A proposed method to determine dynamic interoperability of CANopen devices

Bruce M. Decker, P.E., Schneider Electric, Inc.
Dipl.-Ing. Roland Rauch, Berger Lahr GmbH & Co. KG

Interoperability between CANopen devices is universally desired by most CANopen device integrators and other users. To this end, the CiA has defined a CANopen Conformance Test which checks EDS files and the devices for adherence to CiA specifications. However, these checks are against static requirements, implying that 2 devices which pass the Conformance Test may still not be able to actually interoperate. The CANopen protocol deliberately does not prescribe the dynamic timing parameters needed for true device interoperability, as different applications have differing requirements. Recognizing this fact, this paper does not set out to prescribe such specifications. Instead, building on TR 308, this paper describes a standard set of time value measurement definitions and a means for sharing these dynamic values between CANopen devices which provides a user with a means of quickly assessing interoperability & compatibility.

The Issue of Performance and Dynamic Interoperability

An engineer recently selected a low-cost general purpose IO device for his CANopen-based application. His bus master was from a trusted manufacturer, and the DS 401 device was supposedly CANopen-compliant. But when he connected them, the IO device never came on-line. Convinced he had received a defective device, he unplugged the CAN cable and looked for the vendor's phone number. But when he looked up at the device again, he noticed that the IO device indicators showed it was working! He reconnected the device and was able to configure it. Upon further investigation, the problem was uncovered. The IO device required 12 seconds to boot up and be ready to communicate. The bus master had a 1 second default timeout for SDO requests, after which it sent a Reset Node message to the device. The devices never had any chance to communicate until they were disconnected from each other.

Another engineer tried to transmit a PDO to an analog output device every 2 to 3 milliseconds for the application, but was getting an extraordinary amount of bus errors. Finally, the engineer called the vendor, only to be informed the device was only capable of processing an RPDO every 5 milliseconds.

This same engineer selected a general purpose input device for his application. To insure stability of his control algorithm, he set his SYNC message to be transmitted every three milliseconds. Once again, his loop would tend towards instability. It turned out that while the best SYNC response time was in the range of 1.5 milliseconds, the input device would frequently take over 4 milliseconds to respond to the SYNC.

As these true stories illustrate, if a designer wants to use a CANopen device within their application, they cannot just select a device from a catalog that will meet their application requirements. They must start there, but then they must perform a second step: they must connect the selected device into their CANopen network, and verify that it interoperates with all the other devices on the network. In many cases, it will not interoperate satisfactorily, and sometimes, not interoperate at all. This may leave the user frustrated with the device, the vendor, and possibly even their decision to use the CANopen bus for their system. Their frustration is compounded because frequently, the selected device is certified as having passed the conformance test. Yet obviously, there is no *a priori* means of knowing a device will work.

What prevents these users from being fully satisfied with their CANopen experience?

The devices were certified to be CANopen-conformant. But, they still did not interoperate with other CANopen-conformant devices, and some of them were even from the same manufacturer!

The answer to this question is fairly simple: the different device timings were incompatible with each other. There are no dynamic interoperability requirements prescribed by the CANopen specifications. A device manufacturer is free to design a CANopen device with whatever timings and performance they deem appropriate for their and/or their customer's needs.

What Does the Current Conformance Test Accomplish?

The current CiA conformance test accomplishes an important piece of overall compliance testing. As currently specified, it tests for what we call pseudo-static interoperability. That is, it looks primarily for conformance to static requirements for CANopen devices. This currently consists of checking the format of the EDS file and checking for required entries in the object dictionary. The test also exercises the SDO protocols to look for conformance to the EDS file and DS 301 requirements.

There is currently an effort in the CiA to prescribe conformance testing for DS 401 and DS 402 devices.

What About TR 308?

