Design and Research of Time Synchronization Communication Protocol Based on CAN

Liu Hongqing

Hunan Vocational College of Modern Logistics, Changsha, Hunan, 410131, China

Email:158140027@qq.com

Keywords: Time Synchronization, CAN

Abstract: Controller Area Network (CAN) belongs to fieldbus and is one of the fieldbus standards. It is suitable for industrial control system with many characteristics such as high communication rate, strong reliability, convenient connection, high performance-price ratio and so on.It is a kind of serial communication network which effectively supports distributed control or real-time control. With its short message frame and MAC (media access control) mode of CSMA/CD-AMP (with information priority and carrier sense multiple access of conflict detection) to win the favour of equipment interconnection in the field of industrial automation. CAN can be applied to almost kind of data communication in automotive systems, machinery, technical equipment and industrial automation, ranging from high-speed network to low-cost multi-line network.

1. Introduction

CAN has the following advantages:

It is easy to use. Many CAN controllers realize most functions of CAN physical layer and data link layer. Users only need to initialize the CAN controller and send and receive data on the CAN bus to realize communication.

It has high reliability. The maximum communication rate of CAN can be up to 1 Mbps. CAN bus is a multi-master node, and each node can obtain bus control through bus arbitration. Perfect error handling mechanism ensures the safety and reliability of data transmission in high noise interference environment.

Its system has good expansibility. CAN bus is based on the encoding of sending messages, not the encoding of CAN control nodes, so adding or deleting CAN nodes will not have a great impact on the system.

CAN bus transmits information in message units and supports four different types of message frames: data frame, remote frame, overload frame and error frame. The message contains the information identifier ID, which indicates the priority of the message. Each node on the CAN bus can send messages on its own initiative. Messages on the bus are arbitrated by identifier ID, the smaller the ID value, the higher the priority. Nodes with the highest priority message win the right to use the bus, while other nodes automatically stop sending. When the bus is idle again, these nodes will automatically resend the original message. All nodes in the network can automatically decide whether to receive the message by ID. Each node has an ID register and a shielding register to receive messages, the node begins to receive messages formally only when the shielding function is the same, otherwise, it will ignore the messages of the ID. So as to make the CAN system very flexible and can expand or change the network composition arbitrarily.

C8051F series single chip computer are integrated SOC (system on chip) of mixed signal chip, which have microcontrollers compatible with the MCS-51 kernel and order set. In addition to standard 8051 Digital peripherals, they also integrate analog components that commonly used in data acquisition and control systems and other digital peripherals and functional components. It has been widely used in intelligent instruments, data acquisition, automatic control and other fields due to its small size, high integration, multi-function, easy configuration and use. Therefore, this paper adopts the C8051F120 design and CAN bus communication protocol implementation, giving full

Copyright © (2019) Francis Academic Press, UK DOI: 10.25236/csam.2019.079

play to its small size and high reliability. C8051F120 is a fully integrated MCU chip of mixed-signal on-chip system with 64 digital I/O pins (100-pin TQFP package).

2. Formulation of Data Transmission Format

Only 8 bytes can be transmitted for each sending and receiving according to CAN2.0 specification, which can meet the general requirements of control commands, industrial control status and test data in most industrial fields. However, the requirement of transmitting more than 8 bytes often occurs in practical application, which should be realized through message disassembly and splicing technology. This paper defines a simple transmission format and completes the accurate and fast transmission of single and multi-frame data on the basis of CAN2.0A.

The format of information transmission is shown in Table 1:

Table 1 Format of information transmission

Data bits	7	6	5	4	3	2	1	0
Data type, length	FF	RTR	0	0	DLC.3	DLC.2	DLC.1	DLC.0
Information identifier	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
	ID.20	ID.19	ID.18	RIR	0	0	0	0
DATA0	DATA0.7-DATA0.5			DATA0.4-DATA0.1				DATA0.0
DATA1	X	X	X	X	X	X	X	X

The meanings of each are as follows:

FF:0 represents standard format, 1 represents extended format.

RTR:0 represents data frame,1 represents remote frame.

DLC.X represents data length code bits (0-8).

ID.X represents information identifier bit.

ID.28 ~ ID.26 represents information function identification.

ID.25 ~ ID.21 represents receiving task identification.

ID.20 ~ ID.18 represents the address identification of the receiving node.

X represents reserved bits, default is 0, cannot write reserved bits.

DATAO represents the first byte of a data field, be used in this protocol to indicate the attributes of the data.

