

HS XMODEM v1.1 User Manual

Revision: 1.1

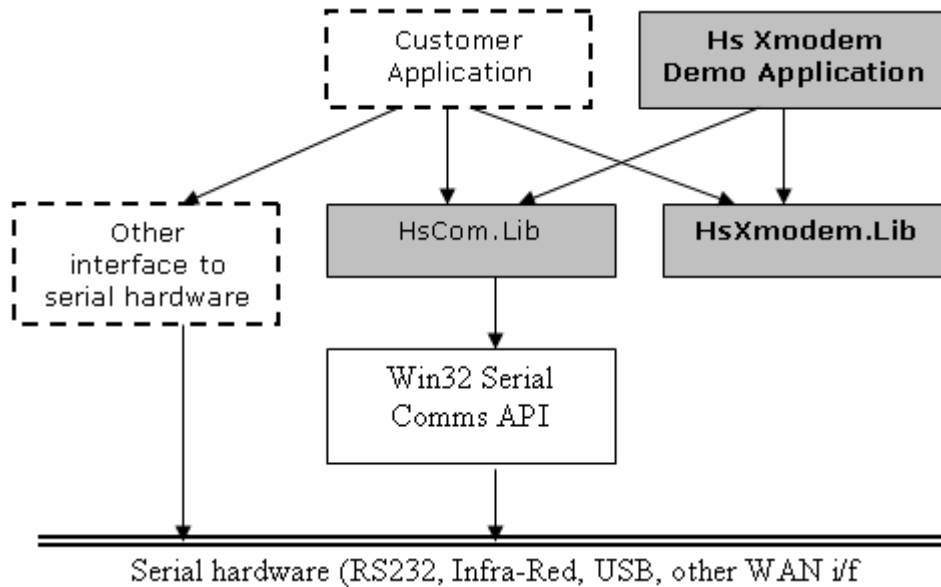
Date: 04 December 2006

HS XMODEM V1.1 USER MANUAL	1
REVISION: 1.1	1
DATE: 04 DECEMBER 2006	1
1 INTRODUCTION	2
2 HS XMODEM API	3
2.1 HSXMINT	3
2.1.1 DEFINITION OF HS_XM_API_T STRUCTURE	4
2.2 HSXMSTARTSEND	6
2.3 HSXMSTARTRECEIVE	7
2.4 HSXMABORT	7

1 Introduction

HS XMODEM is a software library in C (supplied with full source code) that provides a programmer with the off-the-shelf support for XMODEM protocol data transfer capability. Support for both sender and receiver are provided. Other features include 1024 block size vs 128, CRC vs checksum, configurable timers

The library has a flexible and modular architecture:



On initialisation user code provides interface callbacks for the services used by Xmodem protocol module: timer management, serial transmission and reception, memory management and event callbacks. This architecture makes it easy to port Xmodem protocol module to any software environment.

Xmodem module handles all protocol information flow, error recovery, acknowledgements and timeouts. When it is appropriate to send next block of data Hs Xmodem will get next memory block from user application. Similarly, when data has been received Hs Xmodem module will get the next block of memory from user application to store data into.

2 HS XMODEM API

2.1 HsXmInit

Extern int HsXmInit(*hs_xm_api_t* *init, long *handle);

DESCRIPTION

Initializes HS XMODEM Library - this function must be called first before any other functions are called. Init structure contains function pointers which must be initialised with function addresses in user space. HS XMODEM module will call these functions when it needs to manage timers, memory and serial communications

PARAMETERS

Type	Name	Description
<i>hs_xm_api_t</i>	*init	Pointer to initialisation structure: see next section for description
long	*handle	User code must provide pointer to long variable to store the Hs Xmodem context handle returned by Hs Xmodem on exit from this function. Effectively, XMODEM allows multiple concurrent Xmodem sessions some of which may be receivers and some senders. However user code must uniquely associate an Xmodem handle to a physical device such as COM port.

RETURNS

- *HS_XM_RC_INVALID_PAR* – invalid parameter(s) - one of function pointers part of init structure or the pointer to handle variable are NULL
- *HS_XM_RC_NO_FREE_CTX* – no free contexts – Hs Xmodem module cannot handle any more concurrent sessions
- *HS_XM_RC_OK* - success

