**CAN bus**

**The definition of CAN bus:**

CAN, which stands for Controller Area Network, is an ISO international standard serial communication protocol. CAN bus protocol became the standard bus of automotive computer control system. J1939 protocol which was designed for heavy machinery vehicles and large tracks was based on CAN as the underlying protocol. In automotive electronics, engine control units, sensors, antiskid-systems, etc. are connected using CAN with bitrates up to 1Mbit/s.[1]. CAN bus was one of serial communication and it was better than RS485 communication which individual used in Y3 project. Compared with I2C which is synchronous communication modes with clock signal, CAN bus was an asynchronous half-duplex communication instead of synchronous with clock signals. It means that receiving and sending cannot be performed at the same time.

**CAN bus protocol and its composition:**

CAN protocol has two standards ISO11898 standard and IS011519-2 standard after ISO standardization. There were two ways of CAN composition. One of them was that CPU and CAN controller were integrated, then the external CAN transceiver. The other way was that CPU and CAN controller was separate. It should configure the CAN interface circuit when using. STM32 applied the first way which made CAN interface integrated in the chip, and then external CAN transceiver which can receive and transmit.

CAN transceiver was used for TTL level and differential voltage signal conversion. TTL level was directly provided by pin of MCU (logic 0 represents low level and 1 is high level). Differential voltage signals were fixed voltage values.

**The working principle of Transmission:**

CAN bus was completed by two wires: CAN\_H (CAN High) and CAN\_L (CAN Low), which were integrated to be a group of differential signal lines. The data transmission was achieved by the form of differential signal, in other words, the differential voltage of two signal lines of logic 0 and logic 1[2]. For example, if MCU would send logic 1, CAN\_TX should be set as 1, the voltage of CAN\_High and CAN\_Low were 2.5volts after CAN transceiver. Therefore, the differential voltage to the bus was 0 and the state on the bus was logic 1. Similarly, if the voltage of CAN\_High and CAN\_Low on the bus were 3.5volts and 1.5volts respectively, which means that the differential voltage was 2volts, MCU can get logic 0 after CAN transceiver.

The idle state was defined as the condition when node was transferring data. When there were a continuous 11-bit hidden level (logic 1 means implicit and 0 means dominant) on the CAN bus, CAN bus was on the idle state. However, as long as one node outputs the dominant level, the bus would be the dominant level, which means that dominant level had priority on the bus. When the node transmitted data to bus, this data contained its ID information and other control instructions. This kind of data was called data frame. It started by logic 0 and ended by seven continuous logic 1. Between, there were identify section, control section and ACK section. Identify had 11 bit and determined the priority of data frame and whether other devices received this data frame. Data was the core of data frame and it had 0-8 bytes, the first bit was MSB. CRC was used to check error of data transmission. ACK was made up ACK slot and ACK delimiter. In the ACK slot, the sender sent a recessive bit, and the receiver responded by sending a dominant bit.

If two nodes transmitted data at the same time, ID was used to determine the priority of transmission by comparing the number of 0[3]. For example, there were 3 nodes in the bus. Node 1 ID was 0001000110 and node 2 ID was 0001000111. Node1 and node 2 sent byte information to node3 at the same time. At first, bus was on the state of idle. Node1 and node2 sent data frame start signal at the same time and 3 nodes adjust bit to reach synchronization. When compared 11th bit of ID, node 1 had the send priority due to more 0. In the meanwhile, node1 sent its ID to node2 and node3. It also received the ID of node2. Node3 began to receive data of node1. After hard synchronization, when node 1 and node 2 sent a bit of ID, CAN controller of three nodes always checked whether the bit timing was consistent with the bus bit timing. When there was a phase leading or lagging, it will automatically resynchronized.

**Reference**

[1] G. Liu, A.-p. Xiao, and H. Qian, "Communication system design based on TMS320F2407 with CAN Bus," *AASRI Procedia,* vol. 3, pp. 463-467, 2012.

[2] C. Qiu, "A design of CAN-bus test system based on Volkswagen PASSAT B5," in *2011 Second International Conference on Digital Manufacturing & Automation*, 2011: IEEE, pp. 16-19.

[3] M. Marchetti and D. Stabili, "Anomaly detection of CAN bus messages through analysis of ID sequences," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017: IEEE, pp. 1577-1583.