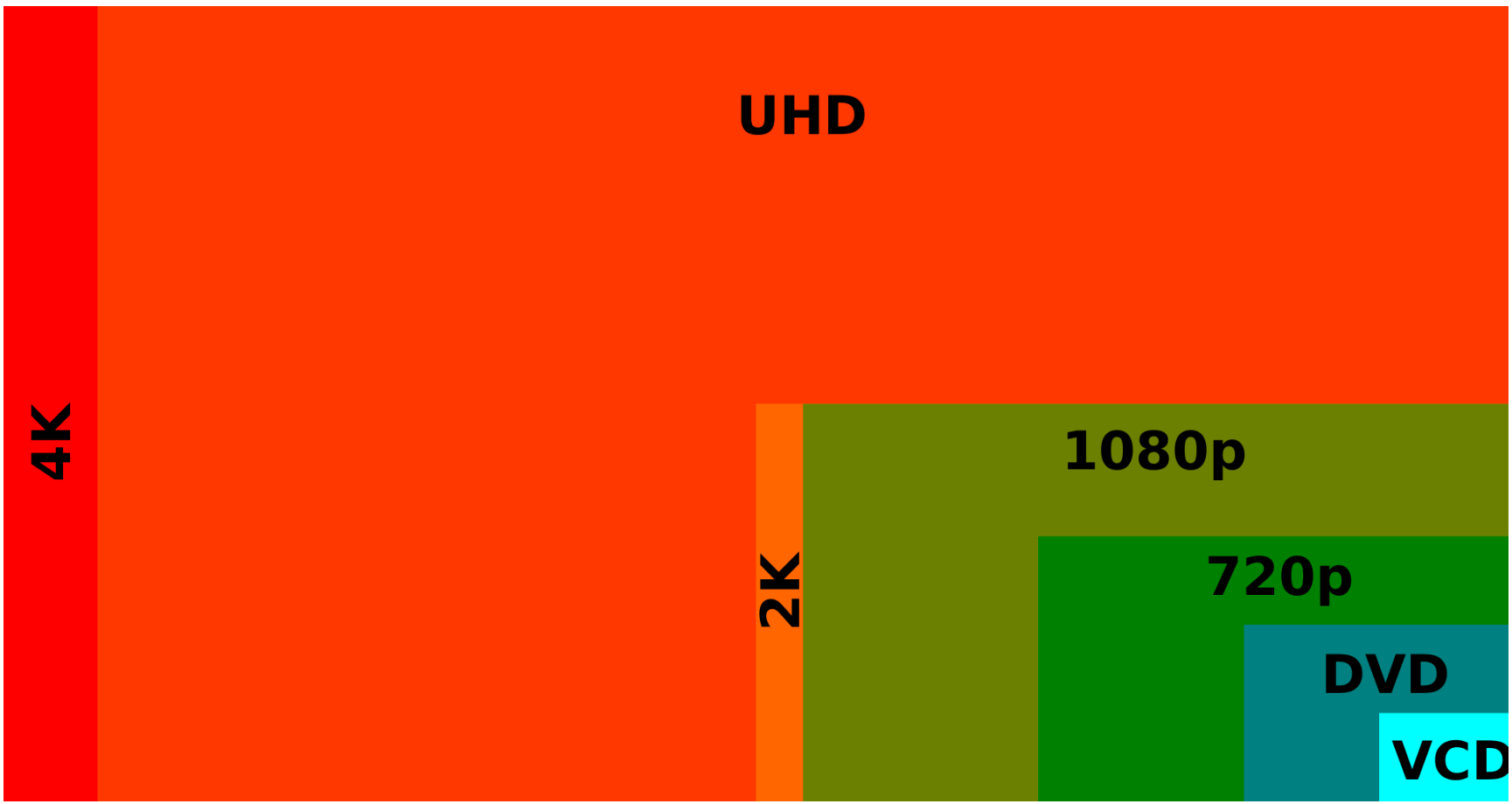
Einführung in Computer Microsystems
Sommersemester 2015
Hörsaalübung 5: Image Processing

