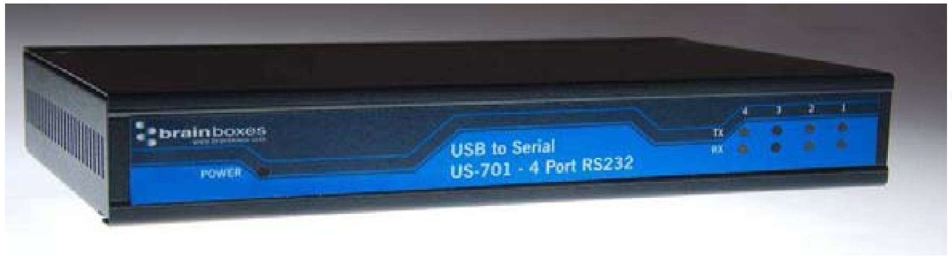
PCIe, USB and Ethernet Serial Devices

Eamonn Walsh
Technical Director
Brainboxes Ltd
Liverpool
Great Britain

25th
1984~2009
Anniversary



Contents

- Overview
- Background To Serial Communications
- Latency
- The Software Challenge
- PCI and PCI Express to Serial
- USB to Serial
- Ethernet to Serial
- Summary
- Recommendations

brainboxes
www.brainboxes.com

Award Winners

- European Manufacturer of the Year 2005

Elektra 05
European Electronics Industry Awards
WINNER
Manufacturer of the Year

- UK Manufacturing Institute: Champions of Best Practice and Small Business of the Year 2007

2007 BUSINESS Awards
THE Manufacturing INSTITUTE

THE Manufacturing INSTITUTE

- Merseyside Exporter of the Year 2005

MERSEYSIDE FAMILY BUSINESS AWARDS
WINNER
2005

My Data, Any Connection.

- Hardware Challenge as PC Platform Evolves:
- To deliver the data from the serial device over whatever physical connection is possible.
- Long gone are the days when we were limited by what ports we had on our individual computer.
  - As long as we have at least one means of getting our PC connected to a network we want access to our devices and their data.
- Connection Agnostic Interface
  - Serial data any time, on any device, over any connection, wired or wireless, local or networked
  - As serial data interface devices become part of the infrastructure- our job is to make the connection, configure and control our device using whatever access device we happen to have in our hands

- So hardware vendors must deliver on these market demands by providing a wide range of Serial to “Other” connection types.

Ways to Connect to Serial

- There are Four ways of adding serial communications to you computer.

- Connected directly in PC using PCI, PCI Express
- Connected over the USB
- Connected via Ethernet
- Connected via Wireless eg Bluetooth

- There are advantages and disadvantages of each interface method.
- We will look at first 3 interface types considering compatibility issues, latency and response times
- Latency plays a big role in performance and compatibility Latency= the time it takes for the PC to respond to a stimulus, eg data, handshake line change, loss of signal etc

- Wireless connections, issues regarding throughput & data security will not be discussed here
- Brainboxes provides a robust compatible connection using all these interfaces

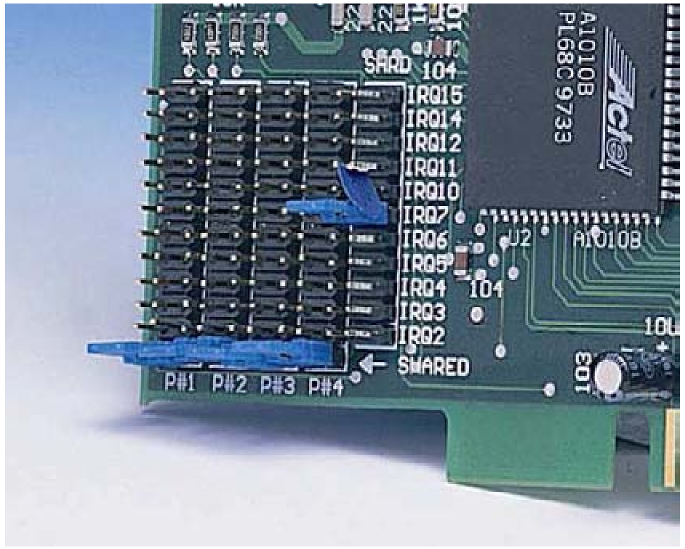
Contents

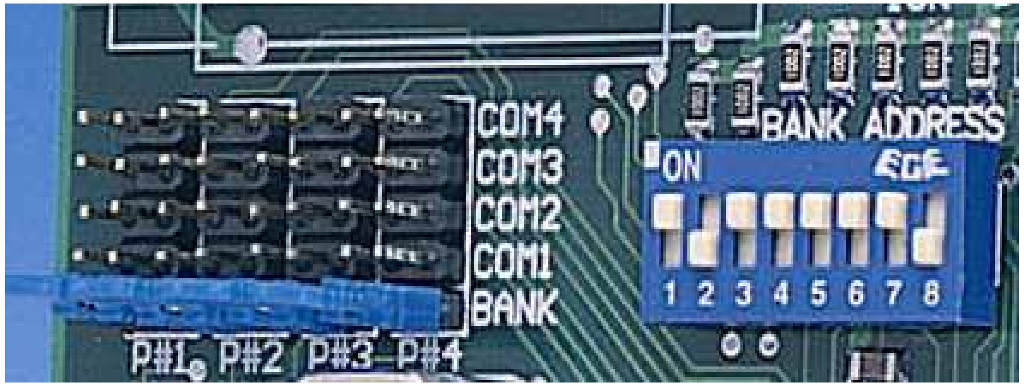
MS DOS

- In The early 1980s when the IBM PC was first introduced IBM and other companies quickly made RS232 serial communications add on boards available to allow the connection of the PC to external devices.
- The PC was generally running the MS DOS operating system and programs were coded so that the program talked directly to whatever hardware resources they needed. Standard Serial ports were always expected to be at fixed i/o addresses and wired to certain interrupts.
- COM1: ADDRESS=03F8 hex   IRQ=04
- COM2: ADDRESS=02F8 hex   IRQ=03
- MSDOS could only run one application program at a time unwritten assumptions of all these programs was that they had exclusive use of and immediate unrestricted access to all PC system resources

