测试程序一 bubble程序

源程序：

void main()

{

int n[10] = { 25,35,68,79,21,13,98,7,16,62 };//定义一个大小为10的数组

int i, j, temp;

for (i = 1; i <= 9; i++)//数组内有10个数，那么就应该比较10-1=9轮

{

for (j = 0; j <= 9 - i; j++)//内层循环比较的是当前一轮的比较次数

{

if (n[j] > n[j + 1])//相邻两个数如果逆序，则交换位置

{

temp = n[j];

n[j] = n[j + 1];

n[j + 1] = temp;

}

}

}

printf("排序过后的数顺序:\n");

for (i = 0; i < 10; i++)

printf("%-4d", n[i]);

printf("\n");

}

Riscv gcc利用rv32i指令集编译汇编后利用Riscv 工具链反汇编后的汇编代码分析如下：

|  |  |
| --- | --- |
| 汇编指令 | 出现次数 |
| add | 7 |
| addi | 16 |
| auipc | 3 |
| bge | 4 |
| j | 3 |
| jalr | 3 |
| li | 5 |
| lui | 3 |
| lw | 32 |
| nop | 1 |
| ret | 1 |
| slli | 7 |
| sub | 1 |
| sw | 21 |

需要注意的是其中的li(load immediate)指令其实是ADDI指令将x0寄存器加12位立即数实现的。Nop指令是通过ADDI指令将x0加0并送入x0实现的。

测试程序二 fiboncci程序

源程序：

int fib(int m)

{

if (m==1||m==2)

return 1;

int a=1,b=1,aw=0;

while(m>=2)

{

aw=aw+a;

a=b;

b=aw;

m=m-1;

}

return aw;

}

int main()

{

int n;

scanf("%d",&n);

printf("%d",fib(n));

return 0;

}

Riscv gcc利用rv32i指令集编译汇编后利用Riscv 工具链反汇编后的汇编代码分析如下：

|  |  |
| --- | --- |
| 汇编指令 | 出现次数 |
| add | 1 |
| addi | 8 |
| auipc | 3 |
| beq | 1 |
| blt | 1 |
| bne | 1 |
| j | 2 |
| jalr | 3 |
| li | 7 |
| lui | 2 |
| lw | 13 |
| mv | 8 |
| ret | 2 |
| sw | 11 |

目前项目的整体架构暂定为如下的六级流水线



现在还在进行模块的代码编写与调试。