CCP based ECU communication with calibration tool

Namrata A. Gajare[#], Bharati M. Narute [#], Bhakti Kalghatgi[*]

[#] *Department of Electronics and Tele-Communication, M.E.S.C.O.E., Pune University*

[1]namrata.gajare@gmail.com

[2] bharati.narute@mescoepune.org

[3]bhakti.kalghatgi@tatatechnologies.com

Received xx/xx/xxxx; Revised xx/xx/xxxx; Accepted xx/xx/xxxx

***Abstract*— Today's automobiles contain a number of different electronic components networked together that as a whole are responsible for monitoring and controlling the state of the vehicle . Modern automobiles contain upwards of 50 electronic control units (ECUs) networked together. The essential safety of the vehicle depend on on near real time communication between these various ECUs. In automotive system different car domains have led to the development of a large number of automotive networks according to their requirements. CAN (Controller Area Network) bus has been widely used in automotive embedded control systems, as well as other industrial monitoring and control systems due to its high speed of communication, good reliability and fault tolerant capability. Communication using CAN protocol reduces wiring harness in vehicle. The CAN Calibration Protocol, which is commonly referred to as CCP, is essentially a software interface used to interconnect ECU calibration and measurement tool with an Electronic Control Unit (ECU). Calibration tool provides for the acquisition of data from the ECU and vehicle busses such as CAN, LIN, Ethernet, and Flex Ray. In this way interface describes methods to handle module calibration, measurement data acquisition, and flash programming actions.**

©ACCET-2015. All rights reserved

***Keywords*: ECU (Electronic control unit), CAN calibration protocol, ECU measurement and calibration tool**

I. INTRODUCTION

This literature relates to communication networks where different electronic control units (ECUs) communicate with each other over Controller Area Network (CAN) protocol. The present paper talks about a new design method for CAN calibration protocol based ECU communication with calibration tool. CAN calibration protocol (CCP) is CAN based application protocol for calibration and measurement data acquisition of ECUs. It is a master-slave type communication protocol and an interface using CAN 2.0B for communication between the master device and the slave device.CCP [3] has defined two CAN data objects to set up communication between the master device and the slave device, command receive object (CRO) and data transmission object (DTO). CRO is used to send CCP commands to the slave device. DTO is used to send responses and measurement data to the master device [3]. The CAN Calibration Protocol, which is generally referred to as CCP, is essentially a software interface used to interconnect a development tool with an Electronic Control Unit (ECU). The interface defines methods to handle module calibration, measurement data acquisition, and flash programming activities. Engine control unit (ECU) is having control of the different parts of the vehicle. ECU communicates with different part of the vehicle though CAN calibration protocol. The changes in ECU parameters affect the performance of the parameter. To optimize the performance of the vehicle it is need to calibrate the parameters of the ECU. This calibration procedure is needed while testing the vehicle in running condition.

II. ELECTRONIC CONTROL UNIT

A) General behavior of ECU:

Automobiles are equipped with sensors designed for various measurements and control, and these devices are controlled by ECU [6]. Before ECUs, air fuel ratio, ignition timing, and idle speed were mechanically set and dynamically organized by mechanical and pneumatic means. Engine control unit measures the values from the sensors located in different parts of the vehicle and control the action of actuators. Engine management ECU provides real- time regulation of both Fuel Injection and Ignition Timing thus simultaneously guaranteeing maximum engine performance and minimum fuel consumption. So, because of use of ECU in automotive system gives Reduced exhaust emission, Increased fuel efficiency, Better drive-ability, Smoother and/or quieter engine operation, Safety functions, Comfort functions, Interactions with other systems and Diagnostics. The developments of an ECU consist of both hardware and software essential to perform the functions expected from that particular module. Depending on what it is used for ECU's are named (ECM) Engine Control Module, (EBCM) Electronic

Brake control module. (PCM)Powertrain control module. (VCM) Vehicle control module,(BCM) Body control module.

