TECHNISCHE UNIVERSITÄT DRESDEN

Entwurf und Implementierung einer komplexen Floating-Point-Einheit für FPGAs

Patrick Russell

Dresden, 11. Juni 2015

Aufgabenstellung

Theoretische Grundlagen

Entwurf

Implementierung & Test

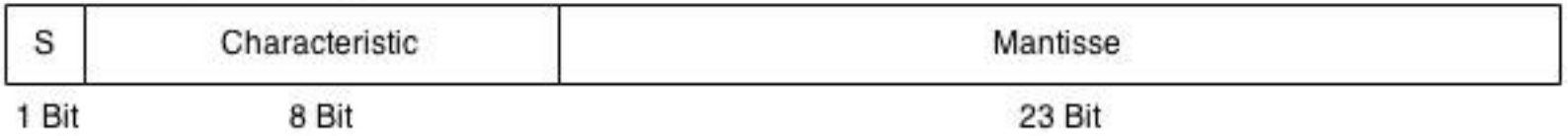
Auswertung

01 Aufgabenstellung

- Implementierung einer FPU auf RT-Ebene mittels VHDL
- Unterstützung der vier Grundrechenarten
- Generisch für einfache als auch doppelte Genauigkeit
- Geringe Priorität für Divisionen (Latenz, Durchsatz)
- Divisionen mittels Newton-Raphson-Verfahren

02 Theoretische Grundlagen - IEEE-754

IEEE Floating Point Single Precision



IEEE Floating Point Double Precision

$$E = C - B$$

$$Z = (-1)^S * M * b^E$$

Gleitkommazahlen - Grundrechenarten

- Bestimmung des vorläufigen Ergebnisexponenten
- Aufbereiten der Operandenmantissen
- Durchführung der Operation
- Normalisierung und Rundung des Ergebnisses

Division (Newton-Raphson Verfahren)

- Multiplikation des Dividenten mit dem Kehrwert des Divisors

$$q = \frac{a}{d} = a * \frac{1}{d}$$

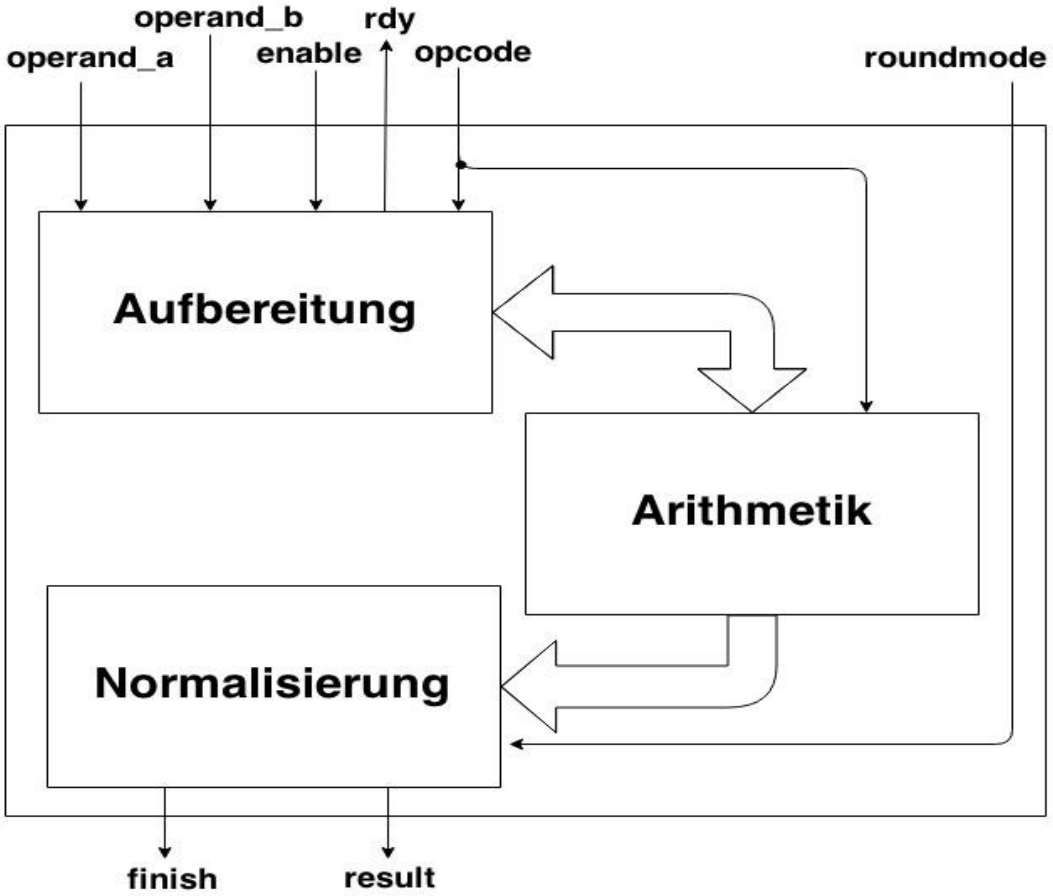
- Nullstellenbestimmung der Funktion f(x)

