**معرفی:**

تراشه های سری AD53xx مبدل های انالوگ به دیجیتال (DAC) می باشند که دارای چهار خروجی هستند. بر خلاف فایل ارائه شده که ذکر گردیده تراشه سری AD5314 یک تراشه 12 بیتی می باشد این تراشه 10 بیتی است و سری AD5324 ان دارای رزولوشن 12 بیت می باشد. این تراشه در پکیج MSOP ارائه می شود. این تراشه ولتاژ کاری 2.5 ولت تا ولت را دارد که با ولتاژ کاری 3 جریان مصرفی حدود 300 میلی امپر خواهد داشت. نکته بسیار مهم این تراشه این است که در خروجی های این تراشه تقویت کننده قرار دارد که بتواند بهترین عملکرد را داشته باشد و ریپل ولتاژ به حداقل برسید. این تراشه با سرعت کلاک 30 مگاهرتز کار می کند و با استانداردهای استاندارد SPI ، QSPI ، MICROWIRE و DSP سازگار است.

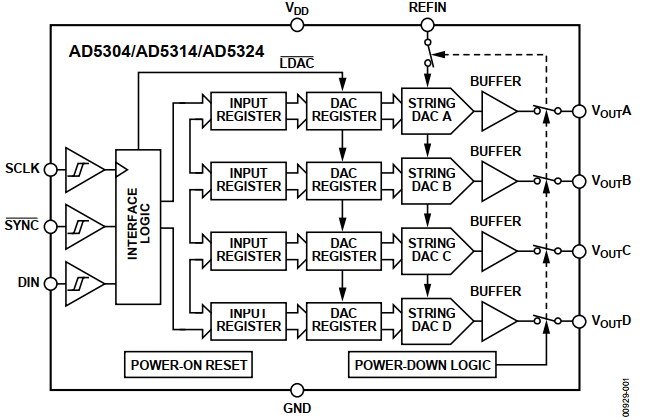
مرجع ولتاژ برای هر چهار خروجی DAC یک پایه می باشد.

یکی از قابلیت های این تراشه استفاده از تکنولوژی برای کاهش جریان مصرفی در هنگام standby می باشد که جریان را به 200 میلی آمپر کاهش میدهد. که همین امر این قطعه را به یک قطعه ایده ال برای تجهیزات قابل حمل که نیاز به مصرف کم دارند و منبع انرژی ان ها باتری است تبدیل می کند.

از کاربرد های متداول این قطعه میتوان به موارد زیر اشاره کرد:

* ابزارهای قابل حمل باتری
* تنظیم دیجیتال و تنظیم افست
* منابع ولتاژ و جریان قابل برنامه ریزی
* تضعیف کننده قابل برنامه ریزی
* کنترل فرآیند صنعتی

در شکل یک نقشه پایه های این تراشه و بلوک دیاگرام داخلی آن نشان داده شده است.



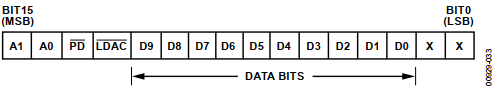
**شکل 1**

**عملکرد:**

power-down mode:

این مود به جهت مصرف کم این تراشه طراحی شده است و با انتخاب این مود ولتاژ تغذیه از 5 به 3 ولت کاهش می یابد (ولتاژ ورودی 5 ولت تغذیه مربوط به مود Normal operation می باشد)

به جهت غیر فعال کردن این مود کاری بیت 13 (PD) باید از صفر به یک (1) تغییر یابد.

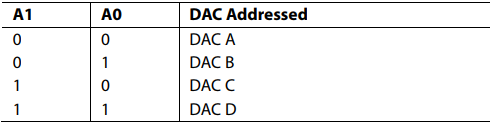


**شکل** 2

Normal operation:

PD ->1

بیت های A1 و A2 مطابق جدول زیر برای تنظیم و انتخاب خروجی مورد نظر می باشد:



بیت LDAC تعیین کننده نحوه بروز رسانی خروجی ها می باشد در صورتی که صفر تنظیم شود همه خروجی ها همزمان به روز می شوند و در صورتی که یک تنظیم شود فقط خروجی آدرس دهی شده به روز می شود . لذا بر اساس صورت پروژه نیاز داریم که فقط خروجی آدرس دهی شده به روز گردد.

LDAC ->1

بیت های D0 تا D9 ) به دلیل 10 بیتی بودن تراشه(مربوط به مقدار دهی می باشند که با تعیین و مقدار دهی ان ها در مد باینری میتوان اعدادی بین 0 تا 1023 را در خروجی مورد نظر که توسط دو بیت A1 و A2 انتخاب می شوند قرار داد.

