

# Entwurfsmethodik für integrierte Mixed-Signal Bildverarbeitungssysteme

Jens Döge, Peter Reichel, Olaf Enge-Rosenblatt

jens.doege@eas.iis.fraunhofer.de

Fraunhofer-Institut für Integrierte Schaltungen IIS, Institutsteil Entwurfsautomatisierung EAS,  
Zeunerstr. 38, 01069 Dresden, Deutschland

**Zusammenfassung** Die Entwicklung Kamera-basierter Messsysteme ist ein komplexer und aufwändiger Prozess, bei dem neben algorithmischen Fragen vor Allem der Einfluss von Nicht-Idealitäten eine wichtige Rolle spielt. Es wird eine Methodik vorgestellt, bei der durch den Einsatz moderner Modellierungsverfahren ein zielgerichteter Entwurf mit wenigen Iterationen erreicht wird. Die Validierung des Gesamtsystems wird dabei durch die Verwendung der entstehenden Modelle als „Goldene Referenz“ unterstützt.

## 1 Einführung

Am Fraunhofer Institut für Integrierte Schaltungen IIS werden im Kundenauftrag optische Messsysteme entwickelt. Der Aufgabenumfang umfasst schnelle Systeme z.B. für den Einsatz in Laser-Lichtschnitwanwendungen oder Systeme für die Detektion beweglicher Objekte in natürlichen Szenen (Präsenzdetektion). Bei der effizienten Entwicklung der eingebetteten Algorithmen müssen unter Anderem folgende Fragen geklärt werden:

- Welches optische Bildaufnahmeprinzip (CCD *multi-tap*, CMOS) eignet sich optimal für die vorliegende Aufgabenstellung?
- Welcher konkrete Bildsensor (Pixelgröße, Dynamikumfang, Auflösung, effektive Bitbreite - ENOB) erfüllt die Anforderungen?
- Wie gut sind die damit erzielbaren Eigenschaften (Dynamik, Messgenauigkeit, Empfindlichkeit)?
- Welche Randbedingungen sind zur Erzielung der Ergebnisse erforderlich (Belichtungszeit, Beleuchtungsverhältnisse, Oberflächeneigenschaften der Probe)?
- Welche Algorithmen eignen sich für die konkrete Implementierung und wie sind sie zu parametrieren (z.B. Bitbreite)?
- Wie lassen sich die ausgewählten Algorithmen partitionieren und ist evtl. eine Implementierung auf einem Bildsensor *System on Chip* (SoC) für die konkrete Aufgabenstellung sinnvoll?

Häufig sind die Anforderungen derart, dass der Einsatz kommerziell verfügbarer Kamerasysteme und Bildverarbeitungsbibliotheken zu sehr guten Ergebnissen führt. Auch die Wiederverwendung bereits bestehender Lösungen ist aus Kostensicht häufig gängige Praxis.

Nicht selten sind jedoch die Spezifikationen nicht einfach erfüllbar oder es stellen sich bei der Inbetriebnahme der Bildverarbeitungssysteme Schwierigkeiten ein, die nicht aus den Datenblättern für Bildsensor oder Kamera ableitbar waren, wie z.B.

- Probleme mit *Missing Codes*,
- Unterschiede zwischen der Farb- und der Monochrom-Version eines Sensors bzw. einer Kamera,
- optisches oder elektrisches Übersprechen im Bildfeld,
- Nichtlinearitäten oder Abweichungen des Sättigungsverhaltens einzelner Pixel,
- pixel- oder spaltenweise Unterschiede z.B. aufgrund des Einschwingverhaltens des Auslesepfads oder
- unerwartete Anforderungen an die Dynamik aufgrund des Messaufbaus.

In solchen Fällen wird es erforderlich, die Aspekte

- Umgebungsbedingungen,
- Anforderungen an das zu entwickelnde Messsystem,
- Nicht-Idealitäten des Bildaufnahmesystems und
- algorithmische Umsetzung der Aufgabenstellung

in einem geeigneten Entwurfsfluss zur Optimierung der Lösung des Anwendungsproblems zu verbinden. Die zugrunde liegende Methodik soll im Folgenden verdeutlicht werden.

