# *Lab 10: SPI (Serial Peripheral Interface)*

Instructor: Sung-Yeul Park
TA: S M Rakiul Islam

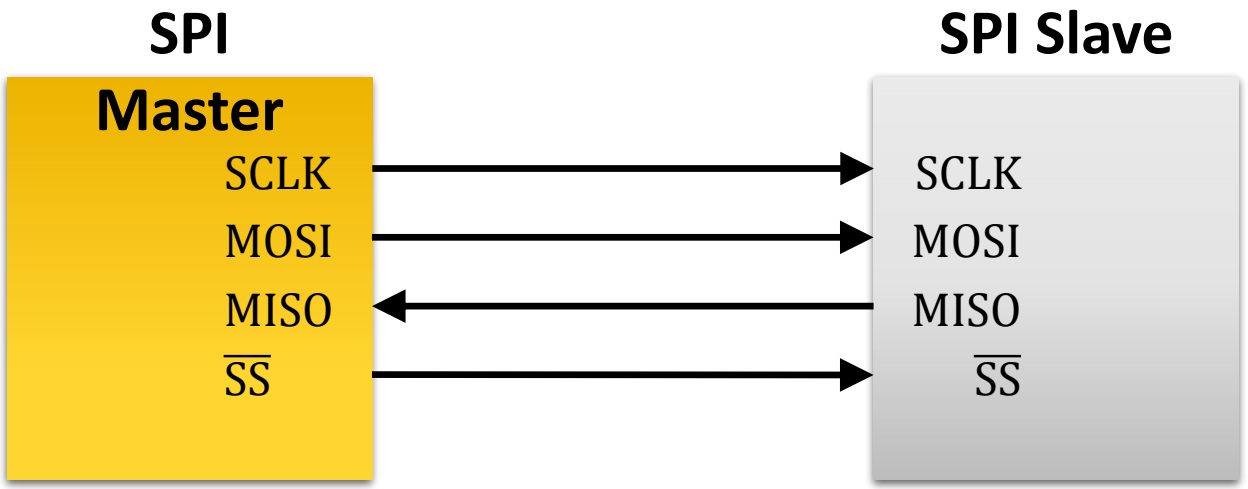
ECE 3411
Department of Electrical and Computer Engineering
University of Connecticut

**February 19th, 2018**

# SPI: Serial Peripheral Interface

The SPI bus specifies four logic signals:

- SCLK : Serial Clock (output from master).
- MOSI : Master Output, Slave Input (output from master).
- MISO : Master Input, Slave Output (output from slave).
- SS : Slave Select (active low, output from master).

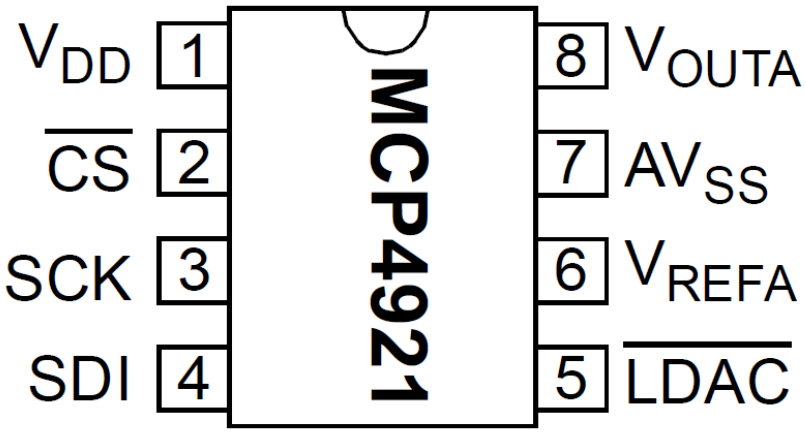


# DAC: Digital to Analog Converter

We use an external DAC for this lab: MCP4921

- 12 bit resolution.
- SPI interface.

