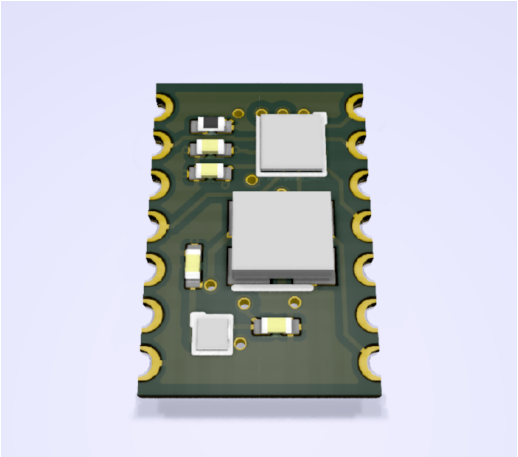
CD485 數據手冊

Duke Fong

June 4, 2017



Contents

1 功能描述

1.1 概述

CD485（又名 CD-BUS）是一種基於 RS485 總線的通訊協議，它同時也指實現此協議的硬件和使用此協議的總線。

CD485 協議由 Duke Fong 於 2009 年設計，實現便捷、自由、多主對等、高速率通訊。

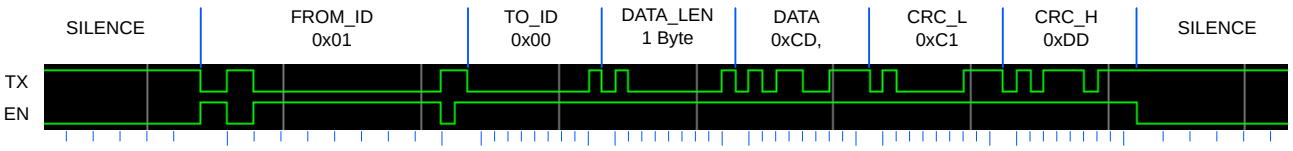
1.2 特色

當前硬件支持的特性:

- CD485 總線上各節點均可主動發起傳輸，通過節點地址仲裁以避免衝突。
- 總線上每個數據包可以含有 0 ~ 253 字節用戶數據
- 共有 8 個 RX 緩存頁和 2 個 TX 緩存頁，每一頁 256 字節
- 硬件爲每個數據包自動完成 16 位 CRC 生成與校驗
- 波特率：412 bps 至 9 Mbps（更換晶振可支持 10 Mbps; 且另有模組支持到 36 Mbps）
- 可分別爲仲裁字段和後續數據設置不同的波特率
- 兼容傳統 RS485 總線
- 提供 SPI 和 I2C 接口
- 配置和操作便捷

2 CD485 協議

參考時序:



