I/O 模擬 UART 鮑率校準應用範例

文件編碼：AN0475T

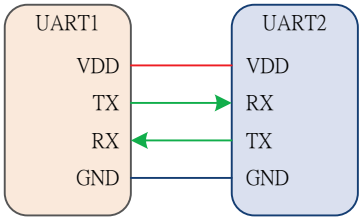
簡介

在 HOLTEK MCU 中並非每一顆都內含有非同步式串列資料傳輸(Universal Asynchronous Receiver/Transmitter, UART)功能，若需用到此功能可以利用 S/W 的方式模擬此功能，但 S/W 的精準度會依系統頻率的誤差而有所改變，本文主要來講解如何利用 S/W 方式來校準自己本身的鮑率，達到雙方溝通時不會造成錯誤，亦或者傳輸錯誤時可以再調整回與對方相同的傳輸頻率。

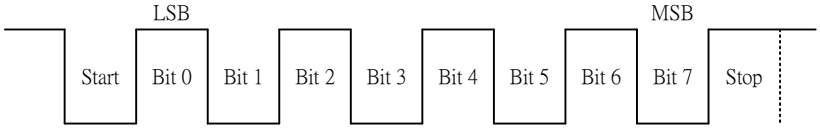
本文以 HT66F4540 為例，實際範例說明 UART 鮑率校準使用方法。

功能說明

1. UART 資料傳輸方式：UART 是使用分時傳送方式，由傳送端每隔一段時間就將一位元的資料狀態傳送給接收端，直到將這筆資料(8 位元)傳輸完畢為止，即完成一筆資料的傳遞，接收端的接收速度需與傳送端的相同，否則收到的資料就不正確。



2. UART 資料傳輸：UART 在資料傳輸時，會先發一個 Start(Low)訊號，來與接收端同步，再從資料的最低位元開始傳輸，直到整個 Byte 傳輸完畢後，會再發送一個 Stop(High)訊號，來告知接收端訊號完成資料傳輸。
   UART 資料傳輸格式為：1 Start Bit + 8 Data Bits + 1 Stop Bit，如下圖：



3. UART 傳輸速率：是以每秒傳幾個 bit 的方式來衡量傳輸速率，其單位：bit/sec，常稱為鮑率(Baudrate)或位元率(bit rate)。
   常用的鮑率：2400、9600、19200、38400，單位：bit/sec，轉換成一個 Bit 的傳輸時間如下表：

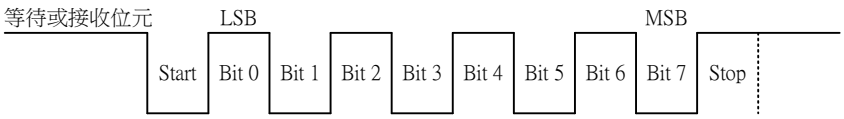
鮑率	一個 Bit 的傳輸時間(μs)
2400	416.67
9600	104.16
19200	52.08
38400	26.04

工作原理

UART 在等待接收或傳輸資料時，RX pin 與 TX pin 準位會維持在"High"的狀態，直到要接收或傳輸資料時，會先將準位拉成"Low"持續一個 Bit 的時間後也就是 START bit，用以與對方同步，由 Bit 0 開始傳輸，整筆資料傳輸完畢後，RX pin 與 TX pin 準位回到"High"的狀態，所以我們在使用 I/O 模擬 UART 功能時，RX Pin 使用外部中斷腳，當有下降緣訊號進來，RX Pin 開始接收資料，而 TX Pin 則使用一般 I/O 功能腳即可，然而在偵測鮑率時，RX Pin 則使用一般 I/O 功能。

