EtherNet/IP™ on TI's Sitara™ processors

Vineet Roy
Software Systems Engineer
Texas Instruments

EtherNet/IP™ (EtherNet/Industrial Protocol) is an industrial automation networking protocol based on the IEEE 802.3 Ethernet standard that has dominated the world of IT networking for the past three decades.

Despite Ethernet's unparalleled success in a wide range of business applications, modifications for industrial applications are necessary because standard Ethernet is not a deterministic protocol and therefore cannot guarantee the realtime operation required by applications such as process control and motor control.

Since it is fully compatible with IEEE 802.3 and the TCP/IP protocol suite, EtherNet/IP can communicate seamlessly with enterprise servers as well as its primary targets – real-time industrial applications. By creating a communications bridge between the factory floor and the enterprise, EtherNet/IP makes it possible to manage production schedules more efficiently, minimize inventory costs and optimize other business oriented functions.

Introduced in 2001, EtherNet/IP is managed by the Open DeviceNet Vendor Association, Inc. (ODVA), which also has responsibility for publishing The EtherNet/IP Specification and coordinating conformance testing.

EtherNet/IP is a member of a family of network protocols that implements the Common Industrial Protocol (CIP) at its upper layers. EtherNet/IP is the name given to CIP when it is implemented over standard Ethernet as defined by IEEE 802.3. Other industrial protocols that utilize CIP include DeviceNet™, ControlNet™ and CompoNet™. **Figure 1** shows the relationship between the four CIP-based protocols and the protocol layers they share, which include connection management, data management services, an object library and a number of use-case profiles.

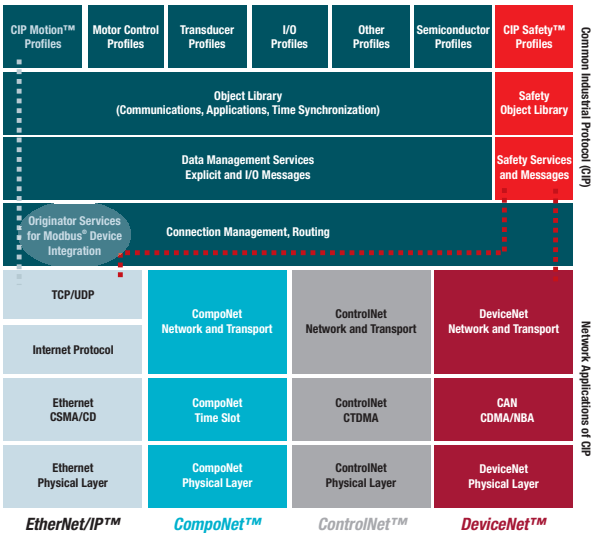

***Figure 1**. DeviceNet, CompoNet & ControlNet share the same CIP application layer with EtherNet/IP.*

EtherNet/IP technology

The CIP that creates a real-time networking environment is a media-independent, connection based, object-oriented protocol that provides a complete set of communication services for factory automation, including control, safety, synchronization, motion, configuration and information. CIP is supported by hundreds of vendors globally, which provides widespread interoperability of devices.

Because of its rigorous conformance programs, CIP offers a unified communication architecture across the manufacturing enterprise. Its most commonly cited benefits are:

- Coherent integration of I/O control, device configuration and data collection.
- Seamless information flow across multiple networks.
- Implementation of multi-layer networks without the cost and complexity of bridges and proxies.
- Minimized investment in system engineering, installation and commissioning.
- Freedom to choose best-of-breed products at competitive prices and low integration cost.

Figure 2 illustrates the value of EtherNet/IP utilizing CIP over standard IEEE 802.3 and the TCP/IP protocol suite to enable a multi-protocol environment. Because EtherNet/IP uses standard Ethernet and TCP/IP technologies, compatibility and coexistence with other applications and protocols is assured. Integration and interoperability are high priorities for EtherNet/IP, which means that more than one path can be taken to implementation.

