Chipimplementation einer zweidimensionalen Fouriertransformation für die Auswertung eines Sensor-Arrays

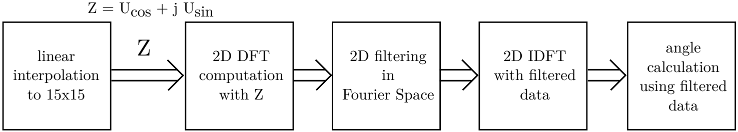
Bachelorkolloquium

Thomas Lattmann

30. April 2018

Inhaltsübersicht

Einleitung: Einordung im ISAR-Projekt



Quelle: Frequency_filtering_and_stray_field_compensation_using_2D-DFT_algorithm.pdf, K.-R. Riemschneider + T. Schüthe

Einleitung: Details zur Hardware

- 350 µm Prozess
- Array von Magnetsensoren
- Sensoren, Signalverarbeitung & Ausgabe des digitalen Nutzsignals auf einem ASIC

Grundlagen: Übersicht

- Interpretation von Dualzahlen
- Komplexe Multiplikation
- Matrixmultiplikation
- DFT
- 2D-DFT
- IDFT

Interpretation von Dualzahlen

Mögliche Arten sind:

- positive Ganzahldarstellung (a)
- Darstellung im Einerkomplement (b)
- Darstellung im Zweierkomplement (c)
- voreichenbehaftete Festkommazahlen (SQ) mit und ohne Vorkommaanteil (d)

1001011010100_2

$$4096 + 512 + 128 + 64 + 16 + 4 = 4820_{10} \tag{a}$$

$$-(512 + 128 + 64 + 16 + 4) = -724_{10} \tag{b}$$

$$-4096 + 512 + 128 + 64 + 16 + 4 = -3372_{10} \tag{c}$$

$$-4 + 0,5 + 0,125 + 0,062 + 0,015625 + 0,00390625 = -3.29296875_{10} \quad \text{in S2Q10 (d)}$$

Komplexe Multiplikation

Komplexe Multiplikation sind 4 einfache Multiplikationen und 2 Additionen.

$$\begin{aligned} e + jf &= (a + jb) \cdot (c + jd) \\ &= a \cdot c + j(a \cdot d) + j(b \cdot c) + j^2(b \cdot d) \\ &= a \cdot c - b \cdot d + j(a \cdot d + b \cdot c) \end{aligned}$$

Wenn einer der beiden Multiplikanden keinen Imaginärteil haben, reduziert sich das zu

$$\begin{aligned} e + jf &= a \cdot (c + jd) \\ &= a \cdot c + j(a \cdot d) \end{aligned}$$

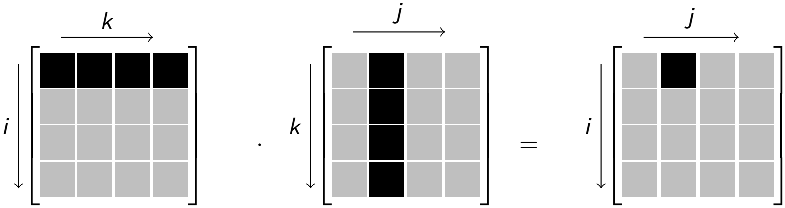


Abbildung: Veranschaulichung der Matrixmultiplikation.

Diskrete Fouriertransformation und ihre inverse

$$X^{*}[m] = \frac{1}{N} \cdot \sum_{n=0}^{N-1} x[n] \cdot e^{-\frac{j2\pi mn}{N}}$$

$$x[n] = \frac{1}{N} \sum_{n=0}^{N-1} X^{*}[m] \cdot e^{+\frac{j2\pi mn}{N}}$$

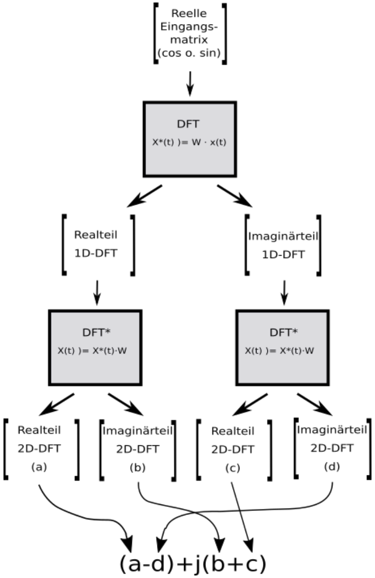
$$X^{*} = W \cdot x$$

$$X = W \cdot X^{*} \cdot W$$

Berechnungsarten der DFT und deren Aufwand

- optimierte Matrixmultiplikation mit reellen Eingangswerten
- optimierte Matrixmultiplikation mit komplexen Eingangswerten
- Fast Fouriertransformation
- variable Matrixmultiplikation

