Section 10

Computer Arithmetic

Slides with white background courtesy of Mano text for this class

1

Digital Hardware Algorithms

- Arithmetic operations
  - Addition, subtraction, multiplication, division
- Data types
  - Fixed-point binary
    - Signed-magnitude representation
    - Signed-2's complement representation
  - Floating-point binary
  - Binary-coded decimal (BCD)

2

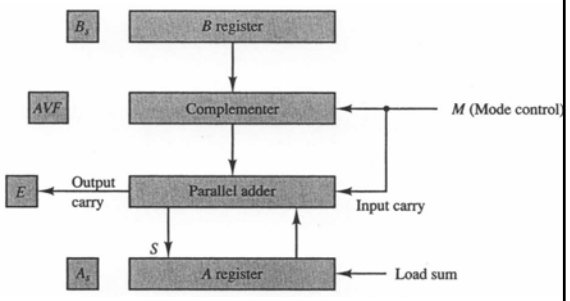
Add / Subtract Signed-Magnitude

Operation	Add Magnitudes	Subtract Magnitudes		
		When $A > B$	When $A < B$	When $A = B$
$(+A) + (+B)$	$+(A + B)$			
$(+A) + (-B)$		$+(A - B)$	$-(B - A)$	$+(A - B)$
$(-A) + (+B)$		$-(A - B)$	$+(B - A)$	$+(A - B)$
$(-A) + (-B)$	$-(A + B)$			
$(+A) - (+B)$		$+(A - B)$	$-(B - A)$	$+(A - B)$
$(+A) - (-B)$	$+(A + B)$			
$(-A) - (+B)$	$-(A + B)$			
$(-A) - (-B)$		$-(A - B)$	$+(B - A)$	$+(A - B)$

Forces zero to be positive

3

Hardware



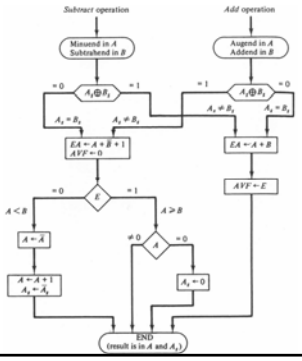
4

Description

- A_S Sign of A
- B_S Sign of B
- A_S & A Accumulator
- AVF Overflow bit for $A + B$
- E Output carry for parallel adder

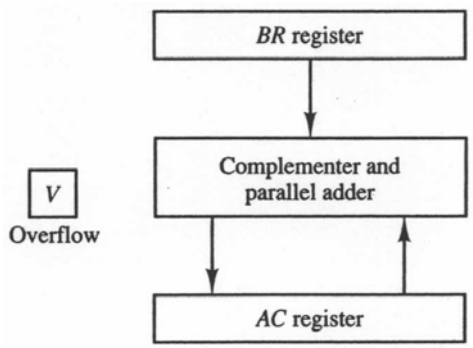
5

Flowchart



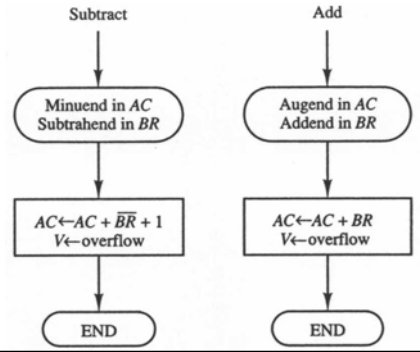
6

Add / Sub Signed-2's Complement



7

Algorithm



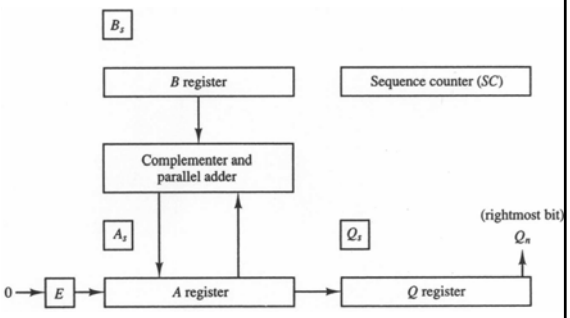
8

Multiply Signed-Magnitude

- Series of successive shift and add operations
- 23 10111 Multiplicand
  19 x 10011 Multiplier
  10111
  10111
  00000
  00000
  10111 +
  437 110110101 Product

