

# XMC4800 EtherCAT® generic slave example

Getting started V1.0  
July 2016

# Agenda

1

Overview and requirements

2

Setup

3

Description

4

How to test

# Agenda

1

Overview and requirements

2

Setup

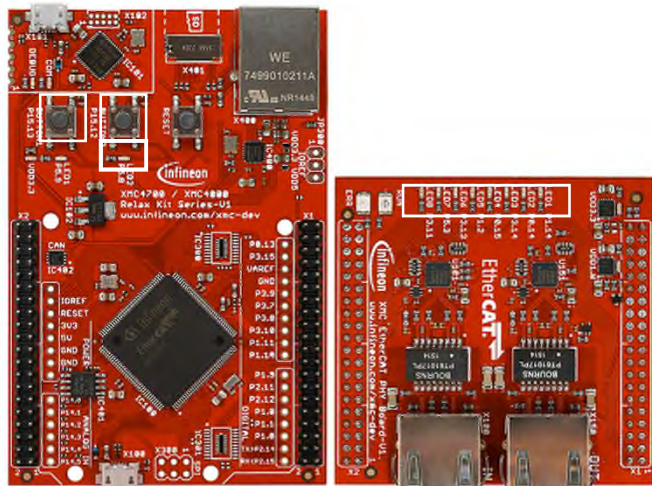
3

Description

4

How to test

# Overview



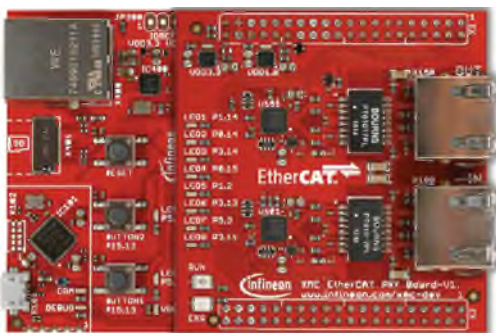
This example demonstrates the usage of an generic EtherCAT® slave node on „XMC4800 Relax EtherCAT® Kit“.

The term „generic“ refers to the fact, that the data structures shared between master and slave do not have a dedicated application specific purpose.

While reviewing this example you will see in output direction the EtherCAT® master controlling the 8 LEDs on the „XMC™ EtherCAT® PHY Board“ and dimming LED2 of the Relax Kit. In input direction you will monitor inside the master device the status of the buttons available on the Relax Kit.

You will observe inside code how you can modify the mapping of the generic data structures to the I/Os for your own evaluations and testing.

# Requirements



XMC4800 Relax EtherCAT® Kit



RJ45 Ethernet Cable



Windows Laptop installed

- DAVE™ v4 (Version 4.1.4 or higher)
- TwinCAT2 or TwinCAT3 Master PLC



Micro USB Cable (Debugger connector)

# Requirements

## Free downloads



TwinCAT2 (30 day trial)  
(used inside this document as reference)  
Link: [Download TwinCAT2](#)

or



TwinCAT3 (no trial period; usability limited)  
Link: [Download TwinCAT3](#)



DAVE™ (v4.1.4 or higher)  
Link: [Download DAVE \(Version 4\)](#)

