一、 设计思路

基本要求：完成多项式 Z=（X+Y）*8/2 的计算，无论是 X86 还是鲲鹏处理器 ARM V8，实现以上计算的大体思路都是先将 x 和 y 两数相加，再进行乘除的运算，在汇编语言中乘除的实现是转化为移位操作进行的，乘 8 除 2 就是将操作数向左移 3 个单位，再右移一个单位，最后再将所得的结果保存。

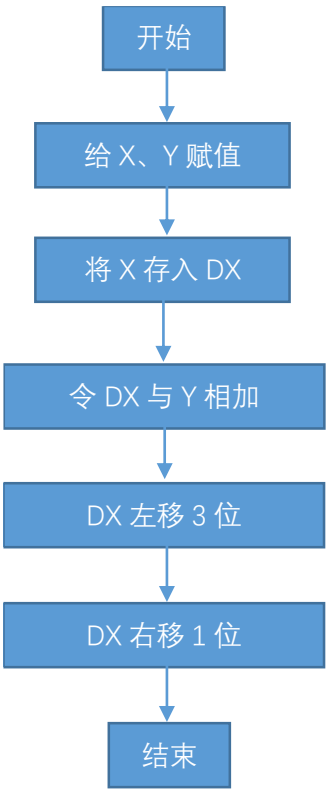
提高要求：在完成了基本要求的同时，我们根据提高要求，基于鲲鹏处理器 arm v8 的 C 语言调用汇编混合编程实现了任意两数的四则运算，我们采用子程序调用的方法实现四则运算，将加，减，乘写成了子程序，在主程序运行时，通过对操作符的判断，来决定是否调用子程序，调用哪个子程序，这样就自然而然的形成了一种条件分支结构。此外我们在 C 语言主程序中实现了简单的人机交互界面。

不足：我们仅完成了两个数的加减乘除计算，并没有再进行拓展，由于任意表达式的计算要考虑到操作符优先级的问题，而我们的时间和学习深度有限，目前还不能实现任意表达式计算的功能。

