1722B – PROPOSAL FOR LARGER BUS_ID SIZES FOR CAN & LIN

DON PANNELL
02 DEC 2019

IEEE 1722b – Larger bus_id size – The Need

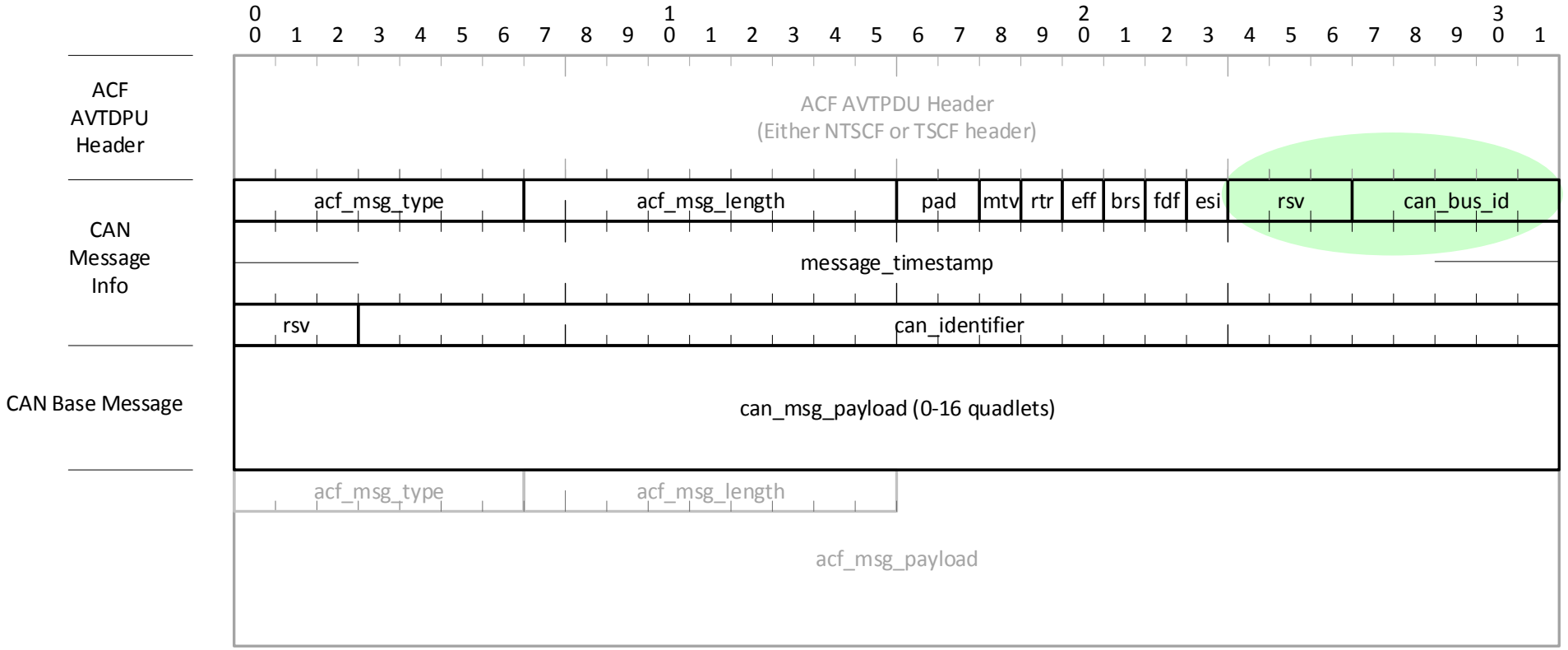
- Many cars are being designed today with many more CAN and LIN buses than in the past
- 1722-2016 supports a 5-bit can_bus_id and a 5-bit lin_bus_id
- These 5-bit fields are close to, if not already at their limits, especially for LIN
- Following the trend, this will only get worse, potentially making these 1722 formats less useful

- The following slides show some proposals. These are examples only to show the complexity (or lack thereof) of some possible solutions.

