使用 STM8 Nucleo-64 板与终端进行 RS232 通信

引言

本应用笔记说明了如何从运行于 PC 上的终端窗口控制 STM8 Nucleo-64 板。PC 要通过 RS232 线连到 STM8S208RBT6（对于 NUCLEO-8S208RB）或 STM8L152R8T6（对于 NUCLEO-8L152R8）板的 UART。

当为开发板添加了所需的元器件并下载应用软件后，用户将能够使用终端来管理 STM8S 系列或 STM8L 系列 GPIO 和 TIM3 定时器，以及配置蜂鸣器输出。

表 1. 适用产品

类型	产品编号
评估板	NUCLEO-8S208RB
	NUCLEO-8L152R8

参考文档

- *STM8 Nucleo-64 板*数据概述（DB3591）
- *STM8L152R8T6 Nucleo-64 板*用户手册（UM2351）
- *STM8S208RBT6 Nucleo-64 板*用户手册（UM2364）

1 先决条件

运行 STM8 Nucleo-64 板终端演示应用所需的材料如下：

- 运行于 PC 上的终端窗口：终端模拟器软件可以是 Windows HyperTerminal（请参见第 附录 B 节 附件 B）、TeraTerm Pro 或任何终端软件。
- RS232 零调制解调器线（收发线交叉连接）。

2 配置 NUCLEO-8S208RB 板

运行应用之前，必须配置 NUCLEO-8S208RB 板（基于 STM8S208RBT6 器件构建）以启用蜂鸣器输出。蜂鸣器输出为 STM8S208RBT6 复用功能。通过将 OPT2 选项字节中的复用功能重映射选项位 AFR7 设为‘1’来启用。

提示 *NUCLEO-8L152R8 板（基于 STM8L152R8T6 器件构建）不需要用户检查或激活该复用功能或蜂鸣器。*

3 应用描述

3.1 硬件要求

本应用使用 STM8 Nucleo-64 板的板上 LED（LD2）及相关电阻（R1）。
应用所需的外部无源元件列表如下。

表 2. 无源元件列表

元件描述	值
B1 蜂鸣器	-
C1、C2、C3、C4、C5 电容	100 nF
DB9 连接器	-

该应用还使用一个 5 V 供电的 ST232B RS232 驱动器/接收器（请参见下表以获取更详细信息）。这一额外元件很关键，因为 PC 的 COM 口工作于 12 V 标称供电。这和 STM8S 系列或 STM8L 系列设备 UART 输入/输出并不兼容，因为它们工作于 5 V。此元件使用 SO16 封装，与 STM8 Nucleo-64 板匹配。若需 ST232B 的更多信息，请参阅 ST232B 数据手册。

表 3. 封装元件列表

部件名称	元件描述	封装
ST232B	超高速度超低功耗 5 V RS232 驱动器/接收器，用于 UART 5/12 V 电平转换器	SO16

3.2 应用原理图

下图显示了应用电气原理图。
若 RS232 线不是零调制解调器线（发送和接收线没有交叉连接），则将 U1 引脚 14 连至引脚 2 的 DB9，将 U1 引脚 13 连至引脚 3 的 DB9。

图 1. **STM8S 系列应用原理图**

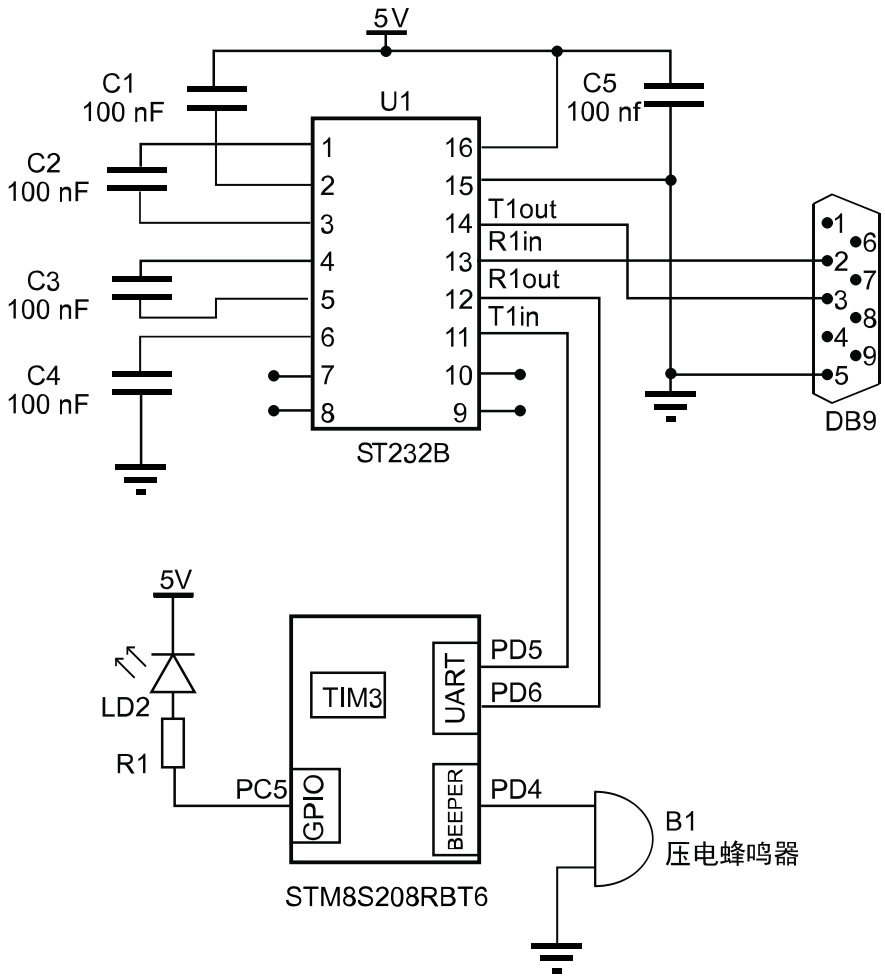
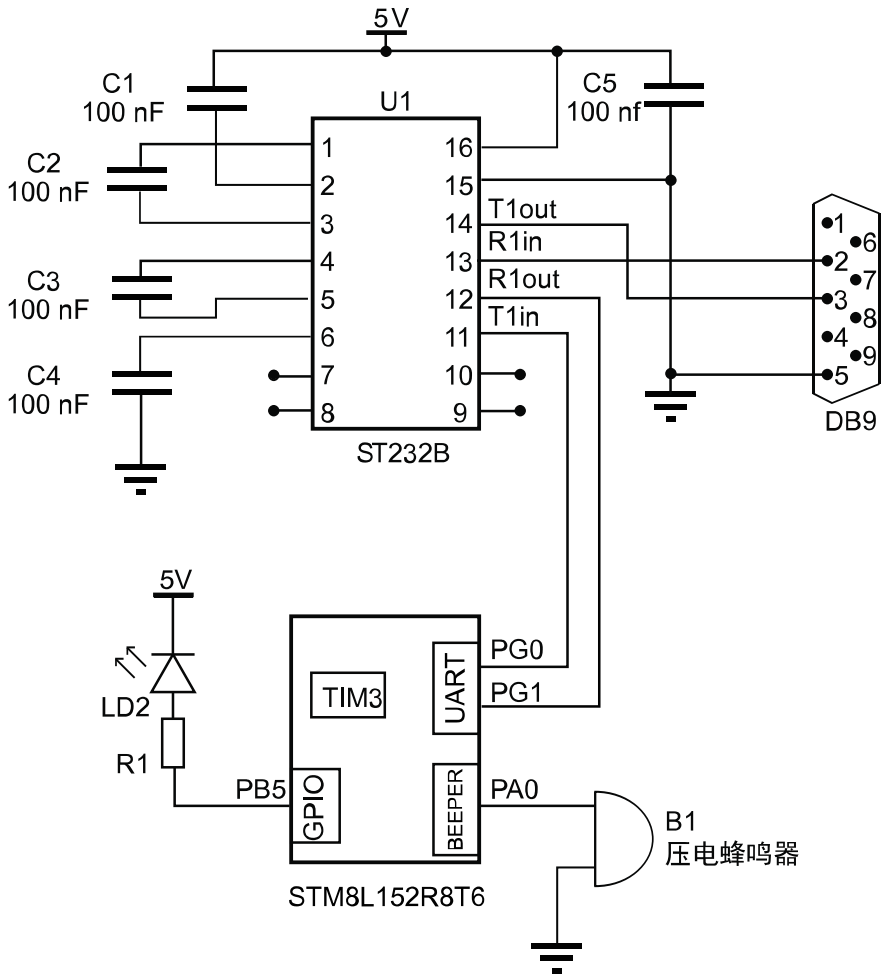


