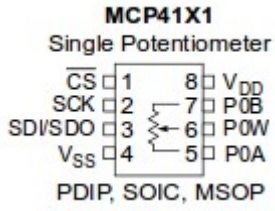
Digitales Potentiometer

Verwendete Pin-Nummern findet man unter hardware/arduino/variants

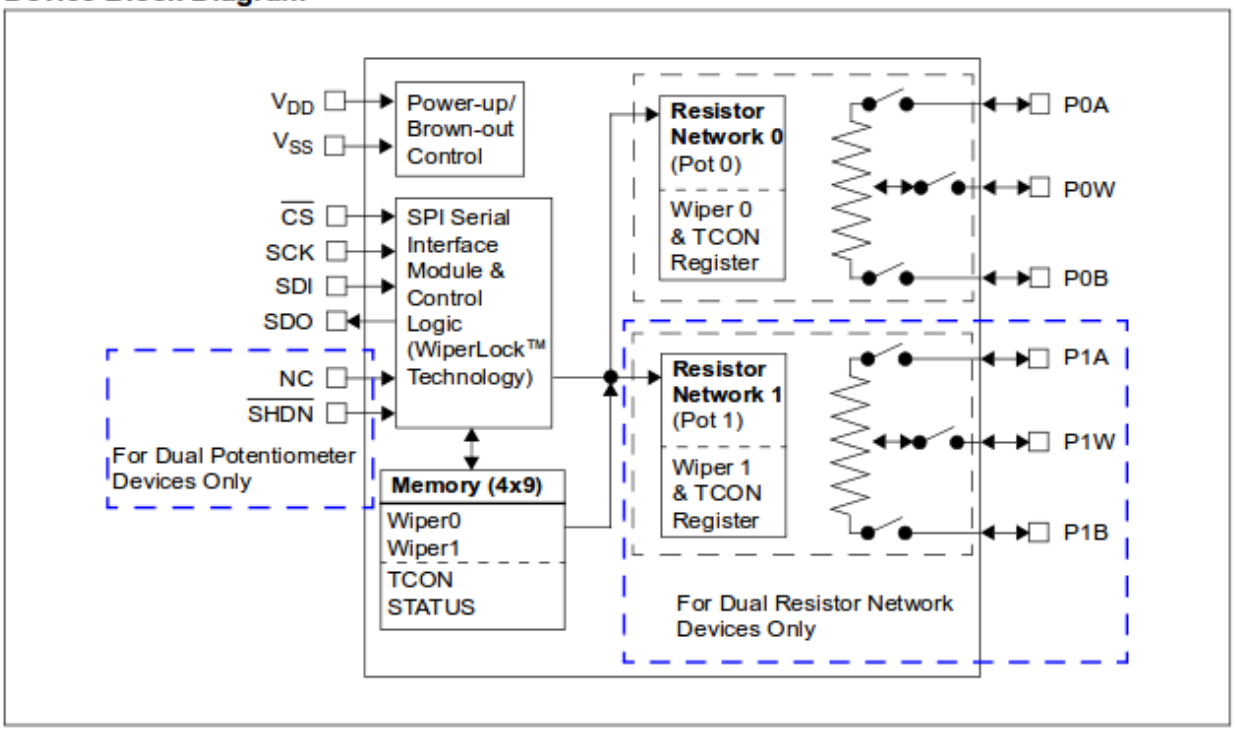
z.B: für das Board **Duemilanove**: SS = 10 (Chip-Select), MOSI = 11 (Master Out/Slave In),MISO=12, SCK = 13

