



# MicroBuffer User Guide

Email: [info@scannex.co.uk](mailto:info@scannex.co.uk)  
Web: [www.scannex.co.uk](http://www.scannex.co.uk)

## Document Release Status:

<b>Date</b>	<b>Document Ref</b>	<b>Issue</b>	<b>Status</b>
01/04/1999	FN01-0003	01	Initial Release
23/04/1999	FN01-0003	02	X-ON / X-OFF added
26/08/1999	FN01-0003	03	Cable specs added
27/04/2000	FN01-0003	04	Figure 1 corrected Cable pin 2/3 arrows corrected
01/06/2001	FN01-0003	05	Added "Data Option" information and "Remote Mode"
22/01/2002	FN01-0003	06	Copyright and minor text changes
19/12/2003	FN01-0003	07	Mode & approvals changes, page layout change

Copyright © UK 2000-3 Scannex Electronics Ltd. All rights reserved.

The information contained herein is the property of Scannex Electronics Ltd and is supplied without liability for errors or omissions and without having any connection with or forming any part of any contract relating to the supply or use of any Scannex Electronics products. No part may be reproduced, used or disclosed, except as authorised by contract or other written permission. The copyright and foregoing restriction on reproduction, use and disclosure extend to all media in which this information may be embodied.

**Contents:**

<b>Introduction</b>	<b>1</b>
<b>Installation</b>	<b>1</b>
<b>Procedure</b>	<b>1</b>
<b>General Information</b>	<b>1</b>
Power requirements:	1
Data Source:	2
PC Communications:	2
<b>Operation</b>	<b>2</b>
<b>Control Signals (computer Port)</b>	<b>2</b>
DSR / CTS	2
DCD	2
DTR / RTS	3
<b>Hardware only Flow Mode</b>	<b>3</b>
<b>Hardware+Software Flow Mode</b>	<b>3</b>
<b>Mode Switching (Guard Sequence)</b>	<b>3</b>
<b>Packet Mode Operation</b>	<b>3</b>
<b>Packet Mode Command Set</b>	<b>4</b>
<b>Commands available from firmware 1.16</b>	<b>8</b>
<b>Specification</b>	<b>9</b>
<b>Approvals</b>	<b>10</b>
<b>CE Approval:</b>	<b>10</b>
<b>FCC Approval:</b>	<b>10</b>
<b>Configuration Utility</b>	<b>11</b>
<b>Appendix A- Autobauding</b>	<b>13</b>
Error detection:	13
Autobauding Sequence:	13
Comms Formats:	13
<b>Appendix B- Packet Command Summary</b>	<b>14</b>
<b>Appendix C- Packet Mode Flow Chart</b>	<b>15</b>
<b>Appendix D- Cable Definitions</b>	<b>16</b>
<b>MicroBuffer to Computer</b>	<b>16</b>
<b>MicroBuffer to Data Source</b>	<b>16</b>

## INTRODUCTION

The MicroBuffer is an intelligent buffer that automatically detects the communications format used by the Data Source, and whether it is configured as a DCE or DTE. It provides 1Mbyte of non-volatile data storage and presents a normalised interface to the PC.

The MicroBuffer is line-powered and can be sited at either the PC or Data Source end of the link, providing PC crash resilience and allowing the PC to be powered down, or upgraded, without data loss.

It operates in two modes, Flow Control or Packet Mode.

The key features are:

### **Ultra low power**

The MicroBuffer does not require external power supplies. It can even get enough power from the line it is logging.

### **Non-volatile data storage**

High capacity (1Mbyte) with 10 year data hold up.

### **Sophisticated autobauding**

Simplifies installation and will maintain synchronisation with the data even if output speed or format changes.

### **Auto pin detection**

The data will be logged from either pin 2 or pin 3 – standard RS232 cables can generally be used.

### **Failsafe logging**

Allows the PC to be powered down, or upgraded, without data loss.

