Design and Implementation of Serial Communication for Automatic Titanium Alloy Sorting System

Shuping cui¹,a, Zhifeng Liu¹,b, Yan,lv¹,c
¹Lab of Industrial Control Network and System Shenyang Institute of Automation, Chinese Academy of Sciences, Liaoning Shenyang, China
ₐcuisp@sia.cn, bliuzhfeng@sia.cn, csylvyan@sia.cn

Keywords: serial communication, sorting machine, machine vision

Abstract. The heat generated during Titanium alloy cutting can cause bad products, which must be eliminated by Sorting System. This paper describes the design and implementation of serial communication between the automatic sorting machine and the host computer, introduces the system framework and the software flow, and gives a detailed introduction to the communication process, parameters setting and also gives part of the MFC dialog-based serial communication program. It shows that the design and implementation completely meets the requirements of system’s performance and function.

Introduction

During the process of Titanium alloy cutting, a lot of heat are generated which will cause the change of materials, and bring bads. Titanium alloy automatic sorting system based on machine vision and image processing can realize the separation of goods and bad ones. The sorting machine consists of the vibration feeding, transmission and dividing mechanism. The main control system is composed of OMRON CP1L and PLC. The communication between PLC and host computer rely on serial ports. Host computer does status inquiry of PLC and then go ahead to control vision and dividing mechanism by serial communication.

RS-232 (ANSI/EIA-232 Standard) is the serial connection found on IBM-compatible PCs. It is used for many purposes, such as connecting a mouse, printer, or modem, as well as industrial instrumentation. RS-232 is limited to point-to-point connections between PC serial ports and devices. RS-232 hardware can be used for serial communication up to distances of 50 feet.

The system framework

The framework of Titanium alloy automatic sorting system is composed of four parts, which are

![Diagram of Automatic Titanium alloy sorting machine system](image-url)

Fig. 1 Automatic Titanium alloy sorting machine system

industrial camera, light source, sorting machine and host computer. Host computer is the master of the system, and all the rest except light source are slaves. The master needs transfer data and commands
with the slaver. To finish these tasks, efficient, reliable and flexible serial communication should be built between host computer and sorting system. Kinds of programming tools can accomplish this work, such as VB, VC++ and Delphi etc. VC++ is the best choice because of its extraordinary capacity. First, the program base on VC++ can run very fast, and has a high degree of portability. Second, VC++ has rich API functions for image processing and communication program. In this system, the image from the camera should be calculated and analyzed to control the activity of sorting machine. The image processing part is not talked about in this paper.

The software flow

The software flow is illuminated by the Fig2. Just as the figure shows, the operation of serial ports goes through the entire program. The program is started with essential initiation. After that it will go into an infinite loop. The first step of the loop is writing the serial ports, and then reading them. Either reading or writing is failed, the program will return. After the analysis on the data from serial communication, the subsequent action can be decided that is going back to the start of the loop, or going forward to the step of taking picture and image processing. According to the result of image processing, current sample can be recognized. The goods will be retained, and the bads will be eliminated by dividing mechanism after host computer finishes writing to the serial ports. Then the program goes back to the start of the loop. So host computer accomplishes the status inquiry and control to the sorting machine by the serial communication.

The main content of software development of automatic Titanium alloy sorting system is to exploit a MFC dialog-based application with windows API functions for serial communication and image processing. Dialog-based application is convenient to debug the operation, such as watching the data transferred or received from serial ports. On the dialog a start button can be created to activate the action of the program. During the flow of the software, all steps except taking picture and processing Image rely on the analysis based on the data from serial communication.

![Diagram of software flow](image)

Fig. 2 The software flow of the automatic Titanium alloy sorting system

The realization of serial communication between host computer and sorting system

There are two ways of building the serial communication on the MFC dialog-based development platform. First is using ActiveX controls. ActiveX control programs are modular programs designed to give a specific type of functionality to a parent application. They are easy to use in MFC dialog-based applications because they have been designed for reuse, but not very flexible and give
you little control of the implementation. Second is using Windows API functions, which is quite flexible and gives engineers much freedom to intervene the program’s operation.

Developing a serial communication application should obey the following four steps.

### 3.1 Open the serial ports

CreateFile function is used to open serial ports. The prototype of this function is as follows:

```c
HANDLE CreateFile( LPCTSTR lpFileName, 
                   DWORD dwDesiredAccess, 
                   DWORD dwShareMode, 
                   LPSECURITY_ATTRIBUTES lpSecurityAttributes, 
                   DWORD dwCreationDistribution, 
                   DWORD dwFlagsAndAttributes, 
                   HANDLE hTemplateFile);
```

The lpFileName is the pointer to the name of serial ports. The dwDesiredAccess is the access status, and can be set to GENERIC_READ or GENERIC_WRITE or both. The dwShareMode is the share mode. Because the serial ports can not be shared, so dwShareMode must be 0. The dwCreationDistribution must be OPEN_EXISTING for serial ports. The dwFlagsAndAttributes can be set to NULL for synchronous operation and FILE_FLAG_OVERLAPPED for asynchronous operation. The hTemplateFile parameter must be NULL for serial ports. If the function fails, the return value is INVALID_HANDLE_VALUE. If it succeeds, the return value is an open handle used by the subsequent functions.

