حسین تاتار – 40133014

سید مهران رسولی – 40131015

هدف از این ازمایش طراحی و پیاده سازی یک بخش محاسبه گر تفریق اعداد اعشاری ممیز شناور بر اساس استاندارد IEEE 754 است.

در استاندار IEEE 754 دو نوع فرمت برای نشان دادن اعداد اعشاری بیان شده که اولی 32 بیتی و دومی 64 بیتی است. نمایش هر عدد اعشاری شامل سه بخش (Sign, Exponent, Fraction) است که مقدار پهنای هر یک از این بخش ها در فرمت استاندارد به صورت زیر است:

فرمت 32 بیتی : S = 1bit, Exponent = 8bit, Fraction = 23bit

فرمت 64 بیتی : S = 1bit, Exponent = 11bit, Fraction = 52bit

ما در این ازمایش از فرمت 32 بیتی استفاده نمودیم و اعداد اعشاری ورودی و خروجی ما 32 بیت هستند.

پیاده سازی الگو برای انجام این عملیات تفریق ما شامل 8 مرحله است که عبارتند از:

IDLE, CHECK_ZERO, UNPACK, ALIGN, SUBTRACT, NORMALIZE, PACK, DONE

ما ابتدا علامت شروع را دریافت کرده و سپس چک میکنیم که ایا یکی از اعداد یا هردو عدد صفر هستند یا خیر، اعداد صفر در نمایش استاندارد شامل بیت های تمام صفر هستند. بعد از ان نوبت به این میرسد که بخش های مختلف را از هم جدا کنیم و بعد می اییم و Exponent جواب را در حالت بعدی مقدار دهی کرده و نوبت به انجام تفریق میرسد.

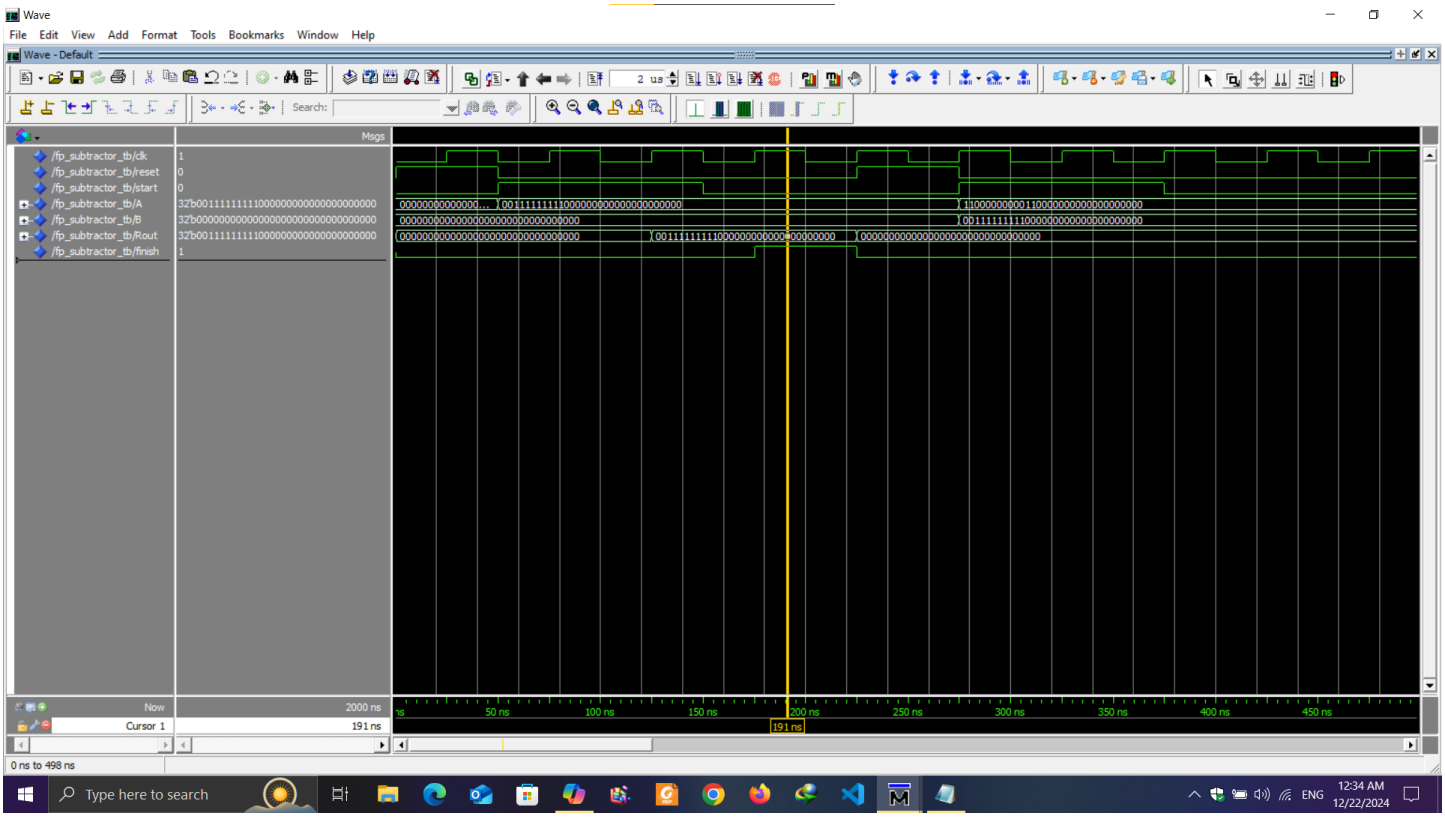
در مرحله انجام تفریق ما تفریق را بر اساس اینکه ایا هم علامتند یا مختلف علامت (A-B, B-A, A+B) انجام داده و بیت علامت خروجی نهایی را هم مشخص میکنیم. در مرحله بعدی می اییم و جواب تفریق را نرمال میکینم چون ممکن است از فرمت استاندارد خارج شده باشد.