### **High speed download**

The PC obtains information from the MicroBuffer at up 3kbytes per second via a COM port.

### **Dual mode download**

Either checksummed packet mode or simple Flow mode. Flow mode is controlled by DTR/RTS control signals with optional X-ON / X-OFF superimposed on top.

### **Data Source port**

RS232: 300 – 19200 baud with 7,8 bit word and odd, even, or no parity.  
Option to store only 7-bit ASCII or full 8-bit data.

### **Computer port**

RS232: 300 – 19200, 57600 baud with 8 bit word, no parity.

## INSTALLATION

### ***Procedure***

- Connect the MicroBuffer “Data Source” to the V24 port to be logged using a straight through cable.
- The red status LED should light solidly for about 5 seconds and pulse slowly. While data is actively being received, the LED should pulse faster (½ sec ON, ½ sec OFF). It will then pulse at a slower rate (½ sec ON, 1½ sec OFF) to indicate that data is stored.
- Connect the MicroBuffer “Computer” to the PC COM Port using the short 9way – 9way serial cable. Note: use the 9-25 way adaptor if the PC COM Port is fitted with a 25-way connector.
- The MicroBuffer can now communicate with the PC

### ***General Information***

#### **Power requirements:**

The MicroBuffer draws its power from any lines connected to its Computer or Data Source ports (see fig 1). It requires a maximum of 5ma from the connected data/control lines without them dropping below  $\pm 4.7V$ .

The MicroBuffer can operate and log data without the Computer port connected, and can usually gain enough power with just the ground and data line of the Data Source connected.

**Data Source:**

The MicroBuffer's Data Source will autodetect pins 2 and 3 for data reception, and will transmit on the opposite pin. It is assumed that the MicroBuffer is the only device connected to the Data Source – if parallel connections are required, then only connect **one** of the two pins of the MicroBuffer to the Data Source.

There is no handshaking at the Data Source end. Pins 7 and 8 are linked together (CTS + RTS), and pins 4 and 6 (DSR + DTR) are linked together. Both pairs are given a 'soft-drive' to try and assert the lines at the Data Source, if needed.

If communication errors are detected, the MicroBuffer will autobaud to locate the baud rate and parity setting. While autobauding, the DCD line back to the PC will be *unasserted*.

**PC Communications:**

The PC always communicates with a comms format setting of 8 bit data, no parity, 1 stop bit. There are two handshake lines recognised by the MicroBuffer, and three lines sent back.

**OPERATION**

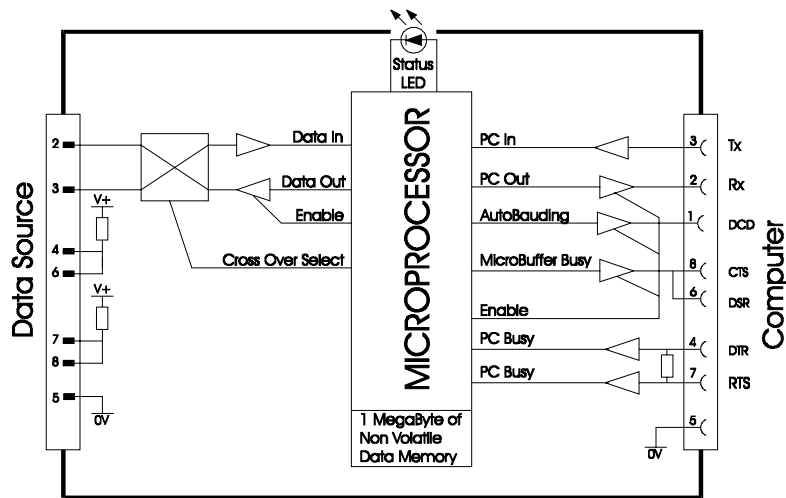


Fig. 1 MicroBuffer Schematic

There are two primary modes of operation that can be set as default, and dynamically switched between: Flow Mode and Packet Mode.

