FAKULTÄT FÜR !NFORMATIK

Faculty of Informatics

182.694 Microcontroller VU

FAKULTÄT FÜR !NFORMATIK

Martin Perner

SS 2017

Featuring Today:
Digital Communication

Weekly Training Objective

- Already done
  - 3.1.3 Floating point operations ∗
  - 3.3.1 Interrupt & callback demo ∗
  - 3.6.1 UART receiver ∗
  - 3.6.2 UART sender ∗
- This week
  - 3.4.4 PWM signals and glitches
  - 3.6.4 TWI (I^2C) ∗
  - 3.9.1 Keypad
- Next week
  - 3.5.2 Noise
  - 3.5.3 Prescaler and accuracy ∗
  - 3.7.5 Dynamic memory analysis

Application 1

- Are there any questions?
- Already started with the theory task?
- Remember: having done Application 1 may not be enough to pass the second exam!

Submission Application 1

Submission

- The uploaded archives (code and protocol) must follow the template, which is provided on the homepage!
- The deadlines are firm!
  14.5. for the code
  21.5. for the protocol

Submission Application 1

Organization

- You need to take part in a delivery talk!
  - A code submission is required in order to enrol to a delivery talk!
- You can use an external libraries, but you will not get points for these modules (e.g., GLCD, ZigBee) if you do!
  - the avr-libc and a FFT library are exceptions.

Submission Application 1

The Delivery Talk

- The talk takes 25 minutes.
  Please, be on time!
- The registration deadline is on the 14.5., and requires a code submission beforehand!
- There will be a slot for everyone.
  Nonetheless, we urge you to register early!
- Print the protocol cover sheet, sign it, and give it to the tutor during the talk.
- The tutor will not only check if your application works, but also ask you some questions about the hardware and your code.

Submission Application 1

After the Submissions

- Do not speculate that you will get enough points for Application 1.
- We will need a few weeks to correct the protocols!

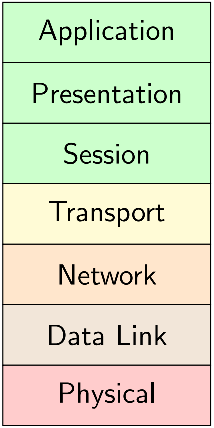
Upcoming Dates

- 8.5. TinyOS Part 1
- 14.5. Application 1 Code Deadline
  Delivery Talk Enrollment Deadline
- 15.5. Recitation for the second Exam, introduction Application 2.
- 19.5. Second Exam
- 21.5. Application 1 Protocol Deadline
- 22.5. TinyOS Part 2
- 26.6. Recitation for the third Exam (in the Lab!)

On the Mondays following the 22.5. will be in the lab at the time of the lecture (16:15-17:45).

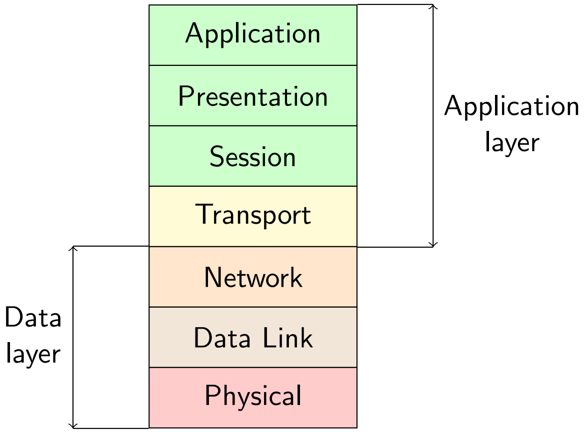
The OSI Model

The 7 layer OSI Model allows to assign every component involved in the communication process to a certain layer.



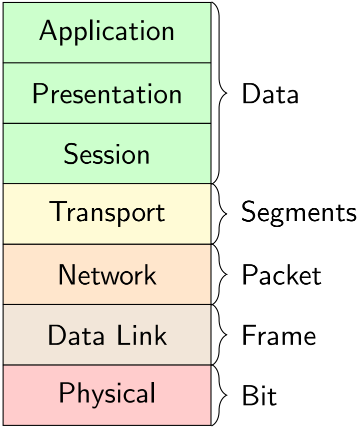
The OSI Model

The 7 layer OSI Model allows to assign every component involved in the communication process to a certain layer.



