Node-ID assignment using LSS

Since several years the technical interest group developing the CiA 305 specification at CiA discusses which CANopen layer setting services (LSS) should be adapted to CANopen FD. This article summarizes the current work status.

An updated version of the CiA 305 specification should be published in 2020. The LSS allow low-level configurations of the used CAN (FD) bit-rate(s) and assignment of the node-IDs and network-IDs of the connected CANopen (FD) devices. Enhancing the existing services to configure switching of the bit-rate is relatively simple. Parameters can be added to support the multiple bit-rate combinations of CANopen FD. When it comes to the node-ID and network-ID assignment, a basic requirement for participating devices is the availability of the object 1018_h (identity object) with all four 32-bit sub-indexes: vendor-ID, product code, revision number, and serial number. Together these make a unique 128-bit value (further called 128-bit LSS ID) by which devices can be identified. If the LSS master knows the entire identity object, then node-ID assignment is simple. In a nutshell, the LSS master asks: "Is there anybody here with this 128-bit LSS ID?". In classic CANopen, this question is fragmented into four CAN frames, in CANopen FD the request goes into a single frame. The LSS device with a matching 128-bit LSS ID replies that it is available. From now on, this device is the only one accepting LSS master configuration commands (all others ignore the commands, as they have not been selected).

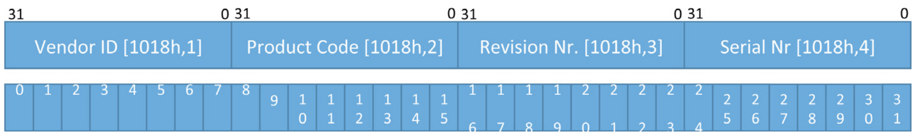


Figure 1: Splitting the 128-bit LSS ID into an array of 32 values of four bit (Source: Embedded Systems Academy)

Over the last years, there have been several solutions published to the more challenging question: how should the LSS master proceed, if the identity objects of the connected devices are not known. The last standardized approach to this challenge was the LSS Fastscan method using a binary or bit-by-bit search. The LSS master would ask such questions as "Is there anybody here whose highest bit in the 128-bit LSS ID is set?" or "Is this particular bit of your 128-bit LSS ID set or not?". All LSS devices that want to answer "yes" to such a question reply with a single "ping" message. The "ping" message is a CAN frame with a fixed CAN-ID and the length of zero. If multiple devices transmit the frame at the same time, then there are no errors on the network, as these frames "overlay" and can be transmitted in parallel by multiple devices at the exact same time. On the bottom line, such a method requires the LSS master to typically send 128 requests narrowing the requests down to finally have an exact 128-bit LSS ID selected. Even with a challenging timeout of 30 ms for each request, this takes about 4 s per device to execute.

In one of the CiA (CAN in Automation) application profiles, CiA 447 (CANopen application profile for special-purpose car add-on devices), an additional manufacturer-specific feedback method can be used to significantly shorten the detection time. Here, LSS devices use feedback messages to the LSS master, to inform it about bits in their 128-bit LSS ID. However, this method uses CAN frames with 29-bit CAN-IDs and is limited to 16 devices.

Re-thinking LSS for CANopen FD

As CAN FD is not backward compatible to Classical CAN, LSS for CANopen FD would also not need to be backward compatible, allowing the experts to re-think if other solutions would make sense. One limiting factor is, that any "ping" ("yes" answer to a request) message that multiple devices could send at the same time would still need to be transmitted in Classical CAN format with data length of zero. In CAN FD, even the messages with zero data length may have a difference in at least one bit (error state). Therefore, the messages cannot reliably overlay when they are transmitted at the same time.

For the CiA group, the open demands for a future CANopen FD LSS method are:

- The LSS device side must be simple to implement
- The method must be reliable
- If feedback messages are used, it must be ensured that these also work for a larger number of participants
- Results should be predictable and repeatable

▷

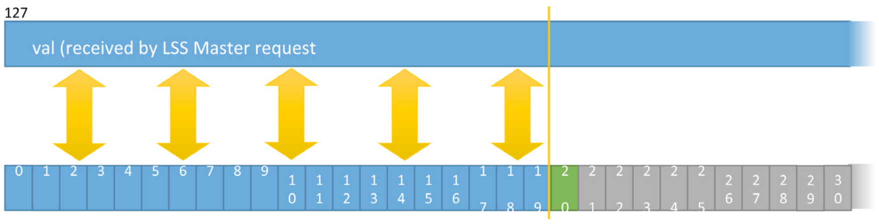
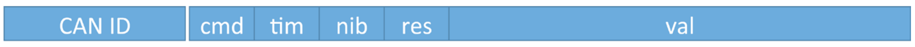


Figure 2: LSS slave handling a request with the nib = 19 (Source: Embedded Systems Academy)



CAN ID:	2021d
cmd:	(uint8_t) Command specifier, 100d or 101d
tim:	(uint8_t) Timeout (ms) used for this message by LSS Master
nib:	(uint8_t) Nibble counter from 0 to 32
res:	(uint8_t) Reserved, set to FFh
val:	(128-bit) current LSS ID comparison value

Figure 3: CAN FD LSS master switch state selective FD request (Source: Embedded Systems Academy)

- The method should be flexible, allowing shortcuts when parts of the 128-bit LSS ID are known

Regarding the predictability (if the same combination of devices is used, the node-ID assignment will be always the same for each node) it must be pointed out, that this was not supported by previous LSS methods either. Any delay in power-up time or internal processing of LSS devices can delay their answers and, thus, participation in LSS assignments. However, once a device has a node-ID permanently assigned (stored in the non-volatile memory) or is known to an LSS master (i.e. stored in a node list on the LSS master side) it can have the same node-ID on every power-up.

