Vehicular Networking
Course Overview

- Vehicular Networking

- Part 1: ...in cars
  - Overview and use cases
  - Architectures
  - Bus systems
  - Electronic Control Units
  - Security and safety

- Part 2: ...of cars
  - Overview and use cases
  - Architectures
  - Communication systems
  - Applications
  - Security and safety

Illustration © 2010 Christoph Sommer
About this slide deck

- These slides are designed to accompany a lecture based on the textbook “Vehicular Networking” by Christoph Sommer and Falko Dressler, published in December 2014 by Cambridge University Press.
- Except where otherwise noted (e.g., logos and cited works) this slide deck is Copyright © 2009-2015 Christoph Sommer, but made available to you under a Creative Commons Attribution-ShareAlike 4.0 International License.
- In brief, this means you can share and adapt the slides as long as you follow the terms set out in this license [1]. If you split this slide deck into multiple parts, make sure to include appropriate attribution in each part.
- This slide deck would not have been possible without the contributions of Falko Dressler, David Eckhoff, Reinhard German, and Kai-Steffen Jens Hielscher.
- Please leave this slide intact, but indicate modifications below.
  - Version 2015-02
    - Improved version for release on book website (Christoph Sommer)
- Updated versions of the original slide deck are available online [2].

[1] http://creativecommons.org/licenses/by-sa/4.0/
# Course Material

<table>
<thead>
<tr>
<th>Title</th>
<th>Vehicular Networking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authors</td>
<td>Christoph Sommer and Falko Dressler</td>
</tr>
<tr>
<td>Publisher</td>
<td>Cambridge University Press</td>
</tr>
<tr>
<td>Date</td>
<td>December 2014</td>
</tr>
<tr>
<td>Format</td>
<td>Hardback (also available as eBook)</td>
</tr>
<tr>
<td>ISBN-10</td>
<td>1107046718</td>
</tr>
<tr>
<td>ISBN-13</td>
<td>9781107046719</td>
</tr>
<tr>
<td>Website</td>
<td><a href="http://book.car2x.org">http://book.car2x.org</a></td>
</tr>
</tbody>
</table>

Book design © 2014 Cambridge University Press
A Short Introduction

...to Vehicular Networking
Introduction

1970s
- Electronic Fuel Injection
- Centralized Warning Lights
- Cruise Control
- Central Locking
-...

1980s
- Electronic transmission control
- Electronic climate control
- ASC (Anti Slip Control)
- ABS (Antilock Braking System)
- Phone
- Seat heater control
- Automatically dimming rear view mirror
-...

1990s
- navigation system
- CD changer
- ACC (Active Cruise Control)
- Airbags
- DSC (Dynamic Stability Control)
- Adaptive Transmission Control
- Roll stabilization
- Xenon light
- BMW Assist
- RDS / TMC
- Voice Recognition
- Emergency call
-...

2000s
- ACC Stop & Go
- BFD
- ALC
- KSG
- 42 volt system
- Internet Portal
- GPRS, UMTS
- Telematics
- Online Services
- Bluetooth
- Car Office
- Local Hazard Warning
- Integrated Safety System
- Steer / Brake-By-Wire
- i-Drive
- Lane keeping support
- Personalization
- SW update
- Force Feedback Pedal
-...

Electronics need communication

- Number of signals: \(k \times n\)
- Coordination overhead: \(q \times n \times (n-1)\)
- Number of potential communication partners: \(n \times (n-1)\)

Component failure rate

Data Source: ADAC Vehicle Breakdown Statistics 2005-2008
Bus systems

- Until the end of the 80s
  - Cars’ control units are isolated, non-networked
  - dedicated wires connect sensors and actors

- Starting with the 90s
  - digital Bus systems
  - CAN-Bus

- Today
  - Rising demands on bus systems
  - networked functionality requires more than one control unit
    - Turn signal: > 8 distinct control units
  - Real time constraints
  - Multimedia
Bus systems

- Complexity is ever increasing
  - From 5 ECUs in a 1997 Audi A6
  - To over 50 ECUs in a 2007 Audi A4
  - Current middle and upper class vehicles carry 80..100 networked Electronic Control Units (ECUs)
- Traditionally: one task $\Leftrightarrow$ one ECU
- New trends:
  - distribution of functions across ECUs
  - integration of multiple functions on one ECU
Multiple bus systems

- Gateway
  - CAN 1
    - Rear View
    - Multi Function Display
  - CAN 2
    - Climate A/C
    - Engine
    - Airbag
  - Flexray
    - Radar
    - ESP
    - Video
    - Front Display
  - MOST
    - DVD Player
    - Radio Tuner
    - MP3 System
Electronics today

- Up to 100 ECUs
- Up to 30% of value creation
- Up to 90% of Innovations
- Up to 3km of wiring for power and data
- Up to 3800 interface points
Electronics tomorrow

- Data will leave confines of single car: inter-vehicle communication
Visionary Applications

- Lane assistant
  - Simple roadside beacons support lane detection
- Lateral collision avoidance
  - More advanced beacons on cars and motorcycles help maintain minimum separation
- Accident reporting
  - Broken down cars can automatically send simple report to central server
- Intersection assistance
  - Pairs of cars automatically coordinate complex maneuvers at intersections
- Cooperative driving
  - The future evolution of autonomous driving: vehicles actively support each other’s route planning, navigating, driving
Visionary Applications

...and much (much) more:


Source: C2CCC, Aktiv-AS/VM
Challenges of communication

- Basic challenges
  - Timeliness
  - Throughput

- Communication in vehicles: stresses...
  - Robustness
  - Cost

- Communication across vehicles: also needs...
  - Interoperability
  - Reachability
  - Security
  - Privacy
Part 1
In-Car Networking
ISO/OSI Layers

- Layered communication architecture
  - One layer ↔ one function ↔ one protocol
  - Layer interacts only with immediate base layer
  - Interfaces follow rigid specification
    - commonly by standards body

- ISO/OSI layered communication model
  - Defines 7 layers
    - see next slide
  - Common architectures relax rigid guidelines
    - cf. TCP/IP
ISO/OSI Layers, Example
ISO/OSI Layers, Functions in Detail

- **Physical Layer**
  - Specifies mechanical, electrical properties to transmit bits
  - Time synchronization, coding, modulation, ...

- **Data Link Layer**
  - Checked transmission of frames
  - Frame synchronisation, error checking, flow control, ...

- **Network Layer**
  - Transmission of datagrams / packets
  - Connection setup, routing, resource management, ...

- **Transport Layer**
  - Reliable end to end transport of segments
ISO/OSI Layers, Functions in Detail

- Session Layer
  - Establish and tear down sessions
- Presentation Layer
  - Define Syntax and Semantics of information
- Application Layer
  - Communication between applications

- Our focus (in part 1 of lecture)
  - Physical Layer
  - Data Link Layer
Why bus systems?

▪ Lower cost
  ▪ Material
  ▪ Weight
  ▪ Volume

▪ Higher modularity
  ▪ customizability of vehicles
  ▪ cooperation with Original Equipment Manufacturers (OEMs)

▪ Shorter development cycles
  ▪ Re-usability of components
  ▪ Standard protocols and testing plans ⇒ less errors
History

- First micro processors in vehicles in 1980s
- Communication via point to point connections
- Simple control lines, little real data transmission
- True data transmission for connection external diagnosis equipment
- Birth of standard for character transmission
  - via K-Line (ISO 9141)
- Finally: introduction of data busses for in-vehicle communication
- Later standardized as CAN (ISO 11898)
- Use in series production models starts 1991
Overview and Use Cases

- State of the art
  - K-Line and CAN are part of On Board Diagnosis (OBD) connector
  - Enables, e.g., reading engine parameters, catcon, oxygen (lambda) sensor
  - Mandatory for newly registered vehicles in both EU and U.S.
Use Cases

- Driveline
  - Engine and transmission control
- Active Safety
  - Electronic Stability Programme (ESP)
- Passive Safety
  - Air bag, belt tensioners
- Comfort
  - Interior lighting, A/C automation
- Multimedia and Telematics
  - Navigation system, CD changer
Classification: On board communication

- On board communication
  - Complex control and monitoring tasks
    - Data transmissions between ECUs / to MMI
    - E.g., engine control, ext. sensors, X-by-Wire
  - Simplification of wiring
    - Replaces dedicated copper wiring
    - E.g., central power locks, power windows, turn signal lights
  - Multimedia bus systems
    - Transmission of large volumes of data
    - E.g., Navigation unit, Radio/CD, Internet
Classification: Off board communication

- Off board communication
  - Diagnosis
    - Readout of ca. 3000 kinds of errors
    - Garage, exhaust emission testing
  - Flashing
    - Initial installation of firmware on ECUs
    - Adaptation of ECU to make, model, extras, ...
  - Debugging
    - Detailed diagnosis of internal status
    - During development
## Classification by use case

<table>
<thead>
<tr>
<th>Application</th>
<th>Message length</th>
<th>Message rate</th>
<th>Data rate</th>
<th>Latency</th>
<th>Robustness</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control and monitoring</td>
<td>★★</td>
<td>★★</td>
<td>★★★★</td>
<td>★★★★</td>
<td>★★★★</td>
<td>★★</td>
</tr>
<tr>
<td>Simplified Wiring</td>
<td>★</td>
<td>★★</td>
<td>★</td>
<td>★★★</td>
<td>★★★</td>
<td>★</td>
</tr>
<tr>
<td>Multimedia</td>
<td>★</td>
<td>★★</td>
<td>★★★★</td>
<td>★</td>
<td>★★</td>
<td>★★★★</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>★</td>
<td></td>
<td></td>
<td></td>
<td>★★</td>
<td></td>
</tr>
<tr>
<td>Flashing</td>
<td>★★</td>
<td>★★</td>
<td>★★</td>
<td>★</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Debugging</td>
<td>★</td>
<td>★</td>
<td>★★</td>
<td></td>
<td>★★</td>
<td></td>
</tr>
</tbody>
</table>

*Vehicular Networking*
## Classification by Society of Automotive Engineers (SAE)

<table>
<thead>
<tr>
<th>Class</th>
<th>Data rate</th>
<th>Characteristics</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class A</td>
<td>~ 16 kBit/s</td>
<td>Cheap</td>
<td>Diagnosis, Sensor-Actor</td>
</tr>
<tr>
<td>Class B</td>
<td>~ 64 kBit/s</td>
<td>Error correcting</td>
<td>Networking ECUs</td>
</tr>
<tr>
<td>Class C</td>
<td>~ 1 MBit/s</td>
<td>Real time requirements</td>
<td>Drive train</td>
</tr>
<tr>
<td>Class D</td>
<td>~ 10 MBit/s</td>
<td>Low latency</td>
<td>X-By-Wire, Multimedia</td>
</tr>
</tbody>
</table>
Network Topologies

- Network topologies
  - Line
    - ✓ Cost
    - ✓ Complexity
    - □ Robustness
  - Star
    - □ Cost
    - ✓ Complexity
    - (✓) Robustness
  - Ring
    - ✓ Cost
    - □ Complexity
    - ✓ Robustness
Network Topologies

- Coupling of bus elements
  - Repeater
    - Signal amplification
    - Signal refreshing
  - Bridge
    - Medium / timing adaptation
    - Unfiltered forwarding
  - Router
    - Filtered forwarding
  - Gateway
    - Address adaptation
    - Speed adaptation
    - Protocol adaptation
Network Topologies

- Medium and Data transmission

- Medium
  - Optical
    - Fiber line
  - Electrical
    - One wire
  - Wireless
    - Bluetooth
    - WiFi

- Data transmission
  - Unicast
  - Broadcast
  - Multicast
Network Topologies

- Concurrent bus access for typical wiring
  - Shared data line connected to pull-up resistors
  - Transistors can pull data line to GND (signal ground)
- Base state
  - transistors non-conductive
  - pull up resistors raise bus level to high
- One or more ECUs turn transistor conductive
  - This connects bus to signal ground
  - Bus level is low independent of other ECUs (⇒ dominant state)
- Wired OR (if low ≡ 1) / Wired AND (if low ≡ 0)
Network Topologies

- **Wired OR**
  - Example (assuming negative logic)
    - $5V = \text{logical 0}$
    - $0V = \text{logical 1}$
Network Topologies

- Wired OR
  - Example (assuming negative logic)
    - 5V = logical 0
    - 0V = logical 1

![Diagram of Wired OR topology with 5V at measurement point, logical 0 at A and B]
Network Topologies

- **Wired OR**
  - Example (assuming negative logic)
    - 5V = logical 0
    - 0V = logical 1

<table>
<thead>
<tr>
<th>A</th>
<th>+</th>
<th>B</th>
<th>=</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ A \oplus B = C \]

A: 0V  B: 5V

Measurement point: 0V (logical 1)
Network Topologies

- Wired AND
  - Example (assuming positive logic)
    - 5V = logical 1
    - 0V = logical 0
Network Topologies

- **Wired AND**
  - Example (assuming positive logic)
    - 5V = logical 1
    - 0V = logical 0

![Diagram of wired AND logic](image)

- Measurement point: 5V (logical 1)
Network Topologies

- Wired AND
  - Example (assuming positive logic)
    - 5V = logical 1
    - 0V = logical 0

\[
\begin{array}{c|c|c|c}
A & \cdot & B & = \\
0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 \\
\end{array}
\]
Network Topologies

- Wave effects
  - Wave effects: Reflections and ends of wire or connectors
  - Non negligible at high data rates, i.e., short bit lengths
  - Propagation velocity of a signal on in-vehicle bus:
    - \( c \approx \frac{1}{3} c_0 \)
  - Signal delay on typical in-vehicle bus:
    - \( t = \frac{l}{c} \approx 200\text{ns} \)
  - Wave effects problematic if:
    - \( t_{bit} < 10t \)
- Countermeasures
  - Add terminator plugs (resistor)
  - Minimize use of connectors
Bit coding

<table>
<thead>
<tr>
<th></th>
<th>Logical 0</th>
<th>Logical 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non return to Zero (NRZ)</td>
<td>-------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td></td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Manchester (original variant)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>——</td>
<td>——</td>
</tr>
<tr>
<td></td>
<td>——</td>
<td>——</td>
</tr>
</tbody>
</table>

NRZ

Manchester

Vehicular Networking
Non Return to Zero (NRZ)

Clock

Signal

Bits

\[0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 0\]
Manchester Code

- Clock
- Signal
- Bits: 0 1 1 0 1 1 1 0 1 1 1 1 0
Reducing Electromagnetic interference (EMI)

- Add shielding to wires
- Use twisted pair wiring
- Reduce steepness of signal slope
- Use coding with few rising/falling signal edges (NRZ)

NRZ

Manchester
Clock drift

- Caused by natural variations of quartz, environment
- Receiver must sample signal at right time instant
- Clock drift leads to de-synchronization
- Bit timing has to be re-adjusted continually
- Commonly used: rising/falling signal edges

NRZ

Manchester
Bit stuffing

- **Problem**
  - When using NRZ coding, sending many identical bits leaves no signal edges that could be used to compensate for clock drift

- **Solution**
  - Insertion of extra bits after n consecutive identical bits

- **Example (stuffing width: 3)**

```
NRZ plain

NRZ w/ bit stuffing
```
Classification according to bus access

Bus access

- Deterministic
  - centralized
  - distributed
- Random
  - non collision free
  - collision free
Deterministic, centralized

- Master-Slave protocols
- Simple request/response pattern
Deterministic, distributed

- Token based protocols, TDMA protocols
Random access, non collision free

- **CSMA/CA (Collision Avoidance)**

  ![Diagram](image)

  - **Client 1**
    - d1
    - d1
    - 
    - 
  - **Client 2**
    - sense
    - sense
    - d2
    - d2
  - **Bus**
    - d1
    - d1
    - d2
    - d2

  ![Data Flow](image)
Random access, non collision free

- **CSMA/CD (Collision Detection)**

  ![Diagram showing client 1, client 2, and the bus with data transmission and collision scenarios]

  - **Client 1**
    - Data 1 (d1) transmitted
    - Collision (jam)
    - Data 1 (d1) retransmitted
  
  - **Client 2**
    - Data 2 (d2) transmitted
    - Collision (jam)
    - Backoff
    - Data 2 (d2) retransmitted
  
  - **Bus**
    - Collision
    - Jam signal
    - Data 1 (d1)
    - Data 2 (d2)
Random access, collision free

- **CSMA/CR (Collision Resolution)**

  - **Client 1**
    - a
    - d1
    - -
    - -

  - **Client 2**
    - b
    - backoff
    - b
    - d2

  - **Bus**
    - a
    - d1
    - b
    - d2

  ![Arbitration, Data 1, Arbitration, Data 2](image)
Typical structure of an ECU

- Separation by Layers
- Physical Layer: Transceiver / Bus driver
- Bus access: Communication controller
- Application layer: Microprocessor
- Commonly with bus guard for *emergency shutdown*
Main Takeaways

- Network Topologies
  - Single wire, two wire
  - Wired OR, wired AND
  - Non Return to Zero (NRZ) vs. Manchester coding
  - Clock drift, synchronization, bit stuffing

- Bus access
  - Deterministic, non-deterministic access
  - CSMA/CA, CSMA/CD, CSMA/CR
  - Bus guard
Protocols

K-Line, CAN, LIN, FlexRay, MOST, Ethernet
K-Line
The K-Line Bus

- The K-Line Bus
  - Industry standard of the 80s, much later standardized as ISO 9141
  - Numerous variants exist (esp. upwards of Link Layer)
  - Lecture focuses on ISO 14230: The KWP 2000 (Keyword Protocol)
  - Specifies Physical and Link layers
  - Bidirectional bus, communicating over 1 wire (the K Line)
The K-Line Bus

- The K-Line Bus (contd.)
  - Optional: additional unidirectional L Line
    - Allows mixed networks (using only K Line / using both K+L Line)
  - Mostly used for connecting ECU ⇔ Tester, seldom ECU ⇔ ECU
  - Logic levels are relative to on board voltage (< 20% and > 80%)
  - Bit transmission compatible to UART (Universal Asynchronous Receiver Transmitter): 1 start bit, 8 data bits, 1 stop bit, optional parity bit
  - Bit rate 1.2 kBit/s ... 10.4 kBit/s
    - Dependent on ECU, not Bus
    - Master must be able to handle multiple bit rates
The K-Line Bus

- **Protocol**
  - **Connection establishment (2 variants)**
    - **5 Baud init**
      - Master sends destination address (using 5 Bit/s)
      - ECU answers: 0x55 (01010101), keyword low Byte, keyword high Byte (with desired data rate)
    - Master derives bit rate from pattern, sends Echo (inv. High Byte)
    - ECU sends Echo (inv. Destination address)

```
Tester → ECU     ECU → Tester     Tester → ECU     ECU → Tester
> 300ms   ~ 2s   < 300ms   < 20ms   < 20ms   < 20ms   < 50ms

K-Line
  Adress byte  Sync. Byte 55h  Keyword LSB  Keyword MSB  Inv. Keyword MSB  Inverted Adress byte

L-Line
  Adress byte
```

5 Bit/s  Fixed bit rate, chosen by ECU, detected and adopted by master
The K-Line Bus

- **Protocol**
  - **Connection establishment (2 variants)**
    - **Fast init** (100 ms, Bitrate always 10,4 kBit/s)
      - Master sends *Wake Up* pattern (25 ms low, 25 ms pause)
      - Master sends *Start Communication Request*, includes dest address
      - ECU answers with keyword, after max. 50 ms
      - Keyword encodes supported protocol variants takes values from 2000 .. 2031 (KWP 2000)
  
  \[
  \begin{array}{c}
  \text{K-Line} \\
  \text{L-Line}
  \end{array}
  \quad
  \begin{array}{c}
  > 55\text{ms} \quad 25\text{ms} \quad 25\text{ms} \quad < 50\text{ms}
  \end{array}
  
  \begin{array}{c}
  \text{Start Communication Service Request (w/ Keyword)} \\
  \text{Start Communication Service Request}
  \end{array}
  
  \begin{array}{c}
  \text{Wake Up} \\
  \text{Fixed Bit Rate 10,4 kbit/s}
  \end{array}
  
  \]
The K-Line Bus

- Protocol
  - Communication always initiated by master
    - Master sends Request, ECU sends Response
  - Addressing
    - Address length is 1 Byte
    - Either: physical addressing (identifies specific ECU)
      - Or: functional addressing (identifies class of ECU)
        - e.g., engine, transmission, ...
    - Differentiated via format byte
  - Duration of single transmission at 10.4 kBit/s
    - best case: 250 ms, worst case 5.5s
    - i.e., application layer data rate < 1 KB/s
The K-Line Bus

- Protocol header
  - Format Byte
    - Encodes presence and meaning of address bytes
    - Short packet length can be encoded in format byte; length byte then omitted
  - Destination address
  - Source address
  - Length
  - Payload
    - Up to 255 Byte
    - First Byte: Service Identifier (SID)
  - Checksum
    - Sum of all Bytes (mod 256)

