

SOES EtherCAT Slave Example

Getting started with EtherCAT and Infineon XMC4



Workflow

1. Overview and Requirements
2. Setup hardware, DAVE and EtherCAT SDK
3. Define the EtherCAT application with EtherCAT SDK
4. Implement the application in DAVE
5. Test the application with EtherCAT SDK

Workflow

1. Overview and Requirements
2. Setup hardware, DAVE and EtherCAT SDK
3. Define the EtherCAT application with EtherCAT SDK
4. Implement the application in DAVE
5. Test the application with EtherCAT SDK

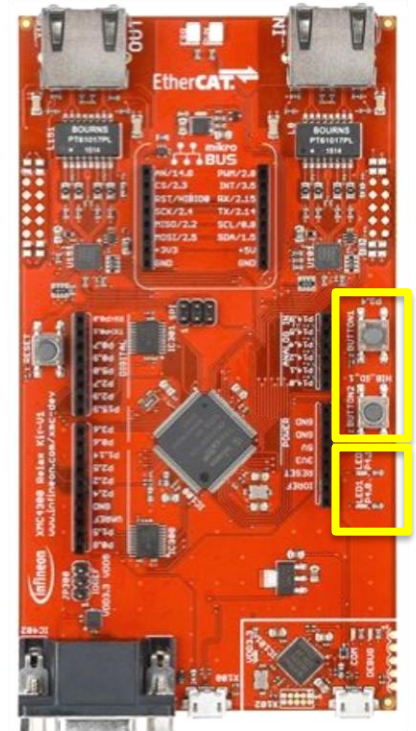
Overview

This example demonstrate how to implement an XMC4300/XMC4800 Relax Kit based EtherCAT slave using rt-labs EtherCAT SDK integrated in DAVE and open source EtherCAT slave stack SOES (Simple Open EtherCAT slave).

While following this getting started guide you will get an introduction to EtherCAT and its components without having any previous experience with EtherCAT.

The goal is to create a simple free-running Input / Output EtherCAT slave that is conformant to the EtherCAT specification and will pass the EtherCAT Conformance Test.

The application will make use of the XMC4300/XMC4800 relax kits on board BUTTONS and LEDS, the documentation present pictures, numbers and names for XMC4300 Relax Kit but it is easy to adopt to fit XMC4800 Relax Kits, simply point out correct GPIOs for the XMC4800.



Requirements

- XMC4 Relax EtherCAT Kit



- RJ45 Ethernet Cable



- Windows Laptop

- DAVE v4
- rt-labs EtherCAT SDK
- Wireshark w/ WinPCAP



- Micro USB Cable (Debugger connector)



Requirements - free downloads



- Install DAVE , get @ [Download DAVE](#)
- Install EtherCAT SDK , get@ [rt-labs](#)



- EtherCAT slave stack SOES, get @ [OpenEtherCATsociety SOES](#)



- Wireshark, get @ [Wireshark.org](#)

Workflow

1. Overview and Requirements
2. Setup hardware, DAVE and EtherCAT SDK
3. Define the EtherCAT application with EtherCAT SDK
4. Implement the application in DAVE
5. Test the application with EtherCAT SDK

Setup - hardware

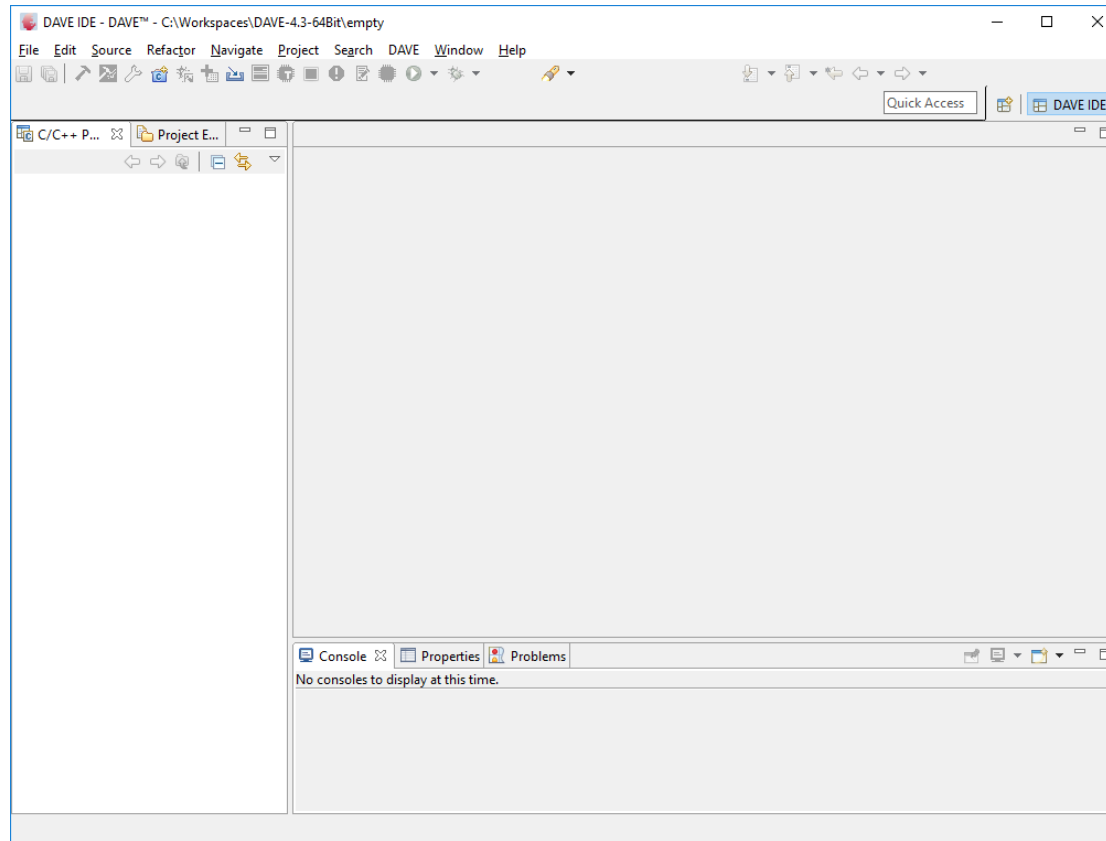
- Micro USB Cable from laptop to debug connector



- Ethernet cable from laptop port to Relax Kit IN-port

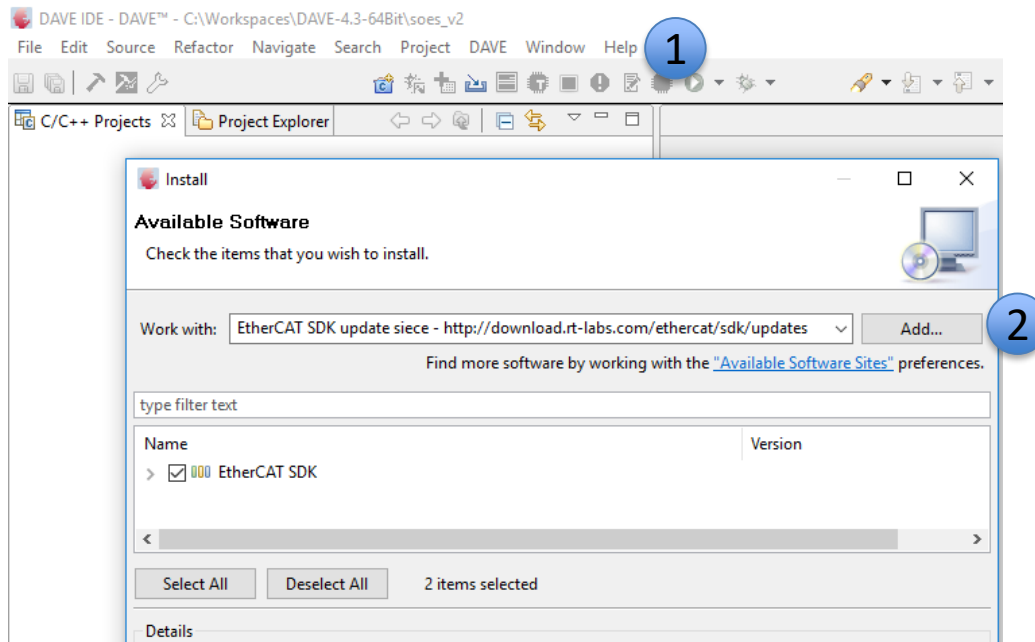


Setup - Start DAVE



Setup- Install EtherCAT SDK

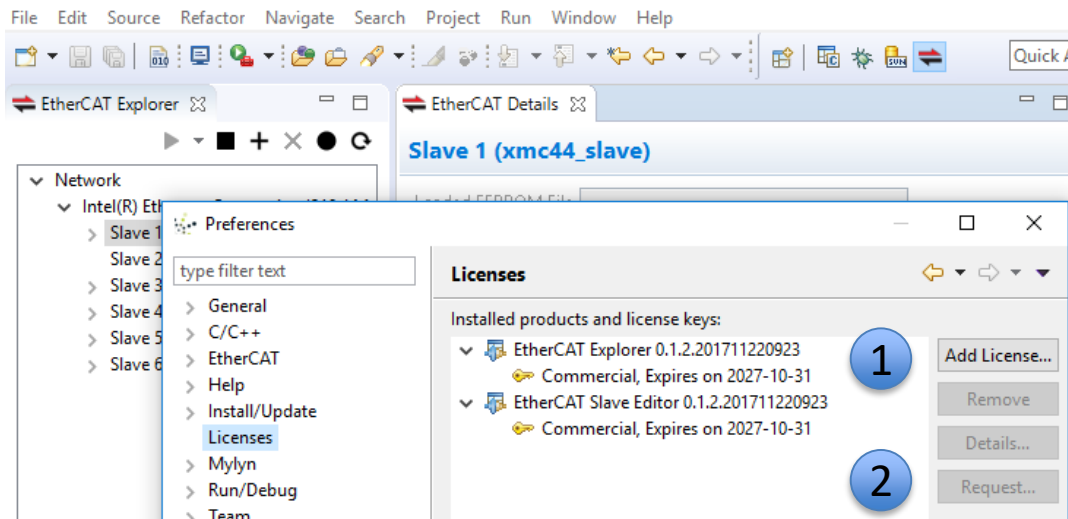
- 1 Go to menu: Help-> Install New Software
- 2 Add Add... download site: <http://download.rt-labs.com/ethercat/sdk/updates>