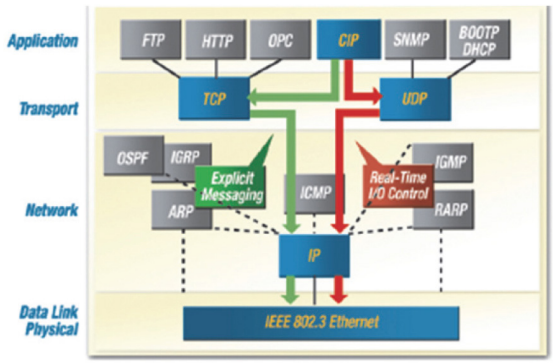

***Figure 2**. Multi-protocol support is possible because CIP is fully compatible with Ethernet and Internet protocols.*

Object-oriented programming model

To simplify the software programming of an application, CIP has adopted an object model in which the CIP application layer defines a set of application objects and device profiles that define common interfaces and behaviors. CIP communication services also enable end-to-end communication between devices on the different CIP networks. To enable multi-vendor interoperability between devices, EtherNet/IP maps CIP communication services to Ethernet and TCP/IP.

Figure 3 on the following page shows how devices are represented using an object model within the CIP application layer. From a functional perspective, three classes of objects are included. Not all of them are required.

- Application objects define a common method for accessing and representing device data.
- Network-specific objects define EtherNet/IP-specific functions and the way in which parameters such as IP addresses are configured.
- Communication objects provide the means to establish communication associations and access device data and services.

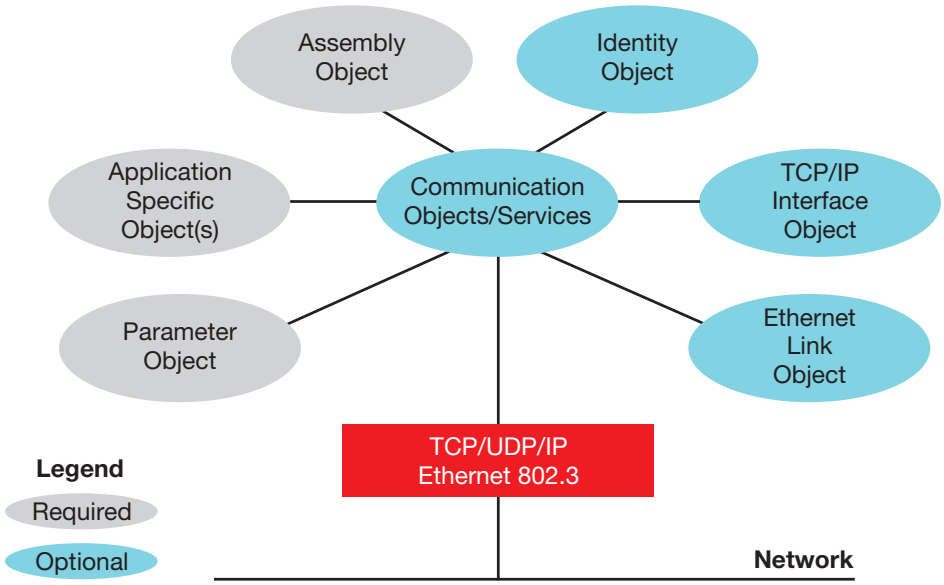

***Figure 3**. In this simplified CIP object model, objects are color coded to indicate whether the object of service is required (grey) or optional (blue).*

Within a device, objects are created by groups of related data and the behaviors associated with the data. CIP requires certain objects to describe a device, how it functions, communicates and its unique identity.

Among the required objects is the Identity Object, which holds information (identity data values) or attributes that include the Vendor ID, Device Type, device serial number and data. CIP does not specify how object data is implemented. It simply sets requirements on what data values or attributes must be made available to other CIP devices. Developers can create other objects that address application-specific and vendor-specific functionalities. Referring again to **Figure 3**, required objects include the Identity Object, the Message Router Object (Ethernet Link Object) and network-specific objects.

Application-specific objects define how data is encapsulated by a device and are specific to the device type and function. An input device, for example, requires an input object with attributes that describe the value and fault status of a particular input point. Vendor-specific objects describe services that are optional and not described in a predefined Device Profile.

The same object model is used to address data within a CIP device. Also consistent with the object-oriented programming paradigm, a set of objects that represent the same type of component constitutes an object class. It is also not uncommon to have multiple copies of the same object in a device, and these are called object instances. Every instance of the object class will have the same set of attributes but a unique set of values. An object instance or an object class has attributes that provide services and implement behaviors.

Types of EtherNet/IP communications

Table 1 is a matrix of the two primary types of communications defined by EtherNet/IP: Explicit and implicit. Although all of the attributes in the matrix are important, they are driven by the Typical Use column, which specifies non-time-critical information, or, real-time I/O data.

Explicit Messaging is primarily a request/reply (or client/server) interaction between devices. It is used for non-real-time data and includes a description of the message's meaning (expressed explicitly). Transmission is less efficient, but very flexible.

