SET UP AN INSPECTION MICROSCOPE AND IMAGE PROCESSING SOFTWARE

Greeshma M Chamarthy

Bachelor of Engineering in Electronics

MACQUARIE University SYDNEY·AUSTRALIA

Department of Electronics Engineering
Macquarie University

November 6th , 2017

Supervisor: Dr Stuart Jackson

ACKNOWLEDGMENTS

I would like to acknowledge my supervisor, Stuart Jackson for the guidance he provided for the project. I would like to express my gratitude to Stuart for guiding me in the process of being an engineer and teaching me a lot of new areas of study, which include the, The study of Microscopes, Optical Fibres, Mechanical Engineering, and the study of Cameras.

I would also like to thank METs, Lindsay from Microscopy lab for the advice I received during the development of this project.

A special Thanks to Jake for always helping me brainstorm and giving me advice.

STATEMENT OF CANDIDATE

I, Greeshma Chamarthy, declare that this report, submitted as part of the requirement for the award of Bachelor of Engineering in the Department of Electronic Engineering, Macquarie University, is entirely my own work unless otherwise referenced or acknowledged. This document has not been submitted for qualification or assessment an any academic institution.

Student's Name: Greeshma Chamarthy

Student's Signature:

Greeshma Chamarthy

Date: 6th November 2017

Abstract

This project aims to re-engineer the Inspection Microscope Opti Spec ME3000 to be an automated PC controlled Inspection Microscope with image processing software. This document describes design and implementation of the mechanical development on the Inspection Microscope along with the image processing software.

Contents

List of Figures

List of Tables:

1. Introduction

The Aim of this project was to re-engineer, ME3000, an optical fibre inspection microscope. The ME3000 is a standard manual inspection microscope with an analog video out, output on to the screen. The final developed ME3000 was expected to be a Fully computer operated Inspection Microscope with an image processing software, to see the digital image on the computer screen.

Optical fibres are widely used in telecommunication to transmit light from one end to the other. This is done by joining the ends of fibres to the desired length and use them as connectors. Optic splicing is a method that is used for joining the ends of the fibre optic cables and this method helps in reducing the loss of light and back reflection. Before the use of spicing method, it is crucial to prepare clean ends of the fibre. To achieve this the fibres are cleaved and polished.

In this project, the D-shaped optical fibre was used for the analysis under the microscope. The first step was to cleave the fibre, this method involved using cleaving tool – the process included holding the fibre under low tension, arranging it at proper location on the tool and carefully applied superior tension to snap or break the fibre with a clean cut.

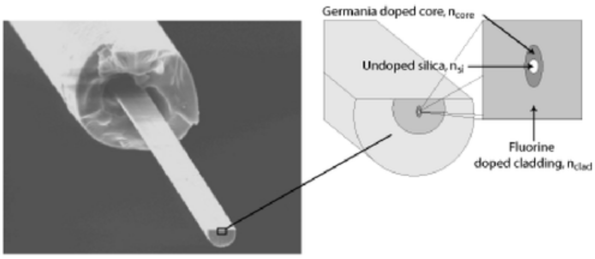


Figure 1: D-Shaped fibre used as a sample for inspection using the ME3000 Inspection microscope

The optical fibre was then inspected under the ME3000 and the displaying the image on the computer, by inspecting the fibre before joining their end surface will save a lot of time and energy as the inspection will help the user detect any bad cleave on the fibre. As a bad end surface on the fibre could lead to 2 ends not aligned properly and this could lead to gap between the fibres that will cause loss in a mechanical splice. Hence it is very important to inspect the end surface of fibre before using them. With the help of image processing software and the computer operated inspection microscope, it is possible to reject the bad cleave, unpolished surface and use only the good fibres. The optical fibre requires the precise preparation of the fibre ends; hence this project will aim towards using highly précised equipment to achieve the quality.

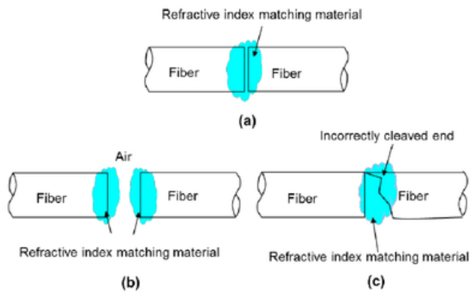


Figure 2: The comparison of a bad fibre to a good fibre. Bad fibres will not connect properly thus losing the light whereas a good fibre end surface helps reduce loss of light.

Thesis Overview

This section details the progress of the project, and details out the topics such as Project Objectives, Project Baseline Review, Time Budget Review, Financial Budget Review.

1.1.1 Project objectives

The project objectives at the beginning of the project were

- Conversion of analog inspection microscope to digital inspection microscope that will be connected to a laptop via USB cable.
- To be able to control the movement of the inspection microscope via Laptop, using stepper motor mount (XYZ focus).
- To develop image processing software that will be connected to the inspection microscope to deliver the 2D image of the optical fibre.

The project objectives have slightly changed with the progression of the project and the important objective that were identified are:

- The machinal mount concept of the inspection microscope, as it is necessary to remove the unnecessary platform of the ME3000 and replace the section that holds the optical fibre sample with a new platform, this new platform needs to hold the 3 axis XYZ positioning linear stage at a precise and accurate location for the optical fibre to be clearly visible in the lens of ME3000. The first and main objective of this project would be the ability to mechanically assemble the section on the inspection microscope for it to be used practically.
- The second most important objective would be the assembly between the micrometre of the XYZ linear stage and the actuator. The idea here is to use the motor in the actuator that will be controlled by the Arduino board using the commands given from PC to influence the movement of the micrometre. Thus, the objective here is to assemble the actuator and micrometre in such a way that the above described system will be created.

- The final objective would be to convert the analog image to a digital image using SDI to HDMI video converter.

The final objectives identified and solved are:
The project objectives:

- ✓ To convert the American standard ME3000 inspection Microscope to Australian Standard Microscope, for this a Step-down transformer of 240V (Australian Standard) to 115V (American Standard) was used
- ✓ The Linear Stage A-LH by Line tool.Co was modified mechanically to customise the setting to meet the needs of the microscope.
- ✓ To drive the actuator using the Arduino board to transform a manual inspection microscope to a computer operated microscope.
- ✓ The analog CCD camera has been replaced with a CMOS digital camera with a C-mount fitting.
- ✓ Image processing software – understanding the use of the digital image processor and experimenting with different images to find all the details and things that can be done with the software during the analysis process.

1.1.2 Project Baseline Review

The time frame for the completion of this project is from Week 1 of the 2nd semester of 2017 to Week 13. In Week 13 the final thesis will be submitted, there will be the final presentation with a working computer operated inspection microscope with image processing software presented along with a poster in relation to the final inspection microscope.

The initial plan was to complete the mechanical section of the inspection microscope and then to do the electrical connection of the ME3000 to PC, as compared to the mechanical assembly of the ME3000 the electrical connectivity is quite simple.

The mechanical development of the linear stage started in Week 4 and after taking out unsuitable ideas and the time for it to be mechanical made has taken 5 week's works. The final product was delivered in Week 9. To drive the actuator using an Arduino needed more research and trial and error processes to final control the actuator in the desired way required for the project and the process has taken more than 5 weeks to complete and the image processing software section, after an unsuccessful attempt to change the analog signal from the CCD camera of the microscope to a digital signal using an DIGI analog to digital conversion adaptor, it was decided to use a Digital CMOS camera with an image processing software.

1.1.3 Time Budget Review

This project is progressing at a good pace and the initial Gantt Chart was:

Figure 3: Gantt Chart of the project

The Task include Understanding the components of the project, Mechanical Set-up – modification of the linear stage, Arduino to drive the Actuator and image processing software on the span of 12 Weeks.

1.1.4 Financial Budget Review

For the financial Budget, this project was allocated with $300. Most of the equipment needed for this project were provided beforehand and all the necessary tools and software along with PC were provided.

Name of the Product	Amount Used in $ (Dollars)
Step-Down Transformer	109
The Australian to USA plug point	20
Iduino Motor Controller	26
Pack of Jump wires	10
Digital Camera	568

Table 1: The division of the money spend from the budget on the project.

2. System model and Development

a. Inspection Microscope ME3000

- **Background of ME3000**

ME3000 Fibre Optic Inspection Microscope is used for visual inspection of the end surface of connector, it is mainly used for finding dirt or scratches and possible defects on fibre optic connectors.

This Microscope has a fixture to hold the optical fibre in the field of view, with the help of a variety of lens and a light source to illuminate the fibre accurately. The ME3000 has a video magnification of 150X – 1400X, hence the use of ME3000 will inspect even the cleaved fibres that require to be viewed only from the side to inspect the breakover and lip.

Fibre Optic Microscopes can be of portable or special design microscopes, ME3000 is designed especially for fibre optics with sophisticated optics and illumination option and was also included with custom made accessories to hold the fibre. The microscope is customised for easy viewing and analyses of fibre and a video capture card is used to save the image of the fibre for further analyses and for records.

When using fibres as connectors, it is ideal to have a well-made connector that full-fills all the requirements such as having smooth, polished and a scratch free finish, and to be able to achieve this well-made fibre we use visual inspection to be able to determine the quality of the connector and also help in termination and diagnosis of a damaged optical fibre. The use of visual inspection at early stages to find faults will help the user in the longer run as a smooth, polished and scratch free fibre will last longer and are unlikely to cause difficulties in the future.

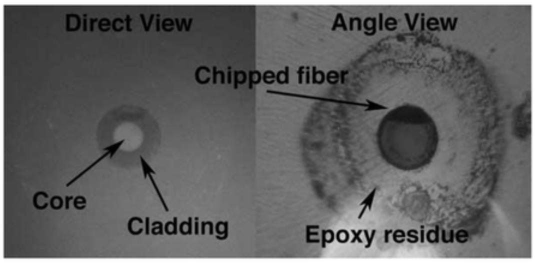


Figure 4: An image of the end-surface of the fibre and the area that needs to inspecting to determine the bad or good cleave of the fibre.

Magnifying aspect of the Inspection microscope is of great importance as it will help with proper viewing of the fibre, the general standard for a good magnification is said to be in the range of 30 – 400 powers and ME3000 is of 150X – 1400X, this magnification will produce the visible image given by the formula,

$$\text{Magnification} = \text{Image} \div \text{Object}$$

The ME3000 Microscope uses visible light from the light source and the objective lenses to magnify the fibre that is not visible to the naked eye. The magnification of the light microscope combines the powers of the eyepiece and the objective lens. The magnification 150X – 1400X, is of the eye-piece and the objective lens of the ME3000 microscope, the “X” in the magnification value indicates that the number is X 10. Hence the total magnification of ME3000 is 1500 – 14000 and this will help the user obtain the image of the fibre in great detail, even though the magnification range of ME3000 is higher than the recommended range, it just means that ME3000 is of higher resolution. While working with higher resolution it is important to be aware of the fact that the fibre is at a very close distance that the ME3000 is capable of identifying ignorable faults as well hence it is the user’s responsibility to analyse the content to check if the fault is ignorable or if it is of concern.

Using the ME3000, the fibre can be inspected in 3 ways for better analysis. The 3 ways include:

1. Viewing directly at the surface of a polished fibre with a common axis to lighting from the light source.
2. Viewing directly with the light source transmitting light though the core of the fibre
3. Viewing at an angle with the light coming in the opposite angle of the fibre

It is a very good idea to be able to inspect the fibre in more than 1 ways for stronger reliability and accuracy of the results and this will be talked about in more detail in the future development section below.

Things to be noted while inspecting the fibre using ME3000 would be to understand the fact that Me3000 is of high magnification, hence not to be overly critical on the scratches of the fibre. The only fibre defect that effects the working of the fibre are over the fibre core and things like chipping of the glass around the outside of cladding is not of interest as this will have no direct effect on fibres ability to work.

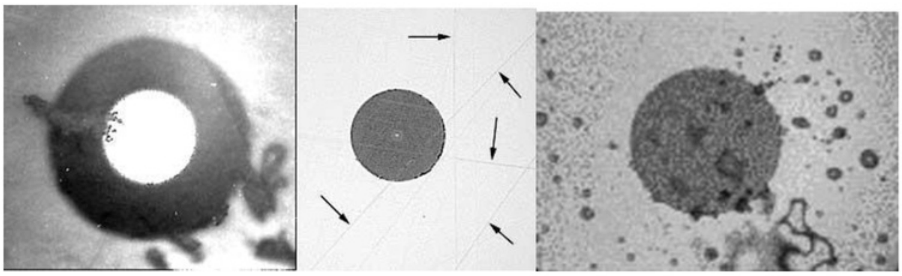


Figure 5: Cracked fibre, Scratched fibre and contaminated fibre respectively images for reference when inspecting with ME3000, it is important to make sure the image on the image processing software is not similar to any of this.

- **Set-up of ME3000 – Illuminator, Objective lens, Camera,**

The ME3000 has three main components namely the light illuminator, microscopic lens, CCD Camera. It also consists of sub-components namely the Camera reflection lens, vertically reflected illuminator. The external components included for the working of ME3000 would be the Power supply and the screen.

The illuminator or light source used for this project is OptiSpec MP35030, with 150 Watts power and variable intensity of light. The purpose of the light source was to provide the microscope with intense and cool illumination continuously. It is designed with optic light guides that supply the light in a guided path continuously and this guided light can be customised by manipulating or directing the rays. For this project, the requirement of the light source was to supply continuous light to ME3000.

The light source is connected to Olympus U-RLA reflected light illuminator through a photo port tube, and the U-RLA is fitted with the objective lens, (S.FlatField 4/0.10, 160/0.17 DIN – LWD M 10/0.22, 240/0 – LWD M 20/0.30, 240/0 – LWD M 40/0.45, 240/0 – Newport M-5X 0.10) so that the light from the light source is guided through to the objective lens and then focuses through the selected lens on to the optical fibre hence reflecting the end surface of the fibre to the CCD camera connected to the U-RLA illuminator and from the video out port of the CCD camera it sends out an analog video to the Screen connected. This set-up gives out an analog video of the end surface of the fibre for analysis.

- **Modification made**

As described in the set-up section of ME3000, the original ME3000 was only able to give out an analog output and was controlled manually. The aim of this project was to modify ME3000 to be digital and to be operated using the computer.

To achieve this the following modification were made on ME3000, the first was to convert the input power of the light source OptiSpec MP35030 as it is an American product hence a transformer was used to step-down from 240 Volts (Australian Standard) to 115 Volts (American Standard).

The next section that was modified was the CCD camera, the original camera was Hitachi Denshi Model KP-M1U CCD camera with analog video out. The orginal was replaced by the digital Proscitech CCD camera. The camera was then connected to the PC to see the image on the image processing software that was included with the digital camera.

The Hitachi Camera –

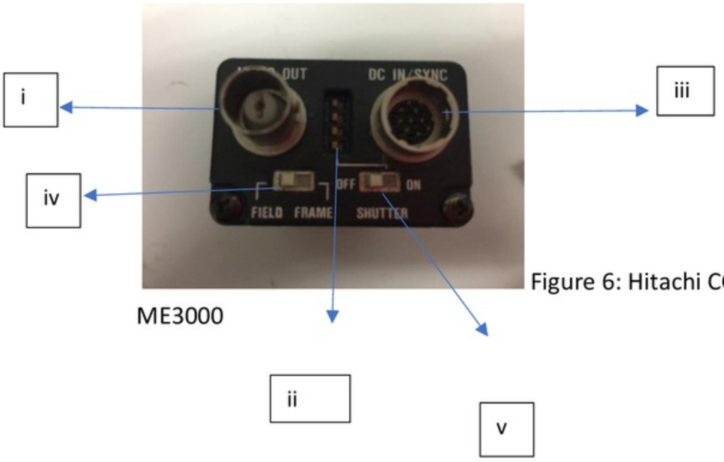


Figure 6: Hitachi CCD camera of ME3000

i. Video Out (BNC) connector – A composite video signal (VS) is fed using this connector through to a 75-Ohm coaxial cable on to the screen.
ii. This switch is used to control the shutter speed
iii. External 12V DC input through this connector
iv. This switch is used to change between the integration modes.
v. To switch ON/OFF the shutter.

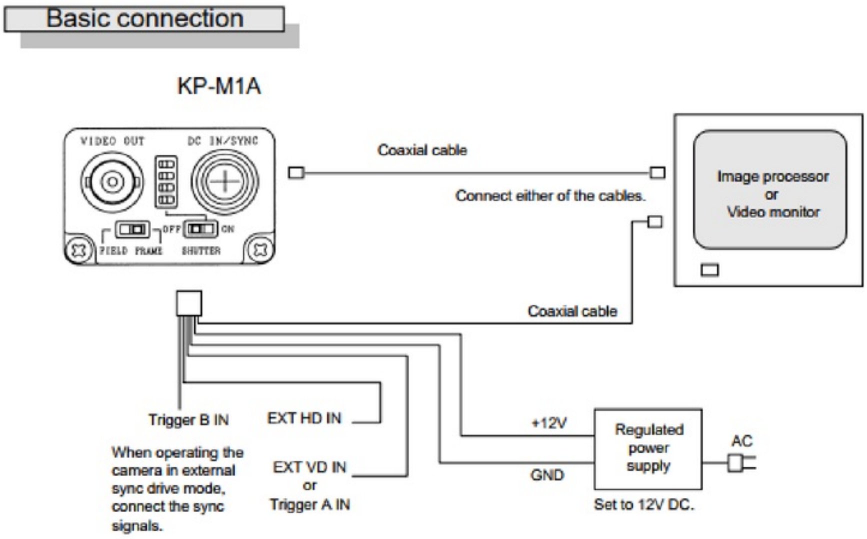


Figure 7: The Basic connection of the Hitachi camera to the Screen

The output signal of the camera is given by a square wave.

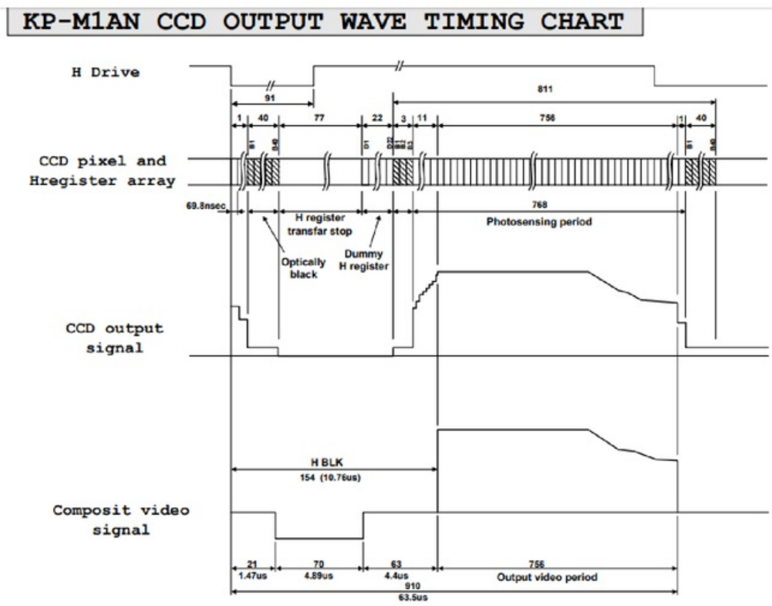


Figure 8: Output of Hitachi CCD Camera.

The CCD Hitachi camera was change to a CMOS Digital Olympus camera. The digital camera was included with an image processing software that enable the analysis and storing of the image easy.

The last and the most complication modification was the stage the fibre sample is attached to for the purpose of analysing the fibre's end surface. For this the block of metal that held the samples was removed and was replaced by a linear stage A-LH, the linear stage had 3 axes namely XYZ along with micrometres built in to them, each for each axis.

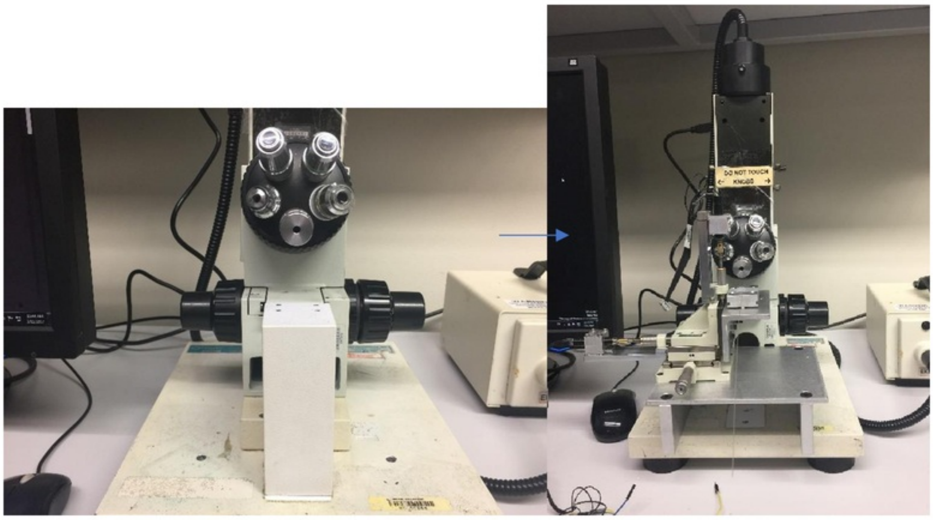


Figure 9: The transformation of Manual ME3000 microscope to the New Computer operated Inspection microscope for optical fibres

- **Working principle includes schematics**

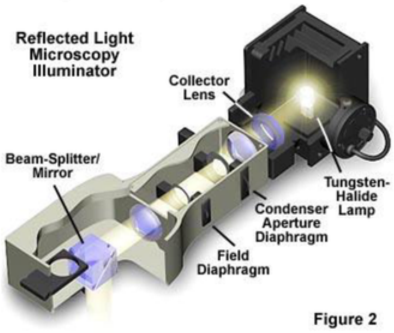
ME3000, Microscope was a system developed by combining different components from different companies and were arranged and customised to meet the need of the user. ME3000 was developed more than 2 years ago and due to the fact that the company that developed ME3000 being no longer in operation it was very hard to find the information or manuals on the product.
The best way to understand ME3000 was by dividing up and part or section and them contacting the company on more information on their products and then putting everything

found and researched in to one document to analyse the complete working and the structure of ME3000.

Further details on the individual part testing will be described in the section below – Testing Procedures.

Reflected Illuminator – Olympus U-RLA – The vertical reflected illuminator is designed to pass the light from the light source through the objective lens of the microscope to the specimen and then through the same objective lens take the light emitted back into the camera. The advantage of using the Olympus U-RLA is that, it first acts as a well refined condenser and it also helps with the image forming light gatherer. It also has the ability to remove unwanted light traveling towards the specimen.

The light from the light source travels through to the illuminator and then is collected by the collector lens as shown in the figure below and a through the centre able aperture diaphragm and through to centre able field diaphragm, the light then travels towards the excitation filter to remove the unwanted light that this process is done by choosing the right wavelengths needed and the selected wavelength is passed through to the beam splitter, this mirror is called the dichroic mirror, the placement of this mirror is 45 degree to that of the illumination and the reflection of the light is 90 degrees directly to the specimen through the object lens.



The diagram is of a later version of the reflected illuminator from the same company Olympus and the structure is very similar to the one used in ME3000 except for the fact that, a Tungstenhalide lamp was used in the later versions, but this project has used the light source for the same purpose.

Figure 10: The Schematic of Reflected illuminator used in ME3000.

The light emitted from the fibre passes through the object lens and then passes through to the dichromatic mirror in the mount attached to the back of ME3000, this helps in removing the unwanted light by their wavelengths and then passes through to barrier filter and finally reached the camera.

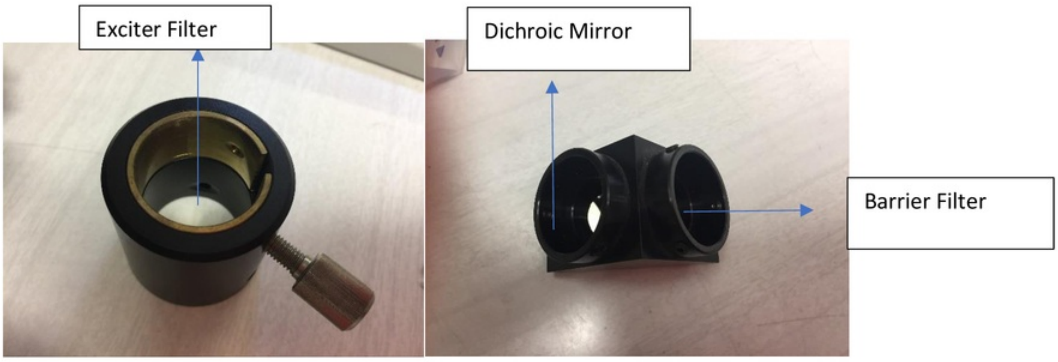


Figure 11: Camera mirrors that take the light emitted from the fibre and modify the light to send to the camera.

ODCM1400C Olympus Digital camera –

The reason to choose a digital image over an analog for this project, the digital image is easier to understand, and analysis compared to an analog image. CMOS image sensor doesn't require more chips for extra circuits like CCD camera as CMOS has the ability to incorporate many circuits on the same chip. CMOS cameras are programmed to switch modes automatically between the still photography and Video recording.

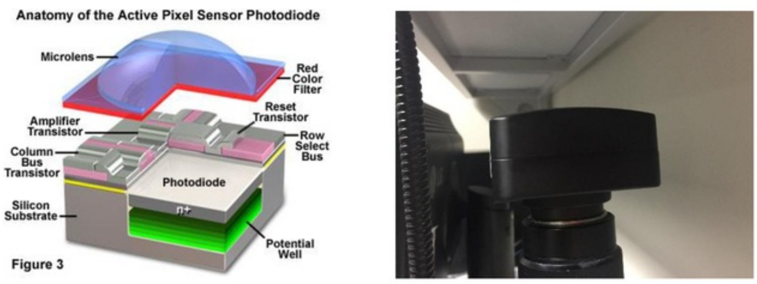


Figure 12: Digital camera ODCM1400C and its Schematic.

The CMOS cameras work on the principle of Photoelectric effect, that happens when photons cooperate with crystallised silicon to help electrons from the valence band into the conduction band. When a broad wavelength band of visible light is incident on specially doped silicon semiconductor, this release a variable number of electrons in proportion to the photon flux density on the surface of the semiconductor. The number of electrons produced depend on the intensity of the light from the light source. These electrons are called in a potential wall until the illumination is finished and then they convert into a voltage. This voltage then moves through an analog to digital converter that helps in the formation of the digital electronic representation of the image that is sensed.

As shown in the figure, the photodiode or pixel plays a crucial role in image sensing, the sensitivity of the camera is the combination of the charge accumulated by the pixel along the conversion rate of photons to electrons and the ability of the device to avoid leakage or spills, accumulate all the charge properly into a space.

Pixels or photodiodes are typically organised in an orthogonal grid, the ranges for this CMOS camera are given with 3 different ranges 4096 x 3286, 2048 x 1644, 1024 x 822, with the order being high resolution to low resolution respectively.

For this microscope, the light emitted by the optical fibre sample through objective lens is focused through a projection lens onto the surface with sensors containing 2-dimensional pixel. Thus, the size of the array and pixel dimensions determine the resolution of the image is determined. The CMOS integrated circuit are only black and white devices that respond to the total number of accumulated electrons in the photodiode or the pixel, the colour that the digital camera detects is by passing of the light through a sequential series of red, green, blue filters in the camera hence the image processing software connected to the camera has the ability to display the image in bot colour and grey.

- **Testing Procedures**

For the light source and the reflected illuminator, the testing was visual by connecting all the available equipment and analysing the happening when ME3000 is switch on and researching the noted finding was the testing procedure for ME3000.

The project need to convert the analog Square wave signal from the Hitachi Camera to a digital signal that can provide the user with an digital image and for this after careful consideration and a detailed research, an DIGITECH analog to digital convertor has been used but due to compatibility issues as the camera was manufactured in 1997, the converter turned out to be useless in this aspect.

Finding from the test procedures include:


Figure 13: Working light source

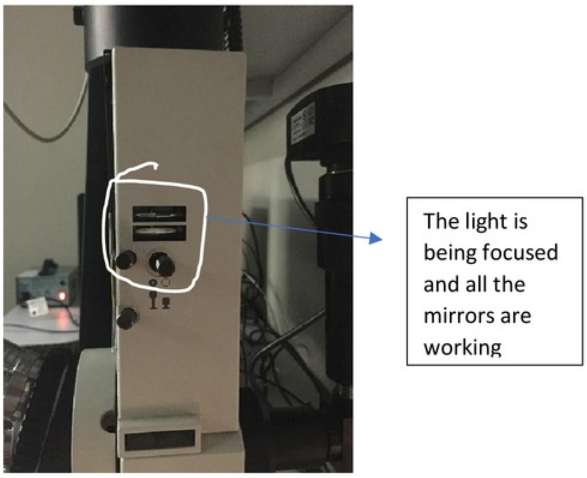


Figure 14: Reflected illuminator

- **Solution**

The solution for the analog Hitachi Camera was to replace the camera itself with a digital camera of C- mount as none of the convertors in the market are compatible with the analog Hitachi camera. Hence the ProSciTech digital camera was used for the purpose of obtaining a digital image of the fibre.

The ODCM1400C ProSciTech Digital camera 14 MP pack includes, an eyepiece for the microscope, image processing software, USB cable and an instruction manual.

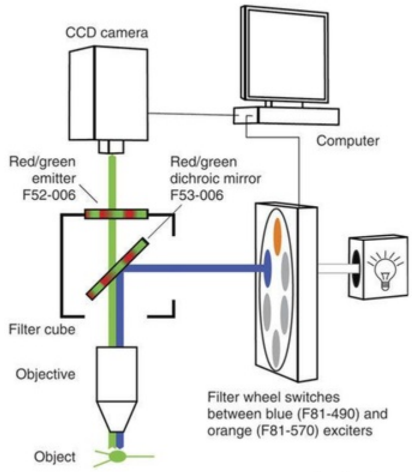


Figure 15: Connection from the light source to the CMOS camera and the Computer Screen.

- **Result**

All the components of ME3000 are in place and the final product was discussed in the section - Computer Operated Optical Fibre Inspection microscope with Image processing software. The solution for the camera has been successful and the image of the fibre sample was successfully shown on the computer screen.

b. Mechanical design of a linear stage

- **Background on Actuator, linear XYZ stage, micrometre**

In this project, the idea was to modify a manual inspection microscope into an automated microscope hence the final result was to have a fully computer operated optical inspection microscope. To achieve this aim, this project used the concept of actuator. This section answers the questions, what is an Actuator? How does it work? How was it useful in this project?

Actuators are rotary encoders that are widely used in electronics, they are popularly known for their accuracy and precision in movement. To operate the movement of the fibre specimen on a mount using a computer, it was necessary to use a motor that will operate on the command from the computer and actuator are simply motors that have great precision, that was well needed for this project.

Actuators work on the principle of converting rotary movement and its angular position into digital signals and are widely used in measurements. Actuators come in 2 types namely absolute or incremental. Actuators used in this project are part of MM-4M-EX stage manufactured by the company called national aperture, the motor is of 13mm diameter with 6-12 Vdc and it is a servo motor with brush. This actuator is of the incremental type where it works on the concept of phase shift to recognize the direction of the rotation and the direction of the rotation can be provided through an external device, in this case was the computer.

Figure 16: Actuators MM-4M-F-25

As this actuator was not an individual product manufactured by national aperture it was very hard to determine its operation or the type but fortunately there was a tag on it that suggested that it was part of MM-4M-Ex linear stage. Knowing this information helped in finding the motor specification and pin arrangements of MM-4M-F-25.

MM-4M-F-25 is specified to be a single-phase shift 90 degree, with a phase relationship of Channel a Leads Channel B when motor rotation is clockwise. From this information taken from the specification sheet provided by national aperture, it can be determined that the actuator uses the Hall Sensor effect. The hall sensor effect, can explained as having magnetic encoders and magnetic sensor that detect the movement and position depending on the magnetic field. The inside of the MM-4M-F-25 can be imagined to be as having magnets placed around the edge of a rotor disc that is assembled in the middle and is attached to the shaft and is positioned in such a way that it is easier for the sensors to detect the change in the magnetic field as the north and south poles alternate when the magnet passes over it.

This set-up of the actuator produces a sine wave output with the frequency that is equal to the shaft's rotational speed. As mentioned in the specification sheet of MM-4M-F, this actuator is of 90-degree phase shift hence the second sensor is set at 90 degrees from the first and this arrangement generates the cosine output, this arrangement helps in detecting the direction of the shaft's rotation and also is capable of obtaining the precise position of the shaft through the sine and cosine signals. This sinusoidal output given by the sensors are changed to sine waves so that quadrature output is gained.

The MM-4M-F is of 16 pulses per revolution with 2 channels namely Channel A and Channel B and has a quadrature output of 64 encoder count. This determines that the location of the shaft can be very precise. the data sheet of MM-4M-EX provides us with the information of the encoder resolution of both the linear and rotatory, this project will focus on the rotary calculations of 16 position resolution encoder of MM-4M-R.

The linear stage used for this project was Model A-LH, is a product of Line Tool.Co and is a 3-axis linear stage. The linear stage was used as a sample holder and was also used for focusing the fibre sample in order to achieve the desired image of the fibre end surface for analysis.

Finally, the use of micrometre for the movement of the sample holder in all 3 directions of XYZ with high precision.

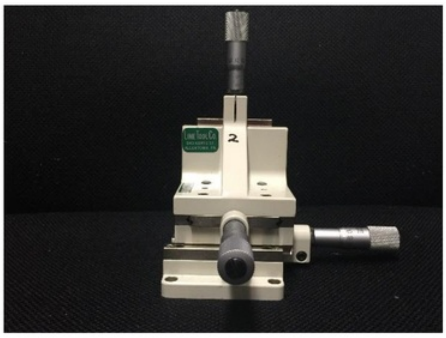


Figure 17: XYZ axes Linear stage A-LH with micrometres

- **Mechanical Set-up of the stage**

For this project, the idea was to use the XYZ linear stage to focus the end surface of the fibre through the objective lens. In order to achieve this, it was necessary to use the actuators that could be controlled using an Arduino board. The use of Arduino board will be discussed in the section below.

