

内 部

“748”会议资料选编

总体运控部分

“七 四 八”会 议 秘 书 处

一九七五年四月

出 版 说 明

在批林批孔运动普及、深入、持久发展的大好革命形势下经国务院、中央军委批准，中国科学院、四机部、国防科委于一九七四年九月二日至十八日，在北京召开了“电子计算机技术经验交流会”（代号为“748会议”）。

根据到会同志的要求和会议领导小组决定，现将十个专业组（总体运控、计算数学、程序系统、内存贮器、结构工艺、外部设备、电源、元器件、模拟计算机、工业控制及数据处理）的技术资料，分别整理选编成册，供参考。如有错误和不妥之处，希批评指正。

“748”会议秘书处

目 录

1. 150 电子计算机研制	1
2. 大型计算机系统设计方案	17
3. 151-3(4) 型数字计算机的总体设计	39
4. 013 机体系方案	93
5. 一台实时控制机的设计	102
6. 一个使用高速缓存进行指令调度的方案	107
7. 超快速数码缓冲存贮器自动调度方案	115
8. 微程序设计及其应用	131
9. 用软硬结合的方法克服流水线障碍之一——运算优化	143
10. 用软硬结合的方法克服流水线障碍之二——转移的处理	149
11. 用软硬结合的方法克服流水线障碍之三——函数的实现	155
12. 多位-乘与多位-除算法	167
13. 711 机的加乘除	194
14. 一个高速简易除法方案	216
15. 迭代除法	229
16. 一个高速加法器的逻辑方案	245
17. 一个加法、逻辑运算的监视线路	250
18. 计算机辅助设计的初步实践	263
19. 先行控制系统模拟	272
20. 专用机超快速小存使用率的模拟统计	293
21. 自动设计计算机中的自动选择布线问题	299
22. 655 机技术总结	313

23. 从我厂生产的计算机谈谈对计算机可靠性的认识	321
24. 150 机可靠性的一些问题	329
25. 提高 JD-2 组件机可靠性的几个措施	347
26. 电子计算机可靠性估算的一些问题	351
27. 中规模计算机的初次尝试	371
28. 一台 ECL 模型机	377
29. 高速 ECL 组件的信号传送与安装	387
30. S-5 机信号传输问题的一些考虑	410
31. 几种电流开关电路	422

150 电子计算机研制

在毛主席的无产阶级革命路线指引下,北京大学、北京有线电厂和燃化部石油物理勘探局等单位共同合作,经过三年多的努力,设计试制成功了150电子计算机。150机在出厂前经过三千多小时的试算和考验,证明这台计算机性能比较稳定,主机、外部设备(除磁盘外)和程序系统都已达到了设计要求。在搬迁到使用单位,经过了短期的安装恢复后,再次进行考核,机器稳定性又有所提高。现在已经交付使用单位使用,为加速我国石油地质勘探提供了新的手段。

150电子计算机是一台百万次的高速集成电路电子计算机,这台计算机的试制成功,是贯彻执行毛主席关于教育革命的指示,坚持生产、使用、科研三结合的结果,也是广泛开展社会主义大协作,大搞群众运动的丰硕成果。

在拟定这台电子计算机的设计方案以及进行设计调试的过程中,我们多次向华东计算所学习了655机的总体设计思想和研制经验,他们给予我们很大的帮助,我们在这里向他们表示感谢。我们还要向北京计算所、十五所和其它很多兄弟单位表示感谢,我们在设计试制150机的过程中,得到他们在物质和精神上的很大支持。

下面先对150电子计算机作一概括叙述,然后将设计、调试过程中所遇到的主要问题和机器使用情况作一介绍。

一、概 况

150机在设计时,燃化部提出了高效率处理石油地震勘探数据的要求,这些要求,基本上都是一台通用电子计算机所应具有的性能,所以我们最后设计的是一台能比较广泛地进行多方面科学计算和数据处理的通用电子计算机。

1. 基本技术性能

(1) 字 长

48位二进制数。数有全字长浮点数、半字长浮点数和半字长整数三种形式,都用补码系统。指令长度只有半字长一种。

150机增加半字长浮点数,主要是为了作数据处理用途时节省内存单元,运算速度并没有因此而增加多少。

(2) 运算速度

全字长浮点加法时间平均0.81微秒

全字长浮点乘法时间1.28微秒/次

解200阶矩阵时间17秒(用语言编写)

主脉冲频率3.1兆赫

(3) 内存贮器

由4个容量为3.2万的磁芯存贮器组成,总容量为13万字,4个体按模4编址。每个磁芯体读写周期为2微秒,采用 $0.8 \times 0.5 \times 0.3$ 毫米宽温锂锰铁涂氧磁芯,二度半方式。

(4) 外存贮器

磁盘存贮器, 用 CP-1 型磁盘存贮器 1 台, 磁盘直径 600 毫米, 18 片, 容量 160 万字以上。

磁带存贮器, 用 DL-1 型磁带机 8 台, 带宽 1 吋, 每盘带的容量 200 万字以上。

(5) 输入输出设备

第一台 150 机配有以下通用输入输出设备

5~8 RG-5 型光电输入机	4 台(2 架)
快启停光电输入机	1 台
CY-160-5 型宽行打印机	4 台
CY-4 型打印机	1 台
LZ-5 型数字式曲线仪	2 台
6813B 型快速凿孔机	1 台
BD055 型控制台打字机	2 台

