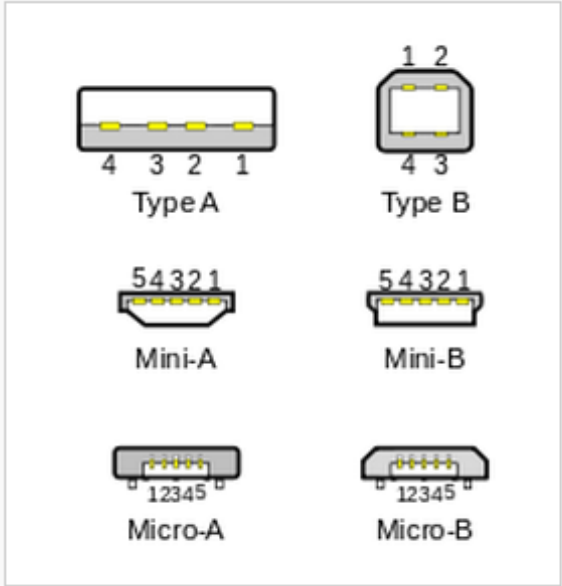
usb

標準 USB 2.0 介面

USB是一種序列資料傳輸界面(serial interface)，設計目標為隨插即用（Plug and Play，簡稱PnP）。亦包含實體層，資料鏈結層，網路層，傳送層...應用層等等。

實體層



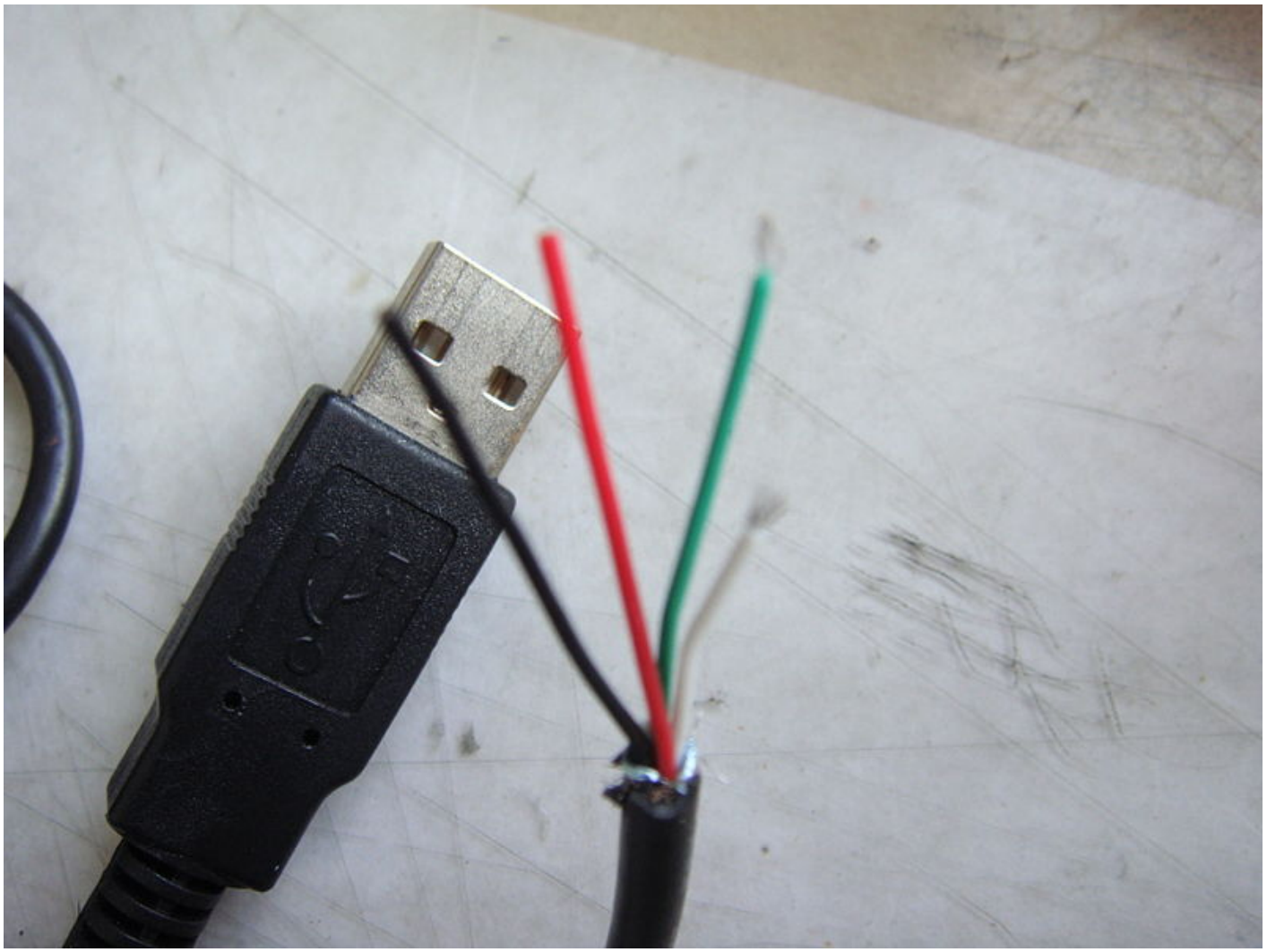
Standard, Mini, and Micro USB plugs (not to scale). The white areas in these drawings represent hollow spaces. As the plugs are shown here, the USB logo (with optional letter A or B) is on the top of the overmold in all cases. Pin numbering looking into receptacles is mirrored from plugs, such that pin 1 on plug connects to pin 1 on the receptacle.[43]

USB 1.x/2.0 standard pinout

Pin	Name	Cable color	Description
1	V_{BUS}	Red (or Orange)	+5 V
2	D−	White (or Gold)	Data −
3	D+	Green	Data +
4	GND	Black (or Blue)	Ground

USB 1.x/2.0 Mini/Micro pinout

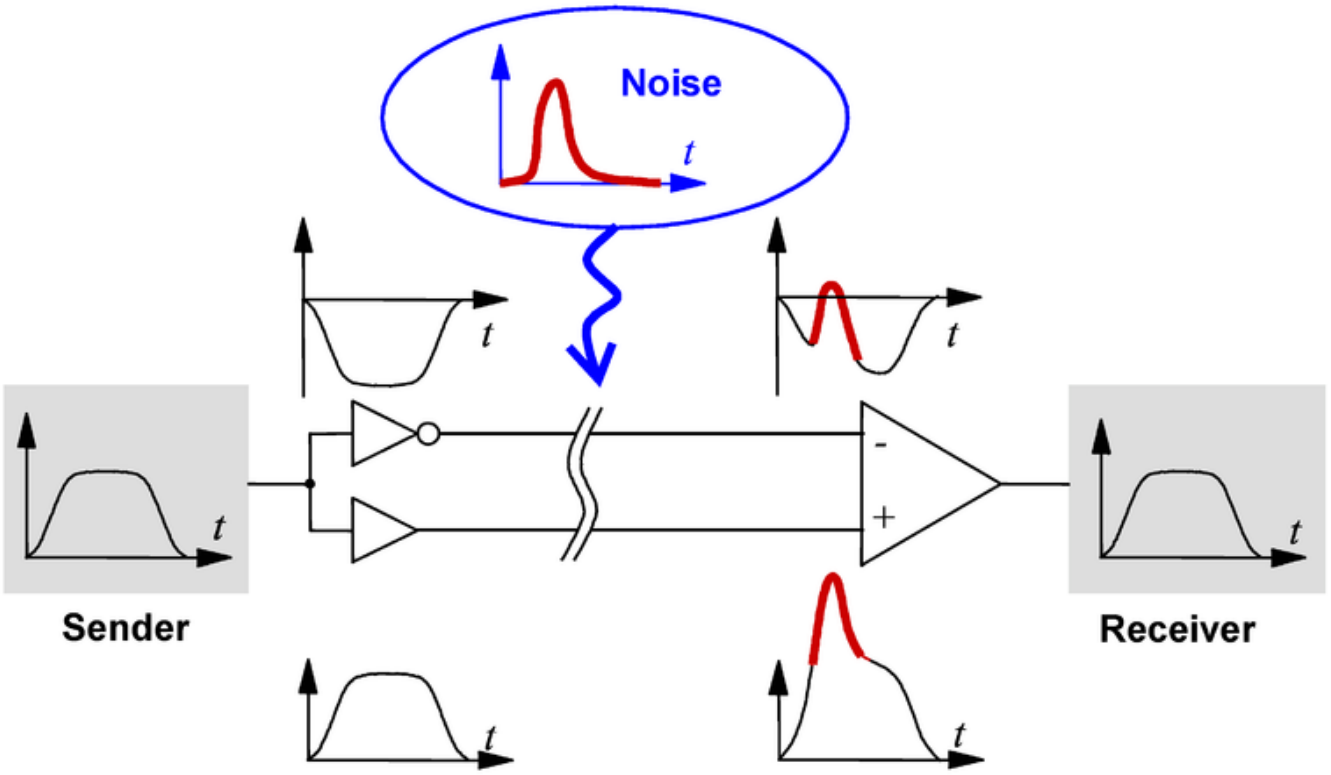
Pin	Name	Cable color	Description
1	V_{BUS}	Red	+5 V
2	D−	White	Data −
3	D+	Green	Data +
4	ID	N/A	Permits distinction of a host connection from device connection: • host: connected to the signal ground • device: not connected
5	GND	Black	Signal ground



