

# AN<405>

SCN2681 / SCN68681 和 SCC2691 数据通信

Rev. \_2 — 21 09 1998

应用规格书

## 文档信息

信息	内容
关键词	Uart, 数据通信
摘要	SCN2681 / SCN68681 和 SCC2691 数据通信

修订历史

版本	日期	说明
_2	19980921	应用规格书（9397 750 04566）。
_1	19860801	应用规格书，初始版本

联系信息

关于额外的信息，请访问：<http://www.semiconductors.philips.com>

关于销售办事处的地址，请发送电子邮件到：[sales.addresses@www.semiconductors.philips.com](mailto:sales.addresses@www.semiconductors.philips.com)

## 1. 概述

### 1.1 介绍

这份SCN2681 / SCN68681和SCC2691数据通信应用规格书包含了一些用户最常提出的问题的解答，本文共分三个主要部分：三个器件共有的功能；SCN2681 / SCN68681特有的功能；以及使用乐器数字接口（MIDI）的SCC2691的典型应用。

### 1.2 DUART/UART 共有的功能

#### 1.2.1 读取保留的寄存器

对02H或0AH单元执行一总线读操作，将会使器件进入诊断操作模式，这种诊断模式用于测试波特率发生器电路，当对任一地址执行一个读操作，器件将在通用输出端口输出2倍于波特率表上频率的时钟脉冲。

有时会意外地进入该模式，在某些场合，监控程序在一个自动读取/验证周期后紧跟一个写操作便会进入该模式，例如，这种类型的监控中对CRA或CRB执行一个写入操作将调用保留测试模式。当一个开发系统是工作在手动模式时必须小心，因为大多数开发系统在一个读取/验证周期后紧跟一个的写操作，很多时候，这种异常情况会发生在SC68681，如果写入脉冲的上升沿发生在CEN的上升沿之前，保留模式将被调用（即使R/WN仅仅提早10到20毫微秒上升），SCN68681的用户一定要确保R/WN的上升沿沿着CEN的上或者是在它后面。

#### 1.2.2 接收器 FIFO

这个器件都具有一个三字节深度的接收FIFO，这些FIFO就像是一个循环队列，接收器具有一个头指针和一个尾部指针，头指针受一个总线读操作的控制，当对接收器执行了一个读操作时，头指针就会跳到下一个位置，任何时候当一个字符在接收移位寄存器中被组装并传输到接收保持寄存器时，尾部指针就跳跃移动。

在施加了一个外部复位或执行了一个复位接收器命令之后，头指针和尾指针都处在FIFO中的相同位置上，虽然数据表中指出当执行一个复位接收器命令时接收器将被清空，但对接收器的内容则不会有任何影响。因此，对接收器连续执行三个读操作将使头指针走了一圈回到起始点，如果接收移位寄存器没有接收到任何新的数据，则旧的数据将仍然存在于FIFO中。

当在手动模式下使用监控时必须小心，因为对发送保持寄存器（THR）执行一个写操作可能会导致头指针的移动。（紧随一个读取/验证操作，对上面已经讨论过的相同地址执行一个写操作，参见读取保留寄存器）。

判定接收器是否应该被读取的最佳方法是查询ISR或SR寄存器中的RxRDY位，如果RxRDY = 1，再次读取接收器，重复执行直到RxRDY = 0为止，一旦达到这种状态，停止读取接收器，否则指针将超过当前有效数据。

#### 1.2.3 检测中止的结束

检测中止是内置到这三个器件中的一个简单功能，接收器连续地对RxD进行取样，如果在全编程数目的位时间内检测到一个低的起始位，则一个零字符就积累在接收FIFO中，如果没有检测到停止位（标记条件），那么接收器将对字符帧再多进行一个位时间的取样，如果检测到一个低位，说明已经检测到一个帧错误，一旦帧错误位被设定在状态寄存

器中，而且一个零字符累积在接收FIFO中，这种状况将使到ISR和SRA或SRB中的接收中止位置位，通过这种方式检测到一个中止的开始。

为了检测中止的结束，必须检测ISR寄存器中的数据中止位，不论CPU是一直查询ISR还是采用中断驱动，必须在中止状态完全结束后才能读取接收FIFO中的零字符和清除SRA或SRB中的接收中止位。为了检测中止的结束，CPU应该执行一个复位中止命令（写50H到CRA或CRB），它将复位ISR的中止位，如果CPU正在使用中断，下一步就是屏数据中止中断，如果CPU正在查询，则应该继续查询直到数据中止。只有当RxD上发生了状态变化时，数据中止位才会被设定和产生中断，当RxD的上升沿发生而且数据中止结束，数据中止位能被清除（写50H到CRA或CRB），接收数据中止和帧错误能够清除（写40H到CRA或CRB），而且可以从接收器上读取零字符和丢弃它。

### 1.2.4 在一个短帧之后禁用发送器

数据表中描述，在最后一个字符被加载到发送移位寄存器中之后，可以禁用发送器。这将使得在最后一个字符移出发送移位寄存器之后，结束一个块传输或让RTS失效，如果使用RTS作为握手信号，则这种禁用发送器的方法是必不可少的。当要对主站或次站的响应而发送短的、单字节或双字节帧时，这不是一个好方法，当发送器被重新启用以发送另一个短帧响应时，问题就出现了，如果在执行一个启用发送器命令时，信息的最后一个字符仍仍然正在被串行化，则串行化的字符将会发生混淆或丢失。