The A-LH linear stage was fitted with a micrometre for each of its axis and as the user turns the micrometre in the clockwise the metal plate attached to that axis is controlled to move in downward direction and similar when the micrometre turn in the anti-clockwise direction it move the metal plate in the upward direction for the Y axis of the Linear stage. The idea was to make use of this manually controlled linear axis to be operated with the computer.

- **Modification made**

The modification made for this part of the project was all mechanical and the main aim was to connect the actuator to the micrometres of the Linear stage. In order to use the actuators to control the movement of the micrometre and thus controlling the fibre sample movement.

Important measures that would need addressing for this arrangement would be that, the actuator should produce enough torque for it to be able to rotate or control the micrometre as the micrometre of the linear stage needed a good torque in order to move. There were several ways on how this can be achieved practically, and the ways that were experimented on will be discussed in the testing procedures section.

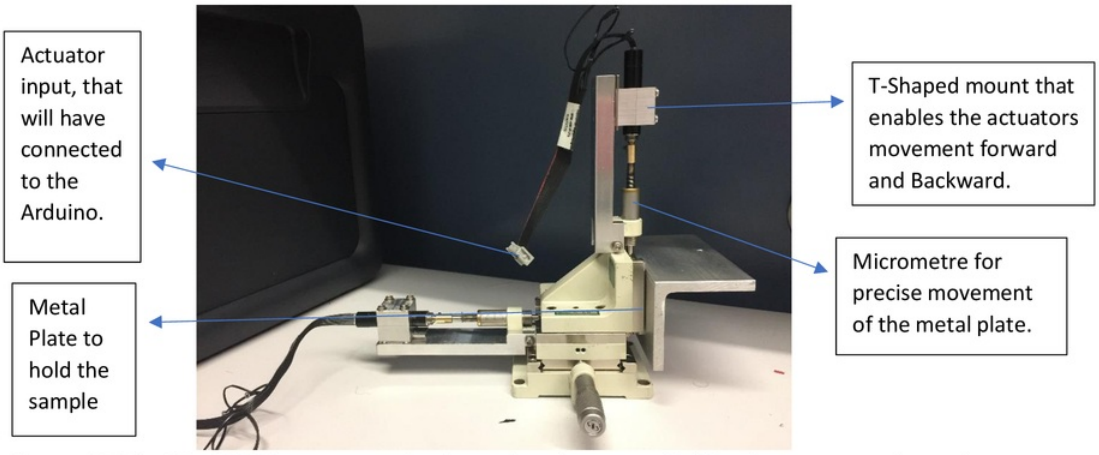


Figure 18: The Final XYZ linear stage that is ready to be controlled by the computer through the Arduino.

The modification was to develop a metal T shaped mount on to the linear stage that fits with the micrometre holder of the linear stage, as shown in figure. This T shaped metal holds the actuator with a set of screws and is aligned with the micrometre and the connectivity between the actuator and the micrometre is held using a set screw. The reason for this metal to be T-shaped was to enable free movement of the micrometre in the direction of the respective axis. For example, considering that the users require the fibre samples to be moved in the downward direction, the T- shaped metal that hold the actuator will help the actuator move down along with the micrometre for the purpose of moving the fibre sample into the desired position similarly when the micrometre is needed to move in upward direction, the actuator follows, and this movement is possible because of the T shaped hook. This example represents only the Y – axis of the linear stage and details of all the axes will be discussed in the working principle.

- **Working principle includes schematics**

The working principle of the computer operated XYZ linear stage is, for this project – 2 actuators (MM-4M-EX) were used the actuator. As shown in the picture MM-4M-EX is a DC motor with a gear box, armature, encoder and a shaft. This actuator uses the concept of Faraday's Law which states that:

Faraday's Law: Any change in the magnetic environment of a coil of wire will cause a voltage to be induced in the coil.

This is how MM-4M-EX actuators can convert electrical energy to mechanical energy, to create the magnetic field in the motor, the current is fed into the armature winding and this interacts magnetic field of the stator created by the permanent magnets and this interaction between the 2 magnets sources the armature rotate.

Figure 19: The Actuator parts or section explained.

The rotatory encoder, background of rotatory encoders is given in the Background section of Actuators, from it – it can be said that they are the devices that convert angular position to a digital code. The position of the motor shaft is read as a digital output of the encoder.

In this project, there were several experimental ways included in controlling the actuator, which will have discussed in the testing procedures. In this section the experiment that was chosen to control the actuator will be discussed. The MM-4M-EX actuator of this project is controlled using an Arduino and motor control Iduino that can control DC, AC and a servo motor all at once. This Iduino motor control consist of 4 H- bridges providing 0.6A per bridge and using the H-bridge the actuator direction (forward or Backward) can be easily controlled, the H-bride consists of high-current switches needed by the motor and these switches can be controlled through the Arduino code.

The motor speed is controlled by the Pulse-Width Modulation, which is avaiable on the Arduino board and can be identified with the symbol (~) next to the pin number. The type of Arudino board used for this project is the Aruidno UNO and the pluse-width modulation pins on this type are 3, 5, 6, 9, 10, and 1. The motor will run when the motor control is sets the duty cycle to 100% and stops when the duty cycle is 0%.

Having more than 1 set of ports on the iduino board to control the actuator has helped in using just a simple, easy and just one board to control both the Y and X axis of the linear stage.

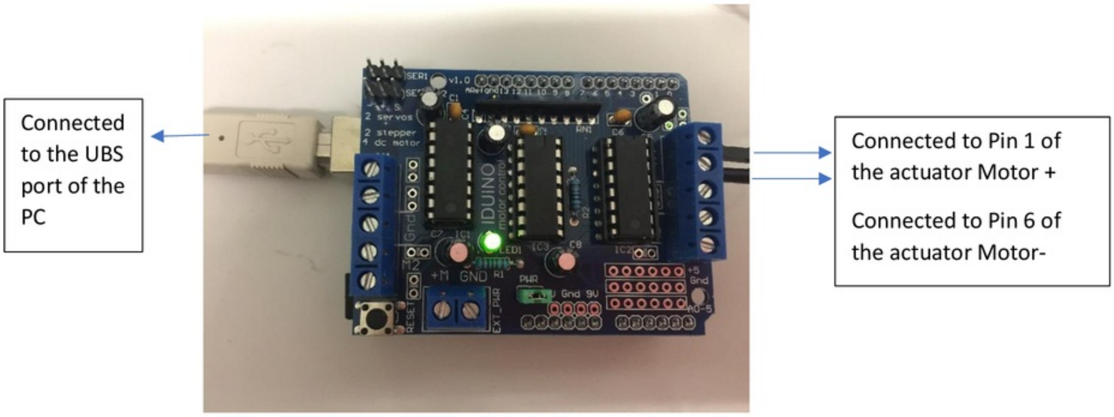


Figure 20: Iduino stacked on the Arduino board to control the actuator.

The next step was to analysis the working principle of the micrometre that was in-built with the linear stage and micrometre used for including 0.50” total travel with 0.01” resolution and are developed for smoot running in their precision-ground ways. All the 3axis of the linear stage consist of ball-bearings in order to reduce rotational friction and to support the axial loads and the radial loads. The final set-up as shown in picture, achieve the aim of controlling the linear stage axes to great precision using computer commands. Unfortunately for this project due to the budget requirement, were only able to work on 2 axes as the budget could only provide 2 actuators. The 2-axis used are the Y axis and the X axis.

- **Testing Procedures**

With the mechanical set-up of the stage mounted on the ME3000, there were several theoretical approaches that were discussed and designed. These include establishing a Gear system on to the linear stage, Mounting the actuators shaft on the head of the micrometre, using the concept of rubber to fit the actuator shaft on to the rubber and then to fit the rubbers other end on to the micrometre head and the final concept was the coupling of the actuator on to the micrometre.

The Gear system, would have been the classic way of developing the stage as the industry uses the gear system to control a wide range of motors to great precision. The mechanical aspect of the gear system would have been very practical and easy to use and understand. The idea was to use the shafts of the actuator and the shaft from removing the micrometre head as the two parallel shafts to transmit the rotation and power between them. As the gear motor produces high torque with good maintenance of low speed motor output, it would have been ideal for this project.

The basic of a gear system is that the rotation from the axis of the 1^{st} gear can be transmitted to the axis of another gear. The teeth of a gear's axis mesh with the teeth of the other gear will create a rotational relationship between the 2 axes hence with one axis spin the other is spun as well. As the shafts of the actuator and the micrometre head is of different sizes the use of spur gears would have been practical. As spur gear can be customized and are easy to make and are commonly used ti transfer the rotational motion from one shaft to the parraelle shaft. The only considerartion to be noted would be the gear ratio as the aim is to obtain high torque.

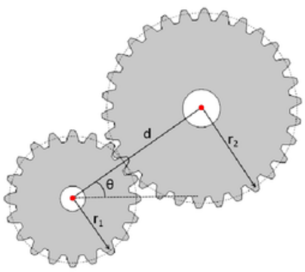


Figure 21: Gear system with the actuators shaft on the small gear and the micrometre shaft on the big gear for the implementation of a more stable linear stage.

The gear ratio – is the ratio between the rotational speed of the actuator shaft to the otational speed of the micrometre head shaft. The easiest way to caucalute this would be by using the formula,

R = N2/N1 (N – number of teeth on each gear)

In the testing procedure the gears that were used consisted of 12 teeth for the actuators shafts and the micrometre consisted of the 24 teeth. The actuator being the input shaft hence N1 is the actuator shaft and the N2 is the micrometre shaft hence the gear ratio was calculated to be 24:12 which is 2:1, which means that every 2 rotation of the actuator the micrometre head turns 1 rotation. The requirement was for the micrometre to turn at the gear ration of 1:1 hence it was required or the gear to be custom made so that the number of gear teeth on both, actuator and the micrometre are the same, but they are of different size, so it would have been necessary to align the gear teeth to achieve the desired set.

Another issue with using gear system was with the fact that mounting 2 sets gear onto the linear stage was mechanically challenging due the limited space and it would require to completely build a gearbox for each of the micrometre axis hence it was decided to go with small mount that could do the same action.

The 2nd testing was with having an industrial rubber to hold the actuator and the micrometre together and this was by having a strong, durable rubber that could hold the shaft of the actuator and can move with great precision as the actuator shaft is very precise and the rubber also required to deliver the same precise output to move the micrometre and should also produce enough torque to be able to move the micrometre. The issues with using this idea was the fact that this was only possible in theoretical aspect and was unsuccessful in the practical experiment as rubber was not precise enough and also was unable to deliver the required torque.

The 3rd testing was with coupling the actuator's shaft with the micrometres head, this test procedure was very straightforward and easy, all that was necessary was a coupling shaft with one end to hold the shaft of the actuator with a set screw and the other end to go on the micrometre head and this idea was developed from the basic concept of the ability to move the micrometre head using humans hand. From this the idea of having the metal couple to do the exact same thing done the human hand but with the help of the actuator and the computer command.

Shaft Coupling

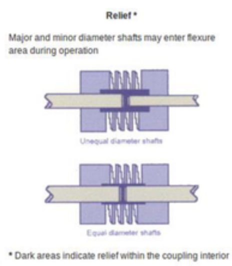


Figure 22

The idea was theoretical possible and after speaking to a mechanical expert it was practical as well, with only one issue which was the movement of the micrometre. The micrometre of the linear stage is built in order to move the metal plate of the linear stage in the desired directions depending the axis being used. And for this the micrometre is required to move in and out and with directly coupling the actuator onto the micrometre would not be able to hold the actuator to move in and out. Due to this issue in hand this experiment was unsuccessful as well.

The 4th experiment and also the chosen one was to build on the idea from the 3rd experiment, the coupling concept. The T – shaped mount was introduced, and this mount was used to solve the problem as the T- shaped mount gave the actuator the ability to move in and out along with the micrometre and this ability has achieved the aim required for this project.

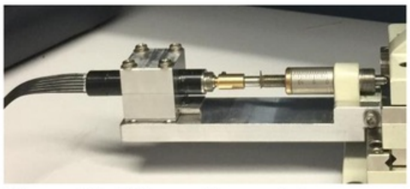


Figure 23: T-Shaped mount to connect the actuator with the head of the micrometre.

- **Result**

The final choice was the coupling of the actuator and the micrometre using the T shaped mount. For this the micrometre's head was taken out and the actuator's shaft was mounted into the micrometre with the help of set screw.

Due to the requirement of removing the head of the micrometre, leading to a decrease in the distance from the point of application of force to the micrometre's shaft and this has led to another requirement which was the actuator's ability to produce 2 times more torque in order to move the micrometre to the desired position with great precision. This can be explained by the equation:

Torque = Distance x Force

Hence the change in the distance from the point of application of the force differs the torque necessary to run the micrometre, and the actuator was not able to produce the torque required to continuously rotate the micrometre to the precision.

- **Solution**

The project didn't have enough time to develop a new mechanical set-up with the issue in hand as the Gear system would take around 2 – 3 weeks to complete.

A temporary solution for the issue at hand was the use of lubricant oil which helped ease the movement of the micrometre to the torque applied from the actuator. This would only be a temporary solution because the lubricant oil is easily wearing off the user needs to keep on lubricating the shaft of the micrometre every other day.

A very well-designed gear box would have both the distance, force and torque aligned perfectly thus satisfying the aim of this project. The only issue would have been the extra space to hold the three gear boxes each for XYZ axes of the linear stage, but the concept would have achieved the precise requirement of the project.

c. Actuator with Arduino

• Background on Arduino

Arduino was first developed at Ivrea Interaction Design Institute as a tool for fast prototyping, for all the student at the institute to use. This development was aimed at the students to be able to develop electronics prototypes even without a background in electronics and programming. Arduino is basically an open-source electronics platform based on easy to use and understand hardware and software.

Figure 24: Arduino board used for this project

Arduino community has grown massively over the years and now Arduino has developed more types of boards to meet the requirements of various industries and the students creative mind. It is compatible with Mac, Windows and Linux, thus reaching to wider audience around the world. Activating the motor and controlling all types of motors is one of the ability of Arduino and the Arduino for this project was used for this very purpose of having the dc motor of the actuator to be controlled by Arduino.

• How Arduino was used to drive the actuator

To drive the actuator using the Arduino, the things required were – the Arduino board, Arduino software (IDE), USB cable to connect the Arduino board to the computer and a controller.

Arduino board is only used as part of the set-up of driving the motor, it doesn't have the ability to directly run the motor. This is because each pin on an Arduino UNO can only take up to 40 milliamperes which is very low, and this current limitation could lead to a condition failure in the voltage regulator on the Arduino board which in turn overheat and destroy itself. Hence it is important to use a controller that can help with current limitations to run the motor.

To drive a dc motor using Arduino, a H-bridge is required and after a thorough research into the Arduino community, the motor controller called Arduino motor controller shield was discovered and this is a newly developed Adafruit module that is primarily focused on driving different kinds of motor. The Iduino that was used for this project had the ability to control2 x 5V servo motor with high-resolution dedicated timers, up to 4-bi-directional DC motors with 8-bit individual speed selection, or up to 2 steeper motors with single stepping. The specification of the iduino include the 5V to 36VDC, it consists of 4 H- Bridges with 0.6A per bridge along with 2 external terminal power interfaces.

Figure 25: Iduino motor controller

This iduino was perfect for the project as it is small, easy to use and had the ability to control more than 1 motor at a time. the iduino had similar pin strcture to Arudino which was explained in the Background section of Arduino. The pins on this motor shield consisted of 6 analog pins that could also be used as digital pins and this pin are numbered from 14 – 19. Digital Pins used to run the 2 actuators are, Pin were M4 and M3 as shown in the picture.

Procedure to run the actuator using the Iduino motor shield include:

Step 1: To collect and arrange all the necessary products, for this project to meet the requirements, it had – An Arduino UNO board, IDE software downloaded, Iduino motor shield, and an USB cable.

Step 2: To carefully stack the Iduino motor shield onto the Arudino UNO while making sure that the pin structure is aligned carefully slowly insert the iduino controller into the Arduino board as shown in figure 24.

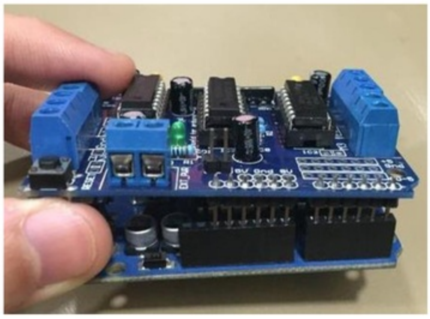
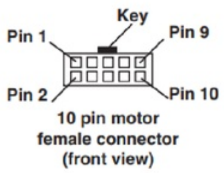


Figure 26

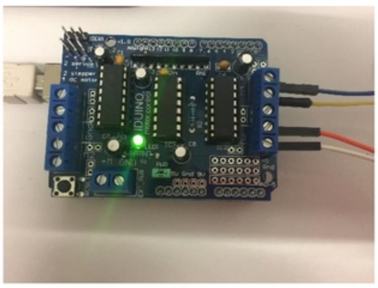
Step 3: To understand the features of the motor shield and for this project, 2 DC motor channels were required and the M4 and M3 channels on the Iduino board are designated for the DC motor, hence carefully unscrew the M3 and M4 channels and insert the jump wires in the 5V and GND pins. Then the 5V pin from the Iduino connects to the Pin 1 of the actuator which is Motor + and the GND of the Iduino will be connected to the Pin 6 of the actuator that is Motor -, carefully make sure that the pins on the actuator are read properly and are inserted firmly.



Pin #	Name	Pin #	Name
1	Motor+	6	Motor -
2	Encoder+V	7	Limit ground
3	Encoder Ch A	8	No connection
4	Encoder Ch B	9	Reverse limit
5	Ground (case)	10	Forward limit

Figure 27: Pin arrangements of actuator MM-4M-F-25

Step 4: To connect the Arduino USB port to the Back of the available USB port of the CPU. Once the connection is set.



The Blue wire – Pin 1 Motor + of Y-axis Actuator.

The yellow wire – Pin 6 Motor – of Y - axis Actuator

Red wire – Pin 1 – of X-axis Actuator

White wire – Pin 6 – of X-axis Actuator.

Figure 28: Arduino connection to the Computer USB port and to the Actuator

Open the Arduino software and start coding. The code used for this project is-

The basic code to run a motor is given in the example section of the Arduino software and after a few experiments with understanding the example code and some help from the coding instruction of Arduino provided for beginners. The following code was developed and has been checked for errors and has been tested to be successful, to control the actuator.

#include < AFMotor.H>

AFMotor.H is an Arduino library that was specially developed for controlling all types of motors. This library is free to doenload from the Arduino board and can be used to control DC motors, AC motors and servo motors. The library includes pre-defined words such as “SetSpeed”, “run”, “FORWARD”,”BACKWARD”, “RELEASE” and “incomingByte” , the functions of these words will be discussed in the code below.

AF_DCMotor Motora(4);
AF_DCMotor Motorb(3);

This section is used to connect the DC motor to the Iduino, Iduino has 2 set of dedicated ports for DC motors, namely M4 and M3. The code above reads that the Y-Axis actuator named as Motora for code identification is connected to port M4. Similarly, the X-axis actuator named as Motorb is connected to port M3.

char val;
int incomingByte;

the char val command is to receive the data that will be entered in the serial monitor of the Arduino software, this function is necessary as the movement of the actuator is connected to the keypad and once the designated number of the movement is typed in this command will enable the number to be read. The command int incomingByte will then read the number and actives that loop.

```
Void Setup() {
        Serial.begin(9600);
        Serial.Println (“ The Motor is now ready to use”);
        AFMS.begin();

Motora.SetSpeed(150);
Motora.run (FORWARD);
Motora.run (RELEASE);

Motorb.SetSpeed(150);
Motorb.run (FORWARD);
Motorb.run (RELEASE);

}
```

This is the section where we control the speed of the motor, the 9600 is the serial library at 9600 bps and the Serial.println command is used when the user need to display a message on the serial monitor which is the output. Hence the function of this command is to display the message “The Motor is now ready to Use” when the code has been successfully uploaded to the board.

The SetSpeed command is a special command for the motors, this command is used to control the speed of the motor as the name states. With Arduino’s the highest speed is 255 and the lowest being 0 where the motor stops running. The command FORWARD is also a motor Arduino command that activate the motor in the forward direction, the BACKWARD command activates the motor in the backward direction and similarly the RELEASE command is used to stop the motor. The FORWARD and BACKWARD commands don’t know that they are running the motor in those directions so, in this project it was important to understand the direction that was necessary to move the actuator in for it to act like expected.

```
Void loop() {
        Uint8_t(i);
If (Serial.available()>0) {
        incomingByte = Serial.Read();
If (incomingByte == '8' ) {
        Motora.run (BACKWARD);
        Motor.setSpeed (10);
}
Else {
        Motora.run (RELEASE);
}
```

This section is to connect the keypad numbers to the movement of the actuator. The incomingByte command checks if there is any serial data stored and then returns by reading the oldest bytein the serial buffer. For example, number 8 was typed into the serial monitor, then the incomingByte is read and the command that says run the motor BACKWARD will be active and the speed is set to be 10 sec as we need the motor to run very precisely, the movement observed on the actuator for this command would be that the actuator will move in the upward direction. The else function is used for accident typing. For example, number 1 is entered into the serial monitor because there is no loop for number 1 this line of else will be active and the motor stays still.

```
If (incomingByte == '2' ) {
        Motora.run (FORWARD);
        Motora.setSpeed (10);
}
Else {
        Motora.run (RELEASE);
}
```

This command line works very similar to the one above, for example – number 2 is entered and this loop will be active, and the motor is commanded to move in the forward direction at 10 seconds speed – observation on the actuator is that the actuator moves in the downward direction.

```
If (incomingByte == '4' ) {
        Motorb.run (BACKWARD);
        Motorb.setSpeed (10);
}
Else {
        Motorb.run (RELEASE);
}
```

This command is the same as discussed above – number 4 will activate the actuator in the left direction at 10 seconds speed.

```
If (incomingByte == '6' ) {
        Motorb.run (FORWARD);
        Motorb.setSpeed (10);
}
Else {
        Motorb.run (RELEASE);
}
```

Like the above – number 6 will activate the actuators movement in the right direction at 10 seconds speed.

```
If (incomingByte == '5'){
        Motora.run (RELEASE);
        Motorb.run (RELEASE);
}
}
}
```

The last command is number 5, this command is to stop the motor and should be used in between the direction changes. The RELEASE is a pre-defined command used to stop the motor.

There were several other programs that were used to run the actuator at the experimental level all the codes were included in the Appendix.

- **Problems Faced**

The microcontroller used for this project between the Arduino board and the Actuator were different from the final Motor shield. The 3 microcontrollers used were namely MC-3SA servo amplifier system by National Aperture, Micro-Mini Controller by National Aperture and finally the Motor Shield by Arduino.

The first experiment was with a range of things such as the micro-mini controller, the controller was only of one axis and the information and the part of the controllers like the cable for connectivity were missing hence the idea of using the micro-mini controller to drive the actuator was dismissed.

It was then necessary to contact the actuator providers, the national aperture as the actuator was a part of a linear stage MM-4M-F and there was very limited information on MM-4M-F-25 actuators online. National aperture did disclose information on the actuator, such as the best suited controller to drive the motor would have a quadrature input to determine the position and a DC output to drive the motor. They have also provided a list of controllers that national aperture has developed from the makers as National instruments, Faulhaber, but all of these controllers proved to be over the budget and hence were not used.

The second experiment was with MC-3SA servo amplifier – The reason, MC-3SA was chosen to be the control was since both the actuators and MC-3SA were developed by national Aperture and hence the compatibility between the 2 products was secure. The MC-3SA unlike the micro-mini controller was a four-axis controller which full-filled the requirement of the necessity to control 3 axes of the linear stage. The MC-3SA had a compatibility socket at the back of the device that could be directly inserted into the input of the actuator.

Using the manual of MC-3SA the Pin Out of the amplifier was noted to be:

Name	Abbr.	Pin No
Signal Ground	GND	1
+/- Motor Control	MOTORn	2
Encoder A Output	ChAn	4
Encoder B Output	CHBn	6
Computer +5V	+5V	50

Table 2: Pin arrangement of MS-3SA

From the table, by using the above pins it is possible to control and drive the motor. The connector is also capable of returning both the encoder outputs from the motor to provide feedback. Hence the controller can be used to drive the actuator with the desired speed and direction.

For set—up circuit using the MC-3SA, the Arduino will first send the required signal to the servo amplifier to rotate the motor and this was possible by using the Pulse Width Modulation pin of the Arduino which is designed to produce 255 different motor speeds. The issue with using the PWM signal is that it emits a high pitch hum during operation even when the motor is run on very low speed. This occurrence is due to the PWM signal being an oscillating signal and is causing the components in the circuit to vibrate. In order to reduce the noise or to completely avoid it, it was necessary to use RC filters to average out the oscillation generated by the PWM signal in to a voltage that is more stable.

In order to not complicate or develop a complex solution for this, an op-amp voltage follower was introduced so that the output voltage from the op-amp would correspond with the output of the filter. This solution was applied to all the PWM pins used for this circuit. This circuit also needed 2 Arduino boards as to control both the forward and reverse direction of the motor. This assembly would allow op-amp to produce the output of positive voltage when the motor is commended to move in the forward direction and similarly move in negative voltage when commended with negative direction. The chip used was single LM348N, it is of 4 op-amps in one circuit for easy implementation.

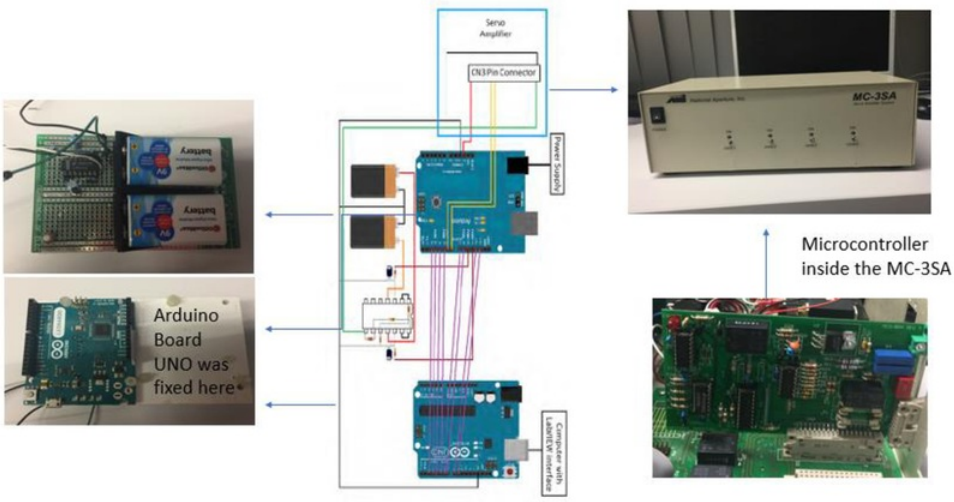


Figure 29: Experiment with MC-3SA to activate the actuators

The resistors (10KΩ) was used to limit the currents to a low 0.1mA. This circuit also needs bipolar power supply, and this was provided by two 9V battery in series with the connection point. The voltage was +9V and –9V for each direction.

The Arduino code developed for this is provided below and was successful. The problem with this set-up was that halfway through the project the Servo amplifier was producing only 1.4V instead of 5V from its motor pin. As the actuator needed 5V to drive its motor the servo amplifier thus was not useful anymore to drive the actuator. It was than necessary to plan for a new way to control the motor and for this, it was necessary to get in contact with the suppliers of MM-4M-EX, in order to determine the motor specification and further details. With better understanding of the working of the actuator and the Arduino and from the help of the Arduino community, the iduino motor shield was identified to be the best solution to drive the actuator.

As explained in the steps above the connection between the Arduino, the iduino motor shield and the actuator turned out to be very well compatible and the circuit had no issues during testing.

- **Final Design with Results**

The final design was to use the Iduino motor shield and the circuit is shown in the picture below.

The actuator speed and direction are controlled by the keypad of the computer.

Number on the Keypad	Direction
2	Motor 1 (Y-Axis) – Forward Direction
8	Motor 1 (Y-Axis) – Reverse Direction
4	Motor 2 (X-Axis) – Forward Direction
6	Motor 2 (X-axis) – Reverse Direction
5	To stop the motor

Table 3: The keypad numbers associated with the direction of the actuator's movement

There is no limit switch on the actuator as the operator can move the axis at the desired speed and the program provides the ability to stop the axis at any given point. Thus, the limit switch would be determined by the way the operator uses the linear stage.

d. Image Processing Software

- **Background**

Image processing software is designed for the analysis of the image taken from the Digital CCD camera of the microscope ME3000. The purpose of the software is to capture the image and covert it to digital form and changes or work can be done on the image using the software tools.
There are many varieties of image processing software in the industry, the image processing software used for this project was the software that was included with the digital camera purchased.

In general, an image processing software differs depending on the industry it is being used in and the purpose it is being used for. All the software's have the same basic working principle and the additional requirements are met by adding to the basic. the two types of digital image software's would be one for the microscopy and the other for general imaging.

As this project is on microscope, the image processing of the digital images obtained through the microscope will be of interest. The microscopy digital image software is widely used in a range of industries such as medicine, biological research, cancer research, drug testing and of course optical fibres. Up until 1990's and even the 2000's most of the microscopy application was done using analog video camera, even ME3000 was designed with the analog video camera. These cameras needed to use a frame to digitize the images, these cameras usually provided the video output at full image frame and the frame rate used to be 25-30 frames per second in order to allow the live video recording and processing.

The digital CCD camera used now provide pixel intensity data with a resolution of
12 – 16 bits and this is much higher than imaging products. The camera of today are of higher speed and they have the ability to observe the image or video in real-time, store the recording, playback, save the image and also analyse different images next to each other.

- **Software Information**

The software used for this project is called the Digital Camera Solution Disk, with the application software – ToupView and ToupLite.

ToupView and ToupLite - is a software for the digital Eyepiece camera, the ODCM1400C is a digital camera supplied by Proscitech that comes with a eyepiece camera and the software ToupView is used when the eyepiece is used on the microscope. It comes with manual, that includes easy to follow instructions.

The image processor of the digital camera was developed used a wide range of programming language to give the user the ability to record, capture, manipulate, store, and analyse the image using a wide range of tools in the tool section. The list of the programming languages used in the development are listed below:
Software Development Kit – Native C/C++, DirectShow, Twain, .Net, C#, VB.net and Labview . This software is compatible with all versions of windows, Mac and Linux.

- **Testing procedures**

The steps taken to test or inspect the fibre under the microscope and to deliver the image from the microscope onto the software were:

Step 1: The digital camera pack includes – USB cord and a CD for drivers' installation. The CD Digital solution once detected by the computer will open on windows screen asking for the permission to install the CD drivers on to the computer Click Next – Next – Finish.

Step 2: After the installation process, it will create image processor icon on the desktop, double click on it and the software is now ready to be used.

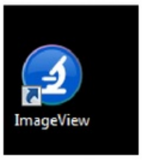


Figure 30

Step 3: Connect the USB cord form the Digital camera to one of the USB ports of the PC. Once this is done, the user will be able to see the camera detected in the software, and the screen is shown to be:

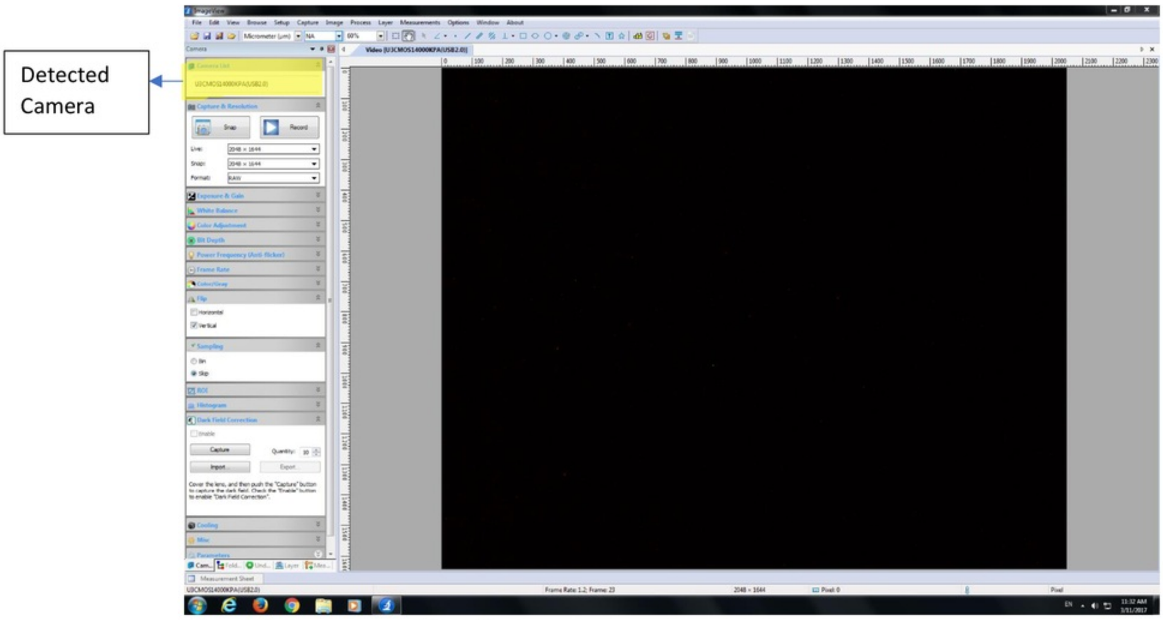


Figure 31: Image processing software

Step 3: The operator should now be able to see the image of the fibre, the fibre as a sample is a D-shaped fibre. The tools on the left-hand side are used for better picture quality, rotation of the screen, the user can choose the image in grey or colour, the brightness of the image can be changed, and the frame rate can be changed too.

In the picture , The “Capture” is used to capture a still image from the real-time view of the fibre sample and if it is necessary to observe the changes of the sample then the option “Record” can be used to record the video for analysis it later. Both the image and video can be saved to your desired location on the computer or on an external device.

