ODVA
2012
Industry Conference
and 15th Annual Meeting

Extracting Energy Data from MODBUS Devices Using CIP

Rick Blair
Schneider Electric

Technical Track

MODBUS vs. CIP

MODBUS

- Flat memory architecture
- 16-bit data organization *Registers) and bits
- No definition for other data types
- No predefined memory layouts
- Inconsistent device identification

CIP

- Object based
- \Consistent identification methods

Current Situation

Many MODBUS devices exist that measure power and energy

No consistent data representation

- Across manufacturers
- Within manufacturers

Need custom software interfaces

Popularity of MODBUS will result in continued similar product offers

CIP Energy Objects

Three new energy objects added to CIP

- Base Energy Object
- Electrical Energy Object
- Non-Electrical Energy Object

Defines standardized data sets and services

Base Energy Object

Energy Supervisor

- Capabilities
- Accuracy
- Paths
  - To Subordinate Objects
  - To Aggregated Objects
- Reports energy and/or power
- Standardized reporting units (kWh/kW)

Electrical Energy Object

Subordinate to Base Energy Object

- Associated Base Energy Object Path EPATH
- Standardized reporting of electrical attributes
  - Energy
  - Power
  - Voltage
  - Current
  - Power Factor
  - etc.

Non-Electrical Energy Object

Inclusive of all energy related resources

- Not only electricity!
- Native reporting units
  - Natural Gas in Therms, Chilled Water in Mbtu, etc.
  - Units from ENGUNIT data type (Appendix D)
  - Or text string
- Standardized reporting units
  - Conversion factor to kWh
  - Permits aggregation of diverse energy resources
  - Multiplier/divisor unit conversion factors

Overview - MODBUS to CIP Energy Data Extractor (MCEDE)

Collects energy data from MODBUS devices and puts it into CIP Energy objects

MODBUS port(s)

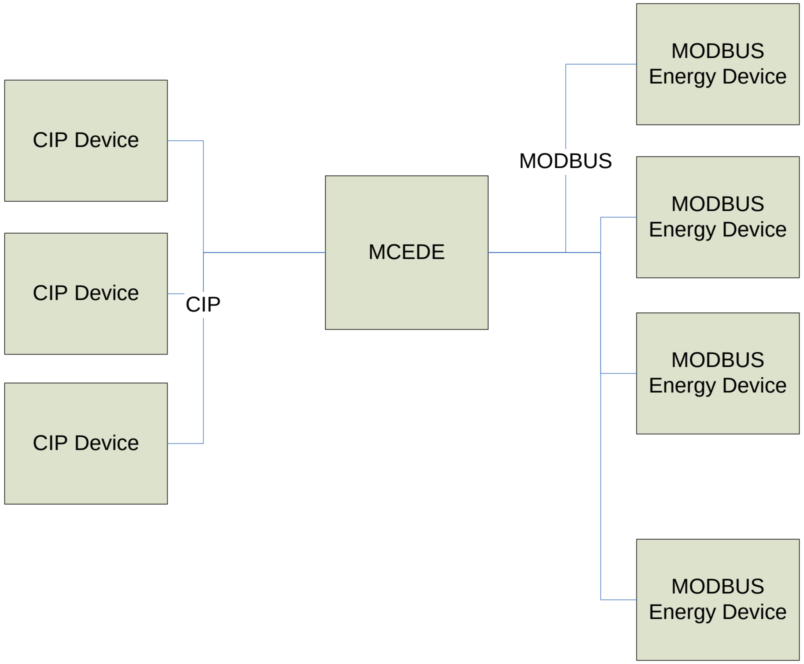
- Serial
- Ethernet

CIP port(s)

- EtherNet/IP
- DeviceNet
- Etc.

