Journal of Electronics and Information Science(2017) 2: 21-29
Clausius Scientific Press, Canada

A Method of Remote Multi-Machine Program Update Based on Wi-Fi Wireless LAN and CAN Bus

Congzheng Ni, Mingyu Gao, Yu Zeng*, Jie Wang

Hangzhoudianzi University, HangZhou, China
E-mail: 15869004522@163.com

Keywords: Boot loader; CAN Bus; Remote transmission; Online update.

Abstract: We provide a remote online program update method for multi-machine control system in distributed industrial control field based on WiFi Wireless LAN and CAN bus in this paper. On this basis to achieve the accurate and efficient update for remote multi-machine.

1. Introduction

At the present stage, program updating in complex industrial environment has many problems such as process cumbersome, poor stability, meanwhile, The increase in the number of machines poses a great challenge to the efficiency of the update. In recent years, although there are some multi-machine Application online upgrade method, but the boot program upgrade are basically a single machine update. The Application's online upgrade method is also mostly based on a single bus, so that the program update should use a dedicated adapter to connect to the bus to update the program, or cannot accurately update a specific device, or because of the huge amount of code makes the update time greatly extended. At present, as the automation and the intelligent system has gotten more and more development on the machining field, the demand for high compatibility, high reliability, low cost and compactsize has been constantly increasing[3]. According to the shortcomings of the existing technology, based on the in-depth analysis of the boot loader mechanism and develop a custom transmission communication protocol, we put forward a remote multi-machine program reliable update method based on Wi-Fi wireless LAN and CAN bus in this paper.

2. The structure of system

Program remote transmission system is constituted by the terminal machine, data transmission bus, data transmission and relay, data server (PC) . The hardware equipment used in the system is the communication network equipment, it mainly includes the host computer, the router, the switch, the RS485 / Ethernet converter, the RS485 / WiFi conversion module, the network cable, the data acquisition relay control board, data terminal control board and so on. The lower computer mainly through RS485, CAN and other bus way to achieve the program to be received. These two kinds of bus is commonly used in industrial communication which can be articulated many nodes, have long communication distance, high transmission rate, anti-interference ability and high reliability. WiFi wireless LAN is a industry standard for wireless network communication defined by IEEE(IEEE 802. 11). It is a very low-cost and high-speed wireless air interface, because the WiFi frequency is free without any telecom operating license[1]. The WiFicommunication functions integrated in the relay control board, the board relay through WiFi wireless LAN or Ethernet via trunk routing connected to the host, based on TCP / IP[5] transmission protocol and a reliable transmission of the verification protocol data flow, to achieve the collection of procedures summary. The overall architecture of the system is shown in Fig.1

