به نام خدا

گزارش کار آزمایشگاه ریزپردازنده

آزمایش ۱۱

مدرس : مهندس بی طالبی

تارا برقیان

مهرشاد سعادتی نیا

نیم سال اول ۰۱-۱۴۰۰

فهرست

مقدمه :

در این تمرین با ترکیب برنامه نویسی زبان اسمبلی ۸۰۸۶ و C++ برای اجرای دستورات SIMD آشنا شدیم.

توضیح کد:

ابتدا یک تابع برای تشخی علامت داریم detectSign()

سپس برای ورودی گرفتن از کاربر از کتابخانه conio استفاده کردیم. برای کامل خواندن اعداد و تشخیص علامت و اعشار ان را کاراکتر به کاراکتر خواندیم.

```
6
7   int detSign(int ch) {
8       if (ch == '-')
9       {
10          return -1;
11      }
12      else if (ch == '+')
13      {
14          return 1;
15      }
16      return 0;
17  }
```

```
float getInput()
{
    // sign = 1 [number is positive]
    // sign = -1 [number is negative]
    int sign = 1;
    int ch = '0';
    int zero = '0';
    float inputNumber = 0;
    int place = 1;

    ch = _getche();

    // determine sign of input number
    sign = detSign(ch);
    if (sign != 0) {
        ch = zero;
    }
    else {
        sign = 1;
    }

    // get integer part of input number
    while (ch >= '0' && ch <= '9')
    {
        inputNumber = (inputNumber * 10) + (ch - zero);
        ch = _getche();
    }

    if (ch == '.') {
        // get fraction
        ch = zero;
        while (ch >= '0' && ch <= '9')
        {
            inputNumber += ((ch - zero) / (float)place);
            place *= 10;
            ch = _getche();
        }
        return (float)sign * inputNumber;
    }
    else {
        return (float)sign * inputNumber;
    }
}
```

سپس با کمک زبان cpp مقدار های x+a و x+b را حساب کرده و حاصل ضربشان را هم محاسبه میکنیم.

و سپس به زبان اسمبلی بر میگردیم و مجدد ضرب دو وکتور را با اسمبلی حاسبه میکنیم. (با کمک دستور mulps)

چون طبق توضیحات منابع در پیوست اینگونه عملیات مورد نظر packed خواهد بود.

توضیحات دستورات SSE بطور مفصل در داکیومنت 0x86 در سایت oracle موجودست.

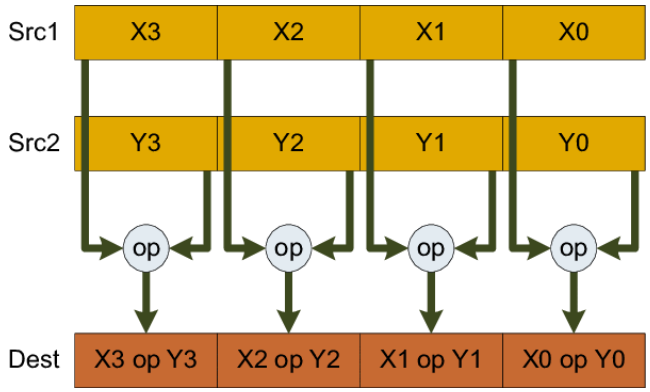
کد زیر قسمت محاسبه ی حاصل ضرب به کمک اسمبلی است که با استفاده از asm block_ می توانیم آنرا داخل کد c++ بنویسیم.

```
_asm
{
    mov ecx, 0
    MUL_LOOP:
    movaps xmm2, oword ptr vi1Copy[ecx]
        movaps xmm0, oword ptr vi2Copy[ecx]
        mulps xmm2, xmm0
        movaps oword ptr vo2Copy[ecx], xmm2
        add ecx, 16
        cmp ecx, 400
        jnz MUL_LOOP
}
```

بعنوان مثال به کمک دستورات movaps می توانیم ۴ مقدار ممیز شناور را از ممووری در رجیستر های xmm قرار دهیم و به کمک آن عملوند های خود را آماده کنیم سپس به کمک دستور mulps می توانیم این دو رجیستر را به صورت packed در هم ضرب کنیم.

سپس مقدار اندیس ها را افزایش می دهیم تا برای چهار مقدار بعدی محاسبه صورت گیرد و به همین ترتیب در یک حلقه مقادیر را محاسبه می کنیم.

روند منطقی این عملیات به کمک شکل زیر بیان میشود.



نتایج بدست آمده (**۲۰ سطر اول**) از وارد کردن دو عدد ۱۰ و 5 را در زیر مشاهده می کنید.