

EC-1 Series

R01AN3780EJ0100

Rev.1.00

SPI Connection Library for the EtherCAT® Slave Controller

Apr 14, 2017

Outline

This application note describes a library for access to the EtherCAT slave controller (“ESC”) of the EC-1 from an external MCU.

The features of this library are described below.

- The library enables access to the ESC from an external MCU by using the SPI in slave mode as a standard serial communications interface. This allows connection to various MCUs which support SPI master mode.
- By combining processing for initialization of the port and PHY module with the library, various boards on which the EC-1 is mounted can be turned into ESC boards which are accessible by an external MCU.
- The items required for porting the EtherCAT slave controller of the ET1100 from Beckhoff Automation GmbH to an EC-1 which incorporates the ESC SPI connection library are described as a sample application.

Target Devices

EC-1

Contents

| | |
|--|----|
| 1. Specifications | 4 |
| 2. Operating Environment..... | 5 |
| 3. Hardware..... | 6 |
| 3.1 Example of the Hardware Configuration | 6 |
| 3.2 Pins | 7 |
| 4. SPI Interface | 8 |
| 4.1 Example of the Software Configuration | 9 |
| 4.2 SPI Address Space | 10 |
| 4.3 Extended SPI Registers | 10 |
| 4.3.1 Device ID Register (DEVICEID)..... | 11 |
| 4.3.2 Revision Number Register (REVNO)..... | 11 |
| 4.3.3 Status Register (STATUS)..... | 11 |
| 4.3.4 Control Register (CONTROL)..... | 11 |
| 4.3.5 Scratch Registers (SCRATCH0 to 9)..... | 11 |
| 4.3.6 Sequential Data Number Register (SQDATNUM) | 11 |
| 4.3.7 Sequential Data Registers (SQDATA0 to 63)..... | 11 |
| 4.4 Format of SPI Packets | 12 |
| 4.4.1 Commands..... | 12 |
| 4.4.2 RD Command | 12 |
| 4.4.3 WR Command..... | 13 |
| 4.4.4 SQRD Command | 14 |
| 4.4.5 SQWR Command | 15 |
| 4.4.6 DTRD Command..... | 16 |
| 4.4.7 DTWR Command..... | 17 |
| 5. Software..... | 19 |
| 5.1 Operation in Outline | 19 |
| 5.2 Procedure for Designing Software | 20 |
| 5.3 Interrupts | 21 |
| 5.4 Fixed-Width Integers | 21 |
| 5.5 Constants and Error Codes..... | 21 |
| 5.6 Structures, Unions, and Enumerations | 23 |
| 5.7 Global Variables | 25 |
| 5.8 Functions..... | 25 |
| 5.9 Functions..... | 26 |
| 5.9.1 Processing by Initial Settings for the SPI Pins..... | 26 |
| 5.9.2 Initialization Processing for SPI Control..... | 26 |
| 5.9.3 Starting SPI Communications..... | 28 |
| 5.9.4 Stopping SPI Communications | 29 |
| 5.9.5 Processing for SPI Transfer..... | 30 |

| | | |
|-------|---|----|
| 6. | Sample Program | 31 |
| 6.1 | Overview | 31 |
| 6.2 | Hardware Configuration | 31 |
| 6.3 | Software Configuration..... | 32 |
| 6.4 | Configuration of Directories and Files | 33 |
| 6.4.1 | Sample Program for the Remote I/O Board..... | 33 |
| 6.4.2 | Sample Program for the Communications Evaluation Board | 33 |
| 6.5 | Tutorial | 34 |
| 6.5.1 | Preparation..... | 34 |
| 6.5.2 | Connection with TwinCAT..... | 35 |
| 7. | Application Example: Migration from ET1100 | 36 |
| 7.1 | Overview | 36 |
| 7.2 | Hardware Related Items | 37 |
| 7.3 | Software Related Items..... | 38 |
| 7.3.1 | Sequence for Reading the Registers..... | 38 |
| 7.3.2 | Sequence for Writing to the Registers | 39 |
| 7.3.3 | Sequence for Sequential Reading from the Process Data RAM | 40 |
| 7.3.4 | Sequence for Sequential Writing to the Process Data RAM | 42 |
| 7.3.5 | Extended Registers: Sequence for Confirming Startup..... | 43 |
| 7.3.6 | Extended Registers: Software Reset Sequence..... | 44 |
| 7.4 | Items Related to Slave Stack Code | 44 |
| 7.4.1 | Hardware Access Files | 44 |
| 7.4.2 | Changing the ESI File | 46 |
| 7.4.3 | Adding the Startup Confirmation Sequence | 46 |
| 8. | Documents for Reference..... | 47 |
| 8.1 | Related Documents..... | 47 |
| 8.2 | Relationship between This Document and Related Documents..... | 47 |
| | Website and Support | 48 |

1. Specifications

Table 1.1 lists the peripheral modules used and their applications. Figure 1.1 shows the operating environment of the sample program which includes the library.

Table 1.1 Peripheral Modules and Their Applications

| Peripheral Module | Application |
|------------------------------------|--|
| EtherCAT slave controller | For use in EtherCAT communications |
| EtherCAT interrupt output signals | CATIRQ, CATSYNC0, and CATSYNC1 are output to an external MCU. |
| Ethernet MAC(ETHERC) | For use in EtherCAT communications |
| Serial peripheral interface (RSPI) | For use in transfer to and from an external MCU (CH0 or CH1 is selectable) |
| Interrupt controller | The reception buffer full, transmission buffer empty, and RSOI error interrupts are used as sources to generate interrupts for the RSPI. |
| Tightly coupled memory | The ATCM is used as the program and variable area for the library. |

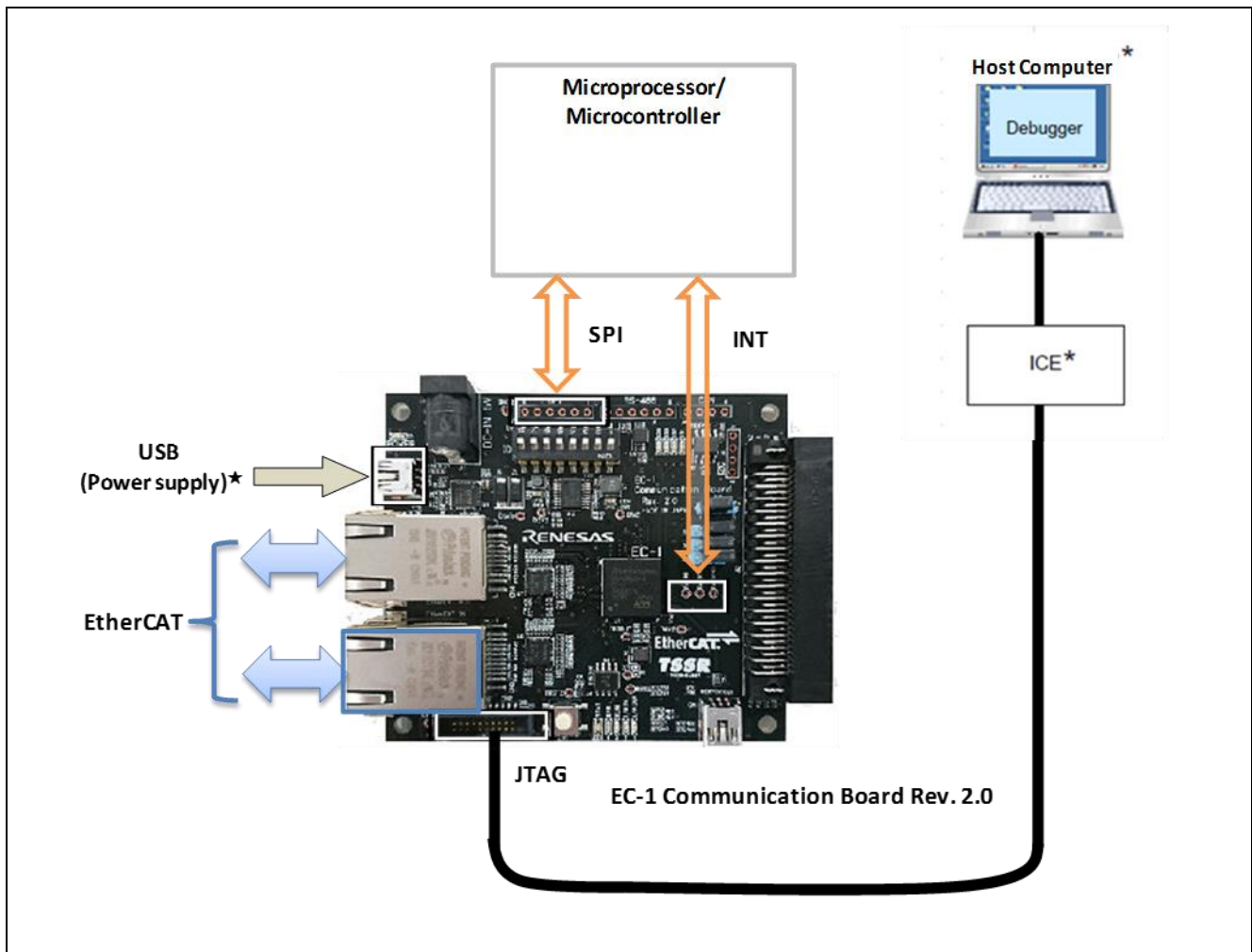


Figure 1.1 Operating Environment

Note: Use the EtherCAT IN port to connect an EtherCAT master.

2. Operating Environment

The library covered in this application note is for the environment below.

Table 2.1 Operating Environment

| Item | Description |
|------------------------------------|---|
| MCU used | EC-1 communications evaluation board EC-1 Communication Board Rev. 2.0 |
| CPU | EC-1 R9A06G043 |
| Operating frequency | CPU clock (CPUCLK): 150 MHz System clock: 25 MHz |
| Operating voltage | 3.3 V |
| Operating mode | SPI boot mode |
| Devices used | <ul style="list-style-type: none"> • Serial flash memory W25Q32JVSFIM from Winbond • EEPROM R1EX24016ASAS0 from Renesas Electronics • Ethernet PHY TLK105 from Texas Instruments |
| Communications protocol | EtherCAT® |
| Integrated development environment | Embedded Workbench® for ARM, version 7.70.1, from IAR Systems |
| Emulator | I-jet from IAR Systems |
| SSC Tool | Slave Stack Code (SSC) Tool, version 5.11, from EtherCAT Technology Group (ETG) |
| Software PLC | TwinCAT® 3 from Beckhoff Automation GmbH |

3. Hardware

3.1 Example of the Hardware Configuration

Figure 3.1 is an example of the hardware configuration where the EC-1 that incorporates the library is combined as an EtherCAT slave with an external MCU.

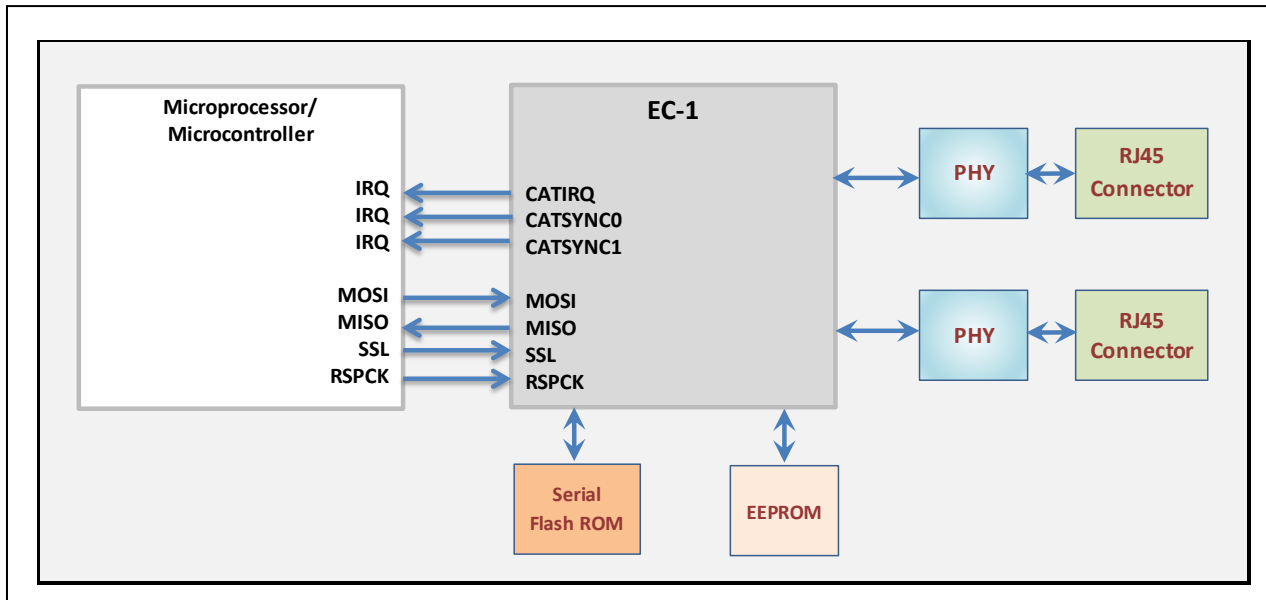


Figure 3.1 Example of the Hardware Configuration of an EtherCAT Slave

The EtherCAT slave transfers data to and from the external MCU by using the four-wire SPI. The four signals are MOSI (Master Out Slave In), MISO (Master In Slave Out), SSL (Slave Select), and RSPCK (RSPI Clock).

The external MCU is the SPI master and the EC-1 is the SPI slave.

The EC-1 outputs CATIRQ, CATSYNC0, and CATSYNC1 as synchronous interrupt signals for EtherCAT.

3.2 Pins

Table 3.1 lists the pins used and their functions.

Table 3.1 Pins Used and Their Functions

| Pin Name | I/O | Description |
|-------------|--------|---|
| ETH_MDIO | I/O | Input and output of the management data signal |
| ETH_MDC | Output | Output of the management interface clock |
| ETH0_RXC | I/O | Input and output of the reception clock |
| ETH1_RXC | | |
| ETH0_RXER | Input | Input of the received data error signal |
| ETH1_RXER | | |
| ETH0_RXDV | Input | Input of the received data enable signal |
| ETH1_RXDV | | |
| ETH0_RXD0~3 | Input | Input of the received data signal |
| ETH1_RXD0~3 | | |
| ETH0_TXC | Input | Input of the 10 M/100 M transmission clock (2.5 MHz/25 MHz) |
| ETH1_TXC | | |
| ETH0_TXER | Output | Output of the transmission error signal |
| ETH1_TXER | | |
| ETH0_TXEN | Output | Output of the transmission enable signal |
| ETH1_TXEN | | |
| ETH0_TXD0~3 | Output | Output of the transmission data signal |
| ETH1_TXD0~3 | | |
| ETH0_COL | Input | Input of the collision detection signal |
| ETH1_COL | | |
| ETH0_CRS | Input | Input of the carrier sense signal |
| ETH1_CRS | | |
| CLKOUT25M0 | Output | Output of the Ethernet PHY external clock |
| CLKOUT25M1 | | |
| PHYLINK0 | Input | Input of the PHY Link signal (for Ethernet Switch) |
| PHYLINK1 | | |
| ETH0_INT | Input | Input of the Ethernet PHY interrupt request signal |
| ETH1_INT | | |
| CATI2CCLK | Output | Output of the EtherCAT EEPROM I2C clock signal |
| CATI2CDATA | I/O | Input and output of the EtherCAT EEPROM I2C data signal |
| CATLINKACT1 | Output | Output of the EtherCAT Link/Activity LED signal |
| CATLINKACT0 | | |
| CATLEDRUN | Output | Output of the EtherCAT RUN LED signal |
| CATLEDSTER | Output | Output of the EtherCAT dual-color state LED signal |
| CATLEDERR | Output | Output of the EtherCAT error LED signal |
| CATIRQ | Output | Output of EtherCAT IRQ |
| CATSYNC0 | Output | Output of EtherCAT SYNC0 |
| CATSYNC1 | Output | Output of EtherCAT SYNC1 |
| RSPCK0 | Input | Input of the RSPI clock |
| MISO0 | Output | Output of RSPI slave send data |
| MOSI0 | Input | Input of RSPI master send data |
| SSL00 | Input | Input of the RSPI slave select signal |

4. SPI Interface

The EC-1 has two serial peripheral interfaces (RSPI).

