CDT Communication Protocol Realization based on STM32

Teng Gao[1,a *],Ji-Yan Zou[1], De-Quan Wang[1], Jin-Hua Ding[1], and Zhen-Yuan Si[2]

[1]Dalian Polytechnic University, Dalian, 116034, China

[2] Henan Senyuan Electric Co., Ltd. Changge, 461500, Henan, China

[a]email gaoteng@dlpu.edu.cn

Keywords: Serial Port; DMA; STM32; CDT protocol; IEC61850

Abstract. In industry 4.0 society, the development of smart grid requires to realize the conversion of substation equipments from CDT protocol to IEC61850 protocol. CDT (Cyclic Digital Transmission) protocol provides the function, the frame structure, the information word structure and transfer rules in the grid data collection and monitoring system. In this paper, the CDT protocol that uses STM32 as the master chip is developed, in order to realize a high degree of automation in the grid scheduling and a stable and reliable communication. Some technologies and functions of STM32 such as DMA, USART, Timer, and so on are applied into the design. The results show that CDT protocol can be applied in the communication between STM32 and peripheral, and using the conversion board to implement the link of CDT and IEC61850 is feasible.

Introduction

At present, substation of electric power system in China is in the process of intellectual upgrade and alteration[1]. The intelligentized work of switchgear is also an urgent issue need to be solved. As a result, the communications between the switchgear monitoring system and dispatch center are indispensable. Currently, the international standard is the IEC61850[2] protocol while CDT[3] protocol is the most popular protocol in China. CDT is the electric power communication protocol based on serial port communication, it is the most widely used protocol in domestic power grid that can meet the requirements of reliability and real-time performance of telecontrol information in the power grid dispatching safety monitoring system in China[4]. This protocol stipulated the transfer rules, the information field structure, the frame structure and function, and so on in the grid data collection and monitoring system[5]. From an economic point of view, the possible solution is to use relay equipment to implement the conversion and links between CDT protocol and IEC61850 protocol. In this case, traditional substation devices can connect to the management computer in the control center far away through the relay equipment. Thus, it is necessary to develop communication software based on CDT protocol between switchgear monitoring system and relay equipment.

System outline

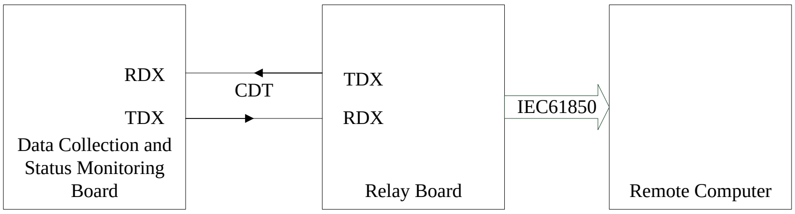


Figure.1 System Structure Diagram

In the modular design, the relay equipment is a normal module, which uses the serial port as the interface to communication with the monitoring system. The structure diagram is described in Fig.1.

