Modellierung und Codegenerierung von SOC-Beschleunigermodulen am Beispiel eines Kalman-Filters

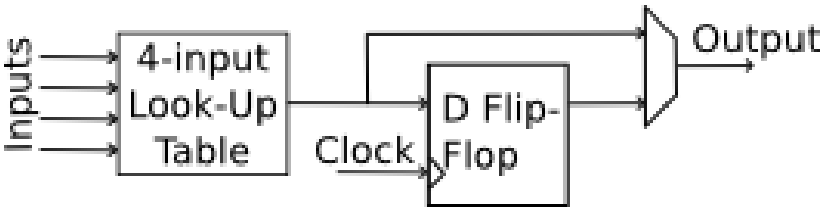
Erik Andresen

26.11.2010

Inhalt

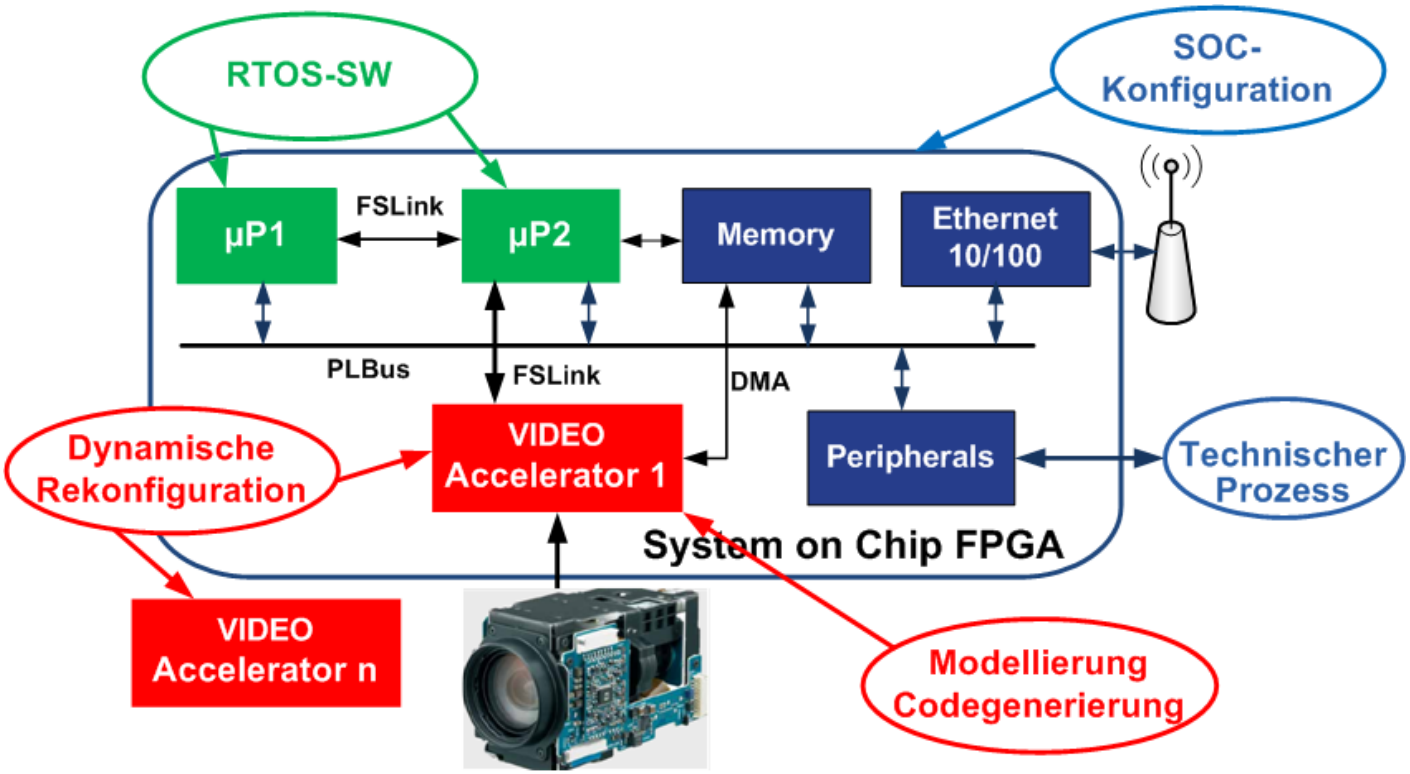
- Beschleunigung von Algorithmen durch Auslagerung in Hardware durch DSP oder **FPGA**.
- Verwendung z.B. zur Bildverarbeitung. Beschleunigung durch Parallelverarbeitung.
- Entlastung des Prozessors.

