## 题目

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

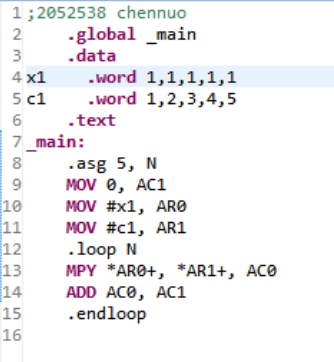
for (i=0，f=0; i<N; i++)

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

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

## 解答

1. C55X汇编代码

****.global \_main

.data

x1 .word 1,1,1,1,1

c1 .word 1,2,3,4,5

.text

\_main:

.asg 5, N ; N=5

MOV 0, AC1 ; AC1=0

MOV #x1, AR0 ; AR0=#x1，即AR0指向x1数组起始地址

MOV #c1, AR1 ; 同上

.loop N ; 定义循环N次

MPY \*AR0+, \*AR1+, AC0 ; AC0=\*AR0+\*AR1, AR0++, AR1++

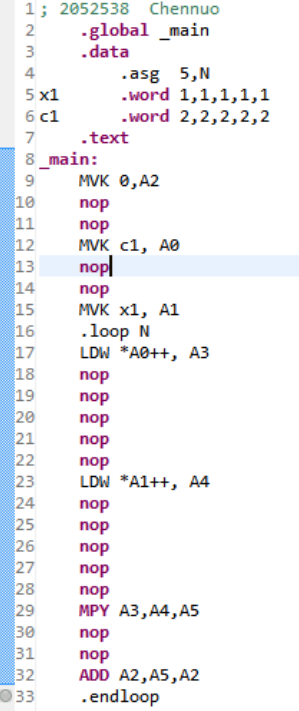
ADD AC0, AC1 ; AC1=AC0+AC1

.endloop ; 结束循环

1. C64X汇编代码：

.global \_main

.data

.asg 5, N ; .asg伪指令

x1 .word 1,1,1,1,1

c1 .word 2,2,2,2,2

.text

\_main:

MVK 0, A2 ; A2=0，A2相当于f

nop

nop

MVK c1, A0 ; A0=c1，A0存储c1的起始地址

nop

nop

MVK x1, A1 ; A1=x1，A1存储x1的起始地址

.loop N+1 ; 定义一个N+1的循环

LDW \*A0++, A3 ; A3=\*A0, A0++

nop

nop

nop

nop

nop

LDW \*A1++, A4 ; A4=\*A1, A1++

nop

nop

nop

nop

nop

MPY A3, A4, A5 ; A5=A3\*A4

nop

nop

ADD A2, A5, A2 ; A2=A2+A5

.endloop

### 比较

C55X和C64X的实现使用了专门的数字信号处理（DSP）指令，如MPY，以便更快地执行乘法运算，而ARM实现使用通用的乘法指令MUL。

C64X使用32位寄存器并支持64位算术运算，而C55X使用16位寄存器并支持32位算术运算。ARM也使用32位寄存器，但支持32位和64位算术运算。

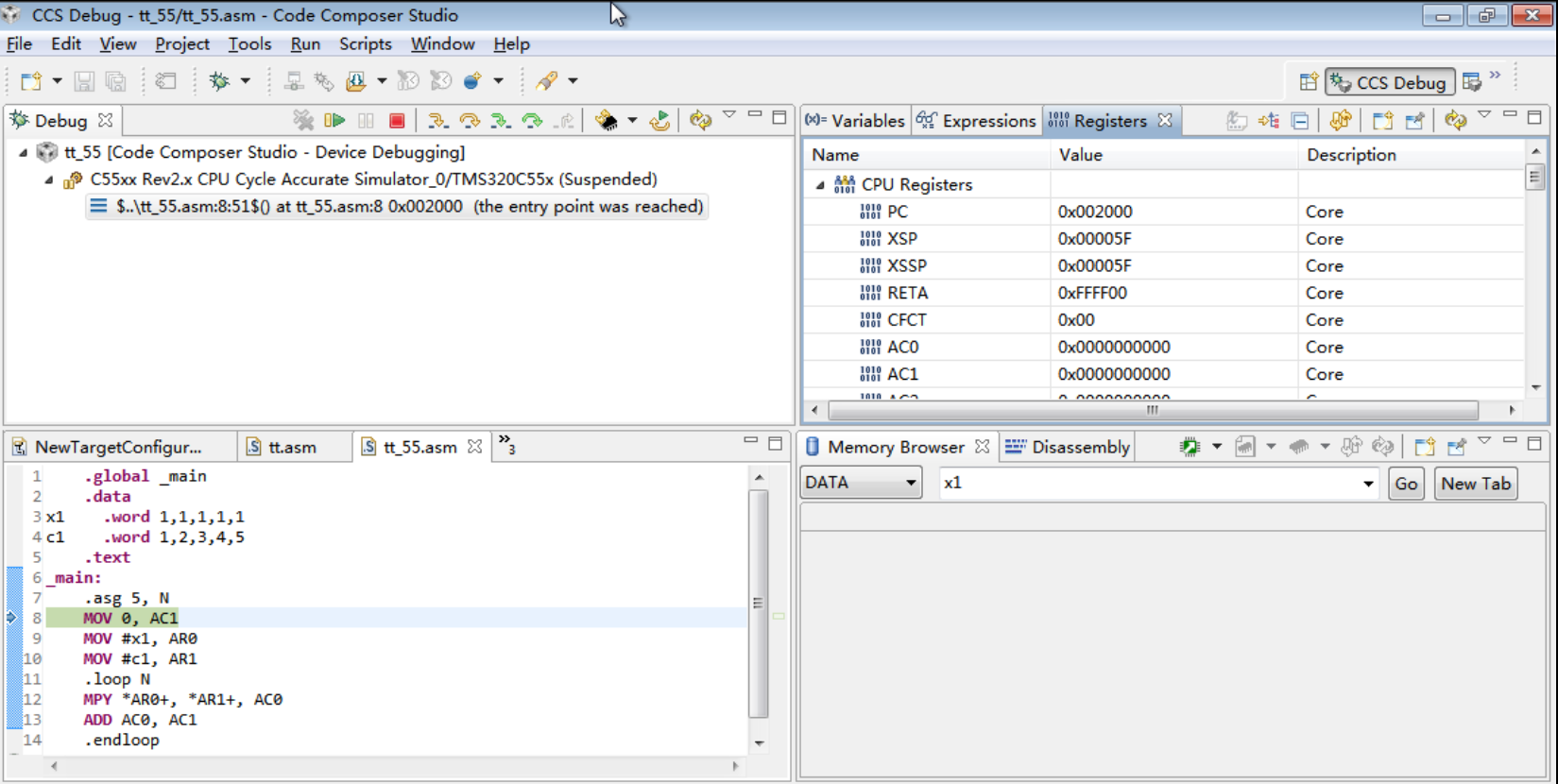
ARM汇编语言具有更多的控制流和寻址模式指令，使得编写复杂程序更加容易。C55X和C64X专门为DSP应用程序设计，因此指令数目较少。