I/O 模擬 TX 傳輸資料步驟

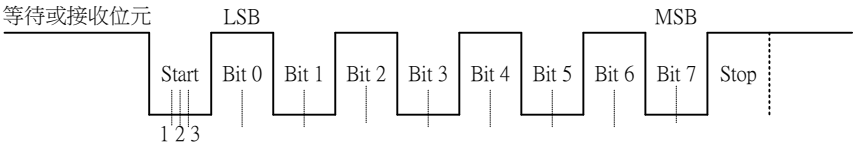
1. 準備傳輸時，TX 準位為 High。
2. 開始傳輸時，將 TX 準位拉 Low，一個 Bit 的時間。(Start Bit)
3. 依序傳輸資料，第一筆為最低位元(Bit 0)資料，接續 Bit 1 → Bit 2 → Bit 3 → Bit 4 → Bit 5 → Bit 6 → Bit 7。
4. 資料傳輸完畢後，將 TX 準位拉 High，一個 Bit 的時間。(Stop Bit)
5. 可以傳輸下一筆資料。



I/O 模擬 RX 接收資料步驟

1. 準備接收時，RX 準位為 High。
2. 當 RX 接收到一個下降緣訊號時，開始計數約 0.4bit 傳輸時間判斷訊號是否為 Low，是 Zero+1，否 One+1，接著繼續判斷 3 次後，看 Zero 是否大於等於 2，是的話代表是正確的數據，否則可能為其他訊號。
3. 確認為正確數據時，延遲 1 bit 傳輸時間，依序接收資料，第一筆為最低位元(Bit 0)資料，接續 Bit 1 → Bit 2 → Bit 3 → Bit 4 → Bit 5 → Bit 6 → Bit 7，前面已延遲約 0.5~0.6 bit 時間，接著延遲 1 bit 時間，是為了在資料的中心抓資料，以免抓取資料是前面或後面 1 筆的資料。

4. 資料接收完畢後，將 RX 準位拉 High。
5. 可以準備接收下一筆資料。



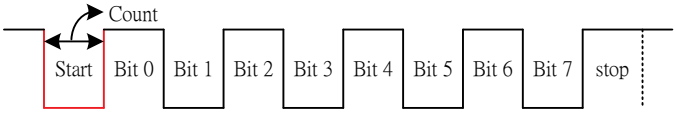
RX 接收鮑率步驟

方法一

1. 準備接收時，RX 準位為 High。
2. 當 RX 接收到 Low 訊號時，開始計數，計數為 Count。
3. 當 RX 接收到 High 訊號時，結束計數。
4. 接收完畢，運用此計數到的值，做一些運算後，作為資料傳輸/接收的時間。

   Note：a. 此方法要確定 Bit 0 為 High，這樣接收的數據才會正確。

   b. 計數迴圈為 4 個指令為一個組。

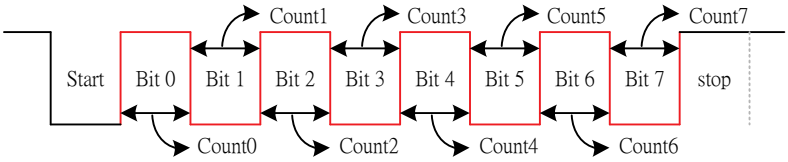


方法二

1. 準備接收時，RX 準位為 High。
2. 當 RX 接收到 Low 訊號時，代表 Start 訊號進來，可以準備開始計數。
3. 當 RX 接收到 High 訊號時，開始計數，計數為 Count0。
4. 當 RX 接收到 Low 訊號時，開始計數，計數為 Count1。
5. 當 RX 接收到 High 訊號時，開始計數，計數為 Count2。
6. 當 RX 接收到 Low 訊號時，開始計數，計數為 Count3。
7. 當 RX 接收到 High 訊號時，開始計數，計數為 Count4。
8. 當 RX 接收到 Low 訊號時，開始計數，計數為 Count5。
9. 當 RX 接收到 High 訊號時，開始計數，計數為 Count6。
10. 當 RX 接收到 Low 訊號時，開始計數，計數為 Count7。
11. 當 RX 接收到 High 訊號時，結束計數。
12. 接收完畢，運用此計數到的值 Count0~Count7，相加後取平均，之後做一些運算後，作為資料傳輸/接收的時間。

