第 5 章　CPU 性能验证

5.1　性能验证数学模型及算法程序

本实验采用的数学模式如下：a、b、c、d 分别为 4 个数组，它们的取值公式由下面的示例给出，现要求求得 c、d 数组中各个数字的值，并依次将其输出。

a[m]，b[m]，c[m]，d[m]；

a[0]＝0.0；

b[0]＝1.0；

a[i]＝a[i－1]＋i；

b[i]＝b[i－1]＋3i；

$$c[i]=\begin{cases}a[i], & 0\leqslant i\leqslant 9\\ a[i]+b[i], & 10\leqslant i\leqslant 29\\ (a[i]*b[i])\ll 1, & 30\leqslant i\leqslant 49\end{cases}$$

$$d[i]=\begin{cases}b[i]+c[i], & 0\leqslant i\leqslant 9\\ a[i]*c[i], & 10\leqslant i\leqslant 29\\ c[i]*b[i]/(d[i-1]\gg 1), & 30\leqslant i\leqslant 49\end{cases}$$

C 语言示例代码如图 5.1 所示。

```
#include "../include/minicrt.h"
#include "../include/system.h"
#include "../include/gpio.h"

int kk = 5;
static unsigned int gpio_data = 0;
static unsigned int gpio_tri = 0;
int a[50],b[50],c[50],d[50];
int i;

void test_app()
{
    for(i = 0;i<= 49;i++)
    {
        if(i == 0)
        {
            a[i] = 0;
            b[i] = 1;
            c[i] = a[i];
            d[i] = b[i]+c[i];
```

图 5.1　C 语言具体实现

```
            }
            else{
                a[i] =a[i-1]+i;
                b[i] =b[i-1]+3 * i;
                if(i<=9)
                {
                    c[i] =a[i];
                    d[i] =b[i]+c[i];
                }
                else if(i<=29)
                {
                    c[i] =a[i]+b[i];
                    d[i] =a[i] * c[i];
                }
                else
                {
                    c[i] = ((a[i] * b[i])) <<1;
                    d[i] =c[i] * b[i]/((d[i-1])>>1);
                }
            }
        }
    }

    //设置 gpio 是输出还是输入
    void set_gpio_tri(unsigned int value,bool is_input){
        unsigned int * gpio_tri_addr  = GPIO_TRI_ADDR;
        if(is_input ==true){
            //这个设计的目的是保证其他位的值不变
            //例如设置第 1 位为输入,is_input = true,value = 0x0000 0001;
            * gpio_tri_addr = gpio_tri | value;
        } else{
            //这个设计的目的是保证其他位的值不变
            //例如设置第 1 位为输出,is_input = false,value = 0xffff fffe;
            * gpio_tri_addr = gpio_tri & value;
        }
    }

    void write_gpio(void){
        unsigned int * gpio_data_addr = GPIO_DATA_ADDR;
        * gpio_data_addr    = gpio_data;
    }

    void digital_led(int id,int digital_num){
        unsigned int seg =0;
        switch(digital_num){
            case 0: seg =0x03;break; //0000 0011
            case 1: seg =0x9f;break; //1001 1111
```

图 5.1 （续）

```
            case 2: seg = 0x25;break; //0010 0101
            case 3: seg = 0x0d;break; //0000 1101
            case 4: seg = 0x99;break; //1001 1001
            case 5: seg = 0x49;break; //0100 1001
            case 6: seg = 0x41;break; //0100 0001
            case 7: seg = 0x1f;break; //0001 1111
            case 8: seg = 0x01;break; //0000 0001
            case 9: seg = 0x09;break; //0000 1001
            case 10: seg = 0x11;break; //0001 0001
            default: seg = 0x00;
        }

        switch(id){
            case 0: seg = 0xfe00| seg;break;              //1111 1110 fe
            case 1: seg = 0xfd00| seg;break;              //1111 1101 fd
            case 2: seg = 0xfb00| (seg&0xfe);break;       //1111 1011 fb
            case 3: seg = 0xf700| seg;break;              //1111 0111 f7
            case 4: seg = 0xef00| seg;break;              //1110 1111 ef
            case 5: seg = 0xdf00| seg;break;              //1101 1111 df
            case 6: seg = 0xbf00| seg;break;              //1011 1111 bf
            case 7: seg = 0x7f00| seg;break;              //0111 1111 7f
            default: seg = 0x00;
        }
        gpio_data = gpio_data & 0xffff0000;
        gpio_data = gpio_data | seg;
        write_gpio();
    }

    int main()
    {
        set_gpio_tri(0xffff0000,false);
        int send,oldsend;
        int cal_time;
        send = get_seconds();
        oldsend = send;
        while(1)
        {
            cal_time = 0;
            send = get_seconds();
            while(send == oldsend)
            {
                send = get_seconds();
                test_app();
                cal_time++;
            }
            oldsend = send;
            for(i=0;i<8;i++)
            {
```

