第一节 计算机系统简介

1.现代计算机的多态性

物联网系统：把感应器嵌入和装备到电网、铁路、桥梁、隧道、公路、建筑、供水系统、大坝、油气管道等各种物体中，并且被普遍连接，形成所谓“物联网”，然后将“物联网”与现有的互联网整合起来，实现人类社会与物理系统的整合，形成智慧地球。

2.计算机软硬件概念

（1）计算机系统由**硬件**和**软件**两部分组成。软件又分为**系统软件**和**应用软件**。

3.计算机的层次结构

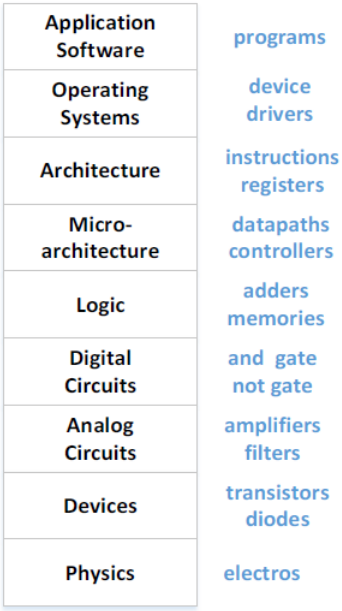


图 1 计算机物理结构

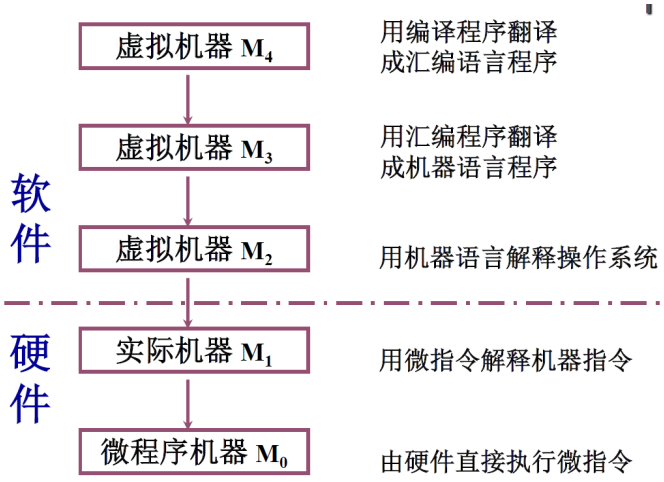


图 2 计算机的层次结构

计算机体系结构：程序员（机器语言）所见到的计算机系统的属性。概念性的结构和功

能特性。(指令系统、数据类型、寻址技术、I/O 机理)

计算机组成：实现计算机体系结构所体现的属性。(具体指令的实现)

第二节 计算机的基本组成

1.冯诺依曼计算机的特点

（1）计算机由五大部件组成：运算器、存储器、控制器、输入设备、输出设备。
（2）指令和数据以同等地位存于存储器，可按地址寻访；
（3）指令和数据用二进制表示；
（4）指令由操作码和地址码组成；
（5）存储程序；
（6）以运算器为中心。