    Note：a. 此方法要確定資料為 55H，這樣接收的數據才會正確。

    b. 計數迴圈為 4 個指令為一個組。



方法三

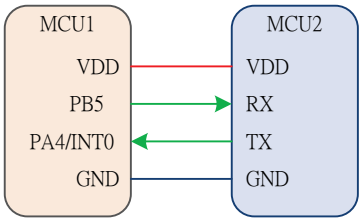
運用 TimeBase 或 TM 去計數一個 Bit 的傳輸時間，並去換/運算作為資料傳輸/接收的時間。

Note：此範例程式是使用方法二來做示範，其他方法在此沒有多做解釋及示範。

硬體說明

MCU1 的 VDD 與 MCU2 的 VDD 對接；MCU1 的 GND 與 MCU2 的 GND 對接；

MCU1 的 TX(PB5)與 MCU2 的 RX 對接；MCU1 的 RX(PA4)與 MCU2 的 TX 對接；



軟體說明

MCU 初始設定如下表：

步驟	操作內容	暫存器	設定位元	功能描述
1	設定 HIRC 使用頻率	SCC	CKS[2:0]	系統時鐘選擇：f_H CKS[2:0] = 000
		SCC	FHS	高頻時鐘選擇：HIRC FHS = 0
		HIRCC	HIRC[1:0]	HIRC 頻率選擇：8MHz HIRC[1:0] = 10
		HIRCC	HIRCEN	HIRC 振盪器設定：使能 HIRCEN = 1
2	設定腳位	PAS1	PAS1[1:0]	PA4 引腳功能設定：INT0 / PA4 PAS1[1:0] = 00
		PBS1	PBS1[3:2]	PB5 引腳功能設定：PB5 PBS1[3:2] = 00
		PAC	PAC4	PA4 引腳類型設定：輸入口 PAC4 = 1
		PBC	PBC5	PB5 引腳類型設定：輸出口 PBC5 = 0
		PAPU	PAPU4	PA4 引腳上拉電阻設定：使用上拉電阻 PAPU4 = 1
		PBPU	PBPU5	PB5 引腳上拉電阻設定：使用上拉電阻 PBPU5 = 1

步驟	操作內容	暫存器	設定位元	功能描述
3	設定看門狗	WDTC	WE[4:0]	設定看門狗：使能 WE[4:0] = 01010
			WS[2:0]	看門狗 Time-Out 週期設定：$2^{18}/f_{LIRC}$ WS[2:0] = 111
4	中斷	INTC0	INT0E	外部中斷 0 開關設定：Enable INT0E = 1
		INTC0	INT0F	外部中斷 0 旗標設定：0 INT0F = 0
		INTC0	EMI	總中斷開關設定：Enable EMI = 1
		INTEG	INT0S[1:0]	外部中斷 0 中斷 Edge 設定：Falling Edge INT0S[1:0] = 01

自定義暫存器名稱說明：

項目	暫存器	位元數	功能描述	備註
1	RX_DATA	8	接收的資料	
2	RX_DATA_ITEMS	8	接收資料位元數	
3	TX_DATA	8	傳送的資料	
4	TX_DATA_ITEMS	8	傳送資料位元數	
5	ONE	8	半週期為"1"的計數值	
6	ZERO	8	半週期為"0"的計數值	
7	RX_FG	1	接收完成旗標	
8	COUNT0	8	Bit 0 的計數寬度	
9	COUNT1	8	Bit 1 的計數寬度	
10	COUNT2	8	Bit 2 的計數寬度	
11	COUNT3	8	Bit 3 的計數寬度	
12	COUNT4	8	Bit 4 的計數寬度	
13	COUNT5	8	Bit 5 的計數寬度	
14	COUNT6	8	Bit 6 的計數寬度	
15	COUNT7	8	Bit 7 的計數寬度	
16	COUNT	16	Bit 0 ~ 7 的計數寬度總和/平均	COUNT_L
				COUNT_H
17	BR_ONE_BIT	8	一個週期的寬度	
18	BR_HALF_BIT	8	半個週期的寬度	

