Serial Interfaces

Lecture 11

Josh Brake

Harvey Mudd College

Outline

- Serial Interfaces Overview
  - Advantages over parallel
  - Major considerations
  - Overview of protocols
- Serial Peripheral Interface
  - Description
  - MCU configuration
- DS1722 SPI temperature sensor
  - Datasheet overview
- CMSIS

Learning Objectives

By the end of this lecture you should be able to…

- See how the SPI peripheral works on the STM32L432KC
- See how to verify the output using a logic analyzer

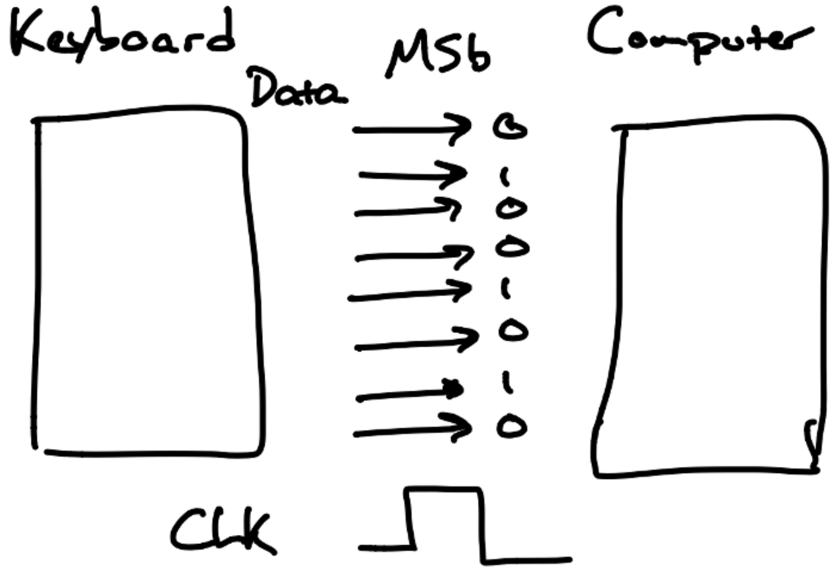
Serial Interfaces Overview

Motivation

How can we interface a peripheral?

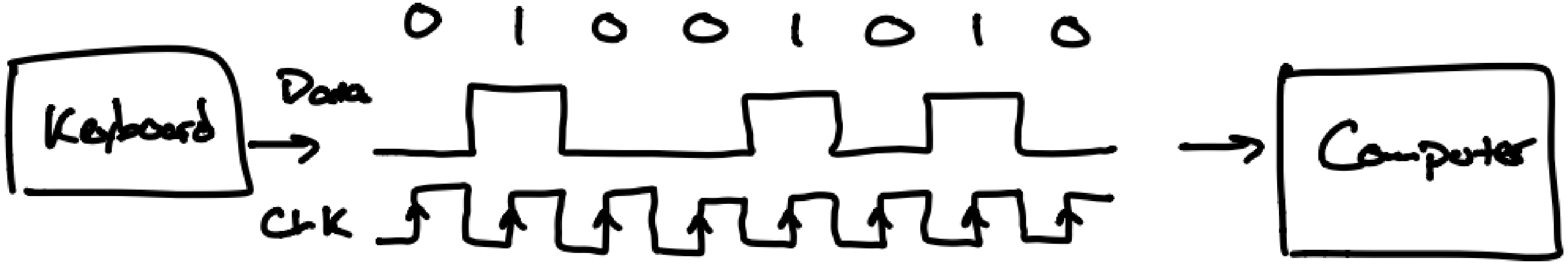
Imagine transmitting a character on a keyboard.

Capital J in ASCII is $74_{10} = 01001010_2$



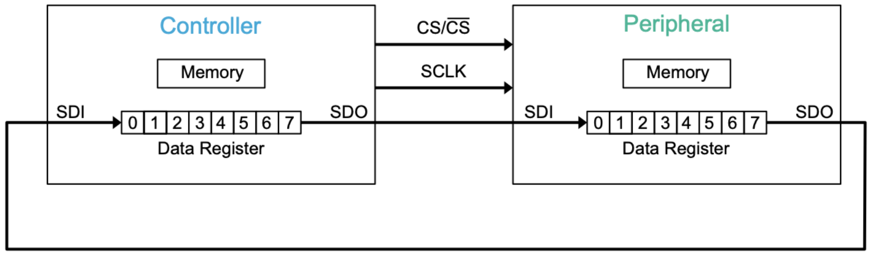
What if we repackage data in a stream?

- Multiplexing in time
- To send N bits, we only need 2 lines (CLK + Data) instead of 9
- Price we pay is time – but often worth it.



Serial Peripheral Interface (SPI)

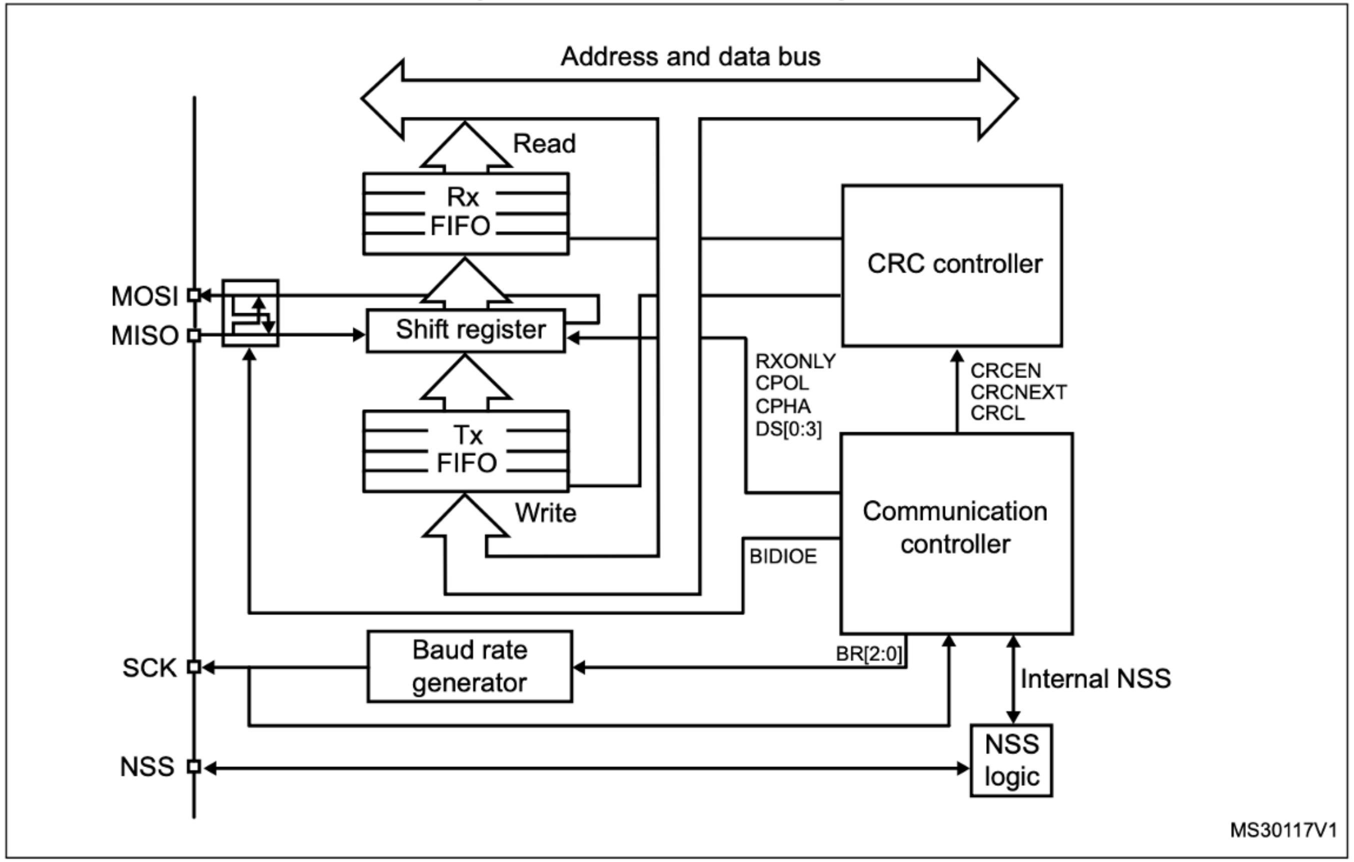
- Developed in the mid-1980s by Motorola
- Used to interface with many peripherals like memory (SD cards, flash), displays, sensors (accelerometers, gyroscopes, temperature sensors, ADCs and DACs).
- Four-wire, synchronous serial bus



