

Architectures for Floating-Point Division

Hooman Nikmehr

B.Sc. University of Tehran

M.Eng.Sc. University of Tehran

A thesis submitted in fulfilment of the requirements for the degree of
Doctor of Philosophy
in the
School of Electrical and Electronic Engineering
The University of Adelaide
Australia

Supervisors: **Dr. Cheng-Chew Lim** and **Dr. Braden Phillips**

August, 2005

Copyright ©2005
Hooman Nikmehr
All Rights Reserved

Contents

Abstract	xv
Statement of Originality	xvii
Acknowledgments	xix
Publications	xxi
List of Principal Symbols	xxiii
List of Abbreviations	xxvii
1 Introduction	1
1.1 Motivation	2
1.2 Overview	2
1.2.1 Importance of FP Division	3
1.2.2 Division Algorithm Taxonomy	4
1.3 Research Objectives	6
1.4 Research Contributions	7
1.5 Thesis Organisation	9
2 Division Algorithms	11
2.1 Introduction	12
2.2 Functional Division Algorithms	12
2.2.1 Newton-Raphson	12
2.2.2 Goldschmidt	13
2.2.3 Newton-Raphson versus Goldschmidt	14
2.2.4 Features	14

2.3	Digit Recurrence Algorithms	15
2.3.1	Definitions and Notations	15
2.3.2	Recurrence	16
2.3.3	Restoring Division	17
2.3.4	Non-Restoring Division	19
2.3.5	Redundant Digit Sets	20
2.3.6	Radix-2 SRT Algorithm	21
2.3.7	High-Radix SRT Algorithm	23
2.4	Digit Recurrence versus Functional	27
2.5	Summary	27
3	SRT Division Algorithm Implementation	29
3.1	Introduction	30
3.2	QDS Function	30
3.2.1	Introduction	30
3.2.2	PD Plot Method	31
3.2.3	Selection Constants Method	38
3.3	Division Radix	42
3.4	Redundancy Factor	43
3.5	PR Representation	44
3.6	Quotient Conversion Method	45
3.7	Overlapping Iteration Components	46
3.7.1	Overlapped QDS Function	46
3.7.2	Overlapped PR Formation	49
3.7.3	Overlapped QDS Function and PR Formation	50
3.7.4	Hybrid Overlap	51
3.8	Number Representation in the IEEE 754 Standard	51
3.9	FP Division Using the SRT Algorithm	52
3.9.1	Rounding and Post-Normalising	53
3.9.2	Assumptions to Match SRT Division with the IEEE 754 Standard	56
3.10	Summary	57
4	Comparison Multiples, a Different Approach to Quotient Digit Selection	59
4.1	Introduction	60

4.1.1	Retimed Low Power Implementation	61
4.1.2	Implementation Used in the ARM FP Macrocell	63
4.1.3	Retimed Implementation of ARM Divider	65
4.2	Comparison Multiples Based FP Division	65
4.2.1	PR Representation	66
4.2.2	Comparison Multiples Based QDS Function	67
4.2.3	QDS Function Structure	68
4.2.4	QDS Function Evaluation	71
4.2.5	FP Division Structure	72
4.2.6	FP Division Evaluation	72
4.3	FP Division Optimisation	73
4.3.1	QDS Function Optimisation	73
4.3.2	Optimised QDS Function Evaluation	77
4.3.3	Recurrence Optimisation	78
4.3.4	Optimised Recurrence Evaluation	78
4.4	QDS Function Operands Precisions	80
4.4.1	e' and c'	80
4.4.2	e'' and c''	82
4.5	Summary	83
5	Comparison Multiples Based Radix-4 and Radix-16 Floating-Point Dividers	85
5.1	Introduction	86
5.2	Radix-4 FP Divider	86
5.2.1	Assumption	86
5.2.2	Precisions	86
5.2.3	QDS Function	87
5.2.4	Recurrence	97
5.2.5	Convert, Round and Normalise Unit	102
5.3	Radix-16 FP Divider	102
5.3.1	Dataflow Through Overlapped Stages	103
5.3.2	Digit Set and Iterations	105
5.3.3	Precisions	105
5.3.4	QDS Function	106
5.3.5	Recurrence	107

