Projektdokumentation

„LBP Benchmark zwischen Software- und Hardwarelösung“

Gruppe 5a

Michael Erkel

Simon Friedrich

**Sergej Zuyev**

**Inhaltsverzeichnis**

[1Projektinitiierung 3](#_Toc493693894)

[1.1 Arbeitspakete (für Details siehe Anhang Arbeitspakete) 3](#_Toc493693895)

[1.2 Projektrahmen 4](#_Toc493693896)

[1.2.1 Personal und Ressourcen 4](#_Toc493693897)

[1.2.2 Ergebnisse 5](#_Toc493693898)

[1.2.3 Zuständigkeiten 5](#_Toc493693899)

[1.2.4 Zeitpläne 5](#_Toc493693900)

[2 Projektplanung 6](#_Toc493693901)

[2.1 Projektstrukturplan 6](#_Toc493693902)

[2.2 Gantt-Diagramm 7](#_Toc493693903)

[3 Projektcontrolling 8](#_Toc493693904)

[3.1 Messverfahren 8](#_Toc493693905)

[3.1.1 Ausgefülltes Prüfprotokoll.. 9](#_Toc493693906)

[3.2 Portierung VHDL-Code auf DE1-Soc-Board 10](#_Toc493693907)

[3.2.1 Prüfprotokoll 12](#_Toc493693908)

[3.3 Open-CL-Lösung 13](#_Toc493693909)

[3.3.1 Prüfprotokoll 14](#_Toc493693910)

[3.4 GANTT-Diagramm mit Vergleich zur Projektplanung 16](#_Toc493693911)

[4 Projektende 17](#_Toc493693912)

[4.1 Aufgetretene Probleme 17](#_Toc493693913)

[4.1.1 MATlab Anwendung 17](#_Toc493693914)

[4.1.2 VHDL-Umsetzung 17](#_Toc493693915)

[4.1.3 Open-CL-Umsetzung 18](#_Toc493693916)

[4.2 Resümee 18](#_Toc493693917)

# 1Projektinitiierung

Nach der gegenseitigen Vorstellung der Teammitglieder studierte jeder für sich die Projektunterlagen, die vom Auftraggeber bereitgestellt wurden (Lasten/Pflichtenheft im Anhang). Gemeinsam im Team diskutierten wir über die Unterlagen und legten die weiteren Arbeitsschritte fest.

Der erste Schritt in der Projektinitiierung ist das Erstellen einer Anforderungsliste. Wir haben hierfür die relevanten Details in den Projektunterlagen auf einem Flipchart zusammengestellt und diese zu Arbeitspaketen zusammengefasst.

## 1.1 Arbeitspakete (für Details siehe Anhang Arbeitspakete)

AP 1.1: Beschaffung: DE1-SOC-Board

AP 1.2: Einrichtung: Arbeitsumgebung

AP 1.3: Beschaffung: VHDL-Quellcode

AP 1.4: Beschaffung: Masterthesis

AP 1.5: Test: Entwicklungsumgebung

AP 2.1: Analyse:Portierbarkeit des VHDL-Quellcodes auf OpenCl prüfen

AP 2.2: Analyse:Portierbarkeit des VHDL-Quellcodes auf das DE1-SoC-Board prüfen

AP 2.3: Analyse:Ansatz für ein Messverfahren erarbeiten

AP 3.1: Entwicklung: Messverfahren

AP 3.2: Entwicklung: VHDL-Code auf DE1-Soc-Board portieren

AP 3.3: Entwicklung: Umsetzung in OpenCL

AP 4.1: Test: Messverfahren

AP 4.2: Test: VHDL-Umsetzung auf DE1-Soc-Board

AP 4.3: Test: OpenCl-Umsetzung

AP 5.1: Präsentation

AP 5.2: Dokumentation

AP 5.3: Produktvorstellung

## 1.2 Projektrahmen

Nachdem die Arbeitspakete klar definiert waren, wurden nun die groben Rahmengrundlagen für die Realisierung des Projektes festgelegt.

