EMS Side Serial Communications Implementation Description

Version : 0.2
Date : 14th of October 2008
Author : Fred Cooke, based on the work of myself, Stu and others.

Foreword

The purpose of this document is to aid in understanding of the source code, as a basis for discussion on the subject, and as a reference work for those implementing the handler at the other end of the data stream. This document assumes that the latest serial specification has been read and understood.

States

The EMS can be in four states.

- Send - Sending a data packet.
- Receive - Receiving a data packet.
- Listen - Listening/Waiting for a packet /Asyncronous Data Log Preperation.
- Ignore - Ignoring incoming data while processing an already received packet.

Out of boot the device will be in listen mode.

Special Bytes

Three bytes are considered special and are escaped before transmission when not being used for their specific purpose :

- Start - Indicates the start of a data packet.
- Stop - Indicates the end of a data packet.
- Escape – Used to indicate the presence of an escaped special byte following it.

Interrupt Handling Semantics

When the serial ISR runs it reads the flags register. Depending upon what is found in that register it will perform certain actions. Currently errors are always counted but otherwise ignored as it is assumed that the escaping scheme and checksum logic will catch bad data. The counting of error flag occurance is primarily to alert the user to a possible issue with their communications equipment.

If the flag indicating that the receive register is full is set then the code to handle the reception of a byte of data is run as described in a following section. If the receive code runs, the send code does not run and vice versa with receive taking priority. This should never be an issue anyway as control of the ISR enable bits is integral to the design.

When the send register empty flag is set, and reception is not occuring the send handling code will run as described in a following section.

Every time the serial ISR runs all relevant error flags are checked and appropriate counters incremented where required.

For specifics on what exactly is done in the serial ISR code for sending and receiving, please see the appropriate sections of this document.

Reception Of Data In Listen Mode

The following will be handled by the interrupt service routine for the serial channel when triggered by the receive register full flag.

Bytes received while listening are checked to see if they are the start byte, if they are not, and we are not currently receiving a packet, they are dropped. When the start byte is received, a flag is set to indicate that we are receiving a data packet. If we were already receiving a data packet the process is reset. All bytes received while the receive flag is set are checked to see if they are the escape byte, or the stop byte, if they are not they are stored into the buffer directly. If one is the escape byte it is not stored, rather a flag is set to indicate that the next byte received should be checked, converted and stored. If the next byte is found to not be the escaped version of one of the three special bytes then the process is reset as the packet must have been malformed or corrupt. If the next byte is the escaped version of one of the three escapable bytes the appropriate pre-escaped version is stored instead. Each byte stored is added to the checksum variable on the fly. This process continues until one of two things happens. Firstly, the buffer could fill up indicating corruption or a packet that does not honour convention. Secondly, a stop byte could be received indicating the end of the packet.

When a packet is complete and buffered (the stop byte is found) the receive interrupt is disabled to put the device in ignore mode. The checksum is then compared against the last byte in the buffer. If it is okay, then a flag is set to indicate that canonical processing of the buffer data should occur. If not, the process is reset as the packet must have been malformed or corrupt.

Sending Data In Send Mode

Similarly to receiving, sending will be done by the interrupt handler when triggered by the send register empty flag.

Initially a piece of non ISR code will configure the interrupt and load the transmit register once. After that the handler will grab a new character from the main memory buffer and deal with it each time it is triggered. In the case of a byte that is not one of the special three it will just be sent. In the case of a byte requiring escaping it will instead send an escape byte, and set the flag indicating that the next byte should be escaped before being sent. When the next interrupt occurs the byte will be converted and sent. As with the receive functionality, the checksum will be calculated on the fly with each byte sent. When the end of the packet is reached the checksum will be sent or escaped and sent if required. The last interrupt will send the stop byte and disable the interrupt enable bit on the send register.

Processing Of A Received Packet

Yet to be implemented.

Generation Of A Packet To Transmit

Yet to be implemented.

Preparation Of A Data Log Packet While Idle

Yet to be implemented.