TECHNISCHE UNIVERSITÄT DARMSTADT



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Hörsaalübung 5

Hörsaalübung 5

TECHNISCHE UNIVERSITÄT DARMSTADT

- Organisatorisches
- Vorstellung des Image Processors aus Übungsblatt 5

Organisatorisches zur Klausur: Übungen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Vorlesung am 15.07.2015
  - Letzte Fragen zur Vorlesung beantworten
  - Bitte in d120 Forum stellen bis Freitag
  - Sprechstunden mit Tutor in Raum E104
    - 14.7.2015 14:00 bis 16:30
    - 16.7.2015 14:00 bis 16:30
  - Beispielaufgaben
- Klausur
  - 21.07.2015: 12:00 bis 13:30
  - Umfasst sämtlichen Stoff der Veranstaltung
  - Praktischer Teil aus Übungen
  - Theoretischer Teil aus Vorlesungen

Hörsaalübung 5: Bildprozessor

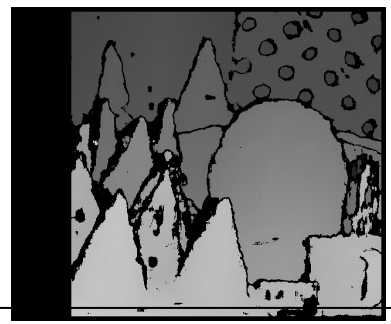
TECHNISCHE UNIVERSITÄT DARMSTADT

- Pixel kommen als Stream in den Prozessor
- Pixel werden bearbeitet
- Pixel kommen als Stream aus dem Prozessor

Hörsaalübung 5: Bildprozessor

TECHNISCHE UNIVERSITÄT DARMSTADT

- Tiefeninformationen aus zwei Bildern gewinnen (z.B. Kinect)



Hörsaalübung 5: Bildprozessor

TECHNISCHE UNIVERSITÄT DARMSTADT

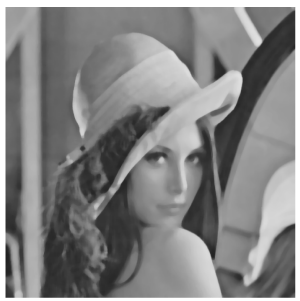
- Objekterkennung und Verfolgung

Hörsaalübung 5: Bildprozessor

TECHNISCHE UNIVERSITÄT DARMSTADT

- Filter pipeline


Blur


Median


Edge-Detect


High-Pass


Dilate


Erode

Hörsaalübung 5: Bildprozessor

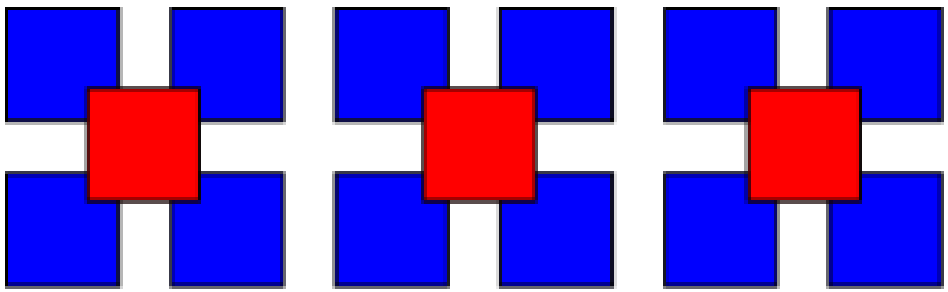
TECHNISCHE UNIVERSITÄT DARMSTADT

- FPGAs ausgezeichnet geeignet
- Hohe Parallelität
- Hohe Flexibilität
- Hohe Konnektivität

