acontis technologies GmbH

SOFTWARE

EC-Master

Feature Pack External-Synchronization

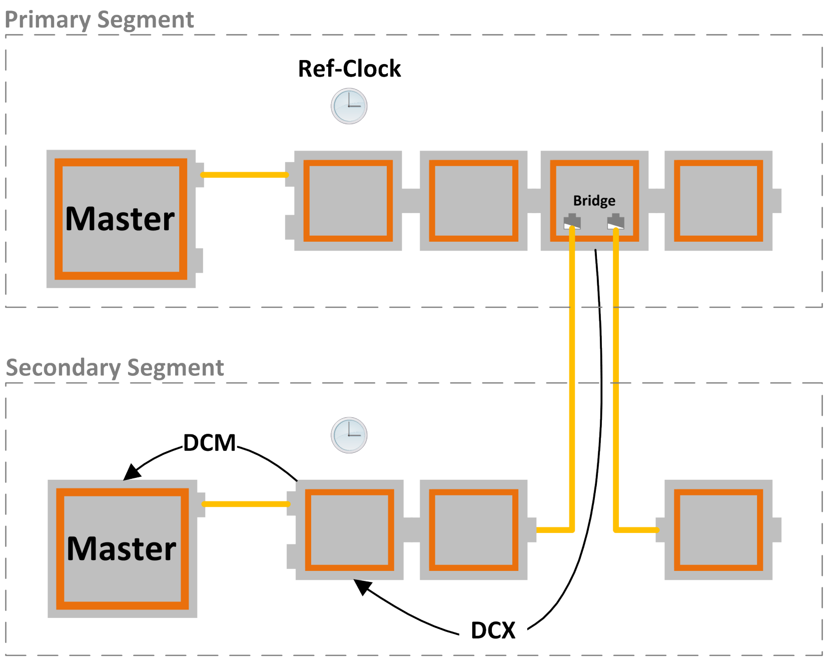
Version 3.2

Edition: September 23, 2025

Contents

1 Introduction

The external synchronization feature pack allows the synchronization of two or more EtherCAT segments by a Bridge device as shown in the picture below.



The Bridge has two EtherCAT connections. The primary port is connected to the primary segment; the secondary port is connected to the secondary segment. The Bridge provides an internal (primary port) and an external (secondary port) time stamp which is used by the Master to adjust the Ref-Clock.

The Bridge device must support the *"External Synchronization Status"* PDO 0x10F4 see document ETG.1020 chapter *"21.1.2 Synchronization by a Bridge device"*.

During startup the two segments can be powered-on at different times. That means that there will be an absolute time difference between the two segments.

The synchronization process is divided in two parts; DCM and DCX.

- DCM: MasterShift, synchronize Master timer to slave.
- DCX: BusShift, Synchronize slaves to bridge device.

Note: This document contains only *"External Synchronization"* specific details. For any basic knowledge about EC-Master, Distributed Clocks and DCM that may be required, see the EC-Master manuals https://developer.acontis.com/ec-master#manuals.