<table>
<thead>
<tr>
<th>0 .. 7</th>
<th>8 .. 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format byte</td>
<td>Destination</td>
</tr>
<tr>
<td>Source</td>
<td>Length</td>
</tr>
<tr>
<td>Payload...</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>Checksum</td>
</tr>
</tbody>
</table>
The K-Line Bus

- **Service Identifiers**
  - **Standard Service Identifiers**
    - Session Initialization and teardown
      - 0x81h Start Communication Service Request
      - 0x82h Stop Communication Service Request
    - Configuring protocol timeouts
      - 0x83h Access Timing Parameter Request (optional)
  
- **Other SIDs are vendor defined**
  - Passed on (unmodified) to application layer
  - Typical use: two SIDs per message type
    - First SID: Positive reply
    - Second: Negative reply
The K-Line Bus

- Error handling
  - If erroneous signal arrives
    - ECU ignores message
    - Master detects missing acknowledgement
    - Master repeats message
  
  - If invalid data is being sent
    - Application layer sends negative reply
    - Master / ECU can react accordingly
Use in On Board Diagnostics (OBD)

- Pin 7 of OBD connector is K-Line
- OBD uses stricter protocol variant
- Bit rate fixed to 10.4 kBit/s
- No changes in timing
- Header no longer variable
  - Length byte never included
  - Address always included
- Max. Message length is 7 Byte
- Shall use logical addressing by tester, physical addressing by ECUs
Main Takeaways

- **K-Line**
  - Mainly for diagnostics
  - Transmission uses UART signaling
  - Communication using Request-Response pattern
CAN
Controller Area Network
The CAN Bus

- „Controller Area Network“
- 1986
- Network topology: Bus
- Many (many) physical layers

- Common:
  - Up to 110 nodes
  - At 125 kBit/s: max. 500m

- Always:
- Two signal levels
  - low (dominant)
  - high (recessive)
The CAN Bus

- In the following: ISO 11898
  - Low Speed CAN (up to 125 kBit/s)
  - High Speed CAN (up to 1 MBit/s)
- Specifies OSI layers 1 and 2
  - Higher layers not standardized by CAN, covered by additional standards and conventions
  - e.g., CANopen
- Random access, collision free
  - CSMA/CR with Bus arbitration
  - (sometimes called CSMA/BA – bitwise arbitration)
- Message oriented
- Does not use destination addresses
  - Implicit Broadcast/Multicast
Physical layer (typical)

- High Speed CAN
  - 500 kBit/s
  - Twisted pair wiring
  - Branch lines max. 30 cm
  - Terminating resistor mandated (120 Ω)
  - Signal swing 2 V
  - Error detection must happen within one Bit’s time

\[ \Rightarrow \text{bus length is limited to } l \leq 50m \times \frac{1 \text{ MBit/s}}{\text{data rate}} \]
Physical layer (typical)

- Low Speed CAN
  - Up to 125 kBit/s
  - Standard two wire line suffices
  - No restriction on branch lines
  - Terminating resistors optional
  - Signal swing 5 V

- Single Wire CAN
  - 83 kBit/s
  - One line vs. ground
  - Signal swing 5 V
**CAN in Vehicular Networks**

- **Address-less communication**
  - Messages carry 11 Bit (CAN 2.0A) or 29 Bit (CAN 2.0B) message identifier
  - Stations do not have an address, frames do not contain one
  - Stations use message identifier to decide whether a message is meant for them
  - Medium access using CSMA/CR with bitwise arbitration
  - Link layer uses 4 frame formats
    - Data, Remote (request), Error, Overload (flow control)

- **Data frame format:**

<table>
<thead>
<tr>
<th>Start Bit</th>
<th>Control Bits</th>
<th>Bus Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td>11+1 or 29+3 Bit Message Identifier</td>
<td>6 bit</td>
<td>15 bit CRC</td>
</tr>
<tr>
<td>Header, 19 or 39 bit</td>
<td>Data 0 . . . 8 Byte</td>
<td>Acknowledge &amp; End of Frame</td>
</tr>
<tr>
<td>Payload, 0 . . . 64 bit</td>
<td>Trailer, 25 bit</td>
<td>≥3 bit</td>
</tr>
</tbody>
</table>
CAN in Vehicular Networks

- CSMA/CR with bitwise arbitration
  - Avoids collisions by priority-controlled bus access
  - Each message contains identifier corresponding to its priority
  - Identifier encodes “0” dominant and “1” recessive: concurrent transmission of “0” and “1” results in a “0”
  - Bit stuffing: after 5 identical Bits one inverted Stuff-Bit is inserted (ignored by receiver)
  - When no station is sending the bus reads “1” (recessive state)
  - Synchronization happens on bit level, by detecting start bit of sending station
CAN in Vehicular Networks

- CSMA/CR with bitwise arbitration
  - Wait for end of current transmission
    - wait for 6 consecutive recessive Bits
  - Send identifier (while listening to bus)
  - Watch for mismatch between transmitted/detected signal level
    - Means that a collision with a higher priority message has occurred
    - Back off from bus access, retry later

- Realization of non-preemptive priority scheme
- Real time guarantees for message with highest priority
  - i.e., message with longest “0”-prefix
The CAN Bus

- CSMA/CR with bitwise arbitration
  - Client 2 recognizes bus level mismatch, backs off from access
The CAN Bus

- CSMA/CA with bitwise arbitration (CSMA/CR)
  - Client 1 recognizes bus level mismatch, backs off from access
The CAN Bus

- CSMA/CA with bitwise arbitration (CSMA/CR)
  - Client 3 wins arbitration
The CAN Bus

- CSMA/CA with bitwise arbitration (CSMA/CR)
  - Client 3 starts transmitting data

![Diagram of the CAN Bus with clients and bus transmission patterns]
The CAN Bus: TTCAN

- **Aside: Time-Triggered CAN (TTCAN)**
  - ISO 11898-4 extends CAN by TDMA functionality
  - Solves non-determinism of regular CAN
    - Improves on mere “smart” way of choosing message priorities
  - One node is dedicated “time master” node
  - Periodically sends reference messages starting “basic cycles”
  - Even if time master fails, TTCAN keeps working
    - Up to 7 fallback nodes
    - Nodes compete for transmission of reference messages
    - Chosen by arbitration

![Diagram of basic cycle and reference message windows](image)
The CAN Bus: TTCAN

▪ Aside: TTCAN Basic Cycle
  ▪ Basic cycle consists of time slots
    ▪ Exclusive time slot
    ▪ Reserved for dedicated client
    ▪ Arbitration time slot
      ▪ Regular CAN CSMA/CR with bus arbitration
  ▪ Structure of a basic cycle arbitrary, but static
  ▪ CAN protocol used unmodified
    ➔ Throughput unchanged

▪ TTCAN cannot be seen replacing CAN for real time applications
  ▪ Instead, new protocols are being used altogether (e.g., FlexRay)
The CAN Bus

- **Message filtering**
  - Acceptance of messages determined by message identifier
  - Uses two registers
    - Acceptance Code (bit pattern to filter on)
    - Acceptance Mask (“1” marks relevant bits in acceptance code)

<table>
<thead>
<tr>
<th>Bit</th>
<th>10</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acceptance Code Reg.</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Acceptance Mask Reg.</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Resulting Filter Pattern</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Data format

- NRZ
- Time synchronization using start bit and stuff bits (stuff width 5)
- Frame begins with start bit
- Message identifier 11 Bit (CAN 2.0A), now 29 Bit (CAN 2.0B)
The CAN Bus

- **Data format**
  - **Control Bits**
    - Message type (Request, Data, Error, Overload)
    - Message length
    - ...

<table>
<thead>
<tr>
<th>0</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>7</th>
<th>8</th>
<th></th>
<th></th>
<th></th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>SB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Identifier</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Control Bits</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Acknowledge &amp; End of Frame</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The CAN Bus

- Data format
  - Payload
    - Restriction to max. 8 Byte per message
    - Transmission time at 500 kBit/s: 260 µs (using 29 Bit ID)
    - i.e., usable data rate 30 kBit/s
The CAN Bus

- Error detection (low level)
  - Sender checks for unexpected signal levels on bus
  - All nodes monitor messages on the bus
    - All nodes check protocol conformance of messages
    - All nodes check bit stuffing
  - Receiver checks CRC

- If any(!) node detects error it transmits error signal
  - 6 dominant Bits with no stuffing

- All nodes detect error signal, discard message
The CAN Bus

- Error detection (high level)
  - Sender checks for acknowledgement
    - Receiver transmits dominant “0” during ACK field of received message

- Automatic repeat of failed transmissions

- If controller finds itself causing too many errors
  - Temporarily stop any bus access

- Remaining failure probability ca. $10^{-11}$
The CAN Bus: Transport Layers

- Not covered by ISO 11898 (CAN) standards
  - Fragmentation
  - Flow control
  - Routing to other networks

- Add transport layer protocol
  - ISO-TP
    - ISO 15765-2
  - TP 2.0
    - Industry standard
  - ...

Vehicular Networking
The CAN Bus: ISO-TP

- ISO-TP: Header
  - Optional: 1 additional address Byte
    - Regular addressing
      - Transport protocol address completely in CAN message ID
    - Extended addressing
      - Uniqueness of addresses despite non-unique CAN message ID
      - Part of transport protocol address in CAN message ID, additional address information in first Byte of TP-Header

- 1 to 3 PCI Bytes (Protocol Control Information)
  - First high nibble identifies one of 4 types of message
  - First low nibble and addl. Bytes are message specific

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>(opt) Addl. Address</td>
<td>PCI high</td>
<td>PCI low</td>
<td>(opt) Addl. PCI Bytes</td>
<td>Payload</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Vehicular Networking
The CAN Bus: ISO-TP

- ISO-TP: Message type “Single Frame”
  - 1 Byte PCI, high nibble is 0
  - low nibble gives number of Bytes in payload
  - PCI reduces frame size from 8 Bytes to 7 (or 6) Bytes, throughput falls to 87.5% (or 75%, respectively)
  - No flow control

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Len</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Address)</td>
<td>0</td>
<td>Len</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The CAN Bus: ISO-TP

- **ISO-TP: Message type „First Frame“**
  - 2 Bytes PCI, high nibble is 1
  - low nibble + 1 Byte give number of Bytes in payload
  - After First Frame, sender waits for Flow Control Frame

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Address)</td>
<td>1</td>
<td>Len</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **ISO-TP: Message type „Consecutive Frame“**
  - 1 Byte PCI, high nibble is 2
  - low nibble is sequence number SN (counts upwards from 1)
    - Application layer can detect packet loss
  - No additional error detection at transport layer

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Address)</td>
<td>2</td>
<td>SN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The CAN Bus: ISO-TP

- ISO-TP: Message type „Flow Control Frame“
  - 3 Bytes PCI, high nibble is 3
  - low nibble specifies Flow State FS
  - FS=1: Clear to Send
    - Minimum time between two Consecutive Frames must be ST
    - Sender may continue sending up to BS Consecutive Frames, then wait for new Flow Control Frame
  - FS=2: Wait
    - Overload
    - Sender must wait for next Flow Control Frame
- Byte 2 specifies Block Size BS
- Byte 3 specifies Separation Time ST

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>(Address)</td>
<td>3</td>
<td>FS</td>
<td>BS</td>
</tr>
</tbody>
</table>
The CAN Bus: TP 2.0

- **TP 2.0**
  - Connection oriented
  - Communication based on channels
  - Specifies Setup, Configuration, Transmission, Teardown

- **Addressing**
  - Every ECU has unique logical address; additional logical addresses specify groups of ECUs
  - For broadcast and channel setup:
    - Logical address + offset = CAN message identifier
  - Channels use dynamic CAN message identifier
The CAN Bus: TP 2.0

- TP 2.0: Broadcast
  - Repeated 5 times (motivated by potential packet loss)
  - Fixed length: 7 Byte
  - Byte 0:
    - logical address of destination ECU
  - Byte 1: Opcode
    - 0x23: Broadcast Request
    - 0x24: Broadcast Response
  - Byte 2, 3, 4:
    - Service ID (SID) and parameters
  - Byte 5, 6:
    - Response: 0x0000
    - No response expected: alternates between 0x5555 / 0xAAAA
The CAN Bus: TP 2.0

- TP 2.0: channel setup
  - Byte 0:
    - logical address destination ECU
  - Byte 1: Opcode
    - 0xC0: Channel Request
    - 0xD0: Positive Response
    - 0xD6 .. 0xD8: Negative Response
  - Byte 2, 3: RX ID
    -Validity nibble of Byte 3 is 0 (1 if RX ID not set)
  - Byte 4, 5: TX ID
    -Validity nibble of Byte 5 is 0 (1 if TX ID not set)
  - Byte 6: Application Type
    - cf. TCP-Ports

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dest</td>
<td>Opcode</td>
<td>RX ID</td>
<td>V</td>
<td>TX ID</td>
<td>V</td>
<td>App</td>
</tr>
</tbody>
</table>
The CAN Bus: TP 2.0

- TP 2.0: channel setup (II)
  - Opcode 0xC0: Channel Request
    - TX ID: CAN msg ID requested by self
    - RX ID: marked invalid
  - Opcode 0xD0: Positive Response
    - TX ID: CAN msg ID requested by self
    - RX ID: CAN msg ID of original sender
  - Opcode 0xD6 .. 0xD8: Negative Response
    - Reports errors assigning channel (temporary or permanent)
    - Sender may repeat Channel Request
  - After successful exchange of Channel Request/Response: dynamic CAN msg IDs now assigned to sender and receiver next message sets channel parameters

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Dest</td>
<td>0xC0</td>
<td></td>
<td>1</td>
<td>TX ID</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>App</td>
</tr>
</tbody>
</table>
The CAN Bus: TP 2.0

- **TP 2.0: set channel parameters**
  - Byte 0: Opcode
    - 0xA0: Channel Setup Request (Parameters for channel to initiator)
    - 0xA1: Channel Setup Response (Parameter for reverse channel)
  - Byte 1: Block size
    - Number of CAN messages until sender has to wait for ACK
  - Byte 2, 3, 4, 5: Timing parameters
    - E.g., minimal time between two CAN messages

- **TP 2.0: misc. channel management and teardown**
  - Byte 0: Opcode
    - 0xA3: Test – will be answered by Connection Setup Response
    - 0xA4: Break – Receiver discards data since last ACK
    - 0xA5: Disconnect – Receiver responds with disconnect, too

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0xA0</td>
<td>BS</td>
<td></td>
<td></td>
<td></td>
<td>Timing</td>
</tr>
</tbody>
</table>
The CAN Bus: TP 2.0

- TP 2.0: Data transmission via channels
  - Byte 0, high nibble: Opcode
    - MSB=0 – Payload
      - /AR=0 – Sender now waiting for ACK
      - EOM=1 – Last message of a block
    - MSB=1 – ACK message only (no payload)
      - RS=1 – ready for next message (flow control)
  - Byte 0, low nibble
    - Sequence number
  - Bytes 1 .. 7: Payload

<table>
<thead>
<tr>
<th>Opcode Nibble</th>
<th>0</th>
<th>0</th>
<th>/AR</th>
<th>EOM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>RS</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Op</td>
<td>SN</td>
<td>Payload</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Main Takeaways

▪ CAN
  ▪ Still standard bus in vehicles
  ▪ Message oriented
  ▪ CSMA with bitwise arbitration
    ▪ Impact on determinism
    ▪ TTCAN (TDMA)
  ▪ Error detection
  ▪ Transport layer: ISO-TP vs. TP 2.0
    ▪ Flow control, channel concept
LIN
Local Interconnect Network
The LIN Bus

- Local Interconnect Network (LIN)
- 1999: LIN 1.0
- 2003: LIN 2.0
  - Numerous extensions
  - Backwards compatible (only)
- Goal of LIN: be much cheaper than low speed CAN
  - Only reached partway
- specifies PHY and MAC Layer, API
The LIN Bus

- Very similar to K-Line Bus
- Master-slave concept with self synchronization
  - no quartz needed
  - lax timing constraints
- LIN master commonly also part of a CAN bus
  - LIN commonly called a sub bus
- Bidirectional one-wire line, up to 20 kBit/s
- Bit transmission UART compatible
  - 1 Start Bit, 8 Data Bits, 1 Stop Bit
- Message oriented
  - No destination address
The LIN Bus

- Rudimentary error detection
  - Sender monitors bus
  - Aborts transmission on unexpected bus state
- No error correction
- Starting with LIN 2.0: Response Error Bit
  - Should be contained in periodic messages
  - Set (once) if slave detected an error in last cycle
- Static slot schedule in the master
  - “Schedule Table”
  - Determines cyclic schedule of messages transmitted by master
    → Bus timing mostly deterministic
  - Slaves do not need to know schedule
    → can be changed at run-time
The LIN Bus

- Data request (sent by master)
  - Sync Break (≥13 Low Bits, 1 High Bit)
    - Not UART compliant $\rightarrow$ uniquely identifiable
  - Sync Byte 0x55 (01010101)
    - Synchronizes bit timing of slave
  - LIN Identifier (6 data Bits (I0 to I5) + 2 parity Bits)
    - Encodes response’s expected message type and length
    - 0x00 .. 0x3B: application defined data types, 0x3C .. 0x3D: Diagnosis,
      0x3E: application defined, 0x3F: reserved
    - Parity Bits: $I0 \oplus I1 \oplus I2 \oplus I4$ and $\neg (I1 \oplus I3 \oplus I4 \oplus I5)$
- Data request triggers data response ($\Leftrightarrow$ next slide)
The LIN Bus

- **Data response** (sent by slave)
  - Slave responds with up to 8 Bytes of data
    - LSB first, Little Endian
    - length was defined by LIN Identifier
  - Frame ends with checksum
    - LIN 1.3: Classic Checksum (only data bytes)
    - LIN 2.0: Enhanced Checksum (data bytes + Identifier)
    - Checksum is sum of all Bytes (mod 256), plus sum of all carries
The LIN Bus

- Types of requests
  - Unconditional Frame
  - Event Triggered Frame
  - Sporadic Frame
  - ...

- Unconditional Frame
  - Most simple frame type
  - Designed for periodic polling of specific data point
  - Exactly one slave answers
  - LIN is a single master system $\Rightarrow$ timing of unconditional frames fully deterministic
  - Sample use case:
    - Request “did state of front left door contact change?” every 15 ms
    - Receive negative reply by front left door ECU every 15 ms
The LIN Bus

- **Types of requests**
  - Unconditional Frame
  - Event Triggered Frame
  - Sporadic Frame
  - ...

- **Event Triggered Frame**
  - Simultaneous polling of multiple slaves, slave answers if needed
  - Collisions possible (→ non-determinism), detect by corrupt. data
    - master switches to individual polling via Unconditional Frames
  - Use whenever slaves unlikely to respond
  - Sample use case:
    - Request “did state of a door contact change?” every 15 ms
    - Change in state unlikely, simultaneous change extremely unlikely
The LIN Bus

- Types of requests
  - Unconditional Frame
  - Event Triggered Frame
  - Sporadic Frame
  - ...

