DEVELOPING MULTIFUNCTIONAL SERIAL-PARALLEL DATA COMMUNICATION INTERFACE FOR PC-BASED CONTROL SYSTEM

Yuliadi Erdani
Manufacturing Automation and Mechatronic Department, Polytechnic Manufacture Bandung – 40135
E-mail: yul_erdani@polman-bandung.ac.id

ABSTRACT

Today, Personal Computers (PC) have increasing role in measurement and control technology. The more and more complex processes in many fields of technique and research can often be controlled only by programmable control systems. The PC-based control system is an alternative and smart solution of programmable control systems. One important thing of the PC-based control system is the serial-parallel data communication. Problems and difficulties with hardware and software concerning serial-parallel data transmission have been discussed in many literatures and are still the important consideration of building PC-based control systems. To overcome mentioned problem, a multifunctional serial-parallel data communication interface has been developed.

This paper describes the development of the interface that covers its design and implementation and the discussion of software and hardware problems, and the solutions.

Keywords: *technique, pc-based control system, data communication*

1. INTRODUCTION

PC is a small computer system, which has been developed for various functions and aims. PC can be used for processing word and data, designing graph, programming control system, etc. The use of PC's for control applications becomes a popular solution to build smart control system. [2] stated that the days of the PC being used just for visualisation and production data acquisition in control and automation applications are rapidly becoming a thing of the past. The PC is now increasingly recognised as an open and powerful hardware platform, which can provide effective and reliable control, with no requirement for additional processors or complex hardware additions. This recognition is being brought about by the successful development and implementation of powerful software based control solutions, designed to run on dedicated industrial PCs, with particular advantages being apparent in areas such as PLC and NC motion control. [3] noted that the use of PC's for industrial control applications is a fast growing market. Over the last few years, industrial transformation has been spearheaded by the Windows-based operating system and PC-based controls [4].

A number of the reasons for this trend are financially motivated - PC-Based control system can tremendously decrease the cost of a control system. However, there are other non-monetary reasons why PC-Based control system is becoming extremely popular. Among benefits, PC-based control easily integrates with other tasks, including data logging and manufacturing execution systems. It also runs on nonproprietary hardware [5]. [6] is an example of PC-based control system. It provides a software system for the PC-based control, programming and visualization of general technical processes in factory, process and building automation. PLC functionality, libraries for project support. [1] is also an example of PC-based control system that uses expert system and web-based application to carry out the control functions.

PC can be used easily and effectively for developing measurement and control application via serial [7] and parallel interface [8]. The uses of serial and parallel interface for control system are still the important consideration in transmitting control data. The serial interface RS 232 e.g. is used in many devices of control system and still the widespread serial communication interface. To arrange a data link via serial interface demands often considerable patience. The reason lies in the exceptionally many possibilities of interface parameter setting and different cable connections. Additional problems are software problems that are caused by the lack of communication between manufacturers [7].

2. DATA COMMUNICATION

This section describes a brief review of serial and parallel data communication. The basic method for transferring data in digital communication system is serial and parallel communication. Serial communications is the process of sending data one bit at one time, sequentially, over a communications channel or computer bus. This is in contrast to parallel communications, where all the bits of each symbol are sent together. Serial communications is used for all long distance communications and most computer networks, where the cost of cable and synchronization difficulties makes parallel communications impractical. Serial computer busses are becoming more common as improved technology enables them to transfer data at higher speeds [9].

The serial port RS 232 on PC is an example of standard serial communications. Serial ports have

a single data wire, and the remainder of the wires is either ground or control signals. Other serial communication standard are USB (Universal Serial Bus) and Firewire. The parallel port LPT on modern PC is an example of standard parallel communications connection. The parallel port has 8 data wires, and a large series of ground wires and control wires. IDE harddisk connectors are another example of standard parallel connections in a computer system.

Parallel ports suffer extremely from inter-symbol interference (ISI) and noise, and therefore the data can be corrupted over long distances. Also, because the wires in a parallel system have small amounts of capacitance and mutual inductance, the bandwidth of parallel wires is much lower then the bandwidth of serial wires. It is known that an increased bandwidth leads to a better bit rate. It is also known that less noise in the channel means we can successfully transmit data reliably with a lower SNR. Another problem with parallel communication is the skew effect.