图 5.1 （续）

```
            digital_led(i,cal_time%10);
            udelay(2000);
            cal_time = cal_time/10;
        }
    }
    return0;
}
```

图 5.1 （续）

汇编语言示例代码如下，注释供参考。

```
    .text
    .align    2
    .globl    main
    .set    nomips16
    .set    nomicromips

    j main
    exc:
    nop
    j exc
main:
    addi $t1,$0,0  #$t1 = a[0] = 0
    addi $t2,$0,1  #$t2 = b[0] = 1
    addi $t5,$0,0  #init $t5 = i = 0
    addi $t3,$t1,0 #$t3 = c[0] = a[0]
    addi $t4,$t2,1 #$t4 = d[0] = b[0]
    addi $t6,$0,10 #$t6 = 20 = 结束条件 1
    addi $t7,$0,30 #$t7 = 40 = 结束条件 2
    addi $t8,$0,50 #$t8 = 60 = 结束条件 3
loop:
    addi $t5,$t5,1         #i = i + 1
    add $t1,$t1,$t5     #a[i] = a[i-1] + i
    add $t2,$t2,$t5
    add $t2,$t2,$t5
    add $t2,$t2,$t5     #b[i] = b[i-1] + 3i
less_than_9:
    bge $t5,$t6,less_than_29 #if(i >= 20) jump
    addi $t3,$t1,0          #c[i] = a[i] + 0
    addi $t4,$t2,0         #d[i] = b[i] + 0
    j loop
less_than_29:
    bge $t5,$t7,less_than_49 #if(i == 40) jump
    add $t3,$t1,$t2         #c[i] = a[i]+b[i]
    mul $t4,$t1,$t3         #d[i] = a[i]*c[i]
```

```
    j loop
less_than_49:
    beq $t5,$t8,exc          #if(i == 60) jump
    mul $t3,$t1,$t2          #c[i] = a[i] * b[i]
    sll $t3,$t3,1           #c[i] = c[i] << 1
    mul $t9,$t3,$t2          #tmp = c[i] * b[i]
    sra $t4,$t4,1           #d[i-1] = d[i-1]>>1
    div $t4,$t9,$t4          #d[i] = tmp/d[i-1]
    j loop
```

