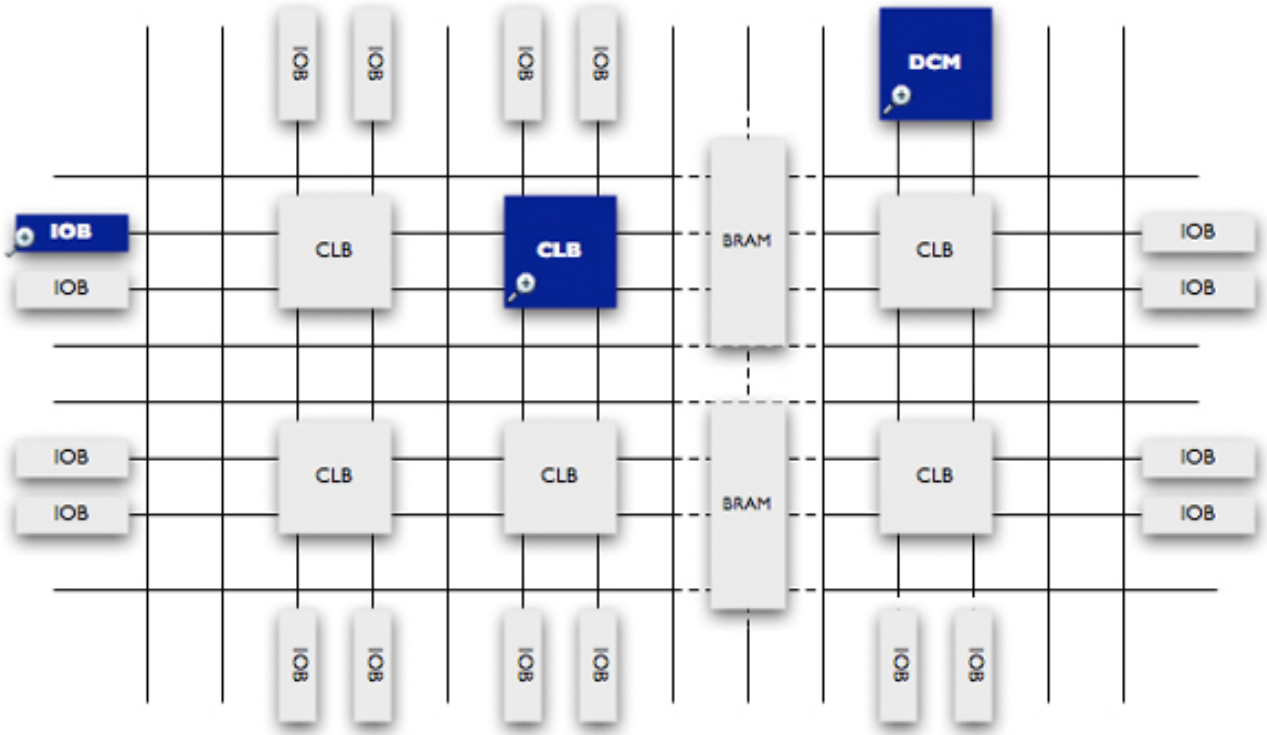
- **F**ield-**P**rogrammable **G**ate **A**rray
- Integrated Circuit
- Configurable behaviour
  - typically defined by Hardware Description Languages (HDL)



Basics: FPGA

GET Lab

- FPGA - Internal Structure

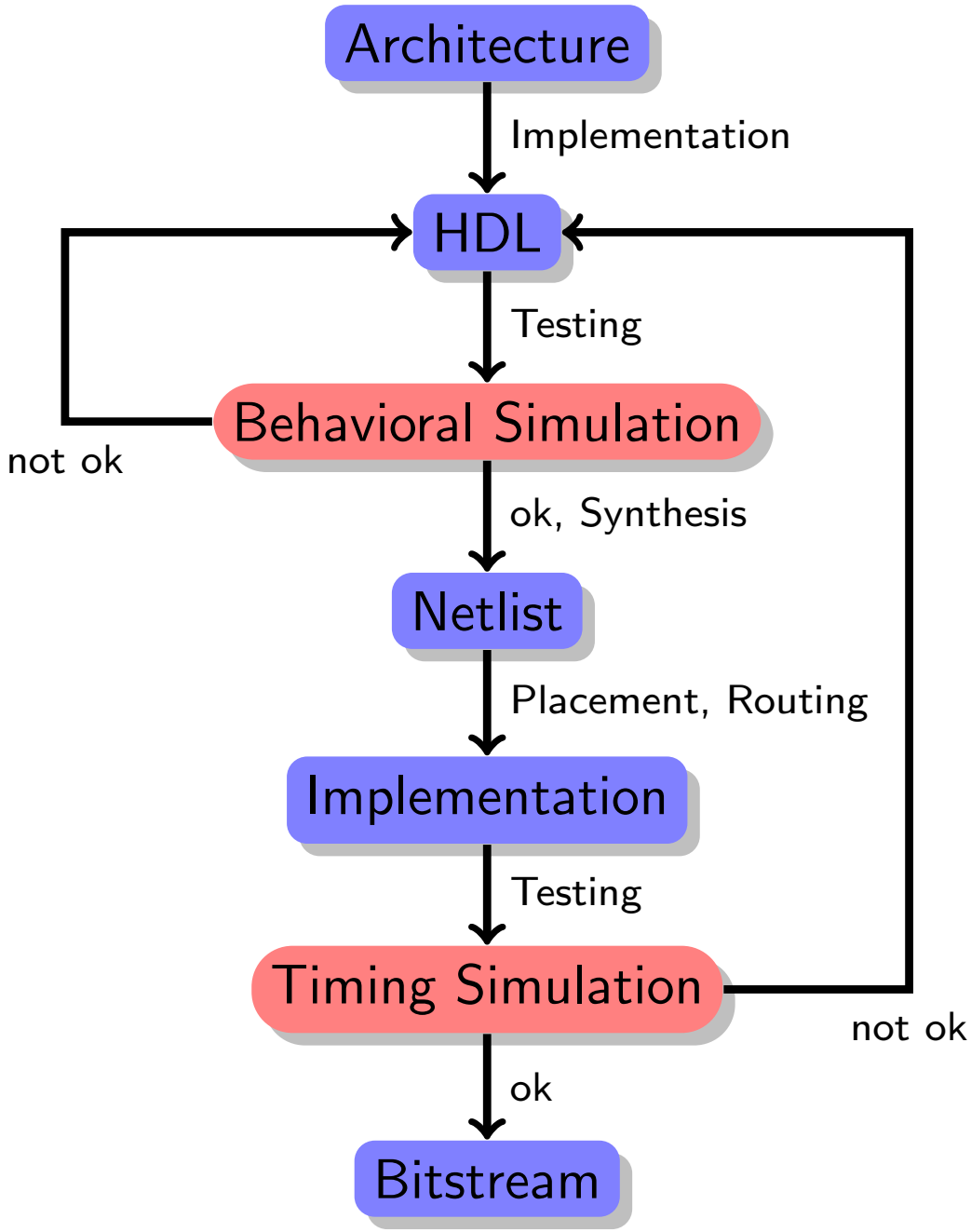


IOB Input/Output Block, **CLB** Configurable Logic Block, **BRAM** Block RAM, **DCM** Digital Clock Manager

- **FPGA:** *Circuit with a configurable structure and behaviour*

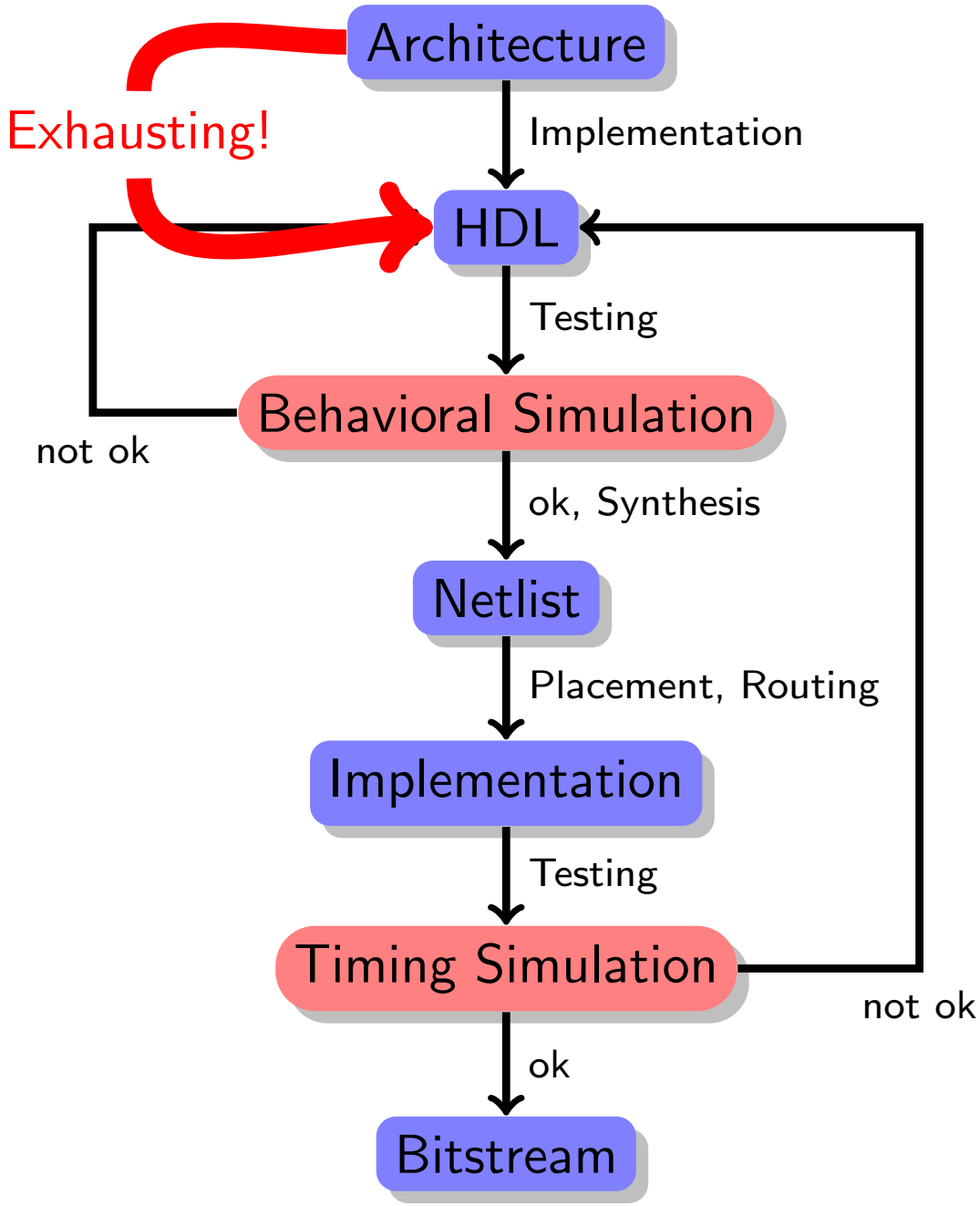
Basics: FPGA Workflow

GET Lab



Basics: FPGA Workflow

GET Lab



Basics: Impulse C C-to-VHDL Compiler

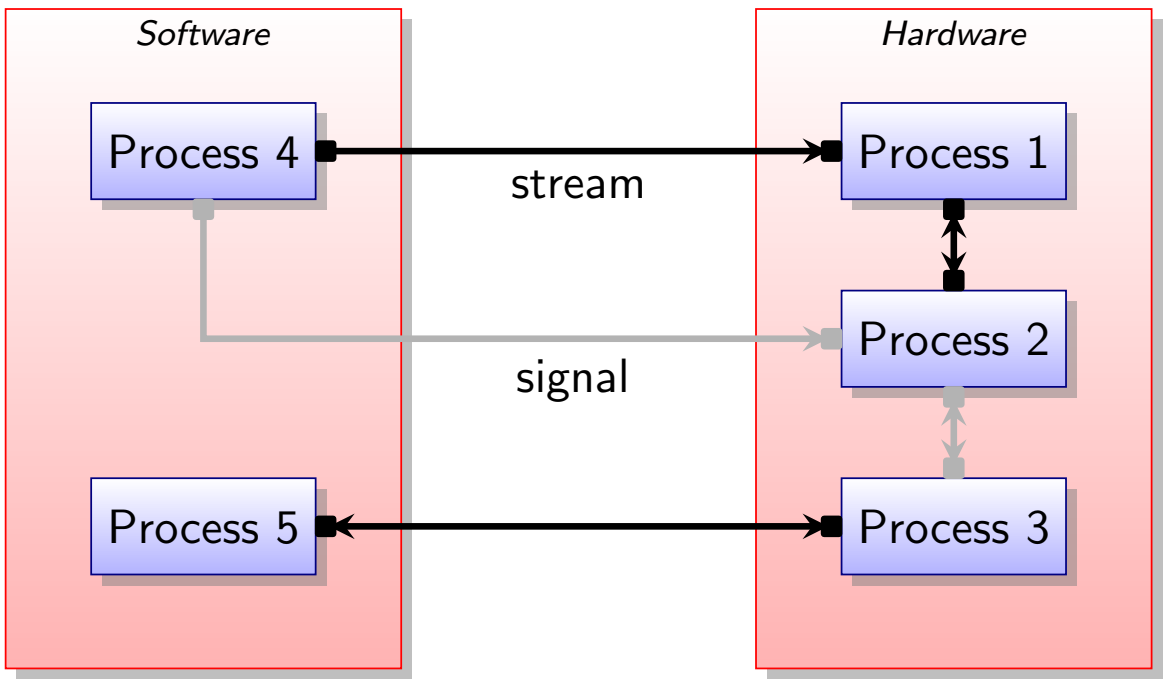
GET Lab

- Impulse C is subset of C programming language
- Allows to write C programs for usage on FPGA
- C is translated into VHDL
- **Advantages:**
  - No need to learn HDL
  - Rapid prototyping possible
  - Simulation on host PC possible
  - Allows hardware/software codesign
- **Disadvantages:**
  - Generated VHDL not as efficient as hand-optimized VHDL
  - Debugging hard in complex applications

Basics: Impulse C programming model

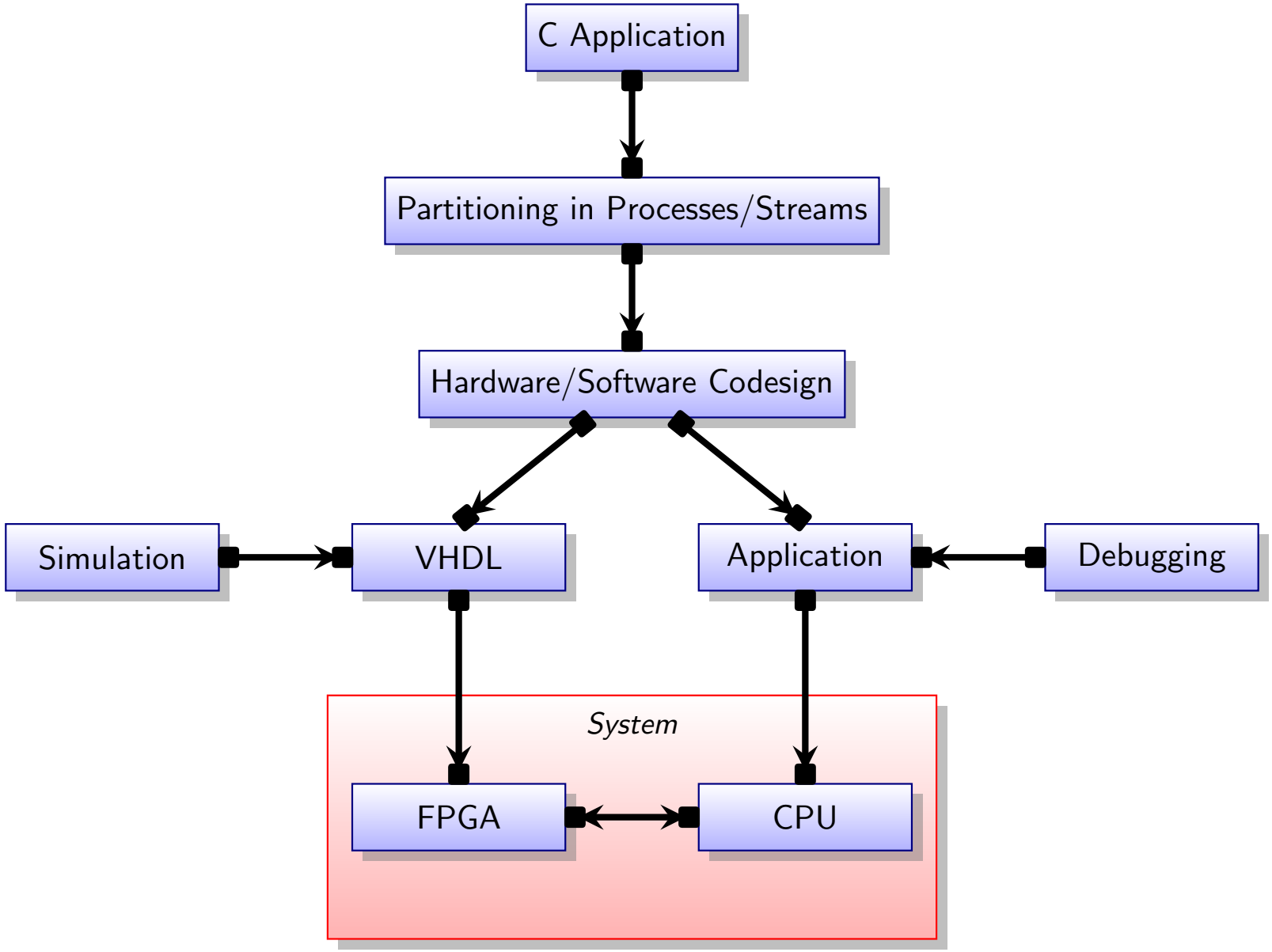
GET Lab

- C program is divided in different *processes*
  - All processes run concurrently
- Processes are interconnected with *communication objects*
  - e. g. streams, signals, semaphores
- Processes can run on processor as well as FPGA
- Different optimizations possible
  - e. g. pipelining, loop unrolling