Technical Report 308 describes a series of bit rate dependent timing measurements for CANopen devices, such as boot up time and SDO response times. The report makes the CiA's position on prescribing timing specifications quite clear: "Regarding most other bus systems, it is fairly straight-forward to measure and publish communication performance figures for most node types. With CANopen this is not the case: the capability of CANopen to tailor the communication to the application needs makes it very difficult to determine valuable performance figures that are independent of the specific network set-up." The authors are in agreement with this position for CANopen in general. Performance limits & expectations should not be unilaterally established by the CiA, as it is counter to the goal of openness. However, we also believe that the lack of standard performance information and agreement on acceptable limits for specific classes of applications may well detract from the widespread acceptance of CANopen. Users may be unwilling or unable to do performance testing on all possible devices for their application. The authors believe, based upon input from users, that they want a statement of conformance that guarantees interoperability with the other CANopen devices in their application. Therein lies an inherent paradox, of openness vs. conformity.

What Is Needed, Long Term

To solve this issue, application-based user and device supplier communities, (e.g. motion, medical devices) must come together to agree upon performance numbers which are acceptable in the scope of the specific application. Once this is accomplished, a test can be written to test the device's conformance to the community's performance requirements.

The Goal of This Proposal

In order to prevent a proliferation of performance standards which may be contrary to each other, there first must be a basic framework established for testing and reporting of dynamic interoperability. Based upon our actual experience integrating real CANopen devices, we propose a framework for assessing & reporting dynamic interoperability parameters within the CANopen community at-large.

Important Performance Measurements

In the authors' experience, the primary performance issues were related to the following timings:

- SDO Response time;
- Minimum RPDO period (not defined in TR 308);
- SYNC Reaction Time;

- Boot-up Time from power-on (not defined in TR 308);
- Boot-up Time. (as defined in TR 308).
- PDO Response Time (as defined later in the paper).

There are certainly other timings which may provide useful information in certain cases, and we provide some suggestions later in the paper.

Modeling a CANopen Device

In order to discuss performance measurements of CANopen devices. It is essential to show an accurate model of the devices.

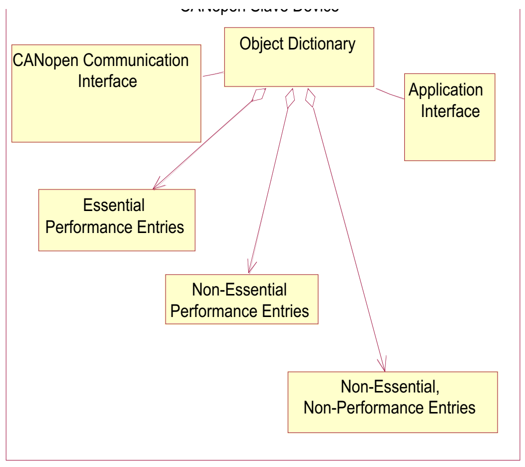
CANopen Communication Interface
Object Dictionary
Application Interface
Essential Performance Entries
Non-Essential Performance Entries
Non-Essential, Non-Performance Entries

Figure 1: Model of CANopen Slave Device

Figure 1 shows the standard model of a CANopen slave device, but extends the standard model by showing a common composition of object dictionary entries. There are the essential, performance-related objects, such as control registers, output registers, and input registers that are frequently, but not always, mapped to PDOs. Then there are the non-essential, but performance-related entries, such as the Store Parameters object (1010h). Then there are the objects which neither define the essence of a device, or have any impact on performance. Visible string identity objects such as 1008h would be in this last category, as would many diagnostic objects. It is important to distinguish between these different types of OD entries when characterizing device performance, as a longer response time for a write to the store parameters object, or reading of a visible string which may physically be stored in a relatively slow memory (e.g. a serial RAM) can unfairly skew the response time averages of other entries which are inherently more important to the user's application. For the purpose of having a shorthand means of distinguishing these different entry types, we label the essential performance entries as "Type 1", the non-essential performance entries as "Type 2", and the non-essential, non-performance entries as "Type 3".

Measurement Proposals

One of the issues with the current edition of TR 308 is that all the measurements are bit-rate and message-size dependent. In order to give a clear picture of the device performance which a user can use to select a device, the manufacturer must test at all possible bit rates and response message sizes. The user may now have too much information! Interoperability is still not guaranteed with this scheme.

For this reason, the authors propose that dynamic interoperability performance information be defined independent of bit rate and response message size.