例如，假设一个双字节帧的最后一个字符被装载到THR中，如果发送器工作在9600的波特，那么，最后一个字符完整地移位到TxD所需的时间为1到2毫秒（当双字节帧里的第一个字符目前处在TSR中时，需要2毫秒的时间），如果CPU需要发送另一个响应，将开始启用发送器，并加载下一个帧的第一个字节，即使上一个帧的第二个字节仍在处在TSR中，由于重新使能命令而使发送器重新执行，那么最后一个字符要么丢失，要么混淆。

避免这个问题的最佳方法是：等最后一个字符完整地移出TSR时才禁用发送器，这可以通过查询TxEMPT位而加以验证，直到它在SRA或SRB中被置位为止。正如前文所述，如果使用RTS时这将失效，因为只有在最后一个数据字符处于THR中时禁用发送器RTS输出才会切换，在这种情况下，用户可以在重新使能发送器前作一个时间的延迟。

### 1.2.5 防止发送器和/或接收器的出错

如果没有首先禁用发送器和接收器便试图对模式寄存器（MR2、MR1）、时钟选择寄存器（CSRA/B）、到ACR [7]进行写操作，将会遇到问题，如果在字符串化仍在进行期间改变模式寄存器，则传送就可能在新设置的模式下启动，如果模式寄存器是在串行化已经完毕，而且发送器为空之后改变，将会有两个潜在的问题出现：TxD可能会出现一小段时间的空白状态，或者ISR中的TxRDY将被置位然后复位。这些结果的产生与写入到模式寄存器上的数据没有关系，对寄存器编程为当前值仍然能够产生出上述结果。

当时钟仍然在运行时便对CSRA/B或ACR [7]执行写操作，问题将更加严重，如果在没有禁用发送器和接收器的情况下就改变发送和/或接收时钟，那么在从一个频率改变到另一个频率的过程中，就可能出现一个断续的或缩短的时钟，这些短的时钟脉冲可能锁定发送器和/或接收器，直到对CRA或CRB的执行重新启用命令为止。

避免这些问题的最佳方法是，在改变模式寄存器、CSR寄存器、或ACR [7] 寄存器之前就禁用发送器和接收器。禁用发送器和接收器将会在改变的时候停止TXC和RXC，当改变完成后，应该通过软件命令对发送器和接收器复位然后重新使能，复位命令将确保在开始修改之前，一切都处于一个确定的状态。

### 1.2.6 设置计数器/计时器作为一个计时器

C/T（计数器/计时器）能够产生的最大频率取决于是否使用1X或16X时钟源模式，如果C/T被用来直接驱动接收器，则必须产生一个16X时钟来对进来的数据流进行取样。

例如，设X1/CLK输入为4 MHz，并设C/T为它的最小值（CTUR/CTLR = 0002H），因为C/T将向下倒数直到零才改变输出状态，所以在C/T输出变回到它的初始状态之前，必须经过两次完整的计数，因此，最大频率输出将等于4 MHz除以4（最小计数）再除以16（对于抽样时钟），这样得出的结果是一个62.5 kHz的波特率。

如果需要较高的C/T时钟速率，则必须编程1X模式，最高的C/T输出是1 MHz（4 MHz除以4），C/T能被编程为输出1 MHz到OP3上，它可以通过线连接到IP3/4或IP5/6上，这些输入能被选择为发送和接收的1X时钟输入。编程此类功能，写入OPCR = 04H，ACR = 60H，CTUR/CTLR = 0002H，CSRA/B = FFH，通过对地址0EH上执行一个读操作（启动计数器命令）来启动C/T。在计时器模式时，C/T将不会停止振荡直到ACR[6: 4]被写入为计数器模式，并紧跟上一个停止时间命令（读取地址0FH）。

可以使用计数器来对外部事件进行计数，或作为一个系统的延时计时器，例如，C/T能被设置为一个2毫秒的延时，并中断CPU（用于刷新动态RAM），在4 MHz的频率下使用X1/CLK作为C/T时钟，计数中的总时间就等于（2ms） / （250ns x16） = 500或01F4H（因为X1/CLK除以16是在计数器模式中唯一可用的内部时钟源，所以要乘以16），这个功能将通过写入ACR = 30H，CTUR/CTLR = 01F4H，和IMR = 04H而实现，对于计数器每次的计数，必须由CPU的命令来控制停止和开始（停止 = 读地址0FH，开始 = 读地址0EH）。

### 1.2.7 多点/唤醒模式

如果MR1[4: 3]=11，则器件为多点或唤醒（SCC2691）模式，这将使发送器送出的数据字符的最后一位将被作为识别地址/数据（A/D）位。如果MR1[2] = 0，则A/D位将等于‘1’，则组成的字符将被次级接收器识别为一个地址，主站和从站二者都必须处于多点模式，为了发送一个后面跟着数据字符的地址字符，必须对MR1执行一次写操作，如果发送器和接收器没有被禁用，则对模式寄存器执行写操作可能导致数据混淆。在对模式寄存器执行写操作时，应该按照下列顺序：

1. 为了保证发送器处于一种静止状态，在TxEMT = 1之前不要写入MR1。
2. 复位MR指针，禁用发送器和接收器。
3. 用原来写入的数据写入MR1，使MR1[2] = 1（Tx地址），复位并启用发送器和接收器。
4. 将地址字符载入到发送保持寄存器（THR）中，传送过程中，A/D位将根据MR1[2]的极性而附着到字符上。
5. 等待，直到TxEMT = 1。（等待发送器腾空）。
6. 复位MR指针，禁用发送器和接收器。
7. 用原来的数据写入MR1，使MR1[2] = 0（Tx数据），复位并使能发送器和接收器。
8. 将第一个数据字符装载到THR中，继续发送数据直到信息完成，要发送一个信息到一个不同的地址，重复步骤1到8。

