



POCKET MCA

MCA8000A

MCA8000A

USB Serial Adapter Protocol

Developer Guide

REVISION 1.42

October 2008

1.	MCA8000A USB Serial Adapter Protocol Summary	2
1.1	Host Computer Interface Improvements	2
1.2	MCA8000A Data Transfer Protocol as a Reference	2
2.	MCA8000A Data Transfer Protocol Update Summary.....	3
2.1	PC / MCA Control	3
2.1.1	SEND / RECEIVE Mode	3
2.1.2	MCA Response to SEND / RECEIVE Mode.....	3
2.1.3	New Data Transfer Commands.....	3
3.	Data Exchange Protocol Update Details	4
3.1	New Data Transfer Commands	4
3.2	New Command Codes	4
3.3	Start Channel	4
3.4	Words Requested	4
3.5	Message CheckSum.....	5
3.6	New Data Transfer Command Format Bit Details	5
3.7	SEND DATA BYTES AND CHECK SUM COMMANDS	6
3.7.1	Send Data Lower Word (Command Code = 40h).....	6
3.7.2	Send Data Upper Word (Command Code = 42h).....	6
3.8	SEND DATA, GROUP AND S/N COMMANDS	7
3.8.1	Send Data Group Lower Word (Command Code = 50h).....	7
3.8.2	Send Data Group Upper Word (Command Code = 52h).....	7
3.9	SEND MODE	8
3.10	RECEIVE MODE	8
3.11	New Data Transfer Command Function Example	9
3.12	Detecting MCA8000A Old and New Data Transfer Protocols	10
3.13	Using the Old Send Data Commands	11
4.	Data Exchange Protocol Unchanged List.....	12

1. MCA8000A USB Serial Adapter Protocol Summary

1.1 Host Computer Interface Improvements

MCA8000A USB Serial Adapter Protocol improves serial communication through USB serial port adapters. This is done by simplifying the serial transfer protocol. The serial transfer protocol has been simplified from a four line proprietary toggle and handshake control to a simple RTS/CTS flow control.

1.2 MCA8000A Data Transfer Protocol as a Reference

The MCA8000A USB Serial Adapter Protocol builds on the MCA8000A Data Transfer Protocol. The MCA8000A Data Transfer Protocol should be used as a main reference. This document describes the updates and changes to the previous protocol.

2. MCA8000A Data Transfer Protocol Update Summary

2.1 PC / MCA Control

The MCA Data Transfer Protocol has been simplified. All MCA DTR (PC DSR) toggles have been replaced by RTS (PC CTS) toggles. Control signals are no longer toggled on a byte-by-byte basis. PC DTR wake-up and baud rate setting functions remain unchanged. These changes result in a significant savings in serial control switching overhead. This shifts the burden of data transfer control to the MCA. The number of bytes transmitted must be controlled and control signal latency must be compensated for. USB serial port adapter control signal latency can be quite large (> 1 millisecond). To control the number of bytes transmitted four new messages have been introduced (See New Data Transfer Commands).

2.1.1 SEND / RECEIVE Mode

The PC port output signal RTS is used by the PC to indicate the direction of the data transfer. RTS control functions remain unchanged. The MCA constantly monitors the PC RTS line and responds accordingly. The action of the PC depends on the selected mode: SEND or RECEIVE.

- RTS = HIGH indicates data transfer from PC to MCA (SEND mode)
- RTS = LOW sets the data transfer direction from MCA to the PC (RECEIVE mode).

2.1.2 MCA Response to SEND / RECEIVE Mode

The MCA response to SEND / RECEIVE Mode has been changed. The MCA now responds with a RTS toggle (PC CTS) indicating the MCA is ready to receive a command. (See **SEND MODE**.) The MCA no longer toggles a control signal after each byte is received.

2.1.3 New Data Transfer Commands

Four new data transfer commands request data blocks of from 1 to 1024 words. See MCA8000A Commands for new command details. The new data transfer commands are divided into two groups based on the status message requested.

2.1.3.1 SEND DATA BYTES AND CHECK SUM COMMANDS

These commands send a specified number of data words with a checksum status.

- Send Data Lower Word (40h)
- Send Data Upper Word (42h)

2.1.3.2 SEND DATA, GROUP AND S/N COMMANDS

These commands send a specified number of data words with a group and serial number status.

- Send Data Group Lower Word (50h)
- Send Data Group Upper Word (52h)

3. Data Exchange Protocol Update Details

3.1 New Data Transfer Commands

Four new data transfer commands request data blocks of from 1 to 1024 words. The new data transfer commands are divided into two groups based on the status message requested. Before sending new data transfer commands the BAUD RATE and current group must be set.

3.2 New Command Codes

The new data transfer command codes correspond to the previous Send Data Commands. The new data transfer command code is calculated by adding the old command code, the lower/upper word offset, and 40h.