DATA0.7 ~ DATAO.5 represents the address identification of the sending node.

DATA0.4 ~ DATAO.1 represents the functional identification of data.

DATA0.00 represents the extended bits,'0' indicates that there is no need to extend when the data length is less than 8,'1' indicates that there is need to extend DATA1 as the index of transfer times when the data length is greater than 8.

When DATA1:DATAO.0 is '0', data is transferred; when DATAO.0 is '1', it is an index of the data transfer times.

The information priority designed in this paper is from high to low in order: information function identification, task function identification and target node address identification. The information function identification is set at the highest number of ID, and eight basic functions can be distinguished in some cases by three-bit function code: these functions can be node status control, node protection, emergency notification and time-marked information; receiving task identification indicates the task attribute of the frame data, its capacity is 32; target node address indicates the destination address of the data,its capacity is 8.

DATA0.0 is used as a flag bit in this protocol to distinguish single-frame transmission from multi-frame transmission, which solves the transmission problem of strings larger than 8 bytes. When the flag bit is '1', it means that multi-frame data is transmitted; when it is '0', it means single-frame data. So as to overcomes the defect that CAN can only transmit data less than or equal to 8 bytes, and achieves data transmission greater than 8 bytes.

This protocol specifies that the first byte of the data field is used as an index of the order of multi-frame data transmission in order to identify the possible re-frame and frame loss phenomena in multi-frame transmission. When transmitting data in the format specified in this protocol, a single frame can transmit up to 7 bytes of actual data: when the length of the data flow is longer than 7 bytes, it will be divided into multi-frame transmission.

3. Application Layer Protocol Design

In CANV2.0 standard, only the physical layer and data link layer of ISO reference model are specified, there are no connection units and resident media, and no application layer are specified. Physical layer is responsible for such functions as physical signal transmission, decoding, bit timing and bit synchronization, while data link layer is responsible for such functions as bus arbitration, information segmentation, data security, data confirmation, error detection, signal transmission and error control. Even when performing some very simple CAN-based distributed systems in fact, in addition to the basic two-tier services, more functions are required or desired such as sending strings longer than 8 bytes, responding or determining data transfer, identifier assignment, network boot or monitoring nodes.

Due to these additional functions directly support the application process, it can be regarded as the "application layer". If it be executed correctly, the application layer and its corresponding application layer interfaces (sub-protocols) would provide a clear definition of the boundaries between communication and application processes in order to distinguish them.In some cases that simple communication protocols can meet the requirements, the use of complex protocols will result in a waste of resources. Moreover, it is inconvenient to use and limits the flexibility of CAN. Therefore, formulating the suitable communication protocol is very important to the development and use of CAN in some cases. According to the requirement of the actual system design, the communication protocol of CAN application layer is formulated on the basis of 2.0 A technical specification in this paper.

CAN application layer protocol is mainly responsible for establishing the bridge between CPU and the bottom layer. It mainly consists of four parts: switch mechanism of nodes, data receiving and receiving mechanism, error handling mechanism and interrupt management mechanism. The four mechanisms are interrelated and mutually restrictive, and work together to maintain the operation of the system. This paper mainly introduces the key data sending and receiving mechanism because of the limitation of the space.

3.1 Data Sending Mechanism

The sending mechanism mainly realizes that the data to be sent by the CPU is received and sorted out to conform to the frame format stipulated by the application layer protocol, the disassembled packets (data frames) are placed in the circular queue in order to be sent. It is also responsible for managing and maintaining the normal operation of the circular queue. Cyclic queues are scanned periodically during timer interruption through sending mechanism. If data is found waiting to be sent in the queue, the sending function is called to send the data to the CAN bus.

A temporary buffer is created to temporarily store packets waiting to be sent at the bottom, it adopts the storage structure of circular queue and implements the first in first out management mode for data. A circular queue is a 42*11 two-dimensional array used to temporarily place the data that the CPU is about to send, and the data is sequentially arranged in a circular queue waiting to be sent. Each time one frame of data is added, the tail pointer of the circular queue is increased by '1'; each time one frame of data is sent successful, the head pointer of the circular queue is reduced by '1'. When there is no data in the circular queue, the state of the queue is 'empty', otherwise it is 'not empty'; if the head and tail pointers of the circular queue coincide and the queue is not 'empty', then the queue is 'full'. When the queue is 'full', it is forbidden to write data to the queue again; otherwise it will easily lead to data coverage or loss. Data in the queue follows the principle of first in first out. CPU loads data from the end of the queue and reads data from the head of the queue when sending data to CAN bus. The collation unit of sending circular queue is frame; each operation is take 11

bytes as unit. Before the sending mechanism is operating, first initialize the sending circular queue, assign the head pointer and tail pointer of the circular queue to zero, and assign the occupied space to zero.