کد مربوط به تبدیل عدد دسیمال به باینری در زیر اورده شده است(num عدد مورد نظر می باشد و str عدد باینری شده):

uint8\_t bitsCount = sizeof( num ) \* 8;

char str[ bitsCount + 1 ];

uint8\_t i = 0;

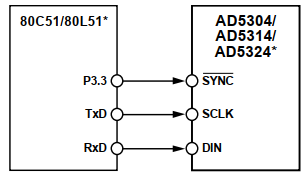
while ( bitsCount-- )

str[ i++ ] = bitRead( num, bitsCount ) + '0';

str[ i ] = '\0';

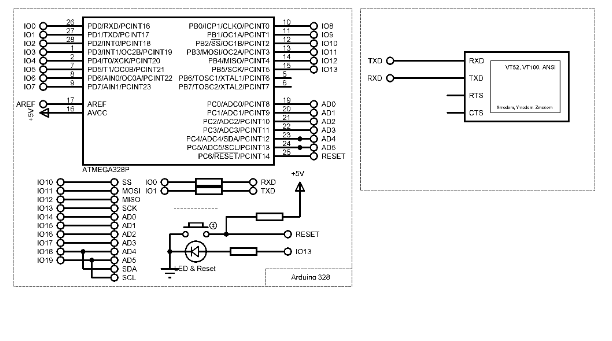
برای ارتباط با این تراشه در مود UART به طریق زیر عمل میکنم

با استفاده از 3 اتصال بین میکرو و تراشه ارتباط بر قرار میگردد. پین P3.3 به جهت برنامه ریزی و انتقال اطلاعات مورد استفاده قرار میگرید.



**شکل** 3

**مدار نرم افزار پروتئوس:**



**شکل** 4

**کد نوشته شده در نرم افزار اردینو:**

#include <SoftwareSerial.h>

SoftwareSerial mySerial(10, 11); // RX, TX

byte LSB;

byte MSB;

void setup()

{

Serial.begin(9600);

mySerial.begin(9600);

pinMode(3, OUTPUT); // pin 3.3 in datasheet

// First config for AD5314:

LSB = 00000000;

MSB = 00110000; ///PD=1 , LDAC=1

digitalWrite(3, LOW);

mySerial.print(LSB, BIN);

digitalWrite(3, HIGH);

mySerial.print(MSB, BIN);

Serial.println("PD and LDAC set to 1 ");

Serial.println("Normal operation is Active");

delay(2000);

}

void loop()

{

WriteDAC('A',250);

delay(700);

}

void WriteDAC(char a, int num){

uint8\_t bitsCount = sizeof( num ) \* 5;

char str[ bitsCount + 1 ];

char t[9];

char t2[3];

uint8\_t i = 0;

while ( bitsCount-- )

str[ i++ ] = bitRead( num, bitsCount ) + '0';

str[ i ] = '\0';

Serial.print( "Binary of cammand: ");

Serial.println( str );

i = 2;

int bitt=9,s=0;

while ( bitt-- )

{

t[s++]=str[i++];

}

i = 0;

s=0;

bitt=2;

while ( bitt-- )

{

t2[s++]=str[i];

i=i+1;

}

// Serial.println( t );

// Serial.println( t2 );

//Select AD5314 Output:

switch (a) {

uint16\_t r;

uint8\_t b1,b2;

uint8\_t bitsCount;

uint8\_t i ;

char str[ 256 ];

case 'A':

Serial.println("Your output is A");

b1 = 0b0000011, b2 = t2;

r = (b1<<4) + b2;

bitsCount = sizeof( r ) \* 4;

i = 0;

while ( bitsCount-- )

str[ i++ ] = bitRead( r, bitsCount ) + '0';

str[ i ] = '\0';

Serial.print( "MSB =");

Serial.println( str );

Serial.print( "LSM =");

Serial.println( t);

digitalWrite(3, LOW);

mySerial.print(LSB, BIN);

digitalWrite(3, HIGH);

mySerial.print(MSB, BIN);

break;

case 'B':

Serial.println("Your output is B");

b1 = 0b0000111, b2 = t2;

r = (b1<<4) + b2;

bitsCount = sizeof( r ) \* 4;

i = 0;

while ( bitsCount-- )

str[ i++ ] = bitRead( r, bitsCount ) + '0';

str[ i ] = '\0';

Serial.print( "MSB =");

Serial.println( str );

Serial.print( "LSM =");

Serial.println( t );

digitalWrite(3, LOW);

mySerial.print(LSB, BIN);

digitalWrite(3, HIGH);

mySerial.print(MSB, BIN);

break;

case 'C':

Serial.println("Your output is C");

b1 = 0b0001011, b2 = t2;

r = (b1<<4) + b2;

bitsCount = sizeof( r ) \* 4;

i = 0;

while ( bitsCount-- )

str[ i++ ] = bitRead( r, bitsCount ) + '0';

str[ i ] = '\0';

Serial.print( "MSB =");

Serial.println( str );

Serial.print( "LSM =");

Serial.println( t );

digitalWrite(3, LOW);

mySerial.print(LSB, BIN);

digitalWrite(3, HIGH);

mySerial.print(MSB, BIN);

break;

case 'D':

Serial.println("Your output is D");

b1 = 0b0001111, b2 = t2;

r = (b1<<4) + b2;

bitsCount = sizeof( r ) \* 4;

i = 0;

while ( bitsCount-- )

str[ i++ ] = bitRead( r, bitsCount ) + '0';

str[ i ] = '\0';

Serial.print( "MSB =");

Serial.println( str );

Serial.print( "LSM =");

Serial.println( t );

digitalWrite(3, LOW);

mySerial.print(LSB, BIN);

digitalWrite(3, HIGH);

mySerial.print(MSB, BIN);

break;

default:

Serial.println("Your output not defined");

break;

}

}