Injector and scanner communication

After release of the CAN FD based CiA 1301 (CANopen FD) specification in 2017, it is time to look at the benefits, which can be offered for communication between injector and scanner as defined in CiA 425-1 and CiA 425-2 for CANopen.

Compared with the Classical CAN (ISO 11898-1:2003), CAN FD (ISO 11898-1:2015[1]) offers the following improvements. It supports bit-rates higher than 1 Mbit/s (up to 5 Mbit/s are currently specified). CAN FD supports payloads longer than 8 byte (up to 64 byte). CAN FD (CAN with flexible data-rate) frames can use a lower bit-rate during the arbitration phase, and switch to a pre-determined higher bit-rate during the data phase.

Compared to CANopen (CiA 301, based on Classical CAN as the data link layer), CANopen FD (CiA 1301, based on CAN FD) has the following aspects improved:

- CiA 1301 revamps the classic SDO (service data object) services, which were limited to unicast within the local CAN network. The new SDO service is called USDO (universal SDO), which additionally supports remote services in unicast and broadcast. A remote service refers to a USDO client requesting a USDO upload or download from one USDO server (unicast) beyond the local CAN FD network or all USDO servers (broadcast) in all CAN FD networks inter-connected via CAN FD routers.
- CiA 1301 supports PDOs of up to 64 byte in payload length.
- CiA 1301 extends the EMCY protocol from 8 byte to 20 byte, carrying more error-relevant information (e.g. time-stamp).

CAN FD

A Classical CAN data frame and a CAN FD data frame of the base format (11-bit CAN-ID) at the MAC (medium access control) sub-layer of the DLL (data link layer) are shown in table 1 and table 2. The related acronyms are shown in table 3.

Table 1: Classical CAN MAC data frame (Source: Bayer)

Frame	Classical CAN MAC Data Frame												IFS
Field	SOF	Arbitration		Control			Payload	CRC		ACK		EOF	
		CAN-ID	RTR	IDE	r0	DLC		crc sequence	del	ack	del		
Code	0	xxxx xxxx xxx	0	0	0	0 - 1000	xxxx…x	xxxx…x	1	1	1	1111 111	111…
Bits*	1	11	1	1	1	4	0 - 64	15	1	1	1	7	≥ 3

** Number of bits before bit-stuffing*

Table 2: CAN FD MAC data frame (Source: Bayer)

Frame	CAN FD MAC Data Frame															IFS
Filed	SOF	Arbitration		Control						Payload	CRC		ACK		EOF	
		CAN-ID	r1	IDE	FDF	r0	BRS	ESI	DLC		crc sequence	del	ack	del		
Code	0	xxxx xxxx xxx	0	0	1	0	x	x	0 - 1111	xxxx…x	xxxx…x	1	1	1	1111 111	111…
Bits*	1	11	1	1	1	1	1	1	4	0 - 512	17 or 21	1	1	1	7	≥ 3
Phase	Arbitration Phase							Data Phase					Arbitration Phase			

** Number of bits before bit-stuffing*

Table 3: Acronym description for table 1 and table 2 (Source: Bayer)

Acronym	Description	Acronym	Description
ACK	Acknowledgement	FDF	FD format
BRS	Bit rate switch	IDE	Identifier extension
CRC	Cyclic redundancy check	IFS	Inter-frame space
DLC	Data length code	RTR	Remote transmission request
EOF	End of frame	SOF	Start of frame
ESI	Error state indicator		

The RTR bit at end of the arbitration field in Classical CAN is transmitted dominant (0) in data frames, and recessive (1) in remote frames. In CAN FD, however, remote frames are no longer supported, so the RTR bit becomes reserved (r1), and CAN FD transmitters always transmit this bit dominant.