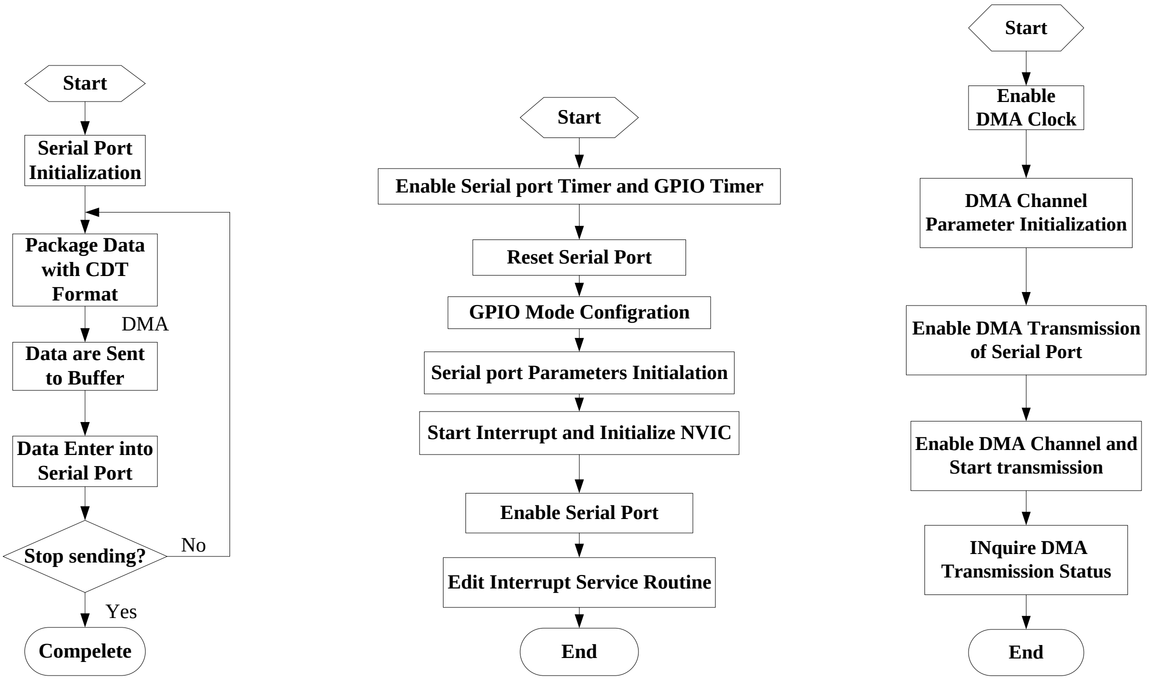
Data collection and status monitoring board is defined as the master computer while the relay board acts on the slave computer in this design. The two parts are connected through the serial port, by which the CDT protocol is applied into serial communication.

Interface hardware design

Based on the functional needs, this system chose STM32F107 as the main controller[6]. The STM32[7] is a family of high performance, lowpowered and low cost microcontrollers[8]. In the master-slave control system, the subordinative controller can exchange data though RS485 or CAN bus with the host controller [9]. USART within STM32F103ZET6 supports three application methods such as query, interruption and DMA(direct memory access). If using DMA mode, high-speed data communications can be achieved[10].

SWM61850_MINI protocol conversion card is chosen for protocol conversion. The SWM 61850_MINI protocol conversion card is installed between a remote master computer and a slave computer, which provides an Ethernet communication interface with integrated IEC61850-8-1 communication protocol to the former and a serial port to the latter.

Communication software design


Start
Serial Port Initialization
Package Data with CDT Format
DMA
Data are Sent to Buffer
Data Enter into Serial Port
Stop sending?
No
Yes
Compelete
Start
Enable Serial port Timer and GPIO Timer
Reset Serial Port
GPIO Mode Configration
Serial port Parameters Initialation
Start Interrupt and Initialize NVIC
Enable Serial Port
Edit Interrupt Service Routine
End
Start
Enable DMA Clock
DMA Channel Parameter Initialization
Enable DMA Transmission of Serial Port
Enable DMA Channel and Start transmission
INquire DMA Transmission Status
End

(a) (b) (c)

Figure. 2 The flowchart of communication

From the hardware structure, it can be affirmed that the data collection board serve as a slave computer while the remote computer acts as the master computer. For the slave computer, the primary work is to execute the initializations of STM32, including the serial port, DMA, timer, etc. After initialization, the data are packaged according to CDT format. Then, the packets are pushed into the transmission buffer. After the timer is time out, the packets are forwarded to the serial port with the communications to the host computer through DMA. The flowchart is set out in Fig.2(a).

Time Delay Function Initialization. The SysTick timer in the Cortex - M3 kernel is a 24-bit count down timer. When the count equals 0, the initial value will be loaded from RELOAD register automatically. Here, STM32 internal SysTick is used to carry out the delay. In this case both interrupts and system timers are not occupied. Two functions, delay_init is used to initialize the parameter fac_ms, at the same time the SysTick Clock source is selected to be the external clock; delay_ms is the time delay function in milliseconds.

Interrupt Priority Set Up. Interrupt grouping needs to be set up before system begins to run. First of all, use function NVIC_PriorityGroupConfig() to confirm the group number. That is, assigning bits of the preempting priority and the subpriority. This function can only be called once, moreover, the group cannot be changed as soon as it was confirmed. After that, the interrupt priority level of each interrupt is set through the function NVIC_Init(). This function contains a number of important parameters, i.e., the NVIC_IRQChannelPriority is used to define the preemption priority level, sub-priority level is determined by NVIC_IRQChannelSubPriority, and NVIC_IRQChannelCmd can enable interrupt.