ECU system blocks contain:

-Microprocessor/Microcontroller which executes the instructions as coded and produces output signals by processing the information using specified control algorithms.

- Input Output interface provides the information about how the Sensors Registers the operating conditions and the desired values and how Actuators converts the electrical output from ECU to mechanical parameters.

- Memory Unit contains Boot memory (ROM/Flash), Program memory(Flash), Calibration memory (Flash) and Data memory (Flash, EEPROM, SRAM/Battery backed RAM.

B) File Format in ECU:

Engine Control Unit(ECU) includes two type of file format Hex and A2L. Hex file format further divided into two types Intel Hex file Format and Motorola S-19 file Format. Representation of this two types of file format is given as:-

a) Intel Hex File Format:

Data Record**:**

: nnaaaattddddddddddddddddddddddddddddd.ccd

Field Contents:

1) Start code (:) = ASCII colon indicates start of data record.
2) Byte count (nn) = Two Hex digit and Count of number of data bytes in record. (In ASCII/HEX).
3) Address (aaaa) = Four hex digit and Load the address of data record. (in ASCII/HEX).
4) Record type (tt) = Two Hex digit defines data field from 00 to 05 Record type. (00=Data, 01=End of file).
5) Data (dd) = Actual data bytes in record (two ASCII Characters per data byte)
6) Checksum (cc) = Checksum of count, address, and data. It is least significant byte of the two's complement of the sum of the values of all fields except fields 1 & 6.

b) Motorola S-19 File Format:

Data Record:

Stnnaaaaddddddddddddddddddddd...cc

Field Contents:

1) Start code (S) = ASCII 'S' indicates start of data record.
2) Record type (t)=One digit Record type, '1','2','3'=data, '9'=end of file.
3) Byte count (nn)=Two hex digit Count of number of bytes in record (including address and checksum).
4) Address (aaaa)=Four, six or eight hex digits as determined by the record type .Load address of data record.
5) Data(dd) =Actual data bytes in record (two ASCII Characters per data byte).
6) Checksum (cc)=Checksum of count, address, and data. It is the least significant byte of one's complement of the sum of the values denoted by the two hex digit pairs for the byte count, address and data field. [8]

c) A2L file format:

An ASAP2 description file (also called A2L) contains all information on the relevant data objects in the ECU such as characteristics (parameters, characteristic curves and maps), real and virtual measurement variables and irregular dependencies. Information is required for each of these objects, such as memory address, storage structure, data type and translation rules for translating them into physical units. In addition, an A2L also consist of the parameters for communication between CAN and the ECU.[3]

III. CAN CALIBRATION PROTOCOL

A) Protocol Description:

The Controller Area Network CAN is a joint progress of Robert Bosch GmbH and Intel Corporation. CAN is used in top quality automotive control systems like engine management as well as in industrial control systems. Controller chips for CAN are accessible from various semiconductor creators. The CAN Calibration Protocol is portion of the ASAP [3] (Arbeitskreis zur **S**tandardisierung von Applikationssystemen; English conversion: Standardization of Application/Calibration Systems task force) standards. It was industrialized and introduced by Ingenieurbüro Helmut Kleinknecht, a manufacturer of calibration systems, and is used in a number of applications in the automotive industry. The CCP was occupied over by the ASAP working group and enriched with optional functions.

The CCP defines the communication of controllers with a master device using the CAN 2.0B (11-bit and 29-bit identifier),[2] which contains 2.0A (11-bit identifier) for data acquisition from the controllers, memory transfers to and control functions in the controllers for calibration .Given that functions as the CCP may be used in the areas of development of electronic control units (ECU) , systems for functional and environmental trials of an ECU, test systems and test views for the controlled devices (combustion engines, gearboxes, suspension systems, climatic regulator systems, body systems, anti-locking systems), on-board test and measurement systems of pre-series vehicles, and any non-automotive application of CAN-based distributed electronic control systems. The master device (host) is a calibration tool or a diagnostic/observing tool or a measurement system presenting the data transfer on the CAN by sending commands to the slave devices[4]. The CCP implementation supports commands for generic control with primitive memory transfers and for data acquisition. These two parts (function sets) of the communication protocol are independent and may

run asynchronously, depending on the implementation in the slave controller. It is also promising that messages of both parts are transmitted in nested order.

