openPOWERLINK

Open Source Industrial Ethernet

CLT2010
Version V1.02  13.03.2010
Contact Details

Dipl.-Inf. Daniel Krüger
SYS TEC electronic GmbH
August-Bebel-Str. 29
D-07973 Greiz
Tel.:  +49 (0)3661 6279-0
Fax.:  +49 (0)3661 6279-99
WWW:  www.systec-electronic.com
Email:  Daniel.Krueger@systec-electronic.com
Limitations of Ethernet

- Non-determinism, because of collisions
- Carrier Sense Multiple Access/Collision Detection (CSMA/CD): retransmission after random time, transmission is not guaranteed

→ Possible solution: classic fieldbusses like CAN, Profibus, LON, etc.
Limitation of classic fieldbusses

- Classic fieldbusses have a low bandwidth that is not sufficient for the higher demands of contemporary applications (e.g. for firmware updates)
- CAN: max. 1 Mbit/s
- LON: max. 78 kbit/s
- Profibus-DP: max. 12 Mbit/s
- Profibus-PA: max. 31.25 kbit/s

Solution: Industrial Ethernet like POWERLINK, EtherCAT, Sercos III, Profinet IRT
Applications of Microcontroller Networks

- Central/decentral control or data capturing of decentral processes
- Data exchange between sensors and actuators
- Data capturing over large distances (>1km)
- Reducing the complexity of the wiring

Typical applications:
- industrial automation
- automotive engineering, shipbuilding
- building control, alarm systems
- power plants
- measurement engineering
- ...
What is POWERLINK?

- *Industrial Ethernet* Fieldbus Protocol
- Based on IEEE 802.3u Fast Ethernet
- CANopen over Ethernet
- Real-time capable via slot communication
- Master-Slave Protocol
- Master = Managing Node (MN)
- Slave = Controlled Node (CN)
- Hot plugging
- Direct cross-traffic
What is openPOWERLINK?

- Open Source implementation of POWERLINK
- Development done by SYS TEC electronic GmbH
- Currently supported target platforms:
  - Linux
  - Windows
  - bare-metal (OS-less)
- License: BSD
- Pure software-based solution on standard Ethernet controllers, but hardware-acceleration possible
How does POWERLINK work?

Slot Communication Network Management
SoC Frame

**SoC**: Start of Cycle synchronization event
PReq Frame

**PReq:** Poll Request from MN for specific CN contains PDO payload
**PRes Frame**

**PRes**: Poll Response from CN / MN contains PDO payload and current NMT state
SoA Frame

**SoA**: Start of Asynchronous assigns asynchronous phase to specific node
ASnd Frame

**ASnd**: Asynchronous Send
NMT commands, SDO, IdentResponse, StatusResponse

TCP/IP
OSI Model
Topologies:
- Star
- Line (daisy chain)
- Tree
- Mixed

Connectors:
- RJ45
- M12

Equipment:
- Hubs
- Switches
Node Addressing Scheme

- Unique Node-ID for each node in the network
- Node-ID of MN: 240 = 0xF0
- Node-ID of CNs: 1 – 239
- Node-ID of Gateway: 254
- Node-ID of diagnostic device: 253
- CN 48 = 0x30
Performance

- 0.1 µs system synchronization
- 100 µs cycle time
- Up to 240 nodes in one network
  - 480 synchronized axes
  - 460,000 digital I/O data points
- Unlimited extension
  - 100 m between any two nodes
  - Larger distances using fiber optics
NMT States and Status LED

- Initialising
- Reset Application
- Reset Communication
- Reset Configuration

- Not Active
- Pre Operational 1
- Pre Operational 2
- Ready-To-Operate
- Operational
- Basic Ethernet
- Stopped
Object Dictionary (3)

- Access rights: ro, rw, wo, const
- Device and application profiles of CANopen are used
- Overall layout according to specification:

<table>
<thead>
<tr>
<th>Index</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0001 – 0xFFFF</td>
<td>Declaration of Data Types (formalism)</td>
</tr>
<tr>
<td>0x1000 – 0x1FFF</td>
<td>Communication Profile Area</td>
</tr>
<tr>
<td>0x2000 – 0x5FFF</td>
<td>Manufacturer Specific Profile Area</td>
</tr>
<tr>
<td>0x6000 – 0x9FFF</td>
<td>Standardised Device Profile Area</td>
</tr>
<tr>
<td>0xA000 – 0xBFFF</td>
<td>Standardised Interface Profile Area</td>
</tr>
<tr>
<td>0xC000 – 0xFFFF</td>
<td>Reserved for further use</td>
</tr>
</tbody>
</table>
Object Dictionary (4)

- **XML Device Description (XDD):**
  - Describes the structure of the OD (including default values) of a device type (e.g. I/O module from manufacturer ABC)
  - Is the basis for configuration tools

- **XML Device Configuration (XDC):**
  - Contains the current configuration of the OD of a specific device (e.g. I/O module at the left drive in the machine)
  - Generated by configuration tools
  - Concise Device Configuration (CDC): binary form of XDC
  - Used by the Configuration Manager (CFM) to configure the nodes in the network at run-time
OD Communication Profile

- 0x1000: NMT_DeviceType_U32
- 0x1006: NMT_CycleLen_U32
- 0x1018: NMT_IdentityObject_REC (VendorID, ProductCode, RevisionNo, SerialNo)
- 0x1400 .. 0x14FF: PDO_RxCommParam_XXh_REC
- 0x1600 .. 0x16FF: PDO_RxMappParam_XXh_AU64
- 0x1800 .. 0x18FF: PDO_TxCommParam_XXh_REC
- 0x1A00 .. 0x1AFF: PDO_TxMappParam_XXh_AU64
- 0x1CXX: Error counters
- 0x1F82: NMT_FeatureFlags_U32
- 0x1F98: NMT_CycleTiming_REC
CANopen Device/Application Profiles

- CiA 401: Generic I/O modules
- CiA 402: Drives and motion control
- CiA 404: Measuring devices and closed-loop controllers
- CiA 406: Encoders (rotating and linear)
- CiA 417: Lift control systems
- CiA 422: Municipal Vehicles (e.g. garbage trucks)
- CiA 445: RFID reader
- CiA 447: Special-purpose car add-on devices (e.g. in taxis)
Process Data Object (PDO)

- For exchange of process data between actuators, sensors and control
- Abstraction layer between source and sink (virtual cabling via variable mapping)
- Up to 1490 Bytes per Ethernet frame
- Structure is stored in OD and can be configured via SDO
- Communication model: Producer-Consumer(s)
- Cyclically transmitted in isochronous phase
- Direct cross-traffic between CNs is possible
**PDO Mapping**

**Managing Node 0xF0**

Master to slave

<table>
<thead>
<tr>
<th>Process Variables:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x2001/1  Output 1, 8 Bit, 0xAB</td>
</tr>
<tr>
<td>0x2001/2  Output 2, 8 Bit, 0xCD</td>
</tr>
</tbody>
</table>

**Controlled Node 0x02**

<table>
<thead>
<tr>
<th>Process Variables:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x6200/1  Output 1, 8 Bit, 0xCD</td>
</tr>
<tr>
<td>0x6200/2  Output 2, 8 Bit, 0xAB</td>
</tr>
</tbody>
</table>

**Tx PDO Mapping Parameter:**

- 0x1A00/0  NrOfEntries: 2
- 0x1A00/1  1. Mapped Object:
  - 0x000800000012001LL
- 0x1A00/2  2. Mapped Object:
  - 0x000800100022001LL

**Rx PDO Mapping Parameter:**

- 0x1600/0  NrOfEntries: 2
- 0x1600/1  1. Mapped Object:
  - 0x0008000000026200LL
- 0x1600/2  2. Mapped Object:
  - 0x0008001000016200LL

**Tx PDO Communication Parameter:**

- 0x1800/0  NrOfEntries: 2
- 0x1800/1  NodeID: 0x02
- 0x1800/2  MappingVersion: 0x00