# Agenda

1

Overview and requirements

2

Setup

3

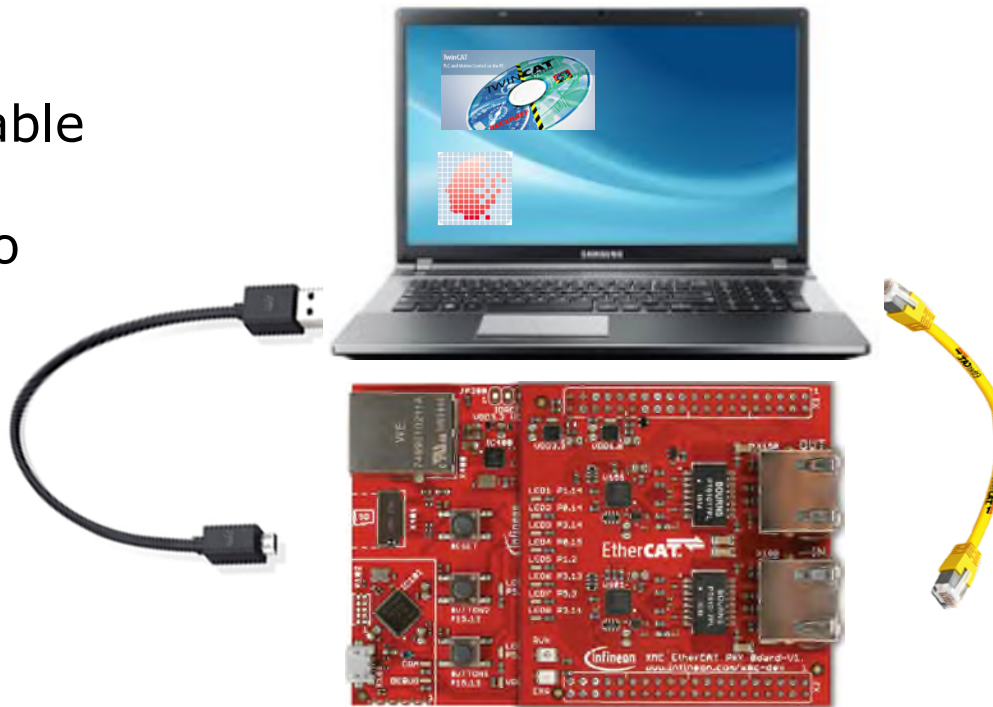
Description

4

How to test

# Setup Hardware

Micro usb cable  
debugger  
connected to  
X101 debug  
connector



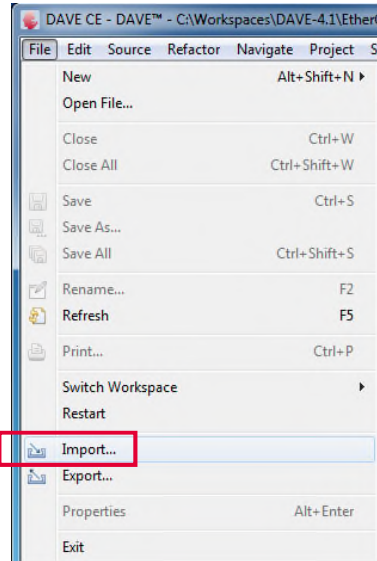
Ethernet cable  
connected to  
in-port



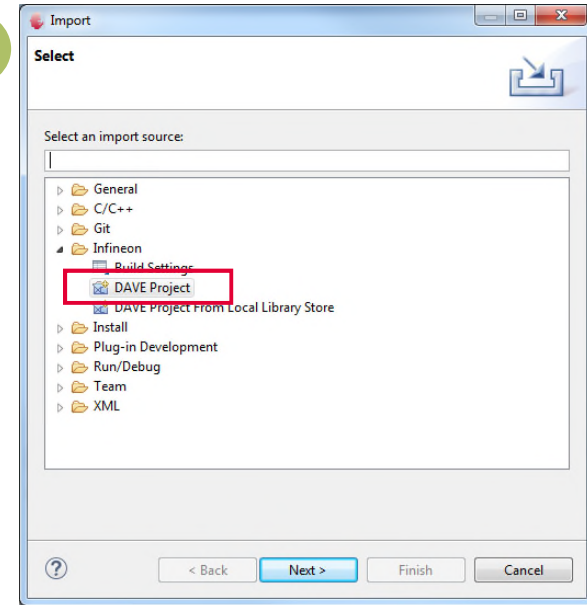
# Setup

## Import example project into DAVE™ & build

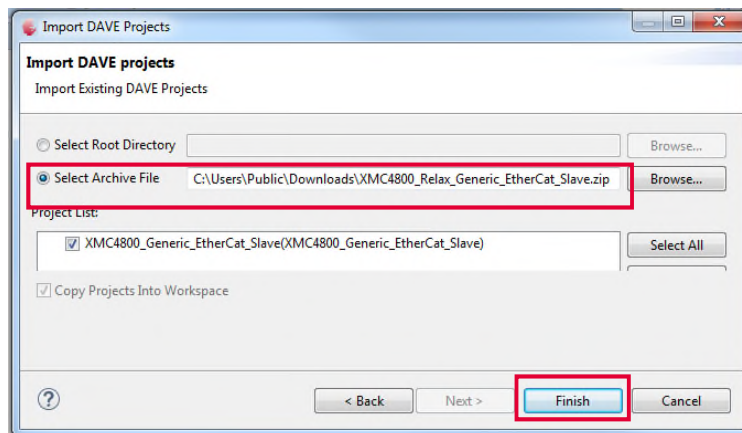
1



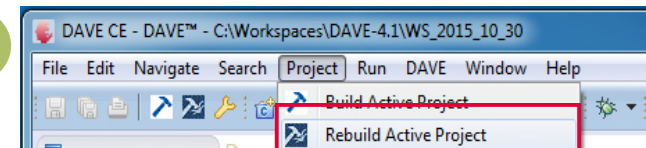
2



3



4



1. Inside the workspace of the DAVE™ example project you find the device description XML-file:  
XMC4800\_Relax\_Generic\_EtherCat\_Slave.xml
2. Copy the device description file to your TwinCAT installation:  
C:\TwinCAT\Io\EtherCAT
3. If TwinCAT System Manager is currently running already, exit TwinCAT.
4. Start TwinCAT. You should see device description cache being re-worked automatically because the new device is added.

# Agenda

1

Overview and requirements

2

Setup

3

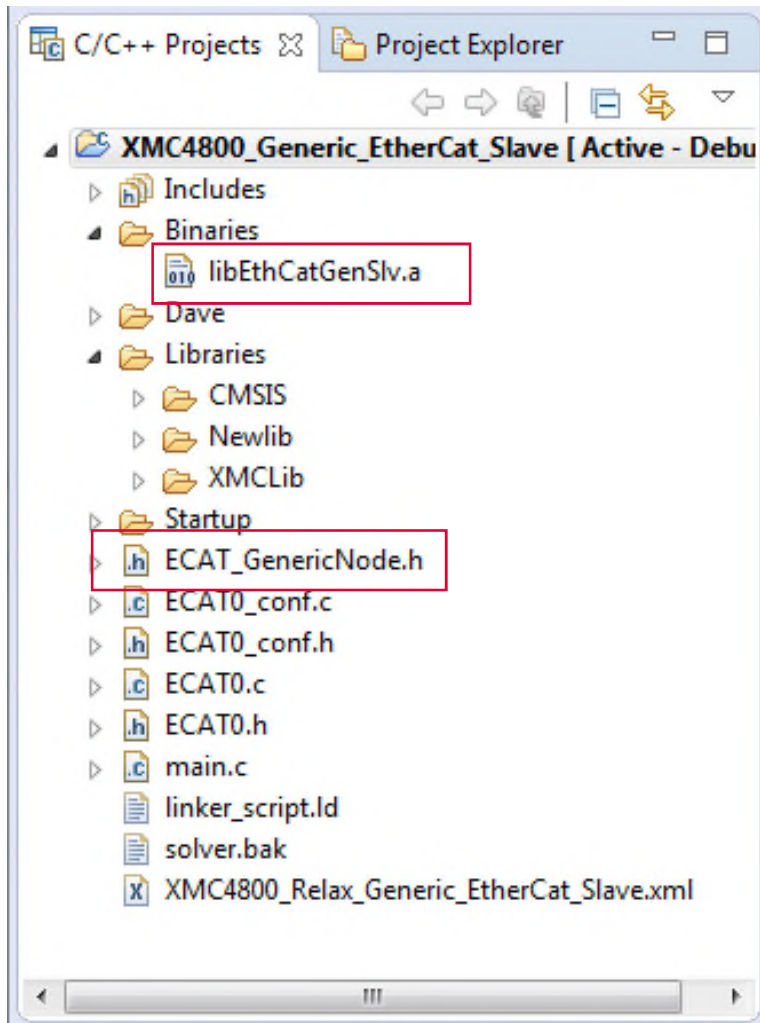
Description

4

How to test

# Description

## Generic EtherCAT® slave library



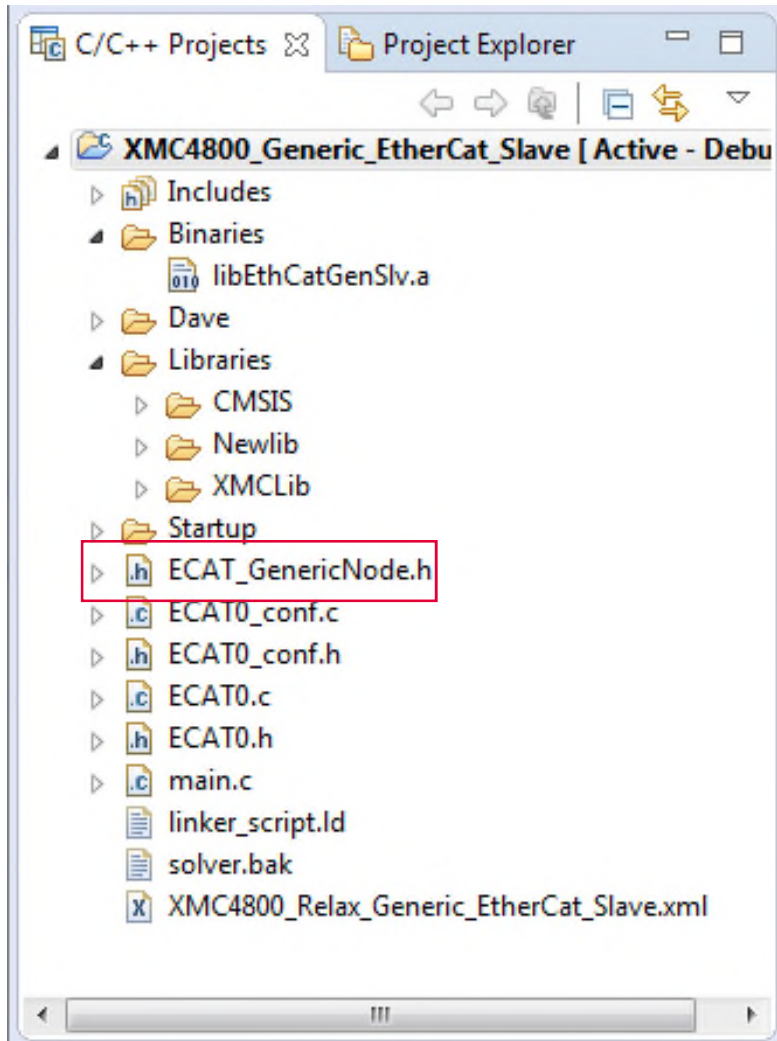
The state machine of the generic slave is implemented inside the library libEthCatGenSlv.a.