SCLK: Serial clock
MOSI: Master Out Slave In
MISO: Master In Slave Out
CE/CS/nCE/nCS: Chip select/enable

SPI Block Diagram on STM32L432KC

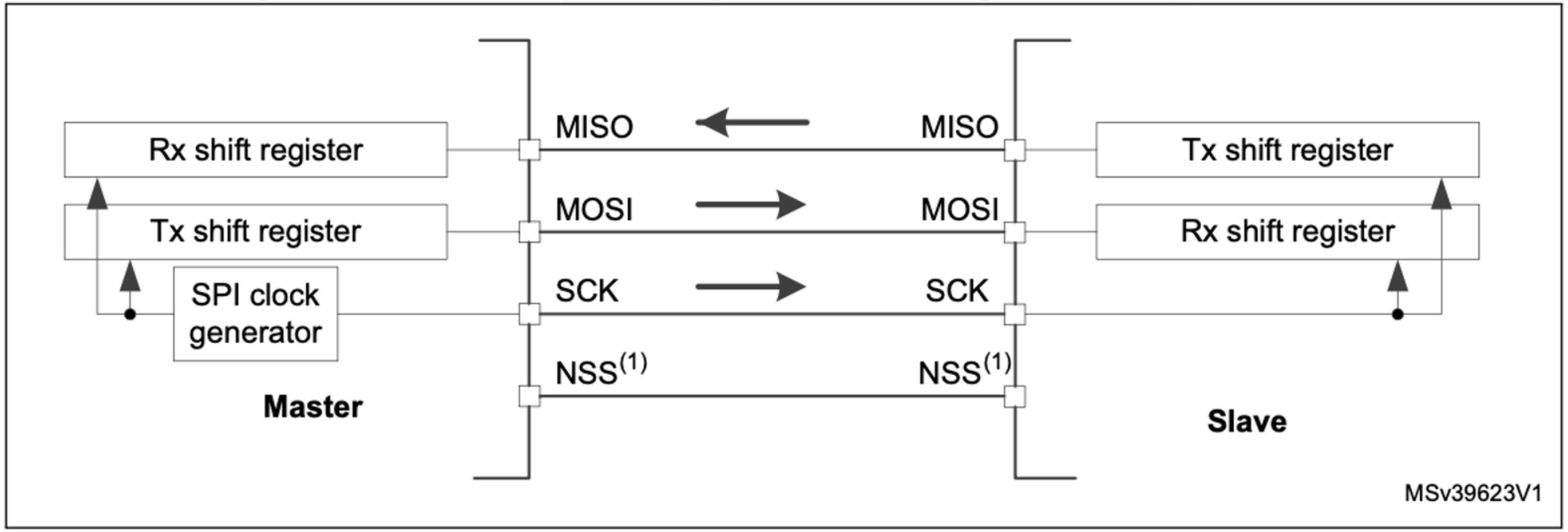
Figure 419. SPI block diagram



RM0394 p. 1305

SPI Block Diagram

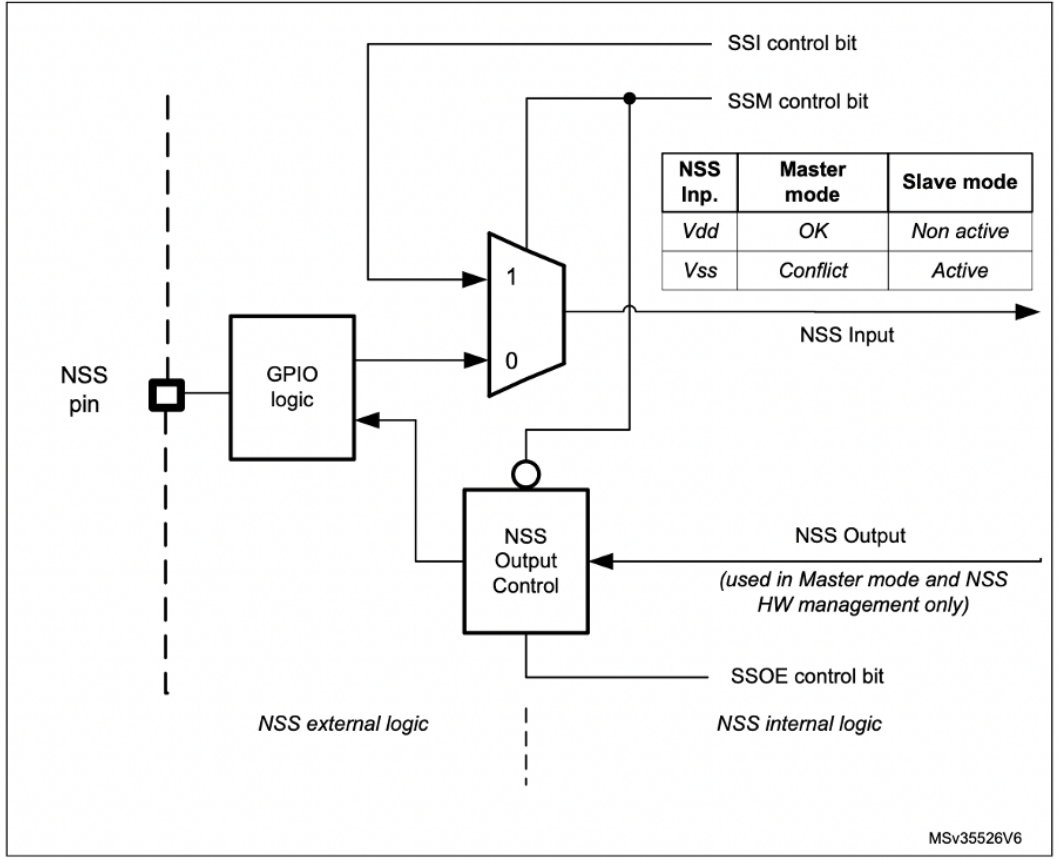
Figure 420. Full-duplex single master/ single slave application



