Predictable communication

Embedded systems engineering

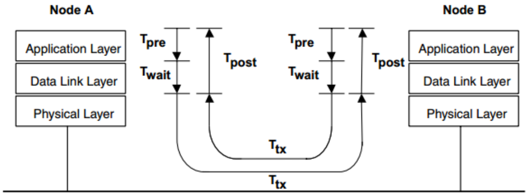
David Kendall

Northumbria University

introduction

- Many embedded systems are distributed
  - i.e. comprise multiple computing nodes that communicate via a network, e.g.
  - *Transportation*: automotive, avionic, railway
  - *Medical*: operating theatre
  - *Process-control*: manufacturing production, chemical, nuclear
- Our ability to forecast the behaviour of distributed embedded systems relies on the use of predictable communication networks
- Predictable communication requires that we can
  - give tight lower and upper bounds for the time between the release of a message by a sending process to the arrival of the message at a receiving process (*message response time*)

End-to-end delay



$$T_{delay} = T_{pre} + T_{wait} + T_{tx} + T_{post}$$

where each component represents a time as follows:

- T_{pre} – generate message and request transmission
- T_{wait} – wait for access to communication medium
- T_{tx} – transmit message across the medium
- T_{post} – retrieve message from network interface and decode

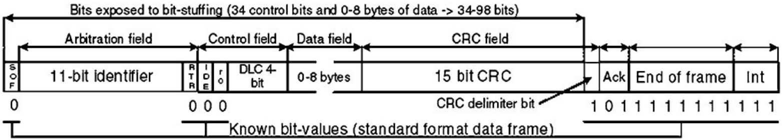
Event-triggered v time-triggered communication

- *Event-triggered* and *time-triggered* are useful concepts when considering approaches not only to scheduling computation but also to communication
- Event-triggered – processes generate and begin transmission of messages in response to events in general
  - Potentially quicker response to asynchronous events
  - Flexible support for configuration changes
- Time-triggered – processes restrict their message transmissions based on a *time event*
  - Reduced jitter
  - Tighter bounds on message response time
  - Easier to contain faults

Controller Area Network (CAN)

- Broadcast, multi-master digital bus
- Developed in mid-1980s by Robert Bosch GmbH
- Standardised in ISO/DIS 11898 and ISO 11519-2
- Operates at speeds from 20kbit/s to 1Mbit/s, dependent on bus length and transceiver speed
- Deterministic resolution of contention
- Low cost
- Easy to implement
- Widely used: automotive, medical, process-control, building control

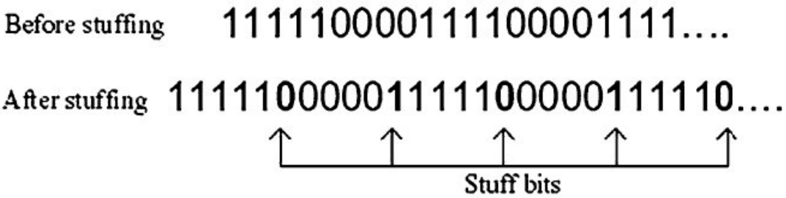
CAN Data Frame



- SOF – start of frame
- IDENTIFIER – used to identify message type and for arbitration
- RTR – remote transmission request
- IDE – identifier extension bit (0 for standard frame)
- r0 – reserved
- DLC – data length in bytes
- DATA – 0 to 8 bytes of message data
- CRC – cyclic redundancy check
- ACK – message acknowledgment
- EOF – end of frame
- IFS – inter-frame space (Int)

Bit stuffing

- CAN uses NRZ (Non-Return to Zero) for physical signalling of bits
- Efficient use of the medium but synchronisation difficult
- Bit-stuffing makes synchronisation possible
  - guarantees an upper bound on the time that can elapse before the occurrence of an edge in the signal
- CAN inserts a *stuff bit* after every 5 consecutive bits of the same polarity



Message transmission time

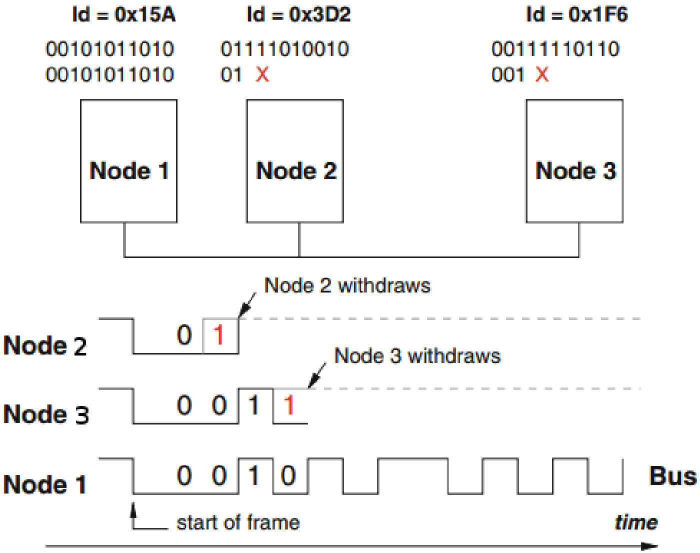
$$C_m = \left(g + 8s_m + 13 + \left\lfloor \frac{g + 8s_m - 1}{4} \right\rfloor \right) \tau_{\text{bit}}$$

- C_m – transmission time
- g – 34 for standard CAN frame
- s_m – number of data bytes in message
- τ_{bit} – transmission time for 1 bit

CAN bus arbitration

- CAN is a multi-master network in which collisions are resolved by arbitration (CSMA/CA)
- Message identifiers are used for arbitration – lowest numbered identifier wins the contest
- Once a message has won the contest it cannot be preempted, i.e. a higher priority message arriving during its transmission will have to wait for the next arbitration phase
- CAN acts as a wired-AND channel connecting all nodes, i.e. 0 bit is *dominant*, 1 bit is *recessive*
- During arbitration a transmitting nodes drops out of contention if it detects a bit on the bus that is different from the one that it transmitted

CAN bus arbitration example



Other CAN frames

- Remote transmission request
  - A node can request a message with specified identifier by transmitting a 1 in RTR slot
- Error
  - Transmitted by any node on detection of error
  - Active error frames and passive error frames
  - Recovery time from instant error detected to start of next message at most 31 bit times
- Overload
  - Six consecutive dominant bits transmitted beginning at one of the first two bits of IFS
  - Used by an overloaded receiver to prevent the start of a new message transmission
  - Not usually needed by newer controllers but required by standard

CAN error detection

- Bit error
  - Occurs when transmitting node detects a bit on the bus that differs from the bit that it transmitted (not in the arbitration field or ACK slot)
- Stuff error
  - Occurs when sixth consecutive identical bit is found in part of frame subject to bit-stuffing
- CRC error
  - Occurs when the CRC computed by a receiver differs from the one stored in the frame
- Form error
  - Occurs when a fixed form bit-field contains one or more illegal bits
- Acknowledgment error
  - Occurs when transmitting node detects a recessive bit in the ACK slot
- Node can be *error active*, *error passive* or *bus off* depending on the values of its transmit- and receive-error counters

Acknowledgements

- M. Di Natale, H.Zeng, P. Giusto, A. Ghosal, *Understanding and using the Controller Area Network Protocol*, Springer Verlag, 2012