FPGA - Field Programmable Gate Array

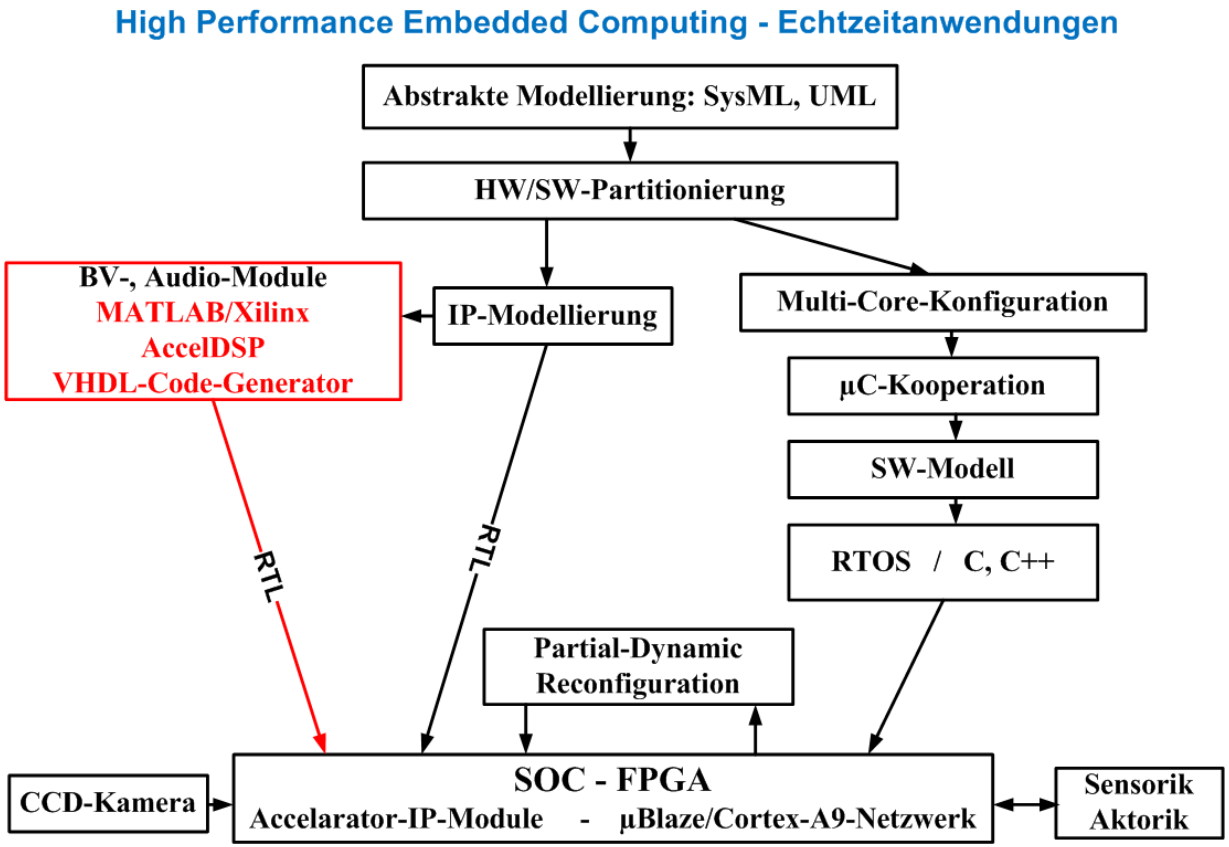


- Lookup-Tabellen bilden alle Logik-Gatter ab.
- Flip-Flops zur Zustandsspeicherung.
- Mehrere Hunderttausend Logikblöcke.
- Durch Hintereinanderschaltung von Logikblöcken realisierung komplexer dig. Hardware (DSP, CPU)
- Schnittstellen zur Aussenwelt.
- Zusätzlich BRAMs, DSP-Elemente, HW-Prozessoren...
- Modellierung bit Hardware-Bechreibungssprachen (HDLs)

FPGA als System on Chip



SOC Roadmap


High Performance Embedded Computing - Echtzeitanwendungen
Abstrakte Modellierung: SysML, UML
HW/SW-Partitionierung
BV-, Audio-Module
MATLAB/Xilinx
AccelDSP
VHDL-Code-Generator
IP-Modellierung
Multi-Core-Konfiguration
µC-Kooperation
SW-Modell
RTOS / C, C++
RTL
RTL
Partial-Dynamic
Reconfiguration
CCD-Kamera
SOC - FPGA
Accelarator-IP-Module - µBlaze/Cortex-A9-Netzwerk
Sensorik
Aktorik

Matlab als Abstraktionslayer

1. Modellierung und Implementierung des Algorithmus in Matlab und AccelDSP .
2. Konvertierung der Fließkommazahl-Arithmetik in Festkomma-Arithmetik mit AccelChip.
3. Vergleich der Modellierungs-Varianten und ggf. Anpassen der Wortbreiten.
4. Optimierung und Generierung in einer Hardware-Beschreibungssprache

- Rekursiver Schätzalgorithmus, 1960 durch den Mathematiker R.E. Kalman vorgestellt.
- Eingesetzt im Apollo-Programm der NASA
- In der Bildverarbeitung zur Objektverfolgung eingesetzt.
- Messungen sind Fehlerbehaftet.
- Messungen mit großer Genauigkeit sollen stärker berücksichtig werden.
- Grundlagen des beobachteten Systems bekannt.
- Messverfälschungen sind bekannt.
- Rekursiv.

