Karlsruher Institut für Technologie

Institut für Technik der Informationsverarbeitung ITIV

Prof. Dr.-Ing. Dr. h.c. Jürgen Becker *(becker@kit.edu)*

Dr.-Ing. Jens Becker *(jens.becker@kit.edu)*

Institut für Technik der Informationsverarbeitung (ITIV)

Communication Systems and Protocols

Session 8: System busses

Clicker Session: Recapitulation

- https://arsnova.eu/mobile/#id/33969518

Recapitulation

- Specifications
  - Dataframe
  - Handshaking
- Error Handling
  - Redundancy
  - CRC

I²C-Bus (Inter-Integrated-Circuit-Bus)

- Serial Bus System
- Multi Master / Multi Slave
- CSMA/CA Arbitration Scheme
- Low Cost Bus System
- …

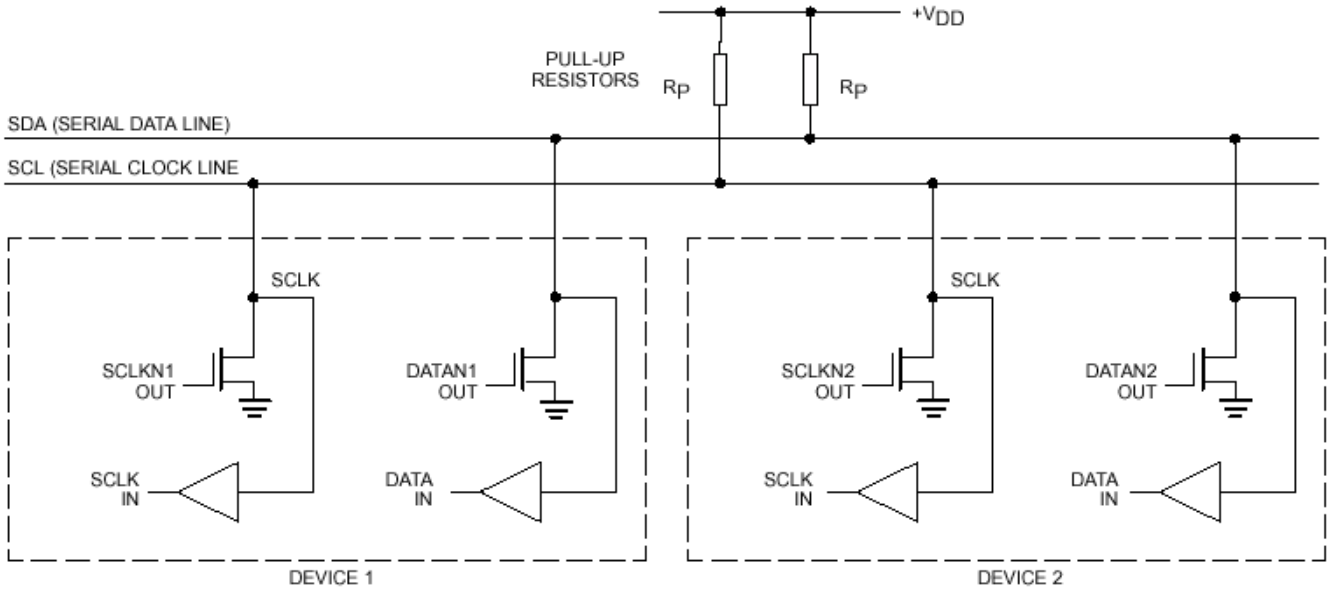
I²C-Bus

- Developed by Philips for interconnecting ICs on printed circuit boards
- Only two wires for data transmission
  - Clock line SCL (serial clock)
  - Data line SDA (serial data)
  - Lower cost and error-prone because of low pin-count
- Multi-Master, Multi-Slave
- Data transmission in packets of 8bit
  - 7bit for addresses → 128 addresses
  - 1bit for toggling between reading/writing
- Transfer rate:
  - 100kbit/s in standard mode
  - 400kbit/s in fast mode
  - 3.4Mbit/s in high-speed mode

Source of I²C images and diagrams: *NXP: UM10204, I2C-bus specification and user manual, Rev. 4 — 13 February 2012*

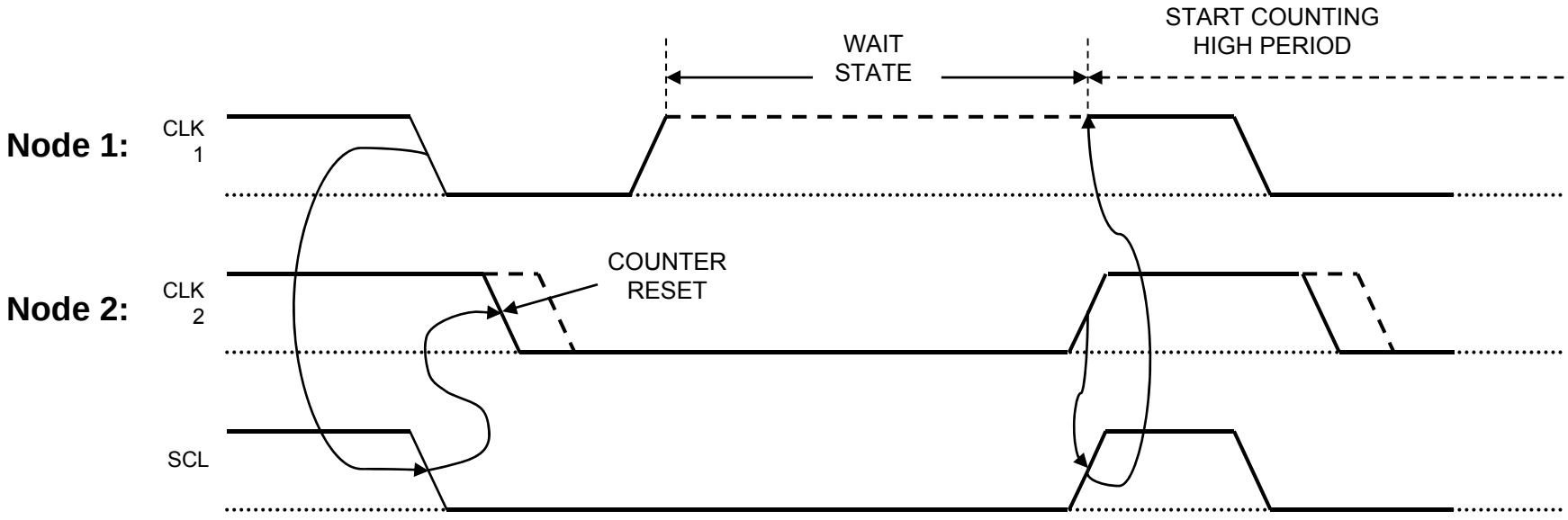
I²C Bus Connection

- Open-Collector outputs for every bus subscriber
  - Wired-AND with dominant ‚0‘ on the bus
  - Pull-Up resistors externally connected
- In idle mode, both SCL and SDA are HIGH
- Concurrent monitoring of the bus at every subscriber



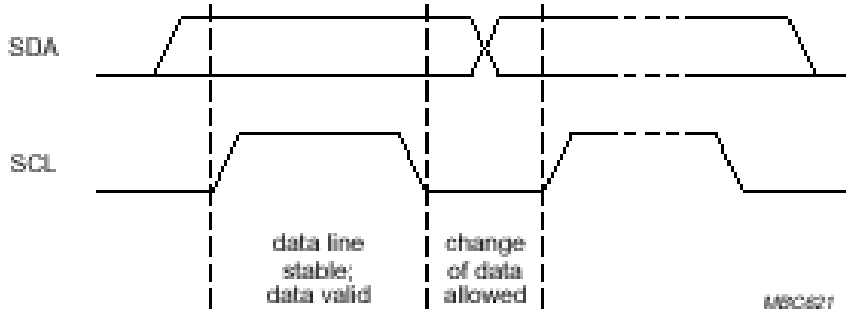
Clock Synchronization

- Clock signal is the sum generated by all nodes
- A slower node can pull down the bus to ‚LOW' in order to insert wait states
- The next cycle is not started before SCL is back to ‚HIGH'

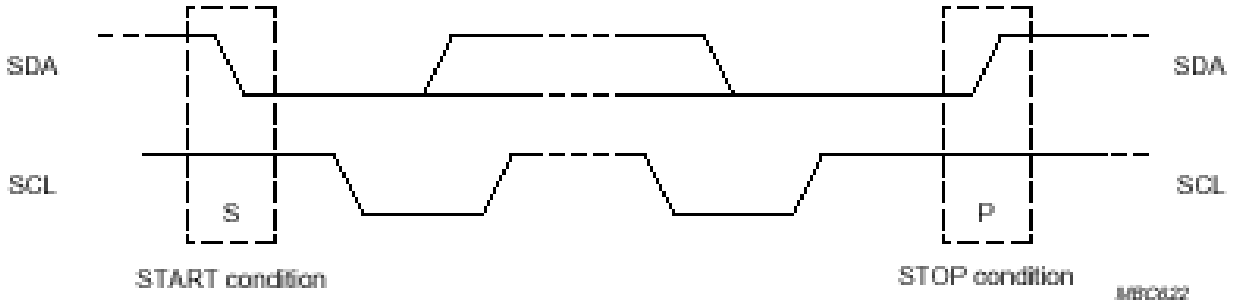


Data Validity

- Data on the bus is only valid when SCL is assigned ‚HIGH‘



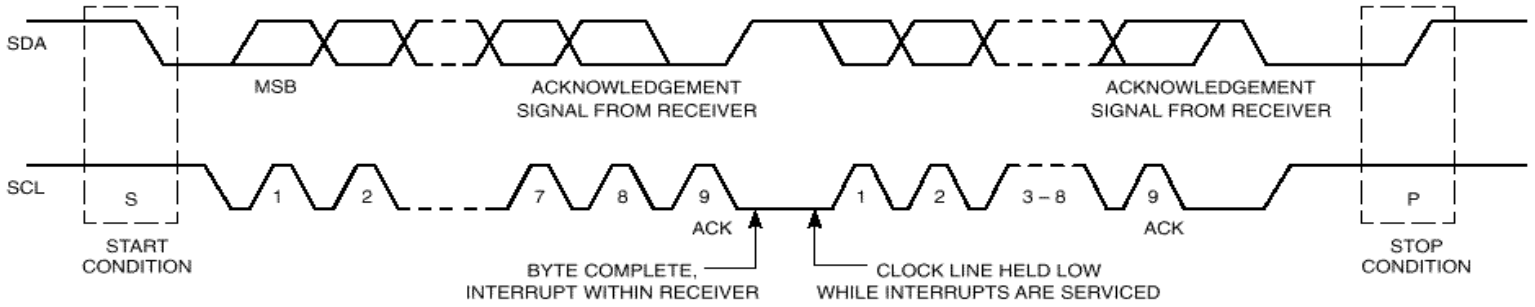
- A switch at SCL = ‚HIGH‘ has a special meaning
  - A switch of SDA from ‚HIGH‘ to ‚LOW‘ is interpreted as start of a transmission (start condition)
  - A switch from ‚LOW‘ to ‚HIGH‘ marks the end of a transmission (stop condition)