- Sporadic Frame
  - Sent (by master) only when needed
  - Shared schedule slot with other Sporadic Frames
  - Use whenever polling for specific data only seldom needed
  - If more than one Sporadic Frame needs to be sent, master needs to decide for one → no collision, but still non-deterministic
  - Sample use case:
    - Request „power window fully closed?“ every 15 ms
    - ...only while power window is closing
The LIN Bus

- Sample schedule table

<table>
<thead>
<tr>
<th>Slot</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unconditional</td>
<td>AC</td>
</tr>
<tr>
<td>2</td>
<td>Unconditional</td>
<td>Rain sensor</td>
</tr>
<tr>
<td>3</td>
<td>Unconditional</td>
<td>Tire pressure</td>
</tr>
<tr>
<td>4</td>
<td>Event triggered</td>
<td>Power window</td>
</tr>
<tr>
<td>5</td>
<td>Sporadic</td>
<td>(unused)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- OR - Fuel level</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- OR - Outside temp</td>
</tr>
</tbody>
</table>
The LIN Bus

- Doing Off-Board-Diagnosis of LIN ECUs
  - Variant 1: Master at CAN bus responds on behalf of ECU on LIN
    - Keeps synchronized state via LIN messages
  - Variant 2: Master at CAN bus tunnels, e.g., KWP 2000 messages
    - Standardized protocol
    - LIN dest address is 0x3C (Byte 1 is ISO dest address)
    - Dest ECU (according to ISO address) answers with address 0x3D
    - Independent of payload, LIN frame padded to 8 Bytes
    - LIN slaves have to also support KWP 2000
    - Contradicts low cost approach of LIN
    - “Diagnostic Class” indicates level of support

Vehicular Networking
Main Takeaways

- LIN
  - Goals
  - Deployment as sub bus
  - Message types and scheduling
  - Determinism
Main Takeaways

- Overall
  - Design goals
  - Message orientation vs. address orientation,
  - Addressing schemes
  - Medium access
  - Flow control
  - Real time guarantees and determinism
FlexRay
FlexRay

- Motivation
  - Drive/Brake/Steer-by-Wire
  - CAN bus is prone to failures
    - Line topology
    - No redundant links
  - CAN bus is slow
  - Need for short bus lines $\implies$ deployment expensive, complicated
  - Non-determinism for all but one message class
    - Worst case delay unacceptably high
  - Early solutions by OEMs proprietary
    - TTCAN, TTP/TTA, Byteflight, ...
  - Foundation of consortium to develop new bus: FlexRay
    - BMW, VW, Daimler, GM, Bosch, NXP, Freescale
  - First series deployment at end of 2006 (BMW X5)
FlexRay

- Bus topology
  - Line, Star with bus termination
  - Max. distance per line: 24m
  - Optional use of second channel
    - Higher redundancy or(!) higher speed
    - Up to 10 MBit/s for single channel, 20 MBit/s for dual channel
FlexRay

- Bit transmission
  - Need synchronized clocks in sender and receiver
  - Thus, need additional bits for synchronizing signal sampling at receiver (done with each 1⇔0 flank)
  - Don’t use bit stuffing otherwise: message length becomes non-deterministic (cf. CAN)
  - New concept: frame each transmission, each frame, each Byte
    - Bus idle (1)
    - Transmission Start Signal (0)
      - Frame Start Signal (1)
        - Byte Start Signal (1)
        - Byte Start Signal (0)
        - 8 Bit Payload (...)
      - Frame End Signal (0)
    - Transmission End Signal (1)
FlexRay

- **Bus access**
  - Bus cycle (ca. 1 μs .. 7 μs)
    - Static Segment
    - Dynamic Segment (opt.)
    - Symbol Window (opt.)
    - Network Idle Time
  - Global *Cycle Counter* keeps track of bus cycles passed

- **Static Segment**
  - Slots of fixed length (2 .. 1023)
  - One Message per Slot
  - Static assignment (of slot and channel) to ECUs (i.e., TDMA)

⇒ bus access is collision free, deterministic
FlexRay

- **Dynamic Segment**
  - Split into minislots (also statically assigned to ECUs)
  - Messages (usually) take up more than one minislot
  - Slot counter pauses while message is being transmitted (thus, slot counters of channels A and B soon desynchronize)
  - Lower priority messages have higher slot number (thus sent later, or not at all)

- **Example:**

<table>
<thead>
<tr>
<th>(mini)slots</th>
<th>Static Segment</th>
<th>Dynamic Segment</th>
<th>Sym</th>
<th>Net Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel A</td>
<td>1 2 3</td>
<td>4 5 6 7 8 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel B</td>
<td>1 2 3</td>
<td>4 5 6</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>
# FlexRay

- **Message format**
  - **Control Bits**
    - Bit 0: Reserved
      - Unused, always 0
    - Bit 1: Payload Preamble Indicator
      - In static segment: first 0 .. 12 Byte payload for management information
      - In dynamic segment: first 2 Byte payload contains Message ID (cf. UDP Port)

<table>
<thead>
<tr>
<th>5 Bit</th>
<th>11 Bit</th>
<th>7 Bit</th>
<th>11 Bit</th>
<th>6 Bit</th>
<th>24 Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Bits</td>
<td>Frame ID</td>
<td>Length</td>
<td>Header CRC</td>
<td>Cycle Counter</td>
<td>Payload</td>
</tr>
</tbody>
</table>
FlexRay

- **Message format**
  - **Control Bits**
    - Bit 2: Null Frame Indicator
      - Indicates frame without payload
      - Allows sending “no message” also in static segment (fixed slot lengths!)
    - Bit 3: Sync Frame Indicator
      - Indicates frame may be used for synchronizing clock
      - To be sent by 2 .. 15 “reliable” ECUs
    - Bit 4: Startup Frame Indicator
      - Used for synchronization during bootstrap
      - Sent by cold start node (⇨ later slides)

<table>
<thead>
<tr>
<th>5 Bit</th>
<th>11 Bit</th>
<th>7 Bit</th>
<th>11 Bit</th>
<th>6 Bit</th>
<th>24 Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Bits</td>
<td>Frame ID</td>
<td>Length</td>
<td>Header CRC</td>
<td>Cycle Counter</td>
<td>Payload</td>
</tr>
</tbody>
</table>
## FlexRay

- **Message format**
  - **Frame ID**
    - Identifies message ($\triangleq$ slot number)
  - **Length**
    - Length of payload (in 16 Bit words)
  - **Header CRC**
  - **Cycle Counter**
    - Global counter of passed bus cycles
  - **Payload**
    - $0..127$ 16 Bit words ($\triangleq 0..254$ Byte of payload)
  - **CRC**

<table>
<thead>
<tr>
<th>5 Bit</th>
<th>11 Bit</th>
<th>7 Bit</th>
<th>11 Bit</th>
<th>6 Bit</th>
<th>24 Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Bits</td>
<td>Frame ID</td>
<td>Length</td>
<td>Header CRC</td>
<td>Cycle Counter</td>
<td>Payload</td>
</tr>
</tbody>
</table>
FlexRay

- Time synchronization
  - Need synchronized bit clock + synchronized slot counter
  - Want no dedicated time master ⇒ Distributed synchronization
  - Configure (typically) three nodes as “cold start nodes”

- Cold start procedure (followed by all cold start nodes):
  - Check if bus idle
    - if bus not idle ⇒ abort (cold start already proceeding or unneeded)
  - Transmit wakeup (WUP) pattern
    - if collision occurs ⇒ abort
    - if no collisions occurred ⇒ this is the leading cold start node

- Cold start procedure (leading cold start node):
  - Send Collision Avoidance Symbol (CAS)
  - Start regular operations (cycle counter starts at 0)
    - Set Bits: Startup Frame Indicator ⊕ Sync Frame Indicator
FlexRay

- Time synchronization
  - Cold start procedure (other cold start nodes)
    - Wait for 4 Frames of leading cold start node
    - Start regular operations
      - Set Bits: Startup Frame Indicator \( \oplus \) Sync Frame Indicator
  - Cold start procedure (regular ECUs)
    - Wait for 2 Frames of 2 cold start nodes
    - Start regular operations

<table>
<thead>
<tr>
<th></th>
<th>WUP</th>
<th>WUP</th>
<th>CAS</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>WUP</td>
<td></td>
<td></td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>( \checkmark )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>( \checkmark )</td>
<td></td>
<td></td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>7</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>
FlexRay

- Example configuration of timing
  - Use fixed payload length of 16 Byte (with header and trailer: 24 Bytes; with FSS, BSS, FES: ca. 250 Bits)
  - 10 Mbps data rate $\Rightarrow$ 25 µs message duration
  - Add 5 µs guard to care for propagation delay and clock drift $\Rightarrow$ 35 µs slot length in static segment
  - One macro tick: 1 µs (can use 1 .. 6 µs)
  - One minislot: 5 macro ticks: 5 µs
  - Tbit = 100 ns, sample rate of bus = Tbit/8 = 12.5 ns
FlexRay

- Example configuration of timing (contd.)
  - Use 64 distinct communication cycles
  - Communication cycle duration: 5 ms
  - Use 3 ms for static segment
  - Remaining 2 ms used for dynamic segment, symbol window, network idle time

- Message repetition interval fully customizable, e.g.:
  - 2.5 ms (one slot each at start and end of static segment)
  - 5 ms (one slot each in every communication cycle)
  - 10 ms (one slot in every second communication cycle)
  - ...
FlexRay

- Error prevention
  - Integrate bus guard
  - Implement separately from communication controller
  - Follows protocol steps in communication controller
  - Can only enable bus driver when allowed to communicate, or permanently disable in case of errors (*babbling idiot problem*)
FlexRay

- **Error handling**
  - Multiple measures for error detection
    - Check cycle counter value
    - Check slot counter value
    - Check slot timing
    - Check header CRC
    - Check CRC
  - Reaction to timing errors
    - Do not automatically repeat messages (⇔ non-determinism)
    - Switch to passive state instead
      - Stop transmitting messages
      - Keep receiving messages (might allow re-synchronization to bus)
  - Reaction to severe, non-recoverable errors
    - Completely switch off bus driver
FlexRay

- AUTOSAR TP
  - Transport protocol of FlexRay
  - Upwards compatible to ISO 15765-2 (ISO TP for CAN)
  - Adjusted and extended for FlexRay
  - Difference in addressing
    - In CAN: CAN message ID assigned arbitrarily
    - In FlexRay: Frame ID \(\triangleq\) Slot Number (i.e., not arbitrary)
    \(\Rightarrow\) cannot use source/destination addresses as IDs in lower layer
    - Address encoded only (and completely) in TP header

- Also:
  - New message types

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1 .. 2 Byte</td>
<td>1 .. 2 Byte</td>
<td>1 .. 5 Byte</td>
<td></td>
</tr>
<tr>
<td>Target Address</td>
<td>Source Address</td>
<td>PCI</td>
<td>Payload</td>
</tr>
</tbody>
</table>

Vehicular Networking
### FlexRay

- **AUTOSAR TP**
  - Frame types: Single Frame *Extended* / First Frame *Extended*
  - Larger *data length* (DL) field allows for longer payload
    - Four kinds of first frames can indicate payloads of up to 4 GiB

<table>
<thead>
<tr>
<th></th>
<th>PCI Byte 0</th>
<th>PCI Byte 1</th>
<th>PCI Byte 2</th>
<th>PCI Byte 3</th>
<th>PCI Byte 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Frame</td>
<td>0</td>
<td>DL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single Frame Extended*</td>
<td>5</td>
<td>0</td>
<td>DL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Frame</td>
<td>1</td>
<td>DL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Frame Extended*</td>
<td>4</td>
<td>1</td>
<td>DL</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>DL</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3</td>
<td></td>
<td>DL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4</td>
<td></td>
<td></td>
<td>DL</td>
</tr>
</tbody>
</table>
FlexRay

- AUTOSAR TP
  - Extended flow control
    - FS values allow triggering abort of ongoing transmission
      - FS=2: Overflow
      - FS=5: Cancel, Data Outdated
      - FS=6: Cancel, No Buffer
      - FS=7: Cancel, Other
    - ST split into two ranges to allow shorter separation times
      - 0x00 .. 0x7F Separation Time in ms
      - 0xF1 .. 0xF9 Separation Time in μs (new!)

<table>
<thead>
<tr>
<th></th>
<th>PCI Byte 0</th>
<th>PCI Byte 1</th>
<th>PCI Byte 2</th>
<th>PCI Byte 3</th>
<th>PCI Byte 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consecutive Frame</td>
<td>2</td>
<td>SN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Consecutive Frame 2*</td>
<td>6</td>
<td>SN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flow Control Frame</td>
<td>3</td>
<td>FS</td>
<td>BS</td>
<td>ST</td>
<td></td>
</tr>
<tr>
<td>Acknowledge Frame*</td>
<td>7</td>
<td>FS</td>
<td>BS</td>
<td>ST</td>
<td>ACK</td>
</tr>
</tbody>
</table>
• AUTOSAR TP
  • Extended flow control
    • CAN: Acknowledgement by transmitting dominant bit in ACK field
    • FlexRay: New Acknowledge Frame (AF)
    • Use after single frame or after all consecutive frames (as ACK) or immediately (as NACK)
    • Functions identical to Flow Control Frame, but adds ACK and SN nibbles
      • ACK is 1 or 0; SN indicates slot number of first defective frame
      • Sender may repeat failed transmissions at earliest convenience (alternately uses CF and CF2 frames)

<table>
<thead>
<tr>
<th></th>
<th>PCI Byte 0</th>
<th>PCI Byte 1</th>
<th>PCI Byte 2</th>
<th>PCI Byte 3</th>
<th>PCI Byte 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consecutive Frame</td>
<td>2</td>
<td>SN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Consecutive Frame 2*</td>
<td>6</td>
<td>SN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flow Control Frame</td>
<td>3</td>
<td>FS</td>
<td>BS</td>
<td>ST</td>
<td></td>
</tr>
<tr>
<td>Acknowledge Frame*</td>
<td>7</td>
<td>FS</td>
<td>BS</td>
<td>ST</td>
<td>ACK SN</td>
</tr>
</tbody>
</table>
MOST

Media Oriented Systems Transport
MOST

- Media Oriented Systems Transport
  - specifies ISO layers 1 through 7
  - Does not focus on sensor/actor tasks (e.g., delay, fault tolerance), but on infotainment (e.g., jitter, data rate)
- History
  - Domestic Data Bus (D2B, later: Domestic Digital Bus) developed by Philips, later standardized as IEC 61030 (still in the 90s)
  - Little adoption in vehicles, thus SMSC soon develops a successor
  - 1998: MOST Cooperation standardizes MOST bus (Harman/Becker, BMW, DaimlerChrysler, SMSC)
  - December 2009: MOST 3.0E1 published
  - Today: MOST cooperation numbers 60 OEMs, 15 vehicle manufacturers
MOST

- **Medium**
  - Plastic Optic Fiber (POF) alternative (copper) variant specified, but little used
  - Data rates specified from 25 (MOST25) to 150 MBit/s (MOST150)
  - Manchester coded bit transmission
  - Dedicated timing master ECU (slaves adopt bit timing)
  - Logical bus topology: ring of up to 64 ECUs
  - Physical bus topology can differ
MOST

- **Link Layer**
  - Synchronous bit stream; all clocks synchronized to timing master
  - Stream divided into blocks; each block traverses ring exactly once
  - Blocks divided into 16 Frames
    - Frame size: 64 Byte (MOST25) to 384 Byte (MOST150)
    - Frame rate static but configurable; recommended: 48 kHz (DVD)
  - Frame divided into
    - Header (with boundary descriptor) and Trailer
    - Data: Synchronous Channel, Asynchronous Channel, Control Channel
MOST

- **Link Layer**
  - **Synchronous Channel**
    - Use case: audio or video
    - TDMA divides frame into streaming channels
      ⇒ deterministic
    - Reserved by messages on control channel
    - Thus, no addressing required
    - Maximum number of streaming channels limited by frame size

<table>
<thead>
<tr>
<th>Streaming Channel 1</th>
<th>Streaming Channel 2</th>
<th>Streaming Channel 3</th>
<th>unused</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD-Audio, Device A</td>
<td>DVD-Video, Device B</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
MOST

- Link Layer
  - Asynchronous Channel
    - Use case: TCP/IP
    - Random access with arbitration (based on message priority)
      ⇒ non-deterministic
    - Single message may take more than one frame
    - Short additional header contains source/destination address, length
    - Short additional trailer contains CRC
    - No acknowledgement, no automatic repeat on errors

<table>
<thead>
<tr>
<th>1 Byte</th>
<th>2 Byte</th>
<th>1 Byte</th>
<th>2 Byte</th>
<th>4 Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arbitration</td>
<td>Target Address</td>
<td>Len</td>
<td>Source Address</td>
<td>...</td>
</tr>
</tbody>
</table>
MOST

- Link Layer
  - Control Channel
    - Management and control data
    - Random access with arbitration (based on message priority)
    - Message length 32 Byte
      - MOST25 control channel uses 2 Bytes per frame
        ⇒ each message takes 16 Frames = 1 Block
    - Message reception is acknowledged by recipient
    - Failed transmissions are automatically repeated

<table>
<thead>
<tr>
<th>1 Byte</th>
<th>2 Byte</th>
<th>2 Byte</th>
<th>1 Byte</th>
<th>17 Byte</th>
<th>2 Byte</th>
<th>1 Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arbitration</td>
<td>Target Address</td>
<td>Source Address</td>
<td>Type</td>
<td>Data</td>
<td>CRC</td>
<td>Trailer</td>
</tr>
</tbody>
</table>

Vehicular Networking
MOST

- Link Layer
  - Control Channel messages
    - Resource Allocation, Resource De-allocation:
      - manage streaming channels in synchronous segment
  - Remote Read, Remote Write
    - accesses registers and configuration of ECUs
  - Remote Get Source
    - query owner of streaming channels in synchronous segment
  - ...
    - Other message types are transparently passed to upper layers
MOST

- Link Layer
  - Addressing
    - 16 Bit addresses
    - physical address
      - According to relative position in ring
      - Master gets 0x400
      - First slave gets 0x401
      - etc.
    - logical address
      - Assigned by master
      - Typically upwards of 0x100 (Master)
  - groupcast
    - Typically 0x300 + ID of function block
  - broadcast
    - Typically 0x3C8
MOST

- Ring disruption
  - Causes
    - ECU stops working
    - Plastic optic fiber gets damaged
  - Symptoms
    - Messages either not transmitted to recipient, or not back to sender thus: total failure of bus
  - Diagnosis
    - Ring disruption easily detected
    - Reason and affected ECUs impossible to determine
  - Workarounds
    - Vendor dependent, proprietary
    - often: use additional single-wire bus for further diagnosis
MOST

- Higher layers: Object oriented MOST Network Services
  - Function block (= class)
    - e.g. audio signal processing (0x21), audio amplifier (0x22), ...
    - Multiple classes per device, multiple devices per class
    - Every device implements function block 0x01 (MOST Netw. Services)
  - Instance
    - Uniquely identifies single device implementing certain function block
  - Property/Method
    - Property (get/set value)
    - Method (execute action)
  - Operation
    - Set/Get/... (Property), Start/Abort/... (Method)
  - 22.00.400.0 (20) ⇒ amplifier number 0: volume set to 20
MOST

- Higher layers: System boot and restart
  - Master node announces reset of global state (all devices change status to Not-OK and cease operations)
  - Master node initiates system scan
    - Iteratively polls all physical addresses for present function blocks
    - Devices answer with logical address, list of function blocks, and instance numbers
  - Master can detect ambiguous combinations of function blocks and instance numbers ⇒ will then assign new instance numbers
  - Master keeps table of all device’s operation characteristics
  - Master reports to all devices: status OK
  - MOST Bus is now operational
MOST

- Higher layers – MAMAC and DTCP
  - Trend towards all-IP in consumer electronics addressed in MOST by introducing MAMAC (MOST Asynchronous Media Access Control)
    - Encapsulates Ethernet and TCP/IP for transmission on MOST bus
    - but: not supported by MOST services; needs to be implemented in software
  - Concerns of music/film industry wrt. digital transmission addressed in MOST by introducing DTCP (Digital Transmission Content Protection)
    - As known from IEEE 1394 (FireWire)
    - Bidirectional authentication and key exchange of sender/receiver
    - Encrypted data transmission
In-Car Ethernet
In-Car Ethernet

- IEEE 802.3
- Bob Metcalfe, David R. Boggs
- 1973, Parc CSMA/CD Ethernet
  - 3 Mbit/s, 256 nodes, 1 km coax cable
- 1980- revised to become IEEE Std 802.3
- Next big thing?
  - “Automotive. Cars will have three networks. 
    (1) Within the car. 
    (2) From the car up to the Internet. And 
    (3) among cars. 
    IEEE 802 is ramping up for these standards now, I hope.”
--/u/BobMetcalfe on http://redd.it/1x3fiq

Vehicular Networking
In-Car Ethernet

- Why?
  - Old concept:
    - Strictly separated domains
    - Each served by specialized bus
    - Minimal data interchange
  - Current trend:
    - Advanced Driver Assistance Systems (ADAS)
    - Sensor data fusion
      - (in-car, between cars)
    - Ex: Cooperative Adaptive Cruise Control (CACC)
  - Move from domain specific buses ⇒ general-purpose bus
Ethernet

- Physical layers
  - 10BASE5 (aka Thicknet, aka IEEE Std 802.3-1985)
    - Manchester coded signal, typ. 2 V rise
    - 10 Mbit/s over 500m coax cable
    - Nodes tap into core (“vampire tap”)
  - 10BASE2
    - 10 Mbit/s over “almost” 200m coax cable
    - BNC connectors, T-shaped connectors

- Medium access: CSMA/CD
  - Carrier sensed $\Rightarrow$ medium busy
  - Collision $\Rightarrow$ jam signal, binary exponential backoff (up to 16 times)
Ethernet

- Physical layers
  - 1000BASE-T
    - 1 Gbit/s over 100m
    - Cat 5e cable with 8P8C connectors, 4 twisted pairs of wires, multi-level signal (-2, -1, 0, +1, +2), scrambling, ...
    - Medium access
    - No longer shared bus, but point to point
    - Auto-negotiated (timing) master/slave

- 100GBASE-ER4
  - 100 Gbit/s over 40 km
  - Plastic Optic Fiber (POF)

- ...

