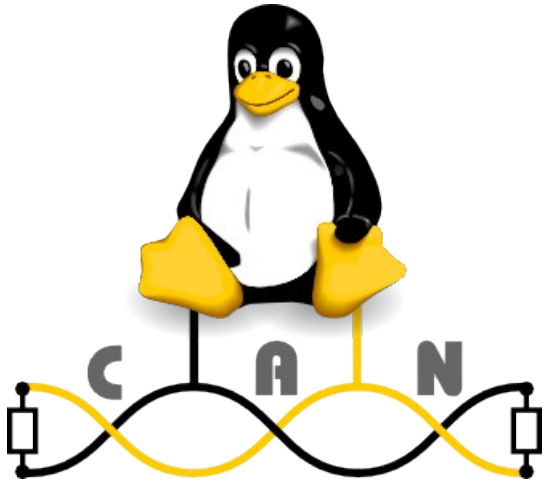


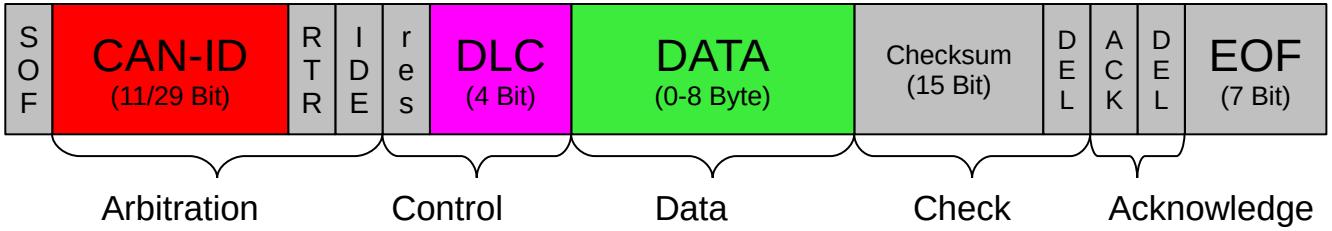
Controller Area Network

CAN CC / CAN FD / CAN XL

CAN data packets on the bus - simplified for nerds

Media access by CSMA/CR (resolution) sometimes CSMA/CA (avoidance)

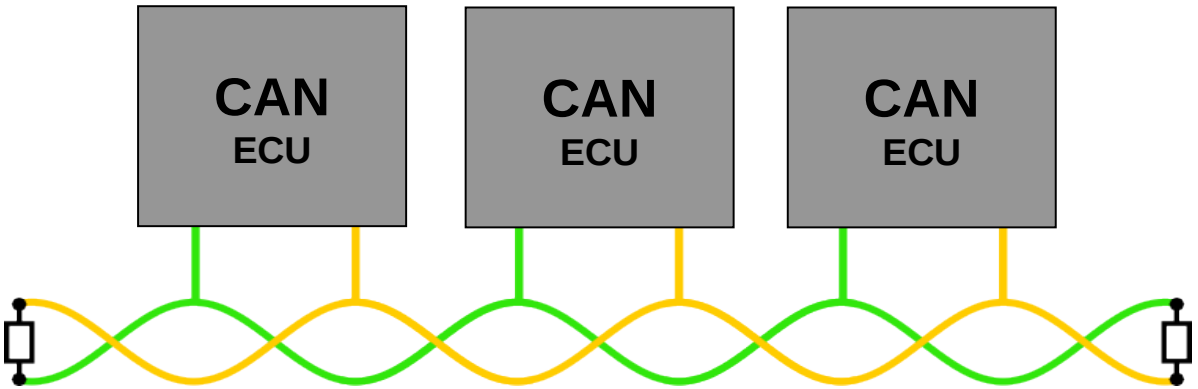
- Structure of a CAN CC (aka Classical CAN, CAN 2.0B) frame:



- Simplified: [CAN Identifier] [Data length] [Data 0..8]
- Content addressing (by CAN Identifier & CAN Bus)
- No MAC number / Node addresses / ARP / Routing – just plain OSI Layer 2
- CAN CC-only controller breaking upgrade to CAN FD (ISO 11898-1:2015)

