بسم الله الرحمن الرحيم

College of Engineering

Mechanical Engineering Department

Mechatronics Engineering

Graduation Project

Chess Playing Robot

Project Team

Mothanna Al-Shareef Mo'men Neiroukh

Hebron-Palestine

2014-2015

Palestine Polytechnic University

Collage of Engineering

Mechanical Engineering Department

Hebron – Palestine

Chess Playing Robot

Project Team

Mothanna Shareef   Mo'men Neiroukh

According to the directions of the project supervisor and by the agreement of all examination committee members, this project is presented to the department of Mechanical Engineering at College of Engineering, for partial fulfillment Bachelor of engineering degree requirements.

Supervisor Signature

……………………………………….

Testing Committee Signature

…………………………………   ……………………………..…..

Department Head Signature

……………………………………….

2014-2015

Dedication

To our parents

To our families

To our supervisors

To our friends for their support

To our teachers and professors for their help

To our University Palestine Polytechnic University

Abstract

Playing robots is good publicity for Mechatronics .First year student will be interested in Mechatronics after playing with robots and defeated by it. One of the playing robots is chess playing robot. The chess playing robot arm is responsible for moving stones from one position to another. The manipulator should follow a specific path. Depending on data received from chess engine.

In 2010, there was a group that design and implement a chess playing robot. This project consist of two parts: artificial intelligence and robotic manipulator based on SCARA configuration .

This year, the purpose of this project is to use the artificial intelligence and built another robotic manipulator using different technologies.

List Of Tables

List Of Figures

Table Of Content

Chapter 1 Introduction

1.1 Overview

Playing robots is good publicity for Mechatronics .First year student will be interested in Mechatronics after playing with robots and defeated by it. One of the playing robots is chess playing robot. The chess playing robot arm is responsible for moving stones from one position to another. The manipulator should follow a specific path. Depending on data received from chess engine.

In 2010, there was a group that design and implement a chess playing robot. This project consist of two parts: artificial intelligence and robotic manipulator based on SCARA configuration .

This year, the purpose of this project is to use the artificial intelligence and built another robotic manipulator using different technologies.

1.2 Previous Project

1.2.1 Specifications

In the previous project, SCARA robot configuration was used. And DC motors were used for actuation. Also, a gear based gripping mechanism was used for chess pieces gripping. Multi-turns potentiometers were used for feedback. Finally, a DAQ card was used for controlling and interfacing the robotic manipulator. This shown in the following figure.

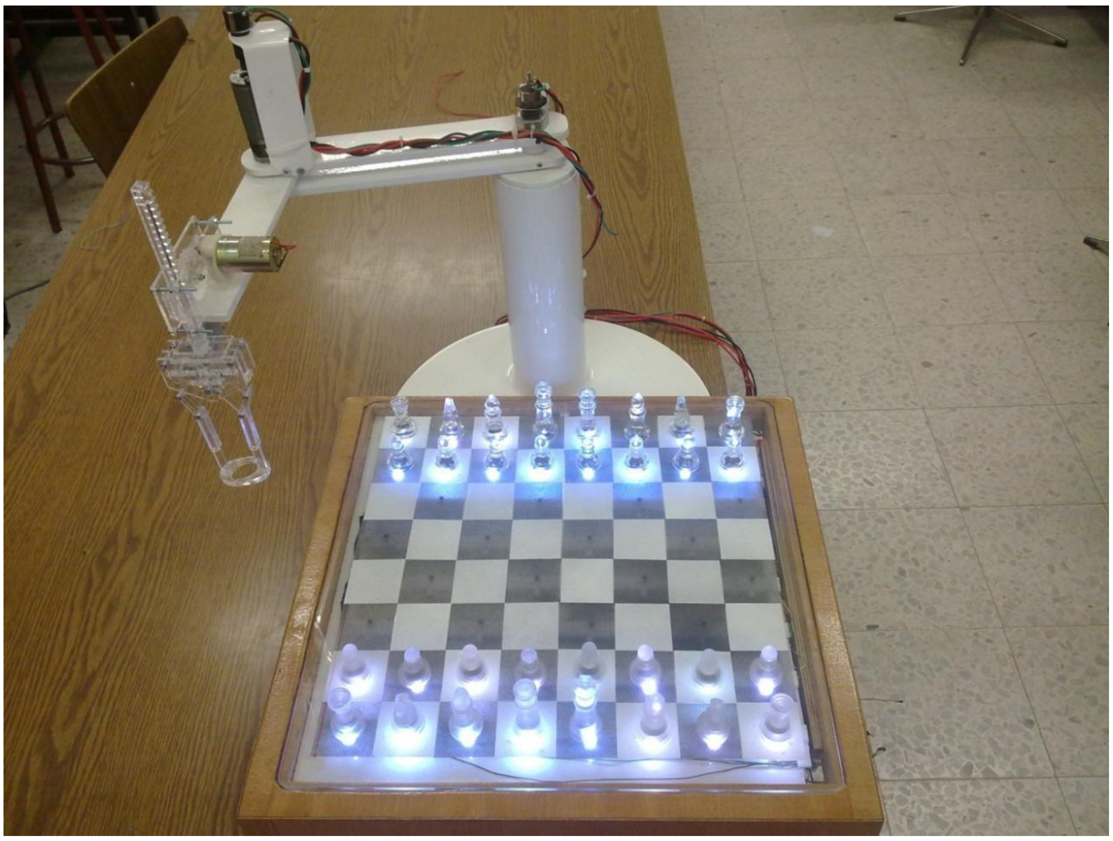


Figure 1.1 Previous Project

When a stone is put on the board, the magnet which is fixed under it will affect the Hall Effect sensor under the board, this will generate a signal that can be read by the keyboard-encoder interfacing card, and then this piece of information will be translated to the chess engine on the PC.

Now it is the turn of the robot to play its game, the chess engine will decide its next movement. This order will be sent through local network to be by the mind of the robot, the MATLAB software, which calculates the motion of the robot arm.

Using MATLAB/Simulink and data acquisition cards (DAQ) these mathematical calculations will be converted to motion, enabling the robot to do the action decided by the chess engine, then the turn of the player will come to repeat the loop.

The monitor of the PC is placed beside the robot; its function is to show the whole process happening on the chessboard. In addition, it shows the countdown timer and the level of difficulty that can be chosen using buttons beside the board.

The timer will give the player a specific time that if exceeded; the player's turn will be ignored and converted to the robot. This interval of time depends on the difficulty level chosen by the user.

Lighting LEDs will interact with the game; they are placed as one LED under each square. The LEDs will tell the user about the possible positions for the hold stone to place on.

This flow of information between the different pieces of the project is shown is figure.

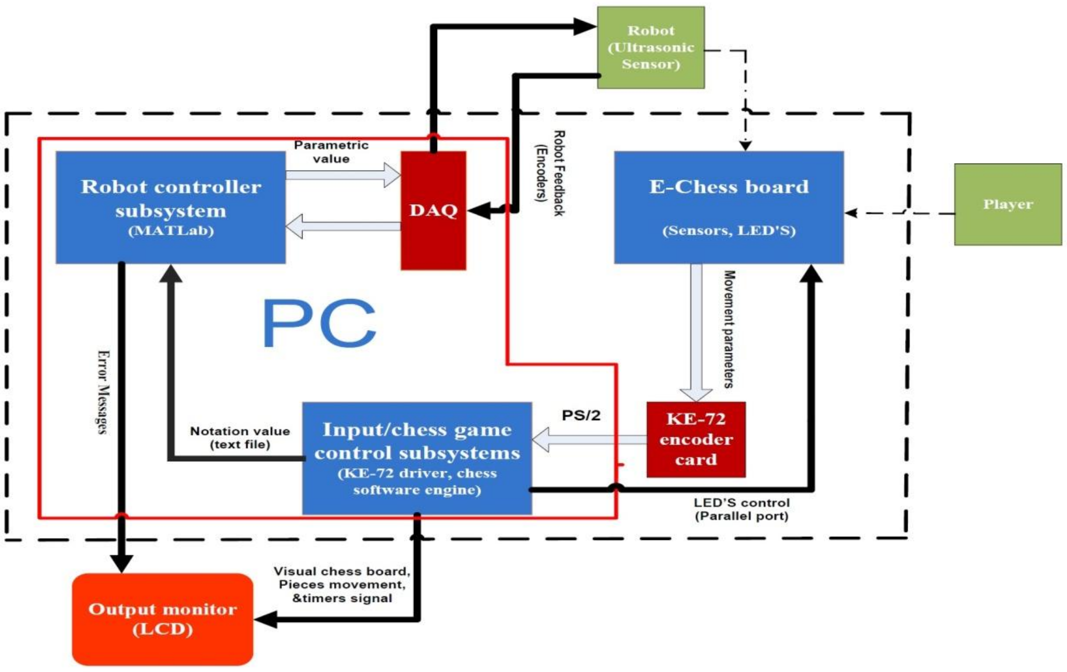

Robot (Ultrasonic Sensor)
Robot Feedback (Encoders)
Parametric value
Robot controller subsystem (MATLab)
DAQ
E-Chess board (Sensors, LED'S)
Player
PC
Movement parameters
Error Messages
Notation value (text file)
Input/chess game control subsystems (KE-72 driver, chess software engine)
PS/2
KE-72 encoder card
LED'S control (Parallel port)
Visual chess board, Pieces movement, &timers signal
Output monitor (LCD)

Figure 1.2 Previous Project Block Diagram

1.2.2 Used Technology

- Chess Board

The used board is a wooden 50cm*50cm with approximately 8cm height stones, it is fixed on a table between the robot and the player, and connected with the computer using the key board encoder KE72 that will be discussed in chapter three in detail, each square in the board contains a sensor that is activated when a stone is put over it.
This switch will generate the signal to the key board encoder KE72.This signal should inform the PC whether there is or there is not a stone in each square.

- Chess Engine/Programs

The used chess software is built using the C#.Net technology, its integrated classes are used to manage the chess board inputs and the intermediate process between MATLAB files and C#.NET.

- MATLAB

MATLAB is used to program the Robots motion, this software acts as a translator, it receives the order from the chess engine and translates it into voltages supplied to the three motors, and thus motion can be performed.

- DAQ Card

DAQ card is used as interfacing card in the project, it is fixed on the PCI slot in the PC and programmed using MATLAB/Simulink, the core function of this card is to convert the analysis done on MATLAB into voltages supplied to the motors, or in other words to connect the Robot with the PC.

- xPC target

The xPC target technique is a solution for prototyping, testing and developing real time systems, using standard PC hardware and its peripherals such as DAQ cards. This technique is a part of MATLAB software provided by MathWorks Company. In particulate xPC is a toolbox with MATLAB/Simulink.
In xPC target technique two PCs are used, host and target. With the host PC, one can design the controller, simulate it and download it to the PC. The target PC which is connected to the controlled plant is just used to run control functions in real time and monitor the controlled application.

1.3 Project Objectives

- Redesign the chess playing robot using simpler and cheaper technology that reduce the complexity of the previous project.
- Since the project team does not have computer engineering student, the work will limited on the robotic manipulator part only.

1.4 Project Report Outline

Chapter 1 : Introduction of project report. Discuss previous project and this project objectives.

Chapter 2 : Project conceptual design. About proposed design and used technology of this project.

Chapter 3 : Hardware design. Presents electrical circuit design of the project.

Chapter 4 : Software design . Discuss the software algorithm of the project code. And the mapping between chess board and motors angels.

Chapter 2 Project Conceptual Design

2.1 Design Requirements

- The ability to pick a chess piece from a certain position then place it in another place.
- The control is an open loop control since the sensors are expensive.

2.2 Design Specifications

SCARA Robot Configuration

A SCARA (Selective Compliance Assembly Robotic Arm) robot is a special subclass of the cylindrical robot family. A SCARA robotic arm allows for 4 degrees of freedom. Freedom in the X-Y plane via two parallel, rotational joints; freedom in the Z plane via one vertical, linear joint, and the freedom to rotate the end effector about the Z axis.

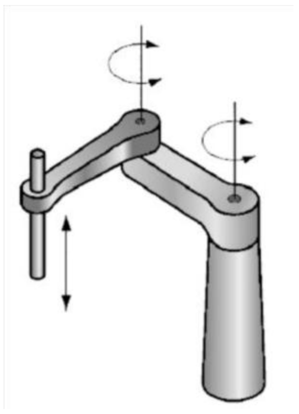


Figure 2.1 SCARA Configuration

The robot must be precise enough to place the chess piece over the hall effect sensor. Since the magnet diameter which is at the bottom of the chess piece is equal to 1.5 cm, the minimum step angle of the stepper motor can be calculated as following:

the length of the total robot arm at maximum extension is 50 cm. And the maximum translational displacement is 0.75 cm "half of the magnet diameter".

Step angle $\theta = \frac{translational\ displacement\ X}{robot\ arm\ length\ R} = \frac{0.75}{50} = 0.015\ rad = 0.9°$

So the minimum step angle should be less than 0.9^0.

Microcontrollers (such as Arduino mega) can work at clock speed up to 16 MHz . This speed enable the robot to work as fast as needed.

since the control is an open loop control, a homing algorithm is needed.

2.3 Project Configuration and Component

2.3.1 Actuators

- Stepper Motors

A stepper motor (or step motor) is a brushless DC electric motor that divides a full rotation into a number of equal steps. The motor's position can then be commanded to move and hold at one of these steps without any feedback sensor (an open-loop controller), as long as the motor is carefully sized to the application.

Bipolar Motor

Bipolar motors have a single winding per phase. The current in a winding needs to be reversed in order to reverse a magnetic pole, so the driving circuit must be more complicated, typically with an H-bridge arrangement (however there are several off-the-shelf driver chips available to make this a simple affair). There are two leads per phase, none are common.

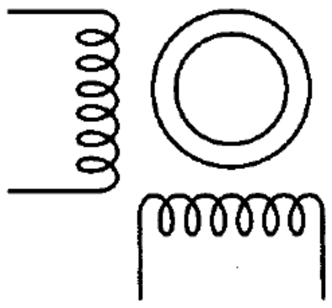


Figure 2.2 Bipolar Motor

2.3.2 Controller

The Arduino Mega is a microcontroller board based on the ATmega1280. It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

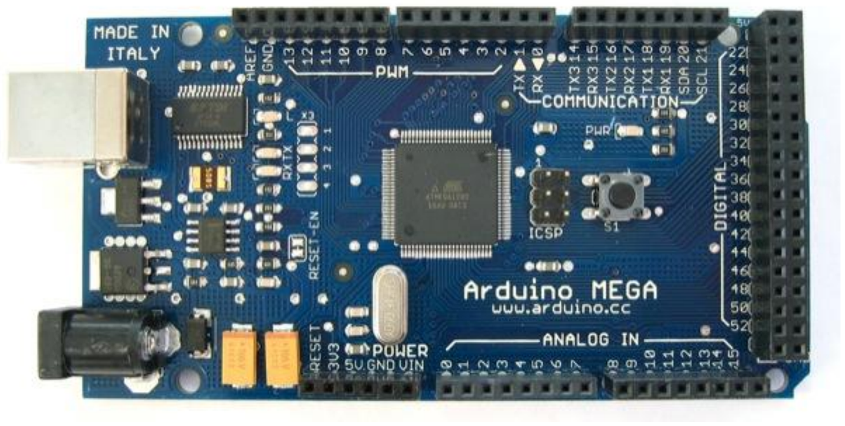


Figure 2.3 Arduino Mega

- Arduino mega specifications

Microcontroller	ATmega1280
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16

DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	128 KB of which 4 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

2.3.3 Stepper motor Driver .

In order to simplify the controlling of stepper motor, a driving circuit is needed to translate the controller digital signal into stepper motor phase voltage. The desired control signal is a clock pulse that determine the magnitude of the angle, and a digital bit that specify the direction of rotation.

Electronics companies provide controller and driver that can divide the step into 16 microstep. Such as A4988 stepper motor driver.

The A4988 stepper motor driver carrier is a breakout board for Allegro's A4988 micro stepping bipolar stepper motor driver. The driver features adjustable current limiting, over current and over temperature protection, and five different microstep resolutions (down to 1/16- step). It operates from 8 – 35 V and can deliver up to approximately 1 A per phase without a heat sink or forced air flow (it is rated for 2 A per coil with sufficient additional cooling).

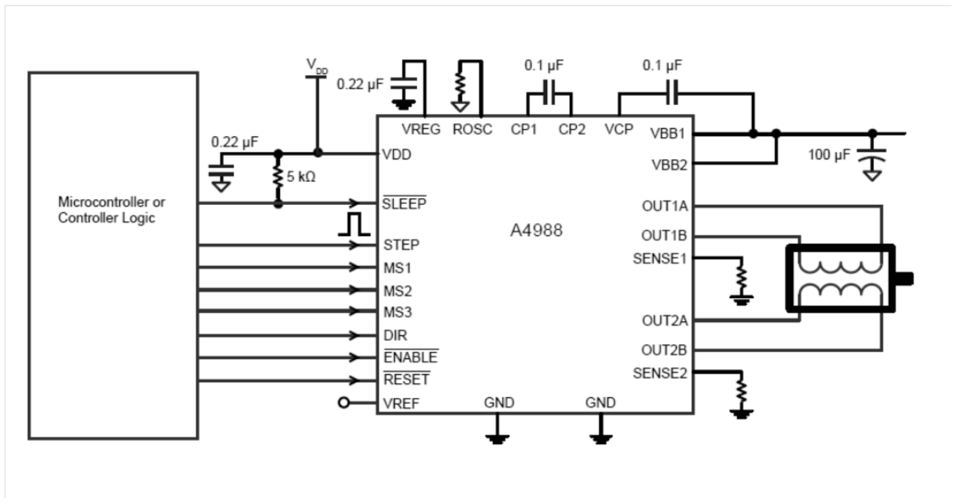


Figure 2.3 stepper motor controller

2.4 System Block Diagram

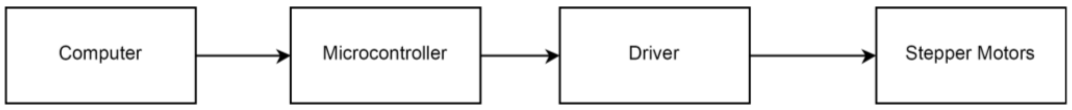


figure 2.4 system block diagram

Computer will be connected to the microcontroller via USB connection to send instructions. the details of the connection and software programming will be presented in the following chapters.

Chapter 3 Hardware Design

3.1 Overview

The robotic manipulator has tow motor to control the position of the end effector. The end effector have tow motor, one for elevation and the other one for gripping mechanism. As illustrated in the following.

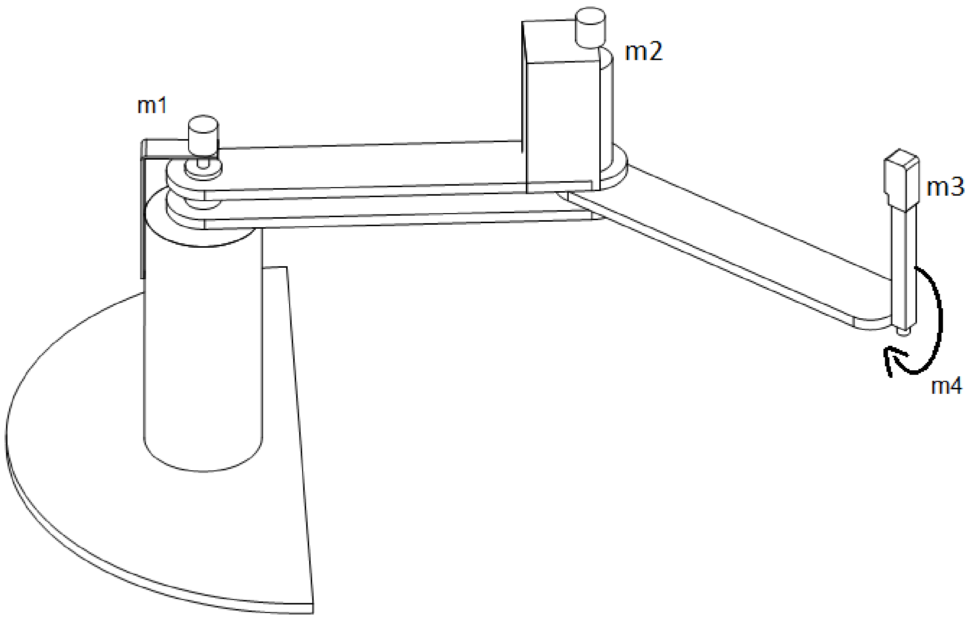


figure 3.1 project robot arm

This will be controlled using Microcontroller. A stepper motor driver is to be used as driving circuit for all motors. Since the robot is to be used for chess game, a connection with computer is needed. Arduino Mega microcontroller is support USB Connection.

