# SyncEthernetNode

## Reference Manual

| Product Info | |
|---|---|
| Product Manager | Sven Meier |
| Author(s) | Sven Meier |
| Reviewer(s) | Thomas Schaub |
| Version | 1.1 |
| Date | 23.01.2024 |

## Copyright Notice



## Disclaimer

The information available to you in this document/code may contain errors and is subject to periods of interruption. While NetTimeLogic GmbH does its best to maintain the information it offers in the document/code, it cannot be held responsible for any errors, defects, lost profits, or other consequential damages arising from the use of this document/code.

NETTIMELOGIC GMBH PROVIDES THE INFORMATION, SERVICES AND PRODUCTS AVAILABLE IN THIS DOCUMENT/CODE "AS IS," WITH NO WARRANTIES WHATSOEVER. ALL EXPRESS WARRANTIES AND ALL IMPLIED WARRANTIES, INCLUDING WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF PROPRIETARY RIGHTS ARE HEREBY DISCLAIMED TO THE FULLEST EXTENT PERMITTED BY LAW. IN NO EVENT SHALL NETTIMELOGIC GMBH BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL, SPECIAL AND EXEMPLARY DAMAGES, OR ANY DAMAGES WHATSOEVER, ARISING FROM THE USE OR PERFORMANCE OF THIS DOCUMENT/CODE OR FROM ANY INFORMATION, SERVICES OR PRODUCTS PROVIDED THROUGH THIS DOCUMENT/CODE, EVEN IF NETTIMELOGIC GMBH HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

IF YOU ARE DISSATISFIED WITH THIS DOCUMENT/CODE, OR ANY PORTION THEREOF, YOUR EXCLUSIVE REMEDY SHALL BE TO CEASE USING THE DOCUMENT/CODE.

# Overview

NetTimeLogic's Synchronous Ethernet (SyncE) Node is a full hardware (FPGA) only implementation of an ESMC frame Handler and State selector.
The whole ESMC frame sending and receiving & decoding as well as State selection according to the selected option mode are implemented in the core, no CPU is required. This allows running SyncE synchronization completely independent and standalone from the user application. The core can be configured either by signals or by an AXI4Lite-Slave Register interface.

# Key Features:

- Synchronous Ethernet (SyncE) Node
- Supports ESMC and Enhanced ESMC Message according to ITU-T G.8264
- Automatic selection of State according to SSM codes for SyncE option 1 or 2 networks and QL values as defined in ITU-T G.781
- Manual State override mode
- AXI4Lite register set or static configuration
- Clock Mux Module
- PPS Generator Module
- Tx Clock Generator Module

# Revision History

This table shows the revision history of this document.

| Version | Date | Revision |
|---|---|---|
| 0.1 | 20.10.2023 | First draft |
| 1.0 | 13.11.2023 | SSM and ESSM Code to QL mapping added and first release |
| 1.1 | 23.01.2024 | Added Interface Adpater |

Table 1: Revision History

# Content

# Definitions

| Definitions | |
|---|---|
| SyncE Node | A node that can synchronize itself to an SyncE input or sync other SyncE nodes based on ESMC |
| SSM Code | Synchronization Status Message code is a specific field within the ESMC message |
| ESMC Message | ESMC messages are a key component of the SyncE protocol used for communication of the clock quality level. |

Table 2: Definitions

# Abbreviations

| Abbreviations | |
|---|---|
| AXI | AMBA4 Specification (Stream and Memory Mapped) |
| BCD | Binary Coded Decimal |
| PWM | Pulse Width Modulation |
| IRQ | Interrupt, Signaling to e.g. a CPU |
| SYNCE | Synchronous Ethernet |
| SSM | Synchronization Status Message |
| ESSM | Enhanced Synchronization Status Message |
| ESMC | Ethernet Synchronization Messaging Channel |
| TS | Timestamp |
| TB | Testbench |
| LUT | Look Up Table |
| FF | Flip Flop |
| RAM | Random Access Memory |
| ROM | Read Only Memory |
| FPGA | Field Programmable Gate Array |
| VHDL | Hardware description Language for FPGA's |

Table 3: Abbreviations

# 1 Introduction

## 1.1 Context Overview

The SyncE Node is meant as a co-processor handling an ESMC Message handling and State selection.
It sends and receives ESMC messages via AXI Stream. The SyncE Node is intended to be used with NetTimeLogic's PTP Ordinary Clock (not a requirement) and intercepts the path between a MAC and PHY before e.g. a PTP OC core. It contains an AXI4Lite slave for configuration and status supervision from a CPU, this is however not required since the SyncE Node can also be configured statically via signals/constants directly from the FPGA.
Additionally it has modules for clock muxing or PPS Generation for numeric frequency adjustments

As mentioned, the SyncE Node can be combined with NetTimeLogic's PTP OC. Contact us for details.

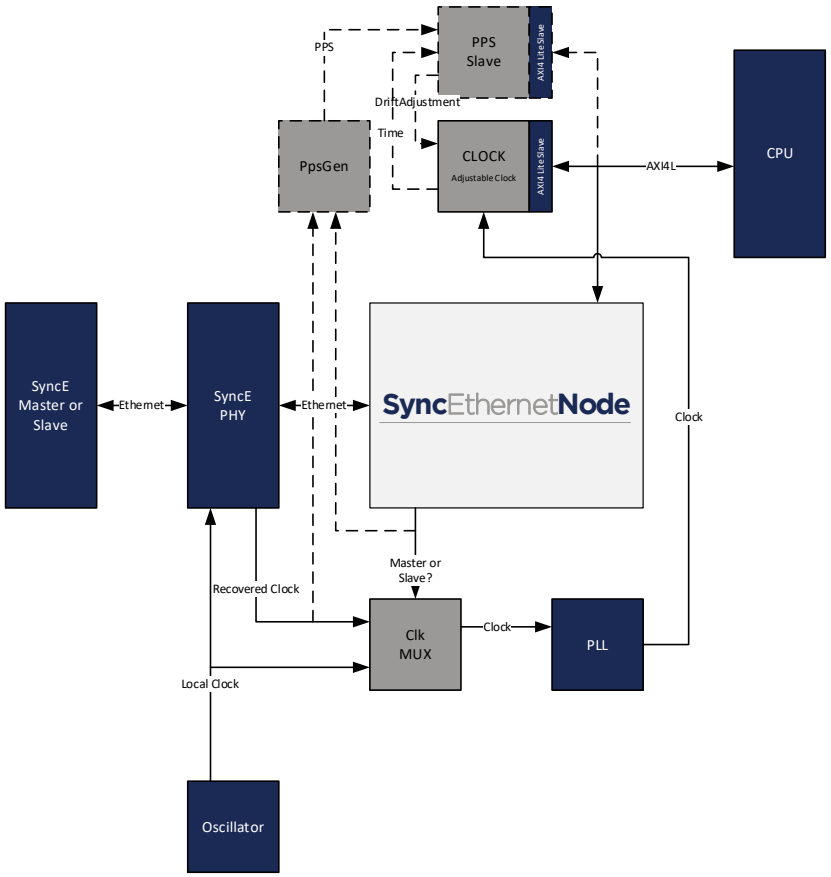


Figure 1: Context Block Diagram

## 1.2 Function

The Synchronous Ethernet Node periodically (1s), and on a change of the SSM code, sends ESMC and Enhanced ESMC frames according to the user configuration. It also receives ESMC and Enhanced ESMC frames extracts the SSM, Enhanced ESMC, Clock Identity etc. from the received TLVs and provides this in Registers to the User and in parallel to a Selection Processor which compares the configured SSM values with the received SSM values and provides a State recommendation according to the Quality Level encoding of ITU-T G.781 for option 1 and 2 networks.
In parallel it makes the Timeout supervision (5s).

## 1.3 Architecture

The core is split up into different functional blocks for reduction of the complexity, modularity and maximum reuse of blocks. The interfaces between the functional blocks are kept as small as possible for easier understanding of the core.

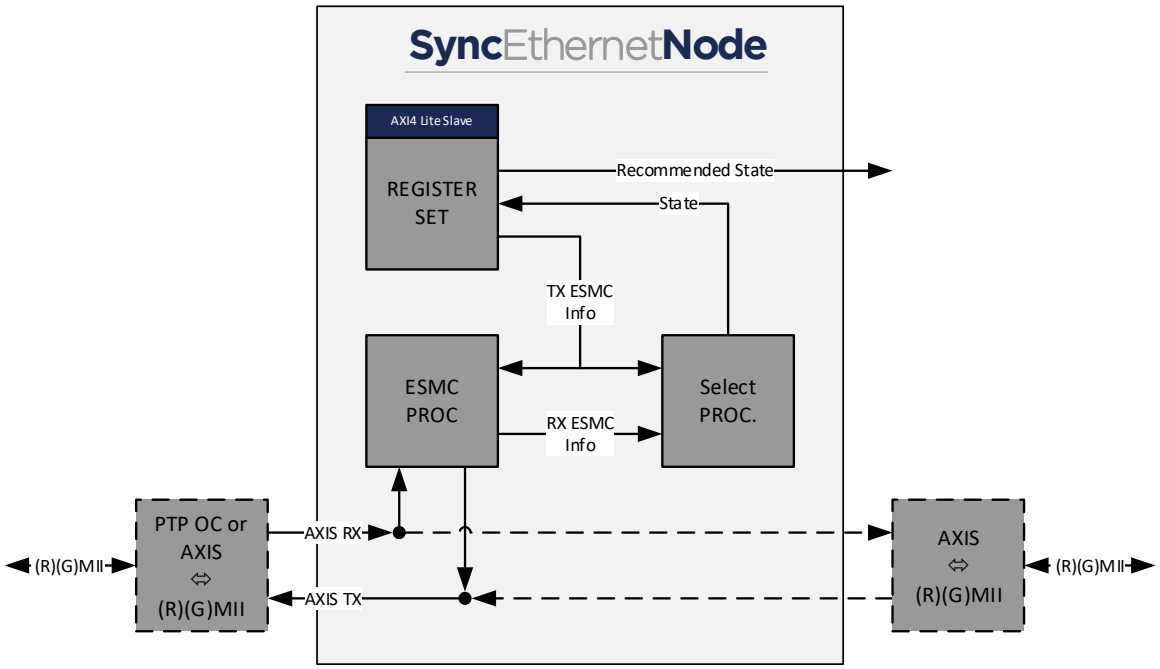


Figure 2: Architecture Block Diagram

### Register Set

This block allows reading status values and writing configuration.