CAN sending mechanism is mainly composed of two modules: packet module and frame sending module. When the CPU needs to send data, it calls the packet function to give the storage address of the data to be sent. The packet function will organize the parameters of sending node address, receiving node address, information type, task identification and data identification into the format of CAN data link layer ID according to the format specified in this agreement, and assemble the data into data frames (packets) conforming to the application layer protocol, so as to realize the packet processing of data longer than 7 bytes, according to the index number filled in. Put them in the sending circular queue in the order of the index numbers filled in and waiting to be sent. The flow chart of the packet function is shown in fig 1.

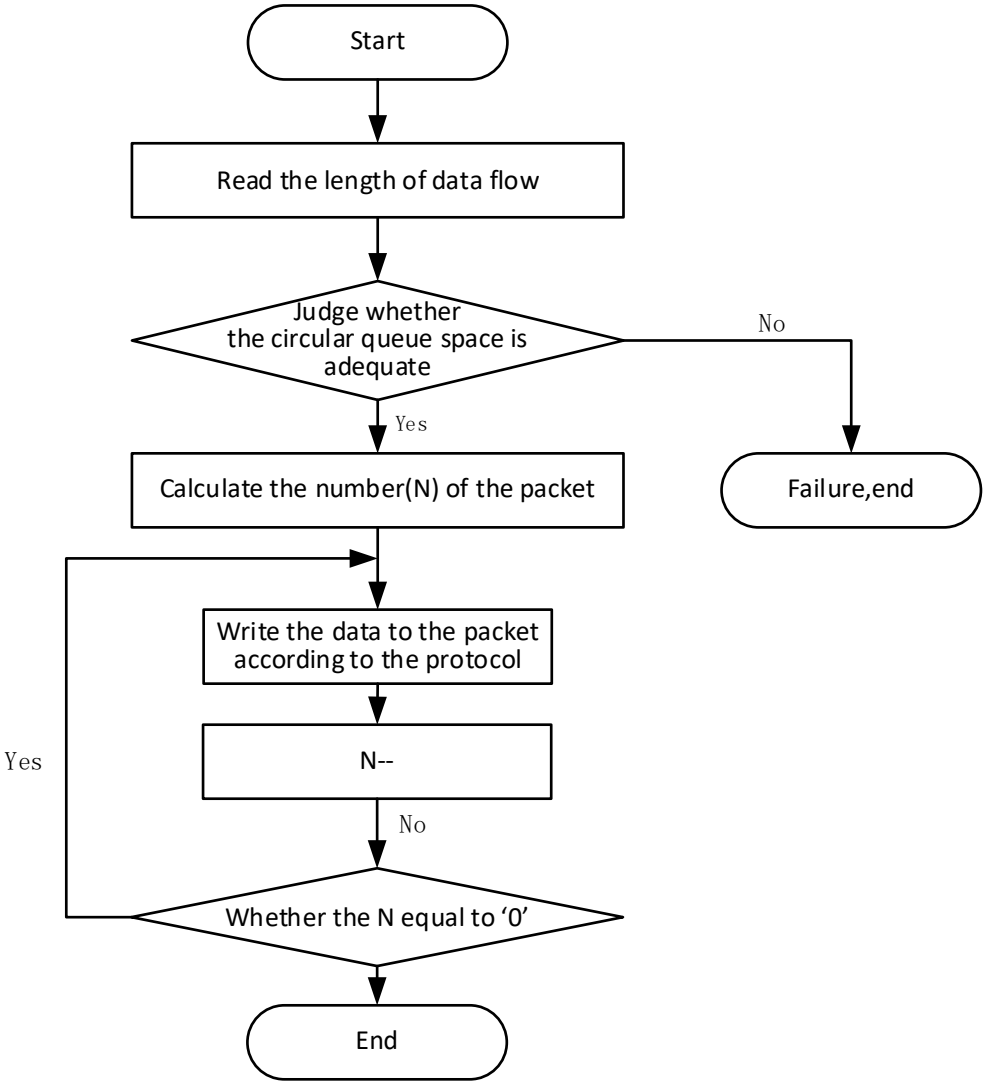

Fig.1 Flow chart of the packet function

3.2 Data Receiving Mechanism

The CAN receiving mechanism is responsible for receiving data from the CAN bus, restoring it to the format before sending according to the address of the data source node, transmitting it to the upper layer accurately and correctly, and providing interface functions for the upper layer. After the CPU takes the data away, empty the corresponding array.

