iTE SDK

Wiegand 模組開發指南

V0.9

ITE TECH. INC.

修訂記錄

修訂日期	修訂說明	頁次
2014/10/27	初建版本 V0.9	

\

目錄

1. 前言

Wiegand 介面是門禁系統或保全系統中非接觸讀卡機常見的溝通介面，ITE SOC 的 Wiegand 模組提供標準 Wiegand 的操作方式用來讀取感應卡中的 Wiegand code，ITE SOC 最多可支援兩組 Wiegand 介面，每個介面都提供的標準 Wiegand 26、Wiegand 34 及 Wiegand 37 協定,使用者也可以自己撰寫符合特定系統的協定長度。

1.1 編寫目的

介紹 Wiegand 模組之功能，說明 Wiegand 模組相關的 API 之操作及使用。

1.2 適用範圍

可視對講系統的中需要 Wiegand 介面讀卡功能的終端設備。

1.3 適用人員

軟體應用程式、驅動程式開發者。

2. Wiegand模組介紹

參考 API 原型 “sdk\driver\itp\itp_wiegand.c”

2.1 Wiegand 輸入格式

ITE SOC 支援三種輸入格式
(a). Wiegand 26
Wiegand 26 協定總共有 26 位二進制數字，其中
bit0 為 bit1~bit12 的偶同位檢查位元,
bit1~bit8 位元為設備碼(Facility code)，共 8 位元
bit9~bit24 位元為使用者資料(User Data)，共 16 位元
bit25 為 bit13~bit24 的奇同位檢查位元

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
P	F	F	F	F	F	F	F	F	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	P
P	E	E	E	E	E	E	E	E	E	E	E	E													
													O	O	O	O	O	O	O	O	O	O	O	O	P

E: Even parity
O: Odd parity
F: Facility Code
D: User Data
P: Parity bit
(b). Wiegand 34
Wiegand 34 協定總共有 34 位二進制數字，其中
bit0 為 bit1~bit16 的偶同位檢查位元,
bit1~bit8 位元為設備碼(Facility code)，共 8 位元
bit9~bit32 位元為使用者資料(User Data)，共 24 位元
bit33 為 bit17~bit32 的奇同位檢查位元

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
P	F	F	F	F	F	F	F	F	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	P
P	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E																	
																	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	P

E: Even parity
O: Odd parity
F: Facility Code
D: User Data
P: Parity bit
(c). Wiegand 37
Wiegand 37 協定總共有 37 位二進制數字，其中
bit0 為 bit1~bit18 的偶同位檢查位元,
bit1~bit35 位元為使用者資料(User Data)，共 35 位元
bit36 為 bit18~bit35 的奇同位檢查位元

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
P	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	P
P	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E																		
																		O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	P

E: Even parity
O: Odd parity
D: Data
P: Parity bit

2.2 啟動及初始化 Wiegand 介面

ITP Wiegand driver 相關程式放在”\sdk\driver\itp\itp_ wiegand.c”
ITP DRIVER 是依據 POSIX 規範實作的 API，可以使用 READ/IOCTL 等函式對 I/O DEVICE 進行如讀寫檔案般的操作。以下是基於這種規範所實做的 Wiegand ITP API。

ITE SOC 可提供兩組 Wiegand 介面，可以同時啟動,也可以擇一啟動，接下來將描述啟動流程及對應的 ioctl method

Wiegand device ID
DEVICE ID : ITP_DEVICE_ WIEGAND0 /ITP_DEVICE_ WIEGAND1 (參考“sdk\include\ite\itp.h”)

Wiegand device 註冊
itpRegisterDevice(ITP_DEVICE_WIEGAND0, & itpDeviceWiegand0);
itpRegisterDevice(ITP_DEVICE_WIEGAND1, & itpDeviceWiegand1);

使用此函式可以將 Wiegand device 註冊到 ITP 的 driver 中，使用者可以透過 ioctl/read/write 等函式來操作 Wiegand 的功能。

Wiegand device 的初始化
ioctl(ITP_DEVICE_WIEGAND0, ITP_IOCTL_INIT, NULL);
ioctl(ITP_DEVICE_WIEGAND1, ITP_IOCTL_INIT, NULL);

Wiegand device 的啟動
ioctl(ITP_DEVICE_WIEGAND0, ITP_IOCTL_ENABLE, NULL);
ioctl(ITP_DEVICE_WIEGAND1, ITP_IOCTL_ENABLE, NULL);