Setup- Add EtherCAT SDK license

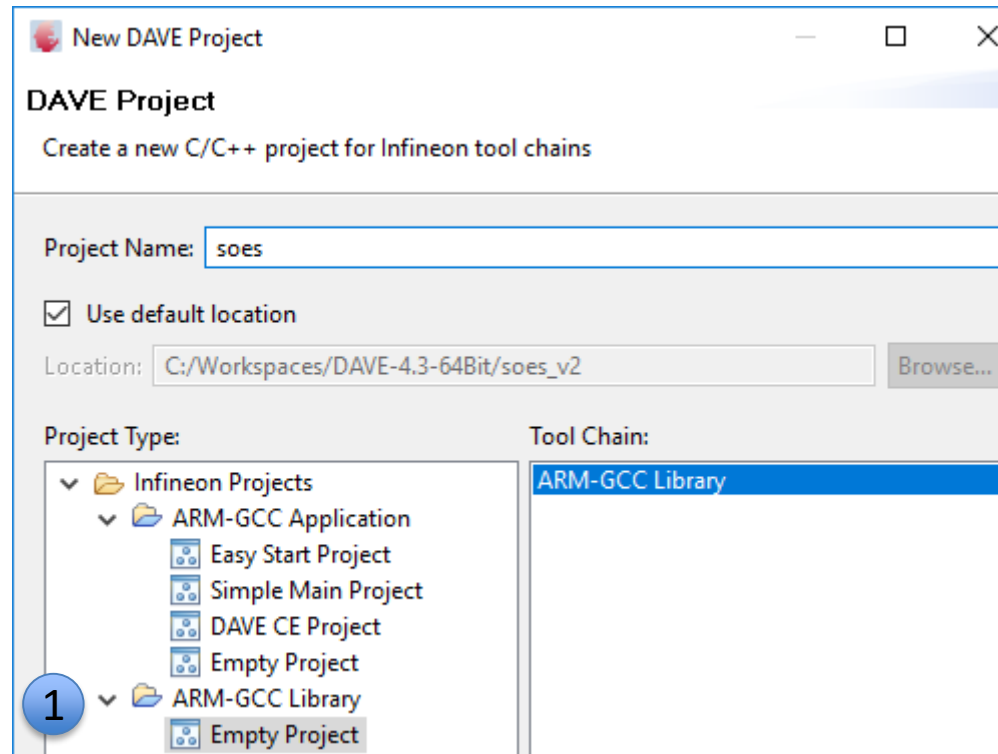
A license is required to be able to use the EtherCAT SDK, to install a license Go to: Windows->Preferences->Licenses. There are two options,

- 1 Add License..., browse an existing license file
- 2 Request an evaluation license by providing name and e-mail, shortly an e-mail with a license will be sent to given e-mail. Use option 1 to add the license.




Setup - Create the EtherCAT slave stack library project

- 1 Go to menu: File->New->DAVE Project-><Project Type of your choice>



Setup - Choose XMC4300 or XMC4800

 New DAVE Project

Microcontroller Selection Page

Select the microcontroller for which the project has to be created

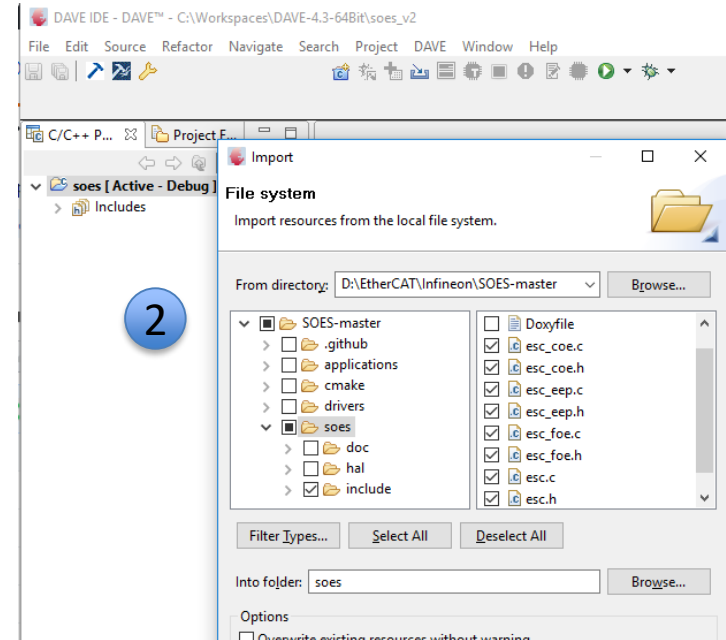
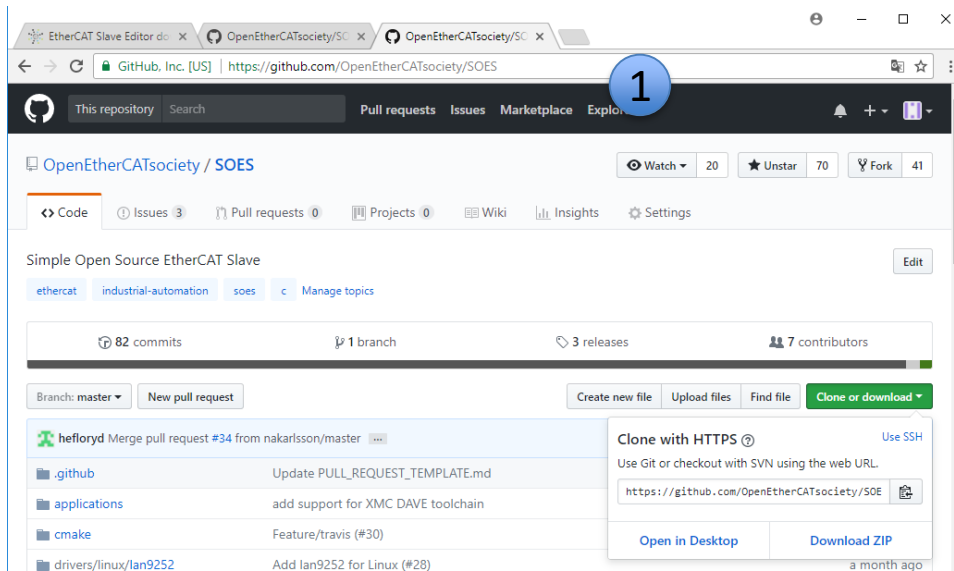
- ✓ ☒ Microcontrollers
 - ✓ ☒ XMC4000
 - > ☐ XMC4800 Series
 - > ☐ XMC4700 Series
 - > ☐ XMC4500 Series
 - > ☐ XMC4400 Series
 - ✓ ☒ XMC4300 Series
 - ☒ XMC4300-F100x256
 - > ☐ XMC4200 Series
 - > ☐ XMC4100 Series
 - ✓ ☐ XMC1000
 - > ☐ XMC1100 Series

Microcontroller Features

Package= LQFP100

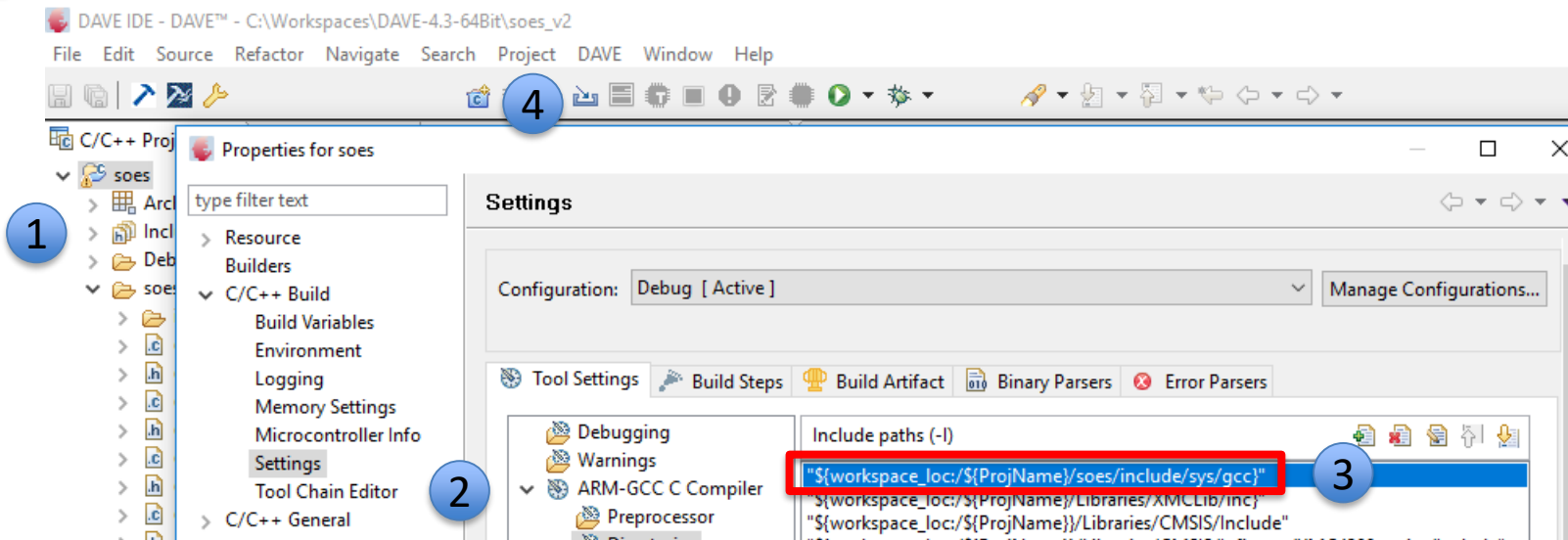
Setup – import files to slave stack project