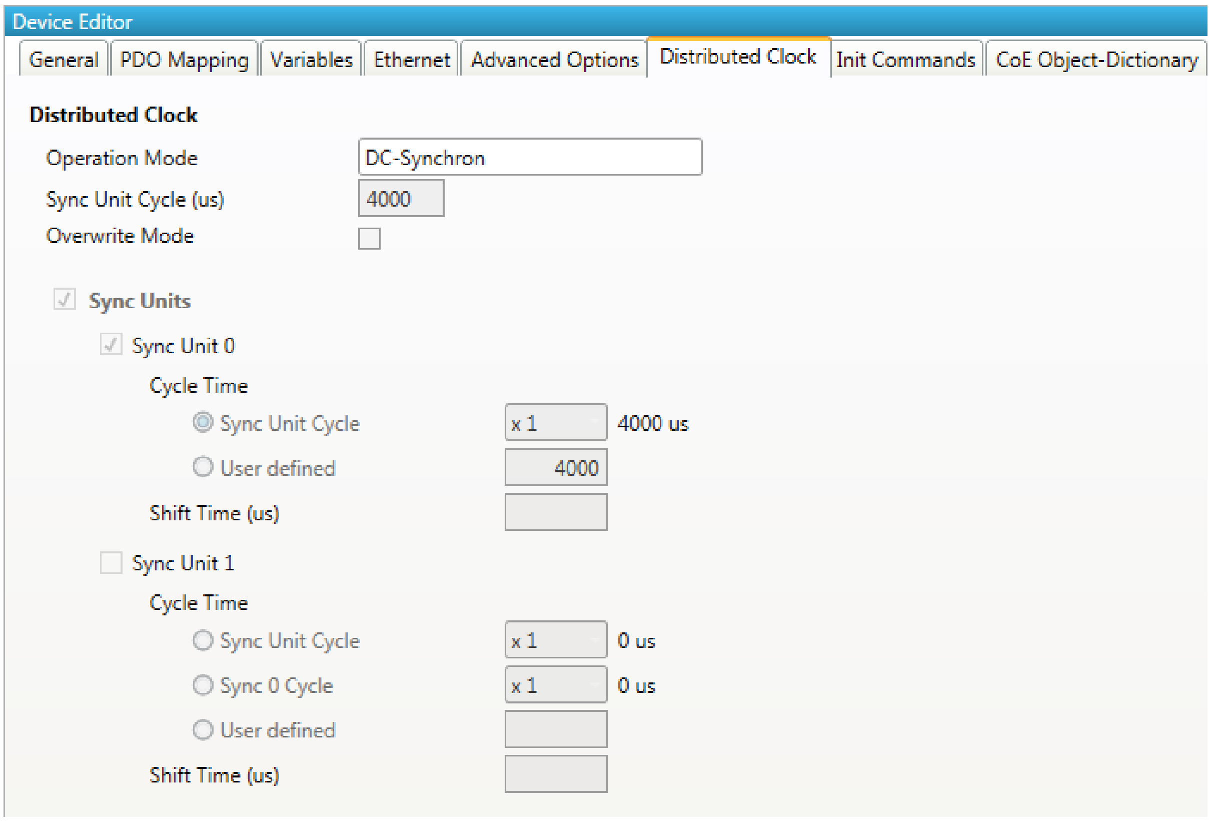
2 Configuration with EC-Engineer

Since version 2.5.0 of the EC-Engineer the configuration of the reference clock adjustment by external synchronization device is supported.

2.1 Primary Segment

The following steps are required to create a DCX-capable configuration in the primary segment.