There are two primary forms of serial transmission: Synchronous and Asynchronous. Depending on the modes that are supported by the hardware, the name of the communication sub-system will usually include a symbol “A” if it supports Asynchronous communications and a symbol “S” if it supports Synchronous communications. The acronym for both forms is: UART and USART. UART stands for Universal Asynchronous Receiver/ Transmitter and USART stands for Universal Synchronous-Asynchronous Receiver/Transmitter. This paper focuses mainly on the serial communication using UART form [10].

The standard serial communication UART controller is the key component of the serial communications subsystem of a computer. The UART takes bytes of data and transmits the individual bits in a sequential fashion. At the destination, a second UART re-assembles the bits into complete bytes. Serial transmission is commonly used with modems and for non-networked communication between computers, terminals and other devices.

Asynchronous transmission allows data to be transmitted without the sender having to send a clock signal to the receiver. Instead, the sender and receiver must agree on timing parameters in advance and special bits are added to each word which is used to synchronize the sending and receiving units.

When a word is given to the UART for Asynchronous transmissions, a bit called the "Start Bit" is added to the beginning of each word that is to be transmitted. The Start Bit is used to alert the receiver that a word of data is about to be sent, and to force the clock in the receiver into synchronization with the clock in the transmitter. These two clocks must be accurate enough to not have the frequency drift by more than 10% during the transmission of the remaining bits in the word. A start bit signals the beginning of each character frame. It is a transition from negative (MARK) to positive (SPACE) voltage; its duration in seconds is the reciprocal of the baud rate. If we're transmitting at 9600 baud, then the duration of the start bit and each subsequent bit will be about 0.104 ms. The entire character frame of eleven bits would be transmitted in about 1.146 ms.

After the Start Bit, the individual bits of the word of data (data bits) are sent, with the Least Significant Bit (LSB) being sent first. Data bits are transmitted "upside down and backwards." That is, inverted logic is used and the order of transmission is from least significant bit (LSB) to most significant bit (MSB). To interpret the data bits in a character frame, you must read from right to left, and read 1 for negative voltage and 0 for positive voltage. For the figure above, this yields 1000001 (binary) or 41 (hex). An ASCII conversion table shows that this is the letter "A".

Each bit in the transmission is transmitted for exactly the same amount of time as all of the other bits, and the receiver ``looks" at the wire at approximately halfway through the period assigned to each bit to determine if the bit is a 1 or a 0. For example, if it takes two seconds to send each bit, the receiver will examine the signal to determine if it is a 1 or a 0 after one second has passed, then it will wait two seconds and then examine the value of the next bit, and so on. The sender does not know when the receiver has ``looked" at the value of the bit. The sender only knows when the clock says to begin transmitting the next bit of the word. When the entire data word has been sent, the transmitter may add a Parity Bit that the transmitter generates.

The Parity Bit may be used by the receiver to perform simple error checking. Then at least one Stop Bit is sent by the transmitter. When the receiver has received all of the bits in the data word, it may check for the Parity Bits (both sender and receiver must agree on whether a Parity Bit is to be used), and then the receiver looks for a Stop Bit.

An optional parity bit follows the data bits in the character frame. The parity bit, if present, also follows inverted logic: read 1 for negative voltage and 0 for positive voltage. This bit is included as a simple means of error checking. The idea is this: one specifies ahead of time whether the parity of the transmission is to be even or odd. Suppose the parity is chosen to be odd. The transmitter will then set the parity bit in such a way as to make an odd number of 1's among the data bits and the parity bit. The transmission in the figure above uses odd parity. There are five 1's among the data bits, already an odd number, so the parity bit is set to 0.

Consider an even parity scheme using nine bit code words, consisting of eight data bits followed by a parity bit. The parity of the data 10111101 is even (there are 6 '1' bits). The parity bit will be 0,

giving the codeword 101111010. The parity of the data 01110011 is odd (there are 5 '1' bits). The parity bit will be 1, giving the codeword 011100111. The parity of the data 00000000 is even (zero is even). The parity bit will be 0, giving the codeword 000000000. A null or non-existent bit stream also has zero '1' bits, so it would get the parity bit 0 in an even parity scheme.

