

國立台北大學資訊工程學系專題報告

GigE Vision Protocol 分析與實作 GigE Vision Protocol Analysis and Implementation

專題組員：莊文立

專題編號：PRJ-NTPUCSIE-101-005

執行期間：101 年 7 月 至 102 年 4 月

1. 摘要

此研究旨在分析 GigE Vision Protocol 分析，包含流程分析與封包解析。對於 Device 端和 Host 端個別作徹底的解析，以便對兩端進程式軟體，以 C 語言進行實作。並對應已有的實體硬體產品做整合測試，最終將軟體整合進入業者的硬體 SoC 平台。

關鍵字：GigE Vision、UDP、Stream

2. 簡介

GigE Vision Protocol 屬於 OSI 模型中最頂端應用層的協定，基於 Ethernet、IP、UDP 發展出一套傳輸影像的協定，同時規範了 host 端（通常是 PC 上的應用程式）和 device 端（即相機本身）有關搜尋、控制相機、串流的行為。相較於其他類似協定，如 CameraLink、IEEE1394，GigE Vision 具有高速、低耗、易於整合和二次開發…等優勢。

為了克服 UDP 的不可靠性，GigE Vision 額外制定了 GigE Vision Control Protocol (GVCP) 和 GigE Vision Stream Protocol (GVSP) 以處理封包可能在傳輸過程遺失的問題。

GigE Vision 的標準目前已經被廣泛的使用在工業攝影機、相機的產品上。而此標準的設計是能夠讓，技術開發者能夠快速且簡單找出 GigE Vision 兼容的產品，而且還有隨插即用的優點。

3. 專題進行方式

本計畫為鑫視覺公司的產學計畫，由我和李炆龍兩人合作，計畫分成三階段。

第一期：兩人各自實作 Control Channel 和 Stream Channel

第二期：整合二人專案。

第三期：整合入業者的 SoC 平台。

3.1 分析

GigE Vision 是由 AIA 所製訂的一個基於 GigE 的工業相機高速傳輸的標準。主要目的是提供一個開放的平台以供相機和電腦間可以利用 Ethernet



相互傳輸大量及時的影像和控制訊號。開放相容是它主的目的，在效能上它提供不同的選擇，廠商可以自行依照產品性能和定位來選擇他們要提供的效能，包括 CPU 的使用率、幀率(fps)、及時性(real-time)和訊號同步性(precisely timed synchronization requirements of multi-element applications)等。

GigE Vision 在 transport layer 使用 UDP，而不使用一般的 TCP/IP。UDP 和 TCP 的差異在於 TCP 用 Hand-shaking 和重送的方式，以保證資料沒有在傳送時遺失。相對於 TCP，UDP 是較為簡單的傳送方式，它不保證不會遺失，但 GigE Vision 將資料遺失的工作，交由位於更上層的網路層中的 GVCP、GVSP 協議來處理。

此標準包含了硬體的規格和標準的相機軟體控制通訊協議(包含 Device Discovery、GVCP、GVSP)。

- Device Discovery: 定義 GigE camera 如何取得 IP 地址和如何在網路上被識別。
- GVCP(GigE Vision Control Protocol): 定義如何描述資料流通道(stream channel)和控制, 以及 GigE camera 的設定。
- GVSP(GigE Vision Stream Protocol): 定義影像在傳送時如何封裝以及 GigE camera 如何將影像和其他資訊傳送給遠端電腦。

3.1.1 Device Discovery

Device Discovery 包含了一些事件需求來對可控制的裝置經由網路介面取得連線，也用來獲得有效的 IP 位置

讓軟體能夠對已知的可控裝置進行存取。

3.1.2 GVCP

GigE Vision control protocol(GVCP)使用 UDP 傳輸協定，讓軟體能夠設定裝置和讓 GVSP 傳送與接收，以及讓裝置通知軟體有事件發生。

GVCP 必須提供只能一隻程式(主要程式)控制裝置的限制，也就是寫入裝置。也可以有許多程式能夠監控裝置(讀取裝置)，但都須經過主要程式才可執行。在 GVCP 下程式就像是長官，裝置就像是士兵。命令的要求都是由程式下達的。