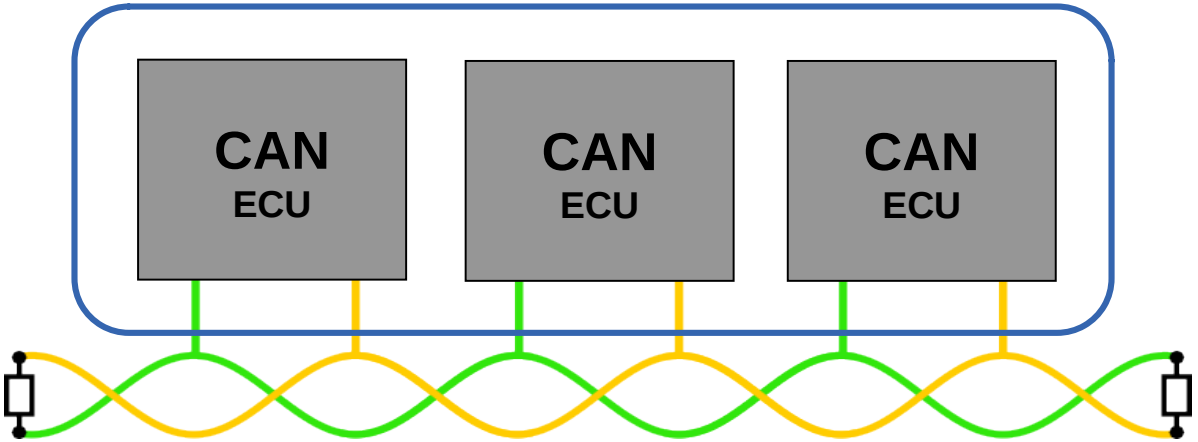
Safe but not secure

- Serial communication protocol, ISO Standard 11898
- Two wires: Unshielded twisted pair, terminated
- Transfer rate up to 1MBit/s (CAN CC) and up to ~4MBit/s (CAN FD)
- Invented by Robert Bosch GmbH, 1983
- Only defines the Media Access Layer



Multi-master bus topology

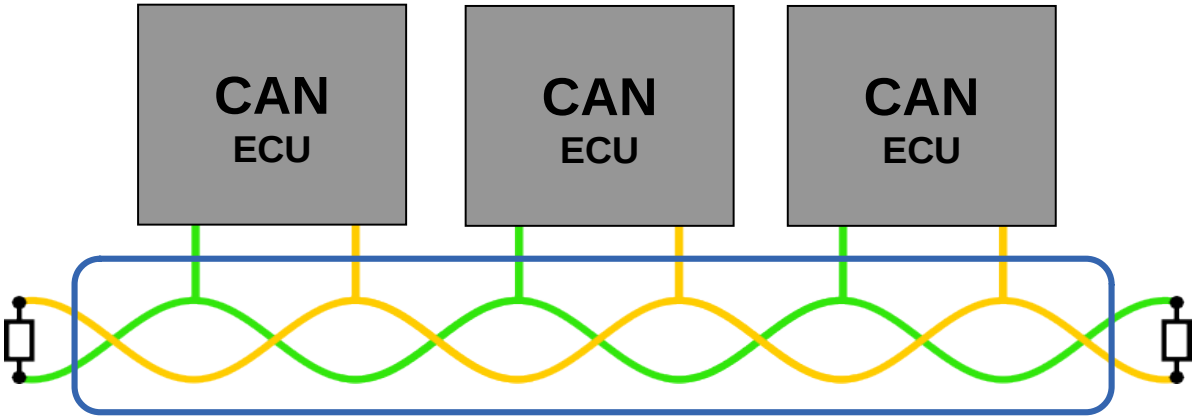
Unlike point-to-point Ethernet all nodes use the same wire



The wires are named CAN_Low and CAN_High

Differential potentials for electro magnetic interference (EMI)

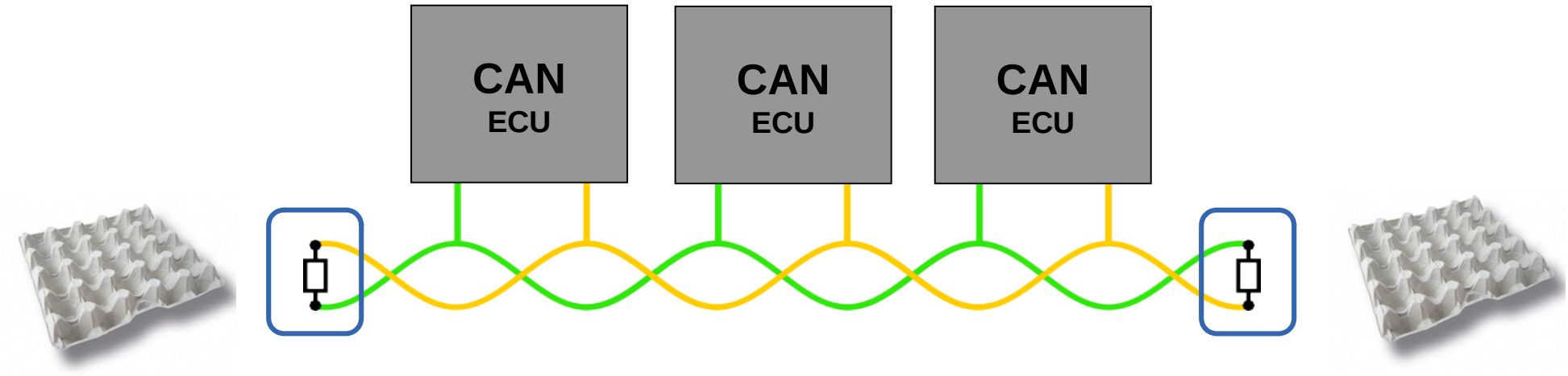
A glitch occurs on both lines at the same time ➔ difference remains stable



