# Manhattan 2 Smart Building Interconnection-Standards Development Initiative

By Glenn Weinreb, CTO, Manhattan 2, Printed Nov 6, 2020

## 1 Introduction

Manhattan 2 (Ma2) intends to develop electrical, mechanical, and communications standards that define how devices interconnect within the building of the future. Devices include motors that control thermal covers over physical wall windows, motors that control curtains and blinds, fans in ducts, dampers in ducts, lights, occupancy sensors, washing machines, ovens, refrigerators, dish washers, HVAC systems, pumps that control 58°F ground source water, thermal storage water, valves on room water-filled radiators, etc.

This initiative involves developing a 2-wire communication standards between multiple devices that provides the following features: supports CANbus communication between ~$1 microprocessors, no damage upon accidental short to power wires, devices use little power when not in use (sleep), wiring supports tree topology (daisy chain not required), hot socket compatible (no damage when attach wires with power on), ≥ 99.999% reliable (not wireless), and transceivers consume ≤ ~10mW of power when signaling (as opposed to ~10x more utilized by RS-485 or 120Ω CANbus).

We are calling this new network “BuildingBus™”, for lack of a better term, and there are two versions. *BuildingBus 48V* caters to lower power and lower voltage whereas *BuildingBus AC* caters to higher power and higher voltage. *BuildingBus 48V* routes 48VDC power to devices with ~200W/network; whereas *BuildingBus AC* routes 110/220VAC power with ~2,000W/network to devices. Fans, industrial lighting, and motors that move heavy windows require 110/220VAC; whereas many other devices are 48V capable. 110/220VAC cable is bulky and needs to conform to high voltage building codes (i.e. conduit more likely), whereas 48VDC cable is lighter and satisfies low voltage building requirements. For details, search for "BuildingBus Development Initiative" in file Active Window.

We make use of existing standards whenever possible, and propose new standards as needed.

Researchers do *not* necessarily design products to be manufactured and sold. Instead, they propose interconnection standards, and prototypes that demonstrate those standards. These are then provided to standards bodies (e.g. IEEE), which modify as desired, and establish plug-and-play standardization.

All materials produced by researchers are given away for free, to encourage utilization by standards bodies, to reduce $CO_2$ emissions. This includes mechanical drawings, electrical schematics and software source code.

Ma2 is also developing standards that define how solar material attaches directly to building surfaces, such as plywood. The proposed solar material is ~1.5cm thick and contains embedded electronics that perform power conversion. This material can be applied to both roof and wall surfaces, edge-to-edge.

Prototypes develop by researchers use the same processor, and utilize common code. This helps researchers move quickly. Cost reduction is a later step, done by industry, after standards are finalized. The Xmc4200 processor, for example, supports almost all devices. A prototype that moves a window thermal cover, and a prototype of the DC-DC converter embedded in solar material, can both be implemented with this one processor, for example. It provides: 16x 12bit a/d channels, 2x 12bit d/a channels, analog comparators, 2 CANbus channels, counter/timers, 256KBFlash, and 40 KB Ram. All within one tiny package.

***See Also:***

- Ma2 Smart Building Interconnection-Standards Development Initiative (Plan)
- Ma2 Active Window Development Initiative (Plan, Research Teams Yr1)
- Ma2 BiPV/LiPV Rollable Photovoltaic Research Initiative (Plan, Research Teams Yr1)
- Ma2 Fan and Damper Interconnection-Standards Development Initiative (Plan)

## 1.1 Example Network Devices

- LED light bulb controller (on/off/dimmer control of LED's, modulated current source)
- Wall light switch (measure voltage from POT)
- Thermal cover for window (motor controller, temperature/light/humidity measurement)
- HVAC compressor motor (motor controller)
- Pump that circulates 55°F water from soil (motor controller, ground source for HVAC)
- Fan within duct (on/off and variable speed motor controller)
- Damper within in-line duct, or at vent opening (motor controller)
- Temperature sensors, Thermostat devices
- Solar PV Array Devices (manage & monitor power conversion, mppt, batteries, grid, loads)
- Passive Solar Devices (manage & monitor pumps, temperatures, pressures, sunlight)

- Wind Turbine Devices (manage & monitor power conversion, batteries, grid, loads)
- Occupancy sensor
- Water radiator controller (resides in room, valve control, temperature/pressure sense)
- Current/temperature measurement at wall outlet
- Door position sense and control (door closer, electronic lock)
- Ground source drill head electronics (200ft below surface, measure temperatures, measure acceleration vibration while drilling)
- Ground source pump (variable speed motor, stepper motor, on/off motor, temperature/pressure measurement)
- Solar PV array (set of batteries from software's point of view)
- Solar array micro-inverters
- Solar array electronics element (power switches, temperature measurement, voltage/current measurement).
- Solar array monitor, fire control and lightning protection system

## 1.2 Example Appliance Devices

Researchers redesign home high-power appliances such as HVAC, heat pump, refrigerator, dish-washer, clothes dryer, stove and oven.

Researchers create suggested 5-wire standardized Power/Data Plug that supports power (VAC hot, neutral, earth) and ≥99.999% reliable wired data communication (wireless is less reliable).

Researchers develop suggested programmable water source standard that involves moving heat/cold via water. Appliances include input and output pipe for Programmable Water source; and valves in house routes one of the following to each appliance: hot water (DHW), domestic cold water, 55°F ground source water, and thermal storage water (hot or cold). Processors communicate which is available, and at what temperature. Solar PV panels and heat pump might heat up 500 gallons of water in basement to 140°F (low cost energy storage), and several appliances might find this helpful (e.g. oven). 55°F ground source water might be helpful in other applications (e.g. refrigerator). Researchers develop mechanical connector standard that supports thermal insulation.

## 1.3 Challenge for Engineers

We would like for our proposed low $CO_2$ technology to cost less than traditional non-automated systems; or if they cost more, they must pay for themselves by consuming less energy over a reasonable period of time (i.e. reasonable payback period).

If cost reduction is not achieved, then what can we do? "Plan B" assumes that the following occurs over time:

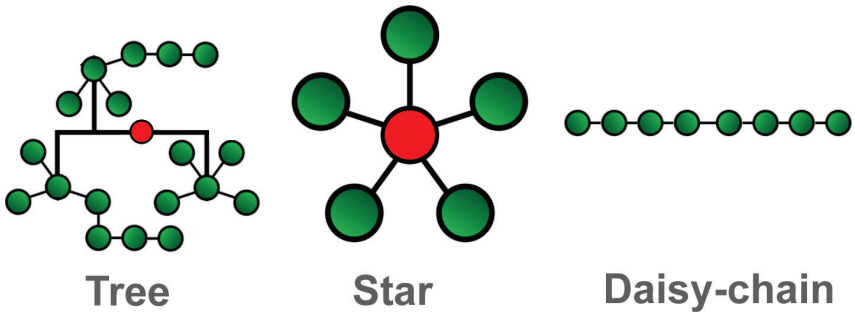
- *Evidence* of climate-change-cost to society increases. For example, Boston, much of which is on landfill, sees water *actually* rising.
- Policy changes to lower $CO_2$ emissions (e.g. new building codes, increase natural gas prices, rebates for low $CO_2$ behavior, etc.).
- Low $CO_2$ technology, developed by us and others, becomes more popular.

Plan B is not so bad, since it might take 3 to 5 years to develop quality prototypes that can be used by industry as reference designs, debug the system, work with standards bodies to write standards, and build production capacity. And it might take 3 to 5 years for society to see evidence of climate change costs and set policy that pushes people toward lower carbon technology.

In reality, some of our technology will have better payback periods than others. And those with better payback will be adopted faster. In summary, we can expect each technology to have its own adoption rate.

## 1.4 Wired IoT Requirements

Networking requirements are: 99.999% reliability, depend on low cost $1 to $3 microprocessors, wire to device, power to device, tree topology wiring, and processor sleeps when not in use (i.e. low power). Note that power-line communication and wireless do not meet these requirements for several reasons.




## 1.5 World Adoption Strategy

To encourage worldwide adoption, we make all materials free and open on a gnu public license basis. This includes schematics, source code, reference designs, and all materials. We are creating a platform on which others build. Alternatively, if this was propriety (e.g. a Philips Lighting controlled by Philips), the world would resist because companies do want to be controlled by an external party, especially a competitor. Also, if Philips, for example, did have a proprietary system, the world would copy it, change it slightly, and create a free and open version. Therefore, to encourage adoption and standardization, for purposes of mitigating climate change, we are free and open.

## 1.6 Road Map

We work toward creating a worldwide standards using the following road map:

(1) Publish concepts (e.g. at Electronics Design News) and receive feedback.

(2) Design, simulate, create cost models, and build prototypes. This requires electrical engineers, mechanical engineers, and computer programmers.

(3) Make sure prototypes work well in a system.

(4) Select best prototypes to be used as suggested reference designs for industry to copy. These are heavily documented, well-engineered, well tested, and polished.

(5) Define standard in rough form. This includes mechanical standards, electrical standards and communications protocols.

(6) Present reference designs and all documentation to international standards body organization, such as IEEE.

(7) Members such as Philips Lighting and GE Lighting noodle and decide whether or not to there is demand for suggested technology.

## 1.7 Manhattan 2 Collaboration System

Researchers can collaborate via the Manhattan 2 Collaboration System, described here:

Manhattan 2 Research Collaboration System
http://www.ma2.life/doc/research/common/Ma2_Collaboration_System.pdf

## 1.8 System Prototype Test Site

A 2000 square foot house-like test site is used to test multiple devices within a large home-like system. This structure is being designed by Manhattan 2 Chief Architect John Meyer using Revit software, and is summarized below:

*Proposed Energy Infrastructure R&D Test Laboratory, 1 Page Summary*
http://www.ma2.life/doc/plan/Ma2_TestSite_Poster.pdf

*Proposed Energy Infrastructure R&D Test Laboratory, Detailed Diagram*
http://www.ma2.life/doc/plan/Energy_Infrastructure_RD_Center_DETAILED_DIAGRAM.pdf

*Proposed Energy Infrastructure R&D Test Laboratory, Requirements, 2k sq ft*
http://www.ma2.life/doc/plan/Energy_Infrastructure_RD_Center_v3.pdf

## 1.9 Applications

The following Electronic Design News article provides a glimpse of next generation wired IoT possibilities:

*Proposed New Standard for Smart LED Lighting & Wired IOT, Electronic Design News, Aug 2019*

- https://www.edn.com/proposed-led-wired-iot-standard-can-reduce-energy-use-part-1/ (Part 1)
- https://www.edn.com/proposed-led-wired-iot-standard-can-reduce-energy-use-part-2/ (Part 2)
- https://www.edn.com/proposed-led-wired-iot-standard-can-reduce-energy-use-part-3/ (Part 3)

The following chapters in the Manhattan 2 Blueprint book provide more information on next generation building automation possibilities. If one downloads this PDF to a computer (File, Save As) and opens with an Acrobat viewer, an easy to use left-side Navigation system exposes the table of contents.

*Manhattan 2 Blueprint Book (180 pages)*
http://www.ma2.life/doc/plan/Manhattan_2_Blueprint.pdf#pagemode=bookmarks

- Chapter 13) Extend Computer Network to Windows, Doors, Ducts & Ceiling
- Chapter 14) Wall Bus Signaling
- Chapter 15) $1 Tiny Nodes on 4Wire Cable
- Chapter 16) Create Standards that Automate Windows and Doors
- Chapter 17) Create Modular Chassis Standard for Windows/Doors/Ceiling
- Chapter 18) Create Standards to Gain Thermal Control over All Rooms

The following documents suggest research opportunities for solar packaging and installation, and ground source systems -- which could potentially connect to devices within a building.

*R&D Plan to Create Standardized Plug-and-Play Solar Direct to Building Surfaces and to Land*
http://www.ma2.life/doc/plan/Ma2_Solar_RD_PLAN.pdf

*R&D Plan to Automate Installation of 55°F Ground Source for Heat Pump*
http://www.ma2.life/doc/plan/Ma2_Vertical_Ground_Source_RD_Plan.pdf

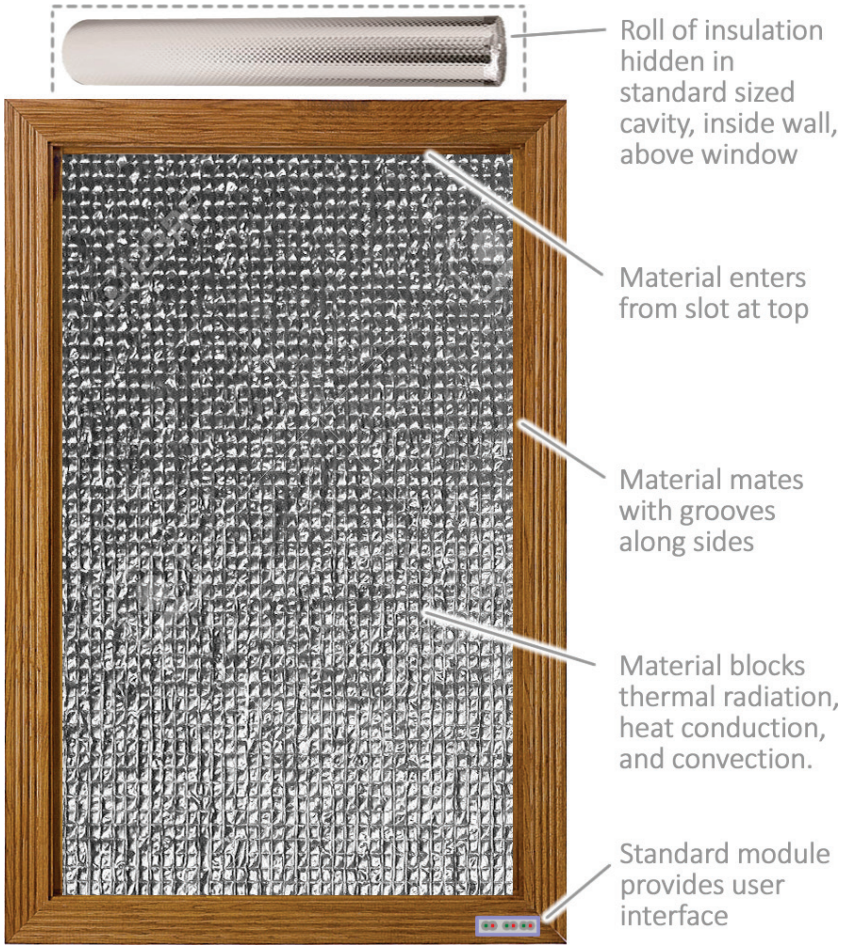
# 2 Manhattan 2 Active Window Development Initiative

## 2.1 What is an Active Window?

There are two types of physical wall windows, Active and Passive. Active is when a window is powered by electronics, contains a microprocessor, and is connected to a building's network. Active is capable of reducing energy consumption via a variety of methods, one of which is deployment of a motorized thermal cover and thermally turning the window into a wall. Active barely exists due to several issues that we intend to address.

Active supports *Motors*:

- Rolled motorized thermal cover mates with rails at window sides and provides additional thermal insulation.
- Motorized 1 to 2 inch thick solid foam thermal barrier embedded in wall, above and/or below window, deploys to cover window. Many windows provide R3 amount of insulation. Solid foam might provide R7 per inch. Two inches is R14 (2x7). R3+R14 = R17. R17 vs R3 means that conducted heat loss is reduced 5-fold. Also one can reduce thermal radiation heat loss with "reflective" coatings.
- Motors also move rolled blinds, venetian blinds, and curtains.




Active can *Sense*:

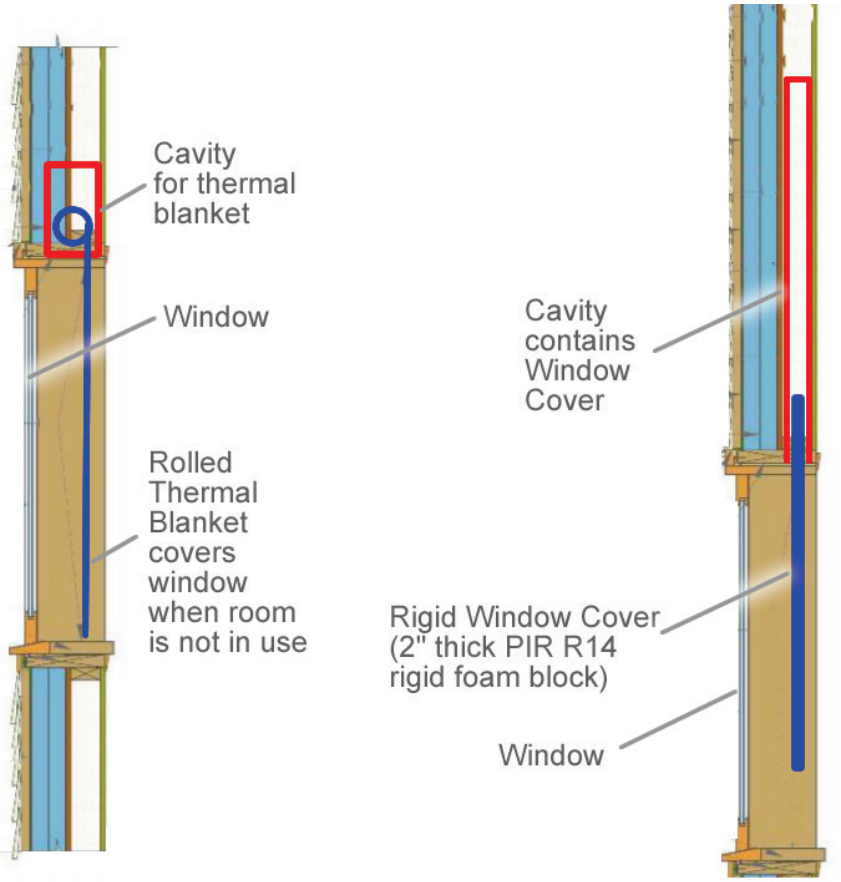
- Measure indoor and outdoor temperature, humidity and pressure.
- Measure outside sunlight

Active can respond to conditions, to reduce energy consumption:

- If indoor heater or air conditioner is on, room is vacant, and we do not want sun to warm the room; then we want as much thermal insulation as possible via thermal covers, curtains and blinds.

- If pressure is greater inside than out (e.g. due to wind direction) and we want air to flow out, then we crack window. We do the same if pressure is greater outside than in, and we want air in.
- Cracking window also helps to control humidity and $CO_2$ (i.e. ventilation).

Products that involve electrified gadgetry for windows exist, yet do not sell well for several reasons:




- There is no standard way to attach to a building, which means one must use a proprietary system that involves costs from components, installation and training.
- If one embeds electronics in the wall and it is co-located with 110/220VAC power, and electrician touches power wire to a data wire (e.g. RS-485), then physical damage occurs to all devices on network.
- If an embedded motor or gearbox fails 10 to 20 years after construction and is no longer manufactured, then building slowly degrades. Building developers are risk averse. They will not construct a building that does not function well over 50 to 100 years. Also, buyers do not want a building that degrades over time, since they lose value.

Manhattan 2 aims to resolve these issues via its Active Window Development Initiative.

# 3 Research Strategy

## 3.1 BuildingBus Development Initiative

Ma2 intends to develop a 2-wire communication standards between multiple devices that provides the following features: supports CANbus communication between ~$1 microprocessors, no damage upon accidental short to power wires, devices use little power when not in use (sleep), wiring supports tree topology (daisy chain not required), hot socket compatible (no damage when attach wires with power on), ≥ 99.999% reliable (not wireless), and transceivers consume ≤ ~10mW of power when signaling (as opposed to ~10x more utilized by RS-485 or 120Ω CANbus). We are calling this new network "BuildingBus™", for lack of a better term.

There are two versions. *BuildingBus 48V* caters to lower power and lower voltage whereas *BuildingBus AC* caters to higher power and higher voltage. *BuildingBus 48V* routes 48VDC power to devices with ~200W/network; whereas *BuildingBus AC* routes 110/220VAC power with ~2,000W/network to devices. Fans, industrial lighting, and motors that move heavy windows require 110/220VAC; whereas many other devices are 48V capable. 110/220VAC cable is bulky and needs to conform to high voltage building codes (i.e. conduit more likely), whereas 48VDC cable is lighter and satisfies low voltage building requirements. For details, search for "BuildingBus Development Initiative" in file Active Window.

## 3.2 Data Link Layer Research

Manhattan 2 Data Link Layer research includes:

- Devices utilize CANbus data layer
  - Auto industry developed CANbus over many years, and ensures its robustness.
  - CANbus controllers are built into many microprocessors, which is a fantastic opportunity.
  - CANbus controller is better than RS-485/UART/Token-Passing in many ways, including:
    - CANbus includes collision avoidance.
    - CANbus controllers manage network traffic, instead of software interrupts on processor which might conflict w/ other processor activities.
  - CANbus requires wire-AND physical layer (if multiple devices are signaling and one device signals "0", then bus goes to "0"; otherwise it stays at "1").
  - CANbus physical layer typically involves driving 60Ω with 30mA. This is too much power; therefore we search for alternative physical layers, described in previous section.

## 3.3 High-Level Software Strategy

We will define "High-Level Software" as the software that implements control functions. This might be referred to as the "Brain", "Artificial Intelligence" "AI", or "System Master Controller". We explore connecting to existing software, and writing our own.

## 3.4 User Interface Software Strategy

We explore attaching to existing user interface software. Also we create our own smartphone app that looks like Windows Device Manager, which can see all devices and all parameters within devices of multiple protocols.

## 3.5 Device Object Model Research Overview

Manhattan 2 device object model research includes:

- Several assumptions
    - Each device contains a unique serial number (e.g. 64bit EUI-64, 32bit MAC address).
    - One Physical Master Controller product for each wired network
        - Manages 2-wire tree topology system
        - Provides network power and termination as needed
        - Connects devices to larger network (e.g. to Ethernet IP).
- Researchers demonstrate co-locating multiple protocols (e.g. DALI, BACnet, KNX, ZigBee, EnOcean, etc.) on common simple devices. For example, LED light controller product supports BACnet *and* KNX. This enables manufacturers to focus on manufacturing fewer products to reduce cost. More RAM and FLASH in microprocessors makes this more feasible.
    - Free and open reference designs with quality software demonstrates multiple protocols running on common devices, to help manufacturers support this effort.
    - DALI, BACnet, KNX, etc. device software is fairly light (not much code).
    - The 512KB Flash/128KB Ram #STM32G474CBU6 microprocessor cost $2.86 and the smaller 128KB Flash/64KB Ram #STM32L431CBY6TR cost $2.13. Notice one can obtain significantly more memory for an additional $0.70. Device manufacturers probably benefit from spending this money, in return for more compatibility and usefulness, especially if quality reference designs help them implement.
    - Common simple devices include things like: light switch (on/off/variable), light (on/off/variable), window cover (on/off/variable), temperature sensor (read value), duct fan (one/off/variable), and duct damper (on/off/variable).
        - Each of these do not have many interface parameters and are fairly simple.

- Zigbee and EnOcean are wireless networking standards, yet in theory, one could attach a wired device to a Zigbee or EnOcean network. This is a bit tricky, since wireless protocols expect fast data transfer rates, which one might have to a Master Controller (MC), yet not along a slow wired network (e.g. 10K bps). Subsequently, one might look at having part of the work done by the fast Ethernet connected MC, and rest done by a tiny processor with low speed communication to the MC.
- Researchers develop object models (e.g. LED Light controller) that consist of a union of features provided by their protocol-specific counterparts. We call this Device Extensive Object Models (DEOM), perhaps. One can then translate a command from a specific protocol to its DEOM counterpart and have the DEOM implement. We focus on a few simple object models to reduce project scope.
    - If one gathers the object model for the same device type (e.g. light) from multiple protocols, they can identify the most complex within the set. For example, the DALI light controller object model supports groups and synchronized dimming at programmable rates, and might be the most complex for lights. One could copy this, call it the DEOM for lights, and identify features that are not supported here that are supported by the other object models, and add those to facilitate the union of all features from all protocols.
    - One might copy the DEOM light control device model, and create a model for Window Thermal Covers, and treat covers in a similar way (e.g. have multiple covers close at same time at same rate). Diagram: http://www.ma2.life/doc/plan/window_thermal_blanket_research.png
    - Software supports read/write for multiple protocols (e.g. DALI, KNX, BACnet) via tunnels to the DEOM. If there are 15 parameters in an object, for example, one might have several lines of code that translate from each specific protocol to DEOM, for each parameter. If there are 3 lines of code per tunnel, one for read and one for write, 20 common objects, and 5 common protocols, then this would work out to 15*2*20*5=3000 lines of code, for example. This is manageable.
    - DEOM software could potentially resides at several locations.
        - Device
        - Master Controller
        - Gateways
        - Client Devices (e.g. smartphone with clones of DEOM libraries)
    - DEOM is extensive. See "Chapter 31 - A Simple Light Bulb" in the following document for example of how simple light bulb is expanded to ~50 parameters. http://www.ma2.life/doc/plan/Manhattan_2_Blueprint.pdf#pagemode=bookmarks
- Analog measurement DEOM object models (e.g. measure temperature) include the following:
    - AccuracyOfSensor immutable parameter (e.g. maximum error between actual stimulus and measured value). This is often ignored yet very important to higher level software.
    - Read sensor X times and return average (e.g. read 1000 times to reduce noise 30-fold).
    - Read sensor X times over Y seconds (e.g. read 1000 times over 1/60$^{th}$ of a second and return average to cause positive part of 60Hz power sinewave to cancel negative part).

- A block of constant (never changes) binary data is associated with each device. This is called a "Device Immutable Binary Block" (DIBB) and it consists of immutable DEOM parameters *and* immutable data associated with each protocol (e.g. KNX device descriptor block). Each device merges all of its immutable data into one big binary block, with a header that describes how to access the contents.
    - This big block of binary data, and its associated serial number (e.g. EUA-64), are transmitted to the Master Controller (MC) when the device first joins network. This data never changes, by rule, therefore each device only joins once. After joining, the MC never asks for this data again. The MC keeps each big block and serial number in a library and holds onto it even if the device seems to disappear (it might be powered off or disconnected). The MC also builds extensive indices and dictionaries that enable it to access data quickly within these big blocks.
    - If each device has 2KB of DIBB data, for example, and you have 1000 devices, then total would be 2MB, which is small relative to controller, gateway and client computer capabilities.
    - Synchronizing and maintaining DIBBs is a project.
    - Master Controller, Gateways, and Client Devices ALL contains clones of DIBB data and responds to read requests locally if data is present and immutable.
        - For example, if your smartphone needs some data, and it is immutable (never changes), we would prefer the smartphone not spend 500mSec chasing down the information.
        - If a read request involving immutable data hits a Master Controller, then it reads from its local DIBB cache and does not touch down-stream devices.
- Researchers write free and open software to encourage collaboration and adoption.

# 4 Suggested Research Projects

## 4.1 RS-486P

### 4.1.1 SUGGESTED PROJECT: Develop New Electrical Signaling Standard (RS-486P)

Define and propose a new 2-wire differential electrical signaling standard that is similar to RS-485, yet provides better support for multi-drop networks within buildings. In this document, we refer to our new proposed physical layer standard as "RS-486P" ("P" for proposed).

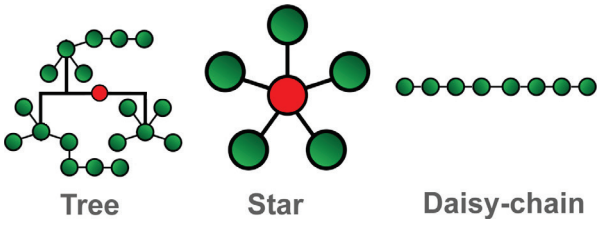
- Researchers design, simulate, and prototype proposed standard
- Researchers design, simulate, and prototype proposed transceiver IC.
- Possibly support 7 different speeds: 1K, 3.3K, 10K, 33K, 100K, 333K, 1M bits-per-second.
- Wires are not terminated and we support tree topology wiring without limitations on stub length and branches. Subsequently, we must control slew rate to avoid ringing. Suggested rise/fall times, maximum network physical size, and receiver bandwidth are listed below.

| Cable Speed (bits-per-sec) | Maximum Network Size (meters) | Rise/Fall time (µSec) | Receiver Bandwidth (kHz) | Application |
|---|---|---|---|---|
| 1K bps | ≤ 1000 meters | 150 µSec | 10 | Connect multiple buildings |
| 3.3K bps | ≤ 300 meters | 45 µSec | 33 | Large commercial building |
| 10K bps | ≤ 100 meters | 15 µSec | 100 | Small or medium size commercial building |
| 33K bps | ≤ 30 meters | 4.5 µSec | 330 | House or large commercial room |
| 100K bps | ≤ 10 meters | 1.5 µSec | 1,000 | vehicle, large machine, or small room |
| 333K bps | ≤ 3.3 meters | 0.5 µSec | 3,330 | Medium sized machine or 19" rack |
| 1M bps | ≤ 1 meters | 0.12 µSec | 10,000 | Medium sized enclosure (e.g. 16"x16"x3" box) |

- Transmitter uses feedback to maintain constant rise/fall time given varying cable capacitance. Slow is beautiful in some cases for many reasons.
- One transceiver IC supports all 7 speeds, where 3 digital inputs selects speed (or some other method). One could control speed by several methods: 3 digital inputs bits (d0..d2), I2C, SPI, pin connected to resistor. A transceiver IC might support multiple methods, one that connects to microprocessor (e.g. I2C or SPI) and the other that is hardwired (e.g. 3bits or set by external resistor). And a pin might be used to select one of these.
- Receiver bandwidth matches transmitter speed, subsequently lower speeds have more EMI low pass filtering, and fewer requirements on twisting/shielding (e.g. 10 KHz receiver bandwidth with 1K bps speed; or 100 KHz receiver bandwidth with 10K bps, etc.)
- If one looks at the time for electricity to move from one side of a network to the other ("flight time", 2nSec/ft) and multiplies this by ~20, and keeps the signal rise/fall time lower than this duration (flight*20), then signal does not ring with unterminated cables. Flight time is <75nSec (1.5µSec/20) and <750nSec (15µSec/20) respectively for 100K and 10K bps speeds, for example.

Electricity travels 1ft per 2nSec, therefore, maximum cable length is 37ft (10m) and 375ft (100m) respectively (75/2, 750/2) for 100K and 10K bps speeds, for example.

- If cable capacitance is 20pF/ft, then one would see a maximum of 750pF (37ft*20) cable capacitance with 10K bps and a maximum of 7.5K pF (375ft*20) with 100K, for example.
    - If the driver needs to move 5Volts in 1.5µS with a 750pF load, or 5V in 15uSec with a 7500pF load, then it would need to pump 2.5mA maximum (I = C*dV/dt, 2.5mA=750pF*5V/1.5µSec, 2.5mA=7500pF*5V/15µSec), for example. We see 2.5mA with both 10K and 100K, since 10x more cable is offset by 10x longer rise time. Subsequently, one IC can support multiple speeds without too much effort.
    - If your driver is good for 2.5mA and this corresponds to 375ft of tree wiring (total amount of wire in entire network), and the largest physical distance between two nodes is less than this, then the driver might hit a limitation before the ringing limitation. Subsequently, one might want drivers to support 5mA instead of 2.5mA, and then say that the maximum total tree wire length is 2 times the maximum distance between two nodes (i.e. physical network size). For example, at 10K bps the 5mA drivers would support ≤750ft of total tree wire length and ≤375ft of maximum distance between two furthest nodes (physical network size).
- Driver supports a maximum of 5mA of drive capability, or some low number (since system is designed to work with low power). Driver cost is related to this number since large currents involve more silicon. 5mA keeps driver cost low.
- A traditional RS-485 system might have termination of 750Ω/120Ω/750Ω ohms to maintain well defined voltages when the cable is not driven (120Ω in the middle is termination for 120Ω impedance cable). Our system is designed to work with lower power devices and therefore might maintain the idle condition differently. For example, one might connect Data+ to 5V via ~10KΩ, and connect Data- to 0V via ~10KΩ (5V/10KΩ = 0.5mA). This corresponds to a 60µSec time constant with 300ft (10KΩ * 6K pF, 20pF/ft), for example. It would be helpful if the driver of the cable returned Data+ to a high level and Data- to a low level, to stabilize the cable more quickly. If this did not occur, the system would transition relatively slowly from 0 to 1 after driving the cable (e.g. over ~6bits if running 100K bps w/ 300ft). Another option is each node has a 1MΩ weak pull up to stabilize the wires when not driven, however if one leaves wires in middle of range after a collision, it would take a long time for them to establish a stable 0 or 1. The traditional RS-485 system needs to deal with wire stabilization when not being driven as well, yet often deals with it via termination resistors (e.g. 750Ω/120Ω/750Ω).
- Remember, we are NOT terminating. We are tree topology, *not* traditional multi-drop along line with termination resistors at both ends. We are not concerned with driving termination resistors or burning power at termination. An important goal is to provide support for low power devices with tiny power supplies (e.g. 5V/16mA power supply for tiny network device).




- If a 14 gauge wire (0.008ohms/meter) drives a 15Amp load 66m (200ft), one will see an 8V drop (15*0.008*66). Therefore, one might see ±15V common mode voltage differences between devices within a building, or more (yet ±6V is more typical). To improve in this situation, we explore expanding the typical common mode voltage used with the current RS-485 IC's (i.e. -7V

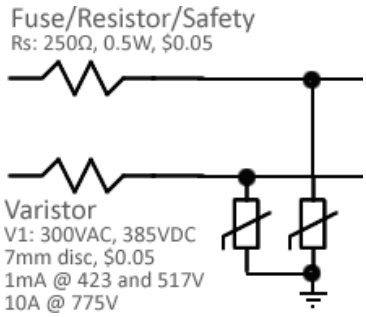
to 12V, +-7V common mode on top of 0 to 5V signaling). Perhaps some 486P transceivers are low cost and support -7V to 12V, whereas others are more costly with more common mode voltage? How much more difficult is it to add another 5V to CMV (i.e. -12V to +17V) or another 10V (-17V to +22V)? Note that the driver does not need to support these voltage, only the receiver. The driver *does* need to be able to take a floating cable at the end of the common mode range (e.g. -12V) and then drive it as needed.

  - If there is 20V difference between Device 1 COM and Device 2 COM (e.g. 20V drop on cable power- wire), and Device 1 drives cable to 5V/0V With Respect To (wrt) earth gnd, releases the bus, and Device 2 drives to 20V/25V wrt earth ground; then we have to wait more time to move to newer voltages since our driver slew rate circuits give us a minimum number of µSec per 5V change. If your change is 20V, for example, then the rise time would increase 4-fold (i.e. i = C*dV/dt, time rise = C*dV/i, i = 3mA drive capability, C = cable capacitance, dV = change in voltage). Subsequently, if you have a large common mode voltage between devices, you might consider telling your transceiver IC to go to one speed, yet run your digital bits-per-second slower. For example one might tell transceiver IC to support 33K bps rise/fall times yet microprocessor CANbus actually runs at 10K baud (100uSec/bit) to give wires more time to slew across large common mode voltages.
  - If you are slewing over a large voltage range due to large voltage drop on power- wire (e.g. 20V), you would want your two data wires to move at the same rate to their new voltages. If one RS-486P wire had more capacitance to earth ground on it than the other data wire, it might move at a different rate and cause the logic level to flip for a short duration, causing an error. This is also an issue when 10KΩ termination is driving the cable (e.g. all devices release after collision). This is more of a concern with larger common mode voltages, since one needs to slew over more voltage to get to target voltage.
  - The RS-486P specification should probably discuss in some way the issue of unbalance capacitance on data+- wires.
  - If receiver IC has external 1MΩ series resistor between cable data+- and IC pins to help protect against over voltage, and IC pin capacitance differs, then one needs to make sure slew rate differences do not lead to momentarily flipped bit.
  - One might want 10KΩ termination to 5V without series diode and another 10KΩ to 0V without series diode, since you don't want one diode off and one diode on due to different common mode voltages between devices, causing wires to move at different rates.

- Researchers develop a strategy for protecting against damage given continuous short circuit to the following voltages, transceiver power on or off:
  - Protected against short to +-30VDC, helpful in 24VDC systems
  - Protected against short to +-60VDC, helpful in 48VDC systems
  - Protected against short to +-265VAC, helpful in 110/220VAC systems

- One might look at protecting with low cost 30V Zener diode (e.g. $0.04 #BZB84-B30 30V) and 100Ω 1W series resistor. 10mA through 30V diode is 300mW, which is the most it can handle. And 10mA through 100Ω is 1V; therefore, we end up protecting against short to +-31V, and we are only ON in the event the voltage is between 30V and 31V (sort of). In summary, series resistor and zener is more helpful with lower voltages and a series on/off switch is better with higher voltages yet at more cost. For more ideas search this file for "Design Current Limiting IC that protects wired IoT devices".

- One could place a varistor on each transceiver wire, to protect against overvoltage spikes that are dozens of microseconds long. Two resistors that are rated for flame/safety/fuse (e.g. #NFR25H0001000JR500, 100ohms, 0.5W, $0.10, 3x6mm, thru hole) and two varistors (e.g. $0.07, 40J, 2.3KA, 7mm disc, thru hole, 300VAC 385VDC working, 1mA @ 423 ... 517V, 10A @775V, #MOV-07D471KTR) cost a total of $0.34. A 385VDC varistor could be used to protect downstream components from +-800V spikes, for example. These are only good for high voltage spikes. If you put 385VDC continuous into this circuit, you will burn up your series resistor and/or varistor. If one places 100 ohm series resistors between cable and transceiver IC in both the signal+ and signal- paths, then 2.5mA will cause a 2.5*100=250mV drop in one resistor and a 250mV drop in the other (0.5V total) while driving cable capacitance, which is acceptable.




- If one wanted to protect against continuous short to 265VAC, they would need the more costly MOSFET switches or transistors. If RS-486P wires are co-located with 110/220VAC power wires, then this might be worth the cost, since an electrician might accidently connect power to data; and damage a large set of difficult-to-access devices at one time, which is costly. People who build buildings are risk adverse. They want reliability and are willing to pay for it, since chasing problems is costly.
- RS-485 involves driving a termination resistor (e.g. 120Ω at each end, 60Ω total, 24mA * 60Ω = 1.4V). This pushes one into working with smaller voltages (e.g. > +200mV differential is logic 1, < -200mV differential is logic 0). RS-486P is not terminated. Therefore, it can work with higher voltages. If driver power is 3.3V and it pulls one wire to 2.7V (3.3V-0.6V) and the other to 0.6V (0V+0.6V) via transistors, then differential voltage between wires would be 2.1Volts (2.7V-0.6V). Subsequently one could increase ±200mV threshold to ~±1400mV (or more), to increase noise immunity by a factor of 7, for example. This reduces the amount of shielding/twisting one must do to keep external signals from coupling into data.
- If you run RS-486P wires physically parallel to 220VAC/20Amps wires in a 5wire 100meter non-twisted cable, for example, and suddenly power on a 220VAC load, then a quick change in current (e.g. high di/dt) would induce a magnetic field around the RS-486P wires, and result in a voltage spike on those wires. If 486P was driven by ~10KΩ resistors (e.g. one 10KΩ to 3.3V and one 10KΩ to GND, logic 1), then this high source impedance (e.g. 5KΩ = 10KΩ // 10KΩ) would need to resist the induced voltage spike. One would want the voltage spike to be less than the logic threshold voltage (e.g. ~1400mV), to keep from receiving a bit error. Susceptibility to inductive coupling is a function of the following:
    - RS-486P transceiver low pass filter attenuates fast spikes, to an extent determined by filter bandwidth and spike rise time.
    - Cable length, cable geometry, wire twisting.
    - Logic differential threshold voltage (e.g. ±1400mV), change of current in power wires (di/dt).
    - References:
        - Inductive Coupling and how to Minimize their Effects in Industrial Installations http://www.smar.com/en/technical-article/inductive-coupling-and-how-to-minimize-their-effects-in-industrial-installations
        - EMC for Product Designers, Chapter 11, 550 page book, By Tim Williams (on gsw's Kindle system)

https://www.amazon.com/dp/0081010168/ref=cm_sw_em_r_mt_dp_U_nklODbHZJZJE5

- Slower rise/fall times translate into less radiated RFI, which reduces coupling spikes into other wires during rise/fall durations. This reduces problems with nearby analog signals, and reduces shielding/twisting requirements.
- Many low cost processors include a CANbus controller (e.g. Xmc1404 $1.20, or Xmc4108 $2.17 with floating point hardware). Researchers explore attaching RS-486P transceivers to a microprocessor's CANbus controller where one uses CANbus Data Link layer to drive RS-486P. This would support multiple devices along one cable (multi-drop) without much effort.
    - In this setup, the CANbus controller provides framing and manages collisions. RS-485/486P is an electrical signaling physical layer standard. CANbus is both electrical signaling physical layer *and* data layer (e.g. framing, collision, media access control).
    - If one can attach RS-486P to a microprocessor's CANbus controller, then engineers worldwide would considered it instead of the traditional CANbus physical layer (e.g. 24mA drives 60Ω to 1.4V). RS-486P drivers require ~2.5mA while driving cable capacitance, which means transceivers cost less money, and also means one could have a much smaller device power supply. If your microprocessor draws 4mA at 5V and your RS-486P transceiver draws 6mA at 5V, then total would be 10mA, for example. If working with CANbus, you would need 4mA for processor, 24mA to drive 60Ω, and 60mA to drive cable capacitance; for a total of 88mA. In summary, we reduce power supply needs significantly when we move from traditional CANbus physical layer to RS-486P (if it works).
    - Note there is a subtle difference between RS-485 and CANbus. CANbus has "termination passively holds at 0V" (logic 1, idle condition) or "pump 24mA to produce 1.4V" (logic 0). This is called "Wired-And" since "1" and "1" give you "1"; otherwise "0". If there is a collision, then CANbus "pump 24" always wins; whereas RS-485 collisions result in unpredictable results (driver IC with more current capability wins). CANbus is designed to allow device with lowest ID to win in the event of collision and continue transmitting, unencumbered (wire-And facilitates this). This would not work with RS-485.
- The RS-486P transceiver IC's should have an input pin where designer selects Wire-And on or off. If off, the driver behaves normally and pulls two signals toward Vcc/Gnd. If On, then 486P behaves differently, and facilitates Wired-And.
    - Wire AND entails releasing bus (go to high impedance) after slewing to logic 1 voltages (e.g. 4.2V on Data+ and 0.8V on Data-). For example, transceiver drives logic 0 and holds it in place after slewing at a specific rate, and then when it returns to logic 1 it drives cable again, slewing as needed, and when it achieves logic 1 voltages it goes into high impedance and lets the two 10KΩ resistors maintain logic 1 (i.e. 10KΩ between Data+ and 5V, 10KΩ between Data- and 0V). Subsequently, if two parties drive cable and disagree, logic 0 will always win.
    - It is not clear this would work, especially with many parties along one cable, each with different common mode voltages between them. Yet this approach does deserve consideration and could possibly be a powerful tool at organizing multiple low power, low cost parties along one pair of tree-topology wiring via internal microprocessor CANbus controllers.