Vehicular Networking 148
Ethernet

- Link layer
  - Lightweight frame type
  - Optional extensions, e.g., IEEE 802.1Q (identifier 0x8100)
  - Directly encapsulates higher layer protocols, e.g., IPv6 (0x86DD)
  - ...or IEEE 802.2 Logical Link Control (LLC) frame (identifier is len) (in Byte)
  - Error-checked, but only best effort delivery of data

<table>
<thead>
<tr>
<th>0</th>
<th>7</th>
<th>8</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preamble (1010..11)</td>
<td>Destination MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source MAC</td>
<td>(opt) 802.1Q tag</td>
<td>Type/len</td>
<td></td>
</tr>
<tr>
<td>Payload (commonly 42-1500 Byte, max 1982 Byte)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Checksum</td>
<td>(Idle time)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In-Car Ethernet

- In-car Ethernet?
  - Almost all “in-car” qualities absent
    - Heavy, bulky cabling
    - Huge connectors
    - Sensitive to interference
    - Needs external power
    - No delay/jitter/... guarantees
    - No synchronization
    - Etc...

- But:
  - ...can be easily extended:
    - New physical layers
    - Tailored higher-layer protocols

Vehicular Networking
In-Car Ethernet

- One-Pair Ether-Net (OPEN) alliance SIG
  - Founded: BMW, Broadcom, Freescale, Harman, Hyundai, NXP
  - 2014: approx. 150 members
  - 100 Mbit/s on single twisted pair, unshielded cable
  - Power over Ethernet (IEEE 802.3at)
  - Manufactured by Broadcom, marketed as BroadR-Reach

- Reduced Twisted Pair Gigabit Ethernet (RTPGE) task force
  - Working on IEEE 802.3bp
  - 1 Gbit/s over up to 15m single twisted pair cable
In-Car Ethernet

- Upper layers: TSN
  - Many solutions (e.g., SAE AS6802 “Time Triggered Ethernet”)
  - Current: IEEE 802.1 Time Sensitive Networking (TSN) task group (aka Audio/Video Bridging AVB task group, up until 2012)
  - Promoted by AVnu Alliance SIG (cf. IEEE 802.11 / Wi-Fi Alliance)

- Concept
  - Needs TSN-enabled switches / end devices
  - Tight global time synchronization
  - Dynamic resource reservation on streams through network
  - IEEE 802.1AS... extensions
    - Layer 2 service
  - IEEE 802.1Q... extensions
    - Frame tagging standard
In-Car Ethernet

- **IEEE 802.1AS Time Synchronizing Service**
  - Subset of IEEE 1588 Precision Time Protocol (PTP)
  - Syncs clock value/frequency of all nodes
  - Election of “master” time master (grandmaster clock), disseminates sync information along spanning tree

- **IEEE 802.1Qat Stream Reservation Protocol (SRP)**
  - *Talker* advertises stream (along with parameters)
  - Advertisement is disseminated through network
  - Intermediate nodes check, block available resources, update advertisement with, e.g., newly computed worst case latency
  - *Listeners* check (annotated) advertisement, send registration message back to *Talker*
  - Intermediate nodes reserve resources, update multicast tree
In-Car Ethernet

- IEEE 802.1Qav etc. Traffic Shaping
  - Prioritize frames according to tags
  - Avoid starvation, bursts, ...
  - e.g., Token bucket, with many more proposed

- IEEE 802.1Qbu Frame Preemption
  - Can cancel ongoing transmissions (if higher priority frame arrives)

- IEEE 802.1Qcb Media Redundancy

- ...

Vehicular Networking
Main Takeaways

- FlexRay
  - Motivation
  - Single or dual channel operation
  - Distributed operation
  - Static and dynamic segment

- MOST
  - Motivation
  - Topology and implications
  - Centralized operation
  - Synchronous and asynchronous channel

- Ethernet
  - Concept
  - Drawbacks of classic standards
  - New PHY layers
  - New upper layers (TSN)
ECUs

Electronic Control Units
Electronic Control Units (ECUs)

- Middle and upper class vehicles carry 80 .. 100 networked ECUs
- Each consisting of
  - Transceiver (for bus access)
  - Power supply
  - Sensor drivers
  - Actor drivers
  - ...and an ECU Core (⇨ next slide)
- Depending on deployment scenario, ECU and components must be
  - Shock resistant
  - Rust proof
  - Water resistant, oil resistant
  - Heat resistant
  - ...

▪ Depending on deployment scenario, ECU and components must be
ECU Core

- △ Personal Computer
- additional external guard hardware (e.g., watchdog) for safety critical applications

Microcontroller (MCU) (⇨ next slide)

- opt. Co-Processors, DSPs, ...

I/O drivers

ext. memory

ext. memory

ASIC

watchdog

...
Architecture

External Bus

Program Memory

CPU

Data Memory

DMA

Sys. Timer

Bus Ctrl.

Bus Ctrl.

Interface to other controllers

Ports

Interrupt Handler

Timers

System Ctrl.

Interfaces (CAN, serial, JTAG, ...)

Serial Bus

A/D Converter(s)
Architecture

- Microcontroller (MCU)
  - 8, 16, 32 Bit
  - Infineon, Freescale, Fujitsu, ...

- Memory
  - Volatile memory
    - SRAM (some kByte)
    - Typically integrated into microcontroller
  - Non-volatile memory
    - Flash (256 kByte .. some MByte)
    - Serial EEPROM (some kByte, e.g., for error log)

- Power supply
  - DC/DC converter, e.g., to 5 V or 3.3 V
Architecture

- Clock
  - Quartz Xtal, some 10 MHz (↔ ECU requires only passive cooling)
- External guard hardware
  - Watchdog
    - Expects periodic signal from MCU
    - Resets MCU on timeout
  - ASIC guard
    - For more complex / critical ECUs
    - ASIC sends question, MCU must send correct answer before timeout
    - Resets (or disables) ECU on timeout or error
- Internal Buses
  - Low-cost ECUs can use shared bus for address and data
  - Parallel
Architecture

- **Sensor drivers**
  - Resistive sensors (e.g., simple potentiometer for length, angle)
  - Capacitive, inductive sensors (e.g., pressure, distance)
  - Active sensors (simple voltage / complex data output)

- **Actor drivers**
  - D/A conversion
  - High-power amplifiers
  - Bridges

- **Further requirements**
  - Electro-magnetic interference (EMI) characteristics
  - Mechanical robustness
  - Water resistance
  - Thermal resistance
  - Chemical resistance
Automotive Operating Systems

- Hardware abstraction
  - Often missing, hardware accessed directly
  - Recent trends towards operating systems

- Application Programming Interface (API)
  - Common for message transmission over external buses

- Software safeguards
  - E.g., stack overflow
  - Particularly helpful during development
Automotive Operating Systems

- Process States

- States:
  - Running
  - Suspended
  - Ready
  - Waiting

- Transitions:
  - Wait
  - Preempt
  - Release
  - Start
  - Activate
  - Terminate
Scheduling

- Set of suspended tasks
- Set of ready tasks
- Priority queue of ready tasks
- Task executed

1. Activation
   - time or event based
   - ready
   - Scheduler
   - Priority queue
   - Order
   - Dispatcher
   - running
Automotive Operating Systems

- **Scheduling**
  - The act of assigning an order of activation, given a process model, activation sequence, and deadlines
    - *dynamic*: Schedule is calculated at run time
    - *static*: Schedule is fixed, e.g., at compile time (⇒ fully deterministic)
  - *Feasible schedule*: all time constraints fulfilled, no deadline violated
  - Dispatcher coordinates context switches

- **Context switches**
  - For one process to change state to *running*, another process may need to be preempted
  - CPU registers etc. will now be occupied by new process, operating system takes care of persisting information
Real Time Properties

- **Latency**
  - Time difference from event to reaction

- **Jitter**
  - Difference of max and min latency
  - High importance in feedback control systems

- **Execution time**
  - Time difference of task start and end
  - **Worst Case Execution Time (WCET)**
    - Defined for program aspects, dependent on platform
    - Considers every possible cause of delay (interrupts, caching, ...)
    - Important for guaranteeing determinism
Real Time Properties

- **Soft deadline**
  - Delivering result after soft deadline less helpful (reduced benefit)
  - e.g., car speeds up $\Rightarrow$ radio gets louder

- **Firm deadline**
  - Delivering result after firm deadline useless (no benefit)
  - e.g., incoming traffic bulletin $\Rightarrow$ SatNav powered up

- **Hard deadline**
  - Delivering result after hard deadline causes damage or harm (negative benefit)
  - e.g., brake pedal is pushed $\Rightarrow$ car decelerates
Real Time Properties

- Real time systems
  - Internal image of system state in memory
  - State described by set of variables
  - Needs continuous update of image

- Real time architecture
  - Event triggered system
    - Image update with every change of state
  - Time triggered system
    - Image update in fixed intervals
    - Internal or global clock (needs synchronization)
OSEK/VDX

- 1993
  - Founded as OSEK – “Offene Systeme und deren Schnittstellen für die Elektronik im Kraftfahrzeug”
  - BMW, Bosch, Daimler Chrysler, Opel, Siemens, VW, Univ. Karlsruhe
- 1994
  - Merged with VDX – “Vehicle Distributed Executive”
  - PSA und Renault

- Today
  - More than 50 partners
  - (Parts) standardized as ISO 17356 series
  - Standardizes common communications stack, network management, operating system (⇒ next slides), ...
  - Many free implementations (freeOSEK, openOSEK, nxtOSEK, ...)
OSEK/VDX

- Properties
  - Operating system for single processor
  - Static configuration
    - Tasks
    - Resources
    - Functions
  - Can meet requirements of hard deadlines
  - Programs execute directly from ROM
  - Very low memory requirements
  - Standardized system (⇔ “OSEK conformant ECUs”)
OSEK/VDX

- Configuration
  - Operating system configured at compile time

- OSEK Implementation Language (OIL)
  - Scheduling strategy
  - Task priorities
  - ...

```c
CPU OSEK_Demo
{

OSEK_Example_OS
{
    MICROCONTROLLER = Intel80x86;
    ...
}
);

TASK Sample_TASK
{
    PRIORITY = 12;
    SCHEDULE = FULL;
    AUTOSTART = TRUE;
    ACTIVATION = 1;
}
);

...
Building of OSEK/VDX firmware

Configurator

*.oil

Generator

os.c  os.h

Compiler

*.c  *.h

*.obj

Linker

os.elf
OSEK/VDX

- Tasks
  - Static priority
  - Relationships of tasks
    - Synchronization
    - Message exchange
    - Signaling
  - Support for time triggered services
  - Error management
  - C macros for definition provided

```c
DeclareTask(SampleTask);
...
TASK(SampleTask) {
    /* read sensors, trigger actors */
    TerminateTask();
}
```
OSEK/VDX

- **Scheduling**
  - Scheduler always chooses highest priority task
  - Configurable modes:
    - Non preemptive: Tasks are never preempted
    - Preemptive: Higher priority tasks always preempt lower priority tasks
    - Mixed: Individual configuration of each task

![Diagram of task scheduling](image)
AUTOSAR

- Traditional paradigm: one function $\rightarrow$ one ECU (incl. software and OS, supplied by OEM)

- AUTOSAR (*Automotive Open System Architecture*) Initiative of automobile manufacturers to make software development independent of operating system

- Mix and match of hardware and software
  - Integration at manufacturer
  - In-house development of software at manufacturer
  - Independence of/from OEM
AUTOSAR

- AUTOSAR Runtime Environment (RTE)
  - Middleware abstracting away from lower layers
- Application Software Components
  - Rely on strict interfaces, independent of MCU, Sensors, Actors
Main Takeaways

- ECUs
  - Principles
  - Architecture
  - Real-time properties (hard, firm, soft deadlines)

- OSEK/VDX
  - Motivation
  - Static configuration
  - Scheduling

- AUTOSAR
  - Motivation
  - Run Time Environment
  - Component Principle
Safety
Aspects of Safety

- Errors can lead to
  - material damage
  - bodily injury

- Check if errors might endanger human lives
  - Concerns not just systems for active safety (Airbag, ABS, ...)
  - Also concerns, e.g., engine ECU (sudden acceleration)

- Integral part of ECU development
  - “First and last step” when introducing new functionality
Aspects of Safety

- **Terminology**
  - **Risk:**
    - Quantitative measure of uncertainty
    \[
    \text{risk} = \text{occurrence probability} \times \text{consequences}
    \]
  - **Limit Risk:**
    - Highest still acceptable risk
  - **Safety:**
    - Condition that does not exceed limit risk
      (cf. DIN VDE 31000, Part 2)
Aspects of Safety

- **Terminology**
  - **Error:**
    - Deviation of calculated, observed, or measured value from true, specified, or theoretical value
  - **Fault:**
    - DIN 40041: unpermitted deviation of one or more properties that allows the discrimination of machines or components
    - IEC 61508, Part 4: exceptional condition that might lead to a component no longer fulfilling (or only partly fulfilling) its function
  - **Failure:**
    - DIN 40041: Component ceases to function (within permissible use)
    - IEC 61508, Part 4: System ceases fulfilling the specified function
  - **Malfunction:**
    - (Potentially dangerous) consequence of failure
      - E.g., ABS: failure must not cause wheels to lock; instead: graceful degradation
Aspects of Safety

- Terminology
  - Functional Safety:
    - Subpart of safety that is reliant on correct function of safety relevant components (as well as external measures for reducing risk)
  - Reliability:
    - Probability that a component does not fail within a defined time window
  - Redundancy:
    - Duplication of components (where only one would be needed)
    - homogeneous redundancy:
      - components are identical
    - diverse redundancy:
      - components are not identical
    - E.g., dual circuit braking
Aspects of Safety

- Laws and Norms
  - Laws
    - minimum conditions (in the shape of general requirements)
    - no verification
    - but: product liability laws might require proof that development corresponds to state of the art
    - E.g., German Regulations Authorizing the Use of Vehicles for Road Traffic (StVZO)
  - Norms
    - e.g. RTCA DO-178B (ED-12B) for aeronautic software
    - IEC 61508: standard for the development of safety critical systems
Aspects of Safety

- IEC 61508
  - Two modes of operation
    - Low Demand Mode
    - High Demand Mode or Continuous Mode

- Low Demand Mode
  - Safety critical system activated no more than twice per year (or maintenance interval)
  - Safety measures are passive (until needed)
  - E.g., airbag deployment on accident

- High Demand Mode or Continuous Mode
  - Safety critical system activated more than twice per year (or maintenance interval)
  - Safety measures keep system within safety margins
  - E.g., ensure that airbag cannot misfire
Aspects of Safety

- IEC 61508
  - 4 Safety Integrity Levels (SIL)
    - Relate to safety measure, not complete system
    - Describes risk reduction by safety measure
    - SIL 4: highest demands
      system failure triggers catastrophic consequences
  
- Process:
  - Damage and risk assessment
  - Determination of
    - Hardware Fault Tolerance (HFT)
    - Safety Failure Fraction (SFF)
  - Check if necessary SIL can be reached
Aspects of Safety

- ISO 26262
  - Adaption of IEC 61508 for automotive engineering
  - "Automotive 61508"
  - SIL ⇔ ASIL (Automotive Safety Integrity Level)
Aspects of Safety

- Methods for analyzing safety and reliability
  - Often rooted in aeronautic and space software development

- Covered in this lecture
  - Failure Mode Effect Analysis (FMEA)
    - Also called: Failure Mode Effect and Criticality Analysis (FMECA)
  - Fault Tree Analysis (FTA)
  - Event Tree Analysis (ETA)
Failure Mode Effect Analysis (FMEA)

- Step 1: list all possible failures
- Step 2: For each failure, list possible causes and consequences
  - Causes have causes
  - Consequences have consequences
  - Results in tree:

```
Cause 2.1  Cause 2  Cause 1
| Cause 2.2 |
| Cause 2.3 |
```

- Consequence 1
  - Consequence 1.1
- Consequence 2
  - Consequence 2.1
  - Consequence 2.2

- Ex: FMEA for Yacht Autopilot
  - Cause: Wind sensor imprecise
    ⇒ Failure: Wrong wind speed
    ⇒ Consequence: Wrong drift calculation
Failure Mode Effect Analysis (FMEA)

- Step 3: transform tree to table
  - Risk priority number = Probability × Severity × Detectability

<table>
<thead>
<tr>
<th>FMEA for Yacht Autopilot</th>
<th>Innsbruck, 2000-04-01</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>Component</td>
<td>Failure</td>
</tr>
<tr>
<td>Lead ship from A to B</td>
<td>Determine position</td>
<td>Wrong position</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Determine wind speed</td>
<td>Wrong wind speed</td>
<td>Wrong drift calculation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fault Tree Analysis (FTA)

- DIN 25424-1,2
- Tree structure of causes
  - i.e., left half of FMEA
- Failures depend on causes
- Leafs: elementary causes (those that do not stem from other causes)
- Connect with logical OR/AND
- Logical OR (if one cause suffices)
  e.g., brake pedal fails ⇒ brake fails (even if other components ok)
- Logical AND
  e.g., dual circuit brake (both circuits have to fail)
Fault Tree Analysis (FTA)

Vehicle does not brake

≥1

Brakes do not react & Brake pedal fails

&

Circuit 1 open & Circuit 2 open

Fault Tree Analysis (FTA)

- Qualitative representation as tree
- Quantitative calculation:
  - OR for exclusive events:
    - \[ p(c) = p(a) + p(b) \]
  - OR for arbitrary events:
    - \[ p(c) = p(a) + p(b) - p(a \times b) \]
  - AND for independent events:
    - \[ p(c) = p(a) \times p(b) \]
  - AND for dependent events:
    - \[ p(c) = p(a) + p(b|a) \]
- Difficulty: How probable are elementary events?
Event Tree Analysis (ETA)

- Analyzes consequences of faults (even if safety measures do not trigger)
- Process:
  - Start with individual fault
  - Fork, depending on which safety measures trigger
  - Multiple endings if more than one safety measure is in place
  - Results in tree of possible consequences
  - Limited quantitative analysis: hard to assign numbers to forks
Event Tree Analysis (ETA)

- Accelerator gets stuck
  - Yes
  - No
    - Brake pushed
    - Clutch disengaged
    - Motor stalled
    - ECU reacts to implausibility of accelerator and brake
      - Yes
      - No
        - No damage
        - Critical rpm
          - Yes
          - No
            - No damage
            - Critical rpm
              - Yes
              - Brake fails
                - No material damage or bodily harm
              - No
                - No damage
      - No damage
        - No damage
          - Yes
          - No
            - Yes
            - No
              - No damage
              - Critical rpm
                - Yes
                - No
                  - No damage
                  - Critical rpm
                    - Yes
                    - No
                      - No damage
                      - Material damage or bodily harm

Main Takeaways

▪ Aspects of Safety
  ▪ Motivation
  ▪ Terminology
  ▪ Failure Mode Effect Analysis (FMEA)
  ▪ Fault Tree Analysis (FTA)
  ▪ Event Tree Analysis (ETA)
  ▪ Commonalities and differences
Car-to-X (C2X) communication patterns

- Vehicle-to-X (V2X),
- Inter-Vehicle Communication (IVC),
- Vehicular ad-hoc network (VANET),
- ...

Illustration: ETSI
Use Cases
Taxonomy of Use Cases

Vehicle-to-X

Non-Safety
- Comfort
- Traffic Information Systems
- Contextual Information
- Entertainment
- Optimal Speed Advisory
- Congestion, Accident Information

Safety
- Situation Awareness
- Warning Messages
- Adaptive Cruise Control
- Blind Spot Warning
- Traffic Light Violation
- Electronic Brake Light
Taxonomy of Use Cases

Vehicle-to-X

Non-Safety
- Many messages
- High data rate
- Low latency demands
- Low reliability demands

Safety
- Few messages
- Small packet size
- High latency demands
- High reliability demands
# Diversity of use cases

<table>
<thead>
<tr>
<th>Application</th>
<th>Distance</th>
<th>Time</th>
<th>Recipient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hazard warning</td>
<td>250m</td>
<td>10s</td>
<td>All</td>
</tr>
<tr>
<td>Location based service</td>
<td>1..5km</td>
<td>Weeks</td>
<td>Subscribers</td>
</tr>
<tr>
<td>City wide alarm</td>
<td>20km</td>
<td>Hours</td>
<td>All</td>
</tr>
<tr>
<td>Travel time information</td>
<td>5km</td>
<td>Minutes</td>
<td>All</td>
</tr>
<tr>
<td>File sharing</td>
<td>250m</td>
<td>Minutes (Index)</td>
<td>Subscribers (Index)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Days (Content)</td>
<td>Peers (Content)</td>
</tr>
<tr>
<td>Interactive Services</td>
<td>1..5km</td>
<td>Minutes</td>
<td>Subscribers</td>
</tr>
</tbody>
</table>

## Diversity of requirements

<table>
<thead>
<tr>
<th>Application</th>
<th>Latency</th>
<th>Reliability</th>
<th># Vehicles</th>
<th>Area</th>
<th>Persistence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information Query</td>
<td>★</td>
<td>★</td>
<td>★★★★</td>
<td>★★★★</td>
<td></td>
</tr>
<tr>
<td>Hazard Warning</td>
<td>★★★★</td>
<td>★★</td>
<td>★</td>
<td>★★★★</td>
<td></td>
</tr>
<tr>
<td>ACC, el. Brake Light</td>
<td>★★★★</td>
<td>★★</td>
<td>★</td>
<td>★</td>
<td></td>
</tr>
<tr>
<td>Cooperative Awareness</td>
<td>★★</td>
<td>★★★</td>
<td>★</td>
<td>★</td>
<td>★</td>
</tr>
<tr>
<td>Intersection Assistance</td>
<td>★★</td>
<td>★★★</td>
<td>★★</td>
<td>★★</td>
<td>★</td>
</tr>
<tr>
<td>Platooning</td>
<td>★★★★</td>
<td>★★★</td>
<td>★★</td>
<td>★</td>
<td>★</td>
</tr>
</tbody>
</table>

Motivation

1970s: bold ideas
- Very visionary, infrastructure-less solutions
- Unsupported by current technology

Early interest of government and industry
- No commercial success

1980s: paradigm shift
- From complete highway automation ⇒ driver-advisory only
- Infrastructure-less ⇒ infrastructure-assisted
- chicken-and-egg type of standoff

New technology re-ignites interest
- latest-generation cellular communication ⇒ early “Car-to-X” systems

Sharp increase in computing power
- Supports fully-distributed, highly reactive ad hoc systems

Renewed interest of government and industry

- Numerous field operational tests
  - simTD (€ 69M), Aktiv (€ 60M), Smart Highway (€ 57M), Drive C2X (€ 19M), TeleFOT (€ 15M), SafeTrip (€ 10M), ...
- Dedicated spectrum in U.S., Europe, Asia

Illustration: FOT-NET Wiki
Motivation

- **Traditional Network**
  - Connection: wired
  - Nodes: non-moving
  - Configuration: static

- **Mobile Ad Hoc Network (MANET)**
  - Connection: wireless
  - Nodes: mobile
  - Configuration: dynamic
  - (Infrastructure: optional)

- **Vehicular Ad Hoc Network (VANET)**
  - Not: “MANET on wheels”
  - Different topology dynamics, communication patterns, infrastructure, ...

Freeway ↔ Urban

- 1D mobility
- Bimodal connectivity
  - Stable connection (vehicles on same lane)
  - AND
  - unstable connection (vehicles on opposite lane)
- High speed
- ...

- 2D mobility
- Bipolar connectivity
  - Many neighbors (when standing)
  - OR
  - Few neighbors (when driving)
- Obstacles
- ...

...
Levels of infrastructure support

- Pure ad hoc communication

- Stationary Support Units (SSU)
  - Radio-equipped autonomous computer
  - Inexhaustible storage, energy supply
  - Known position, high reliability

- Roadside Units (RSU)
  - SSU plus...
  - Ethernet NIC, UMTS radio, ...
  - Connected to other RSUs

- Traffic Information Center (TIC)
  - Central server connected to RSUs
Infrastructure ⇔ No Infrastructure

- Central coordination
  - Resource management
  - Security
- High latency
- High load on core network
- ...

- Self organizing system
  - Channel access
  - Authentication
- Low latency
- Low data rate
- ...

Source: AKTIV CoCar
Convergence towards heterogeneous approaches

- Same system needs to work in multiple environments
  - Vehicle starts to drive in city with infrastructure support
  - Continues driving on freeway (still with infrastructure support)
  - Loses infrastructure support when turning onto local highway
  - Finishes driving in city without infrastructure support
Adoption

- Prognosis (of providers!) in Germany and the U.S.
  - 14..15 years to 100% market penetration
- Compare to navigation systems in German cars
  - 13 years to 14% market penetration
  - And: it is very easy to retrofit a satnav!

Challenges of C2X communication

**Communication**
- Highly varying channel conditions
- High congestion, contention, interference
- Tightly limited channel capacity

**Networking**
- Unidirectional Links
- Multi-Radio / Multi-Network
- Heterogeneous equipment

**Mobility**
- Highly dynamic topology
- But: predictable mobility
- Heterogeneous environment

**Security**
- No (or no reliable) uplink to central infrastructure
- Ensuring privacy
- Heterogeneous user base
Technology
Communication paradigms and media

<table>
<thead>
<tr>
<th>Wireless Communication Technologies</th>
<th>Infrastructure-based</th>
<th>Infrastructureless</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broadcast</td>
<td>Cellular</td>
<td>Short Range</td>
</tr>
<tr>
<td>FM Radio, DAB/DVB, ...</td>
<td>GSM 2G Cellular</td>
<td>802.15.1 Bluetooth</td>
</tr>
<tr>
<td></td>
<td>UMTS 3G Cellular</td>
<td>802.15.4 ZigBee</td>
</tr>
<tr>
<td></td>
<td>LTE / WiMAX 4G Cell.</td>
<td>802.11 Wi-Fi</td>
</tr>
<tr>
<td></td>
<td>Millimeter, Infrared, Visible</td>
<td>DSRC / WAVE</td>
</tr>
</tbody>
</table>

Broadcast Media

- **Traffic Message Channel (TMC)**
  - Central management of traffic information
  - Data sources are varied
    - Federal/local/city police, road operator, radio, ...
  - Transmission in RDS channel of FM radio
    - BPSK modulated signal at 57 KHz, data rate 1.2 kBit/s
    - RDS group identifier 8A (TMC), approx. 10 bulletins per minute

---

[1] ISO 62106, „Specification of the radio data system (RDS) for VHF/FM sound broadcasting in the frequency range from 87.5 to 108.0 MHz“
Broadcast Media

- Traffic Message Channel (TMC)
  - Contents (ALERT-C coded):
    - Validity period
    - Re-routing required?
    - North-east or south-west?
    - Spatial extent
    - Code in event table
      - International
    - Code in location table
      - Country/region specific
      - Must be installed in end device
  - No (real) security measures

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>Standing traffic (generic)</td>
</tr>
<tr>
<td>102</td>
<td>1 km of standing traffic</td>
</tr>
<tr>
<td>103</td>
<td>2 km of standing traffic</td>
</tr>
<tr>
<td>394</td>
<td>Broken down truck</td>
</tr>
<tr>
<td>1478</td>
<td>Terrorist incident</td>
</tr>
<tr>
<td>1</td>
<td>Deutschland</td>
</tr>
<tr>
<td>264</td>
<td>Bayern</td>
</tr>
<tr>
<td>12579</td>
<td>A8 Anschlussstelle Irschenberg</td>
</tr>
</tbody>
</table>


Broadcast Media

- Traffic Message Channel (TMC)
  - Regional value added services
    - Navteq Traffic RDS (U.S.), trafficmaster (UK), V-Trafic (France)

- Ex: TMCpro
  - Private service of Navteq Services GmbH
  - Financed by per-decoder license fee
  - Data collection and processing
    - Fully automatic
    - Deployment of 4000+ sensors on overpasses
    - Use of floating car data
    - Downlink from traffic information centers
  - Event prediction
    - Expert systems, neuronal networks
    - Early warnings of predicted events
  - Restricted to major roads
Broadcast Media

- Transport Protocol Experts Group (TPEG)
  - Planned successor of RDS-TMC/Alert-C
  - Published April 2000
  - Principles:
    - Extensibility
    - Media independence

- Goals:
  - Built for “Digital Audio Broadcast” (DAB)
  - Unidirectional, byte oriented stream
  - Modular concept
  - Hierarchical approach
  - Integrated security

Broadcast Media

- Transport Protocol Experts Group (TPEG)
  - Information types defined by “TPEG Applications”
    - RTM - Road Traffic Message
    - PTI - Public Transport Information
    - PKI - Parking Information
    - CTT - Congestion and Travel-Time
    - TEC - Traffic Event Compact
    - WEA - Weather information for travelers
- Modular concept:
Transport Protocol Experts Group (TPEG)

- tpegML: XML variant of regular (binary) TPEG

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE tpeg_document PUBLIC "-//EBU/tpegML/EN"
 "http://www.bbc.co.uk/travelnews/xml/tpegml_en/tpegML.dtd">
<tpg_document generation_time="2007-09-19T07:22:44+0">
  <tpg_message>
    <originator country="UK" originator_name="BBC Travel News"/>
    <summary xml:lang="en">M5 Worcestershire – Earlier accident southbound between J5, Droitwich and J6, Worcester, heavy traffic.</summary>
    <road_traffic_message>
      <!-- ... tpeg-rtML ... -->
    </road_traffic_message>
  </tpg_message>
  <tpg_message>
    <originator country="UK" originator_name="BBC Travel News"/>
    <summary xml:lang="en">A420 Oxfordshire – The Plain closed westbound at the A4158 Iffley Road junction in Oxford, delays expected. Diversion in operation.</summary>
    <road_traffic_message>
      <!-- ... tpeg-rtML ... -->
    </road_traffic_message>
  </tpg_message>
</tpg_document>
```

Transport Protocol Experts Group (TPEG)

- Hybrid approach to geo-referencing: one or more of
  - WGS84 based coordinates
  - ILOC (Intersection Location)
    - Normalized, shortened textual representation of street names intersecting at desired point
  - Human readable plain text
  - Code in hierarchical location table

![TPEG Location Table](image-url)
Cellular Networks

- Concept
  - Divide world into cells, each served by base station
  - Allows, e.g., frequency reuse in FDMA
Concept

- Strict hierarchy of network components
Cellular Networks

- Can UMTS support Car-to-X communication?
  - Ex: UTRA FDD Release 99 (W-CDMA)
  - Speed of vehicles not a limiting factor
    - Field operational tests at 290 km/h show signal drops only after sudden braking (handover prediction failures)

- Open questions
  - Delay
  - Capacity

- Channels in UMTS
  - Shared channels
    - E.g. Random Access Channel (RACH), uplink and Forward Access Channel (FACH), downlink
  - Dedicated channels
    - E.g. Dedicated Transport Channel (DCH), up-/downlink
Cellular Networks

- **FACH**
  - Time slots managed by base station
  - Delay on the order of 10 ms per 40 Byte and UE
  - Capacity severely limited (in non-multicast networks)
  - Need to know current cell of UE

- **RACH**
  - Slotted ALOHA – random access by UEs
    - Power ramping with Acquisition Indication
  - Delay approx. 50 ms per 60 Byte and UE
  - Massive interference with other UEs
Cellular Networks

- **DCH**
  - Delay: approx. 250 ms / 2 s / 10 s for channel establishment
    - Depends on how fine-grained UE position is known
  - Maintaining a DCH is expensive
    - Closed-Loop Power Control (no interference of other UEs)
    - Handover between cells
    - ...
  - Upper limit of approx. 100 UEs
Cellular Networks

- So: can UMTS support Car-to-X communication?
  - At low market penetration: yes
  - Eventually:
    - Need to invest in much smaller cells (e.g., along freeways)
    - Need to implement multicast functionality (MBMS)
- Main use case for UMTS: centralized services
  - Ex.: Google Maps Traffic
    - Collect information from UMTS devices
    - Storage of data on central server
    - Dissemination via Internet (⇒ ideal for cellular networks)
IEEE 802.11p

- **IEEE 802.11\{a,b,g,n\} for Car-to-X communication?**
  - Can’t be in infrastructure mode and ad hoc mode at the same time
  - Switching time consuming
  - Association time consuming
  - No integral within-network security
  - (Massively) shared spectrum (⇒ ISM)
  - No integral QoS
  - Multi-path effects reduce range and speed
IEEE 802.11p

- **IEEE 802.11p**
  - PHY layer mostly identical to IEEE 802.11a
    - Variant with OFDM and 16 QAM
    - Higher demands on tolerances
    - Reduction of inter symbol interference because of multi-path effects
      - Double timing parameters
      - Channel bandwidth down to 10 MHz (from 20 MHz)
      - Throughput down to 3 ... 27 Mbit/s (from 6 ... 54 Mbit/s)
      - Range up to 1000 m, speed up to 200 km/h
  - MAC layer of IEEE 802.11a plus extensions
    - Random MAC Address
    - QoS (EDCA priority access, cf. IEEE 802.11e, ...)
    - Multi-Frequency and Multi-Radio capabilities
    - New Ad Hoc mode
    - ...
IEEE 802.11p

- Classic IEEE 802.11 Basic Service Set (BSS)
  - Divides networks into logical units
    - Nodes belong to (exactly one) BSS
    - Packets contain BSSID
    - Nodes ignore packets from “foreign” BSSs
    - Exception: Wildcard-BSSID (-1) for probes
    - Ad hoc networks emulate infrastructure mode

- Joining a BSS
  - Access Point sends beacon
  - Authentication dialogue
  - Association dialogue
  - Node has joined BSS
IEEE 802.11p

- New: 802.11 WAVE Mode
  - Default mode of nodes in WAVE
  - Nodes may always use Wildcard BSS in packets
  - Nodes will always receive Wildcard BSS packets
  - May join BSS and still use Wildcard BSS
IEEE 802.11p

- New: 802.11 WAVE BSS
  - No strict separation between Host and Access Point (AP)
  - Instead, loose classification according to:
    - Equipment: Roadside Unit (RSU) / On-Board Unit (OBU)
    - Role in data exchange: Provider / User
  - No technical difference between Provider and User
  - Provider sends On-Demand Beacon
    - Analogous to standard 802.11-Beacon
    - Beacon contains all information and parameters needed to join
  - User configures lower layers accordingly
    - Starts using provided service
    - No additional exchange of data needed
  - BSS membership now only implied
    - BSS continues to exist even after provider leaves
WAVE BSS Internal state machine

- Node will not join more than one WAVE BSS

- WAVE Mode only
  - Application layer stops service
  - Timeout
  - Security error

- In WAVE BSS
  - Application layer starts new service
  - On-Demand-Beacon received

- Application layer starts new, higher priority service
  - On-Demand-Beacon for known, higher priority service received

IEEE 802.11p

- IEEE 802.11 Distributed Coordination Function (DCF)
  - aka “Contention Period”

  SIFS, PIFS, DIFS

  Medium busy  TX starts

  Time slots in contention period

- Priority access via SIFS (ACK, CTS, ...) and DIFS (payload)
- Wait until medium has been free for duration of DIFS
- If medium busy, wait until idle, then wait DIFS plus random backoff time
IEEE 802.11p

- IEEE 802.11 Distributed Coordination Function (DCF)
  - Backoff if
    a) Node is ready to send and channel becomes busy
    b) A higher priority queue (⇨ next slides) becomes ready to send
    c) Unicast transmission failed (no ACK)
    d) Transmission completed successfully

- Backoff: Random slot count from interval [0, CW]
- Decrement by one after channel was idle for one slot (only in contention period)
- In cases b) and c), double CW (but no larger than $CW_{\text{max}}$)
- In case d), set CW to $CW_{\text{min}}$
IEEE 802.11p

- QoS in 802.11p (HCF)
  - cf. IEEE 802.11e EDCA
  - DIFS ⇔ AIFS (Arbitration Inter-Frame Space)
    - DCF ⇔ EDCA (Enhanced Distributed Channel Access)

- Classify user data into 4 ACs (Access Categories)
  - AC0 (lowest priority)
  - ...
  - AC3 (highest priority)

- Each ACs has different...
  - $CW_{\text{min}}, CW_{\text{max}}, AIFS, TXOP$ limit (max. continuous transmissions)

- Management data uses DIFS (not AIFS)
IEEE 802.11p

- QoS in 802.11p (HCF)
  - Map 8 user priorities $\mapsto$ 4 access categories $\mapsto$ 4 queues
  - Queues compete independently for medium access
IEEE 802.11p

- QoS in 802.11p (HCF)
  - Parameterization

<table>
<thead>
<tr>
<th>Parameter</th>
<th>AC_BK</th>
<th>AC_BE</th>
<th>AC_VI</th>
<th>AC_VO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CW_min</td>
<td>CW_min</td>
<td>CW_min</td>
<td>(CW_min+1)/2-1</td>
<td>(CW_min+1)/4-1</td>
</tr>
<tr>
<td>CW_max</td>
<td>CW_max</td>
<td>CW_max</td>
<td>CW_min</td>
<td>(CW_min+1)/2-1</td>
</tr>
<tr>
<td>AIFSn</td>
<td>9</td>
<td>6</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SlotTime</td>
<td>13µs</td>
</tr>
<tr>
<td>SIFS</td>
<td>32µs</td>
</tr>
<tr>
<td>CW_min</td>
<td>15</td>
</tr>
<tr>
<td>CW_max</td>
<td>1023</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>3 .. 27 mbit/s</td>
</tr>
</tbody>
</table>
IEEE 802.11p

Channel Access

Backoff: 0

AC_VO

AC_VI

AC_BE

AC_BK
IEEE 802.11p

Channel busy?

Wait for Idle

Start Contention

Channel Access

Backoff: 0

AC_VO

AC_VI

AC_BE

AC_BK
IEEE 802.11p

Backoff 0?

- Wait for backoff = 0
- Wait AIFS (SIFS + AIFSn * Slot len)

Channel Access

AC_VO
AC_VI
AC_BE
AC_BK

Backoff: 0

Wait
AIFS
(SIFS + AIFSn * Slot len)
IEEE 802.11p

AC_VO  
AC_VI  
AC_BE  
AC_BK

Backoff: 0  
0  
0  
0

Transmission Over

Post Transmit Backoff

Channel Access
IEEE 802.11p

Backoff: 2

AC_VO

AC_VI

AC_BE

AC_BK

Ch becomes busy

AC_VI Queue ready to send... wait AIFS

Backoff

Channel Access
IEEE 802.11p

Channel idle

Channel busy

[Slot time passed] /Decrement Backoff

Channel state changes

AC_VO

AC_VI

AC_BE

AC_BK

Channel Access
**IEEE 802.11p**

- Queue ready to send
- Higher priority queue ready
- Internal Contention Backoff

![Diagram](image)

- Channel Access
IEEE 802.11p

- QoS in WAVE
  - mean waiting time for channel access, given sample configuration (and TXOP Limit=0 ⇔ single packet)
  - when channel idle:
  - when channel busy:

<table>
<thead>
<tr>
<th>AC</th>
<th>CW_{min}</th>
<th>CW_{max}</th>
<th>AIFS</th>
<th>TXOP</th>
<th>t_w (in μs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>15</td>
<td>1023</td>
<td>9</td>
<td>0</td>
<td>264</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>15</td>
<td>6</td>
<td>0</td>
<td>152</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>7</td>
<td>3</td>
<td>0</td>
<td>72</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>7</td>
<td>2</td>
<td>0</td>
<td>56</td>
</tr>
</tbody>
</table>

Figure Source: Eichler, S., "Performance evaluation of the IEEE 802.11p WAVE communication standard," Proceedings of 66th IEEE Vehicular Technology Conference (VTC2007-Fall), Baltimore, USA, October 2007, pp. 2199-2203
UMTS/LTE vs. 802.11p

- **Pros of UMTS/LTE**
  + Easy provision of centralized services
  + Quick dissemination of information in whole network
  + Pre-deployed infrastructure
  + Easy migration to (and integration into) smartphones

- **Cons of UMTS/LTE**
  - High short range latencies (might be too high for safety)
  - Network needs further upgrades (smaller cells, multicast service)
  - High dependence on network operator
  - High load in core network, even for local communication
UMTS/LTE vs. IEEE 802.11p

- **Pros of 802.11p/Ad hoc**
  + Smallest possible latency
  + Can sustain operation without network operator / provider
  + Network load highly localized
  + Better privacy (⇒ later slides)

- **Cons of 802.11p/Ad hoc**
  - Needs gateway for provision of central services (e.g., RSU)
  - No pre-deployed hardware, and hardware is still expensive

- **The solution?**
  - hybrid systems:
    deploy both technologies to vehicles and road,
    decide depending on application and infrastructure availability
Higher Layer Standards: CALM

- Mixed-media communication
  - „Communications access for land mobiles“
  - ISO TC204 Working Group 16
  - Initiative to transparently use best possible medium
- Integrates:
  - GPRS, UMTS, WiMAX
  - Infrared, Millimeter Wave
  - Wi-Fi, WAVE
  - Unidirectional data sources (DAB, GPS, ...)
  - WPANs (BlueT, W-USB, ...)
  - Automotive bus systems (CAN, Ethernet, ...)