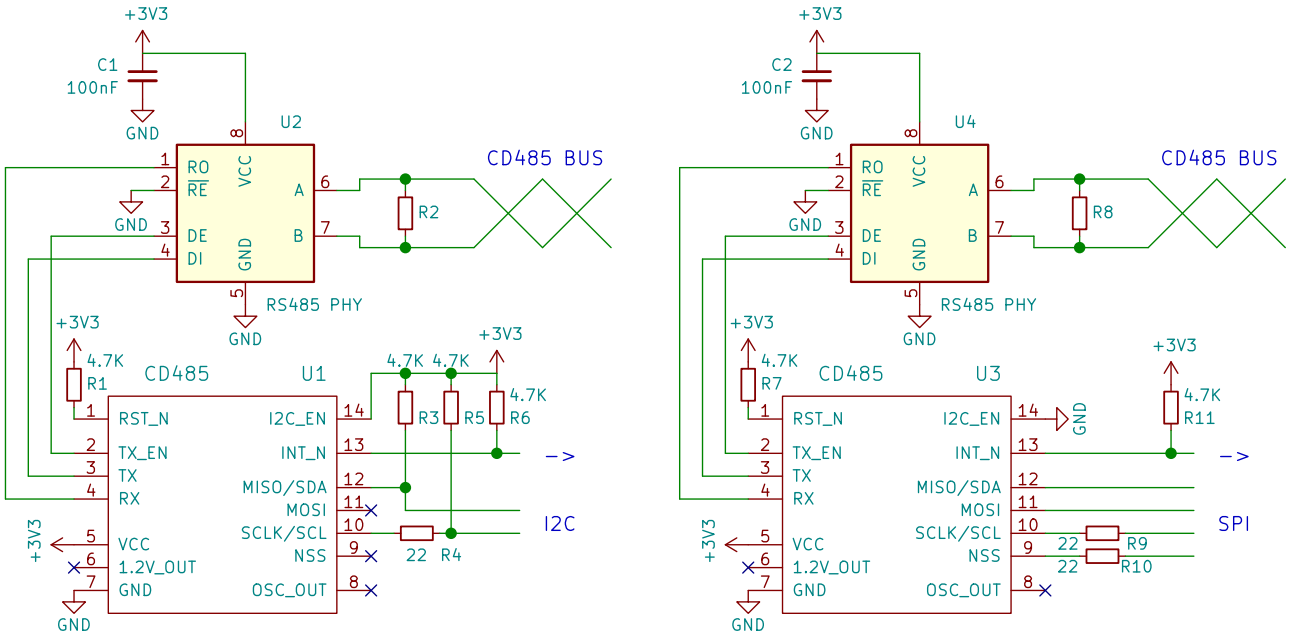
字段	長度 (bytes)	用途
SILENCE	0~25.5 Default: 2 (20 bits)	分隔數據包 總線在數據包結束後繼續保持 SILENCE 位長的時間爲 1, 總線便進入空閒模式。

FROM_ID	1	發送方 ID 當總線進入空閒並保持一段時間（默認設爲 10 bits 長），才允許發送。在發送此字段時，所有的 1 不使能 TX_EN 腳，從而回讀總線狀態以判斷是否有更高優先級節點同時在發送。若有，該節點立即停止並推後發送；若無，在最後一次回讀後使能 TX_EN 並保持到數據包結束。 優先級計算公式：$255 - bit\_reversal(\text{FROM\_ID})$ 此字段會在所有數據 1 的中間位置進行回讀，因爲發送與接收存在延時，如果波特率設的太高，回讀到的將會是前一位發送出的數據，所以通常設置低於 1 Mbps.
TO_ID	1	接收方 ID, 255 標識廣播包。
DATA_LEN	1	用戶數據長度，範圍：0~253 字節，每個緩存頁是 256 字節，最前 3 字節被 FROM_ID、TO_ID 和 DATA_LEN 使用。
DATA	0~253	用戶數據
CRC_L	1	CRC 低 8 位，與 Modbus 使用相同的 CRC 標準。
CRC_H	1	CRC 高 8 位

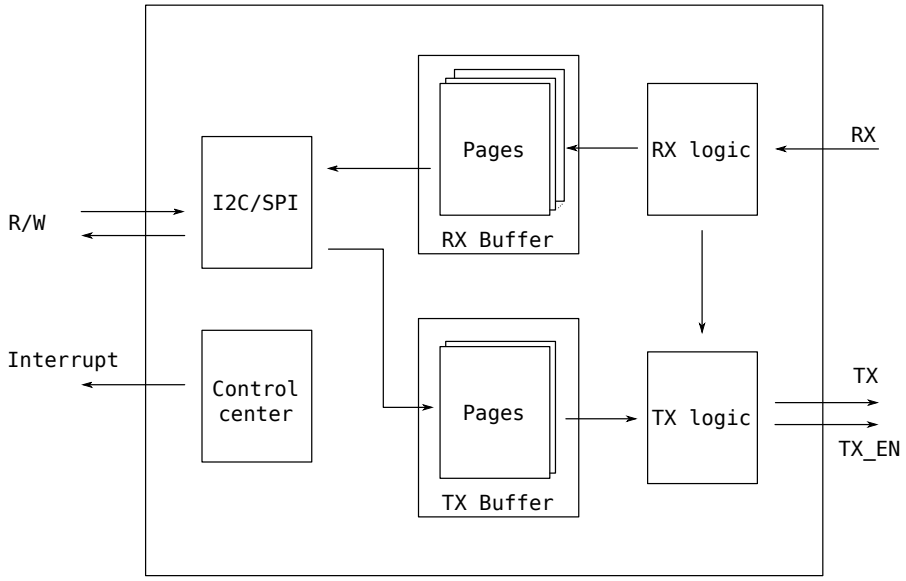
CD485 協議只定義數據包格式，不規定用戶數據格式；只支持单播和廣播，不支持多播；只提供硬件避讓、避讓後自動重傳，而應答及出錯處理則由上層用戶負責。