使用注意事項：

1. 在使用本程式時，要注意系統頻率與鮑率的選擇，因在軟體設計中只有用一個自定義暫存器來儲存鮑率參數，運算時是使用 4 個指令週期為一個 LOOP，故最大只能接受 1020 個指令週期，且在接收與傳輸程式中需運用到 20 個指令，故須特別注意，參考選擇如下：

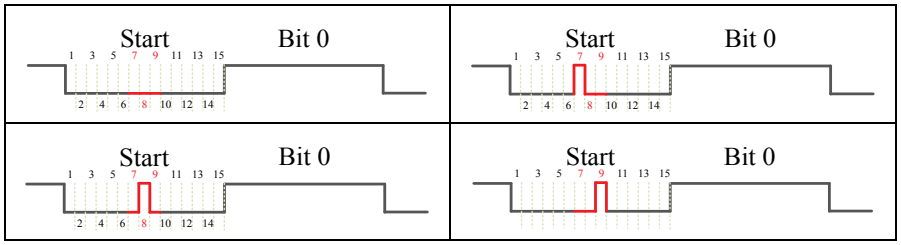
系統頻率	系統時間	1 個指令週期時間	不同鮑率可使用的指令週期數				
fsys	t_{SYS}(μs)	指令週期(μs)(4T)	2400 (416.67μs)	9600 (104.17μs)	19200 (52.08μs)	38400 (26.04μs)	115200 (8.68μs)
2M	0.5	2	208	52	26	13*	4*
4M	0.25	1	416	104	52	26	8*
8M	0.125	0.5	833	208	104	52	17*
16M	0.0625	0.25	1666**	416	208	104	34

Note：* 為受限於 20 個指令週期，故不能使用

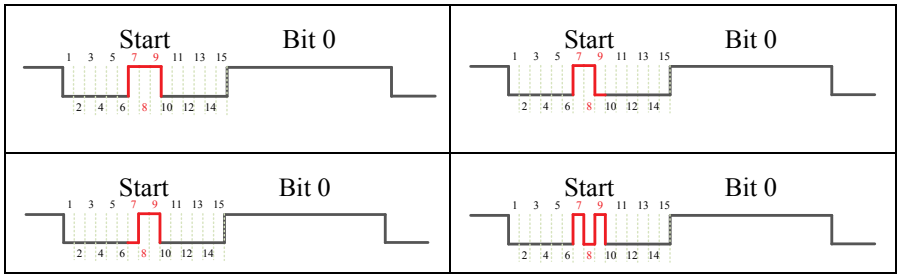
**為受限於 1020 個指令週期，故不能使用

2. 在接收資料時，會先收到一個 Low 的狀態位元，也就是 Start bit，怕在接收時有雜訊，此 Low 訊號並非為 Start bit，故會在接收 Start bit 訊號的過程中，加入 3 個位元的判斷，若有兩個以上(含兩個)零準位訊號，就判斷此訊號是正確的，可以繼續接收後面的訊號，但若只有一個零準位訊號，就代表此訊號不正確，所以不繼續接收，跳出等待下一次的訊號。如下圖：