The library allows the selection of channel 0 or 1 through parameters at the time of initialization.

Table 4.1 lists the specifications of the SPI and Figure 4.1 is a timing chart of the SPI.

Table 4.1 SPI Specifications

| Item | Description |
|--------------------------|---|
| Method of communications | Four-wire SPI |
| Transfer mode | Slave mode |
| Method of communications | Full duplex |
| Signal sense | SSL: Active low RSPCK: Low in the idle state |
| Transfer frame | 16 bits × 3 words |
| Format | MSB first |
| Data sampling timing | Falling edges of RSPCK |
| Maximum transfer rate | 18.75 Mbps |

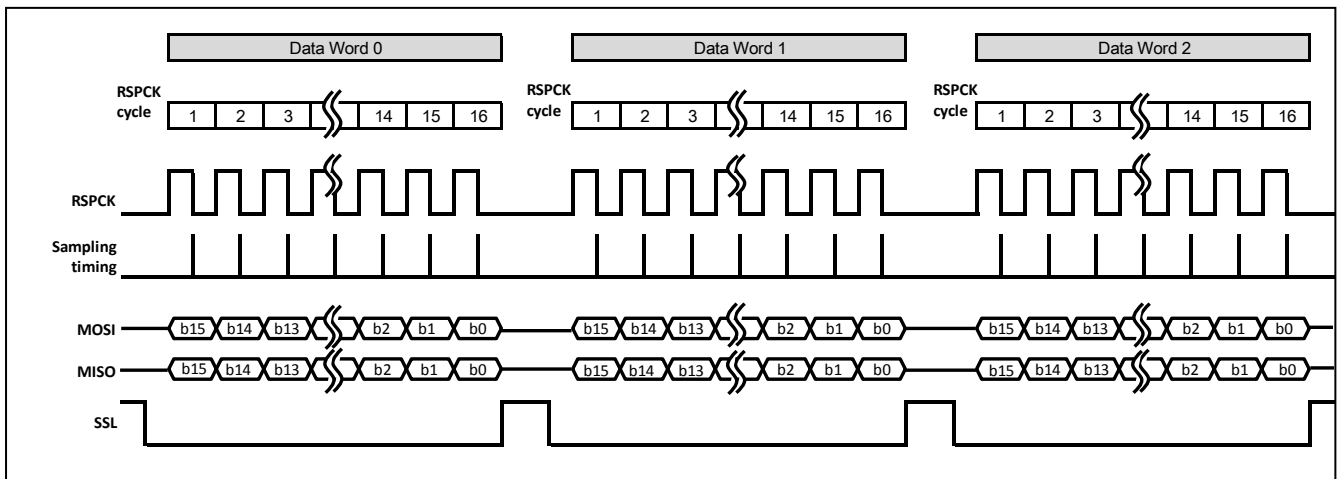


Figure 4.1 Timing Chart of the SPI

4.1 Example of the Software Configuration

Figure 4.2 is an example of the software configuration where the EC-1 is combined as an EtherCAT slave with an external MCU.

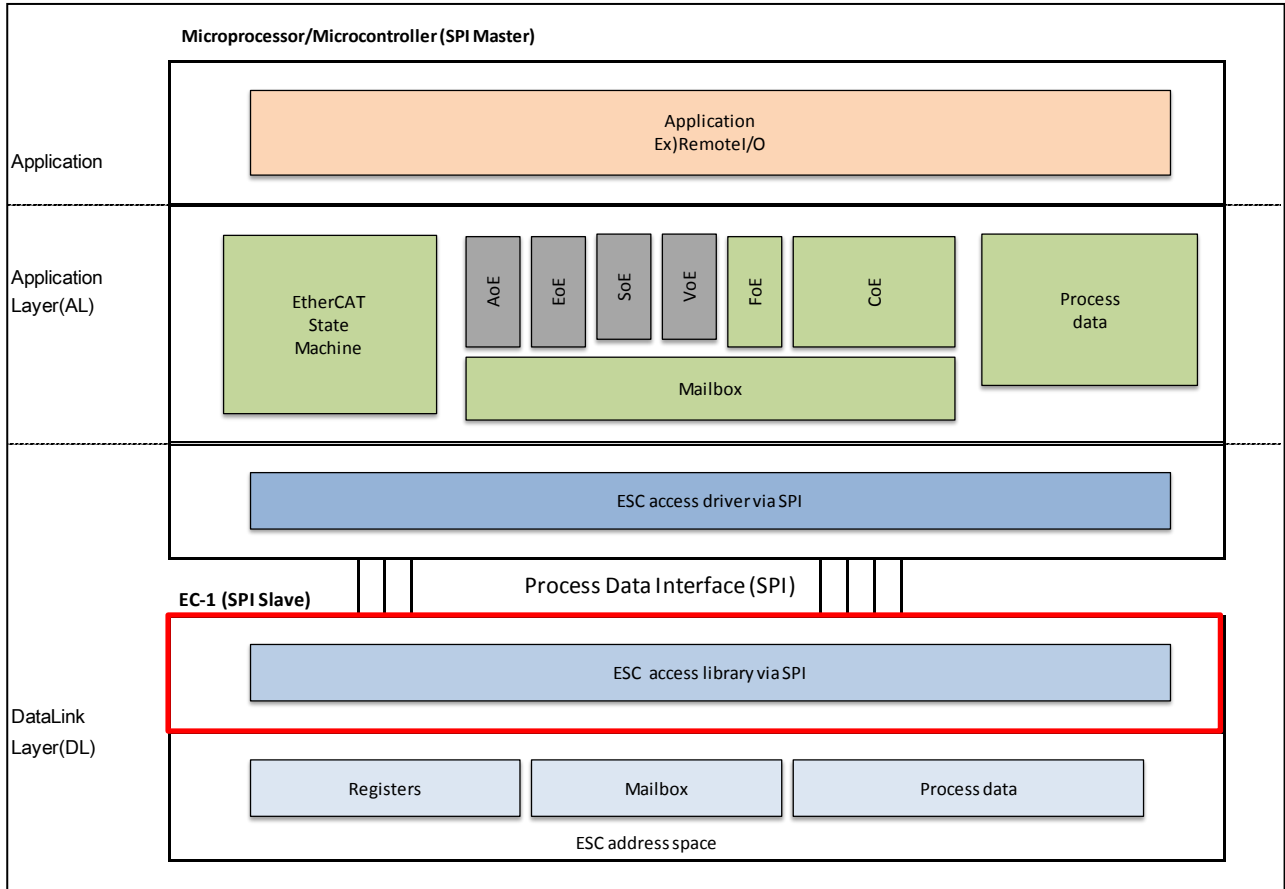


Figure 4.2 Example of the Software Configuration of an EtherCAT Slave

SSCTool: Program generation tool for EtherCAT control from Beckhoff Automation GmbH

The external MCU operates as an SPI master and has drivers for access to the application, EtherCAT protocol stack, and the ESC through the SPI interface.

The EC-1 operates as an SPI slave and returns the result of access to the ESC in response to instructions from the external MCU. The section in the red frame of the figure is the library.

The next and subsequent sections describe the library.

Note that the procedure for access by the external MCU to the ESC through the SPI interface is described in section 7.3.

4.2 SPI Address Space

Figure 4.3 shows the address space for access by the external MCU to the EC-1 through the SPI interface. Of the total space of 16 Kbytes, the ESC of the EC1 is allocated to the 12 Kbytes from 0x0000 to 0x2FFF. The 512 bytes from 0x3000 to 0x31FF are allocated to the extended registers for SPI communications.

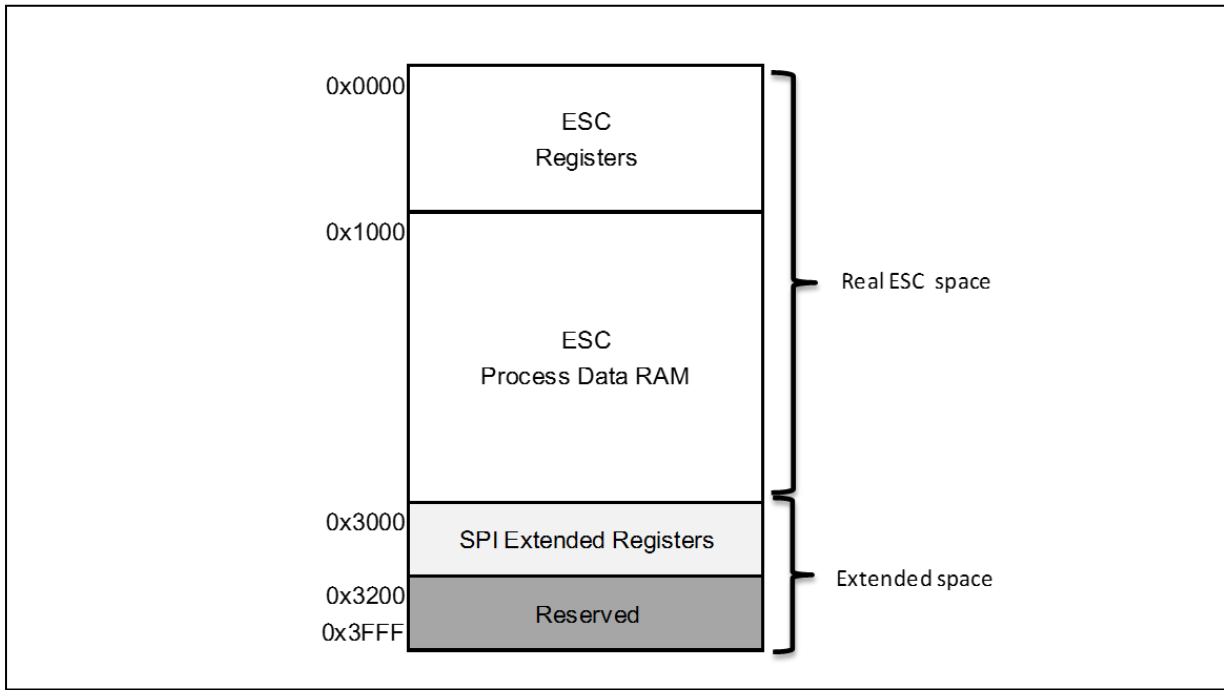


Figure 4.3 SPI Address Space

4.3 Extended SPI Registers

The Extended SPI registers are accessible in 16- or 32-bit units.

Table 4.2 lists the Extended SPI registers.

Note that R and W indicate the directions of access as seen from the external MCU.

Table 4.2 Extended SPI Registers

| Address | Register Name | Bit | R/W |
|----------------------------|--|-----|-----|
| 0x3000 | Device ID Register (DEVICEID) | 32 | R |
| 0x3004 | Revision number register (REVNO) | 16 | R |
| 0x3006 | Status register (STATUS) | 16 | R |
| 0x3008 | Control register (CONTROL) | 16 | W |
| 0x300A +2n ^{(*)1} | Scratch registers (SCRATCH0 to 9) | 16 | R/W |
| 0x301E | Sequential data number register (SQDATNUM) | 16 | R |
| 0x3020+4n ^{(*)2} | Sequential data registers (SQDATA0 to 63) | 32 | R |
| 0x3120-0x31FE | Reserved | --- | --- |

Note 1. n = 0 to 9
 Note 2. n = 0 to 63

4.3.1 Device ID Register (DEVICEID)

| Address | Bit | R/W | Description |
|---------|------|-----|---|
| 0x3000 | 31:0 | R | The device ID can be read here. "EC-1" (0x45435E31) is stored in ASCII code. |

4.3.2 Revision Number Register (REVNO)

| Address | Bit | R/W | Description |
|---------|------|-----|---|
| 0x3004 | 15:0 | R | The version number of the library can be read here. "EC-1" (0x45435E31) is stored in ASCII code. |

4.3.3 Status Register (STATUS)

| Address | Bit | R/W | Description |
|---------|------|-----|--|
| 0x3006 | 15:0 | R | The error code which has occurred during operation of the library can be read here. 0x0000: No error 0x0001: The previously received command is invalid. 0x0002: The word length specified in the previously received command is invalid. 0x0003: The address specified in the previously received command is invalid. 0x0004: The consecutive values that were previously received are invalid. 0x0005: The control register settings that were previously received are invalid. 0x0011: An overrun error occurred. 0x0014: A mode fault error occurred. 0x0018: A parity error occurred (not used this time). |

4.3.4 Control Register (CONTROL)

| Address | Bit | R/W | Description |
|---------|------|-----|--|
| 0x3008 | 15:0 | W | When a control code is set in this register, operation proceeds according to the control code. 0x0001: The error code of the status register is cleared and set to "no error" (0x0000). 0x0080: The EC-1 is rebooted by the application of a software reset. |

4.3.5 Scratch Registers (SCRATCH0 to 9)

| Address | Bit | R/W | Description |
|---------------|------|-----|--|
| 0x300A +2n | 15:0 | R/W | This is a 16-bit register for general use. It is also available for use in transferring data and state information to and from the external MCU. |

Note: n = 0 to 9

4.3.6 Sequential Data Number Register (SQDATNUM)

| Address | Bit | R/W | Description |
|---------|------|-----|--|
| 0x301E | 15:0 | R | The number of transfers in the last R/W sequence to have been received can be read here. |

4.3.7 Sequential Data Registers (SQDATA0 to 63)

| Address | Bit | R/W | Description |
|---------------|------|-----|--|
| 0x3020 +4n | 31:0 | R | In the case of the SQRD command, this holds the data read from the DATAR register. In the case of the SQWR command, this holds the data for writing to the DATAWR register. |

Note: n = 0 to 63

4.4 Format of SPI Packets

4.4.1 Commands

Table 4.3 lists the commands for access by the external MCU to the SPI address space in the EC-1.

For the sequence for access by using the command, see section 7.3.1, Sequence for Reading the Registers to section 7.3.4, Sequence for Sequential Writing to the Process Data RAM.