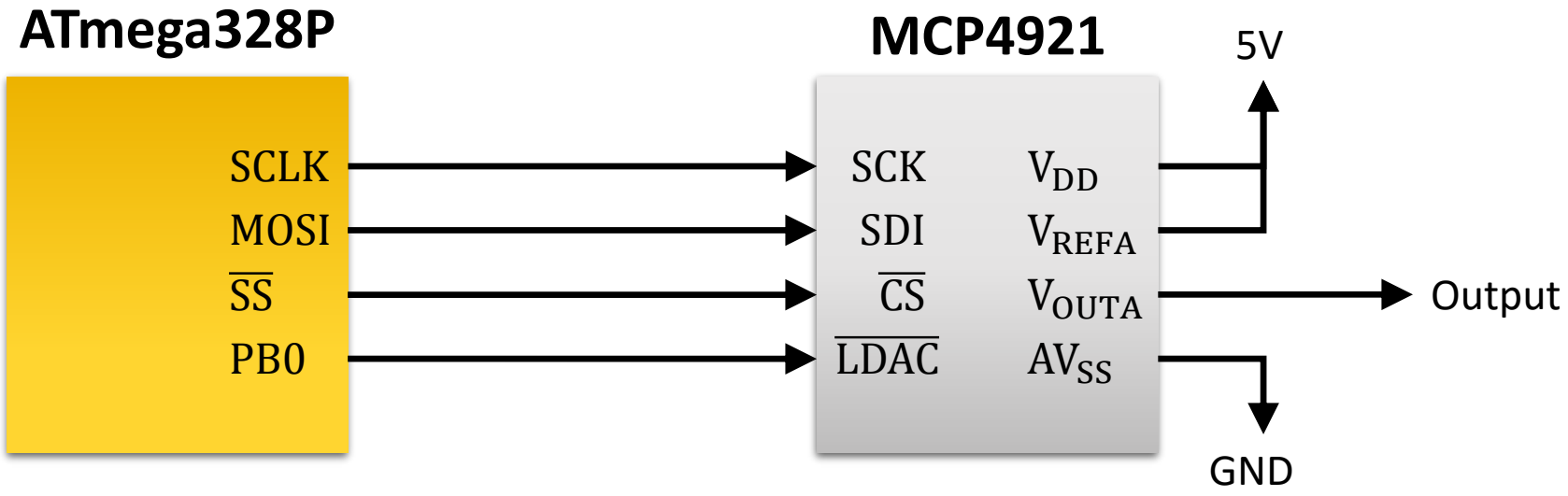
| | | |
|---|---|---|
| 1 | $V_{DD}$ | Positive Power Supply Input (2.7V to 5.5V) |
| 2 | CS | Chip Select Input. (SPI Slave Select) |
| 3 | SCK | SPI Serial Clock Input |
| 4 | SDI | SPI Serial Data Input (MOSI) |
| 5 | LDAC | Synchronization input used to transfer DAC settings from serial latches to the output latches. |
| 6 | $V_{REFA}$ | $DAC_A$ Voltage Input ($AV_{SS}$ to $V_{DD}$) |
| 7 | $AV_{SS}$ | Analog ground |
| 8 | $V_{OUTA}$ | $DAC_A$ Output |



# DAC SPI Interface

MCP4921 acts as SPI Slave and only receives data → MISO is not connected.

- Connect the ATmega328P with MCP4921 as shown in the figure below.
- Notice that LDAC pin also needs to be connected to a GPIO pin on ATmega328P.



# DAC SPI Frame Format

- MCP4921 receives a 16-bit word from the MCU in two 8-bit SPI transactions.
- The format of the 16-bit frame containing 4 command and 12 data bits is shown below.

**REGISTER 5-1: WRITE COMMAND REGISTER**

**Upper Half:**

| W-x | W-x | W-x | W-0 | W-x | W-x | W-x | W-x |
|---|---|---|---|---|---|---|---|
| $\overline{A}$/B | BUF | $\overline{GA}$ | $\overline{SHDN}$ | D11 | D10 | D9 | D8 |
| bit 15 | | | | | | | bit 8 |

**Lower Half:**

| W-x | W-x | W-x | W-x | W-x | W-x | W-x | W-x |
|---|---|---|---|---|---|---|---|
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| bit 7 | | | | | | | bit 0 |

# DAC Command Bits

- The upper 4 bits of the 16 bit word are DAC command bits.
- The description of the 16 bit frame bits is as follows:

bit 15 **A/B:** $DAC_A$ or $DAC_B$ Select bit

1 = Write to $DAC_B$

0 = Write to $DAC_A$

bit 14 **BUF:** $V_{REF}$ Input Buffer Control bit

1 = Buffered

0 = Unbuffered

bit 13 **$\overline{GA}$:** Output Gain Select bit

1 = 1x ($V_{OUT} = V_{REF} * D/4096$)

