KPA EtherCAT Master 2 Run-time

White Paper

Revision date: 2025-03-26

Contents

1. Introduction

This white paper covers the issues related to the following points:

- EtherCAT technology benefits and the EtherCAT protocol advantages
- KPA EtherCAT Master 2 Stack architectural, functional and implementation details
- Explanation of common appliances
- KPA EtherCAT Master 2 Stack additional features

This document is also intended to simplify the process of getting started with KPA EtherCAT Master 2 (hereafter referred to as Master 2).

2. EtherCAT at a glance

Nowadays field bus systems have become an integral part of real-time distributed control. It is an effective way to increase the safety aspect of controlling and monitoring the production process.

2.1. EtherCAT - Ethernet Control Automation Technology

Although conventional fieldbus systems (PROFIBUS, CANopen, DeviceNet, SERCOS, etc.) provide more or less fast and safe data transfer, they are inferior to the EtherCAT technology in special considerations such as speed, overall productivity, reliability and real-time transmission.

EtherCAT is a good chance to eliminate the bottlenecks of conventional fieldbus systems allowing direct data exchange with the application by using shared memory without additional hardware since standard network adapters are enough.

EtherCAT (Ethernet for Control Automation technology) is a hard-real-time technology that realizes the specific transfer of data. It offers real-time performance and is aimed to maximize the utilization of high-speed full-duplex Ethernet data transfer through twisted pair or fiber optic cable for industrial process control needs.

EtherCAT based on the Ethernet technology possesses such advantages as the ease of implementation, cost of ownership and standardization, which makes it a perfect solution for industrial applications intended to maximize the performance of control systems.

Medium access control employs the Master/Slave principle, where the Master node (typically the control system) sends the Ethernet frames to the slave nodes, which extract data from and insert data into these frames on the fly.

2.2. Operating principle

An EtherCAT segment is a single Ethernet device from an Ethernet point of view, which receives and sends standard ISO/IEC 802-3 Ethernet frames. This Ethernet device may consist of a large number of EtherCAT slave devices, which process the incoming frames directly and extract the relevant user data, or insert data and transfer the frame to the next EtherCAT slave device. The last EtherCAT slave device within the segment sends the fully processed frame back, so that it is returned by the first slave device to the Master as a response frame.

This procedure utilizes the full duplex mode of Ethernet, which allows for independent communication in both directions. Direct communication without a switch between a Master device and an EtherCAT segment consisting of one or several slave devices may be established. This demonstrates the flexibility of the EtherCAT operating principle.

2.3. Core features of EtherCAT

The **core features** of EtherCAT should be emphasized:

- **The fastest industrial Ethernet technology**
  EtherCAT is considered to be the fastest industrial Ethernet technology considering the way the protocol is designed in order to fully utilize the advantages of industrial Ethernet
- **High performance**
  Obviously, it is a huge boost to your productivity as EtherCAT is designed to achieve high performance due to the flexibility of its topology, EtherCAT protocol specific features and the way

of data mapping. For example, with direct memory access (DMA) data can be transferred between the network card and the Master processor or slave I/O with minimal CPU usage. Slaves write and read data themselves and there is only one telegram passing through all the slaves, which is returned to the Master processed. This reduces the complexity of the Master and frees its resources

- **Low cost technology**
  EtherCAT is a real-time industrial Ethernet technology that does not demand a special plug-in card for the Master, co-processor or a lot of processing power at all. A standard Ethernet connection without hubs and often even switches is the only necessity. Very low-cost slave controller chips are in the price segment of standard Ethernet chips. It assures the lowest hardware costs on Master side and the smallest expenditures for installation
- **Flexible Ethernet topologies**
  EtherCAT network can support up to 65,535 devices without placing restrictions on their topology:
  - Star
  - Line
  - Tree
  - Droplines
  - Ring
- **Integration**
  - with fieldbuses and networks through gateways:
    - CAN/CANopen
    - Ethernet
    - PROFIBUS
    - SERCOS
  - with third-party tools:
    - Mailbox over UDP
    - Mailbox over TCP
- **Openness of technology**
  - open EtherCAT Master specifications
  - hundreds of members worldwide with expertise in manufacturing of slaves and chips, development of software and firmware