(1) 正確訊號：



(2) 錯誤訊號：



流程圖

整體流程圖

步驟 1：初始化：清除 RAM 資料、設定系統頻率與設置看門狗、I/O 初始化，執行步驟 2。

步驟 2：讀取資料寬度，執行步驟 3。

步驟 3：計算鮑率，執行步驟 4。

步驟 4：設定外部中斷 0 開關與總中斷開關，執行步驟 5。

步驟 5：設定 UART 相關參數，執行步驟 6。

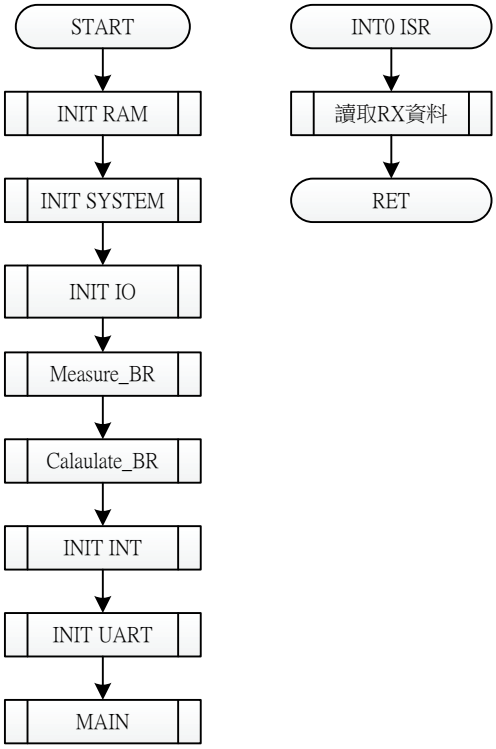
步驟 6：判斷 RX_FG 是否等於 1。

是：執行步驟 7。

否：執行步驟 6。

步驟 7：將接收到的資料回傳，並多傳送一筆 0xAA 的資料，執行步驟 6。

INT0 ISR：當 INT0 腳有下降緣訊號進來後，判斷 Start Bit 接收是否正確，Start Bit 接收正確，開始接收資料並將資料儲存於 RX_DATA 中並設立 RX_FG = 1，若 Start Bit 接收不正確，則跳出中斷。