Unshielded twisted pair

Bus termination to prevent reflection/echos

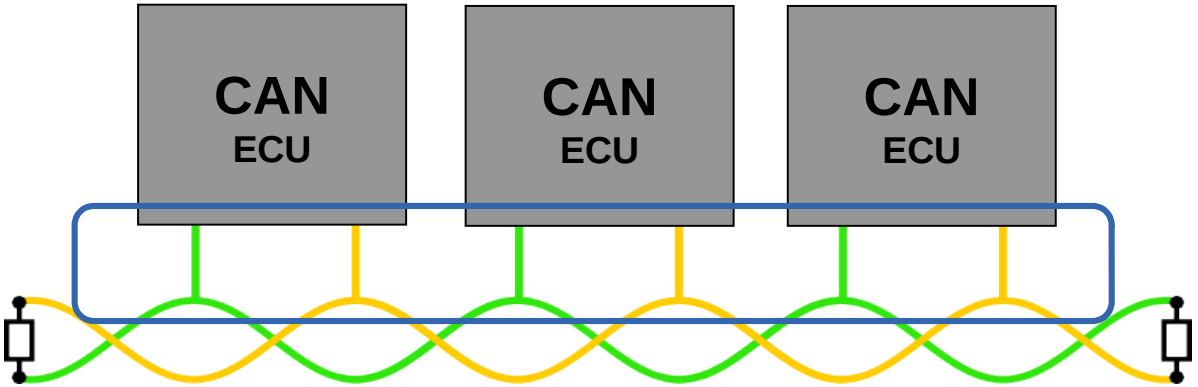
Terminate data transmission to prevent reflection/echos a the end of the wire