Flow mode allows data transmission between the PC and Data Source with differing communication rates and formats to be controlled by Hardware only, or Hardware+Software flow control.

Packet mode allows a more secure, checksummed downloading of data, with packet sequencing and re-transmit on error. A simple command set allows the MicroBuffer to be interrogated and configured.

**Control Signals (computer Port)**

**DSR / CTS**

These handshake lines are output from the MicroBuffer, and are tied together. When *asserted* they show the MicroBuffer is ready to receive data. When *unasserted*, the PC **must** stop sending within 20 characters – otherwise an overrun error will be flagged.

**DCD**

When *unasserted* this shows that the MicroBuffer is currently autobauding. If the PC transmits any data to the MicroBuffer, it will immediately stop the autobauding process.

*Note: If Autobauding is aborted then the Data Source comms format may not be correct.*

## DTR / RTS

The MicroBuffer monitors both of these lines. If only one line is physically connected, it will operate both lines at the MicroBuffer.

- If both lines are *asserted*, the MicroBuffer assumes the PC is ready to accept data.
- If either line is *unasserted*, the MicroBuffer will not send data back to the PC.
- If both lines are *unasserted*, the MicroBuffer will shut down its PC interface to conserve power and will appear busy to the PC.
- If either line is *unasserted* for more than 20 seconds, the MicroBuffer reverts to its default mode.

*Note: data is sent to the PC in 16 byte chunks (in both modes), and flow control is effectively checked between 'chunks', so up to 16 bytes may be output after the lines are unasserted.*

## Hardware only Flow Mode

Data is passed transparently from the Computer to the Data Source, and from the Data Source to the Computer, using the control lines (described above) for flow control. There are specific timings in Flow Mode, in addition to the above, that govern the flow of data back to the PC:

- If DTR or RTS are *unasserted* for less than 5 seconds, then data transmission to the PC will be resumed after ½ second of the lines being *asserted*.
- If DTR or RTS are *unasserted* for more than 5 seconds, it will require the lines to be *asserted* for 5 seconds before transmission to the PC is resumed.

## Hardware+Software Flow Mode

This mode superimposes X-ON / X-OFF software flow control on top of the Hardware Flow control described above, but **DOES NOT** pass any data back to the Data source.

The MicroBuffer always powers up in the X-OFF state, and will revert to this state after the 20 second timeout or if both handshake lines are *unasserted*.

X-OFF is equivalent to CTRL-S (0x13)

X-ON is equivalent to CTRL-Q (0x11)

## Mode Switching (Guard Sequence)

The MicroBuffer can be set into Packet mode from Flow Mode as follows:

- Assert both DTR and RTS lines from the PC
- Pause at least 1 second.
- Send three characters 0x84 (132 decimal), no more than 1 second apart.  
You can use Alt-0132 on the numeric keypad of a Windows machine to send the characters. *Note: these characters are transmitted through to the Data Source if in Hardware Flow control.*
- Pause at least 1 second.
- The MicroBuffer should send back an ACK (0x06) character.
- If the MicroBuffer does not answer within two seconds (because it is autobauding), then send the 0xFF character, and the ACK should pop out.
- Keep the DTR and RTS lines *asserted* to communicate in packet mode. You may *unassert* them temporarily, but keeping either or both *unasserted* for more than 20 seconds will resume the default mode.

## Packet Mode Operation

There are many commands available in packet mode. When first entering the mode, most of the commands are disabled. You must issue the 'enable commands' command first.

Each successful command byte is acknowledged by the MicroBuffer with ACK (0x06).

Unrecognised commands will receive a CAN (0x18, 24dec).

Commands that are not enabled, or allowed, will receive a NAK (0x15, 21dec).

### **Packet Mode Command Set**

#### ***Are you there? – 0x80 (128 dec)***

---

*This command will work whether commands are enabled or disabled.*

Reply: "SCANNEX 1.16" (or other version number)

Note: Do not assume the length. Rather, keep reading until there is a significant pause in the data (e.g. ½ second).