The control of the stepper motor could be open loop or close loop control . An open loop control is going to be used since the sensors for closed loop is expensive. Because of that a homing process is needed each time the robot is operating. So a limit switch will placed for each motor of the position motors.

All of that will be illustrated in the following block diagram.

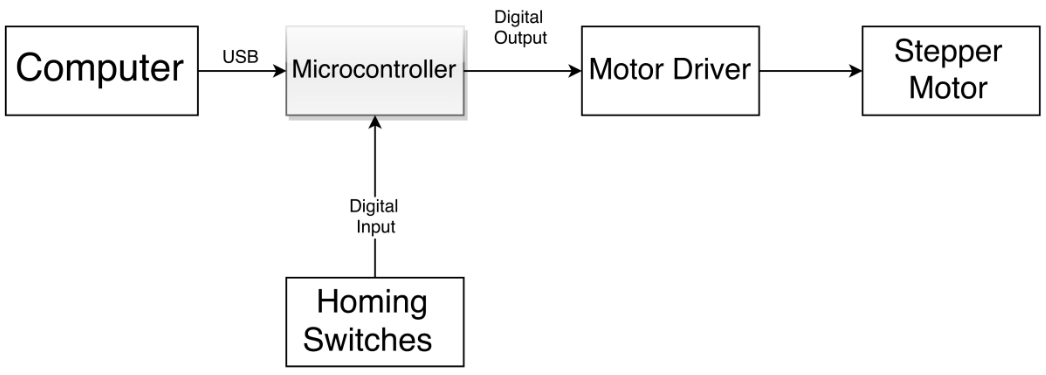


figure 3.2 detailed block diagram

3.2 Driver Connection

For each stepper motor used in the project, a stepper motor driver is used. from datasheet, the wiring connection of the driver is as following:

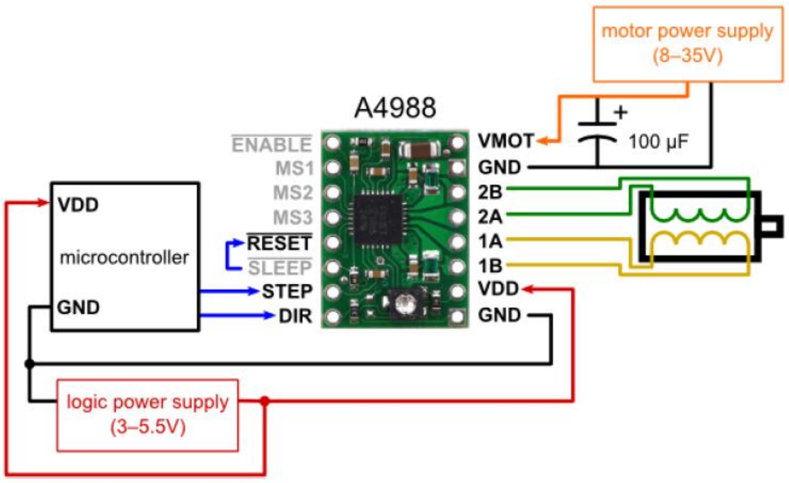
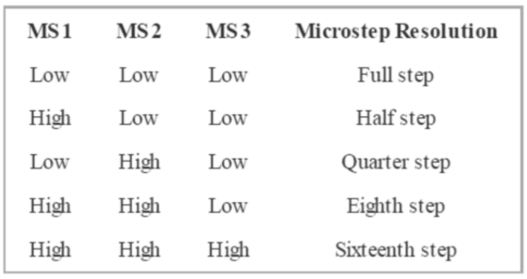


figure 3.3 driver schematic

Driver Control



MS1	MS2	MS3	Microstep Resolution
Low	Low	Low	Full step
High	Low	Low	Half step
Low	High	Low	Quarter step
High	High	Low	Eighth step
High	High	High	Sixteenth step

table 3.1 micro stepping control

Control signals

Each pulse to the STEP input corresponds to one microstep of the stepper motor in the direction selected by the DIR pin. Note that the STEP and DIR pins are not pulled to any particular voltage internally, so you should not leave either of these pins floating in your application. If you just want rotation in a single direction, you can tie DIR directly to VCC or GND. The chip has three different inputs for controlling its many power states: RST, SLP, and EN. For details about these power states, see the datasheet. Please note that the RST pin is floating; if you are not using the pin, you can connect it to the adjacent SLP pin on the PCB to bring it high and enable the board.

Experimentally, best micro stepping connection that gives smoothest move with enough torque was quarter step. Also it achieves the maximum calculated value of a single step. since the selected stepper motor step is 1.8^o, the quarter of that is 0.45^o. So MS1 and MS3 is low and MS2 is high.

3.3 Homing sensor connection

Because of the using of open loop control approach there is no position sensor. instead of that a homing process using limit switch is used. A button is placed for each position motor for homing. After each move a homing process is performed. Each switch is connected so that it gives digital signal to the micro controller .The following circuit shows how the limit switch is connected to the microcontroller.

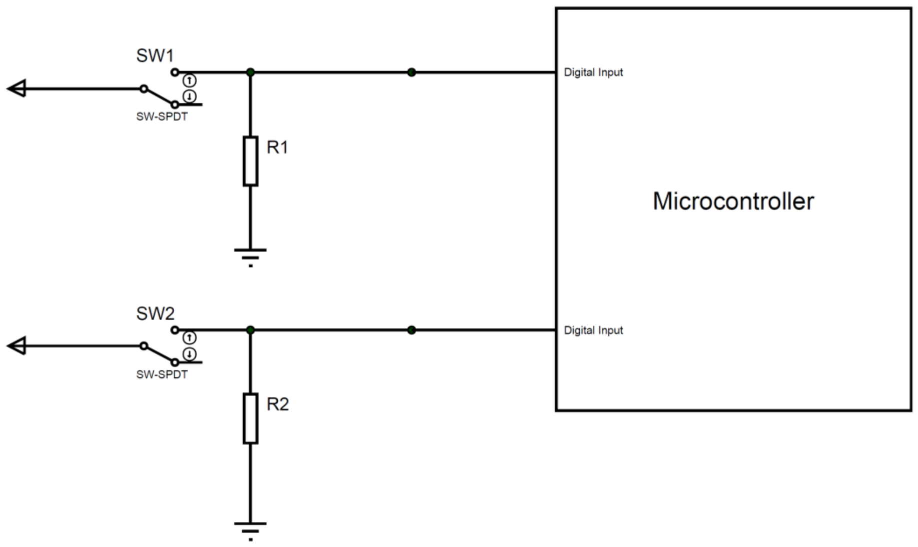


figure 3.4 limit switch circuit

As shown in the previous circuit. if the motor reaches its home position the limit switch send high digital signal to the motor, otherwise it sends low.

3.4 Project Electrical Circuit

In the previous sections the connections between microcontroller, drivers, homing switches and stepper motor have been discussed. In this section the full electrical circuit of the project will be illustrated.

The driver require two control signal, one for magnitude and other for direction. Both signal are digital signal. The magnitude signal is a series of steps that produce the required position. The direction signal either zero or one witch determine the direction of rotation, clockwise or counterclockwise.

Stepper motor driver require bipolar stepper motor connection. So four wires from the drivers is connected to the stepper motor.

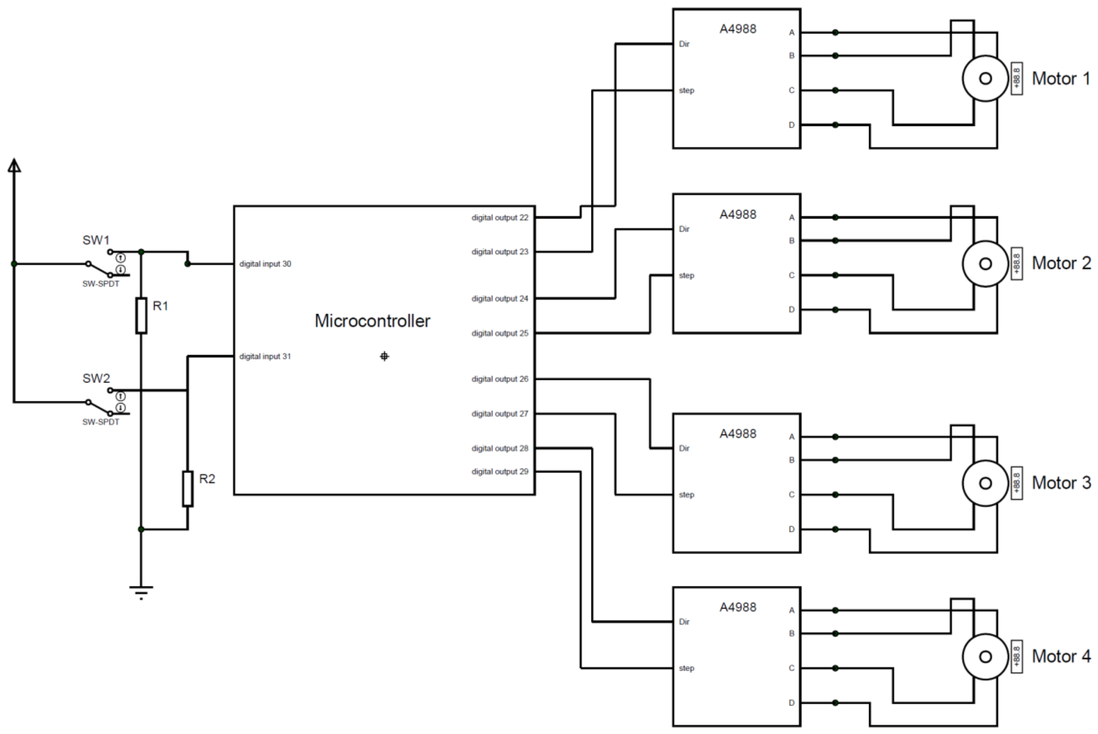


figure 3.5 project circuit diagram

Chapter 4 Software Design

4.1 Overview

The serial communication is used In sending the orders to the arduino. This method will give us the privilege to abandon the use of matlab and use the same computer to send the desired position to the arduino ,which will calculate the needed pulses to reach that position. After reaching the desired position the gripper and the linear actuator start to work.

The arduino allow us to make the homing process. The homing process need to start after two move . and then wait until a new desired position arrived via serial communication.

Using c++ code for arduino helps to satisfy the desired requirement. Using the different functions; help to control each and every part of the robot and makes the navigation in the code easier.

4.2 Software Flowchart

The simplest move of the robot is to pick a chess piece and place it in another position. Any move in chess game is like this move or a combination of moves like this.

the program procedure as following: moving motors to home position, then moving motors to initial position, the pick up the chess piece, after that moving motor to the final position, finally place the chess piece. The program flowchart is illustrated in the following chart.

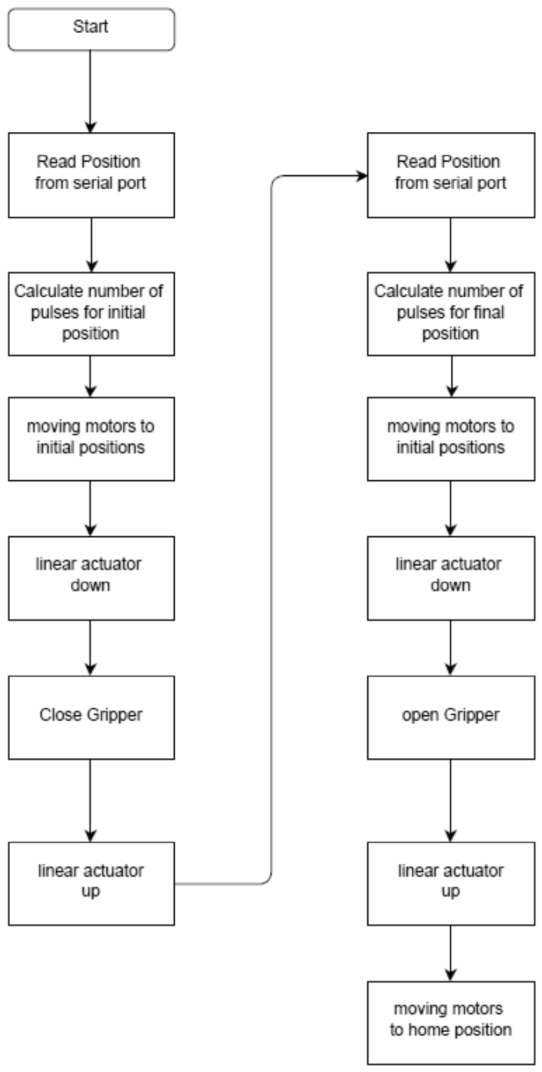

Start
Read Position from serial port
Calculate number of pulses for initial position
moving motors to initial positions
linear actuator down
Close Gripper
linear actuator up
Read Position from serial port
Calculate number of pulses for final position
moving motors to initial positions
linear actuator down
open Gripper
linear actuator up
moving motors to home position

figure 4.1 software flow chart

4.3 Detailed software programming

In this section, each part of the software will be discussed and explained. the total program code will be in the appendix A.

Before presenting the flowchart, a certain parameter should be explained. Mi represent the magnitude of the desired motor position and Di represent the direction. mi and di represent respectively the digital output of the microcontroller to the stepper motor driver.

S1 and S2 represent the digital signal of the limit switches of the homing function. The delay is used to generate digital clock that used to move the stepper motor. The delay time is differ from function to another depending on the speed required in that function.

4.3.1 Motor moving function

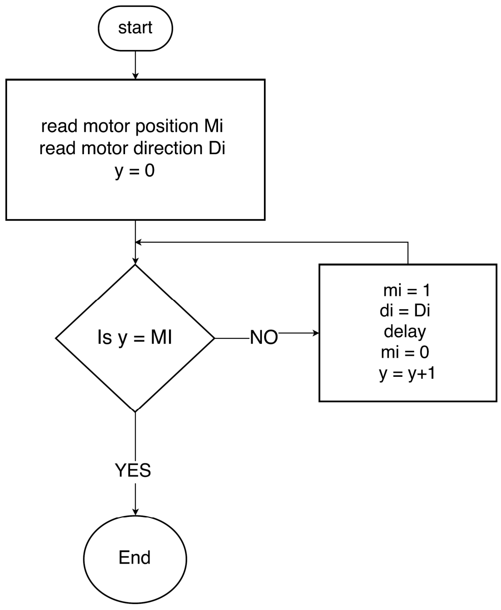


figure 4.2 motor moving function algorithm

this flowchart is translated into the following C++ code :

```
void gomotors(int c,int v){
    delay(10);
      for(int y = 0;y < c;y++){
          digitalWrite(m2,HIGH);
          if(y>=v){
          digitalWrite(m1,LOW);
           }
           else
          {
                 digitalWrite(m1,HIGH);
          }
        delay(20);
        digitalWrite(m1,LOW);
        digitalWrite(m2,LOW);
         delay(20);
     }
}

void gomotors1(int c,int v){
  delay(10);
  for(int y = 0;y < c;y++){
    digitalWrite(m1,HIGH);
    if(y>=v){
      digitalWrite(m2,LOW);
    }
    else
    {
      digitalWrite(m2,HIGH);
    }
    delay(20);
    digitalWrite(m1,LOW);
    digitalWrite(m2,LOW);
    delay(20);
  }
}
```

4.3.2 End Effector Function

The end effector operate within four functions: up, down, grip and release. These function is illustrated in tow flowcharts, one for gripping and another for releasing as following.

Gripping flowchart

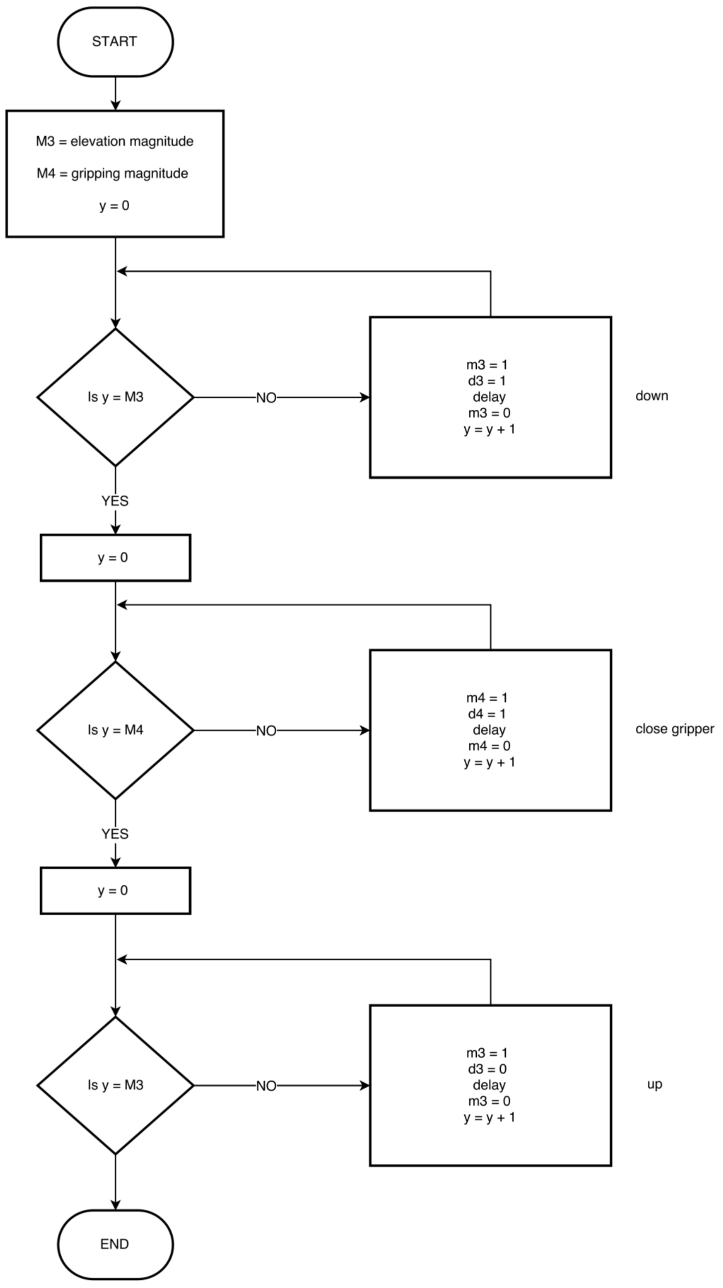


figure 4.3 End Effector gripping function algorithm

Release Flowchart

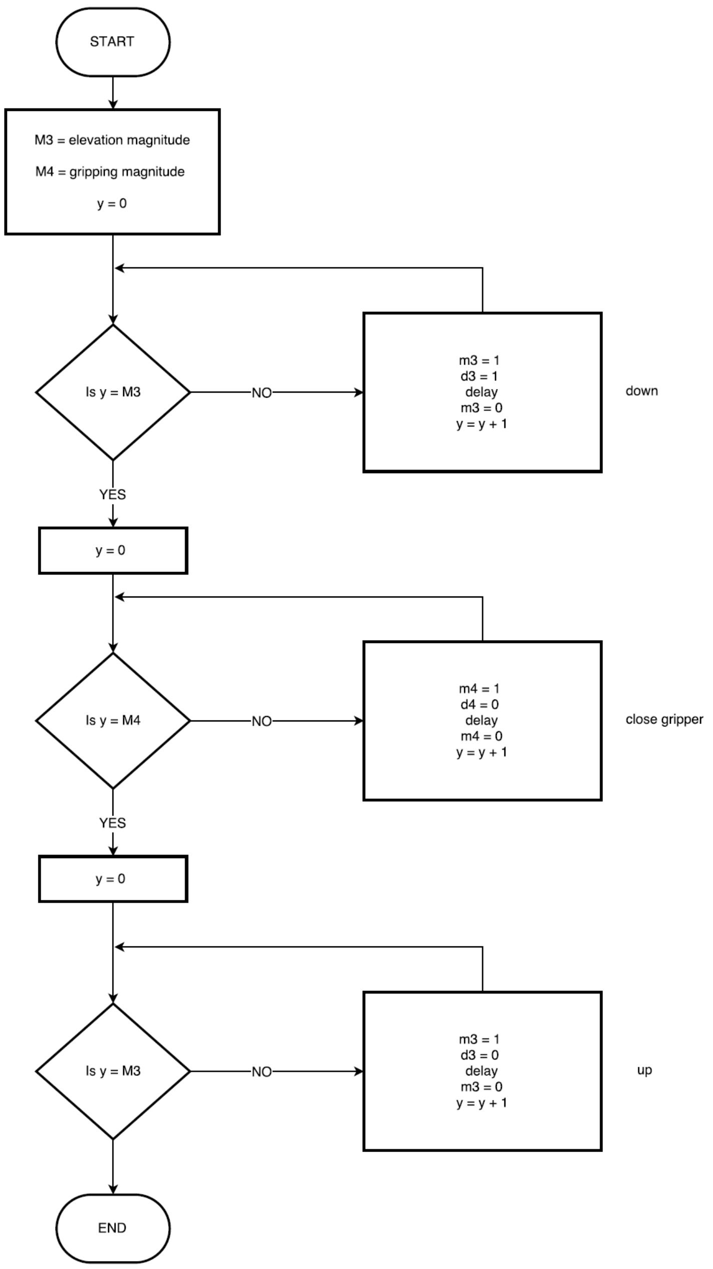


figure 4.4 End Effector Release function algorithm

these flowchart is translated into the following code.

```
void openg(){
  digitalWrite(d3,HIGH);
  delay(10);
  for(int y=0;y<50;y++){
    digitalWrite(m3,HIGH);
    delay(20);
    digitalWrite(m3,LOW);
    delay(20);
  }
  digitalWrite(d3,LOW);
}


void closeg(){
  digitalWrite(d3,LOW);
  delay(10);
  for(int y=0;y<50;y++){
    digitalWrite(m3,HIGH);
    delay(20);
    digitalWrite(m3,LOW);
    delay(20);
  }
}


void upl(){
  digitalWrite(d4,HIGH);
  delay(10);
  for(int y=0;y<180;y++){
    digitalWrite(m4,HIGH);
    delay(20);
    digitalWrite(m4,LOW);
    delay(20);
  }
  digitalWrite(d4,LOW);
}


void downl(){
  digitalWrite(d4,LOW);
  delay(10);
  for(int y=0;y<180;y++){
    digitalWrite(m4,HIGH);
    delay(20);
    digitalWrite(m4,LOW);
    delay(20);
  }
}
```

