

## PAN1080 HAL UART Sample Application Note

PAN-CLT-VER-B0, Rev 0.1

# PanchipMicroelectronics

www.panchip.com



## PAN1080 HAL UART Sample

修订历史

版本	修订日期	描述
V0.1	2023-9-27	初始版本创建



目录

第1章 例程演示内容	5
1.1 测试内容	5
1.2 环境准备	5
1.2.1 软件环境	5
1.2.1.1 待测代码	5
1.2.1.2 软件工具	6
1.2.2 硬件环境	6
第2章 例程演示流程	7
2.1 环境配置	7
2.1.1 测试程序编译烧录	7
2.1.2 硬件接线	7
2.2 UART 工作流程	8
2.3 测试程序初始化	8
2.4 基本功能验证	8
2.4.1 UART 所有寄存器默认状态 UART_RegisterDefaultValueCheckCase0()	8
2.4.2 数据位与停止位选择 UART_DataLenSelectAndStopBitTestCase1();	9
2.4.2.1 5 bit 数据, 1.5 bit 停止位	9
2.4.2.2 6 bit 数据, 2 bit 停止位	11
2.4.2.3 7 bit 数据, 2 bit 停止位	13
2.4.2.4 8 bit 数据, 2 bit 停止位	15
2.4.3 奇偶校验 UART_ParityCheckTestCase2();	17
2.4.3.1 奇校验(ODD)	17
2.4.3.2 偶校验(EVEN)	19
2.4.3.3 =1 校验(MARK)	21
2.4.3.4 =0 校验(SPACE)	23
2.4.3.5 校验错误演示	25
2.4.4 波特率修改 UART_BaudrateTestCase3();	27
2.4.4.1 波特率 9600	27
2.4.4.2 波特率 115200	29
2.4.4.3 波特率 345600	31
2.4.4.4 波特率 1M	32
2.4.4.5 波特率 2M	34
2.4.4.6 波特率随意指定	36
2.4.5 Polling send and receive data UART_TX_Rx_Polling_TestCase4();	38
2.4.5.1 Polling send HAL_UART_SendData()	38
2.4.5.2 Polling receive HAL_UART_ReceiveData()	39
2.4.6 Interrupt send and receive data UART_TX_Rx_Interrupt_TestCase5();	40
2.4.6.1 Interrupt send HAL_UART_SendData_INT	40
2.4.6.2 Interrupt receive HAL_UART_ReceiveData_INT()	41
2.4.7 DMA send and receive data UART_TX_Rx_DMA_TestCase6();	43



## PAN1080 HAL UART Sample

	2.4.7.1 DMA send HAL_UART_SendData_DMA()	43
	2.4.7.2 DMA receive HAL_UART_ReceiveData_DMA()	44
笛3音	使用注音事项	46



# 第1章 例程演示内容

## 1.1 测试内容

- 1. 寄存器默认值
- 2. 数据位与停止位选择
  - a) 5 bit 数据, 1.5 bit 停止位
  - b) 6 bit 数据, 2 bit 停止位
  - c) 7 bit 数据, 2 bit 停止位
  - d) 8 bit 数据, 2 bit 停止位
- 3. 奇偶校验
  - a) 奇校验(ODD)
  - b) 偶校验(EVEN)
  - c) =1 校验(MARK)
  - d) =0 校验 (SPACE)
- 4. 波特率修改
  - a) 波特率 9600
  - b) 波特率 115200
  - c) 波特率 345600
  - d) 波特率 1M
  - e) 波特率 2M
  - f) 波特率随意指定

- (Register default value)
- (Data length and stop bit select)

(Parity check)

 $({\bf Baudrate\ modify})$ 

- 5. 使能 FIFO 并设定中断触发阈值 (Enable FIFO and set interrupt trigger threshold)
  - a) 设定 trigger level: Rx FIFO 有 1 字节, Tx FIFO 空
  - b) 设定 trigger level: Rx FIFO 有 4 字节, Tx FIFO 剩 2 字节
  - c) 设定 trigger level: Rx FIFO 有 8 字节, Tx FIFO 剩 4 字节
  - d) 设定 trigger level: Rx FIFO 有 14 字节, Tx FIFO 剩 8 字节
- 6. 使能 FIFO 并开启 LSR 中断 (Enable FIFO and Line Status Interrupt)
- 7. 使能 Auto Flow Control (Enable Auto Flow Control)
- 8. DMA 传输

- (DMA transfer)
- a) Burst Transaction 收发数据
- b) Single Transaction 收发数据
- c) Burst & Single Transaction 收发数据
- d) 使用 DMA 并开启 UART Auto Flow Control 功能收发数据
- 9. 使能 9-bit Mode

(Enable 9-bit Mode)

### 1.2 环境准备

- 1.2.1 软件环境
- 1.2.1.1 待测代码

测试工程文件:

<PAN1080-DK>\03\_MCU\mcu\_samples\_hal\UART\keil\UART.uvprojx

#### 测试源主文件目录:

<PAN1080-DK>\03\_MCU\mcu\_samples\_hal\UART\src

#### 1.2.1.2 软件工具

- 1、SecureCRT(用于显示 PC 与 EVB 的交互过程,打印 log 等)
- 2、KingstVIS(逻辑分析仪LA1010配套软件)

#### 1.2.2 硬件环境

- 1、PAN1080 EVB 2 块
  - a) UART0(测试交互接口, TX: P00, RX: P01, 波特率: 921600)
  - b) UART1(待测模块, TX: P14, RX: P15, CTS: P16, RTS: P17)
  - c) SWD (用来调试和烧录程序, SWDCLK: P46, SWDIO: P47)
  - d) Output Pin:P12(NBLK\_PIN) 作为辅助引脚,用于测试数据收发是否为非阻塞模式。
- 2、逻辑分析仪(波形抓取工具)
- 3、JLink(SWD调试与烧录工具)



## PAN1080 HAL UART Sample

# 第2章 例程演示流程

### 2.1 环境配置

#### 2.1.1 测试程序编译烧录

打开测试工程,确保可以编译通过。PAN1080 包含 2 个 UART Module: UART0 与 UART1, 但由于 UART0 被用作日志输出,本测试工程默认将 UART1 作为待测 UART。

### 2.1.2 硬件接线

接线方面,不同的测试内容会需要不同的连线方法。

1. 测试项目 1~3,只需要与 PC 串口调试助手通信即可,按照下图的方式接线。其中 DBG\_PORT (UART0)用来输入测试指令和打印 Log,与 PC 端 SecureCRT 对应的串口 相连;TGT\_UART (UART1)是待测模块,与 PC 端串口调试助手对应的串口相连。



#### PN108 COB Test Board

2. 对于测试项目 4-6 只需 DBG\_PORT (UART0)一路串口,用来输入测试指令和打印,接 受发送测试数据。UART0 RX,NBLK\_PIN 分别接入逻辑分析仪的通道 0~通道 1。



## 2.2 UART 工作流程

参考 User Manual 文档。

## 2.3 测试程序初始化

硬件连线完成并烧录测试程序后, EVB 上电, 观察串口是否正常打印测试主菜单。

G Serial-COM5 - SecureCRT	3 <u>111</u>		$\times$
文件(F) 编辑(E) 查看(V) 选项(O) 传输(T) 脚本(S) 工具(L) 帮助(H)			
143 X3 G X3 I I I I I X X I I I I I I I I I I I			Ŧ
Serial-COM5			4 Þ
СРИ @ 6400000НZ			^
PN108D UART Sample Code	<mark>+</mark>		
Press key to start specific testcase:			
Input '0' Testcase 0: Register Default Value Check.			
Input 1 Testcase 1: Data Length Select and Stop Bit Test. Input '2' Testcase 2: Parity Check Test.			
Input '4' Testcase 4: FIFO and Threshold Test.			
Input '6' Testcase 6: Auto Flow Control Test.			
Input '8' Testcase 8: 9-bit Mode Test.			
就绪 Serial: COM5, 115200 8, 1 19行, 86列 VT100		大写	* 数字 .::

2.4 基本功能验证

## 2.4.1 UART 所有寄存器默认状态 UART\_RegisterDefaultValueCheckCase0()

在主菜单下,输入'0'命令打印所有寄存器默认值:

测试目的:

检查所有 UART 相关寄存器复位 Default 值状态。

测试预期:

寄存器默认值应和 Datasheet 上 UART 模块默认值一致。

测试现象:



0 Target UAR	TO F	Register Default Values:
RBR THR DLI	=	0x00000000
IER DLH	=	0x00000000
IIR_FCR	=	0x00000001
LCR	=	0x00000000
MCR	=	0x00000000
LSR	=	0x0000060
MSR	=	0x00000001
SCR	=	0x00000000
LPDLL	=	0x00000000
LPDLH	=	0x00000000
USR	=	0x0000006
TFL	=	0x00000000
RFL	=	0x00000000
DMASA	=	0x00000000
DLF	=	0x00000000
RAR	=	0x00000000
TAR	=	0x00000000
LCR_EXT	=	0x00000000
UART Test (	DK,	Success case: 0

测试分析:

参考芯片手册对比寄存器信息,均与手册一致,测试 OK。

#### 2.4.2 数据位与停止位选择 UART DataLenSelectAndStopBitTestCase1();

在主菜单下,输入'1'命令 进入 Subcase 菜单:

Press key to test specific function (Baudrate = 115200): Input 'A' 5-bit data with 1.5 stop bits. Input 'B' 6-bit data with 2 stop bits. Input 'C' 7-bit data with 2 stop bits. Input 'D' 8-bit data with 2 stop bits. Press ESC key to back to the top level case list.