### ESMC Processor

This block sends and receives ESMC and Enhanced ESMC messages. It also parses and extracts the ESMC TLVs and provides the extracted Info to the Registerset and to the Select Processor.

### Select Processor

This block compares the configured and received SSM codes according to their priority encoding for option 1 & 2 SyncE Networks and provides the recommended State for the Clock Selection.

### (R)(G)MII Receive/Transmit Interface Adapter

These blocks convert the data stream from the (R)(G)MII to a 32bit AXI stream and back form 32bit AXI stream to (R)(G)MII.

# 2 SyncE Basics

Synchronous Ethernet (SyncE) is a networking technology designed to provide precise clock synchronization in Ethernet networks. In traditional Ethernet, devices rely on their internal clocks, which may lead to variations in timing, causing issues in applications that require tight synchronization, such as telecommunications.

SyncE is introducing a synchronization hierarchy where a master clock distributes a highly accurate timing reference to slave clocks throughout the network. One key feature of SyncE is its ability to operate over existing Ethernet infrastructure, making it a cost-effective solution for improving synchronization in network environments. It utilizes specific synchronization messages and mechanisms to align the clocks across devices, ensuring consistent and reliable timing.

These messages are called ESMC. ESMC stands for “Ethernet Synchronization Messaging Channel". ESMC messages are a key component of the SyncE protocol and are used for communication between the master clock and the slave clocks within a synchronized network. These messages play a crucial role in maintaining and distributing accurate timing information across the network.

The SSM (Synchronization Status Message) code is a specific field within the ESMC message. The SSM code is used to convey information about the quality or status of the synchronization signal being transmitted. It provides details on the reliability and accuracy of the synchronization source. The specific codes used in the SSM field are defined by relevant standards to ensure consistency and interoperability in synchronization implementations.
.

# 3 Register Set

This is the register set of the SyncE Node. It is accessible via AXI4Lite Memory Mapped. All registers are 32-bit wide, no burst access, no unaligned access, no byte enables, no timeouts are supported. Register address space is not contiguous. Register addresses are only offsets in the memory area where the core is mapped in the AXI interconnects. Non existing register access in the mapped memory area is answered with a slave decoding error.

## 3.1 Register Overview

| Registerset Overview | | | |
|---|---|---|---|
| Name | Description | Offset | Access |
| Synce NodeControl Reg | SyncE Enable Control Register | 0x00000000 | RW |
| Synce NodeStatus Reg | SyncE Error Status Register | 0x00000004 | WC |
| Synce NodeVersion Reg | SyncE Version Register | 0x0000000C | RO |
| Synce NodeTxEsmcSsmCode Reg | SyncE Transmit ESMC SSM Code Register | 0x00000010 | RW |
| Synce NodeTxEnhEsmcSsmCodeF Reg | SyncE Transmit Enhanced ESMC SSM Code Flags Register | 0x00000020 | RW |
| Synce NodeTxEnhEsmcCockId1 Reg | SyncE Transmit Enhanced ESMC ClockId1 Register | 0x00000024 | RW |
| Synce NodeTxEnhEsmcClockId2 Reg | SyncE Transmit Enhanced ESMC ClockId2 Register | 0x00000028 | RW |
| Synce NodeTxEnhCasNrofEec Reg | SyncE Transmit Enhanced Cascaded Number of Eec Register | 0x0000002C | RW |
| Synce NodeMacControl Reg | SyncE Mac Control Register | 0x00000100 | RW |
| Synce NodeMac1 Reg | SyncE Mac1 Register | 0x00000104 | RW |
| Synce NodeMac2 Reg | SyncE Mac2 Register | 0x00000108 | RW |
| Synce NodeRxControl Reg | SyncE Receive Control Register | 0x00000200 | RW |
| Synce NodeRxEsmcSsmCode Reg | SyncE Receive ESMC SSM Code Register | 0x00000210 | RO |
| Synce NodeRxEnhEsmcSsmCodeF Reg | SyncE Receive Enhanced ESMC SSM Code Flags Register | 0x00000220 | RO |

| | | | |
|---|---|---|---|
| Synce NodeRxEnhEsmcClockId1 Reg | SyncE Receive Enhanced ESMC ClockId1 Register | 0x00000224 | RO |
| Synce NodeRxEnhEsmcClockId2 Reg | SyncE Receive Enhanced ESMC ClockId2 Register | 0x00000228 | RO |
| Synce NodeRxEnhCasNrofEec Reg | SyncE Receive Enhanced Cascaded Number of Eec Register | 0x0000022C | RO |

Table 4: Register Set Overview

## 3.2 Register Descriptions

### 3.2.1 General

#### 3.2.1.1 SyncE Node Control Register

Used for general control over the SyncE Node, all configurations on the core shall only be done when disabled.
It is required to select the right options mode and if ESMC only or ESMC and Enhanced ESMC messages shall be used.
Also the State can be overridden.

| Synce NodeControl Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | | | | | | | | | | | | | | ENHANCED_ESMC | ESMC | - | | SYNCE_OPTION_CODE | | - | | MANUAL_MASTER | MANUAL_SLAVE | - | | | MANUAL_STATE | - | | | ENABLE |
| RO | | | | | | | | | | | | | | RW | RW | RO | | RW | | RO | | RW | RW | RO | | | RW | RO | | | RW |
| Reset: 0x00010000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|

| | | | |
|---|---|---|---|
| - | Reserved, read 0 | Bit:31:18 | RO |
| ENHANCED_ESMC | Enable Enhanced ESMC | Bit:17 | RW |
| ESMC | Enable ESMC (shall always be 1) | Bit:16 | RW |
| - | Reserved, read 0 | Bit:15:14 | RO |
| SYNCE_OPTION_CODE | Sync E option mode:<br>"00": Unknown<br>"01": SyncE option 1 networks<br>"10": SyncE option 2 networks<br>"11": SyncE option 3 networks (not used) | Bit:13:12 | RO |
| - | Reserved, read 0 | Bit:11:10 | RO |
| MANUAL_MASTER | Set to Manual Master (if Manual State is selected) | Bit:9 | RO |
| MANUAL_SLAVE | Set to Manual Slave (if Manual State is selected) | Bit:8 | RO |
| - | Reserved, read 0 | Bit:7:5 | RO |
| MANUAL_STATE | Set Manual States | Bit:4 | RO |
| - | Reserved, read 0 | Bit:3:1 | RO |
| ENABLE | Enable | Bit: 0 | RW |

### 3.2.1.2 SyncE Node Status Register

Shows the current status of the SyncE Node. It shows if there was a Timeout on ESMC messages, what message TLVs were received and what recommended state is selected.

| Synce NodeStatus Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | | | | | | | | | | | | | | ENHANCED_ESMC | ESMC | - | | | | | | MASTER | SLAVE | - | | | | | | | TIMEOUT |
| RO | | | | | | | | | | | | | | RO | RO | RO | | | | | | RO | RO | RO | | | | | | | RO |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0004 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| - | Reserved, read 0 | Bit:31:18 | RO |
| ENHANCED_ESMC | Received Enhanced ESMC | Bit:17 | RO |
| ESMC | Received ESMC | Bit:16 | RO |
| - | Reserved, read 0 | Bit:15:10 | RO |
| MANUAL_MASTER | Mode Master (both set to 0, Unknown) | Bit:9 | RO |
| MANUAL_SLAVE | Mode Slave (both set to 0, Unknown) | Bit:8 | RO |

| | | | |
|---|---|---|---|
| - | Reserved, read 0 | Bit:7:1 | RO |
| TIMEOUT | Timeout (5s without ESMC) | Bit: 0 | RO |

### 3.2.1.3 SyncE Node Version Register

Version of the IP core even though is seen as a 32bit value, bits 31 down to 24 represent the major, bits 23 down to 16 the minor and bits 15 down to 0 the build numbers.

| Synce NodeVersion Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| VERSION | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| RO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset: 0xXXXXXXXX | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x000C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| VERSION | Version of the IP core | Bit: 31:0 | RO |

### 3.2.1.4 SyncE Node Tx ESMC SSM Code Register

This register contains the SSM Code to be transmitted.

| Synce NodeTxEsmcSsmCode Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | SSM_CODE | | | |
| RO | | | | | | | | | | | | | | | | | | | | | | | | | | | | RW | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0010 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| - | Reserved, read 0 | Bit:31:16 | RO |
| SSM_CODE | ESMC Code | Bit: 15:0 | RW |