ISA Cards

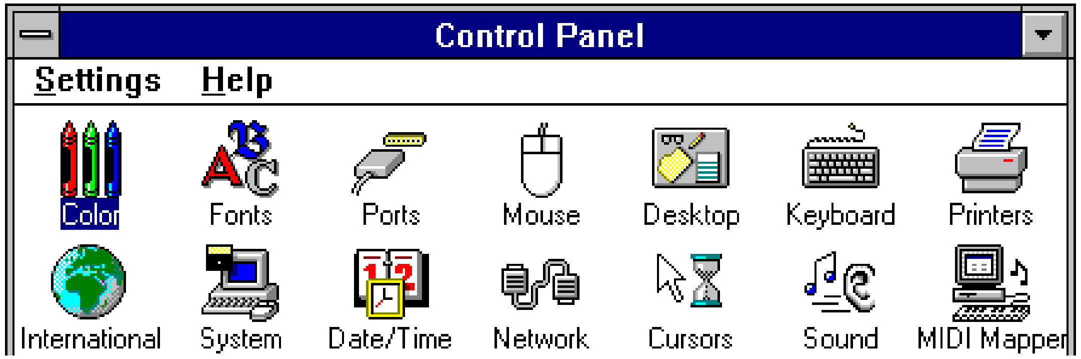
- ISA Cards 1981-1997
- Serial Port Resources are set by the user using DIP switches and jumpers
- ISA no plug and play- the add in card had no mechanism for automatically informing the PC system what resources it needed.
- The PC had no mechanism for allocating resources to add in cards, everything was down to the user knowing exactly what was in the PC and being able to personally ensure that one device in the PC does not clash with any other.





Windows 3.1

- With the tremendous success of Windows 3.1 in early 1990s programmers started to become familiar with writing code so that application programs talked to system Device Drivers that presented a standard API to the outside world. This Windows Device Driver then managed the actual communications to the serial ports hardware.
- Windows 3.1 was a system that implemented cooperative multitasking and so was capable of running multiple well behaved application programs at the same time provided they accessed the system resources strictly via the system Device Drivers through the Microsoft provided Application Programming Interfaces- Device Driver APIs
- It was necessary for the user to tell Windows what hardware resources serial port add in cards needed by entering data using the System in Control Panel.
- Though the Device Drivers in principle allowed serial ports at any otherwise unused i/o address and interrupt in practice COM1: and COM2: used exactly the same values assumed by MSDOS programs. This greatly eased the transition from MSDOS to Windows serial applications



ISA To PCI Transition

- ISA to PCI Change
- In the late 1990s the slots on new PC motherboards underwent a gradual change from being ISA only through a mix of ISA and PCI to being completely PCI slots.
- PCI cards were configured electronically as the PC booted up. The card requests resources from the PC which responds by allocating it i/o address and interrupt. Typically the resources allocated do not correspond to those traditionally used by ISA cards.
- Old MSDOS applications running in a DOS window inside Windows broke mainly because they expected to talk to serial ports at particular i/o addresses and interrupts.
- Huge effort was invested by the industry and customers in migrating their code to work with PCI based PCs. However issues to do with the unwritten assumptions of these programs that they had exclusive use of and immediate unrestricted access to all PC system resources remained hidden in the program code.

brainboxes
www.brainboxes.com

Serial Port Improvement

- The burden of implementing the Windows OS on the CPU where the application programs only actually run during small time slices meant that the PC could no longer do real time control of the serial port. To overcome this extra features were added to the serial port chip- the UART- to lower the burden on the PC the first one being 16 byte transmit & receive FIFO buffers. The data sent and received by Windows to the serial port is no longer a character at a time but is now transferred in FIFO size packets. This trend continues to this day with the following features being implemented in state of the art UARTs.
  - FIFO size grows from 16 to 32 to 64 to 128 and 256 bytes
  - Automatic flow control RTS/CTS or DTR/DSR implemented in hardware
  - XON/XOFF flow control in hardware
  - RS485 Half Duplex Autogating
  - DMA transfers of data to and from the serial port.
  - Baud rates higher than 115,200 and non standard baud rates

- Today the way an application communicates to a serial port has become completely uncoupled from the underlying hardware that actually implements the serial interface. Instead the Application talks to the Windows API in a standard manner, the serial port hardware manufacturer provides a device driver that interfaces to Windows OS and repackages the transactions in a manner that is compatible with the actual serial implementation

brainboxes
www.brainboxes.com

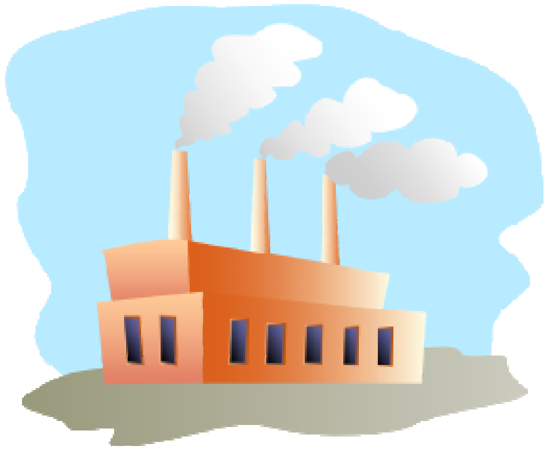
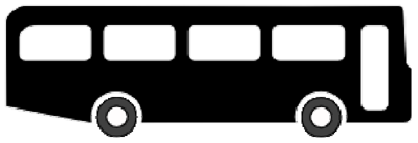
Validation of Platform

- As in the days of DOS, it has proved that there are often many hidden assumptions in code of Windows application programs about the underlying hardware implementation and performance. This results in programs “breaking” when being moved from one version of Windows to another or with different firmware versions or due to changes to the external network system.
- Timeouts in programs that allow an application to recover gracefully when the communications system fail often cause errors to be incorrectly reported when the serial port hardware is changed to one with longer inherent latency.
- These issues may be exacerbated by the use of multi core CPUs in the PC since the context change time from one core to another adds its own latency.
- To mitigate against these issues system suppliers often go through a long costly period of validating a given solution, characterising its operational performance in a large number of defined test cases.
- Sometimes this merely results in a making a particular system work because of tweaking of parameters to match the quirks of the various devices involved, preventing platform evolution over time.