2.4.2.1 5 bit 数据, 1.5 bit 停止位

测试目的:

验证 5-bit 数据、1.5-bit 停止位的设定下收发数据是否正常。

测试预期:

能够准确收发 5-bit 数据,无法准确收发 5-bit 以上的数据。

测试现象:

PC端,串口调试助手先将数据位配置为5,停止位配置为1.5,然后打开串口准备接收数据。

EVB 端,输入'A'命令,串口调试助手收到 16 字节的数据,其中前 8 字节正确,后 8 字节错误;随后使用串口调试助手发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到。



## PAN1080 HAL UART Sample

••	串口调试助手	→ ×
串口设置	数据日志 用户支持 ]	VartAssist V4.3.13
串口号 [CUM24 #S1]	[2022-02-16 10:28:17.481]# RECV HEX>	~
波特率 115200	00 01 02 03 1C 1D 1E 1F	
校验位 NONE 工	[2022-02-16 10:28:17.718]# RECV HEX>	
数据位 5 🖃	00 01 02 03 0C 1D 0E 1F	
停止位 1.5 💌	[2022-02-16 10:28:48.089]# SEND HEX>	
<b>)</b> 美闭	01 02 03 04 05 06 07 08	
C ASCII @ HEX		
☑ 按日志模式显示		
▶ 接收完自动换行		
□ 接收转向至文件		
1 智得接收区亚示	<b>II</b> .	
<u>县110达坝</u> 宜味接收		
发送设置	1	
C ASCII C HEX		~
□ 目动解析转义符 □ AT指令自动回车	数据发送   1.DCD ● 2.RXD ● 3.TXD ● 4	. <u>DTR</u> • 5.GNI - 清除 1.清除
□ 自动发送校验位	12345678	
□ 打开文件数据源		45.5
□ 循环周期 1000 ms		2.K
快捷定义历史发送	1	
☞ 就绪!	4/1 RX:28	TX:8 复位计数

Send data: 0x00 0x01 0x02 0x03 0x1C 0x1D 0x1E 0x1F (should be valid) 0x20 0x21 0x22 0x23 0xCC 0xDD 0xEE 0xFF (should be invalid) Data sent successfully.

Try to receive 8 bytes of data... Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08

Press key to test specific function (Baudrate = 115200): Input 'A' 5-bit data with 1.5 stop bits. Input 'B' 6-bit data with 2 stop bits. Input 'C' 7-bit data with 2 stop bits. Input 'D' 8-bit data with 2 stop bits. Press ESC key to back to the top level case list.

#### 测试分析:

前 8 字节 PC 端发送的均是小于 32 的数据, 5-bit 能够 cover 到, 因此 EVB 端能够准确接收



到,符合预期;

后 8 字节 PC 端发送的均是大于等于 32 的数据, 5-bit 已经无法 cover 到, 因此 EVB 端无法 准确接收, 符合预期;

最后 EVB 端发送 8 字节数据, PC 端也准确接收到, 符合预期。

2.4.2.2 6 bit 数据, 2 bit 停止位

测试目的:

验证 6-bit 数据、2-bit 停止位的设定下收发数据是否正常。

测试预期:

能够准确收发 6-bit 数据,无法准确收发 6-bit 以上的数据。

#### 测试现象:

PC 端,串口调试助手先将数据位配置为6,停止位配置为2,然后打开串口准备接收数据。

EVB 端, 输入 'B' 命令, 串口调试助手收到 16 字节的数据, 其中前 8 字节正确, 后 8 字 节错误; 随后使用串口调试助手发送 8 字节数据, 从 EVB 端的 Log 可以看到也有准确接收到。



## PAN1080 HAL UART Sample

••	串口调试助手	×□- \₩
串口设置	数据日志   用户支持	VartAssist V4.3.13
串口号 [COM24 #Siz]	[2022-02-16 10:32:10.570]# RECV HEX>	~
波特率 115200 _	00 01 02 03 3C 3D 3E 3F	
校验位 NONE	[2022-02-16 10:32:10.821]# RECV HEX>	
数据位 6	00 01 02 03 0C 1D 2E 3F	
停止位 2 🗾	[2022-02-16 10:32:11.852]# SEND HEX>	
<b>美</b> (美)	01 02 03 04 05 06 07 08	
接收设置		
C ASCII @ HEX		
▶ 按日志模式显示		
✓ 接收元目动换行		
□ 接收转向主义(F···· □ 暂停接收区显示		
其他洗顶 清除接收		
发送设置		
C ASCII @ HEX		~
□ 自动解析转义符	数据发送   1.DCD ● 2.RXD ● 3.TXD ● 4.D	TR 🗣 5. GNI 🖌 清除 👠 清除
□ AI指交自动回车 □ 自动发送检验位	12345678	
「打开文件数据源		
□ 循环周期 1000 ms		友氏
快捷定义历史发送		
<b>⊌</b> 就绪!	14/4 RX:108	TX:32 复位计数
b Send data:		2
0x00 0x01 0 0x40 0x41 0	x02 0x03 0x3C 0x3D 0x3E 0x3F (sho x42 0x43 0xCC 0xDD 0xEE 0xFF (sho	ould be valid) ould be invalid)
Try to receive 8 h	tes of data	
Data received: 0x01	. 0x02 0x03 0x04 0x05 0x06 0x07 0	x08

Press key to test specific function (Baudrate = 115200): Input 'A' 5-bit data with 1.5 stop bits. Input 'B' 6-bit data with 2 stop bits. Input 'C' 7-bit data with 2 stop bits. Input 'D' 8-bit data with 2 stop bits. Press ESC key to back to the top level case list.

测试分析:

前 8 字节 PC 端发送的均是小于 64 的数据, 6-bit 能够 cover 到, 因此 EVB 端能够准确接收



到,符合预期;

后 8 字节 PC 端发送的均是大于等于 64 的数据, 6-bit 已经无法 cover 到, 因此 EVB 端无法 准确接收, 符合预期;

最后 EVB 端发送 8 字节数据, PC 端也准确接收到, 符合预期。

2.4.2.3 7 bit 数据, 2 bit 停止位

测试目的:

验证 7-bit 数据、2-bit 停止位的设定下收发数据是否正常。

测试预期:

能够准确收发 7-bit 数据,无法准确收发 7-bit 以上的数据。

测试现象:

PC 端,串口调试助手先将数据位配置为7,停止位配置为2,然后打开串口准备接收数据。

EVB 端, 输入 'C' 命令, 串口调试助手收到 16 字节的数据, 其中前 8 字节正确, 后 8 字 节错误; 随后使用串口调试助手发送 8 字节数据, 从 EVB 端的 Log 可以看到也有准确接收到。



## PAN1080 HAL UART Sample

••	串口调试助手	- □ ×
串口设置 串口号 COM24 #Si▼	数据日志   用户支持	VartAssist V4.3.13
波特率 115200 王	00 01 40 44 68 6C 7E 7F	
校验位 NONE 王	[2022-02-16 10:32:37.723]# RECV HEX>	
● 数据位 10 <u>●</u> ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	[2022-02-16 10:32:38 907]# SEND HEX)	
· <b>美</b> 闭	01 02 03 04 05 06 07 08	
接收设置		
<ul> <li>○ ASCII</li> <li>○ HEX</li> <li>▽ 按日志模式显示</li> <li>▽ 接收完自动换行</li> <li>□ 接收转向至文件</li> <li>□ 暂停接收区显示</li> <li><u>其他洗顶</u><u>清除接收</u></li> </ul>		
- 发送设置 C ASCII C HEX に 自动解析結义符	1	~
	数据发送 1. DCD ● 2. RXD ● 3. TXD ● 4	4 <u>.DTR</u> ● 5.GNI √清除 ~清除
<ul> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li><u>快捷定义</u> 历史发送</li> </ul>	123436/8	发送
[● 就绪!	16/5 RX:124	

C Send data: 0x00 0x01 0x40 0x44 0x68 0x6C 0x7E 0x7F (should be valid) 0x80 0x81 0xAA 0xBB 0xCC 0xDD 0xFE 0xFF (should be invalid) Data sent successfully. Try to receive 8 bytes of data... Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 \_\_\_\_ Press key to test specific function (Baudrate = 115200): Input 'A' 5-bit data with 1.5 stop bits. Input 'B' Input 'C' Input 'D' 6-bit data with 2 7-bit data with 2 8-bit data with 2 stop bits. stop bits. stop bits. Press ESC key to back to the top level case list. ----+

测试分析:

前 8 字节 PC 端发送的均是小于 128 的数据, 7-bit 能够 cover 到, 因此 EVB 端能够准确接



收到,符合预期;

后 8 字节 PC 端发送的均是大于等于 128 的数据, 7-bit 已经无法 cover 到, 因此 EVB 端无 法准确接收, 符合预期;

最后 EVB 端发送 8 字节数据, PC 端也准确接收到, 符合预期。

2.4.2.4 8 bit 数据, 2 bit 停止位

测试目的:

验证 8-bit 数据、2-bit 停止位的设定下收发数据是否正常。

测试预期:

能够准确收发 8-bit 数据。

测试现象:

PC 端,串口调试助手先将数据位配置为8,停止位配置为2,然后打开串口准备接收数据。

EVB 端,输入 'D' 命令,串口调试助手收到 8 字节的数据均正确;随后使用串口调试助手发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到。



## PAN1080 HAL UART Sample

· /	串口调试助手	×□-□×
串口设置 串口号 <sup>COM24 #Si▼</sup>	数据日志   用户支持	VartAssist V4.3.13
波特率 115200 👤	00 01 02 03 CC DD EE FF	^
校验位NONE	[2022-02-16 10:33:09.532]# SEND HEX>	
数据位 8 🗾	01 02 03 04 05 06 07 08	
停止位   2 三		
· 美闭		
接收设置		
C ASCII ● HEX □ ☆ロ主搏子見テ		
▶ 接收完自动换行	<	
□ 接收转向至文件		
□ 暂停接收区显示	<b>H</b>	
其他洗项 清除接收		
发送设置	1	
C ASCII C HEX		~
□ 目初解研转X付 □ ΔT指令白动同车	数据发送 1.DCD ● 2.RXD ● 3.TXD ● 4.	DTR 🗕 5. GNI 🖌 清除 👠 清除
F 自动发送校验位	12345678	
□ 打开文件数据源		
□ 循环周期 1000 ms		A K
快捷定义历史发送		
☞ 就绪!	17/6 RX:132	TX:48 复位计数 //
d Send data:		hould be valid)
Data sent successfu	ully.	
Try to receive 8 by Data received: 0x0	rtes of data L 0x02 0x03 0x04 0x05 0x06 0x07 0	0x08
Press key to 1	test specific function (Baudrate	= 115200):
Input 'A'	5-bit data with 1.5 stop bits.	