3 硬件

3.1 電路參考



3.2 內部結構



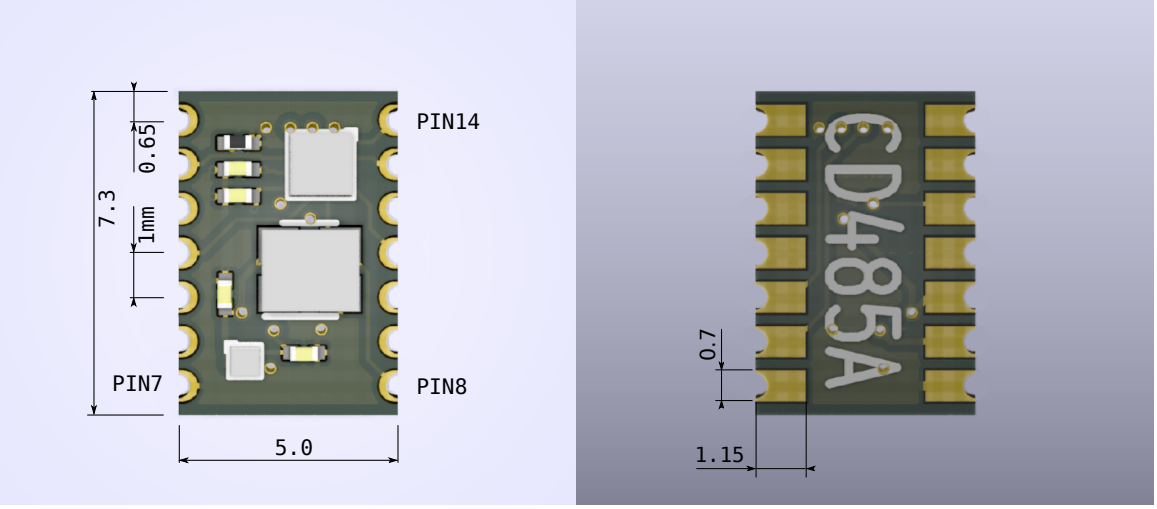
3.3 引腳定義

No	Name	I/O	Pull	Description
1	RST_N	I	-	復位，低有效，需要加外部上拉
2	TX_EN	O	D	使能腳，連接 RS485 PHY（內部 2 KΩ 下拉電阻）
3	TX	O	U	發送腳，連接 RS485 PHY
4	RX	I	-	接收腳，連接 RS485 PHY
5	VCC			電源 3.3V ±5%, ≤5mA
6	1.2V_OUT			內部 1.2V LDO 輸出
7	GND			地
8	OSC_OUT	O		內部 27MHz 時鐘輸出
9	NSS	I	U	SPI NSS（上電或復位時請勿拉低）
10	SCLK/SCL	I	U	SPI/I2C CLOCK
11	MOSI	I	-	SPI MOSI
12	MISO/SDA	I/IO	U	SPI MOSI / I2C SDA
13	INT_N	O	-	中斷，低有效，開漏輸出
14	I2C_SEL	I	-	輸入高選擇 I2C 模式；低爲 SPI 模式