RM0394 p. 1306

SPI Hardware NSS Management

Figure 425. Hardware/software slave select management

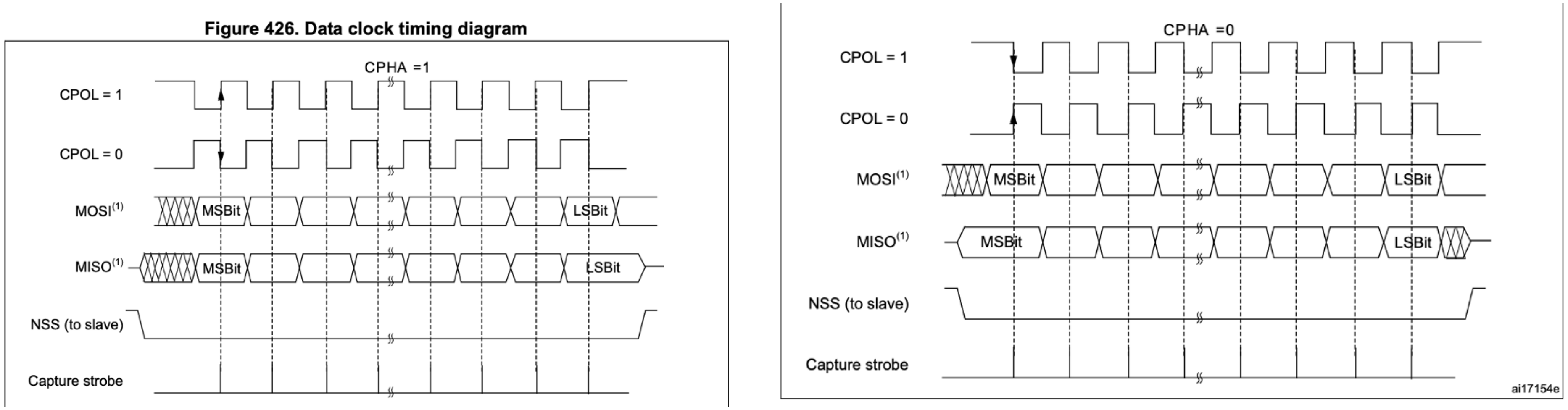


NSS Inp.	Master mode	Slave mode
Vdd	*OK*	*Non active*
Vss	*Conflict*	*Active*

RM0394 p. 1311

Example SPI Traces

Figure 426. Data clock timing diagram



RM0394. p. 1312

SPI Data Frame Sizes

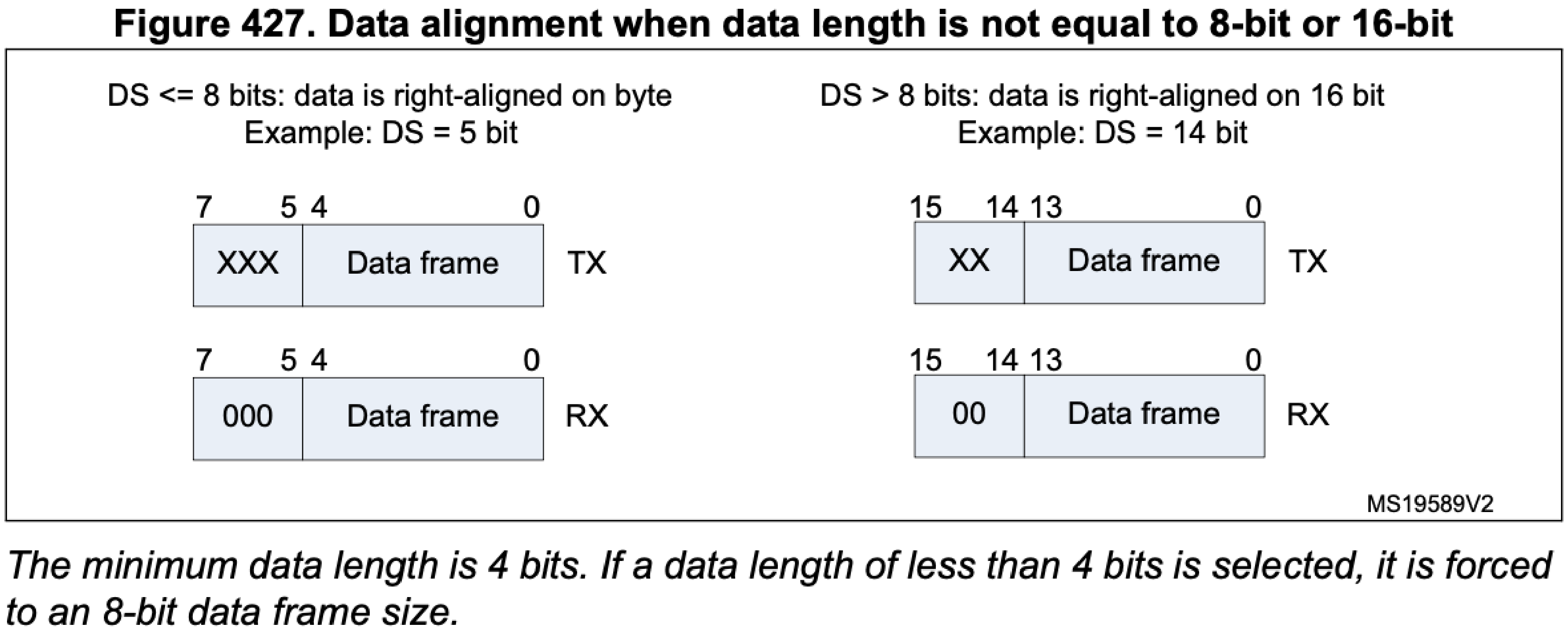
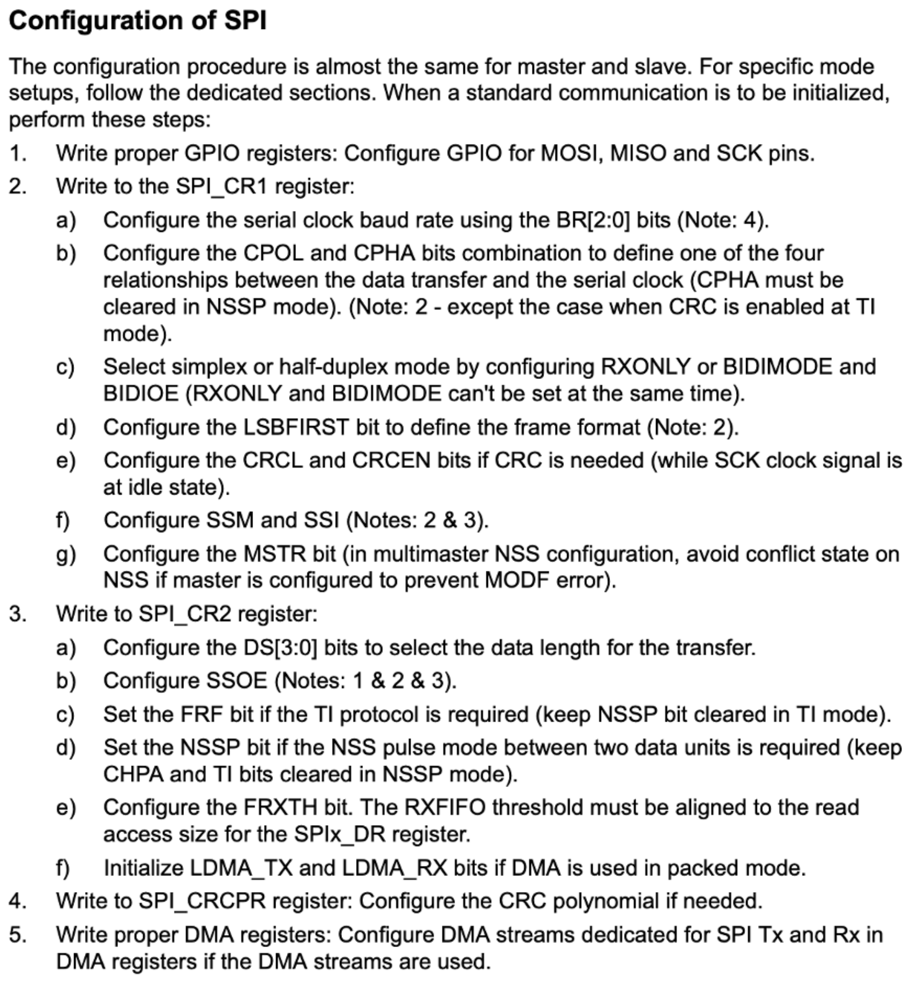