- 1 Download or Clone the EtherCAT Slave Stack SOES@ GitHub
OpenEtherCATsociety/SOES
- 2 Import generic parts of the slave stack \soes and soes\include



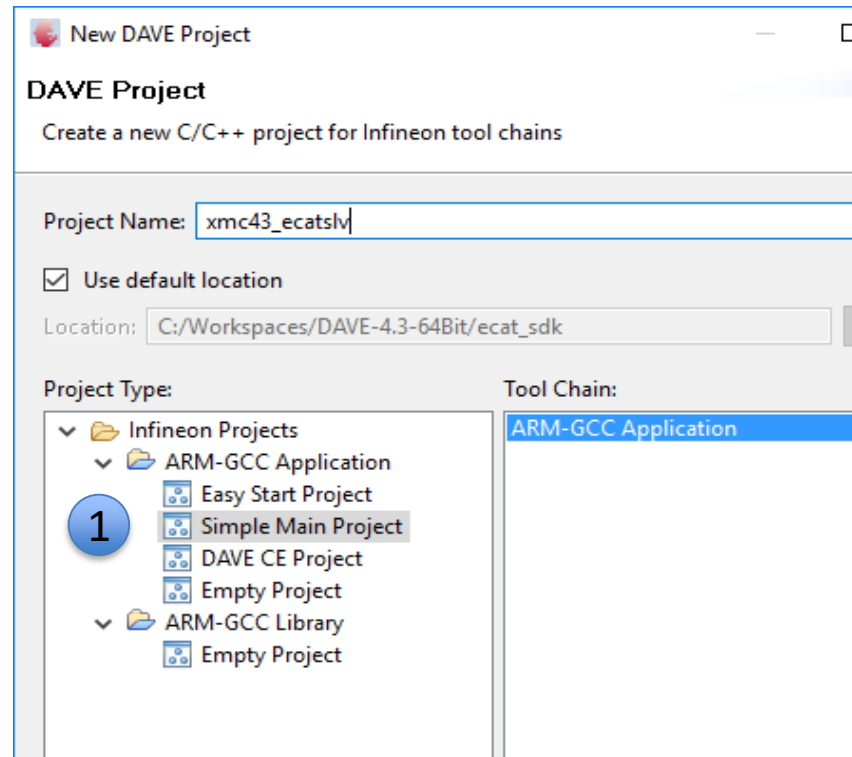
Setup -Add slave stack includes and build

- 1 Right click the Library <Project> and select Properties
- 2 Go to: Properties -> C/C++ Build -> Settings -> ARM-GCC C Compiler
- 3 Add the marked includes
- 4 Goto Project and Build All




Setup - Create the EtherCAT slave application project

1 Go to menu: File->New->DAVE Project-><Project Type of your choice>



OBS! Using DAVE CE may cause pin conflicts not detected by DAVE since SOES also do some hardware setup

Setup - Choose XMC4300 or XMC4800

 New DAVE Project

Microcontroller Selection Page

Select the microcontroller for which the project has to be created

- ▼ ☒ Microcontrollers
 - ▼ ☒ XMC4000
 - > ☐ XMC4800 Series
 - > ☐ XMC4700 Series
 - > ☐ XMC4500 Series
 - > ☐ XMC4400 Series
 - ▼ ☒ XMC4300 Series
 - ☒ XMC4300-F100x256
 - > ☐ XMC4200 Series
 - > ☐ XMC4100 Series
 - ▼ ☐ XMC1000
 - > ☐ XMC1100 Series

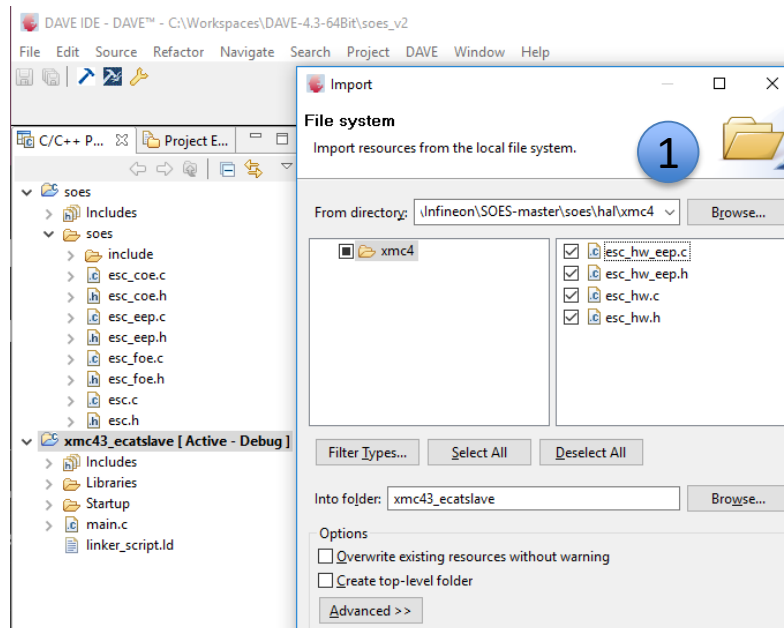
Microcontroller Features

Package= LQFP100

Setup – import files to application project

1

Import SOES DAVE XMC hardware abstraction layer parts of the slave stack
\\soes\\hal\\xmc4



Setup -Add slave stack includes

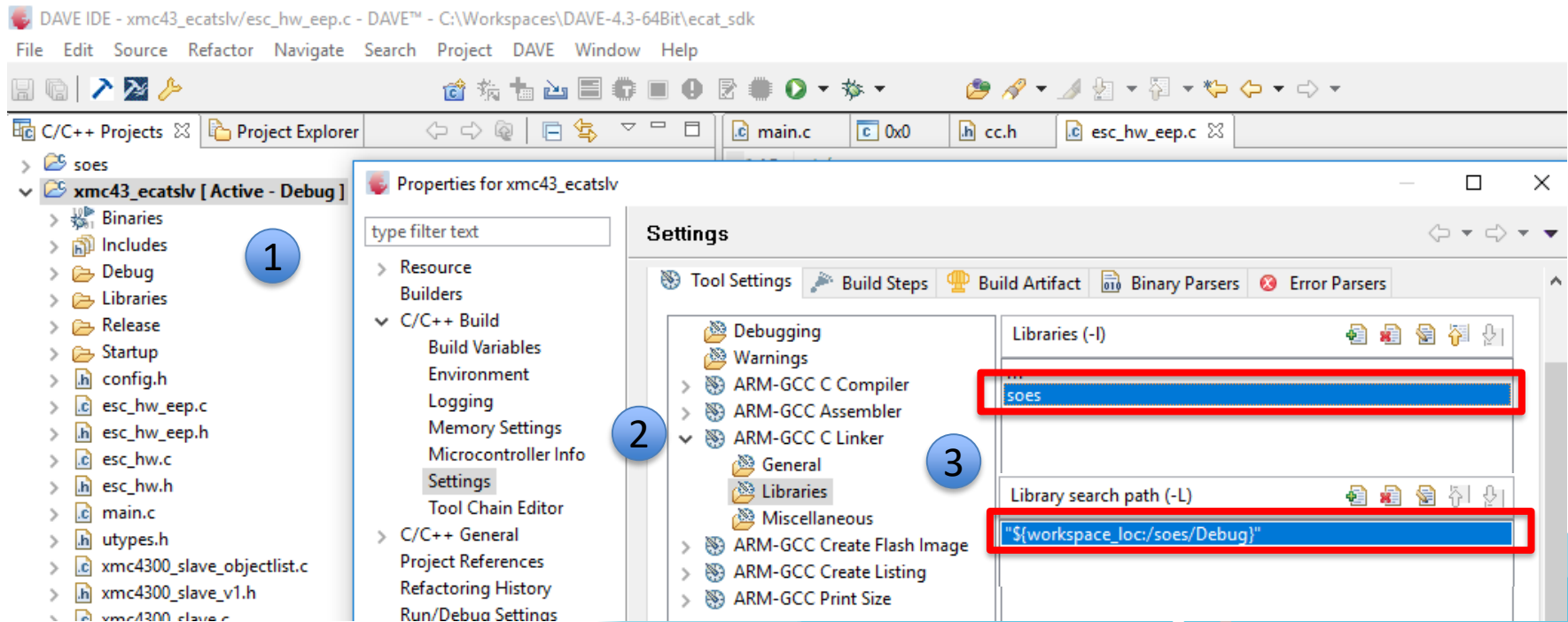
- 1 Right click the Application <Project> and select Properties
- 2 Go to: Properties -> C/C++ Build -> Settings -> ARM-GCC C Compiler
- 3 Add the marked includes

The screenshot illustrates the process of adding slave stack includes in the DAVE IDE. It is divided into three numbered sections:

- Section 1:** The Project Explorer on the left shows the project structure. The project `xmc43_ecatslv` is selected under the `soes` folder. A blue circle with the number '1' is placed over the project name.
- Section 2:** The 'Properties for xmc43_ecatslv' dialog is open. The left sidebar shows the navigation tree with 'C/C++ Build' > 'Settings' selected. A blue circle with the number '2' is placed over the 'Settings' item.
- Section 3:** The 'Settings' dialog for the 'ARM-GCC C Compiler' is shown. The 'Tool Settings' tab is active, and the 'Directories' sub-tab is selected. In the 'Include paths (-I)' list, the path `"${workspace_loc:/soes/soes/include/sys/gcc}"` is highlighted with a red rectangle. A blue circle with the number '3' is placed over the 'Error Parsers' tab label.