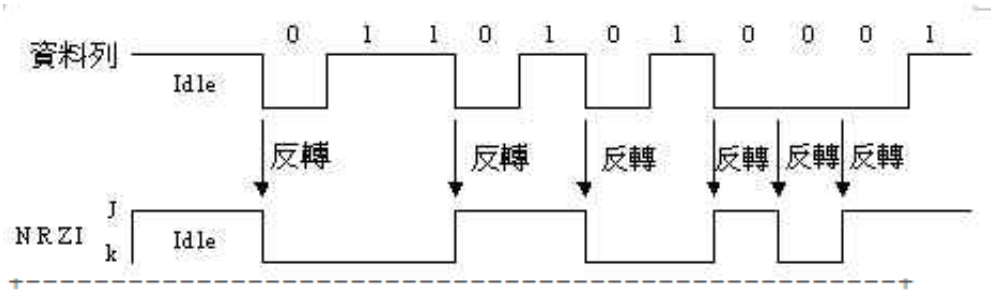
USB 訊號使用標記為D+ 和D- 的雙絞線(twisted pair)傳輸，以抵消長導線的電磁干擾。

它們各自使用半雙工的差動訊號(Differential signal), 並利用NRZI(non-return-to-zero Inverted)的編碼方式來傳送。

- 差動訊號：訊號傳輸在兩根不同的線上，這兩個信號的振幅相等，相位相反。



- NRZI Code: NRZI(Non Return to Zero Invert，不歸零就反向)的編碼方式，無需同步的時脈信號也能產生同步的資料存取。NRZI的編碼規則是，當資料位元為 “1” 時不轉換，為 “0” 時再作轉換。



- J state:Idle state(閒置)
- K state:Resume state(恢復)

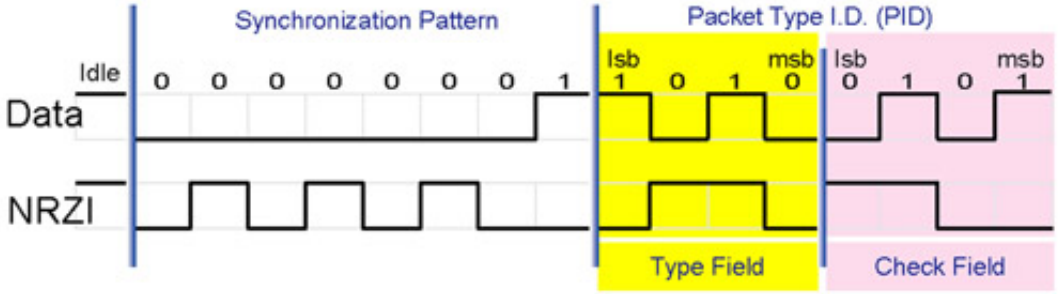
在NRZI編碼下，邏輯0 會造成轉換，所以接受者在接受數據的同時，根據接收到的翻轉信號不斷調整同步頻率，保證數據傳輸正確。

但是，這樣還是會有一個問題，就是雖然接受者可以主動和發送者的頻率匹配，但是兩者之間總會有誤差。

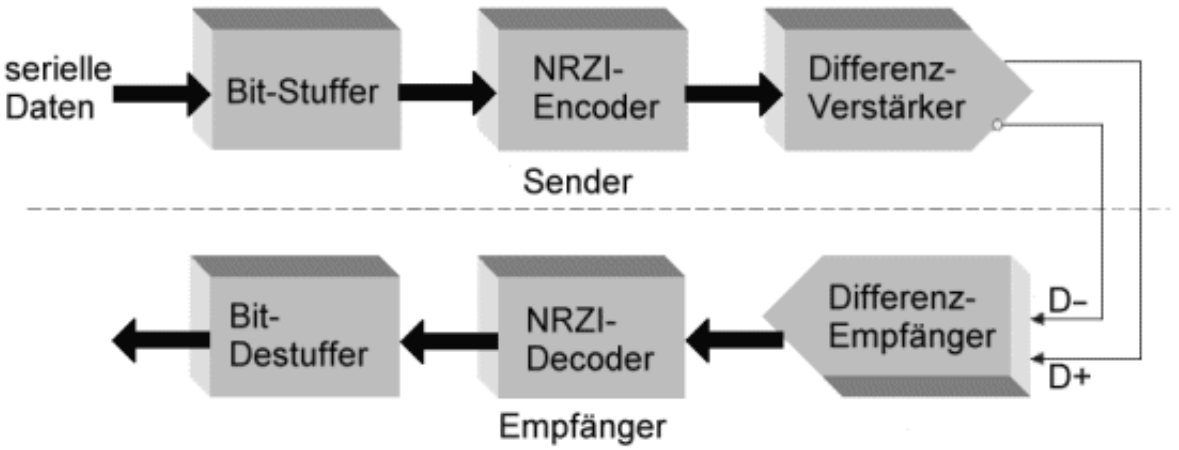
假如數據信號是1000 個邏輯1，資料就會造成長時間無法轉換，在這種情況下，即使接受者的頻率和發送者相差千分之一，就會造成把數據採樣成1001個或者999個1了。

USB 對這個問題的解決辦法，就是強制插0，也就是bit-stuffing，如果要傳輸的數據中有7 個連續的1，發送前就會在第6 個1 後面強制插入一個0，讓發送的信號強制出現翻轉，從而強制接受者進行頻率調整。

- Bit-stuffing : 如下圖所示，若是原始的串列資料中含有連續6個 “1” 位元的話，就需執行位元填塞的工作。
- 相對的，接收端在作資料接收之前，就必需先執行NRZI解碼，然後再作位元反填塞(unBit-Stuffing)。



- 資料傳輸流程

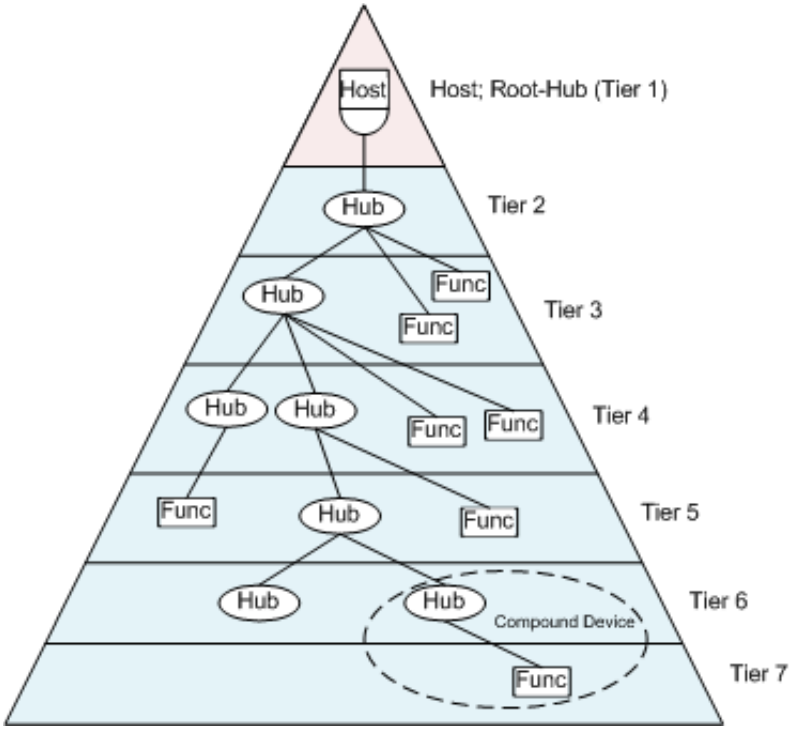


網路層

USB網路是由階梯型星型拓撲(Tiered Star Network)所實作的，有一個 Master(Host) 和多個 Slaves(Devices) 。

USB Host 提供一個接口，如果要連接更多週邊，需要使用Hub。

- 一層：提供給 Host (Bus Master)
- 六層：提供給 Hubs 或 Devices
- 考慮訊號傳遞衰減和延遲，網路架構最多七層，所以USB網路最多可支援127個裝置節點。



USB Device 又可以分為：

- Hub : 用來連接更多裝置
- Function : 提供特殊功能給系統，例如人機裝置, 影像裝置, 儲存裝置等等。
- Compound Device： 本身擁有功能(Function)，也帶有一個Hub，有其他裝置連接時可以作為Host。
- Composite Device：提供超過一個功能的裝置。

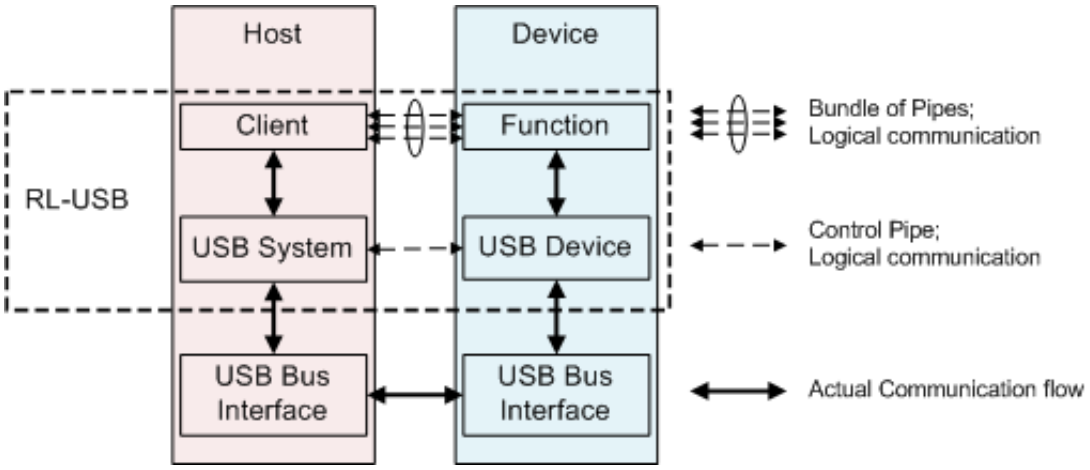
傳輸速率

- Low-Speed : 1.5 Mbps
- Full-Speed : 12 Mbps
- High-Speed : 480 Mbps

USB 通訊模型

從下面的圖可以看到，USB Host 和 Device 是資料傳輸的重點，分別在不同的兩層。 USB是一種Polled Bus，由Host啟動(Initiates)資料傳輸，Device做出回應。

