DJS-062D*微型计算机

马占元

第一章 概 述

DJS—062—D 型微型计算机是在我所研制的 DJS—061,DJS—062A,DJS—062B 型微型计算机的基础上开始研制的，它主要是仿制美国的 MOTOROIA 公司的 MEK—6800D_2，因此,它吸收了 MEK—6800D_2的主要优点

DJS—062—D 型机主要由中大规模集成电路组成，其中有微处理器MPU，通用并行接口PIA，异步串行接口ACIA，只读存贮器ROM，及静态随机存贮器RAM，除了以上这些 NMOS 大规模集成电路外，还有部分TTL电路。

DJS—062—D 型机从硬件方面看，系统简单，灵活性大，容易扩充，从软件方面看具有丰富的寻址方式，编程效率高，在配以相应的外部设备和应用程序后适用于多种用途，总之，DJS—062D型机基本上做到与MEK6800D_2硬件互换，软件相容，因此，可以说DJS—062D 型机的设计达到了预期的技术指标，收到了比较好的效果，现在D型机已被广泛的使用到全国各有关单位，已深入于各行各业，因为本系统设计合理，性能稳定可靠，深受广大用户的欢迎和好评，为了方便广大用户对本机的主要结构性能和特点作进一步了解，下面就本系统主要的几个方面作一简单介绍。

1.1 DJS—062D型机的框图

以下是DJS—062D型机的框图，从这个框图可以看出，DJS—062D 采用双总线结构，并且在主机部分，即该框图虚线以左的部分，双总线不加任何驱动，MPU 不加任何缓冲器直接和ROM,RAM,PIA 以及 ACIA 相连，LN6800 微处理器的地址总线的输出可驱动一个 TTL 负载和 130PF 的容性负载，因为本系统的组成是完全按照美国MOTOROLA公司的MEK—6800D_2 设计的，因此是一个标准的小规模系统，其走线分布电容小于130PF，数据总线也是一样，不同的只是数据总线是双向总线。

为了满足用户对于存贮容量，并行接口，串行接口的更大需求量，因此，增设了扩充板，为了保证LN6800 不致过载，因此LN 6800 和扩充板之间增加了地址总线不反相驱动器和数据双向总线驱动器,实践证明，这种考虑是完全合理的，保证了系统稳定可靠的运行。

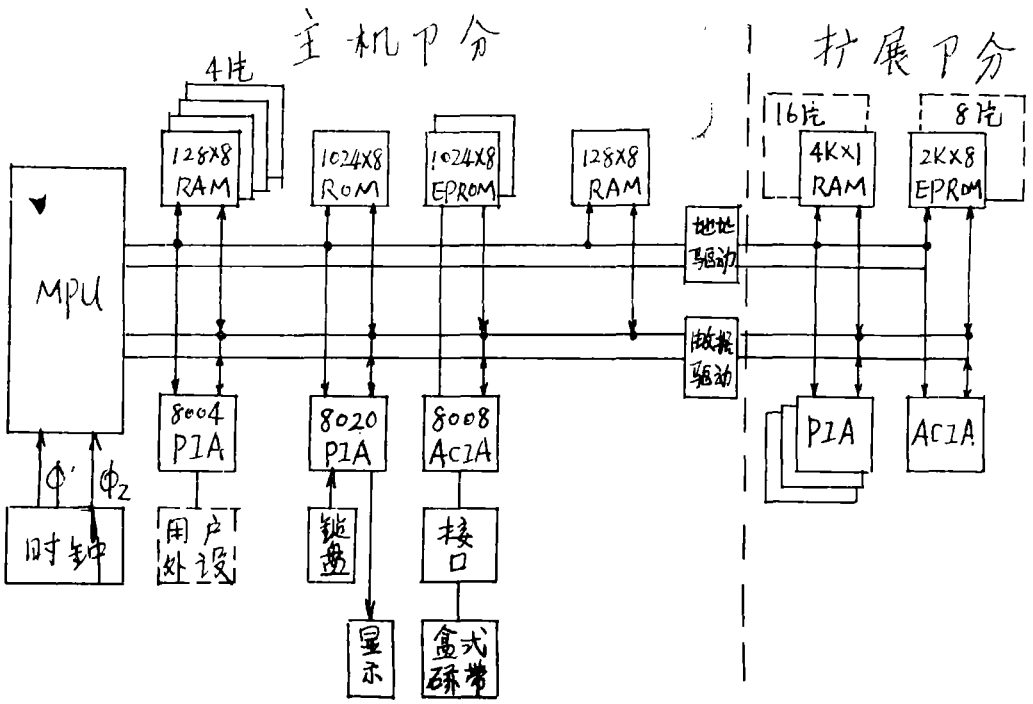
1.2 硬件资源

这个标准的系统包括下面 NMOS大规模集成电路：

数量	器件
1	LN6800MPU
1	LN6830ROM，固存本机监控程序
5	LN6810RAM，128×8 其中一片用于堆栈区，其余 512 字节可放用户的固定程序，
2	LN6820PIA，并行接口适配器。
1	LN6850ACIA异步串行接口。
2	MCM68708EPROM（等于2708）存贮容量为2K字节

除了以上这些NMOS大规模集成路外，本系统还使用的TTL电路如下：

※ 本刊编辑部于82年5月22日收到。



数量	器件
11	T077双D触发器
11	T522C_4双输入器与非门
3	T7417不反相6线驱动器
3	T74368反相三态驱动器
1	T574 双四选一电路
1	T516 单D触发器
3	T521C_4 OC门
1	SD404A 4—10译码器

硬件资源所列器件除了TTL电路中的部分器件是外购件而外，其余所有NMOS大规模集成电路和TTL电路均系本所研制生产的。

1.3 键盘和命令键

DJS—062D型机的键盘是一个包括24个单键的小型键盘，采用扫描技术工作方式，共有16个数字键和8个命令键，该键盘显示模块联带着JBUG监控程序，可以检验微型计算机的运行，程序的输入与程序的故障。八个命令键代表如下各项功能：

M：检查和改变存贮器。

E：撤消进行中的操作。

R：检查微处理器寄存器PC、X、A、B、C、SP的内容。

G：进入规定的程序并开始执行指定的程序。

P：将数据由存贮器录入磁带。

L：由磁带存入存贮器。

N：跟踪一条指令。

V：设置和清除断点。

在调用任何命令之前，显示器应显示闪动的短划。

（1）存贮器的检查和改变(M)

此功能可以检查，必要时可以改变存贮

器的内容。可以用输入地址（用键盘输入十六进制的4位数字）并用M键结尾（即hhhhM）的方式检查存贮单元的情况。此时显示器左面4位将显示输入的地址，而右面两位显示器将显示其内容。用户此时可以有三种选择。1、保留这个单元内容不变，而用通过按G键进入下一个单元。显示器上显示新的地址和内容。2、通过键盘(hh)输入新的数据以改变原数据，显示器上立即显示已引入的新数据。当试图改变只读存贮器(ROM)时，显示器继续显示原数据。3、用E键结束存贮器的检查功能，按下键E将恢复到监控程序重新显示闪动的“短划”。

（2）撤消(E)

此功能用以有秩序的从其它功能中撤出，有关应用的例子，将包括在其它的功能论述中。

（3）寄存器显示(R)

当显示JBUG程序的闪动“短划”出现以后，按下键R，就可以实现检查微处理器寄存器的功能。在闭合R键以后，显示器将显示4位十六进制数字的程序计数器的内容。其余的寄存器，可顺序的通过键G来检查。它们按下面的次序出现：程序计数器变址寄存器，累加寄存器A，累加寄存器B，条件码寄存器，堆机指示器，

显示是循环的，即当显示完堆栈指示器以后，再闭合G键就又重新显示程序计数器，在显示顺序中的任一点上都可以用键E使之回到监控程序。如需改变寄存器的内容，可以用“存贮器改变功能”。当调用R功能时，监控程序执行一个中断顺序，在执行中断程序中，MC6800将寄存器保存在存贮器堆栈中（这些存贮单元，正是R功能检查的），当由R中断服务程序出来以后，MPU使这些数值重新存入寄存器，因此，如果用M功能改变了堆栈中的内容（数据）这个新的数据将进入MPU。寄存器的堆栈单元如下：

A008——堆栈指示器的高字节。

A009——堆栈指示器的低字节。

S＋1——条件码寄存器。

S＋2——累加器B。

S＋3——累加器A。

S＋4——变址寄存器的高字节。

S＋5——变址寄存器的低字节。

S＋6——程序计数器的高字节。

S＋7——程序计数器的低字节。

此外，“S”是现在的堆栈指示的内容并保存在$A008和$A009中，当需要改变寄存器的数值时，注意必须撤出“R显示功能”并引入M功能。

（4）进入用户的程序“G”

如显示闪动的“短划”，并且假定必要的程序已输入，MPU可以开始执行程序，执行的方法很简单，进入程序的开始地址，并以键G结束(hhhhG)。显示器显示的空格表明，MPU已离开监控程序而正在执行用户的程序。MPU将继续执行用户的程序，直至按下键E或者程序完了。从闪动的“短划”表示程序的状态通常可用键E得到。有时候一个不正确的程序，可能引起修改监控程序的可变数据。在这种情况下，需要用微型机模块上的复位开关使之恢复控制。

（5）由存贮器录入磁带“P”

录入功能允许用户把存贮器某个区的内容，保存在普通的音频盒式磁带机里，在录入前，要使用存贮器改变功能，确定存贮器的那些部分要录入。用存贮器改变指令，将需要录入的开始地址送入$A002和$A003单元(高字节进$A002，低字节进$A003)，同样将终了地址的高阶字节和低阶字节分别送入$A004和$A005，通过键E，可撤消存贮器改变命令，这样就可以显示出监控程序闪动的“短划”。当磁带录音机微音器输入端于键盘/显示模块的相应点连接并出现闪点以后，可按如下顺序实现“录入”功能。放好磁带将录音机置于“录音”方式，按下

P键，闪点消失，进行录带程序。当闪点再出现时，即表示磁带功能已完成，（通过闪点的消失超过30秒），由于记录程序格式规定在每一条记录数据之前，要一个三十秒的字头，关于记录程序的格式在磁带机接口部分还要详述。

（6）由磁带存入存贮器（L）

存入功能，可以使上节所述的用“录带”功能存入磁带时数据复原，使录音机耳机的输出端与键盘/显示模块相应的输入端连接，并在显示器上出现闪点后，按如下顺序，实现存入功能。当需要存入时，磁带置于录入部分的估计起点位置上，使录音机置于“放音”方式，按下L键，闪点将消失，当完成存入功能后，闪点重新出现。当闪点重新出现后，可用“存贮器检查功能”检查A002和A003存贮单元，它们的内容是刚存入存贮器的数据所在区的起始地址，本功能不能恢复终点地址。所以在应用“存入”功能时，A004和A005存贮单元中的数是无效的。

（7）断点插入和取消（V）

由于在程序执行过程中，很难进行分析，在调试时，如能在程序的某些位置设置断点，将是很有用处的。〔这样可以使用户运行一部分程序在继续运行之前，检查结果〕。设置断点的方法是输入十六进制的断点地址并以V键结尾(hhhhV)。这样可以重复五次。“断点进入功能”在E键按下即被撤消，监控程序将保留它们的断点直到被清除。

如果当 hhhhV 指令已进入机器而显示器不显示 hhhh（十六进制数）这是因为已存入了五个断点，这最后一个不起作用。当闪点出现以后，送入一个“V”指令，而不在前面加上十六进制数据，将使现有的断点消去，如一个断点已进入而程序也顺序的执行到这一点。左边4个显示器将显示现在程序计数器的数值（这和断点的地址将是相同的）显示器的右边两位将显示这个存贮单元中所包括的数据，这就是操作码。在这一点上，键G能用来顺序通过其它MPU寄存器，如同“寄存器显示功能”，如果由此断点继续前进，则可按E（得到闪动的点）再按G键，这时候MPU将把堆栈内容重新存入寄存器，并继续用户的程序，到遇到下一个断点或重新按下E键。

（8）跟踪一条指令（N）

“跟踪功能可以每次执行一条指令，步进的通过程序。追踪功能可以在用户程序进入断点或已用E键完成撤消时使用，追踪功能不能由起点开始，因追踪程序不知道开始地址在那里。所以在追踪功能使用之前，必须至少有一个 hhhhV命令，要进入追踪功能先设置希望进入追踪的断点，然后调用 hhhhG执行程序，如需要的话，断点可以设置在程序的最前面。在hhhhG命令之后程序将运行到断点，并且停止，如同前进的一样。显示器显示程序计数器的内容，如按下N键，MPU执行下一条指令并重新停止，显示器显示下一条指令地址（程序计数器内容）和该单元操作码。键N能按顺序的显示其它寄存器的情况，键G现在可以用来追踪需要的无论多少条指令。

当调用“追踪功能”时，所有的断点实际上被消去，离开追踪功能后，必须重新设置，这是追踪功能的一个特点。

1.4 微处理器MPU的基本技术指标：

（1）MPU的基本技术指标

字长：8位（一字节）

输出地址：单独16位，三状态总线方式

输入输出数据：单独双向8位，三状态总线方式。

寻址范围：直接寻址65K字节。

I/O接口地址：RAM、ROM、I/O 接口用相同寻址方式。

电源：+5V单一电源。

输入输出逻辑电平：输入输出都和TTL

兼容。

时钟脉冲：不重叠两相时钟。

时钟频率范围：0.1—1MHZ

集成电路工艺：N沟道硅栅MOS。

指令字长：1、2、3字节。

指令数目：基本指令72种。

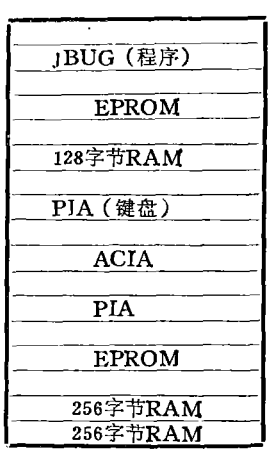
寻址方式：7种（直接，立即、相对、扩充、变址、蕴含、累加器）

内部寄存器：6个（8位的三个，16位的三个）

中断：不能屏蔽的中断(NMI)，可屏蔽的中断(IRQ)，复位(Reset)，软件中断(SWI)

最短指令执行时间：时钟脉冲CP = 1MHz时2μS

第二章　存贮器结构及地址译码



D型机存贮器布局图

上图是这个装置总的存贮器结构，存贮器分布图表在下面以表格的形式表示，在M6800系统里，存贮单元分配是由用于组件的片选择线的组合来决定。

关于地址译码的问题这里需要作一点说明，所有微处理机，不仅仅是6800，选片问题都采用两种技术相结合的方法即线选法和局部译码的方法。

在微处理机系统中，存贮器分为两种，只读存贮器(ROM)和随机存贮器（RAM），前者用于存贮程序和固定的数据表格，后者由MOSRAM的易失性而用于存贮数据和暂时性的信息。

每一种存贮器件必须置于系统存贮布局表中的适当的地方，存贮布局表是地址总线各位寻址的方案表。

系统的每一片RAM或者ROM，都是有一定数量的存贮单元，这意味着要选择任一芯片上的任一存贮单元，要用适当数量的地址线，例如，若存贮芯片包括256个存贮单元，为了能选择256个单元中任意一个单元，需要用8根地址线，除了这8根地址线外，处理机必需做到一次只选择一个器件，因此ROM或者RAM，除地址输入外，还至少有一个选片线，(CS)，这条线上有信号时就允许对器件执行读或写操作。

线选法是将单独的地址线连接到单独的选片输入端，例如，若最高地址位（即A_{15}）连到某个片子的选片输入端，则每当地址的第15位为1时该片就被选中。这就利用了整个存贮单元的一半，现假定ROM通过地址最高位为0来选，RAM用这一位为1来选择，为了寻找每个存贮器件内的特定单元，要将器件的地址线达到相应的地址总线上。

线选法的优点是选择芯片不需要专门的逻辑电路，选择任一新的芯片只要一根专用地址线。正因为如此，所有小的微处理机系统常采用这种方法。例如，应用1片，1K×8ROM1一片512×8的RAM和3个外设芯片，1KROM需10根线（A_0—A_9）作地址选择，一根线（A_{14}）作选片，而RAM用A_0—A_8作地址选择，A_{15}作选片，A_{12}，A_{13}等用于选择其它器件。

然而线选法每用一根单独的地址线，就使可利用的存贮单元减少了一半，因此，要需要的选片线多于可利用的地址线时，就需

要用全译码方法。

全译码方法就是用16根地址线进行选择某一芯片的特定单元。例如，静态随机RAM（128×8），当高位地址线A_8—A_{15}连RAM地址译码器时，低位地址直接连到RAM。因此，RAM中每一个存贮单元仅有一个地址，这种寻址的办法称为完全译码。

