Jacobs University Bremen

SCADA and Communication Protocols

Automation

Course CO23-320203

Course admin

- Lab lecture – 10 May
- Course final – 23 May (tentative)

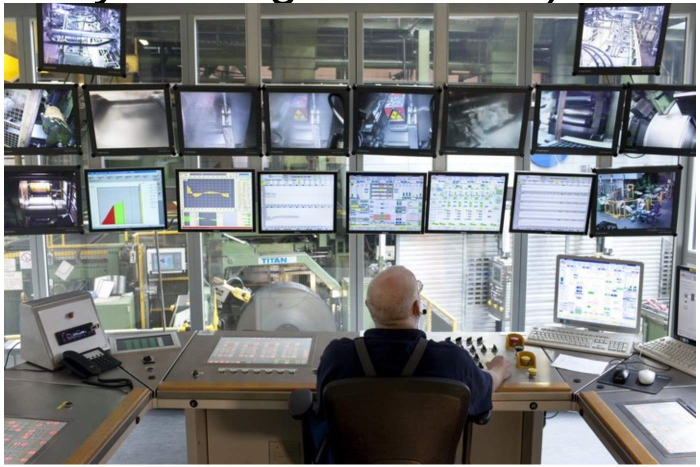
  Potential solution: moving the exam to Tuesday 21 May at 9:00

Plant automation

- Sensors, actuators and control systems (such as PLCs) are a part of a bigger picture
- The picture evolves with time but most automation principles stay valid

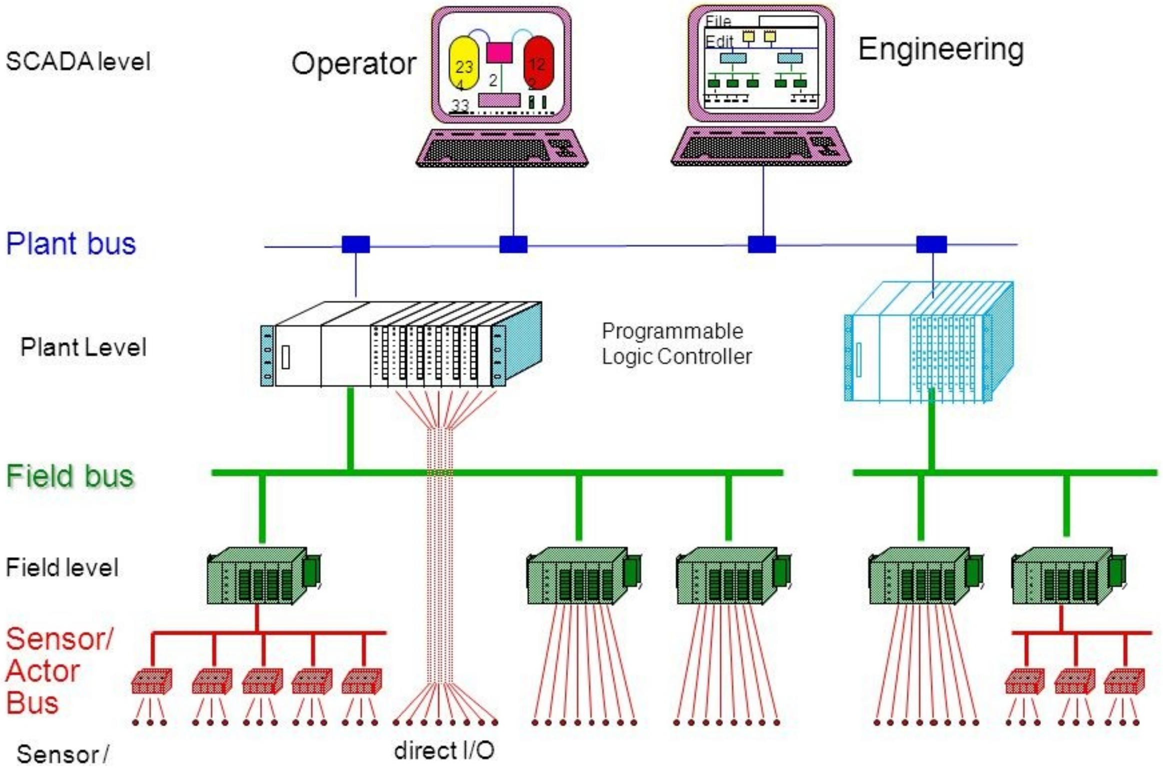
  (example below: steel making control room 40 years ago and now)

[Source: https://i.pinimg.com/originals/4a/6b/58/4a6b5806861f6a7a5c765b0ce21f531e.jpg]



[Source: https://www.sms-group.com/fileadmin/_processed_/9/9/csm_01_Bilstein_115f058a63.jpg]

Plant hierarchy



[Source: http://slideplayer.com/2272594/8/images/3/Location+of+the+field+bus+in+the+plant+hierarchy.jpg]