图 2. **STM8L 系列应用原理图**



3.3 应用原理

本应用在 STM8S208RBT6 或 STM8L152R8T6 微控制器和运行于 PC 上的终端窗口之间建立一个标准通信接口。使用 RS232 协议通过 STM8 设备 UART 执行通信。终端窗口和 UART 必须具有相同配置（请参见第 附录 B 节 附录 B）。

本文仅说明了 STM8 Nucleo-64 板 UART 侧的通信和数据处理。若需 Windows HyperTerminal 或类似软件的更多信息，请参考 Microsoft®帮助或供应商网页。

3.3.1 运行应用

为了运行应用，请执行以下步骤：

1. 在您的 PC 上启动并配置一个终端窗口（请参见第 附录 B 节 附录 B 以获取使用 Windows HyperTerminal 的例子）。
2. 使用 ST Visual Develop（STVD）或其它工具编译并运行应用固件。
3. 通过 RS232 线将您的 PC 连至 STM8 Nucleo-64。
4. 当应用启动时，会在 Windows HyperTerminal 上显示一个菜单。此菜单允许用户：
   - 开启或关闭 LD2。
   - 将 LD2 激活为闪烁模式。
   - 启用/禁用蜂鸣器，并选择蜂鸣器频率。

   此菜单上显示的所有信息都由 STM8S 系列或 STM8L 系列微控制器发送。当在 HyperTerminal 上按下一个键时，相应的 ASCII 值会发送到微控制器并解码。

3.3.2 STM8 Nucleo-64 板和终端之间的通信序列

1. STM8S 系列或 STM8L 系列微控制器向 PC 终端模拟器软件发送字符串”Enter your choice”。
2. 终端显示字符串”Enter your choice”。
3. 用户在键盘上按”2”键。
4. 终端模拟器软件将相应的 ASCII 码（0x52）发送回微控制器（请参见第 附录 A 节 附录 A）。
5. 微控制器解码收到的数据，将代码 0x52 发送回去，显示在终端上，并在存储器中储存”2”值。
6. 终端模拟器软件接收到代码 0x52 并显示”2”。
7. 用户按下”回车”键。
8. 终端模拟器软件将回车相应的代码 0x0D 发送回去（请参见第 附录 A 节 附录 A）。
9. STM8S 系列或 STM8L 系列微控制器解码接收到的数据，将代码 0x0D 发送回去，显示在终端上，然后执行相关的选项 2 行为。

4 软件说明

4.1 应用使用的 STM8S 系列和 STM8L 系列外设

本应用例子使用 STM8S 系列和 STM8L 系列标准固件库来驱动通用的外设功能。本应用使用以下外设：

- UART3（STM8S 系列）或 USART3（STM8L 系列）：它用于和运行在 PC 上的终端窗口通信，必须如下配置：
  - 波特率 = 9600 波特
  - 字长度 = 8 位
  - 1 个停止位
  - 无奇偶校验
  - 启用接收和发送

提示 *若使用了 STM8L 系列，则必须禁用 USART3 时钟。*

通过轮询每个接收和发射操作进行通信。

提示 *终端窗口和 STM8 设备 UART 外设必须配置为相同波特率、字长、停止位数以及奇偶校验。*

- Timer3（TIM3）：TIM3 定时器配置为时间基准，并使能中断以控制 LD2 的闪烁速度。
- GPIO：GPIO 用于 MCU 与外部硬件电路的接口。STM8 系列的 PC5 端口和 STM8L 系列的 PB5 端口配置为推挽输出低模式以驱动 LD2。
- BEEPER：若需驱动蜂鸣器，BEEPER 外设需在 BEEP 输出引脚上输出 1、2 或 4 KHz 的信号。

4.2 STM8 标准固件库配置

4.2.1 STM8S 系列标准固件库

STM8S 系列标准固件库的 *stm8s_conf.h* 文件，可通过启用应用所使用的外设函数，配置该库。
必须存在下面的定义语句：

- `#define _GPIO 1` 启用 GPIO
- `#define _TIM3 1` 启用 TIM3
- `#define _BEEPER 1` 启用 BEEPER
- `#define _UART3 1` 启用 UART3

4.2.2 STM8L 系列标准固件库

STM8L 系列标准固件库的 *stm8l_conf.h* 文件，可通过启用应用所使用的外设函数，配置该库。
必须存在下面的定义语句：

- `#include "stm8l15x_gpio.h"`
- `#include "stm8l15x_tim2.h"`
- `#include "stm8l15x_tim3.h"`
- `#include "stm8l15x_beep.h"`
- `#include "stm8l15x_usart.h"`

