AN14155

Using the i.MX RT1180 EtherCAT together with BECKOFF TwinCAT3 and SSC tool

Rev. 1.0 — 23 May 2024 **Application note**

Document information

Information	Content
Keywords	AN14155, i.MX RT1180 MCU, Evaluation Kit (EVK), MIMXRT1180-EVK, EtherCAT Slave Controller (ESC), TwinCAT3 software, Slave Stack Code (SSC) tool, Software Development Kit (SDK)
Abstract	This document explains the steps for using the EtherCAT peripheral on i.MX RT1180 platform based on MIMXRT1180-EVK board and the i.MX RT1180 SDK.

1 Introduction

The i.MX RT1180 platform integrates a two-port EtherCAT Slave Controller (ESC). This document explains the usage of EtherCAT peripheral on i.MX RT1180 platform based on i.MX RT1180 SDK and MIMXRT1180-EVK board. It also describes the usage of the EtherCAT Slave Stack Code (SSC) tool and TwinCAT3 software in EtherCAT project. The Slave Stack Code (SSC) Tool is a source code example that can be used as a development base for implementing EtherCAT in devices with their own processor.

2 Hardware platform

2.1 i.MX RT1180 crossover MCU

The i.MX RT1180 crossover MCU of NXP is a dual-core device that features an Arm Cortex-M7 (CM7) and a Cortex-M33 (CM33) core. The CM7 core can operate at a speed up to 800 MHz and the CM33 core up to 240 MHz with 1.5 MB on-chip RAM. The family supports multiple protocols, bridging communications between real-time Ethernet and Industry 4.0 systems and offers advanced security with the integrated EdgeLock Secure Enclave.

The i.MX RT1180 MCU includes an integrated Gbit/s time sensitive networking (TSN) switch and EtherCAT Slave Controller (ESC). This makes it well suited for industrial and automotive communication applications. The MCU also supports use of the MCUXpresso ecosystem, which includes an SDK, a choice of IDEs, and secure provisioning and configuration tools to enable rapid development.

2.2 i.MX RT1180 EtherCAT main features

Typically, industrial automation solutions implement the i.MX RT1180 platform, which includes the following main features:

- Transmission speed of 100 Mbit/s for integrated Ethernet transceivers
- 2 EtherCAT ports, 8 Fieldbus Memory Management Units (FMMUs), 8 Sync Managers, 128 bytes of user RAM and 8K bytes of process data RAM
- 64-bit distributed clocks
- Process data interface (PDI) is deactivated (no PDI)

2.3 MIMXRT1180-EVK board

The MIMXRT1180-EVK board is a hardware platform that enables design and evaluation of the most commonly used features of the i.MX RT1180 processor.

Table 1 shows the EtherCAT ports on the MIMXRT1180-EVK board:

Table 1. EtherCAT ports on MIMXRT1180-EVK

EtherCAT port	Connector
EtherCAT port 0	J28
EtherCAT port 1	J32

3 EtherCAT basic overview

EtherCAT is a high-performance real-time Ethernet communication protocol. it is often used for real-time control and communication in Industrial automation fields, like servo motor control. Beckhoff developed EtherCAT in 2003 and the International Electrotechnical Commission (IEC) standardized EtherCAT as the IEC 61158 standard.

EtherCAT is an Industrial Ethernet system and uses standard frames and the physical layer as defined in the Ethernet standard IEEE 802.3. EtherCAT also addresses the specific demands faced in the automation industry, listed below:

- There are hard real-time requirements with deterministic response times.
- The system usually comprises many nodes, each having only a small amount of cyclic process data.
- IT administrators do not commission and maintain fieldbus systems.

The design goal of EtherCAT is to achieve low communication latency and high-bandwidth data transmission, while meeting the needs of high-time precision control. It is implemented through a master-slave architecture. A master station is responsible for coordinating the entire network, while the slave station is responsible for providing input and output data.

EtherCAT is widely used in industrial robots, motion control, automatic assembly systems, and other industrial automation fields. The key benefits of EtherCAT include:

- High communication speed and bandwidth can reach to 100 Mbit/s.
- Sync performance can reach to nanosecond-level. If all slaves have the same time information, they can set their output signals simultaneously and affix their input signals with a highly precise timestamp.
- Support for line, tree, star, or daisy-chain topologies: EtherCAT supports almost all topologies. Pure bus or line topologies with many nodes are possible without limitations.

EtherCAT sends and receives control data from all nodes through a single frame. Figure 1 shows the propagation path between master and two port slaves. When the EtherCAT frame reaches the slave, the slave reads data from the EtherCAT frame and writes data to the EtherCAT frame. That frame passes through every slave and back to the master finally.

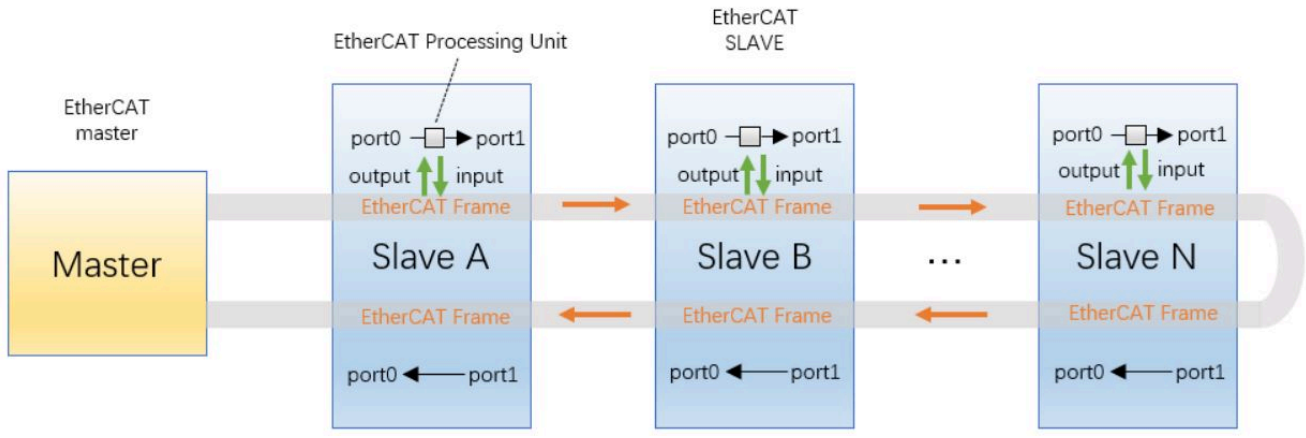


Figure 1. EtherCAT frame propagation path

3.1 Fieldbus Memory Management Units (FMMU)

Fieldbus Memory Management Units (FMMU) convert logical addresses into physical addresses by the means of internal address mapping. Figure 2 shows the way logical address map to physical address.

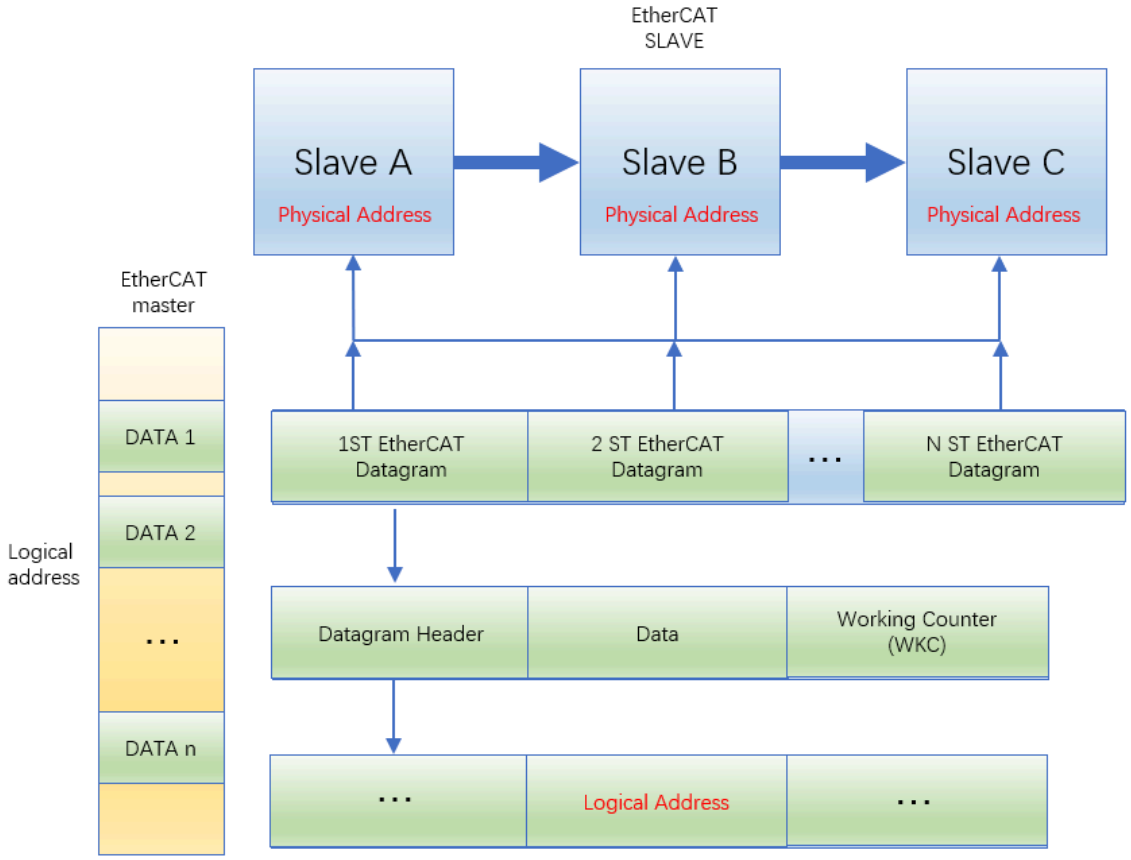


Figure 2. Logical address mapping to physical address

The i.MX RT1180 MCU supports eight FMMUs, which the EtherCAT master initializes. The FMMUs can get data from eight different logical address defined by the EtherCAT master. 'Logical address' refers to the address space in the master whereas 'physical address' refers to the address space in the slave.

3.2 SyncManager (SM)

The memory of an ESC can be used for exchanging data between the EtherCAT master and a local application (on a microcontroller attached to the PDI) without any restrictions. Such applications that use the memory for communication have some drawbacks. These drawbacks are addressed by the SyncManagers (SM) inside the ESCs. The drawbacks are listed below:

- Data consistency is not guaranteed. Semaphores must be implemented in software for exchanging data in a coordinated way.
- Data security is not guaranteed. Security mechanisms must be implemented in the software.
- Both EtherCAT master and the application must poll the memory to find out when the access of the other side has finished.

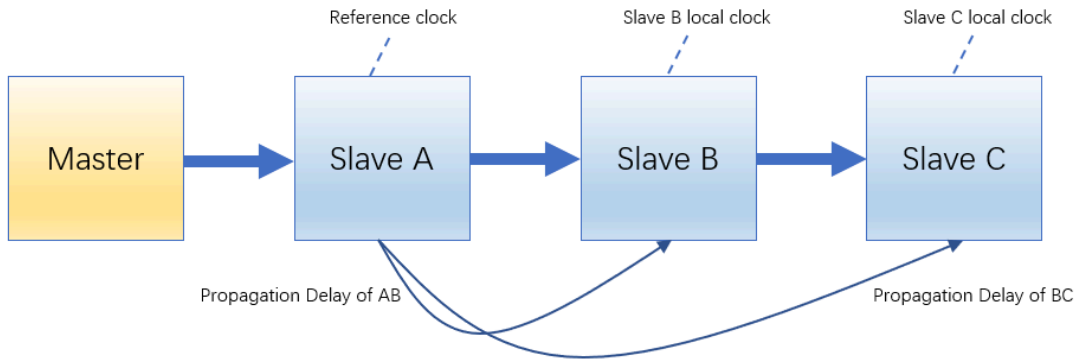
SyncManager supports both the Buffered mode and the Mailbox mode. The i.MX RT1180 supports eight SyncManagers. Typically SM0 and SM1 are used for mailbox, whereas SM2 and SM3 are used for process data output and input. The EtherCAT master also initializes the SyncManager. The EtherCAT master reads SM Information from EEPROM by SII interface. Therefore, SM Initialization information is defined in an XML file created by the SSC tool.

3.3 EtherCAT Distribute clock

EtherCAT Distribute clock enables all the EtherCAT devices to share the same system time. The EtherCAT slave devices can be synchronized to each other, and therefore, the local applications are synchronized as well. For system synchronization, all slaves are synchronized to one Reference Clock. Typically, the first ESC with Distributed Clocks capability after the master within one segment holds the reference time (System Time).

Slave devices output the sync signal according to the system time. That signal can trigger interrupts. Slaves that support the distributed clock are called DC slaves. Figure 3 shows distribute clock and propagation delay.

Figure 3. Distribute clock and propagation delay



For clock synchronization, i.MX RT1180 takes into account the propagation delays, local clock sources drift, and local clock offsets, which are described below:

- Propagation Delay: Propagation Delay from one slave to the other slaves.
- Offset: This time interval is the offset between the local clock and reference clock. It includes the propagation delay from the ESC holding the Reference clock to the device with the slave clock. The initial difference of the local times results due to the ESCs being powered up at different times.
- Drift: This time interval is the time shift introduced due to the Reference Clock and DC slaves being sourced by different clock sources. Their clock sources are subject to small deviations of the clock periods.