Level 4
Production scheduling

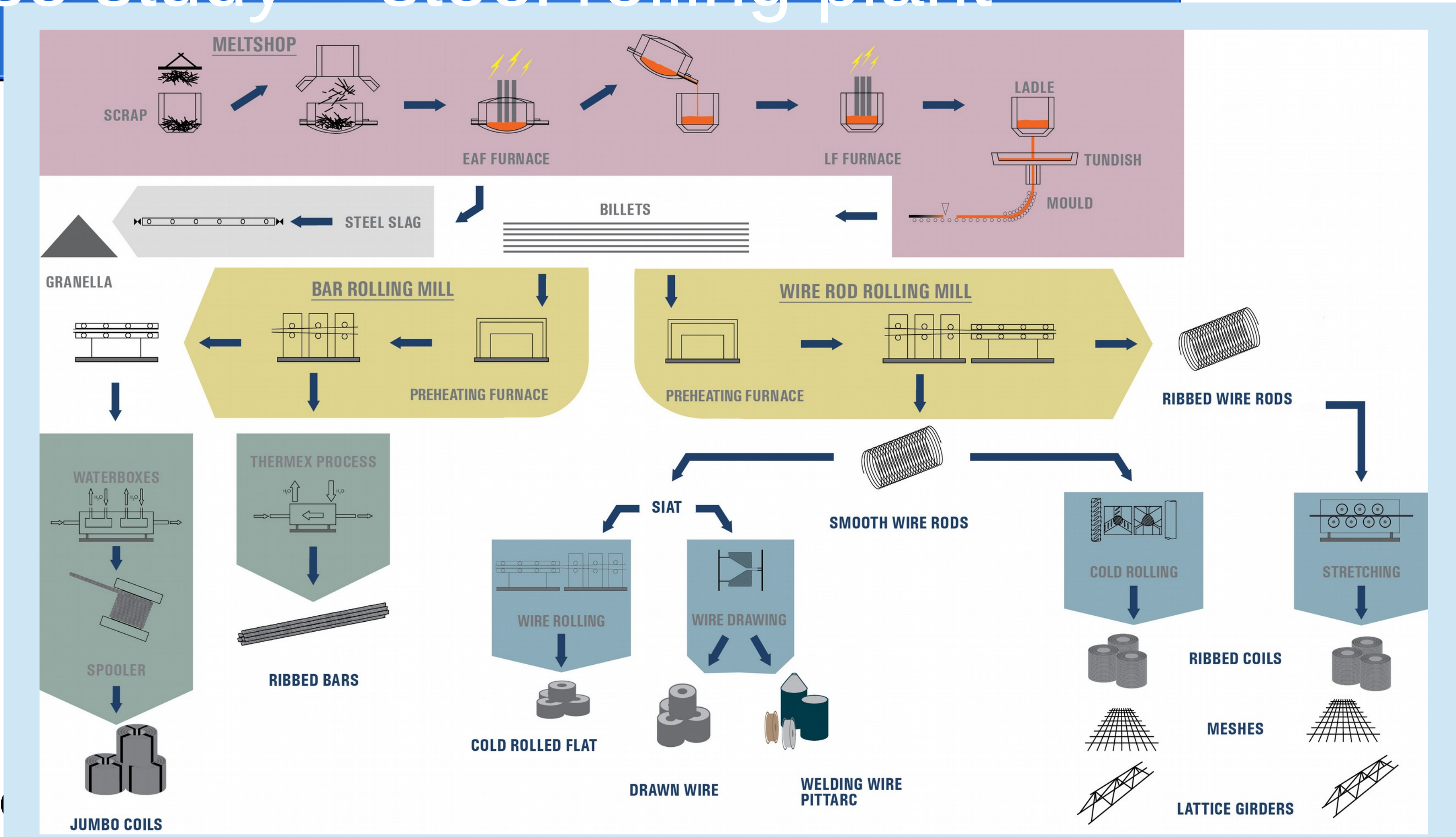
Level 3
Production control

Level 2
Plant supervisory

Level 1
Direct control

Level 0
Field level

Case study – steel rolling plant



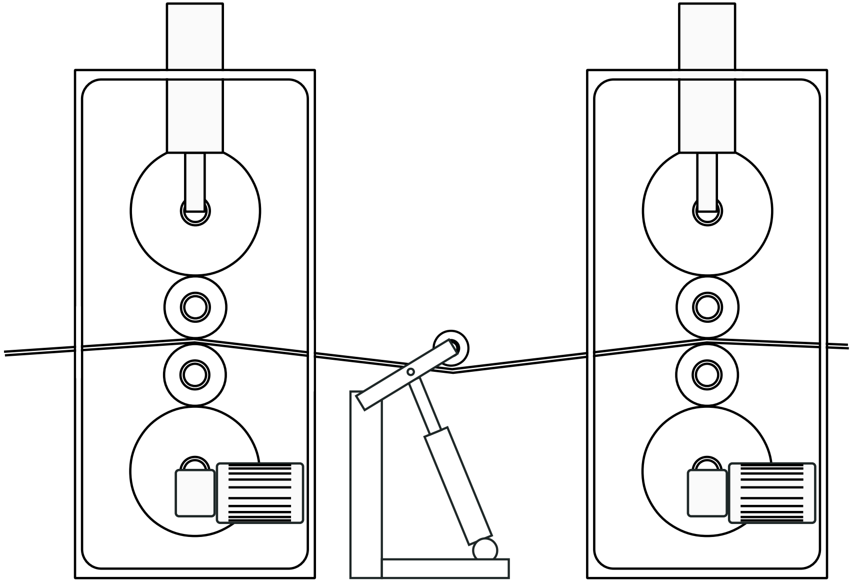
[Source: http://www.pittini.it/wp-content/uploads/schema_produttivo_PITTINI_2015.jpg]

