

EtherNet/IP™ to the Edge – A Concept for "Low-complexity Ethernet"

Dave Alsup
Chief Systems Architect
Innovasic, Inc.

Tom Weingartner
Product Marketing Manager
Innovasic, Inc.

Presented at the ODVA
2017 Industry Conference & 18th Annual Meeting
February 21-23, 2017
Palm Harbor, Florida, USA

Abstract

Simple field devices, such as sensors and actuators, have long resisted incorporating Ethernet as a fieldbus interface. The reasons for this reluctance are plenty, but from a device integration viewpoint, the limiting factors have been the size, power, and cost of the Ethernet interface itself. Many advancements in communication technology have taken place over the past few years that have changed the landscape for Ethernet to address these limitations. This paper defines the concept of "Low-complexity Ethernet" and describes how this concept can be used to bring reliable EtherNet/IP communication to edge devices like sensors and actuators. The paper will also identify future directions that could enable brownfield installations to take advantage of bringing EtherNet/IP to the edge.

Keywords

EtherNet/IP, Industrial Ethernet, Ethernet Interface, Fieldbus, Field Device, Sensors, Actuators, Edge Devices, IIoT,

Background

This paper is primarily concerned with novel approaches to reduce the hardware cost and complexity of industrial Ethernet nodes. While it does not specifically address the reduction of software complexity in terms of the protocols and modes that have to be supported, this is also an important line of research. For a good example of this sort of thinking see, "Extending EtherNet/IP to Resource-Constrained Industrial Things," by Brooks and Xu¹.

Why isn't Ethernet at the Edge Today?

Looking at the traditional hierarchy of automation systems and comparing it with processing power, it is easy to see that processing power scales up from simple device all the way through Programmable Logic Controllers (PLCs), Human-Machine Interfaces (HMIs), etc. It makes obvious sense not to use an ARM A9 down in a simple temperature transmitter from both a technical complexity standpoint as well as device cost standpoint. The same scaling should hold true for complexity of Ethernet in these automation systems – as we go from PLCs down to simple sensors so too should the size, cost, and power of the Ethernet interface. In PLCs there are a large number of connections that need to handle all of the messages for all of the devices. These messages may need to be handled across multiple cycle times. There may also be a need to manage the topology along with the normal network management duties. On the other hand, a simple device has only one connection with small messages and a simple data refresh. It is a topology participant in the network and its network management duties are minimal. The design guide, Ethernet Design Considerations, Chapter 5 by Allen Bradley provides an excellent resource for additional practical Ethernet design information².