Reelle Eingangswerte



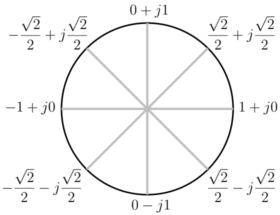
Analyse und Entwicklung: Übersicht

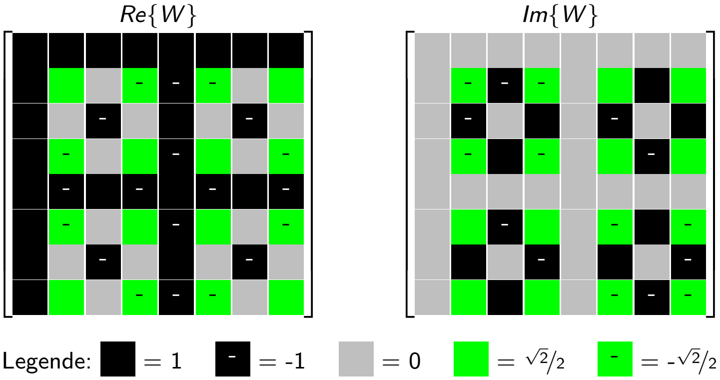
- Gegenüberstellung verschiedener Größen von Twiddlefaktormatrizen
- Optimieren der 8x8-DFT
- Konstantenmultiplikation
- Benötigte Takte
- Zustandsfolge
- Entwickeln der 2D-DFT auf Basis der 1D-DFT

Gegenüberstellung verschiedener Größen von DFT-Twiddlefaktormatrizen

N	8	9	12	15	16
N×N	64	81	144	225	256
trivial $\Re$	48	45	128	81	128
nicht triv. $\Re$	16	36	16	144	128
triv. $\Im$	48	21	96	45	128
nicht triv. $\Im$	16	60	48	180	128
$\sum$ triv.	96	66	224	126	256
$\sum$ nicht triv.	32	96	64	324	256
Anzahl verschiedener nicht trivialer Werte	1	7	1	13	3
Verhältnis $\sum$ trivial / $\sum$ nicht trivial	3	0,6875	3,5	0,3889	1

Optimierung der 8x8-DFT

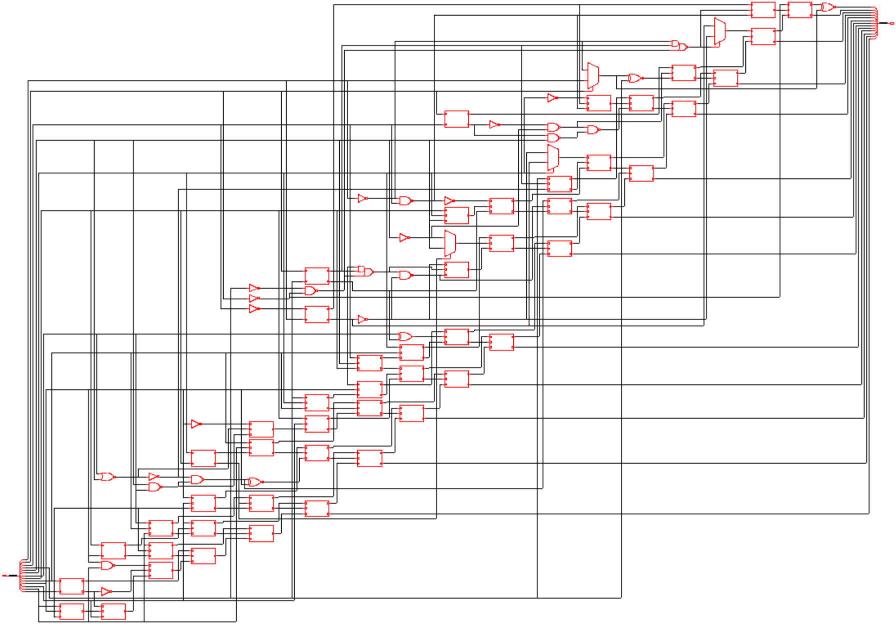




Anzahl reeller Multiplikationen für die Berechnung der 2D-DFT

Methode	Anzahl reeller Multiplikationen
komplexe Eingangswerte	128
reelle Eingangswerte	64
ladbare Matrixmultiplikation	4096
FFT	128

Konstantenmultiplikation mit $\frac{\sqrt{2}}{2} \simeq 0.70703125 = 0001011010100_2$



