Freescale Semiconductor, Inc.
应用笔记

Document Number: AN5034
Rev. 0, 01/2015

使用 FlexIO 模拟 UART

作者：Pavel Krenek，应用工程师

Freescale Roznov，捷克共和国

1 简介

本应用笔记介绍如何使用通用外设模块FlexIO模拟UART总线。Freescale Kinetis KL43 系列首次引入 FlexIO 外设。

FlexIO 是一个高度可配置模块，能够模拟多种不同的通信协议：UART、 I^2C、 SPI、 I^2S 等。

单独的外设模块 FlexIO 不用于代替 UART 外设，但可用作其他 MCU 外设模块。此外设的一个主要优势是能够让用户在 MCU 中直接构建自己的外设。

此用例为UART模块创建了一个简单的基于独立接收器和发送器的软件驱动。在此演示中，我们使用 Freescale 塔式系统。模拟 UART 总线的最大测试波特率为 115200 波特。

内容

2 FlexIO 外设模块的主要特性

- FlexIO 是指灵活的输入和输出外设。
- 具有多种功能的高度可配置模块。
- 允许模拟标准通信接口。
- 支持多种协议和外设，包括：
  - UART
  - I^2C
  - SPI
  - I^2S
  - LCD RGB
  - CMT （载波调制发送器）
  - PWM/ 波形生成
  - SWD （单线调试）
- 在软件模拟的 GPIO 方法和确切的硬件外设模块之间创建互联。

3 所需硬件

本文档介绍基于 Freescale 塔式系统的示例应用。这些基本的概念也可在用户的硬件上轻易实现。

可使用以下塔式系统板轻松搭建此应用：

- TWR-KL43Z48M
- TWR-ELEV （主板和次板）
- 或者， TWR-SER

此示例使用 FlexIO 模块进行 UART 通信，具有以下参数：

- 8 位通信
- 一个停止位
- 无奇偶校验
- 无硬件流控制

可利用 USB 标准的虚拟 CDC 层，通过 OpenSDA 接口轻松实现与 PC 的通信。还可利用 TWR-SER 卡，将 RS232 与 PC 直接连接。 FlexIO 模拟的 UART 数据帧如图 1 所示。

图 1. UART 8 位数据帧

4 UART 概述

通用异步接收器 / 发送器是一种计算机硬件，用于实现并行和串行形式之间的数据转换。 UART 通常配合 EIA、RS-232、RS-422 或 RS-485 等通信标准使用。通用设计意味着数据格式和传输速度是可配置的。电气信号水平和发送方式 （如差分信号等）由 UART 外部的驱动电路处理。

UART 通常为一个独立的集成电路或集成电路的一部分，用于计算机或外设器件串行端口之间的串行通信。

UART 的发送和接收必须设为相同的位速度、字符长度、奇偶校验和停止位，以实现正常操作。 UART 接收方可检测出不匹配的设置并为主机系统设置 “帧错误” 标志位；在异常情况下，UART 接收方将产生不稳定的残缺字符流并将其传输至主机。

用于连接个人电脑和调制解调器的典型串口使用 8 个数据位、无奇偶校验和 1 个停止位；对于这种配置，每秒传输的 ASCII 字符数等于比特率除以 10。

5 通过 FlexIO 模块模拟 UART

可使用两个定时器、两个移位器和两个引脚实现 UART 总线。使用一个定时器、一个移位器和一个引脚实现发送器。另外一个定时器、一个移位器和一个引脚用于接收器部分。发送器和接收器部分均可单独使用。起始和停止位插入由 FlexIO 外设自动处理。模拟外设的最大波特率为 115200 波特。软件实现中允许在中断或轮询模式下使用 UART。

分隔和空闲字符需要软件干预，并且没有在示例应用中实现。使用 DMA 控制器支持可配置的位顺序 （位交换缓冲区 MSB 在前）和多次传输。 FlexIO 模块不允许自动插入奇偶校验位。图 2 所示为 FlexIO 模拟 UART 的内部连接。

