TMC8460-BI

Integrated EtherCAT Slave Controller with Enhanced Functionality

TRINAMIC® Motion Control GmbH & Co. KG
Hamburg, GERMANY
www.trinamic.com

The TMC8460 is an EtherCAT Slave Controller (ESC) used for EtherCAT communication. It provides the interface for data exchange between EtherCAT master and the slave's local application controller.

TMC8460 also provides a large set of complex real-time IO features in hardware with focus on motor and motion control applications and systems.

Focus

- Easiest to use ESC / Simplicity / Industrial
- License-free / requirements for customer
- Robust / Availability / Flexibility
- Automation / Drives / Robotics / Semiconductor / Multi Axis Systems / Motor & Motion Control

Features

- Standard compliant EtherCAT Slave Controller register set with 2 MII ports, 6 FMMU, 6 Sync Managers, Distributed clocks (64 bit), 16 Kbyte ESC RAM size
- External I²C EEPROM
- SPI Process Data Interface (PDI) with up to 30Mbit/s
- SPI interface for Trinamic Multi-Function and Control IO Block (MFCIO) with up to 30Mbit/s
- Optional Device Emulation mode
- Trinamic Multi-Function and Control IO Block (MFCIO)
    - 8 Digital general purpose IO, individually configurable
    - Incremental encoder input (ABN), single ended
    - Step & direction output with internal step rate generator
    - 3-ch PWM block with configurable frequency, duty cycle, dead times
    - Generic SPI master interface with up to 4 slaves, e.g., to connect Trinamic ICs
    - Configurable IRQ and event signal
    - Configurable watchdog for outputs and inputs
    - Simple configuration via EEPROM or from MCU
- 16MHz CLK output, e.g., for MCU or Trinamic ICs or other peripherals
- Operating voltages: 3V3 and 1V2
- Industrial temperature range: -40°C to +100°C
- Package: VFGG400, 17mmx17mm Very Fine Pitch Ball Grid Array, 0.8mm pitch

Table of Contents

List of Figures

List of Tables

1 Abbreviations

AL	
BOOT	Special boot state of the EtherCAT state machine
CS	Chip Select (SPI bus signal)
ECAT	EtherCAT (or sometimes used for EtherCAT Master Interface)
EEPROM	Electrically Erasable Programmable Read Only Memory. Non-volatile memory used to store EtherCAT Slave Information (ESI). Connected to the SII
ENI	EtherCAT Network Information file, holds information on the complete EtherCAT bus structure and its connected slaves
EoF	End of Frame
ESC	EtherCAT Slave Controller
ESI	EtherCAT Slave Information, stored in SII EEPROM, holds slave specific configuration information
ETG	EtherCAT Technology Group, www.ethercat.org
EtherCAT	Ethernet in Control and Automation Technology
FCS	Frame Check Sequence
FMMU	Fieldbus Memory Management Unit
GPI	General Purpose Input(s)
GPO	General Purpose Output(s)
I2C	Inter-Integrated Circuit, serial bus used for SII EEPROM connection
INIT	Initial state of the EtherCAT state machine
IRQ	Interrupt Request
MAC	Media Access Control layer
MCU	Microcontroller Unit
MFCIO	Multi Function and Control Input Output
MI	(PHY) Management Interface
MII	Media Independent Interface: Standardized interface between the Ethernet MAC and PHY
OP	Operational state of the EtherCAT state machine
PDI	Process Data Interface or Physical Device Interface: an interface that allows access to ESC from the process side
PDO	Process Data Object
PHY	Physical layer device that converts data from the Ethernet controller to electric or optical signals
PREOP	Pre-operational state of the EtherCAT state machine
PWM	Pulse Width Modulation
RAM	Random Access Memory. ESC have User RAM and Process Data RAM
RX	Receive path
SAFEOP	Safe operational state of the EtherCAT state machine
SII	EtherCAT Slave Information Interface
SM	SyncManager
SoF	Start of Frame
SPI	Serial Peripheral Interface
TX	Transmit path
µC	
XML	Extensible Markup Language: Standardized definition language that can be interpreted by nearly all parsers.
S/D	Step and Direction interface
PDRAM	Process Data RAM of the ESC
MBx	Memory Block x
IEC	International Electrotechnical Commission
ESM	EtherCAT State Machine

2 Principles of Operation

2.1 Key Concepts

2.1.1 General Information on EtherCAT

EtherCAT (Ethernet in Control and Automation Technology) has been developed and patented by Beckhoff. It is an Ethernet-based technology for data transmission and application control in real time. All process data for all connected nodes are transmitted in a single frame. All nodes connected to the bus interpret, process, and modify their data „on the fly". Ethernet frames are not buffered inside a node but are directly forwarded with minimum additional delay. Maintaining a strict master-slave communication, data exchange utilizes mailbox mechanisms and PDOs (Process Data Objects).

To ensure real-time behavior, frame processing and forwarding requires special hardware.
This special hardware is called ESC (EtherCAT Slave Controller) like the Trinamic TMC8460, TMC8461, or TMC8462.

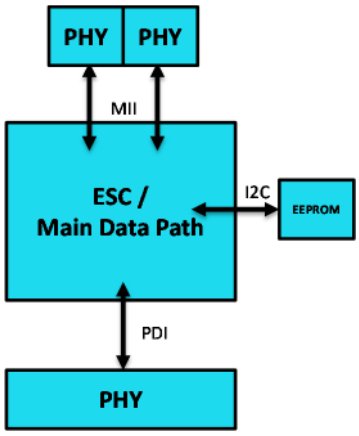
EtherCAT does not require software interaction for data transmission inside the slaves. EtherCAT only defines the MAC layer (similar to CAN). Higher layer protocols are implemented in software on microcontrollers connected to the ESC.

The ETG takes care for standardization activities and conformance testing. EtherCAT is integrated into the following major standards: IEC 61158 (protocols and services), IEC 61784-2 (communication profiles for devices), IEC 61800-7 (drive profiles and communication), SEMI standard E54.20 (since 2007).

For detailed information on the EtherCAT technology, the EtherCAT core mechanisms, and major features we refer to the official standard documentations and guidelines available from ETG (www.ethercat.org, ETG.1000), IEC (http://www.iec.ch, see numbers above), and Beckhoff (http://www.beckhoff.de, technical specification).

2.1.2 EtherCAT Slave Controller (ESC)

The TMC8460 is a standard-conform dedicated EtherCAT Slave Controller providing EtherCAT MAC layer functionality to EtherCAT slaves.
It connects via standard of-the-shelf Ethernet PHYs to the physical medium and provides a digital control interface to a local application controller while also providing the option for standalone operation.

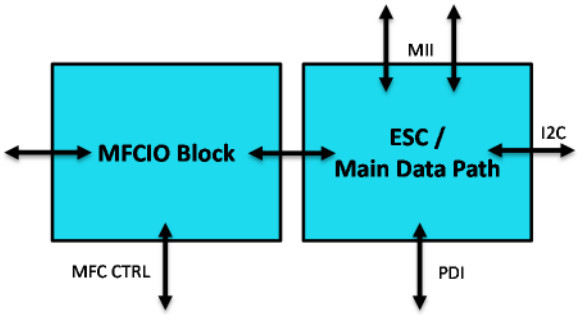


MAJOR ETHERCAT FEATURES

- Ethernet PHY interface: 2 x MII
- 6 FMMUs & 6 Sync Managers
- 16 Kbyte Process Data RAM
- 64 bit Distributed Clocks
- I²C interface for external EEPROM
- SPI Process Data Interface (PDI) with up to 30Mbit/s / optional Device Emulation mode

2.1.3 Trinamic Multi-Function and Control IO Block

Besides the proven EtherCAT functionality and the main EtherCAT data path, TMC8460 comes with a dedicated hardware block providing a configurable set of complex real-time IO functions to smart embedded systems. This IO functionality is called Multi-Function Control and IO block – MFCIO. Its special focus is on motor and motion control applications and systems while it is not limited to this application area.
The MFCIO block combines various functional sub-blocks that are helpful in an embedded design to reduce complexity, simplify the bill of materials (BOM), and to provide hardware acceleration to compute intensive tasks or time critical tasks. These functions can be used from the local application controller using a dedicated SPI interface or can directly be mapped into the Process Data RAM for direct access by the EtherCAT master.



General Purpose IOs

- There are up to 8 outputs or up to 8 inputs
- Each IO is individually configurable

Incremental Encoder Unit

- Incremental encoder inputs (ABN) with configurable counting constant, polarity, N-signal behavior and latch on N-signal
- 32 bit count register

Step & Direction Unit

- Simple internal step rate generator
- Configurable step pulse width and polarity
- Continuous mode or one-shot mode with configurable step number
- Counter for steps that have been done

3-ch PWM

- configurable frequency, duty cycle, polarity, dead times, polarity per channel

SPI Master Interface

- To directly connect to a TMC driver/controller or other SPI slaves
- Up to 4 slaves
- Configurable speed, mode, datagram width up to 64 bits (longer datagrams are possible)

IRQ / Event Output

- Common IRQ signal to indicate various events triggered by the MFCIO block
- Mask register to enable/disable certain event triggers

WATCHDOG

- Configurable for all inputs and outputs
- Outputs will be assigned with configurable level @ watchdog event
- Inputs will trigger a watchdog event only
- ECAT SoF and PDI SPI Chip Select can be monitored with watchdog as well

EMERGENCY SWITCH INPUT

- If used all functional outputs are set to a configurable safe state when the switch is not actively driven high
- Low active: must be pulled high for normal operation if used.

2.2 Configuration Options

The TMC8460 must be configured after power-up for proper operation. The EtherCAT part is automatically configured using configuration data from the connected I2C EEPROM.
The MFCIO block can also be configured using EEPROM configuration data. The EEPROM must therefore contain additional configuration data with category '1', which is automatically copied to ESC configuration RAM at addresses 0x0580:0x05FF.
Another way to configure the MFCIO block is to directly write the configuration bits to this RAM area using the ECAT or the PDI interface.
The MFCIO block can be used directly by a local application controller independent of the EtherCAT data path. Therefore, no upfront configuration is required since the MFCIO blocks comes with a dedicated SPI interface allowing complete access to its functions and local register set.

2.3 Control Interfaces

2.3.1 Ethernet Interface

For connection to the Ethernet physical medium and to the EtherCAT master, the TMC8460-BI offers two MII ports (media independent interface) and connects to standard 100Mbit/s Ethernet PHYs or 1Gbit/s Ethernet PHYs running in 100Mbit/s mode.

2.3.2 Process Data Interface

The Process Data Interface (PDI) is an SPI interface. The TMC8460-BI provides an SPI slave interface with ca. 30Mbit/s to connect to a local MCU or application controller. Typically, the local application controller contains the EtherCAT slave stack to control the EtherCAT state machine and to process state change requests by the EtherCAT master. Pulling the external configuration pin PDI_EMULATION high switches to Device Emulation mode. In Device Emulation mode, the TMC8460-BI can be used in standalone mode without MCU since state change requests by the master are directly forwarded to the state registers inside the TMC8460's ESC part. The PDI SPI interface remains active and can be used by an MCU in device emulation state.

2.3.3 Multi-Function and Control IO Block Interface

The MFCIO block of the TMC8460-BI comes with a dedicated SPI slave interface to allow direct access from a local application controller. It is called MFC CTRL SPI interface. This interface to the MFCIO block's functions is always available even if the EtherCAT state machine is currently not in operational state (OP). Protocol structure and timing are identical to the PDI SPI.

2.3.4 SPI Bus Sharing

Both SPI interfaces – PDI SPI and MFC CTRL SPI – can share the same SPI bus signals using two chip select signals. This reduces overall number of signals on the PCB and requires only one SPI interface on the local MCU. The external configuration pin SHARED_SPI_BUS needs to be pulled high for bus sharing. In this case, the PDI SPI bus is used as shared bus interface together with the chip select line of the MFC CTRL SPI interface.

2.3.5 Configuration Inputs

External package pins allow for selection of configuration options that typically do not change during operation by directly connecting them to 3V3 or ground. These package pins can also be controlled by GPIOs of the local application controller.
These options are for example external EEPROM's size, PHY addressing and MII TX clock shift configuration, polarities of the PHYs' link indicator, device emulation mode and enabling of the 16MHz clock output.

2.3.6 EEPROM Interface

An EEPROM containing boot-up configuration data is required for ESC operation. The EEPROM must come with a standard I2C interface and connects to the PROM interface of the TMC8460. EEPROMs of different size can be used. There is a difference in the I2C protocol when EEPROM parts with >16kbits memory size are used.

2.4 Software View

As seen from an EtherCAT master system, the TMC8460-BI is part of an EtherCAT slave using the register set and functionality according to the EtherCAT standard. It works together with other EtherCAT slaves on the same bus and is accessible via the Ethernet physical medium. According to the slave configuration (ESI) and network configuration (ENI)

As seen from the local application controller, the TMC8460-BI is a peripheral SPI slave device with a number of control and status registers that are accessible using the PDI SPI interface for the main EtherCAT data path or the MFC CTRL SPI interface for the MFCIO block.
For proper EtherCAT functionality, the local application controller runs the EtherCAT slave stack to process master requests regarding state changes in the state machine for example using the PDI SPI interface. In device emulation mode the PDI SPI interface is not used since master requests are handled inside the ESC.

The MFCIO block functions can directly be used with the MFC CTRL SPI interface. This can be done even without using the EtherCAT slave controller part.

Trinamic's TMCL IDE (http://www.trinamic.com/software-tools/tmcl-ide) can be used to access the device with the TMC8460-BI evaluation board. All registers are accessible via the two SPI interfaces. The configuration memory area for the MFCIO block can be read and modified.

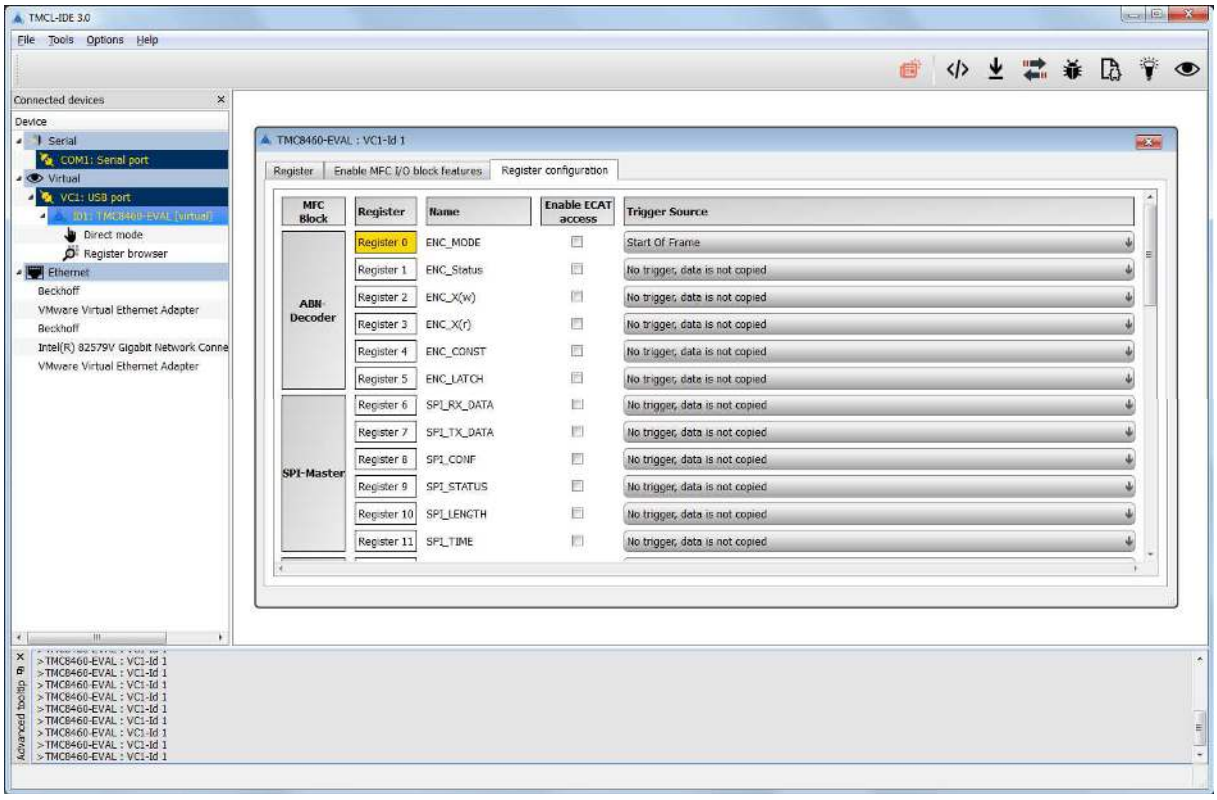


Figure 1 - TMCL-IDE with direct register access to the TMC8460-BI on its evaluation board

A wizard helps and simplifies the configuration and setup of the TMC8460-BI to your specific needs and provides code examples for your configuration to be used inside you microcontroller firmware and the EEPROM for startup configuration.

Figure 2 - Wizard Start Screen

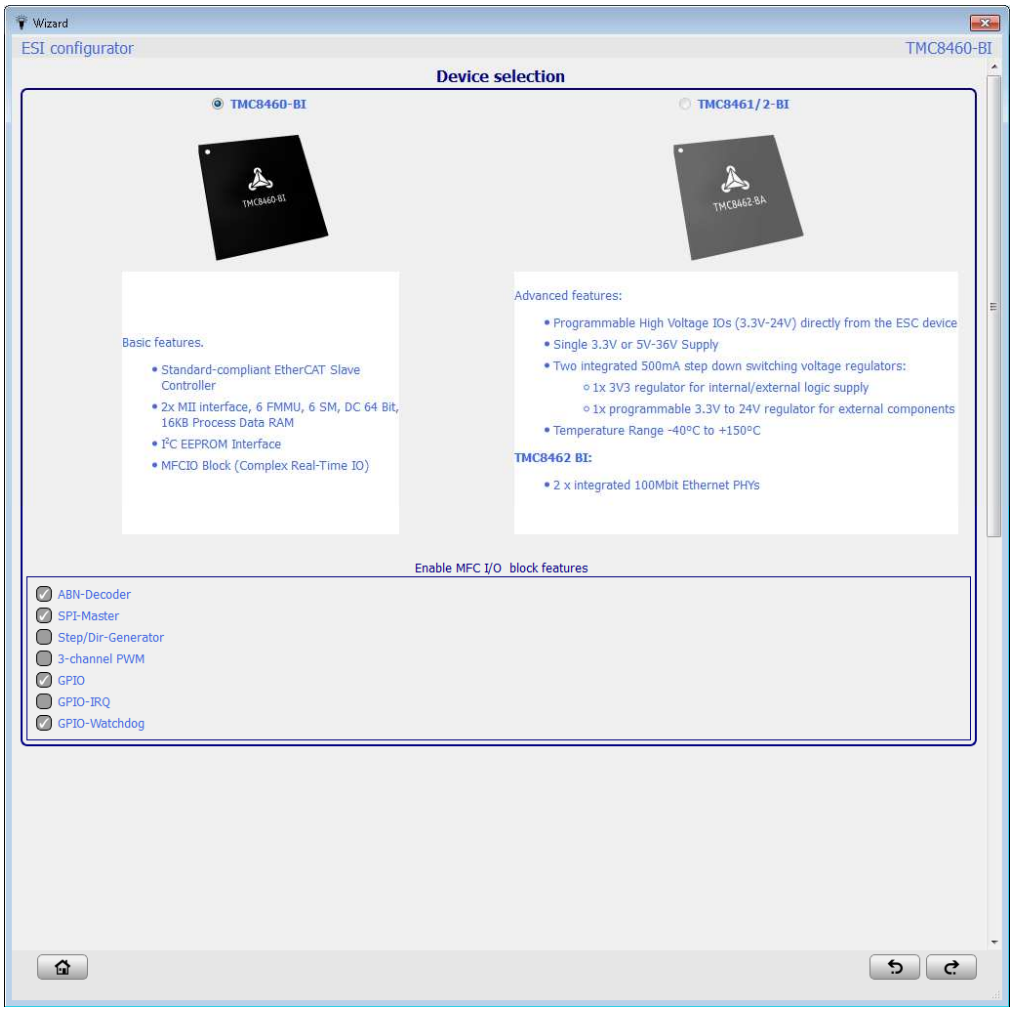


Figure 3 - Wizard Device Selection and Feature Selection

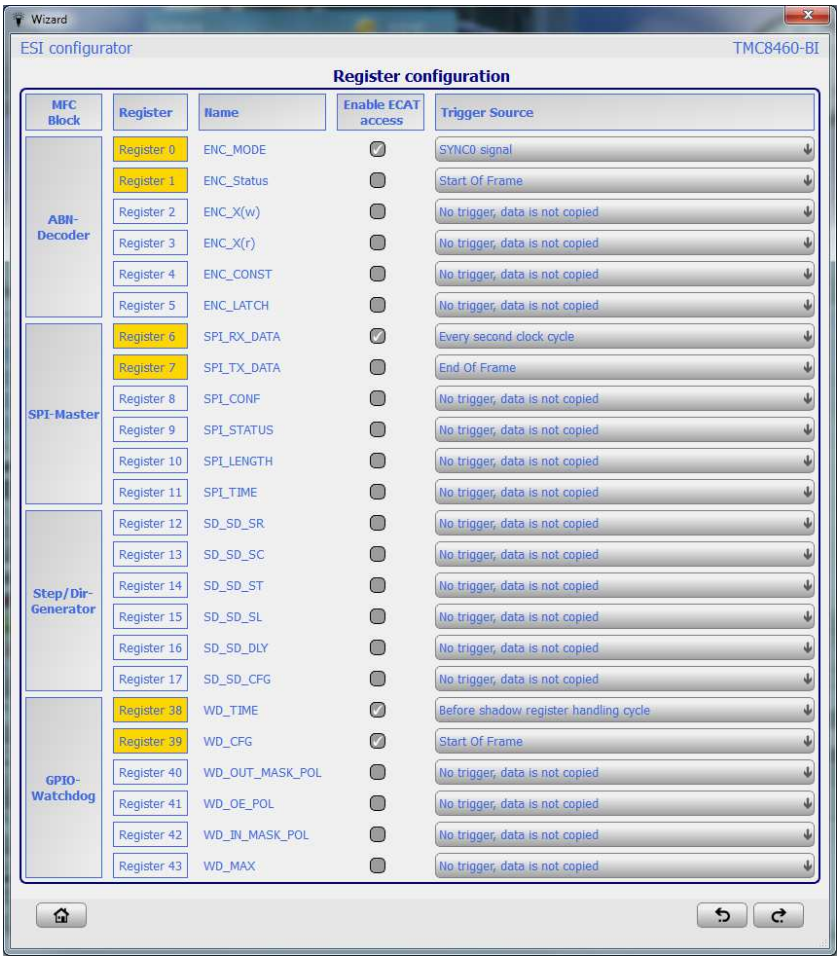


Figure 4 - Wizard Register Selection and Configuration View

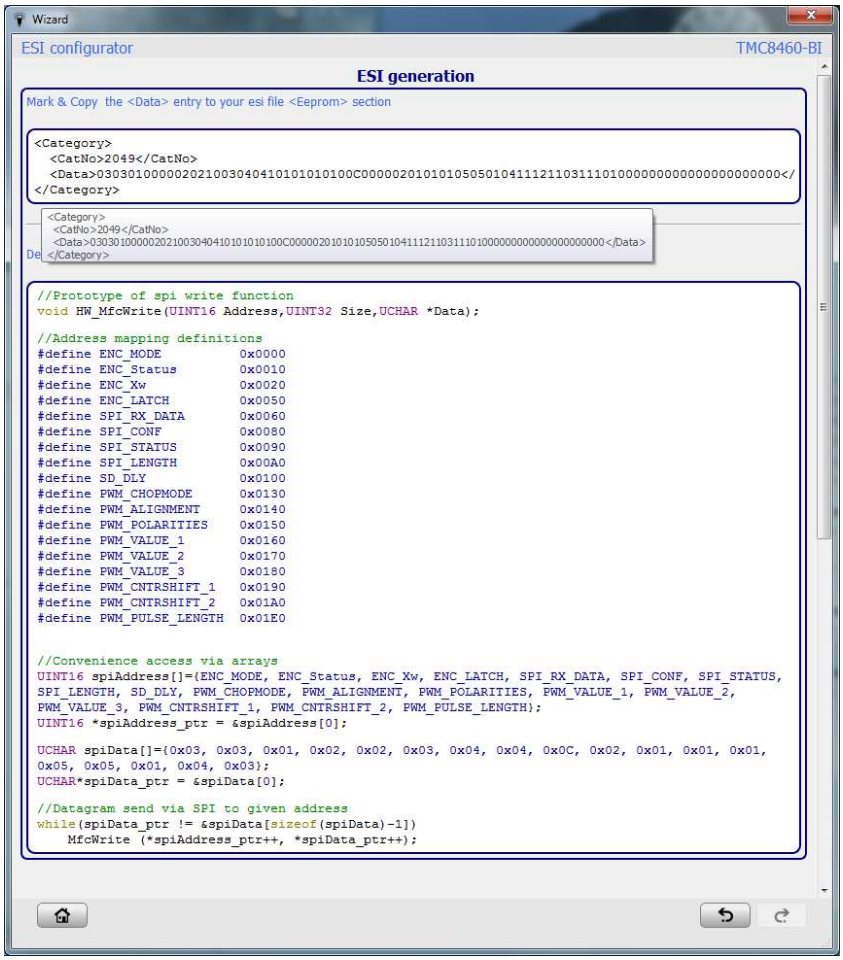


Figure 5 - Wizard output view with EEPROM configuration string and firmware C-code snippets

3 Device Usage and Handling

3.1 Sample Block Diagrams

The TMC8460 allows for flexible system architectures using a microcontroller running the Slave Stack Code (SSC) or using Device Emulation mode without a microcontroller.
The following examples show typical system architectures using the TMC8460.

3.1.1 Typical EtherCAT Slave architecture

The first application diagram shows the TMC8460 in a typical straightforward architecture. The PHYs connect to the TMC8460 using MII interface. Both PHYs and the TMC8460 have the same 25MHz clock source (see Section 3.9.2). The I2C EEPROM is connected to the TMC8460 and contains boot-up configuration required by the ESC after reset or power-cycling. The EEPROM is optionally connected to the µC to allow EEPROM updates via the MCU's firmware.
The µC connects to the TMC8460 using an SPI bus interface to the PDI SPI. The local application is connected to the µC and is controlled by the application layer inside the µC. The application interface depends on the application and is a generic placeholder in this diagram. In this example the MFCIO IO block is not used.

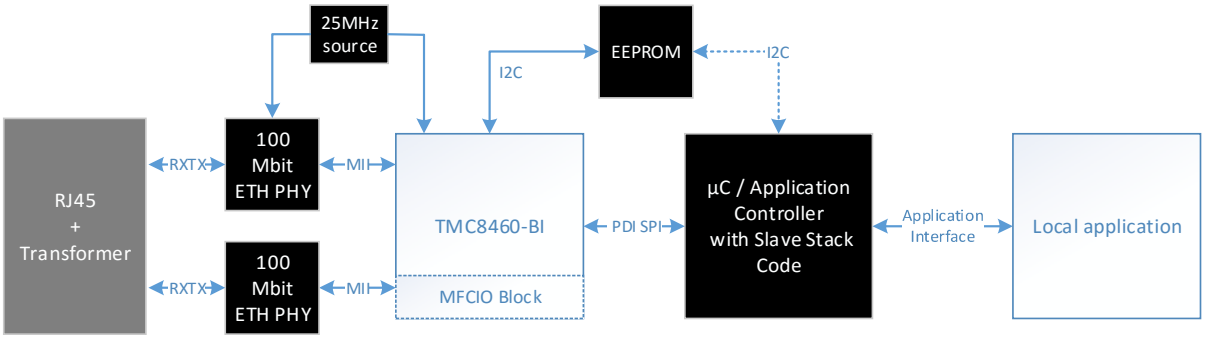


Figure 6 - Application diagram using only the local application controller to interface the application

3.1.2 MFCIO block based Microcontroller Architecture

The second application diagram shows a similar architecture with µC but with extensive use of the MFCIO block features. The special functions of the MFCIO block allow relocating functionality from the µC to the TMC8460. That is, certain compute intense and time-critical functions are moved from software to hardware. The application layer in the µC can focus on interfacing to the real-time bus and for high-level control tasks of the application.
For example, an incremental encoder can directly be connected to the MFCIO block. The µC only reads back the actual position via the dedicated SPI interface MFC CTRL SPI. Additionally, SPI slave chips are directly connected to the MFCIO and not the µC, for example Trinamic's dedicated smart stepper motor drivers, dedicated hardware motion controllers, and simple S/D stepper motor drivers. The MFCIO block master SPI interface, the PWM functions, the S/D function, and the 16MHz clock output are used in this case. The application controller does not need to implement these firmware functions and interfaces but uses the available resources of the TMC8460 instead. Software development is simplified. Other application parts not covered by the MFCIO block still connect to the microcontroller.

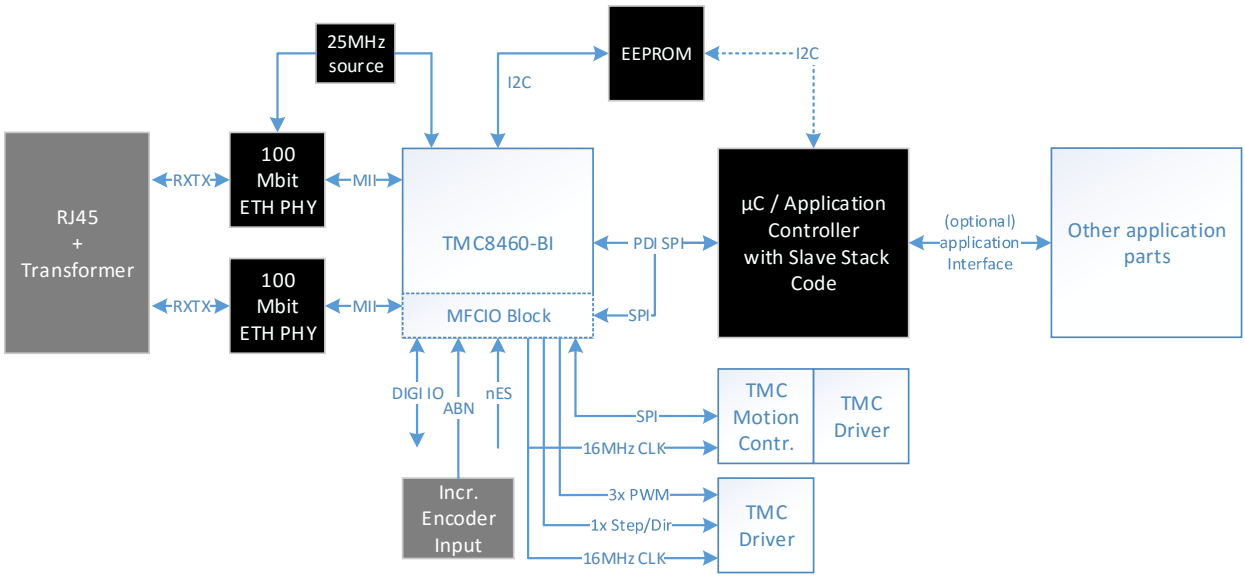


Figure 7 - Application diagram using the MFCIO block features to reduce software overhead and provide real-time hardware support to the MCU. Other application parts may still be connected to the MCU.

3.1.3 Device Emulation Example

The third application diagram shows a more compact architecture using device emulation mode. No µC is required. State machine change requests by the EtherCAT master are directly processed inside the ESC. The MFCIO block is the only application interface available in this architecture and provides the features mentioned under Section 2.1.3. For example, a simple stepper motor slave with hardware motion controller and encoder feedback can be set up without using a µC in the system by only using the features provided by the TMC8460. Since the registers and functions of the MFCIO block can directly be mapped into the PDRAM the EtherCAT master can control the slave.

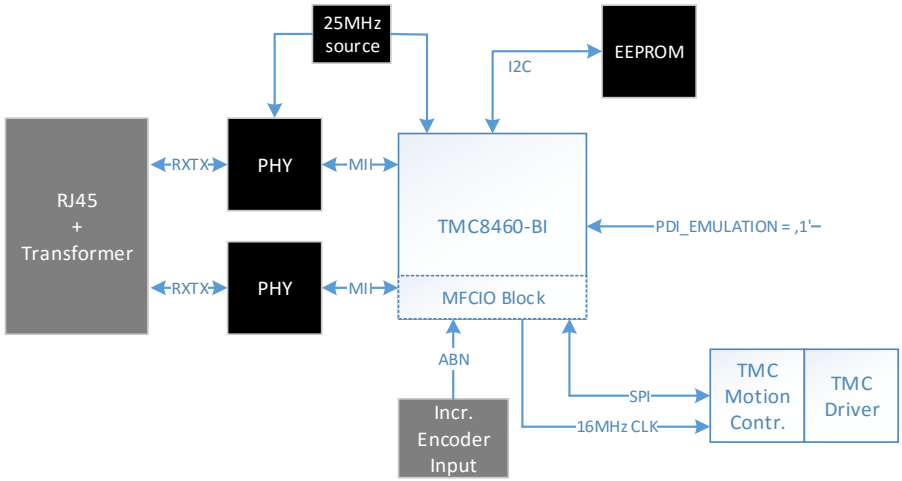
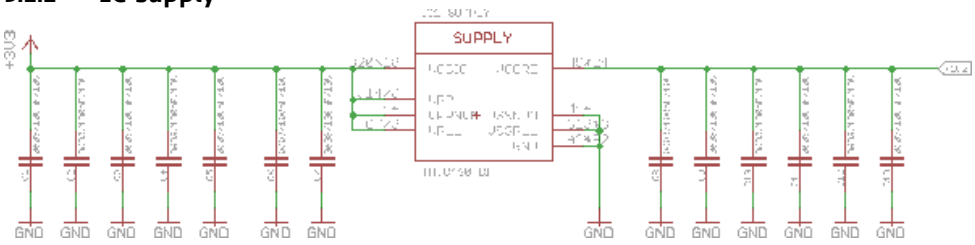


Figure 8 - Application diagram without MCU. The TMC8460 is used in device emulation mode. SPI slave chips and other application peripherals can be connected to the MFCIO block. The EtherCAT master can directly control all the application functions.

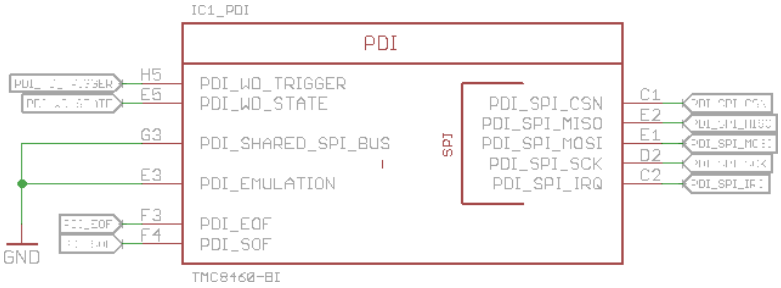
3.2 Samples Circuits

3.2.1 IC supply



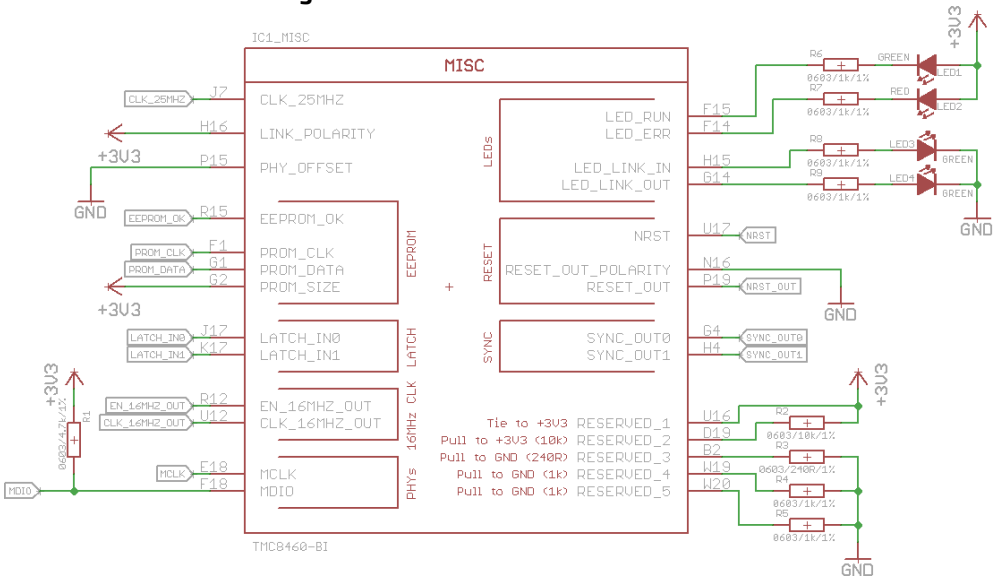
Only a minimal amount of decoupling capacitors is shown here.
If possible every supply pin (1.2V and 3.3V) should have a separate 100nF capacitor connected between it and GND as close to the pin as possible.
Larger capacitor values can be used on the 3.3V and 1.2V supply rails for increased stability.

