

**Making a channel editor, such as Vega,
through RS232 and USB
in Windows system**

Document Summary

--

Change History

[illegible]

Table of Contents

1. RS232 FUNCTION AND SPECIFICATION	5
1.1. FUNCTION IN LOADER	5
1.2. FUNCTIONS IN FIRMWARE	5
2. STRUCTURE OF COMMUNICATION BLOCK	6
2.1. BASIC STRUCTURE OF BLOCK	6
3. FLOW CHART	7
3.1. UPLOAD MECHANISM	7
3.2. DOWNLOAD MECHANISM	8
4. TYPE	9
5. TOOLS	10
5.1. GETCRC16	10
5.2. GET16BIT	10
5.3. PUT16BIT	10
5.4. GET24BIT	11
5.5. PUT24BIT	11
5.6. GET32BIT	12
5.7. PUT32BIT	12
5.8. EXTRACTMJD	12
5.9. MAKEMJD	13
6. DEVICE (RS232 AND USB)	14
6.1. DLL_OPEN_SERIALPORT	14
6.2. DLL_CLOSE_SERIALPORT	14
6.3. DLL_AUTOBOOT	14
6.4. DLL_SERIALUP	15
6.5. DLL_SERIALIZEDOWN	15
6.6. DLL_INITUSB_FUNC	15
6.7. DLL_USBDOWN	15
6.8. DLL_USBUP	16
6.9. DLL_CANCELUSB_FUNC	16

7. DATA	17
7.1. DLL_GETPACKEDFLASHDATA.....	17
7.2. DLL_PUTPACKEDFLASHDATA	17
7.3. DLL_GETEEPROMDATA	17
7.4. DLL_PUTEEPROMDATA.....	18
7.5. DLL_PACKFLASHDATA	18
7.6. DLL_UNPACKFLASHDATA	18
8. ETC FUNCTIONS.....	19
8.1. DLL_GETMODELIDFROMPACKEDFLASHDATA.....	19
8.2. DLL_GETMODELID	19
8.3. DLL_GETSYSTEMIDFROMPACKEDFLASHDATA	19
8.4. DLL_GETSYSTEMID.....	20

1. RS232 Function and Specification

1.1. Function in loader

Downloading the firmware through RS232

TFD-DOWN.EXE made by TOPFIELD is used for downloading firmware.

1.2. Functions in firmware

Downloading/Uploading the channel data.

VEGA.EXE made by TOPFIELD for downloading/uploading/editing the channel data.

2. Structure of communication block

This chapter explains the structure of communication block Host(PC) and Device(Set-Top Box) through RS232.