4.3 应用软件流程图

本章说明了应用软件主循环和控制从/向终端窗口接收/发送数据的函数：

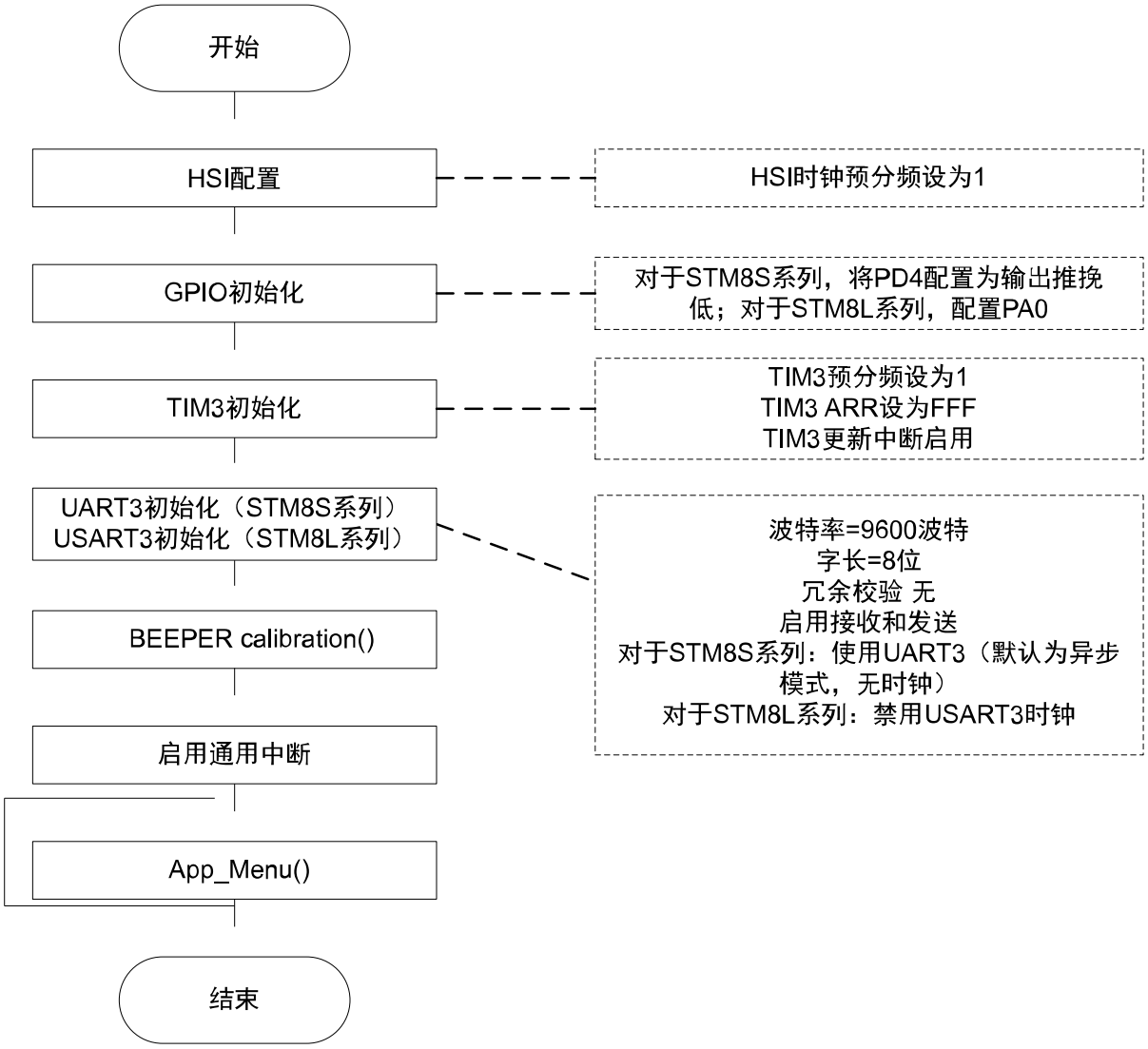
- App_Menu
  此函数用于在终端上显示一个菜单，以及管理用户输入的信息。
- SerialPutString
  此函数用于向终端发送一个字符串。
- SerialPutChar
  此函数用于向终端发送一个字符。

- GetInputString
  此函数用于从终端接收一个字符串。
- GetIntegerInput
  此函数用于从终端接收一个整数。
- Get_Key
  当按一个键时，此函数返回相应的十六进制代码。