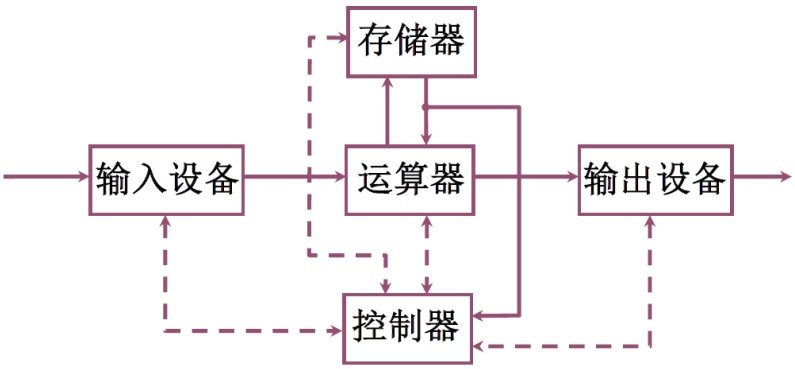


图 1 冯诺依曼结构（实线是数据传输，虚线是地址传输）

2.硬件系统的改进

问题 1：以运算器为中心，导致运算器成为系统的瓶颈。
问题 2：不具有层次化的特征。
改进 1：以存储器为中心的计算机硬件框图

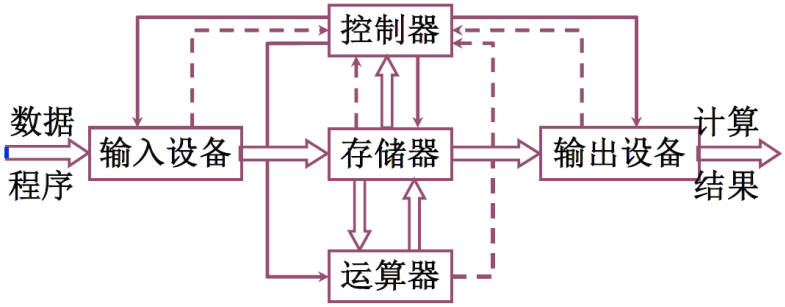


图 2 以存储器为中心的计算机硬件框图

改进 2：
运算器 ALU ---------CPU----------主机
控制器 CU ----|
存储器 -------主存----------|
---|辅存

