

12240

M-3小型电子计算机

苏联部长会议
科学技术委员会

苏联科学院

科学技术情报研究所分所

目 录

绪 言	1
I. 在各种工作情况下, 机器构件间之相互关系	5
输入原始数据和存储器 3 γ 的工作	5
借程序发送器 11 Π 和存储器 3 γ 来自动执行程序	6
在运算器 A γ 内进行运算	8
II. 算术运算和逻辑运算	9
原码, 反码和补码	10
加法	10
按绝对值比较两数	13
减法	13
乘法	15
除法	17
逻辑乘法	19
变形的加法和减法。数的运算	20
III. 线路和部件的元件	22
触发器	22
单稳多谐振荡器	24
形成器	24
门	25
二极管逻辑线路	26
阴极跟随器	28
方框标帜图及时间图	28
脉冲计数器	30
数从一个触发单元传到另一个触发单元的方法	32
IV. 运算器	33
一般特性	33

	实现基本操作时的 Ay 线路的工作	36
V.	程序发送器	41
	局部程序发送器 МПД	42
	操作部件 БС	45
	起动寄存器 ПР 和选择寄存器 СР	47
	脉冲分配部件 РИ	50
VI.	存储器	53
	存储器写入和读出的工作情况	53
	磁鼓	54
	写入和读出放大器	56
	号码脉冲和零脉冲。选地址系统	57
	控制线路 УМП	59
VII.	输入和输出数据的设备	64
	输入代码。穿孔带	64
	发送器	66
	输入时构件的工作情况	69
	输出代码	71
	输出时电传机的工作情况	74
	输出时构件的工作情况	78
	电子输出器	79
VIII.	M-3 机器的电源	81
	电源系统	81
	电机组	82
	电源柜	83
	整流器	83
	电压稳定器	84
	保护电路	84
	电压指示器	86

IX. M-3 的结构	86
控制机构	89
X. M-3 的使用	91
程序设计。编码系统	91
简易题目的程序设计举例	95
数据的准备和纸带的穿孔	97
闭合及断开电源	97
输入程序和原始数据	100
完成计算及输出结果	101
机器的程序校验	102
附录 1. 主要技术特性	103
附录 2. M-3 机器的指令码	105

绪 言

在解决科学和技术题目的巨量计算时，计算过程的自动化有着重要的意义。为此，现今有效地运用着电子数字计算机。它的工作效率比人工要高出几十几万倍。

除大型电子计算机外，小型计算机也得到广泛应用。大型电子计算机，如箭牌计算机需要很大的维护费用，专门的运转条件（大房间、电源、冷却设备等）。小型计算机工作效率较低，但没有上述缺点。

M-3 计算机系由苏联科学院控制机和控制系统研究室(ИЯМ АН СССР)和苏联国家计委会电气工业科学研究所合作制成。

M-3 机系由苏联科学院控制机和控制系统研究室初步设计，与苏联国家计委会电气工业科学研究所共同设计完毕；由电气工艺科学研究所制造。

M-3 通用小型电子计算机按其使用性能来说，是供科学研究所与设计局应用的。M-3 机能解这些题目：积分常微分方程和偏微分方程（线性的以及非线性的），含有多个未知数的线性方程组，代数和超越方程等。

实际上，如果问题的数据在内存储器里放不下时，只要把工作的速度降低就可解任何问题。

在 M-3 机里广泛使用了半导体元件，机器主要的逻辑元件是半导体二极管做成的非符合译码器。很多计算和逻辑线路都用到了它。为这个机器研制了小型写入磁头及写入和读出脉冲的经济的电路，机器做成几个独立的柜子形状，每个柜子是单独的大部件，以利改进机器的某些构件，及便于运输和按装在不同的房间。机器具有自动的电源。

※制定简图、设计与各个阶段的调整工作的参加者是，尤·勃·普尔瑞叶夫斯基，恩·阿·多罗霍娃，阿·勃·扎尔金

德，格，伊，达涅托夫，符，恩，奥符察里柯，阿，雅，雅柯夫莱夫，勃，梅里克——沙赫那扎罗夫，阿，巴，托尔马索夫，阿，恩，巴塔里凯也夫，符，阿，莫罗佐夫，伊，阿，斯克里普金，阿，符，皮皮洛夫，符，恩，谢梅诺娃。