Serial Port Initialization. Serial port communication can be implemented through SCITXD, SCIRXD and earth wire. The initialization process is shown in Fig.2(b). The general steps can be explained as follows.

Step 1: Serial port clock enable; GPIO clock enable. According to the STM32 system architecture diagram, serial port 1 is the peripheral which is connected to the APB2. The function that enables serial port and GPIO Port is RCC_APB2PeriphClockCmd().

Step 2: Serial port reset. When an error occurs in the peripheral, reset setting can recall the peripheral operation. After that, the peripheral and related parameters can be re-configured, so the system runs again. Generally, in the system initialization, a reset of the peripheral operation is implemented once, and the related function is USART_DeInit().

Step 3: GPIO mode setting. GPIO mode should be set to full-duplex serial port. That is, the TX pin should be configured to push-pull output, while RX pin is defined to float empty input or pull-up input, and IO port transfer rate is set to 50Hz.

Step 4: Serial Port parameter initialization. The serial port parameters can be initialized through using the function USART_Init(). This function contains two different parameters, the first parameter is used to determine the serial port number; the second parameter is a structure pointer whose member variables contain number of parameters, such as Baud Rate, Data Bits, Stop Bits, Parity Bit, receive mode, and so on. In general, Baud rate is set to 9600, but it is not absolute. Each packet includes the start and stop bits, data bits, parity bit. The actual data depends on the communication protocol, and the stop bit not only means the end of the data, but also offers an opportunity for the computer to synchronize clock in the asynchronization communication. Parity bit is commonly used in the communication between devices to verify the correctness of the data.

Step 5: Enable Interrupt and initialize NVIC. USART_ITConfig() is used to enable serial port interrupt. There are two entry parameters on this function. The first one indicates the serial port type, while the second one refers to the type of the interrupt.

Step 6: Enable serial port. Use the function USART_Cmd to enable Serial port. After configuring serial port and IO port, the data can be sent and received only if the serial port is turned on.

Step 7: Edit the interrupt handling function. The flag bit RXNE is utilized to decide whether to receive the data from serial port. If serial port receive interrupt has been set, the frame of the data that has been received can be parsed according to CDT protocol.

DMA initialization programming. When DMA is configured, RCC_AHBPeriphClockCmdis called to enable DMA clock above all. Next step, the main parameters in DMA channel such as memory address, peripheral address, data length, data width, and channel priority should be initialized. Here, the base address of the peripheral is the serial port, and the memory base address is the memory address special for storing DMA data. The data transfer direction is determined according to the actual. The serial port DMA transmission is enabled after DMA configuration, which is performed through the function USART_DMACmd. Then, use the function DMA_Cmd() to enable DMA channel and start a USART DMA transmission. Finally, query DMA transfer status. DMA initialization process diagram is given in Fig.2(c).

Timer initialization programming. The general purpose timer of STM32 contains a pre-programmed 16-bit automatically load counter that is guided by a prescaler. RCC clock controller prescaler associated with timer prescaler can achieve pulse length and waveform cycles from several micro-seconds to a few milliseconds. Any general purpose timer in STM32 can be used independently without relationship to each other. The typical steps of timer setting can be summarized in the following steps.

1.Enable timer clock. In this case STM32 Timer 3 is used. The timer 3 is mounted on the APB 1, so this function RCC_APB1PeriphClockCmd() is called to enable the timer clock.

2.Initialize the timer parameters. The important parameters in the timer initialization are automatic reload value, frequency division coefficient, counting method, and so on. If there is no automatically reload value of the timer is set, the counter will stop counting. Frequency division coefficient can be set through prescale register. The frequency of Timer 3's clock is 72MHz, hence the corresponding interrupt time can be worked out based on automatic reload value and frequency division coefficient.

3.Define the type of timer interrupt. Here the update interrupt of the timer 3 is necessary. As a result the registers of the timer will be adjusted, so that the update interrupt can be generated. The related function t can enable timer is TIM_ITConfig().

4.Set interrupt priority levels. The timer interrupt type has previously been set, so it is indispensable that NVIC related registers and the interrupt priority is set before executing interrupt.