4.3.3 Homing function

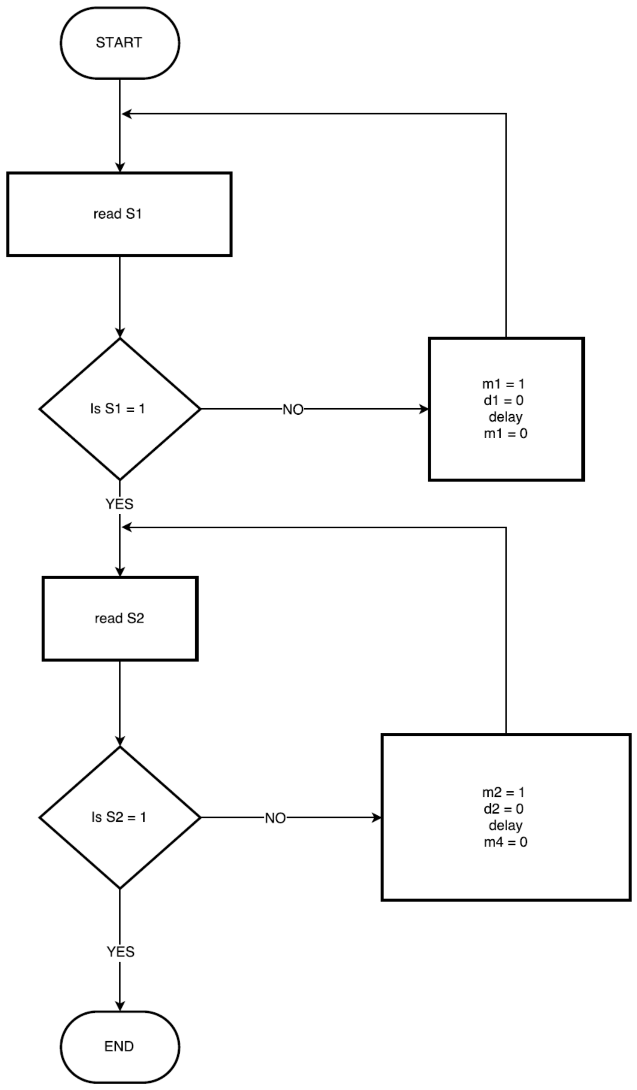


figure 4.5 Homing function algorithm

these flowchart is translated into the following code.

```
void gomotorsback(){
  b=1;
  while(b)
  {
   digitalWrite(d2,LOW);
   digitalWrite(d1,LOW);
```

```
    digitalWrite(m1,HIGH);
    digitalWrite(m2,HIGH);
    delay(20);
    if(!digitalRead(s2)){
      digitalWrite(m1,LOW);
    }
    if(!digitalRead(s1)){
    digitalWrite(m2,LOW);
    }
    delay(20);
    if(digitalRead(s1)&&digitalRead(s2)){
    b=0;
    }
   }
}
```

4.4 Mapping Between Chess Positions and Motors Angles.

Each position in the chess board can be reached by moving the two arms. so each position should be related to two angles.

Using the catia program the angles can be calculated as shown in the following figure:

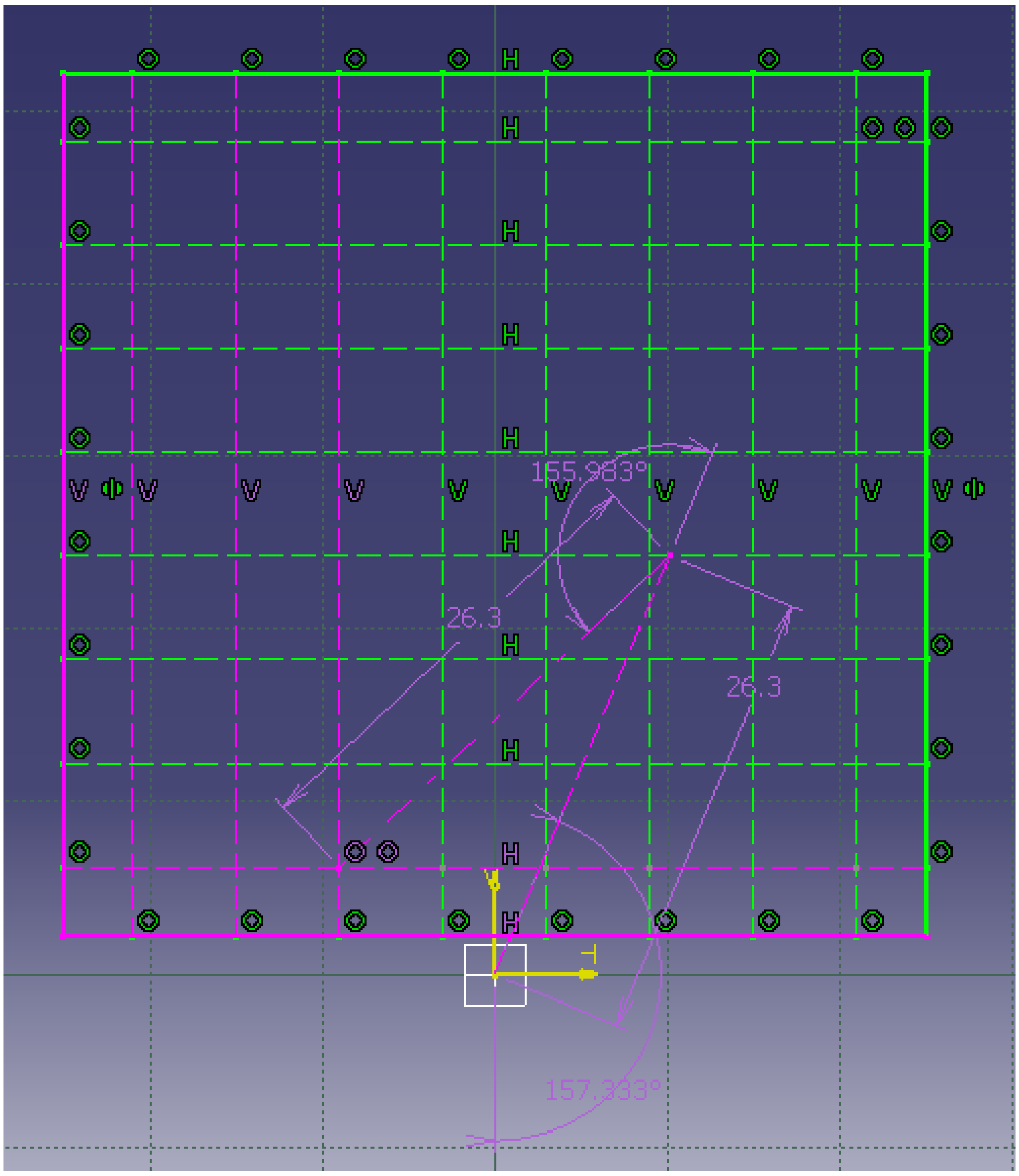


figure 4.6 mapping using catia

and the results were as follow :

		A	B	C	D	E	F	G	H
1	θ_1	41.123	40.527	46.684	71.823	123.277	157.333	175.442	188.094
	θ_2	220.783	234.031	245.983	254.9	254.9	245.983	234.031	220.783
2	θ_1	57.722	60.769	70.418	90.06	117.634	143.133	162.403	177.307
	θ_2	214.971	226.828	236.449	242.306	242.306	236.449	226.828	214.971
3	θ_1	72.869	77.21	86.454	101.212	119.906	139.014	156.119	170.958
	θ_2	206.173	216.671	224.533	228.883	228.883	224.533	216.671	206.173
4	θ_1	86.635	91.036	99.048	110.592	124.711	139.808	154.565	168.475
	θ_2	194.89	204.4	211.144	214.697	214.697	211.144	204.4	194.89
5	θ_1	99.614	103.511	110.258	119.604	130.941	143.421	156.298	169.195
	θ_2	181.191	190.191	196.321	199.455	199.455	196.321	190.191	181.191
6	θ_1	112.655	115.702	121.254	128.981	138.449	149.158	160.688	172.856
	θ_2	164.489	173.61	179.588	182.571	182.571	179.588	173.61	164.489
7	θ_1	127.269	128.865	133.134	139.527	147.655	157.198	167.978	179.846
	θ_2	142.577	153.157	159.668	162.818	162.818	159.668	153.157	142.577
8	θ_1	156.469	146.498	148.285	153.165	160.285	169.427	178.947	203.531
	θ_2	90	122.45	132.288	136.55	136.55	132.288	122.45	90

table 4.1 mapping results in degree using catia

The previous results are represented in degree. after converting them into pulses form, considering that each step is 0.45^0. The mapping become ass following :

		A	B	C	D	E	F	G	H
1	θ_1	91.4	90.1	104	160	274	350	390	418
	θ_2	491	520	547	566	566	547	520	491
2	θ_1	128	135	156	200	261	318	361	394
	θ_2	478	504	525	538	538	525	504	478
3	θ_1	162	172	192	225	266	309	347	380
	θ_2	458	481	499	509	509	499	481	458
4	θ_1	193	202	220	246	277	311	343	374
	θ_2	433	454	469	477	477	469	454	433
5	θ_1	221	230	245	266	291	319	347	376
	θ_2	403	423	436	443	443	436	423	403
6	θ_1	250	257	269	287	308	331	357	384
	θ_2	366	386	399	406	406	399	386	366
7	θ_1	283	286	296	310	328	349	373	400
	θ_2	317	340	355	362	362	355	340	317
8	θ_1	348	326	330	340	356	377	398	452
	θ_2	200	272	294	303	303	294	272	200

table 4.2 mapping results in pulses using catia

Finally, because the two links are not identical and the references are different, the actual angle were evaluated by trial and error as following :

for second motor

490	505	520	535	545	535	520	490
470	485	495	513	520	513	495	470
435	455	475	495	485	485	470	460
415	435	475	475	454	454	430	433
380	403	420	425	435	425	400	370
345	350	360	395	380	350	355	320
280	305	300	335	330	315	295	280
197	197	190	190	215	205	215	215

table 4.3 mapping results in pulses using trial and error "second motor"

for center motor

55	65	85	115	175	235	275	305
85	95	115	140	170	220	260	290
105	125	145	170	190	210	250	270
125	145	140	170	210	230	260	270
155	160	150	205	215	235	260	290
175	185	205	205	235	260	270	300
215	215	230	230	260	270	290	310
260	272	292	305	315	320	330	350

table 4.4 mapping results in pulses using trial and error "center motor"

Appendices

Appendix 1

Project Software

```
int m1=22;
int d1=23;
int m2=24;
int d2=25;
int m3=26;
int d3=27;
int m4=28;
int d4=29;
int m=0;
int x1=0;
int x2=0;
int M1[]={

490,505,520,535,545,535,520,490,470,485,495,513,520,513,495,470,435,455,475,495,485,48
5,470,460,415,435,475,475,454,454,430,433,380,403,420,425,435,425,400,370,345,350,360,
395,380,350,355,320,280,305,300,335,330,315,295,280,197,197,190,190,215,205,215,215};
int M2[]={

55,65,85,115,175,235,275,305,85,95,115,140,170,220,260,290,105,125,145,170,190,210,250
,270,125,145,140,170,210,230,260,270,155,160,150,205,215,235,260,290,175,185,205,205,2
35,260,270,300,215,215,230,230,260,270,290,310,260,272,292,305,315,320,330,350};
int m1p=0;
int m2p=0;
int m1t=0;
int m2t=0;
int mm=0;
int b=0;
int s1=30;
int s2=31;
void setup}()

  pinMode(m1,OUTPUT؛(

  pinMode(d1,OUTPUT؛(

  pinMode(m2,OUTPUT؛(

  pinMode(d2,OUTPUT؛(

  pinMode(m3,OUTPUT؛(

  pinMode(d3,OUTPUT؛(

  pinMode(m4,OUTPUT؛(

  pinMode(d4,OUTPUT؛(

  pinMode(s1,INPUT؛(

  pinMode(s2,INPUT؛(

  Serial.begin(9600 ؛(


{

void loop}()
```

```
  for(int x=0;x<2;x}(++

    m1p=0؛

    m2p=0؛

    while(Serial.available()==0 } (

{

    m=Serial.read؛()

    delay(500؛(

    Serial.flush؛()

    delay(1000؛(

    while(Serial.available()==0 } (

{


//    Serial.println("send second؛("

    n=Serial.read؛()

    Serial.flush؛()

    m=m-48؛

    n=n-48؛

    m=m*10؛

    mm=m+n؛


    delay(500؛(

    x1=M1[mm؛[

    x2=M2[mm ؛[

    delay(500؛(


    if(x1>m1p}(

      m1t=x1-m1p؛

      digitalWrite(d1,HIGH؛(

{

    else

}

      m1t=m1p-x1؛
      digitalWrite(d1,LOW:(
{
    if(x2>m2p}(
      m2t=x2-m2p:
      digitalWrite(d2,HIGH:(
{
    else
}
      m2t=m2p-x2:
      digitalWrite(d2,LOW:(
{
    if(m1t>m2t}(
      gomotors(m1t,m2t:(
{
    else
}
      gomotors1(m2t,m1t :(
{
    if(x==0}(

      downl:()
      closeg:()
      upl:()
{
    if(x==1}(
       downl:()
       openg:()
       upl:()



{
   gomotorsback:()
{
```

```
{

void gomotors(int c,int v}(
  delay(10:(
  for(int y = 0;y < c;y}(++
    digitalWrite(m2,HIGH:(
    if(y>=v}(
      digitalWrite(m1,LOW:(
{
    else
}
      digitalWrite(m1,HIGH:(
{
    delay(20:(
    digitalWrite(m1,LOW:(
    digitalWrite(m2,LOW:(
    delay(20:(
{
{

void gomotors1(int c,int v}(
  delay(10:(
  for(int y = 0;y < c;y}(++
    digitalWrite(m1,HIGH:(
    if(y>=v}(
      digitalWrite(m2,LOW:(
{
    else
}
      digitalWrite(m2,HIGH:(
{
    delay(20:(
    digitalWrite(m1,LOW؛(
    digitalWrite(m2,LOW؛(
    delay(20؛(
{
{

void gomotorsback}()
  b=1؛
  while(b(
}
   digitalWrite(d2,LOW؛(
   digitalWrite(d1,LOW؛(
   digitalWrite(m1,HIGH؛(
   digitalWrite(m2,HIGH؛(
   delay(20؛(
   if(!digitalRead(s2}((
     digitalWrite(m1,LOW؛(
{
   if(!digitalRead(s1}((
   digitalWrite(m2,LOW؛(
{
   delay(20؛(
   if(digitalRead(s1)&&digitalRead(s2}((
   b=0؛
{
{
{



void openg}()
  digitalWrite(d3,HIGH؛(
  delay(10؛(
  for(int y=0;y<50;y}(++
    digitalWrite(m3,HIGH؛(
    delay(20:(
    digitalWrite(m3,LOW:(
    delay(20:(
{
  digitalWrite(d3,LOW:(
{

void closeg}()
  digitalWrite(d3,LOW:(
  delay(10:(
  for(int y=0;y<50;y}(++
    digitalWrite(m3,HIGH:(
    delay(20:(
    digitalWrite(m3,LOW:(
    delay(20:(
{
{
void upl}()
  digitalWrite(d4,HIGH:(
  delay(10:(
  for(int y=0;y<180;y}(++
    digitalWrite(m4,HIGH:(
    delay(20:(
    digitalWrite(m4,LOW:(
    delay(20:(
{
  digitalWrite(d4,LOW:(
{
void downl}()
  digitalWrite(d4,LOW:(
  delay(10:(
  for(int y=0;y<180;y}(++
    digitalWrite(m4,HIGH:(
    delay(20:(
    digitalWrite(m4,LOW);(
    delay(20);(
{
}
```

Appendix 2

Hardware Datasheets

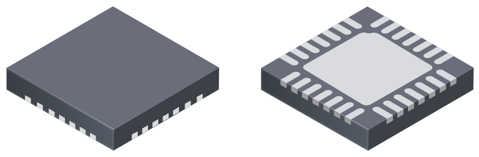
DMOS Microstepping Driver with Translator And Overcurrent Protection

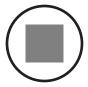
Features and Benefits

- Low $R_{DS(ON)}$ outputs
- Automatic current decay mode detection/selection
- Mixed and Slow current decay modes
- Synchronous rectification for low power dissipation
- Internal UVLO
- Crossover-current protection
- 3.3 and 5 V compatible logic supply
- Thermal shutdown circuitry
- Short-to-ground protection
- Shorted load protection
- Five selectable step modes: full, $^1/_2$, $^1/_4$, $^1/_8$, and $^1/_{16}$

Package:

28-contact QFN
with exposed thermal pad
5 mm × 5 mm × 0.90 mm
(ET package)





Approximate size

Description

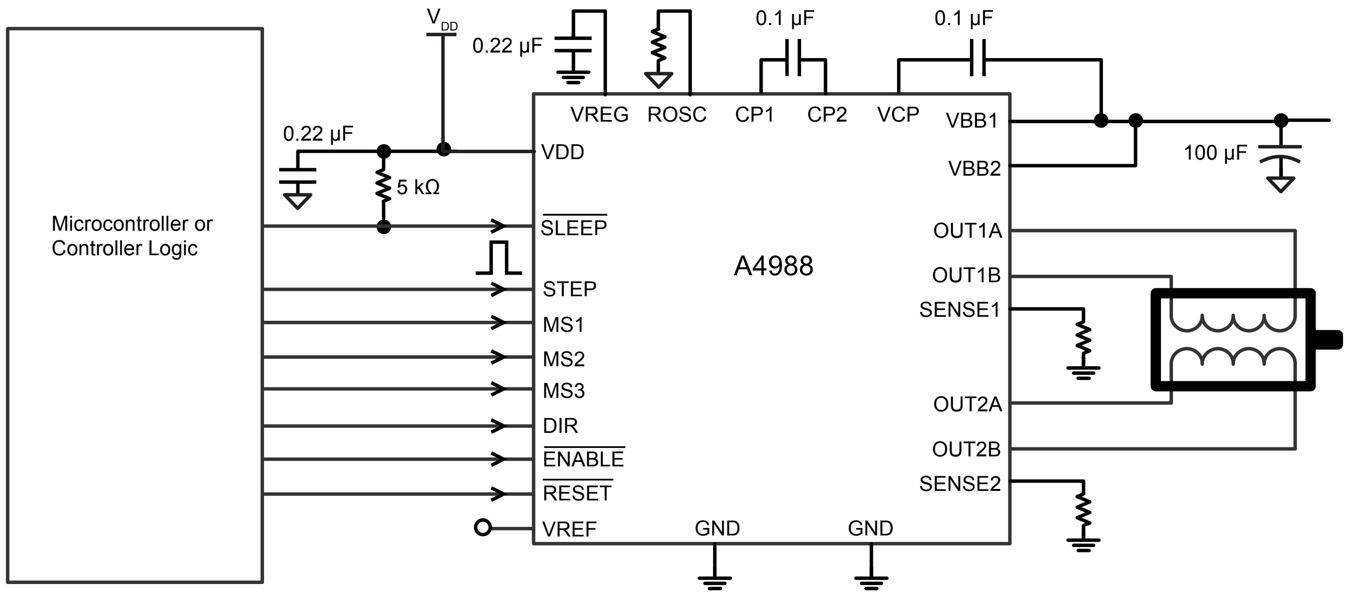
The A4988 is a complete microstepping motor driver with built-in translator for easy operation. It is designed to operate bipolar stepper motors in full-, half-, quarter-, eighth-, and sixteenth-step modes, with an output drive capacity of up to 35 V and ±2 A. The A4988 includes a fixed off-time current regulator which has the ability to operate in Slow or Mixed decay modes.

