Hardware - Tutorial 1 - Correction
Two's Complement and Floating point representation

Exercice 1: Signed Numbers

Encode the following numbers into 8 bits signed binary.

- $-1 =$ 1111 1111
- $-29 =$ 1110 0011
- $-42 =$ 1101 0110
- $-127 =$ 1000 0001
- $-128 =$ 1000 0000
- $-175 =$ Impossible in 8 bits binary

Exercice 2: Signed Operations

Perform the following 8 bits binary operations. Give the result in 8 bits binary and convert it to decimal given the context is signed or unsigned. If an overflow occurs, write down 'ERROR' in the corresponding cell.

Operation	**Binary Result**	**Decimal Value**	
		Unsigned	**Signed**
1111 0101 + 1111 1010	1110 1111	ERROR	-17
1110 1000 - 1100 0110	0010 0010	34	34
0101 1110 - 1001 1110	1100 0000	ERROR	ERROR
0111 1110 + 0000 0101	1000 0011	131	ERROR
1100 1011 - 0001 1010	1011 0001	177	-79
1000 0000 + 1111 1010	0111 1010	ERROR	ERROR
1000 0011 - 0000 1010	0111 1001	121	ERROR

Exercice 3: Decimal to float

Convert the following decimal numbers to their binary **single-precision** floating point representation:

- 128
- -32.75
- 18.125
- 0.0625

Solution:

1) $N = 128$

- $S = 0$
- $m = |128_{10}| = 128_{10} = 10000000_2$
- $\mathbf{M} = 0$
- $e = 7$
- $\mathbf{E} = e + biais = 134_{10}$
- $\mathbf{E} = 10000110_2$
- $128_{10} = 0\ 10000110\ 00000000000000000000000_f$

2) $N = -32.75$

- $S = 1$
- $m = |-32.75_{10}| = 32.75_{10} = 100000.11_2$
- $\mathbf{M} = 0000011$
- $e = 5$
- $\mathbf{E} = e + biais = 132_{10}$
- $\mathbf{E} = 10000100_2$
- $-32.75_{10} = 1\ 10000100\ 00000110000000000000000_f$

3) $N = 18.125$

- $S = 0$
- $m = |18.125_{10}| = 18.125_{10} = 10010.001_2$
- $\mathbf{M} = 0010001$
- $e = 4$
- $\mathbf{E} = e + biais = 131_{10}$
- $\mathbf{E} = 10000011_2$
- $18.125_{10} = 0\ 10000011\ 00100010000000000000000_f$

4) $N = 0.0625$

- $S = 0$
- $m = |0.0625_{10}| = 0.0625_{10} = 0.0001_2$
- $\mathbf{M} = 0$
- $e = -4$
- $\mathbf{E} = e + biais = 123_{10}$
- $\mathbf{E} = 01111011_2$
- $0.0625_{10} = 0\ 01111011\ 00000000000000000000000_f$

Exercice 4: Float to decimal

Convert the following **single-precision** floating point numbers to decimal representation:

- 1011 1101 0100 0000 0000 0000 0000 0000
- 0101 0101 0110 0000 0000 0000 0000 0000
- 1100 0001 1111 0000 0000 0000 0000 0000
- 1111 1111 1000 0000 0000 0000 0000 0000
- 0000 0000 0100 0000 0000 0000 0000 0000

Solution:

1) $F = 1011\ 1101\ 0100\ 0000\ 0000\ 0000\ 0000\ 0000$

- $\mathbf{S} = 1$
- $\mathbf{E} = 01111010$
- $\mathbf{M} = 1$
- $m = 1.M = 1.1$
- $e = E - biais = 122 - 127 = -5$
- $\mathbf{N} = -1 \times 1.1_2 \times 2^{-5} = -1 \times 3_{10} \times 2^{-6}$
- $\boxed{\mathbf{N} = -0.046875}$

2) $F = 0101\ 0101\ 0110\ 0000\ 0000\ 0000\ 0000\ 0000$

- $\mathbf{S} = 0$
- $\mathbf{E} = 10101010$
- $\mathbf{M} = 11$
- $m = 1.M = 1.11$
- $e = E - biais = 170 - 127 = 43$
- $\mathbf{N} = 1 \times 1.11_2 \times 2^{43} = 1 \times 111_2 \times 2^{41}$
- $\boxed{\mathbf{N} = 7 \times 2^{41} \approx 1.53 \times 10^{13}}$