Table 4.3 SPI Commands

| CMD [7:0] | Command Name | Function |
|-----------|--------------|---|
| 0x01 | RD | This command is for reading the ESC or extended registers. |
| 0x02 | WR | This command is for writing to the ESC or extended registers. |
| 0x03 | SQRD | This command initiates sequential reading from the ESC. |
| 0x04 | SQWR | This command initiates sequential writing to the ESC. |
| 0x05 | DTRD | This command is for reading data form the sequential data register. |
| 0x06 | DTWR | This command is for writing data to the sequential data register. |

4.4.2 RD Command

This command is for reading of the ESC or extended registers by the external MCU.

The length of the data and the address from which to read are specified.

1, 2, or 4 bytes are specifiable as lengths.

On receiving the RD command, the EC-1 reads the specified register and sends the results.

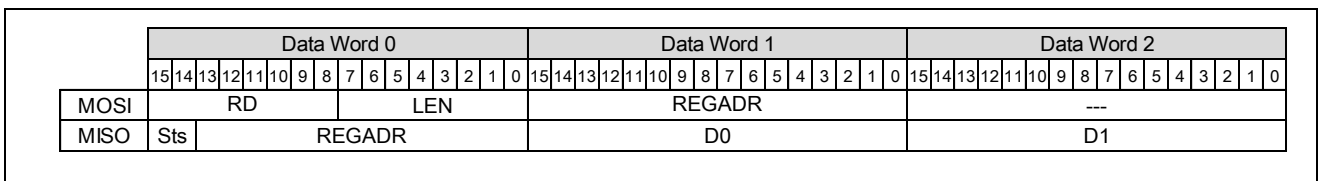


Figure 4.4 Format of RD Command Packets

Table 4.4 RD Command Packets for MOSI (external MCU → EC-1)

| Word No. | Bit | Name | Description |
|----------|------|--------|--|
| 0 | 15:8 | RD | RD command (0x01) |
| | 7:0 | LEN | These bits set the number of bytes to be read. Allowable settings are 1, 2, and 4 bytes. |
| 1 | 15:0 | REGADR | These bits specify the addresses of the ESC or extended registers from which to read. The specifiable address range is from 0x0000 to 0x3FFF. |
| 2 | 15:0 | --- | These bits cannot be used with the RD command. The value is “don’t care”. |

Table 4.5 RD Command Packets for MISO (EC-1 → external MCU)

| Word No. | Bit | Name | Description |
|----------|-------|------|--|
| 0 | 15:14 | Sts | Bit 15: ERROR Indicates the error state of the RD command. 1: An error has occurred. 0: No error has occurred. Bit 14: BUSY Indicates the state of execution of the RD command. 1: The command is being executed. 0: End of the command |
| | 13:0 | | REGADR |
| 1 | 15:0 | D0 | These bits indicate the results of reading the register specified by the RD command while BUSY = 0. The data are flush left when the length of data is 1 or 2 bytes. |
| 2 | 15:0 | D1 | |

4.4.3 WR Command

This command is for writing of the ESC or extended registers by the external MCU.

The length of the data and the address to which to write are specified.

1, 2, or 4 bytes are specifiable as lengths.

On receiving the WR command, the EC-1 writes to the specified register.

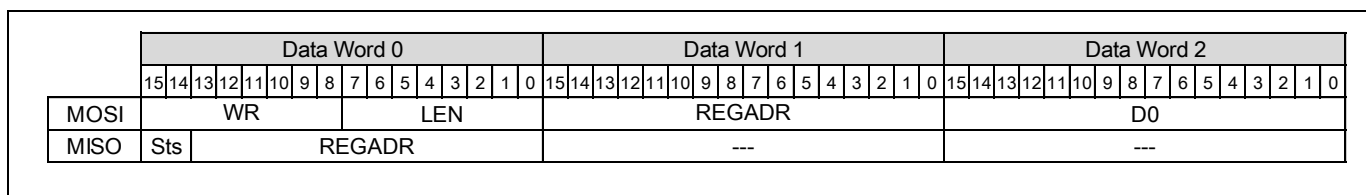


Figure 4.5 Format of SPI Packets for the WR Command

Table 4.6 WR Command Packets for MOSI (external MCU → EC-1)

| Word No. | Bit | Name | Description |
|----------|------|--------|---|
| 0 | 15:8 | WR | WR command (0x02) |
| | 7:0 | LEN | These bits set the number of bytes for writing. Allowable settings are 1 and 2 bytes. |
| 1 | 15:0 | REGADR | These bits specify the addresses of the ESC or extended registers to which to write. The specifiable address range is from 0x0000 to 0x3FFF. |
| 2 | 15:0 | D0 | These bits set a value for writing to the register. The data are flush left when the length of data is 1 byte. |

Table 4.7 WR Command Packets for MISO (EC-1 → external MCU)

| Word No. | Bit | Name | Description |
|----------|-------|--------|--|
| 0 | 15:14 | Sts | Bit 15: ERROR Indicates the error state of the WR command. 1: An error has occurred. 0: No error has occurred. Bit 14: BUSY Indicates the state of execution of the WR command. 1: The command is being executed. 0: End of the command |
| | 13:0 | REGADR | These bits indicate the address of the register specified by the WR command. |
| 1 | 15:0 | --- | These bits cannot be used with the WR command. The value is “don’t care”. |
| 2 | 15:0 | --- | |

4.4.4 SQRD Command

This command is for the sequential reading of the process data RAM of the ESC in the EC-1 by the external MCU.

The length of data and the address where reading of the register is to start are specified.

Up to 255 bytes are specifiable as lengths.

On receiving an SQRD command, the EC-1 reads the specified registers and stores the results in the sequential data register.

The external MCU can read the results of reading from the sequential data register by issuing a DTRD command.

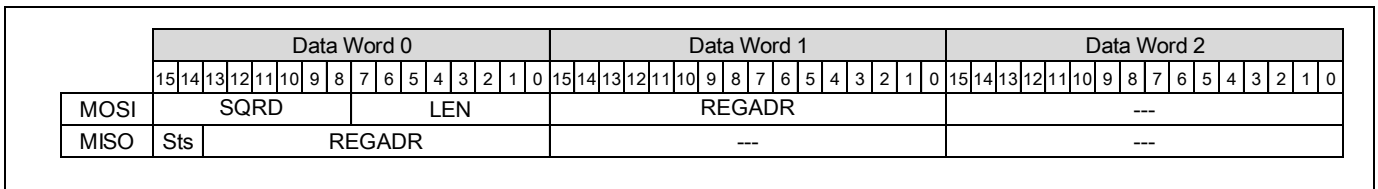


Figure 4.6 Format of SPI Packets for the SQRD Command

Table 4.8 SQRD Command Packets for MOSI (external MCU → EC-1)

| Word No. | Bit | Name | Description |
|----------|------|--------|---|
| 0 | 15:8 | SQRD | SQRD command (0x03) |
| | 7:0 | LEN | These bits set the number of bytes for sequential reading. Allowable settings are up to 255 bytes. |
| 1 | 15:0 | REGADR | These bits specify the addresses of the ESC from which to read. The specifiable address range is from 0x0000 to 0x2FFF. |
| 2 | 15:0 | --- | These bits cannot be used with the SQRD command. The value is “don’t care”. |

Table 4.9 SQRD Command Packets for MISO (EC-1 → external MCU)

| Word No. | Bit | Name | Description |
|----------|-------|--------|--|
| 0 | 15:14 | Sts | Bit 15: ERROR Indicates the error state of the SQRD command. 1: An error has occurred. 0: No error has occurred. Bit 14: BUSY Indicates the state of execution of the SQRD command. 1: The command is being executed. 0: End of the command |
| | 13:0 | REGADR | These bits indicate the address of the register specified by the SQRD command. |
| 1 | 15:0 | D0 | These bits cannot be used with the SQRD command. The value is “don’t care”. |
| 2 | 15:0 | D1 | |

4.4.5 SQWR Command

This command is for the sequential writing of the process data RAM of the ESC in the EC-1 by the external MCU.

The length of data and the address where writing of the register is to start are specified.

Up to 255 bytes are specifiable as lengths.

The data for writing must have been stored in the sequential data register beforehand by issuing a DTWR command.

On receiving an SQWR command, The EC-1 writes data stored in the sequential data register to the ESC.

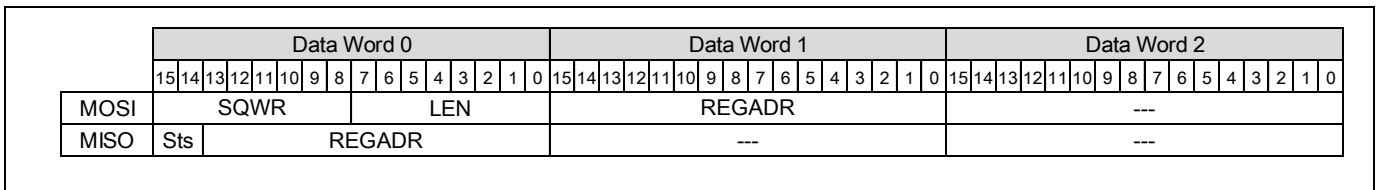


Figure 4.7 Format of SPI Packets for the SQWR Command

Table 4.10 SQWR Command Packets for MOSI (external MCU → EC-1)

| Word No. | Bit | Name | Description |
|----------|------|--------|---|
| 0 | 15:8 | SQWR | SQWR command (0x04) |
| | 7:0 | LEN | These bits set the number of bytes for sequential writing. Allowable settings are up to 255 bytes. |
| 1 | 15:0 | REGADR | These bits specify the addresses of the ESC to which to write. The specifiable address range is from 0x0000 to 0x2FFF. |
| 2 | 15:0 | --- | These bits cannot be used with the SQWR command. The value is “don’t care”. |

Table 4.11 SQWR Command Packets for MISO (EC-1 → external MCU)

| Word No. | Bit | Name | Description |
|----------|-------|--------|--|
| 0 | 15:14 | Sts | Bit 15: ERROR Indicates the error state of the SQWR command. 1: An error has occurred. 0: No error has occurred. Bit 14: BUSY Indicates the state of execution of the SQWR command. 1: The command is being executed. 0: End of the command |
| | 13:0 | REGADR | These bits indicate the address of the register specified by the SQWR command. |
| 1 | 15:0 | D0 | These bits cannot be used with the SQWR command. The value is “don’t care”. |
| 2 | 15:0 | D1 | |

4.4.6 DTRD Command

This command is for reading the sequential data register by the external MCU.

Specify the length of data to be read. Specify four bytes for reading data other than the last of it. For example, to read 10 bytes, specify four bytes for the first and second DTRD commands and specify the remaining two bytes for the third DTRD command.

The EC-1 reads and transmits data from sequential data register 0 in order on receiving the DTRD command.

In the case above, it reads and transmits: four bytes from sequential data register 0 in response to the first DTRD command, four bytes from sequential data register 1 in response to the second DTRD command, and two bytes from sequential data register 2 in response to the third DTRD command.

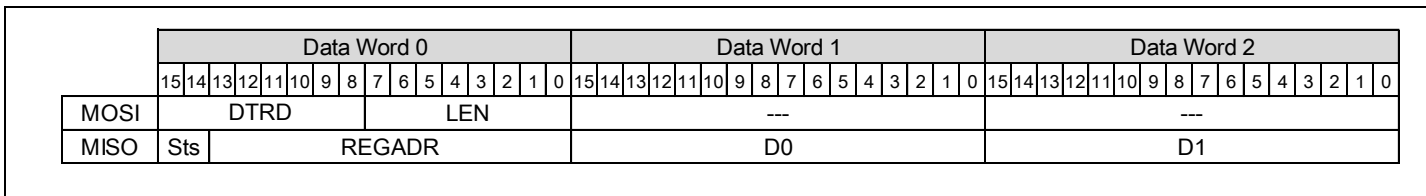


Figure 4.8 Format of SPI Packets for the DTRD Command

Table 4.12 DTRD Command Packets for MOSI (external MCU → EC-1)

| Word No. | Bit | Name | Description |
|----------|------|------|---|
| 0 | 15:8 | DTRD | DTRD command (0x05) |
| | 7:0 | LEN | These bits set the number of bytes to be read in response to the DTRD command. Allowable settings are 1 to 4 bytes. |
| 1 | 15:0 | --- | These bits cannot be used with the DTRD command. The value is “don’t care”. |
| 2 | 15:0 | --- | |

Table 4.13 DTRD Command Packets for MISO (EC-1 → external MCU)

| Word No. | Bit | Name | Description |
|----------|-------|--------|--|
| 0 | 15:14 | Sts | Bit 15: ERROR Indicates the error state of the DTRD command. 1: An error has occurred. 0: No error has occurred. Bit 14: BUSY Indicates the state of execution of the DTRD command. 1: The command is being executed. 0: End of the command |
| | 13:0 | REGADR | These bits indicate the base address of the ESC register read. |
| 1 | 15:0 | D0 | These bits indicate the results of reading the register. |
| 2 | 15:0 | D1 | The data are flush left when the length of data is less than 4 bytes. |

4.4.7 DTWR Command

This command is for writing data to the sequential data register by the external MCU.

Specify the length of data and the data for writing. Specify four bytes for writing data other than the last of it. For example, to write 10 bytes, specify four bytes for the first and second DTWR commands and specify the remaining two bytes for the third DTWR command.

The EC-1 writes the data which have been received from sequential data register 0 in order on receiving the DTWR command.

In the case above, it writes: four bytes to sequential data register 0 in response to the first DTWR command, four bytes to sequential data register 1 in response to the second DTWR command, and two bytes to sequential data register 2 in response to the third DTWR command.

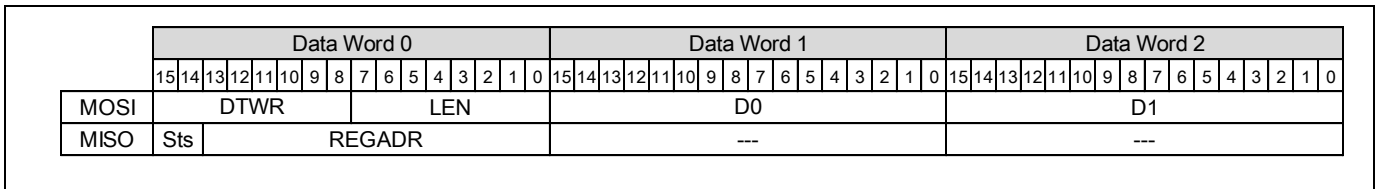


Figure 4.9 Format of SPI Packets for the DTWR Command

Table 4.14 DTWR Command Packets for MOSI (external MCU → EC-1)

| Word No. | Bit | Name | Description |
|----------|------|------|--|
| 0 | 15:8 | DTWR | DTWR command (0x06) |
| | 7:0 | LEN | These bits set the number of bytes for writing in response to the DTWR command. Allowable settings are 1 to 4 bytes. |
| 1 | 15:0 | D0 | These bits set the data for writing to the register. |
| 2 | 15:0 | D1 | The data are flush left when the length of data is less than 4 bytes. |

Table 4.15 DTWR Command Packets for MISO (EC-1 → external MCU)

| Word No. | Bit | Name | Description |
|----------|-------|--------|--|
| 0 | 15:14 | Sts | Bit 15: ERROR Indicates the error state of the DTWR command. 1: An error has occurred. 0: No error has occurred. Bit 14: BUSY Indicates the state of execution of the DTWR command. 1: The command is being executed. 0: End of the command |
| | 13:0 | REGADR | These bits indicate the base address of the sequential data register where data are to be stored in response to the DTWR command. |
| 1 | 15:0 | D0 | These bits cannot be used with the DTWR command. The value is "don't care". |
| 2 | 15:0 | D1 | |

5. Software

5.1 Operation in Outline

Table 5.1 gives an outline of the operation of the ESC SPI connection library. Figure 5.1 is a block diagram of the system.