9

Hardware



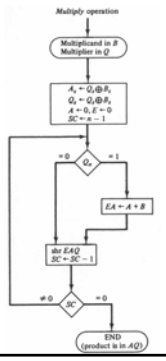
10

Description

- *Q* multiplier
- *B* multiplicand
- *A* 0
- *SC* number of bits in multiplier
- *E* overflow bit for *A*
- Do *SC* times
  - If low-order bit of Q is 1
    - $A \leftarrow A + B$
  - Shift right *EAQ*
- Product is in *AQ*

11

Flowchart



12

Example: 23 x 19 = 437

Multiplicand B = 10111	E	A	Q	SC
Multiplier in Q	0	00000	10011	101
$Q_n = 1$; add B		10111		
First partial product	0	10111		
Shift right EAQ	0	01011	11001	100
$Q_n = 1$; add B		10111		
Second partial product	1	00010		
Shift right EAQ	0	10001	01100	011
$Q_n = 0$; shift right EAQ	0	01000	10110	010
$Q_n = 0$; shift right EAQ	0	00100	01011	001
$Q_n = 1$; add B		10111		
Fifth partial product	0	11011		
Shift right EAQ	0	01101	10101	000
Final product in AQ = 0110110101				

13

Multiply Signed-2's Complement

- Booth algorithm
- QR multiplier
- Q_n least significant bit of QR
- Q_{n+1} previous least significant bit of QR
- BR multiplicand
- AC 0
- SC number of bits in multiplier

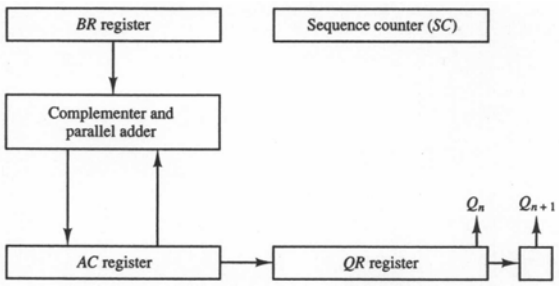
14

Algorithm

- Do $SC + 1$ times
  - $Q_nQ_{n+1} = 10$
    - $AC \leftarrow AC + \overline{BR} + 1$
  - $Q_nQ_{n+1} = 01$
    - $AC \leftarrow AC + BR$
  - Arithmetic shift right AC & QR
  - $SC \leftarrow SC - 1$

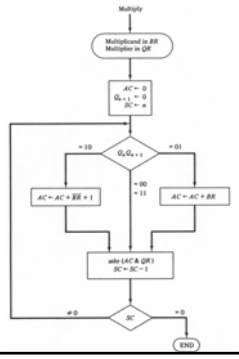
15

Hardware



16

Flowchart



17

Example: -9 x -13 = 117

Q_nQ_{n+1}	BR = 10111 $\overline{BR} + 1$ = 01001	AC	QR	Q_{n+1}	SC
	Initial	00000	10011	0	101
1 0	Subtract BR	01001			
		01001			
	ashr	00100	11001	1	100
1 1	ashr	00010	01100	1	011
0 1	Add BR	10111			
		11001			
	ashr	11100	10110	0	010
0 0	ashr	11110	01011	0	001
1 0	Subtract BR	01001			
		00111			
	ashr	00011	10101	1	000

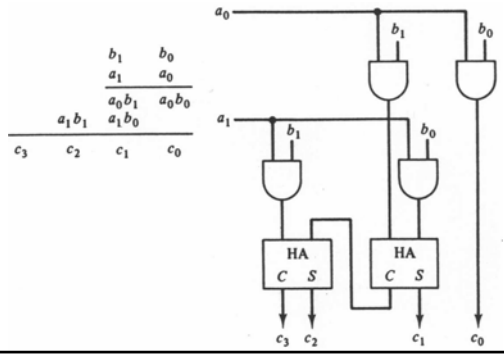
18

Array Multiplier

- Combination circuit
- Product generated in one microoperation
- Requires large number of gates
- Became feasible after integrated circuits developed
- Needed for *j* multiplier and *k* multiplicand bits
  - *j* x *k* AND gates
  - *j* – 1 *k*-bit adders to produce product of *j* + *k* bits