## 2 Entwurfsfluss

### 2.1 Implementierung der Bildverarbeitungsalgorithmen

Als erster Schritt hin zu einem eingebetteten Bildverarbeitungssystem wird ein algorithmischer Prototyp in einer Skript- oder einer Hochsprache realisiert. Dabei kann sich der Entwickler unterschiedlicher Bildverarbeitungsbibliotheken wie z.B. der *Image Processing Toolbox* (MATLAB) [2] oder der *Python Imaging Library* [3] bedienen. Als Stimuli können synthetisch generierte Testdatensätze bzw. bei Vorliegen geeigneter Hardware / Software-Schnittstellen auch echte Bild- bzw. Videodaten verarbeitet werden. Die Ausgabe erfolgt über geeignete Schnittstellen des Hostsystems.

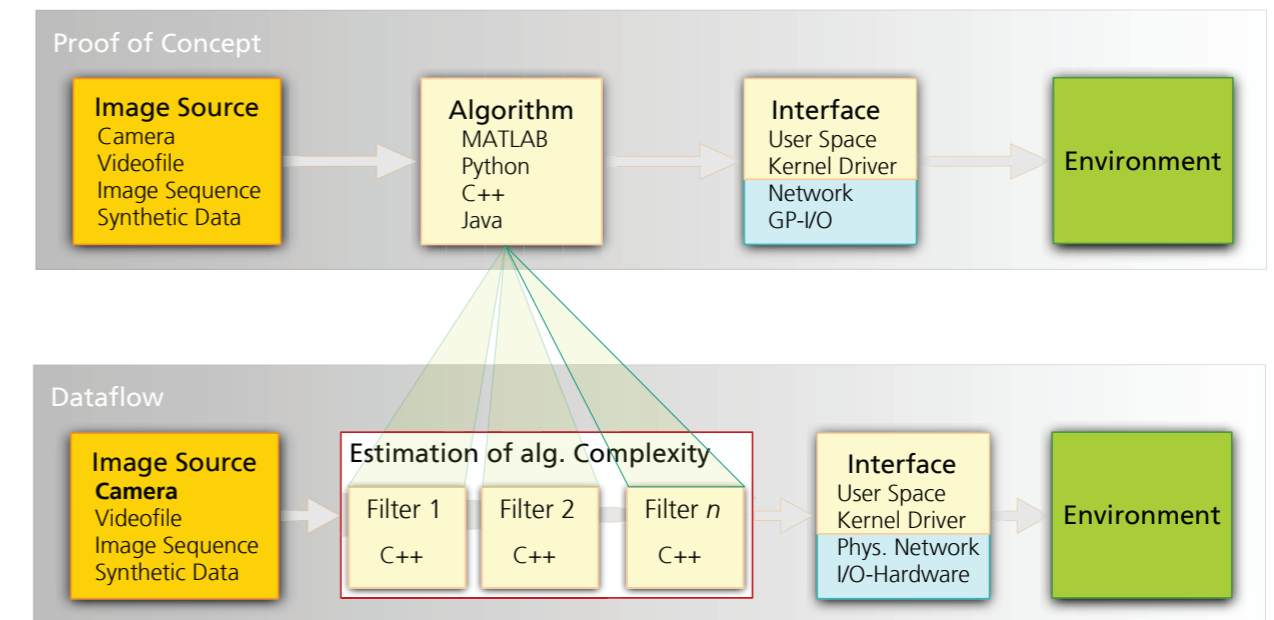


Abbildung 1. Realisierung des Datenflussmodells aus einem vorliegenden Algorithmus

Sind wesentliche algorithmische Fragen geklärt, erfolgt im nächsten Schritt eine Partitionierung des Algorithmus (s. Abbildung 1) mit dem Ziel, die einzelnen Verarbeitungsschritte zu konkretisieren und deren Aufwand im Hinblick auf den erwarteten Ressourcenbedarf abhängig vom notwendigen Datendurchsatz abzuschätzen. Gegebenenfalls sind dabei auch durch die Abbildung auf Hardware motivierte Anpassungen am Algorithmus notwendig.