### 1.2.1 Personal und Ressourcen

**Personal:**

* Michael Erkel
* Simon Friedrich
* Sergej Zuyev

**Ressourcen:**

**Software:**

* Altera Softwareumgebung
* Linux-Distribution für OpenCL
* Mathworks Matlab R2012a

**Hardware:**

* DE1-SOC-Board
* Entwicklungs/Präsentationrechner mit Peripherie
* Micro-SD-Speicherkarte mit mindestens 4 GB

**Unterlagen:**

* Vollständige Dokumentation von DE1-SOC-Board
* Masterarbeit Herr Kumar
* Lastenheft
* Pflichtenheft

### 1.2.2 Ergebnisse

* Messverfahren ist entwickelt
* VHDL-Code wurde auf DE1-SOC-Board portiert
* Open-CL-Lösung ist entwickelt
* Dokumentation ist erstellt
* Präsentation ist erstellt

### 1.2.3 Zuständigkeiten

Michael Erkel:

* VHDL-Code auf DE1-SOC-Board portieren
* Dokumentation erstellen

Simon Friedrich:

* Messverfahren entwickeln
* Präsentation erstellen

Sergej Zuyev:

* Open-CL-Lösung entwickeln

### 1.2.4 Zeitpläne

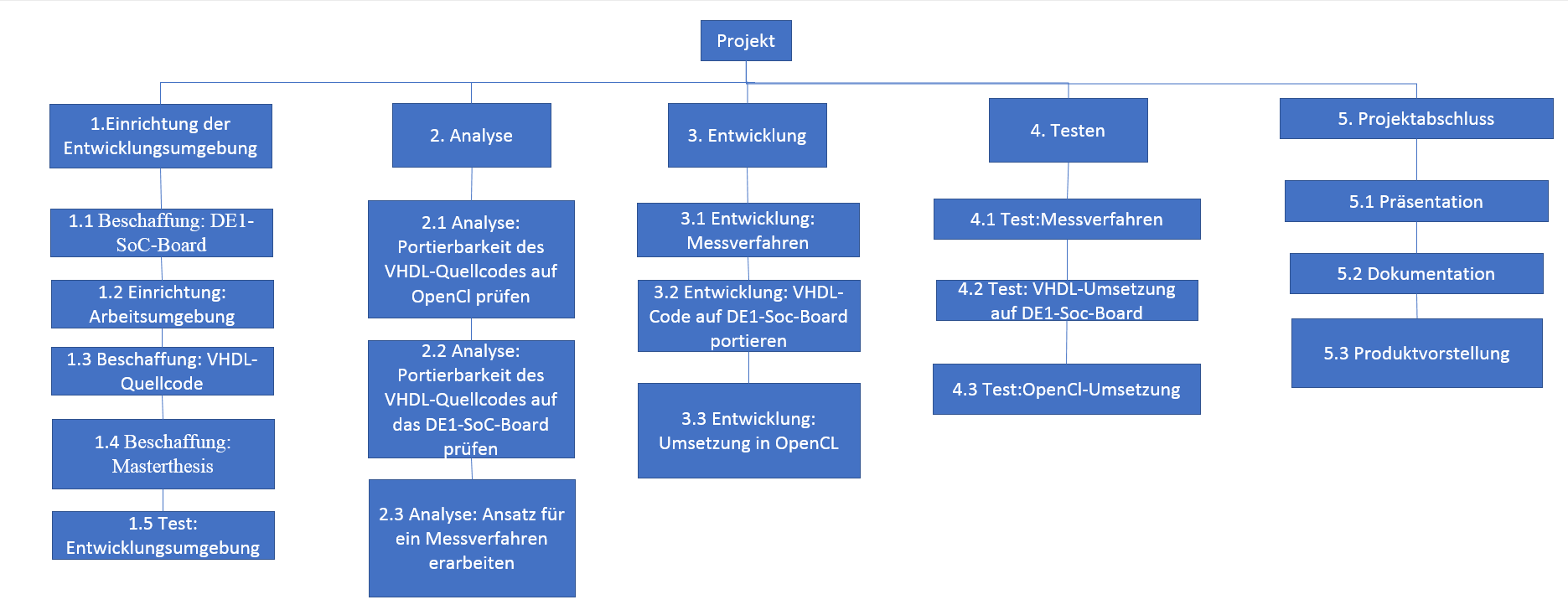
1. Einrichten der Entwicklungsumgebung 01.08.2017 – 09.08.2017
2. Analyse 10.08.2017 – 18.08.2017
3. Entwicklung 19.08.2017 – 09.09.2017
4. Testen 10.09.2017 – 11.09.2017
5. Projektabschluss 12.09.2017 – 21.09.2017

# 2 Projektplanung

Im nächsten Schritt ging es weiter mit der Projektplanung. Die Vorgehensweise wurde detaillierter geplant und die Zeiten und Ressourcen in den Arbeitspaketen nachgepflegt.