Basics: Impulse C Workflow

GET Lab



Basics: Scan-Line based Segmentation

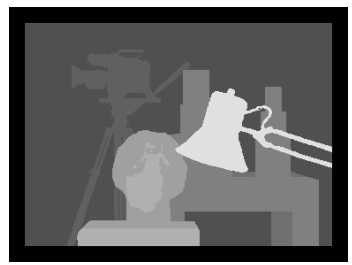
GET Lab

Idea

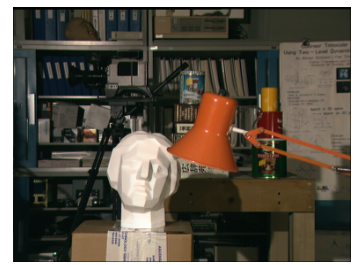
- Stereo vision algorithm, computes disparity map



left image



disparity map

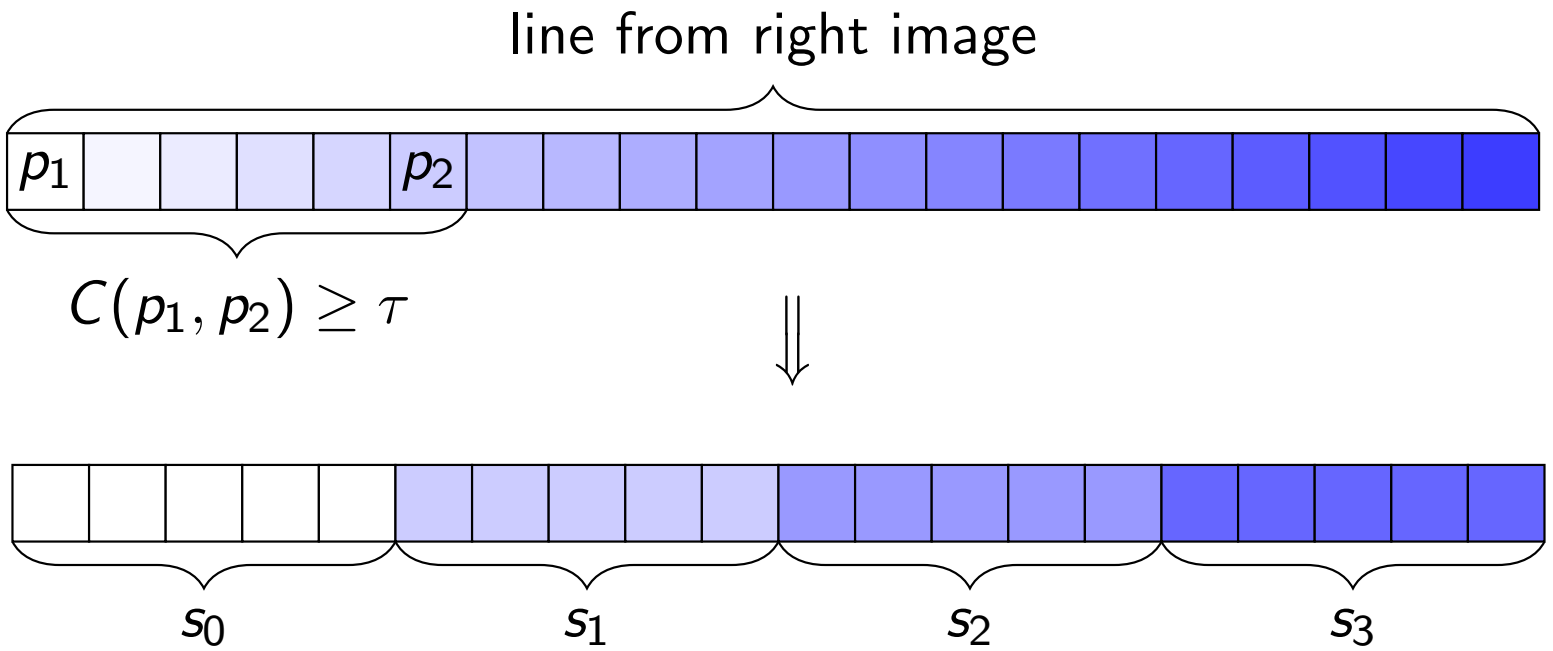


right image

- Local, line by line approach (no $n \times n$ filter window)
- Two phases:
  - Segmentation phase
  - Correspondence phase

GET Lab

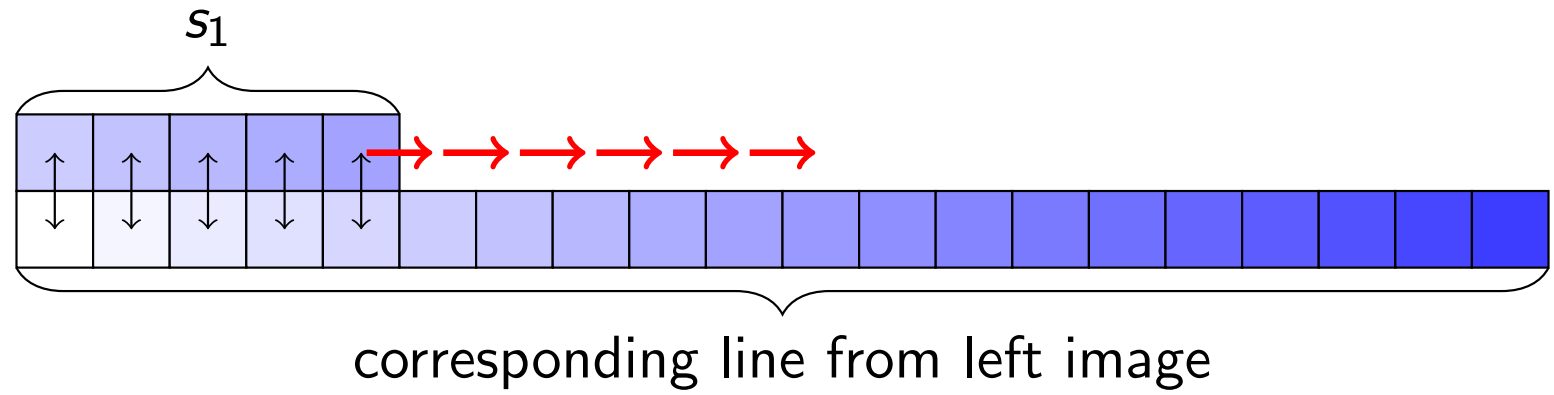
Phase I: Line segmentation

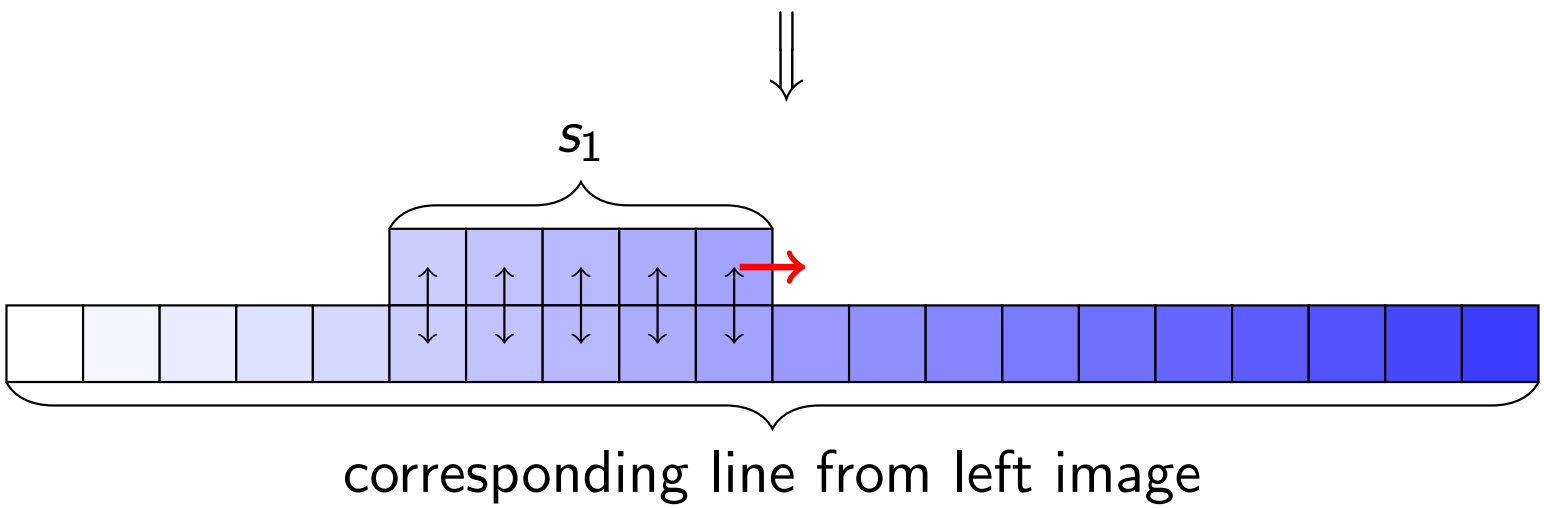


Basics: Scan-Line based Segmentation

GET Lab

Phase II: Correspondence



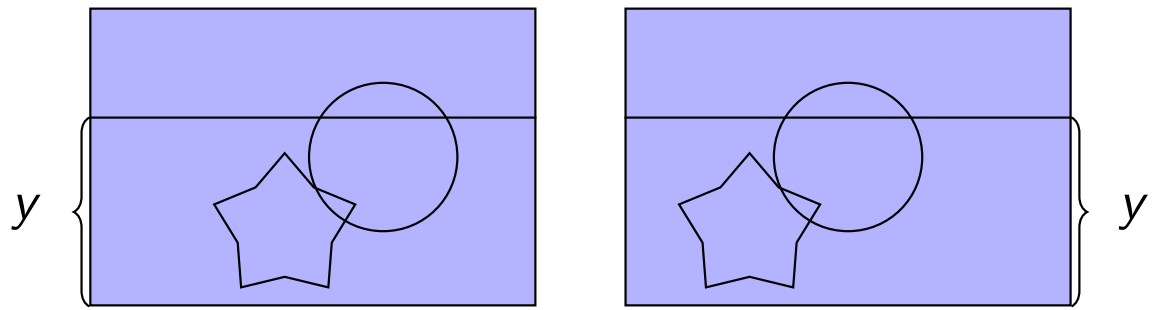


Basics: Scan-Line based Segmentation

GET Lab

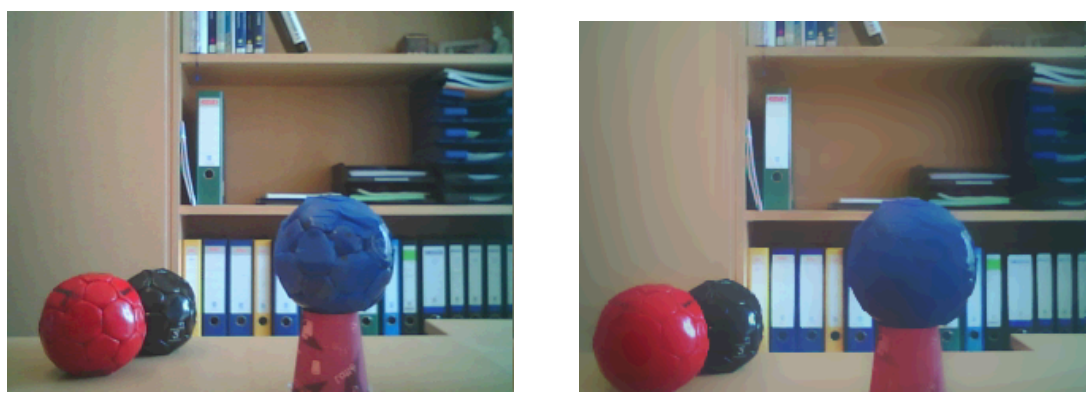
Requirements

- Rectified input images



left image right image

- Scene smooth except at borders of objects



detailed image smooth image

Agenda

GET Lab

Motivation

Basics

System Design
- Requirements
- Designs
- Design Decision
- HDL/Impulse C Partitioning

Stereo Vision System

Results

Summary and Outlook

System Design: Requirements

GET Lab

Requirements

- Image input, processing and output
  - Image Input: Simultaneously captured images
  - Algorithms: Full image pixel access in arbitrarily order
  - High Speed output interface
- Easy usage for C developers
- As much as possible implemented in Impulse C
  - Vision Algorithms
- Cost efficient system
- Long time availability of components
- Low power (robotic)

Tasks

- Choose hardware platform
- System partition in HDL and Impulse C

System Design: Design I

GET Lab

FPGA system with PowerPC processor and USB cameras

- **Processor:** Hard core PowerPC
- **OS:** Typically a Linux system
- **Advantages:**
  - Easy implementation of framegrabbing, data transmission
- **Disadvantages:**
  - FPGA + PowerPC is deprecated
  - Sparse number of development boards
  - Generally quite expensive
- **Partition:**
  - Image Capturing: Linux System
  - In/Output: Linux System
  - Vision Algorithm: Impulse C

FPGA system with MicroBlaze processor and USB cameras

- **Processor:** Soft core MicroBlaze
- **OS:** Typically a Linux system
- **Advantages:**
  - Easy implementation of framegrabbing, data transmission
- **Disadvantages:**
  - Requires logic elements for processor
  - Applicable boards not yet available
- **Partition:**
  - Image Capturing: Linux System
  - In/Output: Linux System
  - Vision Algorithm: Impulse C

System Design: Design III

GET Lab

FPGA system without processor and direct camera interface

- **Processor:** No processor
- **OS:** No operating system
- **Advantages:**
  - Everything on FPGA
  - High logic element efficiency compared to Design I+II
    - $\Rightarrow$ Smaller FPGA sufficient, cheaper
- **Disadvantages:**
  - Camera interfacing without driver support
  - Hard problem, potentially not done yet
  - In/Output in HDL
- **Partition:**
  - Image Capturing: HDL System
  - In/Output: HDL System
  - Vision Algorithm: Impulse C

GET Lab

Summary

	I: FPGA + PPC	II: FPGA + MB	III: FPGA
Availability	○○	○●	●●
Price	○○	○●	●●
Implementation	●●	●●	○○

Decision

- $\Rightarrow$ **Design III**

System Design: HDL/Impulse C Partitioning

GET Lab

Facts

- Memory necessary to allow arbitrarily image pixel access
- Fast output interface necessary
  - Gigabit Ethernet preferred
- **Impulse C**: No direct interfacing possible for
  - Cameras
  - Ethernet
  - Memory

Partition

- Image algorithms in Impulse C
- VHDL: Cameras, memory and Ethernet interface
- **Data Path:** Input(VHDL) $\rightarrow$ Impulse C $\rightarrow$ Output(VHDL)

Agenda

GET Lab

Motivation

Basics

System Design

Stereo Vision System
- System Hardware
- System Architecture
- Vision Framework
- Vision Core

Results

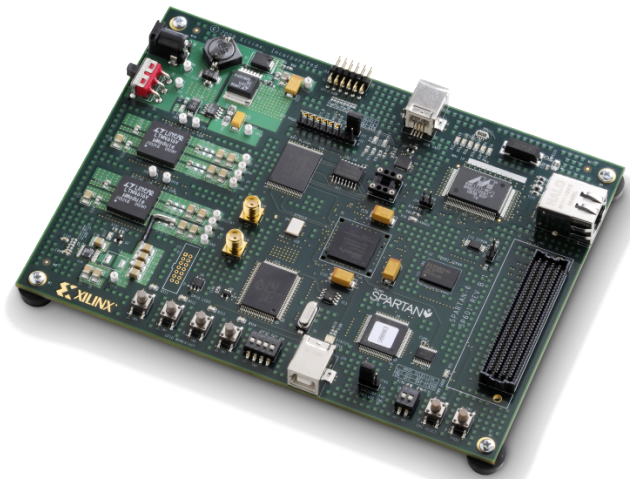
Summary and Outlook

Stereo Vision System: Hardware

GET Lab

SP601 Development Board

- Xilinx Spartan-6 FPGA
  - Integrated Memory Controller Blocks
- 128 MB DDR2 memory
- Gigabit Ethernet transceiver
- Available, complies with requirements
- Costs $\approx$ 300 Euro



How can the development board be used?

