1.6.2 CPU的循环处理

船说：计算机基础

1.6.2 本节主要内容

01 汇编指令处理死循环—简单

02 条件指令介绍

03 GCD的简单实现

04 带条件的循环也不难

05 影响循环处理效率的因素

汇编指令处理死循环

```
1    START
2              mov       r0,#10
3    LOOP
4              sub       r0,r0,#1
5              b         LOOP
```

死循环的对比

```
12  int main()
13  {
14      int x;
15      while(1)
16          x--;
17
18      x = gcd(15,18);
19
20      return 0;
21  }
```

```
36  main:
37          sub     sp, sp, #8
38          mov     r0, #0
39          str     r0, [sp, #4]
40          b       .LBB1_1
41  .LBB1_1:
42          ldr     r0, [sp]
43          sub     r0, r0, #1
44          str     r0, [sp]
45          b       .LBB1_1
```

ARM内核中的可用条件语句

cond	Mnemonic extension	Meaning (integer)	Meaning (floating-point) [ab]	Condition flags
0000	EQ	等于		Z == 1
0001	NE	不等于		Z == 0
0010	CS [c]	进位置位/无符号数大于或相同		C == 1
0011	CC [d]	进位未置位/无符号数小于		C == 0
0100	MI	负数		N == 1
0101	PL	正数或零		N == 0
0110	VS	溢出		V == 1
0111	VC	无溢出		V == 0
1000	HI	无符号数大于		C == 1 and Z == 0
1001	LS	无符号数小于或相同		C == 0 or Z == 1
1010	GE	有符号数大于等于		N == V
1011	LT	有符号数小于		N != V
1100	GT	有符号数大于		Z == 0 and N == V
1101	LE	有符号数小于等于		Z == 1 or N != V
1110	None (AL) [e]	总是执行（无条件）		Any

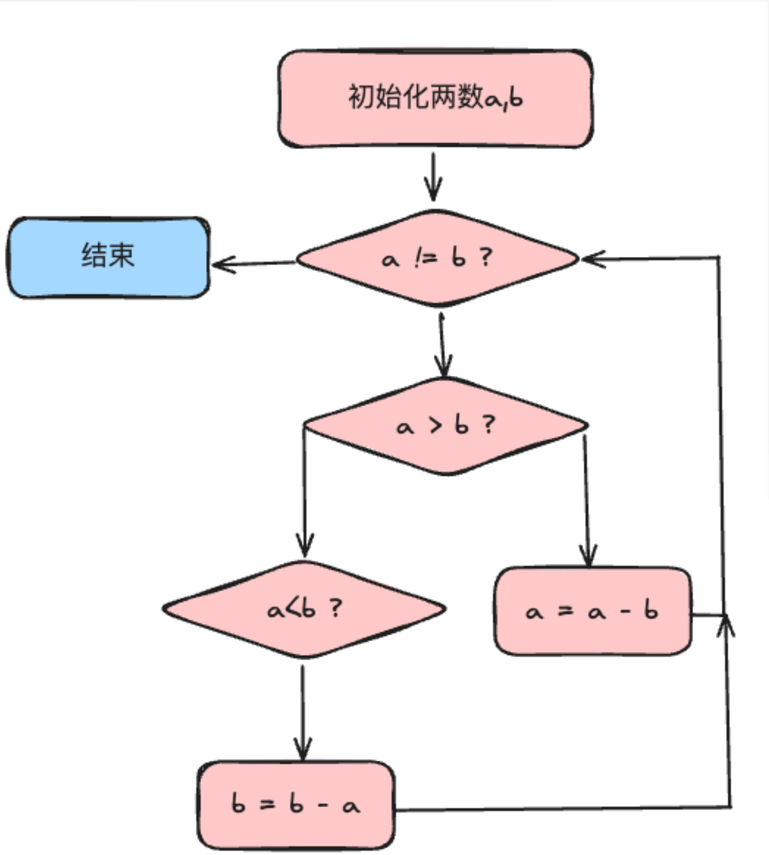
GCD（最大公约数）计算

辗转相减法：

比较要计算的两个数，并将较大数减去较小数，并覆盖掉原来的数，直到两个数相等，那么它们的最大公约数就是最终相等的这个数。

1. CPU的设计与结构

GCD的C语言实现



1. CPU的设计与结构

GCD的C语言实现

```c
#include <stdio.h>

int gcd(int a, int b) {
  while (a != b){ // a != b 一直循环
    if (a > b)   //如果 a > b
      a -= b;    //a = a - b
    else         //如果 a < b
      b -= a;    //b = b - a
  }              //继续去判断a是否等于b
  return a;
}

int main(){
  printf("gcd(27,84) = %d\n",gcd(27,84));
  return 0;
}
```

ARM内核中的可用条件语句

cond	Mnemonic extension	Meaning (integer)	Meaning (floating-point) [ab]	Condition flags
0000	EQ	Equal	Equal	Z == 1
0001	NE	Not equal	Not equal, or unordered	Z == 0
0010	CS [c]	Carry set	Greater than, equal, or unordered	C == 1
0011	CC [d]	Carry clear	Less than	C == 0
0100	MI	Minus, negative	Less than	N == 1
0101	PL	Plus, positive or zero	Greater than, equal, or unordered	N == 0
0110	VS	Overflow	Unordered	V == 1
0111	VC	No overflow	Not unordered	V == 0
1000	HI	Unsigned higher	Greater than, or unordered	C == 1 and Z == 0
1001	LS	Unsigned lower or same	Less than or equal	C == 0 or Z == 1
1010	GE	Signed greater than or equal	Greater than or equal	N == V
1011	LT	Signed less than	Less than, or unordered	N != V
1100	GT	Signed greater than	Greater than	Z == 0 and N == V
1101	LE	Signed less than or equal	Less than, equal, or unordered	Z == 1 or N != V
1110	None (AL) [e]	Always (unconditional)	Always (unconditional)	Any

ARM内核中的可用条件语句

1100	GT	Signed greater than	Greater than	Z == 0 and N == V
1101	LE	Signed less than or equal	Less than, equal, or unordered	Z == 1 or N != V

1. CPU的设计与结构

条件的应用

```
1   START
2           mov     r0,#18
3           mov     r1,#15
4   LOOP
5           cmp     r0,r1
6           beq     STOP
7           subgt   r0,r0,r1
8           suble   r1,r1,r0
9           b       LOOP
10
11  STOP
```

影响循环处理的部分因素

1、数据依赖性
2、分支预测
3、寄存器分配

1.6.2 条件循环指令处理

1. 跳转指令其实就是改变PC寄存器(R15)
2. 如果要带条件就看CPSR寄存器的NZCV
3. 条件执行的区域取决于CPSR的条件变化
4. 指令的条件码根据不同的数据的比较选择

海贼宝藏
专注IT教育在线学习平台

欢迎参与学习

WELCOME FOR YOUR JOINING

船说：计算机基础