### 3.2.1.5 SyncE Node Tx Enhanced ESMC SSM Code and Flags Register

This register contains the Enhanced SSM Code and Flags to be transmitted.

| Synce NodeTxEnhEsmcSsmCodeF Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | | | | | | | | FLAGS | | | | | | | | - | | | | | | | | ESSM_CODE | | | | | | | |
| RO | | | | | | | | RW | | | | | | | | RO | | | | | | | | RW | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0020 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| - | Reserved, read 0 | Bit:31:24 | RO |
| FLAGS | Enhanced ESMC Flags | Bit:23:16 | RW |
| - | Reserved, read 0 | Bit 15:8 | RO |
| ESSM_CODE | Enhanced ESMC SSM Code | Bit:7:0 | RW |

### 3.2.1.6 SyncE Node Tx Enhanced ESMC Clock ID 1 Register

This register contains the Enhanced ESMC Clock Id (first 4 bytes) to be transmitted.

| Synce NodeTxEnhEsmcCockId1 Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CLOCK_ID(3) | | | | | | | | CLOCK_ID(2) | | | | | | | | CLOCK_ID(1) | | | | | | | | CLOCK_ID(0) | | | | | | | |
| RW | | | | | | | | RW | | | | | | | | RW | | | | | | | | RW | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0024 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| CLOCK_ID(3) | Clock ID Byte 3 | Bit:31:24 | RW |
| CLOCK_ID(2) | Clock ID Byte 2 | Bit:23:16 | RW |
| CLOCK_ID(1) | Clock ID Byte 1 | Bit:15:8 | RW |
| CLOCK_ID(0) | Clock ID Byte 0 | Bit:7:0 | RW |

### 3.2.1.7 SyncE Node Tx Enhanced ESMC Clock ID 2 Register

This register contains the Enhanced ESMC Clock Id (second 4 bytes) to be transmitted.

| Synce NodeTxEnhEsmcCockId2 Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CLOCK_ID(7) | | | | | | | | CLOCK_ID(6) | | | | | | | | CLOCK_ID(5) | | | | | | | | CLOCK_ID(4) | | | | | | | |
| RW | | | | | | | | RW | | | | | | | | RW | | | | | | | | RW | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0028 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| CLOCK_ID(7) | Clock ID Byte 7 | Bit:31:24 | RW |
| CLOCK_ID(6) | Clock ID Byte 6 | Bit:23:16 | RW |
| CLOCK_ID(5) | Clock ID Byte 5 | Bit:15:8 | RW |
| CLOCK_ID(4) | Clock ID Byte 4 | Bit:7:0 | RW |

### 3.2.1.8 SyncE Node Tx Enhanced ESMC Cascaded EEC Register

This register contains the Enhanced ESMC Cascaded EEC counts to be transmitted.

| Synce NodeTxEnhCasNrofEec Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | | | | | | | | NR_OF_EEEC | | | | | | | | - | | | | | | | | NR_OF_EEC | | | | | | | |
| RO | | | | | | | | RW | | | | | | | | RO | | | | | | | | RW | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x002C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| - | Reserved, read 0 | Bit:31:24 | RO |
| NR_OF_EEEC | Number of cascaded eEEC | Bit:23:16 | RW |
| - | Reserved, read 0 | Bit 15:8 | RO |
| NR_OF_EEC | Number of cascaded EEC | Bit:7:0 | RW |

### 3.2.1.9 SyncE Node MAC Control Register

This register contains the valid flag for the MAC value.

| Synce NodeMacControl Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | MAC_VAL |
| RO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | RW |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0100 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| - | Reserved, read 0 | Bit:31:1 | RO |
| MAC_VAL | MAC valid (auto cleared) | Bit: 0 | RW |

### 3.2.1.10 SyncE Node MAC 1 Register

This register contains the lower 4 bytes of the MAC address used to send ESMC frames.

| Synce NodeMac1 Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MAC(3) | | | | | | | | MAC(2) | | | | | | | | MAC(1) | | | | | | | | MAC(0) | | | | | | | |
| RW | | | | | | | | RW | | | | | | | | RW | | | | | | | | RW | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0104 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| MAC(3) | MAC Byte 3 | Bit:31:24 | RW |
| MAC(2) | MAC Byte 2 | Bit:23:16 | RW |
| MAC(1) | MAC Byte 1 | Bit:15:8 | RW |
| MAC(0) | MAC Byte 0 | Bit:7:0 | RW |

### 3.2.1.11 SyncE Node MAC 2 Register

This register contains the upper 2 bytes of the MAC address used to send ESMC frames.

| Synce NodeMac2 Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | | | | | | | | | | | | | | | | MAC(5) | | | | | | | | MAC(4) | | | | | | | |
| RO | | | | | | | | | | | | | | | | RW | | | | | | | | RW | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0108 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| - | Reserved, read 0 | Bit:31:16 | RO |
| MAC(5) | MAC Byte 5 | Bit:15:8 | RW |
| MAC(4) | MAC Byte 4 | Bit:7:0 | RW |

### 3.2.1.12 SyncE Node Rx Control Register

This register allows to take a snapshot of the received data.

| Synce NodeRxControl Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| READ_DONE | READ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| RO | RW | RO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0200 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| READ_DONE | Rx Data was read | Bit: 31 | RO |
| READ | Read Rx Data (auto cleared) | Bit: 30 | RW |
| - | Reserved, read 0 | Bit 29:0 | RO |

### 3.2.1.13SyncE Node Rx ESMC SSM Code Register

This register contains the SSM Code to be received.

| Synce NodeRxEsmcSsmCode Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | SSM_CODE | | | |
| RO | | | | | | | | | | | | | | | | | | | | | | | | | | | | RW | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0210 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| - | Reserved, read 0 | Bit:31:16 | RO |
| SSM_CODE | ESMC Code Received | Bit: 15:0 | RW |

### 3.2.1.14 SyncE Node Rx Enhanced ESMC SSM Code and Flags Register

This register contains the Enhanced SSM Code and Flags to be received.

| Synce NodeRxEnhEsmcSsmCodeF Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | | | | | | | | FLAGS | | | | | | | | - | | | | | | | | ESSM_CODE | | | | | | | |
| RO | | | | | | | | RW | | | | | | | | RO | | | | | | | | RW | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0220 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| - | Reserved, read 0 | Bit:31:24 | RO |
| FLAGS | Enhanced ESMC Flags Received | Bit:23:16 | RW |
| - | Reserved, read 0 | Bit 15:8 | RO |
| ESSM_CODE | Enhanced ESMC SSM Code Received | Bit:7:0 | RW |

### 3.2.1.15 SyncE Node Rx Enhanced ESMC Clock ID 1 Register

This register contains the Enhanced ESMC Clock Id (first 4 bytes) to be received.

| Synce NodeRxEnhEsmcCockId1 Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CLOCK_ID(3) | | | | | | | | CLOCK_ID(2) | | | | | | | | CLOCK_ID(1) | | | | | | | | CLOCK_ID(0) | | | | | | | |
| RW | | | | | | | | RW | | | | | | | | RW | | | | | | | | RW | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0224 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| CLOCK_ID(3) | Clock ID Byte 3 Received | Bit:31:24 | RW |
| CLOCK_ID(2) | Clock ID Byte 2 Received | Bit:23:16 | RW |
| CLOCK_ID(1) | Clock ID Byte 1 Received | Bit:15:8 | RW |
| CLOCK_ID(0) | Clock ID Byte 0 Received | Bit:7:0 | RW |

### 3.2.1.16SyncE Node Rx Enhanced ESMC Clock ID 2 Register

This register contains the Enhanced ESMC Clock Id (second 4 bytes) to be transmitted.

| Synce NodeRxEnhEsmcCockId2 Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CLOCK_ID(7) | | | | | | | | CLOCK_ID(6) | | | | | | | | CLOCK_ID(5) | | | | | | | | CLOCK_ID(4) | | | | | | | |
| RW | | | | | | | | RW | | | | | | | | RW | | | | | | | | RW | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x0228 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| CLOCK_ID(7) | Clock ID Byte 7 Received | Bit:31:24 | RW |
| CLOCK_ID(6) | Clock ID Byte 6 Received | Bit:23:16 | RW |
| CLOCK_ID(5) | Clock ID Byte 5 Received | Bit:15:8 | RW |
| CLOCK_ID(4) | Clock ID Byte 4 Received | Bit:7:0 | RW |

### 3.2.1.17 SyncE Node Tx Enhanced ESMC Cascaded EEC Register

This register contains the Enhanced ESMC Cascaded EEC counts received.

| Synce NodeRxEnhCasNrofEec Reg | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reg Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | | | | | | | | NR_OF_EEEC | | | | | | | | - | | | | | | | | NR_OF_EEC | | | | | | | |
| RO | | | | | | | | RW | | | | | | | | RO | | | | | | | | RW | | | | | | | |
| Reset: 0x00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Offset: 0x022C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Name | Description | Bits | Access |
|---|---|---|---|
| - | Reserved, read 0 | Bit:31:24 | RO |
| NR_OF_EEEC | Number of cascaded eEEC Received | Bit:23:16 | RW |
| - | Reserved, read 0 | Bit 15:8 | RO |
| NR_OF_EEC | Number of cascaded EEC Received | Bit:7:0 | RW |

# 4 Design Description

The following chapters describe the internals of the SyncE Node: starting with the Top Level, which is a collection of subcores, followed by the description of all subcores.