Data Transmission

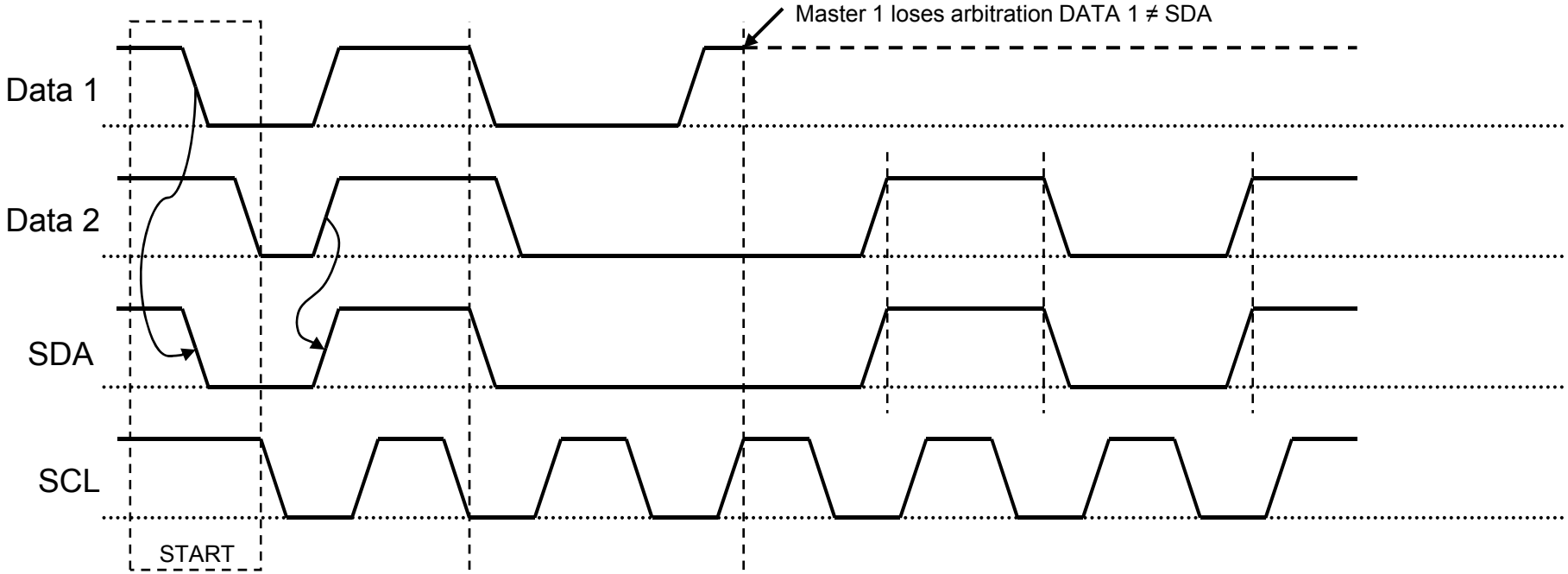
- Data is accepted with the rising edge
- If a slave can not accept data in time it pulls the SCL-line down to ‚LOW‘ in order to insert wait states

- 8 bits per packet, MSB first
- One additional acknowledge bit per packet
  - Receiver pulls SDA to ‚LOW‘ for one clock cycle after data reception
  - If the bus stays ‚HIGH‘, the transfer has to be redone or canceled



Arbitration

- A master is only allowed to send if the bus is free
  - SCL and SDA are ‚HIGH‘
- Several masters can send at the same time
- While sending, all subscribers read back the bus
- If a master detects that data on the bus is not matching the data it has send, it gets passive and does not send further data for this cycle
- No data loss on the bus → CSMA/CA



Framing

- 7bit address of the selected slave is send first
- The 8th bit denotes whether it is a read or write access
- Next, data is transmitted in packets of 8 bit
- After 8 bits there is one acknowledge bit

From Master to Slave

From Slave to Master

A= acknowledge (SDA ‚LOW‘)

*A= not acknowledge (SDA ‚HIGH‘)

S=START condition

P=STOP condition

Addressing

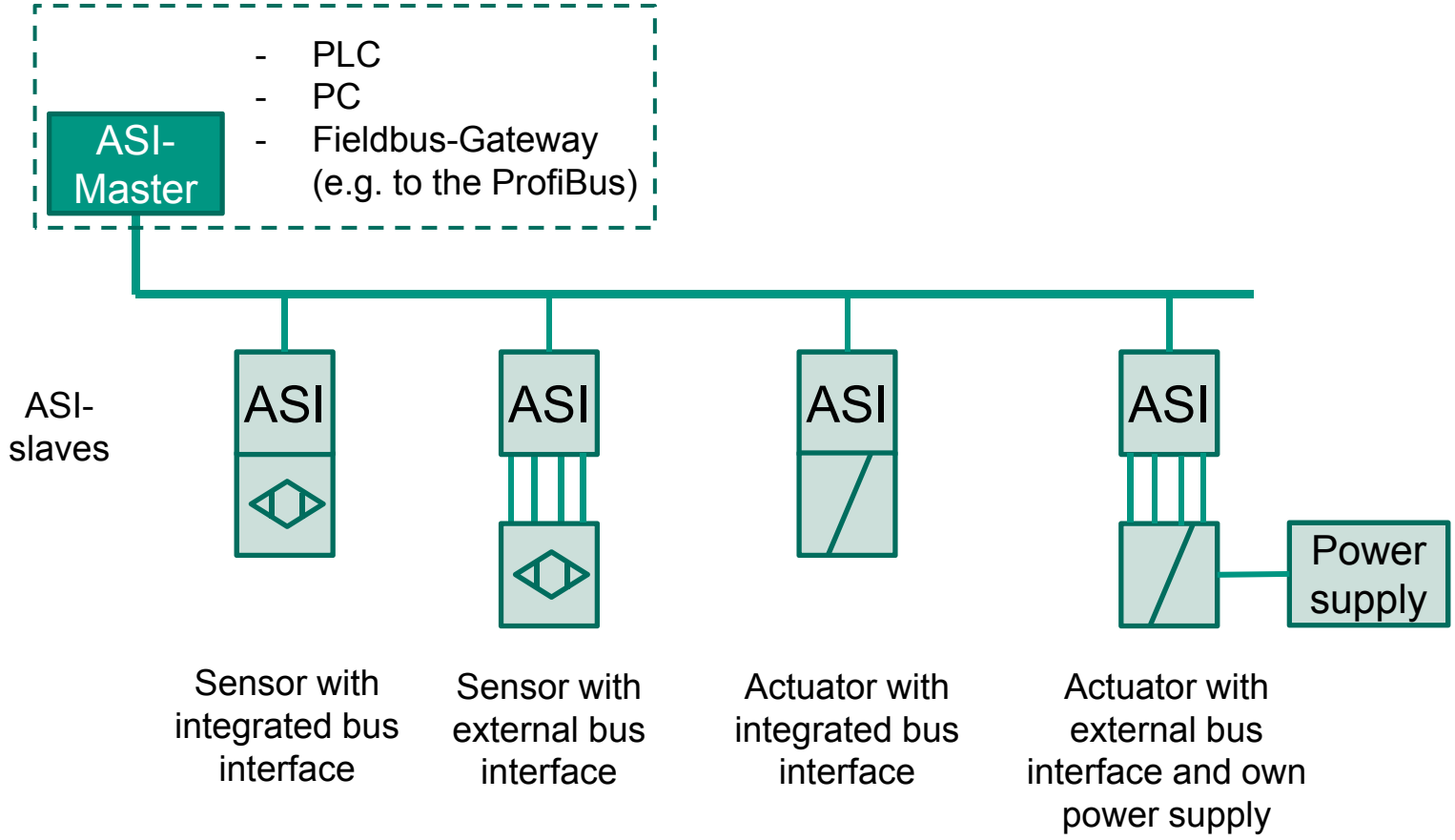
- Problem:
  - Address of bus node can be hard coded by vendor
    → Only one device possible per design
- Solution:
  - Parts of the address can be selected by external wiring
    → DIP-Switches, solder bridge, ...
  - Address can be initialized via the bus
    → Programming using *General Call Address*

The Actuator Sensor Interface (ASI)

- The goal is to directly connect sensors/actuators to the control via a bus
- Requirements:
  - Easy installation
  - Easy commissioning and maintenance
  - Low-priced (because of many bus connections)
- Data and energy should be transmitted on a two-conductor cable for all sensors and most actuators
- Simple and robust transmission procedure without limitations concerning net topology
- Compact and inexpensive bus connection