5.2 性能验证程序下板测试过程与实现

5.2.1 下板过程

(1) 编译C语言程序为目标程序。

打开cmd窗口,进入toolchain文件夹下的build目录下,然后使用make clean & make命令编译成bin文件。编译过程如图5.2所示。

```
D:\>cd D:\my_subjects\computer_arch\Logon_cpu_trans_to_nexy_4\Logon_cpu_trans_to_nexy_4\toolchain\build

D:\my_subjects\computer_arch\Logon_cpu_trans_to_nexy_4\Logon_cpu_trans_to_nexy_4\toolchain\build>
D:\my_subjects\computer_arch\Logon_cpu_trans_to_nexy_4\Logon_cpu_trans_to_nexy_4\toolchain\build>
D:\my_subjects\computer_arch\Logon_cpu_trans_to_nexy_4\Logon_cpu_trans_to_nexy_4\toolchain\build>
D:\my_subjects\computer_arch\Logon_cpu_trans_to_nexy_4\Logon_cpu_trans_to_nexy_4\toolchain\build>make clean
del .\\src\\*.o *.om *.bin *.data *.mif *.asm

D:\my_subjects\computer_arch\Logon_cpu_trans_to_nexy_4\Logon_cpu_trans_to_nexy_4\toolchain\build>make
..\\bin\\mips-mti-elf-as -32 -mips32 -G0 src/startup_mytest.s -o src/startup_mytest.o
src/startup_mytest.s: Assembler messages:
src/startup_mytest.s: Warning: end of file not at end of a line; newline inserted
..\\bin\\mips-mti-elf-gcc   -mips32 -G0 -c src/main.c -o src/main.o
In file included from src/main.c:3:0:
src/main.c: In function 'set_gpio_tri':
src/../include/gpio.h:5:24: warning: initialization makes pointer from integer without a cast
 #define GPIO_TRI_ADDR  0xd0000004
                        ^

src/main.c:47:33: note: in expansion of macro 'GPIO_TRI_ADDR'
  unsigned int* gpio_tri_addr  = GPIO_TRI_ADDR;
                                 ^

src/main.c: In function 'write_gpio':
src/../include/gpio.h:4:26: warning: initialization makes pointer from integer without a cast
 #define GPIO_DATA_ADDR   0xd0000000
                          ^

src/main.c:61:33: note: in expansion of macro 'GPIO_DATA_ADDR'
  unsigned int* gpio_data_addr = GPIO_DATA_ADDR;
                                 ^

..\\bin\\mips-mti-elf-gcc   -mips32 -G0 -c src/entry.c -o src/entry.o
src/entry.c: In function 'exit':
src/entry.c:74:1: warning: 'noreturn' function does return
 }
 ^

..\\bin\\mips-mti-elf-gcc   -mips32 -G0 -c src/malloc.c -o src/malloc.o
src/malloc.c: In function 'brk':
src/malloc.c:83:12: warning: initialization makes integer from pointer without a cast
  int ret = &__HeapBase;
            ^

src/malloc.c:85:2: warning: return makes integer from pointer without a cast
  return (ret + end_data_segment);
  ^

..\\bin\\mips-mti-elf-gcc   -mips32 -G0 -c src/system.c -o src/system.o
..\\bin\\mips-mti-elf-gcc   -mips32 -G0 -c src/udelay.c -o src/udelay.o
..\\bin\\mips-mti-elf-ld -G0 -T ./mytest.ld -I./include  -o test.om src/startup_mytest.o src/main.o src/entry.o src/malloc.o src/sy
stem.o src/udelay.o
..\\bin\\mips-mti-elf-objcopy -O binary test.om test.bin
..\\bin\\mips-mti-elf-objdump -D test.om > ./test.asm

D:\my_subjects\computer_arch\Logon_cpu_trans_to_nexy_4\Logon_cpu_trans_to_nexy_4\toolchain\build>
```

