Sample Point Position Measurement of Controller Area Network Nodes

Jiří Novák

Czech Technical University in Prague, Faculty of Electrical Engineering, Dept. of Measurement
Technická 2, CZ-16627 Prague 6, Czech Republic; phone +420224352807, fax +420233339929;
jnovak@fel.cvut.cz

Abstract-In this paper a method for sample point position measurement in the bit interval of Controller Area Network (CAN) nodes is presented. The position of sample point is an important parameter of CAN network node configuration, as it particularly influences the bit error rate and consequently frame error rate within the whole system. The measurement method, implementation of a specialized test instrument and the measured results evaluation are described, as well as validation procedure of the described method and its results.

I. Introduction

Controller Area Network ([1], [2]) is one of the fieldbus standards that are used in miscellaneous industrial applications. It comes from the automotive industry and up to now it is widely used for communication among Electronic Control Units (ECUs) in vehicles. Timing accuracy and reliability of data delivery play the crucial role in such systems. Correct implementations of the communication interface at all protocol layers as well as the ECU interoperability with the rest of the system influence not only the vehicle functionality, but also the safety of passengers. The sample point position setting is a protocol parameter, which is set by the firmware of particular ECU into the CAN controller and its value is not simply visible from the outside. Wrong setting can cause higher level of frame error rate in the system, as the CAN uses broadcasts and incorrect behavior of any node influences the performance of the whole system. The measurement method and its implementation described below allow early detection of faulty sample point setting and correcting it according to the application requirements.

II. CAN Communication Principle

CAN standard [2] describes a link layer protocol of OSI (Open Systems Interconnection) model. Its functionality requires a special behavior of underlying physical layer. Its signaling must consist of two complementary levels and relative states – the dominant and the recessive. If all CAN nodes transmit the recessive level, there is the recessive state on the bus. If any number (at least one) of nodes transmits the dominant level, there is the dominant state there. CAN frames (see Figure 1) are broadcasted within the system and all nodes receive them simultaneously.

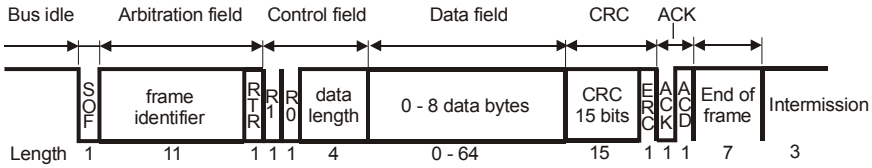

Figure 1. CAN data frame format

A medium access method used in CAN networks is called Carrier Sense Multiple Access with Collision Resolution (CSMA/CR). All nodes have the same right to start transmitting after the idle state is detected on the bus (predefined number of recessive bits is received). The collision that possibly happens, when two or more nodes start transmitting in the same time, is resolved in the following manner. Each frame starts with the start-of-frame bit, which is always dominant, following by so called Arbitration field. The first part of this field is an identifier, that either identifies the frame content, either it defines the frame priority – the lower identifier, the higher priority. When two or more nodes start transmitting simultaneously, the resulting bus state is received back and the transmitting node receiving the dominant level while sending the recessive one stops transmitting. As the identifiers are unique, i.e. only one node can transmit the frame with one particular identifier, the colliding frames differ at least in one identifier bit and thus the collision is resolved. Thanks to the physical layer feature

