Jan. 2012

Multi-Axis Position Control by EtherCAT Real-Time Networking

Motion Control
and Power Solutions 2012

Ronen Boneh
Elmo Motion Control Ltd.

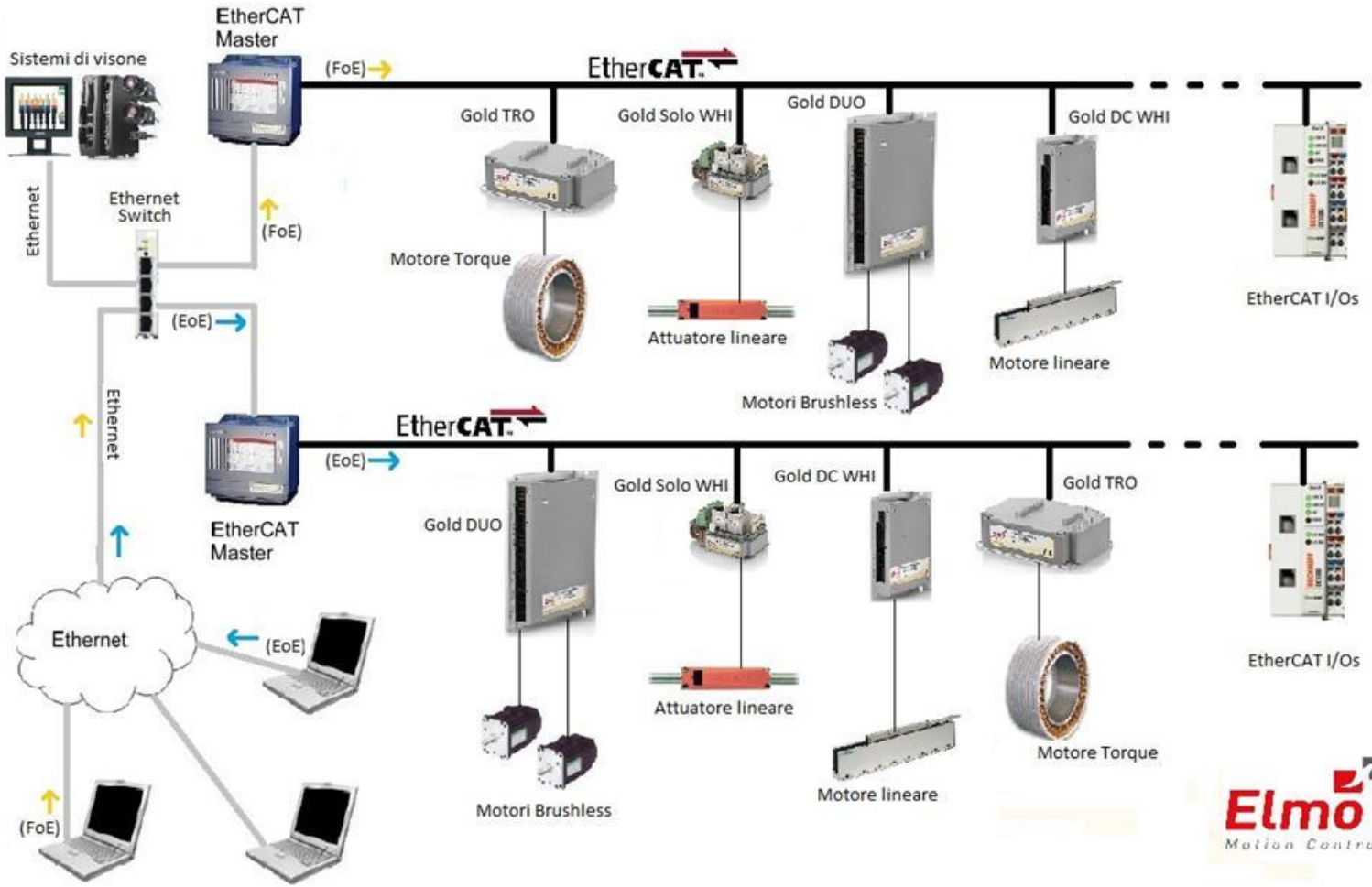
Lecture Topics

- EtherCAT Applications Overview
- The EtherCAT Control Approach
- The EtherCAT Data Processing
- Time Synchronization Requirements in Distributed Systems
- EtherCAT Distributed Clock Mechanism
- Back to EtherCAT and Motion Control
- Servo Drives, DS 402 Device Profiles and CoE
- Practical Real-Time Control Networks Inputs / Outputs Synchronization Requirements
- Summary

EtherCAT Applications Overview

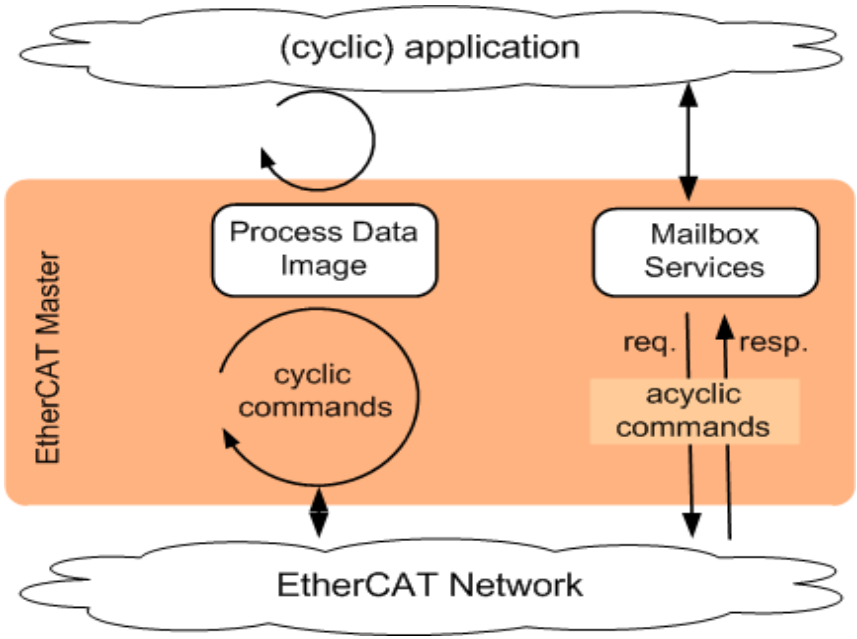
- **Typical Applications**
  - Renewable Energy manufacturing processes
  - Over 300 servo-controled movements coordinated with vision systems and I/O's
  - Wafer Handling Robots
  - XYZ Wafer inspection systems
  - Autonomuos robots
  - Militray applications for turret control and others...

EtherCAT Applications Overview



EtherCAT Control Approach

- **Cyclic Control**
  - Fixed Communication Cycle
- **Process data image is updated every cycle**
  - Process status is known to the control task cyclically
  - Process data compilation does not need to be changed → fast communication task with low resource requirements