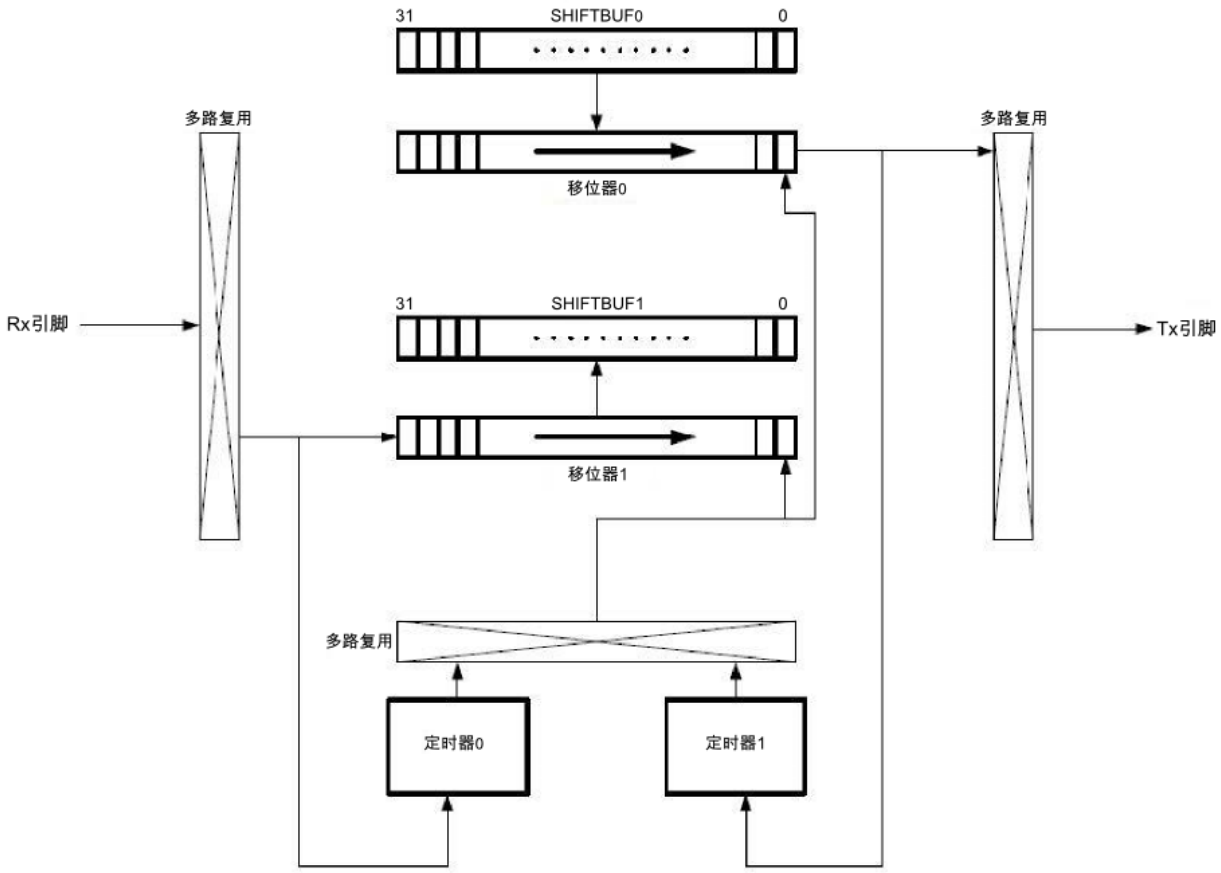

图 2. UART 模拟框图

5.1 发送器

发送过程包括以下步骤：

- 将移位器设为发送模式
- 将数据由移位器缓冲器加载到移位器中
- 将数据移位至引脚输出
- 起始和停止位在数据前后自动加载
- 使用定时器状态标志位发送下一个数据帧

图 3 显示了 UART 发送器模拟原理。在轮询模式下，将会检查定时器状态标志位，并且当使能中断设置时，该模块会生成中断。

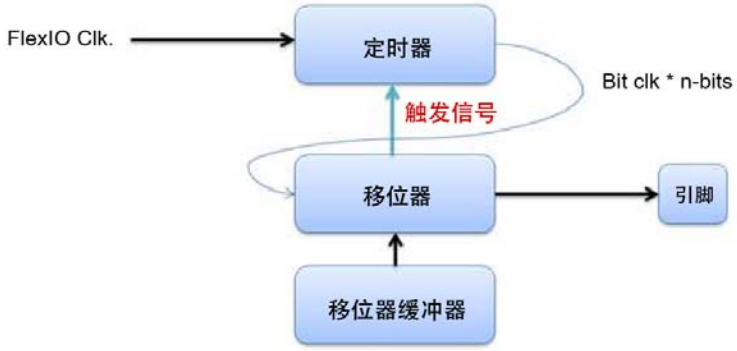

图 3. FlexIO 模块上的 UART 发送器框图

5.2 接收器

接收过程包括以下步骤：

- 将移位器设为接收器模式
- 接收到开始信号后，移入数据
- 状态标志位表示可读取数据的时间 （产生中断）
- 在轮询模式下，等待移位器状态标志位
- 存储到移位器缓冲器
- 读取位与移位器缓冲器交换 （无需任何逻辑操作）