Contents

Latency

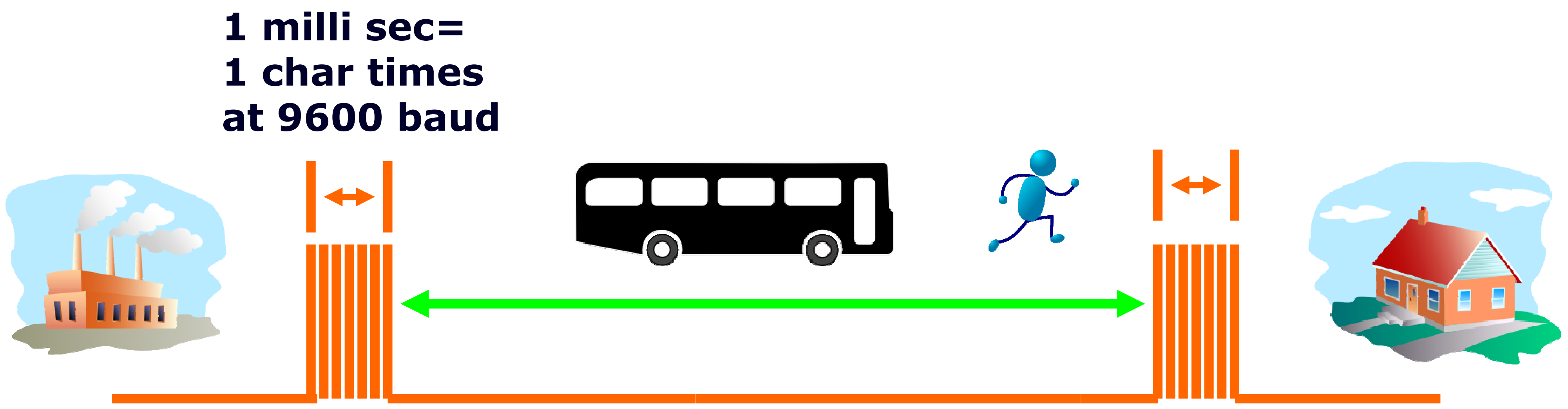
- What is Latency? And why is it such an issue?
- Consider a man's journey home. The bus journey only takes 10 minutes but buses run only once every hour.
- Q: The man leaves work at 5pm at what time will he get home?
- A: If the workman leaves work at 5pm and immediately catches a bus he will be home at 5:10pm.
  But if he has just missed the previous bus he will have to wait an hour for the next bus and be home at 6:10pm.
- His wife at home knows that if she has to wait more than 1hr 10mins for him to travel home then something has gone wrong.
- A return journey can take from 20min to 2hrs 20min

What is Latency?

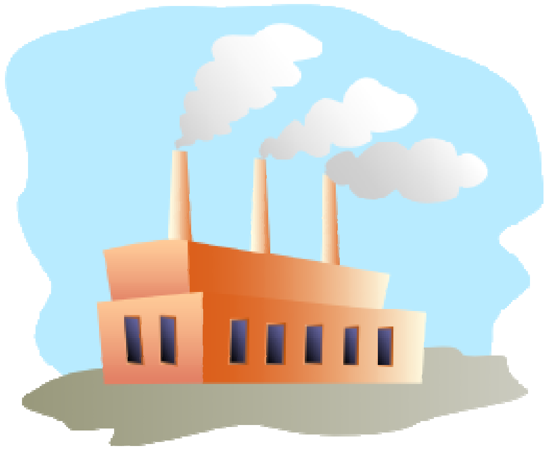
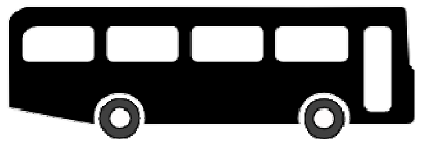
- Latency is a measure of time delay experienced in a system. In the man’s journey the latency is between 10mins and 1hr 10mins. His wife will happily wait 1hr 10mins for him but after this his time is out.

• Typical latency	
• PCI card <1ms	1 char time at 9600 baud
• USB ~16-75ms	16-75 char times
• Local network 5-20ms	5-20 char times
• Internet =100-1,000ms	100 – 1,000 char time



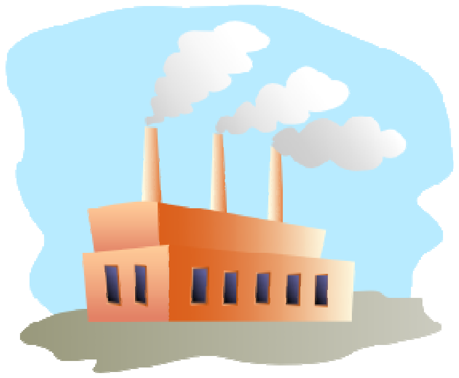
Latency Variations

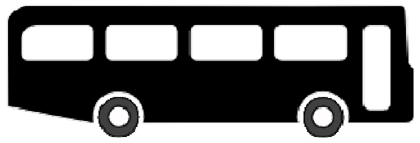
- Latency is not a fixed quantity, it varies, in our example it has a two values that are a factor of 7 in range, 10mins or 1hr 10mins.
- Often other external influences also effect the latency. Imagine that there is a lot of traffic on the road and the bus journey may no longer take 10 minutes.
- Bus systems such as USB and Ethernet share the interface with other connected devices just as the workman's bus shares the road with other traffic. The quoted latency for USB and Ethernet assume no delays due to external traffic on their buses due to other devices.
- User serial programs have to be written to take account of real world latency

Why is Latency An Issue?

- Q: Why is latency such an issue? The different serial interfaces used in a PC have very different characteristic latencies. Programs developed and proved using one interface type often are expecting vastly different timeouts to those found when using other interfaces.
- Timeouts are used in serial communications programs to detect errors or failures such as a device going dead. After a timeout the program signals a failure which often requires user intervention.
- The response time of a system to a stimulus involves a return journey to a device and is thus directly dependant on latency
- Latency effect response times and programs often have inbuilt assumptions about expected response times, and signal success or failure depending whether these response times are met.