The OSI Model

The 7 layer OSI Model allows to assign every component involved in the communication process to a certain layer.



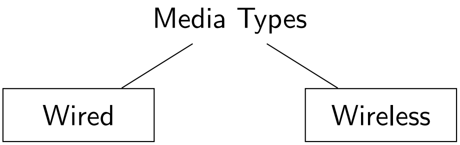
The OSI Model

The 7 layer OSI Model allows to assign every component involved in the communication process to a certain layer.

Application	HTTP
Presentation	XML
Session	PPTP
Transport	TCP
Network	IPv4
Data Link	MAC
Physical	IEEE 802.3

Layer 1 – Physical

Responsible for the actual transfer of the data bits.



Layer 1 – Physical

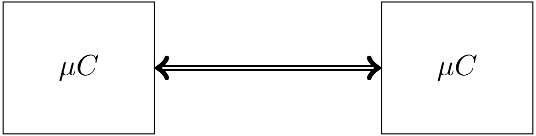
Responsible for the actual transfer of the data bits.

Media Types

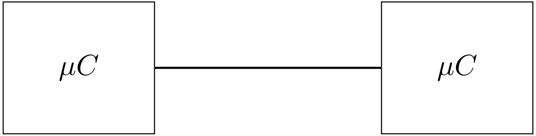
- Wired — a short overview
- Wireless — 389.146 Introduction to Telecommunication

Communication Between Two Endpoints

We will now dive into the topic of media access in wired digital communication.

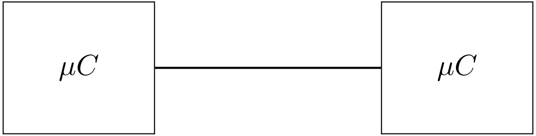


Single Line



Why not just a single line?

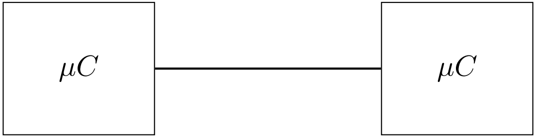
Single Line



Why not just a single line?

- Is there a reference level?

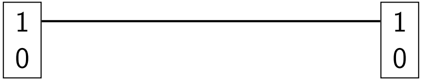
Single Line



Why not just a single line?

- Is there a reference level?
- Recall Exercise 2.2.2 input with floating pins

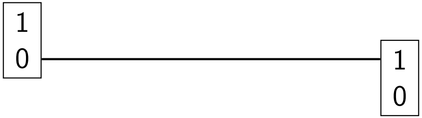
Analogy: Single Line



Consider to following analogy

Two mechanical levers in gravity-free environment.

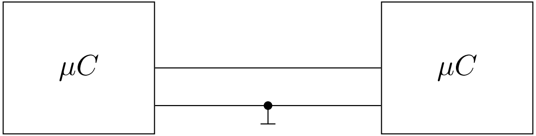
Analogy: Single Line



Consider to analogy

Left lever sets '0' but right lever still reads '1'.
Equivalent to floating pins.

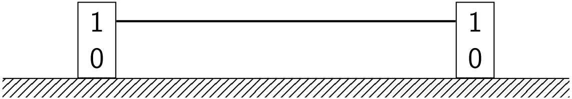
Single Line and Ground



No "default" state

What happens on start-up?
Impact of noise?

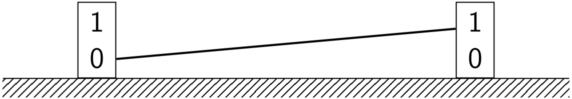
Analogy: Single Line and Ground



Consider to analogy

Fixing the levers to a common plate.