120 Ohms each ➔ 60 Ohms total

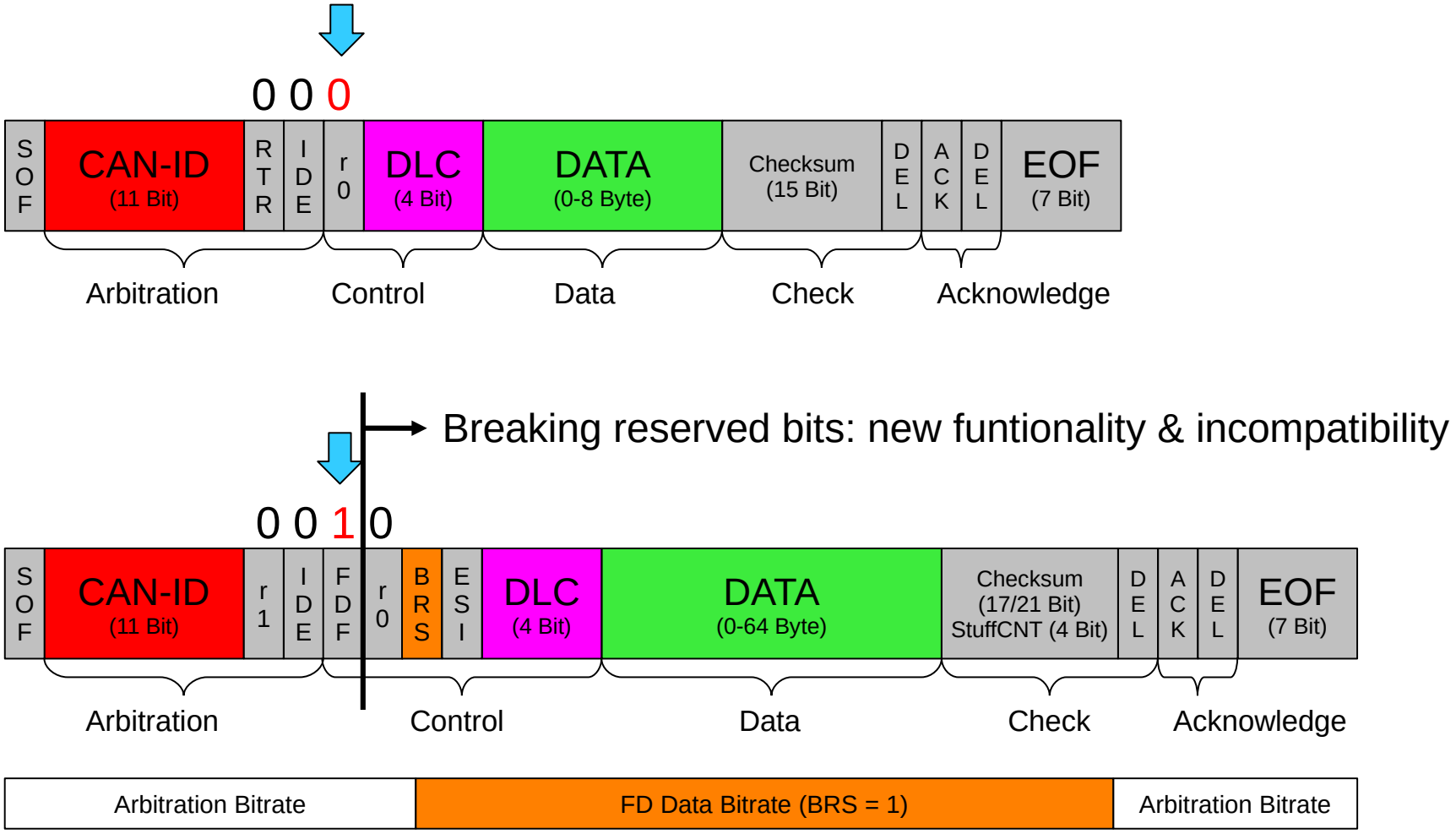
Data acknowledge by receive node

Only acknowledged CAN frames are valid CAN frames.

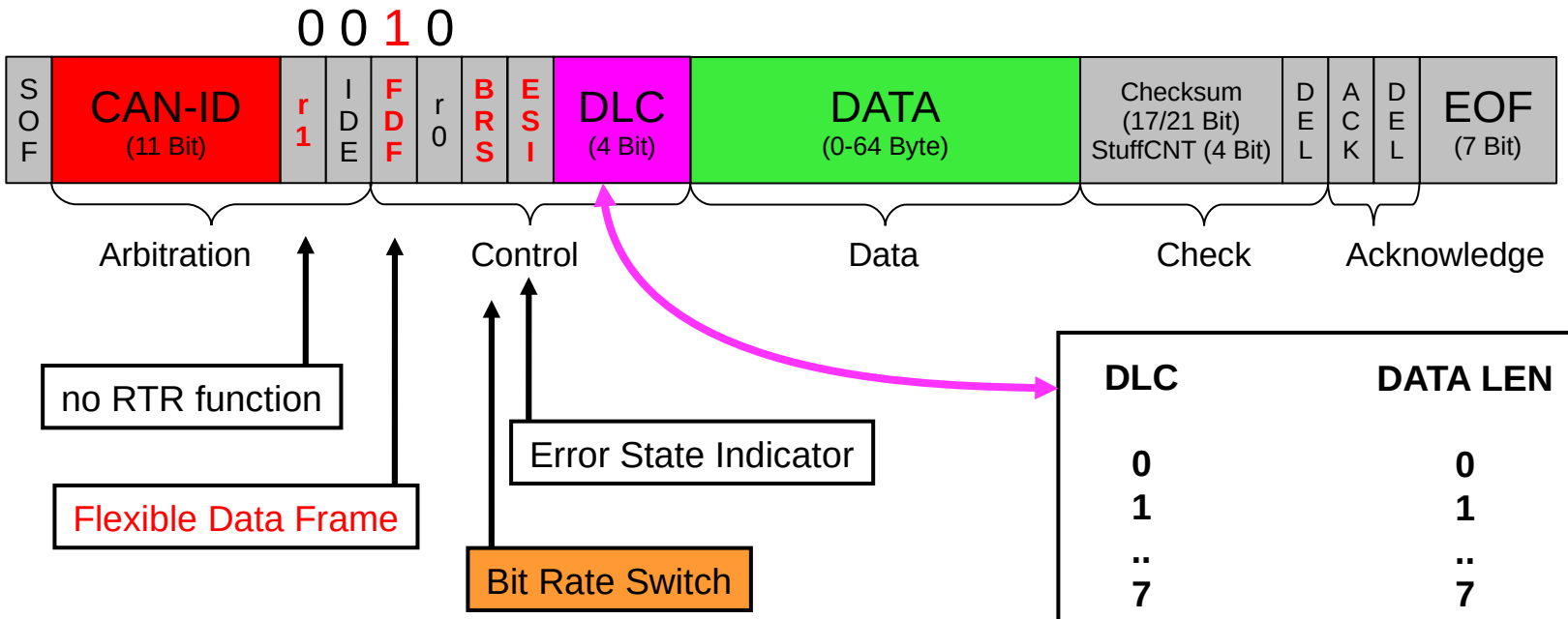


Protocol violations lead to a destroyed CAN frame (error flag)