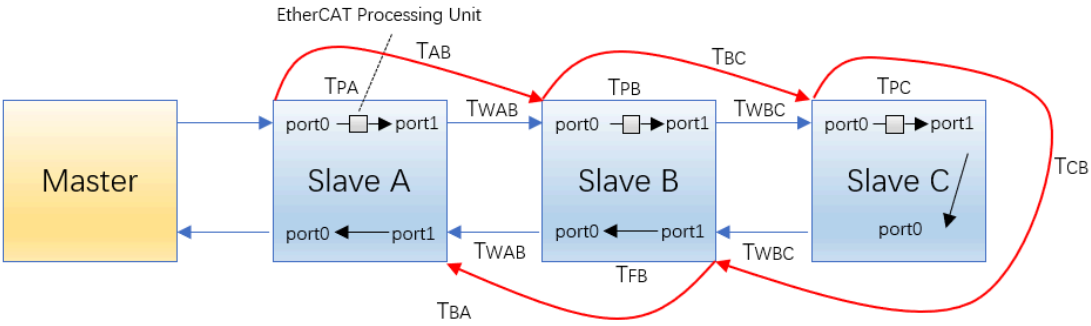


Figure 4. Propagation delay of each slave

Propagation delay between each slave is shown in Figure 4.

- T_{px} indicates the Processing delay of slave x (through EtherCAT Processing Unit, x=A-C).
- T_{wxy} indicates the Wire propagation delay between slaves x and y (assumed to be symmetrical in both directions, x/y=A-C).
- Tx0, Tx1 indicate the Receive Time Port 0/1 values of slave x (time when first preamble bit is detected, x=A-C), measured with a write access to DC Receive Time 0 register.

Propagation delay is calculated between Slave B and C is as follows:

1. $T_{BC}=T_{PB}+T_{WBC}$;
2. $T_{CB}=T_{PC}+T_{WBC}$;
3. $T_{B1}=T_{B0}+T_{BC}+T_{CB}$;

Assuming that the processing delays of slave B and C are identical, the below equation holds true:

$$\mathbf{T_{BC}=T_{CB}=(T_{B1}-T_{B0})/2;}$$

Propagation delay is calculated between Slave B and A,

- T_{FX} indicates the forwarding delay of slave x (alongside the EtherCAT Processing Unit, x=A-C)
- T_{Diff} indicates the difference between the processing delay and the forwarding delay. $T_{Diff}=T_P-T_F$. Then, the following expressions are valid:

1. $T_{AB}=T_{PA}+T_{WAB}$;
2. $T_{BA}=T_{FB}+T_{WAB}$;
3. $T_{BA}= T_{AB}-T_{Diff}$;
4. $T_{A1}=T_{A0}+T_{AB}+T_{BC}+T_{CB}+T_{BA}$;
5. $T_{AB}+T_{BA}=(T_{A1}-T_{A0})-(T_{B1}-T_{B0})$;

So the propagation delay between slave A and B is:

$$\mathbf{T_{AB}=((T_{A1}-T_{A0})-(T_{B1}-T_{B0})+T_{Diff})/2;}$$

$$\mathbf{T_{BA}=((T_{A1}-T_{A0})-(T_{B1}-T_{B0})-T_{Diff})/2;}$$

i.MX RT1180 also takes drifts and offset into consideration. For more details, refer to the i.MXRT1180 Reference Manual.

3.4 EtherCAT operational modes

EtherCAT slaves have the following sync operation modes:

- Free Run mode
- Sync Manager (SM) mode
- Distributed Clock (DC) mode (normal and enhanced)

A description of these modes is provided below.

1. **Free Run mode**
   In Free Run mode, the local control cycle is created by local timer interrupt, Figure 5 shows the Free Run mode timing.

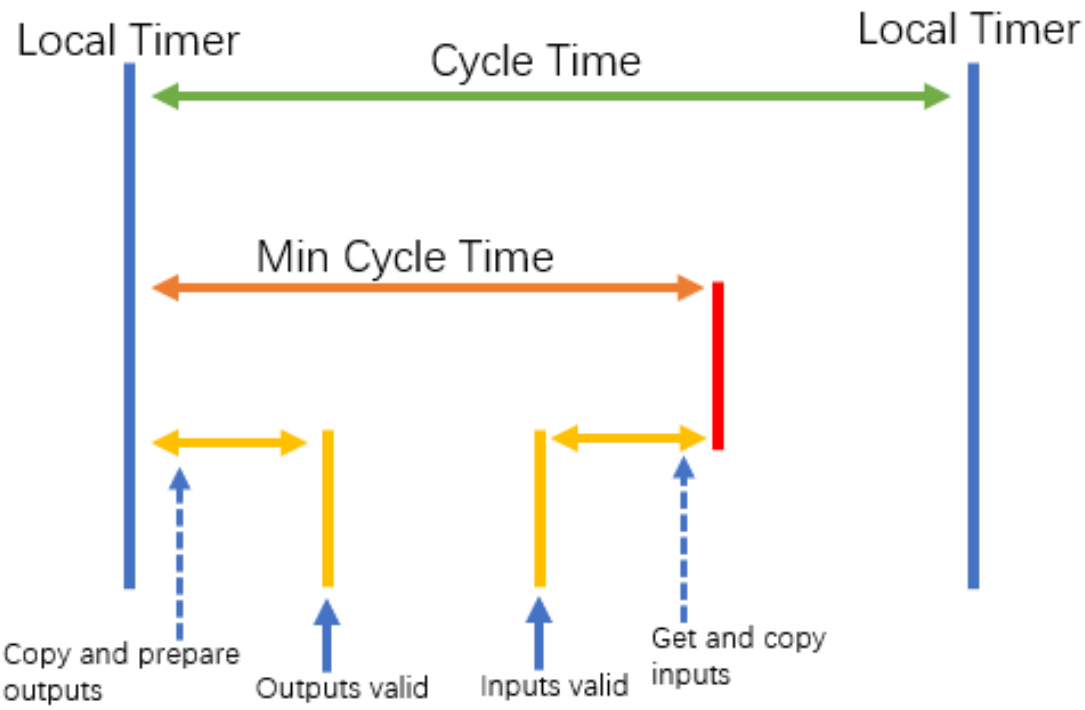


Figure 5. Free Run mode

2. **SM mode**
   In SM mode, the local control cycle is initiated when data input or output event occurs. Figure 6 shows the SM mode timing.

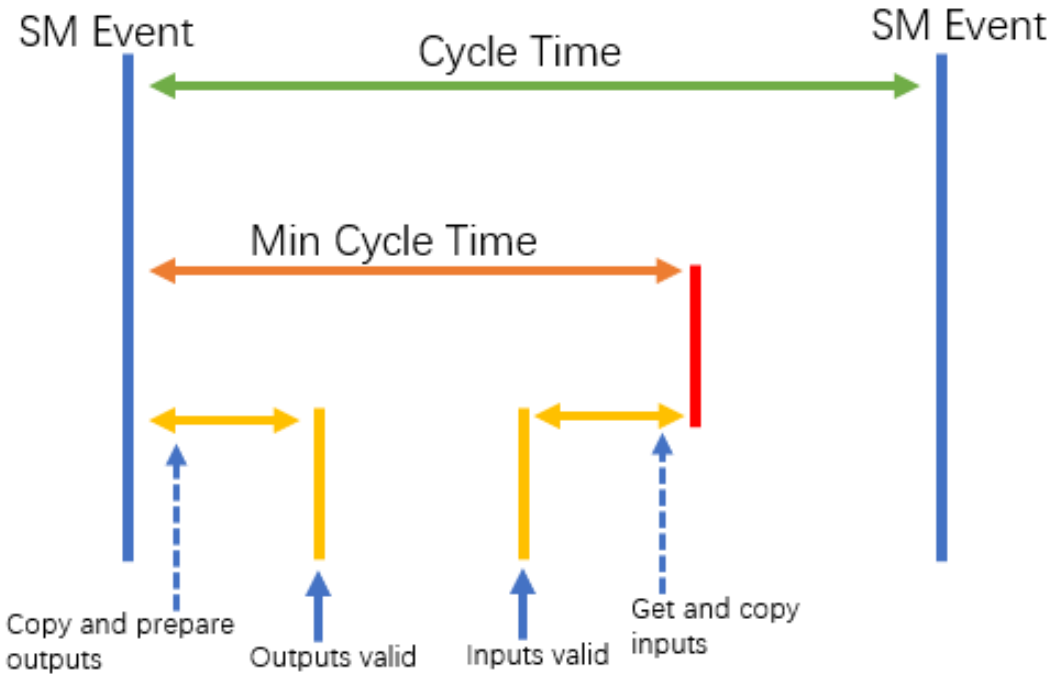


Figure 6. SM mode

3. **DC mode**
   a. **Normal DC mode**
   In normal DC mode, the local control cycle is created by the Sync signal. In this mode, the EtherCAT master must send frame before the Sync event happens. Figure 7 shows the Normal DC mode timing.

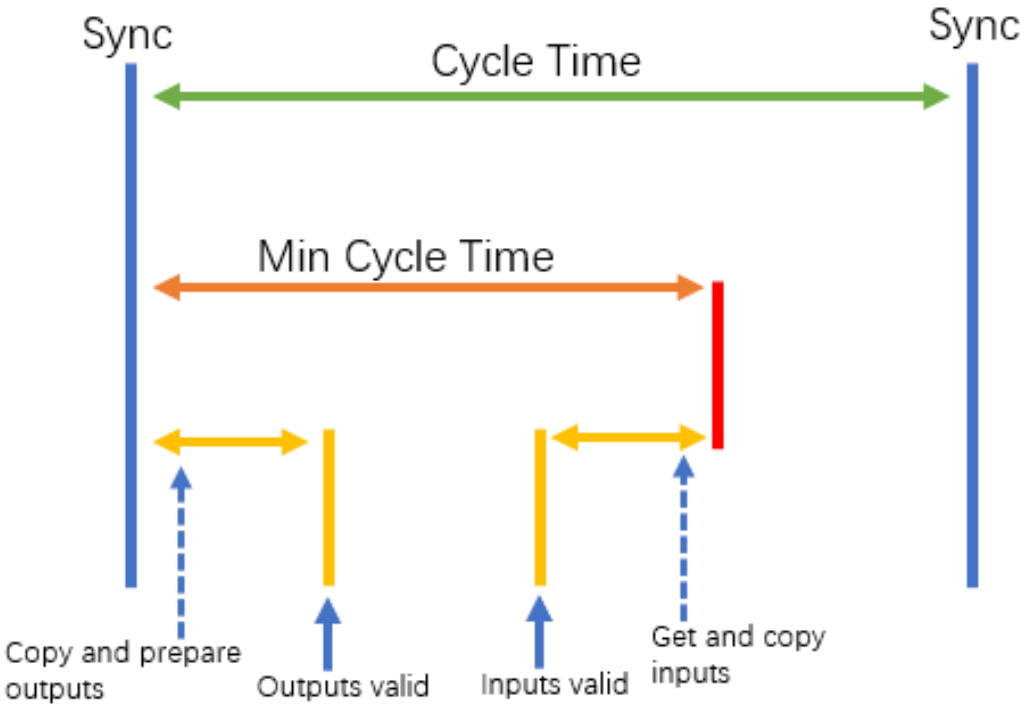


Figure 7. Normal DC mode

b. **Enhanced DC mode**
Enhanced DC mode aims to achieve greater sync performance. In this mode, the EtherCAT slave must copy and prepare outputs before the Sync event. When a sync signal arrives, local operation continues and has a greater performance. Figure 8 shows the Enhanced DC mode timing.

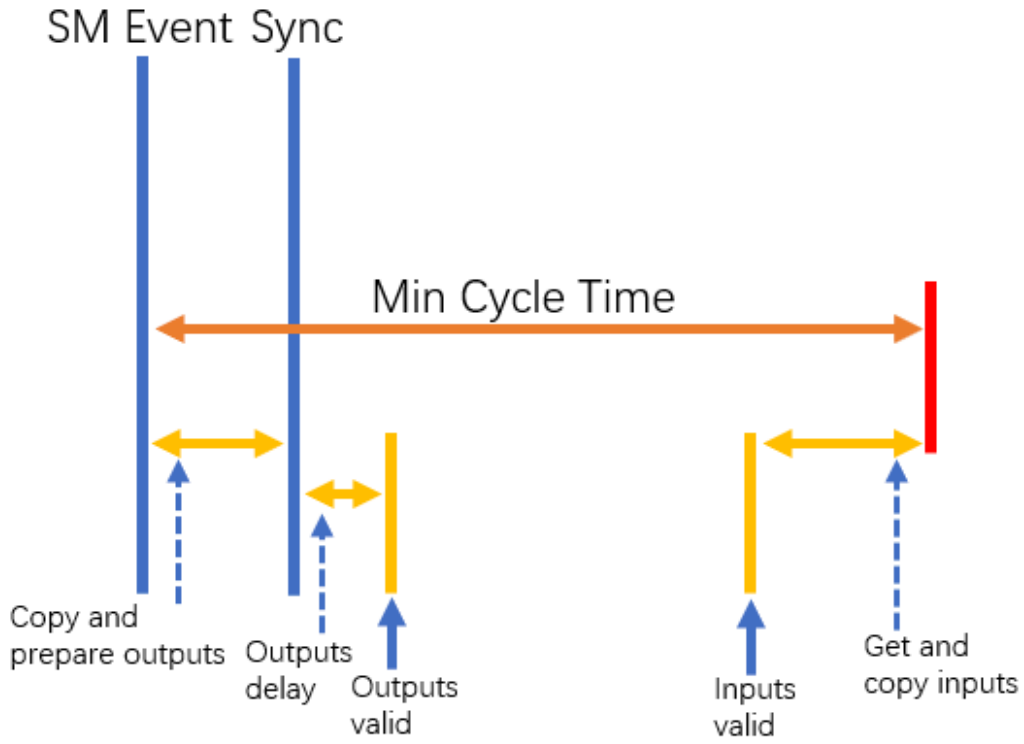


Figure 8. Enhanced DC mode

4 Integrating EtherCAT Slave Stack code to the SDK demo

Follow the steps below to integrate the EtherCAT Slave Stack code to the SDK demo:

1. Download and install the **Beckhoff SSC** tool. To download the SSC tool, you must be a member of the EtherCAT Technology Group (ETG). The SSC tool download link is below:
   http://www.ethercat.org/login.aspx?ReturnUrl=%2fmemberarea%2fstack_code.aspx
   Figure 9 shows the SSC tool login page.