For example to send data with checksum lower word:

- SEND DATA AND CHECK SUM COMMAND Command Code = 0
- Lower/upper word offset = 0 for transfers of the lower word
- 40h is the message modifier to indicate this is a new command
- This is Send Data Lower Word (00h + 00h + 40h = 40h)

Table of the new command codes:

New Command Code	N7	N6	N5	N4	N3	N2	N1	N0
Send Data Lower Word (40h)	0	1	0	0	0	0	0	0
Send Data Upper Word (42h)	0	1	0	0	0	0	1	0
Send Data Group Lower Word (50h)	0	1	0	1	0	0	0	0
Send Data Group Upper Word (52h)	0	1	0	1	0	0	1	0

3.3 Start Channel

The start channel is the the first channel of the current group to be transmitted from the MCA. The channel values start at 0. The start channel range is from 0 to 16383 (3FFFh).

Start Channel	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0
---------------	-----	-----	-----	-----	----	----	----	----	----	----	----	----	----	----

3.4 Words Requested

The words requested indicate the number of lower/upper data words to be transmitted minus one. One is subtracted from the words requested to save on message bit of space allowing the required information to be passed in one command. Each data word is two bytes. The range is 1-1024 words, therefore the words requested value is from 0 to 1023 (3FFh).

Words Requested	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
-----------------	----	----	----	----	----	----	----	----	----	----

3.5 Message CheckSum

Check sum byte, the sum of bytes 0 to 3, MOD 256.

Checksum	S7	S6	S5	S4	S3	S2	S1	S0
----------	----	----	----	----	----	----	----	----

3.6 New Data Transfer Command Format Bit Details

Each command is sent as a packet of 5 bytes. The command bytes are defined as follows:

- Byte 0: New Command code, this indicates the specific action that MCA has to take.
- Byte 1: Start Channel (Bits C5-C0), Words Requested (Bits B1-B0)
- Byte 2: Start Channel (Bits C13-C6)
- Byte 3: Words Requested (Bits B9-B2)
- Byte 4: Check sum byte, the sum of bytes 0 to 3, MOD 256

Table of the new command bytes:

Byte	D7	D6	D5	D4	D3	D2	D1	D0	Description
0	N7	N6	N5	N4	N3	N2	N1	N0	New Command Code
1	C5	C4	C3	C2	C1	C0	B1	B0	Start Channel (C5-C0), Words Requested (B1-B0)
2	C13	C12	C11	C10	C9	C8	C7	C6	Start Channel (C13-C6)
3	B9	B8	B7	B6	B5	B4	B3	B2	Words Requested (B9-B2)
4	S7	S6	S5	S4	S3	S2	S1	S0	Checksum

3.7 SEND DATA BYTES AND CHECK SUM COMMANDS

These commands send a specified number of data words with a checksum status.

3.7.1 Send Data Lower Word (Command Code = 40h)

This command sends a specified number of lower data words with a checksum status.

Byte	D7	D6	D5	D4	D3	D2	D1	D0	Description
0	0	1	0	0	0	0	0	0	New Command Code = 40h
1	C5	C4	C3	C2	C1	C0	B1	B0	Start Channel (C5-C0), Words Requested (B1-B0)
2	C13	C12	C11	C10	C9	C8	C7	C6	Start Channel (C13-C6)
3	B9	B8	B7	B6	B5	B4	B3	B2	Words Requested (B9-B2)
4	S7	S6	S5	S4	S3	S2	S1	S0	Checksum

When this command is issued the received **DataChkSum** is the sum MOD 2^{16} of all bytes of the channel data that were transmitted from the MCA to the computer at the last data exchange preceding the transmission of the status structure. This variable may be used as a check sum of the transmitted channel data.

3.7.2 Send Data Upper Word (Command Code = 42h)

This command sends a specified number of upper data words with a checksum status.

Byte	D7	D6	D5	D4	D3	D2	D1	D0	Description
0	0	1	0	0	0	0	2	0	New Command Code = 42h
1	C5	C4	C3	C2	C1	C0	B1	B0	Start Channel (C5-C0), Words Requested (B1-B0)
2	C13	C12	C11	C10	C9	C8	C7	C6	Start Channel (C13-C6)
3	B9	B8	B7	B6	B5	B4	B3	B2	Words Requested (B9-B2)
4	S7	S6	S5	S4	S3	S2	S1	S0	Checksum

When this command is issued the received **DataChkSum** is the sum MOD 2^{16} of all bytes of the channel data that were transmitted from the MCA to the computer at the last data exchange preceding the transmission of the status structure. This variable may be used as a check sum of the transmitted channel data.