5.3.6	Convert, Round and Normalise Unit	109
5.4	CRN Unit	109
5.4.1	Previous Approaches	109
5.4.2	New Approach	112
5.4.3	Evaluation of the Proposed Rounding Algorithm	117
5.5	Summary	119
6	Decimal Signed-Digit Arithmetic, A New Approach	121
6.1	Introduction	122
6.2	Background	123
6.3	DSD Number Representation	124
6.4	DSD Negation	125
6.5	DSD Carry-Free Addition	125
6.5.1	DCFA with DSD Augend and Addend	125
6.5.2	DCFA with DSD Augend and BCD Addend	132
6.5.3	DCFA with BCD Augend and Addend	136
6.6	DSD Carry-Free Subtraction	137
6.6.1	DSD Minuend and Subtrahend	137
6.6.2	DSD Minuend and BCD Subtrahend	138
6.6.3	BCD Minuend and Subtrahend	139
6.7	DSD to BCD Conversion	140
6.7.1	Assumptions and Definitions	140
6.7.2	Algorithm	141
6.7.3	Implementation	143
6.7.4	DSD Sign Detection Using DSD2BCD Algorithm	145
6.7.5	Combined BCD Adder/Subtractor	145
6.8	Evaluation	145
6.9	Summary	147
7	Comparison Multiples Based Decimal Floating-Point Divider	149
7.1	Introduction	150
7.2	Digit Recurrence Based Decimal Division History	150
7.3	DFP Representation in IEEE 754R Standard	151
7.4	DFP Division Definition	152

7.5	Precision and Rounding Modes	153
7.5.1	Precision	154
7.5.2	Rounding Modes	154
7.6	DFP Division Through SRT Algorithm	155
7.6.1	Assumptions	155
7.6.2	DFP Division Formulation	156
7.6.3	Convert and Round	158
7.6.4	Dealing with Exact Results	158
7.7	Implementation	160
7.7.1	DFP versus Previously Proposed Binary Divider	160
7.7.2	Determining the QDS Function Operands Precisions	161
7.7.3	QDS Function	163
7.7.4	Recurrence	171
7.7.5	Evaluation	179
7.8	Summary	180
8	Timing Evaluation of the Floating-Point Dividers	181
8.1	Introduction	182
8.1.1	Functional Evaluation	182
8.1.2	Timing Evaluation	183
8.2	Logical Effort	185
8.3	Radix-4 FP Divider Timing Evaluation	187
8.3.1	Full-Adders Implemented for Speed	188
8.3.2	Binary Carry Generators Implemented for Speed	189
8.3.3	Recurrence Critical Path	192
8.3.4	Logical Effort Calculation	195
8.3.5	Division Execution Time	199
8.4	Radix-16 FP Divider Timing Evaluation	199
8.4.1	Recurrence Critical Path	199
8.4.2	Logical Effort Calculation	199
8.4.3	Division Execution Time	199
8.5	DFP Divider Timing Evaluation	201
8.5.1	DSD Borrow Generators Implemented for Speed	201
8.5.2	Recurrence Critical Path Choices	203

8.5.3	Logical Effort Calculation	207
8.5.4	Division Execution Time	208
8.6	Discussion	208
8.6.1	Radix-4 FP Divider	208
8.6.2	Radix-16 FP Divider	211
8.6.3	DFP Divider	211
8.7	Summary	212
9	Conclusions and Future Works	213
9.1	Conclusions	214
9.1.1	Comparison Multiples Approach	214
9.1.2	Comparison Multiples Based Radix-4 and Radix-16 FP Divider	215
9.1.3	Comparison Multiples Based DFP Divider	215
9.1.4	Timing Evaluation	215
9.2	Future Work	216
A	Radix-4 and Radix-16 CRN Units Tables	217
B	VHDL Code of the Radix-4 Divider	221
B.1	adjust.vhd	221
B.2	compsd.vhd	222
B.3	comparator.vhd	223
B.4	critical.vhd	224
B.5	divider.vhd	226
B.6	ff.vhd	232
B.7	m1m2invert.vhd	233
B.8	multiplegen.vhd	233
B.9	mux1muxs.vhd	234
B.10	prformation.vhd	236
B.11	prsd.vhd	236
B.12	qds.vhd	237
	Bibliography	241