当从站设定RxRDY = 1时，CPU必须立即读取接收器，以确定地址是否正确，如果接收到的地址相符合，则CPU必须设定CRA或CRB中的RxEN = 1，以便接收信息，在较高的波特率（19.2k和38.4k）条件下，有些用户会丢失了信息的第一个字符，发生这个问题的原因是：在启用接收器之前，CPU从接收器中读取地址，并完成一次比较操作，所有这些要花费一定时间，如果是这种问题，则一旦地址被接收到而且CPU产生中断，CPU就使能接收器（RxEN = 1），稍后，在比较操作完成之后，根据接收到的地址的情况，CPU要么从接收器中读取数据，要么复位接收器（参见图1中给出的软件示例）。

### 1.2.8 处理中断

当发送器被使能且TxRDY在IMR中没被屏蔽时，中断将会立即发生，如果没有信息需要发送，则用户应该屏蔽掉TxRDY（IMR [4: 0] = 0），否则没有其它中断会发生，只有器件的中断部分能复位当前的中断。

例如，装载发送器复位TxRDY，读取接收器复位RxRDY，停止计数器复位计数器准备，等等，如果这些功能没有完全地执行，则所有其它等待的中断都被阻碍，为了避免这个问题，只使能立即投入使用的IMR位，仅在一个块传输开始之前和结束时启用和禁用IMR中的TxRDY位。

### 1.2.9 从外部驱动 X1 或使用一个晶振

如果用户想使用一个外部时钟而不是一个晶振，最佳的方法是驱动X1和让X2接地（只对于SCN2681 / 68681），数据表说明了用来驱动X1使它与X2异相的两个反相器，虽然这可以接受，但它牺牲了一个门的使用，从一个外部源来驱动时钟输入的唯一缺陷是：要求最低的高电平为4.0伏，通过使用一个开集缓冲器（有电阻），或者通过一个上拉电阻连到平常的TTL缓冲器（确保VOL小于0.8伏），就能保证这个大于4.0伏的VOH。另一个要求是，最小高和低时钟脉冲宽度必须是100纳秒，利用一个具有50 %占空比的外部振荡器，就能很好地达到这个参数。

另外比较敏感的问题就是在X1 / X2输入上使用一个晶振时候，如果电容C1和C2为15个pF或更大，则可能会间断地碰到上电的问题，由于电容上的充电不足，晶振可能不振荡，建议C1和C2大约为5 pF，以保证在上电期间能充分充电。当晶振接在X1和X2之间时，许多DUART都出现一个60 / 40的占空比，用一个示波器对X1进行检测时，典型的高位的时间是90纳秒，而低位时间则大约为130纳秒，为了使晶振工作在50 %的占空（让时钟的高位和低位时间大约为110纳秒），应该在X1输入和X2输之间增加一个100 kΩ或更大的电阻器，这将增加振荡器的行程，使晶振的高位和低位时间相等。

SCC2691并不存在以上的晶振问题，当由外部源驱动时，SCC2691与其它器件不相同，每一次当X1被驱动时，必须置X2为开路，如果X2接地，则时钟将不振荡，请注意，X1输出只能驱动一个CMOS外部缓冲器，必须要注意：不使X1/CLK输入过载。

### 1.2.10 X1 / X2 晶振

在订购3.6864 MHz的晶振时，只要频率公差能接近+ 0.005 %，则串行或并行晶振都可以使用。由于X1 / X2电路的特性，应该使用并行晶振。测试表明，如果公差值很低，则频率中的错误可以忽略不计，如果公差小于等于+ 0.005 %，可以使用一个串行晶振。关于晶振样品，请打致电：Saronix，位于美国加利福尼亚的帕洛阿尔托市，请求零件号NMP037：3.6864 MHz HC-18/U。第二个晶片来源是美国晶振公司，位于美国得克萨斯州的沃思堡市，请求零件号为：SIG36864-HC18。

### 1.2.11 普通初始化

图2描述了软件初始化的典型流程，通常情况下，模式寄存器处在第一位，如果在初始化之前执行复位，则不需要复位MR指针，请注意，当任何一个模式寄存器（MR1A/B和MR2A/B）被装载时，应该禁用并复位发送器和接收器。

### 1.2.12 异步诊断

图[3]是一个软件功能程序，它能用来测试数据总线以及TxDA、RxDA、TxDB和RxDB的完整性。该程序一开始便初始化通道‘A’和‘B’，接着复位MR指针，读MR1A并与

写入的值进行比较，如果比较通过，则一个继电器被接通，它将TxDA和RxDA、以及TxDB和RxDB短路，由于通道都处于普通模式，这就构成一个外部回循环。当发送器进入准备状态时，向发送器‘A’装载入256个字符，当接收器产生中断给CPU时，读取接收FIFO，并将其内容与已经发送的内容进行比较，如果全部256个字符都已正确地接收，则信道‘B’也将以相同的方式进行测试。由于SCC2691没有A3来选择信道‘B’，所以这个测试的后一半将是采用不同的初始化值来对SCC2691做一个简单的重复测试。