## 4.1 Top Level – SyncE Node

#### 4.1.1.1 Parameters

The core must be parametrized at synthesis time. There are a couple of parameters which define the final behavior and resource usage of the core.

| Name | Type | Size | Description |
|---|---|---|---|
| ResetBuffer_Gen | boolean | 1 | To add an extra FF for Reset |
| StaticConfig_Gen | boolean | 1 | If Static Configuration or AXI is used |
| ClockClkPeriod Nanosecond_Gen | natural | 1 | Clock Period in Nanosecond: Default for 50 MHz = 20 ns |
| EnhancedEsmc Support_Gen | natural | 1 | Input delay of the SYNCE from the connector to the input signal. |
| EsmcRefresh_Gen | natural | 1 | The refresh rate in Millisec-onds for ESMC Messages, standard 1s |
| EsmcTimeout_Gen | natural | 1 | The Timeout in Milliseconds for ESMC Messages, standard 5s |
| PassThrough_Gen | boolean | 1 | If AXI shall be passed through the core |
| AxiAddressRange Low_Gen | std_logic_vector | 32 | AXI Base Address |
| AxiAddressRange High_Gen | std_logic_vector | 32 | AXI Base Address plus Regis-terset Size<br>Default plus 0xFFFF |
| Sim_Gen | boolean | 1 | If in Testbench simulation mode:<br>true = Simulation, false = |

| | | | |
|---|---|---|---|
| | | | Synthesis |

Table 5: Parameters

### 4.1.1.2 Structured Types

#### 4.1.1.2.1 Synce_NodeStaticConfig_Type

Defined in Synce_NodeAddrPackage.vhd of library SynceLib
This is the type used for static configuration.

| Field Name | Type | Size | Description |
|---|---|---|---|
| OwnMac | Common_ Byte_Type | 6 | MAC Address to be sent |
| Mode | Synce_Select Mode_Type | 1 | Mode: Option 1 or Option 2 mode |
| EsmcSsmCode | std_logic_vector | 3 | ESMC SSM Code |
| Enhanced EsmcSsmCode | std_logic_vector | 8 | Enhanced ESMC SSM Code |
| Enhanced EsmcFlags | std_logic_vector | 8 | Enhanced ESMC Flags |
| Enhanced EsmcClockIdentity | Common_ Byte_Type | 8 | Enhanced ESMC Clock ID |
| Enhanced EsmcCascaded-NrOfEec | std_logic_vector | 8 | Enhanced ESMC Cascaded EEC |
| Enhanced EsmcCascaded-NrOfEnhanced Eec | std_logic_vector | 8 | Enhanced ESMC Cascaded eEEC |

Table 6: Synce_NodeStaticConfig_Type

#### 4.1.1.2.2 Synce_NodeStaticConfigVal_Type

Defined in Synce_NodeAddrPackage.vhd of library SynceLib
This is the type used for valid flags of the static configuration.

| Field Name | Type | Size | Description |
|---|---|---|---|

| | | | |
|---|---|---|---|
| Esmc_Val | std_logic | 1 | If ESMC Message with TLV shall be sent |
| EnhancedEsmc_Val | std_logic | 1 | If ESMC Message with Enhanced TLV shall be sent |
| Enable_Val | std_logic | 1 | Enables the SyncE Node |

Table 7: Synce_NodeStaticConfigVal_Type

#### 4.1.1.2.3 Synce_NodeStaticStatus_Type

Defined in Synce_NodeAddrPackage.vhd of library SynceLib
This is the type used for static status supervision.

| Field Name | Type | Size | Description |
|---|---|---|---|
| CoreInfo | Clk_CoreInfo_Type | 1 | Info about the Cores state |
| UtcInfo | Clk_UtcInfo_Typ | 1 | Info about UTC details |
| EsmcSsmCode | std_logic_vector | 3 | Received ESMC SSM Code |
| Enhanced EsmcSsmCode | std_logic_vector | 8 | Received Enhanced ESMC SSM Code |
| Enhanced EsmcFlags | std_logic_vector | 8 | Received Enhanced ESMC Flags |
| Enhanced EsmcClockIdentity | Common_Byte_Type | 8 | Received Enhanced ESMC Clock ID |
| Enhanced EsmcCascaded-NrOfEec | std_logic_vector | 8 | Received Enhanced ESMC Cascaded EEC |
| Enhanced EsmcCascaded-NrOfEnhanced Eec | std_logic_vector | 8 | Received Enhanced ESMC Cascaded eEEC |

Table 8: Synce_NodeStaticStatus_Type

#### 4.1.1.2.4 Synce_NodeStaticStatusVal_Type

Defined in Synce_NodeAddrPackage.vhd of library SynceLib
This is the type used for valid flags of the static status supervision.

| Field Name | Type | Size | Description |
|---|---|---|---|
| CoreInfo_Val | std_logic | 1 | Core Info valid |
| UtcInfo_Val | std_logic | 1 | UTC Info valid |
| Timeout_Val | std_logic | 1 | Timeout for ESMC messages |
| Esmc_Val | std_logic | 1 | If ESMC Message with TLV was received |
| EnhancedEsmc_Val | std_logic | 1 | If ESMC Message with Enhanced TLV was received |

Table 9: Synce_NodeStaticStatusVal_Type

### 4.1.1.3 Entity Block Diagram

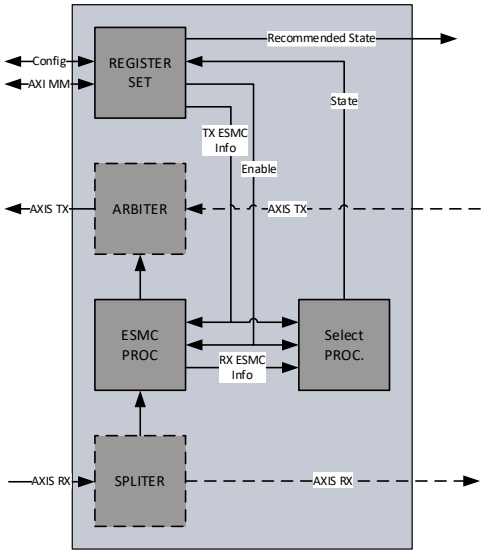


Figure 3: SyncE Node

### 4.1.1.4 Entity Description

#### ESMC Processor

This module handles all EMCS frames. It periodically sends ESMC messages with the configured parameters. It also sends a new ESMC message if the quality indication changes. It supports ESMC and Enhanced ESMC messages. It also receives ESMC frames, parses them and extract the ESMC and Enhanced ESMC Information. In addition, it has Timeout indication if for a defined time no ESMC messages were received.
See 4.2.1 for more details.

#### Select Processor

This module compares the local ESMC configuration parameters with the ESMC information received from the remote node. Based on this comparison, this module provides the recommended state.
See 4.2.2 for more details.

#### Registerset

This module is an AXI4Lite Memory Mapped Slave. It provides access to all registers and allows configuring the SyncE Node. It can be configured to either run in AXI or StaticConfig mode. If in StaticConfig mode, the configuration of the registers is done via signals and can be easily done from within the FPGA without CPU.

If in AXI mode, an AXI Master has to configure the registers with AXI writes to the registers, which is typically done by a CPU.
See 4.2.3 for more details.

#### Arbiter & Splitter

These are two optional modules used in the case when AXIS shall be passed through the Synce Node. It allows to Arbitrate the two TX AXIS streams and split the RX AXIS stream into two.

### 4.1.1.5 Entity Declaration

| Name | Dir | Type | Size | Description |
|---|---|---|---|---|
| Generics | | | | |
| General | | | | |
| ResetBuffer_Gen | - | boolean | 1 | To add an extra FF for Reset |
| StaticConfig_Gen | - | boolean | 1 | If Static Configuration or AXI is used |
| ClockClkPeriod Nanosecond_Gen | - | natural | 1 | Clock Period in Nanosecond: Default for 50 MHz = 20 ns |
| EnhancedEsmc Support_Gen | - | natural | 1 | Input delay of the SYNCE from the connector to the input signal. |
| EsmcRefresh_Gen | - | natural | 1 | The refresh rate in Milliseconds for ESMC Messages, standard 1s |
| EsmcTimeout_Gen | - | natural | 1 | The Timeout in Milliseconds for ESMC Messages, standard 5s |
| PassThrough_Gen | - | boolean | 1 | If AXI shall be passed through the core |

| | | | | |
|---|---|---|---|---|
| AxiAddressRange Low_Gen | - | std_logic_vector | 32 | AXI Base Address |
| AxiAddressRange High_Gen | | std_logic_vector | 32 | AXI Base Address plus Registerset Size Default plus 0xFFFF |
| Sim_Gen | | boolean | 1 | If in Testbench simulation mode: true = Simulation, false = Synthesis |
| **Ports** | | | | |
| System | | | | |
| SysClk_ClkIn | in | std_logic | 1 | System Clock |
| SysRstN_RstIn | in | std_logic | 1 | System Reset |
| Config | | | | |
| StaticConfig_DatIn | in | Synce_Node StaticConfig_Type | 1 | Static Configuration |
| StaticConfig_ValIn | in | Synce_Node StaticConfigVal _Type | 1 | Static Configuration valid |
| Status | | | | |
| StaticStatus_DatOut | out | Synce_Node StaticStatus_Type | 1 | Static Status |
| StaticStatus_ValOut | out | Synce_Node StaticStatusVal _Type | 1 | Static Status valid |
| Timer | | | | |
| Timer1ms_EvtIn | in | std_logic | 1 | Millisecond timer adjusted with the Clock |
| State Output | | | | |
| Recommended SynceState_DatOut | out | Clk_CoreState_Type | 1 | Recommended State |
| AXI4 Lite Slave | | | | |
| AxiWriteAddrValid _ValIn | in | std_logic | 1 | Write Address Valid |
| AxiWriteAddrReady _RdyOut | out | std_logic | 1 | Write Address Ready |
| AxiWriteAddrAddress _AdrIn | in | std_logic_vector | 32 | Write Address |
| AxiWriteAddrProt | in | std_logic_vector | 3 | Write Address |