上拉均爲 50 KΩ，且僅在復位結束後有效。

注意：在上電或復位時的一小段時間，TX 腳會輸出低、NSS 輸出高、SCLK/SCL 輸出一段脈衝，在這段時間裏請不要驅動這些引腳，最好在模塊和 CPU 間爲 NSS 和 SCLK/SCL 串電阻做保護。

3.4 尺寸規格



4 寄存器列表

地址	名稱	R/W	說明
0x00	VERSION	R	硬件版本，當前爲：0xC1
0x01	SETTING	RW	Bits: bit0 OUTPUT_EN 如未置位，TX 和 TX_EN 輸出高阻。 bit1 NO_ARBITRATE 關閉仲裁功能，輸出時 TX_EN 一直有效。 bit2 USER_CRC 關閉硬件 CRC 如果關閉，用戶需要自行計算兩字節 CRC 並追寫在數據之後，且在讀完數據之後再多讀兩字節 CRC 數據以便自行校驗。用戶數據最大長度將會降至 251 字節。 bit3 NO_DROP 如果設置，當接收出錯置位 RX_ERROR 標誌時會同時保留出錯的數據包。 通過 RX_PAGE_FLAG 判斷當前 RX 緩存頁中的數據包是否有錯。 bit[5:4] TX_EN_ADVANCE（僅 NO_ARBITRATE 置位時有效） TX_EN 提前 TX 使能的位長（額外加上 1 個系統時鐘週期）。 默認：xx01x000 (x: 不關心，寫 0)
0x02	SILENCE_LEN	RW	總線在數據包結束後繼續保持 SILENCE 位長的時間爲 1, 總線便進入空閒模式，默認 20 (bits)