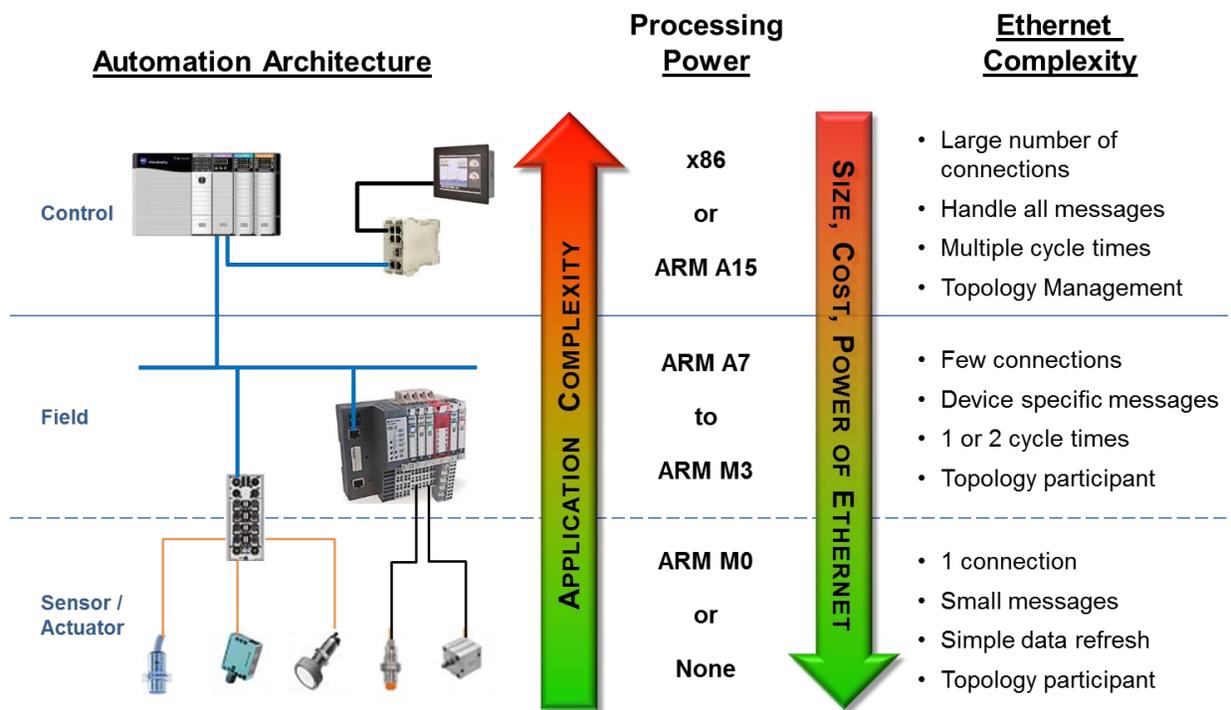


Figure 1: Function vs. Capability

So why not scale the complexity of the Ethernet interface down for simple devices like we do for the processing power? If we can reduce the size, power, and cost of the Ethernet Interface we can bring Ethernet all the way down to the edge. In the next section we take a look at the architecture of an Ethernet node and then take a look at how we can scale it.

Architecture of an Ethernet Node

An Ethernet node consists of both a hardware architecture and a software architecture. On the hardware side, the architecture can either be a processor-MAC-PHY or a processor-MAC-switch-MAC-PHY in the case of multi-port devices. These architectures are shown in the figure below. No matter what processor is used, it will be paired with an Ethernet MAC. And in many cases, the PHY is not integrated on-chip due to power dissipation, chip die area, and cost considerations. Also, PHY performance in real-time applications is critical, so leaving the PHY separate allows the designer to choose the right PHY for the application. So in all cases where the PHY is not integrated with the MAC, the interface to the PHY is MII (IEEE 802.3³) or RMII (RMII Specification⁴) for 10/100BASE-TX (10/100 Mb/s) Ethernet and GMII (IEEE 802.3) or RGMII (IEEE 802.3) for 1000BASE-TX (1 Gb/s) Ethernet. These interfaces involve many high-speed signals consuming numerous pins and additional board space due to layout considerations. The large number of pins required by an MII interface also makes the interface difficult to isolate from the remainder of the design (power, noise, etc.).