### 1.3 SCN2681 / 68681 的独特功能

#### 1.3.1 异常的数据中止

当给“版本E”的器件上电时，有的数据中止位（ISR [6: 2]）被置位，这将可能导致两种问题：（1）如果这些位在IMR中没被屏蔽，则它们将立即产生一个中止中断；

（2），接收到RHR中的第一个字符将会在SRA/B中被标记为出错（帧错误，中止错误或奇偶校验错误），即使字符的接收是正确的。

清除这些错误的方法是，在上电复位1秒后执行一个外部硬件复位，或者是在启用发送器和接收器之前对CRA或CRB执行一个清数据中止命令，最好的方法是：首先禁用发送器和接收器，接着初始化所有的寄存器（Mr1, MR2, CSRA/B, ACR, 等），然后通过命令寄存器清除所有错误，为了使它生效，这必须在初始化流程的末处执行，发给CRA或CRB的命令的结束字符串大体如下：

CRX = 50H （清除数据中止位）  
 CRX = 20H （复位接收器）  
 CRX = 30H （复位发送器）  
 CRX = 45H （清除错误，使能Tx/Rx）

#### 1.3.2 RTS / CTS 功能

在使用RTS / CTS功能时，必须要遵守数据表中的如何设置RTS的流程图，虽然RTS的输出将会自动反转，但在发送器被使能之后，而且在信息的第一个字节载入到THR之前，要对相应的输出引脚写入‘1’来激活输出，当接收器控制RTS的反转时，必须在使能接收器之后立即对相应的输出引脚写‘1’。

当接收器控制RTS的反转时，当FIFO充满，以及在RSR中检测到第四个字符的起始位时，正在发送的发送器将会被停止，如果发送数据的发送器是一个飞利浦器件，则在当前TSR中的字符被完整地移位到TxD上时，传送就将结束。

当CTS变高，在当前字符移出到TxD上之后，发送器时钟就停止。这样导致的唯一问题是，SRA或SRB中的TxEMPT位将不会置位（即使发送器是空的），直到时钟再次开始运行为止（当CTS返回到低位时）。

接收器可以在RHR上保留三个字符，在RSR上保留一个字符，如果正在发送的发送器不是一个飞利浦器件，则RSR中的字符可能会由于第五个字符而出现溢出。例如，如果正在发送的发送器是由英特尔公司制造的，当它的CTS输入是高时，发送器将继续腾空THR和TSR，虽然它不允许发送任何其它字符，但接收器移位寄存器（RSR）仍然溢出。

#### 1.3.3 输入端口

40引脚型号的SCN2681和SCN68681有一个7位输入端口可以通过两种不同的方式来读取，通过对十六进制地址‘OD’上执行一次读取，就能以并行方式读取该端口，输入端口的低四位也能通过输入端口改变寄存器（IPCR）来读取，IPCR中的位将会跟随IP0 - IP3变化，IPCR [0: 3]显示了IP0 - IP3的当前状态，如果从上次读取IPCR后又发生状态的

变化，则IPCR [4: 7]将被置位。用户会注意到，当读取地址‘OD’时，低四位数据可能有差异，出现这种差异的原因是，IPCR的更新是通过一个运行在38.4k时钟上的内部状态机来完成的，更新IPCR将至少花费一个时钟时间（25 μs），由于对输入端口的读取是立即进行的，所以两个值可能存在差异。

为了说明IP0 - IP3是如何中断CPU的，假设IP0输入连接到一些关键元件上，通过对IP0中断（ACR[0]）和输入端口改变掩码（IMR[7]）写入1来使能中断，应该通过执行一个IPCR的读取来复位变化位（IPCR [4: 7]），当IP0变化时，IPCR[4]将被置位，那么ISR[7]也将被置位，从而产生一个中断，该中断可通过对IPCR的执行一个读操作来复位。

### 1.3.4 输出端口

40引脚型号的SCN2681和SCN68681上的输出端口具有8个输出，通过向十六进制地址‘OE’或‘OF’（‘OE’ = 设定输出端口位，‘OF’ = 复位输出端口位）执行一次写操作，可以对每个输出进行置位或复位，这些位初始化时是高，给十六进制地址‘OE’上的一个单元写入一个1，相应的输出将变低；给十六进制地址‘OF’上的相同位置写入一个1，这个输出将复位为高。例如，通过在数据总线上给十六进制地址‘OE’写入十六进制‘80’，可使OP7变低。只要没有在OPCR中将输出编程为特殊功能，则任何输出都能在高位和低位之间进行切换。如果已经选择了OPCR中的一个输出，这个输出将处于内部状态机的控制之下，用户将不能控制它的极性。请注意，如果OPCR [4: 7]中的任何一个位被置位，则输出将是一个开集中断，需要一个上拉电阻器。

### 1.3.5 SCN2681 总线接口

CEN和RDN、以及CEN和WRN信号在SCN2681中内部进行逻辑“与”运算（ANDed），由于这种方式，上一个维持信号将启动周期，而第一个反转的信号将是周期的结束。对于写入操作，CEN或WRN的上升沿将数据锁存到SCN2681寄存器中，并结束周期，由于这些信号之间的关系，能够使用许多不同的总线接口技术，有些用户把CEN接地，使用一地址解码器来给RDN和WRN提供脉冲，有的用户则给CEN提供脉冲，同时用一个静态RN/W的和反转静态线路R/WN的RDN来驱动WRN，另外，还有一些用户使用了传统方法来给所有三根线路提供脉冲，在所有这些情况下接口都能工作。