3.2.2 PDI interface



This is the default configuration for the PDI SPI interface; both PDI_EMULATION and PDI_SHARED_SPI_BUS are tied to GND.
PDI__EMULATION = 0 means that the processor connected to the PDI SPI pins has full control over the ESC registers, the memory and the EtherCAT state machine.
PDI_SHARED_SPI_BUS = 0 means that the signals of the PDI SPI and MFC CTRL SPI buses are completely separate.
The processor can also use the extra signals for Start-/End-Of-Frame and the PDI Watchdog.

3.2.3 Miscellaneous signals



CLK_25MHZ is the clock input, which should be the same signal as the clock for both PHYs. The traces from the oscillator to the TMC8460 and the PHYs should have approximately the same length to avoid timing problems.

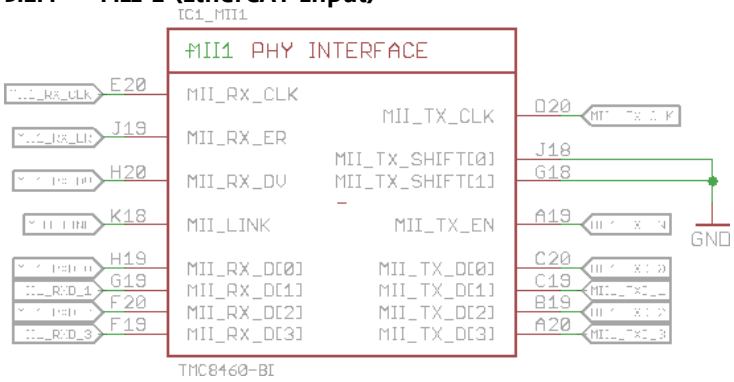
LATCH_IN0, LATCH_IN1, SYNC_OUT0, SYNC_OUT1, NRST_OUT, EN_16MHZ_OUT and CLK_16MHZ_OUT are optional signals that can be used depending on the specific use case.

Configuration Pins with their setting in this example:

LINK_POLARITY	1 (+3.3V)	MII_LINK signal from PHYs is high active
PHY_OFFSET	0 (GND)	PHY address offset is 0
PROM_SIZE	1 (+3.3V)	EEPROM is 32kbit or larger (4Mbit max.)
RESET_OUT_POLARITY	0 (GND)	The RESET_OUT signal is low active

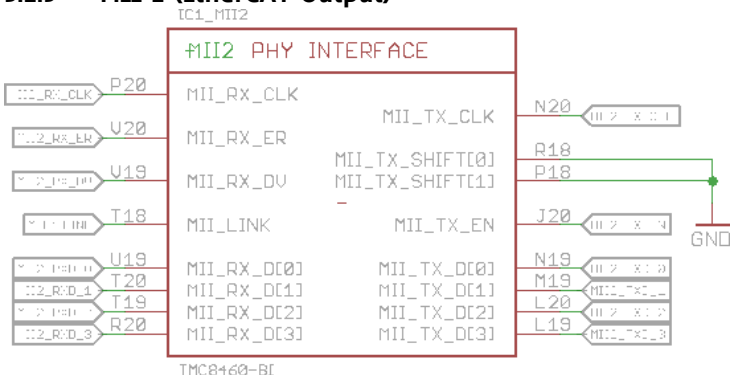
External resistors are required on the MDIO line and for four (4) of the reserved pads.

3.2.4 MII 1 (EtherCAT Input)



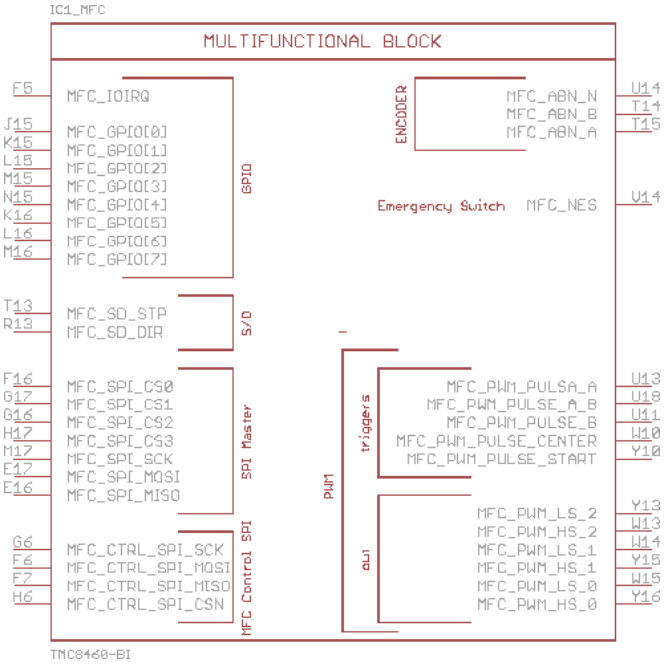
These are the connections to the first PHY, representing the input port of the EtherCAT device.

3.2.5 MII 2 (EtherCAT Output)



These are the connections to the second PHY, representing the output port of the EtherCAT device.

3.2.6 MFC I/Os



The I/Os of the MFC block are shown unconnected here as the actual connections depend on the intended use.

3.3 Pinout and Pin Description

The following table contains the pin description and required pin connections of the TMC8460-BI for the Very Fine Ball Pitch Grid Array package VF400.

Pins not listed in this table are not connected (n.c.) / leave open.

PKG. Pin	Pin Name / Function	DIR	Functional Description / Comments
U17	NRST	I	Low active system reset input
J7	CLK_25MHz	I	25MHz Reference Clock Input, connect to clock source with <25ppm or better, Typically same clock source as used for the PHYs
U12	CLK_16MHz_OUT	O	16MHz auxiliary clock output, enabled by EN_16MHz_OUT
R12	EN_16MHz_OUT	I	Enable signal for 16 MHz auxiliary clock output: 0 = off, 1 = on, 16 MHz available at CLK_16MHz_OUT
P19	RESET_OUT	O	TMC8460 auxiliary output / active level defined by RESET_OUT_POL, reset by ECAT (reset register 0x0040), RESET_OUT has to trigger nRESET, which clears RESET_OUT.
N16	RESET_OUT_POL	I	Configures active level of the RESET_OUT signal: 0 = low active, 1 = high active
R15	EEPROM_OK	O	Signal indicating that EEPROM has been loaded, 0 = not ready, 1 = EEPROM loaded
F1	PROM_CLK	O	External I2C EEPROM clock signal, Use 1K pull up resistor to 3.3V
G1	PROM_DATA	I/O	External I2C EEPROM data signal, Use 1k pull up resistor to 3V3
G2	PROM_SIZE	I	Selects between two different EEPROM sizes since the communication protocol for EEPROM access changes if a size > 16k is used (an additional address byte is required then). 0 = up to 16K EEPROM, 1 = 32 kbit-4Mbit EEPROM
G4	SYNC_OUT0	O	Distributed Clocks synchronization output 0, Typically connect to MCU
H4	SYNC_OUT1	O	Distributed Clocks synchronization output 1, typically connect to MCU
J17	LATCH_IN0	I	Latch input 0 for distributed clocks, connect to GND if not used.
K17	LATCH_IN1	I	Latch input 1 for distributed clocks, Connect to GND if not used.
F14	LED_ERR	O	Error Status LED, connect to red LED (Cathode) 0 = LED on, 1 = LED off
H15	LED_LINK_IN	O	Link In Port Activity, connect to green LED (Anode) 0 = LED off, 1 = LED on
G14	LED_LINK_OUT	O	Link Out Port Activity, connect to green LED (Anode) 0 = LED off, 1 = LED on

F15	LED_RUN	O	Run Status LED, connect to green LED (Cathode) 0 = LED on, 1 = LED off
E3	PDI_EMULATION	I	Selects between normal operation with state machine inside ESC or device emulation mode (no µC required). Has weak internal pull down. 0 = default, no device emulation, µC required 1 = device emulation
F3	PDI_EOF	O	Ethernet End-of-Frame if 1
G3	PDI_SHARED_SPI_BUS	I	Selects between separate SPI busses (SDI, SDO, SCK) or one SPI bus with two CS lines for the PDI and MFCIO SPI interface: 0 = two separate SPI busses, 1 = one shared SPI bus using the PDI_SPI_x bus lines (not the PDI_MFC_SPI bus lines)
F4	PDI_SOF	O	Ethernet Start-of-Frame if 1
C1	PDI_SPI_CSN	I	PDI_SPI is the primary process data interface of the TMC8460 to the MCU. Connect to MCU SPI master interface
C2	PDI_SPI_IRQ	O	Interrupt signal for primary process data interface, Connect to MCU
E2	PDI_SPI_MISO	O	PDI_SPI is the primary process data interface of the TMC8460 to the MCU. Connect to MCU SPI master interface
E1	PDI_SPI_MOSI	I	PDI_SPI is the primary process data interface of the TMC8460 to the MCU. Connect to MCU SPI master interface
D2	PDI_SPI_SCK	I	PDI_SPI is the primary process data interface of the TMC8460 to the MCU. Connect to MCU SPI master interface
E5	PDI_WD_STATE	O	Watchdog state, 0: Expired, 1: Not expired
H5	PDI_WD_TRIGGER	O	Watchdog trigger if 1
H6	MFC_CTRL_SPI_CSN	I	SPI Control Interface to MFCIO special function block, Connect to MCU master SPI interface, Weak internal pull down
F7	MFC_CTRL_SPI_MISO	O	SPI Control Interface to MFCIO special function block, Connect to MCU SPI master interface
F6	MFC_CTRL_SPI_MOSI	I	SPI Control Interface to MFCIO special function block, Connect to MCU SPI master interface, Has weak internal pull down
G6	MFC_CTRL_SPI_SCK	I	SPI Control Interface to MFCIO special function block, Connect to MCU master SPI interface, Weak internal pull up
F5	MFCIO_IRQ	O	MFCIO block IRQ for configurable events, connect to MCU, high active
T15	MFC_ABN_A	I	MFCIO block incremental encoder unit input
T14	MFC_ABN_B	I	MFCIO block incremental encoder unit input
U14	MFC_ABN_N	I	MFCIO block incremental encoder unit input
J15	MFC_GPIO[0]	I/O	MFCIO block GPIO port, configurable
K15	MFC_GPIO[1]	I/O	MFCIO block GPIO port, configurable

L15	MFC_GPIO[2]	I/O	MFCIO block GPIO port, configurable
M15	MFC_GPIO[3]	I/O	MFCIO block GPIO port, configurable
N15	MFC_GPIO[4]	I/O	MFCIO block GPIO port, configurable
K16	MFC_GPIO[5]	I/O	MFCIO block GPIO port, configurable
L16	MFC_GPIO[6]	I/O	MFCIO block GPIO port, configurable
M16	MFC_GPIO[7]	I/O	MFCIO block GPIO port, configurable
Y16	MFC_PWM_HS_0	O	MFCIO block PWM unit high side output
Y15	MFC_PWM_HS_1	O	MFCIO block PWM unit high side output
W13	MFC_PWM_HS_2	O	MFCIO block PWM unit high side output
W15	MFC_PWM_LS_0	O	MFCIO block PWM unit low side output
W14	MFC_PWM_LS_1	O	MFCIO block PWM unit low side output
Y13	MFC_PWM_LS_2	O	MFCIO block PWM unit low side output
V14	MFC_nES	I	low active (not) Emergency Stop/Switch/Halt (to bring PWM or other outputs into a safe state), the event must be cleared actively, has weak internal pull down, must be driven high for normal operation
U13	MFC_PWM_PULSE_A	O	PWM Trigger Pulse for e.g. ADC synchronization, MFC_PWM_PULSE_A is configurable and high for one clock cycle in each PWM cycle
U18	MFC_PWM_PULSE_AB	O	PWM Trigger Pulse for e.g. ADC synchronization, MFC_PWM_PULSE_AB combines MFC_PWM_PULSE_A and MFC_PWM_PULSE_B in one signal
U11	MFC_PWM_PULSE_B	O	PWM Trigger Pulse for e.g. ADC synchronization, MFC_PWM_PULSE_B is configurable and high for one clock cycle in each PWM cycle
W10	MFC_PWM_PULSE_C	O	PWM Trigger Pulse for e.g. ADC synchronization, MFC_PWM_PULSE_C is high for one clock cycle in the center of each PWM cycle
Y10	MFC_PWM_PULSE_S	O	PWM Trigger Pulse for e.g. ADC synchronization, MFC_PWM_PULSE_S is high for one clock cycle on each PWM cycle start
R13	MFC_SD_DIR	O	MFCIO block Step/Direction unit output
T13	MFC_SD_STP	O	MFCIO block Step/Direction unit output
F16	MFC_SPI_CS0	O	MFCIO block SPI master unit interface
G17	MFC_SPI_CS1	O	MFCIO block SPI master unit interface
G16	MFC_SPI_CS2	O	MFCIO block SPI master unit interface
H17	MFC_SPI_CS3	O	MFCIO block SPI master unit interface
E16	MFC_SPI_MISO	I	MFCIO block SPI master unit interface
E17	MFC_SPI_MOSI	O	MFCIO block SPI master unit interface
M17	MFC_SPI_SCK	O	MFCIO block SPI master unit interface
H16	LINK_POLARITY	I	selects polarity of the PHYs link signal: 0 = low active, 1 = high active
K18	MII1_LINK	I	MII interface to PHY of link in port
E20	MII1_RXCLK	I	MII interface to PHY of link in port
H19	MII1_RXD[0]	I	MII interface to PHY of link in port

G19	MII1_RXD[1]	I	MII interface to PHY of link in port
F20	MII1_RXD[2]	I	MII interface to PHY of link in port
F19	MII1_RXD[3]	I	MII interface to PHY of link in port
H20	MII1_RXDV	I	MII interface to PHY of link in port
J19	MII1_RXER	I	MII interface to PHY of link in port
J18	MII1_TX_SHIFT[0]	I	Used for clock shift compensation on TX port, Weak internal pull down
G18	MII1_TX_SHIFT[1]	I	Used for clock shift compensation on TX port, Weak internal pull down
D20	MII1_TXCLK	I	MII interface to PHY of link in port
C20	MII1_TXD[0]	O	MII interface to PHY of link in port
C19	MII1_TXD[1]	O	MII interface to PHY of link in port
B19	MII1_TXD[2]	O	MII interface to PHY of link in port
A20	MII1_TXD[3]	O	MII interface to PHY of link in port
A19	MII1_TXEN	O	MII interface to PHY of link in port
T18	MII2_LINK	I	MII interface to PHY of link out port
P20	MII2_RXCLK	I	MII interface to PHY of link out port
U19	MII2_RXD[0]	I	MII interface to PHY of link out port
T20	MII2_RXD[1]	I	MII interface to PHY of link out port
T19	MII2_RXD[2]	I	MII interface to PHY of link out port
R20	MII2_RXD[3]	I	MII interface to PHY of link out port
V19	MII2_RXDV	I	MII interface to PHY of link out port
V20	MII2_RXER	I	MII interface to PHY of link out port
R18	MII2_TX_SHIFT[0]	I	Used for clock shift compensation on TX port, weak internal pull down
P18	MII2_TX_SHIFT[1]	I	Used for clock shift compensation on TX port, Weak internal pull down
N20	MII2_TXCLK	I	MII interface to PHY of link out port
N19	MII2_TXD[0]	O	MII interface to PHY of link out port
M19	MII2_TXD[1]	O	MII interface to PHY of link out port
L20	MII2_TXD[2]	O	MII interface to PHY of link out port
L19	MII2_TXD[3]	O	MII interface to PHY of link out port
J20	MII2_TXEN	O	MII interface to PHY of link out port
P15	PHY_OFFSET	I	PHY Address Offset: 0 = Offset = 0, 1 = Offset = 1
E18	MCLK	O	PHY management clock, connect all PHYs to this bus
F18	MDIO	I/O	PHY management data, connect all PHYs to this bus if required,

			Use 4K7 pull up resistor to 3.3V
B2	reserved		Connect with 240R to GND
D19	reserved		pin must be pulled high, connect with 10K to 3.3V
W19	reserved	I	Pull down with 1K to GND
W20	reserved		Pull down with 1K to GND
U16	reserved	I	Connect to 3.3V
H11	+1V2 (VCORE)		
H9	+1V2 (VCORE)		
J10	+1V2 (VCORE)		
J12	+1V2 (VCORE)		
K11	+1V2 (VCORE)		
K13	+1V2 (VCORE)		
K9	+1V2 (VCORE)		
L10	+1V2 (VCORE)		
L12	+1V2 (VCORE)		
M11	+1V2 (VCORE)		
M13	+1V2 (VCORE)		
M9	+1V2 (VCORE)		
N10	+1V2 (VCORE)		
N12	+1V2 (VCORE)		
N6	+1V2 (VCORE)		
N8	+1V2 (VCORE)		
P11	+1V2 (VCORE)		
P7	+1V2 (VCORE)		
P9	+1V2 (VCORE)		
R10	+1V2 (VCORE)		
R7	+1V2 (VCORE)		
R8	+1V2 (VCORE)		
T8	+1V2 (VCORE)		
U5	+1V2 (VCORE)		
B20	+3V3 (VCCIO)		
E19	+3V3 (VCCIO)		
F2	+3V3 (VCCIO)		
G15	+3V3 (VCCIO)		
G5	+3V3 (VCCIO)		
G7	+3V3 (VPLL)		
H14	+3V3 (VPLL)		
H18	+3V3 (VCCIO)		
J14	+3V3 (VPP)		
J8	+3V3 (VCCIO)		
L14	+3V3 (VPP)		
L17	+3V3 (VCCIO)		

M20	+3V3 (VCCIO)		
N14	+3V3 (VCCIO)		
P14	+3V3 (VPP)		
P16	+3V3 (VCCIO)		
P5	+3V3 (VPP)		
P6	+3V3 (PLL)		
R14	+3V3 (VCCIO)		
R19	+3V3 (VCCIO)		
T11	+3V3 (VPP)		
T12	+3V3 (VCCIO)		
U15	+3V3 (VCCIO)		
V18	+3V3 (VCCIO)		
W11	+3V3 (VCCIO)		
W17	+3V3 (VPP)		
Y14	+3V3 (VCCIO)		
Y17	+3V3 (VPP)		
A12	GND		
A2	GND		
B15	GND		
B5	GND		
C18	GND		
C8	GND		
D1	GND		
D11	GND		
E14	GND		
E4	GND		
F17	GND		
G11	GND		
G13	GND (PLL)		
G20	GND		
G9	GND		
H10	GND		
H12	GND		
H13	GND		
H3	GND		
H7	GND (PLL)		
H8	GND		
J11	GND		
J13	GND		
J16	GND		
J9	GND		
K10	GND		

K12	GND		
K14	GND		
K19	GND		
K8	GND		
L11	GND		
L13	GND		
L2	GND		
L9	GND		
M10	GND		
M12	GND		
M14	GND		
M5	GND		
M8	GND		
N11	GND		
N13	GND		
N18	GND		
N5	GND (PLL)		
N7	GND		
N9	GND		
P1	GND		
P10	GND		
P12	GND		
P4	GND		
P8	GND		
R6	GND		
R9	GND		
T1	GND		
T10	GND		
T16	GND		
T17	GND		
T2	GND		
T3	GND		
T4	GND		
T5	GND		
T6	GND		
T7	GND		
T9	GND		
U1	GND		
U10	GND		
U2	GND		
U20	GND		
U3	GND		

U4	GND		
U6	GND		
U7	GND		
U8	GND		
U9	GND		
V1	GND		
V10	GND		
V13	GND		
V2	GND		
V3	GND		
V4	GND		
V5	GND		
V6	GND		
V7	GND		
V8	GND		
V9	GND		
W1	GND		
W16	GND		
W3	GND		
W5	GND		
W7	GND		
W9	GND		
Y1	GND		
Y19	GND		
Y3	GND		
Y5	GND		
Y7	GND		
Y9	GND		

3.4 Ethernet PHYs

The TMC8460-BI requires Ethernet PHYs with MII interface.
The MII interface of the TMC8460 is optimized for low additional delays by omitting a transmit FIFO. Therefore, additional requirements to Ethernet PHYs exist and not every Ethernet PHY is suited. Therefore, please see the Ethernet PHY Selection Guide provided by the ETG:
http://download.beckhoff.com/download/Document/EtherCAT/Development_products/AN_PHY_Selection_GuideV2.3.pdf.

The TMC8460-BI has been successfully tested in combination with the following Ethernet PHYs so far:

- IC+ IP101GA: http://www.icplus.com.tw/pp-IP101G.html
- Micrel KSZ8721BLI: http://www.micrel.com/_PDF/Ethernet/datasheets/ks8721bl-sl.pdf
- Micrel KSZ8081: http://www.micrel.com/_PDF/Ethernet/datasheets/KSZ8081MNX-RNB.pdf

The clock source of the PHYs is the same as for the TMC8460-BI (25 MHz).

3.4.1 Ethernet PHY MII interface and MI interface

100-Mbit Ethernet PHYs with MII interface are required. In addition, they need to provide a link signal indicating stable 100Mbit bus connection.
The following diagram shows the interface pins. The table in Section 3.3 contains the pin descriptions of the interface. TX_CLK is optional. It is used for automatic TX Shift compensation.

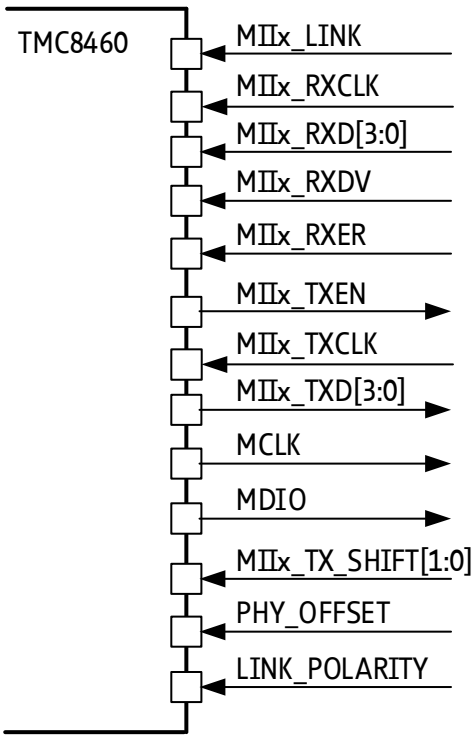


Figure 9 : MII Interface Signals

Table 1 : MII interface signal description and connection

TMC8460 pin	Usage/description	Typical PHY pin name
MIIx_LINK	Input signal provided by the PHY if a stable 100 Mbit/s (Full Duplex) link is established. MIIx_LINK must be constantly driven with the configured polarity during operation.	PHY and configuration dependent, typically one of the status LED pins
MIIx_RXCLK	Receive clock	RX_CLK
MIIx_RXD[3:0]	Receive data	RXD3…RXD0

MIIx_RXDV	Receive data valid	RX_DV
MIIx_RXER	Receive error	RX_ER
MIIx_TXEN	Transmit enable	TX_EN
MIIx_TXCLK	Transmit clock (optional for automatic TX shift)	TX_CLK
MIIx_TXD[3:0]	Transmit data	TXD3…TXD0
MCLK	Management Interface clock, MCLK is driven rail-to-rail, idle value is High.	MDC
MDIO	Management Interface data, MDIO must have a pull-up resistor (4.7 kΩ recommended)	MDIO

3.4.2 PHY Configuration Pins

Besides the standard MII and MI interface signals, the TMC8460 has three configuration pins related to the PHYs.

- **LINK_POLARITY**: this pin allows configuring the polarity of the link signal of the PHY. PHYs of different manufacturers may use different polarities at the PHY's pins. In addition, some PHYs allow for bootstrap configuration with pull-up and pull-down resistors. This bootstrap information is used by the PHY at power-up / reset and also influences the polarity of the original pin function. Therefore, the link polarity needs to be configurable.

- **PHY_OFFSET**: The TMC8460 addresses Ethernet PHYs using logical port number plus PHY address offset. Typically, the Ethernet PHY addresses should correspond with the logical port number, so PHY addresses 0 to 1 are used. A PHY address offset of 1 can be applied (PHY_OFFSET = '1') which moves the PHY addresses to a range of 1 to 2. The TMC8460 expects logical port 0 to have PHY address 0 plus PHY address offset.

- **MIIx_TX_SHIFT[1:0]:** TMC8460 and Ethernet PHYs share the same clock source. Thus, TX_CLK from the PHY has a fixed phase relation to the MII interface TX part of TMC8460. Thus, TX_CLK must not be connected and the delay of a TX FIFO inside the IP Core is saved.
  In order to fulfill the setup/hold requirements of the PHY, the phase shift between TX_CLK and MIIx_TX_EN and MIIx_TXD[3:0] has to be controlled.
  - Manual TX Shift compensation with additional delays for MIIx_TXEN/MIIx_TXD[3:0] of 10, 20, or 30 ns. Such delays can be added using the TX Shift feature and applying MIIx_TX_SHIFT[1:0]. MIIx_TXH_SHIFT[1:0] determine the delay in multiples of 10 ns for each port. Set MIIx_TXCLK to zero if manual TX Shift compensation is used.
  - Automatic TX Shift compensation if the TX Shift feature is selected: connect MIIx_TXCLK and the automatic TX Shift compensation will determine correct shift settings. Set MIIx_TX_SHIFT[1:0] to 0 in this case.

3.5 PDI SPI

The PDI SPI interface is the interface used to access the ESC registers and the process data RAM from an external microcontroller. The SPI clock frequency can be up to 30 MHz.
The interface is configurable via the EEPROM. The default configuration (SPI mode 3, low active chip select) is recommended. For further details, see the ESC SPI slave configuration registers in Section 5.28.
The following diagram shows all signals related to the PDI SPI interface.

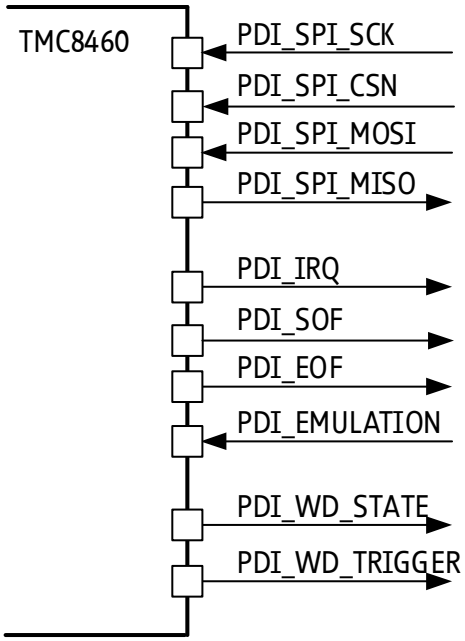


Figure 10 - PDI SPI Interface Signals

Table 2 : PDI SPI interface signal description and connection

TMC8460 pin	**Usage/description**	**Typical µC pin name**
PDI_SPI_SCK	SPI master clock	SCK
PDI_SPI_CSN	SPI chip select for the TMC8460 PDI	SSx
PDI_SPI_MOSI	Master out slave in data	MOSI
PDI_SPI_MISO	Master in slave out data	MISO
PDI_IRQ	Configurable IRQ from PDI	General purpose IO
PDI_EMULATION	0: default mode for complex slaves, state machine changes processed in microcontroller firmware (SSC) 1: device emulation mode for, e.g., simple slaves, state machine changes directly handled in the ESC	General purpose IO or connected to either ground or 3.3V.
PDI_SOF	Indicates start of an Ethernet/EtherCAT frame (MII_RXDV = '1')	General purpose IO
PDI_EOF	Indicates start of an Ethernet/EtherCAT frame	General purpose IO
PDI_WD_STATE	0: Watchdog expired 1: Watchdog not expired	General purpose IO
PDI_WD_TRIGGER	Watchdog triggered if '1'	General purpose IO

3.5.1 SPI protocol description

Each SPI datagram contains a 2- or 3-byte address/command part and a data part. For addresses below 0x2000, the 2-byte addressing mode can be used, the 3 byte addressing mode can be used for all addresses.

Table 3 : PDI-SPI commands

C2	C1	C0	Command
0	0	0	NOP (no operation, no following data bytes)
0	0	1	Reserved
0	1	0	Read
0	1	1	Read with wait state byte

1	0	0	Write
1	0	1	Reserved
1	1	0	Address extension, signaling 3 byte mode
1	1	1	Reserved

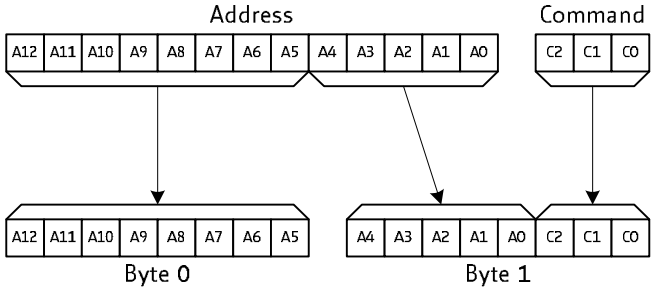


Figure 11 - 2 byte addressing mode

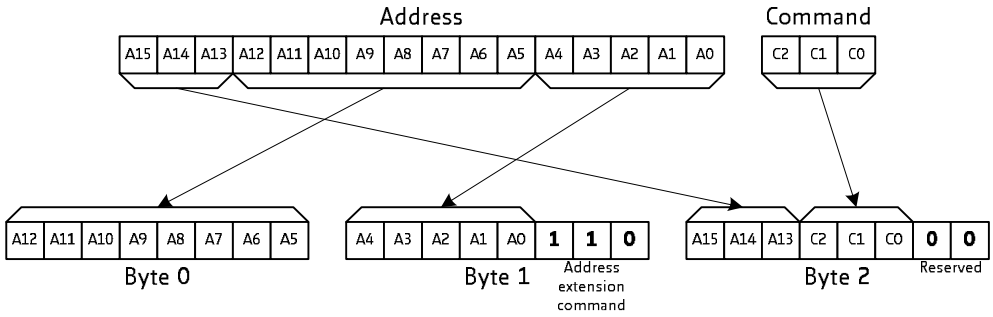


Figure 12 - 3 byte addressing mode

Unless highest performance is required, using only the 3-byte addressing mode and the read with wait state command is recommended since it reduces the need for special cases in the software.

During the address/command bytes, the ESC replies with the contents of the event request registers (0x0220, 0x0221 and in 3 byte addressing mode 0x0222).

COMMAND 0 – NOP

This command can be used for checking the event request registers and resetting the PDI watchdog without a read or write access.

Example datagram: **0x00 0x00**
Example reply (AL Control event bit is set): **0x01 0x00**

COMMAND 2 – READ

With the read command, an arbitrary amount of data can be read from the device. The first byte read is the data from the address given by the address/command bytes. With every read byte, the address is incremented. During the data transfer, the SPI master sends 0x00 except for the last byte where a 0xFF is sent.
When using this command, a pause of 240ns or more must be included between the address/command bytes and the data bytes for the ESC to fetch the requested data.

Example datagram (Read from address 0x0120 and 0x0121): **0x09 0x02 0x00 0xFF**
Example reply (Operational State requested): **0x01 0x00 0x08 0x00**

COMMAND 3 – READ WITH WAIT STATE BYTE

This command is similar to the Read command with an added dummy byte between the address/command part and the data part of the datagram. This allows enough time to fetch the data in any case.

Example datagram (Read starting at address 0x3400): **0xA0 0x06 0x2C 0xFF 0x00 0x00 0x00 0xFF**
Example reply (0xXX is undefined data): **0x00 0x00 0x00 0xXX 0x44 0x41 0x54 0x41**

COMMAND 4 – WRITE

The write command allows writing of an arbitrary number of bytes to writable ESC registers or the process data RAM. It requires no wait state byte or delay after the address/command bytes. After every transmitted byte, the address is incremented.

Example datagram (Write starting at address 0x4200): **0x10 0x06 0x50 0x4C 0x48**
Example reply (0xXX is undefined data): **0x00 0x00 0x00 0xXX 0xXX**
Address 0x4200 now contains 0x4C, Address 0x4201 contains 0x48

COMMAND 6 – ADDRESS EXTENSION

The address extension command is mainly used for the 3-byte addressing mode as shown in Figure 12. For SPI masters that can only process datagrams with an even number of bytes, it might be necessary to pad the datagram. This can be achieved by duplicating the third byte of the 3-byte address/command part and using the address extension command in all but the last duplicate.

For example, a SPI master that is only capable of transmitting a multiple of 4 bytes cannot use the example datagram for a write access above since it contains 5 bytes. With three added padding bytes, the master has to transmit two 4-byte groups.

Example datagram (Write starting at address 0x4200): **0x10 0x06 0x58 0x58 0x58 0x50 0x4C 0x48**
Example reply (0xXX is undefined data): **0x00 0x00 0x00 0xXX 0xXX 0xXX 0xXX 0xXX**

3.5.2 Timing example

This example shows a generic read access with wait state and 2 byte addressing. All configurable options are shown. The delays between the transferred bytes are just to show the byte boundaries and are not required.

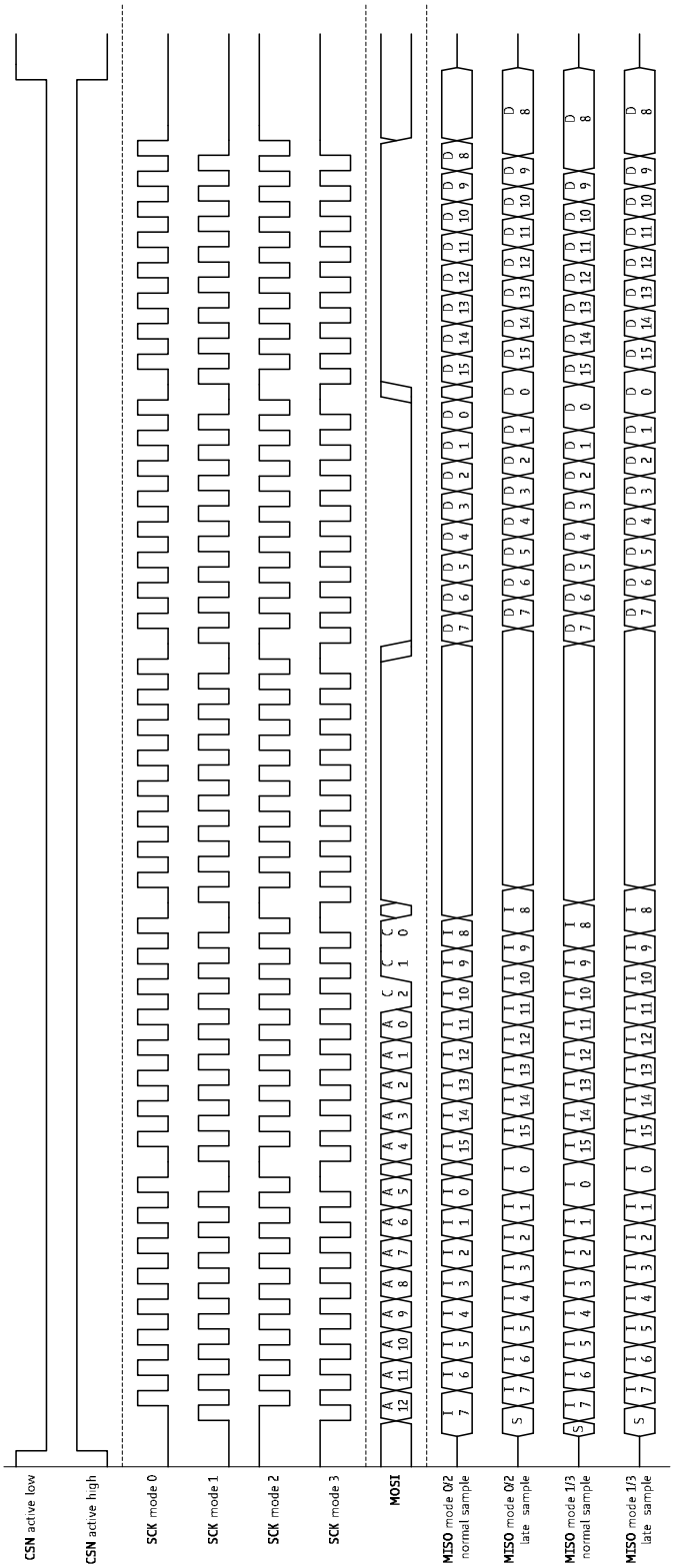


Figure 13 - PDI SPI timing example

3.6 MFC CTRL SPI

The MFC Control SPI is a SPI mode 3 slave with low active chip select. It allows an external microcontroller to access the MFC registers. The SPI clock frequency can be up to 30MHz.
The following diagram shows all signals related to the MFC CTRL SPI interface.