1. Scan the EtherCAT bus.
2. Select the Bridge Device (e.g. EL6692) and enable DC-Synchron mode.



Distributed Clock

Operation Mode	DC-Synchron
Sync Unit Cycle (us)	4000
Overwrite Mode	☐

✓ **Sync Units**

✓ Sync Unit 0

Cycle Time

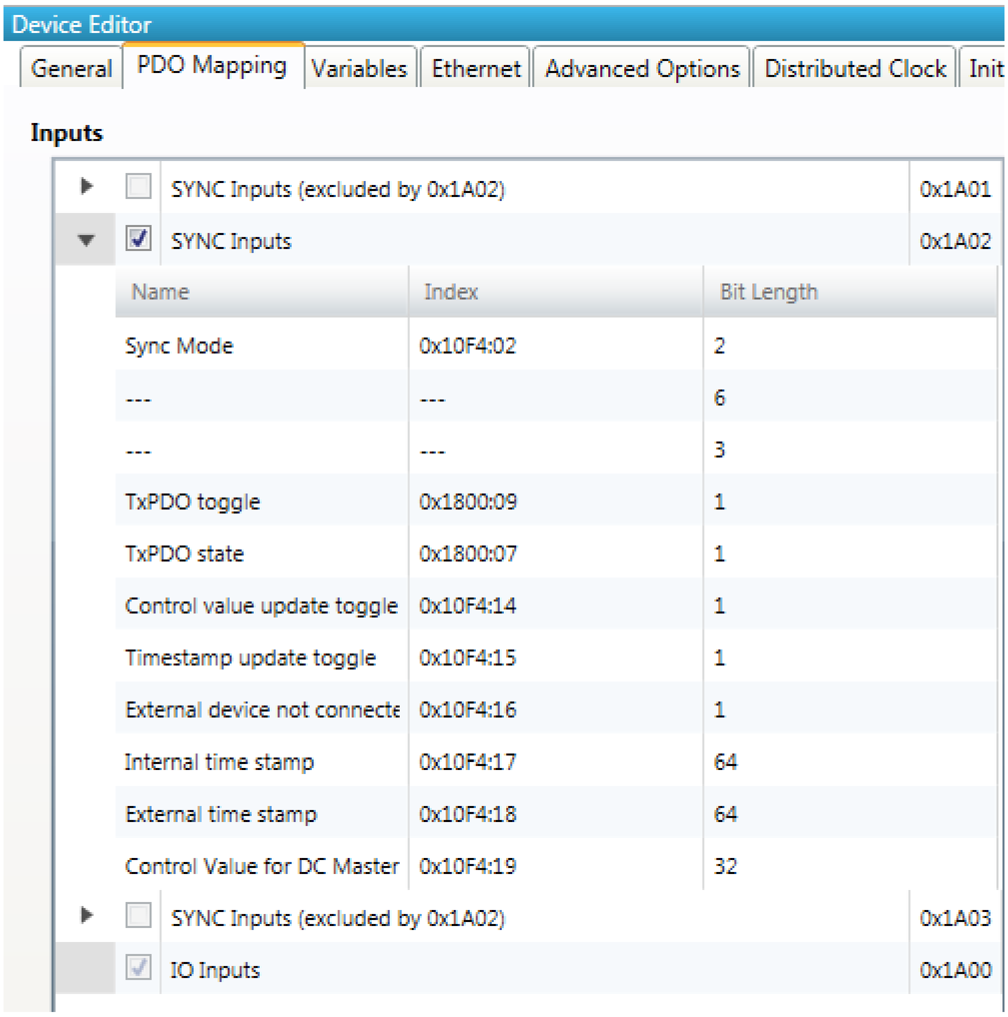
[✓ Sync Unit Cycle]	x 1	4000 us
User defined	4000	
Shift Time (us)		

☐ Sync Unit 1

Cycle Time

Sync Unit Cycle	x 1	0 us
Sync 0 Cycle	x 1	0 us
User defined		
Shift Time (us)		

3. Activate PDO 0x1A02 to display the time stamps.


Device Editor

General | PDO Mapping | Variables | Ethernet | Advanced Options | Distributed Clock | Init

Inputs

☐ SYNC Inputs (excluded by 0x1A02) 0x1A01

✓ SYNC Inputs 0x1A02

Name	Index	Bit Length
Sync Mode	0x10F4:02	2
---	---	6
---	---	3
TxPDO toggle	0x1800:09	1
TxPDO state	0x1800:07	1
Control value update toggle	0x10F4:14	1
Timestamp update toggle	0x10F4:15	1
External device not connecte	0x10F4:16	1
Internal time stamp	0x10F4:17	64
External time stamp	0x10F4:18	64
Control Value for DC Master	0x10F4:19	32

☐ SYNC Inputs (excluded by 0x1A02) 0x1A03

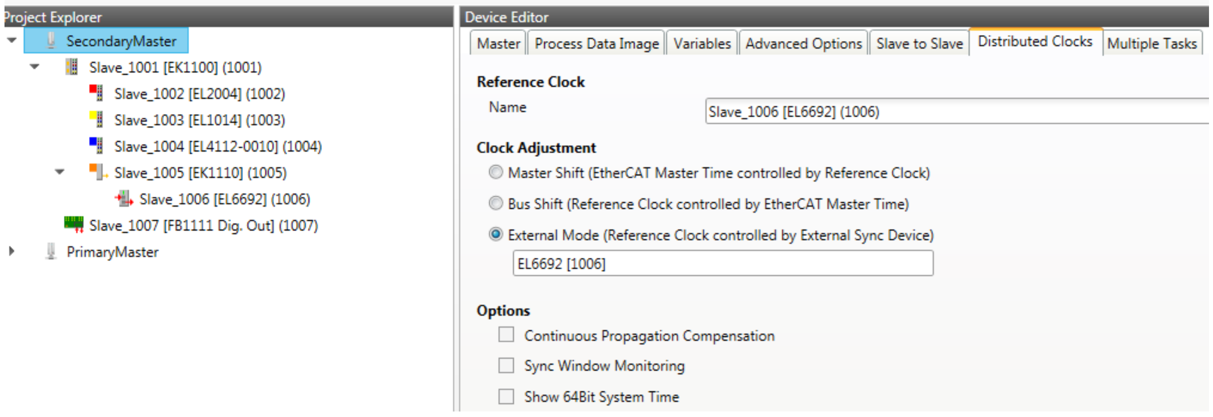
✓ IO Inputs 0x1A00

4. Export ENI

2.2 Secondary Segment

The following steps are required to create a DCX-capable configuration in the secondary segment.