输入设备-------I/O 设备

输出设备-------|

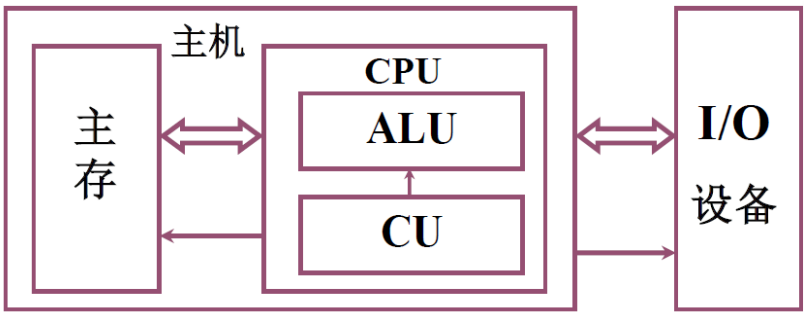


图 3 改进 2 的计算机硬件框图（现代计算机硬件框图）

系统复杂性管理方法 2（3Y）

1.层次化（Hierachy）：将被设计的系统划分为多个模块或者子模块；

2.模块化（Modularity）：有明确定义（well-defined）的功能和接口；

3.规则性（regularity）：模块更容易被重用。

3.计算机的工作步骤

3.1 上机前的准备

①建立数学模型

②确定计算方法

③编织解题程序：程序-运行的**全部步骤**；指令-每一个**步骤**。

编程举例：

计算 ax^2+bx+c

假设系统包含加法指令、乘法指令、取数指令、存数指令、打印指令和停机指令。

假设 a，b，c 已经保存起来了。计算步骤如下：

①首先取 x 至运算器中的累加器（寄存器）中；（**取数指令**）

②乘 x，在运算器中，计算的 x 的平方，保存在累加器 ACC 中；（**乘法指令**）

③乘 a，在运算器中，在累加器 ACC 中保存 ax 的平方；（**乘法指令**）

④存数指令，存 ax 的平方在存储器中；（**存数指令**）

⑤取 b 至运算器中；（**取数指令**）

⑥在运算器中，对 b 乘 x，acc 中保存 bx；（**乘法指令**）

⑦把 $bx+ax^2$，在累加器中；（加法指令）

⑧加 c，在累加器中；（**加法指令**）

一共执行 **8 条**指令，指令较多，运算速度慢存储量较大。但是如果做一个优化，将该式处理为(ax+b)x+c。计算步骤如下：

①取 x 至运算器中；（**取数指令**）

②a 乘 x，在运算器中；（**乘法指令**）

③把 ax+b，在运算器中；（**加法指令**）

④乘 x，在运算器中；（**乘法指令**）

⑤加 c，在运算器中。（**加法指令**）

一共执行 5 条指令，相对来说指令少，运算速度快，存储量也小。

另外要说明的是，累加器是运算器的一个部分。

指令格式举例

操作码	地址码

取数（把内存单元 α 中的数据存储在 ACC 中） [α]→ACC

假设我们的指令有 16 位，操作码为 6 位，地址码为 10 位。则可以如下格式：

000001 0000001000

存数（把 ACC 中的数据保存在内存单元 β 当中） [ACC]→β

加 γ [ACC]+[γ]→ACC

乘 σ [ACC]×[σ]→ACC

打印 θ [θ]→打印机

停机

假设内存和 ACC 均为 8 位长度，那么乘法会使数据边长，必然会导致数据溢出。应予以注意。

计算 ax^2+bx+c 程序清单如下图所示。

指令和数据存于主存单元的地址	指令		注释
	操作码	地址码	
0	000001	0000001000	取数x至ACC
1	000100	0000001001	乘a得ax,存于ACC中
2	000011	0000001010	加b得$ax+b$,存于ACC中
3	000100	0000001000	乘x得（$ax+b$)x,存于ACC中
4	000011	0000001011	加c得ax^2+bx+c,存于ACC
5	000010	0000001100	将ax^2+bx+c,存于主存单元
6	000101	0000001100	打印
7	000110		停机
8	x		原始数据x
9	a		原始数据a
10	b		原始数据b
11	c		原始数据c
12			存放结果

图 4 计算程序清单（x/a/b/c/存储结果均为二进制数存储）

冯诺依曼机中，指令和数据都是**同等地位**以**二进制数据**保存起来的。

3.2 计算机的解题过程

（1）存储器的基本组成

图 5 存储器的基本组成

存储器-存储单元-存储元件（0/1,），类似于大楼-房间-床位（无人/有人）

存储单元：用来存放一串二进制代码；

存储字：存储单元中二进制代码的组合；

存储字长：存储单元中二进制代码的位数。

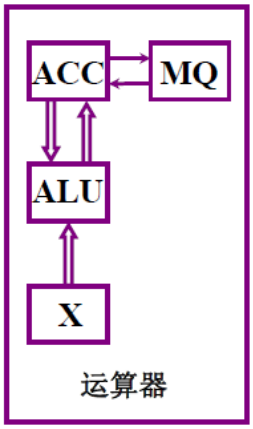
每个存储单元赋予一个地址号，实行**按地址寻访**。

MAR（Memory Address Register）：存储器地址寄存器。反映存储单元的个数。

MDR（Memory Data Register）：存储器数据计算器。反映存储的字长。

设 MAR=4 位，MDR=8 位，则存储单元是 16 个（2^4），存储字长为 8。

（2）运算器的基本组成及操作过程



	ACC	MQ	X
加法	被加数 和		加数
减法	被减数 差		减数
乘法	乘积高位	乘数 乘积低位	被乘数
除法	被除数 余数	商	除数

图 6 运算器各部分组成及功能

运算器核心是算术逻辑单元（ALU），它是一个组合逻辑电路，也就是说当输入撤销之后，输出也会消失，因此后面需要存在一个寄存器，保存参与运算的数据，这两个寄存器为 ACC（累加器）和 X（数据寄存器）。如果做乘法运算，则多余出来的位数存放在 MQ 中。

注意：此结构仅限于本课程的一种类型。

对于加法来说，加数存放在 X 中，被加数存放在 ACC 中，和存放在 ACC 中；对于剪发说，被减数存在 ACC 中，减数存在 X 中，差在 ACC 中；对于乘法来说，乘法可以用加法和移位的方式来计算，被乘数存放在 X 中，乘数放在 MQ 中，乘积高位在 ACC 中，乘积地位在 MQ 中；对于除法来说，除法是用减法和移位来实现的，被除数在 ACC，除数是 X 中，商在 MQ 中，余数在 ACC 中。