- Cameras can't be directly connected
- $\Rightarrow$ Own hardware components are necessary

Considerations

- **Cameras:**
  - With direct interface
  - Replaceable camera lenses
- **Mounting:**
  - Maintain camera orientation
  - Adaptable camera distances
- **Additional Hardware:**
  - Used for data transmission between cameras and SP601
  - Preserve data integrity
  - Perform voltage translation

Stereo Vision System: Hardware

GET Lab

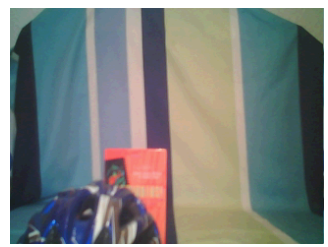
Cameras

- Direct interface to image sensor
- Configured by approx. 200 registers

OV7 Camera

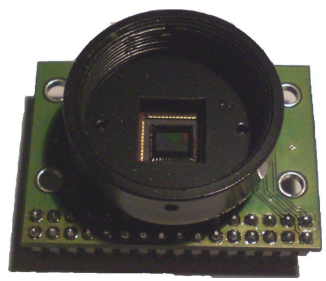
- Evaluation board of Omnivision
- OmniVision OV7720 sensor
- 640 $\times$ 480 pixel resolution, max. 60 fps





MT9V Camera

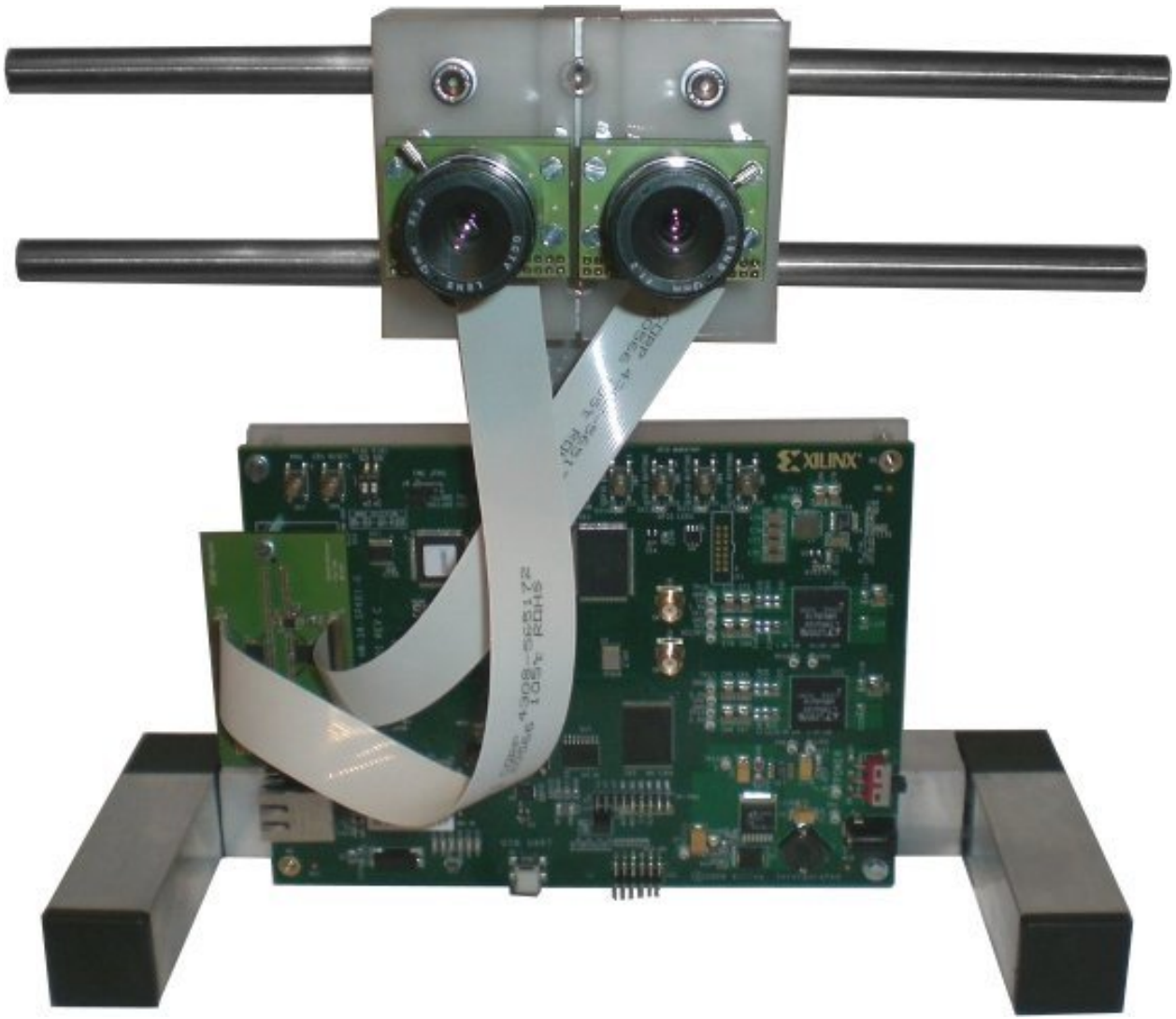
- Self designed and manufactured
- Aptina MT9V032 sensor
- 752 $\times$ 480 pixel resolution, max. 60 fps