1. Scan the EtherCAT bus.
2. Select the Bridge Device (e.g. EL6692) and enable DC-Synchron mode.
3. Enable External Mode in the Distribute Clocks tab of the Master.


Project Explorer

- SecondaryMaster
  - Slave_1001 [EK1100] (1001)
    - Slave_1002 [EL2004] (1002)
    - Slave_1003 [EL1014] (1003)
    - Slave_1004 [EL4112-0010] (1004)
    - Slave_1005 [EK1110] (1005)
      - Slave_1006 [EL6692] (1006)
  - Slave_1007 [FB1111 Dig. Out] (1007)
- PrimaryMaster

Device Editor

Master | Process Data Image | Variables | Advanced Options | Slave to Slave | Distributed Clocks | Multiple Tasks

Reference Clock

Name: Slave_1006 [EL6692] (1006)

Clock Adjustment

○ Master Shift (EtherCAT Master Time controlled by Reference Clock)

○ Bus Shift (Reference Clock controlled by EtherCAT Master Time)

● External Mode (Reference Clock controlled by External Sync Device)

EL6692 [1006]

Options

☐ Continuous Propagation Compensation

☐ Sync Window Monitoring

☐ Show 64Bit System Time

4. Export ENI

3 Application programming interface, reference

3.1 emDcmConfigure

static EC_T_DWORD **ecatDcmConfigure** (
    EC_T_DCM_CONFIG *pDcmConfig,
    EC_T_DWORD dwInSyncTimeout
)

EC_T_DWORD **emDcmConfigure** (
    EC_T_DWORD dwInstanceID,
    EC_T_DCM_CONFIG *pDcmConfig,
    EC_T_DWORD dwInSyncTimeout
)

Configure DC master synchronization.

Parameters

- **dwInstanceID** – [in] Instance ID (Multiple EtherCAT Network Support)
- **pDcmConfig** – [in] Configuration information, a pointer to a structure of type *EC_T_DCM_CONFIG*.
- **dwInSyncTimeout** – [in] Currently not implemented.

Returns

EC_E_NOERROR or error code

struct **EC_T_DCM_CONFIG**

Public Members

EC_T_DCM_MODE **eMode**
[in] DCM mode

EC_T_DCM_CONFIG_BUSSHIFT **BusShift**
[in] BusShift configuration. Valid if eMode is set to eDcmMode_BusShift

EC_T_DCM_CONFIG_MASTERSHIFT **MasterShift**
[in] MasterShift configuration. Valid if eMode is set to eDcmMode_MasterShift

EC_T_DCM_CONFIG_LINKLAYERREFCLOCK **LinkLayerRefClock**
[in] LinkLayerRefClock configuration. Valid if eMode is set to eDcmMode_LinkLayerRefClock

EC_T_DCM_CONFIG_MASTERREFCLOCK **MasterRefClock**
[in] MasterRefClock configuration. Valid if eMode is set to eDcmMode_MasterRefClock

EC_T_DCM_CONFIG_DCX **Dcx**
[in] DCX configuration. Valid if eMode is set to eDcmMode_Dcx

struct **EC_T_DCM_CONFIG_DCX**

Public Members

EC_T_DCM_CONFIG_MASTERSHIFT **`MasterShift`**
[in] DCM MasterShift configuration

EC_T_INT **`nCtlSetVal`**
[in] Controller set value [ns]. This is the time distance between the cyclic frame send time and the DC base on bus (SYNC0 if shift is zero)

EC_T_INT **`nCtlGain`**
[in] Proportional gain in ppt (part per thousand). Default is value 2. A value of 0 let the current setting unmodified