3.8 SEND DATA, GROUP AND S/N COMMANDS

These commands send a specified number of data words with a group and serial number status.

3.8.1 Send Data Group Lower Word (Command Code = 50h)

This command sends a specified number of lower data words with a group and serial number status.

Byte	D7	D6	D5	D4	D3	D2	D1	D0	Description
0	0	1	0	1	0	0	0	0	New Command Code = 50h
1	C5	C4	C3	C2	C1	C0	B1	B0	Start Channel (C5-C0), Words Requested (B1-B0)
2	C13	C12	C11	C10	C9	C8	C7	C6	Start Channel (C13-C6)
3	B9	B8	B7	B6	B5	B4	B3	B2	Words Requested (B9-B2)
4	S7	S6	S5	S4	S3	S2	S1	S0	Checksum

When this command is issued, the received **DataChkSum_0** is the current group number (byte). **DataChkSum_2** and **DataChkSum_3** form a unsigned integer that represent the serial number of the MCA devices connected to the computer.

3.8.2 Send Data Group Upper Word (Command Code = 52h)

This command sends a specified number of upper data words with a group and serial number status.

Byte	D7	D6	D5	D4	D3	D2	D1	D0	Description
0	0	1	0	1	0	0	1	0	New Command Code = 52h
1	C5	C4	C3	C2	C1	C0	B1	B0	Start Channel (C5-C0), Words Requested (B1-B0)
2	C13	C12	C11	C10	C9	C8	C7	C6	Start Channel (C13-C6)
3	B9	B8	B7	B6	B5	B4	B3	B2	Words Requested (B9-B2)
4	S7	S6	S5	S4	S3	S2	S1	S0	Checksum

When this command is issued, the received **DataChkSum_0** is the current group number (byte). **DataChkSum_2** and **DataChkSum_3** form a unsigned integer that represent the serial number of the MCA devices connected to the computer.

3.9 SEND MODE

In this mode PC sends commands to the MCA. Each command sent to MCA consists of five bytes (see earlier section). The sequence to send a command is as follows:

1. Read and save the state of the PC CTS signal (PC CTS bit of the modem status register MSR of the UART). Assume CTS is stored as variable OLDCTS.
2. Set PC RTS signal high (PC RTS bit of the modem control register MCR of the serial port).
3. Point to the first byte of the command sequence.
4. Reset and start a Time-Out Timer (Watch Dog). Allow 110 to 165 ms (2 to 3 clock ticks on IBM compatible PC) for sending a single byte.
5. Check if Transmit Holding Register is empty. If not wait until it becomes empty or exit with time-out error (10).
6. Check if the PC CTS signal has been complemented. If PC CTS is different than OLDCTS. Then send the command sequence.
7. If all of the command bytes have been sent successfully, read (dummy) the Receive Buffer of the UART and then clear PC RTS. Allow 100 to 200 μ s or more before sending next command.
8. Commands that prepare the MCA for sending data (PC enters RECEIVE MODE):
 - a. SEND DATA COMMAND (00h)
 - b. SEND DATA, GROUP AND S/N COMMAND (10h)
 - c. GET START STAMP COMMAND (30h)
 - d. Send Data Lower Word (40h)
 - e. Send Data Upper Word (42h)
 - f. Send Data Group Lower Word (50h)
 - g. Send Data Group Upper Word (52h)
9. If time-out expires set PC RTS = LOW. Allow 100 to 200 μ s before sending next command.

3.10 RECEIVE MODE

The PC enters RECEIVE mode after issuing any of the following commands:

- SEND DATA COMMAND (00h)
 - SEND DATA, GROUP AND S/N COMMAND (10h)
 - GET START STAMP COMMAND (30h)
 - Send Data Lower Word (40h)
 - Send Data Upper Word (42h)
 - Send Data Group Lower Word (50h)
 - Send Data Group Upper Word (52h)
1. Flush the receive buffer at the earliest opportunity.
 2. After the command is sent, allow enough time for the receive data to arrive.
 3. If the receive mode command was a new data transfer command verify the checksum.

3.11 New Data Transfer Command Function Example

This function converts the SEND DATA COMMAND group and count into a new data transfer lower or upper command.

```
static int SendCommandDataEx(WORD group, WORD count)
{
    CommandType command;
    int stat;
    BYTE isUpper;
    BYTE groupH;
    BYTE groupL;
    BYTE countH;
    BYTE countL;
    isUpper = group & 0x02;
    countH = (count & 0x300) >> 8;
    countL = count & 0xFF;
    groupH = (group & 0xFF00) >> 8;
    groupL = (group & 0xFC) + countH;
    command.code = CMD_GET_DATA + 0x40 + isUpper;
    //command.u.data.group = group;
    command.u.b.b1 = groupL;
    command.u.b.b2 = groupH;
    command.u.b.b3 = countL;
    //command.u.data.baudRateDivisor = s_baudRateDivisor;
    stat = SendCommand(&command);
    return stat;
} // End of SendCommandData
```