0 = 2x ($V_{OUT} = 2 * V_{REF} * D/4096$)

bit 12 **$\overline{SHDN}$:** Output Power Down Control bit

1 = Output Power Down Control bit

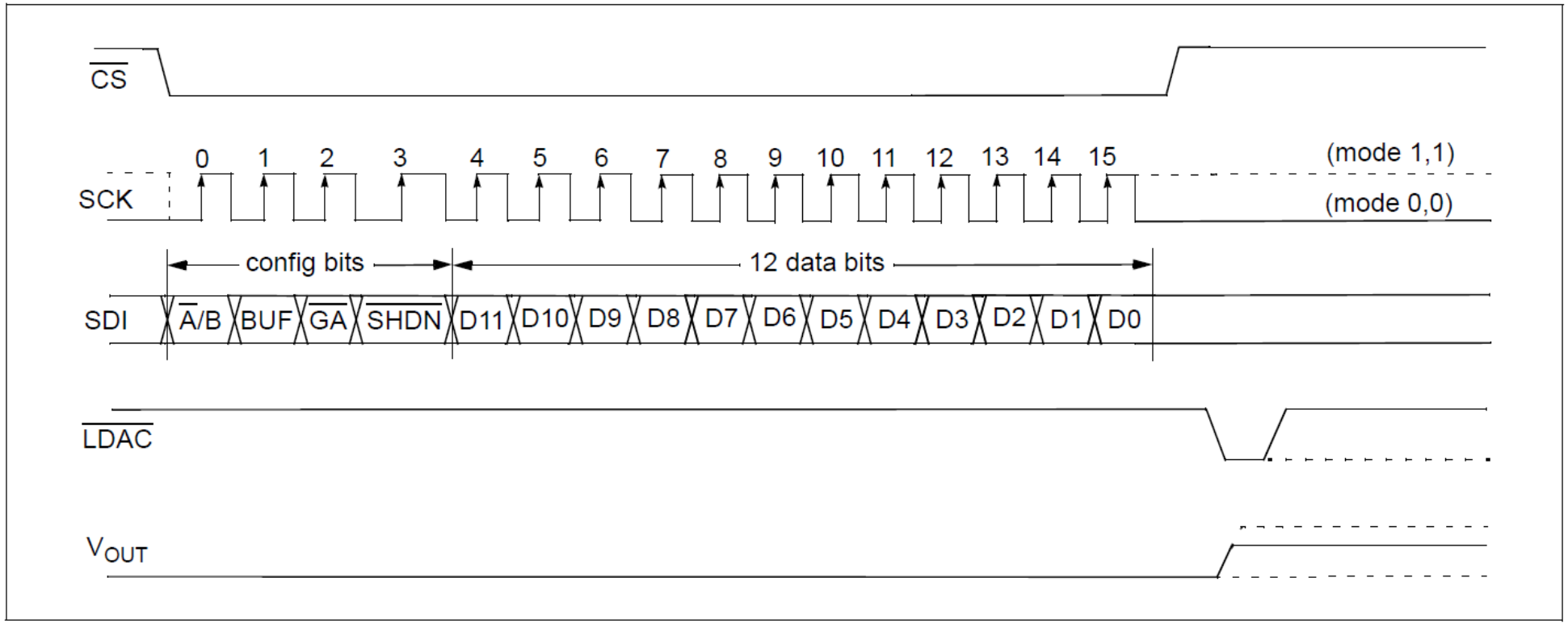
0 = Output buffer disabled, Output is high impedance

bit 11-0 **D11:D0:** DAC Data bits

12 bit number “D” which sets the output value. Contains a value between 0 and 4095.

# DAC SPI Interface Timing

- The figure below shows the timing of one SPI transaction (command + data) between the MCU and DAC.
- You need to implement the same timing through SPI interface on ATmega328P.



# Connections

- Connect the board as in Fig.1.
- You need to connect 12 bit DAC (MCP4921) as SPI device
- You could use 2 potentionmeter
  - 1 for ADC
  - 1 for contrast control
- For pi clarification use datasheet of SPI and Atmega328P