Table 5.1 Operation in Outline

| Function | Outline |
|-----------------------|---|
| Outline of processing | Handles transfer to and from an external MCU as the SPI master and reads from and writes to the ESC and Extended SPI registers. |
| RSPI settings | Channel: CH0 or CH1 is selectable. SPI operation: Four-wire connection Transfer mode: Slave mode (transmission and reception) Data length: 16 bits × 3 words Bit rate: Up to 18.75 Mbps |
| Interrupt sources | Interrupt sources RSPI reception buffer full interrupt RSPI transmission buffer empty interrupt RSPI error interrupt |

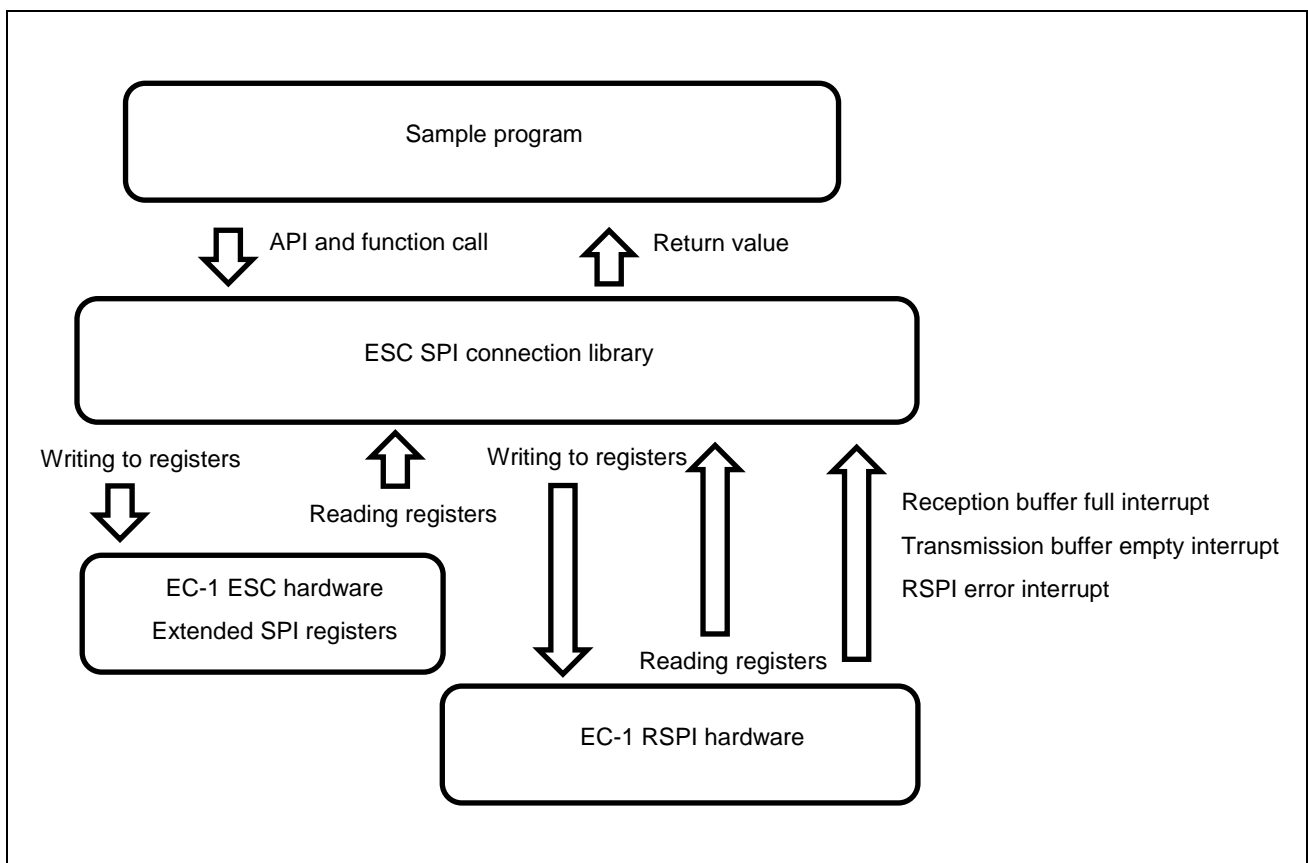


Figure 5.1 Block Diagram of the System

5.2 Procedure for Designing Software

Figure 5.2 is a schematic view of the configuration of the files.

In the sample program, the initial settings file for the board is for the EC-1 communications evaluation board. Modify the file according to the hardware environment you will actually be using.

You can add user applications as required.

However, adding user applications with long occupancy times may lead to the loss of received data.

Take care to make sure that additions do not lead to errors in SPI communications.

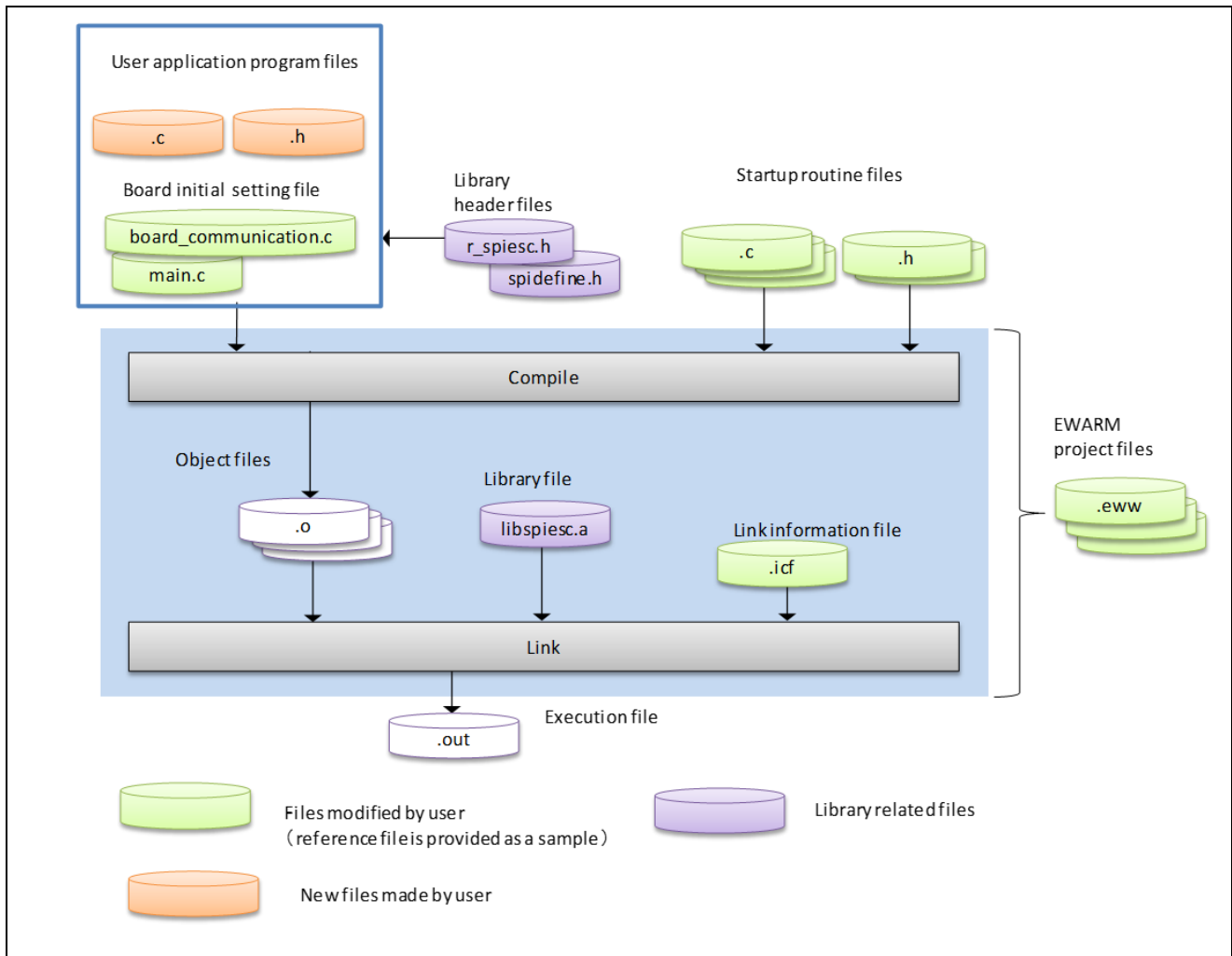


Figure 5.2 Schematic View of the Configuration of the Files

5.3 Interrupts

Table 5.2 lists the interrupts used in the library.

Table 5.2 Interrupts Used in the Library

| Interrupt Source (Source ID) | Priority | Processing in Outline |
|---|----------|--|
| RSPI reception buffer full interrupt (RSPI) | 0 | Processing for completion of reception |
| RSPI transmission buffer empty interrupt (SPTI) | 0 | Processing for completion of transmission |
| RSPI error interrupt (SPEI) | 0 | Overrun, parity, and mode fault error processing |

5.4 Fixed-Width Integers

Table 5.3 lists the fixed-width integers used in the library.

Table 5.3 Fixed-Width Integers Used in the Library

| Symbol | Description |
|----------|---|
| int8_t | 8-bit signed integer (defined in the standard library) |
| int16_t | 16-bit signed integer (defined in the standard library) |
| int32_t | 32-bit signed integer (defined in the standard library) |
| int64_t | 64-bit signed integer (defined in the standard library) |
| uint8_t | 8-bit unsigned integer (defined in the standard library) |
| uint16_t | 16-bit unsigned integer (defined in the standard library) |
| uint32_t | 32-bit unsigned integer (defined in the standard library) |
| uint64_t | 64-bit unsigned integer (defined in the standard library) |

5.5 Constants and Error Codes

Table 5.4 to Table 5.8 list the constants and Table 5.9 lists the error codes used in the library.

Table 5.4 Constants Used in the Library (Initial Settings)

| Constant | Setting | Description |
|-----------------|---------|---|
| SPIESC_CH0 | 0 | RSPI CH0 is selected. |
| SPIESC_CH1 | 1 | RSPI CH1 is selected. |
| SPIESC_LOG_NONE | 0 | Log output specification (no output) |
| SPIESC_LOG_DATA | 1 | Log output specification (output of transferred data) |
| SPIESC_LOG_MSG | 2 | Log output specification (output of error code) |

Table 5.5 Constants Used in the Library (Packet Settings)

| Constant | Setting | Description |
|-------------------|---------|---|
| SPI_PACKET | 3 | Number of words of SPI communications packets |
| SPI_SQDATA_BYTE | 4 | Maximum number of bytes which can be transferred at a time by using the DTRD and DTWR commands. |
| SPI_SQDATA_BUFNUM | 64 | Number of the sequential data registers |

Table 5.6 Constants Used in the Library (Commands)

| Constant | Setting | Description |
|--------------|---------|--------------|
| SPI_CMD_RD | 0x0001 | RD command |
| SPI_CMD_WR | 0x0002 | WR command |
| SPI_CMD_SQRD | 0x0003 | SQRD command |
| SPI_CMD_SQWR | 0x0004 | SQWR command |
| SPI_CMD_DTRD | 0x0005 | DTRD command |
| SPI_CMD_DTWR | 0x0006 | DTWR command |

Table 5.7 Constants Used in the Library (Extended SPI Registers)

| Constant | Setting | Description |
|-------------------|---------|---|
| SPIREG_MISC_BASE | 0x3000 | Extended SPI register base address |
| SPIREG_DEVICEID1 | 0x3000 | Device ID register address (higher-order word) |
| SPIREG_DEVICEID2 | 0x3002 | Device ID register address (lower-order word) |
| SPIREG_REVNO | 0x3004 | Revision number register address |
| SPIREG_STATUS | 0x3006 | Status register address |
| SPIREG_CTRL | 0x3008 | Control register address |
| SPIREG_SCRATCH0 | 0x300A | Scratch 0 register address |
| SPIREG_SQDATANUM | 0x301E | Sequential data number register address |
| SPIREG_SQDATA0 | 0x3020 | Sequential data register address |
| SPIREG_MISC_MAX16 | 0x311E | Upper-limit on the address when access to an extended SPI register is in 16-bit units |
| SPIREG_MISC_MAX32 | 0x311C | Upper-limit on the address when access to an extended SPI register is in 32-bit units |

Table 5.8 Constants Used in the Library (Control Code)

| Constant | Setting | Description |
|--------------------|---------|-----------------------------|
| SPICTRLCODE_STSCLR | 0x0001 | The error code is cleared. |
| SPICTRLCODE_RST | 0x0080 | The EC-1 software is reset. |

Table 5.9 Error Codes Used in the Library

| Constant | Setting | Description |
|--------------------------------|---------|---|
| SPISTATUSCODE_NOERROR | 0x0000 | No error has occurred. |
| SPISTATUSCODE_INVALID_CMD | 0x0001 | Error in the received command |
| SPISTATUSCODE_INVALID_LENGTH | 0x0002 | Erroneous word length in the received command |
| SPISTATUSCODE_INVALID_ADDR | 0x0003 | Erroneous address specified in the received command |
| SPISTATUSCODE_INVALID_SQDATA | 0x0004 | Error in the sequence of the DTRD and DTWR commands |
| SPISTATUSCODE_INVALID_CTRLCODE | 0x0005 | Error in the control code |
| SPISTATUSCODE_COMERROR | 0x0010 | Error in overall SPI communications |
| SPISTATUSCODE_OVRERROR | 0x0011 | Overflow error |
| SPISTATUSCODE_MODERROR | 0x0012 | Mode fault error |
| SPISTATUSCODE_PERERROR | 0x0014 | Parity error |

5.6 Structures, Unions, and Enumerations

Table 5.10 to Table 5.13 list the structures, unions, and enumerations used in the library.