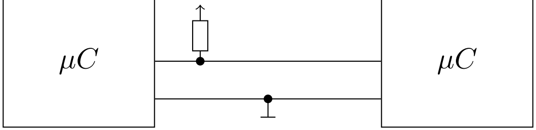
Analogy: Single Line and Ground



Consider to analogy

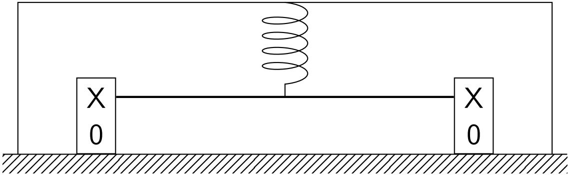
Fixing the levers to a common plate.
Bidirectional sending could result in conflicting driver
$\Rightarrow$ large current flowing and no transmission

Single Line, Ground, and Pull-Up



No output needed to send a '1'

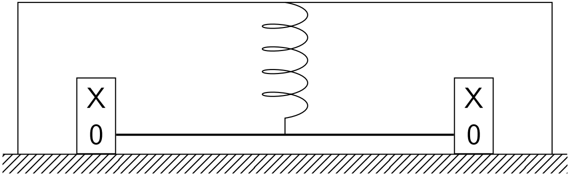
Analogy: Single Line, Ground, and Pull-Up



Consider to analogy

A weak spring keeps the bar in the high state.

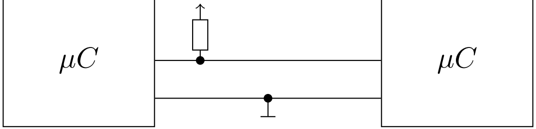
Analogy: Single Line, Ground, and Pull-Up



Consider to analogy

It is not possible to determine who pulled the value to 0!

Single Line, Ground, and Pull-Up

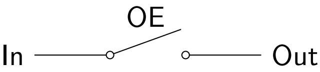


No output needed to send a '1'

How can we do that when we

- do not want to write '0' at the port and
- do not want to have a current flowing when one participant pulls the bus down?

Tri-State



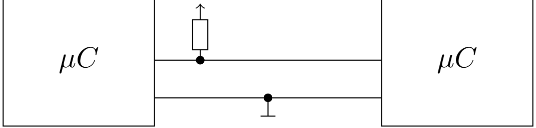
Tri-state, three-state logic, open collector, open drain, . . .

describe the same thing:

- A port that has the usual 0 and 1 (forwarding *In* to *Out*),
- but also third state, the Hi-Z state (*OE* disabled).

In the Hi-Z (high-impedance) state, the port's influence to the connected circuit is removed.

Single Line, Ground, and Pull-Up



The Result

- Tri-state output needed instead of '1'.
- Pull-up introduces recessive state. Default level is high.
- Writing a '0' will bring the bus to '0'; due to the weak pull-up only a small current will flow.
- Can be used for arbitration: '0' is dominant.
- Low data rate or cable length, as signal needs to propagate, especially if used for arbitration!

Single Line, Ground, and Driver

Adding an additional driver

- With increasing wire length, the μC alone might not be powerful enough to keep a constant/stable voltage on the whole length of the wire
- Or a different voltage level/media conversion is desired

Therefore, drivers can be placed between the port of the μC and the cable.

Line Coding

We know how to transmit data from A to B, but ...

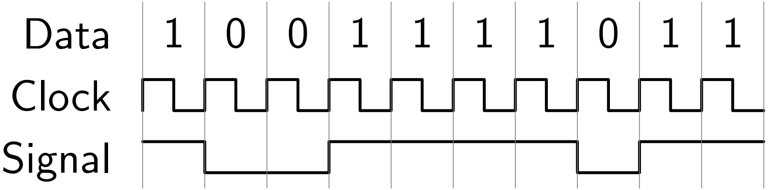
- is that sufficient?
- is that optimal?

Generic Approaches – Basic Line Codes

Why would we not just transmit '0'/'1'?

- No shared clock between the sender/receiver ⇒ loss of synchronization
  - We want to achieve clock regeneration!
  - Avoid long consecutive patterns of the same value
- Can avoid DC component of the signal
  - Allows a AC coupling between transfer medium and transceiver
- ...

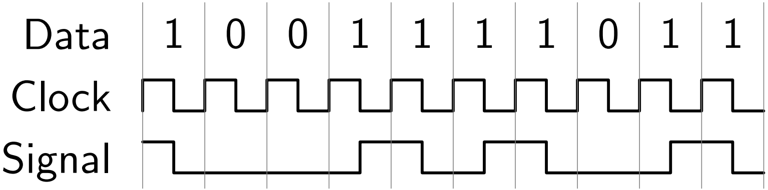
Non-Return-To-Zero



Non-Return-To-Zero

The direct way.

