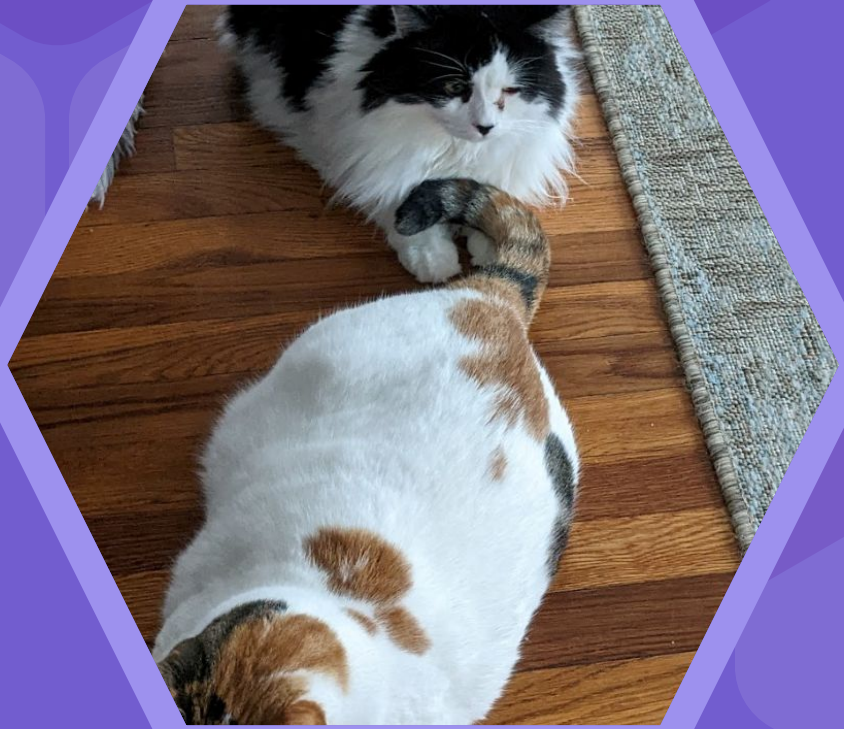
22: Communication protocols

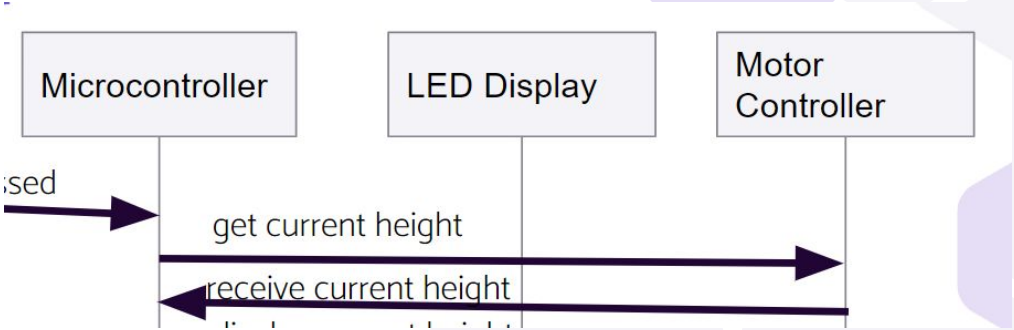


Recovering from issues

Say some sort of collision happens (possible in persistent CSMA)

- How do receiver(s) detect the garbage data?
- How do sender(s) know if their message was accepted or not?

Receiver sends acknowledgement (ACK) or other response back to sender
Built in to protocol (I2C – Friday) and/or engineered in high-level design



Summary: issues w/ communication

- Time synchronization
  - Approaches: Cristian's/Berkeley's algorithm, RZ, (+ other solutions on Fri)
- Collisions
  - Approaches: with controller (polling, TDMA) or without (token ring, CSMA)
- Signal integrity
  - Approaches: differential drivers (hardware), checksums, inserting "edges"

Specific protocols

What a message looks like:

Start bit(s)	Header	Data	Error detection	End bit(s)

Serial protocols: message sent as a sequence of bits on one wire

UART

Start bit: 0	Data (7-9 bits)	Parity bit	End bit: 1

Universal Asynchronous Receiver-Transmitter

Two components communicating

Each has transmit (TX) and receive (RX) line

Do not need synchronized clock (just both components at same frequency)