- 此處的Polling(尋訪)是Host Controller來執行，當Bus需要被注意時，會發出中斷讓OS處理。



USB由幾種Protocol Layers所組成，資料藉由Pipes傳輸。

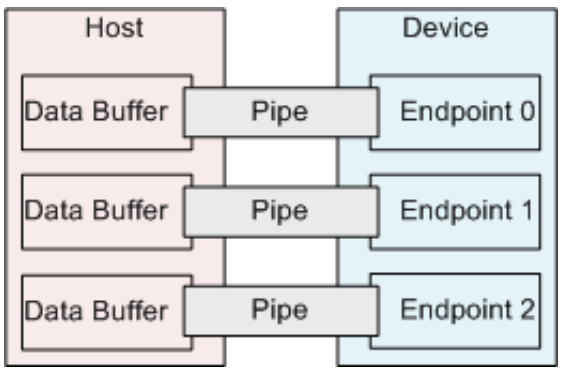
- Stream Pipes : Stream pipes可以被Host或者Device控制，資料的傳輸方向必須事先定義，定義為IN或或OUT。have no defined USB format）支援 Bulk Transfers, Isochronous Transfers, and Interrupt Transfers.
- Message Pipes : 為Host控制來進行資料傳輸設定，允許雙向通訊。只支援 Control Transfers。（have a defined USB format）

Endpoints

Endpoints可以被視為資料的來源或目標(Source or Sink)。一個裝置可以擁有至多16個OUT和16個IN的Endpoints

- OUT : 從Host指向Device的方向
- IN : 指向Host的方向

每個Endpoint只能有一個傳輸方向，只有Endpoint 0 是例外。Endpoint 0 可以有IN和OUT的雙向傳輸（透過Message Pipe），用來控制裝置（Device)。



傳輸型態

USB 定義了四種傳輸型態：

- 控制傳輸（Control Transfers）： 用於控制傳輸命令及狀態操作。像是設定裝置、取得裝置資訊、發送指令到裝置等。每個USB裝置都有一個Endpoint 0，USB Core就是使用他在裝置插入後進行設定。
- 中斷傳輸（Interrupt Transfers）： 與一般常見的中斷不同，需要 host 端先詢問(Polling)才會執行。

用一個固定速傳輸少量資料，像是USB鍵盤和滑鼠就是屬於這種方式。

- 批次傳輸（Bulk Transfers）：用於大量資料傳輸且需要確保資料無誤（如傳給印表機或隨身碟），沒有速度限制，若傳輸失敗就會重傳以確保正確。
- 同時傳輸（Isochronous Transfers)： 同樣用於大量資料傳輸，但不確保資料是否到達。例如例如USB視訊裝置，使用者會期望傳輸聲音或影像的速率是穩定的，若有幾張frame遺失，沒有通過CRC資料也不會重傳。

USB 資料連結

Transaction指USB資料的傳輸，大部分的傳輸包含了三種封包，封包是組成USB傳輸的最小單位。

一個 Transaction 通常由三個封包組成，但依傳輸型態而定，一個 Transaction 可能包含一個、兩個、三個封包。

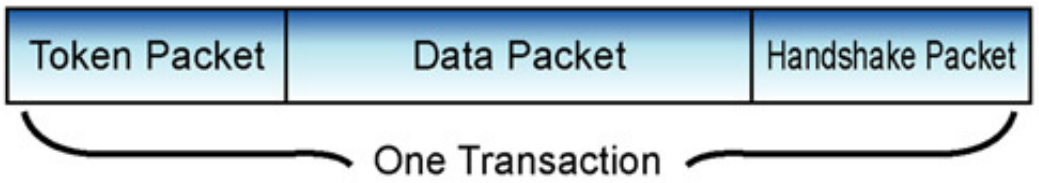


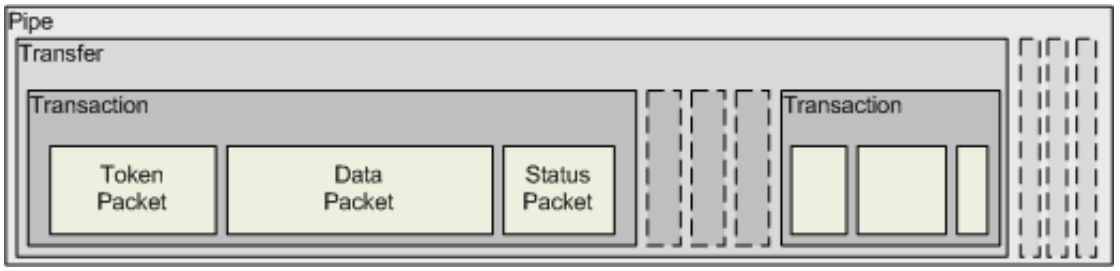
Table 8-1. PID Types

PID Type	PID Name	PID<3:0>*	Description
Token	OUT	0001B	Address + endpoint number in host-to-function transaction
	IN	1001B	Address + endpoint number in function-to-host transaction
	SOF	0101B	Start-of-Frame marker and frame number
	SETUP	1101B	Address + endpoint number in host-to-function transaction for SETUP to a control pipe
Data	DATA0	0011B	Data packet PID even
	DATA1	1011B	Data packet PID odd
	DATA2	0111B	Data packet PID high-speed, high bandwidth isochronous transaction in a microframe (see Section 5.9.2 for more information)
	MDATA	1111B	Data packet PID high-speed for split and high bandwidth isochronous transactions (see Sections 5.9.2, 11.20, and 11.21 for more information)
Handshake	ACK	0010B	Receiver accepts error-free data packet
	NAK	1010B	Receiving device cannot accept data or transmitting device cannot send data
	STALL	1110B	Endpoint is halted or a control pipe request is not supported
	NYET	0110B	No response yet from receiver (see Sections 8.5.1 and 11.17-11.21)
Special	PRE	1100B	(Token) Host-issued preamble. Enables downstream bus traffic to low-speed devices.
	ERR	1100B	(Handshake) Split Transaction Error Handshake (reuses PRE value)
	SPLIT	1000B	(Token) High-speed Split Transaction Token (see Section 8.4.2)
	PING	0100B	(Token) High-speed flow control probe for a bulk/control endpoint (see Section 8.5.1)
	Reserved	0000B	Reserved PID

*Note: PID bits are shown in MSb order. When sent on the USB, the rightmost bit (bit 0) will be sent first.

Transaction

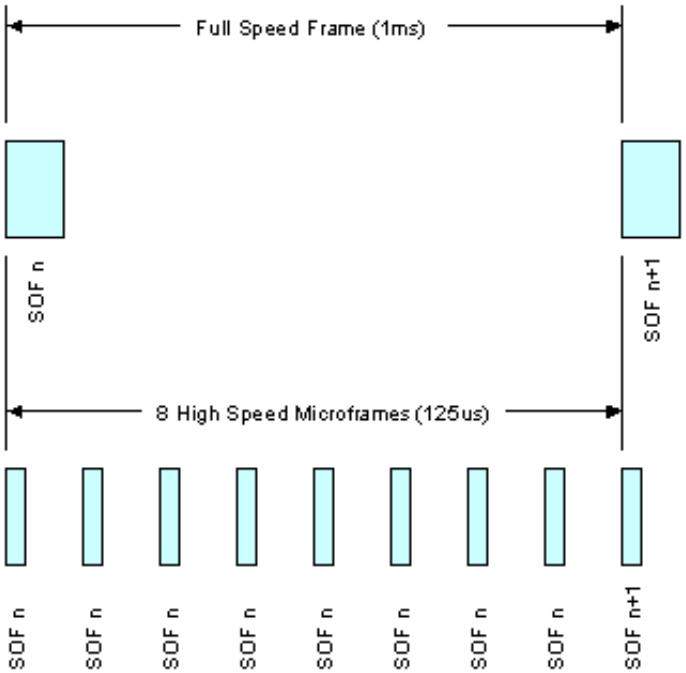
Transaction指USB資料的傳輸，大部分的傳輸包含了三種封包（Token packet, Data packet, Handshake 或稱 Status packet）。



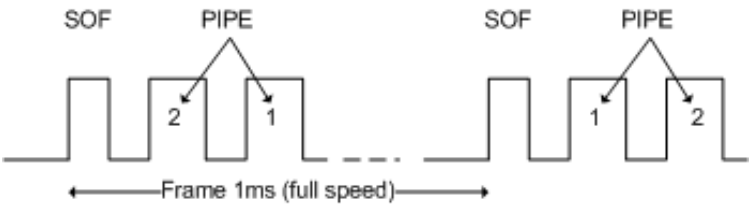
- Transaction可能是從Host傳向Device，或是從Device傳向Host。
- 傳送方向是由Token packet中指定。
- 一般來說，目標端利用Handshake (Status packet) 來判斷本次傳輸是否成功。

Frame

為了確保同步，USB把時間切割成固定長度的小區間。低速和全速的時候是以 1 ms 為單位，稱為一個 Frame；高速時再把一個Frame切成八等分，一個 0.125 ms 為單位，稱為microframes。