Concept of the ASI

- Master slave concept using a single master



Transmission Medium

- As transmission medium, unshielded two-wire lines have been specified (→ max. 2A at 24V)

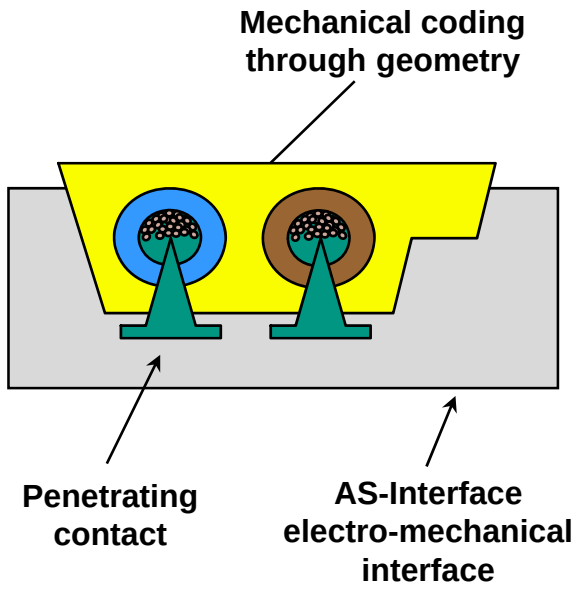
- Two variants:
  - 1.5mm² flat ribbon conductor (→budget-priced)
  - 1.5mm² ASI-specific flat cable (→ advantageous during installation)

- Features:
  - Entire length of the line: 100m (→ extendable by a repeater to 300 m)
  - Max. 100mA current consumption per slave
  - Max. 1 master und 31 slaves (→ max. 124 sensors/actuators) per bus but several ASI-lines can be used in parallel

- Arbitrary topologies (star, tree, line) realizable by a coupling module

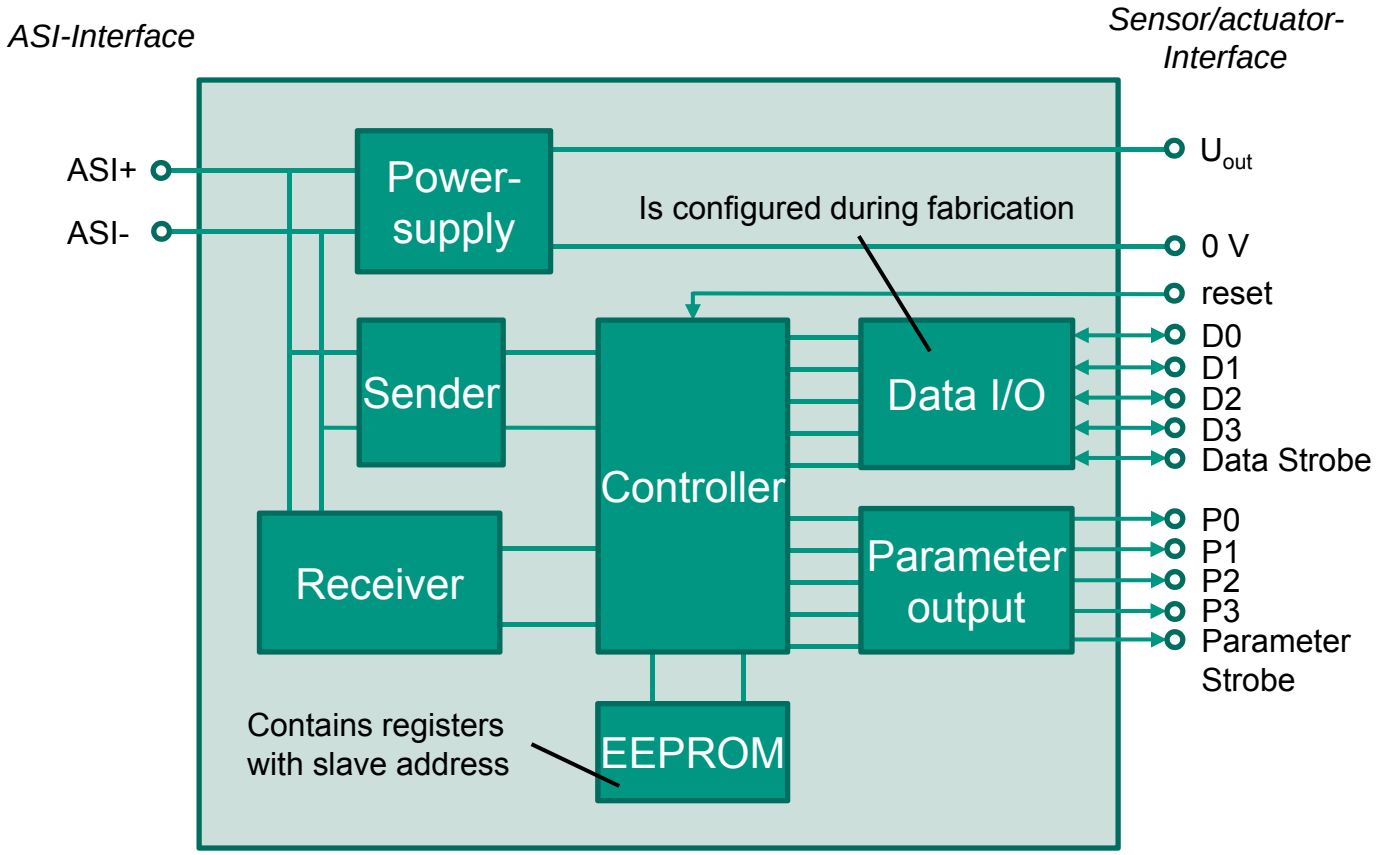
Coupling module

- In the ASI flat cable coupling module the contacting is realized in the form of a penetration technique
- Installation by clipping in the ASI cable without cutting or removing insulation
- Every coupling module can take 2 cables and connect them electrically
- Two types of user modules inside the lid:
  - Active user module: includes electronics for the slave connection → up to 4 conventional sensors/actuators can be connected
  - Passive user module: Without own electronics → for further branching of the ASI line



Structure of an ASI slave

- No complex software needed



Modulation scheme

- The modulation scheme must satisfy the following requirements
  - Free of continuous current, since the data signal is modulated on top of the power supply
  - Master and slave must be able to easily generate the signal
  - Narrow-band since the attenuation of the cables rapidly increases with frequency

- Application of the Alternating Pulse Modulation (APM)
  1. Manchester coding of the raw data (→ phase change with every transition of the sender signal
  2. Generation of an appropriate sender current
  3. Sender current induces a voltage level within an inductivity that is existent only once in the system. The induced voltage can be larger than the supply voltage of the sender (→ neg. voltage at current increase, pos. voltage at descent
     - Low cut-off frequency when voltage pulses are formed as sin²-pulses
     - On specified lines, bit times of 6µs (→167 kBit/s) realizable