| | | | | |
|---|---|---|---|---|
| _DatIn | | | | Protocol |
| AxiWriteDataValid_ValIn | in | std_logic | 1 | Write Data Valid |
| AxiWriteDataReady_RdyOut | out | std_logic | 1 | Write Data Ready |
| AxiWriteDataData_DatIn | in | std_logic_vector | 32 | Write Data |
| AxiWriteDataStrobe_DatIn | in | std_logic_vector | 4 | Write Data Strobe |
| AxiWriteRespValid_ValOut | out | std_logic | 1 | Write Response Valid |
| AxiWriteRespReady_RdyIn | in | std_logic | 1 | Write Response Ready |
| AxiWriteResp Response_DatOut | out | std_logic_vector | 2 | Write Response |
| AxiReadAddrValid_ValIn | in | std_logic | 1 | Read Address Valid |
| AxiReadAddrReady_RdyOut | out | std_logic | 1 | Read Address Ready |
| AxiReadAddrAddress_AdrIn | in | std_logic_vector | 32 | Read Address |
| AxiReadAddrProt_DatIn | in | std_logic_vector | 3 | Read Address Protocol |
| AxiReadDataValid_ValOut | out | std_logic | 1 | Read Data Valid |
| AxiReadDataReady_RdyIn | in | std_logic | 1 | Read Data Ready |
| AxiReadData Response_DatOut | out | std_logic_vector | 2 | Read Data |
| AxiReadDataData_DatOut | out | std_logic_vector | 32 | Read Data Response |
| (R)(G)Mii RX Clk/Rst Input | | | | |
| (R)(G)MiiRxClk_ClkIn | in | std_logic | 1 | RX Clock |
| (R)(G)MiiRxRstN_RstIn | in | std_logic | 1 | Reset aligned with RX Clock |
| (R)(G)Mii TX Clk/Rst Input | | | | |
| (R)(G)MiiTxClk_ClkIn | in | std_logic | 1 | TX Clock |
| (R)(G)MiiTxRstN_RstIn | in | std_logic | 1 | Reset aligned with TX Clock |
| (R)(G)Mii RX Data Input/Output | | | | |
| (R)(G)MiiRxDv_Ena | in/ out | std_logic | 1 | RX Data valid |
| (R)(G)MiiRxErr_Ena | in/ out | std_logic | 1 | RX Error |
| (R)(G)MiiRxData_Dat | in/ out | std_logic_vector | 2-8 | RX Data MII:4, RMII:2, GMII:8, |

| | | | | |
|---|---|---|---|---|
| | | | | RGMII:4 |
| (R)(G)MiiCol_Dat | in/ out | std_logic | 1 | Collision |
| (R)(G)MiiCrs_Dat | in/ out | std_logic | 1 | Carrier Sense |
| (R)(G)Mii TX Data Input/Output | | | | |
| (R)(G)MiiTxEn_Ena | in/ out | std_logic | 1 | TX Data valid |
| (R)(G)MiiTxErr_Ena | in/ out | std_logic | 1 | TX Error |
| (R)(G)MiiTxData_Dat | in/ out | std_logic_vector | 2-8 | TX Data<br>MII:4, RMII:2, GMII:8, RGMII:4 |

Table 10: SyncE Node

## 4.2 Design Parts

The SyncE Node core consists of a couple of subcores. Each of the subcores itself consist again of smaller function block. The following chapters describe these subcores and their functionality.

### 4.2.1 ESMC Processor

#### 4.2.1.1 Entity Block Diagram

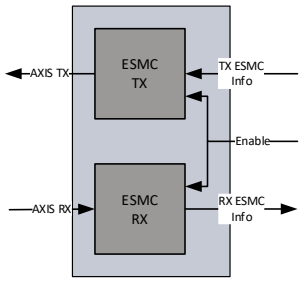


Figure 4: ESMC Processor

#### 4.2.1.2 Entity Description

**ESMC Receiver**

This module handles all incoming EMCS frames. It supports ESMC and Enhanced ESMC messages. It receives ESMC frames, parses them and extract the ESMC and Enhanced ESMC Information. In addition, it has Timeout indication if for a defined time no ESMC messages were received.

**ESMC Transmitter**

This module handles all outgoing EMCS frames. It periodically sends ESMC messages with the configured parameters. It also sends a new ESMC message if the quality indication changes, or the link goes up. It supports ESMC and Enhanced ESMC messages.

#### 4.2.1.3 Entity Declaration

| Name | Dir | Type | Size | Description |
|---|---|---|---|---|
| Generics | | | | |
| General | | | | |
| ClockClkPeriod | - | natural | 1 | Clock Period in |

| | | | | |
|---|---|---|---|---|
| Nanosecond_Gen | | | | Nanosecond |
| EnhancedEsmc Support_Gen | - | natural | 1 | Enhanced ESMC Support |
| EsmcRefresh_Gen | - | natural | 1 | The refresh rate in Milliseconds for ESMC Messages, standard 1s |
| EsmcTimeout_Gen | - | natural | 1 | The Timeout in Milliseconds for ESMC Messages, standard 5s |
| Sim_Gen | - | boolean | 1 | If in Testbench simulation mode |
| **Ports** | | | | |
| System | | | | |
| SysClk_ClkIn | in | std_logic | 1 | System Clock |
| SysRstN_RstIn | in | std_logic | 1 | System Reset |
| Delay Input | | | | |
| Timer1ms_EvtIn | in | std_logic | 1 | Millisecond timer adjusted with the Clock |
| Enable Input | | | | |
| Enable_EnaIn | in | std_logic | 1 | Enables the correc-tion and supervision |
| Config Input | | | | |
| OwnMac_DatIn | in | Common_Byte_Type | 6 | MAC Adress |
| EsmcConfig_DatIn | in | Synce_Esmc_Type | | Configuration |
| EsmcConfig_ValIn | in | Synce_Esmc Val_Type | | Configuration Valid |
| Status Output | | | | |
| EsmcTimeout_DatOut | out | std_logic | 1 | Timeout Indication |
| EsmcStatus_DatOut | out | Synce_Esmc_Type | | Status |
| EsmcStatus_ValOut | out | Synce_Esmc Val_Type | | Status Valid |
| Link Input | | | | |
| Link_DatIn | in | std_logic | 1 | If there is Link |
| LinkSpeed_DatIn | in | Common_LinkSpeed _Type | 1 | What Link Speed there is |
| SYNCE Cable Delay Input | | | | |
| AxisValid_ValIn | in | std_logic | 1 | AXI Stream RX |

| | | | | |
|---|---|---|---|---|
| AxisReady_ValOut | out | std_logic | 1 | frame input |
| AxisData_DatIn | in | std_logic_vector | 32 | |
| AxisStrobe_ValIn | in | std_logic_vector | 4 | |
| AxisKeep_ValIn | in | std_logic_vector | 4 | |
| AxisLast_ValIn | in | std_logic | 1 | |
| AxisUser_DatIn | in | std_logic_vector | 3 | |
| SYNCE Correction Input | | | | |
| AxisValid_ValOut | out | std_logic | 1 | AXI Stream TX |
| AxisReady_ValIn | in | std_logic | 1 | frame output |
| AxisData_DatOut | out | std_logic_vector | 32 | |
| AxisStrobe_ValOut | out | std_logic_vector | 4 | |
| AxisKeep_ValOut | out | std_logic_vector | 4 | |
| AxisLast_ValOut | | std_logic | 1 | |
| AxisUser_DatOut | out | std_logic_vector | 3 | |

Table 11: ESMC Processor

## 4.2.2 Select Processor

### 4.2.2.1 Entity Block Diagram

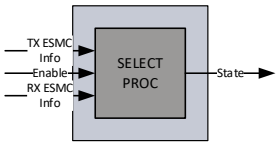


Figure 5: Select Processor

### 4.2.2.2 Entity Description

**Select Processor**

This module compares the local ESMC configuration parameters with the ESMC information received from the remote node. Based on this comparison, this module provides the recommended state.