4.3.1 应用主程序

应用主程序配置 STM8S 系列或 STM8L 系列外设，启用应用使用的所有标准中断。当完成初始化时，主程序在终端窗口上显示应用菜单。

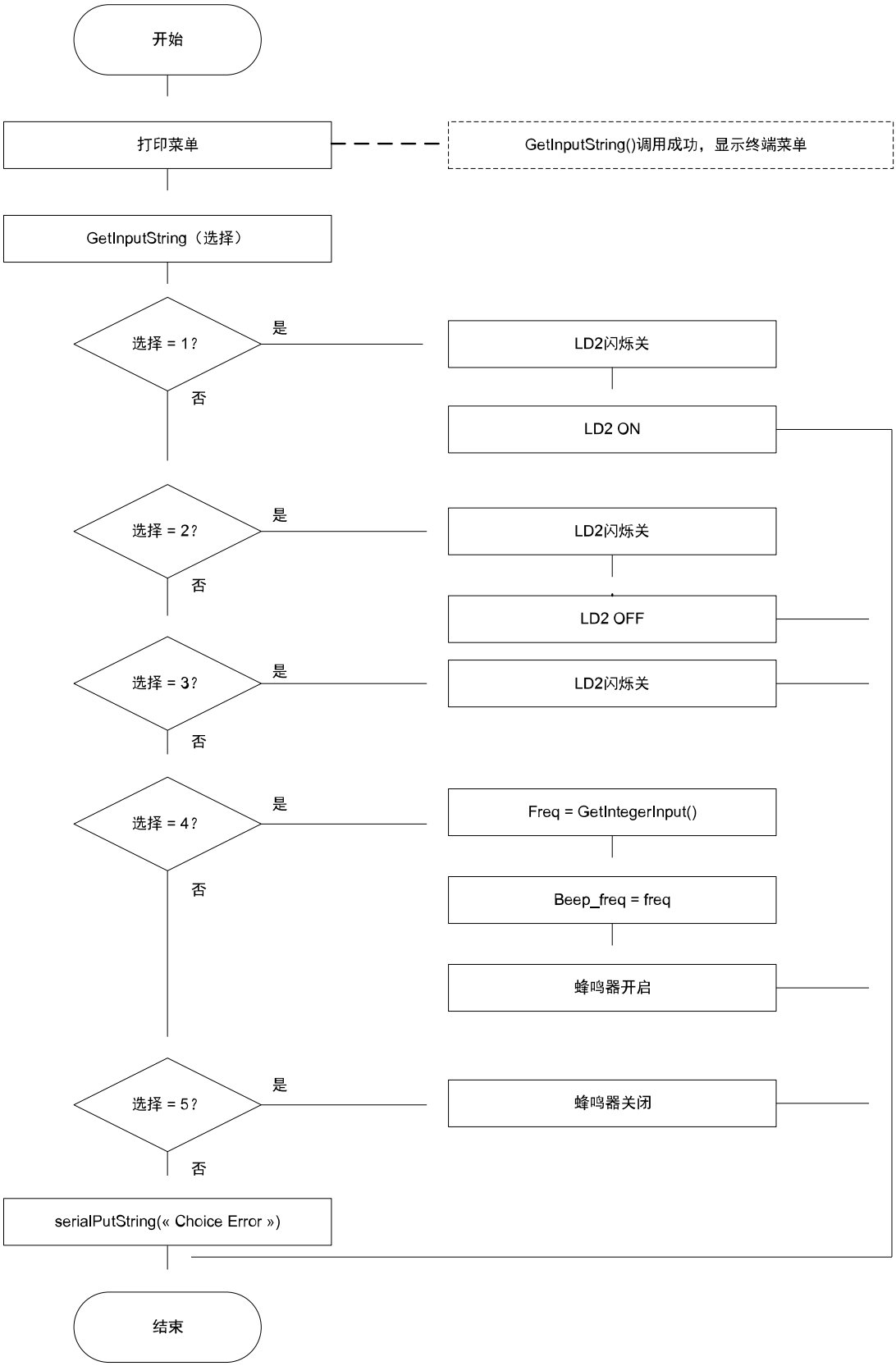
图 3. 主程序流程图


开始
HSI配置
HSI时钟预分频设为1
GPIO初始化
对于STM8S系列，将PD4配置为输出推挽低；对于STM8L系列，配置PA0
TIM3初始化
TIM3预分频设为1
TIM3 ARR设为FFF
TIM3更新中断启用
UART3初始化（STM8S系列）
USART3初始化（STM8L系列）
波特率=9600波特
字长=8位
冗余校验 无
启用接收和发送
对于STM8S系列：使用UART3（默认为异步模式，无时钟）
对于STM8L系列：禁用USART3时钟
BEEPER calibration()
启用通用中断
App_Menu()
结束

4.3.2 App_menu 函数

App_menu 函数为主应用程序。它在终端上显示菜单，通过该菜单可配置 GPIO、TIM3 和 BEEPER。App_menu 调用 GetInputString、GetIntegerInput 和 SerialPutString，以通过 RS232 接口发送和接收数据。

图 4. App_menu 流程图



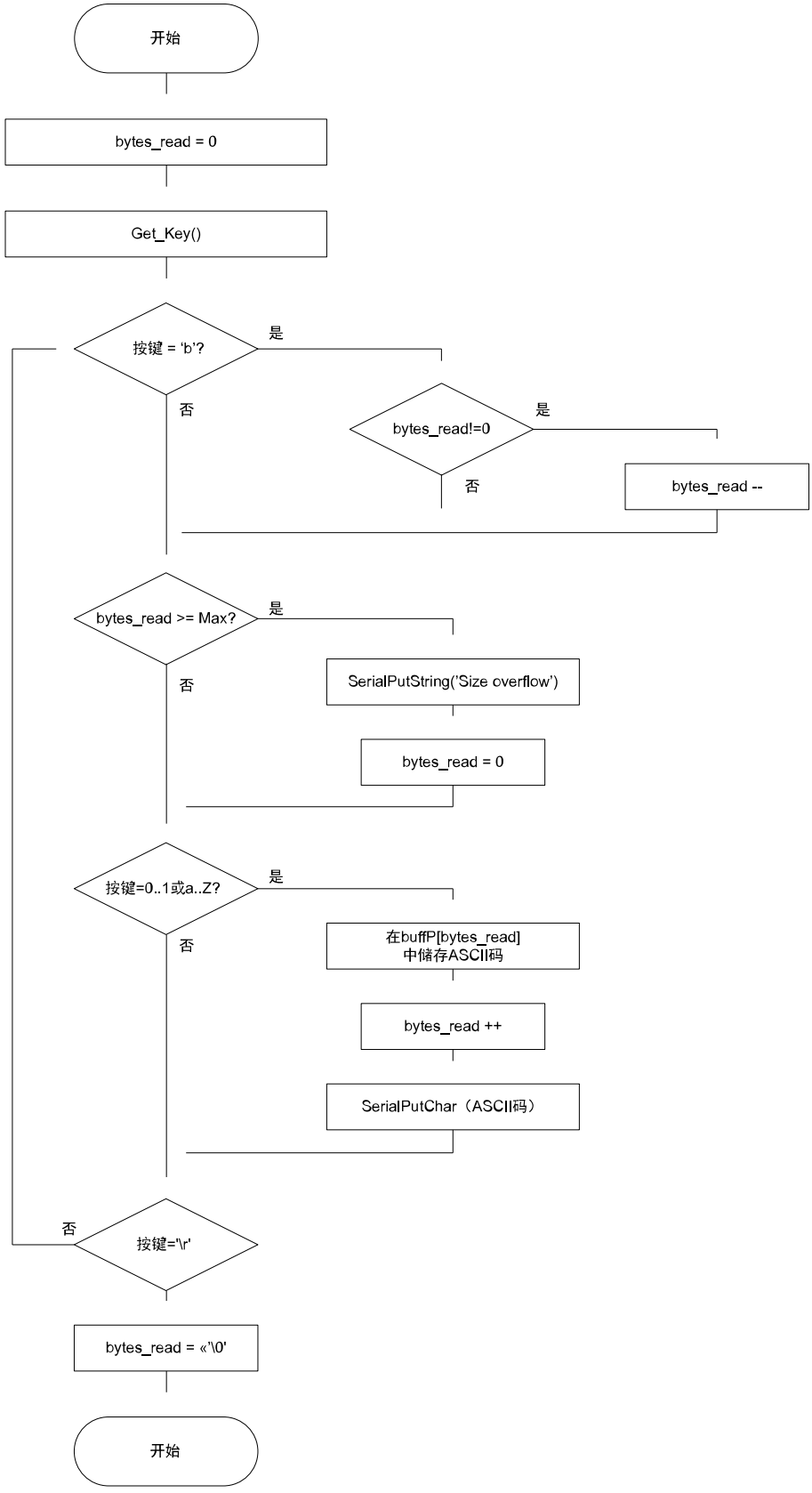
4.3.3 GetInputString 功能

GetInputString 函数用于接收并存储终端窗口发送的字符串。此函数依赖于 Get_key 函数来捕获并解码每个字符（请参见特定章节）。可根据字符的 ASCII 代码值执行不同的行为：