讀取資料寬度流程圖

步驟 1：判斷 UART_RX_PIN = 1。

是：執行步驟 2，代表 UART_RX_PIN Ready。

否：執行步驟 1，代表 UART_RX_PIN 未 Ready。

步驟 2：判斷 UART_RX_PIN = 0。

是：執行步驟 3，代表有資料傳輸。

否：執行步驟 2。

步驟 3：判斷 UART_RX_PIN = 1。

是：執行步驟 4。

否：執行步驟 3，代表 START Bit 尚未傳輸完畢。

步驟 4：COUNT0 加 1，執行步驟 5。

步驟 5：判斷 UART_RX_PIN = 0。

是：執行步驟 6。

否：執行步驟 4，代表 Bit 0 尚未傳輸完畢。

步驟 6： COUNT1 加 1，執行步驟 7。

步驟 7：判斷 UART_RX_PIN = 1。

是：執行步驟 8。

否：執行步驟 6，代表 Bit 1 尚未傳輸完畢。

步驟 8：COUNT2 加 1，執行步驟 9。

步驟 9：判斷 UART_RX_PIN = 0。

是：執行步驟 10。

否：執行步驟 8，代表 Bit 2 尚未傳輸完畢。

步驟 10：COUNT3 加 1，執行步驟 11。

步驟 11：判斷 UART_RX_PIN = 1。

是：執行步驟 12。

否：執行步驟 10，代表 Bit 3 尚未傳輸完畢。

步驟 12：COUNT4 加 1，執行步驟 13。

步驟 13：判斷 UART_RX_PIN = 0。

是：執行步驟 14。

否：執行步驟 12，代表 Bit 4 尚未傳輸完畢。

步驟 14：COUNT5 加 1，執行步驟 15。

步驟 15：判斷 UART_RX_PIN = 1。

是：執行步驟 16。

否：執行步驟 14，代表 Bit 5 尚未傳輸完畢。

步驟 16：COUNT6 加 1，執行步驟 17。

步驟 17：判斷 UART_RX_PIN = 0。

是：執行步驟 18。

否：執行步驟 16，代表 Bit 6 尚未傳輸完畢。

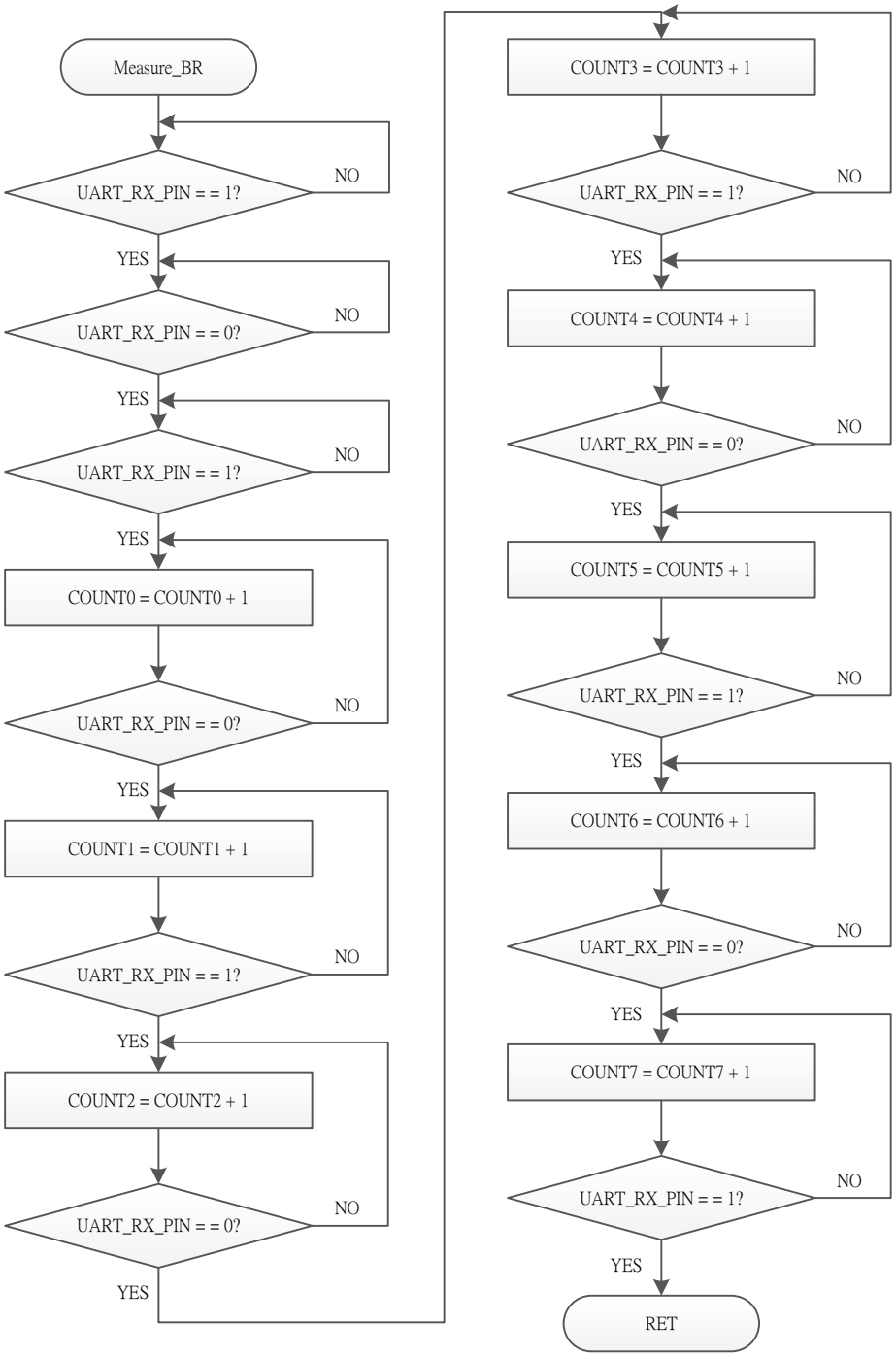
步驟 18：COUNT7 加 1，執行步驟 19。

步驟 19：判斷 UART_RX_PIN = 1。

是：執行步驟 20。

否：執行步驟 18，代表 Bit 7 尚未傳輸完畢。

步驟 20：返回。


Measure_BR
UART_RX_PIN = = 1?
NO
YES
UART_RX_PIN = = 0?
NO
YES
UART_RX_PIN = = 1?
NO
YES
COUNT0 = COUNT0 + 1
UART_RX_PIN = = 0?
NO
YES
COUNT1 = COUNT1 + 1
UART_RX_PIN = = 1?
NO
YES
COUNT2 = COUNT2 + 1
UART_RX_PIN = = 0?
NO
YES
COUNT3 = COUNT3 + 1
UART_RX_PIN = = 1?
NO
YES
COUNT4 = COUNT4 + 1
UART_RX_PIN = = 0?
NO
YES
COUNT5 = COUNT5 + 1
UART_RX_PIN = = 1?
NO
YES
COUNT6 = COUNT6 + 1
UART_RX_PIN = = 0?
NO
YES
COUNT7 = COUNT7 + 1
UART_RX_PIN = = 1?
NO
YES
RET

