我知道你们不会看官方手册的【Doge

我就结合我查的资料简单总结了一下，你们把这个过一遍，再看接下来的作业答案就体会更深刻了

汇编嘛，简单入门就先把握寄存器和指令就行了，再深入才是架构层面。

作业原题目：

**Use TI CC implement the following algorithm in C55X and C64X assembly language:**

**for (i=0，f=0; i<N; i++)**

**f = f + c[i] \* x[i];**

**Compare the assembly program features between ARM,C55X and C64X.**

这里解释一下最后一句话：

C55X和C64X汇编属于DSP汇编，和平时接触的汇编有所不同，原因是处理器特性和应用场景不同，不同只要体现在指令集、数据格式、中断处理、处理器架构和数据通路等方面。

C55X、C64X和ARM汇编平台不同，前两者是DSP（数字信号处理器），后者是ARM处理器。两种硬件分别应用于数字信号处理和通用应用程序的开发。

那数字信号处理和通用应用程序有什么区别呢？

数字信号处理（DSP）是一种特殊的数据处理方式，主要应用于**信号处理领域**，例如音频处理、图像处理、通信等。数字信号处理通常涉及**采样、滤波、变换、编码、解码**等一系列复杂的算法和处理方法。DSP应用需要高性能的处理器和**专门的算法库**来实现。

通用应用程序是指通用计算机应用程序，主要应用于计算机领域的各种应用，例如**文本处理、图像处理、数据库管理、网络通信等**。通用应用程序涉及的数据类型和处理方法较为通用，可以用各种高级语言（如C、Java等）和编程工具来实现，同时也可以使用各种处理器和计算机硬件来实现。

C55X汇编的操作数一般为寄存器，采用A1、A2等符号表示，例如：ADD A2, A3, A4。

ARM汇编的操作数一般为寄存器、立即数或内存地址，采用#n、Rn、[Rn]等符号表示，例如：ADD R0, R1, #4。

C55X汇编和ARM汇编的指令集也有很大的区别。C55X汇编的指令集主要包括定点数运算指令、移位指令、跳转指令等，用于实现数字信号处理算法。

ARM汇编的指令集则更加通用，包括数据处理指令、逻辑指令、分支指令、存储器指令等，用于实现各种应用程序。

C55X和ARM处理器的寄存器也有很大的区别。C55X有16个通用寄存器，每个寄存器都是32位的，可以存储整数或者定点数。

ARM处理器的寄存器数量更多，包括13个通用寄存器、一些特殊寄存器和浮点寄存器等。

**注意**

C55x，C64x汇编里，目的寄存器位于最后一个，而ARM相反，位于第一个。例如：

MOV r2, #0 ; 意味着r2=0

MVK 0, A2 ; 意味着A2=0

下面简单介绍一下C55x和C64x汇编指令。

## C55X和C64X常用的汇编指令

加法指令：ADD、ADD2、ADDC、ADDK、ADDP、ADDSP

减法指令：SUB、SUB2、SUBC、SUBK、SUBSP

乘法指令：MPY、MPY2、MPYU、MPYU2、MPYSP

除法指令：DIVS、DIVU、DIVS2、DIVU2、MPYSU

逻辑指令：AND、OR、XOR、NOT

移位指令：SHL、SHR、SRA、SRL、ROTL、ROTR

条件转移指令：B、BL、BC、BLC、BGT、BGE、BLT、BLE、BEQ、BNE

循环指令：LOOP、LOOPE、LOOPNE、NEXT、REPEAT、REPEATGT、REPEATLT

加载/存储指令：LDW、LDH、LDB、STW、STH、STB

程序控制指令：CALL、JUMP、RTS、NOP

**简单解析**

K代表立即数，所以MVK代表立即数的移动，ADDK代表与一个立即数相加，SUBK代表相减。

P代表protect

C55x和C64x是两种不同的DSP架构，虽然都是由德州仪器公司（TI）设计，但是它们的指令集是不同的。C55x相对于C64x来说，指令集要简单一些。

C55x**没有**C64x中的保护模式，也没有保护指令（如ADDP）和保护标志（如P位）。

C55x的指令集中**没有**乘法累加（MAC）指令和累加运算（ACC）指令，但是提供了其他的一些指令来执行类似的功能。

C55x**没有**C64x中的并行操作指令（如PDP和PDI），也没有C64x中的浮点数指令集（FPU）。

需要注意的是，虽然C55x的指令集相对简单，但是它的执行速度却比C64x要快，因此在某些特定的应用场景下，C55x比C64x更适合用来实现特定的算法。

## C55X和C64X的寄存器

1. 通用寄存器

C55X和C64X都有16个通用寄存器，用于存储算法中的数据和中间结果。这些寄存器按编号从A0到A15命名，每个寄存器的大小为32位，可以存储整数或者定点数。

1. 状态寄存器

状态寄存器包括PSR（程序状态寄存器）和CSR（控制状态寄存器）。它们用于记录程序运行时的状态和标志位，例如条件码、中断状态、进位状态等。在C55X和C64X中，状态寄存器是一个32位的寄存器，其中的各个位表示不同的状态或标志。

1. 控制寄存器

控制寄存器包括ICR（中断控制寄存器）、CSR（控制状态寄存器）、TSCL（时间戳计数器低位）和TSCH（时间戳计数器高位）等。它们用于控制程序的运行和处理器的行为。

1. 浮点寄存器

**C64X还包括浮点寄存器**，用于存储浮点数和双精度浮点数。这些寄存器按编号从D0到D31命名，每个寄存器的大小为64位。**C55X不支持浮点运算，因此没有浮点寄存器**，这是64和55的区别

这些寄存器可以通过汇编指令进行读写和操作，用于实现数字信号处理算法中的数据存储和运算。在编写代码时，需要根据具体应用和处理器型号选择合适的寄存器，并合理利用寄存器，以实现高效的数字信号处理。

**A开头的寄存器和XAR开头的寄存器的区别**

在C55X和C64X DSP中，XAR开头的寄存器是一组16位的通用寄存器，用于存储数据、指针和地址等信息。这些寄存器的名称以XAR开头，后面跟着一个数字，表示寄存器的编号。具体来说，C55X和C64X DSP中的XAR寄存器有XAR0、XAR1、XAR2、XAR3、XAR4、XAR5、XAR6、XAR7共8个，其中XAR0是特殊寄存器，用于存储数据存取指令的地址。

这些XAR寄存器都是16位宽度，可以存储任意的16位数据，包括数据、指针和地址等信息。XAR寄存器也可以用于算术和逻辑运算、比较和移位等操作，是DSP程序设计中非常重要的寄存器。在C语言中，可以使用指针来访问XAR寄存器中存储的数据