Assignment 1

Lokesh Mohanty (SR no: 21014)

September 2022

Question 1

(a) Define mapping $UtoST_w : \mathbb{Y} \to \mathbb{Z}$

Since we know that the function U_w is bijective, hence its inverse exists

$$\implies U_w^{-1} : \mathbb{Y} \to \mathbb{X}$$

Hence U_w^{-1} converts an unsigned integer to its bit representation and since ST_w converts a bit representation to its signed representation, $UtoST_w$ can be defined as

$$UtoST_w(y) = ST_w(U_w^{-1}(y))$$

We know that $U_w(x) = \sum_{i=0}^{w-1} x_i 2^i$ and $ST_w(x) = \sum_{i=0}^{w-2} x_i 2^i - x_{w-1}2^{w-1}$,

$$\implies U_w(x) - ST_w(x) = x_{w-1}2^w$$
$$\implies ST_w(x) = -x_{w-1}2^w + U_w(x)$$

Replacing x with $U_w^{-1}(y)$ we get $ST_w(U_w^{-1}(y)) = -x_{w-1}2^w + y$ where $U_w(x) = y$

We know that, x_{w-1} is 1 only when $y \geq 2^{w-1}$. Hence

$$\begin{aligned} UtoST_w(y) &= -2^w + y, && \text{when } y \geq 2^{w-1} \\ &= y, && \text{when } y < 2^{w-1} \end{aligned}$$

(b) Define mapping $ST_w toU : \mathbb{Z} \to \mathbb{Y}$

Since we know that the function ST_w is bijective, hence its inverse exists

$$\implies ST_w^{-1} : \mathbb{Z} \to \mathbb{X}$$

Hence ST_w^{-1} converts an signed integer to its bit representation and since U_w converts a bit representation to its unsigned representation, $ST_w toU$ can be defined as

$$ST_w toU(z) = U_w(ST_w^{-1}(z))$$

We know that $U_w(x) = \sum_{i=0}^{w-1} x_i 2^i$ and $ST_w(x) = \sum_{i=0}^{w-2} x_i 2^i - x_{w-1}2^{w-1}$,

$$\implies U_w(x) - ST_w(x) = x_{w-1}2^w$$
$$\implies U_w(x) = x_{w-1}2^w + ST_w(x)$$

Replacing x with $ST_w^{-1}(y)$ we get $U_w(ST_w^{-1}(y)) = x_{w-1}2^w + y$ where $ST_w(x) = y$.

We know that, x_{w-1} is 1 only when $y < 0$. Hence

$$\begin{aligned} STtoU_w(y) &= 2^w + y, && \text{when } y < 0 \\ &= y, && \text{when } y >= 0 \end{aligned}$$

Question 2

(a) Convert decimal numbers to IEEE float format and hexadecimal

- 86.125

IEEE single precision float format $((-1)^s \times 1.f \times 2^{-127+e})$:

$$\begin{aligned}86.125 &= (2^6 + 2^4 + 2^2 + 2^1) + (2^{-3}) \\ &= 1010110.001 \\ &= 1.010110001 \times 2^6 = 1.010110001 \times 2^{-127+133} \\ &= (-1)^0 \times 1.010110001 \times 2^{-127+133}\end{aligned}$$

Hence $s: 0,\ f: 010110001000...,\ e: 10000101$

Hexadecimal format(from IEEE format):

binary :	01000010	10101100	01000000	00000000
hexadecimal :	42	*AC*	40	00

Hence hexadecimal format(from IEEE format): 42AC4000

Hexadecimal from decimal: $86.2 = 5(16^1) + 5(16^0) + 5(16^{-1}) = 55.2$

- 0.523

IEEE single precision float format $((-1)^s \times 1.f \times 2^{-127+e})$:

$$\begin{aligned}0.523 &= (2^{-1} + 2^{-6} + 2^{-8} + 2^{-9} + 2^{-10} + 2^{-11} + 2^{-15} + 2^{-16} + 2^{-18} + 2^{-20} + 2^{-22}) \\ &= 0.10000101111000011010101 \times 2^{-1} \\ &= 0.10000101111000011010101 \times 2^{-127+126} \\ &= (-1)^0 \times 1.0000101111000011010101 \times 2^{-127+126}\end{aligned}$$

Hence $s: 0,\ f: 00001011110001101010100,\ e: 01111110$

Hexadecimal format(from IEEE format):