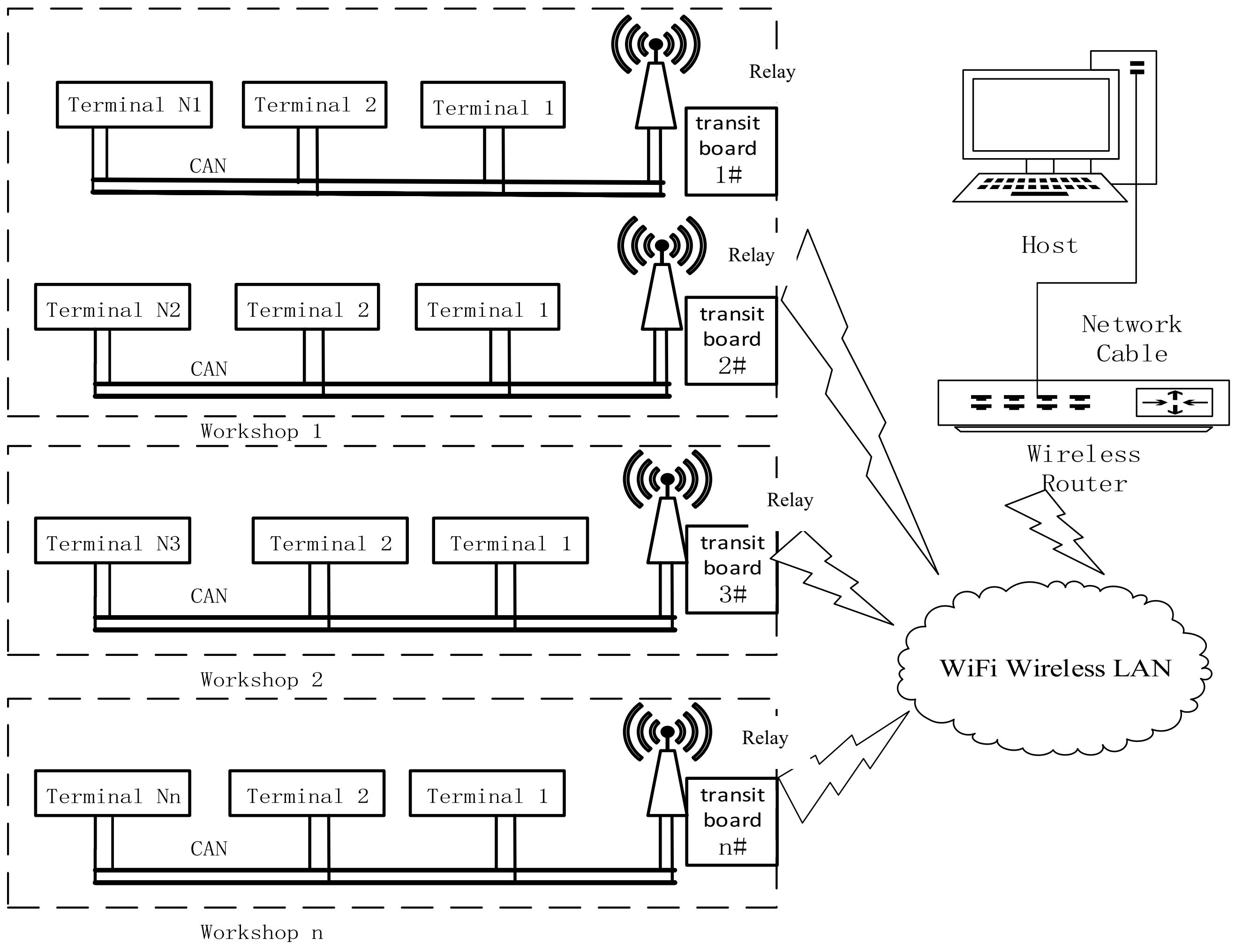


Fig.1. Block diagram of the distributed industrial control system

3. The design of the host computer software

3.1. The host computer handshake flow.

The host computer program is written in C# language, through the user program binary file traversal analysis, it packaged data into data frame which based on CAN bus protocol, then send it to the bootloader program of terminal machine, bootloader program will receive the data frame and store it in the designated user code area of FLASH memory.

When the host computer program is turned on, the first issue is sending the query command to get the relay control board number and its status, then select the bin file which need to be sent to start sending data frames, meanwhile deal with the failure of the frame transition and set re-transmission. When all data frames are received, send the end frame to complete the update. The host computer handshake flow is shown in Fig.2.