The translator is the key to the easy implementation of the A4988. Simply inputting one pulse on the STEP input drives the motor one microstep. There are no phase sequence tables, high frequency control lines, or complex interfaces to program. The A4988 interface is an ideal fit for applications where a complex microprocessor is unavailable or is overburdened.

During stepping operation, the chopping control in the A4988 automatically selects the current decay mode, Slow or Mixed. In Mixed decay mode, the device is set initially to a fast decay for a proportion of the fixed off-time, then to a slow decay for the remainder of the off-time. Mixed decay current control results in reduced audible motor noise, increased step accuracy, and reduced power dissipation.

Continued on the next page…

Typical Application Diagram



Description (continued)

Internal synchronous rectification control circuitry is provided to improve power dissipation during PWM operation. Internal circuit protection includes: thermal shutdown with hysteresis, undervoltage lockout (UVLO), and crossover-current protection. Special power-on sequencing is not required.

The A4988 is supplied in a surface mount QFN package (ES), 5 mm × 5 mm, with a nominal overall package height of 0.90 mm and an exposed pad for enhanced thermal dissipation. It is lead (Pb) free (suffix –T), with 100% matte tin plated leadframes.

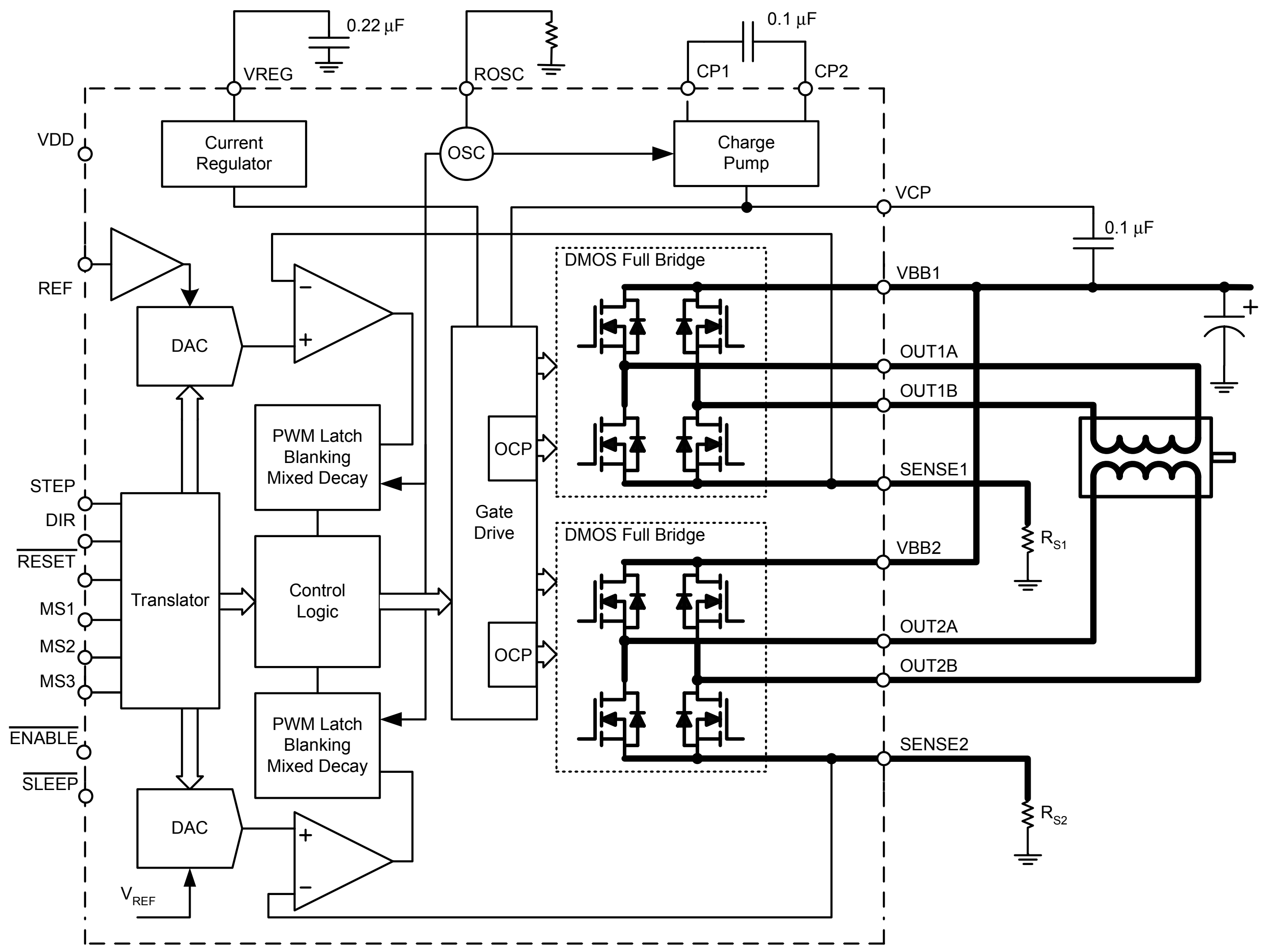
Selection Guide

Part Number	Package	Packing
A4988SETTR-T	28-contact QFN with exposed thermal pad	1500 pieces per 7-in. reel

Absolute Maximum Ratings

Characteristic	Symbol	Notes	Rating	Units
Load Supply Voltage	V_{BB}		35	V
Output Current	I_{OUT}		±2	A
Logic Input Voltage	V_{IN}		–0.3 to 5.5	V
Logic Supply Voltage	V_{DD}		–0.3 to 5.5	V
Motor Outputs Voltage			–2.0 to 37	V
Sense Voltage	V_{SENSE}		–0.5 to 0.5	V
Reference Voltage	V_{REF}		5.5	V
Operating Ambient Temperature	T_A	Range S	–20 to 85	ºC
Maximum Junction	T_J(max)		150	ºC
Storage Temperature	T_{stg}		–55 to 150	ºC

Functional Block Diagram



ELECTRICAL CHARACTERISTICS[1] at T_A = 25°C, V_{BB} = 35 V (unless otherwise noted)

Characteristics	Symbol	Test Conditions	Min.	Typ.[2]	Max.	Units
Output Drivers						
Load Supply Voltage Range	V_{BB}	Operating	8	–	35	V
Logic Supply Voltage Range	V_{DD}	Operating	3.0	–	5.5	V
Output On Resistance	R_{DSON}	Source Driver, I_{OUT} = –1.5 A	–	320	430	mΩ
		Sink Driver, I_{OUT} = 1.5 A	–	320	430	mΩ
Body Diode Forward Voltage	V_F	Source Diode, I_F = –1.5 A	–	–	1.2	V
		Sink Diode, I_F = 1.5 A	–	–	1.2	V
Motor Supply Current	I_{BB}	f_{PWM} < 50 kHz	–	–	4	mA
		Operating, outputs disabled	–	–	2	mA
Logic Supply Current	I_{DD}	f_{PWM} < 50 kHz	–	–	8	mA
		Outputs off	–	–	5	mA
Control Logic						
Logic Input Voltage	$V_{IN(1)}$		V_{DD}×0.7	–	–	V
	$V_{IN(0)}$		–	–	V_{DD}×0.3	V
Logic Input Current	$I_{IN(1)}$	V_{IN} = V_{DD}×0.7	–20	<1.0	20	µA
	$I_{IN(0)}$	V_{IN} = V_{DD}×0.3	–20	<1.0	20	µA
Microstep Select	R_{MS1}	MS1 pin	–	100	–	kΩ
	R_{MS2}	MS2 pin	–	50	–	kΩ
	R_{MS3}	MS3 pin	–	100	–	kΩ
Logic Input Hysteresis	$V_{HYS(IN)}$	As a % of V_{DD}	5	11	19	%
Blank Time	t_{BLANK}		0.7	1	1.3	µs
Fixed Off-Time	t_{OFF}	OSC = VDD or GND	20	30	40	µs
		R_{OSC} = 25 kΩ	23	30	37	µs
Reference Input Voltage Range	V_{REF}		0	–	4	V
Reference Input Current	I_{REF}		–3	0	3	µA
Current Trip-Level Error[3]	err_I	V_{REF} = 2 V, $\%I_{TripMAX}$ = 38.27%	–	–	±15	%
		V_{REF} = 2 V, $\%I_{TripMAX}$ = 70.71%	–	–	±5	%
		V_{REF} = 2 V, $\%I_{TripMAX}$ = 100.00%	–	–	±5	%
Crossover Dead Time	t_{DT}		100	475	800	ns
Protection						
Overcurrent Protection Threshold[4]	I_{OCPST}		2.1	–	–	A
Thermal Shutdown Temperature	T_{TSD}		–	165	–	°C
Thermal Shutdown Hysteresis	T_{TSDHYS}		–	15	–	°C
VDD Undervoltage Lockout	V_{DDUVLO}	V_{DD} rising	2.7	2.8	2.9	V
VDD Undervoltage Hysteresis	$V_{DDUVLOHYS}$		–	90	–	mV

[1]For input and output current specifications, negative current is defined as coming out of (sourcing) the specified device pin.

[2]Typical data are for initial design estimations only, and assume optimum manufacturing and application conditions. Performance may vary for individual units, within the specified maximum and minimum limits.

[3]$V_{ERR} = [(V_{REF}/8) – V_{SENSE}] / (V_{REF}/8)$.

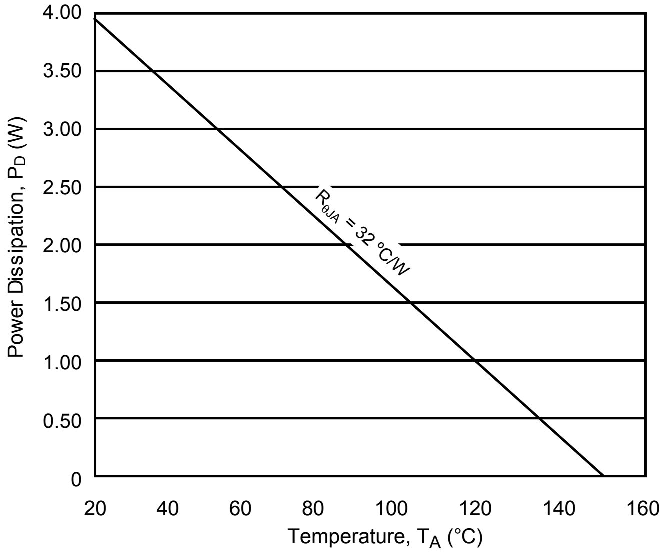
[4]Overcurrent protection (OCP) is tested at T_A = 25°C in a restricted range and guaranteed by characterization.

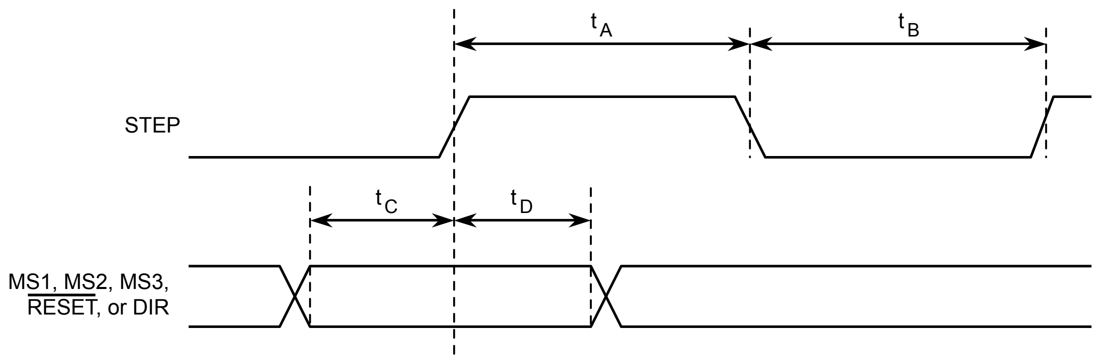
THERMAL CHARACTERISTICS

Characteristic	Symbol	Test Conditions*	Value	Units
Package Thermal Resistance	$R_{\theta JA}$	Four-layer PCB, based on JEDEC standard	32	ºC/W

*Additional thermal information available on Allegro Web site.

Power Dissipation versus Ambient Temperature





Time Duration	Symbol	Typ.	Unit
STEP minimum, HIGH pulse width	t_A	1	μs
STEP minimum, LOW pulse width	t_B	1	μs
Setup time, input change to STEP	t_C	200	ns
Hold time, input change to STEP	t_D	200	ns

Figure 1: Logic Interface Timing Diagram

Table 1: Microstepping Resolution Truth Table

MS1	MS2	MS3	Microstep Resolution	Excitation Mode
L	L	L	Full Step	2 Phase
H	L	L	Half Step	1-2 Phase
L	H	L	Quarter Step	W1-2 Phase
H	H	L	Eighth Step	2W1-2 Phase
H	H	H	Sixteenth Step	4W1-2 Phase

Functional Description

Device Operation. The A4988 is a complete microstepping motor driver with a built-in translator for easy operation with minimal control lines. It is designed to operate bipolar stepper motors in full-, half-, quarter-, eighth, and sixteenth-step modes. The currents in each of the two output full-bridges and all of the N-channel DMOS FETs are regulated with fixed off-time PWM (pulse width modulated) control circuitry. At each step, the current for each full-bridge is set by the value of its external current-sense resistor (R_{S1} and R_{S2}), a reference voltage (V_{REF}), and the output voltage of its DAC (which in turn is controlled by the output of the translator).

At power-on or reset, the translator sets the DACs and the phase current polarity to the initial Home state (shown in Figures 9 through 13), and the current regulator to Mixed Decay Mode for both phases. When a step command signal occurs on the STEP input, the translator automatically sequences the DACs to the next level and current polarity. (See Table 2 for the current-level sequence.) The microstep resolution is set by the combined effect of the MSx inputs, as shown in Table 1.

When stepping, if the new output levels of the DACs are lower than their previous output levels, then the decay mode for the active full-bridge is set to Mixed. If the new output levels of the DACs are higher than or equal to their previous levels, then the decay mode for the active full-bridge is set to Slow. This automatic current decay selection improves microstepping performance by reducing the distortion of the current waveform that results from the back EMF of the motor.

Microstep Select (MSx). The microstep resolution is set by the voltage on logic inputs MSx, as shown in Table 1. The MS1 and MS3 pins have a 100 kΩ pull-down resistance, and the MS2 pin has a 50 kΩ pull-down resistance. When changing the step mode the change does not take effect until the next STEP rising edge.

If the step mode is changed without a translator reset, and absolute position must be maintained, it is important to change the step mode at a step position that is common to both step modes in order to avoid missing steps. When the device is powered down, or reset due to TSD or an over current event the translator is set to the home position which is by default common to all step modes.

Mixed Decay Operation. The bridge operates in Mixed decay mode, at power-on and reset, and during normal running according to the ROSC configuration and the step sequence, as shown in Figures 9 through 13. During Mixed decay, when the trip point is reached, the A4988 initially goes into a fast decay mode for 31.25% of the off-time, t_{OFF}. After that, it switches to Slow decay mode for the remainder of t_{OFF}. A timing diagram for this feature appears on the next page.

Typically, mixed decay is only necessary when the current in the winding is going from a higher value to a lower value as determined by the state of the translator. For most loads automatically-selected mixed decay is convenient because it minimizes ripple when the current is rising and prevents missed steps when the current is falling. For some applications where microstepping at very low speeds is necessary, the lack of back EMF in the winding causes the current to increase in the load quickly, resulting in missed steps. This is shown in Figure 2. By pulling the ROSC pin to ground, mixed decay is set to be active 100% of the time, for both rising and falling currents, and prevents missed steps as shown in Figure 3. If this is not an issue, it is recommended that automatically-selected mixed decay be used, because it will produce reduced ripple currents. Refer to the Fixed Off-Time section for details.

Low Current Microstepping. Intended for applications where the minimum on-time prevents the output current from regulating to the programmed current level at low current steps. To prevent this, the device can be set to operate in Mixed decay mode on both rising and falling portions of the current waveform. This feature is implemented by shorting the ROSC pin to ground. In this state, the off-time is internally set to 30 µs.

Reset Input ($\overline{\text{RESET}}$). The $\overline{\text{RESET}}$ input sets the translator to a predefined Home state (shown in Figures 9 through 13), and turns off all of the FET outputs. All STEP inputs are ignored until the $\overline{\text{RESET}}$ input is set to high.

Step Input (STEP). A low-to-high transition on the STEP

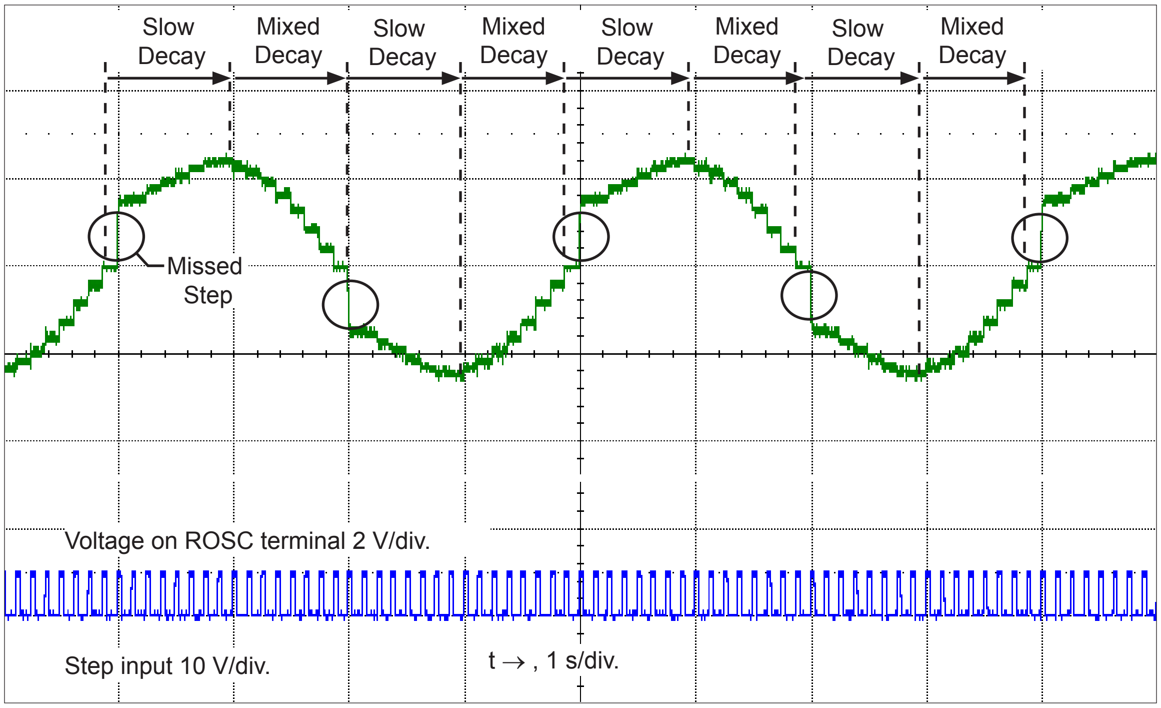


Figure 2: Missed Steps in Low-Speed Microstepping

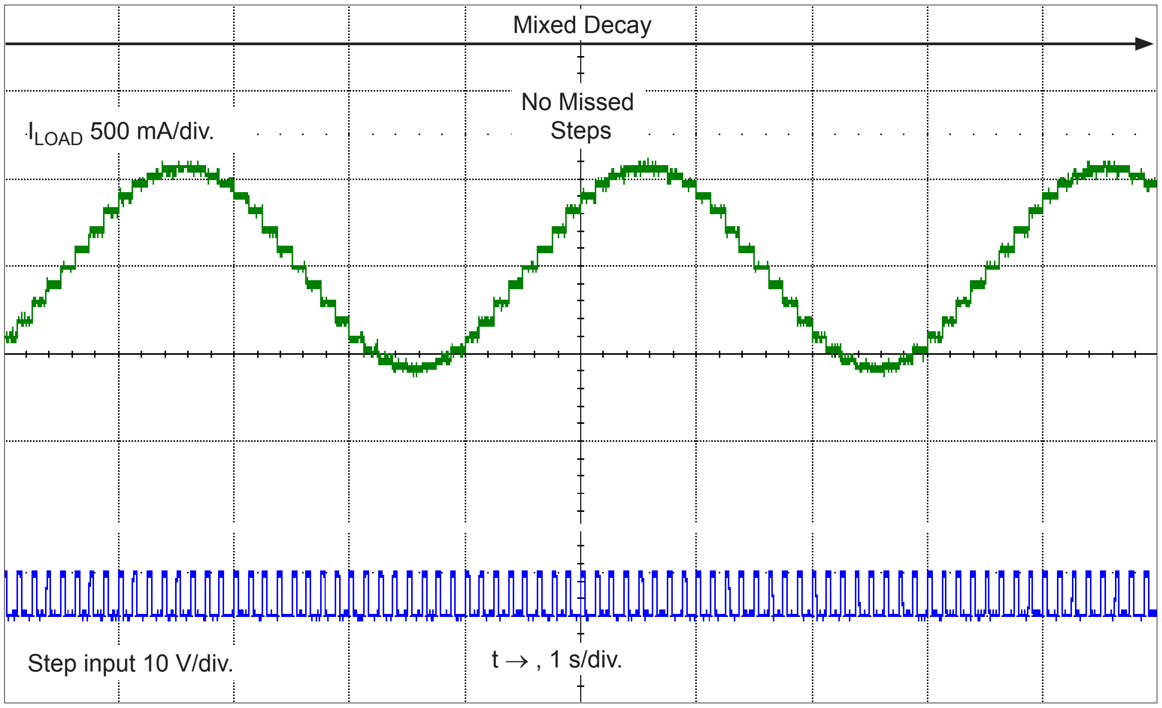


Figure 3: Continuous Stepping Using Automatically-Selected Mixed Stepping (ROSC pin grounded)

input sequences the translator and advances the motor one increment. The translator controls the input to the DACs and the direction of current flow in each winding. The size of the increment is determined by the combined state of the MSx inputs.