MCP 4151-104E/P

http://pdf1.alldatasheet.com/datasheet-pdf/view/305851/MICROCHIP/MCP4151T.html



Device Block Diagram



Device	# of POTs	Wiper Configuration	Control Interface	Memory Type	WiperLock Technology	POR Wiper Setting	Resistance (typical)		# of Steps	V_{DD} Operating Range (2)
							R_{AB} Options (kΩ)	Wiper - R_W (Ω)		
MCP4131 (3)	1	Potentiometer (1)	SPI	RAM	No	Mid-Scale	5.0, 10.0, 50.0, 100.0	75	129	1.8V to 5.5V
MCP4132 (3)	1	Rheostat	SPI	RAM	No	Mid-Scale	5.0, 10.0, 50.0, 100.0	75	129	1.8V to 5.5V
MCP4141	1	Potentiometer (1)	SPI	EE	Yes	NV Wiper	5.0, 10.0, 50.0, 100.0	75	129	2.7V to 5.5V
MCP4142	1	Rheostat	SPI	EE	Yes	NV Wiper	5.0, 10.0, 50.0, 100.0	75	129	2.7V to 5.5V
MCP4151 (3)	1	Potentiometer (1)	SPI	RAM	No	Mid-Scale	5.0, 10.0, 50.0, 100.0	75	257	1.8V to 5.5V
MCP4152 (3)	1	Rheostat	SPI	RAM	No	Mid-Scale	5.0, 10.0, 50.0, 100.0	75	257	1.8V to 5.5V
MCP4161	1	Potentiometer (1)	SPI	EE	Yes	NV Wiper	5.0, 10.0, 50.0, 100.0	75	257	2.7V to 5.5V
MCP4162	1	Rheostat	SPI	EE	Yes	NV Wiper	5.0, 10.0, 50.0, 100.0	75	257	2.7V to 5.5V

TABLE 4-1: MEMORY MAP

Address	Function	Memory Type
00h	Volatile Wiper 0	RAM
01h	Volatile Wiper 1	RAM
02h	Reserved	—
03h	Reserved	—
04h	Volatile TCON Register	RAM
05h	Status Register	RAM
06h-0Fh	Reserved	—

TABLE 7-1: COMMAND BIT OVERVIEW

C1:C0 Bit States	Command	# of Bits
11	Read Data	16-Bits
00	Write Data	16-Bits
01	Increment	8-Bits
10	Decrement	8-Bits

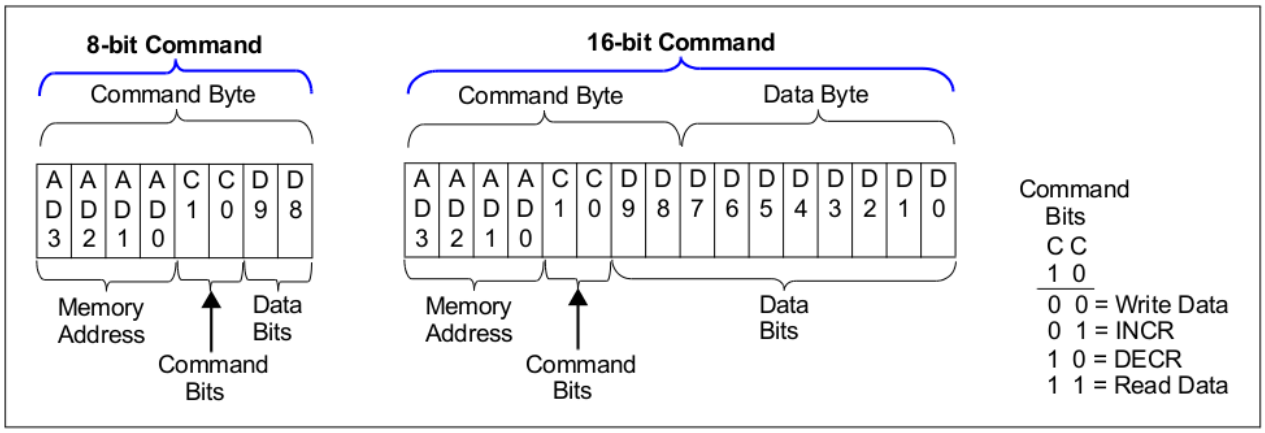
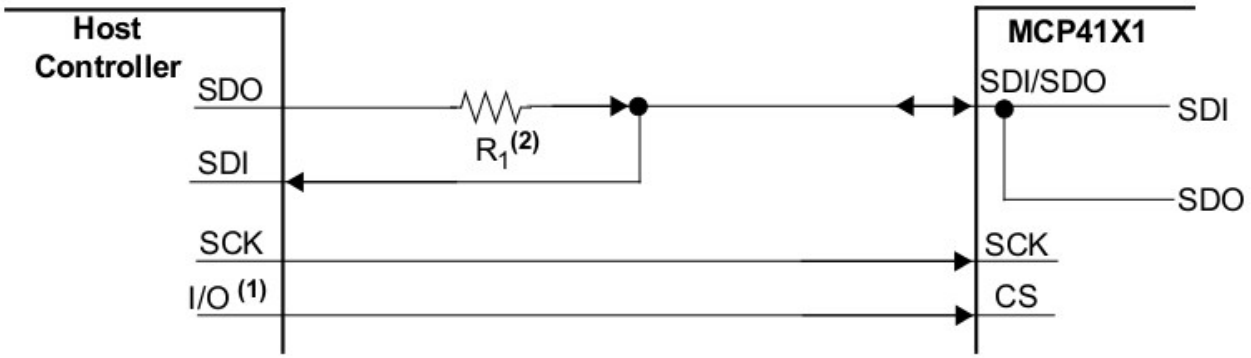