- 若 ASCII 代码 = '\b'
  终端发送退格符。若字符串不为空，则擦除字符串的最后一个字符。
- 若 ASCII 代码属于{0...1 或 a...Z}
  则储存该字符。
- 若 ASCII 代码 = '\r'
  则 GetInputString 函数在字符串末尾储存"字符串结束"值'\0'。
  达到 buffP[bytes_read]缓冲中能够储存的最大 ASCII 代码个数。
  软件擦除记录的字符串，等待终端的另一输入。

若需 ASCII 代码的更多信息，请参考第 附录 A 节 附件 A。

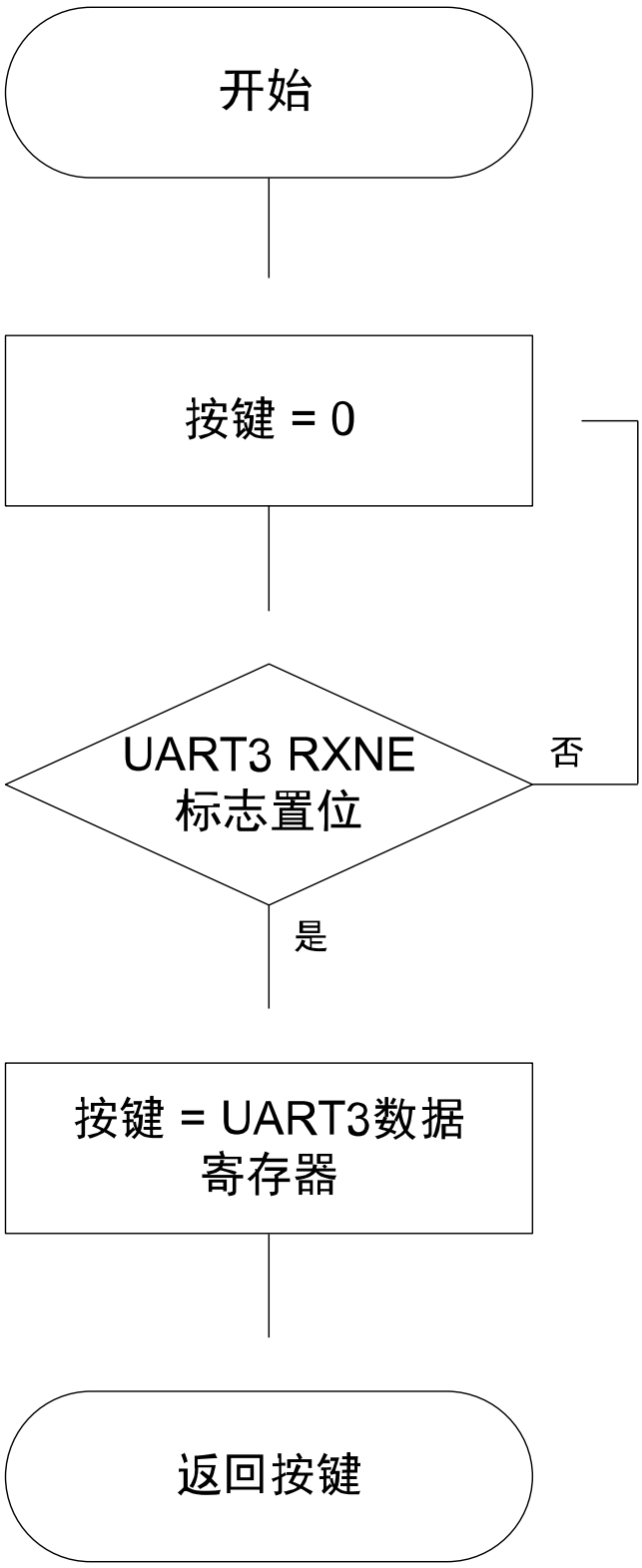
图 5. GetInputstring 流程图