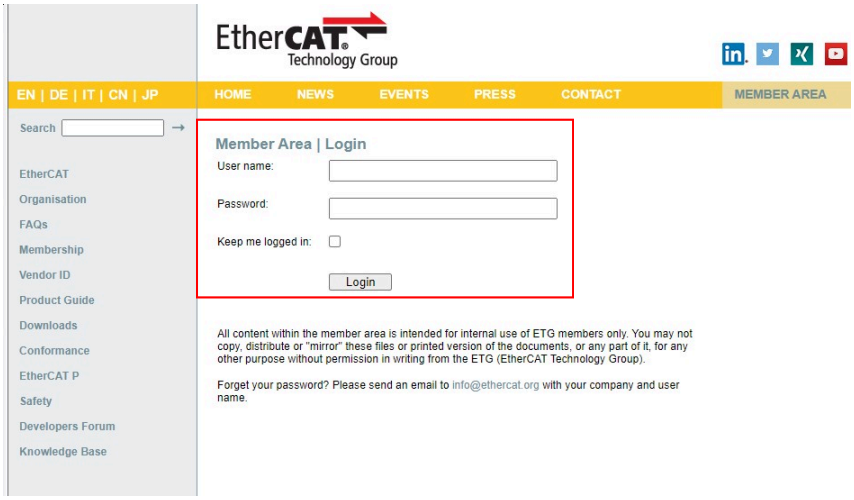


Figure 9. SSC tool download page

2. Start the SSC tool as shown in Figure 10.

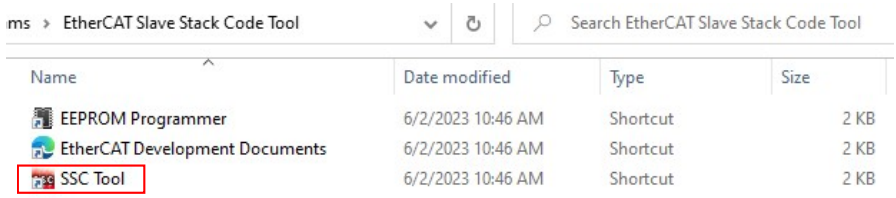


Figure 10. Start the SSC tool

3. Create a new SSC project by using **Select File > New** as shown in Figure 11.

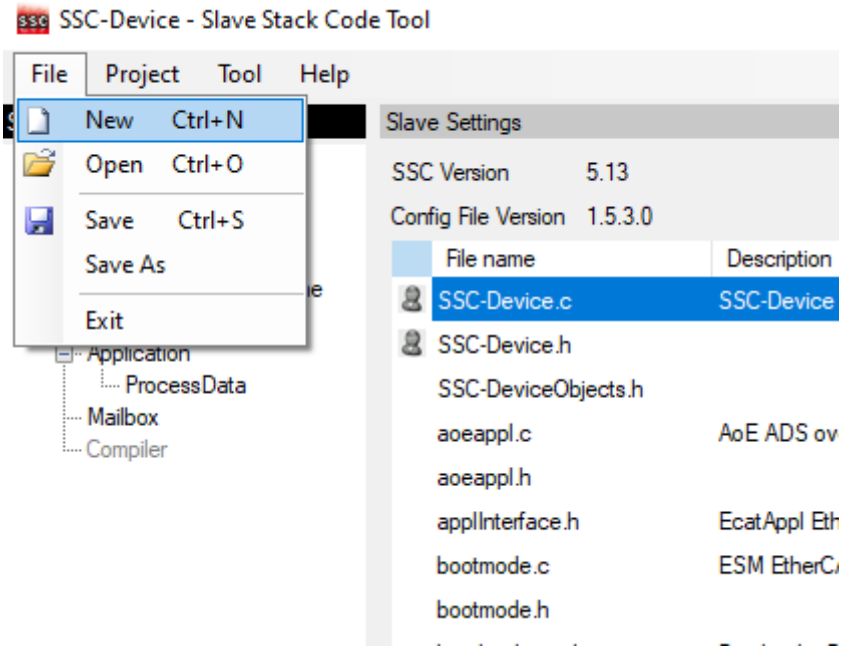


Figure 11. Creating a new SSC project

4. NXP Digital IO project can be added by using the SCC configuration file, located in the MIMXRT1180-EVK SDK, as shown in Figure 12. The SDK version used in this application note is 2.14.1.
   The SCC configuration file path is below:

```
SDK_2_14_1_MIMXRT1180-EVK\boards\evkmimxrt1180\ecat_examples\digital_io
\cm33\SSC\digital_io.xml
```

or

```
SDK_2_14_1_MIMXRT1180-EVK\boards\evkmimxrt1180\ecat_examples\digital_io
\cm7\SSC\digital_io.xml
```

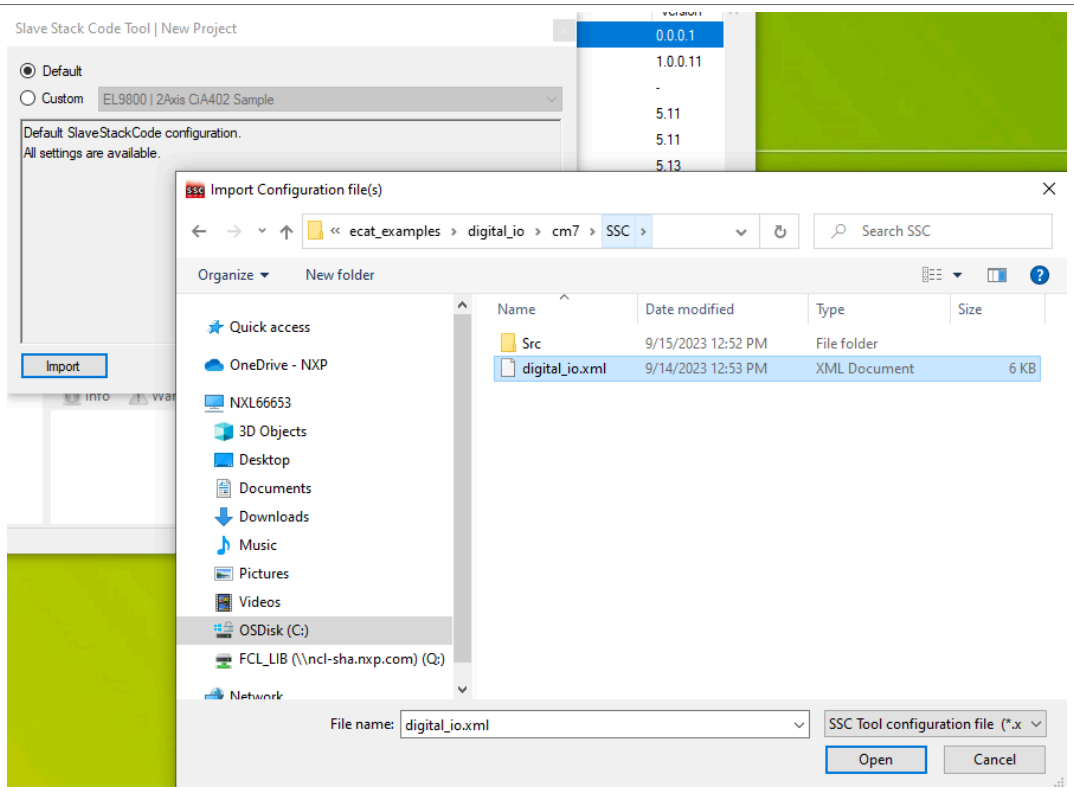


Figure 12. Importing SSC configuration file

5. On this stage, SSC Tool (version 5.13) successfully adds the **NXP ECAT Digital IO modular** project. This project can be located in the drop-down menu as shown in Figure 13.

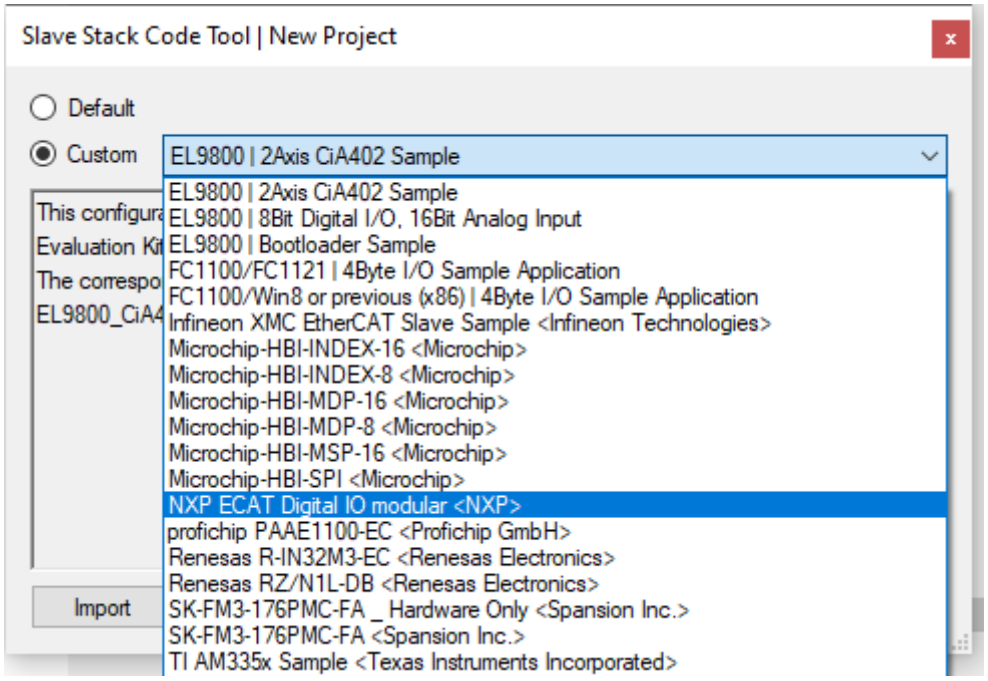


Figure 13. Locate the NXP project in the drop-down menu

6. Clicking **OK** opens the project-related window as shown in Figure 14.

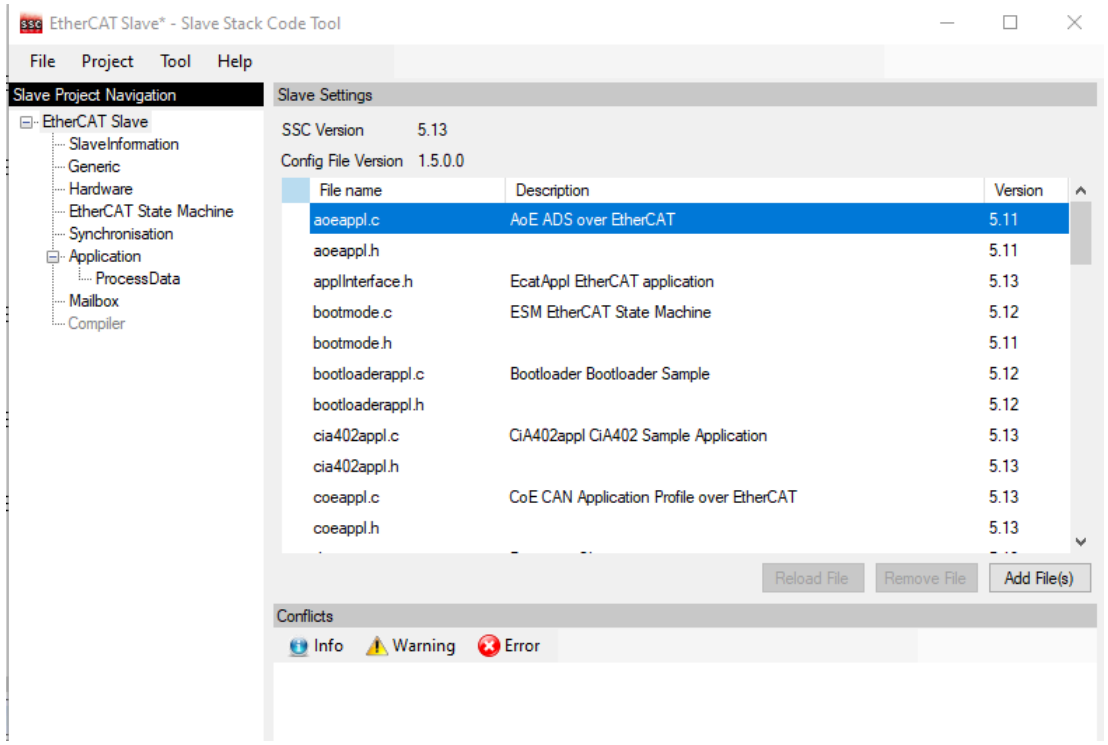


Figure 14. Project-related window

7. To create or import application files, click **Tool > Application**. Select **Import** or **Create New**, as shown in Figure 15. The Application file can also be found in the below Application file path in MIMXRT1180-EVK SDK:

```
SDK_2_14_1_MIMXRT1180EVK\boards\evkmimxrt1180\ecat_examples\digital_io
\cm33\SSC\digital_io.xlsx
```

or

```
SDK_2_14_1_MIMXRT1180EVK\boards\evkmimxrt1180\ecat_examples\digital_io
\cm7\SSC\digital_io.xlsx
```

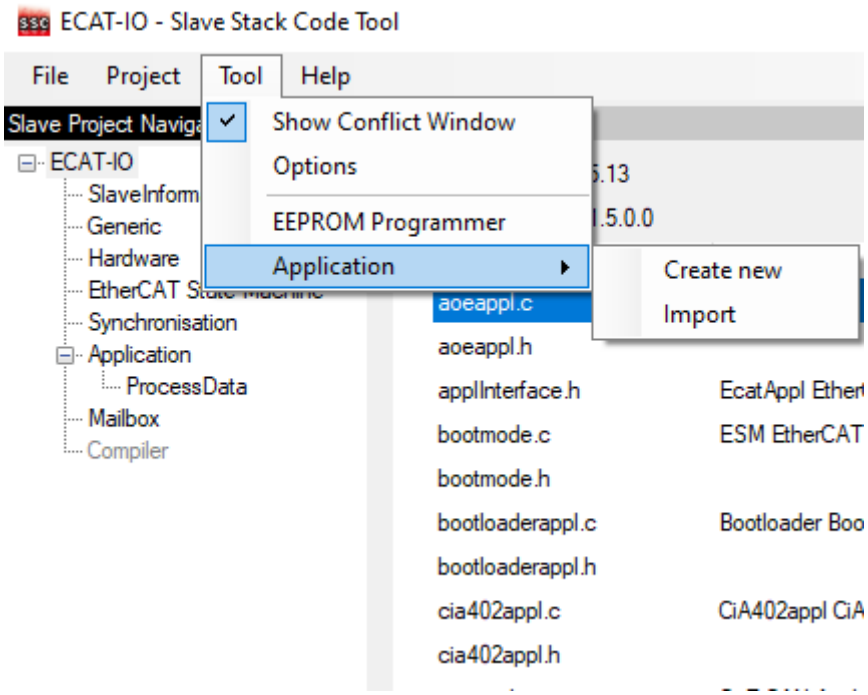


Figure 15. Importing or creating a new Application

8. The Application file can also be configured through Excel, as shown in Figure 16. SSC Tool generates an XML file automatically based on this Excel file.