## 2.1 Projektstrukturplan

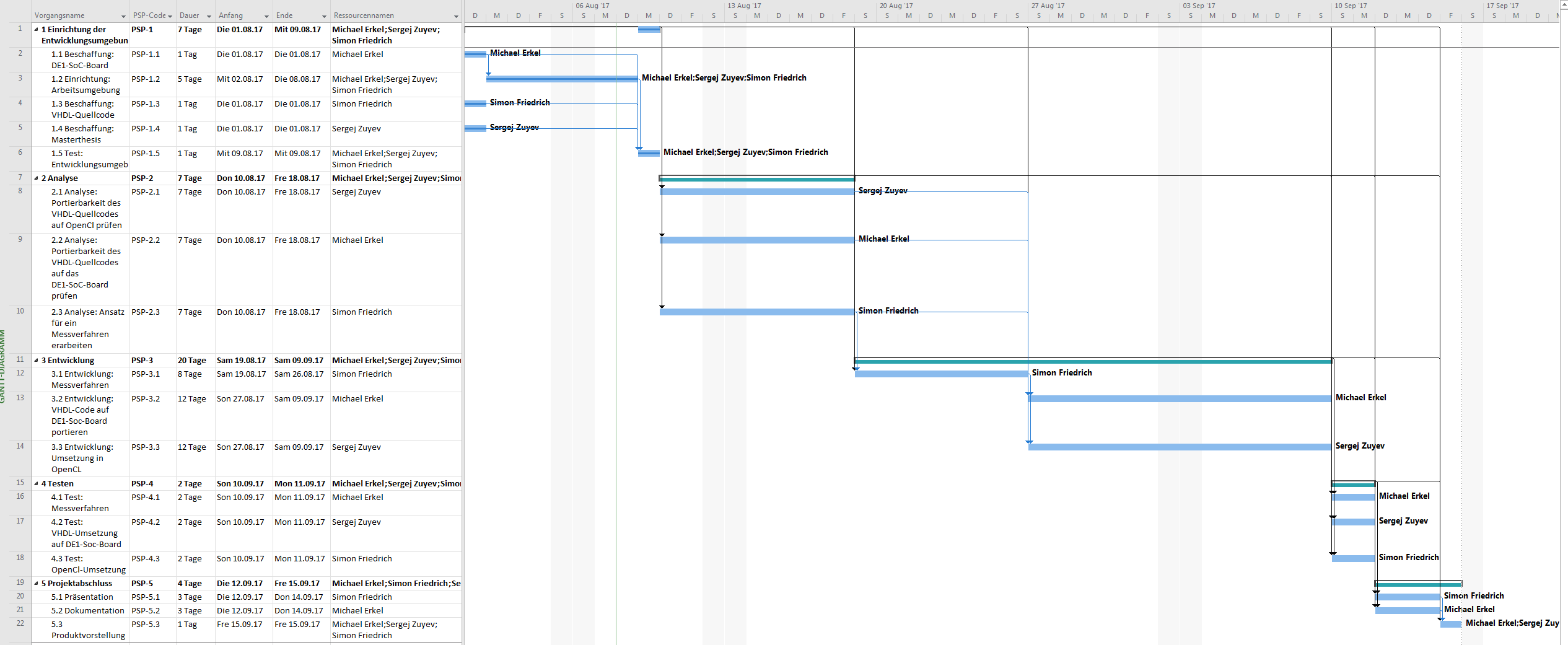
Abgeleitet aus den Arbeitspaketen wurde folgendes PSP erstellt.



## 2.2 Gantt-Diagramm

Danach haben wir uns auf den zeitlichen Auflauf des Projektes geeinigt, welchen wir mit Hilfe des Gantt- Diagramm dargestellt haben. Im Diagramm sind die Sammel- und Einzelvorgänge zu sehen und deren zeitliche Einteilung.

Des weiteren ist im Diagramm die Ressourcenplanung zu erkennen.



# Projektcontrolling

Nun folgte das Projektcontrolling.

Nach jedem Softwareabschnitt haben wir anhand unserer Prüfprotokolle den Projektstatus überprüft.

## 3.1 Messverfahren

**Grundüberlegung:**

Zur Umsetzung des Benchmark-Tests ist ein Messverfahren der Verarbeitung durch den LBP-Operator von Nöten. Weiterhin wird eine Möglichkeit der Darstellung benötigt.