4.3.4 Get_key 函数

Get_key 函数通过轮询 UART RXNE 标志，检测终端上的按键。此函数返回接收的值。

图 6. Get_key 函数流程图



4.3.5 SerialPutString 和 SerialPutChar 函数

SerialPutString 函数用于通过 UART 发送一个字符串。字符串的字符通过 SerialPutChar 函数逐一发送，如下面的流程图中所述。

图 7. SerialPutString 流程图

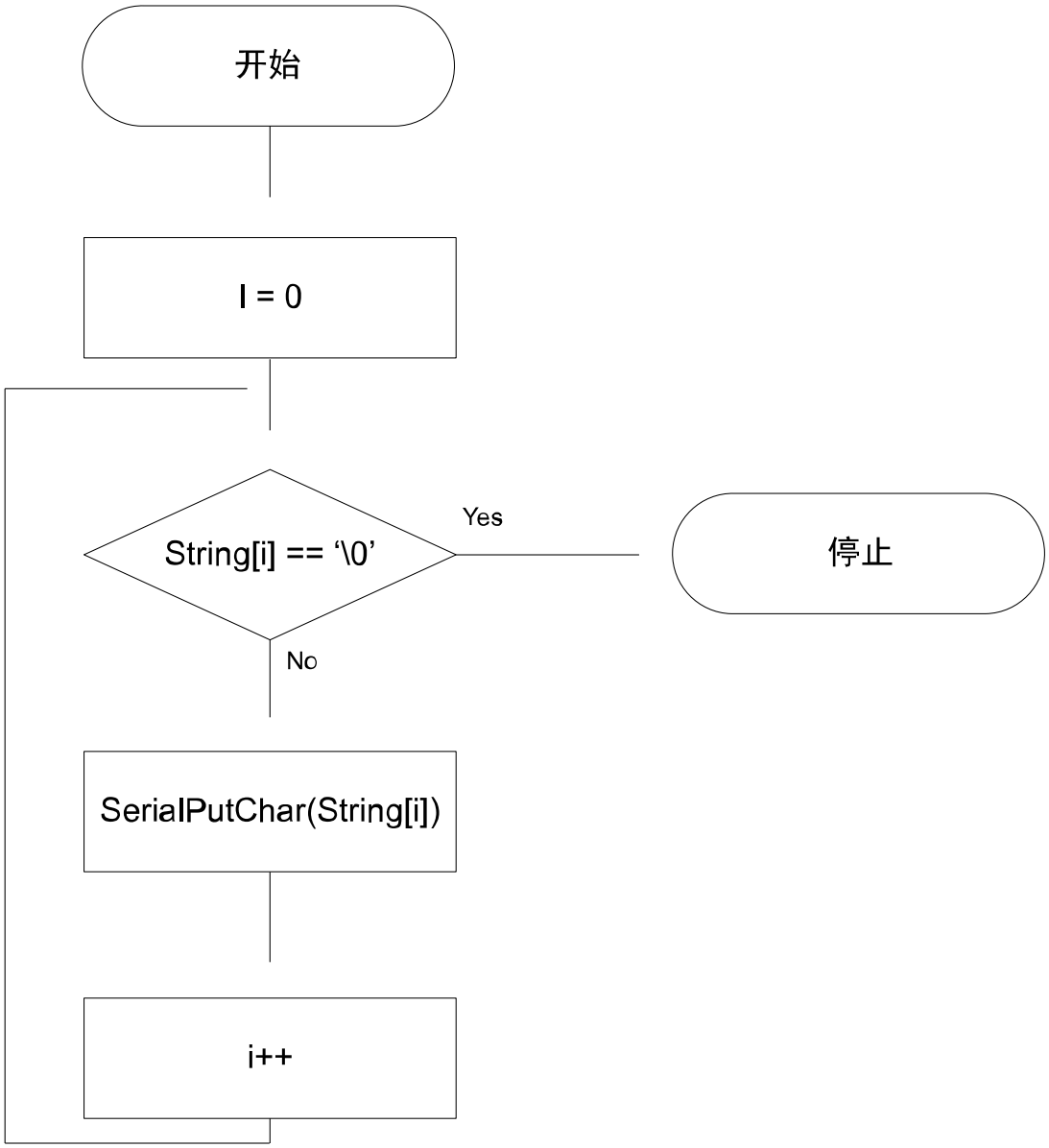
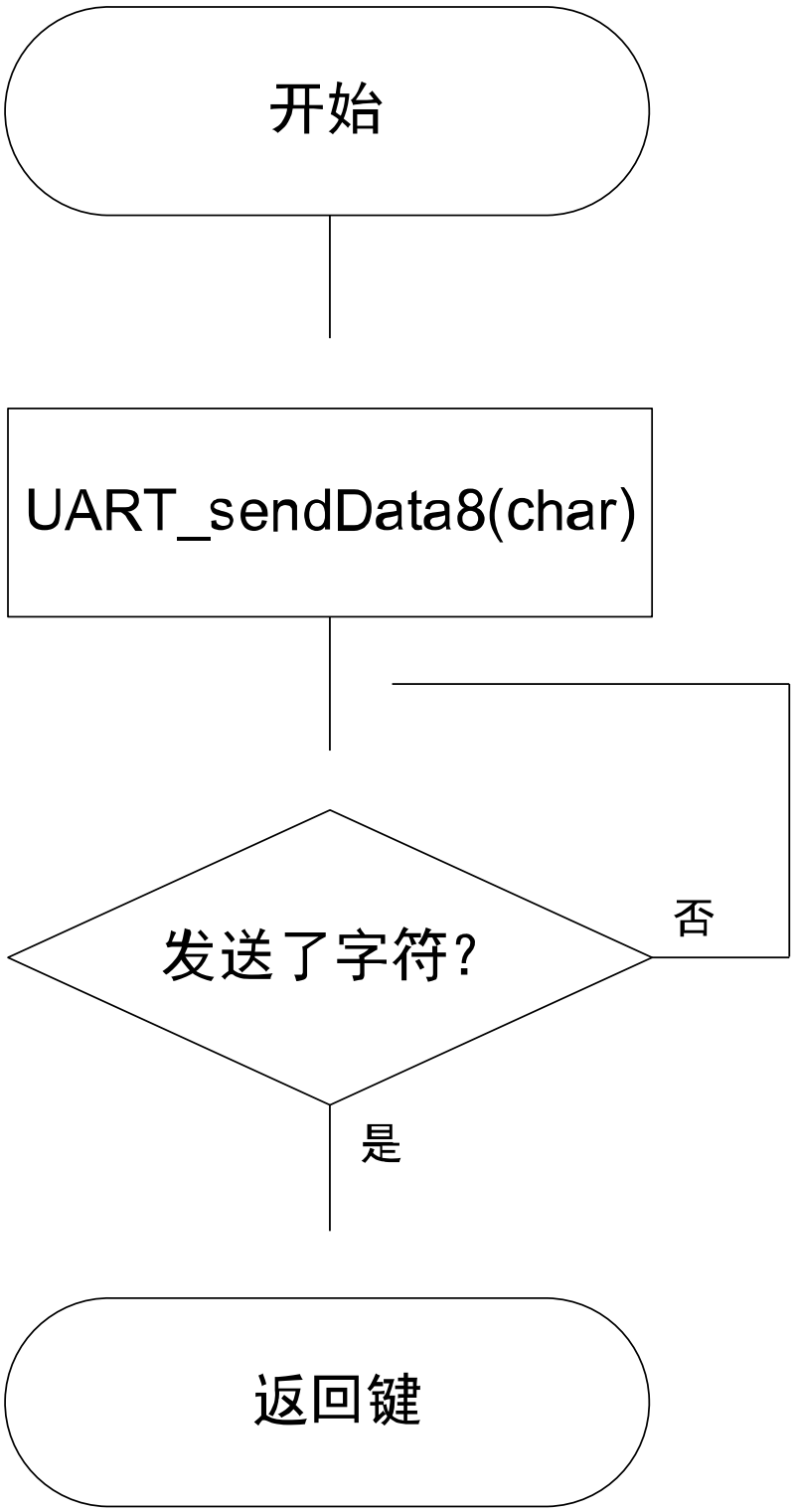