#### ***Enable commands – 0x90 (144 dec)***

---

Send this before issuing other commands.

Reply: ACK

#### ***Disable commands – 0x41 (65 dec)***

---

This will 'lock' the MicroBuffer from accidental commands.

Reply: ACK

#### ***Get Serial Number – 0x53 (83 dec)***

---

Reply: "M01234"

Note: Do not assume there are exactly 5 bytes. Rather, keep reading until there is a significant pause in the data (e.g. ½ second).

#### ***Get Number of Bytes Used – 0x40 (64 dec)***

---

Reply: ACK + B1 + B2 + B3

Where number of bytes is  $B1 * 65536 + B2 * 256 + B3$

#### ***Delete All Data – 0x52 + 0xA5 (82 + 165 dec)***

---

This command simply resets the pointers so that there is no data in the MicroBuffer.

Reply: ACK + ACK

(Each byte is ack'd. If the second byte is not 0xA5, then the reply will be ACK + NAK)

#### ***Force Pointers to "All Data" – 0x54 + 0xA5 (84 + 165 dec)***

---

This command will set the internal pointers to the beginning of flash, and end of flash – thus allowing a full download of any valid data that may have been lost. The data must be downloaded after this. Issuing a "Get Number of Bytes Used" command following this may not match the actual bytes downloaded.

Note: Used only for diagnostic purposes

Reply: ACK + ACK

#### ***Set default mode to Packet – 0x4F (79 dec)***

---

Reply: ACK

The MicroBuffer will remain in packet mode.

#### ***Set default mode to Hardware Flow – 0x4E (78 dec)***

---

Reply: ACK

The MicroBuffer will remain in packet mode until power down, or until the handshake lines stay unasserted for more than 20 seconds.

To enter flow mode, use the "Enter Flow Mode" command.

To re-enter packet mode, use the guard sequence.

#### ***Set default mode to Hardware+Software Flow – 0x5A (90 dec)***

---

Reply: ACK

The MicroBuffer will remain in packet mode until power down, or until the handshake lines stay *unasserted* for more than 20 seconds.

To enter flow mode, use the "Enter Flow Mode" command.

To re-enter packet mode, use the guard sequence.

***Enter Flow Mode – 0x55 (85 dec)***

---

Reply: ACK

The MicroBuffer will switch to Hardware Flow mode (or Hardware+Software Flow mode if this is the default). After the ACK, the MicroBuffer will transmit all further characters to the Data Source. If DTR and RTS are *asserted*, data is returned to the PC after an initial delay of 5 seconds (an X-ON must also be issued if Hardware+Software Flow mode is set).

***Force Autobaud – 0x56 (86 dec)***

---

Reply: ACK

The DCD line will be *unasserted* and the LED lit until the baud detection phase is complete. When the autobauding process is finished, the DCD line will be *asserted* again, and the Data Source parameters can be read with the "Get Data Source Format" command.

**Note:** If the Data Source port is disconnected, or an input character is received on the Computer port, the autobauding sequence is terminated and the communication format may not be correct. Autobauding will be resumed on further errors (due to baud rate differences etc).

**Get Data Source Format – 0x4D (77 dec)**

---

Reply: ACK + DSformat

The DSformat is obtained in the following table:

	7N	7O	7E	8N/8M	8O	8E
19200	0x00 / 0	0x04 / 4	0x05 / 5	0x02 / 2	0x06 / 6	0x07 / 7
9600	0x08 / 8	0x0C / 12	0x0D / 13	0x0A / 10	0x0E / 14	0x0F / 15
4800	0x10 / 16	0x14 / 20	0x15 / 21	0x12 / 18	0x16 / 22	0x17 / 21
2400	0x18 / 24	0x1C / 28	0x1D / 29	0x1A / 26	0x1E / 30	0x1F / 31
1200	0x20 / 32	0x24 / 36	0x25 / 37	0x22 / 34	0x26 / 38	0x27 / 39
600	0x28 / 40	0x2C / 44	0x2D / 45	0x2A / 42	0x2E / 46	0x2F / 47
300	0x30 / 48	0x34 / 52	0x35 / 53	0x32 / 50	0x36 / 54	0x37 / 55

**Set Data Source Format – Dsformat + 0xA5 (Dsformat + 165 dec)**

---

Send byte from above table, followed by 0xA5 to set Data Source baud rate & parity.