the collision is non-destructive and winning node continues in transmitting of the following fields (control, data and CRC).
The CRC is used to secure the frame against the data transmission errors. Its length (compared to the maximum frame length) is relatively high and together with other safety mechanisms it provides highly reliable way for the data exchange. If the CRC check fails or some other rule is violated and any receiver detects an error, it immediately sends so called error frame, consisting of 6 consecutive dominant bits. Error frame occurrence violates the bit-stuffing rule (see below) and all network nodes thus detect an error during the transmission. As the frames are broadcast to the network (to all the nodes simultaneously), one node, which encounters receive problems, could block the whole system by repeated generation of error frames. This is avoided by implementation of error level states for particular CAN nodes. The error state transitions are controlled by values of two error counters (receive and transmit). These values are incremented when an appropriate error (during the reception or transmission) is encountered. If the value of both counters lies below the first limit, the node is in error active state. If one of them crosses this limit the node goes into the error passive state and if the transmit counter reaches the second limit value, the node goes into the bus-off state. Only the error active nodes can actively generate error frames (using 6 consecutive dominant bits), error passive nodes can generate only passive error frames (using 6 consecutive recessive bits) and bus-off nodes are totally disconnected from the bus. The last field is an acknowledge field where receivers acknowledge the frame reception using a dominant bit level while the transmitting node transmits the recessive one.
To keep receivers synchronized within the whole frame reception a bit level synchronization is used that compensates the difference between the transmitter and receivers clock frequencies. Each edge in the incoming signal can therefore be used for resynchronization, typically only the recessive to dominant transitions are used. Long sequence of bits with the same logic value thus has to be avoided, otherwise there are no input changes allowing resynchronization. A bit-stuffing mechanism is used that after each 5 consecutive bits of the same logic level inserts one bit of complementary logic level into the transmitted bit stream. The receivers remove this bit by analogy using the same algorithm. High enough frequency of edges in a received bit stream is thus reached aside from its information content.

III. Measurement Issues Identification

The bit synchronization mechanism makes the sample point position measurement is a rather complex task. The bit interval consists of four parts – segments (see Figure 2). They are the synchronization and propagation segments, phase segment 1 and phase segment 2. Within the synchronization segment the incoming edge is expected. If the edge is detected outside of this segment the time difference is called phase error. For early edges its value is negative, for late edges it is positive. The propagation segment compensates the signal propagation delay between the CAN controller circuits (driver and receiver delay + twice the transmission line delay). The bit value is sampled at the phase segments boundary. Each segment consists of an integer number of basic time interval – a time quantum, which is delivered from the local clock source. Programming the CAN controller sets the number of time quanta for particular segments and thus the position of sample point in the bit interval as well as the nominal bit-rate. Conversely, the sample point position measurement provides the basic prove of correct or incorrect CAN controller programming (together with the bit-rate measurement, which is quite easy).

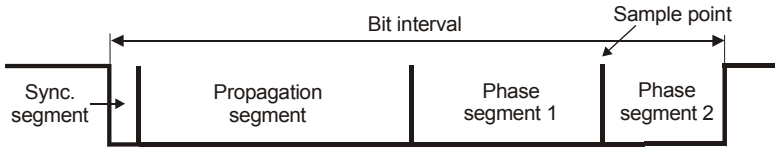

Figure 2. Bit interval structure

Bit synchronization mechanism works as follow (only in case of change the recessive state to the dominant one). The bus state change is always expected within the synchronization segment. If it occurs later, the phase segment 1 is extended, if earlier, the phase segment 2 is shortened. Maximum number of time quanta the segments can be shortened or extended (called synchronization jump width) in single synchronization step can be defined. From this point of view the sample point position measurement is not a trivial task, as the bit interval length may change bit by bit from its nominal value. Additionally, the tested node must be kept in an error active state, as in an error passive state it is not able to actively report detected error condition that is used for measurement. It means there must be at least one other node in the network that receives and acknowledges frames transmitted by the tested ECU. Details concerning the synchronization can be found e.g. in [3].

IV. Measurement Method

The only way to measure the sample point position is indirect (as there is no direct access to the internal ECU data) – by evaluation of the ECU under test reaction in case the CAN protocol is violated. Either the tested ECU transmission or reception can be used for this measurement. The first case is clearly more suitable, as disturbing the node transmission with very tight timing is much more difficult. Therefore the principle of the frame transmission with incorrect timing and evaluation of the tested ECU reaction was chosen as a basis.