Setup -Add slave stack library

- 1 Right click the Application <Project> and select Properties
- 2 Go to: Properties -> C/C++ Build -> Settings -> ARM-GCC C Linker
- 3 Add the marked library and search path



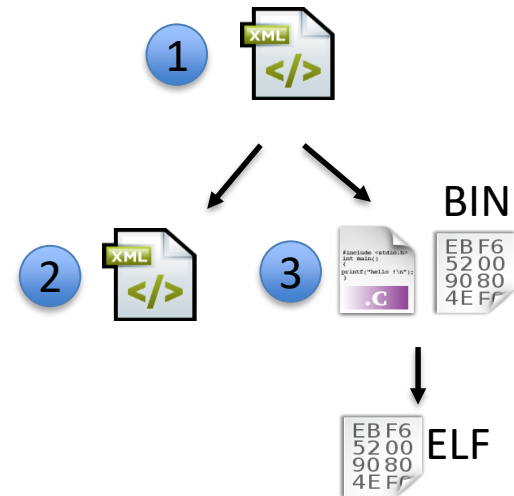
Workflow

1. Overview and Requirements
2. Setup hardware, DAVE and EtherCAT SDK
3. Define the EtherCAT application with EtherCAT SDK
4. Implement the application in DAVE
5. Test the application with EtherCAT SDK

EtherCAT SDK - Slave editor

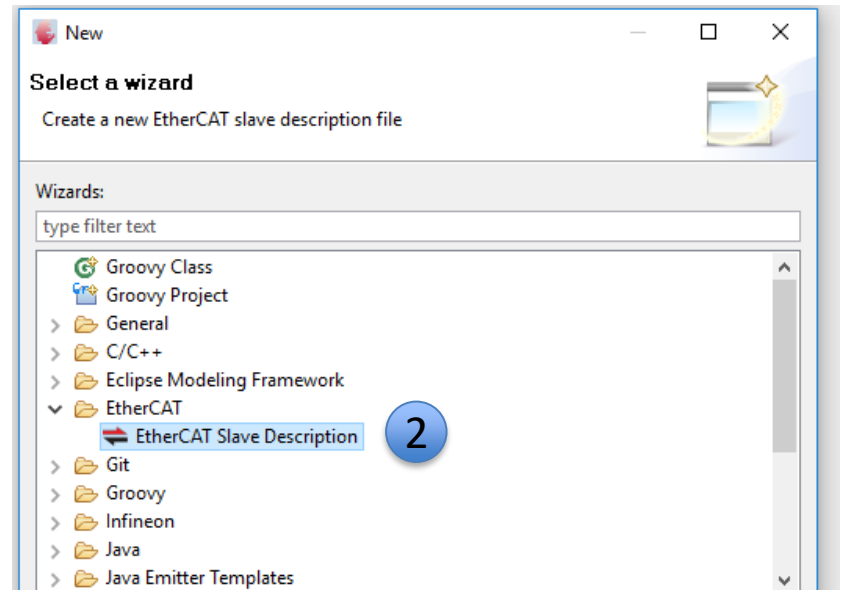
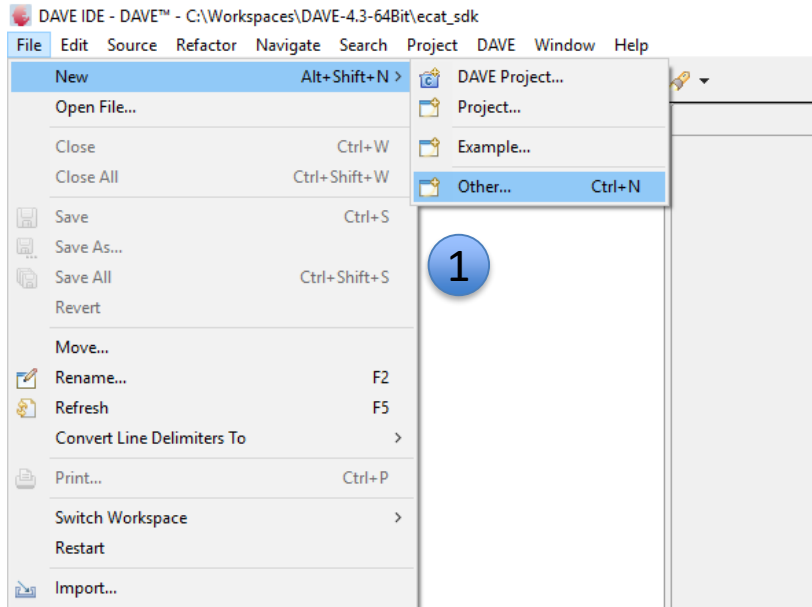
In this example we develop an EtherCAT slave with fixed process data and an Object Dictionary. To run the CTT (**C**onformance **T**est **T**ool) we need to provide three objects, EEPROM, OD (**O**bject **D**ictionary) and ESI-file(**E**therCAT **S**lave **I**nformation). The EEPROM and OD are loaded on the target for online configuration, the ESI is used for offline configuration. The information in these three objects overlap and must be coherent to pass the conformance test. The EtherCAT SDK help you generate these objects and keep the information aligned throughout the EtherCAT slave lifetime.

- 1 The EtherCAT slave editor create and store the information in an esx-file, only used by the Slave Editor
- 2 The tool generates a conformant ESI-file needed to run CTT
- 3 The tool generates a conformant OD and EEPROM that will be included in the application project. The OD is a C array/struct compiled with the C application, since we emulate the EEPROM we will convert the EEPROM binary to a linkable object and link it to the resulting application ELF



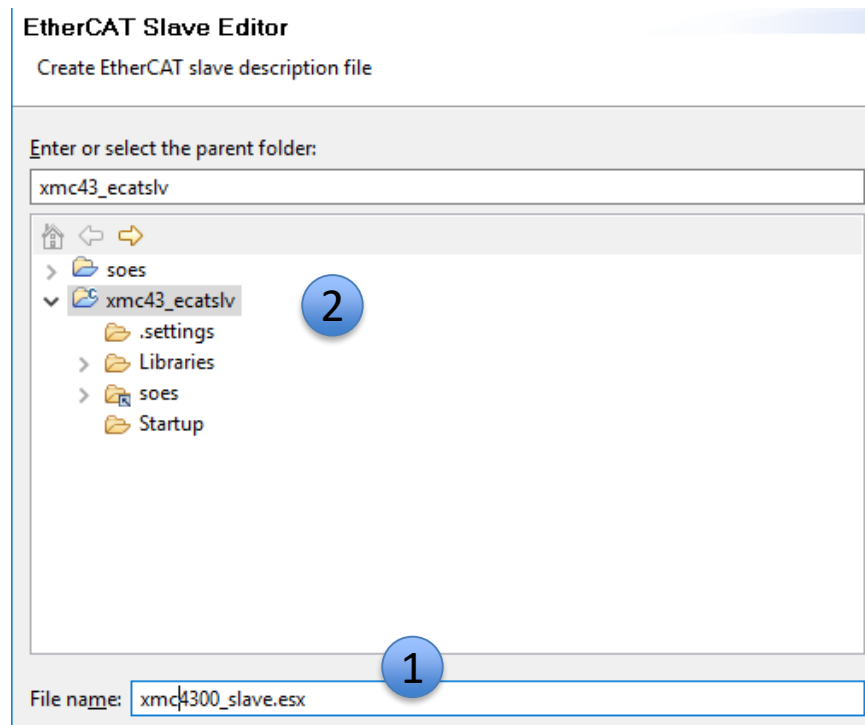
EtherCAT SDK - Slave editor

- 1 Go to menu: File->Other
- 2 Select Wizard: EtherCAT->EtherCAT Slave Description



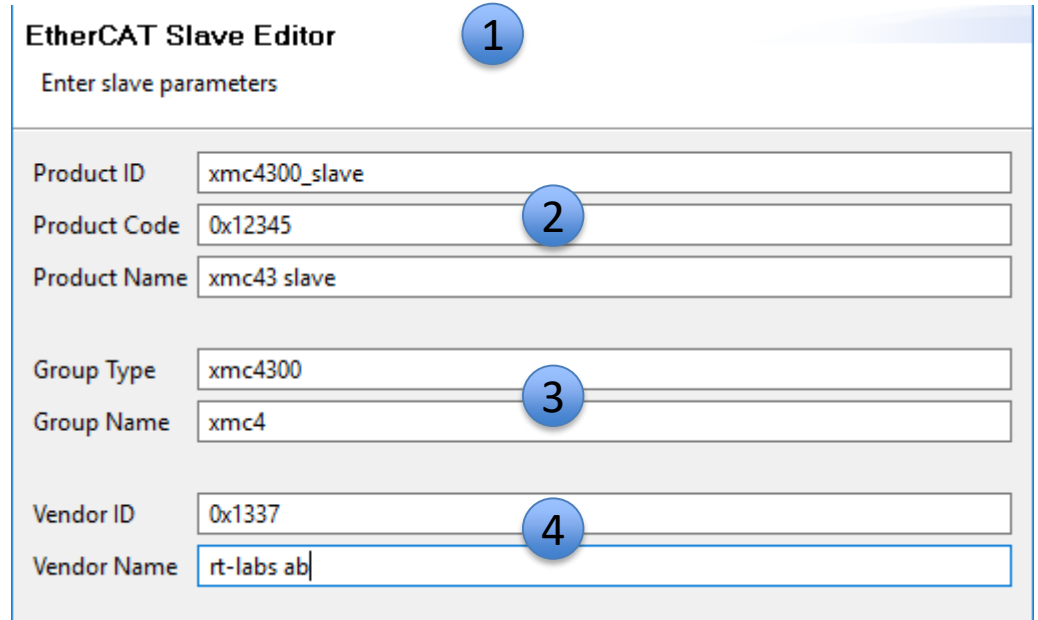
EtherCAT SDK - Slave editor

- 1 Create a slave description project file
- 2 Point out where to place it relative to the work space



EtherCAT SDK- Slave editor

- 1 Enter the base parameters
- 2 Product information, used to uniquely identify the slave device
- 3 Group information, give you possibility to group slaves
- 4 Vendor information, vendor ID is provided by at [ETG](#)



The screenshot shows the 'EtherCAT Slave Editor' window with the title 'Enter slave parameters'. It contains several input fields for configuring a slave device. Blue circles with numbers 1 through 4 are overlaid on the interface to indicate the sequence of steps: 1 points to the title bar, 2 points to the Product Code field, 3 points to the Group Name field, and 4 points to the Vendor ID field.