- The goal is to get consensus to add this work to the PAR for IEEE 1772b

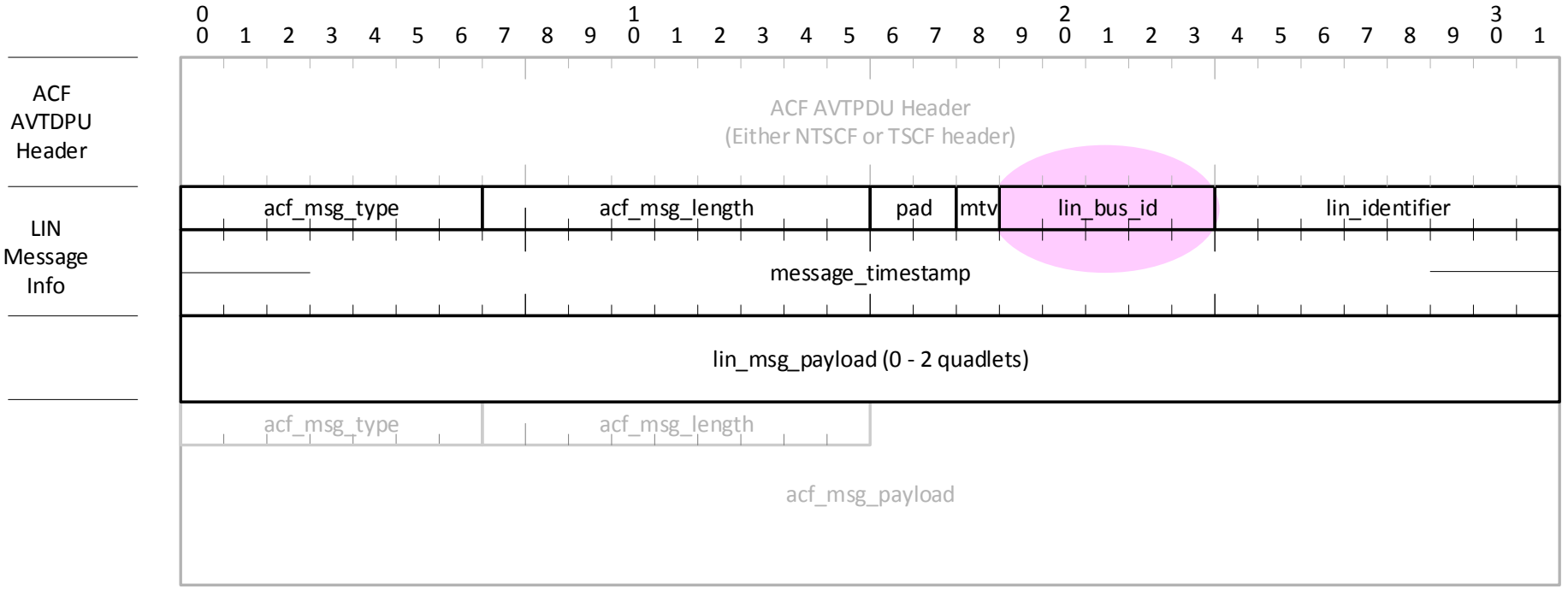
IEEE 1722b – Larger bus_id size – Proposal for CAN

- The can_bus_id is easy to extend to an 8-bit field as there are 3 reserve bits next to the upper can_bus_id bit
- Same adjustment works for both ACF_CAN & ACF_CAN_BRIEF message types



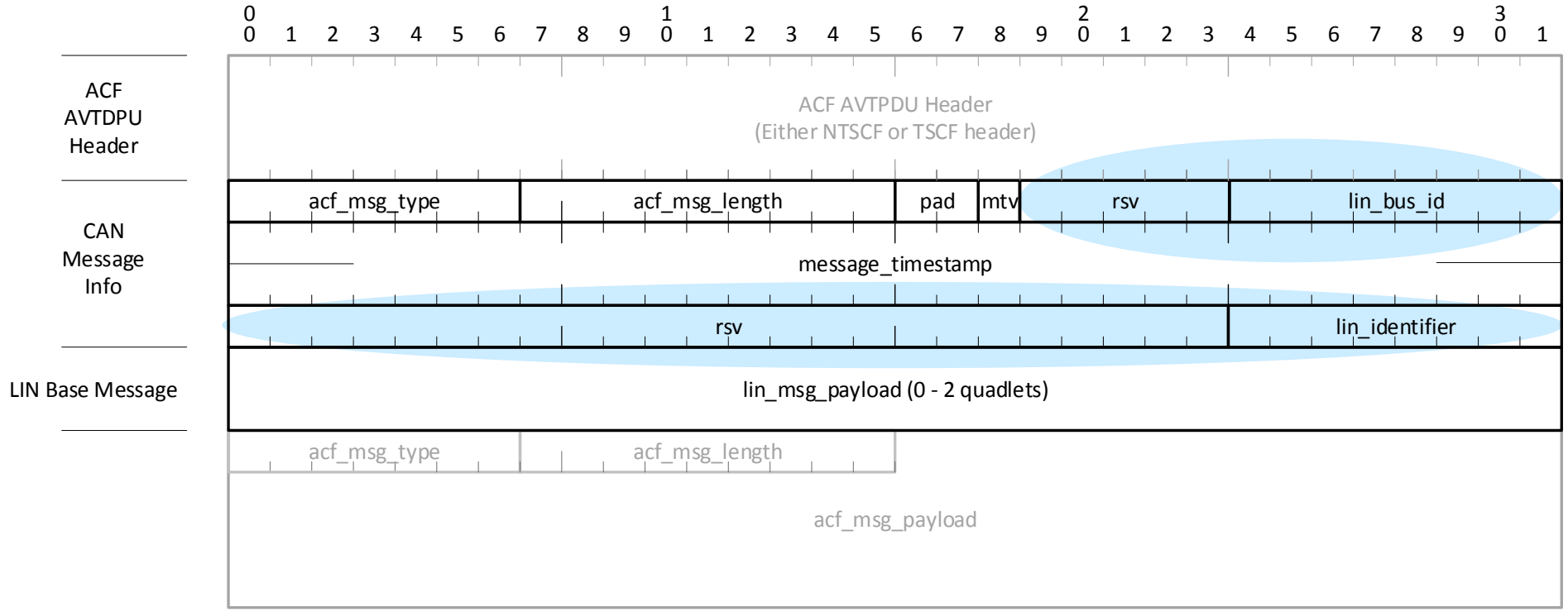
IEEE 1722b – Larger bus_id size – Proposal for LIN