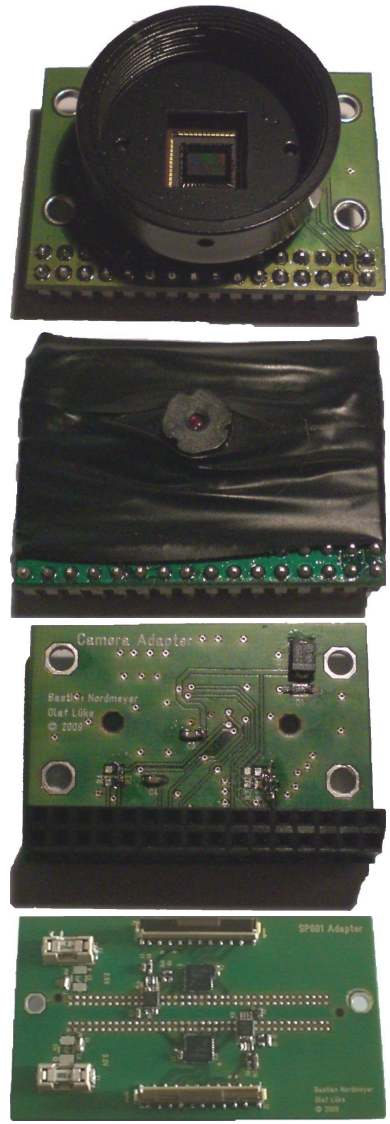




Stereo Vision System: Hardware

GET Lab

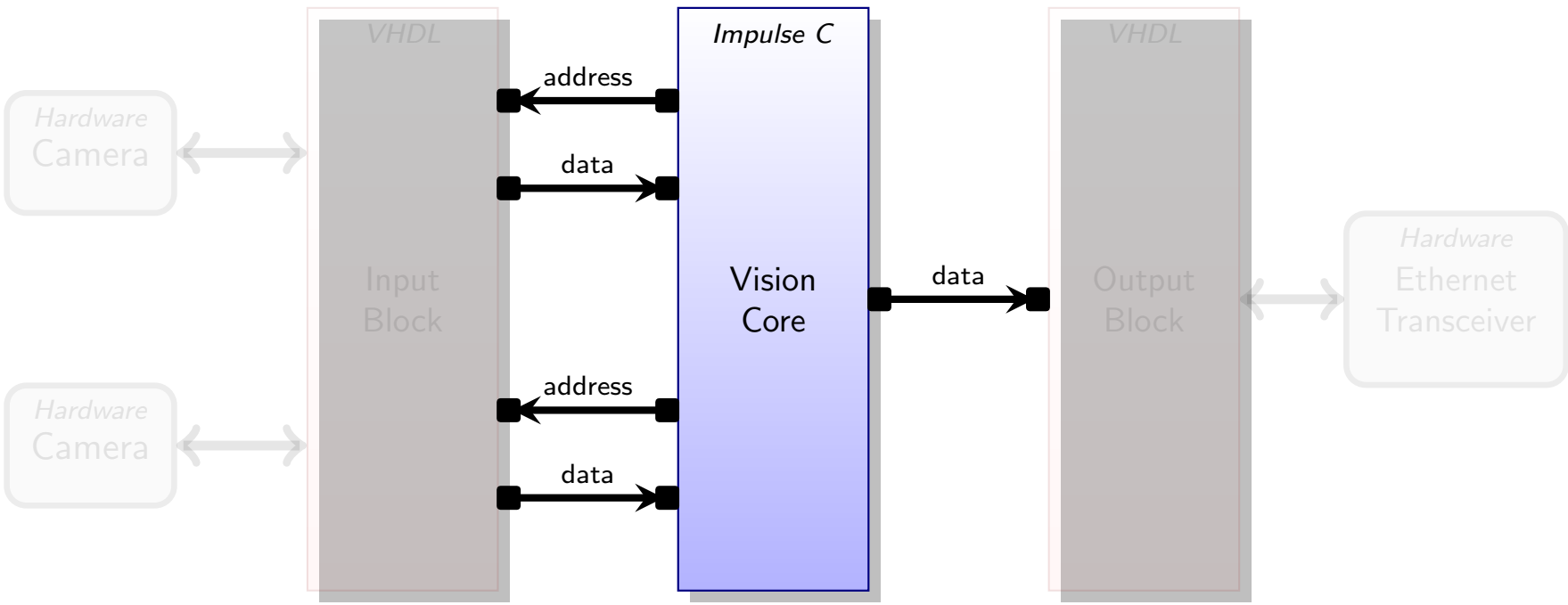




Stereo Vision System: System Architecture

GET Lab

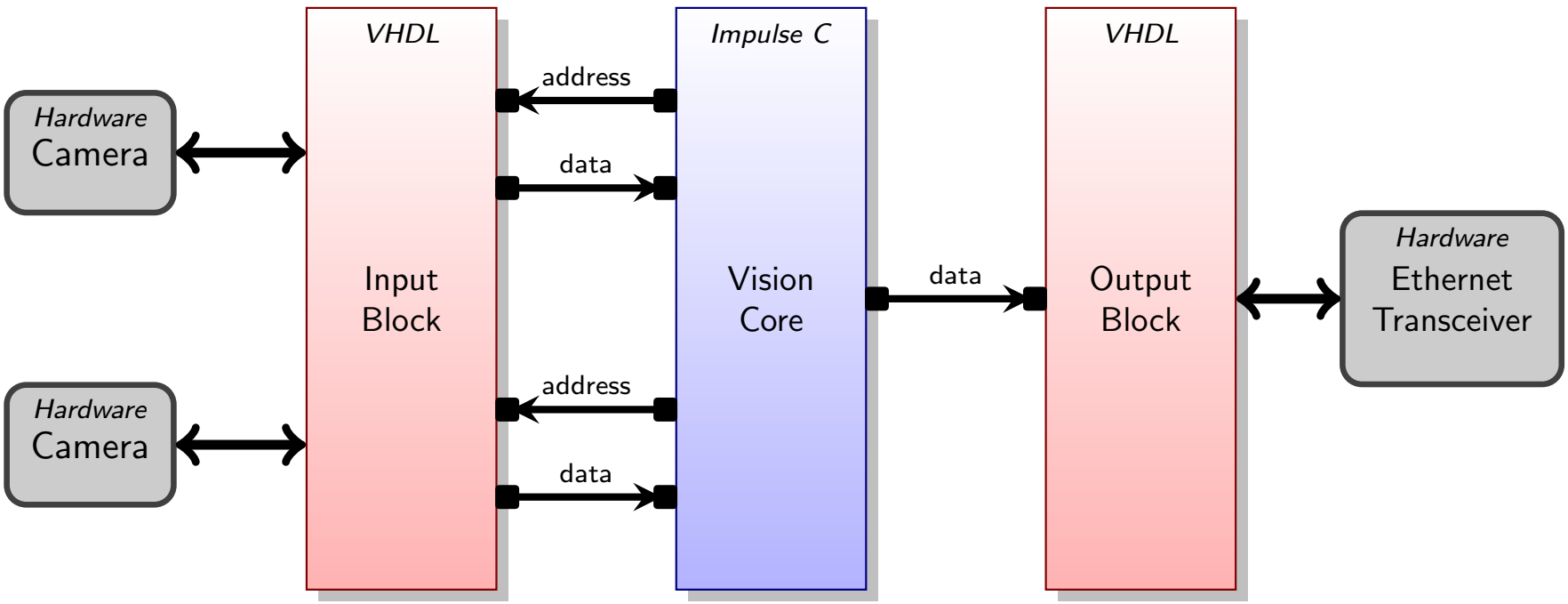
General Architecture



Stereo Vision System: System Architecture

GET Lab

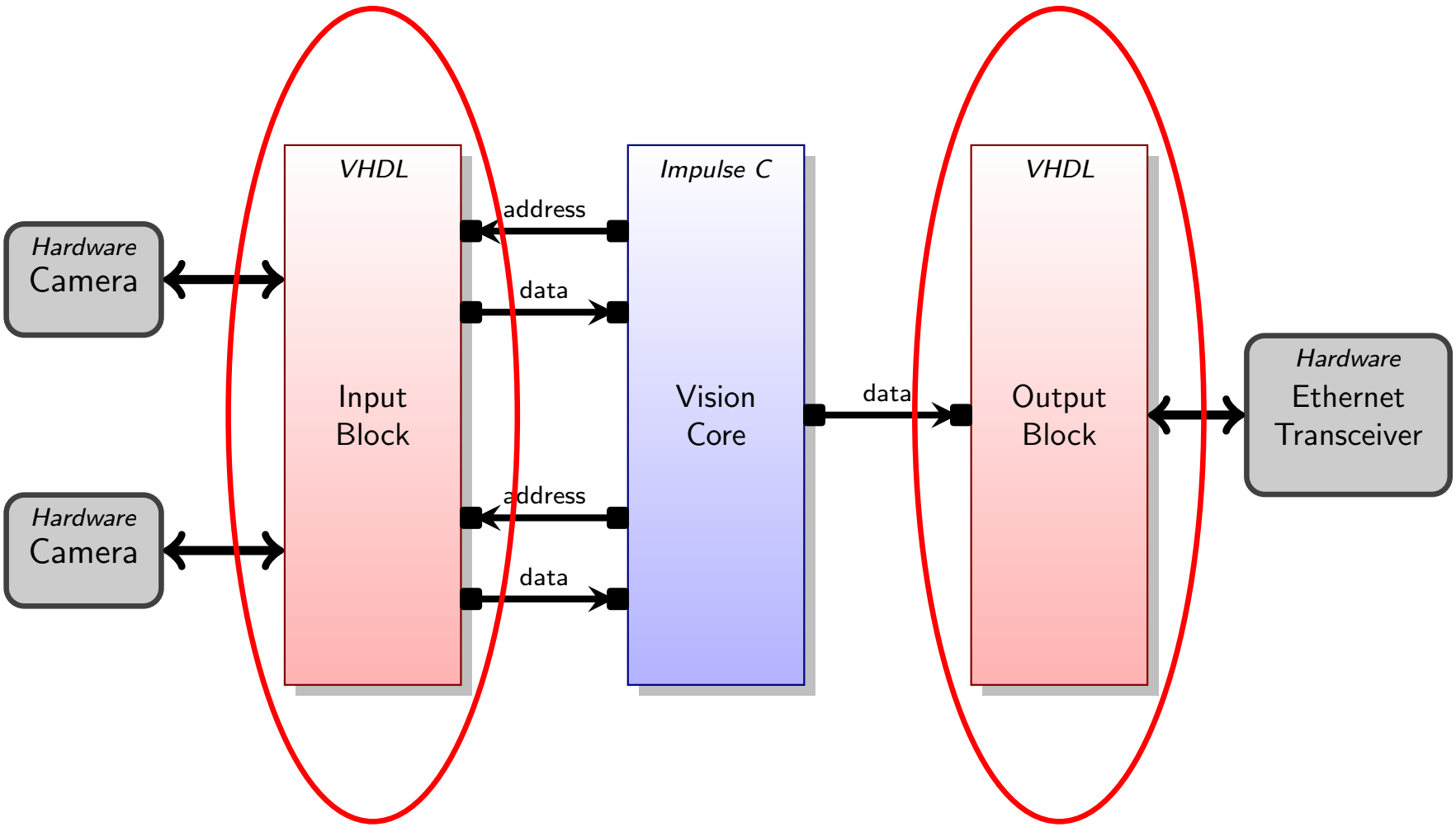
General Architecture



Stereo Vision System: System Architecture

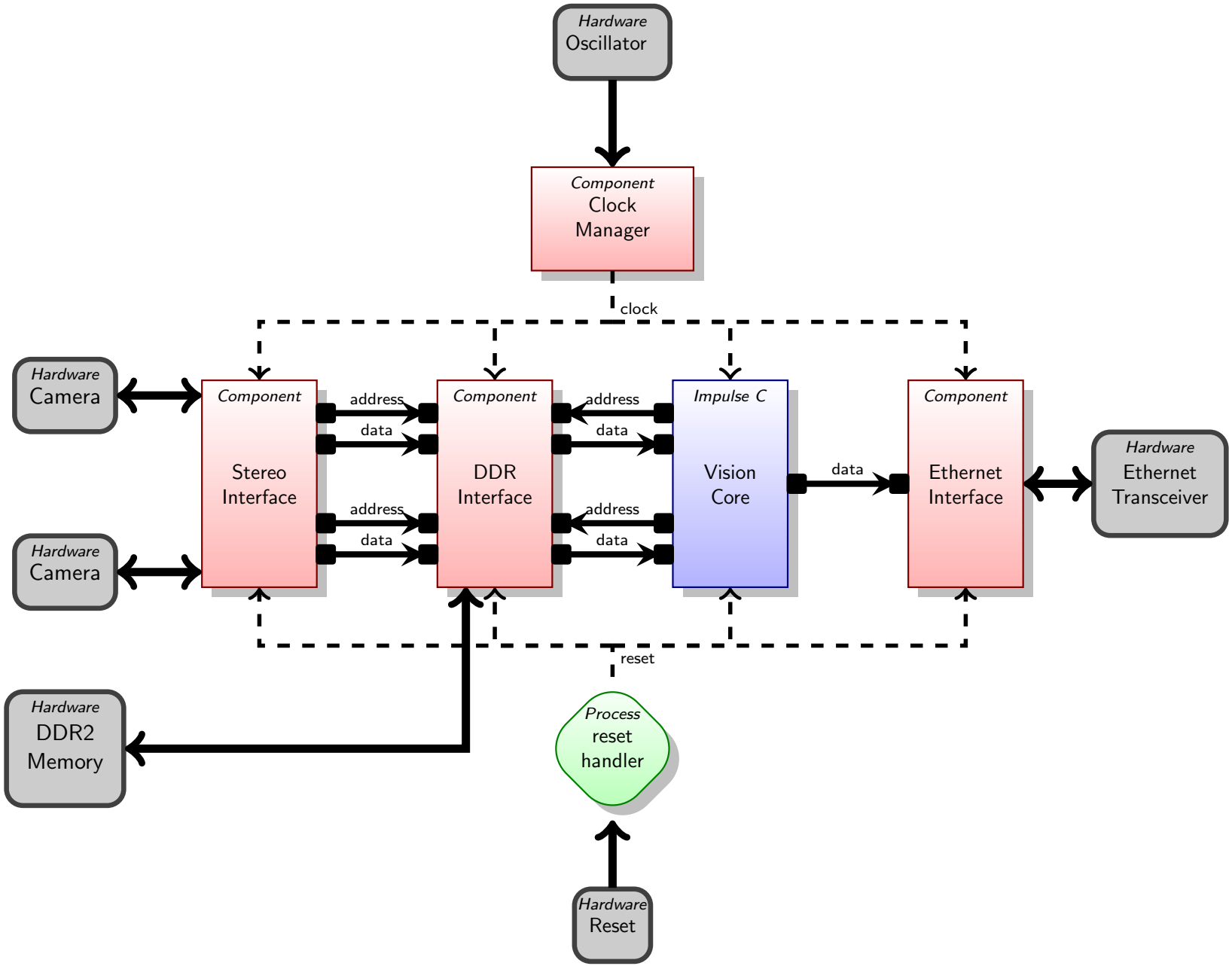
GET Lab

General Architecture

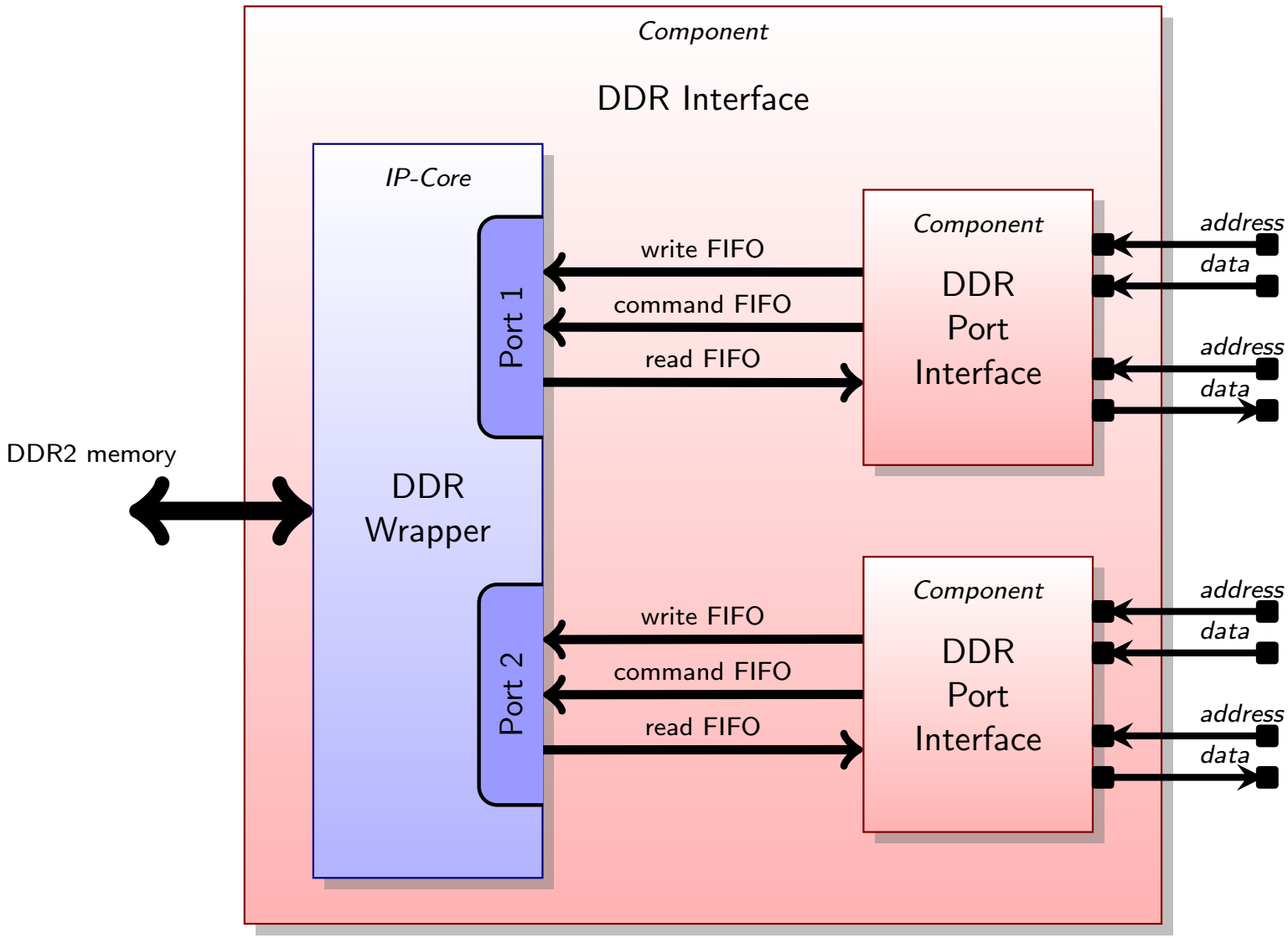


Stereo Vision System: Framework Overview

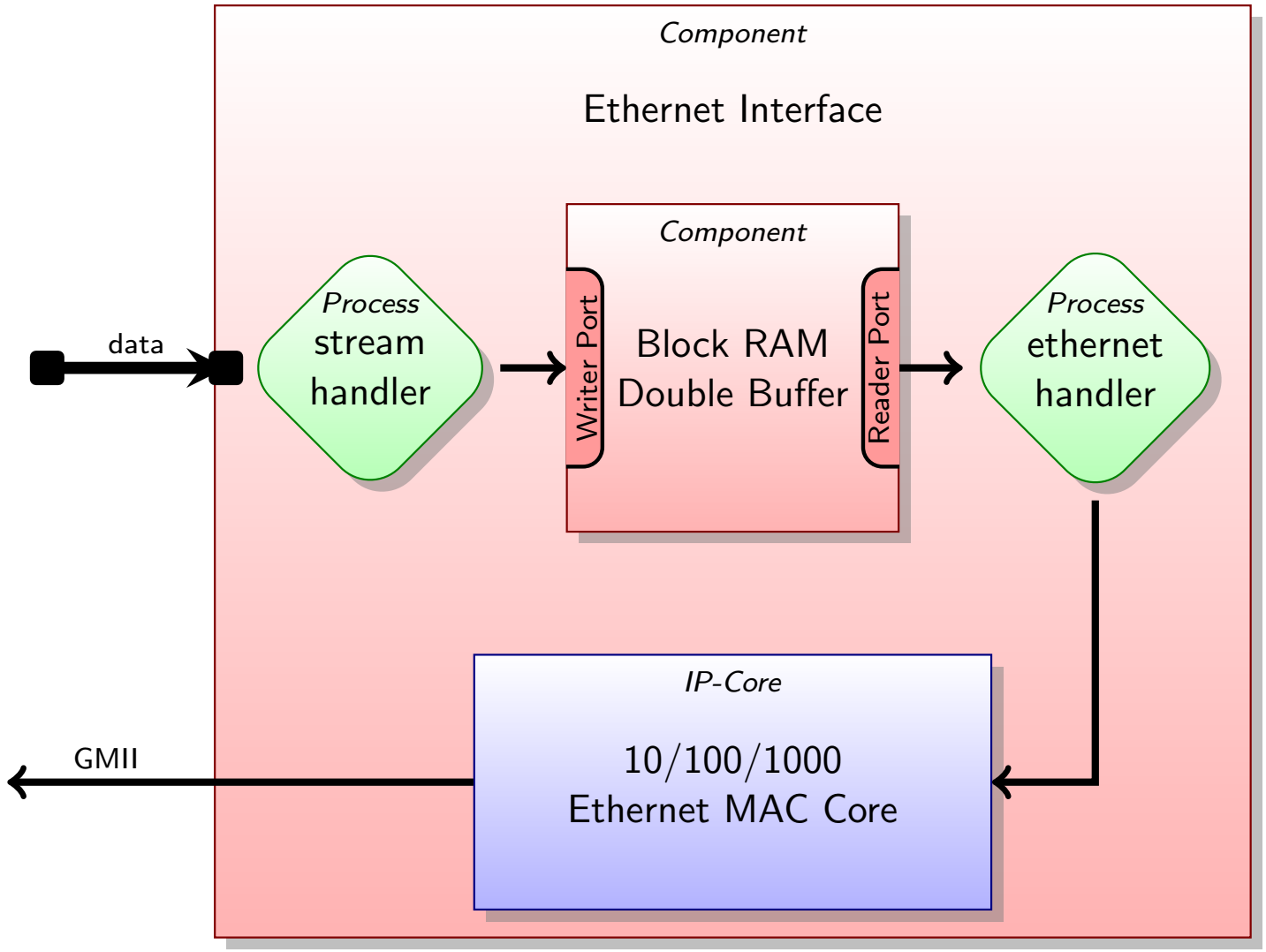
GET Lab



Stereo Vision System: DDR Interface

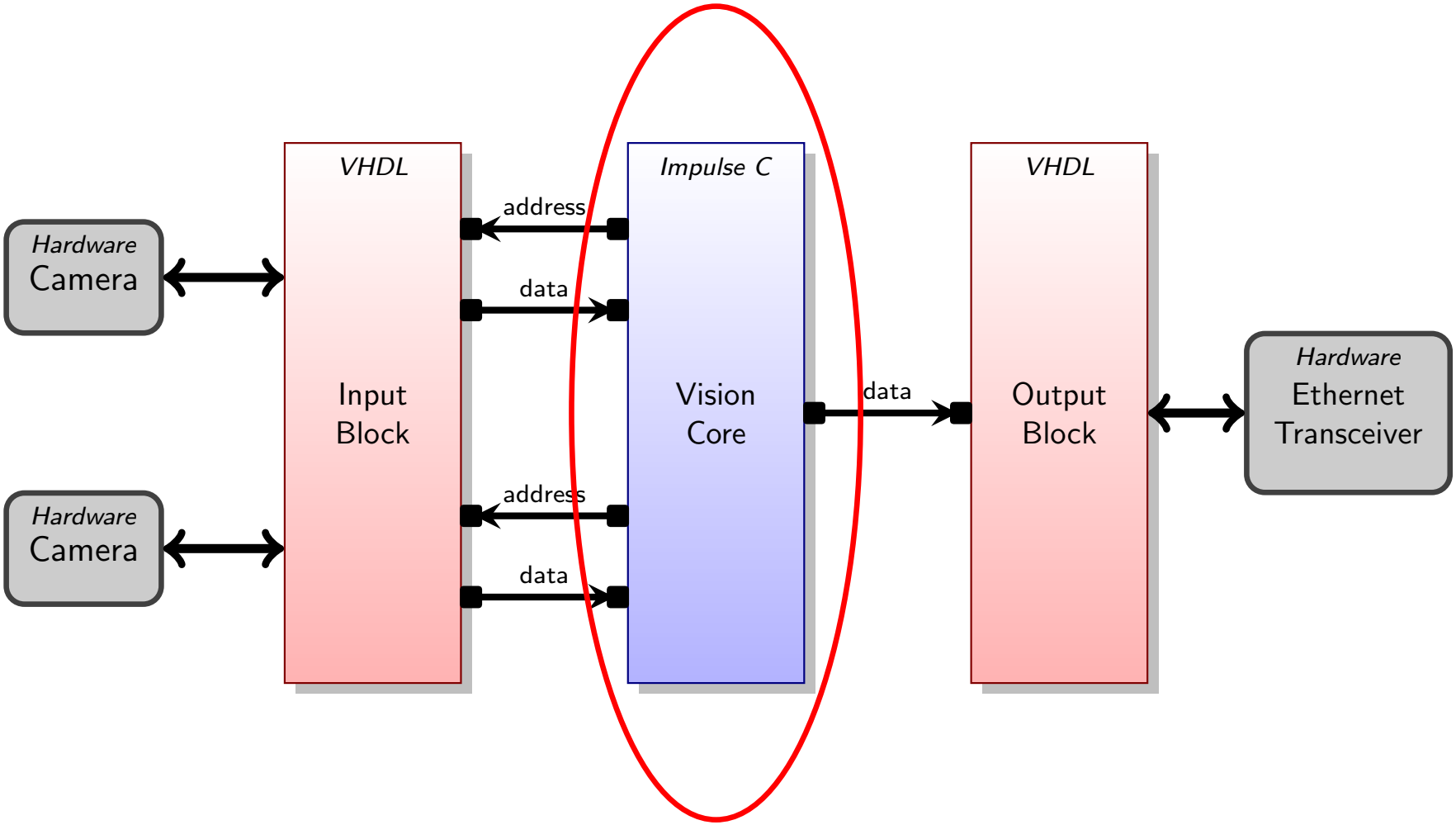


Stereo Vision System: Ethernet Interface



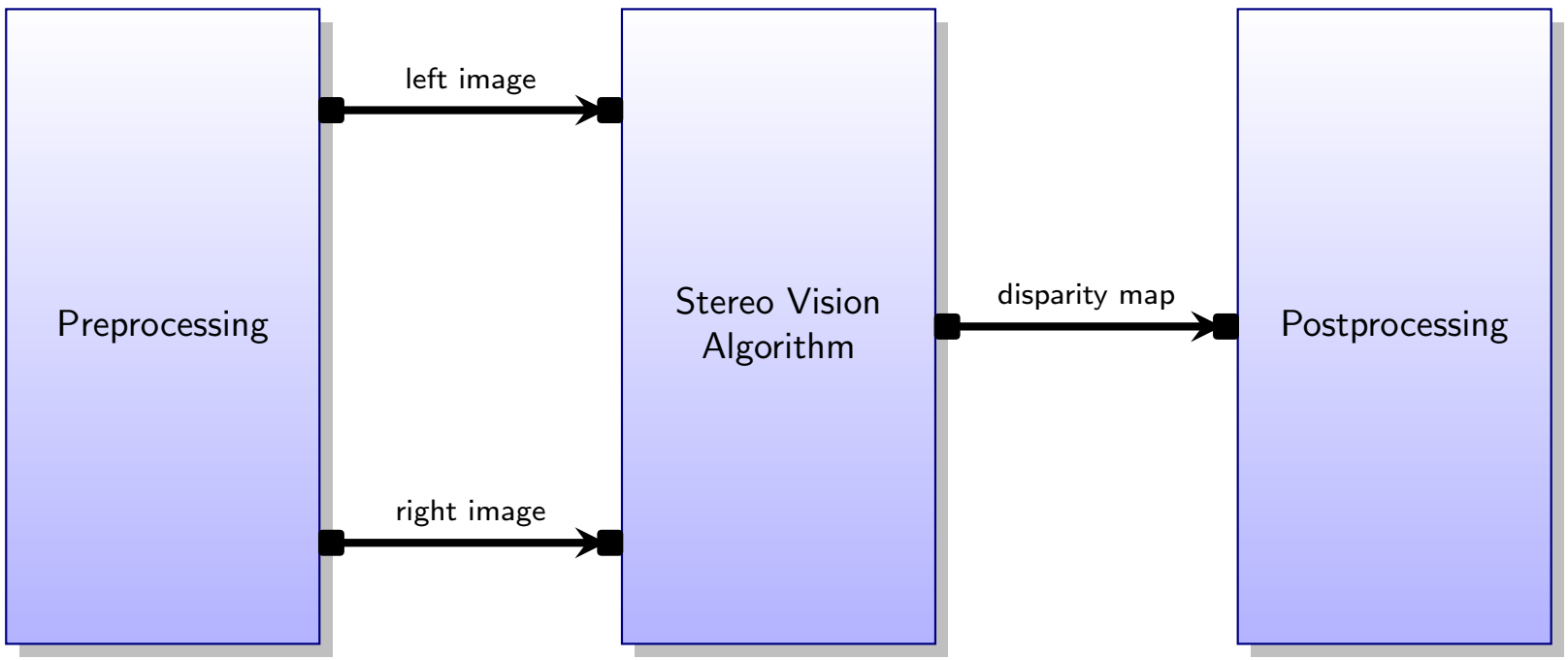
Vision Core: Design

GET Lab