AC电气特征状态中的注释10指出：连续写入到同一命令寄存器（CRA或CRB），在取消了对器件的选择之后，至少要有三个X1时钟的边沿，这样做是必不可少的，因为到命令寄存器的数据只在WRN（或CEN）的上升沿锁存，而且状态机在执行命令时需要三个额外的边沿。

### 1.3.6 SCN68681 总线接口

通过使用CSN或DTACK的下降沿，就能完成SCN68681写操作，与SCN2681的情况一样，对于访问器件的频率有一些限制，这是由参数t<sub>CSW</sub>确定的，它提供了CSN的最小高位时间的值（t<sub>CSW</sub>最小= 160纳秒）。DTACK是一个时钟输出，它是在CSN生效之后X1时钟的前两个上升沿发生的，因为CSN相对于X1的上升沿是异步的，所以DTACK的生效可以有所不同，在器件选择之间可能是一个完整的X1时钟长度。请注意，DTACK在生效是一个开集输出，需要一个上拉电阻器（参见读取保留寄存器部分，复习可能发生的接口问题的）。

### 1.3.7 使用 SCC2691 MIDI 接口

下面的例子很好地说明了SCC2691如何用作乐器使用的（MIDI）串行接口，乐器数字接口（MIDI）是正在所有新的电子乐器建立的标准，合成器、鼓机、音序器、和其它相关的音乐装置，以及家用电脑附件都正在使用这种串行数据接口。数字接口是异步工作的，

波特率为31.25k，有一个起始位、8个数据位、一个停止位。物理接口通过一个光隔离的5mA电流环路工作，通过‘进’、‘出’、或‘连通’端口，多件乐器可以以‘菊花链’的形式相互连接。

### 1.3.8 硬件接口和数据格式

图5显示了MIDI使用的硬件接口，请注意，MIDI‘连通’连接是一个可选连接，它能提供接收到的MIDI‘进’数据的一份复制品。

MIDI数据格式和波特率：

- 起始位 - 1
- 停止位 - 1
- 数据位 - 8
- 奇偶性 - 无
- 波特率 - 31.25k

### 1.3.9 SCC2691 波特率选择

SCC2691 自身用于接收器和发送器的波特率发生器是来自 18 个固定的速率，在给定的时钟频率 3.6864 MHz，通过它内部的除法器电路，实现了在低速数据通信中遇到的全部的常规的波特率。以下波特率可用：

CSR[7: 4]	ACR[7] = 0	ACR[7] = 1
0000	50	75
0001	110	110
0011	200	150
0100	300	300
0101	600	600
0110	1,200	1,200
0111	1,050	2,000
1000	2,400	2,400
1001	4,800	4,800
1010	7,200	1,800
1011	9,600	9,600
1100	38.4k	19.2k
1101	计时器	计时器
1110	MPI - 16X	MPI - 16X
1111	MPI - 1X	MPI - 1X

为了生成所要求的 31.25k 的波特率，需要使用一个适当频率的外部时钟，这对于 MIDI 波特率来说是一个例外，通过改变外部晶振的频率，可达到要求的波特率，参考波特率列表，注意到使用具有标准晶振频率的 38.4k 波特率，计算出的除数为 96，如果将晶振频率改变到 3 MHz，并使除数为 96，则 MIDI 的 31.25k 波特率就能在内部生成，同时注意到，将晶振频率改变到 3 MHz 并不违背最小/最大时钟技术规范（2 到 4 MHz），因此在其余的计时技术规范中不会出现任何问题，为了实施这个时钟方案，需要编程时钟选择寄存器（CSR = CCH）和辅助控制寄存器（ACR[7]

```

BEGIN
;
; 2681多点或2691唤醒模式测试线程
; 通道 'A' TXD输出通过连接线与通道 'B' RXD输入相连.
;
MR1A EQU $7F001
MR1B EQU $7F011
MR2A EQU $7F001
MR2B EQU $7F011
SRA EQU $7F003
SRB EQU $7F013
CSRA EQU $7F003
CSRB EQU $7F013
CRA EQU $7F005
CRB EQU $7F015
RHRA EQU $7F007
RHRB EQU $7F017
THRA EQU $7F007
THRB EQU $7F017
ACR EQU $7F009
ISR EQU $7F00B
IMR EQU $7F00B
CTUA EQU $7F00D
CTUR EQU $7F00D
CTL EQU $7F00F
CTLR EQU $7F00F
RELAY EQU $7FC001
;
INIT:  MOVE.B #$3A, CRA ; RXT TXA
        MOVE.B #$2A, CRB ; RST RXB
        MOVE.B #$1F, MR1A ; 多点, 8位, A/D = 1
        MOVE.B #$1B, MR1B ; 多点, 8位, A/D = 0
        MOVE.B #07, MR2A ; 正常, 停止 = 1
        MOVE.B #07, MR2B
        MOVE.B #$66, CSRA ; TXC = RXC = 1200波特
        MOVE.B #$66, CSRB
        MOVE.B SRB, D1 ; 保存信道B状态
        MOVE.B #06, CRA ; 启用TX信道A
CHK0:  MOVE.B SRA, D0 ; 读取信道A状态
        BTST #03, D0 ; 是TXEMT?
        BEQ CHK0 ; 等到TXEMT = 1
        MOVE.B #$0AA, THRA ; 将地址载入到THRA中
CHK1:  MOVE.B SRA, D0
        BTST #03, D0 ; 是TXEMT=1?
        BEQ CHK1 ; 等到TX变空
        MOVE.B #$0A, CRA ; 禁用TX A
CHK2:  MOVE.B SRB, D2 ; RXRDY=1信道B?
        BTST #0, D2 ; 查询直到RXRDYB=1
        BEQ CHK2
        MOVE.B RHRB, D7 ; 禁用RHR信道B
        CMPI.B #$0AA, D7 ; 信道B是否接收到地址?
        BEQ DATCHK ; 是, 继续, 否则停止
        TRAP #15
DATCHK: MOVE.B #$09, CRB ; 启用信道B TX
        MOVE.B #$1A, CRA ; 复位信道A MR指针
        MOVE.B #$1B, MR1A ; 重写MR1A (A/D = 0)
        MOVE.B #$3A, CRA ; 复位TX A
        MOVE.B #06, CRA ; 启用TX A
        MOVE.B #$55, THRA ; 发送信息 (数据字节 = 55)
ENDCHK: MOVE.B SRB, D0
        BTST #0, D0 ; 等到RXRDY = 1 (信道B)
        BEQ ENDCHK
        MOVE.B RHRB, D7 ; 保存数据
        TRAP #15
    
```