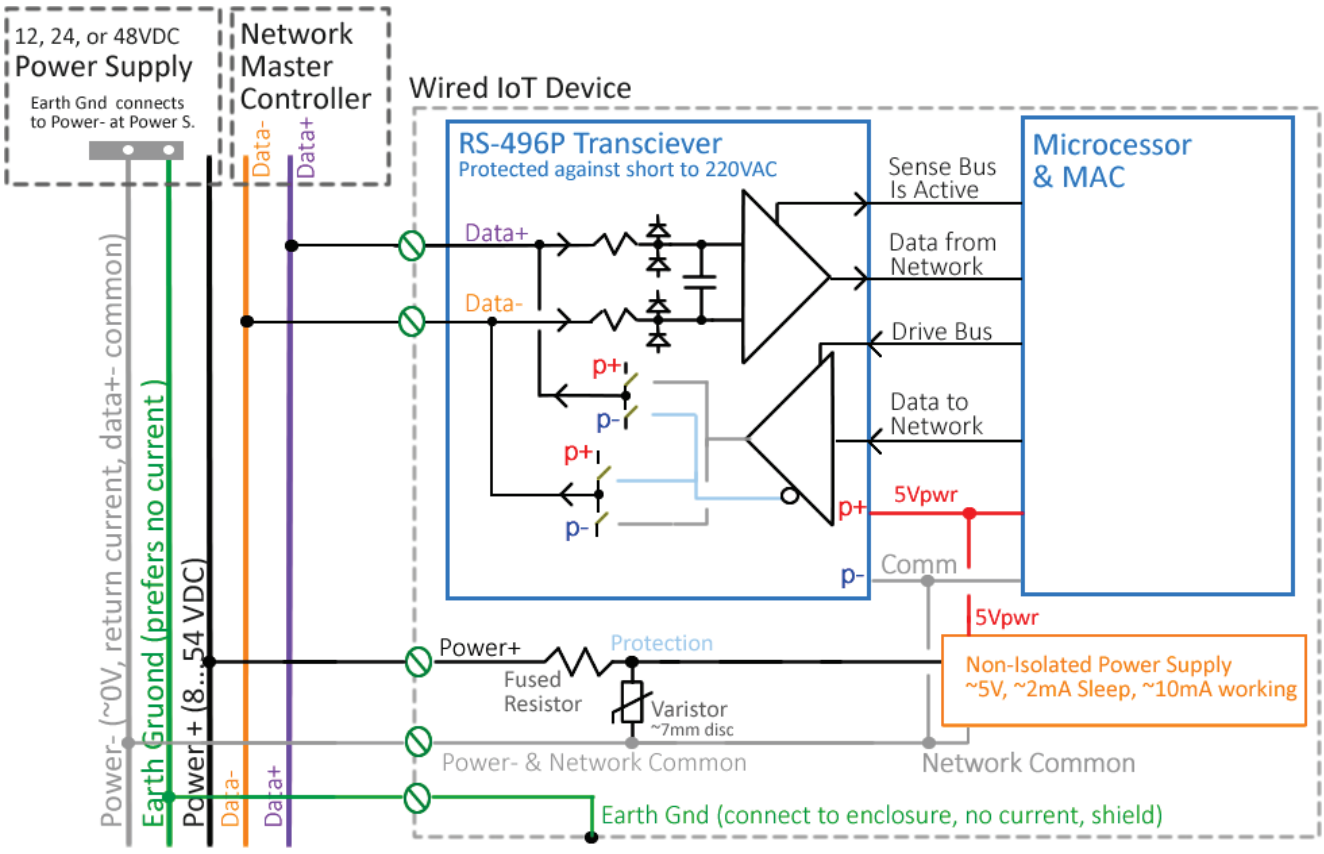
- A disadvantage of this approach is 10KΩ source impedance is not the best when resisting RFI that couples into our data cable. For example, if one routes data and power in the same cable and a 10Amp appliance suddenly turns on, it would cause a possibly large voltage spike to inductively couple from power wires to data wires. Whether or not this is seen as a logic transition depends on several factors: cable length, distance between power and data wires in cable, power wire current rise time (di/dt), receiver bandwidth, and source impedance (e.g. 10KΩ).
- 110/220VAC sometimes include 1KV 20µSec wide spikes. These could inductively couple into physically parallel data+- wires and fight with 10KΩ source impedance trying to establish a stable voltage. If you drop the source impedance 10-fold, the voltage spikes drop ~10-fold, for example. However, with RS-485P, one can dial down the bit rate, which also dials down the bandwidth of the receiver. For example, one might favor 1K bps network speed and 10 kHz receiver bandwidth to significantly increase immunity to 20uSec wide spikes.

- In summary, we want for designers to be able to connect microprocessor CANbus Hi/Lo signals to 486P transceiver IC, tell IC to implement wire-AND, connect cable+- to two 10KΩ resistors at one position, specify cable speed at IC; and then have multiple microprocessors talk along one cable with ready-to-go reliable media access control, framing, data link layer support, and collision avoidance/resolution (last 4 items are implemented by CANbus).
- CANbus Comments:
  - If your RS-486P system wants to do Manchester encoding (>50% of the time cable is at logic 1), then the CANbus NRZ with bit stuffing is not going to cooperate. Search "Manchester" for details.
  - Example CANbus Controller (i.e. Infineon MultiCAN's CAN_CLC/CAN_FDR set baud rate) https://www.infineon.com/dgdl/Infineon-MultiCAN-XMC4000-AP32300-AN-v01_00-EN.pdf?fileId=5546d4624e765da5014ed91d6be32110
  - An alternate to CANbus managing collisions is Token Passing. However, if token is lost, the entire network can become frozen. A monitoring system that looks for this is helpful. CANbus and Wire-AND is a terrific alternative to Token Passing. For a summary of RS-485 media access control via Token Passing (i.e. who controls cable), see: https://cseweb.ucsd.edu/classes/fa11/cse123-a/123f11_Lec6.pdf
- Our RS-486P prototype uses one Microprocessor DIO bit to drive 2:1 multiplexor that connects RS-486P circuit to either Microprocessor UART Tx/Rcv interface or Microprocessor CANbus interface, uses one Microprocessor DIO bit to select Wire-And on/off, uses 3 Microprocessor DIO bits to select speed (e.g. 1K … 1M bps), and includes a tiny FPGA for researchers to explore various ideas.
  - Example CANbus transceiver (PHY, $1.46) w/ external resistors to set rise/fall time (via feedback?): https://datasheets.maximintegrated.com/en/ds/MAX3051.pdf
  - Example Rs485 transceiver protected to ±80V https://para.maximintegrated.com/en/results.mvp?fam=rs485&267=60
  - Example Rs485 transceiver that supports ±25V common mode voltage https://www.analog.com/en/products/ltc2862.html
- Researchers work with simulator to evaluate design ideas.
  - Simulation need to look at multiple devices on cable with different common mode voltages (e.g. +-5V) between devices (e.g. 3 devices in simulation labeled left/middle/right, left circuit

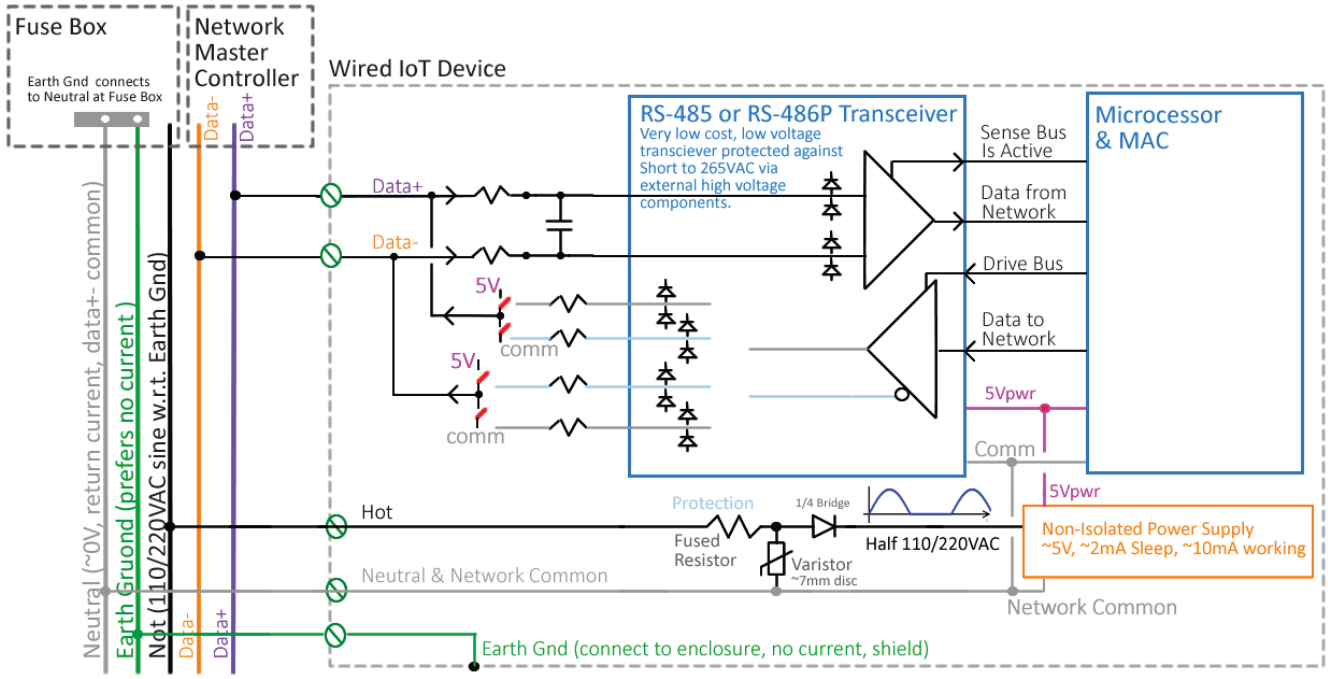
common is -5V wrt middle circuit common, right circuit common is +5V wrt middle circuit common).

  - For a simulation of a DALI-like 16V/250mA system that supports 10K to 30K bit rates via op amp slew rate control, see this pdf schematic and TINA simulation file.

- Our RS-486P prototype uses Op Amps and Instrumentation amplifiers to implement slew rate control.
- See Also: GWeinreb_Manhattan2_ResearchNotes.xlsx / "Wall Bus" / "RS-486P (Proposed)"
- For more details, search "Electrically Connecting a Physical Window to a Network".
- Prototype might fit nicely into one Arduino Uno PCB
  - https://www.mikroe.com/rs485-5v-click
  - https://www.arduino.cc/en/Reference/ArduinoRS485
  - https://www.arduino.cc/en/Guide/MKR485Shield
  - https://store.arduino.cc/usa/arduino-mkr-485-shield
  - https://store.arduino.cc/usa/mkr2uno-adapter
- The below illustration shows one concept for an RS-496P transceiver.




- Alternatively, one could have a variation of this IC that connects to external series resistors and switches and supports accidental short to 265VAC without damage. For details see "Design Current Limiting IC that protects wired IoT devices".




- Research includes: analysis, sketch different circuits, simulate, build prototypes, place notes and calculations in one spreadsheet, write internal report, make all materials available to the public free and open, and publish summary in online publication (e.g. Electronic Design News, www.edn.com).
- Summary: Building automation could benefit from a next generation multi-drop signaling electrical standard that supports tree topology with unlimited stub length (not just daisy-chain), a strategy for defending against continuous short circuit to 265VAC, lower power transceivers (2.5mA drive instead of 60mA), lower cost transceivers (less silicon due to lower drive current), termination not required, and support for lower cost products that utilize extremely lower power microprocessors with tiny power supplies.
- Researchers are not responsible for pushing industry to adopt the proposed standard. Instead, their sole responsibility is to do good work and make materials available to others so that they can improve, build on, and utilize in anyway.

## 4.2 Electrical Protection

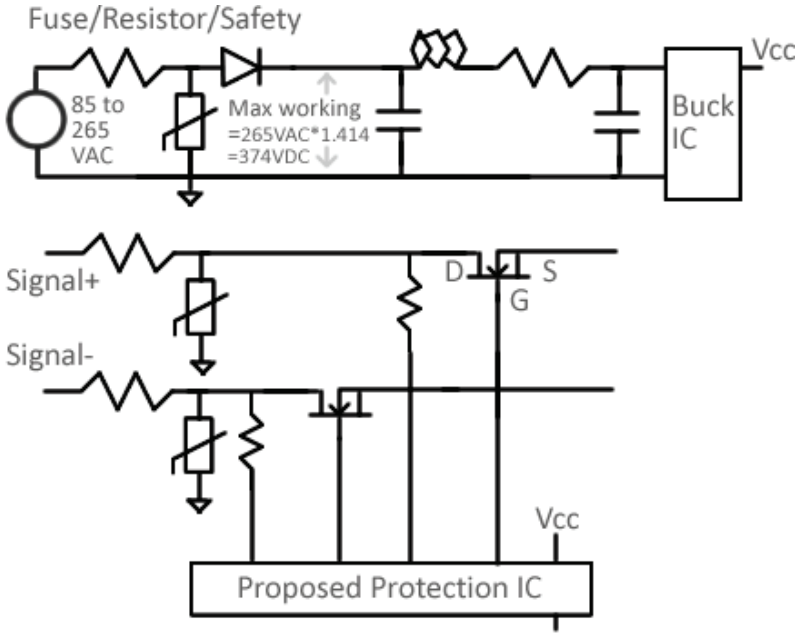
### 4.2.1 SUGGESTED PROJECT: Design Current Limiting IC that protects wired IoT devices

Researchers design low cost IC that protects wired IoT device data and power wires from continuous short circuit to 265VAC. One can assume device receives AC or DC power and communicates via 2 data wires (e.g. RS-422, RS-485, and RS-486P).

- If devices are embedded in walls and are difficult to access, then 100 year high-reliability requires protection against over-voltage.
- There are two power scenarios. One is device is powered with a DC voltage (e.g. 24V, 48V) and we need to also protect these power wires from accidental short to 265VAC. The other is we are

powered with 85 to 265VAC and internal power supply converts this to microprocessor 3.3V or 5V.

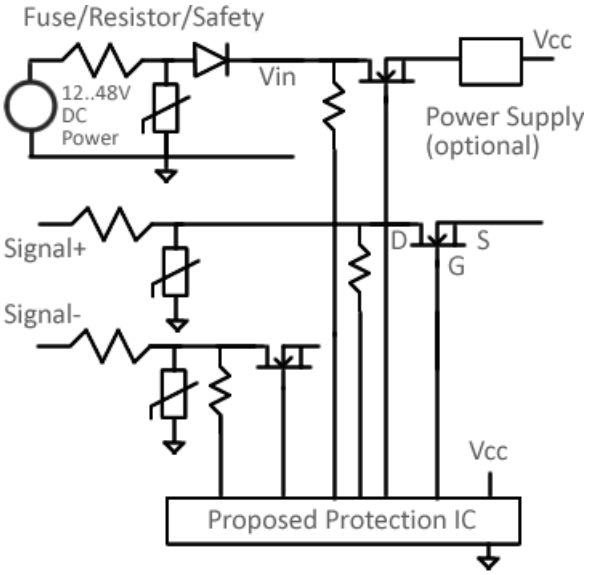
- Pictured to the right is a suggested approach in the case where we have our own power supply powered by 110/220VAC. In this case, external signal+- MOSFETs are turned on by our proposed IC when Vcc is present and 2 sense voltages are ok (e.g. signal+- wires via 1MΩ). The picture shown here is a summary; in practice, one might consider back-to-back MOSFETs and Zener diodes in key locations. The signal+- series resistors and varistors protect MOSFETS from high voltage (e.g. 400VDC varistor and 100Ω series resistor clamps at 775V/10A and protects downstream 800V MOSFET from 4KV spikes).




- Also, we support the scenario where our device receives DC power that powers internal electronics such as a microprocessor; and this DC Power also needs to be protected against short to 265VAC; as pictured below.

- Our protection IC senses voltages via a large resistor (e.g. 4MΩ, 220^2/4M=48mW) and if a voltage is out of range, it turns off all three external power MOSFETS.

- This might require an opto-coupler that produces negative voltages to power mosfet gates, or a small negative voltage power supply. Note that a $0.26 #MAX1720 and two 0805 capacitors can get you -5V for only $0.28 total. Also, a $0.10 #TV-357T opto-coupler might be helpful to get 1V with respect to any node in the circuit. MOSFETS rated for higher voltages often have higher gate-to-source threshold voltages (e.g. 5Volts). Getting this to work is tricky.



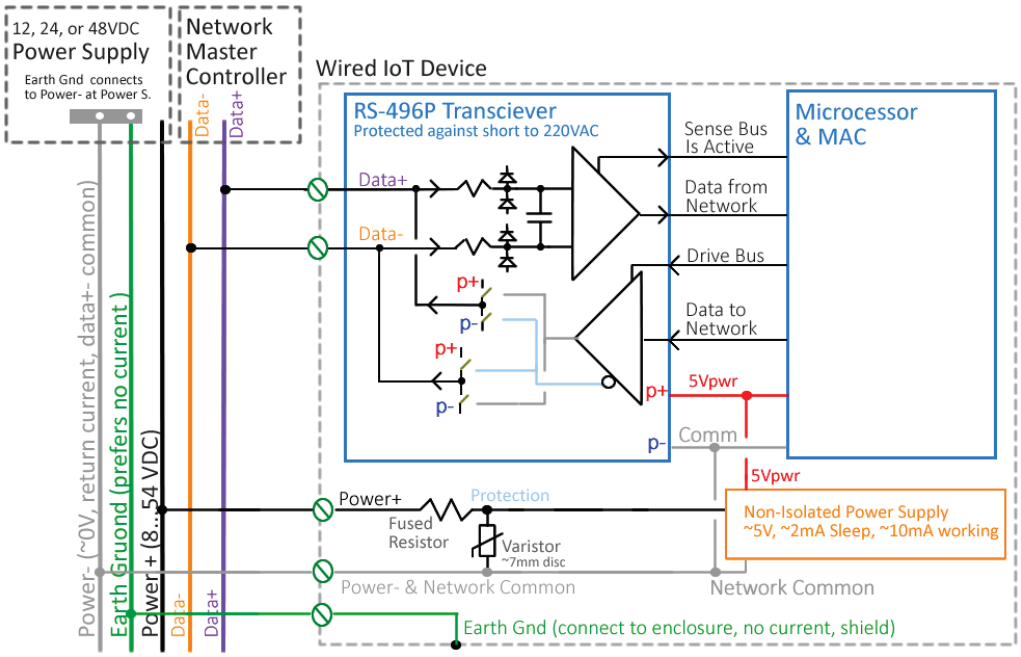

- RS-485 often works with -7V to +12V voltages with respect to circuit common (i.e. 0 to 5V signal with +-7V common mode). One could support this range with a -10V power supply, 3V gate-to-source threshold MOSFET, and a gate-to-source +-22V range, in theory.

- If Signal+- wires are expected to support +-20V common mode voltages, for example, then one would need to make sure the gate voltage was set correctly. Supporting wide common mode voltages such as this typically conflicts with component gate-to-source threshold limitations and

is tricky. For a summary of how to work with MOSFET's, see:
https://www.electronics-tutorials.ws/transistor/tran_7.html

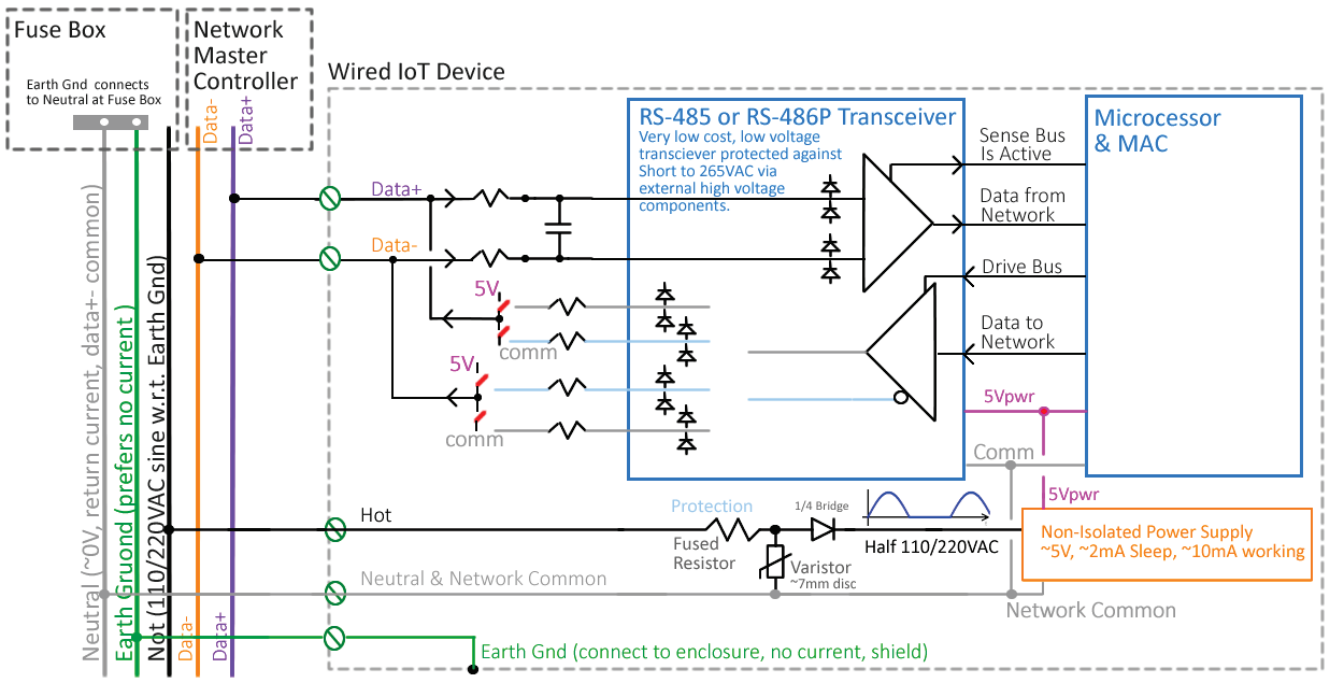
- Figure 1 of MAX4506 datasheets shows classic signal projection circuit where power supply supports signal range (e.g. +-15V power supply for +-15V signal).
https://datasheets.maximintegrated.com/en/ds/MAX4506-MAX4507.pdf
- Design engineer selects external sense resistors and MOSFETS depending on needs (i.e. current, voltage range, and switch $R_{on}$ resistance), to make proposed protection IC useful in a variety of scenarios.
- We might have several circuits inside our IC, and they all need to be operating correctly in order to turn on the three external MOSFET switches.
- We rely on an external MOSFET as our switch (perhaps rated to 800V), therefore our custom IC does not need to be rated for high voltages; and can therefore be low cost, to encourage adoption.
- We assume external series resistors and varistors are limiting 4KV 20µSec voltage spikes to 800V before our circuit (e.g. varistor: 300VAC 385VDC working voltage, 1mA @ 423 ... 517V, 10A @775V). Subsequently, MOSFET components downstream from this clamp need to be rated for 800V, yet not more.
- It is the switch (transistor or mosfet) that needs to be rated for higher voltages. Example high voltage switches are the #TSM1N80 mosfet ($0.23, 800V, 22Ω) and the #STN0214 transistor ($0.43, 1200V). Three #TSM1N80 would cost $0.69, which is acceptable.
  - #TSM1N80: https://www.taiwansemi.com/products/datasheet/TSM1N80_B15.pdf
  - #STN0214: https://www.digikey.com/short/prbndp
- Transistor switch needs current into base to turn ON; whereas mosfet needs voltage between gate and source to turn on. In the case of the #FP0100, it looks like current flows through current shunt resistor (e.g. 100Ω) and when voltage drop is high enough it turns on internal 4Ω mosfet. At first, we see 100Ω series resistance with lower currents and then when voltage drop becomes higher the mosfet turns on and this 100Ω is replaced with 4Ω mosfet series resistance. Exactly how this is done deserves further consideration.
- Notice in Figure 4 of #NSV50010YT1G datasheet we have 150KΩ suppling bias voltage (220^2/150e3 = 0.3Watts). Preferably, we would want something that uses less power (e.g. 5MΩ, 220^2/5e6 = 10mW) to power voltage sense circuits. Also in Figure 4 we have Vin pin limited to within 15V of the Vout pin via two Zeners; showing how one can limit current with an IC that is rated for a low voltage and is inexpensive.
- RS-485 might drive 60mA when driving cable capacitance (e.g. 100m cable, 1M bps); whereas RS-486P drives ~3mA when driving capacitance. Subsequently, one would have external MOSFET protection switches that match each case if they wanted one protection IC to support both 485 and 486P.

- DALI+- wires and power wires (e.g. 48VDC to tiny device power supply) can work with diode drops, series diode, high voltage transistors and current flowing in one direction. However, RS-485 and RS-486P need current to travel in both directions and therefore probably need a MOSFET.
- One could test prototype by placing it in-line with existing devices (e.g. 0 to 20mA loop device, RS-485 device) and then subjecting wires to 265VAC and 4KV 20µSec spikes.
- Researcher's internal report can comment on how proposed IC might be used in the following cases:




https://en.wikipedia.org/wiki/Surge_protector#Standards

- One might conclude that a better approach is for the *transceiver IC* to be responsible for surviving short to 265VAC instead of introducing an additional *protection IC,* as illustrated *above*. Obviously, this would not protect existing RS-485 or 0-to-20mA transceivers.
- Alternatively, one could have a custom low cost ~$0.20 low voltage transceiver IC that drives 4 external switches, as illustrated *below*. Four external 800V diodes and two external resistors might cost $0.15 total. Four external high voltage switches might cost $1. Total parts cost might be $1.40, which is reasonable.




- Research includes: analysis, sketch multiple circuits, simulate, prototype, test prototypes, write report, and publish summary in online publication (e.g. Electronic Design News, www.edn.com).
- We are not fabricating the actual IC, only developing a suggested IC for industry.
- Search this document for "DALI Over-Current and Over-Voltage Protection" and "Protection" for more ideas.
- There are several methods of protection: sense high voltages and turn off switches, current limiter, voltage limiter, and temperature sensor IC detects excess temperature (e.g. 105°C) and turns off switches.
- All materials are free and open, to encourage further development and adoption by industry worldwide.

### 4.2.2 SUGGESTED PROJECT: Build Online Calculator to Help Engineers Design Surge Protection

Build online tool (i.e. write software) that designs and analyzes surge protection circuits. In many cases, engineers place series resistors and transistors into circuits without understanding what they will do, and how they will change over time. One could enter parameters such as series resistor resistance and power rating; varistor working voltage, voltage @ 1mA, voltage at large current, Joule rating, KV rating; and then analyze circuit and see how it does with respect to various standards.

- One could look at a variety of circuit approaches: series resistor plus varistor, series resistor plus Zener, series resistor and safety capacitor (for fast spikes).
- Tool suggests components, specifies cost, and calculates total cost.
- Tool supports different applications, such as: RS-422/485, 0 to 20mA loop, DALI+- wires, CANbus H/L/12V, 24VDC power, 48VDC power.
- Research includes: writing software, make website available to the public, write report, and publish summary in online publication (e.g. Electronic Design News, www.edn.com).

- All materials are free and open, to encourage further development and adoption by industry worldwide.

## 4.3 DALI

### 4.3.1 SUGGESTED PROJECT: Propose DALI 3 electrical signaling standard that is faster than DALI 2

Traditional DALI 2.0 operates at 1200bps. In this project, researchers (try to) develop a version that runs faster. Perhaps researchers propose a DALI 3P standard ("P" for proposed) that supports 5 different speeds: 1.2K (compatible w/ old DALI), 3.3K, 10K, 33K and 100K.

- DALI 3P devices work on a DALI 2 network, and DALI 3P devices work on a DALI 2 network. In order to run at the faster 3P speeds (3.3K to 100K bps), one needs all devices on the network to be DALI 3P. The network controller determines if a device is DALI 2 or 3P when they join the network, and runs the network at traditional 1.2K if at least one device is DALI 2.
- DALI 3P, like DALI 2, is not terminated and supports tree wiring topology.
- Faster networks require shorter networks and we control slew rate to avoid ringing. Rise/Fall times are 150µSec w/ 1.2Kbps (≤1000meters), 45µSec w/ 3.3Kbps (≤300meters), 15µSec w/ 10Kbps (≤100meters), 4.5µSec w/ 33K bps (≤30meters), and 1.5µSec w/ 100Kpbs (≤10meters).
- Controlling slew rates requires a feedback loop where devices sense voltage across DALI wires and short these two wires a variable amount when signaling "0" to facilitate a specific slew rate, to avoid ringing. This requires a custom DALI transceiver IC that supports this in a low cost manner. Perhaps the DALI 3P transceiver IC and the RS-486P transceiver IC are the same IC?
    - Device uses feedback to maintain constant *fall* time given different cable capacitance and device current consumption. Master Controller does the same, to support constant *rise* time. Note that the Device controls the *fall* and the Master Controller controls the *rise*.
- For a simulation of a DALI-like 16V/250mA system that supports 10K to 30K bit rates via op amp slew rate control, see this pdf schematic and TINA simulation file. This simulation has issues, as one can see from the waveforms, yet might be a decent starting point. Researchers consider the various ways of controlling ringing, such as the technique shown here.
- Receiver bandwidth matches transmitter speed, subsequently lower speeds provide more low-pass filtering.
- Devices do not place much capacitance across cable, since it might interfere with signaling. One could require that 3P devices place a current limiter (e.g. 10mA limit, $0.13, #NSV50010YT1G) in front of any internal power capacitor to limit loading the network while capacitor charges. If electric circuit downstream from capacitor (e.g. tiny microprocessor) uses 5mA average and the capacitor charges 50% of the time, for example, then it would need 10mA while charging.
- It is not clear how one sets network speed. Perhaps end user programs speed into Master Controller, depending on network physical size, and master controller communicates this to all devices using the slow default 1200bps. After speed is established, all devices talk at that rate. Master controller would deprecate to 1200 if at least one DALI 2 device was on the network. An

end user with a 10 x 15 x 8 meter (30 x 45 x 24 feet) house might select 33K bps (≤30m network length), for example.

- We assume DALI 3P involves a custom interface IC to help one interface to network. For prototyping purposes, we emulate this IC with multiple analog components.
- Custom IC needs to control rise/fall time ("slew rate limiting"). If microprocessor is on 110/220VAC side of opto-couplers, it might use one additional opto-coupler ($0.10) to tell custom interface IC that microprocessor it talking to custom IC and not to DALI network. This would allow microprocessor to set network speed. Alternatively, if microprocessor was on DALI+- side (or there were no opto-couplers), then microprocessor could communicate more directly with custom IC.
- Infineon has a reference design with an Xmc1200/1400 processor that drives an LED lamp (i.e. current source that is modulated to implement dimming) and implements DALI 2 with the microprocessor is on the 110/220VAC side of the opto-isolators. For details, search this file for "DALI Slave Reference Designs".
- DALI 2.0 uses 1200 bps and is slow enough for microprocessor to manage each bit with software. DALI 3P runs faster and is therefore harder to support with the DALI software interrupt-driven system. Subsequently, researchers explore using the CANbus data link layer which is built into many microprocessors ("CAN Controller"). Note that the Infineon #Xmc1404 microprocessor supports CANbus, cost only $1.20, and can drive motors and LED lamps.
- CANbus's "termination passively holds at 0V" (1) or "pump 24mA to produce 1.4V" (0) behave in a similar way to DALI's "passively rest at 16V/250mA" (1) or "short to 0V" (0). If there is a collision, then both CANbus "pump 24" (0) and DALI "short to 0V" (0) always win (unlike RS-485 which results in IC with most drive current winning).
- DALI hardware expects Manchester encoding. CANbus does NRZ with bit stuffing (you will get at least one 16V "1" bit every five 0V "0" bits) and therefore will starve devices *more* for power.
    - Perhaps DALI 3P devices can be engineered to tolerate 20% (NRZ 5bit stuffing) power support instead of 50% (Manchester)?
        - CANbus Bit Stuffing: https://en.wikipedia.org/wiki/CAN_bus#Bit_stuffing
        - CANbus Frames: https://en.wikipedia.org/wiki/CAN_bus#Data_frame
    - A DALI 2 power capacitor in device might droops 3V in 500µSec while running 1200 bps. If we instead run "NRZ 5bit stuffing" at 10K bps, for example, we will get 16V/250mA for 20µSec and 0V/0mA for 80µSec, and see 0.48V droop (3*80/500), which is ok. Also, if device needs 5mA on average and you supply it 50% of the time, then you need to draw 10mA when powered. Whereas in the 20% scenario, you would need to feed it with ~25mA when powered (5mA/20%). If Master Controller supplies 16V/250mA and each device pulls 25mA on occasion, then it will support a maximum of 10 devices, which is low. If one examines CANbus data frames carefully, one can see the situation is not as bad as this since many bits are fixed at 16V logic 1 by design and also there is time between frames to allow power capacitor to charge. Getting this to work requires attention to a variety of issues.
        - Perhaps DALI 3P transceiver IC (or FPGA on prototype) converts NRZ to Manchester? Search "Manchester" in this file for more details.

- If transmitting data at 10K bps, then a 29bit (3mSec) frame might have a minimum of 30% to 45% of 16V power, and if capacitor droops 3V during this time, it might have enough time to charge in-between frames. Also, one can look at ~10mA current limiters (e.g. $0.13 #NSI50010YT1G) in each device to reduce drawing too much power while charging. This would push device to use time in-between frames for more charging.
- Other possibilities include: device uses less than 5mA on average, current limiter component limits current to less than 10mA, master controller supplies more than 250mA, total number of devices on cable reduced from 64.
- Note that the DALI standard already is asking devices to draw less than 2mA, yet it is not clear how many products actually do this. Reference designs often show drawing more. If you have 64 devices that draw 2mA average and you are doing 50% Manchester encoding then you draw 4mA when powered and 64*4=256mA which is the existing DALI 2 Master Controller mA current capability.
- Many networks do not have 64 devices; therefore designers might ignore this 2mA DALI 2 limit. Subsequently, their products are not controlled, which leads to some networks being buggy. DALI 3P needs to control this in some way to avoid buggy.
- DALI 3 Device Descriptor blocks include parameters that specifies maximum current drawn by device on average, during a 1 signaling and during a 0 signaling. This helps Master Controller calculate if network is starved for power, and reports devices that are drawing significant power.
- One possibility is for DALI 3P Master Controller to support 1Amp instead of 250mA. And, Master controller outputs ~50mA (reduced power) if it sees < ~4.5V (or perhaps something higher like 8V) on the cable, to reduce wasting power during "0" signaling (i.e. where switch shorts power supply to 0V). If this were the case, then 1Amp power burning would occur only during the fall time (e.g. 1.5 µSec fall time for 100K bps 10uSec per bit signaling).
- If LDO (low drop out voltage regulator) needs 1V headroom and you have 3.3V for device power, then you would need 4.3V at LDO input. And, you could have DALI 3P not drop cable down to 0V and instead drop it to something larger like 8.3V with a mosfet or transistor. Subsequently, you could get power to your device 100% of the time (during logic 0 and 1).
  - If you drop 2V on each wire (4V total, 250mA, long cable), you might want drivers to stop at 8.3V so that LDO's far away see 4.3V.
  - If you put 8.3V * 250mA across your transistor or mosfet, you will burn 2Watts, which is a big number. You might only do this 60% of the time, so your switch might need to be set up for 1.2W. The $0.35 #FCD3400N80Z mosfet is rated for 2A and 800V. This means you could use this to drive your cable and also if cable was shorted to 265VAC, you could sense this and open the switch, and have it survive accidental short to mains power (this is very important). With CANbus and 16V/8V signaling on DALI 3P, one could support 64 devices with more like 4mA per device (instead of 2mA); and use CANbus controller built into microprocessor.
  - If you drop master controller output current while signaling "0" (e.g. to 50mA if cable is < 12V) to reduce wasted power, and you have 64 devices that each draw 4mA (256mA), then these devices would load down the cable and cable voltage would drop toward 0,

and it would not move back up quickly when released. A remedy might be for devices to drop current consumption during signaling (i.e. < 12V). This is all a bit tricky and needs more consideration.

  - Another possibility is DALI 3P work Master Controller outputs 24V/1A instead of 16V/250mA, and "0" signally is 10.0V (not 0V). You still waste high current during signaling "0" (e.g. 1A/10V/10W burned by switch).
  - If devices report to Master Controller their maximum power consumption, then Master Controller can limit power to this lower number (with added ~50% buffer), instead of a fixed 250mA. This would help reduce power wasted during "0" signaling.

- If your microprocessor is on the DALI side of isolation, DALI device maximum power consumption is 16V/2mA while transmitting "1", and we are transmitting "1" only 16% of the time (e.g. CANbus with NRZ bit stuffing); then average maximum device power consumption would be 0.32mA (2mA*16%). If one feeds the 16V DALI voltage into a 75% efficient buck converter (e.g. ~$.25 total parts cost based on $0.14 #AP65111AWU) that draws 0.32mA average and reduces voltage 5-to-1 (16V to 3.3V); then output would be 3.3V at 1.2mA (0.32*5*75%). Also, if we sleep the processor 90% of the time on 0.5mA while the CANbus controller runs in the background, then we would have 8mA while running (1.2mA = 0.5mA*90% + 8mA*10%), which is probably sufficient.

  Researchers would need to search microprocessors to find candidates that support low power sleep, low power operation, and CANbus operation while microprocessor sleeps. For details, search this file for "low power processor". Note that power consumption is a function of clock rate. How many clock cycle's does one need to service DALI 3P/CANbus?

  Designing a network device with the following requirements would be an exciting contribution to the planet: (a) tree topology wiring, (b) 1K to 100K bps data transfer rates determined by physical network size, (c) ~16V/~250mA signaling, (d) short circuit protected against connection to 110/220VAC, (e) no power wires (all work is done with 2 wires), (f) possibly use DALI-software protocol, (g) CANbus data layer which allows direct connection to microprocessor CANbus controller, (h) draw 0.32mA at 16V on average (e.g. which supports 8mA/3.3V operation with 0.5mA/3.3V sleep 90% of the time), (i) place microprocessor on DALI side, (j) support applications such as wall light switch input (dimmer or on/off), temperature measurement, and power on/off control via opto-couplers.

  Is this possible? One would need to look at circuit power consumption with a simulator and spreadsheet to get a better sense of feasibility. Also, one would need to do testing with low power microprocessor reference designs to get a better sense of power drawn in various scenarios. For a TINA simulation of a DALI interface, search this file for "DALI Over-Current".

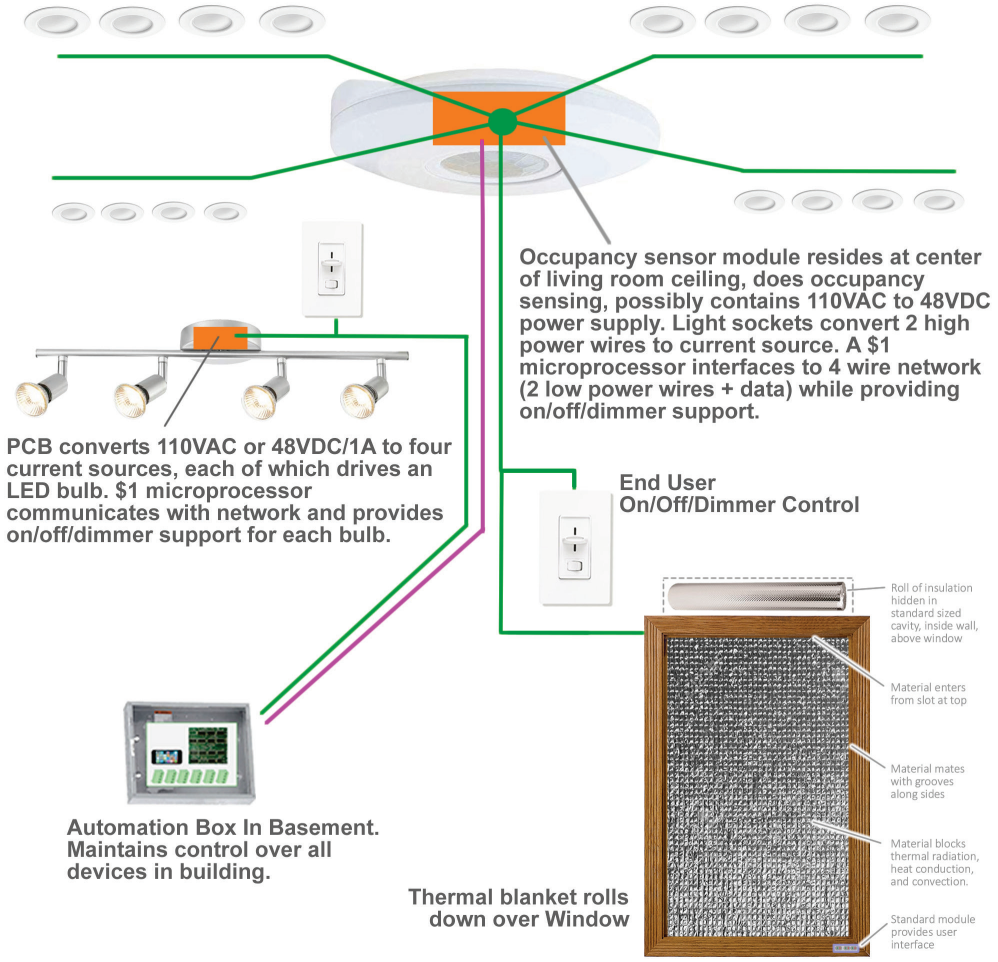
- Our DALI 3P prototype PCB (e.g. Arduino) uses one Microprocessor DIO bit to interface DALI+- wires to either Microprocessors UART Tx/Rcv interface or Microprocessors CANbus interface, and includes an FPGA on the PCB to enabling researchers to explore various options. Perhaps FPGA helps multiplex UART and CANbus signals?
- The CANbus data link layer can probably help with DALI, yet it is not clear exactly how this would work. Search this document for "CANbus" and "LED Driver" for more details.