全译码选择器件不浪费可利用的地址空间，并可构成邻接存贮器，在那里地址从一个器件传到下一个器件而没有大量空着的存贮区和覆盖的存贮区。这种方法的缺点是需花费译码的代价。DJS—062D型机就是采用线选和局部译码相结合的方法。

在下面的表格里，如K_7(ROM)和K_6，(PROM)等等译码信号是SD404A四取一译码器的输出。SD404A接MPU的VMA、A_{15}，A_{14}和A_{13}信号，例如，当这些信号都是高电位时，对应存贮器地址$EOOO译码器的$\overline{K_7}$(ROM)输出是低电位，这个信号用于JBUG-ROM的片选择线CSI，因此，当MPU的输出地址在$EOOO到$EFFFF范围里时，选择这个组件。按接到ROM地址输入端的MPU地址线A_0—A_9来选择ROM里的特定单元。JBUG-ROM位于存贮器的最高地址部分，注意，从MPU来的A_{12}不接到这个ROM，于是当MPU输出它的再启动和中断矢量地址$FFF8—$FFFF时，也将选择它，在监控程序的$FFF8－$FFFF单元里置放适当的中断矢量地址，从而获得启动和中断能力。

为了在中断和子程序期间暂时存贮内部寄存器，MPU使用了只有$A000—A080 LN6810(128×8)RAM。并且用信号$\overline{K_5}$(stack)来选择它。为了存贮JBUG监控程序使用的标志和暂时数据，MPU也用这个区。这个结构使监控程序需要的RAM和用户的RAM清楚的分开。经由$\overline{K_5}$信号，存贮器底部的$000—$01FF单元分配4个用户RAM，（第一个512字节），这个RAM可以用于小的用户程序，由于采用局部译码和线选相结合的技术来进行选片，这就带来了地址空间复用或称模糊区。

两个信号$\overline{K_1}$和$\overline{K_2}$用来选择存贮器的两个外部8K字节区，$\overline{K_1}$线译码存贮器的第一个8K字节区$2000—$3FFF，$\overline{K_2}$线译码下一个8K字节区（$4000—$5FFF），以下表格是本系统译码信号表。

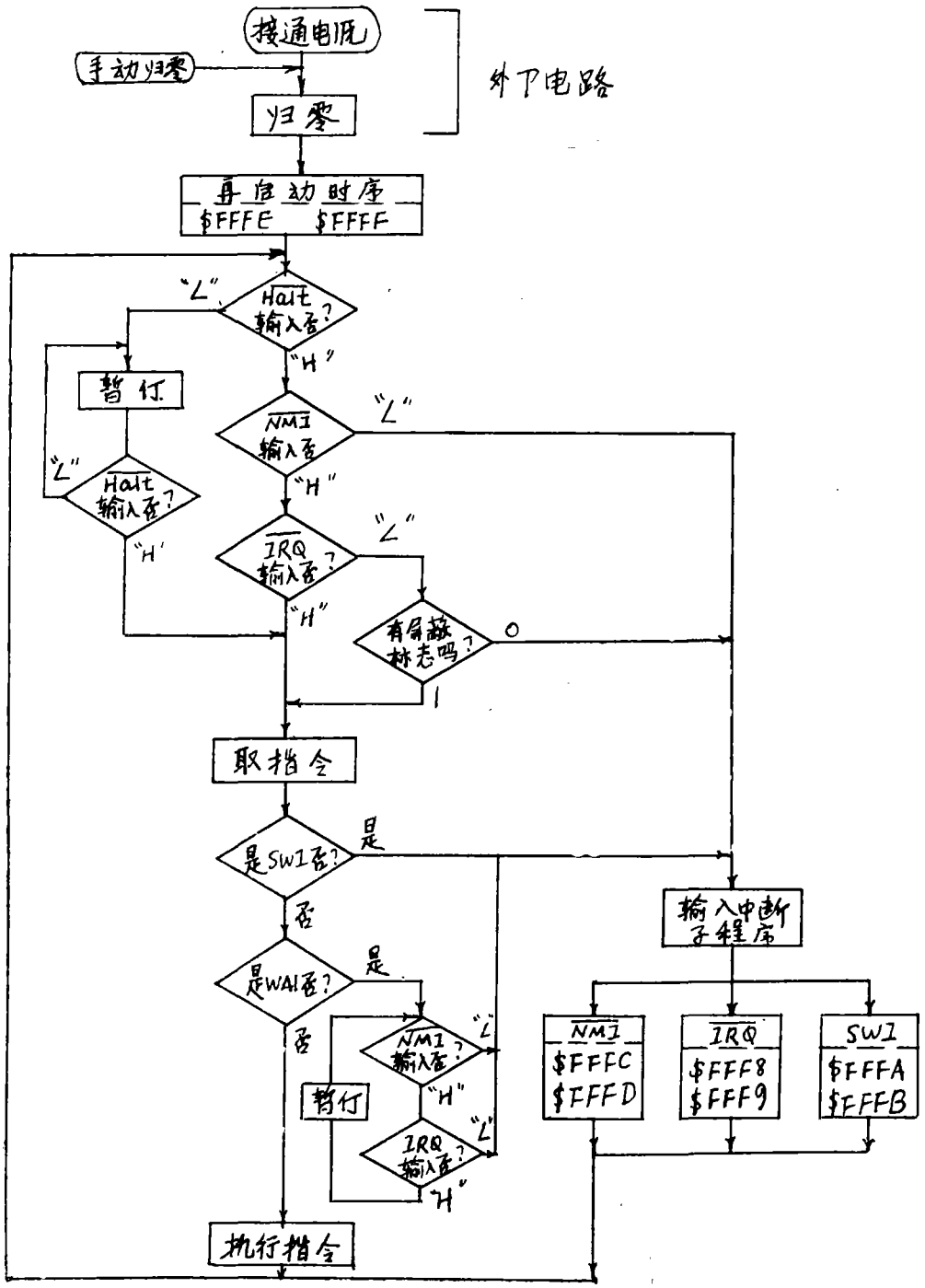
译　码　信　号　表

器　件	地址区间	ϕ_2	R/W	代号	VMA	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
ROM	E000-E3FF	1	1	$\overline{K_7}$	1	1	1	1				×	×	×	×	×	×	×	×	×	×
PROM	C000-C3FF			$\overline{K_6}$	1	1	1	0			+	×	×	×	×	×	×	×	×	×	×
RAM(Stack)	A000-A07F	1	×	$\overline{K_5}$	1	1	0	1	0				0	0	×	×	×	×	×	×	×
PIA	8020—8023	1	×	$\overline{K_4}$	1	1	0	0								1		0 V	0 V	×	×
ACIA	8008—8009	1	×	$\overline{K_4}$	1	1	0	0								0 V		1	0 V		×
PIA	8004—8007	1	×	$\overline{K_4}$	1	1	0	0								0 V		0 V	1	×	×
PROM	6000—7FFF			$\overline{K_3}$	1	0	1	1			+	×	×	×	×	×	×	×	×	×	×
USER	4000—5FFF			$\overline{K_2}$	1	0	1	0													
USER	2000—3FFF			$\overline{K_1}$	1	0	0	1													
RAM(UseR)	0000—007F	1	×	$\overline{K_0}$	1	0	0	0				0	0	0	×	×	×	×	×	×	×
RAM(User)	0080--00FF	1	×	$\overline{K_0}$	1	0	0	0				0	0	1	×	×	×	×	×	×	×
RAM(User)	0100—017F	1	×	$\overline{K_0}$	1	0	0	0				0	1	0	×	×	×	×	×	×	×
RAM(User)	0180—01FF	1	×	$\overline{K_0}$	1	0	0	0				0	1	1	×	×	×	×	×	×	×

×＝用于器件特定单元选址译码。
V＝需要但不用于器件选址译码。
＋＝用于2KX8bit任选RAM译码。

（二）MPU的工作原理

MPU的工作流程图如下：



加了电源以后，或按动了手动开关之后，由外部电路使MPU的归零输入 $\overline{\text{Reset}}$ 经过八个机器周期以上，如果从低电平“L”变到高电平“H”，则完成归零状态。Halt输入如果是低电平“L”，则暂停；如果为高电平“H”，当有中断输入时，就进行中断处理，没有中断输入时，就取指令，指令如果不是WAI(等待中断)就继续执行指令。

（三）MPU的中断功能：

在LN6800中，利用控制输入进行中断有$\overline{\text{NMI}}$和$\overline{\text{IRQ}}$两种，利用程序进行中断的有SWI（软件中断），此外还有$\overline{\text{Reset}}$中断。

$\overline{\text{NMI}}$是不能被屏蔽的中断，$\overline{\text{IRQ}}$是可屏蔽的中断。$\overline{\text{NMI}}$比$\overline{\text{IRQ}}$的优先级要高，同时输入时，$\overline{\text{NMI}}$优先执行。

对$\overline{\text{IRQ}}$输入来说，在中断处理中，如果条件码中的I位为“1”时，则被屏蔽，而放弃中断，对$\overline{\text{NMI}}$来说，则不被屏蔽，而能进入$\overline{\text{NMI}}$处理程序。

若$\overline{\text{IRQ}}$输入为低电平“L”，并且屏蔽位“I”为0时，则把MPU内部寄存器的内容保留在堆栈之中，并转入中断处理程序。保留就是根据堆栈指示器SP所指出的在RAM的堆栈区域之内，存入内部寄存器的内容。这种处理在MPU中是自动进行的，需要占用12个机器周期的时间。

在M6800微处理机中，当内部寄存器需要保留时，在接通电源，MPU启动之后，则在开始保留之前，首先把RAM内作为堆栈使用的地址之中的最大地址放在堆栈指示器中，然后保留现场，堆栈指示器中的数据从指定的RAM地址开始有七个字节，向地址号数减小的方向，依次保留内部寄存器的内容，当堆栈指示器减去7之后，即表示在下一次保留现场时所使用的堆栈的起始地址。

当进入中断处理程序之后，条件码寄存器中的I位即为1，因此在处理过程中，对所加的$\overline{\text{IRQ}}$输入就会进行屏蔽。

本系统$\overline{\text{IRQ}}$中断是供用户使用的，当MPU发出$\overline{\text{IRQ}}$中断矢量地址$FFF8和$FFF9后，立即从ROM的顶部取出$\overline{\text{IRQ}}$中断管理程序的起始地址$E014，从$E014单元开始MPU把RAM中$A000和$A001的内容存入变址寄存器，然后变址转移。这实际上通过监控程序ROM的$\overline{\text{IRQ}}$矢量允许用户自己来安排它的中断管理程序——只要把其中断管理程序的起始地址存入RAM $A000和$A001单元。

关于本系统$\overline{\text{NMI}}$中断，由跟踪命令N引起的$\overline{\text{NMI}}$中断将在后面硬件跟踪电路中叙述。这里略去。现把人为安排的NMI中断简单介绍一下，一但用户给$\overline{\text{NMI}}$输入端一个低电平，MPU就响应$\overline{\text{NMI}}$中断，于是进入以$E019单元为起始地址的NMI中断管理程序，立即将堆栈区用的RAM中的地址$A006和$A007单元的内容送进变址寄存器，然后转到以$A006和$A007的内容为首地址的程序中去，若用户预先于$A006和$A007单元存入用户$\overline{\text{NMI}}$中断管理程序的起始地址，系统即进入中断管理程序中去。

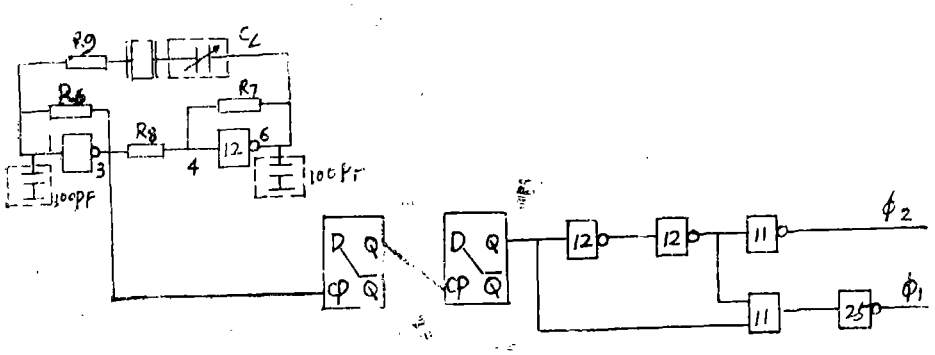
MPU的软件中断(SWI)被用来实现断点，在执行程序过程中遇到软中断指令SWI(3F)，即进入软件中断管理程序——停于该断点，并显示程序计数器的内容，通过G键可检查其它寄存器的内容。

第三章　时钟、跟踪、复位电路及工作原理

（一）时钟电路：

本系统时钟采用612.5KC，这个时钟频率是由2.45MC晶体组成的一个晶体振荡器经D触发器组成的4次计数分频后产生的，因为晶体振荡器较RC振荡等有较高的稳定度，其稳定度可达到10^{-6}，因此，本系统时钟稳定可靠，其电路形式如下：

主振部分中的C_L和双输入与非门输入输



出端的2个 100PF 电容在本电路中未加入，C_L半可变电容是石英晶体所规定的负载电容器，因为本系统允许晶振频率在一定范围内的误差，而不要求百分之百的准确，因此取消了C_L电容。双输入与非门输入端和输出端加的 2 个100PF 接地电容主要为了消除自激振荡，这可视情况而定，如有自激需加入，无自激则取消这两个电容。

主振部分产生的近似于Z、45MC、的振荡频率，经两级D触发器组成的计数分频而产生 612、5KC的时钟频率。计数分频后面的与非门12，11和25是延迟电路，因为本系统要求$\phi_1\phi_2$两相时钟反相并且在高电平时不重叠，这几级与非门就是为了消除重叠而加入的。由此电路产生的ϕ_2时钟，再经 7 级计数分频后产生4800周的音频信号，此4800周的音频信号用于 300 波特速率的串行数据传输的盒式磁带机接口线路。

（二）跟踪电路：

本跟踪电路是为了方便用户调试程序而设计的硬件跟踪电路，此跟踪电路只容许一次执行一条用户指令，即单步的通过用户程序， 执行后将MPU 各寄存器的内容全部显示出来，以便检查。

在调用跟踪命令时，系统必须处于前一次跟踪命令结束，或者是程序运行到一个断点，即系统处于寄存器显示状态。所谓寄存器显示状态即用户寄存器的值压入堆栈之中，监控程序在不断检查新的键闭合及显示再生交替进行，堆栈里保存的用户程序计数器的值指向下一条将被执行的指令，一旦给出跟踪命令：

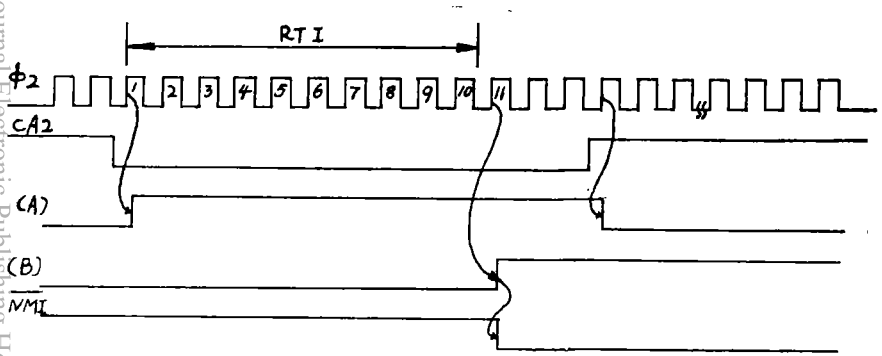
```
即 E1 B A 20 1E  BAR,KEY
                 ⋮
                 DCAN 命令入口地
                 ⋮  址。
   E 1 D A 7F  A01D KEYDCA。
               CLR 清除断点计数单
                   元
   E 1 D D 7C  A018 INC 置跟踪计数
                   标志
   E 1 E 0 86   34   LDAA 设置硬件
                   跟踪
   E 1 E 2 B7   8021 STAA
   E 1 E 5 3B        RTI
```

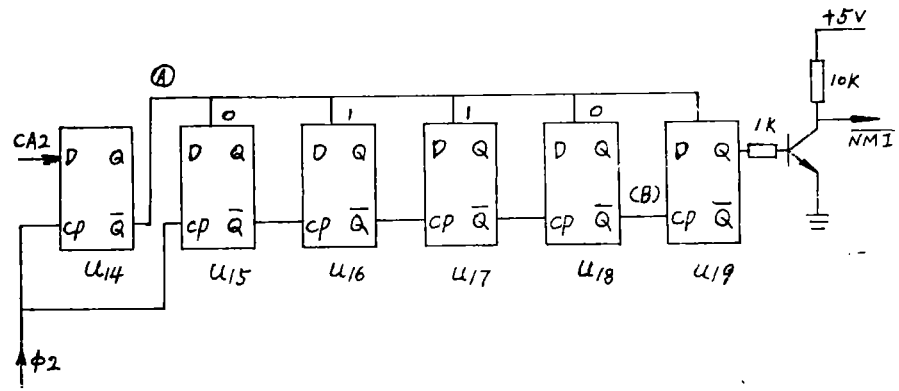
当按下N键时，JBUG 监控 程序进入N命令入口地址并 开始分技转移到 E1DA，于是开始执行这段程序。