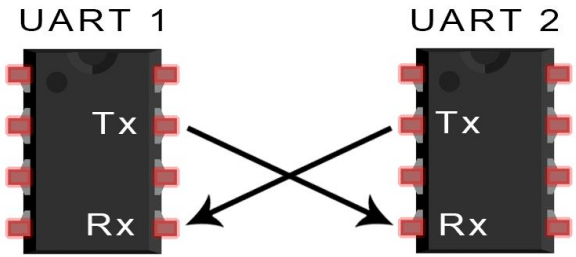


Image source

Problems with UART?

SPI

Serial peripheral interface

Controller sends clock to peripheral and transmits with clock

Transmits clock for longer so peripheral can respond

Multiple peripherals: chip select line

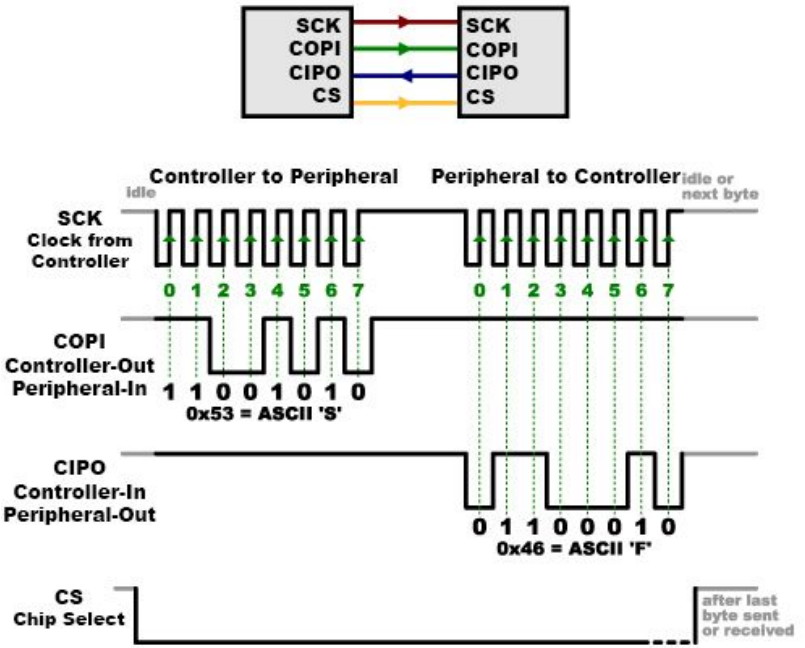


Image source

SPI: one controller, multiple peripherals

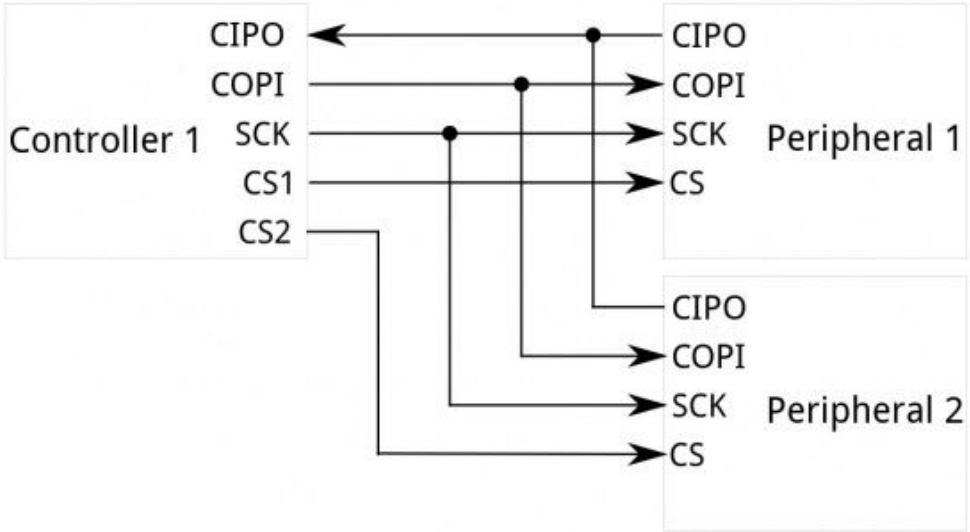


Image source

Problems with SPI?