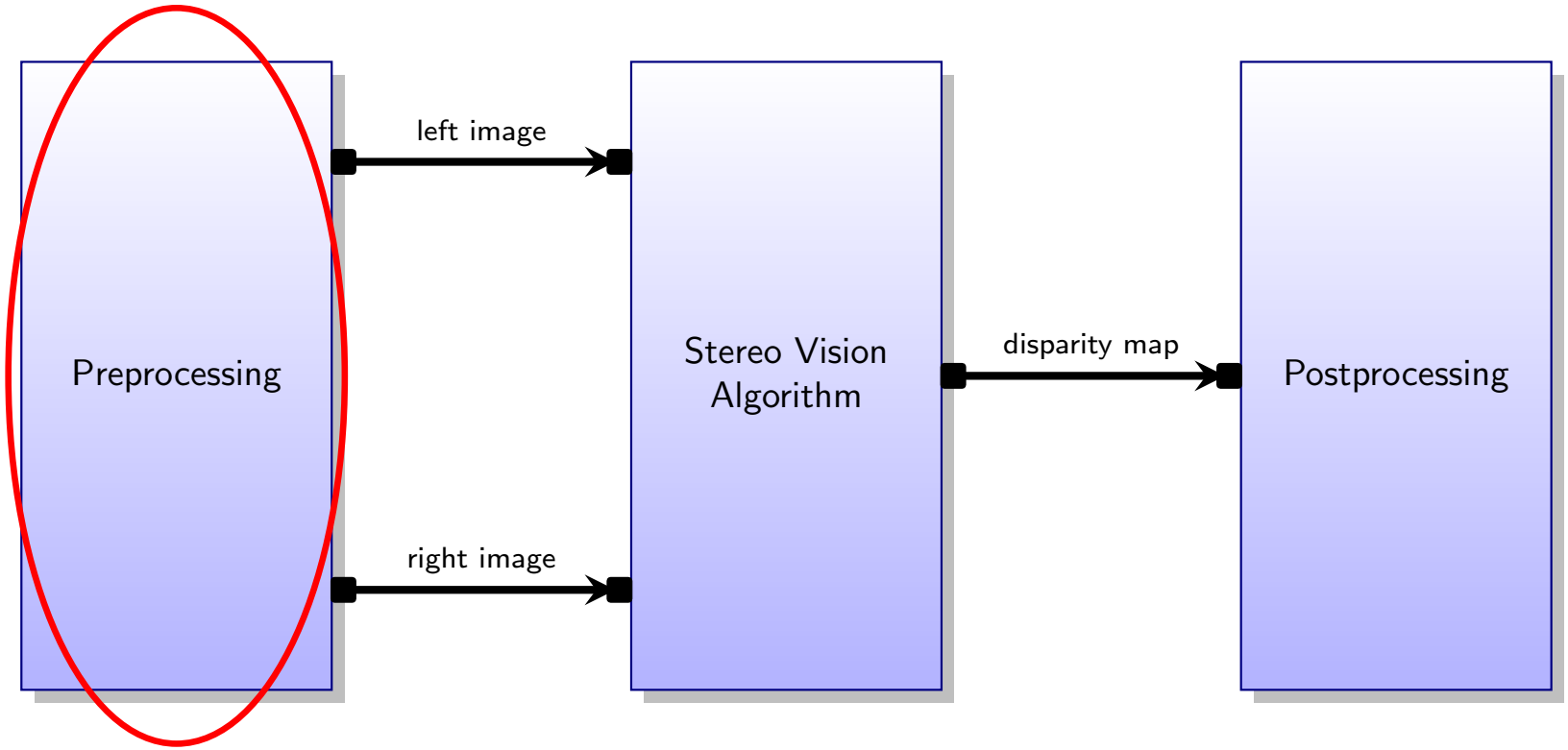
Vision Core: Design

GET Lab



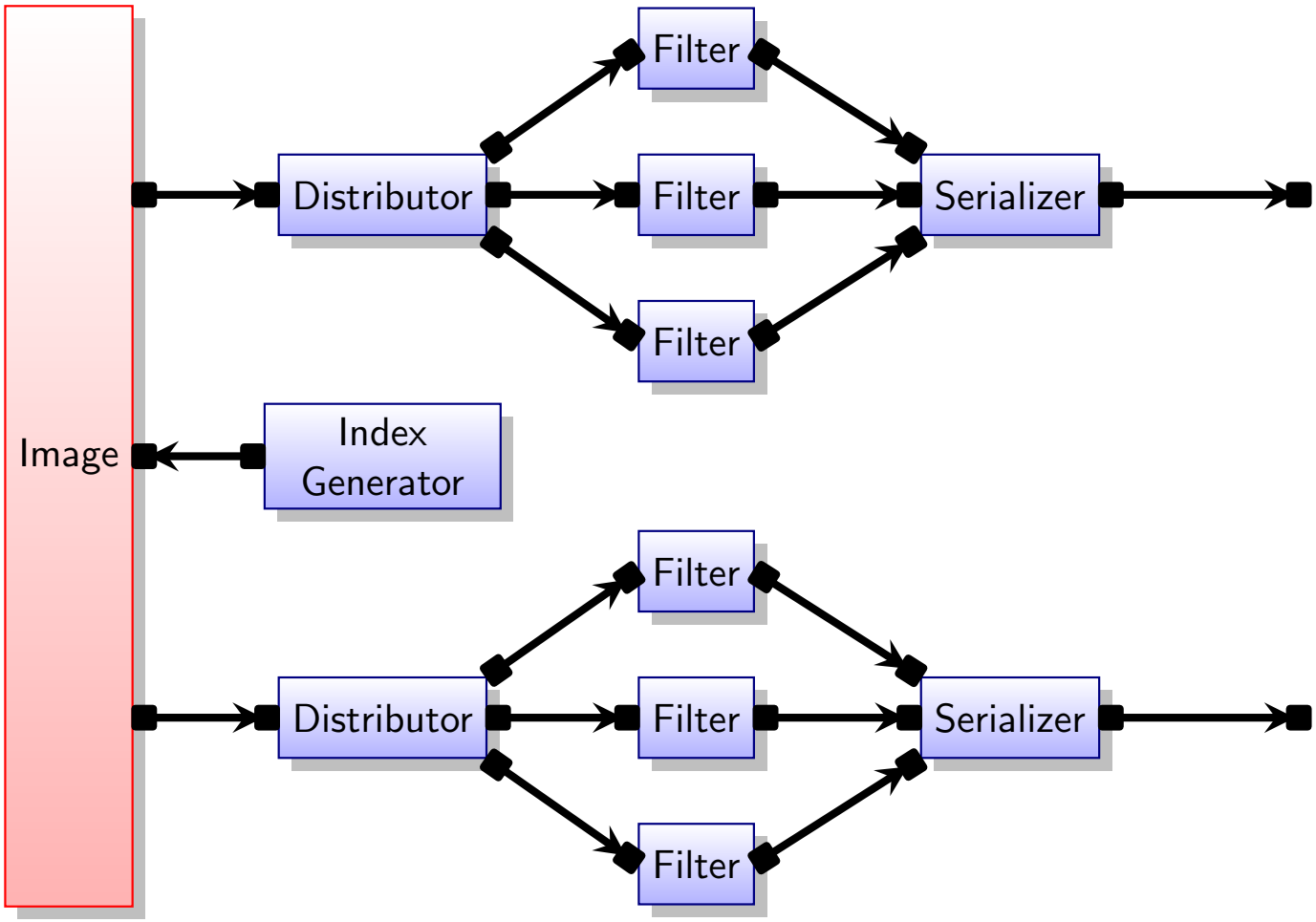
Vision Core: Design

GET Lab



GET Lab

Architecture



Vision Core: Preprocessing

GET Lab

- Four different image filter implemented in **C** and tested:
  - Gaussian filter
  - Median filter
  - Hybrid median filter
  - Bilateral filter
- Bilateral filter implemented in **Impulse C**:
  - Has most parameterization possibilities
  - Can act as Gaussian and median filter
  - Implementable in $\mathcal{O}(1)$ for arbitrary kernel window sizes[1]

[1] Ben Weiss. „Fast median and bilateral filtering“. In: ACM Trans. Graph. 25.3 (2006), pp. 519–526

GET Lab

Illustration

- Three example filter windows with corresponding convolution
- Edge enhancing and smoothing in one run recognizable