整个工作是在苏联科学院通讯院士勃洛克总的领导下进行的。

机器由四个主要构件组成：运算器 AY ，程序发生器 $ПД$ ，存储器 $ЗУ$ 和输入输出装置 $УВВ$ 。

(见方框图，图1)

图1. $M-3$ 的方框图

AY — 运算器； $ПД$ — 程序发送器； $БД$ — 操作部件；
 $МПД$ — 局部程序发送器； $СР$ — 选择寄存器； $ПР$ — 启动寄存器； PN — 脉冲分配器； $УВВ$ — 输入输出设备；
 $ЭУВВ$ — 输入输出设备的电子部件； $ЭВ$ — 电子输出器；
 $ЗУ$ — 存储器； KC — 检验线路； $СММ$ — 号码脉冲计数器；
 $УМП$ — 磁存储器的控制部件； $УЗЧ$ — 写入和读出放大器；
 $НГ$ — 另磁头； $МГ$ — 号码磁头。

机器工作时运算器用来对数进行数学和逻辑运算。

在存储器里存有运算时所必需的原始数据，中间和最后计算

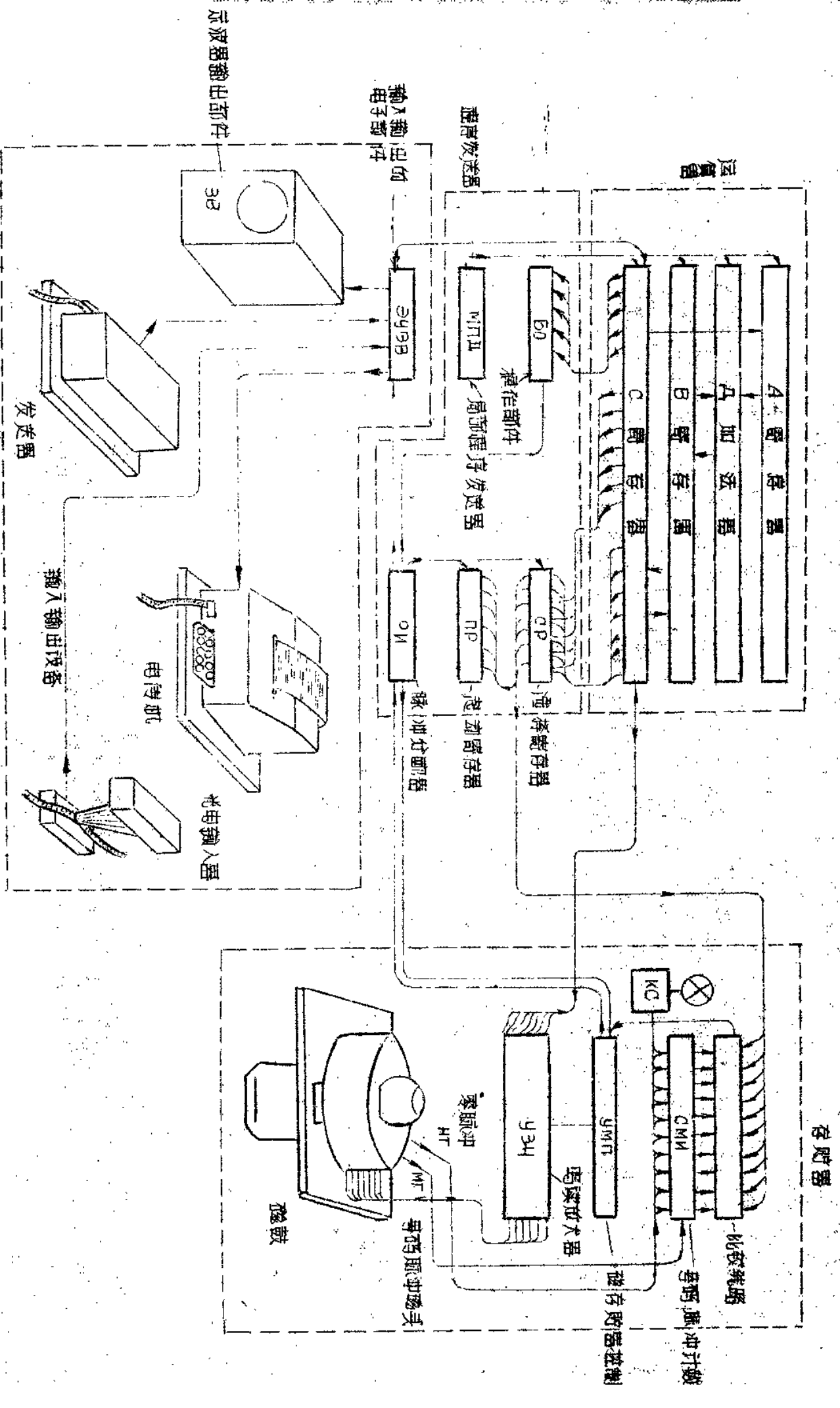


图 1 M-3 的方框图

结果；此外，在存储器存有解题的程序，也就是确定对数进行基本运算（加、减等）的顺序的一组指令（命令）。输入输出装置用来在解题开始前和运算过程中向机器输入原始数据、程序指令以及输出（打印）最后结果。

控制器——程序发送器 ПЗ — 用来控制其他部件。程序发送器利用顺次指令中的命令在存储器里找所需要的数，确定对数进行什么运算，在进行给定的运算时控制运算器，往输出设备送计算结果等。

每一个寄存器的二进制位由存储二进数的触发器与接收信息至触发器的门组成。

I. 在各种工作情况下机器构件间之相互关系

输入原始数据及 34 的工作