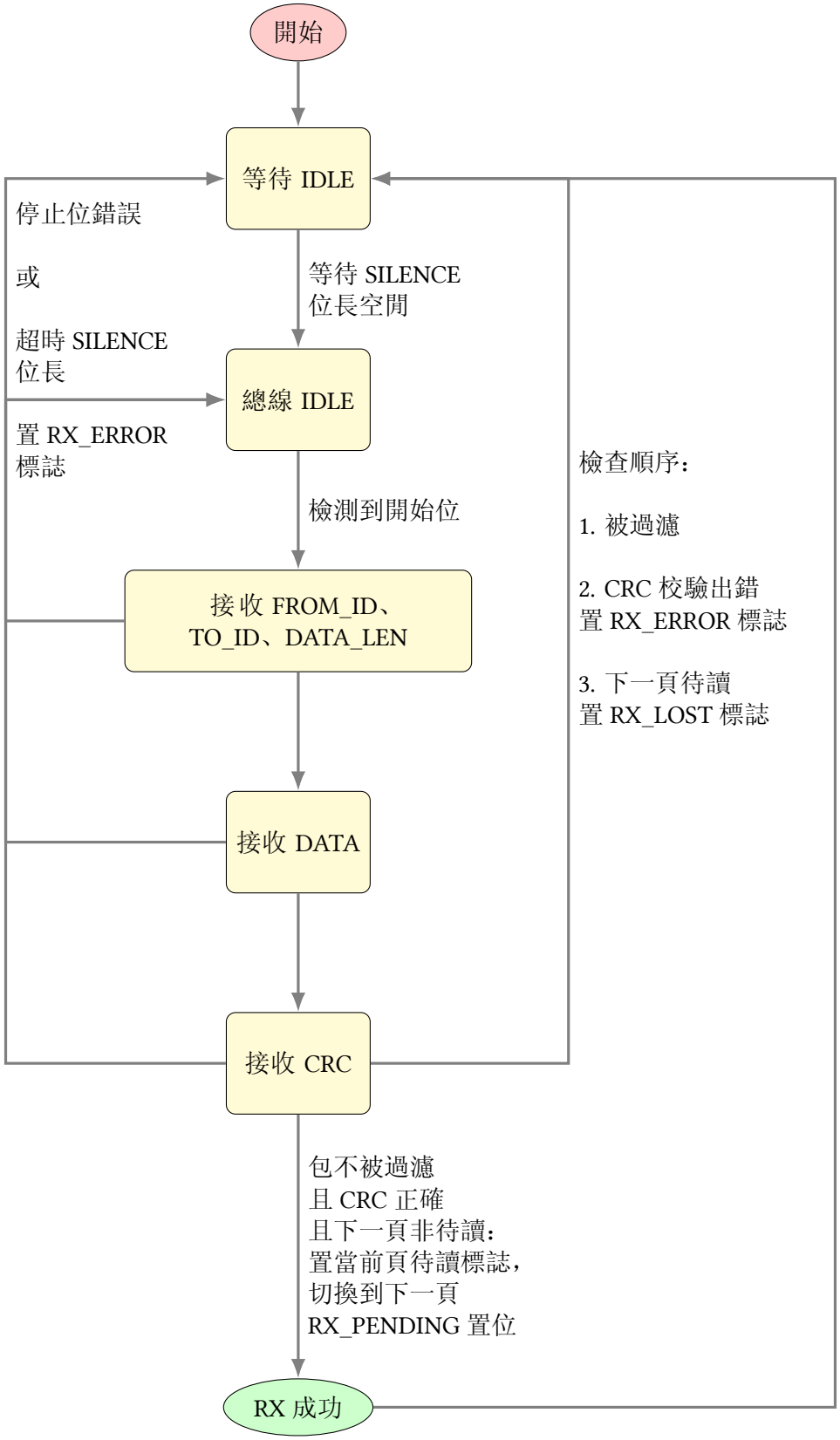
<table>
<tr><td>0x03</td><td>TX_DELAY</td><td>RW</td><td>當總線進入空閒並保持此段時間，才允許發送，默認 10 (bits)
可以爲越高優先級節點設置越低的值，但至少要保留 1 bit, 以確保所有節點都有足夠時間檢測到總線 IDLE 狀態。</td></tr>
<tr><td>0x04</td><td>SELF_ID</td><td>RW</td><td>僅用做接收過濾：（由上至下進行匹配）
<table><tr><th>FROM_ID</th><th>TO_ID</th><th>SELF_ID</th><th>接收或丟棄</th></tr><tr><td>not care</td><td>not care</td><td>255</td><td>接收（嗅探模式）</td></tr><tr><td>= SELF_ID</td><td>not care</td><td>!= 255</td><td>丟棄（避免環路）</td></tr><tr><td>!= SELF_ID</td><td>255</td><td>not care</td><td>接收（廣播）</td></tr><tr><td>!= SELF_ID</td><td>!= 255</td><td>= TO_ID</td><td>接收（點對點）</td></tr><tr><td>not care</td><td>!= 255</td><td>!= TO_ID</td><td>丟棄</td></tr></table>Default: 255</td></tr>
<tr><td>0x05</td><td>PERIOD_LS_L</td><td>RW</td><td>PERIOD_LS 低 8 位 (LS: Low Speed)
爲 SILENCE、TX_DELAY 和 FROM_ID 字段設置波特率（也包括 EN_ADVANCE）。
計算公式 $factor = sysclock \div bond\_rate - 1$
舉例：當前系統時鐘是 27 MHz，設置 233 最接近 115200 bps（默認 233）</td></tr>
<tr><td>0x06</td><td>PERIOD_LS_H</td><td>RW</td><td>PERIOD_LS 高 8 位，共 16 位（默認 0）</td></tr>
<tr><td>0x07</td><td>PERIOD_HS_L</td><td>RW</td><td>PERIOD_HS 低 8 位 (LS: Low Speed, 默認 233)
爲 TO_ID、DATA_LEN、DATA 和 CRC_L/H 字段設置波特率。</td></tr>
<tr><td>0x08</td><td>PERIOD_HS_H</td><td>RW</td><td>PERIOD_HS 高 8 位，共 16 位（默認 0）</td></tr>
</table>