1)Generic Control Commands:

The commands are used to carry out functions with and in the slave device. For this purpose a continuous logical connection is established between the master device and any other station on CAN (slave device). This logical connection is effective until another station is nominated or the current station is explicitly disconnected via command. After the initializing logical connection has been made, data transfers from the master device to the slave device and from the slave device to the master device are controlled by the master device. All commands from the master device have to be prompted by the selected slave device with a handshake message (command return code or error code).

2)Data Acquisition Commands:

These protocol commands are used for continuous data acquisition from a slave device. Any CAN node may periodically transmit internal data corresponding to a list that has been configured by control commands from the master device. Data transmission is initiated by the master device and executed by the slave device.

B)Message Object:

According to the definition of the CAN all messages and data to be shifted are packed into "message objects", holding up to 8 byte of data. A message object is sent from one CAN node to other CAN nodes.

The CCP needs at least two message objects, one for each direction:

1) Command Receive Object: CRO
2) Data Transmission Object: DTO

a) Organization of Message object:

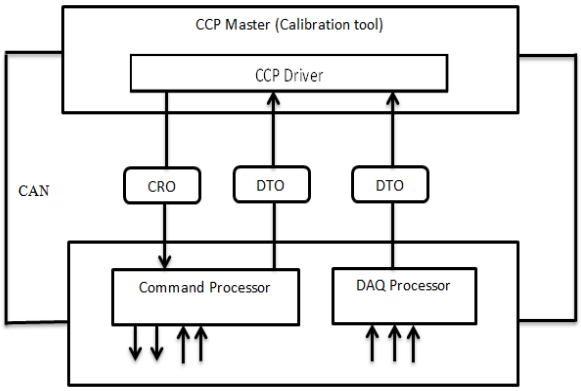


Fig.1 communication flow between CCP master and slave Devices

The CRO is used for reception of command codes and correlated parameters to carry out internal functions or memory transfers between the logically connected CAN devices. The reception of a command has to be prompted with a handshake message using the Data Transmission Object DTO (see below), and in this case a DTO is called a Command Return Message. The return code of this DTO message is used to decide whether the corresponding command has been successfully completed or not.

The assignment of message identifiers to the below defined objects is defined in the slave device description file (e.g. the ASAP2 format description file), which is used to constitute the master device. It is mentioned that the bus priority of the message objects is carefully determined in order to avoid injury to other real-time communication on the bus. Also, the CRO should achieve higher priority than the DTO. For all data transferred by the CCP no byte order for the protocol itself is defined. Because the data association is governed by the ECU's CPU, the byte ordering is defined in the slave device description file. The only exclusions are the station addresses in the TEST, CONNECT and DISCONNECT commands**.**

b)Description of Message Objects:

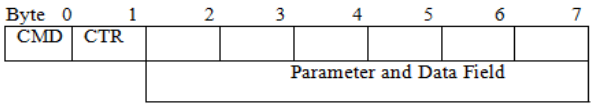
-Command Receive Object (CRO):
A Command Receive Object CRO is sent from the master device to one of the slave devices. The slave device answers with a Data Transmission Object DTO containing a Command Return Message CRM.

Structure of object:

Type	Rx Only
Size	8 byte message field
Purpose	Reception of commands in the slave device

Position Field:

Position	Type	Description
0	Byte	Command Code=CMD
1	Byte	Command Counter=CTR
2..7	Bytes	Command related parameter and data



The data length code of the CRO must always be 8. Unused data bytes, marked as „don't care„ in the command explanations, may have arbitrary values.

- Data Transmission Object (DTO):
DTO has to bring all outgoing slave device messages and data as data packets. The first byte of a data packet (i.e. first byte of the DTO´s data field) is used as the Packet ID . The

DTO is a

- Command Return Message CRM, if the DTO is referred as an answer to a CRO from the master device.