编成数码形式的数据及解题程序（根据“M-3 机的使用”章中的规则）另外借穿孔机打在纸带上，并送到运算器的 C 寄存器内。向寄存器 C 输入每个数时，同时将此数的地址（贮存该数的存储器单元的号码）送到寄存器 CP 内或者利用程序自动地将该数地址送到 CP 内。这两个寄存器皆与存储器 34 相连；寄存器 C 与写入和读出放大器相连，而寄存器 CP 则与比较线路相连。

号码脉冲计数器找出存储器中所需要的单元，这个计数器数着来自特殊的号码读头 МГ 的脉冲。在号码磁头下面，磁鼓的一周共记录 2048 个脉冲。

这样，磁鼓的一周被号码磁头分为 2048 个单元。

在第一个号码脉冲和最后一个号码脉冲之间，由零磁头 НГ 送出的特殊脉冲（“0”脉冲）将 СМИ 清除。

为了在存储器 3 γ 内写数和读数，沿着磁鼓的磁道，设置了一组 31 个读写磁头（按数位的数目）。每一磁头都有本身的放大器，这放大器是通到 C 寄存器相应位去的。

在某一给定时间内，磁头下面的存储器单元号码（地址）放在号码脉冲计数器 CMI 中。当选择寄存器 CP 内的地址与 CMI 内当时的地址符合时，比较线路就给出脉冲。这个脉冲进入到第 1—31 号磁头的读写放大器中。根据 C 寄存器某一位内的二进制数（0 或 1），与该位相应的放大器将正电流或负电流送入自己的磁头内，这电流就将磁头下面的磁鼓表面部分磁化到正方向或负方向。

这样，如果从 4BB 输出的数放在 C 寄存器内——而它的地址处于选择寄存器 CP 内，并且向存储器控制部件 YMI 送入写入指令，则此数就被写到磁鼓的相应单元中。写数指令输入时是以特殊代码的形式打在穿孔带上，并且当纸带通过发送器或光电输入器时这些数据就自动地转送到存储器的所需单元内。

从 3 γ 选取数是以下列形式进行的：为了读出需要的数，要将它的地址记在 CP 内，而读数指令则送入 YMI 内。比较线路与写数时一样，当 CP 和 CMI 中的地址符合时就给出脉冲，这个脉冲进到读出放大器。