0x09	INT_FLAG	R	中斷標誌： bit0 BUS_IDLE 指示總線是否進入 IDLE 模式。 bit1 RX_PENDING 指示 RX 緩存是否有頁待讀。 寫 1 到 RX_CTRL[CLR_RX_PENDING] 清除當前頁待讀標誌。 bit2 RX_LOST 當一個包正確抵達且不被過濾，但卻因爲沒有更多頁用做下一次接收而被丟棄，此標誌置位。 寫 1 到 RX_CTRL[CLR_RX_LOST] 清此標誌。 bit3 RX_ERROR 當一個不被過濾的包停止位錯誤、超時或校驗錯誤，此標誌置位。 寫 1 到 RX_CTRL[CLR_RX_ERROR] 清此標誌。 bit4 TX_BUF_CLEAN 指示是否所有 TX 緩存頁都未標誌爲待發送。 bit5 TX_CD 當檢測到有更高優先級節點時推後發送並置此標誌。 寫 1 到 TX_CTRL[CLR_TX_CD] 清此標誌。 此位用作調試使用。 bit6 TX_ERROR 檢測到衝突後，當總線再次空閒超過設定時間硬件會自動重發，但如果連續重發 3 次都發生衝突，則取消發送，並置位此標記。 寫 1 到 TX_CTRL[CLR_TX_ERROR] 清此標誌。
0x0A	INT_MASK	RW	中斷允許 當 INT_FLAG & INT_MASK != 0 時 INT_N 輸出低，否則輸出高阻（默認 0x00）
0x0B	RX	R	讀 RX 緩存頁數據，地址自動增加 共有 8 個 RX 緩存頁，每一頁 256 字節。 當硬件端成功接收到不被過濾的包：如果下一頁未標誌爲待讀（可用作下一次接收），將當前頁標誌爲待讀並切換到下一頁；否則丟棄該包並置位 RX_LOST. RX_PENDING 位指示用戶端當前頁待讀，寫 1 到 CLR_RX_PENDING 清除當前頁的待讀標誌並切換到下一頁。寫 1 到 RST_RX 清除所有頁的待讀標誌，並復位接收邏輯。

0x0C	TX	W	寫 TX 緩存頁數據，地址自動增加 共有 2 個 TX 緩存頁，每一頁 256 字節。 當用戶寫完數據，需要等待 TX_BUF_CLEAN 置位，然後才可以通過 START_TX 置位當前頁的待發送標誌，並自動切換到下一頁（否則什麼都不會發生）。 當頁的待發送標誌被置上，硬件將會啓動發送，當讀完發送所需數據時，硬件端清頁的待發送標誌並切換到下一頁。
0x0D	RX_CTRL	W	RX 控制: bit0 RST_RX_POINTER 寫 1 歸零當前 RX 緩存頁的讀指針 bit1 CLR_RX_PENDING (自動包含 bit0) bit2 CLR_RX_LOST bit3 CLR_RX_ERROR bit4 RST_RX (自動包含 bit0, 2, 3)
0x0E	TX_CTRL	W	TX 控制: bit0 RST_TX_POINTER 寫 1 歸零當前 TX 緩存頁的寫指針 bit1 START_TX (自動包含 bit0) bit2 SET_TX_BUF_CLEAN_MASK Set INT_MASK[TX_BUF_CLEAN] bit 你可以在置位 START_TX 時同時置此位，當頁的待發送標誌被清除時，便會來 TX_BUF_CLEAN 中斷。 bit3 CLR_TX_CD bit4 CLR_TX_ERROR
0x0F	RX_ADDR	RW	讀寫當前 RX 緩存頁的讀指針
0x10	RX_PAGE_FLAG	R	（僅 NO_DROP 置位時使用） 0 代表當前 RX 緩存頁中的數據包正確； 非 0 表示數據包錯誤，其值指示最後接收到的字節地址，包含 CRC 字段。