Reply: ACK

**Set PC Baud Rate – Pcformat + 0xA5 (Pcformat + 165 dec)**

---

Send the PC baud rate from the table below, followed by 0xA5:

Baud rate	Byte
57600	0x4C / 76
19200	0x4B / 75
9600	0x4A / 74
4800	0x49 / 73
2400	0x48 / 72
1200	0x47 / 71
600	0x46 / 70
300	0x45 / 69

The ACK reply is sent at the *current* baud rate. Further data should be sent at the new baud rate.

**Get Status – 0x50 (80 dec)**

---

*This command will work whether commands are enabled or disabled.*

This will return an ACK, and a bit field as follows:

7	6	5	4	3	2	1	0
DS1	DS0	Overrun	Mode1	Mode0	undefined	Flash overflow	Reset

**Overrun:** Set if the PC has ignored the handshake lines (DSR/CTS) and sent too many bytes to the MicroBuffer.

**Flash overflow:** Set if data has been lost because data has not been downloaded from the MicroBuffer and more than 1Mb has been received.

**Reset:** Set to indicate that a power-up or reset has occurred.

Bits 0, 1 & 5 can be reset with the “Clear Status” command.

Bit definitions:

Mode1	Mode0	Operating Mode
0	0	Hardware
0	1	Hardware+Software
1	0	Packet
1	1	Not Used

DS1	DS0	Data Source State
0	0	Pins 2 and 3 open
0	1	Receiving on pin 3
1	0	Receiving on pin 2
1	1	Cabling Error

**Clear Status – 0x51 (81 dec)**

---

This will clear status bits 0,1,5.

Reply: ACK

**Get Packet – 0x42 (66 dec)**

---

Send this to get the first packet of a download sequence, or resend last packet again.

Note that as soon as this command is issued the MicroBuffer enters **linear mode** – that is, new data is lost if the buffer becomes full. To resume **circular mode** – i.e. old data is lost if the buffer becomes full – an “End Download” command needs to be issued.

Reply

ACK	SeqNo	DataSize (2 bytes)	DataBytes...	Checksum
-----	-------	-----------------------	--------------	----------

Where:

SeqNo: sequence number (0 – 0xFF). It starts at 0 on a fresh download.

DataSize: number of Data Bytes in the packet (2 byte- MSB first).

Data Bytes... maximum number of Data Bytes in a single packet is 256.

Checksum: the 8-bit addition of all bytes from SeqNo to the last Data Byte.

Eg (sending data “hello”)

06	00	00 05	68 65 6C 6C 6F	19
----	----	-------	----------------	----

To download all the buffer, keep downloading until the DataSize equals ‘0000’. The PC may stop downloading safely at any time by storing the packet, then issuing the “End Download” command, even if the last packet was not null.

**Get Next Packet – 0x44 (68 dec)**

---

This command deletes the last packet sent, and sends the next. The sequence number should increment (and wrap over 0xFF). The data format is the same as for “Get Packet” above.

**End Download – 0x43 (67 dec)**

---

This command deletes the last packet sent, and stops the download process. It also puts the MicroBuffer back into circular mode (so that old data is deleted if the buffer becomes full).

If the DTR line is *unasserted* during a packet transfer, data output will pause leaving the device in linear mode until an End Download command is received (even if the default mode is Flow Mode).

### **Commands available from firmware 1.16**

The following commands are available in firmware 1.16 (serial numbers M01300 and above).