Anzahl benötigter Takte je Element, ungerade Zeilen

1. Spalte: $r_0 + r_1 + r_2 + r_3 + r_4 + r_5 + r_6 + r_7$

Takt		Bit
1	$\underbrace{r_0 + r_1} \qquad \underbrace{r_2 + r_3} \qquad \underbrace{r_4 + r_5} \qquad \underbrace{r_6 + r_7}$	12
	$\Downarrow \qquad \Downarrow \qquad \Downarrow \qquad \Downarrow$	13
2	$\underbrace{sum\_1\_1 + sum\_1\_2} \qquad \underbrace{sum\_1\_3 + sum\_1\_4}$	12
	$\Downarrow \qquad \Downarrow$	13
3	$\underbrace{sum\_2\_1 + sum\_2\_2}$	12
	$\Downarrow$	13
	sum_3_1	12

Anzahl benötigter Takte je Element, gerade Zeilen

2. Spalte: $r_1 - r_3 + i_1 - i_7 + i_3 - r_5 + r_7 - i_5 + r_0 - r_4 + i_2 - i_6$

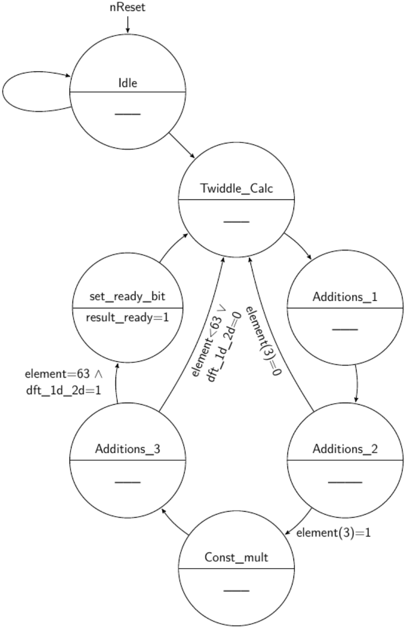
Takt		Bit
1	$\underbrace{r_1 - r_3} \quad \underbrace{i_1 - i_7} \quad \underbrace{i_3 - r_5} \quad \underbrace{r_7 - i_5} \quad \underbrace{r_0 - r_4} \quad \underbrace{i_2 - i_6}$	12
	$\Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow$	13
2	$\underbrace{sum1\_1 + sum1\_2} \quad \underbrace{sum1\_3 + sum1\_4} \quad \underbrace{sum1\_5 + sum1\_6}$	12
	$\Downarrow \quad \Downarrow \quad \Downarrow$	13
3	$\underbrace{sum2\_1 + sum2\_2}$	12
	$\Downarrow \quad \Downarrow$	13
	$sum3\_1$	13
	$\Downarrow \quad \Downarrow$	13
4	$\frac{\sqrt{2}}{2}$	13
	$\Downarrow \quad \Downarrow$	26
5	$\underbrace{sum4\_1 + sum2\_3}$	12
	$\Downarrow$	13
	$sum5\_1$	12

Summe der Takte für die Berechnung der 2D-DFT

Zeile	Additionen pro Element (N)	Takte pro Element ($\log_2(N)$)	Takte für Multiplikation	Summe der Takte
1	8	3	0	3
2	12	3,6	1	5
3	8	3	0	3
4	12	3,6	1	5
5	8	3	0	3
6	12	3,6	1	5
7	8	3	0	3
8	12	3,6	1	5

Summe der Takte ist $(3 + 5) \cdot 4 \cdot 8 \cdot 2 = 512$

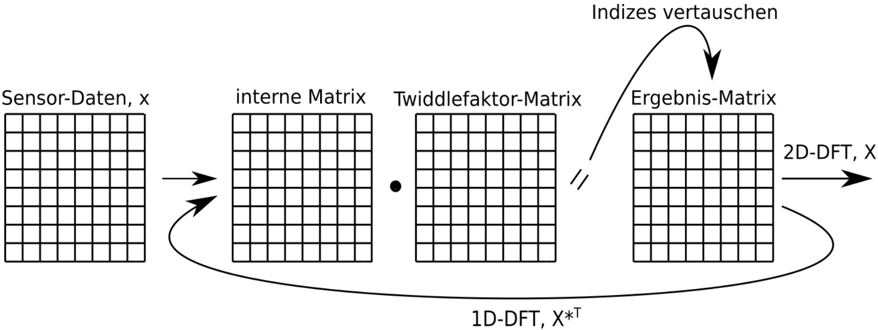
Zustandsfolge



Alternative Schreibweise der 2D-DFT als Matrixmultiplikation

$$
\begin{aligned}
X &= W \cdot x \cdot W \\
&= \left((x \cdot W)^T \cdot W \right)^T \\
&= \left(X^{*T} \cdot W \right)^T
\end{aligned}
$$