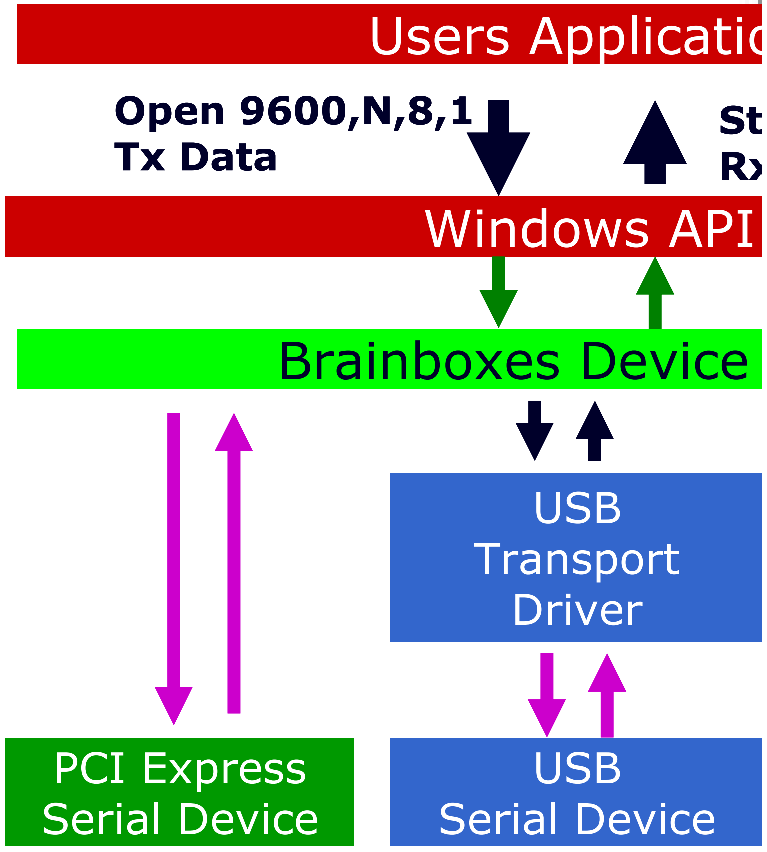
Contents

The Software Challenge

- Software Challenge: users data over any connection.
- #1: To present the data in a uniform manner to the computing device, that is simple to use, easy to configure, is secure and reliable.
- #2: To present the data as if it has a connection that is as robust as a point to point cable, no matter what the underlying physical connection,
  - whether the connection is transient, local or remote, shared or dedicated, wired or wireless.
- Connection Agnostic Interface
  - Software must work the same no matter what the underlying connection
- So hardware vendors must deliver on these market demands

Device Driver Development

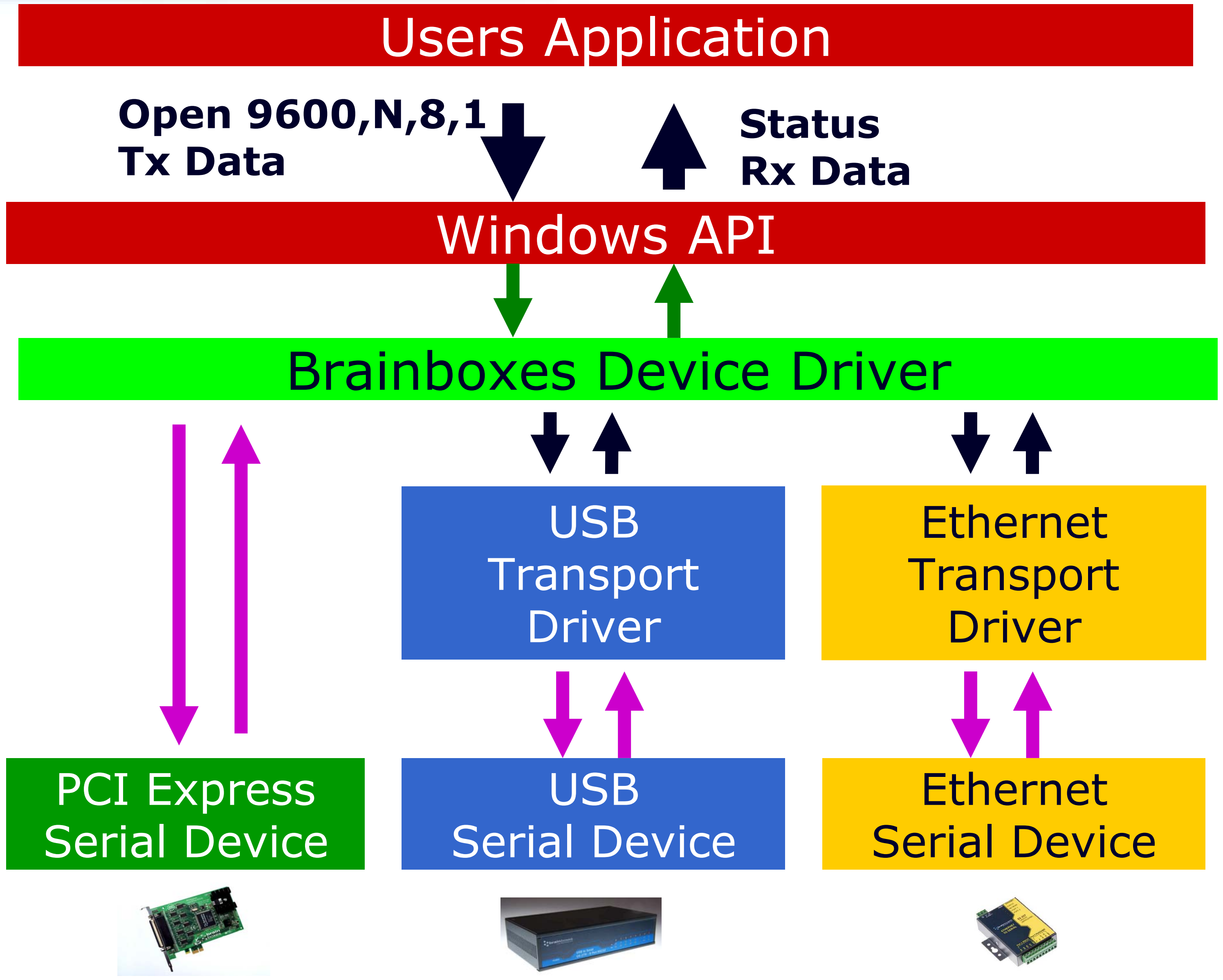
- Serial device manufacturers like Brainboxes base their Device Drivers upon code and recommendations in the Microsoft Device Driver Kit- the DDK.
- How complete an implementation a device Driver is and how compatibly it works varies from one manufacturer to another.
- With over 20 years experience of designing, manufacturing and supplying serial port products Brainboxes have developed an extension set of test cases which we continuously run on device driver code as it being written, so ensuring we have the highest quality code possible.
- Brainboxes always submit our driver code to Microsoft WHQL – the Windows Hardware Quality Labs and obtain the coveted Windows Logo Approval.
  Brainboxes have done so since the earliest days of the HCL – Hardware Compatibly List- instigated with Windows NT in the late 1990s.



