Floating Point Addition/ Subtraction

- 35.23142 + 0.00053

X = 35.23142

X (Binary)
= 100011.0011101111

X (Binary Normalized) =
$1.000110011101111 \times 2^{5}$

Y = 0.00053

Y (Binary)
= 0.0000000000010001011

Y (Binary Normalized) =
1.0001011×2^{-11}

Y (Binary) =
$0.00000000000000010001011 \times 2^{5}$

Rule: Match the Lower Exponent with the Higher Exponent

X + Y = $(1.000110011101111 \times 2^{5}) + (0.00000000000000010001011 \times 2^{5})$

$= (1.000110011101111 + 0.00000000000000010001011) \times 2^{5}$

$= 1.00011001110111110001011 \times 2^{5}$

$= 100011.001110111110001011$

= 35.2342224121 **(Decimal)**

Floating Point Multiplication

- 5.234 x (-0.003)

X = 5.234

X (Binary)
= 101.0011101111

X (Binary Normalized) =
$1.010011101111 \times 2^{2}$

Y = 0.003

Y (Binary)
= 0.0000000011000100101

Y (Binary Normalized) =
$1.1000100101 \times 2^{-9}$

X x Y = $-(1.010011101111 \times 2^{2}) \times (1.1000100101 \times 2^{-9})$

$= -(1.010011101111 \times 1.1000100101) \times 2^{(2+(-9))}$

$= -(1.010011101111 \times 1.1000100101) \times 2^{(-7)}$

$= -10.000000101 \times 2^{(-7)}$

$= -0.0000010000000101 \times 2^{(0)}$

= - 0.0157012939 (Decimal)

Floating Point Arithmetic

- 51500000 – BA10A000

(Biased)

Sign bit	Exponent	Fraction/ Mantissa
1 bit	8 bit	23 bit

X = 51500000

X (Binary)
= 0101 0001 0101 0000 0000 0000 0000 0000 [32 bit]
= 0 10100010 10100000000000000000000

Find Out Exponent and Fraction

Biased Exponent = 10100010

Biased Exponent (Decimal)= 162 For Exponent being 8 bit, Bias = $2^{(8-1)}-1 = 127$

Exponent (Decimal)= 162 - 127
= 35

Fraction/ Mantissa = 0. 10100000000000000000000

X (Binary Normalized) = 1.Franction x $2^{(Exponent)}$

X (Binary Normalized) = 1. 10100000000000000000000 x 2^{35} *Sign bit = 0 (Positive)

Floating Point Arithmetic

- 51500000 – BA10A000

(Biased)

Sign bit	Exponent	Fraction/ Mantissa
1 bit	8 bit	23 bit

Y = BA10A000

Y (Binary)
= 1011 1010 0001 0000 1010 0000 0000 0000 [32 bit]
= 1 01110100 00100001010000000000000

Find Out Exponent and Fraction

Biased Exponent = 01110100

Biased Exponent (Decimal)= 116 For Exponent being 8 bit, Bias = $2^{(8-1)}-1 = 127$

Exponent (Decimal)= 116 - 127
= -11

Fraction/ Mantissa = 0. 00100001010000000000000

Y (Binary Normalized) = 1.Franction x $2^{(Exponent)}$

Y (Binary Normalized) = - 1.00100001010000000000000 x 2^{-11} *Sign bit = 1 (Negative)

Floating Point Addition/ Subtraction

- 51500000 – BA10A000

Rule: Match the Lower Exponent with the Higher Exponent

X = 51500000

X (Binary Normalized) =
1. 10100000000000000000000 x 2^{35}

Y = BA10A000

Y (Binary Normalized) =
- 1.00100001010000000000000 x 2^{-11}

Y (Binary Normalized) =
- 0.[45 0s..] 10010000101000000000000 x 2^{35}

X – (-Y) =
X + Y =

= (1. 10100000000000000000000 x 2^{35}) + (0.[45 0s..] 10010000101000000000000 x 2^{35})

= (1. 10100000000000000000000 + 0.[45 0s..] 10010000101000000000000) x 2^{35}

= 1.101[42 0s..] 10010000101000000000000 x 2^{35}

Floating Point Arithmetic

- 7ACD0000 + 5BCA0000

(Biased)

Sign bit	Exponent	Fraction/ Mantissa
1 bit	8 bit	23 bit

X = 7ACD0000

X (Binary)
= 0111 1010 1100 1101 0000 0000 0000 0000 [32 bit]
= 0 11110101 10011010000000000000000