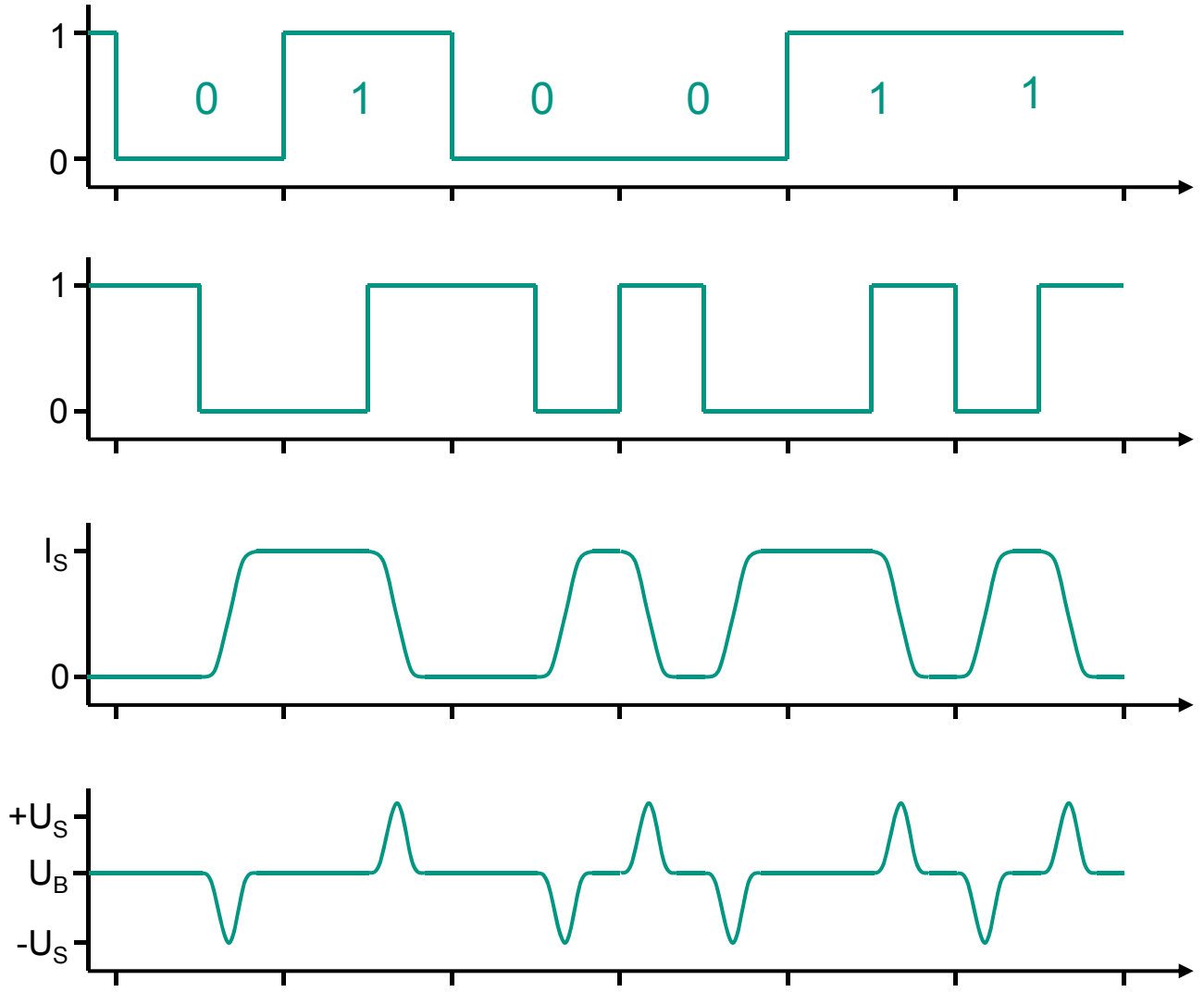
Examples for modulation

Sender bit string

Manchester coded bit string

Sender current

Voltage on signal line

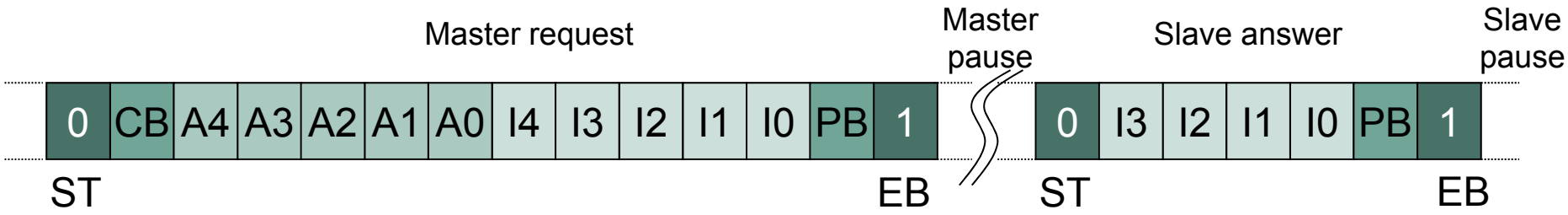


Transmission method

- Bus access: Master-Slave with cyclic polling → real-time capable
- Master sends frame with address of a slave plus data (14 bit with 6µs each)
- Slave answers (7 bit with 6µs each) within a given time
  - Master pause: in general 3, max. 10 bit times (after that, the master assumes that no answer will arrive any more and sends the next request)
  - Slave pause: 1 bit time (6 µs)
- Only 5 bit for information within the frame to keep frames short

  → 14+3+7+1 = 25 bit per cycle → 150 µs per cycle

  → 5 ms overall cycle time with 31 slaves (sufficient for PLC controls)



ST:	Start bit	I4..I0:	Information bits
CB:	control bit	PB:	Parity bit
A4..A0:	Slave address	EB:	End bit

Data Integrity (I)

- Integrity checking uses different criteria than other so far considered bus systems
- Due to the shortness of the frames the checksum overhead would be too large
- ASI tests the signal waveform on physical layer (16 times sampling during one bit time)
- The following rule set is tested by the slave module:
  - Start and stop bits: first pulse must be negative, last pulse must be positive
  - Successive pulses must have different polarity
  - Between two pulses of a frame only one pulse is allowed to be missing
  - No pulse during pause time
  - Even parity

Data Integrity (II)

- High safety:
  - All one- and twofold pulse errors are recognized
  - 99.9999% of all three- and fourfold pulse errors are recognized
  - Parity check takes effect from threefold pulse error on
- Theoretical estimation:
  - Having a bit error rate of 100 errors/s only every 10 years a erroneous frame is not recognized
- Erroneous frames are repeated but do hardly increase the overall cycle time due to their shortness

FireWire

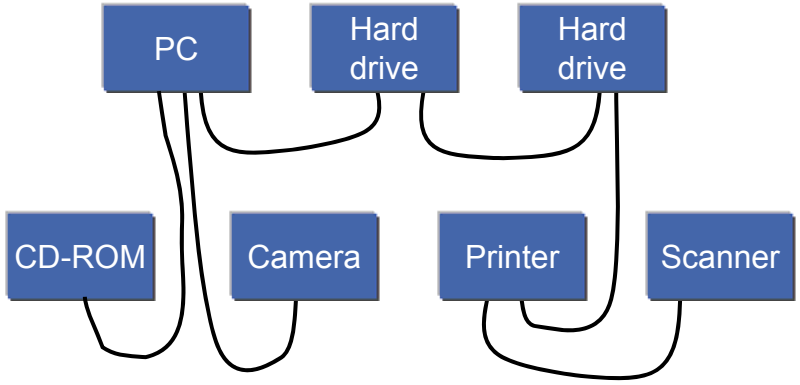
- Multi Master System
- Self-Configuring
- Serial Bus System
- Hot-Plug Capability
- High Data rates
- Real-time capability
- …

IEEE 1394, Firewire (Apple), iLink (Sony)

- Initiated in 1987 by Apple
- Approved as IEEE standard in 1995
- Serial bus, multi-master support
- Differential signaling
- High data transfer rates possible (100Mbit/s, 200Mbit/s … 3.2Gbit/s)
- Hot plugging, self configuration of attached devices
- No PC as controller required
- Uses SCSI-3 Command set
- Implements the first three OSI layers, complemented by the bus manager
- Transmission is possible in asynchronous as well as in synchronous mode

Topology

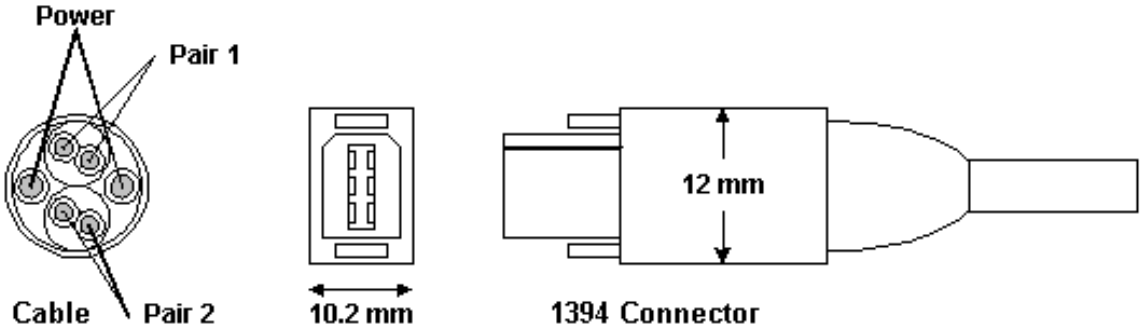
- The standard allows for two different variants:
  - Backplane for installation in device, sporadic use in industrial systems
  - Cable to connect external device
- Several devices can be connected to (cycle free) tree structure
  - All devices are equal
  - No need for dedicated hubs for interconnection
  - Every device can provide several jacks, they are all equal
  - The cable length between two devices is limited to 4.5m
  - At most 16 segments can be chained → maximum length of 72m
- Addressing
  - 64bit addressing according to EUI-64
  - Upper 16bit are used to address the bus and the node
  - Lower 48bit are used for node-internal addressing
  - At most 1023 busses, each with 63 devices possible



Hardware

- Cheap connector
  - 2 variants, 6-pin connector including power supply, 4-pin without power supply

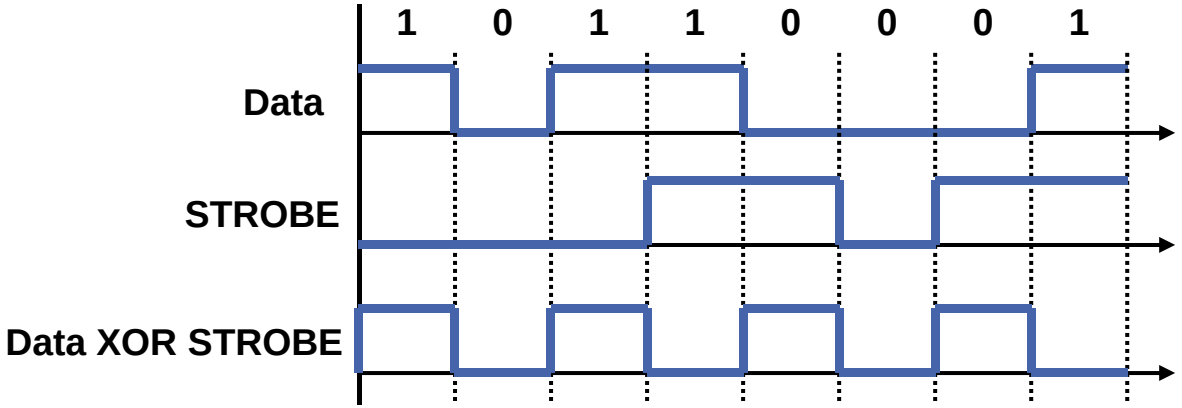
- Supply of external devices possible
  - Supply voltage ranging from 8 to 33V
  - Maximum current 1.5A, max. 48W
  - Operation of smaller devices
  - Supply of the chip-set when device is turned off



- Cable
  - 2 twisted pair wires for data
  - 2 wires for power supply

Encoding

- Data is transmitted differential using two data lines
  - NRZ coding
  - No stuffing or other measures for clock recovery
- The STROBE signal is used to recover the clock signal
  - When equal data bits are to be transmitted, the STROBE signal changes its state
  - Recovery through XOR combination of both signals
  - Advantages
    - Insensitive for interference
    - Lower bandwidth required

