SPI Drivers

For

AVR Microcontrollers

Nti Team

Software Requirement Specification Document

Table of Contents

Version: 1st **Date: 26/10/2023**

1. Scope of Document

This document specifies requirements on the module SPI Driver.

1.1 Constraints

First scope for specification of requirements on basic software modules is systems which are not safety relevant. For this reason safety requirements are assigned to medium priority.

2. Requirements Structure

Each module specific chapter contains a short functional description of the Basic Software Module. Requirements of the same kind within each chapter are grouped under the following headlines (where applicable):

Functional Requirements: -

Configuration

Interface.

Registers of SPI.

Non-Functional Requirements:-

Timing Requirements.

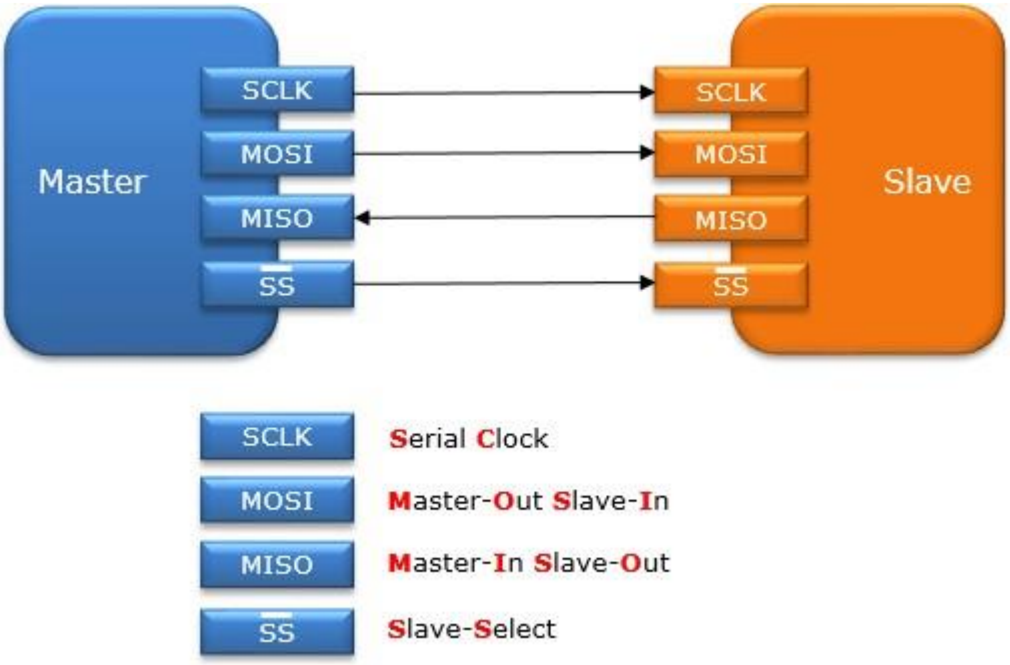
Resource Usage.

Usability.

3. Functional Overview

3.1 SPI Driver, common functionality:

A SPI bus is a master slave multi node bus system, the master sets a Chip Select (CS) to select a slave for data communication. The SPI (Serial Peripheral Interface) has a 4-wire synchronous serial interface. Data communication is enabled with a Chip Select wire (CS). Data is transmitted with a 3-wire interface consisting of wires for serial data input (MOSI), serial data output (MISO) and Serial Clock (SCK). The following SPI module provides data based read, write and transfer access to different devices on SPI busses. A SPI channel represents data elements.



4. Requirement Specification

5.1 Functional Requirements

[SRS_REQ_1] The SPI Driver shall support a function to initialise the SPI by choose the master slave select, clock rate, clock phase, clock polarity, interrupt enable and the order of data.

[SRS_REQ_2] The driver shall be compatible with all AVR microcontrollers.

[SRS_REQ_3] The SPI Driver shall support a specific basic static configuration.

[SRS_REQ_4] The SPI Handler/Driver shall handle the chip

[SRS_REQ_5] The ADC Driver shall support functions that control the interrupt to make it enable or disable.

[SRS_REQ_6] The SPI Handler/Driver shall support the communication to daisy chained HW devices. During the transfer to/from the HW devices, the CS signal shall Remain asserted.

[SRS_REQ_7] The SPI Handler/Driver shall have a scalable functionality to fit the needs of the ECU.

5.2 Non-functional requirements

[SRS_REQ_8] The driver shall be easy to use and understand.

[SRS_REQ_9] The driver shall be well-documented.

[SRS_REQ_10] The driver shall be efficient and use minimal resources.

[SRS_REQ_11] The driver shall be reliable and robust.

[SRS_REQ_12] In addition to the above requirements, the SPI driver shouldalso meet the following non-technical requirements

[SRS_REQ_13] The driver should be open source and freely available to use.

[SRS_REQ_14] The driver should be actively maintained and supported by the community.