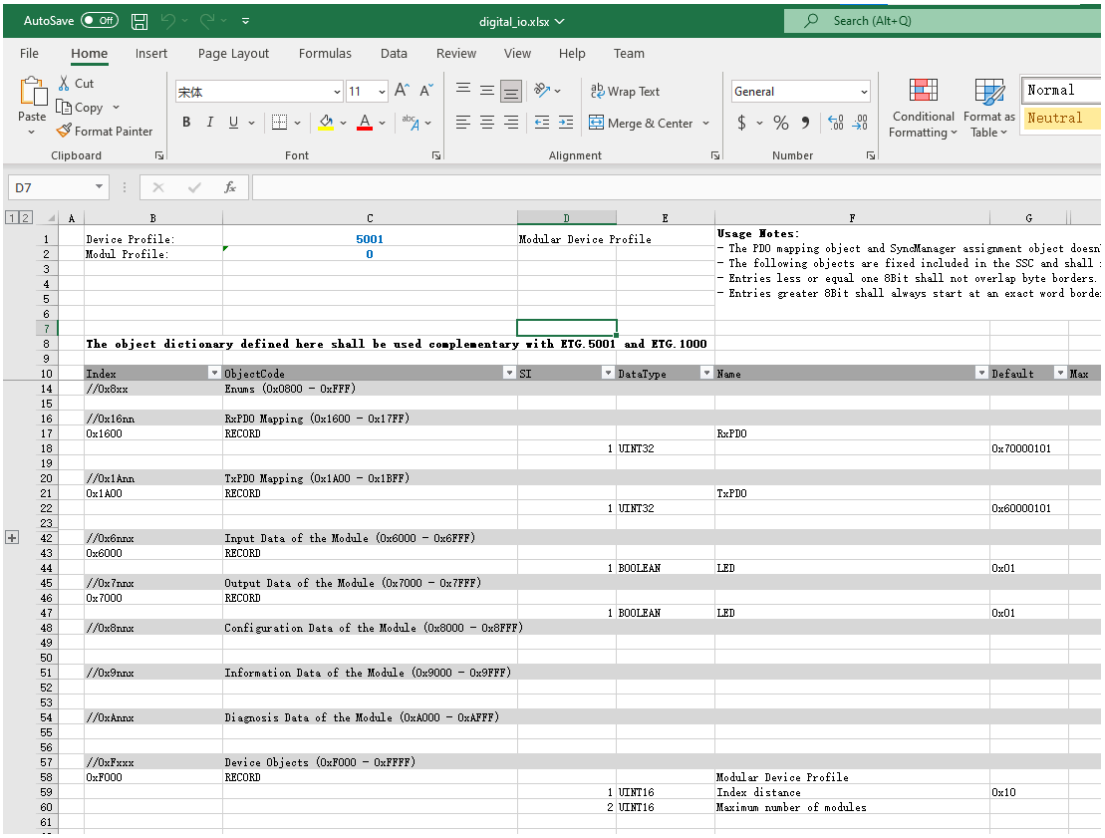


Figure 16. Configuring the Application through Excel

9. After creating or importing an Application file, create new slave files, as shown in Figure 17. Click **Project > Create new Slave Files**.

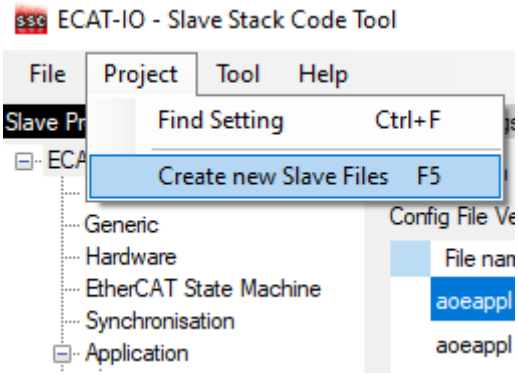


Figure 17. Creating new Slave files

10. Click **Start.** A new project file is created, as shown in Figure 18. XML file and src folder can be found in the related project folder.

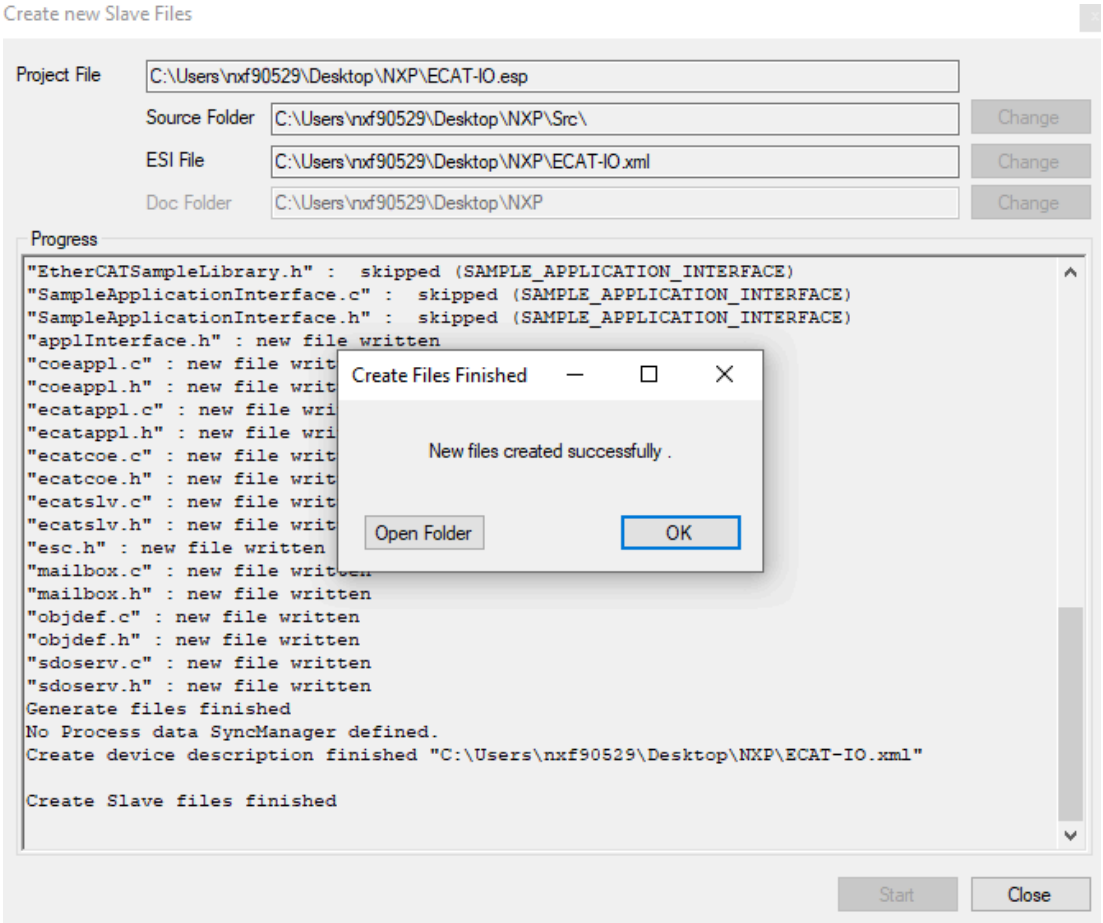


Figure 18. Start creating new Slave files

11. Copy and replace the project file from the `src` folder created by the SSC tool to the `src` folder in the MIMXRT1180-EVK SDK, as shown in Figure 19.

`src` folder path:

```
C:\lxtdoc\2_SDK\SDK_2_14_1_MIMXRT1180-EVK\boards\evkmimxrt1180\ecat_examples\
 digital_io\cm33\SSC\Src
```

or

```
C:\lxtdoc\2_SDK\SDK_2_14_1_MIMXRT1180-EVK\boards\evkmimxrt1180\ecat_examples\
 digital_io\cm7\SSC\Src
```

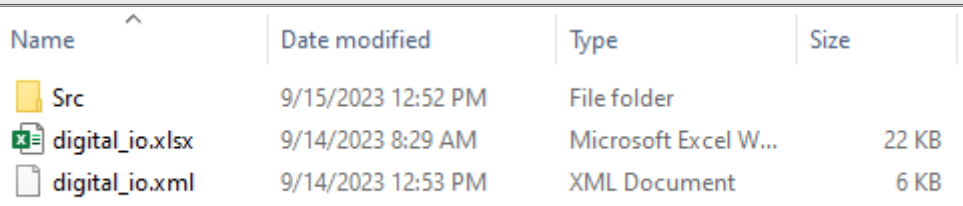


Figure 19. MIMXRT1180-EVK SDK demo folder

12. Build and compile the SDK project and download it to the MIMXRT1180-EVK board, as shown in Figure 20 . Connect MIMXRT1180-EVK EtherCAT port0 with EtherCAT master.

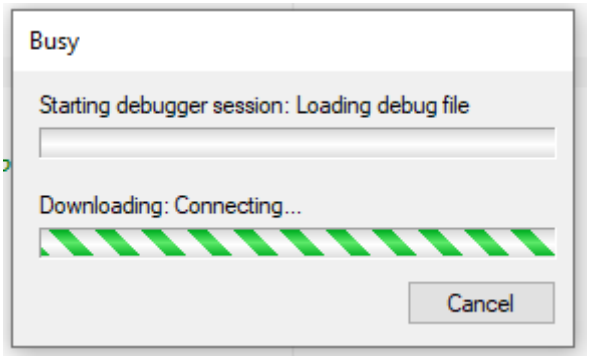


Figure 20. Downloading code to the MIMXRT1180-EVK board

13. SM and FMMU configuration can be found in XML file created by SSC tool.

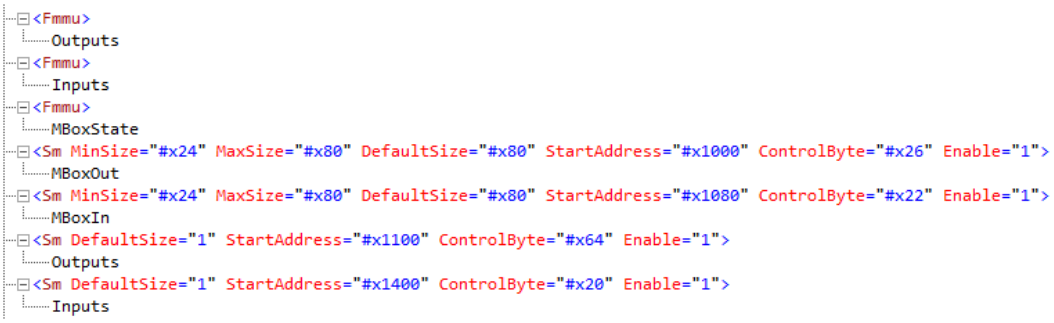


Figure 21. SM and FMMU configuration in XML file

14. Object Dictionary mapping is shown in Figure 22. Object 0x1C13 decides that object 0x1A0n would be used and object 0x1A0n decides that object 0x600n would be used.