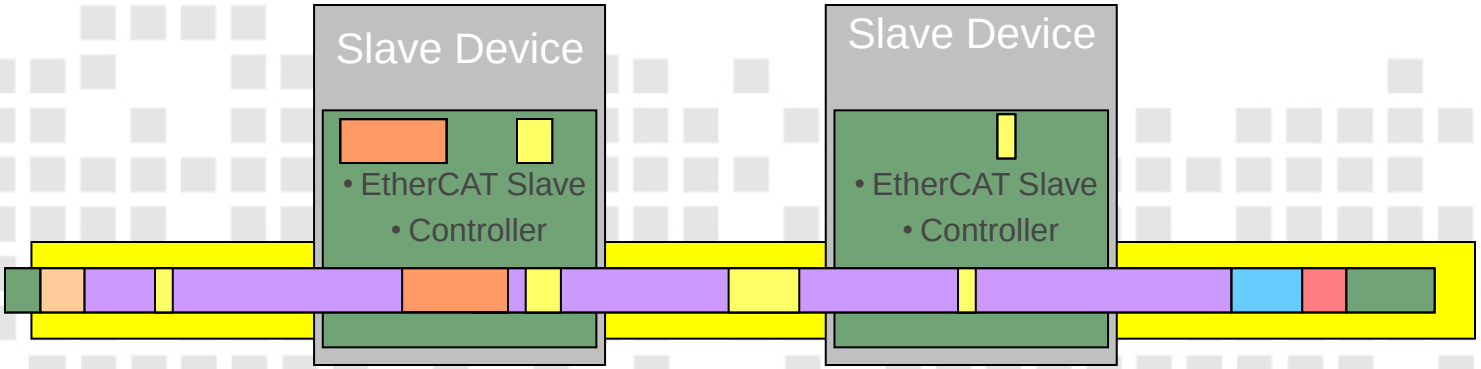
- Stable and fast synchronization handling
- No interference between the real-time process data and the background (mailbox) service
- Hot plug connection

EtherCAT Control Approach

- **What Type of Data is Shared on the Network?**
  - Usually, a control system needs to have, with periodic time intervals, the following:
    - **Inputs:** Latched Sensors Data such as Positions, Velocities, Currents, System Status, IO's etc.
    - **Outputs:** Control Law Commands, Trajectory Information, or Higher Drive Level Commands
  - The specific nature of the data being transferred on the network depends on the operation mode of the slave drive.
  - The terminology used for drive operation is "Device Profile."
  - A common standard for Drive Device Profiles is the "DS-402," CANopen Device Profile, and "CoE" – Can Over EtherCAT. **More on that , Later On …**

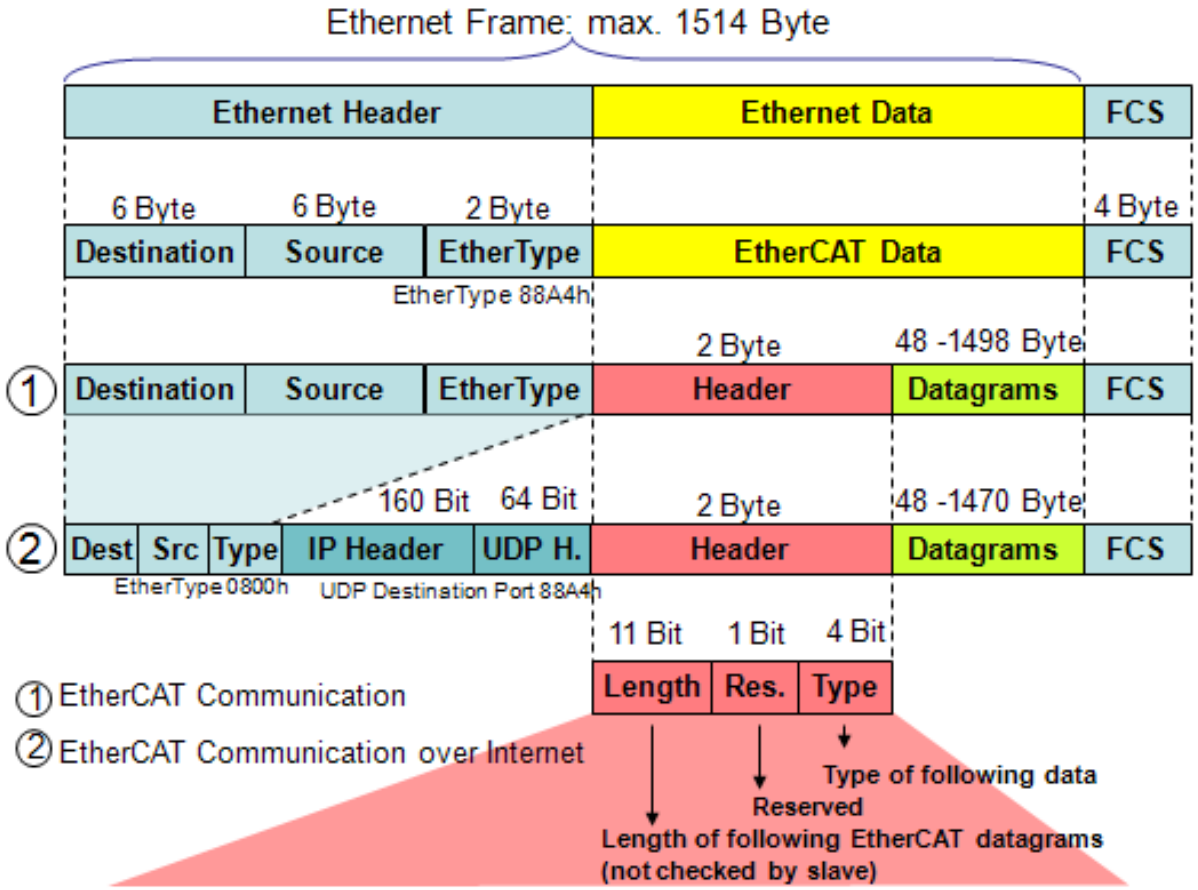
EtherCAT Data Processing

- **On-the-Fly Processing**
  - Process data is extracted and inserted on-the-fly
  - Process data size per slave practically unlimited (1 Bit…60 Kbyte, if needed using several frames)
  - Practical systems used for the process data are no more than a few dozen bytes per node per cycle
  - Support Dispersing cycle data: short cycle times for high demanding axes, and longer cycles for service axes and I/O update is possible…