There is Dark field correction option that can be used when the image secure from the camera is too small when compared to the camera frame then the extra space on the frame is called the dark field and this can be adjusted using this option.

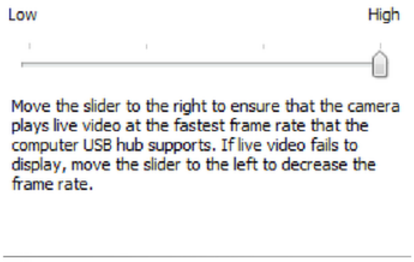
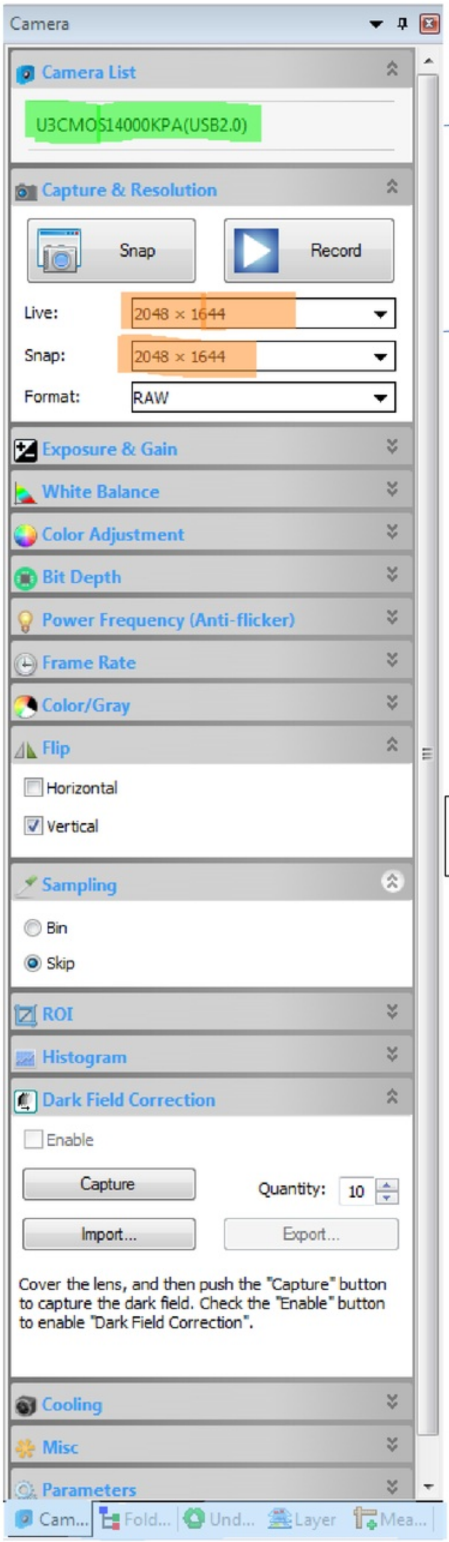


Figure 32: Frame rate

It is important to adjust the frame rate as often when test on this software, the error message that states “the image is beyond the frame rate. Please lower the framerate or connect to a different USB port” this error can be dismissed by lowering the frame rate from the picture.

Figure 33: Choice of colour/grey and horizontal/vertical

The image capture through the camera can be seen on the software as a Gray – Black and gray image or the as a coloured image, the coloured image. Both the Flip and Colour were tested during the experiment stage and results are shown in the Result section below.



This section lists all the camera connected through to this software and also details the port it is connect through.

The software is designed for three ranges of resolution that the image can be seen with, the arrow down here allows the user to change the image resolution to the desired option-

Highest Resolution – 4096 x 3286

Medium Resolution – 2048 x 1644

Low Resolution – 1024 x 822

The format can be either RAW which was used for this project or RGB.

Figure 34: Tools of image processing software

Fields –

Camera is used for the tools that can be used while the camera is detecting an image or video

Folder is used when the user needs to access the files saved for future reference or analysis

Measurement is a tool that can be used when measure the size or the dimension of the image.

Layer is used when the user is required to use more than one layer on the software.

- Result

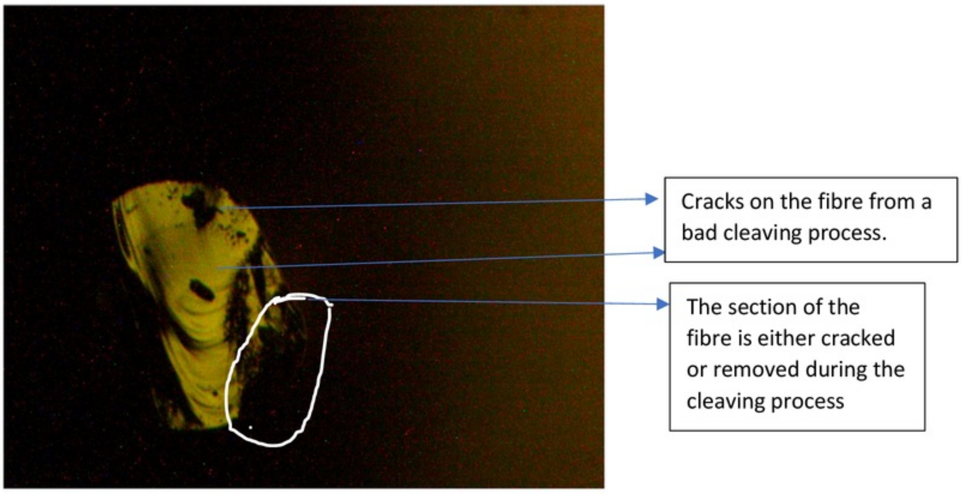


Figure 35: Image of the D-Shaped fibre from the image processing software

The digital image of the D-shaped fibre through ME3000

The image of the fibre can be considered to be of bad cleave as the image doesn't show a smoot, flat surface. The cracks and hackle and the missing part of the complete D- Shape proved that this fibre has not been cleaved properly hence rejected. The ability to identify a bad cleaved fibre at the early stage will help save and materials that will be involved in the process of developing this fibres into connector or for any other industrial purpose.

e. **Computer Operated Optical Fibre Inspection microscope with Image processing software**

- **Set-up of the System with schematics**

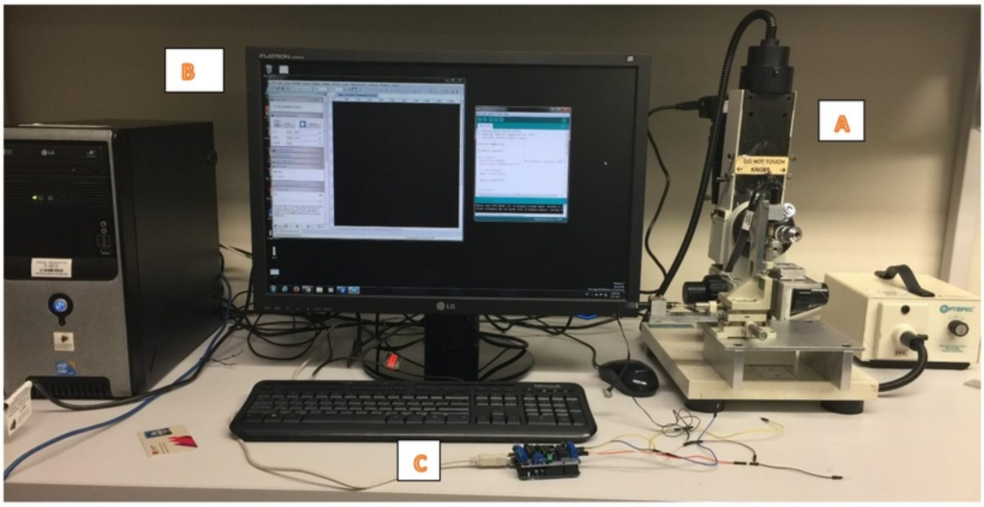


Figure 36: Computer operated ME3000 inspection microscope with image processing software

A: The ME3000 is an inspection microscope for optical fibre, the microscope is supplied with a continuous light from the OptiSpec light source, this light transmits through a illuminator that focuses the light through to the objective lens and the towards the fibre sample and the light emitted from the sample is taken back to the digital camera through a series of filter mirrors and the image is then display on “B” screen in the image processing software.

The ME3000 comes with a linear stage mount o hold and adjust the movement of the sample.

B: The screen consist of the image processing software on the left and the Arduino serial monitor on the right for the control and the output.

C: The microcontroller to control the movement of the linear stage.

- **Working principles**

The ME3000 is a computer operated optical fibre inspection microscope with image processing software, from the set-up picture above the working principle of this system can be explained. The Input of ME3000 is the Light source that supplies continuous light to the illuminator that consist of mirror and plates that guide and focus the light in a vertical path and the light is guided through the objective lens to the sample of the fibre. Once the light is seen through the objective lens, the alignment of the fibre sample needs to right on spot down to micrometres as, to obtain a clear image of the fibre the light should fall directly on to the fibres end surface with perfect alignment and in order to achieve this, the linear stage will be used. The computer screen will display the image processing software on the left-hand side and the Arduino code along with the serial monitor on the right-hand side of the screen.

While the image processing software display the real time of the D-shaped fibre the user would use the serial monitor of the Arduino board to control the actuator on the XYZ axes. To move the fibre upwards the user must press the number 8 in the serial monitor of Arduino and this will activate the Motor 1 of the actuator built on the Y -axis and the actuator will keep rotating until the user types in number 5 from the keypad to stop the motor. Because the real-time on the image process software is delayed by about 10 seconds, it is recommended to wait for 10 seconds after changing the position of the fibre using the linear stage so that the Real-time will be able to display the new image after the change has been made.

The way Arduino serial monitor, and the keypad work is, the serial monitor will say “Motor Ready for Use” when the connection between the code and the Arduino board is set. Once this message is confirmed the number 8 is used to activate the Y axis and the fibre will move in upward direction at a very slow speed, when number 2 is typed into the serial monitor, this will active the Y -axis in downward direction. The number 4 will active the X axis and the sample will move to its left and the number 6 will move the sample to right on the X- axis and it is very important to type in number 5 into the serial monitor before moving from one direction into the other and it is also necessary to pay attention to the limit of the micrometre shaft in order to STOP the actuator in time.

Once the light hits the D-shaped fibre the light is then moved into the camera through a series of mirrors and these mirrors help in guiding the light toward the digital camera mounted on to ME3000 using a C-mount adapter and the mount also has the ability to zoom in and out to obtain the desired image. From the digital camera the image is read on the image processing software at real-time, the real time can be captured as a still image or can be record as a video and is saved for future reference and analysis.

There are 5 different lenses with different magnification that can be used to see the image of the fibre through and the 5 different ranges will help with the analysis of the image in different aspects. The lenses are attached to a rotatory mount that enable the user to select the lens and the light is shone through that particular lens onto the image thus obtaining the image of the fibre to that magnification.

- **Experiments using the System**

The Aim of this project has been met as the system works to meet the project requirements and this was tested in the lab by a series of ways.
The 1st experiment was with the movement of the linear stage while the image of the fibre is being displayed on the monitor. The only set-back on this would be the note of the time difference between the movement of the linear stage to the image displaying time from the camera. The operators need to wait for about 10 seconds after each change in the movement of the stage so as to see if that position is focused or not and after 10 seconds if the image on the screen is still unclear or blurred only then the user can move onto changing the position of the fibre again.

Due to the limited budget available the ME3000 linear stage, X and Y axes can be controlled by the computer but the Z axis needs to manually be done by rotating the knobs on the side of the ME3000 to access the Z axis for the purpose of perfect focus. On the experimental level, once the Z – axis was aligned properly, it can be locked into that position and the small knob extension can be used to clear the image or video to precision.

- **Results**

Before experimenting with the fibre there were other metal and paper that were used to find the focal length of the lens.

The focal length was an important aspect as this changed every time, a different lens was used, and the focus was lost even with a slightest movement. The first

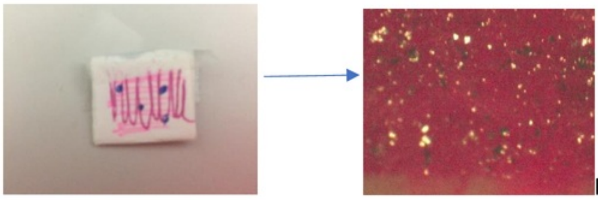


Figure 37: Trail 1 with paper

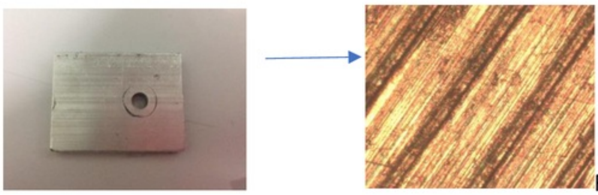


Figure 38: Trail 2 Metal

This have helped in calculating the focal length of the lens and it was calculated to be very close and is 1.4 cm for all the lens and the focus from this will be a bit blur but just a small movement in the Z axis should fix the blur and the final image should be very clear, as shown in figure.

The failed attempts to see a clear image of the optical fibre include:

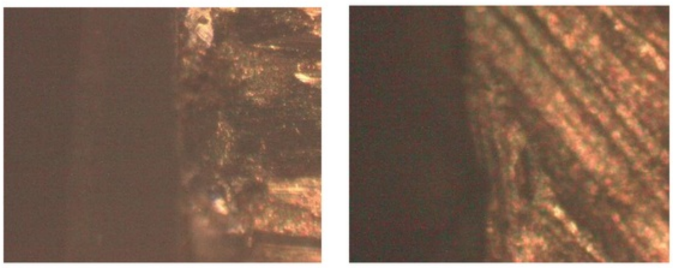


Figure 39: Failed attempt of fibre through ME3000

Due the effect of the focal length microscope was unable to detect the fibre but was able to detect the metal that was holding the fibre sample. It need a great deal of precision in an attempt to see the fibre.

- **Conclusion**

The final set-up of the project meets the requirement as the aim was to set-up a computer operated inspection microscope and the criteria was to work on ME3000, that need to be re-engineered with very limited information available. Due to this this project needed a great deal of research and involved a lot of learning. As I was new to the concepts of photonic, microscope, and mechanical aspect of the engineering, it involved a lot of work to go into just understanding the components and their purpose in the set-up of the microscope. For the purpose of this project I have learned to code with Arduino and also needed to understand the image processing software of the digital camera.

With better time management and organisation, I would have been able to change the linear stage from the T-shaped mount to Gear system as towards the end of the experiments with the T-shaped mount it seemed unreliable as the actuator need a lot of lubricant oil for free movement and with time a gear system would have been more reliable as there won't be a contact between the shafts.

The digital camera used is very easy to understand and the connectivity and picture quality are very reliable, the CMOS (complementary metal-oxide semiconductor) system of the camera even though CMOS camera are more vulnerable to noise due to their traditional manufacturing process the ODCM1400C had high quality pixel.

f. Safety Precautions

- Eye Safety: the light emitted by the optical fibre is infrared and directly looking at the light emitted by the fibre can cause damage to your eyes and because the ME3000 does have an infrared filter it is safe to use on a naked eye, but it would be best to not directly look at the light at the end surface of the fibre. The infrared fibre optics wavelength is at 1300 -1550 nm range and it is doubtful to damage human retina but has the possibility to harm the lens and cornea.

- Hands Safety: Using bare hand to handle the fibre is not recommended as the broken ends and the scraps of the fibre are very sharp and they can pierce into the skin very easily. The broken ends and scraps are very common during the splicing and termination of fibre. In some cases, there is chance of fibre piercing into the skin and with a slight movement they break off and it is very hard to identify and remove them from the skin but using magnification glasses and tweezer, we should be able to find them. Its best use hand gloves that stick to your hand so that you won't trouble handling fibre that are very tiny.

- Fire safety: while working in the lab there is always a chance of explosion due to the presence of combustible substances, that are very vulnerable to fire and even wrong voltage or current connection can lead to an explosion hence it is important to make sure that you have full attention toward the components you are working on and when in doubt it always best to check with your supervisor.

- Electrical Safety: This project requires the use of step- down transformer in order to convert the Australian standard voltage to the American Voltage. It is important to understand the standards and the output power of the equipment so that accidental connections can be avoided. Drinking water next to switched on electric components should be avoided as a spill can destroy the components and also could cause a spark. It is also important to switch off the power before touching or changing equipment around as the current flow could cause a shock

- General Safety: This would to always wear closed shoe while working in the lab, to protect our feet in case any of the equipment slips and fall on the feet. Always be aware of your surrounding and if danger permits leave the premises immediately. Safety is very important while working in labs thus, be careful.

g. Application of the device in real world

For many years optical have been cleaned and inspected to allow proper channel of light through them. To reduce the loss of the light passing through the fibre and to achieve lower attenuation through adaptors it is very crucial to inspect the fibre's end surface.

The ME3000 inspection microscope was designed to ease up the processes of inspecting the fibre optics with great ease and fast. The loss of light is the main issue when it comes to bad or damaged optical fibres. The optical fibres are said to bad when there are crack, dust, oil, and water on their end surface and these are not visible to naked eye hence the use of inspection microscope will help with analysing the fibre easily.

Inspection microscope with eyepiece are easy to use very similar to the ME3000, which uses the camera and the image processing software in order to take a good look at the end surface of the fibre. The only advantage of having a camera than an eyepiece in an inspection microscope would be that, it is a known fact that optical fibres emit inferred and are harmful to human eyes hence having a camera for the analysis instead of an eyepiece would protect the user's eyes.

After the termination and spicing process, directly connecting the fibres without inspecting them and only replacing them after a failed attempt to pass light would be a very unstable approach and could cost money and time, as it is necessary to check unconnected all the fibres and replace the fibre with a new one and then connect them back again.

h. Comparison between the most industry products to this System

ME3000, compared to the inspection microscope of optical fibre of tody is considered to be very traditional made. As from the use of the parts in ME3000 it can be considered to be at least 5 years old and most inspection microscope today are portable one with a mini screen that is loaded with the image processing software.

Advantages of having a desktop Inspection microscope over a portable one is:

- With the desktop inspection microscope the environment such as the lightning of the room for better analysis of the fibre can be easily accommodated for, because with light inspection microscopes such as ME3000, light plays a very important role and during testing the image with a lit room and dark room was different. With a lit room the fibre had a shadow over the surface as the light in the room reflected in the image, once the room was dark the image was of better resolution and image detailed out even the slightest scratch in the end – surface of the fibre. Hence the desktop inspection microscope has the environmental control over the portable one.
- The 2^{nd} advantage would be the cleaving process, in lab that is set up for the purpose of inspecting optical fibre, there is high chance of that lab having all the required equipment needed to turn a bad fibre into good one in less time than it would take a person with a portable inspection microscope being used at the site. In the lab with safety the operator has the chance to quickly cleave the fibre and inspect it again in a matter of time.

Disadvantages of having a desktop inspection microscope over a portable one.

- As the name suggest, a portable inspection microscope is easy to carry around and in case of an issue with a bad end surface of a fibre at an employment or any site, it is very easy to identify the bad fibre using the portable or hand held inspection microscopes.
- Other major advantage would be the maintains of the device, with ME3000, there are so many parts, components and equipment involved in it that the maintains of it would consume time where as with the portable or relatively smaller microscopes with less components would consume less time.

i. Future development

To improve the ME3000, computer operated inspection microscope with image process software would be to work on the stability and working of the linear stage. The idea of using the XYZ linear stage axis and the actuator to move the stage is good but the only issue faced was the mechanical technology and learning needed to understand the best way to connect the actuator with the micrometre of the linear stage.

The T-shaped coupled used in this project is not reliable and wouldn't be suitable to use for long term. Hence this report also indicated an alternative way of approaching the problem using Gear system. Even though theoretically the gear system seems to be of a good fit with the microscope and the also meets the requirements needed. Unless it is built and tested on the results cannot be determined. Hence a good mechanical set-up of the linear stage with increase the value of the inspection microscope for better. The T-shaped coupling is also a good mechanical assembly with a high-powered actuator. The actuator provided for the project MM-4M-F-25 is only of 6 V and the torque delivered is not enough to the run the micrometre as the micrometre of the linear stage is stiff and the stiffness of the micrometre can't be changed as the stiffness is needed to move and hold the metal plate in place. Hence a high voltage actuator would solve the issue as well, along with this adding a distance to the micrometre and the actuator should also help in delivering the torque required to move the system as discussed above.

Another development could be the ability to code and develop a small microcontroller chip in a box that can be inserted into the input of the actuator and this chip would connect to the computer via the wi-fi and the can be controlled using the commands from the computer. This chip would be a replacement for the Arduino board as the connection between the Arduino board to the actuator and then the other of the board to the computer seems to be too much equipment involved. To be able to this a lot more research and more testing and trails would be needed.

The last development which could be possible but not necessary would be to use a smaller version of the light source that could directly connect to the top of the reflected illuminator so as to not use the box of light source which would save space and save effort in case of moving the ME3000 inspection microscope to another location. That being said the light source works perfectly fine and does a good job suppling continuous light to ME3000.

With this project the ME3000 was developed from a manual inspection microscope with analog output into a computer operated inspection microscope with digital image processing software and there is still more improvements and development that can be done on ME3000 for faster, precise and easy results.

j. Bibliography

- https://www.allaboutcircuits.com/projects/use-an-arduino-to-control-a-motor/
- Ackroyd Thomas Thesis research report 2015.
- www.geartechnology.com
- https://www.intechopen.com/books/current-developments-in-optical-fiber-technology/faults-and-novel-countermeasures-for-optical-fiber-connections-in-fiber-to-the-home-networks
- http://www.thefoa.org/tech/ref/testing/test/scope.html
- http://www.electronicdesign.com/components/understanding-resolution-optical-and-magnetic-encoders
- http://www.machinedesign.com/sensors/basics-rotary-encoders-overview-and-new-technologies-0
- http://www.murri.fi/pdf/Lika/Lika-Inkrementaali-Absoluutti-Lineaarianturit.pdf
- http://www.elecrom.com/introduction-arduino-uno-uses-avr-atmega328/
- http://howtomechatronics.com/tutorials/arduino/rotary-encoder-works-use-arduino/
- https://www.allaboutcircuits.com/projects/use-an-arduino-to-control-a-motor/
- http://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/
- http://hardwarefun.com/tutorials/controlling-speed-of-dc-motors-using-arduino
- http://www.instructables.com/id/Control-a-Small-Linear-Actuator-With-Arduino/
- http://www.wikilectures.eu/w/Magnification_of_optical_microscope
- https://www.olympus-lifescience.com/en/microscope-resource/primer/techniques/fluorescence/reflectlightpaths/
- https://www.olympus-lifescience.com/en/microscope-resource/primer/anatomy/reflected/
- https://www.ccdworld.com/wp-content/uploads/2016/09/kp-m1a_om.pdf
- http://opensourceecology.org/wiki/Shaft_Coupling
- https://www.auselectronicsdirect.com.au/arduino-servo-motor-controller-shield?gclid=EAIaIQobChMI6LaO9deo1wIVgx0rCh2XMAqjEAQYASABEgJPCvD_BwE
- https://learn.adafruit.com/adafruit-motor-shield/using-dc-motors
- https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino?view=all
- http://www.instructables.com/id/Adafruit-Motor-Shield-Tutorial/
- http://www.naimotion.com/mm4mf.htm
- https://micro.magnet.fsu.edu/primer/digitalimaging/cmosimagesensors.html
- www.olympus-ims.com
- www.labcompare.com
- www.opticalfibre.com.au
- www.microbehunter.com
- www.flukenetworks.com
- www.thefoa.org
- www.micro.magnet.fsu.edu
- www.optique-ingenieur.org
- www.siliconimaging.com
- www.electronics.howstuffworks.com

- www.webstore.diaginc.com
- www.ncbi.nlm.nih.gov
- www.mdpi.com
- www.phoenixphotonics.com
- www.fibre-optic-tutorial.com
- www.proscitech.com
- www.micro.magnet.fsu.edu
- www.naimotion.com
- www.instructables.com
- www.learn.adafruit.com
- www.auselectronicsdirect.com.au
- www.opensourceecology.org
- www.cs.olympus-imaging.jp
- www.gdlp01.c-wss.com
- www.ccdworld.com
- www.zeiss.com
- www.labx.com
- www.svtechparts.com
- www.spachoptics.com
- www.hitachikokusai.com
- www.thefoa.org
- www.phoenixphotonics.com

Consultation Meetings Attendance Form

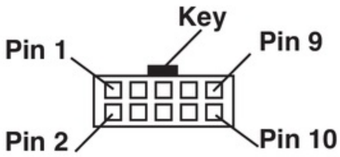
Week	Date	Comments (if applicable)	Student's Signature	Supervisor's Signature
2	8/8/2017		[signature]	[signature]
3	15/8/2017		[signature]	[signature]
3	17/8/2017		[signature]	[signature]
4	22/8/2017		[signature]	[signature]
5	30/8/2017		[signature]	[signature]
6			[signature]	[signature]
7			[signature]	[signature]
8			[signature]	[signature]
9			[signature]	[signature]
10			[signature]	[signature]
11			[signature]	[signature]
12			[signature]	[signature]

MTR-10-10E MicroMini™ Motor (10 mm diameter, 6 Vdc, 10 position encoder)

Linear 80 TPI/Rotary 80:1 Connection Specifications

Motor Type: MTR-10-10E
Connector type: Dual row IDC
***Mate Part# (male pin):**
Pancon part #057-010-115

Pin #	Name	Pin #	Name
1	Motor+	6	Motor -
2	Encoder+V	7	Limit ground
3	Encoder Ch A	8	No connection
4	Encoder Ch B	9	Reverse limit
5	Ground (case)	10	Forward limit



10 pin motor female connector (front view)

Electrical Specifications:

Supply Voltage Nom. (Volts)	6
Armature Resistance (Ohm)±12%	20.1
Maximum power output (watts)(2)	0.42
Maximum Efficiency (%)(2)	67
No Load Speed (RPM) ±12%(2)	18,400
Friction Torque (@ no load speed) (mNm)	0.03
No Load Current (mA) ±50%(3)	10
Stall Torque (mNm)(2)	0.87
Velocity Constant (RPM/Volt)	3,173
Back EMF Constant (mV/RPM)	0.315
Torque Constant (mNm/A)	3.01
Armature Inductance (mH)	0.060
Speed/Torque Gradient(RPM/mNm)	21,185
Maximum permissible speed (RPM)	13,000
Maximum continuous current (mA)	170
Maximum continuous torque (mNm)	0.48

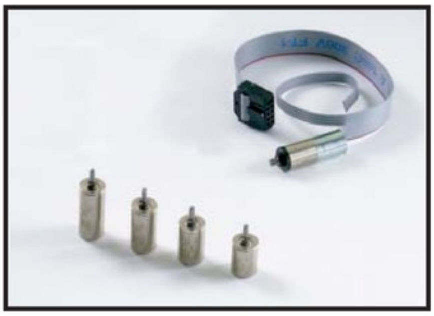
Mechanical Specifications:

Mechanical Time Constant (ms)(2)	13
Armature Inertia (g cm^2)(2)	0.06
Angular Acceleration (x 10^3 rad/sec^2)(2)	145
Rotor Temperature Range	-30ºC to +85ºC
Axial Play	0.2 mm
Shaft Play (measured @ bearing)	
Radial	Less than 0.02mm
Axial	Less than 0.2mm
Maximum Shaft Load	
Radial (@3,000 RPM)	
1.5 mm from bearing (N)	5
Axial @ standstill (N)	5
Weight	6.5 g
Planetary Gearhead recommended	
max continuous input speed	5000 RPM

(1) Ratings are presented independent of each other
(2) Specified at nominal supply voltage
(3) Specified with shaft diameter = 0.8mm at no load
*Mating connectors available through National Aperture, Inc.

Encoder Specifications:

Supply Voltage	5 Vdc Nom.
Maximum Voltage Supply	15 Vdc
Operating Current	5mA Nom. @ 5 Vdc
Signal Phase Shift	90º
Maximum Signal Frequency	7.2 Khz
Temperature Range	-30ºC to +85º C
Output Signal Type	Square wave
Signal Rise Time	Less than 5µs
Phase Relationship	Ch A leads CH B when motor rotation is clockwise as seen from shaft end.
Pulses per Revolution	10 (2 channels)
Quadrature	40 encoder counts
Output signal	CMOS and TTL compatible



The information contained in this data sheet is subject to change without notice. Critical dimensions or specifications should be verified with our technical support staff.

10 Position Encoder Resolution Data Sheet

MTR-10-10E

MM-3M-ST, -F, -FOS, -EX, MM-4M-F		
80 TPI Lead Screw (0.3175 mm/turn)		10 position encoder[1]
GH[2] Ratio	Max Travel Rate[3] (mm/sec)	Resolution (µm/count)
16:1	6.614	0.4961
64:1	1.653	0.1240
256:1	0.413	0.0310
1024:1	0.103	0.0078
MM-3M-ST, -F, -EX, MM-4M-F		
40 TPI Lead Screw (0.635 mm/turn)		10 position encoder[1]
GH[2] Ratio	Max Travel Rate[3] (mm/sec)	Resolution (µm/count)
16:1	13.229	0.9922
64:1	3.307	0.2481
256:1	0.827	0.0620
1024:1	0.207	0.0155

Notes:

1. The 10mm motors used with both linear and rotary stages incorporate dual channel, 10 position, magnetic encoders. The quadrature output is equivalent to 40 encoder counts per motor armature revolution.
2. Gearhead ratio is denoted by GH.
3. Maximum travel rate is calculated with the motor armature turning at a maximum rate of 20,000 RPM.

Linear Travel

Travel rate calculations

Lead screw RPM = motor RPM)/(gearhead ratio)
Distance per minute = (lead screw RPM) x lead; (lead = 0.3175 mm for 80 TPI screw and 0.635 mm for 40 TPI screw)
Distance per second = (distance per minute)/60
Distance in inches = (distance (mm))/(25.4)

Example calculation: with motor RPM = 20,000; GH ratio = 16:1; lead = 0.3175 mm
Distance per second = [(20000 RPM)/(16)] x (0.3175 mm) x (min/60 sec)] = 6.6145 mm/sec

Encoder resolution calculations

Encoder counts per lead screw revolution = (encoder counts per motor revolution) x (gearhead ratio)
Distance per encoder count = lead/(encoder counts per lead screw revolution)

Example calculation: with motor GH ratio = 16:1; lead = 0.3175 mm; 40 encoder counts per motor revolution
Distance per encoder count = (0.3175 mm)/(40 x 16) = 0.4961 µm/count

10 Position Encoder Resolution Data Sheet (cont.)

MTR-10-10E

MM-3M-R			
80:1 Worm Drive Ratio			10 position encoder[1]
GH[2] Ratio	Final Output	Max Travel Rate[3] (rad/sec)	Resolution (µrad/count)
16:1	1,280:1	1.636	122.7185
64:1	5,120:1	0.409	30.6796
256:1	20,480:1	0.102	7.6699
1024:1	81,920:1	0.025	1.9175

Notes:

1. The 10mm motors used with both linear and rotary stages incorporate dual channel, 10 position, magnetic encoders. The quadrature output is equivalent to 40 encoder counts per motor armature revolution.
2. Gearhead ratio is denoted by GH.
3. Maximum travel rate is calculated with the motor armature turning at a maximum rate of 20,000 RPM.

Rotary Travel

Travel rate calculations

Rotor travel rate (RPM) = (motor RPM)/[gearhead ratio) x (worm drive ratio)]
Rotor travel rate (rad/sec) = [rotor travel rate (RPM)] x (min/60 sec) x (6.283185 rad/revolution)

Example calculation: with motor RPM = 20,000; GH ratio = 16:1; lead = 0.3175 mm
Rotor travel rate (rad/sec) = (20000 RPM)/(16 x 80) x (min/60 sec) x (6.283185 rad/revolution) = 1.63624 rad/sec

Encoder resolution calculations

Encoder counts per lead screw revolution = [(encoder counts per motor revolution)] x (gearhead ratio) x (worm drive ratio)
Angular resolution = (6.283185 rad/revolution)/ (encoder counts per lead screw revolution)

Example calculation: with motor GH ratio = 16:1; lead = 0.3175 mm; 40 encoder counts per motor revolution
Angular resolution = (6.283185 rad per lead screw revolution)/[(40 counts per motor revolution) x (16 motor revolutions per gearhead revolution) x (80 gearhead revolutions per lead screw revolution)]
= 122.718 µrad/count

Conversion:

1 inch (in)	= 25.4 mm
1 inch	= 25,400 µm
1 millimeter (mm)	= 39.37E-3 inch
1 micron (µm)	= 39.37E-6 inch
1 deg	= 0.01745329252 rad

The information contained in this data sheet is subject to change without notice. Critical dimensions or specifications should be verified with our technical support staff.

KP-M1A

B/W CCD Camera
Operation Guide

Aug 2001

Hitachi Kokusai Electric Inc.

Table of contents

1. General

The KP-M1A is a black and white camera using the latest high grade CCD having a 2/3-inch image size.

The KP-M1A features high sensitivity, high resolution and high performance.

The KP-M1A is provided with various functions including the multiple step electronic shutter function, field-on-demand function, integration mode switching function, HD/VD pulse switching function and non-interlaced scanning function.

The KP-M1A is most suitable for use with a microscope or an image processing system.

The KP-M1A is compatible with the KP-M2/M3.

KP-M1AN: EIA, KP-M1AP: CCIR

2. Major features

(1)Small size and lightweight

Though the KP-M1A is small in size and light in weight, it offers high performance.

A video signal is obtained only by supplying 12V DC.

(2)High resolution

The latest high grade CCD is used. The horizontal resolution is 570TVL (560TVL CCIR), and the vertical resolution is 485TVL(575TVL CCIR).

(3)Multiple-step electronic shutter function

The multiple-step electronic shutter function is provided.

The eight-step shutter speeds can be selected from 1/100(1/120 for CCIR) to 1/10000.

(4)Internal or external synchronization, and interlaced or non-interlaced operation

The synchronization system and the scanning system are automatically switched by the kind of an external sync signal.

(5)Field-on-demand function

The image captured at an optional timing by an external trigger signal can be displayed instantly.

The capture time can be adjusted by an external trigger signal and the shutter.