CIP Message Type	CIP Communication Relationship	Transport Protocol	Communication Type	Typical Use	Example
Explicit	Connected or unconnected	TCP/IP	Request/reply transactions	Non-time-critical information data	Read/write configuration parameters
Implicit	Connected	UDP/IP	I/O data transfers	Real-time I/O data	Real-time control data from a remote I/O device

***Table 1**. Communication Types.*

It can be used by a human-machine interface (HM) to collect data, or by a device programming tool. Explicit Messaging involves requesting a service – such as a read or write request – of a particular object. For EtherNet/IP, Explicit Messaging uses TCP and can be accomplished with or without previously establishing a CIP connection.

Implicit Messaging is used for time-critical communication. Often referred to as I/O data, implicit messaging implements a real-time data exchange. Implicit messages include very little information about their meaning, so the transmission is more efficient, but less flexible than explicit. An association (a "CIP connection") is established between two devices and the implicit messages are generated according to a predetermined trigger mechanism, typically at a specified packet rate. Both devices agree on data formats (i.e., the format is "implied").

Types of EtherNet/IP devices

Depending on their general behavior and the types of EtherNet/IP communication they support, devices can fall into one of several classifications. Four device types are:

- An explicit message server is the simplest type. These devices respond to requests initiated by explicit message clients. An example of an explicit message server is a bar code reader.
- An explicit message client initiates request/response communications with other devices. Message rates and latency requirements are typically not aggressively real time. Examples include HMI devices, programming tools, or PC- or Linux-based applications that collect data from control devices.
- An I/O adapter receives implicit communication connection requests from an I/O scanner (defined below) and then generates its I/O data at the data rate requested by the I/O scanner. An I/O adapter can be a simple digital input device, or something more complex such as a modular pneumatic valve system. By default it is also an explicit message server. I/O adapters can exchange peer data using explicit messages with any class of device but cannot originate relationships. Examples of I/O adapter type devices:
  - Weigh scales, welders, drives and robots that send and receive real-time data at the request of PLCs and other controllers;
  - Weigh scales, welders, drives and robots that send and receive explicit messages to and from computer interface cards, PLCs and each other; and,
  - HMI products that send or receive explicit or real-time I/O data to/from PLCs.
- An I/O scanner initiates implicit communications with I/O adapters. It deals with issues such as configuration of which connections to make, and how to configure the I/O adapter device. A programmable controller is an example of an I/O scanner.

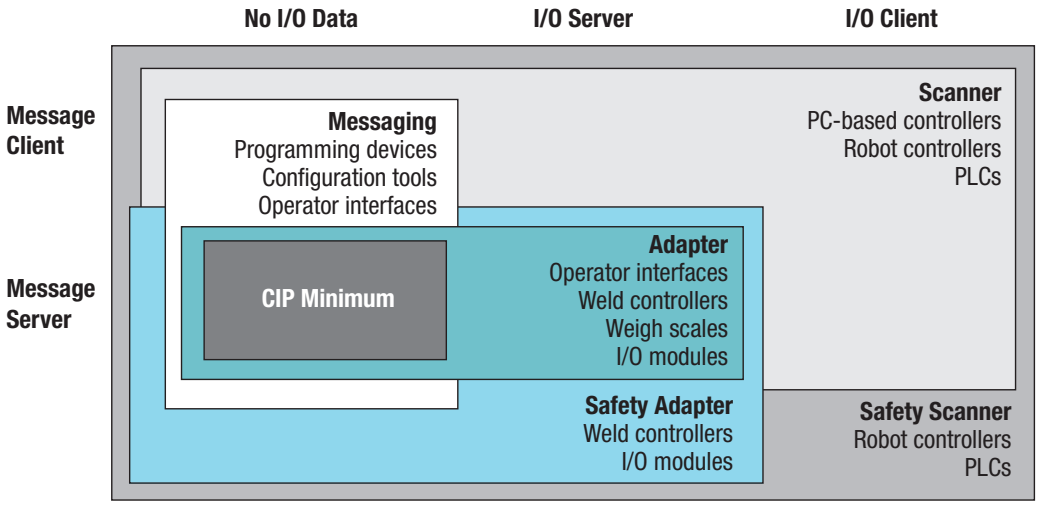

***Figure 4**. EtherNet/IP communications types and device classifications.*

Figure 4 shows the relationship between messaging options and device types. A few examples of types of devices are also included. The headings across the top of the diagram (No I/O Data; I/O Server; and I/O Client) are references to client/server roles and whether or not implicit messaging is involved (and if so, what role it is playing).