3) $F = 1100\ 0001\ 1111\ 0000\ 0000\ 0000\ 0000\ 0000$

- $\mathbf{S} = 1$
- $\mathbf{E} = 10000011$
- $\mathbf{M} = 111$
- $m = 1.M = 1.111$
- $e = E - biais = 131 - 127 = 4$
- $\mathbf{N} = -1 \times 1.111_2 \times 2^4 = -1 \times 1111_2 \times 2^1$
- $\boxed{\mathbf{N} = -30}$

4) $F = 1111\ 1111\ 1000\ 0000\ 0000\ 0000\ 0000\ 0000$

- $\mathbf{S} = 1$
- $\mathbf{E} = 11111111$
- $\mathbf{M} = 0$
- $E = 255, M = 0 \rightarrow$ Infinity
- $\boxed{\mathbf{N} = -\infty}$

5) $F = 0000\ 0000\ 0100\ 0000\ 0000\ 0000\ 0000\ 0000$

- $\mathbf{S} = 0$
- $\mathbf{E} = 0$
- $\mathbf{M} = 1$
- $E = 0, M \neq 0 \rightarrow$ Denormalized Mantissa
- $m = 0.M = 0.1$
- $e = 1 - biais = -126$
- $\mathbf{N} = 1 \times 0.1_2 \times 2^{-126} = -1 \times 1_2 \times 2^{-127}$
- $\boxed{\mathbf{N} = 2^{-127} \approx 5.88 \times 10^{39}}$

Exercice 5: Decimal to double

Convert the following decimal numbers into their binary **double-precision** floating point representation:

- 1
- -64
- 12.06640625
- 0.2734375

Solution:

1) $N = 1$

- $S = 0$
- $m = |1_{10}| = 1_{10} = 1_2$
- $\mathbf{M} = 0$
- $e = 0$
- $\mathbf{E} = e + biais = 1023_{10}$
- $\mathbf{E} = 01111111111_2$
- $1_{10} = 0\ 01111111111\ 00....0_d$

2) $N = -64$

- $S = 1$
- $m = |-64_{10}| = 64_{10} = 1000000_2$
- $\mathbf{M} = 0$
- $e = 6$
- $\mathbf{E} = e + biais = 1029_{10}$
- $\mathbf{E} = 10000000101_2$
- $-64_{10} = 1\ 10000000101\ 00....0_d$

3) $N = 12.06640625$

- $S = 0$
- $m = |12.06640625_{10}| = 12.06640625_{10} = 1100.00010001_2$
- $\mathbf{M} = 10000010001$
- $e = 3$
- $\mathbf{E} = e + biais = 1026_{10}$
- $\mathbf{E} = 10000000010_2$
- $12.06640625_{10} = 0\ 10000000010\ 100000100010....0_d$

4) $N = 0.2734375$

- $S = 0$
- $m = |0.2734375_{10}| = 0.2734375_{10} = 0.0100011_2$
- $\mathbf{M} = 00011$
- $e = -2$
- $\mathbf{E} = e + biais = 1021_{10}$
- $\mathbf{E} = 01111111101_2$
- $0.2734375_{10} = 0\ 01111111101\ 000110....0_d$

Exercice 6: Double to decimal

Convert the following **double-precision** floating point numbers to decimal representation:

- 403D 4800 0000 0000
- C040 0000 0000 0000
- BFC0 0000 0000 0000
- 8000 0000 0000 0000
- FFF0 0001 0000 0000

Solution:

1) $D = 403D$ 4800 0000 0000

- $D_2 = 0100\ 0000\ 0011\ 1101\ 0100\ 1000...0$
- $\mathbf{S} = 0$
- $\mathbf{E} = 100\ 0000\ 0011$
- $\mathbf{M} = 1101\ 0100\ 1000...$
- $m = 1.M = 1.1101\ 0100\ 1000$
- $e = E - biais = 1027 - 1023 = 4$
- $\mathbf{N} = 1 \times 1.1101\ 0100\ 1000_2 \times 2^4 = 1110101001_2 \times 2^{-5}$
- $\boxed{\mathbf{N} = 937 \times 2^{-5}}$

2) $D = C040$ 0000 0000 0000

- $D_2 = 110000000100...0$
- $\mathbf{S} = 1$
- $\mathbf{E} = 100\ 0000\ 0100$
- $\mathbf{M} = 0...$
- $m = 1.M = 1.0$
- $e = E - biais = 1028 - 1023 = 5$
- $\mathbf{N} = 1 \times 1.0_2 \times 2^5$
- $\boxed{\mathbf{N} = -32}$