2.3 設定 Wiegand 協定種類

在實際調用 read method 前，還需要先設定 Wiegand 協定的模式，目前 Wiegand 模組提供 3 種協定模式 : 26 bits、34 bits、37 bits。讀取的 Wiegand code 會根據 2.1 章節中描述的 Wiegand 輸入格式,做奇偶同位檢查，如果檢查錯誤或是 bit count 不符合，則回傳的 card id 字串為”0”。

設定 Wiegand 協定的 Ioctl method 為

```
int bit_count;
bit_count = 26;
ioctl(ITP_DEVICE_WIEGAND0, ITP_IOCTL_SET_BIT_C, &bit_count);
```

使用者也可以根據特定系統，更改成自訂協定模式， 相關細節請參考 3.1.2 章節

2.4 讀 Wiegand code

當啟動及設定協定後，程序需要週期性的調用 read method 來得知是否有 card event 發生

```
char* card_id;

read(ITP_DEVICE_WIEGAND0, &card_id, 0);
```

當沒有 card event 發生或是讀卡發生錯誤，回傳 card id 所帶字串為”0”

Wiegand code 的處理方式
Wiegand 回傳 bit count 預留 64bits，當實際 Wiegand code bit count 不足 64bits 的部分會補’0’

WG bit count 設定為 26
當 card event 發生時，會將得到的資料作奇偶同位檢查，檢查成功後會移除 bit0 及 bit25，不足 64 bit 的部分會補’0’

```
bit_count = ithCodecWiegandReadCard(ctxt->index, &value);
value = (value & 0x1FFFFFE) >> 1;
sprintf(ctxt->card_id, "%016X", value);
```

WG bit count 設定為 34
當 card event 發生時，會將得到的資料作奇偶同位檢查，檢查成功後會移除 bit0 及 bit33，不足 64 bit 的部分會補’0’

```
bit_count = ithCodecWiegandReadCard(ctxt->index, &value);
value = (value & 0x1FFFFFFFE) >> 1;
sprintf(ctxt->card_id, "%016X", value);
```

WG bit count 設定為 37
當 card event 發生時，會將得到的資料作奇偶同位檢查，檢查成功後會移除 bit0 及 bit36，不足 64 bit 的部分會補’0’

```
bit_count = ithCodecWiegandReadCard(ctxt->index, &value);
value = (value & 0xFFFFFFFFE) >> 1;
sprintf(ctxt->card_id, "%016X", value);
```