$$f(x) = \frac{1}{x} - d$$

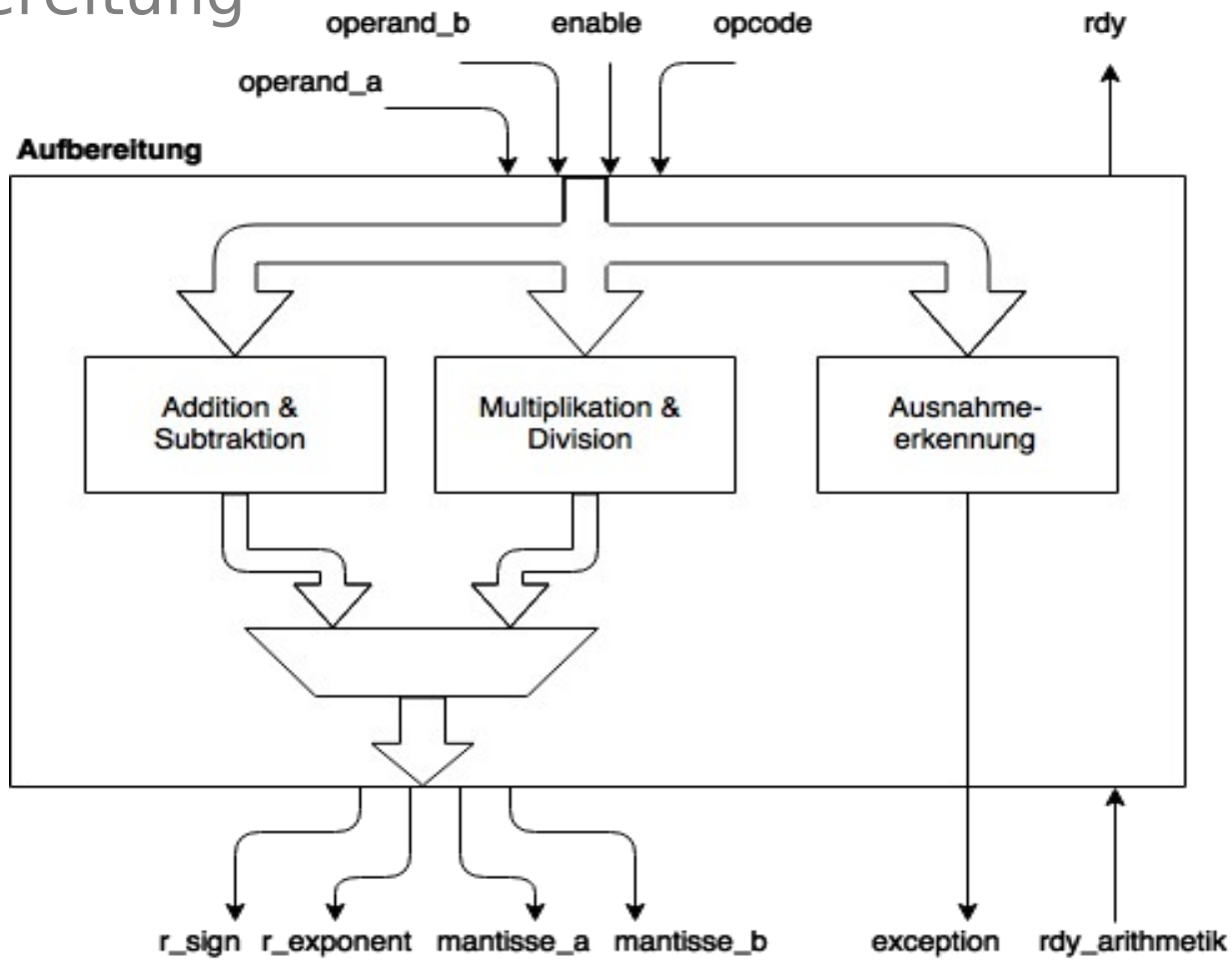
- Iterative Berechnung des Kehrwertes

$$x_{i+1} = x_i(2 - dx_i)$$

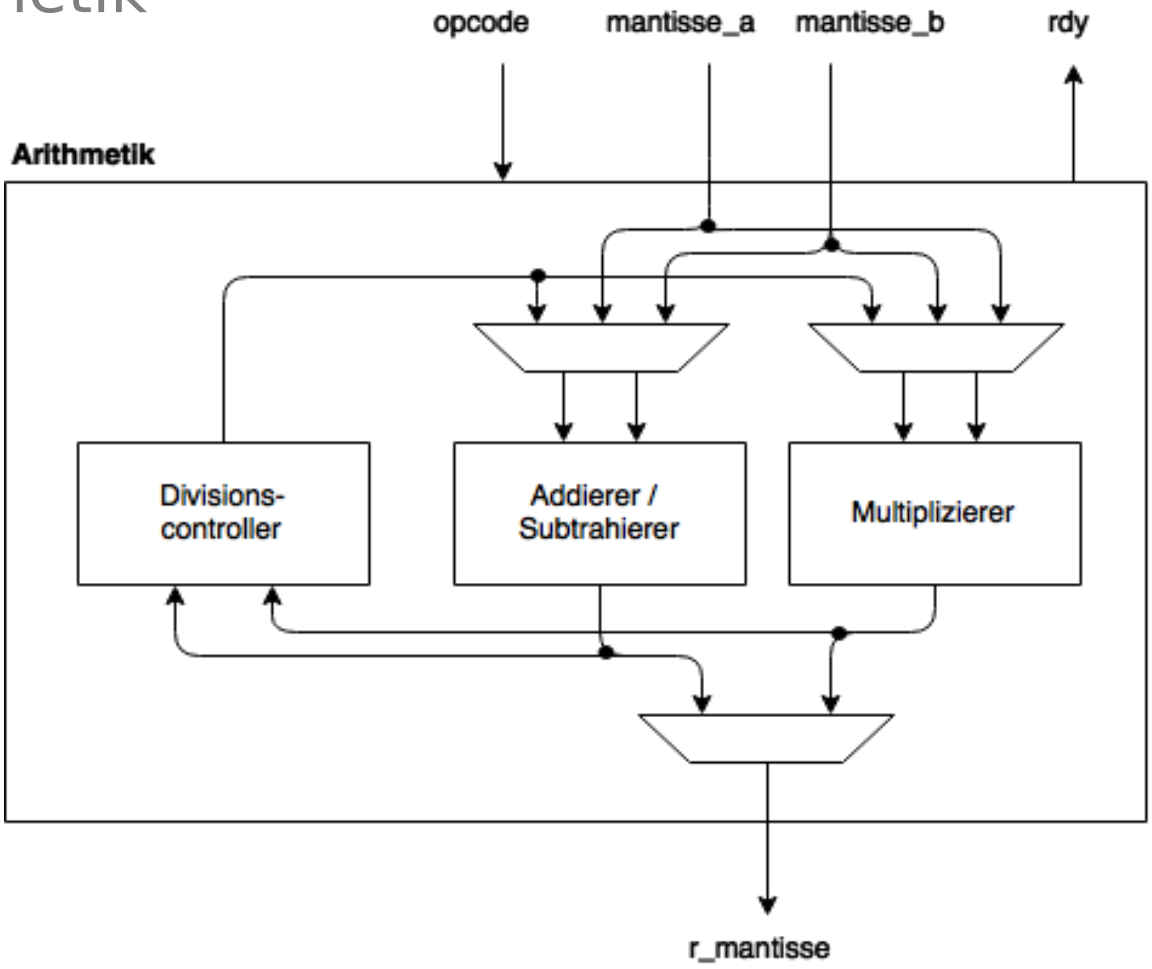
03 Entwurf - Blockschaltbild