All EtherNet/IP devices must have minimal Explicit Message Server capability so that they can respond to simple device identification and configuration requests ("CIP Minimum" in **Figure 4**). An Explicit Message Client is required to enable communication with a device that supports only explicit message server communications.

Understanding whether explicit or implicit messaging is appropriate normally depends on the nature of the communication. Explicit messaging is easier to implement but better suited for modest performance requirements such as request/response communications. Implicit messaging is needed for higher performance and more deterministic communications.

OSI model

The protocol layers of EtherNet/IP can be mapped to the Open Systems Interconnection (OSI) model that characterizes and standardizes the internal functions of a communication system by partitioning it into abstraction layers. Understanding how CIP utilizes the Data Link, Network and Transport Layers are of particular interest because they affect the types and forms of CIP messaging.

The data link layer

IEEE's 802.3 specification is used to transmit packets of data from device to device on the EtherNet/IP Data Link Layer. The same Ethernet CSMA/CD media access mechanism determines how networked devices share a common bus (i.e., cable), and how they detect and respond to data collisions.

Network and transport layers

At the Network and Transport Layers, EtherNet/IP utilizes the TCP/IP protocol suite for messaging. TCP/IP provides the communication protocol features needed to implement fully functional networks that the IEEE specification lacks.

Messages used by all CIP networks are encapsulated, which means a node on the network can embed a message as the data portion in an Ethernet message. The node then sends the message – TCP/IP protocol with the message inside – to an Ethernet chip (the Data Link Layer). By using TCP/IP, EtherNet/IP can send explicit messages, which are used to perform client-server type transactions between nodes.

For real-time messaging, EtherNet/IP employs UDP to multicast to a group of destination addresses and to implement I/O data transfers (implicit messaging). The data field contains no protocol information, only real-time I/O data. Since the data's meaning is pre-defined when the connection is established, processing time is minimized. UDP messages are smaller and can be processed more quickly than explicit messages but UDP is connectionless and does not guarantee that data will be transmitted from one device to another. However, the CIP connection process provides timeout mechanisms that can detect data delivery problems, a capability that is essential for reliable control system performance.

EtherNet/IP uses two forms of messaging:

- Unconnected messaging for infrequent, low-priority messages. Unconnected messages on EtherNet/IP utilize TCP/IP resources to move messages across Ethernet.
- Connected messaging on EtherNet/IP utilizes resources within each node that are dedicated to a particular purpose, such as frequent explicit message transactions or real-time I/O data transfers.

The process of opening a connection is called Connection Origination, and the node that initiates the connection establishment request is called a Connection Originator, or just an Originator. Conversely, the node that responds to the establishment request is called a Connection Target, or a Target.

Components of an EtherNet/IP node

An EtherNet/IP node has four layers corresponding to the modified OSI model shown in **Figure 5**.

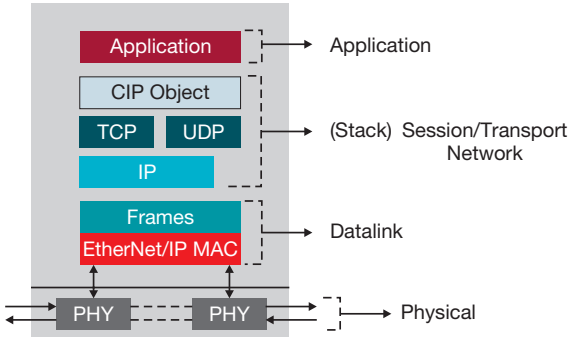

***Figure 5**. EtherNet/IP node.*

The physical layer transmits bitstream data through the network. Because EtherNet/IP is fully compatible with Ethernet, it can use any Ethernet-capable twisted-pair copper or fiber optic cabling that supports 100 Mbit/s data rates. The MAC layer can be implemented in one of three ways: an ASIC, FPGA, or, custom hardware running high-speed firmware. The industrial application has but one restriction. It must support a standard TCP/IP and UDP/IP stack and EtherNet/IP-based device profiles. Within the EtherNet/IP node, the application can run on hardware or a hardware/software combination implemented by an embedded CPU.

Typical EtherNet/IP node

Designers have the choice of three common architectures when implementing an EtherNet/IP node.

For cost-sensitive applications that do not require software because the device's functionality is implemented 100% in hardware, an FPGA or ASIC can be used. This architecture is shown in **Figure 6**.