5.Enable the timer. The timer should be enabled after having initialized timer parameters, interrupts, and priority level. TIM_Cmd(TIM3,ENABLE) is the function to enable the timer.

6.Edit the interrupt service functions. The type of interrupt can be determined through the value of the status register after interrupt. The interrupt service program can be implemented after having confirmed the type. Since the update interrupt is used here, so the lowest bit of the status register SR will set forth the status. This interrupt flag should be clear to 0. Here, the interrupt service function is to return to the main function and trigger DMA sending function.

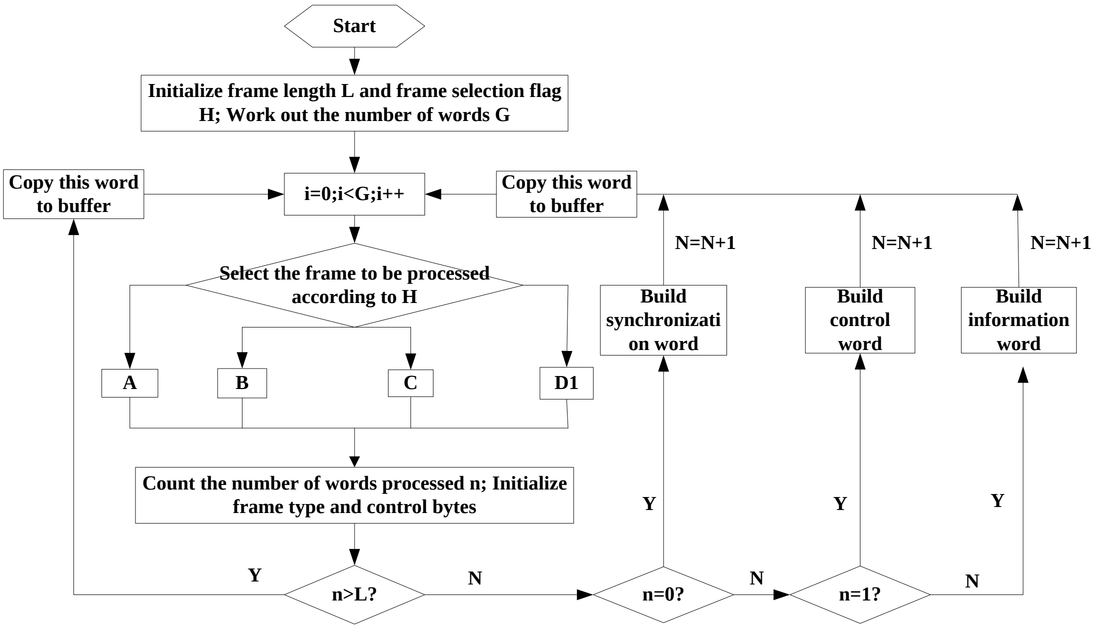

Start
Initialize frame length L and frame selection flag H; Work out the number of words G
Copy this word to buffer
i=0;i<G;i++
Copy this word to buffer
N=N+1
N=N+1
N=N+1
Select the frame to be processed according to H
Build synchronization word
Build control word
Build information word
A
B
C
D1
Count the number of words processed n; Initialize frame type and control bytes
Y
Y
Y
Y
n>L?
N
n=0?
N
n=1?
N

Figure. 3 CDT protocol framing flowchart

CDT protocol programming.

When some data have to be sent, they will be written to send buffer pSendBuf by CPU. The frame length is initialized firstly. Then the bytes to be sent are divided by 6 so as to convert to be the number of words to be written. From the first word on, whether the frame is a normal frame or not is determined primarily. If yes, call frame insertion discrimination function; otherwise, call different framing function according to the framing selection value m_byFrameSCase(for example A frame). At the end of the framing, the framing selection value adds 1 for the next frame.