Table 5.10 Structures and Unions Used in the Library (1)

| Definitions of Structures and Unions | Outline | Definition File |
|--------------------------------------|--|-----------------|
| spi_t | Definition of the structure for setting SPI packets <ul style="list-style-type: none"> • MOSI member The external MCU (master) sets data for transmission. The EC-1 (slave) acquires data as received data. • MISO member The EC-1 (slave) sets data for transmission. The external MCU (master) acquires data as received data. | spidefine.h |
| T_SPIREG_STD16 | Definition of the union for the 16-bit extended SPI registers | spidefine.h |
| T_SPIREG_STD32 | Definition of the union for the 32-bit extended SPI registers | spidefine.h |
| T_SPIREG_STATUS | Definition of the union for the status register | spidefine.h |
| spi_misc_t | Definition of the structure for the extended SPI registers as a whole | spidefine.h |

Table 5.11 Structures and Unions Used in the Library (2)

| Definitions of Structures and Unions | Member | | Description | |
|--------------------------------------|----------------|--|--------------------|---|
| spi_t | union CMDRD | BIT | uint32_t REGADR:14 | MISO REGADR setting |
| | | | uint32_t BUSY:1 | MISO Sts BUSY bit 1: The command is being executed. 0: End of the command |
| | | | uint32_t EEOR:1 | MISO Sts ERROR bit 1: An error has occurred. 0: No error has occurred. |
| | | | uint32_t: 16 | --- |
| | | BYTE | uint8_t B0 | MISO D1 higher-order byte |
| | | | uint8_t B1 | MISO D1 lower-order byte |
| | | | uint8_t B2 | MISO D0 higher-order byte |
| | | | uint8_t B3 | MISO D0 lower-order byte |
| | | WORD | uint16_t W0 | MISO D0 |
| | | | uint16_t W1 | MISO D1 |
| | uint32_t DWORD | | MISO D1/D0 | |
| | CMDWR | uint8_t CMD | | MOSI CMD setting |
| | | uint8_t LEN | | MOSI LEN setting |
| | | uint16_t REGADR | | MOSI REGADR setting |
| T_SPIREG_STD32 DATARD | | Read data setting for the RD and DTRD commands for MISO | | |
| T_SPIREG_STD32 DATAWR | | Write data setting for the WR and DTWR commands for MOSI | | |

Table 5.12 Structures and Unions Used in the Library (3)

| Definitions of Structures and Unions | Member | | Description |
|--------------------------------------|---------------|------------------|--|
| union T_SPIREG_STD16 | BYTE | uint8_t B0 | Generic definition of 16-bit registers Higher-order byte |
| | | uint8_t B1 | Lower-order byte |
| | uint16_t WORD | | Word |
| union T_SPIREG_STD32 | BYTE | uint8_t B0 | Generic definition of 32-bit registers Most significant byte |
| | | uint8_t B1 | Higher-order byte |
| | | uint8_t B2 | Lower-order byte |
| | | uint8_t B3 | Least-significant byte |
| | WORD | uint16_t W0 | Higher-order word |
| | | uint16_t W1 | Lower-order word |
| uint32_t DWORD | | Double word | |
| union T_SPIREG_STATUS | BIT | uint16_t ERROR:1 | Definition of the status register ERROR bit 1: An error has occurred. 0: No error has occurred. |
| | | uint16_t BUSY:1 | BUSY bit 1: The command is being executed. 0: End of the command |
| | | uint16_t :14 | --- |
| | BYTE | uint8_t B0 | Higher-order byte |
| | | uint8_t B1 | Lower-order byte |
| | uint16_t WORD | | Word |

Table 5.13 Structures and Unions Used in the Library (4)

| Definitions of Structures and Unions | Member | Description |
|--------------------------------------|-----------------------------|---|
| spi_misc_t | T_SPIREG_STD16 DEVICEID1 | Device ID register Higher-order word |
| | T_SPIREG_STD16 DEVICEID2 | Device ID register Lower-order word |
| | T_SPIREG_STD16 REVNO | Revision number register |
| | T_SPIREG_STD16 STATUS | Status register |
| | T_SPIREG_STD16 CTRL | Control register |
| | T_SPIREG_STD16 SCRATCH [10] | Scratch registers (0 to 9) |
| | T_SPIREG_STD16 SQDATANUM | Sequential data number register |
| | T_SPIREG_STD16 SQDATA [64] | Sequential data registers (0 to 63) |

5.7 Global Variables

Table 5.14 lists the global variables used in the library.

Table 5.14 Global Variables Used in the Library

| Type | Variable | Description | Function Used |
|------------|----------|---|--|
| spi_t | SPI | Information of SPI packet data settings | R_SPIEsc_Create() R_SPIEsc_Send_Receive() |
| spi_misc_t | SPIMISC | Information of extended SPI registers | R_SPIEsc_Create() R_SPIEsc_Send_Receive() |

5.8 Functions

Table 5.15 lists the functions in the library.

Table 5.15 Library Functions

| Function | Page Number |
|-----------------------|-------------|
| R_SPIEsc_Pin_Init | 26 |
| R_SPIEsc_Create | 26 |
| R_SPIEsc_Start | 28 |
| R_SPIEsc_Stop | 29 |
| R_SPIEsc_Send_Receive | 30 |

5.9 Functions

The following describes the specifications of the library functions.

5.9.1 Processing by Initial Settings for the SPI Pins

R_SPIESC_PinInit

(1) Outline

Processing by initial settings for the pins for SPI communications

(2) C-Language Format

```
void R_SPIESC_PinInit (uint_16t ch);
```

(3) Parameter

| I/O | Parameter | Description |
|-----|-------------|--|
| I | uint16_t ch | RSPI channels to be used in SPI communications SPIESC_CH0: Channel 0 RSPCK0, MISO0, MOSI0, and SSL00 are used. SPIESC_CH1: Channel 1 RSPCK1, MISO1, MOSI1, and SSL10 are used. |

(4) Function

This function makes initial settings for the pins to be used in SPI communications.

(5) Return Value

None

Remark 1. When executing this API function, make sure that writing to the PFS register is enabled by using the write protection register (PWPR).

5.9.2 Initialization Processing for SPI Control

R_SPIESC_Create(uint16_t log)

(1) Outline

Initialization processing for RSPI control

(2) C-Language Format

```
void R_SPIESC_Create(uint16_t log);
```

(3) Parameter

| I/O | Parameter | Description |
|-----|--------------|---|
| I | uint16_t log | Specifies the log output for SPI communications. SPIESC_LOG_NONE: A log is not output. SPIESC_LOG_DATA: A log of the transfer data is output. SPIESC_LOG_MSG: A log of the error codes set in the status register in response to errors is output. |

The logical OR of SPIESC_LOG_DATA and SPIESC_LOG_MSG can be set.

(4) Function

This function handles initialization which is required for control of SPI communications.

When the log output is specified, the log is output by using the printf function. Note that the destination for output depends on the implementation of the printf function.

The log output should only be used for debugging. If a log is output in normal operation, this may lead to received data being lost, constituting a communications error.

(5) Return Value

None

5.9.3 Starting SPI Communications

R_SPESC_Start

(1) Outline

Starting SPI communications

(2) C-Language Format

```
void R_SPESC_Start (void);
```

(3) Parameter

None

(4) Function

This function starts SPI communications for the channel specified for processing by the initial settings for the SPI pins.

(5) Return Value

None

5.9.4 Stopping SPI Communications

R_SPIESC_Stop

(1) Outline

Stopping SPI communications

(2) C-Language Format

```
void R_SPIESC_Stop (void);
```

(3) Parameter

None

(4) Function

This function stops SPI communications for the channel specified for processing by the initial settings for the SPI pins.

(5) Return Value

None

5.9.5 Processing for SPI Transfer

R_SPIESC_Send_Receive

(1) Outline

Processing for SPI transfer

(2) C-Language Format

MD_STATUS R_SPIESC_Send_Receive (uint32_t * Count);

(3) Parameter

| I/O | Parameter | Description |
|-----|----------------|---|
| I | uint32_t Count | Value counted based on the execution of transmission or reception |

(4) Function

This function handles SPI transfer to and from the external MCU (master) through the channel specified for processing by the initial settings for the SPI pins.

It reads from and writes to the ESC or extended SPI register in response to commands issued by the external MCU.

The timeout to wait for transfer is judged by using the value counted based on the execution of transmission or reception.

If data have been transferred, the function ends immediately.

(5) Return Value

| Return Value | Meaning |
|--------------|--|
| MD_OK | Normal completion |
| MD_ERROR1 | A timeout occurred. |
| MD_ERROR2 | An SPI communications error occurred. The error code is stored in SPIMISC.STATUS. |

6. Sample Program

6.1 Overview

This section describes a sample program which configures an EtherCAT slave with the EC-1 remote I/O board and the EC-1 communications evaluation board connected through an SPI interface.

- The EC-1 remote I/O board operates as an SPI master which runs the remote I/O application and EtherCAT protocol stack.
- The EC-1 communications evaluation board operates as an SPI slave which handles hardware access to the ESC.

6.2 Hardware Configuration

Figure 6.1 shows the hardware configuration for the sample program.

To run the sample program, connect pins of the EC-1 remote I/O board and EC-1 communications evaluation board as listed in Table 6.1. Note that the wiring runs should be as short as is possible.

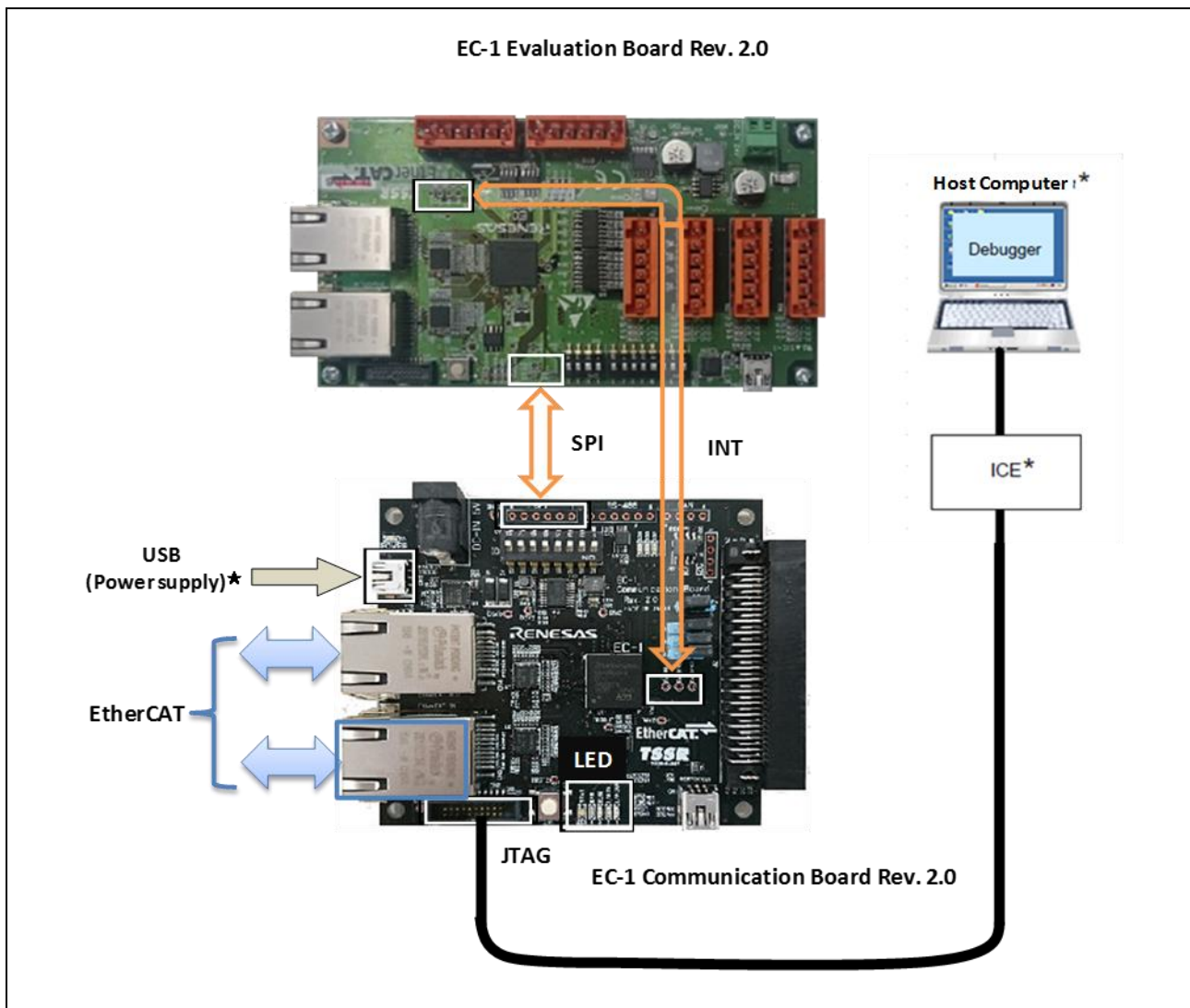


Figure 6.1 Hardware Configuration for the Sample Program

Table 6.1 Connecting Pins of the Boards

| EC-1 Remote I/O Board (SPI Master) | | EC-1 Communications Evaluation Board (SPI Slave) | |
|------------------------------------|----------------------------------|--|-----------|
| Signal | Connector Pin Number Test Pin | Connector Pin Number Test Pin | Signal |
| PG5/SSL10 | J1 #7 | J7 #4 | P75/SSL00 |
| PG4/MOSI1 | J1 #5 | J7 #3 | PA1/MOSI0 |
| PG3/MISO1 | J1 #3 | J7 #2 | PA0/MISO0 |
| PG2/RSPCK1 | J1 #1 | J7 #1 | P77/SPCK0 |
| IRQ4/P94 | P94 | CATIRQ | CATIRQ |
| IRQ13/95 | P95 | CATSYNC0 | CATSYNC0 |
| IRQ7/P97 | P97 | CATSYNC1 | CATSYNC1 |

6.3 Software Configuration

Figure 6.2 shows the software configuration for the sample program.

- The sample program for the remote I/O board is for operation as an SPI master, which consists of the application itself, the EtherCAT protocol stack, and drivers for access to the ESC through the SPI interface.
- The communications evaluation board sample program causes the board to operate as an SPI slave, which returns the results of access to the ESC in response to instructions from the remote I/O board.