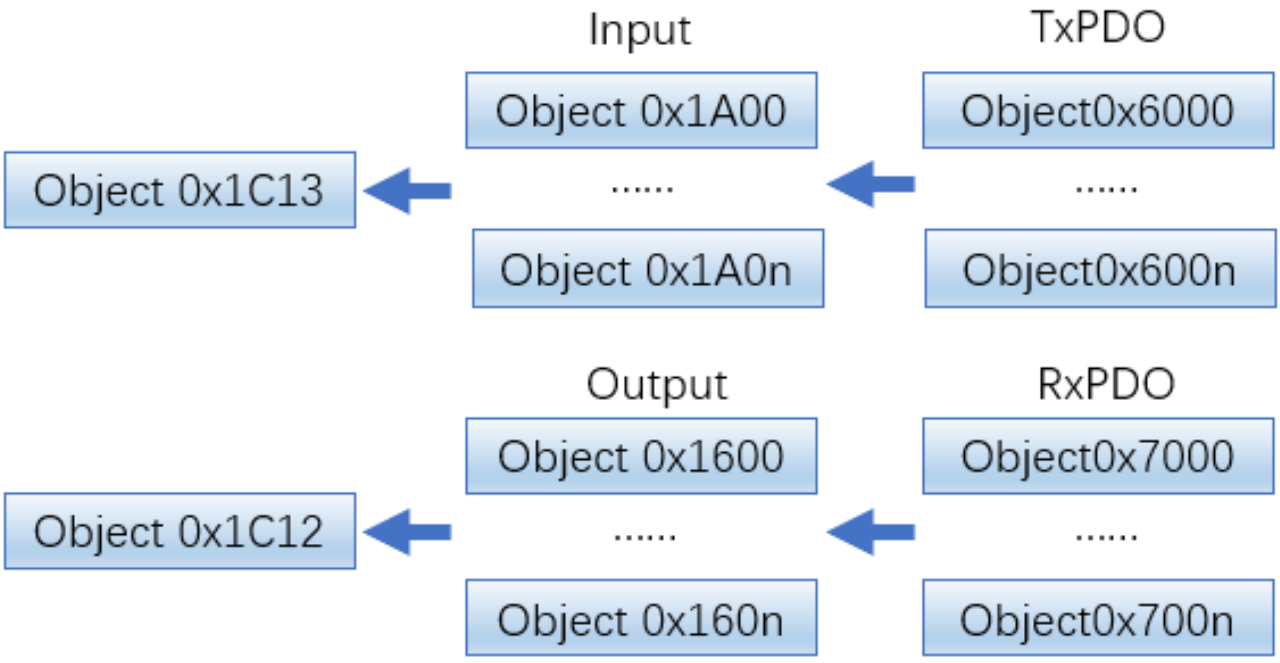


Figure 22. Object Dictionary mapping

5 Setting TwinCAT3 in Config mode

This section describes the process for configuring the TwinCAT3 software in Config mode.

5.1 Installing TwinCAT Master driver and scanning EtherCAT devices

Follow the steps below to install the TwinCAT Master driver and scan EtherCAT device:

1. Open TwinCAT3 and create a new TwinCAT project, as shown in Figure 23.

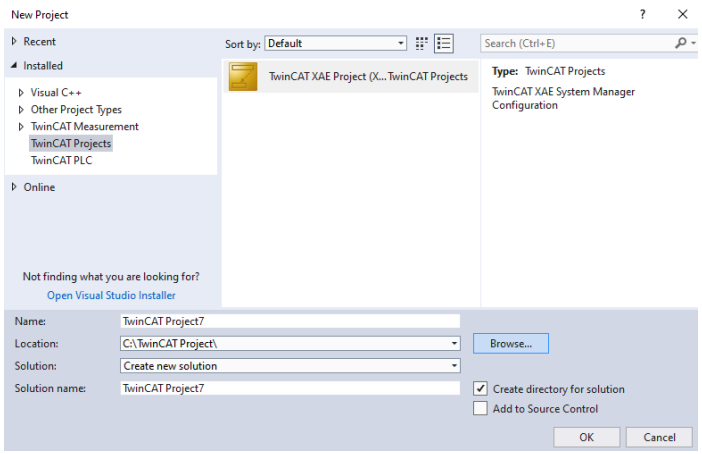


Figure 23. Creating a new TwinCAT project

2. Set TwinCAT3 to restart in Config mode, as shown in Figure 24.

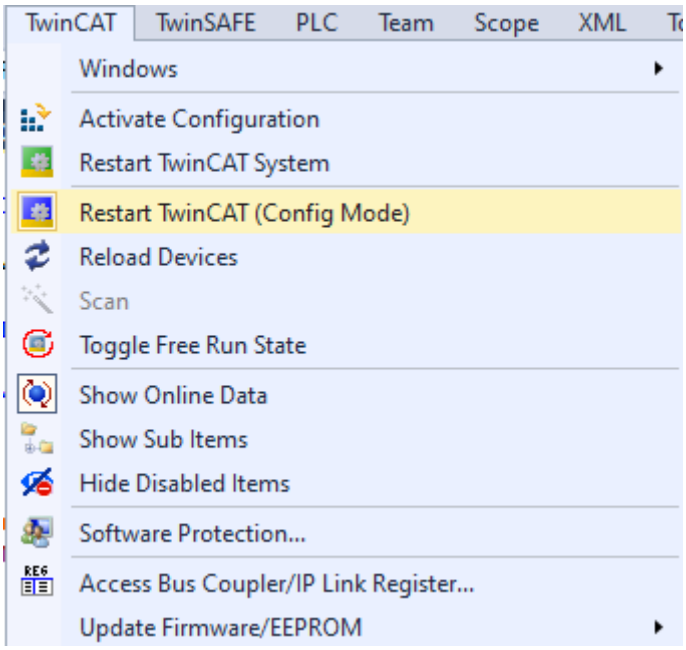


Figure 24. Set TwinCAT3 to Config mode

3. Keep TwinCAT in Config mode and select "**Show Realtime Ethernet Compatible Devices...**", as shown in Figure 25.

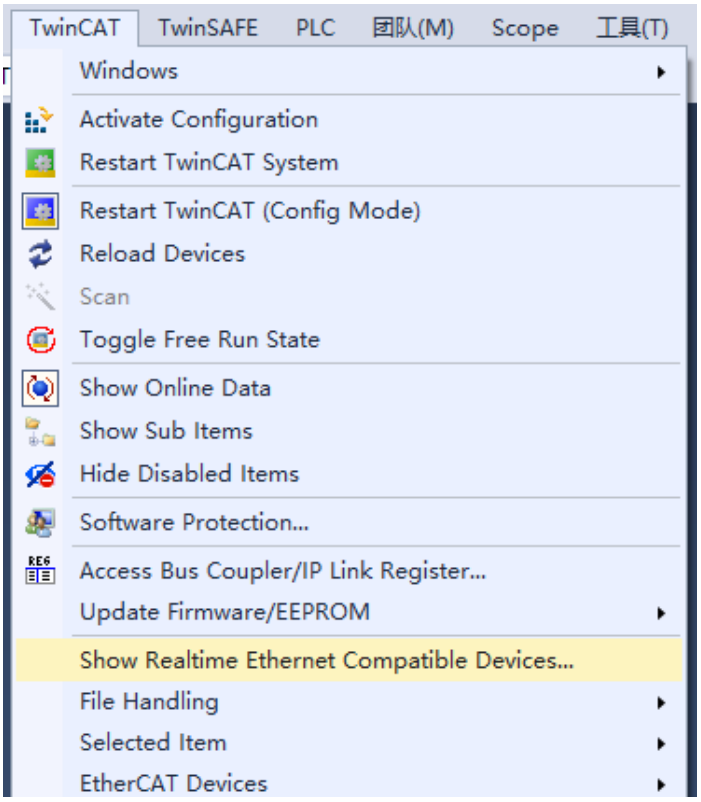


Figure 25. Select "Show Real-time Ethernet Compatible Devices"

4. Make sure that the EtherCAT master device driver is installed on your system. This demo uses the PC as master and Intel (13)1219-LM as the PC network controller as shown in Figure 26. If the PC is used as

TwinCAT master, the PC network controller must support the EtherCAT master. For the list of TwinCAT3 supported network controllers , refer to the URL: https://infosys.beckhoff.com/english.php?content=../content/1033/tcsystemmanager/9810943371.html&id=8751857768543711394

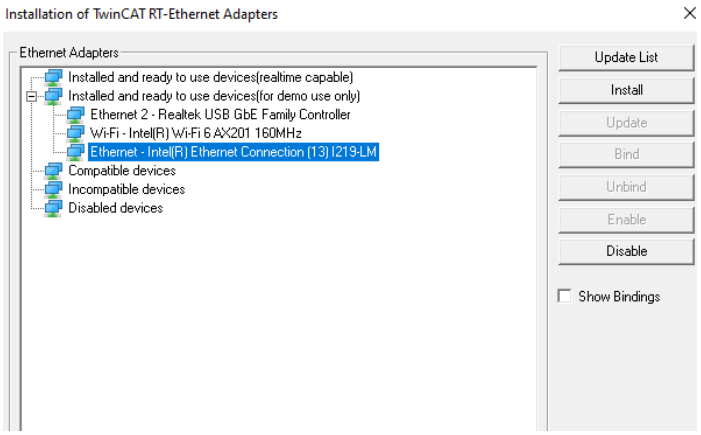


Figure 26. Installation of TwinCAT RT-Ethernet Adapters

5. Place the XML file created by SSC tool in the TwinCAT3 config folder, as shown in Figure 27. The TwinCAT3 config folder path is below:

```
TwinCAT\3.1\Config\Io\EtherCAT
```

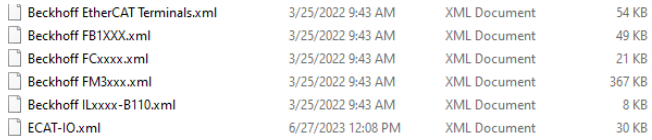


Figure 27. TwinCAT3 config folder

6. Click **Reload Device Descriptions** to update the XML file, as shown in Figure 28.

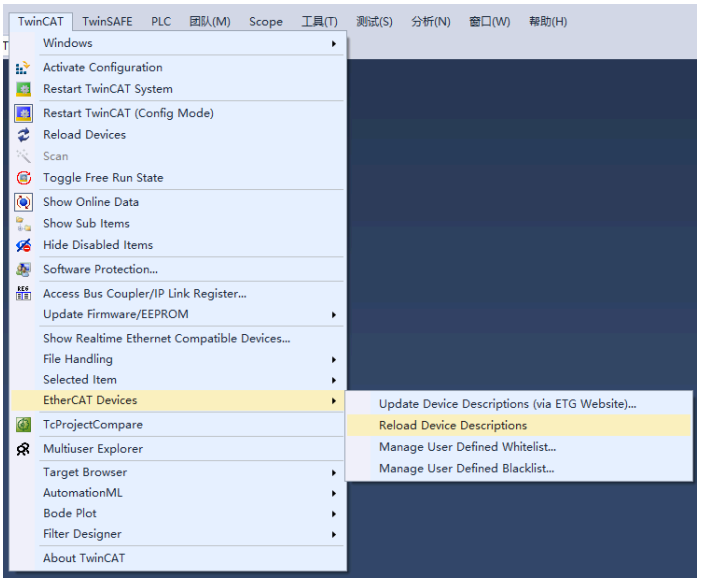


Figure 28. Reloading Device Descriptions

7. Right click **Devices** and select **Scan**, as shown in Figure 29.

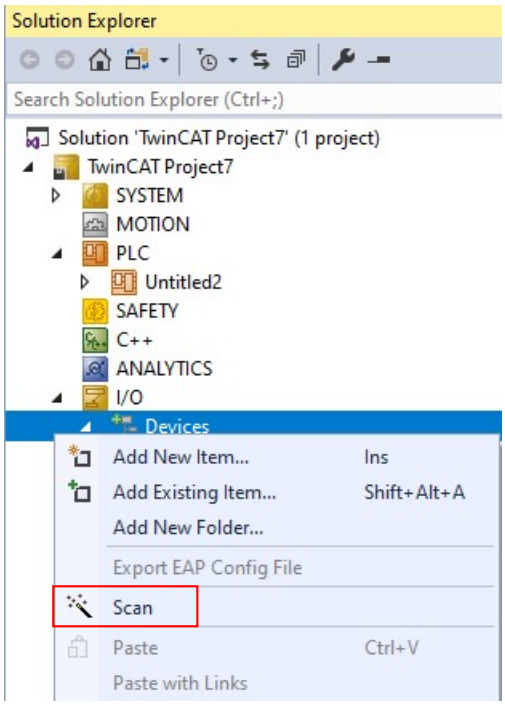


Figure 29. Scan Master Device

8. The EtherCAT master device (Device 3) is detected, as shown in Figure 30.

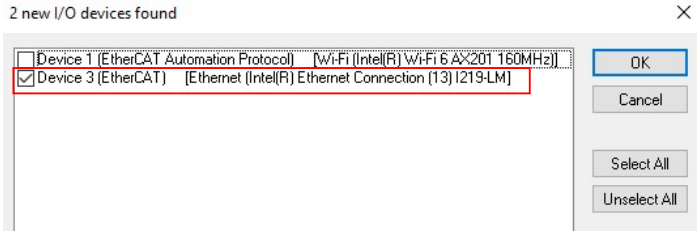


Figure 30. Master Device detected

Right click **Device 3** and click **Scan** to find the associated EtherCAT slave device, as shown in Figure 31.

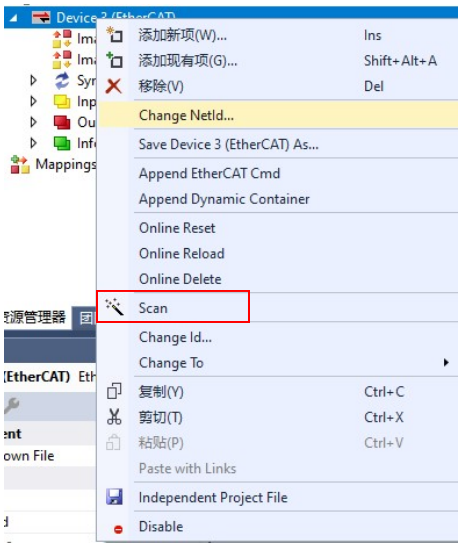


Figure 31. Scan Slave Device

9. At this stage, the Box is detected. If it is the first-time scan of the EtherCAT slave device, the box is displayed as shown in Figure 32.