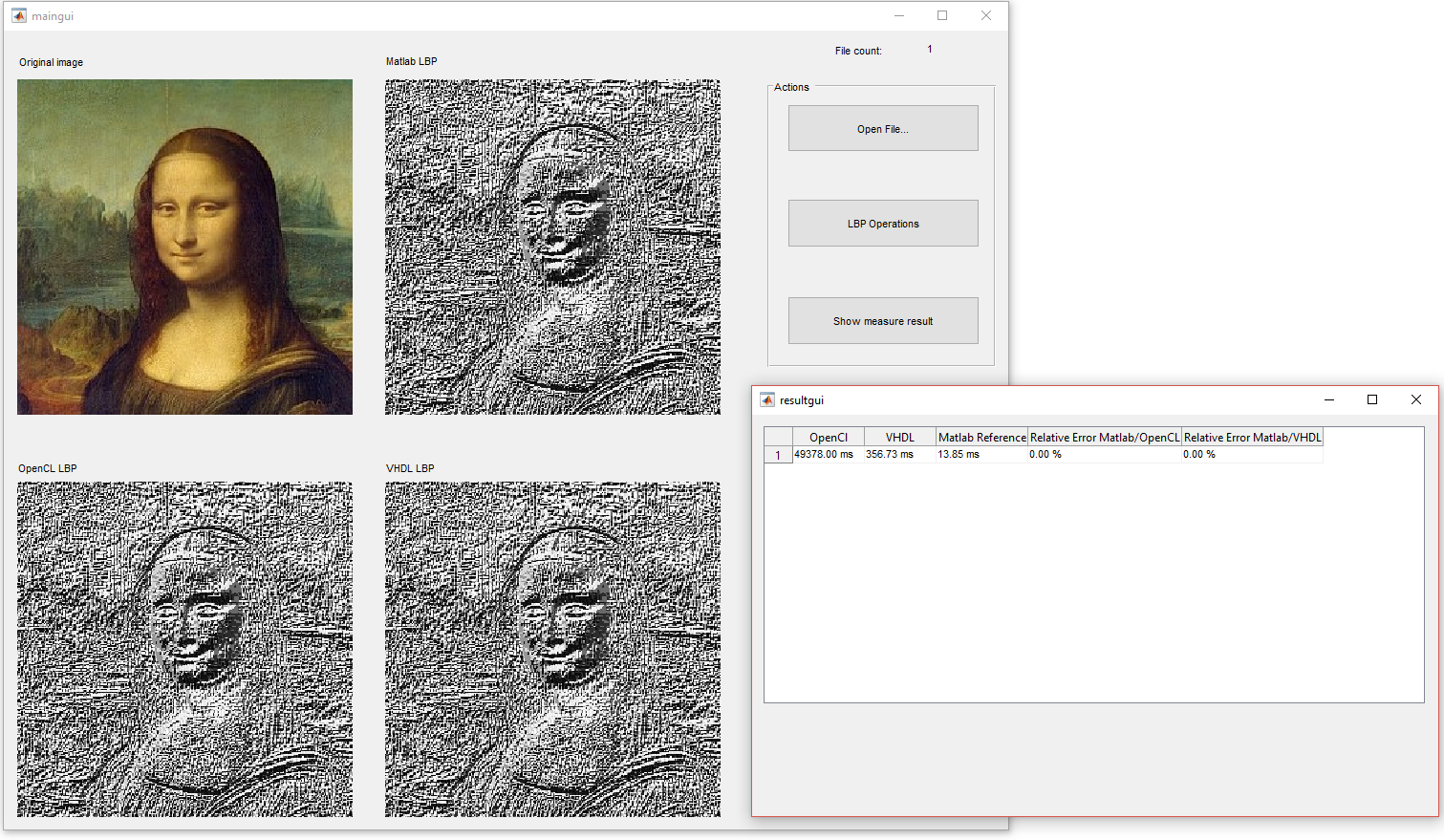
**Umsetzung:**

Es wird eine MATLAB GUIDE Anwendung entwickelt. Diese bietet diverse Möglichkeiten der Darstellung (bspw. Tabellarisch, als Diagramm, etc.), sowie einfache Verfahren zur Messung der unterschiedlichen Lösungen.

Als Zusatz kann ein Vergleich der Ergebnisse mit der MATLAB-Lösung des LBP-Operators von Herrn Kumar angestellt werden.