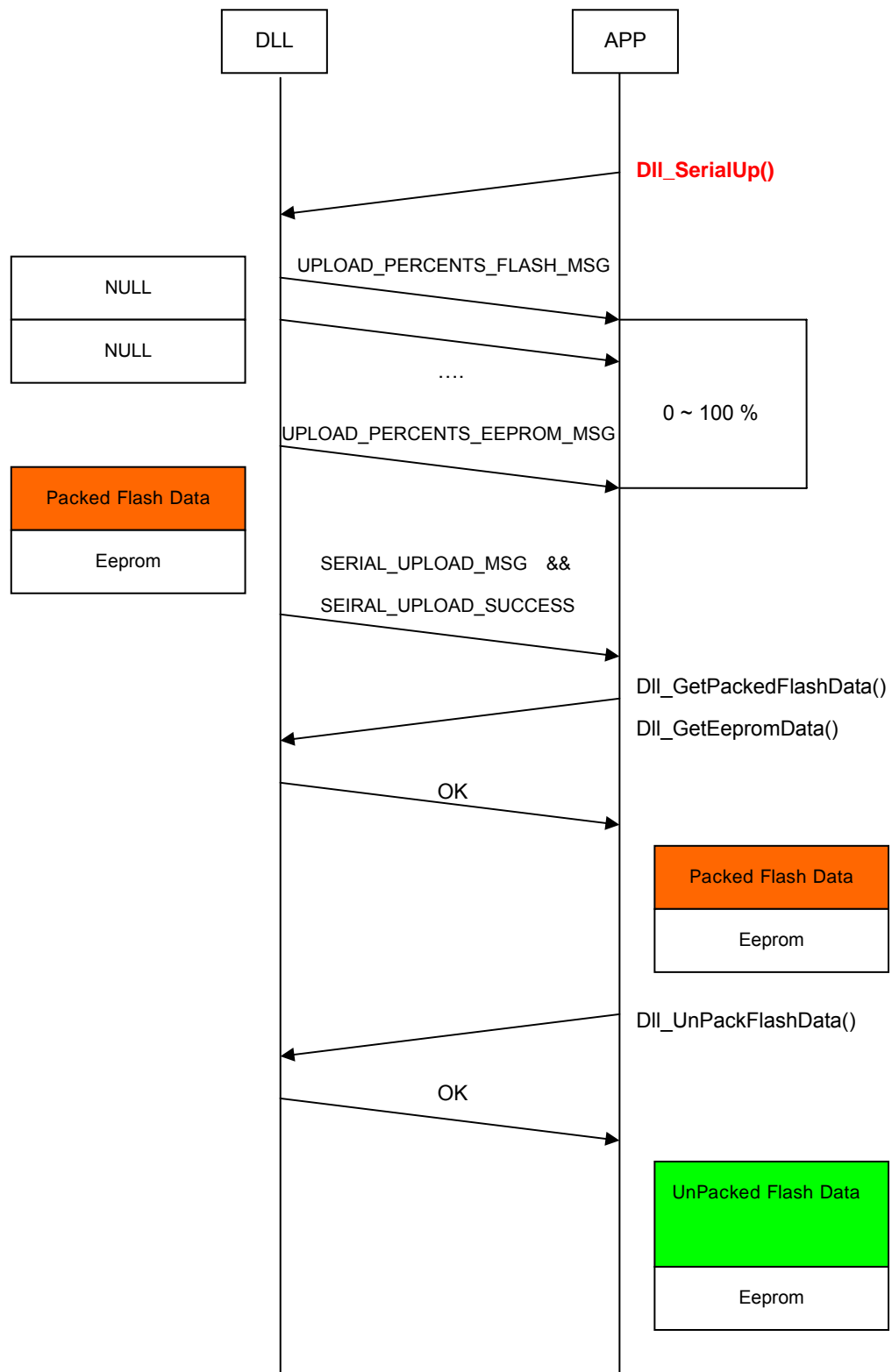
2.1. Basic structure of block

Offset	Field	Size	Description
0	LENGTH	2	Total size of block
2	CRC16	2	CRC16 of DATA in block
4	DATA	LENGTH – 2	Contents of data

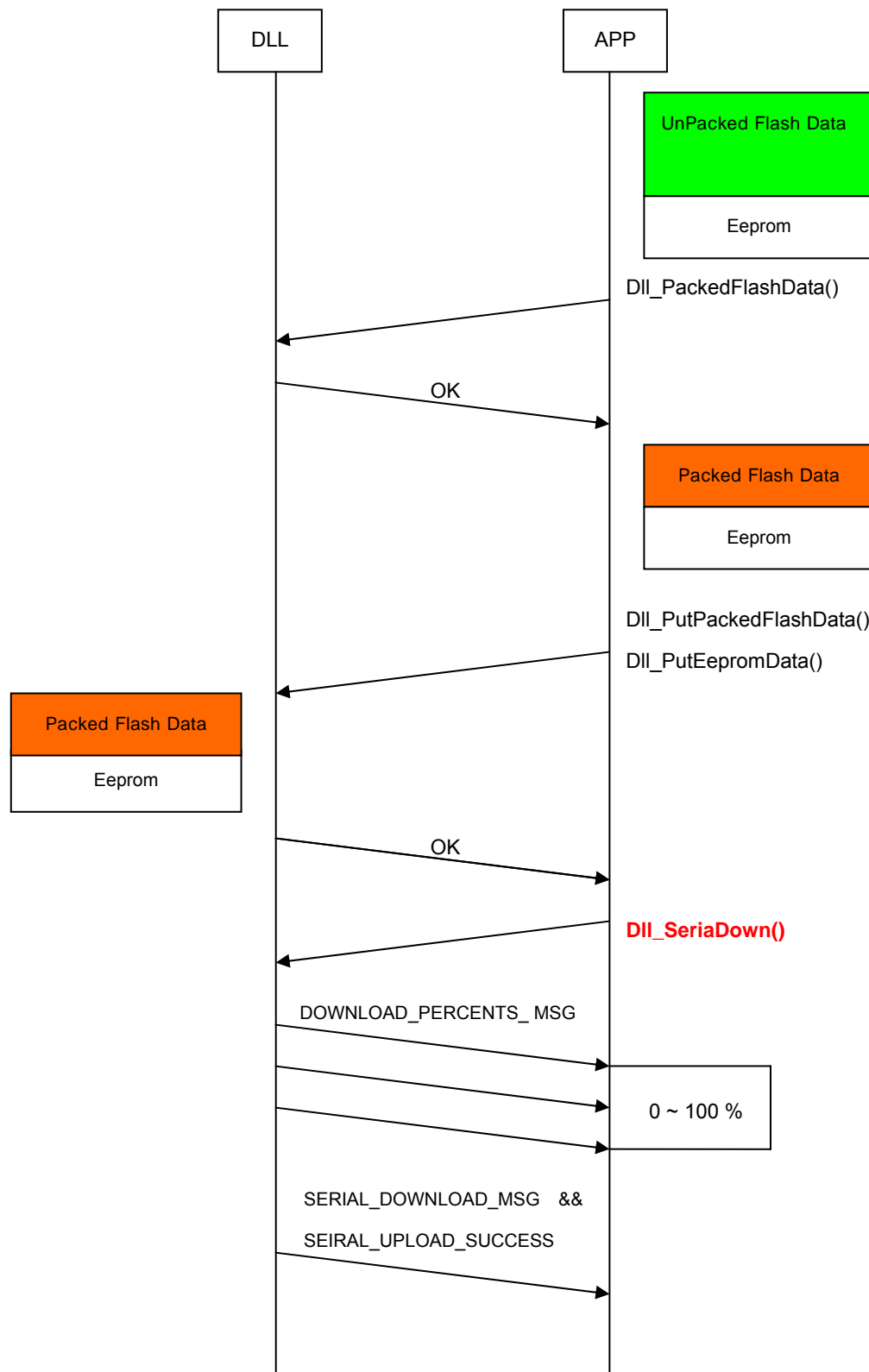
Maximum size of block is 0x8002(32770).