FIGURE 7-1: *General SPI Command Formats.*



Wahl des R1: 1kOhm (10kOhm war zu hochhohmig, konnte den internen Pullup im MCP41X1 nicht überschreiben; je niederohmiger desto schneller wird der Bus

	COMMAND BYTE								DATA BYTE								
SDI	AD3	AD2	AD1	AD0	0	0	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
SDO	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	Valid Address/Command combination
	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	Invalid Address/Command combination (1)

Note 1: If an Error Condition occurs (CMDERR = L), all following SDO bits will be low until the CMDERR condition is cleared (the $\overline{CS}$ pin is forced to the inactive state).

FIGURE 7-2: *Write Command - SDI and SDO States.*

COMMAND BYTE

(INCR COMMAND (n+1))

SDI	AD3	AD2	AD1	AD0	0	1	X	X	
SDO	1	1	1	1	1	1	1*	1	**Note 1, 2**
	1	1	1	1	1	1	0	0	**Note 1, 3**

Note 1: Only functions when writing the volatile wiper registers (AD3:AD0) 0h and 1h.

2: Valid Address/Command combination.

3: Invalid Address/Command combination all following SDO bits will be low until the CMDERR condition is cleared. (the $\overline{CS}$ pin is forced to the inactive state).

4: If a Command Error (CMDERR) occurs at this bit location (*), then all following SDO bits will be driven low until the $\overline{CS}$ pin is driven inactive (V_{IH}).