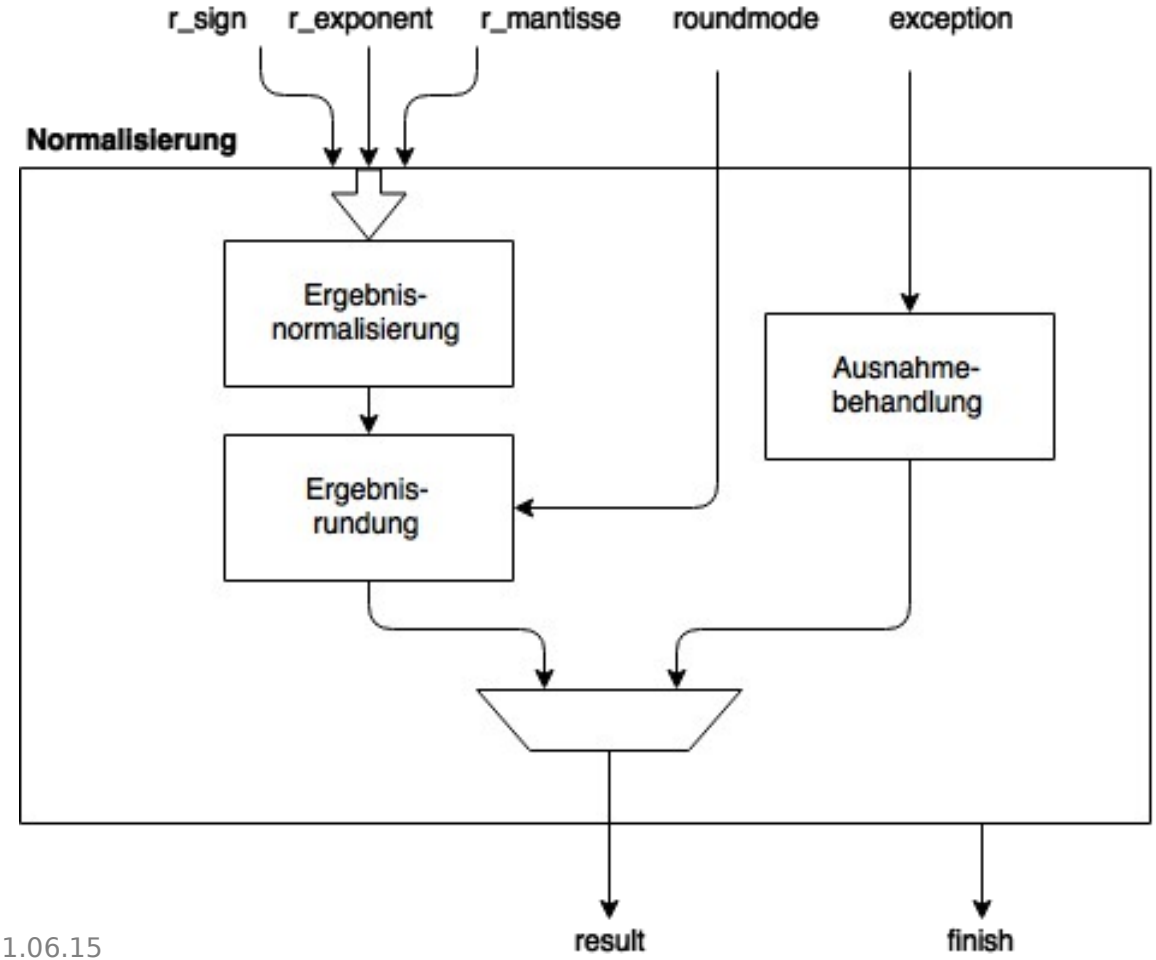
Aufbereitung



Arithmetik


opcode
mantisse_a
mantisse_b
rdy
Arithmetik
Divisions-
controller
Addierer /
Subtrahierer
Multiplizierer