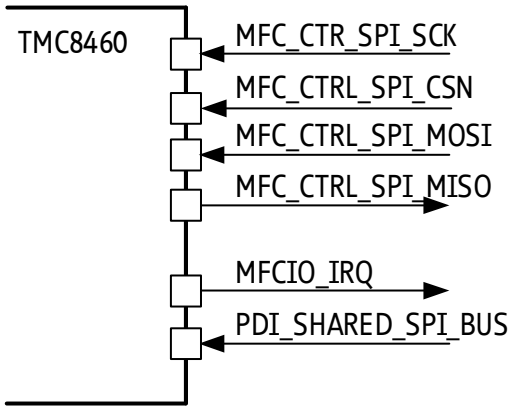


Figure 14 - MFC CTRL SPI interface signals

Table 4 : MFC CTRL SPI interface signal description and connection

TMC8460 pin	**Usage/description**	**Typical µC pin name**
MFC_CTRL_SPI_SCK	SPI master clock	SCK
MFC_CTRL_SPI_CSN	SPI chip select for the TMC8460	SSx
MFC_CTRL_SPI_MOSI	Master out slave in data	MOSI
MFC_CTRL_SPI_MISO	Master in slave out data	MISO
MFCIO_IRQ	Configurable IRQ from MFCIO block	General purpose IO
PDI_SHARED_SPI_BUS	0: separate SPI buses for PDI and MFC CTRL 1: shared/common SPI bus for PDI and MFC CTRL with 2 CSN signals using the PDI SPI bus. The SPI bus signals MFC_CTRL_SPI_SCK, MFC_CTRL_SPI_MISO, MFC_CTRL_SPI_MOSI can be left open in this case	General purpose IO or connected to either ground or 3.3V.

The protocol of the MFC CTRL SPI is the same as the PDI SPI interface. The addresses for register access are calculated using the register number (from 0 to 44) and the byte number in each register. To calculate the address, the register number is shifted left by 4 bits and the byte number is added as the 4 lowest bits.
Access using the 3 byte addressing mode is possible, and can be used when 2 byte mode is not implemented for the PDI SPI but since the highest bits of the address are always 0, accessing the MFC Control SPI via 2 byte mode is sufficient.

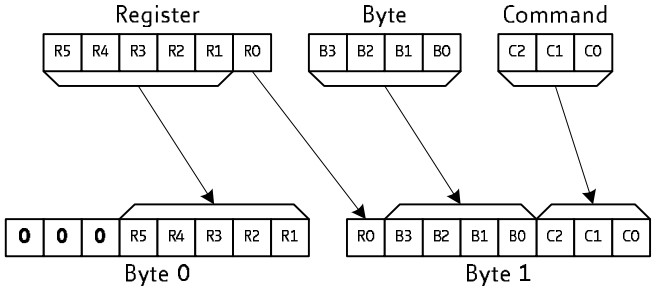


Figure 15 - 2-byte MFC register access

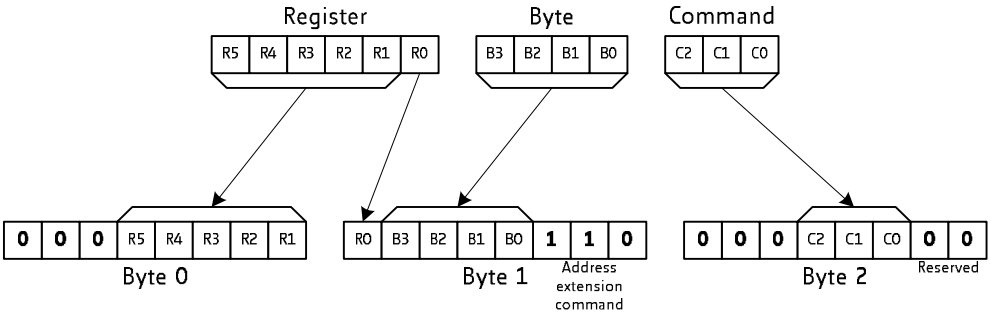


Figure 16 - 3-byte MFC register access

3.6.1 Timing example

This example shows a generic MFC register read access with wait state. The delays between the transferred bytes are just to show the byte boundaries and are not required.

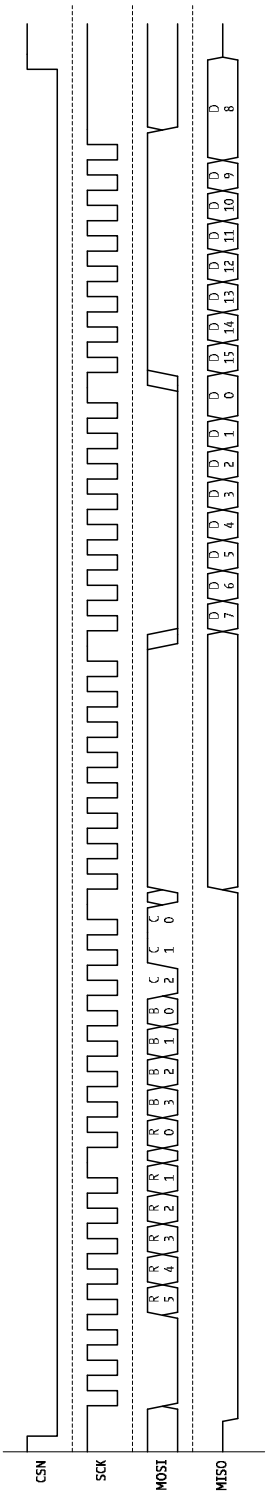


Figure 17 - MFC Control SPI timing example

3.6.2 Sharing Bus Lines with PDI SPI

To reduce overall number of signals on the PCB or if the local application controller has only one SPI interface, the MFC CTRL SPI bus can use the SPI bus signals of the PDI SPI. Therefore, both interfaces are internally switched on the PDI SPI interface. The original MFC CTRL SPI signals (MOSI, MISO, and SCK) remain unconnected in this case. Only the MFC_CTRL_SPI_CSN pin/signal must be used if the MFCIO block should be accessed.
To share the SPI bus lines, configuration pin PDI_SHARED_SPI_BUS must be pulled high as shown in the figure below.

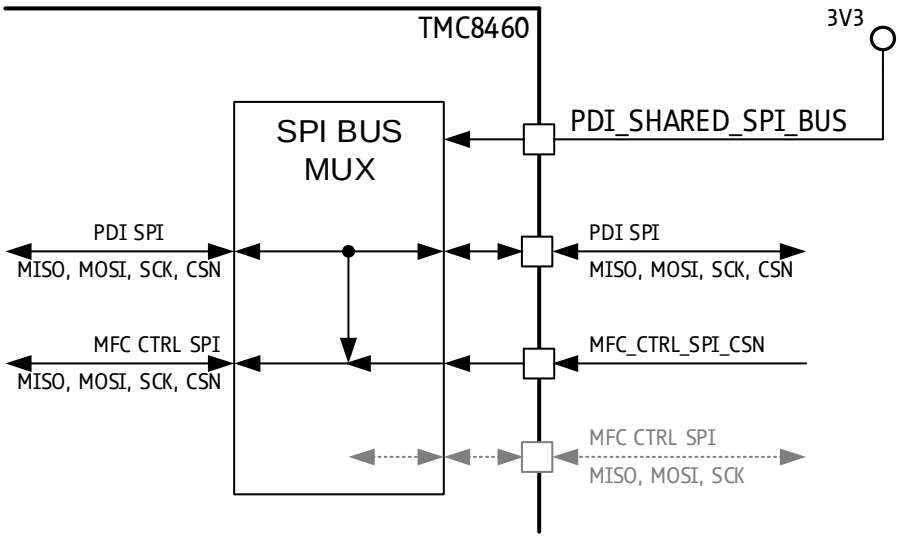


Figure 18 - Shared SPI bus configuration

3.7 EEPROM Interface

The TMC8460 contains an I2C master interface with PROM_CLK and PROM_DATA. Both PROM_CLK and PROM_DATA require an external pull-up (4.7 kΩ recommended).
Both 1 byte and 2 byte addressed EEPROMs are supported. The EEPROM size is configurable using the PROM_SIZE pin (0 = up to 16K EEPROM, 1 = 32 kbit-4Mbit EEPROM).
The output signal EEPROM_OK indicates that the EEPROM content has been successfully loaded.
The EEPROM interface is intended to be a point-to-point interface between TMC8460 and I2C EEPROM. If other I2C masters are required to access the EEPROM using the same I2C bus, the TMC8460 must be in reset state. A typical use case is to (re)program the EEPROM on the board using the application controller.

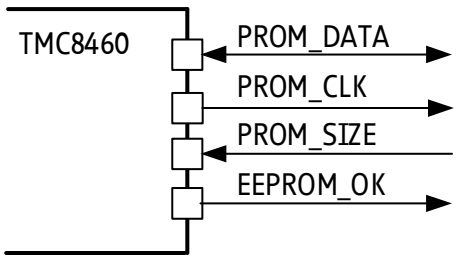


Figure 19 - EEPROM Interface signals

TMC8460 pin	Usage/description
PROM_DATA	I2C data, requires pull-up
PROM_CLK	I2C clock, requires pull-up
PROM_SIZE	Configures EEPROM size
EEPROM_OK	Indicates that EEPROM has be successfully loaded, this may be useful for the local application controller

3.8 Vendor ID, ESC Type, ESC Revision and Build History

Trinamic owns a dedicated EtherCAT vendor ID and holds a specific ID for Trinamic EtherCAT slave controllers to distinguish from other ESC manufacturers.

	hexadecimal	binary	decimal
Trinamic official vendor ID	286	0010 1000 0110	646
Trinamic official ESC type	D0	1101 0000	208
TMC8460 ESC revision at 0x0001	60	0110 0000	96
TMC8461 ESC revision at 0x0001	61	0110 0001	97

Revision Register at 0x0001	Build Register at 0x0002:0x0003	Stepping
0x60	0x0010	TMC8460-BI V10
0x61	0x0010	TMC8461-BI V10

3.9 Electrical Characteristics

3.9.1 Operating Conditions

Stresses beyond those listed in the following table may cause damage to the device. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.
Absolute maximum ratings are stress ratings only. Functional operation of the device at these or

Table 5 : Absolute Maximum Ratings

Symbol	Parameter	Min	Typ	Max	Unit	Notes
VCORE	DC core supply voltage, must always be powered	-0.3	-	1.32	V	
VCCIO	IO supply voltage, must always be powered.	-0.3	-	3.63	V	
VPLL	PLL supply voltage, must always be powered.	-0.3	-	3.63	V	
VPP	Power supply for charge pump, must always be powered.	-0.3	-	3.63	V	
T_{STG}	Storage temperature	-65	-	150	°C	
T_J	Junction temperature	-55	-	125	°C	

Table 6 : Recommended Operating Conditions

Symbol	Parameter	Min	Typ	Max	Unit	Notes
VCORE	DC core supply voltage, must always be powered	1.14	1.2	1.26	V	
VCCIO	IO supply voltage, must always be powered.	3.15	3.3	3.45	V	
VPLL	PLL supply voltage, must always be powered.	3.15	3.3	3.45	V	
VPP	Power supply for charge pump, must always be powered.	3.15	3.3	3.45	V	
T_J	Operating Junction temperature	-40	25	100	°C	

3.9.2 External CLK Source

Both TMC8460-BI and the external Ethernet PHYs must share the same clock source. For proper operation a stable and accurate 25MHz clock source is required. The recommended initial accuracy must be at least 25ppm or better.

The TMC8460-BI has been used with the following crystal oscillators so far:

- FOX Electronics FOX924B TCXO, 25.0MHz, 2.5ppm, 3.3V: http://www.foxonline.com/pdfs/fox924.pdf
- TXC 7M-25.000MAAJ-T XO 25.0MHz, 30ppm: http://www.txccrystal.com/images/pdf/7m.pdf
- CTS 636L5C025M00000, 25MHz, 25ppm: http://www.ctscorp.com/components/Datasheets/008-0250-0.pdf

3.9.3 IO Characteristics

The following table contains information on the IO characteristics.
LVCMOS is a widely used switching standard and is defined by JEDEC (JESD 8-5).
The TMC8460-BI supports LVCMOS standard LVCMOS33, which is a general standard for 3V3 applications.

Symbol	Parameter	Min	Typ	Max	Unit	Notes
C_{IN}	Input capacitance	-		10	pF	
R_{PU}	Weak pull-up at VOH	9.9		14.5	kΩ	
R_{PD}	Weak pull-down at VOL	9.98		14.9	kΩ	
VOH	DC output logic high	VCCIO-0.4		-	V	
VOL	DC output logic low	-		0.4	V	
VIH	DC input logic high	2.0		3.45	V	
VIL	DC input logic low	-0.3		0.8	V	
IIH	Input current high	-		10	µA	
IIL	Input current low	-		10	µA	

3.9.4 Power Consumption

The values given here are typical values only. The real values depend on configuration, activity, and temperature.

Table 7 : TMC8460 power consumption

Symbol	Parameter	Min	Typ	Max	Unit	%	Notes
P_{TOTAL}	Total power consumption	-	237	-	mW	100	P_S + P_D
P_S	Static power consumption	-	19	-	mW	8	
P_D	Dynamic power consumption	-	218	-	mW	92	

Table 8 : Power consumption by rail

Symbol	Parameter	Power (mW)	Voltage (V)	Current (mA)	Notes
Rail VCORE	DC core supply voltage, must always be powered	177.921	1.200	148.268	
Rail VCCIO	IO supply voltage, must always be powered.	46.984	3.300	14.238	
Rail VPLL	PLL supply voltage, must always be powered.	9.000	3.300	2.727	
Rail VPP	Power supply for charge pump, must always be powered.	2.500	2.500	1.000	

3.9.5 Package Thermal Behavior

Dynamic and static power consumption cause the junction temperature of the TMC8460 to be higher than the ambient, case, or board temperature. The equations below show the relationships.

EQ 1: $Theta_{JA} = \frac{T_J - T_A}{P_{Total}}$ EQ 2: $Theta_{JB} = \frac{T_J - T_B}{P_{Total}}$ EQ 3: $Theta_{JC} = \frac{T_J - T_C}{P_{Total}}$

Symbols used:

T_J = Junction temperature
TA = Ambient temperature
TB = Board temperature measured 1.0mm away from the package
TC = Case temperature
$Theta_{JA}$ = Junction-to-ambient thermal resistance
$Theta_{JB}$ = Junction-to-board thermal resistance
$Theta_{JC}$ = Junction-to-case thermal resistance
P_{TOTAL} = Total power consumption

Table 9 : TMC8460 package thermal behavior

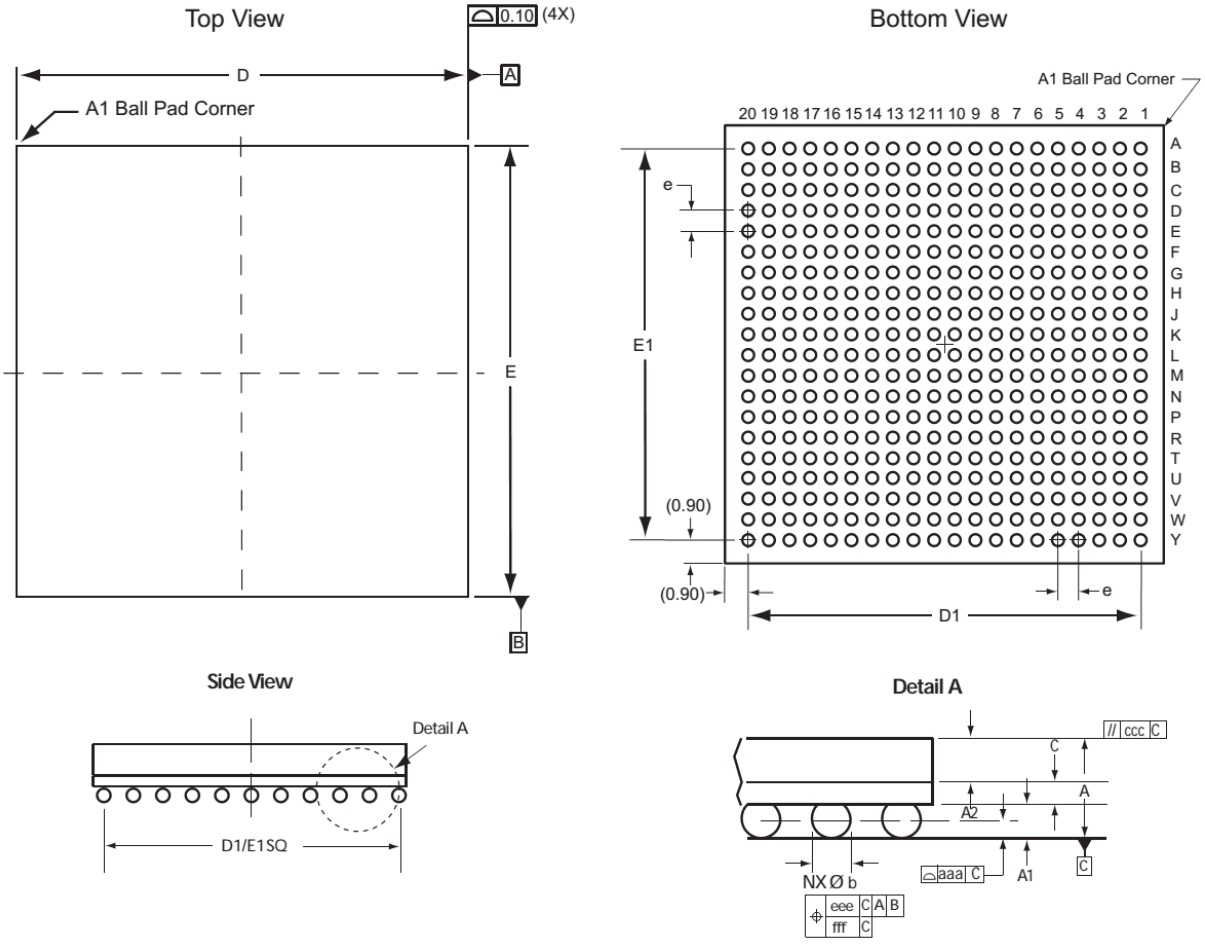
Symbol	Parameter	Typ	Unit	Notes
$Theta_{JA}$	Junction-to-ambient thermal resistance	18.36	°C/W	@ Still air
		14.89	°C/W	@ 1.0 m/s
		13.36	°C/W	@ 2.5 m/s
$Theta_{JB}$	Junction-to-board thermal resistance	7.12	°C/W	
$Theta_{JC}$	Junction-to-case thermal resistance	3.41	°C/W	

3.10 Marking and Order Codes

Type	Code & Marking	Package	Temperature Range	Description	Size [mm²]
TMC8460-BI	TMC8460-BI	VFG400	-40°C ... +100°C	EtherCAT slave controller with enhanced features	17 x 17
TMC8460-EVAL	TMC8460-EVAL	PCB		Evaluation board for TMC8560-BI EtherCAT slave controller	85 x 79
Landungs-bruecke	Landungs-bruecke	PCB		Baseboard for TMC8460-EVAL and further evaluation boards.	85 x 55
Eselsbruecke	Eselsbruecke	PCB		Connector board for plug-in evaluation board system	61 x 38

3.11 Package Dimensions

The TMC8460 comes in a Very Fine Ball Pitch Grid Array package with 400 pins (VFG400).
Ball grid pitch is 0.8mm. Package outline is 17x17mm.
All dimensions are given in mm.



JEDEC Equivalent	VF400		
Dimension	**Min.**	**Nom.**	**Max.**
A	1.31	1.41	1.51
A1	0.32	0.37	0.42
A2	0.65	0.70	0.75
aaa		0.12	
b	0.41	0.46	0.51
c	0.29	0.34	0.39
ccc		0.10	
D/E		17.00 BSC	
D1/E1	–	15.20	–
e		0.80 BSC	
eee		0.15	
fff		0.08	

3.12 Layout Considerations

The required board area for the TMC8460, EEPROM, PHYs, capacitors and LEDs is below 12.5cm² on a six-layer board.
For best soldering results, only the BGA pads should be exposed under the TMC8460, while vias, traces and ground/power planes should be covered by the soldermask. It is not necessary to use blind or buried vias. Use the following guideline for routing to regular vias from the BGA pads. Traces from pads on the outermost rows can also be routed directly on the top layer.

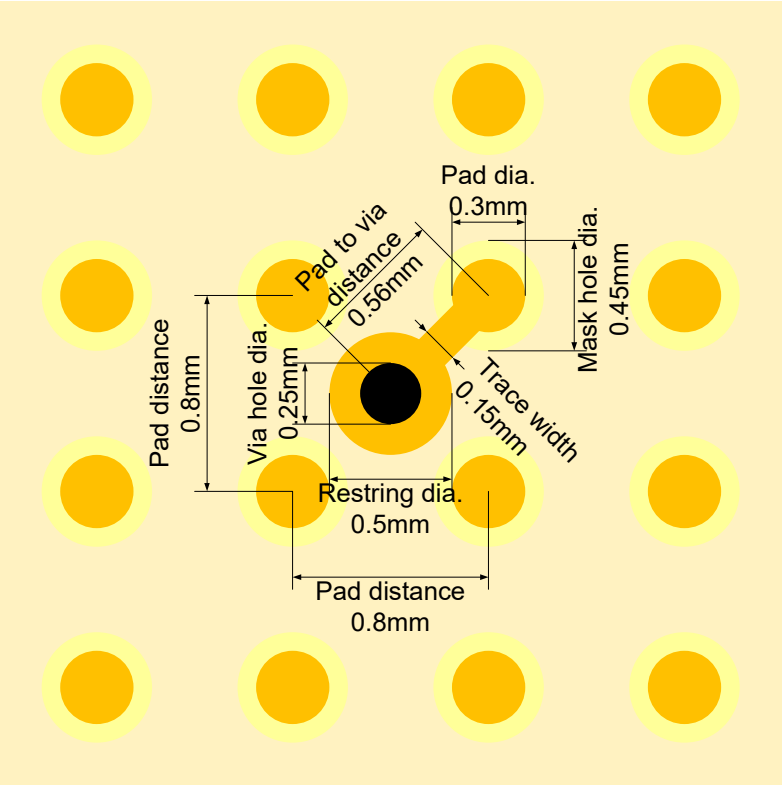


Figure 20 - Recommended land pattern measurements

The routing of the traces to the two PHYs should be as short as possible since these signals are timing critical. Since the pads of the MII signals are all on one side of the TMC8460, most of the signals can be routed only on the top layer without the need for vias.
It is sufficient to treat other traces like SPI or all MFC signals in groups (e.g. the PDI_SPI signals or the PWM outputs), where the variance in trace length is small for all signals of the same group.

3.12.1 Example layout of the TMC8460-Eval

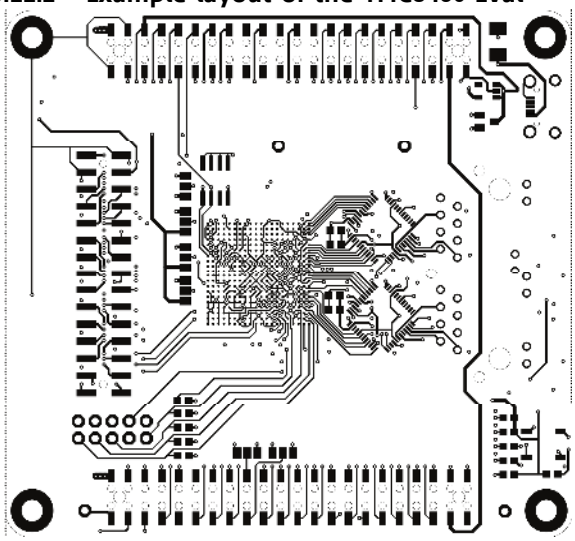


Figure 21 - Top layer (1)

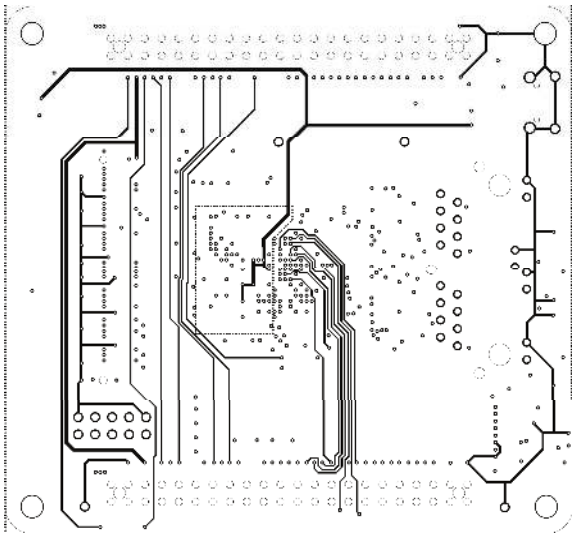


Figure 22 - Inner layer (2)

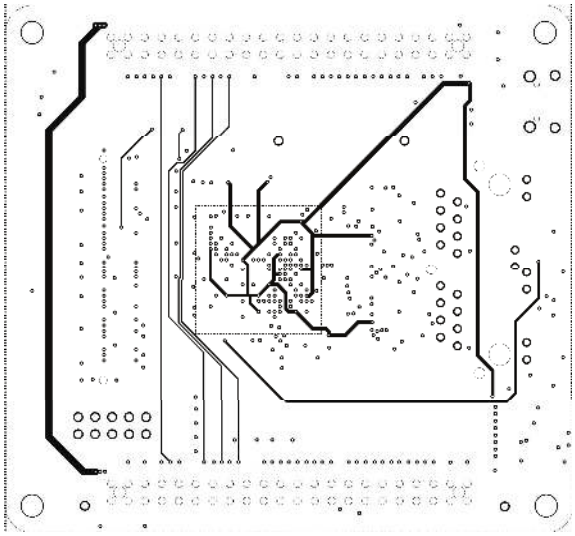


Figure 23 - Inner layer (3)

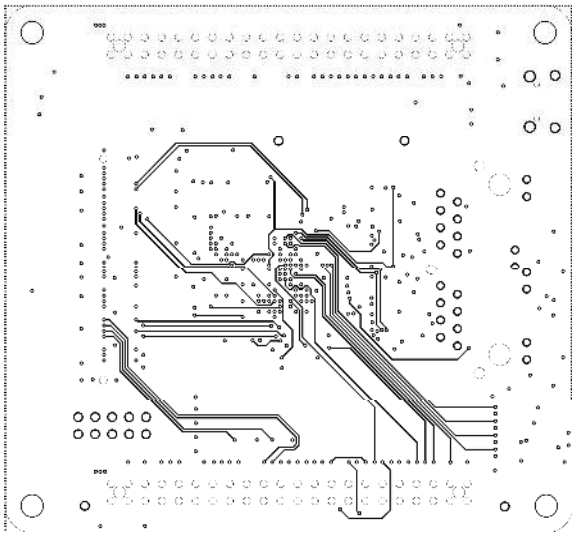


Figure 24 - Inner layer (4)

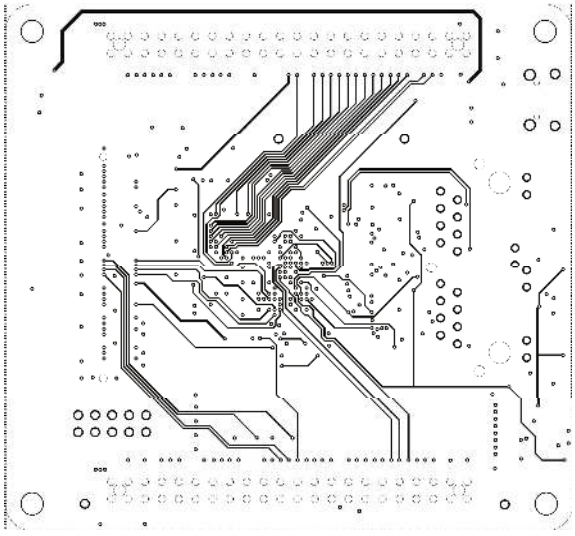


Figure 25 - Inner layer (5)

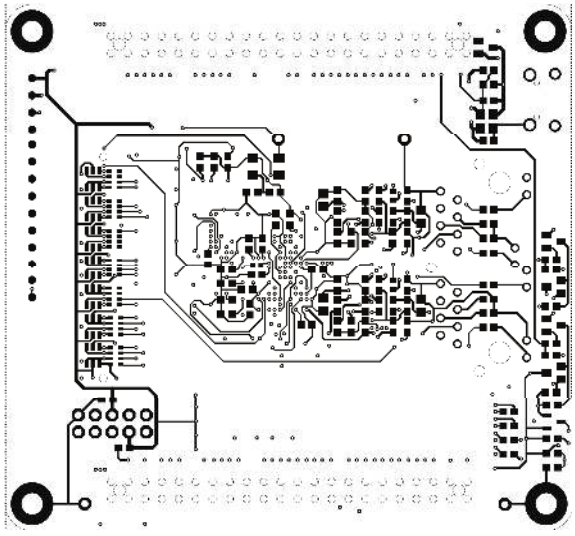


Figure 26 - Bottom layer (6)

3.13 Soldering Profile

The provided reflow soldering profile in accordance to IPC/JEDEC standard J-STD-020 is for reference only. Trinamic advises to optimize to the respective board level parameters to get proper reflow outcome.

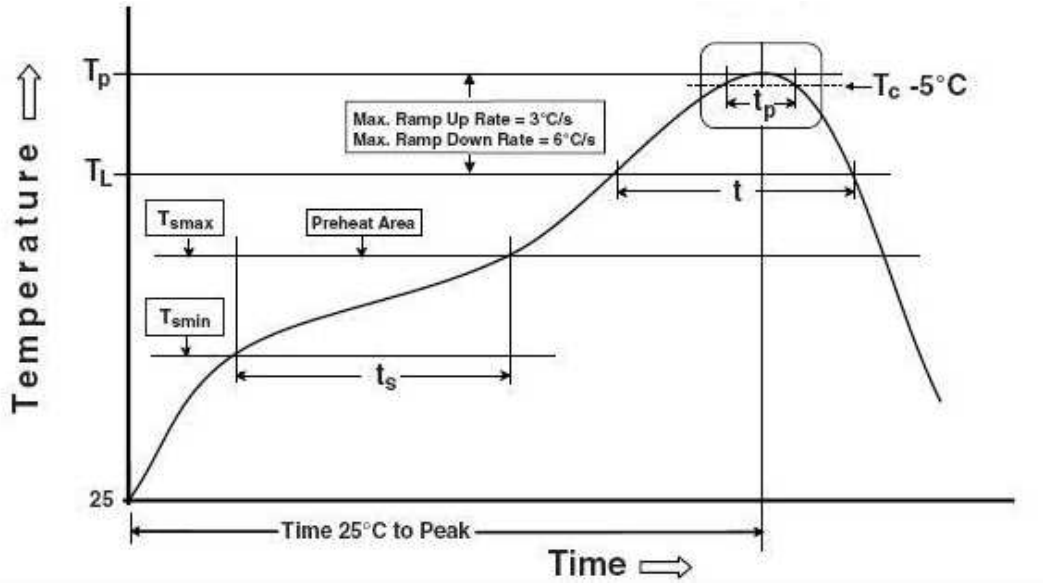


Figure 27 - Soldering Profile

Table 10 : Soldering Profile Parameters

Profile Feature	**Sn-Pb Eutectic Assembly**	**Pb-Free Assembly**
Preheat		
Temperature Min (T_{smin})	100°C	150°C
Temperature Max (T_{smax})	150°C	200°C
Time (min to max) (t_s)	60-120 seconds	60-120 seconds
T_{smax} to T_p		
- Ramp-up Rate	3°C/second max.	3°C/second max.
Time maintained above:		
- Temperature (T_L)	183°C	217°C
- Time (t_L)	60-150 seconds	60-150 seconds
Peak package body temperature (T_p)	235°C	260°C
Time (t_p) within 5°C of classification temperature (T_c)	20 seconds	30 seconds
Ramp-down Rate (T_p to T_{smax})	6°C / second max.	6°C / second max.
Time 25°C to Peak Temperature	6 minutes max.	8 minutes max.

4 EtherCAT Address Space Overview

An EtherCAT Slave Controller (ESC) has an address space of 64Kbyte. The first block of 4Kbyte (0x0000:0x0FFF) is dedicated for ESC- and EtherCAT-relevant configuration and status registers. The PD-RAM starts at address 0x1000. The TMC8460-BI has a PD-RAM size of 16 Kbyte.

The following table lists the standard ESC register set which is also available in the TMC8460-BI.
Section 5 contains a detailed register description.
Section 0 lists the specific enabled EtherCAT features of the TMC8460-BI.

Table 11 : TMC8460 address space

Address[1]	Length (Byte)	Description
		ESC Information
0x0000	1	Type
0x0001	1	Revision
0x0002:0x0003	2	Build
0x0004	1	FMMUs supported
0x0005	1	SyncManagers supported
0x0006	1	RAM Size
0x0007	1	Port Descriptor
0x0008:0x0009	2	ESC Features supported
		Station Address
0x0010:0x0011	2	Configured Station Address
0x0012:0x0013	2	Configured Station Alias
		Write Protection
0x0020	1	Write Register Enable
0x0021	1	Write Register Protection
0x0030	1	ESC Write Enable
0x0031	1	ESC Write Protection
		Data Link Layer
0x0040	1	ESC Reset ECAT
0x0041	1	ESC Reset PDI
0x0100:0x0103	4	ESC DL Control
0x0108:0x0109	2	Physical Read/Write Offset
0x0110:0x0111	2	ESC DL Status
		Application Layer
0x0120:0x0121	2	AL Control
0x0130:0x0131	2	AL Status
0x0134:0x0135	2	AL Status Code
0x0138	1	RUN LED Override
0x0139	1	ERR LED Override

[1] Address areas not listed here are reserved. They are not writable. Read data from reserved addresses has to be ignored. Reserved addresses must not be written.

Address[1]	Length (Byte)	Description
		PDI / ESC Configuration
0x0140	1	PDI Control
0x0141	1	ESC Configuration
0x014E:0x014F	2	PDI Information
0x0150	1	PDI Configuration
0x0151	1	SYNC/LATCH[1:0] PDI Configuration
0x0152:0x0153	2	Extended PDI Configuration
		Interrupts
0x0200:0x0201	2	ECAT Event Mask
0x0204:0x0207	4	PDI AL Event Mask
0x0210:0x0211	2	ECAT Event Request
0x0220:0x0223	4	AL Event Request
		Error Counters
0x0300:0x0307	4x2	Rx Error Counter[3:0]
0x0308:0x030B	4x1	Forwarded Rx Error counter[3:0]
0x030C	1	ECAT Processing Unit Error Counter
0x030D	1	PDI Error Counter
0x030E	1	PDI Error Code
0x0310:0x0313	4x1	Lost Link Counter[3:0]
		Watchdogs
0x0400:0x0401	2	Watchdog Divider
0x0410:0x0411	2	Watchdog Time PDI
0x0420:0x0421	2	Watchdog Time Process Data
0x0440:0x0441	2	Watchdog Status Process Data
0x0442	1	Watchdog Counter Process Data
0x0443	1	Watchdog Counter PDI
		SII EEPROM Interface
0x0500	1	EEPROM Configuration
0x0501	1	EEPROM PDI Access State
0x0502:0x0503	2	EEPROM Control/Status
0x0504:0x0507	4	EEPROM Address
0x0508:0x050F	4/8	EEPROM Data
		MII Management Interface
0x0510:0x0511	2	MII Management Control/Status
0x0512	1	PHY Address
0x0513	1	PHY Register Address
0x0514:0x0515	2	PHY Data
0x0516	1	MII Management ECAT Access State
0x0517	1	MII Management PDI Access State

Address[1]	Length (Byte)	Description
0x0518:0x051B	4	PHY Port Status
0x0580:0x05FF	**128**	**ESC Parameter Ram**
0x0580:0x05AB	43	TMC8460 MFCIO block configuration Can be written at start-up from EEPROM configuration (with category 1) or by ECAT or PDI at runtime.
0x0600:0x06FF	**16x16**	**FMMU[15:0]**
+0x0:0x3	4	Logical Start Address
+0x4:0x5	2	Length
+0x6	1	Logical Start bit
+0x7	1	Logical Stop bit
+0x8:0x9	2	Physical Start Address
+0xA	1	Physical Start bit
+0xB	1	Type
+0xC	1	Activate
+0xD:0xF	3	Reserved

Address[1]	Length (Byte)	Description
0x0800:0x087F	**16x2**	**SyncManager[15:0]**
+0x0:0x1	2	Physical Start Address
+0x2:0x3	2	Length
+0x4	1	Control Register
+0x5	1	Status Register
+0x6	1	Activate
+0x7	1	PDI Control
0x0900:0x09FF		**Distributed Clocks (DC)**
		DC – Receive Times
0x0900:0x0903	4	Receive Time Port 0
0x0904:0x0907	4	Receive Time Port 1
0x0908:0x090B	4	Receive Time Port 2
0x090C:0x090F	4	Receive Time Port 3
		DC – Time Loop Control Unit
0x0910:0x0917	4/8	System Time
0x0918:0x091F	4/8	Receive Time ECAT Processing Unit
0x0920:0x0927	4/8	System Time Offset
0x0928:0x092B	4	System Time Delay
0x092C:0x092F	4	System Time Difference
0x0930:0x0931	2	Speed Counter Start
0x0932:0x0933	2	Speed Counter Diff
0x0934	1	System Time Difference Filter Depth
0x0935	1	Speed Counter Filter Depth
0x0936	1	Receive Time Latch Mode
		DC – Cyclic Unit Control
0x0980	1	Cyclic Unit Control
		DC – SYNC Out Unit
0x0981	1	Activation
0x0982:0x0983	2	Pulse Length of SyncSignals
0x0984	1	Activation Status
0x098E	1	SYNC0 Status
0x098F	1	SYNC1 Status
0x0990:0x0997	4/8	Start Time Cyclic Operation/Next SYNC0 Pulse
0x0998:0x099F	4/8	Next SYNC1 Pulse
0x09A0:0x09A3	4	SYNC0 Cycle Time
0x09A4:0x09A7	4	SYNC1 Cycle Time
		DC – Latch In Unit
0x09A8	1	Latch0 Control
0x09A9	1	Latch1 Control
0x09AE	1	Latch0 Status

Address[1]	Length (Byte)	Description
0x09AF	1	Latch1 Status
0x09B0:0x09B7	4/8	Latch0 Time Positive Edge
0x09B8:0x09BF	4/8	Latch0 Time Negative Edge
0x09C0:0x09C7	4/8	Latch1 Time Positive Edge
0x09C8:0x09CF	4/8	Latch1 Time Negative Edge
		DC – SyncManager Event Times
0x09F0:0x09F3	4	EtherCAT Buffer Change Event Time
0x09F8:0x09FB	4	PDI Buffer Start Event Time
0x09FC:0x09FF	4	PDI Buffer Change Event Time
		ESC specific
0x0E00:0x0EFF	256	TMC8460 Product and Vendor ID
		User RAM/Extended ESC features
0x0F80:0x0FFF	128	User RAM
0x0F80:0x0FA0	33	Extended ESC features
		Process Data RAM
0x1000:0x4FFF	16 KB	Process Data RAM
0x4000:0x4056	87	TMC8460 MFCIO Memory Block 0 for ECAT Write Data
0x4800:0x481C	29	TMC8460 MFCIO Memory Block 1 for ECAT Read Data

For Registers longer than one byte, the LSB has the lowest and MSB the highest address.