Alternative Schreibweise der 2D-DFT als Matrixmultiplikation



Evaluation: Übersicht

- Testumgebung
- Zeitabschätzung
- Chipimplementation

Evaluation: Testumgebung

- Simulation mit NC Sim und SimVision
  - nützlich für Teilfunktionen
  - Betrachtung einzelner Signalverläufe
- Automatisierung durch Shell-Skript
  - Simulation mit NC Sim und TCL-Skript
  - Berechnung mittels Matlab
  - Vergleich

Evaluation: Zeitabschätzung

$$RPM = 8000\,\mathrm{min}^{-1}$$

$$\frac{RPM}{60} = 1333,\bar{3}\,\mathsf{sec}^{-1}$$

$$\curvearrowright 1\,\mathrm{Umdrehung} = \frac{1}{1333,\bar{3}\,\mathsf{sec}^{-1}}$$

$$= 7,5 \cdot 10^{-3}\,\mathrm{sec}$$

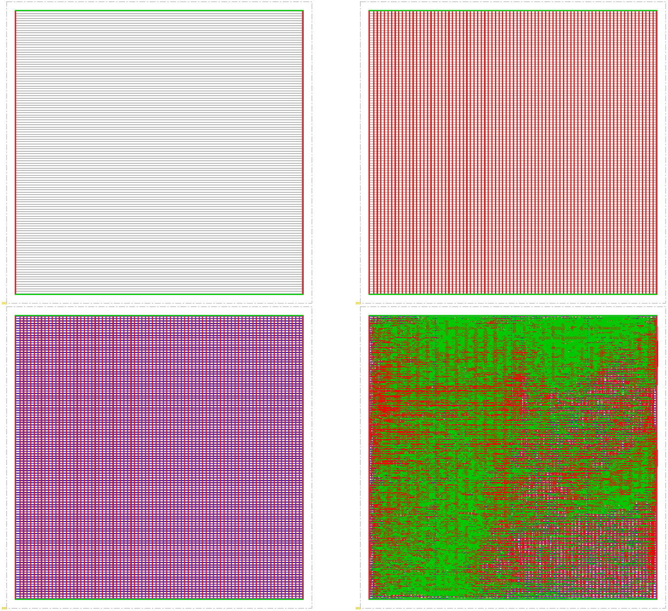
$$1^\circ \mathrel{\widehat{=}} \frac{7,5 \cdot 10^{-3}\,\mathrm{sec}}{360}$$

$$1^\circ \mathrel{\widehat{=}} 20,83 \cdot 10^{-6}\ \mathsf{sec}$$

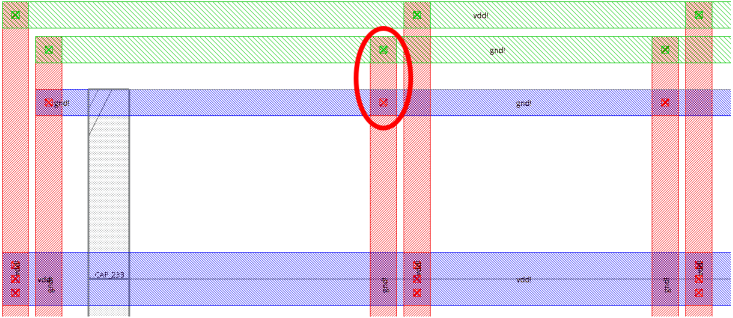
$$100 \cdot 10^{6}\ Hz = 10 \cdot 10^{-9}\ \mathsf{sec}$$

$$\frac{20,83 \cdot 10^{-6}\,\mathsf{sec}}{10 \cdot 10^{-9}\,\mathsf{sec}} = 2083\,\mathrm{Takte}$$

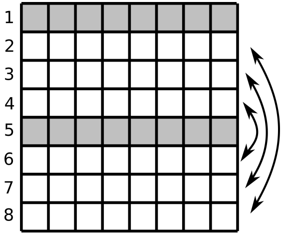
Evaluation: Chipimplementation



Evaluation: Chipimplementation



Zusatzfeature: Implementation der IDFT



Zusammenfassung

- DFT als 8x8 hat sich als effizienteste erwiesen
- Optimierung der Multiplikationen mit der Twiddlefaktormatrix
- Kritischer Pfad scheint Konstantenmultiplikation zu sein
- Berechnung der 1D- und 2D-DFT mit selber Einheit
- Benötigte Takte liegen im realistischen Rahmen
- IDFT kann durch geringe Ergänzungen im Quelltext berechnet werden
- Wertvolle Grundlagen für die Implementation der 15x15 2D-DFT

Ausblick

- Reduzierung des kritischen Pfades
  - auf zwei Gatter aufteilen
  - Wallace-Tree verwenden
- 15x15 mit ähnlich vielen Takten