



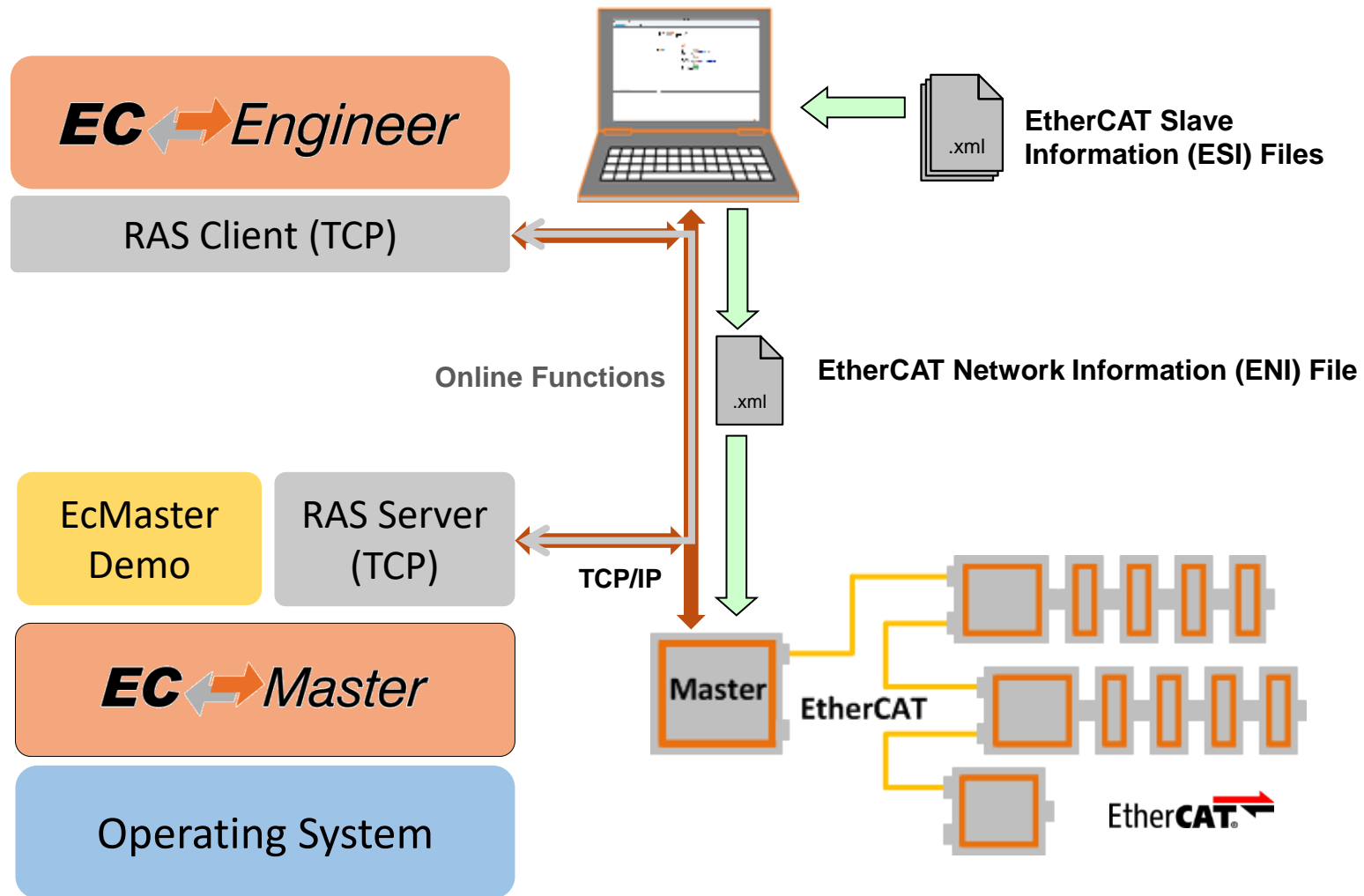
EC-Master Quick Start Guide

Setup an EtherCAT network with acontis products

Content

- Install EC-Master
- Run EcMasterDemo without configuration (ENI file)
- Create network configuration with EC-Engineer
- Run EcMasterDemo with configuration (ENI file)
- Online diagnosis of network with EC-Engineer
- EC-Master Application Programming Interface
- EC-Master Demo Application Overview
- Next steps

EtherCAT System Architecture



- Install the EC-Master from the compressed file
- Optionally extract add-ons and Real-time Ethernet drivers into the installation folder
- Follow operating system-specific steps described in chapter “Platform and Operating Systems (OS)” of the user manual

Run EcMasterDemo without ENI file

- Connect the EtherCAT slave(s)
- Check which network adapter is used for EtherCAT, e.g. eth1
- Run EcMasterDemo on Windows
 - `cd Bin\Windows\x64`
 - `EcMasterDemo.exe -ndis <ip-address> 1 -v 3 -b 4000`
- Run EcMasterDemo on Linux
 - `cd Bin/Linux/x64`
 - `./EcMasterDemo -sockraw eth1 -v 3 -b 4000`
- Run EcMasterDemo on QNX
 - `ifconfig en1 destroy`
 - `cd Bin/QNX71/x64`
 - `LD_LIBRARY_PATH=. ./EcMasterDemo -i8254x 1 1 -v 3 -b 4000`

For other operating systems, please follow steps described in chapter
“Platform and Operating Systems (OS)” of the user manual

Run EcMasterDemo without ENI file

- Master commands all slaves to PREOP state:

```
0000003094: Line Crossed.....: no
0000003094: Line Crossed Flags...: 0x0
0000003094: Cfg Station Address.: 0x03e9 (1001)
0000003094: *****
0000003094: Slave ID.....: 0x00000001
0000003094: Bus Index.....: 1
0000003094: Bus AutoInc Address.: 0xffff
0000003094: Bus Station Address.: 0x03ea (1002)
0000003094: Bus Alias Address...: 0x000d ( 13)
0000003094: Vendor ID.....: 0x00000002 = Beckhoff Automation GmbH
0000003094: Product Code.....: 0x07D43052 = EL2004
0000003094: Revision.....: 0x00100000   Serial Number: 0
0000003094: ESC Type.....: Beckhoff ET1200 (0x12)   Revision: 0   Build: 2
0000003094: Connection at Port A: yes (to 0x00000000)
0000003094: Connection at Port D: no (to 0xFFFFFFFF)
0000003094: Connection at Port B: no (to 0xFFFFFFFF)
0000003094: Connection at Port C: no (to 0xFFFFFFFF)
0000003094: Line Crossed.....: no
0000003094: Line Crossed Flags...: 0x0
0000003094: Cfg Station Address.: 0x03ea (1002)
0000003094: PD OUT   Byte.Bit offset: 0.0   Size: 4 bits
0000003094: *****
0000003101: Master state changed from <UNKNOWN> to <INIT>
0000003127: Master state changed from <INIT> to <PREOP>
```