Designed for Microsoft Windows Server 2003
CERTIFIED FOR Windows Vista
CERTIFIED FOR Windows Server 2008
Designed for Microsoft Windows XP

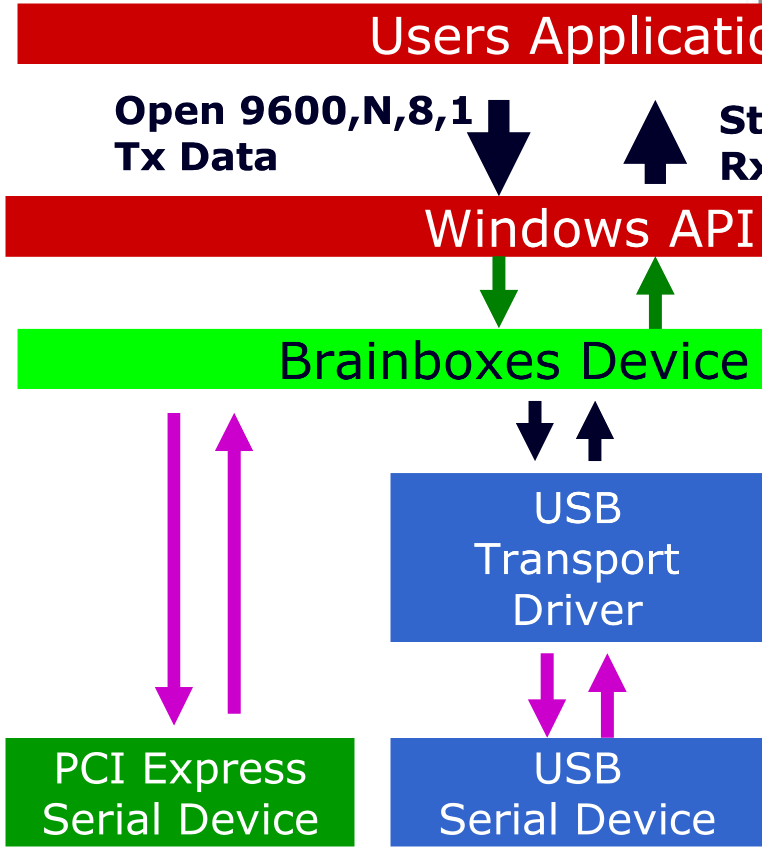
brainboxes
www.brainboxes.com

Layers Between Application and Physical Serial Port



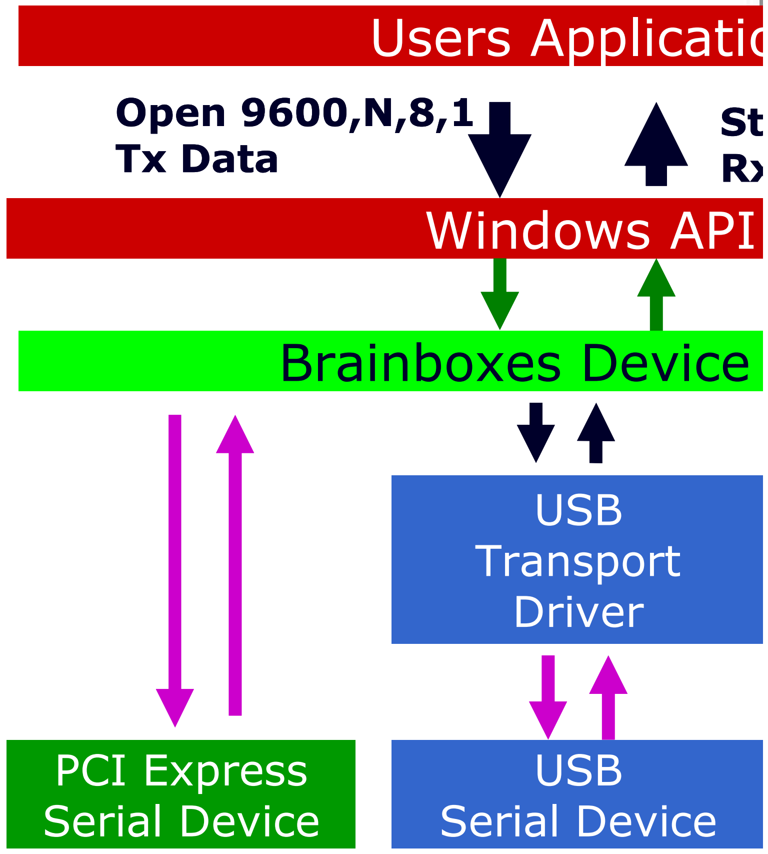
Device Driver Compatibility

- Depending on the physical serial port interface and the actual serial hardware the amount of work needed to implement a device driver varies.
- For ISA serial ports based upon standard 16550 UART chips all that is required is to pass two parameters to Microsoft's own serial port device driver code. This gives the highest level of compatibility since there is no non Microsoft code involved.
- With PCI and PCI Express products a similar process will work.
- However UARTs have had many features added to them to help the CPU cope with the increasing burden of a PC running a pre emptive multi tasking Windows OS. The Microsoft DDK code does not support these features and so serial port manufacturers like Brainboxes must provide their own Device Driver code to implement the now necessary serial port UART features like Automatic handshaking and deeper than 16 byte FIFOs.
- How well these Device Drivers are written is a clear differentiator for Serial port companies.