- Researchers explore how to protect DALI wires against short to 265VAC and short to 4KV 20µSec spikes.
    - Series resistor and 320V varistor (e.g. 50A at 775V) can protect downstream ≥ 800V parts from 4KV spikes. After resistor/varistor, one could have an 800V bridge rectifier (e.g. $0.04, #ABS8TR, passes 265VAC) and then a series mosfet or transistor switch rated for ≥ 800V that opens upon > 32V overvoltage at input. This would protect components further downstream that are rated for 32V, for example.
    - Search this document for "DALI Over-Current and Over-Voltage Protection" and "Protection" for more ideas.
    - How to protect against short to 230VAC
      https://electronics.stackexchange.com/questions/429641/understanding-steval-ilm001v1-dali-transceiver-schematic-adding-protections
- Microprocessor often needs a 110/220VAC to 5VDC (or 3.3V) power supply. If power requirements are low (e.g. 1mA sleep, 10mA operate), then this power supply can be tiny. For details, see GWeinreb_Manhattan2_ResearchNotes.xlsx / "Export" / "Tiny Power Supplies for Tiny Processors".
- Research includes: analysis, sketch multiple circuits, simulate, prototype, test prototypes, write report, and publish summary in online publication (e.g. Electronic Design News, www.edn.com).
- All materials are free and open, to encourage adoption by industry worldwide.
- All software is to be written in a language that is popular in industry, such as C and C++ (compiler produces very fast optimized code and strips out unused code).
- We are not fabricating the actual IC, only developing a suggested IC for industry.
- For details, search "Electrically Connecting a Physical Window to a Network".
- Researchers are not responsible for pushing industry to adopt the proposed standard. Instead, their sole responsibility is to do good work and make materials available to others so that they can improve, build on, and utilize in anyway.

### 4.3.2 SUGGESTED PROJECT: Create *DALI DC* 2-wire 24...48VDC powered system (no 110/220VAC, no additional power wires, 2 wires total)

- Researchers create a 2 wire system similar to DALI 3P, except the Master Controller provides ~5W to ~400W total power to devices on the two DALI wires, depending on selected power supply. This system supports communications *and power* via 2 wires (i.e. *not* 2 data wires plus 2 power wires, this system does not involve 110/220VAC)
- For example, one might have a 48V power supply in ceiling and 2 low voltage wires route from here to the following locations within a room: ceiling LED light, window (for thermal cover), damper duct, temperature sensor and end user on/off/dimmer control. This is illustrated below.

  One possibly variation is room ceiling box is DALI Repeater (orange box) and master controller is in basement. Therefore, a point-to-point (not tree) high speed link (violet line) connect basement Master Controller and room Repeater.




- The DALI product family might offer 48VDC power supplies in 7 varieties: 100mA, 250mA, 500mA, 1A, 2A, 4A and 8A.
    - A 100W/48V/2A power supply sells for $20 (https://www.digikey.com/short/p2qj3n) and a 400W/48V/8A power supply sells for $60 (https://www.digikey.com/short/p2qwp3), for example.
    - A 48V/8Amp power supply might power 40 LED lights that are each 10W (8A * 48V = 400 Watts), for example.
    - Many switching power supplies burn "quiescent" current when there is no or little load (e.g. 50W wasted in heat with 400W power supply and 10W load). We prefer power supplies that are greener and throttle their switching frequency as a function of load to reduce quiescent power and maintain efficiency with small loads.
    - DALI Power Supplies must not make any noise (no buzzing).
    - If you run 4Amps 20meters through 18gauge wire, you will get a 1.7V drop total (both wires) and waste 7W (3% given 48V power supply), for example (e.g. 4A*48V = 200W, twenty 10W LED light bulbs).
        - Note that one sees this issue with 110/220VAC as well. For more details, see "12V Wall Bus Power Supply" in file GWeinreb_Manhattan2_ResearchNotes.xlsx.

        - It would be best for one to have primary power supply in central location and then have wires eminent in multiple directions, to reduce distance between power supply and device.

- Each Device master descriptor data block contains parameters that specify how much current the device draws: (a) while sleeping, (b) processor awake yet peripheral is OFF, and (c) peripheral is ON. Peripheral is something driven by DALI cable, such as LED or Motor.
    - Master controller keeps track of loads at each device, and any requests that exceeds power capabilities results in returned error code ("Not-Enough-Power-On-Cable-To-Satisfy-Request"). For example, if you have a 1Amp master controller (50 Watts), eight LED 10W lights installed on cable, and a command is sent to turn on 6$^{th}$ Light, the Master Controller does not try to turn on light and instead returns an error code.
    - This is a bit complicated since some Device PCB's may not know how much power they will need. It might be a function of which LED bulb is placed into a socket. Motors moving windows might draw 4-times more power for 1 second while window is being unstuck.
    - Some Devices might have a maximum allowed power by their MOSFET and might report this in their device descriptor, yet later, when the device is used, it might operate with significantly less power.
    - An LED controller might be set up to drive an LED string and report that power, yet the LED might not be install in the socket, and therefore actual power would be very little.
    - If a command is given to dim an LED to 50%, then power consumption would drop.
    - A window thermal cover motor might only run for several seconds after receiving a command.
    - If Device buck converters have variable current and voltage limiters, their power needs might change at runtime, as opposed to being reported once in a device's lifetime when it first joins the network. If they do change, then other network devices need to be updated.
    - Accurately characterizing, monitoring and controlling power is tricky.
- Each device utilizes a small buck converter power supply that converts 48VDC to 5V to power a microprocessor, for example. Parts cost for a small power supply is $1.50. For an example design, see "Output: 5V/16mA, Input: 7VDC to 56VDC, Buck Converter" in file GWeinreb_Manhattan2_ResearchNotes.xlsx.
- Each device should have a current limiter that limits how much it loads the cable (e.g. if it is set up to drive 10W LED's, it might limit current to 15W), to make sure it does not load excessively. We do not want one faulty device bringing down the entire network.
- System supports tree topology wiring, 5 nine's reliability (99.999%), device processor sleeps when not being used.
- If one does signaling via AC coupled sinewaves, and also hangs switching powers supplies on a long cable, then the sines will be disrupted by the switching. Subsequently, one might have trouble maintaining reliability via this method. Also, we want device to sleep and not draw much power while listening (i.e. not do millions of math operations per second to pull signal out of noise). It is not clear how AC sines can be implemented with high reliability and low power while listening.

- Let's look at a possible signaling method using a switch (e.g. mosfet) on each Device that shorts cable data wires to indicate a logic "0", similar to DALI. We will look at this, run the numbers, and see what we get. This is just one example approach.
    - Master Controller puts out 48V at full 8A current (or as required by devices) when signaling "1", and Device drops this via mosfet to 40V to signal going to a logic "0".
    - Controller sees < 42V and then moves to outputting 40V instead of 48V.
    - A logic "0" is considered to be < 42V on the cable.
    - Device puts 8A (max) load on cable *only* while slewing to 40V (e.g. for 4.5uSec on 33K bps cable).
    - To move back up to 48V, Device pulls cable momentarily down to 32V (e.g. for 4.5uSec on 33K bps cable, 30uSec bit time), Master Controller sees this, and moves output back to 48V.
    - 8A * 40V = 320W across Device switch (MOSFET), which is very big, yet this is for a short time (e.g. 4.5uSec). If signaling "0" often, one might be slewing 10% of the time. Dissipating 32W average (320*10%) is still too much. If we instead worked with 1Amp max (48V*1A=48W), then we could support five 10W LED lights, yet we would still see 4W average dissipated by switch, which is still too much.
    - In summary, it is not clear how one can do this w/o burning too much power while signaling.
- One might look at Signaling from the point of view of Power. If Maximum power from power supply is X (e.g. 200W) and Devices report they will draw a maximum of Y, and Master Controller sees Z being drawn; then one can look at how many additional watts a device might draw while signaling to clearly be seen as signaling.
    - Let's assume the controller has accurately characterized power consumption by each device (which changes w/ time). The controller could set its current limiter to that amount plus a buffer. Then, signaling device draws more current to exceed this limit, to cause voltage to drop. It reduces voltage (e.g. 48V to 40V) at a maximum rate (e.g. 4.5uSec on 33K bps cable) to signal to the Master Controller to drop voltage to 40V.
    - Device MOSFET ends up dissipating the additional current needed to get to controller set point. For example, if 55W is being drawn by devices and Master Controller sets current limit to 80W, then our signaling switch needs to draw 35W (80-55) while driving cable down to 40V (i.e. only during fall time since Master Controller switches to 40V output as soon as it sees 40V on cable).
    - If we are doing CANbus NRZ signaling, we might see device telling controller to move from 48V to 40V or move from 40V back to 48V (e.g. via 32V dip) to occur 128 times in a 128bit packet (alternative 0's and 1's throughout packet). Let's assume 20% of the time is between packets. This works out to Device driving big current approximately 10% of the time. If we slow down the bit rate by 3-fold, then we can move this down to 3.3%. Ok. 35W * 3.3% = 1.1W average power dissipation for Device Mosfet while it is signaling. 1W is manageable; however, there might be cases when additional load is more than 35W.
    - This would require a Master Controller that monitors voltage and current to see when it needs to dial down or up its voltage.

- Device power supplies (e.g. buck converters) would need to be able to maintain their constant output given an input that moves from 48V to 40V, dips down to 32V, and then back up to 48V.
- If four 10W LED's turn on at the same time, they might put an additional 40W load onto the cable. Master Controller would need to be able tell when additional load is coming from devices or signaling.
- Perhaps we do not need to force cable down to 40V when signaling "0"? Perhaps instead device adds additional load (must be added at a controlled rate to avoid ringing) and Master Controller switches to 40V as soon as it sees this additional load. In other words, Master Controller looks for this additional load (not < ~42V) to consider there being a ''0" signal. Perhaps this reduces the time that Device MOSFETS drive the cable?

- We have looked at slowing down the rise/fall times to eliminate ringing. Another thing we can look at is "Let it Ring" yet only a small amount (e.g. +-1V). If Device shorts 48V to ground, for example, then Device MOSFET will see high current for a moment. Also, Master Controller should see this (i.e. maximum current flow) and then perhaps that is the signal for it to change voltage level (e.g. from 48V to 40V). When all devices see < 42V, they know logic "0" has been signaled. When the Device shorts together the DALI data+- wires (e.g. 48V), it will cause it to ring and overshoot. Yet if it only does this for a short moment, then perhaps the ringing is not so big (< +-1V with 1uSec shorting perhaps)? One can play with a simulator to see if they can get this to work. Perhaps shorting DALI data occurs for a 1uSec period with a 33K bps cable (3% of the time)?
  - How much RFI does this create (probably a lot)?
  - If you draw 8A in 1uSec then you will probably get a big current spike that inductively couple to other wires.
  - It takes electricity 0.6uSec to travel 100meters in wire. If 8A is moving in a wire and it is turned off at the source, the electrons in the cable are still moving when the source is disconnected. The electrons hit the destination and often result in a voltage spike (i.e. called "inductive kick").
  - Device capacitance might want to hold DALI cable at one voltage. If DALI voltage dips then capacitors in Device power supply might affect the DALI cable. Yet not when the voltage drops, due to series diode between DALI wires and capacitors. However, when voltage goes back up (e.g. from 40V to 48V), DALI cable will be effected by Device capacitors. This needs to be worked out (e.g. mandate current limiter? DALI specifies maximum device capacitance by rule?).
- If there is a collision and multiple devices signal with tiny big current spike, then controller would need to make sure the wire-AND cable feature was observed. This is a bit tricky. If cable is at 40V (logic "0") and Device A signals to go back up to logic "1" at the end of the bit, and Device B is signaling 3uSec later to drop to logic "0", then the controller might get confused.
- It is not clear if we can implement DALI with 2 data/power wires *and* achieve the following goals: not waste too much power, be 5 nine's reliable with many switching power supplies along a long cable, support processor sleep, and low cost implementation. If that is the case, we recognize it (sooner than later) and look at more wires.

### 4.3.3 SUGGESTED PROJECT: Create *DALI DC* 3-wire 24...48VDC powered system (no 110/220VAC, 3 wires total)

- Perhaps one needs to consider 3 wires instead of the previously discussed 2 wires. The 3 wires might include two 48VDC power+- wires and one DALI 3P-like signaling wire. In this project, we design a 3wire DALI Power *plus* Data system.
  - In this scheme, controller might output 16V/30mA (or 16V/250mA) on signaling wire, and devices pull low to signal a logic "0". They would still need to adhere to the 15µSec rise/fall with 10Kbps (≤100meters), 4.5µSec w/ 33K bps (≤30meters), etc.; to avoid ringing.
  - One can flip the Data+- wires in DALI 2 and the signaling will still work due to a bridge rectifier. One could still do this with the two 48VDC Power+- wires (i.e. bridge rectifier between cable power+- wires and device 48V/COM). However, one would not want the signaling wire to be transposed with either of the Power+- wires. In other words, the installation technician should not confuse the 3 wires.
- Perhaps we call this "*DALI DC*", which is the same as DALI 3P, except it assumes there are 3 wires (e.g. 48V Power+-, Data+) and power supply is a DALI 48V (or perhaps we also allow 24V?) power supply (e.g. 16 to 54VDC at device) that is also a network device (i.e. power supply itself attaches to DALI data+ wire so it can tell the system what it is doing).
  - 48VDC might be helpful at receiving power from Batteries or from Solar, without going through a 110/220VAC inverter.
  - Perhaps DALI 3P controller products support DALI 3P regular (supplies data+- wires, any power) or, you add a 48VDC power supply and you build a DALI 48V Head that manages a 48VDC powered network.
  - Perhaps DALI 3P Master Controller product is set up to support devices powered by 24VDC, 48VDC, 110VAC, 220VAC, or Flex Power (changes at runtime). For details on flex, search this file for "Power Supply with 85...265VAC *or* 16...54VDC Flexible Input".
- One can also consider adding a 4$^{th}$ wire, which could be used as a "Data-" wire instead of utilizing Power- for the data return path. With 4 wires, the cable would contain: 48VDC Power+- and Data+-.
- If working with 3 wires, one might implement Data+ signaling with 16V/250mA or 16V/30mA. However, if working with 4 wires, then one could also consider RS-486P-Wire-And, discussed previously.
  - If one signals with 3wire 16V DALI, then Data- would be attached to Power- at Master Controller, and Data- and Power- would be at different voltages at each device, due to voltage drops along Power- wire.
  - If one signals with 4wire 16V DALI, one can reverse DALI+- wires (due to bridge rectifier IC) and one can be less influenced by voltage drops on Power- wire.
  - 3wire involves less installation labor and wire cost, when compared to 4wire.
- Standards bodies are influenced by industry and gov't building requirements. Therefore, they might want to add an Earth Ground wire. This could be used to shield the cable, shield the device (attach to enclosure), detect ground fault (look for voltage drop) and reduce shock hazard in the

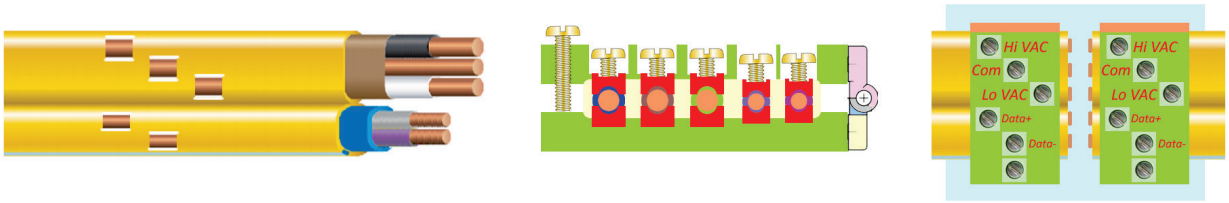
event that electricity breaks through insulation (we want it to flow through ground instead of person).

- To keep this additional wire from becoming another return for current, one might label this "Ground Fault Shield" instead of "Earth Ground", and encourage designers to ***not*** connect circuit common to Shield. If one does, it reduces ground noise 2-fold by adding an additional ground wire (assuming Earth Gnd and Power Return are both same sized wire); however, it also creates voltage drops along this wire and makes it difficult to detect ground fault. One could implement GroundFaultShield as a drain wire connected to foil that wraps around conductors.
- One could look at having Data- and GroundFaultShield share the same wire. In this scenario, we would have 4 wires: Power+, Power-, Data+ and (Data-/GroundFaultShield /EarthGnd).
- We would prefer device not attach GroundFaultShield to DeviceCommon and return current through it, yet some might do this. Also, electrician might attach GroundFaultShield to Earth Ground in a room, and have current flow through this wire. One would need to think about how to handle this.

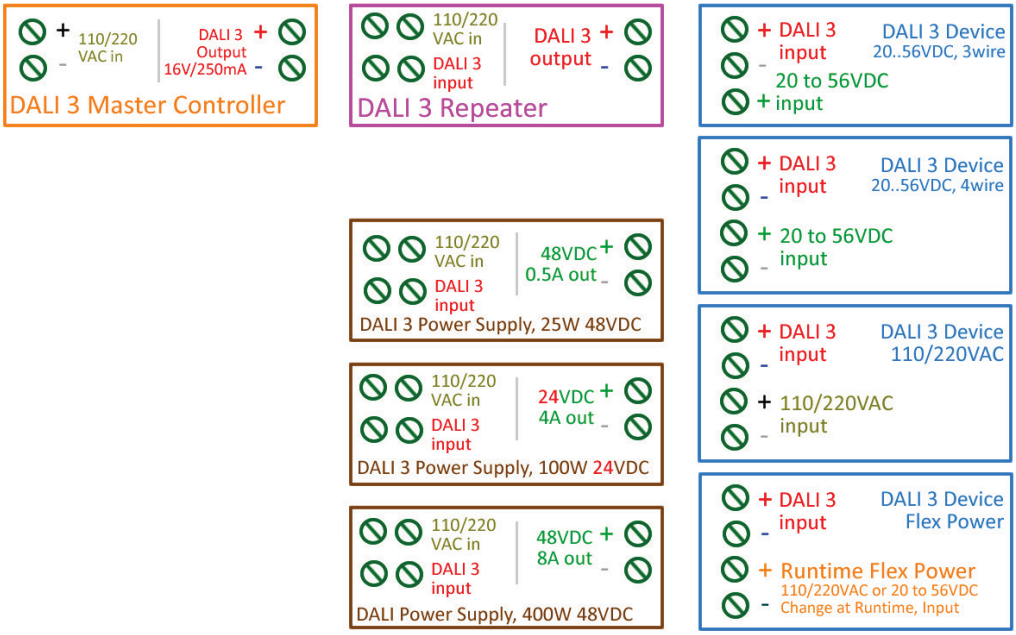
### 4.3.4 SUGGESTED PROJECT: Create *DALI Flex* 5-wire System that supports AC *or* DC, runtime changeable

- *DALI Flex* devices are powered by 24VDC, 48VDC or 110/220VAC; and power can change at runtime. Subsequently, one can power the network directly from batteries (48VDC), solar (48VDC) or Grid (110/220VAC); and switch at runtime.
  - A buck converter power supply that converts 16…54VDC to microprocessor 5VDC (or 3.3VDC) could be adapted to also support 110/220VAC. One possible technique is a "clamp" that only allows < 54VDC to enter buck converter (e.g. ~25% of the time buck converter runs on drooping capacitor disconnected from 110VAC sinewave). A tiny 1206 drooping capacitor can support a microprocessor, yet larger loads (e.g. 10W LED) require a much larger capacitor and therefore would require a 110/220VAC capable switching converter. For details on a $1.50 flexible 3.3V/16mA power supply, search this file for "Power Supply with 85...265VAC *or* 16...54VDC Flexible Input".
  - LED and Motors could be powered by switch-mode power supplies that are powered by one of 16…54VDC or 110/220VAC, by several different methods.
  - Device power supplies would need to make sure LED lights did not dim during switchover (e.g. from 48VDC to 110VAC). Power supply would need to make sure switchover occurred seamlessly without delay
    - We do not want power turned off (e.g. to 0Volts) for 250mSec at switchover, for example.
    - Power supply might wait for 110/220VAC zero crossing before switching over from VAC to VDC; or switch over when VAC sinewave was at the VDC level, to avoid sudden changes of power voltage.
- *DALI Flex* might work with a 5 wire cable: Power+-, Data+-, and EarthGnd. Power+- might be one of 110VAC, 220VAC, 24VDC, or 48VDC (or also 12VDC); and change at runtime.

- Perhaps *DALI Flex* devices include a special fast-install connector, described here: https://www.edn.com/proposed-led-wired-iot-standard-can-reduce-energy-use-part-3/



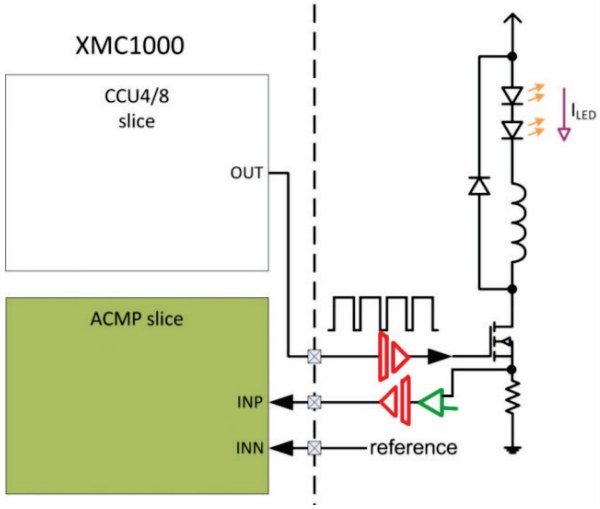

- Below is illustration that shows a possible DALI 3 Product Family. Some devices support flex power, whereas others are less flexible. This does not reflect the Earth Ground issue, which is tricky.




  - Family Includes: 16V/250mA Master Controller (orange), power supplies (brown), Devices (blue), and Repeater (violet).
  - Repeater module moves high speed data point-to-point, for example, from basement to room with many nearby DALI devices (e.g. room contains 20 devices within 30meters running 33K bps, and Repeater attaches room to basement Master Controller 100meters away).
  - DC power supplies (brown) provide DC power at different voltages (i.e. 24VDC, 48VDC) and power (e.g. 25W …400W). Power Supplies also have A DALI interface, enabling the system to monitor voltage, current and power. If there is a problem, we want the system to know how to proceed (e.g. "power supply voltage is too low and current is also too low therefore power supply needs to be replaced").
  - We show several device products (blue), each with different power supplies inputs (e.g. 16 to 54VDC, 110/220VAC, and Flex Power which can change at runtime). For details on flex, search this file for "Power Supply with 85...265VAC *or* 16...54VDC Flexible Input".
  - The devices shown in the above illustration are all compatible with each other since they talk the same language along the "DALI 3" (red) screw terminals.

### 4.3.5 SUGGESTED PROJECT: Create DALI 2 device with low-power processor powered by DALI wires

In traditional DALI, a microprocessor is optically isolated from DALI+- wires and a 110/220VAC to 5V (or 3.3V) power supply powers the microprocessor on the far side of optical isolation. In this initiative, we power microprocessor with DALI+- wires and use opto-coupler to drive external devices such as a 110/220VAC on/off TRIAC switch. The advantage is one does not need to build a protected 110/220VAC to 5VDC power supply for microprocessor (which cost money, adds size, and consumes power). Newer processors involve less power and make this more possible.




- Researchers create a DALI 2 device that utilizes a low power microprocessor which connects to, and is powered by, DALI+- wires. This interfaces to an optically isolated TRIAC IC that is capable of turning on/off 110/220VAC devices, such as 110/220VAC fan in duct, or a 110/220VAC light bulb.
- Researchers explore on/off/dimmer control of a string of LEDs (e.g. 10W, 25V, 400mA) via optically isolated (e.g. #EL817 $0.10) LED driver (e.g. #BCR450 $.24) where resistor at LED driver sense pin sets current and LED string is powered by 8V to 27VDC (or higher voltage), as pictured above. In this illustration, dotted vertical line shows isolation, where 110/220VAC is on the right and DALI+- wires are on the left.
    - One (less accurate) approach is current sense resistor powers opto-coupler diode directly, via series resistor instead of comparator (green). If you drop 1V across current sense resistor and, and 20V total across LED's then you would lose ~5% (1/20) of your energy in the current sense resistor (e.g. 500mW out of 10W), for example. This is probably too much. Perhaps one could reduce current sense voltage 10-fold and amplify it?
    - One could look at something similar to the above, yet current sense circuit puts 0.6V to 1.0V through opto-coupler diode, which results in 0.6V to 1V on DALI+- side, which is compared with a programmable reference voltage, in an attempt to have a programmable current regulator (instead of fixed).
    - In the above illustration we show comparator in green. It is not clear how this is implemented since we do not have a power supply on this side of isolation. This requires further consideration.
    - One might want accurate programmable LED current regulation (e.g. opto-coupler drives 8bit DAC on 110/220VAC side which is compared (green) with current sense resistor voltage), however, that would probably require a 5V or 3V power supply on 110/220VAC side of opto-couplers, and if you install that, then you might as well place microprocessor on far side of opto-couplers. In this project, we look at the more simple things that one can do with a processor on DALI+- side of opto-couplers (e.g. measurement, on/off control, simple fixed current control).

- The above illustration shows a simple switching power supply. It is possible a fly-back configuration would be better suited for this application.

- Explore possible circuits which drive LED's powered with 110/220VAC (e.g. buck converter implemented with Xmc1xxx/Xmc4xxx processors). Is it possible to do this in a low cost manner with microprocessor on DALI+- side of opto-couplers or does this require 110/220VAC-to-5V power supply for some reason?
  - Page 22 to 24 of the following brochure summarizes LED Driver options.
    https://www.infineon.com/dgdl/Infineon-Power_and_Sensing_Selection_Guide_2018-SG-v00_00-EN.pdf?fileId=5546d4625607bd13015621522aa012cb
  - Xmc Processors in Power Conversion Applications:
    https://www.infineon.com/dgdl/Infineon-APP_PowerConversion_XMC_in_Power_Conversion_Applications_XMC-TR-v01_02-EN.pdf?fileId=5546d4624cb7f111014ceb7af35d268e
  - For more details, search this document for "Xmc1302 DALI Slave Reference Design", "DALI Slave Reference Design" and "LED Driver", "Infineon LED Driver Product Summary", and "Infineon Processors".

- If your microprocessor is on the DALI side of isolation, DALI device maximum power consumption is 16V/2mA while transmitting "1", and we are transmitting "1" only 50% of the time (e.g. DALI Manchester encoding); then average maximum device power consumption would be 1mA (2mA*50%). If one feeds the 16V DALI voltage into a 75% efficient buck converter (e.g. ~$.25 total parts cost based on $0.14 #AP65111AWU) that draws 1mA average and reduces voltage 5-to-1 (16V to 3.3V); then output would be 3.3V at 4mA (1mA*5*75%). Also, if we sleep the processor 70% of the time on 1mA while we wait for a DALI bit, then we would have 10mA while running (4mA = 1mA*70% + 10mA*30%), which is probably sufficient.

  Researchers would need to search microprocessors to find candidates that support low power sleep and low power operation. For details, search this file for "low power processor". Note that power consumption is a function of clock rate. How many clock cycle's does one need to service DALI 2?

  Designing a network device with the following requirements would be an exciting contribution to the planet: (a) DALI 2 (i.e. 1200 bps, tree topology, 16V/250mA), (b) short circuit protected against connection to 110/220VAC (search this file for "DALI Over-Current"), (c) draw 1mA at 16V on average (e.g. which supports 10mA/3.3V operation with 1mA/3.3V sleep 70% of the time), (d) place microprocessor on DALI side, (e) support 2-wire applications (no 110/220VAC power wires) such as wall light switch input (dimmer or on/off) and temperature measurement, (f) and support 110/220 power on/off control via opto-couplers.

  Is this possible? One would need to look at circuit power consumption with a simulator and spreadsheet to get a better sense of feasibility. Also, one would need to do testing with low power microprocessor reference designs to get a better sense of power drawn in various scenarios.

- For examples of low power microprocessors with CANbus, search this file for "STM32L4", "Xmc1404" and "low power processor".

- In this initiative, researchers focus on: electrical signaling, circuit power requirements, circuit costs, CANbus controller feasibility, and CANbus data link layer. Networking layers above data link layer are not covered in this initiative, and are handled elsewhere.

- Explore adding stepper motor control, without 110/220VAC-to-5V power supply. Is this possible?
- Some functions are better handled with a microprocessor optically isolated from DALI+- wires and closer to 110/220VAC. For the various functions listed above (search "Target Devices"), specify which can be handled by a processor connected directly to DALI+- wires, and which require an optically isolated microprocessor plus 110/220VAC-to-5VDC power supply.
- Search this document for "dimmer", "motor" and "low power" for more information.
- All software is to be written in a language that is popular in industry, such as C and C++ (compiler produces very fast optimized code and strips out unused code).
- All materials are free and open, to encourage further development and adoption by industry worldwide.

### 4.3.6 SUGGESTED PROJECT: Add Comprehensive Motor Support to DALI 2

- Develop software extension to DALI 2 standard that handles motor related devices (i.e. on/off fan, variable speed fan, damp in duct, window thermal cover). These can utilize groups, fade and variable amplitude features -- similar to lights.
- One can consider adding over-current, over-voltage, and/or over-temperature protection to the DALI interface. For an example, search this file for "DALI Over-Current and Over-Voltage Protection".
- Somfy is an example of an existing system of motorized curtains and blinds. This is similar to our proposed window thermal cover; yet we are looking at more thermal insulation, support for 1" to 2" thick motorized solid foam blocks that cover windows, residential support, and non-proprietary standards. Somfy is controlled by Philips and many non-Philips companies are not inclined to participate. Several Somfy products consume 24VDC (0.5A to 2A per motor, 12W to 50W) whereas others consume 110VAC (0.5A to 4A per motor, 50W to 800W). Quickly closing a large heavy curtain in a commercial building requires a decent amount of power.
- Researchers propose a standard 5pin connector for window thermal covers that includes: DALI+- wires, Earth Gnd, and two power wires.
- Below are several power options:

  a) 110/220VAC (each device accepts either 110VAC or 220VAC yet not both)
  b) 110VAC only (power goes to motor without passing through power supply)
  c) 220VAC only (power goes to motor without passing through power supply)
  d) 16 to 54VDC
  e) Flex power that supports 16 to 54VDC *or* 110/220VAC

- For an example of an *isolated* power supply with 24V/30mA and 5V/30mA unregulated output, and 24 *to* 265 VAC *or* VDC input, click here. This reference design does not provide enough power for a decent sized motor, and is not very efficient; yet does show one way of implementing flex power.

  For details on a $1.50 flexible input, non-isolated, regulated, 3.3V/16mA output power supply; search this file for "Power Supply with 85...265VAC *or* 16...54VDC Flexible Input". This design is not enough power for motors; yet perhaps it could be adapted to support more power? For

general information on flexible power, search "Flex Power" in file GWeinreb_Manhattan2_ResearchNotes.xlsx / "Wallbus".

Also, one might consider two internal power supplies. One that takes 16 to 54VDC *or* 110/220VAC and converts to a fixed DC voltage for motors (e.g. 48VDC). And then a second tiny power supply that converts the 48VDC to 3.3V or 5V for a microprocessor. For an example of a small $0.75, 48VDC-to-3VDC converter, click here.

There is an argument *against* flex power, which is motors often accept a fixed 110VAC, 220VAC or 24VDC. And an easy way to build a power supply is to not build a power supply and instead feed power directly into motors, via a transistor or MOSFET switch. And have a tiny power supply that converts input power to microprocessor 3.3V or 5V.

Also, there is an argument *for* flex power, which is that it pushes one into converting power from the input voltage, which means one can then support 110VAC or 220VAC with all products (as opposed to having two of every product, one for 110VAC and one for 220VAC).

Researchers can look at what it takes to modify a 110/220VAC design to also support 16 to 54VDC input. For an example power supply that converts 85...265VAC to 5V/2A (10W), click here. The voltage after the input filter is approximately 120…370VDC. If this was instead detected as 16…54VDC, one could possibly redirect it via MOSFET or transistor to a different winding on the transformer. In other words, this design could be adapted to support lower voltage DC.

Researchers study costs associated with the 5 above options. Can one support flex power without much additional cost? If so, it becomes more interesting.

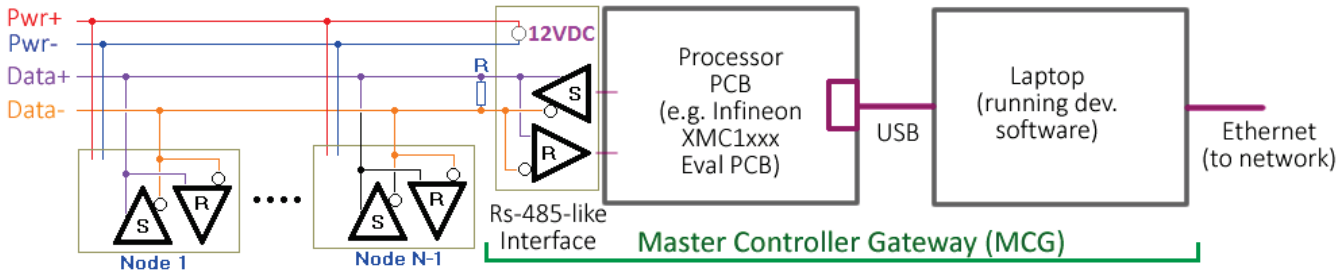
For example AC-to-DC and DC-to-DC power converters, click here. For example 5W to 50W DC-to-DC, converters, click here.

For details, search "Electrically Connecting a Physical Window to a Network".

- All software is to be written in a language that is popular in industry, such as C and C++ (compiler produces very fast optimized code and strips out unused code).
- All materials are free and open, to encourage further development and adoption by industry worldwide.
- Researchers are not responsible for pushing industry to adopt the proposed standard. Instead, their sole responsibility is to do good work and make materials available to others so that they can improve, build on, and utilize in anyway.

### 4.3.7 SUGGESTED PROJECT: Build Low Cost DALI 2 and 3P 16V/250mA Master Controller & Device

Identify free and open DALI 2 master controller & device reference design, and build on top of it.

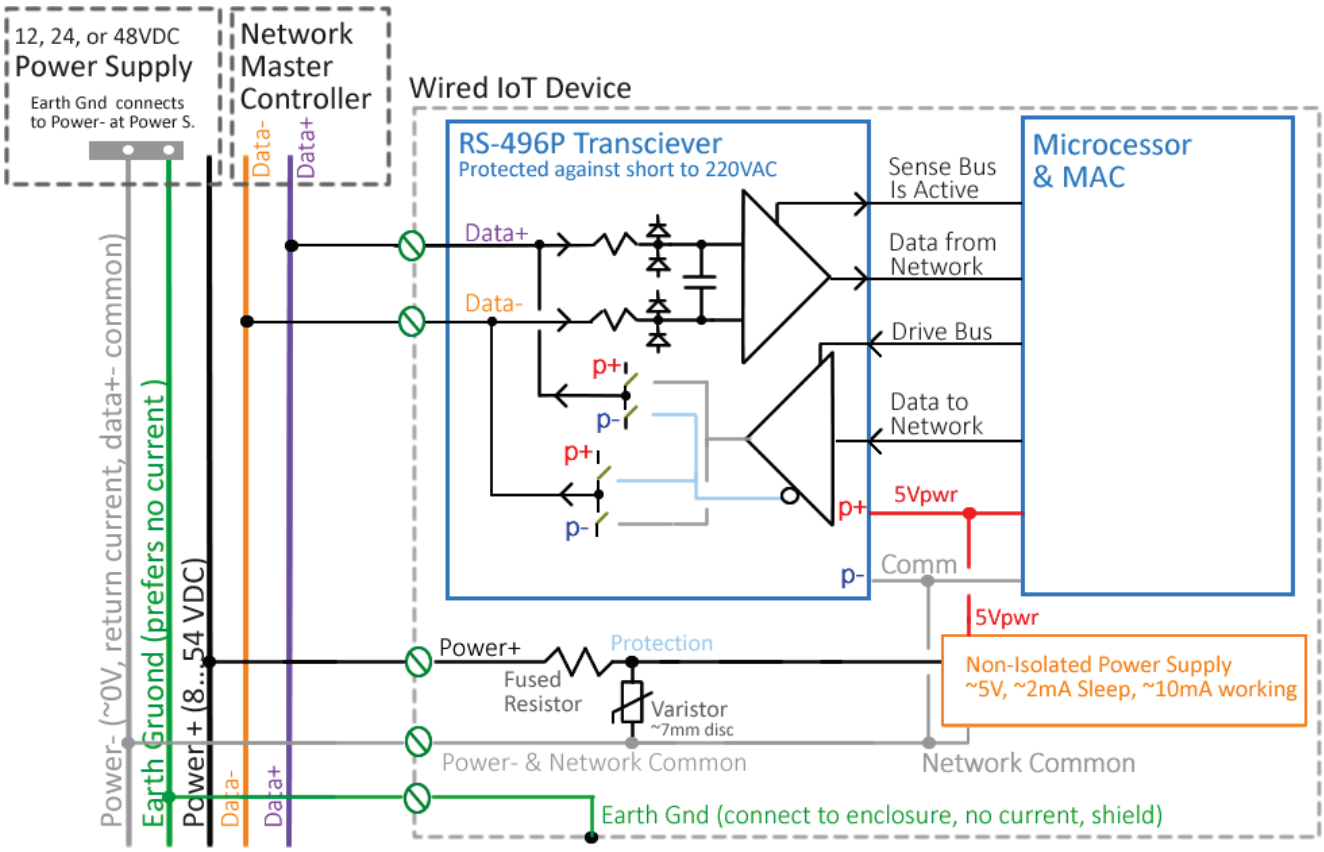



- Place free and open DALI 2 Master Controller software onto CONTROLLER hardware, and place free and open DALI led device software onto DEVICE hardware; and establish communication via 16V/250mA physical layer.
- This can be used to support research projects identified in this document.
- One can switch out Uno shields as needed to support DALI 2 Master Controller and Device; and to support DALI 3P Master Controller and Device.
- Researchers focus on: electrical signaling, circuit power requirements, circuit costs, CANbus controller feasibility, CANbus data link layer, and implementation of DALI 3P. Networking layers above data link layer are not part of this project, and are handled elsewhere.
- Implementation of DALI 3P involves switching out DALI 2 interface PCB (connect to microprocessor UART) and replace with DALI 3P high speed interface PCB (connects to Microprocessor CANbus CAN_H/CAN_L interface).
- Note that a buck converter can take 16V/1mA and convert it to 3.3V/5mA. Therefore, one might have one power supply shield that does this, and one w/ regular linear LDO power supply, for research purposes. For details, search this document for "Power Supply Research".
- Search "Build Universal Controller and Device Development Platform" and "DALI Slave Reference Designs" for support materials.
- All software is to be written in a language that is popular in industry, such as C and C++ (compiler produces very fast optimized code and strips out unused code).
- All materials are free and open, to encourage further development and adoption by industry worldwide.

### 4.3.8 SUGGESTED PROJECT: Build DALI RS-486P/CANbus Master Controller and Device (not 16V/250mA, no 110/220VAC)

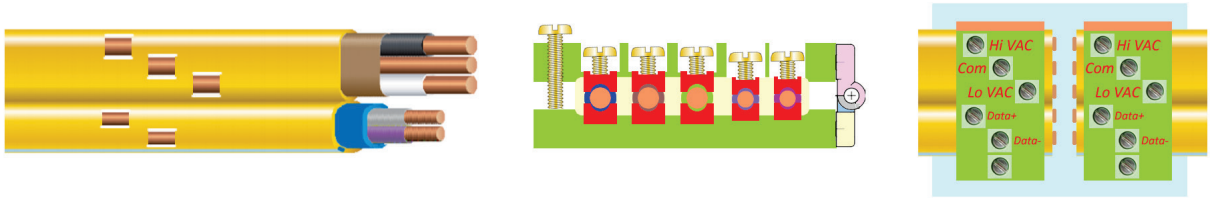
Build DALI 3P/CANbus system on top of non-isolated RS-486P-wiredAnd physical layer *instead of* the traditional DALI optically-isolated 16V/250mA. Utilize the built-in microprocessor CANbus controller instead of the traditional DALI signaling protocol.

- Cable includes two RS-486P Data+- wires and two 8…54VDC power+- wires (no 110/220VAC power).