EC  ***Engineer***

Generate bus configuration with EC-Engineer

EC-Engineer Operating Modes

Offline

Create network configuration
with ESI files

EC  **Engineer**



Online Local

Scan network connected to
Local system

EC  **Engineer**
EC  **Master**

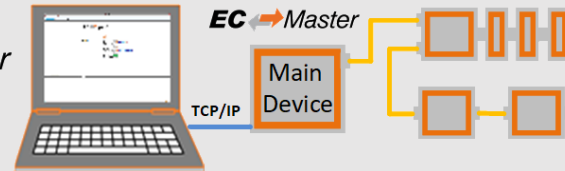


Online Remote

Scan network connected to
Remote system

EC  **Engineer**

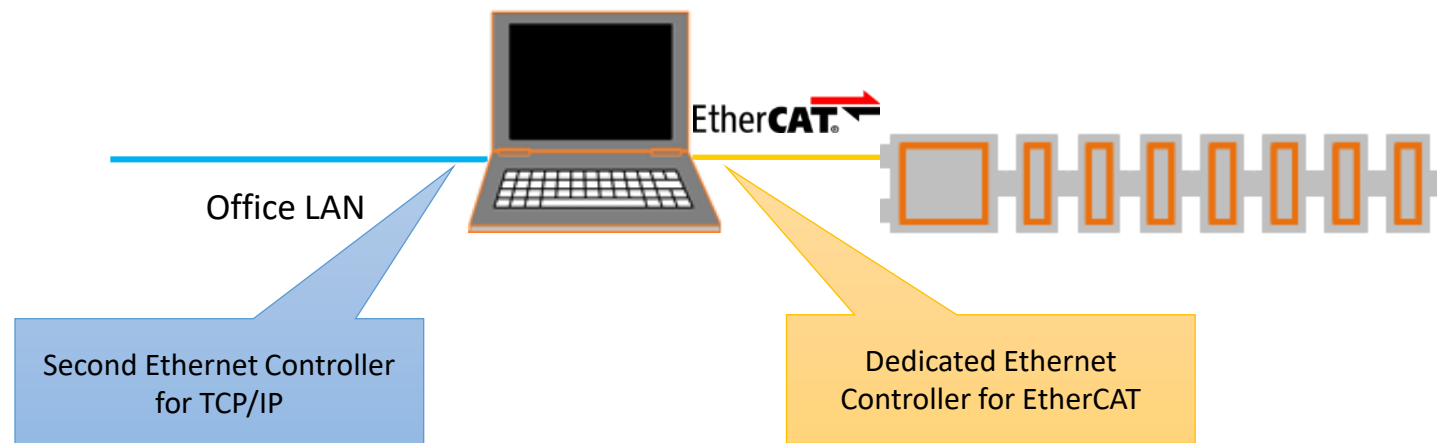
EC  **Master**



Generate bus configuration with EC-Engineer

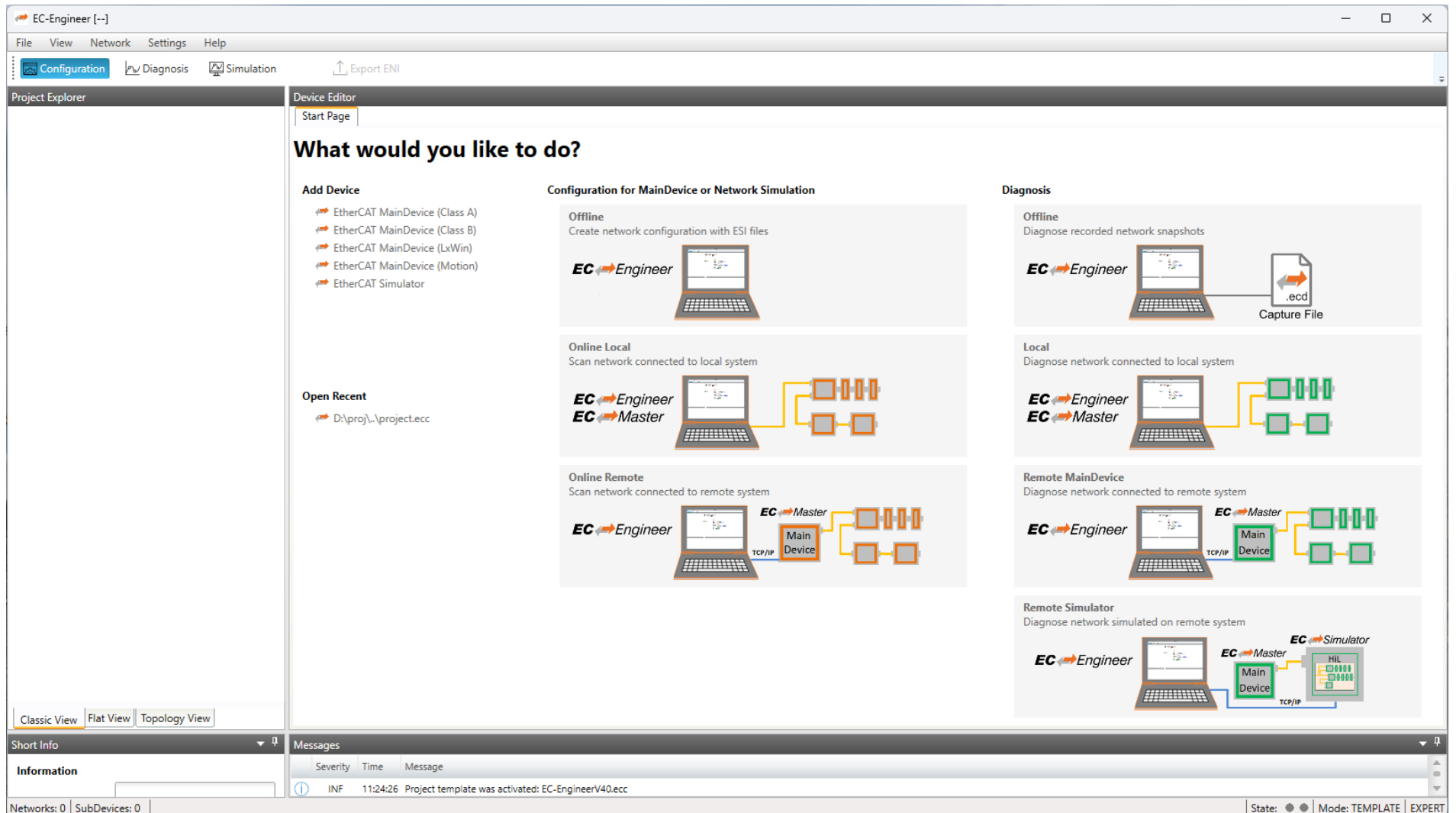
Step 1: Connect EtherCAT Slaves

- EC-Engineer comes with an integrated EtherCAT master for scanning the connected EtherCAT slaves
- Every Ethernet Network Interface with a valid Windows driver can be used
- **Warning:** Do not connect any EtherCAT slaves to your Office LAN
- A second, dedicated Network Interface for EtherCAT is recommended



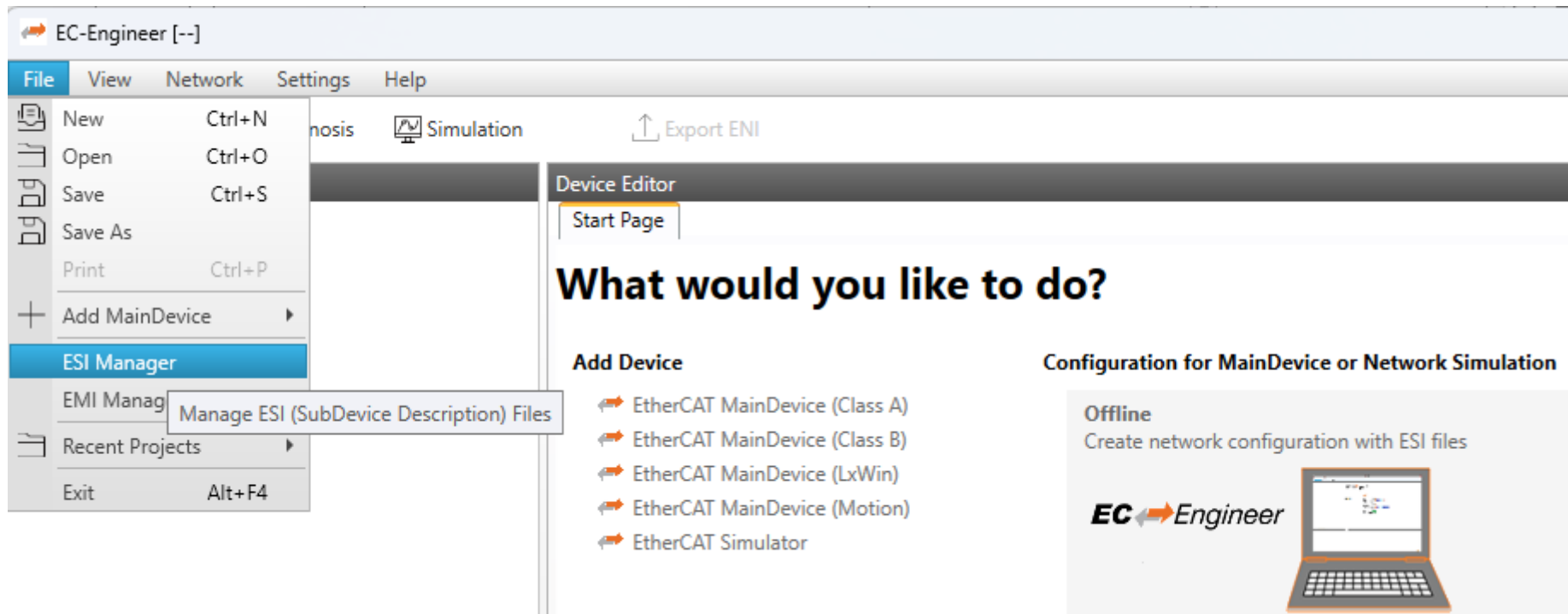
Generate bus configuration with EC-Engineer