#### **Store full 8-bit data – 0x5E (94 dec)**

---

This command enables storage of all 8 bits of incoming data – assuming the data format is an 8 bit data format.

Reply: ACK

#### **Store only 7-bit data – 0x5F (95 dec)**

---

This command strips the top bit from the incoming data source and stores only 7 bits of ASCII data. Do not use this when collecting from binary data sources.

Reply: ACK

#### **Query data option – 0x60 (96 dec)**

---

Queries the data option flag, indicating whether full 8-bit data is stored or only 7-bit ASCII data.

Reply: ACK mode

“mode” is either 0x00 for “8-bit”, or 0x10 (16 dec) for “7-bit”



## **APPROVALS**

### ***CE Approval:***

CE marked to class 'B' (EN55022, EN55024)

### ***FCC Approval:***

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

## CONFIGURATION UTILITY

MBSET.EXE is a DOS-based application that allows an engineer to configure and setup a MicroBuffer and can be used whether the MicroBuffer is set in Flow or Packet mode.

Note: MBSET will temporarily set Packet mode to interrogate the MicroBuffer and return it to its default mode on completion.

To find the command set, just type:

### MBSET

The response will be something like:

```
Scannex MicroBuffer Setter
(c) UK 1998 Scannex Electronics Ltd. All rights reserved
Version 1.20 19-Apr-99
```

Usage:

```
MBSET port baud action
```

```
where port is 1, 2, 3, 4 (the COMx port)
      baud is 300-19200, 57600
      action is F - find baud rate
                ? - show information
                C - configure
                R - reset status
                D - delete all data
                G - get all data to DOWNLOAD.RAW
```

```
e.g. MBSET 2 57600 ?
      MBSET 2 F ?
```

The commands are outlined below. Note that commands can be stacked up on the same command line, and are processed in order (eg. "**MBSET 2 F ?**"). However, the Delete command and the Get all data command are mutually exclusive – Get all data will take priority.

If the MicroBuffer is currently in Hardware Flow Mode, the Data Source will see the following hex bytes transmitted to it whenever MBSET is used:

```
0xFF 0xFF (pause) 0x84 0x84 0x84 (pause)
```

### ***F – Find MicroBuffer***

Typing "**MBSET 2 F**" will probe COM port 2 and attempt to locate an active MicroBuffer. The software will inform you of the current PC baud rate setting that you can use in subsequent communications (e.g. "**MBSET 2 57600 ? C**", though "**MBSET 2 F ? C**" will also work).

Note: Ensure that no other application is using the port.

### ***? – Query MicroBuffer Settings***

This command will show something like:

```
Scannex MicroBuffer Setter
(c) UK 1998-2001 Scannex Electronics Ltd. All rights reserved
Version 1.40 01-Jun-01
```

```
/-Status-----\
|                PC Baud : 57600
|                Data Source : 19200,8N
|                Data Option : 8-bit (binary)
|                ID : SCANNEX 1.16
|                Serial : M01234
|                Data Pin : 3
|                Status : [Reset]
|                Default : Packet mode
|                Used Bytes : 21
|-----\
```

where:

**Data Source** shows the baud rate and parity settings.

**Data Option** shows the 7-bit only or 8-bit mode (in 1.16+ firmware)

**ID** is the MicroBuffer version information.

**Serial** uniquely identifies the MicroBuffer.

**Data Pin** shows which pin is identified as Receive Data (ie 2, 3 or Open)

**Status** will show the following information:

[Reset]	Device has been reset,
[PC Overrun]	PC has ignored handshaking while sending
[Flash Overflow]	Some old data has been lost because more than 1Mb has been received.

(Also see the “**R**” command)

**Default** shows the default mode – either Packet, HW or HW+SW Flow. If the handshake line remains *unasserted* for more than 20 seconds, the default mode will be resumed, as will a power off/power up sequence.

#### **C - Configure MicroBuffer**

---

This command allows the default modes and PC baud rate to be changed.