計算鮑率流程圖

步驟 1：將 COUNT0 ~ COUNT7 全部相加儲存於 COUNT，執行步驟 2。

步驟 2：將 COUNT 取平均儲存於 COUNT，執行步驟 3。

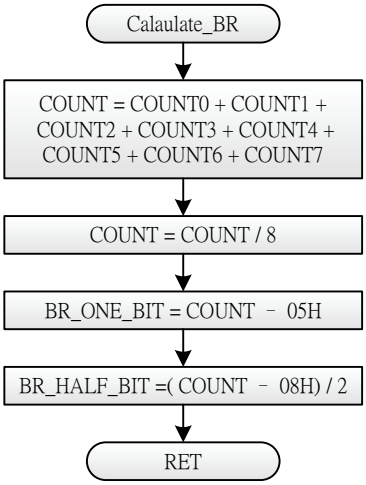
步驟 3：將 COUNT - 5 儲存於 BR_ONE_BIT，執行步驟 4。

步驟 4：將 (COUNT – 8) 除 2 後，儲存於 BR_HALF_BIT，執行步驟 5。

步驟 5：返回。

Note：

1. 在傳送或接收資料時，需 20 個指令週期，轉換為時間就是 5，故 COUNT 值要減掉 5 才會是正確的 BR_ONE_BIT 值。
2. 在半週期中會做三次判斷，判斷 START bit 是否正確，中間判斷用到 32 個指令週期，轉換為時間就是 8，故 COUNT 值要減掉 8 才會是正確的的 BR_HALF_BIT 值。



結論

本文主要是介紹 I/O 模擬 UART 功能時，若鮑率不準可以如何校準使傳輸正確，因 UART 開始傳輸時都會先拉一個 Low 的訊號，當作開始訊號，故可以以此訊號作為資料傳輸的資料寬度，但資料位元中 LSB 不是 High 的訊號這樣偵測上會有問題，所以提供 2 種方法來補償，一是固定資料 LSB 一定要為 High，另一種是固定資料為 55H，這兩種各有各優缺點，第一種，誤差會比較大，第二種因偵測八筆資料故誤差較小，也因本身沒有固定鮑率，所以，在使用時對方一定要先送一筆資料，建立起本身的傳輸速度。

參考資料

參考文件 HT66F45x0 Datasheet。

如需進一步瞭解，敬請瀏覽 Holtek 官方網站 http://www.holtek.com.tw。

版本及修改資訊

Date 日期	Author 作者	Issue 發行、修訂說明
2017.12.01	陳淑娟 (Chen, Shu-Juan)	第一版

免責聲明

本網頁所載的所有資料、商標、圖片、連結及其他資料等 (以下簡稱「資料」)，只供參考之用，盛群半導體股份有限公司 (以下簡稱「本公司」) 將會隨時更改資料，並由本公司決定而不作另行通知。雖然本公司已盡力確保本網頁的資料準確性，但本公司並不保證該等資料均為準確無誤。本公司不會對任何錯誤或遺漏承擔責任。

本公司不會對任何人士使用本網頁而引致任何損害 (包括但不限於電腦病毒、系統固障、資料損失) 承擔任何賠償。本網頁可能會連結至其他機構所提供的網頁，但這些網頁並不是由本公司所控制。本公司不對這些網頁所顯示的內容作出任何保證或承擔任何責任。

責任限制

在任何情況下，本公司並不須就任何人由於直接或間接進入或使用本網站，並就此內容上或任何產品、資訊或服務，而招致的任何損失或損害負任何責任。

管轄法律

本免責聲明受中華民國法律約束，並接受中華民國法院的管轄。

免責聲明更新

本公司保留隨時更新本免責聲明的權利，任何更改於本網站發佈時，立即生效。