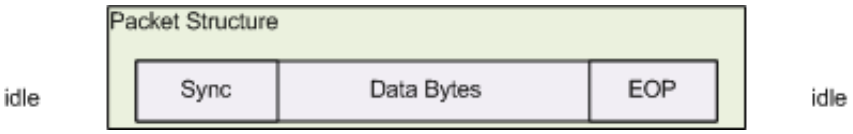


SOF (Start-of-Frame packet) 是一種特殊的封包，他在每一個frame開始時發送。



Packet

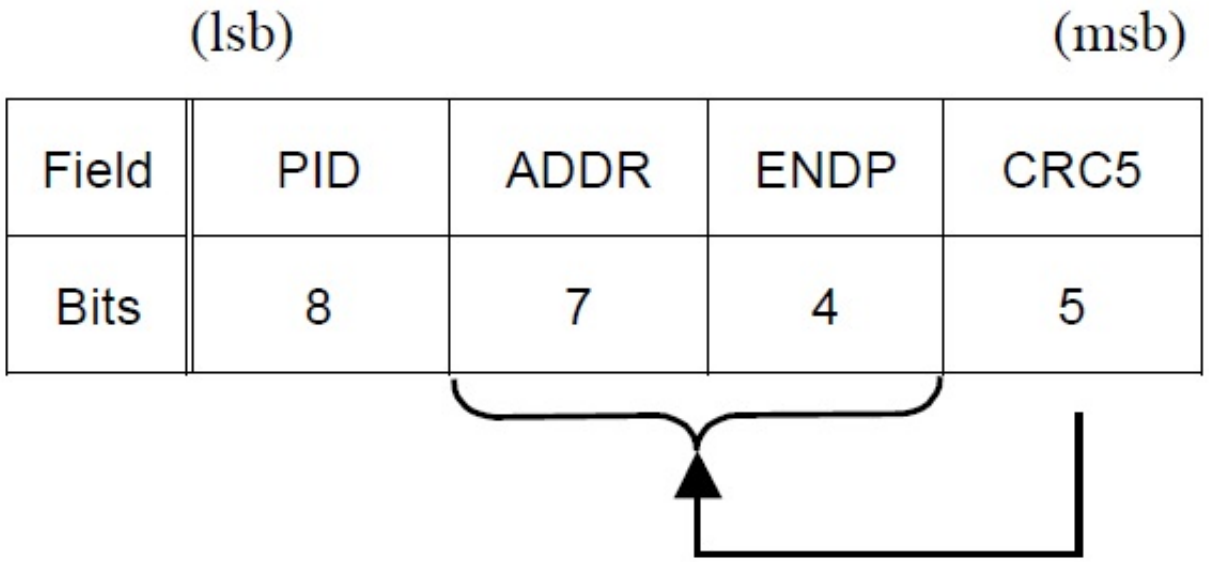
封包被視為每次傳輸的最小單位。



- 封包是以同步訊號（Synchronization Pattern）為開頭。
- 接著是資料區段，順序為Big-endian。（least significant bit first）。
- 最後已EOP訊號（End of Packet）結尾，完成一個封包的傳輸，Bus回到idle的狀態。

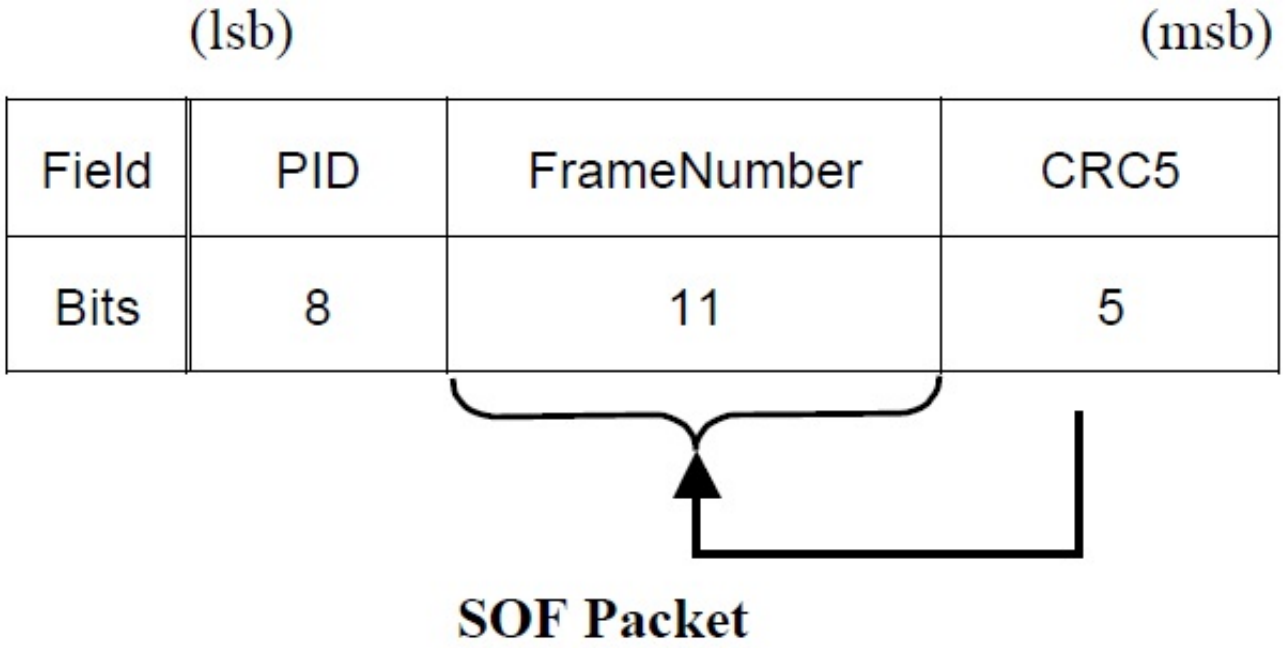
Token 封包

- 每個Transaction以Token封包做起始。
- Token封包定義裝置、Endpoint數量，傳輸的方向。
- 用PID來識別OUT、IN、SOF與SETUP處理。
- 對於OUT和SETUP處理 位址與端點欄用來選擇接收資料端點。
- 對於IN處理 位址與端點欄用來選擇傳送資料端點。



Token Format

SOF(Start-of-Frame)為一個特別的Token封包，包含目前的frame數量，主機每隔一段很小的間隔就會發出一次SOF封包，所有裝置以及hub都會收到SOF封包。對只需要時間資訊的裝置而言，只需要知道PID為SOF即可，剩餘的frame number以及CRC的資訊皆可忽略。



PID: Packet Identifier

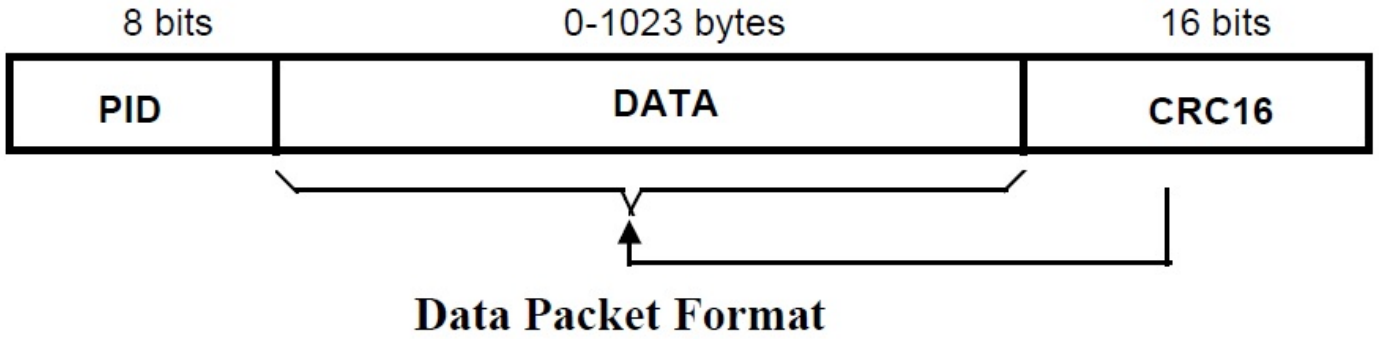
ADDR: Address

ENDP: Endpoint

CRC: Cyclic Redundancy Checks

Data 封包

Data封包包含處理此動作的資料。在low-speed裝置中，Data封包最大的資料量為8 bytes；在full-speed裝置中，Data封包最大的資料量為1023 bytes；在high-speed裝置中，Data封包最大的資料量為1024 bytes。其中 Data0 及 Data1 是兩個基本的資料封包，使用這些資料包以提供一個機制來確定將傳送端和接收端之間的資料切換同步(data toggle synchronization)。另外在 USB2.0 當中更增加了 Data2 及 MData 資料封包，用於執行高速的即時傳輸(Isochronous Transfers)。



Handshake 封包

除了即時型傳輸(Isochronous)外，所有的傳輸都保證資料的傳遞正確，如交握(Handshake)封包回應資料是否正確的被收到，若執行處理動作中發生錯誤，處理動作將重新執行。

Handshake封包由一個PID所組成，用來表示資料傳輸的狀態，部分Handshake封包說明如下。

- ACK表示資料封包沒有bit stuff或是CRC錯誤，也就是PID欄位以及Data欄位沒有出現錯誤。
- NAK表示裝置無法從主機接收資料或是無資料可以傳輸到主機。NAK也被當作流量控制的用途來使用，表示裝置暫時無資料傳送或無法接收資料。
- STALL表示裝置無法傳送或接收資料 需要主機介入來清除延遲狀況。

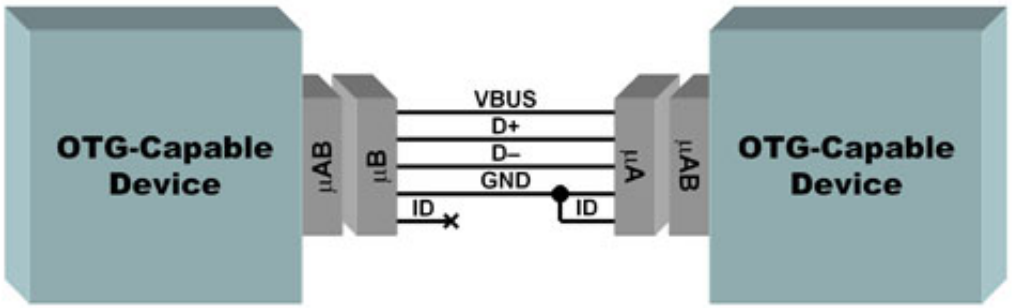
(lsb) (msb)