Step 2: Install and start EC-Engineer



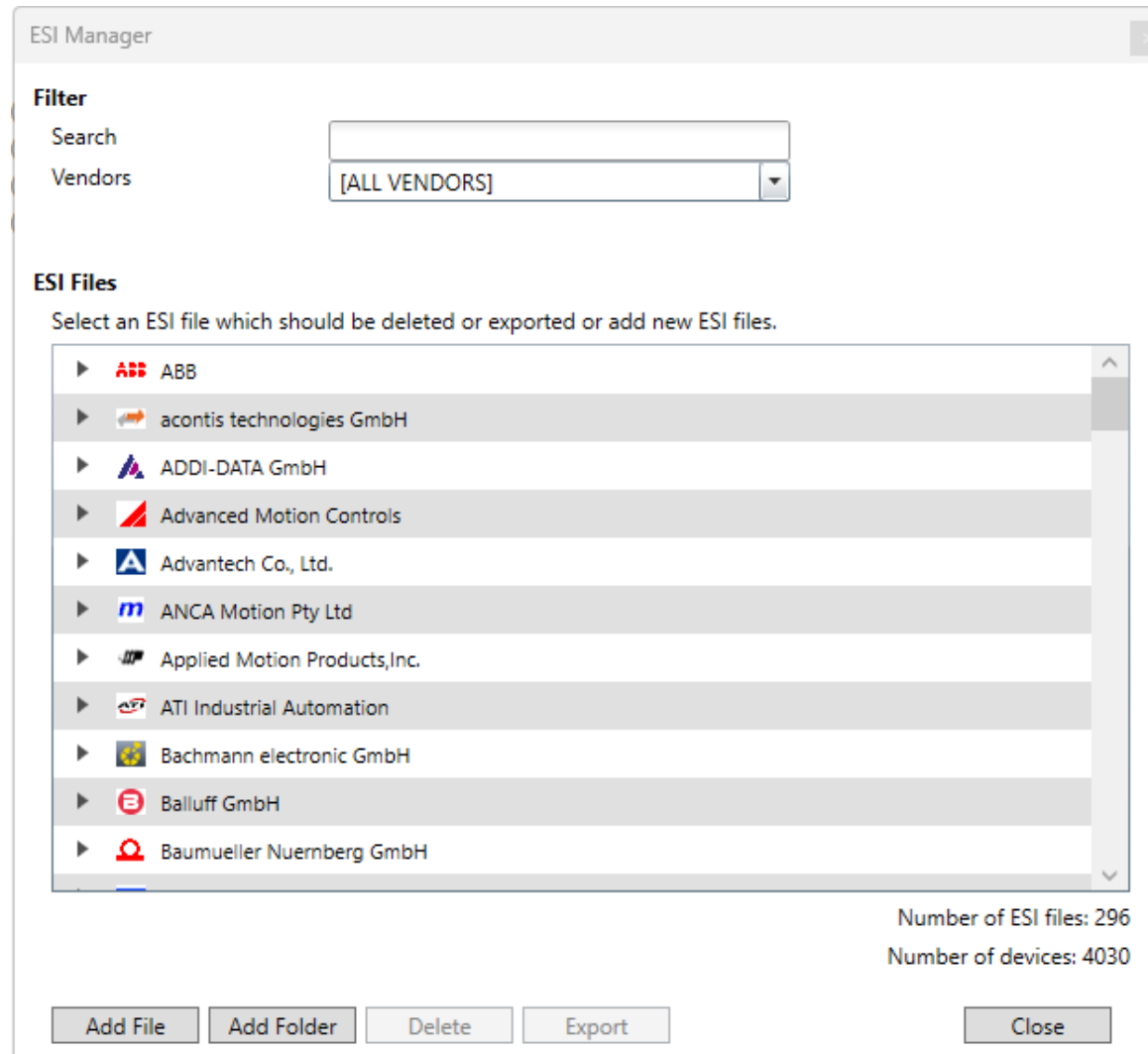
Generate bus configuration with EC-Engineer

Step 3: Open ESI Manager



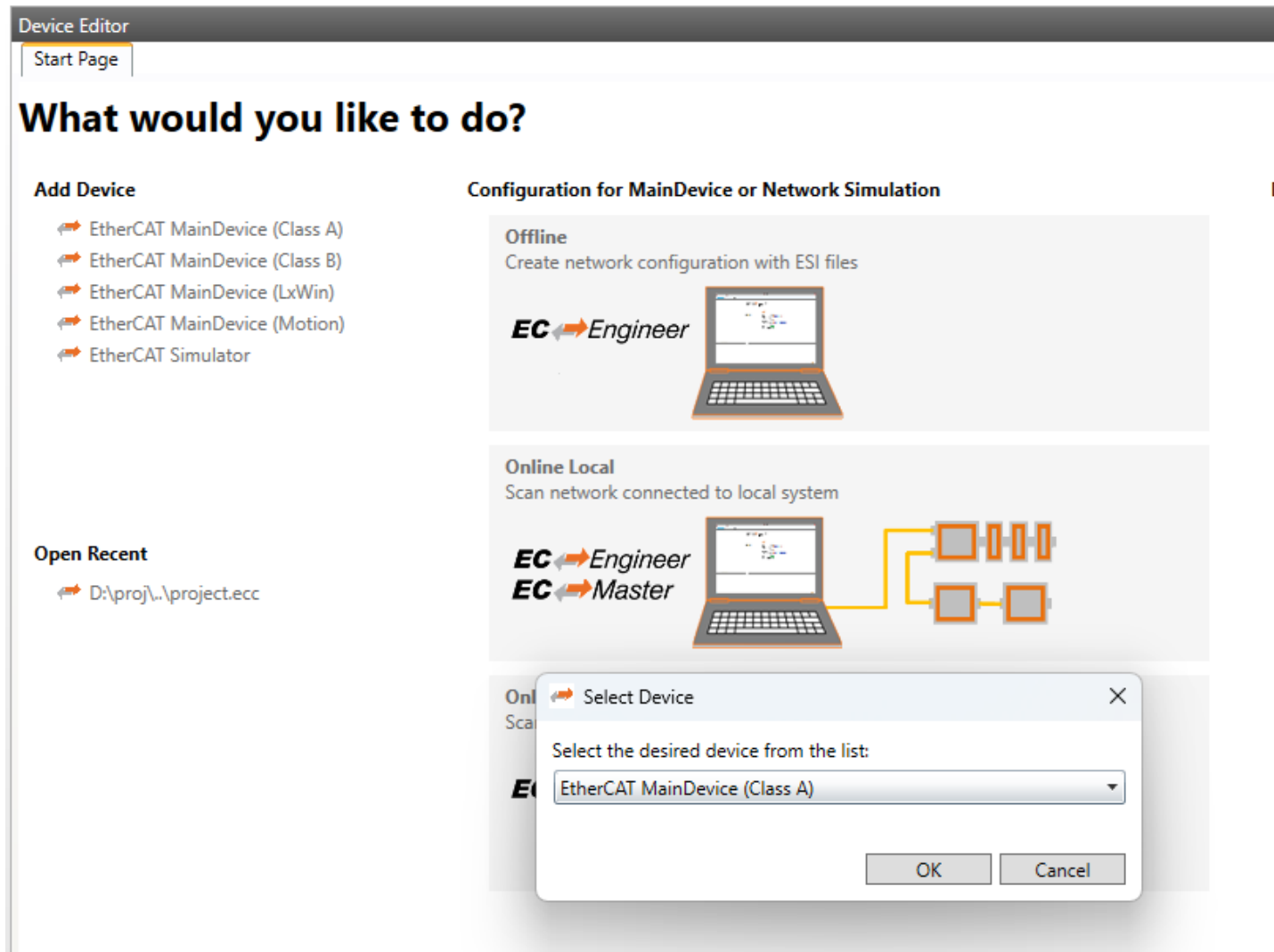
Generate bus configuration with EC-Engineer

Step 4: Add the appropriate ESI File



Generate bus configuration with EC-Engineer

Step 5: Select “Online Configuration” and “EtherCAT Master Unit (Class A)”



Generate bus configuration with EC-Engineer

Step 6: Select a Cycle Time of 4000 us

Device Editor

General

General

Unit Name: Class-A MainDevice

Cycle Time [us]: 1000

Source MAC address:

Local system

Link Layer: Ndis

Network Adapter: Dock (Realtek USB GbE Family Controller)

Requested MainDevice State: Init

Diagnosis Mode: Cycle Time: 4000

Diagnosis Mode: DCM: Off

Select

Define Cycle Time for
integrated EC-Master

Define Cycle Time for
EC-Master controller

Generate bus configuration with EC-Engineer

Step 7: Choose network adapter from list and press “Select”