Implementation Platform(s)

- PC
- Dedicated



Basic MCEDE Functions

- A set of MODBUS energy device descriptions
- A set of data type conversion functions
- A configuration function
- A method to add/delete MODBUS energy device descriptions
- A scan function to periodically read MODBUS data
- A discovery function to search for MODBUS devices (optional)
- A MODBUS driver
- A CIP driver
- An energy object service handler

MODBUS Energy Device Description

Device level descriptors

- Number and type of CIP energy objects needed
- Which MODBUS data should be extracted
- Order of multi-register data values
- MODBUS register blocks to read
- How to identify the MODBUS energy device
- Which CIP energy object services should be provided

For each CIP object attribute supported

- MODBUS register address
- Data type
- Units

MODBUS Device Identification

Only recently standardized

Some MODBUS devices do not support any method for identification

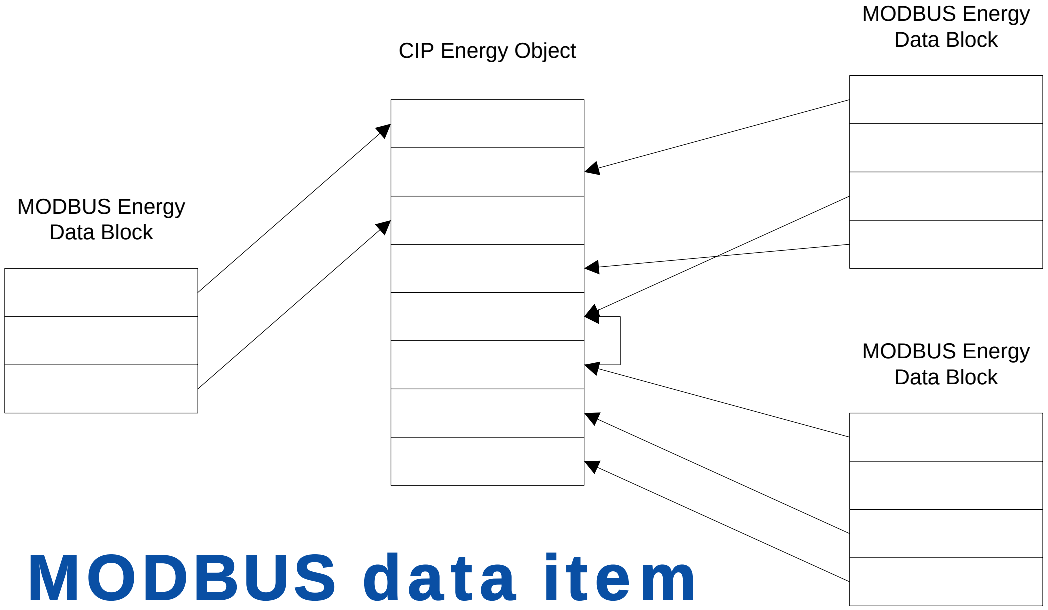
Some use register-based signatures

Standard method uses Read Device Information (RDI) function code

Data Order Considerations

- MODBUS is a 16-bit register based protocol
- No standard for how to represent multi-register data
- Different data order methods exist in current MODBUS devices
- Assume consistent within a MODBUS device

Optimized MODBUS Data Collection



Could read each MODBUS data item

- inefficient

Group multiple items into single read request, even if some data is not used

Describing the MODBUS Energy Device

Use MODBUS device description file

- EDS
- XML (preferred)

Embed In MODBUS device

- Description file using MODBUS Read file function code
- Use predefined registers and signature (e.g. SunSpec)
- Enhance Read Device Information function code to include mapping description

Configuration

Add / remove MODBUS device descriptions

Configure port settings

Map MODBUS devices to CIP object instances

Configure scanning update rates

MODBUS Device Identification

- Scan for MODBUS devices to aid configuration
- Make sure device matches associated description
- Not possible for MODBUS devices with no resident identification method

Data Conversion and Scaling

MODBUS data may be different data type and / or units

Implement conversion routines for each CIP data type

Consider both decrease and increase in max values

Use simple scaling factor for units conversion

Scanning MODBUS Energy Devices

Simple scanner

- Read all from one device
- Move to next device
- Repeat until all devices read
- Start over at the beginning

Complex Scan

- Allow choice of update time for each MODBUS device
- Consider time management for possible scan rate conflicts

Updating CIP Objects

- MODBUS data conversion may take time
- Consider two copies of each instance to better support the Get_Attributes_All service
- Update one copy and then switch context
- Supported services may vary per Energy Object instance

Energy Object Services

MCEDE requires dynamic creation / deletion of CIP Energy objects as MODBUS devices are added / removed

Each instance may contain different optional attributes

Conclusion

MCEDE can be built using PC or dedicated platform

Description file allows use of MODBUS devices without modification

Data order, type and word order may not be similar between MODBUS and CIP

MCEDE should support dynamic addition / removal of MODBUS energy devices