1 Floating point numbers

Consider the following program:

```
section .data
        x DD 3.14
        y DD 2.1

section .text
        MOVSS xmm0, [x]
        MOVSS xmm1, [y]

        ADDSS xmm0, xmm1

        MOV eax, 1
        INT 80h
```

See about **xmm** registers here.

How many **xmm** registers?.

MOVSS instruction is used to move the floating point values to the **xmm** registers. **SS** means **S**caler **S**ingle precision.

We will execute this program with GDB.

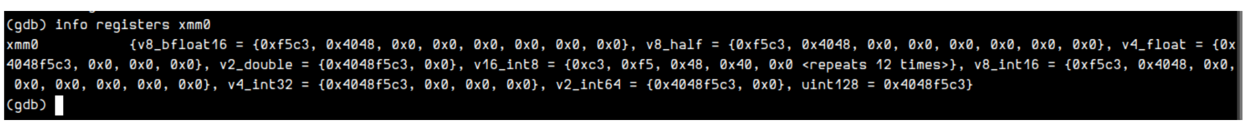


Figure 1: Values in **xmm**

We can see a bunch of arrays(I guess) which have a bunch of values in them. To get the value stored in **xmm0** we particularly use the **v4_float** array.

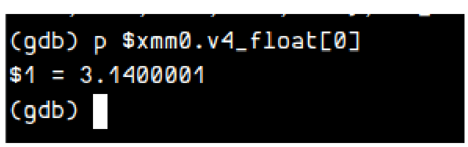
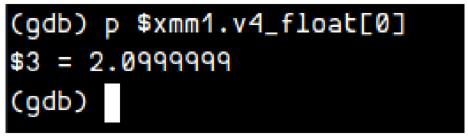


Figure 2: **v4_float** array's 0^{th} element(**xmm0**)

Notice that we assigned the value 3.14 to **xmm0** but in this image it's showing 3.1400001.

Similar behaviour can be seen in **xmm1** as well.

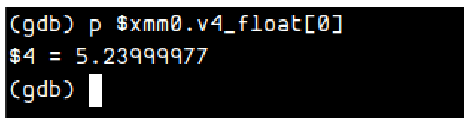


```
(gdb) p $xmm1.v4_float[0]
$3 = 2.0999999
(gdb)
```

Figure 3: **v4_float** array's 0^{th} element(**xmm1**)

The reason why these values are not stored exactly as they were defined is that we use a special notation called IEEE floating-point notations.

From the image below:



```
(gdb) p $xmm0.v4_float[0]
$4 = 5.23999977
(gdb)
```

Figure 4: Sum of **xmm0** and **xmm1**

We can see that the sum is also not exact.