To realize solutions based on EtherCAT, we have developed such products as KPA EtherCAT Master 2 and KPA EtherCAT Studio:

- KPA EtherCAT Master 2 to provide interaction of network devices with IPC systems
- KPA EtherCAT Studio to create and modify EtherCAT network configurations

3. Why KPA EtherCAT Master 2?

KPA EtherCAT Master 2 is an EtherCAT master stack that ensures all benefits of EtherCAT technology, such as real-time operation, extremely short cycle time and maximum performance for minimum expenses. The stack architecture has been conceptualized and developed to provide portability to different operating systems (INtime, Xenomai, Linux,FreeRTOS, QNX, RTX64, VxWorks, Windows), adaptation to various hardware platforms, and scalability by Basic (Class B), Standard (Class A) and Premium packages.

KPA EtherCAT Master 2 supports the EtherCAT Network Information format and is implemented to employ the technology completely and efficiently. The stack has been developed in accordance with ETG specifications using ANSI “C” to comply with technology demands and optimized for short execution time and small memory footprint to meet the requirements of hard real-time operation and enable deployment in embedded systems on different hardware platforms.

KPA EtherCAT Master 2 can be shipped as KPA EtherCAT Master 2 Development Kit (MDK 2) that enables OEMs to integrate KPA EtherCAT Master 2 functionality. The MDK includes:

- KPA EtherCAT Master 2 Run-time (MRT 2) package: binaries, user documentation, one license for Full Master (class A with all feature packs and extensions).
- KPA EtherCAT Master 2 Integration Package (MIP 2): Master 2 Core API, Operation System Abstraction Layer (OSAL), Remote Procedure Call (RPC) Server, samples and documentation (API help).
- KPA EtherCAT Studio Premium run-time including all plug-ins and user manuals for configuration and diagnostics of EtherCAT network, one license and documentation (Studio Help).
- Optional IP-Cores and drivers (depending on target platform and OS): encrypted binaries of IP-Cores, OS dependent drivers in binary an optionally source code.

4. KPA EtherCAT Master 2 functions and classes

4.1. KPA EtherCAT Master 2 functions

- Fast process and diagnostic data exchange from EtherCAT Master to application
- Central parameterization of intelligent slaves via UDP
- Adaptation to operating system by special interface
- API for interaction with runtime or/and configuration tools
- Servers for multiple TCP/IP or/and UDP connections

4.2. KPA EtherCAT Master 2 classes

The following KPA EtherCAT Master 2 software packages are available:

1. Basic - Class B according to ETG.1500: “EtherCAT Master Classes” specification
2. Standard - Class A according to ETG.1500: “EtherCAT Master Classes” specification
3. Premium
4. Extension Packages

4.2.1. Basic class

Requested by ETG Basic Master Definition ETG.1500

- Support of all commands and EtherCAT frames
- Support of EtherCAT State Machine (ESM)
- Checking of network or slaves errors
- Cyclic process data exchange
- Event Handler
- Online scanning
- Network configuration taken from ENI
- Polling of Mailbox State of slave
- Write access to EEPROM
- Statistics gathered from NIC and Master
- PI snapshot
- PI logger
- CoE SDO Info Service
- FoE (File Access over EtherCAT)
- Distributed clocks (DC)
  - DC support
  - Time distribution (Slaves synchronization)
- Slave-to-slave Communication

4.2.2. Standard class

In additional to Basic class features this class includes:

- Distributed clocks (DC)
  - Synchronization of Master and slaves with continuous delay compensation
  - Sync window monitoring
- Servo drive profile over EtherCAT (SoE)
- Ethernet over EtherCAT (EoE)
- Vendor specific protocol over EtherCAT (VoE)
- ADS over EtherCAT (AoE)

4.2.3. Premium class

In additional to Standard class features this class includes:

- Hot-Connect
- Cable Redundancy (ensures access to all slaves even in case of one-line break or one damaged device)
- TCP or UDP/ IP Mailbox Gateway for configuring devices via EtherCAT
- Frame Logger and Data Logger for monitoring traffic with triggering without external tools
- Explicit Device Identification

Some of these features can be included into other classes additionally.