3. FLOW CHART

3.1. Upload Mechanism



3.2. Download Mechanism



4. TYPE

Data types are defined as follows.

typedef unsigned char	byte;
typedef unsigned short	word;
typedef unsigned long	dword;

5. TOOLS

5.1. GetCrc16

CRC-16 value can be calculated by using the following polynomial with initial value 0.

$$P(x) = x^{16} + x^{15} + x^2 + x^0$$

```
word GetCrc16
```

```
(
    void      *data,           In
    dword     size             In
);
```

Parameters:

Parameter	Meaning
data	Data pointer to get CRC-16
size	Size of data
Return value	Calculated CRC-16 value

5.2. Get16bit

It reads 16-bit data in fixed address.

```
dword Get16bit
```

```
(
    void      *addr           In
);
```

Parameters:

Parameter	Meaning
addr	Address of data to be read
Return value	16-bit Value after reading

example) addr → 0010 0101 0101 0011 0110 1011 0101 1101 0101 0000 1111...₂

return value = 0000 0000 0000 0000 **0010 0101 0101 0011**₂

5.3. Put16bit

It replaces 16-bit data in fixed address.

```
void Put16bit
```

```
(
    void      *addr,           Out
    word      data             In
);
```

Parameters:

Parameter	Meaning
addr	Address of data to be replaced
data	16-bit Value to be replaced

example) Previous Execution :

addr → 0010 0101 0101 0011 0110 1011 0101 1101 0101 0000 1111...₂

data = **0000 0111 1101 0000**₂

After Execution:

addr → **0000 0111 1101 0000** 0110 1011 0101 1101 0101 0000 1111...₂

5.4. Get24bit

It reads 24-bit data in fixed address

```
dword Get24bit
(
    void      *addr
);
```

In

Parameters:

Parameter	Meaning
addr	Address of data to be read
Return value	24-bit value after reading

example) addr → 0010 0101 0101 0011 0110 1011 0101 1101 0101 0000 1111...₂

return Value = 0000 0000 **0010 0101 0101 0011 0110 1011**₂

5.5. Put24bit

It replaces 24-bit data in fixed address.

```
void Put24bit
(
    void      *addr,
    dword     data
);
```

Out
In

Parameters:

Parameter	Meaning
addr	Address to be replaced
data	24-bit value to be replaced

example) Previous Execution:

addr → 0010 0101 0101 0011 0110 1011 0101 1101 0101 0000 1111...₂

data = 0000 0000 **1100 0011 0000 0111 1101 0000**₂

After Execution:

addr → 1100 0011 0000 0111 1101 0000 0101 1101 0101 0000 1111...₂

5.6. Get32bit

It reads 32-bit data in fixed address.

```
dword Get32bit
```

```
(  
    void      *addr          In  
);
```