图 4 显示了 UART 接收器模拟原理。在轮询模式下，将会检查移位器状态标志位，并且当使能中断设置时，该模块会生成中断。

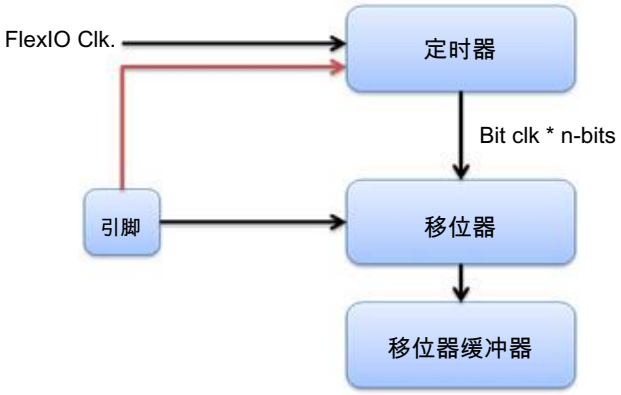

图 4. FlexIO 模块上的 UART 接收器框图

6 软件实现

此示例应用基于使用软件裸跑驱动。该驱动分为两个单独的部分：主要的低层驱动用于外设初始化、移位器和定时器的基本设置，并且支持用户对外设寄存器执行读和写操作。裸跑驱动的子层

主要用于模拟当前外设 (UART)。该驱动可用于其他实例。应用可使用两个驱动层的函数。此外，还实现了回调函数。图 5 显示了裸跑驱动以及针对 FlexIO 外设模块实现的所有驱动的结构。

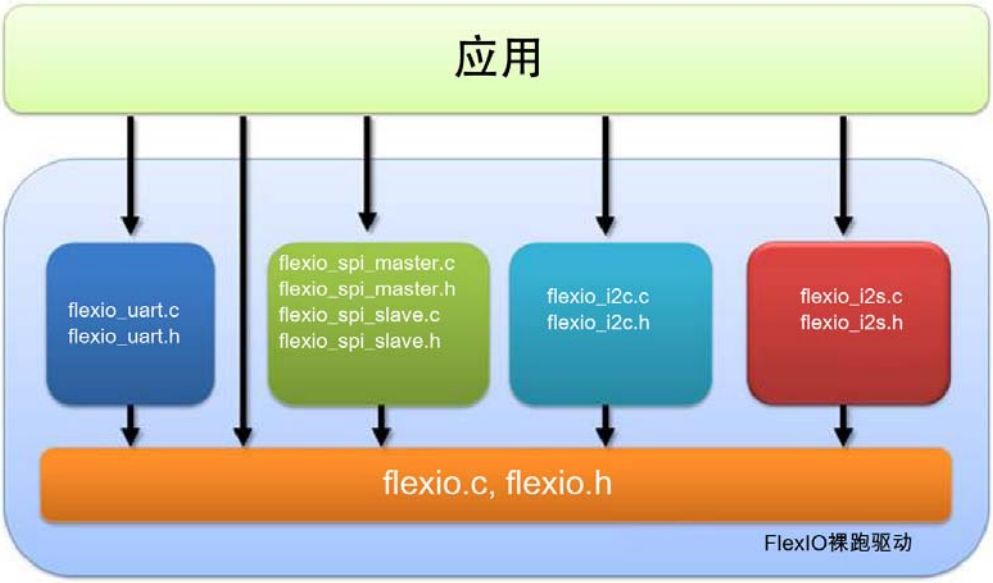

图 5. FlexIO 裸跑驱动框图

6.1 软件设置

在 SIM （系统集成模块）中使能全部所需外设的时钟。高频内部参考时钟 (HIRC) 用作时钟源，为系统提供 48 MHz 频率。

初始化驱动包括两个函数。第一个针对传输侧 *FLEXIO_UART_TxInit*，第二个针对接收侧 *FLEXIO_UART_RxInit*。

所有配置值由文件“*appconfig.h*”提供。这些值用于 UART 波特率的计算，但可以由用户定义替代。

***“appconfig.h”*的宏定义示例：**

```
#define FLEXIO_UART_RX_BAUDRATE          9600
#define FLEXIO_UART_TX_BAUDRATE          9600
#define MCU_SYSTEM_CLOCK                 2000000
```