List of Figures

1.1	Microprocessor stall time distribution.	5
1.2	Consumers of FP division results.	5
1.3	Taxonomy of division algorithms.	6
2.1	Components of an iteration.	17
2.2	Robertson diagram for restoring division when $r = 2$	19
2.3	Robertson diagram of non-restoring division with $r = 2$	20
2.4	Robertson diagram for the radix-2 SRT division.	22
2.5	Robertson diagram for the radix-2 SRT division with $d \in [\frac{1}{2}, 1)$	23
2.6	Allowable region for selecting q_{j+1} in high-radix SRT division.	24
2.7	Robertson diagram of high-radix SRT division.	25
2.8	A maximally redundant QDS function operating based on the separating points $s_k(d)$	26
3.1	The PD plot for $q_{j+1} = k$ and $q_{j+1} = k + 1$	32
3.2	Implementation of the QDS function through the PD plot method.	35
3.3	The PD plot for $r = 4$ and $\rho = 1$	36
3.4	The selection constants for the interval $[d_i, d_{i+1})$	39
3.5	The QDS function implemented through the selection constants method.	41
3.6	Critical path of the SRT division, indicated in red.	42
3.7	A CFA used in the recurrence of high-radix SRT division.	45
3.8	Implementation of the QDS function with a redundant PR.	45
3.9	Overlapping the iteration components.	47
3.10	The design with no overlap among the components.	47
3.11	Overlapping the QDS function.	48
3.12	Overlapping the PR formation.	49
3.13	Overlapping the QDS function and the PR formation.	50

3.14	Hybrid overlapping.	51
3.15	The IEEE 754 standard formats for representing FP numbers.	53
3.16	Structure of FP divider complying the IEEE 754 standard.	54
4.1	Implementation of high-radix SRT division.	62
4.2	Removing buffers from the critical path.	63
4.3	Implementation of the QDS function using the comparators.	64
4.4	Retimed version of the QDS function.	66
4.5	Implementing the QDS function using the comparison multiples method. BSDA indicates the BSD adders.	70
4.6	The proposed FP divider based on the comparison multiples approach. The block named Adj represents the adjust unit.	72
4.7	The two paths run in parallel in the proposed FP divider structure. . . .	72
4.8	Optimised implementing the comparison multiples based QDS function.	76
4.9	The optimised implementation of FP division based on the redefined QDS function.	79
4.10	The three paths run in parallel in the optimised FP divider structure. . .	80
5.1	The general structure of the radix-4 QDS function.	87
5.2	General structure of the comparison multiple generator used in the proposed radix-4 FP divider.	91
5.3	An implementation of a BSD adder with a BSD augend and a 2's com- plement addend.	93
5.4	General structure of the comparator used in the radix-4 FP divider, where $k = 1, 2$ and $\{M_2\}_5 \equiv \{M'_2\}_5$	93
5.5	An architecture for n -digit BSD sign detectors using carry generators. For the sign detectors used in the proposed radix-4 QDS function $n = 7$.	95
5.6	Implementation of the coder used in the proposed radix-4 FP divider. . .	96
5.7	Implementation of the proposed recurrence of the radix-4 FP division. . .	98
5.8	Factor generator used in the implementation of the radix-4 FP division.	99
5.9	Implementation of the PR formation, where w'_0 is shown as $0xx.xx \cdots x00$.	100
5.10	Implementation of the adjust unit.	101
5.11	Implementation of the proposed radix-16 FP division recurrence.	104
5.12	Scheme for implementing the RTNE using on-the-fly rounding algorithm.	111
5.13	Implementation of the radix-4 CRN.	116

