《程序设计课程设计》实验报告

实验名称　　《冯诺依曼计算机 CPU 模拟》（多核版）概要设计<版本号>

班　　级

姓　　名

1 高层数据结构设计

(包括：重要的数据常量定义、数据变量定义，即各模块要共享的数据类型和参数设计，相当于头文件内容，加文字描述)

3.1 全局常量/变量定义

```
extern int Memory[32767],lag1,lag2,sum;//Memory 用来保存内存数据   lag
判断是否继续执行指令 sum 代表指令行数
typedef struct inst
{
    int code; // 指令
   short p1, p2 ,p3;// 参数 1，2,3   p1，p2 为执行指令需用到的数 p3 用来保存指令保存数据对应的寄存器标号
} inst_t;
typedef struct vm_state
{
    int ip; // 指令 ptr
    int flag; // 记录最后判断的标志
    inst_t *code;
    short ax[9];
} vm_state_t;
extern inst_t sample_code[100];//存指令
#define N 100
#define IADD    1 // 加法
#define LADD    111
#define ISUB    2 // 减法
#define LSUB    222
#define IMUL    3 // 乘法
#define LMUL    333
#define IDIV    4 // 除法
#define LDIV    444
#define ICMP    5 // 判断
#define LCMP    555
```

```
#define IJMP     6 //  4种跳转
#define LJMP     66
#define NJMP     666
#define MJMP     6666
#define IMOV     7 // 赋值
#define IGIV     8 // 赋值
#define ISED     9 // 赋值
#define IDOR     10 //休眠
#define IPUT     11 // 输入
#define IOUT     12 // 输出
#define ILOG     13//逻辑指令与
#define LLOG     1313
#define ILOV     14//逻辑指令或
#define LLOV     1414
#define ILON     15 //逻辑指令非
#define LLON     1515
#define ILOC     16 //加锁
#define IUNL     17 //解锁
#define ISTOP    255 //停机

#define FNA      0 //比较后相等为0
#define FEQ      1 //寄存器1中的大 置为1
#define FNE      -1 //立即数大    置为-1
pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;//
锁1
pthread_mutex_t mutex1 = PTHREAD_MUTEX_INITIALIZER;//
锁2
```

```
extern char
Code1[100][9],Code2[100][5],Code3[100][5],Code4[100][17]
,CODESET[100][17],codeSegment[100][33],CODE[100][32];//
//Code1 保存指令前 8 位，Code2 保存 9—12 位，Code3 保存 13—16
位，Code4 保存后 16 位，CODESET 保存前 16 位，codeSegment 和
CODE 保存完整指令
```