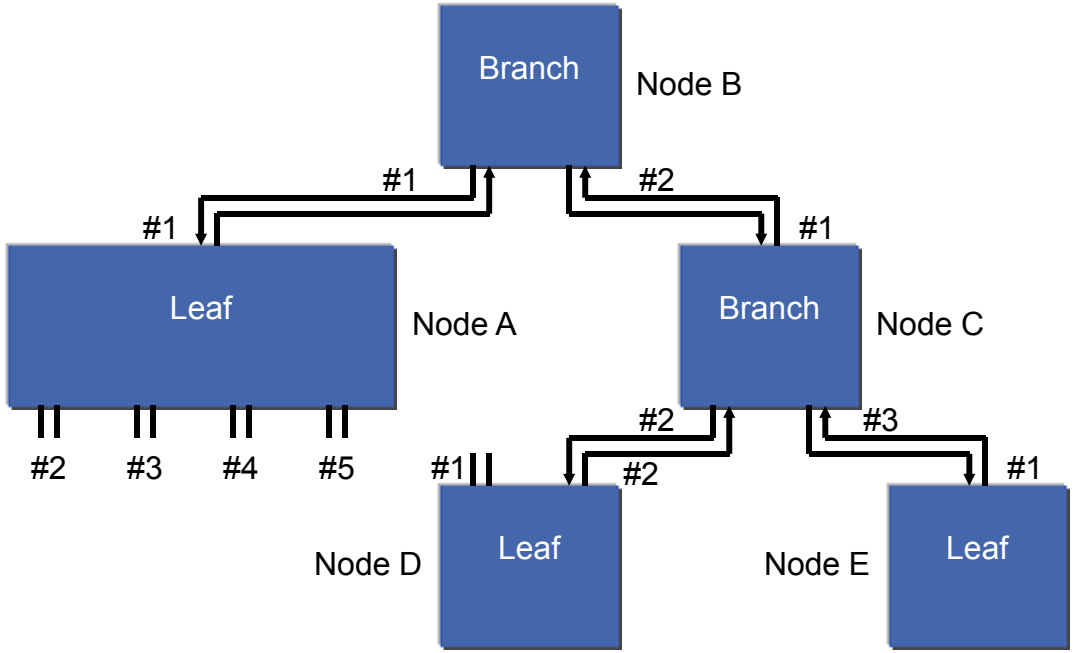


Configuration of the bus

- No user interaction for bus configuration required
- There is no dedicated controller, every bus member can become root of the tree
- Individual addresses and root of the bus is negotiated among the bus members
- Bus assignment is done in three steps
  - Initialization
  - Tree identification
  - Self identification

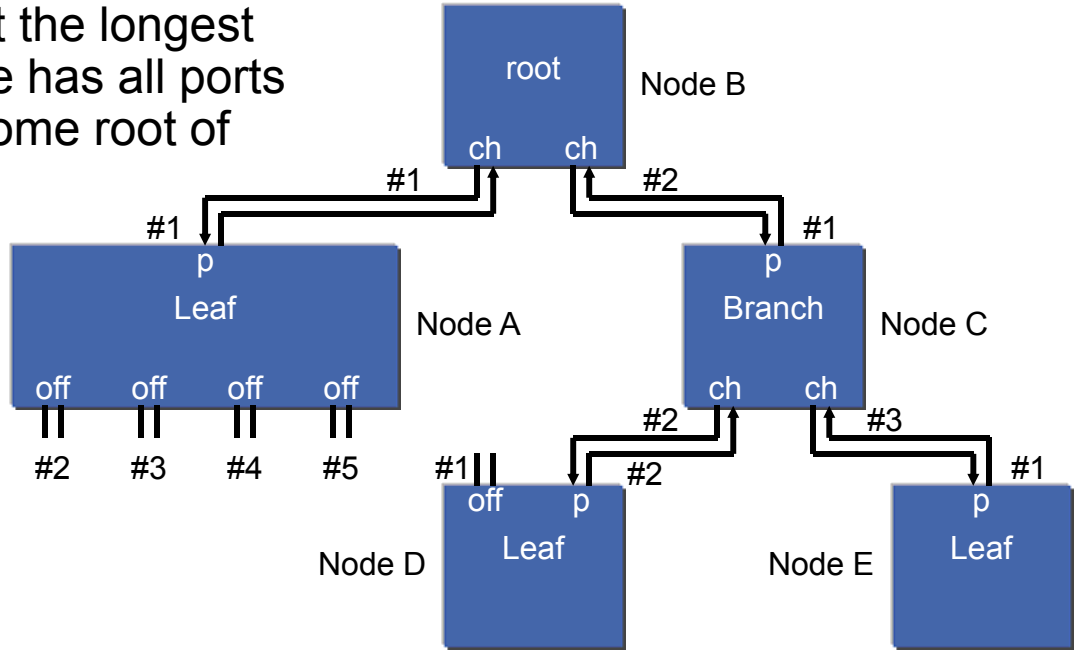
1. Initialization

- After a bus reset, every subscriber only knows the number of outgoing connections
  - One single connection → Leaf
  - Several connections → Branch
- If the bus configuration changes another reset is issued



2. Tree identification

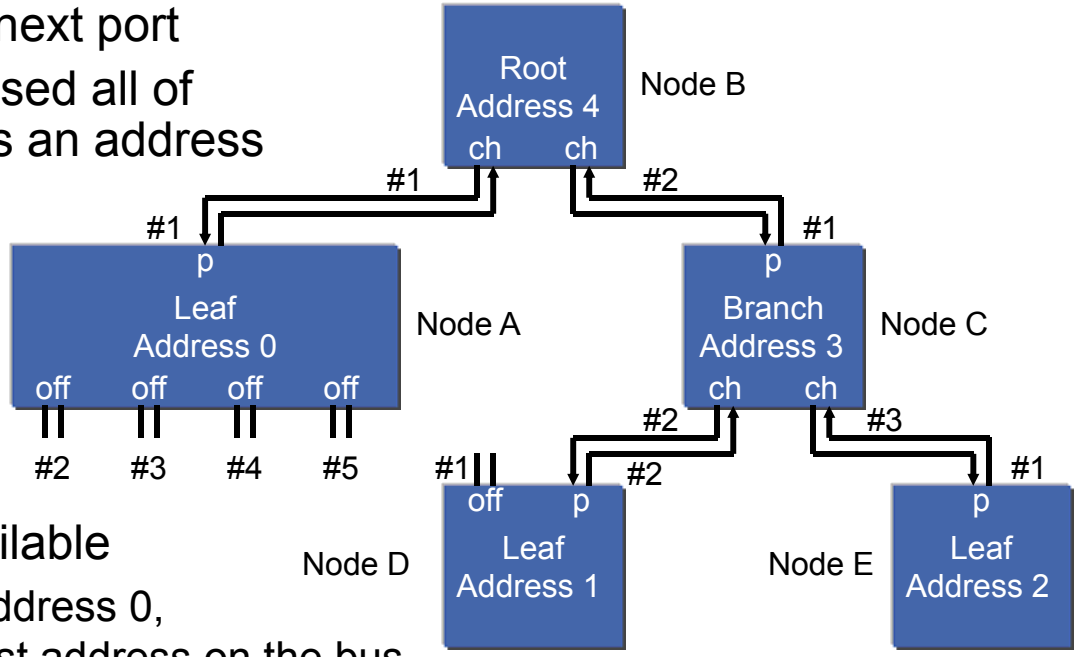
- Method:
  1. Every leaf sends a “parent-notify” packet on the bus and marks the corresponding port as “p” (parent)
  2. The node that receives such a “parent-notify” packet will mark the corresponding port as “ch” (child)
  3. If there is only one unmarked port left in a node, the node will send a “parent-notify” using the unmarked port and marks this port as parent
  4. The node that has to wait the longest time, that in consequence has all ports marked as child, will become root of the tree



- Waiting time and number of hops determine who will become root of the tree
- Also a leaf can become root of the tree if it is waiting long enough

3. Self identification

- Assigning the node address
  1. Root transfers control to the node that is connected to the first port of the root
  2. If this node has children on its own, he will forward control recursively to the first child node
  3. When a leaf is reached, it will claim the next address that is available and sends a packet with its address and additional status information to the bus.
  4. After that, control is returned to the parent node that processes the next port
  5. When a node has processed all of its children, it self-assigns an address and transfers control to its parent node



- Every node counts how many devices have already send their address and then chooses the next address available
  - Thus the first leaf gets the address 0, while the root gets the highest address on the bus

Arbitration

- The bus has to be idle for a certain amount of time *(subaction gap, 10µs)*, before a node can opt for the bus
- The node then sends a *request packet* to its parent
  - If no other request already reached the parent node, it will forward the request one level up. A *data prefix packet* is send to the other ports in order to suppress other requests
  - If there is already another request, the last node gets back a *data prefix packet* to inform the node that the request has been denied
- When the request reaches the root, a *grant packet* is send back to the source node of the request
- The requesting node acknowledges with a data prefix and starts the transmission