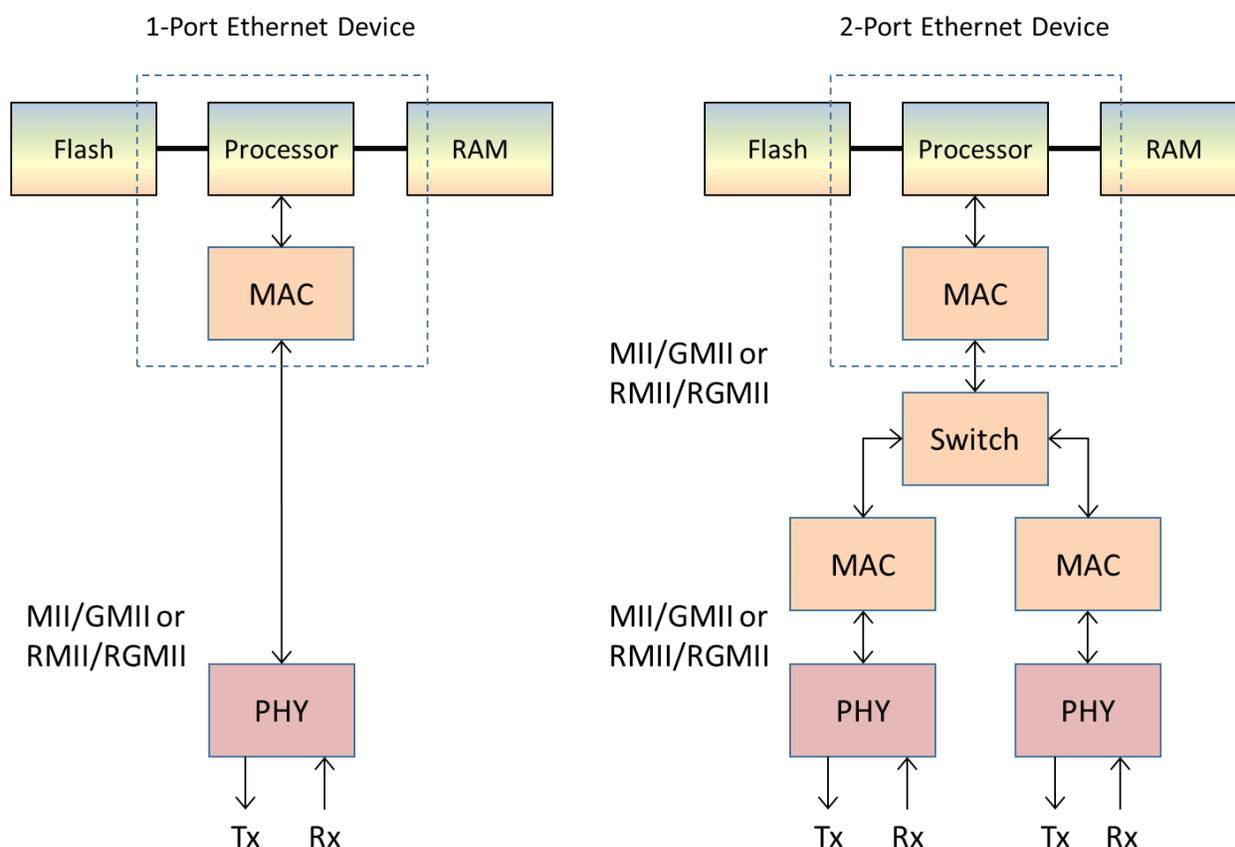


Figure 2: Components and Connections of an Ethernet Device

The hardware architecture must also perform a trade-off between processor performance and memory for queuing Ethernet frames. A simple processor with MAC will often have a simple FIFO for reception and transmission of the frames. Typically such a FIFO will have room for only 1 to 10 frames, depending on the frame size. A simple device must process these frames very rapidly, even if only in short bursts, or

the frames will be dropped because the FIFO is full. This means the hardware designer must either choose a faster processor in order to process every Ethernet frame or choose a larger memory so Ethernet frames are not lost. Therefore, it is important for the hardware designer to understand all of the protocol requirements for the Ethernet interface.

Most MACs have filter logic to try to reduce the number for Ethernet frames, the processor must still service every Ethernet frame that makes it through the filter logic. Because these MAC filters typically only have general filtering capabilities, it puts a heavier burden on the processor and increases the bandwidth reading, evaluating, and discarding frames not critical to the current operating state of the node. Also, as the number of protocols that must be supported for compliance with a particular profile increases, the Flash and RAM sizes of the processor grow as complexity increases. In fact, it is nearly impossible for the hardware designer to guess which protocols will be used, so Flash and RAM sizes are kept large to ensure there is enough room to accommodate all of the protocols. In most cases, on-chip Flash and RAM are not enough to accommodate the protocols. And, of course, this causes the power and cost of the node to increase.

On the software side, the architecture follows the OSI model or TCP/IP model. But in any case, the software will utilize protocols contained in a TCP/IP stack plus an Industrial Ethernet stack which may include some type of PTP stack depending on time synchronization requirements. This software architecture is shown in Figure 3.