Switching from CAN CC to CAN FD by using the reserved bit



CAN FD – new bits and definitions in detail



DLC	DATA LEN
0	0
1	1
..	..
7	7
8	8
9	12
A	16
B	20
C	24
D	32
E	48
F	64

non-linear(!!) **mapping : DLC → payload length**

Compatible data structure layout for CAN CC and CAN FD

- CAN CC data structure (the **len** element was formerly named **can_dlc**)

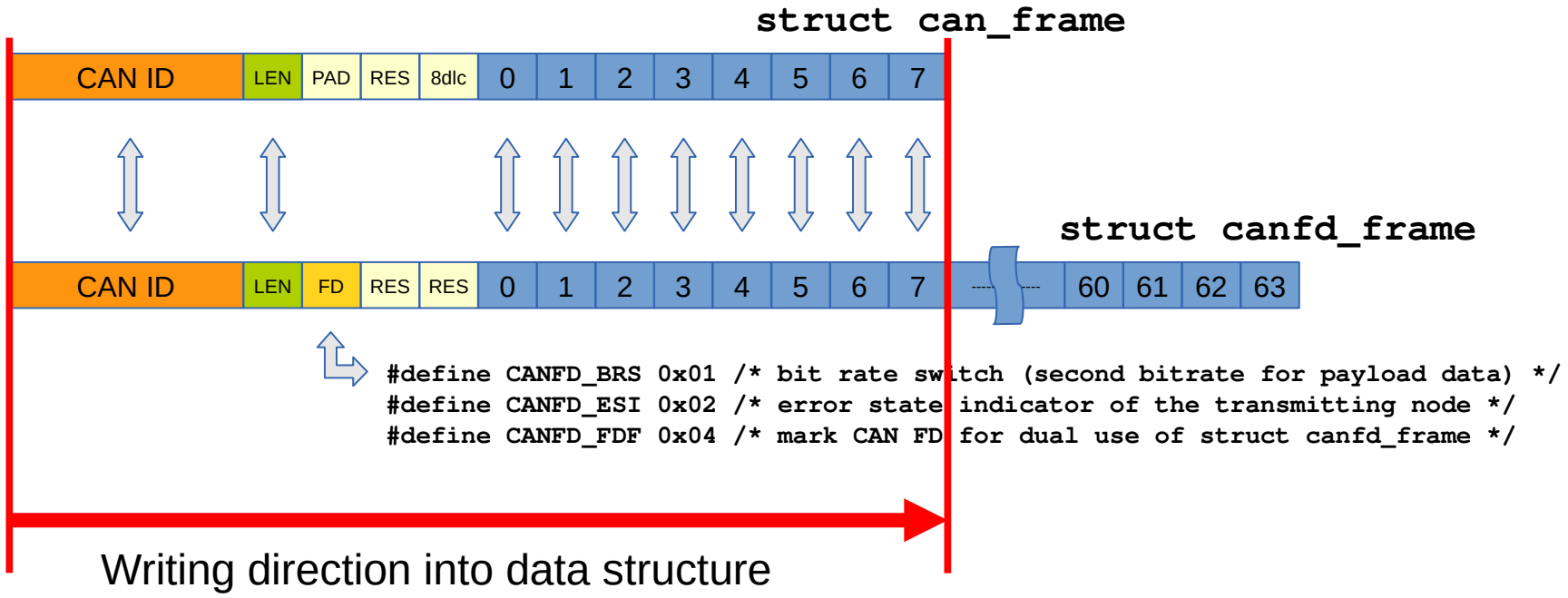
```
struct can_frame {
        canid_t can_id;  /* 32 bit CAN_ID + EFF/RTR/ERR flags */
        __u8    len;     /* frame payload length in byte (0 .. 8) */
        __u8    __pad;   /* padding */
        __u8    __res0;  /* reserved / padding */
        __u8    len8_dlc; /* optional DLC for 8 byte payload length */
        __u8    data[8] __attribute__((aligned(8)));
};
```

- CAN FD data structure

```
struct canfd_frame {
        canid_t can_id;  /* 32 bit CAN_ID + EFF/RTR/ERR flags */
        __u8    len;     /* frame payload length in byte (0 .. 64) */
        __u8    flags;   /* additional flags for CAN FD */
        __u8    __res0;  /* reserved / padding */
        __u8    __res1;  /* reserved / padding */
        __u8    data[64] __attribute__((aligned(8)));
};
```

CAN FD data structure – dual use with CAN CC layout

Writing CAN CC data into a CAN FD data structure creates valid content.