(6)Frame or field integration mode

As the integration mode can be switched, the image according to applications can be obtained by combining with the scanning system and the electronic shutter function.

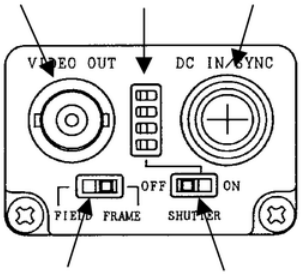
3. Specifications

(1)Imaging device	interline CCD
No. of total pixels	EIA :811(H) ×508(V) CCIR:795(H)×596(V)
Pixel size	EIA :11.64(H)×13.5(V)μm CCIR:11.6(H)×11.2(V)μm
No. of effective pixels	EIA :768(H)×494(V) CCIR:752(H)×582(V)
(2)Sensing area	EIA :8.94 ×6.67mm(2/3 inch size) CCIR:8.72 ×6.52mm(2/3 inch size)
(3)TV format	EIA/CCIR
(4)Lens mount	C-mount
(5)Flange focal distance	17.526mm(Not adjustable)
(6)Hor. Scanning freq.	EIA:15.734kHz CCIR:15.625kHz
(7)Vert. Scanning freq.	EIA:59.94Hz CCIR:50Hz
(8)Sync system	Internal/external (automatic switching)
(9)Int. sync scanning system	2:1 interlaced No. of Hor. lines:525(625 CCIR) fv=2fH/525(625 CCIR)
(10)Ext. sync input	HD/VD : 2 to 6Vp-p,negative Input impedance : 1kΩ Frequency deviation : ±1%
(11)Video output	1.0Vp-p/75Ω, unbalanced Video: 0.7Vp-p Sync: 0.3Vp-p, negative
(12)Resolution	EIA : 570TVL(H)/485TVL(V) CCIR: 560TVL(H)/575TVL(V)
(13)Sensitivity	400Lx, f4, 3200K
(14)Min. illumination	0.3Lx, fl.4, AGC:ON, GAMMA:ON, W/O IR cut filter
(15)Signal-to-noise ratio	56dB
(16)Electronic shutter	1/10000, 1/4000, 1/2000, 1/1000, 1/500, 1/250, 1/125, 1/120(CCIR), 1/100(EIA)sec OFF :Normal exposure Settable from external switch. Set to OFF at factory.

(17)Integration mode	Field or frame (Settable by external switch.) Set to frame integration mode at factory.
(18)Gamma correction	Gamma = 1.0 or correction (Settable by internal switch.) Set to 1.0 at factory.
(19)AGC	Fixed or AGC (Settable by internal switch.) Set to fixed gain at factory.
(20)Field-on-demand	ON or OFF (Settable by internal switch.) Set to OFF at factory.
(21)Restart ・ Reset	ON or OFF (Option) (Settable by chip components) Set to OFF at factory.
(22)Supply voltage	12V DC ± 1V
(23)Power consumption	220mA approx.
(24)Ambient conditions	Operating:-10 to 50℃, 90%RH or less Storage :-20 to 60℃,70%RH or less
(25)Anti-vibration	98 m/s^2 (10 to 60Hz,amplitude:0.98mm constant 60 to 200Hz amplitude, variable) (10 to 200Hz, sweep:1 min, XYZ,duration:30mm.)
(26)Anti-vibration	686 m/s^2 (Once each, top/bottom/left side/right side)
(27)Dimensions	44(W)×29(H)×72(D)mm
(28)Mass	120g approx.

4. Name of each section

(1) (2) (3)



(4) (5)

(1) VIDEO OUT (BNC) connector

A composite video signal (VS) is fed from this connector. Connect a 75-ohm coaxial cable between the connector and a video monitor or other video equipment.

(2) Shutter speed select switch

Use this switch to set the shutter speed.

(3) DC IN/SYNC connector

This connector is for 12V DC input, a composite video signal (VS) output and an external sync signal input.

(4) FIELD/FRAME integration select switch

Use this switch to select an integration mode. This switch is set to FRAME at factory.

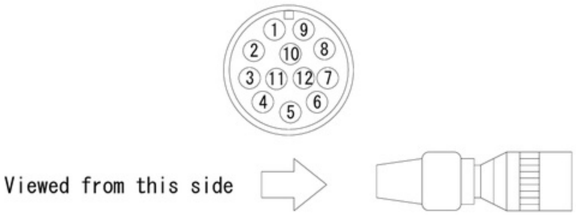
(5) SHUTTER ON/OFF switch

Set the SHUTTER ON/OFF switch to ON to establish the shutter mode.

5. Signal connection to DC IN/SYNC connector.

PIN NO.	Internal sync mode	External sync mode				
		HD/VD	Frame/field on demand			
			ONE trigger	TWO trigger	Fixed shutter	EXT shutter
1	GND	GND	GND	GND	GND	GND
2	+12V	+12V	+12V	+12V	+12V	+12V
3	GND(Vout)	GND(Vout)	GND(Vout)	GND(Vout)	GND(Vout)	GND(Vout)
4	Vout	Vout	Vout	Vout	Vout	Vout
5	-	GND(HD)	-	-	-	GND(HD)
6	-	HD IN	-	-	-	HD IN
7	-	VD IN	Trigger A IN	Trigger A IN	Trigger A IN	VD IN
8	-	-	-	GND(Trig B)	-	GND(Trig B)
9	-	-	-	Trigger B IN	-	Trigger B IN
10	GND	GND	GND	GND	GND	GND
11	+12V	+12V	+12V	+12V	+12V	+12V
12	-	GND(VD)	GND(Trig A)	GND(Trig A)	GND(Trig A)	GND(VD)

12-pin plug HR10A-10P-12S(01) Product code: 23810AX

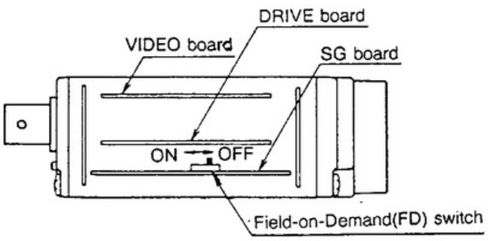


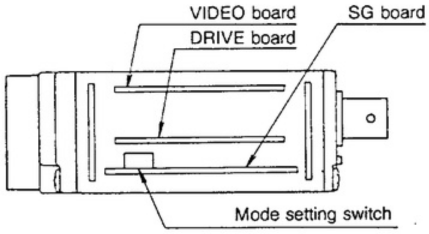
Note:

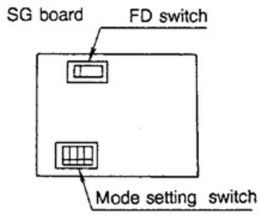
The video signal cannot be fed simultaneously from both the VIDEO OUT connector and the DC IN/SYNC connector. If both the outputs are connected simultaneously, a proper picture cannot be obtained.

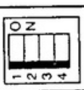
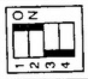
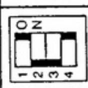
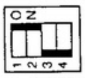
Supply 12V DC in range between 11 and 13V.

6. Arrangement of internal switches

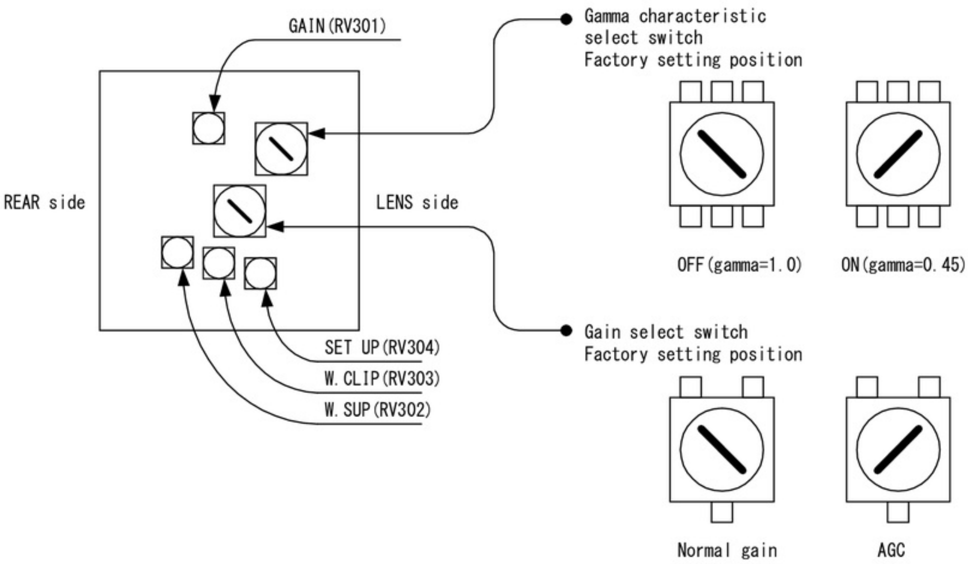






	Normal mode	ONE trigger mode	TWO-trigger mode	Fixed shutter mode	Ext. shutter control mode
Mode setting switch					
FD switch	OFF (Setting at factory)	ON	ON	ON	OFF

Internal controls



Gain control	Adjusts a video level. Set to 0.7Vp-p at 400lx, f4. (Gamma: off, AGC: Normal)
SET UP control	Adjusts a setup level. Set to 75 mVp-p. (Gamma: OFF, AGC: Normal)
W.CLIP control	Clips a video level. Set to 1.0Vp-p. (Max: 1.2Vp-p)
W.SUP control	Prevents white compression of a video signal and extends a dynamic range. Effective when a video level is approximately 120% (Gamma: 1.0). (Approx. 110% for Gamma: ON)

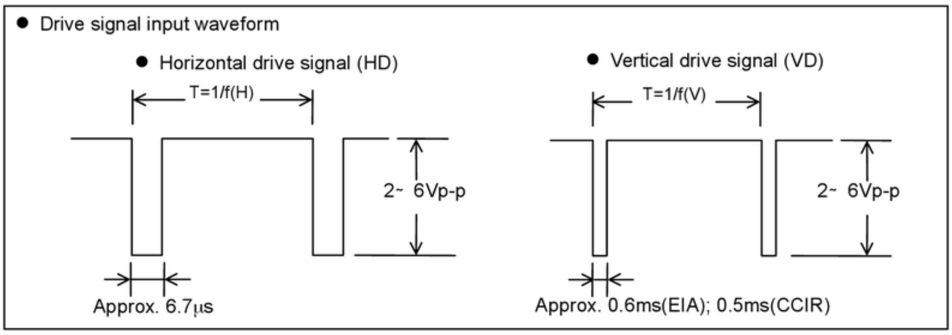
7. External synchronization (2:1 interlaced)

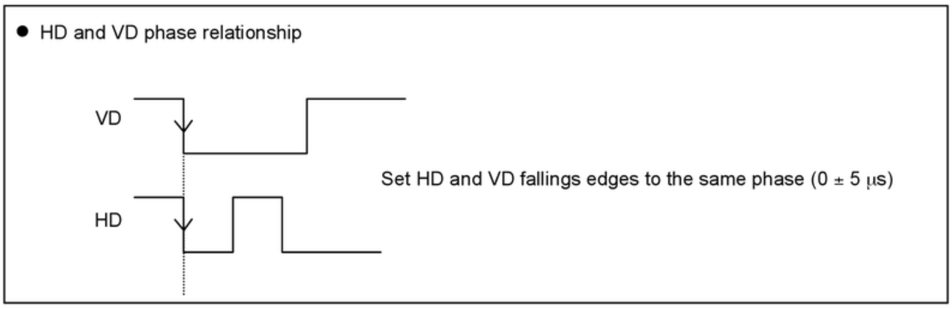
When operating the camera by external drives signals, connect sync drive signals (HD, VD) to the DC IN/SYNC connector, then the mode is automatically switched from the internal sync mode to the external sync mode.

- Horizontal and vertical drive signal inputs
  HD EIA: f(H) = 15.734 kHz ±1 %
  CCIR: f(H) = 15.625 kHz ±1 %
  VD EIA: f(V) = 59.94 Hz (f(V) = f(H) ÷ 262.5)
  CCIR: f(V) = 50 Hz (f(V) = f(H) ÷ 312.5)

- Input level
  HD 2 to 6 Vp-p negative
  VD 2 to 6 Vp-p negative

- Input impedance 1 kΩ





8. Field on demand function

The KP-M1A is provided with the Field-on-Demand function to record a picture obtained at an optional timing by triggering to an image memory, etc. Four modes are switchable by the internal switch.

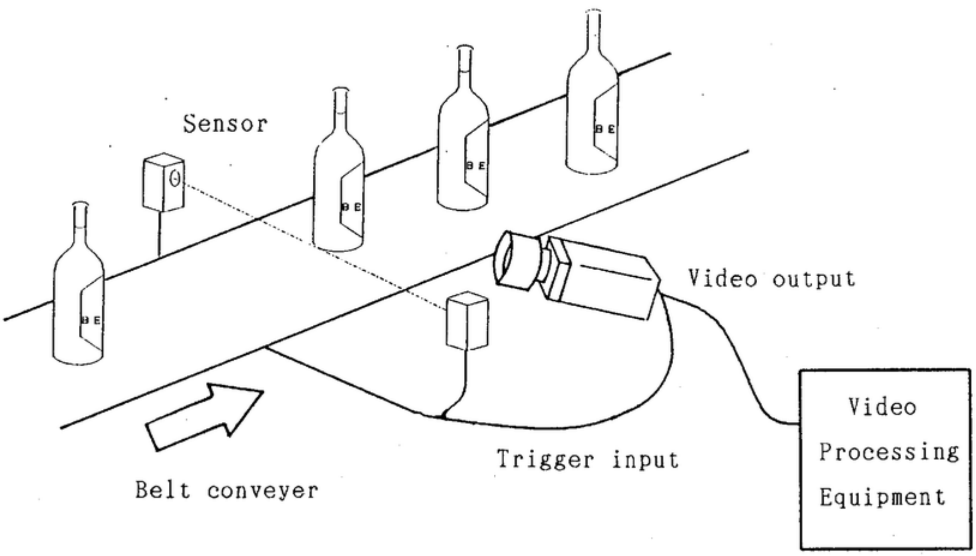
1) ONE trigger mode
2) TWO trigger mode
3) Fixed shutter mode: 1/1600s(EIA) / 1/1000s (CCIR)
4) External shutter control mode

Set the electronic shutter to OFF.

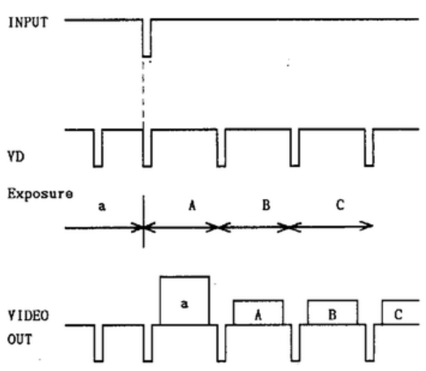
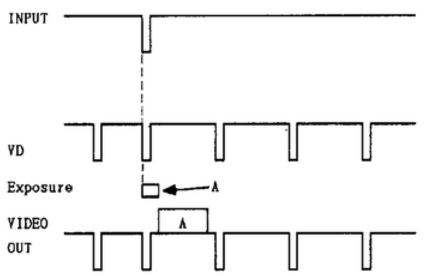
(1) Application

This function is effective to shoot moving objects for image processing.

Example: Defect detection of items on belt conveyer



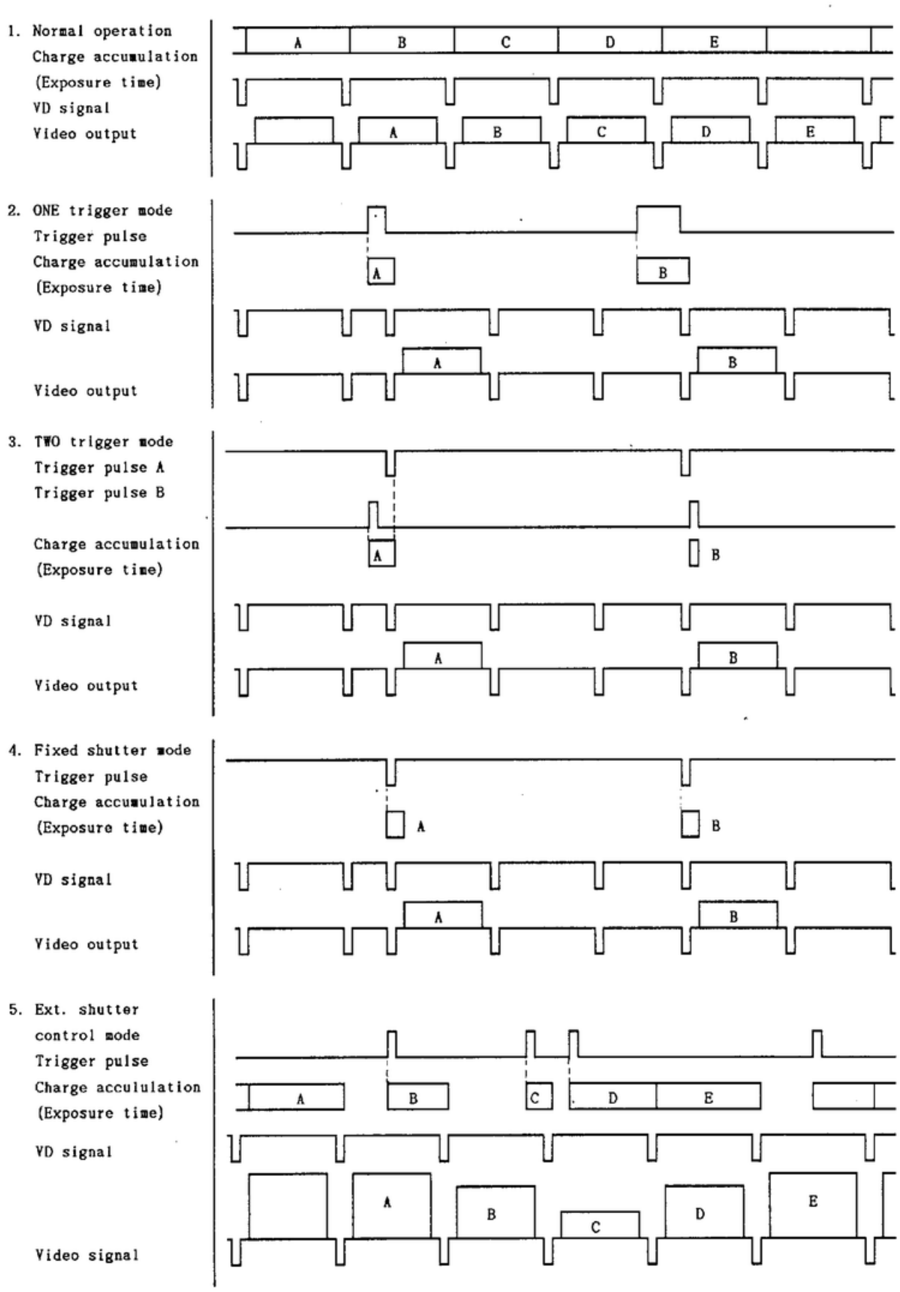
(2) Difference between the restart/reset (R/R) function and the Field-on-Demand (FD) function.

R.R. Restart of sync signal only		A: Image required for the pulse
FD (Fixed shutter) Reset of sync signal and start of shutter		A: Image required for the pulse

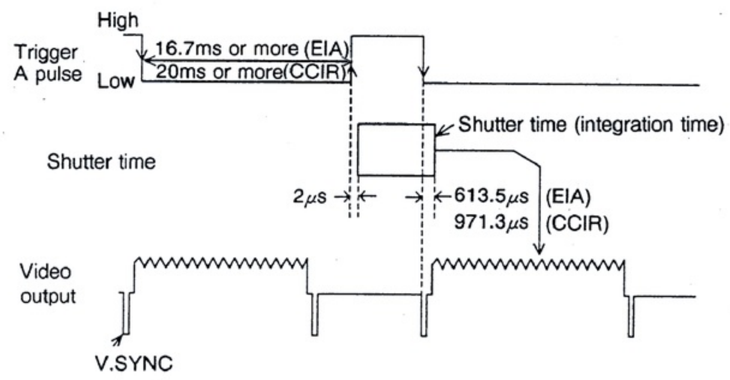
R/R function: The desired image is after one field with respect to the input pulse.

FD function: The desired image is immediately after the input pulse.

(3) Timing diagram



8-1 ONE trigger mode



The shutter is started by the rising edge of the trigger A pulse, and V.SYNC is reset by the falling edge of the trigger A pulse. (After reset, the first field is delivered)

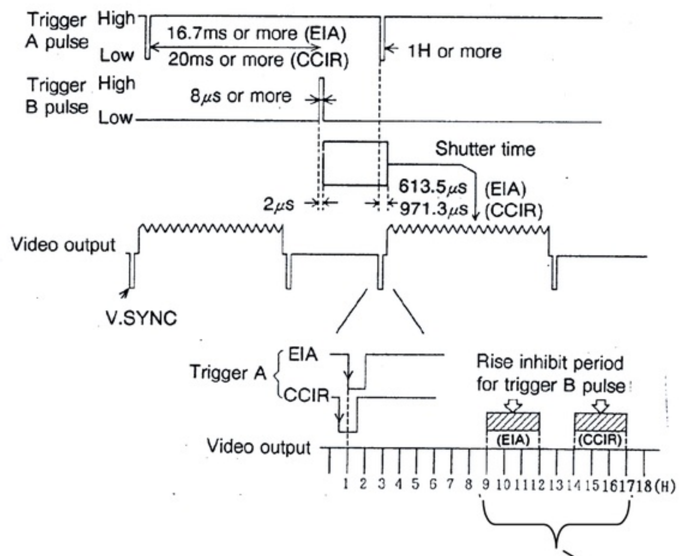
A shutter time is controlled by the duration when the trigger pulse is high.

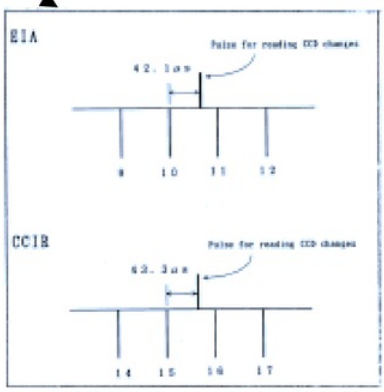
Only one field image is delivered by one trigger pulse, and a sync signal lasts till the next pulse.

Trigger pulse

* 5Vp-p +0.5/-1.0Vp-p
* Low period: EIA: 16.7ms or more
  CCIR: 20ms or more
* High period: 8us or more

8-2 TWO trigger mode





V.SYNC is reset by the falling edge of the trigger A pulse. (After reset, the first field is delivered.)

The shutter is started by the rising edge of the trigger B pulse.

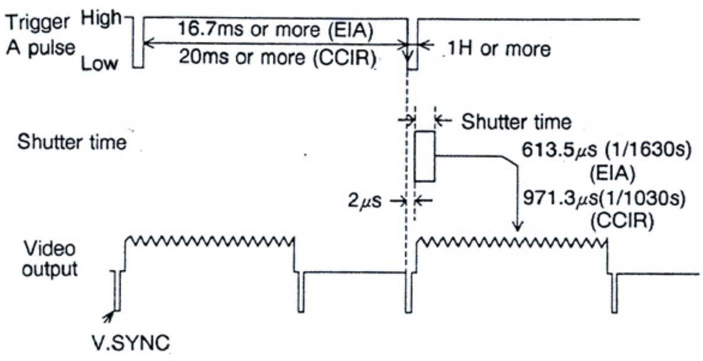
The trigger B pulse is inhibited from the falling edge of the trigger A pulse to 9 to 12H(EIA) /14 to 17H(CCIR), and a correct picture is not obtained.

Trigger pulse

* 5Vp-p +0.5/-1.0Vp-p

* Trigger A:	Low period:	EIA: 63.5us or more
		CCIR: 64us or more
	High period:	EIA: 16.7ms or more
		CCIR: 20ms or more
* Trigger B:	Low period:	Not specified
	High period:	8us or more

8-3 Fixed shutter mode: 1/1600s(EIA), 1/1000s(CCIR)



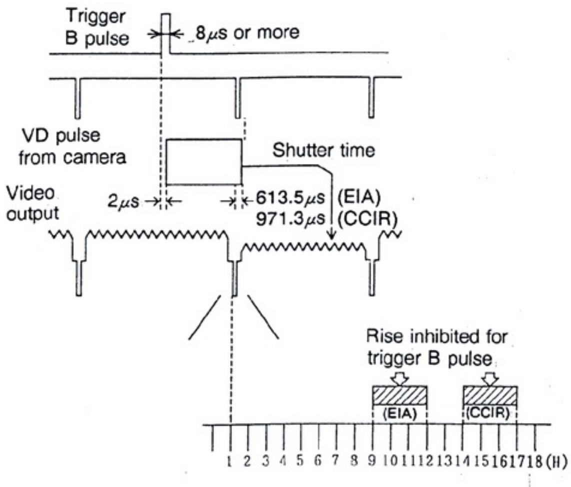
The shutter is started by the falling edge of the trigger A pulse, and at the same time V.SYNC is reset. (After reset, the first field is delivered.)

In this mode, the shutter speed is fixed to 1/1600s (EIA) / 1/1000s(CCIR). Only one field is delivered by one trigger pulse, and a sync signal lasts till the next pulse.

Trigger pulse

* 5Vp-p +0.5/-1.0Vp-p
* Low period: EIA: 63.5us or more
  CCIR: 64us or more
* High period: EIA: 16.7ms or more
  CCIR: 20ms or more

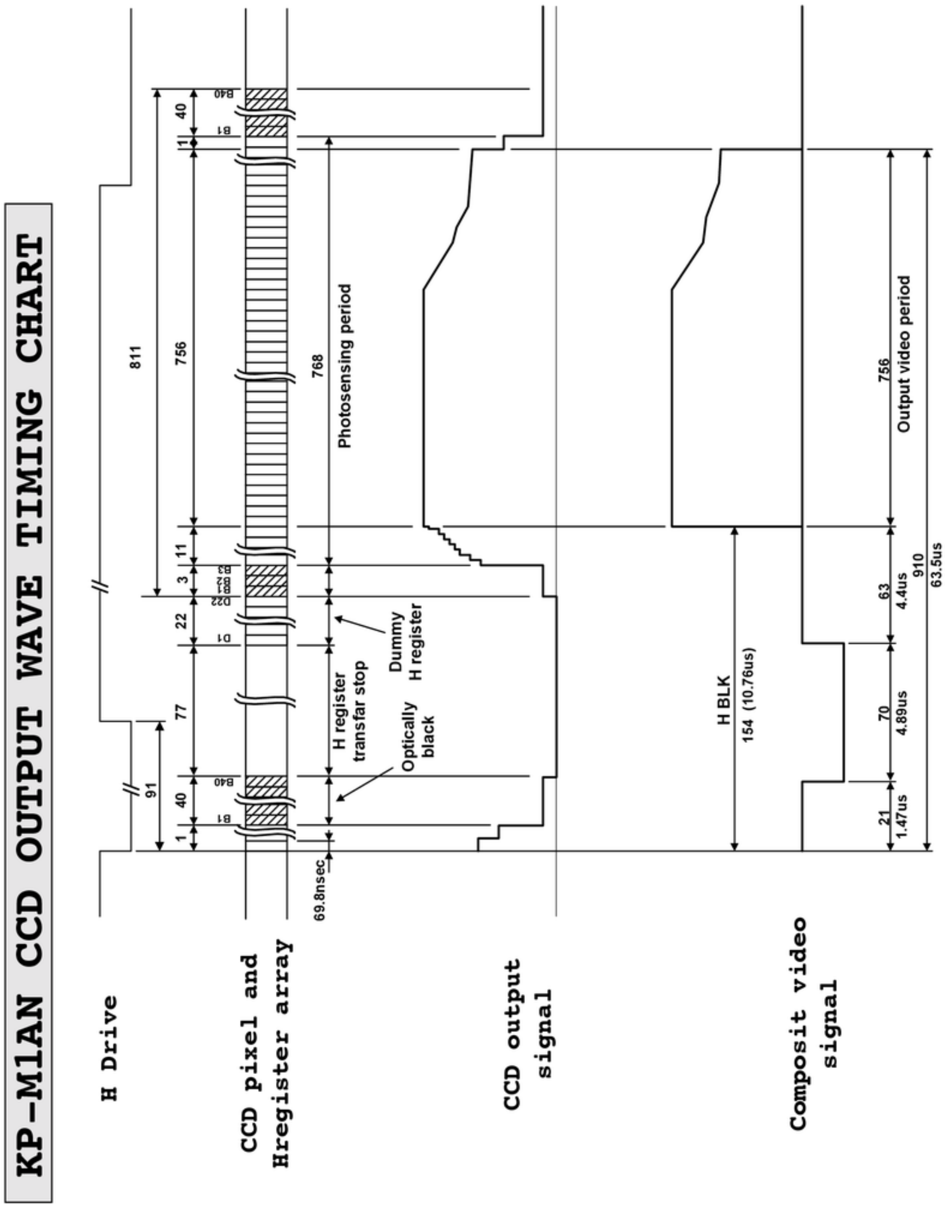
8-4 External shutter control mode



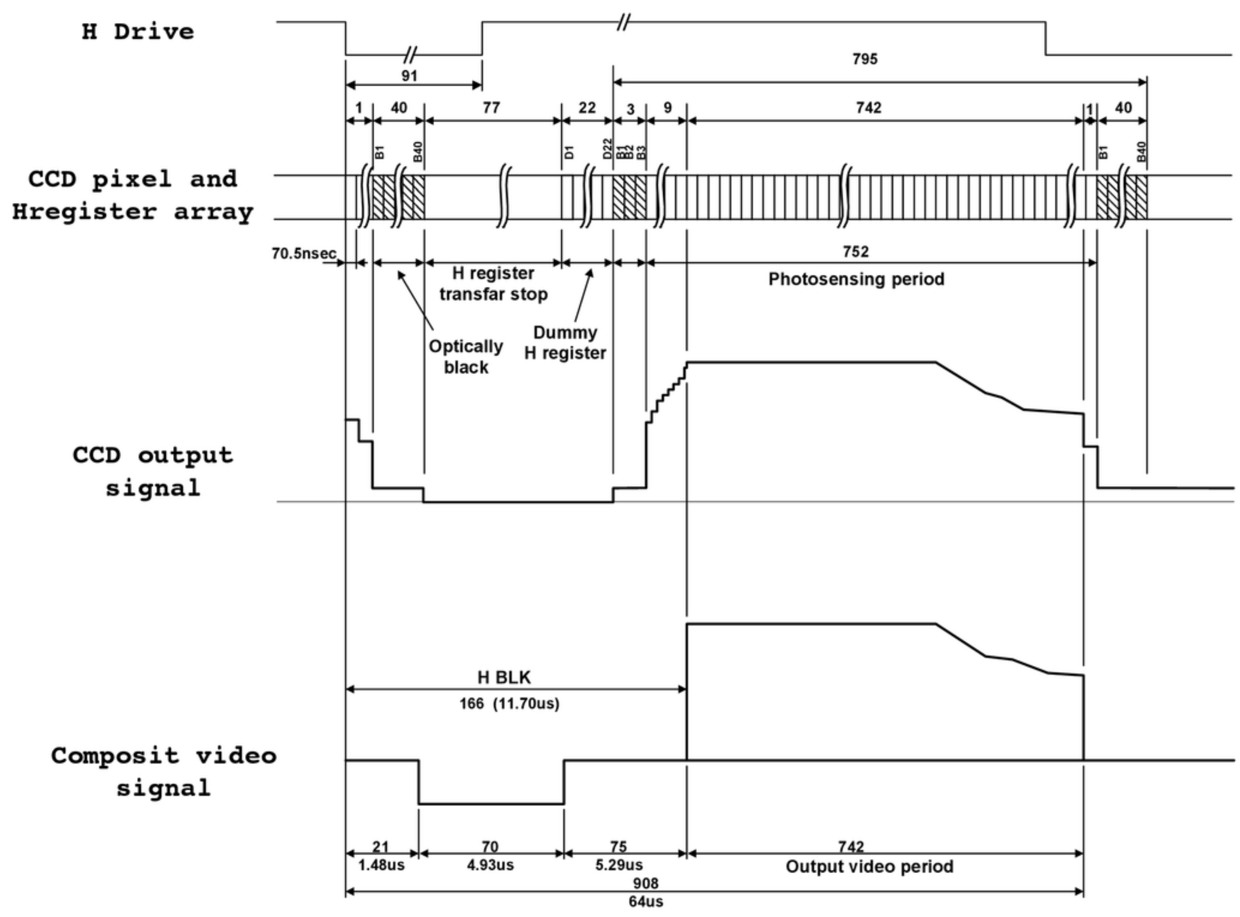
The shutter is started by the rising edge of the trigger B pulse. The shutter is effective only for the next field of the pulse input. When the trigger B pulse is not supplied the normal exposure results.
The V.SYNC pulse of the camera is not reset.