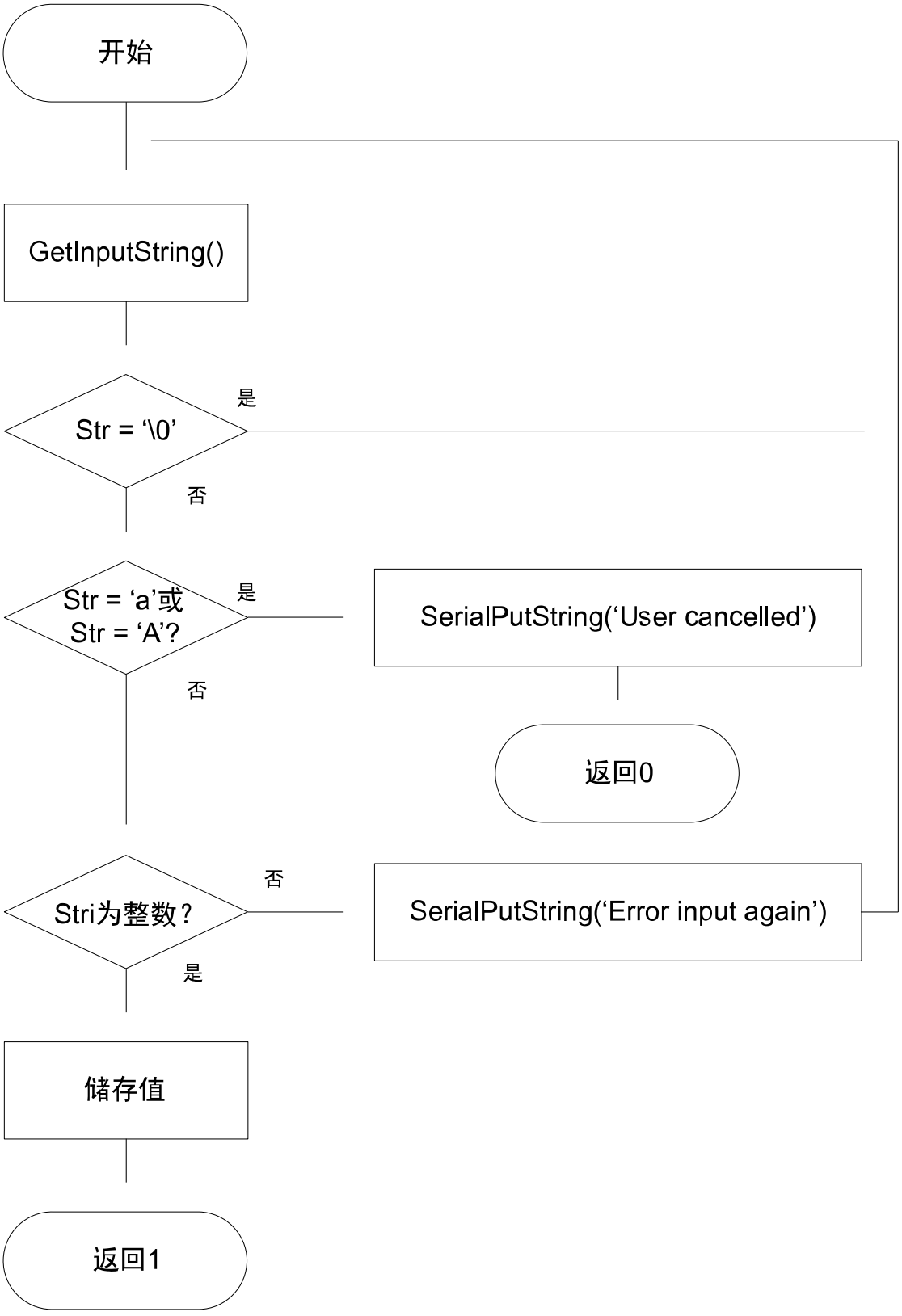
图 8. SerialPutChar 流程图



4.3.6 GetIntegerInput 功能

GetIntegerInput 函数用于检查对应于一个整数的输入数据。若它是整数，则将数据储存在存储器中，否则提示用户输入新数据。

图 9. GetIntegerInput 流程图



附录 A 标准 ASCII 字符代码

表 4. 标准 ASCII 字符代码

Hex	字符	Hex	字符	Hex	字符	Hex	字符
0x00	NULL	0x20	Space	0x40	@	0x60	‘
0x01	头起始	0x21	!	0x41	A	0x61	a
0x02	文本起始	0x22	“	0x42	B	0x62	b
0x03	文本结束	0x23	#	0x43	C	0x63	c
0x04	发送结束	0x24	$	0x44	D	0x64	d
0x05	查询	0x25	%	0x45	E	0x65	e
0x06	ACK	0x26	&	0x46	F	0x66	f
0x07	能听见的铃声	0x27	‘	0x47	G	0x67	g
0x08	退格	0x28	(	0x48	H	0x68	h
0x09	水平 tab	0x29	)	0x49	I	0x69	i
0x0A	换行	0x2A	*	0x4A	J	0x6A	j
0x0B	垂直 tab	0x2B	+	0x4B	K	0x6B	k
0x0C	输入表格	0x2C	,	0x4C	L	0x6C	l
0x0D	回车	0x2D	-	0x4D	M	0x6D	m
0x0E	移出	0x2E	.	0x4E	N	0x6E	n
0x0F	移入	0x2F	/	0x5F	O	0x6F	o
0x10	数据链路退出	0x30	0	0x50	P	0x70	p
0x11	设备控制 1	0x31	1	0x51	Q	0x71	q
0x12	设备控制 2	0x32	2	0x52	R	0x72	r
0x13	设备控制 3	0x33	3	0x53	S	0x73	s
0x14	设备控制 4	0x34	4	0x54	T	0x74	t
0x15	Neg. ACK	0x35	5	0x55	U	0x75	u
0x16	同步空闲	0x36	6	0x56	V	0x76	v
0x17	结束发送块	0x37	7	0x57	W	0x77	w
0x18	取消	0x38	8	0x58	X	0x78	x
0x19	媒体结束	0x39	9	0x59	Y	0x79	y
0x1A	替换	0x3A	:	0x5A	Z	0x7A	z
0x1B	退出	0x3B	;	0x5B	[	0x7B	{
0x1C	文件分隔	0x3C	<	0x5C	\	0x7C	|
0x1D	组分隔	0x3D	=	0x5D	]	0x7D	}
0x1E	记录分隔	0x3E	>	0x5E	^	0x7E	~
0x1F	单元分隔	0x3F	?	0x5F	_	0x7F	<DEL>

附录 B 配置终端窗口

对于所有终端类型，连至 STM8 Nucleo-64 板的终端窗口必须配置如下有效设置：

- 通信端口：COM1 或其它可用端口
- 波特率：9600
- 数据长度：8
- 校验位：无
- 停止位：1
- 流控制：无

为提供准备就绪的应用样例，在项目目录内提供了使用 Windows HyperTerminal 和 COM1 端口的预先配置好的终端。执行项目中的.ht 文件即可方便地将其启动。