ATTENTION: This library is for demonstration purpose within this example only and is not certified, not conform and not planned to be used in any other context than this example here.

Established tool providers may help you in your product development and certification. A list of vendors can be found here: <https://www.ethercat.org/en/products.html>

# Description

## Generic EtherCAT<sup>®</sup> slave library



The API of the generic slave is simple:

### 1. ECAT\_init

To initialize the slave device

### 2. ECAT\_registerCb

To register a callback.

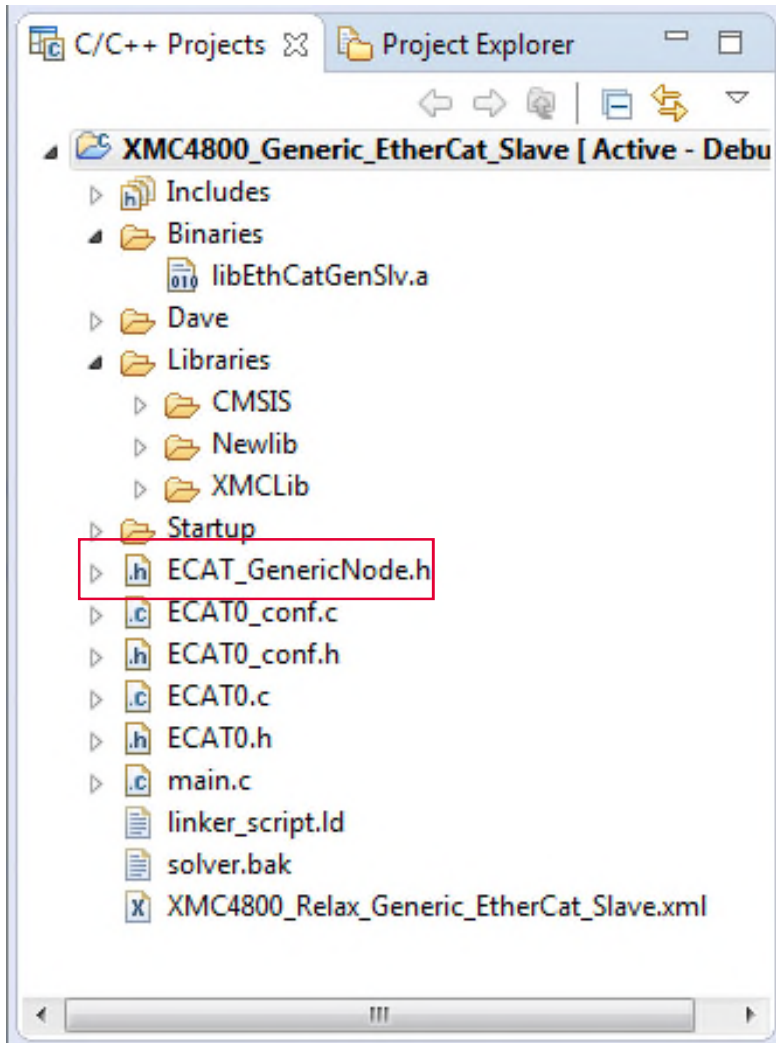
Inside the callback the I/O data from/to master is processed and e.g. connected to hardware (ports, timers, PWM,...)

### 3. ECAT\_main\_loop

Main loop called inside while(1) to process the state machine

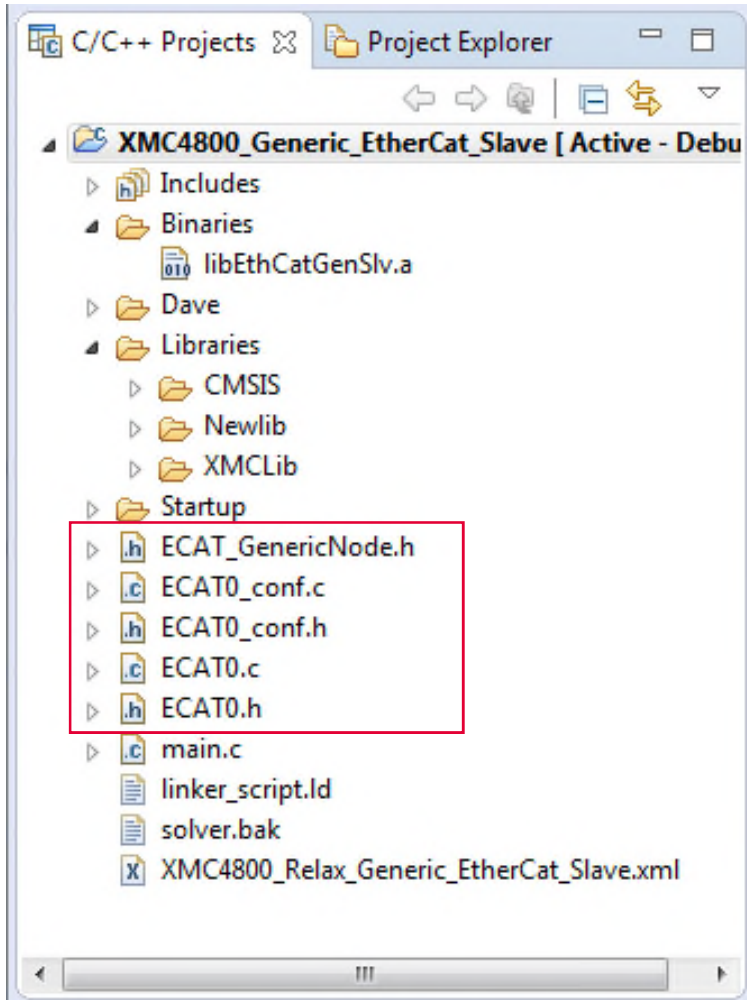
# Description

## Generic EtherCAT<sup>®</sup> slave library



The size of generic data in both directions is 4x16-bit and 8x1-bit.

Check the types IN\_object / OUT\_object inside the headerfile ECAT\_GenericNode.h for details.



Inside the ECAT0\_\* files the physical setup of the EtherCAT® ports is provided.

