
	<p align="center"> <u>Professorship Computer Engineering</u> Automotive Software Engineering Prof. Dr. Dr. h. c. Wolfram Hardt Hasan Saadi Jaber Aljzaere, M. Sc. Dipl.-Inf. René Bergelt </p>	 TECHNISCHE UNIVERSITÄT CHEMNITZ
Practical Unit 2	Filtering of CAN messages	Summer Semester 2023

In order to reduce the number of messages which have to be evaluated by the ECUs application logic, the CAN interface provides a means of filtering incoming messages. Consequently, only messages passing this filter will be written to the Rx buffer, other messages will be discarded.

Can message filtering is a two-step process:

1. The ID of an incoming message will be masked using the value from the mask register (e.g. relevant bits are selected)
2. The masked bits are compared to the corresponding bits of the acceptance register
If all bits match, the message is accepted otherwise it is discarded.

Since, in the practical we are using the CAN Base Frame format where the IDs are 11 bits wide, the values of the register should also be 11 bits wide (with leading zeroes).

Example

Assume that the ECU's software needs data contained in the CAN messages with message IDs 10, 11, 3 and 2 (e.g. in hexadecimal notation: 0x0A, 0x0B, 0x03 and 0x02, respectively). Begin by writing the binary representation of these messages below each other (showing only 4 bits for brevity; since all other bits are zero):

Message ID	Bits in binary representation (from MSB to LSB)			
	4	3	2	1
0x0A	1	0	1	0
0x0B	1	0	1	0
0x03	0	0	1	1
0x02	0	0	1	0

Then identify the bit positions (columns) where the value is the same for each message. In this example this would be the columns for bits 3 and 2 (highlighted).

The value of the mask register is then devised by writing a 0 at the bit positions where there are different values in the above table, and a 1 for the bit position where the value is the same **across all messages**.

The value of the acceptance register is devised by copying the values for the bit positions which are the same for all messages. The remaining bit positions (in this 4 and 1) can have any value (0 or 1) since they will not be checked. However, usually it is a good idea to set them all to 0.

The following table shows the determined values for the mask and acceptance register for the example.

Register	Bits in binary representation (from MSB to LSB)			
	4	3	2	1
Mask register value	0	1	1	0
Acceptance register value	0	0	1	0

The overall goal of filtering is to reduce the number of messages which need to be handled by application logic. Be aware that depending on the message ids required by the application code it is not always possible to configure mask and acceptance in such a way that only these messages come through. Consequently, an additional ID check in software can still be necessary.