①加法操作过程

指令：

加	M

初态：ACC 被加数
第一条指令：[M]→X
执行加法指令：[ACC]+[M]→ACC
如图 7 所示。

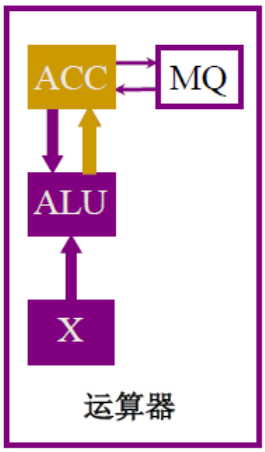


图 7 加法（减法）操作过程图

②减法操作过程
指令：

减	M

初态： ACC 中存在被减数
第一条指令：[M]→X
执行减法指令：[ACC]-[X]→ACC
③乘法操作过程
指令

乘	M

初态 ACC 被乘数
第一条指令（乘数）：[M]→MQ
第二条指令（被乘数）：[ACC]→X
第三条指令（累加之前先将 ACC 置零）：0→ACC
乘法指令（累加+移位）：[X]×[MQ]→ACC//MQ（高位//地位）
只有 ACC 送到 X 以后才可以做乘法，才能把 0 送到 ACC 中。但是第一条指令和第二条指令顺序可以互换。这个先后顺序是**控制器**来控制的。

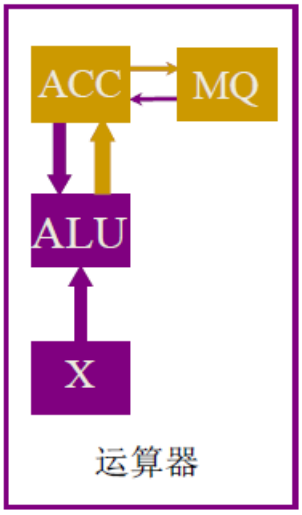


图 8 乘法（除法）操作过程

④除法操作过程

指令

除	M

初态 ACC 被除数

第一步指令：[M]→X

第二步指令：[ACC]÷[X]→MQ

（余数在 ACC 中）

MQ 又叫乘商寄存器。注意：[X]指的是 X 对应地址的内容。

（3）控制器的基本组成

控制器的功能：解释指令（从取指令、分析指令、取操作数……一直到完成指令）；保证指令按序执行。

完成一条指令包括三步：①取指令 PC（程序寄存器）；②分析指令 IR（指令寄存器）；③执行指令 CU。

PC 中存放当前要执行指令的地址，指令地址是相邻的，因此 PC 具有计数功能，即把 PC+1 赋给 PC。

IR 存放当前欲执行的指令，控制单元可以把 IR 中的操作码取出来分析。操作控制就是 CU 完成。

具体框图如下：

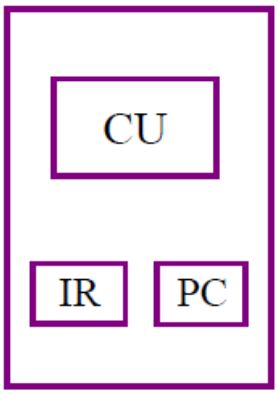


图 9 控制器的组成

（4）主机完成一条指令的过程

①以取数指令为例

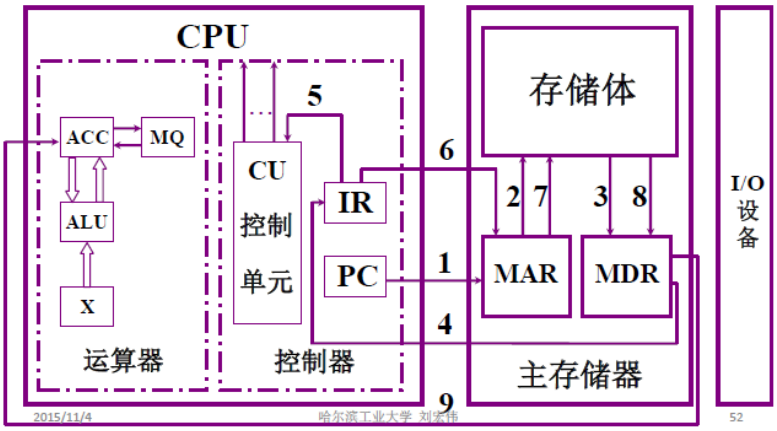


图 10 取数指令的过程

第一步：取指令。要执行指令的地址是 PC 中，指令存在存储体中。首先 PC 把地址给 MAR，然后给存储体，在控制器控制下，存储体中存储的指令取出来送入 MDR 中，然后当前指令的地址存在 IR 中，取指令完成。

