

Design and Experimental Verification of a CAN Bus-Based Vehicle Driving Data Recorder

Zhongwu Cao

Yangtze University

School of Electronic Information and Electrical Engineering

Jingzhou, China

Abstract: A vehicle driving data recorder can acquire and preserve selected operating information for later inspection. This paper presents a proposed recorder architecture based on a C8051F020 microcontroller, an external SJA1000 CAN controller, and a TJA1050 CAN transceiver. The design also considers local flash storage, USB data transfer, a real-time clock, voice indication, and CAN-to-cellular/GPS extensions. The available experimental work focused on communication verification using a PCAN-USB interface and PCAN-View to generate simulated CAN messages rather than data from a real vehicle. An extended frame was configured at 250 kbit/s with a 100 ms transmission interval. For the reported engine-speed example, the two-byte value 0x1111 was converted using a scale factor of 1/8 to obtain 546.125 r/min, matching the value displayed at the receiving endpoint. This result supports the functionality of the tested simulated message path; it does not demonstrate that every proposed recorder module or the complete embedded firmware was validated. The paper distinguishes the proposed system design from the communication test actually performed.

Keywords: CAN bus; vehicle driving data recorder; C8051F020; SJA1000; TJA1050; message decoding

1. INTRODUCTION

Modern vehicles contain multiple electronic control units that exchange operating information during driving. As the number of monitored signals increases, point-to-point wiring can become complex, expensive and difficult to extend. A shared in-vehicle communication bus offers a practical way to collect information from distributed nodes and make selected data available for recording and later analysis.

A vehicle driving data recorder can preserve information such as engine speed, vehicle speed, trip timing and selected vehicle states. These records may support fault inspection and driving-event review. Its usefulness depends not only on acquiring messages, but also on correctly interpreting identifiers and payload fields, transferring data reliably, and distinguishing signals available on a particular vehicle from signals that the device is designed to support.

CAN is widely used in distributed control because it provides message-based communication, bus arbitration and error-handling mechanisms. Previous studies have discussed CAN/LIN-based vehicle body-control systems and CAN-based driving-recording designs [1, 2, 12]. The use of C8051-family controllers in embedded measurement applications has also been reported [4], while the relevant microcontroller, CAN-controller and transceiver functions are described in their respective documentation [3, 5, 6]. These sources provide background for the design considered here.

The two cited application-oriented studies address vehicle networking and recording at the system level, whereas the present manuscript emphasizes a compact hardware concept and a controlled message-transfer example. Its scope is deliberately limited: it does not propose a new CAN protocol, a new transceiver circuit, or a statistically validated performance improvement. The practical value of the reported test is that it documents a repeatable payload configuration and the corresponding decoded result.

A practical limitation of recorder development is that access to a real vehicle and its internal bus signals may be restricted. A CAN interface can instead be used to generate controlled test messages, allowing a message path and decoding procedure to be checked before in-vehicle integration. However, a simulated message test is narrower than a full vehicle trial and should not be interpreted as evidence of automotive reliability or complete system integration.

This paper documents a proposed recorder architecture and a communication experiment performed using PCAN-View and a PCAN-USB interface. The design description specifies a C8051F020 microcontroller, an external SJA1000 CAN controller and a TJA1050 transceiver. In the experiment, simulated CAN frames were transmitted through a data-transmission module and inspected at a receiving endpoint. The reported engine-speed value was 546.125 r/min. The scope and limitations of this evidence are stated explicitly.

2. MATERIAL AND METHODS

2.1. System architecture

The proposed recorder is organized into acquisition, processing, storage and output functions. A CAN interface receives frames from a vehicle bus or a compatible test source. The controller processes frame identifiers and payload fields, then passes selected information to storage or an external communication interface. A real-time clock is intended to associate records with timestamps. Voice indication and cellular/GPS modules are included as extensions in the original design concept; they were not validated by the communication test reported here.

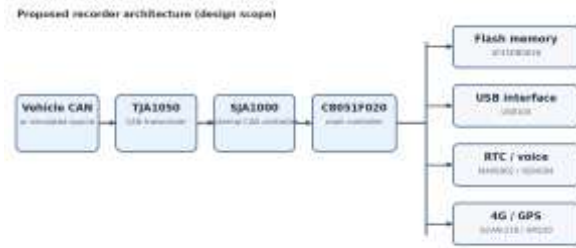


Figure. 1. Proposed CAN-based recorder architecture; peripheral functions are design elements and are not all experimentally verified.