3) $D = BFC0$ 0000 0000 0000

- $D_2 = 1011\ 1111\ 1100...0$
- $\mathbf{S} = 1$
- $\mathbf{E} = 011\ 1111\ 1100$
- $\mathbf{M} = 0$
- $m = 1.M = 1.0$
- $e = E - biais = 1020 - 1023 = -3$
- $\mathbf{N} = 1 \times 1.0_2 \times 2^{-3}$
- $\boxed{\mathbf{N} = -0.125}$

4) $D = 8000$ 0000 0000 0000

- $D_2 = 1000...0$
- $\mathbf{S} = 1$
- $\mathbf{E} = 0$
- $\mathbf{M} = 0$
- $E = 0, M = 0 \rightarrow$ Zero
- $\boxed{\mathbf{N} = -0}$

5) $D = FFF0$ 0001 0000 0000

- $D_2 = 1111\ 1111\ 1111\ 0000\ 0000\ 0000\ 0000\ 0001...0$
- $\mathbf{S} = 1$
- $\mathbf{E} = 111\ 1111\ 1111$
- $\mathbf{M} = 0000\ 0000\ 0000\ 0000\ 0001$
- $E = 2047, M \neq 0 \rightarrow NaN$
- $\boxed{\mathbf{N} = NaN}$

Exercice 7: Float danger

Let us consider the following C program:

```c
void main() {
  float f1, f2, f3, r;

  f1 = 1E25; // f1 = 10^25
  f2 = 16;

  f3 = f1 + f2;
  r = f3 - f1;

  printf("r = %f\n", r);
}
```

Indication: $10^{25} \approx 2^{83}$

1) Once the program has run through, what will the value of r be ? Explain your reasoning.

- We set: $f1 = (1.M1)_2 \times 2^{e1}$
  From the indication, we can assume that $f1 = (1.0)_2 \times 2^{83}$
- We set: $f3 = (1.M3)_2 \times 2^{e3}$
- To compute the result of the addition, we need to put all the floating point numbers under the same exponent.
- Then we add the first 23 bits of the mantissas to get the mantissa of the result.

$$
\begin{array}{rl}
 & \overset{2^{83}}{\uparrow}\ \overset{2^{81}}{\uparrow} \qquad\qquad\qquad \overset{2^{60}}{\uparrow} \qquad \overset{2^{4}}{\uparrow} \\
 & 1.\mathbf{00000000000000000000000}00...000000 \times 2^{83} \\
+ & 0.\mathbf{00000000000000000000000}00...000100 \times 2^{83} \\
\hline
 & 1.\underbrace{\mathbf{00000000000000000000000}}_{\text{M3}}00...000100 \times 2^{83}
\end{array}
$$

As we can see here, the smallest value of $f2$ that can modify $f1$ is 2^{60}. Since $f2 = 2^4$, the $f1$ and $f3$ variables will be encoded the same way.

With that in mind, we can say that the addition $f3 = f1 + f2$ is the same as $f3 = f1$.
Then the substraction $r = f3 - f1$ becomes $r = f1 - f1$.

Once the program has run through, the value of r will be 0.

2) Assuming that $f1 = 10^n$ where n is a natural number, what is the largest value of n that still gives a correct value of r ?

Solution:

- The difference between $f1$ and $f2$ must be small enough so that $f3$ will be affected by the addition.
  Since $f2 = 16$, the largest weight of the least significant bit of M1 must be 2^4, because if this weight is greater than or equal to 2^5, f2 will be too small and the addition will not affect f3.

$$
\begin{array}{rl}
 & \overset{2^{28}}{\uparrow} \qquad\qquad\qquad\qquad \overset{2^{5}}{\uparrow} \\
 & 1.\mathbf{00000000000000000000000}00... \times 2^{28} \\
+ & 0.\mathbf{00000000000000000000000}00... \times 2^{28} \\
\hline
 & 1.\underbrace{\mathbf{00000000000000000000000}}_{\text{M3}}00... \times 2^{28}
\end{array}
$$

- Therefore, the variable $f1$ must be less than 2^{28} so that the addition will affect $f3$
- Wich gives:
  $f1 < 2^{28}$
  $10^n < 2^{28}$
  $n < Log(2^{28})$
  $n < 8.42$
  $\boxed{n_{max} = 8}$

3) Assuming that $f1$, $f2$, $f3$ and r are declared as double, what is the largest value of n that still gives a correct value of r ?

Solution:

With the same line of reasoning:

$f1 < 2^{5+52}$
$f1 < 2^{57}$
$10^n < 2^{57}$
$n < Log(2^{57})$
$n < 17.15$
$\boxed{n_{max} = 17}$