Trigger pulse
* 5Vp-p +0.5/-1.0Vp-p
* Low period: Not specified
* High period: 8us or more

9. Timing chart

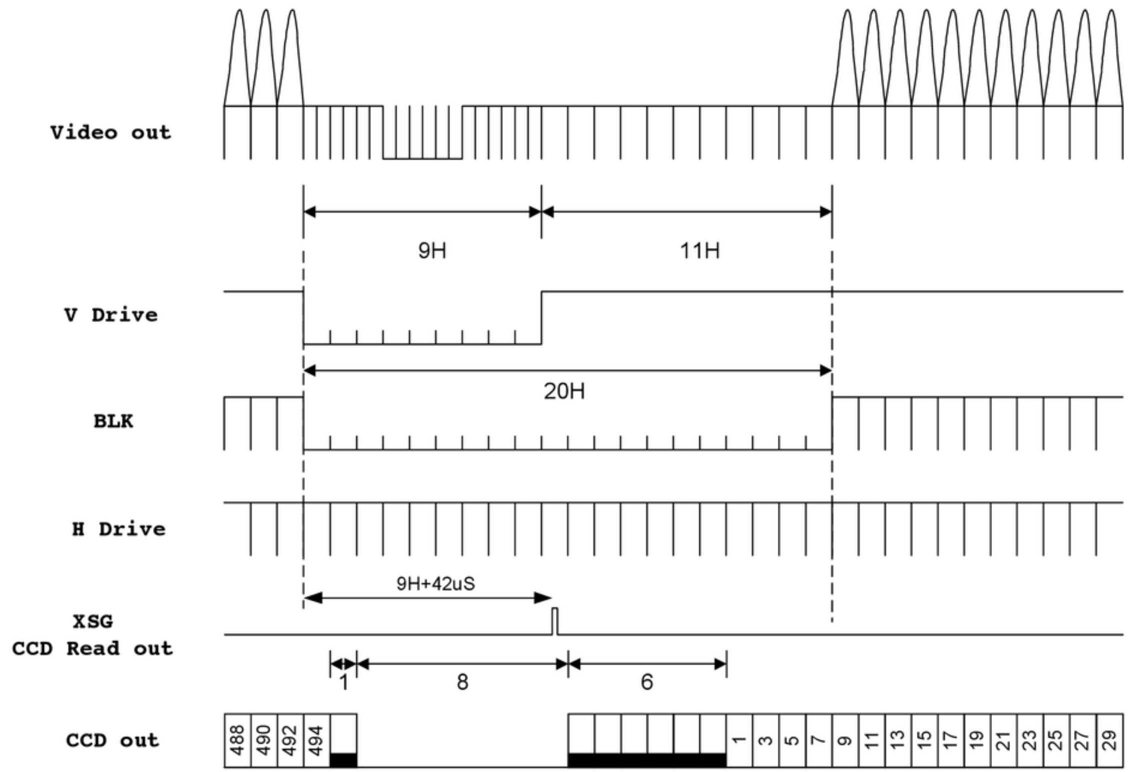


KP-M1AP CCD OUTPUT WAVE TIMING CHART



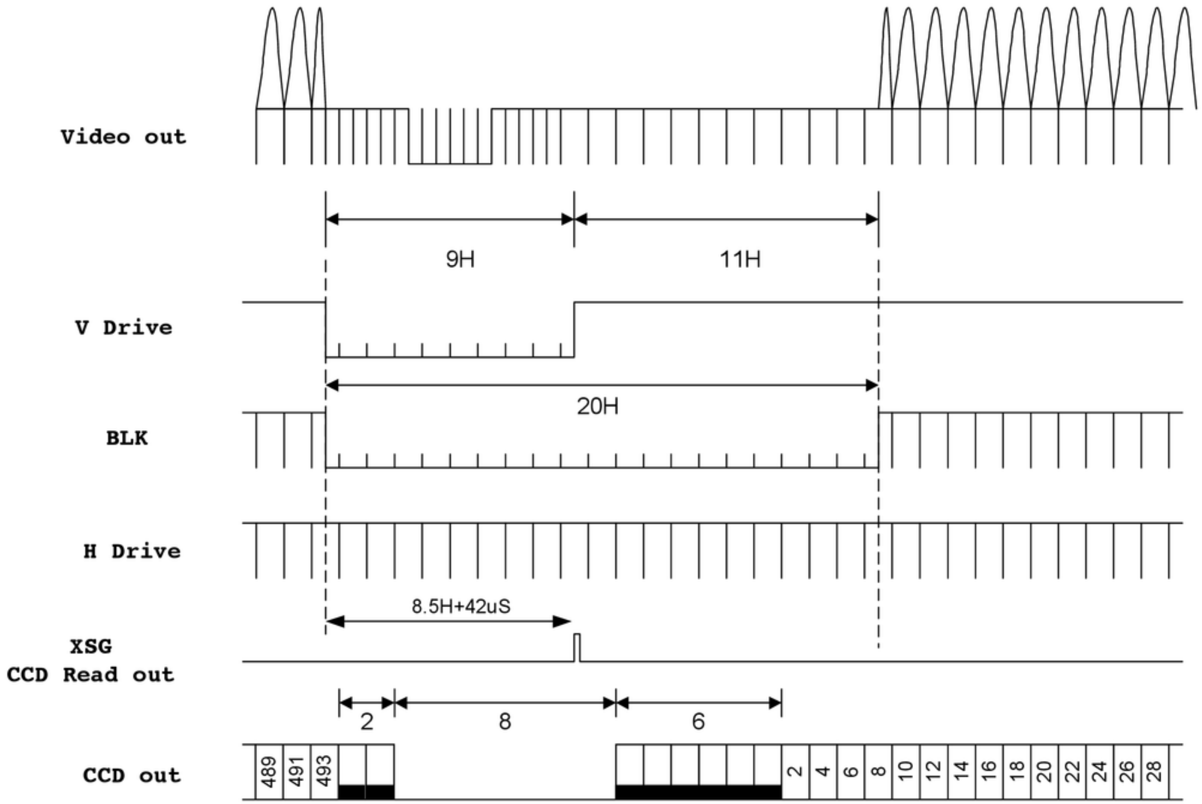
KP-M1AN Timming chart

[ODD Field]



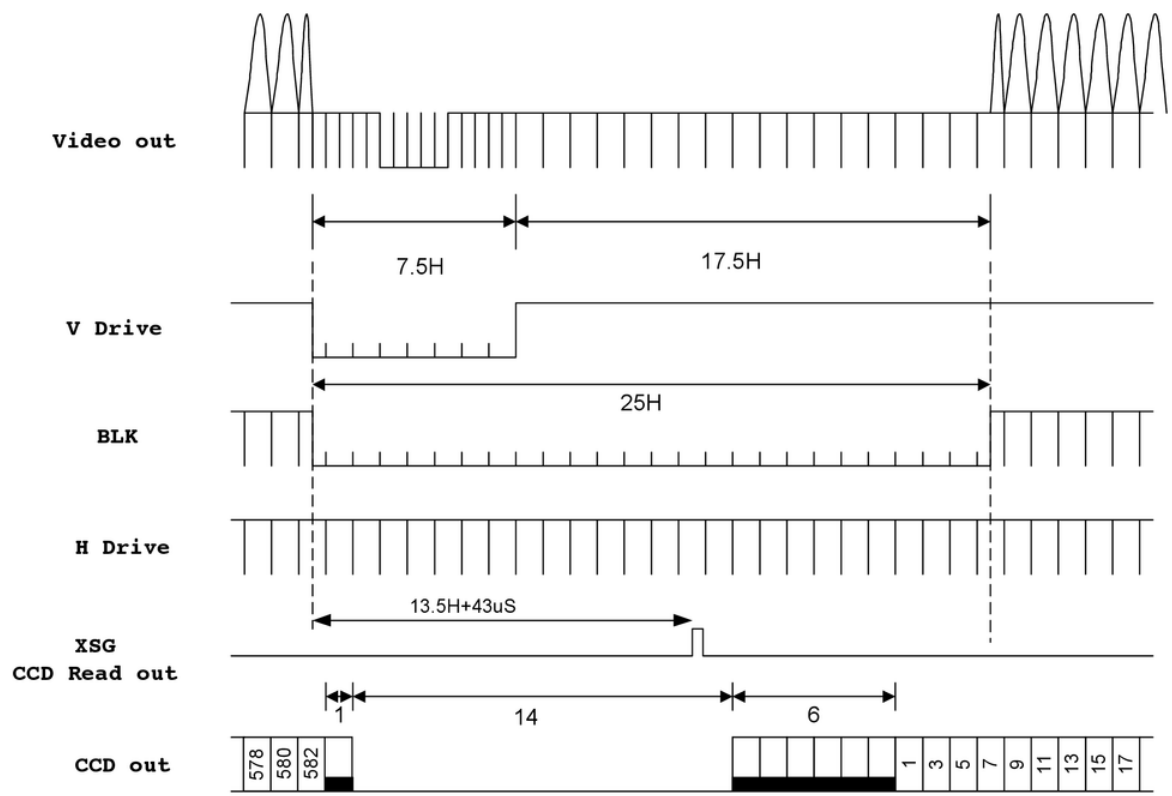
KP-M1AN Timming chart

[EVEN Field]



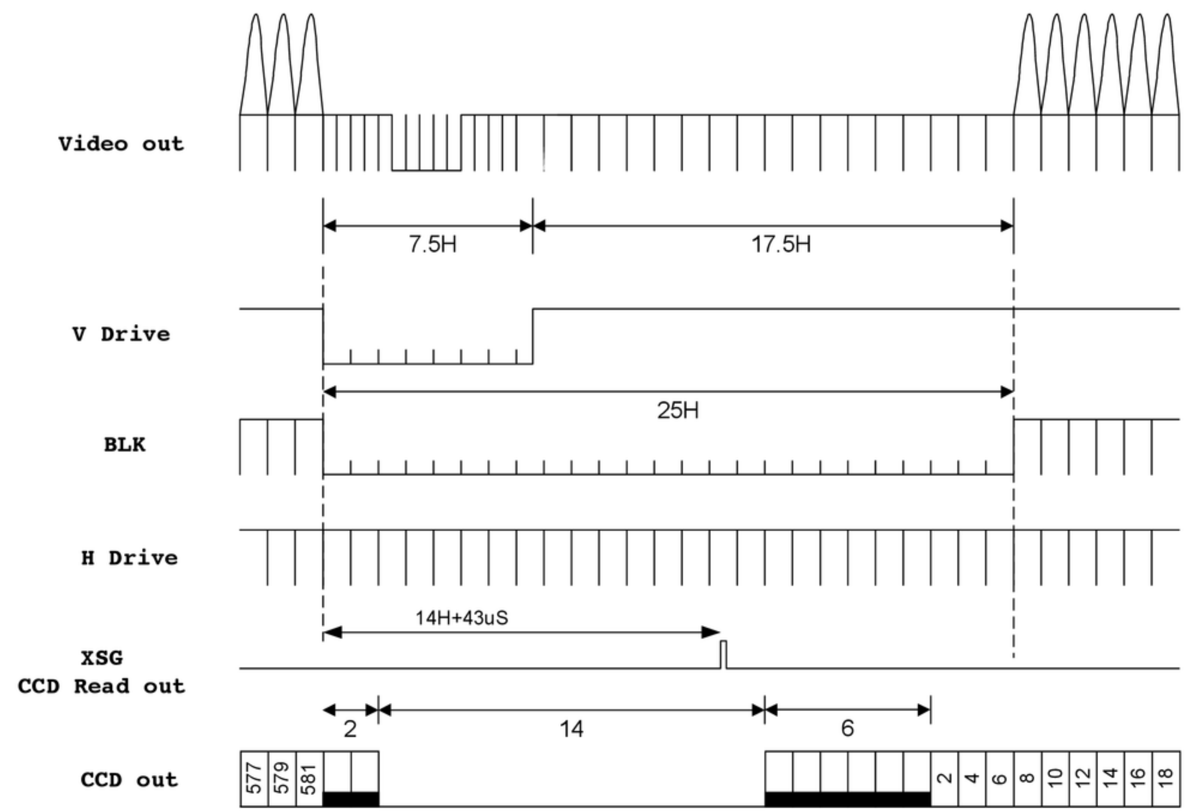
KP-M1AP Timming chart

[ODD Field]



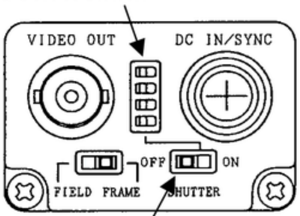
KP-M1AP Timming chart

[EVEN Field]



10. Electronic shutter

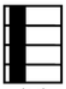
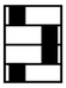
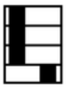
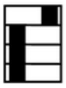
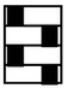
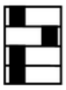
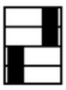
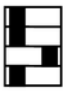
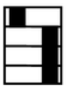
Shutter speed select switch



SHUTTER ON/OFF switch

Set the SHUTTER ON/OFF switch to ON, then set the speed with the shutter speed select switch.

Setting of shutter speed

Setting position	Speed (second)	Relative sensitivity	Setting position	Speed (second)	Relative sensitivity
 (*1)	1/60(EIA) 1/50(CCIR)	1		1/500	1/8
				1/1000	1/16
	1/100(EIA) 1/120(CCIR)	1/1.5		1/2000	1/32
	1/125	1/2		1/4000	1/64
	1/250	1/4		1/10000	1/160

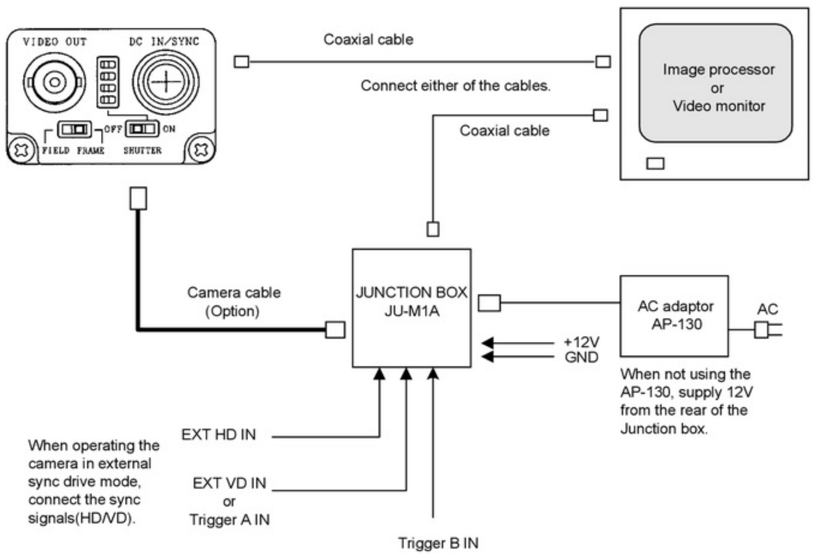
(*1) Or set the shutter ON/OFF switch to OFF.

The higher the shutter speed, the greater the effect. However, since sensitivity lower, adjust the lens iris or increase illumination. And when the shutter is used, the flicker of an object may be emphasized. Use a light which causes no flicker, such as a DC lighting lamp.

11. Connection

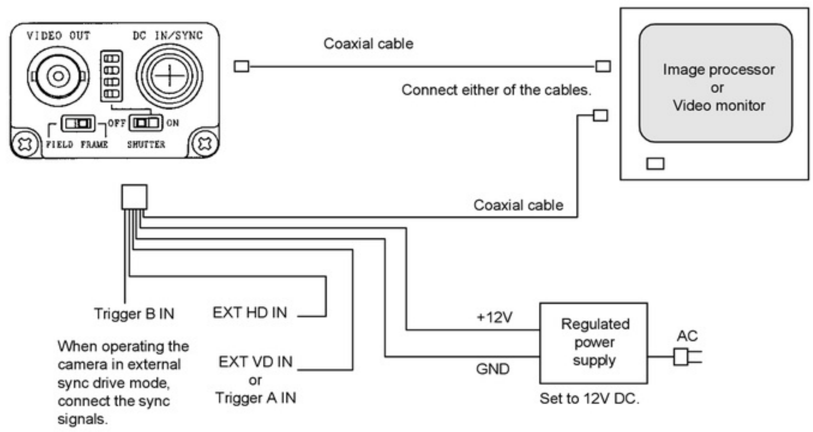
Connection of options

KP-M1A



Basic connection

KP-M1A

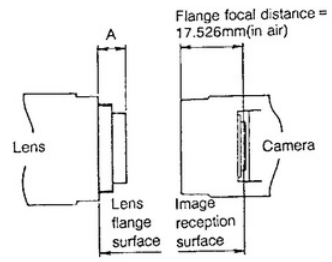


When Connecting more than one monitor, set the 75-ohm termination switch of the last unit only to ON.
When operating the camera in external sync drive mode, input the sync signal(HD/VD).
Available voltage rangeis 11 to 13V.
Before turning on an external power supply unit, be sure to check the polarities of the power supply.

12. Optical system

12-1 Flange focal

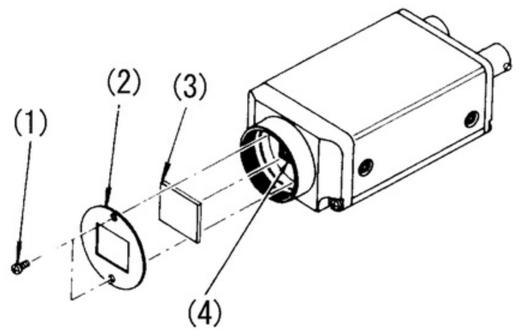
- Image size: 2/3-inch
- The flange focal distance is 17.526mm(in air).
- Flange focal distance cannot be adjusted.



Note:
Select such a lens as the length (A) from the flange surface of the lens to the end of the screw side is 8mm or less.

12-2 Optical filler

This camera is provided with an IR cut filter.



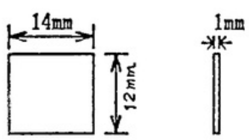
IR cut filter removal

a) Remove two screws (1) shown in Fig., and filter holder (2) will come off.
b) Remove the IR cut filter (3) from filter frame (4).
c) Reinstall and secure filter holder (2) with two screws (1).

Caution
Prior to removing the optical filter, be sure to turn off the power.

IR cut filter IRC650
Dimensions: 14 x 12 x 1.0t
Part code: XMD0006

External view

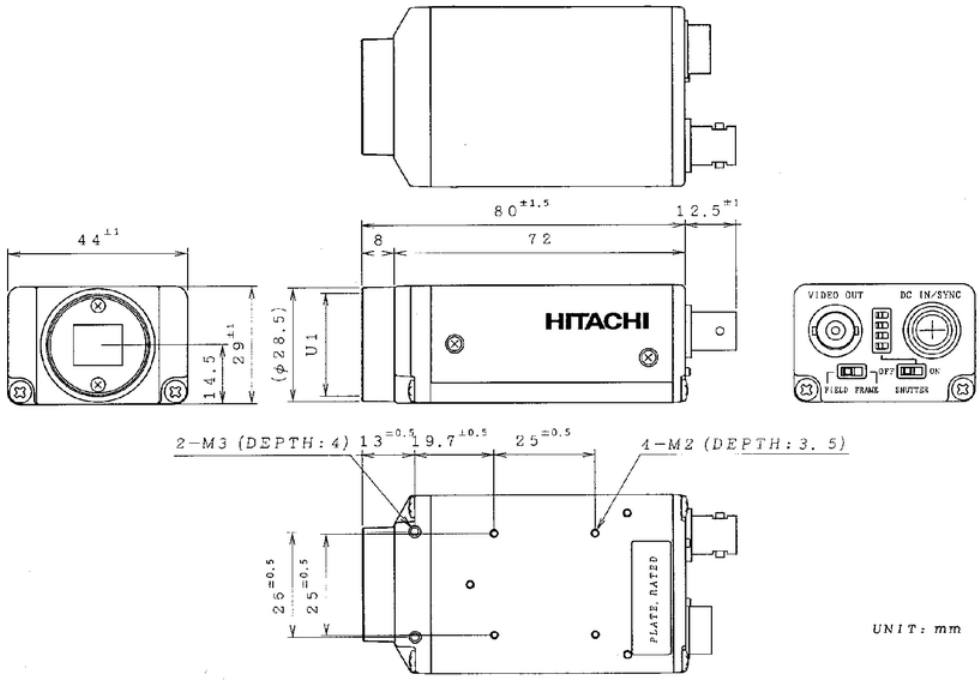


13. Composition

1)Camera(With dummy glass, AR coated)

2)Operation manual

14. External view



15. Optional accessories

1)Tripod adapter, TA-M1

2)12-pin plug, HR10A-10P-12S(01)

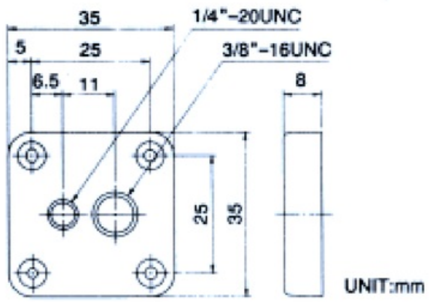
3)Junction box, JU-M1A

4)Dummy glass, ARC1214

5)Camera cables

	Mould Type	Assembly Type	Shield Type
2m	C-201KSM	C-201KS	C-201KSS
5m	C-501KSM	C-501KS	C-501KSS
10m	C-102KSM	C-102KS	C-102KSS

1) Tripod adaptor TA-M1



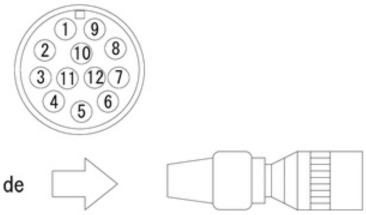
Secure the adaptor to camera mounting holes B, using four supplied screws(M2x5).

Note:

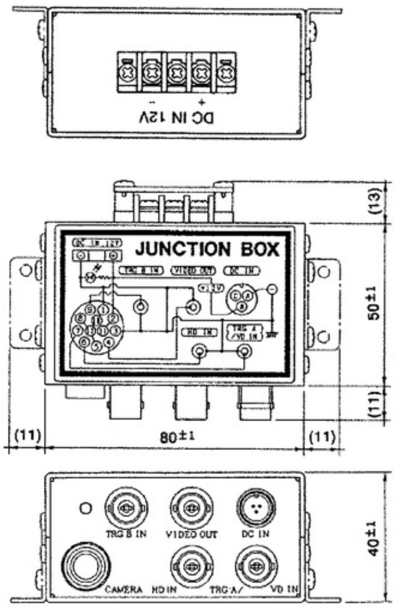
If the screws are too long, they will cause trouble to the camera.

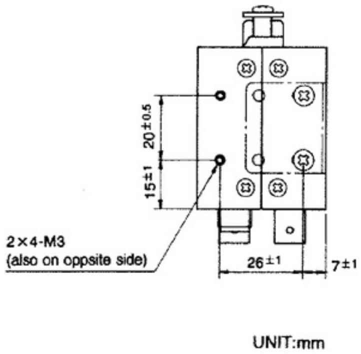
Be sure to check the length before use.

2) 12-pin plug HR10A-10P-12S(01) Product code: 23810AX

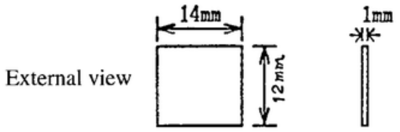


3) Junction box JU-M1A





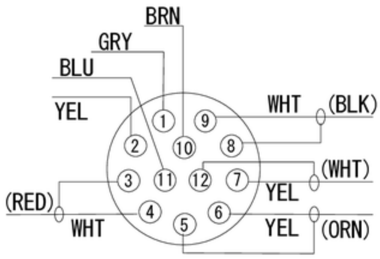
4) Dummy glass ARC1214 Parts code: XMD0009



5) Camera cables

Optional cables are available to connect the camera and the Junction Box JU-M1A.

Length	Type
2m	C-201KSM
5m	C-501KSM
10m	C-102KSM



- Voltage drop due to a cable is about 0.01V per meter.
- The H phase delays by about 5ns per meter.
- When using a cable only to supply power, use the C-201KSM(2m) cable.

Cables other than the above will be prepared upon request.

	Mould type	Assy type	Shielded type
2m	C-201KSM(23861AX)	C-201KS(23856AX)	C-201KSS(23872AX)
5m	C-501KSM(23862AX)	C-501KS(23857AX)	C-501KSS(23873AX)
10m	C-102KSM(23863AX)	C-102KS(23858AX)	C-102KSS(23874AX)

Specify assembly or shielded type at time of order.

( ): Product code

・Attenuation of video signal due to used cable

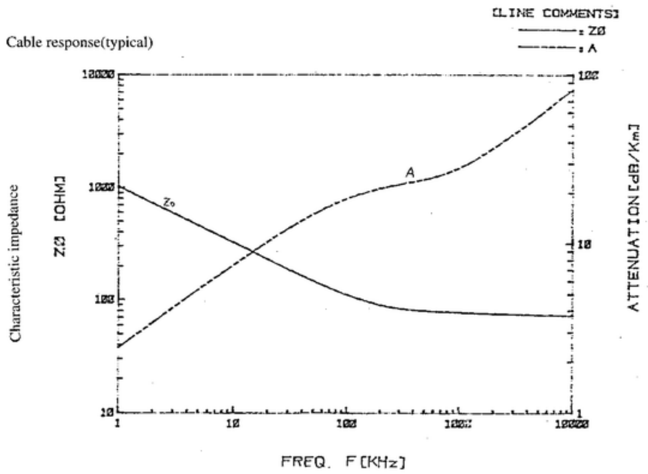
Attenuation due to optional cables C-501KSM and C-102KSM is shown below.

Attenuation is proportionate to the cable length.

Characteristic impedance is kept at constant even at cable length change.

	Cable length	Attenuation at 4MHz 50dB/Km	Attenuation at 7MHz 70dB/Km
Attenuation due to cable length(dB)	1m	0.05	0.07
	2m	0.1	0.14
	5m	0.25	0.35
	10m	0.5	0.7

The video bandwidth obtained by the KP-M1A is up to approximately 7MHz.



16. Notes to users

◆ Power supply

- Connect a 12V DC voltage (11 to 13V) from an external regulated DC power supply.
- Use a stable power supply without ripple and noise.
- Prior to turning on the power switch , check that the polarities of the power cable are correct , referring to the connection diagram

◆ To protect CCD (sensor)

- Do not touch the glass surface of the CCD sensor to avoid deterioration in picture quality due to dirt and scratches.
- If the glass surface of the sensor should become dusty or dirty , remove dust or dirt carefully with a cotton-tipped applicator. Do not wipe the surface with dry cloth or paper tissue to avoid possible damage to the glass surface by static electricity.

◆ Protection of camera

- Do not use or store the camera under direct sunlight , at a place exposed to rain or snow , or at a place where flammable or corrosive gas is present.
- When housing the camera in a camera case , use the utmost care regarding rise of internal temperature.
- When casing the camera , the temperature normally rises by 10 to 20℃ , compared with the outside air temperature. The camera operates in the temperature range from -5 to 45℃. If the camera is used or left in high temperature environment for hours , the life of the camera may be shortened.
- Do not drop the camera. Do not apply strong shock or vibration to the camera.
- Before connecting or disconnecting a connector , turn off the camera and be sure to hold connector body to connect or disconnect the connector.

◆ Camera arrangement

- Mutual interference noise can occur if multiple cameras are arranged in close proximity. Separate the cameras to the extent possible.
- When camera units are installed directly into other equipment , external noise can prevent a normal picture. In such cases , shield the camera units.
- The camera can be damaged by static electricity. Use ample care when installing and arranging.

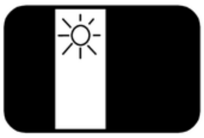
◆ Auto electric shutter

- In regions using 50 Hz power line frequency , flicker can appear on the monitor screen from light sources such as fluorescent or mercury. In such cases , release the auto electronic shutter.

◆ Phenomena inherent to CCD imaging device

◆ Following are phenomena inherent to a CCD imaging device , and not defects.

- Smear and blooming
- When strong light (lamp , fluorescent lamp , reflected light , etc.) is shot , pale bands are displayed vertically above and below the light.
- In this case , change the angle of the camera so that such strong light does not enter the camera through the lens.

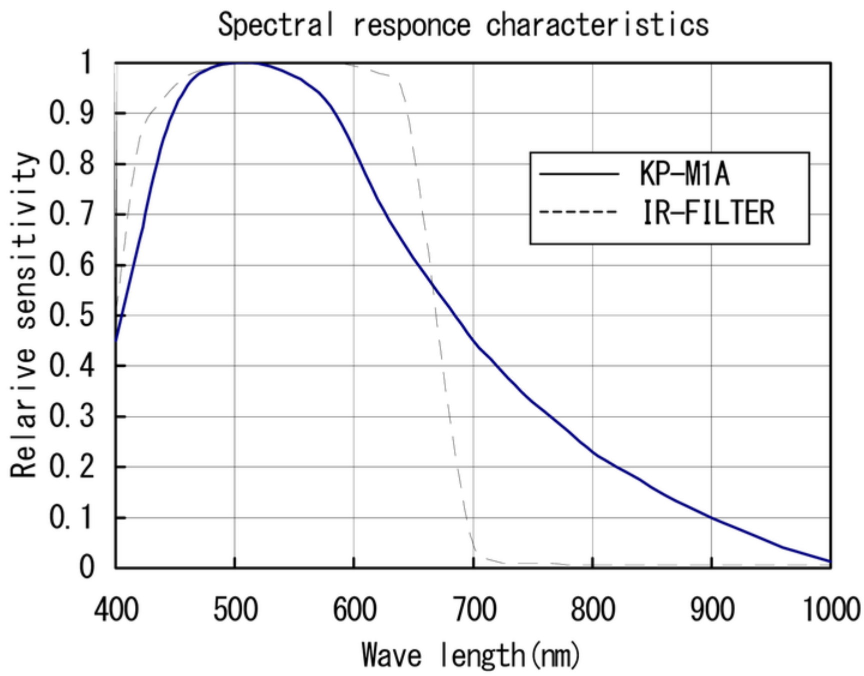


- Fixed pattern noise
- When the camera is operated in a high temperature , fixed pattern noise may appear on the entire screen.
- The higher the sensitivity of camera , the more this fixed pattern noise appears.
-
- Moire
- When fine patterns are shot , moire may be displayed.

◆

- The CE mark is required when exporting to Europe. Obtain the necessary authorization for the customer's system. Enclose the camera in a shielded case and use shielded cable.

Spectral sensitivity (typical example)



Caution

The specifications of this equipment are subject to change without notice for improvement.
Prior to placing your order , be sure to confirm that these specifications are the latest ones.
Hitachi Kokusai Electronic guarantee that the equipment shipped from our factory conforms to the Hitachi Kokusai Electronic 's standard warranty conditions and perform quality control within the range necessary to perform the warranty.

Warranty and After-sales Service

(1) The guarantee period is one year after the date of purchase. However , the defects due to erroneous use or intentional act are excluded.

(2) As the defect after expiration of the guarantee period , Hitachi Kokusai Electronic will repair the equipment if the intended function is restored by the repair work , and the cost is changed to a customer.

(3) Hitachi Kokusai Electronic is not liable for the losses caused when the equipment is used in a system used for business trades , production process , medical fields , crime prevention applications , etc.

(4) The parts used in the equipment have their respective lives. The lives of such parts will be shortened under the environments of high temperature or high humidity. When the stable operation is required for a long time , it is recommended to perform periodical maintenance and inspection every year or every two years.