Zur System-Partitionierung und Abbildung eines komplexen Algorithmus auf einen Datenfluss wurde am Fraunhofer IIS/EAS ein Filter-Framework auf der Basis der C++ Template-Klasse *CImg<T>* [1] entwickelt. Die Grundlage des Frameworks bilden abgeleitete Klassen, die durch ihre einheitliche Schnittstelle beliebig austauschbar sind. Bei der algorithmischen Umsetzung können bestehende Filter wiederverwendet, oder Neue definiert werden. Sie können sowohl sequentiell als auch hierarchisch auf die Bilddaten angewendet werden. Grundsätzlich wird zwischen Eingabe-, Verarbeitungs- und Ausgabefiltern unterschieden, wodurch der Einzug von Bildsequenzen unabhängig vom tatsächlichen Gerät ist, d.h. sowohl synthetische als auch aufgezeichnete oder Live-Daten einer Kamera verarbeitet werden können. Beispielanwendungen für dieses auf Datendurchsatz hin optimierte Framework sind Referenz-Implementierungen eines Laser-Lichtschnitt-Systems sowie eines Systems zur Erkennung bewegter Objekte auf der Basis von Strukturinformationen.

### 2.2 Realisierung einer ausführbaren Spezifikation

Die durch das Datenflussmodell gewonnenen Erkenntnisse im Hinblick auf Anforderungen und Einschränkungen werden genutzt, um ein erstes Modell unter Berücksichtigung des ungefähren Zeitverhaltens zu erzeugen, das für die weiteren Entwurfsschritte als ausführbare Spezifikation (Goldene Referenz, s. Abbildung 2) verwendet wird.

Neben den durch die Anwendung bzw. den Algorithmus eingebrachten Anforderungen fließen in das Modell auch über vorhandene Bibliothekskomponenten definierte Schnittstellen zur Ein- und Ausgabe sowie zur Hardware-Abstraktion ein. Zusätzlich werden wesentliche Eigenschaften der Übertragungsfunktion der Sensorik berücksichtigt, also z.B. das Rauschverhalten, der Dynamikumfang oder diverse Nicht-Idealitäten wie Leckströme oder Sättigungsverhalten. Als Plattform dient hierbei die auf die Bedürfnisse der Systemspezifikation zugeschnittene C++ Bibliothek *SystemC*

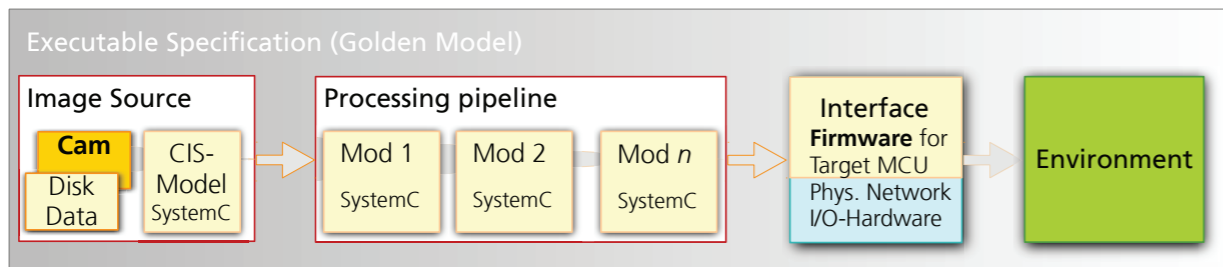


Abbildung 2. Umsetzung in eine Bildverarbeitungs-Pipeline

[4] bzw. zur Beschreibung analoger Zusammenhänge deren am Fraunhofer IIS/EAS entwickelte Erweiterung *SystemC AMS* [6].

Die weitere Optimierung des Algorithmus wird auf der Grundlage des System-Modells durchgeführt. Hard- und Softwareblöcke können parallel entwickelt werden, da der Entwurf der Firmware nicht das Vorhandensein einer Hardware-Plattform voraussetzt, sondern auf der Grundlage des System-Modell erfolgen kann.

### 2.3 Beispiel für die Extraktion und Modellierung ausgewählter Nicht-Idealitäten

Die Modelle der Nicht-Idealitäten können aus verschiedenen Quellen bezogen werden. Erfolgt beispielsweise der Entwurf des Bildsensors im eigenen Haus, so kann das tatsächliche Verhalten der Pixelzellen, der Blöcke zur analogen oder *mixed-signal* Informationsverarbeitung und der Analog/Digital-Umsetzung für die Modellierung herangezogen werden. In Abbildung 3 ist dieser Sachverhalt dargestellt.