**Rx PDO Communication Parameter:**

- 0x1400/0  NrOfEntries: 2
- 0x1400/1  NodeID: 0x00
- 0x1400/2  MappingVersion: 0x00

**Resulting PReq frame to Node 0x02:**

<table>
<thead>
<tr>
<th>SrcNodeId</th>
<th>DstNodeId</th>
<th>MappingVersion</th>
<th>PDO Length</th>
<th>PDO Payload</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xF0</td>
<td>0x02</td>
<td>0x00</td>
<td>3</td>
<td>0xAB 00 CD</td>
</tr>
</tbody>
</table>

**openCONF Demo**
Service Data Object (SDO)

- For configuration of the nodes, to get access to remote ODs
- Transported in asynchronous phase via ASnd frames or encapsulated in UDP datagrams
- Transportation in isochronous phase via container in PDO is possible too, but currently not implemented
- Consists of 2 Layers:
  - Sequence Layer: Peer-to-Peer transport stream with acknowledgement
  - Command Layer: Client-Server communication
SDO Aborts

- SDO transfers can be aborted by both parties
- Common SDO abort codes
  - 0x05040000L: Timeout occurred
  - 0x06010000L: Unsupported access of object
  - 0x06010001L: Read of write-only object
  - 0x06010002L: Write to read-only object
  - 0x06020000L: Object does not exist
  - 0x06090011L: sub-index does not exist
  - 0x08000000L: General error
openPOWERLINK

- Implements Communication profile EPSG DS 1.1.0
- Data link layer and NMT state machine for Controlled and Managing Nodes
- SDO via UDP and EPL ASnd frames
- Dynamic PDO mapping
- User-configurable object dictionary
- Supports the EPL cycle features async-only CN and multiplexed CN
- Implemented in plain ANSI C
- Modular software structure for simple portability to different target platforms
- Supports target platforms with and without operating system
- Event driven Communication Abstraction Layer (CAL)
- Provides Generic API for user-application
Software Structure

- Low-prioritized processes
  - Called: user part

- High-prioritized real-time processes
  - Called: kernel part
# Directory Structure

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edrv</td>
<td>Ethernet driver implementations</td>
</tr>
<tr>
<td>EplStack</td>
<td>EPL protocol stack core components</td>
</tr>
<tr>
<td>Example</td>
<td>Example and test projects</td>
</tr>
<tr>
<td>Include</td>
<td>Generic header files</td>
</tr>
<tr>
<td>Include/kernel</td>
<td>Header files for EPL kernel part</td>
</tr>
<tr>
<td>Include/user</td>
<td>Header files for EPL user part</td>
</tr>
<tr>
<td>ObjDicts</td>
<td>Sample Object dictionaries</td>
</tr>
<tr>
<td>SharedBuff</td>
<td>Shared buffer implementation for CAL and frame queues</td>
</tr>
<tr>
<td>Target/ARCH/OS/C</td>
<td>Target dependant files for architecture ARCH, operating system OS and compiler C</td>
</tr>
</tbody>
</table>
Introduction into demo project

- Makefile
- demo_main.c
  - EplApiInitialize();
  - EplApiLinkObject();
  - EplApiExecNmtCommand(kEplNmtEventSwReset);
  - EplApiShutdown();
  - AppCbEvent()
  - AppCbSync()

- Build and run the project
Extensions and future enhancements?

- Multiplexed CNs (currently only in CN stack supported)
- PResMN
- PResChaining (implementation on-going)
- Future enhancements:
  - High Availability/ Redundancy
  - Safety/ openSAFETY protocol stack
Terms and Abbreviations

- EPL: Ethernet POWERLINK
- EPSG: Ethernet POWERLINK Standardization Group
- NMT: Network Management
- OD: Object Dictionary
- PDO: Process Data Object
- SDO: Service Data Object
- CN: Controlled Node (Slave)
- MN: Managing Node (Master)
References


The End

Any questions about POWERLINK or openPOWERLINK?

Thanks for your attention!