Arduino code tried

https://www.pjrc.com/teensy/td_libs_Encoder.html

Example Program

This examples program is available from the menu: **File > Examples > Encoder > TwoKnobs**.

```
/* Encoder Library - TwoKnobs Example
 * http://www.pjrc.com/teensy/td_libs_Encoder.html
 *
 * This example code is in the public domain.
 */

#include <Encoder.h>

// Change these pin numbers to the pins connected to your encoder.
//   Best Performance: both pins have interrupt capability
//   Good Performance: only the first pin has interrupt capability
//   Low Performance:  neither pin has interrupt capability
Encoder knobLeft(5, 6);
Encoder knobRight(7, 8);
//   avoid using pins with LEDs attached

void setup() {
  Serial.begin(9600);
  Serial.println("TwoKnobs Encoder Test:");
}

long positionLeft  = -999;
long positionRight = -999;

void loop() {
  long newLeft, newRight;
  newLeft = knobLeft.read();
  newRight = knobRight.read();
  if (newLeft != positionLeft || newRight != positionRight) {
    Serial.print("Left = ");
    Serial.print(newLeft);
    Serial.print(", Right = ");
    Serial.print(newRight);
    Serial.println();
    positionLeft = newLeft;
    positionRight = newRight;
  }
  // if a character is sent from the serial monitor,
  // reset both back to zero.
  if (Serial.available()) {
    Serial.read();
    Serial.println("Reset both knobs to zero");
    knobLeft.write(0);
    knobRight.write(0);
  }
}
```

http://www.hessmer.org/blog/2011/01/30/quadrature-encoder-too-fast-for-arduino-with-solution/

```
#include "WProgram.h"

#include <Servo.h>

#include <digitalWriteFast.h>  // library for high performance reads and writes by jrraines

                               // see http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?num=1267553811/0

                               // and http://code.google.com/p/digitalwritefast/


// It turns out that the regular digitalRead() calls are too slow and bring the arduino down when

// I use them in the interrupt routines while the motor runs at full speed creating more than

// 40000 encoder ticks per second per motor.


// Quadrature encoders

// Left encoder

#define c_LeftEncoderInterrupt 4

#define c_LeftEncoderPinA 19

#define c_LeftEncoderPinB 25

#define LeftEncoderIsReversed

volatile bool _LeftEncoderBSet;

volatile long _LeftEncoderTicks = 0;


// Right encoder

#define c_RightEncoderInterrupt 5

#define c_RightEncoderPinA 18

#define c_RightEncoderPinB 24

volatile bool _RightEncoderBSet;

volatile long _RightEncoderTicks = 0;


Servo _RightServo;  // create servo object to control right motor

Servo _LeftServo;  // create servo object to control left motor


int potpin = 0;  // analog pin used to connect the potentiometer
```

```
int val;    // variable to read the value from the analog pin

void setup()
{
  Serial.begin(115200);

  _RightServo.attach(2);  // attaches the servo on specified pin to the
servo object
  _LeftServo.attach(3);  // attaches the servo on specified pin to the
servo object

  // Quadrature encoders
  // Left encoder
  pinMode(c_LeftEncoderPinA, INPUT);      // sets pin A as input
  digitalWrite(c_LeftEncoderPinA, LOW);  // turn on pullup resistors
  pinMode(c_LeftEncoderPinB, INPUT);      // sets pin B as input
  digitalWrite(c_LeftEncoderPinB, LOW);  // turn on pullup resistors
  attachInterrupt(c_LeftEncoderInterrupt, HandleLeftMotorInterruptA,
RISING);

  // Right encoder
  pinMode(c_RightEncoderPinA, INPUT);      // sets pin A as input
  digitalWrite(c_RightEncoderPinA, LOW);  // turn on pullup resistors
  pinMode(c_RightEncoderPinB, INPUT);      // sets pin B as input
  digitalWrite(c_RightEncoderPinB, LOW);  // turn on pullup resistors
  attachInterrupt(c_RightEncoderInterrupt, HandleRightMotorInterruptA,
RISING);
}

void loop()
{
  val = analogRead(potpin);            // reads the value of the
potentiometer (value between 0 and 1023)
  val = map(val, 0, 1023, 0, 179);     // scale it to use it with the servo
(value between 0 and 180)

  _RightServo.write(val);
```

```
  _LeftServo.write(val);


  Serial.print(_LeftEncoderTicks);
  Serial.print("\t");
  Serial.print(_RightEncoderTicks);
  Serial.print("\n");


  delay(20);
}


// Interrupt service routines for the left motor's quadrature encoder
void HandleLeftMotorInterruptA()
{
  // Test transition; since the interrupt will only fire on 'rising' we
don't need to read pin A
  _LeftEncoderBSet = digitalReadFast(c_LeftEncoderPinB);   // read the
input pin


  // and adjust counter + if A leads B
  #ifdef LeftEncoderIsReversed
    _LeftEncoderTicks -= _LeftEncoderBSet ? -1 : +1;
  #else
    _LeftEncoderTicks += _LeftEncoderBSet ? -1 : +1;
  #endif
}


// Interrupt service routines for the right motor's quadrature encoder
void HandleRightMotorInterruptA()
{
  // Test transition; since the interrupt will only fire on 'rising' we
don't need to read pin A
  _RightEncoderBSet = digitalReadFast(c_RightEncoderPinB);   // read the
input pin


  // and adjust counter + if A leads B
  #ifdef RightEncoderIsReversed
```

```
    _RightEncoderTicks -= _RightEncoderBSet ? -1 : +1;
  #else
    _RightEncoderTicks += _RightEncoderBSet ? -1 : +1;
  #endif
}
```

http://yameb.blogspot.com.au/2012/11/quadrature-encoders-in-arduino-done.html

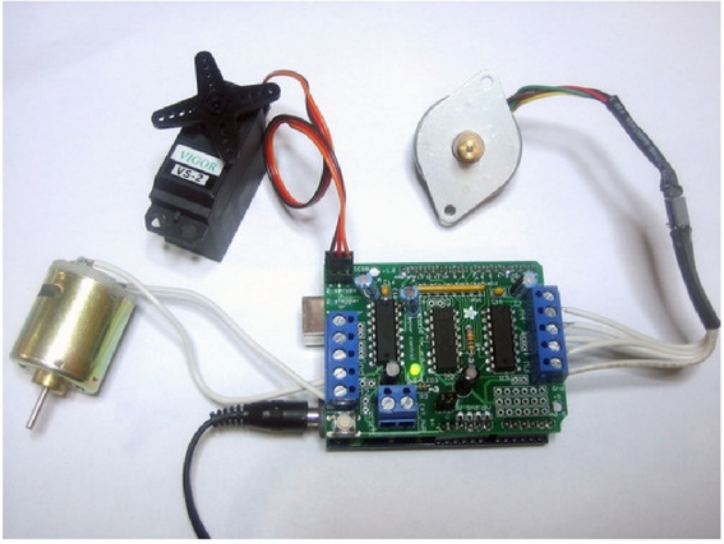
```
*/
#include "Arduino.h"
#include <digitalWriteFast.h>  // library for high performance reads and writes by jrraines
                               // see http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?num=1267553811/0
                               // and http://code.google.com/p/digitalwritefast/

// It turns out that the regular digitalRead() calls are too slow and bring the arduino down when
// I use them in the interrupt routines while the motor runs at full speed.

// Quadrature encoders
// Left encoder
#define c_LeftEncoderInterruptA 0
#define c_LeftEncoderInterruptB 1
#define c_LeftEncoderPinA 2
#define c_LeftEncoderPinB 3
#define LeftEncoderIsReversed

volatile bool _LeftEncoderASet;
volatile bool _LeftEncoderBSet;
volatile bool _LeftEncoderAPrev;
volatile bool _LeftEncoderBPrev;
volatile long _LeftEncoderTicks = 0;
```

```
void setup()
{
  Serial.begin(9600);

  // Quadrature encoders
  // Left encoder
  pinMode(c_LeftEncoderPinA, INPUT);      // sets pin A as input
  digitalWrite(c_LeftEncoderPinA, LOW);  // turn on pullup resistors
  pinMode(c_LeftEncoderPinB, INPUT);      // sets pin B as input
  digitalWrite(c_LeftEncoderPinB, LOW);  // turn on pullup resistors
  attachInterrupt(c_LeftEncoderInterruptA, HandleLeftMotorInterruptA, CHANGE);
  attachInterrupt(c_LeftEncoderInterruptB, HandleLeftMotorInterruptB, CHANGE);
}

void loop()
{
  Serial.print("Encoder Ticks: ");
  Serial.print(_LeftEncoderTicks);
  Serial.print("  Revolutions: ");
  Serial.print(_LeftEncoderTicks/4000.0);//4000 Counts Per Revolution
  Serial.print("\n");
}


// Interrupt service routines for the left motor's quadrature encoder
void HandleLeftMotorInterruptA(){
  _LeftEncoderBSet = digitalReadFast(c_LeftEncoderPinB);
  _LeftEncoderASet = digitalReadFast(c_LeftEncoderPinA);

  _LeftEncoderTicks+=ParseEncoder();
```

```
  _LeftEncoderAPrev = _LeftEncoderASet;
  _LeftEncoderBPrev = _LeftEncoderBSet;
}

// Interrupt service routines for the right motor's quadrature encoder
void HandleLeftMotorInterruptB(){
  // Test transition;
  _LeftEncoderBSet = digitalReadFast(c_LeftEncoderPinB);
  _LeftEncoderASet = digitalReadFast(c_LeftEncoderPinA);

  _LeftEncoderTicks+=ParseEncoder();

  _LeftEncoderAPrev = _LeftEncoderASet;
  _LeftEncoderBPrev = _LeftEncoderBSet;
}

int ParseEncoder(){
  if(_LeftEncoderAPrev && _LeftEncoderBPrev){
    if(!_LeftEncoderASet && _LeftEncoderBSet) return 1;
    if(_LeftEncoderASet && !_LeftEncoderBSet) return -1;
  }else if(!_LeftEncoderAPrev && _LeftEncoderBPrev){
    if(!_LeftEncoderASet && !_LeftEncoderBSet) return 1;
    if(_LeftEncoderASet && _LeftEncoderBSet) return -1;
  }else if(!_LeftEncoderAPrev && !_LeftEncoderBPrev){
    if(_LeftEncoderASet && !_LeftEncoderBSet) return 1;
    if(!_LeftEncoderASet && _LeftEncoderBSet) return -1;
  }else if(_LeftEncoderAPrev && !_LeftEncoderBPrev){
    if(_LeftEncoderASet && _LeftEncoderBSet) return 1;
    if(!_LeftEncoderASet && !_LeftEncoderBSet) return -1
```

Adafruit Motor Shield

Created by lady ada

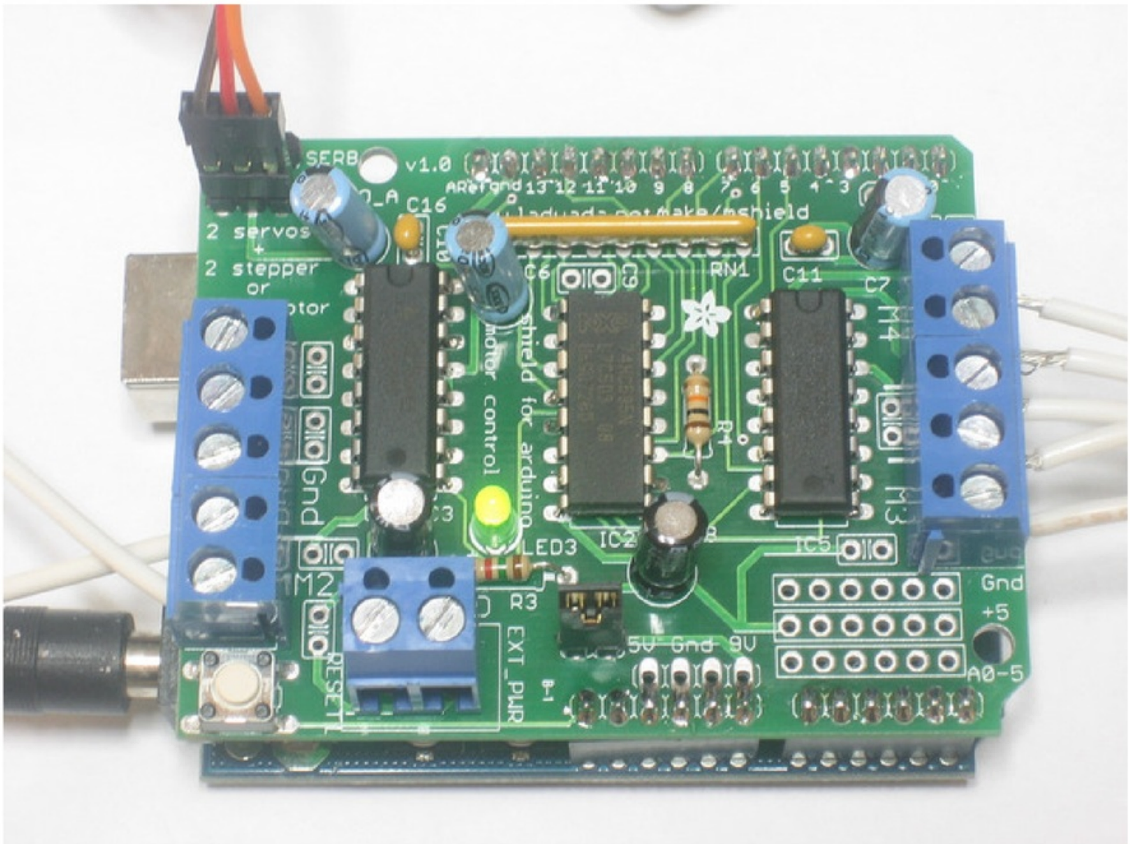


Last updated on 2015-06-07 05:20:10 PM EDT

Guide Contents

Overview

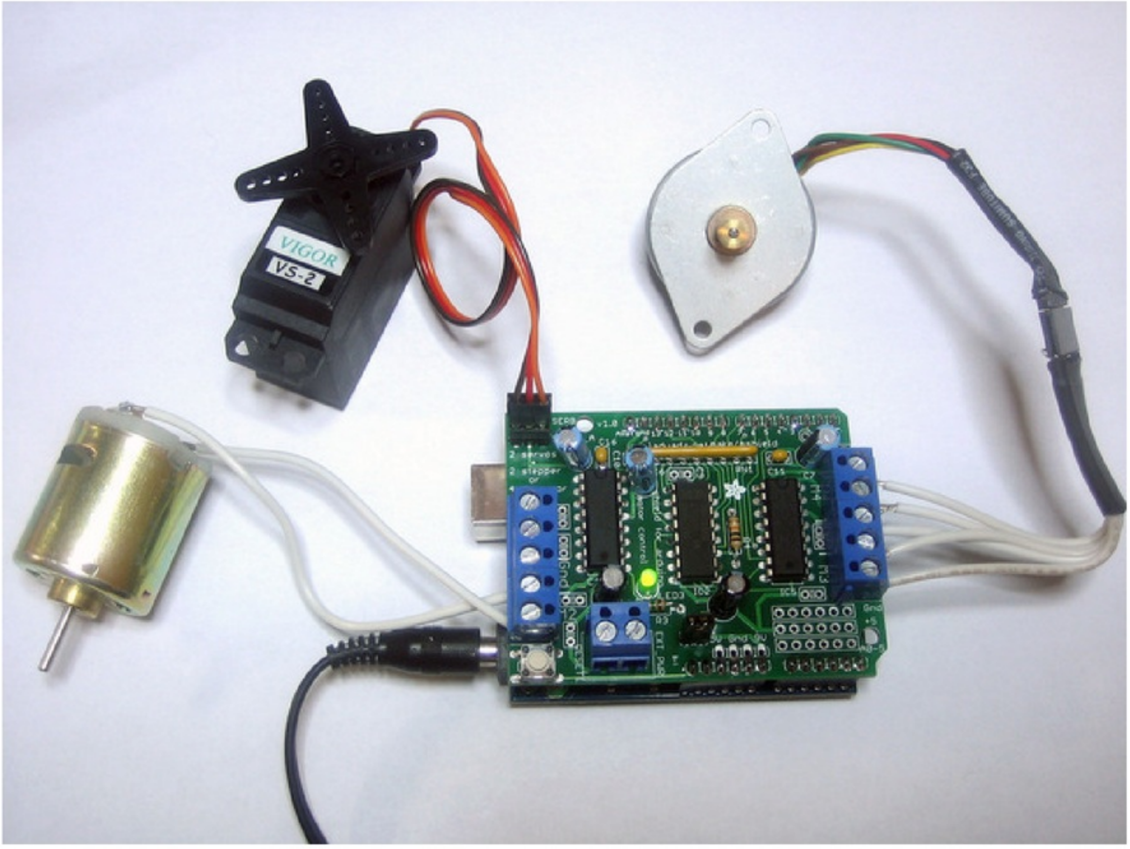
This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!



Arduino is a great starting point for electronics, and with a motor shield it can also be a nice tidy platform for robotics and mechatronics. Here is a design for a full-featured motor shield that will be able to power many simple to medium-complexity projects.

- **2 connections for 5V 'hobby' servos** connected to the Arduino's high-resolution dedicated timer - no jitter!
- **Up to 4 bi-directional DC** motors with individual 8-bit speed selection (so, about 0.5% resolution)
- **Up to 2 stepper motors** (unipolar or bipolar) with single coil, double coil, interleaved or micro-stepping.
- 4 H-Bridges: L293D chipset provides 0.6A per bridge (1.2A peak) with thermal shutdown

protection, 4.5V to 25V
- Pull down resistors keep motors disabled during power-up
- Big terminal block connectors to easily hook up wires (10-22AWG) and power
- Arduino reset button brought up top
- 2-pin terminal block to connect external power, for seperate logic/motor supplies
- Tested compatible with Mega, Diecimila, & Duemilanove
- Full kit available for purchase from the Adafruit shop. (http://adafru.it/81)
- Download the easy-to-use Arduino software libraries and you're ready to go! (http://adafru.it/aON)



FAQ

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!

How many motors can I use with this shield?

You can use 2 DC servos that run on 5V and up to 4 DC motors or 2 stepper motors (or 1 stepper and up to 2 DC motors)

Can I connect more motors?

No, at this time it is not possible to stack the shield or otherwise connect it up easily to control 4 steppers, for example.

HELP! My motor doesnt work! - HELP! My motor doesnt work!...But the servos work FINE!

Is the LED lit? The Stepper and DC motor connections wont do a single thing if the LED is not lit

Don't bother writing up uploading code or wiring up motors if the LED doesn't light up, its not going to work.

What is the LED for?

The LED indicates the DC/Stepper motor power supply is working. If it is not lit, then the DC/Stepper motors will not run. The servo ports are 5V powered and does not use the DC motor supply.

I'm trying to build this robot and it doesn't seem to run on a 9V battery....

Please read the user manual (http://adafru.it/aOz) for information about appropriate power supplies.

Can this shield control small 3V motors?

Not really, its meant for larger, 6V+ motors. It does not work for 3V motors unless you overdrive them at 6V and then they will burn out faster

What is the power connector on the shield for? How do I power my motors?

Please read the user manual (http://adafru.it/aOz) for information about appropriate power supplies.

My Arduino freaks out when the motors are running! Is the shield broken?

Motors take a lot of power, and can cause 'brownouts' that reset the Arduino. For that reason the shield is designed for seperate (split) supplies - one for the electronics and one for the motor. Doing this will prevent brownouts. Please read the user manual (http://adafru.it/aOz) for information about appropriate power supplies.

I have good solid power supplies, but the DC motors seem to 'cut out' or 'skip'.

Try soldering a ceramic or disc 0.1uF capacitor between the motor tabs (on the motor itself!) this will reduce noise that could be feeding back into the circuit (thanks macegr (http://adafru.it/clc)!)

What if I need more than 600mA per motor?

You can subsitute SN754410's (at your risk) or piggyback solder some more L293D drivers on top of the existing ones. (http://adafru.it/aOz)

What pins are not used on the motor shield?

All 6 analog input pins are available. They can also be used as digital pins (pins #14 thru 19)

Digital pin 2, and 13 are not used.

The following pins are in use only if the DC/Stepper noted is in use:
Digital pin 11: DC Motor #1 / Stepper #1 (activation/speed control)
Digital pin 3: DC Motor #2 / Stepper #1 (activation/speed control)
Digital pin 5: DC Motor #3 / Stepper #2 (activation/speed control)
Digital pin 6: DC Motor #4 / Stepper #2 (activation/speed control)

The following pins are in use if any DC/steppers are used
Digital pin 4, 7, 8 and 12 are used to drive the DC/Stepper motors via the 74HC595 serial-to-parallel latch

The following pins are used only if that particular servo is in use:
Digitals pin 9: Servo #1 control
Digital pin 10: Servo #2 control

Which pins are connected to the DC/Stepper motors?

The DC/Stepper motors are NOT connected to the Arduino directly. They are connected to the 74HC595 latch which is spoken to by the Arduino. You CANNOT talk directly to the motors, you MUST use the motor shield library.

Huh? I don't understand...

You can try reading this nice overview written by Michael K (http://adafru.it/aO9)

How can I connect to the unused pins?

The analog pins (analog 0-5 also known as digital pins 14-19) are broken out in the bottom right corner.

Pin 2 has a small breakout since its the only truly unused pin

The remaining pins are not broken out because they could be used by the motor shield. If you are sure that you are not using those pins then you can connect to them by using stacking headers when assembling the kit or soldering onto the top of the header with wires, or using a "Wing shield"

I get the following error trying to run the example code: "error: AFMotor.h: No such file or directory...."

Make sure you have installed the AFMotor library

How do I install the library?

Read our tutorial on libraries (http://adafru.it/aYG)

I have two stepper motors and I want to run them simulaneously but the example code can only control one and then the other?

The stepper motor library step() routine does not have the ability to run both motors at a time. Instead, you will have to 'interleave' the calls. For example, to have both motors step forward 100 times you must write code like this:

```
for (i=0; i<100; i++) {
motor1.step(1, FORWARD, SINGLE);
motor2.step(1, FORWARD, SINGLE);
}
```

If you want more intelligent control, check out the AccelStepper library (in the Downloads section) which has some concurrent stepper motor control examples

What are some 'suggested motors'?

Most people buy motors from surplus shops and no motor will make everyone happy

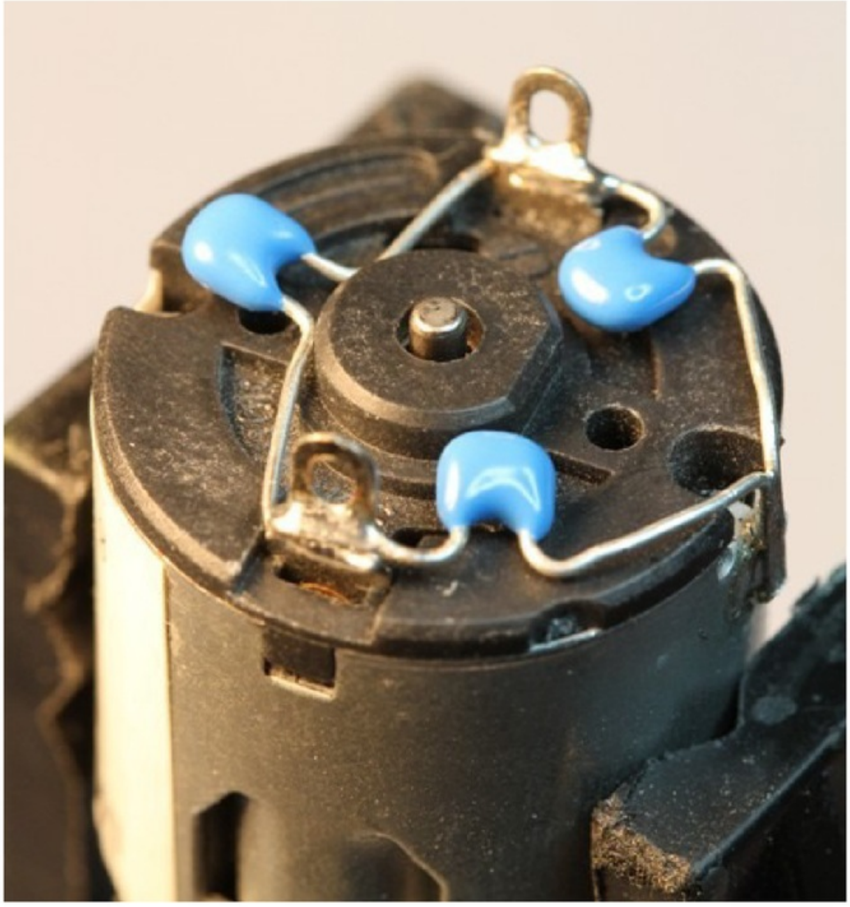
However, since its a popular question, I suggest buying motors from Pololu (DC Servos (http://adafru.it/aOa), DC motors (http://adafru.it/aOb)) or Jameco (all sorts (http://adafru.it/aOc)!) As well as the many surplus webshops (http://adafru.it/aOd).

Is the motor shield compatible with the UNO R3 or Mega R3? What about the extra pins?

The motor shield is compatible with the R3 UNO and MEGA. The R3s have 2 extra pins on each header. These are duplicates of other pins on the header and are not needed by the shield.

I'm using a 4WD robot platform and I can't get anything to work.

The motors used in the 4WD robot platforms from Maker Shed, DF Robotics, Jameco and others have a lot of "brush noise". This feeds back into the Arduino circuitry and causes unstable operation. This problem can be solved by soldering 3 noise suppression capacitors to the motor. 1 between the motor terminals, and one from each terminal to the motor casing.



But my motor already has a capacitor on it and it still doesn't work.
These motors generate a lot of brush noise and usually need the full 3-capacitor treatment for adequate suppression.

Why don't you just design capacitors into the shield?
They would not be effective there. The noise must be suppressed at the source or the motor leads will act like antennae and broadcast it to the rest of the system.

Make It!

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!

Lets go!

This is a vey easy kit to make, just go through each of these steps to build the kit

1. Tools and preparation (http://adafru.it/aOv)
2. Check the parts list (http://adafru.it/aOw)
3. Solder it (http://adafru.it/aOx)

Preparation

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!

Tutorials

- Learn how to solder with tons of tutorials! (http://adafru.it/aOm) (http://adafru.it/aOm)
- Don't forget to learn how to use your multimeter too! (http://adafru.it/aOy)

Tools

There are a few tools that are required for assembly. None of these tools are included. If you don't have them, now would be a good time to borrow or purchase them. They are very very handy whenever assembling/fixing/modifying electronic devices! I provide links to buy them, but of course, you should get them where ever is most convenient/inexpensive. Many of these parts are available in a place like Radio Shack or other (higher quality) DIY electronics stores.

Soldering iron

Any entry level 'all-in-one' soldering iron that you might find at your local hardware store should work. As with most things in life, you get what you pay for.

Upgrading to a higher end soldering iron setup, like the Hakko FX-888 that we stock in our store (http://adafru.it/180), will make soldering fun and easy.

Do not use a "ColdHeat" soldering iron! They are not suitable for delicate electronics work and can damage the kit (see here (http://adafru.it/aOo)).

Click here to buy our entry level adjustable 30W 110V soldering iron. (http://adafru.it/180)

Click here to upgrade to a Genuine Hakko FX-888 adjustable temperature soldering iron. (http://adafru.it/303)

Solder

You will want rosin core, 60/40 solder. Good solder is a good thing. Bad solder leads to bridging and cold solder joints which can be tough to find.

Click here to buy a spool of leaded solder (recommended for beginners). (http://adafru.it/145)

Click here to buy a spool of lead-free solder. (http://adafru.it/734)

Multimeter

You will need a good quality basic multimeter that can measure voltage and continuity.

Click here to buy a basic multimeter. (http://adafru.it/71)

Click here to buy a top of the line multimeter. (http://adafru.it/308)

Click here to buy a pocket multimeter. (http://adafru.it/850)

Flush Diagonal Cutters

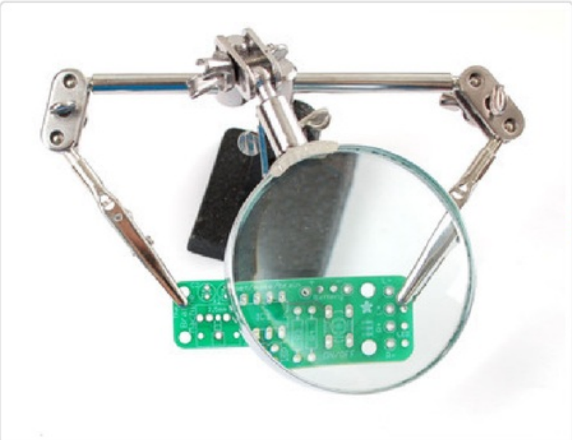
You will need flush diagonal cutters to trim the wires and leads off of components once you have soldered them in place.

Click here to buy our favorite cutters. (http://adafru.it/152)

Solder Sucker

Strangely enough, that's the technical term for this desoldering vacuum tool. Useful in cleaning up mistakes, every electrical engineer has one of these on their desk.

Click here to buy a one. (http://adafru.it/148)



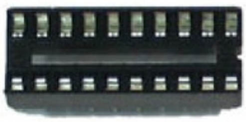
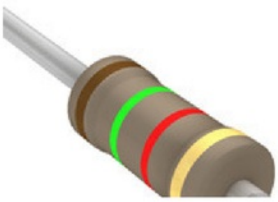
Helping Third Hand With Magnifier

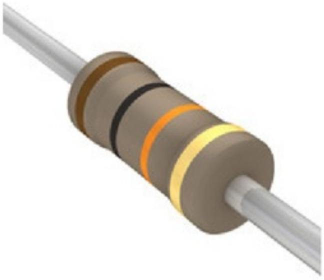
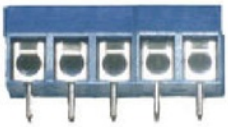
Not *absolutely* necessary but will make things go much much faster, and it will make soldering much easier.

Pick one up here. (http://adafru.it/291)

Parts List

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!

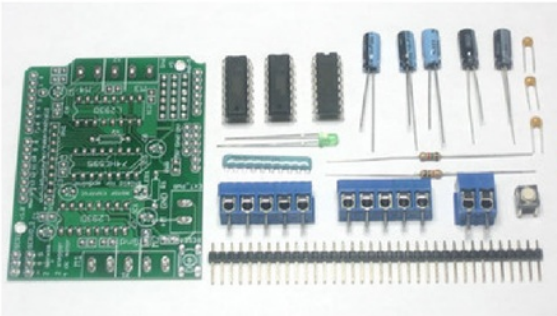
Image	Name	Description	Distributor	Qty
	PCB	Printed Circuit Board	Adafruit	1
	IC1, IC2	L293D Dual H-bridge *See note on usage page for replacing with SN754410	L293D	2
	IC3	74HC595N Serial to parallel output latch	74HC595N	1
	IC1' and IC2'	16 pin sockets (OPTIONAL!) These are included in kits as of July 2010	Generic	2
	LED1	3mm LED, any color Motor power indicator	3mm LED	1
	R1	1.5K resistor for LED1	1/4W 5% resistor	1

				
	R2	10K pulldown resistor Brown, Black, Orange, Gold	1/4W 5% resistor	1
	RN1	10-pin bussed 10K-100K resistor network	100K resistor network	1
	C2, C4, C6	0.1uF ceramic capacitor	Generic	3
	C1, C3, C5	100uF / 6V capacitor (or bigger)	100uF/6V cap	3
	C7, C8	47uF / 25V capacitor (or bigger)	47uF/25V cap	2
	X1	5-position 3.5mm terminal block (Or a 3-position and a 2-position)	3.5mm terminals	2
	X2	2-position 3.5mm terminal block	3.5mm terminals	1

		block		
	RESET	6mm tactile switch	6mm tact switch	1
	PWR	Jumper/shunt	0.1" jumper	1
		36 pin male header (1x36)	Generic	1

Solder It

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!

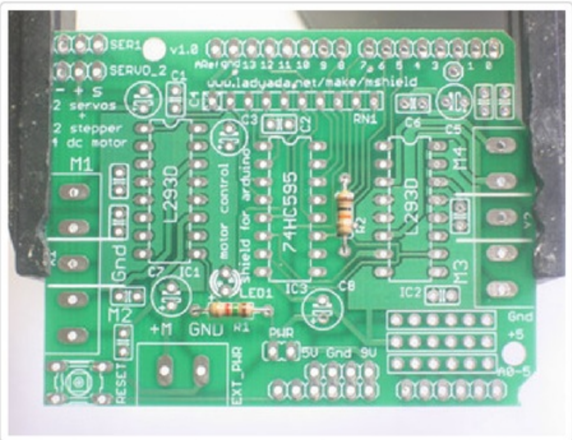


First, check that you have all the parts! Look over the parts list here (http://adafru.it/aOw) and shown on the left.

Also check to make sure you have the necessary tools for assembly. (http://adafru.it/aOv)

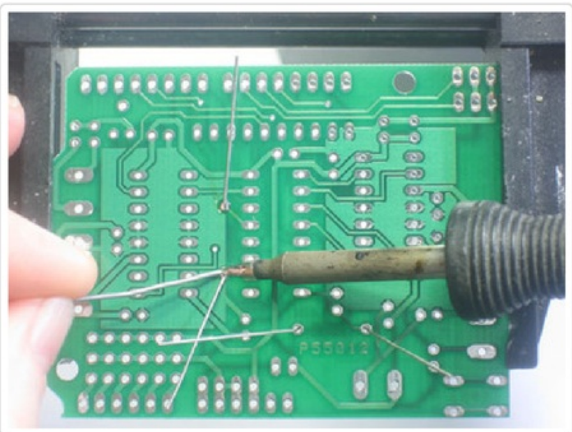
Place the motor shield PCB in a vise or other circuit-board holder and turn on your soldering iron to 700 degrees.

The first parts to go in are the two resistors, **R1** (**Brown Green Red Gold**) and **R2 (Brown Black Orange Gold)**. Bend the resistors so that they look like staples, as seen in this photo.



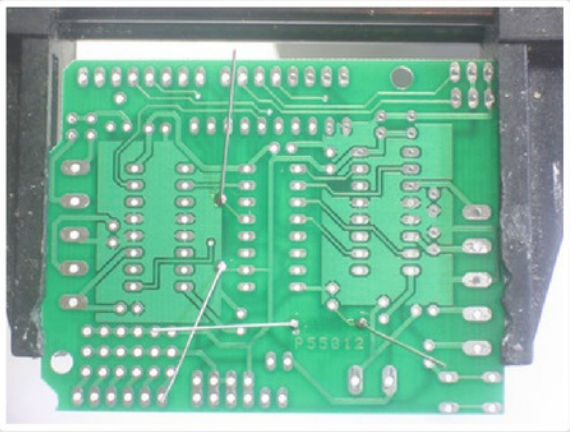
Next, slip the resistors into the PCB as shown, so that they sit flat against the circuit board. Bend the wire legs out a bit so that when the board is flipped over

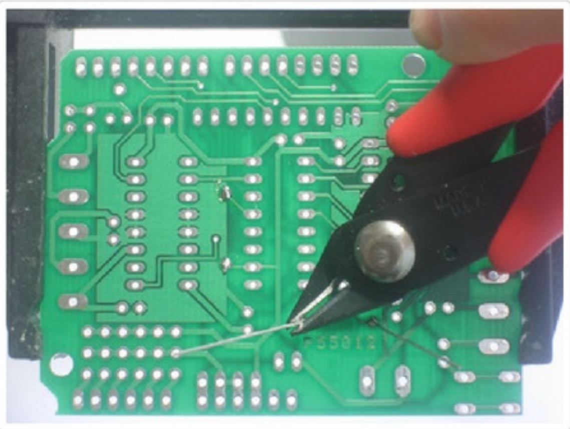
Resistors are not polarized, that means you can put them in "either way" and they'll work just fine.



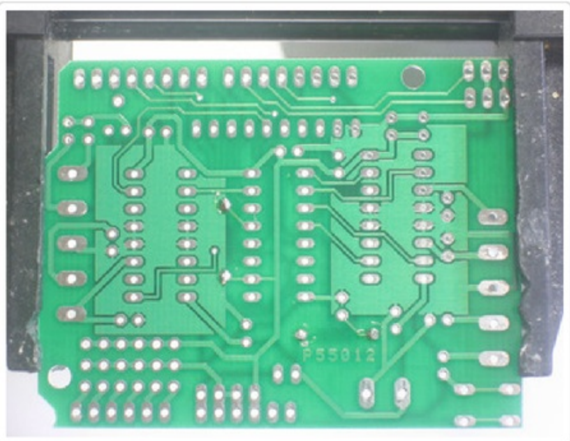
Using your soldering iron tip, heat the resistor wire lead and the metal ring (pad) at the same time, after a few seconds, poke a little solder in so that it melts into a nice cone. Remove the solder and then remove the soldering iron. Do this for all 4 wires.

Check your work, you should have clean solder joints.





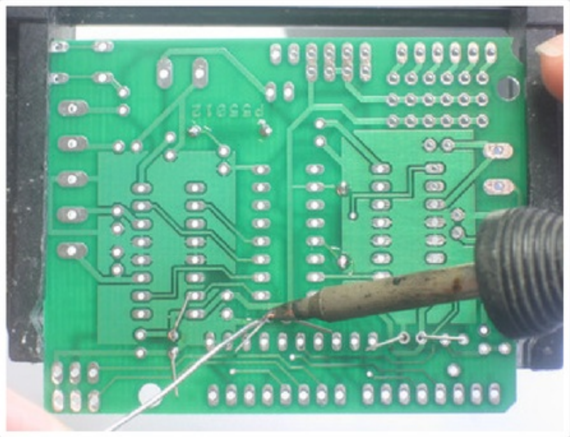
Clip the long leads, just above the solder joint using diagonal cutters.