Input 'B' 6-bit data with 2 stop bits. Input 'C' 7-bit data with 2 stop bits. Input 'D' 8-bit data with 2 stop bits. Press ESC key to back to the top level case list.

测试分析:

PC 端发送的 8 个字节, 而 8-bit 的数据格式能够 cover 到所有情况, 因此 EVB 端准确接收



到,符合预期;

然后 EVB 端发送 8 字节数据, PC 端也准确接收到, 符合预期。

#### 2.4.3 奇偶校验 UART\_ParityCheckTestCase2();

在主菜单下,输入'2'命令 进入 Subcase 菜单:

Press key to test specific function (Baudrate = 115200): 8-bit data with ODD parity check. Input 'A 'B' parity check. 8-bit data with EVEN Input 'C' Input 8-bit data with MARK parity check. Input 8-bit data with SPACE parity check. Press ESC key to back to the top level case list.

#### 2.4.3.1 奇校验(ODD)

#### 测试目的:

验证奇校验功能是否正常。

#### 测试预期:

开启奇校验的情况下,能够准确收发数据。

#### 测试现象:

PC 端,串口调试助手先将校验位设置为 ODD (奇校验),然后打开串口准备接收数据。

EVB 端,输入 'A' 命令, 串口调试助手收到 8 字节的数据均正确;随后使用串口调试助手 发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到,且 Error Flag (从 LSR 中获取) 均为 No Error。



## PAN1080 HAL UART Sample

· ·	串口调试助手	₩ - □ ×
串口设置	数据日志   用户支持	VartAssist V4.3.13
	[2022-02-16 10:34:51.664]# RECV HEX>	~
波特率 110200 上	OU DI UZ US CC III EE FF	
秋掘位 8 ▼	L2022-02-16 10:34:52.728J# SEND HEX> 01 02 03 04 05 06 07 08	
停止位 1 🔽		
· ● 关闭		
接收设置		
C ASCII C HEX		
☑ 按口志模式显示 ☑ 接收完自动换行	<	
□ 接收转向至文件		
□ 暂停接收区显示		
「友法设置 ○ ASCTT ● HFX	Ī	
□ 自动解析转义符		
F AT指令自动回车	叙境友法 1.DCD● 2.RXD● 3.TXD● <u>4.DTR</u> ●	5.GNI 🛊 消除 こ 消除
□ 目动发送仪短辺 □ 打开文件数据源	12545070	
└ 循环周期 1000 ms		发送
快捷定义历史发送		
☞ 就绪!	19/9 RX:148	TX:72 复位计数



PAN1080 HAL UART Sample

```
a
Send data:
          0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF
Data sent successfully.
Try to receive 8 bytes of data...
Data: 0x01, Error Flag: | No Erro
                                No Error
Data: 0x02, Error Flag:
Data: 0x03, Error Flag:
                                No Error
                                No Error
Data: 0x04, Error Flag:
                                No Error
Data: 0x05, Error Flag:
Data: 0x06, Error Flag:
Data: 0x07, Error Flag:
                                No Error
                                No Error
                                No Error
Data: 0x08, Error Flag:
                              No Error
      Press key to test specific function (Baudrate = 115200):
      Input 'A'
Input 'B'
                                                  parity check.
                      8-bit data with ODD
                      8-bit data with EVEN
                                                  parity check.
             'c'
                                                 parity check.
                      8-bit data with MARK
      Input
      Input 'D'
                      8-bit data with SPACE parity check.
      Press ESC key to back to the top level case list.
```

测试分析:

PC 端与 EVB 端均开启奇校验,互相收发数据均正常,符合预期。

2.4.3.2 偶校验(EVEN)

测试目的:

验证偶校验功能是否正常。

测试预期:

开启偶校验的情况下,能够准确收发数据。

测试现象:

PC 端,串口调试助手先将校验位设置为 EVEN (偶校验),然后打开串口准备接收数据。

EVB 端,输入 'B' 命令, 串口调试助手收到 8 字节的数据均正确; 随后使用串口调试助手 发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到,且 Error Flag (从 LSR 中获取) 均为 No Error。



## PAN1080 HAL UART Sample

· ·	串口调试助手	→ ×
串口设置	数据日志   用户支持	VartAssist V4.3.13
串口号 [115200] -	[2022-02-16 10:35:26.518]# RECV HEX>	~
波特率 115266 ·	OU OI OZ OS CC DD EE FF	
	[2022-02-16 10:35:27.968]# SEND HEX> 01 02 03 04 05 06 07 08	
·····································		
接收设置		
C ASCII @ HEX		
✓ 按日志模式显示	<	
□ 接收转向至文件		
□ 暂停接收区显示		
<u> </u>		
发送设置		
C ASCII (● HEX □ 自动解析转义符	1	×.
□ AT指令自动回车	数据发送 1. DCD ◆ 2. RXD ◆ 3. TXD ◆ <u>4. DTR</u> ◆	5.GNI - 清除 1 清除
□ 自动发送校验位 □ 打开立件物理源	12345678	
「循环周期 1000 ms		发送
快捷定义历史发送		
☞ 就绪!	20/10 RX:156	TX:80 复位计数



## PAN1080 HAL UART Sample

```
b
Send data:
          0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF
Data sent successfully.
Try to receive 8 bytes of data...
Data: 0x01, Error Flag:
                                No Error
Data: 0x02, Error Flag:
Data: 0x03, Error Flag:
                                No Error
                                No Error
Data: 0x04, Error Flag:
                                No Error
Data: 0x05, Error Flag:
                                No Error
Data: 0x06, Error Flag:
Data: 0x07, Error Flag:
Data: 0x08, Error Flag:
                               No Error
                               No Error
                             ï
                               No Error
      Press key to test specific function (Baudrate = 115200):
      Input 'A'
Input 'B'
                      8-bit data with ODD
                                                 parity check.
                      8-bit data with EVEN
8-bit data with MARK
      Input
                                                 parity check.
             'c'
                                                 parity check.
      Input
      Input 'D'
                      8-bit data with SPACE parity check.
      Press ESC key to back to the top level case list.
```

测试分析:

PC 端与 EVB 端均开启偶校验,互相收发数据均正常,符合预期。

2.4.3.3 =1 校验(MARK)

测试目的:

验证固定奇校验功能是否正常。

测试预期:

开启固定奇校验的情况下,能够准确收发数据。

测试现象:

PC 端,串口调试助手先将校验位设置为 MARK(固定奇校验,=1 校验),然后打开串口准备接收数据。

EVB 端,输入 'C' 命令, 串口调试助手收到 8 字节的数据均正确;随后使用串口调试助手 发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到,且 Error Flag (从 LSR 中获取) 均为 No Error。



## PAN1080 HAL UART Sample

· ·	串口调试助手	×
串口设置 串口号 <sup>COM24 #Sj</sup> ▼ 波特率 <sup>115200</sup> ▼ 校验位 MARK ▼ 数据位 8 ▼ 停止位 1 ▼	数据日志 用户支持】 [2022-02-16 10:35:56.835]# RECV HEX> 00 01 02 03 CC DD EE FF [2022-02-16 10:35:58.047]# SEND HEX> 01 02 03 04 05 06 07 08	<u>VartAssist V4.3.13</u>
接收设置 ○ ASCII ○ HEX ▼ 按日志模式显示 ▼ 接收完自动换行 □ 接收转向至文件 ■ 暂停接收区显示 <u>其他洗顶</u> <u>清除接收</u>		
<ul> <li>发送设置</li> <li>▲SCII ● HEX</li> <li>目动解析转义符</li> <li>▲T指令自动回车</li> <li>自动发送校验位</li> <li>打开文件数据源</li> <li>循环周期 1000 ms</li> <li>快捷定义 历史发送</li> </ul>	数据发送 1.DCD ● 2.RXD ● 3.TXD ● <u>4.DTR</u> ● 12345678	5.GMI √ 清除 ~ 清除 发送
☞ 就绪!	21/11 RX:164	TX:88 复位计数



PAN1080 HAL UART Sample

```
C
Send data:
          0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF
Data sent successfully.
Try to receive 8 bytes of data...
Data: 0x01, Error Flag:
                                No Error
Data: 0x02, Error Flag:
Data: 0x03, Error Flag:
                                No Error
                                No Error
Data: 0x04, Error Flag:
                                No Error
Data: 0x05, Error Flag:
Data: 0x06, Error Flag:
Data: 0x06, Error Flag:
Data: 0x07, Error Flag:
                                No Error
                                No Error
                                No Error
Data: 0x08, Error Flag:
                             1
                               No Error
      Press key to test specific function (Baudrate = 115200):
     Input 'A'
Input 'B'
                                                 parity check.
                      8-bit data with ODD
                      8-bit data with EVEN
                                                 parity check.
             'c'
                      8-bit data with MARK
                                                 parity check.
      Input
      Input 'D'
                      8-bit data with SPACE parity check.
      Press ESC key to back to the top level case list.
```

测试分析:

PC 端与 EVB 端均开启固定奇校验,互相收发数据均正常,符合预期。

2.4.3.4 =0 校验(SPACE)

测试目的:

验证固定偶校验功能是否正常。

测试预期:

开启固定偶校验的情况下,能够准确收发数据。

测试现象:

PC 端,串口调试助手先将校验位设置为 SPACE(固定偶校验,=0 校验),然后打开串口准备接收数据。

EVB 端,输入 'D' 命令, 串口调试助手收到 8 字节的数据均正确;随后使用串口调试助手 发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到,且 Error Flag (从 LSR 中获取) 均为 No Error。



