DIGITAL COMMUNICATION INTERFACE FOR AN AUTOMOTIVE APPLICATION

M.O. SONNAILLON[1,2], G. BISHEIMER[1], C.H. DE ANGELO[1], R. LEIDHOLD[1], G.O. GARCÍA[1], J.C. BALDA[3] and F.D. BARLOW[3]

[1]*Grupo de Electrónica Aplicada (GEA), Universidad Nacional de Río Cuarto, Ruta Nac. #36 Km. 601, X5804BYA Río Cuarto, Argentina Argentina*
msonnaillon@ing.unrc.edu.ar , gbisheimer@ing.unrc.edu.ar, cdeangelo@ieee.org, rleidhold@ieee.org, g.garcia@ieee.org
[2]*Laboratorio de Cavitación y Biotecnología, Centro Atómico Bariloche - Instituto Balseiro, Av. Bustillo 9500, San Carlos de Bariloche, Argentina*
[3]*Laboratory for Agile Motion Prototyping, University of Arkansas, 3217 Bell Engineering Center, Fayetteville 72701, AR, EEUU - jbalda@uark.edu*

Abstract– **This paper presents the design and implementation of a digital communication interface between a Motor Drive Controller (MDC) and a digital network supervisor. This work is part of the DOE/CARAT project "Integrated Controllers for Automotive Auxiliary Electric Motors", being performed between the University of Arkansas and at GEA (Applied Electronics Group), National University of Rio Cuarto. The Controller Area Network (CAN) is the digital communication protocol of choice for automotive applications. The MDC was implemented using a Digital Signal Processor (DSP) and the network supervisor using a standard personal computer (PC) with a CAN communication board.**

Keywords– **CAN Communication Protocol, Motor Drive Controller, Automotive Applications.**

I. INTRODUCTION

The automotive industry is facing nowadays new challenges regarding the reduction of fuel consumption, because of economic and ecological reasons. Demands for increasing comfort, safety, flexibility and reliability are the motivations for new developments in the automotive industry. Replacing a car's hydraulic system with wires, microcontrollers, and computers promises better safety and handling capabilities (Bretz, 2001). In order to reach these goals, some conventional hydraulics, pneumatic and even mechanical systems are expected to be replaced by x-by-wire systems (electromechanical actuators electronically controlled by-wire, without mechanical links). Therefore, the on-board vehicle electric power needs are increasing, and the on-board electric currents are increasing too. In turn, with the increase in the number of electronic and electrical devices used in modern cars, the following two problems should be addressed,

- boost of power consumption,
- needs to handle the communication between many devices.

M.Sonnaillon, G. Bisheimer, C. De Angelo, R.Leidhold and G.García are with CONICET.

The increase in power consumption means higher on-board currents, therefore, thick, heavy and bulky wires. Increasing the voltage reduces the current required for the same amount of power consumption. A reduction in current is of particular advantage for the use of power electronics modules. Higher currents require a larger silicon area, and since the formula "chip area $\propto$ cost" applies in the semiconductor industry, current reduction means cheaper power electronics devices. For this reason, the Vehicle Electrical System Architecture Forum, in conjunction with the MIT Industry Consortium, have proposed a new 42V supply voltage standard for the automotive industry. The proposal of this Forum has achieved international acceptance and the decision of several companies to begin developing components for this new generation of vehicles (Vehicle Electrical Systems Architecture Forum, 2002).

As mentioned previously, another need is to handle the communication between a growing number of devices, using the minimum numbers of wires (i.e. occupying the minimum space). There are many alternatives to implement a digital communication network to interconnect the on-board devices. Several different digital protocols have been proposed in the literature. As examples, can be cited D2B (Domestic Data Bus), Bluetooth, MOST (Media-Oriented Systems Transport), MML (Mobile Media Link), TTP (Time-Triggered Protocol), LIN (Local Interconnect Network), ByteFlight, Flexray, TTCAN and CAN (Controller Area Network) (Leen and Heffernan, 2002).