6.2 FlexIO 高层驱动 API

*FLEXIO_UART_RESULT FLEXIO_UART_TxInit(FLEXIO_UART_CONFIG * pConfig, uint32_t shifterOutIx, uint32_t timerOutIx, uint32_t pinTxIx, uint32_t baudRateTx, uint32_t flexioClk);*

FLEXIO_UART 模块 Tx 初始化函数：

- *pConfig - 配置结构体指针*
- *shifterOutIx - 发送移位器索引*
- *timerOutIx - 发送定时器索引*
- *pinTxIx - 用于 Tx 信号的 FlexIO 引脚索引*
- *baudRate - 所需的波特率*
- *FlexIoClk - FlexIo 外设的时钟频率*
- *返回初始化操作的结果*

*FLEXIO_UART_RESULT FLEXIO_UART_RxInit(FLEXIO_UART_CONFIG * pConfig, uint32_t shifterInIx, uint32_t timerInIx, uint32_t pinRxIx, uint32_t baudRateRx, uint32_t flexioClk);*

FLEXIO_UART 模块 Rx 初始化函数：

- *pConfig - 配置结构体指针*
- *shifterInIx - 接收和 ACK 移位器的索引*
- *timerInIx - 用于移位器寄存器计时的定时器索引*
- *pinRxIx - 用于 Rx 信号的 FlexIO 引脚索引*
- *baudRate - 所需的波特率*
- *FlexIoClk - FlexIo 外设的时钟频率*
- *返回初始化操作的结果*

*FLEXIO_UART_RESULT FLEXIO_UART_PutStr(FLEXIO_UART_CONFIG * pConfig, const uint8 * str);*

FLEXIO_UART 模块写入字符串：

- *pConfig - 配置结构体指针*
- *str - 字符串指针*
- *返回写入操作的结果*

*FLEXIO_UART_RESULT FLEXIO_UART_SendBuffer(FLEXIO_UART_CONFIG * pConfig, const uint8 * pBuff, uint32 len);*

FLEXIO_UART 模块写入数据函数：

- *pConfig - 配置结构体指针*
- *pBuff - 输入缓冲区指针*
- *返回写入操作的结果*

*uint8_t FLEXIO_UART_GetChar(FLEXIO_UART_CONFIG * pConfig);*

FLEXIO_UART 模块读取数据函数：

- *pConfig - 配置结构体指针*
- *返回中断模式中读取的数据字节*

以下函数仅用于中断模式：

*FLEXIO_UART_RESULT FLEXIO_UART_PutStr(FLEXIO_UART_CONFIG * pConfig, const uint8 * str);*
FLEXIO_UART 模块字符输出函数：

- *pConfig - 配置结构体指针*
- *ch - 输入字符指针*
- *返回写入操作的结果。*

*FLEXIO_UART_SetCallBack(FLEXIO_UART_CONFIG * pConfig, FLEXIO_UART_CALLBACK pCallback);*

FLEXIO_UART 模块注册回调函数：

- *pConfig - 配置结构体指针*
- *pCallback - 回调函数指针 （取消注册为 NULL）*
- *注册回调函数的操作结果*
- *执行如同函数调用。*

7 结语

本应用笔记介绍了可通过 Freescale Kinetis KL43 MCU 提供的 FlexIO 外设模块轻松实现的应用实例。通过创建的示例应用演示在使用飞思卡尔塔式平台的相同应用中，FlexIO 模拟 UART 和硬件 UART 模块的使用情况。本应用笔记以及应用软件可从飞思卡尔网站上获取。

8 参考文献

- 飞思卡尔塔式系统 - freescale.com/tower
- FlexIO 模拟 I^2S 总线主机应用笔记
- TWR-KL43Z48M：Kinetis KL43、 KL33、 KL27、 KL17 48 MHz MCU 塔式系统模块 - freescale.com/webapp/sps/site/prod_summary.jsp?code=TWR-KL43Z48M

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: AN5034
Rev. 0, 01/2015