- For an example $0.75 buck converter that converts 8…54VDC to 3…5VDC, click here. We offer a wide range for input power since buck converters take care of that for us. 8 to 54VDC allows one to power the system with a 12V, 24V, or 48V power supply.
- DALI and CANbus expect wire-AND (1 and 1 is 1; otherwise 0), therefore this version of RS-486P must support wire-AND. For details, search this document for "Wire AND entails releasing bus". The "Develop New Electrical Signaling Standard" project should produce an Arduino shield with the needed RS-486P-wire-AND interface.
- If possible, circuitry is protected against damage in the event of an accidental wiring error (e.g. connect data+- wires to 265VAC). See "Design Current Limiting IC that protects wired IoT devices" for ideas. For an example of a $1.50 power supply that is protected against accidental short to 110/220VAC, search this file for "Power Supply with 85...265VAC *or* 16...54VDC Flexible Input".
- Recall that RS-486P does not utilize termination resistors and it does not connect 16V/250mA to ground each time you transmit a 0 bit. Subsequently, it uses little power (it only uses 3mA when driving capacitance) and is therefore good w/ many small low power devices.
- RS486P is simpler, lower cost, and involves fewer components than the traditional DALI isolated 16V/250mA. The advantage of RS-486P over traditional DALI hardware is lower cost (opto-isolation DALI interface circuit not needed), lower power, and more network bps (I think). The disadvantage is less common mode voltage rejection due to no opto-isolation (i.e. DALI might be better with longer distances), and sensor devices cannot rely on signaling wires for power. RS486P might easily support 1K to 100K bps data rates; whereas research might show faster data rates to be problematic with 16V/250mA.

- If a RS-486P transceiver IC supports +-7V difference between device COM signals (e.g. -7V to +12V at transceiver IC input pin with respect to transceiver IC COM pin), and you had a 10V drop on cable power- wire between devices (power- attaches to device COM), then you would exceed your transceiver IC CMV specification, for example. Subsequently, transceiver IC's with more common mode voltage support would be helpful. Yet at what cost? This is one consideration a designer must make when moving from 16V/250mA isolated to non-isolated 486P. Isolation supports any voltage drop along the power- wire, whereas 486P is vulnerable to excess voltage drops on power- wire, even for a short period. Example Cases:
    - An 18 gauge wire (0.021ohms/meter) drives a 4Amp load 30m (90ft) for a total 5V drop (2.5V per wire, 5 = 4*0.021*30*2). 48VDC at 4Amps (200W) would support twenty 10W LED lights, for example.
    - A 14 gauge wire (0.008ohms/meter) drives a 15Amp load 66m (200ft) for a 16V drop (8V per wire, 16 = 15*0.008*66*2).
- Due to common mode voltage considerations, RS-486P might be preferred by industry when connecting many small loads (e.g. LED lights, damper in duct, window cover stepper motor); yet might be unpopular with longer distances and larger loads.
- DALI is typically not protected against short to 265VAC. If RS-486P was protected against short to 265VAC, then industry might warm to it.
- To reduce wiring errors and increase installation speed with many wires inside one cable, one could utilize a special fast-install connector, described here:
  https://www.edn.com/proposed-led-wired-iot-standard-can-reduce-energy-use-part-3/




- We do not mix RS-48x with 110/220VAC for several reasons:
    - If you run 110/220VAC through a bridge rectifier then circuit COM will be aligned (0.6V away) to sine when it is in the negative part of the cycle, and one cannot connect this to RS-486P transceiver COM pin w/o violating cmv specification (i.e. you do not want your RS-48x transceiver IC ground pin at -370V with respect to Earth GND).
    - If you use the Earth Gnd wire to connect together RS-486P transceiver IC COM pins, then this will not work well if the Earth Gnd is wired incorrectly, which is often the case with traditional wiring.
        - If one has a special 5 wire ribbon cable, they can avoid wiring errors more easily.
        - One could label Earth Ground as "Earth Gnd Signal Common", or something like that, and require that it be wired correctly.
        - One would probably prefer for power current to not flow on Earth Gnd Signal Common (since it would produce voltage drops); however, 110/220VAC loads often connect this wire to device enclosure and often connect 110/220VAC neutral to enclosure as well (which results in current flow and voltage drops on earth ground).

- If you attach 110/220VAC neutral to RS-486P transceiver IC COM pin *via a series diode*, you will not have common mode registration 50% of the time, which is not good.
- One option is to attach the 110/220VAC neutral wire to the transceiver IC COM pin directly, without a series diode. However, a wiring error with Neutral swapped with Hot would cause the 486P circuit to not function (e.g. preferably without permanent damage). Also, this type of wiring error would mean that your device COM trace would be attached to the mains hot wire (e.g. 265VAC w.r.t. Earth Gnd). To avoid a shock hazard, one would need this trace to be physically isolated (i.e. not connected to enclosure). If one uses 5 wire ribbon cable, a hot/neutral swap wiring error becomes less likely.
- To get RS-48x to work with 110/220VAC consider: (a) add a 48x common mode reference wire which connects together the GND pins of 48x transceiver ICs, (b) do not pass current on this reference wire, (c) utilize *isolated* AC-to-DC power supply, (d) connect reference wire to power supply secondary COM nodes, (e) hope a fault condition does not put current on reference wire, and (f) hope all wires are wired correctly.

- RS-486P and DC Power:
  - The Power- wire is connected directly to the RS-486P transceiver IC COM pin (no series diode), and a wiring error (e.g. swap power+- wires) is bad.
- DALI allows one to swap 110/220VAC neutral/hot wires with each other; and swap DALI+- wires with each other without any degradation of the system (i.e. it still runs normally due to bridge rectifiers). However, permanent damage occurs if you attach 110/220VAC hot to DALI+ wire or to DALI- wire; therefore, DALI is vulnerable to wiring errors, unless over-voltage and over-current protected.
- It is possible industry will want something called "DALI DCV", which is similar to 110/220VAC 16V/250mA DALI, yet instead you have RS-486P data+- wires paired with 8V to 54VDC power, or something like that. Buck converter power supply can take that power and create 5V or 3.3V for a microprocessor, for $0.75 in parts cost. This would help people avoid DC-to-AC inverters and avoid AC when working with solar and/or batteries.
- It is possible industry will want something called "DALI Flex", which pairs RS-486P data+- wires with Flex Power. Flex Power *changes at run time* and is one of 8 to 54VDC power *or* 85 to 265VAC. Device buck converter (or transformer fly-back) is designed to handle a wide range of inputs. Subsequently, one can move power from solar array (DC), from battery (DC), or from grid (AC), without conversion, to save energy (and possibly save cost). This might involve a 5 wire ribbon cable, pictured above, with the following wires: Power+-, Data+- and Earth Ground. Earth Ground would probably be connected to the RS-486P transceiver COM pin, and would need to be wired correctly. Also, one would need an isolated power supply that attaches the secondary side COM signal to the earth ground (i.e. one cannot maintain RS-48x common mode reference through a bridge rectifier).
- One can switch out Arduino Uno shields to configure development platform in different ways (e.g. RS-486P to Microprocessor CANbus pins, obtain power via 48VDC shield, obtain power via 110/220VAC shield, etc.). For details, search "Controller and Device Development Platform".
- Researchers can demonstrate the following:
  - Traditional DALI 2 software and traditional DALI 16/250mA signaling.

- DALI 3P/CANbus with high speed 1200 to 100K bps DALI 3P 16V/250mA signaling.
- Same as above, yet RS-486P signaling instead of 16V/250mA signaling.

- This demonstration shows an alternative to DALI hardware. We present accurate cost models, list advantages and disadvantages; and then let industry decide what to do with it.
- All software is to be written in a language that is popular in industry, such as C and C++ (compiler produces very fast optimized code and strips out unused code).
- All materials are free and open, to encourage further development and adoption by industry worldwide.

### 4.3.9 SUGGESTED PROJECT: Design Low Cost Repeater

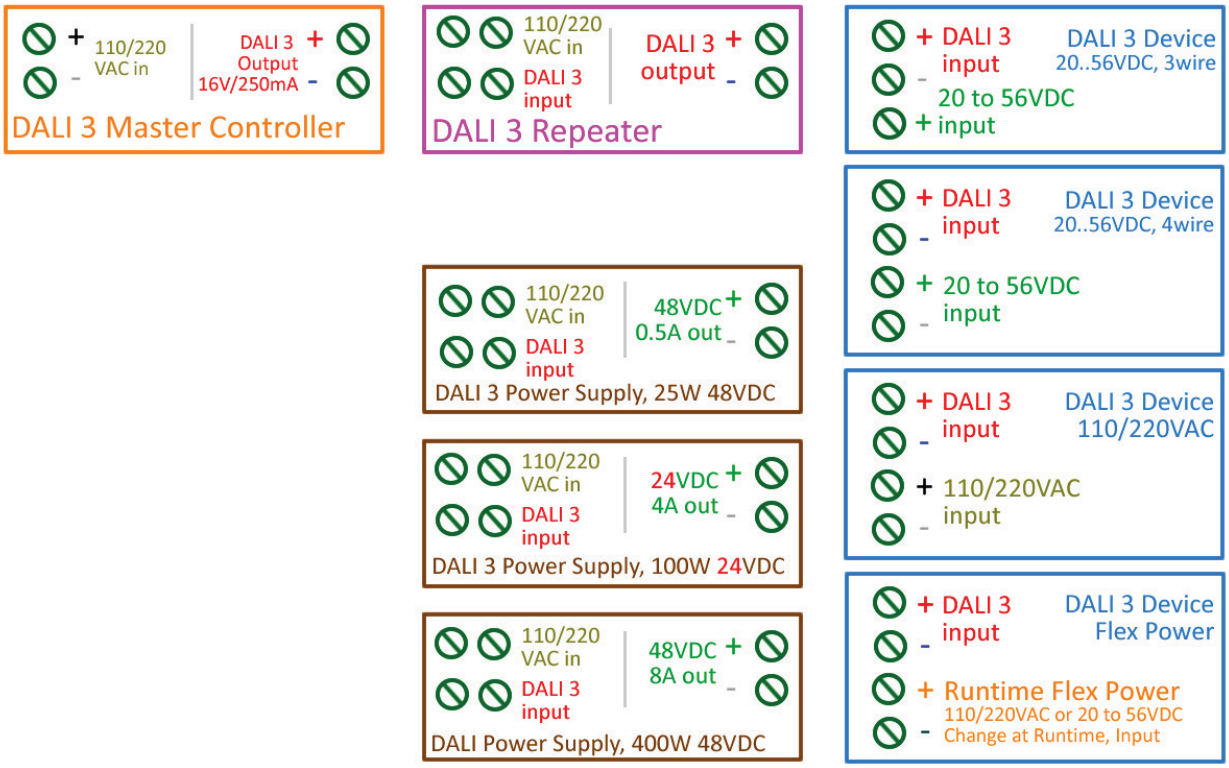
Imagine a commercial building that has twenty 400 sq meter rooms (20 x 20m, 60 x 60ft), for a total of 8000 sq meters; and the owner places a DALI 3P 33Kbps ≤30m network in each room.

Also, a ≤ 100kbps ≤ 300meter point-to-point cable is routed between each room and a central location in basement. This is terminated at each end to avoid reflections (e.g. via 120ohms). There are only two parties on each of these extension cables, one at each end.

- Researchers design a low cost and robust extension systems that extends Master Controller in basement to Repeater in room. Repeater contains a DALI 16V/250mA power supply.

  [DALI Master Controller] [LONG CABLE DRIVER] ----- Long cable ---- [REPEATER] ---- DALI devices in room

- A DALI 3 product family is illustrated below. Each element can be combine in any combination, plug-and-play. The DALI 3 Master Controller (orange) "DALI 3 output" can enter anything marked "DALI 3 Input". Devices (blue) provide DALI 3 input, and are pictured with several power supply variations. A Repeater Box (physically close to devices, communicates point-to-point with faraway Master Controller) is shown below in violet color. Master Controller is capable of driving devices directly, or via Repeater.




- It is not clear how signaling would be done on the long point-to-point cable. We want optical-isolation between DALI Input and DALI Output at Repeater since the common mode voltage difference between basement and room is unknown. We could consider something similar to the CANbus physical layer with 120Ω termination resistor at each end (60Ω total) that holds line at 0V position (logic 1, recessive) and transmitter drives current through it to move it up (e.g. 1.44V across 60ohms = 24mA, logic 0, dominant). Basement Cable Driver module might attach data-wire to earth ground at that one position. Repeater device could have isolated power supply (e.g. #RFM-0505S, $1.12, 200mA, 125mW quiescent no load power) that powers the tiny circuit that interfaces to the long cable. Subsequently, one could route 2 wires between basement and room (mains neutral and earth ground not included).
    - If DALI 2 or 3P network in Room (e.g. 400m$^2$) was at a recessive state (e.g. 16V, logic 1) then repeater would not touch long cable going to basement (it stays recessive at <0.5V logic 1). If Room network goes to dominant (<4.5V, logic 0), then repeater pulls on long cable to make it dominant (>0.9V, logic 0). If master controller in basement wants to drives DALI Room network to 0V, then long cable driver would lift long cable to 1.4V via 24mA (24mA * 1.2V = 30mW, logic 0, dominant). The rise time for the long cable would be < 1.5µSec, and would not ring due to termination.
    - If long 200m (600ft) 20pF/ft (12K pF total) cable was driven only by 120Ω termination (return back to 0V), then time constant would be 12K pF * 120 = 1.4µSec (12e3 *1e-12 * 120/1e-6), which is reasonable for 100k bps speeds. However, if you want to run further, then driving cable with 120 ohm resistor might not be the best choice, and one might prefer an RS-485 driver that will drive more mA (e.g. 60mA) into a long cable. In summary, 120Ω termination might drive a ≤200 meter cable to a recessive state (logic 1), yet not a longer cable.
    - A $3.45 #ADM3056E *isolated* CANbus transceiver at the repeater might be helpful: https://www.analog.com/media/en/technical-documentation/data-sheets/ADM3056E.pdf

- A $0.31 #NCV7344 CANbus transceiver might be helpful at both repeater and driver ends. Notice how this burns up 10mA when not driving cable and 70mA when driving cable, which seems like a lot. One might pair this with opto-couplers (e.g. #EL817 $0.10) and isolated power supply (e.g. #RFM-0505S).
  https://www.onsemi.com/pub/Collateral/NCV7344-D.PDF

- One might also look at RS-486P-wireAnd between Master Controller in basement and Repeater, and then 16V/250mA between Repeater and room Devices. However, it is not clear that 5mA would be enough to drive long cable capacitance at fast rates. For example, 5mA driving a long 200m (600ft) 20pF/ft (12K pF total) cable would result in a 7 µSec rise/fall time, which is too slow for 100K bps (12e3 *1e-12 * 3 / 0.005, dt = C dv/ I). For more info, search this file for "RS-486P".

- Note there is a subtle difference between the traditional RS-485; and DALI cable "short to 0V" or "let float at 16V". If there is a collision, then DALI "short to 0V" always wins; whereas with RS-485 collisions result in unpredictable results (driver with more current capability wins). CANbus and DALI both do wire AND. "1" and "1" always result in "1", "0" and "1" always result in "0", and "0" and "0" always result in "0". The CANbus controller expects this behavior and uses it to determine who controls the bus. Therefore, one might prefer a DALI-like or CANbus like physical layer on this extension cable in order to implement wire-AND.

- Another option is 16V/250mA circuit is used to drive long cable point-to-point; and both driver and receiver communicate in a DALI-like manner where driver supplies power to receiver circuit (no isolated power supply at Repeater). This would be like the regular DALI system where the master controller powers the slave's circuitry and opto-couplers move data downstream across isolation barrier. If repeater circuit draws 10mA and you run this 300meters on 24gauge cable, you would only drop 0.5V on each wire of the cable, which would be ok. In this scenario, the full 250mA would not be utilized. Note that "16V/250mA" means that 250mA is the maximum current output. In actual practice, current is determined by load. In the point-to-point scenario, you would drive 10mA when doing nothing (to power receiver circuit) and then 0 to 60mA only when driving cable capacitance (rise back up to 16V after being forced to 0V).

  - One problem with this is parallel termination draws too much power (16V / 60Ω = 266mA, 4 Watt). One might consider series termination, yet then you get voltage drop across 120ohm series resistors when slave (receiver) shorts wires with transistor. One might consider AC termination, yet it is not clear if this would work reliably in all cases.

  - One might consider running this circuit with less voltage yet it would still burn a lot of power (e.g. 4V instead of 16V, 4V/60Ω = 70mA, 250mW continuous). It is not clear how one can terminate and have transmitter provide power to receiver without losing lots of energy. For details, searching this file for "AN#903" and then view Section #3 and #5 of this document.

- If you used the 16V/250mA electronics in master controller to drive point-to-point extension system, then you could *adapt* it to instead output high impedance (60Ω termination pulls to 0V) during logic "1" (~99% of the time), and 1.4V/24mA during logic "0" signaling.

  - This would support 120Ω resistors at both ends of cable. To protect these resistors against accidental short to 16V/250mA, one could place current limiter ICs in series with 120Ω resistors (e.g. current limiter IC, $0.21, 50mA, 120V, #NSVC2050JBT3G). However, this would not be protected against short to 265VAC. One would need high voltage mosfet for that. Perhaps 16V/250mA electronics can emulate 120Ω resistors via some kind of feedback while

listening? Perhaps diode to Ground can eliminate ringing < -0.6V? Perhaps termination via series resistor and capacitor can provide sufficient termination, survive accidental short to 16V/250mA, and survive accidental short to 265VAC?

- The cable typically sits at logic "1", therefore outputting high impedance (60Ω pulls to 0V) during logic "1" would not burn power while idling.
- The extension cable would not provide power to repeater receiver; therefore one would need isolated power supply in repeater to support receiver (e.g. #RFM-0505S, $1.12, 200mA max, 125mW quiescent no load power).
- The 16V/250mA electronics can control slew rate, and can therefore reduce RFI emissions and RFI susceptibility (e.g. 4.5µSec rise/fall time with 33k bps, or 1.5µSec with 100k bps).
- To support driving 1000ft cable capacitance (20pF/ft, 20K pF total) 1.4V in 1.5uSec, one would need 18mA, for example (I = C*dV/dt, 0.018 = 20e3*1e-12*1.4/1.5e-6). Also, to drive 60Ω termination, one would need 24mA (1.4V/60Ω).
- If Master Controller slews this 1.4V/24mA as needed for network speed, it is possible that one could place an analog isolation amplifier (e.g. #LOC110, $1) at Repeater, to transfer this voltage to local network (instead of having Repeater microprocessor set local network slew rate).
- Remember that extension cable rise/fall times can be fast and not ring due to termination; however, local DALI network (tree topology in room, not terminated) needs slew rate limiting to avoid ringing (e.g. 4.5µSec rise/fall times w/ 33K bps, ≤30m network size). We are looking at the extension cable speed to match the room network speed, to reduce circuitry in the controller. For example, one might set up a room run at 33K bps (≤30m) or 10K bps (≤100m), and the cable extension would run that th
- In order to support CANbus collision management, we would want for delay across Repeater to be short.
- The Master Controller 16V/250mA electronics contains support for slew rate control (reduces emitted RFI) and also is protected to 265VAC, which is helpful.

- The DALI standard could mandate that all Master Controllers must support Repeater extension, to encourage users to fast run long distances fast (e.g. 33Kbps) point-to-point isolated; and then run as-fast-as-possible given final network size.
- The Master Controller could be a 2 screw terminals box or a 4 screw terminal box.

A. 4 Screw Terminals: 2 screw terminals for nearby DALI devices (16V/250mA); and also 2 screw terminals for point-to-point communication with Repeater.

- It might be lower cost to have 4 screw terminals since one can then hardwire 120Ω and CANbus controller IC to the Repeater output, which is easy to do. A $0.30 transceiver (e.g. CANbus transceiver IC or RS-485 transceiver IC) can easily drive a point-to-point cable.
- One might protect the DALI 16V/250mA network against short to 265VAC, yet not the CANbus point-to-point extension system, to save cost.

B. 2 Screw Terminals: 2 multipurpose DALI output screw terminals attach directly to a local DALI network or route to a Repeater via extension cable; where the controller detects function and acts accordingly.

   - Internally, both the 16V/250mA circuit and the repeater circuit (e.g. RS-486P or 120Ω/24mA) are attached to these screw terminals, yet only one is enabled at a time.
   - Master Controller detects function, or function is set by end user via software ("auto", "repeater", "local network").
   - If running point-to-point, 120Ω termination resistor are attached to screw terminals.

- In many cases, the DALI Repeater has a 110/220VAC power input that powers its 16V/250mA voltage/current source; and powers an isolated DC/DC converter that powers its Receiver circuit (since this product's DALI Input is optically isolated from its DALI output).
- The Repeater system needs to insure that collisions are handled properly (i.e. two devices talk at same time, or Master controller talks exactly when Device talks). We are attaching microprocessor CANbus controllers to the DALI 3P 16V/250mA cable, and these CANbus controllers need to synchronize to some extent, to properly manage collisions in a predictable manner. For example, the Master Controller and a Device at the far end of a repeated network need to be able to talk at the same time, and have the system manage the collision per CANbus. Getting this to work correctly and reliably is a little tricky.
- Programmable slew rate limiting when driving Room network (i.e. DALI 3P 16V/250mA) might require a microprocessor at the Repeater. Perhaps Master Controller would be assigned CANbus address 0 (most priority) and Repeater address 1?
- If a Device in the room drove the DALI cable at a different speed (e.g. 1200, 10K, 33K, 100K), the repeater would support that speed (i.e. extension cable would signal at that slow speed).
- What does it take to protect extension cable against accidental short to 265VAC? See "Design Current Limiting IC that protects wired IoT devices" for ideas. 800V transistors that open in the event of over-voltage and/or over-current might help. This is a bit complicated.
- What are the advantages and disadvantages of having Master Controller in Basement with extension/repeater Vs. Master Controller in room with Ethernet connection to basement?
  - Ethernet cable and connectors are flimsy (8 tiny wires), and one therefore might want to keep it within one box (e.g. multiple Master Controllers connected together via Ethernet cable and Ethernet router inside one solid metal Automation box in basement).
  - If connection between room and basement automation box fails, then one needs to limp along and they might do this better if master controller is in room and can assist devices without connection to local area network (perhaps).
  - It would be nice for light switch to turn on/off lights in the event of failure from any of the following: Repeater, Master Controller, and connection to high speed local area network.
- All software is to be written in a language that is popular in industry, such as C and C++ (compiler produces very fast optimized code and strips out unused code).
- All materials are free and open, to encourage further development and adoption by industry worldwide.

## 4.4 BACnet

### 4.4.1 SUGGESTED PROJECT: Build BACnet Controller and Device that uses wired RS-486P/CANbus; and do the same with DALI 16/250mA CANbus Signaling

BACnet typically uses RS-485 physical layer and MS/TP token passing data layer to manage media and collisions. In this project we implement BACnet both on our wire-AND RS-486P/CANbus platform and our DALI 3P signaling/CANbus reference platform. Both platforms utilize CANbus as the data layer.

- Using Arduino Uno shield modules, one can set up a prototype Master Controller and Device hardware and establish communication between each. One can start with several simple devices.
  - LED Light control (on/off/dimmer)
  - Temperature measurement
- Place free and open BACnet Master Controller software onto CONTROLLER hardware, and place free and open BACnet Light control software onto DEVICE hardware; establish communication via the following traditional setup: RS-485, MS/TP, microprocessor UART Interface.
- Modify above system to support RS-486P with wire-AND feature and microprocessor CANbus interface. This involves *trading* MS/TP token passing *for* CANbus data link layer (i.e. one needs to write software to do this).
- Do the same as above, yet implement DALI 3P 16V/250mA CANbus as the physical and data link layer.
- Comments:
  - One of the first steps is to do a survey and locate bug free and well documented free and open BACnet software.
  - The advantage of CANbus wire-AND over traditional RS-485 MS/TP is it utilizes the CANbus Controller built into microprocessors, which makes it much easier for processors. Other advantages are lower cost, tree wiring topology instead of daisy-chain, no termination, and consumes less power. Also, DALI 3p provides optical isolation between data wires and 110/220VAC.
  - This research encourages BACnet to consider 486P/CANbus or DALI 3P/CANbus as an alternative to 485/MSTP.
  - One might still use Token Passing to decide who talks. However, doing this over CANbus instead of UART reduces the probability of collision.
- Researchers present accurate cost models, list advantages and disadvantages; and then let industry decide what it wants to do.
- All software is to be written in a language that is popular in industry, such as C and C++ (compiler produces very fast optimized code and strips out unused code).
- All materials are free and open, to encourage further development and adoption by industry worldwide.

## 4.5 KNX

### 4.5.1 SUGGESTED PROJECT: Build KNX Controller and Device that uses wired RS-486P/CANbus; and do the same with DALI 16/250mA CANbus Signaling

This project is similar to the previously described BACnet project, yet researchers work with KNX (i.e. implement KNX with 486P/CANbus and with DALI 3P signaling/CANbus).

## 4.6 LonWorks

### 4.6.1 SUGGESTED PROJECT: Build LonWorks Controller and Device that uses wired RS-486P/CANbus; and do the same with DALI 16/250mA CANbus Signaling

This project is similar to the previously described BACnet project, yet researchers work with LonWorks (i.e. implement LonWorks with 486P/CANbus and with DALI 3P signaling/CANbus).

## 4.7 EnOcean

### 4.7.1 SUGGESTED PROJECT: Build EnOcean Controller and Device that uses wired RS-486P/CANbus; and do the same with DALI 16/250mA CANbus Signaling

This project is similar to the previously described BACnet project, yet researchers work with EnOcean (i.e. implement EnOcean with 486P/CANbus and with DALI 3P signaling/CANbus).

## 4.8 ZigBee

### 4.8.1 SUGGESTED PROJECT: Build ZigBee Controller and Device that uses wired RS-486P/CANbus; and do the same with DALI 16/250mA CANbus Signaling

This project is similar to the previously described BACnet project, yet researchers work with ZigBee (i.e. implement ZigBee with 486P/CANbus and with DALI 3P signaling/CANbus).

## 4.9 Device Object Model Research

*Please review to the previous "Device Object Model Research Overview" discussion before viewing the below suggested research projects.*

### 4.9.1 SUGGESTED PROJECT: Develop Extensive Software Object Models for Common Devices

Create free and open Device Extensive Object Models (DEOM) for popular devices within a building. Each object model is a set of classes with fields and methods, some of which are marked immutable (never changes, read only). Popular devices include things like: LED light controller, wall light switch, fan on/off/variable controller, window cover controller, temperature sensor, and duct damper controller. Work is free and open to encourage collaboration and adoption.

- Chapter 31 and 32 of the following document suggests an extensive object model for a common light bulb product. If one downloads this PDF to a computer (File, Save As) and opens with an Acrobat viewer, an easy to use left-side Navigation system exposes the table of contents.

  *Manhattan 2 Blueprint Book (180 pages)*
  http://www.ma2.life/doc/plan/Manhattan_2_Blueprint.pdf#pagemode=bookmarks

- Many existing standards were developed when processors were smaller and could not support e excessive information. In this project, we break from that traditional view, and add many fields, some of which are immutable (never changes).

- Device data takes time to move from device to master controller; therefore, master controller maintains a library of permanent information (immutable) from each device, and each device is typically identified with a 64bit unique serial number (e.g. EUI-64, 32bit MAC address). It might take 0.1 to 3 seconds to "check-in" a device when it first *joins* a network. This involves transmitting immutable data, which is data that never changes for a given unique serial number. After this data is captured, anyone on the network can access it without using the slow device-to-controller network. Libraries of device immutable blocks are copied throughout the network to reduce network traffic (e.g. to Master Controller, Gateway boxes, Client computers).

- The proposed Device Extensive Object Model (DEOM) is designed to support connection to multiple standards (e.g. BACnet, KNX, LonWorks, ZigBee, and DALI). For example, if DALI devices respond to light fade over 3 seconds command and BACnet devices do not have this feature, then the proposed extensive object model would support light fade feature. Our extensive object models is the *union* of features found in multiple existing standards.

- A DEOM can help connect different protocols (e.g. KNX to DALI).
  - If you have N protocols and you want code to interface any protocol to any protocol, you would need N*N software functions. However, if you have code that connects each Protocol to DEOM, and code that connects DEOM to each protocol, you can translate with N*2 software functions.
  - Translation (i.e. "tunnel") might occur at the device, master controller, gateway, high level application computer, or client device.
  - Translation code is a bit tricky and might only be done with the most popular devices (e.g. LED light controller, fan on/off/variable speed controller).
  - For more details, search this document for "Write Software that Connects Common Devices of Different Protocols".

- One can build on top of C software written by G. Weinreb that is 20% finished. This software packs immutable C++ class properties into one binary big block and exports this block to a library system on a master controller.

- All software is to be written in a language that is popular in industry and can easily access binary data, such as C and C++. This compiler produces very fast optimized code and strips out unused code (not python).
- All materials are free and open, to encourage further development and adoption by industry worldwide.
- Researchers are not responsible for pushing industry to adopt the proposed standard. Instead, their sole responsibility is to do good work and make materials available to others so that they can improve, build on, and utilize in anyway.

### 4.9.2 SUGGESTED PROJECT: Create Library system for maintaining Multiple Devices

Researchers create a Device Library System (DLS) that mains data for multiple physical devices. The DLS software might physically reside in a master controller, gateway, high level computer, or client device. Data is stored on Flash memory (or hard disk), is moved to RAM for processing (open command), and possibly saved back out to flash (save). Data is either immutable (never ever changes) or read/writeable. If immutable, it is checked into the library when the device is first added and then becomes read only.

- When a device is initially added to the DLS library (done once in device's lifetime), it is assigned a "deviceInlibraryIndex", which is a value that ranges from 1 to the number of devices that have been added to the library (i.e. instance number). This value increments as one adds devices to the library.
  - If we Remove a device, then all assigned deviceInlibraryIndex values stay the same, and instead, the Removed device's deviceInlibraryIndex becomes non-functional and returns an error code when used.
  - The DLS considers a Master Controller (product that manages a wired network) as a "device" as well, and checks it into the library when first discovered.
  - Master Controllers are assigned a masterControllerInlibraryIndex (1 to # of master controllers) when they join, so that one can iterate through them.
  - Devices are assigned multiple persistent instance values when they join. Below are several examples.
    - deviceInlibraryIndex — described previously
    - masterControllerInlibraryIndex — described previously
    - deviceWithinMasterControllerNetworkIndex — 1 to # of devices within *each* master controller's network
    - BACnet_deviceInlibraryIndex — 1 to # of BACnet devices checked into DLS Library
    - KNX_deviceInlibraryIndex — 1 to # of KNX devices checked into DLS Library
    - Zigbee_deviceInlibraryIndex — 1 to # of Zigbee devices checked into DLS Library
    - DALI_deviceInlibraryIndex — 1 to # of DALI devices checked into DLS Library
    - LEDcontroller_deviceInlibraryIndex — 1 to # of LED Controller devices checked into DLS Library
    - DuctDamper_deviceInlibraryIndex — 1 to # of DuctDamper devices checked into DLS Library
  - All of the above values never change; even when a device is removed, power cycled, or loses network connection. The only way these would change is if you destroy the entire library and rebuild the entire library from scratch, which is a big process which could take tens of minutes.

- This above values are local to a specific library system. If one has multiple DLS libraries on their network (e.g. one in a gateway and one in a client computer), for example, the instance values in one library would not match the instance values in other libraries. These instance values help keep track of devices within one instance of a DLS library system, yet are not used to identify devices across the network.

- When a device is added to the library, one also checks-in one immutable block of binary data (never changes, read only) that is saved to flash and is associated with the device for its lifetime. This is referred to as a device's "Device Immutable Binary Block" (DIBB), and might be 200 to 2000 bytes per device. One can pack multiple structs into this block, to maintain permanent information on the device (e.g. device protocol, device type, manufacture name, device model number, etc.).
  - DIBB's can be copied to other DSL libraries across the network, to reduce network traffic. For example, if you have 100 devices in your house, you might have a DLS library on your smartphone with DIBBs for each of those devices. Therefore, you would not need to use the network to get immutable data (never changes) such as manufacturer name.
- A device's unique serial number (e.g. EUI-64), MAC address (e.g. 32bits) and previously described indices (instance values) are permanently associated with each DIBB and these associations never change, by rule.
- The library system maintains Dictionaries and Arrays that enable one to quickly locate a DIBB via various methods. For example, a dictionary might convert unique serial number or MAC address to deviceInlibraryIndex. Or an array converts BACnet_deviceInlibraryIndex to deviceInlibraryIndex.
  - DLS Dictionaries and Arrays are stored in flash memory, and loaded into RAM to facilitate quick finding of information.
  - Adding a device might only occur once in the device's lifetime, yet accessing the device occurs many times, therefore maintaining quick methods for locating information are worth overhead that occurs when one adds.
  - When one first opens the DLS Library system, a variety of Dictionaries and Arrays are loaded into RAM memory, to allow the library system to find information quickly.
- A Device Location Record (DLR) is maintained for each physical device. This contains: unique device serial number, MAC address (supposedly unique number associated with each node), above described instance values (e.g. Zigbee_deviceInlibraryIndex), DIBB length in bytes, filename for file that contains DIBB, DIBB byteOffset location within that file, deviceProtocolType (e.g. KNX, BACnet, etc.).
  - All values within a DLR never ever change.
  - An array of DLR structs is stored in flash memory, and loaded into RAM to facilitate quickly finding information that never changes. Each DLR resides at deviceInlibraryIndex location within the array. Subsequently, one can find this information quickly given a device's deviceInlibraryIndex.
- The DLS library assumes it is executing from within a Master Controller, Gateway, high level computer, or client computer. It assumes processor includes floating point hardware and

available RAM memory and Flash memory is much larger than is needed (e.g. 2KB DIBB per device * 100 devices = 200KB, available Ram ≥ 16MB).

- If someone needs access to a device's DIBB, it is loaded into RAM memory from flash and sits in RAM possibly for a long time. Tables and headers within each DIBB enable one to find information (structs), as needed.
    - For example, let say you have a device key (e.g. device serial number, MAC address, or device IP/port number address) and you want the manufacturer's name. You hand the key to the DLS library system, the key is translated to deviceInlibraryIndex via dictionary, deviceInlibraryIndex is used to find Device Location Record (DLR) within an array, DLR specifies how to locate DIBB within files on flash, DIBB is loaded into RAM, header in DIBB provides offset to Device object (e.g. BACnet), and 'Vender_Name' is pulled out struct. All this might take several microseconds or tens of microseconds on a reasonably fast processor, provided data and dictionaries were already resident in RAM memory.
- Devices contain multiple objects (e.g. C++ classes) and we would like for the DIBB for each device to include all immutable data from *all* of those objects. In other words, the data from multiple objects is packed together into one DIBB binary block. Headers within the DIBB contain offsets that tell us where to find information. Packing and finding data is a bit complicated, and not covered in this document; yet G Weinreb has written code that does some of this. For details, see him.
    - DIBBs include a list of *objects* within each device, and values of immutable parameters within each object.
    - DIBBs are packed differently for each protocol.
    - For example, if you have 100 devices in your house with 2kB DIBB per device (200KB total), immutable data from all *objects* in all *devices* might end up on your phone to be accessed without network I/O.
- When one loads a device DIBB from flash into RAM, they can also attach C++ classes to each object, if C++ class code is available and has been compiled into the software assembly. If not, the best one can do is pass DIBB data to others in the network system who might be able to interpret it.
    - For example, one object might have a 200 byte device descriptor struct that contains several immutable (never changes) fields, one of which is 64 character manufacturer name. This struct might be packed into the DIBB along with immutable data from other objects. A user of the DLS library system might recognize this struct, and utilize this information. The DLS user might instantiate an object (e.g. C++) that has one property that is a pointer to this 200 byte struct. Therefore, methods within the class can easily access this data. From the programmer's point of view, they can access the data with something like:
      classPtr->struct_in_dibbP->manufacturer_name.
- To maintain speed, the DIBB maintenance system might create a file in Flash that is 32KB and then pack it with multiple DIBB blocks (e.g. 200 to 2000 bytes each) by writing to segments within a file (instead of creating a new file for each device). When it exceeds the 32KB size, it might create another 32KB file, and continue to pack it with data.

- If processor loses power while adding a device to the library, we do not want to corrupt the DIBB database and subsequently need to roll back the change. To facilitate rollback, one does not want to recognize the DIBB file writing until the entire operation is complete (e.g. # of valid bytes in file is not augmented until the entire operation has completed).

- DIBB data is immutable and never changes, even when one power cycles or loses network connectivity. Also, the DLS library system maintains persistent arrays of read/write structs that *do* change, and are stored in flash. To set this up, the user of the library creates a persistent array of structs (PAS), specifies the # of bytes per struct (one struct per device) and which instance variable is used to find a device's struct within the array (e.g. deviceInlibraryIndex, BACnet_deviceInlibraryIndex, LEDcontroller_deviceInlibraryIndex).
  - For example, one might maintains BACnet device IP addresses in a 32byte long struct that is maintained even after power cycling. The DLS library creates a PAS array of 32byte long structs (one per device) that are accessed via the BACnet_deviceInlibraryIndex index (i.e. one 32byte element in array for each BACnet device checked into the library). The library system creates a corresponding data file on disk, and when you add a BACnet device, it expands the data file as needed and fills with 0's. It might allocate space for 16 structs at a time (add 512bytes every 16 BACnet devices) so that one does not need to resize the file each time they add a device (only write into existing allocated space on flash).
  - The user of the DLS system opens the array to load it into RAM memory and then saves the RAM data at any time, possibly after each change to maintain data in the event of sudden power loss. The DLS system provides save methods that save one struct or saves all structs in the array.
  - The save methods resist data corruption if one loses power during save by creating a copy of the PAS array on disk, saving to the copy, and only recognize the copy if the operation completes successfully (i.e. DLS provides rollback support).
  - The DLS system maintains the entire PAS array in multiple files, to reduce the amount of overhead that occurs when one saves one struct with rollback support. For example, one might keep 16 structs in each file, and 50 devices would involve 4 different files, the last of which contains data for 2 devices.

- The DLS system supports Device Interrogation upon Joining (DIUJ) that contain values of mostly static parameters that are read when each device first joins the DLS system (possibly only once per device lifetime).
  - For example, the BACnet DIUJ struct might contain Vender_Name, Vender_Identifier, model_name, Protocol_Services_Supported, Protocol_Object_Types_Supported, and Serial_Number (i.e. a handful of parameters that are read from BACnet Device object).
  - These fields tend to not change; *however*, we are not 100% sure about this since we don't control them. Therefore, we refer to this data as "interrogation data upon joining library". These are values of parameters when the device is added to the DLS, and there is a possibility they will changed over the lifetime of the device.
  - It is up to the programmer to use this data as desired. For example, BACnet DeviceType and VenderName might be considered to be stable.

- Each device has a protocol-specific DIUJ struct (parameter values read upon device discovery) that is packed into the DLS library PAS system (persistent array of structs, one struct per device, one array per protocol). For example, the BACnet device DIUJ struct might be called "BACnet_Device_DIUJ_*ReadbackUponJoiningLibrary*".

- The DLS system also supports *Object* Interrogation upon Joining (OIUJ) which is similar to DIUJ (*Device* Interrogation), yet relate to *objects* within devices and not *devices*. The DLS system works with structs that are specific to objects within the various protocols.
  - For example, BACnet requires that all devices have a ~17 parameter "BACnet Device Description" object, therefore, the DLS system might define a BACnet Device Description OIUJ struct. We don't copy the BACnet C source code because BACnet defines this object with function calls for each parameter. Also, not all parameters are placed into a OIUJ struct since some of them change too often (e.g. DeviceStatus changes often and would not be included).
  - This involves many structs (e.g. 10 objects * 6 protocols = 60 structs) and might seem like a daunting task, and it is. Yet many of these structs are small and researchers only need to implement a few important ones in their initial work.
  - In our previously described "Develop Extensive Software Object Models…" initiative (DEOM), we talk about a universal object model that contains all features for each device (i.e. *extensive*).
    - The DEOM initiative defines a DEOM object interrogation upon joining (OIUJ) struct for each object type. And, the "Create Universal Gateway to Connect Servers/Clients of Differing Protocols" initiative, described later in this document, creates code that populates a DEOM OIUJ given a protocol specific OIUJ. For example, if the BACnet OIUJ had a manufacturer name field and the DEOM universal OIUJ had a similar field, this data would be copied.
    - The DLS maintains both the DEOM universal OIUJ, and protocol specific OIUJ's for each object. Each has their advantages and disadvantages. The DEOM struct might *not* include important information specific to a protocol. However, high level code that wants to work with multiple protocols might find it helpful since it connects to devices from different protocols (e.g. read manufacturer name).

- DIBB, DIUJ and OIUJ structs do not allocate fixed space for strings (unless it is tiny). Instead, strings are tightly packed into binary bocks, and stored offsets tell us where to find each string. This reduces wasting memory, and makes it easier to support long strings w/o truncation. For partially complete code that does this packing, see G. Weinreb.

- If one Adds a device to the DLS library system and loses power during the Add operation, the DLS system resists data corruption. This is a bit tricky. For example, if one needs to increase file size (and not lose existing data), the DSL system first creates a new file, copies the old data into the new file, and then does not recognize (commits) the new file until the operation completes.
  - If one needs to insert data into 50 *different* files each time a device is added to the DLS, for example (i.e. little segments that relate to the new device), one might not want to recognize the added data until all 50 have been added successfully. In other words, we don't want to

partially add a device to the system. Instead, we want it either fully added correctly, or not added at all (to resist data corruption).

  - If there is a power failure during an add operation, one probably needs to look for this when the library system is later opened again, to rollback some changes that might be in flux. In a worst case scenario, one tries to add a device, system loses power, the Add is not recognized when power is restored, yet all files are consistent (i.e. all parts of the system do not see that additional device).
  - If one adds a device and rebuilds an index (e.g. list of device serial numbers in numerical order), and needs to rollback, they would need to resort to the original index file. Subsequently, in one keeps a copy of all original indices and rolls back to them in the event of an unsuccessful add operation.
  - To increase speed, the DLS system supports adding a group of devices at one time, to reduce the number of times one needs to reorder a dictionary (i.e. lists are re-ordered after adding N devices, as opposed to after adding each device).
  - The DLS system contains a variety of features to resist data corruption.