请注意这段程序在 E1E0 这个地址中放的是86指令取立即数34，34这个16进制数将8021即A边的控制寄存器置成如下形式：

CRA_5	CRA_4,	CRA_3,	CRA_2,	CRA_1,	CRA_0
1	1	0	0		0

这里CRA_5,CRA_4,CRA_3 置成 110 状态，CRA_5 为 1 把 CA_2 置成输出状态，CRA_4 和 CRA_3的 1， 0 状态使CA_2 处于这种状态即：CA_2 如果处于低电平 “L” 则保持不变，如果处于高电平“H”则MPU读取 PA 输入后，变为低电平 “L” 以后保持不变。其硬件跟踪电路如下：





系统处于平时状态CA_2为高电平，于是U_{14}的$\overline{Q}$为低电平，所以把U_{15}，U_{16}，U_{17}和U_{18}所有计数器全部封锁，这些计数器被置成如下状态即：

U_{15}	U_{16}	U_{17}	U_{18}
0	1	1	0

如前所述：CA_2变低电平是在CA_2作为响应输出时，一旦CA_2变为低电平，CRA_4(控制寄存器第四位)，CRA_3为0时，MPU读取PA输入时变成低电平。这个低电平，使U_{14}的$\overline{Q}$端变成高电平，于是启动U_{15}、U_{16}、U_{17}、U_{18}跟踪计数器，从ϕ_2的第一个脉冲的上沿开始计数ϕ_2的周期，当计数到第十个脉冲时，U_{18}向U_{19}产生一个进位脉冲，这个进位脉冲翻转U_{19}，使Q端变成高电平，于是推动后面的三极管，使三极管导通，于是+5V电压全部降在10K电阻上，向MPU输出一个$\overline{NMI}$。

这个过程从上面跟踪电路和波形上看是U_{15}到U_{18}四个跟踪计数器是在计数ϕ_2的10个周期，实际上监控程序在执行一条RTI返回指令，即把堆栈中的用户寄存器的内容重新弹回MPU各寄存器，这个弹回过程需要十个周期。在第十一个周期的时候，MPU开始执行下条指令，但在这时$\overline{NMI}$中断已经发生，但是MPU的基本特点之一是在执行指令期间突然闯入中断，必须将现行指令执行完才能中断，于是刚刚开始执行的这条指令一旦执行结束，马上产生$\overline{NMI}$中断，这时又要把MPU各寄存器的内容重新压入堆栈，再打一次N键，上述过程重复一遍，就可以跟踪无论多少条用户指令，步进的通过用户程序。

MPU执行完现行指令开始响应非屏蔽中断，进入$\overline{NMI}$中断处理程序：

```
E019  BF   A008
E01C  8D   66
        ⋮
E084  86   3C   LDAA $3C
E086  B7   8021 STAA $8021
E089  B7   8023 STAA $8023
E08C  39        RTS
```

这段程序从EO19开始执行，E019放的BF指令，这条指令就是要把SP_H→A008，SP_L→A009保存堆栈指针后，下条指令转移到E084，请注意，E084中放的是86指令，这就是立即将3C取入累加器A后送入8021和8023。以8021而论。

CRA_5	CRA_4	CRA_3
1	1	1

当控制寄存器予置成如上这种状态CA_2变为响应输出，并且保持高电平不变。这样跟踪计数器U_{14}输入的CA_2又成高电平，$\overline{Q}$又成低电平封锁了$U_{15\text{-}18}$各计数器。以8023而论。

CRB_5	CRB_4	CRB_3	CRB_2	CRB_1	CRB_0
1	1	1	1	0	0

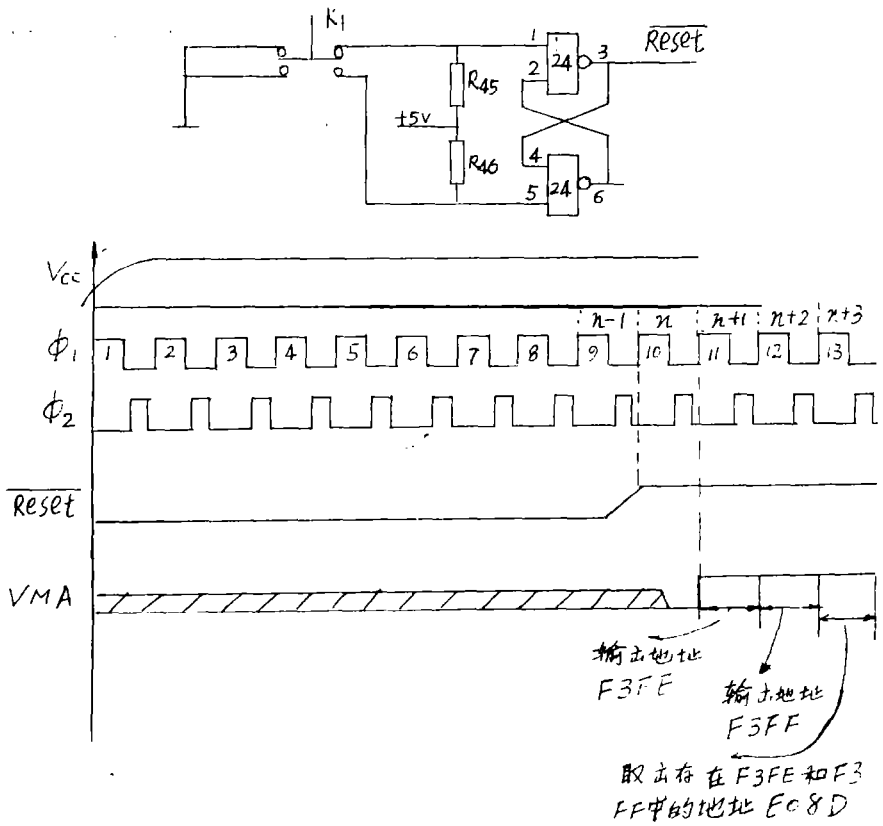
CA_2保持原高电平不变，因为CBR_1，CRB_0均为0，所以关闭CB_1中断请求。最后一条指令是E08C中放的39这条指令，即子程序返回，取出A008和A009中的堆栈指针继续原来的程序。

（三）复位电路

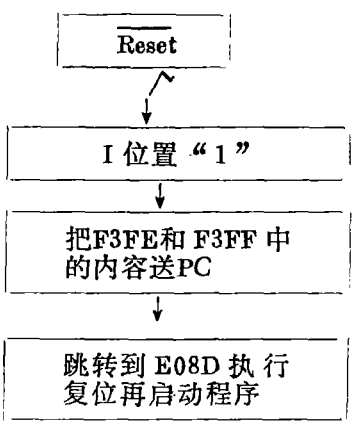
本系统复位启动电路由一双稳态触发器构成，按复位键，使触发器翻转，$\overline{RESET}$端产生一个低电平，加到MPU。

复位中断与$\overline{NMI}$及$\overline{IRQ}$在两方面有所不同，$\overline{Reset}$中断是用来接通电源实现一个予置程序，建立系统的初始条件，如程序计数器，堆栈指示器，PIA模式等的起始内容还可用作系统同步或脱控时的一种再启动方法。因为是用来启动处于断电状态的MPU，故$\overline{Reset}$程序是由E上升沿触发的，同时因为它通常仅用在启动模式故MPU的内容不存放在堆栈中。当$\overline{Reset}$由复位0状态变成1状态后，下一个ϕ_1周期的上升沿，MPU将地址为FFFE及EFFF的两个地址输出到地址总线上，在ϕ_1的第三个周期MPU取出存在FFFE

和FFFF地址中的内容，即向量地址，本系统中FFFE中放的是EO在FFFF中放的是8D（本系统最高地址是E3FF，因此所谓FFFE实际接收的是E3FE，所谓FFFF实际是E3FF），E08D送入PC，于是MPU开始执行复位再启动程序，其复位过程的波形图和电路图如下：



其复位再启动的流程图如下图所示：



第四章　外部接口

DJS—062D型机对外部电路的接口共有两种，一种是并行接口PIA（6820），一种是串行接口ACIA（6850），并行接口PIA共有两片，其地址分别为8004—8007和8020—8023。

8020—8023这个地址上的PIA是供键盘扫描和显示用的接口片子。8004—8007这个地址上的PIA是供用户连接外部设备用的并行接口片子，当然PIA也可以进行串行输出，但这要由软件来实现。

串行接口片子ACIA是在地址8008—8009上，这片ACIA是专供音频盒式磁带机收录用的接口片子，这个芯片连接着音频盒式磁带机接口电路和MPU，它的功能是把从MPU来的并行数据变换成串行数据经接口

电路送至录音机，同时把录音机送来的串行数据经接口电路到 ACIA 变换成并行数据，现把音频盒式磁带机接口的电路，波形和原理叙述如下：

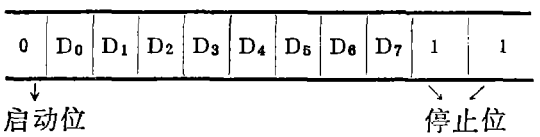
（一）音频盒式磁带机接口：

音频盒式磁带机接口通过异步通讯接口 ACIA 与 MPU 联接，这个线路使用户能以300波特（每秒 30 字符）的基频在普通音频磁带上存取数据，在磁带上存贮数据的格式是：