為了處理多個連接與不同類型的訊息，GigE Vision 提供了 channel 的概念。一個 channel 是一個虛擬的連線，在 GigE Vision 實體之間傳遞訊息。分為三種 channels:

1. Control channel
2. Stream channel
3. Message channel

不同的通道可使用同一網路介面，但指派的 port 是不同的。

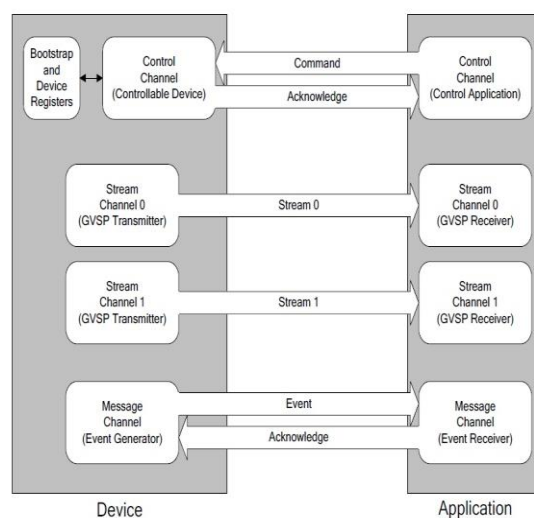


圖 2.1.2 基本 channel 範例

Control channel 是讓所使用的應用程式與裝置進行溝通。GVCP 定義了兩種 Control channel:

1. Primary control channel: 被建立在主要程式上主要用來寫入裝置暫存器，只能被一個程式使用。
2. Secondary control channel: 被建立在次要程式上，只可以用來讀取裝置暫存器。

一個 control channel 被要求發送程式的 command 訊息與接收對應的裝置 acknowledge 訊息。裝置的 bootstrap registers 允許主要的程式建立 message 和 stream channel。如果程式不是第一個取得 control channel 的話將無法使用 message 和 stream channel。

Control Channel Privileges，GVCP 定義了四個等級的特權：

1. Exclusive access: 一個程式有唯一的存取控制(讀寫)裝置，其他程式則無法進行讀寫此裝置，提供了最低限度的安全。
2. Control access: 它可以讓一個程式進行讀寫裝置，允許其他的程式進行讀取。
3. Control access with switchover enabled: 它可以讓一個程式進行讀寫裝置，允許其他的程式進行讀取。這個模式允許另一個程式控制裝置，它具有一個正確的憑據。
4. Monitor access: 一個程式不具任何特權，只能讀取裝置。通常用於幫助測試。

Stream channel 能夠讓資料從一個 GVSP 的傳送端傳送到 GVSP 的接收端，也就是說使用 Stream channel 傳送須受到 GigE Vision streaming protocol(GVSP)的規範。這個 channel 可以傳送任何被定義成此規範的 payload type。但是資料的傳送只能是單向的。

GVCP 封包的標頭分為兩種：

1. command header
2. acknowledge header

command header 包含了認證 key、旗標、command、訊息長度以及訊息編號。認證 key 用來辨識訊息是否為一個 GVCP 訊息。

acknowledge header 則包含了狀態碼、acknowledge、訊息長度、以及訊息的回應編號。每個 command 訊息的內容會依據 command 的種類而不同。

command 的種類有：DISCOVERY、FORCEIP、PACKETRESEND、READREG、WRITEREG、READMEM 以及 WRITEMEM。

Acknowledge 訊息的前兩 bytes 是用來識別，command 訊息是否成功執行所回傳的狀態碼。

3.1.3 GVSP

GigE Vision streaming protocol(GVSP)也使用 UD，允許資料源傳送給接收端。但 UDP 是一種不可靠的傳輸協定，所以 GVSP 提供了一些機制，得在傳輸的過程中可以保證資料的正確與確保最小流量控制。