2 系统模块划分

2.1 系统模块结构图

模块划分思路说明。

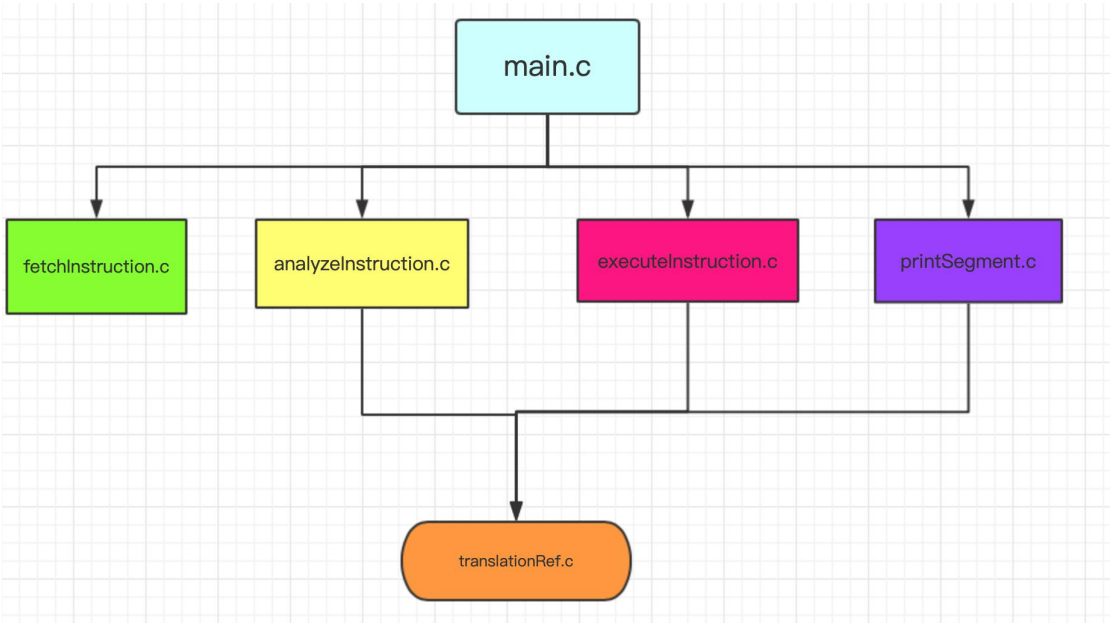

图 2-1 模块结构图示例

模块文件	模块说明	模块包含的函数名
main.c	**主函数模块，完成线程的建立，汇合，调用获取指**	int main(int argn, char *argv[])

	令，分析指令，执行指令，打印地址等函数	static void *doit(void *threadID)
fetchInstruction.c fetchInstruction.h	获取指令模块，完成指令的获取	int fetchInstruction(int tid)
analyzeInstruction.c analyzeInstruction.h	分析指令模块，完成指令的分析	void analyzeInstruction(int sum) int Instruct(char *Code1,char *Code2,char *Code3) int returnNumber1(char *Code1,char *Code2,char *Code3,char *Code4) int returnNumber2(char *Code1,char *Code2,char *Code3,char *Code4) int serialNumber(char *Code1,char *Code2,char *Code3)
executeInstruction.c executeInstruction.h	执行指令模块，完成指令的执行	void executeInstruction(vm_state_t *state,int tid)
printSegment.c printSegment.h	打印地址模块，完成代码段和数据段的打印	void printSegment(int sum)
translationRef.c translationRef.h	进制转换模块，完成十进制对二进制的转换	int translationRef(char *Code)

2.2 各模块函数说明

序号	函数原型	功能	参数	返回值
1	int main(int argn, char *argv[])	在 main.c 中被调用 完成线程的创建和汇合，以及调用打印函数	无	0
2	static void *doit(void *threadID)	在 main.c 模块中被调用 完成 2 个线程的运行	void *threadID 传入参数为线程 ID	无
3	int fetchInstruction(int tid)	在 fetchInstruction.c 模块中被调用 从文件中获取指令，将指令读入 CODE 和 codeSegment 数组中，并将指令前 8 位，第 9-12 位，第	int tid 传入线程 ID	返回指令行数 sum

		13-16 位，后 16 位，前 16 位分别读入 Code1，Code2，Code3，Code4，CODESET 数组中		
4	void analyzeInstruction(int sum)	在 analyzeInstruction.c 模块中被调用 分析指令，并将分析的结果保存在结构体 sample_code[100]中	int sum 参数为指令行数	无
5	int Instruct(char *Code1,char *Code2,char *Code3)	在 analyzeInstruction.c 模块中被调用 根据指令返回对应的操作，比如加减乘除，跳转等,对应结构体 inst_t 中的 code	char *Code1,char *Code2,char *Code3 参数分别为指令前 8 位，第 9-12 位，第 13-16 位	返回值为对应操作的数值
6	int returnNumber1(char *Code1,char *Code2,char *Code3,char *Code4)	在 analyzeInstruction.c 模块中被调用 根据指令返回执行指令中的第一个参数，对应结构体 inst_t 中的 p1 参数	Char *Code1,char *Code2,char *Code3,char *Code4 参数分别为指令前 8 位，第 9-12 位，第 13-16 位，后 16 位	返回值为执行指令要操作的第一个参数
7	int returnNumber2(char *Code1,char *Code2,char *Code3,char *Code4)	在 analyzeInstruction.c 模块中被调用 根据指令返回执行指令中的第二个参数，对应结构体 inst_t 中的 p2 参数	Char *Code1,char *Code2,char *Code3,char *Code4 参数分别为指令前 8 位，第 9-12 位，第 13-16 位，后 16 位	返回值为执行指令要操作的第二个参数
8	int serialNumber(char *Code1,char *Code2,char *Code3)	在 analyzeInstruction.c 模块中被调用 根据指令返回执行指令中的第三个参数，对应结构体 inst_t 中的 p3 参数，这个参数用来保存指令保存数据对应的寄存器标号	char *Code1,char *Code2,char *Code3 参数分别为指令前 8 位，第 9-12 位，第 13-16 位	返回值为执行指令保存数据的寄存器标号，或者 0
9	void executeInstruction(vm_state_t *state,int tid)	在 executeInstruction.c 模块中被调用 完成指令的执行，以及寄存器状态的输出	vm_state_t *state,int tid 参数为结构体 vm_state_t，以及线程 ID	无