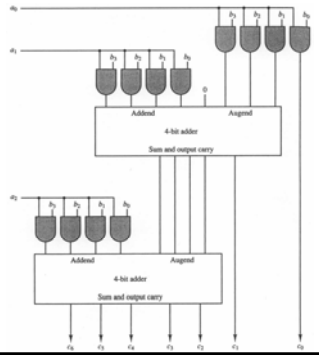
19

2-bit by 2-bit Array Multiplier



20

4-bit by 3-bit Array Multiplier



21

Divide Fixed-Point Signed-Mag

- Series of successive compare, shift, and subtract operations

Divisor:	11010	Quotient = Q
B = 10001	)0111000000	Dividend = A
	01110	5 bits of $A < B$, quotient has 5 bits
	011100	6 bits of $A \geq B$
	-10001	Shift right B and subtract; enter 1 in Q
	-010110	7 bits of remainder $\geq B$
	--10001	Shift right B and subtract; enter 1 in Q
	--001010	Remainder $< B$; enter 0 in Q; shift right B
	---010100	Remainder $\geq B$
	----10001	Shift right B and subtract; enter 1 in Q
	----000110	Remainder $< B$; enter 0 in Q
	-----00110	Final remainder

22

Example: 448 / 17 = 26 r 6

Divisor B = 10001, $\overline{B}$ + 1 = 01111

	E	A	Q	SC
Dividend:		01110	00000	5
shl EAQ	0	11100	00000	
add $\overline{B}$ + 1		01111		
E = 1	1	01011		
Set Q_n = 1	1	01011	00001	4
shl EAQ	0	10110	00010	
Add $\overline{B}$ + 1		01111		
E = 1	1	00101		
Set Q_n = 1	1	00101	00011	3
shl EAQ	0	01010	00110	
Add $\overline{B}$ + 1		01111		
E = 0; leave Q_n = 0	0	11001	00110	
Add B		10001		
Restore remainder	1	01010		2
shl EAQ	0	10100	01100	
Add $\overline{B}$ + 1		01111		
E = 1	1	00011		
Set Q_n = 1	1	00011	01101	1
shl EAQ	0	00110	11010	
Add $\overline{B}$ + 1		01111		
E = 0; leave Q_n = 0	0	10101	11010	
Add B		10001		
Restore remainder	1	00110	11010	0
Neglect E				
Remainder in A:		00110		
Quotient in Q:			11010	

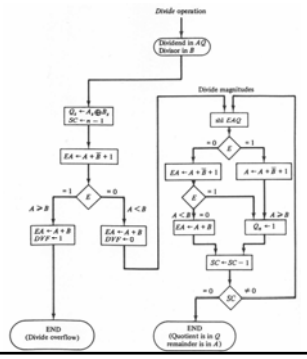
Figure 10-12 Example of binary division with digital hardware.

Initially,
AQ dividend
B divisor

At end of operation,
Q quotient
A remainder
DVF divide overflow

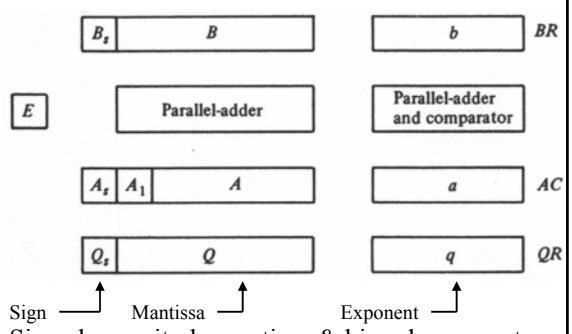
23

Algorithm



24

Floating-Point Registers



Signed-magnitude mantissa & biased exponent

25

Biased Exponent

- Example
  - Real exponent range is -50 to +49
  - Add bias of 50 for new range of 0 to 99
  - Biased exponent is always a positive number
    - Easier to deal with