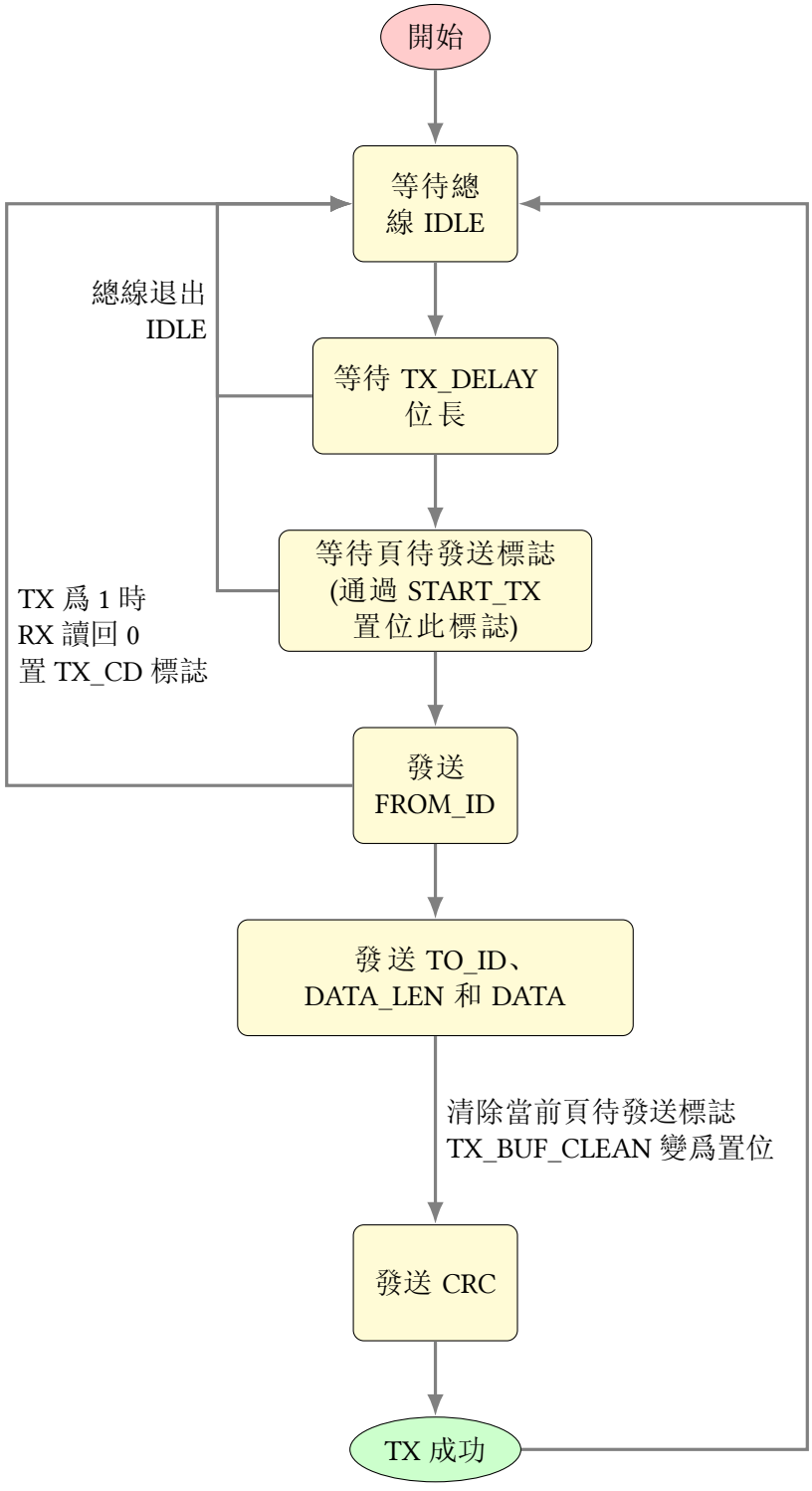
5 流程圖

5.1 RX



如果當前包接收不夠兩個字節，或者將會被丟棄則不會設置 RX_ERROR 標誌。

5.2 TX



6 設備接口

SPI 和 I2C 頻率建議低於 $sysclock \div 10$.

