**1.Find out the details about moxa board ?**

Moxa’s PCI Express serial boards meet the new slot standard for expansion boards and work with any PCI Express slots. The boards have multiple RS-232/422/485 serial ports to connect data acquisition equipment and other serial devices to a PC



Our wide selection of industrial-grade PC/104 and PC/104-Plus serial boards are a popular choice for embedded applications that require connections to multiple serial devices with high data throughput.



Moxa’s new PCI Express Multiport Serial Boards are designed for POS and ATM applications and for use by industrial automation system manufacturers and system integrators.

The boards are compatible with all popular operating systems, and each of them supports data rates of up to 921.6 kbps and provides full modem control signals, ensuring compatibility with a wide range of serial peripherals. In addition, all models work with PCI Express x1, allowing the boards to be installed in any available PCI Express slot (including x1,

x2, x4, x8, x16, x32).

**ADDC™ (Automatic Data Direction Control) for RS-485**

RS-485 uses differential data transmission over two wires to transmit data from one station to another, and allows multiple transmitters and receivers to be used on the same data line. RS-485 uses half-duplex transmission, which means that transmission and reception share the same data channels. For this reason,only one transmitter can be active at any given time.

Moxa’s serial boards have a built-in circuitry to switch transmitters on and off automatically. We call this form of switching ADDC® (Automatic Data Direction Control). ADDC® is much easier to implement than the traditional handshaking method that uses the RTS signal**.**



For application development, Moxa provides an easy-to-use serial communication library called PComm that runs under Windows NT/95/98/2000/XP/2003.

**Applications**

The PCI Express boards are suitable for many different applications, including:

• Internet/Intranet Connections

• Remote Access

• Multi-user Applications

• Industrial Automation

• Office Automation

• Telecommunications

• PC-based Vending Machines and Kiosks

• POS (Point-of-Sale) Systems

**Features**

The PCI Express boards have the following outstanding features:

• PCI Express ×1 compliant

• Low-profile board for compact-sized PCs

• Data flow LED display onboard

• 128-byte FIFO and on-chip H/W, S/W flow control

• 50 bps to 921.6 kbps transmission speed

• Embedded 15 kV ESD surge protection

• Drivers are provided for Windows, Windows CE, Windows XP Embedded, DOS, Linux (32-bit/64-bit),SCO

<https://www.moxa.com/getmedia/3a00f341-5d2c-4048-ac27-3ae2b3274481/moxa-cp-116e-a-manual-v11.3.pdf>

**2. How to print the current time, seconds, milliseconds, nanoseconds in linux ?**

in linux the command "date" command give current time and datre

date +%s returns the number of seconds since the epoch,

date +%s%N returns the number of seconds + current nanoseconds

date +"%T.%N" returns the current time with nanoseconds.

date +"%T.%6N" returns the current time with nanoseconds rounded to the first 6 digits, which is microseconds.

date +"%T.%3N" returns the current time with nanoseconds rounded to the first 3 digits, which is milliseconds

%N nanoseconds (000000000..999999999)

%s seconds since 1970-01-01 00:00:00 UTC(epoch)

**3. Why inline function needs static storage class ?**

Use“static” before inline. Using static keyword forces the compiler to consider this inline function in the linker, and hence the program compiles and run successfully

A function defined with static inline. Stand-alone object code may be emitted if required. You can have multiple definitions in your program, in different translation units, and it will still work.

**4. How the arguments passed or pushed to other function and how it assigns the value to arguments. Explain using diagram for below example**

**int main()**

**{**

**while(1)**

**{**

**data\_acq(int a, int b, int c)**

**}**

**}**

**void data\_acq(int x, int y, int z)**

**{**

**....**

**}**

The function called at program startup is named main .

In a function call, the types of the evaluated arguments must match the types of their corresponding parameters. If they do not match, the following conversions are performed in a manner that depends on whether a prototype is in scope for the function:

Arguments to functions specified with prototypes are converted to the parameter types specified in the prototype, except that arguments corresponding to an ellipsis (...) are converted as if no prototype were in scope.