The University of Arkansas, Fayetteville, USA and GEA (Applied Electronics Group), National University of Rio Cuarto, Argentina, are collaborating on the project "Integrated Controllers for Automotive Auxiliary Electric Motors". The main motivation of this project is to overcome existing technical and economical barriers in the implementation of auxiliary 42V electric motor systems for automotive applications. The main objective of this effort is to develop optimally integrated, energy-efficient PM brushless motor systems, utilizing high-efficiency motors and advanced microelectronic manufacturing and power packaging technologies. The research activity comprises of four strategic tasks: (1) Design of the Motor Controller; (2) Motor Controller Layout and Package Design; (3) Development of a Prototype Smart Monolithic Power Module and Optimally Integrated Motor System; and (4) Economic Viability Assessment.

The GEA is working on the first part of the project, Task 1, specifying, designing and implementing a discrete prototype for a brushless motor controller. This prototype includes a discrete power electronics module and a digital controller with digital communication capabilities.

This paper describes the design and implementation of the digital communication interface between a PC, working as the motor drive supervisor, and a Digital Signal Processor, running as the Motor Drive Controller. CAN protocol was used in this application, since it is the most used protocol in the automotive industry.

The rest of this paper is organized as follows. Section II presents the system specifications. Section III discusses the selection of the digital communication protocol. Sections IV and V provide the design of the digital communication hardware and software, respectively. Finally, Section VI draws some conclusions.

II. SYSTEM SPECIFICATIONS

The main motor drive specifications are the following:

- DC link voltage: 42 V under no load (36 V under load conditions);
- power-module average power: 2.2 kW at rated speed;
- power-module peak power: 4.5 kW at rated speed during 10-sec pulses with 100-sec periods;
- modulation switching frequency: 40 kHz;
- power-module working maximum ambient temperature: 40ºC;
- heat-sink-air thermal resistance: 0.6ºC/W;
- diagnose and protection capabilities: needed;
- digital network communication capability: needed.

The average and peak powers correspond to an example of air conditioning compressor for heavy truck and bus applications. The research ideas are independent of these power levels and application, which were selected to simply demonstrate the feasibility of the proposed ideas and concepts.

The digital controller should be configurable on-line from a PC in order to test the system under different conditions and operating modes.

The PC should be capable of sending the following commands and parameters to the MDC:

- operating mode command (speed controlled mode, current controlled mode, duty cycle controlled mode);
- set points (speed, current or duty cycle references, depending on the selected operating mode);
- parameters (coefficients of the digital controller compensators).

The MDC should be capable of sending the following messages and data to the PC, to be shown on its screen,

- faults messages;
- motor speed, current and PWM duty cycle.

III. DIGITAL COMMUNICATION PROTOCOL

As specified, the MDC should be linked to a supervisor through a bi-directional digital communication interface. The digital protocol that best fit the requirements of this application is CAN (Controller Area Network). It is standard and open for automotive applications. In the today's market different CAN devices can be purchased from different suppliers.

CAN is a serial multi-master communication protocol that efficiently supports distributed real-time control with very high level of data integrity, and communication speeds up to 1 Mbps. The CAN bus is ideal for applications operating in noisy and harsh environments, such as in the automotive and industrial fields that require reliable communication.

Messages with priorities of up to eight bytes in data length can be sent on a multi-master serial bus using arbitration protocol and error-detection mechanism for a high level data integrity.

CAN protocol does not address nodes with physical addresses but instead sends messages with an identifier (ID) that can be recognized by the different network nodes. This identifier has the following two functions:

- message filtering;
- determine message priority.

The ID defines if a transmitted message should be received by any particular CAN node, and also defines the priority of the message when two or more nodes attempt to transmit at the same time. In this case a non-destructive bus arbitration is used to decide which node gets access to transmit its message. The message with the highest priority is transmitted first, followed by the next highest priority message. When one message frame is transmitted, a new round of arbitration begins (Lawrenz, 1997).

The CAN protocol provides the following advantages that make it suitable for this application,

- it is a mature standard (it has existed for more than 14 years);
- there are numerous CAN products and tools on the market;
- it has a combination of error handling and fault confinement with high transmission speed;
- although its physical lair is not defined by the standard, there exist several application examples that use simple transmission medium like twisted pair of wires, optical fivers, etc.;
- it has excellent error handling capabilities;
- it has fine fault confinement;

- it is the most used protocol in automotive applications;
- it has the best performance/price ratio.