Das Kalman-Filter Liefert optimalen Schätzwert für den Zustand x eines Systems

Zustandsgleichung

$\mathbf{x_{k+1}} = \mathbf{A_k} * \mathbf{x_k} + \mathbf{B_k} * \mathbf{u_k} + \mathbf{w_k}$

$\mathbf{x_k}$ Zustand

$\mathbf{A_k}$ Systemmatrix

$\mathbf{B_k}$ Eingabematrix

$\mathbf{u_k}$ Steuer-Vektor

$\mathbf{w_k}$ Systemrauschen

Messgleichung

$y_k = H_k * x_k + v_k$

$\mathbf{y_k}$ Messung

$\mathbf{H_k}$ Messmatrix

$\mathbf{v_k}$ Messrauschen

$\mathbf{e_k^-} = \mathbf{x_k} - \hat{\mathbf{x}}_\mathbf{k}^-$, $\mathbf{P_k^-} = \mathbf{E}(\mathbf{e_k^-} * \mathbf{e_k^{-\tau}})$

$\mathbf{e_k^+} = \mathbf{x_k} - \hat{\mathbf{x}}_\mathbf{k}^+$, $\mathbf{P_k^+} = \mathbf{E}(\mathbf{e_k^+} * \mathbf{e_k^{+\tau}})$

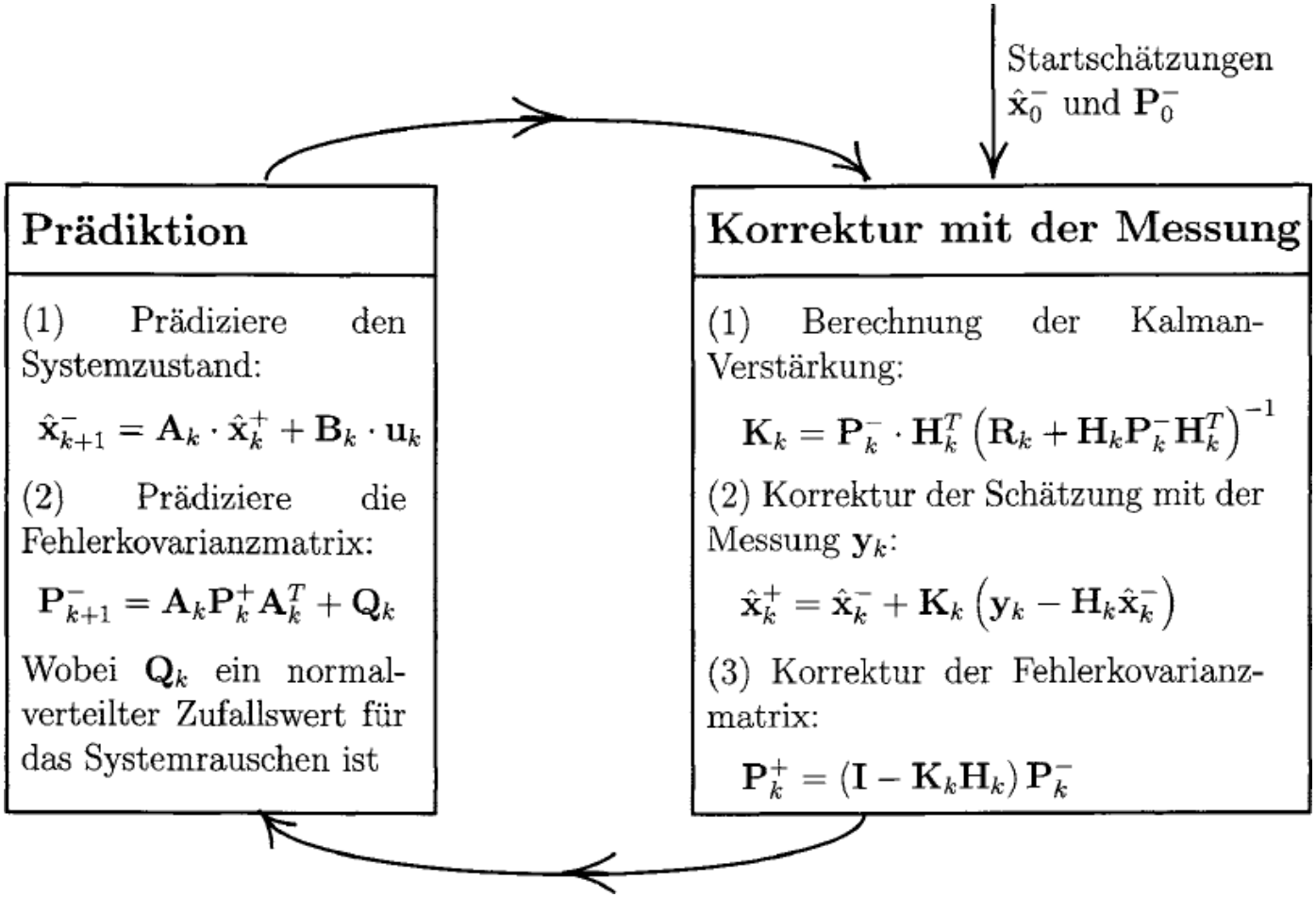
$\hat{\mathbf{x}}_\mathbf{k}^-$, $\hat{\mathbf{x}}_\mathbf{k}^+$ Schätzwert „a priori“ und „a posteriori“