Device Driver Compatibility

- There is a clear one to one correspondence on Windows API calls when using ISA or PCI Express interfaces even when using highly advanced UARTs. When implementing a Device Driver for Serial Ports that are attached on the other side of a transport link like USB or Ethernet involves dealing with many issues. Achieving compatibility on the Windows PC is a non trivial task since it not only demands an in-depth understanding of the Windows DDK structure but also involves judgement calls in what course of action to follow when status or data is not immediately available and has of necessity to come across a link some distance away in time and space.
- This brings up two main areas of incompatibility that causes applications to fail or work inconsistently:-
  1. Wrong return values passed back to Windows and the application due to misunderstanding the serial API
  2. Hidden assumptions in the code regarding response times etc
- Hidden assumptions prove hardest to debug



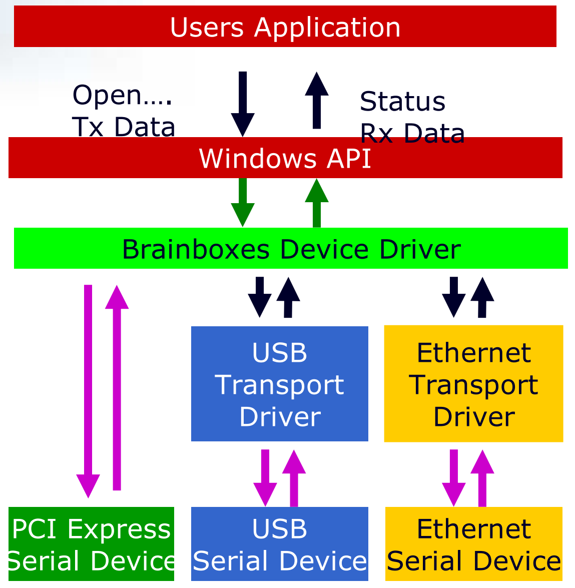
Designed for Microsoft Windows Server 2003
CERTIFIED FOR Windows Vista
CERTIFIED FOR Windows Server 2008
Designed for Microsoft Windows XP

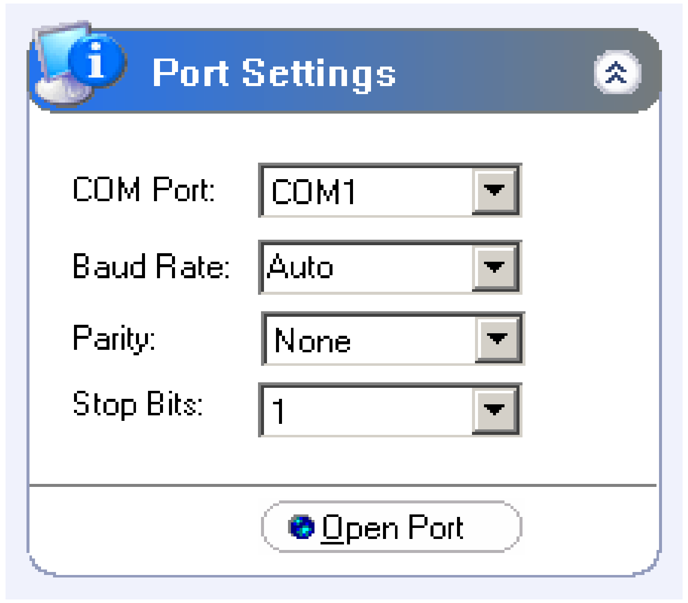
Hidden Assumptions

- Much of the programming model of original ISA interface has been unconsciously carried forward into later written Windows application code.
- Assumption#1: That the serial device is the exclusive use of the Program on the Host PC.
  - True PCI, USB False Ethernet
- Assumption#2: Latency The serial interface is immediately able to provide back to the PC application status information regarding eg the CTS/DCD/DSR/RI handshake inputs of the serial port
  - True PCI False USB, Ethernet
- Assumption#3: Latency Data transmitted and handshake output line status can effected instantaneously
  - True PCI False USB, Ethernet
- Assumption#4: Latency Instructions to change the Baud rate etc or to flush tx/rx data buffers are effected instantaneously
  - True PCI False USB, Ethernet
- Assumption#5+ Programs tweaked to work with particular devices in typical setups will work even with PC platform changes, versions of Windows, different serial device revisions and external end user equipment.
  - True False PCI, USB, Ethernet

Installation and Configuration

- In the past how a port was installed and configured was highly dependant on the interface bus used and the preferences of the device manufacturer.
- As the Windows UI has matured it is increasingly likely that a common unified method based upon underlying Microsoft APIs will be used.
- So today installation is often provoked by a plug and play discovery event based on the familiar USB style device discovery.
- So the COM ports added into the PC system and accessed in a standard way.
- Configuration via the Control Panel or increasingly by a web interface.