- The DLS system is designed to support cloning throughout the network. For example, Gateway #1 might maintain a DLS library with 10 devices, Gateway #2 might maintain a DLS library with 20 devices, and a client computer (e.g. iPhone) might maintain a DLS library with 30 devices (union of the other two). DIBB data (immutable data, never changes) and DIUJ/OIUJ data (device and object interrogation data upon joining network, mostly currently) is copied from one to the other and maintained throughout the network.
- The DLS library system can be used in many ways throughout a network. In summary, it does the following:
  - Maintains tightly packed immutable (never changes) binary data about devices, one big binary block (DIBB) per device.
  - Maintains DIUJ/OIUJ device/object interrogation binary data that is current as of device joining network (yet might change slightly over time).
  - The DLS library does not interrupt data and does not define structs that correspond to packed binary data. That is done by others. Instead, the DLS system maintains blocks of binary data on Flash, moves them to RAM, saves them back to flash upon command, and resists file corruption when one loses power while adding a device.
  - Maintains persistent (resistant to power cycling) changeable (read/write) data in arrays of structs (PAS), one struct per device. One accesses a device's struct with one of the instance variables maintained by the library (e.g. BACnet_deviceInlibraryIndex).
  - Maintains indices to help one find a device's block quickly (e.g. deviceInlibraryIndex).
  - Assumes devices are added occasionally, yet not removed often since a "not seen" device might be turned off or have lost an internet connection. This results in "debris" within Library, which deserves further consideration (i.e. library is designed to grow easily and quickly, yet pruning is tricky).

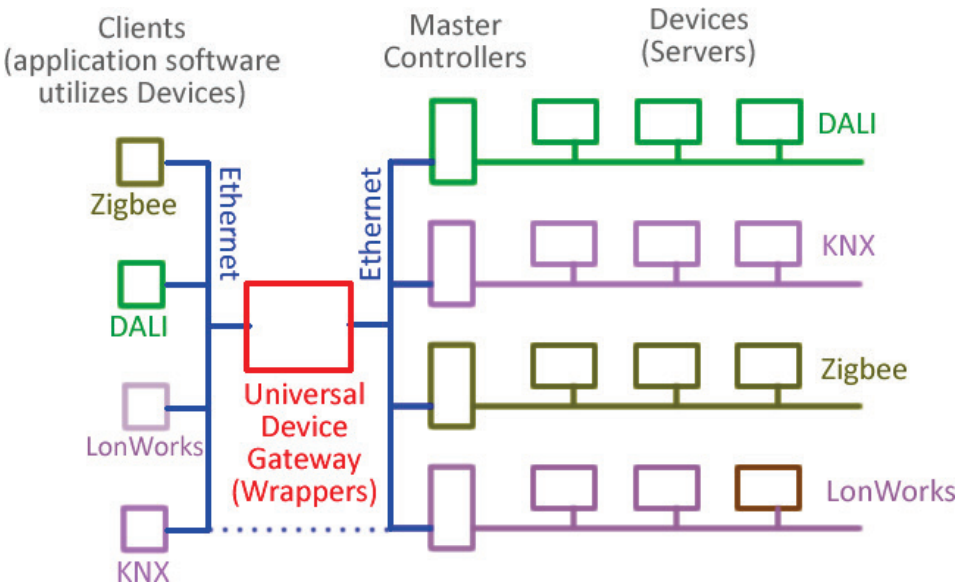
- When a device is added, its DIBB binary block is written to the master data file on disk, and Dictionaries and Arrays are updated to reflect new addition.
- Supports devices with different protocols (e.g. KNX, DALI, BACnet, EnOcean, LonWorks, and Zigbee).

- One can build on top of partially complete DLS software written by G. Weinreb that packs immutable C++ class properties into one binary big block (DIBB) and exports this block to a DLS library system.
- All software is to be written in a language that is popular in industry and can easily access binary data, such as C and C++. This compiler produces very fast optimized code and strips out unused code (not python).
- All materials are free and open, to encourage further development and adoption by industry worldwide.
- Researchers are not responsible for pushing industry to adopt the proposed standard. Instead, their sole responsibility is to do good work and make materials available to others so that they can improve, build on, and utilize in anyway.

### 4.9.3 SUGGESTED PROJECT: Create Universal Gateway to Connect Servers/Clients of Differing Protocols

Create free and open (to encourage collaboration and adoption) interface software that allows devices of different protocols to communicate (e.g. KNX high level software talks to DALI light controller device).

- In this research project we implement a "Universal Device Gateway" (UDG) box which connects hardware devices (servers) of differing type to higher level software (clients) of differing type, as pictured below. Later, our software can be adapted to run at other positions within a network, such as a network device, master controller, gateway box, high-level application computer, or client computer.



- In our previous initiative we developed Device Extensive Object Models (DEOM) for common devices (i.e. search "Develop Extensive Software Object Models"). In this initiative, we create interfaces between these and their comparable objects defined by different protocols.
    - We begin with two protocols (e.g. DALI and KNX), and then expand to more (e.g. LonWorks, BACnet, EnOcean, and ZigBee).
    - We initially support one or two simple objects (e.g. temperature measurement, LED light controller) and later expand to more objects.
    - We assume our interface software runs on a fast processor, has a fast connection to the network, includes floating point math hardware, has plenty of RAM memory, and has plenty of Flash memory.
- We developed a DLS device library system in our previous research initiative (search "Create Library system for maintaining Multiple Device"). Device are added to the library when they join the network; and the library maintains a copy of immutable data for each device, and maintains indices (e.g. dictionaries) that enable one to find data quickly.
    - The UDG gateway uses the DLS library to maintain information on all devices managed by the gateway.
    - The DLS library maintains information specific to each protocol. For example, the library maintains one struct for each BACnet device; and saves it in a persistent array of structs (PAS).
    - The library system maintains dictionaries that convert things like device serial number to library deviceInlibraryIndex. And after one obtains 'deviceInlibraryIndex', they can find much information.
    - A device with a specific serial number is only discovered once (not each time we power cycle or device is connect to network), therefore we can have much overhead (e.g. rebuild numerous dictionaries and indices with rollback support) each time a device is added to the library system.
    - If a device is physically removed from the network or is destroyed (which is rare), then the library system might receive a Destroy Device command. Subsequently, it loads associated DIBB and PAS data structs with 0's, and breaks dictionary associations. Yet it does *not* change indices (i.e. instance values) of other devices in the DLS system, since we rely on those numbers to find devices within our system.
- Discovering devices (i.e. "servers") is a project in and of itself, and is not covered in this document, yet needs attention.
- Several parameters are tightly associated with a device (e.g. 64bit serial number, 32bit MAC address), and one can use dictionaries to find a corresponding DLS deviceInlibraryIndex. Given this value, one can access a Device Location Record (DLR) which contains much information about how the device is handled within the DLS library. Data in the DLR never changes. For more details, search this file for "DLR".
- Each device is associated with a network *address* that *might* change. An example is a DALI master network controller with a programmable IP address (32bit v4 or 128bit v6) and a 0 to 63 DALI device address that is programmed manually or automatically.

Each protocol tends to maintain several different addressing parameters that enable one to find a device on the network.

  - For each protocol (e.g. DALI, BACnet), and each device, we maintain network address data in a struct that is managed by the PAS (persistent array of struct) library system. For example, a DALI device might maintain a struct with an IP address and 6bit DALI address; and a BACnet device might maintain a struct with an IP address and UDP port number. For more details on PAS, search this file for "library creates a persistent".
- We assume network address parameters are kept up to date via some method not described in this document.
- Obviously, we would prefer these not change with no notification.
- We maintain dictionaries that enable us to find a device's deviceInlibraryIndex given address parameters used by the various protocols (e.g. find deviceInlibraryIndex given BACnet IP address and port number).
- Below are several example network address parameters:
  - DALI: Short DALI network address (6bits, 0 to 63 value) and IP Address (4 or 16bytes)
  - BACnet: IP Address (4 or 16bytes) and Port address (2bytes)
  - Zigbee: Identified with 64bit MAC address (i.e. serial number) and 1byte endpoint value. This translates to an IP address and 1byte endpoint value.
- Network addressing enables us to locate a device on the network. Also, one needs to find specific objects within each device (e.g. analog input channel #4, temperature sensor #3), which is important, yet beyond the scope in this document.
- It is not clear how one broadcast one message across multiple protocols. This deserves further consideration.

- In the above illustration, we show how we work with devices of differing protocols and also we work with high level software of different protocols, and glue all devices to all high level software. This requires creating device wrappers for each protocol at both the network layer, and the object model layer.
- As described previously, DIBB's (binary block of data, one per device) contain immutable information about each device, a list of objects within each device, and the values of immutable parameters within each object. For more information on this, search this file for "Tables and headers within each DIBB".
- The UDG universal gateway creates wrappers for each device and its objects, so that they can be seen across protocols.
  - The DSL library maintains information on all downstream devices (right side in above illustration).
  - The DSL library maintains persistent array of structs for maintaining associations with each device, for each protocol.
    - For example, one struct per device contains Network layer information for each BACnet device (e.g. with device IP address and port number)

- Given the above, we create virtual instances of each device, for each networking protocol, for high level software (clients).
  - For example, if we have 25 BACnet, 35 KNX, and 45 DALI devices (105 total); then we maintain for BACnet high level software 105 virtual BACnet devices, we maintain for KNX high level software 105 virtual KNX devices, and we maintain for high level DALI software 105 DALI devices.
  - This means we need to create virtual network addresses for each device. For example, one might utilized the UDG universal gateway IP address and auto-assign a port number for each of our virtualized devices so that high level software can "see" all downstream devices at that virtualized location. If we don't do this network address translation (NAT), then message might bypass the UDG universal gateway and instead go direct to the device and become confused (e.g. BACnet client cannot send BACnet message to DALI controller).
  - If a BACnet *device* wanted to send a message to a DALI *device* (both on right side of above illustration), one might need for the message to travel down a BACnet RS-485 wire, to a BACnet master controller, to our UDG universal gateway, to a DALI master controller, down the DALI cable, and to a DALI device, for example.
- High level software (clients) affiliated with a specific protocol might want to directly access devices of the same protocol without touching our UDG universal gateway, to avoid possible disruption. Subsequently, the UDG would support this as much as possible.
  - This is a bit confusing since devices would appear in the system in two ways (direct and via UDG gateway). In some cases, the UDG gateway/DLS library system might be favored due to its automatic dissemination of immutable data throughout the network.
- Each high level protocol might view our "Universal Device Gateway" (UDG) as a Router device (or gateway), subsequently, we would need to include free and open Router, and Router configuration software, within our system, for each protocol.

- Translating object models from one protocol to another is a bit tricky. We begin with several (e.g. temperature sensor, LED light controller).
  - For each object, the UDG gateway instantiates a DEOM (device extensive object model) that supports all features within all protocols (i.e. *extensive*).
  - For each object, the UDG creates an object model wrapper for each protocol. For example, a DALI temperature sensor would result in creating a virtualized BACnet temperature sensor, virtualized KNX temperature sensor, etc. Subsequently, high level software (server) recognizes objects even if they are associated with a different protocol.
  - The above wrappers talk to the DEOM models. For example, high level BACnet software talks to BACnet virtual wrapper object and asks for temperature. The wrapper asks its associated DEOM model for temperature.
  - Each DEOM contains a tunnel between it and devices of each protocol. For example, the DEOM GetTemperature method in turn calls a specific get temperature subroutine based on the device's actual protocol (e.g. it would call GetTemperature_BACnet() if the device was BACnet and GetTemperature_DALI() if the device was DALI).

- It is not clear when the UDG universal gateway instantiates virtualized sensors and tunnels. Perhaps it does this for each, when needed.

- Portions of the UDG gateway free and open source code could be adapted by a device or master controller (e.g. DALI controller that supports 1 to 63 DALI devices) to facilitate having that device or controller support multiple protocols. In other words, the UDG source code could be compiled elsewhere within the network. Exactly how this is done is beyond the scope of this document.
- There are a variety of complexities that are not covered in this document. In some cases, researchers can find solutions to problems. In other cases, researchers document a complexity as "not supported".
  - Examples of possible complexities include: groups, broadcast message to multiple devices with mask, command to implement functions over time (e.g. dim lights over 3 seconds), data logging, and data pipes.
  - In some cases, a read request returns an "unknownValue". For example, LonWorks might ask for temperature sensor accuracy and DALI might not have that information.
- For more information on the various protocols, search this file for: "DALI Overview", "KNX Overview", "BACnet Overview", "LonWorks Overview", "EnOcean Overview", and "Zigbee Overview".
- Examples of interfaces between protocols
  - BACnet to Zigbee interface
    - Michael Newman's excellent BACnet book, section "8.2 ZIGBEE" https://www.amazon.com/dp/B00E3YAFVS/ref=cm_sw_em_r_mt_dp_U_yjnMDbAP3QF6D
    - Implementation of a BACnet-ZigBee Gateway : https://web.ics.purdue.edu/~lee992/papers/indin2010.pdf
  - BACnet to KNX
    - BACnet communication over a KNX network: https://www.auto.tuwien.ac.at/~wgranzer/bacnet_knx.pdf
    - Example Product: EZ Gateway KNX to BACnet: https://www.sierramonitor.com/ez-gatewayknx-to-bacnet
- Free and Open Source Code
  - BACnet Stack: http://bacnet.sourceforge.net/
  - Search this file for "Free BACnet C Source Code".
  - Search this file for "DALI Slave Reference Designs".
  - Search this file for "Zigbee Reference Designs".
  - Search this file for "KNX Reference Designs".
  - Search this file for "EnOcean Reference Designs".
- All software is to be written in a language that is popular in industry and can easily access binary data, such as C++. This compiler produces very fast optimized code and strips out unused code (not python).
- All materials are free and open, to encourage further development and adoption by industry worldwide.

### 4.9.4 SUGGESTED PROJECT: Add Stimulus Measurement Support to UDG Universal Device Gateway

- Researchers add extensive stimulus measurement support to DEOM universal object models and to the UDG gateway.

- Devices often include stimulus measurement hardware.
- Stimulus measurement is often associated an "object" within a device, where each device contains multiple objects. For example one device might include the following 4 objects: motor control, temperature measurement, pressure measurement, voltage measurement.

- We support many types of stimulus, including: voltage DC, voltage AC (RMS), current DC, current AC, resistance, power (W), Capacitance (pF), Inductance (µH), Frequency (Hz), temperature (C), strain (St), load (Kg), absolute pressure (Pa), differential pressure (Pa), acceleration (g), distance (m), magnetic field (gauss), infrared, ambient light intensity, ultra violet light, moisture, humidity, angular position, angular rate of rotation, velocity, earth compass (degrees), seismic shaking, ECG (heart), EMG (muscle), time, liquefied petroleum gas (ppm), ammonia gas, nitrogen oxide gas, benzene gas, smoke, CO (ppm), $CO_2$ (ppm), formaldehyde gas, alcohol gas, toluene gas, air quality VOC, air quality IAQ, $H_2$ gas (ppm), $CH_4$ gas (ppm), Ozone gas (ppm $O_3$), $NO_2$ gas (ppm), tilt (angle), On(1)/Off(0), True(1)/False(0), Open(1)/Closed(1), Rotary Switch with positions between 0 and N.
- A typical DEOM measurement object might have parameters like lastMeasuredValue, sensorMaxValue, sensorMinValue, systemNoise (amount of expected variation if one reads sensor multiple times with constant stimulus), and sensorAccuracy (maximum difference between measured value and actual stimulus). The DOEM object might maintain these parameters as 32bit floating point values in Celsius units to avoid losing information.
  - For example, client of protocol Y measures temperature from a sensor of protocol X via the following steps: a 0.5°C resolution int8 value is read from protocol X device, DOEM converts this to standards flt32 Celsius units, and then DOEM converts the flt32 value to 0.01°C resolution int16 units used by protocol Y.
  - A temperature might be represented in protocol Z as an int16 +-32K variable in 0.01°C units (-327°C to +327°C). One might consider this very sufficient, yet some cases, this might result in lost resolution or values that exceed range. If one reads a temperature sensor 1000 times and takes an average, they will often get more resolution than 0.01°C, for example. Software might want to see small temperature changes to sense how an HVAC system moves heat within a building. Perhaps this influences positions of dampers within ducts, and the speed of variable speed motors within ducts. In summary, the DOEM relies on floating point values to maintain accuracy, resolution, and range for physical quantities.
  - Translation involves dealing with different units (e.g. Celsius, Kelvin, and Fahrenheit are all temperatures) and as well as dealing with integer quantization (e.g. int16 value in 0.01°C units). For example translation code that supports many different units, see G. Weinreb.
- We support reading device sensor N times and to obtain average, which improves accuracy and resolution. For example, if you read 10bit temperature sensor one time you get a 0 to 1023 value. However, if you read it 4096 (12bits) times, and sum those values, you get a 0 to 1024*4096 (22bit) value. If you want to return full resolution, you need to return those 22bits. If one measures temperature w/ averaging and then turns on a fan and measures again, they might see a small change with this technique, to help understand thermal characteristics of room.

    - We support averaging (integration) N values yet spread out over T seconds (e.g. 20mSec or 16.666mSec). This is the frequency of the 50/60Hz power line, and the sinewave from power wires often couple into sensor signals. Averaging over this duration causes the positive part of a coupled 50/60Hz sinewave to cancel the negative part, and improve accuracy, sometimes dramatically.
- We support multiple channels within each object. For example, one DEOM object might support 16 analog input channels.
- We support the setting up of channels including: measurement range (amplifier gain), integration (averaging), filter options, sensor type selection, and offset/scale mapping. For details on this, see G. Weinreb and ask about "C# LabOffice".
- We support data logging, which involves measuring values at fixed time intervals (e.g. once a second) by device hardware, placing the values into an array, and sending them across the network. The UDG gateway might maintain 3 circular data buffers for each pipe between client and server (i.e. one buffer each for device units, DOEM units, and server units).
    - Getting this to work well involves time synchronizing different devices, which is complicated and not discussed in this document.
- When a device measures a value, it also records the network time of the measurement. Gateways, master controller and devices save both value (lastMeasuredValue and timeOfMeasurement). The system supports read request for "measured value not older than X seconds". Subsequently, if a read request is pushed into the network and a network element (e.g. master controller) sees the value as already being measured within the last X seconds, it returns the stored value, to reduce network traffic.
- We support moving data between devices (servers) and high level computers (clients), as well as between devices, across protocols (value goes from device to UDG gateway, is translated to different units, and this is transmitted to device).
- For details on how devices should characterize a sensor measurement (e.g. specify accuracy, noise, etc.), see Chapter 31 "Sensor Measurement Characterization"
  http://www.ma2.life/doc/plan/Manhattan_2_Blueprint.pdf#pagemode=bookmarks
- For examples of devices that measure stimulus, search this document for "Mikroe has 500 small pcb". There are adaptors that connect Arduino to the Mikroe Click boards. They have C source code that attaches to boards. To open their mpkg files to see their C source code, one might need to first buy their $250 compiler (?).
  https://download.mikroe.com/documents/brochure/click-boards-brochure-2019-web-2.pdf

### 4.9.5 SUGGESTED PROJECT: Develop Website that allows Collaborative Development of Device Object Models

Create a website that allows programmers to collaboratively define standard object models, as described in "Chapter 28 Create Universal Language…" of the following document. If one downloads this PDF to a computer (File, Save As) and opens with an Acrobat viewer, an easy to use left-side Navigation system exposes the table of contents.

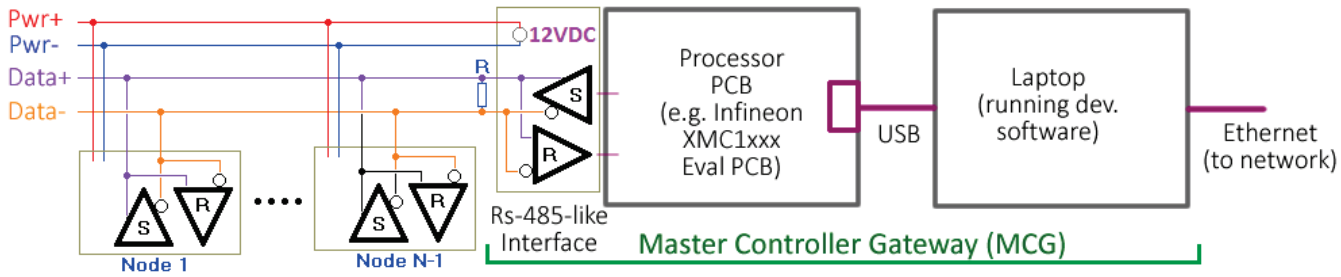
> *Manhattan 2 Blueprint Book (180 pages)*
> http://www.ma2.life/doc/plan/Manhattan_2_Blueprint.pdf#pagemode=bookmarks

- Researchers build on top of existing partially complete C# software written by G. Weinreb, currently referred to as "DIF" (Data Interoperability Foundation).
- DIF website enables web user to create classes and create parameters within each class. For each parameter, web user specifies name, type (e.g. flt32) and immutability (fixed value or not). Website generates server code and client class source code in multiple languages (e.g. C, C++, C#, Python). Website does not define methods for each class. Instead, those are coded by programmers who write methods for child classes that inherit DIF website defined class. DIF website includes source code that packs immutable values from multiple objects into one binary block (DIBB, one block per device); and code that unpacks the blocks and instantiates client objects elsewhere on the network that access that data. Subsequently, device programmers and client programmers have an easy and fast method of talking across the network via multiple programming languages.
- All materials are free and open, to encourage further development and adoption by industry worldwide.

## 4.10 Development Platform

### 4.10.1 SUGGESTED PROJECT: Build Universal Controller and Device Development Platform

Researchers create development platform to support Master Controller and Device research.



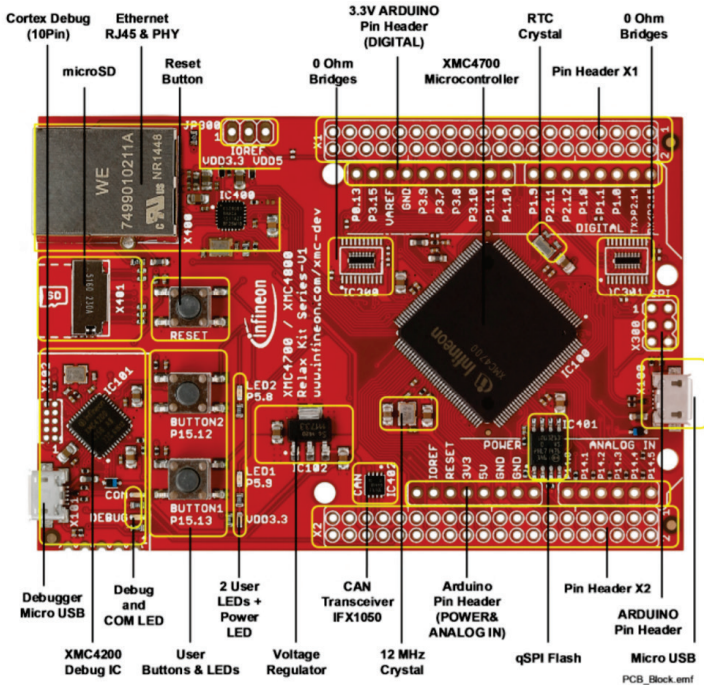

We support multiple physical and data layer schemes:

1. RS-485/MSTP: RS-485 physical layer with token ring data link layer (e.g. traditional BACnet MSTP), RS-485 transceiver IC connected to microprocessor UART interface.
2. CANbus 120Ω: CANbus physical and data link layer, CANbus transceiver IC connected to microprocessor CANbus controller, 120Ω terminators on both ends of daisy-chain cable.
3. RS-486P/MSTP: RS-486P physical layer with token ring data link layer (e.g. BACnet MSTP), connected to microprocessor UART interface. Search this document for "486P" for details.
4. RS-486P-wireAND/CANbus: RS-486P-wireAND physical layer connected to microprocessor CANbus interface, CANbus data link layer, relies on microprocessor CANbus controller.
5. DALI 2: 16V/250mA signaling. Search "DALI 2" and "DALI Slave Reference Designs" for details.
6. DALI 3P/CANbus: DALI 3P connected to CANbus controller. Search "DALI 3P" for details.

The MASTER CONTROLLER is a laptop that interfaces via USB to an Arduino Uno microprocessor board (e.g. KIT_XMC47_RELAX_V1 reference design) where the microprocessor has a built-in CANbus controller (to connect to devices) and Ethernet controller (to connect to network).

We work with multiple Arduino Uno shields (unless we can think of a better approach) where we typically have a stack of 3 or 4 shields, several categories of which are listed below.

1. Arduino Uno Microprocessor Board




- Initially we work with powerful processors to help programmers move quickly. Cost reduction is another step that is done later.
- Devices are developed with a processor board that has CANbus support, floating point hardware, plenty of RAM (>200KB), plenty of flash (>1MB), dimmer/motor controller support (analog comparator and timers), A/D channels, and D/A channels (e.g. to set current/voltage limits).
- Master Controllers are developed with a similar processor (to make it easy for programmers), yet must include Ethernet support (to connect to LAN via IP).
- Developers of different devices use the same processor board, even if only a small portion is used, to make development simpler. Cost reduction and consolidation is another step saved for later.
- The Infineon #XMC4700-F144 (2MB flash, 384KB RAM, Ethernet Support) is a possibility for device and master controller development, yet we are open to other options. The #KIT_XMC47_RELAX_V1 development kit is reasonably priced at $34, and satisfies the above requirements. The Xmc4700 has a simple Ethernet interface; however, for more sophisticated Ethernet, one can see the Xmc4800.
  - Reference Board: https://www.infineon.com/dgdl/Infineon-Board_User_Manual_XMC4700_XMC4800_Relax_Kit_Series-UM-v01_02-EN.pdf?fileId=5546d46250cc1fdf01513f8e052d07fc
  - Xmc4700 Datasheet: infineon.com/dgdl/Infineon-XMC4700-XMC4800-DS-v01_01-EN.pdf?fileId=5546d462518ffd850151908ea8db00b3
  - Xmc4700 Product: https://www.infineon.com/cms/en/product/microcontroller/32-bit-industrial-microcontroller-based-on-arm-cortex-m/32-bit-xmc4000-industrial-microcontroller-arm-cortex-m4/xmc4700/
  - Xmc4700 Ref Manual (3000pgs): https://www.infineon.com/dgdl/Infineon-ReferenceManual_XMC4700_XMC4800-UM-v01_03-EN.pdf?fileId=5546d462518ffd850151904eb90c0044
- (M4 core, CANbus, $2.17, 64K/20K flash/ram, 80MHz, floating point hardware, HRPWM, A/D, D/A, analog comparator) or the #STM32L431CBY6 (Cortex M4, CANbus, $2.13, *low power*, 128K/64K flash/ram, 80MHz, floating point hardware, A/D, D/A, analog comparator)
  - Xmc4108 Product: https://www.infineon.com/cms/en/product/microcontroller/32-bit-industrial-microcontroller-based-on-arm-cortex-m/32-bit-xmc4000-industrial-microcontroller-arm-cortex-m4/xmc4108-f64k64-ba/
  - Stm32L431 Product: https://www.st.com/en/microcontrollers-microprocessors/stm32l4x1.html
- In summary, we use a different processor for development (Arduino stack) and for consolidated design (one PCB with only what is needed). Both processors are powerful and similar. The primary difference is the development processor has more memory (2MB/384KB vs 128K/64K flash/ram).

2. Arduino Uno Communications Shields

- Researcher utilize existing products that communicate with devices via 2 wires:
  - RS-485 transceiver, connected to microprocessor UART
  - DALI 2 slave device, receive 16V/250mA, opto-couplers to processor UART pins, Search "DALI 2" and "DALI Click board" for details.

- Researchers develop several shield PCB's to support communication:
  - RS-486P transceiver[1]
  - Same as above, yet DIO bit & FPGA selects UART Tx/Rcv or CANbus controller[1].
  - DALI 3P slave device (receive 16V/250mA) [1]. For details search "DALI 3P".
  - DALI 2 and 3P Master Controller (supply 16V/250mA, possibly with voltage and current control via additional $1 tiny microprocessor).
  - DALI 3P Extension Cable Driver. For details; search this document for "Cable Driver".
  - DALI 3P Extension Cable Repeater. Search "Design Low Cost Repeater" for details.

[1]Microprocessor DIO bit connects communications wires to either microprocessor UART pins or to microprocessor CANbus H/L interface. This allows researchers to explore using CANbus data link layer, managed by microprocessor CANbus controller, or traditional UART. Tiny FPGA (e.g. $1 #ICE40UL640) is placed onto prototype PCB, to be used as needed by researchers.

3. Power Supply Arduino Uno Shields

- Researcher utilize existing products for power:
  - +5V (or 3.3V) wall mount power supply supports entire system (existing product)
- Researchers develop power supplies for tiny processors (search this document for "Power Supply Research"), and possibly build prototypes following the Arduino Uno form factor. Below are several small power supplies that are being studied. These are intended to be used in a consolidated design (one PCB w/ only what is needed) and might not have enough power to support a development stack.
  - Efficient Buck Converter Output: 5V/16mA, Input: 7VDC to 56VDC
  - LDO, linear, burns power Output: 5V/16mA, Input: 7VDC to 26VDC
  - AC Buck Converter Output: 5V/16mA, Input: 85VAC to 265VAC

4. Specialty I/O Arduino Uno Shields

*Researchers utilize existing products and also develop their own PCB's to support different specialty functions, several of which are listed below:*

- Arduino Uno Adaptor for Mikroe Click boards, existing product, Ref: https://www.mikroe.com/arduino-uno-click-shield
- Shield2Go PCB's via Shield2Go Adaptor, existing product, search "Shield2Go Adaptor".
- Nano PCB's via Nano Adaptor, existing product, search "Nano Adaptor" for details.
- There are many existing products that we can utilize, for ideas, search this file for "Arduino References" and "Arduino Uno".
- Low cost LED Driver (e.g. support on/off/dimming with 10W LED-only bulb). Reduce cost via something like a $1.20 Xmc1404, $2.18 Xmx4108, or $2.13 Stm32L431CBY6 processor which does mA current regulation and PWM w/o utilizing an external LED driver IC. This requires running specialty wires from microprocessor PCB to LED Driver PCB (e.g. PWM, current sense, voltage sense, 110VAC routes to LED PCB and power supply PCB). Search "LED Driver", "LED Driver with Xmc", "LED On/Off/Dimmer" and "BCCU" for details.
- Similar to above, yet stepper motor control (e.g. for damper in duct, or window thermal cover). Use existing Click board? Work directly with processor Timer and Current/Voltage regulation features. Search this file for "Motor Control".
- 110VAC on/off control via optically isolated TRIAC. Use existing Mikroe Click board?
- Sensors Development. Possibly prototype via Click form factor? For example:

- ±0.1°C accuracy temperature measurement.
  E.g. low cost $0.15 #NTCG103JX103 thermistor and resistor.
  Ref: http://gwinst.com/p/i4xx_manual_gen/sch_pdf/i510_schematic.pdf

Modifications to the traditional Arduino system include:

- Researchers consider bussing the following signals through Arduino Uno stack via an additional pcb-to-pcb connector (this is a first draft):
  - CANbus H/L pins from microprocessor
  - Flexible Power (e.g. 16 to 54VDC *or* 85 to 265VAC).
    Routes to power supply PCB, LED driver PCB, motor driver PCB, etc.
  - Microprocessor Current sense and Voltage sense when working with power control.
    Search this document for "LED Driver with Xmc" and "BCCU" for details.
  - 3 Dio and 2 Ain dedicated to Specialty I/O Shields (?)
  - 3 Dio and 2 Ain dedicated to Power Supply Shields (?)
  - 3 Dio and 2 Ain dedicated to Communications Shields (?)
- We might need to add additional connectors, test points, and jumpers when working with existing products. This is acceptable, especially when debugging the entire system.
- All software is to be written in a language that is popular in industry, such as C and C++ (compiler produces very fast optimized code and strips out unused code).
- All materials are free and open, to encourage further development and adoption by industry worldwide.

## 4.11 Tiny Power Supply Research

Tiny power supply research involves designing small low cost power supplies for tiny processors in tiny devices (e.g. 3.3V or 5V, 1mA sleep, 4 to 16mA runtime power).

One can look at building prototypes in an Arduino Uno form factor. However, a typical Arduino stack of shields contain circuitry overhead and might draw 2 to 10 times more power than a cost-reduced and power-reduced design. Tiny power supplies are often tested as an independent unit, and then incorporated into a consolidated design later.

For more information, search this document for "Power Supply Research". For details on the following power supply example designs, search "Power Supply" in file GWeinreb_Manhattan2_ResearchNotes.xlsx / "Wallbus".

- Efficient Buck Converter — Output: 5V/16mA — Input: 7VDC to 56VDC
- LDO, linear, burns power — Output: 5V/16mA — Input: 7VDC to 26VDC
- AC Buck Converter — Output: 5V/16mA — Input: 85VAC to 265VAC
- Flex Power — Output: 3.3V/16mA — Input: 85 to 265VAC *or* 16..54VDC

## 4.12 Device Hardware Research

Researchers develop specialty devices that implement monitoring and control. In some cases, these are existing Arduino Shield or Mikroe Click boards that are assembled with other Arduino boards, and researchers focus on developing software. In other cases, researchers design electrical circuits and build prototype PCB's.

All software is to be written in a language that is popular in industry, such as C and C++ (compiler produces very fast optimized code and strips out unused code). All materials are free and open, to encourage further development and adoption by industry worldwide.

Below is a list of typical electrical circuits found inside Devices:

- On/Off control of 110/220VAC (e.g. 5A) via TRIAC (e.g. for traditional AC appliance, AC motor, fan). TRIAC attaches to heat sink, or external thermal mass (e.g. motor casing)
- On/Off control of <58VDC (e.g. 4A) via MOSFET (e.g. for DC motor, 48V power supply).
- On/off/*dimmer* control of 110/220VAC (e.g. 5A) via chopping TRIAC (e.g. for traditional light bulb or heating element). TRIAC attaches to heat sink, or external thermal mass.
- Driver for LED-only bulb, powered by 15 to 48VDC, on/off/dimmer, drives LED's with chopped current source (e.g. for LED-only bulb controlled by network, output: 300mA at 15 to 24VDC).
- Same as above, yet powered by 110/220VAC (e.g. for LED-only bulb controlled by network).

- Uno Shield with multiple sensors, such as: temperature measurement accurate to ±0.5°C, temperature measurement accurate to ± 0.1°C, low cost light measurement, low cost humidity measurement, and low cost barometric pressure measurement.
    - One can try to implement these with a low cost technique that relies on a microprocessor's existing internal 12bit A/D; instead of a costly external IC with its own a/d.
    - 10K 0.5% thermistor and 10K 0.1% resistor components in voltage divider can provide ~±0.1°C accurate temperature measurement for $0.10 with a 12bit A/D that measures 0V, voltage divider central node, and voltage reference. Note that these parts are mounted on a PCB which is often a little warmer than ambient. Routing (cutting) a slot between sensor and main board might help provide a little more thermal isolation and accuracy, at some additional PCB manufacturing cost. Placing thermistor on cool corner of PCB might help a little.
    - Many microprocessors have an internal temperature sensor accurate to ~±3.0°C, which is helpful in some cases such as fire detection and internal diagnostics.
- Measure AC current, 110/220VAC, 0 to 20Amps. For example IC's, click here.
- User interface for wall on/off/dimmer control connected to 2wire or 4wire network (does not switch 110/220VAC). Assume dimmer is POT. For prototyping purposes, one can salvage POT and switch from existing product.
- Stepper Motor Controller (e.g. for window thermal cover, for duct damper, for liquid valve). Search "Stepper Motor" in this file for details.
- Variable Speed AC Motor Controller, 110/220VAC 50/60Hz input, output same voltage yet different frequency (e.g. for variable speed fan or compressor AC motor).
- Variable Speed DC Motor Controller, ≤48VDC input (e.g. for variable speed fan or compressor DC motor).
- One can look at implementing any of the above power related circuits with Flex Power that changes at runtime and is one of 16 to 54VDC, or 85 to 265VAC. This supports receiving power from grid 110/220VAC, battery 48V, or solar 48V without going through additional inverter(s). For more information on flex, search this file for "Power Supply with 85...265VAC *or* 16...54VDC Flexible Input".

The above power control circuits are implemented with microprocessors that include internal timers and analog comparators, instead of costly external IC's (e.g. we get communication and power control with $2 Infineon #Xmx4108). For details, search this file for "comparator" and "power conversion".

The above circuits satisfy many of the energy related measurement and control needs within a building.

Prototypes initially are clipped together PCB's (e.g. Arduino shields) and are not packaged for deployment. They are "workbench" devices that do not leave the laboratory. Packaging, consolidation, design-for-manufacture, and deployment is a project in an of itself and is implemented after workbench units are debugged and frozen (i.e. stop working on software,

electrical schematic and component selection).

Each device provides a set of features (i.e. multiple software objects). These can be things like on/off control (e.g. Triac 110/220VAC on/off control), output variable control (e.g. 0 to 255 to set LED dimmer), and stimulus measurement (e.g. temperature, humidity).

Hardware supports multiple software protocols since a researcher can easily load any software onto a hardware platform.

### 4.12.1 SUGGESTED PROJECT: Develop $1.50 Power Supply With 85...265VAC *or* 16...54VDC Flexible Input and non-isolated regulated 3.3V/16mA output with ~75% efficiency

There has been much debate over what type of power should be routed in a building for purposes of wall dimmer switch input, LED light control, window motorized thermal cover, fan in duct, damper in duct and temperature measurement. One option is to not decide and instead support Flex Power, which *changes at run time,* and is one of 16 to 54VDC *or* 85 to 265VAC.

Initial work shows that one can implement flexible power with $1.50 total parts cost by placing an on/off transistor in front of a 12...54VDC input buck converter (e.g. #Mp2459), and have the transistor only pass power when the voltage is < 54Volts. This does not cost more than the VAC power supply, so in a sense, one could say it is free to those that work with VAC. Also, if one is working with DCV voltages, the inputs of a flex power supply can accidently be shorted to 265VAC and not incur damage. Subsequently, DCV designers can think of this as DCV with VAC overvoltage protection.

For a schematic of this initial concept which has been simulated yet not prototyped, click here. For design notes and calculations, search "DESIGN: Tiny Power Supply -- Output: 3.3V/16mA" within file GWeinreb_Manhattan2_ResearchNotes.xlsx / "Wallbus".

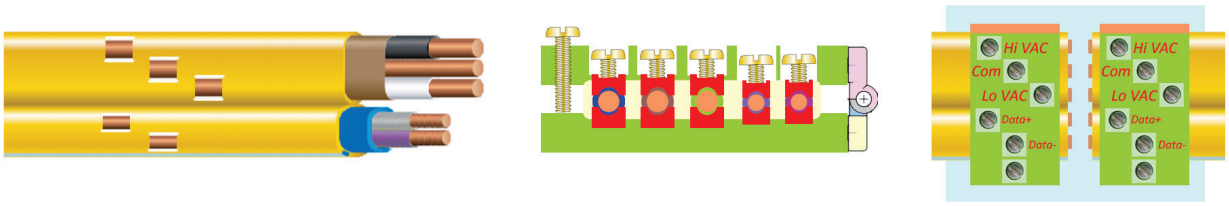
This initial design is only a starting point. Can one improve efficiency and/or reduce cost? The design notes in the .xlsx file contain a list of suggested next steps which include improving current design, additional simulation, spreadsheet modeling, breadboard prototyping, testing, PCB layout, building a prototype, testing and writing documentation.

For an example of an *isolated* power supply with 24V/30mA and 5V/30mA *unregulated* output, and 24 *to* 265 VAC *or* VDC input, click here. This design is not very efficient and involves the additional cost of a transformer; yet does show one way of implementing flexible input power. Can this approach be implemented for <= $1.50 and adapted to support lower input voltages (e.g. 9V or 16V) and *regulated* 3.3V output?

The parts for a 110/220VAC to 3.3V/16mA power supply cost approximately $1.50 (e.g. #VIPER011XS) and the parts for a 12...54VDC to 3.3V/16mA power supply cost approximately $0.75 (e.g. #Mp2459), for example.

## 4.13 Connector Hardware Research

Researchers develop a new connector system for devices in buildings that support 2 to 5 wires, tree topology wiring, electrical connections via screw force, low labor installation costs, and low device connector costs. Below is an example concept that involves a router-like hand tool which removes plastic insulation exposing conductors that are clamped with screw force. Subsequently, one can move from device to device without breaking the cable.




For details, please see

*Proposed New Standard for Smart LED Lighting & Wired IOT, Electronic Design News, Aug 2019*
https://www.edn.com/proposed-led-wired-iot-standard-can-reduce-energy-use-part-3/ (part 3)

One can expect connectors to appear in several sizes, several possibilities of which are listed below:

- Three Romex-like power wires and two data wires:
  - Type "3x18-2x20" three power wires 18awg (e.g. 6A), two data 20awg (e.g. 4A)
  - Type "3x16-2x20" three power wires 16awg (e.g. 10A), two data 20awg (e.g. 4A)
  - Type "3x14-2x20" three power wires 14awg (e.g. 15A), two data 20awg (e.g. 4A)
  - Type "3x12-2x20" three power wires 12awg (e.g. 20A), two data 20awg (e.g. 4A)
  - Type "3x10-2x20" three power wires 10awg (e.g. 30A), two data 20awg (e.g. 4A)
- Low voltage DC wiring:
  - Type "2x20-2x20" two power wires 20awg (e.g. 4A), two data 20awg (e.g. 4A)
  - Type "2x18-2x20" two power wires 18awg (e.g. 6A), two data 20awg (e.g. 4A)
  - Type "2x16-2x20" two power wires 16awg (e.g. 10A), two data 20awg (e.g. 4A)

Electrical engineering researchers create tables that suggest dimensions for each type. And these turn into suggested requirements for mechanical engineers.

The motorized installation hand Tool identifies size and acts accordingly. Hand tool includes a camera to help identify size and inspect routed area (plastic debris could affect connection). Tool determines wear on router bit and coach's user to replace as needed. Tool also turns screws to secure cover and clamp each wire; stopping at appropriate torque.

- All software is to be written in a language that is popular in industry and can easily access binary data, such as C++. This compiler produces very fast optimized code and strips out unused code (not python).
- All materials are free and open, to encourage further development and adoption by industry worldwide.