5 EtherCAT Register Description

5.1 Type (0x0000)

Table 12: Register Type (0x0000)

Bit	Description	ECAT	PDI	Reset Value
7:0	Type of EtherCAT controller	r/-	r/-	Trinamic ESC Type: 0xD0

5.2 Revision (0x0001)

Table 13: Register Revision (0x0001)

Bit	Description	ECAT	PDI	Reset Value
7:0	Revision of EtherCAT controller.	r/-	r/-	TMC8460: 0x60 TMC8461: 0x61

5.3 Build (0x0002:0x0003)

Table 14: Register Build (0x0002:0x0003)

Bit	Description	ECAT	PDI	Reset Value
15:0	Actual build of EtherCAT controller.	r/-	r/-	TMC8460: 0x02 = Version 1.0 TMC8461: 0x10 = Version 1.0

5.4 FMMUs supported (0x0004)

Table 15: Register FMMUs supported (0x0004)

Bit	Description	ECAT	PDI	Reset Value
7:0	Number of supported FMMU channels (or entities) of the EtherCAT Slave Controller.	r/-	r/-	TMC8460: 6 TMC8461: 8

5.5 SyncManagers supported (0x0005)

Table 16: Register SyncManagers supported (0x0005)

Bit	Description	ECAT	PDI	Reset Value
7:0	Number of supported SyncManager channels (or entities) of the EtherCAT Slave Controller	r/-	r/-	TMC8460: 6 TMC8461: 8

5.6 RAM Size (0x0006)

Table 17: Register RAM Size (0x0006)

Bit	Description	ECAT	PDI	Reset Value
7:0	Process Data RAM size supported by the EtherCAT Slave Controller in KByte	r/-	r/-	TMC8460: 16 TMC8461: 16

5.7 Port Descriptor (0x0007)

Table 18: Register Port Descriptor (0x0007)

Bit	Description	ECAT	PDI	Reset Value
	Port configuration: 00: Not implemented 01: Not configured (SII EEPROM) 10: EBUS 11: MII / RMII / RGMII			
1:0	Port 0	r/-	r/-	TMC8460: 11 TMC8461: 11
3:2	Port 1	r/-	r/-	TMC8460: 11 TMC8461: 11
5:4	Port 2	r/-	r/-	TMC8460: 00 TMC8461: 00
7:6	Port 3	r/-	r/-	TMC8460: 00 TMC8461: 11

5.8 ESC Features supported (0x0008:0x0009)

Table 19: Register ESC Features supported (0x0008:0x0009)

Bit	Description	ECAT	PDI	Reset Value
0	FMMU Operation: 0: Bit oriented 1: Byte oriented	r/-	r/-	TMC8460: 0
1	Reserved	r/-	r/-	TMC8460: 0
2	Distributed Clocks: 0: Not available 1: Available	r/-	r/-	TMC8460: 1
3	Distributed Clocks (width): 0: 32 bit 1: 64 bit	r/-	r/-	TMC8460: 1
4	Low Jitter EBUS: 0: Not available, standard jitter 1: Available, jitter minimized	r/-	r/-	TMC8460: 0
5	Enhanced Link Detection EBUS: 0: Not available 1: Available	r/-	r/-	TMC8460: 0
6	Enhanced Link Detection MII: 0: Not available 1: Available	r/-	r/-	TMC8460: 1

Bit	Description	ECAT	PDI	Reset Value
7	Separate Handling of FCS Errors: 0: Not supported 1: Supported, frames with wrong FCS and additional nibble will be counted separately in Forwarded RX Error Counter	r/-	r/-	TMC8460: 1
8	Enhanced DC SYNC Activation 0: Not available 1: Available NOTE: This feature refers to registers 0x981[7:3], 0x0984	r/-	r/-	TMC8460: 1
9	EtherCAT LRW command support: 0: Supported 1: Not supported	r/-	r/-	TMC8460: 0
10	EtherCAT read/write command support (BRW, APRW, FPRW): 0: Supported 1: Not supported	r/-	r/-	TMC8460: 0
11	Fixed FMMU/SyncManager configuration 0: Variable configuration 1: Fixed configuration (refer to documentation of supporting ESCs)	r/-	r/-	TMC8460: 0
15:12	Reserved	r/-	r/-	TMC8460: 0

5.9 Configured Station Address (0x0010:0x0011)

Table 20: Register Configured Station Address (0x0010:0x0011)

Bit	Description	ECAT	PDI	Reset Value
15:0	Address used for node addressing (FPxx commands)	r/w	r/-	0

5.10 Configured Station Alias (0x0012:0x0013)

Table 21: Register Configured Station Alias (0x0012:0x0013)

Bit	Description	ECAT	PDI	Reset Value
15:0	Alias Address used for node addressing (FPxx commands). The use of this alias is activated by Register DL Control Bit 24 (0x0100.24/0x0103.0) NOTE: EEPROM value is only taken over at first EEPROM load after power-on or reset.	r/-	r/w	0 until first EEPROM load, then EEPROM ADR 0x0004

5.11 Write Register Enable (0x0020)

This register/function is not available with TMC8460-BI.

Table 22: Register Write Register Enable (0x0020)

Bit	Description	ECAT	PDI	Reset Value
0	If write register protection is enabled, this register has to be written in the same Ethernet frame (value does not care) before other writes to this station are allowed. Write protection is still active after this frame (if Write Register Protection register is not changed).	r/w	r/-	0
7:1	Reserved, write 0	r/-	r/-	0

5.12 Write Register Protection (0x0021)

This register/function is not available with TMC8460-BI.

Table 23: Register Write Register Protection (0x0021)

Bit	Description	ECAT	PDI	Reset Value
0	Write register protection: 0: Protection disabled 1: Protection enabled Registers 0x0000-0x0137, 0x013A-0x0F0F are write protected, except for 0x0030.	r/w	r/-	0
7:1	Reserved, write 0	r/-	r/-	0

5.13 ESC Write Enable (0x0030)

This register/function is not available with TMC8460-BI.

Table 24: Register ESC Write Enable (0x0030)

Bit	Description	ECAT	PDI	Reset Value
0	If ESC write protection is enabled, this register has to be written in the same Ethernet frame (value does not care) before other writes to this station are allowed. ESC write protection is still active after this frame (if ESC Write Protection register is not changed).	r/w	r/-	0
7:1	Reserved, write 0	r/-	r/-	0

5.14 ESC Write Protection (0x0031)

This register/function is not available with TMC8460-BI.

Table 25: Register ESC Write Protection (0x0031)

Bit	Description	ECAT	PDI	Reset Value
0	Write protect: 0: Protection disabled 1: Protection enabled All areas are write protected, except for 0x0030.	r/w	r/-	0
7:1	Reserved, write 0	r/-	r/-	0

5.15 ESC Reset ECAT (0x0040)

Table 26: Register ESC Reset ECAT (0x0040)

Bit	Description	ECAT	PDI	Reset Value
Write				
7:0	A reset is asserted after writing 0x52 ('R'), 0x45 ('E') and 0x53 ('S') in this register with 3 consecutive frames.	r/w	r/-	0
Read				
1:0	Progress of the reset procedure: 01: after writing 0x52 10: after writing 0x45 (if 0x52 was written before) 00: else	r/w	r/-	00
7:2	Reserved, write 0	r/-	r/-	0

5.16 ESC Reset PDI (0x0041)

Table 27: Register ESC Reset PDI (0x0041)

Bit	Description	ECAT	PDI	Reset Value
Write				
7:0	A reset is asserted after writing 0x52 ('R'), 0x45 ('E') and 0x53 ('S') in this register with 3 consecutive commands.	r/-	r/w	0
Read				
1:0	Progress of the reset procedure: 01: after writing 0x52 10: after writing 0x45 (if 0x52 was written before) 00: else	r/-	r/w	00
7:2	Reserved, write 0	r/-	r/-	0

5.17 ESC DL Control (0x0100:0x0103)

Table 28: Register ESC DL Control (0x0100:0x0103)

Bit	Description	ECAT	PDI	Reset Value
0	Forwarding rule: 0: EtherCAT frames are processed, Non-EtherCAT frames are forwarded without processing 1: EtherCAT frames are processed, Non-EtherCAT frames are destroyed The source MAC address is changed for every frame (SOURCE_MAC[1] is set to 1 – locally administered address) regardless of the forwarding rule.	r/w	r/-	1
1	Temporary use of settings in Register 0x101: 0: permanent use 1: use for about 1 second, then revert to previous settings	r/w	r/-	0
7:2	Reserved, write 0	r/-	r/-	0
9:8	Loop Port 0: 00: Auto 01: Auto Close 10: Open 11: Closed NOTE: Loop open means sending/receiving over this port is enabled, loop closed means sending/receiving is disabled and frames are forwarded to the next open port internally. Auto: loop closed at link down, opened at link up Auto Close: loop closed at link down, opened with writing 01 again after link up (or receiving a valid Ethernet frame at the closed port) Open: loop open regardless of link state Closed: loop closed regardless of link state	r/w*	r/-	00
11:10	Loop Port 1: 00: Auto 01: Auto Close 10: Open 11: Closed	r/w*	r/-	00

Bit	Description	ECAT	PDI	Reset Value
13:12	Loop Port 2: 00: Auto 01: Auto Close 10: Open 11: Closed	r/w*	r/-	00
15:14	Loop Port 3: 00: Auto 01: Auto Close 10: Open 11: Closed	r/w*	r/-	00
18:16	RX FIFO Size (ESC delays start of forwarding until FIFO is at least half full). RX FIFO Size/RX delay reduction** : Value: EBUS: MII: 0: -50 ns -40 ns 1: -40 ns -40 ns 2: -30 ns -40 ns 3: -20 ns -40 ns 4: -10 ns no change 5: no change no change 6: no change no change 7: default default NOTE: EEPROM value is only taken over at first EEPROM load after power-on or reset	r/w	r/-	7, later EEPROM ADR 0x0005[11:9] inverted
19	EBUS Low Jitter: 0: Normal jitter 1: Reduced jitter	r/w	r/-	0
21:20	Reserved, write 0	r/w	r/-	0, later EEPROM ADR 0x0005[5:4]
22	EBUS remote link down signaling time: 0: Default (~660 ms) 1: Reduced (~80 µs)	r/w	r/-	0, later EEPROM ADR 0x0005[6]
23	Reserved, write 0	r/w	r/-	0, later EEPROM ADR 0x0005[7]
24	Station alias: 0: Ignore Station Alias 1: Alias can be used for all configured address command types (FPRD, FPWR, ...)	r/w	r/-	0
31:25	Reserved, write 0	r/-	r/-	0

* Loop configuration changes are delayed until the end of a currently received or transmitted frame at the port.

** The possibility of RX FIFO Size reduction depends on the clock source accuracy of the ESC and of every connected EtherCAT/Ethernet devices (master, slave, etc.). RX FIFO Size of 7 is sufficient for 100ppm accuracy, FIFO Size 0 is possible with 25ppm accuracy (frame size of 1518/1522 Byte).

5.18 Physical Read/Write Offset (0x0108:0x0109)

This register/function is not available with TMC8460-BI.

Table 29: Register Physical Read/Write Offset (0x0108:0x0109)

Bit	Description	ECAT	PDI	Reset Value
15:0	Offset of R/W Commands (FPRW, APRW) between Read address and Write address. RD_ADR = ADR and WR_ADR = ADR + R/W-Offset	r/w	r/-	0

5.19 ESC DL Status (0x0110:0x0111)

Table 30: Register ESC DL Status (0x0110:0x0111)

Bit	Description	ECAT	PDI	Reset Value
0	PDI operational/EEPROM loaded correctly: 0: EEPROM not loaded, PDI not operational (no access to Process Data RAM) 1: EEPROM loaded correctly, PDI operational (access to Process Data RAM)	r*/-	r/-	0
1	PDI Watchdog Status: 0: Watchdog expired 1: Watchdog reloaded	r*/-	r/-	0
2	Enhanced Link detection: 0: Deactivated for all ports 1: Activated for at least one port NOTE: EEPROM value is only taken over at first EEPROM load after power-on or reset	r*/-	r/-	TMC8460: 1 until first EEPROM load, then EEPROM ADR 0x0000.9 or 0x0000[15:12]
3	Reserved	r*/-	r/-	0
4	Physical link on Port 0: 0: No link 1: Link detected	r*/-	r/-	0
5	Physical link on Port 1: 0: No link 1: Link detected	r*/-	r/-	0
6	Physical link on Port 2: 0: No link 1: Link detected	r*/-	r/-	0
7	Physical link on Port 3: 0: No link 1: Link detected	r*/-	r/-	0
8	Loop Port 0: 0: Open 1: Closed	r*/-	r/-	0
9	Communication on Port 0: 0: No stable communication 1: Communication established	r*/-	r/-	0
10	Loop Port 1: 0: Open 1: Closed	r*/-	r/-	0
11	Communication on Port 1: 0: No stable communication 1: Communication established	r*/-	r/-	0

Bit	Description	ECAT	PDI	Reset Value
12	Loop Port 2: 0: Open 1: Closed	r*/-	r/-	0
13	Communication on Port 2: 0: No stable communication 1: Communication established	r*/-	r/-	0
14	Loop Port 3: 0: Open 1: Closed	r*/-	r/-	0
15	Communication on Port 3: 0: No stable communication 1: Communication established	r*/-	r/-	0

* Reading DL Status register from ECAT clears ECAT Event Request 0x0210[2].

Table 31: Decoding port state in ESC DL Status register 0x0111 (typical modes only)

Register 0x0111	Port 3 (not supported by TMC8460)	Port 2 (not supported by TMC8460)	Port 1	Port 0
0x55	No link, closed	No link, closed	No link, closed	No link, closed
0x56	No link, closed	No link, closed	No link, closed	Link, open
0x59	No link, closed	No link, closed	Link, open	No link, closed
0x5A	No link, closed	No link, closed	Link, open	Link, open
0x65	No link, closed	Link, open	No link, closed	No link, closed
0x66	No link, closed	Link, open	No link, closed	Link, open
0x69	No link, closed	Link, open	Link, open	No link, closed
0x6A	No link, closed	Link, open	Link, open	Link, open
0x95	Link, open	No link, closed	No link, closed	No link, closed
0x96	Link, open	No link, closed	No link, closed	Link, open
0x99	Link, open	No link, closed	Link, open	No link, closed
0x9A	Link, open	No link, closed	Link, open	Link, open
0xA5	Link, open	Link, open	No link, closed	No link, closed
0xA6	Link, open	Link, open	No link, closed	Link, open
0xA9	Link, open	Link, open	Link, open	No link, closed
0xAA	Link, open	Link, open	Link, open	Link, open
0xD5	Link, closed	No link, closed	No link, closed	No link, closed
0xD6	Link, closed	No link, closed	No link, closed	Link, open
0xD9	Link, closed	No link, closed	Link, open	No link, closed
0xDA	Link, closed	No link, closed	Link, open	Link, open

5.20 AL Control (0x0120:0x0121)

Table 32: Register AL Control (0x0120:0x0121)

Bit	Description	ECAT	PDI	Reset Value
3:0	Initiate State Transition of the Device State Machine: 1: Request Init State 3: Request Bootstrap State 2: Request Pre-Operational State 4: Request Safe-Operational State 8: Request Operational State	r/(w)	r/ (w ack)*	1
4	Error Ind Ack: 0: No Ack of Error Ind in AL status register 1: Ack of Error Ind in AL status register	r/(w)	r/ (w ack)*	0
5	Device Identification: 0: No request 1: Device Identification request	r/(w)	r/ (w ack)*	0
15:6	Reserved, write 0	r/(w)	r/ (w ack)*	0

NOTE: AL Control register behaves like a mailbox if Device Emulation is off (0x0140.8=0): The PDI has to read/write* the AL Control register after ECAT has written it. Otherwise ECAT cannot write again to the AL Control register. After Reset, AL Control register can be written by ECAT. (Regarding mailbox functionality, both registers 0x0120 and 0x0121 are equivalent, e.g. reading 0x0121 is sufficient to make this register writeable again.)
If Device Emulation is on, the AL Control register can always be written, its content is copied to the AL Status register.

* PDI register function acknowledge by Write command is disabled: Reading AL Control from PDI clears AL Event Request 0x0220[0]. Writing to this register from PDI is not possible.
PDI register function acknowledge by Write command is enabled: Writing AL Control from PDI clears AL Event Request 0x0220[0]. Writing to this register from PDI is possible; write value is ignored (write 0).

5.21 AL Status (0x0130:0x0131)

Table 33: Register AL Status (0x0130:0x0131)

Bit	Description	ECAT	PDI	Reset Value
3:0	Actual State of the Device State Machine: 1: Init State 3: Request Bootstrap State 2: Pre-Operational State 4: Safe-Operational State 8: Operational State	r*/-	r/(w)	1
4	Error Ind: 0: Device is in State as requested or Flag cleared by command 1: Device has not entered requested State or changed State as result of a local action	r*/-	r/(w)	0
5	Device Identification: 0: Device Identification not valid 1: Device Identification loaded	r*/-	r/(w)	0
15:6	Reserved, write 0	r*/-	r/(w)	0

NOTE: AL Status register is only writable from PDI if Device Emulation is off (0x0140.8=0), otherwise AL Status register will reflect AL Control register values.

* Reading AL Status from ECAT clears ECAT Event Request 0x0210[3].

5.22 AL Status Code (0x0134:0x0135)

Table 34: Register AL Status Code (0x0134:0x0135)

Bit	Description	ECAT	PDI	Reset Value
15:0	AL Status Code	r/-	r/w	0

5.23 RUN LED Override (0x0138)

This register/function is not available with TMC8460-BI.

Table 35: Register RUN LED Override (0x0138)

Bit	Description	ECAT	PDI	Reset Value
3:0	LED code: (FSM State☺) 0x0: Off (1-Init) 0x1-0xC: Flash 1x – 12x (4-SafeOp 1x) 0xD: Blinking (2-PreOp) 0xE: Flickering (3-Bootrap) 0xF: On (8-Op)	r/w	r/w	0
4	Enable Override: 0: Override disabled 1: Override enabled	r/w	r/w	0
7:5	Reserved, write 0	r/w	r/w	0

NOTE: Changes to AL Status register (0x0130) with valid values will disable RUN LED Override (0x0138[4]=0). The value read in this register always reflects current LED output.

5.24 ERR LED Override (0x0139)

This register/function is not available with TMC8460-BI.

Table 36: Register ERR LED Override (0x0139)

Bit	Description	ECAT	PDI	Reset Value
3:0	LED code: 0x0: Off 0x1-0xC: Flash 1x – 12x 0xD: Blinking 0xE: Flickering 0xF: On	r/w	r/w	0
4	Enable Override: 0: Override disabled 1: Override enabled	r/w	r/w	0
7:5	Reserved, write 0	r/w	r/w	0

NOTE: New error conditions will disable ERR LED Override (0x0139[4]=0). The value read in this register always reflects current LED output.

5.25 PDI Control (0x0140)

Table 37: Register PDI Control (0x0140)

Bit	Description	ECAT	PDI	Reset Value
7:0	Process data interface: 0x00: Interface deactivated (no PDI) 0x01: 4 Digital Input 0x02: 4 Digital Output 0x03: 2 Digital Input and 2 Digital Output 0x04: Digital I/O **0x05: SPI Slave** 0x06: Oversampling I/O 0x07: EtherCAT Bridge (port 3) 0x08: 16 Bit asynchronous Microcontroller interface 0x09: 8 Bit asynchronous Microcontroller interface 0x0A: 16 Bit synchronous Microcontroller interface 0x0B: 8 Bit synchronous Microcontroller interface 0x10: 32 Digital Input and 0 Digital Output 0x11: 24 Digital Input and 8 Digital Output 0x12: 16 Digital Input and 16 Digital Output 0x13: 8 Digital Input and 24 Digital Output 0x14: 0 Digital Input and 32 Digital Output 0x80: On-chip bus Others: Reserved	r/-	r/-	TMC8460: 0x05, later EEPROM ADR 0x0000

5.26 ESC Configuration (0x0141)

Table 38: Register ESC Configuration (0x0141)

Bit	Description	ECAT	PDI	Reset Value
0	Device emulation (control of AL status): 0: AL status register has to be set by PDI 1: AL status register will be set to value written to AL control register	r/-	r/-	TMC8460: 1 with PDI_EMULATION pin, later EEPROM ADR 0x0000
1	Enhanced Link detection all ports: 0: disabled (if bits [7:4]=0) 1: enabled at all ports (overrides bits [7:4])	r/-	r/-	0, later EEPROM ADR 0x0000
2	Distributed Clocks SYNC Out Unit: 0: disabled (power saving) 1: enabled	r/-	r/-	TMC8460: Depends on configuration later EEPROM ADR 0x0000
3	Distributed Clocks Latch In Unit: 0: disabled (power saving) 1: enabled	r/-	r/-	
4	Enhanced Link port 0: 0: disabled (if bit 1=0) 1: enabled	r/-	r/-	0, later EEPROM ADR 0x0000
5	Enhanced Link port 1: 0: disabled (if bit 1=0) 1: enabled	r/-	r/-	
6	Enhanced Link port 2: 0: disabled (if bit 1=0) 1: enabled	r/-	r/-	

Bit	Description	ECAT	PDI	Reset Value
7	Enhanced Link port 3: 0: disabled (if bit 1=0) 1: enabled	r/-	r/-	

5.27 PDI Information (0x014E:0x014F)

Table 39: Register PDI Information (0x014E:0x014F)

Bit	Description	ECAT	PDI	Reset Value
0	PDI register function acknowledge by write: 0: Disabled 1: Enabled	r/-	r/-	TMC8460: Depends on configuration
1	PDI configured: 0: PDI not configured 1: PDI configured (EEPROM loaded)	r/-	r/-	0
2	PDI active: 0: PDI not active 1: PDI active	r/-	r/-	
3	PDI configuration invalid: 0: PDI configuration ok 1: PDI configuration invalid	r/-	r/-	
7:4	Reserved	r/-	r/-	

5.28 PDI Configuration (0x0150:0x0153)

The PDI configuration register 0x0150 and the extended PDI configuration registers 0x0152:0x0153 depend on the selected PDI.

The Sync/Latch[1:0] PDI configuration register 0x0151 is independent of the selected PDI.

The TMC8460-BI only supports SPI PDI. Thus the PDI number is always 0x05.

5.28.1 PDI SPI Slave Configuration

Table 40: Register PDI SPI Slave Configuration (0x0150)

Bit	Description	ECAT	PDI	Reset Value
1:0	SPI mode: 00: SPI mode 0 01: SPI mode 1 10: SPI mode 2 11: SPI mode 3 NOTE: SPI mode 3 is recommended for Slave Sample Code NOTE: SPI status flag is not available in SPI modes 0 and 2 with normal data out sample.	r/-	r/-	TMC8460: 3
3:2	SPI_IRQ output driver/polarity: 00: Push-Pull active low 01: Open Drain (active low) 10: Push-Pull active high 11: Open Source (active high)	r/-	r/-	
4	SPI_SEL polarity: 0: Active low 1: Active high	r/-	r/-	
5	Data Out sample mode: 0: Normal sample (SPI_DO and SPI_DI are sampled at the same SPI_CLK edge) 1: Late sample (SPI_DO and SPI_DI are sampled at different SPI_CLK edges)	r/-	r/-	
7:6	Reserved, set EEPROM value 0	r/-	r/-	

Table 41: Register PDI SPI Slave extended configuration (0x0152:0x0153)

Bit	Description	ECAT	PDI	Reset Value
15:0	Reserved, set EEPROM value 0	r/-	r/-	TMC8460: 0

5.28.2 Sync/Latch[1:0] PDI Configuration

Table 42: Register Sync/Latch[1:0] PDI Configuration (0x0151)

Bit	Description	ECAT	PDI	Reset Value
1:0	SYNC0 output driver/polarity: 00: Push-Pull active low 01: Open Drain (active low) 10: Push-Pull active high 11: Open Source (active high)	r/-	r/-	TMC8460: 10
2	SYNC0/LATCH0 configuration*: 0: LATCH0 Input 1: SYNC0 Output	r/-	r/-	TMC8460: 1
3	SYNC0 mapped to AL Event Request register 0x0220.2: 0: Disabled 1: Enabled	r/-	r/-	TMC8460: Depends on configuration
5:4	SYNC1 output driver/polarity: 00: Push-Pull active low 01: Open Drain (active low) 10: Push-Pull active high 11: Open Source (active high)	r/-	r/-	TMC8460: 10
6	SYNC1/LATCH1 configuration*: 0: LATCH1 input 1: SYNC1 output	r/-	r/-	TMC8460: 1
7	SYNC1 mapped to AL Event Request register 0x0220.3: 0: Disabled 1: Enabled	r/-	r/-	TMC8460: Depends on configuration

* The TMC8460 has concurrent SYNC[1:0] outputs and LATCH[1:0] inputs, independent of this configuration.

5.29 ECAT Event Mask (0x0200:0x0201)

Table 43: Register ECAT Event Mask (0x0200:0x0201)

Bit	Description	ECAT	PDI	Reset Value
15:0	ECAT Event masking of the ECAT Event Request Events for mapping into ECAT event field of EtherCAT frames: 0: Corresponding ECAT Event Request register bit is not mapped 1: Corresponding ECAT Event Request register bit is mapped	r/w	r/-	0

5.30 PDI AL Event Mask (0x0204:0x0207)

Table 44: Register PDI AL Event Mask (0x0204:0x0207)

Bit	Description	ECAT	PDI	Reset Value
31:0	AL Event masking of the AL Event Request register Events for mapping to PDI IRQ signal: 0: Corresponding AL Event Request register bit is not mapped 1: Corresponding AL Event Request register bit is mapped	r/-	r/w	0x00FF:0xFF0F

5.31 ECAT Event Request (0x0210:0x0211)

Table 45: Register ECAT Event Request (0x0210:0x0211)

Bit	Description	ECAT	PDI	Reset Value
0	DC Latch event: 0: No change on DC Latch Inputs 1: At least one change on DC Latch Inputs (Bit is cleared by reading DC Latch event times from ECAT for ECAT controlled Latch Units, so that Latch 0/1 Status 0x09AE:0x09AF indicates no event)	r/-	r/-	0
1	Reserved	r/-	r/-	0
2	DL Status event: 0: No change in DL Status 1: DL Status change (Bit is cleared by reading out DL Status 0x0110:0x0111 from ECAT)	r/-	r/-	0
3	AL Status event: 0: No change in AL Status 1: AL Status change (Bit is cleared by reading out AL Status 0x0130:0x0131 from ECAT)	r/-	r/-	0
4 5 ... 11	Mirrors values of each SyncManager Status: 0: No Sync Channel 0 event 1: Sync Channel 0 event pending 0: No Sync Channel 1 event 1: Sync Channel 1 event pending ... 0: No Sync Channel 7 event 1: Sync Channel 7 event pending	r/-	r/-	0
15:12	Reserved	r/-	r/-	0

5.32 AL Event Request (0x0220:0x0223)

Table 46: Register AL Event Request (0x0220:0x0223)

Bit	Description	ECAT	PDI	Reset Value
0	AL Control event: 0: No AL Control Register change 1: AL Control Register has been written[2] (Bit is cleared by reading AL Control register 0x0120:0x0121 from PDI)	r/-	r/-	0
1	DC Latch event: 0: No change on DC Latch Inputs 1: At least one change on DC Latch Inputs (Bit is cleared by reading DC Latch event times from PDI, so that Latch 0/1 Status 0x09AE:0x09AF indicates no event. Available if Latch Unit is PDI controlled)	r/-	r/-	0
2	State of DC SYNC0 (if register 0x0151.3=1): (Bit is cleared by reading SYNC0 status 0x098E from PDI, use only in Acknowledge mode)	r/-	r/-	0
3	State of DC SYNC1 (if register 0x0151.7=1): (Bit is cleared by reading of SYNC1 status 0x098F from PDI, use only in Acknowledge mode)	r/-	r/-	0
4	SyncManager activation register (SyncManager register offset 0x6) changed: 0: No change in any SyncManager 1: At least one SyncManager changed (Bit is cleared by reading SyncManager Activation registers 0x0806 etc. from PDI)	r/-	r/-	0
5	EEPROM Emulation: 0: No command pending 1: EEPROM command pending (Bit is cleared by acknowledging the command in EEPROM command register 0x0502 from PDI)	r/-	r/-	0
6	Watchdog Process Data: 0: Has not expired 1: Has expired (Bit is cleared by reading Watchdog Status Process Data 0x0440 from PDI)	r/-	r/-	0
7	Reserved	r/-	r/-	0
8 9 …. 23	SyncManager interrupts (SyncManager register offset 0x5, bit [0] or [1]): 0: No SyncManager 0 interrupt 1: SyncManager 0 interrupt pending 0: No SyncManager 1 interrupt 1: SyncManager 1 interrupt pending … 0: No SyncManager 15 interrupt 1: SyncManager 15 interrupt pending	r/-	r/-	0
31:24	Reserved	r/-	r/-	0

[2] AL control event is only generated if PDI emulation is turned off (PDI Control register 0x0140.8=0)

5.33 RX Error Counter (0x0300:0x0307)

Errors are only counted if the corresponding port is enabled.

Table 47: Register RX Error Counter Port y (0x0300+y*2:0x0301+y*2)

Bit	Description	ECAT	PDI	Reset Value
7:0	Invalid frame counter of Port y (counting is stopped when 0xFF is reached).	r/ w(clr)	r/-	0
15:8	RX Error counter of Port y (counting is stopped when 0xFF is reached). This is coupled directly to RX ERR of MII interface/EBUS interface.	r/ w(clr)	r/-	0

NOTE: Error Counters 0x0300-0x030B are cleared if one of the RX Error counters 0x0300-0x030B is written. Write value is ignored (write 0).

5.34 Forwarded RX Error Counter (0x0308:0x030B)

Table 48: Register Forwarded RX Error Counter Port y (0x0308+y)

Bit	Description	ECAT	PDI	Reset Value
7:0	Forwarded error counter of Port y (counting is stopped when 0xFF is reached).	r/ w(clr)	r/-	0

NOTE: Error Counters 0x0300-0x030B are cleared if one of the RX Error counters 0x0300-0x030B is written. Write value is ignored (write 0).

5.35 ECAT Processing Unit Error Counter (0x030C)

Table 49: Register ECAT Processing Unit Error Counter (0x030C)

Bit	Description	ECAT	PDI	Reset Value
7:0	ECAT Processing Unit error counter (counting is stopped when 0xFF is reached). Counts errors of frames passing the Processing Unit (e.g., FCS is wrong or datagram structure is wrong).	r/ w(clr)	r/-	0

NOTE: Error Counter 0x030C is cleared if error counter 0x030C is written. Write value is ignored (write 0).

5.36 PDI Error Counter (0x030D)

Table 50: Register PDI Error Counter (0x030D)

Bit	Description	ECAT	PDI	Reset Value
7:0	PDI Error counter (counting is stopped when 0xFF is reached). Counts if a PDI access has an interface error.	r/ w(clr)	r/-	0

NOTE: Error Counter 0x030D and Error Code 0x030E are cleared if error counter 0x030D is written. Write value is ignored (write 0).

5.37 SPI PDI Error Code (0x030E)

Table 51: Register SPI PDI Error Code (0x030E)

Bit	Description	ECAT	PDI	Reset Value
	SPI access which caused last PDI Error.	r/-	r/-	0
2:0	Number of SPI clock cycles of whole access (modulo 8)			
3	Busy violation during read access			
4	Read termination missing			
5	Access continued after read termination byte			
7:6	SPI command CMD[2:1]			

NOTE: Error Counter 0x030D and Error Code 0x030E are cleared if error counter 0x030D is written. Write value is ignored (write 0).

5.38 Lost Link Counter (0x0310:0x0313)

Table 52: Register Lost Link Counter Port y (0x0310+y)

Bit	Description	ECAT	PDI	Reset Value
7:0	Lost Link counter of Port y (counting is stopped when 0xff is reached). Counts only if port loop is Auto.	r/ w(clr)	r/-	0

NOTE: Only lost links at open ports are counted. Lost Link Counters 0x0310-0x0313 are cleared if one of the Lost Link Counters 0x0310-0x0313 is written. Write value is ignored (write 0).

5.39 Watchdog Divider (0x0400:0x0401)

Table 53: Register Watchdog Divider (0x0400:0x0401)

Bit	Description	ECAT	PDI	Reset Value
15:0	Watchdog divider: Number of 25 MHz tics (minus 2) that represents the basic watchdog increment. (Default value is 100µs = 2498)	r/w	r/-	0x09C2

5.40 Watchdog Time PDI (0x0410:0x0411)

Table 54: Register Watchdog Time PDI (0x0410:0x0411)

Bit	Description	ECAT	PDI	Reset Value
15:0	Watchdog Time PDI: number or basic watchdog increments (Default value with Watchdog divider 100µs means 100ms Watchdog)	r/w	r/-	0x03E8

Watchdog is disabled if Watchdog time is set to 0x0000. Watchdog is restarted with every PDI access.

5.41 Watchdog Time Process Data (0x0420:0x0421)

Table 55: Register Watchdog Time Process Data (0x0420:0x0421)

Bit	Description	ECAT	PDI	Reset Value
15:0	Watchdog Time Process Data: number of basic watchdog increments (Default value with Watchdog divider 100µs means 100ms Watchdog)	r/w	r/-	0x03E8

There is one Watchdog for all SyncManagers. Watchdog is disabled if Watchdog time is set to 0x0000. Watchdog is restarted with every write access to SyncManagers with Watchdog Trigger Enable Bit set.

5.42 Watchdog Status Process Data (0x0440:0x0441)

Table 56: Register Watchdog Status Process Data (0x0440:0x0441)

Bit	Description	ECAT	PDI	Reset Value
0	Watchdog Status of Process Data (triggered by SyncManagers) 0: Watchdog Process Data expired 1: Watchdog Process Data is active or disabled	r/-	r/ (w ack)*	0
15:1	Reserved	r/-	r/ (w ack)*	0

* PDI register function acknowledge by Write command is disabled: Reading this register from PDI clears AL Event Request 0x0220[6]. Writing to this register from PDI is not possible.

PDI register function acknowledge by Write command is enabled: Writing this register from PDI clears AL Event Request 0x0220[6]. Writing to this register from PDI is possible; write value is ignored (write 0).

5.43 Watchdog Counter Process Data (0x0442)

Table 57: Register Watchdog Counter Process Data (0x0442)

Bit	Description	ECAT	PDI	Reset Value
7:0	Watchdog Counter Process Data (counting is stopped when 0xFF is reached). Counts if Process Data Watchdog expires.	r/ w(clr)	r/-	0

NOTE: Watchdog Counters 0x0442-0x0443 are cleared if one of the Watchdog Counters 0x0442-0x0443 is written. Write value is ignored (write 0).

5.44 Watchdog Counter PDI (0x0443)

Table 58: Register Watchdog Counter PDI (0x0443)

Bit	Description	ECAT	PDI	Reset Value
7:0	Watchdog PDI counter (counting is stopped when 0xFF is reached). Counts if PDI Watchdog expires.	r/ w(clr)	r/-	0

NOTE: Watchdog Counters 0x0442-0x0443 are cleared if one of the Watchdog Counters 0x0442-0x0443 is written. Write value is ignored (write 0).