Field	PID
Bits	8

Handshake Packet

USB OTG (On-The-Go)

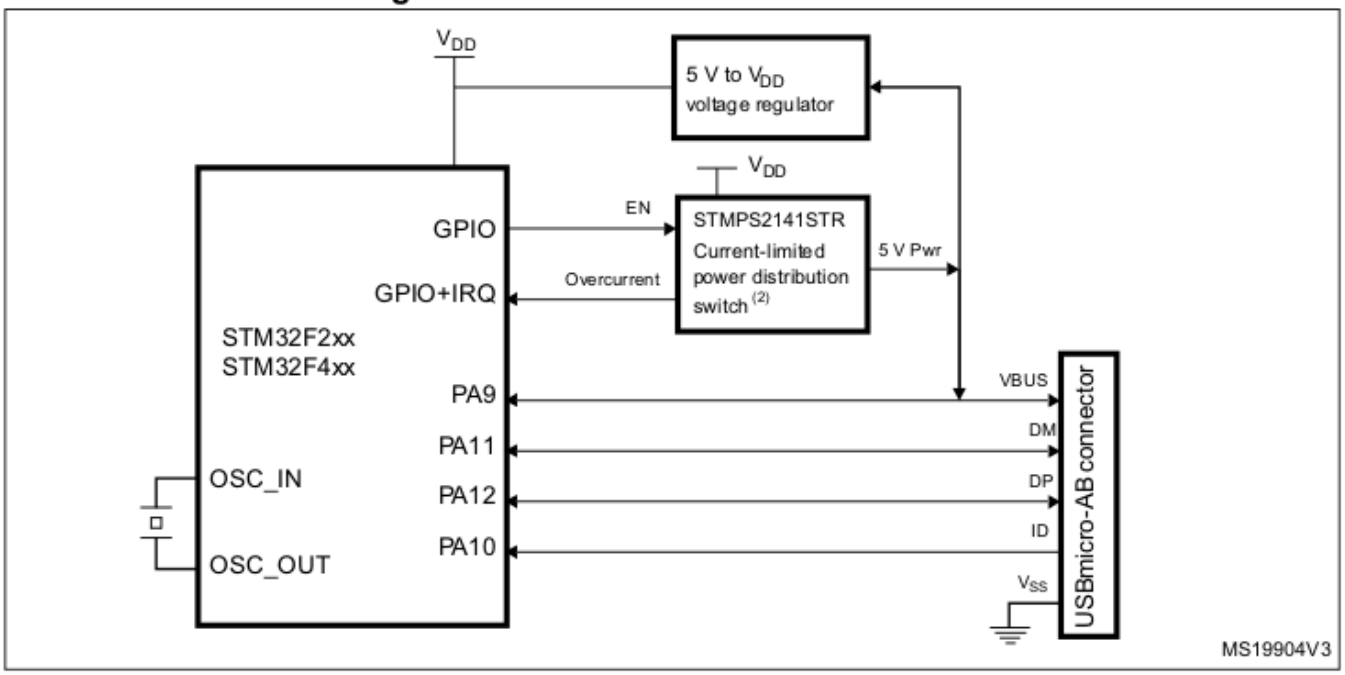
原先USB裝置的角色都是固定的(主機或週邊)。 USB OTG讓原本只能為週邊的裝置也能成為主機,例如讓鍵盤接上手機使用。



圖中type A connector作為host裝置，用來提供電源；type B則是目標裝置，接收電源。如此，可以避免兩個供應電源的裝置相接，引發傷害裝置的情形(嚴重甚至起火)。

- Dual Role Device (OTG-DRD)

Figure 386. OTG A-B device connection



1. External voltage regulator only needed when building a V_{BUS} powered device
2. STMPS2141STR needed only if the application has to support a V_{BUS} powered device. A basic power switch can be used if 5 V are available on the application board.

協定(Protocol)

OTG設備使用插頭中的ID引腳來區分A/B Device。

- ID接地被稱作為A-Device，為連接時候的USB Host，A-Device始終為總線提供電力
- ID懸空被稱作為B-Device，為連接時候的USB Device

設備的USB Host/USB Device角色可以通過HNP切換。

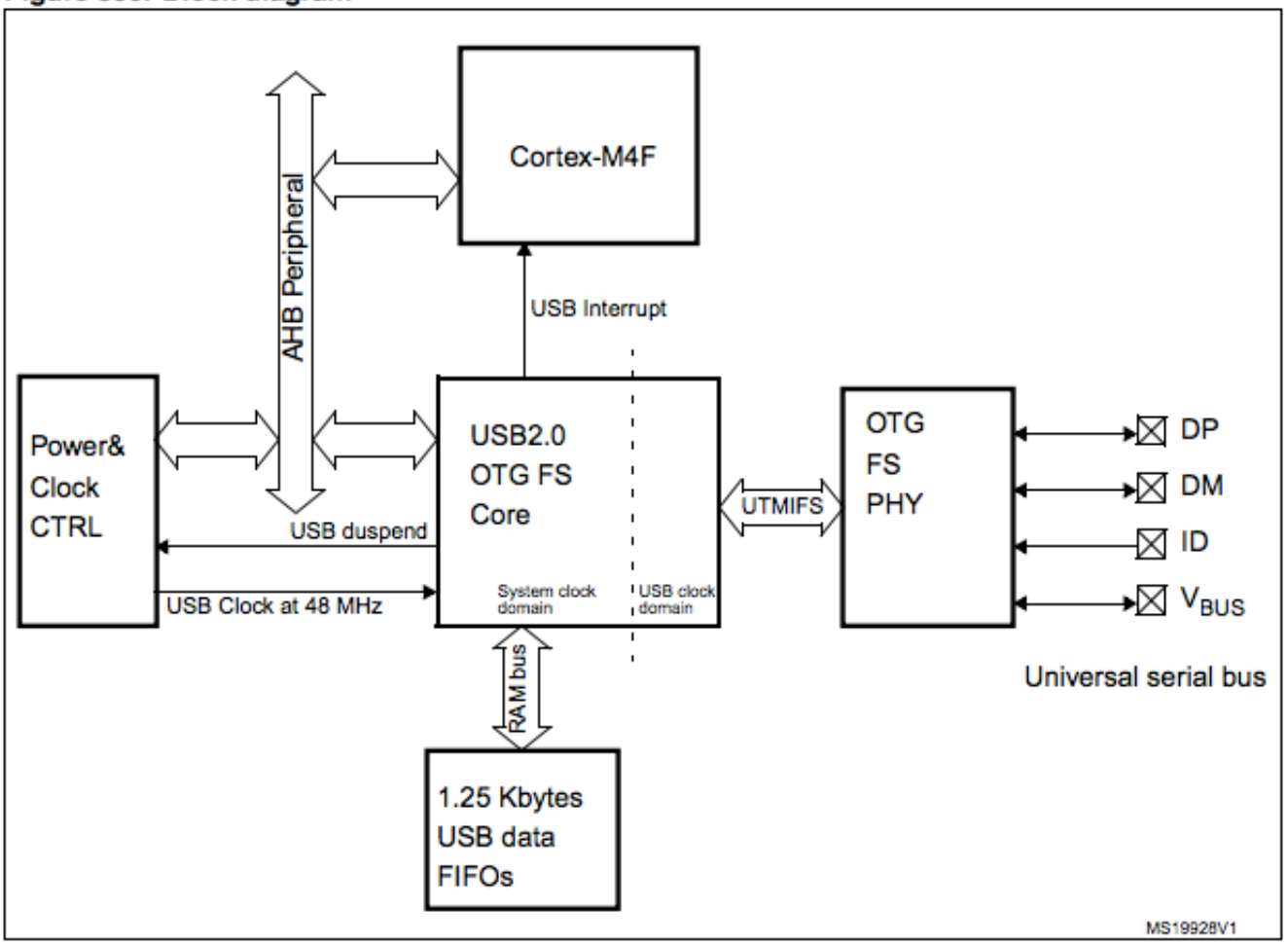
Session Request Protocol (SRP, 對話請求協議)

Host Negotiation Protocol (HNP, 主機通令協議)

Attach Detection Protocol (ADP)

Block Diagram

Figure 356. Block diagram



- ID pin 用來辨識OTG(on-the-go)裝置是主機還是週邊
- marco-A插頭的ID pin接地 vs marco-B插頭的ID pin懸空，用來區別主機和週邊（marco-A為主機，marco-B為週邊）
- DP/DM description – in reference manual 30.3.2
- AHB – Advanced High-performance Bus
- UTMI – USB 2.0 transceiver macrocell interface