4.2.4. Extension Packs

- External synchronization for two or more EtherCAT buses with an external reference clock via Master
- Access Rights for different users of Master
- CAN DBC driver
- COM port emulation
- Multimaster
- Autoconfigurator
- Master Redundancy
- Hardware Timed Send
- Hardware Send Scheduler

All of these features can be included into Standard or Premium classes additionally.

4.3. Trial mode limitation

KPA EtherCAT Master 2 in the trial mode is full-featured KPA EtherCAT Master 2 with 60 minutes (1 hour) time limitation for operating and a trial license applied. When the trial time is expiered, the Master 2 will be switched to INIT state and all I/O operations will be interrupted.

5. KPA EtherCAT Master 2 architecture

The core of the Master 2 can be divided into the following logical parts:

- User Application Programming Interface (API) which allows to configure and manage EtherCAT bus
  - Master initialization and configuration
  - Process image handling
  - Mailbox protocols handling
  - Master/slave state control
  - Network state (slaves scanning)
  - Master statistics, diagnostics
  - Traces/error handling
- The core as itself
  - Process Image
  - Frame and Task schedulers
  - Set of Modules where each of them executes a certain functionality (DC, Event Handler, CoE, FoE, S2S, Synchronization queue, etc.)
- OS abstraction layer
  - EtherCAT network driver (API functions: ecat_open, ecat_close, ecat_is_opened, ecat_send_frame, ecat_recv_frame, ecat_get_statistics, ecat_reset_statistics, ecat_get_link_state, ecat_set_callback, ecat_is_available, ecat_get_adapterslist. Driver is closely bonded to OS dependent network stack or network card driver, i.e. underlying network layer)
  - Wrappers for thread handling functions, process synchronization functions (mutexes, semaphores, etc.), time handling functions

The KPA EtherCAT Master 2 architecture with the bundled modules is represented as depicted below.

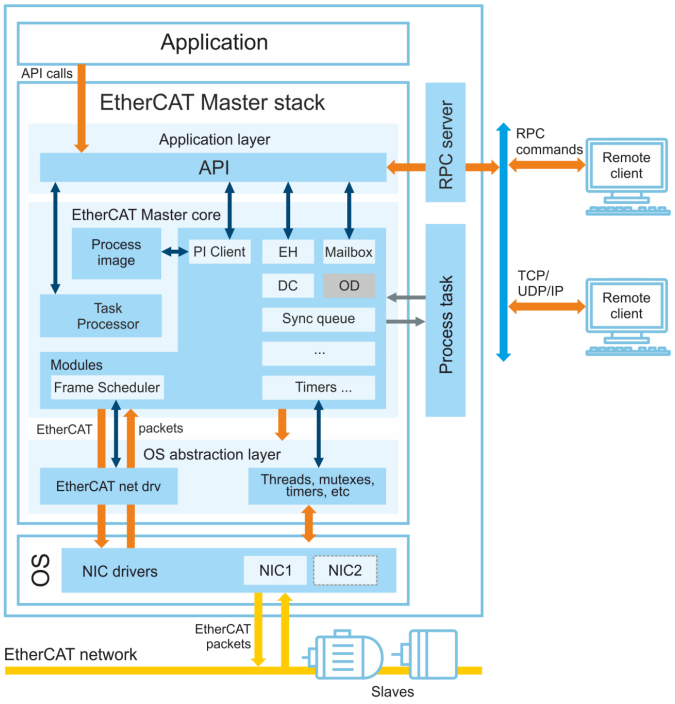

Figure 1. KPA EtherCAT Master 2 architecture

Items description:

Application - this is a separate process which invokes Master 2 API functions. The application performs almost full control over Master 2 by means of API: starts/stops the Master, configures it, updates PI, etc.

API - user Application Programming Interface (API) which allows to configure and manage EtherCAT bus.

RPC server - part of KPA EtherCAT Master 2 stack. The RPC-server is responsible for establishing connection with a remote client (for example, with KPA EtherCAT Studio) and forwarding requests to the Master 2 stack core. It supports user-to-user and user-to-kernel modes (through IOCtl calls).