(6) 交换器

加接外部设备的通道最多为 32 个, 每个通道上最多可接 2 台相同的但不可同时工作的外部设备, 全部通道总的信息流量可达 500 千赫字节以上。(按 8 位的“字节”计算)

(7) 变址寄存器

共 32 个, 可扩充到 64 个

(8) 中断系统

用来解决外部设备的管理, 多道程序的实现, 控制台和打字机的使用, 出错的处理, 以及供电断电时信息的保护等方面的问题。对于实时控制留有通路。

(9) 多道程序

最多可以同时执行 4 道算题程序。

(10) 电源容量

全机直流电源总功率不超过 15 瓩。

(11) 配有以下程序系统

管理程序

语言程序

符号程序

检查程序

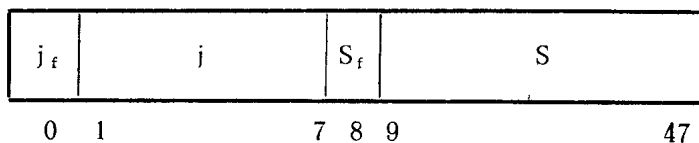
(12) 全机规模

机 柜	25 个
通用外部设备	9 种 24 台
插 件	3082 块
集成电路(TTL)	57331 块
厚膜电路	7611 块
三极管	9363 只
二极管	48592 只

2. 指令系统

(1) 数的表示

全字长浮点数, 48 位

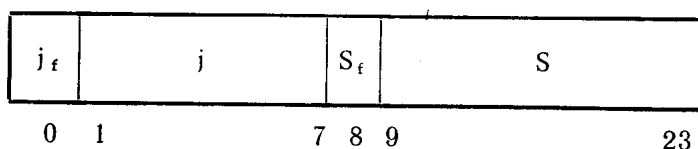


$$X = 2^j \cdot S$$

其中 j 是阶码部分, 阶符 $j_f = \begin{cases} 0 & \text{表示阶为正} \\ 1 & \text{表示阶为负} \end{cases}$

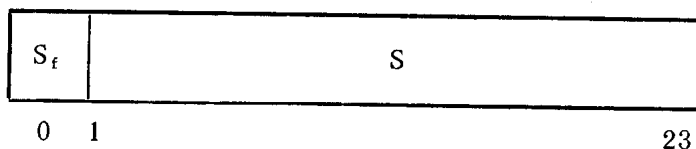
S 是尾数部分, 数符 $S_f = \begin{cases} 0 & \text{表示数为正} \\ 1 & \text{表示数为负} \end{cases}$

半字长浮点数, 24 位



符号意义同全字长浮点数。

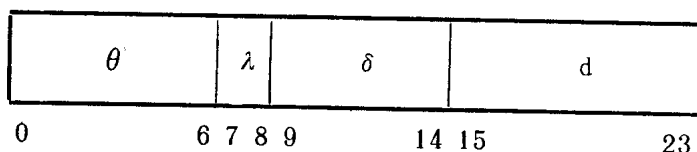
整数, 24 位



数符 $S_f = \begin{cases} 0 & \text{表示数为正} \\ 1 & \text{表示数为负} \end{cases}$

150 机中整数是不成系统的, 没有相应的整数运算指令, 只提供了“整数→浮点数”、“分离整数”两条指令。若进行整数运算时可用“整数→浮点数”指令把整数化成浮点数, 然后用浮点运算, 最后将结果取整而得。

(2) 指令形式



其中: θ —— 操作码

δ —— 变址器地址

λ —— 变址特征位

d —— 形式地址

变址特征位 λ 在算术和逻辑运算以及存取数时的意义如下 (对于其它类型的指令有其它意义, 或不起作用):

$\lambda = 0$ 时 直接地址 $D = \delta d + Xd$

$\lambda = 1$ 时 变址 $D = (\delta) + d + Xd, (\delta) + d \rightarrow \delta$

$\lambda=2$ 时 变址 $D=(\delta)+d+Xd, (\delta)\rightarrow\delta$

$\lambda=3$ 时 变址 $D=(\delta)+d+Xd, (\delta)+\Delta\rightarrow\delta$

其中: $\Delta=\begin{cases} 2, & \text{当 } \theta \text{ 为全字长操作时} \\ 1, & \text{当 } \theta \text{ 为半字长操作时} \end{cases}$

Xd 为下界地址, 对于某一道题目, Xd 就是它所占内存的首条的绝对地址。直接地址只有 15 位, 所以每道程序中能使用直接地址的只限于前 32768 个单元。

(3) 指令概况

操作码共有 94 种, 分类统计如下:

控制组 11 种	逻辑组 8 种
变址组 12 种	存数组 6 种
转移组 15 种	取数组 6 种
移位组 13 种	半字长算术组 8 种
算术组 8 种	半字长逻辑组 7 种