Non-Return-To-Zero Inverted



Non-Return-To-Zero Inverted

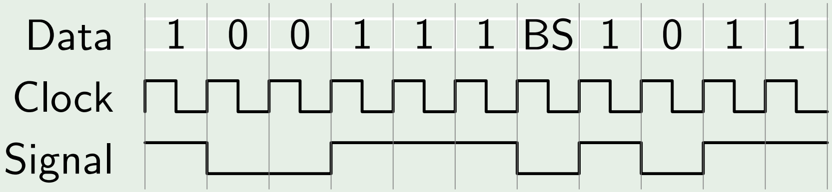
- Zero: No transition.
- One: Transition at half-clock.

Bit Stuffing

- Assume that the clock at the receiver and the sender drift.
- What happens after a long phase without transition?
  - They can drift apart and lose synchronization $\Rightarrow$ bit loss
- Solution: add a transition after a (protocol) specific amount of non-transition symbols.
- Allows resynchronization of receiver onto sender.

Bit Stuffing

Example (NRZ with bit stuffing after 3 bits)

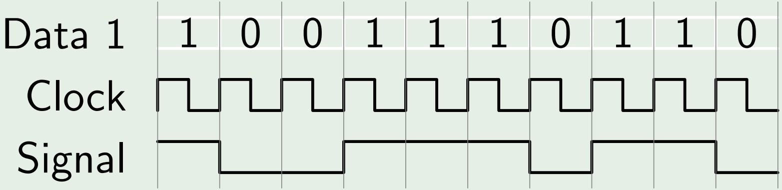


Bit Stuffing

Can we omit the stuffing bit if the next bit causes a transition?

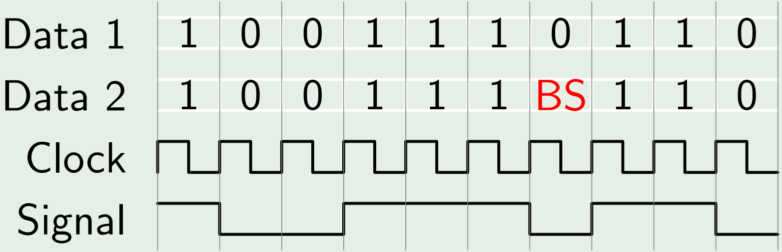
Bit Stuffing

Example (NRZ with bit stuffing after 3 bits, omit stuffing if next symbol has transition)



Bit Stuffing

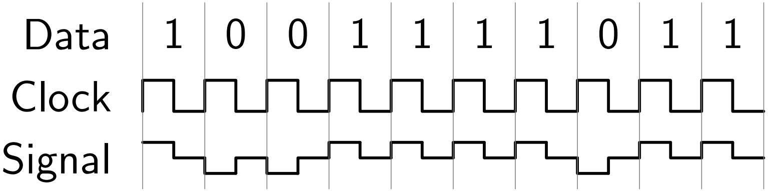
Example (NRZ with bit stuffing after 3 bits, omit stuffing if next symbol has transition)



Bit Stuffing

Can we omit the stuffing bit if the next bit causes a transition?

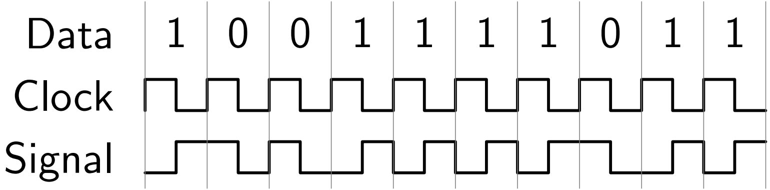
No, we can not!
Otherwise there is no way to distinguish between a stuffing bit and a valid signal change!



Return-To-Zero

- Uses 3 signal levels: $+1$, 0, and -1.
- At half-clock return to 0.
- Allows constant clock regeneration at the cost of bandwidth.

Manchester



Manchester

- Similar to return-to-zero but only 2 signal levels.
- Data = Clock XOR "Manchester Value"
  The Manchester Value (of a single bit) used depends on the standard, the one used above is used in IEEE 802.3.
- DC–balanced (if signal levels are $\pm X$ V).
- But again, we lose half of the bandwidth.

DC–balanced Encoding

8b/10b Encoding

- Use one 10 bit symbol to encode 8 data bits.
- DC–balanced in the long run (± 1 disparity/offset at end of a symbol).
- Current disparity can influence the choice of the next symbol, i.e., there may be multiple symbols for the same 8 data bits.