Zur konkreten Messung der Verfahren wird eine Zeitmessung inklusive der Datenübertragung an das Board durchgeführt. In Abhängigkeit des Aufwandes wird optional eine Zeitmessung ohne Datenübertragung umgesetzt.

Die Anwendung soll ebenfalls Dateien (Bilder) öffnen können und diese zur Vorbereitung in Graustufen umwandeln. Zudem werden die umgewandelten Daten von der Anwendung an das DE1-SoC Board gesendet.



### 3.1.1 Ausgefülltes Prüfprotokoll

1.Funktionsprüfung

**1.Testfall**

Dateiname1: IMG0.jpg

Dateiname2: IMG0.jpg

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plattform | Dateien  konnten Fehlerfrei eingelesen werden | | Dateien  werden dargestellt | | Übereinstimmung in % | Kommentar |
| ja | nein | ja | nein |
| VHDL |  | x |  | x | 0 | Nicht lauffähig |
| OpenCL | x |  | x |  | N/A | FPGA: falsch, GPU: visuell korrekt |
| Matlab  (Referenz) | x |  | x |  | 100 |  |

**2.Testfall**

Dateiname1: IMG1.jpg

Dateiname2: IMG1.jpg

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plattform | Dateien  konnten Fehlerfrei eingelesen werden | | Dateien  werden dargestellt | | Übereinstimmung in % | Kommentar |
| ja | nein | ja | nein |
| VHDL |  | x |  | x | 0 | Nicht lauffähig |
| OpenCL | x |  | x |  | N/A | FPGA: falsch, GPU: visuell korrekt |
| Matlab  (Referenz) | x |  | x |  | 100 |  |

**3.Testfall**

Dateiname1: IMG2.jpg

Dateiname2: IMG2.jpg

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plattform | Dateien  konnten Fehlerfrei eingelesen werden | | Dateien  werden dargestellt | | Übereinstimmung in % | Kommentar |
| ja | nein | ja | nein |
| VHDL |  | x |  | x | 0 | Nicht lauffähig |
| OpenCL | x |  | x |  | N/A | FPGA: falsch, GPU: visuell korrekt |
| Matlab  (Referenz) | x |  | x |  | 100 |  |

**4.Testfall**

Dateiname1: IMG3.jpg

Dateiname2: IMG3.jpg

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plattform | Dateien  konnten Fehlerfrei eingelesen werden | | Dateien  werden dargestellt | | Übereinstimmung in % | Kommentar |
| ja | nein | ja | nein |
| VHDL |  | x |  | x | 0 | Nicht lauffähig |
| OpenCL | x |  | x |  | N/A | FPGA: falsch, GPU: visuell korrekt |
| Matlab  (Referenz) | x |  | x |  | 100 |  |

**5.Testfall**

Dateiname1: IMG4.jpg

Dateiname2: IMG4.jpg

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plattform | Dateien  konnten Fehlerfrei eingelesen werden | | Dateien  werden dargestellt | | Übereinstimmung in % | Kommentar |
| ja | nein | ja | nein |
| VHDL |  | x |  | x | 0 | Nicht lauffähig |
| OpenCL | x |  | x |  | N/A | FPGA: falsch, GPU: visuell korrekt |
| Matlab  (Referenz) | x |  | x |  | 100 |  |

**6.Testfall**

Dateiname1: IMG5.jpg

Dateiname2: IMG5.jpg

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plattform | Dateien  konnten Fehlerfrei eingelesen werden | | Dateien  werden dargestellt | | Übereinstimmung in % | Kommentar |
| ja | nein | ja | nein |
| VHDL |  | x |  | x | 0 | Nicht lauffähig |
| OpenCL | x |  | x |  | N/A | FPGA: falsch, GPU: visuell korrekt |
| Matlab  (Referenz) | x |  | x |  | 100 |  |

**7.Testfall**

Dateiname1: IMG6.jpg

Dateiname2: IMG6.jpg

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plattform | Dateien  konnten Fehlerfrei eingelesen werden | | Dateien  werden dargestellt | | Übereinstimmung in % | Kommentar |
| ja | nein | ja | nein |
| VHDL |  | x |  | x | 0 | Nicht lauffähig |
| OpenCL | x |  | x |  | N/A | FPGA: falsch, GPU: visuell korrekt |
| Matlab  (Referenz) | x |  | x |  | 100 |  |