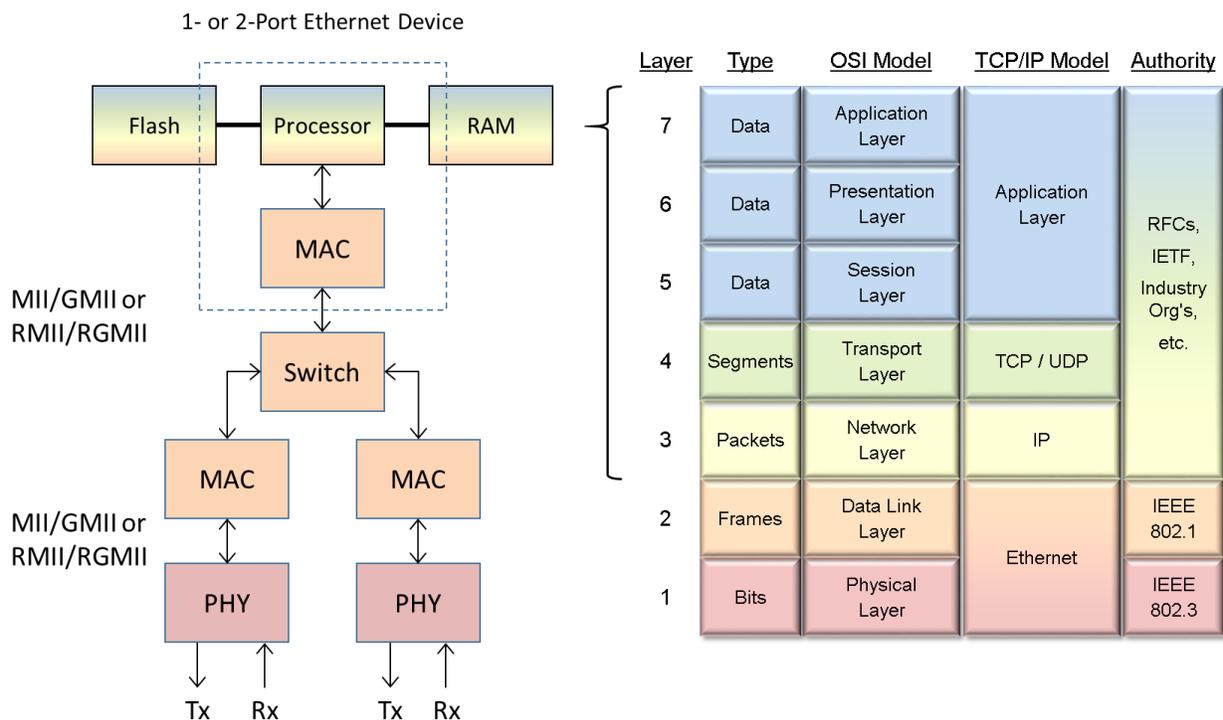


Figure 3: Physical Layers vs. Logical Layers

At the upper layers of the OSI model or Application layer of the TCP/IP layer, there is the I/O application, calibration functions, history functions, filtering, among others. Also

at this layer are the protocols used to communicate with applications on a PLC, configuration application, etc. These include EtherNet/IP, HTTP SNMP, PTP and others.

The Application Layer elements utilize the Transport Layer services of a TCP/IP stack, typically through a sockets interface. Some Application Layer functions may communicate directly to the Data Link Layer (LLDP, RSTP, PTP, etc.). The sockets layer is also where security is often managed using SSL/TLS protocols. Much of this traffic is next passed through TCP or UDP and down to IP in the TCP/IP stack. Typically the same stack that manages IP, TCP/UDP also handles a host of other protocols that are necessary for the establishment and maintenance of layer 3 communications. These include ARP, ICMP, DHCP, BOOTP and others⁵.

Finally, driver software interfaces these software elements to the Ethernet MAC hardware. The complexity of this software is dependent on the hardware interfaces (FIFO, DMA, etc.) and the various filtering and levels of service that must be handled for different types of frames at different load levels and device application states.

This variety of Ethernet traffic imposes multiple burdens on the hardware system. This burden is illustrated in Figure 3 by the “rainbow effect” of colors in the processor, Flash, and RAM – meaning these hardware elements are required to support layer 2 processing (orange) all the way through layer 7 processing (blue). Simple filtering at the MAC Address level does little to differentiate these forms of traffic. Differentiating frames in driver interrupt handlers imposes bandwidth overhead that must be executed at a higher priority than most other communication processing, delaying critical operations with filtering and organizing low priority frames. It also imposes buffering issues because all of these processes are not coordinated in time on the network and can cause sudden overload conditions on a node/network when they happen to generate traffic simultaneously. The typical solution to this problem is to increase buffer space and processor bandwidth. This increases device cost and complexity.

Scaling an Ethernet Node

To reduce the cost, size, power, and complexity of hardware in an EtherNet/IP edge device, the focus will be on reducing the following features:

- Target small-scale single-chip processing solution by reducing
 - Processor speed/performance
 - Flash memory size
 - RAM size
- Reduce interconnect complexity from processor to network interface
- Reduce pin-count and complexity of network interface

The addition of an Advanced MAC to the network interface (PHY/switch chip) accomplishes most of the size reduction goals. The Advanced MAC performs intelligent/dynamic frame filtering and buffering before any frames are communicated to the processor chip. This filtering manages priorities among protocols, and surges in frame receipt due to alignment of various protocols. The intelligent filtering reduces overall buffer space (frames are retained based on priority, application state, and

processor load conditions) and substantially reduces the amount of data that has to be transferred to and processed by the application processor under high load conditions.

The reduction in frame communications to the processor allows a simpler interface to be used for that communication. SPI provides a well understood and common interface with low pin count, frequency based on the capabilities of the processor chip, and ease of electrical isolation between the application and the network communications. The SPI interface and frequency is unchanged whether the device is communicating at 10 Mbit, 100 Mbit, or 1000 Mbit, just as the application communication requirements of an edge device are not changed by the network bandwidth. The SPI interface allows for very low power and low frequency processors that cannot manage the load necessary to manage a standard MII interface. The frequency and timing of the interface can be managed to avoid noise issues in critical application interfaces, whereas a MAC receive operation is asynchronous to the application operation.

Additional features can be added to the advanced MAC for common and well understood functions such as synchronization (e.g. IEEE802.1AS), further reducing processor frame processing and RAM/Flash requirements. Functions associated with security (key generation and management, hash generation/checking, encryption and decryption) can also be incorporated into the Advanced MAC, although this requires work to select the appropriate approaches for the data and security characteristics of edge devices as opposed to standard SSL/TLS.

Further reduction of Flash requirements (and the associated elimination of external ROM for the processor and reduction of cost/complexity of the edge device) requires elimination and/or simplification of protocols that are required for more complex devices.