In Classical CAN, the control field takes up six bits: 1-bit IDE, 1-bit reserved (r0), and 4-bit DLC. CAN FD extends the control field to nine bits (highlighted in table 2) and uses the reserved bit for the FDF. In Classical CAN and CAN FD, a reserved bit is transmitted as a dominant bit. In CAN FD, the FDF bit is recessive. This is how a CAN FD controller can differentiate between a Classical CAN data frame and a CAN FD data frame, so that a CAN FD controller can “downgrade” into a CAN controller able to receive Classical CAN data frames. A Classical CAN controller is not able to handle CAN FD frames. It can be implemented, however, to tolerate CAN FD frames.

The three new bits are 1-bit reserved (r0), 1-bit BRS, and 1-bit ESI. If BRS is dominant, no bit-rate switch occurs for the data phase. Otherwise a higher bit-rate is used for the data phase. The ESI bit represents the error state of the transmitter, with dominant bit as error free and recessive bit as error active. In CAN FD, DLC continues to have four bits, which can normally represent a maximum value of 16. Therefore, a special encoding is required (see table 4) to represent payloads longer than 16 byte.

As shown in table 4, not every payload length can be represented. Specifically, the possible payload lengths are 0 to 8, 12, 16, 20, 24, 32, 48, and 64 byte. Any application payload that is not in one of those lengths must be padded with dominant bits to the next length.

In a Classical CAN data frame, the CRC field takes up 16 bit, with a 15-bit CRC sequence and a 1-bit delimiter (del). With payload length increased by 8 times, CAN FD extends the CRC sequence to 17 bit (for payloads of up to 16 byte), or 21 bit (for payloads between 20 and ▷

Table 4: DLC encoding in CAN FD (Source: Bayer)

DLC (4 bits)				Payload Length (bytes)
DLC3	DLC2	DCL1	DLC0	
0000 – 1000$_b$ (same as classical CAN)				0 – 8
1	0	0	1	12
1	0	1	0	16
1	0	1	1	20
1	1	0	0	24
1	1	0	1	32
1	1	1	0	48
1	1	1	1	64

64 byte). To further improve transmission reliability, the CRC field is bit-stuffed with a scheme called fixed bit-stuffing. The bit-stuffed CRC sequence is prepended with a stuff-bit counter (SBC), which consists of a 1-bit FSB (fixed stuff bit), 3-bit count (CNT), containing the number of stuff bits in the CRC sequence, and a 1-bit parity (PTY). The CRC sequence also starts with an FSB, followed by a 4-bit CRC segment, then another FSB and 4-bit CRC segment, and so on until all CRC bits are exhausted. The FSB code is always the opposite of its previous bit. This CRC bit-stuffing scheme makes the whole CRC field 28 bit or 33 bit long depending on the payload length, as shown in table 5 and table 6, respectively.

CANopen FD (CiA 1301)

CiA 1301 [2] specifies the basic communication layer on top of the CAN FD data link layer and is the basis for the CANopen FD specifications. There are some differences in comparison with the CiA 301 [3] CANopen specification.

The most significant difference in CiA 1301 is the redesigned SDO (service data object) protocols called USDO (universal SDO). According to CiA 301, up to 128 server SDOs with CAN-IDs defined by objects 1200$_h$ to 127F$_h$ could be configured in an SDO server's OD (object dictionary). Up to 128 client SDOs (CAN-IDs defined in 1280$_h$ to 12FF$_h$) could be configured as well. The USDO CAN-IDs are not configurable. Therefore, objects 1200$_h$ to 12FF$_h$ are reserved in CiA 1301 (see table 10). Instead, USDO adds the destination address field to its requests and responses. The CAN-ID for a request from a USDO client is always 600$_h$ + client's node-ID, and the CAN-ID for a response from a USDO server is always 580$_h$ + server's node-ID. In case of the USDO abort protocol the CAN-ID 600$_h$ + client node-ID is used, if the USDO client issues the USDO abort transfer. In case a USDO server issues the USDO abort transfer, the CAN-ID is 580$_h$ + server node-ID.