r_mantisse

Normalisierung



04 Implementierung - Vorzeichenwechsel

- Vorzeichen im IEEE-Format als Vorzeichenbit angegeben.
- Arithmetik muss Vorzeichenwechsel beachten.
- Ordnen der Operanden spart 2er-Komplement Konvertierung.
- Direkte Berechnung des Ergebnisvorzeichens.
- Operation muss angepasst werden.

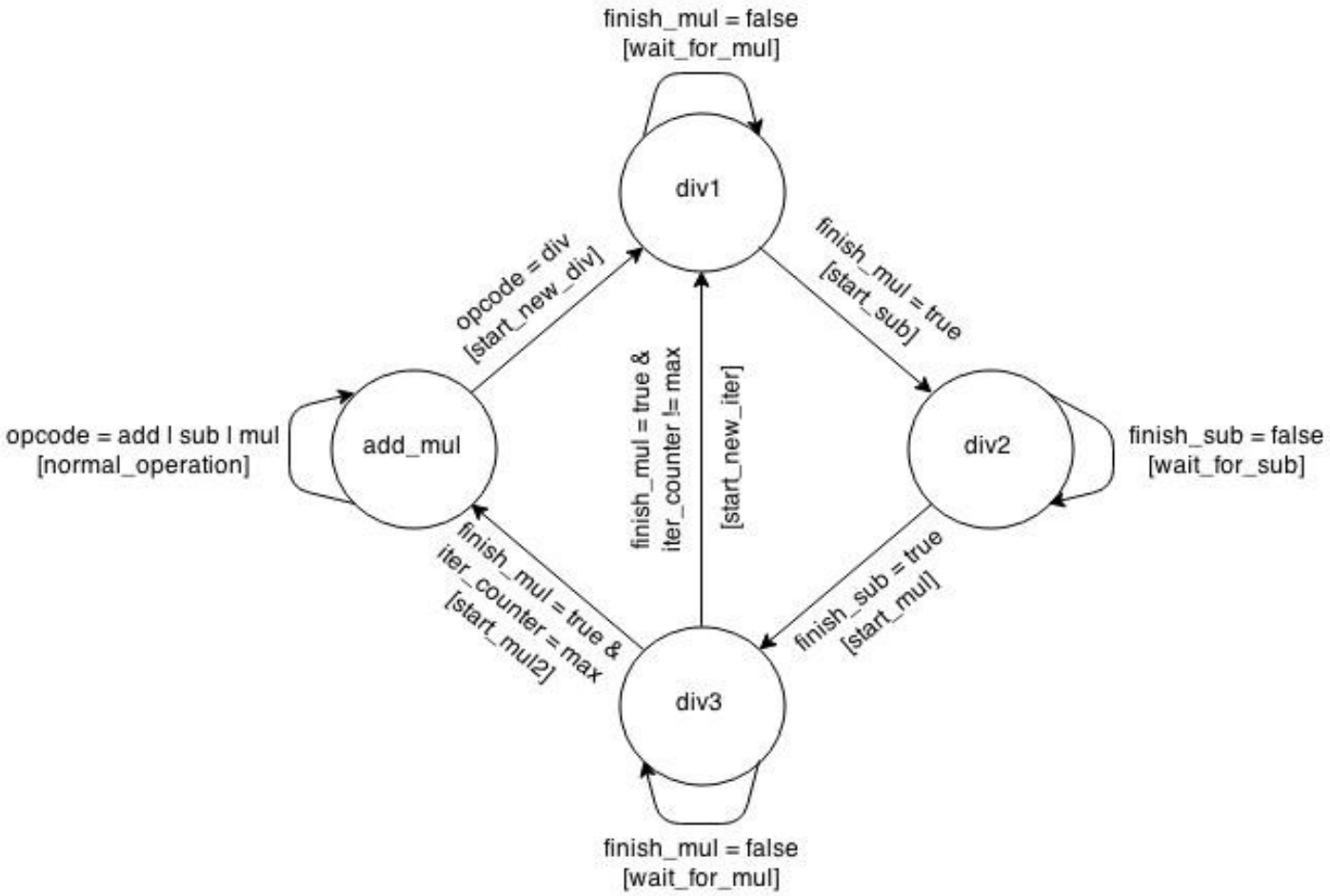
Vorzeichenwechsel – Ordnen der Operanden