Use the “+” and “-”, arrows, PgUp, PgDn, Home, End keys to scroll through available values when setting, and use the “**Enter**” key to accept each change. The “**Space**” key will call up the default.

When setting the comms format, you can also use the “**8**”, “**7**”, “**N**”, “**O**”, and “**E**” keys.

#### **D - Delete all data**

---

Simply wipes out the internal memory of the MicroBuffer. The software will tell you how many bytes have been deleted.

#### **R - Reset Status**

---

Resets the status bits as read out with the “?” command.

#### **G - Get all data**

---

This command will simply download all data in the MicroBuffer to a file “DOWNLOAD.RAW”. The data is appended to the file if it already exists.

No locking is employed, so be sure not to have the file open in another window or on another machine when executing this command.

## APPENDIX A- AUTOBAUDING

The 'Data Source' port of the MicroBuffer provides an automatic facility to lock on, and programme itself, to the communications format of the incoming RS-232 serial data.

### Error detection:

A format error will be identified if the Stop bit is not correctly received or if the Parity, if used, is incorrect.

A single byte error will not trigger the autobaud sequence. A certain number of errors in a fixed sliding window are required.

A programmed format of 8-bit data, No parity will not detect format errors if 7-bit data, Odd or Even parity, characters are received.

### Autobauding Sequence:

- 1) Sample a fixed number of transition periods to determine the baud rate. The LED will stay lit solidly during this phase and received data is lost.
- 2) Wait for a gap in data of 1½ character periods to synchronise to start of next byte.
- 3) Determine the data bit length. Requires 16 characters without format errors.
- 4) Check the collected data for parity, programme the comms format and store the data.

### Notes:

1. Step 1: typically takes 20 characters (depends on randomness of data). Remaining steps take 16 characters but **will not** start until the 1½ character gap occurs.
2. If the Data Source port is disconnected, or an input character is received on the Computer port, the autobauding sequence is terminated.
  - i. If no format information has been determined, it will be left at 19200,7,E.
  - ii. If the baud rate has been identified (eg 2400 baud) it will be left at 2400,7,E.
  - iii. If the MicroBuffer is turned off in this state, it will revert to the original stored format when power is restored.

### Comms Formats:

Baud Rate: 300, 600, 1200, 2400, 4800, 9600, 19200

Formats: 7 data, Odd, Even or No parity, 1 Stop bit

8 data, Odd, Even or No parity, 1 Stop bit

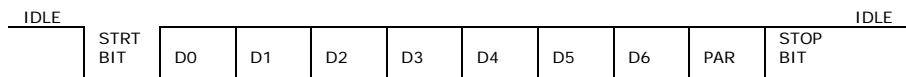


Fig. 2 Asynchronous Serial Data (7-bit data with parity)