Many protocols that make sense for a full-purpose node may be too expensive for a small node. For example, LLDP Transmitter is a simple protocol easily managed in a constrained device. LLDP receiver is somewhat more complex in itself, but still not generally too difficult for a node with only 1 or two Ethernet ports. The complexity comes with support for protocols like SNMP that are used to query LLDP receivers. Single-port devices can benefit from being connected to infrastructure switches that support LLDP receive function, thus the single-port low cost node only needs to be an LLDP transmitter. For two-port devices in a line topology, another solution is necessary.

Network management protocols (RSTP, etc.) are another burden for small nodes and can typically be eliminated from two-port devices as long as the rules for setting up a network are carefully worked out. Specific work to develop widely accepted protocols for line topologies is needed. There is much opportunity for adapting existing protocols (LLDP, RSTP, etc.) in such a way that they can be implemented with much lower overhead in a line topology.

Other areas that can be excluded in general include security (SSL/TLS), device management (DHCP, BOOTP, ICMP), application interfaces (Berkeley sockets), etc. Each of these reduces memory usage by some small amount and takes with it some convenience in system configuration and management.

The answer for EtherNet/IP to the edge lies somewhere between a current full-up device and a hardware-only device. From a protocol development perspective

- Select only the protocols relevant to the application
- Limit the message sizes
- Limit message traffic

While some advocate only a limited set of protocols to reduce footprint, it would be better to select from a wide range of protocols only those that are required by the application. In other words, if an application doesn't need to support, say, ICMP, then it should be optimized out of the TCP/IP stack.

From a standards development perspective

- Develop robust Ethernet physical layer implementations for extended range (200m, 1000m) with reduced cost and complexity of cabling (e.g. single unshielded twisted pair).
- Develop stripped down topology discovery and management protocols suitable to small nodes with one or two ports
- Develop security approaches appropriate to low-latency and small footprint devices.

From a hardware development perspective

- reduce pin counts of devices
- provide low-cost low power solutions for single-port and dual-port network interconnects including Ethernet PHY, magnetics, ESD protection, electrical isolation, ...

By advancing the capabilities of an Ethernet MAC and an Ethernet Switch, this allows the processor, Flash, and RAM requirements to be significantly reduced. It also allow for a "reallocation" of the Ethernet function out of the processor and into the PHY. The previous figure showing the architecture of an Ethernet communication system now becomes more simplified with an isolated SPI interface to processor. And while we still have the "rainbow effect" of layer 2 through 7 processing, the number and types of protocols needing to be handled by the processor are greatly reduced. The next section illustrates how this reallocation and protocol simplification can be applied to a space, power, and area constrained device used in automation systems.