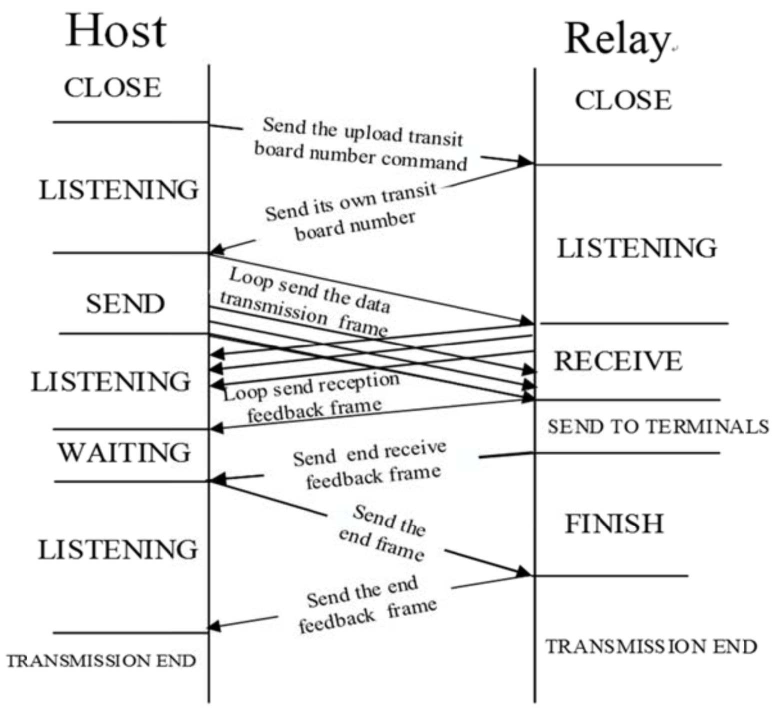

Host
Relay
CLOSE
LISTENING
SEND
LISTENING
WAITING
LISTENING
TRANSMISSION END
CLOSE
LISTENING
RECEIVE
SEND TO TERMINALS
FINISH
TRANSMISSION END
Send the upload transit board number command
Send its own transit board number
Loop send the data transmission frame
Loop send reception feedback frame
Send end receive feedback frame
Send the end frame
Send the end feedback frame

Fig.2. Host computer handshake flow

3.2. Reliable transmission verification protocol communication frame format.

The host computer software communication frame format and terminal frame format are based on reliable custom communication protocol to ensure the stability of the communication data.

Table1. Program update start frame (Host ->Relay)

Header	*	Control word	Code length	Sum check bit	Tail	CRC check bit
AA BB		0XCA	3B	2B	FF EE	2B

Program update start frame: The program need to compare the existing code of terminal machine and the original code whether the terminal machine needs to be updated (the first update and verification failed update). If the match is consistent, means the code has been downloaded, it will automatically skip rather than re-download.

Table2. Program update handshake frame (Relay->Host)

Header	Relay number	Control word	Tail	CRC check bit
AA BB	1~n(2B)	0XCA	FF EE	2B

Program update handshake frame as shown in Table 2: After Checking, the host will receive this frame and then send the data transmission frame 1s later.

Table3. Data transmission frame (Host -> Relay -> Terminal)

Data segment	Sum check bit	Segment number
10K	4B	1B

Data transmission frame as shown in Table 3: After the send start, the host uniformly transfer the Application code snippet, each transmission has 10K +5 bytes, the first 4 bytes are sum check bytes, the fifth byte is the data segment number which is used to identify the success or failure of this data segment, last 10K bytes are the data segment, each segment transmission interval of 500ms.

Table4. Application segment reception feedback frame (Terminal-> Relay)

Header	Relay number	Control word	Segment number	Success/failure flag	Tail	CRC check bit
AA BB	1~n(2B)	0XCF	1~n(2B)	0/1	FF EE	2B

Application segment reception failure feedback frame as shown in Table 4: Which is used to ensure the correct transmission of the segment on the terminal, the control word is 0XCF. If this frame is received, it indicates that the transmission terminal has a transmission error, and the relay needs to inform the host to resend this data frame.

Table5. Application segment reception feedback frame (Relay -> Host)

Header	Machine number	Control word	Segment number
BB AA	1~n(2B)	0XCF	1~n(2B)

Application segment reception feedback frame as shown in Table 5: Used to ensure the correct transmission of the segment. The control word is 0XCF, if the feedback of the success / failure flag is 0, it indicates that the transmission terminal has a transmission error, host need to resend this data frame.

Table 6. Application end receive failure feedback frame (Relay -> Host)

Header	Machine number	Control word	Tail	CRC check bit
AA BB	1~n(2B)	0XDD	FF EE	2B

Application end receive failed feedback frame as shown in Table 6: Used to ensure the correct transmission of all procedures, the control word is 0XDD, If this frame is received, it means the program verification does not pass, transmission failure, host need to resend the whole data frame from scratch.

4. The design of bootloader structure

Bootloader is called boot loader in embedded systems. The boot is a small program to run after power up or reset the system, this program leads the hardware environment of the system to an appropriate state to prepare a good fit for the final calling to the application environment[2].

Order by the interrupt vector number of the interrupt source, the interrupt vector table is a section of the storage area where the interrupt program entry address is stored. The start address of the program is given by the interrupt vector, so the re-positioning of the interrupt vector is an important step in designing the bootloader.