### 4.13.1 SUGGESTED PROJECT: Develop New Automated Power/Data Connector Standard

Mechanical engineers develop a connector standard that supports the above requirements. This includes mechanical drawings, simulations, calculations of electrical properties, cost models, and prototypes. Simulations look at high reliability over a 100 year lifespan. All materials are free and open to encourage adoption and collaboration.

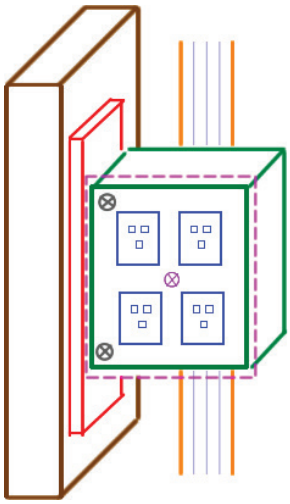
### 4.13.2 SUGGESTED PROJECT: Develop Hand tool to support new Connector

Mechanical engineers develop hand tool with router that removes material above *and* below wire, and rotates screws as needed. All materials, including software, are free and open to encourage adoption and collaboration.

### 4.13.3 SUGGESTED PROJECT: Develop Camera System to Support Hand tool

Researchers develop built-in low cost camera + light system that assists above tool. This identifies size, inspects routing, inspects router bit, inspects clamping, etc. All materials, including software, are free and open to encourage adoption and collaboration.

### 4.13.4 SUGGESTED PROJECT: Develop Alternative to Standard Electrical Box with Outlet



Researchers design alternatives to the traditional 110/220VAC power outlets mounted inside an electrical box.

One possible concept is illustrated to the right. The electrical outlets and an electrical box reside in one self-contained Module (green) that attaches to a wood stud (brown) via a metal bracket (red). Multi-wire cable (orange, 2 to 5wires) attaches to back of this module via hand Tool that prepares cable (described previously). The module (green) is one self-contained unit that is manufactured in a factory (outlets are *not* installed into a box in the field).

To replace Module (green) from within room, one disconnects it from bracket via 2 screws (gray), pulls module into room through drywall hole (dotted violet), and then unscrews cable from rear connector (not shown). Cable includes enough slack to accommodate this procedure. Not shown is faceplate that covers Module.

This approach requires less labor since electrician does not install electrical outlets into box and attach each wire individually.

If we route power and data (5 wires) to Module, then we can monitor current flow at each outlet, and provide feedback to end user. This might cost $5 in additional electronic components for each outlet. If a house has 30 outlets, for example, then one might be looking at $150 total for electronic components (1000qty), which is probably acceptable.

One can also look at on/off control of each outlet, which has its own batch of advantages, disadvantages, costs and issues (e.g. heat sink for 20Amp TRIAC).

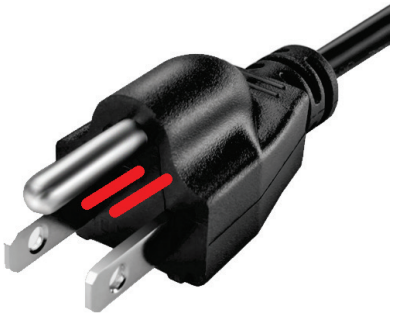
What else can one do with this? Temperature sensor? Fire sensor?

All materials, including software, are free and open to encourage adoption and collaboration.

### 4.13.5 SUGGESTED PROJECT: Devise Connector System that Routes Data to Appliance

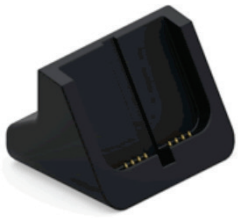
Researchers design a connector system that routes 1 to 3 data wires to an appliance, along with 2 to 3 power wires. This enables one to communicate with an appliance with 99.999% reliability (much more than wireless) and supports low power processor sleep while listening.

Wi-Fi burns energy while listening and in many cases completely fails for several reasons (e.g. spectrum completely utilized, metal shielding blocks transmission). However, it has one big advantage over a slow and reliable wired connection, which is faster data transfer rates (which support larger file sizes). One could utilize both systems. Wired could be used for: (1) building automation and control (e.g. refrigerator request 55°F ground source water pump be turned on) and (2) tell Wi-Fi transceiver to power on and listen for a short duration. And then Wi-Fi could be used for larger files such as such as video, pictures, and audio (e.g. Alexa microphone built into refrigerator door communicates with Amazon central computer).



Let's assume we are working with an 110VAC American 3-prong Power plug and we want to add two data wires. Also, let's assume we want for the "new" 5-pin plug to mate with the traditional old-style 3-prong socket; and we want for a traditional 3-prong plug to fit into a new 5-pin socket. What are our options? Below are several possibilities. Our data wires are something like RS-486P-wiredAnd or DALI 3P, discussed earlier. These are low voltage, low current.

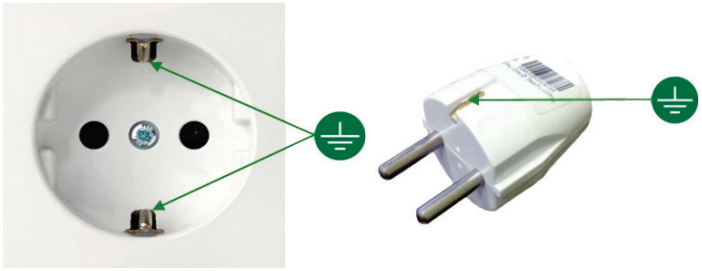
Shown above are 2 additional data pins in red color. In order for this to fit into the traditional 3-prong socket, these would need to be spring loaded (i.e. they retract when pressed against solid surface). These pins could insert into a socket, shown to the right, or press against a flat metal surface, shown to the left.



If working with a socket, then side forces within socket help make contact, yet this requires more forward pin spring force to get the pin into the socket. Also, if plug enters from an angle, it might jam fully or jam partially and require more spring force.

Alternatively, if retractable pins press against flat surface (instead of entering a socket), insertion is easier, yet connection is also less reliable.

Disadvantages of a spring mechanism are: (1) it could become damaged, (2) it might degrade over time and provide less force, and (3) spring force pushes plug outward which is not desired plug direction.



Another option involves side-features, an example of which is pictured to the right. The advantage of this approach is it provides a reliable connection. The disadvantage is that recessed socket makes it more difficult for user to grab plug while removing; and a 5-prong plug must have a large diameter if it is going to mate with side pieces, and also clear the traditional 3-prong plug.

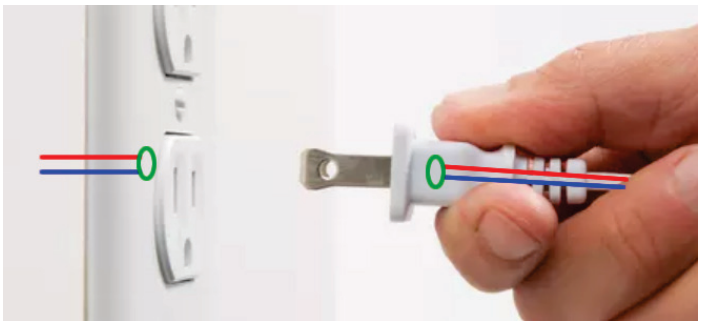
Our 2 data wires are routed to multiple devices in a tree network and both the socket and the corded device are part of that network. In other words, the socket itself contains embedded electronics that are considered one device on the network. Subsequently, in order for the socket to identify its companion device, it might measure current flow during corded device data transmission, or utilize some other method to identify companion device.

Researchers develop multiple concepts, simulate, produce cost models, and identify advantages and disadvantages. How one produces a reliable and low cost connection is not obvious.

All materials, including simulation and analysis, are free and open to encourage utilization, adoption and collaboration.

### 4.13.6 SUGGESTED PROJECT: Develop Near-Field Communication between Socket and Corded Device

Researchers explore ways of having a powered appliance communicate with a power socket without a mechanical data connection (e.g. do not add two pins to plug as discussed previously). Connection must be ≥ 99.999% reliable and support low-power processor sleep while listening.

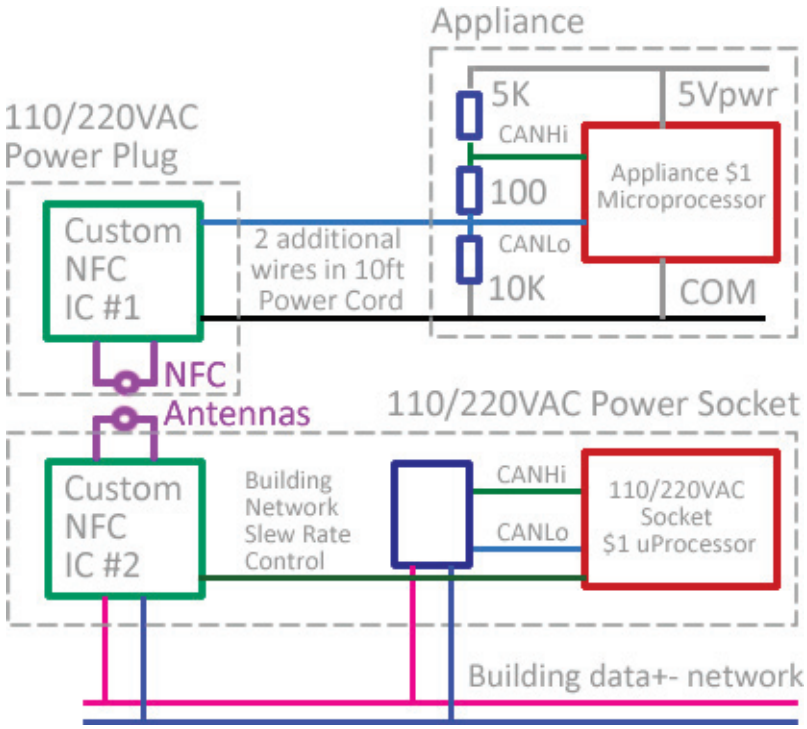


One possible implementation is sockets and plugs each include a sub-surface antenna that communicates via near-field communication (NFC), as illustrated to the right. This would be mechanically elegant, from the end user's point of view, since the communication system would not be visible.

If one wants to communicate at 100K bps, for example, they might utilize a 10MHz sinewave to communicate a logic 0 (dominant), and no sinewave to communicate a logic 1 (recessive, idle state). This would entail 100 cycles of 10MHz (10µSec / 10M = 100) and the receiver might use a notch filter and amplifier to look for the transmission. Preferably, the circuit does not use digital-signal-

processing (DSP) while listening, since that burns power. Researchers would look at different frequencies, each of which has their advantages and disadvantages.

Researchers develop a transceiver IC that connects directly to the popular CANbus communication system embedded into many microprocessors (i.e. this IC ultimately attaches to CAN_Hi/Lo wires). It is assumed that a CANbus capable low cost microprocessor exists in the corded device.




Researchers explore placing this custom IC several mm from the antenna (or have antenna built into IC) and embedding it into the plug and socket plastic molding. Subsequently, one could run long digital wires between transceiver IC and microprocessor. The advantage of this technique is the signal received at the antenna is low voltage and is susceptible to interference from power wires; whereas higher voltage digital signaling is more resilient to interference.

One would prefer to run as few wires as possible between plug and corded device. Perhaps two wires would be sufficient?

Researchers study various digital wiring options including twisting and foil shielding. More shielding involves more cost, yet also helps to reduce RFI emissions (outgoing) and susceptibility (incoming).

Alternatively, one could look at coax shielding of received antenna signal, and place custom IC with amplifier closer to microprocessor and away from plug. Or have two custom IC's. One with antenna and amplifiers, and another that is next to microprocessor.

One advantage of NFC over the previously discussed 2 additional data prongs is metal data prongs are susceptible to attack (e.g. child jams butter knife in-between them) which could potentially halt communication among all devices on the data network. In the NFC case, it would be difficult for building occupant to disrupt other devices (e.g. 10 LED lights, 4 window covers, and 3 light switches -- all connected to the same two data wires).

If one places 5KΩ/100Ω/10KΩ resistors in series between appliance 5Vpower and microprocessor COM (0V), then they would see 3.333V and 3.327V across the 100Ω with respect to COM. If one connects the 3.333V node to microprocessor CAN_Hi and the 3.327V node to CAN_Lo, then microprocessor would see something similar to a CANBus network. Then, one could route the

3.327V CAN_Lo and Microprocessor COM down the power cord to the custom IC, which would see 3.327V while idling (logic 1) and ~1V while microprocessor is momentarily signaling a logic 0. Also, the custom IC would momentarily short the 3.327V wire to ~1V to signal a logic 0 coming from the network. The custom IC might use a diode and capacitor to maintain power during logic 0 signaling (e.g. 1µF 10V 0603 $0.01).

If a 5KΩ termination resistor network drives the cable with active electronics off and a 20pF/ft power cord is 10ft long (200pF total), then rise time would be ~1uSec (5KΩ*200pF). If flight time down cord is 20nSec (2nSec/ft * 10ft, time for electricity to travel 10ft), and you want rise time to be 20-times slower to avoid ringing, then you would need it to be ≥0.4µSec (20*20), which is feasible. In other words, one would want digital signaling to be slew-rate limited to ≥0.4uSec in order to avoid ringing.

If an appliance has an existing $1 to $2 microprocessor with a $1.50 power supply (e.g. 110VAC-to-5VDC), then total electronics parts cost might be at $4. If one adds another $1 for communication to building network, this would increase total to $5, which is reasonable.

If socket contains microprocessor, it might also monitor power current flow at each outlet, to help building occupant focus more on large consumers of electricity.

Researchers explore ways of implementing 110/220VAC current measurement. The $2 Allegro #ACS726LLFTR-40B supports 0 to 40A measurement accurate to 1%, for example (see also: Allegro Micro Systems Current Measurement IC's and https://www.digikey.com/short/pd1rzb). If house has 30 outlets and cost to measure power is $10*30 = $300, for example, then owners might consider this reasonable. One obvious cost reduction method is to measure total power going to bank of 2 to 4 outlets, instead of power going to each.

If building network involves slew rate control over data signaling to avoid ringing with tree topology wiring (e.g. 45µSec w/ 3.3Kbps ≤300meters, 4.5µSec w/ 33K bps ≤30meters), then one might need a microprocessor at socket to control custom interface IC slew rate (i.e. for when this IC drives building data+- wires).

One would need to examine the above concepts more carefully to make sure they are feasible.

Researchers simulate, prototype and propose a custom IC(s), yet do not fabricate silicon (that is another step).

Researchers develop multiple concepts, simulate, produce cost models, and identify advantages and disadvantages.

All materials, including simulation and analysis, are free and open to encourage utilization, adoption and collaboration.

Suggested Requirements

- Transceiver electronics, wiring to plug, wiring to socket, and antennas must be low cost in order to encourage adoption.
- Must be ≥ 99.999% reliable (i.e. much more than wireless).
- Supports low power processor sleep while listening (no DSP that pulls signal out of noise).
- Ultimately connects directly to CANbus controller (idle at logic 1 and initiator transmits logic 0 to indicate a data transmission) that is built into ~$1 microprocessors.
- Assume data network physical layer is something like DALI 3P or RS-486P-wireAnd; and assume data link layer is CANbus.
- Researchers develop one custom IC to connect appliance microprocessor CANbus to antenna in plug; and also develops one custom IC to connect antenna in socket to data network (i.e. to DALI 3P or RS-486P-wireAnd).
- Sockets (not plug) may or may not include a microprocessor that is attached to the network. If microprocessor is present, it needs to identify its companion appliance by some method (e.g. NFC and antenna).

Challenges for engineers:

- RFI from power wires might disrupt communication (e.g. 20A 220VAC load suddenly turns on)
- Transceiver electronics for both corded device and socket might be elaborate & costly.

See Also

- Near-field Communication: https://en.wikipedia.org/wiki/Near-field_communication
- RFID Antennas: https://www.digikey.com/short/pd13pt
- RFID / NFC Transceivers: https://www.digikey.com/short/pd13fh
- RFID Development kits: https://www.digikey.com/short/pd1mnw
- CANbus: https://en.wikipedia.org/wiki/CAN_bus

# 5 REFERENCE: Solar Arrays

## 5.1 Microinverters

### 5.1.1 Microinverter Overview

- Microinverters, Wikipedia
  https://en.wikipedia.org/wiki/Solar_micro-inverter
- Digikey, list of micro inverters
  https://www.digikey.com/products/en/line-protection-distribution-backups/dc-to-ac-power-inverters/833?k=inverter

### 5.1.2 Microinverter Reference Designs

- TI Reference Designs
  http://www.ti.com/solution/solar-micro-inverter-diagram
- TI 500W Reference Design
  http://www.ti.com/lit/ug/tidub21c/tidub21c.pdf
- 250W Microinverter, ST Microsystems, Reference Design, AN4070
  https://www.st.com/content/ccc/resource/technical/document/application_note/fa/f1/fe/3d/81/1e/47/45/DM00050692.pdf/files/DM00050692.pdf/jcr:content/translations/en.DM00050692.pdf
- Microchip Reference Design (2010)
  http://ww1.microchip.com/downloads/en/appnotes/01338d.pdf

### 5.1.3 Microinverter Products

- 250W, 1 inch thick, Grid Tie inverter, $130 USA list price
  https://www.amazon.com/dp/B01KMXB0PS/ref=cm_sw_em_r_mt_dp_U_ZnmzDbQAAW4TJ
  https://www.gogreensolar.com/products/enphase-m215-60-2ll-s22-micro-inverter-mc4

### 5.1.4 Microinverter Books

- "The Art of Electronics", by Horowitz
  https://www.amazon.com/Art-Electronics-Paul-Horowitz/dp/0521809266
- "Solar Energy: The Physics and Engineering of Photovoltaic Conversion, Technologies and Systems"
  https://www.amazon.com/gp/product/1906860327/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1

# 6 REFERENCE: Microprocessor Companies

## 6.1 Microprocessor Summary

### 6.1.1 Cortex M0 ... M4 Processors

- Infineon Company
  - Xmc1xxx Cortex M0. e.g. Xmc1404 (CANbus, $1.20, drive motors/LEDs, DC/DC, 2mA sleep)
  - Xmc4xxx Cortex M4
- ST Microsystems Company
  - STM32L4 Cortex M4, low power, $2
    https://www.st.com/content/ccc/resource/sales_and_marketing/presentation/product_presentation/4a/c2/01/20/3d/e6/4c/08/stm32l4_marketing-pres.pdf/files/stm32l4_marketing-pres.pdf/jcr:content/translations/en.stm32l4_marketing-pres.pdf

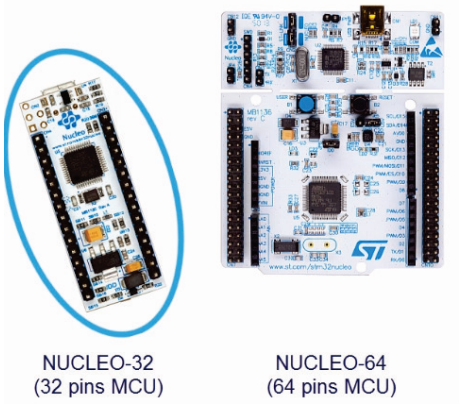
## 6.2 ST Microsystems Company

### 6.2.1 STM32 Processors

- List of 3000 different STM32 Coretex M0/M3/M4/M7 Processors from ST Microsystems
  https://www.digikey.com/products/en/integrated-circuits-ics/embedded-microcontrollers/685?k=stm32
- List of STM32 IC's that support CANbus, both low power (L) and not low power (F)
  https://www.digikey.com/products/en/integrated-circuits-ics/embedded-microcontrollers/685?FV=1f140000%2Cffe002ad%2C116403f4%2C116403f5%2C116403f6%2C116403f7%2C11640402%2C11640411%2C11640434%2C11640435%2C11640021%2C1164014c%2C116401bb%2C11640265%2C11640274%2C11640275%2C11640290%2C11640291%2C116402ee%2C116402ef%2C11640305%2C11640334%2C11640343%2C11640344%2C11640345%2C11640357%2C1164035f%2C1164039f%2C116403a0%2C116403af%2C116403bc%2C116403bd%2C116403c0%2C116403c6%2C116403c7%2C116403c8%2C116403cd%2C116403ce%2C116403cf%2C116403d0%2C116403d1%2C116403d6%2C116403da%2C116403db%2C116403dc&quantity=0&ColumnSort=1000011&page=1&k=stm32&pageSize=25&pkeyword=stm32
- List of STM32 IC's that support CANbus *and* are low power (L), lowest cost MCU is STM32L431 ($2)
  https://www.digikey.com/products/en/integrated-circuits-ics/embedded-microcontrollers/685?FV=116403f4%2C116403f5%2C116403f6%2C116403f7%2C11640402%2C11640411%2C11640434%2C11640435%2C11640021%2C1164014c%2C116401bb%2C11640265%2C11640274%2C11640275%2C11640290%2C11640291%2C116402ee%2C116402ef%2C11640305%2C11640334%2C11640343%2C11640344%2C11640345%2C11640357%2C1164035f%2C1164039f%2C116403a0%2C116403af%2C116403bc%2C116403bd%2C116403c0%2C116403c6%2C116403c7%2C116403c8%2C116403cd%2C116403ce%2C116403cf%2C116403d0%2C116403d1%2C116403d6%2C116403da%2C116403db%2C116403dc%2C1f140000%2Cffe002ad%2Cffecc011%2Cffecf697%2Cffecf698&quantity=0&ColumnSort=1000011&page=1&k=stm32&pageSize=25&pkeyword=stm32
- STM32L ... low power processor guide
  https://www.st.com/en/microcontrollers-microprocessors/stm32-ultra-low-power-mcus.html
- STM32L0 ... Low Power ($1, no CANbus, Cortex M0+)
  https://www.st.com/en/microcontrollers-microprocessors/stm32l0-series.html
- STM32L1 -- similar to L0 yet more flash/ram
- #STM32L431, $2.13, low power processor, CANbus, 128K/64K flash/ram, cortex M4, 80MHz
  - For details on low power processor consumption, search datasheet for "Supply Current":
    https://www.st.com/resource/en/datasheet/stm32l431kb.pdf
- #STM32F042 is $.98 processor (cortex M0+) that supports CANbus yet it is not low power
- #STM32L031F6P6 -- Example Cortex M0+ processor, $0.98, no CANbus, 32k/8k flash/ram, Tssop20, A/D, Analog Comparators, Timers
  https://www.digikey.com/product-detail/en/stmicroelectronics/STM32L031F6P6TR/497-19367-2-ND/8257862

### 6.2.2 STM32 Processor & Expansion Boards (e.g. Arduino Nano = "Nucleo 32", Arduino Uno R3 = "Nucleo 64")

- Brochure that summarizes processor boards and shields (good)
  https://www.st.com/content/ccc/resource/sales_and_marketing/promotional_material/brochure/8d/9e/d5/a3/84/16/46/df/brstm32ode.pdf/files/brstm32ode.pdf/jcr:content/translations/en.brstm32ode.pdf
- Video that summarizes processor boards and shields (good)
  https://www.st.com/en/evaluation-tools/nucleo-l432kc.html
- "NUCLEO-32 MCU" = processor, Arduino Nano (32pin dip)
- "NUCLEO-64 MCU" = processor, Arduino Uno connector
- Overview of STM32 Nucleo Expansion Boards based on Arduino Uno R3 Connector
  https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards.html#overview
- List of 100's of STM32 Nucleo Expansion Boards
  https://www.st.com/en/ecosystems/stm32-nucleo-expansion-boards.html#products




### 6.2.3 STM32 Arduino Uno Boards (64pin, June 2018, "Nucleo-64")

- Datasheet that describes STM32 Arduino Uno Processor Boards
  https://www.st.com/resource/en/data_brief/nucleo-l053r8.pdf
- Brochure that describes STM32 Arduino Uno Processor Boards
  https://www.st.com/content/ccc/resource/sales_and_marketing/promotional_material/brochure/8d/9e/d5/a3/84/16/46/df/brstm32ode.pdf/files/brstm32ode.pdf/jcr:content/translations/en.brstm32ode.pdf

| What you want to do | What we provide | Components | Board reference |
|---|---|---|---|
| **Process*** | Ultra-low power | STM32L0 - ARM® Cortex®-M0+ ultra-low power 32-bit MCU | NUCLEO-L053R8 |
| | High performance | STM32F4 - ARM® Cortex®-M4 high-performance 32-bit MCU | NUCLEO-F401RE |
| | Rich peripheral set | STM32L4 - ARM® Cortex®-M4 ultra-low power, high-performance 100DMIPS 32-bit MCU with USB-OTG, rich peripheral set and security features | NUCLEO-L476RG |
| **Sense** motion, pressure, humidity, temperature, distance, light, sound | Motion & Environmental sensors | LSM6DSL 3-axis accelerometer + 3-axis, LSM303AGR 3-axis magnetometer + 3-axis accelerometer, HTS221 humidity and temperature, LPS22HB pressure | X-NUCLEO-IKS01A2 |
| | Proximity sensors | VL6180X FlightSense™ proximity, gesture and ambient light sensor | X-NUCLEO-6180XA1 |
| | | VL53L0X FlightSense™ ranging and gesture sensor | X-NUCLEO-53L0A1 |
| | | VL53L1X FlightSense™ ranging and gesture sensor | X-NUCLEO-53L1A1 |
| | Microphone | MP34DT01-M digital microphone | X-NUCLEO-CCA02M1 |
| **Connect** | Bluetooth Low Energy 4.1 | BlueNRG-MS based Bluetooth Low Energy (V4.1) Module | X-NUCLEO-IDB05A1 |
| | Sub-GHz radio | SPIRIT1 RF SPSGRF-868 module | X-NUCLEO-IDS01A4 |
| | | SPIRIT1 RF SPSGRF-915 module | X-NUCLEO-IDS01A5 |
| | | SPIRIT2 RF Sub-1 GHz 868 | X-NUCLEO-S2868A1 |
| | NFC | M24SR Dynamic NFC tag | X-NUCLEO-NFC01A1 |
| | | M24LR Dynamic NFC tag | X-NUCLEO-NFC02A1 |
| | | CR95HF NFC Reader | X-NUCLEO-NFC03A1 |
| | | ST25DV Dynamic NFC/RFID tag | X-NUCLEO-NFC04A1 |
| | | ST25R3911B NFC Reader | X-NUCLEO-NFC05A1 |
| | Modem | ST7580 Power line communication | X-NUCLEO-PLM01A1 |
| **Move/Actuate** | Motor driver | L6474 Stepper motor driver | X-NUCLEO-IHM01A1 |
| | | L6470 Two Axes motor driver | X-NUCLEO-IHM02A1 |
| | | PowerSTEP01 High power stepper motor driver | X-NUCLEO-IHM03A1 |
| | | L6206 Dual brush DC motor driver | X-NUCLEO-IHM04A1 |
| | | L6208 Bipolar Stepper motor driver | X-NUCLEO-IHM05A1 |
| | | STSPIN220 Low-voltage stepper motor driver | X-NUCLEO-IHM06A1 |
| | | L6230 3-phase Brushless DC motor driver | X-NUCLEO-IHM07M1 |
| | | L6470 F7 MOSFET Low-Voltage BLDC Motor Driver | X-NUCLEO-IHM08M1 |
| | | Motor control connector | X-NUCLEO-IHM09M1 |
| | | STSPIN230 Low-voltage BLDC 3-phase motor driver | X-NUCLEO-IHM11M1 |
| | | STSPIN240 Low-voltage dual-brush DC motor driver | X-NUCLEO-IHM12A1 |
| | | STSPIN250 Low-voltage DC motor driver | X-NUCLEO-IHM13A1 |
| | | STSPIN820 Stepper motor driver | X-NUCLEO-IHM14A1 |
| | | STSPIN840 Dual-brush DC motor driver | X-NUCLEO-IHM15A1 |
| | | STSPIN830 Three-phase brushless DC motor driver | X-NUCLEO-IHM16M1 |
| | | STSPIN233 Low-voltage 3-phase brushless DC motor driver | X-NUCLEO-IHM17M1 |
| **Power/Drive** | Battery and energy management | VPS2535H 24V Intelligent power switch | X-NUCLEO-IPS02A1 |
| | LED Lighting | LED6001 Single channel LED driver with integrated boost controller | X-NUCLEO-LED61A1 |
| | | 16-channel LED driver board | X-NUCLEO-LED16A1 |
| **Translate** signal conditioning | Audio processing | STA350BW High-efficiency digital audio system | X-NUCLEO-CCA01M1 |
| | Op Amp | Operational Amplifiers (TSZ124) | X-NUCLEO-IKA01A1 |
| | Industrial Input/Output | CLT01 Protected digital termination array and VNI8200XP smart power solid state relay | X-NUCLEO-PLC01A1 |
| | | ISO8200BQ Industrial digital output | X-NUCLEO-OUT01A1 |
| **Discovery and Form Factor Boards** | IoT Discovery Kit | STM32L4 Discovery kit IoT node, low-power wireless, BLE, NFC, SubGHz, Wi-Fi | B-L475E-IOT01A |
| | SensorTile | STM32L4 Form Factor module for motion, audio, environmental sensing and Bluetooth Low Energy | STEVAL-STLKT01V1 |
| | BlueCoin | STM32F4 Form Factor module for motion, audio, environmental sensing and Bluetooth Low Energy | STEVAL-BCNKT01V1 |
| | NFC Sensor TAG | STM32L0 Smart and flexible NFC Tracker evaluation board with sensors | STEVAL-SMARTAG1 |

### 6.2.4 STM32 Arduino Nano processor board (32pin DIP, "Nucleo-32")

- STM32 Nucleo-32 Nano Processor Board (32pin DIP, Arduino Nano)
  - We do not see processor on Nano board with w/ CANbus or LED Driver control
  - Product: https://www.st.com/en/evaluation-tools/nucleo-l432kc.html
  - Datasheet: https://www.st.com/content/ccc/resource/technical/document/user_manual/e3/0e/88/05/e8/74/43/a0/DM00231744.pdf/files/DM00231744.pdf/jcr:content/translations/en.DM00231744.pdf

### 6.2.5 STM32 Arduino Uno R3 Shields

- Current Measurement, Instrumentation Amplifier, Comparator, #x-nucleo-ika01a1
  https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-translate-hw/x-nucleo-ika01a1.html
- Stepper Motor Controller, #X-NUCLEO-IHM03A1
  https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-move-actuate-hw/x-nucleo-ihm03a1.html
- Stepper Motor Controller, #X-NUCLEO-IHM01A1
  https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-move-actuate-hw/x-nucleo-ihm01a1.html

- Stepper Motor Controller, #X-NUCLEO-IHM05A1
https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-move-actuate-hw/x-nucleo-ihm05a1.html
- Dual brush DC motor driver, #X-NUCLEO-IHM04A1
https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-move-actuate-hw/x-nucleo-ihm04a1.html
- PLC, Solid State Relay, Protected Digital Inputs, #X-NUCLEO-PLC01A1
https://www.st.com/en/ecosystems/x-nucleo-plc01a1.html
- Solid state Relays, #X-NUCLEO-OUT02A1
https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-translate-hw/x-nucleo-out02a1.html
- Three-phase brushless DC motor driver, #X-NUCLEO-IHM07M1
https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-move-actuate-hw/x-nucleo-ihm07m1.html
- Electrochemical Sensors
https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-sense-hw/p-nucleo-ika02a1.html
- Range Sensor
https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-sense-hw/x-nucleo-53l0a1.html
- LED Driver (on/off/dimmer, driven by #LED6001)
https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-power-drive-hw/x-nucleo-led61a1.html
- Power Line Communication, #X-NUCLEO-PLM01A1
https://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-connect-hw/x-nucleo-plm01a1.html

### 6.2.6 STM32 Application Notes

- List of 200's STM32 Application Notes
https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html#resource
- STM32L1 Analog Comparators, Application Note
https://www.st.com/content/ccc/resource/technical/document/application_note/f8/aa/e6/68/82/a4/40/81/CD00280599.pdf/files/CD00280599.pdf/jcr:content/translations/en.CD00280599.pdf
- AN4445, STM32L0xx ultra-low power (XLP) features overview
https://www.st.com/content/ccc/resource/technical/document/application_note/27/58/8e/81/79/fb/4f/ac/DM00108286.pdf/files/DM00108286.pdf/jcr:content/translations/en.DM00108286.pdf
- AN4776, Timer Cookbook
https://www.st.com/content/ccc/resource/technical/document/application_note/group0/91/01/84/3f/7c/67/41/3f/DM00236305/files/DM00236305.pdf/jcr:content/translations/en.DM00236305.pdf
- AN5058, Low Cost Power Supply From Mains (110/220VAC)
https://www.st.com/content/ccc/resource/technical/document/application_note/group0/c6/9e/d6/c8/39/93/41/00/DM00415532/files/DM00415532.pdf/jcr:content/translations/en.DM00415532.pdf
- AN2834, How to get the best ADC accuracy in STM32 microcontrollers
https://www.st.com/content/ccc/resource/technical/document/application_note/group0/3f/4c/a4/82/bd/63/4e/92/CD00211314/files/CD00211314.pdf/jcr:content/translations/en.CD00211314.pdf
- AN3070, Managing the Driver Enable signal for RS-485 and IO-Link communications with the STM32 USART
https://www.st.com/content/ccc/resource/technical/document/application_note/52/57/b8/e4/08/38/43/cd/CD00249778.pdf/files/CD00249778.pdf/jcr:content/translations/en.CD00249778.pdf
- AN2841, LED Driver implemented on STM32™ microcontroller (2008)
https://www.st.com/content/ccc/resource/technical/document/application_note/64/4d/e3/1d/b3/34/4f/0b/CD00213809.pdf/files/CD00213809.pdf/jcr:content/translations/en.CD00213809.pdf
- AN4232, Analog comparators for STM32F3 (STM32L4 has one too)
https://www.st.com/content/ccc/resource/technical/document/application_note/4f/92/37/3e/55/66/46/e2/DM00074240.pdf/files/DM00074240.pdf/jcr:content/translations/en.DM00074240.pdf

## 6.3 Infineon Company

### 6.3.1 Infineon Processors

- List of Infineon Xmc Processor Boards Compatible with Arduino Uno Formfactor with Arduino IDE (processors: Xmc1100, Xmc1202, Xmc4700):

- List: https://www.infineon.com/cms/en/tools/landing/infineon-for-makers/microcontroller-boards/

- Xmc4108Q48K64 (M4 core, CANbus, $2.17, 64K flash, 20K RAM, floating point hardware, HRPWM)
  - Xmc4000 comparators include filtering, blanking and clamping capabilities as well as a DAC for automatic reference or slope generation (i.e. motor control, LED control, dc/ac conversion)
  - User's Manual: https://www.infineon.com/dgdl/Infineon-xmc4100_xmc4200_rm_v1.6_2016-UM-v01_06-EN.pdf?fileId=db3a30433afc7e3e013b3c44ccd35c20
  - Product Webpage: https://www.infineon.com/cms/en/product/microcontroller/32-bit-industrial-microcontroller-based-on-arm-cortex-m/32-bit-xmc4000-industrial-microcontroller-arm-cortex-m4/xmc4108-f64k64-ba/
  - Data Sheet: https://www.infineon.com/dgdl/Infineon-XMC4100_XMC4200_DS-DS-v01_04-EN.pdf?fileId=5546d462696dbf120169817056f938ff

- Xmc1404Q048X0064 (M0+ core, CANbus, $1.50, 64K flash, 16K RAM)
  - Product Webpage: https://www.infineon.com/cms/en/product/microcontroller/32-bit-industrial-microcontroller-based-on-arm-cortex-m/32-bit-xmc1000-industrial-microcontroller-arm-cortex-m0/xmc1404-q064x0200-aa/
  - For details on low power processor operation, search datasheet for "Typical Active Current" and "Power Supply Current". Note that "test conditions" refer to $f_{MCLK}$ / $f_{PCLK}$ in units of MHz (e.g. 48/96 refers to 48MHz/96MHz).
    https://www.infineon.com/dgdl/Infineon-XMC1400-DataSheet-v01_04-EN.pdf?fileId=5546d46250cc1fdf015110a2596343b2

- Power and Sensing Guide (300 pages) explains how Xmc1xxx processors interface to gadgets
  https://www.infineon.com/dgdl/Infineon-Power_and_Sensing_Selection_Guide_2019-SG-v00_00-EN.pdf?fileId=5546d4625607bd13015621522aa012cb

- Infineon CANbus Controller (Controller Area Network Controller, MultiCAN)
  - App Note: https://www.infineon.com/dgdl/Infineon-MultiCAN-XMC4000-AP32300-AN-v01_00-EN.pdf?fileId=5546d4624e765da5014ed91d6be32110
  - Power Point: https://www.infineon.com/dgdl/Infineon-IP_MultiController_Area_Network_MultiCAN-TR-v01_00-EN.pdf?fileId=5546d46254e133b401554de8f66b5e19

### 6.3.2 Xmc4xxx Power Conversion Application Notes

- XMC™ in Power Conversion (see pages 8..11 for info on HRPWM High Resolution PWM)
  https://www.infineon.com/dgdl/Infineon-Application-Power-Conversion-XMC-in-Power-Conversion-Applications_2-TR-v01_00-EN.pdf?fileId=5546d462576f34750157dcc78cd22555

- Synchronous buck converter with XMC™ Digital Power Explorer Kit (Xmc4200), #AP32319
  https://www.infineon.com/dgdl/Infineon-AP32319_Synchronous_Buck_converter_with_XMC_Digital_Power_Explorer_Kit-AN-v01_00-EN.pdf?fileId=5546d462557e6e8901559b9168515eec

- Introduction to Digital Power Conversion
  https://www.infineon.com/dgdl/Infineon-XMC4000_XMC1000_%20Introduction_to_Digital_Power_Conversion-AN-v01_00-EN.pdf?fileId=5546d4624b0b249c014b497c0c33089e

- Peak current control in XMC4xxx
  https://www.infineon.com/dgdl/Infineon-XMC-peak-current-control-in-XMC4-TR-v01_00-EN.pdf?fileId=5546d46253f650570154610003fd7f70

- XMC in application - Power Management Bus (PMBus)
  https://www.infineon.com/dgdl/Infineon-Application-power-conversion-power-management-Bus-PMBus-TR-v01_00-EN.pdf?fileId=5546d4625607bd1301564f46c4845cdb

- CCU8: Capture and Compare Unit 8
  https://www.infineon.com/dgdl/Infineon-IP_CCU8_XMC-TR-v01_02-EN.pdf?fileId=5546d4624ad04ef9014b0780b3482262

- CCU4: Capture and Compare Unit 4
  https://www.infineon.com/dgdl/Infineon-IP_CCU4_XMC-TR-v01_02-EN.pdf?fileId=5546d4624ad04ef9014b0780bb082263

- HRPWM: High Resolution Pulse Width Modulation Timers
  https://www.infineon.com/dgdl/Infineon-IP_HRPWM_XMC4-TR-v01_02-EN.pdf?fileId=5546d4624ad04ef9014b0780db972267

- VADC: Versatile Analog to Digital Converter
  https://www.infineon.com/dgdl/Infineon-Peripheral%20-%20XMC4000%20Versatile%20Analog%20to%20Digital%20Converter%20(VADC)-TR-v01_00-EN.pdf?fileId=5546d462580663ef01582a3df0783e2f

- DAC: Digital to Analog Converter
  https://www.infineon.com/dgdl/Infineon-IP_DAC_XMC4-TR-v01_01-EN.pdf?fileId=5546d4624ad04ef9014b07777ae2225f

### 6.3.3 Infineon Arduino Products

- Infineon Arduino Products
  https://www.infineon.com/cms/en/tools/landing/infineon-for-makers/?redirId=57438#shields-for-arduino

### 6.3.4 List of Infineon Arduino Uno Shields (e.g. added via Uno-to-Shield2Go adaptor)

LIST: https://www.infineon.com/cms/en/tools/landing/infineon-for-makers/arduino-shields/

- Lite DCDC System Basis Chip Shield with TLE9471-3ES (CAN BUS support, #MCP2515 PHY)
  https://www.infineon.com/cms/en/product/evaluation-boards/sbc-shield_tle9471/
  https://www.microchip.com/wwwproducts/en/en010406#additional-features
- BLDC Shield with TLE9879QXA40
  https://www.infineon.com/cms/en/product/evaluation-boards/bldc_shield_tle9879/
- Dual H-Bridge Shield with IFX9202ED
  https://www.infineon.com/cms/en/product/evaluation-boards/ifx9202ed_dev_board/
- RGB LED Driver Shield with Xmc1202
  https://www.infineon.com/cms/en/product/evaluation-boards/kit_led_xmc1202_as_01/
- DC Motor Control Shield with BTN8982TA, 12VDC Power source
  https://www.infineon.com/cms/en/product/evaluation-boards/dc-motorcontr_btn8982/
- Stepper Motor Control Shield with IFX9201 & XMC1300
  https://www.infineon.com/cms/en/product/evaluation-boards/kit_xmc1300_ifx9201/?redirId=59404
- Low-Side Switch Shield with BTF3050TE
  https://www.infineon.com/cms/en/product/evaluation-boards/24v_shield_btt6030/
- 12V Protected Switch Shield with BTS50015-1TAD
  https://www.infineon.com/cms/en/product/evaluation-boards/shield_bts50015-1tad/

### 6.3.5 Infineon LED Driver

- Build LED Driver with Xmc1x00, Application Note, very good
  https://www.infineon.com/dgdl/Infineon-AP201609_Driving_LEDs_with_XMC1000-AN-v01_00-EN.pdf?fileId=5546d462580663ef01581a09fde446bd
- Brightness and Color Control Unit (BCCU), Application Note
  https://www.infineon.com/dgdl/Infineon-BCCU-XMC1000-AP32275-AN-v01_01-EN.pdf?fileId=5546d4624e765da5014ed8cabae512ad
- Infineon Xmc1302 DALI Slave Reference Design, drives LED (on/off/dimmer) with buck converter given 24 to 48VDC power, < 800mA to LED, very good
  https://www.infineon.com/cms/en/product/evaluation-boards/kit_xmc1_led_cc_exp_001/

## 6.3.6 Infineon LED Driver Product Summary

**Recommended LED driver products**

| Functional block | Product type | IC product family | MOSFET technology | Voltage class |
|---|---|---|---|---|
| **PFC stage** | PFC | IRS2505 | CoolMOS™ P7 | 600/700/800/950 V [1] |
| **Main stage** | PFC + LCC (constant current)<br>PFC + LLC (constant current) | ICL5102 [2] | CoolMOS™ P7 (up to 600 mΩ) | 600 V/650 V |
| | | | CoolMOS™ CE (above to 600 mΩ) | 600 V |
| | PFC + flyback (dual stage) | XDPL8220 [3] / XDPL8221 [2] | CoolMOS™ P7 | 800 V/950 V |
| | PFC/flyback (single-stage constant voltage) | XDPL8105 | CoolMOS™ P7 | 800 V/950 V |
| | PFC/flyback (single-stage constant voltage) | XDPL8218 | CoolMOS™ P7 | 800 V/950 V |
| **Buck / linear solutions** | Secondary buck (single-channel)<br>Secondary buck (multichannel) | ILD6150 / ILD8150 | Integrated | 60 V/80 V |
| | | XMC1300 / XMC1400 [1] | OptiMOS™ | 100 V/150 V/ 200 V/250 V/ 300 V |
| | Secondary linear | BCR601 | OptiMOS™ | 75 V/100 V |
| **Synchronous rectification** | Synchronous rectification controller | IR1161 / IR11688 | OptiMOS™ | 100 V/150 V/200 V |
| **Dimming** | 0-10 V dimming interface IC | CDM10V | - | - |
| | | CDM10VD | - | - |
| **Hardware based security** | OPTIGA™ | OPTIGA™ Trust | - | - |
| **MCU** | XMC™ microcontroller | XMC1100 | - | - |
| **Sensors** | XENSIV™ radar sensor IC | BGT24LTR11 | - | - |

**Linear/switch mode LED driver IC product portfolio**

| Functional block | Topology | IC product family | MOSFET technology | Voltage class |
|---|---|---|---|---|
| **Linear LED driver IC** | Linear | BCR400 series | Integrated (extra transistor for BCR450) | - |
| | | BCR602 | External N-channel MOSFET | 75 V / 100 V |
| **Switch mode LED driver IC** | Buck | ILD6000 series | Integrated | - |
| | | XMC1300/XMC1400* | OptiMOS™ | 100 V/150 V/200 V/250 V/ 300 V |
| | Buck/boost | ILD1151 | OptiMOS™ | 60 V/100 V |

## 6.3.7 Infineon Motor Control

- Infineon CCU8 module helps with motor control
  - https://www.infineon.com/dgdl/Infineon-CCU8-XMC1000_XMC4000-AP32288-AN-v01_01-EN.pdf?fileId=5546d4624e765da5014ed8dd5c7d1730
- Digital Power Explorer Kit
  - https://www.infineon.com/cms/en/product/evaluation-boards/kit_xmc_dp_exp_01/
  - https://www.infineon.com/dgdl/Infineon-XMC_DigitalPowerExplorer_GettingStarted-GS-v01_00-EN.pdf?fileId=5546d4625185e0e201518c0d45c13e50
- Stepper Motor Control with Xmc1302
  https://www.infineon.com/cms/en/product/evaluation-boards/kit_xmc1300_ifx9201/
- Motor Control with Xmc1302 or Xmc1402 (250W, 3-phase), does not support CANbus
  https://www.infineon.com/dgdl/Infineon-XMC_250W_Motor_Control_kit_customer_presentation-PP-v01_00-EN.pdf?fileId=5546d46269e1c019016ab69b87d5564a
- iMOTION™ Modular Application Design Kit (MADK) - 3Phase Motor Control
  - https://www.infineon.com/dgdl/Infineon-iMOTION_MADK_Evaluation_Platform-PB-v01_00-EN.pdf?fileId=5546d46265487f7b0165d356fa786cd6
  - https://www.infineon.com/dgdl/Infineon-An2018-01_EVAL-M1-101T_User_Manual-UM-v01_06-EN.pdf?fileId=5546d462625a528f01629491d86c39ba
  - https://www.infineon.com/cms/en/product/power/motor-control-ics/digital-motor-controller-imotion/imotion-modular-application-design-kit/?redirId=68319
  - https://www.infineon.com/dgdl/Infineon-AN2017-13_EVAL-M1-183M_User_Manual-UM-v01_02-EN.pdf?fileId=5546d4625fe367840160209336161 1da
  - https://www.infineon.com/dgdl/Infineon-AN2018-02_EVAL-M3-102T_User_Manual-UM-v01_05-EN.pdf?fileId=5546d462625a528f01629491cef239b7
- Allegro Company has many interface IC's as well, which can attach to Xmc1xxx processors
  https://www.allegromicro.com/-/media/Files/Allegro-Selection-Guide.ashx

### 6.3.8 Power Conversion (AC-to-DC, DC-to-DC, Drive LEDs)

- Xmc Processors in Power Conversion Applications
  https://www.infineon.com/dgdl/Infineon-APP_PowerConversion_XMC_in_Power_Conversion_Applications_XMC-TR-v01_02-EN.pdf?fileId=5546d4624cb7f111014ceb7af35d268e

- Xmc4000 comparators include filtering, blanking and clamping capabilities as well as a DAC for automatic reference or slope generation.