5.45 SII EEPROM Interface (0x0500:0x050F)

Table 59: SII EEPROM Interface Register overview

Register Address	Length (Byte)	Description
0x0500	1	EEPROM Configuration

Register Address	Length (Byte)	Description
0x0501	1	EEPROM PDI Access State
0x0502:0x0503	2	EEPROM Control/Status
0x0504:0x0507	4	EEPROM Address
0x0508:0x050F	4/8	EEPROM Data

EtherCAT controls the SSI EEPROM interface if EEPROM configuration register 0x0500.0=0 and EEPROM PDI Access register 0x0501.0=0, otherwise PDI controls the EEPROM interface.

In EEPROM emulation mode, the PDI executes outstanding EEPROM commands. The PDI has access to some registers while the EEPROM Interface is busy.

Table 60: Register EEPROM Configuration (0x0500)

Bit	Description	ECAT	PDI	Reset Value
0	EEPROM control is offered to PDI: 0: no 1: yes (PDI has EEPROM control)	r/w	r/-	0
1	Force ECAT access: 0: Do not change Bit 501.0 1: Reset Bit 501.0 to 0	r/w	r/-	0
7:2	Reserved, write 0	r/-	r/-	0

Table 61: Register EEPROM PDI Access State (0x0501)

Bit	Description	ECAT	PDI	Reset Value
0	Access to EEPROM: 0: PDI releases EEPROM access 1: PDI takes EEPROM access (PDI has EEPROM control)	r/-	r/(w)	0
7:1	Reserved, write 0	r/-	r/-	0

NOTE: r/(w): write access is only possible if 0x0500.0=1 and 0x0500.1=0.

Table 62: Register EEPROM Control/Status (0x0502:0x0503)

Bit	Description	ECAT	PDI	Reset Value
0	ECAT write enable*[2]: 0: Write requests are disabled 1: Write requests are enabled This bit is always 1 if PDI has EEPROM control.	r/(w)	r/-	0
4:1	Reserved, write 0	r/-	r/-	0
5	EEPROM emulation: 0: Normal operation (I^2C interface used) 1: PDI emulates EEPROM (I^2C not used)	r/-	r/-	TMC8460: 0
6	Supported number of EEPROM read bytes: 0: 4 Bytes 1: 8 Bytes	r/-	r/-	0
7	Selected EEPROM Algorithm: 0: 1 address byte (1KBit – 16KBit EEPROMs) 1: 2 address bytes (32KBit – 4 MBit EEPROMs)	r/-	r/-	TMC8460: pin PROM_SIZE
10:8	Command register*[2:] Write: Initiate command. Read: Currently executed command Commands: 000: No command/EEPROM idle (clear error bits) 001: Read 010: Write 100: Reload Others: Reserved/invalid commands (do not issue) EEPROM emulation only: after execution, PDI writes command value to indicate operation is ready.	r/(w)	r/(w) r/[w]	0
11	Checksum Error at in ESC Configuration Area: 0: Checksum ok 1: Checksum error	r/-	r/-	0
12	EEPROM loading status: 0: EEPROM loaded, device information ok 1: EEPROM not loaded, device information not available (EEPROM loading in progress or finished with a failure)	r/-	r/-	0
13	Error Acknowledge/Command*[3]: 0: No error 1: Missing EEPROM acknowledge or invalid command EEPROM emulation only: PDI writes 1 if a temporary failure has occurred.	r/-	r/- r/[w]	0
14	Error Write Enable*[3]: 0: No error 1: Write Command without Write enable	r/-	r/-	0
15	Busy: 0: EEPROM Interface is idle 1: EEPROM Interface is busy	r/-	r/-	0

NOTE: r/(w): write access depends upon the assignment of the EEPROM interface (ECAT/PDI). Write access is generally blocked if EEPROM interface is busy (0x0502.15=1).

NOTE: r/[w]: EEPROM emulation only: write access is possible if EEPROM interface is busy (0x0502.15=1). PDI acknowledges pending commands by writing a 1 into the corresponding command register bits (0x0502[10:8]).

Errors can be indicated by writing a 1 into the error bit 0x0502.13. Acknowledging clears AL Event Request 0x0220[5].

*) Write Enable bit 0 is self-clearing at the SOF of the next frame, Command bits [10:8] are self-clearing after the command is executed (EEPROM Busy ends). Writing "000" to the command register will also clear the error bits [14:13]. Command bits [10:8] are ignored if Error Acknowledge/Command is pending (bit 13).

*) Error bits are cleared by writing "000" (or any valid command) to Command Register Bits [10:8].

Table 63: Register EEPROM Address (0x0504:0x0507)

Bit	Description	ECAT	PDI	Reset Value
31:0	EEPROM Address 0: First word (= 16 bit) 1: Second word ... Actually used EEPROM Address bits: [9:0]: EEPROM size up to 16 kBit [17:0]: EEPROM size 32 kBit – 4 Mbit [32:0]: EEPROM Emulation	r/(w)	r/(w)	0

NOTE: r/(w): write access depends upon the assignment of the EEPROM interface (ECAT/PDI). Write access is generally blocked if EEPROM interface is busy (0x0502.15=1).

Table 64: Register EEPROM Data (0x0508:0x050F [0x0508:0x050B])

Bit	Description	ECAT	PDI	Reset Value
15:0	EEPROM Write data (data to be written to EEPROM) or EEPROM Read data (data read from EEPROM,. lower bytes)	r/(w)	r/(w) r/[w]	0
63:16	EEPROM Read data (data read from EEPROM, higher bytes)	r/-	r/- r/[w]	

NOTE: r/(w): write access depends upon the assignment of the EEPROM interface (ECAT/PDI). Write access is generally blocked if EEPROM interface is busy (0x0502.15=1).

NOTE: r/[w]: write access for EEPROM emulation if read or reload command is pending. See the following information for further details:

5.45.1 EEPROM emulation with TMC8460

Write access to EEPROM Data register 0x0508:0x050F is possible if EEPROM interface is busy (0x0502.15=1). PDI places EEPROM read data in this register before the pending EEPROM Read command is acknowledged (writing to 0x0502[10:8]). For Reload command: place the following information in the EEPROM Data register before acknowledging the command. This data is automatically transferred to the designated registers when the Reload command is acknowledged:

Table 65: Register EEPROM Data for EEPROM Emulation Reload (0x0508:0x050F)

Bit	Description	ECAT	PDI	Reset Value
15:0	Configured Station Alias (reloaded into 0x0012[15:0])	r/-	r/[w]	0
16	Enhanced Link Detection for all ports (reloaded into 0x0141[1])	r/-	r/[w]	0
20:17	Enhanced Link Detection for individual ports (reloaded into 0x0141[7:4])	r/-	r/[w]	0
24:21	ESC DL configuration (loaded into register 0x0100[23:20]) NOTE: This value is only taken over at the first EEPROM loading	r/-	r/[w]	0
27:25	FIFO Size reduction (loaded into register 0x0100[18:16]: 000: FIFO Size 7 001: FIFO Size 6 010: FIFO Size 5 011: FIFO Size 4 100: FIFO Size 3 101: FIFO Size 2 110: FIFO Size 1 111: FIFO Size 0 NOTE: This value is only taken over at the first EEPROM loading	r/-	r/[w]	0
31:28	Reserved, write 0	r/-	r/[w]	0

NOTE: r/[w]: write access for EEPROM emulation if read or reload command is pending.

5.46 MII Management Interface (0x0510:0x0515)

Table 66: MII Management Interface Register Overview

Register Address	Length (Byte)	Description
0x0510:0x0511	2	MII Management Control/Status
0x0512	1	PHY Address
0x0513	1	PHY Register Address
0x0514:0x0515	2	PHY Data
0x0516	1	MII Management ECAT Access State
0x0517	1	MII Management PDI Access State
0x0518:0x051B	4	PHY Port Status

PDI controls the MII management interface if MII Management PDI Access register 0x0517.0=1, otherwise EtherCAT controls the MII management interface.

Table 67: Register MII Management Control/Status (0x0510:0x0511)

Bit	Description	ECAT	PDI	Reset Value
0	Write enable*: 0: Write disabled 1: Write enabled This bit is always 1 if PDI has MI control.	r/(w)	r/-	0
1	Management Interface can be controlled by PDI (registers 0x0516-0 x0517): 0: Only ECAT control 1: PDI control possible	r/-	r/-	TMC8460: 1
2	MI link detection (link configuration, link detection, registers 0x0518-0x051B): 0: Not available 1: MI link detection active	r/-	r/-	TMC8460: 0
7:3	PHY address of port 0	r/-	r/-	TMC8460: Depends on configuration
9:8	Command register*: Write: Initiate command. Read: Currently executed command Commands: 00: No command/MI idle (clear error bits) 01: Read 10: Write Others: Reserved/invalid commands (do not issue)	r/(w)	r/(w)	0
12:10	Reserved, write 0	r/-	r/-	0
13	Read error: 0: No read error 1: Read error occurred (PHY or register not available) Cleared by writing to this register.	r/(w)	r/(w)	0
14	Command error: 0: Last Command was successful 1: Invalid command or write command without Write Enable Cleared with a valid command or by writing "00" to Command register bits [9:8].	r/-	r/-	0
15	Busy: 0: MI control state machine is idle 1: MI control state machine is active	r/-	r/-	0

NOTE: r/ (w): write access depends on assignment of MI (ECAT/PDI). Write access is generally blocked if Management interface is busy (0x0510.15=1).

* Write enable bit 0 is self-clearing at the SOF of the next frame (or at the end of the PDI access), Command bits [9:8] are self-clearing after the command is executed (Busy ends). Writing "00" to the command register will also clear the error bits [14:13]. The Command bits are cleared after the command is executed.

Table 68: Register PHY Address (0x0512)

Bit	Description	ECAT	PDI	Reset Value
4:0	PHY Address	r/(w)	r/(w)	0
6:5	Reserved, write 0	r/-	r/-	
7	Show configured PHY address of port 0-3 in register 0x0510[7:3]. Select port x with bits [4:0] of this register (valid values are 0-3): 0: Show address of port 0 (offset) 1: Show individual address of port x	r/(w)	r/(w)	0

NOTE: r/ (w): write access depends on assignment of MI (ECAT/PDI). Write access is generally blocked if Management interface is busy (0x0510.15=1).

Table 69: Register PHY Register Address (0x0513)

Bit	Description	ECAT	PDI	Reset Value
4:0	Address of PHY Register that shall be read/written	r/(w)	r/(w)	0
7:5	Reserved, write 0	r/-	r/-	0

NOTE: r/ (w): write access depends on assignment of MI (ECAT/PDI). Write access is generally blocked if Management interface is busy (0x0510.15=1).

Table 70: Register PHY Data (0x0514:0x0515)

Bit	Description	ECAT	PDI	Reset Value
15:0	PHY Read/Write Data	r/(w)	r/(w)	0

NOTE: r/ (w): write access depends on assignment of MI (ECAT/PDI). Access is generally blocked if Management interface is busy (0x0510.15=1).

Table 71: Register MII Management ECAT Access State (0x0516)

Bit	Description	ECAT	PDI	Reset Value
0	Access to MII management: 0: ECAT enables PDI takeover of MII management control 1: ECAT claims exclusive access to MII management	r/(w)	r/-	0
7:1	Reserved, write 0	r/-	r/-	0

NOTE: r/ (w): write access is only possible if 0x0517.0=0.

Table 72: Register MII Management PDI Access State (0x0517)

Bit	Description	ECAT	PDI	Reset Value
0	Access to MII management: 0: ECAT has access to MII management 1: PDI has access to MII management	r/-	r/(w)	0
1	Force PDI Access State: 0: Do not change Bit 517.0 1: Reset Bit 517.0 to 0	r/w	r/-	0
7:2	Reserved, write 0	r/-	r/-	0

NOTE: r/ (w): assigning access to PDI (bit 0 = 1) is only possible if 0x0516.0=0 and 0x0517.1=0, and if the SII EEPROM is loaded (0x0110[0]=1).

Table 73: Register PHY Port y (port number y=0 to 3) Status (0x0518+y)

Bit	Description	ECAT	PDI	Reset Value
0	Physical link status (PHY status register 1.2): 0: No physical link 1: Physical link detected	r/-	r/-	0
1	Link status (100 Mbit/s, Full Duplex, Autonegotiation): 0: No link 1: Link detected	r/-	r/-	0
2	Link status error: 0: No error 1: Link error, link inhibited	r/-	r/-	0
3	Read error: 0: No read error occurred 1: A read error has occurred Cleared by writing any value to at least one of the PHY Status Port y registers.	r/(w/clr)	r/(w/clr)	0
4	Link partner error: 0: No error detected 1: Link partner error	r/-	r/-	0
5	PHY configuration updated: 0: No update 1: PHY configuration was updated Cleared by writing any value to at least one of the PHY Status Port y registers.	r/(w/clr)	r/(w/clr)	0
7:6	Reserved	r/-	r/-	0

NOTE: r/ (w): write access depends on assignment of MI (ECAT/PDI).

5.47 Parameter RAM (0x0580:0x05AB) for TMC8460 MFCIO Block Configuration

The content of these registers can be automatically loaded from EEPROM after reset/power-up. Therefore, the configuration date in the EEPROM must contain a section with category 1 and the appropriate configuration vector.

There are 44 MFCIO registers. Each byte configures one of the MFCIO registers.

y is in the range of 0 to 43 and is used as offset with respect to address 0x0580.

Table 74:MFCIO Register Configuration (0x0580+y)

Bit	Description	ECAT	PDI	Reset Value
3:0	Update trigger selection for the respective MFCIO register	r/w	r/w	0
4	0: ECAT mapping disabled, general read/write access only via MFC CTRL SPI to/from MCU 1: ECAT mapping enabled, general write access only from ECAT master using configured SyncManager, always readable by MCU via MFC CTRL SPI	r/w	r/w	0
7:5	reserved			0

When (re)configuring the EEPROM from an EtherCAT master system special care must be taken. Not every master allows writing a category 1 entry to the EEPROM. There are different ways to write this into the EEPROM for automatically loading the MFCIO block configuration at power-up:

- Use preprogrammed EEPROMs.
- Use a different category, e.g., 2049, first. Then overwrite the upper byte with 0 with a single EEPROM byte write.
- Use the local microcontroller (if available) and have it connected to the EEPROM via I2C. Hold the TMC8460 in reset while programming the EEPROM first. Afterwards tristate the local I2C bus so that the TMC8460 has control over the IC2 interface and release the TMC8460 from reset.

5.48 FMMU (0x0600:0x06FF)

Each FMMU entry is described in 16 Bytes from 0x0600:0x060F to 0x06F0:0x06FF.

y is the FMMU index.

The TMC8460-BI supports up to 6 FMMUs. Thus TMC8460-BI supports y=0…5.

Table 75: FMMU Register overview

Register Address Offset	Length (Byte)	Description
+0x0:0x3	4	Logical Start Address
+0x4:0x5	2	Length
+0x6	1	Logical Start bit
+0x7	1	Logical Stop bit
+0x8:0x9	2	Physical Start Address
+0xA	1	Physical Start bit
+0xB	1	Type
+0xC	1	Activate
+0xD:0xF	3	Reserved

Table 76: Register Logical Start address FMMU y (0x06y0:0x06y3)

Bit	Description	ECAT	PDI	Reset Value
31:0	Logical start address within the EtherCAT Address Space.	r/w	r/-	0

Table 77: Register Length FMMU y (0x06y4:0x06y5)

Bit	Description	ECAT	PDI	Reset Value
15:0	Offset from the first logical FMMU Byte to the last FMMU Byte + 1 (e.g., if two bytes are used then this parameter shall contain 2)	r/w	r/-	0

Table 78: Register Start bit FMMU y in logical address space (0x06y6)

Bit	Description	ECAT	PDI	Reset Value
2:0	Logical starting bit that shall be mapped (bits are counted from least significant bit (=0) to most significant bit(=7)	r/w	r/-	0
7:3	Reserved, write 0	r/-	r/-	0

Table 79: Register Stop bit FMMU y in logical address space (0x06y7)

Bit	Description	ECAT	PDI	Reset Value
2:0	Last logical bit that shall be mapped (bits are counted from least significant bit (=0) to most significant bit(=7)	r/w	r/-	0
7:3	Reserved, write 0	r/-	r/-	0

Table 80: Register Physical Start address FMMU y (0x06y8-0x06y9)

Bit	Description	ECAT	PDI	Reset Value
15:0	Physical Start Address (mapped to logical Start address)	r/w	r/-	0

Table 81: Register Physical Start bit FMMU y (0x06yA)

Bit	Description	ECAT	PDI	Reset Value
2:0	Physical starting bit as target of logical start bit mapping (bits are counted from least significant bit (=0) to most significant bit(=7)	r/w	r/-	0
7:3	Reserved, write 0	r/-	r/-	0

Table 82: Register Type FMMU y (0x06yB)

Bit	Description	ECAT	PDI	Reset Value
0	0: Ignore mapping for read accesses 1: Use mapping for read accesses	r/w	r/-	0
1	0: Ignore mapping for write accesses 1: Use mapping for write accesses	r/w	r/-	0
7:2	Reserved, write 0	r/-	r/-	0

Table 83: Register Activate FMMU y (0x06yC)

Bit	Description	ECAT	PDI	Reset Value
0	0: FMMU deactivated 1: FMMU activated. FMMU checks logical addressed blocks to be mapped according to mapping configured	r/w	r/-	0
7:1	Reserved, write 0	r/-	r/-	0

Table 84: Register Reserved FMMU y (0x06yD:0x06yF)

Bit	Description	ECAT	PDI	Reset Value
23:0	Reserved, write 0	r/-	r/-	0

5.49 SyncManager (0x0800:0x087F)

SyncManager registers are mapped from 0x0800:0x0807 to 0x0818:0x087F.

y specifies the SyncManager number.

The TMC8460-BI supports up to 6 SyncManagers. Thus TMC8460-BI supports y=0…5.

Table 85: SyncManager Register overview

Register Address Offset	Length (Byte)	Description
+0x0:0x1	2	Physical Start Address
+0x2:0x3	2	Length
+0x4	1	Control Register
+0x5	1	Status Register
+0x6	1	Activate
+0x7	1	PDI Control

Table 86: Register physical Start Address SyncManager y (0x0800+y*8:0x0801+y*8)

Bit	Description	ECAT	PDI	Reset Value
15:0	Specifies first byte that will be handled by SyncManager	r/(w)	r/-	0

NOTE r/(w): Register can only be written if SyncManager is disabled (+0x6.0 = 0).

Table 87: Register Length SyncManager y (0x0802+y*8:0x0803+y*8)

Bit	Description	ECAT	PDI	Reset Value
15:0	Number of bytes assigned to SyncManager (shall be greater 1, otherwise SyncManager is not activated. If set to 1, only Watchdog Trigger is generated if configured)	r/(w)	r/-	0

NOTE r/(w): Register can only be written if SyncManager is disabled (+0x6.0 = 0).

Table 88: Register Control Register SyncManager y (0x0804+y*8)

Bit	Description	ECAT	PDI	Reset Value
1:0	Operation Mode: 00: Buffered (3 buffer mode) 01: Reserved 10: Mailbox (Single buffer mode) 11: Reserved	r/(w)	r/-	00
3:2	Direction: 00: Read: ECAT read access, PDI write access. 01: Write: ECAT write access, PDI read access. 10: Reserved 11: Reserved	r/(w)	r/-	00
4	Interrupt in ECAT Event Request Register: 0: Disabled 1: Enabled	r/(w)	r/-	0
5	Interrupt in PDI Event Request Register: 0: Disabled 1: Enabled	r/(w)	r/-	0
6	Watchdog Trigger Enable: 0: Disabled 1: Enabled	r/(w)	r/-	0
7	Reserved, write 0	r/-	r/-	0

NOTE r/(w): Register can only be written if SyncManager is disabled (+0x6.0 = 0).

Table 89: Register Status Register SyncManager y (0x0805+y*8)

Bit	Description	ECAT	PDI	Reset Value
0	Interrupt Write: 1: Interrupt after buffer was completely and successfully written 0: Interrupt cleared after first byte of buffer was read NOTE: This interrupt is signaled to the reading side if enabled in the SM Control register.	r/-	r/-	0
1	Interrupt Read: 1: Interrupt after buffer was completely and successful read 0: Interrupt cleared after first byte of buffer was written NOTE: This interrupt is signaled to the writing side if enabled in the SM Control register.	r/-	r/-	0
2	Reserved	r/-	r/-	0
3	Mailbox mode: mailbox status: 0: Mailbox empty 1: Mailbox full Buffered mode: reserved	r/-	r/-	0
5:4	Buffered mode: buffer status (last written buffer): 00: 1. buffer 01: 2. buffer 10: 3. buffer 11: (no buffer written) Mailbox mode: reserved	r/-	r/-	11
6	Read buffer in use (opened)	r/-	r/-	0
7	Write buffer in use (opened)	r/-	r/-	0

Table 90: Register Activate SyncManager y (0x0806+y*8)

Bit	Description	ECAT	PDI	Reset Value
0	SyncManager Enable/Disable: 0: Disable: Access to Memory without SyncManager control 1: Enable: SyncManager is active and controls Memory area set in configuration	r/w	r/ (w ack)*	0
1	Repeat Request: A toggle of Repeat Request means that a mailbox retry is needed (primarily used in conjunction with ECAT Read Mailbox)	r/w	r/ (w ack)*	0
5:2	Reserved, write 0	r/-	r/ (w ack)*	0
6	Latch Event ECAT: 0: No 1: Generate Latch event if EtherCAT master issues a buffer exchange	r/w	r/ (w ack)*	0
7	Latch Event PDI: 0: No 1: Generate Latch events if PDI issues a buffer exchange or if PDI accesses buffer start address	r/w	r/ (w ack)*	0

* PDI register function acknowledge by Write command is disabled: Reading this register from PDI in all SyncManagers which have changed activation clears AL Event Request 0x0220[4]. Writing to this register from PDI is not possible.

PDI register function acknowledge by Write command is enabled: Writing this register from PDI in all SyncManagers which have changed activation clears AL Event Request 0x0220[4]. Writing to this register from PDI is possible; write value is ignored (write 0).

Table 91: Register PDI Control SyncManager y (0x0807+y*8)

Bit	Description	ECAT	PDI	Reset Value
0	Deactivate SyncManager: Read: 0: Normal operation, SyncManager activated. 1: SyncManager deactivated and reset SyncManager locks access to Memory area. Write: 0: Activate SyncManager 1: Request SyncManager deactivation NOTE: Writing 1 is delayed until the end of a frame which is currently processed.	r/-	r/w	0
1	Repeat Ack: If this is set to the same value as set by Repeat Request, the PDI acknowledges the execution of a previous set Repeat request.	r/-	r/w	0
7:2	Reserved, write 0	r/-	r/-	0

5.50 Distributed Clocks (0x0900:0x09FF)

5.50.1 Receive Times

Table 92: Register Receive Time Port 0 (0x0900:0x0903)

Bit	Description	ECAT	PDI	Reset Value
31:0	Write: A write access to register 0x0900 with BWR or FPWR latches the local time of the beginning of the receive frame (start first bit of preamble) at each port. Read: Local time of the beginning of the last receive frame containing a write access to this register.	r/w (special function)	r/-	Undefined

NOTE: The time stamps cannot be read in the same frame in which this register was written.

Table 93: Register Receive Time Port 1 (0x0904:0x0907)

Bit	Description	ECAT	PDI	Reset Value
31:0	Local time of the beginning of a frame (start first bit of preamble) received at port 1 containing a BWR or FPWR to Register 0x0900.	r/-	r/-	Undefined

NOTE: Register 0x0910:0x0913[0x910:0x0917] is described in the next chapter.

Table 94: Register Receive Time ECAT Processing Unit (0x0918:0x091F)

Bit	Description	ECAT	PDI	Reset Value
63:0	Local time of the beginning of a frame (start first bit of preamble) received at the ECAT Processing Unit containing a write access to Register 0x0900 NOTE: E.g., if port 0 is open, this register reflects the Receive Time Port 0 as a 64 Bit value.	r/-	r/-	Undefined

5.50.2 Time Loop Control Unit

Time Loop Control unit is usually assigned to ECAT. Write access to Time Loop Control registers by PDI (and not ECAT) is only possible with explicit configuration.

Table 95: Register System Time (0x0910:0x0913 [0x0910:0x0917])

Bit	Description	ECAT	PDI	Reset Value
63:0	ECAT read access: Local copy of the System Time when the frame passed the reference clock (i.e., including System Time Delay). Time latched at beginning of the frame (Ethernet SOF delimiter)	r	-	0
63:0	PDI read access: Local copy of the System Time. Time latched when reading first byte (0x0910)	-	r	
31:0	Write access: Written value will be compared with the local copy of the System time. The result is an input to the time control loop. NOTE: written value will be compared at the end of the frame with the latched (SOF) local copy of the System time if at least the first byte (0x0910) was written.	(w) (special function)	-	
31:0	Write access: Written value will be compared with Latch0 Time Positive Edge time. The result is an input to the time control loop. NOTE: written value will be compared at the end of the access with Latch0 Time Positive Edge (0x09B0:0x09B3) if at least the last byte (0x0913) was written.	-	(w) (special function)	

NOTE: Write access to this register depends upon ESC configuration (typically ECAT). The TMC8460-BI is **not** configured for this option.

Table 96: Register System Time Offset (0x0920:0x0923 [0x0920:0x0927])

Bit	Description	ECAT	PDI	Reset Value
63:0	Difference between local time and System Time. Offset is added to the local time.	r/(w)	r/(w)	0

NOTE: Write access to this register depends upon ESC configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI controlled). Reset internal system time difference filter and speed counter filter by writing Speed Counter Start (0x0930:0x0931) after changing this value.

Table 97: Register System Time Delay (0x0928:0x092B)

Bit	Description	ECAT	PDI	Reset Value
31:0	Delay between Reference Clock and the ESC	r/(w)	r/(w)	0

NOTE: Write access to this register depends upon ESC configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI controlled). Reset internal system time difference filter and speed counter filter by writing Speed Counter Start (0x0930:0x0931) after changing this value.

Table 98: Register System Time Difference (0x092C:0x092F)

Bit	Description	ECAT	PDI	Reset Value
30:0	Mean difference between local copy of System Time and received System Time values	r/-	r/-	0
31	0: Local copy of System Time greater than or equal received System Time 1: Local copy of System Time smaller than received System Time	r/-	r/-	0

Table 99: Register Speed Counter Start (0x0930:0x931)

Bit	Description	ECAT	PDI	Reset Value
14:0	Bandwidth for adjustment of local copy of System Time (larger values → smaller bandwidth and smoother adjustment) A write access resets System Time Difference (0x092C:0x092F) and Speed Counter Diff (0x0932:0x0933). Minimum value: 0x0080 to 0x3FFF	r/(w)	r/(w)	0x1000
15	Reserved, write 0	r/-	r/-	0

NOTE: Write access to this register depends upon ESC configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI controlled).

Table 100: Register Speed Counter Diff (0x0932:0x933)

Bit	Description	ECAT	PDI	Reset Value
15:0	Representation of the deviation between local clock period and Reference Clock's clock period (representation: two's complement) Range: ±(Speed Counter Start – 0x7F)	r/-	r/-	0x0000

NOTE: Calculate the clock deviation after System Time Difference has settled at a low value as follows:

$$\text{Deviation} = \frac{\text{SpeedCounterDiff}}{5(\text{SpeedCounterStart} + \text{SpeedCounterDiff} + 2)(\text{SpeedCounterStart} - \text{SpeedCounterDiff} + 2)}$$

Table 101: Register System Time Difference Filter Depth (0x0934)

Bit	Description	ECAT	PDI	Reset Value
3:0	Filter depth for averaging the received System Time deviation TMC8460: A write access resets System Time Difference (0x092C:0x092F)	r/(w)	r/(w)	4
7:4	Reserved, write 0	r/-	r/-	0

NOTE: Write access to this register depends upon ESC configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI controlled).

Table 102: Register Speed Counter Filter Depth (0x0935)

Bit	Description	ECAT	PDI	Reset Value
3:0	Filter depth for averaging the clock period deviation TMC8460: A write access resets the internal speed counter filter.	r/(w)	r/(w)	12
7:4	Reserved, write 0	r/-	r/-	0

NOTE: Write access to this register depends upon ESC configuration (typically ECAT, PDI only with explicit ESC configuration: System Time PDI controlled) .

Table 103: Register Receive Time Latch Mode (0x0936)

Bit	Description	ECAT	PDI	Reset Value
0	Receive Time Latch Mode: 0: Forwarding mode (used if frames are entering the ESC at port 0 first): Receive time stamps of ports 1-3 are enabled after the write access to 0x0900, so the following frame at ports 1-3 will be time stamped (this is typically the write frame to 0x0900 coming back from the network behind the ESC). 1: Reverse mode (used if frames are entering ESC at port 1-3 first): Receive time stamps of ports 1-3 are immediately taken over from the internal hidden time stamp registers, so the previous frame entering the ESC at ports 1-3 will be time stamped when the write frame to 0x0900 enters port 0 (the previous frame at ports 1-3 is typically the write frame to 0x0900 coming from the master, which will enable time stamping at the ESC once it enters port 0).	r/w	r/-	0
7:1	Reserved	r/-	r/-	0

NOTE: There should not be frames traveling around the network before and after the time stamps are taken, otherwise these frames might get time stamped, and not the write frame to 0x0900.

5.50.3 Cyclic Unit Control

Table 104: Register Cyclic Unit Control (0x0980)

Bit	Description	ECAT	PDI	Reset Value
0	SYNC out unit control: 0: ECAT controlled 1: PDI controlled	r/w	r/-	0
3:1	Reserved, write 0	r/-	r/-	0
4	Latch In unit 0: 0: ECAT controlled 1: PDI controlled NOTE: Always 1 (PDI controlled) if System Time is PDI controlled. Latch interrupt is routed to ECAT/PDI depending on this setting	r/w	r/-	0
5	Latch In unit 1: 0: ECAT controlled 1: PDI controlled NOTE: Latch interrupt is routed to ECAT/PDI depending on this setting	r/w	r/-	0
7:6	Reserved, write 0	r/-	r/-	0

5.50.4 SYNC Out Unit

Table 105: Register Activation register (0x0981)

Bit	Description	ECAT	PDI	Reset Value
0	Sync Out Unit activation: 0: Deactivated 1: Activated	r/(w)	r/(w)	0
1	SYNC0 generation: 0: Deactivated 1: SYNC0 pulse is generated	r/(w)	r/(w)	0
2	SYNC1 generation: 0: Deactivated 1: SYNC1 pulse is generated	r/(w)	r/(w)	0
3	Auto-activation by writing Start Time Cyclic Operation (0x0990:0x0997): 0: Disabled 1: Auto-activation enabled. 0x0981.0 is set automatically after Start Time is written.	r/(w)	r/(w)	0
4	Extension of Start Time Cyclic Operation (0x0990:0x0993): 0: No extension 1: Extend 32 bit written Start Time to 64 bit	r/(w)	r/(w)	0
5	Start Time plausibility check: 0: Disabled. SyncSignal generation if Start Time is reached. 1: Immediate SyncSignal generation if Start Time is outside near future (see 0x0981.6)	r/(w)	r/(w)	0
6	Near future configuration (approx.): 0: ½ DC width future (2^{31} ns or 2^{63} ns) 1: ~2.1 sec. future (2^{31} ns)	r/(w)	r/(w)	0
7	SyncSignal debug pulse (Vasily bit): 0: Deactivated 1: Immediately generate one ping only on SYNC0-1 according to 0x0981[2:1] for debugging This bit is self-clearing, always read 0.	r/(w)	r/(w)	0

NOTE: Write to this register depends upon setting of 0x0980.0.

Table 106: Register Pulse Length of SyncSignals (0x0982:0x983)

Bit	Description	ECAT	PDI	Reset Value
15:0	Pulse length of SyncSignals (in Units of 10ns) 0: Acknowledge mode: SyncSignal will be cleared by reading SYNC[1:0] Status register	r/-	r/-	TMC8460: Depends on configuration, later EEPROM ADR 0x0002

Table 107: Register Activation Status (0x0984)

Bit	Description	ECAT	PDI	Reset Value
0	SYNC0 activation state: 0: First SYNC0 pulse is not pending 1: First SYNC0 pulse is pending	r/-	r/-	0
1	SYNC1 activation state: 0: First SYNC1 pulse is not pending 1: First SYNC1 pulse is pending	r/-	r/-	0
2	Start Time Cyclic Operation (0x0990:0x0997) plausibility check result when Sync Out Unit was activated: 0: Start Time was within near future 1: Start Time was out of near future (0x0981.6)	r/-	r/-	0
7:3	Reserved	r/-	r/-	0

Table 108: Register SYNC0 Status (0x098E)

Bit	Description	ECAT	PDI	Reset Value
0	SYNC0 state for Acknowledge mode. SYNC0 in Acknowledge mode is cleared by reading this register from PDI, use only in Acknowledge mode	r/-	r/ (w ack)*	0
7:1	Reserved	r/-	r/ (w ack)*	0

* PDI register function acknowledge by Write command is disabled: Reading this register from PDI clears AL Event Request 0x0220[2]. Writing to this register from PDI is not possible.

PDI register function acknowledge by Write command is enabled: Writing this register from PDI clears AL Event Request 0x0220[2]. Writing to this register from PDI is possible; write value is ignored (write 0).

Table 109: Register SYNC1 Status (0x098F)

Bit	Description	ECAT	PDI	Reset Value
0	SYNC1 state for Acknowledge mode. SYNC1 in Acknowledge mode is cleared by reading this register from PDI, use only in Acknowledge mode	r/-	r/ (w ack)*	0
7:1	Reserved	r/-	r/ (w ack)*	0

* PDI register function acknowledge by Write command is disabled: Reading this register from PDI clears AL Event Request 0x0220[3]. Writing to this register from PDI is not possible.

PDI register function acknowledge by Write command is enabled: Writing this register from PDI clears AL Event Request 0x0220[3]. Writing to this register from PDI is possible; write value is ignored (write 0).

Table 110: Register Start Time Cyclic Operation (0x0990:0x0993 [0x0990:0x0997])

Bit	Description	ECAT	PDI	Reset Value
63:0	Write: Start time (System time) of cyclic operation in ns Read: System time of next SYNC0 pulse in ns	r/(w)	r/(w)	0

NOTE: Write to this register depends upon setting of 0x0980.0. Only writable if 0x0981.0=0.
Auto-activation (0x0981.3=1): upper 32 bits are automatically extended if only lower 32 bits are written within one frame.

Table 111: Register Next SYNC1 Pulse (0x0998:0x099B [0x0998:0x099F])

Bit	Description	ECAT	PDI	Reset Value
63:0	System time of next SYNC1 pulse in ns	r/-	r/-	0

Table 112: Register SYNC0 Cycle Time (0x09A0:0x09A3)

Bit	Description	ECAT	PDI	Reset Value
31:0	Time between two consecutive SYNC0 pulses in ns. 0: Single shot mode, generate only one SYNC0 pulse.	r/(w)	r/(w)	0

NOTE: Write to this register depends upon setting of 0x0980.0.

Table 113: Register SYNC1 Cycle Time (0x09A4:0x09A7)

Bit	Description	ECAT	PDI	Reset Value
31:0	Time between SYNC1 pulses and SYNC0 pulse in ns	r/(w)	r/(w)	0

NOTE: Write to this register depends upon setting of 0x0980.0.

5.50.5 Latch In unit

Table 114: Register Latch0 Control (0x09A8)

Bit	Description	ECAT	PDI	Reset Value
0	Latch0 positive edge: 0: Continuous Latch active 1: Single event (only first event active)	r/(w)	r/(w)	0
1	Latch0 negative edge: 0: Continuous Latch active 1: Single event (only first event active)	r/(w)	r/(w)	0
7:2	Reserved, write 0	r/-	r/-	0

NOTE: Write access depends upon setting of 0x0980.4.

Table 115: Register Latch1 Control (0x09A9)

Bit	Description	ECAT	PDI	Reset Value
0	Latch1 positive edge: 0: Continuous Latch active 1: Single event (only first event active)	r/(w)	r/(w)	0
1	Latch1 negative edge: 0: Continuous Latch active 1: Single event (only first event active)	r/(w)	r/(w)	0
7:2	Reserved, write 0	r/-	r/-	0

NOTE: Write access depends upon setting of 0x0980.5.