SD00665

图1. SCN2681多点或SCC2691唤醒模式

1.3.10 通过模式选择来去除硬件

正如以上提到的，MIDI 技术规范包括一个可选的‘连通’接头，用以给‘菊花链’下游的其它 MIDI 装置提供接收数据流的直接备份。通过利用 SCC2691 的自动回应特点，就能去除‘连通’接头所需的附加硬件，在自动回应模式中，接收到的数据被将重新产生时钟，并使用接收器时钟来重新传输到 TxD 输出上，接收器和 CPU 之间的通信正常继续进行，但 CPU 到发送器的链路被禁用，为了调用这种工作模式，设定模式寄存器 2 (MR2 [7: 6] = 01)。在从自动回应模式切换到普通模式时，如果取消选定的动作在接收器检采样停止位且发送器刚好使能之后马上发生，发送器将保持在自动回应模式，直到停止位被发送为止。请注意：在自动回应模式中，没有必要使能发送器。

1.3.11 唤醒模式

SCC2691 唤醒模式提供了通过地址帧识别来自动唤醒接收器，在这种模式中，主站发送一个地址字符，后面跟着要发送到副站的数据，在普通情况下从站的接收器是禁用的，它将检验接收到的数据流，并在收到一个地址字符时唤醒 CPU，SCC2691 数据表描述了这个过程。

