# 数据结构与算法分析技术栈

自己本身不是计算机科班出身的，但是以后一方面是为了自己快速成长的需要，另一方面是为了与科班出身的同学竞争，所以需要学习他们本科的课程。而《数据结构与算法分析》又是其中非常重要的理论基础课程，接下来就努力吧^- ^

## 计算机系统概论

### 计算机系统简介

计算机系统是由硬件和软件组成的，而软件又分为系统软件和应用软件。

多级层次结构的计算机系统：

1. 虚拟机器M4（高级语言机器）
2. 虚拟机器M3（汇编语言机器）
3. 虚拟机器M2（操作系统机器）
4. 传统机器M1（机器语言机器）
5. 微指令机器M0（微指令系统）

本书着重讲授传统机器M1和微指令机器M0，M0可以看作是M1的分解。

软硬件的划分并不是一成不变的，目前 操作系统已实现部分固化，这意味着软件永恒的存在于只读存储器当中。

而目前发挥怎趋势是往M4以上发展。

计算机的结构问题：包含有那些指令系统，如乘法指令，除法指令；

计算机的组成问题：具体指指令如何实现，即如何取指令，分析指令，去操作数，运算，送结果等；

本书主要研究计算机的组成原理

### 计算机的基本组成

***冯·诺依曼机的特点：***

1. 计算机由运算器，存储器，控制器，输入设备和输出设备五大件组成；
2. 指令和数据以同等的几位存储在存储器，并可以按址寻访；
3. 指令和数据都是二进制数；
4. 指令由操作码和地址码组成；
5. 机器以运算器为中心；

***现代计算机的特点：***

1. 机器以运算器为中心，计算的中间结果暂存于运算器中；
2. 运算器和控制器在大规模集成电路制作工艺出现后，并在一起，即所谓的CPU（Central Processing Unit），运算器中的核心部件包括算术逻辑单元（Arithmetic Logic Unit，ALU）和累加器（ACC）。控制器中的核心部件是控制单元（Control Unit）
3. 输入和输出设备简称为I/O设备或者外部设备
4. 现代计算机可以认为是有CPU，I/O设备和主存储器（Main Memory，MM）组成，CUP和主存储器又可以并成为主机；

操作码和地址码加起来一般有16位，前面6位是操作码，后面10位是地址码

1. 取数：000001，放到ALU中的累加器ACC中
2. 存数：000010
3. 加：000011
4. 乘：000100
5. 打印：000101
6. 停机：000110

***主存储器***

1. MAR

Memory Address Register，存储器地址寄存器，用来存储与访问的存储单元的地址，其位数对应这存储单元的个数，如MAR有10位，则对应210=1024个存储单元，1K

1. MDR

Memory Data Register，存储器数据寄存器，存放从存储体某单元取出的代码或者准备往存储体存入的代码，其位数跟存储字长相等。我觉得类似与缓存的作用，用来暂存数据的。

1. 存储体M

随着硬件的发展，目前很多的MAR和MDR都集成在CPU当中

***运算器***

一般有三个寄存器 + ALU

1. ACC(Accumulator)，累加器

被加数及和，被减数和差，乘积高位，被除数和余数，所有的被运算数都从ACC中拿出

1. MQ(Multiplier-Quotient Register)，乘商寄存器

乘数及乘积低位，商

1. X，操作数寄存器

加数，减数，被乘数（来源于ACC）和除数

简要分析一下运算过程，设M地址号单元，【M】地址号单元中的内容，X表示X寄存器，【X】表示X寄存器中的内容；ACC表示累加器，【ACC】表示累加器中的内容；MQ表示乘商寄存器，【MQ】表示乘商寄存器中的内容。