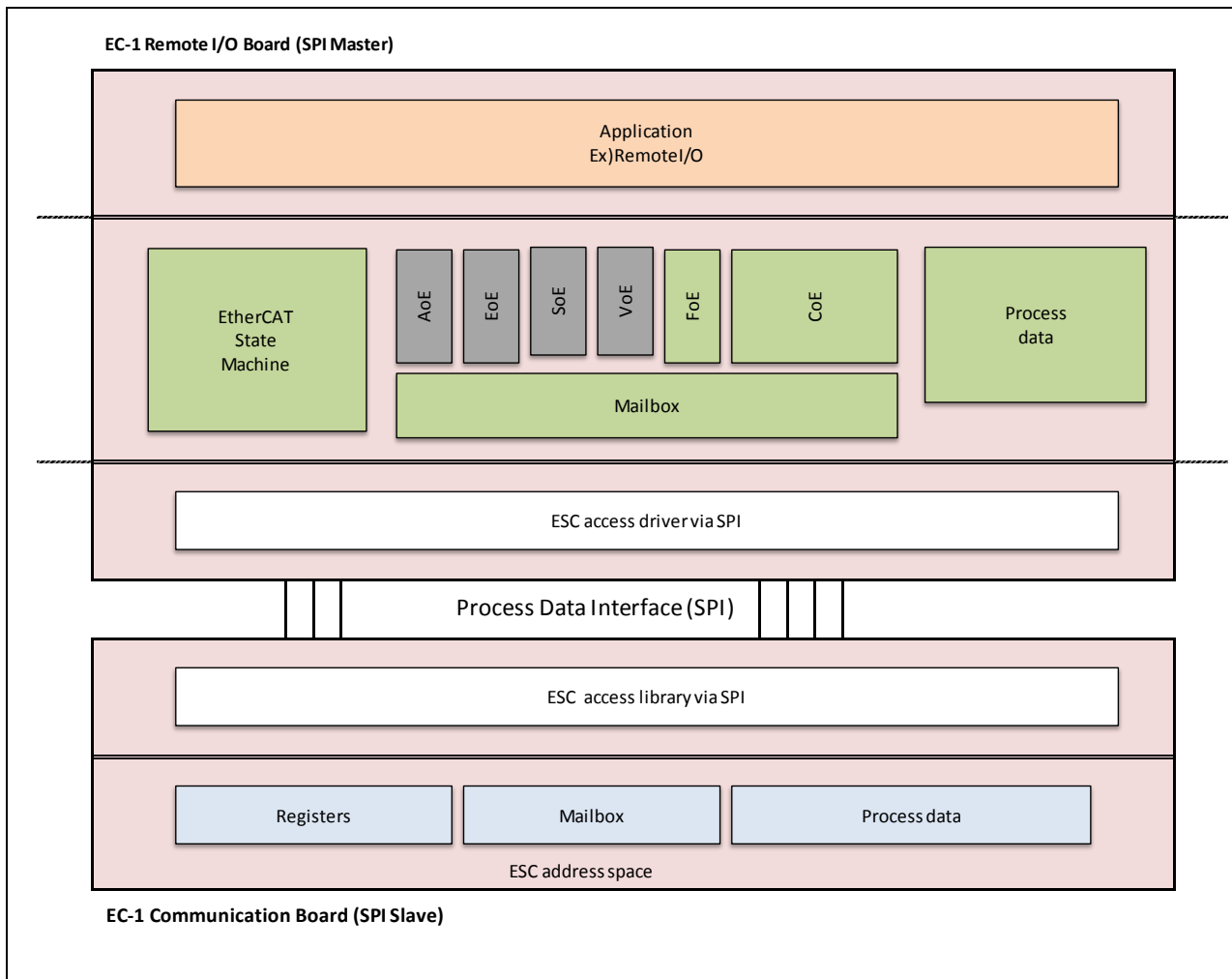


Figure 6.2 Software Configuration for the Sample Program

6.4 Configuration of Directories and Files

6.4.1 Sample Program for the Remote I/O Board

This sample program is based on the regular version of the sample program, which runs on a single remote I/O board by using the internal ESC module of the EC-1.

Table 6.2 lists the files which have been added or changed in this sample program from the regular version of the program. The other files have the same configuration as in the regular version.

The RSPI driver is set to operate at a transfer rate of 12.5 Mbps in SPI master mode based on the SPI communications specification.

The hardware access files `renesashw.c` and `renesashw.h` for access to the ESC have been changed so they now handle access through the SPI.

Table 6.2 Configuration of the Files of the Remote I/O Board Sample Program

| Directory | File | Description | Classification |
|------------------------|----------------------------|---|----------------|
| <code>rspi/src/</code> | <code>r_rspi.c</code> | RSPI CH1 SPI master mode driver | Addition |
| | <code>r_rspi_user.c</code> | RSPI user definition | Addition |
| <code>rspi/inc/</code> | <code>r_rspi.h</code> | Prototype declaration of the RSPI driver | Addition |
| | <code>spidefine.h</code> | Definitions of structures and unions for access to the SPI and its extended registers | Addition |
| <code>SSC/</code> | <code>renesashw.h</code> | Hardware access file prototype declaration | Change |
| | <code>renesashw.c</code> | Hardware access file | Change |
| <code>SSC/Src/</code> | <code>main.c</code> | Remote I/O sample main processing source file | Change |

6.4.2 Sample Program for the Communications Evaluation Board

This sample program is built on the EC-1 peripheral driver package.

Table 6.3 to Table 6.5 list the files which have been added for the communications evaluation board sample program.

(1) `./Include`: Include Files

Table 6.3 Configuration of Files for the Communications Evaluation Board Sample Program (1)

| Directory | File | Description | Classification |
|-------------------|-------------------------|---|----------------|
| <code>spi/</code> | <code>r_spi.h</code> | SPI driver prototype declaration | Addition |
| | <code>r_spiesc.h</code> | SPI ESC driver prototype declaration | Addition |
| | <code>spiesc.h</code> | Definitions of structures and unions for access to the SPI and its extended registers | Addition |

(2) `./Library`: Library

Table 6.4 Configuration of Files for the Communications Evaluation Board Sample Program (2)

| Directory | File | Description | Classification |
|------------------|--------------------------|------------------------|----------------|
| <code>../</code> | <code>libspiesc.a</code> | SPI ESC access library | Addition |

(3) ./Source/Project/spiesc_sample: Sample Application

Table 6.5 Configuration of Files for the Communications Evaluation Board Sample Program (3)

| Directory | File | Description | Classification |
|-----------|-----------------------------|--|----------------|
| ../ | main.c | ESC slave sample main processing source file | Addition |
| | board_communication.c | EC-1 communications evaluation board configuration source file | Addition |
| IAR/ | EC_1_spiesc_serial_boot.eww | IAR project file | Addition |
| | EC_1_spiesc_serial_boot.ewd | IAR project related file | Addition |
| | EC_1_spiesc_serial_boot.ewp | IAR project related file | Addition |
| | EC_1_spiesc_ram_debug.eww | IAR project file | Addition |
| | EC_1_spiesc_ram_debug.ewd | IAR project related file | Addition |
| | EC_1_spiesc_ram_debug.ewp | IAR project related file | Addition |

6.5 Tutorial

The following describes the procedures for checking the operation of the sample program.

6.5.1 Preparation

(1) Connecting the boards

Connect the SPI signals and interrupt signals of the EC-1 remote I/O board and the EC-1 communications evaluation board.

For the wiring for connection, see section 3.1, Example of the Hardware Configuration in this application note.

(2) Starting the communications evaluation board sample program

Start the ESC SPI connection library sample program for the EC-1 communications evaluation board.

For the IAR project files of the sample program, see section 6.4.2, Sample Program for the Communications Evaluation Board in this application note.

For the overall sample program, refer to *EC-1 Application Note: Peripheral Drivers*.

(3) Starting the remote I/O board sample program

Start the sample program for the EC-1 remote I/O board. For instructions on the sample program from generation to startup, refer to *EC-1 Application Note: Remote I/O*.

Remark 1. The program must be started with the EC-1 communications evaluation board (SPI slave) first, then the EC-1 remote I/O board (SPI master), in that order.

(4) Preparation for EtherCAT communications

For the preparation for EtherCAT communications, refer to *EC-1 Application Note: Remote I/O*.

6.5.2 Connection with TwinCAT

For reading and writing data from and to the remote I/O board from startup of TwinCAT, refer to *EC-1 Application Note: Remote I/O*.

Remark 2. If the communications status LED on the communications evaluation board lights up in red and the device cannot be found with I/O device scanning by TwinCAT, SPI communications may not be operating properly.

In such cases, re-check the wiring between the EC-1 communications evaluation board and the EC-1 remote I/O board for correctness.

If there is no problem with the wiring but this still does not solve the problem, try to solve it by retarding the SPI transfer rate of the EC-1 remote I/O board (SPI master).

7. Application Example: Migration from ET1100

7.1 Overview

This section describes the items which are required for porting the EtherCAT slave controller from an ET1100 from Beckhoff Automation GmbH to an EC-1 incorporating the ESC SPI connection library.

Figure 7.1 shows an example of the configuration of the ET1100 and EC-1 as EtherCAT slaves.

Table 7.1 gives a comparison of the ESC specifications for the ET1100 and EC-1.

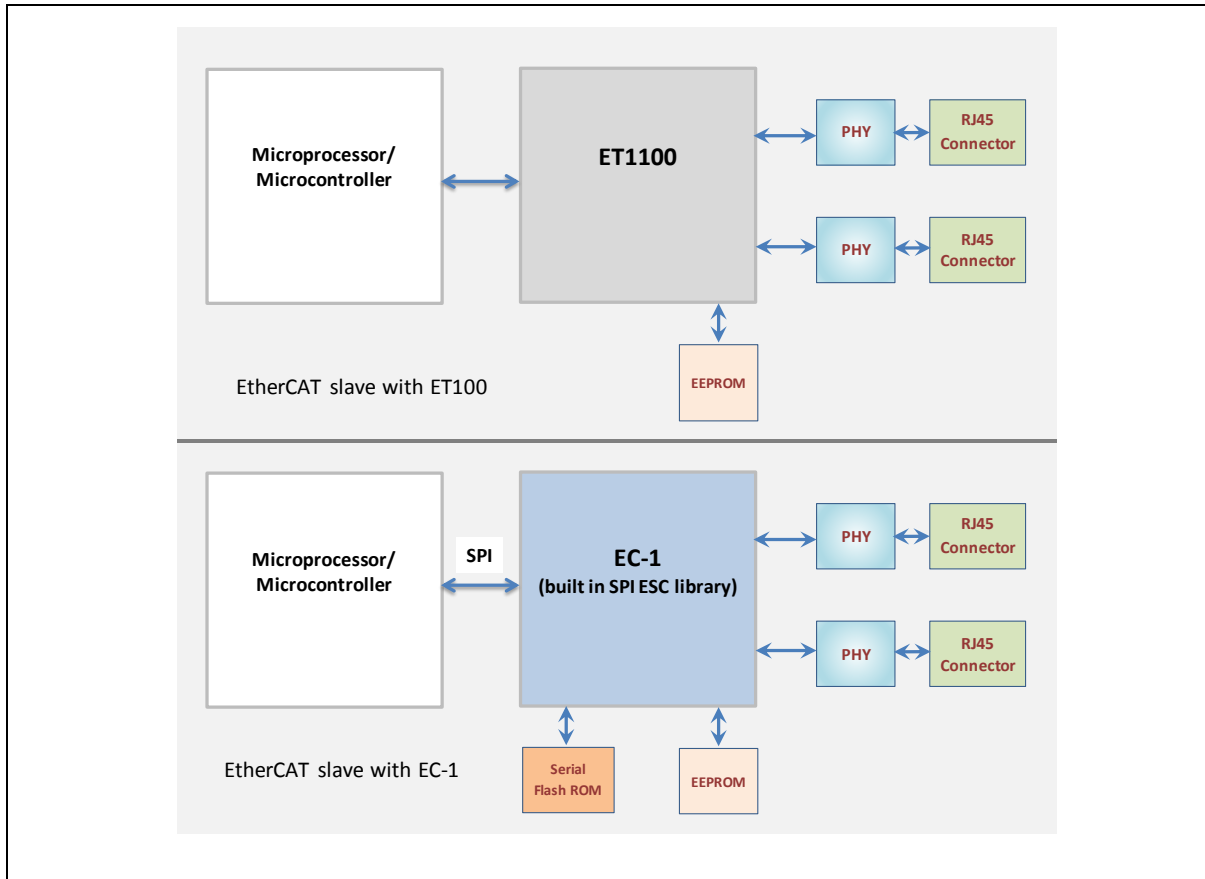


Figure 7.1 Example Configuration of EtherCAT Slaves

Table 7.1 Comparison of the ESC Specifications

| Features | | ET1100 | EC-1 |
|------------------------------|------------|--------------|----------------------------|
| Number of boards | | 2 to 4 | 2 |
| FMMU | | 8 | 8 |
| Sync Manager | | 8 | 8 |
| Process data RAM (Kbyte) | | 8 | 8 |
| Distributed clock | | 64 bits | 64 bits |
| EBUS | | Yes (0 to 4) | No |
| Process data interface (PDI) | Digital IO | Yes | No |
| | SPI I/F | Yes | Supported in this library. |
| | Host I/F | Yes | No |

7.2 Hardware Related Items

Connecting an external MCU with the EC-1 requires a total of seven signals, consisting of the four signals for SPI communications and three external interrupt input signals.

Table 7.2 lists the signals required for the connection and the pin names on the EC-1 for connection of these signals. It also lists the connector pins for connection with the EC-1 communications evaluation board.

For the external interrupt input pins, select the detection by the pins of interrupts as rising edges of the signals.

Table 7.3 lists the SPI specifications to make on the external MCU for connection with the EC-1. Check this table for reference in setting the SPI module of the external MCU.

Table 7.2 Signals for Connection with the EC-1

| No. | External MCU | | Destination to which the EC-1 is Connected | Destination for connection with the EC-1 Communications Evaluation Board | |
|-----|----------------------------|--------|--|--|---------------------------------------|
| | Signal Name | Active | Pin Name | Pin Name | Connector Pin Number Test Pin Name |
| 1 | SPI slave selection output | H | SSL00, SSL10 | P75/SSL00 | J7 #4 |
| 2 | MOSI output | --- | MOSI0, MOSI1 | PA1/MOSI0 | J7 #3 |
| 3 | MISO input | --- | MISO0, MISO1 | PA0/MISO0 | J7 #2 |
| 4 | SCK output | --- | RSPCK0, RSPCK1 | P77/SPCK0 | J7 #1 |
| 5 | External interrupt input 1 | H | CATIRQ | CATIRQ | CATIRQ |
| 6 | External interrupt input 2 | H | CATSYNC0 | CATSYNC0 | CATSYNC0 |
| 7 | External interrupt input 3 | H | CATSYNC1 | CATSYNC1 | CATSYNC1 |

Table 7.3 SPI Specifications for Connection with the EC-1

| Item | Description |
|--------------------------|---|
| Method of communications | Four-wire SPI |
| Transfer mode | Master mode |
| Method of communications | Full duplex |
| Signal sense | Slave select signal: Active low Serial clock signal: Low in the idle state |
| Transfer frame | 16 bits × 3 words |
| Format | MSB first |
| Data sampling timing | Falling edges of the serial clock signal |
| Maximum transfer rate | 18.75 Mbps |
| CPHA and CPOL settings | CPHA = 1, CPOL = 0 |

7.3 Software Related Items

The following describes the procedures for access to the ESC and extended SPI registers by the external MCU through the SPI interface.

For the specifications of the SPI interface, see section 4, SPI Interface in this application note.

7.3.1 Sequence for Reading the Registers

Figure 7.2 shows the sequence for reading the ESC and extended registers of the SPI by using the RD command.

The results of reading can be acquired immediately in response to issuing each RD command.

1, 2, or 4 bytes of data can be read at a time.