در نهایت نیز ما خروجی سه قسمت جواب را به هم چسبانده و فرمت جواب براساس استاندارد IEEE 754 بدست می اید و کار به اتمام میرسد.

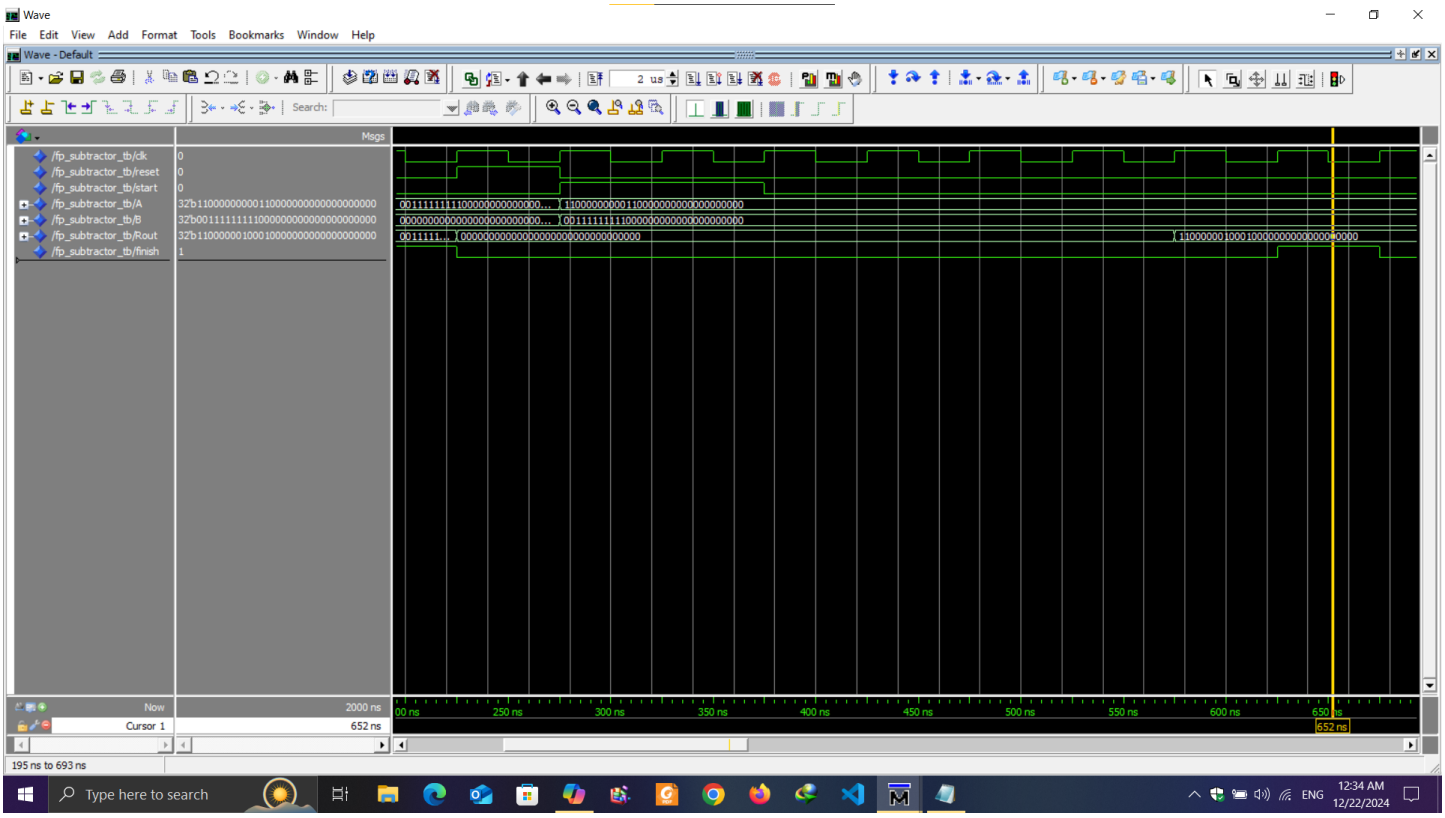
در نهایت ما خروجی این کد را برای دو حالت که در تست بنچ ما نوشته شده است ازمایش کرده و جواب را میبینیم.....

```
    -- Test process
    stim_proc: process
    begin
        -- Reset the system
        reset <= '1';
        wait for 50 ns;
        reset <= '0';

        -- Test case: Subtract 1.5 by 0 (expected result: 1.5)
        -- A = 1.5 => 0 01111111 10000000000000000000000
        -- B = 0   => 0 00000000 00000000000000000000000
        start <= '1';
        A <= "00111111111000000000000000000000"; -- 1.5
        B <= "00000000000000000000000000000000"; -- 0
        wait for 100 ns;
        start <= '0';
        wait until finish = '1';
        wait for 50 ns;

        -- Test case 2: Subtract -2.75 + 1.5 (expected result: -4.25)
        -- A = -2.75 => 1 10000000 01100000000000000000000
        -- B = 1.5   => 0 01111111 10000000000000000000000
        reset <= '1';
        wait for 50 ns;
        reset <= '0';
        start <= '1';
        A <= "11000000000110000000000000000000";
        B <= "00111111111000000000000000000000";
        wait for 100 ns;
        start <= '0';
        wait until finish = '1';
        wait for 50 ns;

        wait;
    end process;
```

کد تست بنچ و اعداد ورودی برای تست کد



خروجی کد برای تست اول (1.5 = 0 – 1.5)



خروجی کد برای تست دوم (-4.25 = (+1.5)– (-2.75