## PAN1080 HAL UART Sample

· ·	串口调试助手	4 - O ×
串口设置	□数据日志 ┃ 用户支持 ┃	VartAssist V4.3.13
	[2022-02-16 10:36:36.002]# RECV HEX>	~
波特率 115200 工	OU OI OZ OS CC DD EE FF	
	[2022-02-16 10:36:36.977]# SEND HEX>	
● 刻楣□□ □ □ □		
·••• 关闭		
接收设置		
C ASCII 📀 HEX		
▼ 按日志模式显示		
▶ 接收转向至文件		
□ 暂停接收区显示		
其他洗项 清除接收		
发送设置		
C ASCII C HEX		~
□ 目动解析转义符 □ AT指令自动回车	数据发送   1. DCD ● 2. RXD ● 3. TXD ● <u>4. DTR</u> ●	5.GNI 🖌 清除 👠 清除 🚽
□ 自动发送校验位	12345678	
□ 打开文件数据源		发送
□ 1/目が「向期] 1000 ms 中博会公 市中安洋		
☞ 就绪!	22/12 RX:172	TX:96 复位计数



PAN1080 HAL UART Sample

```
d
Send data:
        0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF
Data sent successfully.
Try to receive 8 bytes of data...
Data: 0x01, Error Flag:
                            No Error
Data: 0x02, Error Flag:
Data: 0x03, Error Flag:
                            No Error
                            No Error
Data: 0x04, Error Flag:
                            No Error
Data: 0x05, Error Flag:
                            No Error
Data: 0x06, Error Flag:
                            No Error
Data: 0x07, Error Flag:
                            No Error
Data: 0x08, Error Flag:
                          1
                            No Error
     Press key to test specific function (Baudrate = 115200):
     Input 'A'
Input 'B'
                   8-bit data with ODD
                                            parity check.
                   8-bit data with EVEN
                                            parity check.
           'C'
                   8-bit data with MARK
                                            parity check.
     Input
           'D'
                   8-bit data with SPACE parity check.
     Input
     Press ESC key to back to the top level case list.
```

#### 测试分析:

PC 端与 EVB 端均开启固定偶校验,互相收发数据均正常,符合预期。

#### 2.4.3.5 校验错误演示

测试目的:

故意造出 Parity Bit (校验位)错误的情况,确认 EVB 的 UART 是否能够正确识别奇偶校验 错误的情况。

#### 测试预期:

PC 端发送具有错误校验位的数据, EVB 能够接收并识别到校验错误。

EVB 端发送具有错误校验位的数据, PC 端能够接收并识别到校验错误。

#### 测试现象:

PC 端,串口调试助手先将校验位设置为 ODD (奇校验),然后打开串口准备接收数据; EVB 端,输入 'B' 命令,进入 EVEN (偶校验)收发流程,这样对 PC 和 EVB 端来说,接收到的数据的校验位均是错的。

现象是, PC 端串口调试助手收到 8 字节的数据均正确;随后使用串口调试助手发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到, Error Flag 显示有奇偶校验错误。



## PAN1080 HAL UART Sample

· ·	串口调试助手	4 - O ×
串口设置	数据日志│用户支持│	VartAssist V4.3.13
串口号 [CUM24 #S1]]	[2022-02-16 10:37:06.335]# RECV HEX>	~
波特率 115200 王	00 01 02 03 CC DD EE FF	
	[2022-02-16 10:37:08.272]# SEND HEX>	
	01 02 03 04 05 08 07 08	
得止位「」		
接收设置		
C ASCII 📀 HEX		
▼ 按日志模式显示	~	
✓ 接收元目初换行		
□ 暂停接收区显示		
其他洗顶 清除接收		
发送设置		
C ASCII @ HEX		~
□ 自动解析转义符 □ AT指令自动回车	数据发送   1. DCD ● 2. RXD ● 3. TXD ● <u>4. DTR</u> ● 5	.GNT √清除 1/清除
□ 自动发送校验位	12345678	
□ 打开文件数据源		发送
□ 循外周期 1000 ms 仲捷完议 历中发送		
▲1世上へ // 2.21× ● 就绪!	23/13 RX:180 T	



## PAN1080 HAL UART Sample

```
b
Send data:
          0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF
Data sent successfully.
Try to receive 8 bytes of data...
Data: 0x01, Error Flag:
                                Parity Error
Data: 0x02, Error Flag:
                                Parity Error
Data: 0x03, Error Flag:
Data: 0x04, Error Flag:
                                Parity Error
                                Parity Error
Data: 0x05, Error Flag:
                                Parity Error
Data: 0x06, Error Flag:
                                Parity Error
Data: 0x07, Error Flag: | Parity Error
Data: 0x08, Error Flag: | Parity Error
      Press key to test specific function (Baudrate = 115200):
      Input 'A'
                      8-bit data with ODD
                                                 parity check.
      Input 'B'
Input 'C'
Input 'D'
                      8-bit data with EVEN
                                                 parity check.
                      8-bit data with MARK parity check.
8-bit data with SPACE parity check.
      Press ESC key to back to the top level case list.
```

#### 测试分析:

从现象来看,PC 端串口调试助手是准确收到 EVB 发来的数据,但由于串口助手没有显示奇 偶校验错误的功能,因此无法判断奇偶校验错误,这是符合预期的;随后使用串口调试助手发送 数据,EVB 端也准确接收到,但 Error Flag (从 LSR 中获取)显示有奇偶校验错误,也符合预期。

#### 2.4.4 波特率修改 UART\_BaudrateTestCase3();

在主菜单下,输入'3'命令 进入 Subcase 菜单:

Press	key t	o test s	specif	fic fu	nction:	
Input	'A'	8-bit	data	under	9600	baudrate.
Input	'B'	8-bit	data	under	115200	baudrate.
Input	'c'	8-bit	data	under	345600	baudrate.
Input	'D'	8-bit	data	under	1M	baudrate.
Input	'E'	8-bit	data	under	2M	baudrate.
Input	'F'	8-bit	data	under	customized	baudrate.
Press	ESC k	ev to ba	ack to	o the 1	top level ca	ase list.

#### 2.4.4.1 波特率 9600

测试目的:

验证 9600 波特率下是否工作正常。

#### 测试预期:

9600 波特率下能够准确收发数据。

#### 测试现象:

PC端,串口调试助手先将波特率设置为9600,然后打开串口准备接收数据。



## PAN1080 HAL UART Sample

EVB 端, 输入 'A' 命令, 串口调试助手收到 8 字节的数据均正确; 随后使用串口调试助手 发送 8 字节数据, 从 EVB 端的 Log 可以看到也有准确接收到。

	串口调	试助手		₩ - □ ×
串口设置	数据日志		VartAs	sist V4.3.13
	[2019-10-18 21:43:38.	068]# RECV HEX>		~
波特率 9600	00 01 02 03 CC DD EE	FF		
校验位 NONE 工	[2019-10-18 21:43:47.	017]# SEND HEX>		
数据位 8 🖃	01 02 03 04 05 06 07	08		
停止位 1 🖃				
· 美闭				
接收设置				
C ASCII 📀 HEX				
▶ 按日志模式显示				
▶ 接收完自动换行				
□ 接收转向至文件				
1 首使接收区亚小				
<u>县1111达坝</u> 宜味接收				
发送设置				
C ASCII C HEX				
✓ 目初解析转义付	1			
▼ AI指支目初回半 □ 自动发送检验位	数据发送   1.DCD ◆	2. RXD 🗢 3. TXD 🗢	4. DTR • 5 5	青除 1 清除
「打开文件数据源	12345678			
□ 循环周期 1000 ms				发送
快捷定义历史发送				
☞ 就绪!	16/13	RX:128	TX:104	复位计数

a Send data: 0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF Data sent successfully. Try to receive 8 bytes of data... Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 \_\_\_\_\_ Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'B' Input 'C' Input 'D' 8-bit data under 115200 8-bit data under 345600 8-bit data under 1M baudrate. baudrate. baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press ESC key to back to the top level case list.



#### 测试分析:

PC 端与 EVB 端互相收发数据均正常,符合预期。

#### 2.4.4.2 波特率 115200

#### 测试目的:

验证 115200 波特率下是否工作正常。

#### 测试预期:

115200 波特率下能够准确收发数据。

#### 测试现象:

PC 端,串口调试助手先将波特率设置为115200,然后打开串口准备接收数据。

EVB 端,输入 'B' 命令, 串口调试助手收到 8 字节的数据均正确; 随后使用串口调试助手 发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到。