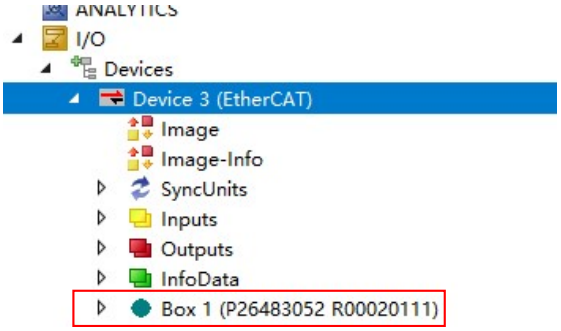


Figure 32. EtherCAT slave device detected

5.2 Updating EEPROM data and writing a value online in Config mode

This section describes the steps to update EEPROM data and writing the write value online in Config mode.

1. Click **Box 1** and select **Advanced Settings** in EtherCAT pane, as shown in Figure 33.

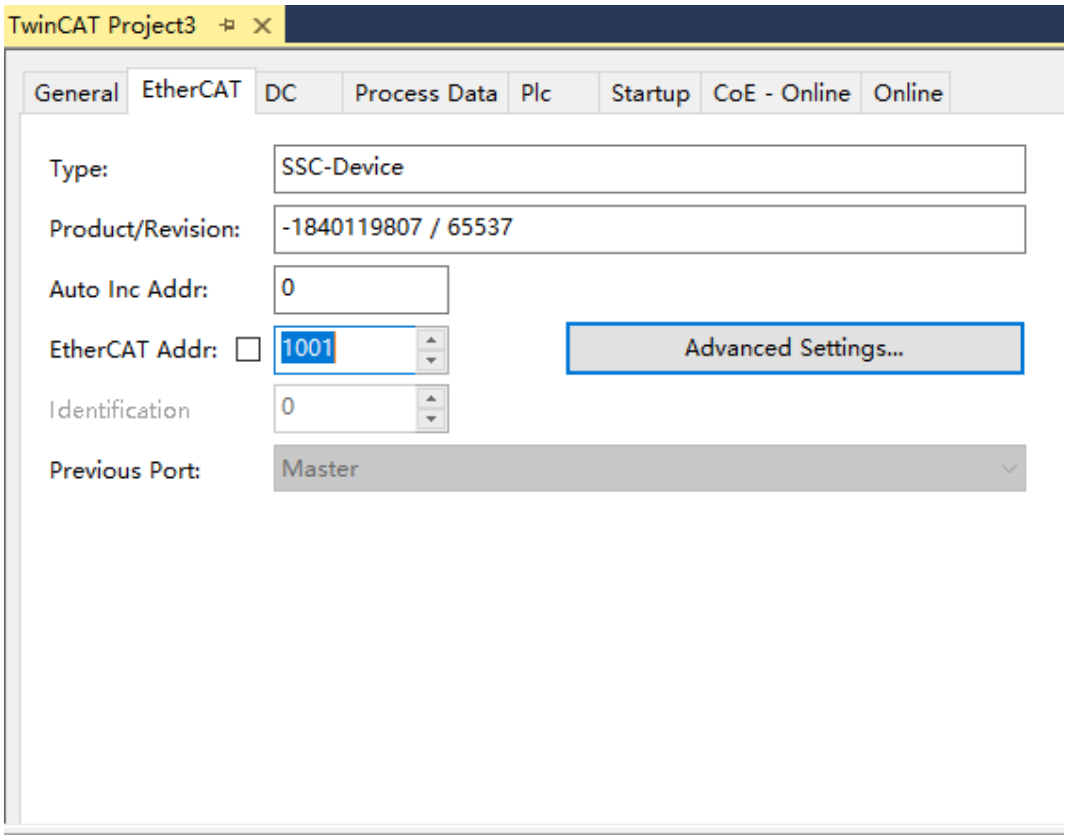


Figure 33. EtherCAT slave device detected

2. Click **Write EEPROM** and choose XML file created by SSC tool. On clicking **OK**, the EEPROM is written as per the selected configuration as shown in Figure 34.

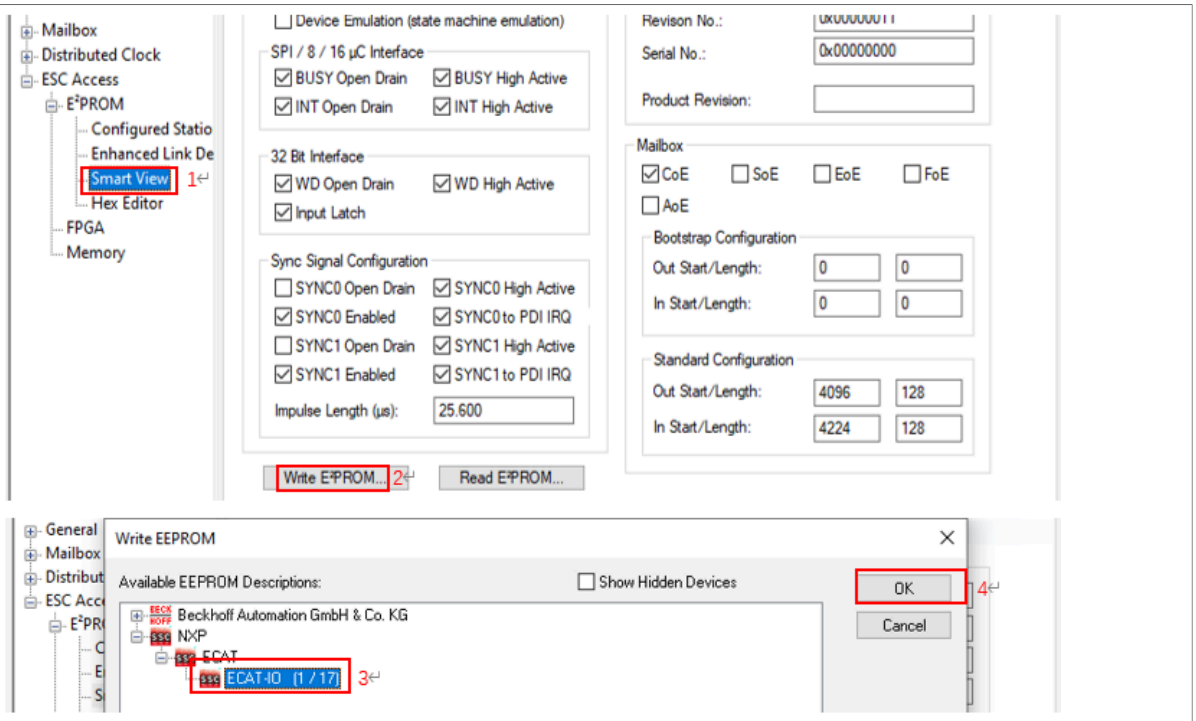


Figure 34. Smart View Editor displaying the options available to write EEPROM

3. Delete the old box and scan the box again. The new box with EEPROM burned can be found as shown in Figure 35.

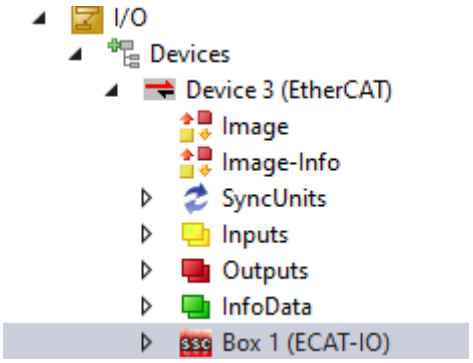


Figure 35. New box can be found

4. Activate Free Run, as shown in Figure 36.

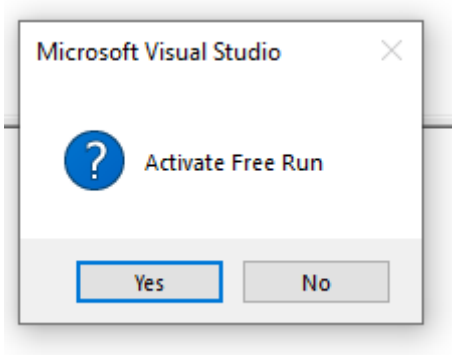


Figure 36. Active Free Run

5. Make sure EtherCAT slave is in Operation (OP) mode, as shown in Figure 37.

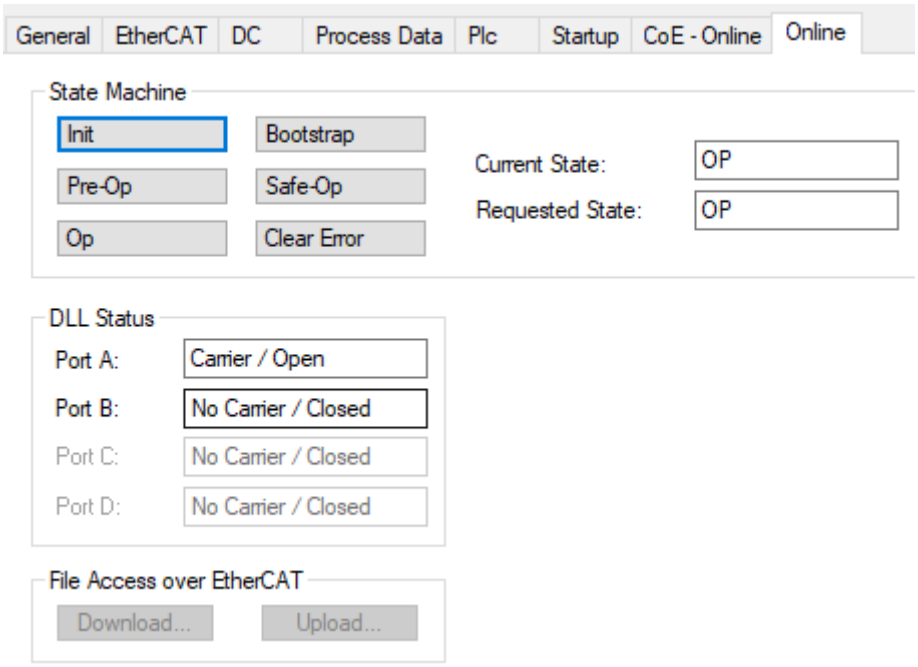


Figure 37. EtherCAT slave in OP mode

6. In TwinCAT Config mode, you can pass an 'Online Write value' (**Online Write '0'** or **Online Write '1'**) to the EtherCAT slave. For example, if value ‘1’ is written to the RxPDO variable LED, the user LED on the MIMXRT1180-EVK board lights up.

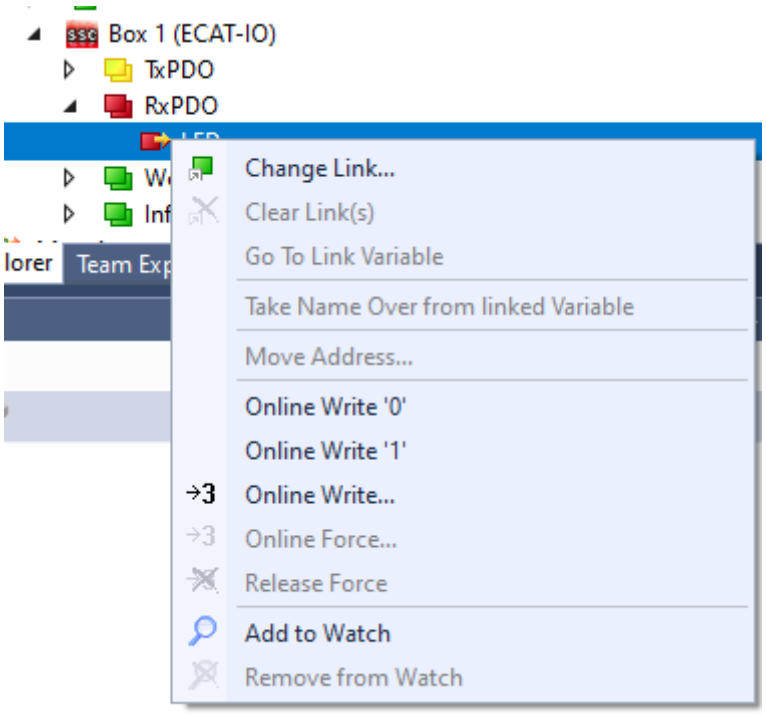


Figure 38. Online write in Config mode

6 Using TWINCAT3 in Run mode

This section describes the process for configuring the TwinCAT3 software in Run mode.