Hörsaalübung 5: Bildprozessor

TECHNISCHE UNIVERSITÄT DARMSTADT

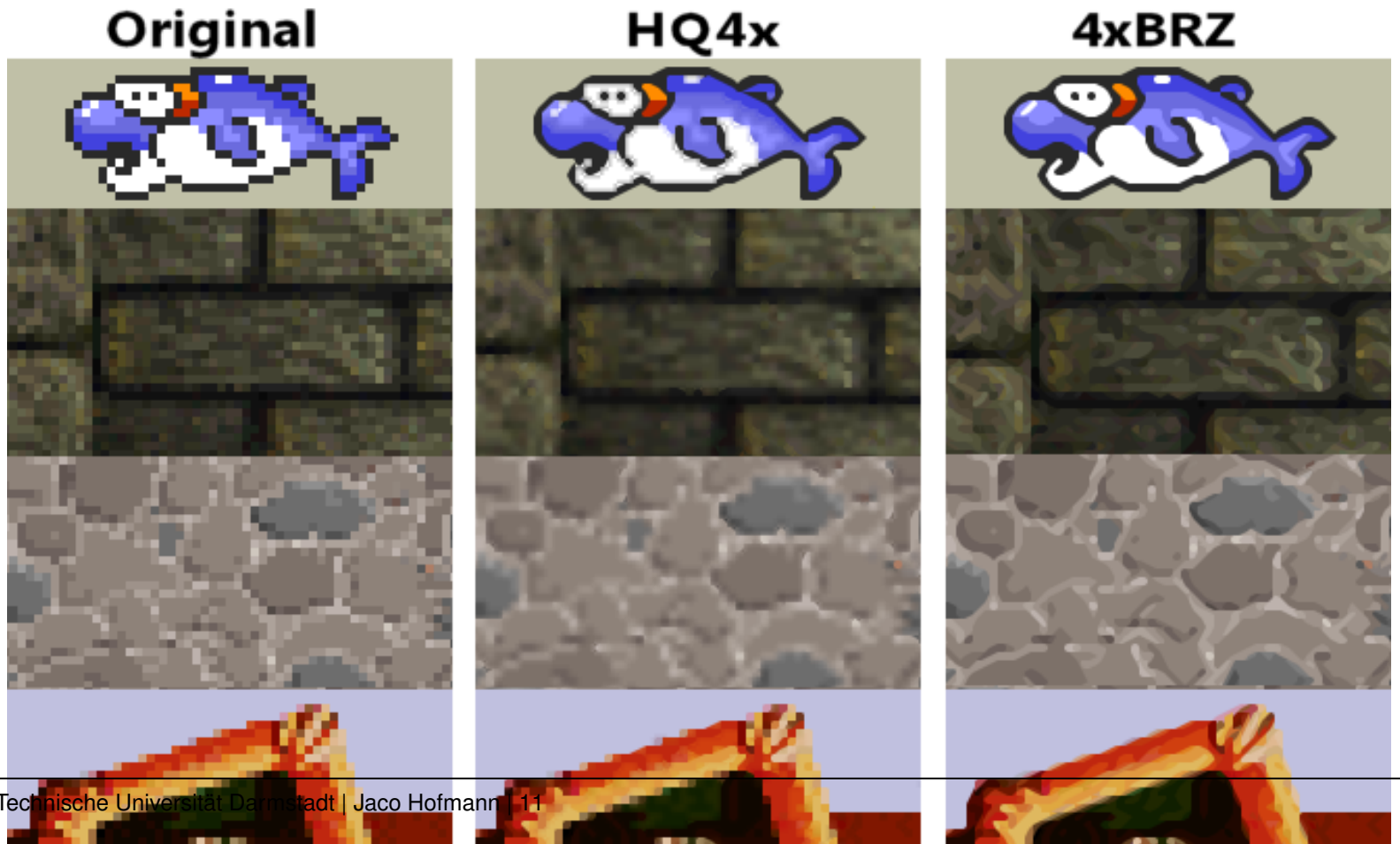
- Einfaches Beispiel für diese Übung
- Auflösung von Bild reduzieren
- Faktor 2 in Höhe und Breite
- Ausgabepixel ist Durchschnitt der vier relevanten Eingabepixel



Hörsaalübung 5: Bildprozessor

TECHNISCHE UNIVERSITÄT DARMSTADT

- Bessere und spezialisierte Algorithmen verfügbar



Hörsaalübung 5: Datentypen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Datentyp für (RGB-)Pixel

```
1    typedef struct {
2        UInt#(8) r;
3        UInt#(8) g;
4        UInt#(8) b;
5    } Pixel deriving(Bits, Eq);
```

