

Case Study: IEC 60870-5-101 Enhancement Project

Provider Name : KALKI Communication Technologies (P) Ltd.

Client Name : Fortune 100 Utility Automation Multi-national

Project Title : IEC 60870-5-101 Enhancement Project

The Problem

The client required to enhance the IEC 60870-5-101 Interoperability Support on their existing RTU. The existing RTU used a protocol converter card, which converted the OEM's proprietary protocol to IEC 60870-5-101 protocol. The existing implementation required support for additional baud-rates, multiple IEC 60870-5-101 Slave port support, Scan Group Support, File Transfer Support etc.. Hence it was required to implement the required Function Codes to the existing IEC 60870-5-101 protocol stack.

The Solution

The stated driver was developed using the following Resources:

1. Protocol Converter Hardware and Software Documentation
2. IEC 60870-5-101 specifications and Proprietary protocol specifications
3. IEC 60870-5-101 Interoperability Document Support for multiple master support

The existing RTU protocol converter card supports a real-time task switching kernel. This kernel runs a multitude of tasks including the proprietary protocol to communicate with the RTU Controller, MODBUS Protocol Task, DNP3 Task, IEC Task etc., Each task can be assigned specific priorities and gets processor time slices accordingly. The protocol converter architecture supports protected intermediate event and static database. This data base is used by the different protocols Tasks.

Our scope in the project was to implement functions like Scan groups, File Transfer, Multiple Port Support, Counter Interrogation etc, and to maintain the software and provide onsite and off-shore support to the implementations. This required corresponding feature enhancement of the communication interface with the RTU Controller, to ensure that there is bi-directional support for the feature enhancements. The enhanced features were implemented and tested with standard protocol test simulators to ensure that true interoperability is achieved.

Tools Used:

- Borland C Compiler
- Proprietary Real Time Task Switching Kernel
- Embedded 80C186 Processor Development and Debugging Environment
- IEC 60870-5-101 Test Simulator
- Proprietary Protocol Test Simulator