使用者也可以自訂 card_id 的處理，相關細節請參考 3.1.2 章節

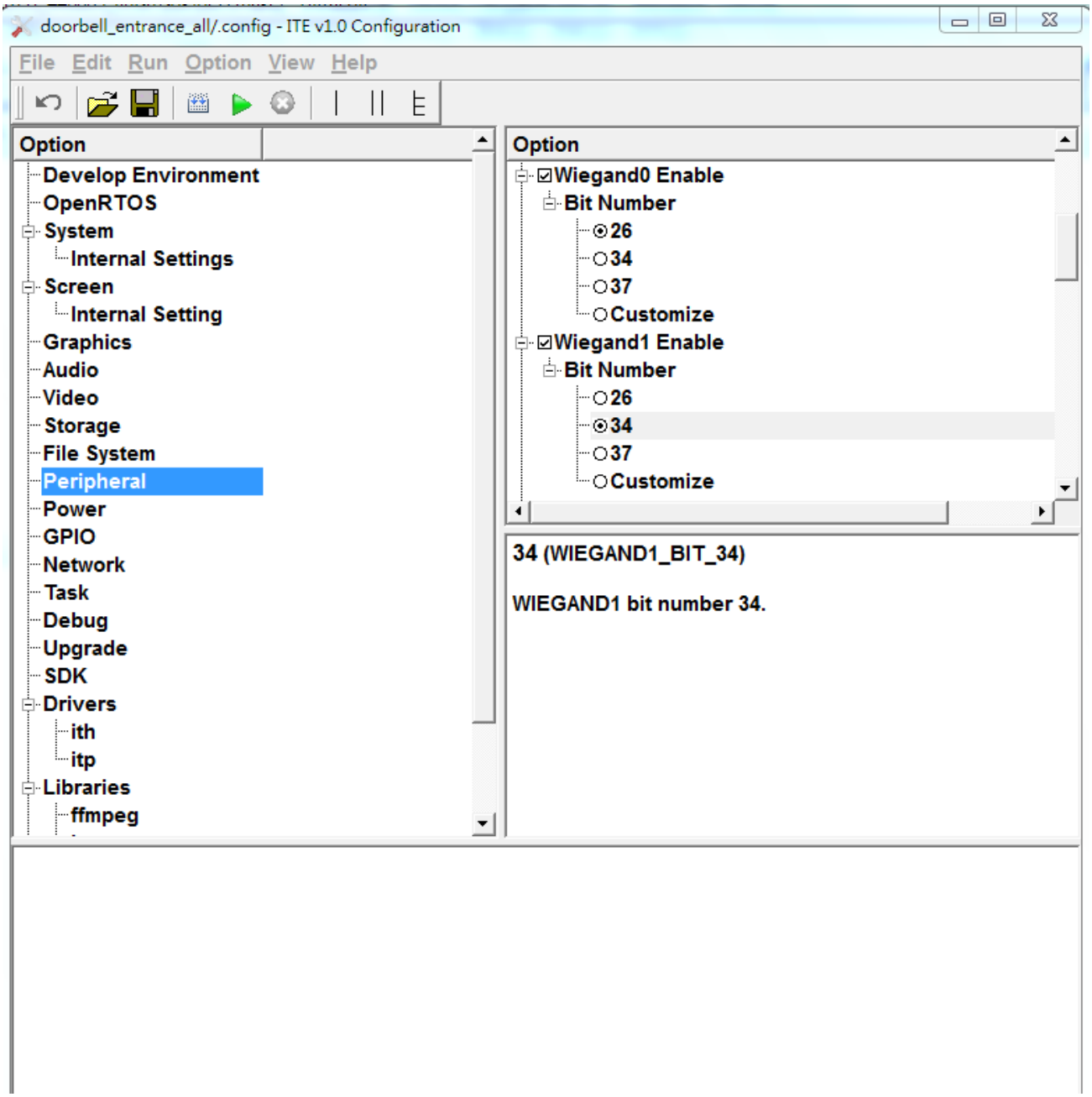
3. 軟件配置說明

3.1 KConfig

Wiegand 模組可藉由 Kconfig 來做基本的設定

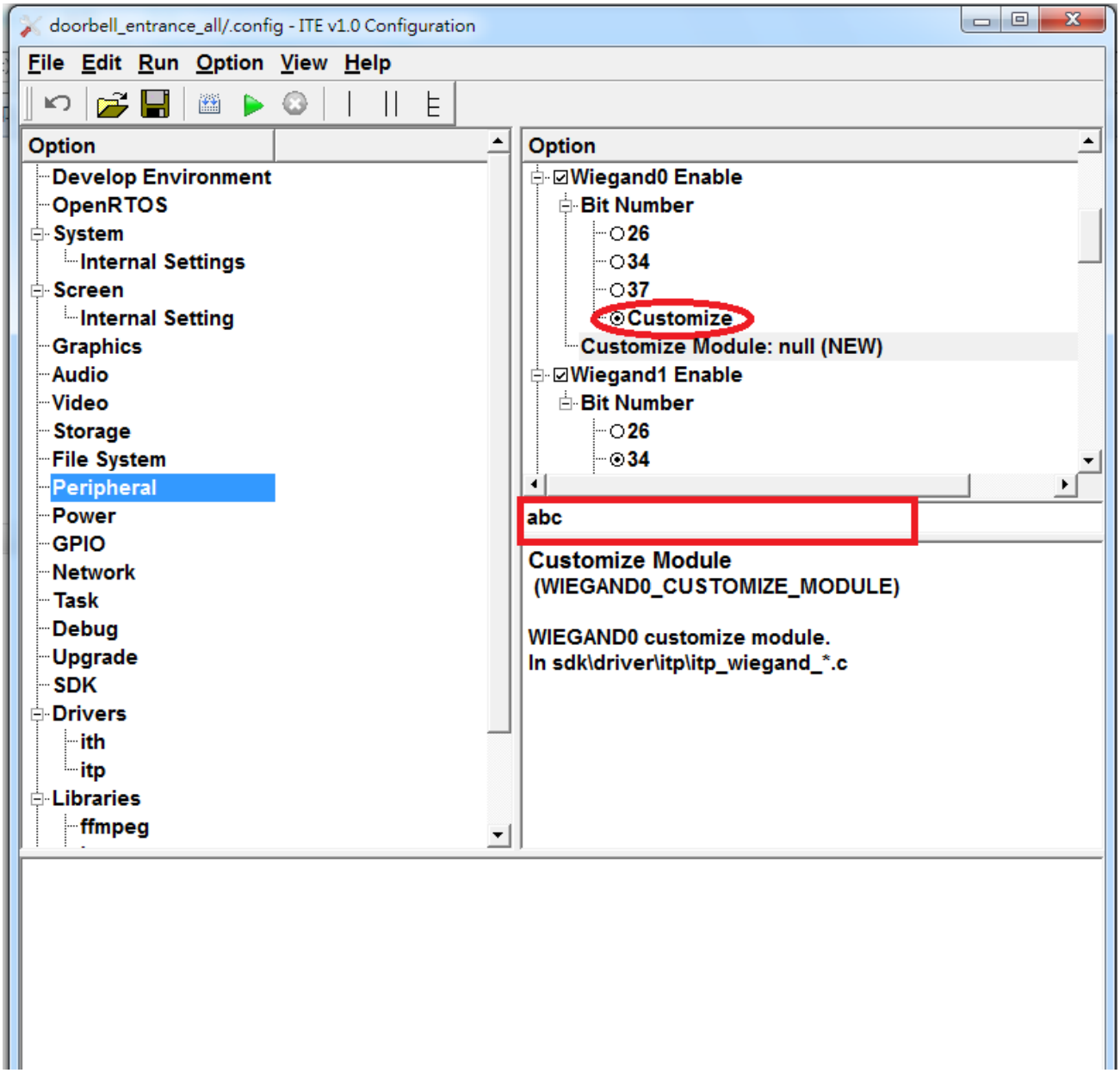
3.1.1 啟動 Wiegand 模組

Option 選擇 Peripheral，根據使用者的系統應用，打開一組或兩組的 Weigand 介面，同時可以個別選擇協定模式，如下圖所示



3.1.2 客製化

有些情況下，使用者需要自訂 Wiegand 協定，可以藉由實作相關程式碼來達成，步驟方法如下所示



Option 選擇”Peripheral”，勾選需要客製化之 Wiegand 模組後，在”Bit Count”選擇”Customize”，則會出現”Customize Module:null”選項， 連按兩下“Customize Module:null”即可從紅色框框輸入檔名更換 sdk\driver\itp\wiegand\itp_wegand_*.c 的 driver

Example: 若此 project 使用的 abc 的 driver，則連按兩下 “Customize Module:null”,
在紅色框框輸入 driver 的名稱,就可以替換, 同時在 sdk\driver\itp 底下需有一 itp_wiegand_abc.c 的 driver.

如果要修改 Wiegand 協定，請參考“sdk\driver\itp\itp_wiegand.c” ，修改 WiegandRead function

3.2 範例

```
#include "ite/itp.h"            //for ITP_DEVICE_WIEGAND0 & ITP_DEVICE_WIEGAND1