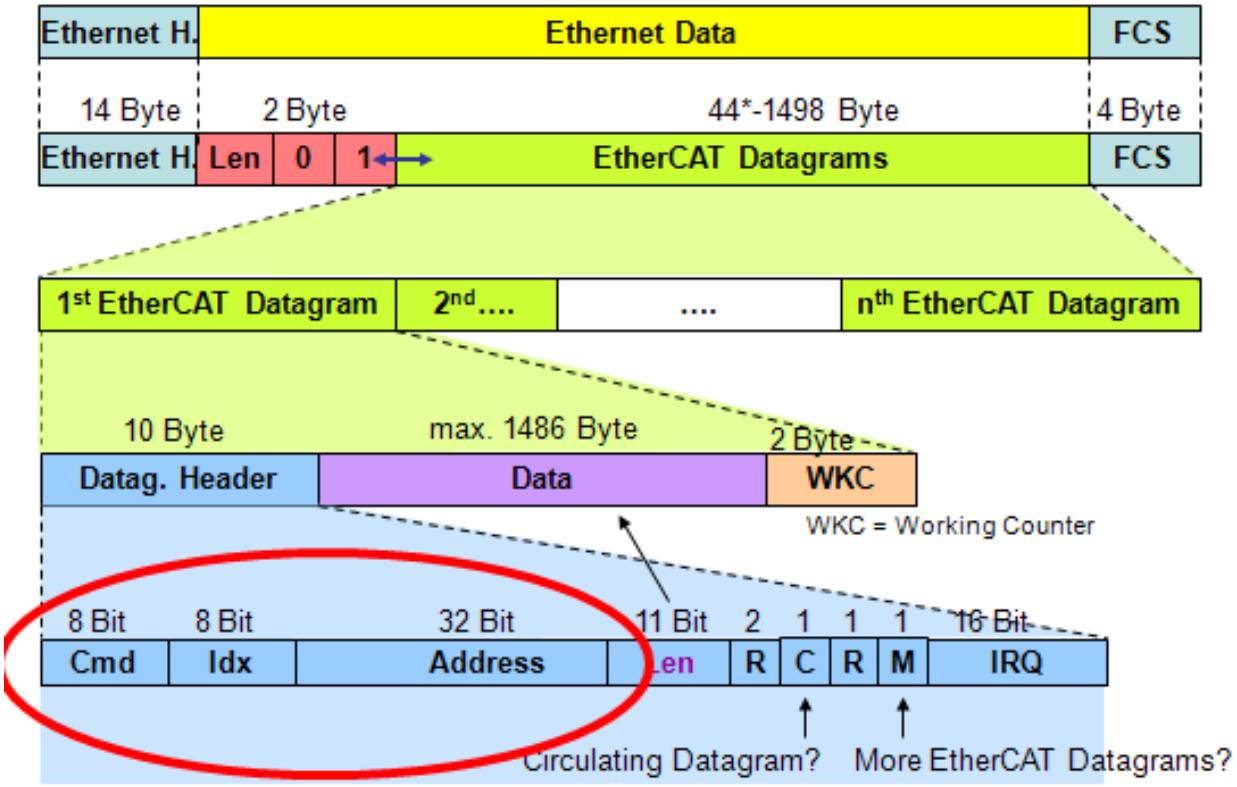
EtherCAT Data Processing

- EtherCAT Data Frame Structure



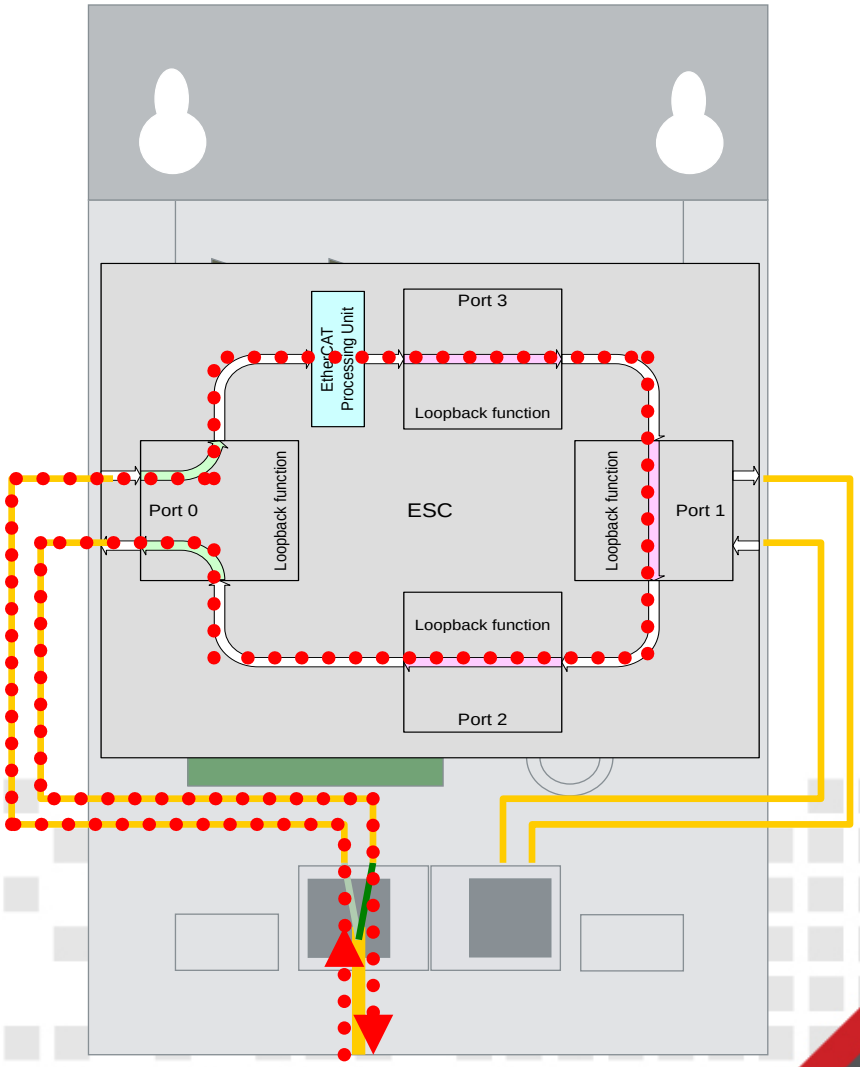
EtherCAT Data Processing

- EtherCAT Data Frame Structure



EtherCAT Data Processing

- **EtherCAT Frame Processing**
  - The real-time EtherCAT frame processing is done by the ESC: EtherCAT Slave Controller.



EtherCAT Distributed Clocks

- **Synchronization Requirements in Control**
  - Centralized Motion Control Systems use a single processor (one servo loop interrupt) for controlling all axes.
  - Jitter between axes is minimal and usually relates to h/w latching of peripherals (FPGA etc.). This is in the area of a few tens of nanoseconds.
  - Distributed Control topology comprises a remote master and multiple slaves, each with its own processing unit, synchronization and timings.
  - Modern Distributed Motion Control Systems mostly rely on Ethernet-based communication links.
  - Standard Ethernet is too reliant on software, thus its not deterministic enough for real-time motion control tasks.

EtherCAT Distributed Clocks

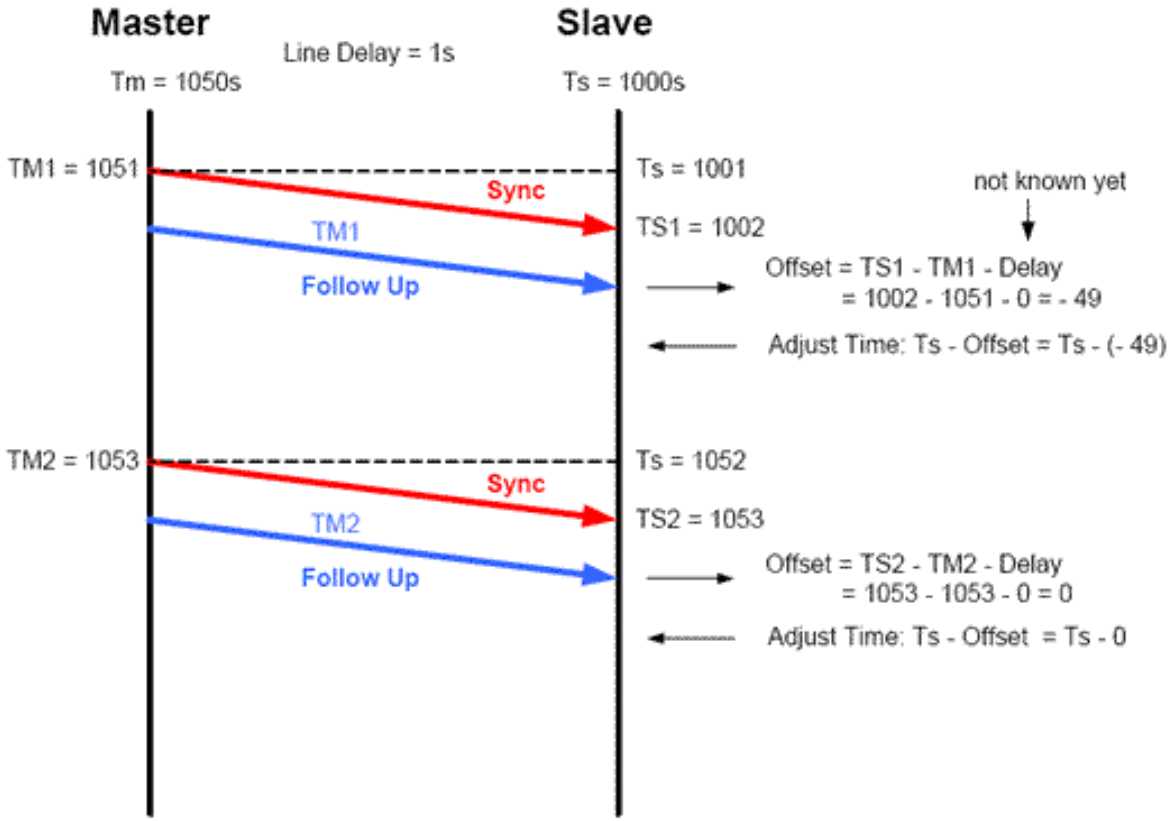
- **Synchronization Requirements in Control**
  - As an example, at 1 m/sec linear speed, 1 µsec shift in position latch lock is equivalent to 1 µm.
  - The velocity calculations will be more severely affected.
  - Modern digital control systems, running at 20 kHz servo loop rate or higher, latch a new position data every 50 µsec, and calculate the speed based on the position difference at the given time.
  - So in this case, 1 µsec jitter, is equivalent to 2% speed jitter error, which is much more than typically allowed in high-performance systems.
  - **High-end motion control systems need a synchronization level between all slaves that is far better than1 µsec!**
  - **SO, how is this achieved? …**