As shown in Fig.3, according to the requirements, the FLASH[4] will be divided into bootloader program area and user program area. The bootloader starting position is 0X08005000, store 0x2B000 bytes of content, including bootloader start code and the corresponding interrupt vector table. The user program starting position is 0X08030000, store 0x50000 bytes of content, including the APP code and the corresponding interrupt vector table.

The interrupt vector relocation means the original default start address in the 0X08000000 of the user program will be change to the 0X08030000. When chip reset, the first step is execute the bootloader program in address 0X08005000, after the completion of verification, the program read the interrupt vector table's firth two bytes, the first byte is the initial value of MSP, which is used to store the top stack address to initialize the program stack pointer. The second byte for the program new start address, through the PC pointer jump to the after-reset the vector address 0X08030000.

At the same time, in the terminal APP programming, during system initialization, the program's interrupt vector start address register SCB-> VTOR is set to the new start address of the current program via the function NVIC_SetVectorTable (NVIC_VectTab_FLASH, 0x5000).

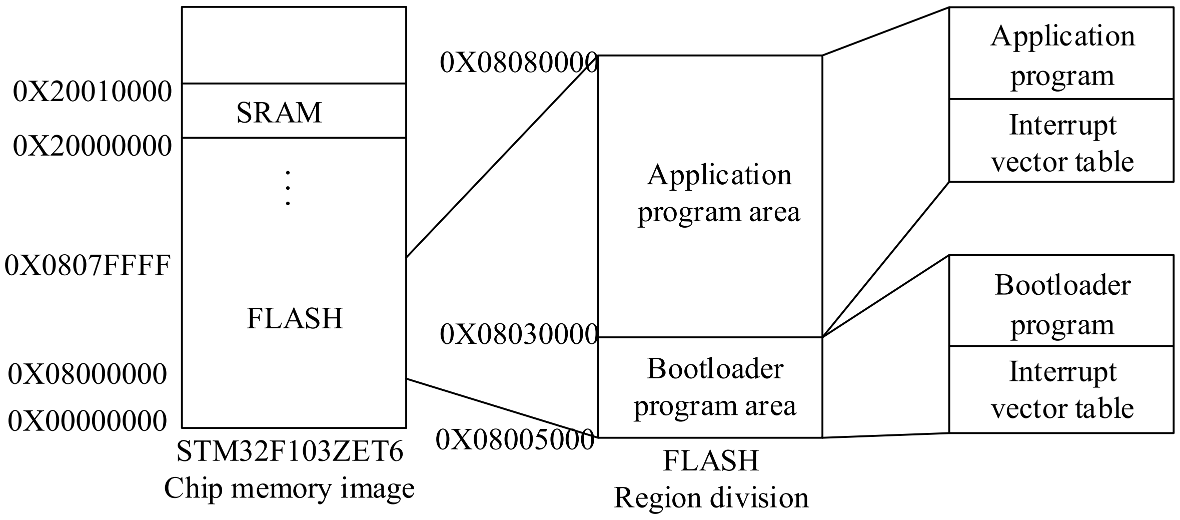


Fig.3. Interrupt vector table relocation diagram

5. System process framework and software implementation

Multi-machine program upgrade method overall process includes the following steps.

5.1. Application program update flow.

Step 1: After the system is powered on, the online ala number will be displayed in the host. User will decide to update part or all of terminal machine under this relay. If the current state is changed into the upgrade mode, the terminal machine under relay will receive the upgrade order sent by host.

Step 2: User select the appropriate file copy to the host to decide whether update the part or total application, or update the bootloader program. In order to shorten the application update time, usually some of the relatively independent and read-only high-volume data of application will be allocated in a stable address range, named as block A, and other program data and code is allocated in another address range, named as block B. If the application is updated once, the follow-up program update can modify the program only choose to update B part of the program.

Step 3: All the terminal machine automatically enter the bootloader and verify the application after powered-on, if pass the check, bootloader jump into the application’s start address, otherwise wait for re-update.

Step 4: When the terminal machine is running the application, if the host has performed steps 1 and 2, it will receive the update command forwarded by the relay, reset itself automatically and enter the bootloader, wait for the update. When the terminal receives the host command to prepare update, it will compare the program’s own check code and the update program check code, if the two check codes are inconsistent, means the reset command is ready to be updated. If two checksums are equal, indicating that the terminal has upgraded the program, the update command will not executed. So that user can avoid update the machine which has been updated under the same relay, reduce the programming times of terminal machine’s program memory .