EC_T_INT **`nCtlDriftErrorGain`**
[in] Multiplier for drift error. Default value is 3. A value of 0 let the current setting unmodified

EC_T_INT **`nMaxValidVal`**
[in] Error inputs above this value are considered invalid. If error input prediction is valid then the difference between the error input and the expected value is taken. Default value is 3000. A value of 0 let the current setting unmodified

EC_T_BOOL **`bLogEnabled`**
[in] If set to EC_TRUE, logging information are generated and can be get calling emDcmGetLog

EC_T_DWORD **`dwInSyncLimit`**
[in] Limit [ns] for InSync monitoring. Default value is 20% of the cycle time. A value of 0 sets the default value.

EC_T_DWORD **`dwInSyncSettleTime`**
[in] Settle time [ms] for InSync monitoring. Default value is 1500ms. A value of 0 sets the default value.

EC_T_BOOL **`bCtlOff`**
[in] If set to EC_TRUE, control loop is disabled. Combined with bLogEnabled, it makes possible to analyze the natural drift between the stack cycle and the reference clock. Also it provides reading of current adjustment value using emDcmGetAdjust function

EC_T_WORD **`wExtClockFixedAddr`**
[in] Fixed address of external clock slave (publishing PDO 0x10F4) (optional if ENI is generated by EcEngineer)

EC_T_DWORD **`dwExtClockTimeout`**
[in] Wait timeout for external clock slave

EC_T_DWORD **`dwInSyncStartDelayCycle`**
[in] Delay time [ms] before InSync monitoring start

EC_T_DWORD **`dwMaxErrCompensableOnExtClockReconnect`**
[in] Maximum error in nanoseconds that should be compensated after a reconnect of the external clock device. Synchronization restart if error exceeds this limit.

3.2 emDcxGetStatus

```
static EC_T_DWORD ecatDcxGetStatus (
        EC_T_DWORD *pdwErrorCode,
        EC_T_INT *pnDiffCur,
        EC_T_INT *pnDiffAvg,
        EC_T_INT *pnDiffMax,
        EC_T_INT64 *pnTimeStampDiff
)
EC_T_DWORD emDcxGetStatus (
        EC_T_DWORD dwInstanceID,
        EC_T_DWORD *pdwErrorCode,
        EC_T_INT *pnDiffCur,
        EC_T_INT *pnDiffAvg,
        EC_T_INT *pnDiffMax,
        EC_T_INT64 *pnTimeStampDiff
)
```

Get DC master external synchronization controller status.

Parameters

- **`dwInstanceID`** – [in] Instance ID (Multiple EtherCAT Network Support)
- **`pdwErrorCode`** – [out] DCX controller error code
- **`pnDiffCur`** – [out] Current difference between set value and actual value of controller in nanoseconds.
- **`pnDiffAvg`** – [out] Average difference between set value and actual value of controller in nanoseconds.
- **`pnDiffMax`** – [out] Maximum difference between set value and actual value of controller in nanoseconds.
- **`pnTimeStampDiff`** – [out] Difference between external and internal timestamps

Returns

EC_E_NOERROR or error code

3.3 Notifications

At startup the EC-Master raises the notifications `EC_NOTIFY_DC_SLV_SYNC`, `EC_NOTIFY_DC_STATUS` and `EC_NOTIFY_DCM_SYNC`, `EC_NOTIFY_DCX_SYNC` at master state transition from INIT to PREOP.

The order is typically as follows (`EC_NOTIFY_DCM_SYNC`, `EC_NOTIFY_DCX_SYNC` may be before or after reaching PREOP):

INIT

`EC_NOTIFY_STATECHANGED`

[…]

`EC_NOTIFY_DC_SLV_SYNC`

[…]

`EC_NOTIFY_DC_STATUS`

[…]

[`EC_NOTIFY_DCM_SYNC`]

[`EC_NOTIFY_DCX_SYNC`]

PREOP

EC_NOTIFY_STATECHANGED

[…]

[EC_NOTIFY_DCM_SYNC]

[EC_NOTIFY_DCX_SYNC]