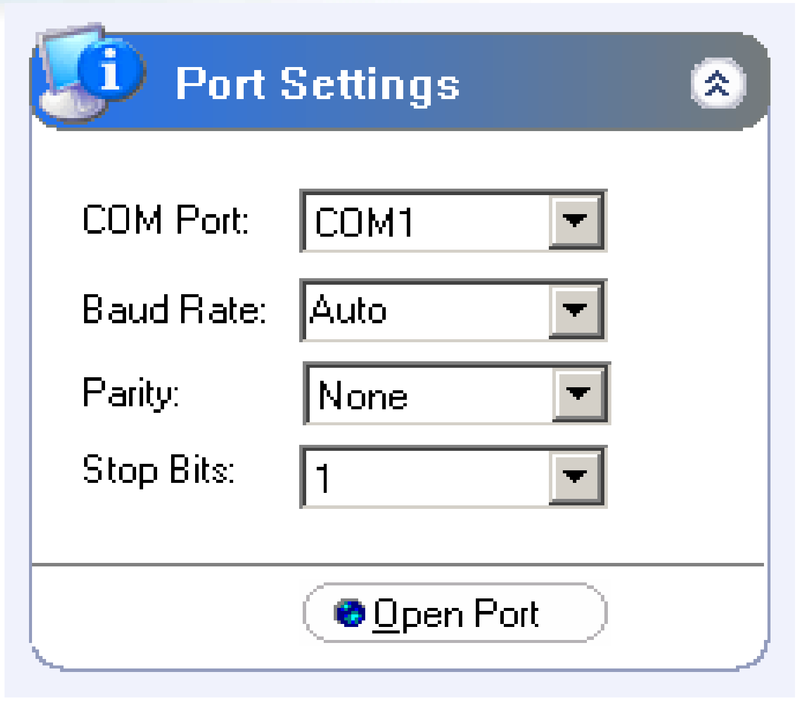



Port Settings

COM Port: COM1
Baud Rate: Auto
Parity: None
Stop Bits: 1

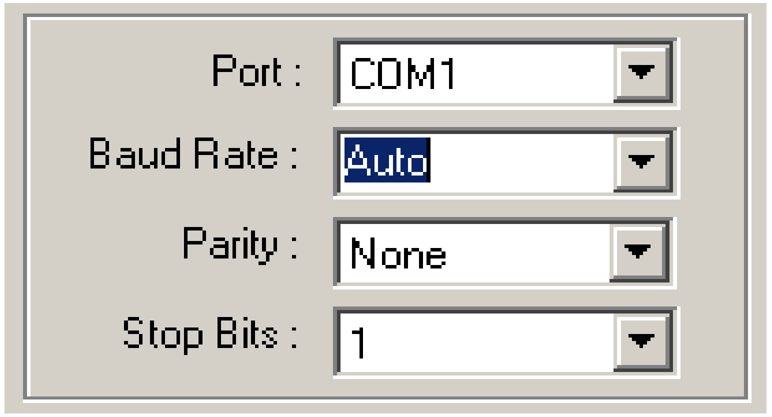
Traditional COM Port Interface

- Our software gives your applications a true COM port interface and handles all the issues about transport layers, latency and connection protocols.
- Allows you to keep making a return on your proven existing software investment.
- Upgrade to a new connection topology without any fuss.



Port Settings

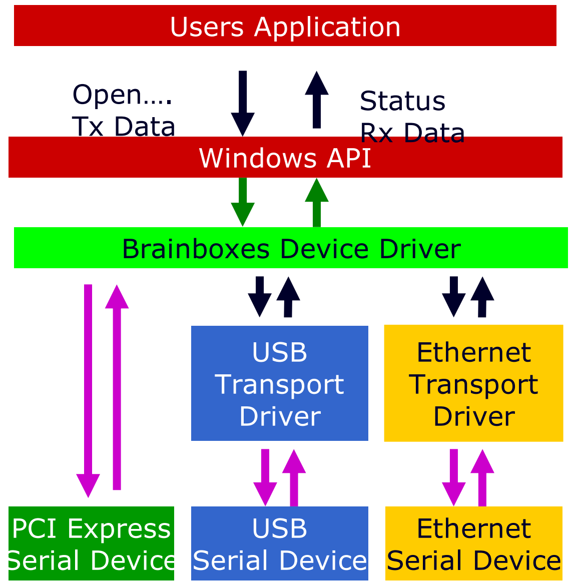
COM Port:	COM1
Baud Rate:	Auto
Parity:	None
Stop Bits:	1

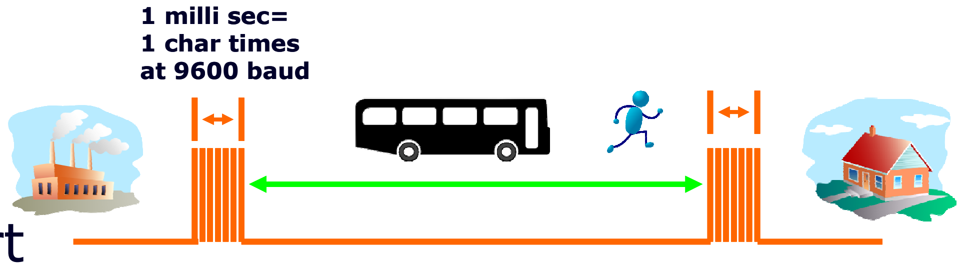


Port :	COM1
Baud Rate :	Auto
Parity :	None
Stop Bits :	1

Latency

- Each layer has its own standardised way of communicating configuration information, serial port data and status information.
- As the data passes up and down the layers from the Application to the physical serial port these processes can add delays:-
  - Packetising of data to improve throughput
  - Data decoded from one standard and re encoded in the next standard.
  - The actual transfer of data across an interface
  - The serialising of the data as it is transmitted or received
- This total delay is called the latency.
- Typically the larger the packet of data being transferred the less import the latency becomes when expressed as a % of the total transfer time.





Contents

PCI & PCI Express Serial Journey

- In mid 1990s PCI Serial Cards were introduced as successor to ISA to Serial cards issues caused initial low take up rates because of:-
  - Software compatibility broken due to PC automatically assigning irq and i/o addresses not compatible with ISA cards, re write of huge number of legacy serial application programs necessary.
  - Plug and Play was nicknamed Plug and Pray
  - Higher cost due to extra circuitry of implementing PCI bus.
  - Slots not available in all PCs.
  - Confusion over PCI and PCI-X, PCI-66 and 5Volt, 3Volt or Universal slots
- PCI to PCI Express transition in 2007-9 avoids all these issues.
  - However initial higher cost of these higher technology devices and the fact that full size desktops are no longer the highest volume PC products hinders early adoption.

PCI Express Serial Journey

- In 2010 PCI Express to Serial devices are widely accepted method for full size desktop PC connection because of:-
  - Benchmark for software compatibility
  - May be bandwidth throttled by wait states on PCI/PCI Express bus between UART transactions for large COM Port count installations.
  - Lowest latency and best response of any serial device
  - Good quality Device Drivers from established serial port specialists like Brainboxes.
  - De facto standard regarding installation and configuration
  - Excellent solution requiring no learning curve
  - Very cost effective with low chip count devices.