## PAN1080 HAL UART Sample

#口设置       数据日志       Uset Assist V4.3.13         #口号       COM4 #UST ×       COUP-10-18 21:45:10.313]# EECV HEX>       COUP-10-18 21:45:10.313]# EECV HEX>         00 01 02 03 0C DD EE FF       COUP-10-18 21:45:11.449]# SEND HEX>       COUP-10-18 21:45:11.449]# SEND HEX>         #WWGE       F       COUP-10-18 21:45:11.449]# SEND HEX>       COUP-10-18 21:45:11.449]# SEND HEX>         #WWGE       F       F       COUP-10-18 21:45:11.449]# SEND HEX>       COUP-10-18 21:45:11.449]# SEND HEX>         #WWGE       F       F       F       F       F         #WWGE       F       F       F       F         #WWGE       #WWGE       F       F       F         #WWGE       #WWGE       #WWGE       #WWGE       F       F         #WWGE       #WWGE <td< th=""><th>· ·</th><th>串口调试助手</th><th>₩ - □ ×</th></td<>	· ·	串口调试助手	₩ - □ ×
田中島 (CMA #151 -)     波特案 [15200 -)     びの 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	串口设置		VartAssist V4. 3. 13
波特案       115200 /       00 01 02 03 0C DD EE FF         検验位       FUNE       -         数据位       9       -         資止位       1       -         (*) 公式       00 00 00 00 00 00 00 00 00 00 00 00 00	串口号 [COM4 #USI]	[2019-10-18 21:45:10.313]# RECV HEX>	~
軟鉛位       INDE       Image: Send HEX>         軟銀位       Image: Send HEX>         動銀位       Image: Send HEX>         ●       Image: Send HEX>         ●       XH         ●       XH <t< td=""><td>波特率 115200</td><td>00 01 02 03 CC DD EE FF</td><td></td></t<>	波特率 115200	00 01 02 03 CC DD EE FF	
動振位 6       ●         ●       □         ●       ★	校验位 NONE	[2019-10-18 21:45:11.849]# SEND HEX>	
@⊥位       ✓         @· ★in          WbWgmm          C ASCII • HEX       ✓         ✓       按由志模式显示         ✓       按由志模式显示         ✓       按由志模式显示         ▲ 按約時面至文件       ●         ●<	数据位 8 🔄 🗾	01 02 03 04 05 06 07 08	
★ 关闭 按收设置 ASCII ● HEX F 按由志權式显示 按规实自动执行 按相比反显示 其他法迈 書話接地 发送设置 C ASCII ● HEX F 自动解析转文符 AT描令自动回车 自动发送校验位 打开文件執握第 值研展期 1000 ms 快捷定义 历史发送 F 就绪: 12345678 1234578	停止位 1 🗾		
##₩₩₩         ##₩₩         ₩₩         ##₩         ##₩         ### <t< td=""><td>· 使 关闭</td><td></td><td></td></t<>	· 使 关闭		
C ASCII	接收设置		
✓ 按日志模式显示       ✓ 接收完自动执行         「 接收转向至文件]       暫停撥收区显示         重他洗面       蕭孫接收         发送设置       C ASCII ⓒ HEX         ♡ 自动解析转义符       ✓ 和指令自动回车         目訪发送校验位       「 打开文件数据源]         「 循环周期 1000 ms       ■         快援完义 历史发送       12 3 4 5 6 7 8         ▶       ★         Y to receive & bytes of data       ●         Data sent successfully.       TX:112         Try to receive & bytes of data       ●         Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08         +       ●         Press key to test specific function:         Input 'A' 8-bit data under 9600       baudrate.         Input 'A' 8-bit data under 115200       baudrate.         Input 'A' 8-bit data under 345600       baudrate.         Input 'E' 8-bit data under 2M       baudrate.         Input 'F' 8-bit data under 2M       baudrat	C ASCII 📀 HEX		
<pre>     # 接收院目面預行     # 接收转向至文件     # 僅接收区显示     #他选面    董能接收     # # # # # # # # # # # # # # # # #</pre>	▼ 按日志模式显示		
I Havkalue_Ztr         I Hokkalue_Ztr         I hokkalue_Ztr      <	✓ 接收元目动换行		
其他选顶 直路接收         发送设置         C ASCII • HEX         ● 自动解析转义符         ✓ 和指令自动回车         目动发送校验位         「 打开文件教掘原]         「 循环周期[1000 ms         慢撞定义 历史发送         ●         ●         Send data:         0X00 0X01 0x02 0x03 0xcc 0xDD 0xEE 0xFF         Data sent successfully.         Try to receive 8 bytes of data         Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08         Press key to test specific function:         Input 'A' 8-bit data under 115200       baudrate.         Input 'A' 8-bit data under 14500       baudrate.         Input 'A' 8-bit data under 115200       baudrate.         Input 'C' 8-bit data under 11M       baudrate.         Input 'E' 8-bit data under 2M       baudrate.         Input 'F' 8-bit data under 2M       baudrate.<	□ 摄线短向主义件		
发送设置       ASCII	其他选项		
C ASCII ⓒ HEX ♥ 自动解析转义符 ♥ 和指令自动回车 自动发送校验位 □ 打开文件数据源 □ 循环周期 1000 ms <u>快捷定义</u> 历史发送 ♥ 就绪! 17/14 RX:136 TX:112 夏位计数 b Send data: 0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF Data sent successfully. Try to receive 8 bytes of data Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'A' 8-bit data under 115200 baudrate. Input 'A' 8-bit data under 115200 baudrate. Input 'B' 8-bit data under 345600 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'E' 8-bit data under customized baudrate. Press ESC key to the to the toon level case list	发送设置	1	
<pre></pre>	C ASCII 🛈 HEX		
✓ Alfa⊽elsnipet       ● Alfa⊽elsnipet         ● 自动发送校验位       ● 1.DCD ● 2.RXD ● 3.TXD ● 4.DTR ● 5 ✓ 清除 ▲ 清除         ● 目动发送校验位       ● 17开文件数据源         ● 循环周期[1000 ms       ■         ● 塘庄空义       历史发送         ●       ●         ●       ●         ●       ●         ●       ●         ●       ●         ●       ●         ●       ●         ●       ●         ●       ●         ●       ●         Send data:       ○         ○x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF         Data sent successfully.         Try to receive 8 bytes of data         Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08         ●	▶ 自动解析转义符	1	
I 打开文件数据源       I 2 3 4 5 6 7 8         I 打开文件数据源       I 1 2 3 4 5 6 7 8         I 插环周期 1000 ms       I 1 2 3 4 5 6 7 8         皮描定义 历史发送       I 1 7/14         IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII	▶ ▲ 相名 日初四年	数据发送   1. DCD ◆ 2. RXD ◆ 3. TXD ◆ <u>4. D</u>	III.● 5 √ 清除 ~ 清除
循环周期 1000 ms       发送         快捷空义 历史发送       17/14       RX:136       TX:112       复位计数         b       Send data:       0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF         Data sent successfully.       Try to receive 8 bytes of data       Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08         Press key to test specific function:       Input 'A' 8-bit data under 9600 baudrate.       Daudrate.         Input 'A' 8-bit data under 115200 baudrate.       Input 'C' 8-bit data under 14500 baudrate.       Daudrate.         Input 'D' 8-bit data under 1M       baudrate.       Input 'E' 8-bit data under 2M       baudrate.         Input 'E' 8-bit data under 2M       baudrate.       Daudrate.       Input 'E' 8-bit data under 2M       baudrate.         Input 'F' 8-bit data under 2M       baudrate.       Input 'E' 8-bit data under 2M       baudrate.	「打开文件数据源	12345678	
b Send data: 0x00 0x01 0x02 0x03 0xcc 0xDD 0xEE 0xFF Data sent successfully. Try to receive 8 bytes of data Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'A' 8-bit data under 115200 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'E' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press FSC key to back to the top level case list	□ 循环周期 1000 ms		发送
b Send data: 0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF Data sent successfully. Try to receive 8 bytes of data Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'E' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press ESC key to back to the top level case list	快捷定义历史发送		
b Send data: 0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF Data sent successfully. Try to receive 8 bytes of data Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'E' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press ESC key to back to the top level case list	☞ 就绪!	17/14 RX:136	TX:112 复位计数
b Send data: 0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF Data sent successfully. Try to receive 8 bytes of data Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press FSC key to back to the top level case list			
0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF Data sent successfully. Try to receive 8 bytes of data Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under 2M baudrate. Press FSC key to back to the top level case list	b Send data:		
Try to receive 8 bytes of data Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'E' 8-bit data under 2M baudrate. Press FSC key to back to the top level case list	0x00 0x01 0 Data sent successfu	0x02 0x03 0xCC 0xDD 0xEE 0xFF	
Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press FSC key to back to the top level case list	Try to receive 8 by	/tes of data 0x02 0x03 0x04 0x05 0x06 0x07 0x	:08
Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press FSC key to back to the top level case list	+		
Input 'A' 8-bit data under 9600 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press FSC key to back to the top level case list	Press key to t	est specific function:	
Input 'C' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press FSC key to back to the top level case list	Input 'A' 8	3-bit data under 9600 baudra	ite.
Input 'D' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press FSC key to back to the top level case list	Input 'C' 8	3-bit data under 115200 baudra 3-bit data under 345600 baudra	ite.
Input 'F' 8-bit data under customized baudrate.	Input 'D' 8	3-bit data under 1M baudra 3-bit data under 2M baudra	ate.
The set we have a set to the top level case list.	Input 'F' 8 Press ESC kev	3-bit data under customized baudra to back to the top level case lis	te.

测试分析:

+



PC 端与 EVB 端互相收发数据均正常,符合预期。

#### 2.4.4.3 波特率 345600

#### 测试目的:

验证 345600 波特率下是否工作正常。

#### 测试预期:

345600 波特率下能够准确收发数据。

#### 测试现象:

PC 端,串口调试助手先将波特率设置为345600,然后打开串口准备接收数据。

EVB 端,输入 'C' 命令,串口调试助手收到 8 字节的数据均正确;随后使用串口调试助手 发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到。

•	串口	週试助手		名 - □ ×
串口设置	数据日志		<u>Var tAs</u>	sist ¥4.3.13
串口号 CUM4 #USI →	[2019-10-18 21:47:4	5.619]# RECV HEX>		~
波特率 343000 工	00 01 02 03 CC DD E	E FF		
	[2019-10-18 21:47:4 01 02 03 04 05 06 0	9.274]# SEND HEX> 7 08		
	1			
·····································				
接收设置				
C ASCII · HEX				
<ul> <li>✓ 按日志視式显示</li> <li>✓ 接收完自动换行</li> </ul>				
□ 接收转向至文件				
□ 暫停接收区显示 其他洗面 法除接收				
反法设立 CASCII GHEX				
▶ 自动解析转义符	<u> </u>			~
▼ AT指令自动回车	数据发送   1.DCD ●	2. RXD 🔷 🛛 3. TXD 🗢	4.DTR • 5	青除 七清除
「日本が反法代報」 「打开文件数据源	12345678			
□ 循环周期 1000 ms <u>快捷定义 历史发送</u>				发送
☞ 就绪!	18/15	RX:144	TX:120	复位计数



## PAN1080 HAL UART Sample

C send data: 0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF Data sent successfully. Try to receive 8 bytes of data... Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 Press key to test specific function: Input 'A' Input 'B' Input 'C' 8-bit data under 9600 baudrate. 8-bit data under 115200 8-bit data under 345600 baudrate. baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'E' Input 'F' Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press ESC key to back to the top level case list.