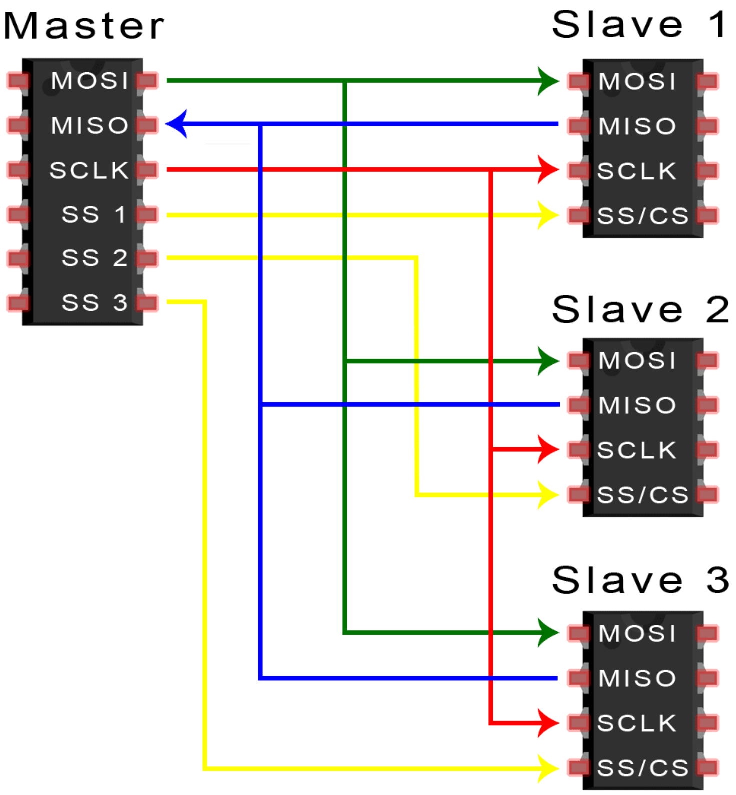
[SRS_REQ_15] The driver should be well-tested and documented.

.

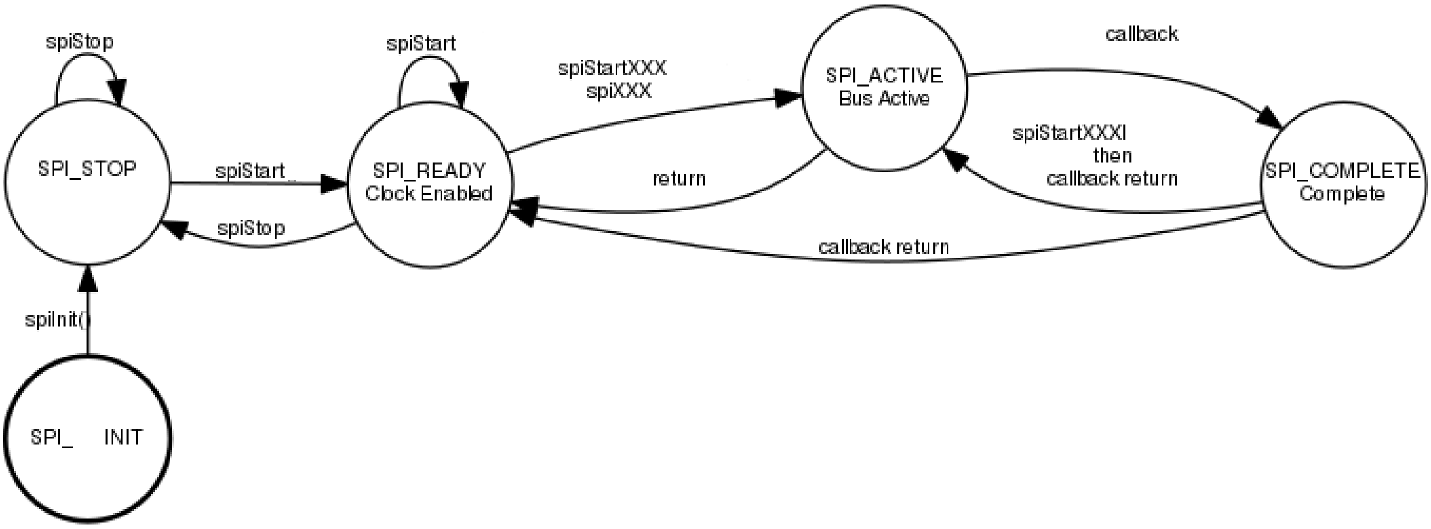
5. Daisy chain

With this scheme all data sent by the master is shifted into all devices and all data sent from each device is shifted out to the next (shown by red dotted arrow). For this scheme to work you have to make sure that each slave uses the clock in the same way and you have to get the right number of bits, so there is more work to do in software.

The advantage of the Daisy-Chain method is that you save a chip select signal for each slave SPI device.



6. State machine



7. Acceptance Criteria

The SPI driver shall be accepted when it meets the following criteria:

- The driver shall compile and run without errors on all AVR microcontrollers.
- The driver shall pass all unit tests.
- The driver shall pass all integration tests.
- The driver shall pass all system tests.

8. References

1. Developers of NTI team.
2. AVR Microcontroller Datasheets.