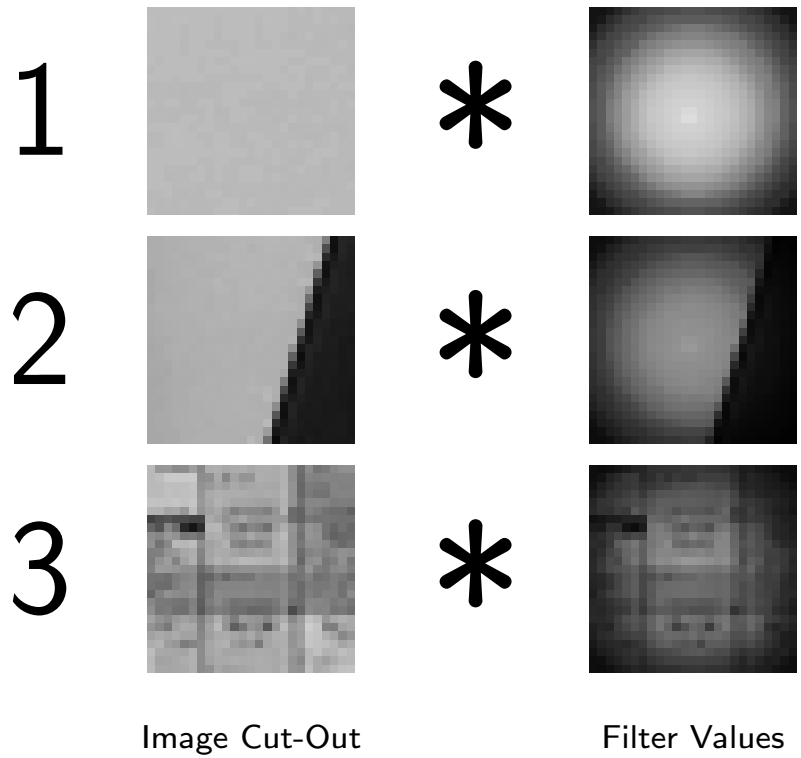


Image Cut-Out Filter Values

Vision Core: Bilateral filter

GET Lab

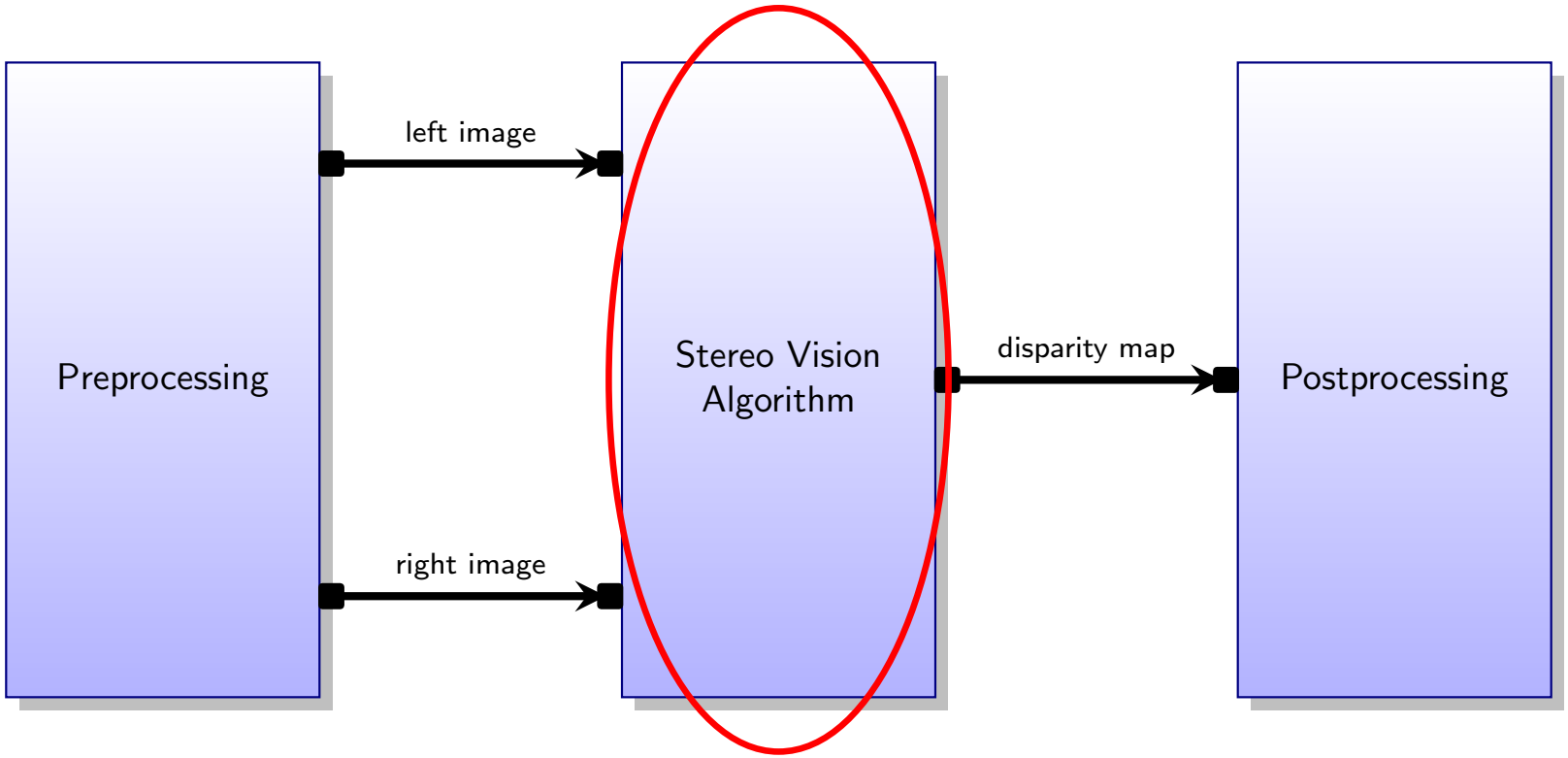
Illustration





Vision Core: Scan-Line based Segmentation

GET Lab



Vision Core: Scan-Line based Segmentation

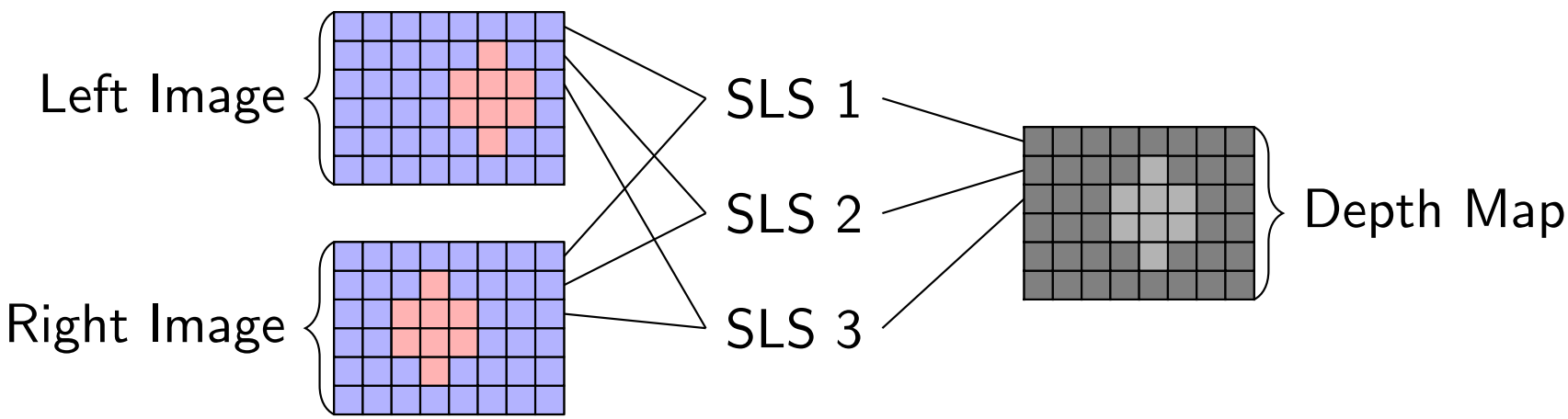
GET Lab

Implementation

- Straight forward
- Buffer one line from left and right image respectively
- Parallelization as in preprocessing problematic
- Other parallelization possibilities?

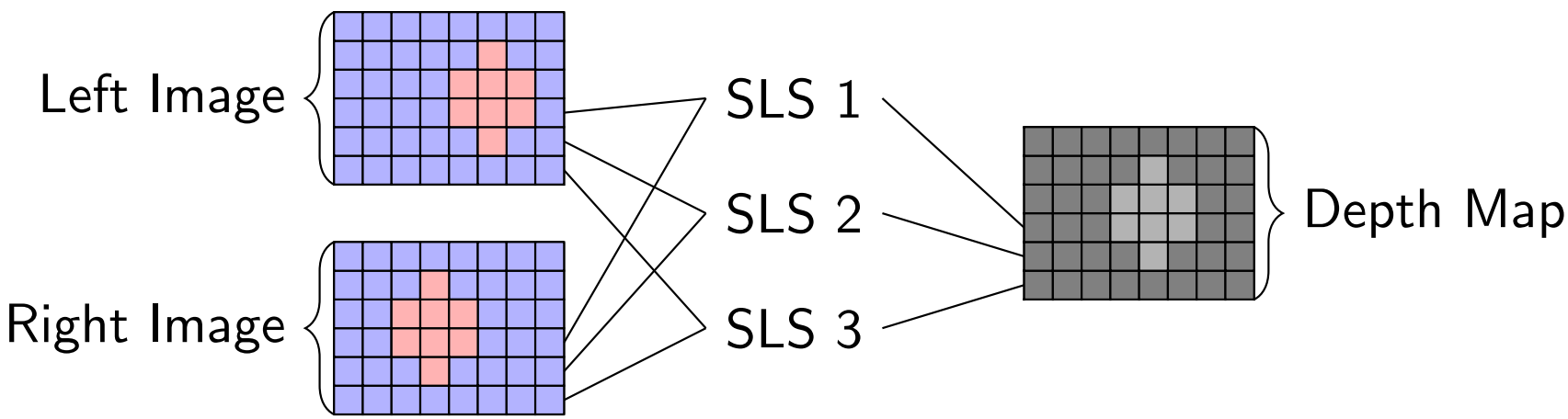
Implementation

- Straight forward
- Buffer one line from left and right image respectively
- Parallelization as in preprocessing problematic
- Other parallelization possibilities?



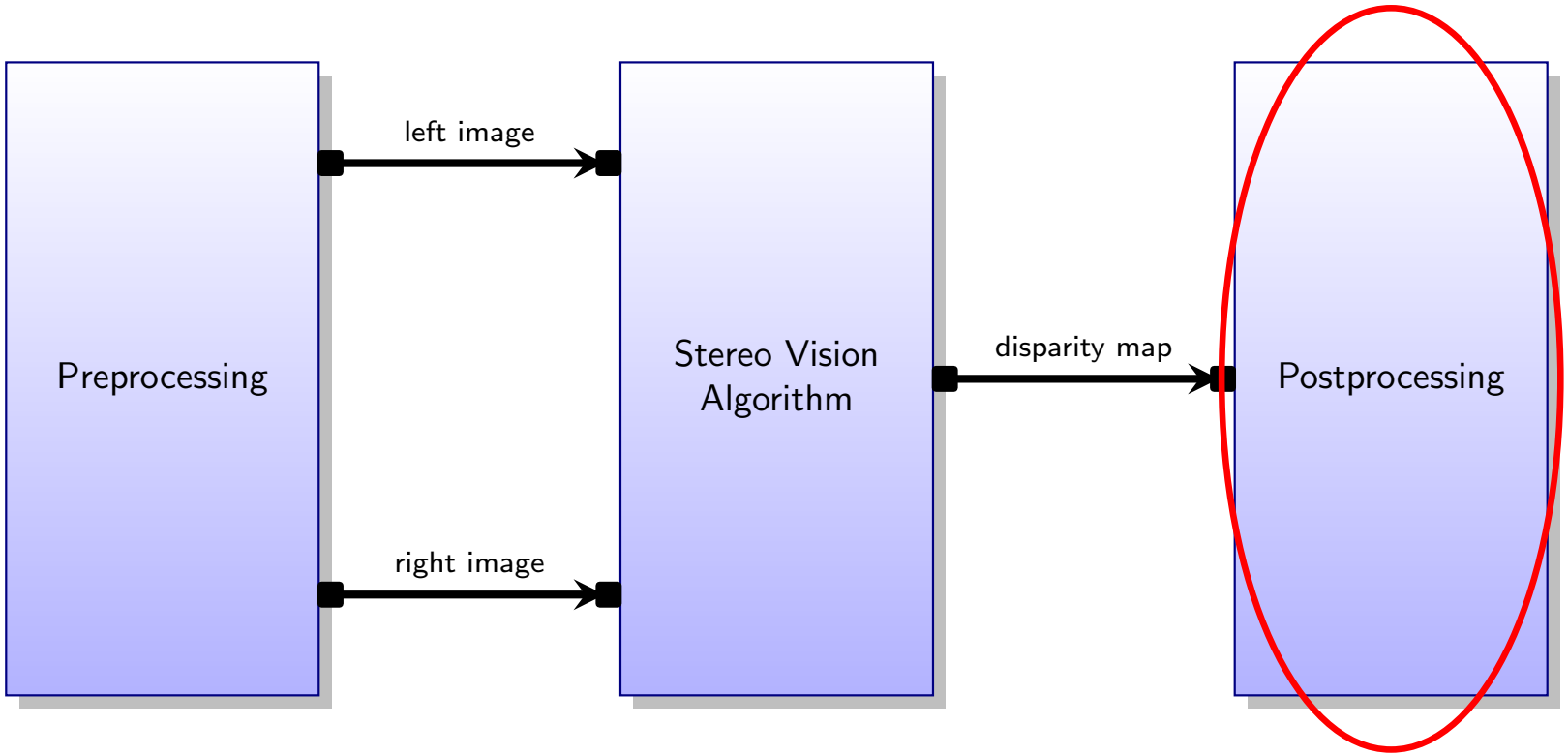
Implementation

- Straight forward
- Buffer one line from left and right image respectively
- Parallelization as in preprocessing problematic
- Other parallelization possibilities?



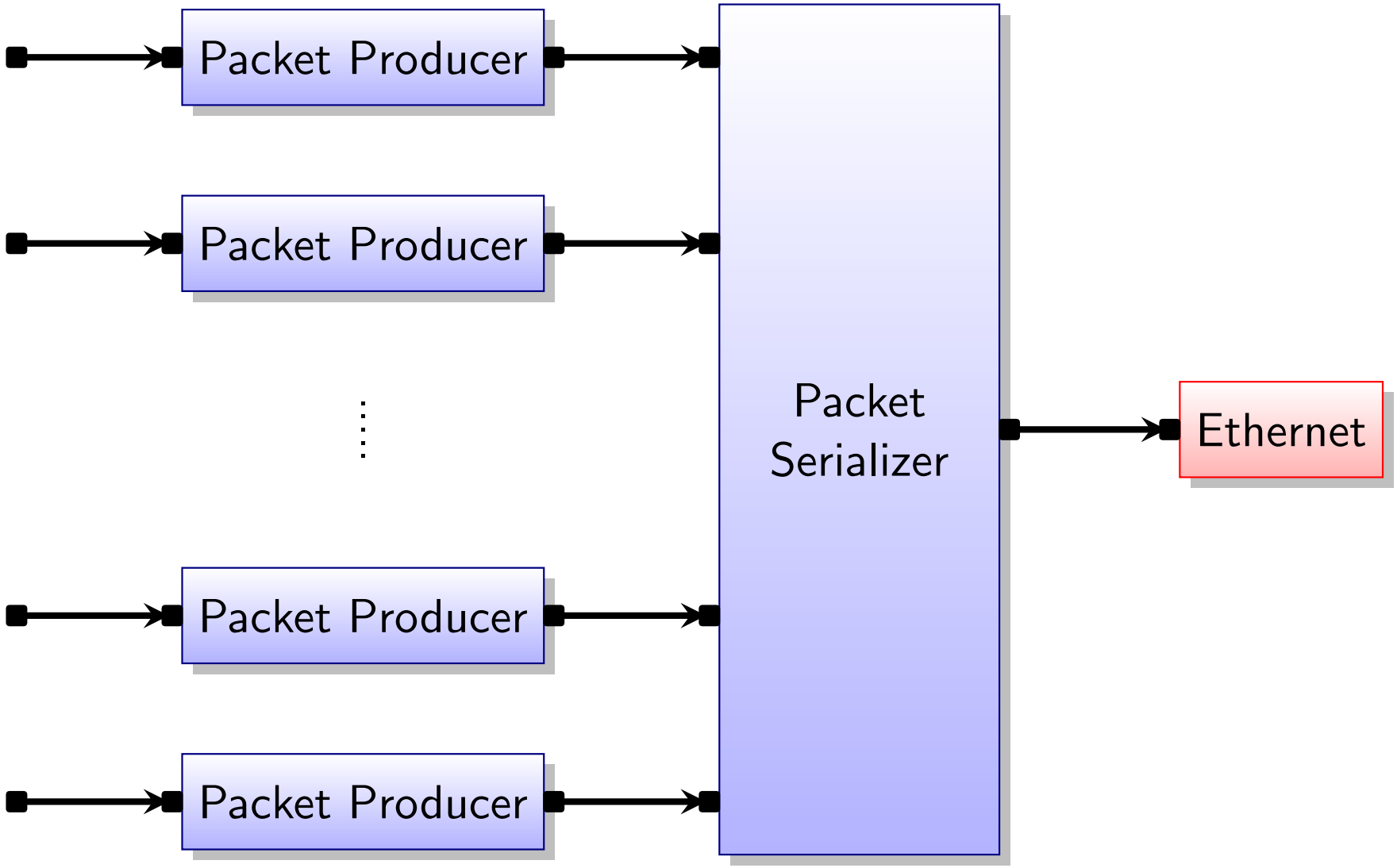
Vision Core: Scan-Line based Segmentation

GET Lab



Vision Core: Postprocessing

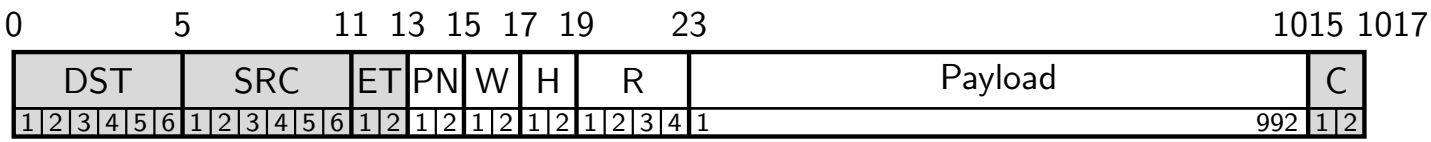
GET Lab



GET Lab

Packet Producer

- Packets must allow host PC to distinguish between images
  - Packet type (ET)
  - Packet number (PN)
  - Width (W) and height (H)
- Raw Ethernet frames (IEEE 802.3) are used



Packet Serializer

- Buffers packets for VHDL Ethernet entity

Agenda

GET Lab

Motivation

Basics

System Design

Stereo Vision System

Results

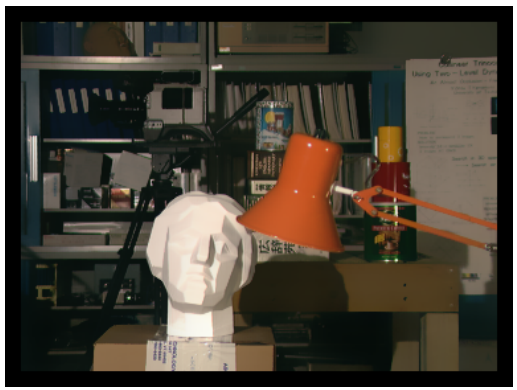
Summary and Outlook

Results: Bilateral Filter

GET Lab



Unfiltered



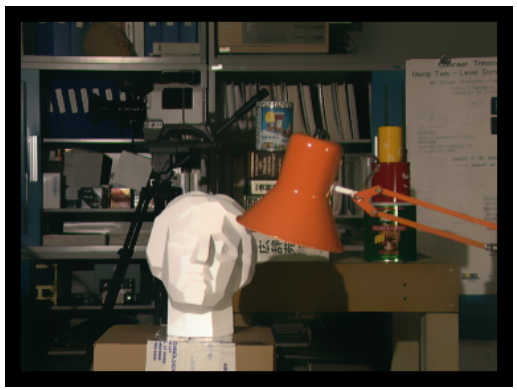
$\sigma = 0.05$



$\sigma = 0.15$

Results: Bilateral Filter

GET Lab





Unfiltered | $\sigma = 0.05$ | $\sigma = 0.15$

Results: Scan-Line based Segmentation

GET Lab



Left image



Right image



Bilateral filtered



Without bilateral filter



With bilateral filter

Results: Scan-Line based Segmentation

GET Lab



Left image



Right image



Bilateral filtered



Without bilateral filter



With bilateral filter

Results: Interpretation

GET Lab

- Smoothing is important in the present of noise
- Rectification more important
- Good illumination necessary for good results

Results: Performance

GET Lab

Framework

	Frequency	Read transfer rate	Write transfer rate
Ethernet	10 MHz	–	154 Mbit/s
	25 MHz	–	386 Mbit/s
	50 MHz	–	772 Mbit/s
	100 MHz	–	954 Mbit/s
	200 MHz	–	954 Mbit/s
DDR	10 MHz	80 Mbit/s	53 Mbit/s
	25 MHz	200 Mbit/s	133 Mbit/s
	50 MHz	400 Mbit/s	267 Mbit/s
	100 MHz	457 Mbit/s	457 Mbit/s
	200 MHz	533 Mbit/s	533 Mbit/s

Results: Performance

GET Lab

Vision Core

Component	Optimization	FPS
Bilateral Filter	without	3.742
	with	5.403
Median Filter	without	10.69
	with	14.79
Scan-Line based Segmentation	without	2.23
	with	5.70
Parallel Scan-Line based Segmentation	without	5.16
	with	12.08

Agenda

GET Lab

Motivation

Basics

System Design

Stereo Vision System

Results

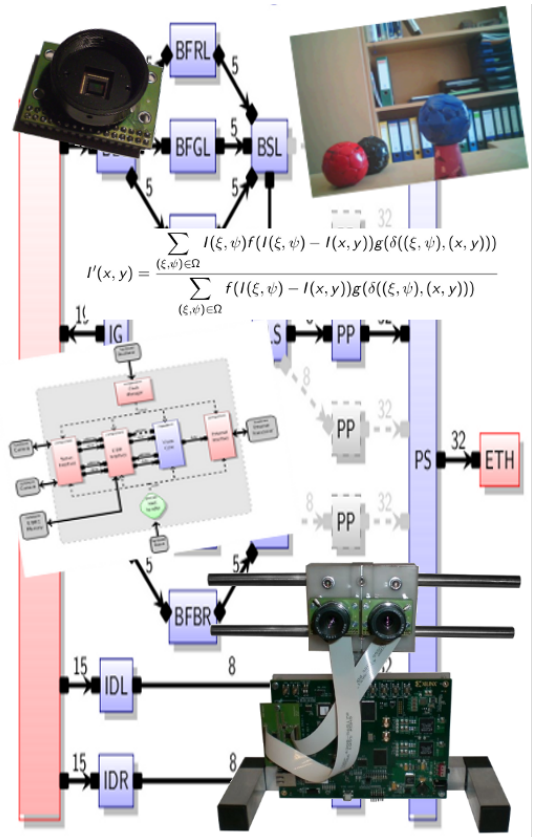
Summary and Outlook

Summary

GET Lab

Contributions:

- Achieved cost-efficient prototyping Stereo Vision System
- Performed requirement analysis
- Analyzed available hardware components
- Designed own hardware adapters and cameras
- Developed generic framework for C developers
- Implemented Vision Core in Impulse C
- Developed system model for simulation
- Evaluated and optimized power dissipation, performance and resource requirements



Outlook

Possible enhancements:

- More sophisticated pre and post-processing
  - Other solutions often use simple algorithms (SAD) with complex pre and post-processing
- System configurable at run-time

Questions?