SDO services (in CiA 301) are local (within the same CANopen network) and only possible in unicast (between one SDO client and one SDO server). The USDO additionally supports remote services between different ▷

Table 5: Bit-stuffed CRC field (payload ≤ 16 bytes) (Source: Bayer)

CRC Field	SBC			Bit-stuffed CRC Sequence										Del
	FSB	CNT	PTY	FSB	CRC 16-13	FSB	CRC 12-9	FSB	CRC 8-5	FSB	CRC 4-1	FSB	CRC 0	del
Code	x	xxx	x	x	xxxx	x	xxxx	x	xxxx	x	xxxx	x	x	1
Bits	1	3	1	1	4	1	4	1	4	1	4	1	1	1

Sontheim

HIGH-END CONNECTIVITY AND DATA MANAGEMENT

TELEMATICS AND CLOUD SYSTEMS FOR IOT AND SERVICE 4.0

www.s-i-e.de

Continuous digitization for smart vehicles

Modular on-board units with Linux – ready for condition based monitoring. Including flash-over-the-air and embedded diagnostic functionality.

Sontheim IoT Device Manager and IoT Analytics Manager – for a highly secure, comfortable and individual visualization and management of your data.

Telematic ECU – COMhawk® xt

IoT Device Manager and IoT Analytics Manager

Integrated flash-over-the-air functionality

Modular on-board telematics series

Embedded diagnostics functionality

Multi-protocol support (J1939, J2534, UDS, KWP, ...)

Ready for condition based monitoring

Table 6: Bit-stuffed CRC field (20 ≤ payload ≤ 64 bytes) (Source: Bayer)

CRC Field	SBC			Bit-stuffed CRC Sequence												Del
	FSB	CNT	PTY	FSB	CRC 20-17	FSB	CRC 16-13	FSB	CRC 12-9	FSB	CRC 8-5	FSB	CRC 4-1	FSB	CRC 0	del
Code	x	xxx	x	x	xxxx	x	xxxx	x	xxxx	x	xxxx	x	xxxx	x	x	1
Bits	1	3	1	1	4	1	4	1	4	1	4	1	4	1	1	1

CANopen FD networks. The local and remote services are possible in unicast and broadcast. The data sent in one USDO upload response (read data) or a USDO download request (data to be written) can have a size of up to 56 byte. This means a 14 times higher throughput compared to the expedited SDO transfer in CANopen. However, the redesigned USDO protocol proves that CiA 1301 is not backward compatible with CiA 301. Device OEMs (original equipment manufacturer) that plan to upgrade their CAN device to a CAN FD device will have to upgrade their CiA-301-compliant CANopen stack to a CiA-1301-compliant CANopen FD stack.

Table 7: EMCY defined in CiA 301 (Source: Bayer)

Byte	0 - 1	2	3 - 7
Field	error code	error reg	device-specific

A PDO (process data object) in CiA 1301 can carry up to 64 byte of payload. Since the PDO mapping parameters (0021_h) already allowed up to 64 object elements to be mapped to one PDO in CiA 301, there is no change regarding the PDO mapping parameters. In CiA 1301, the object elements are interpreted as 64 object elements with one-byte length each, instead of one bit as in CiA 301. In other words, the level of granularity for PDO mapping is 8 bit in CiA 1301, and one bit in CiA 301. The PDO communication parameters (0020_h) are not changed either.

Table 8: EMCY defined in CiA 1301 (Source: Bayer)

Byte	0	1	2 - 3	4 - 5	6	7 - 11	12	13	14 - 19
Field	LDN	r0	CiA spec number	error code	error reg	device-specific	status	r1	timestamp

LDN – logical device number (01 to 08_h)

The EMCY (emergency message) as defined in CiA 301 is 8 byte long. In CiA 1301, the 20-byte EMCY (see table 8) carries more information about errors. The status field (byte 12, see table 9) provides the information about the EMCY error state (occurred or removed), classification (recoverable or not), and priority. If available, the timestamp contains the time of the error occurrence.

The differences in the object dictionary layout between CiA 301 and CiA 1301 are summarized in table 10.