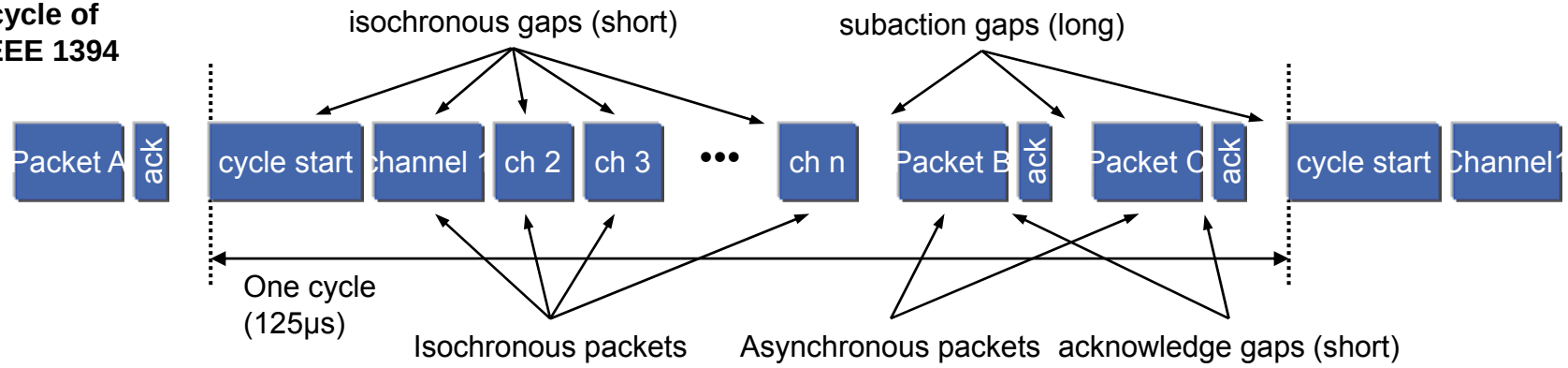
Fairness

- In principle, the node directly connected to the root has the highest chance to acquire the bus
- To preserve fairness, every node is only allowed to acquire the bus once in a cycle
  - Therefore every node has an *arbitration enable bit*, that is cleared after a successful access
  - The node is then not allowed to apply for the bus again
- If there was no access to the bus for a certain amount of time *(arbitration reset gap, 20µs)*, all nodes reset their *arbitration enable bit* and a new cycle is started

Isochronous transfers

- The cycle master (root) controls the transfer in frames of 125µs
- An asynchronous packet can have a maximum length of 62µs
- Addressing is done using channel numbers that have been negotiated at the beginning
- Nodes that want to send isochronous packets apply directly for the bus without waiting for the *subaction gap*
  - *Thus isochronous transfers have higher priority then asynchronous transfers*
  - *If no other node wants to start an isochronous transfer there will arise a subaction gap so that other nodes can apply for the bus*

Bus cycle of the IEEE 1394

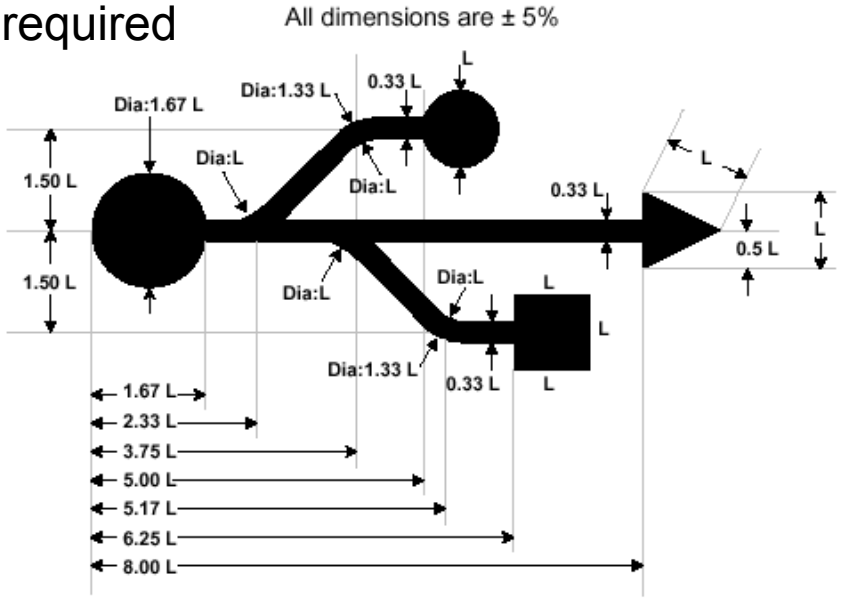


Universal Serial Bus (USB)

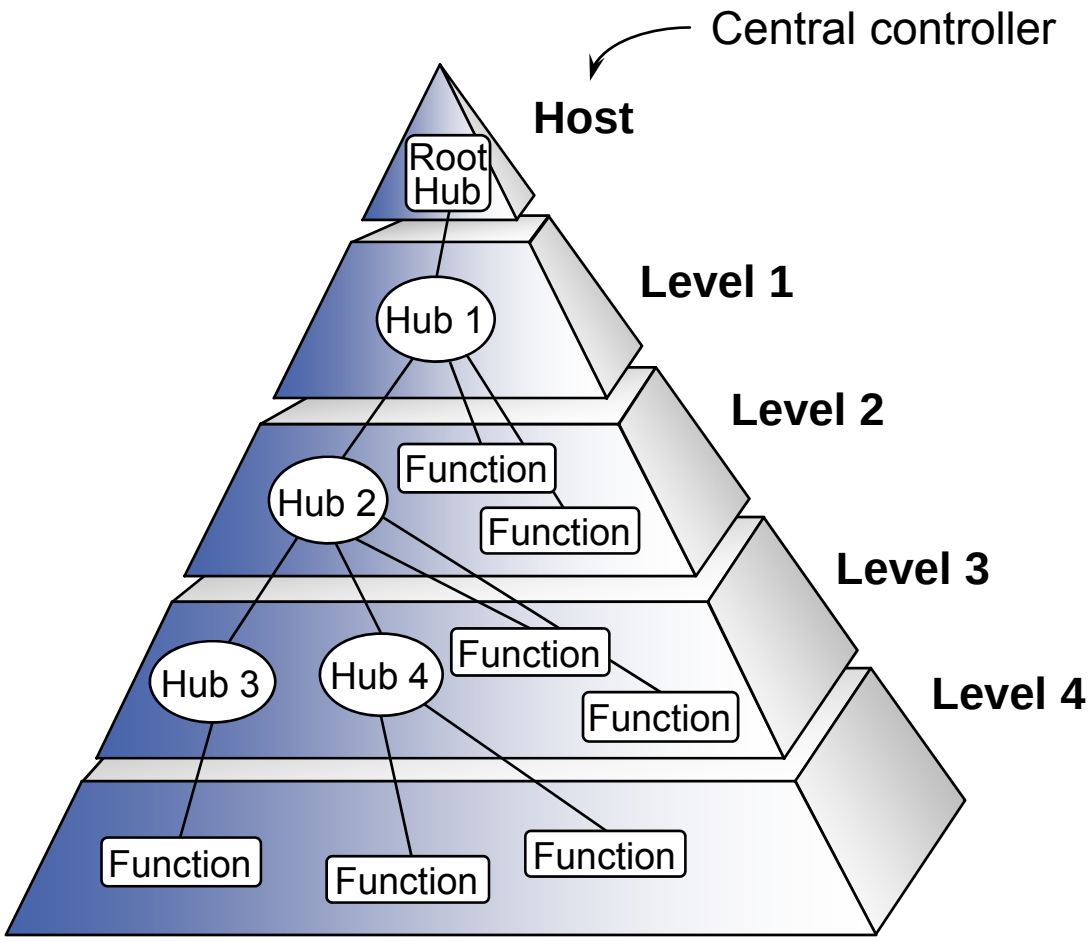
- Serial Bus
- NRZ-I Encoding
- Single Master / Multi Slave
- Hot-Plug Capability

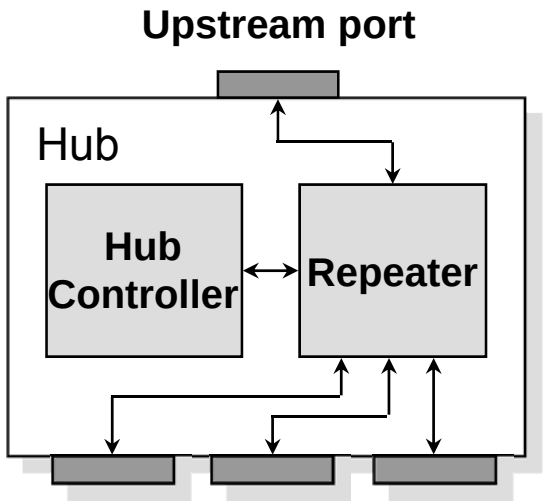
Universal Serial Bus (USB)

- Bus system used to connect peripheral devices like mouse, keyboard and printer to the PC
- Serial bus
  - Hot-plugging, auto configuration
  - At most 127 attached devices possible
  - Tree topology, branching is done in hubs
  - Point-to-point connections between individual devices
  - Central control through one single host (PC)
  - Polling is done by host, no arbitration required
  - Different speeds:
    1.5 Mbit/s, 12 Mbit/s,
    480 Mbit/s (USB 2.0),
    5000 Mbit/s (USB 3.0)
- Mechanical connection
  - Different types of connectors
- Electrical characteristics
  - Differential signaling
  - Integrated power supply