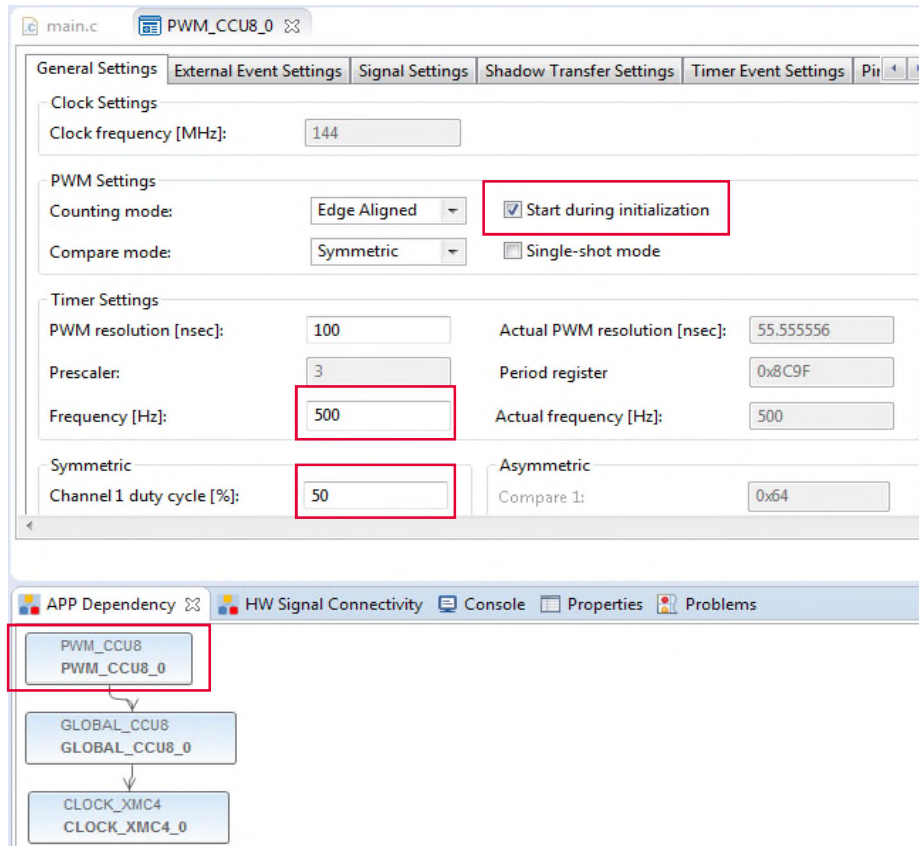
For the example provided, the configuration is fixed to the LQFP144 which fits to the XMC4800 Relax EtherCAT® Kit.

Inside main() during initialization the port configuration is set in place by calling ECAT0\_Init.

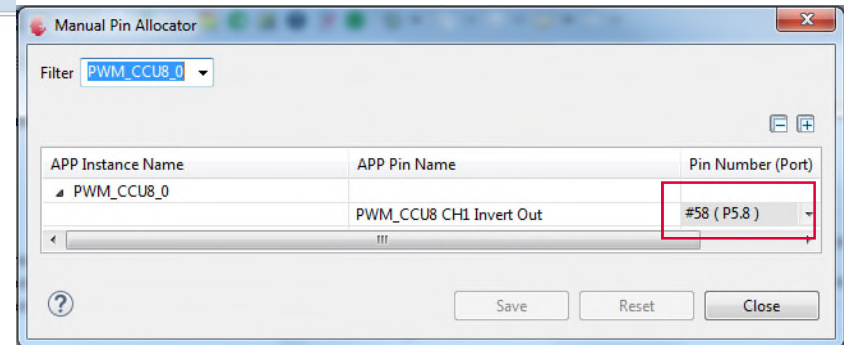


# Description

## Setup PWM for dimmable LED2 on P5.8



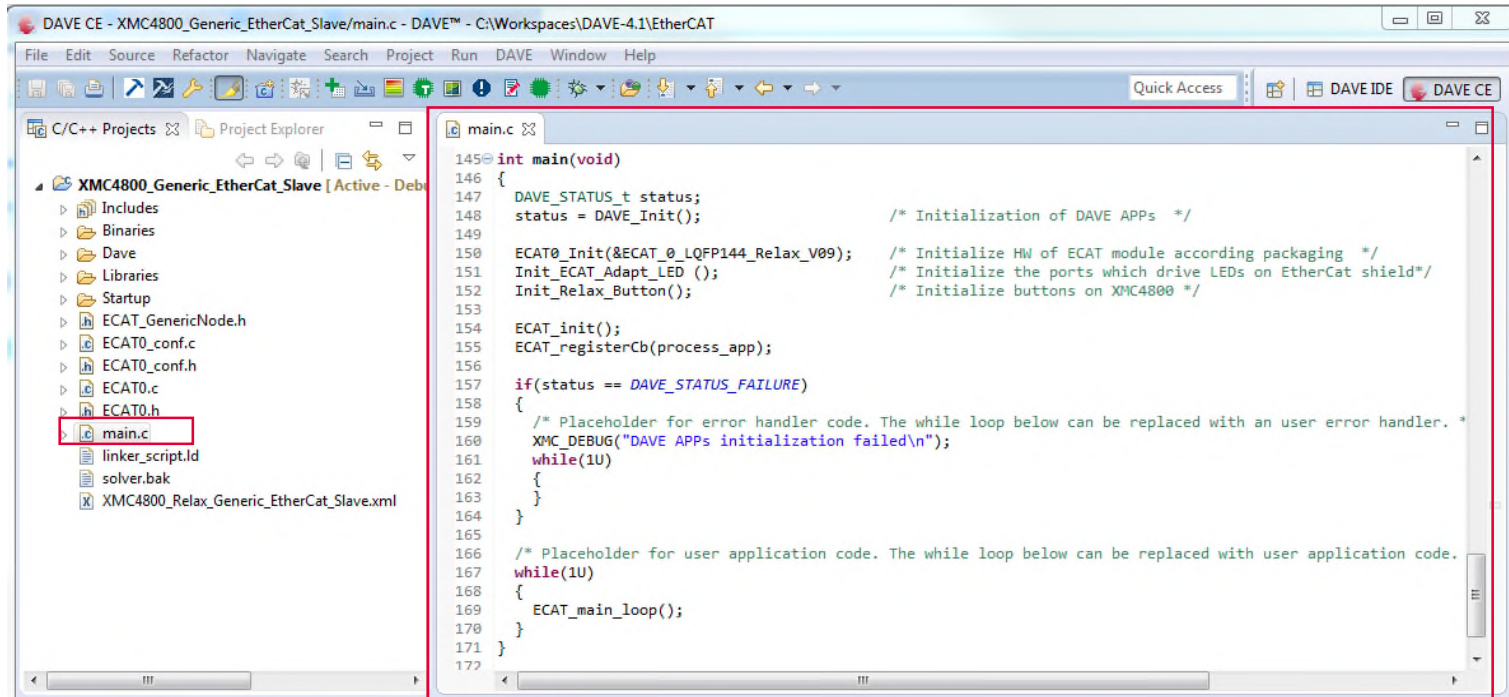
For driving the dimmable LED2 on the XMC4800 Relax EtherCAT® Kit PWM\_CCU8 is used. The PWM is set to start during initialization, frequency is 500 Hz and the initial duty cycle is set to 50%. The pin P5.8 is allocated to channel1 of CCU8 inside manual pin allocator (right-click PWM\_CCU8 APP).