CiA 425-1 and CiA 425-2

CiA 425-1 [4] and CiA 425-2 [5] specify the application profile for injectors in the medical field, using CANopen as the underlying communication layer. If CANopen FD would be used, which benefits would it have for the injector and scanner applications? A higher bit-rate and a longer payload would contribute to a higher message throughput. But the bit-rate for the communication between an injector and a scanner is not as demanding as in other fields. Therefore, this section will focus on the longer payload.

Heartbeat and emergency

CANopen FD supports heartbeat as the only error control mechanism. This has no impact on CiA 425, as it already defines heartbeat as the acceptable error control mechanism.

CiA 425-2 defines the lower two bytes of the EMCY's 5-byte device-specific field (see table 7 and table 11). One byte provides the error classification and the other points to a sub-index of the object 6060_h (error text). The latter provides a short description of the EMCY error code.

In CiA 1301, the error classification is a part of the EMCY protocol (see table 8 and table 9). Thus, the error classification byte in the device-specific field (CiA 425-2) would become redundant (even though the CiA 1301 error classification does not specify the warning class).

As EMCY error codes are all pre-defined in CiA 301 and CiA 425-2 specifications, and their meanings are well understood both by injectors and scanners, 6060_h is usually not supported in the injector's OD. This is also due to the fact that even a short error text in Unicode string requires to use the block SDO transfer. Therefore, this pointer byte was never used. In CiA 1301, an expedited USDO can carry a maximum of 56 byte of data, which converts to 28 Unicode characters, still too short for a meaningful error description. So, it is still unlikely that injector OEMs would support object 6060_h even in the age of CANopen FD. Therefore, the EMCY as defined in CiA 1301 can be directly used in CANopen FD applications without losing any critical information that was specifically defined by CiA 425-2.

PDO transfer improvement

First of all, none of the PDOs defined by CiA 425-2 is allowed to be RTR-triggered. Specifically, TPDO 2, TPDO 3, and TPDO 4 are timer-triggered. Thus, it would make no difference for CiA 425-2 compliant applications that the RTR bit (remote transmission request) is not supported by CAN FD. As the PDOs in CANopen FD can transmit up to 64 byte, the TPDO 2, TPDO 3, and TPDO 4 (22 byte in total, timer-triggered) could be defined as TPDO 2 (or a new TPDO). This would significantly improve the CAN traffic and simplify the scanner's message handling. Currently, these TPDOs are independently sent to the scanner periodically (frequently during an injection) and could (in each cycle) all arrive very close to each other if not at the same time.

Potential USDO usage

The expedited USDO can transfer a maximum of 56 byte of data in one transmission. Some of the

Table 9: EMCY status bit layout (Source: Bayer)

Bit	7	6	5	4	3	2	1	0
Field	reserved			error state	error class	error priority		
Code	0			0 | 1	0 | 1	$0 - 7_h$		
						0 - highest, to 7 - lowest		
					0 - recoverable 1 - non-recoverable			
				0 - error removed 1 - error occurred				

▷

Table 10: Object dictionary changes (Source: Bayer)

Object			CiA 301	CiA 1301
1000_h	Device type	object type	VAR	ARRAY
1001_h	Error register	bit-0	generic error	error FSA state
1003_h	Pre-defined error field*		defined	reserved
$100C_h$	Guard time		defined	reserved
$100D_h$	Life time factor		defined	reserved
1021_h	Store EDS		defined	reserved
1022_h	Store format		defined	reserved
1030_h	Version information		not defined	defined
1031_h	Active error history		not defined	defined
1032_h	Active error list		not defined	defined
1040_h	Port network allocation (CAN FD router)		not defined	defined
1041_h	Routing table (CAN FD router)		not defined	defined
1200_h | $127F_h$	SDO server parameters		defined	reserved
1280_h | $12FF_h$	SDO client parameters		defined	reserved
$1F80_h$	NMT startup (NMT master only)		not defined	defined

** 1003_h is replaced with 1031_h and 1032_h by CiA 1301.*

objects defined in CiA 425-2 can be restructured to take advantage of it. This object restructuring would be not backward compatible to a CANopen-based injector. But CANopen FD (CiA 1301) is already not backward compatible to CANopen (CiA 301). For example, the mandatory object 1000_h (device type) has been changed from object type "variable" to "array" (see table 10).