There are also other encodings with different symbol length.

Comparison

	NRZ	NRZI	RZ	Manchester	8b/10b
bandwidth utilization	≈ 1	≈ 1	$^1/_2$	$^1/_2$	$^8/_{10}$
clock regeneration	$\times$	$\sim$	✓	✓	✓
bit stuffing required	✓	$\sim$	$\times$	$\times$	$\times$
required signal levels	2	2	3	2	2

Baud-rate vs. Bit-rate

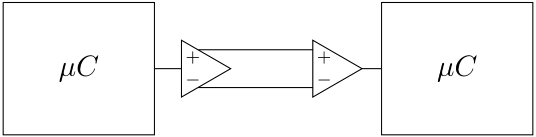
What is the Baud-rate?

Baud-rate is the number of 'signal changes' on the wire.
Thus, the ratio between the baud-rate and the bit-rate gives the bandwidth utilization.
Note that overhead due to the transmission protocol counts also towards the baud-rate, and thus reduced the bit-rate.

Example

A wire allows a baud-rate of 500 kbit/s.
Manchester coding has a bandwidth utilization of $^1/_2$, the maximal bit-rate achievable is 250 kbit/s.
NRZ, without bit stuffing, has a maximal bit-rate equal to the baud-rate.

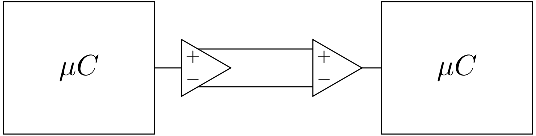
Differential Signaling



Functionality

Uses two wires to transmit data, which is encoded in a voltage difference between the two.

Differential Signaling



Functionality

Uses two wires to transmit data, which is encoded in a voltage difference between the two.

Warning

Although only two wires are required for the communication, without a common ground (or a third wire), the potential difference may lead to voltages outside the maximum ratings of the receiver.

Differential Signaling

Differences and Benefits

- Usually twisted-pair cables:
  - More resilient to noise.
  - Produces very low electromagnetic interference.
  - No shielding necessary.
- Can detect open/short cable.
- No common ground needed (in certain scenarios).
- No clock regeneration $\Rightarrow$ 8b/10b encoding or similar techniques are required.

General Warning

A word of warning

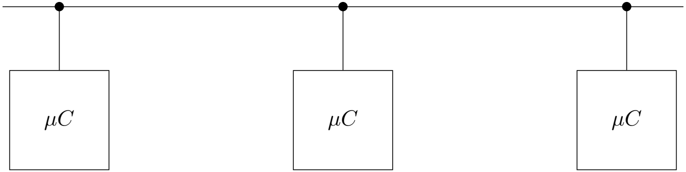
Whenever two distinct, unsynchronized clock domains exchange data, the possibility of metastability is non-zero!

Network Structure

We have spoken about how to transmit the data . . .

but how can we connect multiple components to allow communication among them?

Bus



- (Single) shared medium
- How to control access/priority?
  - Master/Slave
  - Arbitration
  - Collision detection
- Transfer rate limited by length of bus and signal propagation