C55X和C64X针对并行处理进行了优化，可以同时执行多个指令，而ARM针对顺序处理进行了优化，一次只能执行一条指令。

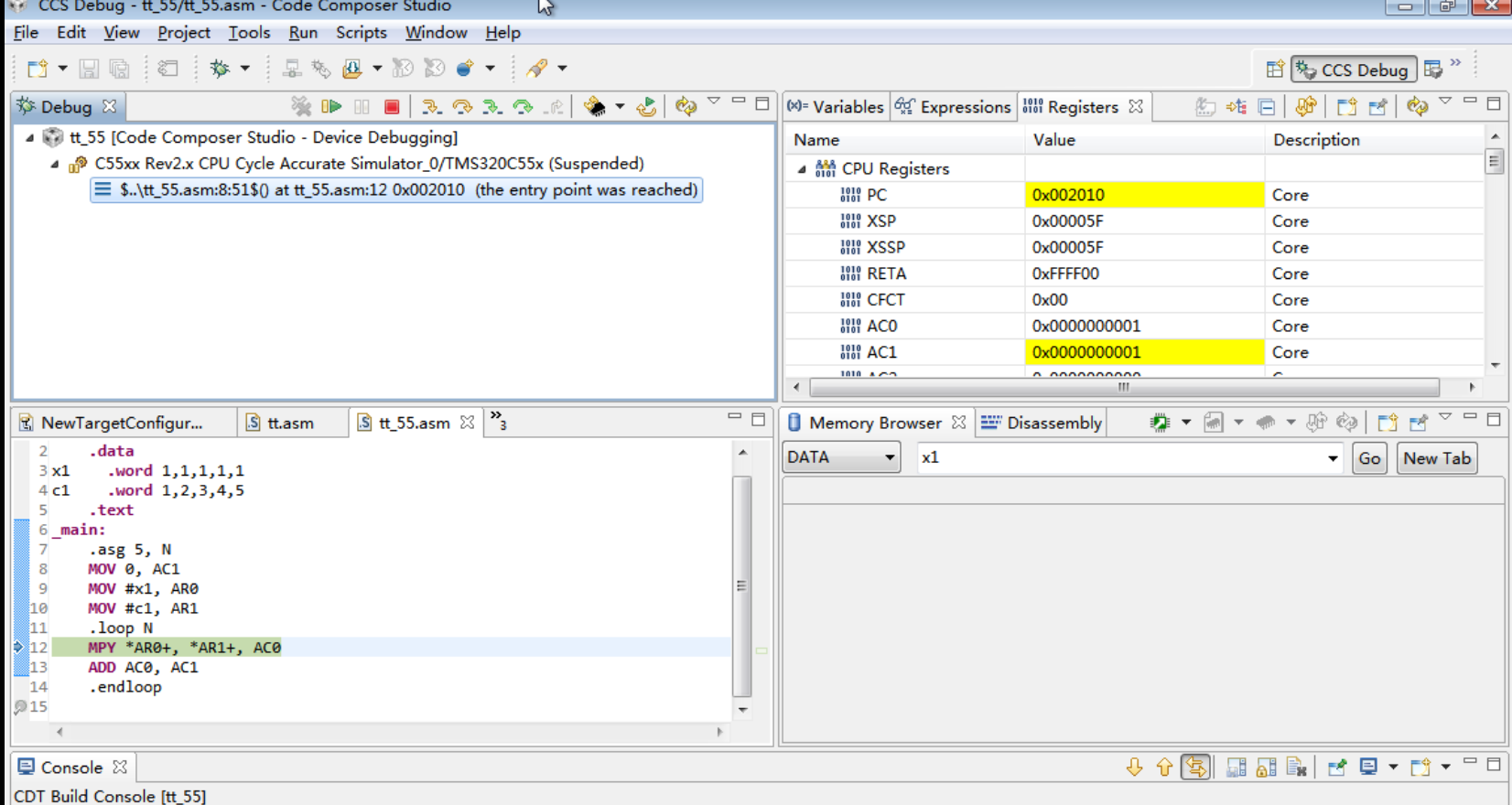
### 运行结果截图以及解析

#### C55x