Direction Input (DIR). This determines the direction of rotation of the motor. Changes to this input do not take effect until the next STEP rising edge.

Internal PWM Current Control. Each full-bridge is controlled by a fixed off-time PWM current control circuit that limits the load current to a desired value, I_{TRIP}. Initially, a diagonal pair of source and sink FET outputs are enabled and current flows through the motor winding and the current sense resistor, R_{Sx}. When the voltage across R_{Sx} equals the DAC output voltage, the current sense comparator resets the PWM latch. The latch then turns off the appropriate source driver and initiates a fixed off time decay mode

The maximum value of current limiting is set by the selection of R_{Sx} and the voltage at the VREF pin. The transconductance function is approximated by the maximum value of current limiting, $I_{TripMAX}$ (A), which is set by

$$I_{TripMAX} = V_{REF} / (8 \times R_S)$$

where R_S is the resistance of the sense resistor (Ω) and V_{REF} is the input voltage on the REF pin (V).

The DAC output reduces the V_{REF} output to the current sense comparator in precise steps, such that

$$I_{trip} = (\%I_{TripMAX} / 100) \times I_{TripMAX}$$

(See Table 2 for $\%I_{TripMAX}$ at each step.)

It is critical that the maximum rating (0.5 V) on the SENSE1 and SENSE2 pins is not exceeded.

Fixed Off-Time. The internal PWM current control circuitry uses a one-shot circuit to control the duration of time that the DMOS FETs remain off. The off-time, t_{OFF}, is determined by the ROSC terminal. The ROSC terminal has three settings:

- ROSC tied to VDD — off-time internally set to 30 µs, decay mode is automatic Mixed decay except when in full step where decay mode is set to Slow decay
- ROSC tied directly to ground — off-time internally set to 30 µs, current decay is set to Mixed decay for both increasing and decreasing currents for all step modes.
- ROSC through a resistor to ground — off-time is determined by the following formula, the decay mode is automatic Mixed decay for all step modes except full step whic is set to slow decay.

$$t_{OFF} \approx R_{OSC} / 825$$

Where t_{OFF} is in µs.

Blanking. This function blanks the output of the current sense comparators when the outputs are switched by the internal current control circuitry. The comparator outputs are blanked to prevent false overcurrent detection due to reverse recovery currents of the clamp diodes, and switching transients related to the capacitance of the load. The blank time, t_{BLANK} (µs), is approximately

$$t_{BLANK} \approx 1\ \mu s$$

Shorted-Load and Short-to-Ground Protection. If the motor leads are shorted together, or if one of the leads is shorted to ground, the driver will protect itself by sensing the overcurrent event and disabling the driver that is shorted, protecting the device from damage. In the case of a short-to-ground, the device will remain disabled (latched) until the $\overline{\text{SLEEP}}$ input goes high or VDD power is removed. A short-to-ground overcurrent event is shown in Figure 4.

When the two outputs are shorted together, the current path is through the sense resistor. After the blanking time (≈1 µs) expires, the sense resistor voltage is exceeding its trip value, due to the overcurrent condition that exists. This causes the driver to go into a fixed off-time cycle. After the fixed off-time expires the driver turns on again and the process repeats. In this condition the driver is completely protected against overcurrent events, but the short is repetitive with a period equal to the fixed off-time of the driver. This condition is shown in Figure 5.

During a shorted load event it is normal to observe both a positive and negative current spike as shown in Figure 3, due to the direction change implemented by the Mixed decay feature. This is shown in Figure 6. In both instances the overcurrent circuitry is protecting the driver and prevents damage to the device.

Charge Pump (CP1 and CP2). The charge pump is used to generate a gate supply greater than that of VBB for driving the source-side FET gates. A 0.1 µF ceramic capacitor, should be connected between CP1 and CP2. In addition, a 0.1 µF ceramic capacitor is required between VCP and VBB, to act as a reservoir for operating the high-side FET gates.

Capacitor values should be Class 2 dielectric ±15% maximum, or tolerance R, according to EIA (Electronic Industries Alliance) specifications.

V_{REG} (VREG). This internally-generated voltage is used to operate the sink-side FET outputs. The nominal output voltage of the VREG terminal is 7 V. The VREG pin must be decoupled with a 0.22 µF ceramic capacitor to ground. V_{REG} is internally monitored. In the case of a fault condition, the FET outputs of the A4988 are disabled.

Capacitor values should be Class 2 dielectric ±15% maximum, or tolerance R, according to EIA (Electronic Industries Alliance) specifications.

Enable Input ($\overline{\text{ENABLE}}$). This input turns on or off all of the FET outputs. When set to a logic high, the outputs are disabled. When set to a logic low, the internal control enables the outputs as required. The translator inputs STEP, DIR, and MSx, as well as the internal sequencing logic, all remain active, independent of the $\overline{\text{ENABLE}}$ input state.

Shutdown. In the event of a fault, overtemperature (excess T_J) or an undervoltage (on VCP), the FET outputs of the A4988 are disabled until the fault condition is removed. At power-on, the UVLO (undervoltage lockout) circuit disables the FET outputs and resets the translator to the Home state.

Sleep Mode ($\overline{\text{SLEEP}}$). To minimize power consumption when the motor is not in use, this input disables much of the internal circuitry including the output FETs, current regulator, and charge pump. A logic low on the $\overline{\text{SLEEP}}$ pin puts the A4988 into Sleep mode. A logic high allows normal operation, as well as start-up (at which time the A4988 drives the motor to the Home microstep position). When emerging from Sleep mode, in order to allow the charge pump to stabilize, provide a delay of 1 ms before issuing a Step command.

Mixed Decay Operation. The bridge operates in Mixed Decay mode, depending on the step sequence, as shown in Figures 9 through 13. As the trip point is reached, the A4988 initially goes into a fast decay mode for 31.25% of the off-time, t_{OFF}. After that, it switches to Slow Decay mode for the remainder of t_{OFF}. A timing diagram for this feature appears in Figure 7.

Synchronous Rectification. When a PWM-off cycle is triggered by an internal fixed-off time cycle, load current recirculates according to the decay mode selected by the control logic. This synchronous rectification feature turns on the appropriate FETs during current decay, and effectively shorts out the body diodes with the low FET $R_{DS(ON)}$. This reduces power dissipation significantly, and can eliminate the need for external Schottky diodes in many applications. Synchronous rectification turns off when the load current approaches zero (0 A), preventing reversal of the load current.

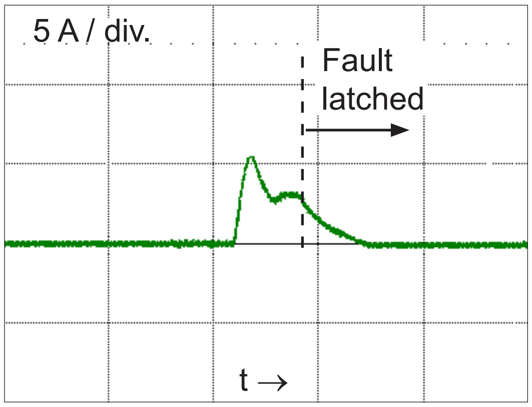


Figure 4: Short-to-Ground Event

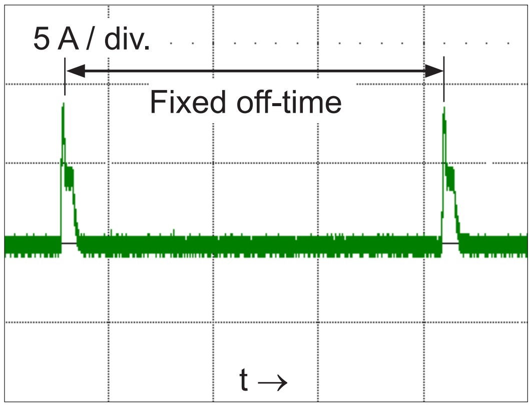


Figure 5. Shorted Load (OUTxA → OUTxB) in Slow Decay Mode

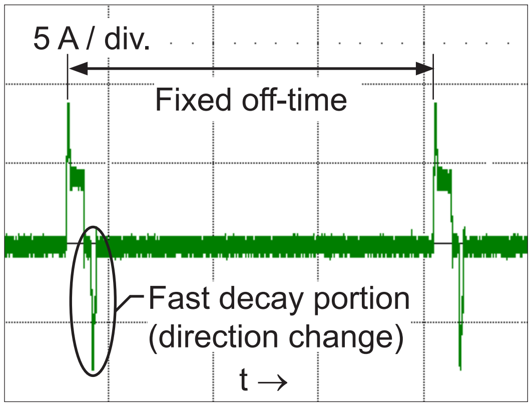
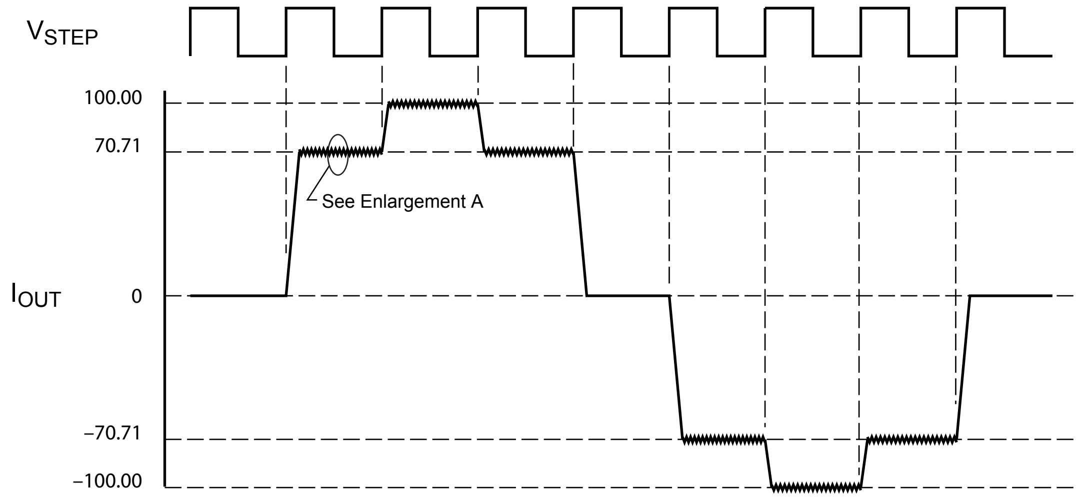
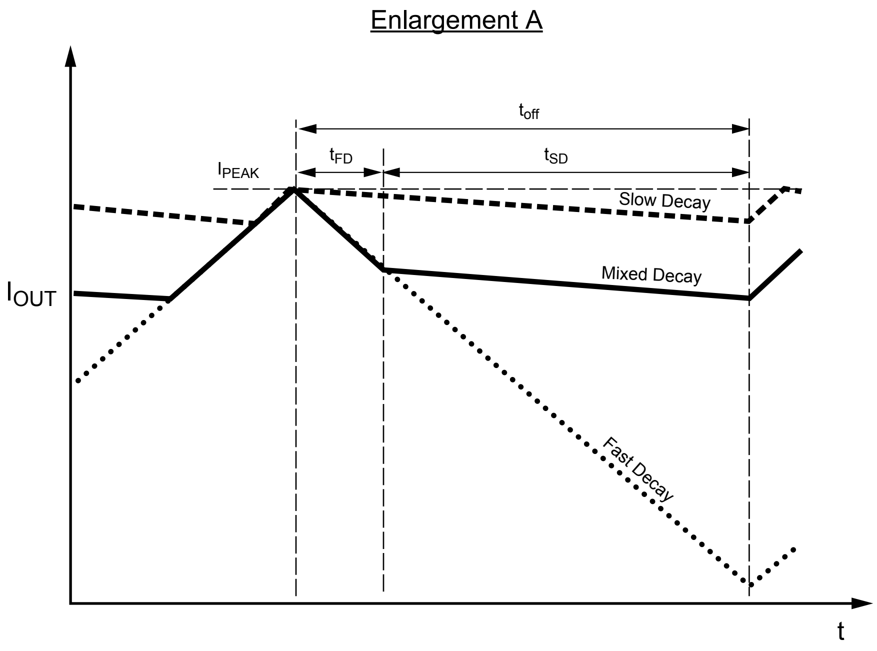


Figure 6: Shorted Load (OUTxA → OUTxB) in Mixed Decay Mode





Symbol	Characteristic
t_{off}	Device fixed off-time
I_{PEAK}	Maximum output current
t_{SD}	Slow decay interval
t_{FD}	Fast decay interval
I_{OUT}	Device output current

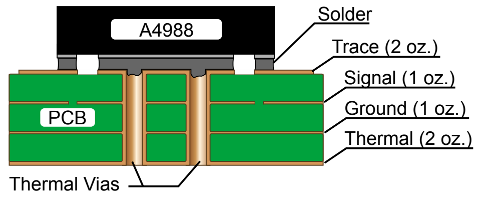
Figure 7: Current Decay Modes Timing Chart

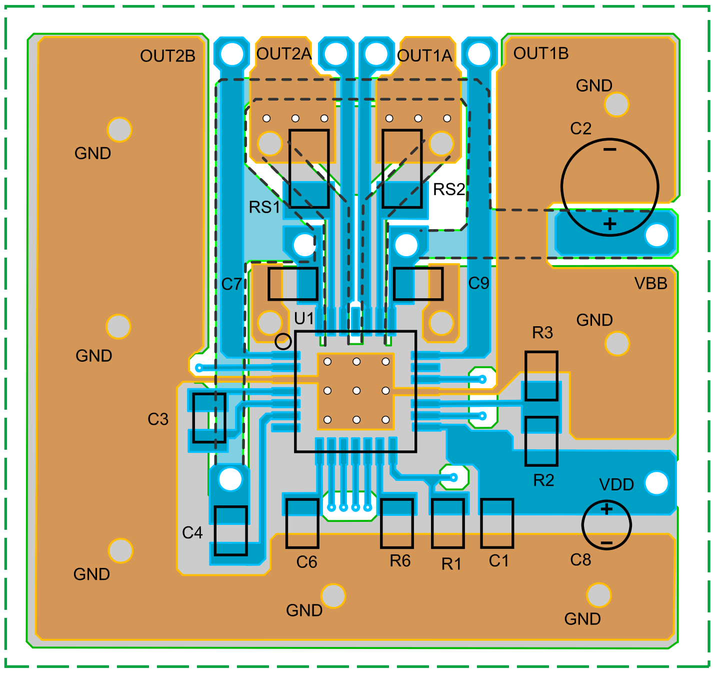
Application Layout

Layout. The printed circuit board should use a heavy groundplane. For optimum electrical and thermal performance, the A4988 must be soldered directly onto the board. Pins 3 and 18 are internally fused, which provides a path for enhanced thermal dissipation. Theses pins should be soldered directly to an exposed surface on the PCB that connects to thermal vias are used to transfer heat to other layers of the PCB.

In order to minimize the effects of ground bounce and offset issues, it is important to have a low impedance single-point ground, known as a *star ground*, located very close to the device. By making the connection between the pad and the ground plane directly under the A4988, that area becomes an ideal location for a star ground point. A low impedance ground will prevent ground bounce during high current operation and ensure that the supply voltage remains stable at the input terminal.

The two input capacitors should be placed in parallel, and as close to the device supply pins as possible. The ceramic capacitor (CIN1) should be closer to the pins than the bulk capacitor (CIN2). This is necessary because the ceramic capacitor will be responsible for delivering the high frequency current components. The sense resistors, RSx, should have a very low impedance path to ground, because they must carry a large current while supporting very accurate voltage measurements by the current sense comparators. Long ground traces will cause additional voltage drops, adversely affecting the ability of the comparators to accurately measure the current in the windings. The SENSEx pins have very short traces to the RSx resistors and very thick, low impedance traces directly to the star ground underneath the device. If possible, there should be no other components on the sense circuits.