Vorz. A	Vorz. B	Operation	angepasste Operation	Vorz. Ergebnis
+	+	add	add	+
+	-	add	sub	- wenn Tausch sonst +
-	+	add	sub	+ wenn Tausch sonst -
-	-	add	add	-
+	+	sub	sub	- wenn Tausch sonst +
+	-	sub	add	+
-	+	sub	add	-
-	-	sub	sub	+ wenn Tausch sonst -

Divisionscontroller I

- Steuerung des Datenpfades innerhalb der Arithmetik
- Abfangen von Divisionen
- Blockierung der Arithmetik
- Abarbeitung der nötigen Iterationsschritte

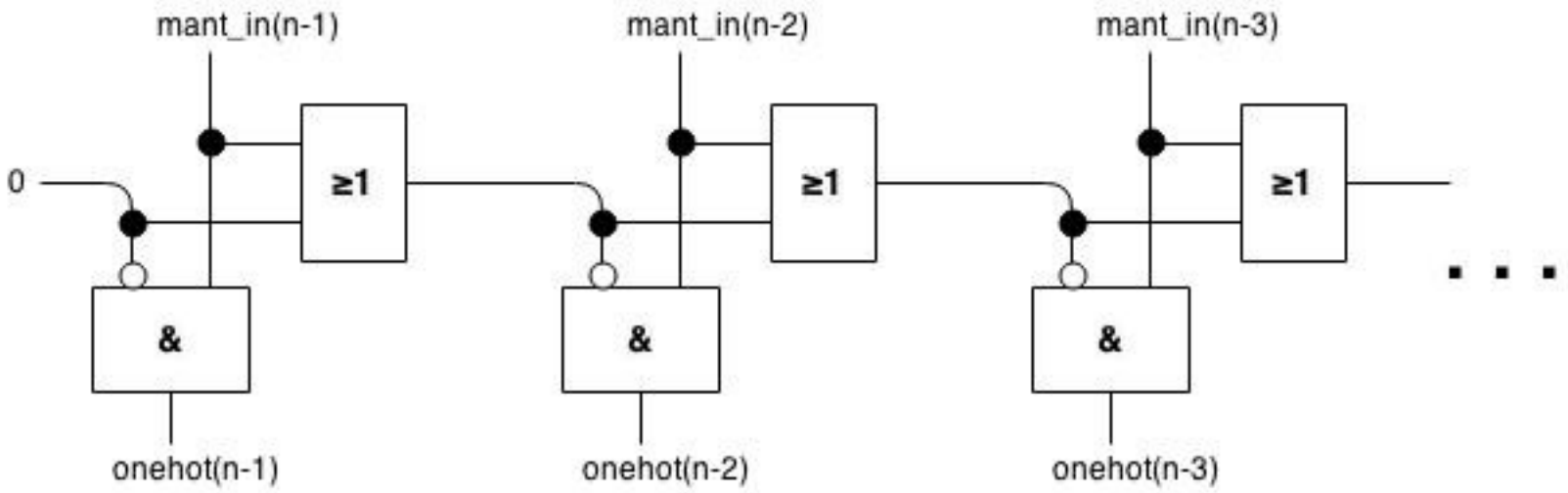
Divisionscontroller II



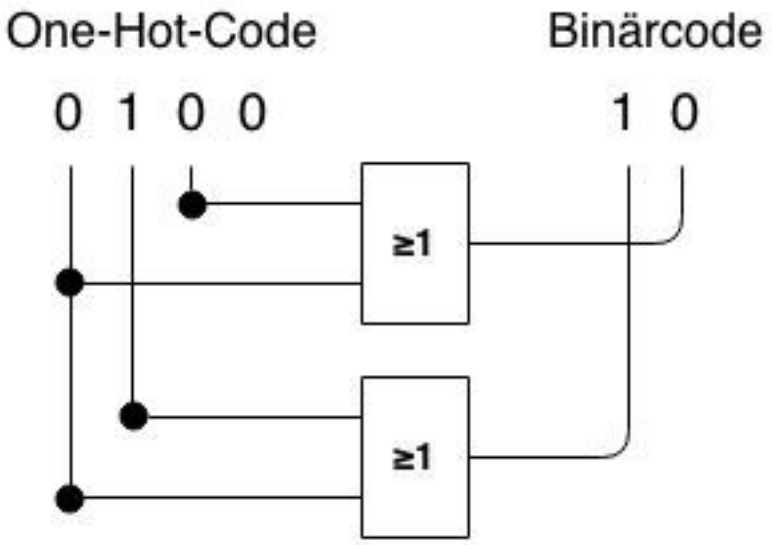
Normalisierung

- Schiebeoperation mit dynamischer Weite
- Lokalisieren der ersten Eins innerhalb der Mantisse
- One-Hot-Kodierung der Mantisse
- Binärkodierung des One-Hot-Codes

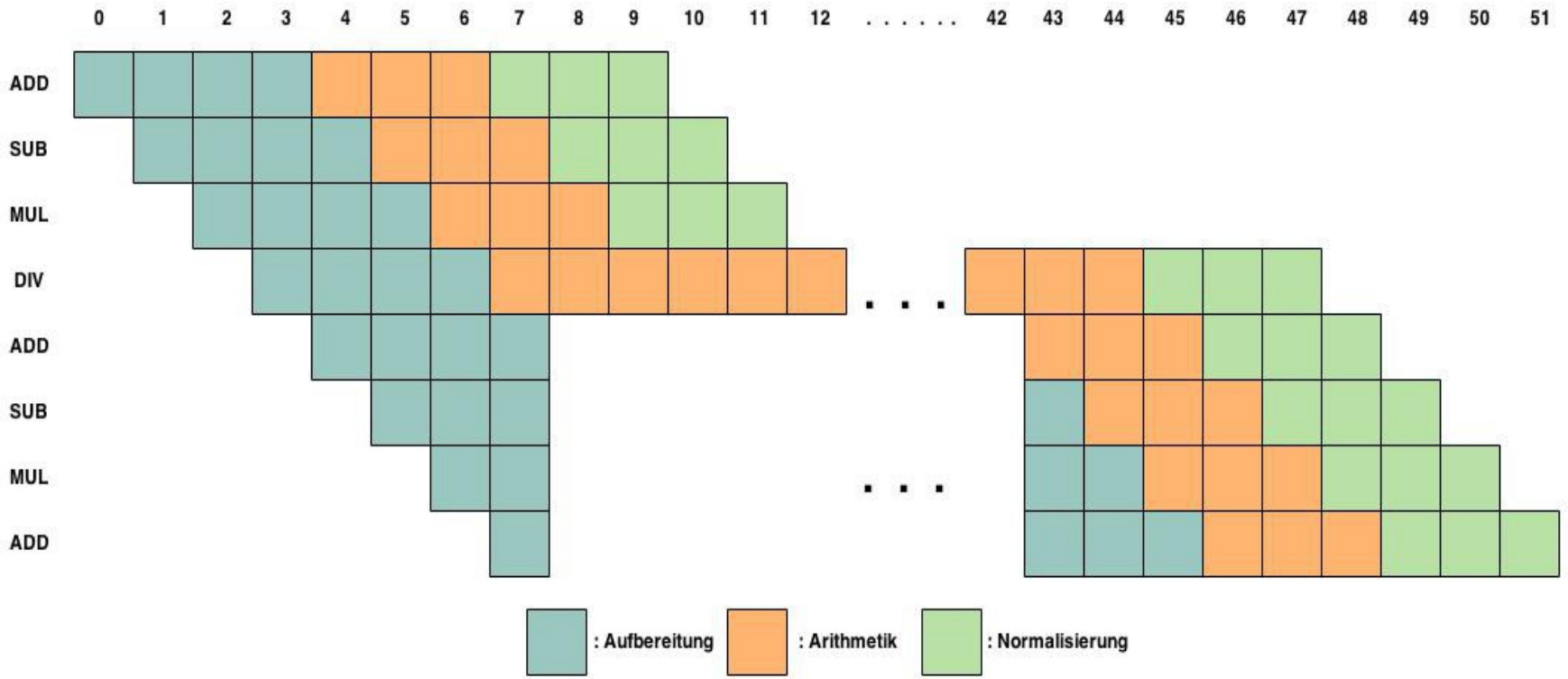
Normalisierung (One-Hot-Kodierung)