1. 逻辑“1”用2400周信号的 8 个周期来记录。

2. 逻辑“0”用1200周信号的 4 个周期来记录。

3. 一个被记录的字符由作为启动位的一个“0” 8 个数据位以及作为停止位的两个“1”所组成



4. 字符之间的间隔由 2400 周讯号组成。

5. 每个字符，最低位(LSB) 先传递，最高位后传递。

6. 在磁带的始端(BOT)首先是1024个引导码（大约30秒钟的“FF”码）接着后面记录了字母 B 的 ASCⅡ码。

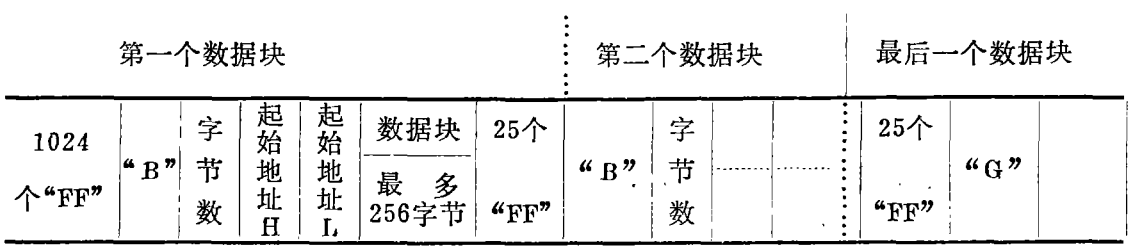
7. 接着 B 之后是以块字节数为内容的一个字节（每个块最多到256个字节）

8. 接着两个字节是数据块的起始地址。

9. 每个数据块最多记录到 256 个字节的数据，随后又有25个“FF”码。

10. 若需要录入磁带的数据字节数超过256个字节，则另起一个数据块，格式相同。

11. 在最后一个数据块结束后的 2^5 个“FF”码后面记录一个字母 G 的 ASCⅡ码

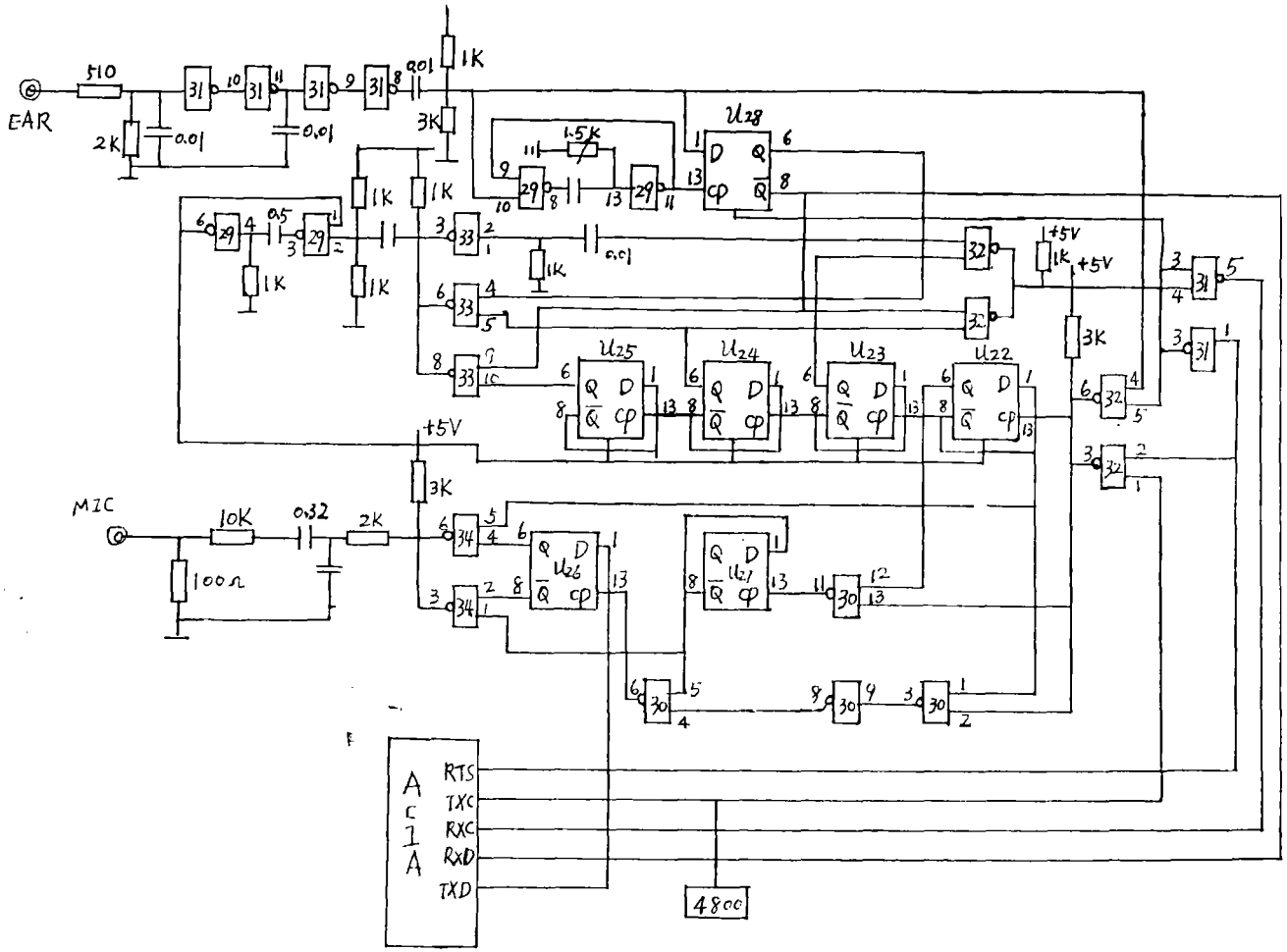


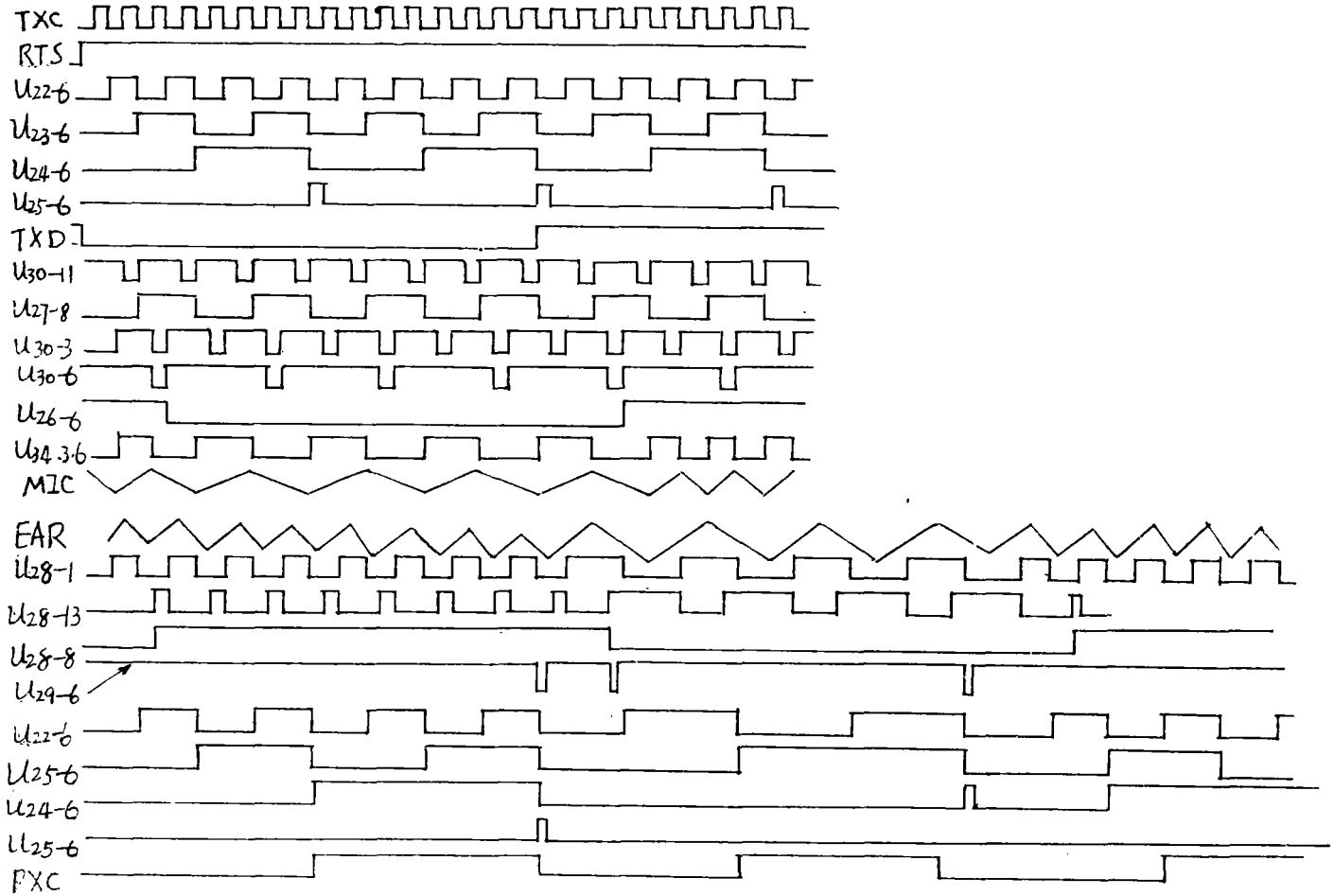
磁带机接口原理图和波形图示于下图

现在叙述一下磁带机接口电路的工作物理过程：

（1）发送时的工作原理：

当按下“P”命令时 ACIA 的发送接收端子RTS置“1”，于是双输入与非门32的 2 脚置“1”，与非门32被打开，4800 周的音频信号加到32的 1 脚，于是 32 的 3 为4800周音频信号，加至U_{22}的CP 端，U_{22}的Q端和D端均为2400 周的信号，U_{22}D 端的2400周信号加到双输入与非门的 5 输入端，双输入与非门30的12脚输入为 2400 信号即U_{22}Q 端的输出信号，双输入与非门30的13脚为4800周信号，双输入与非门30的输出端加到U_{27}的 CP 端，径U_{27}的计数分频$\overline{Q}$端为1200周信号加到双输入与非门30的 5 脚，这个1200周的音频信号又加到双输入与非门34的 1 脚。所以双输入与非门34的 5 脚为2400周信号，34的脚 1 为1200周信号。发送数据TXD直接加到U_{26}的D端，因为这个 TXD 是ACIA用300 波特时钟产生的移位信号，所以其发送速率为300波特，300波特对2400周而





言是8倍关系，300波特对1200周而言是4倍关系，因此当Q端为1时即发送“1”时，为2400周的8个脉冲，当$\overline{Q}$为1时即发送“0”时为1200周的4个脉冲，这样经泸波网络就把ACIA发送的1或0用2400和1200的8个脉冲或4个脉冲记录下来录入磁带机。

（2）接收时的工作原理：

当由录音机往内存送数时，其工作过程较发送时的工作过程要复杂一些。一方面接收部分要把2400信号的8个脉冲和1200信号的4个脉冲变成300波特的1或者0，另一方面要从接收数据中产生300波特的接收时钟。

录音机送来的信号经双输入与非门U_{31}的整形和泸波加到U_{28}的D端，另一方面经由双输入与非门U_{29}的两个门组成的单稳态振荡器送至U_{28}的CP端，D端和CP端的信号波形关系请看磁带机接口部分波形图（U_{28}），在D端转入为8个脉冲的时候，CP端的打入脉冲是在D端输入的8个脉冲的负半周，而在D端输入为4个脉冲的时候CP端的打入脉冲是在其D端4个脉冲的正半周，这样，U_{28}的$\overline{Q}$端就在接收8个脉冲时为“1”，在接收4个脉冲时为“0”，这个“1”或者“0”信号直接为ACIA的接收数据。但是接收移位脉冲的300波特时钟是由分频器产生的，当接收2400周的8个脉冲时，双输入与非门32的4为2400周的信号，双输入与非门32的5是一个“1”电平。为什么是一个“1”电平呢？因为在接时ACIA的接收发送端子RTS为“0”电平，经双输入与非门31的反相成为“1”电平。于是2400周的信号经双输入与非门32反相加到计数分频器U_{22}的CP端，经U_{22}的分频$\overline{Q}$端输出为1200周，再经U_{23}分频，U_{23}的输出$\overline{Q}$端为600周，再经U_{24}分频成为300波特，这个300周的信号就是接收2400周信号时的移位时钟，在接收1200周信号时，其过程与上相同，只是300周的时钟在U_{23}的Q端产生，于是接收数据和接收时钟都已经产生并送至ACIA、ACIA再把串行的数据变成并行数据送往MPU。这样磁带机接口电路在发送时把发送的“1”码和“0”码变成8个脉冲或4个脉冲，并被录音机记录下来。在接时又把8个脉冲和4个脉冲变换成“1”或“0”码，并送往MPU。ACIA完成串并，或并串变换。

MC6854（1.0MHz）
MC68A54（1.5MHz）
MC68B54（2.0MHz）
高级数据链控制器（ADLC）

ADLC高级数据链控制器采用N沟道硅栅E/D MOS工艺操作，按工作频率可划分为三种型号：MC6854(1.0MHz)，MC68A54(1.5MHz)；MC68B54(2.0MHz)；。它们是Motorola公司生产的M6800微型计算机系统的重要接口电路之一。

MC6854 ADLC作为“高级数据通信控制程序”(ADCCP)，“高级数据链控制”(HDLC)和“同步数据链控制”(SDLC)规范，完成复杂的MPU/数据通信链功能。ADLC是改进软件效率的一种关键接口。它是为主站与次站之间提供具有独立的，终

端定时询问的环路结构而设计的数据通信接口。

· M6800兼容的

· 主要特点

· 自动标志检测和同步

· 零插入和删除

· 可扩充地址、控制和逻辑控制字段（任意地）

· 可变字长信息字段—5，6，7或8位可变

· 自动帧校验序列的产生和校验

· 中停检测和发送

· 空闲检测和发送

· 环路模式工作

· 环路反回自测试模式

· NRZ/NRZI模式

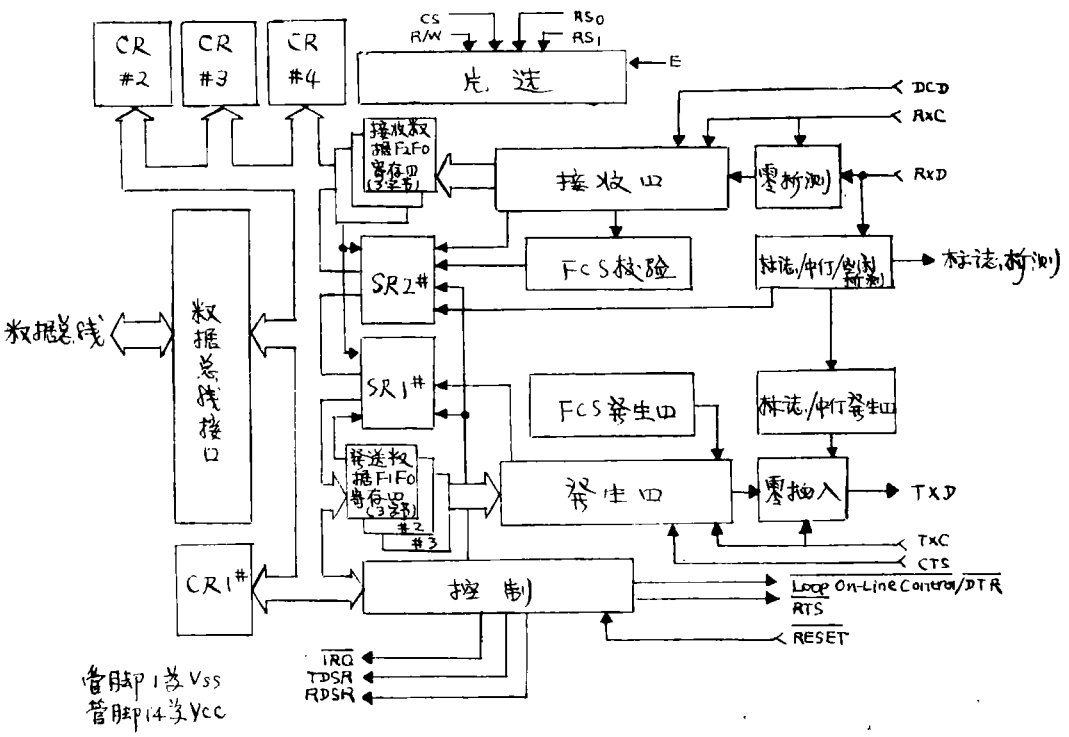
· 对每一个 R_x 和 T_x 具有四重数据缓冲器。

· 优先级状态寄存器（任意的）

· MODEM/DMA/环路接口

· MIL-STD-883，可通用的B类和C类器件

图1、ADLC通用方框图



ADLC性能参数表

1. 器件分类参数

速　　度	器　　件	温　度　范　围
1.0MC	MC6854P,L MC6854CP,CL	0℃～+70℃ −40℃～+85℃
MIL-STD-883B MIL-STD-883C	MC6854BQCS MC6854CQCS	−55℃～+125℃
1.5MC	MC68A54P,L MC68A54CP,CL	0℃～+70℃ −40℃～+85℃
2.0MC	MC68B54P,L	0℃～+70℃

2. 最大额定值

额　　定　　值	符　　号	数　　值	单　　位
电源电压	V_{cc}	−0.3～+7.0	Vdc
输入电压	V_{in}	−0.3～+7.0	Vdc
工作温度范围	T_A	T_L-T_H	℃
MC6854,MC68A54,MC68B54		0～70	
MC6854C,MC68A54C		−40～85	
MC6854BTCS,MC6854CTCS		−55～125	
存贮温度范围	T_{stg}	−65～+150	℃
热　阻　塑　料	θ_{JA}	115	℃/W
陶　瓷		60	

3、 电特性($V_{CC}=5.0V\pm5\%$, $V_{SS}=0$, $T_A=T_L$至T_H，另有注明的除外)

特　　性	符　号	最　小	典　型	最　大	单　位
输入高电压	V_{IH}	$V_{SS}+2.0$			V_{dc}
输入低电压	V_{IL}			$V_{SS}+0.8$	V_{dc}
输入漏电流($V_{in}=0\sim5.25V_{dc}$)($D_0\sim D_7$除以外的所有输入)	I_{In}		1.0	2.5	μA_{dc}
三态（关态）输入电流$D_0\sim D_7$ ($V_{in}=0.4\sim2.4V_{dc}$, $V_{cc}=5.25V_{dc}$)	I_{TSI}		2.0	10	μA_{dc}
输出高电压$D_0\sim D_7$ ($I_{Load}=-205\mu A_{dc}$, $I_{Load}=-100\mu A_{dc}$) 所有其它输入	V_{OH}	$V_{SS}+2.4$ $V_{SS}+2.4$			V_{dc}
输出低电压($I_{Load}=1.6mA_{dc}$)	V_{OL}			$V_{SS}+0.4$	V_{dc}
输出漏电流（关态）$\overline{IRQ}$($V_{OH}=2.4V_{dc}$)	V_{LOH}		1.0	10	μA_{dc}
功　　耗	P_D			850	mW
电容$D_0\sim D_7$ ($V_{in}=0$, $T_A=25℃$, $f=1.0MHz$) 所有其它输入	C_{in}			12.5 7.5	PF
$\overline{IRQ}$ 所有其它输出	C_{out}			5.0 10	PF

（续“电特性”上表）

特　　性	符　号	MC6854		MC68A54		MC68B54		单　位
		最　小	最　大	最　小	最　大	最　小	最　大	
最小时钟脉冲宽度 低 高	PW_{CL} PW_{CH}	700 700		450 450		280 280		nS nS
时钟频率	fc		0.66		1.0		1.5	MHz
接收数据建立时间	t_{RDSU}	250		200		120		nS
接收数据保持时间	t_{RDH}	120		100		60		nS
请求——发送延迟时间	t_{RTS}		680		460		340	nS
发送器时钟——数据延迟时间	t_{TDD}		460		320		250	nS
标志检测延迟时间	t_{PD}		680		460		340	nS
数据终端准备延迟时间	t_{DTR}		680		460		340	nS
连环控制延迟时间	t_{loc}		680		460		340	nS
RDSR延迟时间	t_{RDSR}		540		400		340	nS
TDSR延迟时间	t_{TDSR}		540		400		340	nS
中断请求释放时间	t_{IR}		1.2		0.9		0.7	μS
复位最小脉冲宽度	t_{Res}	1.0		0.65		0.40		μS
输入上升和下降时间（除启动外）（0.8至2.0伏）	$t_{r,tf}$		1.0*		1.0*		1.0*	μS

* 1.0μS或脉宽的10%，无论哪一个都是比较小的。

4、总线定时特性

（$V_{CC}=5.0V\pm5\%$，$V_{SS}=0$，$T_A=0$至70℃，另有注明的除外）

特性	符号	MC6854		MC68A54		MC68B54		单位
		min	max	min	max	min	max	
读								
启动周期时间	t_{cyE}	1.0		0.666		0.50		μs
启动脉冲宽度（高）	PW_{EH}	0.45		0.28		0.22		μs
启动脉冲宽度（低）	PW_{EL}	0.43		0.28		0.21		μs
建立时间 地址和R/W对启动正变换有效	t_{AS}	160		140		70		ns
数据延迟时间	t_{DDR}		320		220		180	ns
数据保持时间	t_H	10		10		10		ns
地址保持时间	t_{AH}	10		10		10		ns
启动输入时的上升和下降时间	t_{Er},t_{Ef}		25		25		25	ns
写								
启动周期时间	t_{cycE}	1.0		0.666		0.50		μs
启动脉冲宽度 高	PW_{EH}	0.45		0.28		0.22		μs
启动脉冲宽度 低	PW_{EL}	0.43		0.28		0.21		μs
置位时间 地址和R/W对启动正变换有效	t_{AS}	160		140		70		ns
数据建立时间	t_{DSW}	195		80		60		ns
数据保持时间	t_H	10		10		10		ns
地址保持时间	t_{AH}	10		10		10		ns
启动输入时的上升和下降时间	t_{Er},t_{Ef}		25		25		25	ns