3. 框 图

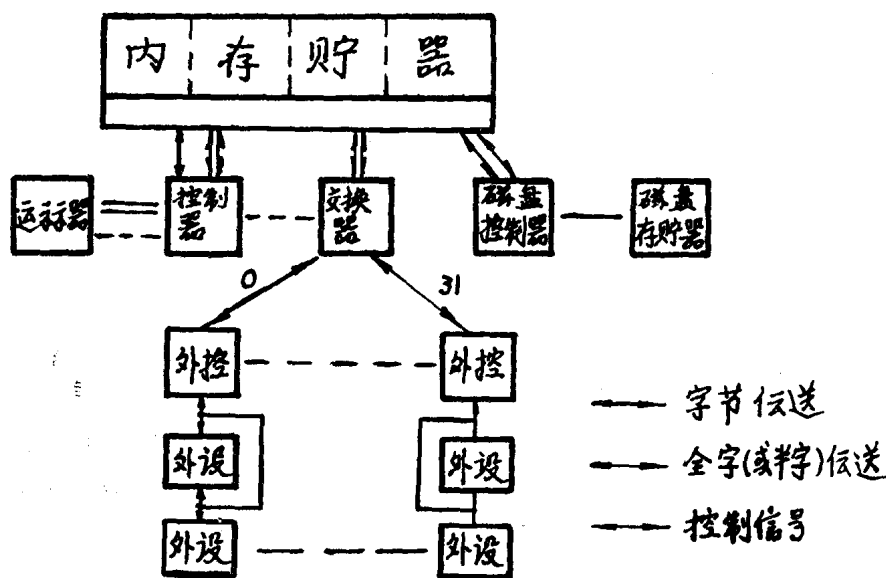


图 1 150 机框图

150 机全机由运算器、中央控制器、内存贮器、磁盘控制器、交换器及各种外部设备组成。全机框图见图 1。

二、设计过程中所考虑的主要问题

1. 内存贮器模 4 工作及访内分配器

150 机要求运算速度高, 存贮容量大, 在进行总体考虑时, 首先遇到的就是大容量内存贮器速度慢, 满足不了每秒百万次运算速度的需要这样一个矛盾。当时两度半方式的磁芯存贮器, 容量为 3.2 万字时, 读写周期只能做到 2 微秒, 所以在方案中选取了模 4 方式工作。这样在较好的情况下, 基本上能满足运算速度的要求。这可以通过下面的粗略估计看出:

访内分配器的节拍安排使磁芯体在 7 个机器节拍内完成一次读写访问，4 个磁芯体在充分交叉工作时，每 7 拍可以读出 4 个全字长数，即在理想的情况下平均每 1.75 拍可以访问内存一次，读出一个全字长数。

中央控制器每加工一条指令按平均需要访问内存 1.3 次计算，其中取指令占 0.5 次（每次可取 2 条指令），取数或写数占 0.8 次（假设有 80% 的指令要取数或写数），访问内存 1.3 次需要 $1.3 \times 1.75 = 2.275$ 拍。

运算器平均每 2.985 拍完成一次运算，这和上面所举内存提供指令和数的速度相比还有 0.71 拍的余量。这个余量就准备用来对付外部设备的访问、内存地址冲突造成的等待损失以及由中央控制器吸收的一部分控制型指令的访问等。

以上计算总的来说是个比较乐观的估计。每次访问内存平均 1.75 拍是在 4 个体充分平行工作时的最好情况才能达到的。实际上，由于访问内存的地址不均匀分布于 4 个体，特别是磁盘交换器后进先出区转接栈，大量访问内存引起地址冲突，造成 4 个磁芯体在工作中处于忙闲不均的状态，每个体不可能一直不停顿地工作。所以 4 个体的实际效能，要小于一个体的 4 倍。一般计算在 2~4 倍之间。150 机在运行后的初步观察，磁芯存储器驱动电流的平均值只有最大值的 70% 左右，由此推知，磁芯体只有 70% 左右时间在工作。如按此估计，4 个体的效能只相当 2.8 个体左右，访问内存 1.3 次就要超过 3 拍。所以用 2 微秒的磁芯存储器按模 4 交叉工作供给每秒一百万次的运算，特别是对于 150 机这样的单累加器计算机来说，是略感不足的。但限于当时存储器技术水平和不想使机器太庞大，所以内存存储器仍采用模 4 工作。在访内分配器的设计中，保留了一定的灵活性，可以随着磁芯存储器速度的提高而减少一个磁芯存取周期的节拍数，为今后改善速度匹配问题提供方便。

下面简单讲一下访内分配器：

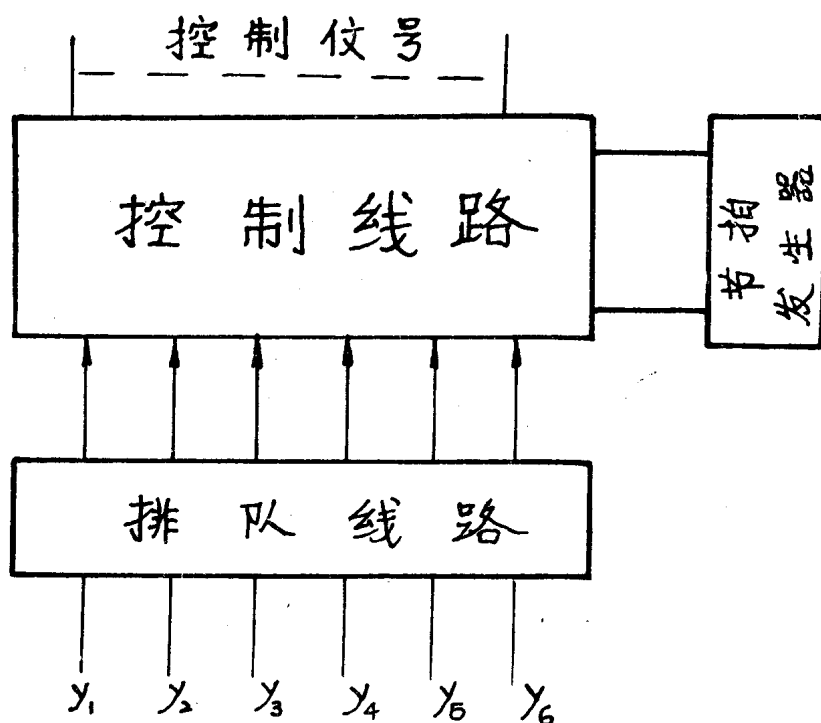


图 2 访内分配器原理示意图

150 机的内存贮器由访内分配器进行统一控制管理, 访内分配器原理示意图见图 2。全机共有 6 个访问内存的请求源, 按排队优先次序, 它们是:

- (1) 磁盘存贮器 Y_{11} ;
- (2) 交换器 Y_2 ;
- (3) 先行写数栈 Y_3 ;
- (4) 先行读数栈 Y_4 ;
- (5) 后进先出区转接栈 Y_5 ;
- (6) 先行指令栈 Y_6 。

150 机的地址码为 18 位, 其中最后一位 D_{18} 表示单元是左半字还是右半字, 最后第二、三两位反映单元在哪一个体中, 当 6 个请求源之一发出请求后, 究竟哪个体的访内分配器响应, 由这个请求源的地址寄存器的 D_{16} , D_{17} 决定。当同时有几个请求源向同一个体发出请求时, 由排队线路保证只响应优先级别最高的一个请求源。在图 3 中, 下面的一排门电路就是

D_1		D_{16}	D_{17}	D_{18}
-------	-------	----------	----------	----------

其中: 当 $D_{18}=0$, 左半字 $D_{18}=1$, 右半字

$D_{16}D_{17}=00$, 单元在 0 体中

01, 单元在 1 体中

10, 单元在 2 体中

11, 单元在 3 体中

排队线路的原理示意。排上队的请求就被接收到 6 个执行触发器 CY_1-CY_6 之一中去, 这个执行触发器在整个访内工作周期中保持不变。而各请求源的请求触发器在被响应后可以立即清除, 以便再次发出新的请求。

访内分配器同中央控制器的主脉冲同步, 在响应请求后, 用七拍时间完成一次访问内存的工作。这七拍电位 W_1^* , $-W_1^*$ 由计数器 JFN_1-JFN_3 产生, 另有 W_0^* 拍为空闲状态, 访内分配器在没有请求的空闲时间里, 都停留在 W_0^* 拍上。

由执行触发器和节拍控制电位经控制线路, 组合成一系列的控制信号, 来完成下列三项任务。

(1) 向内存某体写入时, 开有关的门将地址码与数码从请求源送入这个磁芯体的地址寄存器和数码寄存器, 并发启动磁芯体写入的信号;

(2) 从内存某体读出时开有关的门, 将地址码由请求源送入这个磁芯体的地址寄存器, 并启动磁芯体, 等磁芯体读出后, 开有关的门把数码送给请求源;

(3) 完成其它操作。例如: 清除请求触发器及给请求源发回答信号等。

2. 先行控制及所带来的问题

内存贮器按模 4 方式工作以后, 如果不采用先行控制技术, 运算器和控制器还象较小的计算机那样, 等到要用什么指令或什么数时, 才临时到内存中去取, 那么每次还不得不等待 2 微秒才能得到所需要的指令或数。这样, 每次还是只能一个磁芯存贮器在工作, 其他三个空等着, 结果丝毫发挥不出 4 个体平行工作的潜力。因此, 150 机的中央控制器采用了先行控制技术。所谓先行控制就是使中央控制器的工作比运算器“先行”一些, 提前按顺序从内存取出指令, 并对这些指令预先进行加工, 遇到要由运算器执行的指令时则把操作码及操作数

——如果运算器需要的话，提前准备好。运算器要存数到内存时，也先把数送入中央控制器的先行写数寄存器中，然后再由中央控制器把数写入内存。这样运算器只要一拍（320 微秒）的时间就可取得“指令”（操作码）和存取操作数，且取操作码和取操作数可以在一拍中重叠进行，使得运算器不受“慢速”内存的牵制，而实现百万次的运算速度。以上工作过程可以图示如下：