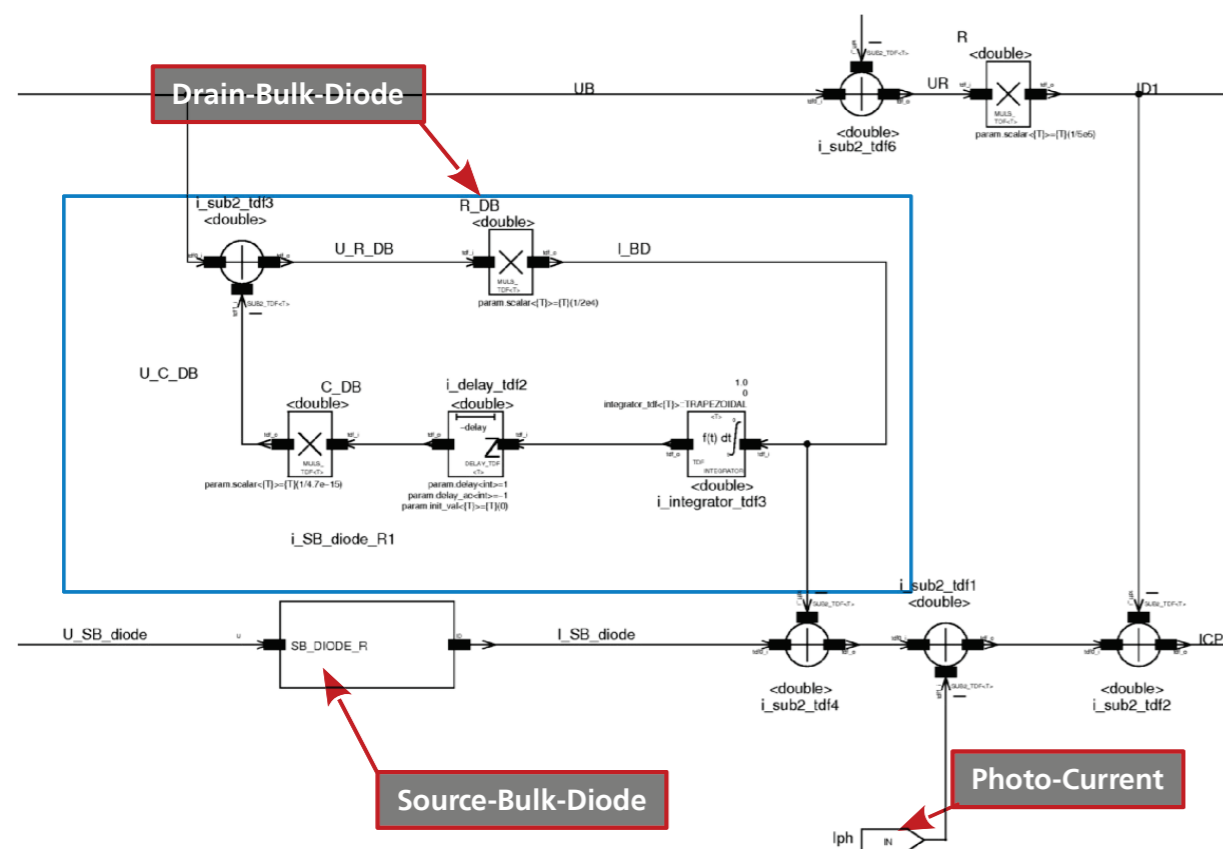


Abbildung 3. Physikalisches Modell einer Pixelzelle [8]

Ausgehend von der elektrischen Schaltung und dem Verhalten des optischen Stacks wurde ein *SystemC AMS* Modell für eine Pixelzelle entwickelt. Es beinhaltet verschiedene Elemente wie Dioden und Kapazitäten, für die unter

Berücksichtigung der konkreten Genauigkeitsanforderungen explizite Gleichungen, linearisierte Beschreibungen oder Tabellenmodelle verwendet werden können. Beispielhaft ist in Abbildung 4 der Quellcode für eine der beiden Dioden dargestellt.

```

51//-----//
52
53#ifndef DONT_INCLUDE_IMPLEMENTATION
54
55#include <cmath>
56
57//-----//
58SB_diode R::states& SB_diode_R::create_states() { return *(new states); }
59//-----//
60
61
62//////////////////////////////////////
63// method processing //
64//////////////////////////////////////
65void SB_diode_R::processing()
66{
67    double delta_UD;
68    double I_help;
69    double f;
70    double fprime;
71    double Usum = this->U.read();
72
73    do // newton
74    {
75        I_help = this->IS * exp(this->UD / (this->UT * this->n));
76        f = (I_help - this->IS)*this->R + this->UD - Usum;
77        fprime = I_help / this->UT * this->R + 1;
78        delta_UD = f/fprime;
79        this->UD -= delta_UD;
80    } while( abs(delta_UD) > 1e-6);
81
82    this->I_diode = (Usum - this->UD)/this->R;
83    ID.write(this->I_diode );
84}
85
86
87
88#endif //ifndef DONT_INCLUDE_IMPLEMENTATION
89
90} //end namespace Chip5_AMS_namespace
91
92//clear temporary defines
93#undef DONT_INCLUDE_IMPLEMENTATION
94