A sub-unit: hot rolling mill

http://www.nationalbronze.com/News/wp-content/uploads/2014/11/High-performance_hot-strip_rolling_mill.jpg

A sub-unit: hot rolling mill

- Every mill element has independently controllable
  - Hydraulic cylinders
  - Rollers’ motors
- Tensioning arm between every mill
- Degree of precision
  - Fraction of a millimeter of thickness of the rolled sheet
  - At the same time: the press force causes the mills’ metal frames to stretch by centimeters
- Continuous process
- Dependent on the speed and temperature of material arriving from upstream (casting process)
- Production cannot be achieved without a local high speed coordination between the mills and the tensioning arms → Level 1 PLC probably plays this role
- Data must also be sent to the machines further down the treatment process



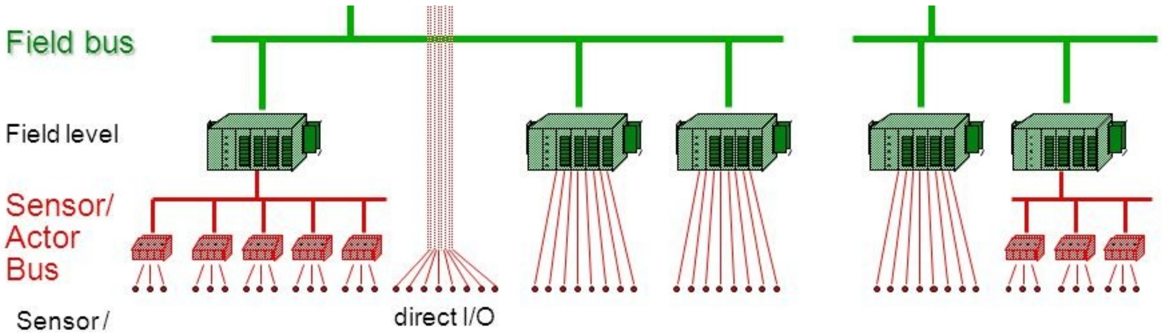
Plant control levels – Level 0 & 1

- Maintain direct control of the given plant unit
- Detect and respond to any emergency condition
- Collect information about the unit and send it up
- Provide input and output to a local operator’s HMI
- Perform diagnostics on themselves
- Update any standby systems

Control enforcement

System coordination and reporting

Reliability assurance



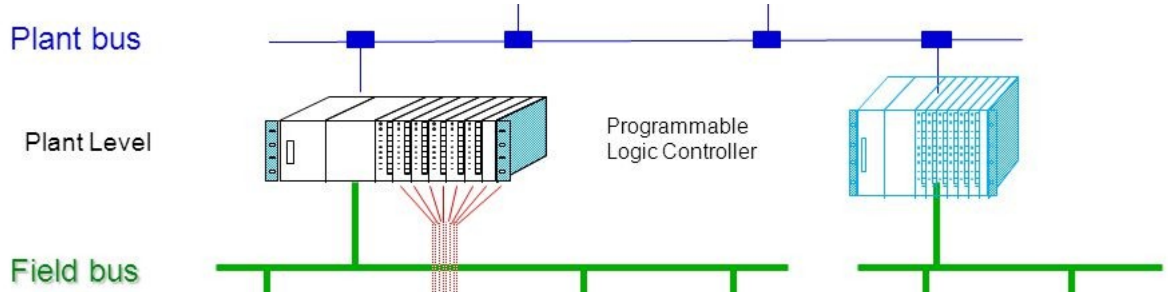
Plant control levels – Level 2

- Detect and respond to any emergency condition at the plant's region
- Locally optimise (inter)operation of units given production schedule
- Collect and maintain information about the production and send it up
- Provide input and output to the lower and higher levels
- Provide local HMIs
- Perform diagnostics on themselves
- Update any standby systems