- Event Message, if the DTO reports internal slave status variations in order to raise error recovery or other services.
- Data Acquisition Message, if the Packet ID points to a conforming Object Descriptor Table (ODT) that defines which data acquisition elements (i.e. variables) are enclosed in the remaining 7 data bytes of the packet. The ODTs are initialized and modified via protocol commands.

Structure of object:

Type	Tx (and Remote Tx Request reception)
Size	8 bytes message field
Purpose	Command Return Message CRM or Event Message or Data Acquisition Message

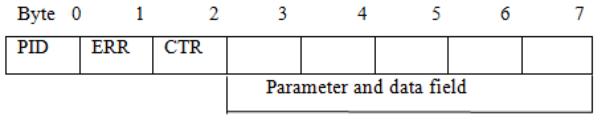
Parameters in the field:

Position	Type	Description
0	Byte	data packet ID = PID
1	Byte	data packet

Meaning of PID:

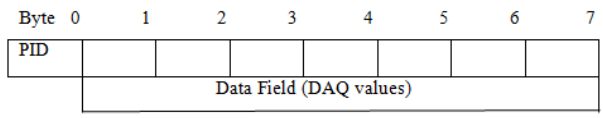
PID	Interpretation
0xFF	DTO contains a Command Return Message
0Xfe	DTO contains an Event Message
$0 \leq n \leq$ 0xFD	DTO contains a Data Acquisiton Message corresponding to ODT n

Command Return Messages have the following format:



ERR: Command Return- / Error Code. CTR: Command Counter as received in CRO with the last command. The data lenght code of the CRM must always be 8. Unused data bytes, marked as „don't care„ in the command explanations, may have arbitrary values.

Data Acquisiton Messages have the following format: PID = n; DTO contains a Data Acquisition Message corresponding to Object Descriptor.



The data length code of the DTO may be set to the actual size.

CRO and DTO commands are send using code mention in below table.

TABLE I

CONTENT OF COMMAND CODE

Code	Commands
0x01	CONNECT
0x1B	GET_CCP_VERSION
0x17	EXCHANGE_ID
0x02	SET_MTA
0x03	DNLOAD
0x04	UPLOAD
0x14	GET_DAQ_SIZE
0x15	SET_DAQ_PTR
0x16	WRITE_DAQ
0x06	START_STOP
0x07	DISCONNECT

C) Applications of Controller Area Network Calibration Protocol (CCP)

- Development of electronic control units (ECU).
- Systems for functional and environmental investigations of an ECU,
- Test systems and test stands for the controlled devices (combustion engines,gearboxes, suspension systems, climatic control systems, body systems, anti-locking systems),
- On-board test and measurement systems of pre-production vehicles
- Any non-automotive application of CAN-based spread electronic control systems.
- CCP has been designed to handle the restrictions and demands of both 8-bit microcontrollers and ECU's with high performance

D) Calibration System:

ECU calibration is the process of determining the optimal calibration boards for an engine. This multistep procedure involves designing tests, gathering data, evaluating the data, and calibrating lookup tables to model the engine. This process helps to detect the optimal balance of engine performance, emissions, and fuel economy. The calibration system has been applied to ECU in electric vehicles. Consequently, the ECU's utility has been optimized expressively and efficiently. The calibration tool is industrialized which is used for ECU development and test as well as for validation and calibration of electronically controlled systems on the test bench and in the vehicle. Also tool offers a wide variety of functions extending from pre calibration of function models on the PC, ECU flash programming, measurement data analysis, and calibration data management to automated optimization of ECU parameters

like Ignition Timing ,engine coolant temperature, Transient fueling, Low fuel pressure modifier, Waste gate control, Staged injection, Gear control, engine RPM, engine temperature**,** Absolute Throttle Position, Air Fuel Ratio, Battery Voltage, Average Speed, Fuel Used, Absolute Evaporative Vapor Pressure, Barometric Pressure, Engine Fuel Rate, Engine Oil Temp, Fuel Injection Timing, Relative Accelerator Pedal Position, Calculated Load, Exhaust Pressure, Relative Intake Air etc