5.14	Realisation of the radix-16 CRN unit.	117
6.1	The general structure of a 1-digit DD-DCFA.	127
6.2	An n -digit DD-DCFA implemented using 1-digit DD-DCFA blocks. . .	128
6.3	The implementation for the FRFU used in DD-DCFA.	128
6.4	An implementation of a 4-bit (4:2)-compressor.	129
6.5	The implementation of the adjust circuit used in FRFU.	130
6.6	The implementation of the TDSU used in DD-DCFA.	131
6.7	The implementation of the FRSU employed in DD-DCFA.	132
6.8	The general structure of a 1-digit DB-DCFA.	134
6.9	An n -digit DB-DCFA implemented using 1-digit DB-DCFA blocks. . . .	134
6.10	The implementation of FRFU employed in DB-DCFA.	135
6.11	The implementation of the TBSU used in DB-DCFA.	136
6.12	The implementation of FRSU employed in DB-DCFA.	136
6.13	An n -digit BB-DCFA implemented using 1-digit BB-DCFA building blocks.	137
6.14	A DSD adder/subtractor with DSD input operands using an n -digit DD-DCFA.	138
6.15	A DSD adder/subtractor with one DSD and one BCD input using an n -digit DB-DCFA.	139
6.16	A DSD adder/subtractor with BCD inputs using an n -digit BB-DCFA. .	140
6.17	An implementation for the proposed DSD2BCD converter.	144
6.18	An implementation of a combined decimal adder/subtractor.	146
7.1	The implementation of the proposed decimal QDS function.	164
7.2	The circuit mapping BCD digit $z = z_3z_2z_1z_0$ to the corresponding 9's complement value $z^C = z_3^Cz_2^Cz_1^Cz_0^C$	165
7.3	The implementation of the comparison multiple generator, for $k =$ $2, 3, \dots, 8, 9$. The final results are in the BCD format.	166
7.4	The implementation of the comparators used in the proposed decimal QDS function, for $k = 1, 2, \dots, 8, 9$	167
7.5	An implementation for 1-digit DB-DCFA', an alternative to DB-DCFA. .	168
7.6	The architecture used for implementing the comparison sign detectors and the PR employed in the proposed DFP divider.	169
7.7	Structure of the recurrence of the proposed DFP division.	172
7.8	The implementations of MUX 11:1 and MUX 10:1.	173

7.9 The implementation of the PR Formation used in the DFP divider. . . . 175

7.10 The implementation of CIRCUIT₁ used in the decimal PR formation. . . 176

7.11 An implementation for DB-DCFA'' used in the decimal PR formation. . 177

7.12 An implementation for the adjust unit used in the decimal PR formation. 179

8.1 A piece of VHDL code used for functional evaluation. 184

8.2 Implementations for 1-bit full-adder. 188

8.3 Realisations for the modified 1-digit BSD adder used in the comparators. 189

8.4 Delay estimations on Kogge-Stone and Han-Carlson based adders with
different operand widths. 192

8.5 The comparison sign detector implemented using Kogge-Stone approach. 193

8.6 A comparison sign detector realised using the MRC approach. 194

8.7 Suggested critical paths for the proposed radix-4 FP divider using the
comparator given in Figure 8.3(a). 195

8.8 Suggested critical paths for the proposed radix-4 FP divider using the
comparator given in Figure 8.3(b). 196

8.9 Critical path of the proposed radix-16 FP divider. 200

8.10 The design producing p_i and g_i for every $P_k = z_i$, where $k = 1, 2, \dots, 8, 9$. 202

8.11 An implementation for circuit producing p_i and g_i using Kogge-Stone
based borrow generator. 202

8.12 An implementation for circuit producing p_i and g_i using an MRC based
borrow generator. 203

8.13 An implementation for the network producing $P_{i;j}$ and $G_{i;j}$ from p_i and
 g_i using the Kogge-Stone approach. 204