Process Image - Process Image consists of Shadow buffer (i.e. data transferred directly to/from the network) and Active buffer (i.e. buffer which contains data received/sent from/to clients of Master 2).

Process task (external task) - callback function which implements application-specific control algorithm. This callback function is called at every repetition of PI update (high-priority) cycle.

Frame scheduler - module which assembles EtherCAT frames and forwards them to the EtherCAT network driver according to their priorities.

EtherCAT net drv (EtherCAT network driver) - module which abstracts the Master 2 stack core from the underlying network implementation.

Threads, mutexes, timers, etc. - wrappers for OS-dependent functions that deal with threads, timers, mutexes.

NIC drivers - network interface card drivers.

NIC1, NIC2 - network interface card is a physical device which sends/receives frames to/from the EtherCAT network.

Remote Client - KPA EtherCAT Studio or another compatible application.

5.1. Modules

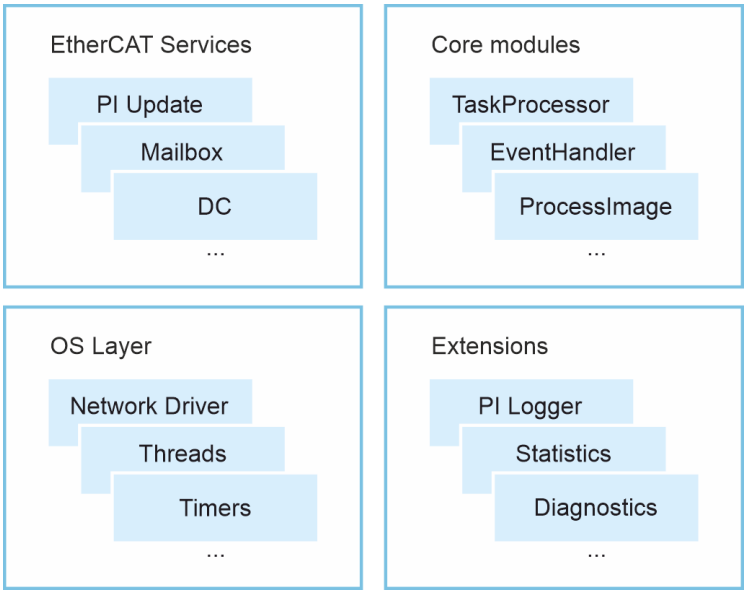

Figure 2. KPA EtherCAT Master 2 modules

Actually, all Master 2 functionalities are implemented as independent modules. Each module is isolated from others and performs a particular service (task/functionality: work with CoE, FoE, VoE, DC, S2S, PI Client, etc.). It allows to build a Master 2 core according to the user's needs and achieve optimal performance and functionality balance (e.g. using a license to activate Master 2 Basic class with adding to it DC and Hot-Connect functionalities).

5.2. Task scheduller

Each module, while executing a task, creates a request with a certain priority. The set of requests from different modules is passed to task scheduler which sorts all tasks by priority and initializes their execution.

5.3. Synchronization queue

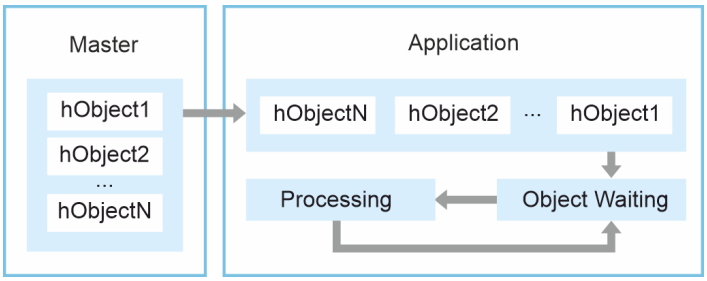

Figure 3. KPA EtherCAT Master 2 synchronization queue

Synchronization queue is a transport level for communication between Master 2 and user-application. While working, Master 2 creates and processes different kinds of objects. Handles of corresponding objects are put into synchronization queue (in order of their appearance, that is practically FIFO method - first IN first OUT: an object of an event that happens earlier will be put into synchronization queue earlier). User-application waits until some objects appear in the queue (commonly it is recommended to wait infinitely (till an object appears) or with zero time out (multiple check of objects existence without delays). After receiving the object's handle, the application may compare the received and expected handles and, depending on the comparison result, execute some actions. By completion of object process, the application goes back to waiting for a new object in the queue.