The selection process is based on the QL indications represented by the SSM Code or combined SSM & ESSM Codes. The following are the Qality Levels according to ITU in decending order; not mentioned QL Levels get the priority 255 :

ITU Option Mode 1, SSM only:

| Name | SSM Code | ESSM Code | Priority |
|---|---|---|---|
| QL-PRC | 0010 | - | 2 |
| QL-SSU-A | 0100 | - | 3 |
| QL-SSU-B | 1000 | - | 4 |
| QL-SEC / EEC1 | 1011 | - | 6 |
| QL-DNU | 1111 | - | 7 |

Table 12: SSM to QL Encoding and Priority ITU Option 1

ITU Option Mode 2, SSM only:

| Name | SSM Code | ESSM Code | Priority |
|---|---|---|---|
| QL-PRS | 0001 | - | 2 |
| QL-STU | 0000 | - | 3 |
| QL-ST2 | 0111 | - | 4 |
| QL-TNC | 0100 | - | 5 |
| QL-ST3E | 1101 | - | 6 |

| | | | |
|---|---|---|---|
| QL-ST3 / EEC2 | 1010 | - | 7 |
| QL-SMC | 1100 | - | 8 |
| QL-PROV | 1110 | - | 10 |
| QL-DUS | 1111 | - | 11 |

Table 13: SSM to QL Encoding and Priority ITU Option 2

ITU Option Mode 3, SSM only:

| Name | SSM Code | ESSM Code | Priority |
|---|---|---|---|
| QL-UNK | 0000 | - | 2 |
| QL-SEC | 1011 | - | 4 |

Table 14: SSM to QL Encoding and Priority ITU Option 3

ITU Option Mode 1, SSM & ESSM:

| Name | SSM Code | ESSM Code | Priority |
|---|---|---|---|
| QL-ePRTC | 0010 | 0x21 | 0 |
| QL-PRTC | 0010 | 0x20 | 1 |
| QL-PRC | 0010 | 0xFF | 2 |
| QL-SSU-A | 0100 | 0xFF | 3 |
| QL-SSU-B | 1000 | 0xFF | 4 |
| QL-eEEC | 1011 | 0x22 | 5 |
| QL-SEC / EEC1 | 1011 | 0xFF | 6 |
| QL-DNU | 1111 | 0x07 | 7 |

Table 15: SSM & ESSM to QL Encoding and Priority ITU Option 1

ITU Option Mode 2, SSM & ESSM:

| Name | SSM Code | ESSM Code | Priority |
|---|---|---|---|
| QL-ePRTC | 0010 | 0x21 | 0 |
| QL-PRTC | 0010 | 0x20 | 1 |
| QL-PRS | 0001 | 0xFF | 2 |
| QL-STU | 0000 | 0xFF | 3 |
| QL-ST2 | 0111 | 0xFF | 4 |
| QL-TNC | 0100 | 0xFF | 5 |
| QL-ST3E | 1101 | 0xFF | 6 |
| QL-ST3 / EEC2 | 1010 | 0xFF | 7 |
| QL-SMC | 1100 | 0xFF | 8 |

| | | | |
|---|---|---|---|
| QL-PROV | 1110 | 0xFF | 10 |
| QL-DUS | 1111 | 0xFF | 11 |

Table 16: SSM & ESSM to QL Encoding and Priority ITU Option 2

ITU Option Mode 3, SSM & ESSM:

| Name | SSM Code | ESSM Code | Priority |
|---|---|---|---|
| QL-ePRTC | 0010 | 0x21 | 0 |
| QL-PRTC | 0010 | 0x20 | 1 |
| QL-UNK | 0000 | 0xFF | 2 |
| QL-eEEC | 1011 | 0x22 | 3 |
| QL-SEC | 1011 | 0xFF | 4 |

Table 17: SSM & ESSM to QL Encoding and Priority ITU Option 3

#### 4.2.2.3 Entity Declaration

| Name | Dir | Type | Size | Description |
|---|---|---|---|---|
| **Generics** | | | | |
| General | | | | |
| ClockClkPeriod Nanosecond_Gen | - | natural | 1 | Clock Period in Nanosecond |
| EnhancedEsmc Support_Gen | - | natural | 1 | Enhanced ESMC Support |
| EsmcRefresh_Gen | - | natural | 1 | The refresh rate in Milliseconds for ESMC Messages, standard 1s |
| EsmcTimeout_Gen | - | natural | 1 | The Timeout in Milliseconds for ESMC Messages, standard 5s |
| Sim_Gen | - | boolean | 1 | If in Testbench simulation mode |
| **Ports** | | | | |
| System | | | | |
| SysClk_ClkIn | in | std_logic | 1 | System Clock |
| SysRstN_RstIn | in | std_logic | 1 | System Reset |
| Timer | | | | |

| | | | | |
|---|---|---|---|---|
| Timer1ms_EvtIn | in | std_logic | 1 | Millisecond timer adjusted with the Clock |
| Link Input | | | | |
| Link_DatIn | in | std_logic | 1 | If there is Link |
| LinkSpeed_DatIn | in | Common_LinkSpeed _Type | 1 | What Link Speed there is |
| Config Input | | | | |
| EsmcConfig_DatIn | in | Synce_Esmc_Type | | Configuration |
| EsmcConfig_ValIn | in | Synce_Esmc Val_Type | | Configuration Valid |
| Status Input | | | | |
| EsmcTimeout_DatIn | in | std_logic | 1 | Timeout Indication |
| EsmcStatus_DatIn | in | Synce_Esmc_Type | | Status |
| EsmcStatus_ValIn | in | Synce_Esmc Val_Type | | Status Valid |
| Config Input | | | | |
| SelectConfig_DatIn | in | Synce_Select_Type | 1 | Selection Configura-tion |
| SYNCE Cable Delay Input | | | | |
| SelectStatus_DatOut | out | std_logic_vector | 16 | Selection Status |
| SelectStatus_ValOut | out | | | Selection Status Valid |

Table 18: Select Processor

## 4.2.3 Registerset

### 4.2.3.1 Entity Block Diagram

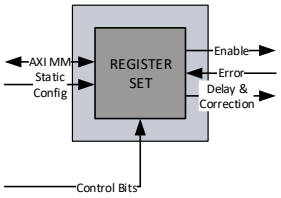


Figure 6: Registerset

### 4.2.3.2 Entity Description

#### Register Set

This module is an AXI4Lite Memory Mapped Slave. It provides access to all registers and allows configuring the SyncE Node. AXI4Lite only supports 32-bit wide data access, no byte enables, no burst, no simultaneous read and writes and no unaligned access. It can be configured to either run in AXI or StaticConfig mode. If in StaticConfig mode, the configuration of the registers is done via signals and can be easily done from within the FPGA without CPU. For each parameter a valid signal is available, the enable signal shall be set last (or simultaneously). To change configuration parameters the clock has to be disabled and enabled again, the cable delay value and correction can be changed at runtime. If in AXI mode, an AXI Master has to configure the registers with AXI writes to the registers, which is typically done by a CPU. Parameters can in this case also be changed at runtime.

### 4.2.3.3 Entity Declaration

| Name | Dir | Type | Size | Description |
|---|---|---|---|---|
| Generics | | | | |
| Register Set | | | | |
| StaticConfig_Gen | - | boolean | 1 | If Static Configuration or AXI is used |
| EnhancedEsmcSupport_Gen | - | natural | 1 | Enhanced ESMC Support |
| AxiAddressRangeLow_Gen | - | std_logic_vector | 32 | AXI Base Address |

| | | | | |
|---|---|---|---|---|
| AxiAddressRange High_Gen | - | std_logic_vector | 32 | AXI Base Address plus Registerset Size |
| **Ports** | | | | |
| System | | | | |
| SysClk_ClkIn | in | std_logic | 1 | System Clock |
| SysRstN_RstIn | in | std_logic | 1 | System Reset |
| Config | | | | |
| StaticConfig_DatIn | in | Synce_Node StaticConfig_Type | 1 | Static Configuration |
| StaticConfig_ValIn | in | Synce_Node StaticConfigVal _Type | 1 | Static Configuration valid |
| Status | | | | |
| StaticStatus_DatOut | out | Synce_Node StaticStatus_Type | 1 | Static Status |
| StaticStatus_ValOut | out | Synce_Node StaticStatusVal _Type | 1 | Static Status valid |
| AXI4 Lite Slave | | | | |
| AxiWriteAddrValid _ValIn | in | std_logic | 1 | Write Address Valid |
| AxiWriteAddrReady _RdyOut | out | std_logic | 1 | Write Address Ready |
| AxiWriteAddrAddress _AdrIn | in | std_logic_vector | 32 | Write Address |
| AxiWriteAddrProt _DatIn | in | std_logic_vector | 3 | Write Address Protocol |
| AxiWriteDataValid _ValIn | in | std_logic | 1 | Write Data Valid |
| AxiWriteDataReady _RdyOut | out | std_logic | 1 | Write Data Ready |
| AxiWriteDataData _DatIn | in | std_logic_vector | 32 | Write Data |
| AxiWriteDataStrobe _DatIn | in | std_logic_vector | 4 | Write Data Strobe |
| AxiWriteRespValid _ValOut | out | std_logic | 1 | Write Response Valid |
| AxiWriteRespReady _RdyIn | in | std_logic | 1 | Write Response Ready |
| AxiWriteResp Response_DatOut | out | std_logic_vector | 2 | Write Response |
| AxiReadAddrValid _ValIn | in | std_logic | 1 | Read Address Valid |

| | | | | |
|---|---|---|---|---|
| AxiReadAddrReady_RdyOut | out | std_logic | 1 | Read Address Ready |
| AxiReadAddrAddress_AdrIn | in | std_logic_vector | 32 | Read Address |
| AxiReadAddrProt_DatIn | in | std_logic_vector | 3 | Read Address Protocol |
| AxiReadDataValid_ValOut | out | std_logic | 1 | Read Data Valid |
| AxiReadDataReady_RdyIn | in | std_logic | 1 | Read Data Ready |
| AxiReadData Response_DatOut | out | std_logic_vector | 2 | Read Data |
| AxiReadDataData_DatOut | out | std_logic_vector | 32 | Read Data Re-sponse |
| State Output | | | | |
| RecommendedSyn-ceState_DatOut | out | Clk_CoreState_Type | 1 | Additional correc-tion to the received UTC time |
| Config Output | | | | |
| OwnMac_DatIn | out | Common_Byte_Type | 6 | MAC Adress |
| EsmcConfig_DatIn | out | Synce_Esmc_Type | | Configuration |
| EsmcConfig_ValIn | out | Synce_Esmc Val_Type | | Configuration Valid |
| SelectConfig_DatOut | out | Synce_Select_Type | 6 | Config of Selection |
| Status Input | | | | |
| EsmcTimeout_DatIn | in | std_logic | 1 | ESMC Timeout |
| EsmcStatus_DatIn | in | Synce_Esmc_Type | | ESMC Status |
| EsmcStatus_ValIn | in | Synce_Esmc Val_Type | | ESMC Status Valid |
| SelectStatus_DatIn | in | Synce_Select_Type | | Select Status |
| SelectStatus_ValIn | in | std_logic | | Select Status Valid |
| Enable Output | | | | |
| SynceNode Enable_DatOut | out | std_logic | 1 | Enable |