IV. DESIGN OF THE DIGITAL COMMUNICATION HARDWARE

The MDC is implemented using a TMS320LF2403 DSP (member of the C2000 Texas Instruments DSP family), which has several on-chip peripherals that make it suitable for the application target of this work.

This DSP has a full-CAN on-chip controller. It contains a message handler (for transmission and reception management, and frames storage) and needs less CPU overhead than with a conventional out-of-chip CAN controller. This on-chip controller fulfills the CAN 2.0B active specification, meaning that the module can send and accept the standard 11-bit identifier and the extended frames 29-bit identifier (Texas Instruments, 1998).

The CAN controller needs a transceiver, according to the adopted physical layer protocol, in order to be connected to the CAN bus. For this application it was chosen the SN65HVD230 transceiver. Designed to interface the Texas Instruments TMS320Lx240x 3.3-V DSP family (Texas Instruments, 2001) with a CAN bus. It is intended for applications employing the CAN serial communication physical layer in accordance with the ISO 11898 standard. This transceiver has differential transmit and receive capability between the DSP and the bus, with speeds up to 1 Mbps. It operates in a −2V to 7V common-mode range on the bus, and it can withstand common-mode transients of ±25 V (Texas Instruments, 2001). It was designed for operation in especially-harsh environments, features cross-wire protection, loss-of-ground and over-voltage protection, over-temperature protection, as well as wide common-mode range.

The supervisor node was implemented with a standard PC holding a National Instruments PCI-CAN board. This board has 2 CAN ports that can be used independently, and supports a wide variety of transfer rates up to 1 Mb/s (National Instruments, 2002).

Inside the board, the CAN bus hardware interfacing is accomplished using the Intel 82527 CAN controller chip. The high-speed CAN physical layer fulfills the ISO 11898 physical layer CAN specification. The physical layer is optically isolated up to 500V, and can be powered either internally (from the card) or externally (by the bus cable).

In the present application, the physical medium that links the digital MDC with the PC is a twisted pair of copper wires, that are connected between MDC and the PC board by standard DB-9 connectors. If it is desired, other CAN nodes (PCs, DSPs, etc.) can be connected to the same network without any hardware change.

The twisted wires constitutes a transmission line. If the transmission line is not adequately terminated, signal can be reflected producing communication failures. Because signals flow both ways, the CAN bus requires that both ends of the cable be terminated. However, this requirement does not mean that every device should have a termination resistor. If multiple devices are placed along the cable, only the devices on the ends of the cable should have termination resistors.

The termination resistors on a cable should match the nominal impedance of the cable. ISO 11898 requires a cable with a nominal impedance of 120Ω (Philips, 2003). Therefore, a 120Ω resistor should be used at each end of the cable. Each termination resistor should be capable of dissipating 0.25 W of power (National Instruments, 2002).

Figure 1 shows a block diagram of the implemented CAN bus.

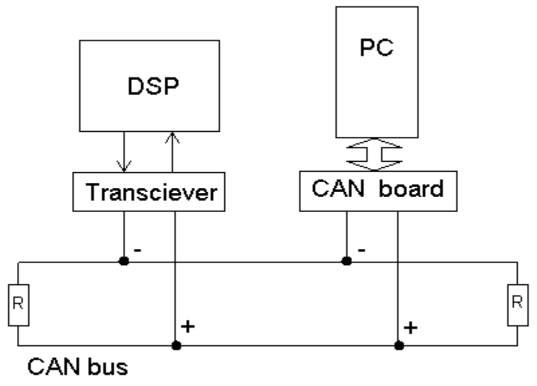


Figure 1: Block diagram of the implemented CAN bus.

The maximum transmission rate depends on the bus length, for a bus shorter than 40 meters the transmission rate is up to 1 Mbit/s. Table 1 shows typical values for thick and thin cable (National Instrument, 2002).

Table 1: Maximum transmission rate and length vs. the cable size.

Bit Rate	Thick Cable	Thin Cable
500 kbps	100 m	100 m
250 kbps	200 m	100 m
100 kbps	500 m	100 m

V. DESIGN OF THE DIGITAL COMMUNICATION SOFTWARE

The design of the digital communication software can be divided in the following three parts:

- Message definitions;
- DSP communication software;
- PC supervisory software.