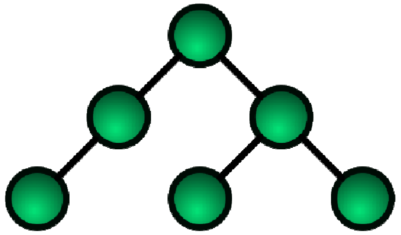
Contents

USB Serial Journey

- In July 1998 Microsoft introduced Windows 98, this provided built in support for USB 1.0 devices, a feature that was lacking in the hugely popular Windows 95.
- USB driving goals ease of use and absolute low cost.
- Due to the disappearance of on board serial ports low cost USB to Serial port devices start becoming popular.
- However only less than 50% of applications worked and USB to Serial devices gained a reputation for being unreliable. The root cause of this was a combination of
  - Low USB 1.0 achievable maximum bandwidth 12Mb/s shared between all devices **if** sub 1GHz PCs can deliver it.
  - Poor implementation of the Device Driver by non specialist serial device manufacturers
  - Incomplete and low performance early hardware implementations

USB Serial Journey

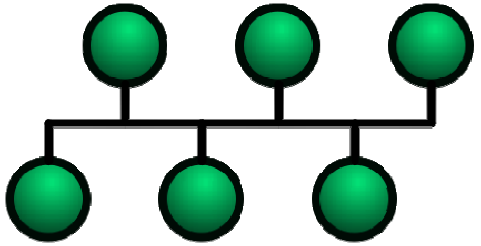
- In 2010 USB to Serial devices are extremely robust and very well received in the market even for critical applications that previously fell over due to:-
  - Higher bandwidth USB 2.0 480Mb/s – a 12Mb/s link is easily sustained
  - Higher performance host PC eg multi core each 2 or 3GHz can more than adequately service USB 2.0
  - 3rd or 4th generation higher performance USB to Serial peripheral hardware
  - Good quality Device Drivers from established serial port specialists like Brainboxes.
- Unfortunately many customers who had poor experiences with early USB serial devices have written the technology off and have moved on to other solutions.



Contents

Ethernet Serial Journey

- Late 1990s Ethernet to Serial devices first introduced in an industrial environment.



- Not widely accepted due to a combination of
  - Programs not working due to hidden dependencies/assumption.
  - Wide spread use of Hubs which flood all segments of the network with all network traffic causing congestion.
  - maximum bandwidth 10Mb/s shared between all devices on network
  - Long network latency
  - Poor implementation of the Device Driver by non specialist serial device manufacturers
  - Incomplete and low performance early hardware implementations
  - Big learning curve for average users requiring having to understand many issues at the System Administrator level
  - No plug and play, no web page configurations
  - High cost devices

brainboxes
www.brainboxes.com

Ethernet Serial Journey

- In 2010 Ethernet to Serial devices are widely accepted as THE method of controlling serial equipment in an industrial environment because of:-
  - Re written programs due to better understanding of networking remote serial devices
  - Wide spread use of Switches which isolate network traffic effectively ensuring point to point communications.
  - maximum bandwidth 100Mb/s shared but effectively much greater due to widespread use of Switches between all devices on network
  - Low network latency
  - Good quality Device Drivers from established serial port specialists like Brainboxes.
  - Backwards compatibility of Windows device drivers greatly reduce the learning curve
  - Easier configuration using Universal Plug and Play and can be managed with a browser via web page interface.
  - Very Cost effective devices due to using high performance, highly integrated microcontrollers.

Contents

Summary

Interface	PCI	PCI Express 1.0	USB 1.0	USB 2.0	Ethernet
Type	Bus	Point to Point	Bus	Bus	Bus
Full Duplex	No	Yes	No	No	No
Slave or Master	Slave	Slave	Slave	Slave	Master
Raw Bit Rate	266 Mega b/s	2.5 Giga b/s	12 Mega b/s	480 Mega b/s	100 Mega b/s
Distance to Host PC	Internal	Internal	5 metre	5 metre	500 metres
Latency	1mS	1mS	100mS	16-75mS	5-20mS
Processor	PC CPU	PC CPU	8 Bit Micro	8 Bit Micro	32 bit ARM
Clock Speed	2GHz PC CPU	2GHz PC CPU	6MHz	6MHz	50MHz
Hot Plug	No	No	Yes	Yes	Yes
Cost	Medium	Medium	Low	Low	Medium

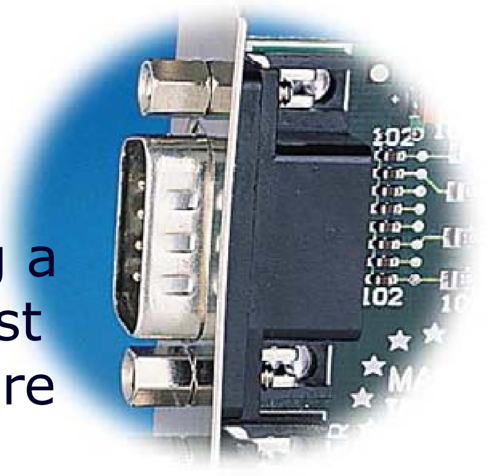
Contents

Recommendations

- Price
  - There is no significant difference between prices of the individual interface units suitable for an industrial environment irrespective of connection methodology.
- PC Choice
  - A PCI/PCI Express solution requires a PC with appropriate internal expansion slot at higher cost.
  - USB and Ethernet solutions can be used with the lower cost Ultra Small Form Factor PC.
- Ease of cabling
  - Ease and price of implementing the different cabling options must be considered.
    A PCI/PCI Express solution can use existing cabling.
  - A USB solution can also use the existing cabling.
  - An Ethernet solution would require provision of new switches and Ethernet cabling.
- Considerations
  - The company strategy regarding long term topology of the manufacturing system should be the guiding light in making choices going forward.
  - There is no budget nor time available for long validation testing and customer does not desire any downtime or production scrap. Thus only systems with the highest supplier confidence must be chosen and any incremental cost incurred considered as the price of ensuring greatest confidence.
  - USB to Serial has no track record of use in similar environments despite it being the lowest cost solution, it has the highest latency and so may be most likely to fall foul of timing matters.
- Recommendations
  - PCI/PCI Express is well attested in this environment and is a powerful contender.
  - Ethernet to Serial is well proven in this environment and is the recommended interface of choice even though the initial cost of cabling is the highest as this system delivers the best long term benefits and highest proven reliability.

Conclusions

- Serial ports are here to stay, for at least another 10 years.
- The market will continue to grow at better than 10% per year.
- Increasingly serial devices are being connected using a wide variety of physical connection mechanisms whilst at the same time requiring a simple, common software interface to allow applications to work robustly.



- Brainboxes will supply hardware & software products in all the different market segments you require to meet these demands.

Your Data, Any Connection, Any Device

- Make Brainboxes your technology partner for any and all your Serial connections

- Connect, Configure, Control your serial ports no matter what computing equipment you have, no matter how it is connected.

- Serial connectivity products of every kind with full software support and technical backup