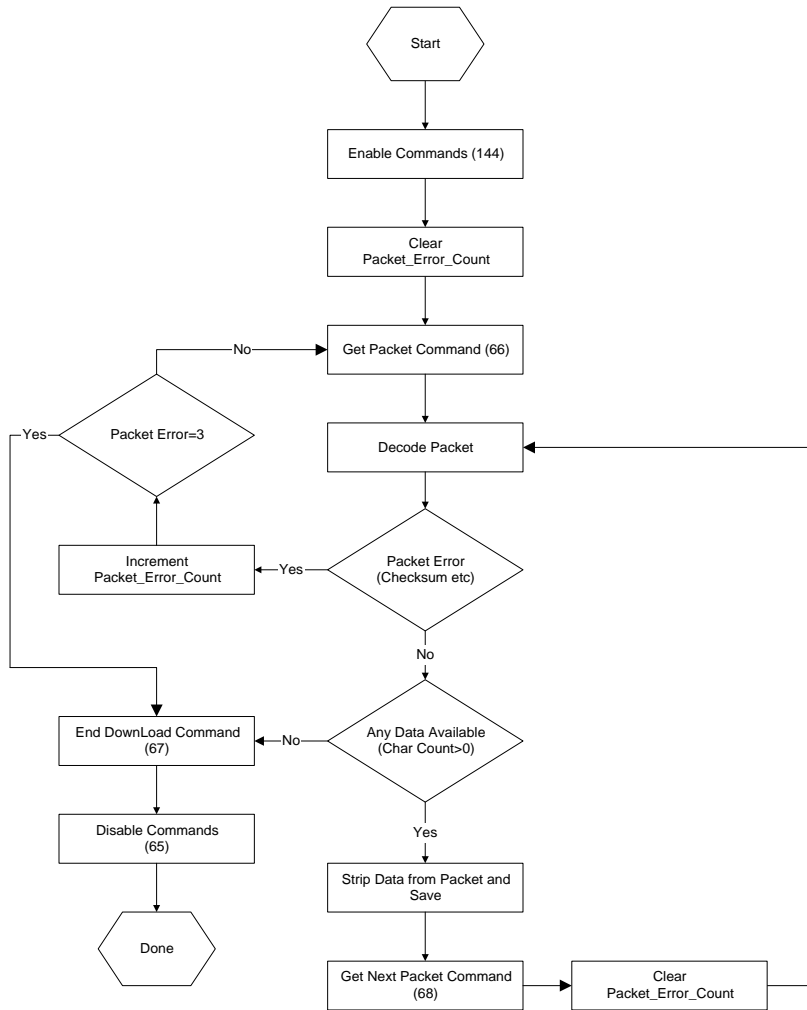
## APPENDIX B- PACKET COMMAND SUMMARY

Command	Decimal	Action	
0x40	64	Get Number of Bytes Used	
0x41	65	Disable Commands	
0x42	66	Get Packet	
0x43	67	End Download	
0x44	68	Get Next Packet	
0x4D	77	Get Data Source Format	
0x4E	78	Set Default Mode to HW Flow	
0x4F	79	Set Default Mode to Packet	
0x50	80	Get Status	*
0x51	81	Clear Status	
0x52 + 0xA5	82 + 165	Delete All Data	
0x53	83	Get Serial Number	
0x54 + 0xA5	84 + 165	Force Pointers to "All Data"	
0x55	85	Enter Flow Mode	
0x56	86	Force Autobauding	
0x5A	90	Set Default Mode to HW+SW Flow	
0x5E	94	Data option = 8-bits of data	
0x5F	95	Data option = 7-bits of data	
0x60	96	Query data option	
0x80	128	Are You There	*
0x90	144	Enable Commands	*
Dsformat + 0xA5	Dsformat + 165	Set Data Source Format	
Pcformat + 0xA5	Pcformat + 165	Set PC Baud Rate	

\* These commands are always enabled

## APPENDIX C- PACKET MODE FLOW CHART

The following flowchart is a guide to downloading from the MicroBuffer using Packet Mode commands:



## APPENDIX D- CABLE DEFINITIONS

All cables must be screened to the connector shell (can) to meet EMC requirements.

### *MicroBuffer to Computer*

MicroBuffer		Computer	
9-pin PLUG		9-pin SOCKET	25-pin SOCKET
1	→	1	8
2	→	2	3
3	←	3	2
4	←	4	20
5	---	5	7
6	→	6	6
7	←	7	4
8	→	8	5
Can	---	Can	Can

**Note:** A Standard serial extension cable configured 1:1 can be used

### *MicroBuffer to Data Source*

MicroBuffer	Data Source	
9-pin SOCKET	9-pin (25-pin)	
2	Tx / Rx	The MicroBuffer will auto-detect pins 2 or 3 for data transmit & receive.
3	Tx / Rx	
4		Linked together by the MicroBuffer and weakly asserted (not required by MicroBuffer).
6		
7		Linked together by the MicroBuffer and weakly asserted (not required by MicroBuffer).
8		
5		Signal Ground
Can	Can	Frame Ground

**Note:** A Standard serial extension cable configured 1:1 can generally be used