A. Message data structure

In this application, the supervisory node (the PC) must be able to parameterize and configure the MDC. Moreover, the MDC should report its status when the supervisory ask for it.

The following 4 messages that can be transmitted from the PC to the DSP were defined:

ID: 0.
Function: Defines the working mode.
Size: 1 word (2 bytes).
Data:
Word 0 is interpreted by the controller software as sixteen bits unsigned integer,
= 0, all MOSFETs are turned Off and the compensators (current and speed) are reset,
= 1, ask the MDC go to the Speed Controlled mode,
= 2, ask the MDC go to the Duty Cycle controlled mode,
= 3, ask the MDC go to the Motor Current controlled mode,
= other value, same as 0.

ID: 1.
Function: Defines set-points.
Size: 2 words.
Data:
Word 0 is interpreted by the controller software as 16-bit signed integer, Q15 scaled variable (Lapsley *et. al.*, 1997),
= indicate the value of the speed, duty cycle or motor phase current reference, depending on the present working mode;
Word 1 is interpreted by the controller software as sixteen bits unsigned integer,
= 0, indicate to the DSP that the control algorithm has to use the sextant value read by the rotor position sensor,
= 1 to 6, indicates that the control algorithm has to use a constant sextant value from 1 to 6,
= other value, it is ignored.

ID: 2.
Function: Configures the speed PI compensator parameters for the speed control loop.
Size: 3 words.
Data: Word 0 to Word 3 are interpreted by the controller software as a sixteen bits signed integer, Q15 scaled variables;
Word 0,
= kpw, is the speed PI proportional constant;
Word 1,
= kiw, is the speed PI integral constant;
Word 2,
= kcw, is the speed PI anti-reset windup constant.

ID: 3
Function: Configures the current PI compensator parameters for the current control loop.
Size: 3 words
Data: Word 0 to Word 3 are interpreted by the controller software as a sixteen bits signed integer, Q15 scaled variables;
Word 0,
= kpi, is the current PI proportional constant;
word 1,
= kii, is the current PI integral constant;
word 2,
= kci, is the current PI anti-reset windup constant.

The MDC uses the DSP's "automatic replay to a remote request" (ARRR) or "auto-answer" feature that is a standard in CAN controllers (Texas Instruments, 1998). This feature allows the DSP to answer a data request without interrupting the task it is being run. When the DSP CAN controller receives an auto-answering request, identified with ID 4, it sends, automatically through the CAN bus the actual values of rotor speed, phase current, PWM duty cycle and fault status, with the following format:

ID: 4
Function: Informs the actual speed, duty cycle and phase current.
Size: 4 words
Data: Word 0 to Word 2 are interpreted by the supervisor software as a sixteen bits signed integer, Q15 scaled variables;
Word 0,
= speed;
word 1,
= duty cycle;
word 2,
= phase current;
word 3, is interpreted by the supervisor software as a sixteen bits unsigned integer,
= 1, fault, otherwise normal.

B. DSP node CAN software

The DSP CAN module has six mailboxes that are used to send and receive messages. For each mailbox, the user must configure the ID number, the mode (input, output or auto-answer) and some other parameters. The basic block diagram is shown in Fig. 2 (Texas Instrument, 1998).

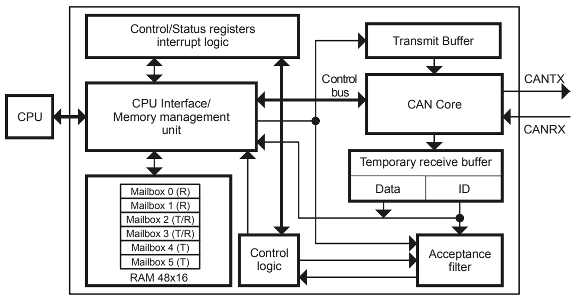


Figure 2: TMS320x240x CAN module block diagram

Each receiving mailbox has an acceptance filter that determines which ID messages will receive. Each time the CAN module receives a message with an ID that corresponds to the previously configured in any of its receiving mailboxes, an interrupt is generated to let the DSP process the recently received data.

When the CAN module sends a message to the CAN bus and no CAN node sends back the acknowledgement signal, an interrupt is generated to alert that nobody is receiving the outgoing messages (probably because of a bus fault).