**8.Testfall**

Dateiname1: IMG7.jpg

Dateiname2: IMG7.jpg

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plattform | Dateien  konnten Fehlerfrei eingelesen werden | | Dateien  werden dargestellt | | Übereinstimmung in % | Kommentar |
| ja | nein | ja | nein |
| VHDL |  | x |  | x | 0 | Nicht lauffähig |
| OpenCL | x |  | x |  | N/A | FPGA: falsch, GPU: visuell korrekt |
| Matlab  (Referenz) | x |  | x |  | 100 |  |

**9.Testfall**

Dateiname1: IMG8.jpg

Dateiname2: IMG8.jpg

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plattform | Dateien  konnten Fehlerfrei eingelesen werden | | Dateien  werden dargestellt | | Übereinstimmung in % | Kommentar |
| ja | nein | ja | nein |
| VHDL |  | x |  | x | 0 | Nicht lauffähig |
| OpenCL | x |  | x |  | N/A | FPGA: falsch, GPU: visuell korrekt |
| Matlab  (Referenz) | x |  | x |  | 100 |  |

**10.Testfall**

Dateiname1: IMG9.jpg

Dateiname2: IMG9.jpg

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plattform | Dateien  konnten Fehlerfrei eingelesen werden | | Dateien  werden dargestellt | | Übereinstimmung in % | Kommentar |
| ja | nein | ja | nein |
| VHDL |  | x |  | x | 0 | Nicht lauffähig |
| OpenCL | x |  | x |  | N/A | FPGA: falsch, GPU: visuell korrekt |
| Matlab  (Referenz) | x |  | x |  | 100 |  |

2. Vergleichbarkeit der einzelnen Messungen

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Plattform: | Messergebnis in ms (s für OpenCL) | | | | | | | | | | Mittelwert  In ms |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Matlab | 5 | 5 | 4 | 6 | 4 | 5 | 4 | 4 | 4 | 4 | 4.5 |
| VHDL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OpenCL | 46 | 46 | 46 | 46 | 46 | 46 | 46 | 46 | 46 | 46 | 46000 |

## Portierung VHDL-Code auf DE1-Soc-Board

**Grundüberlegungen:**

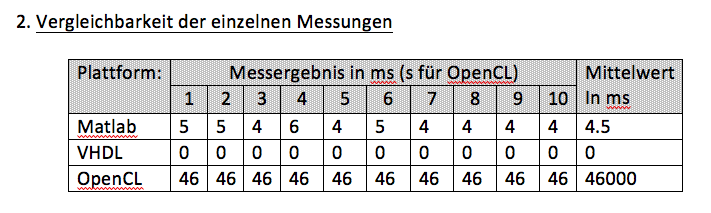
* LBP Operator – Verechnet ein 3x3 Datenfenster
* LBP-Kernel - Verarbeitet Daten in einem Speicherbereich
* LBP Kernel Array - Mehrere Kernel
* Processing unit
  + Hauptrecheneinheit, beinhaltet:
    - Eingabepuffer
    - Ausgabepuffer
    - Kernel array
* Processing unit input memory
  + Eingabepuffer, Speicherbereich für das komplette Eingangsbild
  + Block-RAM basiert
  + Multi-port fähig
* Processing unit output memory
  + Ausgabepuffer, Speicherbereich für das verarbeitete Ausgangsbild
  + Block-RAM basiert
  + Bedingt Multi-port fähig
* Control unit
  + Hauptsteuereinheit
  + Zuständig für Datentransfers und Steuerung vom Host
  + Implementiert das oben beschriebene (Matlab -> ZedBoard) Protokoll
* Host interface UART (PHY)
  + UART Schnittstelle
* Avalon Schnittstelle
  + Kommunikation mit dem ARM Prozessor über die Avalon / AXI Schnittstelle

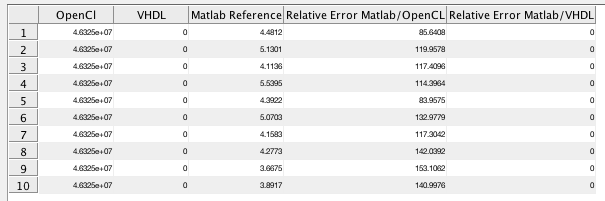
**Umsetzung:**

* Für das DE1-SoC ist der Aufwand enorm:
  + - Avalon MM Schnittstelle, damit eine Anbindung an den ARM Hauptprozessor stattfinden kann
    - Linux-Kernelmodul als Treiber
    - Userspace-Programm für den eigentlichen Datentransfer (Ansteuerung vom Host via SSH, vgl. dazu DatenTransfer für OpenCL)
* Es wurde ein Qsys-Projekt erstellt
  + - lbp-modul mit Avalon Schnittstelle als IP-Core
    - Hard Processor System IP-Core
* Umsetzung konnte nicht komplett abgeschlossen werden.