EtherCAT Distributed Clocks

- **Synchronization in a Distributed System**
  - The task of synchronizing multiple clocks in a distributed system is not uniquely specific to Motion Control, but is common in many computer and network systems.
  - There are a few methods to synchronize slave nodes over a network. One common way is the IEEE 1588 precision time protocol (PTP) (defined as early as 2002).
  - It is a technology for sharing clocks between distributed systems.
  - IEEE 1588 provides a distributed time base used to timestamp data with sub-microsecond precision.
  - The EtherCAT Distributed Clock (DC) uses the same concept of distributed time base.

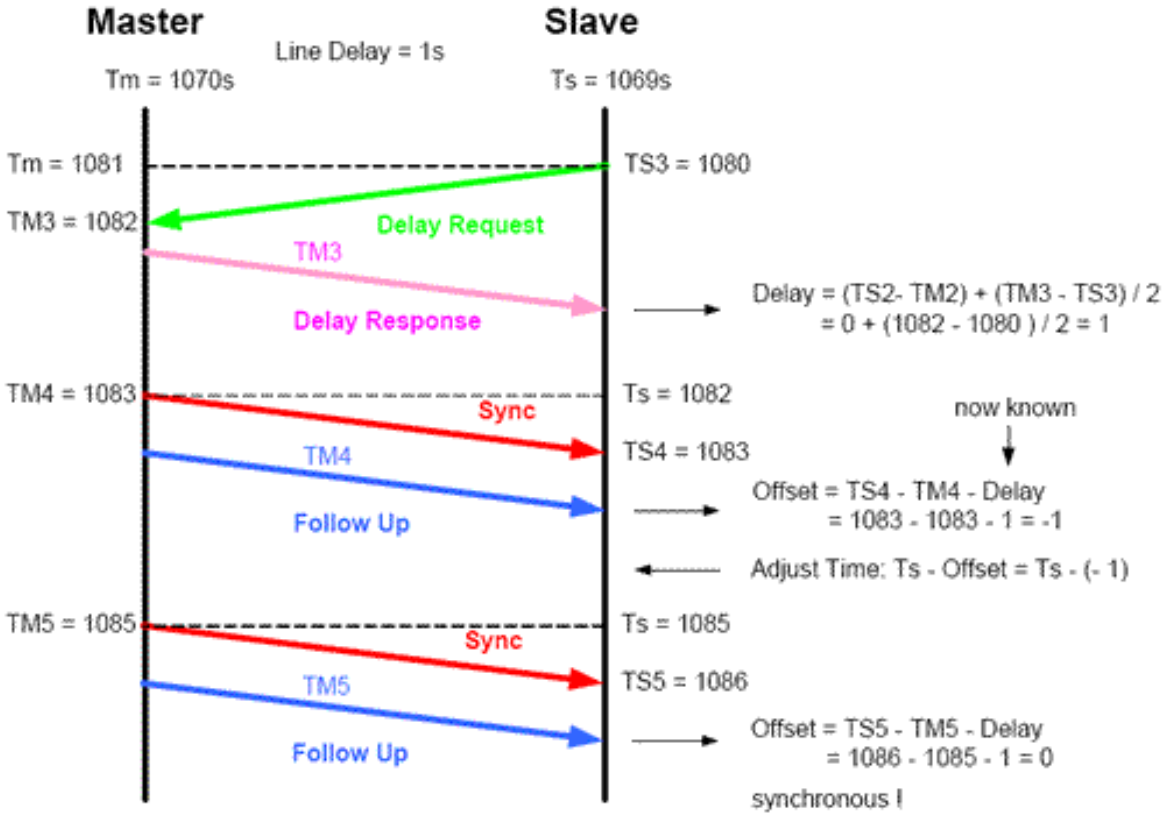
EtherCAT Distributed Clocks

- **Synchronization in a Distributed System**
  - Offset Measurement …



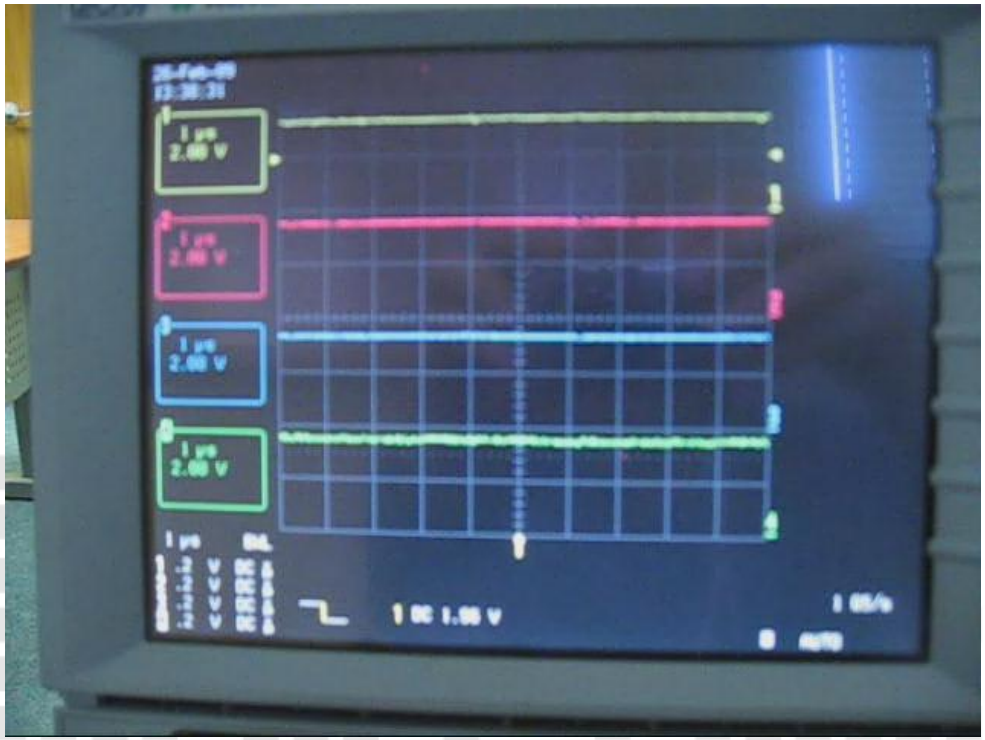
EtherCAT Distributed Clocks

- **Synchronization in a Distributed System**
  - Delay Measurement …



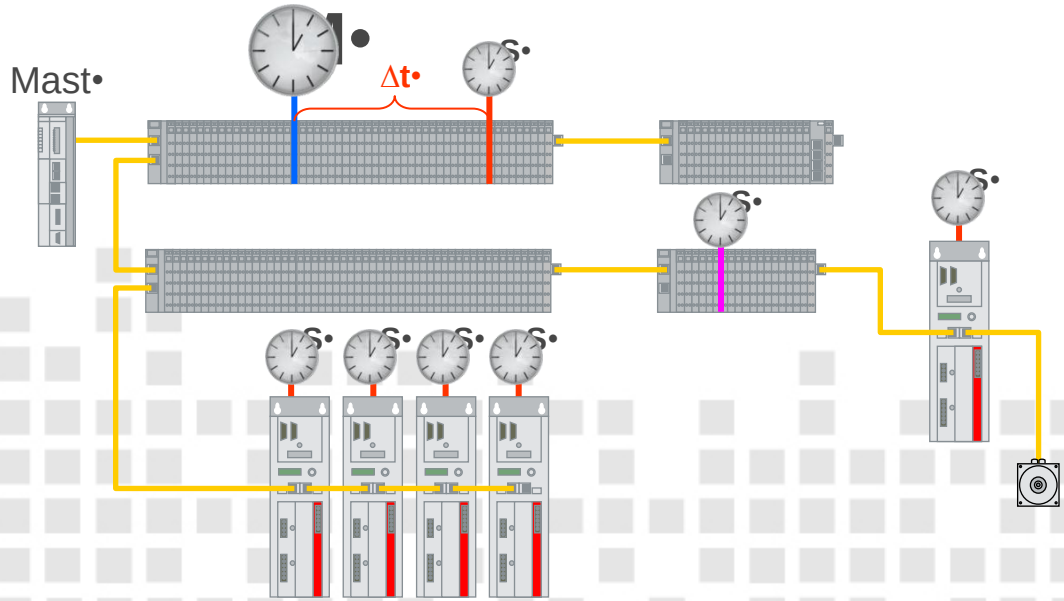
EtherCAT Distributed Clocks

- Synchronization in a Distributed System
  - It actually works …



EtherCAT Distributed Clocks

- **Purpose of Distributed Clocks in EtherCAT**