C. PC node CAN software

The software implemented in the PC runs under Windows and was programmed using C++. The National Instruments PCI-CAN board has its own libraries that can be used in any C++ program (National Instruments, 2002). The PCI-CAN board has the following main functions:

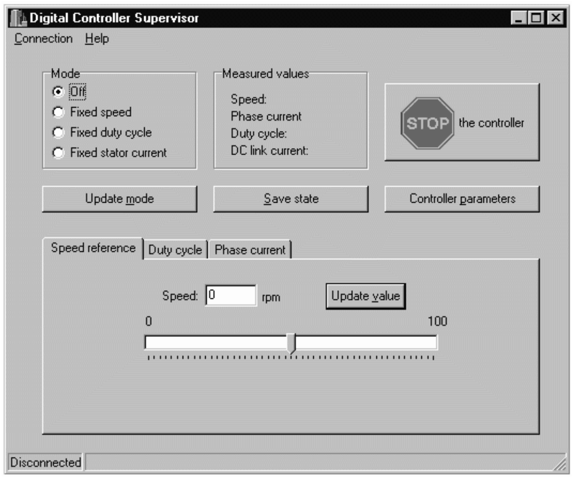
- `ncConfig`: creates an object and defines its parameters (like the ID, mode, sample rate, type and size of data, maximum and minimum values);
- `ncOpenObject`: this function is necessary to start using a message;
- `ncRead`: reads a particularly message from the CAN input buffer;
- `ncWrite`: sends a message through CAN bus;
- `ncWaitForState`: waits until the board completes an operation;
- `ncCloseObject`: clears a message previously opened with the `nctOpenObject` command.

All functions return a status variable that indicates that the operation was performed successfully or an error has occurred. The implemented software makes use of these six functions to carry out the communication with the CAN bus.

The user interface has some protections to prevent a MDC damage, in the motor or in the power stage. The user is not allowed to change the operation mode without first turning off the system. All the references are initialized to zero when the system is turned on. The main window has also an easy-to-find button that stops the system instantaneously, if an emergency occurs.

When the software is started, it loads some parameters from a file, like the default controller's parameters, the maximum and minimum values of current and some conversion constants.

The software running in the PC requests the MDC its working status every 500ms, and displays this information in the PC window. Figure 3 shows the supervisory window.



Mode: [✓ Off] Fixed speed Fixed duty cycle Fixed stator current

Speed: 0 rpm

Disconnected

Figure 3: Screen-shot of the PC supervisory software

V. CONCLUSIONS

This paper illustrated a communication interface for automotive applications, including hardware and software.

Different digital network protocols have been considered. It was concluded that the Controller Area Network (CAN) protocol best fits the requirements of the proposed application.

The system has been tested in a harsh environment (power electronics switches devices commutating at 100A and 40kHz), over a 30m cable with a transfer rate of 100 kbps. The implemented prototype demonstrated excellent performance and data integrity, as expected.

REFERENCES

Bretz, E. A., "By-Wire Cars Turn the Corner". *IEEE Spectrum*, **38,** 4, 68-73 (2001).

Lapsley, P., J. Bier, A. Shoham and E. A. Lee, *DSP Processor Fundamentals*, IEEE Press, New York (1997).

Lawrenz, W., *CAN System Engineering*, Springer (1997).

Leen, G. and D. Heffernan, "Expanding Automotive Electronic Sysytems*", IEEE Computer* **35**, 88-93 (2002).

National Instruments, *NI-CAN Hardware and Software Manual*, B8-B10 (2002).

Philips, "TJA1040 CAN High-Speed Transceiver", *Application Note* **10211**, 22-23 (2003).

Texas Instruments, "SN65HVD230 Datasheet" (2001)

Texas Instruments*, TMS320F240x DSP Controllers Reference Guide*, 10.1-10.37 (2000).

Texas Instruments, "Understanding the CAN Controller on the TMS320C24x DSP Controller", *Application Report* **SPRA500** (1998).

Vehicle Electrical Systems Architecture Forum, "Why 42 Volts in Motor Vehicles?", www.bordnetzforum-42v.de/bordnetz/42v_e.html (2002).

Received: October 8, 2003.
Accepted: October 27, 2004.
Recommended by Guest Editor O. Crisalle.