Description of MT Communication Protocol

The MT Protocol is a simple handshaking communication protocol between 2 parties (e.g. Master and Slave). The original design idea of this protocol was to overcome the shortage of serial ports on microcontrollers and enable them to use any 3 available I/O pins as an extra communication channel.

There are two variants of this protocol, one provides a timeout capability to avoid waiting indefinitely if the communication link is lost and the other has no timeout capability.

In order to understand the working mechanism of this protocol, let's define 3 I/O pins or lines as follows:

```
#define MASTER  1    // S1, I/O pin 1
#define SLAVE   2    // S2, I/O pin 2
#define DATA    3    // S3, I/O pin 3
```

The MASTER line of party A connects to the MASTER line of party B and the same connections apply to SLAVE and DATA lines. The DATA line is bi-directional, i.e. it is used to send and receive data. A single byte (i.e. 8 bits) is the minimum amount of data unit that can be sent at a time.

The following describes the steps required when the Master sends data to the Slave.

1. Initially, both parties set all 3 lines HIGH.
2. Slave waits for the MASTER line to go LOW.
3. Master pulls the MASTER line LOW and waits for Slave to acknowledge.
4. Slave acknowledges by pulling the SLAVE line LOW.
5. Masters puts the first bit (LSB, ie. rightmost bit) of the data byte on the DATA line by setting it HIGH or LOW accordingly, then sets the MASTER line HIGH and waits for the Slave to acknowledge.
6. Slave retrieves the bit and acknowledges by pulling the SLAVE line HIGH.
7. Master puts the second bit of the data byte on the DATA line by setting it HIGH or LOW accordingly, then sets then MASTER line LOW and waits for the Slave to acknowledge.
8. Slave retrieves the bit and acknowledges by pulling the SLAVE line LOW.
9. Repeat step 5-8 for the remaining 6 bits.
10. Both parties reset all the 3 lines HIGH.

NOTE: Using the MT protocol, communication between two parties can easily be achieved, even if the MCUs are different types and/or are running at different speeds. To avoid 'hanging', a timeout (10ms) is placed at those steps that wait for acknowledgements.

The **MT Serial LCD** module is implemented with the timeout capability, whereas the **MT Motor Controller** is implemented without the timeout capability. This is because motors require some timing intervals to turn which are much longer than the timeout period.

Two sample TinyC implementations of this communication protocol are demonstrated in "C:\MT\App\mtp.prg" and "C:\MT\App\mtrl.prg".