The last part of a character frame consists of 1, 1.5, or 2 stop bits. These bits are always represented by a negative voltage. If no further characters are transmitted, the line stays in the negative (MARK) condition. The transmission of the next character frame, if any, is heralded by a start bit of positive (SPACE) voltage [11]. If the Stop Bit does not appear when it is supposed to, the UART considers the entire word to be garbled and will report a Framing Error to the host processor when the data word is read. The usual cause of a Framing Error is that the sender and receiver clocks were not running at the same speed, or that the signal was interrupted.

Regardless of whether the data was received correctly or not, the UART automatically discards the Start, Parity and Stop bits. If the sender and receiver are configured identically, these bits are not passed to the host. If another word is ready for transmission, the Start Bit for the new word can be sent as soon as the Stop Bit for the previous word has been sent. Because asynchronous data is ``self synchronizing", if there is no data to transmit, the transmission line can be idle. Figure 1 shows the structure of serial data in UART form.

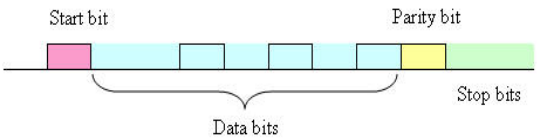

Figure 1. Serial data in UART form

3. SYSTEM AND APPLICATION

In the PC-based control system, the PC is the primary tool in providing control function. The communication between PC and its environment is the communication using serial and parallel ports [7] [8]. In principle both communication ports can be used directly to control the plant digitally. But to give better communication performance, some additional hardware should be implemented and connected into communication ports, such as serial receiver, serial sender, A/D converter, D/A converter, parallel interface bus, motor driver, etc. Figure 2 shows the block diagram of PC-based control system.

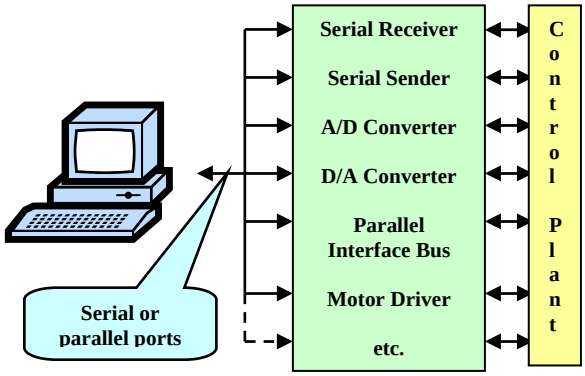

Figure 2. Block diagram of PC-based control system

The developed system and application focuses only on developing serial-parallel data communication interface that enable the PC to communicate to the environment. The serial sender and receiver would be the alternative to build the interface. It is a fully hardware interface. The cost of the hardware is quite chip. It amounts US$ 10,-. The problem of this system is the simplicity and expandability. The system can not process data bidirectional. To send data from PC via serial port, the PC needs serial receiver. To receive the data from the plant, the PC needs serial sender. The number of I/O (Input/Output) for parallel data can not be extended.

Another alternative is to uses microcontroller AT 89X051 from ATMEL. It is a computer system that is packed in one chip. It is known as single chip microcontroller [12]. The microcontroller system needs hardware and software effort. Due to the simplicity and market growth of microcontroller, the finished product of complete microcontroller system can be found easily in the market. It cost about US$ 20,-. The advantages of the system are expandability and extendibility. The number of I/O can be expanded easily. The function of the system can also be extended, changed or modified by software programming without having to add new hardware.

Using microcontroller, the developed data communication interface can transfer data bidirectional. The interface serves as serial sender and receiver at the same time. The interface is able to detect any coming data serial or parallel. The serial data will be converted to parallel data and sent to the parallel port. The parallel data will be converted to serial data and sent to serial port. Figure 3 illustrates the diagram block of the system.

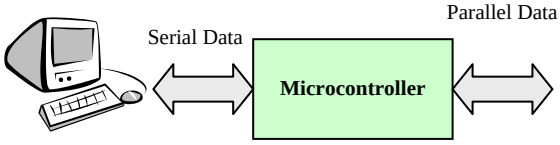

Figure 3. Diagram block of the system

Due to the character of the microcontroller, it is not easy to serve as serial sender and receiver at the same time. To realize the function, there are two strategy are used in transferring data, i.e. polling system and interrupt system.