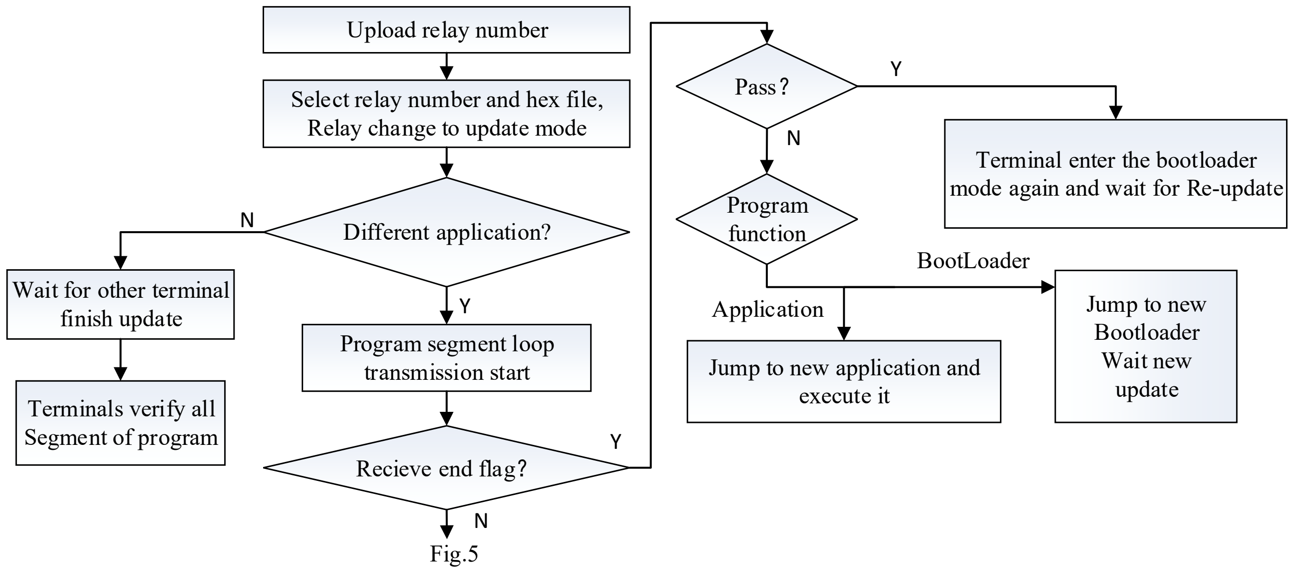

Upload relay number
Select relay number and hex file, Relay change to update mode
Different application?
N
Y
Wait for other terminal finish update
Terminals verify all Segment of program
Program segment loop transmission start
Recieve end flag?
Y
N
Fig.5
Pass?
Y
N
Terminal enter the bootloader mode again and wait for Re-update
Program function
BootLoader
Application
Jump to new application and execute it
Jump to new Bootloader Wait new update

Fig.4. System prepare and finish update flowchart

Step 5: After the update starts, the host will split the program into multiple data to be sent, each segment the relay received will be verified. If the verification is wrong, then relay will request the host to re-send this segment, otherwise the program will be sent to the terminal machine under this board through the CAN bus.