If A frame function is called, count the words have been treated with m_wMessageNum. If this is the first word in the frame, initialize A frame length, frame type, and control bytes. After that, compare the number processed with the number specified in the frame. If the former is larger, the framing has been completed, and then clear the number flag of the word processed, as well as set the end flag of framing to 1. If the latter is larger, it means the framing is not finished. The subsequent function will be determined in terms of the control byte and frame type. After this function is called, the value of the words processed should be checked. If m_wMessageNum equals 0, the synchronization word(3 groups of EB90H) has to be added. At this time, synchronization word grouping function should be called. After that, make m_wMessageNum add 1 and send the data copy to pSendBuf, and then process the next word. If the m_wMessageNum is 1, the control word needs to be added, control word grouping function must be called, and then make m_wMessageNum add 1 and send the data copy to pSendBuf, and process the next word as well. If the m_wMessageNum is greater than 1, it indicates that the information words should be grouped; as a result information word grouping function is called. When the information word is telemeter or tele-signaling, the flag bit that reflects the tele-signaling and telecontrol reverse check should be check whether it is set to 1. If yes, the tele-signaling or telecontrol reverse check processing function is called. After the information word is grouped over, make m_wMessageNum add 1 and send the data copy to pSendBuf, and then process the next word. CDT protocol framing process is shown in Fig.3.

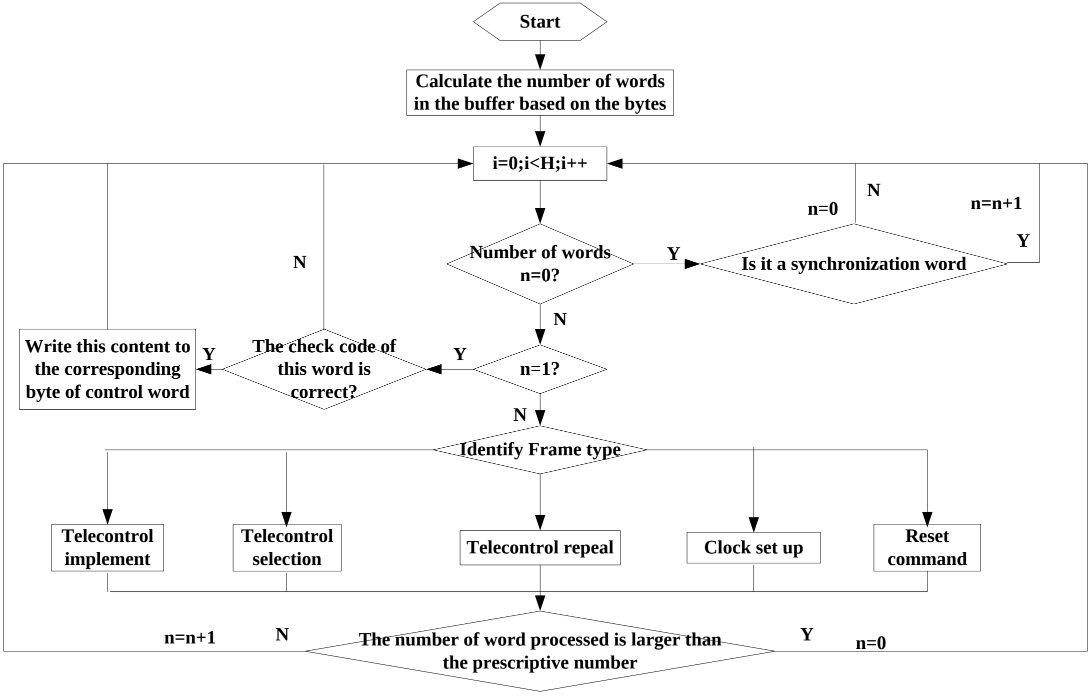

Start
Calculate the number of words in the buffer based on the bytes
i=0;i<H;i++
N
n=0
n=n+1
Y
Number of words n=0?
Y
Is it a synchronization word
N
N
Write this content to the corresponding byte of control word
Y
The check code of this word is correct?
Y
n=1?
N
Identify Frame type
Telecontrol implement
Telecontrol selection
Telecontrol repeal
Clock set up
Reset command
n=n+1
N
The number of word processed is larger than the prescriptive number
Y
n=0