图 2 、总线定时测试负载

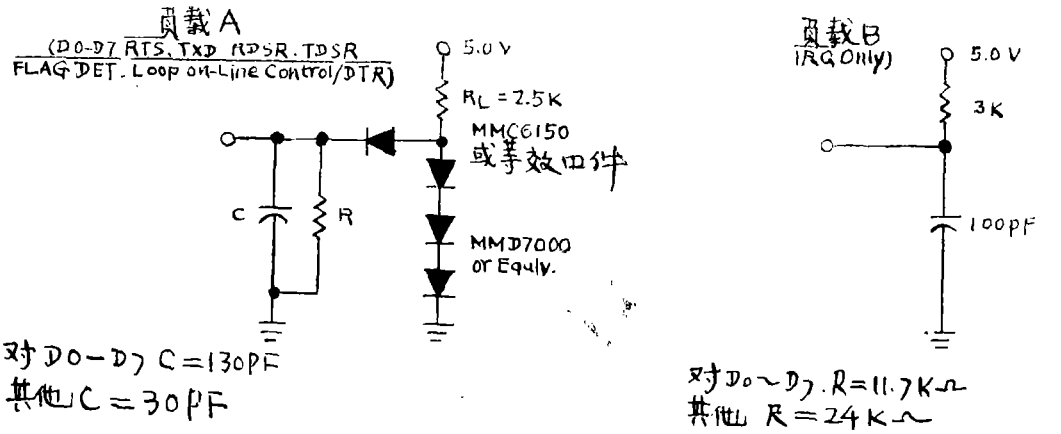


图 3 、接收数据建立/保持，标志检测及环路控制延迟时间

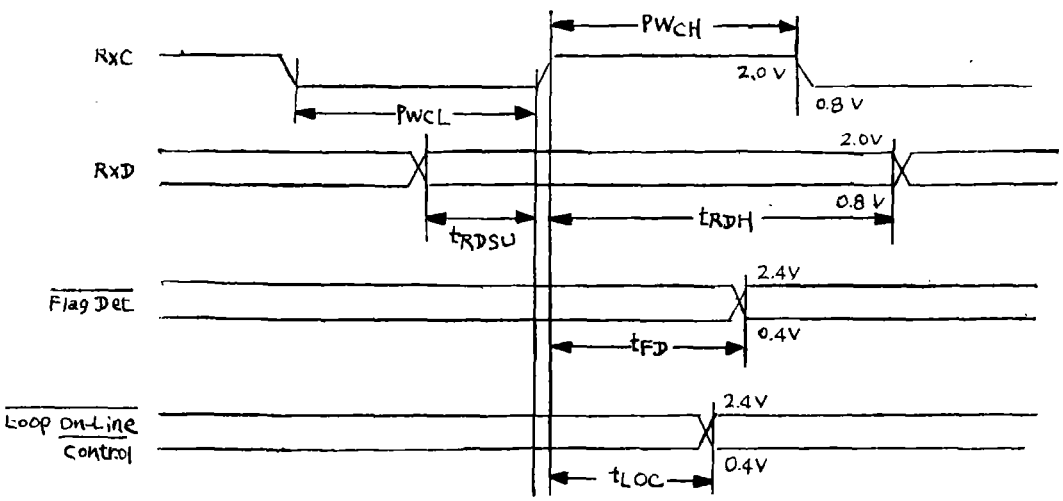


图 4 、发送数据输出延迟及请求发送延迟时间

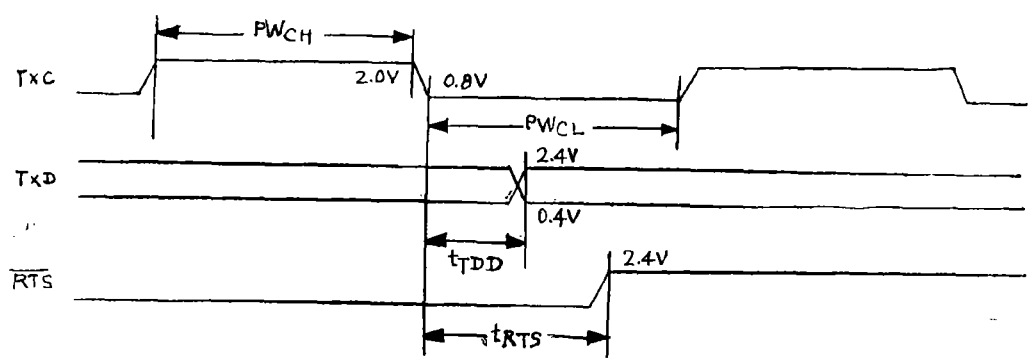


图 5 、TDSR/RDSR延迟，TRQ释放延迟，RTS和DTR延迟时间

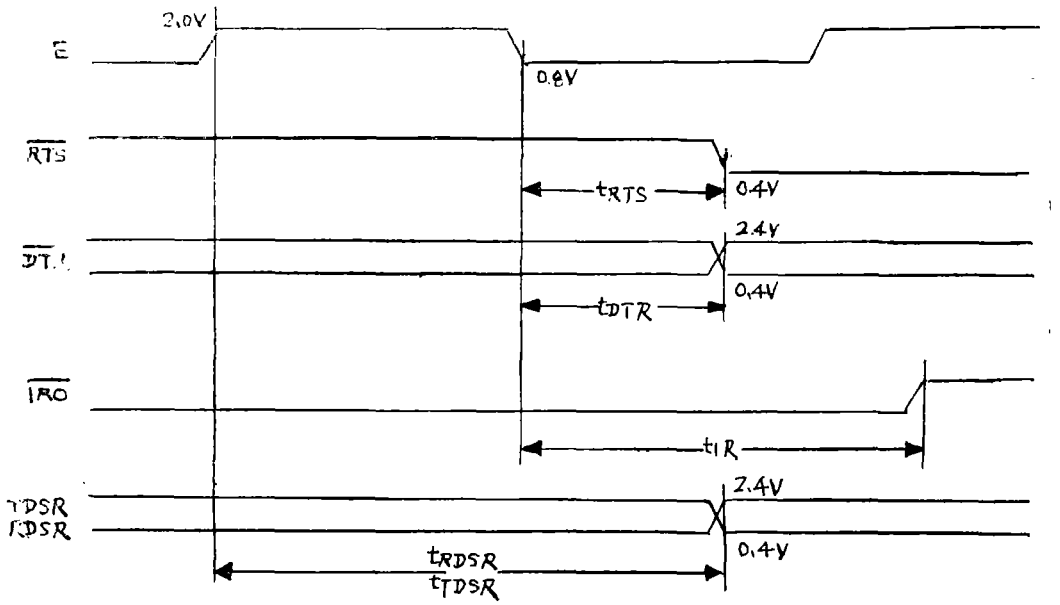
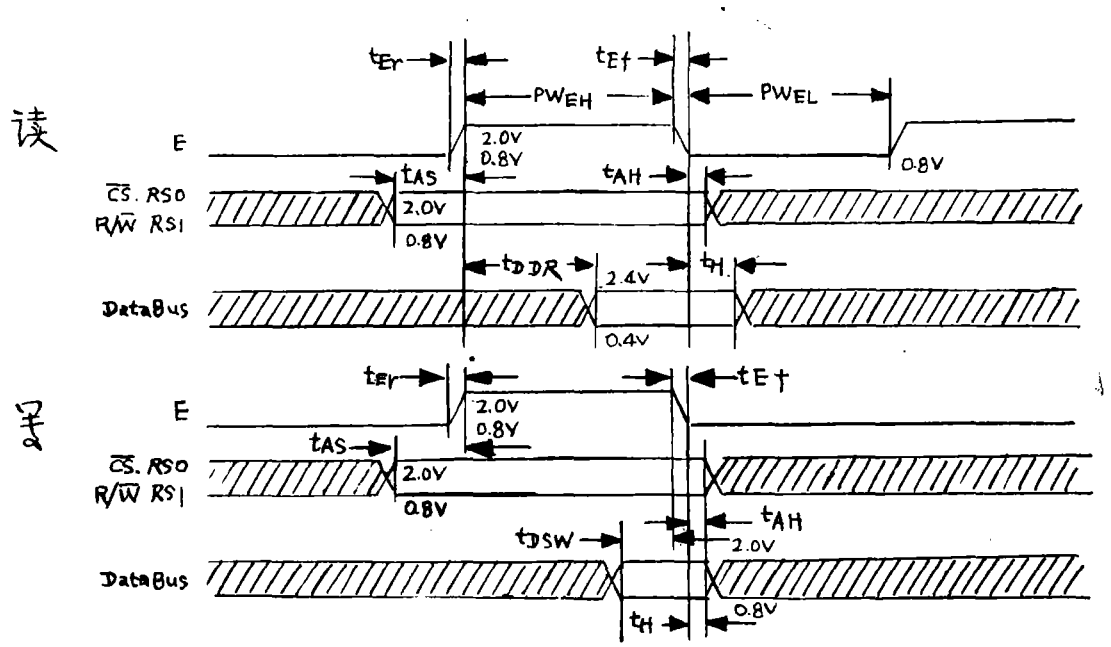


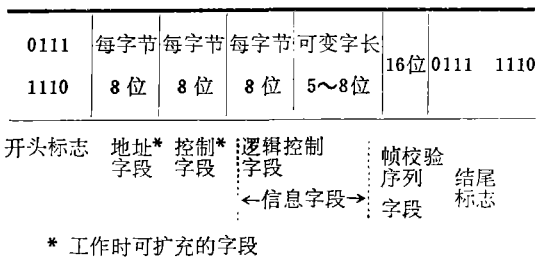
图 6 、总线读/写时间特性



帧 格 式

ADLC发送和接收数据是按被称为帧的格式进行的。所有的帧都是以开头标志起始，以结尾标志结束。在开头标志与结尾标志之间的一个帧包含了地址字段，控制字段，信息字段以及帧校验序列字段。

图7 帧的数据排列格式



标志(F)——标志是一个独特的二进制形式（0111 1110），它提供了帧的边界和帧中每一字段的起始位置。

在ADLC发送器内能产生一个标志符，并自动地把起始标志和结尾标志添加到帧上。如果控制寄存器内的“FF”/“F”控制位被复位，则连读的二个帧可共用一个标志，即第一个帧的结尾标志为第二个帧的开头标志。

ADLC接收器一位一位地搜索和随时辨认标志，并建立带有每个标志的帧同步。

位传送次序——地址字段，控制字段和信息字段的字节，通过数据总线并行地在MPU与ADLC之间传送。从数据总线上第○位(D_0)先串行发送，并将接收到的这一位首先传送到MPU。而帧校验序列字段则先从最高位开始发送和接收的。

地址(A)字段——接着开头标志的八位是A字段。若在控制存器3*中选中“自动地址扩充模式”时，则A字段是可扩充的，并且其八位位组的D_0位就变成了扩充的控制位。当它为“○”时，则其它的八位位组是连续的；而当为“1”时，则地址扩充被终止。所有的“0”地址不能扩充。

控制(C)字段——接着A字段的八位是C字段。若在控制寄存器3*中选中扩充控制字段位时，则C字段可扩充到16位。

信息(I)字段——接着C字段的是I字段，其字长可由控制寄存器4*中的控制位，选择每字节5至8位。它是连续的，直至以FCS字段和结尾标志结束为止。I字段中包含了被传送的“数据”。接收器具有处理“部分”最后字节的能力，此最后字节可以为1至8位间的任一字长。若1字段中的最后字节小于被选中的字长。则接收器将所接收的位向右靠齐，用“○”填充接收器移位寄存器，并将填满的字节传送到R_xFIFO中。不管选择的字节长度如何，ADLC都将传送8位数据到数据总线上。字长为5，6和7位中的无用位都为“○”。

逻辑控制(LC)字段——当控制寄存器3*中的“LC字段选择”位被选中时，ADLC将I字段分成两个子字段，其中之一为LC字段，另一个为“数据”部分。LC字段为8位，在C字段后面。若被选中，则可按八位位组进行扩充，最后一位(D_7)是扩充控制位，当为“1”时，LC字段就扩充一个八位位组。请注意，以下我们提及的I字段仅指它的“数据”部分。

帧校验序列(FCS)字段——在结尾标志之前的16位是FCS字段。它是一种“循环冗余校验字符”(CRCC)。发送器和接收器都使用$X^{16}+X^{12}+X^5+1$多项式。在进行FCS计算之先，发送器和接收器多项式寄存器全都置“1”。计算FCS时，发送器对A字段，C字段，LC字段，I字段中的所有数位进行校算，并以核算余项的补码作为FCS发送。接收器亦作上述相同的核算，并接收FCS字段和FOB8（16进制的）进行比较，当结果与FOB8一致时，“帧有效状态”位在状态寄存器中置位；如不一致，则“错误状态”位置位。

无效帧——凡有效帧在开头和结尾标志之间至少要有A字段，C字段和FCS字段。当ADLC接收到无效帧时，将作以下处理：

（1）当接收到两个标志之间小于25位的短帧时，ADLC将不过问，也不报告到MPU。

（2）当接收到两个标志之间小于32位，或被扩充后的A字段或C字段才多于32位的不完整帧时，若它被传送到 R_xFIFO 中，则由FCS/IF错误状态位在帧的末尾指示出接收的为无效帧。

（3）当接收到中停或DCD失效而出故障的帧也是无效帧。如何处理，请参看“中停”和“DCD状态”位。

零插入和零删除——由ADLC自动地完成。在发送一个帧（包括A,C,LC,I和FCS字段）内连续5个“1”以后，通过发送器插入1个二进制“〇”；在接收一个帧连续5个“1”以后，接收器就删除一个二进制“〇”。

中停——过早地终止或不到终点而中途停止数据链的称为“中停”。在发送器连续发送多于8个“1”时，控制寄存器4*内的 T_x 中停控制位置“1”以后，立即中停一个帧（在此时刻，通过 T_x 中停控制位将 T_xFIFO 也清零）。在中停发生后，如4*内的“中停扩充控制”位置“1”，则中停可以扩充到至少连续16个“1”。接收到连续多于7个“1”时，它作为接收器产生的中停解释。接收器对接收中停的回答如下：

（1）“超出帧”条件下的中停——在空闲或满载(time fill)期间的中停无意义。当连续接收多于15个“1”以后，中停信号就消失，此时接收的空闲状态为“1”。

（2）在开头标志以后接收少于26位后“入帧”中停——此条件下中停帧的任一字段也不会被传送到MPU。ADLC清除在FIFO中的中停帧数据和标志同步。

（3）在开头标志以后接收超过26位后“入帧”中停——此条件下中停帧的某些字段可以传送至数据总线。中停状态存贮在接收状态寄存器中。ADLC清除中停帧数据以及同步标志。

空闲和满载——发送器处于“超出帧”为空闲状态。此时根据标志/标记空闲控制位，空闲状态下接收的不是选择连续标志（满载）就是选择标记空闲。当接收器接收多于15个连续“1”时，接收空闲状态位置“1”，并产生中断。

操　作

起　始——电源接通后，ADLC经由$\overline{RESET}$输入复位，并内部锁住复位状态，以防止错误输出传送。四个控制寄存器必须在复位条件释放之先编好程序。复位条件的释放是经由软件将“0”写入到 R_xRS 接收控制位（接收）和/或 T_xRS 发送控制位（发送）来完成的。$\overline{RESET}$达到高电平后，复位条件必须释放。

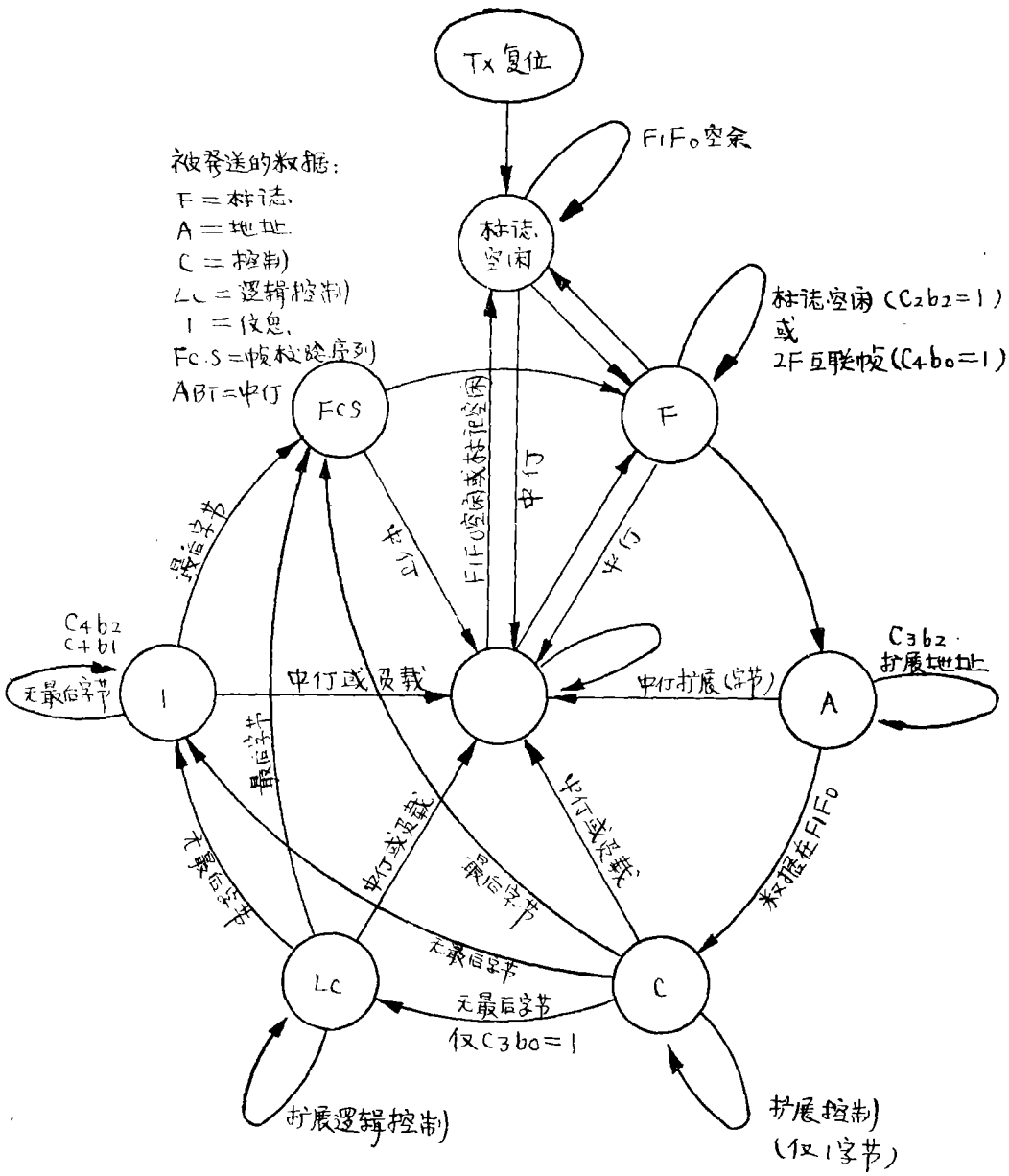
在操作中的任一时刻，只要将“1”写入到 R_xRS 控制位或 T_xRS 控制位，则相应的接收器或发送器就会产生复位状态。