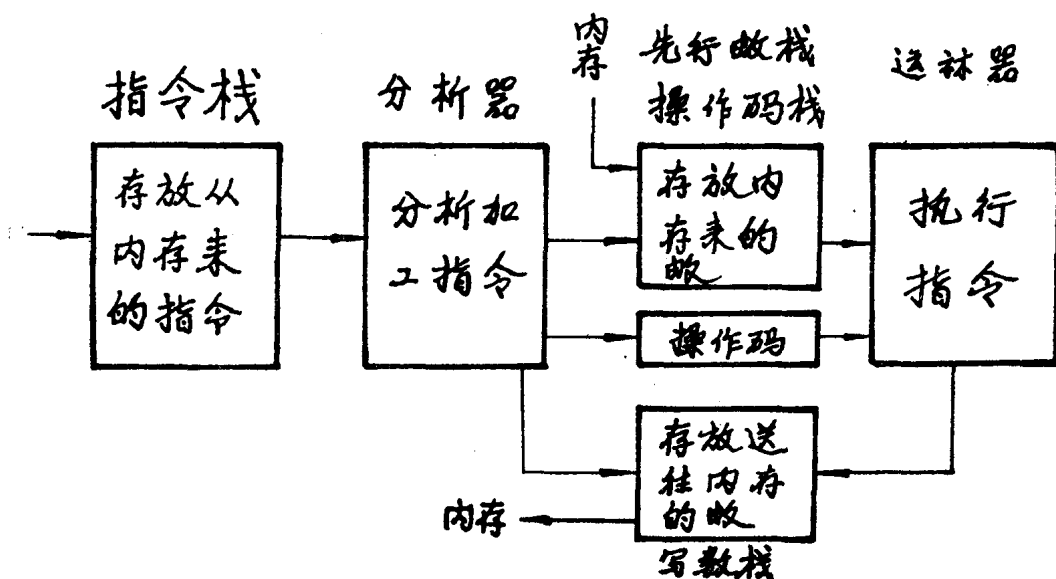


图3 先行控制器工作过程图解

代号说明：

- $J_{z0}-J_{z3}$ —— 先行指令寄存器
- J_{xs} —— 先行写数寄存器
- J_{sz1} —— 先行取指令计数器
- J_z —— 指令分析寄存器
- J_{sc} —— 后进先出区计数器
- X_d —— 下界地址寄存器
- J_{szf} —— 取指请求计数器
- J_z —— 指令发送计数器
- J —— 地址发送计数器
- J_{sf} —— 数发送计数器
- $J_{s0}-J_{s3}$ —— 先行读数寄存器
- $J_{s0}-J_{o3}$ —— 先行操作码寄存器
- J_{sz} —— 现行指令计数器
- J_{xsd} —— 写数地址寄存器
- G —— 后进先出区转接线
- S_d —— 上界地址寄存器
- J_{zd} —— 指令接收计数器

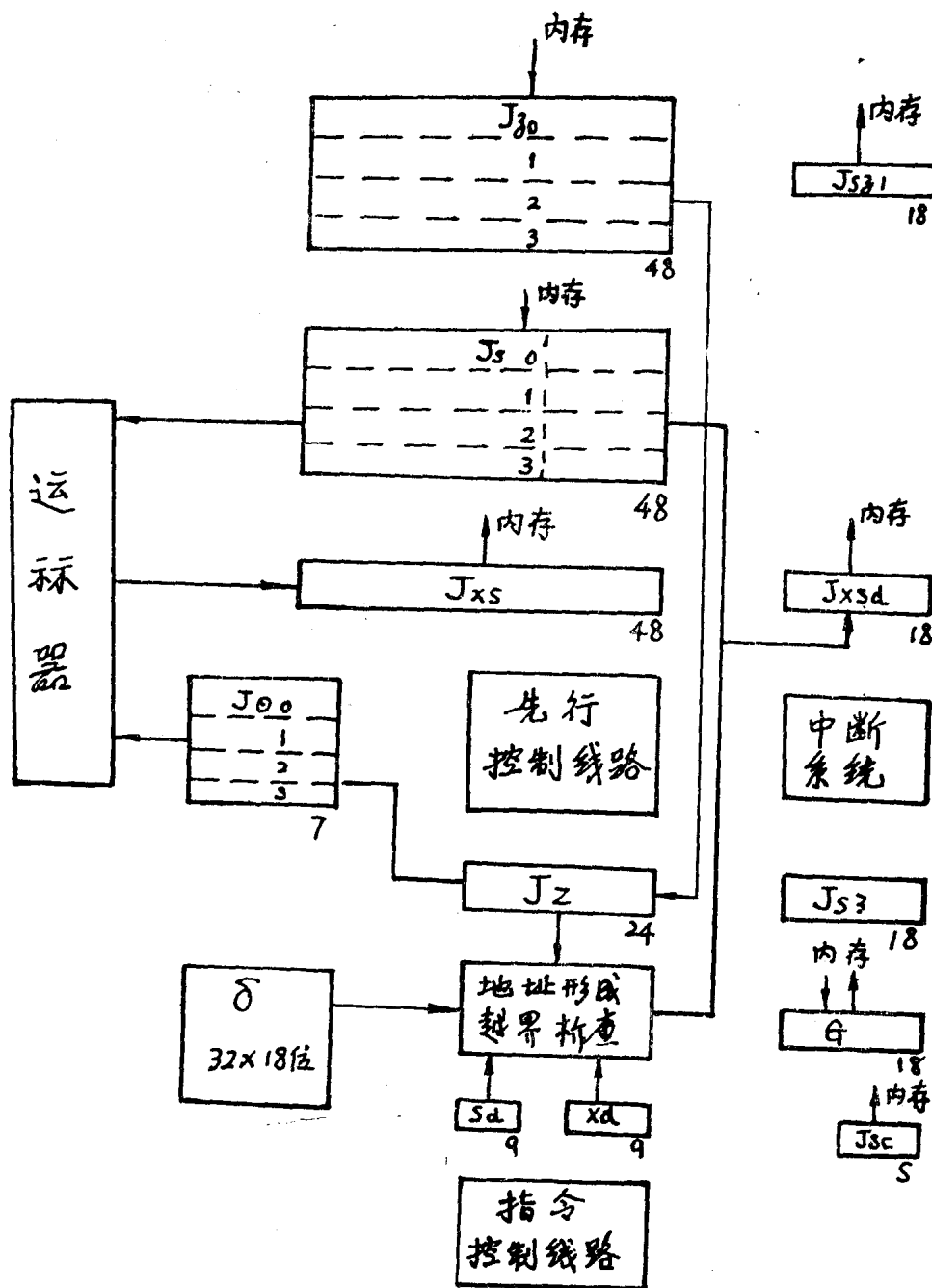


图4 中央控制器框图

J_{dj} —— 地址接收计数器

J_{sj} —— 数接收计数器

J_{qf} —— 操作码发送计数器

中央控制器框图如图4，图中只画了先行控制器和指令控制器两部分。先行控制器部分主要是由13个先行寄存器组成的先行指令栈、先行读数栈、先行操作码栈和先行写数栈，用

来分别存放从内存先行取来的指令，为运算器预先准备好的操作数、操作码和要存入内存的运算结果。指令控制器部分主要是对先行取来的指令进行分析加工。这里不再赘述先行控制器的工作过程，只做如下几点说明。