依照每一个读头的信号极性，符合脉冲或者通过或者不通过放大器的输入门线路而到达 C 寄存器相应位的触发器，并将它建立在需要的状态。因此，当读数时，自 3 γ 中取出的数是在 C 寄存器内。

借程序发送器 ПД 和存储器 3 γ 来自动执行程序

在解题过程中，ПД 控制着机器，从 3 γ 取出的指令码由 ПД 进行译码。此时由 ПД 决定，送到运算器的数从那里选取，对数进行何种操作，结果从 A γ 传送到那里去及下一个指令放在 3 γ 的那一个地址。

将程序的第一个指令的地址置于 ΠD 后，以后的计算就自动地进行。每一个指令在 ΠD 内都是在 8 拍内完成的。在 ΠD 的每一工作节拍时，脉冲分配器 PI 送出相应的脉冲至机器的各个部件，并接收这些部件的关于这些指令已经完成的回答脉冲。只有在接到回答脉冲后才做下一拍。

第一拍时，指令地址从 ΠP 传送到 CP ， PI 将读数命令送到部件 YMP 。

第二拍时，自 3γ 中取出的指令放到 C 寄存器内。

第三拍时，将 ΠP 内的指令地址加 1，因为通常指令皆是依次选取的，只有转移指令才改变这个顺序。此外，在第三拍时，指令操作码进入到操作部件 BO 内，同时指令的第一地址（第一个数的地址）送到 CP ；部件 PI 又将读数命令送到 YMP 内。

第四拍时，部件 PI 将指令的第二地址（第二个数的地址）从寄存器 C 送到 CP ，此后，第一个数就送到 C 寄存器。

第五拍时，部件 PI 将第一数传送到 A 寄存器并送出读数命令。

第六拍时，第二数自 3γ 取出并送到 C 寄存器。

第七拍时，部件 PI 将第二数自 C 寄存器送到 B 寄存器并送出操作命令。

由 PI 进到 BO 的操作命令以命令的形式从 BO 输出来，在它输出后，当下一个第 0 节拍时，起动局部程序发送器 $M\Pi D$ 来执行某一个规定的运算（加，乘等等）。此 $M\Pi D$ 在起动后送出一定的脉冲序列至 $A\gamma$ 之控制线路。操作完成后， $M\Pi D$ 把写数命令及始终处在 C 寄存器内的运算结果送入 3γ ，记下 CP 内的第二个数的地址。在这以后， PI 就转到执行下一个指令。

上面所述顺序是相当于执行代码为 00, 01, 02, 03, 06 的指令（即加，减，除，乘和逻辑乘）。作其他代码的运算时可以不读第二数，不写数等等。也可以用另外的形式来执行数的传送，条件转移和无条件转移的运算指令。

在 A_4 内进行运算

在 A_4 内完成的基本运算是两数相加。取寄存器 A 和 B 中的数为补数的设备使得在 A_4 中能作减法，寄存器 B 和 C 能左右两面移位使 A_4 可做乘法和除法。

加法时，寄存器 A 和 B 中的数相加，相加的结果送到寄存器 B 来代替被加数。

加法器的触发寄存器 Π 是为了形成进位用的。加法系并联完成；每一位皆有相全的逻辑线路，这些逻辑线路按二进制加法的规则将寄存器 A，B 和 Π 各位组合进行译码，以便将和送到 B 寄存器中。

作除法时，B 寄存器中放被除数，A 寄存器放除数（它在 A 中已变为补码），除法是以连续的“试”和“减”来完成的。所得的商数在 C 寄存器中移位。乘以 2^{30} 的余数在寄存器 B 内形成。

逻辑乘法（由数中抽出一组数位）按下列形式进行。

A 寄存器内放“抽数”，即这个数中是“1”的那些数位就相当于是需要从第二个数中抽出的那些数位。A 寄存器的数位中为“1”，则 C 寄存器中对应于这些数位的数字不发生变化。A 寄存器的数位中为“0”，则 C 寄存器的对应数位皆被置于“0”状态。