8.14 An implementation for the network producing $P_{i;j}$ and $G_{i;j}$ from p_i and
 g_i using the MRC approach. 204

8.15 Suggested critical paths for the proposed DFP divider using the p_i/g_i
generator shown in Figure 8.11. 205

8.16 Suggested critical paths for the proposed DFP divider using the p_i/g_i
generator shown in Figure 8.12. 206

List of Tables

1.1	Performance of the FPUs of the recent microprocessors with double precision operands.	4
2.1	Decimal number 23 represented in a decimal SD set with $a = 7$	21
2.2	Possible SD sets for radices 2, 4 and 8.	21
3.1	Cases to be investigated before using the upper bounds.	34
3.2	The selection intervals and $m_k(i)$ for $r = 4$ and $\rho = 1$	41
3.3	Delay per iteration versus the radix in high-radix SRT division.	42
3.4	An example of the rounding errors for the RTNE scheme.	54
4.1	The alternative expression for the QDS function.	68
5.1	The most convenient for generating values for M_1 and M_2	88
5.2	Carry generating rule for digitwise constant-time addition $a_i^+ - a_i^- + b_i = 2s_{i+1}^+ - s_i^-$	92
5.3	Values of q_{j+1} constructed by $Mag(q_{j+1})$ and $Sign(q_{j+1})$	96
5.4	Reformatting process carried out by the adjust unit, for $k \in \{0, 1, 2\}$	101
5.5	The rules used by the radix-4 CRN unit to represent the unnormalised and unrounded quotient in the IEEE 754 standard format.	116
5.6	The rules used by the radix-16 CRN unit to represent the unnormalised and unrounded quotient in the IEEE 754 standard format.	118
6.1	Relationship among t_{out} , t_{in} , p and the final result digit.	126
6.2	Transfer digit t_{out} versus $(1 \pm p)$ signs.	131
6.3	Relationship between t_{out} , t_{in} , p and the final result digit.	133
7.1	The DFP representing specifications defined by the IEEE 754R standard.	152
7.2	Values of p corresponding to the representation format.	156

7.3	The rules used by the decimal CR units to represent the unrounded quotient in the IEEE 754R standard format.	159
7.4	The ranges, which M_k are defined.	165
7.5	Alternative rules for performing 1-digit DB-DCFA.	168
7.6	Values of q_{j+1} constructed by $Mag(q_{j+1}) = q_3q_2q_1q_0$ and $Sign(q_{j+1})$	170
8.1	Logical efforts and parasitic delays of the components used in this chapter.	197
8.2	Logical effort calculations on the critical paths in Figures 8.7 and 8.8. . .	198
8.3	Logical effort calculations on the critical path shown in Figure 8.9. . . .	200
8.4	Logical effort calculations on the critical paths in Figures 8.15 and 8.16. .	207
8.5	Critical path delays and the execution times of Dividers A, B, C, D and E.	210
A.1	The truth table of the signals generated by the radix-4 CRN unit. The last quotient digit q_{28} is represented as $Sign(q_{28})Mag(q_{28})$	217
A.2	The truth table of the signals generated by the radix-16 CRN unit. . . .	218

Abstract

Almost all recent microprocessors and DSP chips perform addition, subtraction, multiplication and division in hardware. However, studying their performance reveals that division is not carried out as fast as the other three operations. One investigation shows that while floating-point division, with about 3% of the dynamic floating-point instruction count, seems to be a relatively unimportant instruction, it may cause about 40% degradation to the overall system performance.

Several mathematical algorithms have been developed over the past 50 years to perform division quickly, with high precision. However, only a few are suitable for implementation in VLSI. Among them, digit recurrence algorithms are the most widely accepted methods of performing floating-point division in the latest processors. A survey shows that out of 13 recent processors, 11 use SRT division¹ for performing floating-point division. Investigations show that SRT division gives the best trade-off between delay and area. Selecting SRT division for implementing floating-point division is a reasonable choice because, unlike the other class of division algorithms, i.e. functional, it produces a correctly rounded quotient conforming to the IEEE 754 standard.

There are techniques for improving the performance of SRT division. Of these, increasing the speed of quotient digit selection (QDS), making the best balance between the radix and the redundancy factor, representing the partial remainder in a redundant form, converting the quotient from redundant to conventional form the on-the-fly and overlapping the division recurrence components are the most important.