Topology

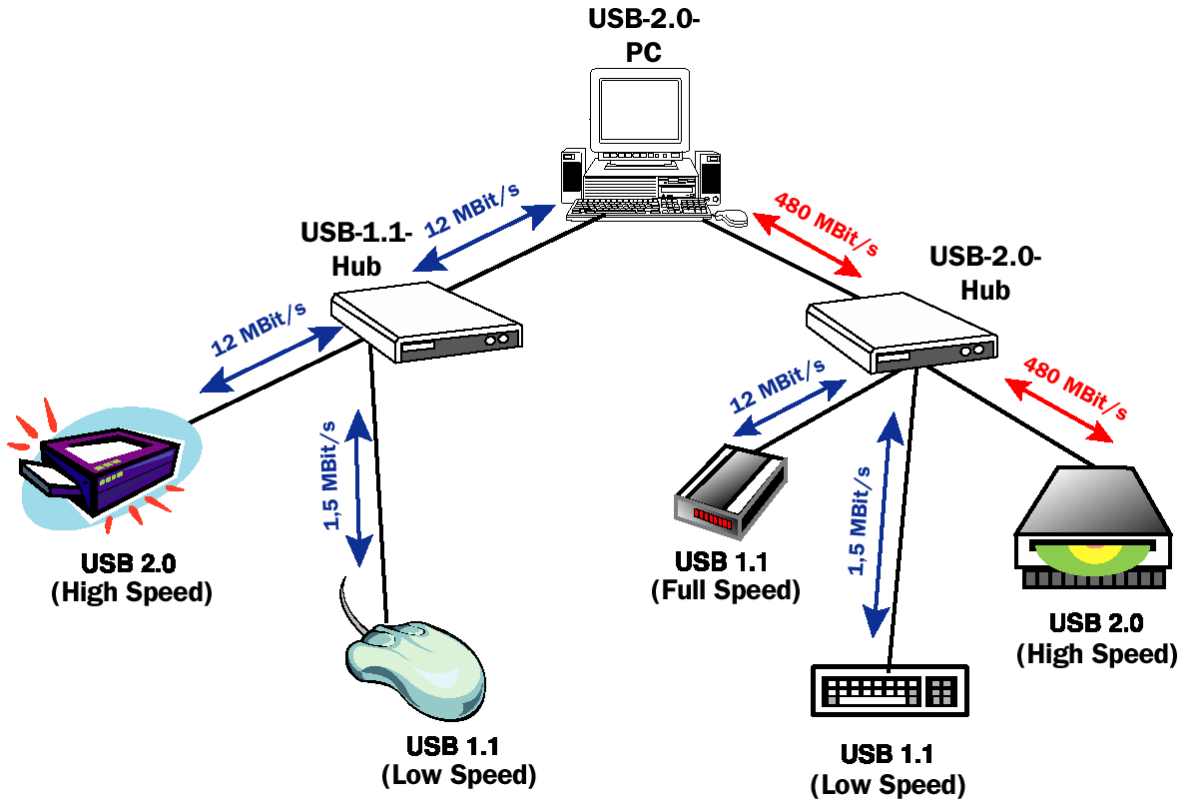




active: max. 500 mA
passive: max. 100 mA

Topology in hybrid mode with USB 1.1 and 2.0

- Hybrid mode
  - USB 2.0 is compatible to USB 1.1
  - High Speed mode is only possible if root hub as well as all intermediate stations support USB 2.0

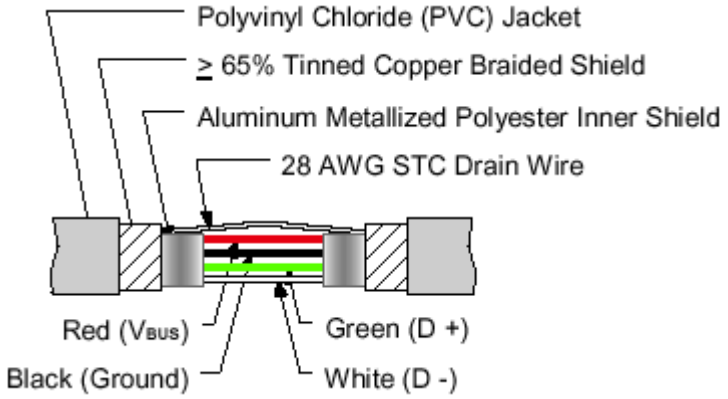


Mechanical connection

- In order to avoid illegal connections, two different types of connectors exist: Series A and Series B
  - Series B: pins in pairs one upon the other
  - Power supply pins are 1mm longer (hot plugging)

Bottom: serie A, top: series B

- Cable
  - 4 wires
  - 2 data wires
  - Integrated power supply



Differential signaling

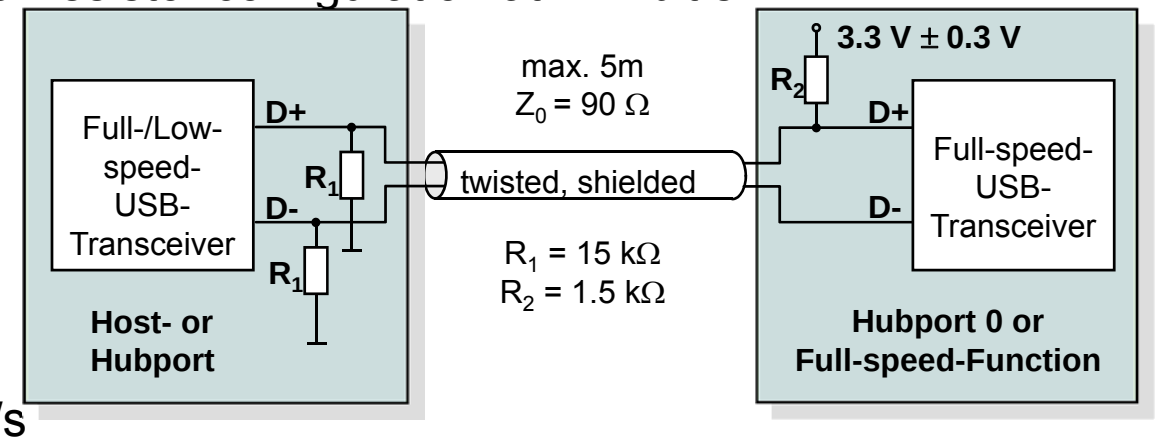
- Voltage on the wires 0-3.3V
- Differential transmission V_{D+}, V_{D-}
  - Differential ONE: $V_{D+} > V_{D-}$
  - Differential ZERO: $V_{D+} < V_{D-}$
- Additional single ended signals are possible
  - V_{D+} and V_{D-} < 0.3V → single ended Zero (SE0)
  - Marks the end of a packet

- Different interpretation of the states depending on transmission speed
  - State ‚J‘
    - Differential ZERO at Low-Speed (1.5Mbit/s)
    - Differential ONE at Full-Speed (12Mbit/s)
  - State ‚K‘
    - Differential ONE at Low-Speed (1.5Mbit/s)
    - Differential ZERO at Full-Speed (12Mbit/s)

- In High-Speed mode (480Mbit/s) signaling is done using constant currency of 17.78mA

Transmission speed detection

- Devices announce their transmission speed using different pull-up resistors at the data wires
- Example for resistor configuration at 12Mbit/s



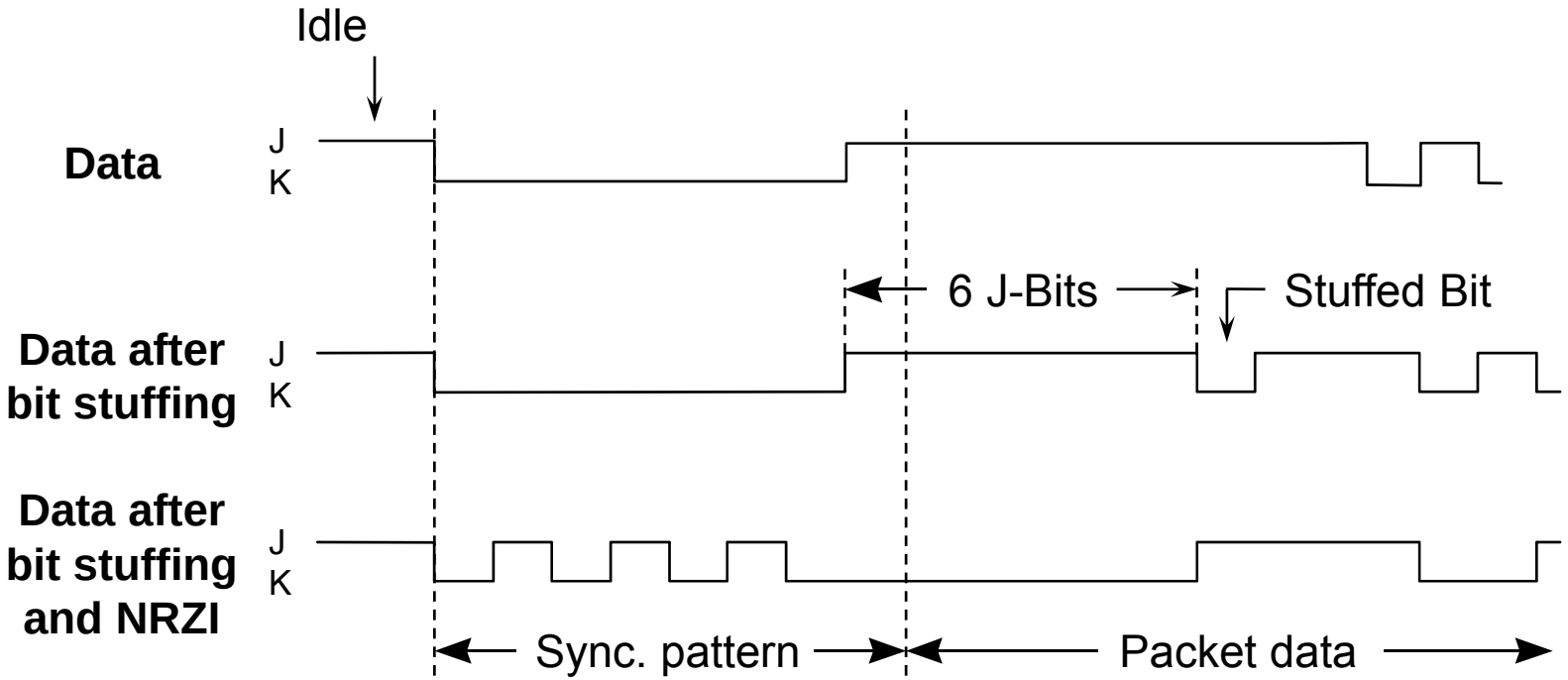
- At 1.5Mbit/s
  - Low-Speed Transceiver
  - R_2 at D-
  - Cable does not need to be twisted/shielded
  - Z_0 arbitrary
  - Max. 3m wire length
- High-speed devices announce themselves as Full-speed devices (12Mbit/s) first
  - At configuration time, the host is informed about the higher transmission rate possible
  - In normal operation, the resistor is disabled in order to provide symmetrical wire behavior