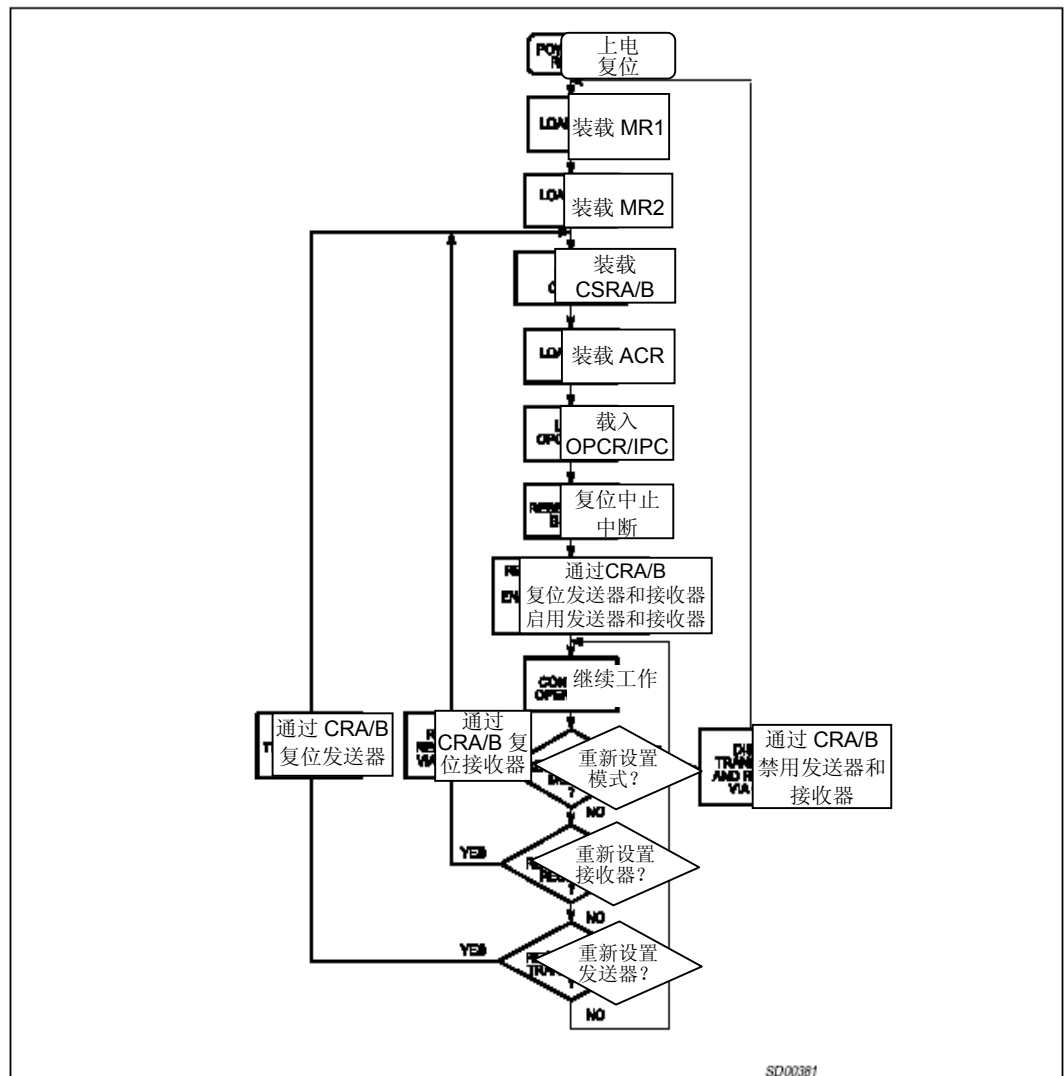


图2. 异步初始化

```

;
; 2681/2691 功能程序
; 本程序通过读回MR1中的值来验证数据总线是否完好,
; 如果第一个测试通过, 则通过将256个字符写入到发送器 (A & B),
; 并读取接收器 (A & B) 的内容来验证, 通过这种方式来测试信道 'A', 然后是 'B'。
;
INIT:  MOVE.B #$1A, CRA
        MOVE.B  #$30, CRA    ; 复位TX
        MOVE.B  #$20, CRA    ; 复位RX
        MOVE.B  #$13, MR1A   ; 无奇偶性, 8位
        MOVE.B  #7, MR2A     ; 正常, 停止 = 1
        MOVE.B  #$66, CSRA   ; TXC = RXC = 1200波特
        MOVE.B  #$10, CRA    ; 复位指针, 禁用TX-RX
        MOVE.B  MR1A, D1     ; 读取MR1A
        MOVE.B  #$13, D0
        CMP.B   D0, D1      ; 比较数据值
        BEQ    TEST1       ; 如果匹配则数据总线完好
        TRAP   #15         ; 如果失败则停止
;
; 这次测试发送数据FF通过00 THRA, 然后将它们读回。
; RX数据只与通道 'A' 的TX数据进行比较。
; 注意: 对于后面的两个测试, 用继电器将TX短接到RX上。
;
TEST1:  MOVE.B #1, RELAY    ; 短接TX到RX, CTS到RTS
        MOVE.B  #$50, CRA   ; 复位中止状态
        MOVE.B  #$20, CRA   ; 复位接收器
        MOVE.B  #$30, CRA   ; 复位发送器
        MOVE.B  #$45, CRA   ; 清除错误, 启用TX-RX
        MOVE.W  #$100, D7   ; 设置第一个发送字符
TEST1A: SUBI.B #1, D7       ; D7减1, 直到D7 = 0
        BEQ    RELAY2      ; 如果D7 = 0, 到下一个测试
        MOVE.B  D7, THRA   ; 发送下一个字符到发送器
WAIT1:  BTST   #0, SRA     ; 接收器准备好?
        BEQ    WAIT1
        MOVE.B  RHRA, D1   ; 获取接收到的字符
        CMP.B   D7, D1    ; 发送的字符 = 接收到的字符?
        BEQ    TEST1A     ; 如果是, 重复运行
        TRAP   #15        ; 如果失败, 则停止
;
; 这个测试利用测试1中检查信道 'A' 的相同方法来检查信道 'B'。
; 只有数据格式已经发生变化。(由于2691没有地址A2,
; 这个测试是对串行信道的一次简单的重复测试)。
;
TEST2:  MOVE.B #$1A, CRB   ; 禁用TX-RX, 复位MR指针
        MOVE.B  #7, MRB1   ; 奇校验, 8位
        MOVE.B  #$0F, MR2B  ; 普通, 2个停止位
        MOVE.B  #$0BB, CSRB ; 9600波特
        MOVE.B  #$50, CRB   ; 复位中止状态
        MOVE.B  #$30, CRB   ; 复位TX
        MOVE.B  #$20, CRB   ; 复位RX
        MOVE.B  #$45, CRB   ; 清除错误, 启用TX & RX
        MOVE.W  #$100, D7
TEST2A: SUBI.B #1, D7       ; D7减1
        BEQ    STOPIT     ; 如果D7 = 0, 停止发送字符
        MOVE.B  D7, THRB  ; 如果D7 > 0, 写TX保持寄存器
WAIT2:  BTST   #0, SRB     ; 测试RX是否准备好
        BEQ    WAIT2     ; 如果未准备就绪, 循环
        MOVE.B  RHRB, D1  ; 收取接收到的字符
        CMP.B   D7, D1    ; 比较TX字符与RX字符
        BEQ    TEST2A    ; 如果比较匹配, 再次发送
STOPIT: TRAP   #15        ; 结束线程
        END    INIT
    
```

SD00666

图3. SCN2681/SCC2691功能程序

对于MIDI应用，这个模式以下列方式来起作用，MIDI数据类型划分为状态字节和数据字节，接收到数据的MSB (D7) 确定是收到一个状态字节 (D7 = 1) 还是一个数据字节 (D7 = 0)，相对于传输的8个数据位，通过将接收器编程为7个数据位，第8个接收到的位就能被解释为地址/数据 (A/D) 位，从而指示了一个状态字节发送或数据字节发送。

如果接收器被禁用，则一个状态字节的发送 (A/D = D7 = 1) 将置位RxRDY状态位，并将字符载入到接收寄存器FIFO中，这个RxRDY状态能够被编程输出到多用途输出引脚上从而“唤醒”CPU，根据状态信息，CPU将忽略 (Rx禁用) 或响应 (Rx启用) 下来跟随的数据字节。