  - With the Distributed Clocks Mechanism (DC), Precise Synchronization (<< 1 µs!) can be achieved between slaves within a network, by exact adjustment of internal node system time.

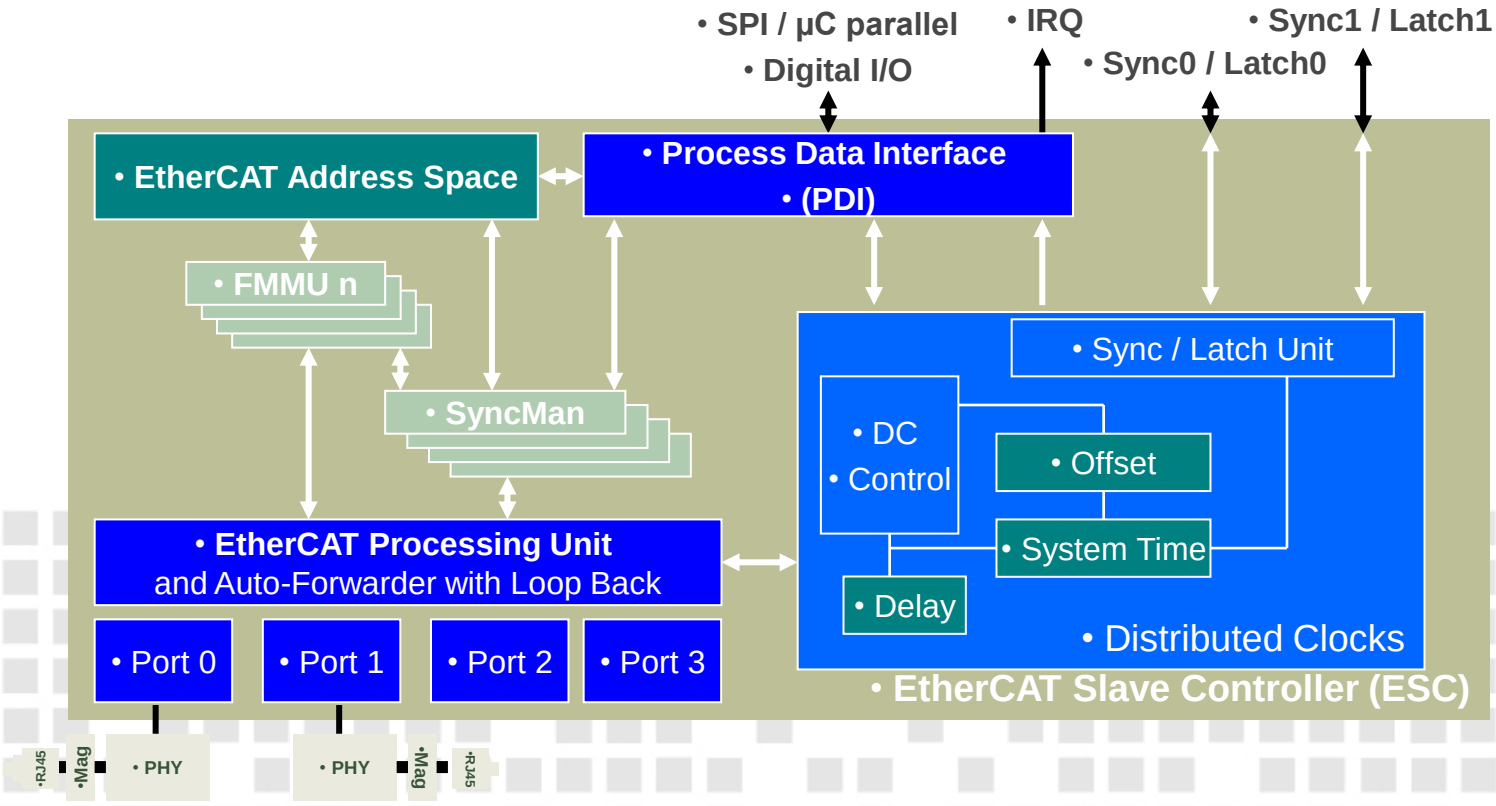


EtherCAT Distributed Clocks

- **Purpose of Distributed Clocks in EtherCAT**
  - Synchronization of EtherCAT devices
  - Definition of a System Time
    - Beginning on January 1, 2000 at 0:00h
    - Base unit is 1 ns
    - 64 bit value (enough for over 500 years)
    - Lower 32 bits spans over 4.2 seconds
      - Normally enough for communication and time stamping
  - Definition of a Reference Clock
    - One EtherCAT slave will be used as a Reference Clock
    - Reference Clock distributes its clock cyclically
    - Reference Clock adjustable from a "global" Reference Clock – IEEE 1588

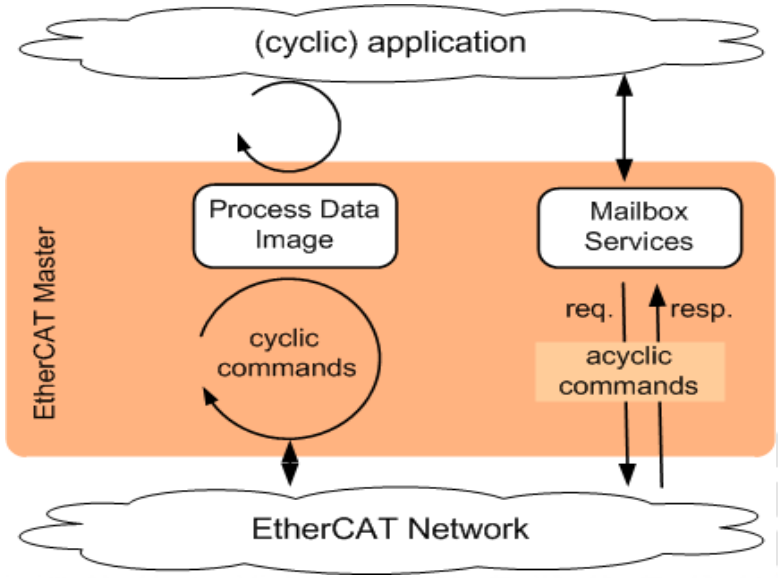
The EtherCAT Slave Controller

- The ESC, EtherCAT Slave Controller is responsible for all this …



Back to the Motion Control …

- **So, we understand how data is transferred and synchronized, but what does that have to do with the Motion Control System?**
  - Remember the Drive Device Profiles …
  - CANopen/DS402 …
  - "CoE" – Can Over EtherCAT
  - **More about that, now …**