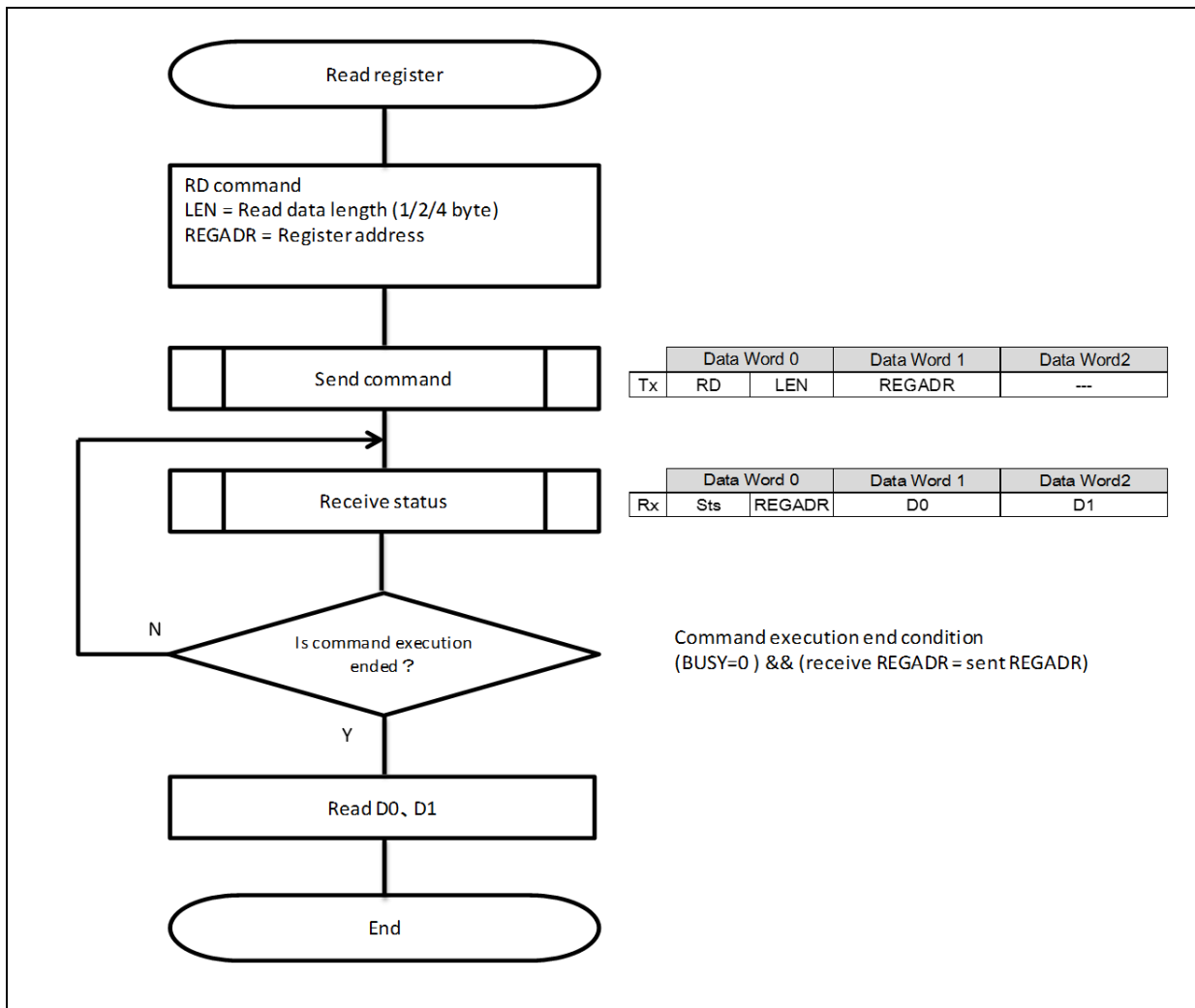


Figure 7.2 Sequence for Reading the Registers

7.3.2 Sequence for Writing to the Registers

Figure 7.3 shows the sequence for writing to the ESC and extended registers of the SPI by using the WR command.

Data to be written can be reflected immediately in response to issuing each WR command.

1 or 2 bytes of data can be written at a time. Note that 4-byte writing is not allowed.

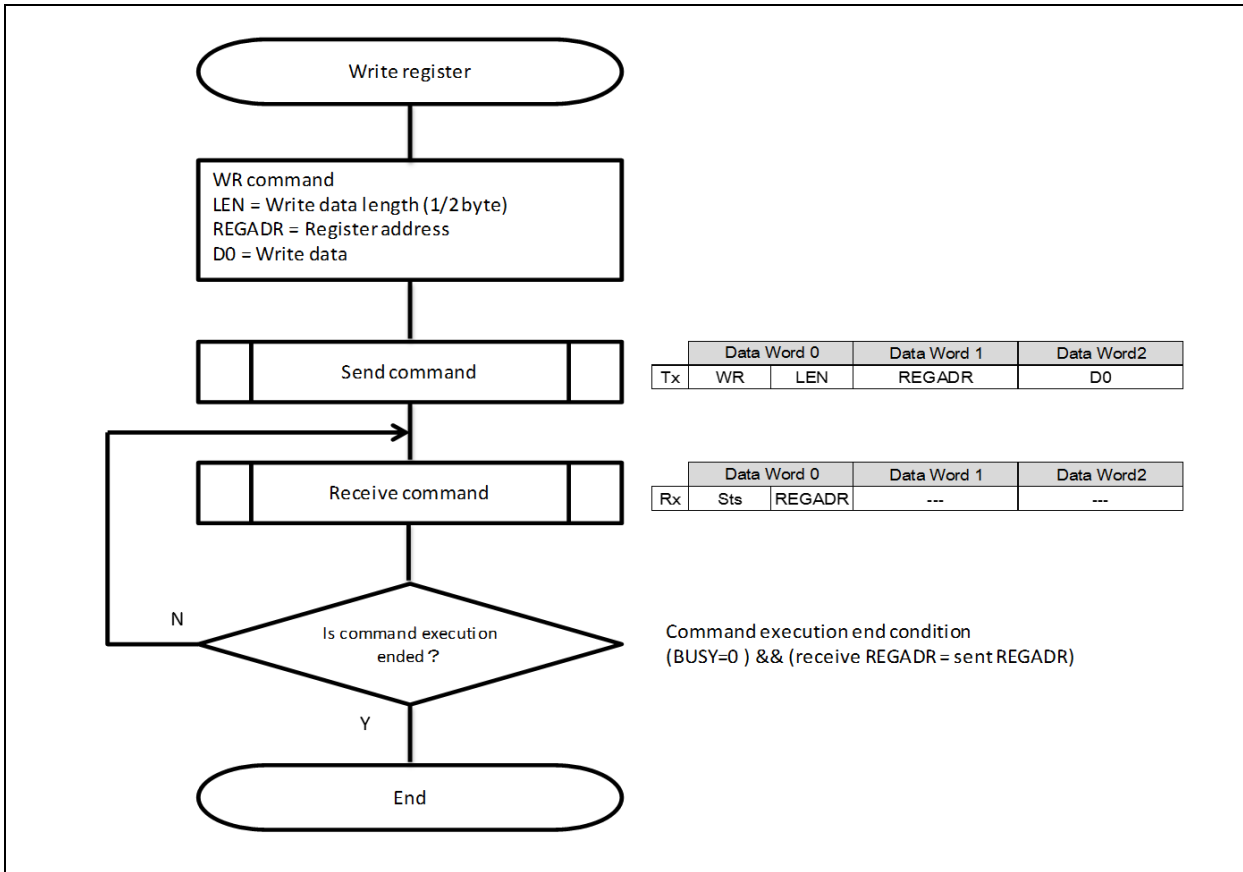


Figure 7.3 Sequence of Writing to the Registers

7.3.3 Sequence for Sequential Reading from the Process Data RAM

Figure 7.4 shows the sequence for sequential reading of data from the process data RAM of the ESC.

The SQRD and DTRD commands are used.

First, the address where the process data RAM starts and the data length (up to 255 bytes) to be read by using the SQRD command are specified.

Upon receiving the SQRD command, the EC-1 stores data in a sequential data register of the extended registers by reading the specified area of the process data RAM.

After that, issue a single DTRD command to dummy read the sequential data register. The data in the sequential data register are then acquired. In receiving data by using the DTRD command, confirming the completion of execution of the command is not required, so the continuous reception of data is possible.

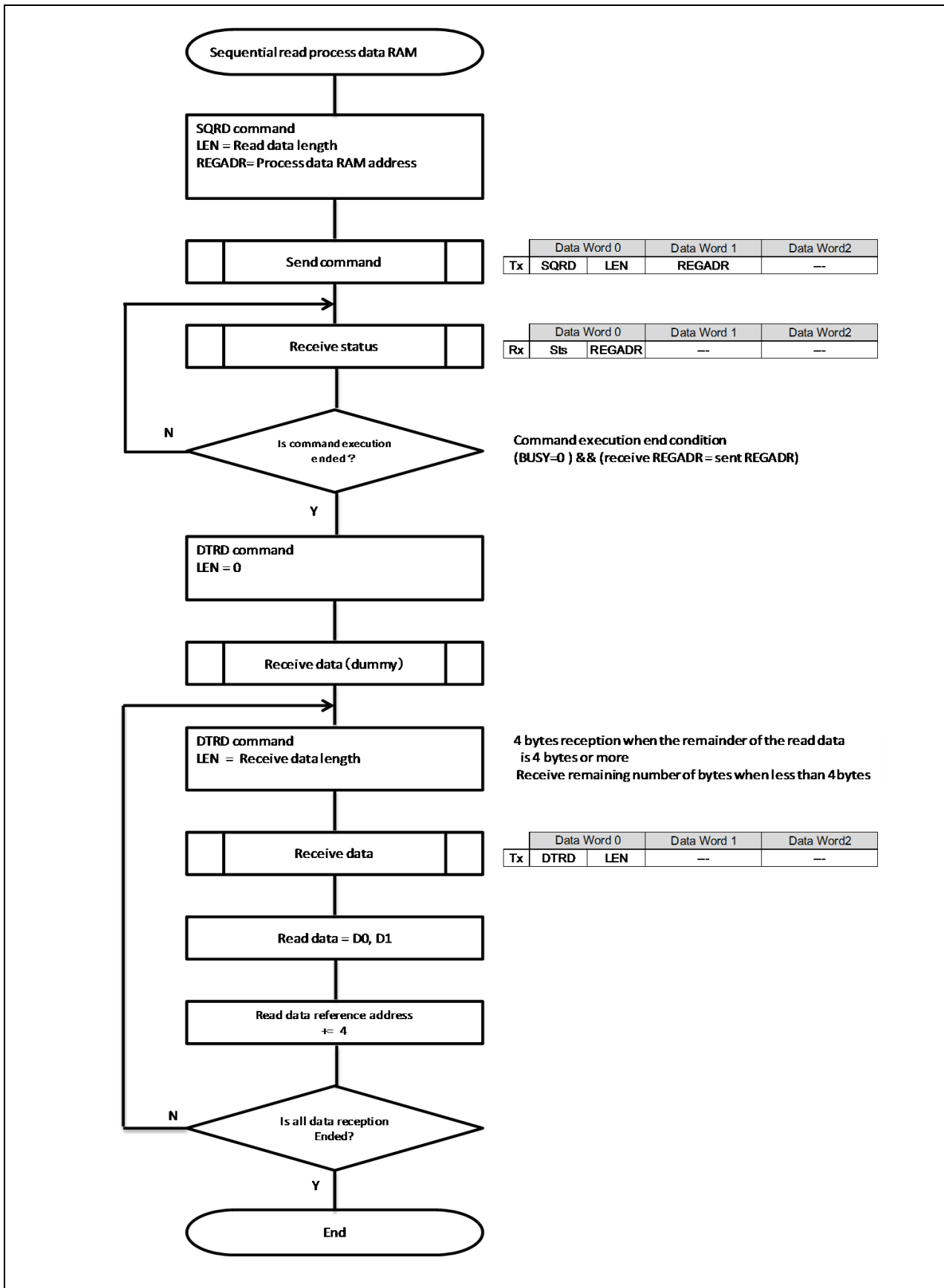


Figure 7.4 Sequence for Sequential Reading from the Process Data RAM

7.3.4 Sequence for Sequential Writing to the Process Data RAM

Figure 7.5 shows the sequence for sequential writing of data to the process data RAM of the ESC.

The DTWR and SQWR commands are used.

First, all data to be written by using the DTWR command are stored in a sequential data register. In transmitting data by using the DTWR command, confirming the completion of execution of the command is not required, so the continuous transmission of data is possible.

After that, the address where the process data RAM starts and the data length (up to 255 bytes) are specified by using the SQWR command.

Upon receiving the SQWR command, the EC-1 writes data to the specified area of the process data RAM in a sequential data register of the extended registers.

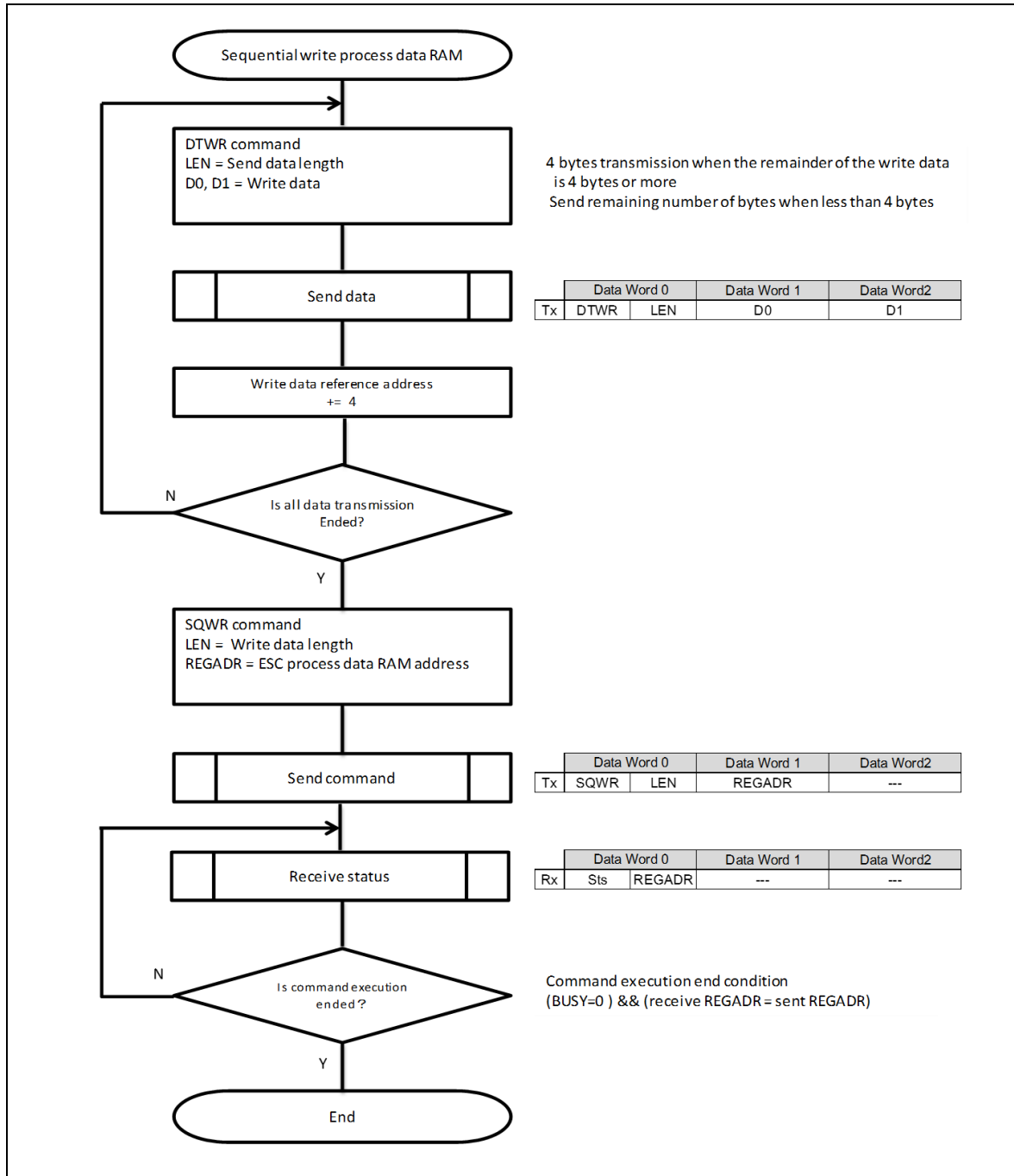


Figure 7.5 Sequence for Sequential Writing to the Process Data RAM

7.3.5 Extended Registers: Sequence for Confirming Startup

Before running the EtherCAT protocol, confirm that SPI communications with the EC-1 are trouble-free.