5.4. Common operation modes

There are three common modes supported by the Master 2:

1. **Asynchronous mode**

Application task (cycle), which implements control algorithms, and a Master 2 cycle, which updates Process Image, run independently. The whole data exchange is done through Process Image.

HI = PI update cycle

NR = Mailbox cycle

LO = Diagnostics cycle

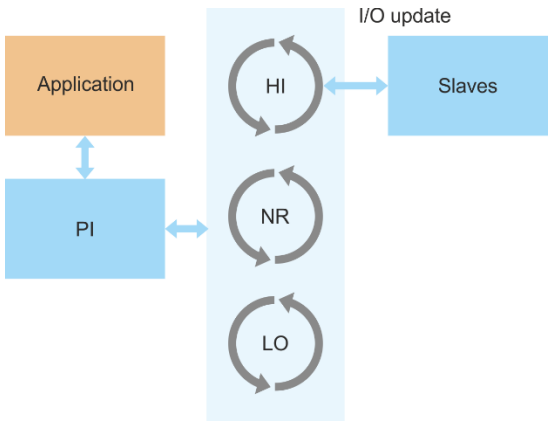

Figure 4. Asynchronous mode

2. **Synchronous 1 mode**

There is no separate application cycle in contrast to Asynchronous mode. All application algorithms are synchronized via Event Handler (EH).

HI = PI update cycle

NR = Mailbox cycle

LO = Diagnostics cycle

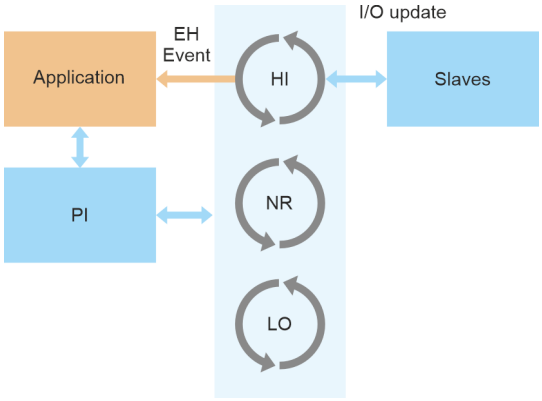

Figure 5. Synchronous 1 mode

3. **Synchronous 2 mode**

There is no separate application cycle in contrast to Asynchronous mode. All application algorithms are synchronized via Event Handler (EH). An application controls the start of cyclic tasks.

HI = PI update cycle

NR = Mailbox cycle

LO = Diagnostics cycle

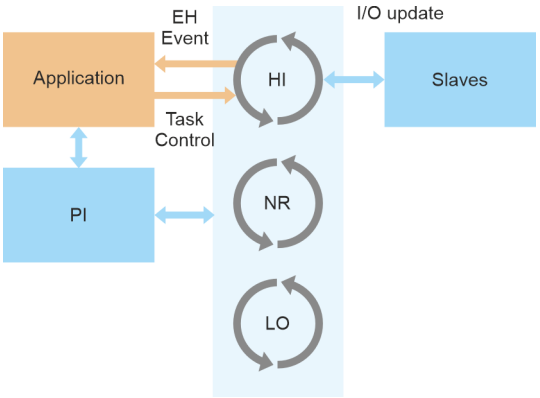

Figure 6. Synchronous 2 mode

6. KPA EtherCAT Master 2 getting started

6.1. Before starting

It is strongly recommended to **use only dynamic library linkage** to avoid library initialization problems. In other cases, **user application has to call EcatMkpaInit (NULL) routine** before any Master 2 API call and EcatMkpaDestroy() at final stage.

This way, it is necessary to load Master 2 Library (ecatmkpa.dll for Windows, ecatmkpa.rsl for INtime, RTXEcatKPAMaster.rtdll for RTX, etc.) before using Master 2 API. For example, for Windows you may load the Library by calling LoadLibrary function.