Servo Drive Device Profile

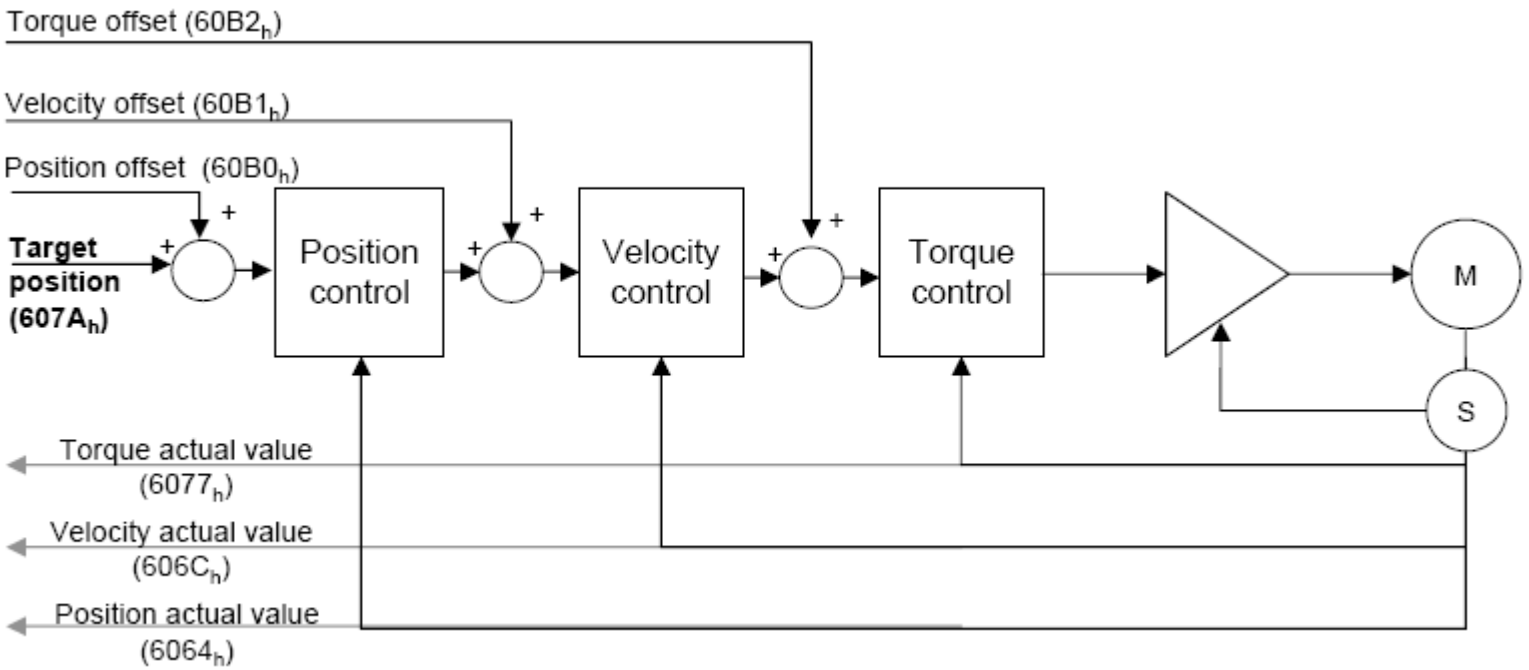
- **The CANopen DS402 and CoE**
  - The CAN in AUTOMATION (CiA) Group, established the DS402 ("DS" stands for Draft Standard) for "Drives and Motion Control Device Profile."
  - The aim of this standard is to provide a common platform that defines the general behavior and operation modes that should be supported by drive manufacturers, to enable as much as possible, interoperability between masters and slaves within motion control systems.
  - It was created for the "old" CAN technology, but is widely adopted by the ETG and many drive manufacturers, with the CoE … CAN Over EtherCAT.

DS402 Drive Device Profile

- **The CANopen DS402 and CoE**
  - The DS402 device profile defines several modes of operation, including:
    - Profile position mode
    - Homing modes
    - Interpolated position mode,
    - Profile velocity mode
    - Profile torque mode
    - Velocity mode
    - Cyclic synchronous position mode
    - Cyclic synchronous velocity mode
    - Cyclic synchronous torque mode