Find Out Exponent and Fraction

Biased Exponent = 11110101

Biased Exponent (Decimal)= 245 For Exponent being 8 bit, Bias = $2^{(8-1)}$ -1 = 127

Exponent (Decimal)= 245 - 127
= 118

Fraction/ Mantissa = 0.10011010000000000000000

X (Binary Normalized) = 1.Franction x $2^{(Exponent)}$

X (Binary Normalized) = 1. 10011010000000000000000 x 2^{118} *Sign bit = 0 (Positive)

Floating Point Arithmetic

- 7ACD0000 + 5BCA0000

(Biased)

Sign bit	Exponent	Fraction/ Mantissa
1 bit	8 bit	23 bit

Y =5BCA0000

Y (Binary)
= 0101 1011 1100 1010 0000 0000 0000 0000 [32 bit]
= 0 10110111 10010100000000000000000

Find Out Exponent and Fraction

Biased Exponent = 10110111

Biased Exponent (Decimal)= 183 For Exponent being 8 bit, Bias = $2^{(8-1)}-1 = 127$

Exponent (Decimal)= 183 - 127
= 56

Fraction/ Mantissa = 0. 10010100000000000000000

Y (Binary Normalized) = 1.Franction x $2^{(Exponent)}$

Y (Binary Normalized) = 1. 10010100000000000000000 x 2^{56}

Floating Point Addition/ Subtraction

- 7ACD0000 + 5BCA0000

Rule: Match the Lower Exponent with the Higher Exponent

X = 51500000

X (Binary Normalized) =
1. 10011010000000000000000 x 2^{118}

Y = 3A10A000

Y (Binary Normalized) =
1. 10010100000000000000000 x 2^{56}

Y (Binary Normalized) =
0.[61 0s..] **10010100000000000000000** x 2^{118}

X + Y =

=1. 10011010000000000000000 x 2^{118} +0.[61 0s..] 10010000101000000000000 x 2^{118}

= (**1. 10011010000000000000000** + **0.[61 0s..] 10010000101000000000000**) x 2^{118}

= 1.1001101[54 0s..] **1**0010000101000000000000 x 2^{118}

8. Suppose X=19.454 and Y=3.0124, perform X*Y using IEEE floating-point representation.

Answer8:

X= 19.454
X (Binary) = 10011.01110100
X (Normalized) = $1.001101110100 \times 2^4$
Y= 3.012
X (Binary) = 11.0000001100
X (Normalized) = 1.10000001100×2^1

X * Y =
$(1.001101110100 \times 2^4) \times (1.10000001100 \times 2^1)$
$= (1.001101110100 \times 1.10000001100) \times 2^{4+1}$

$= 1.11010100101100101110000 \times 2^5$
= 111010.100101100101110000
= 58.58734...(Decimal)

```
  1.001101110100
x 1.10000001100
-----------------------------
  0000000000000
 1001101110100xx
1001101110100xxx
1001101110100xxxxxxxxxx
1001101110100xxxxxxxxxxx
-----------------------------
1.11010100101100101110000
```

Practice

Consider the value 63.7813

a) Let's assume you have a 21-bit register having 6-bit for exponent. Now convert this value using IEEE floating-point representation. Also, convert this into hexadecimal form.
Ans: 49FC8

b) Let's assume you have a 12-bit register having 4-bit for exponent. Now convert this value using IEEE floating-point representation. Also, convert this into hexadecimal form
Ans: 67F

c) Suppose X= -9.435 and Y= 15.129, perform X*Y using IEEE floating-point representation.
Ans: -142.719955... (Decimal)

