Robust and Energy-Efficient Real-Time Systems

D7020E

Lecture 7:
Real-time communication
A concurrent system

Time

objects
A distributed system...
A distributed system...

Time

node

node

node

comm. delay

blocked

objects
A distributed system...

Time

node

node

node

comm.
delay

blocked

objects

no common clock

T
A distributed system...

- Time
- node
- node
- node
- blocked
- comm. delay
- data corruption / loss
- no common clock
- T

Objects
A distributed system...

Time

node

node

failed node

comm.
delay

blocked

objects

D7020E
Distributed system characteristics

- Physically distributed nodes
  - Separate compilation & deployment

- No common memory
  - Only message-based communication

- Non-neglectable communication delay
  - Affects end-to-end response times

- No common clock
  - Each node has its own notion of time

- No delivery guarantees
  - Data loss and partial failures
Distributed Timber systems

External events

Baselines = event reception
On communication delay

Sender baseline

Sender deadline

Receiver baseline

communication

\( m \)

\( m' \)

Separately scheduled!
On communication delay

total communication delay

predictable

receive
interrupt

predictable

predictable

pack

wait in queue

transmit

unpack

transmit
command

predictable

predictable?

Depends on protocol!
On communication protocols

- Mostly presented as layers in a protocol stack (c.f. the OSI model, or TCP/IP)
- Common assumption: an unknown number of nodes, with unknown behavior, share the network
- Another common assumption: a late delivery is better than a failed one, leading to protocols that rely on retransmissions to achieve reliability in presence of losses
- While these are valid assumptions for general purpose networks & the Internet, a consequence is that real-time guarantees cannot be given for such networks
Real-time communication

- Requires problem space to be restricted:
  - Bounded number of communicating nodes
  - Bounded message delays; that is, bounded (and low!) number of retransmissions (corruption $\approx$ loss)
  - Bounded drift rate between any two node clocks
- Enables full communication delays to be predicted
- Enables failures to be detected using timeouts
- I.e.: timely delivery is better than 100% reliability
- Note: reliability can be built on top of communication with bounded delay if needed, but not vice versa!
Real-time communication

- Bounded number of nodes means local area networks (LANs) in practice
- Interconnected LANs possible – limited real-time routing ok if network structure is statically known
- Two special addressing modes well suited for LANs:
  - Broadcast – send to all nodes on the network
  - Multicast – send to all nodes belonging to some identified group
- With LAN communication, queueing time is only determined by the underlying medium access protocol
MAC protocols

- MAC (Medium Access Control) protocols ensure that
  - only one node is using the communication medium at a time
  - those nodes that wish to use the medium are offered to do so in an orderly fashion

- As such, a LAN with a MAC protocol acts as a shared resource for which the connected nodes compete

- This resource is *not controlled by the local task schedulers*, though, but by specific networking devices & drivers

- Clearly, the *scheduling algorithm* used by the chosen MAC protocol will affect communication timing
Communication scheduling

- Contention-free communication
  - Nodes use a *predetermined time-slot* to send their data
  - Example: TDMA (time-division, multiple access)

- Token-based communication:
  - Nodes send their data in a *predetermined order*
  - Example: Token Ring (IEEE 802.5)

- Collision-based communication:
  - Nodes *content* for the shared medium at run-time
  - Example: Ethernet (IEEE 802.3), CAN (Controller Area Network)
TDMA

- Time-division, multiple access
- Run the bus as a time-triggered process
- Local nodes are typically time-triggered as well, synchronized to the bus period
The TDMA example: TTP

- The Time-Triggered Protocol (H. Kopetz, TU Wien)
- Advanced hardware/software protocol that uses dual time-shared buses for increased robustness

The Single Fault Hypothesis: Any electric/electronic component in the system can fail, but no two independent faults will occur within a certain amount of time.

- Widely deployed in the avionics industry, e.g. to connect flap controllers to its sensors & actuators
- A differently packaged variant called FlexRay is targeting the automotive industry
Ethernet

- Original network medium used a *coaxial cable* shared by all connected nodes

- The Ethernet MAC protocol uses CSMA/CD (*Carrier Sense, Multiple Access with Collision Detection*) technique:
  - *Senders only send when the medium is silent*
  - *Two senders starting simultaneously cause detectable collision*
  - *At collision detect: wait a random length of time and retry*

- Due to the built-in repetitions and randomized waiting times, original Ethernet gives *no timing guarantees*
Switched Ethernet

- Improved variant, uses private half- or full-duplex *twisted pair* cables connecting each node to a shared switch:

- Protocol still based on CSMA/CD, but if local links are full duplex, *collisions cannot occur!*

- Instead, *buffer memory* in the switch *may overflow*

- Still, for $N$ known nodes and a switch fitting at least $N$ packets, each node can send a packet every $N \cdot t_{packet}$ units without overflow, and with a *delay bound* of $N \cdot t_{packet}$ units
CAN

- The Controller Area Network (Bosch 1991, SAE 1993)
- Also based on contention for a shared bus, but with a controlled arbitration phase that avoids destructive collisions

- Simple but sturdy cabling: just two wires (vehicle is gnd)
- Used extensively in today's cars to connect embedded controllers and their sensors/actuators, for example in brake anti-locking systems
CAN arbitration

- A node wishing to send a packet of CAN must first initiate an arbitration phase
- Arbitration is based on the priority of each packet
- Nodes losing an arbitration need to wait until the bus is free and another arbitration can take place
- Packet queuing time is thus determined by its priority
- Prioritites can be assigned using RM (based on sender's packet rate) or EDF (based on packet deadline)
- Bounds on communication delay can be calculated using well-known techniques
CAN arbitration

- **Wire-OR** bus connection: sending a '1' means pulling the wire low, sending a '0' means leaving it floating

- A node accidentally sending at the same time as someone else may thus detect this fact if sending a '0' still results in a low wire

- Simultaneous sending utilized during arbitration phase:
  - Each node transmits its packet priority bit by bit, from highest to lowest significance
  - If a node sends a '0' but sees a '1' it drops out – somebody else's priority is higher
  - With unique priorities, only one node will succeed!
### CAN arbitration

<table>
<thead>
<tr>
<th>SOF</th>
<th>11-bit priority</th>
<th>control</th>
<th>0-8 bytes of data</th>
<th>checksum</th>
<th>EOF</th>
</tr>
</thead>
</table>

- 11 priority bits gives 2048 priority levels
- However, if priorities are determined locally at run-time, clashes cannot be excluded...
- In such cases, some priority bits need to be reserved for statically unique node IDs to use for arbitration
- 60 nodes (say) require 6 bits – only 5 bits for priorities...
- Even with static priorities, note the relatively high ratio between priority bits and packet payload!