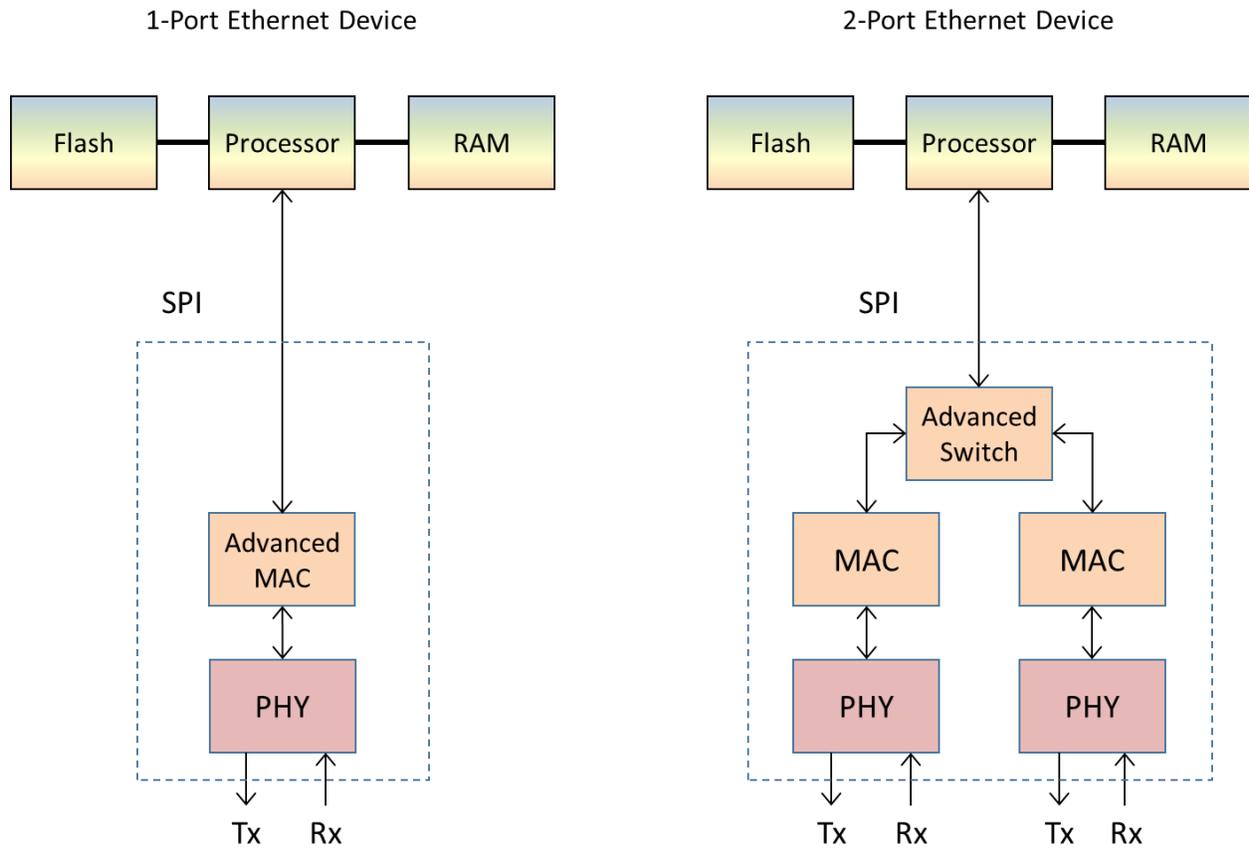


Figure 4: 1 Port and 2 Port Devices

Example of a Low-complexity Ethernet Node

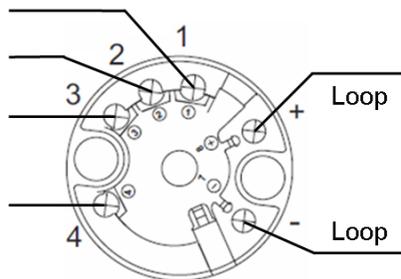
Now that we have discussed the typical architecture of an Ethernet node and the considerations for scaling it to reduce power, area, and cost, this section puts it all together by taking a look at an example. Perhaps one of the most area and power constrained devices in automation systems is the Temperature Transmitter. This device takes signals from a temperature probe and converts the probe's information into a 4-20mA signal to transmit temperature to the automation system. HART can also be overlaid onto the 4-20mA signal in order to control set points or change calibration variable as well as receive diagnostic information. The temperature transmitter could also do this via Ethernet, but there are issues putting Ethernet in such a constrained environment. There are also issues related to the desire to use brownfield cabling, but this is not the subject of this paper. Other groups such as the 10SPE Task Group in IEEE or the APL Group in Germany are addressing this issue.

Assuming there will be a means to get Ethernet to a temperature transmitter, we still need a low power, reduced area, cost effective way to put Ethernet inside the transmitter. The figure below compares how a temperature transmitter communicates today with 4-20mA technology and how these requirements look like using Ethernet.

4..20 mA with HART

Connections for:

- 2-wire Potentiometer / RTD
- 3-wire Potentiometer / RTD
- 4-wire Potentiometer / RTD
- Voltage Measurement
- Thermocouple



4..20 mA

- Process Variable (Temperature)

HART

- Calibration
- Alarms
- Set Points

Supply Power: 12..42V

Power Consumption: < 1.5W

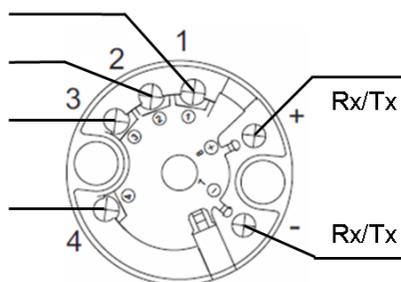
Cable length: up to 3km

Bandwidth: 1200 Baud

Ethernet with future 2-wire

Connections for:

- 2-wire Potentiometer / RTD
- 3-wire Potentiometer / RTD
- 4-wire Potentiometer / RTD
- Voltage Measurement
- Thermocouple



Ethernet

- Process Variable (Temperature)