USB host and device

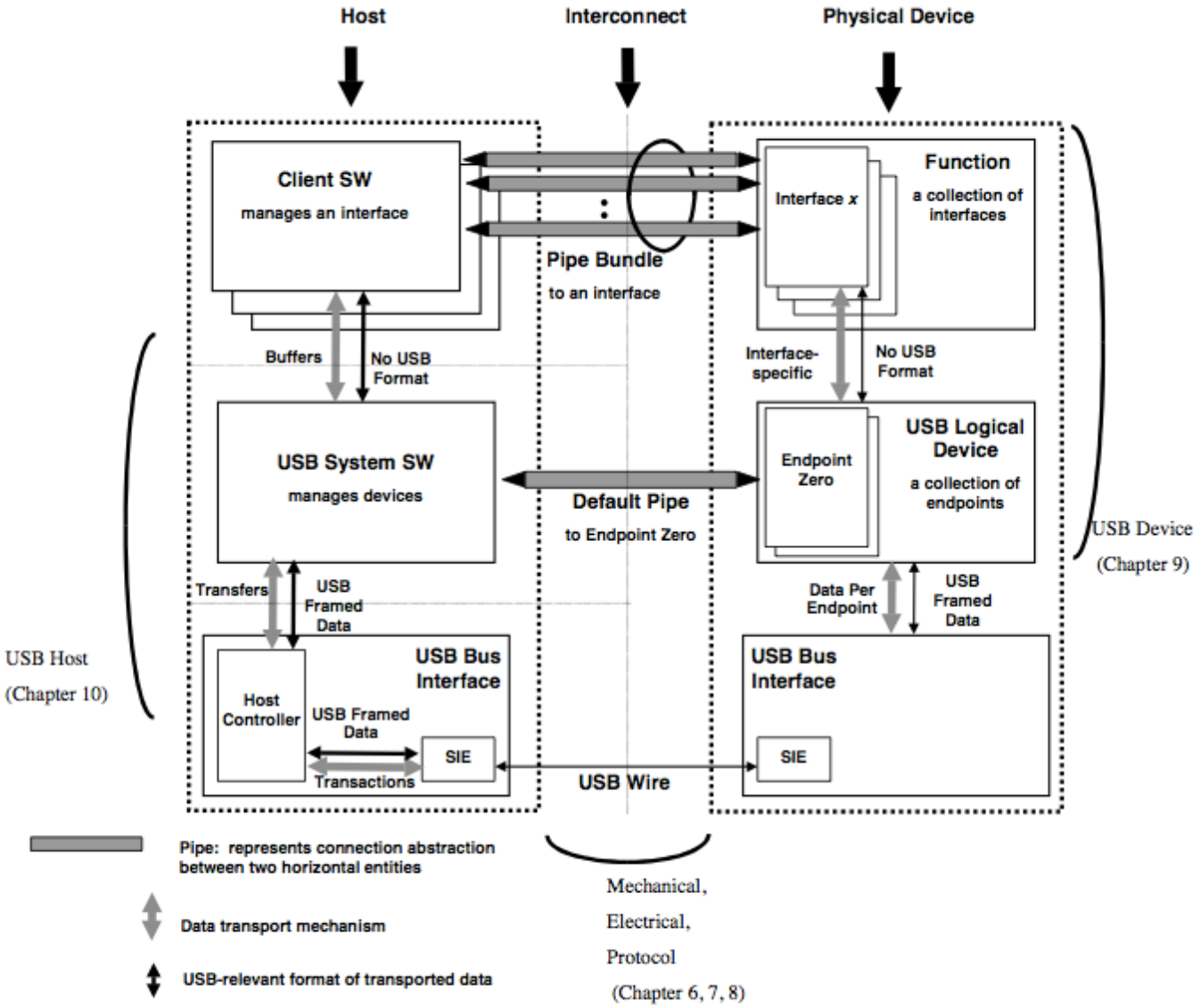


Figure 5-9. USB Host/Device Detailed View

Bus Timing/Electrical Characteristics

可以看參考資料USB 2.0 Specification的7.3.2章的內容

Table 7-7. DC Electrical Characteristics

Parameter	Symbol	Conditions	Min.	Max.	Units
Supply Voltage:					
High-power Port	V_{BUS}	Note 2, Section 7.2.1	4.75	5.25	V
Low-power Port	V_{BUS}	Note 2, Section 7.2.1	4.40	5.25	V
Supply Current:					
High-power Hub Port (out)	I_{CCPRT}	Section 7.2.1	500		mA
Low-power Hub Port (out)	I_{CCUPT}	Section 7.2.1	100		mA
High-power Function (in)	I_{CCHPF}	Section 7.2.1		500	mA
Low-power Function (in)	I_{CCLPF}	Section 7.2.1		100	mA
Unconfigured Function/Hub (in)	I_{CCINIT}	Section 7.2.1.4		100	mA
Suspended High-power Device	I_{CCSH}	Section 7.2.3; Note 15		2.5	mA
Suspended Low-power Device	I_{CCSL}	Section 7.2.3		500	µA
Input Levels for Low-/full-speed:					
High (driven)	V_{IH}	Note 4, Section 7.1.4	2.0		V
High (floating)	V_{IHZ}	Note 4, Section 7.1.4	2.7	3.6	V
Low	V_{IL}	Note 4, Section 7.1.4		0.8	V
Differential Input Sensitivity	V_{DI}	|(D+)-(D-)|; Figure 7-19; Note 4	0.2		V
Differential Common Mode Range	V_{CM}	Includes V_{DI} range; Figure 7-19; Note 4	0.8	2.5	V
Input Levels for High-speed:					
High-speed squelch detection threshold (differential signal amplitude)	V_{HSSQ}	Section 7.1.7.2 (specification refers to differential signal amplitude)	100	150	mV

Table 7-7. DC Electrical Characteristics (Continued)

Parameter	Symbol	Conditions	Min.	Max.	Units
High speed disconnect detection threshold (differential signal amplitude)	V_{HSDSC}	Section 7.1.7.2 (specification refers to differential signal amplitude)	525	625	mV
High-speed differential input signaling levels		Section 7.1.7.2 Specified by eye pattern templates			
High-speed data signaling common mode voltage range (guideline for receiver)	V_{HSCM}	Section 7.1.4.2	-50	500	mV
Output Levels for Low-/full-speed:					
Low	V_{OL}	Note 4, 5, Section 7.1.1	0.0	0.3	V
High (Driven)	V_{OH}	Note 4, 6, Section 7.1.1	2.8	3.6	V
SE1	V_{OSE1}	Section 7.1.1	0.8		V
Output Signal Crossover Voltage	V_{CRS}	Measured as in Figure 7-8; Note 10	1.3	2.0	V
Output Levels for High-speed:					
High-speed idle level	V_{HSOI}	Section 7.1.7.2	-10.0	10.0	mV
High-speed data signaling high	V_{HSOH}	Section 7.1.7.2	360	440	mV
High-speed data signaling low	V_{HSOL}	Section 7.1.7.2	-10.0	10.0	mV
Chirp J level (differential voltage)	V_{CHIRPJ}	Section 7.1.7.2	700	1100	mV
Chirp K level (differential voltage)	V_{CHIRPK}	Section 7.1.7.2	-900	-500	mV
Decoupling Capacitance:					
Downstream Facing Port Bypass Capacitance (per hub)	C_{HPB}	VBUS to GND, Section 7.2.4.1	120		µF
Upstream Facing Port Bypass Capacitance	C_{RPB}	VBUS to GND; Note 9, Section 7.2.4.1	1.0	10.0	µF
Input Capacitance for Low-/full-speed:					
Downstream Facing Port	C_{IND}	Note 2; Section 7.1.6.1		150	pF
Upstream Facing Port (w/o cable)	C_{INUB}	Note 3; Section 7.1.6.1		100	pF
Transceiver edge rate control capacitance	C_{EDGE}	Section 7.1.6.1		75	pF

Table 7-7. DC Electrical Characteristics (Continued)

Parameter	Symbol	Conditions	Min.	Max.	Units
Input Impedance for High-speed:					
TDR spec for high-speed termination		Section 7.1.6.2			
Terminations:					
Bus Pull-up Resistor on Upstream Facing Port	RPU	1.5 kΩ ±5% Section 7.1.5	1.425	1.575	kΩ
Bus Pull-down Resistor on Downstream Facing Port	RPD	15 kΩ ±5% Section 7.1.5	14.25	15.75	kΩ
Input impedance exclusive of pullup/pulldown (for low-/full-speed)	ZINP	Section 7.1.6	300		kΩ
Termination voltage for upstream facing port pullup (RPU)	VTERM	Section 7.1.5	3.0	3.6	V
Terminations in High-speed:					
Termination voltage in high-speed	VHSTERM	Section 7.1.6.2	-10	10	mV

- Table 7-8. High-speed Source Electrical Characteristics
- Table 7-9. Full-speed Source Electrical Characteristics
- Table 7-10. Low-speed Source Electrical Characteristics
- Table 7-11. Hub/Repeater Electrical Characteristics

以上Table可以參閱USB 2.0 Specification的7.3.2章

USB Descriptors

- Device Descriptors : 提供裝置基本資訊，例如：USB 版本、最大的 packet 大小、廠商與型號資訊。每個 USB 裝置只能 device descriptors 只能有一個。
- Configuration Descriptors : 指出 USB 裝置的供電方式、interface 數量。每個 USB 裝置可以擁有不只一個 Configure Descriptor，可視為 interface descriptor 之 header。
- Interface Descriptors : 提供介面資訊，可視為 Endpint descriptor 之 header。在 configuration descriptor 中會指定該 configuration 所擁有的 interface 數量，因此 interface descriptor 可以擁有一個以上。
- Endpoint Descriptors : 描述 endpoint 之行為與頻寬。
- String Descriptors : 提供人類可讀的資訊。

USB Device Class

USB 依照裝置的不同，定義許多不同的 class code ，用來聯絡 host 端與 device 端該做出什麼適當的反應，以下為幾個常見分類：

HID Class

Human Interface Device class，HID 設備屬於人機交互操作的設備，用於操作電腦，如 USB 滑鼠，USB 鍵盤，USB 觸控版，USB 軌跡球等等設備。HID 設備不一定非要是這些人機交互設備，只要符合HID設備級定義規範要求的都可以認為是 HID 設備。

使用 HID 設備的一個好處就是，作業系統自帶了 HID 類的驅動程序，而用戶無需去開發很麻煩的驅動程序，只要直接使用 API 調用即可完成通信。所以很多簡單的 USB 設備，喜歡舉例成 HID 設備，這樣就可以不用安裝驅動而直接使用。

CDC Class

Communications Device Class，CDC 類是 USB 通信設備類的簡稱。CDC 類是 USB 組織定義的一類專門給各種通信設備（電信通信設備和中速網絡通信設備）使用的 USB 子類。大部分的作業系統都帶有支持 CDC 類的設備驅動程序，可以自動識別 CDC 類的設備，這樣不僅免去了寫專用設備驅動的負擔，同時簡化了設備驅動的安裝。