I2C

Start bit	Address (7 or 10 bits)	R/W bit	Data (8 bits)	ACK bit	Data (8 bits)	ACK bit	...	End bit

Inter-integrated circuit

Controller uses address to select which peripheral it is communicating with

Timing of SDA/SCL means this protocol supports multiple controllers

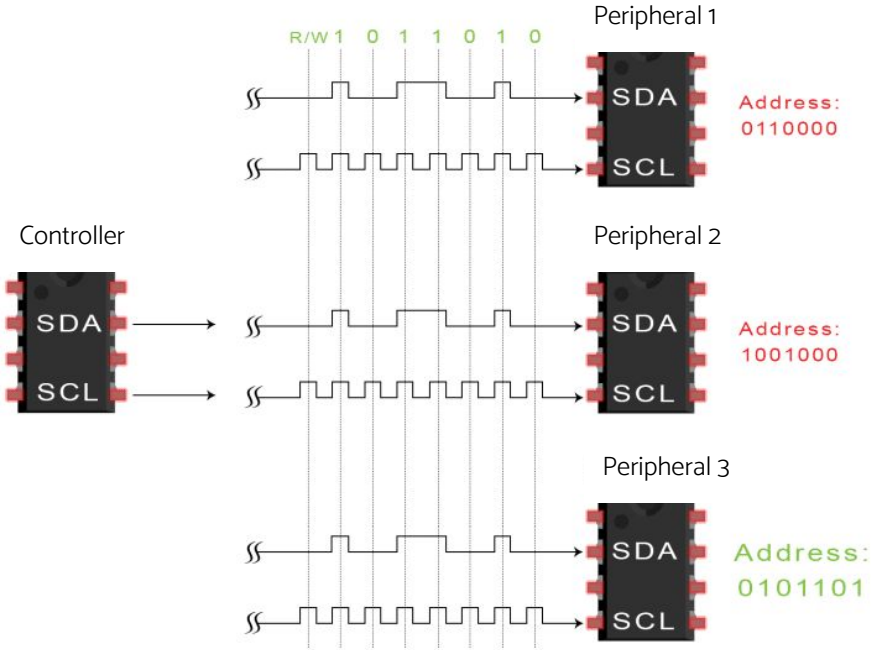
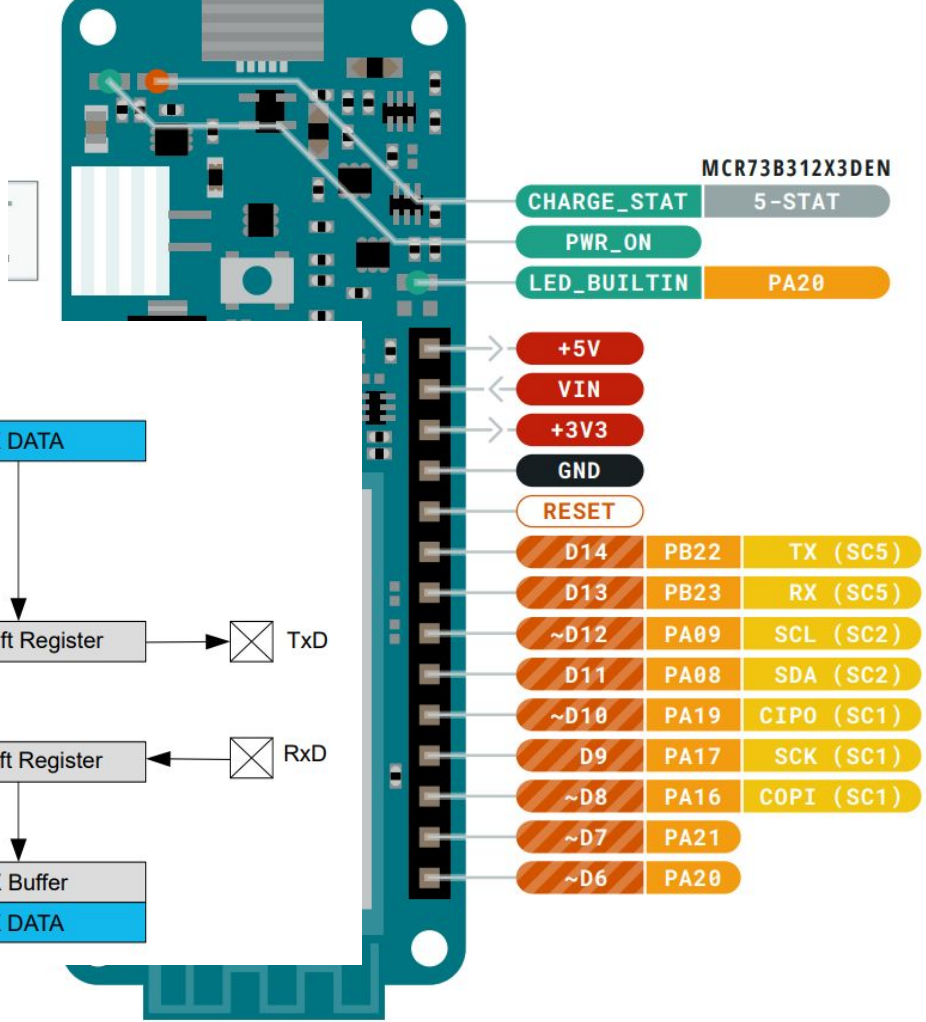