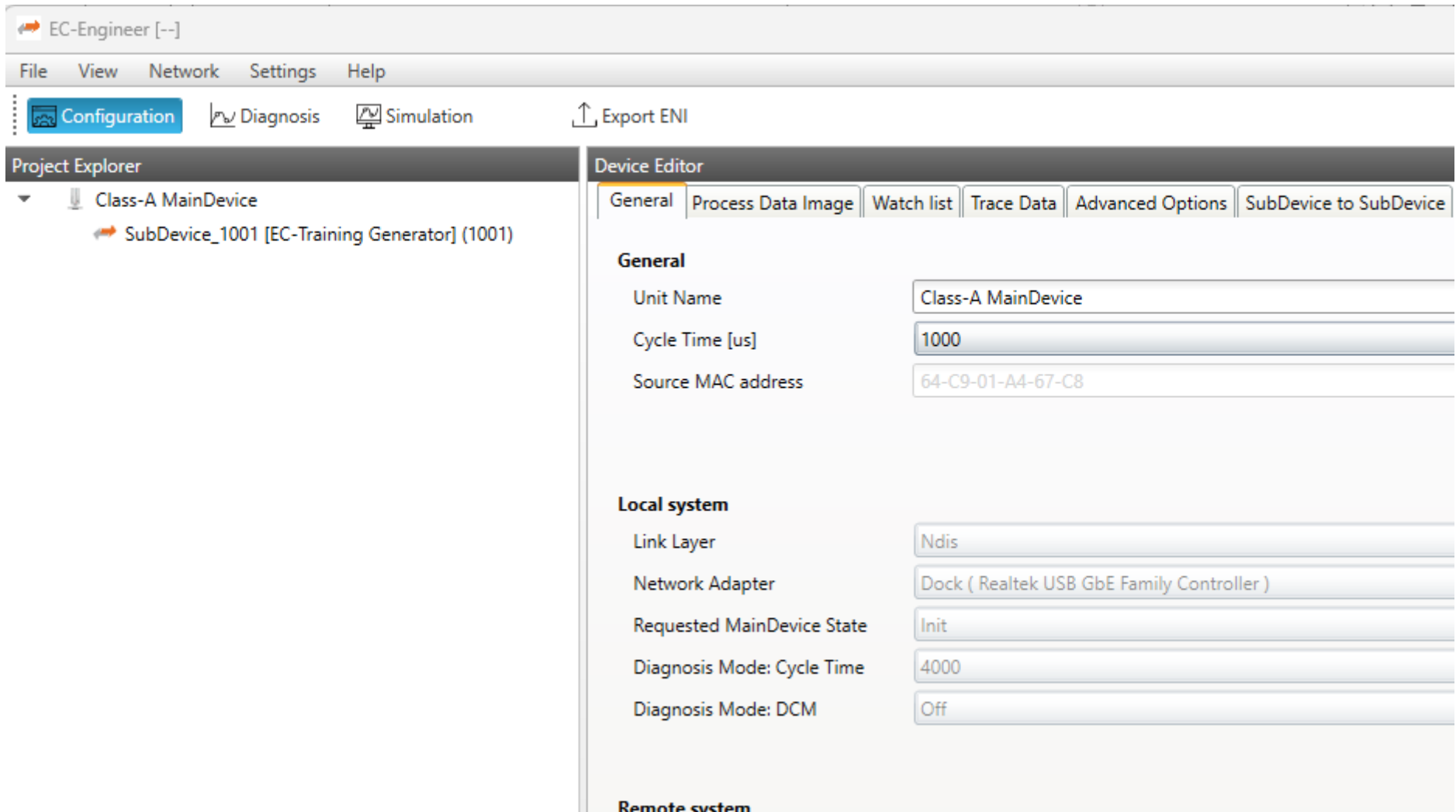
EC ↔ *Engineer*

Local system	
Link Layer	Ndis
Network Adapter	Dock (Realtek USB GbE Family Controller)
Requested MainDevice State	Init
Diagnosis Mode: Cycle Time	4000
Diagnosis Mode: DCM	Off
<div>Select</div>	

Press “Select”

Generate bus configuration with EC-Engineer

Step 8: The found slave devices are listed in the tree



The screenshot displays the EC-Engineer software interface. The top menu bar includes File, View, Network, Settings, and Help. Below the menu bar are tabs for Configuration (selected), Diagnosis, Simulation, and Export ENI. The Project Explorer on the left shows a tree structure with 'Class-A MainDevice' expanded, revealing 'SubDevice_1001 [EC-Training Generator] (1001)'. The Device Editor on the right shows the 'General' tab with the following settings:

General	
Unit Name	Class-A MainDevice
Cycle Time [us]	1000
Source MAC address	64-C9-01-A4-67-C8

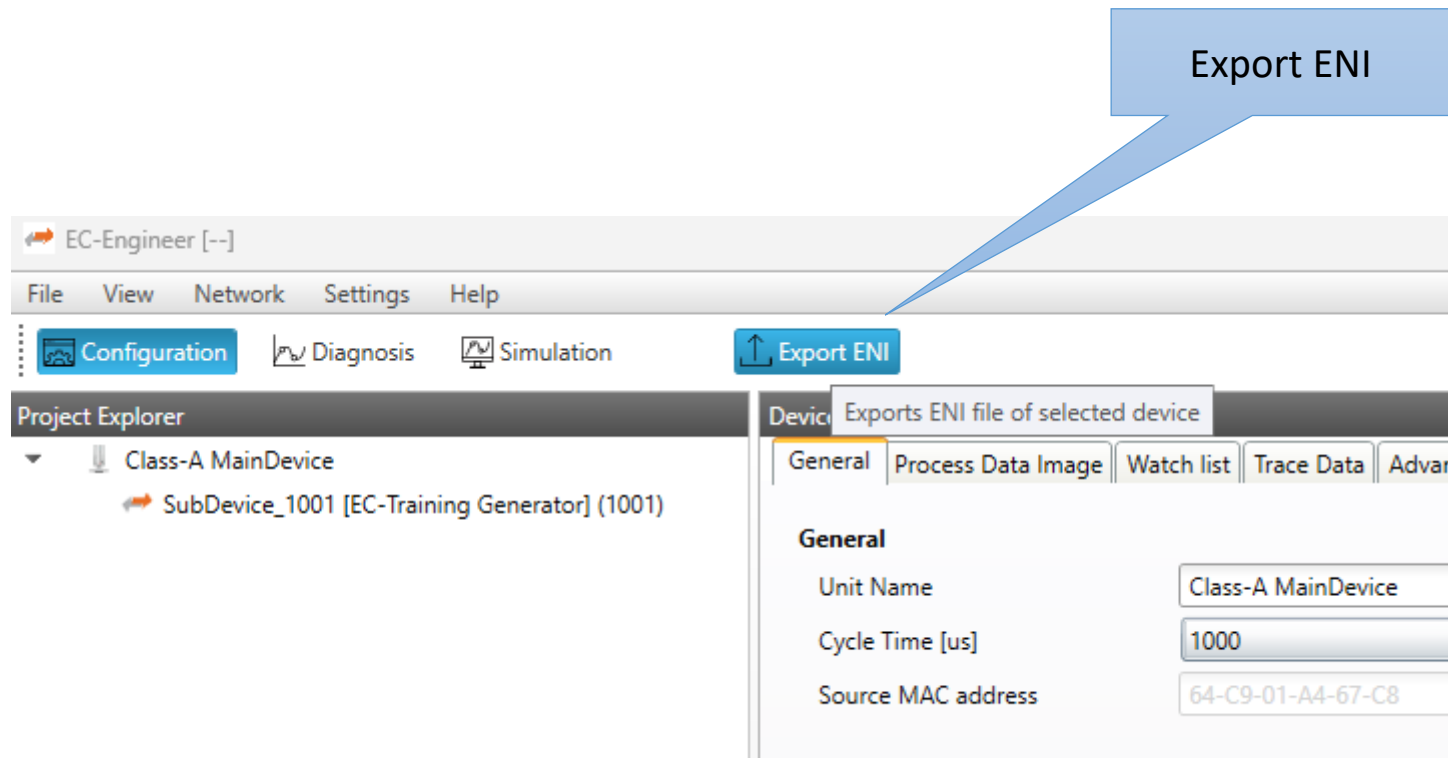
Local system	
Link Layer	Ndis
Network Adapter	Dock (Realtek USB GbE Family Controller)
Requested MainDevice State	Init
Diagnosis Mode: Cycle Time	4000
Diagnosis Mode: DCM	Off

Remote system

Generate bus configuration with EC-Engineer

Step 9: Export ENI file to, e.g., *D:\eni.xml*

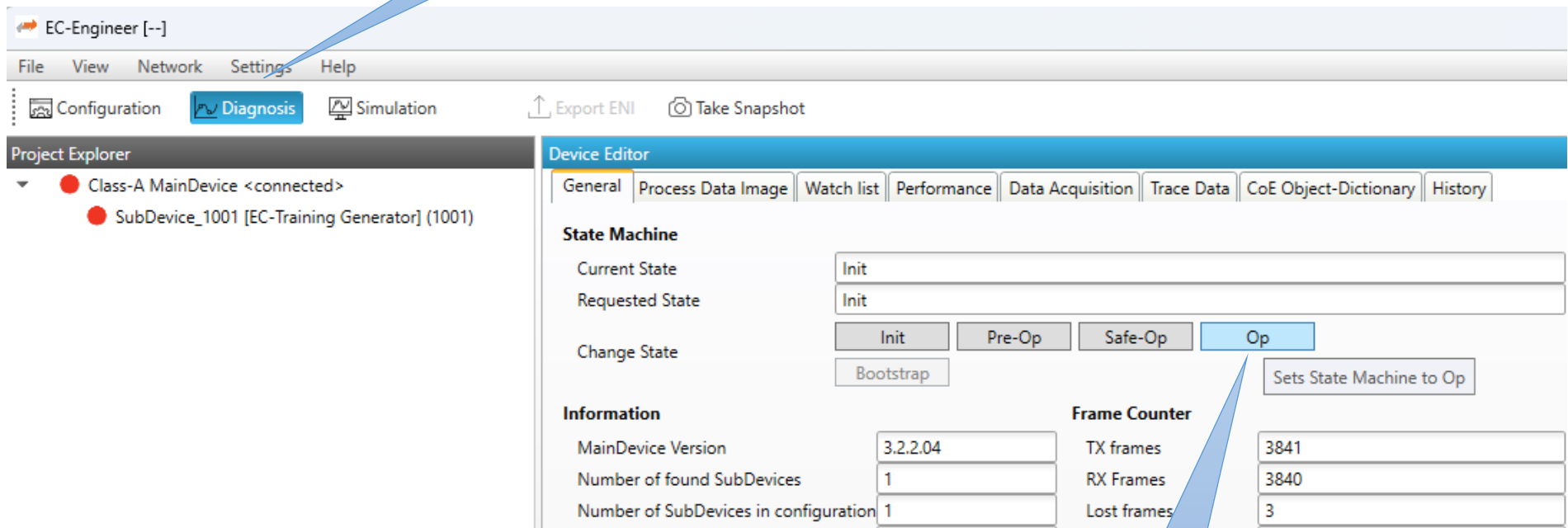
EC  **Engineer**



Generate bus configuration with EC-Engineer

Step 10: Switch to “Diagnosis Mode” and set state to OPERATIONAL

Switch to
“Diagnosis Mode”