```

Abbildung 4. SystemC AMS Beschreibung einer Komponente aus Abbildung 3 [8]

Meist steht der genaue interne Aufbau eines Bildsensors jedoch nicht zur Verfügung. Einige zur Verhaltensmodellierung erforderlichen Parameter müssen unter diesen Umständen anders, z.B. durch optische bzw. elektrische Messungen ermittelt werden.

Mit dem Standard EMVA1288 [7] hat die *European Machine Vision Association* eine Vorschrift entwickelt, mittels der die Eigenschaften von Bildsensoren und Kameras unter zeitlich konstanter und homogener Beleuchtung erfasst werden können. Bildgebende Systeme mit einer linearen Kennlinie können somit charakterisiert und verglichen werden.

Erfordert die Aufgabenstellung jedoch konkrete Einblicke in das Verhalten der Pixelzelle wie z.B.

- die Wirkung des optischen oder elektrischen Übersprechens zwischen einzelnen Pixeln,
- die Verteilung der lokalen Empfindlichkeit innerhalb der strahlungsempfindlichen Anordnung oder
- Ursachen für undokumentierte Effekte,

ist es unter Umständen erforderlich, ausgewählte Pixel ortsaufgelöst *subpixel* genau zu beleuchten und für einen bezüglich der Strahlform, der optischen Leistung und der Wellenlänge bekannten optischen Stimulus das Ausgangssignal des Sensors oder der Kamera zu erfassen.

Eine beispielhafte Untersuchung ist in Abbildung 5 angegeben. Dabei wurde ein Bildsensor auf der Grundlage von schachbrettartig angeordneten Pixelzellen mit einer Kantenlänge von  $4\mu\text{m} \times 4\mu\text{m}$  durch einen fokussierten Laserstrahl

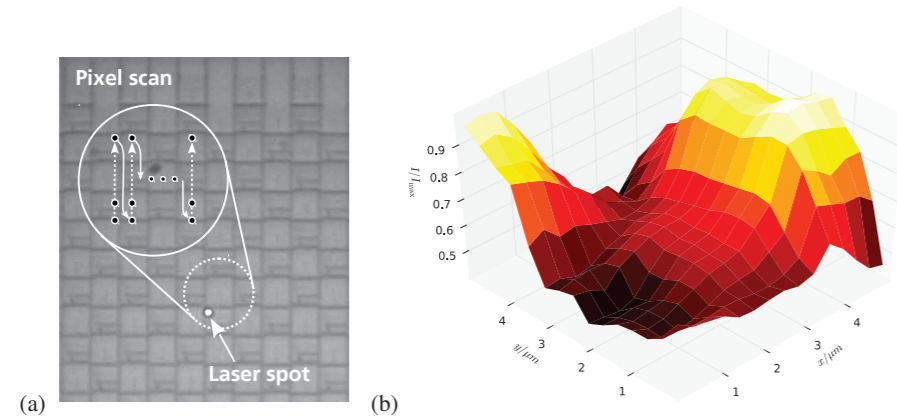


Abbildung 5. Scannen der Bildsensor-Matrix mit einem Laser-Spot (a) und relative Darstellung der lokalen Empfindlichkeiten (b)

( $\lambda=500$  nm) beleuchtet. Die Probe wurde im Raster von  $0.5 \mu\text{m}$  in X- und Y-Richtung verfahren und der photogenerierte Strom für jede Position gemessen. Anhand der Messwerte (b) konnte das mathematische Modell der Pixelzelle experimentell validiert werden. [5]