(1) 先行读数栈的 4 个寄存器和先行操作码栈的 4 个寄存器采取一一对应的关系，所以送给运算器的指令只是个操作码，没有地址，这对指令的加工和控制来说是比较简单的。

(2) 选取先行操作码栈的深度，应能缓冲掉取数的时间。平均取数时间，由于内存地址冲突引起等待，实际上要大于 1 个内存周期。如按一个半内存周期计算（即 10.5 拍），深度为 4 时，先行操作码栈可以缓冲 $2.985 \times 4 = 12$ 拍时间，所以是基本可以的。但运算器完成一次运算的节拍应按小于平均节拍来计算（因为除了算术运算指令外，其它运算型指令几乎都是 1 拍、2 拍指令），如果运算器按 2 拍计算，深度为 4 就显得较少。这个问题在一定程序上也要通过提高内存速度加以改善。

(3) 如果运算器完成一次运算按 2 拍计算，则指令控制器，2 拍分析加工一条指令（个别指令需要 4 拍），也显得稍慢，特别是因为它除了要加工运算型指令外，还要自己吸收掉一部分控制型指令。这样就会使运算器因指令供不应求而空转，机器效率下降。

(4) 写数栈只有 1 个寄存器，显得稍少了些。但只有一个写数寄存器在解决读写数地址符合问题上比较简单。

下面再讲一下采用先行控制后带来的三个问题。

(1) 转移的处理

在先行控制计算机中，由于转移往往要造成运算器的停步，对机器工作效率影响很大。

<1> 无条件转移

无条件转移指令，设计中安排在指令控制器分析时发现它，这时要把先行指令栈中已提前取出的那些指令作废，重新按转移地址请求取指令，分析指令及取数。这样的处理方法比较简单，如果要提前在先行指令栈取指令时发现处理起来就复杂多了。一般说来，无条件转移造成的时间损失比较少。如果分析到无条件转移指令时，先行操作码栈正好满着，那么还有 4 条指令可以缓冲（运算速度如按每条 2 拍计算，可以缓冲 8 拍时间），减少了时间损失。转移后按转移地址重新取指令、分析指令和取数，在最快的情况下，需要 $7 + 2 + 7 = 16$ 拍时间（如果这条指令不需要取数则为 9 拍），所以对于无条件转移造成的时间损失难以全部缓冲（除非操作码栈中有除法一类的多节拍指令）。

无条件带回转指令（转子）和无条件转移相同，但要将返回地址按后进先出区计数器 J_{sc} 内容存入内存中的后进先出区。由于利用内存存数返回地址。为减少延误，发挥先行控制潜力，专门设置了一个 18 位的寄存器 G 作为后进先出区转接栈，把存放在 J_{sc} 所指出的内存单元中的返回地址或循环起始地址，同时也保留在 G 中，这样在返回时就可以直接从 G 中取得返回地址。

<2> 条件转移

条件转移对机器速度影响比较大，指令控制器分析到条件转移指令时，采取了停步等到运算器把先行操作码栈的内容全部做完的办法，全空后说明转移条件已经产生，再按转移条件决定转向，继续工作。这时一切工作都要从头做起，在重新取得第一条指令（如果要重新取指令），分析加工这条指令，并取来操作数（如果要取操作数）的整个过程中，运算器一直要停止工作。为了补救这一损失，可以有一些办法，例如在遇到条件转移时一方面按转移方

向取指令以备转移成功时用，一方面按顺序方向继续进行有条件的分析加工，以便转移不成功时立即使用等。但这样做都比较复杂，我们就采取了停步等待这种最简单的办法。在设计中采取的具体做法是：分析到条件转移指令时，总是由先行指令栈按转移成功的路径提前取指令，指令分析则停步等待。如果最后证实转移成功，就可以减少取新指令的这一部分时间损失。这样做主要是为了同循环转移处理的方法取得一致；同时希望程序设计者能按转移成功几率大的情况来选择使用“0 转移”或“1 转移”指令。

〈3〉循环指令

150 机用循环开门和循环关门指令实现程序计数循环，指令控制器在分析到循环关门指令时，可以立即按循环次数判断继续循环或退出循环段。如继续循环时，从 G 寄存器中取出循环起始地址，相当于做一次无条件转移。所以每循环一次就要把先行指令栈的指令全部作废一次，很影响机器速度。为了解决这个问题 150 机设计了小循环开门和小循环关门指令，只要循环段指令不超过 8 条（包括开门和关门指令，而如果开门指令在右半字时，可达 9 条），就可以用小循环开门关门指令使先行指令栈把循环段内的指令全部保留不动（第一次循环时，先行取出的指令要作废一次），在先行指令栈内部进行循环，这时已取出的指令不但不作废，而且也不必再从内存取新的指令，因此机器在执行小循环时速度特别快。