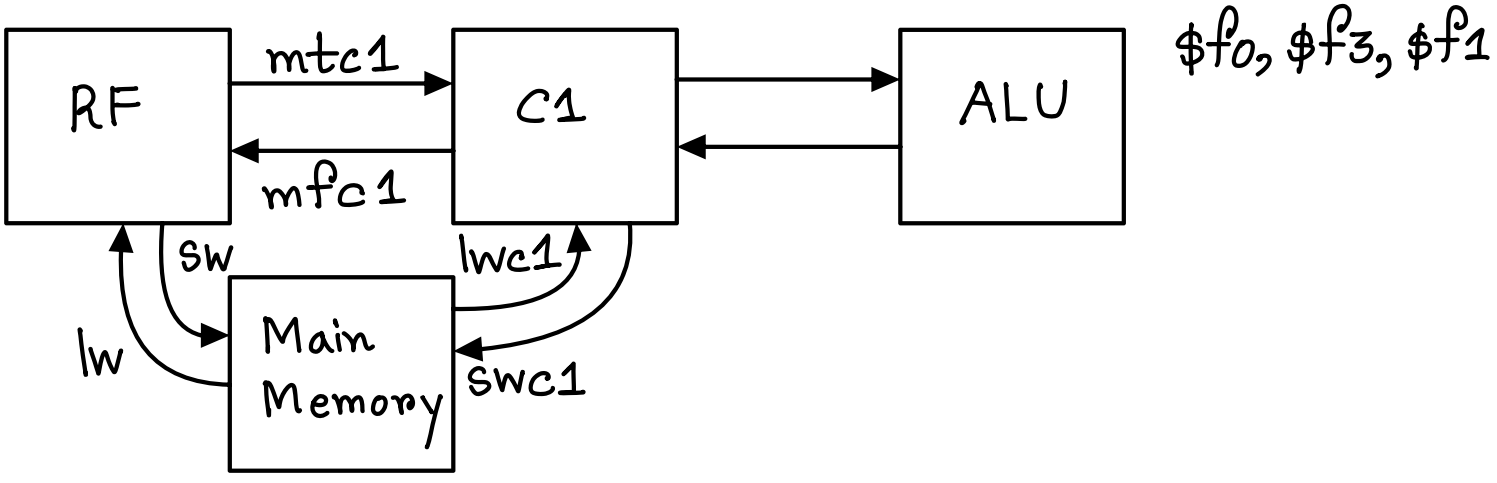
MIPS Division

- Use HI/LO registers for result
  - HI: 32-bit remainder
  - LO: 32-bit quotient
- Instructions
  - div rs, rt / divu rs, rt
  - No overflow or divide-by-0 checking
    - Software must perform checks if required
  - Use mfhi, mflo to access result

Now Written Part!

Next Quiz: Chapter → 3

Floating point Registers and Instruction:



type

A[3] = (float) x

Where x is in $f0 and base address of A is in $s1.

swc1 $f0, 12($s1)

float (a) = B[2]

Where a is in $f0 and base address of B is in $s1.

lwc1 $f0, 8($s1)

Single Precision

$f0 = $f1 + $f2

add.s $f0, $f1, $f2

$f0 = $f1 - $f2

sub.s $f0, $f1, $f2

Double Precision

$f0 = $f2 + $f4

add.d $f0, $f2, $f4

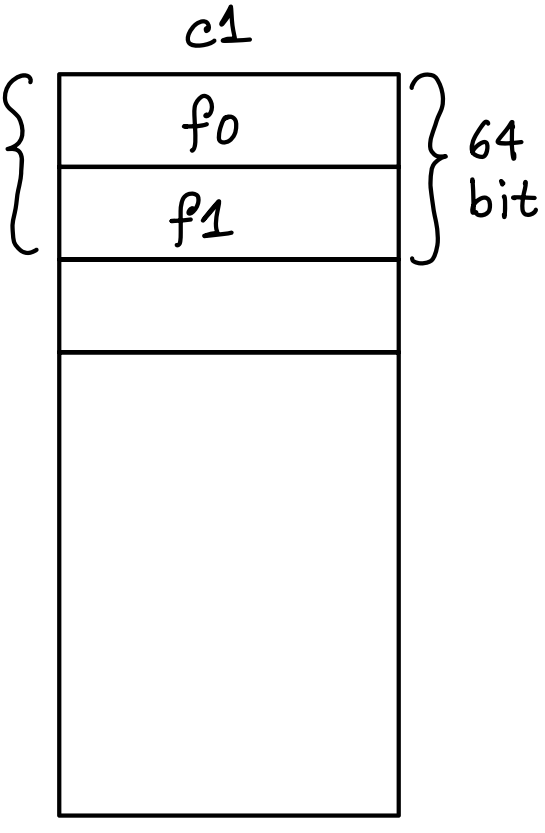
$f0 = $f2 - $f4

sub.d $f0, $f2, $f4

Moving values from register

mtc1 $s0, $f1

mfc1 $s0, $f1



Convert a register value from float to integer for single precision: cvt.w.s $f0, $f1

float

Convert a register value from integer to float for single precision:

cvt.s.w $f0, $f1

int

Double precision to single precision

cvt.s.d $f0, $f1

Single precision to double precision

cvt.d.s $f0, $f1

float(x) = (int)y + float(z)

(int)y = (float)x - (float)z

Suppose, x, y, z are in $f1, $s1, and $f2.

Mips Code:

```
mtc1 $s1, $f3
cvt.s.w $f3, $f3 (Possible)
add.s $f1, $f3, $f2
sub.s $f4, $f1, $f2
cvt.w.s $f4, $f4
mtc1 $s1, $f4
```