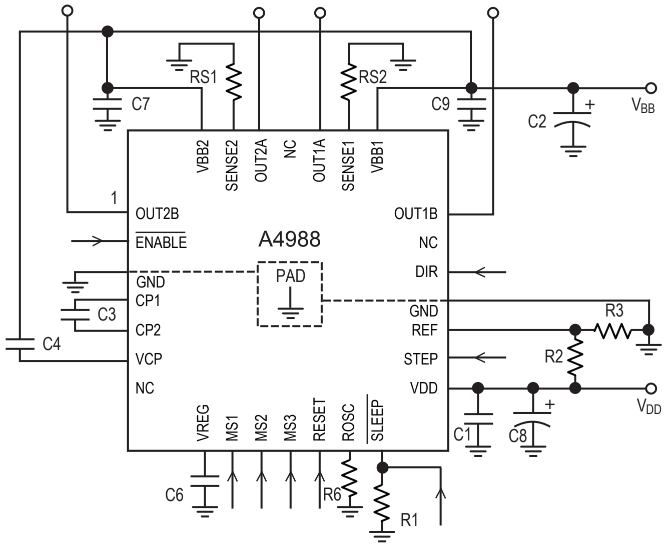
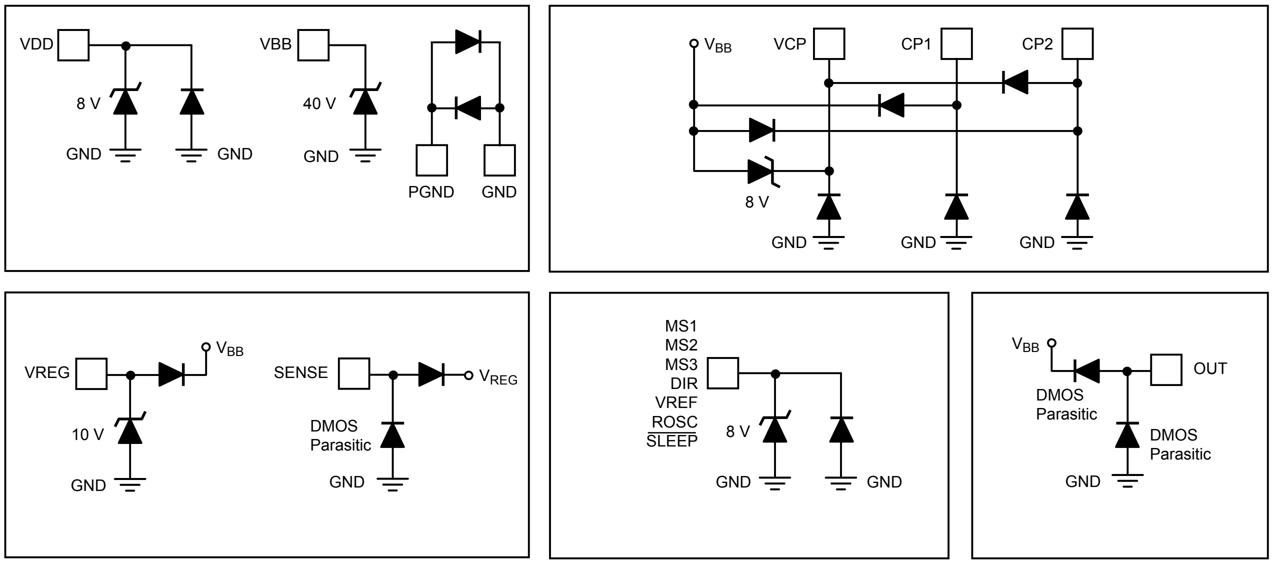


Figure 8: Typical Application and Circuit Layout

Pin Circuit Diagrams



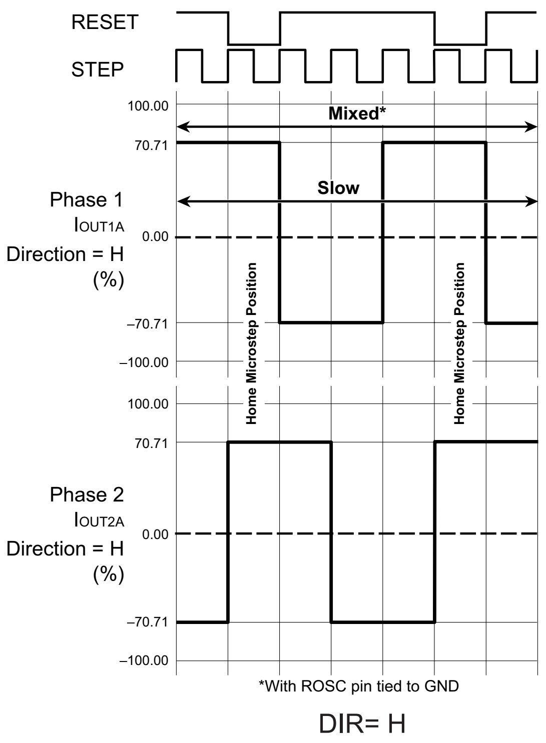


Figure 9: Decay Mode for Full-Step Increments

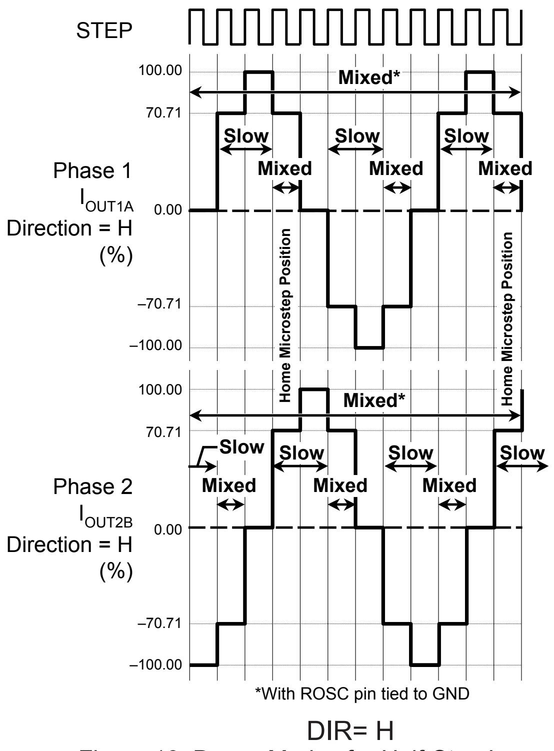


Figure 10: Decay Modes for Half-Step Increments

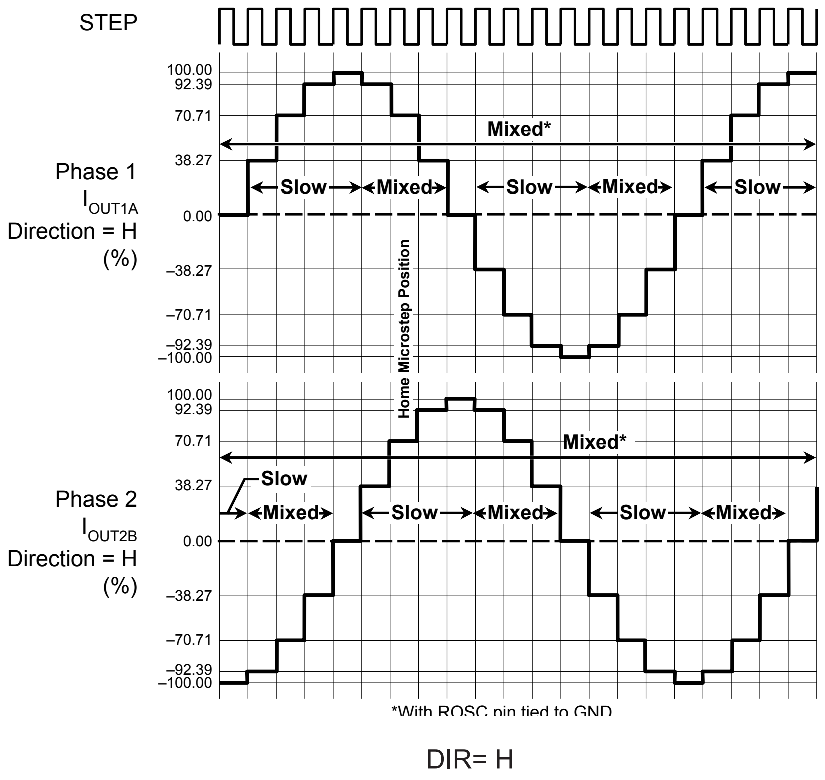


Figure 11: Decay Modes for Quarter-Step Increments

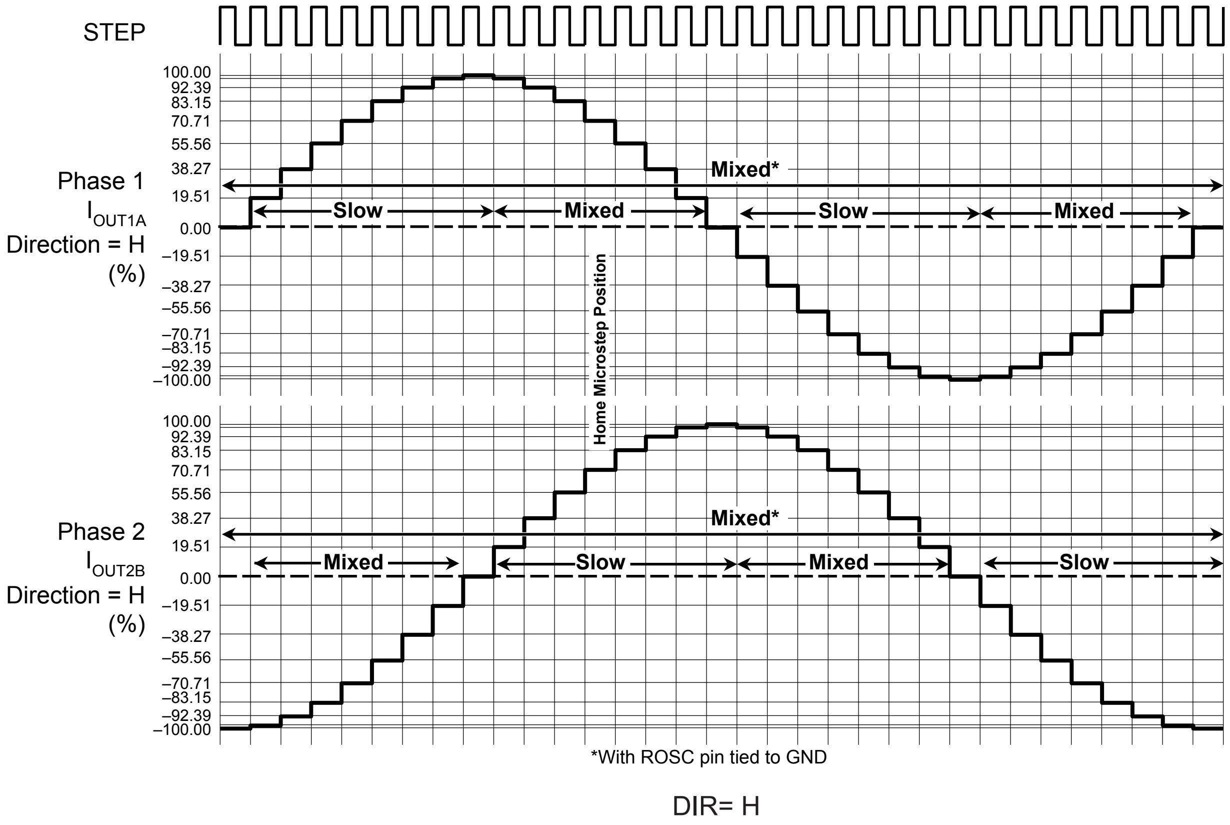
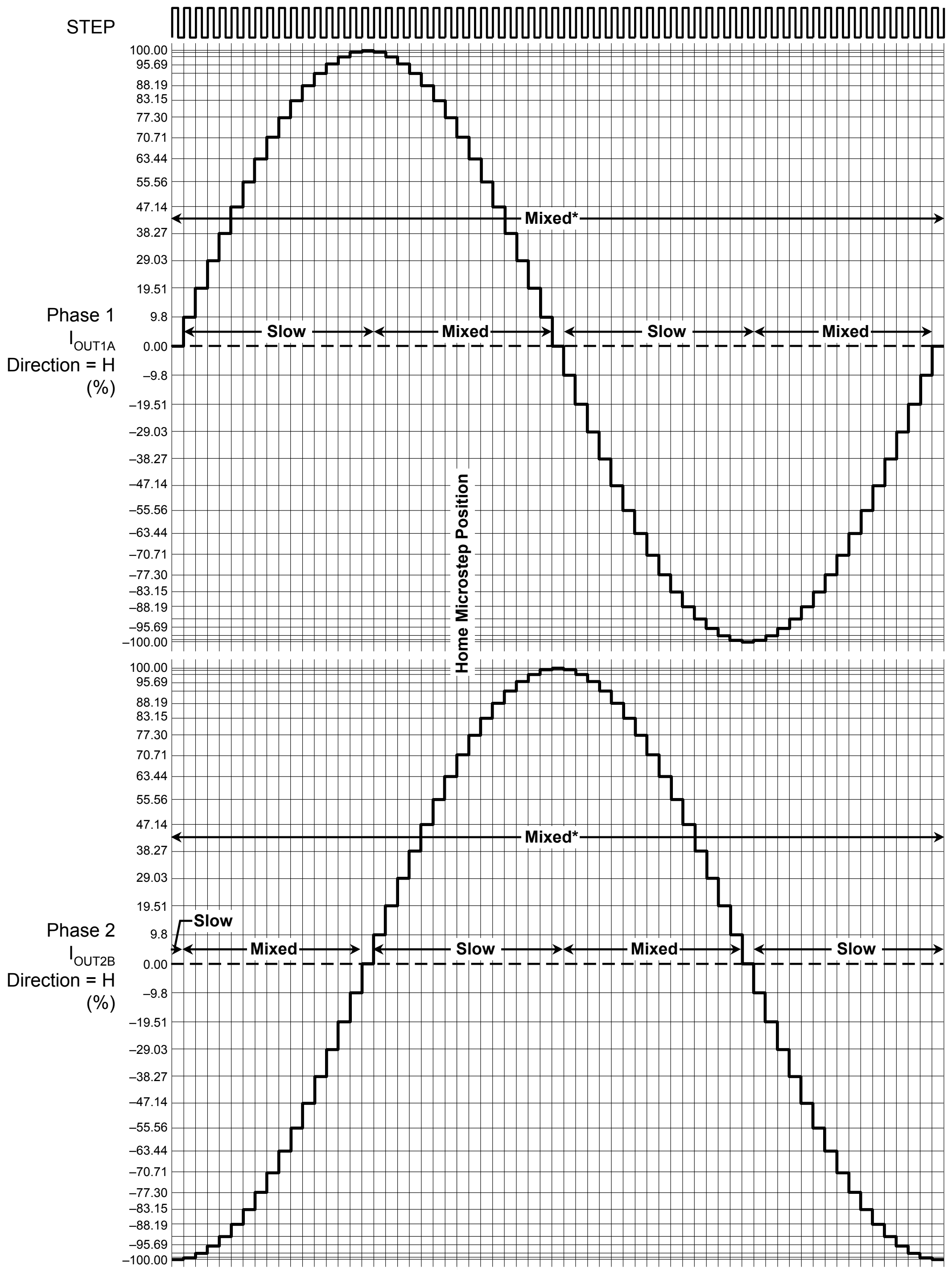


Figure 12: Decay Modes for Eighth-Step Increments



*With ROSC pin tied to GND

DIR= H

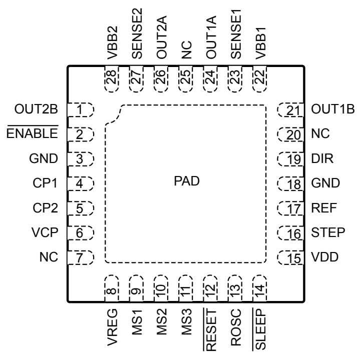
Figure 13: Decay Modes for Sixteenth-Step Increments

Table 2: Step Sequencing Settings

Home microstep position at Step Angle 45º; DIR = H

Full Step #	Half Step #	1/4 Step #	1/8 Step #	1/16 Step #	Phase 1 Current [% $I_{tripMax}$] (%)	Phase 2 Current [% $I_{tripMax}$] (%)	Step Angle (º)
	1	1	1	1	100.00	0.00	0.0
				2	99.52	9.80	5.6
			2	3	98.08	19.51	11.3
				4	95.69	29.03	16.9
		2	3	5	92.39	38.27	22.5
				6	88.19	47.14	28.1
			4	7	83.15	55.56	33.8
				8	77.30	63.44	39.4
1	2	3	5	9	70.71	70.71	45.0
				10	63.44	77.30	50.6
			6	11	55.56	83.15	56.3
				12	47.14	88.19	61.9
		4	7	13	38.27	92.39	67.5
				14	29.03	95.69	73.1
			8	15	19.51	98.08	78.8
				16	9.80	99.52	84.4
	3	5	9	17	0.00	100.00	90.0
				18	–9.80	99.52	95.6
			10	19	–19.51	98.08	101.3
				20	–29.03	95.69	106.9
		6	11	21	–38.27	92.39	112.5
				22	–47.14	88.19	118.1
			12	23	–55.56	83.15	123.8
				24	–63.44	77.30	129.4
2	4	7	13	25	–70.71	70.71	135.0
				26	–77.30	63.44	140.6
			14	27	–83.15	55.56	146.3
				28	–88.19	47.14	151.9
		8	15	29	–92.39	38.27	157.5
				30	–95.69	29.03	163.1
			16	31	–98.08	19.51	168.8
				32	–99.52	9.80	174.4
	5	9	17	33	–100.00	0.00	180.0
				34	–99.52	–9.80	185.6
			18	35	–98.08	–19.51	191.3
				36	–95.69	–29.03	196.9
		10	19	37	–92.39	–38.27	202.5
				38	–88.19	–47.14	208.1
			20	39	–83.15	–55.56	213.8
				40	–77.30	–63.44	219.4
3	6	11	21	41	–70.71	–70.71	225.0
				42	–63.44	–77.30	230.6
			22	43	–55.56	–83.15	236.3
				44	–47.14	–88.19	241.9
		12	23	45	–38.27	–92.39	247.5
				46	–29.03	–95.69	253.1
			24	47	–19.51	–98.08	258.8
				48	–9.80	–99.52	264.4
	7	13	25	49	0.00	–100.00	270.0
				50	9.80	–99.52	275.6
			26	51	19.51	–98.08	281.3
				52	29.03	–95.69	286.9
		14	27	53	38.27	–92.39	292.5
				54	47.14	–88.19	298.1
			28	55	55.56	–83.15	303.8
				56	63.44	–77.30	309.4
4	8	15	29	57	70.71	–70.71	315.0
				58	77.30	–63.44	320.6
			30	59	83.15	–55.56	326.3
				60	88.19	–47.14	331.9
		16	31	61	92.39	–38.27	337.5
				62	95.69	–29.03	343.1
			32	63	98.08	–19.51	348.8
				64	99.52	–9.80	354.4

Pin-out Diagram

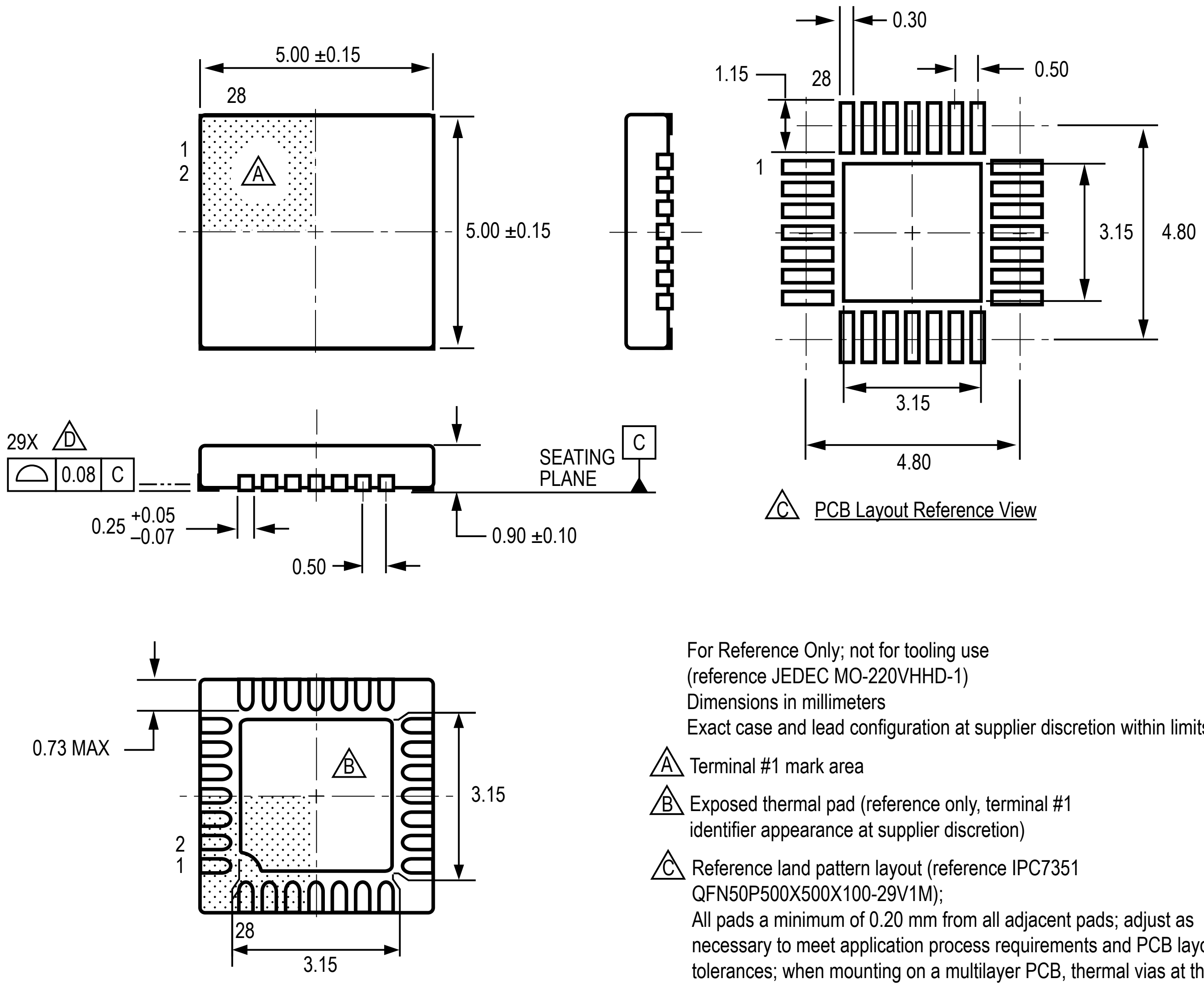


Terminal List Table

Name	Number	Description
CP1	4	Charge pump capacitor terminal
CP2	5	Charge pump capacitor terminal
VCP	6	Reservoir capacitor terminal
VREG	8	Regulator decoupling terminal
MS1	9	Logic input
MS2	10	Logic input
MS3	11	Logic input
$\overline{\text{RESET}}$	12	Logic input
ROSC	13	Timing set
$\overline{\text{SLEEP}}$	14	Logic input
VDD	15	Logic supply
STEP	16	Logic input
REF	17	G_m reference voltage input
GND	3, 18	Ground*
DIR	19	Logic input
OUT1B	21	DMOS Full Bridge 1 Output B
VBB1	22	Load supply
SENSE1	23	Sense resistor terminal for Bridge 1
OUT1A	24	DMOS Full Bridge 1 Output A
OUT2A	26	DMOS Full Bridge 2 Output A
SENSE2	27	Sense resistor terminal for Bridge 2
VBB2	28	Load supply
OUT2B	1	DMOS Full Bridge 2 Output B
$\overline{\text{ENABLE}}$	2	Logic input
NC	7, 20, 25	No connection
PAD	–	Exposed pad for enhanced thermal dissipation*

*The GND pins must be tied together externally by connecting to the PAD ground plane under the device.