EtherCAT Slave Editor	
Enter slave parameters	
Product ID	xmc4300_slave
Product Code	0x12345
Product Name	xmc43 slave
Group Type	xmc4300
Group Name	xmc4
Vendor ID	0x1337
Vendor Name	rt-labs ab

EtherCAT SDK -Slave editor

Slave TAB: Edit and add base parameters.

- 1 To modify texts, click the label and edit in-place

The screenshot displays the 'Slave' editor window for a device named 'xmc4300_slave.esx'. The interface is divided into two main sections: 'Slave' and 'Group'.

Slave Section:

- ID:** xmc4300_slave
- Product Code:** 0x12345
- Revision No:** 0
- Name:** A list containing 'English (... xmc43 slave)' with a blue circle '1' highlighting the text. To the right are 'Add', 'Remove', 'Up', and 'Down' buttons.
- URL:** A list with an empty entry, with 'Add', 'Remove', 'Up', and 'Down' buttons to the right.

Group Section:

- Group Type:** xmc4300
- Group Name:** A list containing 'English (... xmc4)' with 'Add', 'Remove', 'Up', and 'Down' buttons to the right.

Vendor Section:

- ID:** 0x1337
- Name:** A list containing 'English (... rt-labs ab)' with 'Add', 'Remove', 'Up', and 'Down' buttons to the right.

EtherCAT SDK - Slave editor

Configuration TAB: Basic configuration settings

- 1 FMMU types
- 2 SM types, address, control, size
- 3 Supported protocols and configuration
- 4 DC – leave as is since the example is a free-run slave

Configuration

FMMU

FMMU Details

Type: Outputs

FMMU0
FMMU1
FMMU2

Add
Remove

Sync Manager

SM

SM0
SM1
SM2
SM3

Add
Remove

SM Details

Type: MBoxOut

Start Address: 0x1000
Default Size: 128
Min Size: 0
Max Size: 0
Control: 0x26

Mailbox

☒ CoE supported
☒ FoE supported
☐ VoE supported

Configuration

	Start	Length
Bootstrap	0x1000	128
Standard	0x1000	128

Dc

Mode

Add
Remove

Mode Details

Name
Description
Assign/Activate

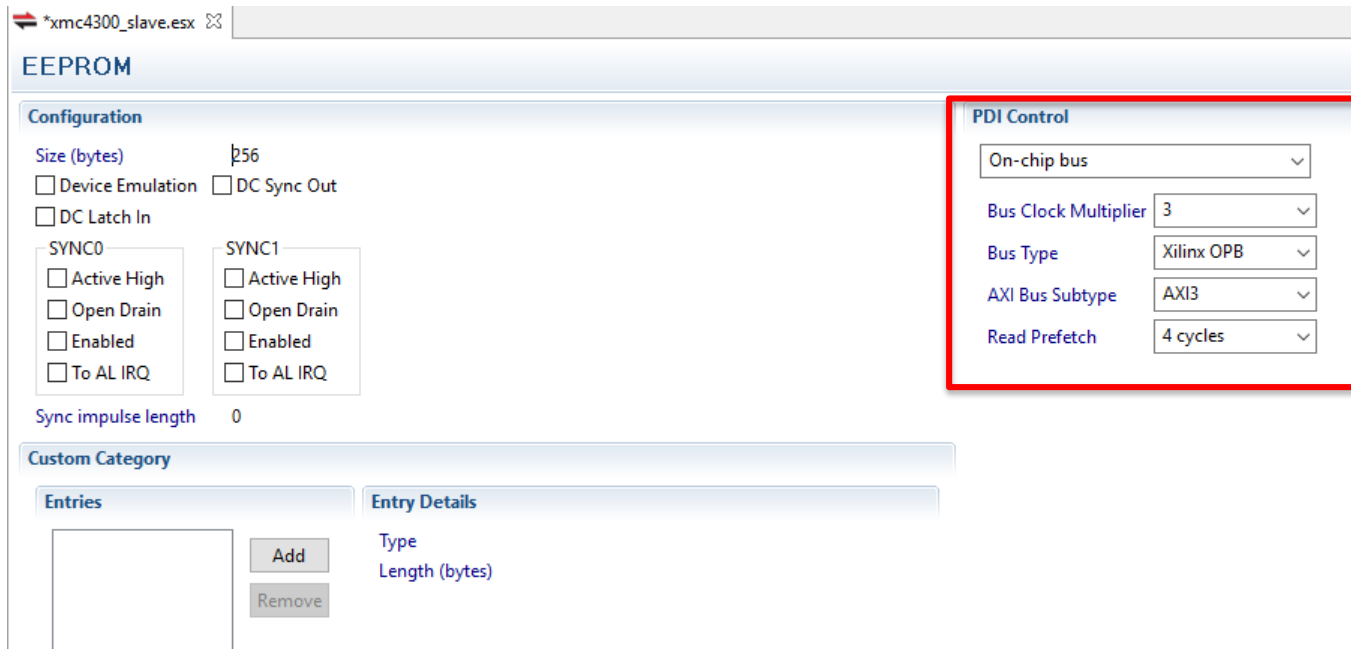
Sync configuration

Cycle Time

SYNC0
SYNC1

EtherCAT SDK - Slave editor

EEPROM TAB: Use this PDI Control settings, they're aligned with ReadOnly ESC values according to the XMC ref manual.



The screenshot shows the EEPROM configuration window for a slave device. The title bar indicates the file path `*xmc4300_slave.esx`. The main tab is labeled "EEPROM".

Configuration

Size (bytes) `256`

☐ Device Emulation ☐ DC Sync Out

☐ DC Latch In

SYNC0

☐ Active High
☐ Open Drain
☐ Enabled
☐ To AL IRQ

SYNC1

☐ Active High
☐ Open Drain
☐ Enabled
☐ To AL IRQ

Sync impulse length `0`

PDI Control (highlighted with a red box)

On-chip bus

Bus Clock Multiplier `3`

Bus Type `Xilinx OPB`

AXI Bus Subtype `AXI3`

Read Prefetch `4 cycles`

Custom Category

Entries

Entry Details

Type

Length (bytes)

EtherCAT SDK - Slave editor

Application TAB: Add application IO, the Slave Editor will automatically use the correct index ranges according to the Modular Device Profile. The specified PDOs will also automatically be added to the SyncManager, RxPDO and TxPDO objects.

- 1 Add PDO record which automatically will be added to the TxPDO/RxPDO objects
- 2 Add PDO record variable with name, datatype and default value
- 3 Add configuration parameters (optional)

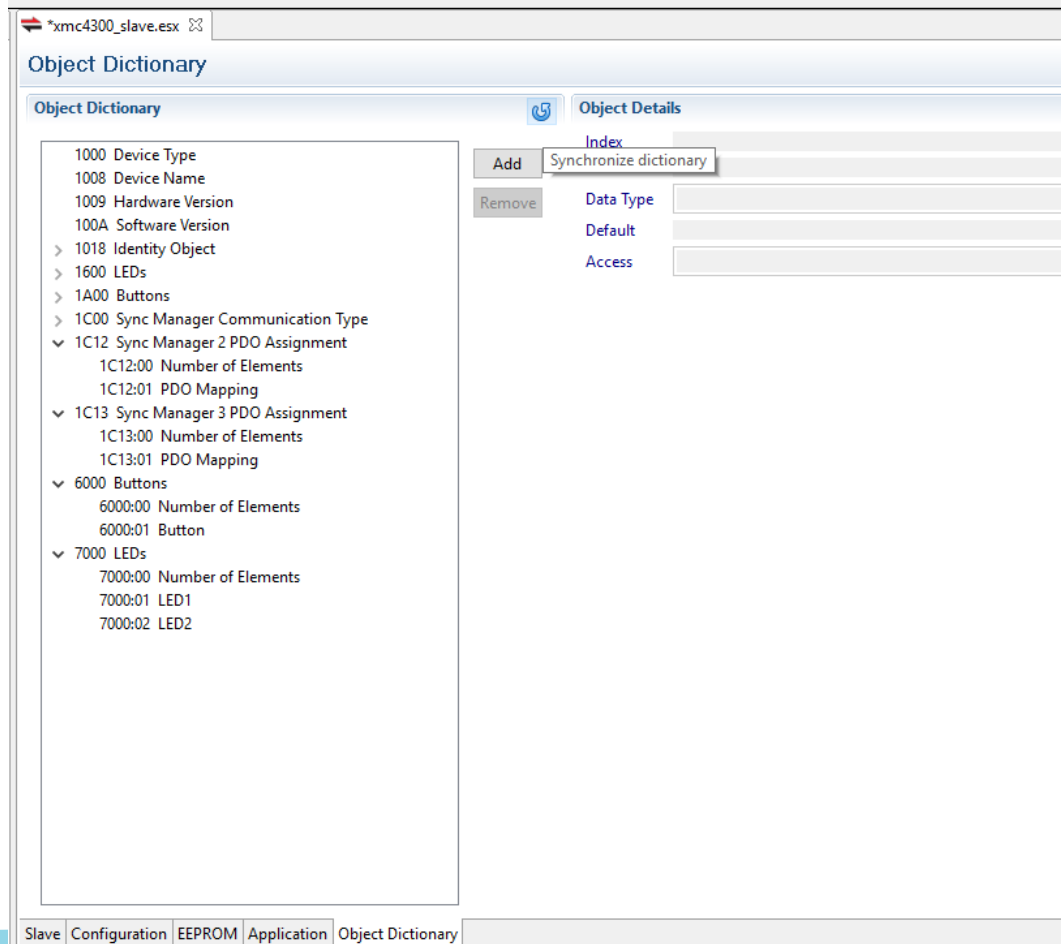
The screenshot shows the 'Application' tab of the EtherCAT SDK Slave Editor. The interface is divided into three main sections: Inputs, Outputs, and Parameters. Each section has a list of items and a 'Variable Details' or 'Record Details' panel on the right.