資料傳輸時將資料切割成若干個

小元素，但必須提供這個資料區塊的訊息好讓接收端進行解碼。也就是說，第一個資料區塊要提供這個資料的 Data Payload 或者而外的資訊幫助解碼。Payload packets 必須根據 packet_id(packet_id32) 順序的發送。

在 Standard Transmission mode 模式下，Data Leader、Data Payload 和 Data Trailer 的結尾是被分在不同的封包裡，可以清楚地分割連串的兩個資料區塊。

Standard Transmission mode 有三種不同的 packet type:

1. Data Leader Packet: 為一個新的資料的開頭 packet。
2. Data Payload Packet: 一個或多個包含實際資料進行串流傳輸的 packet。
3. Data Trailer Packet: 做為資料結尾訊號的 packet

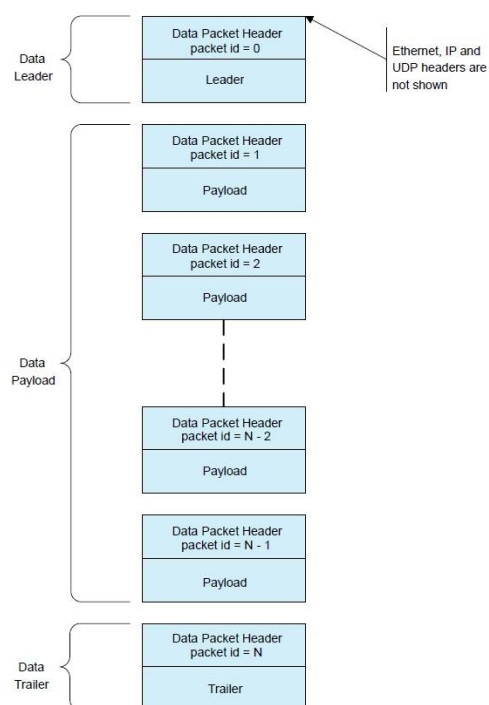


圖 2.1.3 Data Block-Standard

Transmission Mode

Data Block Packet Header 中所有的 packet 都使用相同一個基本 header。而 GVSP protocol header 不包含長度的資料，而是用 UDP 的長度訊息來決定資料的大小。

3.1.4 Bootstrap Registers

GigE Vision 定義了許多裝置相關訊息所儲存的固定位置。製造商也可自行定義相關的資料於裝置中，自定義的暫存器位置開始於地址 0xA000。通常 XML 對暫存器的位置進行描述，有可能是增量訊息、最小值、以及最大值。

3.2 實作

3.2.1 架構

所需設備：一台符合標準的相機、Giga Ethernet 介面卡、10 Base-T Ethernet 雙絞線、個人電腦。

最好將 device 和 host 直接對連，不要經過額外的節點，網路線長度至多 100 公尺，以降低封包遺失機率。

3.2.2 運作流程

首先啟動和 device 和 host 端程式。先由 host 對 255.255.255.0 廣播 discovery 訊息（稱為 Discovery Ping）。此時 Device 需對於每一個傳來的 Discovery Ping 做回應（稱為 Discovery Pong）給 host。一次完整

的 Ping、Pong 過程，host 即可確認發現相機。

完成 discovery 後，host 端透過 GVCP 向 device 要求權限 (privilege)，若成功取得特定權限，即可寫入特定暫存器，以啟動 GVSP 影像串流。

唯需注意，GigE Vision 只處理讀寫暫存器或記憶體和傳輸影像的部份，而不負責調整相機參數，如感光、色調、光圈大小…等等，這部份通常由另一個協定 GeniCam 處理，並不屬於 GigE Vision 的實作範圍。

3.2.3 GVCP

GVCP 負責所有關於控制 device 的命令 (command)，共分九大類：Discovery、Force_IP、Read_Reg、Write_Reg、Read_Mem、Write_Mem、Packet_Resend、Pending、Action，具有下列共通特性：