Using the first bit of the frame (start-of-frame bit) for this measurement seems to be the simplest test method, as it does not require special test instrument and only the programmable generator allowing the pulse length setting satisfies the needs. The principle of measurement is very simple. The generator generates just one bit of logic 0. If the generated start-of-frame bit is longer than the sum of synchronization, propagation and phase 1 segments, the reception starts, as the tested ECU has detected a valid start-of-frame bit. After the five consecutive recessive bits (logic 1) the bit stuffing mechanism requires a zero bit insertion, but in our case another logic one bit is received. Under this condition the error frame is generated by the tested ECU. It confirms the tested ECU has previously started the reception and thus the reception of valid start-of-frame bit. If the generated start-of-frame bit is shorter than the sum above, the bit is not detected as valid start-of-frame bit and no reception thus starts. Unfortunately, most of CAN controllers use additional mechanisms (time-domain filtering) that check the validity of start-of-frame bit to avoid erroneous start of reception caused by line disturbances. Thus this simple evaluation of sample point position in the start-of-frame bit is not possible.

Measurement method similar to that above was finally chosen with the tested bit in the data field. The use of a simple pulse generator is therefore impossible, as the test generator should follow the CAN specification and also provide some special features not available in standard controllers. The most important one is the ability to change a particular bit length with a suitable resolution. Additionally, a CAN node that acknowledges frames transmitted by the ECU, which sample point position is measured, is necessary to keep the tested ECU in an error active state, allowing active error frame generation (for the above described start-of-frame bit based measurement method it is necessary too).

Final test setup consists of three nodes – the tested ECU, CAN test generator and the auxiliary node, which is either used to acknowledge frames sent by the tested ECU, either to detect error frames generated by the tested ECU when an error (bit stuffing) is detected. The test generator sends a frame with ID = 0x555, data length = 8 and data 0xAA, 0xA0, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF and 0xFF. The bit sequence up to the third data byte is shown in Figure 3. It contains maximum number of logic level transitions before the bit where the test takes place in order to synchronize the receiver.

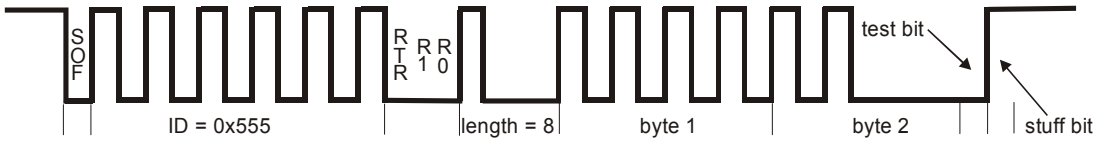

Figure 3. Test frame format

The measurement sequence is implemented by increasing the bit length of the last bit of the second data byte (*test bit*). The tested ECU is last synchronized by recessive to dominant change after the third bit of the second data byte. After the *test bit* the bit-stuffing bit must be inserted (because of 5 dominant bits). The test bit length is increased step by step during the test with the minimum generator resolution. If the length of the test bit reaches the nominal bit length plus the length of synchronization, propagation and phase 1 segments, the controller samples sixth bit in logic 0 level and the bit-stuffing rule is thus violated from the point of view of the tested ECU. It immediately transmits an active error frame, which is detected by the auxiliary node (together with the error type and its position within the test frame). The phase segment 2 of the auxiliary node should be as short as possible, as the position of its sample point cuts an upper limit of measurement range. In case the sample point position of the auxiliary node is closer to the start of the bit than that of the tested ECU, its value is measured instead of the tested ECU one. The measurement has to be repeated several times and the maximum value used as a result because of the uncertainty of the last synchronization edge position in the synchronization segment of the tested ECU. The number of repetitions depends on the time quantum value (generally unknown, maximum can be estimated), and the required A type uncertainty of the result of measurement. The B type uncertainty primarily depends on the difference between the CAN generator and tested ECU clock frequencies. In typical applications of this method where the total required measurement uncertainty is up to 2% this can be omitted.