- Inputs:** A list shows '6000 Buttons' expanded to '01 Button'. A blue circle with the number '1' is next to the list. To the right, the 'Variable Details' panel shows: Index 0x01, Name Button, Data Type UNSIGNED8, and Default 0. A blue circle with the number '2' is next to the 'Data Type' field.
- Outputs:** A list shows '7000 LEDs' expanded to '01 LED1' and '02 LED2'. A blue circle with the number '1' is next to the list. To the right, the 'Variable Details' panel shows: Index 0x02, Name LED2, Data Type UNSIGNED8, and Default 0. A blue circle with the number '2' is next to the 'Data Type' field.
- Parameters:** An empty list. A blue circle with the number '3' is next to the list. To the right, the 'Record Details' panel is empty.

At the bottom, there are tabs for 'Slave', 'Configuration', 'EEPROM', 'Application' (which is selected), and 'Object Dictionary'.



EtherCAT SDK - Slave editor

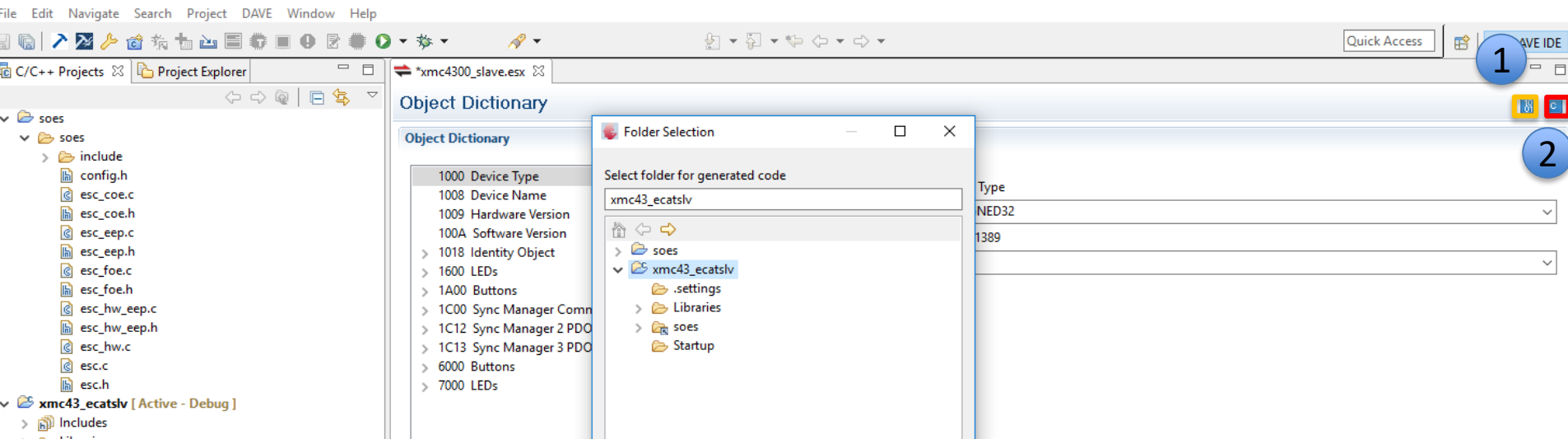
Object Dictionary TAB: Overview of the resulting Object Dictionary



EtherCAT SDK - Slave editor

Generate output:

- 1 Press  button to generate ESI data XML & EEPROM
- 2 Press  button to generate C code, Object Dictionary & Slave application



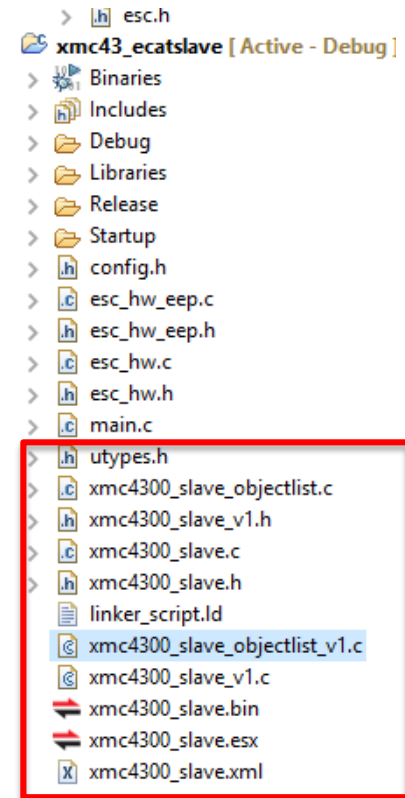
EtherCAT SDK - Slave editor

Overview of the generated output:

- **utypes.h**, types for user defined application data
- <slave_editor project> **_objectlist.c**, Object Dictionary
- <slave_editor project> **_slave.[c,h]**, slave stack application functions and declarations
- <slave_editor project> **_slave.bin**, EEPROM
- <slave_editor project> **_slave.xml**, ESI file

Don't EDIT this files since the will be overwritten if code is generated again.

- Files with post fix _v1 are generated for SOES v1 (old),
exclude from build from Resource Configurations -> Exclude from Build



Workflow

1. Overview and Requirements
2. Setup hardware, DAVE and EtherCAT SDK
3. Define the EtherCAT application with EtherCAT SDK
4. **Implement the application in DAVE**
5. Test the application with EtherCAT SDK

Implement Application – integrate stack

- 1 Add application code in a NOT generated file.
- 2 Add calls to the SOES slave stack , **ecat_slv_init** to initialize the stack and **ecat_slv** to run the stack periodically in free running mode.

```
8
9
10
11 /**
12
13  * @brief main() - Applicati
14  *
15  * <b>Details of function</b>
16  * This routine is the appli
17  */
18
19 int main(void)
20 {
21     /* Placeholder for user ap
22     ecat_slv_init(&config);
23
24     while(1)
25     {
26         ecat_slv();
27     }
28 }
```

Implement Application – configure stack

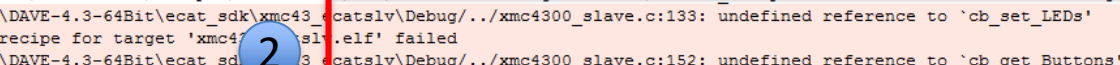
- 1 The stack is configured by passing a configuration arg to **ecat_slv_init**. Consult the user manual for more information on different options, for a free-run slave the configuration below will do. Copy config from samples in soes\applications
- 2 `use_interrupt = 0`
- 3 `watch_dog = <your cycle count>`
- 4 `mbx`, `mb`, `mb_boot`, `pdo_sm` pass values generated from the SlaveEditor tool in `config.h`
- 5 XMC must have an emulated EEPROM handler
- 6 Add application includes

```
18 #include "xmc_gpio.h"
19 #include "config.h"
20 #include "esc_hw.h"
21 #include "xmc4300_slave.h"
22
23 /* Configuration parameters for SOES
24  * SM and Mailbox parameters comes from the
25  * generated config.h
26  */
27 static esc_cfg_t config =
28 {
29     .user_arg = NULL,
30     .use_interrupt = 0,
31     .watchdog_cnt = 2500,
32     .mbxsize = MBXSIZE,
33     .mbxsizeboot = MBXSIZEBOOT,
34     .mbxbuffers = MBXBUFFERS,
35     .mb[0] = (MBX0_sma, MBX0_sml, MBX0_sme, MBX0_smc, 0),
36     .mb[1] = (MBX1_sma, MBX1_sml, MBX1_sme, MBX1_smc, 0),
37     .mb_boot[0] = (MBX0_sma_b, MBX0_sml_b, MBX0_sme_b, MBX0_smc_b, 0),
38     .mb_boot[1] = (MBX1_sma_b, MBX1_sml_b, MBX1_sme_b, MBX1_smc_b, 0),
39     .pdosm[0] = (SM2_sma, 0, 0, SM2_smc, SM2_act),
40     .pdosm[1] = (SM3_sma, 0, 0, SM3_smc, SM3_act),
41     .pre_state_change_hook = NULL,
42     .post_state_change_hook = NULL,
43     .application_hook = NULL,
44     .safeoutput_override = NULL,
45     .pre_object_download_hook = NULL,
46     .post_object_download_hook = NULL,
47     .rxpdo_override = NULL,
48     .txpdo_override = NULL,
49     .esc_hw_interrupt_enable = NULL,
50     .esc_hw_interrupt_disable = NULL,
51     .esc_hw_eep_handler = ESC_eep_handler
52 };
53
54 int main(void)
55 {
56     /* Placeholder for user application code. The while loop below can be
57     ecat_slv_init(&config);
58 }
```

Implement Application – Compile

First time compiling is expected to fail

- 1 Undefined references to `"_binary_sii_eeprom_bin_end"` and `"_binary_sii_eeprom_bin_start"` are extern declared variables in the stack to the emulated EEPROM not yet added. (How to fix in coming slides)
- 2 `cb_set_XX` and `cb_get_YY` are slave stack callbacks for handling our added application IO, these have been generated by the EtherCAT SDK Slave Editor (How to fix in coming slides)



```
CDT Build Console [xmc43_ecatslv]
soes();
^

'Building target: xmc43_ecatslv.elf'
./soes/esc_hw_eep.o: In function `init_flash_data':
C:/Workspaces/DAVE-4.3-64Bit/ecat_sdk/soes/esc_hw_eep.c:300: undefined reference to `_binary_sii_eeprom_bin_end'
'Invoking: ARM-GCC C Linker'
C:/Workspaces/DAVE-4.3-64Bit/ecat_sdk/soes/esc_hw_eep.c:300: undefined reference to `_binary_sii_eeprom_bin_start'
./xmc4300_slave.o: In function `DIG_process':
"C:/DAVEv4-64Bit/DAVE-4.3.2/eclipse/ARM-GCC-4.8.3/bin/arm-none-eabi-gcc -I../linker_scripts -nostdlib -xlinker -gc-sections -spe
C:\Workspaces\DAVE-4.3-64Bit\ecat_sdk\xmc43_ecatslv\Debug\..\xmc4300_slave.c:133: undefined reference to `cb_set_LEDs'
makefile:54: recipe for target `xmc43_ecatslv.elf' failed
C:\Workspaces\DAVE-4.3-64Bit\ecat_sdk\xmc43_ecatslv\Debug\..\xmc4300_slave.c:152: undefined reference to `cb_get_Buttons'
collect2.exe: error: ld returned 1 exit status
make: *** [xmc43_ecatslv.elf] Error 1
```

Implement Application - EEPROM

Right click the Application <Project> and select Properties

- 1 Add an pre-build option to copy and create a linkable object file.
- 2 The only required change is to add your EEPROM bin in the copy command, eg. replace **xmc4300_slave.bin** with your name.