1. 所有命令之集合稱為 Control Channel。
2. 一律由 device 作為 server，固定監聽 3956 埠；host 作為 client 沒有一定埠號。
3. 為確保所有的封包不可被 IP 分割 (Fragmentation)，GVCP 的標頭 (header) 固定 8 位元組，標頭後的附加資料 (payload) 至多 540 位元組。
4. GVCP 的 payload 長度需要 4 位元的整數倍

5. 從 host 發出的每道命令 (稱為 REQ)，device 接收、分析、執行之後無論成功與否均需回應 (稱為 ACK)。每一個 REQ 都有唯一的 req_id，相對應的 ACK 也須有同樣的 ack_id。
6. 若 host 在 Pending Timeout 內未收到正確的 ack_id，則會重新傳送同樣的 REQ，直到收到正確的 ACK、或者達到重送次數上限為止。
7. 承上，Pending Timeout 預設 200ms，重送次數上限為 3 次。

以下分別解說所有命令。

1. DISCOVERY

host 端以廣播方式傳送至 255.255.255.0，限定在 3 秒之內收到回應。device 端無論什麼情況，收到 Discovery 命令必須立刻回應正確的 ACK。ACK 亦以廣播方式送回。ACK 需包含 device 的硬體資訊：IP、MAC、子網路、預設閘道、製造商、出廠序號…等等。

2. FORCE_IP

host 端一次寫入 IP、subnet 和 MAC address，需提供 device 的 MAC 以核對。目前只能指定靜態 IP，不支援 DHCP。device 端只有在滿足「host 端具有 FULL 權限」且「MAC 位址正確」且「網路設定正確」三條件時才執行命令，缺一不可。

3. READ_REG

host 端需提供暫存器位址，都是 32bit 的無號整數。可以一次讀取多個暫存器的值，列出多個位址即可。標準裡所規定的值，都已定在 reg_addr 裡。device 端依 REQ 提供的暫存器順序，依序填入暫存器的值；但若位址錯誤，則回應 0。每個值都轉換成 32bit 的無號整數

4. WRITE_REG

host 端暫存器位址、預計寫入的值都是 32bit 的無號整數，。可以一次更改多個暫存器的值，列出多個位址和值即可。標準裡所規定的值，都已定在 reg_addr 裡。device 端依 REQ 提供的暫存器順序，依序填入「寫入成功與否(1/0)」一定要回應具有 FULL 權限的 host。Heartbeat、CCP、Streaming 功能均透過寫入暫存器開啟或結束

5. READ_MEM

Host 端「記憶體位址」都是 32bit 的無號整數，「預計讀取的位元數」為 16bits。device 端只有在符合權限，且要求之位址有效才讀出記憶內容。每個值都轉換成 32bit 的無號整數通常是 GeniCam 讀取參數用(zipped XML file)

6. WRITE_REG

host 端記憶體位址是 32bit 的無號整數，之後再接預寫入的內容。device 端只有在符合權限，且要求之位址有效才寫入記憶體一定要回應具有 FULL

權限的 host。通常是 GeniCam 設定參數用

7. PACKETRESEND

host 端可指定重送 GVCP 或 GVSP 的封包。device 端為省流量，PACKETRESEND 不需 ACK，直接重送指定封包即可。

8. PENDING 目前沒有實作

9. ACTION 目前沒有實作

另一方面，device 的權限系統 (Control Channel Privilege) 分為四個等級

1. MONITOR

只能讀取不能寫入（故無法取得控制權限，也無法開啟串流）

2. FULL

最高權限，可讀可寫

3. READONLY

可讀；只可寫入「與串流相關之暫存器或記憶體」

4. READONLY_WRITE_CCP

可讀；只可寫入「與串流相關之暫存器或記憶體」和 CCP。

最後，device 操作暫存器，目前是以「讀檔 (camera.txt) 至程式主記憶體」的方式模擬。

3.2.4 GVSP

除了 MONITOR 權限之外，任何 host 只要接收埠號寫入 device 端的暫存器，就能開啟串流，稱之 Stream Channel。

標準裡規定 stream 永遠從 device 傳至 host，而且最多一次可有 512 個 Stream Channel（也就是 device 可以同時把影像傳給 512 個對象），編號 0 至 511。不過…現在只有實作一條 Stream Channel。