- Xmc4108Q48K64 -- M4 core, CANbus, $2.17, 64K flash, 20K RAM, floating point hardware, HRPWM

### 6.3.9 Infineon Shield2Go Sensor PCB's (similar to Arduino Uno or Mikroe Click, yet different)

- List of Shield2Go
  - https://www.infineon.com/cms/en/tools/landing/infineon-for-makers/shield-2go-my-iot/
  - https://www.infineon.com/cms/en/product/promopages/sensors-2go/#shields2go-myiot

- Arduino Uno to Shield2Go Adaptor
  - https://www.infineon.com/cms/en/product/evaluation-boards/my-iot-adapter/
  - https://www.infineon.com/dgdl/Infineon-WhitePaper_Shield2Go_boards_and_My_IoT_adapter-WP-v01_01-EN.pdf?fileId=5546d46267c74c9a016831aff3ef626d
  - https://www.infineon.com/dgdl/Infineon-S2Go_MyIoT_Fast_flexible_and_easy_prototyping_for_IoT-ABR-v01_02-EN.pdf?fileId=5546d462647040d10164708e322705c4

## 6.4 Microchip Company

### 6.4.1 Microchip PICxx Microprocessors

- Microchip 8bit Low Cost AVR & PIC16 Processors (no CANbus)
  - webpage: https://www.microchip.com/design-centers/8-bit/avr-mcus
  - AVR Guide: http://ww1.microchip.com/downloads/en/DeviceDoc/30010135D.pdf (good)
  - PIC Guide: http://ww1.microchip.com/downloads/en/DeviceDoc/30010068F.pdf (good)

- Microchip Low Power Processors
  - Webpage: https://www.microchip.com/design-centers/lowpower
  - Guide: http://ww1.microchip.com/downloads/en/DeviceDoc/30009941J.pdf

- Microchip 16bit Processors (Pic24, ref boards, little CANbus, low power discussion, motor control)
  - Notes: Pic24F is extreme low power (XLP) yet *not* CANbus
  - Brochure: http://ww1.microchip.com/downloads/en/DeviceDoc/00001032R.pdf
  - Quick Ref: http://ww1.microchip.com/downloads/en/DeviceDoc/30010109F.pdf

- PIC16F18446 Processor ($1, Nano Ref Board, analog comparator, A/D, low power)
  - Webpage: https://www.microchip.com/wwwproducts/en/PIC16F18446
  - Nano Ref Board: https://www.mouser.com/datasheet/2/268/PIC16F18446-Curiosity-Nano-Hardware-User-Guide-500-1507312.pdf

### 6.4.2 Microchip Arduino Products

- Microchip Processors on Arduino Shield
  https://www.microchip.com/sitesearch/search/Product%20and%20Development%20Tools/arduino\

### 6.4.3 Microchip Motor Control

- Motor Control (processors, overview)

http://ww1.microchip.com/downloads/en/DeviceDoc/00000896M.pdf

# 7 REFERENCE: Electrical Signaling

## 7.1 RS-485 Electrical Signaling Protocol

### 7.1.1 RS-485 Overview

- Rs-485 Overview, Wikipedia
  https://en.wikipedia.org/wiki/RS-485
- RS-485 Cable Specification Guide (crosstalk, ringing), Maxim Application Note, 763
  https://www.maximintegrated.com/en/app-notes/index.mvp/id/763
- RS-485/RS-422 Circuit Implementation Guide, Analog Devices, AN#960, very good
  https://www.analog.com/media/en/technical-documentation/application-notes/AN-960.pdf?doc=an-1123.pdf
- Ten Ways to Bulletproof RS-485 Interfaces, AN#1057, very good
  http://www.ti.com/lit/an/snla049b/snla049b.pdf
- A Comparison of Differential Termination Techniques, AN#903
  - For details, see 4 references at bottom of this article: http://www.ti.com/lit/an/snla034b/snla034b.pdf
- Ten Ways to Bulletproof RS-485 Interfaces, AN#1057, very good
  http://www.ti.com/lit/an/snla049b/snla049b.pdf
- RS-485 Cable Specification Guide, AN#763, very good
  https://www.maximintegrated.com/en/app-notes/index.mvp/id/763
- Trim the fat off of RS-485 (i.e. reduce power consumption).
  - For details, see 4 references at bottom of this article: https://www.eetimes.com/document.asp?doc_id=1224773#

### 7.1.2 RS-485 Advantages/Disadvantages

- Advantages: Existing electrical interface
- Disadvantages: burns a lot of power (60mA driver), fast rise time leads to ringing if cable is not properly terminated, -7V to +12V common mode voltage range may not be enough with 16 gauge wire running 15A 100ft (9V drop) therefore this is good w/ shorter networks when 110/220VAC neutral wire is used as a ground reference.

### 7.1.3 RS-485 IC's

- Rs-485 transceiver (PHY), #sp4082e, $0.64, slew rate limited (600nSec), 100K bits/sec, 1mA quiescent pwr,
  > 600nSec rise time is good for 600/20 = 30nSec = 15ft (e.g. multiple PCB's in a box) without termination
  > if you want 300ft, then you need 12µSec rise time
  https://www.maxlinear.com/ds/sp4082e_100_122007.pdf
- Rs-485 transceiver (PHY), #THVD1500, $0.54, -7V to +12V common mode vin, 1M bits/sec
  Shows suggested protection Circuit with varistor and 10ohm series resistors
  http://www.ti.com/lit/ds/symlink/thvd1500.pdf
- Rs-485 transceiver (PHY), Maxim, > +-65V of fault protection, $3
  https://para.maximintegrated.com/en/results.mvp?fam=rs485&267=60

### 7.1.4 RS-485 Interface Boards

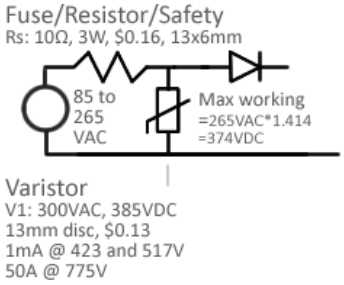
- RS-485 Interface Click board, #MAX3471, (MIKRO-2700)
  Uses UART TX, UART RX, and Output Enable (DE)
  - 5V: https://www.mikroe.com/rs485-2-click
  - 3V: https://www.mikroe.com/rs485-33v-click

## 7.2 Protection against Overvoltage

### 7.2.1 Overview

- If one routes a cable to 30 devices in a room and these are embedded in a wall, one might want to ruggedize electronics to avoid damage. Note that one 4KV 20µSec spike on cable goes to all devices and they might be in hard to get places. Also, if they are co-located with 110/220VAC power, one could accidently touch high VAC to digital+- wires for a moment and cause damage.
- There are several things that need protection against 4KV 20µSec spikes and accidental continuous short to 265VAC:
  - Digital+-: DALI+- wires, RS-485, RS-486P, 0 to 20mA loop, 4 to 20mA loop, and CANbus.
  - DC Power: 48VDC power, 24VDC power, 12VDC power, CANbus 12VDC power
- Surge protection and voltage spikes on power lines, Wikipedia:
  https://en.wikipedia.org/wiki/Surge_protector

### 7.2.2 Protect against 4KVolts 20µSec spikes




- 110/220VAC wires typically include high voltage spikes (e.g. 4KV, 20µSec) which can easily damage components. Electricians will occasionally connect 110/220VAC to device data+- wires, by accident.
- Traditional solution to 4KV spikes is series resistor and varistor, as pictured to the right. Series resistor should be rated for "pulses" (survives big pulses), "flame" (no fire if overpowered), and "fuse" (breaks w/o smoke/fire when overpowered).
- If your current only flows in one direction, then you can add a series diode after the series resistor. This needs to survive in the case where the varistor turns on (e.g. 1000V diode with 775V @ 50A varistor). If you are protecting against continuous short to 265VAC, this needs to not turn on varistor; otherwise varistor burns up.
- This circuit involves a voltage drop across the series resistor. If one is powering a DC/DC converter that outputs 10mA/5V and inputs 2mA/48VDC, for example, then 100Ω series resistance would drop 0.2V (100Ω*2mA), which would be acceptable. Below are example parts that would support this simple microprocessor power supply:
  - Resistor/Fuse: 100ohm, .5W, $0.05, Flame/Safety, 3x6mm, Thru Hole, #NFR25H0001000JR500
  - Varistor, $0.07, 40J, 2.3KA, 7mm disc, 300VAC 385VDC working, 1mA @ 423 … 517V, 10A @775V, #MOV-07D471KTR
  - Diode: 1A/1000V, 1.3Vf @ 1A, 5uA@1000Vr, $0.02, 500nSec, DO-214AC (5x3mm), #FS1M-TP

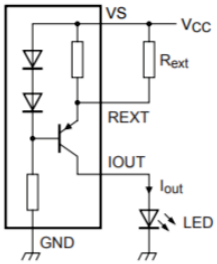
- If one is powering a string of DALI devices with 250mA current, for example, then one might look at a 4Ω series resistor and a 1V drop while signaling recessive (250mA * 4Ω = 1V, 250mW). Below are example parts:
  - Resistor/Fuse: 4ohm, 3W, $0.03, Flame/Safety, 15x5mm, Thru Hole, #FMP300FRF73-4R
  - Varistor: $0.13, 170J, 6KA, 13mm disc, 300VAC 385VDC working, 1mA @ 423 ... 517V, 50A @775V, #ERZ-E11F471

### 7.2.3 Protect against accidental connection to 265VAC

- If devices are co-located with 110/220VAC power then they are susceptible to having their data wires accidently touching this high voltage, for a short or long duration, which could potentially zap all digital devices on the network. If one is driving 3mA for example (e.g. RS-486P), then a 100Ω series resistor would drop 300mV (3mA*100Ω, 1mW) while driving cable capacitance (assuming cable is long), which is acceptable. However, if one puts 220V across this resistor, then 220V^2/100=500W, which is high and will lead to permanent damage. The only way to protect against continuous short to 265VAC is a series switch that opens in the event of overvoltage or overcurrent.

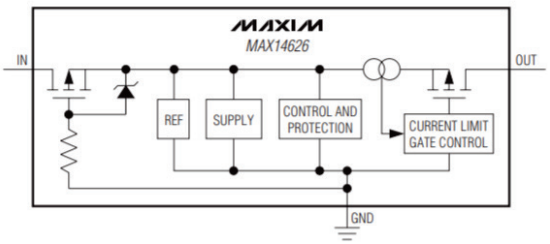
### 7.2.4 Existing Overvoltage Protection IC's

- The $0.83 #FP0100 is an example of a current limiter IC that protects downstream circuits from up to 100V input. Also, if you use an external MOSFET, you can project up to 450V as shown in datasheet Figure 4. This circuit requires 15V headroom between input and output and is not too helpful, yet is instructive. The FP0100 is expensive, yet is helpful at showing how one can protect via switch and current limiting.
  http://ww1.microchip.com/downloads/en/DeviceDoc/FP0100%20C070113.pdf
- The $0.13 #NSV50010YT1G is similar to the #FP0100 yet is much less costly and works with a maximum of 50V instead of 100V. It is available in several different versions that limit current to a fixed 10mA, 15mA, 20mA, 25mA or 50mA. If more is flowing, it opens its transistor and drops voltage. Also, if the voltage drop is too high, then the current is reduced (i.e. it has a power limiter after a current limiter). Notice this part includes a diode voltage drop (e.g. 500mV at 20% of threshold current, and 1800mV at 80%). In some cases this is acceptable, yet in others it is not. Also, this part passes current in one direction, which is only helpful in some cases.
  #NSV50010YT1G: https://www.digikey.com/product-detail/en/on-semiconductor/NSV50010YT1G/NSV50010YT1GOSTR-ND/5258037
  Foldback Current: https://en.wikipedia.org/wiki/Foldback_(power_supply_design)
  Current Circuit: http://www.eeeguide.com/foldback-current-limiting/
- The #PSSI2021SAY is a similar to the above #NSV50010YT1G, yet differs in several ways. This part, pictured to the right, shows how one can utilize an internal self-biased transistor.
  https://assets.nexperia.com/documents/data-sheet/PSSI2021SAY.pdf




- The MAX16010 … MAX16014 are examples of protection IC's. The MAX16011TAC cost $0.70, yet the others cost $2.
  Datasheet: https://datasheets.maximintegrated.com/en/ds/MAX16010-MAX16014.pdf
  Webpage: https://www.maximintegrated.com/en/products/power/supervisors-voltage-monitors-sequencers/MAX16011.html/tb_tab1
- The MAX20313 current limiter is only rated for 5V input, yet shows a typical circuit on datasheet page 13.
  https://datasheets.maximintegrated.com/en/ds/MAX20313-MAX20316.pdf

- The MAX14626, shown to the right, is an example 4 to 20mA protection IC that guards against > 32mA overcurrent. This $0.67 part only protects to 36V, yet if one utilized an external 800V mosfet, this concept could be adapted to protect against more voltage.
  https://datasheets.maximintegrated.com/en/ds/MAX14626.pdf




- List of Maxim overvoltage protection IC's:
  https://para.maximintegrated.com/en/search.mvp?fam=ovpcontrollers&hs=1&tree=interface

### 7.2.5 References

- Adjustable Current Limiting of Smart Power Switches, TI App Note, very good
  http://www.ti.com/lit/an/slva859b/slva859b.pdf
- Active High-Voltage Transient Protectors Trump Conventional Approaches in Automotive Electronics, Maxim App Note, very good
  https://pdfserv.maximintegrated.com/en/an/AN4240.pdf
- System level protection for analog switch, TI App Note
  http://www.ti.com/lit/an/sbaa227/sbaa227.pdf
- Series Protection for Power-Line Transients, Maxim App Note
  https://pdfserv.maximintegrated.com/en/an/AN3472.pdf
- Low Voltage Fault Protection, Maxim App Note
  https://www.maximintegrated.com/en/app-notes/index.mvp/id/2854
- Add fault protection for 0-to-20mA loop, Maxim App Note
  https://pdfserv.maximintegrated.com/en/an/AN3195.pdf
- Handy Circuit Gives Systems Flexible Fault Protection, Maxim App Note
  https://pdfserv.maximintegrated.com/en/an/AN2378.pdf
- Snubber Protection, Maxim App Note
  https://pdfserv.maximintegrated.com/en/an/AN848.pdf
- Protection: How much is enough for an analog output?
  https://pdfserv.maximintegrated.com/en/an/AN6008.pdf
- For more information on circuits that attach to DC power wires (e.g. 48VDC power), see: GWeinreb_Manhattan2_ResearchNotes.xlsx / "Export" / "Tiny Power Supplies for Tiny Processors".

## 7.3 LINbus References

### 7.3.1 LINbus Overview (1 wire serial bus, 1 master, 1 to 16 slaves)

- LINbus, Wikipedia
  https://en.wikipedia.org/wiki/Local_Interconnect_Network
- Mikroe Provides LINbus Click board (#ATA663211)
  https://www.mikroe.com/ata663211-click

## 7.4 Peripheral Single Wire Protocol Master Interface (SWPMI, 1wire+gnd between processor and 1 chip)

- ST Explanation of SWPMI (good)
  https://st-onlinetraining.s3.amazonaws.com/STM32L4_Peripheral_Single_Wire_Protocol_Master_Interface_(SWPMI)/index.html
- Single Wire Protocol, Wikipedia
  https://en.wikipedia.org/wiki/Single_Wire_Protocol

# 8 REFERENCE: Building Automation Standards

## 8.1 Networking Protocols

### 8.1.1 List of All Network Busses

- List of All Wired Network Busses, Wikipedia
  https://en.wikipedia.org/wiki/List_of_network_buses
- Overview of Building Network Protocols (KNX, LonMark, BACnet, EnOcean, ZigBee, DALI) Thesis, 80pgs
  https://www.iea.lth.se/publications/MS-Theses/Full%20document/5267_full_document.pdf

### 8.1.2 Communications Protocols

- Communications Protocols, General Discussion, Wikipedia
  https://en.wikipedia.org/wiki/Communication_protocol
- List of Communications Protocols, Wikipedia
  https://en.wikipedia.org/wiki/Lists_of_network_protocols
- List of Automation Protocols, Wikipedia
  https://en.wikipedia.org/wiki/List_of_automation_protocols
- Summary of Automation Protocols, Siemens
  https://www.downloads.siemens.com/download-center/Download.aspx?pos=download&fct=getasset&id1=A6V10209534
- Carrier Sense Collision Avoidance (in data link layer)
  https://en.wikipedia.org/wiki/Carrier-sense_multiple_access_with_collision_avoidance

## 8.2 BACnet (building automation network)

### 8.2.1 BACnet Summary

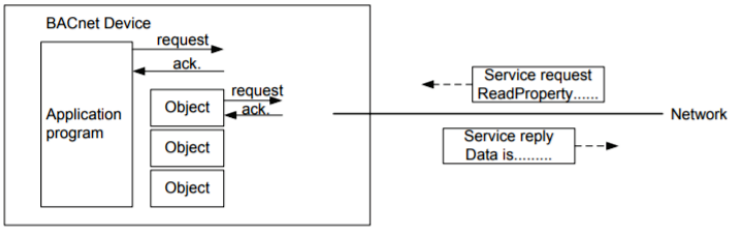
- Automation and Control. Good with HVAC and analog input/output.
- Devices contain a set of objects (e.g. Class in C++)
- BACnet defines standardized Objects
  - E.g. Analog Input (includes PresentValue property), Analog Output, Binary Input, Binary Output, etc.
  - Objects have required and optional properties.
  - Output objects (e.g. Analog Output) are "commandable". One specifies priority when they write.
  - More complex objects such as Scheduler.
  - List of ~60 Objects (e.g. LED Lighting Output): https://en.wikipedia.org/wiki/BACnet#BACnet_objects
- 35 types of messages ("services", "subroutines"), 5 different classes of messages.
- ReadProperty() and WriteProperty() must identify property via: device instance (i.e., which device on the network), object type (analog input, binary input), object instance (i.e., which analog input), property (present value, object name, status flags)

- Change of Value (COV) message is broadcast when something happens (e.g. user flips light switch)
- Devices "subscribe" to other's device's COV (Change of Value)
- System supports Alarm, Events, and File uploading/downloading, Virtual terminal functions
- Router: Moves data and does not translate it.
  Gateway: Translates message from one protocol to another (e.g. LonTalk to BACnet).
- BACnet IP: Messages have IP address
  BBMD: BACnet/IP Broadcast Management Device (device broadcasts msgs on local network)
- BACnet MS/TP - Network with only one master at a time and all other devices are Slaves. Device with Token is the Master. All Slaves listen while Master talks. An example of this is an RS-485 network with multiple devices that are talking on the same 2 data wires (i.e. only device with token can talk).
- Active Physical Window Support: provides "Multi-state Output" yet no support for blind, window
  https://www.digitalilluminationinterface.org/dali/standards.html
  http://www.bacnet.org/Bibliography/ES-7-96/ES-7-96.htm

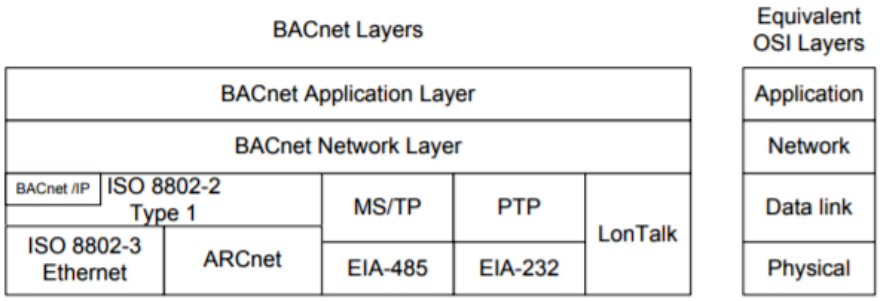
### 8.2.2 BACnet Overview

- BACnet Overview, Wikipedia
  https://en.wikipedia.org/wiki/BACnet
- BACnet Organization, Wiki Site, very good
  http://www.bacnetwiki.com/wiki/index.php?title=Main_Page
- BACnet 101, 2pg Summary, very good
  https://www.csimn.com/CSI_pages/BACnet101.html
- BACnet, 3pg summary, see pdf page#36
  https://www.iea.lth.se/publications/MS-Theses/Full%20document/5267_full_document.pdf
- Book: BACnet: The Global Standard for Building Automation, 400 page book, on gsw kindle
  If programming BACnet, this book is ESSENTIAL
- BACnet Explained, By Newman & Wichenko, 10pgs, very good
  - Part 1: https://www.ashrae.org/File%20Library/Technical%20Resources/Bookstore/BACnet-explained-Pt1.pdf
  - Part 2: https://www.ashrae.org/File%20Library/Technical%20Resources/Bookstore/BACnet-Explained-Pt2.pdf
- To examine C source code, open:
  … Manhattan2\projects\BACnet Interface\Weinreb_VisualStudio_Bacnet_project\Bacnet_gsw_test
  … Manhattan2\projects\BACnet Interface\bacnet-stack-0.8.6, unzipped\ports\win32\Microsoft Visual Studio 2010
- Free BACnet C Source Code, very good
  - Source: https://sourceforge.net/projects/bacnet/?source=directory
  - Documentation: http://bacnet.sourceforge.net/ Links to Free Source Are Shown Here
  - Analog Input: …bacnet-stack-0.8.6, unzipped\ports\atmega168\ai.c, .h
  - Ai.c http://svn.code.sf.net/p/bacnet/code/trunk/bacnet-stack/demo/object/ai.c
  - Device.c http://svn.code.sf.net/p/bacnet/code/trunk/bacnet-stack/demo/object/device.c
- BACnet is controlled by industry organization ASHRAE
  "American Society of Heating, Refrigerating and Air-Conditioning Engineers"
  https://www.ashrae.org/technical-resources/bookstore/bacnet

- BACnet protocol defines a number of data link / physical layers, including Master-Slave/Token-Passing (MSTP) over RS-485, ZigBee, and LonTalk.
  https://en.wikipedia.org/wiki/BACnet#Protocol_overview
- BACnet addressing summarized on page 8:
  https://dms.hvacpartners.com/docs/1000/Public/04/11-808-417-01.pdf
- Master-Slave/Token-Passing (MSTP) Timing
  https://www.cimetrics.com/blogs/news/bacnet-ms-tp-timing-explained
- User's Guide for typical RS-485 based BACnet system
  http://www.neptronic.com/Controls/PDF/EVC/BACnetModbus/BACnet%20MSTP%20Overview%20Manual-160405.pdf
- Book: BACnet for Beginners, 83pgs, very good
  http://www.modbusbacnet.com/includes/pdf/Bacnet%20For%20Beginners.pdf
- MSTP Introduction, page 28 in below book (only device with token can initiate, to avoid collision)
  http://www.modbusbacnet.com/includes/pdf/Bacnet%20For%20Beginners.pdf
- BACnet objects receive "request" and reply with "data".




- BACnet supports multiple physical and data layers.




- Below is a typical BACnet object:

Example of the properties and their values for a BACnet data object.

| | |
|---|---|
| **object-identifier:** | *analog-input [180]* |
| **object-name:** | *One_sec_Ph_A-NVolt* |
| **object-type:** | *analog-input* |
| **present-value:** | *100.000000* |
| **status-flags:** | *In-Alarm=[false], Fault=[true], Overriden=[false], Out-Of-Service=[false]* |
| **event-state:** | *normal* |
| **reliability:** | *unreliable-other* |
| **out-of-service:** | *False* |
| **units:** | *Volts* |
| **description:** | *Zero length/empty string* |

## 8.3 CANbus

### 8.3.1 CANbus Summary

- CANbus Features: Multi-master (not master/slave), everyone broadcast to everyone else, everyone maintains values of "parameters" (ParameterIndex and ParameterValue in ram).
- Four Frame Types: Data Frame (0 to 8 data bytes), Remote Frame (device asks someone to send data), Error Frame (any device reports error), Overload Frame (device reports that data frames are too fast & need to slow down).
- Higher Protocols Based on CANbus: DeviceNet (industrial automation), CANopen (industrial automation), etc.
- Rise/fall time is typically < 25% of the bit duration, as noted in section 4.10 of application note SLLA270. For more details, search for "CANBus Physical Layer Signaling" in this file.
- CAN_H/CAN_L are both at ~2.5V for logic 1 (recessive, released) and then pulled to ~3.5V/~1.5V for logic 0 (dominant, bus driven). Dropping ~2V drop across ~60ohms involves ~33mA.
- CANbus uses Non-Return-To-Zero (NRZ, not-Manchester) encoding. The below picture on the left is NRZ whereas the other is Manchester.

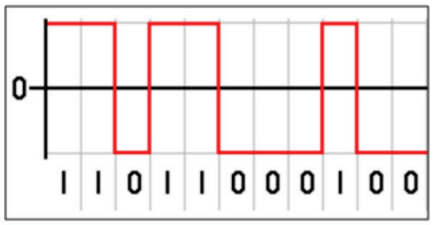



*Figure 1: An example of NRZ encoding*

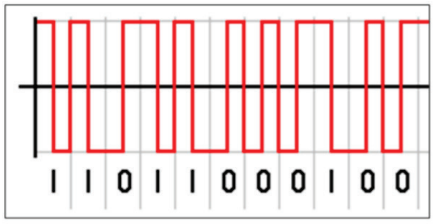
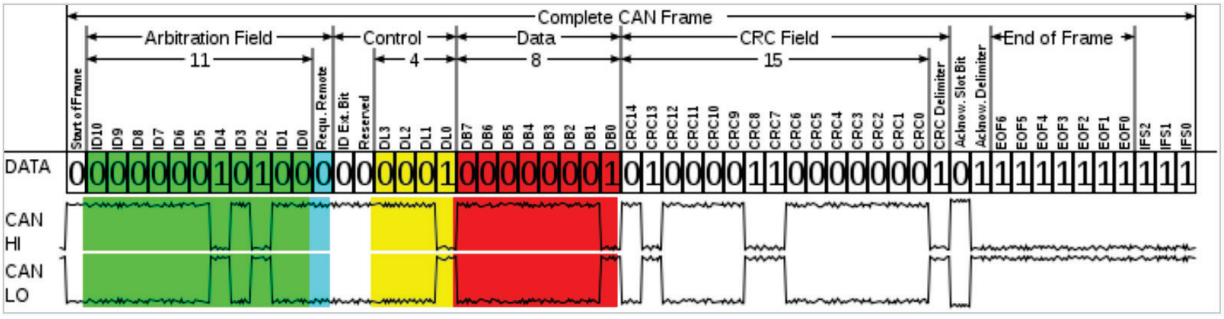



*Figure 2: An example of Manchester encoding*

  - NRZ: https://en.wikipedia.org/wiki/Non-return-to-zero
  - Manchester: https://en.wikipedia.org/wiki/Manchester_code

- The below picture shows a frame of CAN data.




- The below picture shows the CANbus OSI 7-layer model:

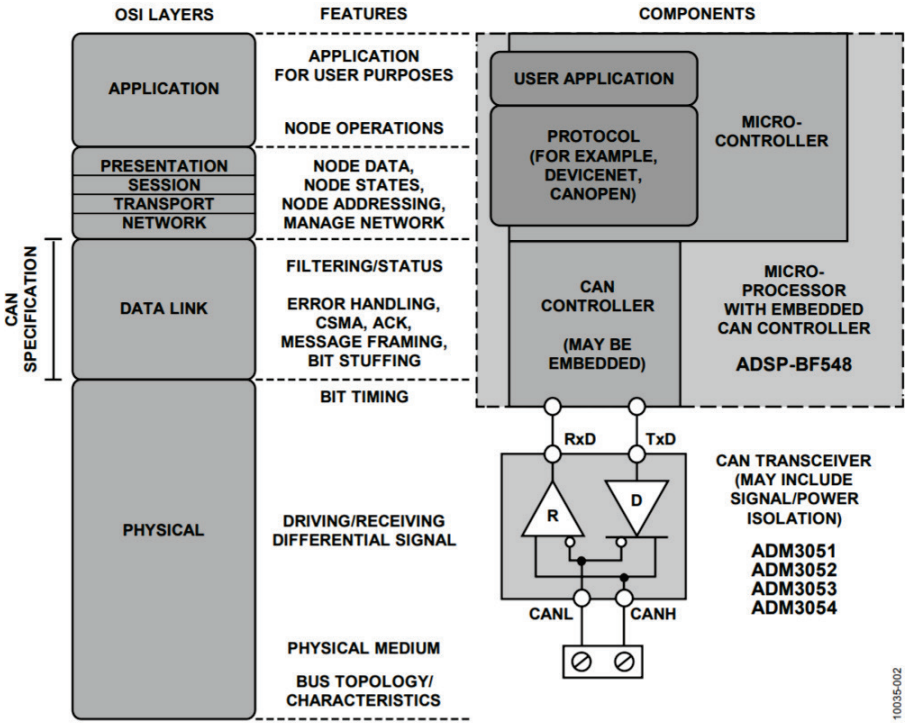



*Figure 2. CAN Implementation Blocks as Related to OSI Layers and Features*

### 8.3.2 CANbus Overview

- CANbus, Wikipedia
  https://en.wikipedia.org/wiki/CAN_bus
- CANbus Guide, By Wilfried Voss, very good
  http://www.copperhilltechnologies.com/online-books/a-comprehensible-guide-to-controller-area-network/
- CANbus Guide – Message Frame Format, By Wilfried Voss, very good
  http://www.copperhilltechnologies.com/can-bus-guide-message-frame-format/
- Learn CANbus, by Mikroe Company
  https://www.mikroe.com/blog/can-bus
- Intro To CANbus (1pg summary)
  https://learn.sparkfun.com/tutorials/ast-can485-hookup-guide/all#introduction-to-can-bus
- CANbus Guide to Resources and Vender webpages, good
  http://www.interfacebus.com/CAN-Bus-Description-Vendors-Canbus-Protocol.html
- Voss's webpage with ~50 links to useful resources, including CANbus monitoring/debuggers, good
  https://copperhilltech.com/documentation/

### 8.3.3 CANBus Physical Layer Signaling

- CANbus Physical Layer User's Guide, good
  http://www.inp.nsk.su/~kozak/canbus/canphy.pdf
- A CAN Physical Layer Discussion, Microchip #AN228, good
  http://ww1.microchip.com/downloads/en/appnotes/00228a.pdf
- Controller Area Network Physical Layer Requirements, TI #SLLA270, good
  https://www.ti.com/lit/an/slla270/slla270.pdf?ts=1596997033981&ref_url=https%253A%252F%252Fwww.google.com%252F

- CANbus, Electrical Explanation, Analog Devices #AN1123, good
  https://www.analog.com/media/en/technical-documentation/application-notes/AN-1123.pdf

### 8.3.4 CANbus Protocols

- CANbus, Lower-Layer Standards, Wikipedia
  https://en.wikipedia.org/wiki/CAN_bus#CAN_lower-layer_standards
- CANbus, Higher-Layer Standards, Wikipedia
  https://en.wikipedia.org/wiki/CAN_bus#CAN-based_higher-layer_protocols
- Infineon CANbus Controller hardware module (MultiCan)
  https://www.infineon.com/dgdl/Infineon-MultiCAN-XMC4000-AP32300-AN-v01_00-EN.pdf?fileId=5546d4624e765da5014ed91d6be32110
- Different types of Encoding (Manchester, NRZ)
  http://cdn.teledynelecroy.com/files/whitepapers/configurable_protocol_decoding_manchester_and_nrz_encoded_signals_whitepaper.pdf
- Protocol transmits 4095 byte data packets via multiple CANbus 29bit frames, ISO 15765-2, Transport Layer, good
  https://en.wikipedia.org/wiki/ISO_15765-2

### 8.3.5 CANbus Books

- A Comprehensible Guide to Controller Area Network, Voss, $15, good
  https://www.amazon.com/dp/0976511606/ref=cm_sw_em_r_mt_dp_S8-nFbPJY5M53
- Embedded Networking with CAN and CANopen, Pfeiffer et al, $35, good
  https://www.amazon.com/Embedded-Networking-CANopen-Olaf-Pfeiffer/dp/0692740872

### 8.3.6 CANopen Software above CANbus Layer

- CANopen, Protocol for managing devices and parameters within devices, Wikipedia
  https://en.wikipedia.org/wiki/CANopen
- MicroCANopen User's Manual
  http://www.esacademy.org/products/getfile.php?filename=MicroCANopen%20Manual.pdf

### 8.3.7 CANbus Interface IC's

- CANbus transceiver (PHY), $0.38, 10mA quiescent, 70mA pwr when tmit, #IFX1050GVIO
  https://www.infineon.com/cms/en/product/transceivers/industrial-transceiver/ifx1050g-vio/
- CANbus transceiver (PHY), $0.68, 10mA quiescent, 70mA pwr when tmit, #MCP2561
  https://www.microchip.com/wwwproducts/en/MCP2561
- Microchip CANbus PHY and MAC Interface IC (#ATA6561, $0.43, Listen 3mA, transmit logic "0" 70mA (drive 120ohm), transmit logic "1" 5mA
  https://www.microchip.com/wwwproducts/en/ATA6561
- CANbus Transceiver (PHY), $1.46, external resistor sets rise/fall time (via feedback?), #MAX3051
  https://datasheets.maximintegrated.com/en/ds/MAX3051.pdf
- CANbus Transceiver (PHY), $2.36, ±65V protection, ±25V cmv, #MAX13054A
  https://www.maximintegrated.com/en/products/interface/transceivers/MAX13054A.html

### 8.3.8 CANbus Click (not Arduino Nano) Interface Boards

- CANbus interface Click board (#ATA6563)
  https://www.mikroe.com/ata6563-click
- CANbus 3.3V and 5V interface Click board (#MCP2561 PHY, #MCP2515 MAC)
  https://www.mikroe.com/can-spi-33v-click
  https://www.mikroe.com/can-spi-5v-click
  https://www.mikroe.com/mcp25625-click
- Isolated CANbus interface Click board, ADM5053 ($8), #MIKROE-2627
  https://www.mikroe.com/can-isolator-click
  https://download.mikroe.com/documents/add-on-boards/click/can-isolator/can-isolator-click-schematic.pdf (schematic)

### 8.3.9 CANbus Arduino Nano (not Click) Interface Boards

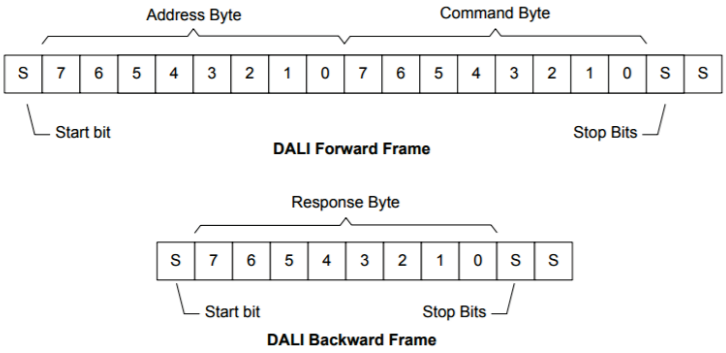
- Arduino Nano CANbus interface board, uses #MCP2515 MAC, #TPS54232 buck converter
  https://store.arduino.cc/usa/mkr-can-shield

### 8.3.10 CANbus Advantages/Disadvantages

- Advantages: Existing interface, many processors provide MAC and/or PHY
- Disadvantages: Burns lots of power, transmitter typically drives 120Ω termination (or you can reduce yet you need to be careful not to get ringing), designed for 1M bit/sec daisy-chain with termination on both ends (not tree topology).

## 8.4 DALI (Digital Addressable Lighting Interface)

### 8.4.1 Summary




- DALI Features: 1200 bits/sec, 1 master and multiple slaves (except DALI 2 which supports slave initiation).
- 16V/250mA shorted to 0V is logic "0", 2 data wires typically routed alongside 110/220VAC, supports tree topology wiling, device can be powered from DALI wires (2mA max) yet this is rare, $0.10 opto-coupler requires 2mA.
- Speed is kept low (1200bps, slow rise/fall time); subsequently, no termination resistors are required and data is transmitted using relatively high voltages (0V ± 4.5V for logic "0", and 16V ± 6.5V for logic "1") enabling reliable communications in the presence of significant electrical noise.