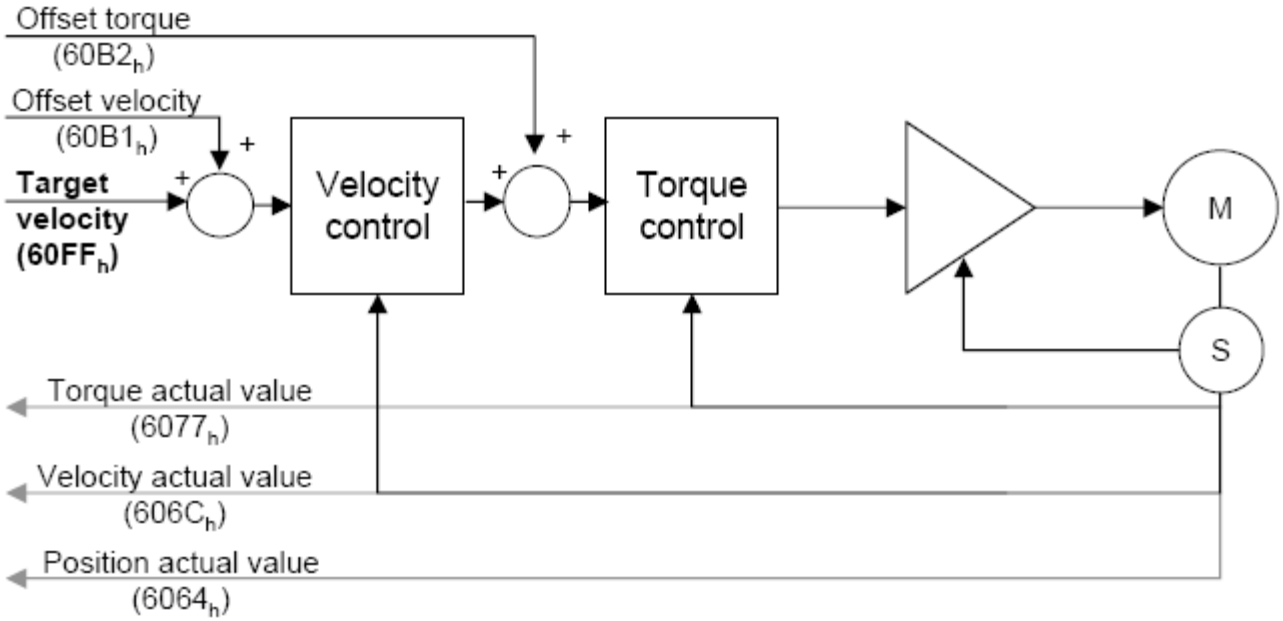
DS402 Drive Device Profile

- Cyclic Synchronous Position Mode



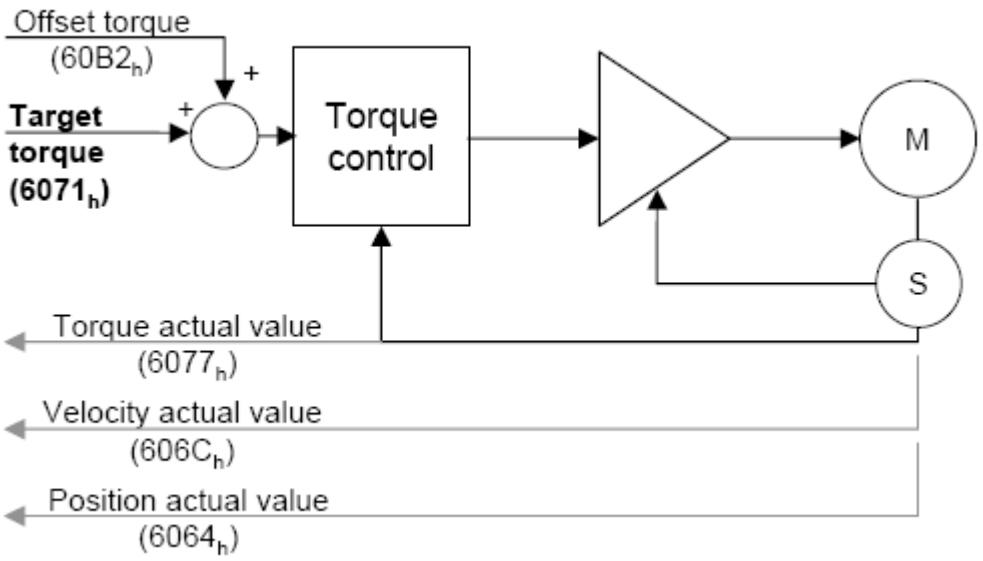
DS402 Drive Device Profile

- Cyclic Synchronous Velocity Mode



DS402 Drive Device Profile

- Cyclic Synchronous Torque Mode

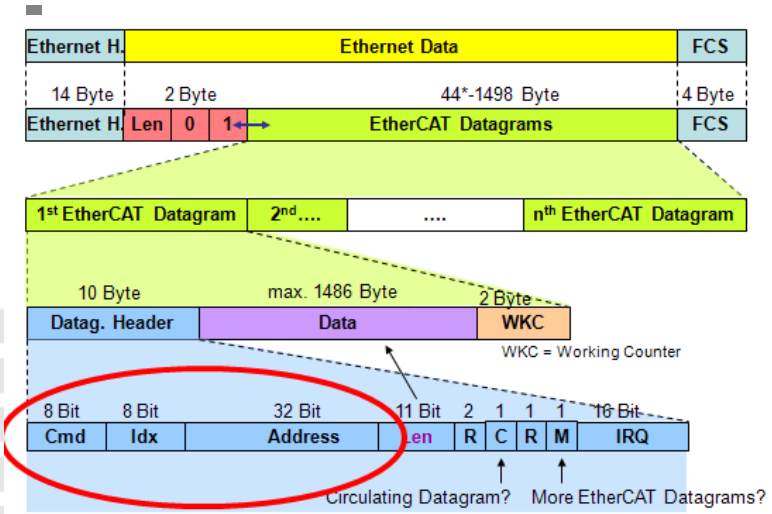


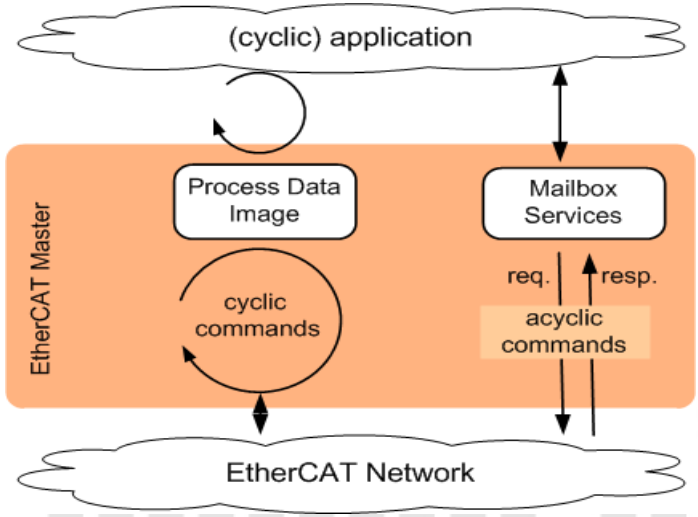
Real-Time Network Inputs/Outputs

- **So, the data we need to swap between the Master Controller and our Drives, depends on the "Device Profile" and "Operation Mode" used. It is generally comprised of:**
  - **Outputs (from the Master to the Drive):** Device Operation Mode Request (CW/OM), Target Position, Target Velocity, Target Torque, Various Offsets, etc.
  - **Inputs (from the Drive to the Master):** Actual Device Operation Mode and Status (SW), Actual Position, Actual Velocity, Actual Torque, etc.
  - These Periodic or Cyclic Data Objects, are also known as PDO's — Process Data Objects.

Real-Time Network Inputs/Outputs

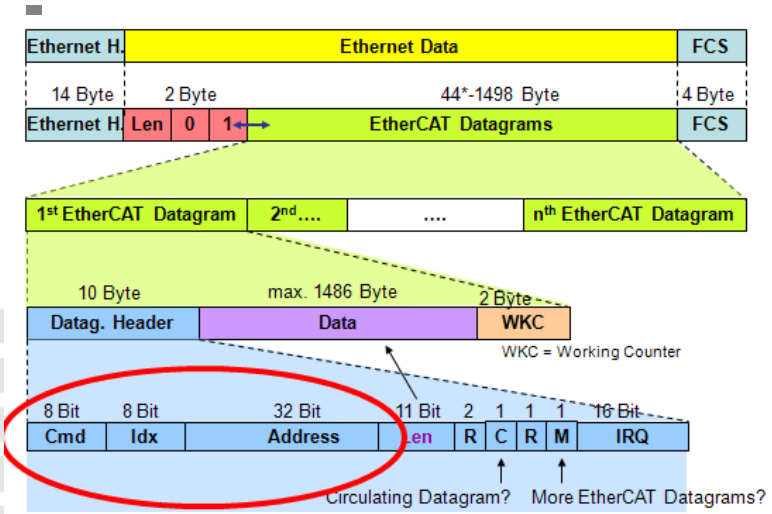
- **Process Data and EtherCAT Payloads …**
  - Each PDO has an "Address entry" within the Device "Object Dictionary," and is part of the Data Payload carried out over the EtherCAT Datagram Message …

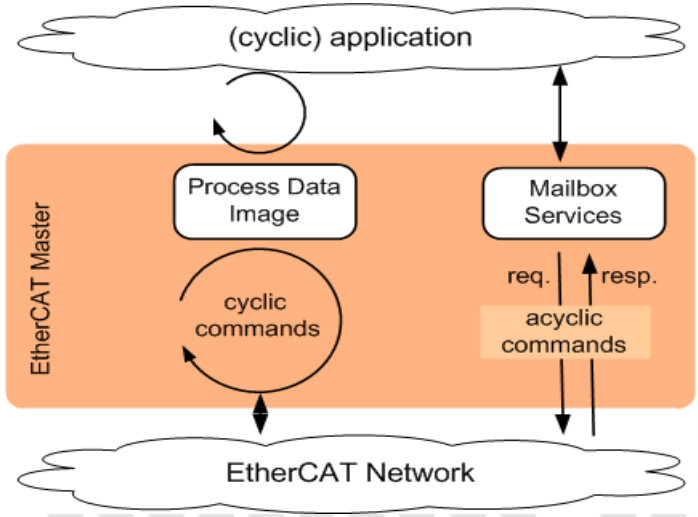




Real-Time Network Inputs/Outputs

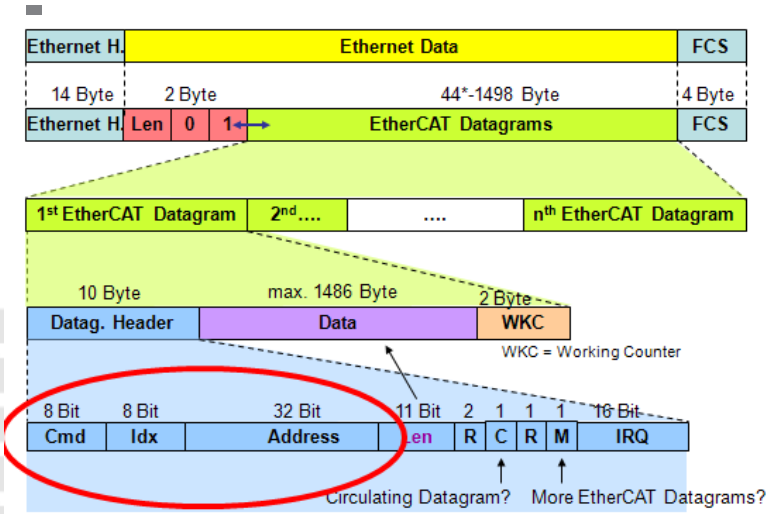
- **Process Data and EtherCAT Payloads …**
  - Each PDO has an "Address entry" within the Device "Object Dictionary," and is part of the Data Payload carried out over the EtherCAT Datagram Message …