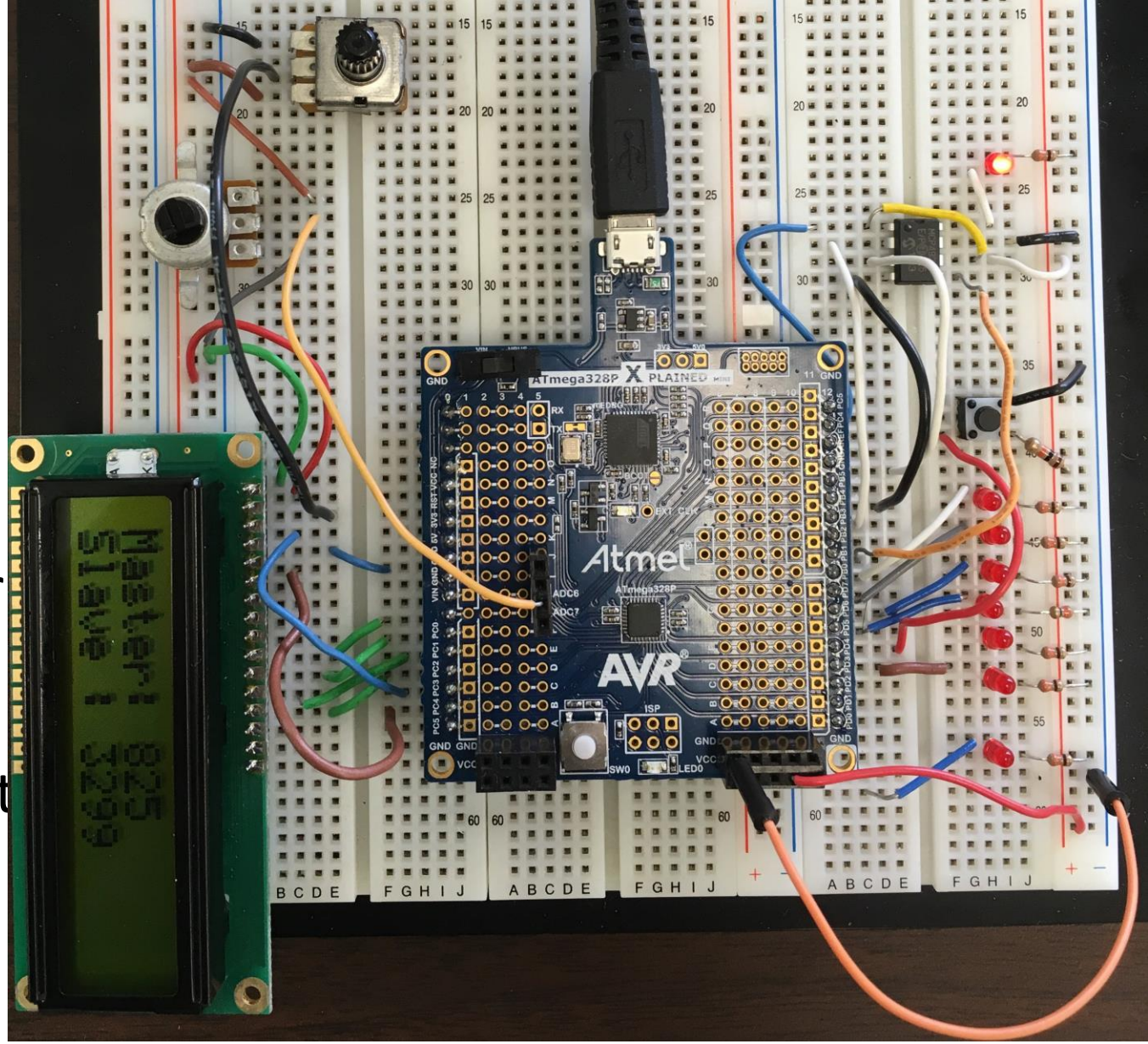
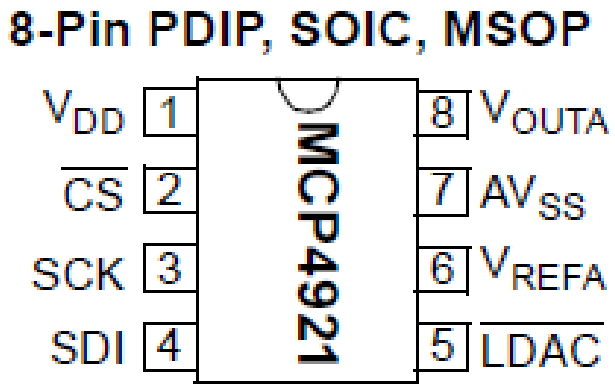
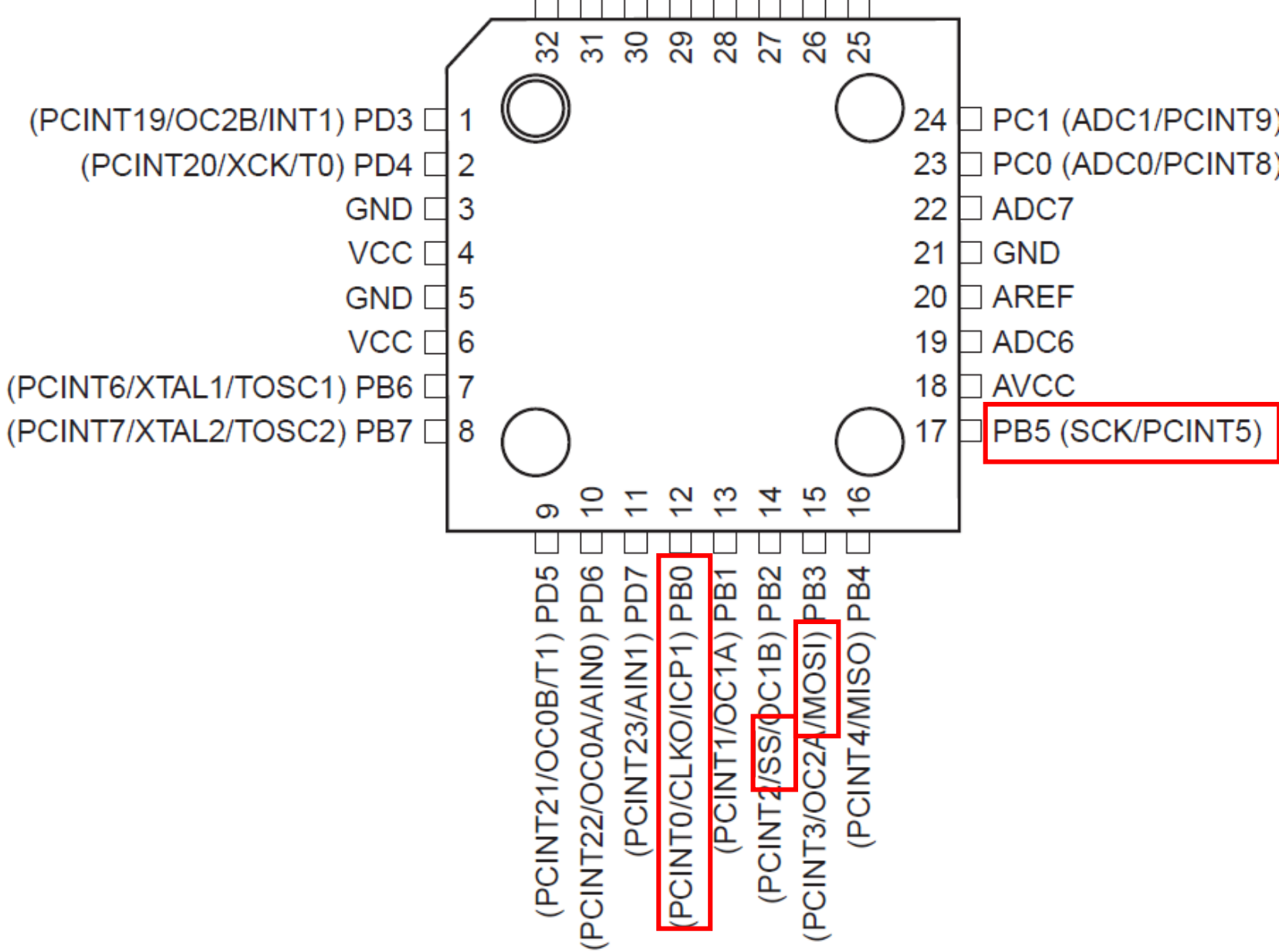