SAFEOP

EC_NOTIFY_STATECHANGED

OP EC_NOTIFY_STATECHANGED

3.3.1 emNotify – EC_NOTIFY_DCX_SYNC

emNotify - EC_NOTIFY_DCX_SYNC

Parameter

- `pbyInBuf`: [in] Pointer to notification descriptor EC_T_DCX_SYNC_NTFY_DESC
- `dwInBufSize`: [in] sizeof(EC_T_DCX_SYNC_NTFY_DESC).
- `pbyOutBuf`: [out] Should be set to EC_NULL
- `dwOutBufSize`: [in] Should be set to 0
- `pdwNumOutData`: [out] Should be set to EC_NULL

struct **`EC_T_DCX_SYNC_NTFY_DESC`**

Public Members

EC_T_DWORD **`IsInSync`**
EC_TRUE if external(other EtherCAT segment) and internal reference clock are in sync respectively

EC_T_INT **`nCtlErrorNsecCur`**
Current DCX controller error [ns]

EC_T_INT **`nCtlErrorNsecAvg`**
Average DCX controller error [ns]

EC_T_INT **`nCtlErrorNsecMax`**
Maximum DCX controller error [ns]

EC_T_INT64 **`nTimeStampDiff`**
Difference between external and internal time stamp [ns]

EC_T_DWORD **`dwErrorCode`**
DCX external clock error code

4 Examples

4.1 Configuration

The following example code demonstrates the DCX external synchronization configuration of the secondary segment. To configure the primary segment in DCM Mastershift mode, see document EC-Master ClassA.

```
    /* additional DC configuration */
    EC_T_DC_CONFIGURE oDcConfigure;
    OsMemset(&oDcConfigure, 0, sizeof(EC_T_DC_CONFIGURE));
    /* Enable acyclic distribution if cycle time is above 1000 usec to get DCX in␣
↪sync */
    oDcConfigure.bAcycDistributionDisabled = EC_FALSE;

    /* configure DCX external synchronization */
    EC_T_DWORD dwBusCycleTimeUsec = 1000;
    EC_T_DCM_CONFIG oDcmConfig;
    OsMemset(&oDcmConfig, 0, sizeof(EC_T_DCM_CONFIG));

    oDcmConfig.eMode = eDcmMode_Dcx;
    /* Mastershift */
    oDcmConfig.u.Dcx.MasterShift.nCtlSetVal = (dwBusCycleTimeUsec * 1000 * 2) / 3;␣
↪/* 66% */
    /* 20 % limit in nsec for InSync monitoring */
    oDcmConfig.u.Dcx.MasterShift.dwInSyncLimit = (dwBusCycleTimeUsec * 1000) / 5; ␣
↪/* 20% */
    oDcmConfig.u.Dcx.MasterShift.bLogEnabled = EC_FALSE;
    /* Dcx Busshift */
    oDcmConfig.u.Dcx.nCtlSetVal = (dwBusCycleTimeUsec * 1000 * 2) / 3; /* 66% */
    /* 20 % limit in nsec for InSync monitoring */
    oDcmConfig.u.Dcx.dwInSyncLimit = (dwBusCycleTimeUsec * 1000) / 5;
    oDcmConfig.u.Dcx.bLogEnabled = EC_FALSE;
    oDcmConfig.u.Dcx.dwExtClockTimeout = 1000;
    oDcmConfig.u.Dcx.wExtClockFixedAddr = 0; /* 0 only when clock adjustment in␣
↪external mode configured by EcEngineer */

    dwRes = emDcmConfigure(dwInstanceId, &oDcmConfig, 0);
```

4.2 EcMasterDemoDc

The external synchronization feature is also included in the demo application *EcMasterDemoDc*.

EcMasterDemoDc must run on the primary and secondary EtherCAT segments with different DCM configurations:

Primary
`-dcmmode mastershift`

Secondary
`-dcmmode dcx`

To start the demo the full path and file name of the configuration file has to be given as a command line parameter as well as the appropriate Real-time Ethernet Driver.

Example primary segment:

```
> EcMasterDemoDc -intelgbe 1 1 -f PrimaryENI.xml -dcmmode mastershift
```

Example secondary segment:

```
> EcMasterDemoDc -intelgbe 1 1 -f SecondaryENI.xml -dcmmode dcx
```