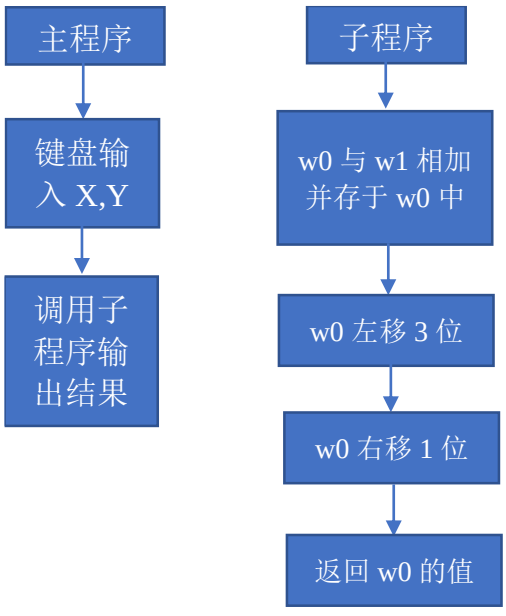
二、 程序框架

基础要求（完成多项式运算，如：Z=(X+Y)×8÷2）：

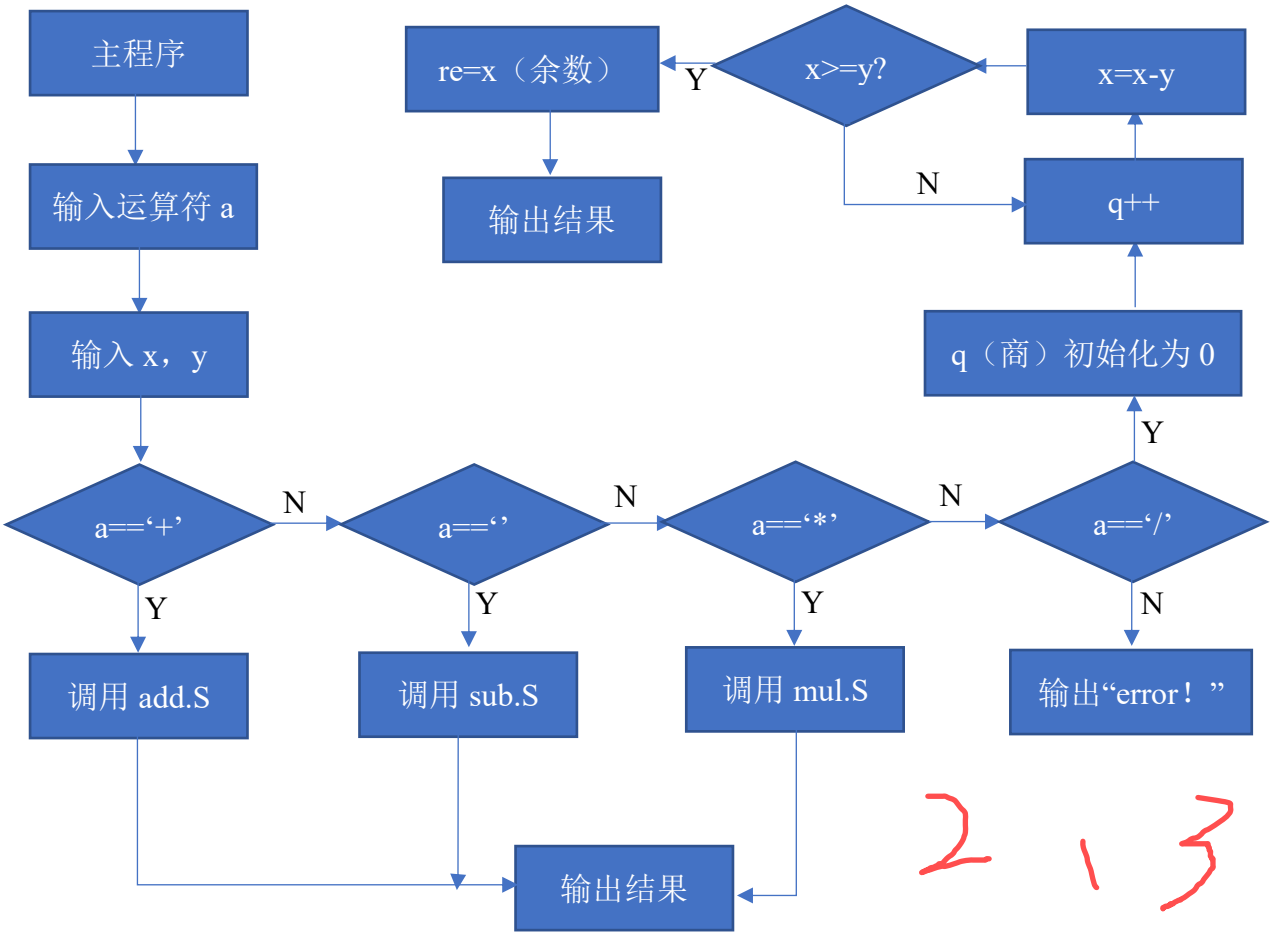
基于 x86 的程序框架



基于 ARM V8 的程序框架



提高要求（基于 ARM V8 完成任意两数的四则运算）：



其中三个子程序均仅有相应的运算指令，故不再展示其框架。

三、 程序源代码

基础要求（x86）：

```
STACK SEGMENT STACK
        DB 10 DUP(?)
STACK ENDS
DATA SEGMENT
VARX DW ?
VARY DW ?
RESULT DW ?
DATA ENDS
CODE SEGMENT
      ASSUME CS:CODE,DS:DATA,SS:STACK
START:
MOV AX,DATA
        MOV DS,AX
MOV DX,VARX
```

```
        ADD DX,VARY
        MOV CL,3
        SAL DX,CL
        SAR DX,1
        MOV RESULT,DX
        MOV AH,4CH
        INT 21H
        CODE ENDS
    END START
```