Figure 7.6 shows the sequence for confirming startup.

4 bytes of the device ID register (0x3000) among the extended registers are read by using the RD command. If 'EC-1' (0x45435E31) is read, communications are proceeding normally.

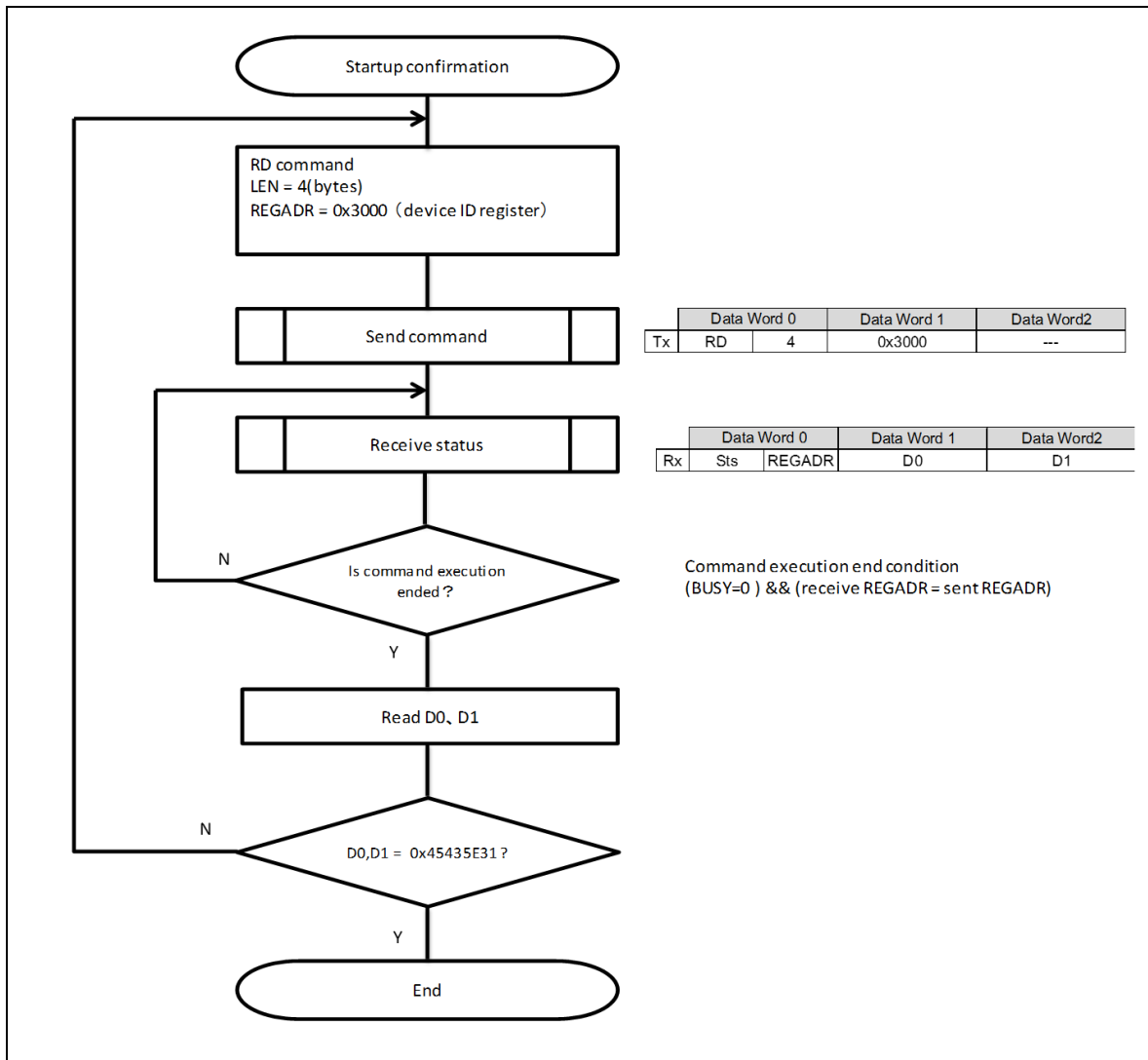


Figure 7.6 Sequence for Confirming Startup

7.3.6 Extended Registers: Software Reset Sequence

Figure 7.7 shows the sequence for a software reset by the EC-1.

The WR command is used to write 2 bytes of code (0x0080) to apply a software reset through the control register (0x3008) among the extended registers.

The EC-1 executes a software reset immediately upon reception of the command.

SPI communications do not proceed while the EC-1 is being restarted. Wait and then execute the startup confirmation sequence.

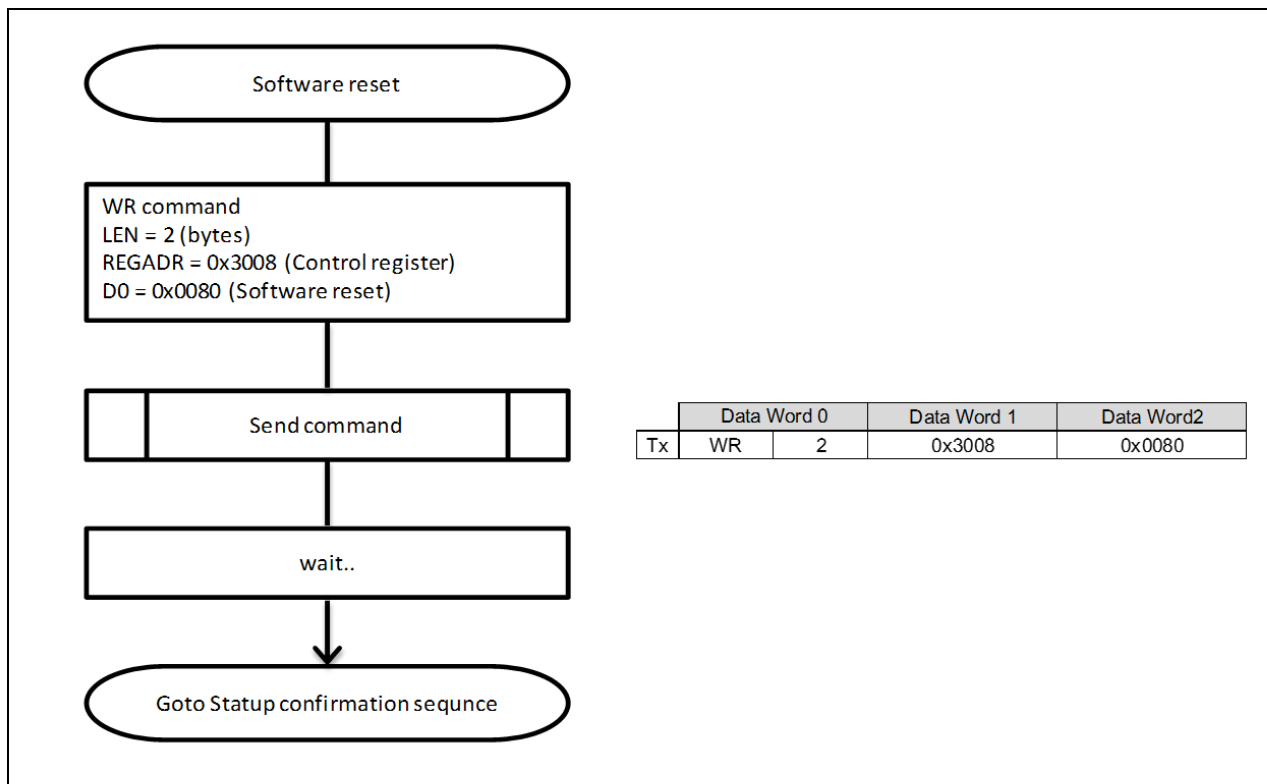


Figure 7.7 Software Reset Sequence

7.4 Items Related to Slave Stack Code

7.4.1 Hardware Access Files

The remote I/O board sample program includes the hardware access files for access to the ESC through the SPI.

Table 7.4 lists the hardware access files for the EC-1.

The hardware access files for the ET1100 can be replaced by the files in the table.

Table 7.4 Hardware Access Files for the EC-1

| Directory | File | Description |
|-----------|-------------|---|
| SSC/ | renesashw.h | SPI interface ESC access driver prototype declaration |
| | renesashw.c | SPI interface ESC access driver |

The hardware access files for the EC-1 are designed to run on the hardware of the EC-1, so you need to change them to suit the hardware of the external MCU you will be using.

Table 7.5 lists the functions which require changes and the hardware modules used.

Table 7.5 Functions Requiring Changes

| Function Name | Outline of the Role | Modules Used |
|-----------------|---------------------------|----------------------|
| HW_Init | Hardware initialization | Timer module |
| HW_GetTimer | Acquiring the timer value | Timer module |
| HW_ClearTimer | Clearing the timer value | Timer module |
| DISABLE_ESC_INT | Enabling interrupts | Interrupt controller |
| ENABLE_ESC_INT | Disabling interrupts | Interrupt controller |
| rspi_xfer | SPI transfer | SPI module |

In the hardware access files for the EC-1, the details of processing by the read and write functions for the ESC registers are the same whether execution is from the interrupt handler or the main function.

Table 7.6 lists the read and write functions to be executed from the interrupt handler.

Change them as required if interrupts must be taken into account.

Table 7.6 Functions Requiring Consideration

| Function Name | Outline of the Functions |
|---------------------------|--|
| HW_GetALEventRegister_Isr | Reading the AL event register |
| HW_EscReadIsr | Sequential reading of bytes from the ESC |
| HW_EscReadDWordIsr | Reading of 32 bits from the ESC |
| HW_EscReadWordIsr | Reading of 16 bits from the ESC |
| HW_EscReadByteIsr | Reading of 8 bits from the ESC |
| HW_EscWriteIsr | Sequential writing of bytes to the ESC |
| HW_EscWriteDWordIsr | Writing of 32 bits to the ESC |
| HW_EscWriteWordIsr | Writing of 16 bits to the ESC |
| HW_EscWriteByteIsr | Writing of 8 bits to the ESC |

Table 7.7 lists the definitions of the hardware configuration included in ecat_def.h. The hardware access files for the EC-1 presuppose these definitions.

Change the stack code as required.

Table 7.7 Definitions of the Hardware Configuration

| Definition Name | Value |
|-------------------------|------------|
| CONTROLLER_16BIT | 0 |
| CONTROLLER_32BIT | 0 |
| ESC_16BIT_ACCESS | 0 |
| ESC_32BIT_ACCESS | 0 |
| MBX_16BIT_ACCESS | 0 |
| MEM_ADDR | UINT8 |
| GET_MEM_SIZE (ByteSize) | (ByteSize) |

7.4.2 Changing the ESI File

The statements dependent on the hardware configuration of the ESC must be changed for the EC-1.

Table 7.8 lists the point to be changed and its value in the ESI file.

Table 7.8 Point to be Changed in the ESI File

| Type | Element | Data | Description |
|--------|------------|------------------|---|
| Eeprom | ConfigData | 800E42EE10270000 | Value of the configuration area of the EEPROM |

7.4.3 Adding the Startup Confirmation Sequence

Check that SPI communications with the EC-1 can proceed before operating the slave stack.

For the startup confirmation sequence, see section 7.3.5, Extended Registers: Sequence for Confirming Startup.

8. Documents for Reference

8.1 Related Documents

The documents related to descriptions in this manual are listed below.

Download the latest version from the Renesas Electronics website.

- EC-1 User's Manual: Hardware
- EC-1 Application Note: Communications Board
- EC-1 Application Note: Remote I/O
- EC-1 Application Note: Peripheral Drivers

8.2 Relationship between This Document and Related Documents

The following are supplementary notes on the relationship between this document and related documents.

- EC-1 User's Manual: Hardware
This manual describes the hardware specifications of the EC-1.
- EC-1 Application Note: Communications Board
This manual describes the specifications of the communications board on which the sample program runs.
- EC-1 Application Note: Remote I/O
This manual describes how to use the remote I/O sample program.
- EC-1 Application Note: Peripheral Drivers
This manual describes the drivers and sample program which can be used with the EC-1.

Website and Support

Renesas Electronics website

<http://www.renesas.com/>

Inquiries

<http://www.renesas.com/inquiry>

All trademarks and registered trademarks are the property of their respective owners.

Revision History

| Rev. | Date | Description | |
|------|--------------|-------------|-----------------------|
| | | Page | Summary |
| 1.00 | Apr 14, 2017 | — | First edition issued. |

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

- The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

The state of the product is undefined at the moment when power is supplied.

- The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

Access to reserved addresses is prohibited.

- The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

- When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

- The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

- ARM and Cortex are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.
- Ethernet is a registered trademark of Fuji Xerox Co., Ltd.
- IEEE is a registered trademark of the Institute of Electrical and Electronics Engineers Inc
- TRON is an acronym for "The Real-time Operation system Nucleus.
- ITRON is an acronym for "Industrial TRON.
- μ ITRON is an acronym for "Micro Industrial TRON.
- TRON, ITRON, and μ ITRON do not refer to any specific product or products.
- EtherCAT® is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
- Additionally all product names and service names in this document are a trademark or a registered trademark which belongs to the respective owners.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
 2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
 3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
 4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
 5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.
"Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.
"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
 6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
 7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
 8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
 9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
 10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
 11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
 12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.
- (Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.
(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.



SALES OFFICES

Renesas Electronics Corporation

<http://www.renesas.com>

Refer to "<http://www.renesas.com/>" for the latest and detailed information.

Renesas Electronics America Inc.

2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited

9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited

Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH

Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.

Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.

Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited

Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.

13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.

80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.

Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.

No.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.

12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141