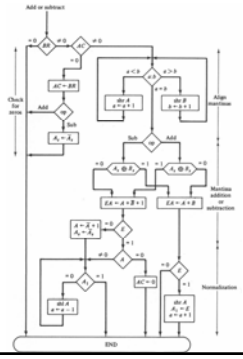
26

Floating-Point Add / Subtract

- Check for zeros
- Align the mantissas
- Add or subtract the mantissas
- Normalize the result

27

F-P Add / Subtract Flowchart



$AC \leftarrow AC + BR$
or
$AC \leftarrow AC - BR$

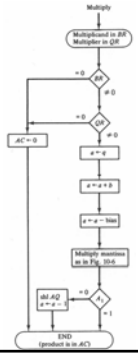
28

Floating-Point Multiply

- Check for zeros
- Add the exponents
- Multiply the mantissas
- Normalize the product

29

F-P Multiply Flowchart



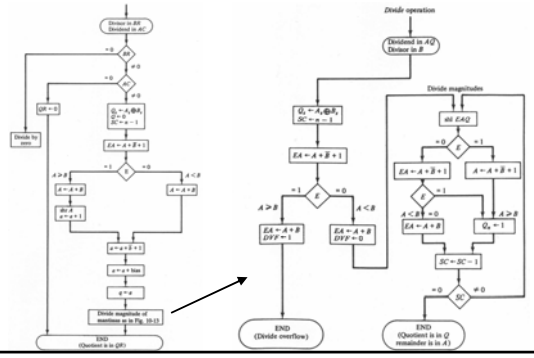
30

Floating-Point Division

- Check for zeros
- Initialize registers and evaluate the sign
- Align the dividend
- Subtract the exponents
- Divide the mantissas

31

F-P Division Flowchart



32

33

34

35

Booth Multiplication Algorithm

- Zeros in multiplier require no addition
  - But shifting still required
- String of 1s in the multiplier from weight 2^k to 2^m can be rewritten as $2^{k+1} - 2^m$
  - Example: 001110 [+14]
    - String of 1s from 2^3 to 2^1: $2^4 - 2^1 = 16 - 2 = 14$
    - Multiplicand M: $M \times 14 = M \times 2^4 - M \times 2^1$
    - Product obtained by M 4 times to the left and subtracting M shifted left once

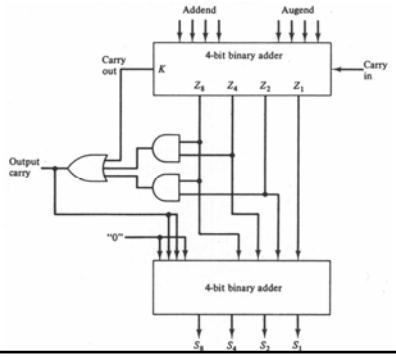
36

BCD Adder

- Output can't exceed 9 + 9 + 1 = 19
- If binary sum in BCD digit > 1001, add 0110
- Given
  - Output of binary adder is $Z_8Z_4Z_2Z_1$
  - Output carry K
  - BCD output carry $C = K + Z_8Z_4 + Z_8Z_2$

37

Block Diagram BCD Adder



38

Examples

9	1001	9	1001	6	0110
7	0111	9	1001	4	0100
16	1 0000	18	1 0010	10	1010
	0110		0110		0110
	0110		1000		1 0000

39

BCD Subtraction

- Subtract by adding 9s complement of subtrahend to minuend
- First 9s complement algorithm
  - Complement bits
  - Add 1010 (decimal 10) and discard carry
- Second 9s complement algorithm
  - Add 0110 (decimal 6)
  - Complement bits