加法操作：

【M】->X

【ACC】+【X】->ACC

减法操作：

【M】->X

【ACC】-【X】->ACC

乘法操作：

【M】->MQ

【ACC】->X

0->ACC

【X】\*【MQ】->ACC//MQ（//表示两个寄存器串接）

除法操作：

【M】->X

【ACC】/【X】-> MQ

商保留在ACC中

1. ALU

***控制器***

一条操作指令分为取指，分析和执行三个阶段

取指就是取出指令，分析就是分析操作内容是什么，执行就是根据操作数的地址和操作码完成相应的操作；

控制器由下列部件组成

1. 程序计数器（Program Counter，PC）

用于存放当前欲执行指令的地址，他与主存的MAR有一条直通的通道，且具有自动加1的功能，即自动形成下一指令的地址；

1. 指令寄存器（Instruction Register，IR）

用于存放当前的指令，其指令内容主要来自于主存的MDR。不同于程序计数器单纯的读命令操作，IR主要根据指令的具体内容而行动。

1. 控制单元（CU）

其操作流程主要是，启动机器后：

1. 取指过程
   1. 【PC】-> MAR
   2. 【存储体M】->MDR
   3. 【MDR】->IR
2. 分析过程

OP(IR) -> CU

1. 执行过程
2. Ad(IR) -> MAR
3. 【存储体M】->MDR
4. 【MDR】->ACC

完成后PC自动加1，准备执行下一条指令

### 计算机硬件的主要技术指标

机器字长

指的是CPU一次能处理数据的位数，通常与CPU的寄存器位数有关

1. 机器字长越长，精度越高；
2. 会影响成本和计算速度；

存储容量

存储容量=存储单元个数 \* 存储字长

（MAR） （MDR）

MAR的位数反映了存储单元的个数，比如10位的话代表有1024个存储单元；

每个存储单元中的容量有MDR反映，MDR中有32位，意味着每个存储单元中有有32位的容量，那么总的存储容量=210\*32=215位（bit）=212Byte(B，或者叫字节)=22KB=4KB

1M=220bit

运算速度

1. MIPS

单位时间内执行指令的的平均条数来衡量，并用MIPS（Million Instruction Per Second，百万条指令每秒）

1. CPI

执行一条指令所需要的时钟周期，Cycle Per Instruction

## 系统总线

### Constuctor对象

创建一个对象的方法

1. new调用的是public的构造方法，可以调用无参或者是有参的构造方法；
2. 利用.newInstance()方法创建一个实例，底层是调用有参的构造方法；
3. 通过构造方法对象创建一个类的实例Constructor

应用实例

1. 获得单个公共方法

Constructor constructor=类的class对象.getConstructor(参数的Class对象)；

类名 clazz =（类名）constructor. newInstance(参数)；

1. 获得单个方法

上述方法只能获得一个公共构造方法，若要获得私有的构造方法，需要使用增强方法getDeclareConstructor()方法，只要声明了就能找到：

Constructor constructor=类的class对象. getDeclareConstructor (参数的Class对象)；

constructor.setAccessible(true); //暴力访问；

类名 clazz =（类名）constructor. newInstance(参数)；

1. 获得所有公共方法

Constructor[] constructors=类的class对象.getConstructors()；

for(Constructor constructor: constructors){

System.out.println(constructor);

}

1. 获得所有方法

Constructor[] constructors=类的class对象.getDeclareConstructors()；

for(Constructor constructor: constructors){

constructor.setAccessible(true);

System.out.println(constructor);

}

### Field对象

1. 获得所有公共成员变量：

Class clazz = Class.forName(“com.lvhongbin.bean.User”);

Field[] fields=clazz.getFileds();

For(Filed field: fields){

System.out.println(field);

}

1. 获得单个成员变量（不管是私有的或者是共有的）

Class clazz = Class.forName(“com.lvhongbin.bean.User”);

类名 obj =（类名）clazz. newInstance(参数)；

Field field=clazz.getDeclareFiled(“成员变量名称”);

field.setAccessible(true);

System.out.println(field);

//给field赋值

field.set(obj, 值)；

### Method对象

1. 获得单个成员变量（不管是私有的或者是共有的）

Class clazz = Class.forName(“com.lvhongbin.bean.User”);

类名 obj =（类名）clazz. newInstance(参数)；

Method method =clazz.getDeclareMethod (“成员方法名称”，参数的类对象);

method.setAccessible(true);

System.out.println(method);

// 执行method方法

field.invoke (obj, 参数值)；

1. 获得所有公共成员变量：

Class clazz = Class.forName(“com.lvhongbin.bean.User”);

Field[] fields=clazz.getFileds();

For(Filed field: fields){

System.out.println(field);

}