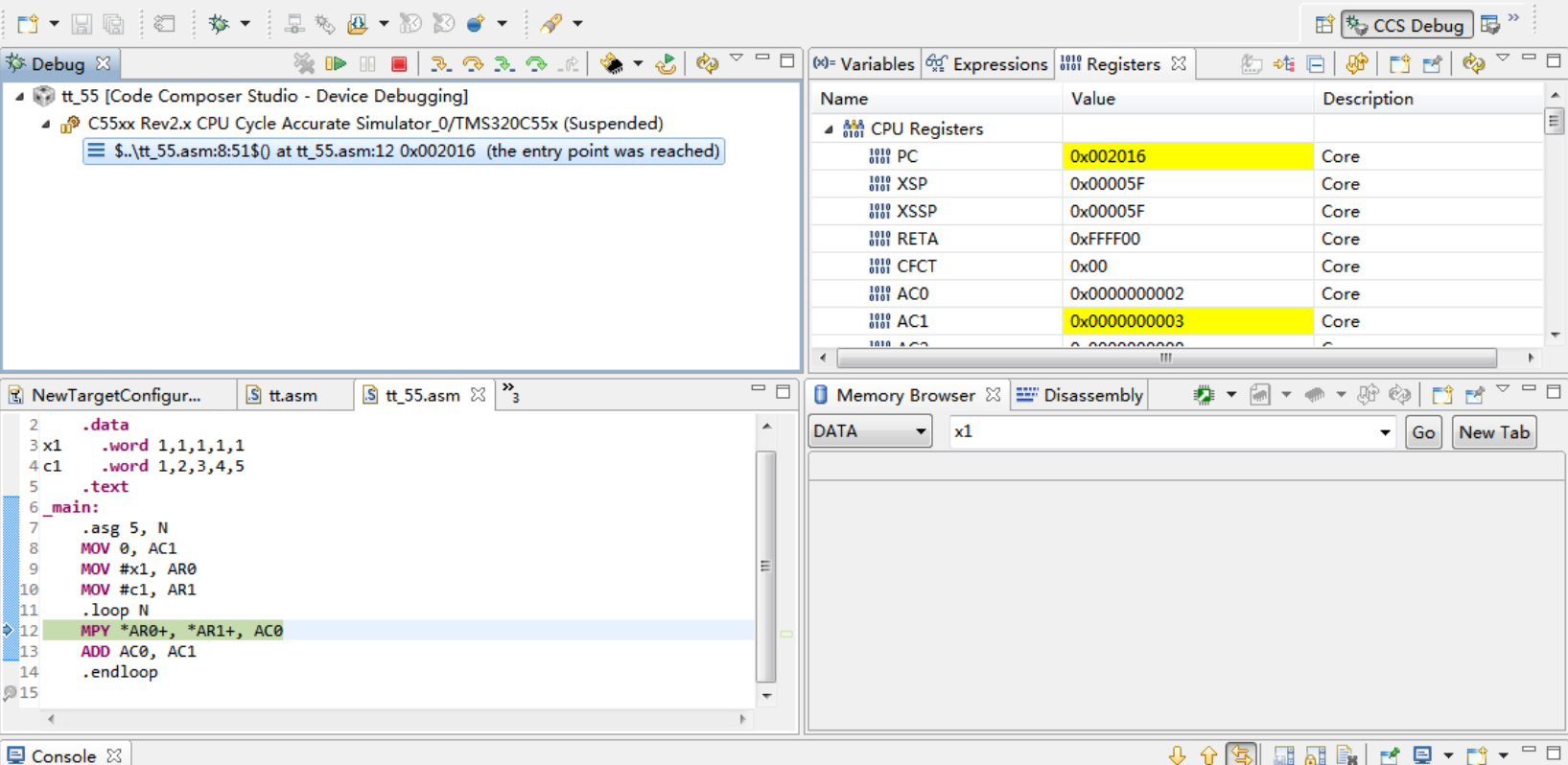
将最终结果存储在AC1中，也就是AC1代表f



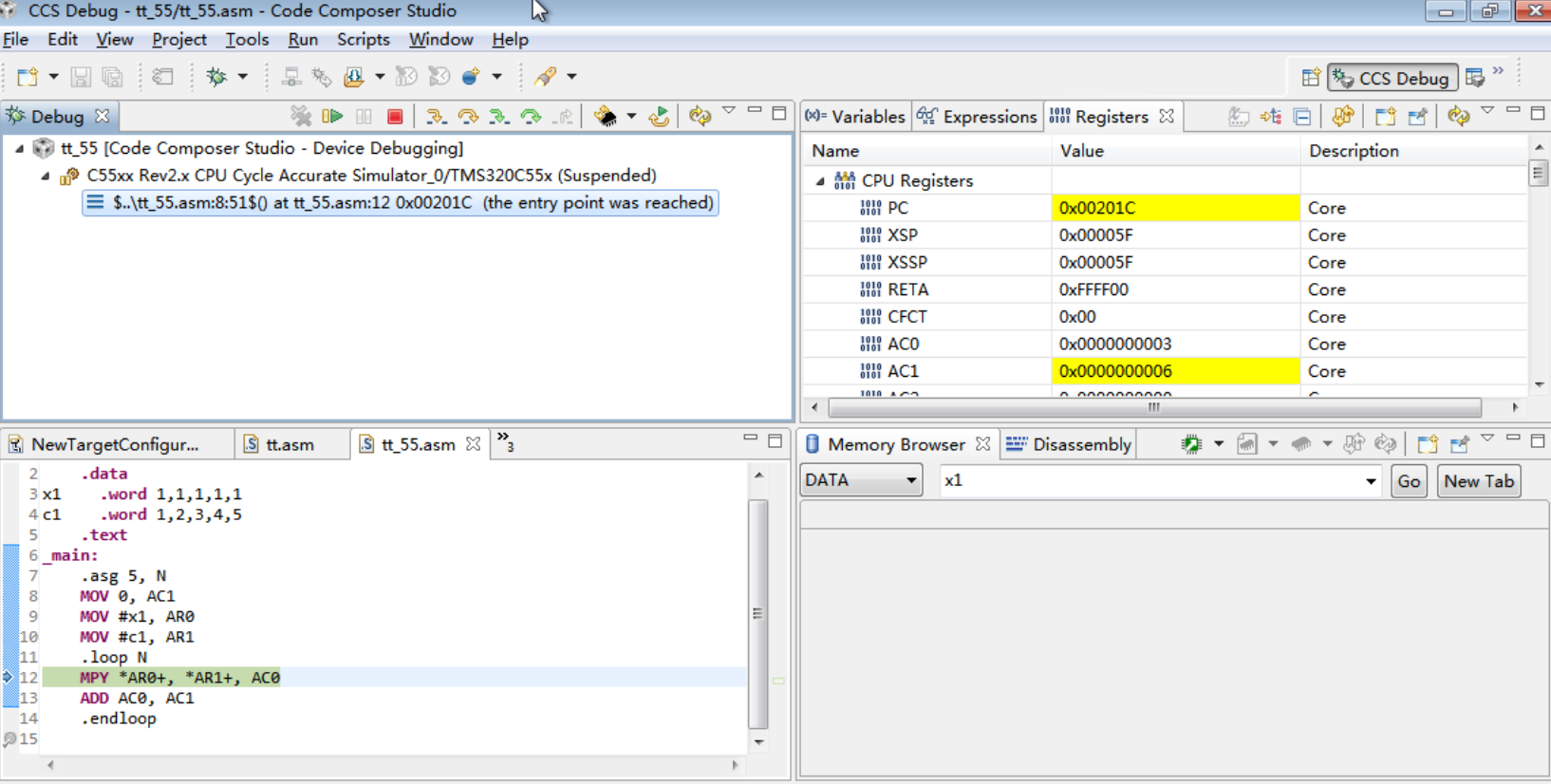
第一轮循环，AC0=1\*1, AC1=0+1\*1=1