**Screenshots**

### 3.2.1 Prüfprotokoll





### 3.2.2 Simulation LBP-Kernel mit „Sliding Window“



## 3.3 Open-CL-Lösung

**Umsetzung:**

2 Varianten wurden entworfen. Beide wurden zuvor auf dem Rechner getestet (7700K, 32GB, GTX1080Ti)

* Die erste Variante war nicht lauffähig aufgrund von Überschreitung der FPGA-Ressourcen. Wird im Kapitel 4.1.3 genauer beschrieben.
* Die zweite Variante wurde stark vereinfacht, ohne Gleitkomma- und trigonometrischen Operationen umgesetzt, dadurch konnte das Speicherproblem umgangen werden.
  + Anzahl Samples wurde auf 8 fixiert

Zum Testen auf dem PC wurde ein Programm in Python entwickelt.

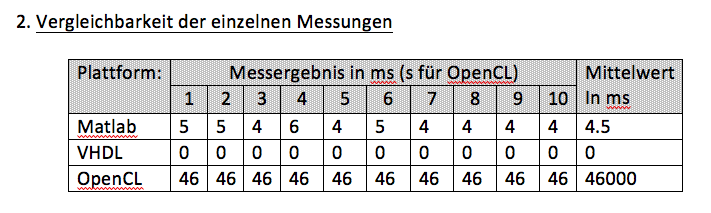
**Ergebnis:**

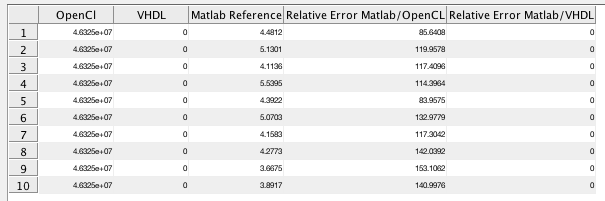
Die 2. Variante kann verwendet werden und belegt 19-20% der Ressourcen.

Es können mehrere Kernels synthetisiert werden.