第二步：分析指令。把 IR 中的指令的操作码部分送入 CU，译码之后进行下面的操作。

第三步：执行指令。在控制器的控制之下，要把 IR 指令当中的地址部分送给存储器（MAR），以便取出数据。然后 MAR 把地址送给存储体重，存储体把数送入 MDR，然后送入 ACC。

②以存数指令为例

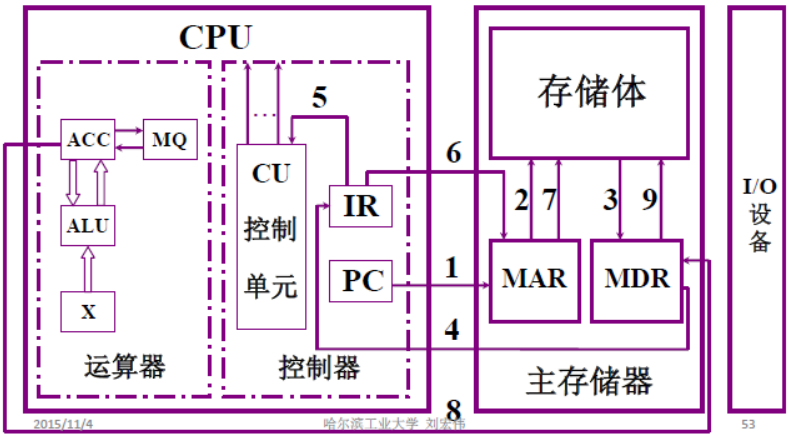


图 11 存数指令的过程

第一步：取指令。首先 PC 把当前指令的地址发送给 MAR，然后 MAR 又发送到存储体，在控制器控制下，存储体中存储的指令取出来送入 MDR，MDR 将该指令存入 IR 之中。

第二步：分析指令。把 IR 中的指令的操作码送给 CU，由 CU 对指令进行分析，CU 发出响应控制信号。

第三步：执行指令。首先在 CU 控制下，把 IR 中的指令的地址码送入 MAR 中，MAR 送入存储体，告诉存储体现在有一个数据要存进来和地址是多少。然后在 CU 控制下，ACC 中的数据被送入到 MDR 中，然后再送入到存储体中，在存储体中的地址按照 MAR 的要求来确定。

（5）ax^2+bx+c 程序的运行过程

①将程序通过输入设备送至计算机。

②程序首地址送入 PC 中。

③启动程序运行：
取指令 PC→MAR→M→MDR→IR，（PC）+1→PC
分析指令：OP(IR)→CU（OP 是操作码）
执行指令：加：Ad(IR)→MAR→M→MDR→ACC。（Ad 代表地址码的意思）
④打印结果
⑤停机

第三节 计算机硬件的主要技术指标

1.机器字长

CPU 一次能处理数据的位数 与 CPU 中的 **寄存器位数** 有关。

2.运算速度

运算速度
- 主频
- 核数，每个核支持的线程数
- 吉普森法 $T_{\mathrm{M}} = \sum_{i=1}^{n} f_i t_i$
- **CPI** 执行一条指令所需时钟周期数
- **MIPS** 每秒执行百万条指令
- **FLOPS** 每秒浮点运算次数

吉普森法：第一种，静态水平：直接拿程序清单里面来计算，不用管实际运算。第二种：程序执行起来，计算一下实际执行中出现的频率，动态水平（更准确）。

3.存储容量

3.存储容量 存放二进制信息的总位数

主存容量
- 存储单元个数×存储字长
  如 MAR MDR 容量
  10 8 1 K×8位
  16 32 64 K×32位
- 字节数
  如 2^{13} b=1 KB（$1B = 2^3 b$）
  2^{21}b=256 KB

（$1K = 2^{10}$）

辅存容量 字节数 80 GB（$1GB = 2^{30}B$）

2015/11/4 哈尔滨工业大学 刘宏伟 57