### 8.4.2 DALI Overview

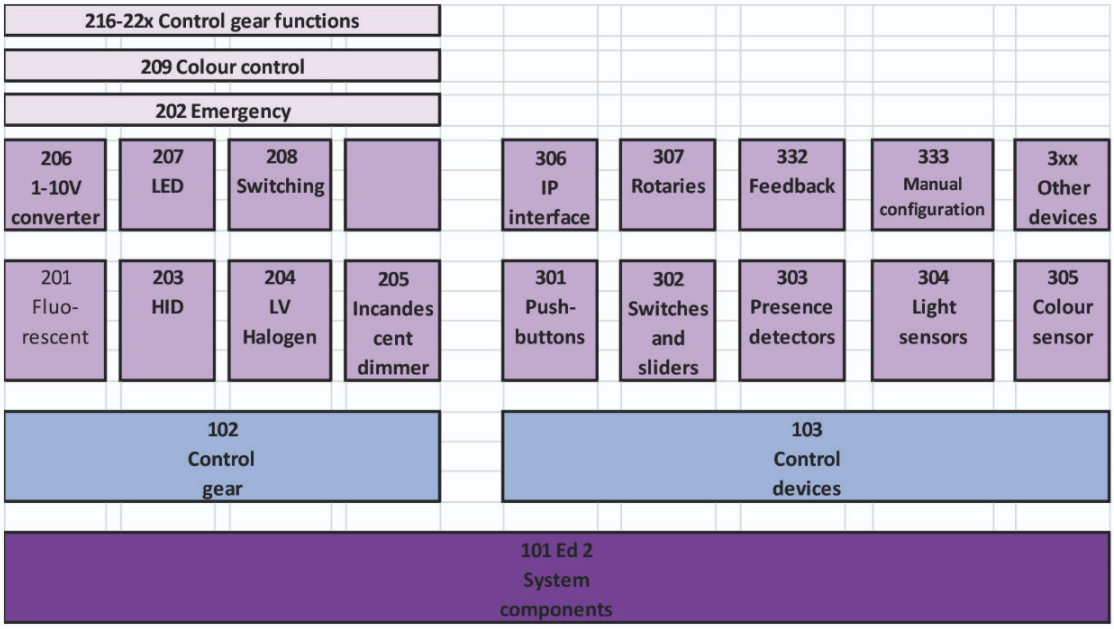
- DALI Overview, Wikipedia
  https://en.wikipedia.org/wiki/Digital_Addressable_Lighting_Interface

- List of Device Models (e.g. light, occupancy sensor) and Specifications (good)
  https://www.digitalilluminationinterface.org/dali/standards.html
- DALI, 5pg summary, see pdf page#48
  https://www.iea.lth.se/publications/MS-Theses/Full%20document/5267_full_document.pdf
- DALI User's Guide (good description)
  http://artisticlicence.com/WebSiteMaster/User%20Guides/the%20dali%20guide.pdf
- DALI Consortium ($5k/yr membership fee)
  https://www.digitalilluminationinterface.org/membership/benefits.html
- DALI Product Database
  http://www.digitalilluminationinterface.org/products
- DALI Explained, magazine article
  https://www.buildings.com/news/industry-news/articleid/1463/title/dali-explained#targetText=LED%20drivers%20and%20ballasts%20are,to%20a%20DALI%20control%20device.
- ABC's of DALI, by Philips
  https://cedia.a2zinc.net/CEDIA2018/Custom/Handout/Speaker0_Session4655_11.pdf
- History of DALI, By Henk Rotman, Philips Lighting South Africa, 2017
  https://www.ee.co.za/article/development-dali-interface.html
- Multiple DALI Products, Brochure, by Beckhoff Company
  https://download.beckhoff.com/download/document/Application_Notes/DK9222-0810-0031.pdf
- Description of Many DALI Products, Tridonic Company
  https://www.tridonic.se/se/download/technical/DALI-manual_en.pdf
- Example software interface for LED controller device ("Control Gear"), Infineon Company
  https://www.compel.ru/wordpress/wp-content/uploads/2018/06/ap32400_dali2_0_control_gear_stack_v1.pdf
- Active Physical Window Support: general "102 control category", "103 Control Devices", and "307 Rotaries" yet no support for blind, motor, window
  https://www.digitalilluminationinterface.org/dali/standards.html

### 8.4.3 DALI PowerPoint Presentations

- DALI 2 Summary (24pgs, Oct 2018, good)
  https://www.digitalilluminationinterface.org/data/downloadables/9/4/1810_lux-webcast-diia-presentation-oct-2018.pdf
- Introducing DALI (organization, overview, certification, testing)
  Mr. Ruud van Bokhorst, General Manager, DALI, GM@digitalilluminationinterface.org
  https://www.digitalilluminationinterface.org/data/downloadables/1/0/2/1903_diia-introduction-v22_mar2019.pdf
- TN1711, Differences between DALI-1 and DALI-2 (2pgs)
  https://www.digitalilluminationinterface.org/data/downloadables/5/4/1711_technical-note-dali-2-the-new-standard.pdf

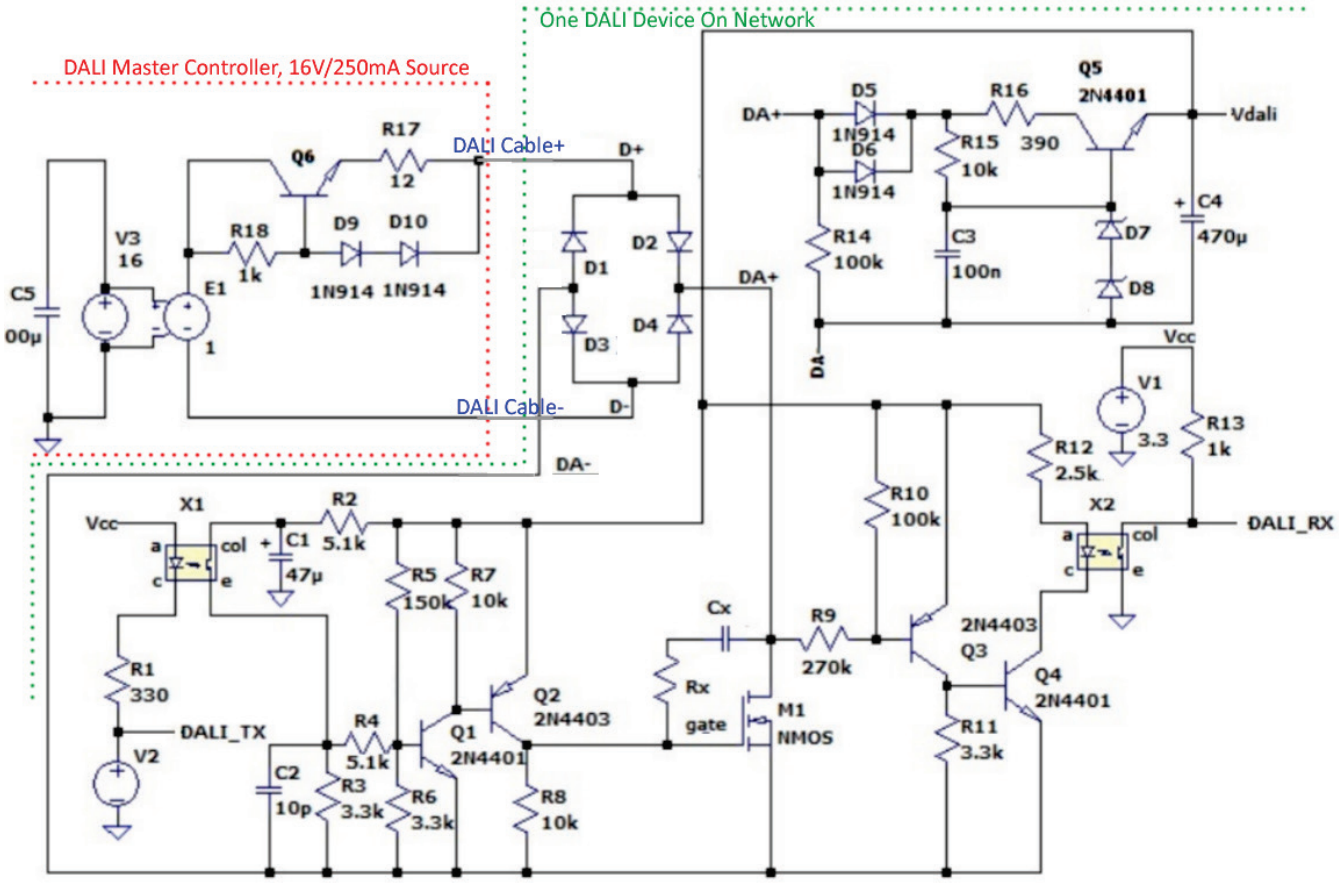
### 8.4.4 IEC 62386 standard that define DALI software




### 8.4.5 DALI Cable, Electrical Specification

- DALI LIGHTING CONTROL INTERFACE CIRCUIT AND ELECTRICAL CHARACTERISTICS OF A TYPICAL DALI CABLE (paper, cable interface), Mika Maaspuro
  http://www.arpnjournals.com/jeas/research_papers/rp_2015/jeas_0915_2476.pdf

  Below is a copy of Mika's circuit. This shows the DALI Network Master Controller (16V/250mA source) at the upper left, and one DALI Device lower and right. The DALI Cable connects the two and is labeled D+/D-. In summary, the Device transmits by energizing X1, Q1, Q2 and finally mosfet M1 (which shorts out the 16V DALI cable and causes it to limit at ~1V/250mA). The Device receives by detecting DALI low voltage at Q3, Q4, and then X2.




D1/D2/D3/D4 is a bridge at the Device that enables one to swap DALI D+/D- wires without degrading operation.

Opto-couplers X1 and X2 enable one to isolate their PCB COM from the DALI D+/D- network. Consequently, the Earth Ground wire that connects devices can be broken and operation is not degraded. If one wants to connect devices and not rely on a correctly wired Earth Ground wire, then opto-couplers are probably required.

< 10V across DALI wires D+/D- is like receiving a logic 0; whereas shorting these two wires via mosfet M1 is like transmitting a logic 0.

Much of the device circuit could be consolidated into a transceiver support IC while several components are kept external (e.g. 600V bridge, 600V mosfet, 600V sense resistor, and opto-coupler).

The DALI circuit supports 1200bits/sec; however, researchers might be able to adapt this approach to support speeds that are 10 to 30 times faster.

### 8.4.6 DALI Master Controller

- Example DALI Master Controller
  https://www.wago.com/wagoweb/documentation/750/eng_manu/modules/m07500641_00000000_0en.pdf

### 8.4.7 DALI Slave Reference Designs

- DALI Click board (interface from DALI to TTL), #MIKROE-1297

https://www.mikroe.com/dali-click
https://download.mikroe.com/documents/add-on-boards/click/dali/dali-click-manual-v100.pdf (schematic)
https://libstock.mikroe.com/projects/view/499/dali-3-3v-click-example (C source code in .zip file)

- DALI Click board (interface from DALI to TTL, schematic), see "Mikroe" discussion for details
- Example of DALI slave device (AtTiny817 ($1), schematic)
  http://ww1.microchip.com/downloads/en/DeviceDoc/50002631A.pdf
- Example of DALI slave device (AtTiny817 ($1), schematic)
  http://ww1.microchip.com/downloads/en/DeviceDoc/50002631A.pdf
- DALI based light and motor control system for movable spot luminaires, Paper
  https://ieeexplore.ieee.org/document/8089199
- Microchip DALI Reference Boards http://ww1.microchip.com/downloads/en/DeviceDoc/30010020A.pdf
  #DM160214 Lighting Ref Board: https://www.microchip.com/DevelopmentTools/ProductDetails/PartNO/DM160214
  #DV160214-1 Lighting Ref Board: https://www.microchip.com/developmenttools/ProductDetails/DV160214-1
  User's Guide: https://ww1.microchip.com/downloads/en/DeviceDoc/40001717A.pdf
- Microchip DALI Development Kit, Digikey, $190
  https://www.digikey.com/product-detail/en/microchip-technology/DV160214-1/DV160214-1-ND/4147277
- ST Microsystems DALI Interface Schematic (good, no capacitor on power supply, 5µSec TC tx)
  https://www.st.com/content/ccc/resource/technical/document/data_brief/60/23/00/3a/93/2d/48/4b/CD00294976.pdf/files/CD00294976.pdf/jcr:content/translations/en.CD00294976.pdf
  https://www.st.com/en/evaluation-tools/steval-ilm001v1.html
- How to protect against short to 230VAC:
  https://electronics.stackexchange.com/questions/429641/understanding-steval-ilm001v1-dali-transceiver-schematic-adding-protections
  One participant's design:
  https://share.ondrovo.com/2019-05-16/Screenshot_20190516_154004.png
- Infineon Xmc1302 DALI Slave Reference Design, drives LED (on/off/dimmer) with buck converter given 24 to 48VDC power, < 800mA to LED, very good
  https://www.infineon.com/cms/en/product/evaluation-boards/kit_xmc1_led_cc_exp_001/
- Infineon Xmc1302 DALI Slave Reference Design w/ schematic and source code.
  Uses #ADM2687 $11 tri-opto-coupler and #Ltc3630 buck converter (costly).
  https://www.infineon.com/cms/en/product/evaluation-boards/kit_xmc_led_dali_20_rgb/
- STM32 DALI slave ref design w/ schematic and source code. Utilizes the following processors:
  STM32F .. ($1.42), #UM1728
  STM32L .. ($0.62), #UM1629
  STM8L .. ($0.42), #UM1632
  - https://www.st.com/en/evaluation-tools/steval-ilm001v1.html
  - https://www.st.com/content/ccc/resource/technical/layouts_and_diagrams/schematic_pack/16/ad/3f/05/ec/5c/44/63/steval-ilm001v1_schematic.pdf/files/steval-ilm001v1_schematic.pdf/jcr:content/translations/en.steval-ilm001v1_schematic.pdf

### 8.4.8 DALI Over-Current and Over-Voltage Protection

- One can consider adding over-current and over-voltage projection to the DALI interface. For an example of this, see pdf schematic and TINA simulation file. These files show an example of a DALI Interface that is both current limited (to 400mA) and voltage limiting (to 24V). One can short DALI wires to 265VAC w/o damage. This has only been simulated, not prototyped (it might be buggy). It uses a $3.60 110/220VAC-to-5VDC isolated power supply module to create 5V pwr for processor and electronics. Since this module isolates via transformer, optically isolators are

not placed between DALI and processor. Alternatively, one might have a non-isolated power supply and use opto-isolators.

If one placed 22V (have not yet reached voltage limiter) across DALI wires with unlimited source current, then our current limiter turns on at 400mA and puts .4A * 22V = 10Watts across our output transistor (too much) -- this is an example of a vulnerable area. Instead of voltage and/or current limiting, one might place a 110°C temperature sensor near the MOSFET (e.g. # TMP302DDRLR, $0.29).

This design uses transistors, yet one could consider something similar using MOSFETS.

One might also consider adding a series resistor and Varistor (TVS) to project against 20uSec >400V spikes.

There are several methods of protection: sense high voltages and turn off switch, current limiter, voltage limiter, and temperature sensor IC detects excess temperature (e.g. 105°C) and turns off switch.

For details, search GWeinreb_Manhattan2_ResearchNotes.xlsx for "Varistor" and "Protected DALI Interface".

### 8.4.9 DALI Reference Designs That Drive LED's

- Reference Design, Xmc1200 processor & BCR450 LED Driver via DALI, By Infineon Company. Uses DALICG02 software for DALI stack and MAC01 to rcv/tmit frames (Manchester encoding).
  - Presentation: https://www.infineon.com/dgdl/Infineon-Application_Lighting_Digital_Addressable_Lighting_Interface_(DALI)_Control_Gear-TR-v01_00-EN.pdf?fileId=5546d4625696ed76015698e9d86c7cc5
  - Software: https://www.compel.ru/wordpress/wp-content/uploads/2018/06/ap32400_dali2_0_control_gear_stack_v1.pdf
  - Free Download: https://www.infineon.com/dgdl/Infineon-DALIv2_Download_Info-SW-v01_00-EN.pdf?fileId=5546d462625a528f01628bbc77826152

### 8.4.10 DALI Advantages/Disadvantages

- Advantages: Existing interface, good support for lights, good isolation from 110/220VAC
- Disadvantages: Slow (1200bits/sec), 16V/250mA (4W) wastes power when shorted low, need opto-couplers, assumes 110/220VAC is available when doing control, no good interface IC's (lots of overhead), rise time is not controlled (ringing risk?)

### 8.4.11 Different Isolation Techniques

- Understanding Signal and Power Isolation Techniques
  https://www.electronicdesign.com/power/understanding-signal-and-power-isolation-techniques
- Summary of Analog Devices ADUM products ($1 min cost)
  https://www.analog.com/en/technical-articles/micro-transformers-provide-signal-and-power-isolation.html

### 8.4.12 DALI Comments

- One burns less power if slave receive opto-coupler diode is off while in idle.
- 100Ω 0.5W fuse/resistor and varistor can better protect DALI circuit from overvoltage.

- If one is driving LED's, then is it better to place processor on DALI data+- side of opto-couplers and use opto-isolated gate drive MOSFET (e.g. $0.34, 600mA) or place processor on 110/220VAC side of opto-isolators and have 110/220VAC-to-5VDC power supply?

### 8.4.13 DALI Standard Documents

- DALI Part 207 - Particular Requirements for Control Gear - LED Driver Modules 62386-207-2009
  https://u.dianyuan.com/upload/space/2011/09/07/1315386342-525930.pdf
- Part 250 Power Supply, Preliminary
  https://www.digitalilluminationinterface.org/specifications/download.html

### 8.4.14 DALI Components

- Opto-coupler, $0.10, 2mA through diode gets you 50µSec, transistor output, #TV-357T
  https://www.mouser.com/ProductDetail/Lite-On/LTV-357T?qs=sGAEpiMZZMteimceilVCB4LDcV%252BVgRO4QTxJvewiAqo%3D
- Opto-coupler, $0.24, 1.6mA through diode gets you 4µSec, OC out, Schmidt trigger, #H11L1
  https://www.digikey.com/product-detail/en/everlight-electronics-co-ltd/H11L1S-TA/1080-1201-2-ND/2675692
- MOSFET, $0.22, 0.9A, 600V, DPAK, #SSR1N60BTM-WS
  https://www.digikey.com/product-detail/en/on-semiconductor/SSR1N60BTM-WS/SSR1N60BTM-WS-ND/2061756

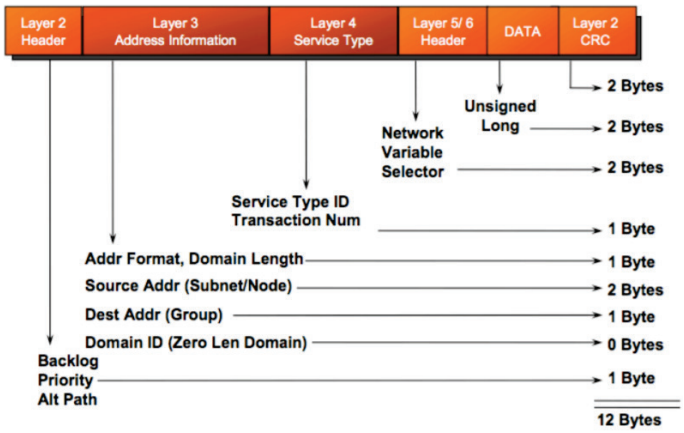
## 8.5 LonWorks

### 8.5.1 LonWorks Overview

- LonWorks, Wikipedia
  https://en.wikipedia.org/wiki/LonWorks
- LonWorks, 4pg summary, see pdf page#32
  https://www.iea.lth.se/publications/MS-Theses/Full%20document/5267_full_document.pdf
- LonWorks 101 (layers, 60pg presentation)
  https://www.lonmark.org/connection/presentations/TFM2007/LonWorksTechnology101.pdf
- Active Physical Window Support: SFPTactuateSunshade, SFPTsunblindActuator, SFPTsunblindController
  https://www.lonmark.org/fps/

### 8.5.2 Summary

- Each device is a set of "Functions", and over 80 Functions have been defined (e.g. analog input, led driver).
- 750 products have been certified.
- Any device talks to any other, no master/slave.
- SNVTs (Standard Network Variable Types), e.g. temperature is -274°C to 6279.5°C, 0.1°C units, uint16 internal variable.



- Echelon sells microprocessors w/ built-in MAC interface.
- Unique ID for each device (48bits, serial number, "Neuron-ID").
- All devices communicate via the LonTalk protocol (i.e. ANSI/EIA/CEA-709.1).
- System Elements: Networks, Domains, Subnets, Groups, Channels, and Devices.
- ≤127 devices in a subnet, ≤255 subnets in a domain, ≤32K devices in a domain, < 2^48 domains in a network, ≤255 groups in a domain, any number of channels in a network.

### 8.5.3 LonWorks Data Layers

- Data Layer is ISO/IEC 14908, Part 1/2/3/4
  https://www.lonmark.org/technical_resources/standards
  https://gostperevod.com/catalogsearch/result/?q=ISO%2FIEC+14908
  https://en.wikipedia.org/wiki/LonTalk#Protocol
- Download Files
  - Guidelines: http://www.lonmark.org/technical_resources/guidelines/developer
  - Resource Files: http://www.lonmark.org/technical_resources/resource_files/

### 8.5.4 Application Layer

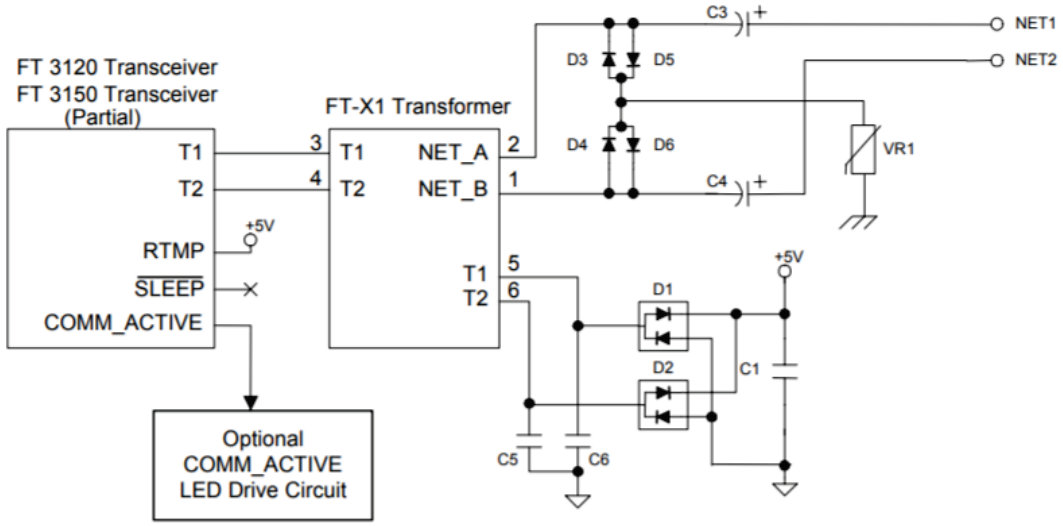
- Device Classes (e.g. HVAC, LED Controller, Temperature Sensor)
  https://www.lonmark.org/fps/
- SNVT Master List (standardized variables)
  http://www.lonmark.org/technical_resources/resource_files/1500/snvt.pdf
- SCPT Master List (standardized configuration properties, enum)
  http://www.lonmark.org/technical_resources/resource_files/1500/scpt.pdf

### 8.5.5 LonWorks Physical Layers

- Physical Layer -- Official LonWorks Guideline
  http://www.lonmark.org/technical_resources/guidelines/docs/LmPhy34.pdf
- Physical layers (approximately 10 to choose from):
  - TP/FT-10: tree topology, 1 terminator 50Ω + 100µF, 78Kbps, 3m max stub length, 500m if tree (free) topology and 2700m if bus (daisy-chain), 22awg, transformer isolation, 5MHz clock, e.g. LonWorks 3050 IC, CEA Specification #709.3-A, Manchester encoding.
  - TP/LP-10: link power ("LP"), similar to above yet device gets power from data+-, 16awg.
  - TP/XF-1250: daisy-chain ("bus"), 1250kbps, 500Ω termination at each end, used w/ backbone, 130m, transformer coupled ("XF"), 22awg, short circuit protected to 260VAC.
  - TP/XF-78: similar to the above yet 78kbps.
  - PLT-22: Powerline.
  - RS-485-39: twisted pair, 39kbps ("-39"), 120ohm termination.

| Channel Type | Medium | Bit Rate | Characteristics | Notes |
|---|---|---|---|---|
| TP/XF-1250 | Twisted pair cabling | 1250kbps | Transformer-coupled | Bus topology |
| TP/XF-78 | Twisted pair cabling | 78kbps | Transformer-coupled | Bus topology |
| TP-RS485-39 | Twisted pair cabling | 39kbps | EIA RS-485 specifications | Bus topology |
| TP/FT-10 | Twisted pair cabling | 78 kbps | Free topology with optional link power | Based on ANSI/EIA/ CEA 709.3 |

- TP/XF-10 (78Kbps, transformer coupling) -- description & protection circuits
  https://www.echelon.com/assets/bltf34c104ee95ed4b5/005-0166-01A.pdf




### 8.5.6 LonWorks Microprocessor & Transceiver Products

- Microprocessor with Built-in LonWorks Interface, $4, Model 6050, #14550R-500.
  https://www.echelon.com/assets/blt4cab5e1757e2cb21/Neuron_6050_Datasheet.pdf
- Microprocessor + Transceiver module for TP/XF-10 (78Kbps, transformer coupling, 64KB flash), #55040R-10, $27/qty1.
  https://www.echelon.com/assets/blt896253f1424397be/Control-Point-Module-5000-datasheet.pdf
  https://www.digikey.com/products/en?mpart=55040R-10&v=1499
- Transceiver module for TP/XF-10 (78Kbps, transformer coupling), #55010R-10, $27/qty1.
  https://www.digikey.com/product-detail/en/echelon-corporation/50010R-10/1499-1064-ND/7311892
  http://downloads.echelon.com/support/documentation/datashts/50010_20.pdf
- Transceiver Module, #LPT11, connect to 78Kbps line powered network, $27.
  https://www.echelon.com/assets/blt9d2201bb38df9a45/LPT11-Link-Power-Twisted_Pair-Transceiver-Model-50040-02-datasheet.pdf
- Microprocessor + MAC + PHY, #FT 5000 (connects to transformer, 78Kbps, FT or LP), $9/1K quantity.
  https://www.echelon.com/assets/blt8a0c31cdc92c7291/FT-5000-smart-transceiver-datasheet.pdf
  https://www.digikey.com/catalog/en/partgroup/ft-5000-smart-transceivers/52903
- #FT 6000 -- similar to above yet newer, $8/1K quantity.
  https://www.echelon.com/assets/blte8a01dc38bea0c74/FT_6000_Datasheet.pdf

- Transformers, $1.30/100, #FT-X…
  https://www.digikey.com/products/en/transformers/specialty-transformers/165?v=1499,1500
  https://www.echelon.com/assets/blt8a0c31cdc92c7291/FT-5000-smart-transceiver-datasheet.pdf
- Echelon products sold at Digikey.com
  https://www.digikey.com/en/supplier-centers/e/echelon

### 8.5.7 LonWorks Advantages/Disadvantages

- Advantages: Existing interface, many products, 78Kbps is decent speed
- Disadvantages: TP/xx-10 maximum stub is 3m (mostly daisy-chain, not fully tree topology due to desire to control impedance with termination), driving 50Ω requires power (dozens of mA), very costly when purchasing Echelon parts, $1.30 transformer is costly/bulky relative to two opto-isolators ($0.24*2=$0.48 w/ Schmidt trigger).

### 8.5.8 Suggestions for Improvements

- Utilize physical layer RS-486P instead of RS-485 (lower power, stub length > 3m). Specify integration during measurement (32bit, more resolution)? Declare measurement accuracy (via more resolution)?
- Software improvements to fix above listed "disadvantages".
- LonWorks might block devices that attach to their TP/XF-78 network without using their components. However, if one has an RS-486P or DALI 3P network with non-LonWorks devices, then the master controller in software could wrap those devices to look like LonWorks devices, and attach to the LON system, perhaps.

## 8.6 EnOcean

### 8.6.1 EnOcean Overview

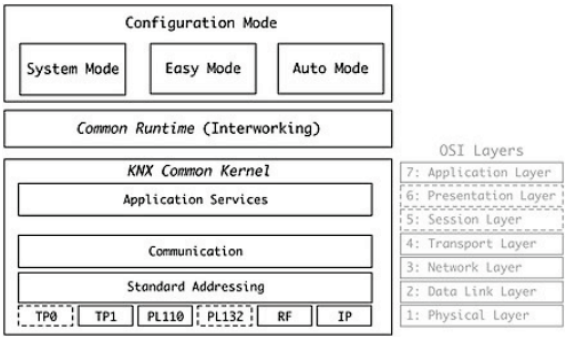
- Features: These are very low power wireless devices that harvest energy from wherever they can get it (e.g. solar PV). Reliability is low.
- EnOcean Overview, Wikipedia
  https://en.wikipedia.org/wiki/EnOcean
- EnOcean, 3pg summary, see pdf page#39
  https://www.iea.lth.se/publications/MS-Theses/Full%20document/5267_full_document.pdf
- EnOcean Application Notes
  https://www.enocean.com/en/support/application-notes/

### 8.6.2 EnOcean Reference Designs

- EnOcean Reference Designs and Software (search this page for "software")
  https://www.enocean.com/en/support/application-notes/

## 8.7 KNX

### 8.7.1 KNX Overview




- KNX Standard, Wiki
  https://en.wikipedia.org/wiki/KNX_(standard)
- KNX, 4pg Summary, see pdf page#28
  https://www.iea.lth.se/publications/MS-Theses/Full%20document/5267_full_document.pdf
- What is KNX?
  https://www2.knx.org/no/knx/association/what-is-knx/index.php
- Overview of KNX Specification (35pgs)
  https://paginas.fe.up.pt/~ee07367/wordpress/wp-content/uploads/2013/03/KNX%20System%20Architecture.pdf
- KNX Association sells terrific books via Amazon
  https://www.amazon.com/s?i=digital-text&rh=p_27%3AKNX+Association&s=relevancerank&text=KNX+Association&ref=dp_byline_sr_ebooks_1
  - "KNX Basic Course Documentation", https://www.amazon.com/dp/B00XHVSB82/ref=cm_sw_em_r_mt_dp_U_Cp6HDbXZW3YRM
  - "KNX Advance Course Documentation", https://www.amazon.com/dp/B00XHXJZBC/ref=cm_sw_em_r_mt_dp_U_Zq6HDbW9F2KQP
  - "KNX Handbook for Home and Building Control", https://www.amazon.com/dp/B00XC0L4P0/ref=cm_sw_em_r_mt_dp_U_yK6HDb17VJEE0
    >> gsw bought several of these for his kindle
- Active Physical Window Support: Simple "move up" / "move down" blind commands

### 8.7.2 Summary

- 16bit M.L.D address: **M**ain Line (0..15, 4bits), **L**ine (0..15, 4bits), **D**evice (0..255, 8bits)
- Data packet: source address, destination address, 1 to 16 bytes of data

### 8.7.3 KNX Reference Designs

- Calimero KNX Server: https://github.com/calimero-project/calimero-server
- KNX Thermostat Reference Design, Texas Instruments (TI): http://www.ti.com/lit/ug/tidub27a/tidub27a.pdf
- KNX NCN5120 Reference Design, On Semiconductor: https://www.onsemi.com/pub/Collateral/EVBUM2169-D.PDF

## 8.8 ZigBee

### 8.8.1 ZigBee Overview

- ZigBee Overview, Wikipedia
  https://en.wikipedia.org/wiki/EnOcean
- ZigBee, 3pg summary, see pdf page#43
  https://www.iea.lth.se/publications/MS-Theses/Full%20document/5267_full_document.pdf
- Devices based on Bluetooth wireless standard. Reliability is low.
  https://en.wikipedia.org/wiki/Zigbee
- ZigBee Cluster Library User Guide, 2015, 800 pages, good
  https://www.nxp.com/docs/en/user-guide/JN-UG-3077.pdf

- Active Physical Window Support: Simple "temperature measurement" / "level control" commands

### 8.8.2 Zigbee Reference Designs

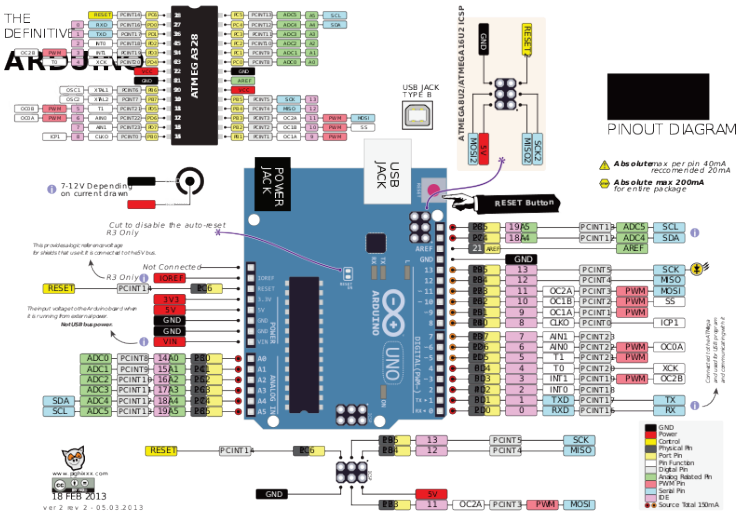
- Silicon Labs Zigbee Reference Designs
  https://www.silabs.com/support/getting-started/mesh-networking/zigbee
  https://www.silabs.com/products/development-tools/wireless/mesh-networking
  https://www.silabs.com/products/development-tools/software/mesh-networking-software-tools
  https://www.silabs.com/products/development-tools/wireless/mesh-networking#dev-kits
  http://www.wless.ru/files/ZigBee/ConnectedHome/EthGW/ug129-zigbee-gateway-ref-design-guide.pdf
- TI Zigbee Reference Designs
  http://www.ti.com/tool/Z-STACK

# 9 REFERENCE: Development Hardware

## 9.1 Arduino References

### 9.1.1 Overview

- Arduino, Wikipedia
  https://en.wikipedia.org/wiki/Arduino




### 9.1.2 Arduino Uno and Nano Processor Boards

- Arduino Processor Boards
  https://store.arduino.cc/usa/arduino/boards-modules

### 9.1.3 Arduino Uno Shields

- Arduino Store's Shields
  https://store.arduino.cc/usa/arduino/shields
- Arduino Uno Shields
  - 4 relays: https://store.arduino.cc/usa/4-relays-shield
  - Motor Driver: https://store.arduino.cc/usa/arduino-motor-shield-rev3

### 9.1.4 Arduino Products

- Digikey, Product Selector, Arduino (good)
  https://www.digikey.com/en/maker/search-results?y=bfd623107788471f9319d4735e2eadb8&t=094a789bf5a14c83a4a4deb90b8b5bd7&g=newest&page=1
- List of Arduino Boards
  https://en.wikipedia.org/wiki/List_of_Arduino_boards_and_compatible_systems

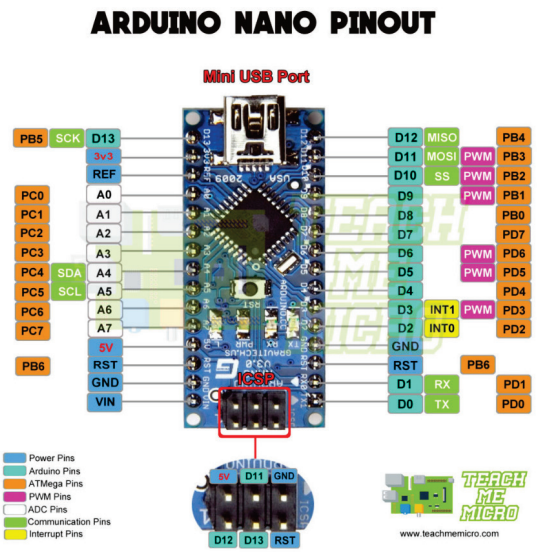
### 9.1.5 Arduino Projects

- List of 300 or so Arduino Projects
  https://playground.arduino.cc/Main/SimilarBoards/
- Digikey, Articles on Projects ("Blogs") (good)
  https://www.digikey.com/en/maker/search-results?y=cb5252a72f0549558ffaaa2a80d3a1ed&t=094a789bf5a14c83a4a4deb90b8b5bd7&g=newest&page=1

### 9.1.6 Arduino Uno R3 Pinout

- Arduino Uno R3 Pinout, Explained (good)
  https://www.circuito.io/blog/arduino-uno-pinout/

## 9.2 Arduino Nano References (32pin dip)



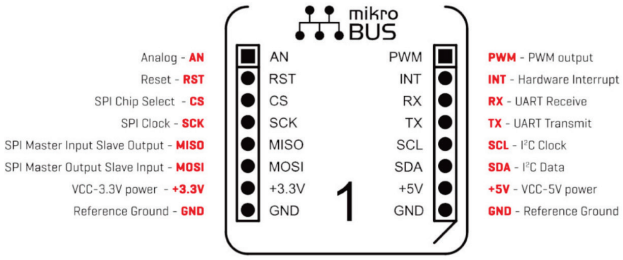

### 9.2.1 Overview

- Arduino Nano Pinout
  https://www.teachmemicro.com/arduino-nano-pinout-diagram/
- Official Arduino Nano Product
  Product: https://store.arduino.cc/usa/arduino-nano
  Pin Description: https://store.arduino.cc/usa/arduino-nano
  User's Manual: https://www.arduino.cc/en/uploads/Main/ArduinoNanoManual23.pdf
- Nano Adaptor that puts 2 Nano's onto one Arduino Uno Shield
  https://www.mikroe.com/arduino-uno-click-shield
- List of official Arduino Nano shields
  https://store.arduino.cc/usa/arduino/shields
- Arduino Nano shield with sensors (e.g. temperature)
  https://store.arduino.cc/usa/mkr-env-shield
- Arduino Nano shield with solid state relays
  https://store.arduino.cc/usa/mkr-relay-proto-shield

## 9.3 Mikroe Company

### 9.3.1 Mikroe Overview

- Mikroe Board Specifications
  https://download.mikroe.com/documents/standards/mikrobus/mikrobus-standard-specification-v200.pdf
- Mikroe has 500 small pcb's called Click boards (not the same as Arduino Nano 32pin)
  - Brochure: https://download.mikroe.com/documents/brochure/click-boards-brochure-2019-web-2.pdf
  - Products: https://www.mikroe.com/click
  - Connector: https://www.mikroe.com/blog/mikrosdk-board-definition-files
- Mikroe has C source code that supports their Click boards. To open their mpkg files to see their C source code, I think you need to first buy their $250 compiler (?)
  https://libstock.mikroe.com/




### 9.3.2 Mikroe Products

- Mikroe has adaptors that connect Arduino to Click boards.
  - https://www.mikroe.com/arduino-uno-click-shield
  - https://www.mikroe.com/flipclick-pic32mz
- Processor Boards, "Clicker", processor PCB with one Click socket
  https://www.mikroe.com/starter-boards/clicker

- Processor Boards, "Clicker 2", processor PCB with multiple Click sockets
  https://www.mikroe.com/starter-boards/clicker-2
- Processor Boards, "MINI", processor PCB fits into DIP40 socket
  https://www.mikroe.com/starter-boards/mini
- Processor Boards, "MicroBoard", processor PCB fits into DIP80 socket
  https://www.mikroe.com/starter-boards/mini

## 9.4 LED On/Off/Dimmer Control

### 9.4.1 LED Driver Click Control Boards (not Arduino Nano, search "LED Driver" for more details)

- There are many Mikroe Click LED control boards:
  https://www.mikroe.com/click/display-and-led Selection tool
- There are multiple Click boards that do LED control: "LED Driver Click", "LED Driver 2 Click", "LED Driver 3 Click", "LED Driver 4 Click", "LED Driver 5 Click".
- Example Click board that does LED control (#TPS61160ADRVT current source)
  https://www.mikroe.com/led-driver-4-click MIKRO-3037, "LED Driver 4 Click"
- Page 22 to 24 of the following brochure summarizes LED Driver options.
  Xmc1300/1400 µProcessors can implement Buck Converter with 110/220VAC input.
  https://www.infineon.com/dgdl/Infineon-Power_and_Sensing_Selection_Guide_2018-SG-v00_00-EN.pdf?fileId=5546d4625607bd13015621522aa012cb
- There are several options with LED driving:
  - Buck converter (PWM, current sense) converts 7…50VDC to constant current (e.g. 25V, 300mA), and this is modulated by PWM to implement dimming.
  - Buck converter converts 110/220VAC to constant current, no transformer.
  - Fly-back converter transforms 110/220VAC to constant current via transformer.
- The following books are helpful:
  - *The Art Of Electronics*, *By Horowitz* "9.6 Switching Regulators"
    "9.7 AC-Line Powered ("offline") switching converters
    "Figure 12.63, PWM dimming with 12VAC"
  - *Fundamentals of Solid State Lighting, Khanna* "21.6 Buck Converter for LED's"
    "Ch22 AC Driving Circuits (e.g. 110/220VAC)
- Digital Control Using PWM

  https://www.infineon.com/dgdl/Infineon-Whitepaper_lighting_ICs_XDP_digital_power_-_dimming_control_using_a_PWM_signal-WP-v01_00-EN.pdf?fileId=5546d462689a790c016910d26f083f2e
- Search this document for "LED Driver" for more information.

## 9.5 Motor Control

- There are many Mikroe Click boards that control motors: "Click Brushless", "Click DC Motor", "Click Fan", "Click PWM Driver", "Click Stepper", "Click H-Bridge" products
  https://www.mikroe.com/click/motor-control Selection tool
- Example Click board that does motor control:
  https://www.mikroe.com/brushless-2-click #MIKRO-2754, brushless motor,
  https://www.mikroe.com/brushless #MIKRO-2441, brushless motor,
  https://libstock.mikroe.com/projects/view/1951/brushless-click source code
- For more information, search this file for "Motor Control".
- Infineon CCU8 module helps with motor control
  https://www.infineon.com/dgdl/Infineon-CCU8-XMC1000_XMC4000-AP32288-AN-v01_01-EN.pdf?fileId=5546d4624e765da5014ed8dd5c7d1730
- Scroll to bottom of webpage to see multiple articles on Motor Control:
  https://www.edn.com/mosfet-drivers-for-motor-drives-start-with-your-motors-specs/

## 9.6 Schematic Capture and Simulation Software

- Schematic Capture and Simulation Software, free, online, by Michael Robbins (MIT EE) author of Ultimate Electronics
  https://www.circuitlab.com/