**Screenshots**

### 3.3.1 Prüfprotokoll

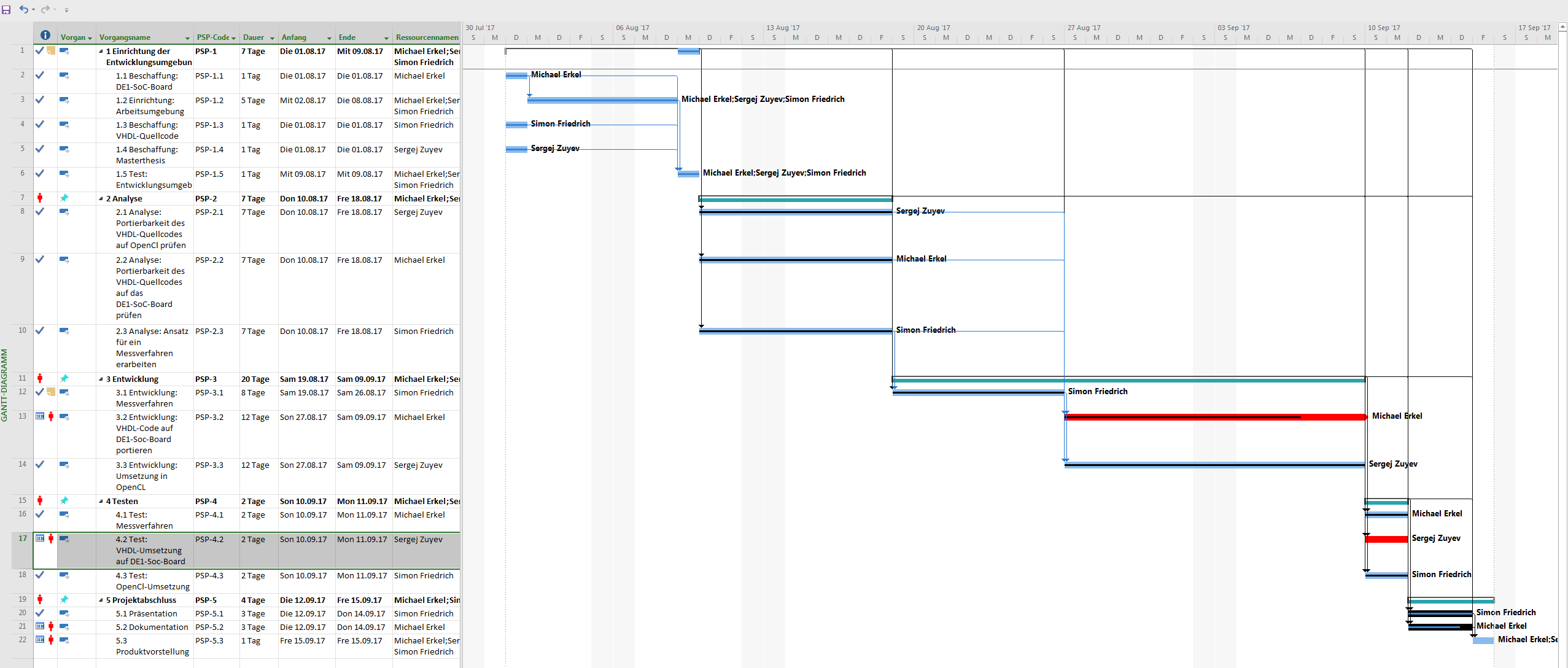




## 3.4 GANTT-Diagramm mit Vergleich zur Projektplanung

Die Themen von Herrn Erkel konnten aus gesundheitlichen Gründen nicht beendet werden.

Eine Übergabe an Herrn Zuyev ist erfolgt.



# Projektende

Zum Abschluss des Projekts wurde im Schritt Projektende eine digitale Präsentation erstellt. Mit dieser stellen wir unsere Ergebnisse des Projektes den anderen Teams vor.

## Aufgetretene Probleme

### 4.1.1 MATlab Anwendung

Während der Entwicklung der Anwendung gab es nur wenige kleine Probleme. Diese begrenzten sich meist auf Schwierigkeiten in der Umsetzung, wie beispielsweise die Formatierung der Ergebnisse in Tabellenform.

Eine weitere Schwierigkeit stellte die Einbindung des LBP Operators von Herrn Kumar dar, da der gegebene Quellcode nur wenig kommentiert und auch sonst nicht viele Informationen zur Programmlogik vorhanden waren.

### 4.1.2 VHDL-Umsetzung

* Aufgrund von Fehlern bei der Umsetzung des LBP-Moduls nicht synthetisierbar
* Potentielle Probleme mit dem Betriebssystem / Treiber falls zur Laufzeit das FPGA umprogrammiert wird (Wechsel zwischen VHDL / OpenCL Lösung)
* Auf eine weitere Entwicklung wurde verzichtet und ein reiner Hardware-basierter Ansatz auf Basis des ZedBoards (um Rekonfiguration bei der Präsentation zu vermeiden) mit UART-Schnittstelle weiterverfolgt

### 4.1.3 Open-CL-Umsetzung

* viele Gleitkomma Operationen
* round() auf GTX1080Ti ist fehlerhaft. Dadurch gab es sämtliche Speicherbereichsüberschreitungen. Ersetzt durch eine eigene Umsetzung.
* nicht Synthetisierbar
  + - * Gleitkommaarithmetik und trigonometrische Funktionen
      * Sehr große generierte "Schaltung"
      * Passt nicht auf das Cyclone V des DE1-SoC: 97% aller Logikzellen belegt
* Neuentwicklung des Kernels, Reduktion auf das Wesentliche

### 

## Resümee

* Mit OpenCL ist ein Ergebnis deutlich schneller zu erzielen
* OpenCL heißt nicht gleich OpenCL. Für die Hardwaresynthese müssen viele Details beachtet werden wie z.B Vermeidung von Gleitkommaoperationen.
* Die Ergebnisse der OpenCL-Lösung, ausgeführt auf der Grafikkarte und dem FPGA, unterscheiden sich enorm trotz gleichem Quellcode.
* Die OpenCL-Lösung ist auf dem DE1-SoC sehr langsam
* Die VHDL-Lösung ist nicht lauffähig
* Die VHDL-Lösung bereitet einen sehr viel höheren Entwicklungsaufwand

## 

|  |  |  |
| --- | --- | --- |
| **Änderungshistorie** | | |
| Version | Art der Änderung | Datum |
| 0 | Erstausgabe | 23.08.2017 |
|  |  |  |

Anhänge:

Lastenheft

Pflichtenheft

Arbeitspakete

div. Unterlagen