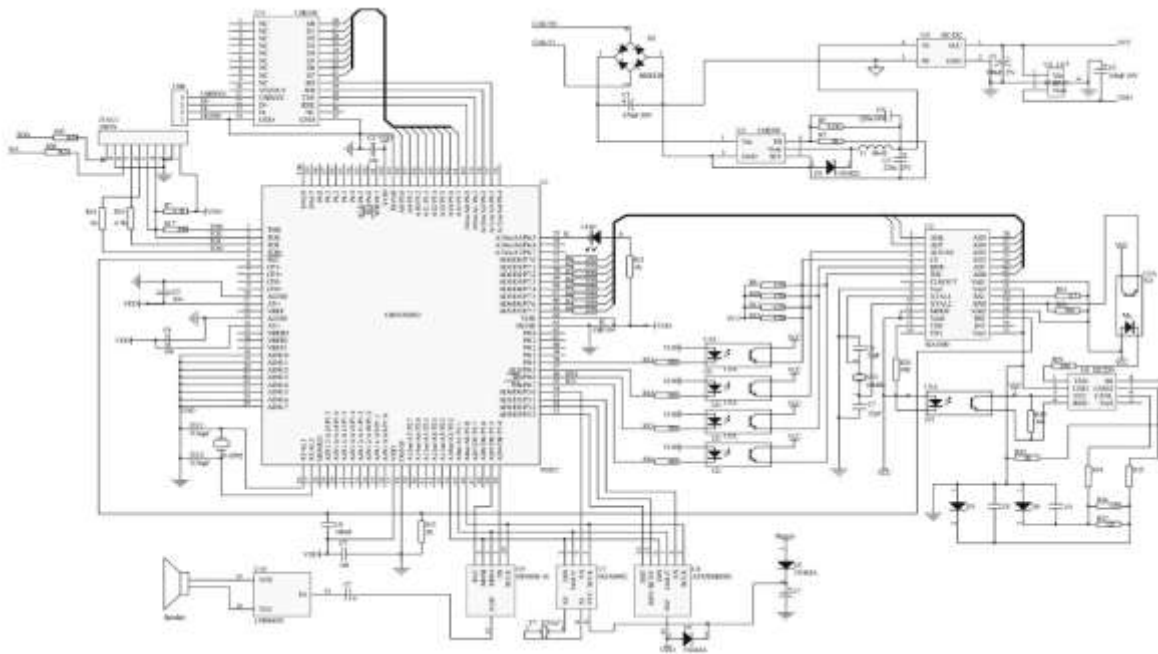


Figure. 2. Overall circuit schematic of the vehicle driving data recorder.

2.2. Hardware design

2.2.1. Main controller and CAN interface

The C8051F020 is specified as the main controller [3]. In the proposed architecture, it communicates with an external SJA1000 CAN controller [5], which handles CAN frame-level operations including message transmission, reception and acceptance filtering. A TJA1050 transceiver [6] is specified as the physical-layer interface between the controller-side logic and the differential CANH/CANL bus. The author has confirmed that TJA1050 was the transceiver used in the project. The original circuit illustration labels the device PCA82C250, whereas TJA1050 is specified for the reported hardware. This component-label discrepancy remains in the source schematic.

The undergraduate report lists a 25 MHz clock, 22 interrupt sources, eight 8-bit I/O ports and 4352 bytes of on-chip RAM among the C8051F020 characteristics used to justify the selection. These are component-level specifications cited by the original report; no independent electrical characterization was conducted in the reported communication test.

The SJA1000 configuration described in the source includes selection of the operating mode, bus-timing configuration, acceptance-filter setup and output-control settings. The controller must be placed in its configuration/reset state before those settings are written, and its status should be checked before normal

operation. The source report also discusses transmit and receive buffers, which are important for avoiding overwritten or unreleased messages in a continuously operating node.

2.2.2. Power conversion, storage and peripherals

The original design further describes an LM2596 switching regulator and a 1117-series regulator for power conversion, an AT45DB081B serial flash device for local storage [8], a USB100 interface for data transfer [9], a MAX6902 real-time clock, and an ISD4004-series voice device [10]. A GCAN-218 CAN-to-4G interface and a UM220 GPS receiver are also listed. These are design elements documented in the undergraduate report; the available test evidence does not establish that all modules were integrated and tested together.

The original power-supply calculation uses a 150 kHz switching frequency for the LM2596 and reports a calculated inductance of 68 μ H for the selected conversion conditions [7]. It also describes a 1N582x-series Schottky diode and a 1117-series regulator for the lower-voltage rail. These details are reported as part of the proposed circuit; the present test did not measure output ripple, efficiency, temperature rise or load regulation.

The source report describes the AT45DB081B as a page-organized serial flash memory with 4096 pages and two 264-byte SRAM buffers [8]. This organization was selected to support buffered data

acquisition. Those capacity figures describe the selected component, not a measured 20-day recording result; actual record duration would depend on record size, sampling frequency, write policy and usable memory.