A temporary buffer is created to temporarily store the data received from the CAN bus at the bottom. The temporary buffer uses a two-dimensional array format and has a capacity of 4*3*64.

The capacity of the first dimension is 4, indicating the address of the sending node respectively; the capacity of the second dimension is 3, indicating the number of packets from the same node can be stored continuously as an extended cache of data; and the capacity of the third dimension is 64, which is used to store the data after collation. Opening up a two-dimensional space can turn the protocol around generally, but CPU may not be able to remove the finished data and then receive new data from the same node in the actual system, which may lead to new data covering the original data and resulting in data loss. A three-dimensional space has been opened up in order to avoid this potential danger. In the worst case, each node can accommodate three packets of data from the same node, which greatly reduces the possibility of data loss. The receiving data flow is shown in fig.2

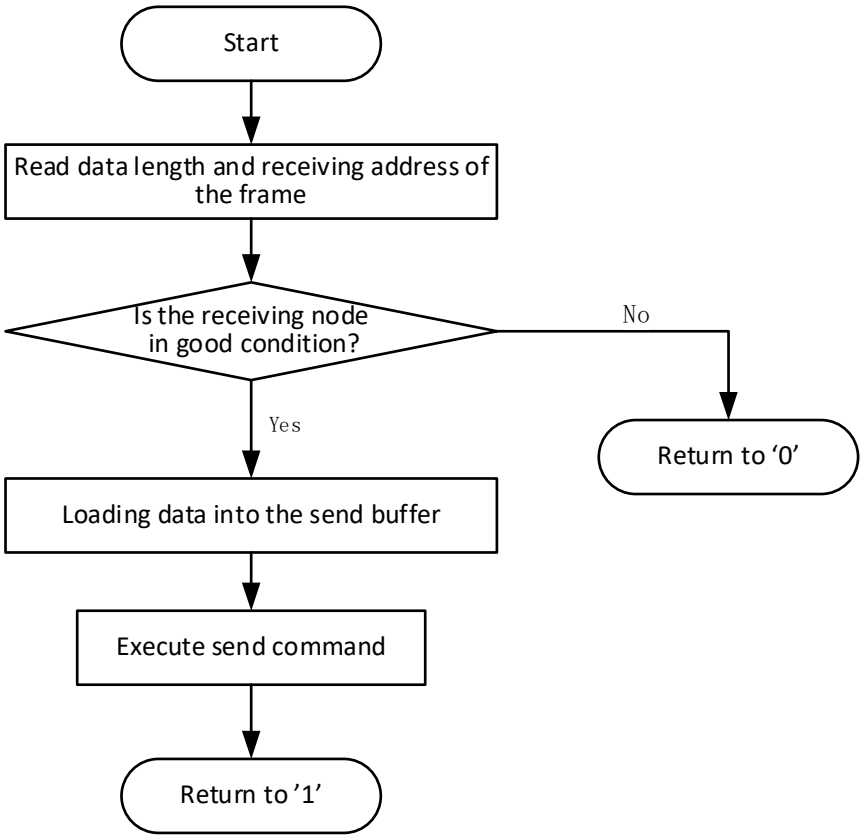

Fig.2 The receiving data flow

4. Conclusion

The innovation of this paper: CAN bus has been widely used in the interconnection of industrial process monitoring equipment, valued by industry and recognized as one of the most promising fieldbus on the basis of its excellent characteristics, high reliability and unique design. CAN protocol has been widely welcomed as a general, effective, reliable and economical platform. Users are objectively required to establish high-level protocols to improve CAN due to the inherent limitations of CAN2.0 specification. The CAN bus application layer protocol designed in this paper has been put into use, which is simple, flexible and easy to transplant.

References

[1] Yang Xianhui, Fieldbus technology and application, Beijing, Tsinghua University Press, 1999

[2] Wu Kuanming, Selection of fieldbus technology application. Beijing, Beijing University of Aeronautics and Astronautics Press, 2003

[3] Lian Baowang, Li Yong, Zhang Yi. CAN bus system design and implementation, Wireless Engineering, 2000.1

[4] Zhang Zhuancheng, Zou Tao, Feng Lijie, Lai Yuqiang. A distributed fire alarm control system based on CAN bus [J] micro-computer information, 2006.5-2:P29-31