上述的全部算术运算和逻辑运算，皆可以根据已经操作完毕的第二个数的地址写入结果来结束这些操作。

此外，每次操作的结果在 B 寄存器中一直保存到接收新数为止。第一，这可以不将中间结果送入 3_4 ，利用 B 寄存器来作存储中间结果的辅助存储单元（当上次运算结果在下次运算时要应用时）。因此，有可能减少访问存储器的次数，从而提高机器的工作速度。

这种操作称作“累加操作”。所有上面提到的操作皆可以用适当的形式来编码。

第二，有时需要不按照第二个数的地址，而是按某一个另外的地址来写入结果。根据任意地址来写入结果是由无条件转移操作来实现的，同时它能转移到执行不按地址顺序排列的下一条指令。

无条件转移操作就象条件转移和条件进位操作一样，与在 Ay 中执行的那一种运算无关。作无条件转移时，上次运算的结果从 B 寄存器送到 C 寄存器并依照所需的地址记在那儿。当数从一个单元传到其他单元时，自 $3y$ 读出的数存于 C 寄存器内一直到往 $3y$ 按新地址写数时为止。作条件转移时，根据 B 寄存器内由上次运算遗留下来的数的符号或者送这个指令的第一地址或者送第二地址到起动寄存器。

将命令送到输入输出的控制器从而完成输入和输出的操作。任何操作的结果，如果要记入 $3y$ 中的话，皆可以借 C 寄存器连续向左移位的方法来取出结果并在电传机上一个一个相挨地打成数字。在“输入操作”时，应用发送器或光电输入器将数从纸带输入到机器的存贮器。

II. 算术运算及逻辑运算的完成

$M-3$ 计算机采用定点的二进位记数制。在机器内进行操作的全部数字，它的绝对值应小于 1，或者说，机器按模数 1 进行计算，所以加上一整数后，原数不变。举例来说，设 a 是机器内表示的数，则 $|a| + m = |a|$ ，其中 m 是任意整数。一旦当中间计算结果值等于或大于 1 时，就自动地中断计算，同时将机器停止。

触发器及寄存器用来存放运算器及控制器中的数字，同时也用来表示代数符号。

在这一章里，叙述并联运算器内进行算术运算及逻辑运算的

一种方法 (M-3 所采用的方法)。

原码、反码及补码

所有的数都可以用原码、反码及补码来表示。原码——数的原来表示形式——在 M-3 中作为基本数码。反码 (对二进制而言) 将原码的全部的 1 代以 0, 0 代以 1。要得到补码需要 (按模数小于 1 的表示数) 从 1 减去用原码表示的数。例如, 原码表示为:

$$a_{np} = 01101;$$

这一数表示为反码:

$$a_{op} = 10010;$$

这一数表示为补码:

$$a_{on} = 10011.$$

从而可知, 要得到补码必须在表示为反码数的最后一位加上 1。

因此 $|a_{on}| = 1 - |a_{np}|$, $|a_{on}| = |a_{op}| + 2^{-n}$, 其中 n 是有效位数。

在 M-3 运算器中, 可以从存放在寄存器 A 及 B 内的数得到补码; 要在最后一位附加 1, 可以把加法器 II 寄存器的最后一个触发器迁立“1”状态。

加 法

加法是运算器的基本运算。实际上, 其他的算术运算都可归结为加法。

通过一个例子来导出加法规则, 最为方便。将二个五位二进制数相加:

$$\begin{array}{r}
 0,0110 \\
 + 0,0111 \\
 \hline
 0,1101
 \end{array}$$

在进行按位加法时，要考虑到二个相加数的相应位，同时也要考虑到前一位的二进位数进位，并且，进位数应当在和数位形成之前预先形成。

在运算器内，二个相加的数放在寄存器 A、B 中，二进位进位数在寄存器 M 中形成，和数用改变一个被加数的方法放在寄存器 B。

由此可见，加法时各寄存器内的数分配如下：

$$\begin{array}{rcl}
 \text{被加数} & 0,0110 & \text{—— 寄存器 A} \\
 \text{进位数} & 0,1100 & \text{—— 寄存器 M} \\
 \text{加数} & 0,0111 & \text{—— 寄存器 B} \\
 \hline
 & 0,1101 & \text{—— 寄存器 B}
 \end{array}$$