Figure. 4 CDT frame parse process

A buffer zone occupied N bytes is built in the custom receive buffer pRecvBuf, in which CPU stores the data received from serial port. Since there are 6 bytes in the synchronization word, control word and information words in the frame of CDT, so N/6 words can be saved in the data buffer. The data in the pRecvBuf are checked firstly to determine whether there are 6 bytes or not. If yes, set “waiting for processing” flag to 1 as a guideline to find synchronization word. If less than 6 bytes, end subroutine and receive the data for the next cycle. If more than 6 bytes, the synchronization word discrimination function is called. This function is to extract the first 6 bytes from the words to be processed, and determine whether they are the synchronization word. If this word does not meet the structure of synchronization word, then the first byte is discarded. Then extract the following byte, and repeat the above method. The above content should be run in each cycle until the synchronization word is located accurately. After two adjacent synchronization words are located, the data between the two synchronization words are transferred and stored in the staging area of control word and information word, and "frame to be processed" flag is set. Having located the synchronization word, the program enters into control word processing function, that is, the parse of the control word of the data frame starts. The control word with 6 bytes is verified by using the CRC block. If the checksum is correct, the number of information words of the data frame can be obtained from the 3rd byte of the control word, which will be stored in the custom variable. The frame type flag is also set in accordance with the 2nd byte of the control word, so that the different telemetry, tele-signaling, and telecontrol back-check information can be identified based on the flag by information word processing function. At last, the information word processing flag will be set. If the checksum is not correct, reset the "frame to be processed" flag and start looking for the synchronization word again. Information words can be parsed according to the frame type and the information word length supplied by control word processing function. 6 bytes will be still extracted each time, and CRC checksum function is called to implement code check as well. If checksum is wrong, this information word will be discarded, and then the next information word is extracted to parse. Otherwise, the information word is retained, at the same time the different information word processing function is called based on the frame type. The first byte of the information word is the function code, through which the definite meaning of an information word can be determined. In this way, the data of the information word can be classified and extracted to be stored into specific staging area. No matter whether CRC verification code is correct or not, "information word processed" variable adds 1. When the count value of this variable is equivalent to the overall length of the information words, it means that the data frame has been fully parsed. The "information word processing" flag is also reset and the program waits for the next frame. CDT frame parse process is shown in Fig.4.

Conclusion

The protocol conversion technology of intelligent switch data acquisition system is introduced in this paper, which adopts STM32 to carry out serial port data communication and other functions, as well as realizes the CDT communication protocol. Thus, the hardware design is greatly simplified, and the cost is also saved. With the development of electric power system, this kind of intelligent electric power data acquisition system will obtain higher application value and broad market prospect.

Acknowledgements

This work was funded in part by a grant from the general program of Liaoning Provincial Department of Education Science Research, No. L2013210. This work was also partly supported by the fund of the Dalian Polytechnic University Youth grants, No. QNJJ201307.

References

[1] Y.G. Ma, J.Z. Liu, Research and realize of conversion between auxiliary system protocol and IEC 61850 in Smart Substation. 2011 Asia-Pacific Power and Energy Engineering Conference. (2011)

[2] Information on http://en.wikipedia.org/wiki/IEC61850.

[3] Information on http://wenku.baidu.com/view/fb74500abb68a98271fefa8c.

[4] S.H. Chen, Research on software realization of protocol conversion between CDT and IEC60870-5-101, Water Power, 34(8): 105-107. (2008)

[5] Y.C. Ren, T. Xing, Y. Yao, Research on CDT Data Collection Methods and Algorithm, Microcomputer Information, 25(5-1):95-97. (2009)

[6] X.H. Li, J.B. Liu, W.Y. Cai, Design and Implementation of Web Server Control System Based on STM32, 5th International Conference on Intelligent Networks and Intelligent Systems, p 146-149. (2012)

[7] Information on http://www.st.com/st-web-ui/static/active/en/resource/technical/document

/datasheet/CD00191185.pdf.

[8] H.F. Zhang, J. Zhao, The design of RF data acquisition system based on STM32 and FPGA, 2011 International Conference on Multimedia Technology, ICMT 2011, p 832-834. (2011)

[9] Y.D. Zhang, Q.X. Zhang, J.G. Jiang, Y.J. Zhao, Q.W. Qian, D.D. Tang, D.F. Tian, The Controller Development of Multi-layer Parking Equipment Based on STM32, International Journal of Smart Home, 8(1): 303-310. (2014)

[10] H. Zhou, Realization of HJ/T212 communication protocol based on STM32 microcontroller, Electronics World, (24):36-37. (2012)