Figure References

GET Lab

Slide Motivation 1

https://getwww.uni-paderborn.de/wiki/msr/images/2/2f/TSR.jpg
http:
//www-robotics.jpl.nasa.gov/projects/projectImage.cfm?Project=1&Image=335
http://astronomy.hccs.edu/astronomy/images/1152619main_panelsdeploy_med.jpg
http://www.eri.harvard.edu/faculty/peli/projects/enhanced.jpg

Slide Basics FPGA

http://samkerr.files.wordpress.com/2009/09/556px-fpga_xilinx_spartan.jpg

Slide Basics FPGA 2

http://www.xilinx.com/images/circuitblock.jpg

Slide Hardware: SP601

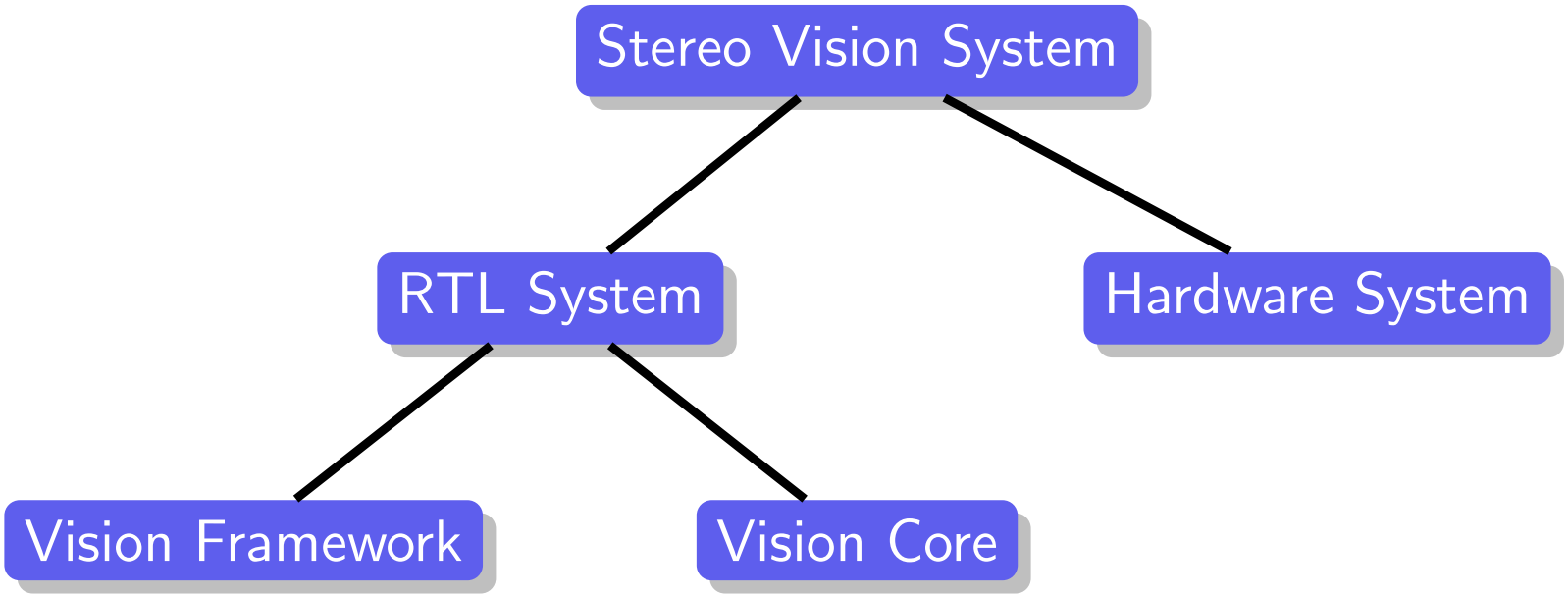
http://www.xilinx.com/products/devkits/EK-S6-SP601-G-image.htm

Stereo Example Pictures

http://cat.middlebury.edu/stereo/data.html

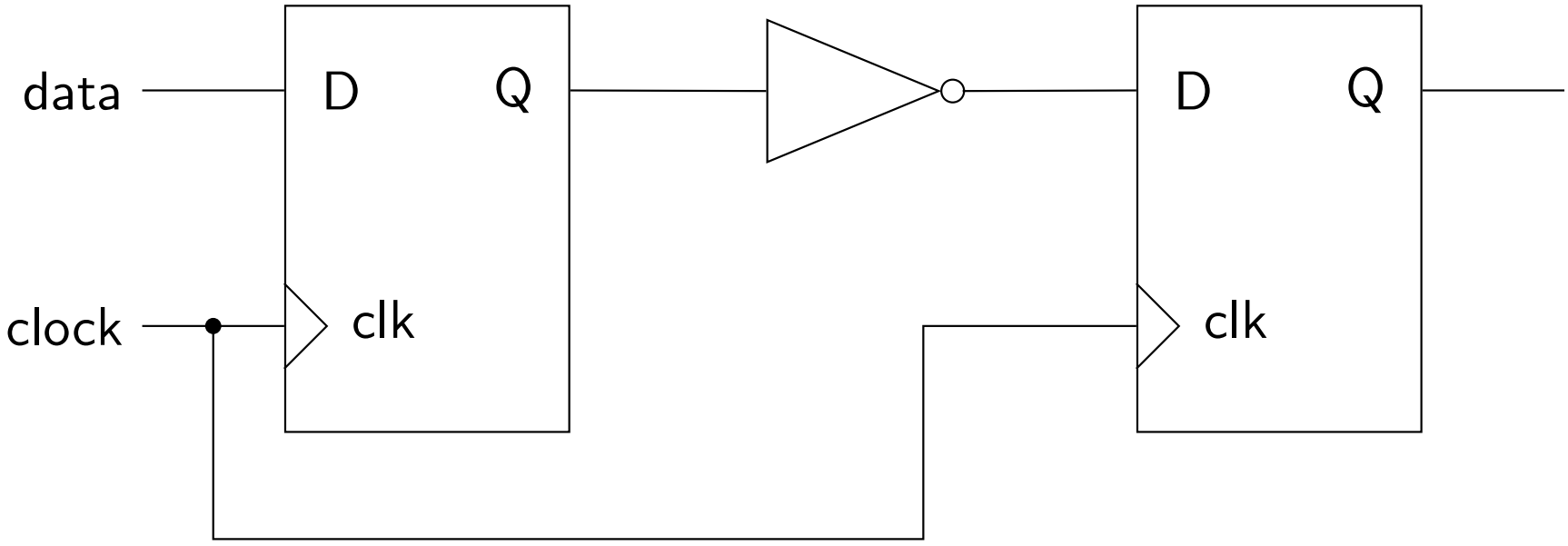
Appendix: Taxonomy

GET Lab



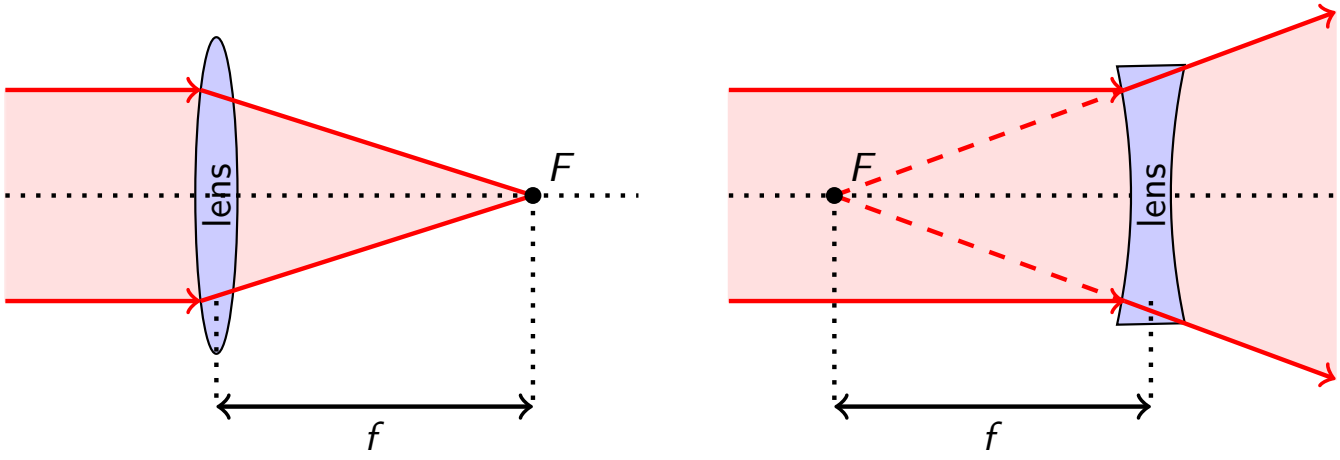
Appendix: Register Transfer Logic

GET Lab



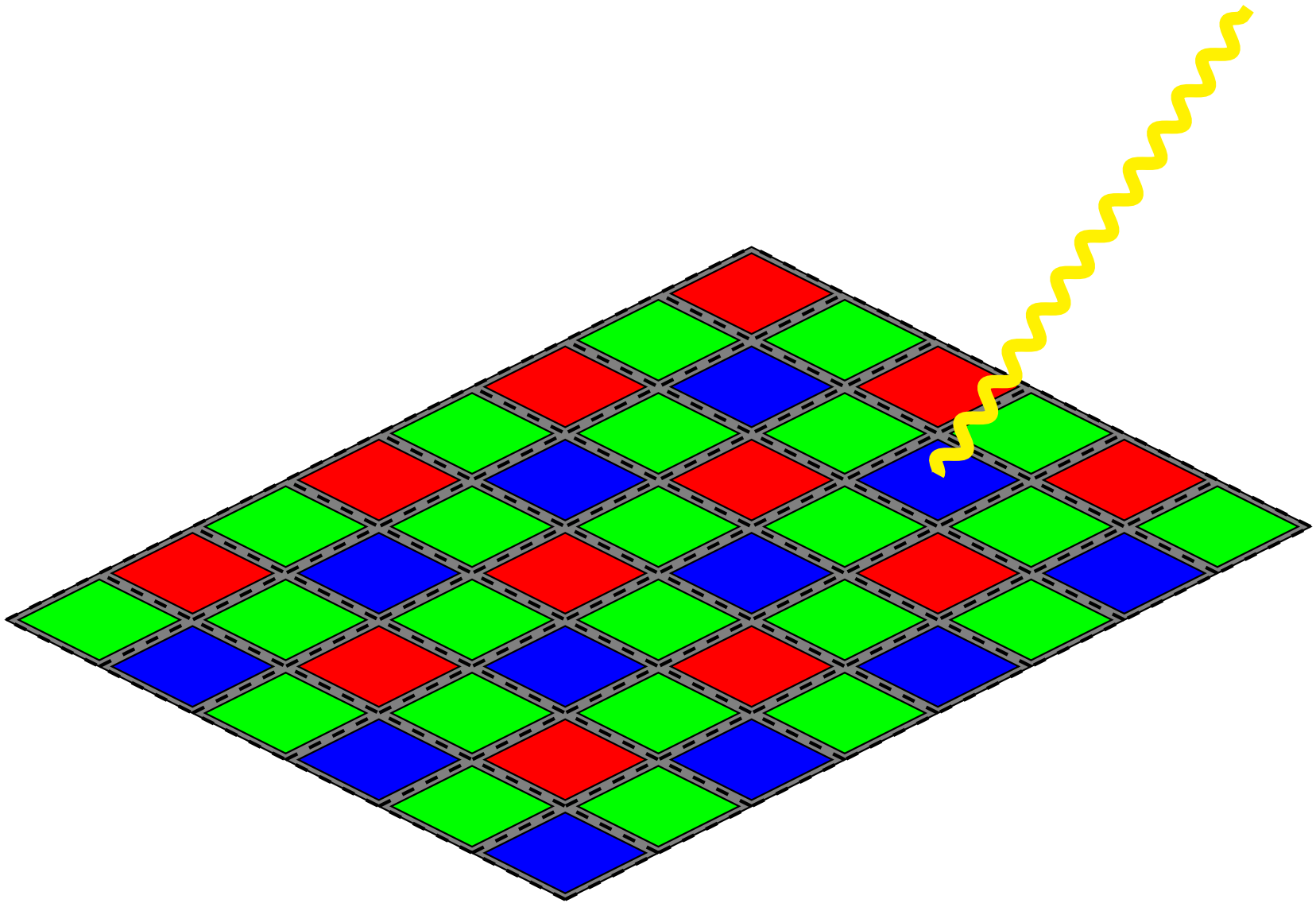
Appendix: Focal Length

GET Lab



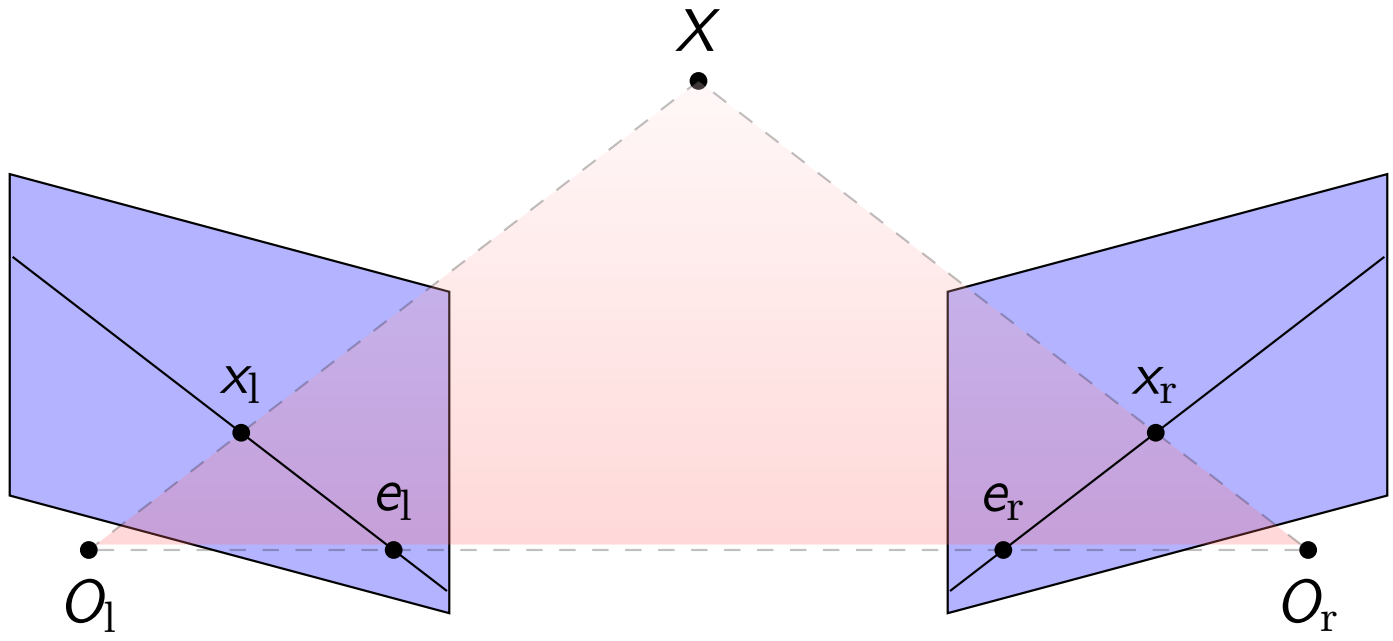
Appendix: Bayer Pattern

GET Lab



Appendix: Epipolar Geometry

GET Lab



Appendix: Formulae 1

GET Lab

Gaussian

$$G_\sigma(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}},$$

Bilateral Filter

$$I'(x,y) = \frac{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(\xi,\psi) f(I(\xi,\psi) - I(x,y)) g(\delta((\xi,\psi),(x,y)))\,\mathrm{d}\xi\,\mathrm{d}\psi}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(I(\xi,\psi) - I(x,y)) g(\delta((\xi,\psi),(x,y)))\,\mathrm{d}\xi\,\mathrm{d}\psi}$$