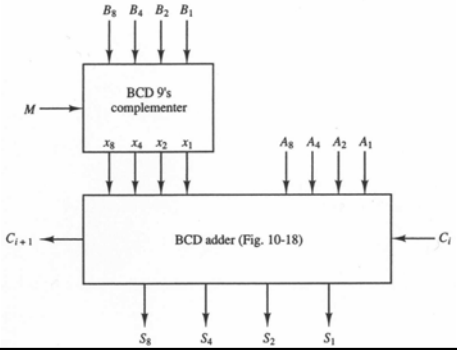
40

Examples

0111 decimal 7	0111
1000 complement	+ 0110 add decimal 6
+ 1010 decimal 10	1101
1 0010 decimal 2	0010 complement

41

Stage of Decimal Arithmetic Unit



42

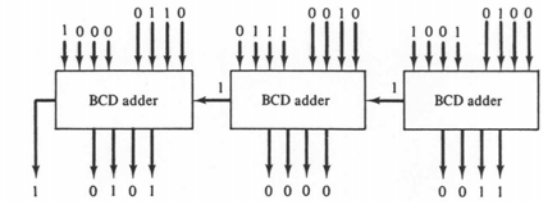
Decimal Arithmetic Microops

TABLE 10-5 Decimal Arithmetic Microoperation Symbols

Symbolic Designation	Description
$A \leftarrow A + B$	Add decimal numbers and transfer sum into A
$\overline{B}$	9's complement of B
$A \leftarrow A + \overline{B} + 1$	Content of A plus 10's complement of B into A
$Q_L \leftarrow Q_L + 1$	Increment BCD number in Q_L
dshr A	Decimal shift-right register A
dshl A	Decimal shift-left register A

43

Parallel Decimal Addition

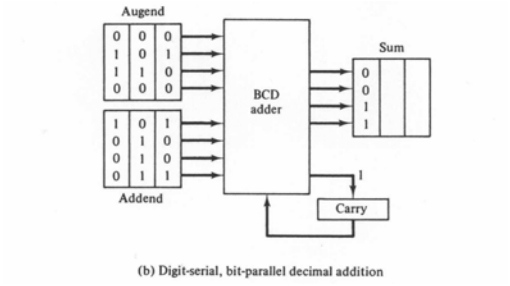


(a) Parallel decimal addition: 624 + 879 = 1503

Fast

44

Digit-Serial, Bit-Parallel Dec Add

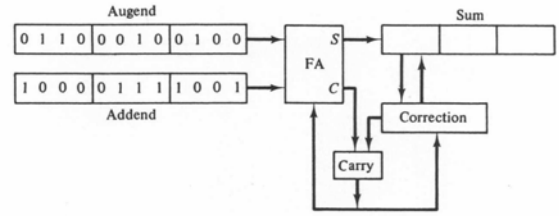


(b) Digit-serial, bit-parallel decimal addition

Slow

45

All Serial Decimal Addition

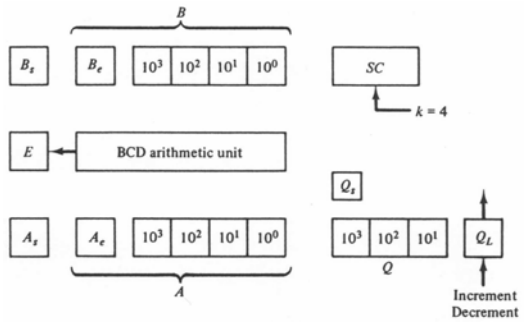


(c) All serial decimal addition

Very slow

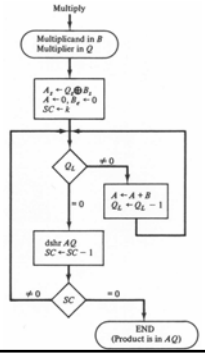
46

Dec Arith Registers for Mult & Div



47

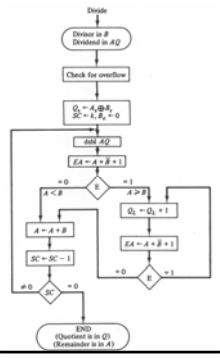
Decimal Multiplication Flowchart



0456	0456	
x 0123	0456	
56088	+ 0456	
	1368	
	0136 8	shift
	0456	
	+ 0456	
	1048	
	0104 88	shift
	+ 0456	
	0560	
	0056 088	shift
	0005 6088	shift

48

Decimal Division Flowchart



49