#### 测试分析:

PC 端与 EVB 端互相收发数据均正常,符合预期。

#### 2.4.4.4 波特率 1M

#### 测试目的:

验证 1M 波特率下是否工作正常。

#### 测试预期:

1M 波特率下能够准确收发数据。

#### 测试现象:

PC 端,串口调试助手先将波特率设置为1M,然后打开串口准备接收数据。

EVB 端, 输入 'D' 命令, 串口调试助手收到 8 字节的数据均正确; 随后使用串口调试助手 发送 8 字节数据, 从 EVB 端的 Log 可以看到也有准确接收到。



## PAN1080 HAL UART Sample

•	串口调试助手	×
串口设置 串口号 COM4 #USI ▼ 波特率 1000000 ▼ 校验位 NONE ▼ 数据位 8 ▼ 停止位 1 ▼	数据日志 [2019-10-18 21:49:22.652]# RECV 00 01 02 03 CC DD EE FF [2019-10-18 21:49:24.913]# SENT 01 02 03 04 05 06 07 08	UartAssist V4.3.13 V HEX>
<ul> <li>● 关闭</li> <li>接收设置</li> <li>▲ ASCII ● HEX</li> <li>▼ 按日志模式显示</li> <li>▼ 接收完自动执行</li> <li>■ 接收转向至文件</li> <li>■ 暂停接收区显示</li> <li>其他洗顶 清除接收</li> </ul>		
<ul> <li>发送设置</li> <li>○ ASCII ● HEX</li> <li>✓ 自动解析转义符</li> <li>✓ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li>快捷定义 历史发送</li> </ul>	数据发送 1.DCD ● 2.RXD ● 3 1 2 3 4 5 6 7 8	3. TXD ♦ <u>4. DTR</u> ◆ 5 <b>「</b> 清除 <b>仁</b> 清除 <b>发送</b>
☞ 发送完毕	19/16 RX:15	52 TX:128 复位计数
d Send data: 0x00 0x01 Data sent successf	0x02 0x03 0xcc 0xDD 0xEE	0xff

Try to receive 8 bytes of data... Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press ESC key to back to the top level case list.

测试分析:

PC 端与 EVB 端互相收发数据均正常,符合预期。

注意:当使用 USB 转串口板与 PC 通信时,某些品牌的转板可能无法支持非标准波特率(1M、2M 等)通信。

2.4.4.5 波特率 2M

测试目的:

验证 2M 波特率下是否工作正常。

测试预期:

2M 波特率下能够准确收发数据。

#### 测试现象:

PC 端,串口调试助手先将波特率设置为 2M,然后打开串口准备接收数据。

EVB 端,输入 'E' 命令,串口调试助手收到 8 字节的数据均正确;随后使用串口调试助手发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到。



## PAN1080 HAL UART Sample

•	串口训	<b>试助手</b>	)	×
串口设置 史口号 COM4 #VSI▼	数据日志		VartAss	ist ¥4.3.13
波特率 2000000 I	[2019-10-18 21:50:05 00 01 02 03 CC DD EF	5.993]# RECV HEX> : FF		~
1x xx12     1       数据位     8       「停止位     1	L2019-10-18 21:50:07 01 02 03 04 05 06 07	7.530]# SEND HEX> 7.08		
· 美闭				
接收设置	1			
C ASCII 📀 HEX				
▼ 按日志模式显示				
▼ 接收完自动换行				
Ⅰ 接收转向至文件 □ 新信協助区見テ				
<u>其他洗顶</u> <u>清除接收</u>				
发送设置	1			
○ ASCII ○ HEX I 自动解析转义符				4
▼ AT指令自动回车	数据发送 1.DCD ◆	2. RXD 🗢 3. TXD 🗢	4. DTR • 5 🗸 🖁	除 1 清除
□ 自动发送校验位 □ 1========1	12345678			
□ 打开文件数据源				发送
法理定义 历史友庆	]			
☑ 就绪!	20/17	RX:160	TX:136	复位计数
e Send data:				

0x00 0x01 0x02 0x03 0xCC 0xDD 0xEE 0xFF Data sent successfully.

Try to receive 8 bytes of data... Data received: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08

Press key to test specific function: Input 'A' 8-bit data under 9600 baudrate. Input 'B' 8-bit data under 115200 baudrate. Input 'C' 8-bit data under 345600 baudrate. Input 'D' 8-bit data under 1M baudrate. Input 'E' 8-bit data under 2M baudrate. Input 'F' 8-bit data under customized baudrate. Press ESC key to back to the top level case list.

测试分析:

PC 端与 EVB 端互相收发数据均正常,符合预期。

注意:当使用 USB 转串口板与 PC 通信时,某些品牌的转板可能无法支持非标准波特率(1M、2M 等)通信。

#### 2.4.4.6 波特率随意指定

#### 测试目的:

验证自定义波特率是否工作正常。

#### 测试预期:

自定义波特率下能够准确收发数据。

#### 测试现象:

PC 端, 串口调试助手先将波特率设置为一个自定义值, 如 500000, 然后打开串口准备接收数据。

EVB 端,输入 'F'命令,按照提示再输入 '500000'并回车,此时串口调试助手收到 8 字节的数据,检查发现均正确;随后使用串口调试助手发送 8 字节数据,从 EVB 端的 Log 可以看到也有准确接收到。