Table 19: Registerset

## 4.2.4 Ethernet Interface Adapter

### 4.2.4.1 Entity Block Diagram

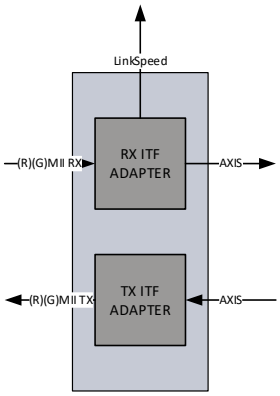


Figure 7: Ethernet Interface Adapter

### 4.2.4.2 Entity Description

#### RX Interface Adapter

This module convert the Media Independent Interface (R)(G)MII data stream (2/4/8bit) into a 32bit AXI stream. First bytes on the cable are mapped to the AXI MSB of the data array. It contains an asynchronous Fifo to on one hand do clock domain crossing from the external clock to the system clock and on the other hand also to minimal buffer data for speed differences. The Fifo size is kept quite small to assure correct timestamp alignment with the frame. It converts the different data widths into a 32bit block AXI stream. The Preamble and SFD are removed on reception. Also, it detects the link speed based on the interface clock.

#### TX Interface Adapter

This module convert the 32bit AXI stream into a Media Independent Interface (R)(G)MII data stream (2/4/8bit) which is continuous. The MSB of the AXI data array is mapped to the first byte on the cable. It contains an asynchronous Fifo to on one hand do clock domain crossing from the system clock to the external clock and on the other hand also to minimal buffer data for speed differences. The Fifo size is kept quite small to assure correct timestamp alignment with the frame. It converts the 32bit block AXI stream into the different data widths. The Preamble and SFD are added before transmission. It also assures the correct interframe gap between frames.

#### 4.2.4.3 Entity Declaration

| Name | Dir | Type | Size | Description |
|---|---|---|---|---|
| **Generics** | | | | |
| Interface Adapter | | | | |
| ClockClkPeriodNano-second_Gen | - | natural | 1 | Integer Clock Period |
| IoFf_Gen | - | boolean | 1 | Shall IO flip flops be instantiated |
| **Ports** | | | | |
| System | | | | |
| SysClk_ClkIn | in | std_logic | 1 | System Clock |
| SysRstN_RstIn | in | std_logic | 1 | System Reset |
| (R)(G)Mii RX Clk/Rst Input | | | | |
| (R)(G)MiiRxClk_ClkIn | in | std_logic | 1 | RX Clock |
| (R)(G)MiiRxRstN_RstIn | in | std_logic | 1 | Reset aligned with RX Clock |
| (R)(G)Mii TX Clk/Rst Input | | | | |
| (R)(G)MiiTxClk_ClkIn | in | std_logic | 1 | TX Clock |
| (R)(G)MiiTxRstN_RstIn | in | std_logic | 1 | Reset aligned with TX Clock |
| (R)(G)Mii RX Data Input/Output | | | | |
| (R)(G)MiiRxDv_Ena | In/ out | std_logic | 1 | RX Data valid |
| (R)(G)MiiRxErr_Ena | In/ out | std_logic | 1 | RX Error |
| (R)(G)MiiRxData_Dat | In/ out | std_logic_vector | 2-8 | RX Data MII:4, RMII:2, GMII:8, RGMII:4 |
| (R)(G)MiiCol_Dat | In/ out | std_logic | 1 | Collision |
| (R)(G)MiiCrs_Dat | In/ out | std_logic | 1 | Carrier Sense |
| (R)(G)Mii TX Data Input | | | | |
| (R)(G)MiiTxEn_Ena | In/ out | std_logic | 1 | TX Data valid |
| (R)(G)MiiTxErr_Ena | In/ out | std_logic | 1 | TX Error |
| (R)(G)MiiTxData_Dat | In/ | std_logic_vector | 2-8 | TX Data |

| | | | | |
|---|---|---|---|---|
| | out | | | MII:4, RMII:2, GMII:8, RGMII:4 |
| Link Speed Output | | | | |
| LinkSpeed_DatOut | out | Common_ LinkSpeed_Type | 1 | Link Speed of the interface |
| Axi Input | | | | |
| AxisValid_ValIn | in | std_logic | 1 | AXI Stream frame input |
| AxisReady_ValOut | out | std_logic | 1 | |
| AxisData_DatIn | in | std_logic_vector | 32 | |
| AxisStrobe_ValIn | in | std_logic_vector | 4 | |
| AxisKeep_ValIn | in | std_logic_vector | 4 | |
| AxisLast_ValIn | in | std_logic | 1 | |
| AxisUser_DatIn | in | std_logic_vector | 3 | |
| Axi Output | | | | |
| AxisValid_ValOut | out | std_logic | 1 | AXI Stream frame output |
| AxisReady_ValIn | in | std_logic | 1 | |
| AxisData_DatOut | out | std_logic_vector | 32 | |
| AxisStrobe_ValOut | out | std_logic_vector | 4 | |
| AxisKeep_ValOut | out | std_logic_vector | 4 | |
| AxisLast_ValOut | out | std_logic | 1 | |
| AxisUser_DatOut | out | std_logic_vector | 3 | |

Table 20: Ethernet Interface Adapter

## 4.3 Configuration example

In both cases the enabling of the core shall be done last, after or together with the configuration.

### 4.3.1 Static Configuration

```
constant SynceStaticConfigSlave_Con : Synce_NodeStaticConfig_Type := (
  OwnMac                                        => (
      0                                         => x"00",
      1                                         => x"11",
      2                                         => x"22",
      3                                         => x"33",
      4                                         => x"44",
      5                                         => x"55"),
  Mode                                          => Mode1_E, -- Option 1 Network
  EsmcSsmCode                                   => x"3",
  EnhancedEsmcSsmCode                           => x"4E",
  EnhancedEsmcFlags                             => x"00",,
  EnhancedEsmcClockIdentity                     => (
      0                                         => x"00",
      1                                         => x"11",
      2                                         => x"22",
      3                                         => x"FF",
      4                                         => x"FE",
      5                                         => x"33",
      6                                         => x"44",
      7                                         => x"55"),
  EnhancedEsmcCascadedNrOfEec                   => x"12",
  EnhancedEsmcCascadedNrOfEnhancedEec           => x"34"
);
```

Figure 8: Static Configuration

The cable delay can be changed at runtime. It is always valid.

### 4.3.2 AXI Configuration

The following code is a simplified pseudocode from the testbench: The base address of the SyncE Node is 0x10000000.

```
-- SRC MAC 00:11:22:33:44:55
AXI AXI0 WRITE 20000104 33221100
AXI AXI0 WRITE 20000108 00005544
AXI AXI0 WRITE 20000100 00000001

-- Check Status Unknown
```

```
AXI AXI0 READ 20000004 00000000 EQUAL

-- TxEsmcSsmCode
AXI AXI0 WRITE 20000010 00000004
-- TxEnhancedEsmcSsmCodeFlags
AXI AXI0 WRITE 20000020 00000020
-- TxEnhancedEsmclockId1
AXI AXI0 WRITE 20000024 FF221100
-- TxEnhancedEsmclockId2
AXI AXI0 WRITE 20000028 554433FE
-- TxEnhancedCascadedNrofEec
AXI AXI0 WRITE 2000002C 00120034

-- Enable, Esmc, Mode 1
AXI AXI0 WRITE 20000000 00011001
```

Figure 9: AXI Configuration

## 4.4 Clocking and Reset Concept

### 4.4.1 Clocking

To keep the design as robust and simple as possible, the whole SyncE Node, including the Counter Clock and all other cores from NetTimeLogic are run in one clock domain. This is considered to be the system clock. Per default this clock is 50MHz. Where possible also the interfaces are run synchronous to this clock. For clock domain crossing asynchronous FIFOs with gray counters or message patterns with meta-stability flip-flops are used. Clock domain crossings for the AXI interface is moved from the AXI slave to the AXI interconnect.

| Clock | Frequency | Description |
|---|---|---|
| System | | |
| System Clock | 50MHz (Default) | System clock where the PS runs on as well as the counter clock etc. |
| AXI Interface | | |
| AXI Clock | 50MHz (Default) | Internal AXI bus clock, same as the system clock |

Table 21: Clocks

### 4.4.2Reset

In connection with the clocks, there is a reset signal for each clock domain. All resets are active low. All resets can be asynchronously set and shall be synchronously released with the corresponding clock domain. All resets shall be asserted for the first couple (around 8) clock cycles. All resets shall be set simultaneously and released simultaneously to avoid overflow conditions in the core. See the reference designs top file for an example of how the reset shall be handled.

| Reset | Polarity | Description |
|---|---|---|
| System | | |
| System Reset | Active low | Asynchronous set, synchronous release with the system clock |
| AXI Interface | | |
| AXI Reset | Active low | Asynchronous set, synchronous release with the AXI clock, which is the same as |

| | | |
|---|---|---|
| | | the system clock |

Table 22: Resets

# 5 Resource Usage

Since the FPGA Architecture between vendors and FPGA families differ there is a split up into the two major FPGA vendors.