Ethernet

- Calibration
- Alarms
- Set Points

Supply Power: 12..42V

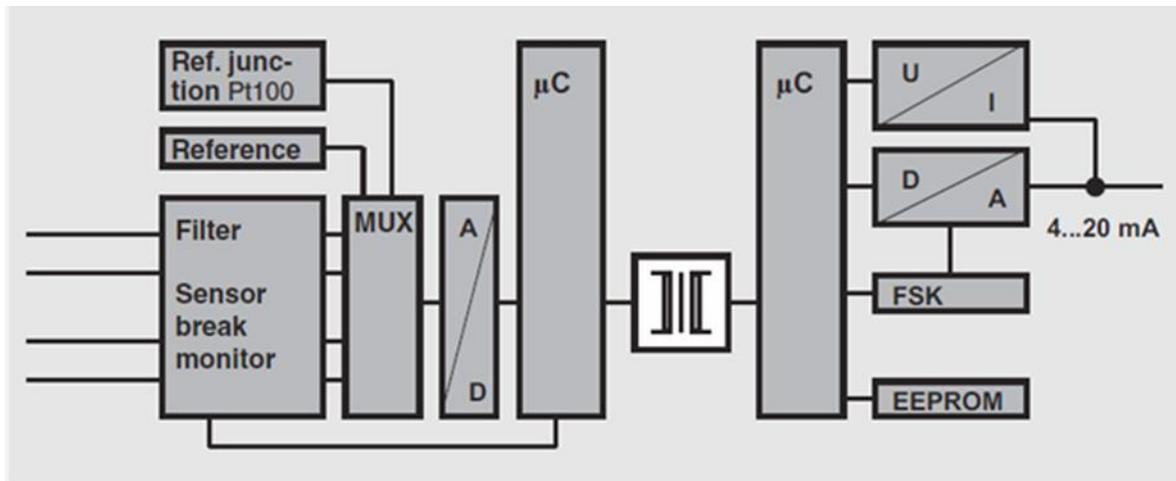
Power Consumption: < 1.5W

Cable length: 200m..1km

Bandwidth: 100Mb/s..10Mb/s

Figure 5: Replacing HART with Ethernet

The architecture of a temperature transmitter currently uses a microcontroller for temperature calibration, control, and diagnostics and a microcontroller for communication as shown in the figure 6. In the top half of the figure, the microcontroller on the communication side interfaces to the 4.20mA interface through a Digital-to-Analog Converter (DAC). If HART is used, a HART modem is also connected to the microcontroller and DAC. The microcontroller used for communication is also connected to the microcontroller performing the temperature calibration, control, and diagnostics. This communication path is done through isolation in order to keep the two sides electrically independent.



(Source: ABB, "Industrial Temperature Measurement Basics and Practice")

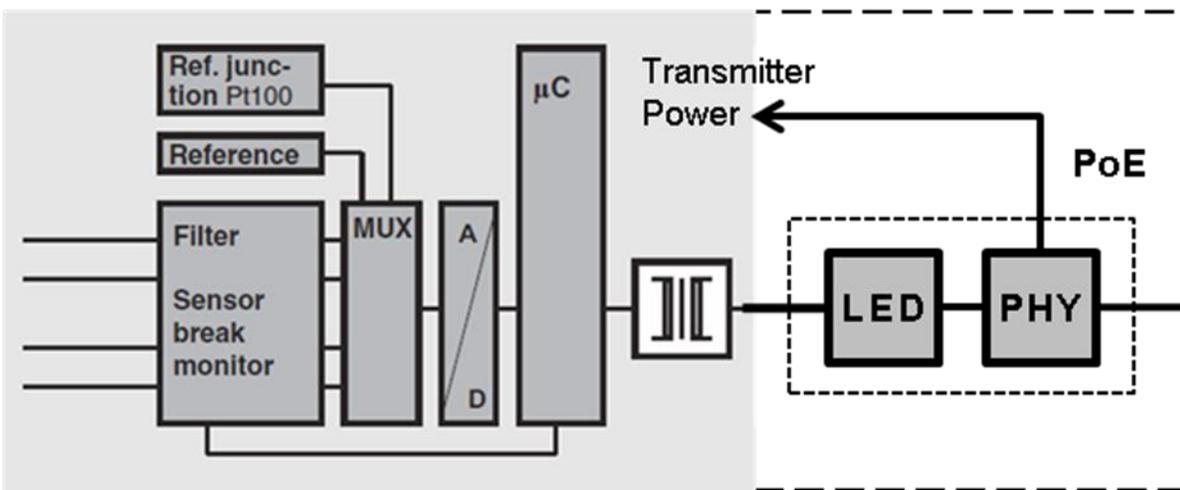


Figure 6: Reducing Isolation Complexity

If we scale Ethernet in the manner described in the section above, we can replace the microcontroller used for communication, the DAC, the HART modem, and EEPROM with a Low-complexity Ethernet Device (LED) with PHY. This can either be a one-port or a 2-port device depending on the topology of the Ethernet network. And since the LED interface to the main microcontroller is SPI, this is a widely used interface that can be easily isolated. Given the low-complexity of the temperature transmitter communication variables, it is possible to scale the EtherNet/IP communication down to 1 implicit message and 3 explicit messages. And by using a space optimized version of the EtherNet/IP stack along with a minimum TCP/IP stack implementation executing on the microcontroller, it is possible to have a software footprint for the communication software on the order of 128Kbytes of Flash and 64 Kbytes of RAM.