To keep a method flexible for faster identification of a partially known 128-bit LSS ID, the LSS device side should be simple. There is already a fast node-ID assignment method when the entire 128-bit LSS ID is known. Optimizations are still required for the few cases where only parts of the 128-bit LSS ID are known.

The new switch state selective FD service

To fulfill the switch state selective FD service, the 128-bit LSS ID has to be divided into 32 pieces (nibbles) of four bit each. The numbers of these nibbles (0 to 31) are shown in Figure 1.

In classic CANopen, LSS devices use a single CAN frame as the “ping” (“yes”) answer to an LSS master identification request. For the new service, 16 feedback messages with the CAN-IDs from 07D0$_h$ to 07DF$_h$ are used. When the LSS master asks, “What is your first nibble?”, then devices reply with 07D0$_h$ if their first nibble is zero, 07D1$_h$ if it is one and so forth until 07DF$_h$ if their nibble is 0F$_h$. The LSS master now picks the first response and packs the nibble value into the next request. Along with the question: “Those of you with the previous nibble matching the one I am sending now, what is your next nibble?” this cycle is repeated until all nibbles have been processed.

Figure 2 shows what happens if an LSS slave receives an LSS master request with a nibble value of 19. If the first 20 nibbles of the LSS slave’s own 128-bit LSS ID matches the one in the request, then it transmits the appropriate response containing the value of its own nibble 20.

Figure 3 shows the contents of the LSS master request. The first byte is the command specifier (cmd). A cmd value of 100$_d$ indicates that only unconfigured nodes should participate in the response. A cmd value of 101$_d$ indicates that all nodes should participate. This allows the LSS master to address nodes that already have a node-ID but should possibly get another one. The second value is the current timeout value (in ms) used by the LSS master (see explanation further in the article). The third is the number of the current requested ▷ nibble (0 to 32, last is needed for final confirmation) and the fourth is the current comparison value of the 128-bit LSS ID.

The tests have shown that even with 29 non-configured devices total start-up time is less than 25 s, if devices can initiate feedback transmission within a few single milliseconds. This is not a problem for embedded devices in configuration mode, as during this mode typically no or only a minimal part of the application code is executed.

The code for the LSS device does not require a complex state machine. Each LSS master request is handled by the LSS slaves individually, independent of previous requests.

Timeout handling

LSS devices with limited real-time capability shall only transmit their feedback message, if their internal processing/delay time (i.e. worst delay from receive of a CAN frame to transmitting the response) is 10 ms below the demanded value. For example, devices running a non-real-time operating system (OS) should only provide feedback, when the LSS master timeout is greater than 60 ms. This ensures that devices with slow responses do not delay the entire cycle. Slaves that do not know their response times have to use the largest specified value (100 ms).

The code for the LSS master loops through the 33 (0 to 32) stages of a complete identification cycle. With each transmission of the next LSS master request, the LSS master starts a timeout for collecting feedbacks for the last request. Only the very first feedback received is used, all others are ignored.

Conclusion

The efficiency of the scan cycle using the new switch state selective FD service depends on the LSS master timeout. The tests have shown, that a timeout of 25 ms is realistic and reliable (results in per-node-ID assignment of below 1 s), if all participating LSS devices generate their responses within 10 ms after reception of the LSS master message. For reliability it is essential, that devices not capable of such fast responses only send their response, if the LSS master message signals that a higher timeout is currently in use. ◀

Author

Olaf Pfeiffer
Emsa (Embedded Systems Academy)
info@esacademy.com
www.esacademy.de

CiA

CAN in Automation

The non-profit CiA organization promotes CAN and CAN FD, develops CAN FD recommendations and CANopen specifications, and supports other CAN-based higher-layer protocols such as J1939-based approaches.

Join the community!

- Initiate and influence CiA specifications
- Get credits on CiA training and education events
- Download CiA specifications, already in work draft status
- Get credits on CiA publications
- Receive the exclusive, monthly CiA Member News (CMN) email service
- Get CANopen vendor-IDs free-of-charge
- Participate in plugfests and workshops
- Get the classic CANopen conformance test tool
- Participate in joint marketing activities
- Develop partnerships with other CiA members
- Get credits on CiA testing services

For more details please contact CiA office at headquarters@can-cia.org

www.can-cia.org