Figure 427. Data alignment when data length is not equal to 8-bit or 16-bit

The minimum data length is 4 bits. If a data length of less than 4 bits is selected, it is forced to an 8-bit data frame size.

SPI Configuration



Configuration of SPI

The configuration procedure is almost the same for master and slave. For specific mode setups, follow the dedicated sections. When a standard communication is to be initialized, perform these steps:

1. Write proper GPIO registers: Configure GPIO for MOSI, MISO and SCK pins.
2. Write to the SPI_CR1 register:
   a) Configure the serial clock baud rate using the BR[2:0] bits (Note: 4).
   b) Configure the CPOL and CPHA bits combination to define one of the four relationships between the data transfer and the serial clock (CPHA must be cleared in NSSP mode). (Note: 2 - except the case when CRC is enabled at TI mode).
   c) Select simplex or half-duplex mode by configuring RXONLY or BIDIMODE and BIDIOE (RXONLY and BIDIMODE can't be set at the same time).
   d) Configure the LSBFIRST bit to define the frame format (Note: 2).
   e) Configure the CRCL and CRCEN bits if CRC is needed (while SCK clock signal is at idle state).
   f) Configure SSM and SSI (Notes: 2 & 3).
   g) Configure the MSTR bit (in multimaster NSS configuration, avoid conflict state on NSS if master is configured to prevent MODF error).
3. Write to SPI_CR2 register:
   a) Configure the DS[3:0] bits to select the data length for the transfer.
   b) Configure SSOE (Notes: 1 & 2 & 3).
   c) Set the FRF bit if the TI protocol is required (keep NSSP bit cleared in TI mode).
   d) Set the NSSP bit if the NSS pulse mode between two data units is required (keep CHPA and TI bits cleared in NSSP mode).
   e) Configure the FRXTH bit. The RXFIFO threshold must be aligned to the read access size for the SPIx_DR register.
   f) Initialize LDMA_TX and LDMA_RX bits if DMA is used in packed mode.
4. Write to SPI_CRCPR register: Configure the CRC polynomial if needed.
5. Write proper DMA registers: Configure DMA streams dedicated for SPI Tx and Rx in DMA registers if the DMA streams are used.

RM0394 p. 1313

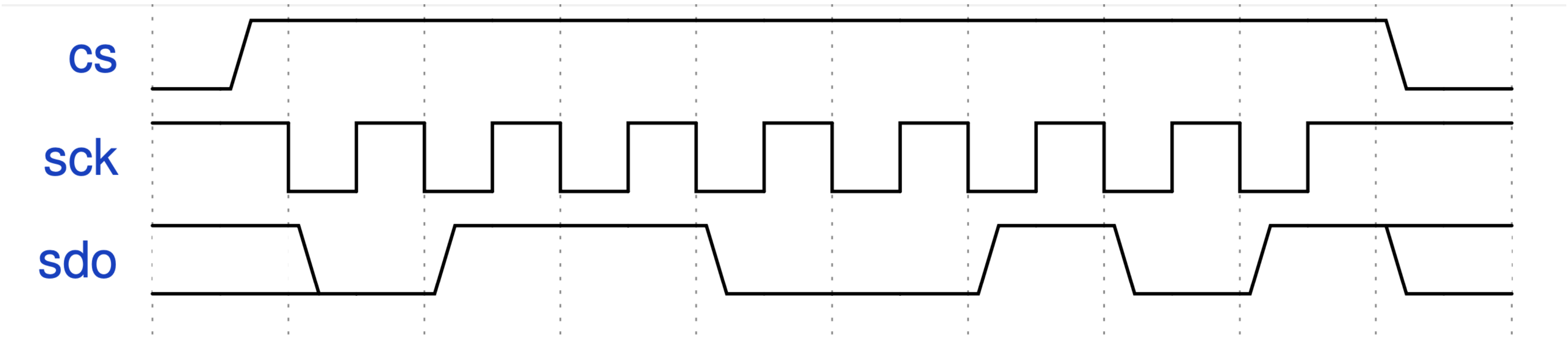
SPI Clock Polarity and Phase

- Clock polarity (CPOL) refers to the state of the clock line at idle
  - 0: clock is low when idle
  - 1: clock is high when idle
- Clock phase (CPHA) refers to when data is sampled vs. when new data is shifted out
  - 0: the first clock transition is the first data capture edge
  - 1: the second clock transition is the first data capture edge
- The clock transition (rising or falling) depends on the clock polarity
- 4 combinations or modes (CPOL,CPHA) = (0,0), (0,1), (1,0), (1,1)
- Must pay attention to match this mode to the peripheral!