$binary:$	00111111	00000101	11100011	01010100
$hexadecimal:$	$3F$	05	$E3$	54

Hence hexadecimal format(from IEEE format): 3F05E354

Hexadecimal from decimal: $0.523 \approx 8(16^{-1}) + 5(16^{-2}) + 14(16^{-3}) = 0.85E$

- -0

  0 is stored in denormalized as it cannot be stored in normalized form of IEEE single precision float format

  IEEE single precision float denormalized form $((-1)^s \times 0.f \times 2^{-127})$: Hence for -0, $s : 1$, $f : 000...$, $e : 00000000$

$$(-1)^0 \times 0.00... \times 2^{-127}$$

  Hexadecimal format(from IEEE format):

$binary:$	10000000	00000000	00000000	00000000
$hexadecimal:$	80	00	00	00

  Hence hexadecimal format(from IEEE format): 80000000

  Hexadecimal from decimal: $0 = 0(16^0) = 0$

(b) Number of numbers between

- -2^{-12} and -2^{-11}

  -2^{-12} is represented as $(-1)^1 \times 1.0 \times 2^{115-127}$ -2^{-11} is represented as $(-1)^1 \times 1.0 \times 2^{116-127}$

  All the numbers between -2^{-12} and -2^{-11} will have fixed $s : 1$ and $e : 115$, f can be anything other than all zeros i.e., each bit in f has 2 options (1 or 0), and since there are 23 bits in f, there are $2^{23} - 1$ possibilities other than all zeros.
  Hence number of numbers between -2^{-12} and -2^{-11} are $2^{23} - 1$

- -2^{-13} and -2^{-12}

  -2^{-13} is represented as $(-1)^1 \times 1.0 \times 2^{114-127}$ -2^{-12} is represented as $(-1)^1 \times 1.0 \times 2^{115-127}$

  All the numbers between -2^{-13} and -2^{-12} will have fixed $s : 1$ and $e : 114$, f can be anything other than all zeros i.e., each bit in f has 2 options (1 or 0), and since there are 23 bits in f, there are $2^{23} - 1$ possibilities other than all zeros.
  Hence number of numbers between -2^{-13} and -2^{-12} are $2^{23} - 1$

Hence in both cases the number of numbers between consecutive powers of 2 are same.

(c) Smallest value of n(natural number) that cannot be represented using

- 32-bit IEEE single-precision float representation

  We know that the gap between any 2 consecutive numbers between 2^m and 2^{m+1} is 2^{m-23} as

$$\begin{array}{lr} 1.00000000000000000000000 \times 2^m : & 2^m \\ 1.00000000000000000000001 \times 2^m : & 2^m + 2^{m-23} \end{array}$$

  Hence for all $m > 23$, the gap is greater than 1 (i.e., the natural numbers in between cannot be precisely represented)

  The smallest value of such m is 24 i.e., 2^{24} and the next number that can be stored is $2^{24} + 2^1$ which implies that $2^{24} + 1$ cannot be stored. Hence $n = 2^{24} + 1$

- 32-bit signed integer representation

  We know that in integer representation the gap always remains constant$(= 1)$.
  Hence the smallest n that cannot be represented should be out of range.

  Range: -2^{31} to $2^{31} - 1$
  Hence $n = 2^{31}$

- 32-bit unsigned integer representation

We know that in integer representation the gap always remains constant($= 1$).
Hence the smallest n that cannot be represented should be out of range.

Range: 0 to $2^{32} - 1$
Hence $n = 2^{32}$

Question 3

(a)

(i) minimum number of bits required to store the score(w):

Range of the scores is from 1 to 10. Since the range has 10 distinct values and since $2^3 \leq 10 \leq 2^4$, a minimum of 4 bits will be required. Where 1 can be stored as `0001` and 10 can be stored as `1010`.Hence $w = 4$.

(ii) Explaination of the counter-intuitive behaviour

Using the above 4-bit unsigned representation (assume it to be $b_4b_3b_2b_1$), 2 points would be deducted from their score when they achieve democracy, which is $b_4b_3b_2b_1-$ `0010`. Hence when the country X achieves democracy and since it started with score of 1, its final score will be `0001 - 0010` which is equal to `1111`. And since `1111` is greater than 10, the coutry X becomes overly aggressive and starts bombing other countries.