Table 116: Register Latch0 Status (0x09AE)

Bit	Description	ECAT	PDI	Reset Value
0	Event Latch0 positive edge. 0: Positive edge not detected or continuous mode 1: Positive edge detected in single event mode only. Flag cleared by reading out Latch0 Time Positive Edge.	r/-	r/-	0
1	Event Latch0 negative edge. 0: Negative edge not detected or continuous mode 1: Negative edge detected in single event mode only. Flag cleared by reading out Latch0 Time Negative Edge.	r/-	r/-	0
2	Latch0 pin state	r/-	r/-	0
7:3	Reserved	r/-	r/-	0

Table 117: Register Latch1 Status (0x09AF)

Bit	Description	ECAT	PDI	Reset Value
0	Event Latch1 positive edge. 0: Positive edge not detected or continuous mode 1: Positive edge detected in single event mode only. Flag cleared by reading out Latch1 Time Positive Edge.	r/-	r/-	0
1	Event Latch1 negative edge. 0: Negative edge not detected or continuous mode 1: Negative edge detected in single event mode only. Flag cleared by reading out Latch1 Time Negative Edge.	r/-	r/-	0
2	Latch1 pin state	r/-	r/-	0
7:3	Reserved	r/-	r/-	0

Table 118: Register Latch0 Time Positive Edge (0x09B0:0x09B3 [0x09B0:0x09B7])

Bit	Description	ECAT	PDI	Reset Value
63:0	Register captures System time at the positive edge of the Latch0 signal.	r(ack)/-	r/ (w ack)*	0

NOTE: Register bits [63:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value. Reading this register from ECAT clears Latch0 Status 0x09AE[0] if 0x0980.4=0. Writing to this register from ECAT is not possible.

* PDI register function acknowledge by Write command is disabled: Reading this register from PDI if 0x0980.4=1 clears Latch0 Status 0x09AE[0]. Writing to this register from PDI is not possible.
PDI register function acknowledge by Write command is enabled: Writing this register from PDI if 0x0980.4=1 clears Latch0 Status 0x09AE[0]. Writing to this register from PDI is possible; write value is ignored (write 0).

Table 119: Register Latch0 Time Negative Edge (0x09B8:0x09BB [0x09B8:0x09BF])

Bit	Description	ECAT	PDI	Reset Value
63:0	Register captures System time at the negative edge of the Latch0 signal.	r(ack)/-	r/ (w ack)*	0

NOTE: Register bits [63:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value. Reading this register from ECAT clears Latch0 Status 0x09AE[1] if 0x0980.4=0. Writing to this register from ECAT is not possible.

* PDI register function acknowledge by Write command is disabled: Reading this register from PDI if 0x0980.4=1 clears Latch0 Status 0x09AE[1]. Writing to this register from PDI is not possible.
PDI register function acknowledge by Write command is enabled: Writing this register from PDI if 0x0980.4=1 clears Latch0 Status 0x09AE[1]. Writing to this register from PDI is possible; write value is ignored (write 0).

Table 120: Register Latch1 Time Positive Edge (0x09C0:0x09C3 [0x09C0:0x09C7])

Bit	Description	ECAT	PDI	Reset Value
63:0	Register captures System time at the positive edge of the Latch1 signal.	r(ack)/-	r/ (w ack)*	0

NOTE: Register bits [63:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value. Reading this register from ECAT clears Latch0 Status 0x09AF[0] if 0x0980.5=0. Writing to this register from ECAT is not possible.

* PDI register function acknowledge by Write command is disabled: Reading this register from PDI if 0x0980.5=1 clears Latch0 Status 0x09AF[0]. Writing to this register from PDI is not possible.
PDI register function acknowledge by Write command is enabled: Writing this register from PDI if 0x0980.5=1 clears Latch0 Status 0x09AF[0]. Writing to this register from PDI is possible; write value is ignored (write 0).

Table 121: Register Latch1 Time Negative Edge (0x09C8:0x09CB [0x09C8:0x09CF])

Bit	Description	ECAT	PDI	Reset Value
63:0	Register captures System time at the negative edge of the Latch1 signal.	r(ack)/-	r/ (w ack)*	0

NOTE: Register bits [63:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value. Reading this register from ECAT clears Latch0 Status 0x09AF[1] if 0x0980.5=0. Writing to this register from ECAT is not possible.

* PDI register function acknowledge by Write command is disabled: Reading this register from PDI if 0x0980.5=1 clears Latch0 Status 0x09AF[1]. Writing to this register from PDI is not possible.
PDI register function acknowledge by Write command is enabled: Writing this register from PDI if 0x0980.5=1 clears Latch0 Status 0x09AF[1]. Writing to this register from PDI is possible; write value is ignored (write 0).

5.50.6 SyncManager Event Times

Table 122: Register EtherCAT Buffer Change Event Time (0x09F0:0x09F3)

Bit	Description	ECAT	PDI	Reset Value
31:0	Register captures local time of the beginning of the frame which causes at least one SyncManager to assert an ECAT event	r/-	r/-	0

NOTE: Register bits [31:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value.

Table 123: Register PDI Buffer Start Event Time (0x09F8:0x09FB)

Bit	Description	ECAT	PDI	Reset Value
31:0	Register captures local time when at least one SyncManager asserts an PDI buffer start event	r/-	r/-	0

NOTE: Register bits [31:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value.

Table 124: Register PDI Buffer Change Event Time (0x09FC:0x09FF)

Bit	Description	ECAT	PDI	Reset Value
31:0	Register captures local time when at least one SyncManager asserts an PDI buffer change event	r/-	r/-	0

NOTE: Register bits [31:8] are internally latched (ECAT/PDI independently) when bits [7:0] are read, which guarantees reading a consistent value.

5.51 ESC Specific Product and Vendor ID

Table 125: Register Product ID (0x0E00:0x0E07)

Bit	Description	ECAT	PDI	Reset Value
63:0	Product ID	r/-	r/-	TMC8460: 0x0000 0000 0100 8460

Table 126: Register Vendor ID (0x0E08:0x0E0F)

Bit	Description	ECAT	PDI	Reset Value
31:0	Vendor ID: NOTE: Test Vendor IDs have [31:28]=0xE	r/-	r/-	TMC8460: 0x0000 0286
63:32	Reserved	r/-	r/-	

5.52 User RAM (0x0F80:0x0FFF)

Table 127: User RAM (0x0F80:0x0FFF)

Bit	Description	ECAT	PDI	Reset Value
----	Application specific information	r/w	r/w	TMC8460: Random/undefined

5.53 Process Data RAM (0x1000:0xFFFF)

The Process Data RAM starts at address 0x1000. The TMC8460 Process Data RAM has a size of 16kBytes.

Table 128: Process Data RAM (0x1000:0x4FFF)

Bytes	Description	ECAT	PDI	Reset Value
----	Process Data RAM	(r/w)	(r/w)	Random/undefined

NOTE (r/w): Process Data RAM is only accessible if EEPROM was correctly loaded (register 0x0110.0 = 1).

Two constant memory blocks are used for data to and from the MFCIO block when at least one register is enabled.

The first block from 0x4000 to 0x405F contains the data to be written into the MFCIO block, the second block from 0x4800 to 0x4823 contains data read from the MFCIO block.

For compatibility, 2 byte registers are placed at even addresses, 4 and 8 byte registers are placed at addresses 0x...0, 0x...4, 0x...8 or 0x...C. To match these offsets, padding bytes are added in a few places. Data written to these padding bytes in the first memory block are ignored, the padding bytes of the second block read 0x00.

To use this data, SyncManagers (SM) should be set up. For simplicity, one SM can be set up for each memory area to span the whole area. In case only a few registers are used, the SM can span just the area that is actually used.

Examples for SM using only smaller memory chunks are shown in the next two segments

5.53.1 MFCIO Block ECAT Write Data Memory Block (0x4000:0x405F)

Data in this memory block is written by the ECAT master to the MFCIO block.

Table 129: MFCIO Block ECAT Write Data Memory Block (0x4000:0x405F)

Start Address	Size (bytes)	MFCIO Register Name
0x4000	2	ENC_MODE
0x4004	4	ENC_X
0x4008	4	ENC_CONST
0x400C	8	SPI_TX_DATA
0x4014	2	SPI_CONF
0x4016	1	SPI_LENGTH
0x4017	1	SPI_TIME
0x4018	4	SD_SR
0x401C	4	SD_ST
0x4020	2	SD_SL
0x4022	2	SD_DLY
0x4024	1	SD_CFG
0x4026	2	PWM_MAXCNT
0x4028	2	PWM_CHOPMODE
0x402A	1	PWM_ALIGNMENT
0x402B	1	PWM_POLARITIES
0x402C	2	PWM_VALUE_0
0x402E	2	PWM_VALUE_1
0x4030	2	PWM_VALUE_2

Start Address	Size (bytes)	MFCIO Register Name
0x4032	2	PWM_CNTRSHIFT_0
0x4034	2	PWM_CNTRSHIFT_1
0x4036	2	PWM_CNTRSHIFT_2
0x4038	2	PWM_PULSE_A
0x403A	2	PWM_PULSE_B
0x403C	1	PWM_PULSE_LENGTH
0x403D	1	PWM_BBM_H
0x403E	1	PWM_BBM_L
0x4040	2	GPO_OUT_VAL
0x4042	1	GPIO_CONFIG
0x4044	2	IRQ_CFG
0x4048	4	WD_TIME
0x404C	1	WD_CFG
0x4050	8	WD_OUT_MASK_POL
0x4058	4	WD_OE_POL
0x405C	4	WD_IN_MASK_POL

Table 130: Padding bytes

Start Address	Number of padding bytes
0x4002	2
0x4025	1
0x403F	1
0x4043	1
0x4046	2
0x404D	3

Example:
To use only the PWM unit, a smaller SM can be set up with a start address of 0x4026 and a size of 25 bytes.

5.53.2 MFCIO Block ECAT Read Data Memory Block (0x4800:0x4823)

Data in this memory block is written by the MFCIO block and read by the ECAT master.

Table 131: MFCIO Block ECAT Read Data Memory Block (0x4800:0x4823)

Start Address	Size (bytes)	MFCIO Register Name
0x4800	1	ENC_STATUS
0x4804	4	ENC_X
0x4808	4	ENC_LATCH
0x480C	8	SPI_RX_DATA
0x4814	1	SPI_STATUS
0x4818	4	SD_SC
0x481C	1	GPI_IN_VAL
0x481E	2	IRQ_FLAGS
0x4820	4	WD_MAX

Table 132: Padding bytes

Start Address	Number of padding bytes
0x4801	3
0x4815	3
0x481D	1

Example:
When only the encoder position (ENC_X) is required via EtherCAT, a SyncManager starting at 0x4804 with a length of 4 bytes can be set up.

6 MFCIO Block Register and Functional Description

6.1 MFCIO Block General Information

The MFCIO block is a separate independent function block next to the main ESC data path. It connects to the ESC via a configuration interface and a memory bridge. It comes with its own SPI interface (MFC CTRL SPI) to connect to an application controller. Detailed information on the SPI interface protocol and characteristics is given in section 2.3.
Besides the configuration interface and the memory bridge the MFCIO block uses various status and control signals coming from the ESC.

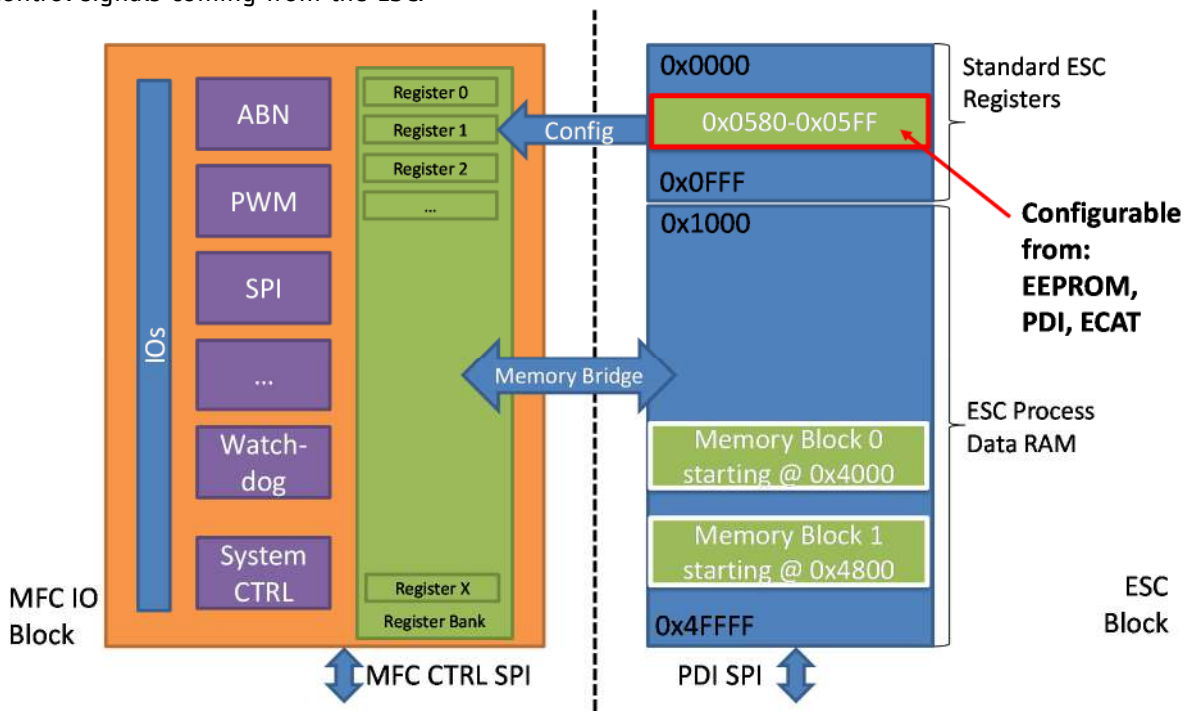


Figure 28 - MFCIO block interfaces to ESC PDRAM

The MFCIO block and its functions are based on the register set summarized in 6.2.
Which functional unit is used and which not depends on the application. All functions have dedicated IO pins on the TMC8460 and can be used at in parallel.

The MFCIO block can be accessed by the EtherCAT master and by a local MCU at the same time. However, a single register of the MFCIO block can be accessed either only by the ECAT interface or only by the MFC CTRL SPI interface.
The access rights as well as the update trigger source for all registers are configurable on a per register base. The configuration of the MFCIO access rights and update trigger sources is located in the ESC RAM at addresses 0x0580:0x05FF (ESC Parameter RAM). This configuration data is made available to the MFCIO block using the configuration interface.

If an MFCIO register is configured to be accessed by the ECAT interface, it is directly mapped to a fixed position in a memory block in the PDRAM using the memory bridge.
Two memory blocks (MB0 and MB1) are defined in the PDRAM – one for writable registers and one for read-only registers – to have continuous memory regions for which SyncManagers can be configured to care for, e.g., data consistency.

The MFCIO register contents of all readable registers that can be accessed from ECAT interface are only updated if the actual state of the EtherCAT state machine is SAFEOP or OP.
The MFCIO register contents of all writable registers that can be accessed from ECAT interface are only updated if the actual state of the EtherCAT state machine is OP.

All functional output signals of the MFCIO block are linked to the OP state of the EtherCAT state machine. As long as the actual state is not OP, all functional output states are not driven but in high impedance state.

6.2 MFCIO Block Address Space Overview

The MFCIO block provides functionality useful in an embedded system with focus on motor and motion control.

Table 133 : MFCIO Block Register Overview

Reg. no.	Register name	MFC block	R/W	Configuration address (EEPROM)	Configuration address (PDRAM)	Memory Block 0 Address (PDRAM)	Memory Block 1 Address (PDRAM)	Data size
0	ENC_MODE	ABN-Decoder	W	0x0084	0x0580	**0x4000**	-	2
1	ENC_STATUS		R	0x0085	0x0581	-	**0x4800**	1
2	ENC_X		W	0x0086	0x0582	0x4004	-	4
3	ENC_X		R	0x0087	0x0583	-	0x4804	4
4	ENC_CONST		W	0x0088	0x0584	0x4008	-	4
5	ENC_LATCH		R	0x0089	0x0585	-	0x4808	4
6	SPI_RX_DATA	SPI-Master	R	0x008A	0x0586	-	0x480C	8
7	SPI_TX_DATA		W	0x008B	0x0587	0x400C	-	8
8	SPI_CONF		W	0x008C	0x0588	0x4014	-	2
9	SPI_STATUS		R	0x008D	0x0589	-	0x4814	1
10	SPI_LENGTH		W	0x008E	0x058A	0x4016	-	1
11	SPI_TIME		W	0x008F	0x058B	0x4017	-	1
12	SD_SR	Step/Dir-Generator	W	0x0090	0x058C	0x4018	-	4
13	SD_SC		R	0x0091	0x058D	-	0x4818	4
14	SD_ST		W	0x0092	0x058E	0x401C	-	4
15	SD_SL		W	0x0093	0x058F	0x4020	-	2
16	SD_DLY		W	0x0094	0x0590	0x4022	-	2
17	SD_CFG		W	0x0095	0x0591	0x4024	-	1
18	PWM_MAXCNT	3-ch PWM	W	0x0096	0x0592	0x4026	-	2
19	PWM_CHOPMODE		W	0x0097	0x0593	0x4028	-	2
20	PWM_ALIGNMENT		W	0x0098	0x0594	0x402A	-	1
21	PWM_POLARITIES		W	0x0099	0x0595	0x402B	-	1
22	PWM_VALUE_0		W	0x009A	0x0596	0x402C	-	2
23	PWM_VALUE_1		W	0x009B	0x0597	0x402E	-	2
24	PWM_VALUE_2		W	0x009C	0x0598	0x4030	-	2
25	PWM_CNTRSHIFT_0		W	0x009D	0x0599	0x4032	-	2
26	PWM_CNTRSHIFT_1		W	0x009E	0x059A	0x4034	-	2
27	PWM_CNTRSHIFT_2		W	0x009F	0x059B	0x4036	-	2
28	PWM_PULSE_A		W	0x00A0	0x059C	0x4038	-	2
29	PWM_PULSE_B		W	0x00A1	0x059D	0x403A	-	2
30	PWM_PULSE_LENGTH		W	0x00A2	0x059E	0x403C	-	1
31	PWM_BBM_H		W	0x00A3	0x059F	0x403D	-	1
32	PWM_BBM_L		W	0x00A4	0x05A0	0x403E	-	1
33	GPO_OUT_VAL	GPIO	W	0x00A5	0x05A1	0x4040	-	2
34	GPI_IN_VAL		R	0x00A6	0x05A2	-	0x481C	1
35	GPIO_CONFIG		W	0x00A7	0x05A3	0x4042	-	1
36	IRQ_CFG	MFCIO-IRQ	W	0x00A8	0x05A4	0x4044	-	2
37	IRQ_FLAGS		R	0x00A9	0x05A5	-	0x481E	2
38	WD_TIME	GPIO-Watch-dog	W	0x00AA	0x05A6	0x4048	-	4
39	WD_CFG		W	0x00AB	0x05A7	0x404C	-	1
40	WD_OUT_MASK_POL		W	0x00AC	0x05A8	0x4050	-	8
41	WD_OE_POL		W	0x00AD	0x05A9	0x4058	-	4
42	WD_IN_MASK_POL		W	0x00AE	0x05AA	0x405C	-	4
43	WD_MAX		R	0x00AF	0x05AB	-	0x4820	4
44	AL_STATE_OVERRIDE	Override	W	-	-	-	-	1

The table above gives an overview of all MFCIO block registers. Read-only registers are marked blue, registers that are writable (write-only via EtherCAT) are marked red.

The register number (Reg. No.) is required for addressing registers via the MFC CTRL SPI interface. When using EtherCAT access, this is not necessary.

The column *Configuration address (EEPROM)* shows the location where each registers configuration byte is stored in the EEPROM. For the full EEPROM map, see 6.3.

On startup, the EEPROM configuration is copied into the PDRAM, starting at address 0x0580. The address for each register is shown in the column *Configuration address (PDRAM)*. The configuration for a register can be changed in this address space. If no configuration is stored in the EEPROM, the MCU or the EtherCAT master can set the configuration.

The data to be written to the registers and the data read from the registers can be found at the addresses given in the *Memory Block x Address (PDRAM)* columns. These addresses are used for EtherCAT access. When accessing the PDRAM, the MFCIO block has the same behavior as an external MCU, so when SyncManagers are set up on the two memory blocks, an MCU does not have access to these regions using the PDI SPI interface.

All registers can be read at any time via the MFC CTRL SPI, if EtherCAT access for a writable register is disabled, this register can also be written via the MFC CTRL SPI.

The register 44 (AL_STATE_OVERRIDE) has a special function and can only be accessed via MFC CTRL SPI. Usually the outputs of the MFCIO block are only active when the EtherCAT slave is in operational state. With this register they can be activated regardless of the EtherCAT state.

The base addresses of the two memory blocks are fix and this portion of the PDRAM can only be used for MFCIO data when MFCIO registers are configured for ECAT access.
The memory blocks should be managed by SyncManagers, which must be configured separately using the ESC's configuration options.

6.3 MFCIO Block EEPROM Parameter Map

The configuration data for the MFCIO block registers can be stored in the I2C EEPROM to be automatically loaded into the ESC after power-up or after reset.
This configuration category must be the first in the EEPROM, located at address 0x0080 (word 0x40). The category type must be 1 so that the TMC8460 loads the configuration into the memory area starting at 0x0580.
This section describes the part of the EEPROM content and XML/ESI file that is used to configure the MFCIO block registers at power-up or after reset.

Table 134 : EEPROM Parameter Map & ESC RAM Address Mapping for TMC8460-BI

EEPROM Address	Function	Value
General Category Information (Values must not be changed)		
0x0080:0x0081	Category Type	0x0001
0x0082:0x0083	Category Data Size	0x002C
MFCIO Register Configuration		
0x0084:0x0089	Encoder Unit Registers	Depends on the required configuration, 0x00 for unused registers. See sections 6.4 for details
0x008A:0x008F	SPI Unit Registers	
0x0090:0x0095	Step/Direction Unit Registers	
0x0096:0x00A4	PWM Unit Registers	
0x00A5:0x00A7	General Purpose I/O Unit Registers	
0x00A8:0x00A9	MFCIO IRQ Registers	
0x00AA:0x00AF	MFCIO Watchdog Registers	

When (re)configuring the EEPROM from an EtherCAT master system special care must be taken. Not every master allows writing a category 1 entry to the EEPROM. There are different ways to write this into the EEPROM for automatically loading the MFCIO block configuration at power-up:

- Use preprogrammed EEPROMs.
- Use a different category, e.g., 2049, first. Then overwrite the upper byte with 0 with a single EEPROM byte write.
- Use the local microcontroller (if available) and have it connected to the EEPROM via I2C. Hold the TMC8460 in reset while programming the EEPROM first. Afterwards tristate the local I2C bus so that the TMC8460 has control over the IC2 interface and release the TMC8460 from reset.

6.4 MFCIO Register Configuration

Each register of the MFCIO block can be configured using an 8-bit EEPROM entry. Sections 6.2 and 6.3 provide information on the individual EEPROM addresses for configuring each MFCIO block register.
The following parameters and syntax are used for every MFCIO register for configuration.

Table 135 : MFCIO register configuration byte

bit	7	6	5	4	3	2	1	0
function	Unused / reserved			Enable ECAT access	Shadow trigger source			

MFCIO block registers have 2 stages – a bank of shadow registers and the main registers.
When writing or reading data to/from the MFCIO block register there is always a shadow register in between. The shadow registers always hold the latest data written by the MFCIO block or by the ECAT interface. The content of the shadow registers is copied to the main register depending upon configurable trigger sources.
Bits 3..0 define the when the transfer between a shadow register and the main register of the MFCIO unit happens. For write registers the data is copied from the shadow register into the MFC unit register, for read registers, the data is copied from the MFC unit register into the shadow register. Each MFC unit register has a corresponding shadow register.
For write registers, Bit 4 determines whether the register will be accessed from the ECAT master (1) or from the MFC CTRL SPI interface (0). Only one interface can write.
For read registers, Bit 4 determines whether the register is also copied to PDRAM to be read by the ECAT master.

! Read access (not write) from the MFC CTRL SPI interface is always possible both for read and write registers.

Table 136 : MFCIO register Shadow trigger source configuration

Bits 3..0	Decimal value	Trigger source
0000	0	Always triggered
0001	1	SYNC0 signal
0010	2	SYNC1 signal
0011	3	LATCH0 signal
0100	4	LATCH1 signal
0101	5	Start Of Frame (SOF)
0110	6	End Of Frame (EOF)
0111	7	PDI-SPI Chip Select
1000	8	PDI-SPI Chip Deselect
1001	9	MFC-CTRL-SPI Chip Select
1010	10	MFC-CTRL-SPI Chip Deselect

1011	11	Before shadow register handling cycle
1100	12	After shadow register handling cycle
1101	13	Trigger at start of MFC PWM cycle
1110	14	Trigger when data value changes
1111	15	Always triggered

6.5 MFCIO Block Example Configuration and Example XML/ESI File

This section provides two examples on how to parameterize the Trinamic MFCIO block for a specific application. It provides examples of the final XML/ESI files.

The manual configuration requires the following steps:

1.) Define which functional units and their registers shall be used from ECAT master side set the RegisterEnable-Bits in the respective register configuration.

    Configuration of the Trigger Source for each Configuration or Data Register of the MFCIO block functional units.
    This defines how and when the content of a register is transferred to the slave controller's SyncManager memory areas for readable registers and how and when new content/data is updated to a writable register inside the MFCIO block.

2.) The base addresses of the used Memory Blocks are fix for the TMC8460.
    Memory Block MB0 is used for write access data.
    Memory Block MB1 is used for read only data.
    In addition, the explicit address of registers inside the memory blocks are fix.
    For each memory block, a sync manager shall be configured using the start and end address of the memory block. Usually not all functional units and registers are required. It is also possible to use only a contiguous part of the memory block in a SyncManager to reduce the amount of data.

3.) The XML entry for the ESI-file including the EEPROM configuration string is generated.

Later on, this will be done using a graphical tool that allows for simple and easy visual configuration.

6.5.1 Example 1

TMC8460 with active device emulation used as an ABN decoder

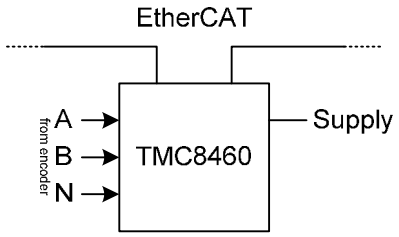


Figure 29 - Connections to TMC8460 in Example 1

In the first example, the device emulation feature is used instead of adding a microcontroller for the EtherCAT state machine. In this example, the status is not evaluated, so the ENC_STATUS register is not enabled and the encoder position will never be set, so ENC_X (W) is also not enabled.

Step 1:

The following registers should be accessed via EtherCAT:

Register 0, ENC_MODE, Shadow register triggered when value changes	Configuration 0x1E
Register 3, ENC_X (R), Shadow register is transparent (always triggered)	Configuration 0x10
Register 4, ENC_CONST, Shadow register triggered when value changes	Configuration 0x1E
Register 5, ENC_LATCH, Shadow register is transparent (always triggered)	Configuration 0x10

The configuration data for the whole MFC block is (ABN-Decoder block underlined):

0x1E0000101E100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000

Since the reset value of the configuration block is 0x00, the trailing configuration bytes for registers 6 to 43 can be omitted in the EEPROM.

Step 2:
The memory areas where all register values for the MFC-Block are located are shown below.

Base Address	+0x0	+0x1	+0x2	+0x3	+0x4	+0x5	+0x6	+0x7	+0x8	+0x9	+0xA	+0xB
0x4000	ENC_MODE		Padding		ENC_X (W)				ENC_CONST			
0x4800	ENC_STATUS	Padding			ENC_X (R)				ENC_LATCH (R) (7:0)			

In this example, the ENC_X (W) and ENC_STATUS registers are unused. While the ENC_X (W) register is in the middle of the contiguous block, the ENC_STATUS register and the next three padding bytes can be excluded from the SyncManager address range.
One SyncManager covers the writable registers with the address range 0x4000:0x400B.
The other SyncManager covers only the smaller address range of the readable registers 0x4804:0x480B.

The definition of FMMUs, SMs and the PDOs in the ESI file looks like this:

```
<Fmmu Sm="0">Outputs</Fmmu>
<Fmmu Sm="1">Inputs</Fmmu>
<Sm DefaultSize="12" StartAddress="#x4000" ControlByte="#x04" Enable="1">Outputs</Sm>
<Sm DefaultSize="8" StartAddress="#x4804" ControlByte="#x00" Enable="1">Inputs</Sm>

<RxPdo Mandatory="true" Fixed="true" Sm="0">
  <Index>#x1601</Index>
  <Name>Writable-Registers</Name>
  <Entry>
    <Index>#x7000</Index>
    <SubIndex>0</SubIndex>
    <BitLen>16</BitLen>
    <Name>ENC_MODE</Name>
    <DataType>UINT</DataType>
  </Entry>
  <Entry>
    <Index>#x7001</Index>
    <SubIndex>0</SubIndex>
    <BitLen>16</BitLen>
    <Name>Padding</Name>
    <DataType>UINT</DataType>
  </Entry>
  <Entry>
    <Index>#x7002</Index>
    <SubIndex>0</SubIndex>
    <BitLen>32</BitLen>
    <Name>ENC_X</Name>
    <DataType>UDINT</DataType>
  </Entry>
  <Entry>
    <Index>#x7003</Index>
    <SubIndex>0</SubIndex>
    <BitLen>32</BitLen>
    <Name>ENC_CONST</Name>
    <DataType>UDINT</DataType>
  </Entry>
</RxPdo>


<TxPdo Mandatory="true" Fixed="true" Sm="1">
  <Index>#x1a00</Index>
  <Name>Readable-Registers</Name>
  <Entry>
    <Index>#x6004</Index>
    <SubIndex>0</SubIndex>
    <BitLen>32</BitLen>
    <Name>ENC_X</Name>
    <DataType>UDINT</DataType>
  </Entry>
  <Entry>
    <Index>#x6005</Index>
    <SubIndex>0</SubIndex>
    <BitLen>32</BitLen>
    <Name>ENC_LATCH</Name>
    <DataType>UDINT</DataType>
```



```
    </Entry>
  </TxPdo>
```

Step 3:
Additional to the MFC Block Configuration, the ESI-File contains the configuration for the EtherCAT Slave Controller itself. This configuration is later read from the EEPROM on reset.
In the order, it is defined in the ConfigData section of the ESI file:
0x0003: No PDI Interface, Enhanced Link Detection and Device Emulation are enabled
0x0000: No configuration for PDI and Sync/Latch IOs since they are disabled
0x0000: Sync-Pulse length unconfigured (disabled)
0x0000: Extended PDI configuration is unused
0x0000: Station alias
0x0000: reserved (0)
0x0000: reserved (0)

The complete Eeprom section of the ESI file looks like this:

```
<Eeprom>
  <ByteSize>2048</ByteSize>
  <ConfigData>0003000000000000000000000000</ConfigData>
  <Category>
    <CatNo>1</CatNo>
    <Data>1E0000101E10</Data>
  </Category>
</Eeprom>
```

Step 5:
To use the unit, ENC_MODE and ENC_CONST must be set via EtherCAT, a basic setting for testing that everything is working correctly would be:
ENC_MODE = 0x18
ENC_CONST = 0x00010000

For every A/B quadrature step, ENC_X counts up respectively down by one. In case of an N event, the position is latched into ENC_LATCH.

6.5.2 Example 2

TMC8460 used to generate step/direction signals via EtherCAT and a microcontroller for the EtherCAT state machine and to control the PWM block.

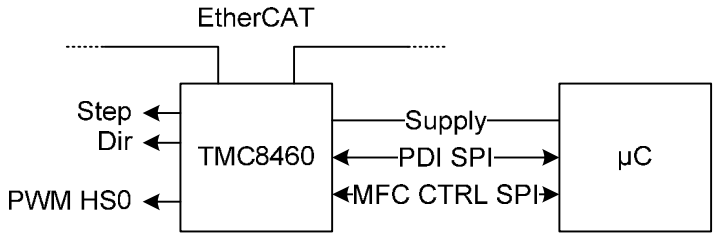


Figure 30 - Connections to TMC8460 in Example 2

In this example the MFC block configuration consists of two parts, the EtherCAT enabled access to the step/direction unit and the access to the PWM unit via the MFC CTRL SPI.

Step 1:
Unused registers are 16 (SD_DLY), 23, 24 (PWM_VALUE_1 and 2), 26, 27 (PWM_CNTRSHIFT_1 and 2), 28, 29 (PWM_PULSE_A and B) and 30 (PWM_PULSE_LENGTH). The configuration for these registers is 0x00.

Since configuration bytes for all registers must be included unless no more active registers are following, the configuration data has to start with the bytes for the unused first 12 registers (0-11): `0x000000000000000000000000`.

The configuration for the step/direction registers (12-17) with EtherCAT access is straightforward. All writable registers except for 16 – SD_DLY – should be updated when the value changes, their configuration byte is 0x1E. The readable register 13 – SD_SC – can be used with a transparent shadow register and configuration 0x10. This part of the configuration data is `0x1E101E1E001E`.
In the PWM unit, which is only accessed via SPI, the bit for enabling EtherCAT access stays 0. All registers are writable and the shadow registers should be updated when the data changes. With the exception of the registers mentioned above, all configuration bytes are 0x0E, resulting in this configuration data: `0x0E0E0E0E0E00000E00000000000E0E`.
All other configuration bytes are 0x00 after that and can be omitted from the complete configuration data:
`0x000000000000000000000000001E101E1E001E0E0E0E0E0E00000E00000000000E0E00000000000000000000`

Step 2:
The memory areas that are used via EtherCAT and thus need SyncManagers cover only the Step/Direction unit.

Base Address	+0x0	+0x1	+0x2	+0x3	+0x4	+0x5	+0x6	+0x7	+0x8	+0x9	+0xA	+0xB	+0xC
0x4018	SD_SR				SD_ST				SD_SL		SD_DLY		SD_CFG
0x4818	SD_SC												

The writable registers, covered by the first SyncManager, are in the memory range 0x4018:0x4024. The two bytes of the unused SD_DLY register are in this area but the padding byte after the SD_CFG register (0x4025) can be excluded from the SyncManager.
The only readable register, which is the only one in the second SyncManager, is in the memory range 0x4818:0x481B.

The FMMU, SM and PDO configuration from the ESI file looks like this:

```
<Fmmu Sm="0">Outputs</Fmmu>
<Fmmu Sm="1">Inputs</Fmmu>
<Sm DefaultSize="13" StartAddress="#x4018" ControlByte="#x04" Enable="1">Outputs</Sm>
<Sm DefaultSize="4" StartAddress="#x4818" ControlByte="#x00" Enable="1">Inputs</Sm>


<RxPdo Mandatory="true" Fixed="true" Sm="0">
  <Index>#x1601</Index>
  <Name>Writable-Registers</Name>
  <Entry>
     <Index>#x7008</Index>
     <SubIndex>0</SubIndex>
     <BitLen>32</BitLen>
     <Name>SD_SR</Name>
     <DataType>UDINT</DataType>
  </Entry>
  <Entry>
     <Index>#x7009</Index>
     <SubIndex>0</SubIndex>
     <BitLen>32</BitLen>
     <Name>SD_ST</Name>
     <DataType>UDINT</DataType>
  </Entry>
  <Entry>
     <Index>#x7010</Index>
     <SubIndex>0</SubIndex>
     <BitLen>16</BitLen>
     <Name>SD_SL</Name>
     <DataType>UINT</DataType>
  </Entry>
  <Entry>
     <Index>#x7011</Index>
     <SubIndex>0</SubIndex>
     <BitLen>16</BitLen>
     <Name>SD_DLY</Name>
     <DataType>UINT</DataType>
  </Entry>
  <Entry>
     <Index>#x7012</Index>
     <SubIndex>0</SubIndex>
     <BitLen>8</BitLen>
     <Name>SD_CFG</Name>
     <DataType>USINT</DataType>
  </Entry>
</RxPdo>


<TxPdo Mandatory="true" Fixed="true" Sm="1">
  <Index>#x1a00</Index>
  <Name>Readable-Registers</Name>
  <Entry>
```



```
      <Index>#x6010</Index>
      <SubIndex>0</SubIndex>
      <BitLen>32</BitLen>
      <Name>SD_SC</Name>
      <DataType>UDINT</DataType>
    </Entry>
  </TxPdo>
```

Step 4:
The configuration of the EtherCAT Slave Controller, which is read from the EEPROM, is `0x0502030000000000000000000000`.
0x0503: SPI PDI, Enhanced Link Detection enabled
0x0300: SPI mode 3
0x0000: Sync-Pulse length unconfigured (disabled)
0x0000: Extended PDI configuration is 0
0x0000: Station alias
0x0000: reserved (0)
0x0000: reserved (0)

The complete Eeprom section of the ESI file looks like this:

```
<Eeprom>
  <ByteSize>2048</ByteSize>
  <ConfigData>0003000000000000000000000000</ConfigData>
  <Category>
    <CatNo>1</CatNo>
    <Data>00000000000000000000000001E101E1E001E0E0E0E0E0E00000E00000000000E0E</Data>
  </Category>
</Eeprom>
```

Step 5:
To set up the Step/Direction-Unit via EtherCAT to generate a constant step frequency, the following PDOs must be set (Steprate and –length are example values):
SD_SR = 180000
SD_SL = 100
SD_CFG = 2

The Stepcount PDO (SD_SC) shows the number of steps made.

The PWM registers are accessible only from the microcontroller via the MFC_CTRL_SPI interface using the register number and the addressing scheme described in Section 2.3.3
Since it is not configured to be used via EtherCAT, the register SD_DLY can also be written from the microcontroller.