The general behavior of calibration function in CCP based ECU communication with calibration tool is shown below

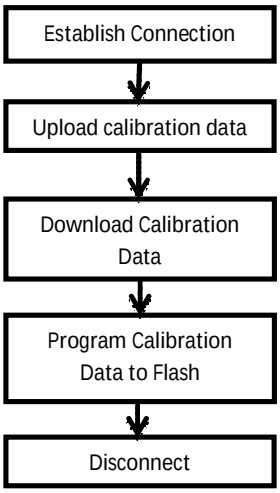


Fig.2 Flowchart of Calibration Function

Based on the calibration system[1] suppose we consider example of battery management system then the parameters to be calibrated can be generally categorized as follows.

- Parameters for battery model.
- Threshold values of the battery for fault diagnosis.
- The initial value of the bsattery's parameters, such as state of charge (SOC), internal resistance, rated capacity.
- Modified coefficient and filter factor for data acquisition program of Battery management system.

IV. ECU COMMUNICATION SYSTEM

In ECU communication system we get information about how different steps are necessary for establishing the ECU connection and which types of data base file is required for connections. ECU Measurement and Calibration (ECU MC) Toolkit [5] software available in LabVIEW [7] provides the low cost testing, data acquisition, measurement and calibration of the vehicle. below explaination of commands gives the idea about commands in Labview as well as in C. here depending upon whether A2L file contains communication parameters or not how the communication occurred is explained. in this way below flowchart provides detail flow for ECU communication based on CAN calibration protocol.

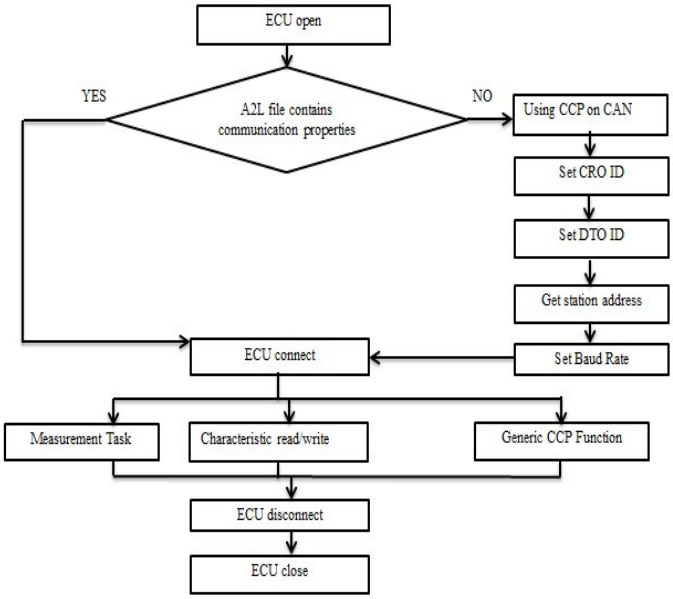


Fig.3 ECU communication Flowchart

-ECU Open:

The ECU Open function combines the opening of a selected ASAM MCD 2MC database file with the .A2L file extension and the selection of a stored ECU name. The required parameters are the ASAM MCD 2MC database path and filename, and the dedicated CAN interface if we are using CCP with CAN. The CAN interface is used for communication with the ECU. The function to open and select an ECU is MC ECU Open.vi in LabVIEW or mcDatabase Open followed by mcECUSelectEx in C.

If our ASAM MCD 2MC database file already encloses communication properties, we can directly open the communication to our selected ECU.

If the communication properties are not stored in the ASAM MCD 2MC file, the communication properties must be manually set. To establish communication through CCP with CAN, the target ECU slave should be addressed by setting the following properties.

-CRO ID:

The CRO ID (Command Receive Object) is used to send commands and data from the host to the slave device.