$\mathbf{P_k^-}$, $\mathbf{P_k^+}$ Schätzfehlerkovarianz „a priori“ „a posteriori“

K_k Kalman-Verstärkung
R_k Varianz des Messfehlers

$$K_k = \frac{P_k^- * H_k^T}{R_k + H_k P_k^- H_k^T}$$

- Wenn Varianz des Messfehlers R_k steigt, wird der Messung weniger Vertraut, der Schätzung jedoch mehr.
- Wenn die Schätzfehlerkovarianz P_k^- zunimmt, wird der Messung mehr vertraut, der Schätzung jedoch weniger.



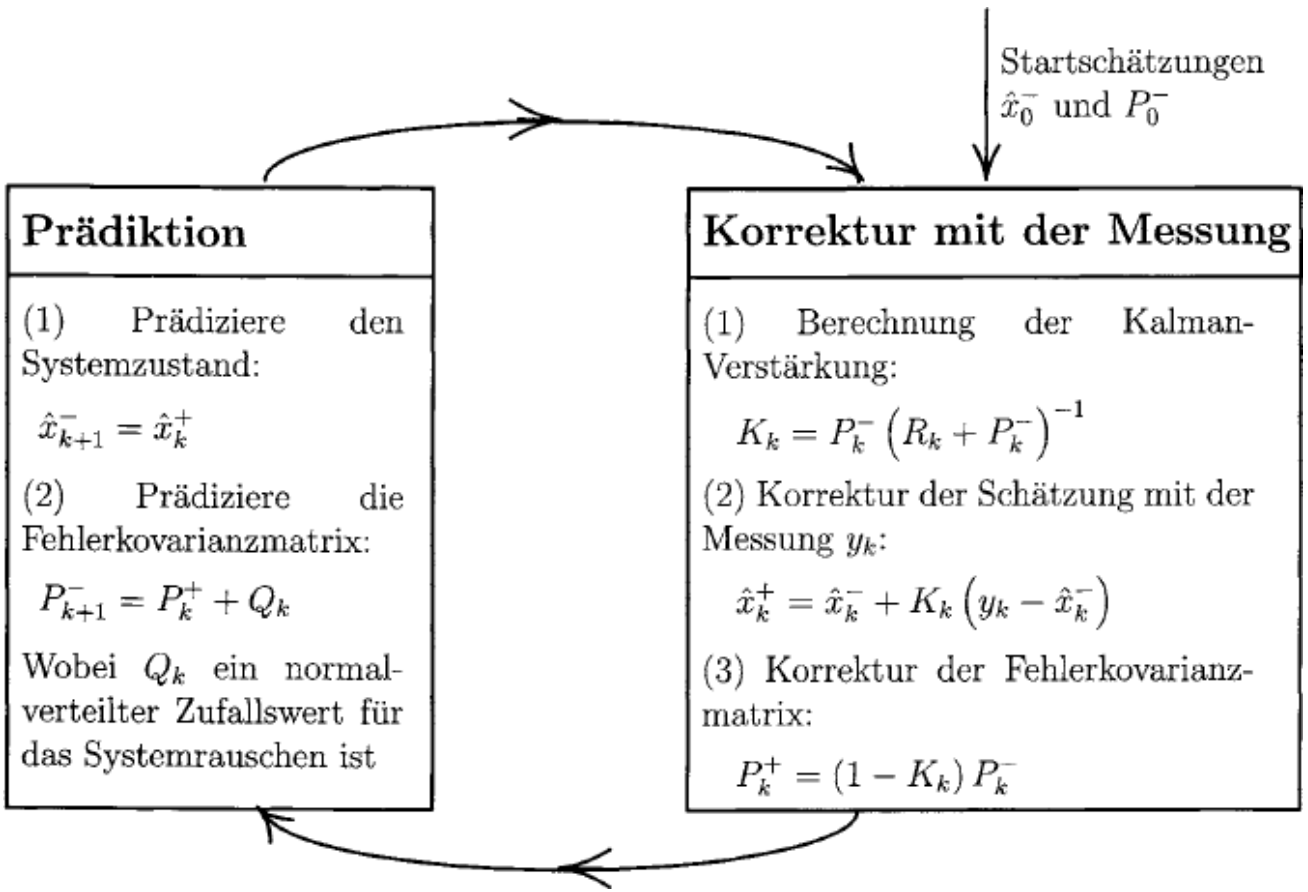
I Einheitsmatrix

Q_k Systemrauschen

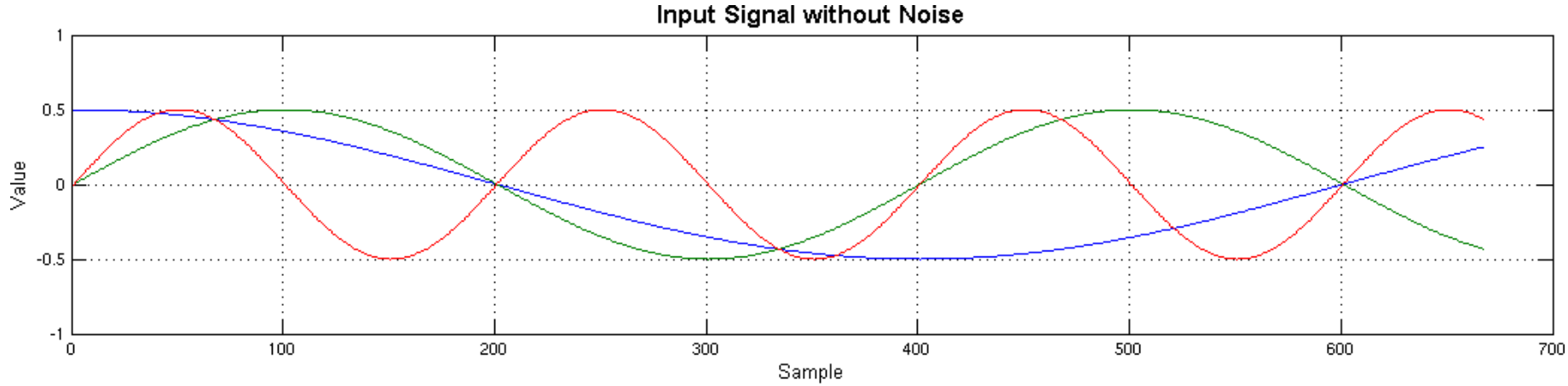
- Beispiel aus Xilinx Xcell Journal #53: Verrauschtes Signal
- Reales System: Erzeugung von Sinus und Cosinus Werten. (Ausgangswert)
- Künstlich verfälscht durch Zufallswerte.
- Ziel: Annährung an die Ausgangswerte.
- $A_k = 1$ Zustand ist zeitlich konstant.