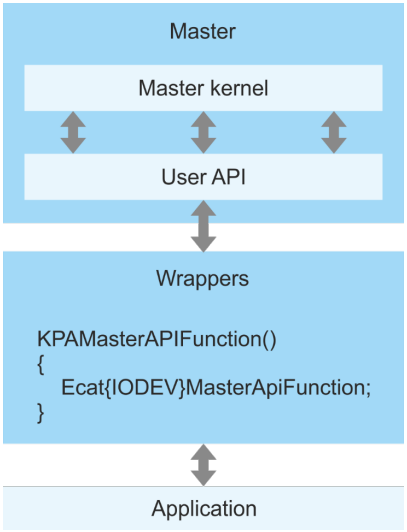

Figure 7. LoadLibrary Function

On the other hand, the usage of wrappers makes your application more platform-independent and allows simplifying the process of adapting your application to different OS.

6.2. Master 2 initialization

It requires configuring Master 2 once during user application start. Master 2 initialization procedure includes the following steps:

1. EcatMkpaInit(NULL)
2. Get a list of available Network interfaces
3. Create a unique identifier for Master that will be used to reference Master later
4. Connect chosen Network Interface to created Master, preliminary specifying expected operating mode
5. Operating mode without redundancy (using only one Network interface)
6. Operating mode with redundancy (using two Network interfaces)
7. Load ENI file
8. Initiate cyclic data exchange
9. Set time for auxiliary tasks (scanning of number of slaves on the bus and others)

6.3. Process Image

Process Image (PI) is a global Master storage that reflects Master state, input and output signals, cyclic frames, diagnostics, statistics, PI variables etc. Logically, it is divided into two separate parts, Inputs and Outputs. Inputs part is intended for input data like receiving frame, receiving signal from the bus etc. Outputs part is intended for output data to be sent to the bus. Also, Outputs can be used to control Master while it will report its states to Inputs. Access to Process Image is performed by PI Client. Each part of PI can be owned by only one client, that restricts a write access to PI. Only one client can write data to specified PI area, all others are disallowed.

The sample 07_ProcessImageDelivery.c shows how to create and use PI Client object to read from PI and write data to PI.

6.3.1. PI Client

PI Client is an object to provide read/write access, delivery data from PI to internal client's buffer, subscription to events and PI changes. It protects PI from writing with help of priority parameter. To configure a client with PI area, mapping is used. Any area of PI can be mapped for read, and the only one client is allowed to write to specified PI area. Write PI areas of different clients cannot intersect each other. Mapping is performed with help of API functions: EcatPIClientMap EcatPIClientMapCustom.

Client can be created either for reading or for writing. Single client doesn't support simultaneous usage of both operations. If requested PI area does not exist or cannot be written, map is requested, map fails. Map operation is asynchronous, the PI Client user should wait for synchronization queue for operation's completion. On update operation all mapped PI parts are passed into a single buffer that is delivered from PI to client at reading or from client to PI at writing. On attach PI Client allocates an array of buffers that are used for data delivery. The size of this array is set by EcatPIClientSetPoolSize function.

Writing PI

Writing data to PI is performed by setting values in PI Client buffer with EcatPIClientSetVariable or EcatPIClientSetDeliveryBuffer and subsequent call of EcatPIClientUpdate. EcatPIClientUpdate is asynchronous call so the user should wait for synchronization queue for its completion.

Reading PI

Reading data from PI could be organized in several ways. It may be updated manually by calling EcatPIClientUpdate and then EcatPIClientGetVariable or EcatPIClientGetDeliveryBuffer. Or the PI Client may be attached to write event by using EcatPIClientListenPI. In this case when another client writes data to specified PI area, this will schedule update of current client buffer. When the synchronization queue receives PI Client handle, the user can read data from it with help of EcatPIClientGetVariable or catPIClientGetDeliveryBuffer.