3.12 Detecting MCA8000A Old and New Data Transfer Protocols

The **MCA8000A Data Transfer Protocol** in use can be determined by monitoring the RS232 control signals during power-up or device connection. The old data protocol toggles DSR on power-up. The new data protocol toggles CTS on power-up. The following example demonstrates MCA data transfer protocol detection. The PMCA DLL uses this detection method (search for: request processing "case OP_POWER_UP").

```
bool isUSBMCA;
static int ExamplePowerUp() {
    #define POWERUP_TICK 500.0 // 0.5 milliseconds x2 (1kHz)
    int i;
    bool initCTS, newCTS, isUSBMCACTS;
    bool initDSR, newDSR, isUSBMCADSR;
    if (s_connected)
        ExecDisconnect();
    stat = OpenComPort(request->p.powerUpPort);
    if (!stat)
    {
        initCTS = GetCts();
        initDSR = GetDsr();
        SetRts();
        for (i = 200; --i;) // 200 milliseconds
        {
            SetDtr();
            s_mTime.Wait(POWERUP_TICK);
            ResetDtr();
            s_mTime.Wait(POWERUP_TICK);
        }
        Sleep(3800);
        newCTS = GetCts();
        newDSR = GetDsr();
        if (initCTS != newCTS) {
            isUSBMCACTS = true;
        } else {
            isUSBMCACTS = false;
        }
        if (initDSR != newDSR) {
            isUSBMCADSR = false;
        } else {
            isUSBMCADSR = true;
        }
        if (isUSBMCACTS == isUSBMCADSR) {
            isUSBMCA = isUSBMCACTS;
        } else {
            isUSBMCA = false; // error can't identify
        }
        ResetRts();
        CloseComPort();
    }
} // End of ExamplePowerUp
```

3.13 Using the Old Send Data Commands

The old commands function normally with the old MCA8000A.

The old send data commands required the PC RS232 signal toggles to determine data block size and data flow control. To allow for efficient USB RS232 serial adapter operation, most PC RS232 signal toggle functions had to be removed. Therefore some the two old protocol "send data" commands now have some operational restrictions. If the old send data commands are used, the status must be requested using control signal toggles. This stops the MCA transmitter after data transfer. A code example follows (from the PMCA DLL). Also the checksum of the old send data commands should be ignored.

To test using the old send data commands with the new MCA8000A replace the SendCommandDataEx function calls in the PMCA DLL to SendCommandData calls and comment out the checksum test in the ReceiveDataWords function.

```
static int PromptPmcaForStatus(void)
{
    int stat;
    BOOL success;
    DWORD modemStatus;
    // First clear RTS to make sure that PMCA is in the send state
    ResetRts();
    s_mTime.Wait(200.0);
    stat = GetModemStatus(&modemStatus);
    if (!stat)
    {
        // Switch PMCA to the receive state
        SetRts();
        if (isUSBMCA) {
            // Wait until PMCA goes into the receive state (CTS is toggled)
            stat = WaitForCtsToggle(&modemStatus);
            success = PurgeComm(s_comHandle, PURGE_RXCLEAR);
        } else {
            // Wait until PMCA goes into the receive state (DSR is toggled)
            stat = WaitForDsrToggle(&modemStatus);
        }
        // Switch PMCA to the send status state
        ResetRts();
        s_mTime.Wait(200.0);    // Wait few hundred microseconds
        // Now PMCA is going to send status when requested to send
    }
    return stat;
} // End of PromptPmcaForStatus
```

4. Data Exchange Protocol Unchanged List

Most commands and functions remain unchanged. Below is a list of unchanged protocol items. See MCA8000A Data Exchange Protocol Rev 1.4.1 for details.

- 1 MCA8000A Data Format
 - 1.1 Spectral Data Structure
 - 1.2 MCA Status Data Structure
 - 1.3 MCA8000A Start Stamp Structure
- 2 MCA8000A Commands
 - 2.1 Command Format
 - 2.2 SEND DATA AND CHECK SUM COMMAND
 - 2.3 SEND DATA, GROUP AND S/N COMMAND
 - 2.4 GET START STAMP COMMAND
 - 2.5 SET START DATE COMMAND
 - 2.6 SET START TIME COMMAND
 - 2.7 SET GROUP COMMAND
 - 2.8 SET MCA LOCK COMMAND
 - 2.9 CONTROL COMMAND
 - 2.10 PRESET TIME COMMAND
 - 2.11 DELETE COMMAND
- 3 Data Exchange Sequence
 - 3.1 Remote Power Control
 - 3.2 BAUD RATE SETTING MODE