SPI Example 1

Determine the following properties for the waveform below:

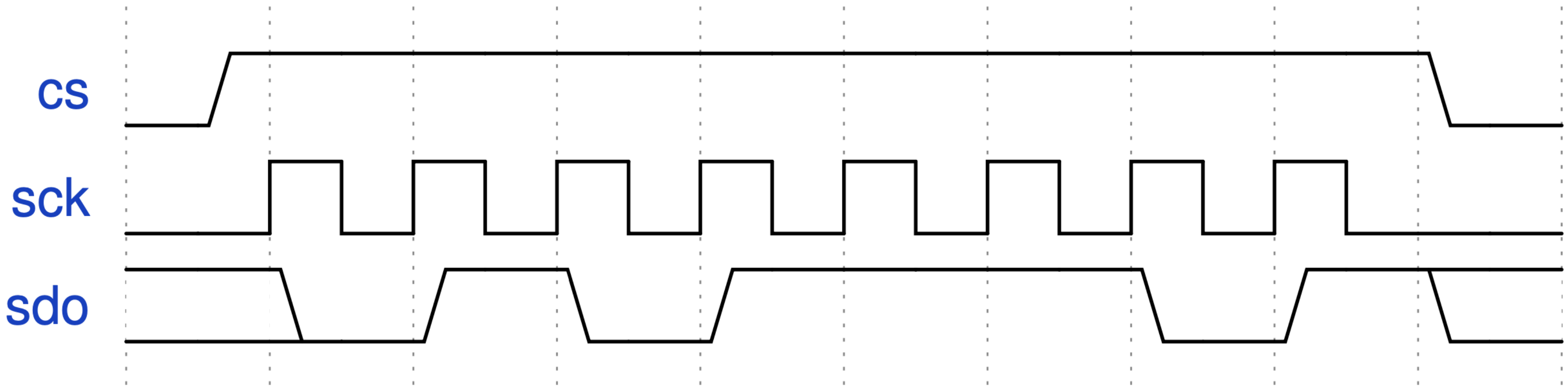
1. Clock Polarity: 1
2. Clock Phase: 1
3. Data Packet Value: `0x65`



SPI Example 2

Determine the following properties for the waveform below:

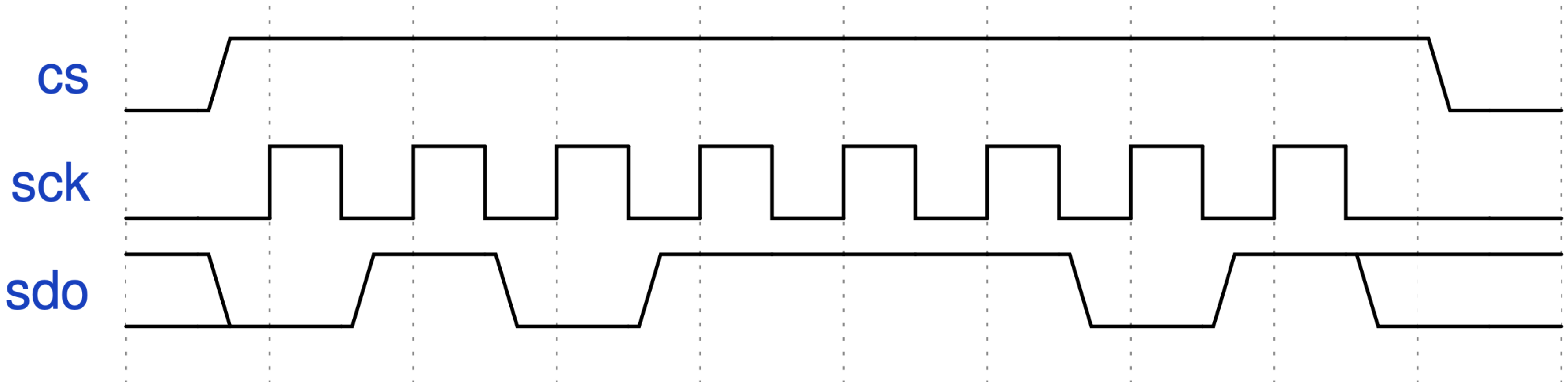
1. Clock Polarity: 0
2. Clock Phase: 1
3. Data Packet Value: `0x5d`



SPI Example 3

Determine the following properties for the waveform below:

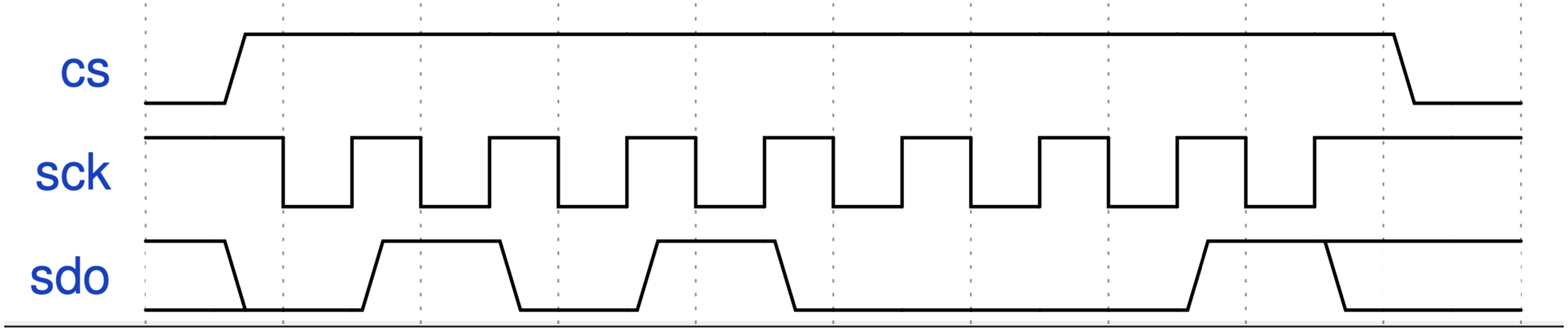
1. Clock Polarity: 0
2. Clock Phase: 0
3. Data Packet Value: `0x5d`



SPI Example 4

Determine the following properties for the waveform below:

1. Clock Polarity: 1
2. Clock Phase: 0
3. Data Packet Value: `0x51`



Basic Configuration in Master/Controller Mode

- Configure clock tree
- Turn on SPI clock domain
- Set SPI parameters
  - Clock rate using baud rate divisor
  - CPOL and CPHA to match slave
  - DFF to 8- or 16-bit data frame format
  - Set LSBFIRST bit to set whether lsb or msb is sent first (normally msb)
  - Configure the NSS pin (can either use software management or a separate GPIO set as an output and manually toggle it)
  - Set to master mode MSTR
- Enable SPI – Set SPE bit to 1