Geometric Distance

$$\delta((\xi,\psi),(x,y)) = \sqrt{(\xi-\psi)^2 + (x-y)^2}.$$

Appendix: Formulae 2

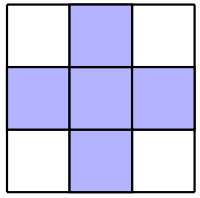
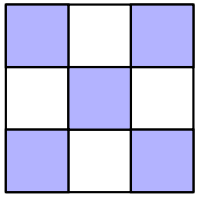
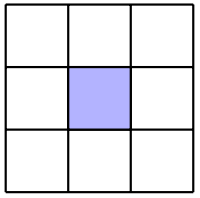
GET Lab

Bilateral Filter with Histogram

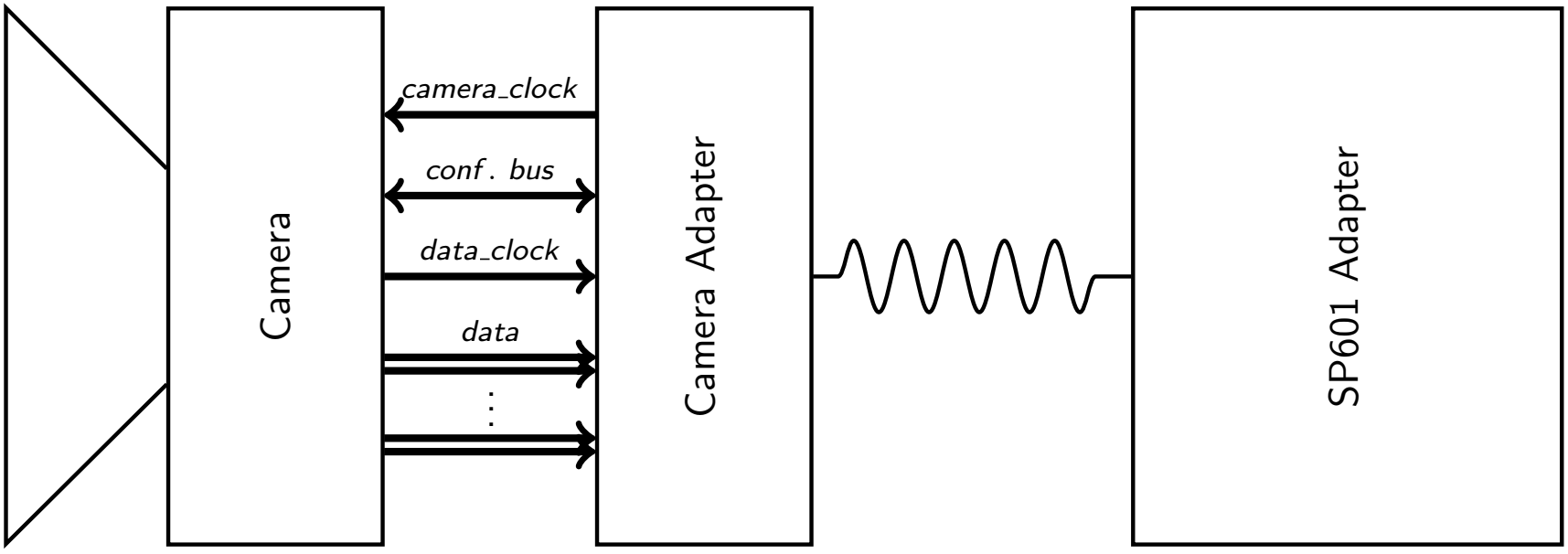
$$I'(x,y) = \frac{\sum_{i=0}^{2^d-1} i \cdot G_\sigma(i - I(x,y)) \cdot H(i)}{\sum_{i=0}^{2^d-1} G_\sigma(i - I(x,y)) \cdot H(i)}.$$

Appendix: Hybrid Median

GET Lab

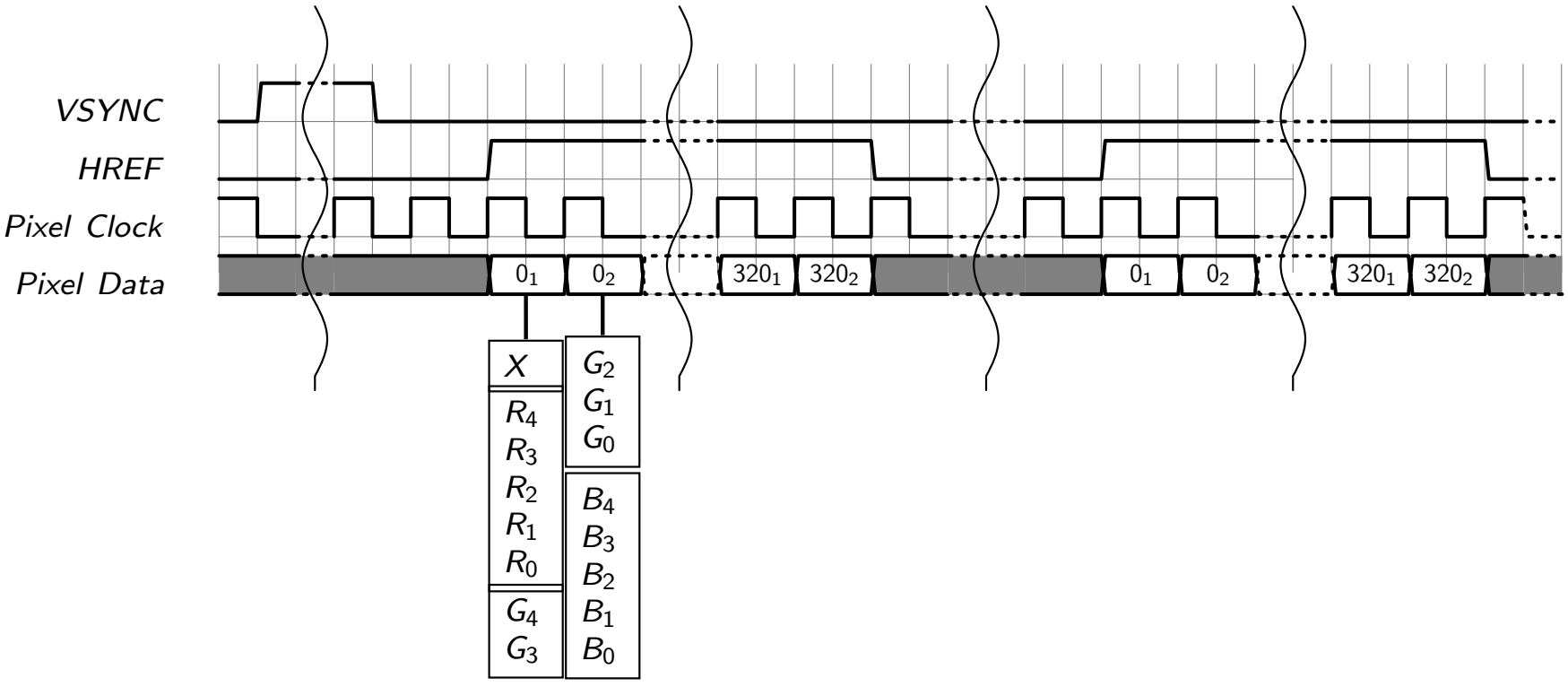
$$\text{med}\left(\text{med}\left(\ \right.\right.$$  $$\left.\left.\ \right), \text{med}\left(\ \right.\right.$$  $$\left.\left.\ \right), \text{med}\left(\ \right.\right.$$  $$\left.\left.\ \right)\right)$$

Appendix: Adapter Board Architecture



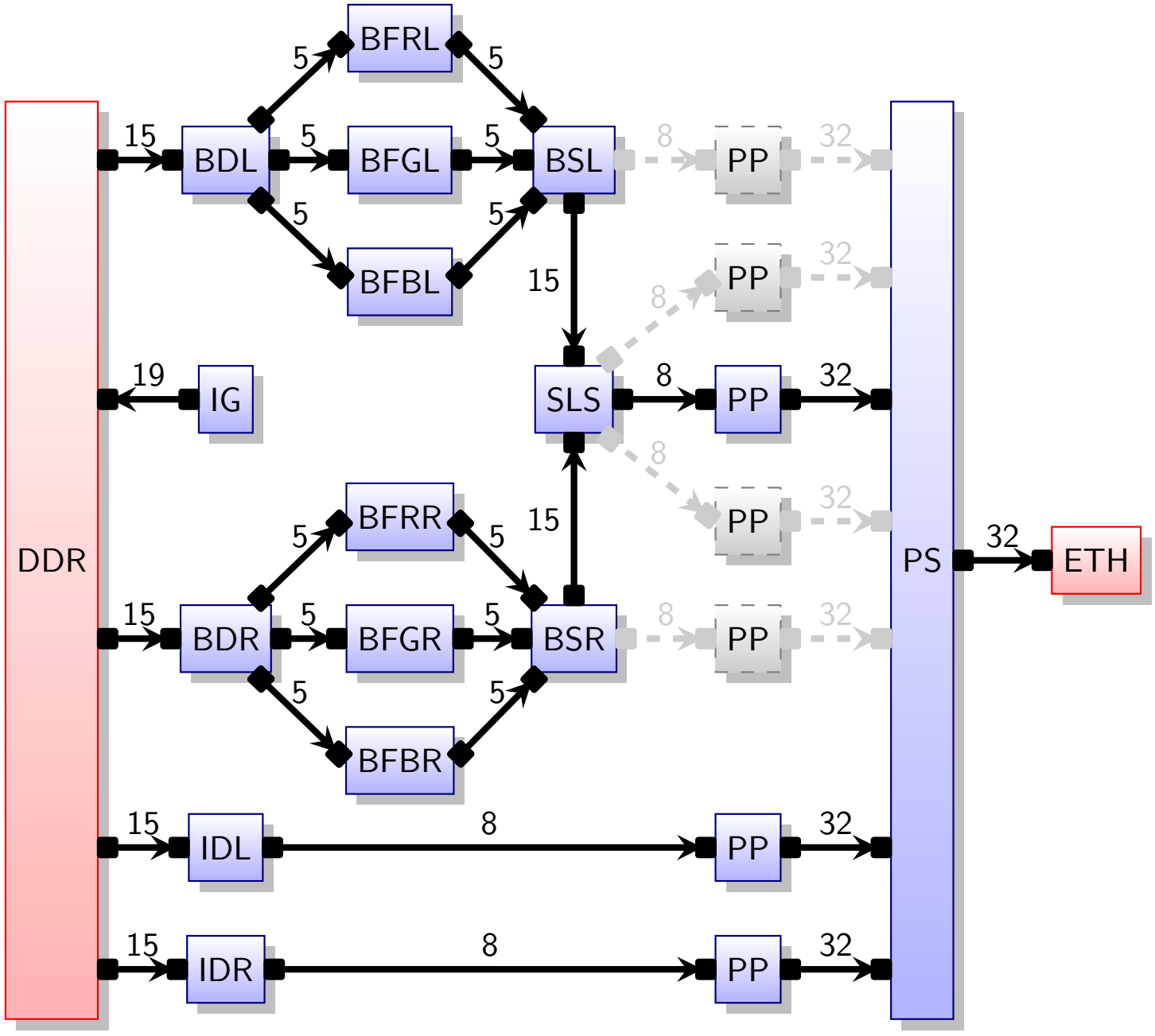
Appendix: Camera Timing Diagram

GET Lab



Appendix: Vision Core Architecture

GET Lab



Appendix: Vision Framework Clocks

GET Lab

Signal	Speed	Purpose
*CLK*600	600 MHz	Doubled clock for DDR memory
*CLK*300	300 MHz	Clock for DDR memory
*CLK*200	200 MHz	Clock for DDR Interface operations
*CLK*125	125 MHz	Clock for Ethernet transceiver
*CLK*100	100 MHz	Clock for Stereo Interface
*CLK*66	66 MHz	Clock for Vision Core
*CLK*25	25 MHz	Clock to drive external cameras

Appendix: Hardware Selection

GET Lab

	Development Board	Number of Slices/LE	Price	USB 2.0 High Speed	Gigabit Ethernet
Altera	Stratix IV GX	124 k	4,500 $	no	yes
	Stratix IV E	531 k	10,000 $	no	yes
	Arria II GX	124 k	1,500 $	no	yes
	Cyclone II	20 k	200 $	no	no
	Cyclone III	24 k	200 $	no	no
Xilinx	ML405	19 k	800 $	no	yes
	ML605	241 k	2,500 $	no	yes
	SP601	14 k	300 $	no	yes
	SP605	43 k	500 $	no	yes

Appendix: Vision Framework Slices

GET Lab

Component	Slices	Slice Register	Slice LUTs	BRAM blocks
Available on SP601	2278	18224	9112	32^2
Clock Manager	5	25	1	0
Block RAM Double Buffer	16	52	14	1
Stereo Interface (OV7)	382	675	889	2
OV7 Interface	181	295	407	0
Stereo Interface (MT9)	374	675	909	4
MT9 Interface	173	295	417	2
Camera Configurator	120	183	330	0
DDR Interface	336	634	668	0
DDR Port Interface	66	215	105	0
Ethernet Interface	536	1259	1040	4

Appendix: Vision Core Slices

GET Lab

		Slices	Slice Register	Slice LUTs	BRAM blocks
	Available on SP601	2278	18224	9112	32/64[3]
	Optimization				
Bilateral Filter	without	138	309	339	9
	+ pipel. convolution	35	106	93	9
	+ pipel. histogram update	51	103	132	18
	+ pipel. distributor/serializer	55	106	131	11
	+ non-recursive arrays	54	103	132	9
	+ primitive function flatten	56	106	130	14
Median Filter	without	64	107	145	14
	+ pipel. convolution	65	127	164	14
	+ pipel. histogram update	90	146	222	14
	+ pipel. distributor/serializer	96	146	218	14
	+ non-recursive arrays	94	145	211	14
	+ primitive function flatten	96	140	224	14
SLS	without	320	340	1046	16
	+ pipel. pixel conversion	279	398	874	36
	+ pipel. distance calculation	424	825	1089	36
	+ pipel. image calculation	403	826	1078	36
	+ non-recursive arrays	438	826	1065	36
PSLS	without	1104	1020	2987	33
	+ SLS optimizations	1101	2406	3295	30
	+ pipel. distributor/serializer	1155	2406	3309	30

Appendix: Vision Core Slices 2

GET Lab

		Slices	Slice Register	Slice LUTs	BRAM blocks
Available on SP601		2278	18224	9112	32/64
Filter	Optimized				
Median	no	1742	3382	5279	35
	yes	1949	4122	5953	35
Bilateral	no	2118	4474	6718	35
	yes	2224	5677	7623	35

Appendix: Vision Core Power Dissipation

GET Lab

	Optimization	Overall power consumption
Bilateral Filter	without	316.36 mW
	+ pipelining convolution	309.79 mW
	+ pipelining histogram update	309.89 mW
	+ pipelining distributor/serializer	306.85 mW
	+ non-recursive arrays	307.73 mW
	+ primitive function flatten	311.46 mW
Median Filter	without	309.14 mW
	+ pipelining convolution	310.79 mW
	+ pipelining histogram update	312.13 mW
	+ pipelining distributor/serializer	300.75 mW
	+ non-recursive arrays	299.14 mW
	+ primitive function flatten	298.61 mW
SLS	without	322.79 mW
	+ pipelining pixel conversion	321.77 mW
	+ pipelining distance calculation	321.28 mW
	+ pipelining image calculation	321.67 mW
	+ non-recursive arrays	320.43 mW
PSLS	without	379.84 mW
	+ SLS optimizations	379.84 mW
	+ pipel. distributor/serializer	359.15 mW