- $u_k = 0$ Keine äußeren Einflüsse.
- $H_k = 1$ System ist direkt messbar.

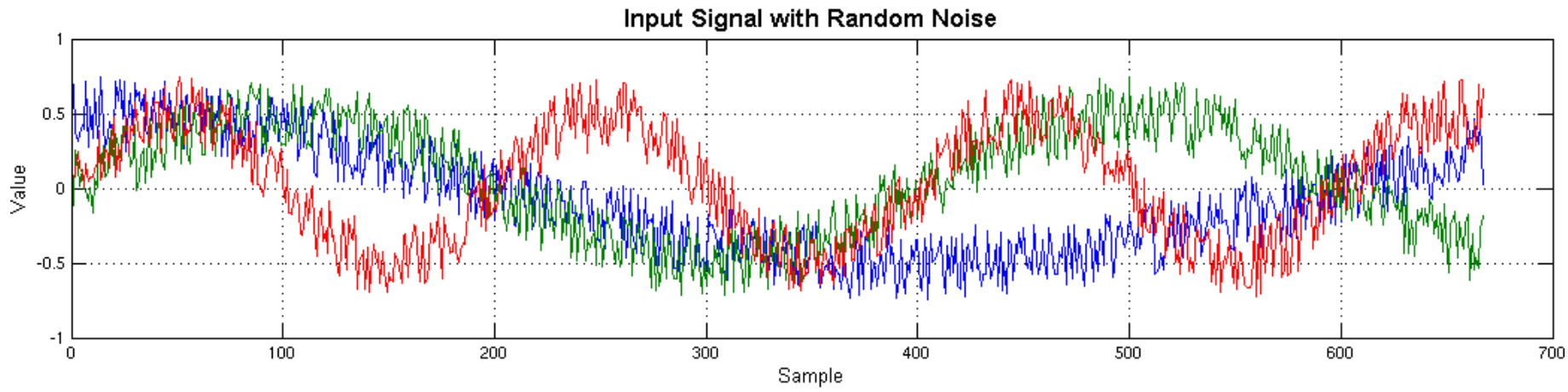
```
0001 % Real Input Signal
0002 A = [
0003     cos(0:pi/400:(5/3)*pi)/2;
0004     sin(0:pi/200:(10/3)*pi)/2;
0005     sin(0:pi/100:(20/3)*pi)/2
0006     ]';
0007
0008 % Add some noise
0009 A_in = A + 0.5*(rand(size(A)) - 0.5);
0010
0011 % Run Kalman
0012 for i=1:size(A,1),
0013     [S(i,:)] = simple_kalman(A_in(i,:));
0014 end
0015
0016 % Plot
0017 subplot(3,1,1);
0018 plot(A);
0019 axis([0 700 -1 1])
0020 title('Input Signal without Noise')
0021
0022 subplot(3,1,2);
0023 plot(A_in);
0024 axis([0 700 -1 1])
0025 title('Input Signal with Random Noise')
0026
0027 subplot(3,1,3);
0028 plot(S);
0029 axis([0 700 -1 1])
0030 title('Filtered Output Signal')
```