SPI Demo

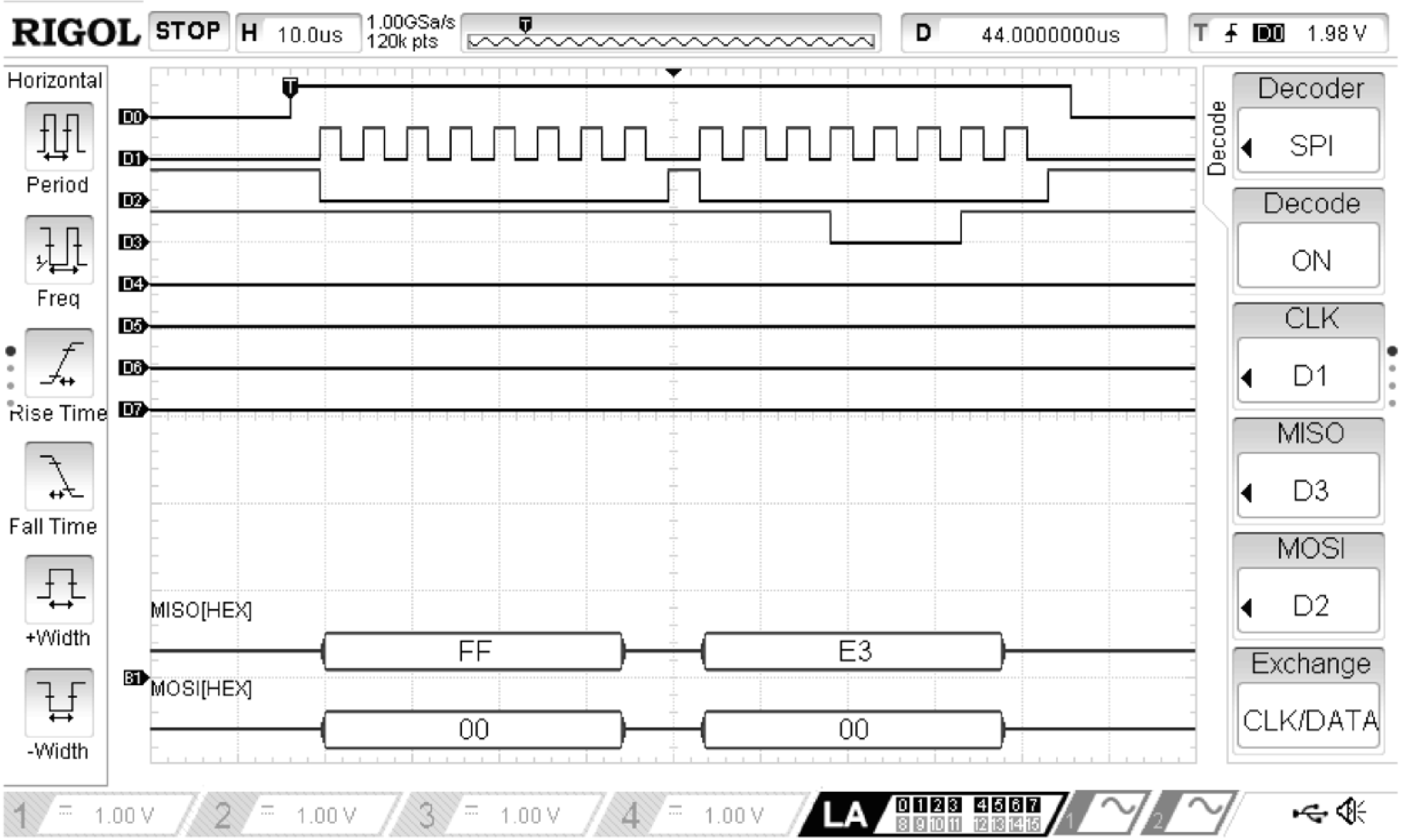
spi_demo.c: main()

```c
int main(void) {
  configureFlash();
  configureClock();

  gpioEnable(GPIO_PORT_A);
  gpioEnable(GPIO_PORT_B);
  gpioEnable(GPIO_PORT_C);

  RCC->APB2ENR |= (RCC_APB2ENR_TIM15EN);
  initTIM(TIM15);

  initSPI(2, 0, 0);

  while(1) {
    digitalWrite(PA11, PIO_HIGH);
    spiSendReceive(0xAB);
    digitalWrite(PA11, PIO_LOW);
    delay_millis(TIM15, 10);
  }
}
```

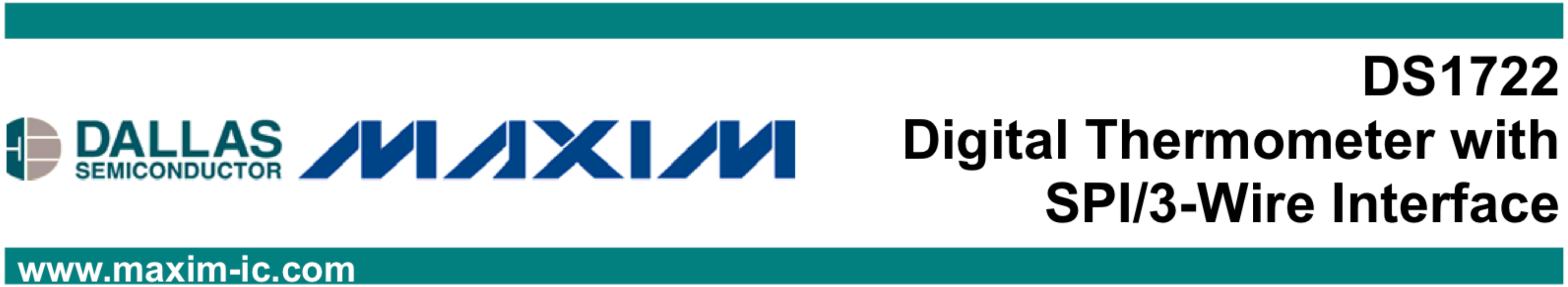
SPI.h Function Prototypes

```c
 /* Enables the SPI peripheral and intializes its clock speed (baud rate), polarity, and phase.
 *    -- br[2:0]: (0x0 to 0x7). The SPI clk will be the master clock / clkdivide.
 *    -- cpol: clock polarity (0: inactive state is logical 0, 1: inactive state is logical 1).
 *    -- cpha: clock phase (0: the first clock transition is the first data capture edge,
 *                          1: the second clock transition is the first data capture edge)
 * Refer to the datasheet for more low-level details. */
void spiInit(int br, int cpol, int cpha);

/* Transmits a character (1 byte) over SPI and returns the received character.
 *    -- send: the character to send over SPI
 *    -- return: the character received over SPI */
char spiSendReceive(char send);
```