Ring

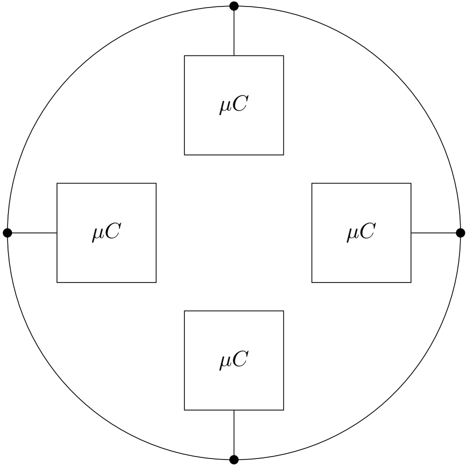


Figure: Logical View

Ring

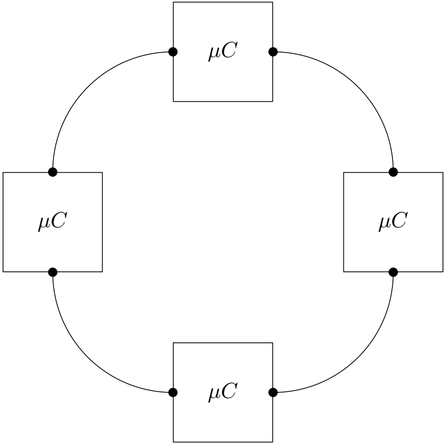
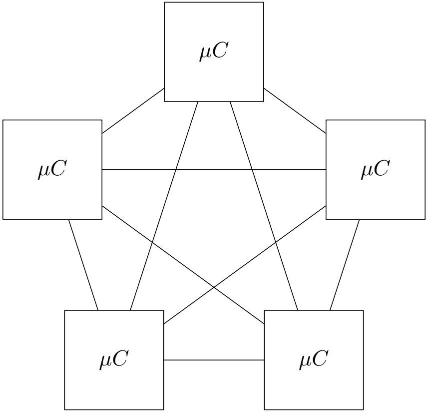


Figure: Physical View

Ring

- Network structured as a ring
- Adds a second, redundant, path
- How to control access/priority?
  - Token based (Token Ring)
  - Timeslot based
- Prevention of cyclic transmission/propagation of a signal required!

Mesh



Mesh

- Add multiple redundant paths, up to a completely connected graph
- Attention: there is a difference between a mesh structure in wired and wireless networks!
  - In wired network the components have to be connected with cables.
    Highly redundant network, but high costs
  - Wireless networks use inherently broadcasting to all reachable neighbors. Therefore mesh in wireless networks is used to cover a large area by repeating messages received; but there is no guarantee that anyone received a message sent.
- Routing of a message is complex:
  - Handle faults
  - Message loss (wireless)

Protocol Categorization

Properties of communication protocols

- How is the access to the medium controlled? Master/Slave, collision avoidance, collision detection, . . .
- Can a slave initiate/request communication?
- Is the communication half or full duplex?
- Is the data transferred in parallel or serial?
- Explicit clock transmission – synchronous/asynchronous clock?
- Data rate, maximum cable length?

Short, Incomplete, Overview of a Few Protocols

For in-depth information take a look at

- the according standards or
- the manuals of the used hardware components.

UART

Universal Asynchronous Receiver/Transmitter

More a schema to transmit parallel data over a serial line then a protocol.

- Multi-Master/Slave bus
- Arbitration depends on the underlying hardware used!
- Can be half or full-duplex, depending on hardware wiring and support of the used chip.
- Serial data transmission
- Universal $\Rightarrow$ data format and transmission speeds configurable

UART

Frame Format

- Idle state of the bus is high
- Layout of a UART frame

Start bit	Low logic level to distinguish from idle state of the bus.
Data bits	Typically 5–9 bits are supported by the hardware.
Parity bit	Optional, even or odd parity.
Stop bits	1 or 2 stop bits, high logic level.

- The usual notation used to describe a frame format is as follows: baudrate, data bits, parity (N, E, O), and stop bit.
  E.g. 9600 baud with 8 data bits, no parity, and 1 stop bit are written as 9600 8N1.

UART

- Receiver oversamples the incoming signal (typically 8 times) to detect signal changes.
- The ATmega1280 manual, Section 22.7, is quite extensive on this topic!

USART

Universal Synchronous/Asynchronous Receiver/Transmitter

- Same characteristics as UART
- but with an additional clock line $\Rightarrow$ synchronous
- Forces a master/slave architecture where the master provides the clock.
- Start/Stop bits are not required any more
- but control signals need to be send during idle periods.

SPI

Serial Peripheral Interface Bus

- Master/Slave, multiple slaves with separate select lines
- Full duplex
- Data is send from master to slave, and vice versa, in serial.
- Synchronous, clock provided by master
- No inherent error detection
- Not a formal standard

RS-232 – "The serial interface"

- Master/Slave between the DTE (data terminal equipment, master) and the DCE (data communication equipment, slave).
- Flow control with various handshaking lines (RTS, CTS, RTR, . . .).
- Standard defines the electrical characteristics and mechanical parameters of the interfaces/connectors:
  - Dedicated data lines for send and receive.
  - Uses bipolar non-return-to-zero with 3 to 15 V.
  - Logic one on the data lines is defined as the negative polarity.
- The data framing, error detection, data rates, . . . are not defined in the standard!
- Commonly used standard for data transmission.