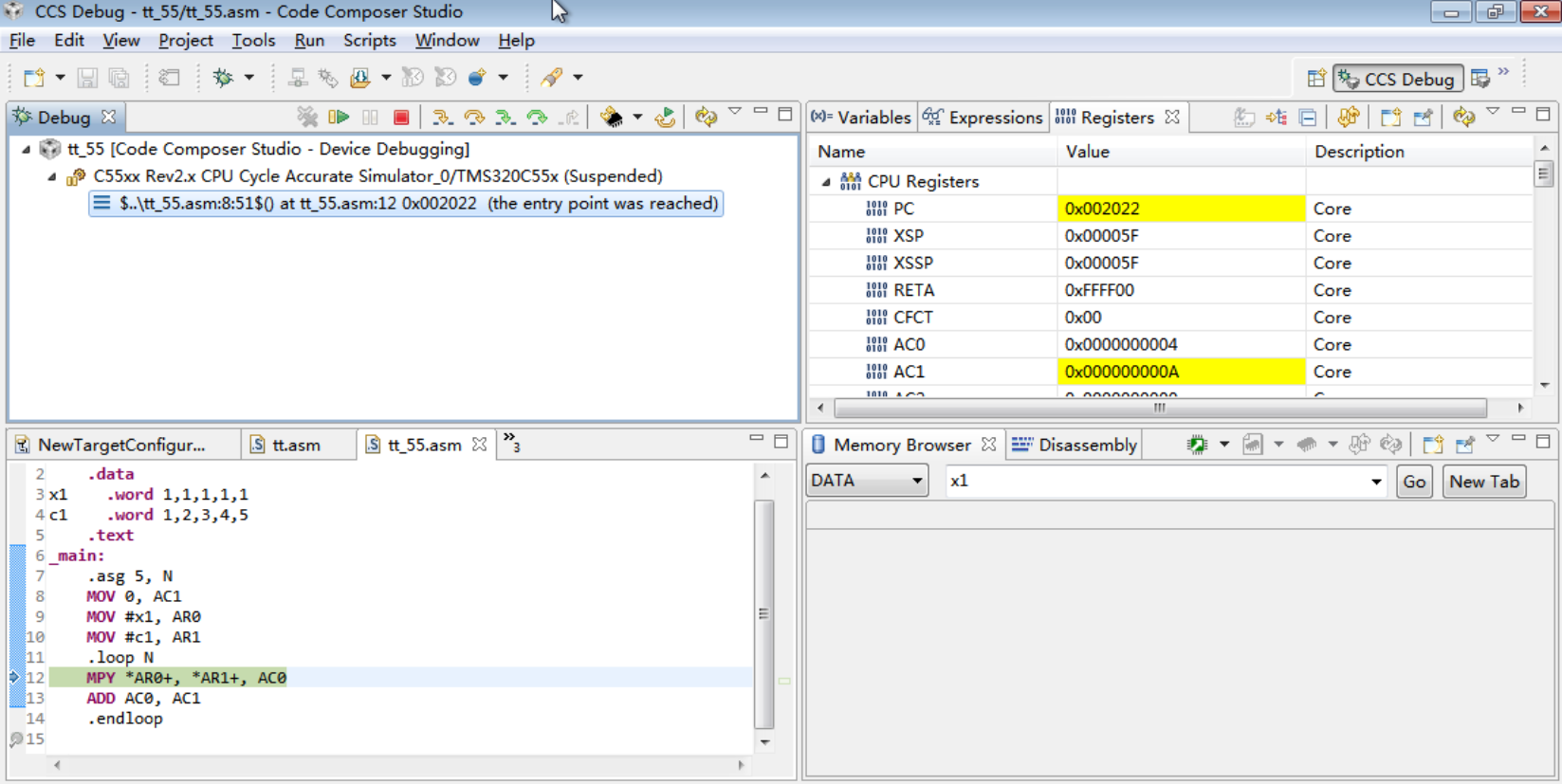
第二轮循环，AC0=1\*2, AC1=0+1\*1+1\*2=3



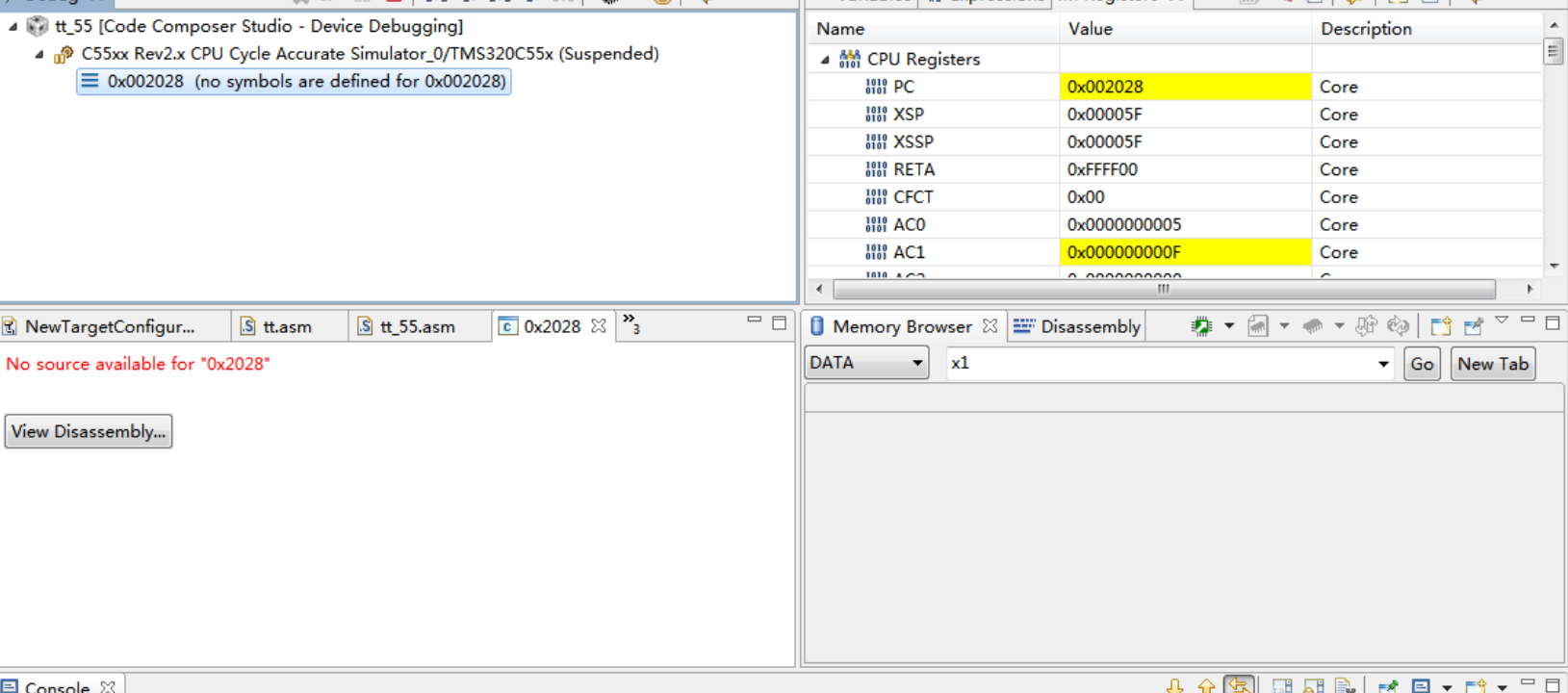