An Example SPI Transaction



DS1722 SPI Temperature Sensor

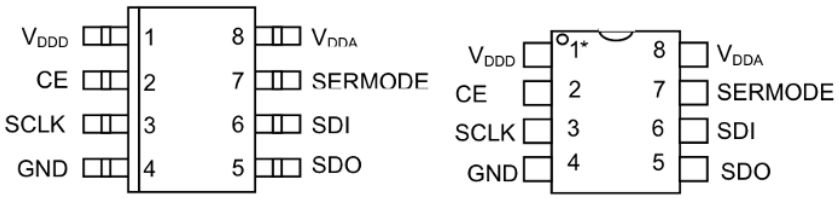


DALLAS SEMICONDUCTOR MAXIM

DS1722
Digital Thermometer with SPI/3-Wire Interface

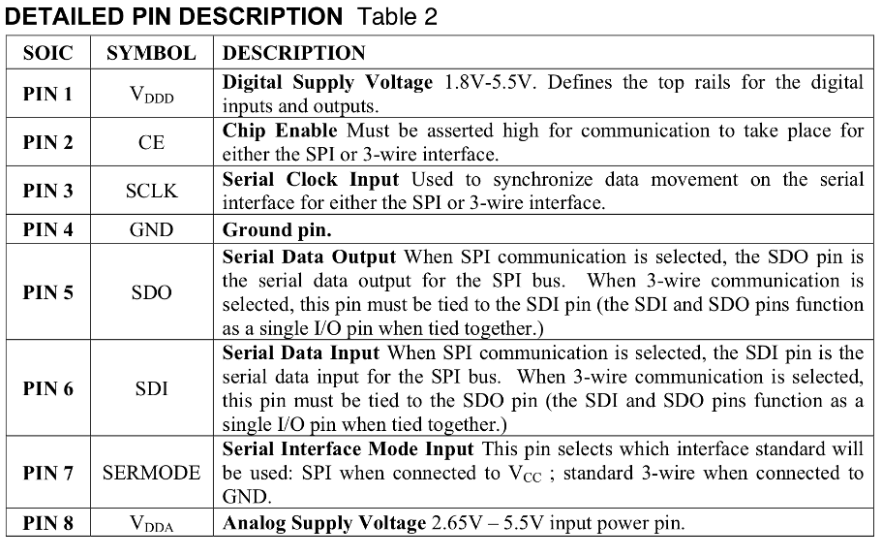
www.maxim-ic.com

PIN ASSIGNMENT



DS1722S
8-Pin SO (150mil)