例如，如果MIDI分配的通道为5，则由MIDI主机生成的给通道5的一个状态字节 (D0 - D3, 可能是音频信息) 可能如图4所示。随着A/D位被置位，接收器翻译地址 (状态)，载入数据到RHR FIFO中，并在状态寄存器 (SR) 中直位RxRDY，这个RxRDY位能完成对CPU的唤醒功能，然后CPU将检查数据 (状态字节) 以确定是否是接收器 (通道) 被寻址，如果是，如上例所示，CPU然后将启用接收器以接收进来的数据字节，一旦进入的数据字节接收完毕，那么CPU将禁用接收器，并继续处理其它功能直到接收器再次指示接收到一个状态时为止。

### 1.4 系统互连和其它注意事项

SCC2691的总线电路十分灵活，可容易地接入到现有的任何微处理器或微控制器上，CE和RDN/WRN线路在内部进行逻辑“或”处理以允许RDN、WRN、或CE之任何一个线路来激发一次数据传输。上次生效的信号启动周期，最先失效的信号结束周期，图5说明了一个全CMOS的MIDI接口 (不同于光耦合器)，它使用了一个80C49、HC逻辑和SCC2691，这个电路在全速下的电流消耗小于50mA，在备用模式下小于1mA，通过使用飞利浦的SMD封装，电路板空间可得到有效地减小，从而提供了一个紧凑的、低功率的MIDI接口。



图4. MIDI主机生成的给通道5的状态字节



## 2. 否认声明

**生命保障**——这些产品在设计时并没有考虑到可以用于生命保障器具、装置、或系统；在此类场合，这些产品的故障能够明显地导致人员伤亡。对于使用或销售这些产品的飞利浦半导体公司的用户，如果他们想在此类应用中使用这些产品，则他们必须自行承担风险，并同意在由于此类应用而导致任何损坏时全额向飞利浦半导体公司进行赔偿。

**进行修改的权利**——飞利浦半导体公司保留对此处描述或包含的产品进行修改的权利，其中包括电路、标准单元、和/或软件，以便能够改善产品的设计和/或性能。当产品已经投入批量生产时（状态“生产”），有关的修改将会通过一个《用户产品/过程修改通知书（CPCN）》进行公告。如未另行规定，飞利浦半导体公司不会对任何一个这些产品的使用承担任何责任或义务，不向这些产品转让任何受专利、版权、或掩码著作权保护的许可权或所有权，也不会做出任何表述或担保、说明这些产品没有侵犯任何专利、版权、或掩码著作权。

**应用信息**——对于任何一件此类产品，此处描述的应用情况仅仅是为了演示性目的。在没有进行进一步的试验或变更之前，飞利浦半导体公司并没有做出任何表示或担保，声明此类应用将会适应于特定的用途。

## 3. 许可

### 飞利浦 I<sup>2</sup>C 零件的购买



飞利浦 I<sup>2</sup>C 零件的购买转让一个飞利浦 I<sup>2</sup>C 专利保护的许可在 I<sup>2</sup>C 系统中使用零件从而与飞利浦制定的规范一致。这个规范可以用代码 9398 393 40011 命令。

### 飞利浦 RC5 零件的购买

飞利浦 RC5 零件的购买转让一个飞利浦 RC5 专利保护的许可在 RC5 系统中使用零件从而与飞利浦制定的详细的控制命令 RC5 标准 UATM-5000 的分配规范一致。

## 4. 专利

同此通告主要器件使用一个或多个下列专利每个专利可能就其它权限有相应的专利。

<专利号> — <专利所有者>

## 5. 商标

<商标名称> — 使所有者的商标

<注册商标名称> — 是被所有者注册的商标

## 6. 目录

<b>1.</b>	<b>概述.....</b>	<b>3</b>
1.1	介绍.....	3
1.2	DUART/UART 共有的功能.....	3
1.2.1	读取保留的寄存器.....	3
1.2.2	接收器 FIFO.....	3
1.2.3	检测中止的结束.....	3
1.2.4	在一个短帧之后禁用发送器.....	4
1.2.5	防止发送器和/或接收器的出错.....	4
1.2.6	设置计数器/计时器作为一个计时器.....	5
1.2.7	多点/唤醒模式.....	5
1.2.8	处理中断.....	6
1.2.9	从外部驱动 X1 或使用一个晶振.....	6
1.2.10	X1 / X2 晶振.....	6
1.2.11	普通初始化.....	6
1.2.12	异步诊断.....	6
1.3	SCN2681 / 68681 的独特功能.....	7
1.3.1	异常的数据中止.....	7
1.3.2	RTS / CTS 功能.....	7
1.3.3	输入端口.....	7
1.3.4	输出端口.....	8
1.3.5	SCN2681 总线接口.....	8
1.3.6	SCN68681 总线接口.....	8
1.3.7	使用 SCC2691 MIDI 接口.....	8
1.3.8	硬件接口和数据格式.....	9
1.3.9	SCC2691 波特率选择.....	9
1.3.10	通过模式选择来去除硬件.....	11
1.3.11	唤醒模式.....	11
1.4	系统互连和其它注意事项.....	13
<b>2.</b>	<b>否认声明.....</b>	<b>15</b>
<b>3.</b>	<b>许可.....</b>	<b>15</b>
<b>4.</b>	<b>专利.....</b>	<b>15</b>
<b>5.</b>	<b>商标.....</b>	<b>15</b>
<b>6.</b>	<b>目录.....</b>	<b>16</b>