How to activate CAN FD on a CAN_RAW socket

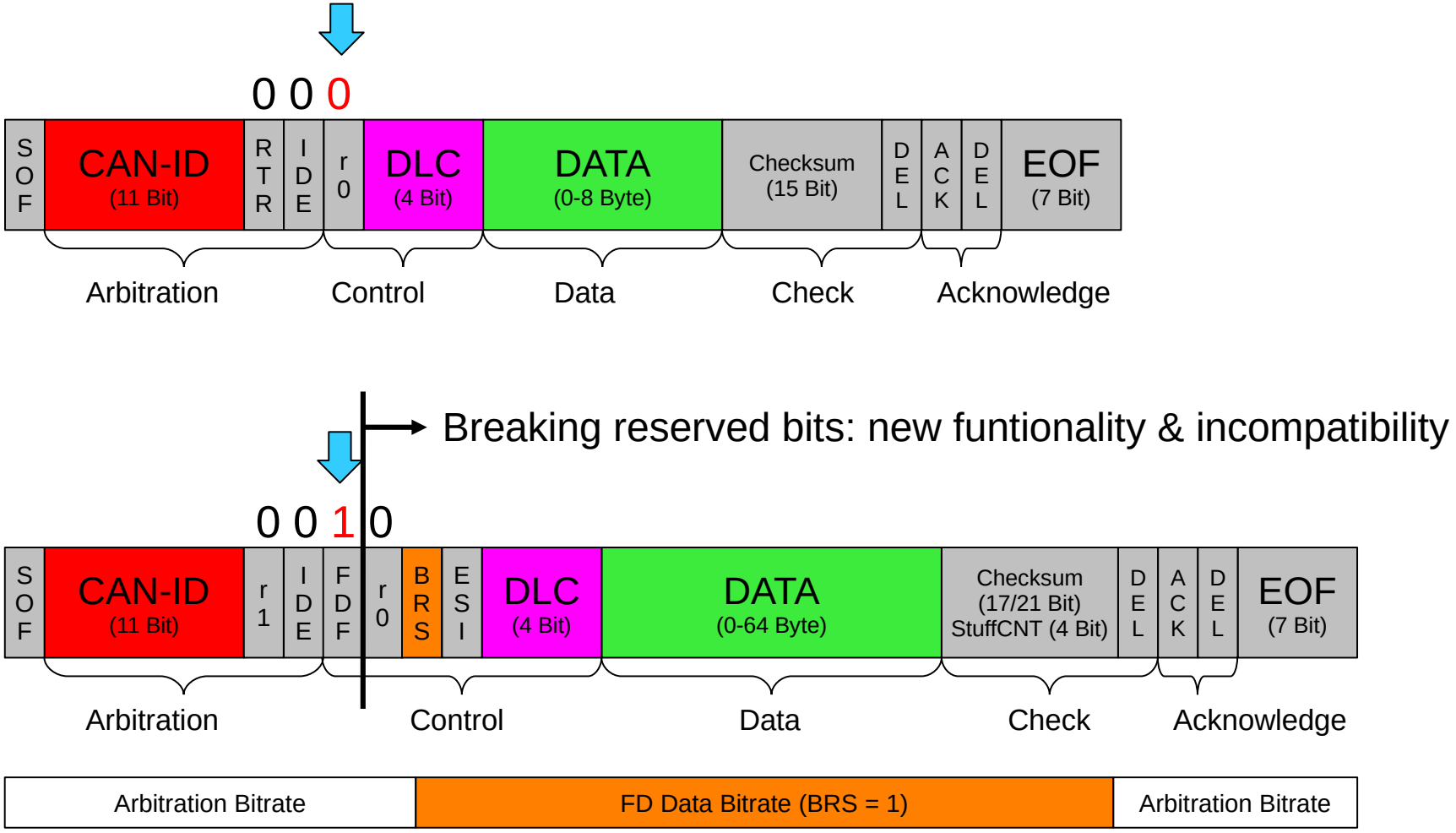
- Reading and writing CAN data structures with CAN CC

```
struct can_frame cframe;
int s = socket(PF_CAN, SOCK_DGRAM, CAN_RAW);
(..)
nbytes = read(s, &cframe, sizeof(struct can_frame));
```