第三轮循环，AC0=1\*3, AC1=0+1\*1+1\*2+1\*3=6



第四轮循环结束，AC0=1\*4, AC1=0+1\*1+1\*2+1\*3+1\*4=A

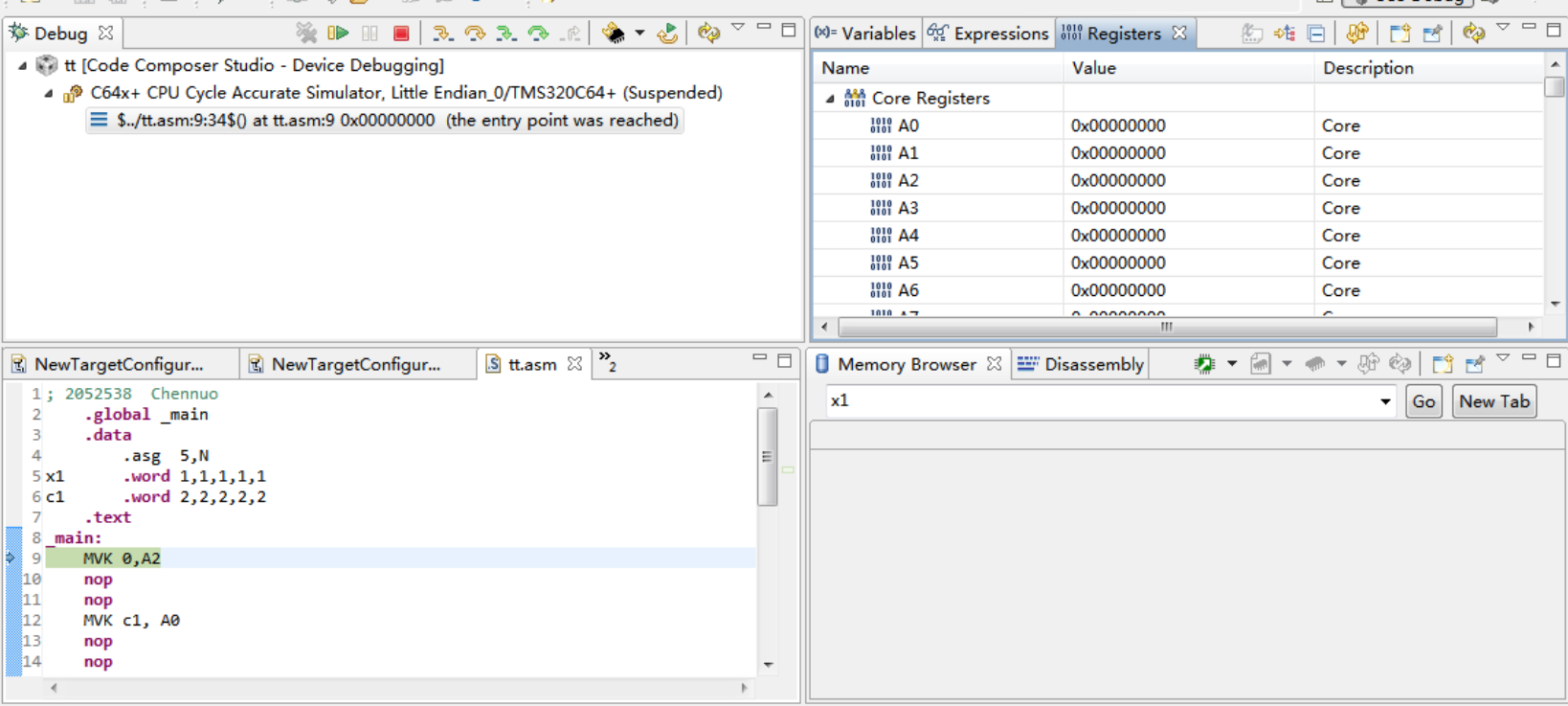