发送器操作——在复位状态，T_xFIFO 不能予先加载。复位释放后，标志/标记空闲控制位或选择标记空闲态（无效空闲态）或选择标志“定时加载”态（有效空闲态）。有效或无效空闲态将持续至数据装入 T_xFIFO。

T_xFIFO 的有效性是在“2字节/1字节”控制位控制下通过TDRA状态位来表示的。在 T_xRS 位或 $\overline{CTS}$ 输入为高电平时，TDRA状态被禁止。在TDRA变为高电平时，选中1字节模式，则可发送FIFO的1字节数据，选中2字节模式，则可发送连续的2字节数据。

在“帧连续”地址时，A字段的第一字节将写入 T_xFIFO，然后再自动地开始帧发送。若发送器为标记空闲态，则在2或3个

图8A ADLC 发送器状态流程图
（c_ib_i相对于控制寄存器位）

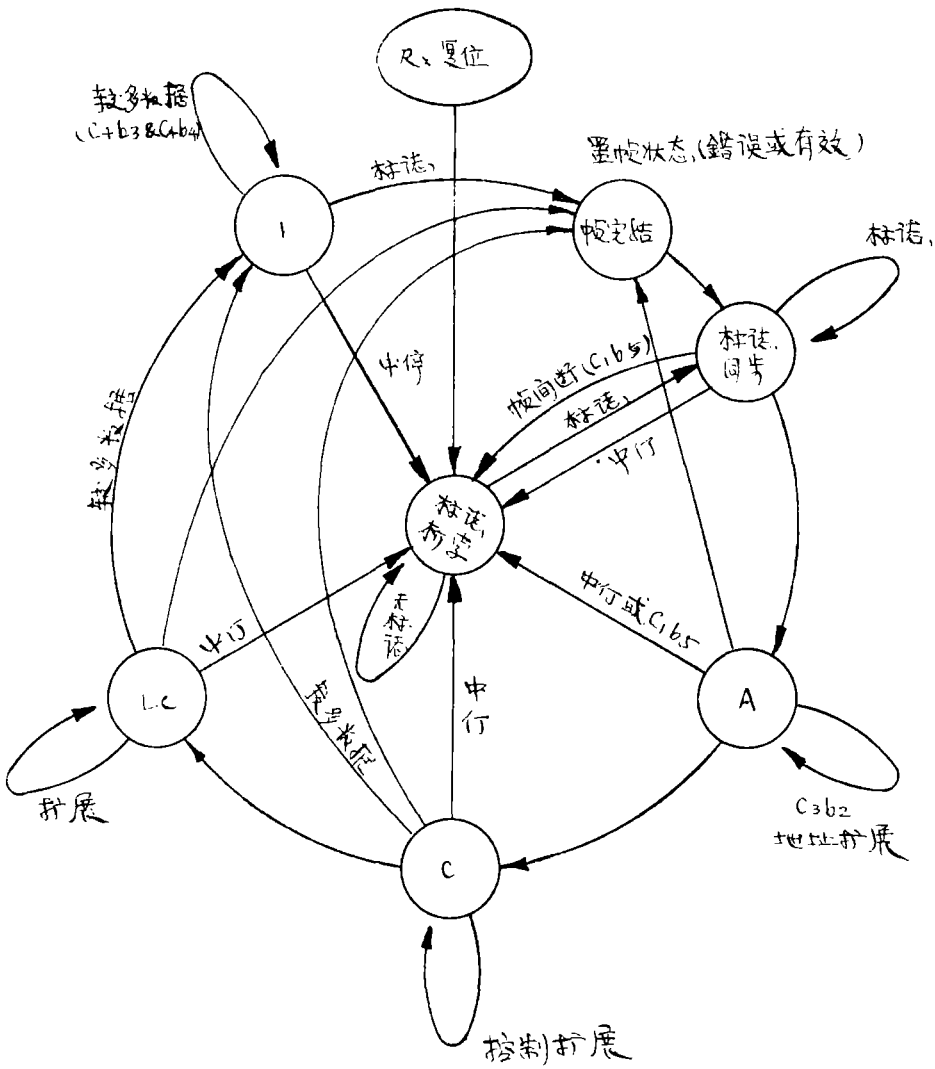


发送器时钟周期内，地址发送就产生一个开头标志，若发送器为满载状态，则假定通常发送的满载标志是开头标志，随后为A字段。

在“帧连续”地址，只要数字被写入T_xFIFO，则帧连续延伸。ADLC内将保持帧内字段顺序不变。

帧的终止有两种方法。一种方法是从软件观点看，最有效的是将最后数据字符写入“发送FIFO帧终止”地址（RS_1，RS_0＝11），而不是发送(FIFO帧连续”地址（RS_1，RS_0＝10）。另一方法是将最后数据写进T_xFIFO

图8B，ADLC接收器状态流程图



“帧连续”地址后，紧接着将“发送最后数据”控制位置“1”。此两种方法中的任何一种都能引起最后字符的发送，并且，FCS字段自动地与结尾标志一起附加上。若TDRA为高电平，则在此帧数据之后可立即将新帧数据装入T_xFIFO。结尾标志可作下一帧的开头标志用，或用来将发送的开头标志和结尾标志分开。若新帧不准备发送，ADLC将自动地发送有效或无效空闲态。

若T_xFIFO在帧发送的任一时刻变为空态（FIFO在下一个数据的最后一半到代码的最末尾的发送时间，没有数据向发送器移位寄存器发送），则将产生欠载运行(Underrun)，并且发送器通过发送一个中停，使它自动终止帧的发送。

在“发送中停”控制位置“1”的任一时间内，发送器立即中停帧，并对T_xFIFO清零。若在“中停扩充控制”位置“1”时，则发送一个空闲（至少16个连续“1”）。

$\overline{CTS}$输入和$\overline{RTS}$输出是为MODEM或其它硬件接口提供的。

TDRA/FC状态位根据帧完成产生中断。

接收器操作——通过接收数据(R_XD)和接收时钟(R_XC)输入，为ADLC接收器部分提供了数据和予同步时钟。数据是一串连续的二进制位，除引起中停，标志或空闲状态外，最多可连续产生5个“1”。接收器连续（一位一位）地检索标志和中停。

当检测到标志时，接收器对标志同步建立帧同步。若接收到一连串的标志，则接收器对每一标志都要进行再同步。

如果帧在内部缓冲器周期完结之前（开头标志以后的帧数据小于25位））就终止了，这样的帧则可简单地不予理采，可视为无效帧。在满载期间数据输入端(R_XD)的噪声是形成这种无效帧的原因。

一旦同步实现，而且其内部缓冲器时间（24位时间）已经完成，则数据将自动地传送到R_X数据FIFO中。“R_X数据”FIFO按E时钟进行动作，而E可使接收的数据通过FIFO移动到最后空着的寄存器地址。对1字节传输模式，出现在最后的寄存器（即3*寄存器）中的数据，表示为“接收数据有效状态”位(RDA)。当最后两个FIFO寄存器地址（2*和3*寄存器）被填满时，2字节传送模式使RDA状态位所表示的数据是有效的。如果FIFO中的数据字符是一个地址八位字组。则状态寄存器将显示出地址现行状态条件。接收数据FIFO中有效的数据引起一个中断的开始（假定接收器中断被许可，即RIE=“1”）。MPU将作为中断结果或转入终端设备定时询问程序来读ADLC状态寄存器。RDA或地址现行状态将表明接收器数据是有效的。同时，MPU将接着就读R_X数据FIFO寄存器。随后中断和状态位就自动复位。如果接收多于一个字符。而存放在接收数据FIFO中，则随后的E时钟将使FIFO修正，并且RDA状态位和中断将重新置位。在二字节传送模式中，两种数据字节都可在连续的E周期中读出，现行地址仅提供1字节传送。

在接收的帧中，每种字段的顺序由ADLC自动来处理。

当接收到结尾标志时，帧就终止。放在结尾标志之先的16位是作为FCS来看待的，并且它不传送到MPU。出现在接收缓冲寄存器的最有效字节部分里不管是什么数据，都认为是对的，并传送到R_XFIFO。在“R_XFIFO寄存器”一节中曾解释过“帧边界指示字”能自动地插入到R_X FIFO中。当帧的最后字节出现在R_X FIFO的最后位置时，帧边界指示字对“帧有效状态”位（当帧完成时无错误）或“FCS/IF错误状态”位（当帧完成时有错误）置位。只要此状态位置“1”，就禁止数据从R_X FIFO的第二地址传送到R_X FIFO的最后位地址。

环路模式操作——ADLC在环路模式不仅要完成上面所介绍的数据帧发送和接收，而且还要有得到和放弃环路控制的其它功能。图9a是描述环路模式操作的一种结构。整个系统由一个主站和几个次站构成。环路经常在主站控制下，当主站要接收数据时，它发送一个终端设备定时询问程序，并允许帧发送到环路上的次站。每个次站都是串联连接的，并附加一个延迟位到环路上。图中的次站A从主站经由“R_X数据输入”接收数据，延迟一位，并经由“T_X数据输出”将它发送到次站B。次站B，C和D亦按同样方式操作。

次站联入环路（其发送器输出连接到环路上），主动加入环路（开始传送它本身次站的数据到环路）和脱开环路（其发送器输出与环路断开）这些方式必须遵循某些规定的规则。即数据总是按相同的路线流动，而且其次站终端接收的次序是由硬件结构来决

图9a，典型的环路结构

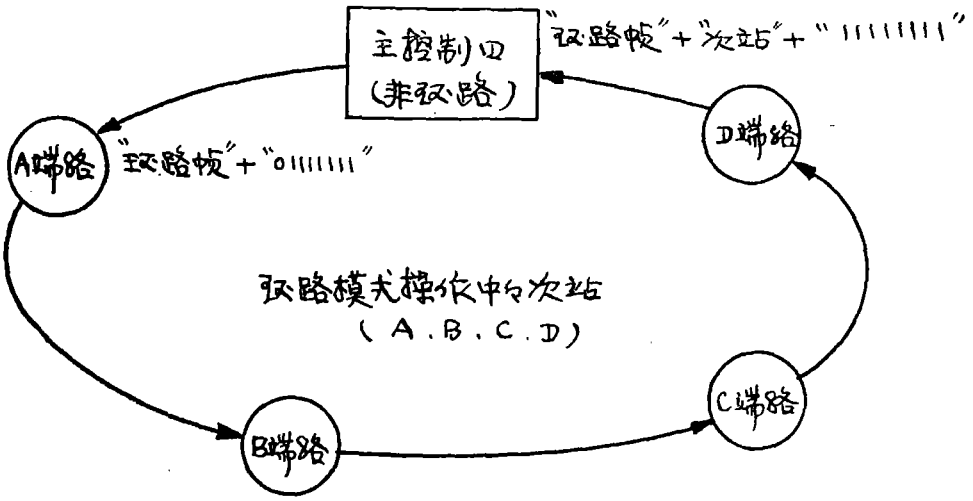
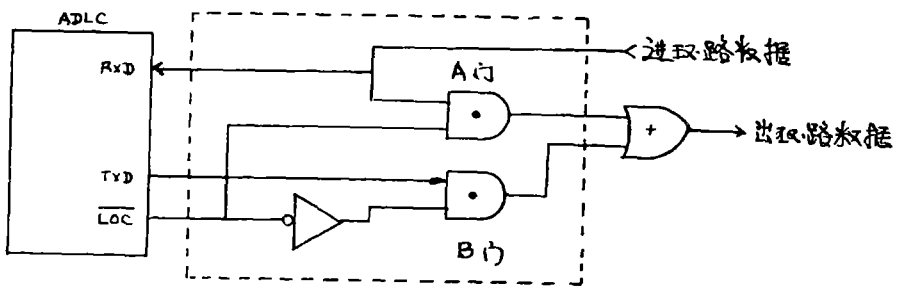


图9b，外部环路逻辑实例



定的，环路的数据不允许到其它次站出环路。主站控制器通过环路按排时间延迟。如果超过了n+1位时间，表明环路出现故障。n为环路上的次站终端数。接着来自主站终端定时询问帧（响应次站请求）的结尾标志，通过发送一个“向前”（Go Ahead）信号，将其发送到次站。“向前”信号是一个二进制“0”和七个二进制“1”。接着“向前”信号的是标记空闲。主站可以中断标记空闲来中停，此时次站将立即停止发送，并将控制权交还给主站。当次站完成其帧传送时，发送结尾标志，接着为全“1”。主站检测最终的 0111 1111……，控制权还给主站。若环路最末的次站（D次站）需要在最前面的次站A以后插入信息，则到次站D的“向前”信号是从次站A的结尾标志最后为“0”，以后又全为“1”。

ADLC作主站将按非环路的全双工模式操作；作各次站将以环路模式操作。ADLC能识别数据自动流向入环路/出环路和某一次站插入数据的必要程序。环路操作过程扼要地列在表1中。

表1. 模路模式操作概要

状　　态	Rx　部　分	Tx　部　分	环路状态位
离开环路	Rx 部分从环路接收数据，接收连续七个“1”后，检索联入环路（环路控制位置位）	不动作 1）NRZ模式：Tx 数据输出保持“高”（标记） 2）NRZI模式：Tx数据输出使Rx数据输入延迟1位。	“0”
联入环路	1）当“联入环路动作”位置位，Rx部分达到 0111 1111，变成联入环路的动作终端。 2）当“联入环路控制”位置位，Rx部分接收连续8个“1”后（检索）脱离环路。	不动作 1）NRZ模式：Tx数据输出反映延迟1位时间的Tx数据输入状态。 2）NRZI模式：Tx 数据输出反映延迟2位时间的Rx数据检入状态。	“1”
动　　作	Rx部分在Rx数据输入检索标志接收标志使$\overline{FD}$ 输出变低。若RIE和FDSE控制位置位，则产生IRQ	Tx数据出现在ADLC内，直至“联入环路动作“位复位以及标志或中停完结，然后转入联入环路状态。	“0”

（1）联入环路——接通ADLC 电源以后，终端站将脱机，因此，首要任务是要将其变为环路上的动作终端。ADLC 必须经由如图9b所示的外部开关连接到环路上。硬件复位后，ADLC 的“$\overline{LOC}/\overline{DTR}$ 输出” 将为高电平，并联入环路接收数据重复通过A门而到达脱出环路次站。“环路模式/非环路模式控制”位（控制寄存器3*中的5位）必须置“1”，以使 ADLC 处于环路模式。环路模式操作可用“状态寄存器1*中的”“环路状态”位来进行监视，在通电和复位后，此状态位为“0”。当 ADLC 接收七个连续“1”时，$\overline{LOC}/\overline{DTR}$ 输出变为低电平，图9b中的A门截止，B门导通，并且 ADLC 的 Tx 数据输出端连到脱出环路的次站。当入环路数据通过ADLC 进行循环时，数据中插入一位延迟（对 NRZI 模式，则有2位延迟）。此时，ADLC 处于联机，并且状态寄存器1*中的环路状态位是“1”。

（2）定期询问后进行动作——接收器部分用一通用的或编址的终端设备定时询问指令对入环路数据进行监视，而 TxFIFO 应装有数据，以便当检测到零以后七个1(0111 1111) 的“向前”信息时，发送就能立即开始。当检测到终端设备定时询问帧时”，“联入环路动作控制”位必须置“1”（控制寄存器中的6位）。当接收器检测到“向前”信号时，ADLC 将自动地把第七个“1”变成“0”，结果从图9b中B门出来的重复序列现在变为开头标志（0111 1110）。当ADLC达到联入环路动作时，状态寄在器 1* 中的环路状态位为“0”。接收器检索的标志表明，主站中断了通常的操作

（3）联入环路不动作——当帧发送完成后，环路将自动释放，次站正好在环路的1位延迟时恢复原状。若“联入环路动作控制”位通过软件不复位，将不释放入环路的数据，即联入环路不动作，则必须利用“T_x 中停”指令。在“向前”字符变为标志之前，T_xFIFO没有予先装入数据，ADLC或者发送标志直至装入数据；或者变成欠载运行状态，并发送一个中停。之后“联入环路动作控制”位自动复位，ADLC 将转回到循环模式（T_xD = 延迟的R_xD）。

（4）脱离环路——ADLC 可以脱离环路（脱机），这相似于联机的方式。当“环路联机控制”位复位时，则在 $\overline{LOC}/\overline{DTR}$ 输出转至高电平（无效状态）之前，ADLC接收器部分寻找 8 个连续的“1”。图9b中A门导通，B门关闭，以使环路持久保持无故