(iii) Fix for the above counter-intuitive behaviour

The techincal term for the above behaviour is called "negative overflow". This can be fixed by setting a minimum value which the score can take(=1) and adding an if condition which doesn't subtract 2 if score is 1 and subtracts 1 if score is 2.

(b)

(i) Number of bits(w) to store the timing counter in w-bit signed integer

The duration is stored in centiseconds

$$\begin{aligned} 249 \text{ days} &= 249 * 24 * 60 * 60 * 100 \text{ centiseconds} \\ &= 2151360000 \\ 2^{31} &\leq 2151360000 \leq 2^{32} \end{aligned}$$

We know that range of a w-bit signed integer is -2^{w-1} to $2^{w-1} - 1$

Since the overflow happens at around 249 days, it should not able to be stored in the counter. Hence, if w = 32, then the max value that can be stored will be $2^{31} - 1$ which is less than 249 days. For any value of w ¡ 32, the overflow will happen much earlier and for any value of w ¿ 32, the overflow will happen much later.

$\therefore w = 32$

(ii) Period after which the timing counter will overflow in a w-bit unsigned integer

Range for an unsigned w-bit integer is 0 to $2^w - 1$. Since $w = 32$, The error will occur just after the timing counter reaches $2^{32} - 1$ (i.e., the error will occur at 2^{32} centiseconds)

(iii) Period after which the timing counter will overflow in a 2w-bit signed integer

Range for an signed 2w-bit integer is 0 to $2^{2w-1} - 1$. Since $w = 32$, The error will occur just after the timing counter reaches $2^{63} - 1$ (i.e., the error will occur at 2^{63} centiseconds)

(c) Convert 16-bit float to decimal and 16-bit signed integer

We know that a 16-bit floating point is represented as: $(-1)^s \times 1.f \times 2^{-15+e}$, with 1 bit in s, 5 bits in e and the remaining 10 bits in f.

(i) **Stage I**: 0110001111111011

Representation $\rightarrow$ s: 0, e: 11000, f: 1111111011

$$\begin{aligned} 1.1111111011 \times 2^{24-15} &= 1.1111111011 \times 2^9 \\ &= 1111111101.1 = 1021 + 0.5 \\ &= 1021.5 \end{aligned}$$

Hence its decimal value is $\boxed{1021.5}$

When converting to 16 bit integer, the value after the decimal point cannot be stored hence its binary representation is $\boxed{0000001111111101}$

(ii) **Stage II**: 0110011111101100

Representation -¿ s: 0, e: 11001, f: 1111101100

$$\begin{aligned} 1.1111101100 \times 2^{25-15} &= 1.1111101100 \times 2^{10} \\ &= 11111101100 = 2028 \end{aligned}$$

Hence its decimal value is $\boxed{2028}$ and
16 bit integer is $\boxed{0000011111101100}$

(iii) **Stage III**: 0111001101101101

Representation -¿ s: 0, e: 11100, f: 1101101101

$$\begin{aligned} 1.1101101101 \times 2^{28-15} &= 1.1101101101 \times 2^{13} \\ &= 11101101101000 = 15208 \end{aligned}$$

Hence its decimal value is $\boxed{15208}$
16 bit integer is $\boxed{0011101101101000}$

(iv) **Stage IV**: 0111100000011111

Representation -¿ s: 0, e: 11110, f: 0000011111

$$\begin{aligned} 1.0000011111 \times 2^{30-15} &= 1.0000011111 \times 2^{15} \\ &= 1000001111100000 = 33760 \end{aligned}$$

Hence its decimal value is $\boxed{33760}$

When converting to 16 bit integer, the value is greater than the range and hence causes error. This error is denoted by $\boxed{1000000000000000}$

$\therefore$ At **Stage IV**, the value of V_H becomes large enough to throw an error.