The USB100 interface is proposed for transferring stored records to a computer. The source firmware description uses the module TXE and RXF status signals to control data writes and reads [9]. The original material does not provide a test log for sustained USB transfer or verification of the integrity of exported records.

The MAX6902 real-time clock is intended to provide date and time information, while the ISD4004-series device is intended to provide audible prompts when a configured condition is detected [10, 11]. The design specification also lists a GCAN-218 CAN-to-4G interface and a UM220 GPS receiver. No measurements of the clock accuracy, voice-alert operation, positioning accuracy or cellular throughput appear in the available test record.

2.3. Software design

The proposed embedded software is intended to initialize the controller, configure CAN timing and acceptance filters, receive frames, parse selected data fields, and transfer records to storage or an external interface. The SJA1000 must be configured before normal communication, and received frames must be read and released according to the controller interface [5]. The original software chapter also contains XGATE and MSCAN/CANCTL terminology that does not match the stated C8051F020-plus-SJA1000 architecture. Since no verified firmware build or on-target execution evidence has been supplied, those code fragments are not treated here as a demonstrated implementation.

The intended receive path also needs a policy for frames that do not match a known identifier, a strategy for buffer overflow, and a defined record format for persistent storage. Those implementation details are important for a deployable recorder but are not documented sufficiently in the current test material to be claimed as implemented.

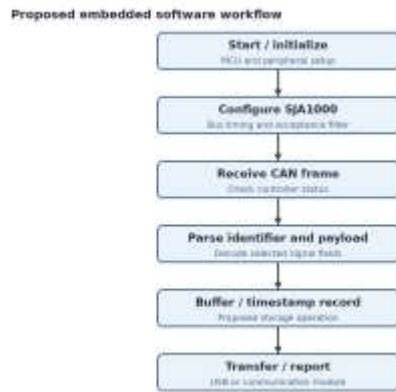


Figure. 3. Proposed embedded software workflow, reconstructed from the design description.

The performed PCAN test followed a separate software path: PCAN-View was used to configure and transmit the test frame, while HslCommunication was used at the receiving side as an MQTT client. This distinction is important. The PCAN-View test verifies the configured communication example, but it does not by itself verify the proposed microcontroller firmware.

2.4. CAN message interpretation

The test message was configured with the identifier 0x00F00400, an eight-byte payload and an extended-frame format. The original report associates the F004 field with engine speed and places the two-byte value in the fourth and fifth data positions. Both bytes were set to 0x11, giving the hexadecimal value 0x1111. The report applies a scale factor of 1/8 r/min per count [12].

$$0x1111 = 4369; \quad n = 4369 / 8 = 546.125 \text{ r/min}$$

The identifier, byte positions and scale factor are reproduced from the original experiment description. Identifier notation and byte numbering must remain unambiguous when comparing this reported example with a vehicle-specific signal definition.

2.5. Design targets in the original specification

The undergraduate specification listed a 2.7–3.6 V supply range, a 12.5 Mbit/s 4G rate, an 8 Mbit/s USB rate, GPS accuracy targets of 5 m horizontally and 10 m vertically, a default positioning update interval of 1 s, a 20-day continuous storage target, and a 4 kHz voice sampling frequency. These values are original design targets only; the PCAN/MQTT experiment did not measure or validate them.

3. RESULTS AND DISCUSSION

3.1. Experimental setup

Because a real vehicle was unavailable, a PCAN-USB interface was used to generate simulated CAN traffic. The interface and a data-transmission module were provided for the undergraduate test. The author performed the PCAN-View message configuration and transmission and carried out the receiving-side check using the provided module and software. The documented connection used a serial (RS-232) link between the PCAN interface and the data-transmission module, with the module powered separately.



Figure. 4. PCAN-USB interface used to generate simulated CAN traffic.



Figure 5. Laboratory connection of the PCAN interface and data-transmission module.

3.2. CAN message configuration

The PCAN-View interface was configured for an extended frame at 250 kbit/s. The message identifier was entered as 00F00400 (hexadecimal), the data length was set to eight bytes, and the transmission cycle was set to 100 ms. The fourth and fifth data bytes were set to 0x11, while the remaining bytes were set to zero, as shown in the original test material.



Figure 6. PCAN-View connection and frame-format configuration.

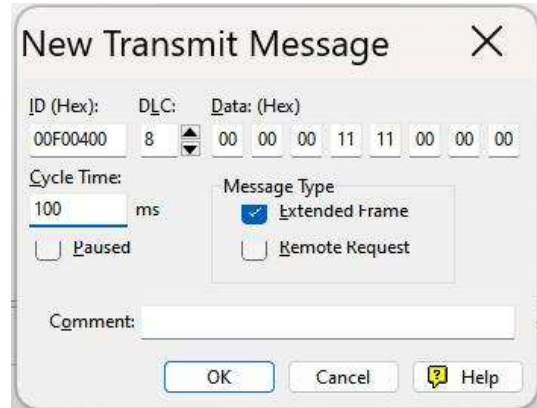


Figure 7. Periodic CAN message configuration in PCAN-View.