In this thesis a different method of implementing the QDS function is proposed. This approach, which is described mathematically and architecturally, is based on the new *comparison multiples* idea. Unlike the traditional implementation of the QDS function, which searches for the quotient digit in a lookup table, the proposed method calculates

¹SRT division is a type of non-restoring digit recurrence division.

the quotient digit directly in sign and magnitude format. This approach almost halves the fan out of some critical path components, which therefore operate faster. Having received the truncated partial remainder, the QDS function compares it with truncated multiples of the divisor to find the range in which the partial remainder belongs. The results of the comparisons are converted to the magnitude of the quotient digit using simple logic called the coder. Concurrently, another circuit checks the truncated partial remainder to determine whether the quotient digit is negative. This circuit operates off the critical path since the comparison multiples based QDS function calculates the sign and magnitude of the quotient digit separately. Having applied these changes, a faster QDS function and consequently, a shorter critical path delay for the floating-point divider is obtained. Implementations of radix-4 and radix-16 floating-point dividers are investigated and optimised to further decrease the cycle time.

The idea of comparison multiples is extended to radix 10 to implement a decimal floating-point divider complying with the IEEE 754R standard. To achieve this goal, decimal signed-digit arithmetic along with implementations of carry-free addition and subtraction are proposed. The original comparison multiples based implementation of high-radix SRT division is modified to suit radix 10.

The binary and decimal implementations of comparison multiples based division are evaluated for delay. Using the method of logical effort, the radix-4, radix-16 and decimal floating-point dividers are found to be faster than corresponding circuits reported in the public literature.

Statement of Originality

I hereby declare that this work contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution and to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text.

I give consent to this copy of my thesis, when deposited in the University Library, being available for loan and photocopying.

Hooman Nikmehr

15 August 2005

Acknowledgments

I would like to thank my supervisors, Dr. Cheng-Chew Lim and Dr. Braden Philips, who have technically and mentally supported me during my PhD research. Dr. Lim taught me how to successfully pass academic milestones by hardwork and punctuality and Dr. Philips opened my eyes to different perspectives of the design for digital arithmetic.

I would also like to express my sincere thanks to Mr. Ron Seidel, who is one of the best friends I have ever had. I have benefited from his priceless advice to cope with the problems, fears and confusion I have had during my residency in Australia.

I must express my gratitude to my mother who took care of the bureaucracy related to my scholarship in Iran.

Finally, without the endless love, intense passion and inexpressible patience that my wife, Mehrnaz, has offered me, even dreaming of finishing this thesis was a dream.

Publications

1. H. Nikmehr and C. C. Lim. A New On-the-fly Summation Algorithm. In *Proceedings of 8th Asia-Pacific Computer Systems Architecture Conference ACSAC 2003*, volume 2823 of *Lecture Notes in Computer Science*, pages 258267, Aizu-Wakamatsu, Japan, 2326 September 2003.
2. H. Nikmehr and B. Phillips and C.C. Lim. A Decimal Carry-Free Adder. In *Proceeding of SPIE conference on Smart Materials, Nano-, and Micro-Smart Systems 2004*, pages 786-797, Sydney, Australia, 13-15 December 2004.

List of Principal Symbols

$1.f$	significand or mantissa (IEEE 754 standard)
a	largest digit in a SD set
b	borrow
B_i	group borrow
β	representation radix (IEEE 754 standard)
β_P	number of integer bits of the shifted PR (PD plot)
C_{in_i}	gate i input capacitance (logical effort)
C_{out_i}	load capacitance driven by the gate i (logical effort)
c'	number of fractional digits of the shifted PR used in the comparisons
c''	number of fractional digits of the shifted PR used in the PR sign detection
d	divisor significand or <i>coefficient</i>
D	d (PD plot)
D	divisor
\widehat{D}	minimum path delay (logical effort)
E	exponent (IEEE 754 standard)
e'	number of shifted PR integer digits used in the comparisons
e''	number of shifted PR integer digits used in the PR sign detection
ε_D	precision at which D is examined (PD plot)
ε_P	precision at which P is examined (PD plot)
ε_q	error of q with respect to an infinite precision quotient $\frac{x}{d}$
$\varepsilon_q[j + 1]$	error of q after $(j + 1)$ -th iteration
e_D	divisor exponent
e_Q	quotient exponent
e_X	dividend exponent
f_i	stage effort (logical effort)
\widehat{f}	minimum stage effort (logical effort)