There are two main topics that have to be considered for time measurement. The first is the reaction time of the DUT (device under test) to an event on the CAN bus (CAN message), and the second is reaction time of a physical input (e.g. a digital input). As there are several sources of influence on the timing of a device, it is necessary to know the main mechanisms by which a device is designed and built. This will be discussed in a later section.

About Time Measurement

Depending on the measurement accuracy requirements, there has to be consideration that the selected CAN bit rate has significant influence on the results. Usually, measurement on CAN is done by using the occurrence of CAN interrupts. In a CAN diagnostic tool, each received message causes an interrupt and the exact time of this interrupt can be captured. This provides precise information on the time interval between two messages, but not on the device's inherent performance.

However, the actual reaction of a device is already finished when the device starts to transmit the corresponding response message. As an example, consider an SDO response telegram which is an 8 data byte CAN message. It has, at 125 kbit/s, a duration between 0.89 and 1.04 ms. If the result of the SDO response time measurement is 2 ms, this value has a large error - 80.2% up to 108.3% with respect to measuring the actual reaction time of the device. With a lower bit rate the result is even worse. If the measured reaction time has a greater value, then the error is lessened (Figure 2). The question is, in which range will the reaction time be expected? And, how much measurement accuracy is required?

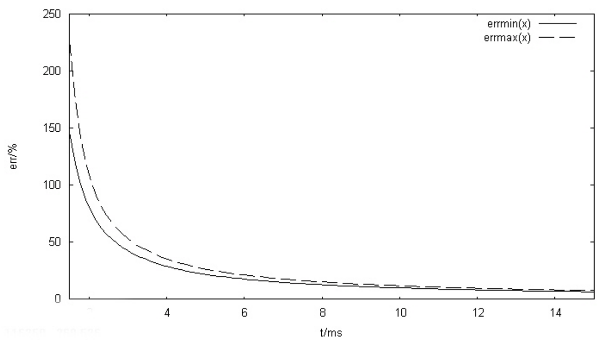

Figure 2: Measurement Error in % in relation to the time between rx- and tx-interrupt

As a result of this measurement bias inherent in the standard measurement definitions, a measurement method was developed that eliminates the dependency of the measurement result on the CAN bit rate and message size. The principle of this method is that the time of the request is captured by the tester exactly at the moment when the CAN message rx-interrupt occurs on the DUT, meaning the request has been transmitted completely. The device's response time is calculated from the tx-interrupt of the DUT's response, minus the duration of the response message. With a simple algorithm, it is possible to calculate the real duration of this message as a function of its contents. For a detailed explanation, see Figure 3 which shows the timing from the DUT's point of view.

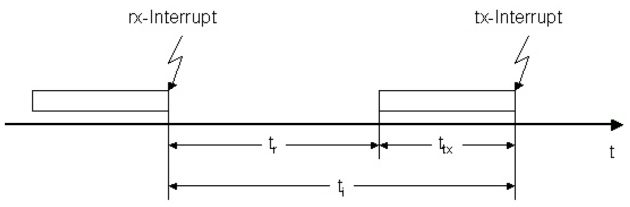

Figure 3: Reaction Time of a Device

The time t_i is the time between rx- and tx-interrupts and t_{tx} is the duration of the received message. These two values are used to calculate the device's response time, t_r:

$$t_r = t_i - t_{tx}$$

With this method, it is possible to distinguish the response time of a CAN device independently of the actual bit rate.

Physical Reaction Times

This discussion does not attempt to address the physical reaction times of a device, i.e. the amount of time from an event on the CAN bus until the physical output is at the commanded level (reverse this sequence for an input device). The authors do not address this here because it is a topic all to itself. For example, since a physical input on a DS 401 is not necessarily electrical (it can also be pneumatic, hydraulic, etc.) the authors believe it is too large a subject to address here. Therefore, this discussion is limited to observable CAN bus behavior. This does not diminish its importance to the application, however.

Device Structures

One significant aspect of timing behavior is the device's software structure and selected OS (operating system). If the software is built up in a completely deterministic way, it means the same operation always requires the same amount of time. In contrast, the measurement result will be different when compared to a system where, for example, the operating system or other design features will add some random timing components to the device's response time (see Figure 3).