6.6 MFCIO Incremental Encoder Unit

The incremental encoder unit allows the decoding of ABN quadrature signals with zero pulse up to a high speed without CPU load. The parameters for the unit are configurable to fit a wide range of quadrature encoders. The internal accumulator works as a 48 bit fixed point value with a 32 bit integer part which is visible through the encoder position register and a 16 bit fractional part which is only used internally.

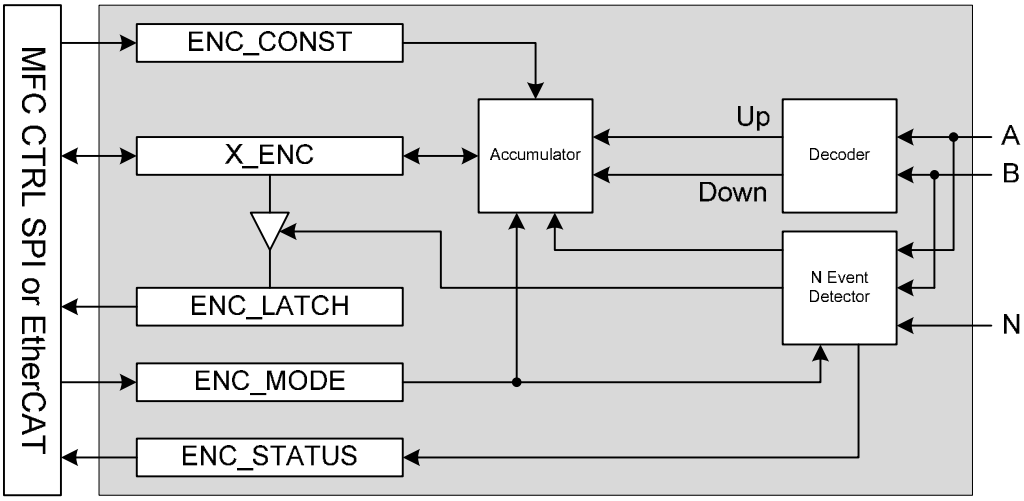


Figure 31 - Block structure of the incremental encoder unit

The basic configuration requires the register ENC_CONST to be set to a value matching the encoder. Advanced configuration options are available in the ENC_MODE register for encoders requiring an encoder constant with a decimal fraction and to configure the N event detector and the actions taken at a N event.

6.6.1 Incremental Encoder Unit Signals

Table 137 : MFCIO incremental encoder unit signals

Signal	I/O	Function
A	In	Quadrature encoder signal A
B	In	Quadrature encoder signal B
N	In	Encoder zero pulse N

6.6.2 Incremental Encoder Unit Register Set

Table 138 : MFCIO incremental encoder unit register set

MFCIO direct access register address #	Register Name	R/W	Size [bit]	Description
0	ENC_MODE	W	16	Configuration of incremental encoder unit Bit 15: clear_status — Clears the status flag when written to 1 Bit 10: enc_sel_decimal — Fractional part of ENC_CONST is decimal Bit 8: clr_enc_x — ENC_X is set to 0 at N event

				Bit 7: neg_edge becomes inactive	N event is generated when N input
				Bit 6: pos_edge becomes active	N event is generated when N input
				Bit 5: clr_once	N event is generated only once when all conditions are met
				Bit 4: clr_cont	N event is generated every time all conditions are met
				Bit 3: ignore_AB B signals	N event is generated regardless of A and
				Bit 2: pol_N	Active polarity for N input
				Bit 1: pol_B	Required B input polarity for N event
				Bit 0: pol_A	Required B input polarity for N event
1	ENC_STATUS	R	8	Status flag showing that N event was generated Bit 0: Status bit	
2	X_ENC (W)	W	32	Register to set a new encoder position	
3	X_ENC (R)	R	32	Encoder position	
4	ENC_CONST	W	32	Encoder constant in 16 bit integer and 16 bit fractional part	
5	ENC_LATCH	R	32	Latched encoder position	

6.6.3 ENC_MODE

clear_status (Bit 15)
The status flag is not cleared by reading ENC_STATUS but needs to be reset by writing this bit to 1.

enc_sel_decimal (Bit 10)
With this bit set to 0, the encoder constant is interpreted as a fixed point binary number with the lower 16 bits representing the fractional part.
Whis this bit set to 1, the lower 16 bits of the encoder constant represent 1/10000th step. The range for the fractional part in this case is 0 <= x <= 9999 and should not be set higher. A fractional part of 10000 equals an additional integer part of 1.

clr_enc_x (Bit 8)
If this bit is set to 1, the encoder counter (ENC_X) is reset to 0 in case of an N event.
Regardless of this bit, the value of the encoder counter is always transferred to the ENC_LATCH register at an N event.

The N event is generated when some configurable conditions are met:
- If the N input matches the pol_N bit (**Bit 2**) the input signal is considered active.
- If both pos_edge (**Bit 6**) and neg_edge (**Bit 7**) are 0, the whole active signal is considered as an event, if pos_edge is 1, an event is generated on the inactive->active transition of the signal, if neg_edge is 1, an event is generated at the active->inactive transition. Two events are generated if pos_edge and neg_edge are both 1.
- If ignore_AB (**Bit 3**) is 0, the event is only passed through when the A and B inputs match the pol_A (**Bit 0**) and pol_B (**Bit 1**) bits. If ignore_AB is 1, the event is always passed through.
- Finally, either clr_once (**Bit 5**) or clr_cont (**Bit 4**) needs to be set to either pass through only the next event or all following events.

6.6.4 ENC_STATUS

Only Bit 0 is used in this register, the set status bit indicates that an N event occurred. ENC_STATUS is not clear-on-read but must be cleared by setting Bit 15 of ENC_MODE.

6.6.5 ENC_X (W)

Writing a value to this register loads the internal position accumulator to this value.

6.6.6 ENC_X (R)

This register contains the encoder position

6.6.7 ENC_CONST

The encoder constant is a 32-bit fixed-point value that is added to or subtracted from the internal accumulator. Depending on the bit enc_sel_decimal in the ENC_MODE register, the fractional part is interpreted differently:
When enc_sel_decimal is 0:
(ENC_CONST_INT+(ENC_CONST_FRACT/65536)) is added to or subtracted from the accumulator
When enc_sel_decimal is 1:
(ENC_CONST_INT+(ENC_CONST_FRACT/10000)) is added to or subtracted from the accumulator

6.6.8 ENC_LATCH

The encoder position X_ENC is latched into this register on each N event.
Encoder Configuration Example:

ENC_MODE = 0x001c
Everytime while N is active (high), generate an N event regardless of the A and B inputs. Only latch the value, do not clear X_ENC.
clr_enc_x = 0
neg_edge = 0
pos_edge = 0
clr_once = 0
clr_cont = 1
ignore_AB = 1
pol_N = 1
pol_B = 0
pol_A = 0

ENC_CONST = 0x00010000 (=65536)
Encoder constant = 1.0 (→ each quadrature signal change counts either up or down)

Encoder constant examples for different encoders on a motor with 200 fullsteps/revolution and 256 microsteps/fullstep (51200 microsteps/revolution) to match encoder position and microstep position:

Table 139 : list of typical encoder constants

Encoder resolution	Required encoder factor	ENC_CONST	Comment
200	256	0x01000000	No fractional part
360	142.2222 = 9320675.5555 / 2^16 = 1422222.2222 / 10000	0x008E38E4	No exact match possible. Since the absolute error of the binary representation is smaller, this value should be used.
500	102.4 = 6710886.4 / 2^16 = 1024000 / 10000 = 102 + (4000/10000)	0x00660FA0	Exact match with decimal setting enc_sel_decimal = 1
1000	51.2 = 51 + (2000/10000)	0x003307D0	Exact match with decimal setting enc_sel_decimal = 1
1024	50	0x00320000	No fractional part
4000	12.8 = 12 + (8000/10000)	0x000C1F40	Exact match with decimal setting enc_sel_decimal = 1
4096	12.5	0x000C8000	Binary fractional part

	= 12 + (2^15/2^16)		
16384	3.125 = 3 + (2^13/2^16)	0x00032000	Binary fractional part

6.7 MFCIO SPI Master Unit

The SPI Master Unit provides an interface for up to four SPI slaves with a theoretically unlimited datagram length using multiple accesses.

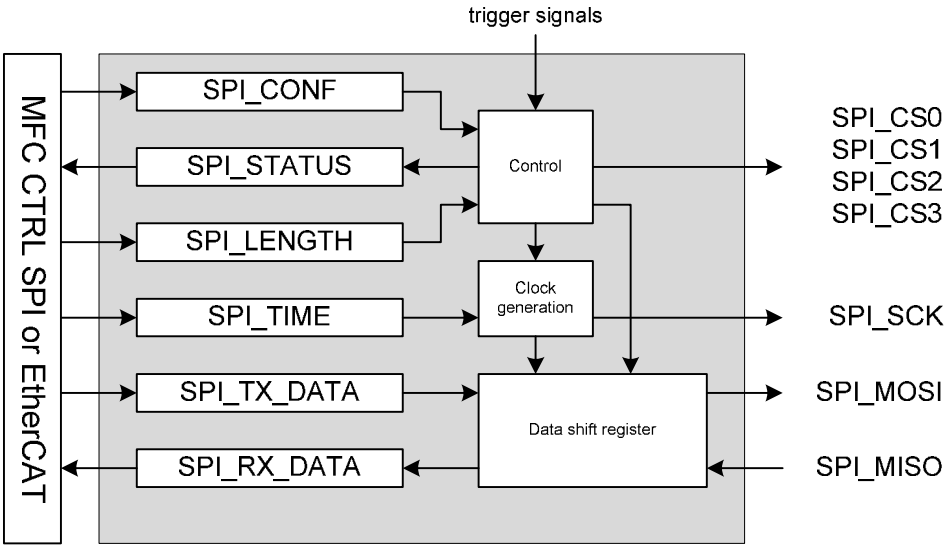


Figure 32 - Block structure of SPI Master Unit

The basic configuration requires setting the SPI frequency/bit length, the datagram length and the SPI mode (clock polarity and phase). Extended settings are a special start-of-transmission trigger linked to the PWM unit, the bit order, selection of one of the four SPI slaves and datagram length extension.

6.7.1 SPI Master Unit Signals

Table 140 : MFCIO master unit signals

Signal	I/O	Function
SPI_CS0…SPI_CS3	Out	Chip select lines 0..3
SPI_SCK	Out	SPI clock output
SPI_MOSI	Out	SPI data output (Master Out Slave In)
SPI_MISO	In	SPI data input (Master In Slave Out)

6.7.2 SPI Master Unit Register Set

Table 141 : MFCIO SPI master unit register set

MFCIO direct access register address #	Register Name	R/W	Size [bit]	Description
6	SPI_RX_DATA	R	64	Received data from last SPI transfer
7	SPI_TX_DATA	W	64	Data to transmit on next SPI transfer
8	SPI_CONF	W	16	SPI Master configuration and control Bit 15: Start transfer once when this bit is set and trigger config is set to 7 Bits 10..8: Trigger config for transmission start values

				000: Start when data is written into TX register 001: Start on beginning of PWM cycle 010: Start on center of PWM cycle 011: Start on PWM A mark 100: Start on PWM B mark 101: Start on PWM A&B marks 111: Start on single trigger (Bit 15) Bit 6: SPI clock polarity Bit 5: SPI clock phase Bit 4: LSB first Bit 3: Keep CS low after transfer for transfers over 64bit Bits 1..0: Selection of SPI slave
9	SPI_STATUS	R	8	Bit 0: ready-indicator, if set a new transmission can be started Bits 7:1: unused
10	SPI_LENGTH	W	8	Bits 5..0: SPI datagram length = SPI_LENGTH+1 Bits 7..6: unused
11	SPI_TIME	W	8	Defines bit length and thereby the SPI frequency

6.7.3 SPI_RX_DATA

For SPI transfers with less than 64 bit, the upper bits of this register are unused.

6.7.4 SPI_TX_DATA

For SPI transfers with less than 64 bit, the upper bits of this register are unused.
Unless configured differently in SPI_CONF Bits 10..8, writing to this register starts the SPI transfer.

6.7.5 SPI_CONF

Bits 10..8 allow a configuration when the data transmission should start, they are interpreted as a 3 bit number:

- In the reset configuration 0, the transmission always starts when data is written to the SPI_TX register.
- The settings 1 to 5 link the start of the transmission to the PWM unit, allowing synchronization between the PWM cycle and for example a SPI ADC for current measurement. The trigger sources are the five PWM_PULSE signals that are also available as external signals. Please refer to Section 6.9 for details about these pulses.
- Setting 7 is a single shot trigger that starts only one transmission when **Bit 15** is written to 1.

Bit 6 and 5 define the clock polarity and phase of the SPI signals which define what the idle state of the SCK signal is and when output data is changed and when input data is sampled.

Table 142 : MFCIO SPI master unit SPI mode configuration

Clock polarity	Clock phase	SPI mode	MOSI change	MISO sample
0	0	0	SCK falling edge	SCK rising edge
0	1	1	SCK rising edge	SCK falling edge
1	0	2	SCK rising edge	SCK falling edge
1	1	3	SCK falling edge	SCK rising edge

Bit 4 reverses the bit order in the transmission, the least significant bit of SPI_TX_DATA (Bit 0) is transmitted first, the least significant bit of SPI_RX_DATA is the first received bit, the most significant bit of SPI_TX_DATA is transmitted last and the most significant bit of SPI_RX_DATA is the last bit received.

Bit 3 can be used for datagrams longer than 64 bit. With this bit set, the chip select line is held low after the transmission, allowing more transmissions in the same datagram. Before the last transmission, this bit must be set to 0 again so that the chip select line goes high afterwards, ending the datagram.

Bits 1 and 0 define which chip select line (which slave) is used for the next transmission.

6.7.6 SPI_STATUS

Bit 0 of this register is the Ready indicator for the SPI master unit. When this bit is set, a new transfer can be started. When this bit is 0 and the start of a new transfer is triggered, the trigger is ignored, the currently active transfer is finished but the new transfer is not started.

6.7.7 SPI_LENGTH

This register defines the SPI datagram length in bits. Any length from 1 to 64 bits is possible.

SPI datagram length = SPI_LENGTH+1

6.7.8 SPI_TIME

This register defines the bit length and thus the SPI clock frequency.
The duration of one SPI clock cycle can be calculated as t_SCK = (4+(2*SPI_TIME))/25MHz, the SPI clock frequency is f_SCK = 25MHz/(4+(2*SPI_TIME)).
The delay between the falling edge of CSN (becoming active) and the first SCK edge and the last SCK edge and the rising edge of CSN is always a half SCK clock cycle (t_SCK/2).

6.7.9 SPI Examples

TMC262 ON SPI CHANNEL 0

This example shows the configuration of the SPI master unit for a TMC262 as SPI slave 0 and the transfer of data to the TMC262's DRVCONF register.

Use 3.125 MHz SPI clock (25MHz/(4+(2*2))) = (25MHz/8)
    SPI_TIME <= 0x02
Use 20 bit datagrams
    SPI_LENGTH <= 0x13
Start on TX write, SPI-Mode 3, MSB first, single datagrams, Slave 0)
    SPI_CONF <= 0x0060
Wait until SPI-Master is ready
    while (SPI_STATUS & 0x01 != 0x01)
Write Data into TX register (e.g. TMC262 DRVCONF register, all 64bit are shown)
    SPI_TX_DATA <= 0x00000000000EF010
Wait until SPI-Master is ready
    while (SPI_STATUS & 0x01 != 0x01)
Read Data from RX register
    rxdatagram = SPI_RX_DATA

CHAIN OF 10 74XX595 SHIFT REGISTERS USED AS DIGITAL 80 OUTPUTS (GOOD EXAMPLE)

This example shows the transmission of a longer datagram, in this case 80 bits that are shifted into a chain of 74xx595 shift registers. The NCS of the SPI interface can be used as the storage clock of the 74xx595 to transfer the contents of the shift register into the storage register. The data that should be sent is 0x5555AAAA5555AAAA55AA.
It is recommended to split the data into two chunks of 40 bits each: 0x5555AAAA55 and 0x55AAAA55AA.

Configuration and first transmission

Use 6.25 MHz SPI clock (25MHz/(4+(2*0))) = (25MHz/4)
    SPI_TIME <= 0x00
Use a 40 bit datagram


```
        SPI_LENGTH <= 0x28
Start on TX write, SPI-Mode 3, MSB first, Keep CS low, Slave 0)
        SPI_CONF <= 0x0068
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
Write Data for the first 64 outputs into TX register
        SPI_TX_DATA <= 0x5555AAAA55
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
Start on TX write, SPI-Mode 3, MSB first, Drive CS high at the end, Slave 0)
        SPI_CONF <= 0x0060
Write Data for the last 16 outputs into TX register
        SPI_TX_DATA <= 0x55AAAA55AA
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
Next transmission with inverted data
Start on TX write, SPI-Mode 3, MSB first, Keep CS low, Slave 0)
        SPI_CONF <= 0x0068
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
Write Data for the first 64 outputs into TX register
        SPI_TX_DATA <= 0xAAAA5555AA
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
Start on TX write, SPI-Mode 3, MSB first, Drive CS high at the end, Slave 0)
        SPI_CONF <= 0x0060
Write Data for the last 16 outputs into TX register
        SPI_TX_DATA <= 0xAA5555AA55
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
```

CHAIN OF 10 74XX595 SHIFT REGISTERS USED AS DIGITAL 80 OUTPUTS (BAD EXAMPLE)

This bad example is the same as the previous one but with the non-recommended datagram split of 64 bits + 16 bit. This requires more communication since not only the SPI_CONF register needs to be changed between the SPI_TX_DATA writes but also the SPI_LENGTH register changes every time.

```
Configuration and first transmission
Use 6.25 MHz SPI clock (25MHz/(4+(2*0))) = (25MHz/4)
        SPI_TIME <= 0x00
Use a 64 bit datagram
        SPI_LENGTH <= 0x3F
Start on TX write, SPI-Mode 3, MSB first, Keep CS low, Slave 0)
        SPI_CONF <= 0x0068
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
Write Data for the first 64 outputs into TX register
        SPI_TX_DATA <= 0x5555AAAA5555AAAA
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
Use a 16 bit datagram (remaining outputs)
        SPI_LENGTH <= 0x0F
Start on TX write, SPI-Mode 3, MSB first, Drive CS high at the end, Slave 0)
        SPI_CONF <= 0x0060
Write Data for the last 16 outputs into TX register
        SPI_TX_DATA <= 0x55AA
Wait until SPI-Master is ready
```

```
        while (SPI_STATUS & 0x01 != 0x01)
```

Next transmission with inverted data

```
Use a 64 bit datagram
        SPI_LENGTH <= 0x3F
Start on TX write, SPI-Mode 3, MSB first, Keep CS low, Slave 0)
        SPI_CONF <= 0x0068
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
Write Data for the first 64 outputs into TX register
        SPI_TX_DATA <= 0xAAAA5555AAAA5555
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
Use a 16 bit datagram (remaining outputs)
        SPI_LENGTH <= 0x0F
Start on TX write, SPI-Mode 3, MSB first, Drive CS high at the end, Slave 0)
        SPI_CONF <= 0x0060
Write Data for the last 16 outputs into TX register
        SPI_TX_DATA <= 0xAA55
Wait until SPI-Master is ready
        while (SPI_STATUS & 0x01 != 0x01)
```


6.8 MFCIO Step Direction Unit

The MFC is equipped with a step-direction unit. Programming of the step pulse frequency occurs by writing an accumulation constant to a register. Toggle of the MSB of the accumulation register value generates an internal step pulse of one internal clock cycle. The direction signal is the MSB of the accumulation constant. Therefore, the sign of the accumulation constant defines the direction signal polarity. The step-to-direction timer (STP2DIR) takes care of possible external signal delay paths by programmable delay of the first step after write of accumulation constant. The pulse stretcher forms step and direction pulses of programmable length for adaption to external signal paths. The step direction unit can either run in free running mode just generating step pulses with programmed frequency. Alternatively, is can generate a defined number of step pulses with programmed frequency. An interrupt output signal IRQ_TARGET_REACHED indicates the reached target count of step pulses.

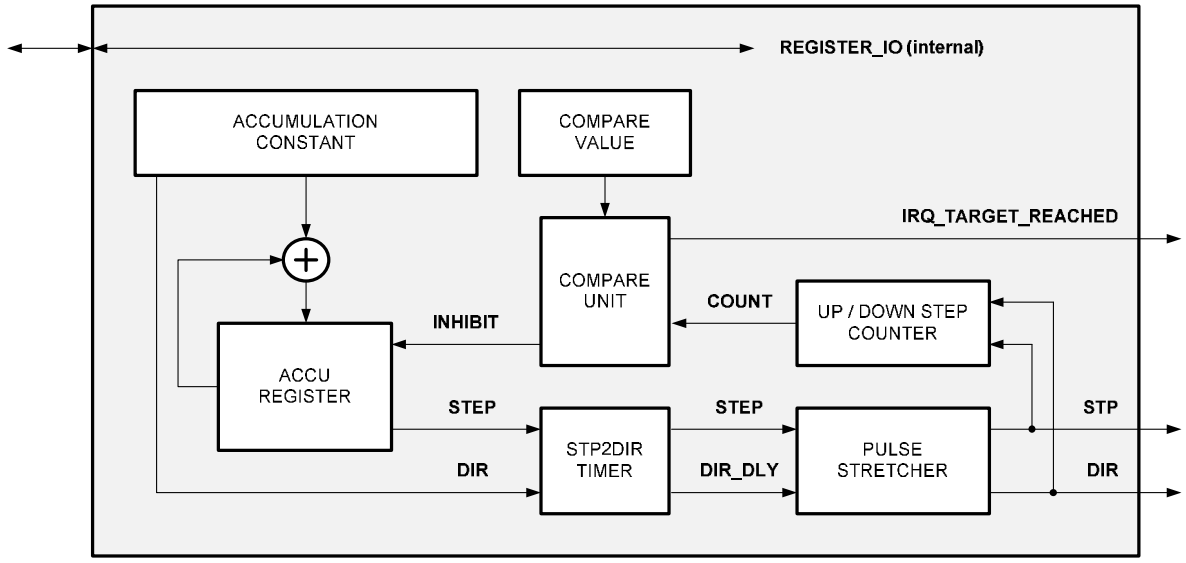


Figure 33 - Step Direction Unit Block Diagram

2.1.1 Step Direction Unit Timing

Write to the accumulation constant register starts step pulse generation. The first step pulse occurs after a time tSTEP1st. Following step pulses come after each tSTEP. The pulse length of the step pulses is iSTEP_PULSE. On change of direction by writing the accumulation constant with a constant of different sign, the first step pulse after write occurs after tSTP2DIR.

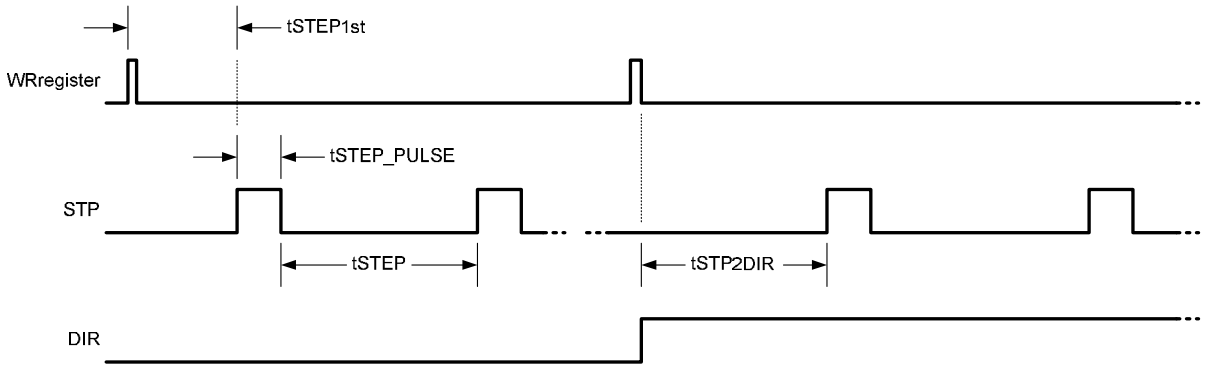


Figure 34 - Step-Direction Timing

2.1.2 Step Direction Unit Signals

Table 143 : MFCIO S/D unit signals

Signal	I/O	Function
STP	Out	Step Pulse Output
DIR	Out	Direction Signal Output
IRQ_TARGET_REACHED	Out	Interrupt Request Output indicating target reached condition

2.1.3 Step Direction Unit and Register Set

Table 144 : MFCIO S/D unit register set

MFCIO direct access register address #	Register Name	R/W	Size	Description
12	SD_CH_SR	W	31 … 0	Signed accumulation constant, this accumulation constant determines the time tSTEP between two successive steps rate resp. the step frequency, the Sign (MSB) of this accumulation constant is used for the direction signal output (DIR_P, DIR_N); the accumulation constant c is 2th complement (section 6.8.1)
13	SD_CH_SC	R	31 … 0	step counter, counting up/down depending on step direction, the direction is programmed via sign (2th complement) of SD_CH_SRs (section 6.8.2)
14	SD_CH_ST	W	31 … 0	steps to be done (section 0)
15	SD_CH_SL	W	15 … 0	Step pulse length in terms of CLK25MHz cycles, with tSTP = … (section 6.8.4)
16	SD_CH_DLY	W	15 … 0	Delay between change of direction and first step pulse, delay is … (section 6.8.5)
17	SD_CFG	W	7 … 0	7: - 6: - 5: 0 : stp pulse / 1 toggle 4: 0 - / 1 clears step count 3: dir signal polarity 2: step pulse polarity 1: 0=n pulses / 1=continuous 0: 0=disable / 1=enable

CH = channel (in case of more than one step direction unit)

Table 145 : MFCIO S/D Unit real world unit relations

Parameter	Value	Description / Function	Comment
f_{CLK} [Hz]	25 MHz	clock frequency of step direction unit	clock frequency of the step direction unit
t_{CLK} [s]	40 ns	clock period length	$t_{CLK}[s] = 1 / f_{CLK}[Hz]$
f_{STEP} [Hz]		$f_{STEP}[Hz] = (f_{CLK}[Hz]/2^{32})*(SD\_CH\#\_SR)$	step frequency, programmed via step rate accumulation constant SD_CH#_SR
Max. f_{STEP} [Hz]	12.5 MHz		Theoretical maximum value for fStep. Usable step frequency depends on step pulse length configuration.
t_{STEP} [s]		$t_{STEP} = 1 / f_{STEP}[Hz]$	time between steps
t_{STEP_PULSE} [s]		$t_{STEP\_PULSE}[s] = (STP\_LEN\_I+1) / f_{CLK}[Hz]$	step pulse length must be lower than time between step pulses! $t_{STEP\_PULSE}[s] < t_{STEP}[s]$
DIR		DIR = 0 ⇔ positive direction, DIR = 1 ⇔ negative direction, direction is depending on sign of step rate register SD_CH#_SR where the step rate register is 2th complement	direction signal, depending of step rate (SR) parameter, DIR = 0 if SR > 0 or SR = 0, DIR = 1 if SR < 0
$t_{STEP1st}$ [s]		time to 1st step pulse since WR=0 with $t_{STEP1st}[s] = 2^{32} / STP\_RT\_I * t_{CLK}[s] + (STP\_DLY\_I + 1) * t_{CLK}[s] + (2 * t_{CLK}[s]);$	Time between write until the first step pulse occurs
$t_{STEP1stWR}$ [s]		time to 1st step pulse since WR=0 step delay plus 1 internal clock plus 2 clock cycles to pulse length	Internal processing adds an delay
SD_CH_SR		Writing SD_CH_SR = 0 clears the step accu register step_accu_ff and stops step pulse generation.	

6.8.1 Step Direction Accumulation Constant

The step direction accumulation constant determines the time t_{STEP} between two successive step pulses – the step rate (SR). Each internal PWM clock accumulates an accumulator according to a = a + c with the accumulator constant c. Toggle of the MSB of the accumulator register a triggers a step pulse. With this principle, the step frequency is smarter adjustable compared to a simple frequency divider. Writing c = 0 clears the accumulator and stops the step pulse generation. The step pulse frequency $f_{STEP}[Hz] = (f_{CLK}[Hz] / 2^{32}) * c$. To calculate the accumulation c one just have to calculate

$$c = int(2^{32} * (f_{CLK}[Hz] / f_{STEP}[Hz])) = int (4294967296 * (f_{CLK}[Hz] / f_{STEP}[Hz])$$

TIME TO FIRST STEP

Due to internal processing, the time to the first step since write register write is

$$t_{STEP1st}[s] = 2^{32} / SD\_SR * t_{CLK}[s] + (STP\_SDY\_I + 1) * t_{CLK}[s] + (2 * t_{CLK}[s])$$

6.8.2 Step Counter

The step counter counts the number of steps, taking the direction into account. This is a read only register. For initialization to zero a configuration bit within the step direction configuration register hast to be written.

6.8.3 Step Target

The step target defines the number of steps to be made for the step mode until stop. This register can be overwritten at any time. When the number of steps has been made, the unit stops outputting S/D pulses. When read, it gives the remaining numbers that must still be made.

6.8.4 Step Length

The duration of the step pulse – the step length (SL) - signal is programmable for adaption to external power stages.

MAXIMUM STEP LENGTH

The step pulse length t_{STEP_PULSE}[s] must be lower than the time t_{STEP}[s] between step pulses to have step pulses. The condition t_{STEP_PULSE}[s] < t_{STEP}[s] must be ensured by the application.

6.8.5 Step-to-Direction Delay

The delay between the first step pulse after a change of the direction is programmable for adaption to external power stages to take external delay paths into account.

6.8.6 Step Direction Unit Configuration

The step direction configuration defines the mode of operation (continuous or finite number of step pulses), polarity of step pulse signal and direction signal. One bit is for zeroing of step pulse counter. On bit is for enabling and disabling of the step pulse unit.

6.8.7 Interrupt Output Signal

A signal IRQ_TAR_REACHED of a single clock pulse length indicated indicates that a TARGET position is reached in terms of step counts.

6.9 MFCIO PWM Unit

The MFC is equipped with a tripe pulse width modulation (PWM) block including programmable brake before make (BBM) unit. Both high side and low side control signals are available as separate outputs. A single PWM counter generates three synchronous PWM signals. Setting the maximum count as limit defines the PWM frequency. Left aligned PWM, Centered PWM, and right aligned PWM is selectable. The BBM take care concerning the BBM timing. The BBM timing is individually programmable for high side and low side. Fixed pulses are available for triggering of ADCs or triggering interrupts of a CPU. Programmable trigger output signals are available. The position and the duration of two theses pulses is programmable. The pulse PULSE_ZERO indicating the beginning of a PWM cycle and the pulse PULSE_CENTER indicating the center of the PWM are fixed. The two programmable signals PULSE_A and PULSE_B are for advanced ADC triggering. The trigger signal output PULSE_AB is the logical or of PULSE_A and PULSE_B. The polarities of the output signals of the PWM unit is programmable. An input signal ES with programmable active polarity disables the PWM signals.

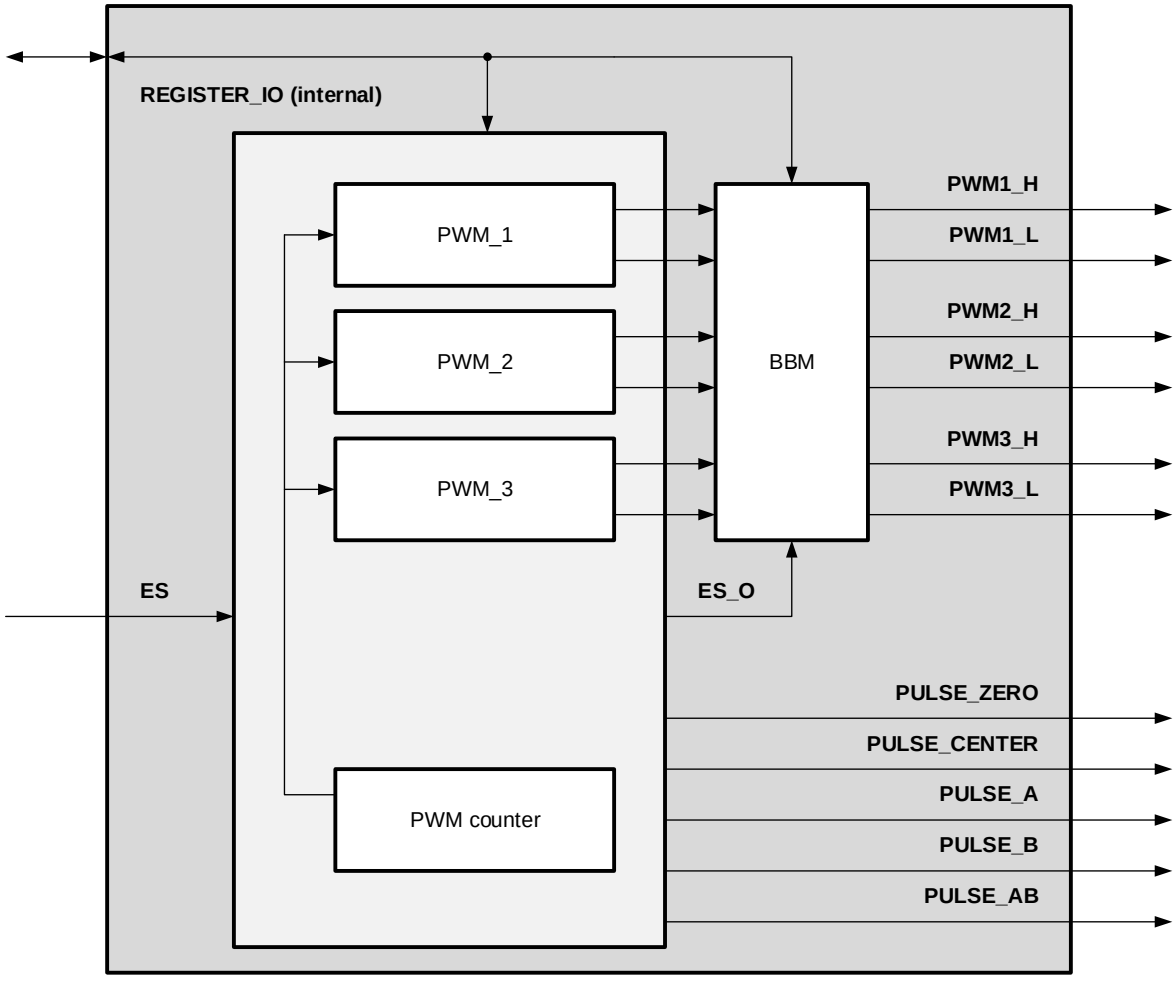


Figure 35 - PWM Block Diagram

2.1.4 PWM Unit Signals

Table 146 : MFCIO PWM unit signals

Signal	I/O	Function
ES	In	Emergency Switch Input
PULSE_ZERO	Out	Pulse indicating the beginning of a new PWM cycle
PULSE_A	Out	Pulse with programmable positon within the PWM period
PULSE_CENTER	Out	Pulse indicating the middle of the PWM cycle
PULSE_B	Out	Pulse with programmable positon within the PWM period
PULSE_AB	Out	Logical or of PULSE_A and PULSE_B
PWM0_H	Out	High side control output of PWM unit 0
PWM0_L	Out	Low side control output of PWM unit 0
PWM1_H	Out	High side control output of PWM unit 1
PWM1_L	Out	Low side control output of PWM unit 1
PWM2_H	Out	High side control output of PWM unit 2
PWM2_L	Out	Low side control output of PWM unit 2

2.1.5 PWM Unit and Register Set

Table 147 : MFCIO PWM unit register set

MFCIO direct access register address #	Register Name	R/W	Size [bit]	Description
18	PWM_MAXCNT	W	16	11…0: This parameter determines the duration of the PWM period resp. the PWM frequency
19	PWM_CHOPMODE	W	16	This parameter defines the chopper mode (no chopper, high side chopper, low side chopper, complementary chopper) for the three PWM units (section 7.10.2), each PWM unit can be configured separately. 10:8 6:4 2:0
20	PWM_ALIGNMENT	W	8	1..0: This parameter defines the alignment of the PWM: 00 = off, 10 = left, 01 = right, 11 = centered (6.9.3 7:2 : unused
21	PWM_POLARITIES	W	8	0: pwm_low_sides 1: pwm_high_sides 2: pulse_ab 3: pulse_b 4: pulse_centerf 5: pulse_a 6: pulse_zero 7: unused
22	PWM_VALUE_0	W	16	11:0 : These three registers define the PWM duty cycle (on time) for each PWM output
23	PWM_VALUE_1	W	16	
24	PWM_VALUE_2	W	16	