F	path effort (logical effort)
f	a dimension related to the fan out (parallel-prefix taxonomy)
g_i	stage logical effort (logical effort)
G	path logical effort (logical effort)
g	generate
G	guard bit (rounding)
h_i	stage electrical effort (logical effort)
\widehat{h}_i	minimum stage electrical effort (logical effort)
H	path electrical effort (logical effort)
k	kill
l	dimension related to number of logic levels (parallel-prefix taxonomy)
LE_i	total logical effort born by a component
L	last bit (rounding)
L_k	continuity condition lower bound
$Mag(k)$	magnitude of SD number k
$m.n$	number represented in m integer and n fractional digits/bits
m_k	selection constant
M_k	comparison multiple
\widehat{N}	best number of stages (logical effort)
N_D	total number of bits of D , which are examined (PD plot)
N_P	total number of bits of P , which are examined (PD plot)
O	proportional to
$P_{i:j}$	group propagate
p	propagate
P	PR (PD plot)
p	precision (DFP)
p_i	stage parasitic delay (logical effort)
P	path parasitic delay (logical effort)
q	quotient significand or <i>coefficient</i>
Q	quotient
$q[j + 1]$	q after $(j + 1)$ -th iteration
QDS^*	QDS function without the PR sign detector
$q_{H_{j+1}}$	the most significant part of a radix-16 quotient digit
q_{j+1}	$(j + 1)$ -th quotient digit

Q_{j+1}	q in the 2's complement representation, after q_{j+1} is selected
$q_{L_{j+1}}$	least significant part of a radix-16 quotient digit
r	radix
R	round bit (rounding)
rem	remainder
ρ	redundancy factor
S	sign (IEEE 754 standard)
S	sticky bit (rounding)
s_D	dividend sign
$Sign(k)$	sign of SD number k
s_k	separating point
S_k	sign of SD number k
s_Q	quotient sign
$S_{rw[j]}$	shifted PR sign
s_X	dividend sign
t	dimension related to number of wiring tracks (parallel-prefix taxonomy)
τ	inverter delay with an identical inverter as the load
t_{in}	transfer digit from the adjacent addition position in the right
t_{out}	transfer digit sent to the adjacent addition position in the left
U_k	continuity condition upper bound
ulp	unit of last position
$w[j + 1]$	$(j + 1)$ -th PR
w_{INT}	intermediate PR (radix-16 division)
x	dividend significand or <i>coefficient</i>
X	dividend
$\langle y \rangle$	y -th digit/bit
$\lfloor y \rfloor$	the largest integer smaller than or equal to y
$\lceil y \rceil$	the smallest integer larger than or equal to y
$\neg y$	inverted y (y is a bit vector)
\bar{y}	$-y$ (y is a digit)
z^C	9's complemented z
lsf	digits involved in the least significant formation (radix-16 division)
$\{z\}_x$	number z truncated to x fractional digits/bits

List of Abbreviations

BB-DCFA	A DCFA with BCD addend and augend
BS	Borrow Save
BSD	Binary SD
BUF	Buffer
CAD	Computer Aided Design
CFA	Carry-Free Adder
CLA	Carry Look Ahead
CMOS	Complementary Metal Oxide Semiconductor
CR	Convert and Round
CRN	Convert, Round and Normalise
CS	Carry-Save
DB-DCFA	A DCFA with DSD addend and BCD augend
DCFA	Decimal Carry-Free Adder
DD-DCFA	A DCFA with DSD addend and augend
DFP	Decimal Floating-Point
DSD	Decimal Signed-Digit
DSD2BCD	DSD to BCD
DSP	Digital Signal Processor
EDA	Electronic Design Automation
FP	Floating-Point
FPU	FP Unit
FRFU	Final Result Formation Unit
FRSU	Final Result Selection Unit
IEEE	Institute of Electrical and Electronics Engineers
MRC	Multilevel Reverse-Carry
MUX	Multiplexer

PLA	Programmable Logic Array
PR	Partial Remainder
QDS	Quotient Digit Selection
RHE	Round-Half-Even
RTL	Register Transfer Language
RTNE	Round To Nearest Even
SD	Signed-Digit
SRT	Type of non-restoring digit recurrence division named after D. Sweeney, J. E. Robertson and T. D. Tocher
TBSU	Transfer Bit Selection Unit
TDSU	Transfer Digit Selection Unit
VLSI	Very Large Scale Integration