第五轮循环结束，AC0=1\*5, AC1=0+1\*1+1\*2+1\*3+1\*4+1\*5=F

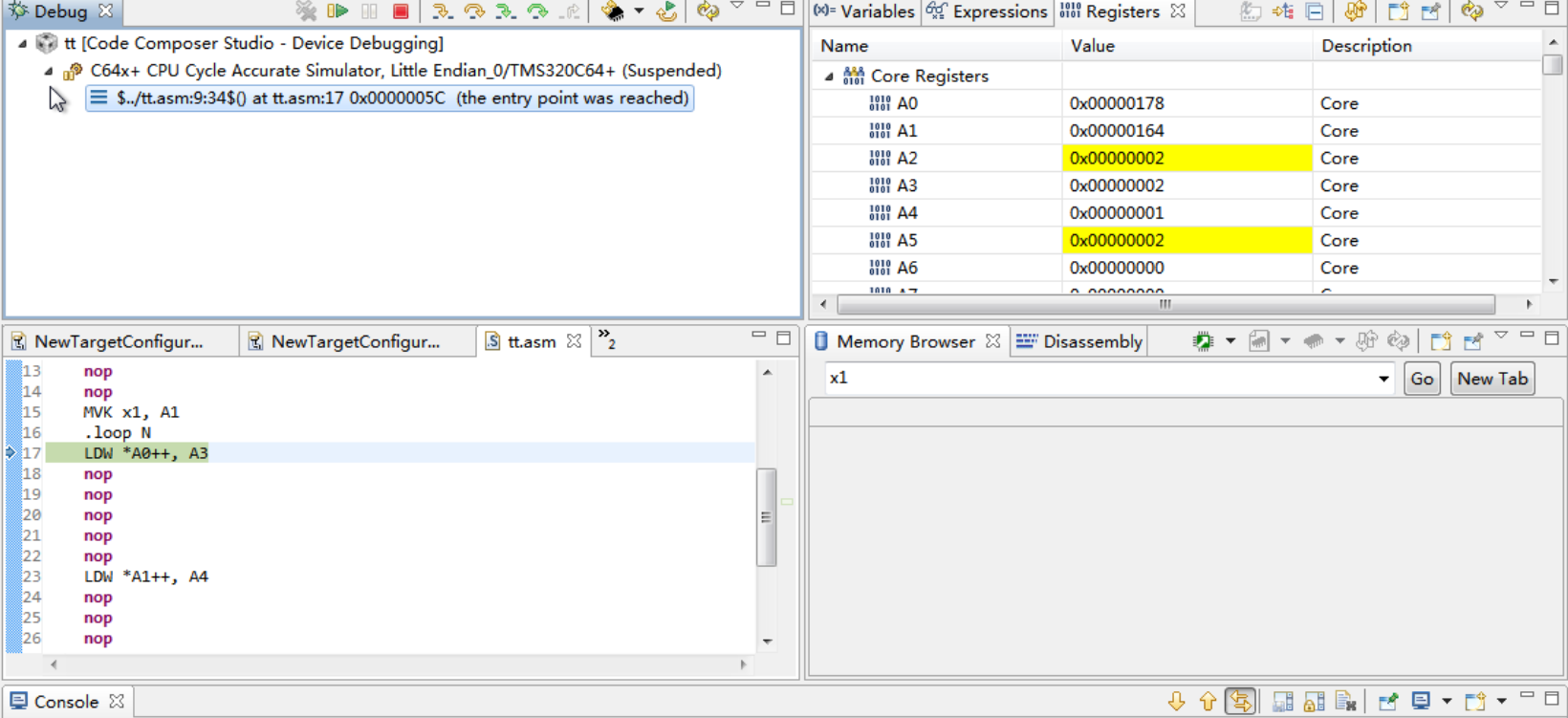


#### C64x

