EARTH PEOPLE TECHNOLOGY

EPT-4901-DA-S1 Digital To Analog Converter Board User Manual

The EPT-4901-DA-S1 is a single channel eight bit DAC. It includes a potentiometer which varies the voltage applied to the VREF pin of the DAC. This provides an amplitude control for the output of the DAC. The SPI interface allows easy connection with an Arduino using the SPI library. There is a power indicator Green LED to indicate the board is powered. It has Headers that are surface mounted on the bottom of the board. Access to the SPI bus is made using these header pins. The DAC output is available on the under board headers or the top facing header. The board also includes a 8Mb Flash chip which utilized only in the +5V mode.

The EPT-4901-DA-S1 is designed to provide an analog output for any Arduino board. A 20KΩ potentiometer is connected to the VREF pin. This varies the voltage applied to the VREF pin and provides amplitude control for the analog output voltage. The MCP4901 DAC chip will convert any eight bit digital word into an analog voltage. The settling time for the chip is 4.5µseconds. The output analog voltage can reproduce a sine wave of approximately 800Hz.

Using the EPT-4901-DA-S1

This board is highly compatible with the Arduino library. Include the “SPI.h” library in your sketch and call the “SPI.Transfer” function. There are no registers to set and no registers to read from in the MCP4901. The chip has a maximum SPI clock frequency of:

- 20MHz

It requires to eight bit bytes to be written to the chip in order to perform a Digital to Analog conversion.

Register Set

There is only one register accessible on the MCP4901, Write Command Register.

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	BUF	GA	SHDN	D7	D6	D5	D4	D3	D2	D1	D0	X	X	X	X

Where:

bit 15

0 = Write to DAC register
1 = Ignore this command

bit 14 **BUF**: VREF Input Buffer Control bit

1 = Buffered
0 = Unbuffered

bit 13 **GA**: Output Gain Selection bit

1 = 1x (VOUT = VREF * D/256)
0 = 2x (VOUT = 2 * VREF * D/256)

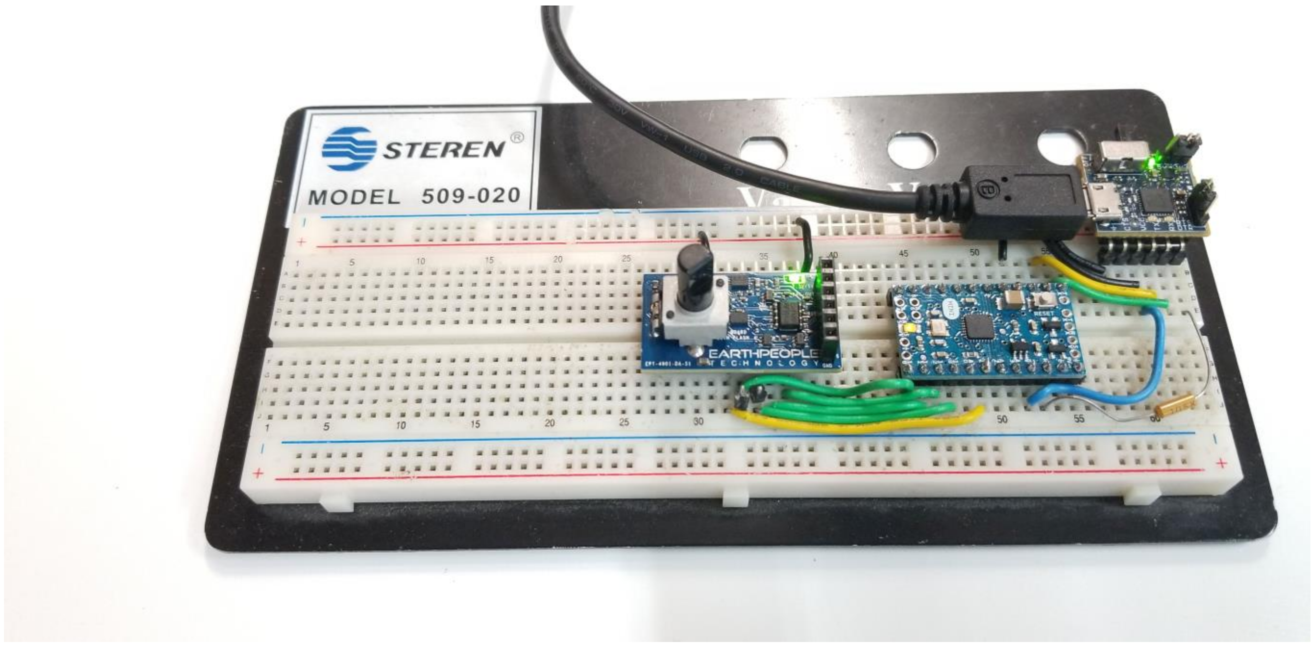
bit 12 **SHDN**: Output Shutdown Control bit