25	PWM0_CNTRSHFT	W	16	11..0 : These three registers define the shift time for the PWM units, it is intended to create a time gap between PWM edges for phases current measurements (section 6.9.9)
26	PWM1_CNTRSHFT	W	16	
27	PWM2_CNTRSHFT	W	16	
28	PWM_ PULSE_A	W	16	11..0 : This is a programmable ADC trigger pulse A (section 6.9.6)
29	PWM_PULSE_B	W	16	11..0 : This is a programmable ADC trigger pulse B (section 6.9.7)
30	PWM_PULSE_LENGTH	W	8	7:0 : This register represents the programmable pulse length of the trigger pulses A and B and pwm_start and pwm_center (section 6.9.8)
31	PWM_BBM_H	W	8	7:0 : This parameter is the brake before make time in terms of clock cycles for the high side MOS-FET control (section 6.9.11
32	PWM_BBM_L	W	8	7:0 : This parameter is the brake before make time in terms of clock cycles for the high side MOS-FET controls (section 6.9.12)

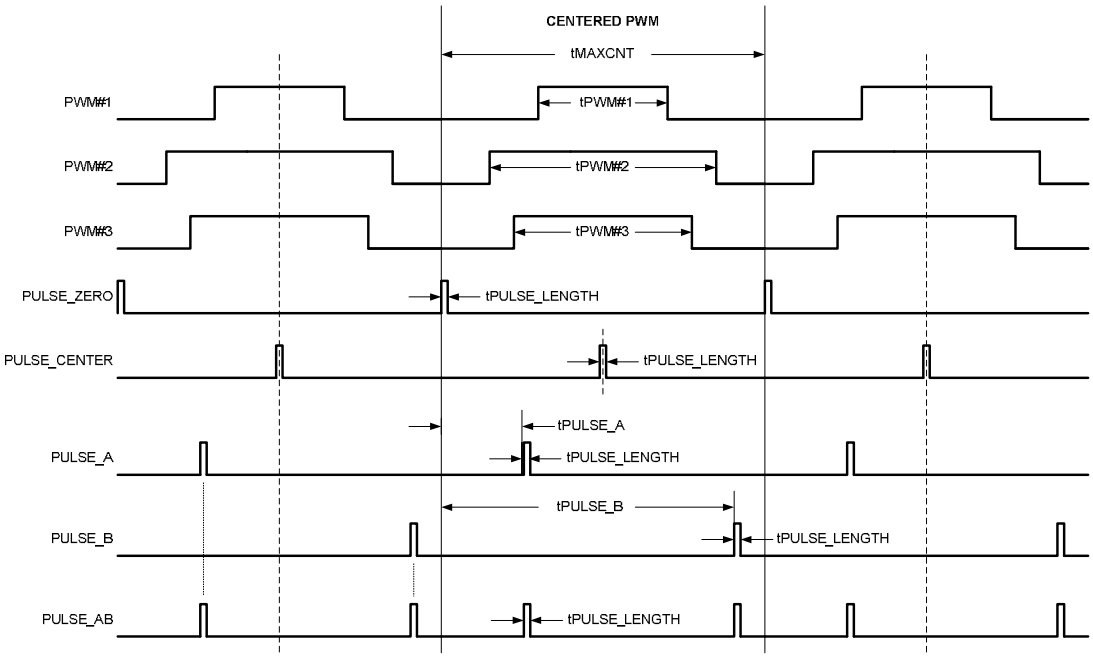


Figure 36 - PWM Timing (centered PWM)

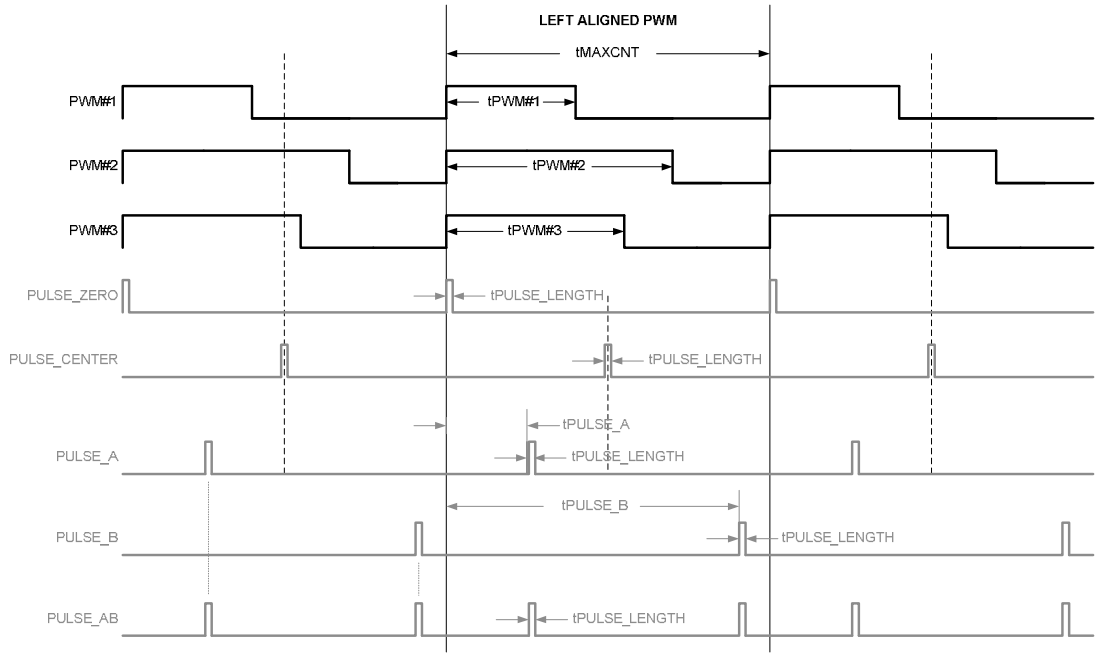


Figure 37 - PWM Timing (left aligned PWM)

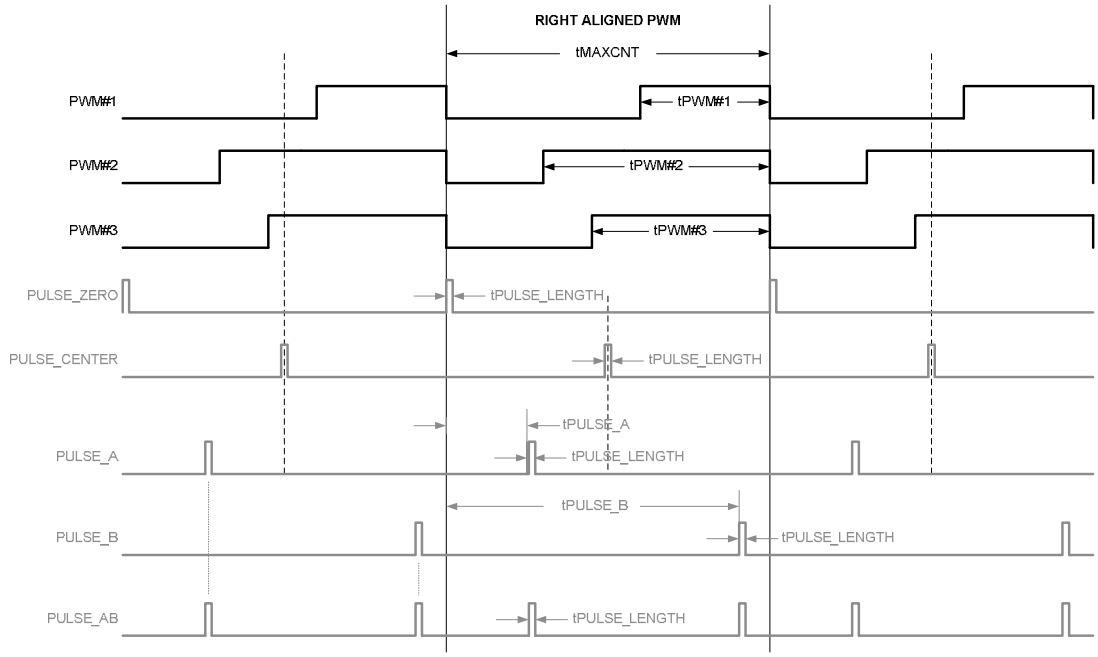


Figure 38 - PWM Timing (right aligned PWM)

Table 148 : MFCIO PWM block real world unit relations

Parameter	Value	Description / Function	Comment
f_{CLK}[Hz]	100 MHz	clock frequency of PWM unit	f_{CLK}[Hz] = 1 / t_{CLK}[s]
t_{CLK}[s]	10 ns	clock period length	t_{CLK}[s] = 1 / f_{CLK}[Hz]
max. t_{PWM}[s]	40,96us	Length of PWM period t_{PWM} = t_{CLK} * (1 + PWM_MAXCNT)	Maximum t_{PWM} with maximum PWM resolution of 12 bit.
min. f_{PWM}[Hz]	24.414 kHz	PWM frequency = 1 / t_{PWM}	Minimal PWM frequency with maximum PWM resolution of 12 bit.
t_{PULSE_LENGTH}		Length of the ADC trigger pulses with t_{PULSE_LENGTH} = PULSE_LENGTH * t_{CLK}	pulse length is adjustable to trigger external ADCs t_{CLK} = 10ns
t_{BBM}		Brake Before Make time tBBM with t_{BBM_H} = BBM_H * t_{CLK} t_{BBM_L} = BBM_L * t_{CLK}	Individually programmable for high side and low side due to different timing requirements, especially when using PMOS @ High Side and NMOS @ Low Side

6.9.1 PWM_MAXCNT Configuration Register

This register commonly defines the number of counts per PWM cycle for three PWM units. This determines the length tPWM of each PWM cycle resp. the PWM frequency fPWM. It is programmable for adjustment of the PWM frequency fPWM.

6.9.2 PWM_CHOPMODE Configuration Register

This Register commonly selects the chopper mode of the three PWM units. The following table gives the available chopper modes.

PWM channel 0 chopper mode = PWM_CHOPMODE(2:0)
PWM channel 1 chopper mode = PWM_CHOPMODE(6:4)
PWM channel 2 chopper mode = PWM_CHOPMODE(10:8)

Table 149 : PWM chopper mode selection

Selection	Chopper	High Side (HS)	Low Side (LS)	Function
%000	no	off	off	no chopper, all off
%001	no	off	on	no chopper, LS permanent on
%010	no	no	off	no chopper, HS permanent on
%011	no	off	off	no chopper, all off, not used
%100	no	off	off	no chopper, all off, not used
%101	yes	off	PWM	chopper LS, HS off
%110	yes	PWM	Off	chopper HS, LS off
%111	Yes	PWM	not PWM	chopper HS and LS complementary, brake-before-make is handled by programmable BBB unit

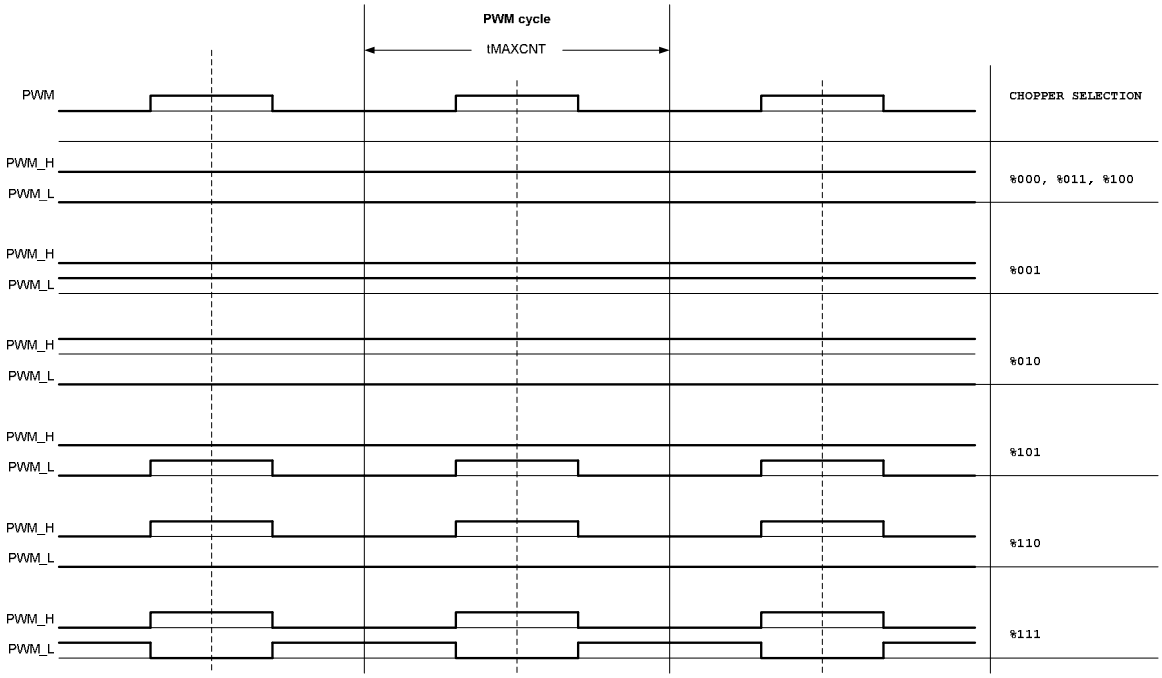


Figure 39 - Chopper Modes (OFF, Low Side ON, High Side ON, Low Side Chopper, High Side Chopper, Complementary Low Side and high Side Chopper)

6.9.3 PWM_ALIGNMENT Configuration Register

This register commonly determines the alignment of the three PWM units. The alignment can be programmed left aligned, centered, or right aligned (pls. refer Figure 36, Figure 37, and Figure 38).

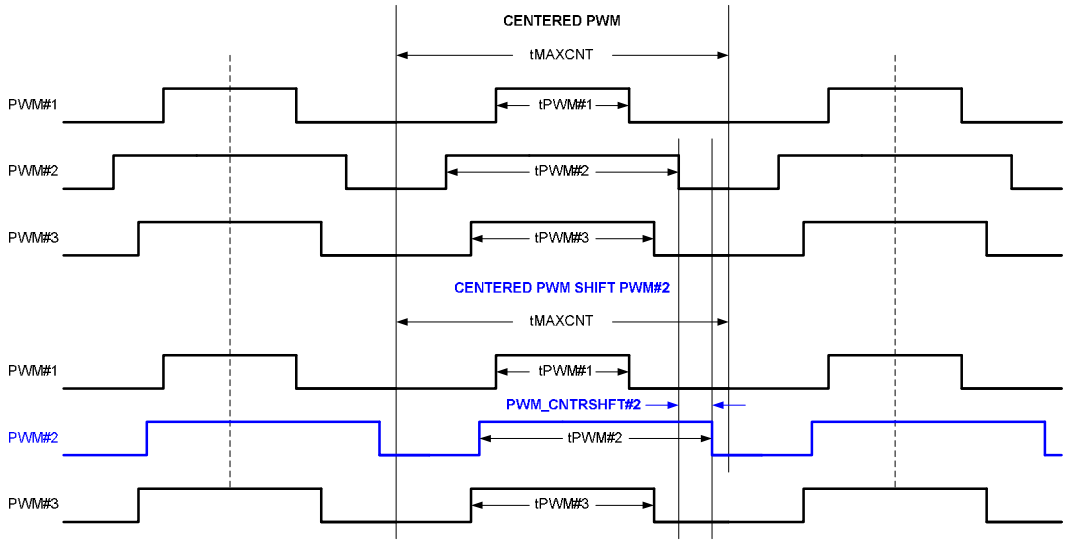


Figure 40 - Centered PWM with PWM#2 shifted from Center (Example)

6.9.4 POLARITIES Configuration Register

The PWM signals of the three are of positive logic. So, logical one level means ON and logical zero level means OFF. Depending on the MOS-FET drivers, switching on a MOS-FET might require an inverted logical level. The polarities configuration register determines the switching polarities for the high side MOS-FETs and switching polarities for the low side MOS-FETs.

6.9.5 PWM Value Registers

Together with the programmed PWM counter length, the PWM values determine the PWM duty cycle. The PWM duty cycle is individually programmable for each of the three PWM units. Programming the three PWM values sets up a vector of effective three voltages.

6.9.6 PULSE_A Configuration Register

The position of the trigger pulse A is programmable within the PWM cycle. A second trigger pulse within the PWM cycle is programmable individually. This pulse is intended to trigger an ADC.

6.9.7 PULSE_B Configuration Register

The position of the trigger pulse B is programmable within the PWM cycle. A second trigger pulse within the PWM cycle is programmable individually. This pulse is intended to trigger an ADC.

6.9.8 PULSE_LENGTH Configuration Register

To take the timing of different ADCs into account, the length of PULSE_A and PULSE_B and the fixed trigger pulses named PULSE_CENTER and PULSE_ZERO, is commonly programmable in terms of clock cycles by this register.

6.9.9 Asymmetric PWM Configuration Registers

To open a wider time window between PWM switching events that are close to each other PWM an asymmetric PWM shift can be programmed individually for each PWM unit. This leaves the PWM duty cycles unchanged. It is useful for current measurement with sense resistors at the bottom of the MOS-FET half bridges. In contrast to current sensing with differential sense amplifiers or with hall sensor bases current sensors within the current phase, this feature has no advantage.

6.9.10 Brake-Before-Make (BBM)

To avoid cross conduction, of the half bridges, the brake before make (BBM) timing is programmable. In most cases, the same BBM time is sufficient for both, low side and high side. The BBM time should be programmed as short as possible and as long as necessary. A too long BBM time causes conduction of the bulk diodes of the power MOS-FETs and that causes higher power dissipation and less motor performance concerning current regulation. In case of using PMOS-FETs for high and NMOS-FETs for low side with asymmetric switching characteristics, it might be advantageous to program different BBM_H time and BBM_L time.

6.9.11 BBM_H Configuration Register

This register programs the time from switch off the low side to switch on the high side in terms of clock cycles. The BBM_H is common for all three high side power MOS-FETs.

6.9.12 BBM_L Configuration Registers

This register programs the time from switch off the high side to switch on the low side in terms of clock cycles. The BBM_L is common for all three high side power MOS-FETs.

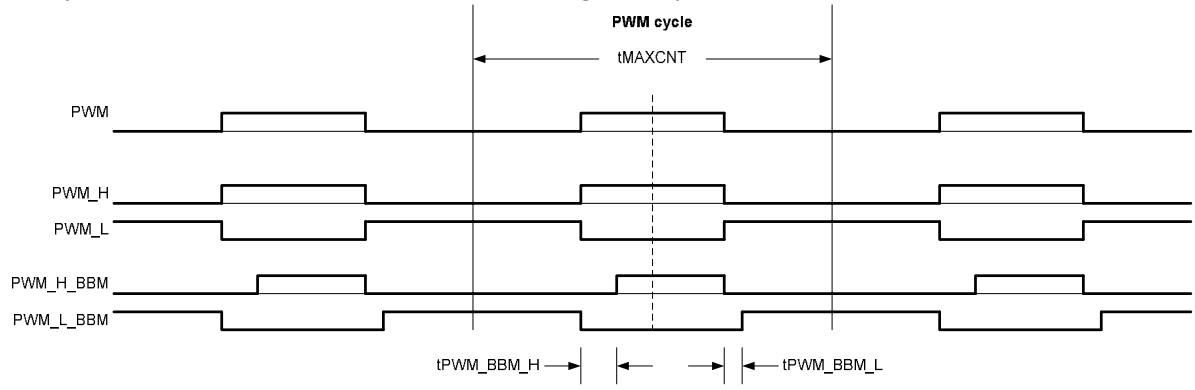


Figure 41 - Brake Before Make (BBM) Timing (individual programmable for Low Side and High Side)

6.9.13 Emergency Switch Input Off-State

The emergency switch input MFC_nES of the TMC8460-BI can be configured to disable the MOSFET power stage immediately, e.g., under control of an external over current comparator (or similar trigger).
nES is thereby used as external trigger. It is low active and contains an internal pull down resistor. For normal operation it must be actively pulled high externally.

When pulled to ground, MFC_nES terminates the actual PWM cycle and brings the low side and high side gate outputs into off-state. The off-state is defined by the PWM_POLARITIES register using bits 0 (low sides) and bit 1 (high sides).

PWM_POLARITIES(0) = 0 → 0 = off / 1 = on
PWM_POLARITIES(0) = 1 → 1 = off / 0 = on

6.10 MFCIO General Purpose IO Unit

The MFC GPIO Unit handles the general-purpose inputs and outputs.
Eight (8) independent GPIO ports are available.
Each GPIO is configurable either as input or as output.

6.10.1 GPIO Registers

Three registers are required for MFC GPIO configuration:
GPO_OUT_VAL defines the output values of the outputs.
GPI_IN_VAL is read only and contains the actual input values.
GPIO_CONFIG allows to configure the GPIO direction.

Table 150 : MFCIO GPIO unit registers

MFCIO direct access register address #	Register Name	R/W	Size [bit]	Description
33	GPO_OUT_VAL	W	16	7:0 : GPO output values 15:8 : GPO safe state (when MFC_nES = '0')
34	GPI_IN_VAL	R	8	7:0 : GPI input values
35	GPIO_CONFIG	W	8	7:0 : GPIO direction (0 = input, 1 = output)

6.10.2 General purpose Inputs (GPI)

If configured as input, each GPI value can be read from the GPI_IN_VAL from the respective bit position.

6.10.3 General purpose Outputs (GPO)

If configured as output, each GPO takes the value defined in GPO_OUT_VAL[7:0].

For output data to be visible on the output signals, the following conditions have to be met:

- ESC must be in OP state
- Your output data must be written to the register
- MFC_nES must be driven high externally

6.10.4 Emergency Switch Input State

The external emergency switch input MFC_nES (if masked/enabled) brings all GPIOs currently configured as outputs into a configurable safe state.
The GPO_OUT_VAL register high byte (bits 15:8) defines the safe state.

6.11 MFCIO Watchdog Unit

6.11.1 General Function

The watchdog timer allows monitoring of external signals, or monitoring of ETHERCAT activity. A certain condition can be chosen for retriggering the watchdog, i.e. a certain input signal constellation. In case this constellation does not occur at least once within a pre-programmable time period, the watchdog timer will expire and will trigger a certain watchdog action. To avoid static reset of the watchdog, the watchdog input condition is edge sensitive, i.e. it becomes reset when the condition goes active respectively goes inactive. Once the watchdog expires, the watchdog safety circuitry becomes active. This action can bring I/O lines into a certain state, in order to allow the system to return to a known, safe condition. Therefore, all I/O lines are directly mapped to the GPIO ports of the chip, so that they perform independently of the actually configured peripheral configuration. The watchdog action can be chosen to remain active continuously, until it becomes reset by a watchdog re-configuration, or it can be programmed to return to normal operation state, once the selected condition becomes true again.

In an optional use case, the watchdog timer can be used to measure the maximum delay in between of the occurrence of certain input conditions, in between of SPI frames, etc.

6.11.2 Watchdog Register Set

Once initialized, the watchdog timer monitors the application for activity and allows setting of pre-programmed I/O patterns, in case the time limit is expired without activity. In order to allow tuning of this time limit, the maximum time between two trigger events becomes measured. This function also allows delay time measurement for input channels (i.e. when no watchdog action is chosen). The watchdog timeout counter starts from zero up to WD_TIME. When it reaches WD_TIME, it triggers the watchdog action.

The selected watchdog event resets the timeout counter. As trigger sources, the internal ETHERCAT start of frame, the two SPI chip select signals as well as any combination of I/O lines can be used. For the I/O lines, the polarity and edge are programmable. When using a GPIO programmed to output as watchdog trigger, the watchdog circuitry will monitor the real output by checking the polarity of the output signal. This way, also a short circuit condition will be detected. The chip select signals respond to a rising edge (i.e. when the SPI interface loads the SPI shift register data into the corresponding registers).

Table 151 : MFCIO watchdog unit registers

MFCIO direct access register address #	Register Name	R/W	Size [bit]	Description	
38	WD_TIME	W/R	32	Watchdog time 32 bit, unsigned 0 = Watchdog off *Time* = Number of 25MHz clocks	
39	WD_CFG	W/R	8	Bit 0	*cfg_persistent* 0 The watchdog action ends when the next trigger event occurs 1 A timeout situation can only be cleared by rewriting WD_TIME
				Bit 1	*cfg_pdi_csn_enable* 1 Retrigger by positive edge on PDI_SPI_CSN
				Bit 2	*cfg_mfc_csn_enable* 1 Retrigger by positive edge on MFC_CTRL_SPI_CSN

				Bit 3	*cfg_sof_enable* 1 Retrigger by ETHERCAT start of frame
				Bit 4	*cfg_in_edge* 0 Retrigger by input condition becoming false 1 Retrigger by input condition becoming true
		R		Bit 7	*cfg_wd_active* 1 Signals an active watchdog timeout
40	WD_OUT_ MASK_POL	W/R	64	Mask for outputs to be affected by watchdog action 31:0 : WD_OUT_POL, polarity for outputs affected by watchdog action 32 bit, each bit corresponds to one output line The polarity describes the output level desired upon watchdog action 63:32: WD_OUT_MASK, each bit corresponds to one output line 0 Output is not affected 1 Output [i] becomes set to *WD_OUT_POL*[i] upon watchdog action See Table xxx below for the detailed signal mapping / indices of WD_OUT_MASK_POL	
41	WD_OE_ POL	W/R	32	I/O Output enable level for outputs affected by watchdog action 32 bit, each bit corresponds to one output line The polarity describes the OE setting desired upon watchdog action (1= output, 0= input)	
42	WD_IN_ MASK_POL	W/R	32	Selection of input signals for watchdog retriggering 15:0 : WD_IN_POL, Input signal levels for watchdog retriggering 16 bit, each bit corresponds to one input line The polarity describes the input level for signals selected by *WD_IN_MASK* required to re-trigger the watchdog timer 31:16 : WD_IN_MASK, each bit corresponds to one input line 0 Input is not selected 1 Input I/O[i] must reach polarity *WD_IN_POL*[i] to re-trigger the watchdog timer See Table xxx below for the detailed signal mapping / indices of WD_IN_MASK_POL	
43	WD_MAX	R	32	Peak value reached by watchdog timeout counter Reset to 0 by writing to *WD_TIME*	

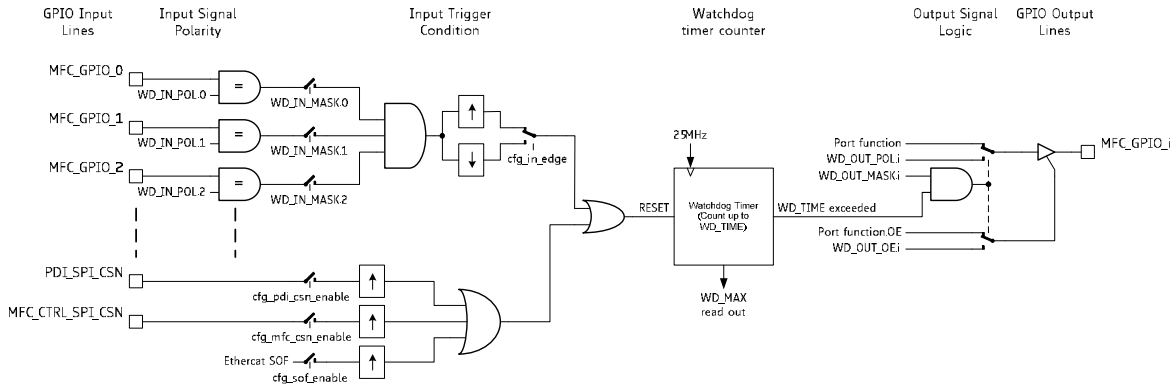


Figure 6.42 - Structure of the watchdog unit

The following table contains the assignments of ports/signals to the configuration bits in the *WD_OUT_MASK_POL* register.

Table 152 : Watchdog unit output port configuration assignment

Bit #	Signal	Description
0	GPO0	WD_OUT_POL
1	GPO1	
2	GPO2	
3	GPO3	
4	GPO4	
5	GPO5	
6	GPO6	
7	GPO7	
8	PWM_LS0	
9	PWM_LS1	
10	PWM_LS2	
11	PWM_HS0	
12	PWM_HS1	
13	PWM_HS2	
14	PWM_PULSE_AB	
15	PWM_PULSE_B	
16	PWM_PULSE_CENTER	
17	PWM_PULSE_A	
18	PWM_PULSE_START	
19	SD_DIR	
20	SD_STP	
21	SPI_CS3	
22	SPI_CS2	
23	SPI_CS1	
24	SPI_CS0	
25	SPI_SDI	
26	SPI_SCK	
27	reserved	
28	reserved	
29	reserved	
30	reserved	
31	reserved	

Bit #	Signal	Description
32	GPO0	WD_OUT_MASK
33	GPO1	
34	GPO2	
35	GPO3	
36	GPO4	
37	GPO5	
38	GPO6	
39	GPO7	
40	PWM_LS0	
41	PWM_LS1	
42	PWM_LS2	
43	PWM_HS0	
44	PWM_HS1	
45	PWM_HS2	
46	PWM_PULSE_AB	
47	PWM_PULSE_B	
48	PWM_PULSE_CENTER	
49	PWM_PULSE_A	
50	PWM_PULSE_START	
51	SD_DIR	
52	SD_STP	
53	SPI_CS3	
54	SPI_CS2	
55	SPI_CS1	
56	SPI_CS0	
57	SPI_SDI	
58	SPI_SCK	
59	reserved	
60	reserved	
61	reserved	
62	reserved	
63	reserved	

The following table contains the assignments of ports/signals to the configuration bits in the *WD_IN_MASK_POL* register.

Table 153 : Watchdog unit input port configuration assignment

Bit #	Signal	Description
0	GPI0	WD_IN_POL
1	GPI1	
2	GPI2	
3	GPI3	
4	GPI4	
5	GPI5	
6	GPI6	
7	GPI7	
8	SPI_SDO	
9	ABN_N	
10	ABN_B	
11	ABN_A	
12	reserved	
13	reserved	
14	reserved	
15	reserved	

Bit #	Signal	Description
16	GPI0	WD_IN_MASK
17	GPI1	
18	GPI2	
19	GPI3	
20	GPI4	
21	GPI5	
22	GPI6	
23	GPI7	
24	SPI_SDO	
25	ABN_N	
26	ABN_B	
27	ABN_A	
28	reserved	
29	reserved	
30	reserved	
31	reserved	

6.12 MFCIO IRQ Unit and Register Set

The MFCIO block of the TMC8460 provides a dedicated IRQ output signal to a local microcontroller to indicate certain events and conditions.
The pin MFCIO_IRQ can be configured and read out using the two registers shown in the next table.

Table 154 : MFCIO IRQ unit registers

MFCIO direct access register address #	Register Name	R/W	Size [bit]	Description
36	IRQ_CFG	W	16	Masking / Enabling of the required IRQ sources which are or-ed to set the MFCIO_IRQ output signal Bit 0 : Encoder unit N-event Bit 1 : Step/Direction unit target reached signal Bit 2 : SPI unit new data available Bit 3 : Watchdog unit timeout Bit 4 : PWM Unit PWM cycle start trigger Bit 5 : PWM Unit PWM cycle center trigger Bit 6 : PWM Unit configurable pulse A Bit 7 : PWM Unit configurable pulse B Bits 14..8 : reserved, unused Bit 15 : activation of external low active emergency switch signal MFC_nES
37	IRQ_FLAGS	R	16	Status register to read the IRQ flags of the masked IRQ sources

2.1.6 IRQ_CFG Register

The IRQ mask register allows to enable/disable certain IRQ trigger events of the MFCIO block.
This is useful when MFCIO sub blocks are used at the same time or are not needed.

A special option is to enable the MFC_nES emergency switch input signal. Setting the mask bit here in the IRQ_CFG register has two effects:

a) It functionally enables this pin. Now all functional outputs of the TMC8460 are affected if MFC_nES goes low.
b) It masks the MFC_nES event (falling edge) as IRQ source to the MFCIO_IRQ output.

In case of an event on the MFC_nES input, the respective flag in the IRQ_FLAGS[15] register can only be unset be either doing a device reset or by actively writing 2 times into the IRQ_CFG register at bit position 15. Thereby, the existing IRQ mask at bit 15 must first be set to zero and then set back to 1 again. This way, the internal emergency flag is unset.

2.1.7 IRQ_FLAGS Register

This register can be read out after the IRQ was set to identify the IRQ source, especially when more than one IRQ source was enabled.
Each flags corresponds to the same IRQ source bit position as in IRQ_CFG register.

Reading this registers clears all individual IRQ source bits except the MFC_nES flag bit at position 15.

6.13 MFCIO Emergency Switch Input

The TMC8460-BI comes with an additional emergency switch input pin. This input pin connects to the MFCIO block. Its function is to indicate an error condition and to switch certain outputs of the MFCIO block into configurable safe states.

The emergency switch input is called MFC_nES. It is low active.
MFC_nES input has impact on the following functional units and outputs:

- PWM high side and low side gate outputs are set to a defined configurable safe off state.
- All GPIO ports that are configured as outputs are set to a defined configurable safe off state.
- SD step output and internal step counter freeze
- The MFCIO_IRQ signal will be triggered

6.13.1 Activation & Usage

To activate the emergency switch input masked bit 15 in the IRQ_CFG register.
Otherwise, MFC_nES is ignored.
A microcontroller or another circuit must actively drive high VCCIO level (3.3V) at the MFC_nES input for normal operation.

IRQ_CFG[15] = 0 → MFC_nES not active, input level is ignored
IRQ_CFG[15] = 1 → MFC_nES is active, must actively be driven high for normal operation

MFC_nES is triggered by setting it to low level (0V, GND) resulting in the following actions:

- IRQ_FLAGS[15] is set
- External signal/pin MFC_IRQ is set / driven high
- All output signals of the MFCIO block take their configured safe values, the S/D unit is freezed

6.13.2 Re-Activation

The internal emergency switch flag IRQ_FLAGS[158] remains set even when the pin MFC_nES is already driven high again.
The emergency flag can only be unset be either doing a complete chip reset, by power cycling, or by actively writing 2 times into the IRQ_CFG register at bit position 15:

First write operation: IRQ_CFG[15] = 0
Second write operation: IRQ_CFG[15] = 1

This way, the internal emergency flag is unset.
This can be done by the ECAT master or by the local application controller.

6.14 Auxiliary Clock Output

The TMC8460 provides a dedicated clock output to supply a 16MHz clock source to peripheral components. The 16MHz clock is internally derived from the PLL logic which itself uses the 25MHz clock input at pin CLK_25MHz as base clock.
If not required the clock can be switched off using an enable pin.

Table 155 : TMC8460 auxiliary clock pins

PKG. Pin	Pin Name / Function	DIR	Functional Description / Comments
U12	CLK_16MHz_OUT	O	16MHz auxiliary clock output, enabled by EN_16MHz_OUT
R12	EN_16MHz_OUT	I	Enable signal for 16 MHz auxiliary clock output: 0 = off 1 = on, 16 MHz available at CLK_16MHz_OUT

6.15 AL-State Override Configuration

All functional output signals of the MFCIO block are linked to the OP state of the EtherCAT state machine. As long as the actual state is not OP, all functional output ports are not driven but in high impedance state.

The AL_STATE_OVERRIDE register allows overriding this behavior.
The AL_STATE_OVERRIDE register can only be accessed from MFC CTRL SPI interface.
It can be accessed neither from ECAT interface nor from PDI SPI interface.

While the main purpose for AL_STATE_OVERRIDE is for testing it can also be used in the applications when the application controller has access to the MFC CTRL SPI interface. Nevertheless, special care must be taken since it overrides the original behavior of the outputs with respect to the EtherCAT state machine.

Each bit in the AL_STATE_OVERRIDE register controls override configuration of a specific MFCIO sub-block regarding the output ports availability.
If a bit is set to '1', the AL status register (0x0130:0x0131) is ignored and the output ports of this MFCIO sub-block are fully available using the MFC CTRL SPI interface.
The incremental encoder block, the IRQ configuration, and the watchdog block are not affected by the AL_STATE_OVERRIDE register since they only have input ports.

Table 156 : AL-state override register

MFCIO direct access register address #	Register Name	R/W	Size [byte]	Description
44	AL_STATE_OVERRIDE	W (only from MFC CTRL SPI)	1	0 – SPI block 1 – S/D block 2 – PWM block 3 – GPIO block 7:4 - reserved

7 ESD Sensitive Device

The TMC8460 is an ESD sensitive CMOS device sensitive to electrostatic discharge. Take special care to use adequate grounding of personnel and machines in manual handling. After soldering the devices to the board, ESD requirements are more relaxed. Failure to do so can result in defect or decreased reliability.

8 Disclaimer

TRINAMIC Motion Control GmbH & Co. KG does not authorize or warrant any of its products for use in life support systems, without the specific written consent of TRINAMIC Motion Control GmbH & Co. KG. Life support systems are equipment intended to support or sustain life, and whose failure to perform, when properly used in accordance with instructions provided, can be reasonably expected to result in personal injury or death.

Information given in this data sheet is believed to be accurate and reliable. However no responsibility is assumed for the consequences of its use nor for any infringement of patents or other rights of third parties which may result from its use.

Specifications are subject to change without notice.

All trademarks used are property of their respective owners.

9 Revision History

Version	Date	Author SK= Stephan Kubisch	Description
V1.00	2016-Sep-01	SL, SK	First release version

Table 157: Documentation Revisions