Image source

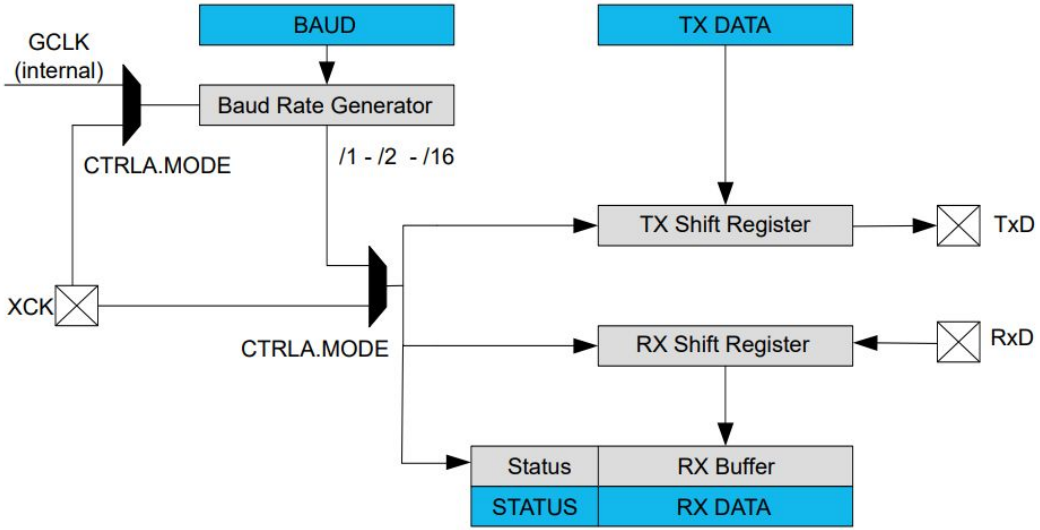
Problems with I2c?