1 = Active mode operation. VOUT is available.
0 = Shutdown the device. Analog output is not available. VOUT pin is connected to 500 kΩ (typical).

bit 11-0 D11:D0: DAC Input Data bits. Bit x is ignored.

EPT-4901-DA-S1 and Arduino Programming

The Arduino IDE makes coding the MCP4901 DAC quite easy. Everything needed to communicate with the DAC is in the “SPI” library. Just include the “SPI.h” file in your Arduino sketch. Connect EPT-4901-DA-S1 to the Arduino and plug the USB cable into a port on the PC.



A simple code example to output a Sine Wave:

```
 /*
DAC Out 5V

Platform: Arduino Mini
*/

#include "EPT_SineTool.h"
#include <SPI.h>

//DAC Configuration Register Bits
#define DAC_WRITE       0x00 //Bit 15: 0 to Write to DAC register
#define DAC_BUFFER       0x40 //Bit 14: 1 to Buffer VREF
#define DAC_GAIN_SELECT  0x20 //Bit 13: 1 to set Gain to 1x
```

```
#define DAC_SHDN        0x10 //Bit 12: 1 to set Active Mode operation


const int DACSelectPin = 10;


unsigned char DACConfigAndCountValue;

unsigned char DACCountHighValue;

unsigned char DACCountLowValue;

unsigned char byteToSendToDAC;

unsigned char sineWaveIndex;

unsigned char upperByteForDAC;

unsigned char lowerByteForDAC;


void setup() {

  pinMode (DACSelectPin, OUTPUT);


  DACConfigAndCountValue = DAC_WRITE | DAC_BUFFER | DAC_GAIN_SELECT |

  DAC_SHDN;


  DACCountHighValue = 0;

  DACCountLowValue = 0;

  byteToSendToDAC = 0;

  sineWaveIndex = 0;

  upperByteForDAC = 0;

  lowerByteForDAC = 0;


  //Initialize SPI

  SPI.begin();
  //Initialize the Sine Wave Tool
  EPT_SineToolInit();

  //Initialize the Debug Port
  //Serial.begin(115200);

}

void loop()
{
   //Select Sine Wave data value
   byteToSendToDAC = EPT_SineToolData(sineWaveIndex);
   //delay(1);

   //Set the upper byte to send to the DAC
   DACCountHighValue = byteToSendToDAC>>4;
   upperByteForDAC = DACConfigAndCountValue | DACCountHighValue;

   //Set the lower byte to send to the DAC
   lowerByteForDAC = byteToSendToDAC<<4;
   //Select the DAC chip;
   digitalWrite(DACSelectPin, LOW);

   //Send in the address and value via SPI:
   SPI.transfer(upperByteForDAC);
   //Send Header Byte 1
   SPI.transfer(lowerByteForDAC);



   //Take the SS pin high to de-select the DAC chip;
   digitalWrite(DACSelectPin, HIGH);

   //delay the sine wave;
   EPT_SineToolDelay(0);

   //Update the sine wave index
   sineWaveIndex += 1;

}
```

There are two files that are required from EPT to prepare the sine wave:

- EPT_SineTool.cpp
- EPT_SineTool.h

These files provide a lookup table for the correct sine wave output per given unit time. The frequency of the sine wave can be modified by adding a delay time to each fetch cycle of the 255 sample read cycle.