Table 11: EMCY defined in CiA 425-2 (Source: Bayer)

Byte	0 - 1	2	3	4	5 - 7
Field	error code	error reg	error class	text ptr	OEM-specific

Object 6003_h (set date and time) is used by the scanner to set the date and time on the injector. But this object is of type "array" with 6 sub-indexes, representing year, day, month, hour, minute and second, respectively. To set the object values requires six SDO download transmissions. When the last transmission is complete, the time (or even date) may no longer be correct. Using CANopen FD, this object could be changed to a variable with the data type Time_of_day (6 byte), which requires one single USDO download transmission. Furthermore, the new type has a precision of milliseconds, which could be crucial in the collaboration between an injector's log and a scanner's log in order to trace down communication issues between them.

The injection protocol configuring (or programming) objects (e.g. 6020_h, 6024_h, 6025_h, 6027_h, 6031_h, 6032_h) are "arrays" with each sub-index representing one phase of an injection protocol. Via USDO (e.g. expedited USDO download) a protocol parameter (e.g. flowrate) for all phases of an injection can be transmitted in one transfer. The largest data type used by the mentioned objects is Unsigned32 (e.g. 6027_h). In this case, data for 14 phases can be transmitted in one expedited USDO download. This is still far more than the number of phases, which injectors on the market currently support for one injection.

Currently, proposals for supporting of a multi-injection protocol in CiA 425-2 are under discussion among the participants of the CiA's SIG (special interest group) injection interface. Obviously, future support for the multi-injection protocol will somehow have an impact on these protocol configuring objects. One possible solution is to have these objects remain of type "array", but with each sub-index representing one injection. Consequently, each of these objects would consist of an array of injections. Each injection would contain all required phases in one sub-index. This concept of object restructuring, using object 6020_h (configured phase type) as an example, is demonstrated in table 12. To note is that one injection may have a different number of phases from the next one in the same injection protocol.

Table 12: 6020_h for a multi-injection protocol (Source: Bayer)

6020_h		Phase 1	Phase 2	Phase 3	...	Phase *m*
Sub1	**Injection 1**	Injection	Delay	Injection		
Sub2	**Injection 2**	Injection	Injection			
...		…				
Sub*n*	**Injection *n***	Injection	Injection	Delay	…	Injection

Conclusion

It is only possible for a Classical CAN controller to tolerate CAN FD frames. CANopen FD (CiA 1301) is not backward compatible with CANopen (CiA 301). Replacing a CANopen device with a CANopen FD device requires a new CANopen FD stack. Dropping support for the node-(life-)guarding error control mechanism by CANopen FD has no impact on communications between an injector and a scanner. PDO in CANopen FD can carry up to 64 byte of payload. Thus, TPDO 2, TPDO 3, and TPDO 4 in CiA 425-2 could be merged to one single TPDO to improve CAN traffic and simplify message handling at the scanner's side. The payload-increase in the USDO transfer also provides the opportunity for objects in the injector's object dictionary to be restructured so that they can be transmitted more efficiently. Injection protocol configuring objects are good candidates for object restructuring, as they constitute the most intensive SDO transmissions in a typical scanner workflow. The CANopen FD EMCY message transmits more error-relevant information and would be sufficient even without using the device-specific field currently defined in CiA 425-2. ◀

References

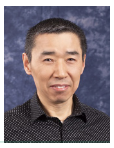
[1] ISO 11898-1:2015: Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signaling, 2015

[2] CiA 1301: CANopen FD application layer and communication profile, v. 1.0.0

[3] CiA 301: CANopen application layer and communication profile, v 4.2.0

[4] CiA 425-1: Application profile for medical diagnostic add-on modules, Part 1: General definitions, v. 2.1.0, 2011

[5] CiA 425-2: Application profile for medical diagnostic add-on modules, Part 2: Injector, v. 2.3.4, 2017



Author

Ron Kong
Bayer US
ron.kong@bayer.com
www.bayer.us