1. First, create a new PLC project. Right click **PLC** and then click **Add New Item**, as shown in Figure 39.

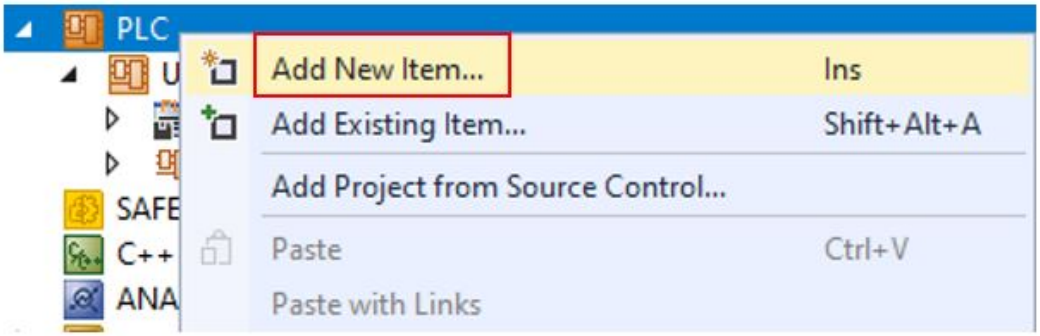


Figure 39. Creating a new PLC project in TwinCAT3

2. Add a **Standard PLC Project**, as shown in Figure 40.

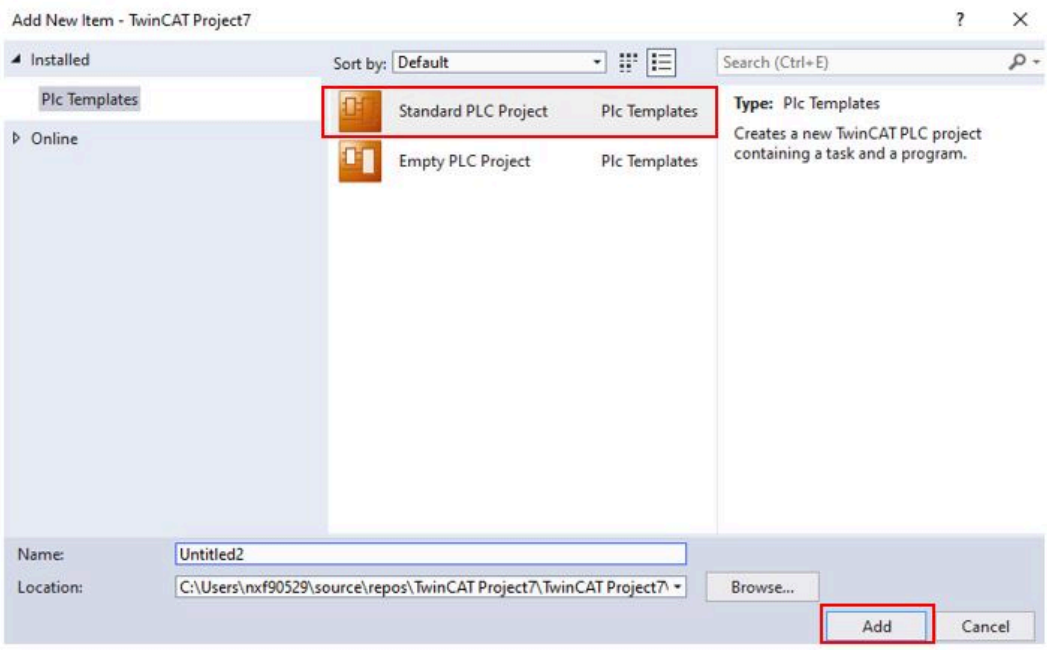


Figure 40. Creating a Standard PLC Project in TwinCAT3

3. Write PLC related codes as shown in Figure 41.

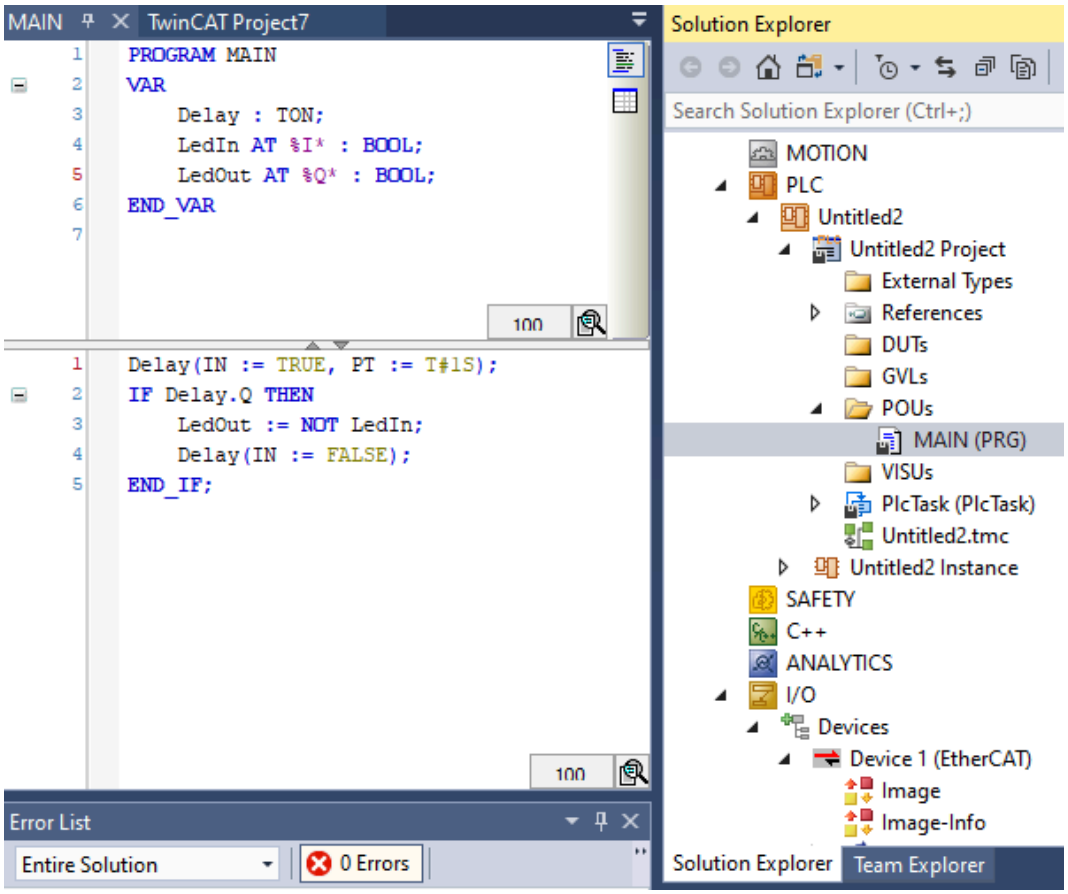


Figure 41. Writing PLC project codes

4. Link the ESC input and output variables to the PLC project variables. For this, right click '**LED**' and choose **Change Link**, as shown in Figure 42.

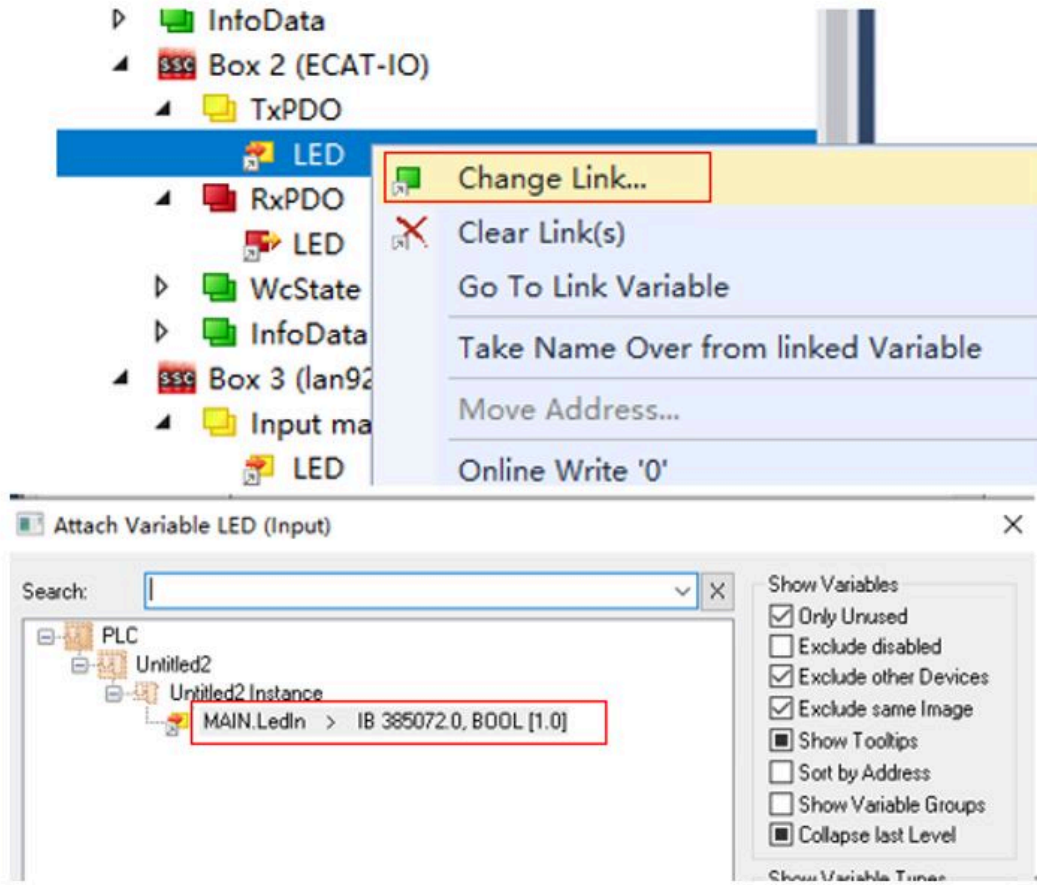


Figure 42. Linking ESC variables to PLC variables

5. Switch to Run mode as shown in Figure 43 and ensure that the EtherCAT slave runs in OP mode.

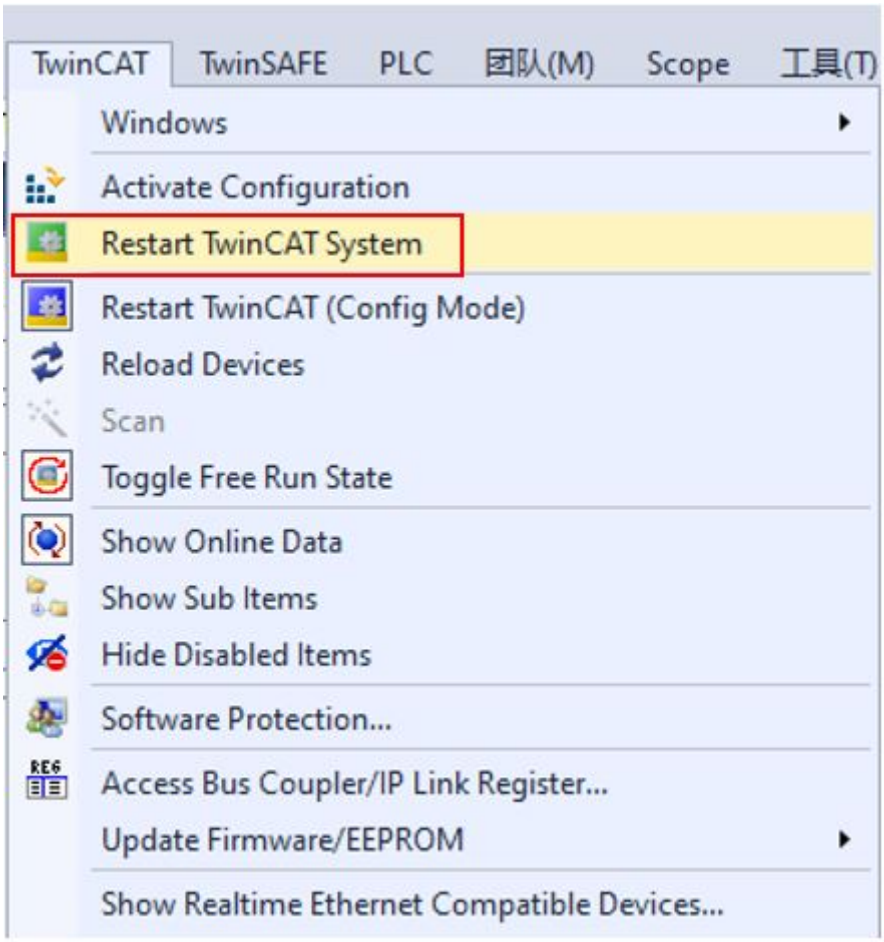


Figure 43. Switching TwinCAT to Run mode

6. Then, click **Activate Configuration**, as shown in Figure 44.

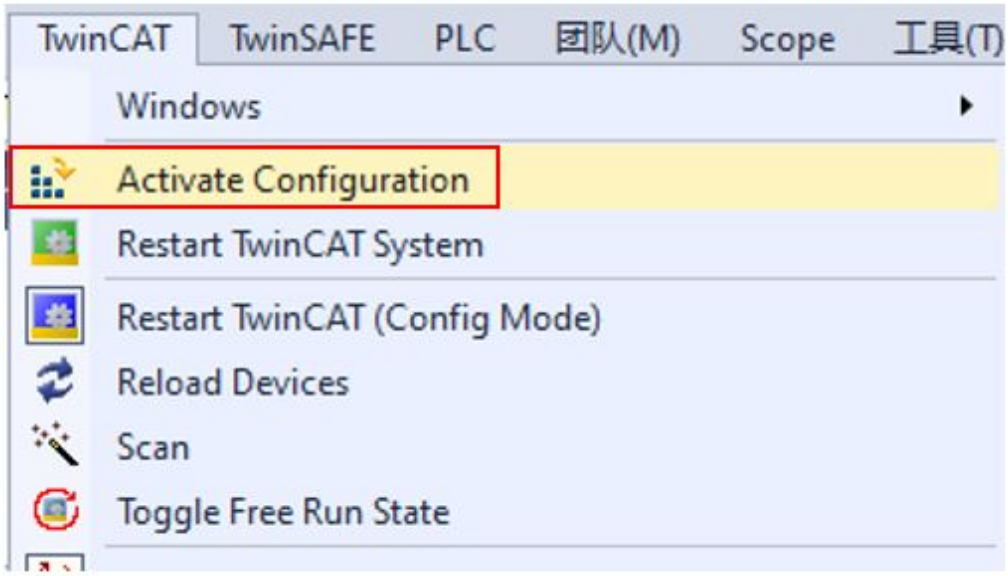


Figure 44. Activatng the Configuration

7. Click **Run PLC** button as shown in Figure 45. Now, the PLC project runs and the user LED on MIMXRT1180-EVK board blinks every one second.