-DTO ID:

The DTO ID (Data Transmission Object) is used by the ECU to respond to CCP commands, and to send data and status information to the CCP master.

-Station Address

CCP is based on the idea that several ECUs can share the same CAN Arbitration IDs for CCP communication. To avoid communication conflicts, CCP defines a Station Address that

must be unique for all ECUs sharing the same CAN Arbitration IDs. Unless an ECU has been addressed by its Station Address, the ECU must not respond to CCP commands sent by the CCP master.

-Baudrate:

The Baudrate property may be missing in an A2L database file and can be set explicitly within the application. This property provides the baud rate at which communication will occur, and applies to all tasks initialized with the interface. we can specify one of the predefined baud rates, or specify advanced baud rates which refer to the settings of the Bit Timing Register 0 (BTR0) and 1 (BTR1). For more information, refer to the Interface Properties dialog in MAX, or the NI-CAN Hardware and Software Manual. The baud rate is originally set within MAX.

-ECU Connect:

The ECU Connect function establishes communication to the selected ECU through CCP using the CCP CONNECT command. It establishes a logical connection to an ECU. Unless a slave device (ECU) is unconnected, it must not execute or respond to any command sent by the application. The only exception to this rule is the Test command, in which case the slave with the specified address may acknowledge the command. Only a single slave can be connected to the application at a time. After a successful ECU Connect we can create a Measurement Task or read/write a Characteristic.

The function to open and select an ECU is MC ECU Connect.vi in LabVIEW and mcECUConnect in C.

-ECU Disconnect:

The ECU Disconnect function permanently disconnects the specified slave and ends the measurement and calibration session. When the measurement and calibration session is terminated, all DAQ lists for the device are stopped and cleared, and the protection masks of the device are set to their default values.

The function to disconnect an ECU is MC ECU Disconnect.vi in LabVIEW or mcECUDisconnect in C.

ECU Close

The MC ECU Close function deselects the ECU and closes the remaining database reference handle. MC ECU Close must always be the final ECU M&C function call. If we do not use MC ECU Close, the remaining task configurations can cause problems in the execution of subsequent ECU M&C applications.

The function to close an ECU is MC ECU Close.vi in LabVIEW. To reject the ECU and close the database reference handle in C, call the function mcECUDeselect followed by mcDatabaseClose.

V. CONCLUSION

In this paper we get information about how The CCP (CAN Calibration Protocol) is used in automotive industry for communication between ECU and calibration tool .

- Depending upon communication properties in A2L file how connection is established between ECU and calibration tool is concluded and which type of command codes are necessary for establishing these connections
- Calibration tool supports both Intel hex and Motorola S19 format and according to that how field contents are arranged for calculating checksum.
- ECU Measurement and Calibration (ECU MC) Toolkit software available in LabVIEW provides the low cost testing, data acquisition, measurement and calibration of the vehicle

REFERENCES

[1] X. Du, B. Qi, and M. Zheng, "Calibration system for battery management system based on can calibration protocol," in *Industrial Electronics and Applications (ICIEA), 2011 6th IEEE Conference on*. IEEE, 2011, pp. 2310–2312.

[2] Rahul B. Adsul "Development & Calibration of CAN for Designing Vehicle Electronic Architecture", IJECS Volume 2 Issue 6 June, 2013 Page No.2072-2085

[3] ASAP, "Can calibration protocol version 2.1," Online, 1 8-feb-1999

[4] ASAP "Applications Systems Standardization Working Group" Version 1.51, Release 2002-02-05

[5] ECU Measurement and Calibration Toolkit User Manual ECU M&C Toolkit User Manual September 2012 371601K-01.

[6 NICOLAS NAVET, YEQIONG SONG, FRANÇOISE SIMONOT-LION, AND CÉDRIC WILWERT "Trends in Automotive Communication Systems" IEEE, VOL. 93, NO. 6, JUNE 2005.

[7] www.india.ni.com, "Lab-view," Online, 2011

[8] J. G. Ganssle, *The art of programming embedded systems*. Academic Press, 1992.