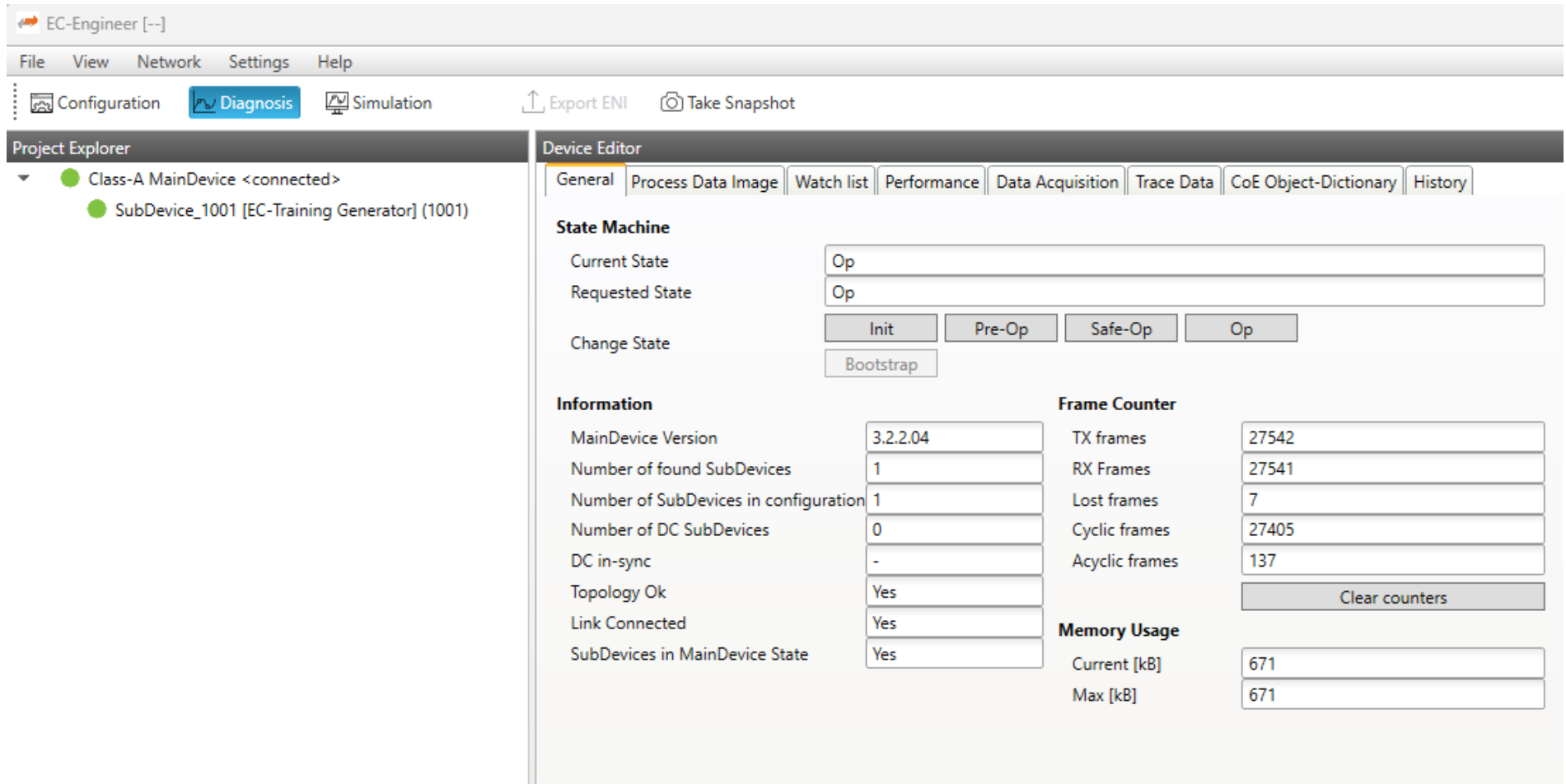
The screenshot shows the EC-Engineer software interface. The top menu bar includes File, View, Network, Settings, and Help. Below the menu bar, there are tabs for Configuration, Diagnosis (selected), and Simulation. The Project Explorer on the left shows a tree structure with 'Class-A MainDevice <connected>' and 'SubDevice_1001 [EC-Training Generator] (1001)'. The main area is the 'Device Editor' with several tabs: General, Process Data Image, Watch list, Performance, Data Acquisition, Trace Data, CoE Object-Dictionary, and History. The 'General' tab is active, showing the 'State Machine' section. The 'Current State' and 'Requested State' are both set to 'Init'. The 'Change State' section has buttons for 'Init', 'Pre-Op', 'Safe-Op', 'Op' (highlighted), and 'Bootstrap'. A tooltip for the 'Op' button says 'Sets State Machine to Op'. Below the State Machine section is the 'Information' section with fields for 'MainDevice Version' (3.2.2.04), 'Number of found SubDevices' (1), and 'Number of SubDevices in configuration' (1). To the right of the Information section is the 'Frame Counter' section with fields for 'TX frames' (3841), 'RX Frames' (3840), and 'Lost frames' (3).

Information		Frame Counter	
MainDevice Version	3.2.2.04	TX frames	3841
Number of found SubDevices	1	RX Frames	3840
Number of SubDevices in configuration	1	Lost frames	3

Set state to
OPERATIONAL

Generate bus configuration with EC-Engineer

Step 11: Network is OPERATIONAL



The screenshot displays the EC-Engineer software interface. The top menu bar includes 'File', 'View', 'Network', 'Settings', and 'Help'. Below the menu is a toolbar with icons for 'Configuration', 'Diagnosis' (highlighted), and 'Simulation', along with 'Export ENI' and 'Take Snapshot' buttons. The 'Project Explorer' on the left shows a tree structure with 'Class-A MainDevice <connected>' and 'SubDevice_1001 [EC-Training Generator] (1001)'. The main 'Device Editor' panel has tabs for 'General', 'Process Data Image', 'Watch list', 'Performance', 'Data Acquisition', 'Trace Data', 'CoE Object-Dictionary', and 'History'. The 'General' tab is active, showing the 'State Machine' section with 'Current State' and 'Requested State' both set to 'Op'. Below these are buttons for 'Init', 'Pre-Op', 'Safe-Op', 'Op', and 'Bootstrap'. The 'Information' section lists various device parameters, and the 'Frame Counter' section shows statistics for TX frames, RX frames, lost frames, cyclic frames, and acyclic frames. A 'Clear counters' button is also present. The 'Memory Usage' section shows 'Current [kB]' and 'Max [kB]' both at 671.

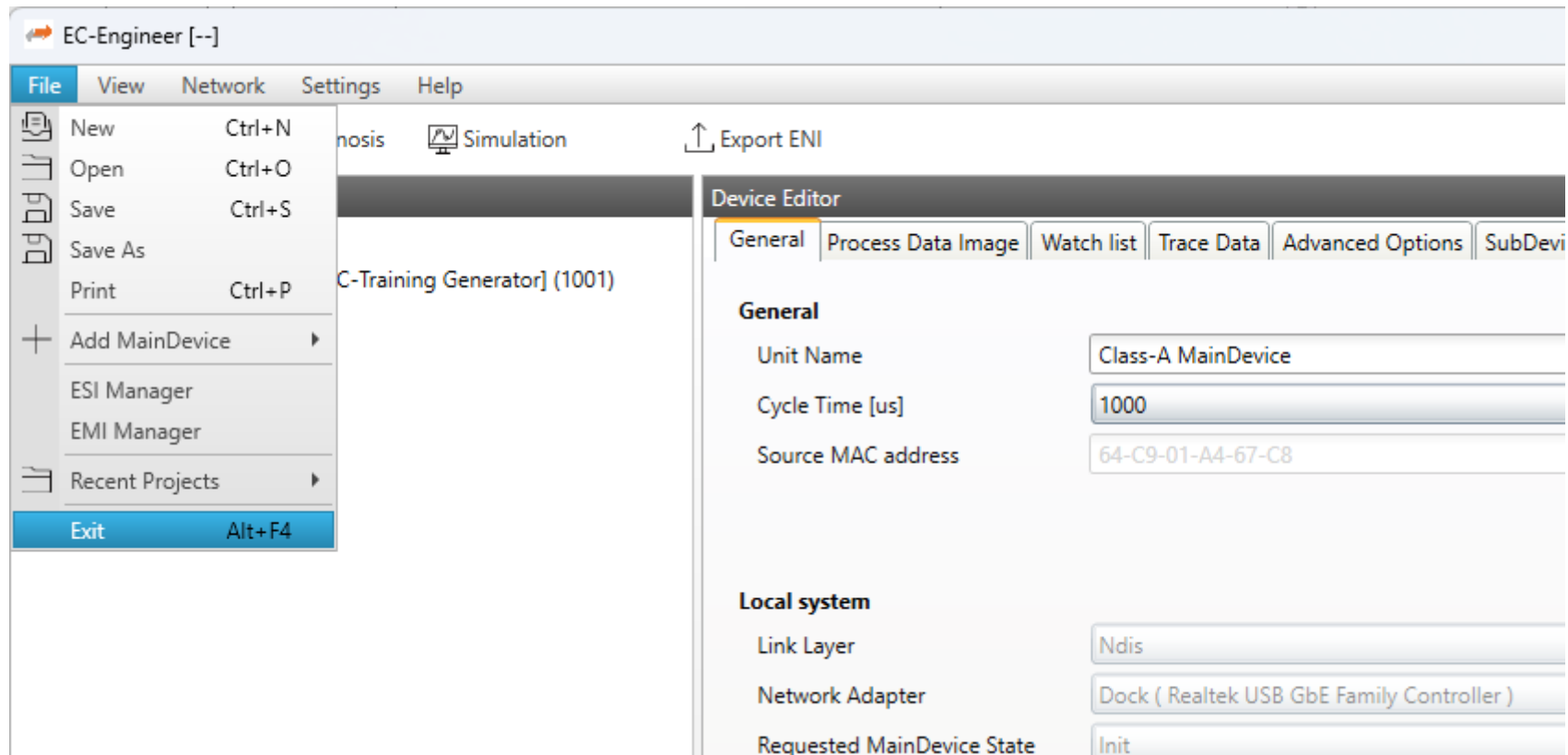
Information	
MainDevice Version	3.2.2.04
Number of found SubDevices	1
Number of SubDevices in configuration	1
Number of DC SubDevices	0
DC in-sync	-
Topology Ok	Yes
Link Connected	Yes
SubDevices in MainDevice State	Yes