这里有一点要说明的，选取先行指令栈的缓冲深度，应能缓冲掉指令的时间。150 机先行指令栈有 8 条指令，可以缓冲 14 拍（指令控制器每 2 拍分析加工一条指令），比实际需要的多。深度采用 8 主要是照顾到小循环段有较长的长度。

（2）形成数和形成指令

〈1〉形成数

由于采用了先行控制技术，运算结果真正写入内存的时间要比运算器执行完存数指令滞后一段时间。但是运算器所需的操作数却早在运算器执行之前已经先行取出。如果在设计中不采取措施，那么边形成边使用的数就要发生错误。例如：

K L → a; 累加器 L 中形成的数送内存 a 单元
K+1 → L a; 从内存 a 单元中取数入累加器 L

第 K+1 条取数指令先行取数时，第 K 条存数指令要存的数还未存入内存，取出来的数还是老内容，发生错误。为避免这一点，150 机中在这一部分设置了地址符合线路，当指令控制器分析到取数型指令时，都要和写数地址寄存器 J_{xsd} 中的写数地址进行比较，如果相同，就从先行写数栈 J_{xs} 中直接取得，这样来保证读取的数不会发生错误。

在 150 机中，由于有半字长和全字长两种数，使地址符合问题处理起来相当繁琐。因此采取的办法，只能在半字长写半字长读（或全字长写全字长读）同一单元内容时直接把先行写数栈 J_{xs} 内容送入先行读数栈，而在半字长写全字长读或全字长写半字长读虽然遇到地址符合，也是采取等待的办法，暂时封锁向内存取数，直到先行写数栈 J_{xs} 中的数真正写入内存时再申请读取。

〈2〉形成指令

在先行取指令时无法与写数地址寄存器 J_{xsd} 进行地址符合比较，因为这时和取指令地址相关的存数指令可能还未到指令控制器分析加工，写数地址还未送入 J_{xsd} 中。所以一般形成指令的办法不能使用，用了就要造成错误，在要用形成指令时，150 机中设置了执行指令来实现。这时程序编写如下：

K: TL → M, 把形成指令送到另一个单元 ϕM 中。

K+1 ↑ Z M, 执行指令, 执行 M 中的指令

.....

M: (形成指令)

执行指令的功能是: 按照操作地址 D 临时从内存中取一条指令来执行 (现行指令计数器——加工分析指令的计数器 J_{sz} 不加“1”), 执行后再继续做第 K+2 条。150 机在做“执行指令”时, 执行的指令是按读数, 方式从先行读数栈中取进来的, 可以发挥地址符合线路的作用, 保证取来的是新形成的指令。

另外, 也可以用编程序上的一些技巧或整机设计来解决这一问题, 这里不赘述。

(3) 中断的断点问题

由于采用了先行控制技术, 指令控制器在响应中断请求后, 不能立即执行中断隐指令进入中断子程序, 因为这时先行读数栈向内存请求取的数可能还未到来, 先行写数栈向内存请求写的数可能还未送出, 而且运算器中执行的指令其地址已经丢失 (这样的设计使机器紧凑些), 所以不能立即中断, 否则无法保留断点。

在 150 机中, 指令控制器在响应中断后, 就停止工作, 但一直要等到运算器把先行操作码栈中的指令全部执行完, 并在先行写数栈和后进先出区转接栈 G 中的内容都存入内存后, 才能转而执行中断隐指令进入中断。这样, 才能把精确的断点保留下来。

但上述情况只适用于外部中断。如果在运算中发生运算器溢出、内存偶错、写入禁写区等错误中断时, 原程序继续做下去已无意义, 就不进行等待, 立即转中断子程序。这时没有保留精确断点。对溢出只提供当时的现行指令计数器 J_{sz} 的内容 (指令控制器加工指令的下一个地址) 和出错误指令及其下一条指令的两个操作码; 对偶错和写入禁写区等错误, 只提供现行指令计数器 J_{sz} 的内容; 供查找断点时分析用。

为了帮助查找这些错误中断的精确断点, 150 机在操作控制台上设置了两种按键: “慢行”和“追踪”。

3. 提高运算器运算速度的主要措施

150 机的运算器有五个寄存器, 其中有一个作为累加器。设五个寄存器是作多位乘法和乘加指令所需要的 (多位乘法为了“存贮进位”要加第四个寄存器, 作乘加又要再加一个寄存器), 寄存器没有编址。

在设计中主要着眼于提高运算速度, 使之达到百万次的指标, 主要采取以下措施:

(1) 加法采取分组跳跃进位, 全加器的阶码和尾数都分成四位一组, 在组内产生组内跳跃, 在组间产生组间跳跃。进位链自加数和被加数形成后到产生和数, 阶码为 6.8 级, 尾数为 7.2 级, 使求和最长级数不超过 22 级 (与非门为一级, 与或非门 1.4 级计算)。

(2) 乘法采取 12 位一乘的多位乘法, 用 4 拍可完成一次全字长乘法。由于采取存贮进位的方法, 使乘法最长级数也不超过 22 级。

(3) 除法采取跳“0”“1”方法, 并增加了 $\pm \frac{\text{除数}}{2}$ 的比较点, 线路保证每次最多可以上商 4 位, 使平均每拍能上商 2.5 位左右。全字长除法平均为 18.5 拍。