ET Package, 28-Pin QFN with Exposed Thermal Pad



For Reference Only; not for tooling use
(reference JEDEC MO-220VHHD-1)
Dimensions in millimeters
Exact case and lead configuration at supplier discretion within limits shown

A Terminal #1 mark area

B Exposed thermal pad (reference only, terminal #1 identifier appearance at supplier discretion)

C Reference land pattern layout (reference IPC7351 QFN50P500X500X100-29V1M);
All pads a minimum of 0.20 mm from all adjacent pads; adjust as necessary to meet application process requirements and PCB layout tolerances; when mounting on a multilayer PCB, thermal vias at the exposed thermal pad land can improve thermal dissipation (reference EIA/JEDEC Standard JESD51-5)

D Coplanarity includes exposed thermal pad and terminals

Revision History

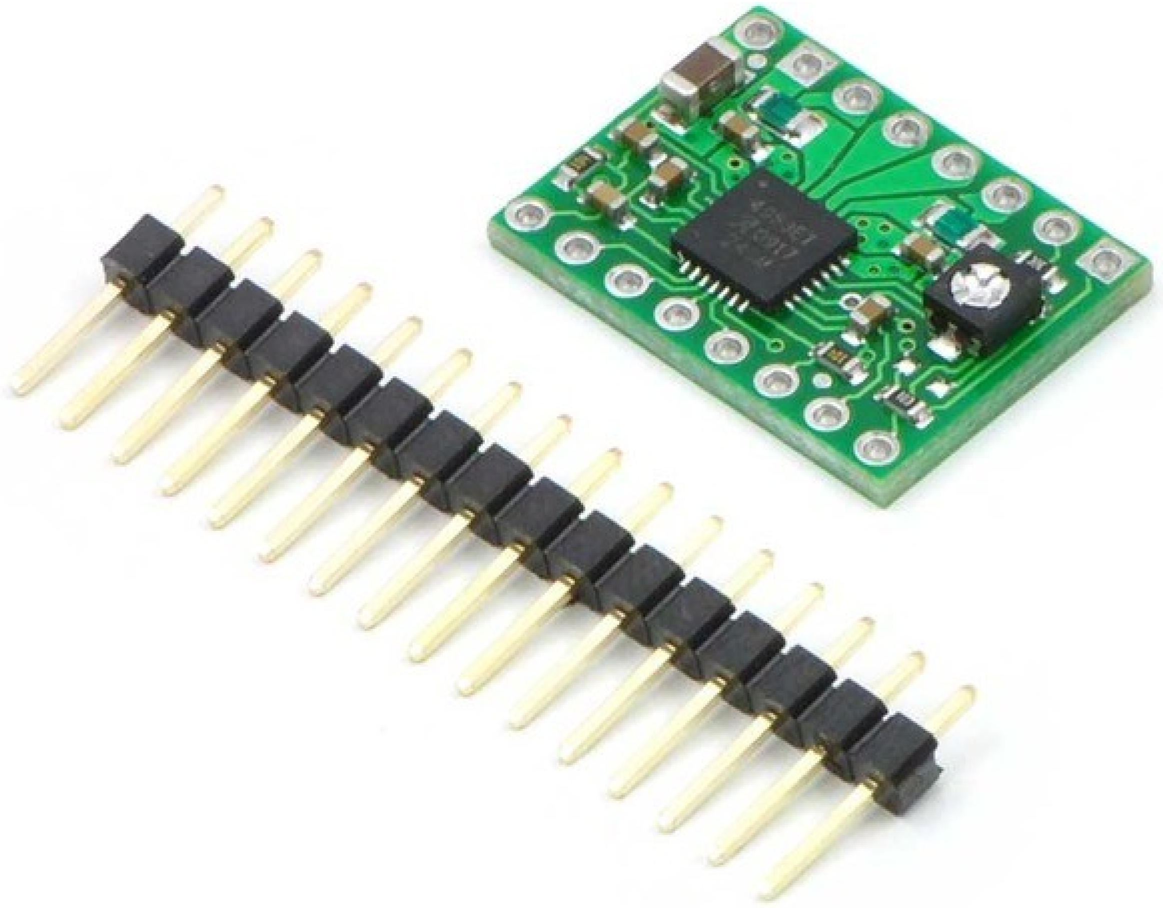
Revision	Revision Date	Description of Revision
4	January 27, 2012	Update I_{OCPST}
5	May 7, 2014	Revised text on pg. 9; revised Figure 8 and Table 2

For the latest version of this document, visit our website:

www.allegromicro.com

RB-Pol-176

Pololu 8-35V 2A Single Bipolar Stepper Motor Driver A4988



A4988

Stepper Motor Driver Carrier

The A4988 stepper motor driver carrier is a breakout board for Allegro’s easy-to-use A4988 microstepping bipolar stepper motor driver and is a drop-in replacement for the A4983 stepper motor driver carrier. The driver features adjustable current limiting, overcurrent protection, and five different microstep resolutions. It operates from 8 – 35 V and can deliver up to 2 A per coil.

Note: This board is a drop-in replacement for the original A4983 stepper motor driver carrier. The newer A4988 offers overcurrent protection and has an internal 100k pull-down on the MS1 microstep selection pin, but it is otherwise virtually identical to the A4983.

Description

Overview

This product is a carrier board or breakout board for Allegro's A4988 DMOS Microstepping Driver with Translator and Overcurrent Protection; we therefore recommend careful reading of the A4988 datasheet (380k pdf) before using this product. This stepper motor driver lets you control one bipolar stepper motor at up to 2 A output current per coil (see the Power Dissipation Considerations section below for more information). Here are some of the driver's key features:

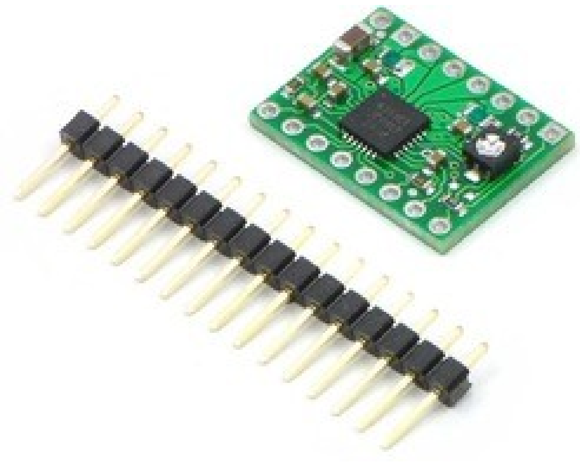
- •Simple step and direction control interface
- •Five different step resolutions: full-step, half-step, quarter-step, eighth-step, and sixteenth-step
- •Adjustable current control lets you set the maximum current output with a potentiometer, which lets you use voltages above your stepper motor's rated voltage to achieve higher step rates
- •Intelligent chopping control that automatically selects the correct current decay mode (fast decay or slow decay)
- •Over-temperature thermal shutdown, under-voltage lockout, and crossover-current protection
- •Short-to-ground and shorted-load protection (this feature is not available on the A4983)

Like nearly all our other carrier boards, this product ships with all surface-mount components —including the A4988 driver IC—installed as shown in the product picture.

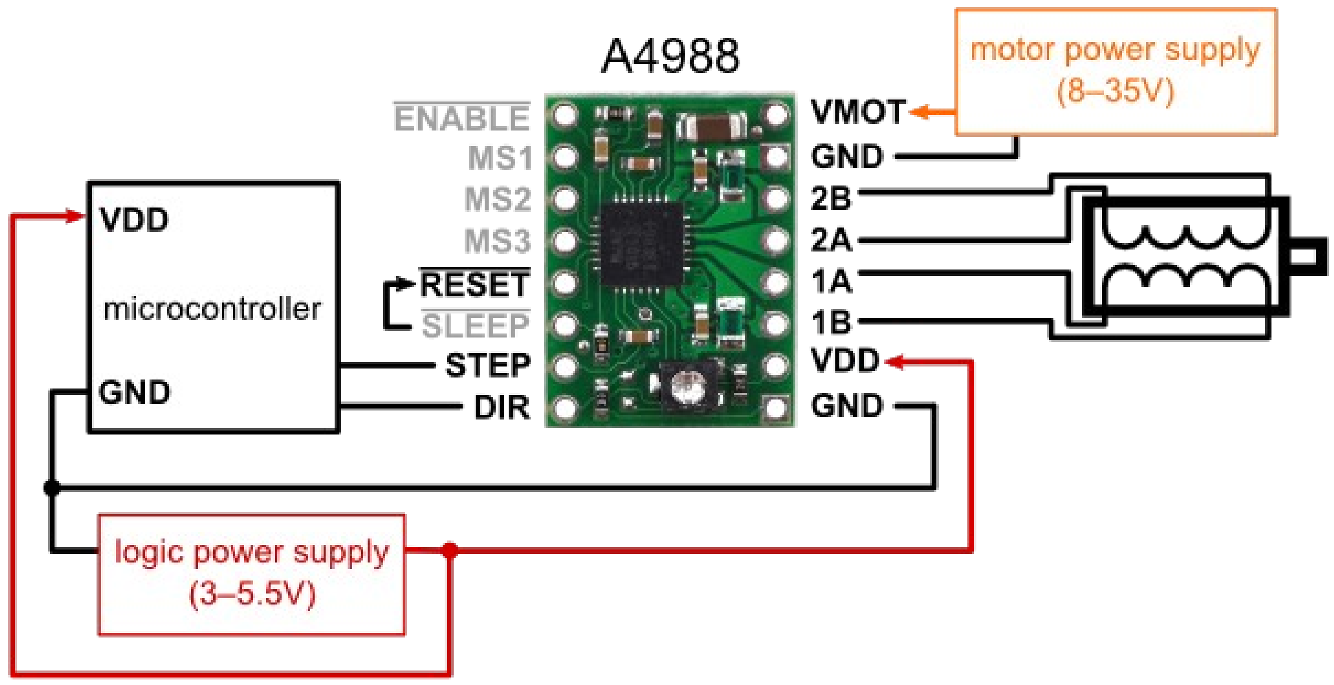
We also sell a larger version of the A4988 carrier that has reverse power protection on the main power input and built-in 5 V and 3.3 V voltage regulators that eliminate the need for separate logic and motor supplies.

Included hardware

The A4988 stepper motor driver carrier comes with one 1×16-pin breakaway 0.1" male header. The headers can be soldered in for use with solderless breadboards or 0.1" female connectors. You can also solder your motor leads and other connections directly to the board.



Using the driver



Minimal wiring diagram for connecting a microcontroller to an A4988 stepper motor driver carrier (full-step mode).

Power connections

The driver requires a logic supply voltage (3 – 5.5 V) to be connected across the VDD and GND pins and a motor supply voltage of (8 – 35 V) to be connected across VMOT and GND. These supplies should have appropriate decoupling capacitors close to the board, and they should be capable of delivering the expected currents (peaks up to 4 A for the motor supply).

Motor connections

Four, six, and eight-wire stepper motors can be driven by the A4988 if they are properly connected; a FAQ answer explains the proper wirings in detail.
Warning: Connecting or disconnecting a stepper motor while the driver is powered can destroy the driver. (More generally, rewiring anything while it is powered is asking for trouble.)

Warning: Connecting or disconnecting a stepper motor while the driver is powered can destroy the driver. (More generally, rewiring anything while it is powered is asking for trouble.)

Step (and microstep) size

Stepper motors typically have a step size specification (e.g. 1.8° or 200 steps per revolution), which applies to full steps. A microstepping driver such as the A4988 allows higher resolutions by allowing intermediate step locations, which are achieved by energizing the coils with intermediate current levels. For instance, driving a motor in quarter-step mode will give the 200-step-per-revolution motor 800 microsteps per revolution by using four different current levels.

The resolution (step size) selector inputs (MS1, MS2, MS3) enable selection from the five step resolutions according to the table below. MS1 and MS3 have internal 100kΩ pull-down resistors and MS2 has an internal 50kΩ pull-down resistor, so leaving these three microstep selection pins disconnected results in full-step mode. For the microstep modes to function correctly, the current limit must be set low enough (see below) so that current limiting gets engaged. Otherwise, the intermediate current levels will not be correctly maintained, and the motor will effectively operate in a full-step mode.

MS1	MS2	MS3	Microstep Resolution
Low	Low	Low	Full step
High	Low	Low	Half step
Low	High	Low	Quarter step
High	High	Low	Eighth step
High	High	High	Sixteenth step

Control inputs

Each pulse to the STEP input corresponds to one microstep of the stepper motor in the direction selected by the DIR pin. Note that the STEP and DIR pins are not pulled to any particular voltage internally, so you should not leave either of these pins floating in your application. If you just want rotation in a single direction, you can tie DIR directly to VCC or

GND. The chip has three different inputs for controlling its many power states: RST, SLP, and EN. For details about these power states, see the datasheet. Please note that the RST pin is floating; if you are not using the pin, you can connect it to the adjacent SLP pin on the PCB.

Current limiting

To achieve high step rates, the motor supply is typically much higher than would be permissible without active current limiting. For instance, a typical stepper motor might have a maximum current rating of 1 A with a 5Ω coil resistance, which would indicate a maximum motor supply of 5 V. Using such a motor with 12 V would allow higher step rates, but the current must actively be limited to under 1 A to prevent damage to the motor.

The A4988 supports such active current limiting, and the trimmer potentiometer on the board can be used to set the current limit. One way to set the current limit is to put the driver into full-step mode and to measure the current running through a single motor coil without clocking the STEP input. The measured current will be 0.7 times the current limit (since both coils are always on and limited to 70% in full-step mode). Please note that the current limit is dependent on the Vdd voltage.

Another way to set the current limit is to measure the voltage on the “ref” pin and to calculate the resulting current limit (the current sense resistors are 0.05Ω). The ref pin voltage is accessible on a via that is circled on the bottom silkscreen of the circuit board. See the A4988 datasheet for more information.

Power dissipation considerations

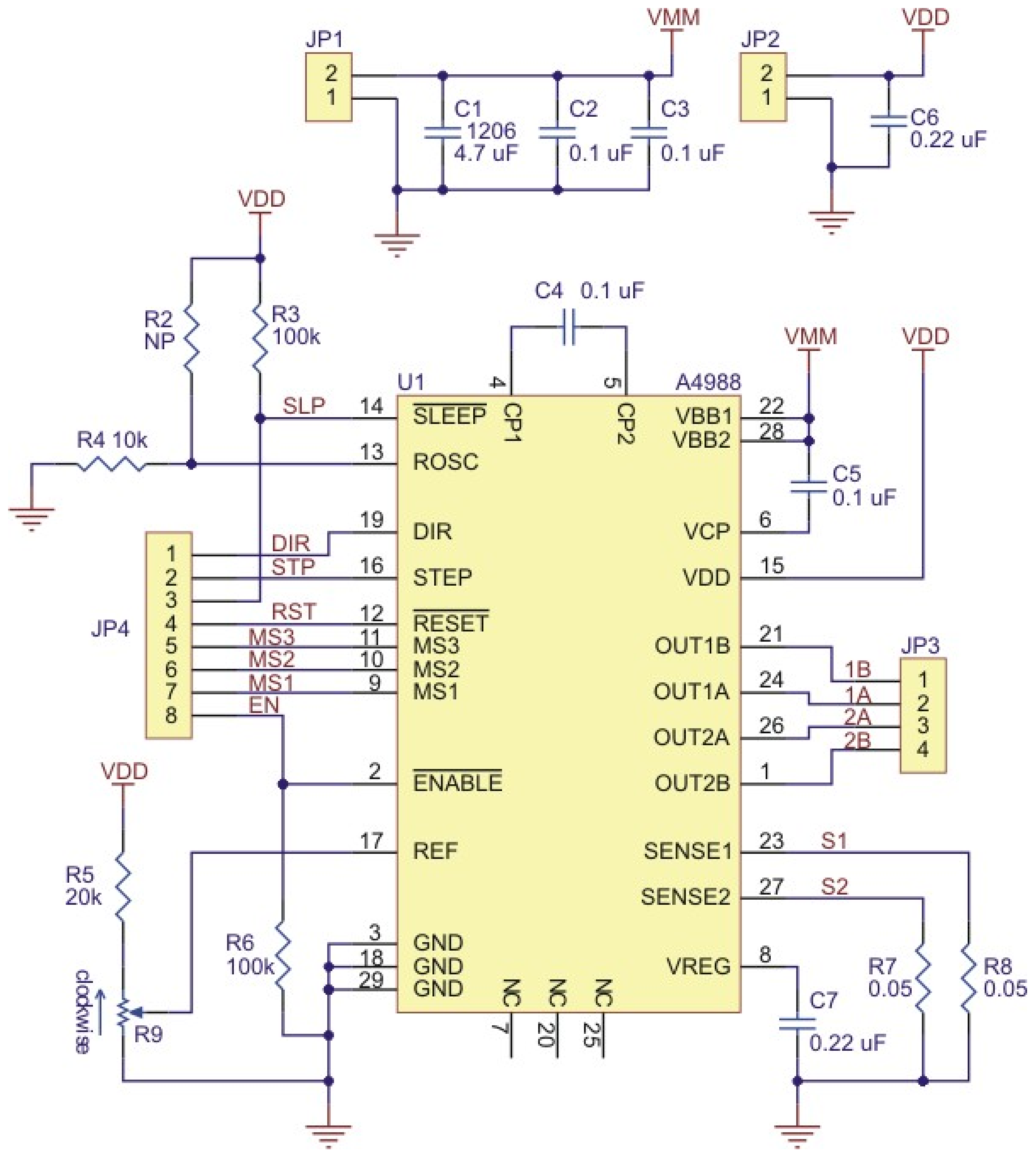
The A4988 driver IC has a maximum current rating of 2 A per coil, but the actual current you can deliver depends on how well you can keep the IC cool. The carrier’s printed circuit board is designed to draw heat out of the IC, but to supply more than approximately 1 A per coil, a heat sink or other cooling method is required.

This product can get hot enough to burn you long before the chip overheats. Take care when handling this product and other components connected to it.

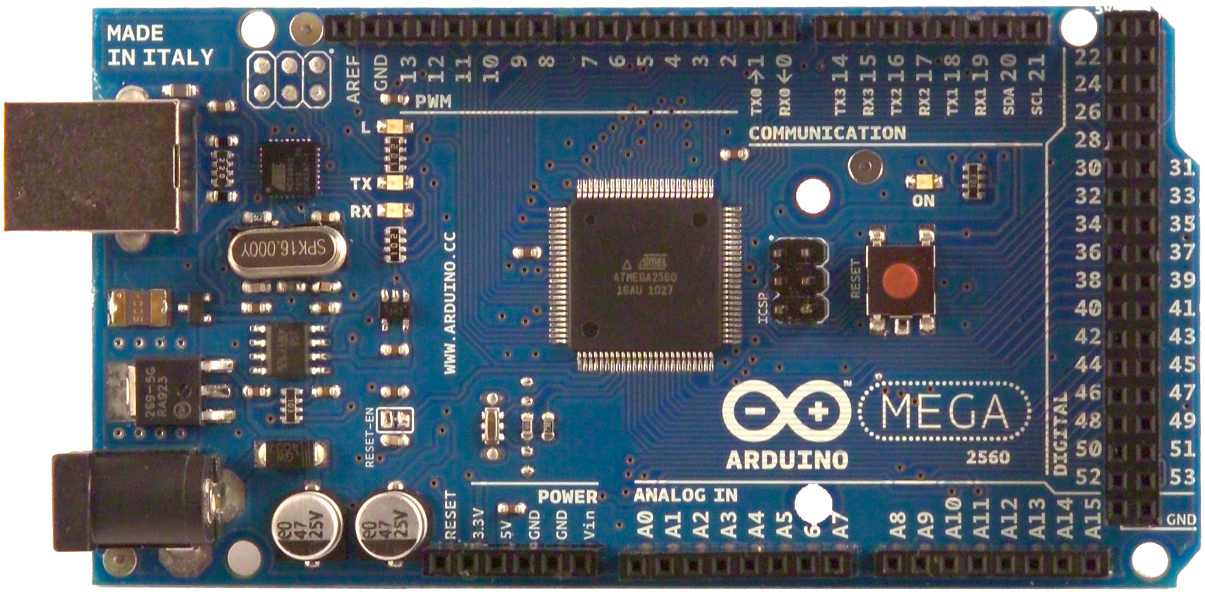
Please note that measuring the current draw at the power supply does not necessarily provide an accurate measure of the coil current. Since the input voltage to the driver can be significantly higher than the coil voltage, the measured current on the power supply can be quite a bit lower than the coil current (the driver and coil basically act like a switching step-down power supply). Also, if the supply voltage is very high compared to what the motor needs to achieve the set current, the duty cycle will be very low, which also leads to significant differences between average and RMS currents.