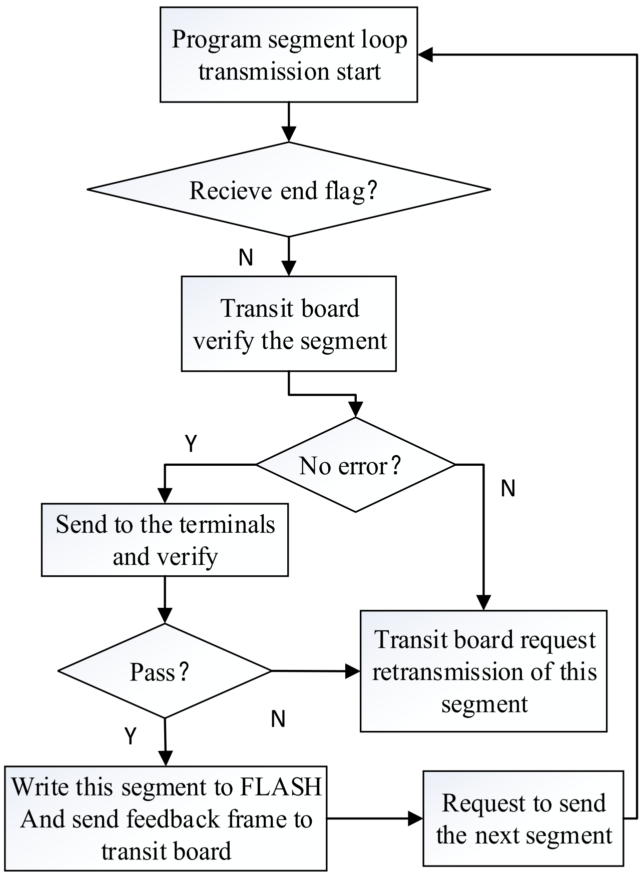

Program segment loop transmission start
Recieve end flag?
N
Transit board verify the segment
No error?
Y
N
Send to the terminals and verify
Pass?
N
Y
Transit board request retransmission of this segment
Write this segment to FLASH And send feedback frame to transit board
Request to send the next segment

Fig.5. Program segment transmission flow chart

Step 6: When the terminal machine receives the data, each segment sent from relay will be verified, if the verification is wrong, then request to re-send this segment, otherwise the program will be written to the terminal machine, then request to send next segment.

Step 7: Host determines whether the application has been totally sent, if done then send the end command of update, otherwise repeat steps 5 and 6.

Step 8: When relay receives the end command, it will reset and turn to work mode, meanwhile, the terminal machine verify all the code again, then jump to the start address if the verification is passed, otherwise report the corresponding terminal machine number to the host, repeat steps 1 to 7. The total program segment transmission flow is shown in Fig.5.

5.2. Bootloader own program update flow.

As shown in Fig.6,the bootloader update flow has follow steps. Step 1: Firstly, user need to update the terminal machine application to a specific bootloader write program. This program is similar to the normal bootloader program, but its program addresses are different from where normal application program stored. The normal bootloader program address range named segment A, normal application program address range named segment B. The bootloader write program address will be saved into segment B, update the application program(the new bootloader program)will be saved intosegment A. Afterselectingthe bootloader write program file in step 2, follow steps 3 through 8 to complete the update. After the flow is complete, the terminal machine will start the bootloader write program, continue to wait for the new bootloader program update.

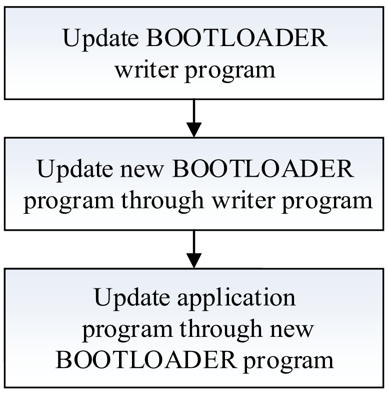

Update BOOTLOADER writer program
Update new BOOTLOADER program through writer program
Update application program through new BOOTLOADER program

Fig.6: Bootloader update flowchart

Step 2: Update the real bootloader program, then select the new bootloader program update file, repeat steps 3 to 8 operations to complete the update .

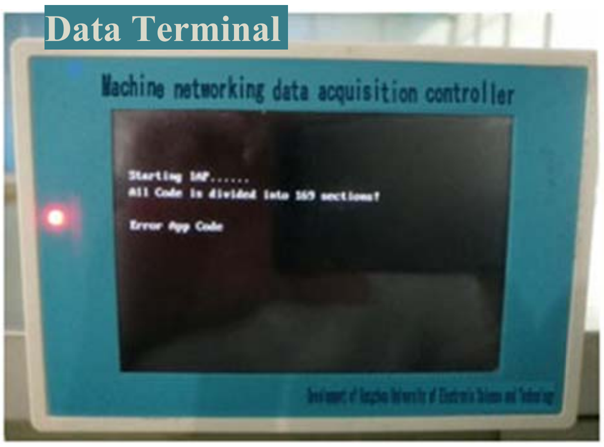
Step 3: After completing bootloader program update, user will need to update the user application of the terminal machine again.