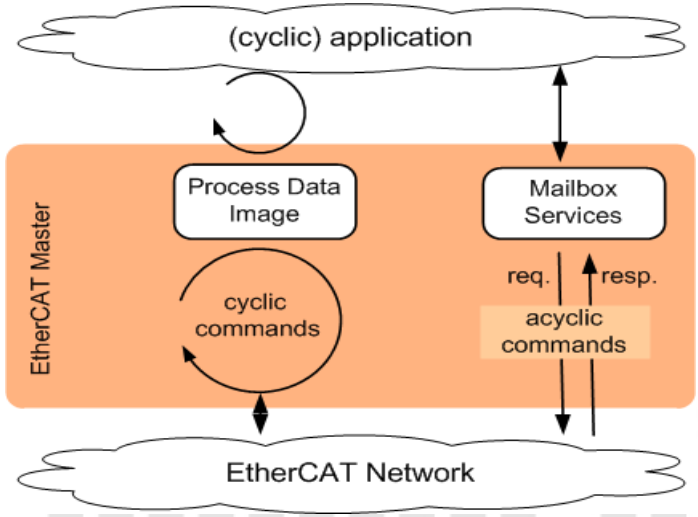




Real-Time Network Inputs/Outputs

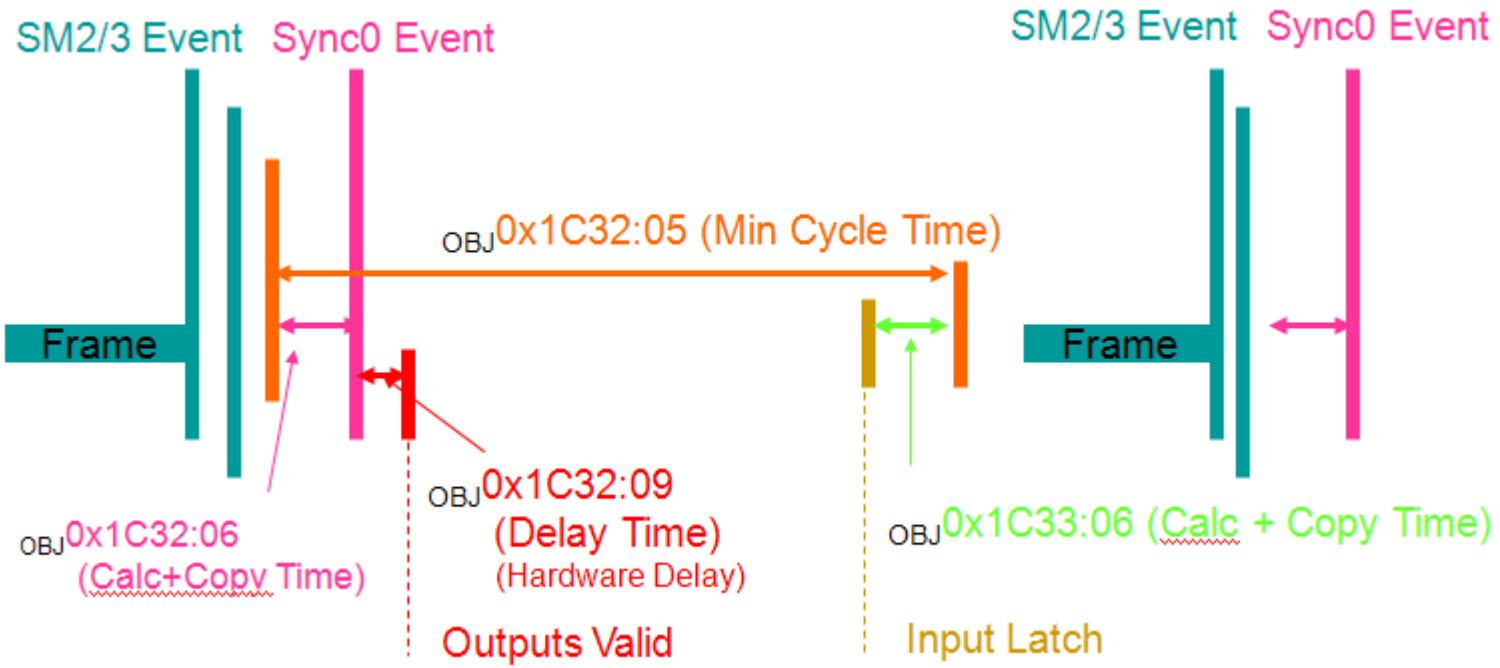
- **Process Data and EtherCAT Payloads …**
  - Each PDO has an "Address entry" within the Device "Object Dictionary," and is part of the Data Payload carried out over the EtherCAT Datagram Message …





Real-Time Network Inputs/Outputs

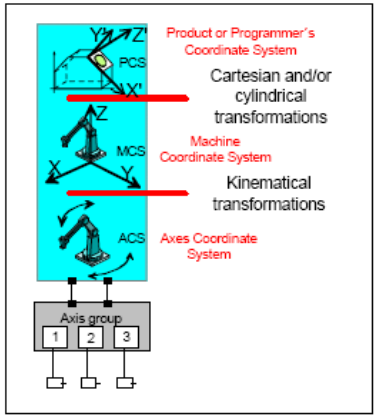
- Process Data Synchronization in EtherCAT

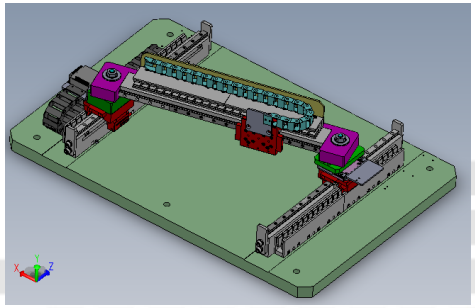


Real Time Network Inputs/Outputs

Practical Synchronization considerations …

- Systems and drives working in the "relatively simple" "Profile Position" or "Profile Velocity" Modes, don't really need strict synchronization.
- Systems aiming at synchronizing multiple axes Motion Trajectories, like pick-and-place machines, 3D robots etc., will require a high-level of synchronization, with cycle times in the area of ~1 millisecond.
- The most demanding systems are those that need to close the servo loops over the network. The highest level of synchronization and cycle update rates are needed – typically ~100 µsec or less.





Summary

- EtherCAT is spreading, at a much faster rate then any other filed bus, and commonly adopted for a variety of applications by many vendors.
- The strength of EtherCAT synchronization techniques allows it to be compatible with both low and high demanding applications.
- Combined with the correct implementation of both the network protocol, and a proper device profile, a true Distributed Motion control can now be achieved.

For more information, contact us at:

 info@elmomc.com

 +972 3 929 2300

Elmo For Better Results

One Solution, Any Application

Elmo
Motion Control