3.3. Reception result

After transmission, the HslCommunication test tool was opened and its MQTT client was configured to subscribe to the topic Tbox0000MSG at the server configured for the experiment. The receiving display showed the value 546.125. This matched the value calculated from the configured two-byte payload and the scale factor used in the undergraduate report.



Figure 8. MQTT client display showing the received data.

3.4. Discussion and limitations

The agreement between the calculated value and the receiving display supports successful transmission and reception of the configured simulated message along the tested communication path. It does not establish real-vehicle interoperability, CAN error rate, end-to-end latency, long-term stability, flash-storage reliability, GPS accuracy, voice-alert performance or cellular throughput. The source record does not provide repeated-trial statistics or measurement uncertainty; no such performance claims are made here.

Only one payload pattern and one reported decoded value are documented. The test did not compare multiple bit rates, message loads, identifiers or error conditions. It also did not quantify latency between PCAN-View transmission and MQTT reception. Consequently, the observed result should be described as a functional communication check, not a comprehensive performance benchmark.

The PCAN-View screenshots document frame generation, while the MQTT screenshot documents the receiving-side display. They do not independently prove that the C8051F020 firmware, SJA1000 initialization code, local flash storage and every peripheral described in the design were operational during the test.

A follow-up test should use the final hardware assembly, record the exact wiring and firmware version, vary payloads and identifiers, repeat transmissions, and measure message loss and retrieval integrity.

4. CONCLUSION

This paper has organized a CAN-based vehicle driving data recorder design and documented a preliminary communication test. The proposed architecture specifies a C8051F020 microcontroller, an SJA1000 CAN controller and a TJA1050 transceiver, together with storage and communication peripherals. In the performed PCAN-View test, an extended CAN frame was transmitted at 250 kbit/s with a 100 ms interval. The payload value 0x1111 was converted to 546.125 r/min, matching the receiving display. This result verifies the reported simulated message example only. Integration of the complete recorder and validation under real vehicle conditions remain future work.

5. ACKNOWLEDGMENTS

The author acknowledges the engineering collaborator who provided the PCAN-USB interface and data-transmission module used in the test. The message configuration, transmission and receiving-side check described in this paper were performed by the author.

6. REFERENCES

- [1] Y. Ding, J. Wang, and N. Nan, "Research on the body control system of electric buses based on CAN/LIN," *Automotive Electrical Equipment*, no. 2, pp. 1–4, 8, 2006. (in Chinese).
- [2] "Freescale system basis chip simplifies and helps protect automotive networks," *Microcontrollers & Embedded Systems Applications*, no. 9, pp. 80–81, 2009. (in Chinese; author information not listed in the source thesis).
- [3] Z. Pan, C8051F020/1/2/3 Mixed-Signal ISP Flash Microcontroller Data Sheet. Shenyang: Xinhua Long Electronics Co., Oct. 2002.
- [4] X. Zhiqiang, L. Zhongli, N. Yi, et al., "An automotive center-of-gravity detection system based on the C8051F005 microcontroller," *Journal of Shandong University of Technology (Natural Science)*, vol. 24, no. 1, pp. 27–29, 2010. (in Chinese).
- [5] Guangzhou ZLG Microcontroller Development Co., SJA1000 Independent CAN Controller User Manual.
- [6] NXP Semiconductors, "TJA1050 high-speed CAN transceiver," product documentation. Available: <https://www.nxp.com.cn/products/interfaces/can-transceivers/legacy-can/high-speed-can-transceiver:TJA1050>.
- [7] Y. Huang, "Introduction to the LM2596/LM2596S switching regulator," *CSDN Blog*, Dec. 25, 2022. (in Chinese; as cited in the original thesis).
- [8] Atmel, AT45DB081B Data Sheet.
- [9] Z. Zhang, Z. Zhang, W. He, et al., "Implementation of high-speed communication between a C8051F020-based USB module and a PC," *Computer Applications*, suppl. 2, pp. 461–463, 2003. (in Chinese).
- [10] "ISD4004," Baidu Baike, May 11, 2023. (in Chinese; as cited in the original thesis).
- [11] J. Nan and B. Liu, *Automotive Microcontrollers and In-Vehicle Bus Technology*. Beijing: Beijing Institute of Technology Press, 2005. (in Chinese).
- [12] L. Li and X. Xie, "Design of a vehicle driving recorder based on CAN bus," *Computer Engineering and Design*, vol. 30, no. 22, pp. 5120–5123, 2009, doi: 10.16208/j.issn1000-7024.2009.22.003. (in Chinese).