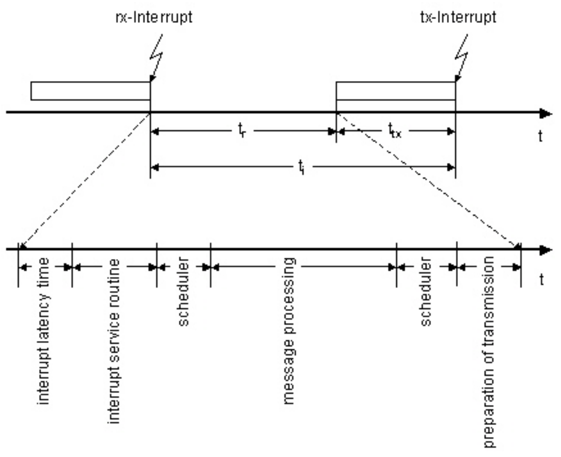

Figure 4: Different Influences on Device Response Time

The more random the different time components are, the more the total response times will vary. To explain this feature in more detail, the SDO response time will be used. A SDO read request is transmitted to the DUT, and the duration t_r will be evaluated. If the same object dictionary entry is accessed several times, and there is no other influence on the reaction time, then the maximum response time will be the same as the average response time (Figure 5). Such a device allows a very predictable access to its object dictionary. This type of device is obviously unrealistic.

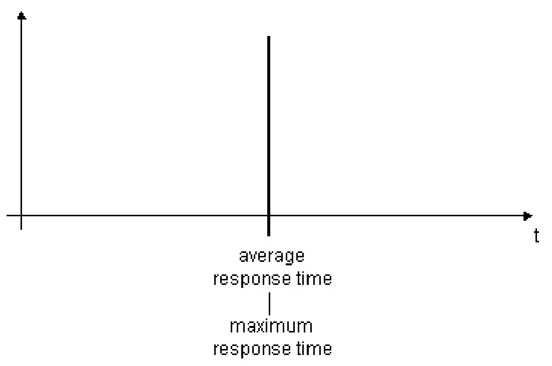

Figure 5: Predictable Device Response Time

Under more realistic conditions the device shows a statistical distribution of reaction times and the average response time does not give sufficient information about the maximum response time.

From the statistical point of view, the maximum response time in this case is defined as the smallest time in which the device responds to a request with a probability of 1.0.

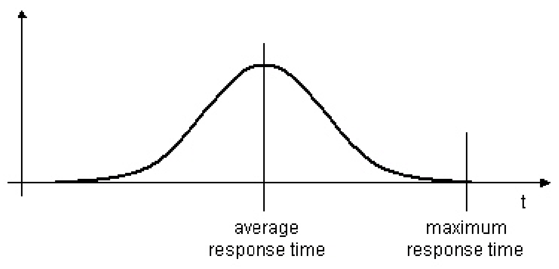

Figure 6: Distribution of Device Response Times

A similar distribution will probably occur if the SDO access is expanded to all object dictionary entries of a device, more or less independently of the device's structure. If by means of a series of measurements a reliable value for the maximum device response time can be given, then the overall performance of a CANopen network can be increased significantly. For accuracy, the SDO response time is measured for each type of OD entry shown in the model. With this knowledge, configuration of CANopen devices within the bus master can be sped up, as each device has its own SDO timeout and not just the maximum timeout of the entire network (in this case, the one of a lower - performing device). Of course, this would require the bus master to support a separate timeout set for each connected device.

If we translate this result over to PDO response time, where the principles remain the same, this will also lead to higher performing CAN communication. PDO response time in this case is defined as the duration between the reception of a command by PDO and the transmission of the corresponding result. For example, in a servo drive application, the motion controller sends a new velocity command value and gets as a response, the actual motor position back using an asynchronous PDO transmission type. If the PDO response time is highly predictable, with a known range of variance, then even control loops can be closed over the CAN bus by means of PDOs as deterministic communication can be guaranteed!

Consequences for Controller Devices

All these aspects we have discussed can only be used to effect if controlling devices are enabled to use the obtained information on device response capabilities and performance. Therefore, a sufficient means is required that makes these values accessible to configuration tools and controlling devices.