Schematic diagram

Schematic diagram of the md09b A4988 stepper motor driver carrier.



Arduino MEGA 2560



Product Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Index

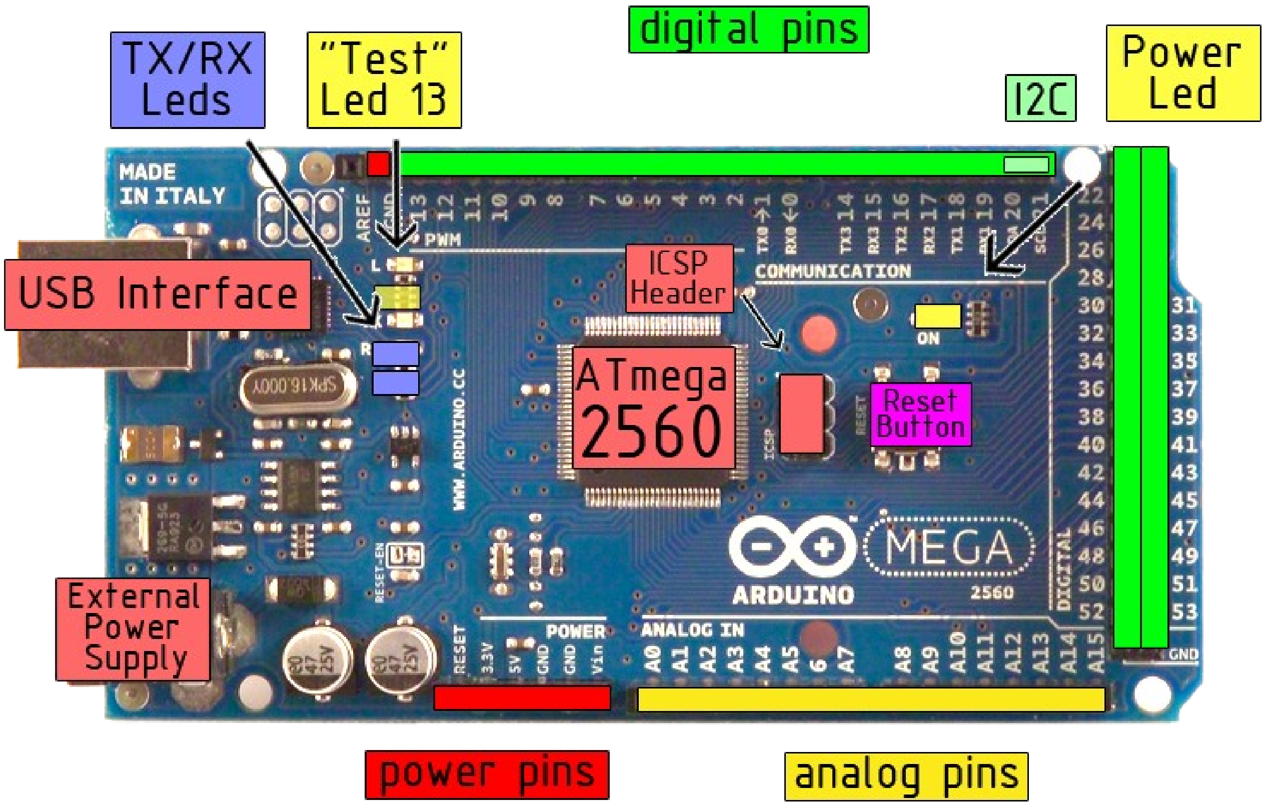
Technical Specification

EAGLE files: arduino-mega2560-reference-design.zip Schematic: arduino-mega2560-schematic.pdf

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

the board



Power

The Arduino Mega2560 can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **I^{2}C: 20 (SDA) and 21 (SCL).** Support I^{2}C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I^{2}C pins on the Duemilanove.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega's digital pins.

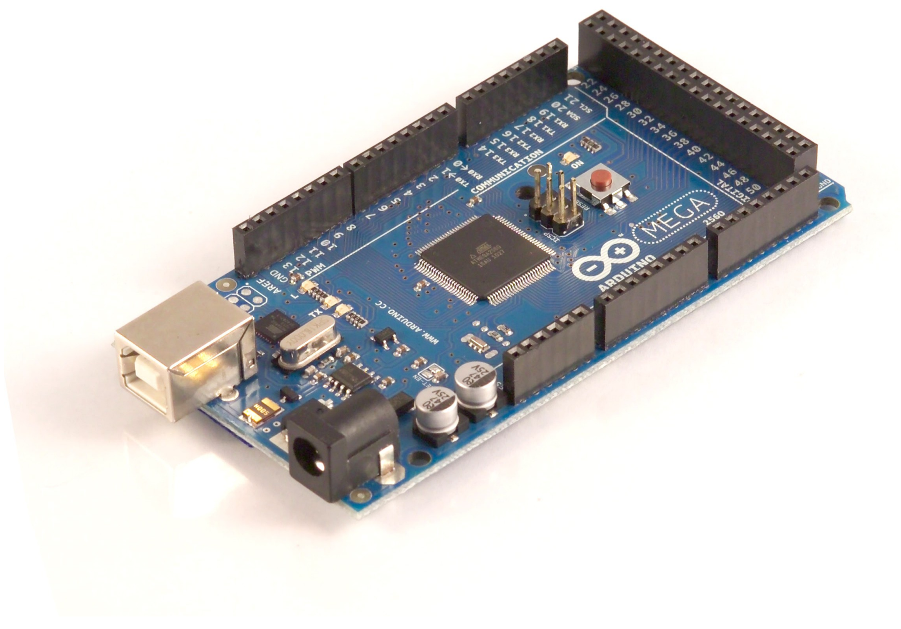
The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. To use the SPI communication, please see the ATmega2560 datasheet.

Programming

The Arduino Mega2560 can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The Atmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.



Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Mega has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics and Shield Compatibility

The maximum length and width of the Mega PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega is designed to be compatible with most shields designed for the Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega and Duemilanove / Diecimila. **Please note that I^{2}C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).**

How to use Arduino

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

Arduino is a cross-platoform program. You'll have to follow different instructions for your personal OS. Check on the Arduino site for the latest instructions. *http://arduino.cc/en/Guide/HomePage*

Linux Install   Windows Install   Mac Install

Once you have downloaded/unzipped the arduino IDE, you can Plug the Arduino to your PC via USB cable.

Blink led

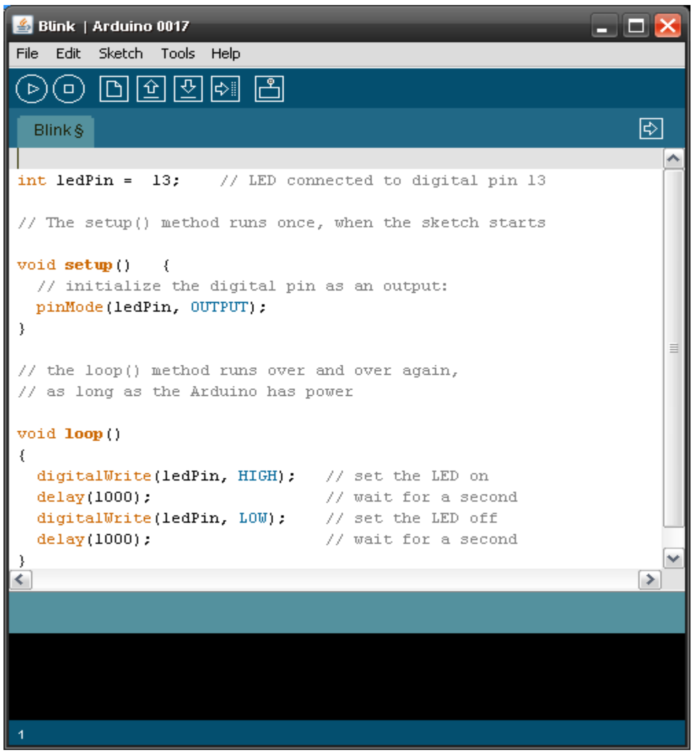
Now you're actually ready to "burn" your first program on the arduino board. To select "blink led", the physical translation of the well known programming "hello world", select

File>Sketchbook> Arduino-0017>Examples> Digital>Blink

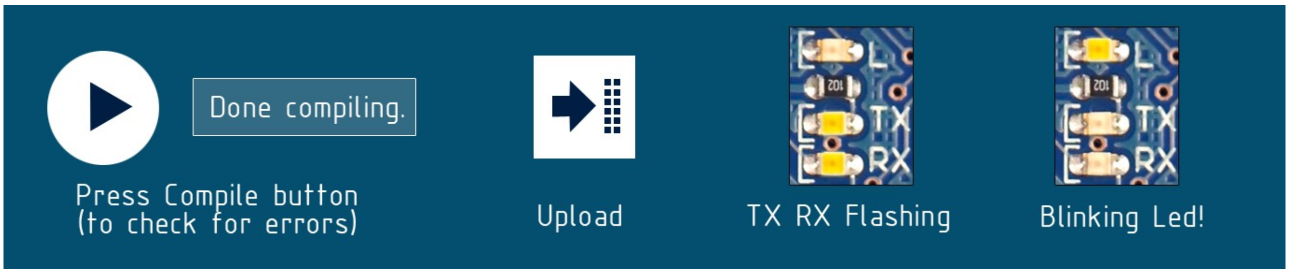
Once you have your skecth you'll see something very close to the screenshot on the right.

In **Tools>Board** select MEGA

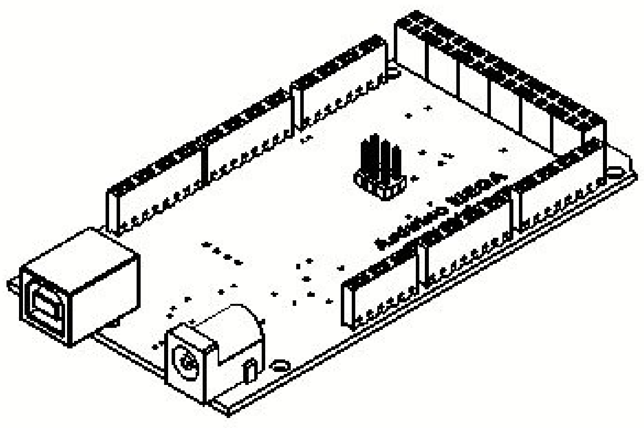
Now you have to go to **Tools>SerialPort** and select the right serial port, the one arduino is attached to.

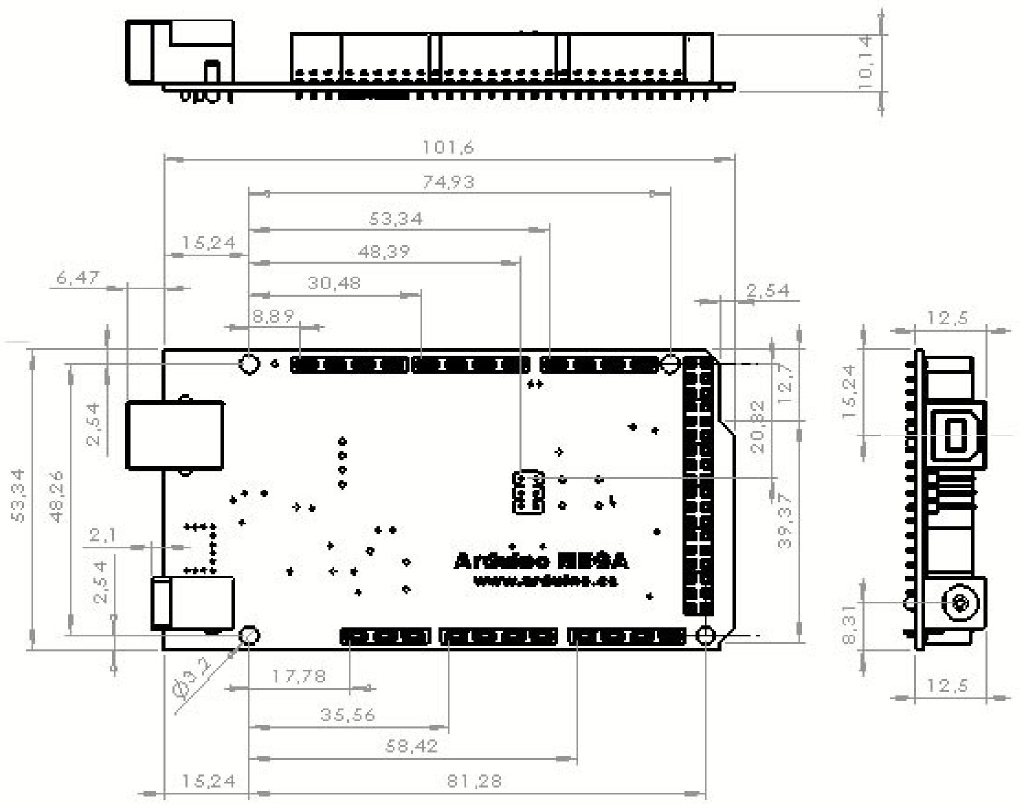


```
int ledPin =  13;    // LED connected to digital pin 13

// The setup() method runs once, when the sketch starts

void setup()   {
  // initialize the digital pin as an output:
  pinMode(ledPin, OUTPUT);
}

// the loop() method runs over and over again,
// as long as the Arduino has power

void loop()
{
  digitalWrite(ledPin, HIGH);   // set the LED on
  delay(1000);                  // wait for a second
  digitalWrite(ledPin, LOW);    // set the LED off
  delay(1000);                  // wait for a second
}
```



Dimensioned Drawing





Terms & Conditions

1. Warranties

1.1 The producer warrants that its products will conform to the Specifications. This warranty lasts for one (1) years from the date of the sale. The producer shall not be liable for any defects that are caused by neglect, misuse or mistreatment by the Customer, including improper installation or testing, or for any products that have been altered or modified in any way by a Customer. Moreover, The producer shall not be liable for any defects that result from Customer's design, specifications or instructions for such products. Testing and other quality control techniques are used to the extent the producer deems necessary.

1.2 If any products fail to conform to the warranty set forth above, the producer's sole liability shall be to replace such products. The producer's liability shall be limited to products that are determined by the producer not to conform to such warranty. If the producer elects to replace such products, the producer shall have a reasonable time to replacements. Replaced products shall be warranted for a new full warranty period.

1.3 EXCEPT AS SET FORTH ABOVE, PRODUCTS ARE PROVIDED "AS IS" AND "WITH ALL FAULTS." THE PRODUCER DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE

1.4 Customer agrees that prior to using any systems that include the producer products, Customer will test such systems and the functionality of the products as used in such systems. The producer may provide technical, applications or design advice, quality characterization, reliability data or other services. Customer acknowledges and agrees that providing these services shall not expand or otherwise alter the producer's warranties, as set forth above, and no additional obligations or liabilities shall arise from the producer providing such services.

1.5 The Arduino™ products are not authorized for use in safety-critical applications where a failure of the product would reasonably be expected to cause severe personal injury or death. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Arduino™ products are neither designed nor intended for use in military or aerospace applications or environments and for automotive applications or environment. Customer acknowledges and agrees that any such use of Arduino™ products which is solely at the Customer's risk, and that Customer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

1.6 Customer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products and any use of Arduino™ products in Customer's applications, notwithstanding any applications-related information or support that may be provided by the producer.

2. Indemnification

The Customer acknowledges and agrees to defend, indemnify and hold harmless the producer from and against any and all third-party losses, damages, liabilities and expenses it incurs to the extent directly caused by: (i) an actual breach by a Customer of the representation and warranties made under this terms and conditions or (ii) the gross negligence or willful misconduct by the Customer.

3. Consequential Damages Waiver

In no event the producer shall be liable to the Customer or any third parties for any special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of the products provided hereunder, regardless of whether the producer has been advised of the possibility of such damages. This section will survive the termination of the warranty period.

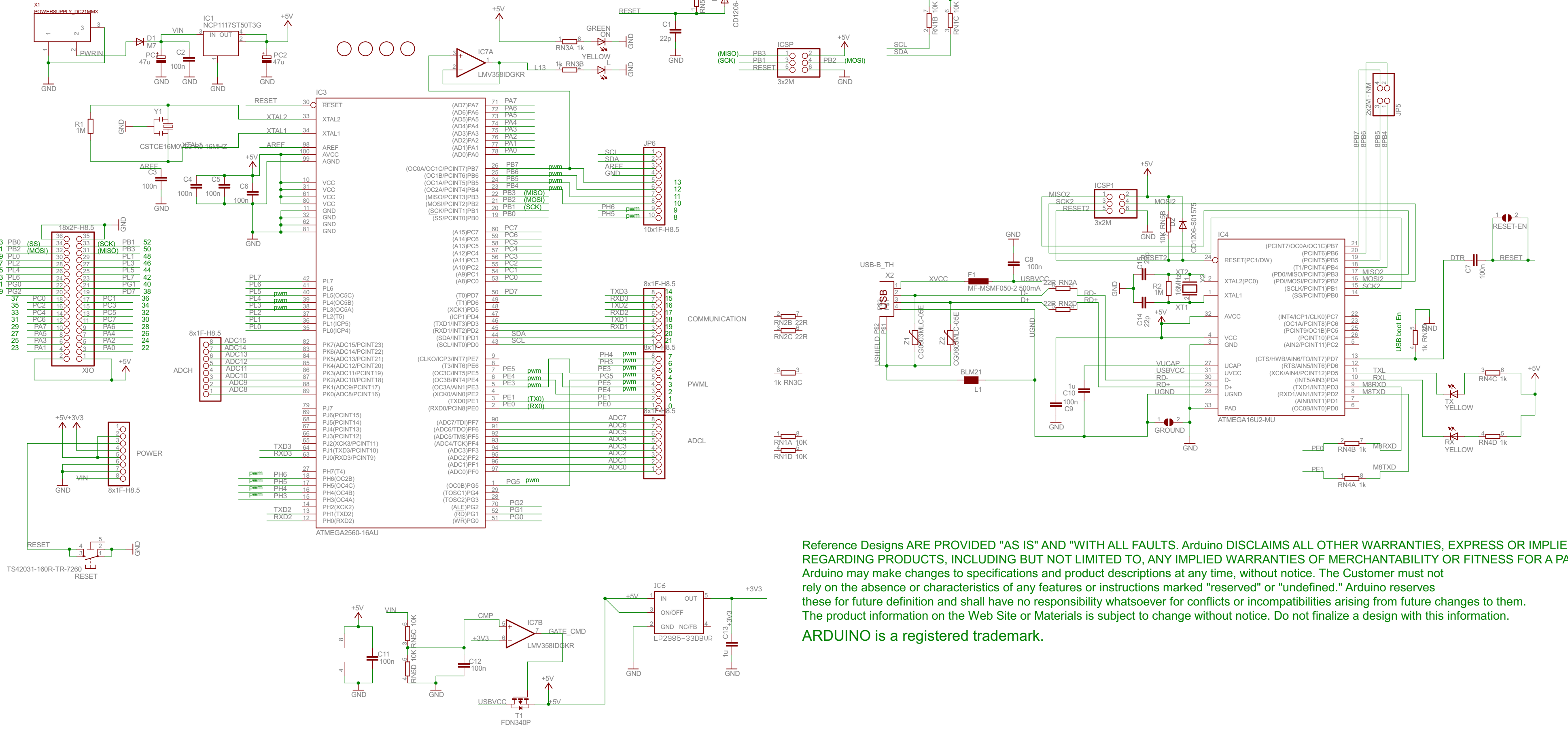
4. Changes to specifications

The producer may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." The producer reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

Enviromental Policies

The producer of Arduino™ has joined the Impatto Zero® policy of LifeGate.it. For each Arduino board produced is created / looked after half squared Km of Costa Rica's forest's.

Arduino™ MEGA 2560



Reference Designs ARE PROVIDED "AS IS" AND "WITH ALL FAULTS. Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE
Arduino may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Arduino reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.
The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

ARDUINO is a registered trademark.