Assignment 3-7 Check out the value of Nan

1. **Make an Assembly program that shows the NaN. Divide the 0 by 0.**

```
## Program: 0 / 0
        .text
      .globl  main
main:
   li $v0, 2       # print floating service code

   li              $t0, 0
   mtc1            $t0, $f2        # $t0 = 0;
   cvt.s.w         $f4, $f2    # Convert the double precision number to single precision
   div.s           $f9, $f4, $f4   # Largest Number / 0
   li              $v0, 2
   mfc1            $t0, $f9
   mov.s           $f12, $f9
   syscall
## End of file
```

2. **Print the result of division**

```
FG8     = 0.00000
FG9     = NaN
FG10    = 0.00000
FG11    = 0.00000
FG12    = NaN
FG13    = 0.00000
```

```
Output

-nan
```

Negative Not a Number

a. **Elaborate on the numbers on the register $f9**
   $f9 contains NaN, because $f9 is the dividend of $f4 divided by $f4. Contained in $f4 is 0, and 0/0 is undefined not only in mathematics but for IEEE 754 floating point arithmetic due to it's undefined ambiguous nature of being.
   Also noticed this pathing,
   $0 (built in register that holds 0) -> $8 -> $f2 -> $f4 -> $f4/$f4 = Nan -> $f9 -> $f12 to output.

b. **And $t0 based on the IEEE 754 format, Is this NaN? Why?**

0xFFC00000 = 1111 1111 1100 0000 0000 0000 0000 0000
1 = sign bit = **Negative**
1111 1111 = Exponent = When the exponent bits are all 1's, it indicates that the value is **either Infinity or NaN.**
100 0000 0000 0000 0000 0000 = Mantissa = When the mantissa bits are non-zero, in this case the leading 1 is making that condition true, when the exponent bits are all 1's, this tells us that the value is **NaN.**
Knowing that the sign bit is negative, we can conclude that this is a **Negative NaN**
This is confirmed by the output we got as seen above.
It's NaN because the value being divided is zero to itself, which is undefined/NaN.

*Notes: Division by zero is **undefined** in classical arithmetic because there is no number that can satisfy the operation consistently across different contexts.*
*When both the numerator and denominator are zero, the problem becomes even more ambiguous, as there are **infinitely many potential answers**.*
*To avoid giving an incorrect or misleading result, the IEEE 754 standard defines such cases as **NaN**, signaling to the system and the user that the result of the computation is not a valid real number.*