A first way could be that the controlling device or the configuration tool executes the performance measurement of the devices inside a given network. But this is really not feasible at all, as a lot of accurate measurement cycles are required which will consume a huge amount of time. The number of measurements and their precision will have a considerable impact on the quality of the results. In addition, this method is contrary to the stated goal of making CANopen simpler for the user.

Another attempt leads to the proposal to enlarge the EDS file to include these measurements, or later on, the CANopen XML device description. Having a certified performance measurement tool will allow device manufacturers to provide precise, reproducible and comparable results. As a first step, performance information could be placed inside an EDS within a [Comments] section. With a suitable parsing tool, a CANopen configuration tool would be able to derive these data from the [Comments] section. This allows the performance information to be available to the user without requiring a change to the EDS file specification.

Eventually, adding an additional section in the EDS file which contains the device’s performance values makes them available to any controlling device or configuration tool that is capable of processing EDS files. These approach would have the considerable advantage of providing the information in a way consistent with all the other information in the EDS. For example, performance entries could appear in the EDS file as follows:

```
[DevicePerformance]
AverageType1SdoResponseTime=
MaximumType1SdoResponseTime=
AverageType2SdoResponseTime=
MaximumType2SdoResponseTime=
AverageType3SdoResponseTime=
MaximumType3SdoResponseTime=
AveragePdoResponseTime=
MaximumPdoResponseTime=
AveragePdoTurnaroundTime=
MaximumPdoTurnaroundTime=
MinimumPdoPeriod=
MinimumSyncInhibitTime=
AverageRtrResponseTime=
MaximumRtrResponseTime=
AverageBootUpTime=
MaximumBootUpTime=
AveragePowerUpTime=
MaximumPowerUpTime=
```

Figure 7: EDS Extension for Performance

Proposed Framework for the Short Term

In the short term, we would suggest device manufacturers follow the following procedure for reporting dynamic interoperability numbers:

1. Divide the Object Dictionary objects into Types 1, 2 and 3.
2. Measure the SDO response time for each object in each OD entry type to get a statistically significant average & maximum time.
3. Using the default or a typical configuration, measure the other timings outlined in the paper. There is no need to measure a timing if it does not make sense for a specific device (e.g. RTR response times are not necessary for devices that do not support RTR).
4. Report these numbers, using the suggested labels, in a [Comments] section of the EDS files.

These numbers would also be useful on the device's technical data sheets.

Once the numbers are available, the application engineers can then use the numbers to perform worst-case timing analysis on their systems.

Framework for the Long Term

The authors realize this is a very early work in what promises to be an evolutionary process. We have suggested some possible changes to EDS files and CANopen master devices to improve interoperability. But we do not pretend that our ideas are the only good ideas for solving these dynamic interoperability issues. In the end, it will be the application communities coming together to set reasonable and acceptable limits for themselves which will make using CANopen devices a completely satisfactory experience.

Summary

All possible performance parameters have not been covered in this discussion, but it will be easy to enhance the performance section by more entries. As an important aspect of this discussion, the question remains: which of the entries in this section will be optional and which mandatory? This can only be answered by the application communities themselves. The authors' hope is that this work will give those communities a solid place to begin.

Acknowledgement

The authors would like to thank Messrs. Ken Lee, Steven Adiconis, Sylvain Olier, Everett Kimball, Kerry Van de Steeg, and Rick Blair of Schneider Electric, Inc., for their diligent background work which made this paper possible.

References

CiA Technical Report 308, *CANopen Performance Test*, Version 1.0, 1st December 2001, CAN in Automation e. V.

Bruce M. Decker, P.E.
Schneider Electric
American Global Products Division
Mail Stop 7-3B
One High Street
North Andover, MA, 01845-2699 USA
Phone: +1-978-975-9617
Fax: +1-978-975-9110
E-mail: bruce.decker@modicon.com
Website: http://www.modicon.com

Dipl.-Ing. Roland Rauch
Berger Lahr GmbH & Co. KG
Maybachstraße 13
88094 Oberteuringen-Neuhaus
Germany
Phone: +49 (0) 7546 920-48
Fax: +49 (0) 7546 920-80
E-mail: roland.rauch@berger-lahr.com
Website: http://www.berger-lahr.com