## 2.4 Intelligentes Kamerasystem

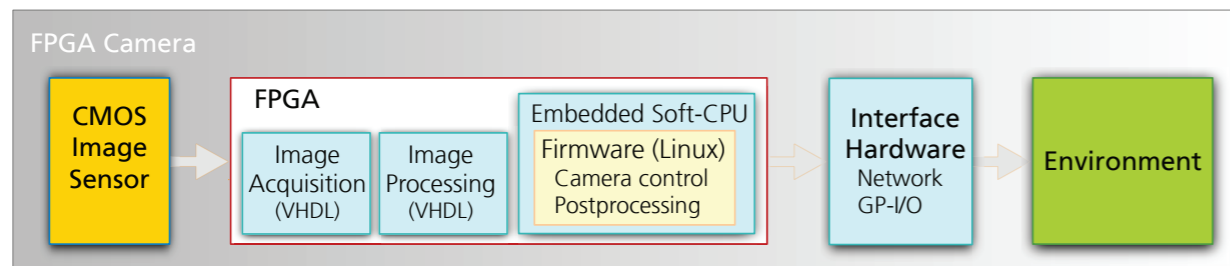


Abbildung 6. Überführung in ein intelligentes Kamerasystem

Bei der aktuell verwendeten Entwicklungsplattform (s. Abbildung 6) liegt der Fokus auf der sensornahen, hochparallelen Bildverarbeitung, weshalb ein relativ großer FPGA mit direkter Speicheranbindung verwendet, jedoch auf eine eigenständige CPU oder einen DSP verzichtet wird. Nach Abschluss der Evaluierung im Systemmodell erfolgt daher die eigentliche Implementierung des Kamerasystems, indem die wesentlichen Teile der Verarbeitung in VHDL umgesetzt und somit für eine konkrete FPGA-Realisierung bzw. die Integration in ein Bildsensor-SoC vorbereitet werden. Neben der Bereitstellung zahlreicher Bibliothekskomponenten unter Anderem zur Einbettung in bestehende Infrastruktur (z.B. Linux-Betriebssystem) wird der Entwurf vor Allem durch die Möglichkeit zur Validierung durch Vergleich des Verhaltens mit dem als Goldene Referenz verwendeten Systemmodell erleichtert.

## 3 Zusammenfassung

Anhand der verschiedenen Einzelschritte wurde ein stukturierter und anwendungsgetriebener Entwurfsfluss für Bildverarbeitungssysteme dargestellt. Dabei wird das Bildverarbeitungssystem als Ganzes unter Berücksichtigung von Umwelteinflüssen und Beschränkungen betrachtet sowie auf unterschiedlichen Abstraktionsebenen modelliert. Durch diese Vorgehensweise ist es möglich, die Modelle der einzelnen Komponenten schrittweise zu verfeinern und den Entwurfsraum zielgerichtet zu explorieren. Auf der Grundlage einer goldenen Referenz in Form von SystemC (AMS)-Modellen können Software und Hardware eines Bildverarbeitungssystem effizient parallel entwickelt werden.

## Literatur

1. *The Cimg Library*. <http://cimg.sourceforge.net>, 22 Oktober 2012
2. *Image Processing Toolbox*. <http://www.mathworks.de/products/image>, 22 Oktober 2012

3. *Python Imaging Library (PIL)*. <http://www.pythonware.com/products/pil>, 22 Oktober 2012
4. BLACK, D. C. et. a.: *SystemC: From the Ground Up. 2*. Springer Science+Business Media, LLC, 2010
5. BLANCO-FILGUEIRA, B. ; LÓPEZ, P. ; DÖGE, J. ; ROLDÁN, Juan B.: Evidence of the lateral collection significance in small CMOS photodiodes. In: *Proceedings of the IEEE International Symposium on Circuits and Systems*, 2012
6. EINWICH, Karsten: Introduction to the SystemC AMS extension standard. In: *Proceedings of the 14th IEEE International Symposium on Design and Diagnostics of Electronic Circuits and System, DDECS 2011*, 2011
7. EUROPEAN MACHINE VISION ASSOCIATION: *Standard for Measurement and Presentation of Specifications for Machine Vision Sensors and Cameras*. <http://www.emva.org/cms/index.php?idcat=26>, 22 Oktober 2012
8. PRUSSEIT, Steffen: *Systemmodellierung eines Hochgeschwindigkeitsbildsensors mit integrierter Bildverarbeitung in SystemC AMS*, Technische Universität Dresden, Diplomarbeit, 15 Juni 2011