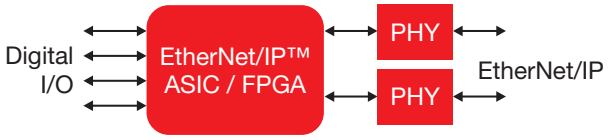

***Figure 6**. Basic digital I/O EtherNet/IP node.*

When more processing power is needed, it is frequently provided by the addition of an external processor with on-chip Flash memory. The ASIC or FPGA is still an integral part of the architecture (see **Figure 7** on the following page). Sensor applications frequently utilize this type of node. The processor operates the sensor, implements the device driver and runs the EtherNet/IP protocol stack. The additional hardware increases cost compared to simple digital I/O device implementations but it also allows designers to select a processor that fits their needs and cost targets.

***Figure 7**. EtherNet/IP with ASIC and external processor.*

The third common architecture for implementing EtherNet/IP applications turns the EtherNet/IP node into one of the peripherals in a device with an integrated CPU. This architecture is shown in **Figure 8**. The processor may be configured using available gates in an FPGA. Another option available with some FPGAs is to use one with an integrated processor. Similarly, ASIC vendors have integrated EtherNet/IP and a processor on their device. FPGA implementations offer the advantage of being flexible but have a downside as well because of the possibility of not meeting cost or operating frequency targets depending on the processor available on the FPGA.

***Figure 8**. Integrated EtherNet/IP with processor.*

Ethernet/IP solutions from Texas Instruments

Texas Instruments (TI) has integrated EtherNet/IP functionality into its Sitara™ processors. These devices are highly integrated with peripherals and interfaces that make them ideal for industrial automation applications.

The Sitara processors include the programmable real-time unit industrial communication subsystem (PRU-ICSS), which supports very low-level interaction with the MII interfaces and, therefore, can easily implement EtherNet/IP. The entire Ethernet MAC layer is encapsulated in the PRU-ICSS through firmware.

As a processing efficiency measure, EtherNet/IP nodes process only those packets that are addressed to them and forward all other frames to the next device. Communication with the application and the Arm core running the EtherNet/IP stack (Layer 7) is accomplished by using interrupts. When EtherNet/IP is integrated into a Sitara processor, almost all of the low-level, high-speed EtherNet/IP functionality (DLR and PTP/1588) is handled by the PRU-ICSS. When this is the case, the Arm core can allocate almost all of its processing power to running the stack and complex applications such as motor control.

Ethernet PHY devices such as TI's TLK110 or DP83822 complete TI's Sitara EtherNet/IP solutions. The TLK110 is optimized for low latency between the MII and PHY interfaces, which is an important performance attribute. The TLK110 also has advanced cable diagnostics features that can quickly locate cable faults.

EtherNet/IP software architecture

As shown in **Figure 9**, three software components will comprise EtherNet/IP slave implementations on TI Sitara devices: (1) firmware that implements Layer 2 functionality in the PRU; (2) the EtherNet/IP slave stack that runs on the Arm core; and (3) the industrial application. TI provides additional supporting components such as the protocol adaptation layer and device drivers in its software development kit.

Firmware

The firmware architecture is shown in **Figure 10**.

When EtherNet/IP is integrated into a Sitara processor, the PRU-ICSS implements basic Ethernet switch protocols including features such as MAC learning, storm prevention and packet statistics. The two real-time cores that make up each PRU-ICSS are independently responsible for controlling the two physical ports. Each PRU core is responsible for one RX/TX combination as shown in the diagram. The PRU cores communicate with each other to ensure coordination using a set of special instructions and shared memory. TI's PRU-ICSS architecture allows low latency store and forward between the ports based on configurable parameters. PRUs also have the ability to interrupt Arm core execution in real time to ensure deterministic processing.

DLR and PTP/1588

In addition to its primary mission of running a set of stack features on top of basic Ethernet, EtherNet/IP executes two other valuable features: A ring redundancy protocol known as Device Level Ring (DLR) and an IEEE standard for high accuracy time