Hörsaalübung 5: Interface

TECHNISCHE UNIVERSITÄT DARMSTADT

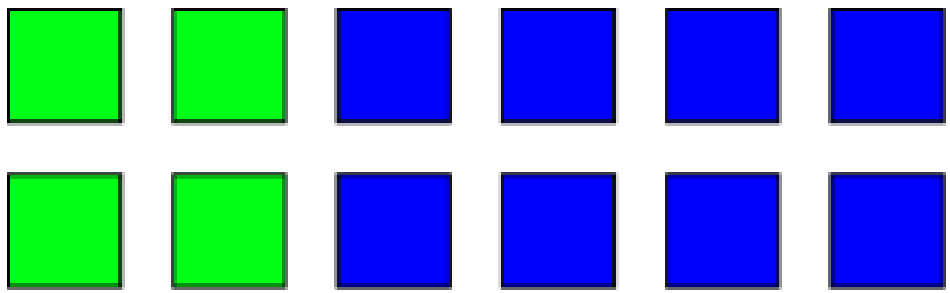
- Eingabestream in den Core leiten
- Ausgabestream weiterleiten
- Zeilenbreite einstellen

```
1    interface ImageProcessor;
2        interface Server#(Pixel, Pixel) server;
3        method Action setRowWidth(UInt#(12) rowWidth);
4    endinterface
```

Hörsaalübung 5: Eingabe

TECHNISCHE UNIVERSITÄT DARMSTADT

- Welche Daten der Eingabe benötigen wir?

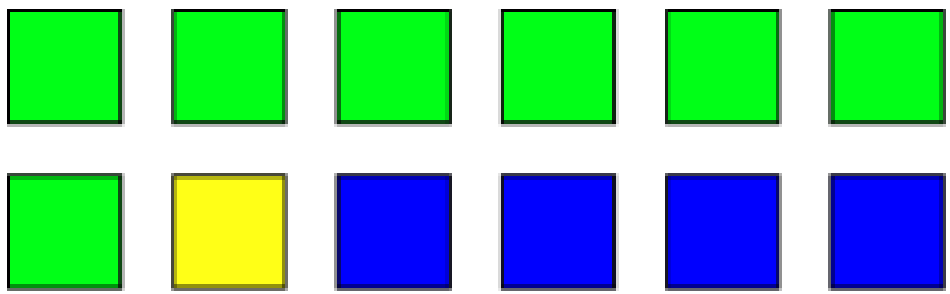


- Zwei Pixel aus der ersten Reihe und zwei Pixel aus der Zweiten!
- Wann können wir mit der Berechnung anfangen?

Hörsaalübung 5: Eingabe

TECHNISCHE UNIVERSITÄT DARMSTADT

- Speichern der ersten Reihe
- Dann:
  - Erster Takt: Pixel speichern
  - Zweiter Takt: Durchschnitt berechnen



Hörsaalübung 5: Eingabe

TECHNISCHE UNIVERSITÄT DARMSTADT

- Server Put Interface
- Aufgaben:
  - Anzahl der eingetroffenen Pixel zählen
  - Pixel speichern
  - Berechnung veranlassen

Hörsaalübung 5: Eingabe

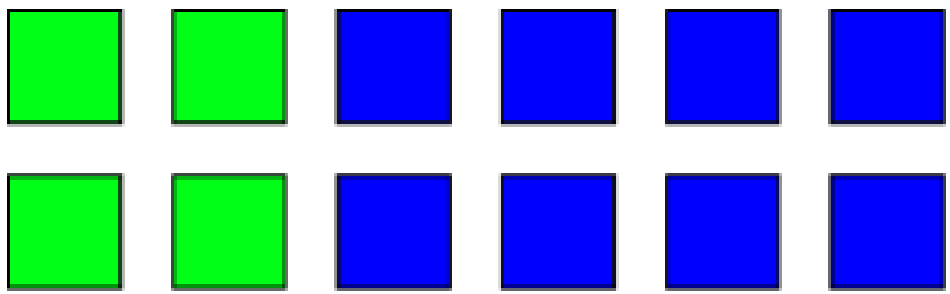
TECHNISCHE UNIVERSITÄT DARMSTADT

```
1   method Action put(Pixel p) if(outFIFO.notFull());
2       mem.upd(currentStoragePosition, p);
3       if(pixelThisLine == (rowWidthMax * 2) - 1) begin
4           pixelThisLine <= 0;
5           currentStoragePosition <= 0;
6           doCalc[1] <= True;
7       end else if(pixelThisLine >= rowWidthMax) begin
8           if(currentStoragePosition == rowWidthMax) begin
9               currentStoragePosition <= rowWidthMax + 1;
10          end else begin
11              currentStoragePosition <= rowWidthMax;
12              doCalc[1] <= True;
13          end
14          pixelThisLine <= pixelThisLine + 1;
15      end else begin
16          currentStoragePosition <= currentStoragePosition + 1;
17          pixelThisLine <= pixelThisLine + 1;
18      end
19  endmethod
```