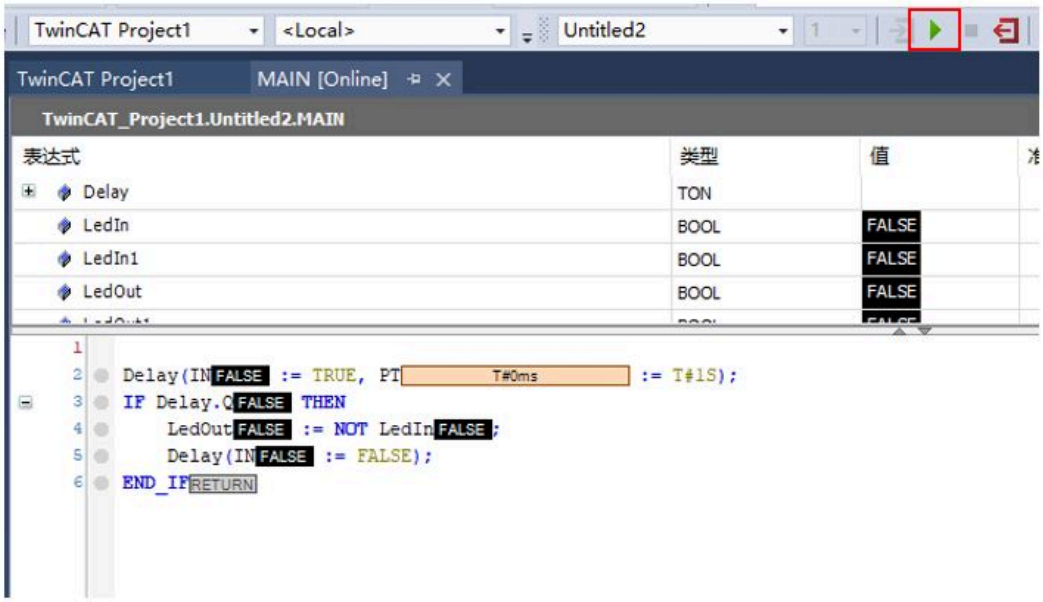


Figure 45. Running the configuration

7 EtherCAT operational modes and mode switching in TwinCAT3

7.1 Switching EtherCAT operational modes in TwinCAT3

This section describes the process for configuring the operation modes using the TwinCAT3 software.

1. Configure SM mode in TwinCAT3. The steps to set SM mode in TwinCAT3 are shown in Figure 46.

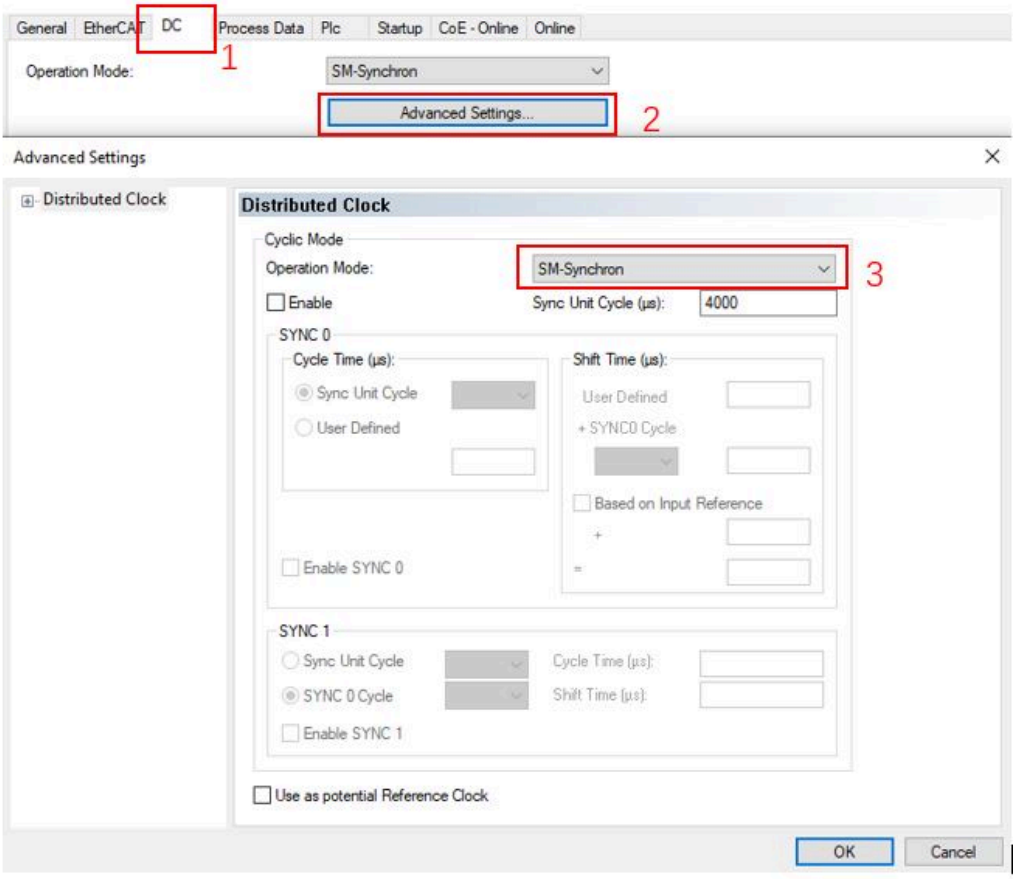


Figure 46. Set SM mode in TwinCAT3

2. Configure DC mode in TwinCAT3. The steps to set DC mode in TwinCAT3 are shown in Figure 47.

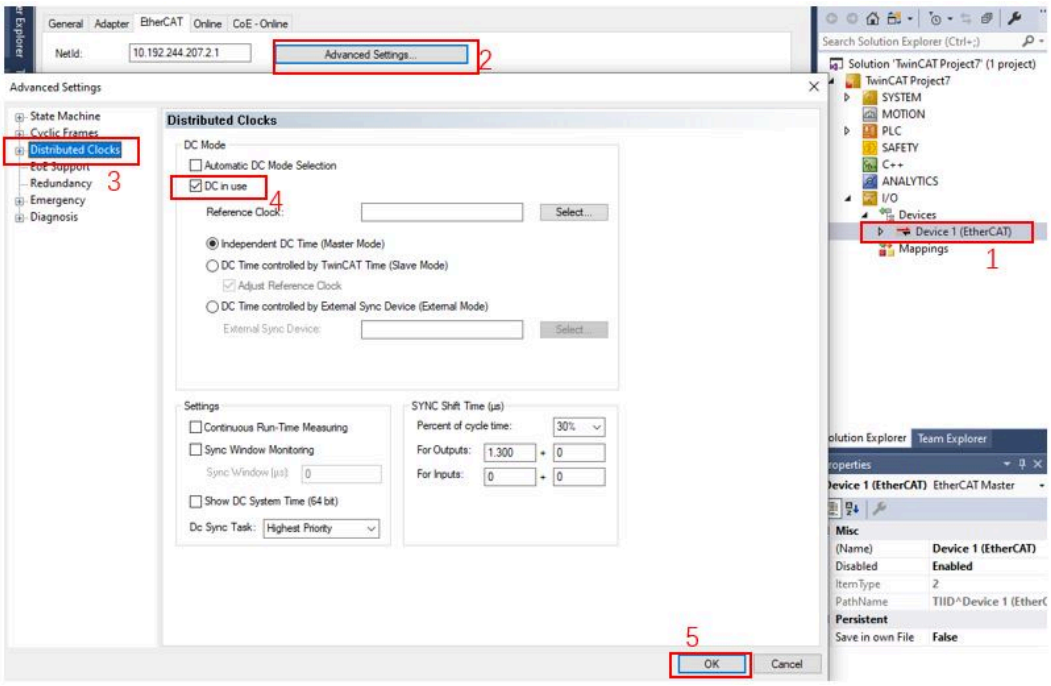


Figure 47. Master Device Set DC mode in TwinCAT3

3. Configure DC mode in TwinCAT3. The steps to set DC mode in TwinCAT3 are shown in Figure 48.

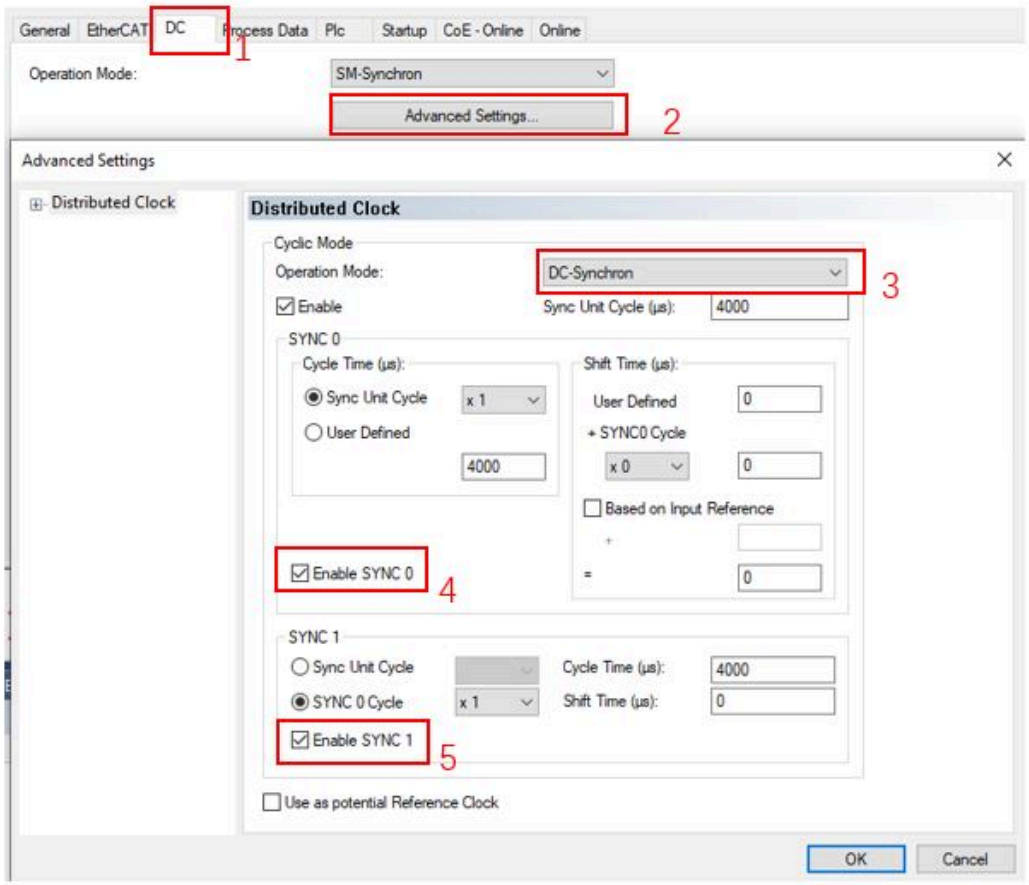


Figure 48. Slave Device Set DC mode in TwinCAT3

8 Conclusion

This application note describes the usage of EtherCAT peripheral on i.MX RT1180 platform. It describes the functionality of FMMU, SyncManager, 64-bit distribute clock, the calculation of propagation delay, and EtherCAT operational modes. It also describes the usage of SSC tool and TwinCAT3 based on the RT1180 SDK and i.MX RT1180 Evaluation Kit (EVK-MIMXRT1180) board.

9 Related documentation

Table 2 lists and explains the additional documents and resources that can be referred to for more information about the MIMXRT1180-EVK board. Some of the documents listed below may be available only under a nondisclosure agreement (NDA). To request access to these documents, contact your local field applications engineer (FAE) or sales representative.

Table 2. Related documentation

Document	Description	Link/how to access
i.MX RT1180 Reference Manual (*IMXRT1180RM*)	Provides a detailed description about the i.MX RT1180 and its features, including memory maps, power supplies, and clocks.	Contact your local field applications engineer (FAE) or sales representative.
i.MX RT1180 Crossover MCUs for Consumer Products Data Sheet (*IMXRT1180CEC*)	Provides information about electrical characteristics, hardware design considerations, and ordering information	
i.MX RT1180 Crossover MCUs for Industrial Products Data Sheet (*IMXRT1180IEC*)	Provides information about electrical characteristics, hardware design considerations, and ordering information	
MIMXRT1180-EVK-UM	Provides a detailed description about MIMXRT1180-EVK board components, interfaces, and functionality	
Security Reference Manual for the i.MX RT1180 Processor	Provides detail about various chip security components	
MCUXpresso Software Development Kit (SDK) documentation	MCUXpresso Software Development Kit (SDK) is a comprehensive software enablement package designed to simplify and accelerate application development with NXP MCUs based on Arm Cortex -M cores.	MCUXpresso Software Development Kit (SDK) documentation

For more information about the i.MX RT1180 MCU, refer to the URL: https://www.nxp.com/products/processors-and-microcontrollers/arm-microcontrollers/i-mx-rt-crossover-mcus/i-mx-rt1180-crossover-mcu-with-tsn-switch-and-edgelock:i.MX-RT1180#documentation.

10 Acronyms

Table 3 lists the acronyms used in this document.

Table 3. Acronyms

Acronym	Description
CM33	Cortex-M33 core
CM7	Cortex-M7 core
DC mode	Distributed Clock mode
EtherCAT	Ethernet for Control Automation Technology
EEPROM	Electrically Enabled Programmable Read Only Memory
ESC	EtherCAT Slave Controller
FMMU	Fieldbus Memory Management Unit
IDE	Integrated Design Environment
PDI	Process Data Interface
SDK	Software Development Kit
SM	Sync Manager
SSC	Slave Stack Code
TSN	Time Sensitive Networking
TwinCAT	The Windows Control and Automation Technology

11 Revision history

Table 4 summarizes the revisions to this document.

Table 4. Revision history

Document ID	Release date	Description
AN14155 v.1.0	23 May 2024	Initial public release

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Contents