障，环路状态位将是脱机状态（即为逻辑0）。

输入/输出特性

ADLC所有的输入为高阻抗，与TTL输入电平兼容。其所有的输出与标准的TTL兼容。然而中断请求（$\overline{IRQ}$）是一个开漏输出（无内部推挽输出）。

“ADLC输入/输出引线端如图10所示。除电源电压和地线端以外，其余26条引线端分别作为MPU接口连线，DMA接口连线，发送器和接收器时钟和数据线，以及外设/调制解调器控制线。下面简单地对各引线功能分别作以介绍。

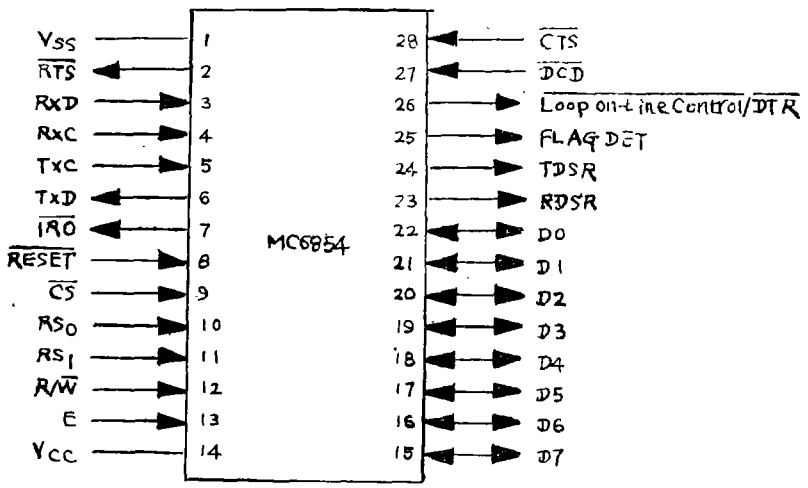


图10 MC6854(ADLC)引线图

MPU接口线

D_0～D_7双向据数总线——这些I/O数据总线允许数据在ADLC与系统总线之间传送。它们是一些三态驱动器，除当MPU进行ADLC读操作以外，平时都保持高阻抗状态（即关态）。

E 启动时钟——E对地址输入（$\overline{CS}$、RSO和RSI）以及R/W输入起作用，并能启动数据在数据总线上传送。E也可通过T_xFIFO与R_xFIFO移动数据。E时钟同MC6800 MPU的系统时钟一样，具有固有的频率。

$\overline{CS}$ 片选——只有当$\overline{CS}$输入为低电平，E时钟输入为高电平（即$E\cdot\overline{CS}$）时，ADLC才能实现读或写的操作。

RS_0、RS_1 寄存器选择——当通过（$E\cdot\overline{CS}$）寄存器选择输入被启动时，则它们与读/写输入和地址控制位（控制寄存器1#的0位）一道选择内部寄存器。寄存器寻址由表2确定。

R/$\overline{W}$ 读/写控制线——当R/$\overline{W}$通过（$E\cdot\overline{CS}$）被启动时，它控制数据总线上数据流动的方向。当R/$\overline{W}$为高电平时，I/O缓冲器的作用象一个输出驱动器；当R/$\overline{W}$为低电平时，其作用为一个输入缓冲器。它也可以选择ADLC内部的只读和只写寄存器。

$\overline{RESET}$ 复位输入——复位输入提供了从硬件对ADLC进行复位。在“低电平状态时，复位输入产生下列情形：

R_x复位和T_x复位置“1”，使得接收器和发送器这两部分都抑止在复位条件。

* 复位下列控制位：发送中停，RST，环路模式以及环路联机TSDTR。

* 清除各状态寄存器所有存贮的状态条件。

* 输出：$\overline{RTS}$和$\overline{LOC/DTR}$变高。T_xD变成标记状态（即所有的1都被发送）。

当$\overline{复位}$转向“高”电平（无效态）时，发送器和接收器部分将保持在复位状态，直至T_x复位和R_x复位在软件控制下经过数据总线清零。当$\overline{复位}$为”低”电平时，受$\overline{复位}$影响的控制寄存器位不能变化。

$\overline{IRQ}$ 中断响应输出——如果存在一个中断状态，而且适当的中断启动已经建立，则$\overline{IRQ}$将为低电平。只要引起中断的原因存在，启动已经置1，则中断仍然保持。

发送器和接收器的时钟和数据

T_xC 发送器时钟输入——发送器在T_xC时钟输入的负跳变时移动数据，当选择了“环路模式”或“测试模式”时，R_xC与R_xC有相同的频率和相位。发送器发送数据的速率将不超过E频率。

R_xC 接收器时钟输入——接收器在T_xC时钟正跳变时，对数据取样。RxC与外部的接收数据同步。

T_xD 发送数据输出——来自发送器的串行数据是按NRZ或NRZI，数据格式编码的。

R_xD 接收器数据输入——ADLC接收到的串行数据可按NRZ或NRZI数据格式编码。接收器数据速率不超过E频率。如果在帧末尾接收一部分字节是可能的话，则接收器的最大数据速率由下式来表示：

$$f_{RXC} \leqslant \frac{1}{2t_E + 300ns}$$

式中：t_E是E的周期时间。

ns是毫微秒。

外设/调制解调器控制

$\overline{RTS}$ 请求发送输出——请求发送输出是受与发送器部分状态连在一起的“请求发送控制”位控制的。当RTS位变高电平时，$\overline{RTS}$输出被迫变为低电平。当RTS位转回低电平时，$\overline{RTS}$输出仍维持低电平直至帧结束‘而且对新帧T_xFIFO中没有其它数据。当一个标志完成后，$\overline{RTS}$的正跳变引起一个中停，或者在标记空闲态时RTS控制位被复位。$\overline{RESET}$输入为低电平时，$\overline{RTS}$输出变高。

$\overline{CTS}$ 清零发送输入——它为TDRA状态位及其有关的中断提供一种实时禁止。$\overline{CTS}$正跳变时将CTS信息存入ADLC中。在清除T_x状态位或发送器复位时，通过写“1”来清除存贮的CTS信息及其与此相关的IRQ。

$\overline{DCD}$ 数据载波检测输入——$\overline{DCD}$输入为接收器部分提供一种实时禁止。它为高电平时，复位和禁止接收器的寄存器，但前帧在R_xFIFO中的数据不受妨碍。$\overline{DCD}$正跳变时信息存入ADLC中。清除R_x状态位或复位R_x来清除存贮的$\overline{DCD}$信息及其与此有关的IRQ。

$\overline{LOC/DTR}$ 环路联机控制/数据终端准备输出——它在非环路模式中作$\overline{DTR}$输出；在环路模式中作$\overline{LOC}$输出。当在执行$\overline{DTR}$功能时，由“$\overline{LOC/DTR}$控制”位来接通和断开。$\overline{LOC/DTR}$为高电平时$\overline{DTR}$为低电平。当执行$\overline{LOC}$功能时，由$\overline{LOC/DTR}$提供控制外部环路接口硬件去联机/脱机的方式。当LOC/DTR控制位置“1”，并且环路多于七位时间已“空闲”（0111 1111……）时，$\overline{LOC/DTR}$变为低电平（联机）；当LOC/DTR为低，并且环路多于八位时间已“空闲”时，$\overline{LOC/DTR}$变为高电平（脱机）。$\overline{RESET}$输入变为低电平使$\overline{LOC/DTR}$输出为高电平。

$\overline{FD}$ 标志检测输出——这一输出表明环路模式操作时标志的接收和启动一外部暂停计数器。由接收时作取样，从标志字符最后一位开始的一位时间，$\overline{FD}$输出变为低电平。

直接存贮器存取(DMA)接口

RDSR “接收器数据服务请求”输出——RDSR主要用于DMA模式操作以及表示R_x FIFO请求服务。若选择优先级状态模

式，则当任何其它接收器状态条件存在时，RDSR被禁止。当读R_XFIFO 时，RDSR 变为低电平。

TDSR 发送器数据服务请求输出——它主要用于DMA模式操作以及表示T_XFIFO 请求服务。当 T_X FIFO 被装入 时， TDSR 变为低电平。由 “T_XR_S 控制” 位置 “1”，$\overline{RESET}$为低电平或$\overline{CTS}$ 为高电平，TDSR被禁止。若使用优先级状态 模 式，则 “T_X欠载运行” 也禁止TDSR。除 F/C 模式以外，TDSR都影响TDRA状态位。

ADLC中的寄存器

ADLC 中有八个寄 存器， 它 们可通过MPU 数据和地址总线访问。 根据信息流的方向可以确寄存器是只读的还是只写的。它们的地址由表 2 来确定。发送器 FIFO 寄存器可由两个不同的地址即 “帧终止” 和帧连续” 地址来访问。

表 2　寄存器地址

选中的寄存器	R/W	RSI	RSO	地址控制 位 (C_1 b_0)
写控制寄存器1*	0	0	0	×
写控制寄存器2*	0	0	1	0
写控制寄存器3*	0	0	1	1
写发送FIFO（帧连续）	0	1	0	×
写发送FIFO（帧终止）	0	1	1	0
写控制寄存器4*	0	1	1	1
读状态寄存器1*	1	0	0	×
读状态寄存器2*	1	0	1	×
读接收FIFO	1	1	×	×

接收数据先进先出寄存器(R_XFIFO)

R_X FIFO——它包 含三个八位寄存器，用于接收数据的缓冲存贮。数据字节总是从装满的寄存器传送到邻近的空余的寄存器，E时钟的两个相位都可产生数据传送。每个寄存器都有一个指示帧边界的指示位。当这些指示位出现在最后的 FIFO 位置时，它们修正 “现行地址” 位、“帧有效” 位或FCS/IF “错误状态位

RDA状态位表示R_XFIFO 状态。当 RDA 状态位为“1” 时。R_XFIFO准备读出。RDA 状态由 2 字节/1 字节控制位控制。当 发 生溢出时， R_XFIFO 中第一字节的数据不再有效。

R_X 复位 位 和$\overline{复位}$输入都能对 R_X FIFO 清零。中停和$\overline{DCD}$输入高电平也能对R_XFIFO 清零。但是，由帧边界指示位分开的前一帧的最后字节不会受到影响。

发送器数据先进先出寄存器(T_XFIFO)

T_XFIFO 一它包含用作发送的数据进 行缓冲存贮的三个八位寄存器。数据总是从满的寄存器向邻近的空的寄存器发送，E 时钟的两个相位都可产生数据发送。T_XFIFO 用” 帧终止” 和“帧连续”两种不同地址来编址。每个寄存器都有指示帧边界的指示位，它通过 FIFO 移位。数据字节写入 “帧连续” 地址时，第一个 FIFO 指示位置 “1”；写入 “帧终止” 时，则复位。R_XR_S 或 T_X中停复位所有指示位。当在 FIFO 的第三个位 置检测到正跳受时发送器启动一个带有开头标志的帧；而在这个位置检测到负跳变时，发送器结束一个帧，并把 FCS和结尾标志添加到帧的最后字节的后面。

T_X 的最后一位控制位可当作“帧终止” 地址来用。当T_X的最后一位控制位置 “1” 时，则逻辑搜索最后字节在 FIFO中的位置，并对 FIFO寄存器中的指示位进行复位。

T_XFIFO的状态由TDRA 状态位来表示。当 TDRA 为 “1” 时，T_XFIFO 装入的数据是有效的。TDRA 状态受 “2 字 节/1 字节控制” 位控制，而T_XFIFO 是由 “T_X复位” “和$\overline{RESET}$输入” 复位的。这样的复位条件或 $\overline{CTS}$ 输入为高电平， TDRA 状态位 被 封锁，并禁止数据装入。

控制寄存器

控制寄存器 1 （CR1）

控制寄存器 1 (CR1)

RS_1	RSO	R/W	AC	7	6	5	4	3	2	1	0
0	0	0	X	TxRS	RxRS	不连续	TDSR 模式	RDSR 模式	TIE	RIE	AC

ADLC 内部寄存器结构

位*	$RS_1RSO=00$	$RS_1RS_0=01$	$RS_1RS_0=10$	$RS_1RS_0=11$
	状态寄存器1*	状态寄存器2*	接收数据寄存器	
0	RDA	现行地址	位 0	与
1	状态2*读请求	帧有效	〃 1	
2	环路	不动作空闲接收	〃 2	$RS_1RS_0=10$
3	标志检测	中停接收	〃 3	相同
4	$\overline{CTS}$	FCS错误	〃 4	
5	Tx欠载运行	$\overline{DCD}$	〃 5	
6	TDRA/帧完结	Rx溢出	〃 6	
5	IRQ请求	RDA	〃 7	

位*				发送数据	发送数据	
	CR_1*	CR_3*($C_1b_0=0$)	CR_3*($c_1b_0=1$)	连续数据	最后数据($c_1b_0=0$)	CR_4*($c_1b_0=1$)
0	地址控制 ((AC)	优先级状态启动	LC字段选择	位 0	位 0	双标志/单标记内帧控制
1	RIE	2 字节/1字节传送	扩展控制字段选择	〃 1	〃 1	字长选择发送1*
2	TIE	标志/标记空闲	自动地址扩展模式	〃 2	〃 2	〃 2*
3	DMA	帧完结/TDRA选择	01/11空闲	〃 3	〃 3	字长选择接收1*
4	DMA	发送最后数据	标志选择状态启动	〃 4	〃 4	〃 2*
5	Rx帧间断	CLR Rx状态	环路/非环路模式	〃 5	〃 5	发送中停
6	RxRESET	CLRTx标志	联机动作/测试	〃 6	〃 6	中停扩展
7	TxRESET	RTS控制	联机控制DTR	〃 7	〃 7	NRZI/NRI

b_0 地址控制(AC)——AC 提供另外的一个内部RS（寄存器选择）的信号。AC位与RSO，RS1和R/W 输入一起，用来选择特殊的寄存器，如表 2 所示。

b_1 接收器中断启动 (RIE)——RIE 启动断开中断请求是由接收器部分产生的。当为 1 时为启动，0 时为断开。

b_2 发送器中断启动(TIE)——TIE 启动/断开中断请求是由发送器产生的。为启动，0 为断开。

b_3 接收器数据服务请求模式 (RDSR模式) ——RDSR 模式位在与优先级状态模式

一起使用的DMA模式中，由总线系统提供操作能力。当RDSR模式置位时，由RDA状态产生的中断请求被禁止。ADLC不能请求数据经由$\overline{IRQ}$输出发送。

b_4 发送器数据服务请求模式(TDSR模式) ——TDSR模式位在与优先级状态模式一起使用的DMA模式中，由总线系统提供操作能力。当TDSR模式置位时，由TDSR状态产生的中断请求被禁止，ADLC不能请求数据经由$\overline{IRQ}$输出发送。

b_5 R_X帧间断——间断位置位时，通常接收的帧无作用，并且ADLC废弃这帧的数据。当帧的最后字节被废弃时，间断位自动复位。当由于接收一个中停或DCD故障而使无用帧中停时，间断位也复位。

b_6 接收器复位(R_XR_S)——当R_XR_S位是“1”时，接收器部分停留在复位状态。包括R_XFIFO寄存器和二个状态寄存器中的接收状态位的所有接收器部分都复位。(此时，DCD状态复位。) R_XR_S强制$\overline{RESET}$为低电平或从数据总线对它写“1”而置位。当$\overline{RESET}$变为高电平后，R_XR_S必须通过数据总线写一个“0”来复位。

b_7 发送位复位(T_XR_S)——T_XR_S为“1”时，发送器部分停留在复位状态，并发送全“1”标记。此时，存贮的CTS状态亦复位，但它随$\overline{CTS}$输入而定，T_XR_S强制$\overline{RESET}$为低电平或通过数据写一个“1”置位。在$\overline{RESET}$变为低电平之后，它必须通过写一个“0”来复位。

控制寄存器2 (CR2)

控制寄存器2(CR2)