FIGURE 7-6: *Increment Command - SDI and SDO States.*

TABLE 7-2: MEMORY MAP AND THE SUPPORTED COMMANDS

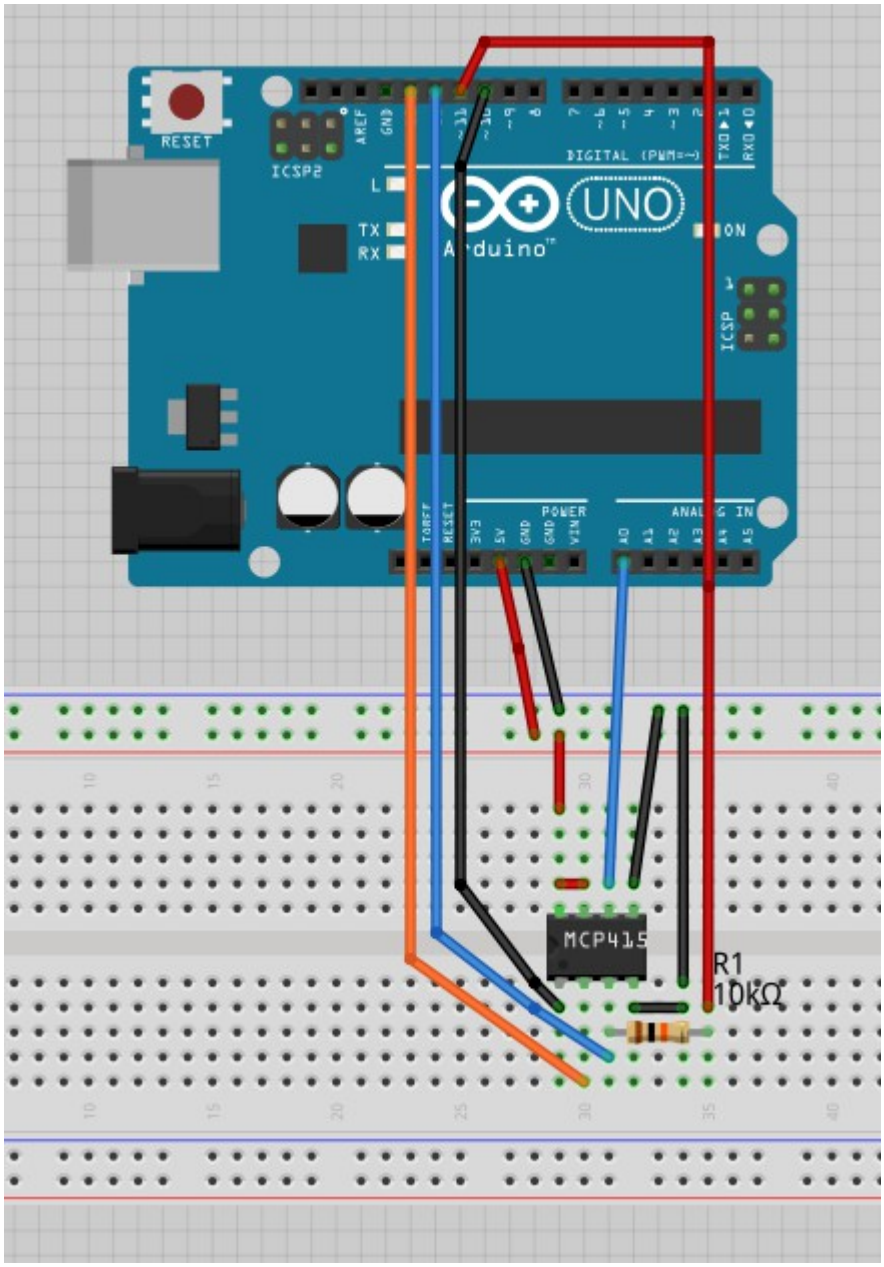
Address		Command	Data (10-bits) (1)	SPI String (Binary)	
Value	Function			MOSI (SDI pin)	MISO (SDO pin) (2)
00h	Volatile Wiper 0	Write Data	nn nnnn nnnn	0000 00nn nnnn nnnn	1111 1111 1111 1111
		Read Data	nn nnnn nnnn	0000 11nn nnnn nnnn	1111 111n nnnn nnnn
		Increment Wiper	—	0000 0100	1111 1111
		Decrement Wiper	—	0000 1000	1111 1111

REGISTER 4-2: TCON BITS (1, 2)

R-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
D8	R1HW	R1A	R1W	R1B	R0HW	R0A	R0W	R0B
bit 8								bit 0

REGISTER 4-1: STATUS REGISTER

R-1	R-1	R-1	R-1	R-0	R-x	R-x	R-x	R-x
D8:D5				RESV	RESV	RESV	SHDN	RESV
bit 7								bit 0