基础要求（ARM V8）：

```
        ----------------------------calcu.c------------
        #include <stdio.h>
        #include <stdlib.h>
        typedef unsigned int u32;
        extern int calcu(u32, u32);
        int main()
        {
            u32 x;
            u32 y;

            scanf("%d%d",&x,&y);

            printf("z=%d\n", calcu(x,y));

            return 0;
        }

        --------------------------calcu.S-----------------------------
        #include "calcu.h"
        ENTRY(calcu)
        add w0,w1,w0
        lsl w0,w0,3
        lsr w0,w0,1
        ret
        ENDPROC(calcu)
```

提高要求：

```
--------------------------- jisuan.c-----------------------------------
#include <stdio.h>
#include <stdlib.h>
typedef unsigned int u32;
extern int add(u32, u32);
extern int sub(u32, u32);
```

```
extern int mul(u32, u32);
//extern int divd(u32, u32);
int main()
{
	u32 x;
	u32 y;
	char a;
	u32 q=0;
	u32 re=0;
	printf("请输入运算符  (+, -, *, /): ");
	scanf("%c",&a);
	printf("请输入两个数字: ");
	scanf("%d%d",&x,&y);

	switch(a)
	{
		case '+':
		printf("x+y=%d",add(x,y));
		break;

		case '-':
		printf("x-y=%d",sub(x,y));
		break;

		case '*':
		printf("x*y=%d",mul(x,y));
		break;

		case '/':
		{
			while(x>=y)
			{
				q++;
				x=x-y;
			}
			re=x;
			printf("x/y=%d……%d",q,re);
		}
		break;

		default:
		printf("error!");
	}
```

----------add.S----------

```
#include "calcu.h"
ENTRY(add)
add w0,w1,w0
ret
ENDPROC(add)
```

----------sub.S----------

```
#include "calcu.h"
ENTRY(sub)
sub w0,w0,w1
ret
ENDPROC(sub)
```

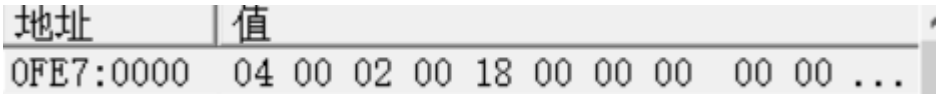
----------mul.S----------

```
#include "calcu.h"
ENTRY(mul)
mul w0,w1,w0
ret
ENDPROC(mul)
```

四、 运行结果

基础要求（x86）：

为 X 赋值 4，为 Y 赋值 2，计算结果为 18H（24）。



基础要求（ARM V8）：

```
[root@ecs-cbj ~]# gcc -g calcu.c calcu.S -o calcu
[root@ecs-cbj ~]# gdb calcu
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-120.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "aarch64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /root/calcu...done.
(gdb) run
Starting program: /root/calcu
4 2
z=24
[Inferior 1 (process 11978) exited normally]
Missing separate debuginfos, use: debuginfo-install glibc-2.17-292.el7.aarch64
(gdb)
```

提高要求：

```
(gdb) run
Starting program: /root/jisuan
输入操作符 (+, -, *, /): +
输入两个数字: 5 8
x+y=13[Inferior 1 (process 11997) exited normally]
```

```
Starting program: /root/jisuan
输入操作符 (+, -, *, /): -
输入两个数字: 8 5
x-y=3[Inferior 1 (process 12000) exited normally]
```

```
Starting program: /root/jisuan
输入操作符 (+, -, *, /): -
输入两个数字: 5 8
x-y=-3[Inferior 1 (process 12001) exited normally]
```

```
Starting program: /root/jisuan
输入操作符 (+, -, *, /): *
输入两个数字: 5 8
x*y=40[Inferior 1 (process 12002) exited normally]
```

```
Starting program: /root/jisuan
输入操作符 (+, -, *, /): /
输入两个数字: 16 8
x/y=2……0[Inferior 1 (process 12003) exited normally]
```

```
输入操作符 (+, -, *, /): /
输入两个数字: 5 8
x/y=0……5[Inferior 1 (process 12004) exited normally]
```