## 5.1 Intel/Altera (Cyclone V)

| Configuration | FFs | LUTs | BRAMs | DSPs |
|---|---|---|---|---|
| Minimal (Static configuration and only ESMC) | 602 | 725 | 18 | 0 |
| Maximal (AXI configuration and Enhanced ESMC) | 1056 | 1227 | 18 | 0 |

Table 23: Resource Usage Intel/Altera

## 5.2 AMD/Xilinx (Artix 7)

| Configuration | FFs | LUTs | BRAMs | DSPs |
|---|---|---|---|---|
| Minimal (Static configuration and only ESMC) | 593 | 827 | 5 | 0 |
| Maximal (AXI configuration and Enhanced ESMC) | 1040 | 1290 | 5 | 0 |

Table 24: Resource Usage AMD/Xilinx

# 6 Delivery Structure

```
AXI                                 -- AXI library folder
 |-Library                          -- AXI library component sources
 |-Package                          -- AXI library package sources

CLK                                 -- CLK library folder
 |-Library                          -- CLK library component sources
 |-Package                          -- CLK library package sources

COMMON                              -- COMMON library folder
 |-Library                          -- COMMON library component sources
 |-Package                          -- COMMON library package sources

SYNCE                               -- SYNCE library folder
|-Core                              -- SYNCE library cores
 |-Doc                              -- SYNCE library cores documentations
 |-Library                          -- SYNCE library component sources
 |-Package                          -- SYNCE library package sources
 |-Refdesign                        -- SYNCE library cores reference designs
 |-Testbench                        -- SYNCE library cores testbench sources and sim/log

SIM                                 -- SIM library folder
 |-Doc                              -- SIM library command documentation
 |-Package                          -- SIM library package sources
 |-Testbench                        -- SIM library testbench template sources
 |-Tools                            -- SIM simulation tools
```

# 7 Reference Designs

The PTP Ordinary Clock reference design contains a PLL to generate all necessary clocks (cores are run at 50 MHz), an instance of the SyncE Node IP core and PHY Config module and optionally an instance of the Adjustable Counter Clock IP core and PPS Slave (needs to be purchased separately) connected to the PPS Generator based on the MII Clock and SyncE State. The Reference Design is intended to be connected to any SyncE Master or Slave device. The Reference Design is using MII in 100Mbit full duplex as Ethernet link.
All generics can be adapted to the specific needs.

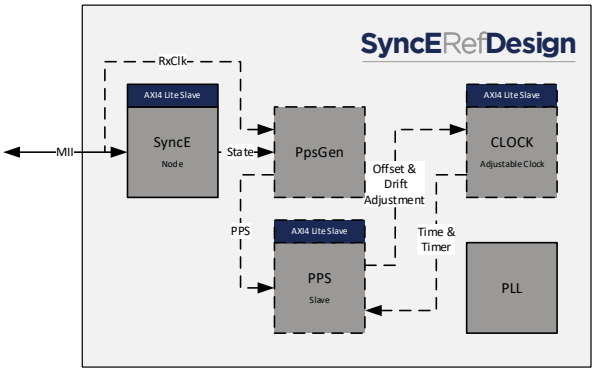


Figure 10: Reference Design

## 7.1 AMD/Xilinx: Digilent Arty

The Arty board is an FPGA board from Digilent Inc. with an Artix7 FPGA from AMD/Xilinx. (http://store.digilentinc.com/arty-board-artix-7-fpga-development-board-for-makers-and-hobbyists/)

1. Open Vivado 2019.1.
   Note: If a different Vivado version is used, see chapter 7.2.
2. Run TCL script /PTP/Refdesign/Xilinx/ArtyA7/SynceNodeMii/SynceNode.tcl
   a. This has to be run only the first time and will create a new Vivado Project
3. If the project has been created before open the project and do not rerun the project TCL
4. If the optional cores PPS Slave Clock, Adjustable Counter Clock is available add the files from the corresponding folders (PPS/Core, PPS/Library and PPS/Package & CLK/Core, CLK/Library and CLK/Package) to the corresponding Library (PpsLib & ClkLib).

5. Change the generics (FrequencySync_Gen) in Vivado (in the settings menu, not in VHDL) to true for the optional cores that are available.
6. Rerun implementation
7. Download to FPGA via JTAG

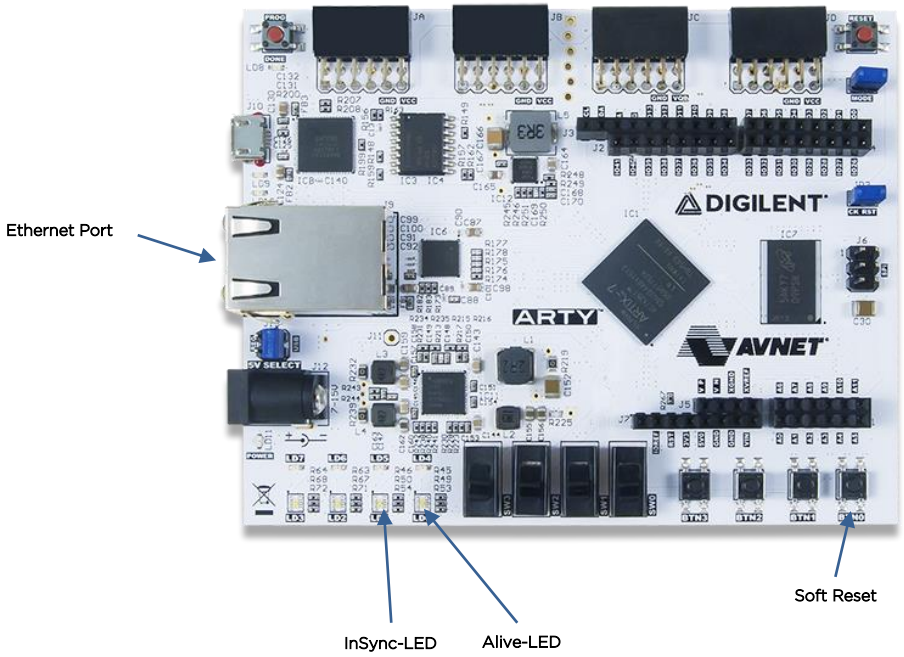


Figure 11: Arty (source Digilent Inc)

## 7.2 AMD/Xilinx: Vivado version

The provided TCL script for creation of the reference-design project is targeting AMD/Xilinx Vivado 2019.1.
If a lower Vivado version is used, it is recommended to upgrade to Vivado 2019.1 or higher.
If a higher Vivado version is used, the following steps are recommended:

- Before executing the project creation TCL script, the script's references of Vivado 2019 should be manually replaced to the current Vivado version. For example, if version Vivado 2022 is used, then:
  - The statement occurrences:

    ```
    set_property flow "Vivado Synthesis 2019" $obj
    ```

    shall be replaced by:

    ```
    set_property flow "Vivado Synthesis 2022 $obj
    ```

  - The statement occurrences:

    ```
    set_property flow "Vivado Implementation 2019" $obj
    ```

shall be replaced by:

```
set_property flow "Vivado Implementation 2022" $obj
```

- After executing the project creation TCL script, the AMD/Xilinx IP cores, such as the Clocking Wizard core, might be locked and a version upgrade might be required. To do so:
  1. At "Reports" menu, select "Report IP Status".
  2. At the opened "IP Status" window, select "Upgrade Selected". The tool will upgrade the version of the selected IP cores.

# 8 Testbench

The SyncE Node testbench consist of 2 parse/port types: AXI and ETH.
The ETH port allows to send, receive and check ethernet frames. It is used to generate the ESMC messages. In addition, for configuration and result checks an AXI read and write port is used in the testbench.

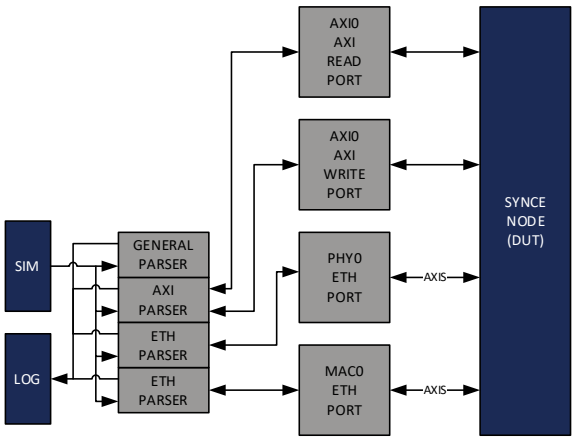


Figure 12: Testbench Framework

For more information on the testbench framework check the Sim_ReferenceManual documentation.

With the Sim parameter set the time base for timeouts are divided by 1000 to 100000 to speed up simulation time.

## 8.1 Run Testbench

1. Run the general script first

```
source XXX/SIM/Tools/source_with_args.tcl
```

2. Start the testbench with all test cases

```
src XXX/SYNCE/Testbench/Core/SynceNode/Script/run_Synce_Node_Tb.tcl
```

3. Check the log file LogFile1.txt in the XXX/SYNCE/Testbench/Core/SynceNode/Log/ folder for simulation results.

# A List of tables



# B List of figures