Control enforcement

System coordination and reporting

Reliability assurance



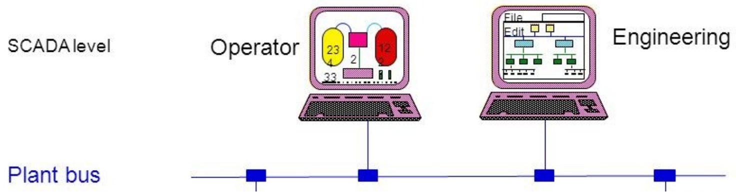
Plant control levels – Level 3

- Establish immediate production schedule (incl. transportation)
- Locally optimise the costs of individual production area
- Make area production reports
- Use and maintain area practice files
- Track inventory, personnel, raw materials and energy usage
- Maintain communication with lower and higher levels
- Operation data collection and off-line analysis
- Provide HMI for the local plant
- Perform diagnostics on themselves *and the lower levels*

Control enforcement

System coordination and reporting

Reliability assurance



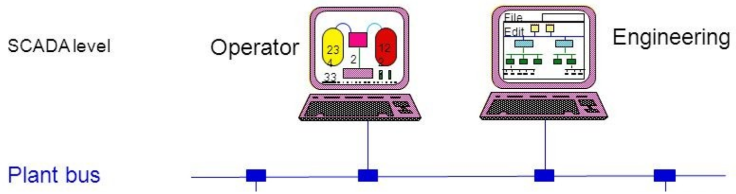
Plant control levels – Level 4

- Establish and globally optimise plant production schedule
- Determine the optimal inventory levels
- Make area production reports
- Interface with Purchasing (inventory), Accounting (energy use), etc.
- Store inventory, personnel, raw materials and energy usage files
- Collect quality control information
- Maintain communication with the lower level
- Provide HMI to the Plant Management
- Perform diagnostics on themselves and the lower levels

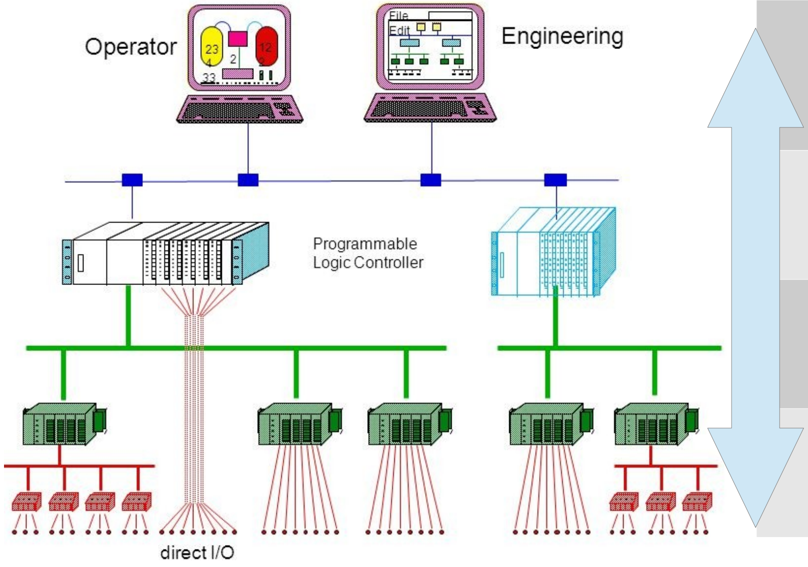
Control enforcement

System coordination and reporting

Reliability assurance



Plant hierarchy 2



Area controlled	Control time scale	Failure consequence	Interface for ...	Data production	Data storage
Plant(s)	Hours, days	Financial and scheduling problems	Management and engineering	Low	High
Single machine	Real time, milliseconds	Catastrophic failures, loss of life	Technician, engineer	High	Low

SCADA

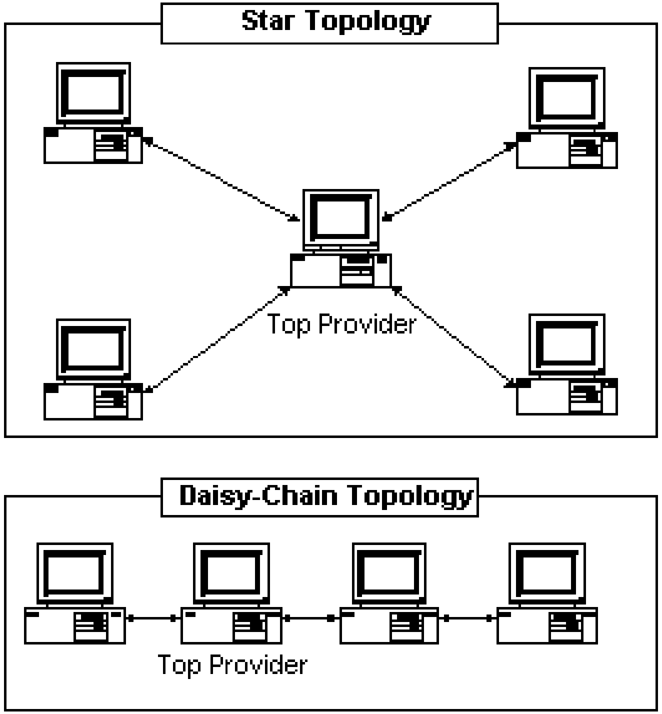
- Supervisory Control and Data Acquisition (SCADA) is an umbrela term for an integration of several technologies
- It is not necessary geographically in one place, it can unify geographically remote terminals through telemetry
- The plant hierarchy pyramid is heavily interconnected