V. Implementation, Results and Evaluation

Implementation of the above-described measurement method is quite difficult task, as it requires a special CAN test generator allowing precise setting of particular bit timing. Such a generator was developed in past for testing the CAN nodes and systems behavior under the defined error conditions. It can send any type of CAN frame (standard or extended format, error, data or remote request frame). The frame can conform to the standard or it can contain nearly any type of error or group of errors. The detailed generator description can be found in [4]. Later its functionality was extended especially in terms of performance (timing resolution, number of possible bit lengths, number of frame definitions) and the generator was transformed into IP (intellectual property) form written in VHDL. This generator version implemented in FPGA together with the standard SJA1000 controller playing the role of the auxiliary node have been integrated on a single PC board allowing specific CAN implementation parameters testing. The CAN test generator timing resolution is 20 ns.

To evaluate the proposed method several CAN nodes were used that allow sample point position programming. They also used different type of CAN controller implementations (I80C527, SJA1000, MC9S12DG128) to prove whether the method is independent on it. The auxiliary node was based on the SJA1000 controller that was used to detect the error frames generated by the tested ECU. Not only the presence of the error frame on the bus has been evaluated, but also the direction (receive) and field in frame where the error was detected in order to avoid the usage of error frames that occur from different reasons. The sample point position of auxiliary CAN controller was set at more than 90% in all cases in order to provide the highest possible value of the upper limit of measurement (the tested ECU sample position can be evaluated up to the value set in the auxiliary node). The measurements were run using the high-speed physical layer transceivers according to [1], but the method is not sensitive to the transceiver delay (unless the values of dominant to recessive transition and vice versa are significantly different) and it should provide correct results for other physical layers too.

Following results of measurement come from testing this method on the PCMCIA CAN interface card equipped with SJA1000 CAN controller. Results were measured for 500 kB/s; CAN generator resolution of 20 ns provided the sample point position resolution of 1%. Two controller configurations were measured, the first for 16 time quanta per bit and the second for 8 time quanta per bit. The test cycle was repeated 500 times and maximum measured value was used as a result. For 16 time quanta per bit the 50%, 56%, 62%, 68%, 75%, 81% and 87%, for 8 time quanta per bit the 50%, 62% and 75% sample point positions were evaluated.

In all the above-mentioned cases the absolute error of measured sample point position was not higher then the timing resolution of the generator, i.e. 1% of the bit time interval.

VI. Conclusions

The measurement method allowing measurement of sample point position within a bit interval of CAN node was designed and evaluated. A test setup requires a special instrument – CAN test generator, which provides the possibility of independent change of particular bit width in transmitted CAN frame. The method was evaluated using the high-speed physical layer transceivers and several CAN controllers from different manufacturers. The measured results correspond to the actual sample point position settings in all tested CAN nodes. The maximum difference between actual and evaluated value is 2%, which is precise enough for the purpose of measurement.

For practical purposes the test can be improved using additional knowledge about the tested ECU CAN interface implementation, e.g. the crystal oscillator frequency and CAN controller type. Using this information a set of possible controller settings that allow the required transmission bit-rate can be prepared (supposed the bit-rate is programmed correctly) and the most probable one chosen using the result of the above described measurement method of the sample point position.

References

[1] ISO-WD 11898-1, Road vehicles – Controller Area Network (CAN) – Part 1: Data link layer and physical signaling, 1999.

[2] Robert Bosch GmbH, CAN Specification v. 2.0, 1991

[3] K. Etschberger, “Controller Area Network”, IXXAT Press, Weingarten, Germany, 2001.

[4] J. Novák, A. Fried, and M. Vacek, “CAN Generator and Error Injector”, *Proceedings* of *IEEE International Conference on Electronics, Circuits and Systems ICECS2002*, CD ROM, pp. 967–970, September 2002, Dubrovnik, Croatia.