用户也能基于 Windows HyperTerminal，使用 STM8 Nucleo-64 板建立一个新连接，并按如下步骤关联到本样例：

1. 打开 Windows HyperTerminal 应用，选择连接名称，例如“MyConnection”，点击 OK 验证。

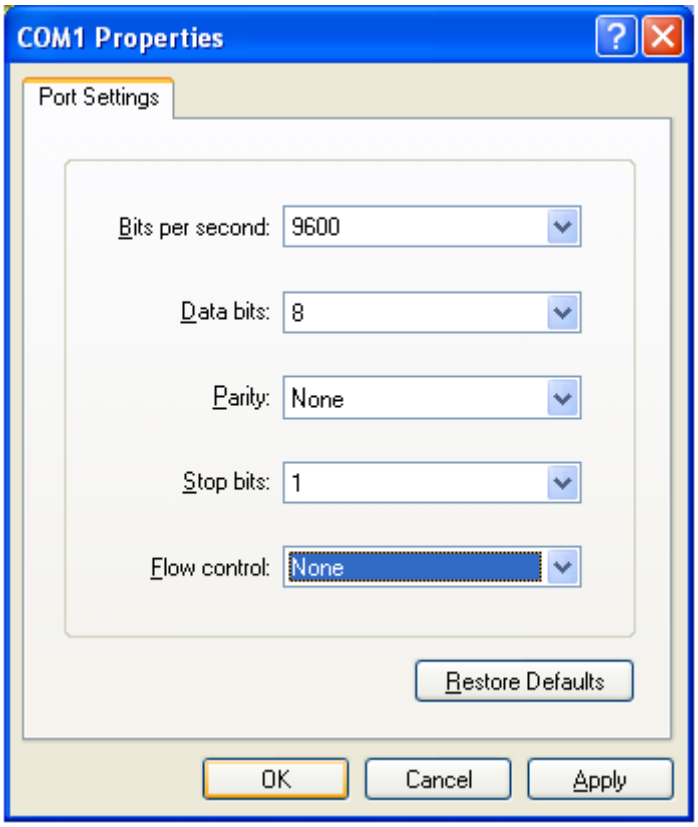
图 10. 启动 Windows HyperTerminal

2. 选择 COM1 或您电脑上的其它可用端口，点击 OK 验证您的选择，其它字段可设为默认值。

图 11. 选择通信端口

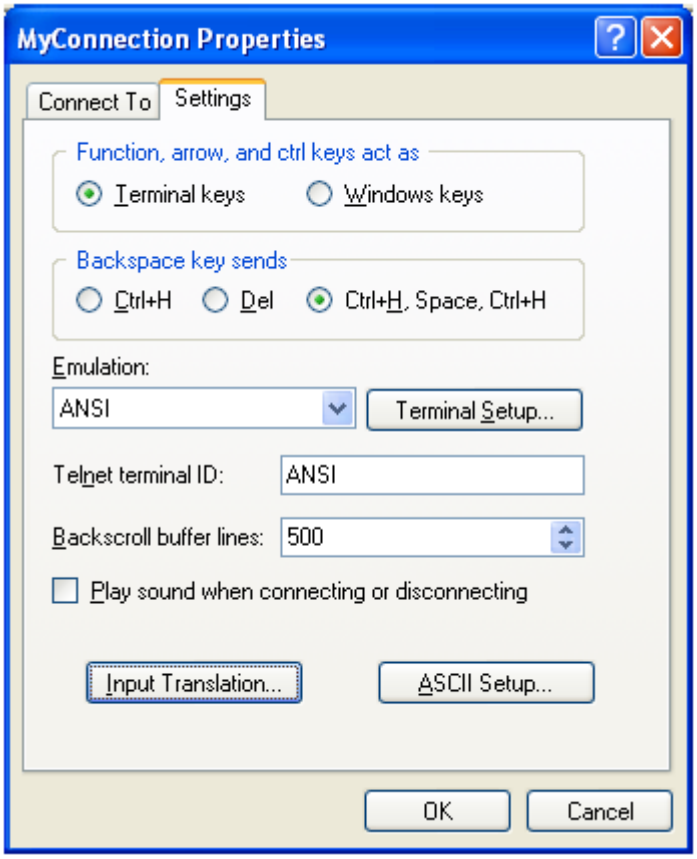
3. 按下图所示配置通信端口属性。Windows HyperTerminal 启动，可以开始通信。

图 12. 配置连接属性



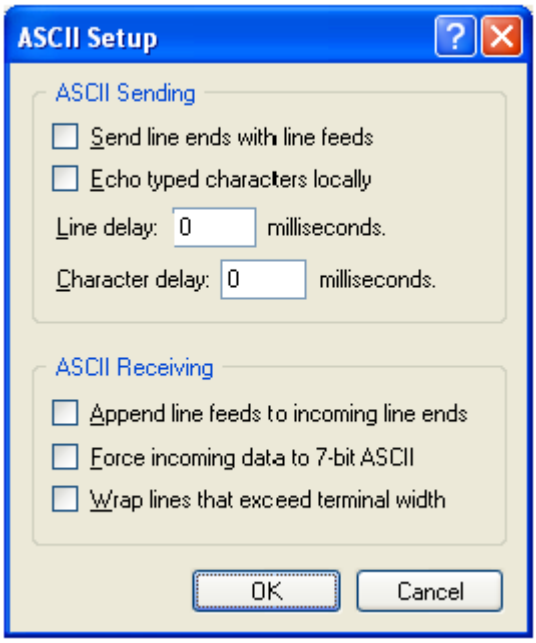
4. 若需检查通信设置：
   – 从 HyperTerminal 主菜单选择 Call > Disconnect，断开 HyperTerminal。
   – 通信结束后，转至 MyConnection Properties 菜单中的 Settings 选项卡。参数设置应如下所示。

图 13. 检查通信设置



   – 最后，点击 MyConnection Properties 菜单中的 ASCII Setup，确保 ASCII 参数与下图中显示的相匹配。

图 14. **ASCII 设置参数**



– 关闭 MyConnection Properties 菜单，从 HyperTerminal 主菜单选择 Call > Call 重新启动通信。现在，STM8 Nucleo-64 应用已经准备好启动了。

版本历史

表 5. 文档版本历史

日期	版本	变更
2018 年 6 月 29 日	1	初始版本。

目录

表一览

图一览