图5.2 cmd窗口运行结果

(2) 进行 synthesis(综合)以及 implementation(布线)然后生成 bit 流。

(3) 将 test.bin 文件烧入开发板的 flash 中,如图 5.3 所示。

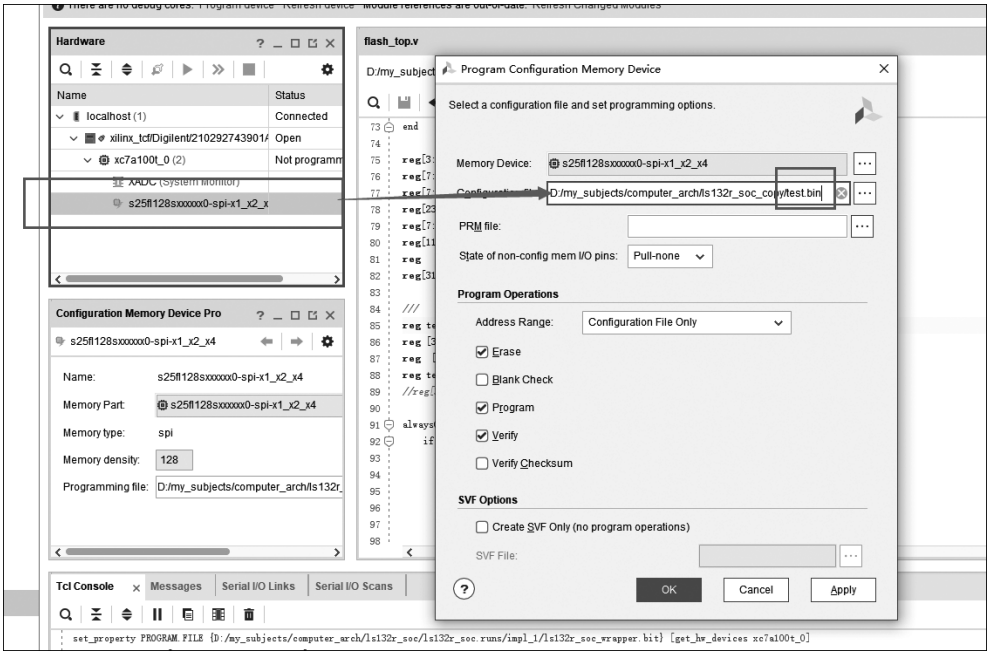

图 5.3　将 bin 文件烧入开发板中

(4) 将 FPGA 程序烧入开发板中运行,结果如图 5.4 所示。

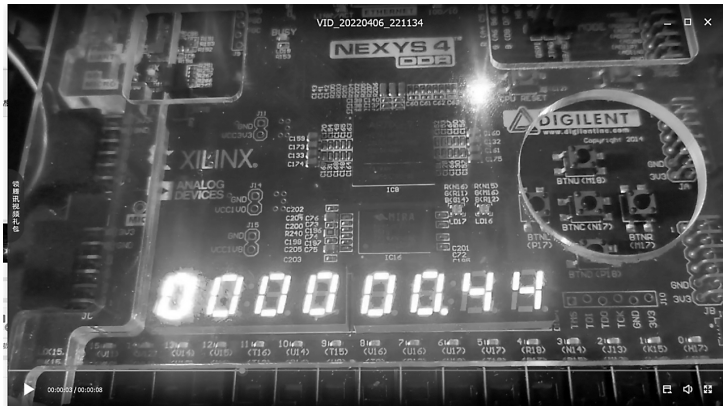

图 5.4　开发板下板结果

可以看到该程序每秒运行的次数。

5.2.2　程序性能分析

查看此 C 语言程序中性能测试函数的汇编代码,查看运行一次该函数大概要运行多

少次定点运算。

机器代码如下。

```
bfc00490 <test_app>:
bfc00490: 27bdfff8   addiu  sp,sp,-8
bfc00494: afbe0004   sw  s8,4(sp)
bfc00498: 03a0f025   move  s8,sp
bfc0049c: 3c02c000   lui  v0,0xc000
bfc004a0: ac400158   sw  zero,344(v0)
bfc004a4: 10000101   b  bfc008ac <test_app+0x41c>
bfc004a8: 00000000   nop
bfc004ac: 3c02c000   lui  v0,0xc000
bfc004b0: 8c420158   lw  v0,344(v0)
bfc004b4: 14400036   bnez  v0,bfc00590 <test_app+0x100>
bfc004b8: 00000000   nop
bfc004bc: 3c02c000   lui  v0,0xc000
bfc004c0: 8c430158   lw  v1,344(v0)
bfc004c4: 3c02c000   lui  v0,0xc000
bfc004c8: 00031880   sll  v1,v1,0x2
bfc004cc: 244202ec   addiu  v0,v0,748
bfc004d0: 00621021   addu  v0,v1,v0
bfc004d4: ac400000   sw  zero,0(v0)
bfc004d8: 3c02c000   lui  v0,0xc000
bfc004dc: 8c430158   lw  v1,344(v0)
bfc004e0: 3c02c000   lui  v0,0xc000
bfc004e4: 00031880   sll  v1,v1,0x2
bfc004e8: 24420090   addiu  v0,v0,144
bfc004ec: 00621021   addu  v0,v1,v0
bfc004f0: 24030001   li  v1,1
bfc004f4: ac430000   sw  v1,0(v0)
bfc004f8: 3c02c000   lui  v0,0xc000
bfc004fc: 8c440158   lw  a0,344(v0)
bfc00500: 3c02c000   lui  v0,0xc000
bfc00504: 8c430158   lw  v1,344(v0)
bfc00508: 3c02c000   lui  v0,0xc000
bfc0050c: 00031880   sll  v1,v1,0x2
bfc00510: 244202ec   addiu  v0,v0,748
bfc00514: 00621021   addu  v0,v1,v0
bfc00518: 8c430000   lw  v1,0(v0)
bfc0051c: 3c02c000   lui  v0,0xc000
bfc00520: 00042080   sll  a0,a0,0x2
bfc00524: 2442015c   addiu  v0,v0,348
bfc00528: 00821021   addu  v0,a0,v0
bfc0052c: ac430000   sw  v1,0(v0)
```

```
bfc00530: 3c02c000   lui  v0,0xc000
bfc00534: 8c440158   lw  a0,344(v0)
bfc00538: 3c02c000   lui  v0,0xc000
bfc0053c: 8c430158   lw  v1,344(v0)
bfc00540: 3c02c000   lui  v0,0xc000
bfc00544: 00031880   sll  v1,v1,0x2
bfc00548: 24420090   addiu  v0,v0,144
bfc0054c: 00621021   addu  v0,v1,v0
bfc00550: 8c430000   lw  v1,0(v0)
bfc00554: 3c02c000   lui  v0,0xc000