Higher Layer Standards for IEEE 802.11p

- **Channel management**
  - Dedicated frequency band at 5.9 GHz allocated to WAVE
    - Exclusive for V2V und V2I communication
    - No license cost, but strict rules
    - 1999: FCC reserves 7 channels of 10 MHz ("U.S. DSRC")
      - 2 reserved channels, 1+4 channels for applications
    - ETSI Europe reserves 5 channels of 10 MHz

<table>
<thead>
<tr>
<th>U.S. allocation</th>
<th>Critical Safety of Life</th>
<th>SCH</th>
<th>SCH</th>
<th>Control Channel (CCH)</th>
<th>SCH</th>
<th>SCH</th>
<th>Hi-Power Public Safety</th>
</tr>
</thead>
<tbody>
<tr>
<td>European allocation</td>
<td>SCH</td>
<td>SCH</td>
<td>SCH</td>
<td>SCH</td>
<td>CCH</td>
<td>SCH</td>
<td>SCH</td>
</tr>
<tr>
<td>IEEE Channel</td>
<td>172</td>
<td>174</td>
<td>176</td>
<td>178</td>
<td>180</td>
<td>182</td>
<td>184</td>
</tr>
<tr>
<td>Center frequency</td>
<td>5.860 GHz</td>
<td>5.870 GHz</td>
<td>5.880 GHz</td>
<td>5.890 GHz</td>
<td>5.900 GHz</td>
<td>5.910 GHz</td>
<td>5.920 GHz</td>
</tr>
</tbody>
</table>

[1] ETSI ES 202 663 V1.1.0 (2010-01) : Intelligent Transport Systems (ITS); European profile standard for the physical and medium access control layer of Intelligent Transport Systems operating in the 5 GHz frequency band
Higher Layer Standards for IEEE 802.11p

- Need for higher layer standards
  - Unified message format
  - Unified interfaces to application layer

- U.S.
  - IEEE 1609.*
  - WAVE („Wireless Access in Vehicular Environments“)

- Europe
  - ETSI
  - ITS G5 („Intelligent Transportation Systems“)
IEEE 1609.* upper layers (building on IEEE 802.11p)

- IEEE 1609.2: Security
- IEEE 1609.3: Network services
- IEEE 1609.4: Channel mgmt.
- IEEE 1609.11: Application “electronic payment”

IEEE 1609

- Channel management
  - WAVE allows for both single radio devices & multi radio devices
  - Dedicated Control Channel (CCH) for mgmt and safety messages
    ⇝ single radio devices need to periodically listen to CCH
- Time slots
  - Synchronization envisioned via GPS receiver clock
  - Standard value: 100ms sync interval (with 50ms on CCH)
  - Short guard interval at start of time slot
    - During guard, medium is considered busy (⇝ backoff)

IEEE 1609

- Packet transmission
  - Sort into AC queue, based on WSMP (or IPv6) EtherType field, destination channel, and user priority
  - Switch to desired channel, setup PHY power and data rate
  - Start medium access
IEEE 1609

- Channel management
  - Control Channel (CCH):
    - Default channel upon initialization
    - WAVE service advertisements (WSA), WAVE short messages (WSM)
    - Channel parameters take fixed values
  - Service Channel (SCH):
    - Only after joining WAVE BSS
    - WAVE short messages (WSM), IP data traffic (IPv6)
    - Channel parameters can be changed as needed
IEEE 1609

- WAVE service advertisement (WSA)
  - Broadcast on Control Channel (CCH)
  - Identifies WAVE BSSs on Service Channels (SCHs)
  - Can be sent at arbitrary times, by arbitrary nodes
  - Only possibility to make others aware of data being sent on SCHs, as well as the required channel parameters to decode them
IEEE 1609

- WAVE service advertisement (WSA)
  - WAVE Version (= 0)
  - Provider Service Table (PST)
    - n × Provider Service Info
      - Provider Service Identifier (PSID, max. 0x7FFF FFFF)
      - Provider Service Context (PSC, max. 31 chars)
      - Application priority (max priority: 63)
      - (opt.: IPv6 address and port, if IP service)
      - (opt.: Source MAC address, if sender ≠ data source)
      - Channel number (max. 200)
    - 1..n × Channel Info (for each channel used in PST table)
      - Data rate (fixed or minimum value)
      - Transmission power (fixed or maximum value)
  - (opt.: WAVE Routing Announcement)

WAVE service advertisement (WSA)

- Provider Service Identifier (PSID) defined in IEEE Std 1609.3-2007

<table>
<thead>
<tr>
<th>PSID</th>
<th>Service Description</th>
<th>Provider Service Identifier (PSID)</th>
<th>Provider Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x000 0000</td>
<td>system</td>
<td>0x000 000D</td>
<td>private</td>
</tr>
<tr>
<td>0x000 0001</td>
<td>automatic-fee-collection</td>
<td></td>
<td>multi-purpose-payment</td>
</tr>
<tr>
<td>0x000 0002</td>
<td>freight-fleet-management</td>
<td></td>
<td>dsrs-resource-manager</td>
</tr>
<tr>
<td>0x000 0003</td>
<td>public-transport</td>
<td></td>
<td>after-theft-systems</td>
</tr>
<tr>
<td>0x000 0004</td>
<td>traffic-traveler-information</td>
<td>0x000 0011</td>
<td>cruise-assist-highway-system</td>
</tr>
<tr>
<td>0x000 0005</td>
<td>traffic-control</td>
<td>0x000 0012</td>
<td>multi-purpose-information system</td>
</tr>
<tr>
<td>0x000 0006</td>
<td>parking-management</td>
<td>0x000 0013</td>
<td>public-safety</td>
</tr>
<tr>
<td>0x000 0007</td>
<td>geographic-road-database</td>
<td>0x000 0014</td>
<td>vehicle-safety</td>
</tr>
<tr>
<td>0x000 0008</td>
<td>medium-range-preinformation</td>
<td>0x000 0015</td>
<td>general-purpose-internet-access</td>
</tr>
<tr>
<td>0x000 0009</td>
<td>man-machine-interface</td>
<td>0x000 0016</td>
<td>onboard diagnostics</td>
</tr>
<tr>
<td>0x000 000A</td>
<td>intersystem-interface</td>
<td>0x000 0017</td>
<td>security manager</td>
</tr>
<tr>
<td>0x000 000B</td>
<td>automatic-vehicle-identification</td>
<td>0x000 0018</td>
<td>signed WSA</td>
</tr>
<tr>
<td>0x000 000C</td>
<td>emergency-warning</td>
<td>0x000 0019</td>
<td>ACI</td>
</tr>
</tbody>
</table>
IEEE 1609

- **WAVE Short Message (WSM)**
  - **Header (11 Byte)**
    - Version (= 0)
    - Content type: plain, signed, encrypted
    - Channel number (max. 200)
    - Data rate
    - Transmission power
    - Provider Service Identifier (Service type, max. 0x7FFF FFFF)
    - Length (max. typ. 1400 Bytes)
  - **Payload**
IEEE 1609

- IP traffic (UDP/IPv6 or TCP/IPv6)
  - Header (40+n Byte)
    - Version
    - Traffic Class
    - Flow Label
    - Length
    - Next Header
    - Hop Limit
    - Source address, destination address
    - (opt.: Extension Headers)
  - Payload

- No IPv6-Neighbor-Discovery (High overhead)
- All OBUs listen to host multicast address, all RSUs listen to router multicast address
IEEE 1609

- Channel quality monitoring
  - Nodes store received WSAs, know SCH occupancy
  
  - Received Channel Power Indicator (RCPI) polling
    - Nodes can send RCPI requests
    - Receiver answers with Received Signal Strength (RSS) of packet

- Transmit Power Control (TPC)
  - Nodes can send TPC requests
  - Receiver answers with current transmission power and LQI

- Dynamic Frequency Selection (DFS)
  - Nodes monitor transmissions on channel (actively and passively)
  - If higher priority third party use (e.g., RADAR) is detected, nodes cease transmitting
IEEE 1609

- **Security in WAVE**
  - Nature of WAVE messages mandates trust between nodes
    - Ex: Green wave for emergency vehicles
  - Security is built into WAVE (IEEE 1609.2)
  - WAVE can transparently sign, verify, encrypt/decrypt messages when sending and receiving
    - Ex: WSA → Secure WSA
  - Authorization of messages needed
    - By role: CA, CRL-Signer, RSU, Public Safety OBU (PSOBU), OBU
    - By application class (PSID) and/or instance (PSC)
    - By application priority
    - By location
    - By time

IEEE 1609

- Security concepts
  - Basic security goals
    - Integrity, Confidentiality, Authenticity
    - Non-Repudiation
  - Mechanisms
    - Symmetric encryption
      - Secret Key Cryptography
      - Ex: Caesar cipher, Enigma, AES
    - Asymmetric encryption
      - Public Key Cryptography
      - Ex: RSA, ElGamal, ECC
    - (cryptographic) hashing
      - Ex: MD5, SHA-1
Asymmetric Cryptography

- Relies on certain mathematical procedures being very hard to invert
  - Product $\leftrightarrow$ factorization (RSA)
  - Nth power $\leftrightarrow$ Nth logarithm (DH, ElGamal)
- Two keys: Public Key ($K^+$), Private Key ($K^-$)

- Can be used in both directions
  - Encryption: $E(K^+, m)$, Signing: $E(K^-, h(m))$

- Drawback:
  - Much slower than symmetric cryptography
IEEE 1609

- Asymmetric Cryptography Example: RSA
  - Chose two primes: q, p with q \neq p
  - Calculate n = p \cdot q
  - Calculate \( \phi(n) = (p - 1) \cdot (q - 1) \)
    - \( \phi(x) \) gives number of (smaller) co-primes for \( x \).
    - Based on \( \phi(a \cdot b) = \phi(a) \cdot \phi(b) \cdot (d/\phi(d)) \) with \( d = \gcd(a, b) \)
      If \( x \) is prime, this is \( x - 1 \).
  - Choose \( e \) co-prime to \( \phi(n) \) with \( 1 < e < \phi(n) \)
  - Calculate \( d \) using EEA, so that \( e \cdot d \mod \phi(n) = 1 \)
  - Public Key: \( K^+ = \{e, n\} \), Private Key: \( K^- = \{d, n\} \).
  - En/Decryption:
    - \( M^e \mod n = C \)
    - \( C^d \mod n = M \)
IEEE 1609

- Certificates
  - Encryption is useless without authentication
    - Alice $\leftrightarrow$ Eve $\leftrightarrow$ Bob
    - Eve can pretend to be Alice, replace $K^+_A$ with own key $K^+_E$

- Solution: use Trusted Third Party (TTP) and certificates
  - TTP signs (Name, Key) tuple, vouches for validity and authorization:
    “Alice has Public Key $K^+_A$, may participate as OBU until 2019”
  - not: “whoever sends this packet is Alice”
  - not: “whoever sends this packet has Public Key $K^+_A$”

- Send $K^+_A$ together with certificate vouching for tuple
IEEE 1609

- Implementation in WAVE
  - Certificate signature chains
    - Root certificate $\rightarrow$ certificate $\rightarrow$ certificate $\rightarrow$ payload
    - Root certificates pre-installed with system
    - Other certificates cannot be assumed to be present

- Nodes must download certificates and:
  - Include chain of certificates
  - ...or SHA-256 of first certificate in chain

(if receiver can be assumed to have all required certificates)
IEEE 1609

- Implementation in WAVE
  - X.509 formats too large ⇒ new WAVE certificate format
    - Version
    - Certificate
      - Role (RSU, PSOBU, OBU, ...)
      - Identity (dependent on role)
      - Restrictions (by application class, priority, location, ...)
      - Expiration date
      - Responsible CRL
      - Public Keys
    - Signature
  
  - New: Restriction by location
e.g.: none, inherited from CA, circle, polygon, set of rectangles

  - Public Key algorithms (motivated by key size):
    ECDSA (NIST p224), ECDSA (NIST p256), ECIES (NIST p256), ...
Complete packet format of a WSM:

<table>
<thead>
<tr>
<th>Length</th>
<th>Field</th>
<th>Signer</th>
<th>Type = certificate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WSM version</td>
<td>certificate</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Security Type = signed(1)</td>
<td>message flags</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Channel Number</td>
<td>application_data</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Data Rate</td>
<td>transmission_time</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>TxPwr_Level</td>
<td>transmission_location</td>
<td>latitude</td>
</tr>
<tr>
<td>4</td>
<td>PSID</td>
<td>elevation_and_confidence</td>
<td>longitude</td>
</tr>
<tr>
<td>1</td>
<td>PSC Field Length</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>PSC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>WSM Length</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>signer</td>
<td>signature</td>
<td></td>
</tr>
<tr>
<td>125</td>
<td>WSM Data</td>
<td>ecdsa_signature</td>
<td>r</td>
</tr>
<tr>
<td>2</td>
<td>unsigned_wsm</td>
<td></td>
<td>s</td>
</tr>
<tr>
<td>32</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Complete packet format of a WSM (certificate part):

<table>
<thead>
<tr>
<th>Length</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>certificate_version = 1</td>
</tr>
<tr>
<td>1</td>
<td>subject_type = obu_identified</td>
</tr>
<tr>
<td>8</td>
<td>signer_id</td>
</tr>
<tr>
<td>1</td>
<td>scope</td>
</tr>
<tr>
<td>8</td>
<td>subject_name_length</td>
</tr>
<tr>
<td>2</td>
<td>applications</td>
</tr>
<tr>
<td>1</td>
<td>type = from_issuer</td>
</tr>
<tr>
<td>4</td>
<td>expiration</td>
</tr>
<tr>
<td>4</td>
<td>crl_series</td>
</tr>
<tr>
<td>1</td>
<td>length of public key field</td>
</tr>
<tr>
<td>1</td>
<td>public_key</td>
</tr>
<tr>
<td>29</td>
<td>algorithm = ecdsa_nistp224..</td>
</tr>
<tr>
<td>32</td>
<td>signature</td>
</tr>
<tr>
<td>32</td>
<td>ecdsa_signature</td>
</tr>
</tbody>
</table>

| r       |
| s       |
Drawbacks of Channel Switching

1) Goodput
   - User data must only be sent on SCH, i.e. during SCH interval ⇒ goodput cut in half

Drawbacks of Channel Switching

- 2) Latency
  - User data generated during CCH interval is delayed until SCH intv.

Drawbacks of Channel Switching

- 3) Collisions
  - Delay of data to next start of SCH interval
    ⇒ increased frequency of channel accesses directly after switch
    ⇒ increased collisions, packet loss

ETSI ITS G5

- Motivation
  - European standardization effort based on IEEE 802.11p
  - Standardization to include lessons learned from WAVE
  - Different instrumentation of lower layers
  - Different upper layer protocols
  - Fine-grained service channel assignment
    - ITS-G5A (safety)
    - IST-G5B (non safety)
ETSI ITS G5

- Protocol stack
  - PHY and MAC based on IEEE 802.11p
  - Most prominent change: cross layer Decentralized Congestion Control (DCC)

![Diagram of protocol stack]

- Applications Layer
- Facilities Layer
- Networking & Transport Layer
- Access Layer
- Management
- DCC
- Security
ETSI ITS G5

- Channel management
  - Multi radio, multi antenna system
    - No alternating access
      ⇒ Circumvents problems with synchronization
      ⇒ No reduction in goodput
    - Direct result of experiences with WAVE

- One radio tuned to CCH
  - Service Announcement Message (SAM)
  - Periodic:
    Cooperative Awareness Messages (CAM)
  - Event based:
    Decentralized Environment Notification Message (DENM)

- Addl. radio tuned to SCH
  - User data
ETSI ITS G5

- Medium access
  - Separate EDCA systems
  - Different default parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>AC_BK</th>
<th>AC_BE</th>
<th>AC_VI</th>
<th>AC_VO</th>
</tr>
</thead>
<tbody>
<tr>
<td>CW_{min}</td>
<td>CW_{min}</td>
<td>(CW_{min}+1)/2-1</td>
<td>(CW_{min}+1)/4-1</td>
<td>(CW_{min}+1)/4-1</td>
</tr>
<tr>
<td>CW_{max}</td>
<td>CW_{max}</td>
<td>CW_{max}</td>
<td>(CW_{min}+1)/2-1</td>
<td>(CW_{min}+1)/2-1</td>
</tr>
<tr>
<td>AIFS_{n}</td>
<td>9</td>
<td>6</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

- *Contention Window* – less distance to lower priority queues
  ⇒ less starvation of lower priority queues
ETSI ITS G5

- DCC
  - Core feature of ETSI ITS G5
  - Adaptive parameterization to avoid overload
  - Configurable parameters per AC:
    - TX power
    - Minimum packet interval
    - Sensitivity of CCA (Clear Channel Assessment)
    - Data rate
  - State machine determines which parameters are selected; available states:
    - Relaxed
    - Active (multiple sub states)
    - Restrictive
ETSI ITS G5

- DCC
  - State machine for Control Channel:

    \[
    \text{minChannelLoad(1s)} \geq 15\% \\
    \text{minChannelLoad(1s)} \geq 40\% \\
    \text{maxChannelLoad(5s)} < 15\% \\
    \text{maxChannelLoad(5s)} < 40\%
    \]

    - \text{min/maxChannelLoad(x)}: record fraction of time in \([t_{\text{now}}-x..t_{\text{now}}]\) that channel was sensed busy
    - subdivide interval into equal parts (e.g. 50 ms), take min/max
    - Channel busy ↔ measured received power (signal or noise) above configured sensibility
ETS I TS G5

- DCC
  - Selection of parameters when changing states
  - Service Channel: Active state has four sub-configurations
  - Control Channel: Single configuration for active state
  - Example (“ref”: Value remains unchanged)

<table>
<thead>
<tr>
<th>State</th>
<th>Relaxed</th>
<th>Active</th>
<th>Active</th>
<th>Active</th>
<th>Active</th>
<th>Restrictive</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TX power</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>33dBm</td>
<td>ref</td>
<td>25dBm</td>
<td>20dBm</td>
<td>15dBm</td>
<td>-10dBm</td>
</tr>
<tr>
<td></td>
<td>Min pkt interval</td>
<td>0,04s</td>
<td>ref</td>
<td>ref</td>
<td>ref</td>
<td>ref</td>
</tr>
<tr>
<td></td>
<td>Data rate</td>
<td>3Mbit/s</td>
<td>ref</td>
<td>ref</td>
<td>ref</td>
<td>ref</td>
</tr>
<tr>
<td></td>
<td>Sensitivity</td>
<td>-95 dBm</td>
<td>ref</td>
<td>ref</td>
<td>ref</td>
<td>ref</td>
</tr>
</tbody>
</table>
ETSI ITS G5

- Cooperative Awareness Message
  - Periodic (up to 10Hz) safety message
  - Information on state of surrounding vehicles:
    - Speed, location, ...
  - Message age highly relevant for safety
    - Need mechanisms to discard old messages
  - Safety applications rely on CAMs:
    - Tail end of jam
    - Rear end collision
    - Intersection assistance...
- Sent on CCH
- Generated every 100ms .. 1s, but only if Δangle (>4°), Δposition (>5m), Δspeed (>1m/s)
## ETSI ITS G5

<table>
<thead>
<tr>
<th>Length[byte]</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>msgageld (0=CAM, 1=DENM)</td>
</tr>
<tr>
<td>8</td>
<td>generationTime</td>
</tr>
<tr>
<td>4</td>
<td>StationId</td>
</tr>
<tr>
<td>1</td>
<td>StationCharacteristics</td>
</tr>
<tr>
<td></td>
<td>mobileITSStation</td>
</tr>
<tr>
<td></td>
<td>privateITSStation</td>
</tr>
<tr>
<td></td>
<td>physicalRelevantITSStation</td>
</tr>
<tr>
<td>8+8+4</td>
<td>ReferencePosition</td>
</tr>
<tr>
<td></td>
<td>Longitude/Longitude/Elevation</td>
</tr>
<tr>
<td>4</td>
<td>Heading</td>
</tr>
<tr>
<td>32+4</td>
<td>Streetname/RoadSegment ID</td>
</tr>
<tr>
<td>1</td>
<td>Position/Heading Confidence</td>
</tr>
<tr>
<td>1</td>
<td>CamParameters</td>
</tr>
<tr>
<td></td>
<td>vehicleCommonParameters</td>
</tr>
<tr>
<td></td>
<td>vehicleType</td>
</tr>
<tr>
<td></td>
<td>Length/Width</td>
</tr>
<tr>
<td></td>
<td>Speed</td>
</tr>
<tr>
<td></td>
<td>Acceleration</td>
</tr>
<tr>
<td></td>
<td>AccelerationControl (break, throttle, ACC)</td>
</tr>
<tr>
<td></td>
<td>exteriorLights</td>
</tr>
<tr>
<td></td>
<td>Occupancy</td>
</tr>
<tr>
<td></td>
<td>crashStatus/dangerousGoods</td>
</tr>
</tbody>
</table>
ETSI ITS G5

- Decentralized Environmental Notification Message (DENM)
  - Event triggered (e.g., by vehicle sensors)
    - Hard braking
    - Accident
    - Tail end of jam
    - Construction work
    - Collision imminent
    - Low visibility, high wind, icy road, ...
  - Messages have (tight) local scope, relay based on
    - Area (defined by circle/ellipse/rectangle)
    - Road topology
    - Driving direction
## DENM format (excerpt)

<table>
<thead>
<tr>
<th>Length[byte]</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>messageId (0=CAM, 1=DENM)</td>
<td>Who sent this?</td>
</tr>
<tr>
<td>6</td>
<td>generationTime</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Management</td>
<td>Is this an update on a situation?</td>
</tr>
<tr>
<td>2</td>
<td>Data Version</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Frequency</td>
<td>When can I expect an update?</td>
</tr>
<tr>
<td>1</td>
<td>Reliability</td>
<td>Should I trust a single notification?</td>
</tr>
<tr>
<td>1</td>
<td>IsNegation</td>
<td>Does this cancel an earlier notification?</td>
</tr>
<tr>
<td>1</td>
<td>CauseCode</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>SubCauseCode</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Severity</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>LocationContainer</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Situation_Latitude</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Situation_Longitude</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Situation_Altitude</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Accuracy</td>
<td></td>
</tr>
<tr>
<td>N-40</td>
<td>Relevance Area</td>
<td></td>
</tr>
</tbody>
</table>
ETSI ITS G5

- Service Announcement
  - Message on Control Channel to advertise services offered on Service Channels
    - Channel number
    - Type of service
    - ...