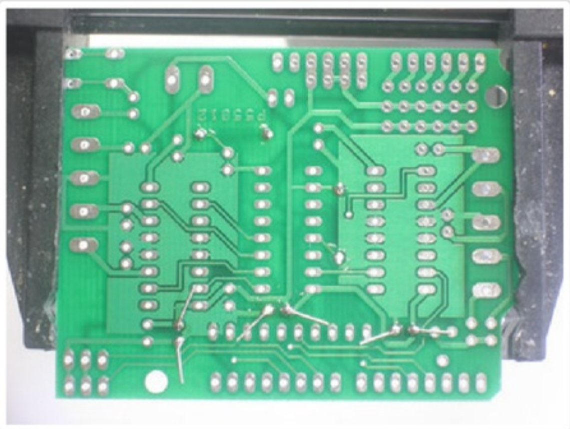


Next place the three yellow ceramic capacitors **C4**, **C2** and**C6**. Ceramic capacitors are not polarized so you can put them in "either way" and they work fine.

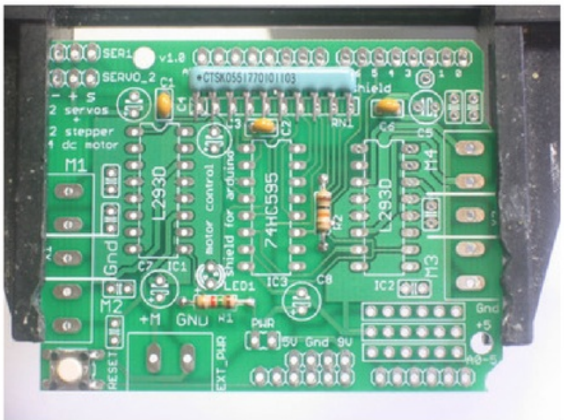
Bend the leads out just like you did with the resistors.

Solder all 6 wires, then clip them as you did with the resistors.



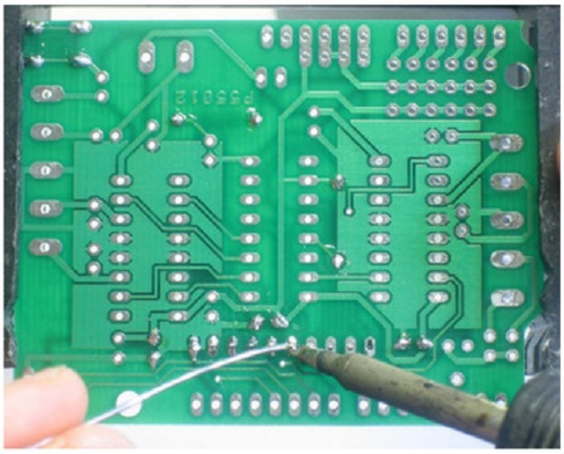


Next is the 6mm tactile switch **RESET** and the resistor network **RN1**. The tact switch is used to reset the Arduino since its not possible to reach the reset button once the motor shield is on.

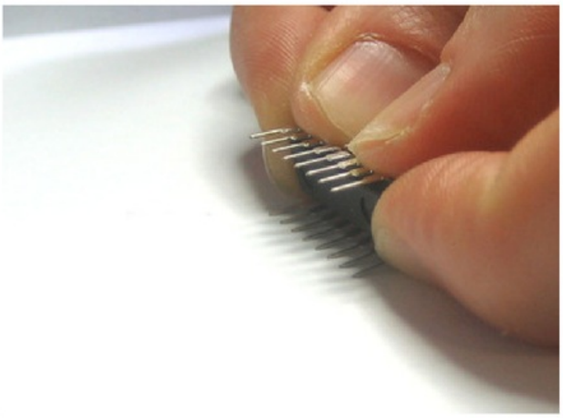


The resistor network is used to*pull-down* the pins on the motor driver chips so that they don't power up the motors before the Arduino sketch tells them to.

The tactile switch can go in 'either way'. The resistor network, however, must go in a certain way. Make sure the end with a dot is posititioned so it is at the same end as the X in the silkscreened image of the resistor network. (See picture on left).



Flip the board over and solder in the resistor network and switch. You won't need to clip the leads as they are quite short aleady.

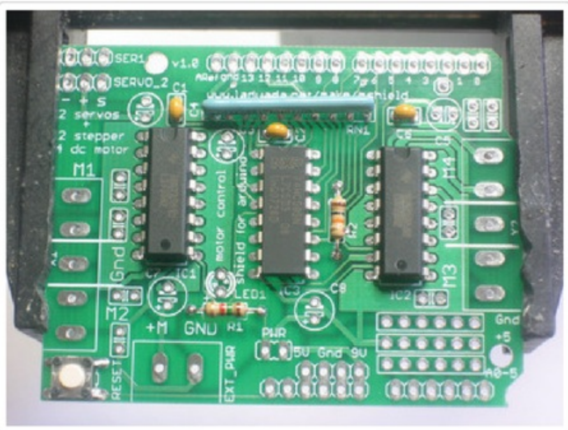


Next are the three integrated circuits (ICs) **IC1, IC2** and **IC3**. When ICs come from the factory, the legs are angled out somewhat which makes it difficult to insert them into the PCB. Prepare them for soldering by gently bending the legs against a flat tabletop so that they are perfectly straight.

The latest kits from Adafruit come with 2 16-pin sockets for the L293D motor drivers. They are OPTIONAL and not necessary for operation.

If you are not experienced with driving motors (your likelyhood of wiring up a mis-specified motor is high) you should install these so if the L293Ds are destroyed you can easily replace them

If you are experienced with driving motors, you may want to skip the sockets as the decrease the chips' heat-sinking abilities.



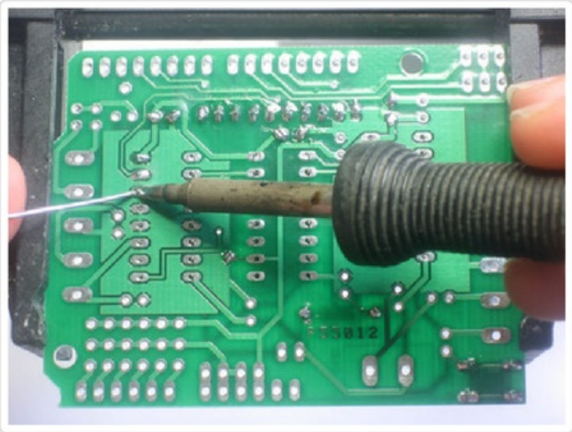
ICs must be placed in the correct orientation to work properly. To help with placement, each chip has a U notch at the top of the chip. On the circuit board there is a printed out image of the chip outline and one end has a U notch. Make sure the chip notch is on the same end as the image notch. In this PCB, all are facing the same way.

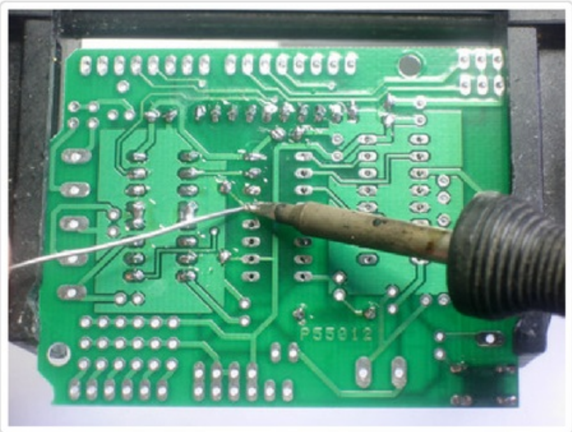
Gently insert the three chips. Check to make sure none of the legs got bent or broken.

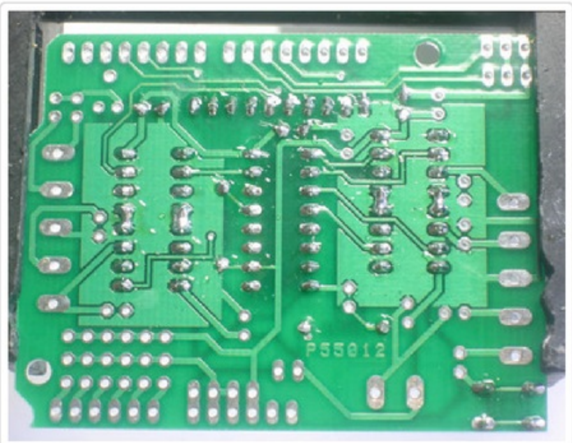
The 74HC595 goes in the middle, and the two L293Ds go on either side.

Solder each pin of the chips.

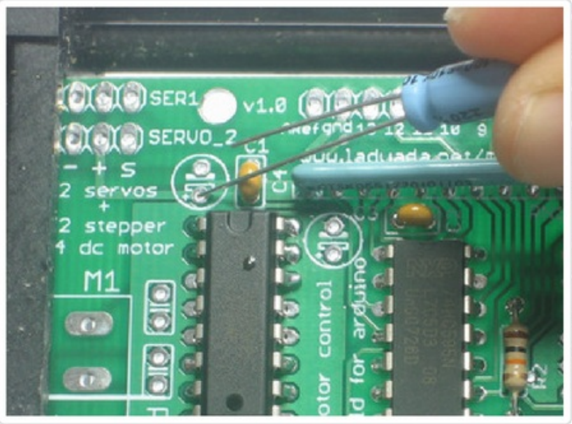
The four 'middle' pins of the L293D motor driver chips are tied to a large heat sink and thus may end up getting 'bridged' with solder as shown in the second image.





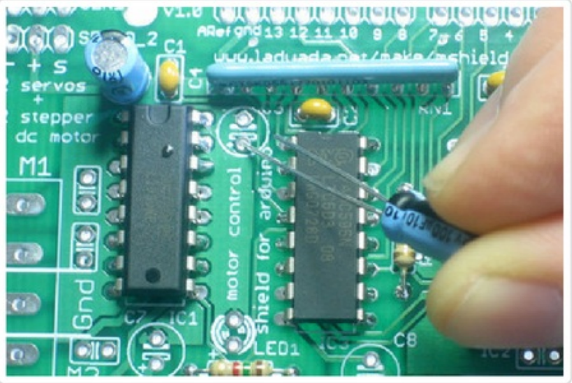


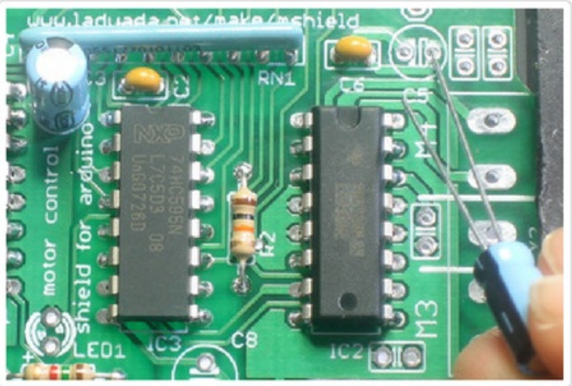
Next are the three 100uF electrolytic

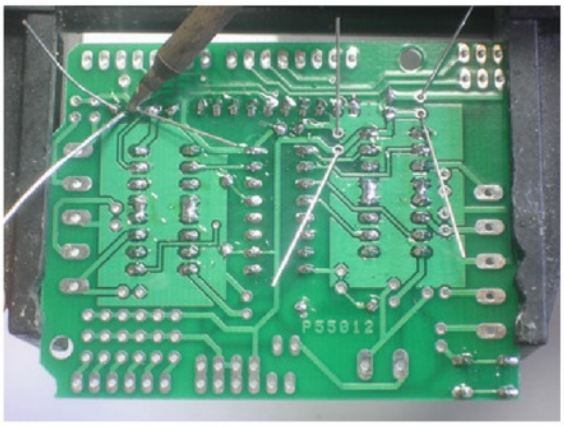


capacitors **C1**, **C3**and **C5**. Electrolytic capacitors are polarized and must be placed in the correct orientation or they could pop! The long leg of the capacitor is the positive (+) leg and goes into the hole marked with a +. The close-up images shown here indicate with hole is the + one.

Capacitors are not color-coded. The body color can vary from blue to violet to green to black so**be sure to read the value on the side, don't depend on the color!**



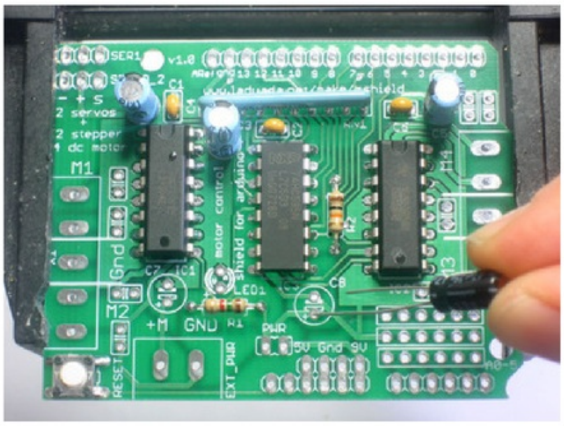




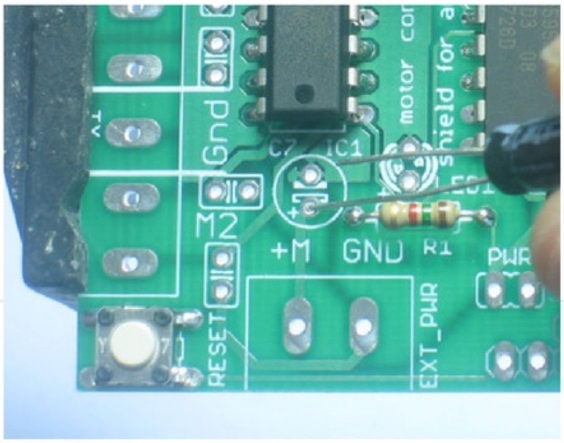
After double-checking their polarity, solder and clip the three capacitors.

Place the two 47uF remaining electrolytic capacitors, **C7** and**C8**

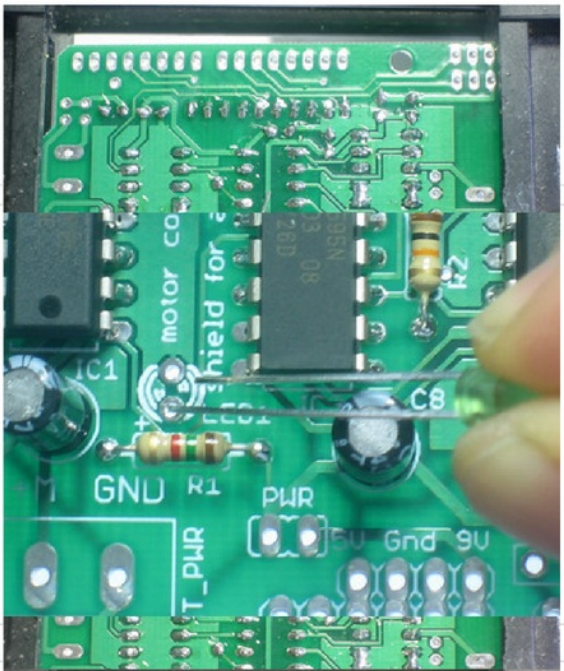
These are also polarized so make sure the long lead is inserted into the + hole in the silkscreened image.



Capacitors are not color-coded. The body color can vary from blue to violet to green to black so **be sure to read the value on the side, don't depend on the color!**

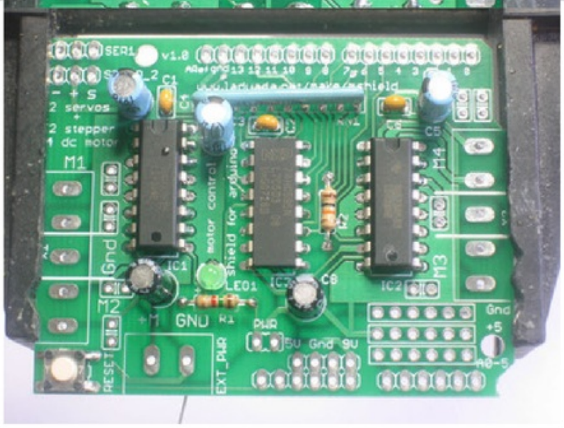


Solder and clip the two capacitors.

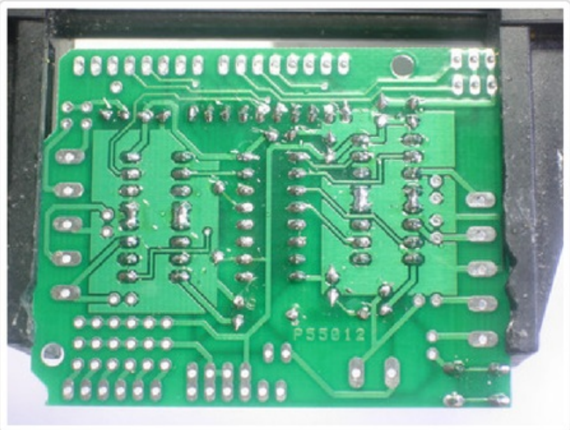


Next is the 3mm LED used to indicate motor power. LEDs are polarized, just like capacitors, and the long lead is the positive (+) lead.

Make sure the LED is placed correctly otherwise it wont work!



Solder and clip the LED leads.

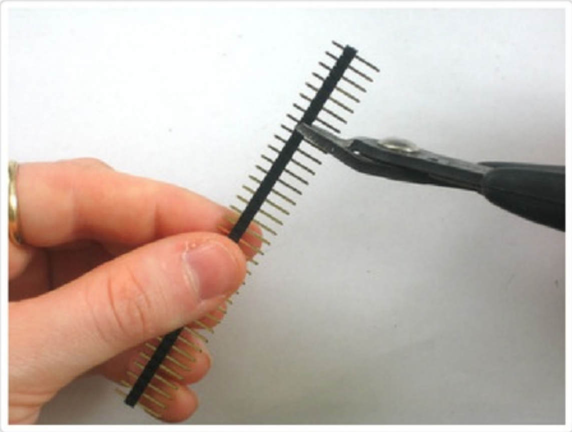


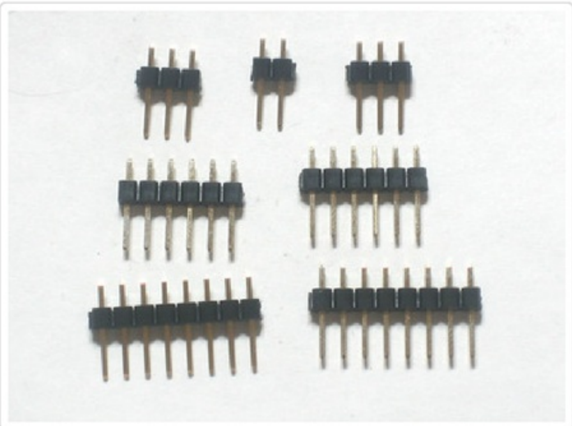
Next its time to make the headers for the jumper, servos and arduino.

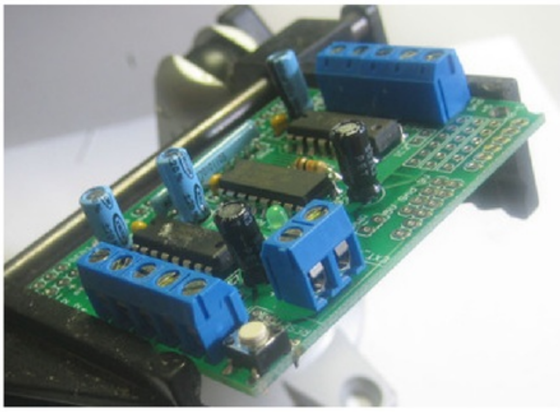
We use one stick of 36-pin 'breakaway' header, and break it apart to make smaller strips. You can use diagonal cutters or pliers to snap off the pieces.

Break the 36-pin header into 2 8-pin, 2 6-pin, 2 3-pin and 1 2-pin headers.

If you have an NG arduino, you may want 1 6-pin header and 1 4-pin header instead of 2 6-pin headers.

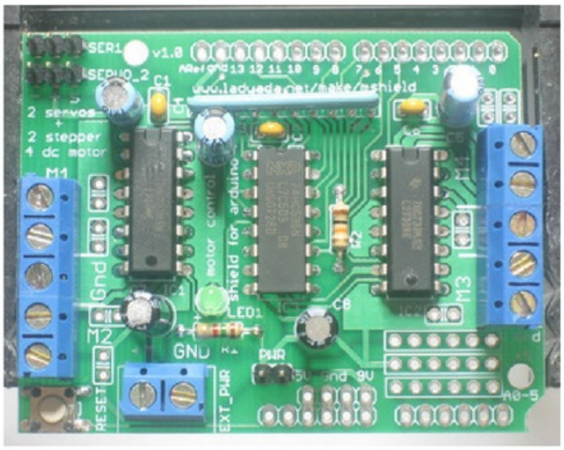




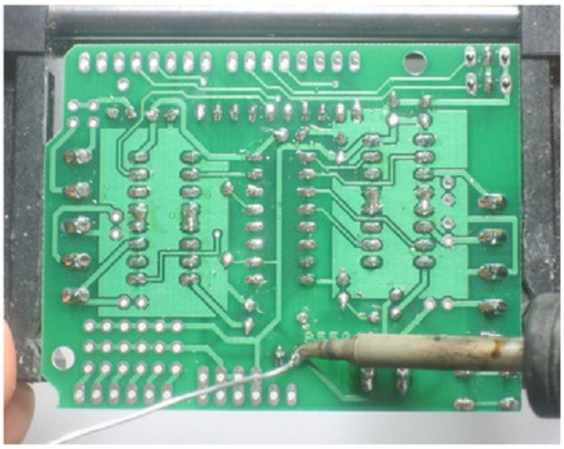


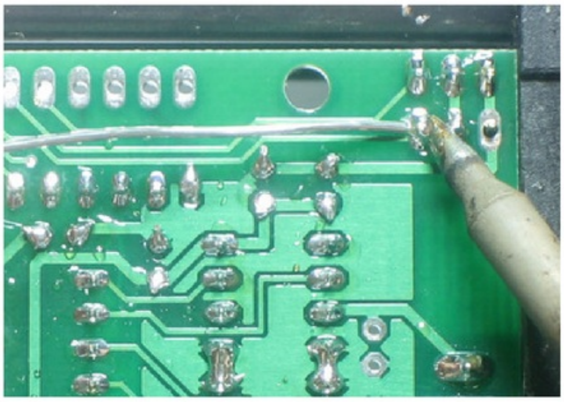
The 2 3-pin pieces go in the servo connections in the top left corner. The 2-pin piece goes in the PWR jumper in the bottom center.

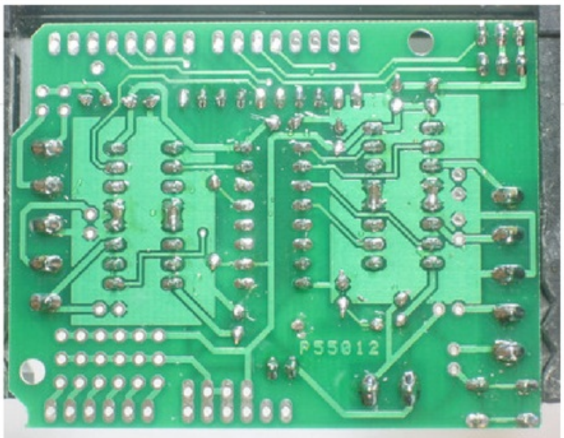
Also, place the 3 large screw terminals for the motor and external motor-power wires. If you received only 2 and 3-position terminal blocks, slide them together so that you have 2 5-position terminals and 1 2-position terminal.



Solder in the 3 pieces of header and the three terminal blocks.

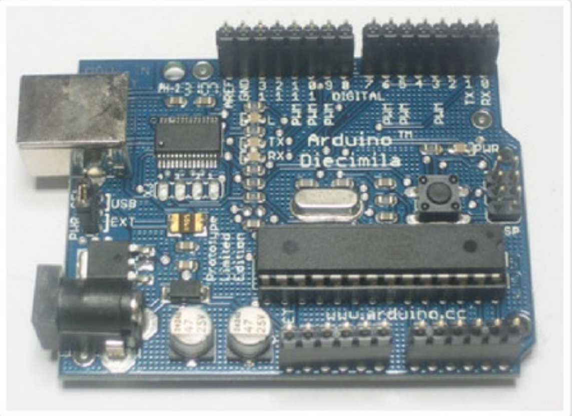


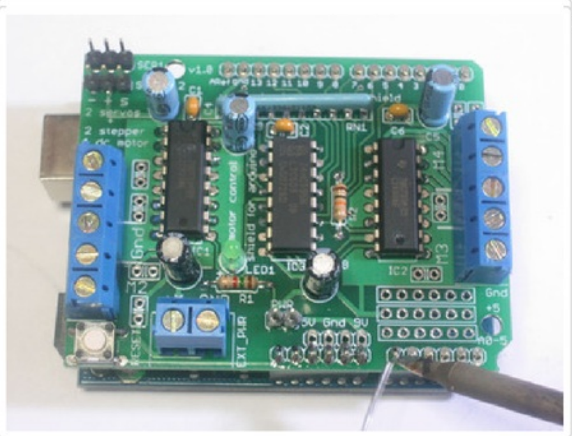




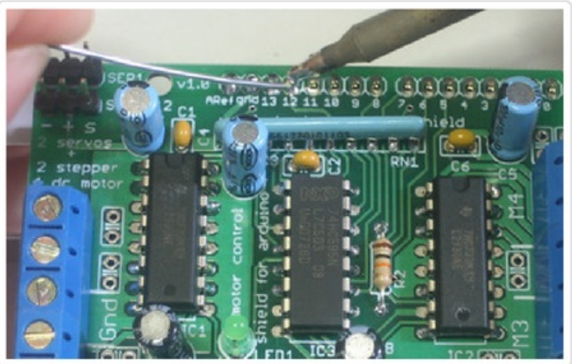
Next, place the 8-pin and 6-pin headers into the Arduino board. This will make sure that the headers are perfectly lined up. Make sure the Arduino is not plugged in or powered!

Place the motor shield on top of the Arduino, making sure that all the header lines up.





Solder in each pin of the header.



You're done!

Now go read the user manual. (http://adafru.it/aOz)

Use It!

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!

The Adafruit Motor Shield kit is a great motor controller for Arduino, but it does a little care to make sure it's used correctly. Please read through all the User manual sections at left, especially the section about library installation and power requirements!

Library Install

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!

First Install the Arduino Library

Before you can use the Motor shield, you **must** install the **AF_Motor** Arduino library - this will instruct the Arduino how to talk to the Adafruit Motor shield, and it isn't optional!

1. First, grab the library from github (http://adafru.it/aOA)
2. Uncompress the ZIP file onto your desktop
3. Rename the uncompressed folder **AFMotor**
4. Check that inside **AFMotor** is **AFMotor.cpp** and **AFMotor.h** files. If not, check the steps above
5. Place the **AFMotor** folder into your *arduinosketchfolder/**libraries*** folder. For Windows, this will probably be something like **MY Documents/Arduino/libraries** for Mac it will be something like**Documents/arduino/libraries**. If this is the first time you are installing a library, you'll need to create the **libraries** folder. Make sure to call it **libraries** exactly, no caps, no other name.
6. Check that inside the **libraries** folder there is the **AFMotor** folder, and inside **AFMotor** is**AFMotor.cpp AFMotor.h** and some other files
7. Quit and restart the IDE. You should now have a submenu called **File->Examples->AFMotor->MotorParty**

Power Usage

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!

Powering your DC motors, voltage and current requirements

Motors need a lot of energy, especially cheap motors since they're less efficient. The first important thing to figure out what voltage the motor is going to use. If you're lucky your motor came with some sort of specifications. Some small hobby motors are only intended to run at 1.5V, but its just as common to have 6-12V motors. The motor controllers on this shield are designed to run from **4.5V to 25V**.
MOST 1.5-3V MOTORS WILL NOT WORK

Current requirements: The second thing to figure out is how much current your motor will need. The motor driver chips that come with the kit are designed to provide up to 600 mA per motor, with 1.2A peak current. Note that once you head towards 1A you'll probably want to put a heatsink on the motor driver, otherwise you will get thermal failure, possibly burning out the chip.

On using the SN754410: Some people use the SN754410 (http://adafru.it/aOB) motor driver chip because it is pin-compatible, has output diodes and can provide 1A per motor, 2A peak. After careful reading of the datasheet and discussion with TI tech support and power engineers it appears that **the output diodes were designed for ESD protection only** and that using them as kickback-protection is a hack and not guaranteed for performance. For that reason the kit does not come with the SN754410 and instead uses the L293D with integrated kickback-protection diodes. If you're willing to risk it, and need the extra currrent, feel free to buy SN754410's and replace the provided chips.

Need more power? Buy another set of L293D drivers and solder them right on top of the ones on the board (piggyback) (http://adafru.it/aOC). Voila, double the current capability! You can solder 2 more chips on top before it probably isnt going to get you much benefit

You can't run motors off of a 9V battery so don't even waste your time/batteries! Use a big Lead Acid or NiMH battery pack. Its also very much suggested that you set up two power supplies (split supply) one for the Arduino and one for the motors. **99% of 'weird motor problems**' are due to noise on the power line from sharing power supplies and/or not having a powerful enough supply!

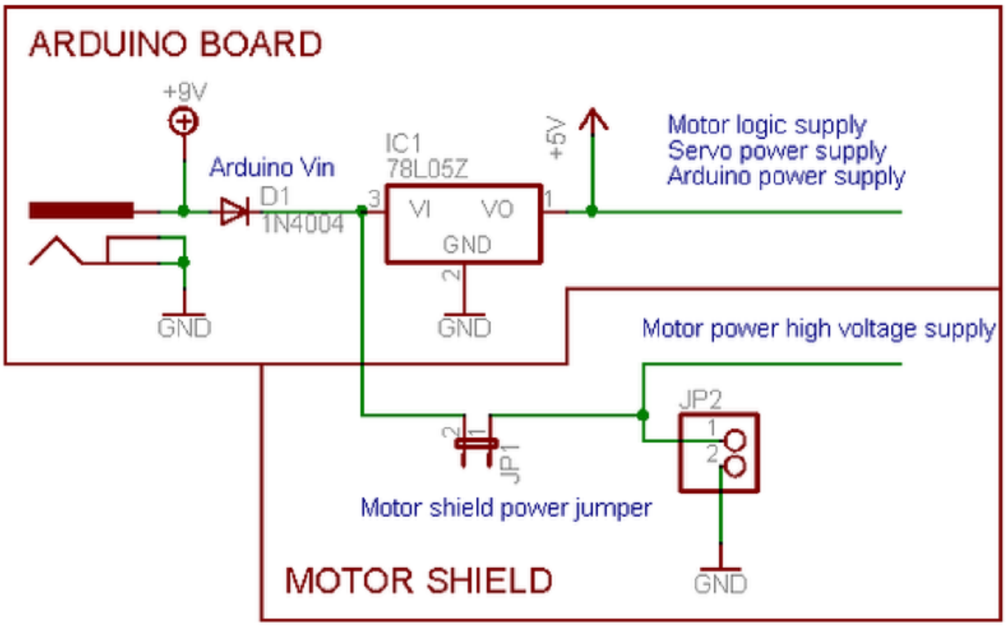
How to set up the Arduino + Shield for powering motors

Servos are powered off of the same regulated 5V that the Arduino uses. This is OK for the

small hobby servos suggested. If you want something beefier, cut the trace going to + on the servo connectors and wire up your own 5-6V supply!
The DC motors are powered off of a 'high voltage supply' and NOT the regulated 5V. **Don't connect the motor power supply to the 5V line**. This is a very very very bad idea unless you are sure you know what you're doing!

There are two places you can get your motor 'high voltage supply' from. One is the DC jack on the Arduino board and the other is the 2-terminal block on the shield that is labeled **EXT_PWR**. The DC Jack on the Arduino has a protection diode so you won't be able to mess things up too bad if you plug in the wrong kind of power. However the **EXT_PWR terminals on the shield do not have a protection diode** (for a fairly good reason). **Be utterly careful not to plug it in backwards** or you will destroy the motor shield and/or your Arduino!

Here's how it works:



If you would like to have a **single DC power supply for the Arduino and motors**, simply plug it into the DC jack on the Arduino or the 2-pin PWR_EXT block on the shield. Place the power jumper on the motor shield.
If you have a Diecimila Arduino, set the Arduino power source jumper to EXT.
Note that you may have problems with Arduino resets if the battery supply is not able to provide constant power, and it is not a suggested way of powering your motor project

If you would like to have the **Arduino powered off of USB** and the **motors powered off of a DC power supply**, plug in the USB cable. Then connect the motor supply to the PWR_EXT block on

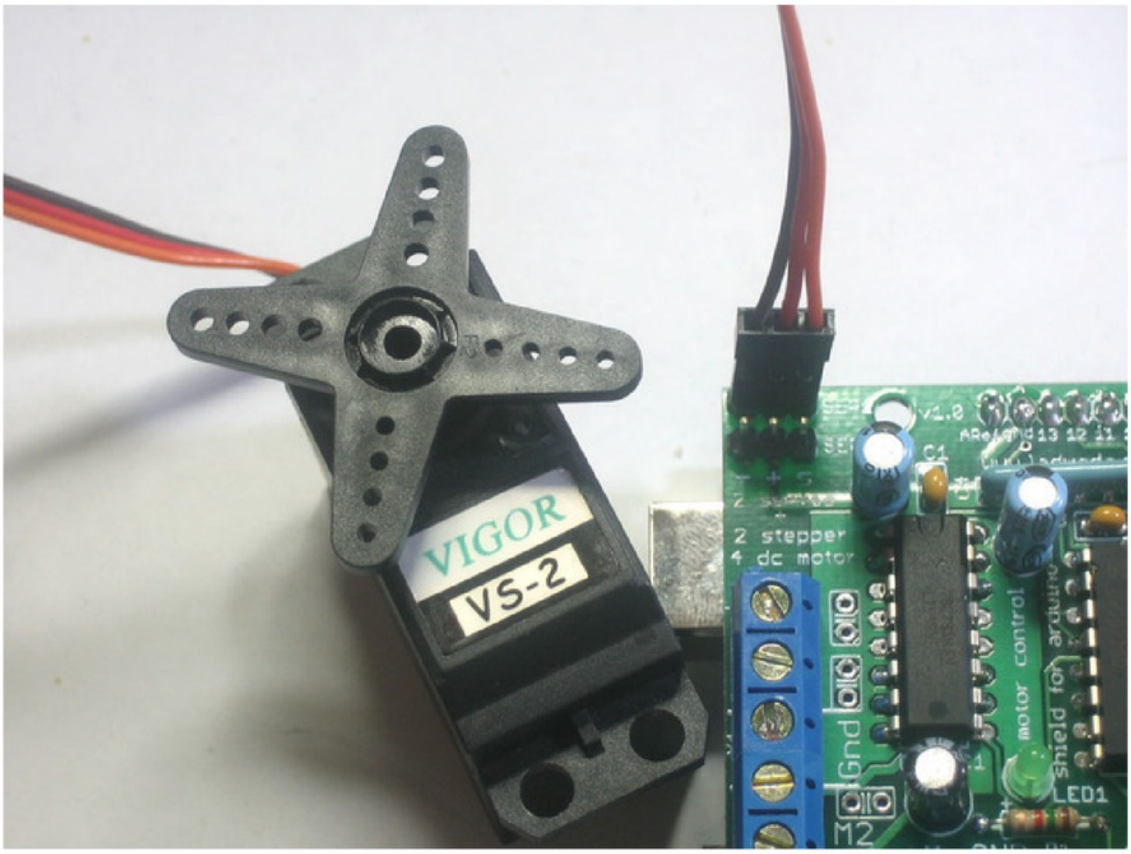
the shield. Do not place the jumper on the shield. This is a suggested method of powering your motor project
(If you have a Diecimila Arduino, don't forget to set the Arduino power jumper to USB. If you have a Diecimila, you can alternately do the following: plug the DC power supply into the Arduino, and place the jumper on the motor shield.)