# Description

## Initialization inside main.c

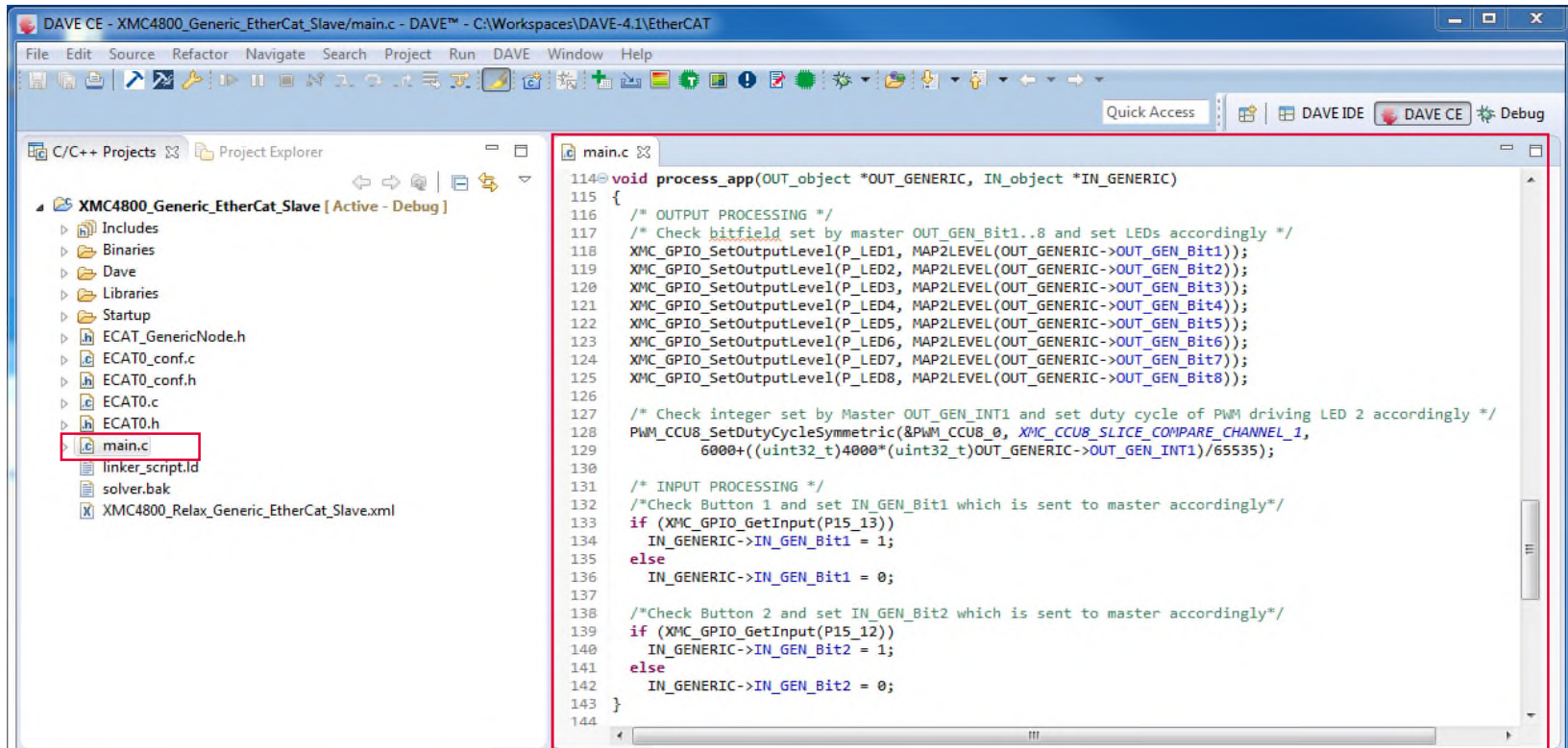


```
145 int main(void)
146 {
147     DAVE_STATUS_t status;
148     status = DAVE_Init();           /* Initialization of DAVE APPs */
149
150     ECAT0_Init(&ECAT_0_LQFP144_Relax_V09); /* Initialize HW of ECAT module according packaging */
151     Init_ECAdapt_LED ();               /* Initialize the ports which drive LEDs on EtherCat shield*/
152     Init_Relax_Button();              /* Initialize buttons on XMC4800 */
153
154     ECAT_init();
155     ECAT_registerCb(process_app);
156
157     if(status == DAVE_STATUS_FAILURE)
158     {
159         /* Placeholder for error handler code. The while loop below can be replaced with an user error handler. */
160         XMC_DEBUG("DAVE APPs initialization failed\n");
161         while(1U)
162         {
163         }
164     }
165
166     /* Placeholder for user application code. The while loop below can be replaced with user application code. */
167     while(1U)
168     {
169         ECAT_main_loop();
170     }
171 }
172
```

Inside main() the DAVE™ system (PWM CCU8) and the ports of the EtherCAT® module are initialized. InitECAT\_Adapt\_LED() and Init\_Relax-Button() are used to initialize the buttons and LED1 to 8 of the „XMC™ EtherCAT® PHY Board“. Finally the generic EtherCAT® slave node is initialized, the callback is registered and the ECAT\_main\_loop is called cyclic to process the state machine.

# Description

## Callback to process input and output



```
DAVE CE - XMC4800_Generic_EtherCat_Slave/main.c - DAVE™ - C:\Workspaces\DAVE-4.1\EtherCAT
File Edit Source Refactor Navigate Search Project Run DAVE Window Help
Quick Access DAVE IDE DAVE CE Debug

C/C++ Projects Project Explorer
XMC4800_Generic_EtherCat_Slave [Active - Debug]
  Includes
  Binaries
  Dave
  Libraries
  Startup
  ECAT_GenericNode.h
  ECAT0_conf.c
  ECAT0.c
  ECAT0.h
  main.c
  linker_script.ld
  solver.bak
  XMC4800_Relas_Generic_EtherCat_Slave.xml

main.c
114 void process_app(OUT_object *OUT_GENERIC, IN_object *IN_GENERIC)
115 {
116     /* OUTPUT PROCESSING */
117     /* Check bitfield set by master OUT_GEN_Bit1..8 and set LEDs accordingly */
118     XMC_GPIO_SetOutputLevel(P_LED1, MAP2LEVEL(OUT_GENERIC->OUT_GEN_Bit1));
119     XMC_GPIO_SetOutputLevel(P_LED2, MAP2LEVEL(OUT_GENERIC->OUT_GEN_Bit2));
120     XMC_GPIO_SetOutputLevel(P_LED3, MAP2LEVEL(OUT_GENERIC->OUT_GEN_Bit3));
121     XMC_GPIO_SetOutputLevel(P_LED4, MAP2LEVEL(OUT_GENERIC->OUT_GEN_Bit4));
122     XMC_GPIO_SetOutputLevel(P_LED5, MAP2LEVEL(OUT_GENERIC->OUT_GEN_Bit5));
123     XMC_GPIO_SetOutputLevel(P_LED6, MAP2LEVEL(OUT_GENERIC->OUT_GEN_Bit6));
124     XMC_GPIO_SetOutputLevel(P_LED7, MAP2LEVEL(OUT_GENERIC->OUT_GEN_Bit7));
125     XMC_GPIO_SetOutputLevel(P_LED8, MAP2LEVEL(OUT_GENERIC->OUT_GEN_Bit8));
126
127     /* Check integer set by Master OUT_GEN_INT1 and set duty cycle of PWM driving LED 2 accordingly */
128     PWM_CCUB_SetDutyCycleSymmetric(&PWM_CCUB_0, XMC_CCUB_SLICE_COMPARE_CHANNEL_1,
129     6000+((uint32_t)4000*(uint32_t)OUT_GENERIC->OUT_GEN_INT1)/65535);
130
131     /* INPUT PROCESSING */
132     /* Check Button 1 and set IN_GEN_Bit1 which is sent to master accordingly */
133     if (XMC_GPIO_GetInput(P15_13))
134         IN_GENERIC->IN_GEN_Bit1 = 1;
135     else
136         IN_GENERIC->IN_GEN_Bit1 = 0;
137
138     /* Check Button 2 and set IN_GEN_Bit2 which is sent to master accordingly */
139     if (XMC_GPIO_GetInput(P15_12))
140         IN_GENERIC->IN_GEN_Bit2 = 1;
141     else
142         IN_GENERIC->IN_GEN_Bit2 = 0;
143 }
144
```

Inside the callback function the binary output data (master->slave) is used to set the level of LED1 to 8 of „XMC™ EtherCAT® PHY Board“. The integer output data is used to set the duty cycle of the dimmable LED2. Finally the states of the buttons are checked and propagated to the input data (slave->master).