(v) **Stage V**: 0111101000111111

Representation → s: 0, e: 11110, f: 1000111111

$$\begin{aligned} 1.1000111111 \times 2^{30-15} &= 1.1000111111 \times 2^{15} \\ &= 1100011111100000 = 51168 \end{aligned}$$

Hence its decimal value is $\boxed{51168}$

When converting to 16 bit integer, the value is greater than the range and hence causes error. This error is denoted by $\boxed{1000000000000000}$

(d)

(i) Binary representation of $0.1 - x$

$$\begin{aligned} &0.1 && \rightarrow 0.00011001100110011001100[0011]..._2 \\ &x && \rightarrow 0.00011001100110011001100_2 \\ \implies &0.1 - x && = 0.00000000000000000000000[0011]_2 \end{aligned}$$

(ii) Approximate decimal value of $0.1 - x = 9.54 \times 10^{-8}$

(iii) $\Delta t =$ (time in centiseconds)$\times(0.1 - x)$

$$\begin{aligned} &\Delta t = 60 \times 60 \times 10 \times t \times (0.1 - x) \\ \implies &\Delta t = 0.0034344 \times t \end{aligned}$$

For $t = 50$

$$\begin{aligned} &\Delta t = 0.0034344 \times 50 \\ \implies &\Delta t = 0.17172 \end{aligned}$$

For $t = 100$

$$\begin{aligned} &\Delta t = 0.0034344 \times 50 \\ \implies &\Delta t = 0.34344 \end{aligned}$$

(iv) $x_{err} = v \times \Delta t = 3000 \times \Delta t$

$$\begin{aligned} x_{err} &= 3000 \times \Delta t \\ &= 3000 \times 0.0034344 \times t = 28.62 \times t \end{aligned}$$

if $x_{err} = 500 \implies 500 = 28.62 \times t \implies t = 17.470$ hours

Question 4

(a) $y = \sum_{i=1}^{\infty} d_i \times 10^{n-i}, \psi_k(y) = \sum_{i=1}^{k-1} +d'_k \times 10^{n-k}$

$$\frac{|y - \psi_k(y)|}{|y|} = \frac{\sum_{i=k}^{\infty} -d'_k \times 10^{n-k}}{\sum_{i=1}^{\infty} \times 10^{n-i}}$$

(b)

$$\begin{aligned}
\frac{p!}{3!(p-3)!} &\leq 0.9999 \times 10^{15} \\
\implies \frac{p(p-1)p-2)}{6} &\leq 9999 \times 10^{11} \\
\implies p(p-1)p-2) &\leq 59994 \times 10^{11} \\
\implies p^3 &\leq 59994 \times 10^{11} \\
\implies p &\leq 181706.0020
\end{aligned}$$

(c)

Question 5

(a) Number of FLOPS in the pseudo-code:
In every iteration of the loop, 1 multiplication and 1 addition takes place. And since the iteration happens n times

$\boxed{\text{Number of FLOPS: } 2n}$

(b) Number of FLOPS in the pseudo-code:
In every iteration of the inner loop, 1 multiplication and 1 addition takes place. And since the inner loop iteration happens $m \times n$ times

$\boxed{\text{Number of FLOPS: } 2mn}$

(c) Number of FLOPS in the pseudo-code:
In every iteration of the inner loop, 1 multiplication and 1 addition takes place. And since the inner loop iteration happens $m \times r \times n$ times

$\boxed{\text{Number of FLOPS: } 2mrn}$

(d) Number of FLOPS in the pseudo-code:

(i) $\mathbf{ABx}$:

$$\begin{aligned} \mathbf{AB} &\to 2mnr, & (\mathbf{AB})\mathbf{x} &\to 2mr, Total : 2mr(n+1) \\ \mathbf{Bx} &\to 2nr, & \mathbf{A}(\mathbf{Bx}) &\to 2mn, Total : 2n(m+r) \end{aligned}$$

If m = 10000, n = 5000, r = 500, p = 150,

$$\begin{aligned} 2mr(n+1) &= 2(10000)(500)(5001) = 5.001 \times 10^{10} \\ 2n(m+r) &= 2(5000)(10500) = 1.05 \times 10^{8} \end{aligned}$$

Hence minimum number of operations are 1.05×10^8

(ii) $\mathbf{ABC}$:

$$\begin{aligned} \mathbf{AB} &\to 2mnr, & (\mathbf{AB})\mathbf{C} &\to 2mrp, Total : 2mr(n+p) \\ \mathbf{BC} &\to 2nrp, & (\mathbf{A})(\mathbf{BC}) &\to 2mnp, Total : 2np(r+m) \end{aligned}$$

If m = 10000, n = 5000, r = 500, p = 150,

$$
\begin{aligned}
2mr(n+p) &= 2(10000)(500)(5150) = 5.15 \times 10^{10} \\
2np(r+m) &= 2(5000)(150)(10500) = 1.575 \times 10^{10}
\end{aligned}
$$

Hence minimum number of operations are 1.575×10^{10}