Parameters:

Parameter	Meaning
addr	Address of data to be read
Return value	32-bit value after reading

example) addr → 0010 0101 0101 0011 0110 1011 0101 1101 0101 0000 1111...₂

return = **0010 0101 0101 0011 0110 1011 0101 1101**₂

5.7. Put32bit

It replaces 32-bit data in fixed address.

```
void Put32bit
```

```
(  
    void      *addr,          Out  
    dword     data            In  
);
```

Parameters:

Parameter	Meaning
addr	Address to be replaced
data	32-bit value to be replaced

example) Previous Execution:

addr → 0010 0101 0101 0011 0110 1011 0101 1101 0101 0000 1111...₂

data = **1111 0101 1100 0011 0000 0111 1101 0000**₂

After Execution

addr → 1111 0101 1100 0011 0000 0111 1101 0000 0101 0000 1111...₂

5.8. ExtractMjd

It gets date information(year/month/day) from MJD(Modified Julian Date) format.

```
byte ExtractMjd
```

```
(
    word    mjd,           In
    word    *year,         Out
    word    *month,        Out
    word    *day,           Out
    word    *weekDay        Out
);
```

Parameters:

Parameter	Meaning
mjd	MJD Value
year	Year
month	Month
day	Date
weekDay	A day of week(0=Monday, 1=Tuesday, ..., 6=Sunday)
Return value	0 = Not a Mjd format 1 = Complete

5.9. MakeMjd

It gets MJD from date information of year/month/day .

```
word MakeMjd
```

```
(
    word    year,           In
    byte    month,          In
    byte    day             In
);
```

Parameters:

Parameter	Meaning
year	Year
month	Month
day	Day
Return value	0 = Can not support year or month , Day 0x3AAC..0xFFFF = MJD value

6. DEVICE (RS232 and USB)

6.1. Dll_OpenSerialPort

It opens device if the device can be opened.

```
BOOL Dll_OpenSerialPort
```

```
(
    char    *portName,           In
    dword   baudRate,           In
    word    portId              In
);
```

Parameters:

Parameter	Meaning
portName	RS232 port name
baudRate	Baud rate
portId	Port ID (com1:0, com2:1)
Return value	1 – Success 0 – Fail

Example) If you want to open COM1 port with 115200 baud rate

```
If(Dll_OpenSerialPort("COM1",115200,0))
    MessageBox("SerialPort Open Success");
else
    MessageBox("SerialPort Open Fail");
```

6.2. Dll_CloseSerialPort

It closes device.

```
void CloseSerialPort
```

```
(
    void
);
```

6.3. Dll_AutoReboot

It reboots STB(set-top box) If STB status is "End" and device is opened.

```
void Dll_AutoReboot
```

```
(
    void
);
```

6.4. Dll_SerialUp

It gets flash and eeprom data from STB If Dll_OpenSerialPort is success.

```

BOOL    Dll_SerialUp
(
    HWND hWnd,                In
    WPARAM wParam             In
);

```

Parameters:

Parameter	Meaning
hWnd	Windows handle to receive message when the job is completed or failed
wParam	WParam value to receive with WM_COMMAND
Return value	TRUE – Upload start is success. FALSE – Upload start is fail.

6.5. Dll_SerialDown

It sends packed flash data and eeprom data to STB If Dll_OpenSerialPort is success

```

BOOL    Dll_SerialDown
(
    CString *cwdPath,         In
    HWND hWnd,               In
    WPARAM wParam             In
);

```

Parameters:

Parameter	Meaning
cwdPath	Application path (current working directory)
hWnd	Windows handle to receive message when the job is completed or failed
wParam	WParam value to receive with WM_COMMAND
Return value	TRUE – Download start is success. FALSE – Download start is fail.

6.6. Dll_InitUsbFunc

It initializes the USB functions.

```

BOOL    Dll_InitUsbFunc
(
    void
);

```

6.7. Dll_UsbDown

It sends the packed flash data and eeprom data to STB via USB.

```

BOOL    Dll_UsbDown
(
    CString *cwdPath,
    HWND hWnd,
    WPARAM wParam
);

```

In
In
In

Parameters:

Parameter	Meaning
cwdPath	Application path (current working directory)
hWnd	Windows handle to receive message when the job is completed or failed
wParam	WParam value to receive with WM_COMMAND
Return value	TRUE – Download start is success. FALSE – Download start is fail.