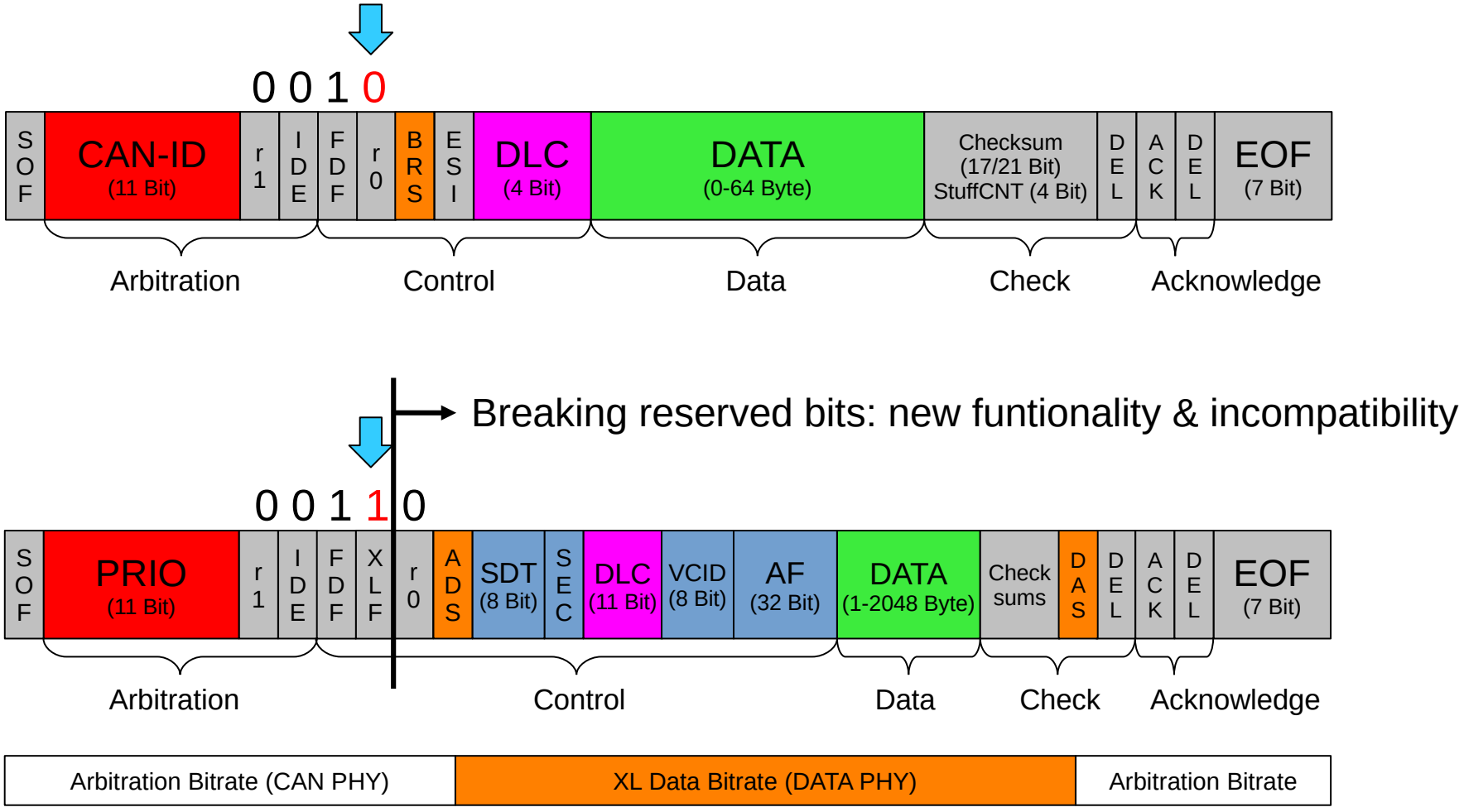
- Switch the socket into CAN FD mode with **setsockopt()** syscall

```
struct canfd_frame cframe;
int s = socket(PF_CAN, SOCK_DGRAM, CAN_RAW);
setsockopt(s, SOL_CAN_RAW, CAN_RAW_FD_FRAMES, ...);
(..)
nbytes = read(s, &cframe, sizeof(struct canfd_frame));
```

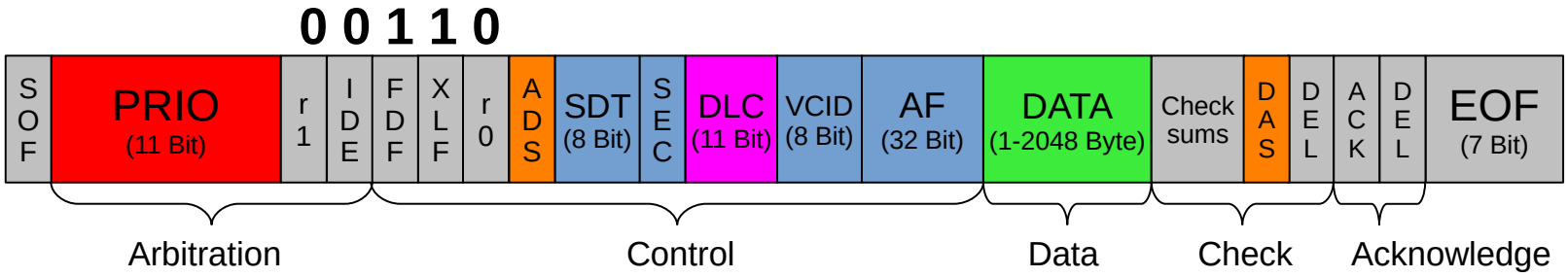
Switching from CAN CC to CAN FD by using the reserved bit (recap)