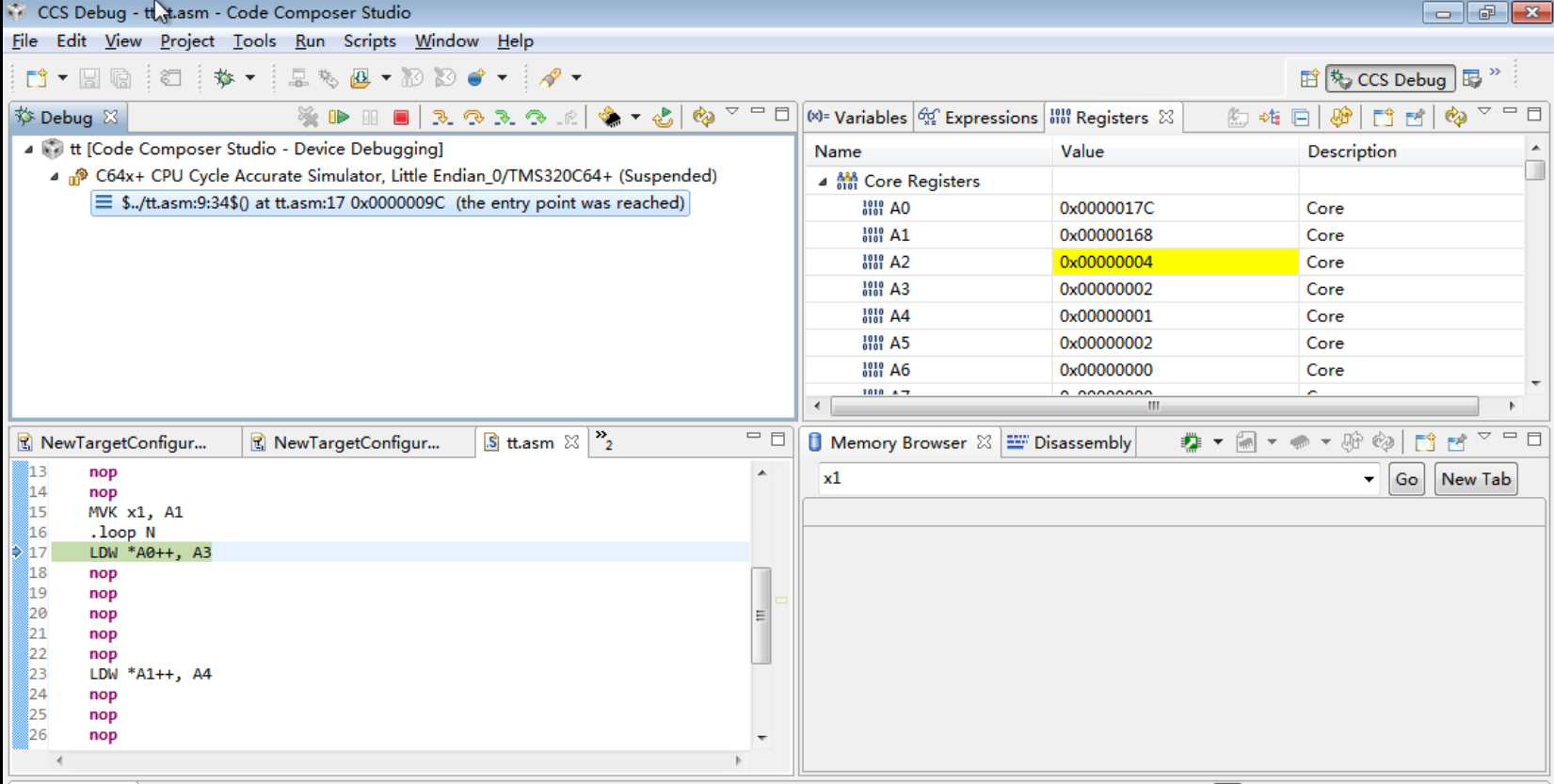
将最终结果存储在A2中，也就是A2代表f

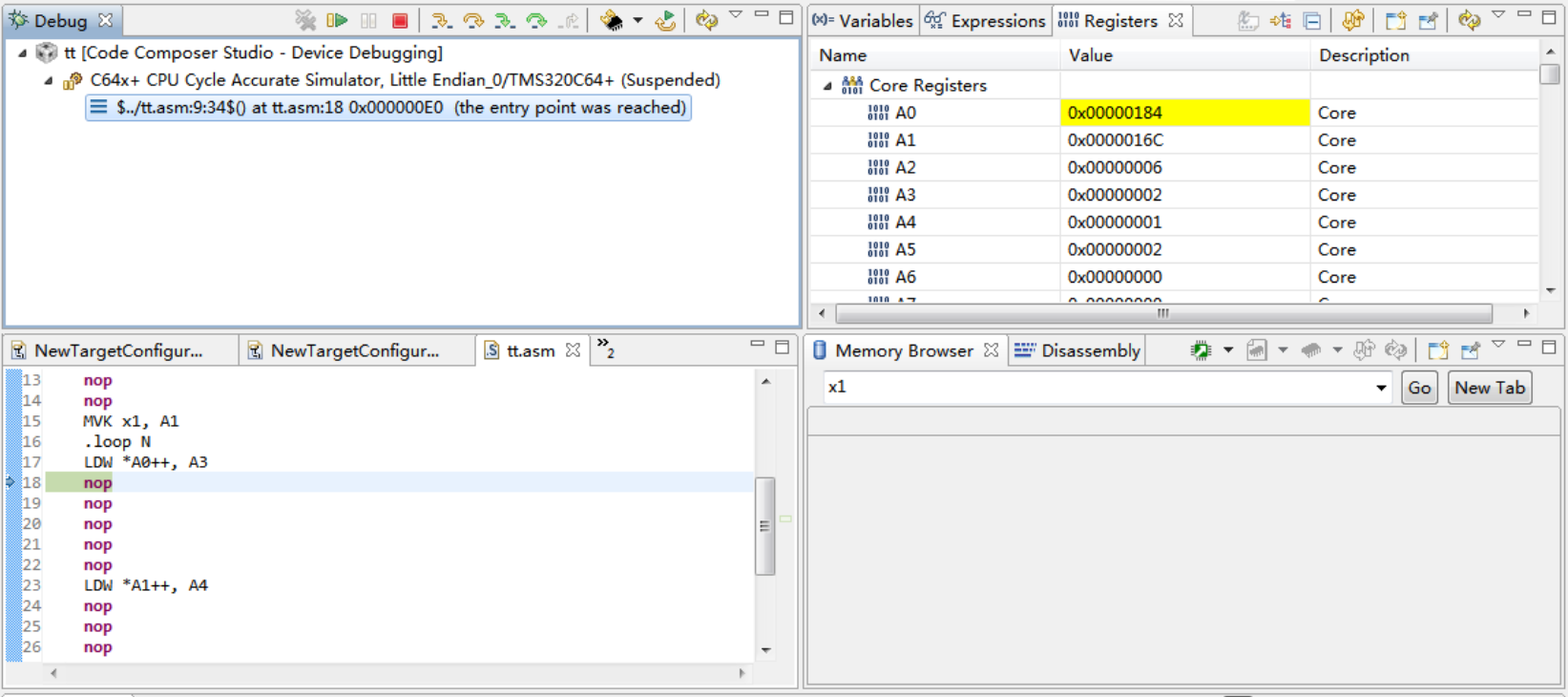


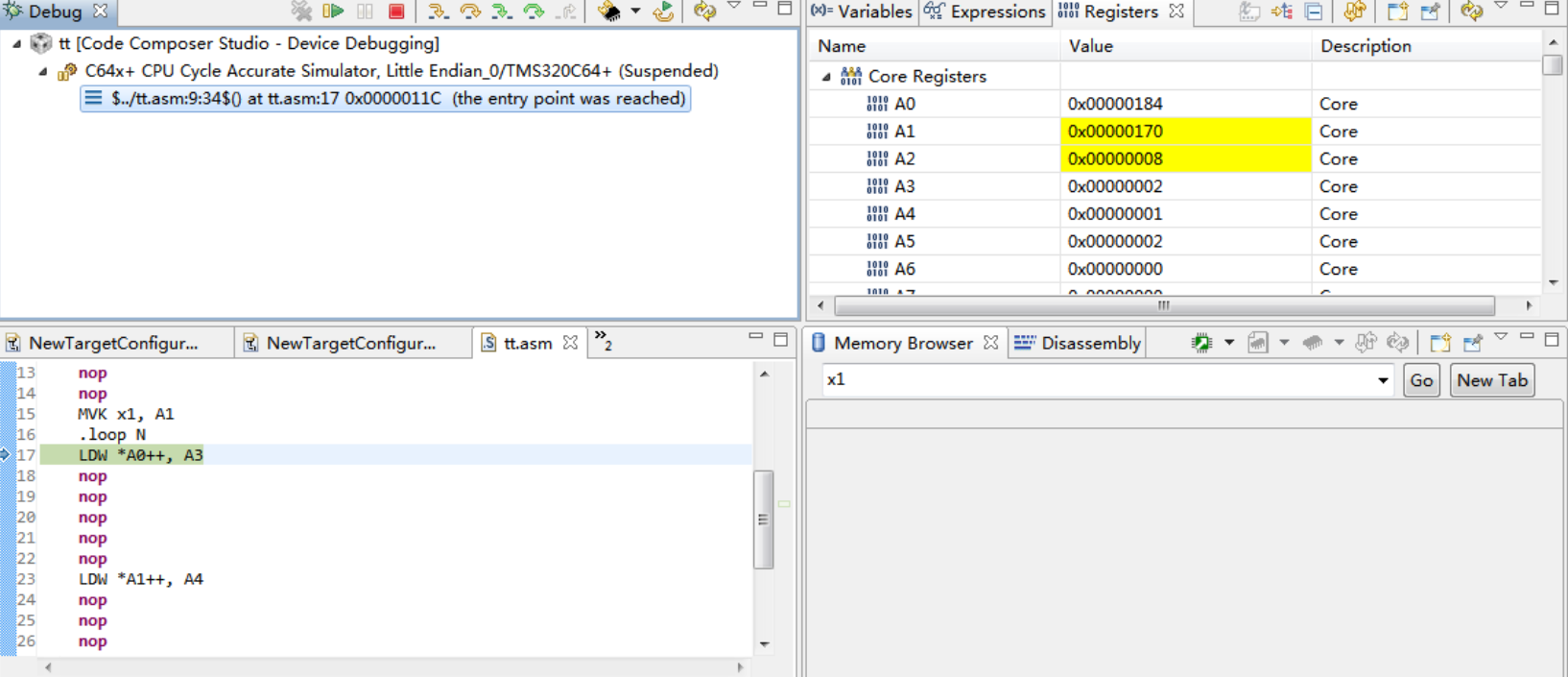
第一轮循环以后A2=1\*2=2



第二轮循环以后，A2=1\*2+1\*2=4







第五轮循环结束以后，A2=1\*2+1\*2+1\*2+1\*2+1\*2=0xA

代码执行完毕，所以提示No source available