int main(void)
{
int bit_count;

      // wiegand 0
      itpRegisterDevice(ITP_DEVICE_WIEGAND0, &itpDeviceWiegand0);
      ioctl(ITP_DEVICE_WIEGAND0, ITP_IOCTL_INIT, NULL);
      ioctl(ITP_DEVICE_WIEGAND0, ITP_IOCTL_ENABLE, NULL);
      bit_count = 26;
      ioctl(ITP_DEVICE_WIEGAND0, ITP_IOCTL_SET_BIT_COUNT, &bit_count);

      // wiegand 1
      itpRegisterDevice(ITP_DEVICE_WIEGAND1, &itpDeviceWiegand1);
      ioctl(ITP_DEVICE_WIEGAND1, ITP_IOCTL_INIT, NULL);
      ioctl(ITP_DEVICE_WIEGAND1, ITP_IOCTL_ENABLE, NULL);
      bit_count = 34;
      ioctl(ITP_DEVICE_WIEGAND1, ITP_IOCTL_SET_BIT_COUNT, &bit_count);

      while(1)
      {
          char* card_id0, card_id1;

      read(ITP_DEVICE_WIEGAND0, &card_id0, 0);

  read(ITP_DEVICE_WIEGAND1, &card_id1, 0)

          usleep(10000);
}
}
```