DS1722U
8-PIN µMAX

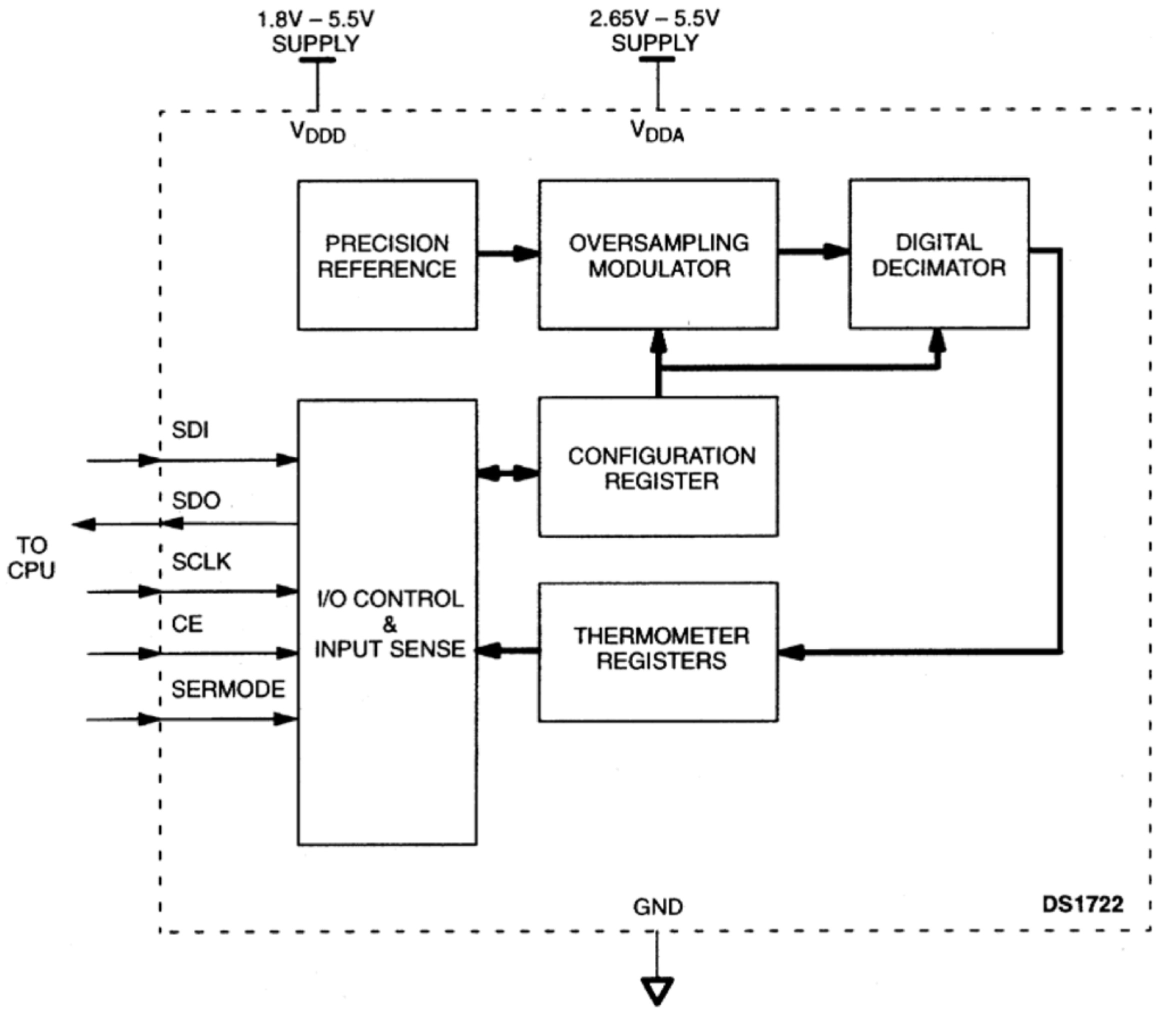


DETAILED PIN DESCRIPTION Table 2

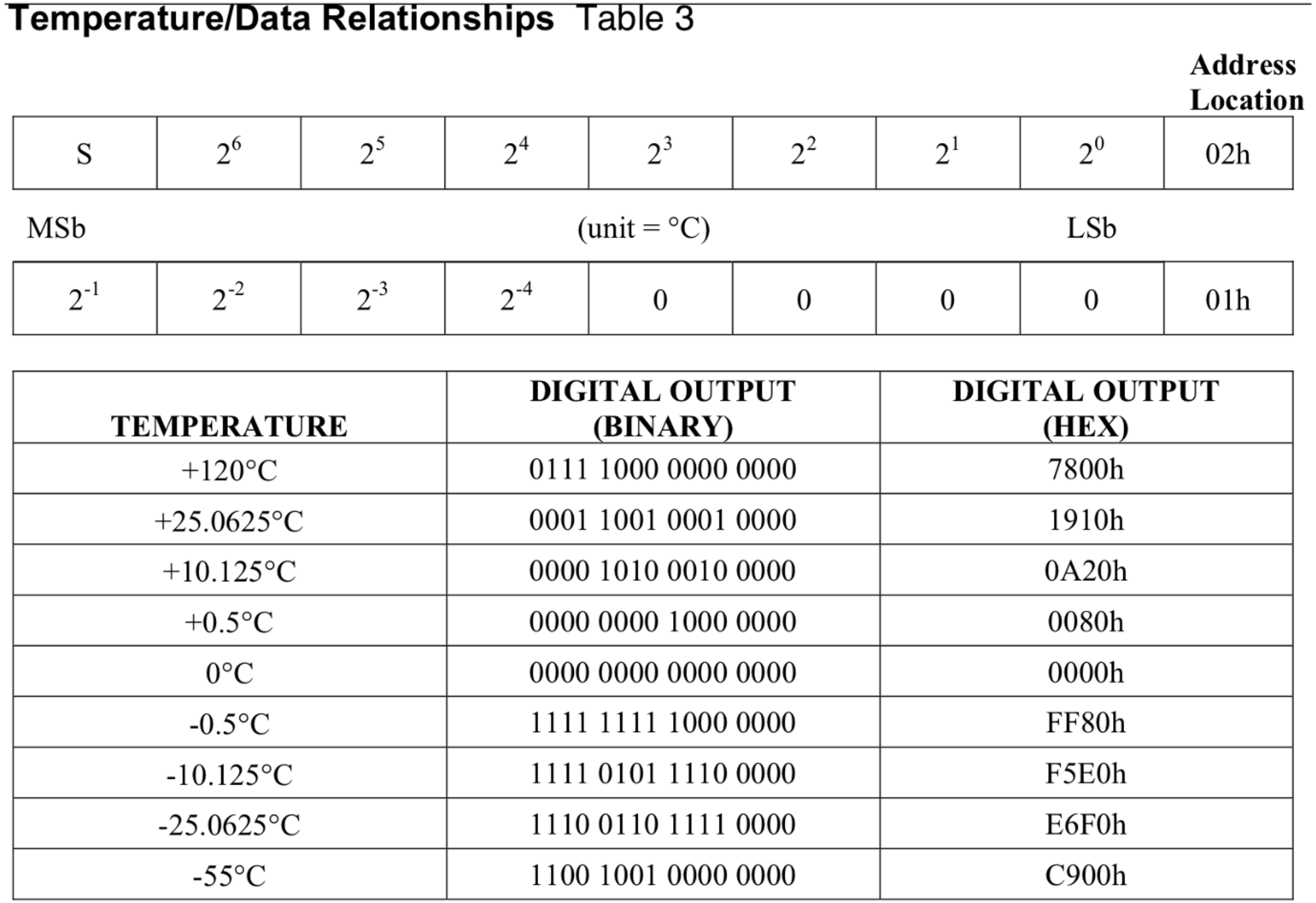
SOIC	SYMBOL	DESCRIPTION
PIN 1	V_{DDD}	**Digital Supply Voltage** 1.8V-5.5V. Defines the top rails for the digital inputs and outputs.
PIN 2	CE	**Chip Enable** Must be asserted high for communication to take place for either the SPI or 3-wire interface.
PIN 3	SCLK	**Serial Clock Input** Used to synchronize data movement on the serial interface for either the SPI or 3-wire interface.
PIN 4	GND	**Ground pin.**
PIN 5	SDO	**Serial Data Output** When SPI communication is selected, the SDO pin is the serial data output for the SPI bus. When 3-wire communication is selected, this pin must be tied to the SDI pin (the SDI and SDO pins function as a single I/O pin when tied together.)
PIN 6	SDI	**Serial Data Input** When SPI communication is selected, the SDI pin is the serial data input for the SPI bus. When 3-wire communication is selected, this pin must be tied to the SDO pin (the SDI and SDO pins function as a single I/O pin when tied together.)
PIN 7	SERMODE	**Serial Interface Mode Input** This pin selects which interface standard will be used: SPI when connected to V_{CC} ; standard 3-wire when connected to GND.
PIN 8	V_{DDA}	**Analog Supply Voltage** 2.65V – 5.5V input power pin.

DS1722 Functional Block Diagram

DS1722 FUNCTIONAL BLOCK DIAGRAM Figure 1



Temperature Data Register Format



Temperature/Data Relationships Table 3

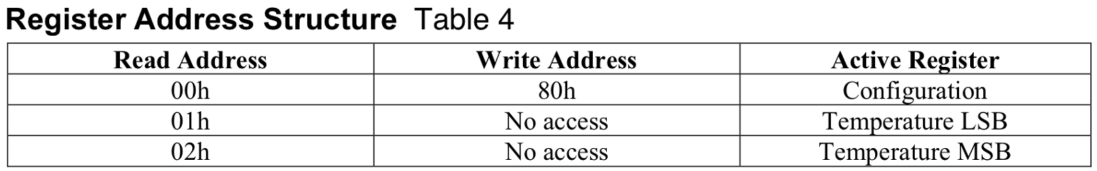
								Address Location
S	2^6	2^5	2^4	2^3	2^2	2^1	2^0	02h

MSb (unit = °C) LSb

2^{-1}	2^{-2}	2^{-3}	2^{-4}	0	0	0	0	01h

TEMPERATURE	DIGITAL OUTPUT (BINARY)	DIGITAL OUTPUT (HEX)
+120°C	0111 1000 0000 0000	7800h
+25.0625°C	0001 1001 0001 0000	1910h
+10.125°C	0000 1010 0010 0000	0A20h
+0.5°C	0000 0000 1000 0000	0080h
0°C	0000 0000 0000 0000	0000h
-0.5°C	1111 1111 1000 0000	FF80h
-10.125°C	1111 0101 1110 0000	F5E0h
-25.0625°C	1110 0110 1111 0000	E6F0h
-55°C	1100 1001 0000 0000	C900h

SPI Transactions

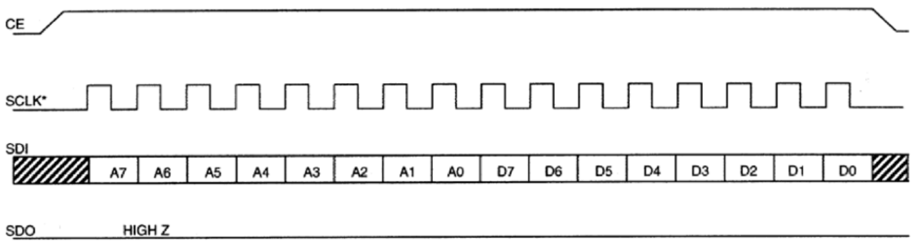


Register Address Structure Table 4

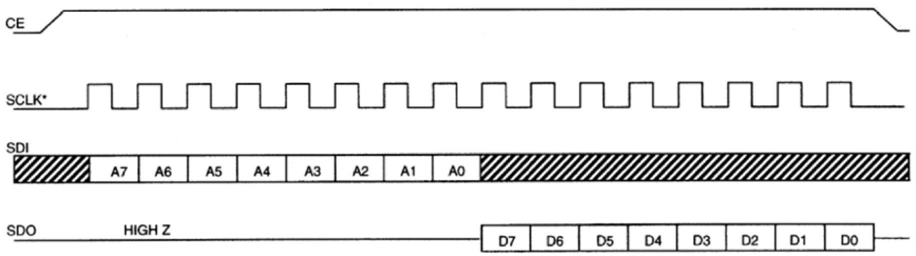
Read Address	Write Address	Active Register
00h	80h	Configuration
01h	No access	Temperature LSB
02h	No access	Temperature MSB

Register structure

SPI SINGLE BYTE WRITE Figure 4



SPI SINGLE-BYTE READ Figure 5



Example waveforms