我们现在来讨论三个寄存器中同一位数的各种可能组合状态（见表 1）

表 1

寄存器 A 被加数	寄存器 M 来自低位的进位	寄存器 B 被加数	寄存器 B 和数	寄存器 M 向下一高位的进位数
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

根据此表，可以得出下列二条在运算器加法器内作加法时的逻辑规则：

I, 求和时, 如果寄存器 A 与 B 的同一位触发器处于不同状态, 则必须改变寄存器 B 该位触发器的状态; 如果寄存器 A 与 B 的触发器处于同一状态, 则寄存器 B 的触发器的状态就无须改变。

II, 在形成进位数时, 如果同一位触发器 (A、B 和 C) 中有两个处于“1”的状态, 则必须将寄存器 C 的下一高位触发器置为“1”的状态; 如果同一位触发器 (A、B 和 C) 中有两个或三个处于“0”状态, 则下一位的进位触发器也应置为“0”状态。

作加法时, 可能得到等于或大于 1 的和。

例如:

寄存器 A 0.1010

寄存器 B 1.1100

寄存器 C 0.1011

寄存器 C 1.0101 和数超过 1。

和数形成前, 就可以根据寄存器 C 的最高一位触发器是否存在进位 1 而知道和数等于或超过 1。这个触发器称为寄存器 C 的零位触发器 (T-O-C), 如果存放在寄存器 A 与 B 中的和数等于或大于 1, 它一直处于“1”状态; 反之, 则触发器处于“0”状态。因此, 在作加法时触发器 T-O-C 出现“1”就应当停止计算。

在作加法时可能发生这种情况, 即最低一位形成的进位会一直进位到最高位。例如, 各数这样配合:

寄存器 A	0	1	0	0	1	1
寄存器 B	1	1	1	1	1	0
寄存器 C	0	0	1	1	0	1

在这种情况下和数等于 1, 所以加法就不能进行。

全部进位所需的时间是计算器的一个很重要的指标, 它在很大程度上决定着加法的速度, 从而也决定着其他操作的速度。每当转移到下一次操作的时候, 必须检查触发器 T-O-C 的状态,

这就得花费一定的时间。

二数绝对值的比较

比较是减法与除法运算的一个组成部分。二数的比较要用到寄存器 A、B 与 T。在寄存器 A 中取出补数，然后根据触发器 T-O-T 的状态就可以知道那个数较大。

设 a 为放在寄存器 A 中的数， b 为放在寄存器 B 中的数。
在寄存器 A 中取补码后，则为：

$$|a'| = 1 - |a|.$$

$|a'|$ 与 $|b|$ 的和为

$$S = |a'| + |b| = 1 - |a| + |b| = 1 - (|a| - |b|)$$

$$\text{如 } |a| > |b| \quad S < 1$$

$$\text{如 } |a| < |b| \quad S > 1$$

$$\text{如 } |a| = |b| \quad S = 1$$

由此，将放在寄存器 B 的数取补码后，如果触发器 T-O-T 处于“1”的状态，则表示 $|a| \leq |b|$ ；如果是“0”状态，则 $|a| \geq |b|$ 。

M-3 机器运算器线路的特点是比较二数时，无须对二数本身进行什么操作（除了将其中一个数取补数以外），这就使得除法大为简化。

减 法

作减法时，减数 $|b|$ 存放在寄存器 B，被减数 $|a|$ 存放在寄存器 A 内，差数，在寄存器 B 中形成。

在 AX 中作减法可以归结于数值相减，如果 $|b| > |a|$ 则 $|b| - |a|$ ，或者如果 $|a| > |b|$ ，则 $|a| - |b|$ ；因此，减法可以是 $|a|$ 、 $|b|$ 二数的任意组合。

求二数之差是把被减数与减数的补数相加，为了决定那一个