2.1.1 Definition of `hs_xm_api_t` structure

Function pointer	Prototype and Description
<code>hs_xm_get_buf_t *get_tx_buffer</code> <code>hs_xm_get_buf_t *get_rx_buffer</code>	<p>Prototype <code>unsigned char *hs_xm_get_buf_t(long handle, int length, int *given_len);</code></p> <p>Parameters: handle – HS Xmodem module handle length – requested length of memory (Xmodem may request either a next block to send or a space for next block to copy received data into. Pointer to integer to length of block actually given to HS Xmodem module.</p> <p>Return: pointer to memory buffer in user code or NULL if no memory available or nothing to give (end of transmission or end of file)</p> <p>Description: This function will be called from HS Xmodem module when it needs to send next block of data (on reception of ACK or NAK from remote) or when data has been successfully received and now needs to be copied to user space from HS Xmodem space.</p>
<code>hs_xm_start_timer_t *start_timer_fn</code>	<p>Prototype: <code>long hs_xm_start_timer_t(long handle, unsigned long secs, hs_xm_timer_callback_t *callback);</code></p> <p>Parameters: handle – HS Xmodem module handle secs – number of seconds to timeout after. *callback – function pointer – callback in Hs Xmodem code to be called after time specified in secs expired</p> <p>Timer callback prototype: <code>void hs_xm_timer_callback_t(long handle);</code> Handle – HS Xmodem handle.</p> <p>Return: Timer handle. Same handle will be passed in a function to destroy (stop) the timer.</p> <p>Description: This function in user code will be called from Hs Xmodem code to start a timer with a specified number of seconds. Provided callback function must be called from user code when the time elapsed.</p>
<code>hs_xm_stop_timer_t *stop_timer_fn</code>	<p>Prototype: <code>typedef void hs_xm_stop_timer_t(long handle);</code></p> <p>Parameters: Handle – timer handle, the same value as was returned with <code>start_timer_fn</code></p> <p>Return: No return</p> <p>Description: This function will be used by HS Xmodem module to stop (destroy) a timer previously started with <code>start_timer_fn</code>.</p>
<code>hs_xm_ev_fn_t *ev_callback</code>	<p>Prototype: <code>void hs_xm_ev_fn_t(long handle, int ev_code);</code></p> <p>Parameters: Handle – Application handle, same are passed to <code>XmInit</code> function in init structure.</p> <p>Ev_code – Hs Xmodem Library event code explained below:</p> <ul style="list-style-type: none"> • <code>HS_XM_USER_EV_REMOTE_TERMINATION</code> – transfer cancelled by remote end

	<ul style="list-style-type: none"> • HS_XM_USER_EV_LOCAL_TERMINATION – transfer terminated by Hs Xmodem module due to an unrecoverable error. • HS_XM_USER_EV_SEND_TRANSFER_COMPLETE – XMODEM send operation completed successfully • HS_XM_USER_EV_RECV_TRANSFER_COMPLETE – Xmodem receive operation completed successfully <p>Return: No return</p> <p>Description: This function will be called by Hs Xmodem module with one of above described events to signal file transfer termination or completion.</p>
hs_xm_tx_t *tx_data	<p>Prototype: void hs_xm_tx_t(long handle, unsigned char *buf, int len);</p> <p>Parameters: Handle – Application handle, same are passed to Xmlnit function in init structure</p> <p>*buf – pointer to buffer to transmit over serial interface length – length of buffer to transmit</p> <p>Return: No return</p> <p>Description: This function will be called from Hs Xmodem module whenever it needs to transmit a buffer to remote end over serial interface.</p>
hs_xm_rx_t *rx_data	<p>Prototype: void hs_xm_rx_t(long handle, unsigned char c);</p> <p>Parameters: Handle – HS Xmodem module handle C – character received from remote end</p> <p>Return: No return</p> <p>Description: This is a function pointer to a function callback in Hs Xmodem code which the user code is to call whenever data (single character) is received from serial port and now needs to be passed to Hs Xmodem protocol engine.</p> <p>This function pointer will be set by Hs Xmodem module on return from HsXmlnit call. The user code must save this function pointer.</p>
long user_handle	<p>Application context handle. This value shall be stored unmodified by HS XMODEM in associated context and passed back in functions callbacks as defined above.</p>

2.2 HsXmStartSend

Extern int HsXmStartSend(long handle, hs_xm_options_t *opt)

DESCRIPTION

This function is called to initiate Xmodem File Send operation

PARAMETERS

Type	Name	Description
Long	handle	HS Xmodem context handle
hs_xm_options_t	*opt	<p>Pointer to options structure in user code as follows: typedef struct { int crc16; int pkt_1k; int sender_timeout; } hs_xm_options_t;</p> <p>crc16: 1= support 2 byte CRC instead of checksum pkt_1k: 1=use 1024 payload packets instead of 128 sender_timeout: Xmodem sender timeout in seconds, set to 0 to use default timeout of 60 seconds. Sender timeout is a high level timer used by Xmodem sender engine. Transfer times out on expiration of this timer. Hs Xmodem module implements XMODEM protocol version that is completely receiver driven, Receiver has a shorter non-configurable 10 second retry timer.</p>

RETURNS

- *HS_XM_RC_INVALID_PAR – invalid parameter(s) – invalid handle or options pointer*
- *HS_XM_RC_OK – success*

2.3 HsXmStartReceive

extern int HsXmStartReceive(long handle, hs_xm_options_t *opt)

DESCRIPTION

This function is called to initiate Xmodem File Receive operation

PARAMETERS

Type	Name	Description
long	handle	HS Xmodem context handle
hs_xm_options_t	*opt	Pointer to options structure in user code as follows: typedef struct { int crc16; int pkt_1k; int sender_timeout; } hs_xm_options_t; crc16: 1= attempt 2 byte CRC instead of checksum pkt_1k: 1=support 1024 payload packets sender_timeout – unused for receiver

RETURNS

- *HS_XM_RC_INVALID_PAR* – invalid parameter(s) - invalid handle or options pointer
- *HS_XM_RC_OK* – success

2.4 HsXmAbort

extern void HsXmAbort(long handle)

DESCRIPTION

This function is used to abort current transfer in progress.

PARAMETERS

Type	Name	Description
long	handle	HS Xmodem context handle

RETURNS

- *HS_XM_RC_INVALID_PAR* – invalid parameter(s) –invalid handle
- *HS_XM_RC_OK* - success