The screenshot shows the IDE's 'Tool Settings' dialog for the 'ARM-GCC C Linker'. The 'Pre-build steps' tab is selected. The 'Command:' field is highlighted with a red box and contains the command: `copy "..\xmc4300_slave.bin" sii_eeprom.bin & "${ARM_GCC_HOME}/bin/arm-none-eabi-objcopy.exe" -I binary -O elf32-littlearm -B arm sii_eeprom.bin sii_eeprom.o`. A red box also highlights the 'Add file path' dialog box, which is open and shows the file path `sii_eeprom.o` in the 'File:' field. The 'Add file path' dialog box has buttons for 'OK', 'Cancel', 'Workspace...', and 'File system...'. The 'Add file path' dialog box is also highlighted with a red box.

- 1
- 2
- 3

- 3 Add the resulting object file to the linker.

Implement Application - GPIO

Add application code in a NOT generated file.

- 1 Add init of GPIO
- 2 Add LEDs callbacks
- 3 Add Buttons callbacks

```
4
5 int main(void)
6 {
7     XMC_GPIO_Init(GPIO_BUTTON1, &gpio_config_btn);
8     XMC_GPIO_Init(GPIO_LED1, &gpio_config_led1);
9     XMC_GPIO_Init(GPIO_LED2, &gpio_config_led2);
10
11     ecat_slv_init(&config);
12
13     while(1)
14     {
15         ecat_slv();
16     }
17 }
```

```
33 /**
34  * This function gets input values and updates Rb.Buttons
35  */
36 void cb_get_Buttons(void)
37 {
38     Rb.Buttons.Button1 = (uint8_t)XMC_GPIO_GetInput(GPIO_BUTTON1);
39 }
40 /**
```

```
40 /**
41  * This function sets output values according to Wb.I
42  */
43 void cb_set_LEDs(void)
44 {
45     if (Wb.LEDs.LED1)
46     {
47         XMC_GPIO_SetOutputHigh(GPIO_LED1);
48     }
49     else
50     {
51         XMC_GPIO_SetOutputLow(GPIO_LED1);
52     }
53
54     if (Wb.LEDs.LED2)
55     {
56         XMC_GPIO_SetOutputHigh(GPIO_LED2);
57     }
58     else
59     {
60         XMC_GPIO_SetOutputLow(GPIO_LED2);
61     }
62 }
63
```

EtherCAT SDK - Agile EtherCAT development

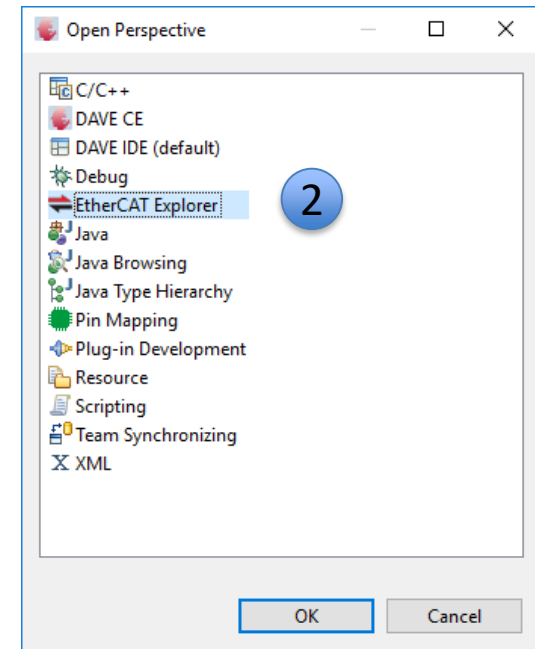
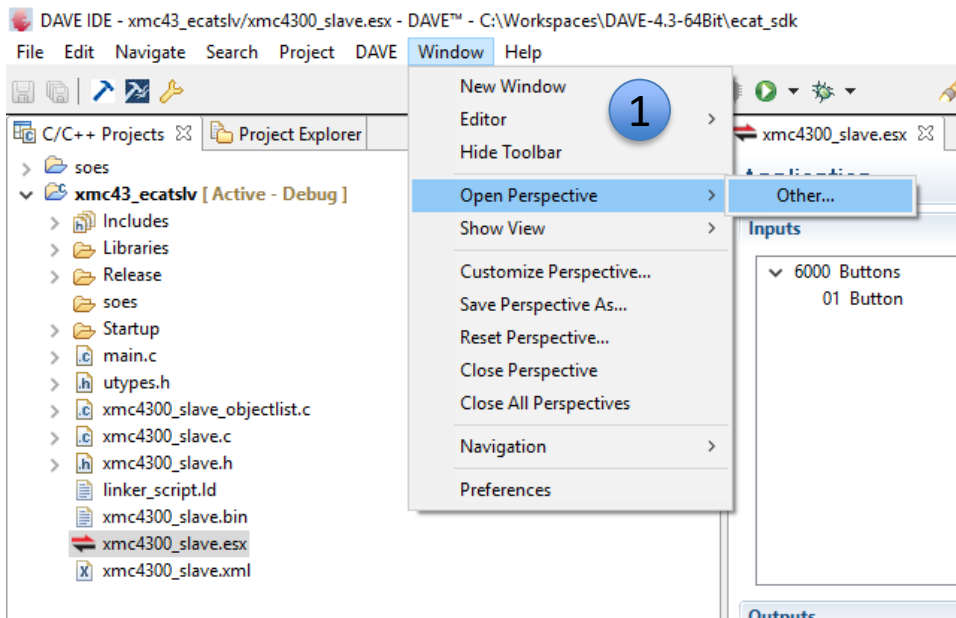
- The whole idea with the slave editor is to keep EEPROM, ESI and OD aligned, you should always be CTT ready.
- To continue editing, just double click the Slave Editor project file, re-generate the code, build and debug.
- Test the slave with EtherCAT SDK – EtherCAT Explorer

Workflow

1. Overview and Requirements
2. Setup hardware, DAVE and EtherCAT SDK
3. Define the EtherCAT application with EtherCAT SDK
4. Implement the application in DAVE
5. Test the application with EtherCAT SDK

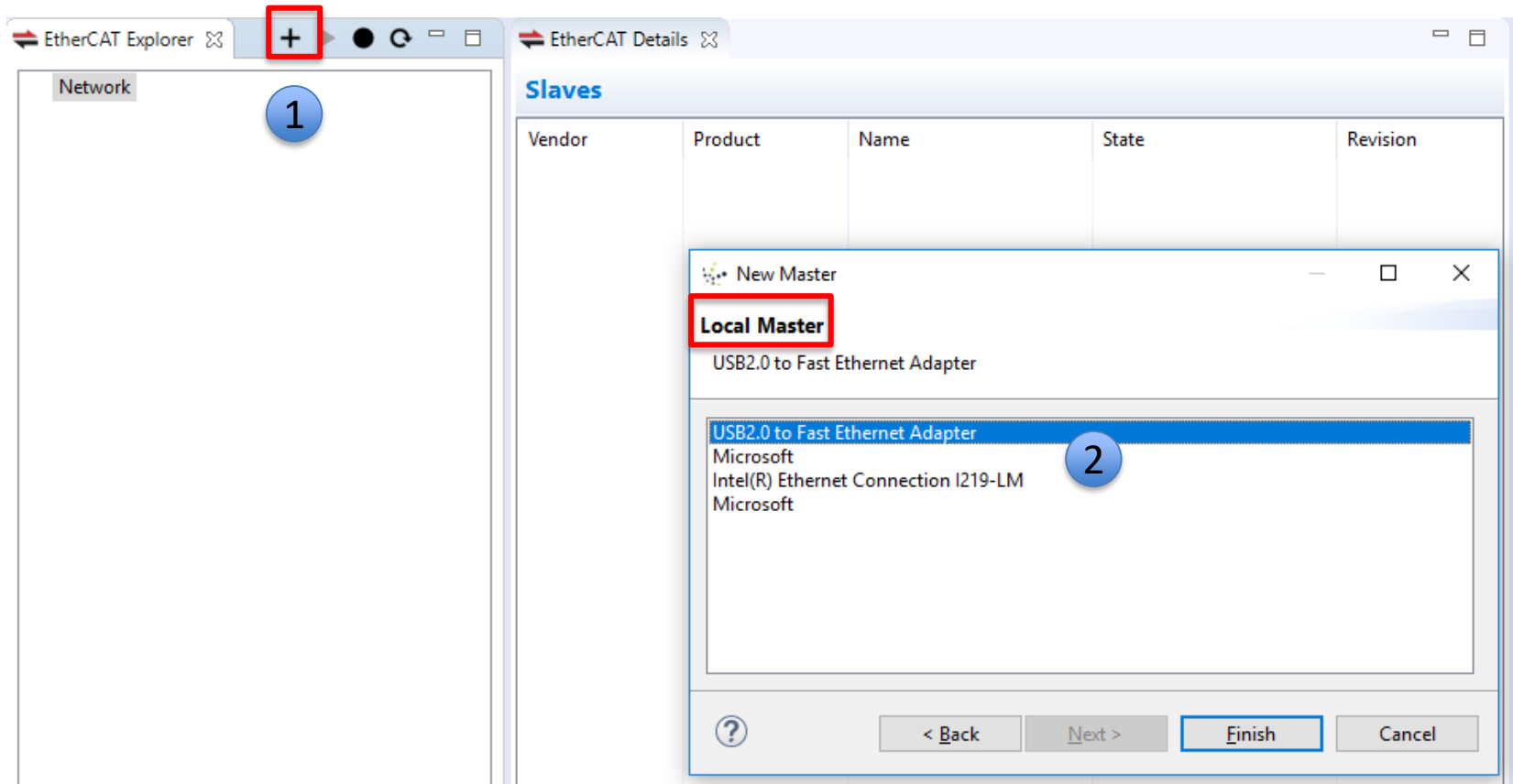
Test Slave – Open EtherCAT Explorer

- 1 Go to menu: Windows -> Open Perspective -> Other ...
- 2 Select EtherCAT Explorer