bfc00558: 8c450158   lw  a1,344(v0)
bfc0055c: 3c02c000   lui  v0,0xc000
bfc00560: 00052880   sll  a1,a1,0x2
bfc00564: 2442015c   addiu  v0,v0,348
bfc00568: 00a21021   addu  v0,a1,v0
bfc0056c: 8c420000   lw  v0,0(v0)
bfc00570: 00621821   addu  v1,v1,v0
bfc00574: 3c02c000   lui  v0,0xc000
bfc00578: 00042080   sll  a0,a0,0x2
bfc0057c: 24420224   addiu  v0,v0,548
bfc00580: 00821021   addu  v0,a0,v0
bfc00584: ac430000   sw  v1,0(v0)
bfc00588: 100000c3   b  bfc00898 <test_app+0x408>
bfc0058c: 00000000   nop
bfc00590: 3c02c000   lui  v0,0xc000
bfc00594: 8c440158   lw  a0,344(v0)
bfc00598: 3c02c000   lui  v0,0xc000
bfc0059c: 8c420158   lw  v0,344(v0)
bfc005a0: 2443ffff   addiu  v1,v0,-1
bfc005a4: 3c02c000   lui  v0,0xc000
bfc005a8: 00031880   sll  v1,v1,0x2
bfc005ac: 244202ec   addiu  v0,v0,748
bfc005b0: 00621021   addu  v0,v1,v0
bfc005b4: 8c430000   lw  v1,0(v0)
bfc005b8: 3c02c000   lui  v0,0xc000
bfc005bc: 8c420158   lw  v0,344(v0)
bfc005c0: 00621821   addu  v1,v1,v0
bfc005c4: 3c02c000   lui  v0,0xc000
bfc005c8: 00042080   sll  a0,a0,0x2
bfc005cc: 244202ec   addiu  v0,v0,748
bfc005d0: 00821021   addu  v0,a0,v0
bfc005d4: ac430000   sw  v1,0(v0)
bfc005d8: 3c02c000   lui  v0,0xc000
bfc005dc: 8c440158   lw  a0,344(v0)
```

```
bfc005e0: 3c02c000   lui  v0,0xc000
bfc005e4: 8c420158   lw  v0,344(v0)
bfc005e8: 2443ffff   addiu  v1,v0,-1
bfc005ec: 3c02c000   lui  v0,0xc000
bfc005f0: 00031880   sll  v1,v1,0x2
bfc005f4: 24420090   addiu  v0,v0,144
bfc005f8: 00621021   addu  v0,v1,v0
bfc005fc: 8c450000   lw  a1,0(v0)
bfc00600: 3c02c000   lui  v0,0xc000
bfc00604: 8c430158   lw  v1,344(v0)
bfc00608: 00601025   move  v0,v1
bfc0060c: 00021040   sll  v0,v0,0x1
bfc00610: 00431021   addu  v0,v0,v1
bfc00614: 00a21821   addu  v1,a1,v0
bfc00618: 3c02c000   lui  v0,0xc000
bfc0061c: 00042080   sll  a0,a0,0x2
bfc00620: 24420090   addiu  v0,v0,144
bfc00624: 00821021   addu  v0,a0,v0
bfc00628: ac430000   sw  v1,0(v0)
bfc0062c: 3c02c000   lui  v0,0xc000
bfc00630: 8c420158   lw  v0,344(v0)
bfc00634: 2842000a   slti  v0,v0,10
bfc00638: 10400027   beqz  v0,bfc006d8 <test_app+0x248>
bfc0063c: 00000000   nop
bfc00640: 3c02c000   lui  v0,0xc000
bfc00644: 8c440158   lw  a0,344(v0)
bfc00648: 3c02c000   lui  v0,0xc000