Here are the codes:

```c
HANDLE hCom;
hCom=CreateFile("COM1", 
                 GENERIC_READ|GENERIC_WRITE, 
                 0, 
                 NULL, 
                 OPEN_EXISTING, 
                 0, 
                 NULL);
if(hCom==(HANDLE)-1) {//} else {//}
SetupComm(hCom,5096,5096);
```

### 3.2 Initiate the serial ports

After the serial ports are opened, they require initiation with the DCB (device-control block) structure. The DCB structure defines the control setting for a serial communications. The members of DCB structure include baud rate, byte size, parity, stop bits etc. Before changing the settings of serial ports, GetCommState command should be used to get the current control settings. Filling DCB structure with needed values can change the connection parameters to those needed at the moment. Then SetCommState command configures the serial ports according to specifications in a DCB structure.

The following code illustrates setting baud rate to 115200bps, byte size to 7, stopbits to 2, and parity to 2(even parity).

```c
DCB dcb;
GetCommState(hCom,&dcb);
dcb.BaudRate=115200;
dcb.ByteSize=7;
dcb.Parity=EVENPARITY;
dcb.StopBits=TWOSTOPBITS;
SetCommState(hCom,&dcb);
```
3.3 Setting Time-out

To deal with the situation of exception during data transfer, the COMMTIMEOUTS structure is used in the SetCommTimeouts and GetCommTimeouts functions to set and query the time-out parameters for serial ports which determine how long it is before ReadFile or WriteFile operation is abandoned.

```c
COMMTIMEOUTS TimeOuts;
GetCommTimeouts(hCom,&TimeOuts);
TimeOuts.ReadIntervalTimeout=1000;
TimeOuts.ReadTotalTimeoutMultiplier=500;
TimeOuts.WriteTotalTimeoutConstant=5000;
TimeOuts.WriteTotalTimeoutMultiplier=500;
TimeOuts.WriteTotalTimeoutConstant=2000;
SetCommTimeouts(hCom,&TimeOuts);
```

3.4 Writing and reading the serial ports

WriteFile function writes to the serial ports and ReadFile functions reads data from the serial ports. The prototype of WriteFile function is as follows:

```c
BOOL WriteFile( HANDLE hFile,
    LPCVOID lpBuffer,
    DWORD nNumberOfBytesToWrite,
    LPDWORD lpNumberOfBytesWritten,
    LPOVERLAPPED lpOverlapped);
```

The hfile is the handle of the serial port got from the CreateFile function. The lpBuffer is the pointer to data to write. The nNumberOfBytesToWrite is the number of bytes to write. The lpNumberOfBytesWritten is the address of number of bytes written. The lpOverlapped is address of structure for overlapped write.

The following codes illustrate how to use WriteFile function:

```c
char lpOutBuffer[100]="@00FA0800002000000000FC000101B1000A0000017B*";
DWORD dwBytesWrite=100;
BOOL bWriteStat;
COMSTAT ComStat;
DWORD dwErrorFlags;
ClearCommError(hCom,&dwErrorFlags,&ComStat);
PurgeComm(hCom,PURGE_TXCLEAR|PURGE_RXCLEAR);
bWriteStat=WriteFile(hCom,lpOutBuffer,dwBytesWrite,&dwBytesWrite,NULL);
```

The prototype of ReadFile function is as follows:

```c
BOOL ReadFile(HANDLE hFile,
    LPUINT lpBuffer,
    DWORD nNumberOfBytesToRead,
    LPDWORD lpNumberOfBytesRead,
    LPOVERLAPPED lpOverlapped);
```

The member definition of the ReadFile function is similar with the WriteFile function. The following codes illustrate how to use ReadFile function:

```c
char str[50];
DWORD wCount;
BOOL bReadStat;
bReadStat=ReadFile(hCom,str,50,&wCount,NULL);
```
The performance analysis of communication

From long time testing, the serial communication between host computer and sorting machine is very efficient, reliable and flexible. The button capture one on the dialog window controls the operation of the software. The dialog window is shown as Fig3.

Fig3 The software window

Under the above configuration of serial ports, WriteFile function costs about 4ms. ReadFile function costs 3ms. Without image processing, once loop time is only about 16ms, make great contribution to increase the performance of the whole system. With the help of ActiveX controls, the documents are used to watch the data for debugging.

Conclusion

The design and implementation of serial communication between the automatic Titanium alloy sorting machine and the host computer completely meets the requirements of system’s performance and function. The operation of the program is proved to be efficient, reliable and flexible.

Acknowledgements

This work was financially supported by the National High Technology Research and Development Program of China (863 Program) under grant 2011AA040102 and 2012AA041701.

References

[1] Wang Cheng-hu: Serial Communications Programming in Windows 95 with API Functions, Wuhan University Journal of Natural Sciences Vol. 5 No. 3 2000, 323~328

[2] Shuleng Dong: Serial Communication between PLC and Host Computer in Spatial, Applied Mechanics and Materials Vols. 190-191 (2012) pp 168-172, © (2012) Trans Tech Publications, Switzerland doi:10.4028/www.scientific.net/AMM.190-191.168, Carport (Trans Tech Publications, Switzerland 1987).

[3] Zeng Xianchao and Wu Shide, serial communication program design between the PLC and the host-computer, Process Automation Instrumentation, 2002, Vol.17, pp.51-53.

[4] Jiang Min, Ma Xudong, Serial communication of tel-monitoring on elevator model Based on PC, Control & Automation, 2001, vol.17, pp.13-15

[5] Dong Shuleng, Mao Hongwei, PLC Serial Communication in the Intellective Control system of The Slicing Machine, Instrumentation Technology, 2009, July, pp 19-21.

[6] Berry John Thomas. The Waite Group's C++ Programming. Beijing: Xueyuan Press, 1994.

[7] Mirho Charles A, Terrisse Andre. Windows 95 Communications Programming. Beijing: Tsinghua Press, 1997.

[8] Information on http://www.msdn.microsoft.com