- Similar to WAVE Service Announcement (WSA)

- Receiver can tune (its second radio) to advertised channel
ETSI ITS G5

- Security and privacy
  - No published specification (yet)
    - Kerberos or WAVE-like PKI
    - Restrict participation to authorized vehicles
    - Sign messages
    - Limit V2I and I2V traffic where possible

- Use pseudonyms to protect privacy
  - Use base identity (in permanent storage) to authenticate with infrastructure
  - Infrastructure generates pseudonym for vehicle
ETSI ITS G5: Analysis and Problems

- Oscillating channel load (both local and global!)
  - ...caused by channel access being too restrictive (standard parameters)

ETSI ITS G5: Analysis and Problems

- Latencies
  - Choosing minimum packet intervals (TRC) too high can introduce high latencies

---

ETSI ITS G5: Analysis and Problems

- Update frequency
  - Standard parameters are too restrictive
  - Channel resources are not used optimally

Main Takeaways

- Broadcast Media
  - TMC, TPEG
- UMTS
  - Channels, Pros / Cons
- DSRC/WAVE lower layers
  - 802.11p vs. old 802.11: commonalities and differences
  - HCF (EDCA QoS)
- IEEE 1609.* upper layers
  - Channel management
  - Security / Certificates
- ETSI ITS G5
  - Channel management
  - DCC: Decentralized Congestion Control
  - Message types
  - Commonalities and differences wrt. IEEE 1609.*
Broadcast, Geocast, Routing
Routing

- Classical approaches to routing
  - Distance Vector Routing
    - Nodes keep vector of known destinations, store distance and next hop
    - Ex: DSDV
  
- Link State Routing
  - Nodes keep track of all links in network
  - Pro: fast and guaranteed convergence
  - Con: high overhead
  - Ex: OLSR
Routing

- Classical approaches to routing (II)
  - Reactive (on demand) routing
    - Routes established when needed
    - Routing messages only exchanged if (or while) user data is exchanged
    - Unused routes expire
    - Ex: AODV, DYMO
  - Proactive (table driven) routing
    - Routes are established and maintained continuously
    - No route setup delay when data needs to be sent
    - High overhead
    - Ex: OLSR, DSDV
Routing

- Classical approaches to routing (III)
  - Hop-by-Hop Routing
    - Each packet contains destination address
    - During routing, each hop chooses best next hop
    - Ex: AODV
  - Source Routing
    - Each packet contains complete route to destination
    - During routing, nodes rely on this information
    - Ex: DSR
Routing

- Georouting
  - Primary metrics: position / distance to destination
  - Requires node positions to be known (at least for the destination)
  - Two operation modes (typ.):
    - Greedy mode: choose next hop according to max progress
    - Recovery mode: escape dead ends (local maxima)
  - Must ensure that message never gets lost
Routing

- Georouting: CBF
  - „Contention Based Forwarding“
  - Reduction (or complete avoidance) of duplicates

- Outline
  - Given: position of message destination, position of last hop
  - Do not forward message immediately, but wait for time $T$
  - Choose wait time $T$ according to suitability of node
  - Do not forward message if another forward was overheard

- Problem
  - Potential forwarders must be able to overhear each others’ transmissions

Routing

- **Georouting: CBF**
  - Potential forwarders are contained in Reuleaux triangle (1) (use estimated communication range for thickness of triangle)
  - Waiting time is $T = 1 - P$
    $$P(f, z, n) = \max \left\{ 0, \frac{\text{dist}(f, z) - \text{dist}(n, z)}{r_{\text{radio}}} \right\}$$
  - (z: destination, f: last hop forwarder)
  - If last hop overhears no node forwarding the message, message is re-sent for nodes in (2), then (3)

Routing

- Reflection on classical routing approaches
  - Q: Can (classical) routing work in VANETs?
  - A: Only in some cases.
  - Commonly need multicast communication, low load, low delay
  - Additional challenges and opportunities: network partitioning, dynamic topology, complex mobility, ...

Flooding

- Flooding (Multi-Hop Broadcast)
  - Simplest protocol: „Smart Flooding“:

- Problem: Broadcast Storm
  - Superfluous re-broadcasts overload channel
Flooding

- Consequences of a broadcast storm
  - Interference → impact on other systems
  - Collision → impact on other users
  - Contention → impact on other applications
Flooding

- Solving the broadcast storm problem

- Classical approaches
  - Lightweight solutions (e.g., probabilistic flooding)
  - Exchange of neighbor information, cost/benefit estimations
  - Topology creation and maintenance (Cluster, Cord, Tree, ...)

- Drawbacks
  - Blind guessing (or scenario dependent parameterization)
  - Additional control message overhead
  - Continuous maintenance of topology
Flooding

- **VANET specific solution: Broadcast Suppression**
  - Needs no neighbor information
  - Needs no control messages
  - Maximizes distance per hop
  - Minimizes packet loss

- **Approach**
  - Node receives message, estimates distance to sender
  - Selectively suppresses re-broadcast of message
  - Alternatives
    - weighted p-persistence
    - slotted 1-persistence
    - slotted p-persistence

Flooding

- Broadcast Suppression
  - Estimate distance to sender as $0 \leq \rho_{ij} \leq 1$ based on $R$ ("approximate transmission radius")
  - Variant 1: GPS based
    \[
    \rho_{ij} = \begin{cases} 
    0 & \text{if } D_{ij} < 0 \\
    \frac{D_{ij}}{R} & \text{if } 0 \leq D_{ij} < R \\
    1 & \text{otherwise}
    \end{cases}
    \]
  - Variant 2: RSS based
    \[
    \rho_{ij} = \begin{cases} 
    0 & \text{if } \text{RSS}_x \geq \text{RSS}_\text{max} \\
    \frac{\text{RSS}_\text{max} - \text{RSS}_x}{\text{RSS}_\text{max} - \text{RSS}_\text{min}} & \text{if } \text{RSS}_\text{min} \leq \text{RSS}_x \leq \text{RSS}_\text{max} \\
    1 & \text{otherwise}
    \end{cases}
    \]
Flooding

- Broadcast Suppression
  - Weighted $p$-persistence
    - Probabilistic flooding with variable $p_{ij}$ for re-broadcast
    - Thus, higher probability for larger distance per hop
Flooding

- Broadcast Suppression
  - Weighted p-persistence
    - Wait WAIT\_TIME (e.g., 2 ms)
    - choose $p = \min(p_{ij}) = \min(p_{ij})$ of all received packets
      (probability for re-broadcast of packet)
    - Ensure that at least one neighbor has re-broadcast packet
Flooding

- **Broadcast Suppression**
  - **Slotted 1-persistence**
    - Suppression based on waiting and overhearing
    - Divide length of road into slots
    - More distant slots send sooner
    - Closer slots send later (or if more distant slots did not re-broadcast)
    - Thus, higher probability to transmit over longer distance

\[ p_{ij} \]

\[ t=0 \quad t=\tau \quad t=2\tau \quad t=3\tau \]
Flooding

- Broadcast Suppression
  - Slotted 1-persistence
    - Divide “communication range“ into $N_s$ slots of length $\tau$
    - Nodes wait before re-broadcast, waiting time $T_{ij}=\tau \times \lceil N_s(1-\rho_{ij}) \rceil$
    - Duplicate elimination takes care of suppression of broadcasts
Flooding

- Broadcast Suppression
  - Slotted $p$-persistence
    - Cf. slotted 1-persistence
    - Fixed forwarding probability $p$ (instead of 1)
Flooding

- Broadcast Suppression
  - Slotted p-persistence
    - Wait for $T_{ij}$ (instead of fixed WAIT\_TIME)
    - Use probability $p$ (instead of 1)
    - Ensure that at least one neighbor has re-broadcast the packet by waiting for $\delta' > \max(T_{ij})$
Flooding

- Broadcast Suppression
  - Solves Broadcast Storm Problem
  - Maximizes distance per hop
  - Minimizes packet loss
  - But: Much higher per-message delay
Remaining problems

- Temporary network fragmentation

- Undirected message dissemination
Flooding + X

- **DV-CAST**
  - Idea: detect current scenario, switch between protocols
  - Check for fragmented network
    - Network connected ➞ perform broadcast suppression
    - Network fragmented ➞ perform Store-Carry-Forward

Flooding + X

- **DV-CAST: Mechanism**
  - Nodes periodically send *Hello* beacons containing position, speed
  - Nodes maintain 3 neighbor tables
    - Same direction, ahead
    - Same direction, driving behind
    - Opposite direction
  - Messages contain source position and Region of Interest (ROI)

- For each message received, evaluate 3 Flags:
  - Destination Flag (DFlg):
    Vehicle in ROI, approaching source
  - Message Direction Connectivity (MDC):
    ∃ neighbor driving in same direction, further away from source
  - Opposite Direction Connectivity (ODC):
    ∃ neighbor driving in opposite direction
Flooding + X

- **DV-CAST**
  - **Algorithm:**

![Flowchart showing DV-CAST algorithm](image)

---

Flooding + X

- DV-CAST
  - Decision matrix:

<table>
<thead>
<tr>
<th>MDC</th>
<th>ODC</th>
<th>DF lg</th>
<th>Derived Scenario</th>
<th>Actions Taken by DV-CAST Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>×</td>
<td>1</td>
<td>Well Connected</td>
<td>Broadcast Suppression</td>
</tr>
<tr>
<td>1</td>
<td>×</td>
<td>0</td>
<td>Well Connected</td>
<td>Help relay the packet by doing broadcast suppression</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Sparsely Connected</td>
<td>Rebroadcast and assume that the ODN will help relay or rebroadcast</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Sparsely Connected</td>
<td>Rebroadcast and help carry &amp; forward the packet to the first new neighbor in the opposite direction or in the message direction encountered</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>×</td>
<td>TotallyDisconnected</td>
<td>Wait and forward the packet to the first neighbor in the opposite direction or in the message direction encountered.</td>
</tr>
</tbody>
</table>
Flooding + X

- **DV-CAST**
  - Simulation results show that (on freeways with low to medium node densities) DV-CAST beats simple flooding in terms of broadcast success rate and distance covered
Intermediate Summary

- Remaining problems
  - Temporary network fragmentation (solved)

- Undirected message dissemination
Geocast

- **TO-GO**
  - „Topology-Assisted Geo-Opportunistic Routing“
  - Nodes periodically send *Hello* beacons; Contents:
    - Number of neighbors
    - Bloom filter of neighbor IDs
    - IDs of neighbors furthest down the road/roads
  - Thus, nodes know about all 2-hop neighbors

Geocast

- Bloom Filter
  - Idea:
    - Bloom filter is a bit field $X$
    - Hash functions $h_1$ to $h_k$ map input data $x \rightarrow$ one bit (each) in $X$
    - Insertion of $x$: Set $X[h_i(x)] \leftarrow 1 \quad \forall i \in [1..k]$
    - Test for $x \in X$: Check $X[h_i(x)] \not= 1 \quad \forall i \in [1..k]$
  - Probabilistic test for “$x \in X$”
    - Possible results: no / maybe (→ chance of false positives)
  - Allows for very compact representation of $X$

Geocast

- TO-GO
  - Step 1: Find best next hop (Target Node, T)
    - Find N: Furthest neighbor towards destination
    - Find J: Furthest neighbor towards destination, currently on junction
    - Find N_j: Furthest neighbor towards destination, as seen by J
    - if N, N_j are on the same road (and running in greedy mode), pick N else, pick J

![Image of vehicles on a road with nodes N, J, and N_j indicated]
Geocast

- **TO-GO**
  - **Step 2: Find Forwarding Set (FS)**
    - Nodes in the FS will compete for relaying of the message
    - Only one node in FS should relay thus, all nodes in FS must hear each other
    - Finding optimal solution is *NP complete*
    - TO-GO uses approximation:
      - Bloom filter entries indicate who can hear whom
      - Given the target node $T$, find its neighbor $M$ with the maximum number of neighbors
      - Include all those neighbors in FS, which
        - can hear $M$, and
        - are heard by $M$, and
        - are heard by all current members of FS
Geocast

- TO-GO
  - Step 3: Multicast message to all nodes in FS
    - Nodes in the FS compete for relaying of the message
    - Ensure maximum progress within FS
    - Delay re-broadcast by $t$
    - Suppress re-broadcast if another nodes forwards within $t$
    - $t = \tau \times d_T / d_{\text{max}}$
      with:
      - $\tau$: Maximum delay per hop
      - $d_T$: Distance to Target Node
      - $d_{\text{max}}$: Distance from last hop to Target Node
Intermediate Summary

- Remaining problems
  - Temporary network fragmentation (solved)
  - Undirected message dissemination (solved)
Scalability

- Do the presented approaches scale?
- Analytical evaluation [1]:
  - Capacity of wireless channel is limited
  - Amount of information transported across any (arbitrary) border must be upper-bounded

Scalability

- Solution?
  - Define maximum dissemination range of any information
  - Reduce update frequency with increasing distance
  - Aggregate information as distance increases

- Pre-condition for scalability of dissemination approach?
  - Used bandwidth reduces as distance to source increases
  - Upper bound: $1 / d^2$

```
used bandwidth B

\[ \propto 1 / d^2 \]
```
Main Takeaways

- Classic information dissemination
  - Distance vs. link-state
  - Reactive vs. proactive
  - Hop-by-hop vs. source routing
  - Geo-routing (CBF)

- Examples of VANET-centric information dissemination
  - Flooding (Weighted/Slotted 1/p-Persistence)
  - Fragmentation (DV-Cast)
  - Directedness (To-Go)

- Scalability
Beaconing and TIS
Traffic Information Systems
You are here:

Vehicle-to-X

Non-Safety

Safety

Comfort

Traffic Information Systems

Situation Awareness

Warning Messages

Contextual Information

Entertainment

Optimal Speed Advisory

Congestion, Accident Information

Adaptive Cruise Control

Blind Spot Warning

Traffic Light Violation

Electronic Brake Light

Motivation

- **Goals:**
  - Increase comfort
  - Reduce (or avoid) traffic jams
  - Relieve driver
  - Decrease travel times
  - Smooth traffic flow
  - Decrease Emissions (?)
    - CO$_2$, NO$_X$, Noise, ...

- **Recall: Traditional TIS**
  - Traffic Information Center (TIC) collects data, creates bulletins
  - Bulletins are disseminated via RDS-TMC or TPEG
  - Navigation assistant reacts by re-routing
Motivation

- Problem of traditional TIS:
  - High delay
    - „Jams reported no earlier than they dissolved on their own“
  - Low data rate
    - Can only send few, most important bulletins
  - Low reliability
    - Centralized system
    - Human factor
    - Radio reception / Internet connection
Motivation

▪ Goal: use vehicles as both information sink and source
  ▪ Vehicles measure road conditions, travel time, ...
  ▪ ... disseminate information to neighbors
  ▪ ... help relay information further
  ▪ and can promptly react to new information

▪ Typical requirements
  ▪ Vehicle-to-vehicle communication
    ▪ Technology? Availability?
  ▪ GPS receiver
    ▪ Precision?
  ▪ Digital road map
    ▪ Same data basis?
Motivation

- Data
  - Must be kept current
  - High spatial resolution
  - But: no random access

- Users
  - Distributed in wide area
  - Mobile
  - Data source and sink
PeerTIS

- PeerTIS
  - Based on cellular communication (UMTS)
  - Paradigm: Peer-to-Peer

- Benefits
  - UMTS
    - Established technology
    - Infrastructure already present
  - Peer-to-Peer
    - Resources scale with number of participants
    - Decentralized System $\rightarrow$ Independence, robustness, ...

PeerTIS

- PeerTIS
  - Based on cellular communication (UMTS)
  - Paradigm: Peer-to-Peer

- Drawbacks
  - UMTS
    - High latency, high packet loss
    - Cost?
  - Peer-to-Peer
    - Coordination
    - Security?

PeerTIS

- Stored (and exchanged) information
  - Road
  - Mean speed
  - Time
  - User ID

- Properties
  - High spatial resolution
  - High precision
  - Short query interval
PeerTIS

- Data
  - Structured
  - Geo-referenced
- Typical use case
  - Join PeerTIS network
  - Calculate naïve route
    - ...and alternatives
  - Query data
    - segment by segment
  - Pick best route, start driving
  - Periodically check for updates
- Observation
  - Query pattern of segments not random, but predictable
PeerTIS

- Peer-to-Peer
  - All users treated equally

- Technique
  - Unstructured overlay networks
  - Structured paradigms

- Distributed Hash Tables (DHTs)
  - Create hash value of information to store
  - Use as index for information storage and retrieval
  - Each user is assigned part of the key space, stores information that has hash value in assigned range
PeerTIS

- Joining a PeerTIS network
  - Split any node’s key space in two
  - Take over half of data, assigned range
PeerTIS

- Drawback of unmodified DHT algorithm:
  - Hashing leads to random distribution of data
  - Leads to long query times, high load, ...
PeerTIS

- Improved storage in PeerTIS:
  - Content Addressable Network (CAN), but: no hashing
  - Physical neighbors responsible for neighboring areas
  - Thus: faster lookup of information close by
  - Optimization: hop-by-hop forward of query to all hosts; results appended on reverse path
PeerTIS

- Additional optimization
  - Exploit time correlation of queries (initial query, then periodic updates):
    - Store address of all nodes that answered initial query
    - Try getting updates directly from these nodes
PeerTIS

- Third optimization
  - Exploit spatial distribution of nodes
  - Do not assign random geographic area to nodes.
  - Instead, chose area close to their start of route

- Result: more resources allocated to areas of high traffic density
PeerTIS

- Evaluation:
  - Impact on network load tolerable for low to medium density
  - Even distribution of network load
  - Speed superior to naïve P2P algorithm
PeerTIS

- Possible improvements
  - Subscribe to (bigger) changes in data
  - Multicast distribution of data

- Open Questions
  - Replication of data?
  - Security?
  - Going infrastructure-less?
  - Heterogeneous map data?
SOTIS

- Self-Organizing Traffic Information System (SOTIS)
  - Each node maintains local knowledge base
  - Periodically sends single-hop broadcasts with information (Beacon)
    - Weather information gets sent with longer interval
    - Accident messages get sent with shorter interval
  - Integrates received information with knowledge base

- Techniques
  - WiFi (IEEE 802.11) in Ad-hoc-Mode
  - SODAD (Segment-oriented data abstraction and dissemination)

SOTIS

- Evaluation
  - Speed of information dissemination depends on traffic density and market penetration, varies in 120 .. 600 km/h
SOTIS

- Open issues
  1. Infrastructure-less operation: needs high marked penetration
  2. Required/tolerable beacon interval highly dependent on scenario
  3. Design needs dedicated channel capacity

- Real networks are heterogeneous
  1. Roadside infrastructure present vs. absent
  2. Freeway scenario vs. inner city
  3. Own protocol ↔ other, future, and legacy protocols

- How to do better?
  1. Dynamically incorporate optional infrastructure
  2. Dynamically adapt beacon interval
  3. Dynamically use all free(!) channel capacity
Adaptive Traffic Beacon (ATB)

- Adaptive use of infrastructure
  - Independent operation
  - Road Side Units
  - Traffic Information Center uplink

Adaptive Traffic Beacon (ATB)

- Adaptive selection of beacon interval $\Delta I$
  - Consider message utility $P$
  - Consider channel quality $C$