RS_1 RS_0 R/W AC

7	6	5	4	3	2	1	0
RTS	CLR T_XST	CLR R_XST	T_X 最后	FC/ TDRA 选择	F/M 空闲	2/1 字节	PSE

b_0 优先级状态启动(PSE)——当PSE置位时，两个状态寄存器中的状态位优先级由“状态寄存器”部分来确定。PSE为低电平时，除始终封锁TDRA状态的$\overline{CTS}$以外，其余都表示当时状态。

b_1 2字节字节/1字节数据传送(2/1字节) ——它复位时，对单字节传送，TDRA和RDA表示与它们有关的数据寄存器可用。同样，若2/1字节置位，则它们表示2字节数据可以移动。

b_2 标志/标记空闲选择(F/M空闲) ——F/M对空闲态选择标志字符或空闲标记。选中标记空闲时，在环路操作时与01/11空闲位(c_3b_3)产生“向前”代码。“1”表示时间满填；“0”表示标记空闲。

b_3 帧完成TDRA选择(FC/TDRA选择) ——它选择TDRA或FC状态。“1”表示FC状态；“0”表示TDRA状态。

b_4 发送最后数据($T_XI_{as}t$)——T_X最后数据位为终止一个帧提供了另一种方法。当Tx最后位恰好在装入数据字节后置位时，则ADLC假定此字节为最后字节，并且通过添加上CRCC和一个结尾标志，使此帧终止。这控制位通常用于DMA操作。T_X最后位能自动地回到“0”状态。

b_5 接收器状态清零($CLRR_XST$)——当$CLRR_X$ ST位写入一个“1”时，状态寄存器1*和2*中的接收器状态位(除AP和RDA位以外)产生了一个复位信号，此信号只有在最后”读状态“操作中已经存在的那些位才能被启动。$CLRR_XST$位自动地回到“0”状态。

b_6 发送器状态清零（CLR T_xST）——当CLR T_x ST位写入一个“1”时，对状态寄存器1*中的发送器状态位（除TDRA以外）产生了一个复位信号。此信号能被最后“读状态”操作时已经存在的那些位启动。CLR T_x ST 位能自动地回到“0”状态。

b_7 请求发送控制(RTS)——当RTS位为高电平时，使得$\overline{RTS}$输出变为低电平（有效状态）；当RTS位转回到低电平，并且数据已被发送时，$\overline{RTS}$输出保持低电平直至帧的最后字符（结尾标志或中停）已完结以及T_x FIFO空余为止。当RTS位转回低电平时，如果发送器正空着，则$\overline{RTS}$将在2个位的时间内变为高电平（无效状态）。

控制寄存器3(CR3)

b_0 逻辑控制字段选择（LCF）——LCF选择位使属于信息字段的数据第一字节保持8位字符，直至逻辑控制信息完成为止。逻辑控制字段（当被选中时）是可以自动扩充的一种字段，在逻辑控制字符的第八位（即位7）为“1”时它就可被扩充。当LCF选择位复位时，ADLC假定无论是发送或是接收数据通道都无逻辑控制信息存在。当LCF终止时，则信息数据字长由WLS_1和WLS_2来确定。

控制寄存器3(CR3)

RS_1	RS_0	R/W	AC
0	1	0	1

7	6	5	4	3	2	1	0
LOC/DTR	GAP/TST	环路	FDSE	01/11空闲	AEX	CEX	LCF

b_1 **扩充的控制字段选择**(C_{EX})——C_{EX}位为“1”时，控制字段是被扩充了的，并且假定为16位；当C_{EX}位为“0”，控制字段假定是8位。

b_2 **自动/地址扩充模式**(A_{EX})——它为“低”时，由于地址扩展被禁止，因而允许用地址八位位组的整个八位来寻址。A_{EX}为“高”时，地址八位位组的第0位为“0”，使地址字段扩充为一个八位位组。第一个八位位组全为“0”时为无效地址。

b_3 **01/11空闲——它决无用空闲态否**是从“0”开始。若b_3置位，则结尾标志(或中停）后面接着是0111 1111……型式，这在环路模式中用“向前”字符是需要的。当b_3复位时，空闲态全为“1”。

b_4 标志检测状态启动(FDSE)——它启动状态寄存器1*中的FD状态位以表示接收的标志字符出现。RIE置位，表示这些状态同时发生一中断。

b_5 **环路/非环路模式**——环路位置位即选中环路模式操作。同时选中GAP/TST，LOC/DTR和$\overline{LOC/DTR}$，以完成环路控制功能。环路复位，ADLC以点对点的数据通信模式工作。

b_6 **联机动作**/TST (GAP/TST)——在环路模式，GAP/TST位用于响应定时询问序列和起始发送。当b_6置位时，接收器对“向前”进行搜索。“向前”转换成开头标志，并开始ADLC的自身发送。此时b_6复位，帧的结束（标志或中停结束）使“联机动作”操作终止，并重新建立“R_x数据”到“T_x数据”链。

在非环路模式，b_6用于自检测。通常操作时b_6应是复位的。如置位，则T_xD输出在内部与R_xD连接，并提供“环路反馈”性能。

b_7 环路联机控制/DTR控制(LOC/DTR)——在环路模式，LOC/DTR用于联机或脱机工作。它置位时，R_xD输入产生连

续7个“1”以后，ADLC达到联机状态；复位时，RxD产生连续8个“1”后达到脱机状态。

在非环路模式，LOC/DTR直接控制$\overline{\text{LOC/DTR}}$输出状态。“1”表示$\overline{\text{DTR}}$输出变低电平；“0”为高电平。

控制寄存器4 (CR4)

控制寄存器4 (CR4)

7	6	5	4	3	2	1	0
RS_1	RS_0	R/W	AC	Rx WLS_2	Rx WLS_1	Tx WLS_2	Tx WLS_1	“FF”“F”

b_0 **双标志/单标志互联帧控制**(“FF”/“F”)——它在一些帧连续地发送时确定发送器是否将发送结尾标志和开头标志分开。当“FF”/“F”控制位为低时，第一个帧的结尾标志作为第二个帧的开头标志用；当为高时，开头标志和结尾标志将单独发送。

b_1,b_2 **发送器字长选择** (T_x WLS_1 & WLS_2)——$T_x$$WLS_1$ & WLS_2 位用于选择发送器信息字段的字长。其编码格式如下表3所示。

表3 I字段字符长度选择

WLS_1	WLS_2	I字段字符长度
0	0	5位
1	0	6位
0	1	7位
1	1	8位

b_3,b_4 **接收器字长选择** (R_x WLS_1 & WLS_2)——R_x WLS_1 & WLS_2 用于选择接收器信息字段的字长。其编码格式如表3所示，

b_5 **发送中停**(ABT)——ABT位产生一个被发送的中停(至少有连续的8个“1”)。当控制位变为高时，中停启动，并且T_xFIFO被清零。一旦中停开始，T_x中停控制位出现低电平状态。

b_6 中停扩充(ABT_{EX})——如果ABT_{EX}置位时，由ABT启动的中停代码扩充至少有16位连续“1”，即标记空闲状态。

b_7 NRZI (**零补码**)/NRZ**选择**(NRZI/NRZ)——NRZI/NRZ位选择环路模式或非环路模式两种操作，发送器/接收器数据格式是NRZI还是NRZ。当NRZI模式被选中前，则在发送的数据上附加1位延迟，以供NRZI编码。1表示NNRZ，0表示INRZ。

注意：NRZI编码——串引数据保持相同状态发送一个二进制“1”，而变换成相反状态则发送一个二制“0”。

状态寄存器

状态寄存器1*是主要的状态寄存器。IRQ位表示ADLC请求服务与否。S_2RQ位表示状态寄存器2*中的任意位是否有任意请求服务。由于TDRA和RDA是最通常用的，所以它们位于易检测位的位置，RDA反映状态寄存器2*中的RDA位的状态。

状态寄存器2*提供了包含在S_2RQ位中的详细的状态情况，而这些位反映接收器的状态。FD位只是接收器的状态，它在状态寄存器2*中无表示。

优先级状态模式为搜索各状态位提供了最大效率，以及起到表示ADLC服务所要求的最重要作用。两种状态寄存器所提供的优先级排列如图11所示。

读状态寄存器是一个非破坏过程。状态清除的方法依赖于各状态位的功能。状态寄存器中的每一状态位将在下面讨论。

状态寄存器1 (SR1)

RS_1	RS_0	R/W	AC
0	0	1	X

7	6	5	4	3	2	1	0
IRQ	TDRA/FC	TXU	CTS	FD	LooP	SZRQ	RDA

b_0 接收器数据有效(RDA)——RDA状态位反映状态寄存器2# 中的RDA状态位的状态。在全双工模式中它提供实现接收数据传送的方式，不必读两状态寄存器。

b_1 状态寄存器2#读请求(S_2RQ)——状态寄存器2# 中的所有状态位（除RDA 位以外）都是逻辑或的，由S_2RQ状态位来表示。因此，S_2RQ 表示的状态寄存器 2#需要被读出。当 S_2RQ 为“0”时，则不一定要读状态寄存器2#。当状态寄存器2#适当的位被清零时或当用了R_X 复位时，此位被清零。

b_2 **环路状态**——环路状态位用于监控ADLC 的环路操作。此位不会产生 IRQ。当非环路模式被选中时，环路位停留在“0”当环路模式被选中时，“联环状态”环路状态位变成“1”。当ADLC 处于“离环状态”或“联环动作”状态，环路状态位为“0”。

b_3 **标志检测** (FD)——如果“标志检测”启动控制位已经置位，则 FD 状态位表示已接收到了一个标志。在接收到标志字符最后位（当$\overline{\text{标志检测}}$输出变低电平）时，FD 状态位变为高电平，并且一直存贮到由“清除R_X状态”或“R_X复位”清零。

b_4 **清零—发送**(CTS)——$\overline{\text{CTS}}$输入的正跳变存入状态寄存器，并产生一个IRQ(如果是启动的）。存入的CTS 状态及“IRQ”由清除“T_X 状态控制位”或“T_X 复位”位来清零。存入的状态复位后，CTS 状态位反映$\overline{\text{CTS}}$ 输入状态。

b_5 **发送器欠载运行** (T_XU)——在帧发送期间，当发送器数据缺少时，就会产生欠载运行，并且通过发送一个中停，帧就自动终止，欠载运行条件由T_XU状态来表示。通过清除“T_X 状态控制”位或“T_X 复位”就能将T_XU清零。

b_6 **发送器数据寄存器可用/帧完结** (TDRA/FC)——TDRA 状态位起两个作用，它随“帧完结”/TDRA“选择”控制位而定。当它作为TDRA 状态位时，表明发送的数据能够装入T_X数据 FIFO 寄存器中。“T_X 数据 FIFO”的第一个寄存器（即 7# 寄存器）由“1 字节传送模式”中的 TDRA状态位来表示，在“2 字节传送模式”中，TDRA 为高电平时，最先的二个寄存器（即1#和2# 寄存器）必定是空着状态在T_X复位或$\overline{\text{CTS}}$为高电平时，TDRA 被禁止。

当选中操作“帧完结模式”时，在发送了一个中停或一个标志而无数据在T_X FIFO 时。TDRA/FC 状态位变成高电平。此位一直保持高电平，到通过 TDRA/FC 控制位复位或T_X复位位置才被清零。

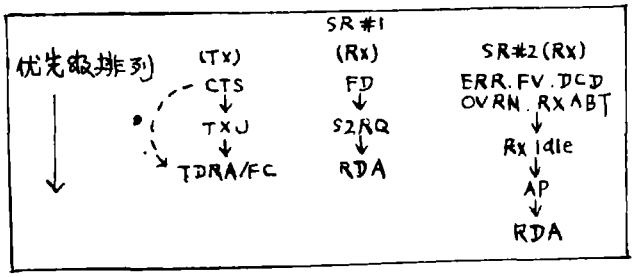
b_7 中断请求(IRQ)——中断请求状态位表示$\overline{\text{IRQ}}$输出处于有效状态（即$\overline{\text{IRQ}}$输出 = “0”)。IRQ状态位与 $\overline{\text{IRQ}}$ 输出一样都受相同的中断启动（RIE,TIE）所支配。在服务请求由单独一位表示的定时询问系统，IRQ 状态位简化了状态询问。

状态寄存器2 (SRZ)

b_0 **现行地址**(AP)——AP 状态位提供了帧的边界，并且表明，一个地址八位位组在R_X数据 FIFO 寄存器中是有效的。在扩充的寻址模式中，AP 位连续表示地址直至地址字段完结为止。现行地址状态位通过读数据或 R_X 复位来清零。

b_1 **帧有效**(FV)——FV 状态位提供了

图11　状态寄存器优先级形势(PSE = 1)



• 即使PSE=0时,也优先排队

*　甚至当PSE = 0时也是优先级化的。

注意：上一位状态位将禁止下一位状态位。

状态寄存器2 (SR2)

状态寄存器2(SR2)

RS_1	RS_0	R/W	AC	7	6	5	4	3	2	1	0
0	1	1	X	RDA	OVRN	DCD	ERR	Rx ABT	Rx 空着	FV	AP

到 MPU 的帧边界标记，同时也表明了帧的完结而无错误。当帧的最后数据字节被发送至R_X FIFO 的最后位置（通过 MPU 可以读出）时，FV状态位置位。一旦FV状态置位，ADLC就停止了数据进一步向R_XFIFO最后位置传送（以防止二个帧混合），直至清除R_X状态位或R_X复位，才被清零。

b_2 **无效空闲接收**(R_X 空闲)——R_X空闲状态位表示已经至少接收到连续15个“1”。并存贮在状态寄存器内，产生一个中断。这个中断和存贮的状态由清除“R_X状态控制”位来清零。状态位是接收器空闲检测器及存贮的无效空闲状态的逻辑或。

b_3 **中停接收**(R_XABT)——R_XABT表示已接收到多于七个连续“1”。在帧失调的情况下，中停无意义。它既不产生中断，也不产生状态存贮。b_3是R_X中停检测逻辑和存贮状态的逻辑或。它在连续产生15个“1”以后被清零。而存贮的中停状态是通过Rx状态控制位或R_X复位来清零。

b_4 帧校验序列/无效帧错误 (ERR)——当帧用 CRC 错误或短帧错误完成时，ERR位置位以替代帧有效状态位。帧边界标记和控制作用，事实上与帧有效状态位相同。

b_5 **数据载波检测** (DCD)——$\overline{DCD}$输入端的正跳变存贮状态寄存器，并产生一个IRQ。DCD和IRQ由清除R_X状态控制位或R_X复位来清零。存贮的DCD状态和$\overline{DCD}$输入为高电平时，使接收器部分复位。

b_6 **接收器溢出** (OVRN)——OVRN状态表示接收器数据传送至R_XFIFO，当装满时就会产生数据损失。它由清除R_X状态位或R_X复位来清零。连续的溢出，仅能使节一个FIFO寄存器中的数据无效。

b_7 **接收器数据有效**(RDA)——RDA表

示接收器数据可以从R_x数据 FIFO中读出。当用优先级状态模式时，RDA表示R_x FIFO中的无地址和无最后数据是有效的。在“1字节传送模式”中，正在 FIFO 最后寄存器中的接收器数据使得 RDA 变为高电平。这表明当采用“2 字节传送模式”时，最后的两个寄存器是全满的。当数据无效时，RDA自动复位。

程序设计考虑

（1）**状态优先级**——当采用优标级状态模式时，最好是首先检查最低级优先级的条件。因为最低级优先级条件最通常产生，并且当处理器中断时是最可能出现的状态。

（2）**存贮存态对现行状态**——某些状态位（如DCD,CTS,R_x中停或R_x 空闲）表示存贮的与现行的状态的逻辑或的状态。存贮的状态位产生一个中断，它由状态清零控制位来清零。清零以后，状态寄存器将反映任一输入或接收器输入顺序的现行状态。

（3）**状态寄存器清零**——为了用两个状态控制位消除中断，必须在其清零之前先读出一个。因为在优先级模式中，由于较低级优先级先读出，它的状态被抑制了，因而在清除较高级优先级状态时，导致了其它的$\overline{IRQ}$ 的产生，这样就保证了状态不再被偶然清零。

（4）R_xFIFO **清零**——一个R_x复位将有效地清除了三个 R_x FIFO 字节的全部内容。然而，当产生中停或DCD失效时，FIFO可能包含 2 个不同帧的数据。此时，来自上一帧结尾（即已接收到结尾标志的帧）的数据将不被破坏。

（5）在 2 字节模式中 R_x FIFO 服务——由于读数据最后字节的顺序是相同的，所以不管帧中所含字节数是偶数还是奇数。连续读 2 个字节直至由帧状态（FV或ERR）的结束引起的中断出现为止。当出现中断时，表明最后字节或是已经读出或正准备读出，暂时变换到无优先级状态（CR2*）的 1 字节模式。检测RDA以表示读 1 字节是否完成，然后再清除帧的结束状态。

（9）帧完成状态和$\overline{RTS}$释放——在许多场合中，MODEM将需要延迟以释放$\overline{RTS}$。通过采用一个中停，可以把一个 8 位或16位延迟加添到 ADLC 的$\overline{RTS}$输出上。在发送结束时，帧完结的状态将表示帧的完结。此状态变为高电平后，将“1”写入到中停控制位（以及当需要16位延迟时写入到中停扩充位）。中停控制位复位后，将“0”写入到 RTS 控制位。发送器将发送 8 个或 16个“1”，然后$\overline{RTS}$变为高电平（无效状态）。

（7）不使用 MC6800 的用户注意——(a)在高频连续 E 正脉冲时，在完成一个写后接着就读时应当小心。因为状态变化必须有一定时间。如果此脉冲已经完成，连续读/写的 E 脉冲之间为低的时间至少有 500nS。（b）ADLC 完全是一个静态部件，然而，E 频率将必须足够高，以通过 FIFO移动数据和服务于外设。为了保证数据总线与外设之间的同步，连续的 E 脉冲之间的周期应小于R_xC或T_xC的周期。

吕必惠摘译　　张正刚　校