- The lin_bus_id is not so easy to extend but it is the one hitting the 5-bit limit faster
- For LIN there are many possible ways to extend the lin_bus_id, all of which make the frame a quadlet bigger, unless the upper byte of the message_timestamp is used



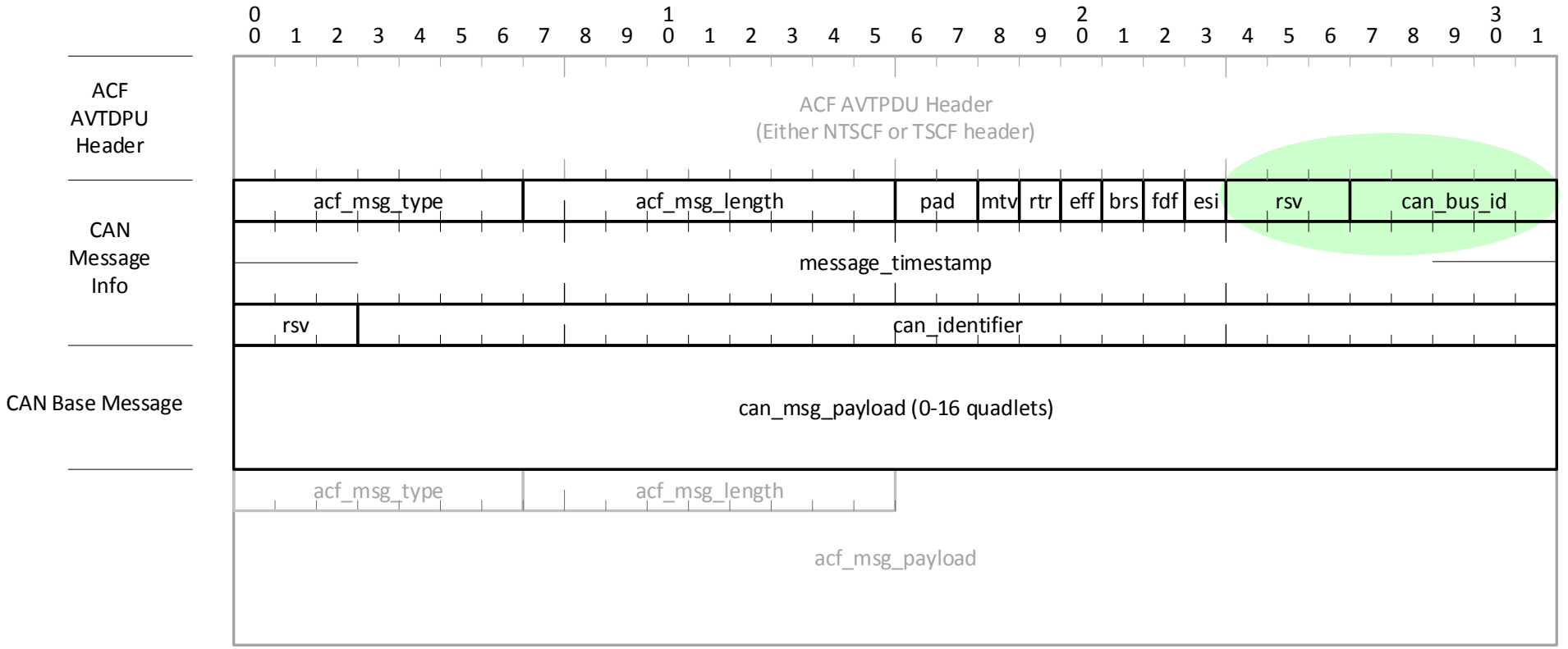
IEEE 1722b – Larger bus_id size – Proposal for LIN

- If the frame size is increased by a quadlet, then maybe it is best to make the LIN frame more like the CAN frame as shown below
  - All the common fields are in the same place as can be seen by flipping this slide with the next slide



IEEE 1722b – Larger bus_id size – Proposal for CAN - again

- This is re-shown here so that flipping between this slide and the previous slide is easy to see the common field layout between CAN & LIN



IEEE 1722b – Possible Work Items – Page 2

- For CAN, we need to decide if new acf_message_types need to be allocated as the rsv bits are now defined
  - Maybe we don't have to if we feel this is still backwards compatible

- For LIN, a new acf_message_type is needed as fields need to move & it can't be backwards compatible

- I would even propose to add two new acf_message_type's as follows
  - One for ACF_CAN_V2 – w/8-bit can_message_id
  - One for ACF_LIN_V2 – w/8-bit lin_message_id (& 1 quadlet larger)
    - A new acf_message_type for ACF_CAN_BRIEF is probably not needed as these messages need to be padded up for 8-byte CAN frames anyway

SECURE CONNECTIONS
FOR A SMARTER WORLD