bfc0064c: 8c430158   lw  v1,344(v0)
bfc00650: 3c02c000   lui  v0,0xc000
bfc00654: 00031880   sll  v1,v1,0x2
bfc00658: 244202ec   addiu  v0,v0,748
bfc0065c: 00621021   addu  v0,v1,v0
bfc00660: 8c430000   lw  v1,0(v0)
bfc00664: 3c02c000   lui  v0,0xc000
bfc00668: 00042080   sll  a0,a0,0x2
bfc0066c: 2442015c   addiu  v0,v0,348
bfc00670: 00821021   addu  v0,a0,v0
bfc00674: ac430000   sw  v1,0(v0)
bfc00678: 3c02c000   lui  v0,0xc000
bfc0067c: 8c440158   lw  a0,344(v0)
bfc00680: 3c02c000   lui  v0,0xc000
bfc00684: 8c430158   lw  v1,344(v0)
bfc00688: 3c02c000   lui  v0,0xc000
bfc0068c: 00031880   sll  v1,v1,0x2
```

```
bfc00690: 24420090   addiu  v0,v0,144
bfc00694: 00621021   addu  v0,v1,v0
bfc00698: 8c430000   lw  v1,0(v0)
bfc0069c: 3c02c000   lui  v0,0xc000
bfc006a0: 8c450158   lw  a1,344(v0)
bfc006a4: 3c02c000   lui  v0,0xc000
bfc006a8: 00052880   sll  a1,a1,0x2
bfc006ac: 2442015c   addiu  v0,v0,348
bfc006b0: 00a21021   addu  v0,a1,v0
bfc006b4: 8c420000   lw  v0,0(v0)
bfc006b8: 00621821   addu  v1,v1,v0
bfc006bc: 3c02c000   lui  v0,0xc000
bfc006c0: 00042080   sll  a0,a0,0x2
bfc006c4: 24420224   addiu  v0,v0,548
bfc006c8: 00821021   addu  v0,a0,v0
bfc006cc: ac430000   sw  v1,0(v0)
bfc006d0: 10000071   b  bfc00898 <test_app+0x408>
bfc006d4: 00000000   nop
bfc006d8: 3c02c000   lui  v0,0xc000
bfc006dc: 8c420158   lw  v0,344(v0)
bfc006e0: 2842001e   slti  v0,v0,30
bfc006e4: 10400030   beqz  v0,bfc007a8 <test_app+0x318>
bfc006e8: 00000000   nop
bfc006ec: 3c02c000   lui  v0,0xc000
bfc006f0: 8c440158   lw  a0,344(v0)
bfc006f4: 3c02c000   lui  v0,0xc000
bfc006f8: 8c430158   lw  v1,344(v0)
bfc006fc: 3c02c000   lui  v0,0xc000
bfc00700: 00031880   sll  v1,v1,0x2
bfc00704: 244202ec   addiu  v0,v0,748
bfc00708: 00621021   addu  v0,v1,v0
bfc0070c: 8c430000   lw  v1,0(v0)
bfc00710: 3c02c000   lui  v0,0xc000
bfc00714: 8c450158   lw  a1,344(v0)
bfc00718: 3c02c000   lui  v0,0xc000
bfc0071c: 00052880   sll  a1,a1,0x2
bfc00720: 24420090   addiu  v0,v0,144
bfc00724: 00a21021   addu  v0,a1,v0
bfc00728: 8c420000   lw  v0,0(v0)
bfc0072c: 00621821   addu  v1,v1,v0
bfc00730: 3c02c000   lui  v0,0xc000
bfc00734: 00042080   sll  a0,a0,0x2
bfc00738: 2442015c   addiu  v0,v0,348
bfc0073c: 00821021   addu  v0,a0,v0
```