Qassim University

Department of Computer Engineering

COE223: Computer architecture

Chap 1, Exercises

Exercise 1:

What is the delay for the following types of 64-bit adders? Assume that each two-input gate delay is 150 ps and that a full adder delay is 450 ps.

(a) a ripple-carry adder

(b) a carry-lookahead adder with 4-bit blocks

(c) a prefix adder

Correction 1:

Compare delay of: 64-bit ripple-carry, carry-lookahead, and prefix adders

CLA has 4-bit blocks

2-input gate delay = 150 ps; full adder delay = 450 ps

t_{ripple} = NtFA = 64(450 ps)

= 28.8 ns

t_{CLA} = tpg + tpg_block + (N/k – 1)tAND_OR + ktFA

= [150 + 900 + (15)300 + 4(450)] ps

= 7.35 ns

t_{PA} = tpg + log2N(tpg_prefix) + tXOR

= [150 + $\log_2$46(300) + 150] ps

= 2.1 ns

Exercise 2:

1. Express the following base 10 numbers in 16-bit fixed-point sign/ magnitude format with eight integer bits and eight fraction bits. Express your answer in hexadecimal.

   (a) −13.5625

   (b) 42.3125

2. Convert the following two's complement binary fixed-point numbers to base 10. The implied binary point is explicitly shown to aid in your interpretation.

   (a) 0101.1000

   (b) 1111.1111

Correction 2:

1.

(a) −13.5625

$(13)_{10} = (1101)_2$.
Since we need 8 bits for the integer part:
00001101
The fractional part 0.5625 can be converted into binary.
0.5625×2=1.125→1
0.125×2=0.25→0
0.25×2=0.5→0
0.5×2=1.0→1
So, (0.5625) = (0.1001)

Now, express this as an 8-bit binary fraction:
10010000

For the magnitude (without the sign):
00001101.10010000

The final result in sign/magnitude format is:
10001101.10010000

Convert to hexadecimal
$(1000_1101_1001_0000)_2 = (8D90)_{16}$.

b) 42.3125

Since the number is positive, the MSB will be 0.

42.3125=42+0.3125
$(42)_{10}=(101010)_2$
We need 8 bits, so:
00101010
Convert.3125 to binary:
0.3125×2=.625→0
0.625×2=1.25→1
0.25×2=0.5→0
0.5×2=1.0→1
So, $(0.3125)_{10}=(0.0101)_2$

Then we have; 0 0101010.01010000

$(0010_1010_0101_0000)_2=(2A50)_{16}$.

2.

(a) 0101.1000

$(0101)_2=(5)_{10}$

$(0.1000)_2=1\times2^{-1}=0.5$.

Answer: 5.5

(b) 1111.1111

1111.1111 in two's complement is a negative number (since the MSB is 1).

Invert all bits: 0000.0000
Add 0000.0001
$(0000.0001)_2=(0.0625)_{10}$.
Since the number is negative, the value is −0.0625

Answer: -0.0625

Exercise 3:

Add the following IEEE 754 single-precision floating-point numbers:

40123400+ 41C564B7

Correction 3:

Step 1: Convert the hexadecimal numbers to binary

1. **40123400** in binary: `0100 0000 0001 0010 0011 0100 0000 0000`
2. **41C564B7** in binary: `0100 0001 1100 0101 0110 0100 1011 0111`

Step 2: Extract the components (sign, exponent, mantissa)

For 40123400:

- **Sign**: `0` (positive)
- **Exponent**: `10000000` (128 in decimal, biased exponent)
  - Unbiased exponent = 128−127=1128 - 127 = 1128−127=1
- **Mantissa**: `00100100011010000000000`

For 41C564B7:

- **Sign**: `0` (positive)
- **Exponent**: `10000011` (131 in decimal, biased exponent)
  - Unbiased exponent = 131−127=4
- **Mantissa**: `10001010110010010110111`

Step 3: Align the exponents

- **Exponent of 40123400**: 1
- **Exponent of 41C564B7**: 4

To align the exponents, we need to shift the mantissa of **40123400** to the **right by 3 positions** (the difference between 4 and 1):

- Mantissa of **40123400** after shifting:
  1.00100100011010000000000⇒0.00100100100011010000000

Step 4: Add the mantissas

After aligning the exponents, the mantissas are:

- Mantissa of **40123400** (after shift): `0.00100100100011010000000`
- Mantissa of **41C564B7**: `1.10001010110010010110111`

Adding these two mantissas in binary:

0 . 0 0 1 0 0 1 0 0 1 0 0 0 1 1 0 1 0 0 0 0 0 0 0+
1 . 1 0 0 0 1 0 1 0 1 1 0 0 1 0 0 1 0 1 1 0 1 1 1

=1 . 1 0 1 0 1 1 1 1 0 1 0 1 0 1 1 0 0 1 1 0 1 1 1

Step 5: Normalize the result

The result is already normalized, so we keep:

- **Mantissa**: `1.10101111001100100110111`
- **Exponent**: 4 (remains unchanged)

Step 6: Rebuild the number in IEEE 754 format

We have:

- **Sign**: 0 (positive)
- **Exponent**: 4 → 4+127=131which in binary is: `10000011`
- **Mantissa**: `10101111010101100110111`

In IEEE 754 format, this gives:

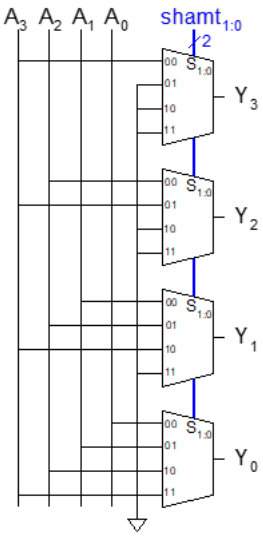
0 10000011 10101111010101100110111

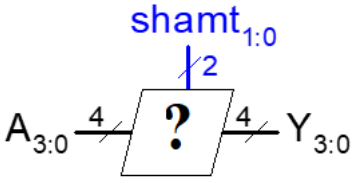
Conversion to hexadecimal

The final result in hexadecimal is **41D7AB37.**

Exercise 4:

Determine the value of Y and the type of shifter if the value of shamt is $(10)_2$, $(11)_2$.





Correction 4 :

If shamt=’10 ‘ then Y=0 0 A_3A_2→Logical shifter right >>2

If shamt=’11 ‘ then Y=0 0 0 A_3→Logical shifter right >>3