# Agenda

1

Overview and requirements

2

Setup

3

Description

4

How to test

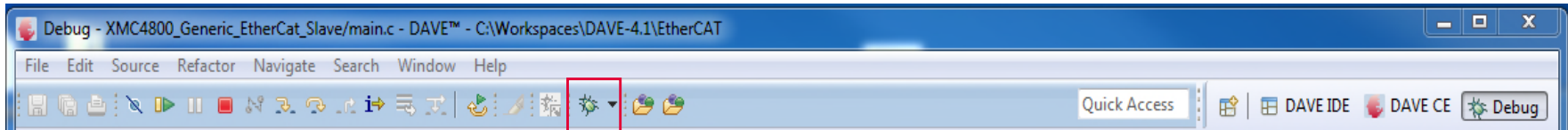
# How to test

## Start the slave to run

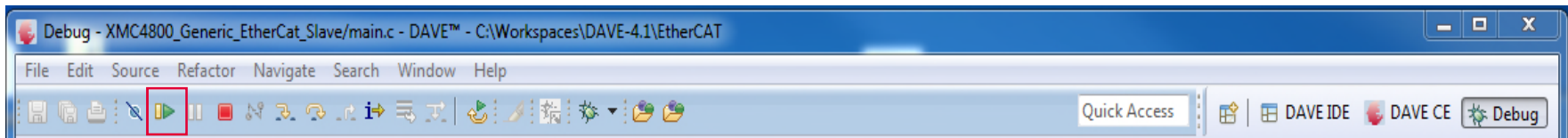


### ACTIONS

1. Download software to XMC4800 and start debugger



2. Start the software to run



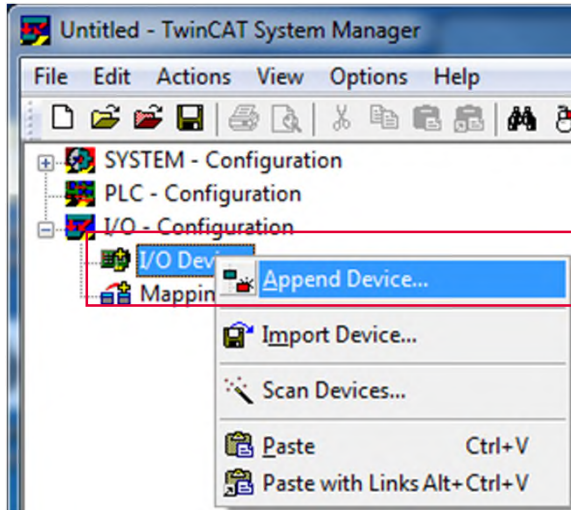
### OBSERVATIONS

1. The ERR-LED on the "XMC™ EtherCAT® PHY Board" will turn on and immediately turn off again.
2. The LED2 on the "XMC4800 Relax EtherCAT® Kit" will remain turned on.

# How to test

## Start the TwinCAT 2 master to run (1/4)

1

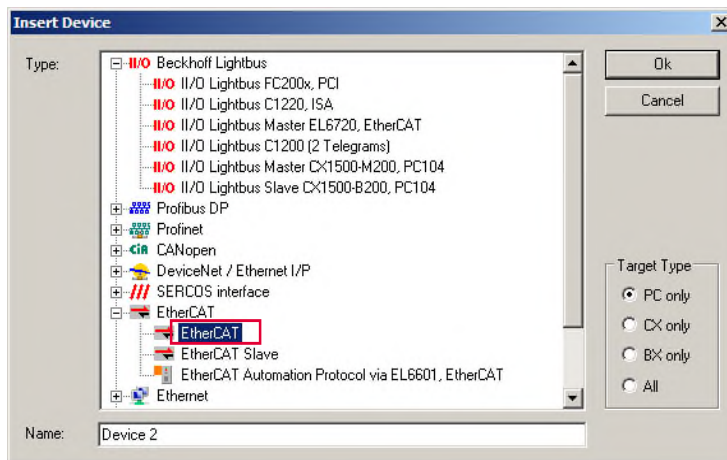


### ACTIONS

After starting the TwinCAT System Manager from windows start menu:

- 1 Right Click I/O-Devices and select „Append Device...”
- 2 Create an EtherCAT® master device by double click

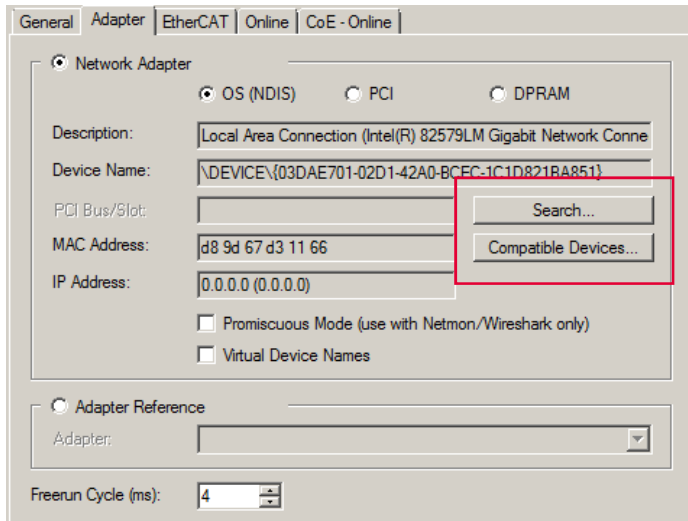
2



# How to test

## Start the TwinCAT 2 master to run (2/4)

3



### ACTIONS

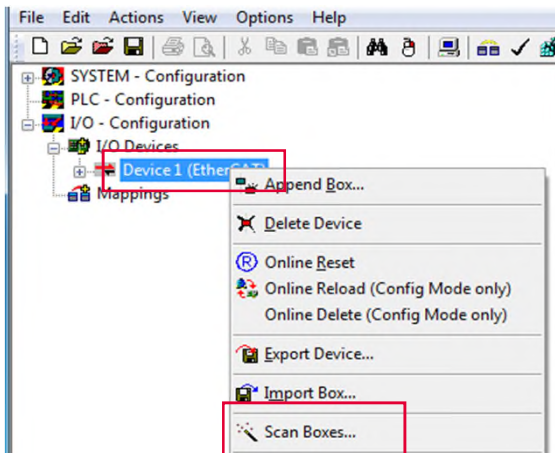
3 Select the network adapter you want to use (search and select).