Exchanging data

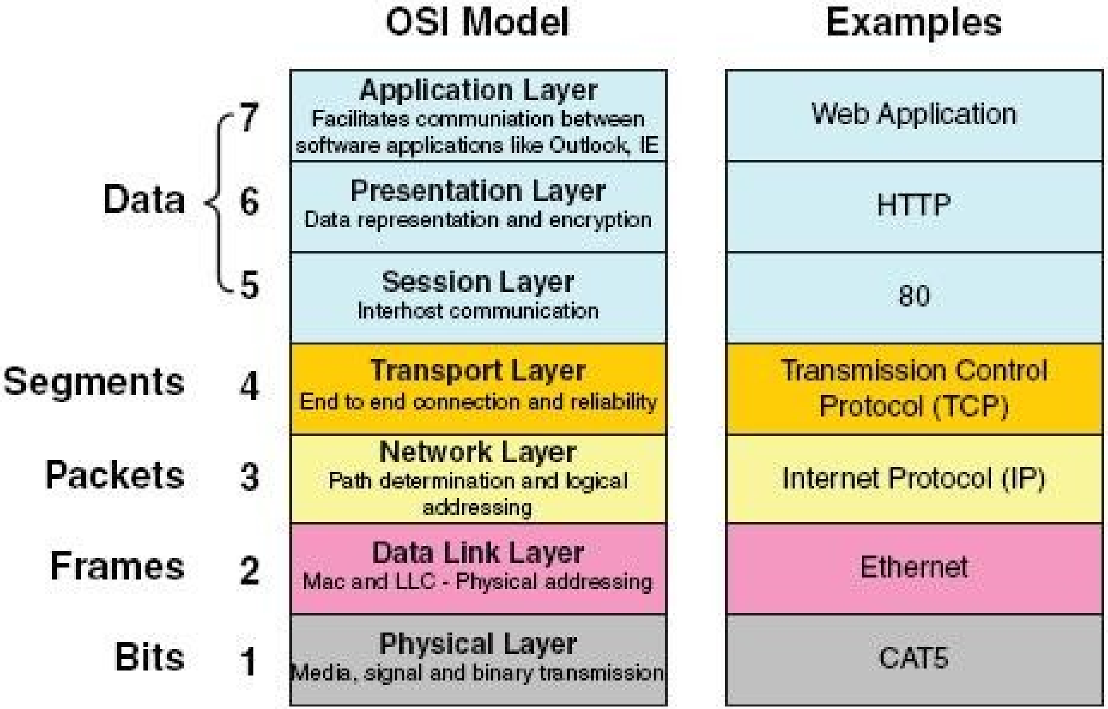
- Connectivity: where?
  - Sensor to controller (sensor bus)
  - Controller inter-connections (field bus)
  - Controllers to SCADA (plant bus)
- Industrial fieldbuses
  - ...why not just use Ethernet?
    - Determinism
    - Real-time requirement
- What must be defined?
  - Hardware (example: RS-232)
    - Caution: plenty of aspects - wiring and plugs, voltage levels, sensing
  - Protocol (example: Modbus)
    - Caution: there exist different layers of protocols!
  - Hardware + protocol (example: Profibus)
- … Wikipedia lists more than 80 communication protocols dedicated to automation!

Profibus plugs and cables
[Source: https://upload.wikimedia.org/wikipedia/commons/thumb/8/8a/0x-pb-stecker-verschieden.jpg/1280px-0x-pb-stecker-verschieden.jpg]

Networking – basic notions



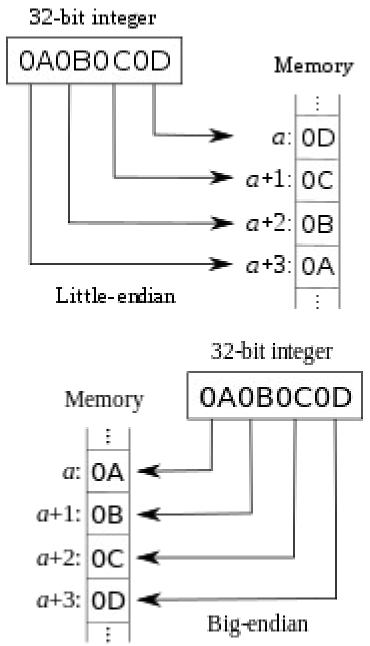
[Source: https://www.info-it.net/images/IC184860.gif]



[Source: http://tri-tel.com/wp-content/uploads/sites/94/2013/10/osi_model.jpg]