## PAN1080 HAL UART Sample

	串口调试助手	
串口设置	数据日志↓用户支持↓	VartAssist V4.3.1
	[2020-08-04 13:54:58.356]# RECV HEX>	~
波特率   500000 _	00 01 02 03 CC DD EE FF	
校验位 NONE	[2020-08-04 13:55:00.163]# SEND HEX>	
数据位 8	01 02 03 04 05 06 07 08	
停止位 1 一		
<b>●</b> 关闭		
C ASCII @ HEX	m	
▶ 按日志模式显示		
▶ 接收完自动换行		
□ 接收转向至文件		
甘油洪而、海险按收		
发送设置		
○ ASCII (● HEX □ 白計解析時义符	1	
	4446422	
「 AT指令自动回车		5.GND● 6. ♦ 清际 1_清际
□ AT指令自动回车 □ 自动发送校验位	叙述反达   1.DCD● 2.RXD● 3.TXD● <u>4.DTR</u> ●     1 2 3 4 5 6 7 8     1 2 3 4 5 6 7 8	5.GND● 6. ♦ 清除 1_ 清除
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> </ul>	叙境友达   1.DCD● 2.RXD● 3.TXD● <u>4.DTR</u> ● 1 2 3 4 5 6 7 8	5.GND ● 6. ◆ 清除 1_ 清际 发送
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> </ul>	叙境友达   1.DCD● 2.RXD● 3.TXD● <u>4.DTR</u> ● 1 2 3 4 5 6 7 8	5.GND ● 6. ↓ 清除 1_ 清除 发送
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li>☆捷定义 历史发送</li> </ul>	叙境友达   1.DCD● 2.RXD● 3.TXD● <u>4.DTR</u> ● 1 2 3 4 5 6 7 8	5.GND ● 6. ◆ 清除 1_ 清除 发送
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li>快捷定义 历史发送</li> <li>■ 6次送的字节数</li> </ul>	我描点法 1.DCD● 2.RXD● 3.TXD● 4.DTR●     1 2 3 4 5 6 7 8     35/29 RX:280	5.GHD ● 6. ◆ 清除 ℃ 清除 发送 TX:232 <u>复位计数</u>
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li>快捷定义 历史发送</li> <li>□ 6发送的字节数</li> </ul>	我提表法 1.DCD● 2.RXD● 3.TXD● 4.DTR●     1 2 3 4 5 6 7 8     35/29 RX:280	5. GHD ● 6. ◆ 清除 1 清除 发送 TX:232 复位计数
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li><u>快捷定义 历史发送</u></li> <li>I 已发送的字节数</li> <li>f</li> <li>Please input new b</li> </ul>	<pre>31. MAALE   1. DCD● 2. RXD● 3. TXD● 4. DTR● 1 2 3 4 5 6 7 8 35/29 RX:280 audrate (Press <enter> to confirm):</enter></pre>	5. GHD ● 6. ◆ 清除 ℃ 清除 发送 TX:232 复位计数
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li><u>快捷定义</u> 历史发送</li> <li>I C发送的字节数</li> <li>f</li> <li>please input new br 500000</li> <li>Baudrate changed, place</li> </ul>	<pre>31.144 (Press <enter> to confirm): new value: 500000</enter></pre>	5. GND ● 6. ◆ 清保 1 清除 发送 TX:232 <u>复位计数</u>
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li>快捷定义 历史发送</li> <li>● 已发送的字节数</li> <li>f</li> <li>please input new b</li> <li>500000</li> <li>Baudrate changed, in</li> <li>Send data:</li> </ul>	<pre>31括反法 1.DCD● 2.RXD● 3.TXD● 4.DTR● 1 2 3 4 5 6 7 8 35/29 RX:280 audrate (Press <enter> to confirm): new value: 500000</enter></pre>	5. GHD ● 6. ◆ 清除 1 清除 发送 TX:232 <u>复位计数</u>
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li>快捷定义 历史发送</li> <li>☞ 已发送的字节数</li> <li>f</li> <li>Please input new b</li> <li>500000</li> <li>Baudrate changed, i</li> <li>Send data: 0x00 0x01</li> <li>Data sent successf</li> </ul>	<pre>     #U構成法   1.DCD ● 2.RXD ● 3.TXD ● 4.DTR ●     1 2 3 4 5 6 7 8     35/29     RX:280  audrate (Press <enter> to confirm): new value: 500000 0x02 0x03 0xCC 0xDD 0xEE 0xFF ully. </enter></pre>	5. GHD ● 6. ◆ 清除 1 清除 发送 TX:232 复位计数
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li><u>快捷定义 历史发送</u></li> <li>☞ 已发送的字节数</li> <li>f</li> <li>Please input new b</li> <li>500000</li> <li>Baudrate changed, 0</li> <li>Send data:</li> <li>0x00 0x01 0</li> <li>Data sent successf</li> <li>Try to receive 8 b</li> </ul>	秋焼反法 1.DCD ● 2.RXD ● 3.TXD ● 4.DTR ●     1 2 3 4 5 6 7 8     35/29 RX:280	5. GHD ● 6. ◆ 清除 1 清除 发送 TX:232 夏位计数
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li>快捷定义 历史发送</li> <li>☞ 已发送的字节数</li> <li>f</li> <li>Please input new b</li> <li>500000</li> <li>Baudrate changed, m</li> <li>Send data:</li> <li>0x00 0x01</li> <li>Data sent successf</li> <li>Try to receive 8 b</li> <li>Data received: 0x01</li> </ul>	<pre>     #U構成法 1.DCD ● 2.RXD ● 3.TXD ● 4.DTR ●     1 2 3 4 5 6 7 8     35/29 RX:280     audrate (Press <enter> to confirm):     new value: 500000     0x02 0x03 0xCC 0xDD 0xEE 0xFF     ully.     ytes of data     1 0x02 0x03 0x04 0x05 0x06 0x07 0x08 </enter></pre>	5. GHD ● 6. ◆ 清除 € 清除 发送 TX:232 <u>复位计数</u>
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li><u>快捷定义 历史发送</u></li> <li>☞ 已发送的字节数</li> <li>f</li> <li>Please input new b</li> <li>500000</li> <li>Baudrate changed,</li> <li>Send data: 0x00 0x01</li> <li>Data sent successf</li> <li>Try to receive 8 b</li> <li>Data received: 0x00</li> <li>Press key to 1</li> </ul>	<pre>     #U構成法   1.DCD ● 2.RXD ● 3.TXD ● 4.DTR ●     1 2 3 4 5 6 7 8     35/29     RX:280  audrate (Press <enter> to confirm): new value: 500000 0x02 0x03 0xcc 0xDD 0xEE 0xFF ully. ytes of data 1 0x02 0x03 0x04 0x05 0x06 0x07 0x08 test specific function: </enter></pre>	5. GHD ● 6. ◆ 清除 € 清除 发送 TX:232 复位计数
<ul> <li>▲T指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li>快捷定义 历史发送</li> <li>☞ 已发送的字节数</li> <li>f</li> <li>please input new b</li> <li>500000</li> <li>Baudrate changed,</li> <li>Send data:</li> <li>0x00 0x01</li> <li>Data sent successf</li> <li>Try to receive 8 b</li> <li>Data received: 0x02</li> <li>Press key to 1</li> <li>Input 'A'</li> <li>Trout 'A'</li> </ul>	30.000 3.TXD ● 4.DTK ●     1 2 3 4 5 6 7 8     1 2 3 4 5 6 7 8     35/29 RX:280     35/29 RX:280 RX:	5. GHD ● 6. ◆ 清除 ℃ 清除 发送 TX:232 复位计数
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li>快捷定义 历史发送</li> <li>☞ 已发送的字节数</li> <li>f</li> <li>Please input new b</li> <li>500000</li> <li>Baudrate changed,</li> <li>Send data:</li> <li>0x00 0x01</li> <li>Data sent successf</li> <li>Try to receive 8 b</li> <li>Data received: 0x01</li> <li>Press key to 1</li> <li>Press key to 1</li> <li>Input 'A'</li> <li>Input 'A'</li> <li>Input 'A'</li> <li>Input 'A'</li> <li>Input 'A'</li> <li>Input 'A'</li> </ul>	Alta LE       1.DCD • 2.RXD • 3.TXD • 4.DTR •         1 2 3 4 5 6 7 8         1 2 3 4 5 6 7 8         35/29         RX:280         audrate (Press <enter> to confirm):         new value: 500000         0x02 0x03 0xCC 0xDD 0xEE 0xFF         ully.         ytes of data         1 0x02 0x03 0x04 0x05 0x06 0x07 0x08         test specific function:         8-bit data under 9600       baudrate         8-bit data under 9600       baudrate         8-bit data under 345600       baudrate</enter>	5. GHD ● 6. ◆ 清除 ℃ 清除 发送 TX:232 <u>复位计数</u>
<ul> <li>□ AT指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li><u>快捷定义</u> 历史发送</li> <li>☞ 已发送的字节数</li> </ul> f Please input new b. 500000 Baudrate changed, Send data: <ul> <li>0x00 0x01</li> </ul> Data sent successf Try to receive 8 by Data received: 0x02 F Press key to 1 <ul> <li>Input 'A'</li> <li>Input 'B'</li> <li>Input 'C'</li> <li>Input 'D'</li> <li>Input 'E'</li> </ul>	Alta LE       1. DCD • 2. RXD • 3. TXD • 4. DTR •         1 2 3 4 5 6 7 8         1 2 3 4 5 6 7 8         35/29         RX:280         audrate (Press <enter> to confirm):         new value: 500000         0x02 0x03 0xCC 0xDD 0xEE 0xFF         ully.         ytes of data         1 0x02 0x03 0xCC 0xDD 0xEE 0xFF         ully.         ytes of data         1 0x02 0x03 0x04 0x05 0x06 0x07 0x08         test specific function:         8-bit data under 9600       baudrate         8-bit data under 115200       baudrate         8-bit data under 2M       baudrate</enter>	5. GHD ● 6. ◆ 清除 ℃ 清除 发送 TX:232 夏位计数
<ul> <li>▲T指令自动回车</li> <li>□ 自动发送校验位</li> <li>□ 打开文件数据源</li> <li>□ 循环周期 1000 ms</li> <li>使捷定义 历史发送</li> <li>☞ 已发送的字节数</li> <li>f</li> <li>Please input new b</li> <li>500000</li> <li>Baudrate changed,</li> <li>Send data:</li> <li>0x00 0x01</li> <li>Data sent successfi</li> <li>Try to receive 8 b</li> <li>Data received: 0x02</li> <li>Fress key to</li> <li>Input 'A'</li> <li>Input 'A'</li> <li>Input 'A'</li> <li>Input 'A'</li> <li>Input 'C'</li> <li>Input 'E'</li> <li>Input 'F'</li> <li>Press ESC key</li> </ul>	Altic LE       1.000 2.RXD 3.TXD 4.0TK	5. GHD ● 6. ◆ 清除 ℃ 清除 发送 TX:232 <u>复位计数</u>



#### 测试分析:

PC 端与 EVB 端互相收发数据均正常,符合预期。

**注意:**当使用 USB 转串口板与 PC 通信时,某些品牌的转板可能无法支持非标准波特率(1M、2M 等)通信。

#### 2.4.5 Polling send and receive data UART\_TX\_Rx\_Polling\_TestCase4();

在主菜单下,输入'4'命令 进入 Subcase 菜单:

Press key to test specific function: < Baudrate=115200, Parity=ODD, DataBits=8, StopBits=1 > Input 'A' Send data. Input 'B' Prepare to receive data. Press ESC key to back to the top level case list.

#### 2.4.5.1 Polling send HAL\_UART\_SendData()

测试目的:

验证 Polling 模式下,发数据是否正常。

测试预期:

收数据正常。

测试现象:

EVB 端,输入'A'命令,发送三组 0-9 的 ascii 码值,值如代码段数组所示。

PC端,串口调试助手接收3组0-9,串口工具已转义为三组0-9的30个数值。

Press key to test specific function: < Baudrate=115200, Parity=ODD, DataBits=8, StopBits=1 > Input 'A' Send data. Input 'B' Prepare to receive data. Press ESC key to back to the top level case list.

0123456789012345678901234567890123456789



#### 测试分析:

PC 端准确接收到,符合预期。

#### 2.4.5.2 Polling receive HAL\_UART\_ReceiveData()

测试目的:

验证 Polling 模式下,收数据是否正常。

#### 测试预期:

发数据正常。

#### 测试现象:

EVB 端, 输入'B'命令, 等待接受数据。

PC 端,串口调试助手打开串口发送数据。以16 进制发送如下256 个数据:

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF

EVB 端,接收如上数据,并打印。



Press key to test specific function: < Baudrate=115200, Parity=ODD, DataBits=8, StopBits=1 > Input 'A' Send data. Input 'B' Prepare to receive data. Press ESC key to back to the top level case list.

0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0x0d 0x0e 0x0f 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1a 0x1b 0x1c 0x1d 0x1e 0x1f 0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2a 0x2b 0x2c 0x2d 0x2e 0x2f 0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3a 0x3b 0x3c 0x3d 0x3e 0x3f 0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5a 0x5b 0x5c 0x5d 0x5e 0x5f 0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6a 0x6b 0x6c 0x6d 0x6e 0x6f 0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7a 0x7b 0x7c 0x7d 0x7e 0x7f 0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89 0x8a 0x8b 0x8c 0x8d 0x8e 0x8f 0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9a 0x9b 0x9c 0x9d 0x9e 0x9f Oxa0 Oxa1 Oxa2 Oxa3 Oxa4 Oxa5 Oxa6 Oxa7 Oxa8 Oxa9 Oxaa Oxab Oxac Oxad Oxae Oxaf Oxb0 Oxb1 Oxb2 Oxb3 Oxb4 Oxb5 Oxb6 Oxb7 Oxb8 Oxb9 Oxba Oxbb Oxbc Oxbd Oxbe Oxbf OxeO Oxel Oxe2 Oxe3 Oxe4 Oxe5 Oxe6 Oxe7 Oxe8 Oxe9 Oxea Oxeb Oxee Oxed Oxee Oxef OxdO Oxd1 Oxd2 Oxd3 Oxd4 Oxd5 Oxd6 Oxd7 Oxd8 Oxd9 Oxda Oxdb Oxdc Oxdd Oxde Oxdf OxeO Oxe1 Oxe2 Oxe3 Oxe4 Oxe5 Oxe6 Oxe7 Oxe8 Oxe9 Oxea Oxeb Oxec Oxed Oxee Oxef OxfO Oxf1 Oxf2 Oxf3 Oxf4 Oxf5 Oxf6 Oxf7 Oxf8 Oxf9 Oxfa Oxfb Oxfc Oxfd Oxfe Oxff

测试分析:

PC 端发送 256 个字节数据, EVB 端准确接收到, 符合预期。

#### 2.4.6 Interrupt send and receive data UART\_TX\_Rx\_Interrupt\_TestCase5();

在主菜单下,输入'5'命令 进入 Subcase 菜单:

Press key to test specific function:

< Baudrate=115200, Parity=ODD, DataBits=8, StopBits=1 >

Input 'A' Prepare to interrupt send data. Input 'B' Prepare to interrupt receive data. Press ESC key to back to the top level case list.