Application hint:

In case the device is not found please install the respective device driver by following the instructions given by TwinCAT through the „Compatible Devices...” button.

4 Right Click EtherCAT® master and select „Scan Boxes...”.

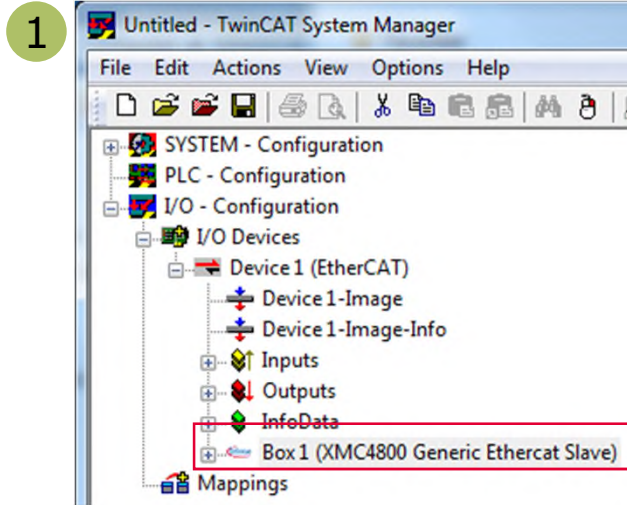
4





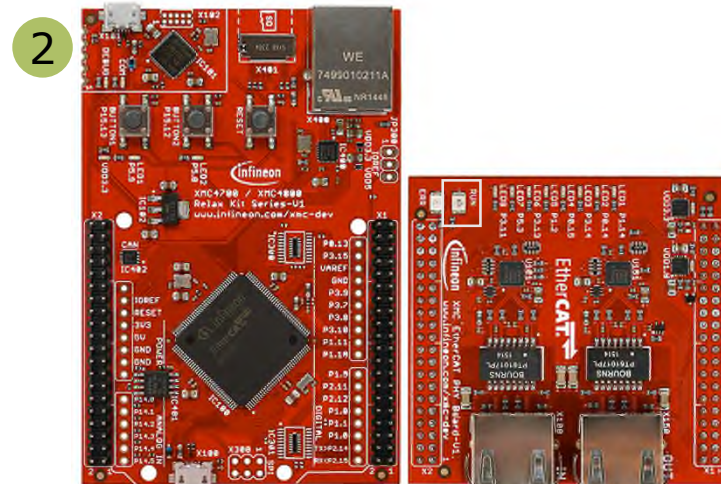
# How to test

## Start the TwinCAT 2 master to run (3/4)



### OBSERVATIONS

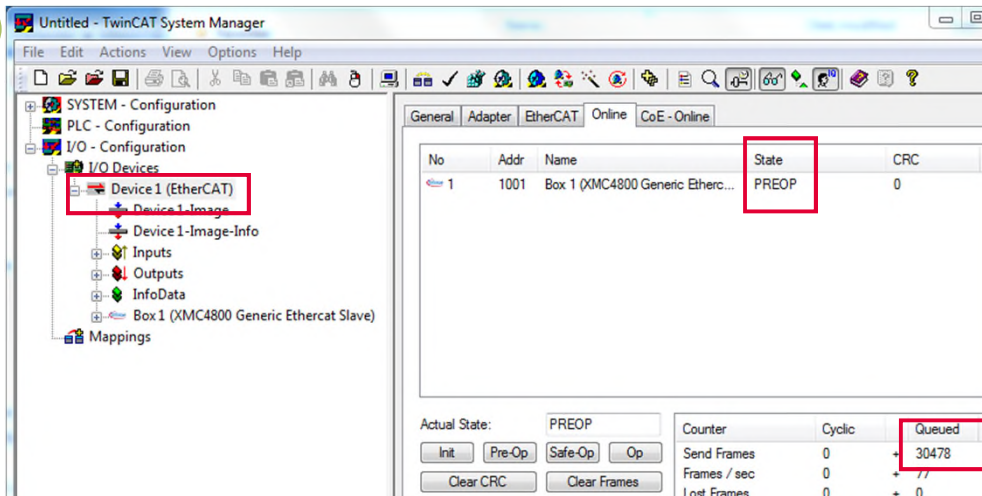
- 1 The slave appears as a node on the bus of the EtherCAT® master.
- 2 The RUN-LED is flashing indicating PREOP-state.



# How to test

## Start the TwinCAT 2 master to run (4/4)

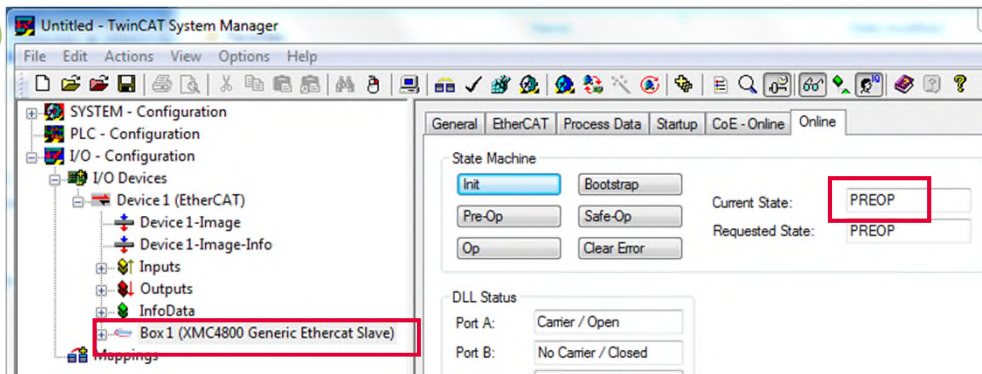
3



### OBSERVATIONS

3 EtherCAT® master view:  
Inside the EtherCAT® master online state you see the queued frames counting up, the connected slave and its PREOP state.

4



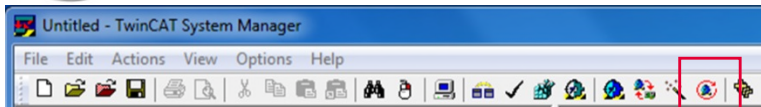
4 EtherCAT® slave view:  
The PREOP-state of the slave is indicated within the TwinCAT system manager.