6. Workshop specific operation flow

In the workshop debugging, according to the needs of client and workshop conditions, the terminal machine should be connected through the CAN bus as the units under relays. Place the industrial wireless router at the right place on the workshop and connect all WiFi launcher of relays to it. While the router connected to the Internet, the host which connected to the router will be enable to update the program of terminal machines.

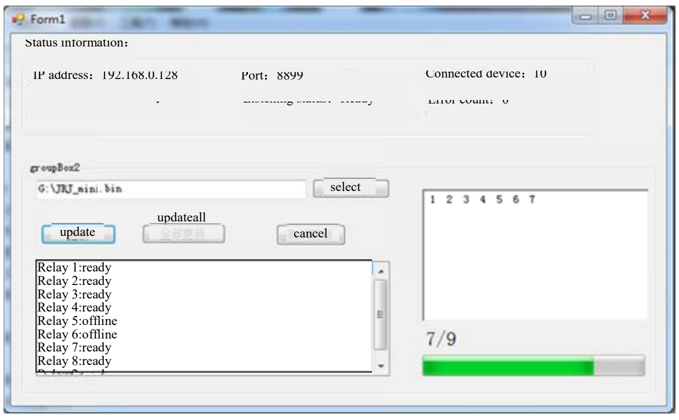
6.1. User application update.

After the system is powered on, open the update client program in the host, the online relay number will be displayed to the online list, according to the needs, user select the units to update, the selected units will suspension of the current work and turn to update mode, At the same time the terminal machine under the selected relay will receive the command. Terminal machine's listening status shown in Fig.7.



```
Starting IAP......
All Code is divided into 169 sections!

Error App Code
```

Fig.7. Terminal machine's listening status



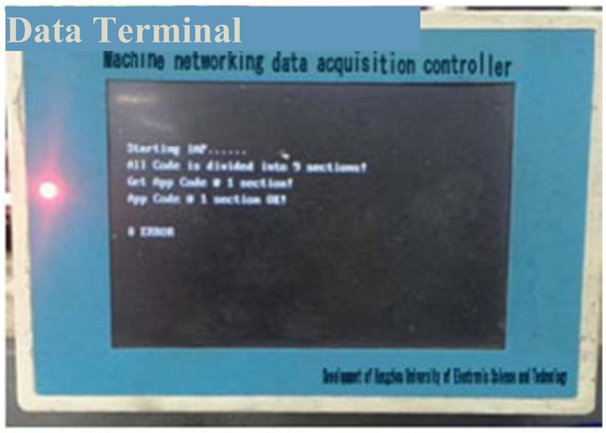
```
Status information:
IP address: 192.168.0.128        Port: 8899        Connected device: 10
                                                   Error count: 0
groupBox2
G:\JRJ_mini.bin
Relay 1:ready
Relay 2:ready
Relay 3:ready
Relay 4:ready
Relay 5:offline
Relay 6:offline
Relay 7:ready
Relay 8:ready

1 2 3 4 5 6 7
7/9
```

Fig.8. Host program update interface

Before clicking to start the update, the user need to be selected the required bin file from host, then the host read the bin file, split and convert it to the data stream format, divide into multi-segment data, then send to the terminal machine through the CAN bus,the interface is shown in Fig.8.

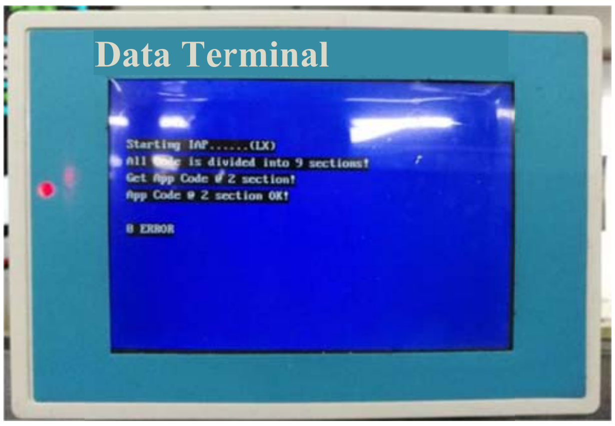
The terminal machine verified each section received and decide whether the request command should be sent back to the host to re-send this segment, otherwise the program will be written to the terminal machine, then request to send the next segment.