If you would like to have **2 seperate DC power supplies for the Arduino and motors**. Plug in the supply for the Arduino into the DC jack, and connect the motor supply to the PWR_EXT block. Make sure the jumper is removed from the motor shield.
If you have a Diecimila Arduino, set the Arduino jumper to EXT. This is a suggested method of powering your motor project

Either way, if you want to use the DC motor/Stepper system the motor shield LED should be lit indicating good motor power

Using RC Servos

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!



Hobby servos are the easiest way to get going with motor control. They have a 3-pin 0.1" female header connection with +5V, ground and signal inputs. The motor shield simply brings out the 16bit PWM output lines to two 3-pin headers so that its easy to plug in and go. They can take a lot of power so a 9V battery wont last more than a few minutes!

The nice thing about using the onboard PWM is that its very precise and goes about its business in the background. You can use the built in **Servo** library

Using the servos is easy, please read the official Arduino documentation for how to use them and see the example Servo sketches in the IDE (http://adafru.it/aOD).

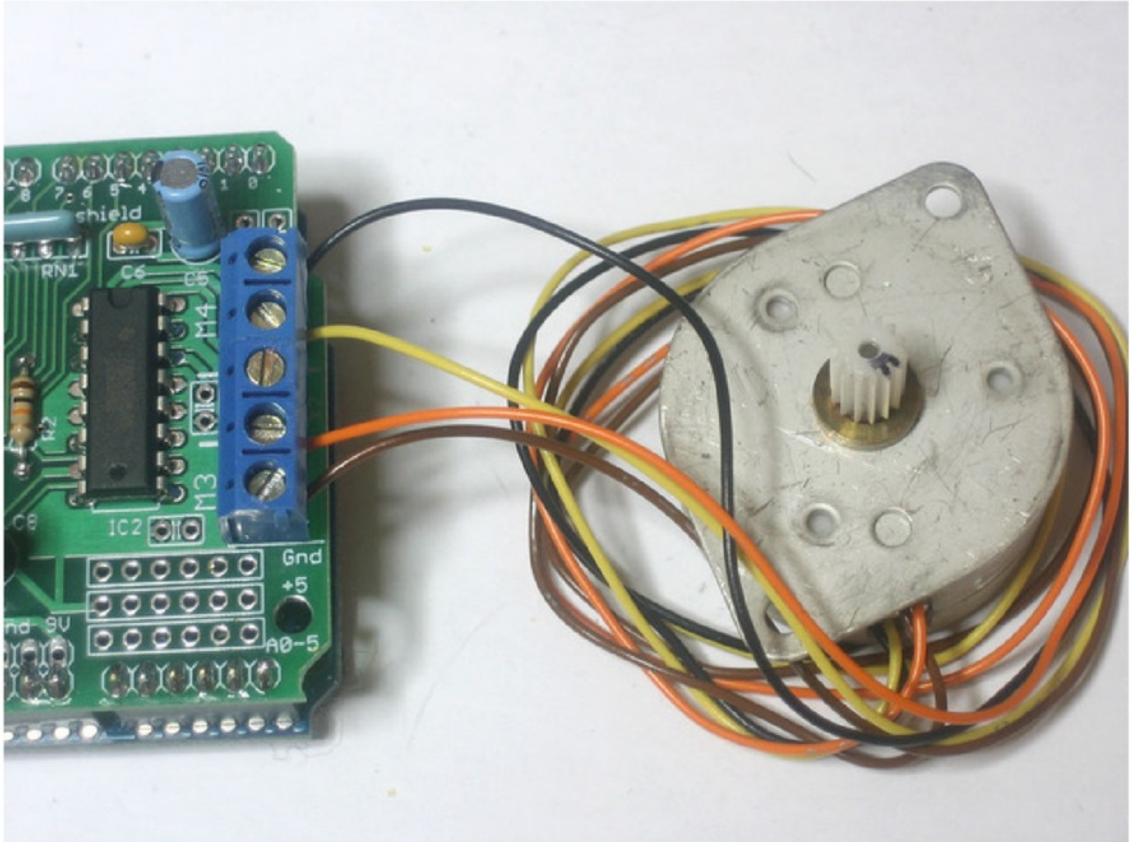
Power for the Servos comes from the Arduino's on-board 5V regulator, powered directly from

the USB or DC power jack on the Arduino. If you need an external supply, cut the trace right below the servo pins (on v1.2 boards) and connect a 5V or 6V DC supply directly. Using an external supply is for advanced users as you can accidentally destroy the servos by connecting a power supply incorrectly!

When using the external supply header for servos, take care that the bottom of the header pins do not contact the metal USB port housing on the Arduino. A piece of electrical tape on the housing will protect against shorts.

Using Stepper Motors

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!



Stepper motors are great for (semi-)precise control, perfect for many robot and CNC projects. This motor shield supports up to 2 stepper motors. The library works identically for bi-polar and uni-polar motors

For unipolar motors: to connect up the stepper, first figure out which pins connected to which coil, and which pins are the center taps. If its a 5-wire motor then there will be 1 that is the center tap for both coils. Theres plenty of tutorials online on how to reverse engineer the coils pinout. (http://adafru.it/aOO) The center taps should both be connected together to the GND terminal on the motor shield output block. then coil 1 should connect to one motor port (say M1 or M3) and coil 2 should connect to the other motor port (M2 or M4).

For bipolar motors: its just like unipolar motors except theres no 5th wire to connect to ground. The code is exactly the same.

Running a stepper is a little more intricate than running a DC motor but its still very easy

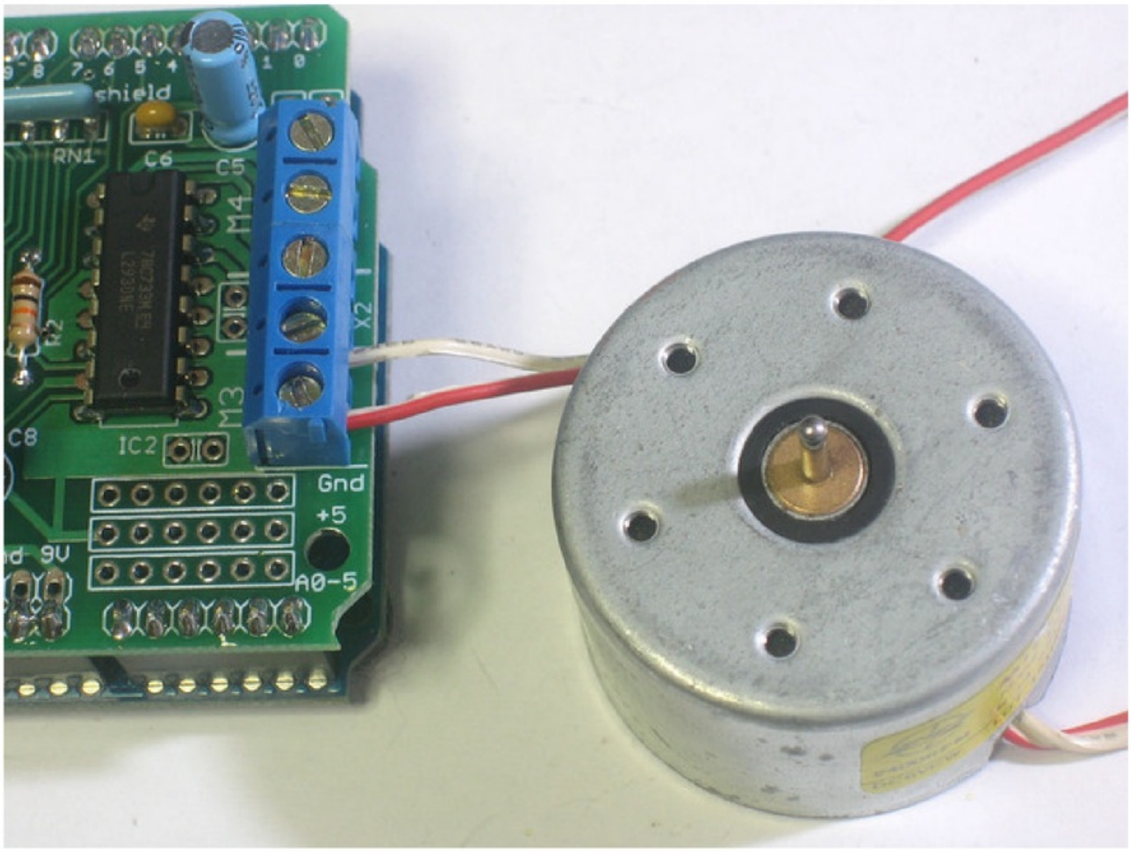
1. Make sure you #include <AFMotor.h>
2. Create the stepper motor object with **AF_Stepper(*steps*, *stepper#*)** to setup the motor H-bridge and latches. ***Steps*** indicates how many steps per revolution the motor has. a 7.5degree/step motor has 360/7.5 = 48 steps. ***Stepper#*** is which port it is connected to. If you're using M1 and M2, its port 1. If you're using M3 and M4 it's port 2
3. Set the speed of the motor using **setSpeed(*rpm*)** where ***rpm*** is how many revolutions per minute you want the stepper to turn.
4. Then every time you want the motor to move, call the **step(*#steps*, *direction*, *steptype*)** procedure.***#steps*** is how many steps you'd like it to take**.** ***direction*** is either **FORWARD** or **BACKWARD** and the step type is **SINGLE, DOUBLE. INTERLEAVE** or **MICROSTEP**. "Single" means single-coil activation, "double" means 2 coils are activated at once (for higher torque) and "interleave" means that it alternates between single and double to get twice the resolution (but of course its half the speed). "Microstepping" is a method where the coils are PWM'd to create smooth motion between steps. Theres tons of information about the pros and cons of these different stepping methods in the resources page. (http://adafru.it/aOO) You can use whichever stepping method you want, changing it "on the fly" to as you may want minimum power, more torque, or more precision.
5. By default, the motor will 'hold' the position after its done stepping. If you want to release all the coils, so that it can spin freely, call **release()**
6. The stepping commands are 'blocking' and will return once the steps have finished.

Because the stepping commands 'block' - you have to instruct the Stepper motors each time you want them to move. If you want to have more of a 'background task' stepper control, check out AccelStepper library (http://adafru.it/aOL) (install similarly to how you did with AFMotor) which has some examples for controlling two steppers simultaneously with varying accelleration

```
#include <AFMotor.h>


AF_Stepper motor(48, 2);


void setup() {
  Serial.begin(9600);           // set up Serial library at 9600 bps
  Serial.println("Stepper test!");

  motor.setSpeed(10);  // 10 rpm

  motor.step(100, FORWARD, SINGLE);
  motor.release();
  delay(1000);
}

void loop() {
  motor.step(100, FORWARD, SINGLE);
  motor.step(100, BACKWARD, SINGLE);

  motor.step(100, FORWARD, DOUBLE);
  motor.step(100, BACKWARD, DOUBLE);

  motor.step(100, FORWARD, INTERLEAVE);
  motor.step(100, BACKWARD, INTERLEAVE);

  motor.step(100, FORWARD, MICROSTEP);
  motor.step(100, BACKWARD, MICROSTEP);
}
```

Using DC Motors

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!



DC motors are used for all sort of robotic projects.

The motor shield can drive up to 4 DC motors bi-directionally. That means they can be driven forwards and backwards. The speed can also be varied at 0.5% increments using the high-quality built in PWM. This means the speed is very smooth and won't vary!

Note that the H-bridge chip is not meant for driving loads over 0.6A or that peak over 1.2A so this is for *small* motors. Check the datasheet for information about the motor to verify its OK.

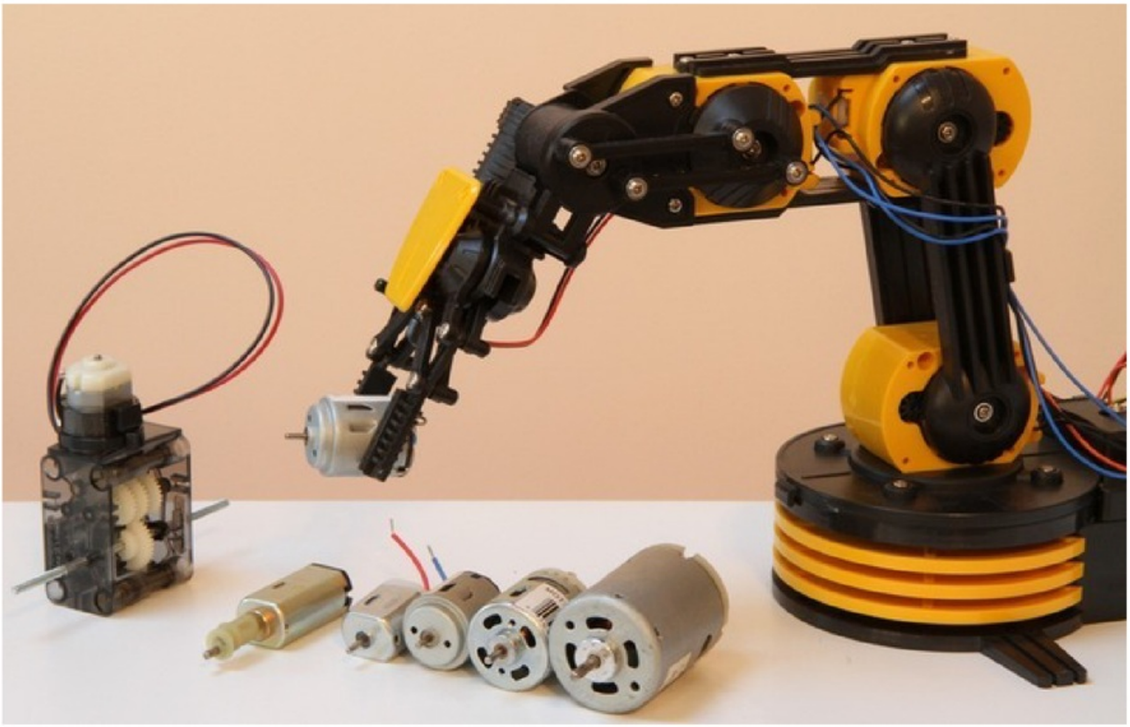
To connect a motor, simply solder two wires to the terminals and then connect them to either the **M1, M2, M3,** or **M4**. Then follow these steps in your sketch

1. Make sure you #include <AFMotor.h>
2. Create the AF_DCMotor object with **AF_DCMotor(*motor#, frequency*)**, to setup the motor H-bridge and latches. The constructor takes two arguments.
   The first is which port the motor is connected to, **1, 2, 3** or **4**.
   frequency is how fast the speed controlling signal is.
   For motors 1 and 2 you can choose **MOTOR12_64KHZ**, **MOTOR12_8KHZ**, **MOTOR12_2KHZ**, or**MOTOR12_1KHZ**. A high speed like 64KHz wont be audible but a low speed like 1KHz will use less power. Motors 3 & 4 are only possible to run at 1KHz and will ignore any setting given
3. Then you can set the speed of the motor using **setSpeed(*speed*)** where the ***speed*** ranges from 0 (stopped) to 255 (full speed). You can set the speed whenever you want.
4. To run the motor, call **run(*direction*)** where ***direction*** is **FORWARD**, **BACKWARD** or **RELEASE**. Of course, the Arduino doesn't actually know if the motor is 'forward' or 'backward', so if you want to change which way it thinks is forward, simply swap the two wires from the motor to the shield.

```
#include <AFMotor.h>

AF_DCMotor motor(2, MOTOR12_64KHZ); // create motor #2, 64KHz pwm

void setup() {
  Serial.begin(9600);           // set up Serial library at 9600 bps
  Serial.println("Motor test!");

  motor.setSpeed(200);     // set the speed to 200/255
}

void loop() {
  Serial.print("tick");

  motor.run(FORWARD);      // turn it on going forward
  delay(1000);

  Serial.print("tock");
  motor.run(BACKWARD);     // the other way
  delay(1000);

  Serial.print("tack");
  motor.run(RELEASE);      // stopped
  delay(1000);
}
```

AF_DCMotor Class

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!



The AF_DCMotor class provides speed and direction control for up to four DC motors when used with the Adafruit Motor Shield. To use this in a sketch you must first add the following line at the beginning of your sketch:

```
#include <AFMotor.h>
```

AF_DCMotor *motorname*(*portnum*, *freq*)

> This is the constructor for a DC motor. Call this constructor once for each motor in your sketch. Each motor instance must have a different name as in the example below.

Parameters:

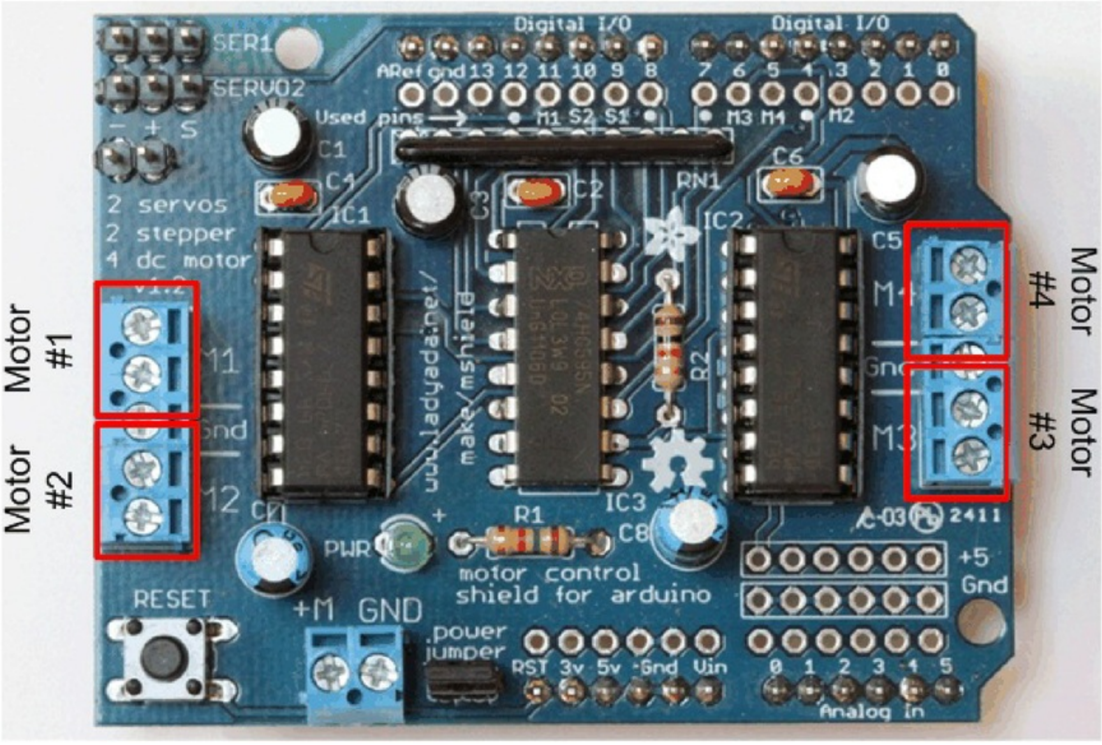
- **port num** - selects which channel (1-4) of the motor controller the motor will be connected to

- **freq** - selects the PWM frequency. If no frequency is specified, 1KHz is used by default.

> > Frequencies for channel 1 & 2 are:
> >
> > - MOTOR12_64KHZ
> > - MOTOR12_8KHZ
> > - MOTOR12_2KHZ
> > - MOTOR12_1KHZ
> >
> > Frequencies for channel 3 & 4 are:
> >
> > - MOTOR34_64KHZ
> > - MOTOR34_8KHZ
> > - MOTOR34_1KHZ

Example:

```
AF_DCMotor motor4(4); // define motor on channel 4 with 1KHz default PWM
AF_DCMotor left_motor(1, MOTOR12_64KHZ);  // define motor on channel 1 with 64KHz PWM
```



Note: *Higher frequencies will produce less audible hum in operation, but may result in lower torque with some motors.*

setSpeed(*speed*)

> Sets the speed of the motor.

Parameters:

- **speed**- Valid values for 'speed' are between 0 and 255 with 0 being off and 255 as full throttle.

Example:

Note: *DC Motor response is not typically linear, and so the actual RPM will not necessarily be proportional to the programmed speed.*

run(*cmd*)

> Sets the run-mode of the motor.

Parameters:

- **cmd** - the desired run mode for the motor

> Valid values for cmd are:
>
> - **FORWARD** - run forward (actual direction of rotation will depend on motor wiring)
> - **BACKWARD** - run backwards (rotation will be in the opposite direction from FORWARD)
> - **RELEASE** - Stop the motor. This removes power from the motor and is equivalent to setSpeed(0). The motor shield does not implement dynamic breaking, so the motor may take some time to spin down

Example:

```
motor.run(FORWARD);
delay(1000);  // run forward for 1 second
motor.run(RELEASE);
delay(100);  // 'coast' for 1/10 second
motor.run(BACKWARDS);  // run in reverse
```

AF_Stepper Class

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!

The AF_Stepper class provides single and multi-step control for up to 2 stepper motors when used with the Adafruit Motor Shield. To use this in a sketch you must first add the following line at the beginning of your sketch:

```
#include <AFMotor.h>
```

AF_Stepper *steppername*(*steps*, *portnumber*)

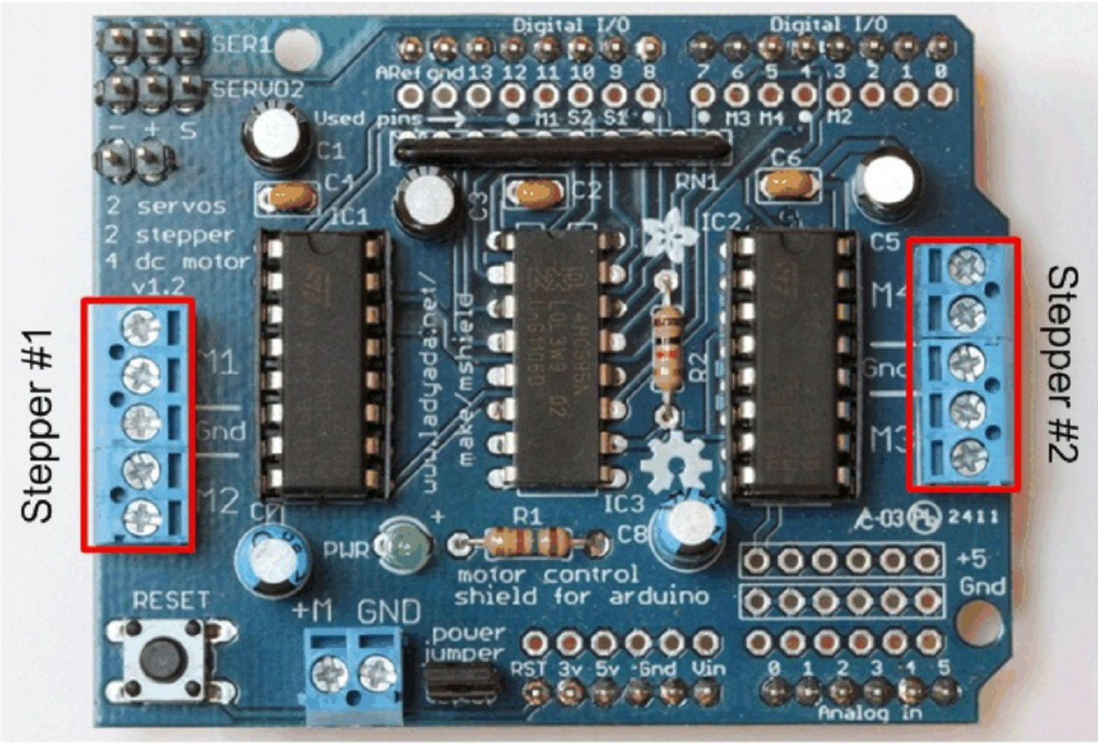
The AF_Stepper constructor defines a stepper motor. Call this once for each stepper motor in your sketch. Each stepper motor instance must have a unique name as in the example below.

Parameters:

- **steps** - declare the number of steps per revolution for your motor.
- **num** - declare how the motor will be wired to the shield.

> Valid values for 'num' are 1 (channels 1 & 2) and 2 (channels 3 & 4).

Example:




```
AF_Stepper Stepper1(48, 1);  // A 48-step-per-revolution motor on channels 1 & 2
AF_Stepper Stepper2(200, 2);   // A 200-step-per-revolution motor on channels 3 & 4
```

step(*steps, direction, style*)

Step the motor.

Parameters:

- **steps** - the number of steps to turn
- **direction** - the direction of rotation (**FORWARD** or **BACKWARD**)
- **style** - the style of stepping:

> Valid values for 'style' are:

> - **SINGLE** - One coil is energized at a time.
> - **DOUBLE** - Two coils are energized at a time for more torque.
> - **INTERLEAVE** - Alternate between single and double to create a half-step in between. This can result in smoother operation, but because of the extra half-step, the speed is reduced by half too.
> - **MICROSTEP** - Adjacent coils are ramped up and down to create a number of 'micro-

> steps' between each full step. This results in finer resolution and smoother rotation, but with a loss in torque.

Note: *Step is a synchronous command and will not return until all steps have completed. For concurrent motion of two motors, you must handle the step timing for both motors and use the "onestep()" function below.*

```
Stepper1.step(100, FORWARD, DOUBLE); // 100 steps forward using double coil stepping
Stepper2.step(100, BACKWARD, MICROSTEP);   // 100 steps backward using double microstepping
```

setSpeed(*RPMspeed*)

set the speed of the motor

Parameters:

- Speed - the speed in RPM

Note: *The resulting step speed is based on the 'steps' parameter in the constructor. If this does not match the number of steps for your motor, you actual speed will be off as well.*

Example:

```
Stepper1.setSpeed(10);  // Set motor 1 speed to 10 rpm
Stepper2.setSpeed(30);  // Set motor 2 speed to 30 rpm
```

onestep(*direction, stepstyle*)

Single step the motor.

Parameters:

- **direction** - the direction of rotation (**FORWARD** or **BACKWARD**)
- **stepstyle** - the style of stepping:

> Valid values for 'style' are:

> - **SINGLE** - One coil is energized at a time.
> - **DOUBLE** - Two coils are energized at a time for more torque.
> - **INTERLEAVE** - Alternate between single and double to create a half-step in between. This can result in smoother operation, but because of the extra half-step, the speed is reduced by half too.

- **MICROSTEP** - Adjacent coils are ramped up and down to create a number of 'micro-steps' between each full step. This results in finer resolution and smoother rotation, but with a loss in torque.

Example:

```
Stepper1.onestep(FORWARD, DOUBLE);  // take one step forward using double coil stepping
```

release()

Release the holding torque on the motor. This reduces heating and current demand, but the motor will not actively resist rotation.

Example:

```
Stepper1.release(); // stop rotation and turn off holding torque.
```

Resources

Motor ideas and tutorials

- Wikipedia has tons of information (http://adafru.it/aOF) on steppers
- Jones on stepper motor types (http://adafru.it/aOH)
- Jason on reverse engineering the stepper wire pinouts (http://adafru.it/aOI)

Downloads

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!

Schematics & Layout

You can grab the latest Schematic, Layout files (EagleCAD format from github. Click the **ZIP download** button at top middle to download the entire zip. (http://adafru.it/aOJ)

Firmware

- Arduino Stepper/Servo software library with microstepping support (http://adafru.it/aOK).
  To install, click on **Downloads** in the middle of the page, select **Download as zip** and uncompress the folder.
  Rename the folder to **AFmotor** (check that the renamed folder contains the .cpp and .h files) and install into the **Arduinosketches/libraries** folder. For information how to use and install libraries, see our tutorial (http://adafru.it/aYG)! This version now works with with the Mega. Public domain!
- AccelStepper library (http://adafru.it/aOL) with AFMotor support. This library allows for advanced stepper control including accelleration and decelleration, and concurrent stepper control! You *still need AFmotor above!*
  To install, click on **Download** in the middle of the page, select **Download as zip** and uncompress the folder.
  Rename the folder to **AccelStepper** (check that the renamed folder contains the .cpp and .h files) and install into the **Arduinosketches/libraries** folder. For information how to use and install libraries, see our tutorial (http://adafru.it/aYG)!

Forums

Forums (http://adafru.it/aOM)

10 Position Encoder Resolution Data Sheet

MTR-10-10E		
MM-3M-ST, -F, -FOS, -EX, MM-4M-F		
80 TPI Lead Screw (0.3175 mm/turn)		10 position encoder[1]
GH[2] Ratio	Max Travel Rate[3] (mm/sec)	Resolution (µm/count)
16:1	6.614	0.4961
64:1	1.653	0.1240
256:1	0.413	0.0310
1024:1	0.103	0.0078
MM-3M-ST, -F, -EX, MM-4M-F		
40 TPI Lead Screw (0.635 mm/turn)		10 position encoder[1]
GH[2] Ratio	Max Travel Rate[3] (mm/sec)	Resolution (µm/count)
16:1	13.229	0.9922
64:1	3.307	0.2481
256:1	0.827	0.0620
1024:1	0.207	0.0155

Notes:

1. The 10mm motors used with both linear and rotary stages incorporate dual channel, 10 position, magnetic encoders. The quadrature output is equivalent to 40 encoder counts per motor armature revolution.
2. Gearhead ratio is denoted by GH.
3. Maximum travel rate is calculated with the motor armature turning at a maximum rate of 20,000 RPM.

Linear Travel

Travel rate calculations

Lead screw RPM = motor RPM)/(gearhead ratio)
Distance per minute = (lead screw RPM) x lead; (lead = 0.3175 mm for 80 TPI screw and 0.635 mm for 40 TPI screw)
Distance per second = (distance per minute)/60
Distance in inches = (distance (mm))/(25.4)

Example calculation: with motor RPM = 20,000; GH ratio = 16:1; lead = 0.3175 mm
Distance per second = [(20000 RPM)/(16)] x (0.3175 mm) x (min/60 sec)] = 6.6145 mm/sec

Encoder resolution calculations

Encoder counts per lead screw revolution = (encoder counts per motor revolution) x (gearhead ratio)
Distance per encoder count = lead/(encoder counts per lead screw revolution)

Example calculation: with motor GH ratio = 16:1; lead = 0.3175 mm; 40 encoder counts per motor revolution
Distance per encoder count = (0.3175 mm)/(40 x 16) = 0.4961 µm/count

10 Position Encoder Resolution Data Sheet (cont.)

MTR-10-10E

MM-3M-R			
80:1 Worm Drive Ratio			10 position encoder[1]
GH[2] Ratio	Final Output	Max Travel Rate[3] (rad/sec)	Resolution (µrad/count)
16:1	1,280:1	1.636	122.7185
64:1	5,120:1	0.409	30.6796
256:1	20,480:1	0.102	7.6699
1024:1	81,920:1	0.025	1.9175

Notes:

1. The 10mm motors used with both linear and rotary stages incorporate dual channel, 10 position, magnetic encoders. The quadrature output is equivalent to 40 encoder counts per motor armature revolution.
2. Gearhead ratio is denoted by GH.
3. Maximum travel rate is calculated with the motor armature turning at a maximum rate of 20,000 RPM.

Rotary Travel

Travel rate calculations

Rotor travel rate (RPM) = (motor RPM)/[gearhead ratio) x (worm drive ratio)]
Rotor travel rate (rad/sec) = [rotor travel rate (RPM)] x (min/60 sec) x (6.283185 rad/revolution)

Example calculation: with motor RPM = 20,000; GH ratio = 16:1; lead = 0.3175 mm
Rotor travel rate (rad/sec) = (20000 RPM)/(16 x 80) x (min/60 sec) x (6.283185 rad/revolution) = 1.63624 rad/sec

Encoder resolution calculations

Encoder counts per lead screw revolution = [(encoder counts per motor revolution)] x (gearhead ratio) x (worm drive ratio)
Angular resolution = (6.283185 rad/revolution)/ (encoder counts per lead screw revolution)

Example calculation: with motor GH ratio = 16:1; lead = 0.3175 mm; 40 encoder counts per motor revolution
Angular resolution = (6.283185 rad per lead screw revolution)/[(40 counts per motor revolution) x (16 motor revolutions per gearhead revolution) x (80 gearhead revolutions per lead screw revolution)]
= 122.718 µrad/count

Conversion:

1 inch (in) = 25.4 mm
1 inch = 25,400 µm
1 millimeter (mm) = 39.37E-3 inch
1 micron (µm) = 39.37E-6 inch
1 deg = 0.01745329252 rad