Networking – basic notions

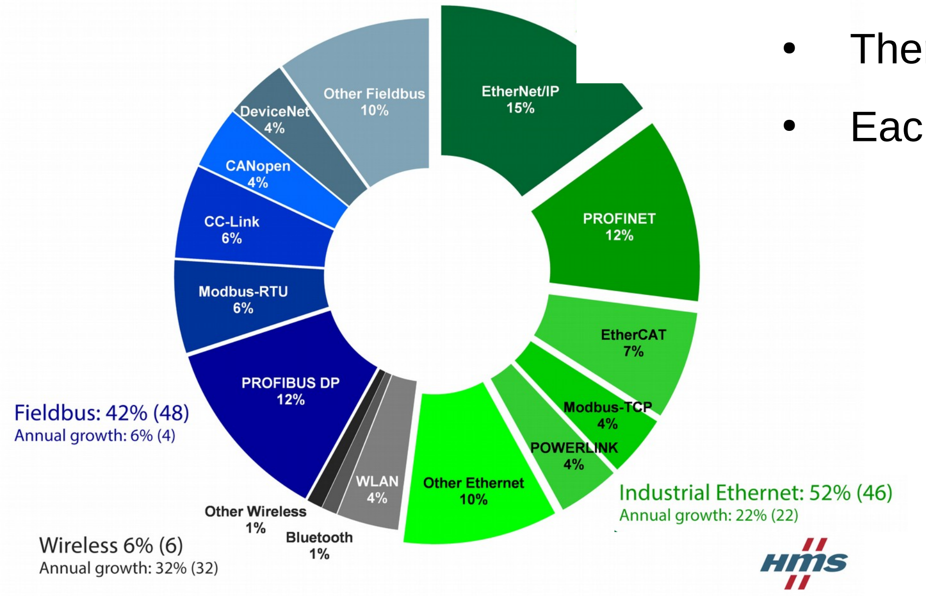
- Encoding
  - ASCII – slower but easy to debug
    - Example: NMEA “$GPGSV,3,2,09,07,77,299,47,11,07,087,00,16,74,041,47,20,38,044,43*73”
  - Binary (example: Modbus RTU) – can be efficient but must come with conventions
    - Variable description: signed/unsigned, length, type
    - In which order are the data bytes stored?
      - Little Endian = LSB (least significant byte) first, MSB (least significant byte) last
      - Big Endian – the opposite
      - Mixed schemes are also possible
    - Binary strings? Even more convention trouble: ASCII, UTF, ...
- Who controls the connection?
  - Peer-to-peer (practically not used in automation)
  - Master – Slave (master initiates all exchanges and controls the bus)
  - Client – Server (equivalent newer term coming from the IT industry)
    - Client = Master, Slave = Server
    - In newer protocols, servers are sometimes authorized to initiate an exchange
  - Cloud computing (can be used at top levels of automation pyramid for data storage)



[Source: https://en.wikipedia.org/wiki/Endianness]

Industrial buses – the big picture

- The application often decides which data bus is chosen for the given project – most have strong points in certain domains (e.g. high speed data transfer for industrial vision or low latency)
- Many proprietary solutions move to open standards to increase their reach



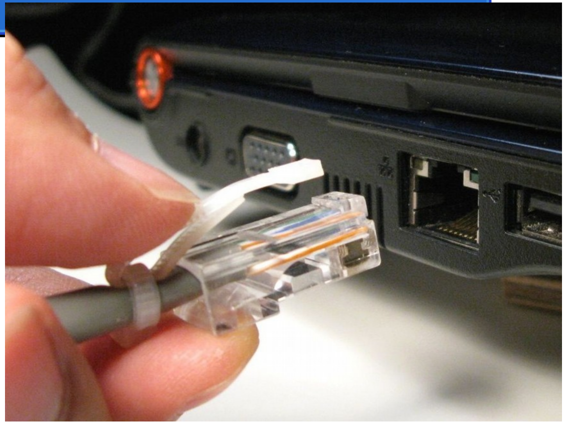
2018 market share of fieldbuses and industrial Ethernet
[Source: https://www.anybus.com/images/librariesprovider7/default-album/network-shares-according-to-hms-2018.jpg?sfvrsn=aedb9dd6_0]

- There is a huge number of competing brands
- Each might define different element(s) of the protocol

[Source: http://www.china-pilz.com/uploads/allimg/180108/1-1P10R20244M9.jpg]

Networking – everyday examples

- Automation networks may have similar elements with the commonly used communication interfaces
  - Popular standard plug examples:
    RJ45 (Ethernet) and DB9 (Serial port)
  - It does not mean they use the same protocol!
- The automation protocols often reuse a restricted set of definitions of lower layer communication from common standards
  - Serial communication
    - RS-232, RS-422, RS-485
    - Daisy chaining is possible, no star connection schema
  - Ethernet
    - Star configuration through switches/hubs
  - Wireless
    - Not so common in automation but rising in popularity