MSC Class

Mass Storage device Class，大容量存儲裝置類別。一種電腦和行動裝置之間的傳輸協議，它允許一個通用串行總線（USB）裝置來訪問主機的計算裝置，使兩者之間進行文件傳輸。

MSC 支持目前大多數的主流操作系統，許多舊版本的操作系統經過版本升級或者係統補丁也能實現對 MSC 的支持。MSC 的通用性和操作簡單使他成為行動裝置上最常見的文件系統，USB MSC 並不需要任何特定的文件系統, 它提供了一個簡單的界面來讀寫接口用於訪問任何硬碟驅動。

Demo 程式 - USB Host ^^^^^^^^^^

```
git clone https://github.com/kvzhao/USB-Demo
```

如果make有錯誤，請先建一個build資料夾

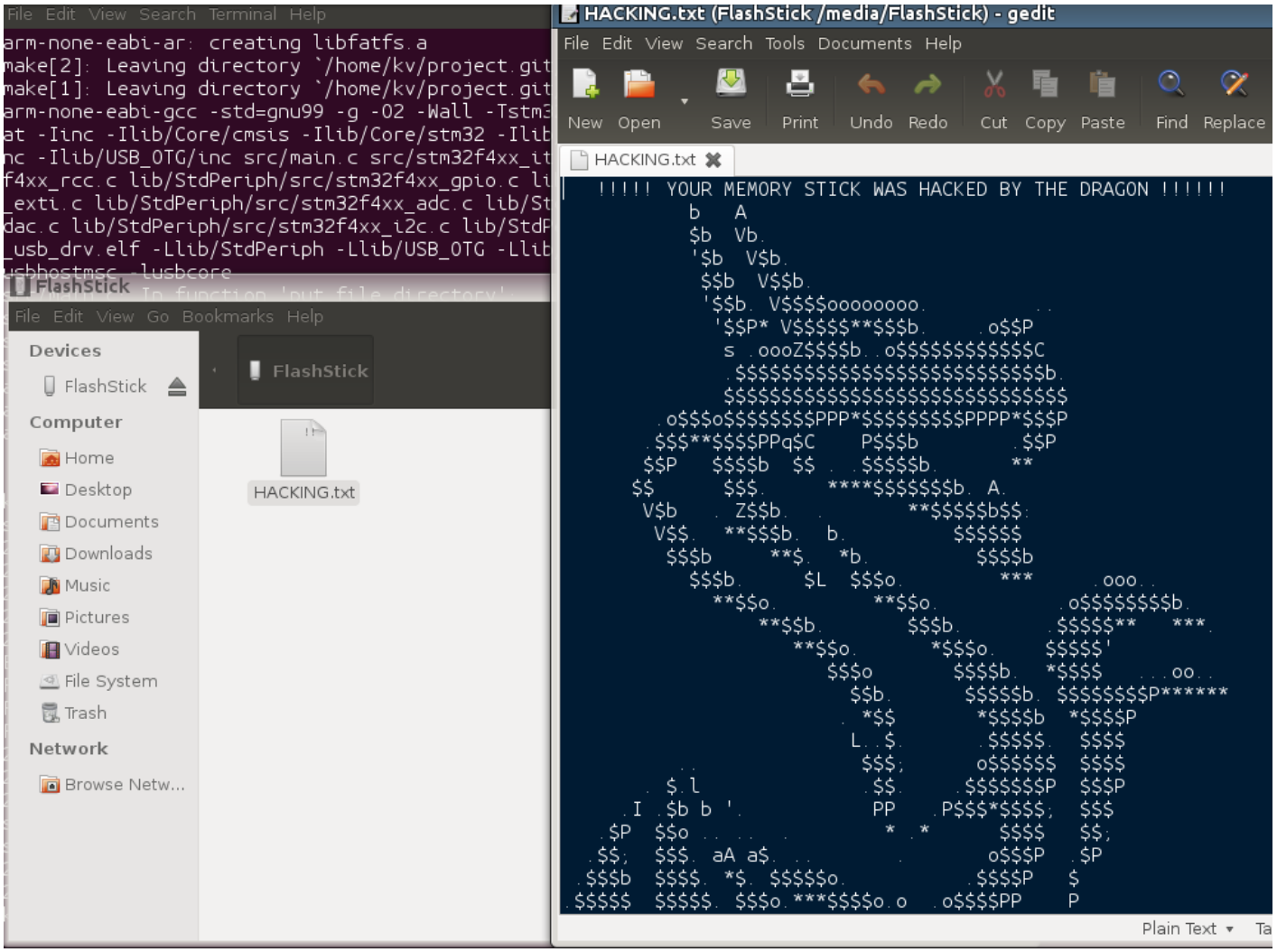
```
mkdir build && make
```

目的

把STM32F4作為Host，當插入隨身碟後在裡面寫入一份文件。

步驟與現象

- 紅燈亮起：偵測到隨身碟
- 綠燈亮起：file system mount完成
- 橘燈亮起：已打開隨身碟中的目錄
- 藍燈亮起：完成寫入簽名，並且關閉file
- 當隨身碟拔出後，所有LED燈會熄滅，結束實驗，並且在隨身碟中留下文件。