Test Slave- Select NIC for Master

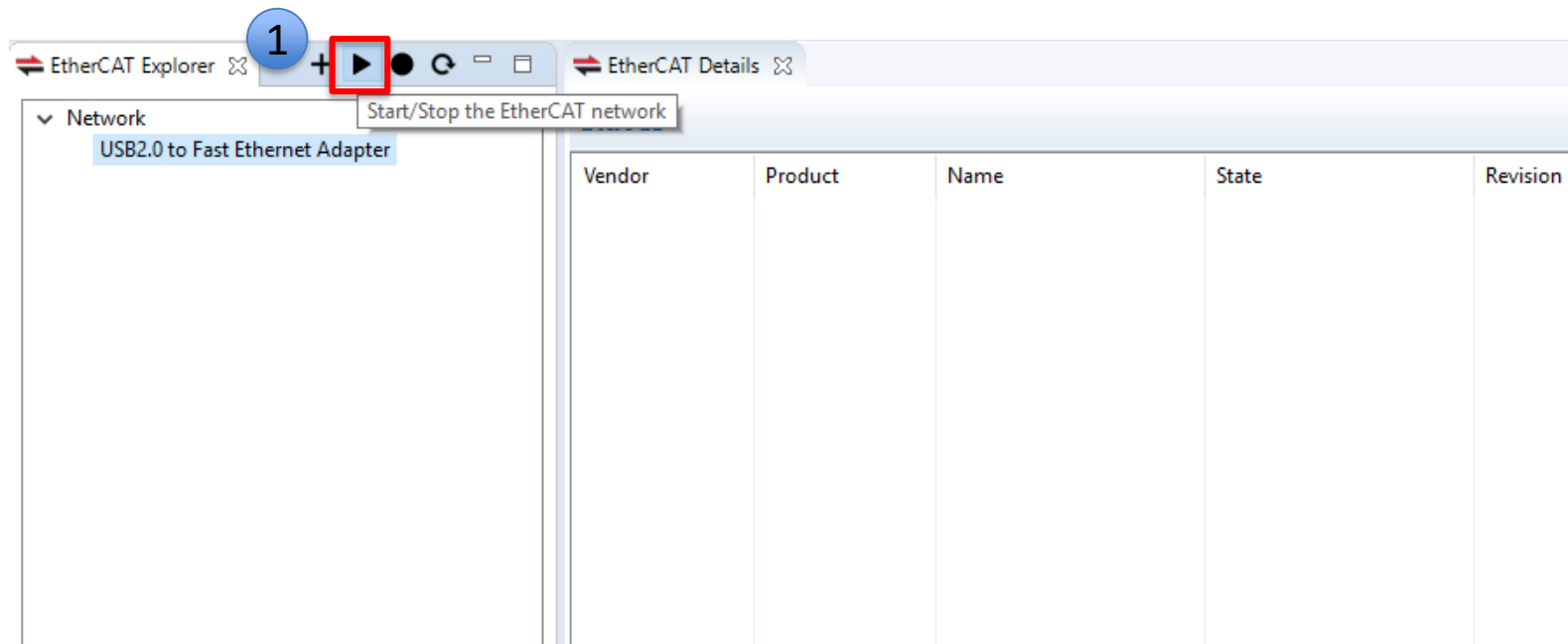
- 1 Open and select NIC for a Local Master (Remote is when running EAP)
- 2 Select one of available adapters (Ethercat Explorer require Winpcap)



Test Slave - EtherCAT Explorer

Make sure the EtherCAT Relax Kit slave is started and connected to the Laptop

- 1 Start the master with the arrow. A started master is shutdown with the arrow.



Test Slave- Examine the ProcessData

- 1 Explore the slave process data,
Action: Press the Relax Kit button , **Result:** The value in the GUI should toggle
Action: Click a LED and enter value 1, **Result:** The LED should go On on the Relax Kit.

The screenshot shows two windows from the EtherCAT software. The 'EtherCAT Explorer' window on the left shows a tree view with 'Network' expanded, then 'Intel(R) Ethernet Connection I219-LM', and finally 'Slave 1 (xmc43_slave)'. The 'EtherCAT Details' window on the right has the 'Process Image' tab selected. It features a 'Filter signals' input field and a table with the following data:

Name	Type	Position	Bit Length	Value
Intel(R) Ethernet Connection I219-LM.Slave 1 (xmc43_slave).LEDgroup0.LED0	UNSIGNED8	0.0	8	1
Intel(R) Ethernet Connection I219-LM.Slave 1 (xmc43_slave).LEDgroup1.LED1	UNSIGNED8	1.0	8	0
Intel(R) Ethernet Connection I219-LM.Slave 1 (xmc43_slave).Buttons.Button1	UNSIGNED8	2.0	8	1

A red box highlights the 'Value' column, and a blue circle with the number '1' is placed over the first row's value '1'. At the bottom of the 'EtherCAT Details' window, there are tabs for 'Slaves', 'Process Image', 'Object Dictionary', 'Configuration', and 'EEPROM'. A 'Console' window is also visible at the very bottom.

EtherCAT SDK - EtherCAT Explorer

- 1 Explore the slave Object Dictionary. Refresh values with arrow

The screenshot shows the EtherCAT Explorer application. The left pane displays a tree view with the following structure:

- Network
 - USB2.0 to Fast Ethernet Adapter
 - Slave 1 (xmc43_slave)

The main pane shows the 'Slave 1 (xmc43_slave)' details. At the top right, there is a 'Refresh' button (circular arrow icon) highlighted with a blue circle and the number '1'. Below this is a 'Filter parameters' input field. The main area contains a table of object dictionary entries:

Index	Name	Datatype	Access	Value
> 1000	Device Type			
> 1008	Device Name			
> 1009	Hardware Version			
> 100A	Software Version			
> 1018	Identity Object			
✓ 1600	LEDs			
1600:00	Number of Elements	UNSIGNED8	RO	2
1600:01	LED1	UNSIGNED32	RO	1879048456
1600:02	LED2	UNSIGNED32	RO	1879048712
✓ 1A00	Buttons			
1A00:00	Number of Elements	UNSIGNED8	RO	1
1A00:01	Button	UNSIGNED32	RO	1610613000
> 1C00	Sync Manager Communi...			
> 1C12	Sync Manager 2 PDO Assi...			
> 1C13	Sync Manager 3 PDO Assi...			
✓ 6000	Buttons			
6000:00	Number of Elements	UNSIGNED8	RO	1
6000:01	Button	UNSIGNED8	RO	1
> 7000	LEDs			

The bottom of the window features a tab bar with the following tabs: Slaves, Process Image, Object Dictionary (selected), Configuration, and EEPROM.

Final step – run CTT

The screenshot displays the EtherCAT Conformance Test Tool (CTT) interface. The main window is titled "Untitled.ctp* - EtherCAT Conformance Test Tool". The interface is divided into several panes:

- Project Explorer:** Shows a tree view of the test configuration. The "Tests" folder is expanded, showing "EtherCAT" and "EtherCAT Devices". Under "EtherCAT Devices", "Slave1 xmc4300_slave[Default]" is selected.
- Test:** The "Slave1 xmc4300_slave[Default]" test is selected. The "Start Page" tab is active, showing the "General Info" section. The "Attributes of DeviceType" table is displayed, showing the configuration for the "xmc4300_slave" device type.
- Logger:** The bottom pane shows the test results. It indicates 211 Success, 0 Warnings, 0 Errors, 135 Skipped, 1163 Outputs, and 4770 Verbose. The log entries show successful completion of various tests.

The "Attributes of DeviceType" table is as follows:

Attributes of DeviceType	
Invisible	
Physics	YY
Crc32	
Device Type	
Type	xmc4300_slave
HideType	No Items
Alternative Type	No Items
Name	Items: 1
Comment	No Items
URL	No Items
Info	not specified
Group Type	xmc4300
Profile	Items: 1
Fmmu	Items: 3
Sync Manager	Items: 4
RxPdps	Items: 1
TxPdps	Items: 1
Mailbox	specified
Distributed Clocks	not specified
Slots	not specified

The Logger pane shows the following log entries:

Log No.	Time Stamp	Test Case	Info
1	2017-01-21 17:11:24:467	Secondary V...	Successful Valid Vendor ID in ESI file (0x1337)
3	2017-01-21 17:11:24:492	Device.Revis...	Success ESI element Device:RevisionNo is used.
11	2017-01-21 17:11:24:507	ESI Port Typ...	Success Port configuration elements in ESI are valid.

Licensed to: RT-Labs AB

EtherCAT provided by rt-labs

Visit us at :

<http://www.rt-labs.com/ethercat/why-choose-rt-labs/>

<https://github.com/OpenEtherCATsociety>