PI Client state machine

Here is a state machine of PI Client:

Figure 8. PI Client state machine

6.4. Implementation of Master 2 modes

6.4.1. Asynchronous mode

In Master 2 cyclic operation is initiated (and stopped) by functions **EcatStartCyclicOperationAsync** and **EcatStopCyclicOperationAsync** (for asynchronous mode). These functions are asynchronous and should be called with synchronization queue handle.

EcatStartCyclicOperationAsync can be called with **ECAT_MASTER_CYCLE_TIME_AUTO** as value for cycle timeout, and Master 2 will calculate cycle time automatically basing on cyclic tags' description in ENI.

6.4.2. Synchronous 1,2 modes

To start Master 2 in any synchronization mode, call **EcatStartCyclicOperationAsync**. In case of Synchronous 2 mode it is called with **ECAT_MASTER_START_EXTERNAL_CYCLE** as value for cycle timeout.

Process Image update control can be performed by writing PI variable "Task.<No>.Control". This variable is created in PI for each cyclic tag in ENI, and allows to initiate PI update by writing value to this variable.

In Master 2 cyclic exchange is controlled by events. Update of PI variable "Task.<No>.Control" leads to generation of ECAT_PIUPDATE_EVENT_CYCLIC_UPDATE_START(0), ECAT_PIUPDATE_EVENT_CYCLIC_UPDATE_START_ALL events. Notification about completion of updating is sent by events ECAT_PIUPDATE_EVENT_CYCLIC_UPDATE_DONE(0), ECAT_PIUPDATE_EVENT_CYCLIC_UPDATE_DONE_ALL. The caller can subscribe to these events to control over the start and finish of update.

7. Shutting down KPA EtherCAT Master 2

To complete Master 2 operation, do the following:

1. Stop cyclic data update
2. Release used Network interfaces
3. Free Master resource
4. Release used Master’s resources (used Master’s libraries)

8. Software packages

Due to support of wide range of platforms different software packages are available.

At present the set of platforms supported by the KPA EtherCAT Master 2 includes:

- Windows
- INtime
- RTX64
- Linux
- Xenomai
- QNX 6.x
- QNX 7.x
- VxWorks
- FreeRTOS
- ITRON
- Intewell

Any other operating system can be supported upon customer's request. In addition, we have ported KPA EtherCAT Master 2 to Windows XP, CE6/7, OnTime RTOS-32, PikeOS, RTAI, etc. but we do not actively support these operating systems.

9. KPA EtherCAT Master 2 is proven

This paper has discussed the KPA EtherCAT Master 2 architecture, distinctive features and core functions and revealed the advantages of the Master 2 implementation.

With the KPA EtherCAT Master 2 you are able to easily make the process of data exchange fast, secure, and effective. You will considerably reduce expenses as no special Master plug-in cards are required but a standard Ethernet card can be used. The platform independent source code allows making the Master 2 portable to any operating system with fewer efforts. Another crucial factor is the adaptation of Master 2 to any hardware platform implementing only the network adapter driver logic. The architecture provides the Master 2 scalability to fit the size of application, development and customization of each part keeping the functionality of the others untouched.

The KPA EtherCAT Master 2 provides you with the latest approach to enhancing the whole process of data exchange with the network.

For more detailed information on KPE EtherCAT Master 2, please, refer to documentation that is provided within the Master 2 Development Kit package.

10. Imprint and disclaimer

10.1. Trademarks

EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

straton® is a registered trademark of Ing. Punzenberger COPA-DATA GmbH, Austria.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

10.2. Disclaimer

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event, shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

10.3. Quality Management

Our quality management system meets ISO standards and covers all our products and services. All company processes, from a product order to technical support, are managed according to our quality management system.

10.4. Contacts

For more information about products and services, please visit company website: www.koenig-pa.de.
For getting technical support or solving issues arouse from the use of our products, there is a ticketing system in the Customer Portal where you can apply for assistance. Note that the Customer Portal is available for registered users only.
In urgent cases, you have the following options:

- Contact resellers in your country or region.
- Get assistance by phone: +49 9128 725 330, +49 9123 960 5796.
- Contact our Support Team at support@koenig-pa.de.

10.5. Mailing address

koenig-pa GmbH
Im Talesgrund 9a
91207 Lauf a.d. Pegnitz, Germany

10.6. Copyright