相關的暫存器如下：

- **SCP0**: Stream Channel Port (host 端指定的 port，亦是 Stream Channel 的開關)
- **SCPS0**: Stream Channel Packet Size (封包大小上限)
- **SCDA0**: Stream Channel Destination (host 端的 IP)
- **SCCFG0**: Stream Channel Configure File (載入設定檔，尚未實作)
- **SCPD0**: Stream Channel Packet Delay (封包傳送間隔，用於流量控制，尚未實作)
- **SCSP0**: Stream Channel Source Port (device 端串流埠號，尚未實作，交由作業系統決定)

在實作上，GVCP 和 GVSP 分別是個執行緒。一旦寫入 **SCP0**、**SCDA0**、**SCPS0** 便自動開始影像串流；反之，若要停止影像串流，將 **SCP0** 改為 0 即可。此外，標準裡規定了十數種影像格式，但目前只實作 **Rawdata**，且一律使用 **Extended Mode**。

Device 透過 `int imgFetch( byte * data )` 將影像存入 data，實作上以讀取 `pic/pic*` 檔案模擬。每一個影格

(single frame) 都是完整的圖片，在 Stream Channel 裡稱之為 **block**。由於是高畫質相機，所以每個 block 的資料都很大，以 Basler Ace 330gc 相機為例，Mono8 黑白影像 Mono8 約為 1.2 MBytes、彩色影像 YUV422Packed 約為 2.3 MBytes；但是，常見的 MTU 最高只有 9000Bytes，UDP 理論上的封包上限也只有 65508Bytes，因此自然無法整個 block 都放在同一個封包，必須分成十數個到上數千個小片段傳送。

雖然 GVSP 沒有限制封包大小，但 Device 送出的每個小封包的大小由 **SCPS0** 決定。以 Pylon API 為例：每張 YUV422Packed 彩色影像為 2,448,648Bytes，設定封包大小上限 1500，其中 IP、UDP、GVSP 的標頭共佔去了 $20+8+160=188$ Bytes，因此每個小封包的 payload 長度為 $1500-188=1312$ Bytes，也就是每個影格都要分成 $\text{ceil}(2448648 / 188) = 1867$ 個封包分別傳送。

每當 block 被傳送時，都先給予唯一的 **block_id**，再分成三部份：

- 一個 leader packet：代表 block 的開始，載明影像格式、block_id、時間戳記、影像大小；
- 數個 payload packet：代表 block 的影像資料，只有 block_id 和影像資料的片段；
- 一個 trailer packet：代表 block 的結束。

GigE Vision Protocol 詳細地處理封包遺失，但並無規定實作的細節與方法。在我們的實作裡，每個封包（leader、payload、trailer）都有唯一的 packet_id，從 Stream Channel 開始時由 1 開始遞增。在 device 端的 Stream Channel 用 cyclic queue 備份前 10 個 block 的所有封包，當 host 發現 packet_id 不連續即代表某些封包遺失，則透過 Control Channel 指定 packet_id，發出 **PACKETRESEND** 命令要求 device 重送該封包。若該封包的備份尚未被丟棄，那麼就會重新發送。

在 host 程式裡，若完整的收到一個 block 裡的所有封包（leader+payload+trailer），那就交給在 showImg() 顯示影像，showImg() 借用了 Pylon API 的秀圖函式；又，若在 timeout（預設 3ms）內仍然沒收到遺失的封包，那麼就放棄此 block。通常 FPS 在 30~50 左右，所以放棄一個影格並無大礙。

4. 成果

經實驗和封包觀測軟體，我們的實作傳輸速度在不同環境下有相當的差異，但都在 **450~800Mbps** 左右，FPS 約為 **30~40**，符合工業相機需求；此外，我們的實作能與廠商提供之 GigE Camera 溝通無礙，證明我們的實作符合標準。

5. 參考文獻

[1] GigE Vision® Specification version 2.0, 2011 AIA.