Normalisierung (Binärkodierung)



Pipelining


0 1 2 3 4 5 6 7 8 9 10 11 12 . . . . . 42 43 44 45 46 47 48 49 50 51
ADD
SUB
MUL
DIV
ADD
SUB
MUL
ADD
: Aufbereitung
: Arithmetik
: Normalisierung

Testmustergenerierung

- Automatische Testmustergenerierung (C-Programm)
- Händisch erstelle Testdaten (Ausnahmen etc.)
- Vergleich der Ergebnisse durch VHDL-Testbench

05 Auswertung - Funktionsumfang

- Grundrechenarten mit einfacher und doppelter Genauigkeit
- Ein Ergebnis pro Takt Durchsatz und 10 Taktzyklen Latenz
- Division blockiert 35 bzw. 42 Taktzyklen die Arithmetik
- Einfache Funktionserweiterbarkeit
- Derzeit noch Rundungsfehler bei der Division

Ressourcenverbrauch

	Virtex6	Single Precision		Double Precision	
Logic Utilization	Available	Used	Utilization	Used	Utilization
Slice Register	93120	951	1%	1933	2%
Slice LUTs	46560	1555	3%	2978	6%
Fully used LUT-FF-pairs	4007	529	26%	906	22%
BUFG/BUFGCTRLs	32	1	3%	1	3%
DSP48E1s	288	2	0%	12	4%
Maximum frequency		268.479MHz		119.610MHz	

Ressourcenverbrauch - Vergleich

	FPU	Xilinx IP-Cores
Slice Register	1933	4643
Slice LUTs	2978	4392
DSP48E1s	12	14
Maximum frequency	119.610MHz	275.160MHz

Ausblick

- Korrektur der Division
- Funktionserweiterung (Wurzel, CORDIC)
- Status Flags (Overflow, Division durch Null)
- Weitere Pipelinestufen

Literatur

[1] LogiCORE IP FLoating-Point Operator v6.1. http://www.xilinx.com/support/documentation/ip_documentation/floating_point/v6_1/pg060-floating-point.pdf, 2012 [Online; abgerufen am 30. Mai 2015]

[2] Wikipedia. IEEE_754. http://de.wikipedia.org/wiki/IEEE_754, 2015. [Online; abgerufen am 18. Mai 2015]

Divisionsfehler

$$a = 1.10010_2 = 1.5625_{10}$$

$$d = 0.10100_2 = 0.625_{10}$$

$$\frac{1}{d} = 1.10011_2 = 1.59375_{10}$$

$$q_1 = \frac{a}{d} = 10.1000000000_2 = 2.5_{10}$$

$$q_2 = a * \frac{1}{d} = 10.0111110110_2 = 2.490234375_{10}$$