# How to test Setting slave to operational mode



## ACTION

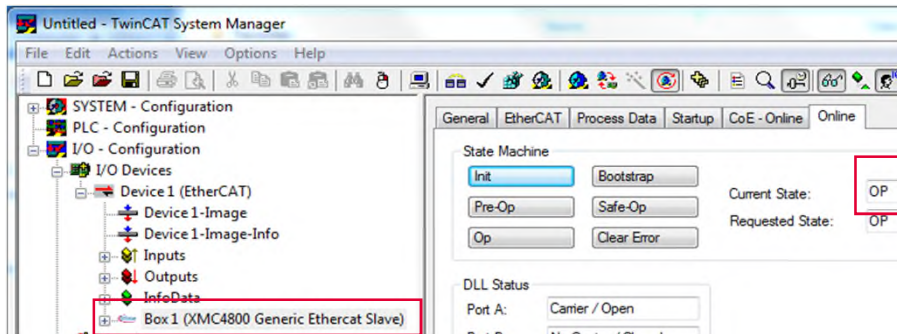


Set master device to free run mode



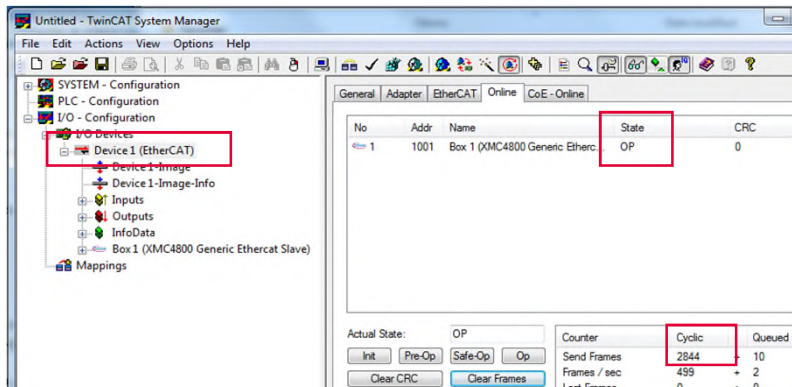
## OBSERVATIONS

1



1 EtherCAT® slave view:  
Online status of slave shows  
the slave in OP state.

2



2 EtherCAT® master view:  
Online status of master  
shows the slave in OP state.  
Frames are no more queued.  
Cyclic counter is  
incrementing.

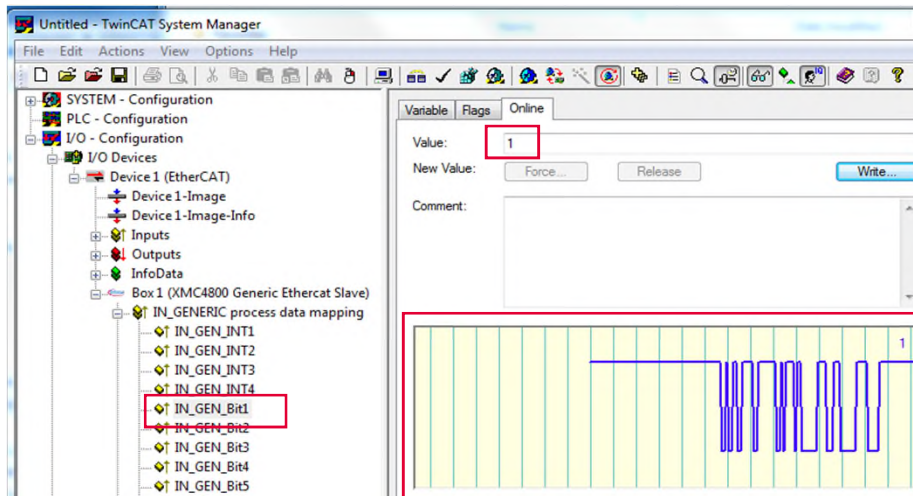
3 „XMC™ EtherCAT® PHY  
Board“:  
RUN-LED is static turned on  
indicating OP-state.

# How to test Monitoring slave inputs on master



## ACTIONS

While pushing Button1 on „XMC4800 Relax EtherCAT® Kit“ button state is updated on host.



State of IN\_GEN\_Bit1 changes according state of BUTTON1. Same is true for IN\_GEN\_Bit2 and BUTTON2.

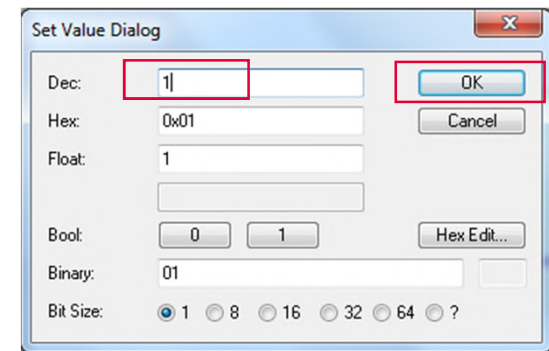
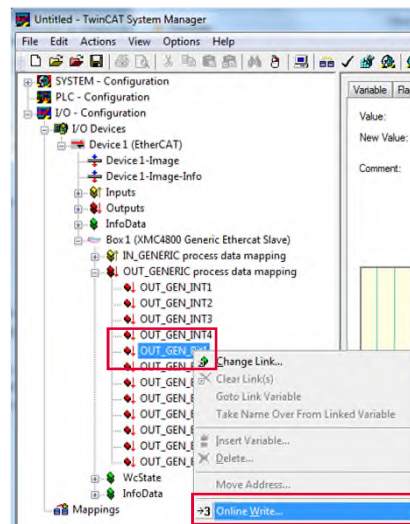
# How to test

## Setting slave outputs on master (1/2)



### ACTIONS

Right click on OUT\_GEN\_Bit1 of the slave node and select „Online Write...” inside the context menu. Change the value from 0 to 1 to switch on LED1/ from 1 to 0 to switch off LED1.



### OBSERVATION

LED1 „XMC™ EtherCAT® PHY Board” is turned on/off according to OUT\_GEN\_Bit1 setting.

# How to test

## Setting slave outputs on master (2/2)



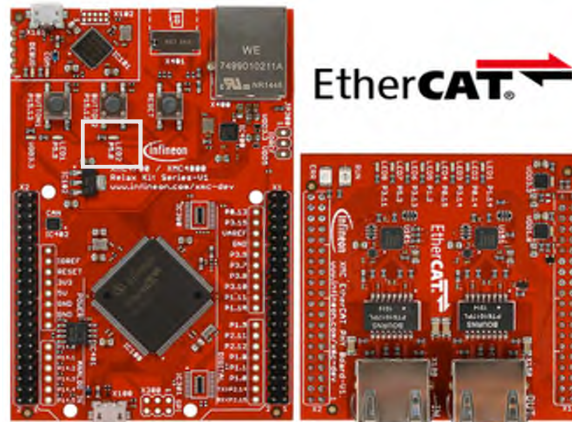
### ACTION

1. Right click on OUT\_GEN\_INT1 of the slave node and select „Online Write...” inside the context menu. Change the value from 0 to 50000.



### OBSERVATION

1. Brightness of LED2 on „XMC4800 Relax EtherCAT® Kit” is dimmed. Setting OUT\_GEN\_INT1 to 65535 the LED2 is turned off. The lower the value, the brighter the LED2.



# Support material

## Collaterals and Brochures



- › Product Briefs
- › Selection Guides
- › Application Brochures
- › Presentations
- › Press Releases, Ads

› [www.infineon.com/XMC](http://www.infineon.com/XMC)

## Technical Material



- › Application Notes
- › Technical Articles
- › Simulation Models
- › Datasheets, MCDS Files
- › PCB Design Data

› [www.infineon.com/XMC](http://www.infineon.com/XMC)

› [Kits and Boards](#)

› [DAVE™](#)

› [Software and Tool Ecosystem](#)

## Videos



- › Technical Videos
- › Product Information Videos

› [Infineon Media Center](#)

› [XMC Mediathek](#)

## Contact



- › Forums
- › Product Support

› [Infineon Forums](#)

› [Technical Assistance Center \(TAC\)](#)

# Disclaimer

The information given in this training materials is given as a hint for the implementation of the Infineon Technologies component only and shall not be regarded as any description or warranty of a certain functionality, condition or quality of the Infineon Technologies component.

Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this training material.





Part of your life. Part of tomorrow.