10	**void printSegment(int sum)**	**在 printSegment.c 中被调用 完成代码段和数据段的打印**	**int sum 参数为指令条数**	**无**
11	**int translationRef(char *Code)**	**在 translationRef.c 中被调用 完成十进制转化成二进制**	**char *Code 参数为一段数字字符**	**返回值为对应的二进制数字大小**

2.3 函数调用图示及说明

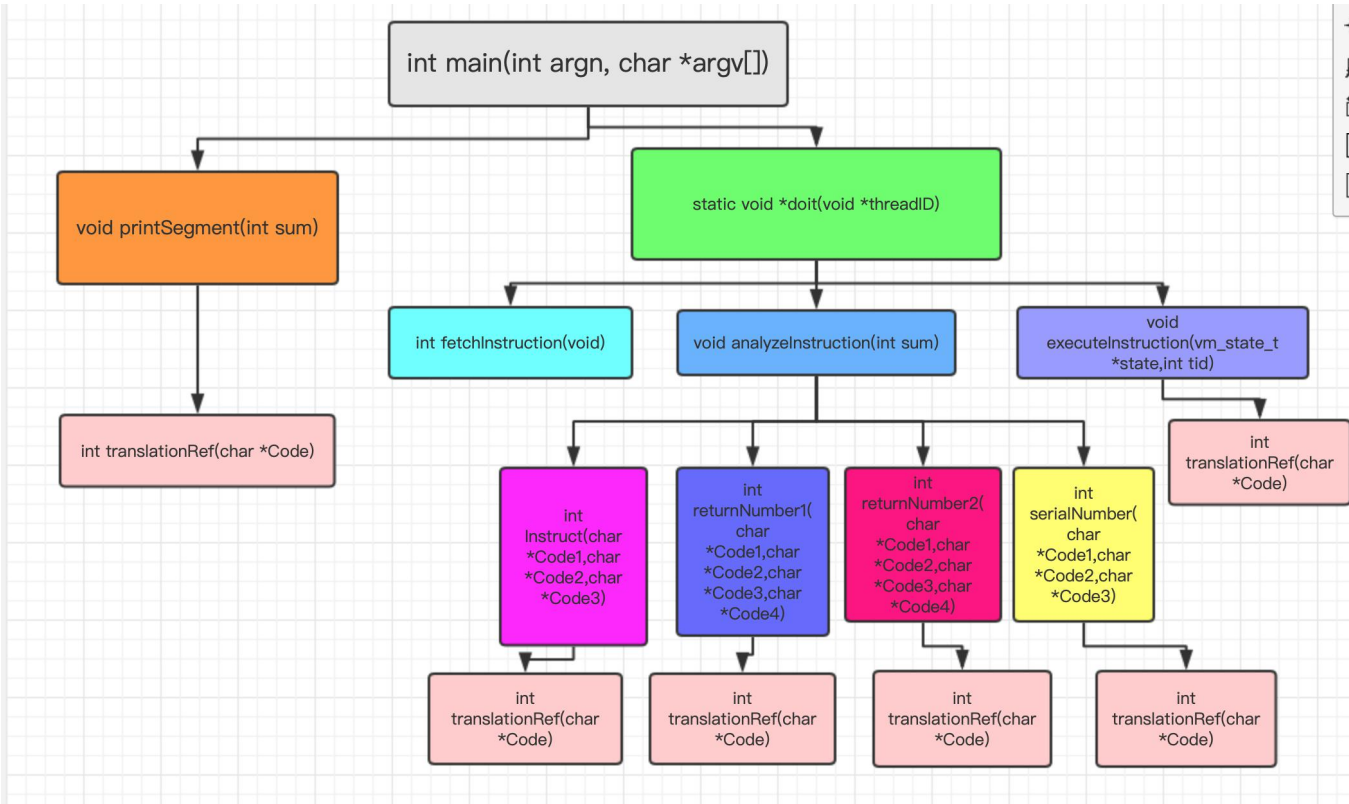

图 2-2 函数调用关系图示例