Hardware support



Block Diagram

Figure 26-1. USART Block Diagram



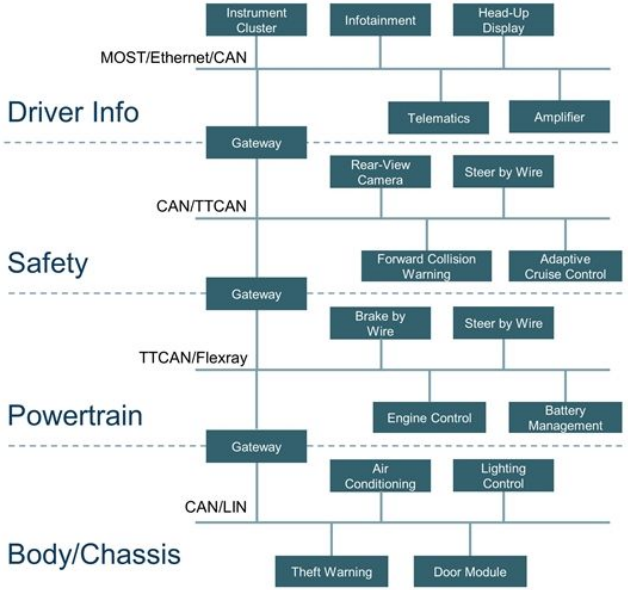
UART/SPI/I2C summary

	Send clock?	# of devices	control
UART			
SPI			
I2C			

Modern Vehicle Electronics Architecture

Visteon

- **Four different computing domains**
  - Vastly different software in each domain
- **Large number of Electronic Control Units (ECU)**
  - 30-150 ECUs in cars today … and growing
- **Large software code base**
  - 100+ million lines of code in premium cars



Modern car is an increasingly complex network of electronic systems

5

Image source

CAN

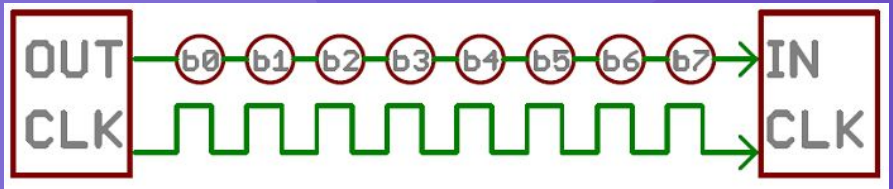
Start bit	CAN-ID (29 bits)	Transmission request bit	Control (6 bits)	Data (0-8 **bytes**)	CRC (16 bits)	ACK (2 bits)	End bit

Controller Area Network

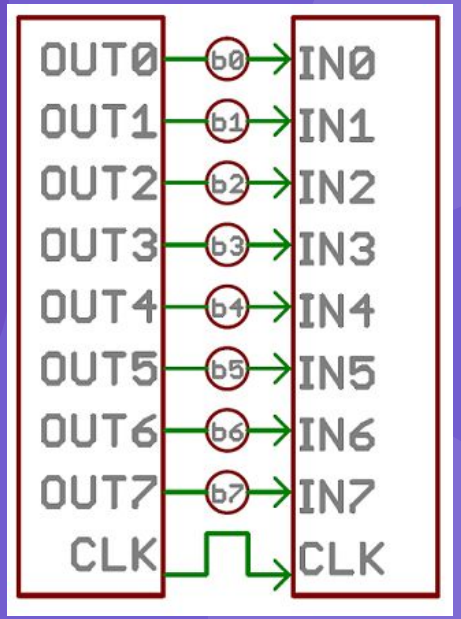
Used for safety-critical applications (cars)

Binary countdown for arbitration

Add edges with bit stuffing



We have discussed serial buses. Why are parallel buses challenging?



(image source)

Keeping track of data - buffers

Way for main process and transmitter/receiver to produce/consume data at different rates