(4) 在对阶和规格化时, 一次最多能左右移 4 位。

(5) 节拍重叠使运算器能重叠执行 3 条指令, 即在每条指令执行到最后一拍时, 同时完成下一条指令的操作数和下下条指令的操作码的动作。

(6) 数采用补码系统表示, 可以节省加减法运算的求补节拍。

(7) 设置一系列“控制触发器”, 提前一拍把它们置“1”, 为下一拍的控制动作做好准备, 这样可以缩短控制电位的逻辑级数(使控制电位的条件组合级数在上一拍中“吸收”)。

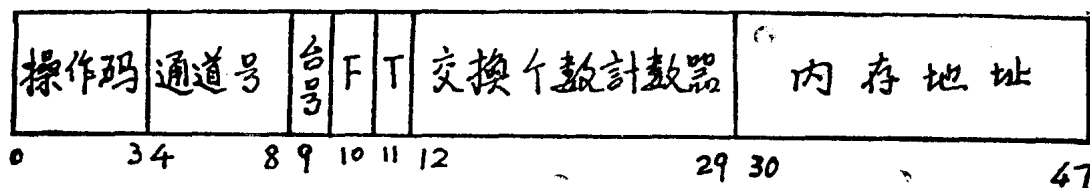
(8) 其它(在各处都尽量设法减少逻辑级数)。

4. 交换器

为了实行多道运行充分发挥机器高速运算的能力, 150 机设计有交换器, 使得外部设备和主机之间能同时工作, 外部设备间也能同时工作。

交换器有一个容量为 128 字, 字长 48 位(加校验位共 54 位)读写周期为 $0.9\mu\text{s}$ 的快速磁芯存储器, 作为外部设备和内存贮之间进行信息交换时的数据缓冲和存放通道控制字及通道控制字地址。

150 机的通道控制字共有 13 条, 其一般形式如下:



FT 是两个特征位。

外部设备和交换器之间的数据交换以字节(8 位二进制代码)交叉方式进行, 6 个字节凑成一个全字长数。交换器和内存贮器之间则以 48 位全字长数并行传送(见图 1, 150 机框图)。

交换器和外部设备控制器之间是用具有 28 根双扭线的标准电缆相连接, 这样使外部设备的加接、更新、位置的变动等具有较大的灵活性。

设计过程中曾考虑把磁盘存储器纳入交换器, 但因磁盘存储器的频率太高($250\text{KC}\sim 400\text{KC}$)会造成信息丢失。最后还是使磁盘单独和内存贮器交换数据。

5. 多道程序运行

多道程序运行, 可以发挥机器效率, 有可能使机器一直处于运行状态。150 机最多允许四道算题程序参与运行。在这里只介绍 150 机中为实现多道程序从整机方面给予的保证。这些整机的保证, 除了上述的交换器外, 还有中央控制器中的 3 位道号寄存器、1 位状态触发器、9 位上界地址寄存器和 9 位下界地址寄存器等。

(1) 道号寄存器和状态触发器

多道运行程序在选道时, 用专门的“送道号”指令把选中的程序的道号送入道号寄存器。在 4 道算题程序都暂时“挂起”或机器没有题目要算时, 还有一道进行等待的空转程序, 所以道号寄存器有 3 位。状态触发器用来表明机器在执行管理程序(管理态)还是在执行算题程序(目的态)或空转程序(服务态)。

道号除供管理程序查用外, 在对指定的某一道算题程序进行“追踪”或“慢行”等操作时, 也要用到。

(2) 浮动地址和越界保护

每道算题程序都由管理程序分配内存单元, 在形成地址时由求地址线路自动加下界地址。并且还要进行上下界的越界检查, 如果形成的操作地址 D 小于下界地址或大于上界地址就是

地址越界，发错误中断加以保护。上界地址和下界地址都只是9位，各设置了一个9位的寄存器存放。每次转道时，由管理程序用专门的“置下界”和“置上界”指令把下界和上界地址分别送入这两个寄存器。这两个地址长度只有9位，在形成地址时通过线路把它们变成18位（线路自动在下界地址之后补9个“0”，在上界地址之后补9个“1”，这说明内存在分配给各道程序时只能以512单元为最小的分配单位）。

(3) 管理程序的保护——禁写区

150机中把第4096到16383号单元（共12288个半字长单元）作为专放管理程序的禁写区（对每一个磁芯体来说就是第511—2047号全字长单元）用线路进行保护。如写入时的地址在此范围内，即发出错误中断，并中止内存工作，进行保护。

(4) 控制台打字机的使用

在多道运行时，一般是用控制台打字机打入“操作命令”的方法来进行人机联系，机器根据操作命令，通过管理程序实现人对主机和外部设备的指挥。

三、调试过程中的主要问题

150机在调试过程中遇到了因设计考虑不周而带来的逻辑上的问题，但主要的还是元件、工艺可靠性方面的问题。前者容易发现和解决，后者的发现和解决往往要费很大的气力。因此，这里主要谈谈元件、工艺可靠性方面的问题。

1. 固体组件

150全机共用TTL固体组件近六万块，上机前都是经过常温功率老化、高低温冲击和交、直流参数测试等步骤筛选的。尽管如此，上机后仍然陆续发现失效组件，由于组件失效往往有个缓慢过程，因此调机过程中，为了找一块将要失效的组件有时竟化一两天时间。

150机的组件失效情况如图5。