Fritzing-Verdrahtungsplan; Achtung: Pin A vom MCP4151 sollte auf VCC und Pin B auf GND verdrahtet werden, hier sinkt bei steigendem Digital-Wert die Spannung am Wiper Ausgang. Der Widerstandswert mit 10k ist zu hoch: es gelingt damit nicht, gegen den inneren Pullup den Pegel auf Null zu ziehen.

```
/*
  kner 2013 MCP4151 Digital Poti
  Pin 5(A) - Full Scale End -  connect this to VDD
  Pin 6(W) - wiper - connet to Analog In A0
  Pin 7(B) - GND
  Pin 1(CS) - arduino pin 10  (SS pin)
  Pin 3(SDI/SDO) - arduino pin 11 (MOSI pin) via resistor for writing to poti
  Pin 3(SDI/SDO) - arduino pin 12 (MISO pin) direct for reading from poti
  Pin 2(SCK) - arduino pin 13 (SCK pin)

 data frame
 increment: 1 byte    0000 0100
 decrement: 1 byte    0000 1000
```

```
 write value: 2byte 0000 00dd dddd dddd  values from 0x00 to 0x100
 read status register (address 5): 2 byte 0101 1100 xxxx xxdx        - bit 1 =
hardware shutdown bit;
                                                                    a shutdown
disconnects pin A and sets the wiper to B
 tcon register (address 4): can connect/disconnect terminals(A,B,W) and
hardware-shutdown(HW) the device
 9 bits: 1 1111 R0HW-R0A-R0W-R0B   bit 8 - bit 4 not used/
 to shutdown:  0000 0001 1111 0111    (bit 9 is not used)
 to disconnect A: 111111011
*/


// inslude the SPI library:
#include <SPI.h>
#include <avr/io.h>


// set pin 10 as the slave select for the digital pot:
const int slaveSelectPin = 10;

void setup() {
  Serial.begin(9600);
  pinMode (slaveSelectPin, OUTPUT);
  SPI.begin();
  SPI.setClockDivider(SPI_CLOCK_DIV128); //depends on resistors
}

void loop() {
   Serial.print("Max: ");
   //after POR: wiper is in the middle
   digitalPotWrite(0x100);  //Maximum
   Serial.println(analogRead(A0));
   delay(3000);

    //Shutdown output should be 0
    Serial.print("Shutdown:");
    digitalPotWrite16Bit(0x40F7);
    Serial.println(analogRead(A0));
    Serial.print("Status:");
    Serial.println(digitalPotReadControlregister(),HEX); // did it work?
    // i am not successful in reading register 5 (Status Register)


    //reactivate after Shutdown
    Serial.print("Finish shutdown:");
    digitalPotWrite16Bit(0x40FF);
    Serial.println(digitalPotReadControlregister(),HEX);
   Serial.print("Min: ");
   digitalPotWrite(0x0);  //Minimum
   Serial.println(analogRead(A0));
   delay(3000);
   Serial.println("inc: ");
   for (int level = 0; level < 257; level++) {
      digitalPotInc();
      Serial.println(analogRead(A0));
      delay(10);
    }
    Serial.println("dec: ");
   //decrement
   for (int level = 0; level < 257; level++) {
      digitalPotDec();
```

```
      Serial.println(analogRead(A0));
      delay(10);
    }
    Serial.println("inc level: ");
    for (int level = 0; level < 257; level++) {
      digitalPotWrite(level);
      Serial.println(analogRead(A0));
      delay(10);
    }
    Serial.println("dec level: ");
    for (int level = 257; level >0; level--) {
      digitalPotWrite(level);
      Serial.println(analogRead(A0));
      delay(1);
    }
    while(1);
}

void digitalPotInc() {
  digitalWrite(slaveSelectPin,LOW);
  SPI.transfer(4);
  digitalWrite(slaveSelectPin,HIGH);
}
void digitalPotDec() {
  digitalWrite(slaveSelectPin,LOW);
  SPI.transfer(8);
  digitalWrite(slaveSelectPin,HIGH);
}

void digitalPotWrite(int value) {
  digitalWrite(slaveSelectPin,LOW);
  SPI.transfer((value>>8)&0b00000001);  //high byte least significant bit
  SPI.transfer(value & 0xff);
  digitalWrite(slaveSelectPin,HIGH);
}

void digitalPotWrite16Bit(int value) {
  digitalWrite(slaveSelectPin,LOW);
  SPI.transfer(value>>8);  //the short way also works
  SPI.transfer(value);
  digitalWrite(slaveSelectPin,HIGH);
}

uint16_t digitalPotReadControlregister() {
  digitalWrite(slaveSelectPin,LOW);
  int highbyte = SPI.transfer(0x4F);
  int lowbyte =   SPI.transfer(0xFF);
  digitalWrite(slaveSelectPin,HIGH);
  return highbyte*256+lowbyte;
}
```