Fig.9 shows that the interface in terminal machine screen which displays the total number of segments divided, the segment number that is currently being transmitted, and the flag indicates whether the current segment's transmission is successful.



```
Starting IAP......
All Code is divided into 9 sections!
Get App Code # 1 section!
App Code # 1 section OK!

0 ERROR
```

Fig.9. The data begin to update interface



```
Starting IAP......(LX)
All Code is divided into 9 sections!
Get App Code # 2 section!
App Code # 2 section OK!

0 ERROR
```

Fig.10. The bootloader update mode interface

When all the segments are transmitted, the relays will receive the flag to the end of the update and then resume normal operation. At the same time, the re-transmission command will be sent to the host and repeat the above update flow if relays receive the application segment reception failure feedback frame from terminal machine.

After the terminal application program check through, reset to the normal working state, return to the working interface, this update flow end.

6.2. Bootloader program update flow.

Firstly, user need to first update the terminal machine application to a specific bootloader writer program. After the update is complete, the terminal machine will start the bootloader write program, continue to wait for the new bootloader program update. The bootloader update mode interface is shown in Figure 10. Then, select the bin file of new bootloader program, repeat the update process steps to complete the bootloader program upgrade.

Finally, after the completion of the bootloader program upgrade, if user want to let the terminal machine turn to working mode, repeat the application update process.

7. Conclusion

The characteristics of multi-machine program update technology enabled users to update application program of part or all the units in long distancethrough Internet remote rather than staying in the workshop, facilitates the management and upgrading of discrete distributed systems, saves the maintenance costs. The use of relay and the terminal machine double-check mode, ensured the accuracy of high-volume data transmission. The system can re-update a separate unit which has transmission error, to avoid the situation that other unit repeat the update flow. Using sub-section update method, which can solve the problem that the application program size beyond the terminal machine memory capacity so that it can not be updated at one time. The use of program segment design, segmented the terminal storage area, only to update modified part, to avoid the same content is updated several times, greatly improving the efficiency and success rate of the update.

Acknowledgements

This project is supported by Science and Technology Innovation Program for College Students of Zhejiang Province (2016R407066). and has been patented (NO. 201610840480. 1).

References

[1]Wang Jianguo, GuoZhiqiang, Gao Ming, Li Kun, A WiFi Based IAP Technology for SOPC Systems[A].Proceedings of the 8th International Conference on Measurement and Control of Granular Materials[C]. Northeastern University: 2009: 6.

[2]Tingqing Tan, Hanhan Tang, Yaling Zhou. Design and Implementation of Bootloader for Vehicle Control Unit Based on Can Bus[A]. Society of Automotive Engineers of China、International Federation of Automotive Engineering Societies. Proceedings of the FISITA 2012 World Automotive Congress--Volume 6: Vehicle Electronics[C]. Society of Automotive Engineers of China、International Federation of Automotive Engineering Societies: , 2012: 11.

[3]Zhang Ying . An Intelligent Instrument Based on Controller Area Network (CAN)[A]. IACSIT、IEEE China Council、IEEE Beijing Section、Sichuan Computer Federation. Proceedings of 2010 3rd IEEE International Conference on Computer Science and Information Technology VOL. 3[C]. IACSIT、IEEE China Council、IEEE Beijing Section、Sichuan Computer Federation: , 2010: 4.

[4]Jie Yan, XiaoSu Xu, GuoLong Zhang, LiHui Wang, XiXiang Liu Key. Design of Secondary Bootloader for Embedded System based on DSP[A]. IEEE. Proceedings of 2011 4th IEEE International Conference on Computer Science and Information Technology(ICCSIT 2011) VOL02[C]. IEEE: 2011: 4.

[5]Kanwalvir Singh Dhindsa, Parminder Singh and Dr. Himanshu Aggarwal. Performance of TCP/IP over wireless Networks using unaware Approach[J]. International Journal of Computer and Network Security, 2010, 21.