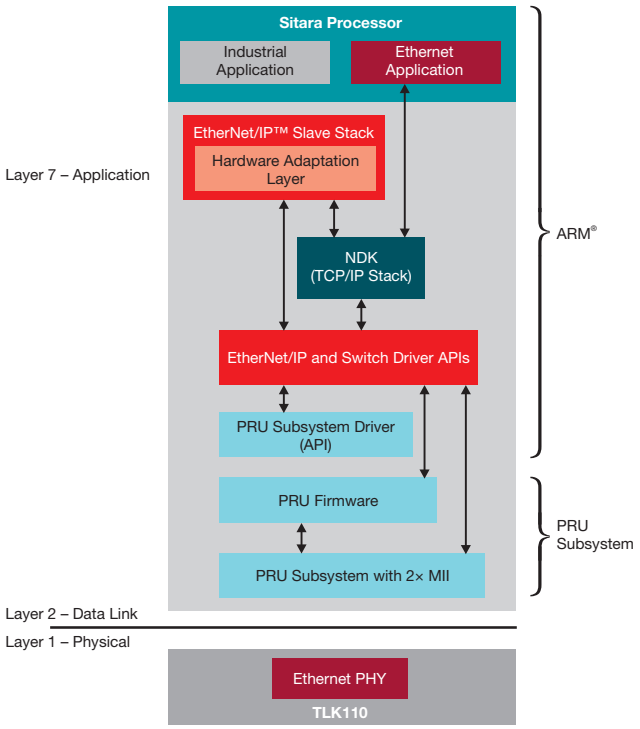

***Figure 9**. Software architecture for EtherNet/IP slave.*

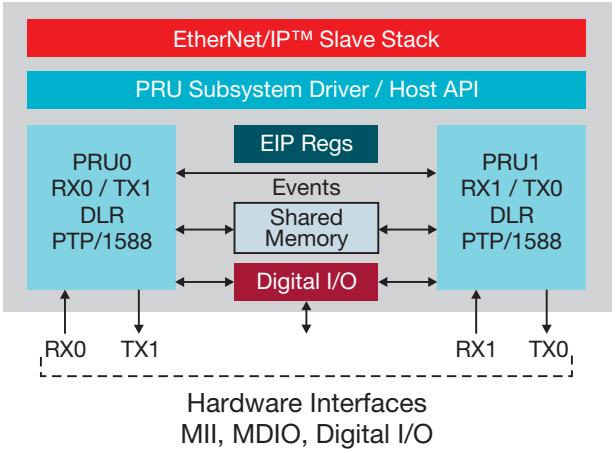

***Figure 10**. Firmware architecture.*

synchronization across devices known as PTP/1588. When integrated into a Sitara processor, the PRU-ICSS implements both these features. Thanks to its deterministic real-time processing capabilities, the PRU-ICSS processes these frames with very low latency. While the main state machine is on the PRU-ICSS, the Arm core is also partially involved in the state machine required to execute DLR and PTP/1588.

TI's integrated EtherNet/IP processor, with TI's TLK110 or DP83822 Ethernet PHY device has a latency of less than 2 µs, which places it among the leading EtherNet/IP slave solutions.

Easy EtherNet/IP integration

TI is making it easy to integrate EtherNet/IP with its Sitara processors. All of the tools and software code required to integrate EtherNet/IP slaves are available as part of the Processor Software Development Kit (Processor SDK) with protocol-specific software available for download through the PRU-ICSS industrial software page, which includes firmware for the EtherNet/IP protocol, software drivers, hardware initialization routines, an adaptation layer for the stack application programming interface (API), EtherNet/IP protocol stack and the application itself. The SDK comes with supporting documentation that will help users modify and build new features into applications.

TI development tools for EtherNet/IP implementation

TI offers several industrial hardware development platforms to assist customers with their implementations. All design data for these platforms, including schematics and layout, is available to help customers accelerate development and time to market.

For more information on the tools available for specific processors, click here.

In addition, TI also collaborates with external vendors for an additional development platform targeted for industrial applications.

Summary

Ethernet's lack of determinism has limited its usefulness in industrial applications that require real-time responsiveness. EtherNet/IP offers an efficient solution by adding a set of stack features that runs on top of basic Ethernet to handle real-time applications. While there are several approaches to implementing an EtherNet/IP node, the most flexible and powerful is to integrate EtherNet/IP functionality in an embedded processor. TI's Sitara processors have all of the resources needed to accomplish this integration. In fact, TI has integrated EtherNet/IP slave capabilities into Sitara processors, providing developers with a powerful, low-power solution that offers lower-cost end products without compromise of operational requirements.

TI also offers the transceivers with built-in isolation for the industrial communication interfaces such as EtherCAT, PROFINET, PROFIBUS, CAN, RS-485 and more. TI's comprehensive software and hardware development tools, worldwide support and an active TI E2E™ developer community provide developers with greatly simplified EtherNet/IP integration and significant cost savings.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265