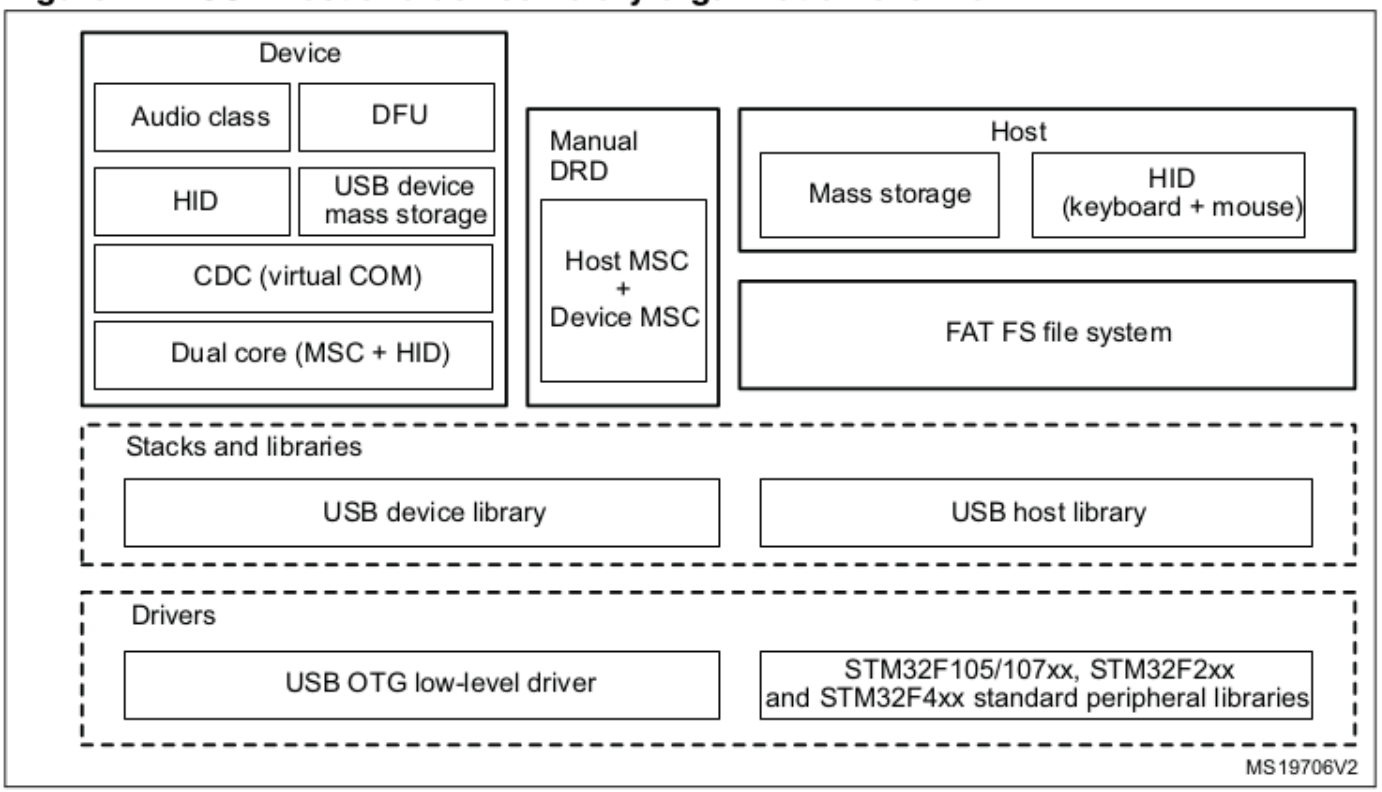


- ASCII Art
- Matlab image generator

程式說明

- STM32提供大量的函式庫來操作USB，包跨底層的driver到提供使用者應用的API。

Figure 1. USB host and device library organization overview

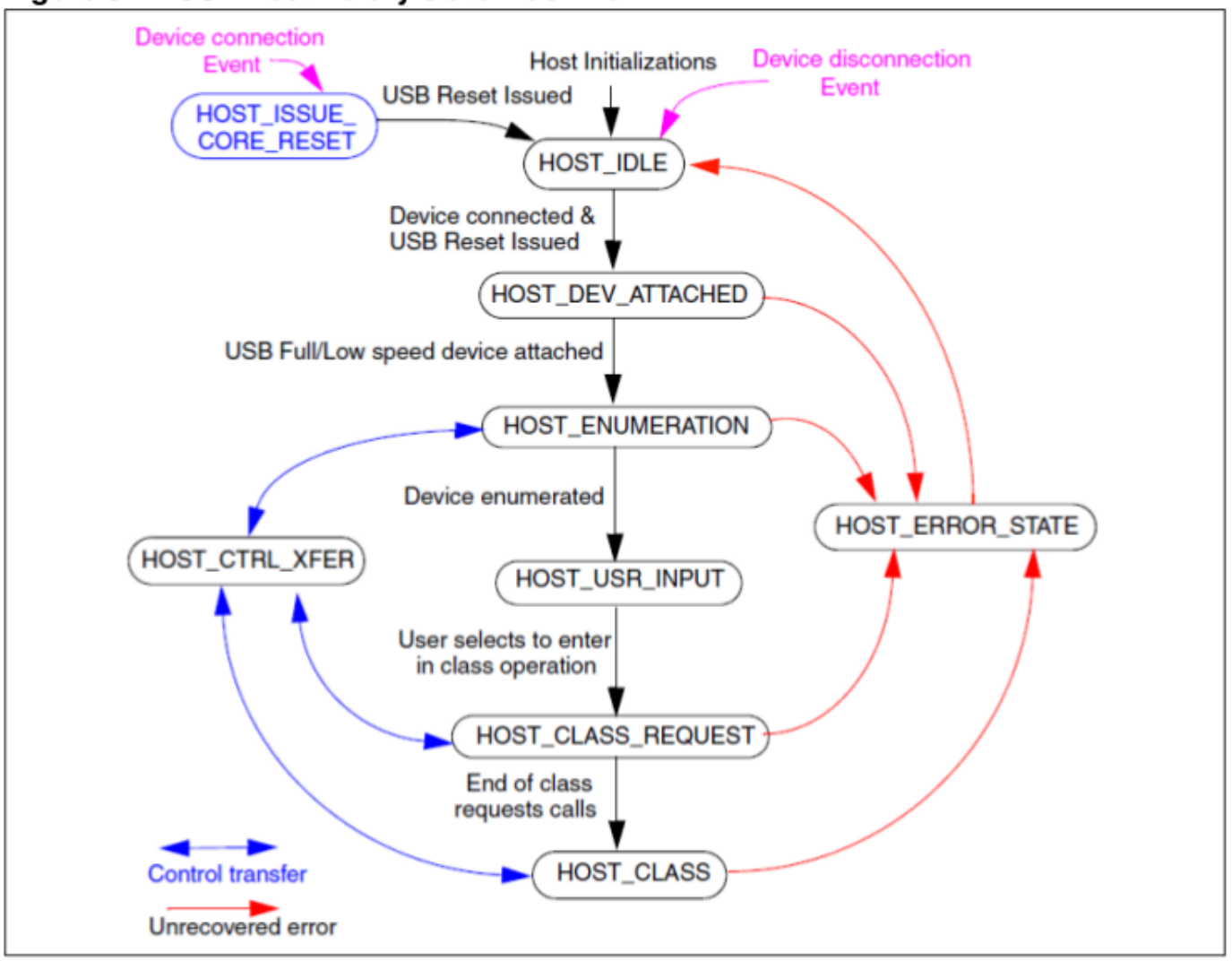


The USB host and device libraries are built around the common STM32 USB OTG low level driver and the USB device and host libraries.

本實驗主要將裝置STM32F4設定為Host，而隨身碟為Device。

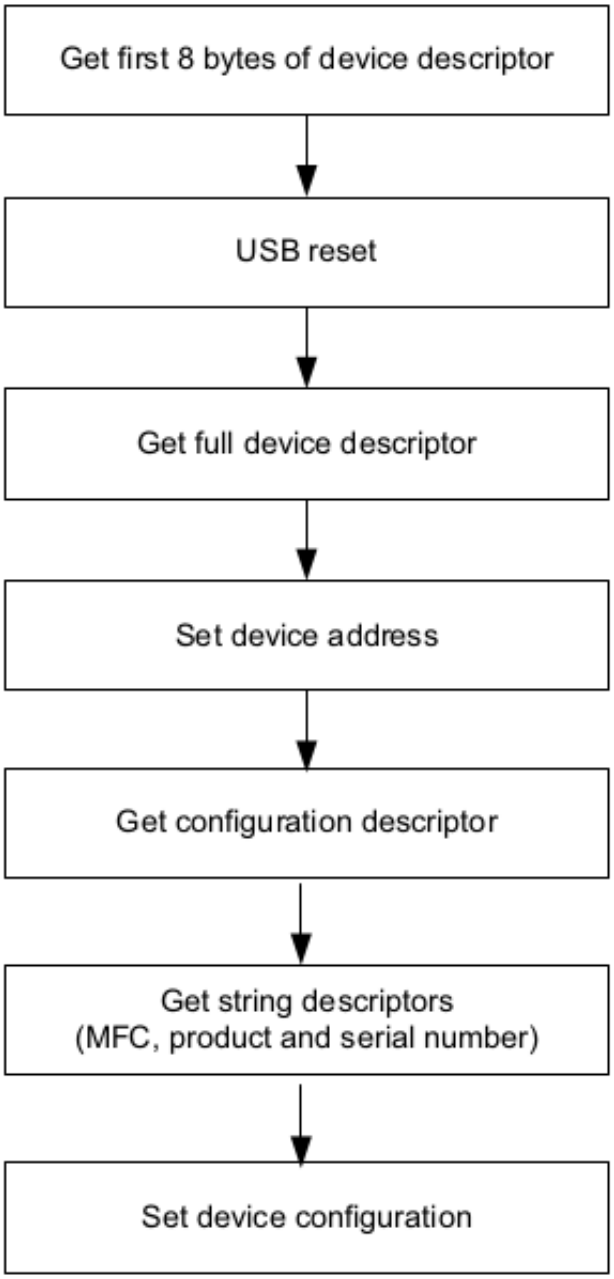
- Host state machine

Figure 34. USB host library state machine



- HOST_IDLE : 初始化之後，USB Core會不斷輪詢（Polling）是否有USB裝置被偵測。此時VBus為了省電會關起來。
- HOST_ISSUE_CORE_RESET : 當有裝置連接時，USB為了建立BUS，會進入Reset狀態。
- HOST_DEV_ATTACHED : 當裝置連接完成，Core也偵測到了以後，會進入裝置列舉（Enumeration）的狀態。
- HOST_ENUMERATION : USB Core 會開始列舉所有 USB Device。當列舉完成後，會選擇裝置的初始狀態。（Default device configuration is selected.）
- HOST_USR_INPUT : 一個中間狀態，等待使用者有進一部的指令，若收到後開始 USB Class 的操作。
- HOST_CLASS_REQUEST : 當進入這個狀態，就由驅動程式（Class Driver）接管，處理一些初始的控制請求，像是Get_Report_Request（HID）等等，當這些 Request 正確完成後，就進入HOST_CLASS狀態。
- HOST_CLASS : 此時狀態機會呼叫與應用相關的操作（Class-related Operation），完成指定的任務。
- HOST_CTRL_XFER : 當需要傳送控制訊息（Control Request）都會進入這個狀態。
- HOST_ERROR_STATE : 無法修復的錯誤發生時將進入這個狀態，之後 Host Library將會從新初始化。

- Enumeration



Q & A

Q: 如何推導出 USB 3種速度（low speed：1.5 Mb/s、full speed 12 Mb/s、high speed 480 Mb/s）的理論值？
A: 這是原本定義的各種傳輸速度主頻率。可以利用pull-up resistor來控制。

Q: packet ID 實際上只使用到前四個 bits，為什麼後四個 bits（check field）是採用前四個 bits 的補數？
A: USB 採用的是 NRZI 編碼，此作法方便執行，更可避免出現 bit-stuffing。

> Q: smart phone 接上 PC 後封包傳遞順序？
> A: 主機端的 USB 集線器監視著它的每個端口的信號線的電壓，當USB設備插入主機時，信號線的電壓會發生變化，此時主機知道有新設備插入了。
> 當主機檢測到設備的插入後會首選重啟這個設備，接著主機發出 Get_Port_Status 請求來驗證設備是否已經重啟，設備重啟後主機通過檢測根信號線的電壓狀態判斷設備的速度。
> 主機發送第一次 Get_Descriptor（wValue 字段的高字節為 0x01，表示設備描述符）請求取得設備描述符，設備描述符提供了設備的多種信息，包括：設備通訊終端 0 的最大封包的大小、設備支持的配置號以及有關這個設備的其它信息、主機通過對這些信息的分析以確定接下來的通信動作。
> device descriptor 裡規定了設備一個或多個 configuration descriptor，主機再次或多次發出 Get_Descriptor（wValue 字段的高字節為 0x02，表示 configuration）指令來讀取這些 configuration descriptor，第一次只讀出 configuration descriptor 的前 9 個字節，其中包含了 configuration descriptor 和它的所有從屬描述符（interface descriptor、endpoint descriptor）的總長度，然後主機根據這個長度讀出設備的所有 configuration descriptor（當然包括其所有從屬描述符）。
> 讀取完 configuration descriptor 後，若之間讀取的 configuration descriptor 中指定了相關 string descriptor（用來描述廠商、產品和設備序列號信息的）的索引，主機將發出若干次 Get_Descriptor（wValue 字段的高字節為 0x03，表示string descriptor）命令來獲得這些 string descriptor，此時主機將會彈出窗口，展示發現新設備的信息，產商、產品描述、型號等。在主機已經從它的 descriptor 中知道了能夠知道的所有信息後，便開始為這個設備安裝驅動程序。加載了 USB 設備驅動以後，主機發送 Set_Configuration 命令請求為該設備選擇一個合適的配置。

> Q: CRC的長度如何得知？
> A: token packet using 5bit CRC
>> data packet using 16bit CRC

參考資料

行動電化學蝕刻

Basic Concept of USB, Keil

USB Made Simple

STM32Fxx USB on-the-go Host and device library

USB Audio Class

USB OTG 中文

- MSC
- USB 3.1 Specification
- STM32F407xx Reference Manual
- USB On The Go
- USB in a NutShell
- USB 2.0 Specification