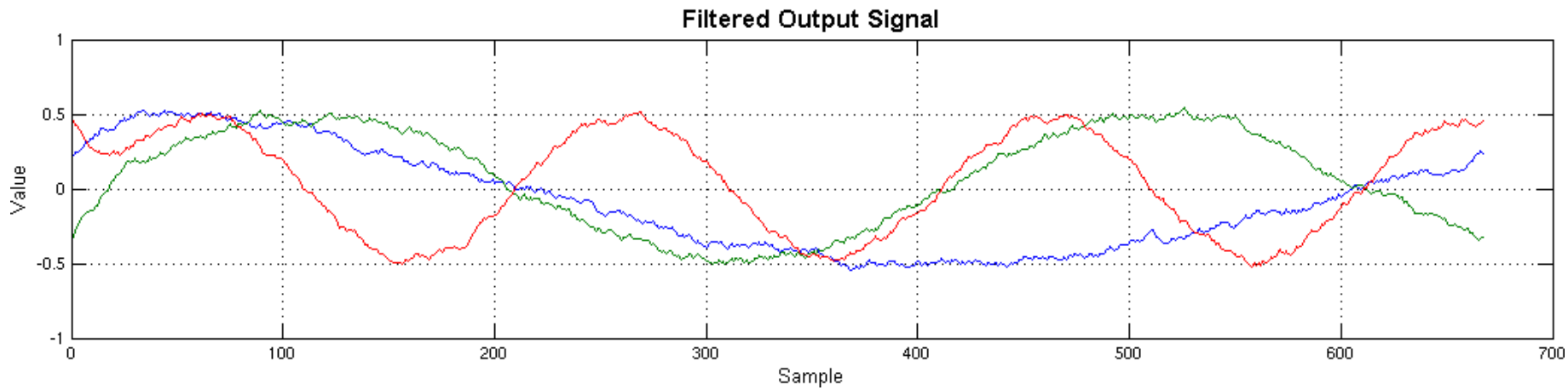
Vereinfachter Algorithmus

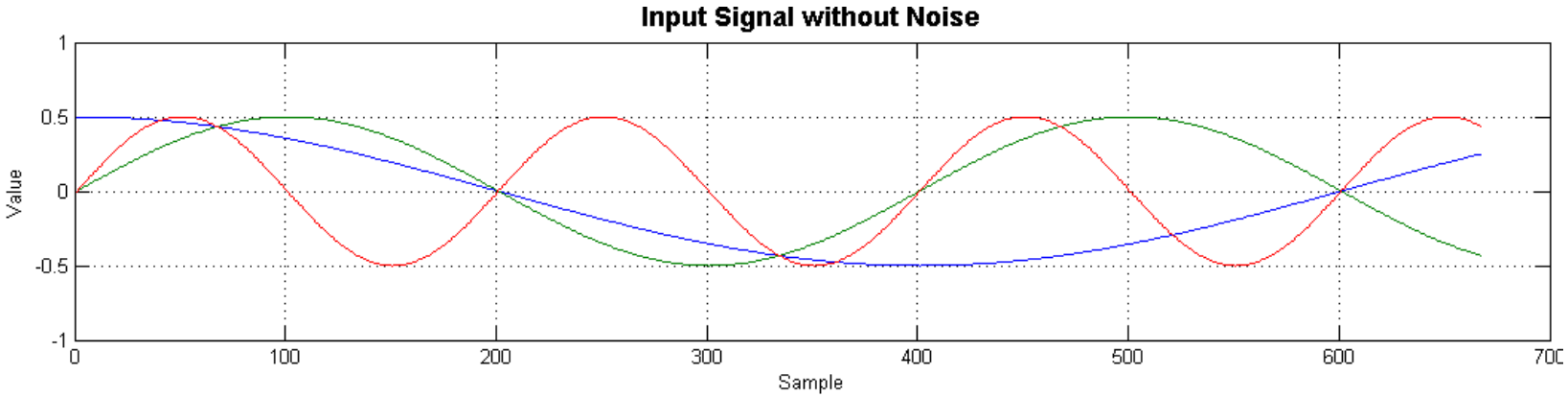


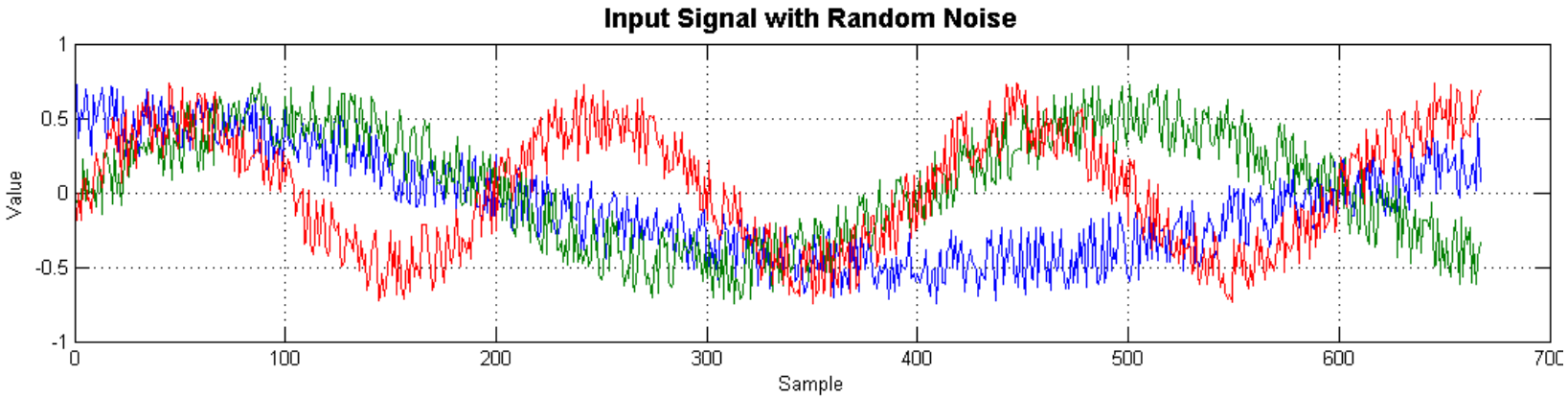
```
0001 function [S] = simple_kalman(y)
0002     persistent x_est P_est
0003
0004     if isempty(P_est)
0005         P_est = eye(size(y,2))*8; % Schätzfehlerkovarianz = 8
0006         x_est = ones(size(y,2),1)/2; % Systemzustand = 0.5
0007     end;
0008
0009     I = eye(size(y,2)); % Einheitsmatrix
0010     Q = eye(size(y,2)); % Systemrauschen = 1
0011     R = eye(size(y,2))*128; % Varianz des Messfehlers = 128
0012
0013     % correction step:
0014     % (1) Berechnung der Kalman-verstärkung
0015     K = P_est * inv(R + P_est);
0016     % (2) Korrektur der Schätzung mit der Messung
0017     x = x_est + K * (y' - x_est);
0018     % (3) Korrektur mit der Messung
0019     P = (I - K)*P_est;
0020
0021     % estimate step:
0022     % (1) Präzidiere den Systemzustand
0023     x_est = x;
0024     % (2) Präzidiere die Fehlerkovarianzmatrix
0025     P_est = P+Q;
0026
0027     % Return
0028     S = x';
```

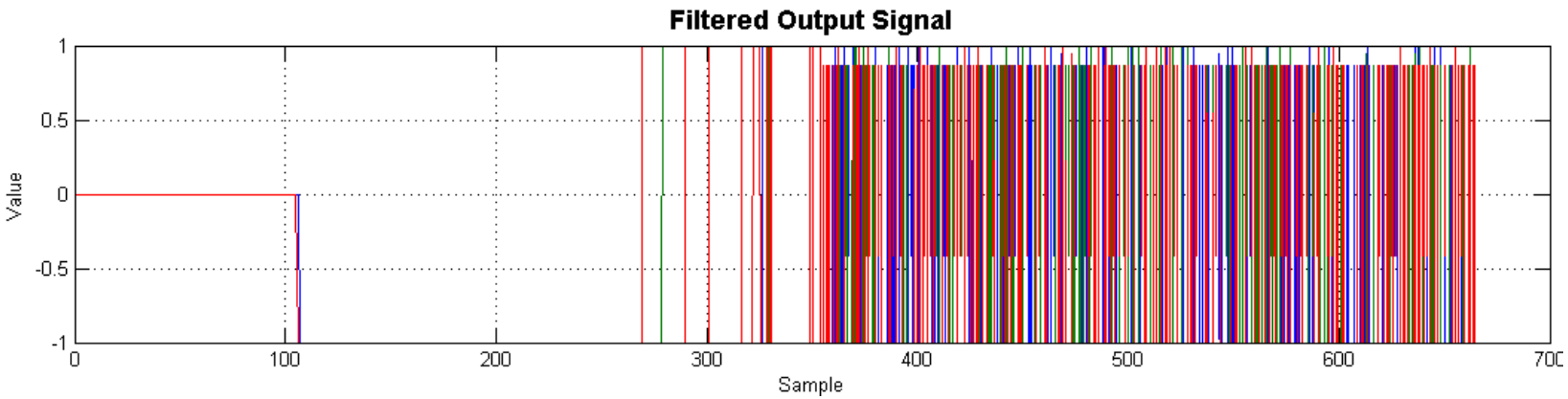

Input Signal without Noise
Value
Sample
1
0.5
0
-0.5
-1
0
100
200
300
400
500
600
700