Frame Counter	
TX frames	27542
RX Frames	27541
Lost frames	7
Cyclic frames	27405
Acyclic frames	137
<button>Clear counters</button>	

Memory Usage	
Current [kB]	671
Max [kB]	671

Generate bus configuration with EC-Engineer

Step 12: Switch back to “Configuration Mode” and Exit



Run EcMasterDemo with ENI file

- Transfer ENI to target device (if not Windows)
- Append ENI path, e.g. *-f D:\eni.xml* to Command Line Parameters
- Master commands all slaves to OP state:

```
0000000182: Bus scan successful - 1 slaves found
0000000185: *****
0000000185: Slave ID.....: 0x00000000
0000000185: Bus Index.....: 0
0000000185: Bus AutoInc Address.: 0x0000 ( 0)
0000000185: Bus Station Address.: 0x03e9 (1001)
0000000185: Bus Alias Address...: 0x0000 ( 0)
0000000185: Vendor ID.....: 0x00004154 = acontis technologies GmbH
0000000185: Product Code.....: 0x00000101 = Unknown
0000000185: Revision.....: 0x00000000 Serial Number: 0
0000000185: ESC Type.....: Infineon (0x98) Revision: 1 Build: 1
0000000185: Connection at Port A: yes (to 0x00010000)
0000000185: Connection at Port D: no (to 0xFFFFFFFF)
0000000185: Connection at Port B: no (to 0xFFFFFFFF)
0000000185: Connection at Port C: no (to 0xFFFFFFFF)
0000000185: Line Crossed.....: no
0000000185: Line Crossed Flags..: 0x0
0000000185: Cfg Station Address.: 0x03e9 (1001)
0000000185: PD IN   Byte.Bit offset: 0.0 Size: 72 bits (MSU 0)
0000000185: PD OUT  Byte.Bit offset: 0.0 Size: 72 bits (MSU 0)
0000000185: *****
0000000217: Master state changed from <UNKNOWN> to <INIT>
0000000361: Master state changed from <INIT> to <PREOP>
0000000471: Master state changed from <PREOP> to <SAFEOP>
0000000520: Master state changed from <SAFEOP> to <OP>
0000000525: EcMasterDemo will stop in 600s...
```

Online diagnosis of network with EC-Engineer

Step 1: Start EcMasterDemo

- Append `-sp` to Command Line Parameters
- Start EcMasterDemo

```
0000000019: emllNdis(\DEVICE\{866DD08-2B16-4778-84F7-D985180E2DA6}): Ecat Ndis Driver Version 3.1.3.4
0000000019: EtherCAT network adapter MAC: 00-1B-21-AB-D1-93
0000000072: Protected version, stop sending ethernet frames after 60 minutes if not licensed!
0000001966: Bus scan successful - 1 slaves found
0000001992: *****
0000001992: Slave ID.....: 0x00000000
0000001992: Bus Index.....: 0
0000001992: Bus AutoInc Address.: 0x0000 ( 0)
0000001992: Bus Station Address.: 0x03e9 (1001)
0000001992: Bus Alias Address...: 0x0000 ( 0)
0000001992: Vendor ID.....: 0x00004154 = acontis technologies GmbH
0000001992: Product Code.....: 0x00000101 = Unknown
0000001992: Revision.....: 0x00000000 Serial Number: 0
0000001992: ESC Type.....: Infineon (0x98) Revision: 1 Build: 1
0000001992: Connection at Port A: yes (to 0x00010000)
0000001992: Connection at Port D: no (to 0xFFFFFFFF)
0000001992: Connection at Port B: no (to 0xFFFFFFFF)
0000001992: Connection at Port C: no (to 0xFFFFFFFF)
0000001992: Line Crossed.....: no
0000001992: Line Crossed Flags..: 0x0
0000001992: Cfg Station Address.: 0x03e9 (1001)
0000001992: PD IN   Byte.Bit offset: 0.0   Size: 72 bits (MSU 0)
0000001992: PD OUT  Byte.Bit offset: 0.0   Size: 72 bits (MSU 0)
0000001992: *****
0000002250: Master state changed from <UNKNOWN> to <INIT>
0000003433: Master state changed from <INIT> to <PREOP>
0000004331: Master state changed from <PREOP> to <SAFEOP>
0000004739: Master state changed from <SAFEOP> to <OP>
0000004761: EcMasterDemo will stop in 600s...
```

Online diagnosis of network with EC-Engineer

Step 2: Start EC-Engineer and select “Diagnosis / Remote MainDevice”

Device Editor

Start Page

What would you like to do?

Add Device

- ↔ EtherCAT MainDevice (Class A)
- ↔ EtherCAT MainDevice (Class B)
- ↔ EtherCAT MainDevice (LxWin)
- ↔ EtherCAT MainDevice (Motion)
- ↔ EtherCAT Simulator

Open Recent

- ↔ D:\proj\w\project.ecc

Configuration for MainDevice or Network Simulation

Offline

Create network configuration with ESI files



Online Local

Scan network connected to local system



Online Remote

Scan network connected to remote system



Diagnosis

Offline

Diagnose recorded network snapshots



Local

Diagnose network connected to local system



Remote MainDevice

Diagnose network connected to remote system



Connect EC-Engineer to EcMasterDemo

Step 3: Set IP address and press “Select”

General

Unit Name:

Cycle Time [us]:

Source MAC address:

Local system

Link Layer:

Network Adapter:

Requested MainDevice State:

Diagnosis Mode: Cycle Time:

Diagnosis Mode: DCM:

Remote system

Protocol:

IP Address:

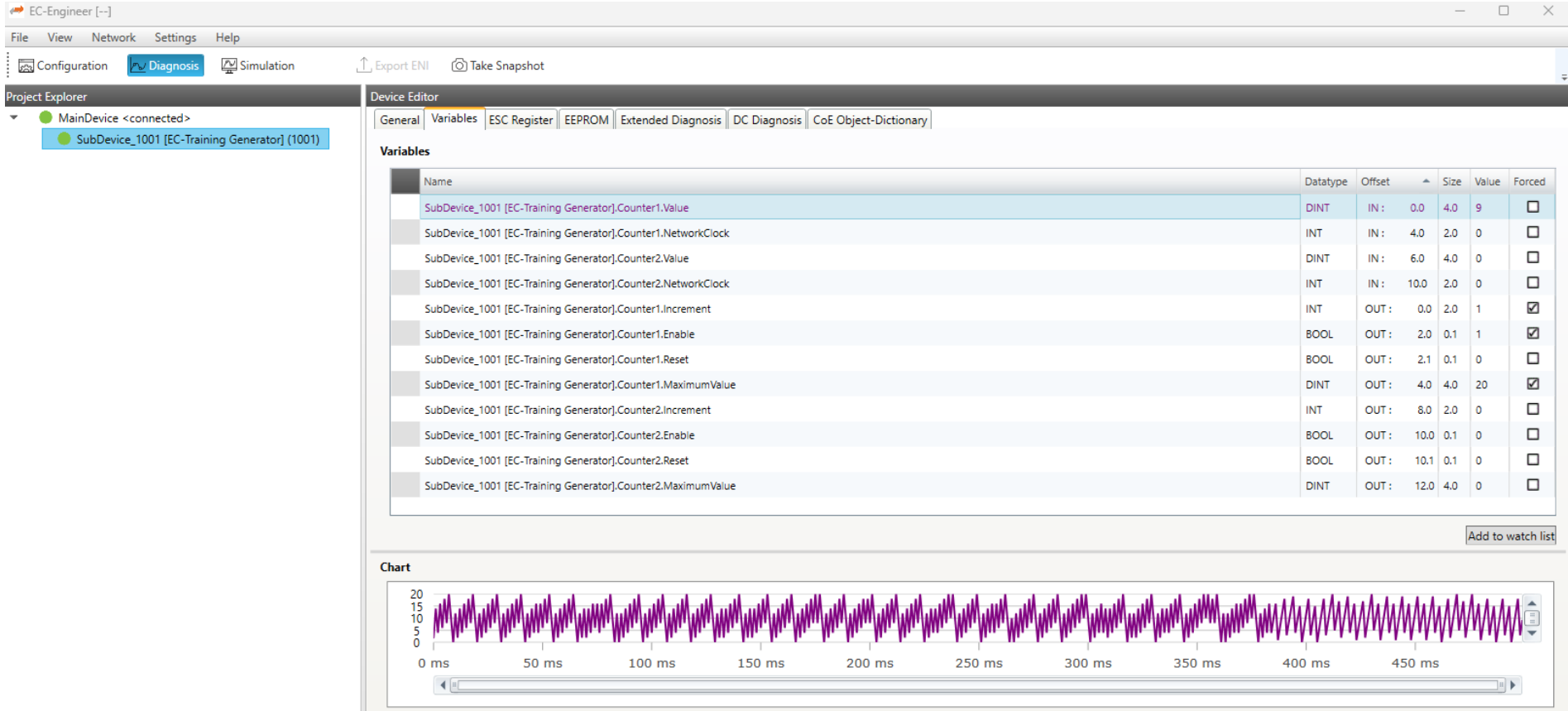
Port:

Instance:

Press “Select”

Online diagnosis of network with EC-Engineer

Step 4: Select device, check inputs, force outputs, etc.



The screenshot displays the EC-Engineer software interface. The top menu bar includes File, View, Network, Settings, and Help. Below the menu is a toolbar with icons for Configuration, Diagnosis (active), Simulation, Export ENI, and Take Snapshot. The Project Explorer on the left shows a tree structure with 'MainDevice <connected>' and 'SubDevice_1001 [EC-Training Generator] (1001)'. The Device Editor on the right has tabs for General, Variables (active), ESC Register, EEPROM, Extended Diagnosis, DC Diagnosis, and CoE Object-Dictionary. The Variables tab shows a table of variables for 'SubDevice_1001 [EC-Training Generator]'. Below the table is a 'Chart' section displaying a waveform plot over time.

Name	Datatype	Offset	Size	Value	Forced
SubDevice_1001 [EC-Training Generator].Counter1.Value	DINT	IN : 0.0	4.0	9	<input type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter1.NetworkClock	INT	IN : 4.0	2.0	0	<input type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter2.Value	DINT	IN : 6.0	4.0	0	<input type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter2.NetworkClock	INT	IN : 10.0	2.0	0	<input type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter1.Increment	INT	OUT : 0.0	2.0	1	<input checked="" type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter1.Enable	BOOL	OUT : 2.0	0.1	1	<input checked="" type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter1.Reset	BOOL	OUT : 2.1	0.1	0	<input type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter1.MaximumValue	DINT	OUT : 4.0	4.0	20	<input checked="" type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter2.Increment	INT	OUT : 8.0	2.0	0	<input type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter2.Enable	BOOL	OUT : 10.0	0.1	0	<input type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter2.Reset	BOOL	OUT : 10.1	0.1	0	<input type="checkbox"/>
SubDevice_1001 [EC-Training Generator].Counter2.MaximumValue	DINT	OUT : 12.0	4.0	0	<input type="checkbox"/>

Chart

20
15
10
5
0

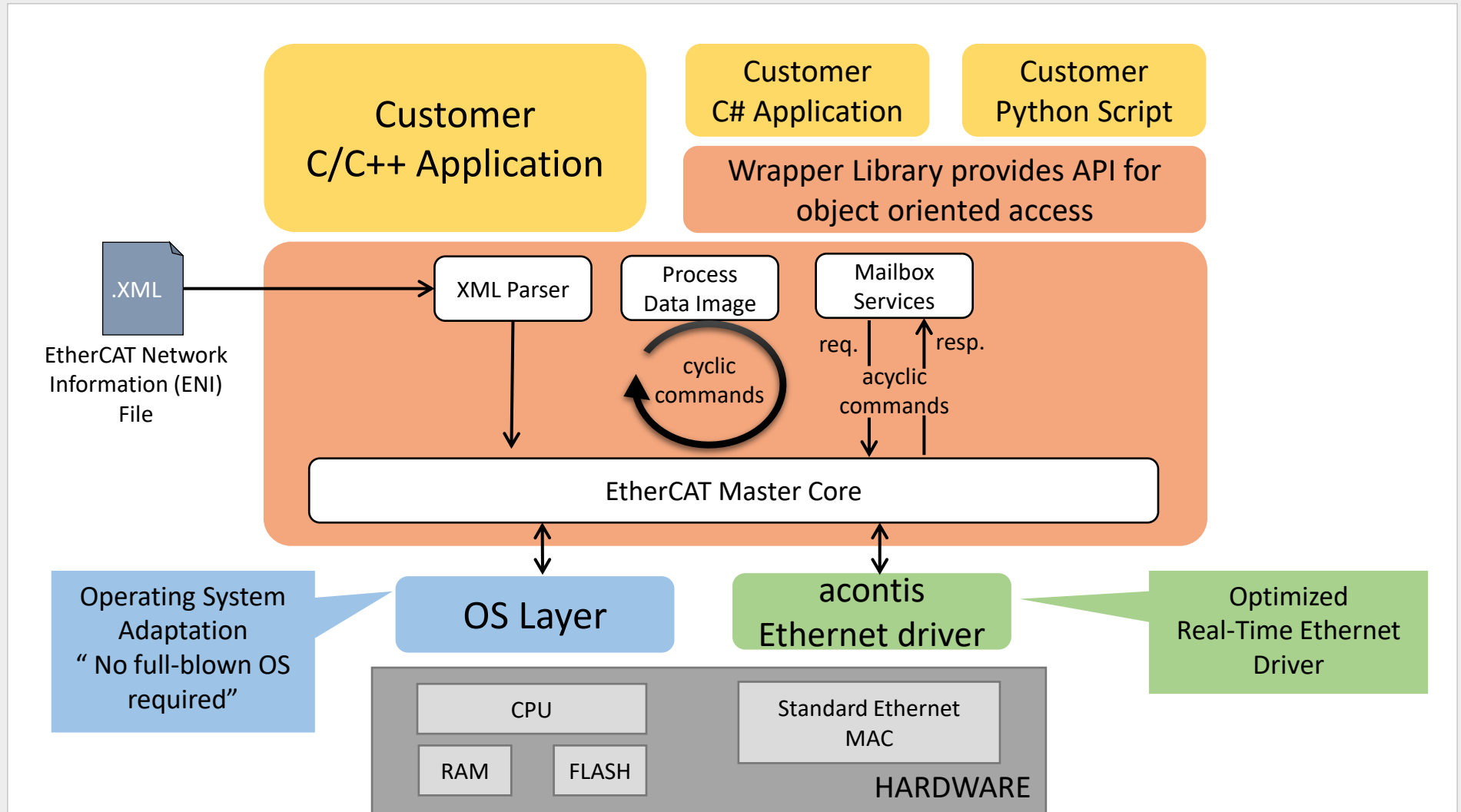
0 ms 50 ms 100 ms 150 ms 200 ms 250 ms 300 ms 350 ms 400 ms 450 ms

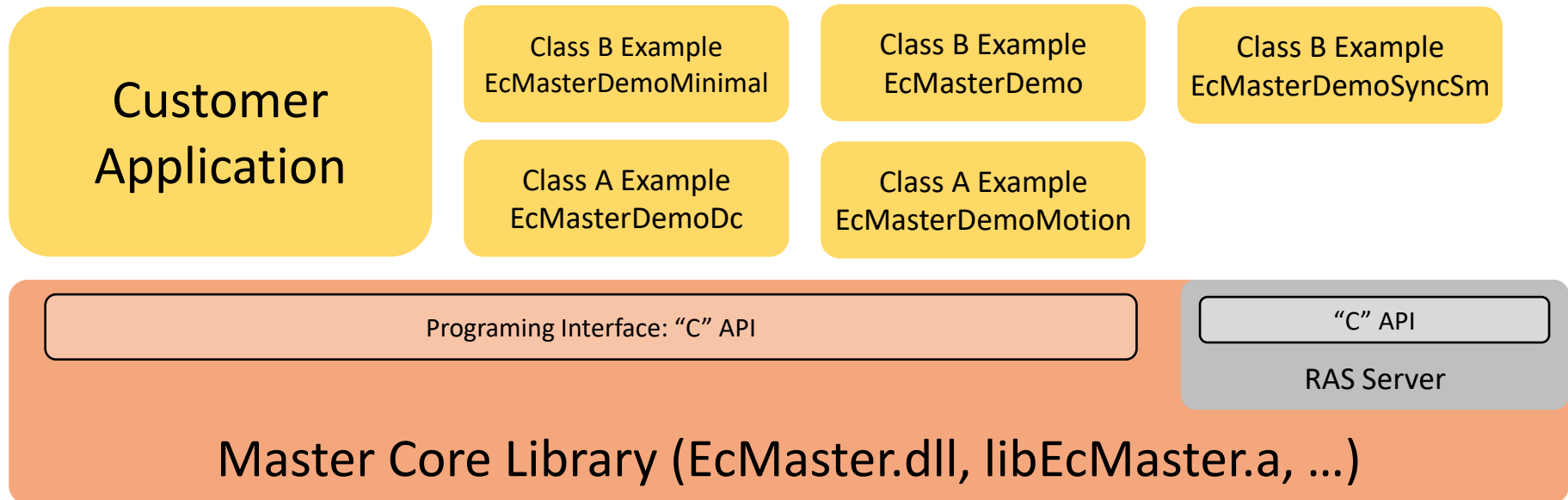
- Monitor inputs
- Force outputs
- Slave states
- Change master and/or slave state
- CoE Object Dictionary
- Extended Diagnosis

EC  ***Master***

Application Programming Interfaces

EC-Master Architecture





The EtherCAT Master Core Library and the RAS Server Module are implemented in C++. The API interfaces are C language interfaces, thus the master can be used in ANSI-C as well as in C++ environments.