Polling can be defined as making continuous requests for data from another device. For example, modems that support polling can call another system and request data [13]. The polling system can also be called as a communications control method used by some computer/terminal systems whereby a "master" station asks many devices attached to a common transmission medium, in turn, whether they have information to send. The polling system is used in receiving parallel data and transferring to serial port.

To receive parallel data using the polling system, the hardware connection must be set up as follows: 8 bits of parallel port are used for data bits and other 2 bits of parallel port are used for handshaking Figure 4 shows the illustration of cable connection for receiving parallel data.

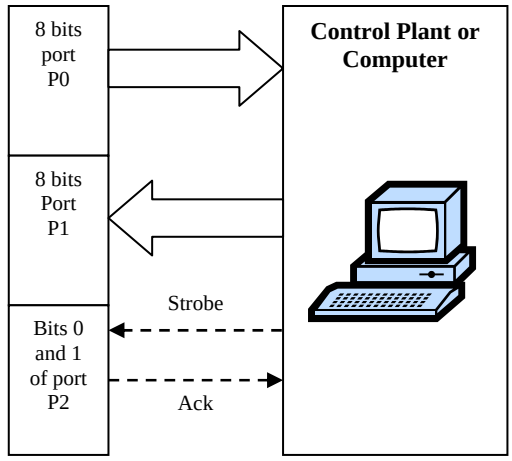

Figure 4. Port cable connection

In telecommunication and microprocessor systems, the term handshaking has the following meanings: In data communications, a sequence of events governed by hardware or software, requiring mutual agreement of the state of the operational modes prior to information exchange. The process used to establish communications parameters between two stations. Figure 5 show the illustration of handshaking process of the data communication.

The process of receiving parallel data is as follows: The microcontroller polls data port via Strobe bit. The computer or control plant sends data to the microcontroller. Strobe is set to 1. It means that the data has been sent. The microcontroller receives the data and sends acknowledgment (ack=1) to the Computer or Control Plant if it has received the data. The computer or control plant resets Strobe. The microcontroller resets again the Acknowledgment.

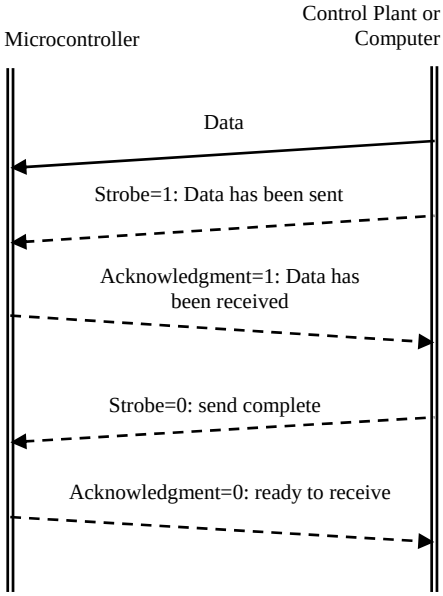

Figure 5. Handshaking process

The next strategy is the interrupt system. It is used to receive data from serial port. Interrupt is a hardware signal that causes a CPU to set aside normal processing and begin execution of an interrupt handler. An interrupt is parameterized by the type of bus and the interrupt level, and possibly with an interrupt vector number. The kernel uses this information to select the interrupt handler for that device. Receiving serial data has priority high. It is illustrated in Figure 6.

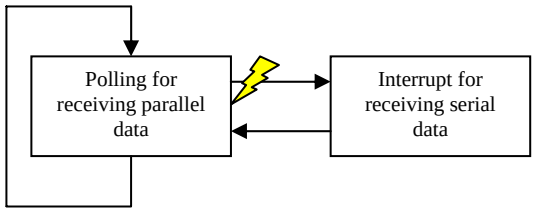

Figure 6. Process of data transfer

The algorithm is the important part in developing application software. The algorithm for the whole system has been designed and can be seen as follows:

ISSN: 1907-5022

```
Declare variables (define addresses)
Set up communication parameters

// Parallel-Serial Data Transfer
// this part is the main loop. It will be interrupted
// if the Serial Interrupt comes and the program
// part "Serial-Parallel Data Transfer" will be
// executed immediately
Repeat
    While Strobe = 0 Do Loop
    Read Parallel Input Port
    Send data to Serial Port
    While Serial Transmit Interrupt = 0 Do Loop
    Set Serial Transmit Interrupt = 0
    Set Acknowledgement = 1
    While Strobe = 1 Do Loop
    Set Acknowledgement = 0
Until Program ends

// Serial-Parallel Data Transfer
Disable Serial Interrupt
While Serial Receive Interrupt = 0 Do Loop
Read Serial Port
Send to Parallel Output Port
Set Serial Receive Interrupt = 0
Enable Serial Interrupt
Return from Interrupt
```

4. CONCLUSION

The developed system integrates the function of serial sender and serial receiver hardware. The system can convert and transmit serial to parallel data and backward. The data are transmitted byte-wise. The use of microcontroller brings the advantages in reducing hardware efforts during implementation phase.

The complex way of communication process is intended to ensure the accuracy of data and to reduce the error possibility during transfer. The developed system is a small step to overcome the problems concerning serial and parallel data communication but has potential to give a new strategy and contribution in developing more complex data communication system.

Feature work of developed concept will mainly focus on the attempt to expand the functionality of the system, to develop data protocol and check sequence system that can improve data transfer performance, to develop error handling system, and to analyze data transfer duration and the complexity of the algorithm.

REFERENCES

[1] Yuliadi Erdani, "Applying simply web-application and expert system into PC-based control system", in *Proceeding the 7th Seminar on Intelligent Technology and Its Applications (SITIA)*, Surabaya, Indonesia, May 2, 2006. ISBN 979-95989-8-2.

[2] Stephen Hayes, *The advantages of PC-based control systems*, EngineeringTalk, December 2000, http://www.engineeringtalk.com/news/hay/hay112.html

[3] Cindy Hollenbeck, *PC-Based Control - It's not Just About Saving $$$*, SoftPLC Corporation, 2004, http://www.softplc.com/notjustsaving.php

[4] Sathyajit Rao, *Industrial/Automotive PC-based Control System Set to Grow*, Technology Report, Electronic Design, 2003, http://www.elecdesign.com/Articles/ArticleID/5066/5066.html

[5] Murray A. Mickey, *Combining PC Control and HMI*, InTech, 2002, http://www.isa.org/InTechTemplate.cfm?Section=InTech&template=/ContentManagement/ContentDisplay.cfm&ContentID=16206

[6] Company Product Article, *4Control PC-based – PC-based Control System*, Softing AG, 2004-2005, http://www.softing.com/en/controls/products/pc_based.htm

[7] Burkhard Kainka, *Messen, Steuern und Regeln ueber die RS-232 Schnittstelle*, ISBN 3-7723-6057-2, Franzis-Verlag Gmbh, Munich, 1994.

[8] Wolfgang Link, *Messen, Steuern und Regeln ueber die Parallel Schnittstelle*, ISBN 3-7723-5954-X, Franzis-Verlag Gmbh, Munich, 1994.

[9] Wikipedia – the free encyclopedia, *Serial Communications*, 2006. http://en.wikipedia.org/wiki/Serial_communications

[10] Durda, Frank IV, *Serial and UART Tutorial*, the FreeBSD Documentation Project, 2005. http://www.freebsd.org/doc/en_US.ISO8859-1/articles/serial-uart/

[11] Courtois, Francis, *Interfacing a Mac Serial Port to RS 232 Lab Equipment*, 1997. http://francis.courtois.free.fr/jc1/serial/main.html

[12] Budioko, Totok, *Belajar dengan mudah dan cepat Pemrograman Bahasa C dengan SDCC (Small Device C compiler) pada Mikrokontroller AT 89X51/ AT 89C51/52, Teori, Simulasi dan Aplikasi*. Penerbit Gaya Media, 2005. ISBN 979-3469-59-5

[13] Larus CXR, *Glossary of Common Telecom Terms and Acronyms*, 2006. http://www.cxrlarus.com/assets/glossary.html