- Choose interval from range $I_{\text{min}}$ to $I_{\text{max}}$
  - Use factor $w_I$ to increase weight of $C$ (ex. $w_I=0.75$)
  - $\Delta I = ((1 - w_I) \times P^2 + (w_I \times C^2)) \times (I_{\text{max}} - I_{\text{min}}) + I_{\text{min}}$
Adaptive Traffic Beacon (ATB)

- Adaptive selection of beacon interval $\Delta I$
  
  - Calculation of message utility $P$ based on metrics of (ex.)
    - $A$: age of information
    - $D_e$: distance to source of information
    - $D_r$: distance to closest Road Side Unit (RSU)
    - $B$: ratio of beacon contents received from Road Side Unit (RSU)

  - Calculation of channel quality $C$ based on metrics of (ex.)
    - $N$: (estimated) number of neighbors ($\rightarrow future$)
    - $S$: (observed) signal-to-noise ratio ($\rightarrow present$)
    - $K$: (measured) collisions on channel ($\rightarrow past$)

  $P = \frac{A + D_e + D_r}{3} \times B$ \\
  $C = \frac{N + w_c(S + K) / 2}{1 + w_c}$
Envisioned Scenario

- Highly dynamic network
Simulative Performance Analysis

- Comparison with static beaconing
  - Static beaconing gets better as beaconing frequency increases
    - but channel load increases sharply
  - ATB performs as good as static beaconing at highest frequencies
    - at the same time keeps load lower than at lowest frequency
Main Takeaways

• Traffic Information System (TIS)
  ▪ Goals
  ▪ Principles
• PeerTIS
  ▪ Use of infrastructure
  ▪ DHT, CAN concepts
• SOTIS
  ▪ Beacons
  ▪ Knowledge bases
• ATB
  ▪ Adaptivity
Privacy
Motivation

- Aspects of privacy
  - Privacy of location
    - Where is the target individual?
    - Where was the target individual at a given time?
    - Where will the target individual likely be at a given time?
  - Privacy of interests
    - Hobbies, services, news sources, ...
  - Privacy of social standing
    - Job, income, debt, home, contractual obligations, ...
  - Privacy of social network
    - Family, friends, friends-of-friends, acquaintances, ...
Motivation

- Who (and how powerful) is the attacker?
  - Government
    - Can mandate access to all information
  - System operator
    - Can obtain access to all information
  - Service provider
    - Has access to all information
  - Application developer
    - Can access device

- Company
  - High coverage using wide area deployments (e.g., WiFi-APs)

- Organization
  - Good coverage via cooperations

- Private individual
  - Must target individual areas
Motivation

- **Level of risk**

  - Exact position, Tracking
  - City, City district
  - State, Conurbation

<table>
<thead>
<tr>
<th>Power of attacker</th>
<th>System operator</th>
<th>Application provider</th>
<th>Private individual</th>
</tr>
</thead>
<tbody>
<tr>
<td>negligible risk</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>extreme risk</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Motivation

- Tracking
  - Vehicles periodically send Hello Beacons, received by observers
  - Beacons contain uniquely-identifying information
Motivation

- Safety and Security
  - Authentication, Authorization, Accounting, Auditing, ...
  - ...often need unique identification of peers

- Conflicts with users’ privacy
  - Very long life-cycle of vehicles
  - Information can be aggregated over very long time periods
  - Correlating identity ⇔ location allows for tracking

Consequences

- Examples:
  
  - Police records movement traces
    - Citation for breaking the speed limit
  
  - Employer identifies parking vehicles
    - Keeps records of when employees come and go
  
  - Insurance company buys movement traces
    - Denies contract renewal because of too many trips to hospital
Privacy

- Need to protect against
  - Identification of vehicle
  - Re-identification of vehicle

- Identifying properties
  - Characteristic properties of application, system, radio
    - Timing, packet size, RF-fingerprint, ...
  - Plain identifiers
    - MAC address, IP address, Login, ...
    - Certificate (necessary for participation!)

- Absolute Anonymity?
  - Made impossible by most protocols and/or use cases

Anonymity

Anonymity is...

“the state of being not identifiable within a set of subjects, the anonymity set”

(Pfitzmann/Hansen)

Pseudonymity

- Communication using pseudonyms
  - Sign messages using pseudonymous certificates
  - Receiver can check if signed by trusted CA
  - Base identity never revealed to other vehicles
- Revocation of Pseudonyms
  - Dissemination of Certificate Revocation List (CRL) via Internet, RSUs, or Car-to-Car
  - Open questions: availability, scalability (speed, size of CRL)
  - CA knows mapping from base identity ⇔ pseudonym; can revoke all related pseudonyms
Pseudonymity

- Obtaining pseudonymous certificates
  1. Certificate authority (CA) sends base identities to manufacturer
  2. Manufacturer installs one base identity each in new cars
  3. When cars start operating, they create pseudonyms and have them signed by CA (using their base identity)
Pseudonymity

- Limits of pseudonymity
  1. Does not (per se) prevent re-identification
     - Re-identification can still reveal identity of user
     - Ex from [1]:
       - Knowledge of census tract (1500 people each) for home and workplace
       - Reduces anonymity set to $\leq 5$ people (for 25% of population)
       - Reduces anonymity set to $\leq 2$ people (for 7% of population)
  2. Operator knows mapping from pseudonym $\Leftrightarrow$ identity

Pseudonymity

- Goal
  - Vehicles periodically send Hello Beacons, received by observers
  - Now: use of many pseudonymous identifiers to prevent tracking
Pseudonym Pools

- How to prevent re-identification?

- Pseudonym pools
  - Create pool of pseudonyms (instead of single one)
  - Switch between different pseudonyms

- Validity
  - No restrictions
  - Spatial restrictions
  - Temporal restrictions

- Switching strategies
  - How to enhance anonymity?
  - Tradeoff between safety and privacy
    - MUST have static identifiers for safety, MUST NOT have for privacy
Pseudonym Pools

- Pseudonym selection strategies:
  - Fully random
  - Periodic
    - Switch to another pseudonym every x seconds
  - Geographic
    - Switch to another pseudonym depending on region
  - Context sensitive
    - Switch when confusion of (potential) attackers is maximal
      - Wait for high number of neighbors
      - Wait for neighbors with similar position, angle, speed, ...


Pseudonym Pools

- Enhanced strategies:

- Random Silent Periods
  - After switching pseudonyms, cease transmitting for random time

- Group strategic approaches
  - Coordinate silent periods among neighbors
  - Coordinate pseudonym changes among neighbors
  - Elect group leader, establish encrypted tunnel, use as proxy


Pseudonym Pools

- How to prevent tracking by operator?
- Destroy operator’s mapping of pseudonym(s) ⇔ identity:
  - Exchange of pseudonyms among vehicles
    - Hand over key material to random neighbor
    - Take over pseudonym of neighbor
- Blind signatures
  - Encrypt 100 pseudonyms, transmit to CA for signing
  - CA requests, checks contents of 80
  - Blindly signs remaining 20 (if successful)


Quantifying Privacy

- How to compare different strategies?
  - Need a way to quantify privacy

- Next slides: three selected privacy metrics
  - Maximum Tracking Time
  - Anonymity Set Size
  - Entropy

- Still: impact of strategies hard to gauge
  - Need to know power/methods of attacker
  - Also: scenario dependent (traffic density, network topology, ...)

Vehicular Networking 372
Quantifying Privacy

- Dead Reckoning
  - Simplest algorithm to track cars

- Algorithm
  1. Predict new position based on heading, speed
  2. Consider maximum change in speed, maximum turning angle
  3. Remove candidates with positions that differ too much
Quantifying Privacy

- **Privacy Metric: Maximum Tracking Time**
  - How long was an attacker able to follow a vehicle’s path?
  - The shorter, the better

- **Algorithm**
  - For every vehicle, keep track of current position, start timer
  - Correlate position samples over time (e.g., via dead reckoning)
  - If correlation impossible (or leads to false result), stop timer

- **Drawbacks of Max Tracking Time**
  - Distinguishing correct/false results needs *oracle*
  - Prediction can yield more than one potential position, thus:
    Tendency to underestimate maximum tracking time, i.e., to overestimate level of privacy
Quantifying Privacy

- Privacy Metric: Anonymity Set Size
  - Recall: anonymity set ⇔ all nodes indistinguishable from target
    - Target T located in one of vehicles $a_i$
  - Metric: cardinality of anonymity set
    - $A_T = \{a_1, a_2, \ldots, a_i, \ldots, a_n\}$
    - $|A_T| = n$
  - Indication of degree of uncertainty
Quantifying Privacy

- Anonymity Set Size: Problem
  - Not every vehicle \( a_i \) is equally probable, thus:
    Tendency (of anonymity set size) to overestimate level of privacy

\[
|A_T| = 2 \quad =? \quad |A_T| = 2
\]
Quantifying Privacy

- Towards a solution:
- Consider *probabilities* of members in anonymity set
  - Let $p_i$: probability of $i$-th node of anonymity set being the target individual $T$
  - Let sum of all $p_i$ be 1
  - $A_T = \{a_1, a_2, \ldots, a_i, \ldots, a_n\}$
  - $|A_T| = n$
  - $\sum_{i=1}^{|A_T|} p_i = 1$
Quantifying Privacy

- Privacy Metric: (information theoretic) entropy
  - Degree of uncertainty for mapping node ⇔ target
  - Calculate entropy $H_p$ as:
    - $H_p = - \sum_{i=1}^{|A|} p_i \times \log_2 p_i$
    - $H_{p,max} = - \sum_{i=1}^{|A|} p_i \times \log_2 p_i = \log_2 |A| \quad \text{if } \forall p_i = \frac{1}{|A|}$
    - i.e., maximum entropy if all probabilities $p_i$ equal:

Quantifying Privacy

- Benefit of using entropy as privacy metric:

- Example 1:
  - Equal probability of mappings in anonymity set
  - Let $p_0 = 50\%$; $p_1 = 50\%$
  - $H = -(0.50 \log_2 0.50 + 0.50 \log_2 0.50) = 1$

- Example 2:
  - (Very) unbalanced mappings
  - Let $p_0 = 99\%$; $p_1 = 1\%$
  - $H = -(0.99 \log_2 0.99 + 0.01 \log_2 0.01) \approx 0.08$

- Very low level of anonymity (even though equal set size)
Main Takeaways

- Threats to privacy
- Definition of anonymity

- Pseudonyms
  - Pseudonymous certificates
  - Selection strategies

- Quantifying privacy
  - Maximum tracking time
  - Anonymity set size
  - (Information theoretic) entropy
  - Pros / Cons
Performance Evaluation

...how to tell what works and what does not
Approaches to Performance Evaluation

- **Field Operational Tests**
  - Highest degree of realism
  - No in-depth investigations of network behavior
  - Non-suppressible side effects
  - Limited extrapolation from field operational tests
    - Some 100 vehicles $\Leftrightarrow$ 2% market penetration? (or 10%, or 100%)

- **Analytical evaluation**
  - Closed-form description allows for far-reaching conclusions
  - May need to oversimplify complex systems

- **Simulation**
  - Can serve as middle ground
Requirements for Simulation

- **Models**
  - Network protocol layers
  - Radio propagation
  - Node mobility
  - Model of approach to be investigated (e.g., flooding)

- **Scenarios**
  - Road geometry, traffic lights, meta information
  - Normal traffic pattern
  - Scenario of use case to be investigated (e.g., accident)

- **Metrics**
  - Network traffic metrics (delay, load, ...)
  - Road traffic metrics (travel time, stopping time, emissions, ...)
  - Metric of use case to be investigated (e.g., time until jam resolved)
Modeling Network Protocols

- Dedicated simulation tools
  - Discrete Event Simulation (DES) kernel
  - Manages queue of events (e.g., “an IP fragment was received”)
  - Delivers events to simulation models

- Model libraries
  - Simulate components’ reaction to events
  - E.g., HTTP server, TCP state machine, radio channel, human, ...
  - “when enough IP fragments received ⇒ tell TCP: packet received”

<table>
<thead>
<tr>
<th>Engine</th>
<th>Language</th>
<th>Library</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>OMNeT++</td>
<td>C++</td>
<td>MiXiM</td>
<td>C++</td>
</tr>
<tr>
<td>ns-2 / ns-3</td>
<td>C++</td>
<td>ns-2 / ns-3</td>
<td>Objective Tcl / Python</td>
</tr>
<tr>
<td>JiST</td>
<td>Java</td>
<td>SWANS</td>
<td>Java</td>
</tr>
</tbody>
</table>
Modeling Radio Channel

- Simple model: unit disk
  - Fixed radio “range”
  - Node within range ⇔ packet received

- Enhanced models:
  - For each packet, consider
    - Signal strength
    - Interference (other radios)
    - Noise (e.g., thermal noise)
  - Calculate “signal to noise and interference ratio” (SNIR)
  - Derive packet error rate (PER)
Modeling Radio Propagation

- **Signal attenuation**
  - Received power depends on
    - transmitted power,
    - antenna gains
    - loss effects
  - \[ P_r [\text{dBm}] = P_t [\text{dBm}] + G_t [\text{dB}] + G_r [\text{dB}] - \sum L_x [\text{dB}] \]

- **Free space path loss**
  - \[ L_{\text{freespace}} [\text{dB}] = 20 \log \left( 4\pi \frac{d}{\lambda} \right) \]

- **Empirical free space path loss**
  - \[ L_{\text{freespace,emp}} [\text{dB}] = 10 \log \left( 4\pi \frac{d}{\lambda} \right)^\alpha \]
Modeling Radio Propagation

- Two Ray Interference path loss

\[ L_{\text{tri}}[\text{dB}] = 20 \log \left( \frac{4 \pi d}{\lambda} \left| 1 + \Gamma \right| e^{i \varphi} \right)^{-1}, \]  
substituting

\[ \varphi = 2 \pi \frac{d_{\text{los}} - d_{\text{ref}}}{\lambda}, \quad \Gamma = \frac{\sin \theta_i - \sqrt{\epsilon_r - \cos^2 \theta_i}}{\sin \theta_i + \sqrt{\epsilon_r - \cos^2 \theta_i}}, \]

\[ d_{\text{los}} = \sqrt{d^2 + (h_t - h_r)^2}, \quad d_{\text{ref}} = \sqrt{d^2 + (h_t + h_r)^2}, \]

\[ \sin \theta_i = (h_t + h_r) / d_{\text{ref}}, \quad \cos \theta_i = d / d_{\text{ref}}. \]

Modeling Radio Propagation

- Comparison: Two Ray Interference vs. Free Space

Modeling Radio Propagation

- Lognormal shadowing
  - Lognormal distribution of losses via random process
  - $L_{\text{lognorm}}[\text{dB}] \sim \mathcal{N}(0, \sigma^2)$
- Very(!) simple obstacle model
  - Take into account: distance through matter, number of walls
  - $L_{\text{obs}}[\text{dB}] = \beta n + \gamma d_m$
Modeling Mobility

- Traditional approach in network simulation: Random Waypoint (RWP)
  - „pick destination, move there, repeat“

- First adaptation to vehicular movement
  - Add mass, inertia
  - Add restriction to “roads”
  - Add angular restrictions

- Problem
  - Very unrealistic (longitudinal) mobility pattern

Modeling Mobility

- First approach: Replay recorded trace data
  - Use GPS
  - Install in Taxi, Bus, ...
  - Highest degree of realism

- Problems:
  - Invariant scenario
  - No extrapolation
    - To other vehicles (cars, trucks, ...)
    - To more vehicles
    - To fewer vehicles


Modeling Mobility

- Improved approach: Replay artificial trace data
  - Microsimulation of road traffic
  - Pre-computation or live simulation
  - Problem: how to investigate traffic information systems (TIS)?


Modeling Mobility

- Latest approach: Bidirectional coupling
  - Road traffic simulator and network simulator run in parallel, exchange data in both directions
  - Network traffic can influence road traffic


Modeling Road Traffic

- Road traffic microsimulation
  - Ex.: SUMO – Simulation of Urban Mobility
  - Time discrete microsimulation

- Car following models (Krauss, IDM)
- Lane change models
- Road topology
  - Speed limits
  - Traffic lights
  - Access restrictions
  - Turn restrictions
  - ...

Modeling Road Traffic

- Road traffic microsimulation
  - Ex.: PTV VISSIM

- Car following and lane change model
  - Wiedemann (psycho-physiological model)

- High precision modeling
  - Pedestrians
  - Motorbikes

- Comparatively slow simulation, thus limited to small area

Modeling Car Following

- **Krauss car following model**
  - Maximum velocity $v_{max}$ ⇔ safe gap $g_{des}$ ⇔ dawdle factor $\epsilon$
  - $v_{safe} = v \pm \frac{g - g_{des}}{\tau_b + \tau}$
  - $v_{des} = \min\{v_{max}, v + a\Delta t, v_{safe}\}$
  - $v(t + \Delta t) = \max\{0, v_{des} - \eta\}$
  - $\eta = \text{rand}[0, \epsilon a]$

- **Intelligent Driver Model (IDM)**
  - Desired velocity $v_0$ ⇔ safe distance $s^*$
  - $s^* = s_0 + s_1 \sqrt{\frac{v}{v_0} + vT + \frac{v\Delta v}{2\sqrt{ab}}}$
  - $\dot{v} = a \left(1 - \left(\frac{v}{v_0}\right)^\delta - \left(\frac{s^*}{s}\right)^2\right)$

Simulation Frameworks

- Examples of coupled simulation frameworks
  - IDM/MOBIL ⇔ OMNeT++/INET [1]
  - VGSim: VISSIM traces ⇔ ns-2 [2]

- Examples of bidirectionally coupled frameworks
  - Veins: SUMO ⇔ OMNeT++/MiXiM [3]
  - TraNS: SUMO ⇔ ns-2 [4]
  - NCTUns (hand-made simulator) [5]
  - iTETRIS: SUMO ⇔ ns-3
  - VSimRTI: VISSIM ⇔ JiST/SWANS

---

VSimRTI

- “V2X Simulation Runtime Infrastructure”
- Inspired by High Level Architecture (HLA).
- Multiple Simulators connect to VSimRTI
  - Application Simulator
  - Traffic Simulator
  - Communication Simulator
  - Environment Simulator
- Each simulator needs only be extended by a common coupling interface, the “Federate Ambassador”

iTETRIS

- EU FP7 project to create a simulation framework for ETSI ITS
- NS-3, SUMO, and Application Simulator are connected to iCS process
- iCS process takes care of synchronization and control, implements part of the ETSI ITS “Facilities Layer”

Veins

- Oldest and most cited of the three
- Open Source vehicular network simulation \textit{framework}
- Module library for OMNeT++ network simulator
  - Based on well-established simulator, easy to use for research and teaching
  - Easily extensible, re-configurable for new projects
    - Diverse modules of IVC-specific channels and protocol stack complement other module frameworks
- Road traffic simulation as auxiliary part
  - SUMO simulator


Veins

- **OMNeT++**
  - Discrete-Event Simulation (DES) kernel
    - Simulate model’s reaction to queue of events
  - Main use case: network simulation
    - e.g., MANETs, Sensor nodes

- **MiXiM**
  - Model library for OMNeT++ for PHY layer and mobility support
  - Event scheduling
  - Signal propagation
  - SINR / bit error calculation
  - Radio switching
  - ...

Veins

- Coupling OMNeT++ and SUMO
  - Synchronize time steps
  - Exchange commands and status information
Veins

- Traffic Control Interface (TraCI)
  - Generic API
  - Exchange commands via TCP connection
- Simple request-response protocol
  - OMNeT++ sends request for simulator parameters, vehicle position, etc. \(\Rightarrow\) SUMO responds
  - Can also “subscribe” to changes in the simulation \(\Rightarrow\) automatically receive change notifications for vehicle positions, vehicles starting in the simulation, etc.


Simulation Scenarios

- Mobility model consists of two parts
  - Motion constraints
  - Traffic demand
- Motion constraints are...
  - Road topology
  - Speed limits
  - ...
- Traffic demand is...
  - Which cars start where
  - How driver behaves during trip
  - ...

Simulation Scenarios

- Nowadays, motion constraints are easy to obtain
  - Freely available road topology and speed limit information from OpenStreetMap project

- Traffic demand is much harder
  - Often no publicly available information on traffic flows
  - If synthetic traffic demand does not match motion constraints, scenario fails
    - E.g. only few cars using main roads + lots of cars using small roads ⇒ traffic lights not calibrated to this demand ⇒ gridlock
Metrics

- Assumed “benefit” of solutions depends fully on metric
- Ex.: when is it beneficial to take a detour around an accident?
  - Might be beneficial in terms of travel time
  - ...but not beneficial in terms of CO₂ emissions

Main Takeaways

- Approaches to performance evaluation
  - Pros/Cons
- Requirements for simulation
  - Models, Scenarios, Metrics
- Simulation
  - Modeling network traffic
  - Modeling road traffic
- Scenarios
  - What’s in a scenario?
- Metrics
Vehicular Networking