Fig1. Connections for SPI

# Pins


8-Pin PDIP, SOIC, MSOP
VDD 1
CS 2
SCK 3
SDI 4
MCP4921
8 VOUTA
7 AVSS
6 VREFA
5 LDAC


(PCINT19/OC2B/INT1) PD3 1
(PCINT20/XCK/T0) PD4 2
GND 3
VCC 4
GND 5
VCC 6
(PCINT6/XTAL1/TOSC1) PB6 7
(PCINT7/XTAL2/TOSC2) PB7 8
24 PC1 (ADC1/PCINT9)
23 PC0 (ADC0/PCINT8)
22 ADC7
21 GND
20 AREF
19 ADC6
18 AVCC
17 PB5 (SCK/PCINT5)
32 31 30 29 28 27 26 25
9 10 11 12 13 14 15 16
(PCINT21/OC0B/T1) PD5
(PCINT22/OC0A/AIN0) PD6
(PCINT23/AIN1) PD7
(PCINT0/CLKO/ICP1) PB0
(PCINT1/OC1A) PB1
(PCINT2/SS/OC1B) PB2
(PCINT3/OC2A/MOSI) PB3
(PCINT4/MISO) PB4

# SPI Master & Slaves: for further learning

- The SPI bus can operate with a single Master device and with one or more Slave devices.
- In case of multiple slaves, the master selects the slave device with a logic 0 on the select (SS) line.
- During each SPI clock cycle, the master sends a bit on the MOSI line and the slave reads it, while the slave sends a bit on the MISO line and the master reads it.
- This sequence is maintained even when only one-directional data transfer is intended.