Switching from CAN FD to CAN XL by using the reserved bit



New/changed CAN XL frame content (simplified checksums)



- **PRIO** – 11 bit priority for arbitration (former 11 bit CAN Identifier)
- **ADS/DAS** – switch sequence between arbitration and data phase
- **SDT** – 8 bit Service Data Unit Type
- **SEC** – 1 bit simple extended content (e.g. for security/segmentation)
- **DLC** – 11 bit data length code (0 .. 2047) => 1 .. 2048 data bytes
- **VCID** – 8 bit virtual CAN network identifier (analogue to VLAN identifier)
- **AF** – 32 bit acceptance field (function depending on SDT)
- **DATA** – 1 .. 2048 data bytes (depenting on DLC value)

Data structure layout for CAN FD in comparison to CAN XL

- CAN FD data structure

```
struct canfd_frame {
        canid_t can_id;  /* 32 bit CAN_ID + EFF/RTR/ERR flags */
        __u8    len;     /* frame payload length in byte (0 .. 64) */
        __u8    flags;   /* additional flags for CAN FD */
        __u8    __res0;  /* reserved / padding */
        __u8    __res1;  /* reserved / padding */
        __u8    data[64] __attribute__((aligned(8)));
};
```

- CAN XL data structure (big endian representation of canfd_frame.can_id space)

```
struct canxl_frame {
        __u8    __res0; /* reserved / padding must be set to zero */
        __u8    vcid;   /* virtual CAN network identifier */
        __u16   prio;   /* 11 bit priority for arbitration */
        __u8    flags;  /* additional flags for CAN XL */
        __u8    sdt;    /* SDU (service data unit) type */
        __u16   len;    /* frame payload length in byte (1 .. 2048) */
        __u32   af;     /* acceptance field */
        __u8    data[2048];
};
```

Details of the data structure layout for CAN XL

```
struct canxl_frame {
        __u8    __res0; /* reserved / padding must be set to zero */
        __u8    vcid;   /* virtual CAN network identifier */
        __u16   prio;   /* 11 bit priority for arbitration */
        __u8    flags;  /* additional flags for CAN XL */
        __u8    sdt;    /* SDU (service data unit) type */
        __u16   len;    /* frame payload length in byte (1 .. 2048) */
        __u32   af;     /* acceptance field */
        __u8    data[2048];
};
```

- **vcid** – contains values 0x00 .. 0xFF (0x00 = untagged / no VCID)
- **prio** – shares the position and functionality with the 11 bit CAN ID in CAN CC/FD
- **flags** – shares the position with the length (**len**) information in CAN CC/FD frames
  To intentionally break with valid CAN CC/FD length values (0 .. 64) **CANXL_XLF** must be set in CAN XL frames:

```
#define CANXL_XLF 0x80 /* mandatory CAN XL frame flag (must always be set!) */
#define CANXL_SEC 0x01 /* Simple Extended Content (security/segmentation) */
```

CAN XL frame content: Service Data Unit Types (SDT)

Service Data Unit Types (0x00 .. 0xFF) – according to CAN CiA 611-1 (working draft)

- 0x00 : reserved
- 0x01 : Content based addressing
- 0x03 : CC CAN / CAN FD
- 0x04 : IEEE 802.3 (MAC frame)
- 0x05 : IEEE 802.3 (MAC frame) extended
- 0x06 : CAN CC
- 0x07 : CAN FD
- 0x08 : CiA 611-2 (Multi-PDU)
- 0x09 : AUTOSAR Multi-PDU
- 0x0A : CiA 613-2 (CANsec key agreement protocol)
- 0x0B .. 0xDF : reseved for future use
- 0xE0 .. 0xFE : manufacturer specific
- 0xFF : reserved

CAN XL summary

- CAN XL is like „Ethernet with CSMA/CR arbitration“
- High reliability (CRC/bitstuffing) with 10 Mbit/s data rate
- CAN XL controller support CAN XL / CAN FD / CAN CC
- CAN XL transceivers with switchable physical layer (arbitration/data)
- Virtual CAN interface identifiers (analogue ethernet VLANs)
- SDU (service data unit) types for multiple content use-cases