- **EcMasterDemo** is the basic example application for EC-Master. The example shows how to initialize the master and how to put the network into operational state. Based on the provided ENI file this example can handle all kind of EtherCAT Slaves.
- **EcMasterDemoMinimal** is a more simple demo to ease into the EC-Master code. The simple demo focuses just on the necessary APIs and functions to startup the EC-Master along with the attached network and bring everything into operational state.
- **EcMasterDemoSyncSm** demonstrates a different network timing which requires the interrupt from the Ethernet controller used by the master. This cyclic frame is transmitted at the begin of the cycle and the process data are updated immediately after the frame returns.
- **EcMasterDemoDc** is a good starting point for application requiring the accurate synchronization of slaves based on the Distributed Clocks (DC) technology. To synchronize the master controller with the slaves several modes can be selected.
- **EcMasterDemoMotion** comes with a simple motion control library to control drives implemented according to the profile CiA402 and the ETG Implementation Directive ETG.6010. The example supports the operation modes Cyclic Synchronous Position (CSP) and Cyclic Synchronous Velocity (CSV).



EC-Master Demo Application Overview

- Source code identical for all supported platforms
- Demonstrates basic EtherCAT communication
- Master stack initialization
- Set all slaves into OPERATIONAL state
- The output messages of the demo application will be printed on the console as well as in some files. The following log files will be created:
 - ecmaster0.log: all messages
 - error0.log: application error messages (logged via LogError function)

Examples\Common\Windows

EcDemoMain
Application Entry Point

EcDemoPlatform
Thread Priorities

EcDemoTimingTask
Platform independent default timing task

Examples\Common

EcDemoParams
Parse Command Line

EcSelectLinkLayer
Parse and Create Link Layer Params

EcDemoTimingTaskPlatform
Platform dependent optimized timing task

Examples\EcMasterDemo

EcDemoApp
Example EtherCAT Application

Examples\Common

EcNotification
Notification Handling

EcLogging
Logging to Console and File

EcSlaveInfo
Common Helper Functions

EcSdoServices
Read and Print Object Dictionary

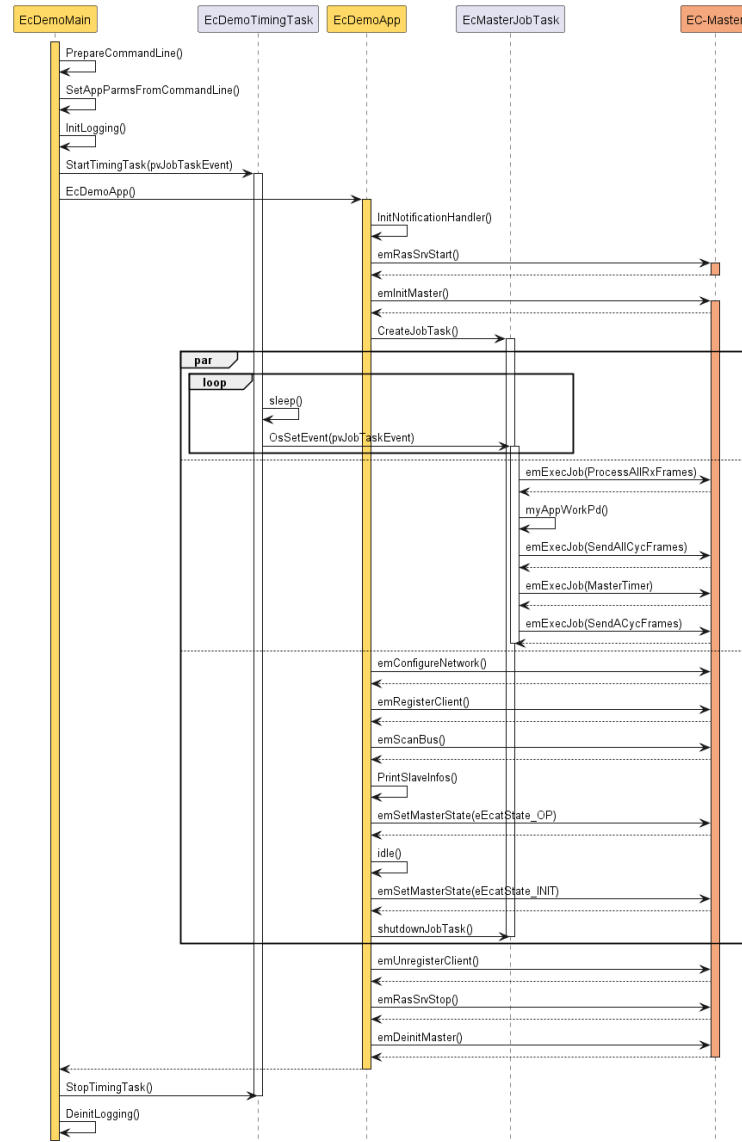
Sources\Common

EcTimer
Timeout Helper

The EC-Master Demo application consists of the following files

EcDemoMain.cpp	Entrypoint for the different operating systems
EcDemoPlatform.h	Operating system specific settings (task priorities, timer settings)
EcDemoTimingTask.h/.cpp	Operating system independent default timing task implementation (base class)
EcDemoTimingTaskPlatform.h/.cpp	Operating system dependent performance increasing overrides of EcDemoTimingTask
EcDemoApp.cpp	Initialize, start and terminate the EtherCAT master
EcDemoApp.h	Application specific settings for EcDemoApp
EcDemoParms.cpp	Parsing of command line parameters
EcDemoParms.h	Basic configuration structs and parameters (EtherCAT master parameter)
EcSelectLinkLayer.cpp	Common functions which abstract the command line parsing into Link Layer parameters
EcNotification.cpp	Slave monitoring and error detection (function emNotify())
EcSdoServices.cpp	Read CoE object dictionary example
EcSlaveInfo.cpp	Slave information services (bus scan, slave properties, getting information of slaves connected to the EtherCAT bus)
EcLogging.cpp	Message logging functions
EcTimer.cpp	Start and monitor timeouts

Demo Life Cycle



Important demo functions

- **EcDemoMain()**
Operating system specific main function to start the demo.
Implement command line parsing for individual parameter settings.
- **EcTimingTask()/CEcDemoTimingTaskPlatform** class
Timing thread.
This thread sets the timing event that triggers the **EcMasterJobTask()** for the next cycle.
Do not use sleep() herein, since with DCM, the event **must trigger with extreme low drifting cycle (average 999.4 - 1000.6). Using sleep() leads to too high drift!**
- **EcDemoApp()**
Demo application. The function takes care of starting and stopping the master and all related tasks. In between, the function runs idle, while all relevant work is done by the **EcMasterJobTask()**
- **EcMasterJobTask()**
Thread that does the necessary periodic work.
Very important here is **myAppWorkPd()** between *eUsrJob_ProcessAllRxFrames* and *eUsrJob_SendAllCycFrames*.
Application-specific manipulations of the process image, which must be synchronous with the bus cycle, can be carried out here

- **emInitMaster()**
Prepare the master for operation and set operational parameters, e.g. used Link Layer, buffer sizes, maximum number of slaves,
- **emConfigureMaster()**
Loads the configuration from the ENI (XML file)
- **emRegisterClient()**
Register the application as a client at the EC-Master to receive event notifications
- **emSetMasterState()**
Startup the EtherCAT master and switch the bus to the different states from INIT to OPERATIONAL.
- **emDeinitMaster()**
Free allocated resources and unbind from devices.

EC  ***Master***

Next Steps

- Read EC-Master User Manual Chapter 2 [“Getting Started”](#)
- Read EC-Master User Manual Chapter 3 [“Software Integration”](#)
- More Information
 - acontis Developer Center:
<https://developer.acontis.com/>
 - EC-Master User Manuals, Tutorials, Training Slides, etc.:
<https://developer.acontis.com/ec-master>
 - [acontis technologies YouTube channel](#)