除了 RX 和 TX, 其餘寄存器通常只讀寫 1 字節。

6.1 SPI

讀寫：

```
start (NSS = 0)
Write reg address with bit7:  0: read, 1: write
Read or write arbitrary length of data
stop (NSS = 1)
```

6.2 I2C

寫地址：0xc0
讀地址：0xc1

寫：

```
start
write the write address
write 1 byte reg address
write arbitrary length of data
stop
```

讀：

```
start
write the write address
write 1 byte reg address
restart (or stop + start)
write the read address
read arbitrary length of data, ACK all bytes except last byte
stop
```

7 操作示例

7.1 初始化

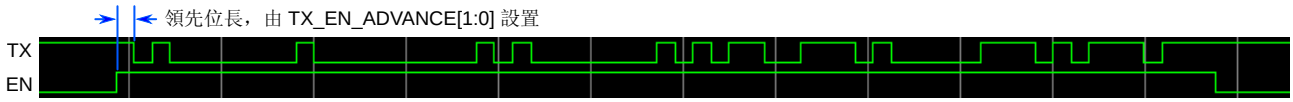
```
// select system clock, enable OUTPUT
CD485_write(REG_SETTING, F27M | OUTPUT_EN);

// set SELF_ID
CD485_write(REG_SELF_ID, 0xcd);

// set bondrate, PERIOD_XX_H default 0
CD485_write(REG_PERIOD_LS_L, 35); // 750000 bps
CD485_write(REG_PERIOD_HS_L, 2);  // 9 Mbps

// clean RX buffer
CD485_write(REG_RX_CTRL, RST_RX);
```

```
// enable interrupt
CD485_write(REG_INT_MASK, RX_ERROR | RX_LOST | RX_PENDING);
```

7.1.1 兼容模式和傳統模式

PERIOD_LS 和 PERIOD_HS 設置相同爲兼容模式。

進一步置位 NO_ARBITRATE 進入傳統模式：



7.2 TX

```
header_buf[0] = 0xcd; // FROM_ID
header_buf[1] = 0x02; // TO_ID
header_buf[2] = 12;   // DATA_LEN

CD485_write_chunk(REG_TX, header_buf, 3);             // write HEADER
CD485_write_chunk(REG_TX, data_buf, header_buf[2]);   // write DATA

while (CD485_read(REG_INT_FLAG) & TX_BUF_CLEAN == 0); // make sure TX_BUF_CLEAN is set
// sent packet, and enable TX_BUF_CLEAN interrupt
CD485_write(REG_TX_CTRL, SET_TX_BUF_CLEAN_MASK | START_TX);

// write next packet
// send next packet when the TX_BUF_CLEAN interrupt occur

// if no further packet need send, disable TX_BUF_CLEAN interrupt:
CD485_write(REG_INT_MASK, CD485_read(REG_INT_MASK) & ~TX_BUF_CLEAN);
```

7.3 RX

```
// when RX_PENDING interrupt occur:

CD485_read_chunk(REG_RX, header_buf, 3);           // read HEADER
CD485_read_chunk(REG_RX, data_buf, header_buf[2]); // read DATA

CD485_write(REG_RX_CTRL, CLR_RX_PENDING);          // release page
```

8 版權說明

CD485（又名 CD-BUS）是一個相當開放的協議，硬件實現也相對簡單，任何人都可以在自己的產品中免費使用此協議或其變種：譬如汽車生產商可以在汽車內部使用 CD485 協議、機器臂公司可以在機械臂內部使用此協議，同時也可以提供對外的，兼容 CD485 協議的接口；但如果是銷售獨立的控制器芯片和模組，則需要支付版權費用。無論何種情形，你需要在自己的產品網頁中保留原始的協議名稱和版權信息。

聯絡：duke@dukelec.com