Energy consumption

- Devices can be supplied with energy (5V) over the bus
- 3 different classes have to be distinguished
  - Low-power devices: max 100mA
  - High-power devices: max 500mA
  - Self-powered devices: max 100mA from the bus, the rest has to be fetched from their own power supply during operation → otherwise they can be disabled by the host
- Devices have to announce their power consumption at configuration time and are not allowed to go beyond this limit
- Devices have to support standby mode
  - Maximum power consumption of 500µA
  - Devices have to go into standby after 3ms of inactivity on the bus or by dedicated command of the host

Coding

- Individual bits are encoded using NRZ-I code: Level changes only in K-state
- Bit stuffing: after 6 J-Bits, one K-Bit is inserted



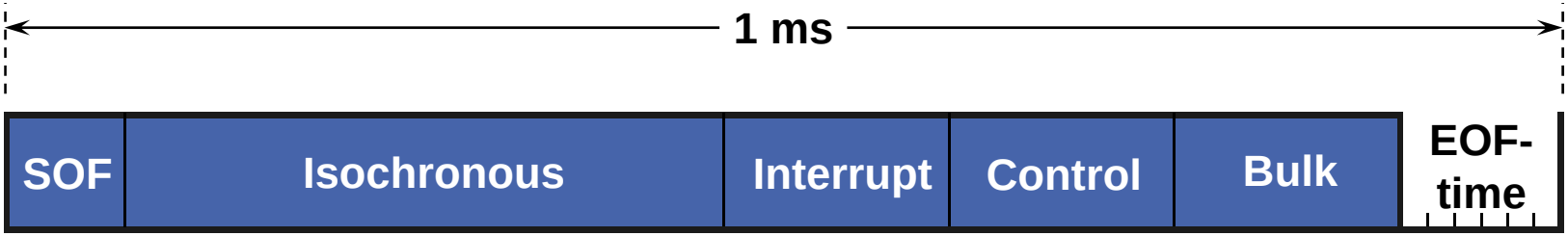
Data transmission

- Bus is controlled only by host
- Host is polling every subscriber und sends or requests data
  - No arbitration scheme and collision avoidance necessary
  - High burden on host because of polling → intelligent controller required
  - Prioritization is determined by host only through order of polling
- Isochronous transmission
  - Fixed data rate, no error handling, e.g. for video transmission
- Asynchronous transmission

Control-Transfer	Interrupt-Transfer	Bulk-Transfer
♦ Used for commands and status information ♦ Initialization	♦ Requires a certain service period ♦ Only for IN direction	♦ For large amounts of data ♦ Long waiting time possible ♦ Only at 12 Mbit/s

Data format

- Every frame has a length of 1ms
- Composition of the frame
  - Start-Of-Frame (SOF) packet for synchronization and identification
  - Transfers of the different categories, depending on available time budget
    - Prioritization is as follows:
      - Isochronous, Interrupt, Control, Bulk
    - Max. 90% of the bandwidth can be used for isochronous and interrupt transfers
    - Max. 10% can be used for control transfers
    - Remaining bandwidth can be used for control and bulk transfers, where control transfers have higher priority
  - End-Of-Frame (EOF) time for the rest of the cycle length of 1ms



Transmission packets I

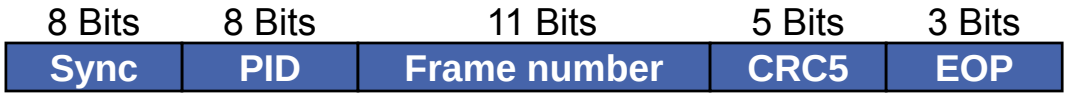
- The packets used for data transmission are composed of the following fields
  - Sync (synchronization)
    - All packets start with a sync field
    - It consists of 7 K-bits and 1 J-bit and is used for synchronization of the senders and receivers clock
  - PID (Packet ID)
    - Determines the type of the packet
  - ADDR (Address)
    - The target address of the packet
    - Length 7bit → 127 devices can be addressed, address 0 is reserved for initialization

Transmission packets II

- ENDP (Endpoint)
  - Sub address within a device
  - Length of 4bit, in Low-Speed devices, only 2 sub addresses are allowed
- CRC (Cyclic Redundancy Check)
  - Checksum for error detection
  - In token packets 5bit, in data packets 16bit long
- EOP (End-Of-Packet)
  - Marks the end of a packet
  - Signalling is done issuing SE0 for 2 bit times followed by one J-bit

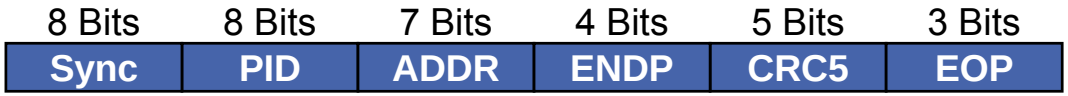
Packet types I

- Start-Of-Frame packet
  - Marks the start of a new transfer frame



SOF-Token

- Token packet
  - Determines type and direction of the data to be transmitted
    - IN: Transfer from device to host
    - OUT: Transfer from host to device
    - SETUP: Marks a control transfer



Token

Packet types II

- Data packet
  - User data of the transmission

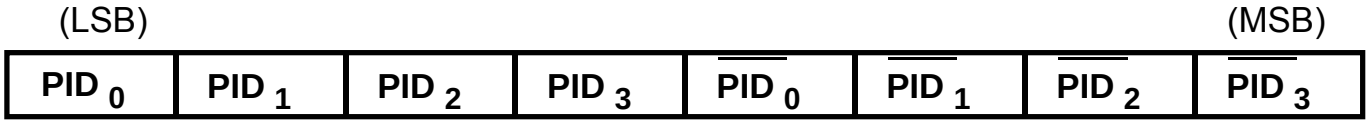
Data

- Handshake packet
  - Used to acknowledge transferred data
  - ACK: Packet has been received successfully
  - NAK: At the moment, device is not able to send/receive data
  - STALL: Error case, intervention of the host is required

Handshake / Low-Speed-Preamble

PID field

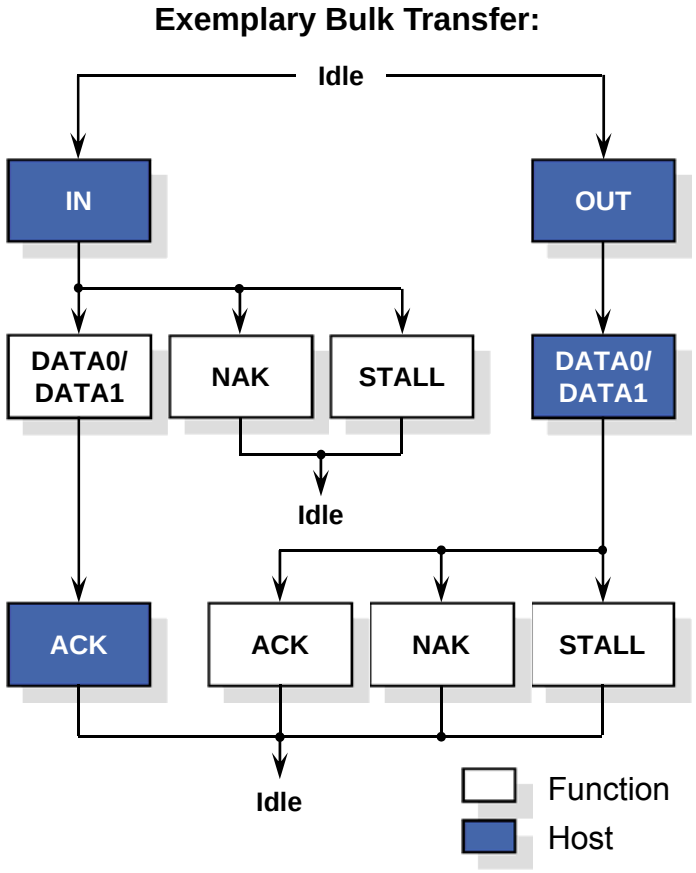
- Using the PID field, the device is able to detect the type of packet
- The PID has a length of 4bit, for error detection purpose, these bits are additionally appended in inverted form
  - If the bits do not match, the packet is dropped



Paket type	PID name	PID [bit3:bit0]	Description
Token	OUT	0001	Transfer from host to device
	IN	1001	Transfer from device to host
	SOF	0101	Start of a new frame, includes frame number
	SETUP	1101	Device is to be configured by the host
Data	DATA0	0011	Changes after each successful transmission
	DATA1	1011	
Handshake	ACK	0010	Reception without errors
	NAK	1010	Device is not able to send/receive data
	STALL	1110	Same as NAK, but action from the host required
Special	PRE	1100	Used to access a low-speed function

Transfer process

- The host starts a transfer by sending a corresponding token packet
- Then one ore more data packets are transmitted
- Each packet is acknowledged by a handshake packet
- If errors have occurred during a transmission, no handshake packet is send. The sender has to retransmit the packet
  - For isochronous transfers there is no handshake. Defective data is not transmitted again!



Adding a device to the bus (Enumeration)

1. The hub/host detects a new device via the pull-up resistors
2. A reset is being performed in order to set the device to a well defined state. The device then uses the reserved address 0.
3. The host queries the configuration parameters of the device
4. The host assigns an address to the device and configures the device using control transfers
5. The device is now ready to operate on the bus

Pictures: http://www.yatego.com

USB on-the-go (OTG)

- Extension of USB 2.0 used to connect two devices directly
  - No host required
  - Can be used e.g. to connect digital camera and printer
- Requirements for such a dual-role device
  - Simple host capabilities
  - List of supported devices (not all possible USB devices have to be supported)
  - A micro-AB connector (new type of connector)
  - Has to be able to drive 8mA to the bus

Outlook

- Presentation of actual communication systems

- Automotive Busses
  - CAN
  - LIN
  - FleyRay