Summary

The concept of low-complexity Ethernet holds the promise of providing cost effective, low power, reduced area connectivity for simple EtherNet/IP devices. Such a concept

can be realized in an open manner coupled with advanced features in next generation Ethernet MACs and Ethernet switches. By scaling a device's communication software as described in this paper, it is possible to fulfill the connectivity requirements of EtherNet/IP systems while minimizing the software footprint. This scaling, in turn, reduces the Flash and RAM hardware requirements to the point where a single chip processor with memory can be utilized in the design of a field device.

Further, by taking advantage of advanced features in next generation Ethernet MACs and Ethernet Switches, it is possible to reallocate the Ethernet function from the processor into the PHY thereby creating a Low-complexity Ethernet Device. Such a device provides the processor with a simple SPI interface to the Ethernet network. Not only does this SPI interface have the advantage that it is easier to electrically isolate from the other circuitry in the design, but it also eases the processing requirements since the controller no longer has to process every Ethernet message from the network. Rather than needing an A-class ARM processor to execute application and network software, a single chip processor with memory can be a simple, low-cost M-class controller. It is through this combination of hardware partitioning and software tailoring that we can achieve the concept of bringing EtherNet/IP to the edge.

References

1. Extending EtherNet/IP to Resource-Constrained Industrial Things, Dayin Xu, Paul Brooks
https://www.odva.org/Portals/0/Library/Conference/2015_ODVA_Conference_Xu-Brooks_Resource-Constrained-Things.pdf
2. Ethernet Design Considerations, Allen-Bradley™,
http://literature.rockwellautomation.com/idc/groups/literature/documents/rm/enet-rm002_en-p.pdf
3. IEEE 802.3, IEEE Standard for Ethernet, <https://standards.ieee.org>
4. RMII Specification, Rev. A, RMII Consortium 1998,
http://ebook.pldworld.com/eBook/-Telecommunications,Networks-/TCPIP/RMII/rmii_rev12.pdf
5. TCP/IP Guide: A Comprehensive, Illustrated Internet Protocols Reference, Charles M. Koziarck
6. HART information – <https://www.fildcommgroup.org>
7. Industrial Temperature Measurement Basics and Practice, ABB,
https://library.e.abb.com/public/c037f06cf0fe45d1c1257bf600344ae9/03_TEMP_EN_D.pdf

Acronym List

APL – Advanced Physical Layer
ARP – Address Resolution Protocol
BOOTP – Boot Protocol
DAC – Digital-to-Analog Converter
DHCP – Dynamic Host Configuration Protocol
DMA – Direct Memory Access
EEPROM – Electrically Erasable Programmable Read Only Memory
ESD – Electro-Static Discharge
FIFO – First In First Out
GMII – Gigabit Media Independent Interface
HART – Highway Addressable Remote Transducer
HMI – Human Machine Interface
HTTP – Hyper Text Transfer Protocol
ICMP – Internet Control Message Protocol
IEEE – Institute for Electrical and Electronic Engineers
IP – Internet Protocol
LED – Low-complexity Ethernet Device
LLDP – Link Layer Discovery Protocol
MAC – Media Access Controller
MII – Media Independent Interface
PHY – Physical Layer Device
PLC – Programmable Logic Controller
PTP – Precision Time Protocol
RAM – Random Access Memory
RGMII – Reduced Gigabit Media Independent Interface
RMII – Reduced Media Independent Interface
RSTP – Rapid Spanning Tree Protocol
SPI – Serial Peripheral Interface
SNMP – Simple Network Management Protocol
SSL – Secure Socket Layer
TCP – Transmission Control Protocol
TLS – Transport Layer Security
UDP – User Datagram Protocol
10SPE – 10 Megabit Single Pair Ethernet

The ideas, opinions, and recommendations expressed herein are intended to describe concepts of the author(s) for the possible use of ODVA technologies and do not reflect the ideas, opinions, and recommendation of ODVA per se. Because ODVA technologies may be applied in many diverse situations and in conjunction with products and systems from multiple vendors, the reader and those responsible for specifying ODVA networks must determine for themselves the suitability and the suitability of ideas, opinions, and recommendations expressed herein for intended use. Copyright ©2017 ODVA, Inc. All rights reserved. For permission to reproduce excerpts of this material, with appropriate attribution to the author(s), please contact ODVA on: TEL +1 734-975-8840 FAX +1 734-922-0027 EMAIL odva@odva.org WEB www.odva.org. CIP, Common Industrial Protocol, CIP Energy, CIP Motion, CIP Safety, CIP Sync, CIP Security, CompoNet, ControlNet, DeviceNet, and EtherNet/IP are trademarks of ODVA, Inc. All other trademarks are property of their respective owners.