#### 2.4.6.1 Interrupt send HAL\_UART\_SendData\_INT

#### 测试目的:

验证 Interrupt 模式下,发数据是否正常。

#### 测试预期:

收数据正常。

#### 测试现象:

EVB 端,输入'A'命令,发送三组 0-9 的 ascii 码值,值如代码段数组所示。

PC端,串口调试助手接收3组0-9,串口工具已转义为三组0-9的30个数值。并打印中断



PAN1080 HAL UART Sample

回调 Log。

Press key to test specific function: < Baudrate=115200, Parity=ODD, DataBits=8, StopBits=1 > Input 'A' Prepare to interrupt send data. Input 'B' Prepare to interrupt receive data. Press ESC key to back to the top level case list.

012345678901234567890123456789 Interrupt tx callback

在串口发送数据过程中,NBLK 引脚有电平变化,证明数据是以非阻塞的方式发送。



测试分析:

PC 端准确接收到,符合预期。

#### 2.4.6.2 Interrupt receive HAL\_UART\_ReceiveData\_INT()

#### 测试目的:

验证 Interrupt 模式下,收数据是否正常。

#### 测试预期:

发数据正常。

#### 测试现象:

EVB 端,输入'B'命令,等待接受数据。

PC 端,串口调试助手打开串口发送数据。以16 进制发送如下256 个数据:

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F



## PAN1080 HAL UART Sample

30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF

EVB 端,接收如上数据并打印。以及打印中断回调 Log。

Press key to test specific function: < Baudrate=115200, Parity=ODD, DataBits=8, StopBits=1 > Input 'A' Prepare to interrupt send data. Input 'B' Prepare to interrupt receive data. Press ESC key to back to the top level case list.

Prepare to receive data...

Non-blocking mode Interrupt rx callback Flag is UART\_CB\_FLAG\_RX\_FINISH 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0a 0x0b 0x0c 0x0d 0x0e 0x0f Ox10 Ox11 Ox12 Ox13 Ox14 Ox15 Ox16 Ox17 Ox18 Ox19 Ox1a Ox1b Ox1c Ox1d Ox1e Ox1f 0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2a 0x2b 0x2c 0x2d 0x2e 0x2f 0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3a 0x3b 0x3c 0x3d 0x3e 0x3f 0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 Ox58 Ox59 Ox5a Ox5b Ox5c Ox5d Ox5e Ox5f 0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6a 0x6b 0x6c 0x6d 0x6e 0x6f Ox78 Ox79 Ox7a Ox7b Ox7c Ox7d Ox7e Ox7f 0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 Ox88 Ox89 Ox8a Ox8b Ox8c Ox8d Ox8e Ox8f 0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99 0x9a 0x9b 0x9c 0x9d 0x9e 0x9f Oxa8 Oxa9 Oxaa Oxab Oxac Oxad Oxae Oxaf OxaO Oxa1 Oxa2 Oxa3 Oxa4 Oxa5 Oxa6 Oxa7 Oxb0 Oxb1 Oxb2 Oxb3 Oxb4 Oxb5 Oxb6 Oxb7 Oxb8 Oxb9 Oxba Oxbb Oxbc Oxbd Oxbe Oxbf OxeO Oxel Oxe2 Oxe3 Oxe4 Oxe5 Oxe6 Oxe7 Oxe8 Oxe9 Oxea Oxeb Oxee Oxed Oxee Oxef Oxdo Oxdi Oxd2 Oxd3 Oxd4 Oxd5 Oxd6 Oxd7 Oxd8 Oxd9 Oxda Oxdb Oxdc Oxdd Oxde Oxdf OxeO Oxe1 Oxe2 Oxe3 Oxe4 Oxe5 Oxe6 Oxe7 Oxe8 Oxe9 Oxea Oxeb Oxec Oxed Oxee Oxef OxfO Oxf1 Oxf2 Oxf3 Oxf4 Oxf5 Oxf6 Oxf7 Oxf8 Oxf9 Oxfa Oxfb Oxfc Oxfd Oxfe Oxff

测试分析:



PC 端发送 256 个字节数据, EVB 端准确接收到, 符合预期。

#### 2.4.7 DMA send and receive data UART\_TX\_Rx\_DMA\_TestCase6();

在主菜单下,输入'6'命令 进入 Subcase 菜单:

Press key to test specific function:

< Baudrate=115200, Parity=ODD, DataBits=8, StopBits=1 >

Input'A' Prepare to DMA send data. Input'B' Prepare to DMA receive data.

Press ESC key to back to the top level case list.

#### 2.4.7.1 DMA send HAL\_UART\_SendData\_DMA()

测试目的:

验证 DMA 模式下,发数据是否正常。

测试预期:

收数据正常。

测试现象:

EVB 端,输入'A'命令,发送三组 0-9 的 ascii 码值,值如代码段数组所示。

PC 端,串口调试助手接收3组0-9,串口工具已转义为三组0-9的30个数值。并打印回调Log。

Press key to test specific function:

< Baudrate=115200, Parity=ODD, DataBits=8, StopBits=1 >

```
Input 'A' Prepare to DMA send data.
Input 'B' Prepare to DMA receive data.
```

Press ESC key to back to the top level case list.

0123456789012345678901234567890123456789DMA tx callback

在串口发送数据过程中,NBLK 引脚有电平变化,证明数据是以非阻塞的方式发送。





#### 测试分析:

PC 端准确接收到,符合预期。

#### 2.4.7.2 DMA receive HAL\_UART\_ReceiveData\_DMA()

测试目的:

验证 DMA 模式下,收数据是否正常。

#### 测试预期:

发数据正常。

#### 测试现象:

EVB 端,输入'B'命令,等待接受数据。

PC 端,串口调试助手打开串口发送数据。以16 进制发送如下256 个数据:

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F

30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F

50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F

60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F

70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F

80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F

90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F

A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF

B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF





C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF

EVB 端,接收如上数据并打印。以及打印回调 Log。

Press key to test specific function: < Baudrate=115200, Parity=ODD, DataBits=8, StopBits=1 > Input 'A' Prepare to DMA send data. Input 'B' Prepare to DMA receive data. Press ESC key to back to the top level case list.

Prepare to receive data...

Non-blocking modeUMA rx callback Flag is UART\_CB\_FLAG\_UMA OxOO 0xO1 0xO2 0xO3 0xO4 0xO5 0xO6 0xO7 0xO8 0xO9 0xOa 0xOb 0xOc 0xOd 0xOe 0xOf 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19 0x1a 0x1b 0x1c 0x1d 0x1e 0x1f 0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29 0x2a 0x2b 0x2c 0x2d 0x2e 0x2f 0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x3a 0x3b 0x3c 0x3d 0x3e 0x3f 0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59 0x5a 0x5b 0x5c 0x5d 0x5e 0x5f 0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69 0x6a 0x6b 0x6c 0x6d 0x6e 0x6f 0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7a 0x7b 0x7c 0x7d 0x7e 0x7f 0x80 0x81 0x82 0x83 0x94 0x85 0x86 0x87 0x88 0x89 0x8a 0x8b 0x8c 0x8d 0x8e 0x8f 0x90 0x91 0x92 0x93 0x94 0x85 0x66 0x67 0x68 0x69 0x9a 0x9b 0x9c 0x9d 0x9e 0x9f 0xa0 0xa1 0xa2 0xa3 0x44 0xa5 0x66 0x67 0x68 0x69 0x6a 0x6b 0x6c 0x6d 0x6e 0x6f 0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79 0x7a 0x7b 0x7c 0x7d 0x7e 0x7f 0x80 0x81 0x82 0x83 0x94 0x95 0x96 0x97 0x98 0x99 0x9a 0x9b 0x9c 0x9d 0x9e 0x9f 0xa0 0xa1 0xa2 0xa3 0xa4 0xa5 0x66 0xa7 0x68 0x69 0x9a 0xaa 0xab 0xac 0xad 0xae 0xaf 0x40 0x41 0x42 0x33 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x40 0x41 0x42 0x33 0x44 0x45 0x46 0x47 0x88 0x99 0x9a 0x9b 0x9c 0x9d 0x9e 0x9f 0x00 0x11 0x22 0x3 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x40 0x41 0x42 0x33 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x40 0x41 0x42 0x33 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x40 0x41 0x42 0x33 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4b 0x4c 0x4f 0x40 0x41 0x42 0x33 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x40 0x41 0x42 0x33 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x40 0x41 0x42 0x33 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x40 0x41 0x42 0x33 0x44 0x45 0x46 0x47 0x48 0x49 0x4a 0x4b 0x4c 0x4d 0x4e 0x4f 0x40 0x41 0x42 0x33 0x44 0x45 0x46 0x47 0x4

测试分析:

PC 端发送 256 个字节数据, EVB 端准确接收到, 符合预期。



# 第3章 使用注意事项

- 1、波特率不可以设置太高,原因如果很高(如超过 apbclk/16)的时候,Uart Init flow 计算 integerdivider 会小于 100,导致 DLH 和 DLL 都是 0,uart 硬件会被认为不使能而不工作;
- 2、 在使用 USB 转串口转接板, CP2102 不支持非标准波特率, 而 CH340 是支持的, 因此测波 特率的 testcase 最好使用 CH340 来测;
- 3、Cob 上使用的 usb 转串口对波特率速率有一定影响,使用 usb 转串口最大不超过 600K,直接使用 CH340 接 IO 可以达到最大 3M 以上(再大工具不支持)
- 4、UART 中断 handler 中,某些情况下会触发 Uart\_event\_none (编号为 0x1) 这个 event,软件无需处理,直接忽略掉即可;
- 5、 Auto flow control, Rx FIFO data 超过阈值,硬件自动拉高 RTS 电平,随后在 Rx handler 中 需要将 FIFO 读空,硬件才会重新将 RTS 电平拉低;
- 6、使用 Auto flow control 或 Uart DMA,可以开 Uart 发送中断(IER.bit[1])/接收中断(IER.bit[0]), 然后在中断处理函数中处理 data,也可以不开这两个中断,直接使用主程序处理 data;
- 7、若 dma source/destination 是 memory,则对应的 srctfr/dsttfr 中断应当 disable (enable 无意义,因为 memory 不存在 transaction level 的概念)