Hörsaalübung 5: Eingabe

TECHNISCHE UNIVERSITÄT DARMSTADT

- Kommen wir ohne speichern der gesamten Reihe aus?
- Pixel direkt aufsummieren
- Spart Speicher und verkürzt möglicherweise kritischen Pfad
- Hier einfache Lösung behandelt



Hörsaalübung 5: Berechnung

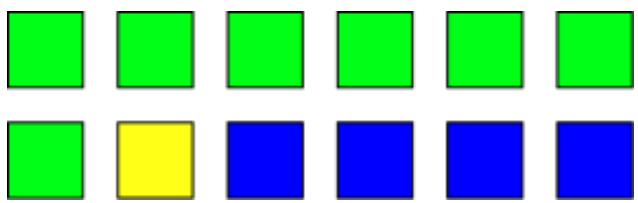
TECHNISCHE UNIVERSITÄT DARMSTADT

- Berechnung in zwei Schritten
  - Pixel aus Speicher lesen
  - Durchschnitt berechnen

Hörsaalübung 5: Berechnung

TECHNISCHE UNIVERSITÄT DARMSTADT

```
1     rule getPixels (doCalc[0]);
2         let pixelPos0 = currentCalcPosition;
3         let pixelPos1 = addOneAndCap(currentCalcPosition);
4         let pixelPos2 = rowWidthMax;
5         let pixelPos3 = rowWidthMax + 1;
6         let pix       = tuple4(mem.sub(pixelPos0),
7                                mem.sub(pixelPos1),
8                                mem.sub(pixelPos2),
9                                mem.sub(pixelPos3));
10        storeInter.enq(pix);
11        currentCalcPosition <= addOneAndCap(pixelPos1);
12        doCalc[0] <= False;
13    endrule
```



Hörsaalübung 5: Berechnung

TECHNISCHE UNIVERSITÄT DARMSTADT

```
1   rule calc;
2       let data = storeInter.first(); storeInter.deq();
3       let pix0 = tpl_1(data);
4       let pix1 = tpl_2(data);
5       let pix2 = tpl_3(data);
6       let pix3 = tpl_4(data);
7       let r = calcAverage(pix0.r, pix1.r, pix2.r, pix3.r);
8       let g = calcAverage(pix0.g, pix1.g, pix2.g, pix3.g);
9       let b = calcAverage(pix0.b, pix1.b, pix2.b, pix3.b);
10      Pixel p = Pixel{r:r,g:g,b:b};
11      outFIFO.enq(p);
12  endrule
```

Hörsaalübung 5: Achtung: Bitbreite!

TECHNISCHE UNIVERSITÄT DARMSTADT

```
1    function UInt#(8) calcAverage(UInt#(8) c0, UInt#(8) c1, UInt#(8) c2,
     ↪  UInt#(8) c3);
2        UInt#(10) c0t = extend(c0);
3        UInt#(10) c1t = extend(c1);
4        UInt#(10) c2t = extend(c2);
5        UInt#(10) c3t = extend(c3);
6        return truncate((c0t + c1t + c2t + c3t) / 4);
7    endfunction
```

Hörsaalübung 5: Testen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Bluespec erlaubt Einbindung von externen C Funktionen
- Echtes Bild laden und nach Bearbeitung durch simulierten Core speichern

Hörsaalübung 5: Testen

TECHNISCHE UNIVERSITÄT DARMSTADT

- BDPI: C Funktionen direkt in Bluespec

```
1   import "BDPI" function ActionValue#(Bit#(32)) getPixel(UInt#(64) ptr,
     ↪   UInt#(32) x, UInt#(32) y);
```

```
1   unsigned long long getPixel(unsigned long long myImg, int x, int y)
```