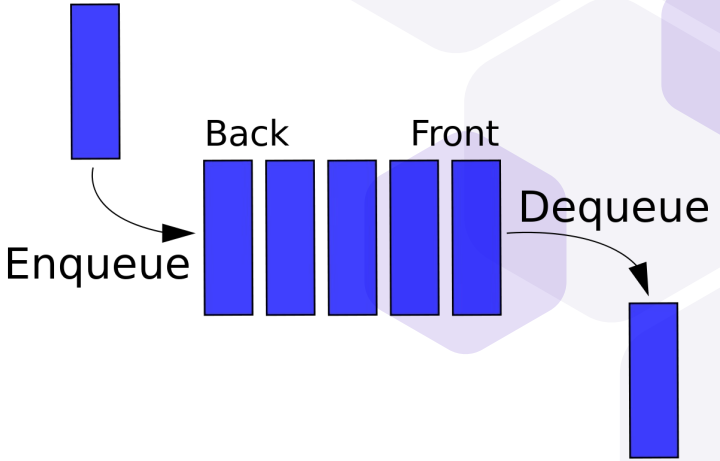
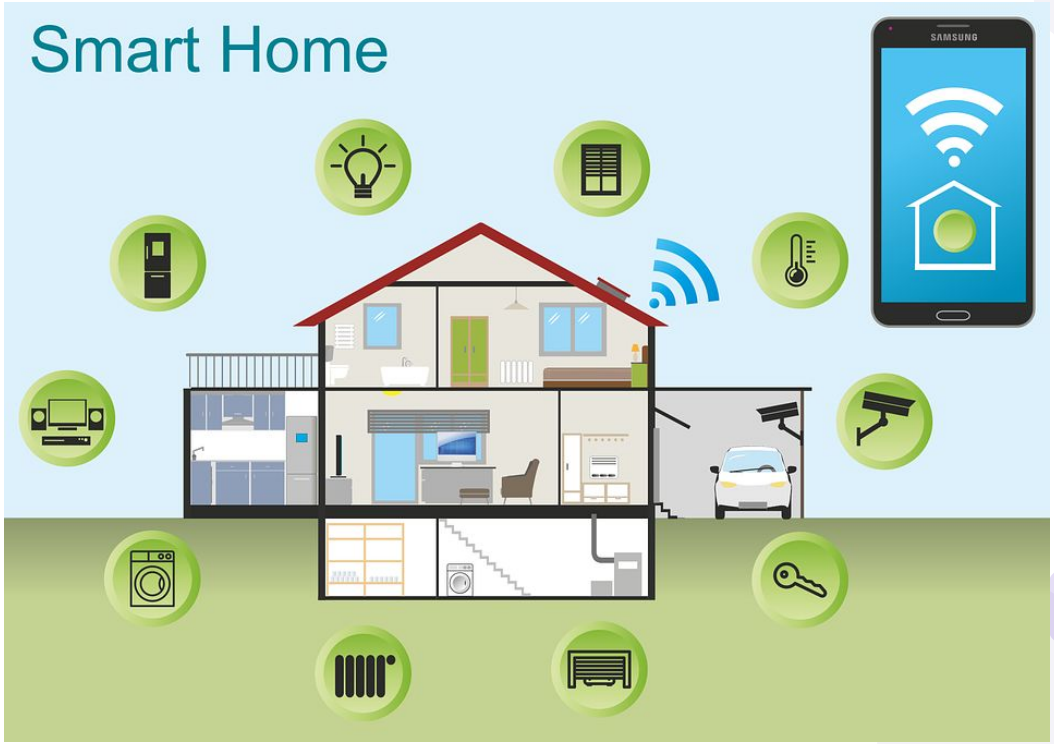


Image source

Wireless communication



What are some concerns/considerations that are specific to wireless communication?

Wireless communication

Image source

WiFi: Internet connectivity (complex and layered)

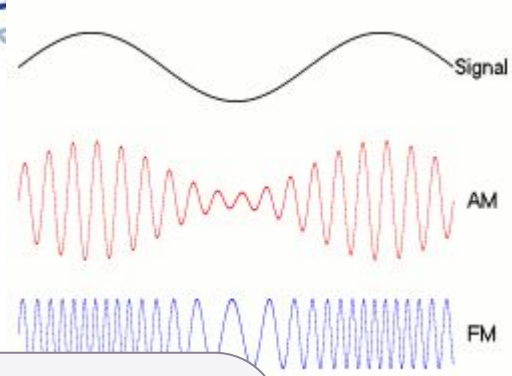


Image source

RFID: Active reader sends radio signal
Passive tag has small amount of data

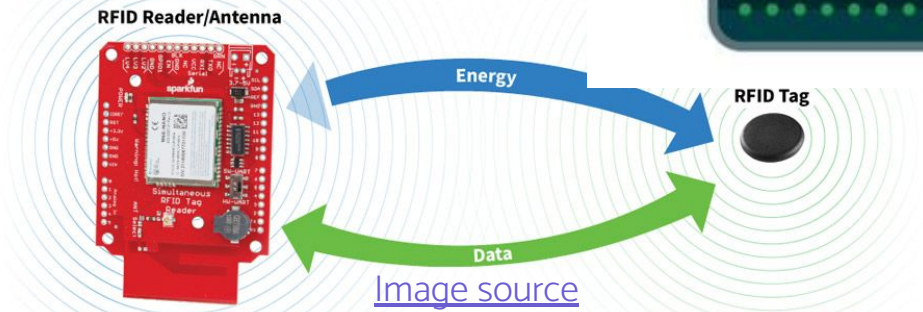


Image source

Bluetooth: scatternets/piconets, frequency-hopping on FM