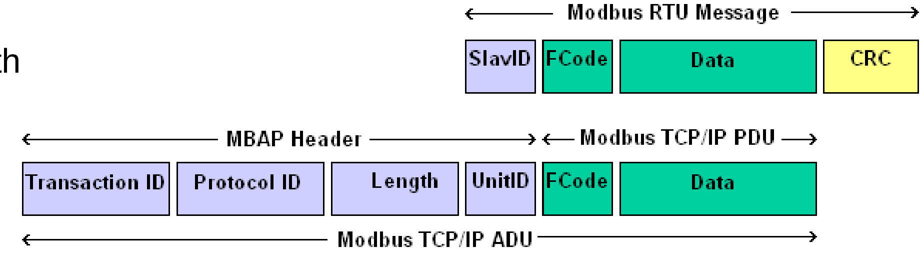
RJ45 Plug [Source: https://cdn.instructables.com/FXZ/EHBK/FZHLAEHA/FXZEHBKFZHLAEHA.LARGE.jpg]

DB9 Plug [Source: http://www.innovatic.dk/jpg/serialPlug.jpg]

Example 1: Modbus

- Developed together with the MODICON PLCs
- Today: open standard
- Three variants:
  - ASCII (a way to encode values as readable characters)
  - RTU (binary coding + CRC checking)
  - TCP/IP (using encapsulated RTU encoding or MBAP (Modbus Application Header))
    - Can have multiple Masters
  - In any case: data packets (or ADUs – Application Data Units) with
    - Slave ID
    - Function code
    - Register to write to / read from
    - Data values
    - Cyclic Redundancy Check (CRC)
- Can be used over a RS-xxx or Ethernet connection
- One client *(master)* issues commands, servers *(slaves)* respond

[Source: http://c3.chipkin.com/assets/uploads/2017/dec/01-18-21-56_modbus-Logo.png]



Modbus packets: RTU and TCP/IP with MBAP
[Source: http://www.simplymodbus.ca/images/adu_pdu.PNG]

Modbus

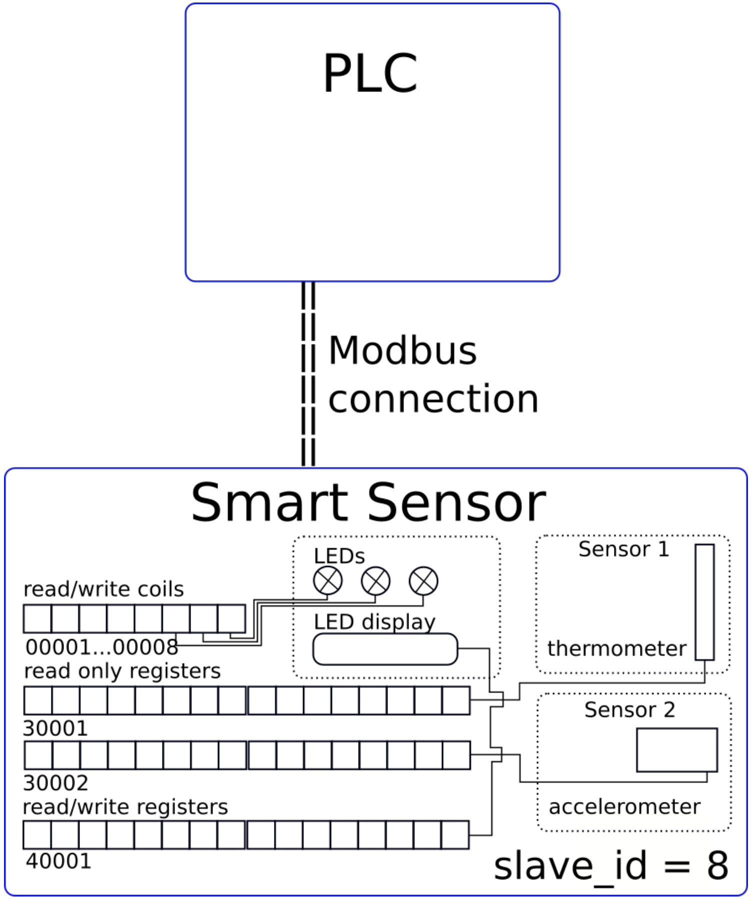
- Data tables – addressing hardware memory registers
  - Every device can have many different registry types with specific functions
  - Modbus protocol reserves specific address ranges for referring to them
  - Coils (1 bit data, BOOL) – example: on/off relays or valves, indicator light
    - 00001 – 09999 (read/write)
    - 10001 – 19999 (read only)
  - Registers (16 bit values, WORD) – example: sensor data or control setpoint
    - 30001 – 39999 (read only)
    - 40001 – 49999 (read/write)
    - Splitting over several registers is obligatory for longer data types (e.g. FLOATs, DINTs = DOUBLE INTEGERs)
    - but: Modbus protocol does not define in which order bytes are sent!
- Function codes
  - Example: 1 = *Read Coil Status*, 3 = *Read Holding Registers*,
16 = *Multiple Register Write*