RS-422

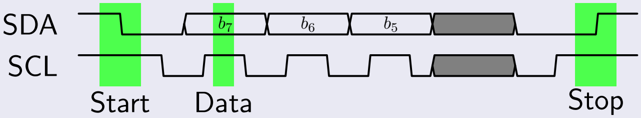
- Basically RS-232 but with differential signaling.
- Supports longer cables than RS-232 (up to 1500 m vs. 300 m).
- Faster data rates (up to 10 Mbit/s vs. 115 kbit/s).

I^2C

Inter-Integrated Circuit

- Multi-Master/Slave shared bus, with arbitration due to dominant bus state during the sending of the address.
- Serial communication, MSB first
- Two wires: data (SDA) and clock (SCL); both have a pull-up
- Typical data rate of 100 kbit/s (depending on used standard revision/mode up to 3.4 Mbit/s)
- Short range (a few meters at most).
- Slaves can "stretch" the clock, and thus slow down the transmission speed, by keeping the clock line low.

Message Format



1. Start Bit
2. Followed by 7 address bits and a $R/\overline{W}$ bit
3. ACK from the addressed device(s) (SDA = 0)
4. A variable amount data bytes, each followed by an ACK from the receiving device.
   - To end a read, the master sends no ACK for the last byte.
5. Stop Bit

I^2C

- A data transfer is either read or write.
- What if we have some kind of indirect addressing schema?

Repeated Start

- Instead of a Stop Bit another Start Bit is send.
- No limit on the number of Repeated Starts.
- Arbitration can still be in progress during a Repeated Start!
  A state machine which can handle this is not trivial.

I^2C

How do you configure the address of a slave?

There are usually a few pins available to select a few of the lower address bits.
Problem with address clashes!

10-bit Addresses

Use 11110XX as address, and send the remaining 8 bits of the address as the first data byte.
Must be supported by the slave!

TWI

Two Wire Interface

Essentially I^2C, but with reduced, and varying, feature support including

- no clock stretching
- no arbitration
- only 7 bit addresses

In a single master environment with "dumb" slaves this is usually enough.

CAN

Controller Area Network

- Multi-Master/Slave, arbitration with dominant bus state during the sending of the message ID.
- Serial with non-return-to-zero on a shared bus. Bit stuffing after 6 consecutive, equal, bits.
- Data rate limited by bus length, between 1 Mbit/s below 40 m and 125 kbit/s at 500 m.
- Used in the automotive sector, as developed by Bosch.

CAN

Message Format

1. Start of frame, 1 bit
2. 11 bit message ID and a R/W bit
3. Control data (message length, . . .), 6 bits
4. 0-8 bytes of data
5. 15 bit CRC
6. ACK, 2 bits
7. End of frame, 7 bits

TDMA

Time-Division Multiple Access

- Up to now we had only Multi-Master protocols with arbitration.
- We cannot even guarantee a upper time bound until a message with the second highest priority is delivered!
- Using a Time-Division Multiple Access schema allows to set upper bound for every message.
- Rather complex setup, but proven to work (e.g., TTP/A)

IEEE 802.15.4

Basics

- Designed for LR-WPAN (low-rate wireless personal area network).
- Standard defines Layers 1 and 2 (PHY and MAC).
- Different protocols define layer 3.

Device types

- Differentiates between two device types:
  - full-function devices (FFD)
  - reduced-function devices (RFD)
- Each device has a 64-bit identifier
- Inside a PAN, 16-bit identifiers are used
- One FFD per network is needed (PAN coordinator)

Topology

- Star, with the central point being the PAN coordinator
- Peer-to-Peer/Tree
  - PAN coordinator required
  - FFD connect the network (forwarding of messages not defined on this protocol level!)
  - RFD only as leafs

ZigBee

Basics

- Based on IEEE 802.15.4
- Defines three devices types
  - ZigBee Coordinator (PAN coordinator)
  - ZigBee Router (FFD which forwards messages)
  - ZigBee End Device
- Defines also profile for applications and security.

Conclusion

- There is no “one fits it all”.
- Selection of the used system depends on the
  - environment
  - distances needed to be covered
  - required speed
  - allowed costs
  - available resources
  - required reliability (detection of transmit errors)
  - ...

In short: it depends on the application.

Questions?