6.8. Dll_UsbUp

It gets the flash and eeprom data from STB via USB.

```

BOOL    Dll_UsbUp
(
    HWND hWnd,
    WPARAM wParam
);

```

In
In

Parameters:

Parameter	Meaning
hWnd	Windows handle to receive message when the job is completed or failed
wParam	WParam value to receive with WM_COMMAND
Return value	TRUE – Upload start is success. FALSE – Upload start is fail.

6.9. Dll_CancelUsbFunc

It forces to terminate the task of USB Up or USB down.

```

BOOL    Dll_CancelUsbFunc
(
    void
);

```


7. Data

7.1. Dll_GetPackedFlashData

It gets packed flash data from VegaMfcDll.dll. It is possible when Application receives SERIAL_UPLOAD_MSG and SERIAL_UPLOAD_SUCCESS messages from VegaMfcDll.dll

```

BOOL    Dll_GetPackedFlashData
(
    void *pBuf                                Out
);
  
```

Parameters:

Parameter	Meaning
pBuf	Pointer of data to get a packed Flash data in VegaMfcDll.dll
Return value	TRUE – Success FALSE – Fail

7.2. Dll_PutPackedFlashData

It puts a packed flash data in VegaMfcDll.dll. It needs to execute “Dll_SerialDown” function

```

void Dll_PutPackedFlashData
(
    void *pBuf                                In
);
  
```

Parameters:

Parameter	Meaning
pBuf	Pointer of data to be written
Return value	void

7.3. Dll_GetEepromData

It gets eeprom data from VegaMfcDll.dll. It is possible when Application receives SERIAL_UPLOAD_MSG and SERIAL_UPLOAD_SUCCESS messages from VegaMfcDll.dll

```

BOOL    Dll_GetEepromData
(
    void *pBuf                                Out
);
  
```

Parameters:

Parameter	Meaning
pBuf	Pointer of data to get the eeprom data in VegaMfcDll.dll
Return value	void

7.4. Dll_PutEepromData

It puts a EEPROM data in VegaMfcDll.dll. It needs to execute "Dll_SerialDown" function.

```
void    Dll_PutEepromData
(
    void *pBuf                In
);
```

Parameters:

Parameter	Meaning
pBuf	Pointer of data to be written
Return value	void

7.5. Dll_PackFlashData

It packs the unpacked flash data in the packed flash data

```
BOOL    Dll_PackFlashData
(
    void *pUnPackedFlash,    In
    void *pPackedFlash       out
);
```

Parameters:

Parameter	Meaning
pUnPackFlash	Pointer of data to be packed in pPackedFlash
pPackFlash	Pointer of data to be written in the packed flash data
Return value	TRUE – Success FALSE – Fail

7.6. Dll_UnPackFlashData

It unpacks the packed flash data in the unpacked flash data

```
BOOL    Dll_UnPackFlashData
(
    void *pPackedFlash,      In
    void *pUnPackedFlash     Out
);
```

Parameters:

Parameter	Meaning
pUnPackedFlash	Pointer of data to be packed in pPackedFlash
pPackedFlash	Pointer of data to be written in packed flash data
Return value	TRUE – Success FALSE – Fail

8. Etc Functions

They are to get modelID and systemID from the packed flash data or VegaMfcDll.dll

8.1. Dll_GetModelIdFromPackedFlashData

It gets modelID from the packed flash data.

```

BOOL Dll_GetModelIdFromPackedFlashData
(
    void *pPack,           in
    void *pModelId        out
);

```

Parameters:

Parameter	Meaning
pPack	Pointer of the packed flash data
pModelId	Pointer of modelID to be written
return value	TRUE - Success to get modelID FALSE – Failure to get modelId

8.2. Dll_GetModelId

It gets modelID from “VedaMfcDll.dll”.

```

BOOL Dll_GetModelIdFromPackedFlashData
(
    void *pModelId        out
);

```

Parameters:

Parameter	Meaning
pModelId	Pointer of modelID to be written
return value	TRUE - Success to get modelID FALSE – Failure to get modelId

8.3. Dll_GetSystemIdFromPackedFlashData

It gets systemID from the packed flash data.

```

word Dll_GetModelIdFromPackedFlashData
(
    void *pPack           in
);

```

Parameters:

Parameter	Meaning
-----------	---------

pPack	Pointer of the packed flash data
return value	systemID

8.4. Dll_GetSystemId

It gets systemID from "VegaMfcDll.dll".

```
word Dll_GetSystemId  
(  
    void  
);
```

Parameters:

Parameter	Meaning
return value	systemID