Input Signal with Random Noise
Value
Sample
1
0.5
0
-0.5
-1
0
100
200
300
400
500
600
700


Filtered Output Signal
Value
Sample
1
0.5
0
-0.5
-1
0
100
200
300
400
500
600
700

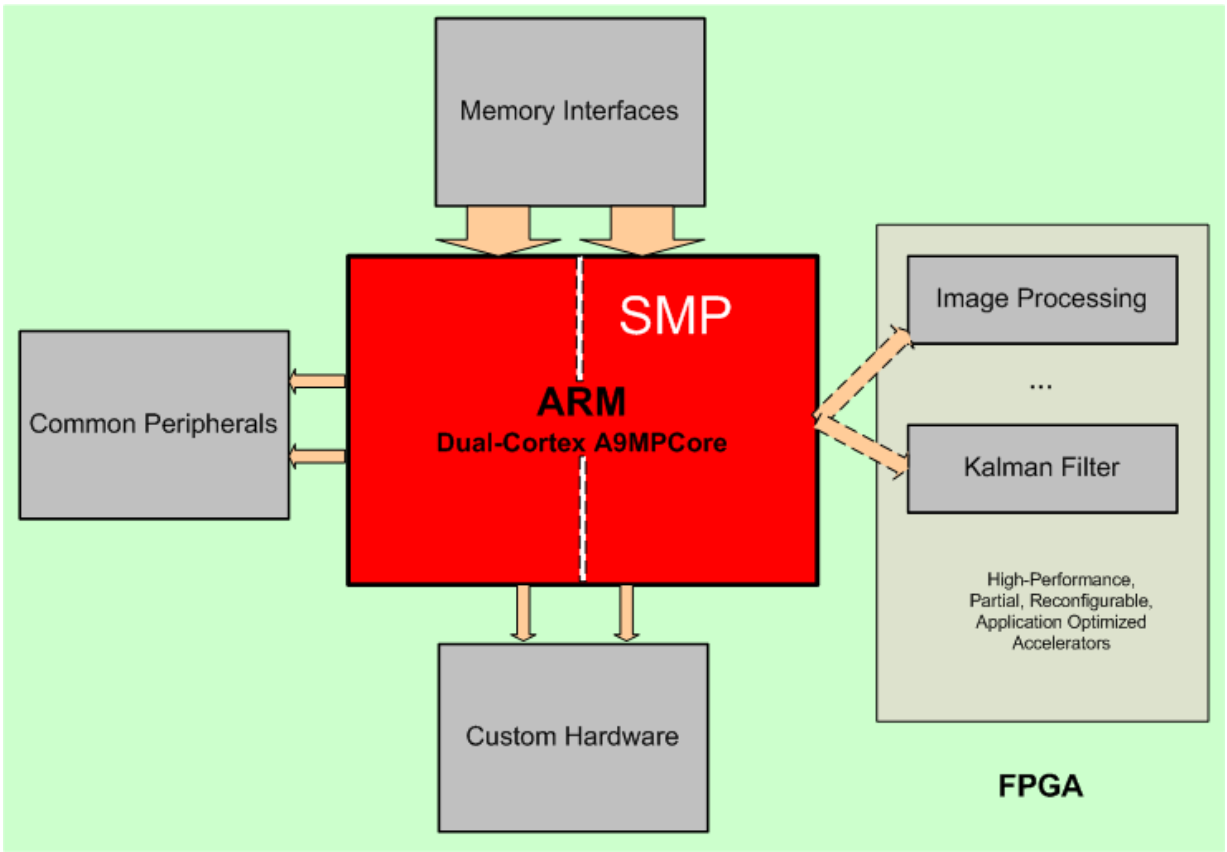

Input Signal without Noise
Value
Sample
1
0.5
0
-0.5
-1
0
100
200
300
400
500
600
700


Input Signal with Random Noise
Value
Sample
1
0.5
0
-0.5
-1
0
100
200
300
400
500
600
700


Filtered Output Signal
Value
Sample
1
0.5
0
-0.5
-1
0
100
200
300
400
500
600
700

- 42 Multiplizierer
- 70 Addierer
- 65 Subtrahierer
- 3*13 Eingänge für Matrix A
- 3 Steuerleitungen als Eingang
- 3*13 Ausgänge für Matrix S
- 1 Steuerleitung als Ausgang

Zukunft mit DualCore ARM CPU



Ausblick

- Modell vervollständigen und für Ganzzahlen anpassen.
- Parameter für Kalman-Filter, Objektverfolgung.
- Dual-Core Plattform.

Quellen

- Nischwitz, Fischer, Haberäcker: Masterkurs Computergrafik und Bildverarbeitung: Alles für Studium und Praxis. Verlag Vieweg+Teubner, 2. Auflage.
- Xilinx DSP Magazine #1 (2005). - `http://www.xilinx.com/publications/`
- Xilinx Xcell Journal #53 (2005). - `http://www.xilinx.com/publications/xcellonline/`
- Xilinx Xcell Journal #71 (2010). - `http://www.xilinx.com/publications/xcellonline/`