Modbus example

- Example function codes
  1 – read coil status
  2 – write coil status
  3 – read register
  4 – write register

- Example packets
  (values of the fields in Modbus RTU packet, not actual binary values sent)

Switch LED 2 on SlaveId = 8, Fcode = 2, Data = {00007, 1}, CRC = xxx
Read accelerometer measurement SlaveId = 8, Fcode = 3, Data = 30002, CRC = yyy
Display value 116 on the LED display SlaveId = 8, Fcode = 4, Data = {40001, 116}, CRC = zzz



Modbus CRC – code

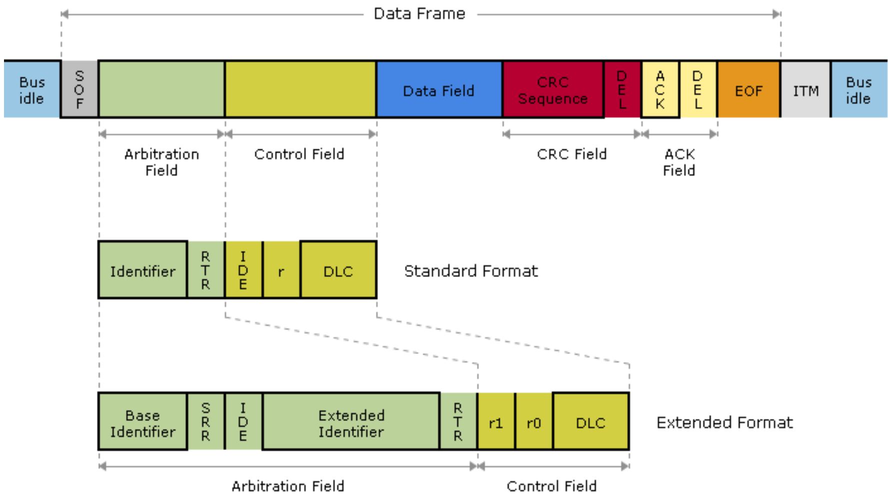
```c
// Compute the MODBUS RTU CRC in C
UInt16 ModRTU_CRC(byte[] buf, int len) {
  UInt16 crc = 0xFFFF;

  for (int pos = 0; pos < len; pos++) {
    crc ^= (UInt16)buf[pos];          // XOR byte into least
                                      // significant byte of
crc
    for (int i = 8; i != 0; i--) {    // Loop over each bit
      if ((crc & 0x0001) != 0) {      // If the LSB is set
        crc >>= 1;                    //  Shift right
        crc ^= 0xA001;                //  and XOR with 0xA001
      }                               //  = 0b1010000000000001
      else                            // Else (LSB is not set)
        crc >>= 1;                    //  Just shift right
    }
  }
  // Watch out for the order of the low and the high byte
  return crc;
}
```

- CRC is a method to ensure the integrity of the transmitted data
- It is applied to the entire packet body (SlavID + FCode + Data)
- The receiver recalculates it and compares to the received value. Identical value = no error
- This code calculates the 2-byte CRC value for any value stored in the buffer **`buf`** of length **`len`**
- The generating polynomial is **`0xA001`**
- Efficient bitwise implementation with bit shifts (**>>**) and XOR operation (**^=**)
  - **`crc ^= 0xA001`** ≡ **`crc = crc ^ 0xA001`**
  - 0^0=0, 0^1=1, 1^0=1, 1^1=0
- This code respects Modbus convention of sending the LSB first

The source specification:
”Modicon Modbus Protocol Reference Guide”
modbus.org/docs/PI_MBUS_300.pdf

Example 2: Controller Area Network (CAN)

- Conceived by BOSCH. Today used in automotive, marine, elevators, medical, military and machinery control
- CANbus – data + physical layer definitions, CANOpen (Network layer + above) – protocol definitions for CANbus
  - CAN 2.0A – 11 bit messages
  - CAN 2.0B – 29 bit messages (longer addressing and data lengths possible)
  - CAN FD – flexible length
- Frame:
  - Arbitration = addressing
    - Message priority (some packets are more urgent than the other)
    - Is it Data Frame or Remote frame (= message request; slaves can also request from slaves)
  - Control (mostly message length)
  - Data
  - CRC
  - Acknowledgement field
  - End of frame (fixed bits)
- Speed vs Line length trade-off – examples:
    - 125 kbs – 500m cable link possible
    - 1Mbs – 40m maximal cable length
    - 15Mbs – 10m maximal calble length (FD only)
- Extension of CAN: SAE (Society of Automotive Engineers) J1939
  - defines data formats, messages and diagnostic flags, specific to vehicles



Standard (11-bit) and extened (29-bit) CAN packets
[Source:https://elearning.vector.com/portal/medien/vector_elearning/flash/can_v1/chapter_3/EN/CAN_3.2_GRA_StandardExtendedDataFrame_EN.png]