Additive Manufacturing – Module 3

Spring 2015

Wenchao Zhou

zhouw@uark.edu

(479) 575-7250

The Department of Mechanical Engineering
University of Arkansas, Fayetteville

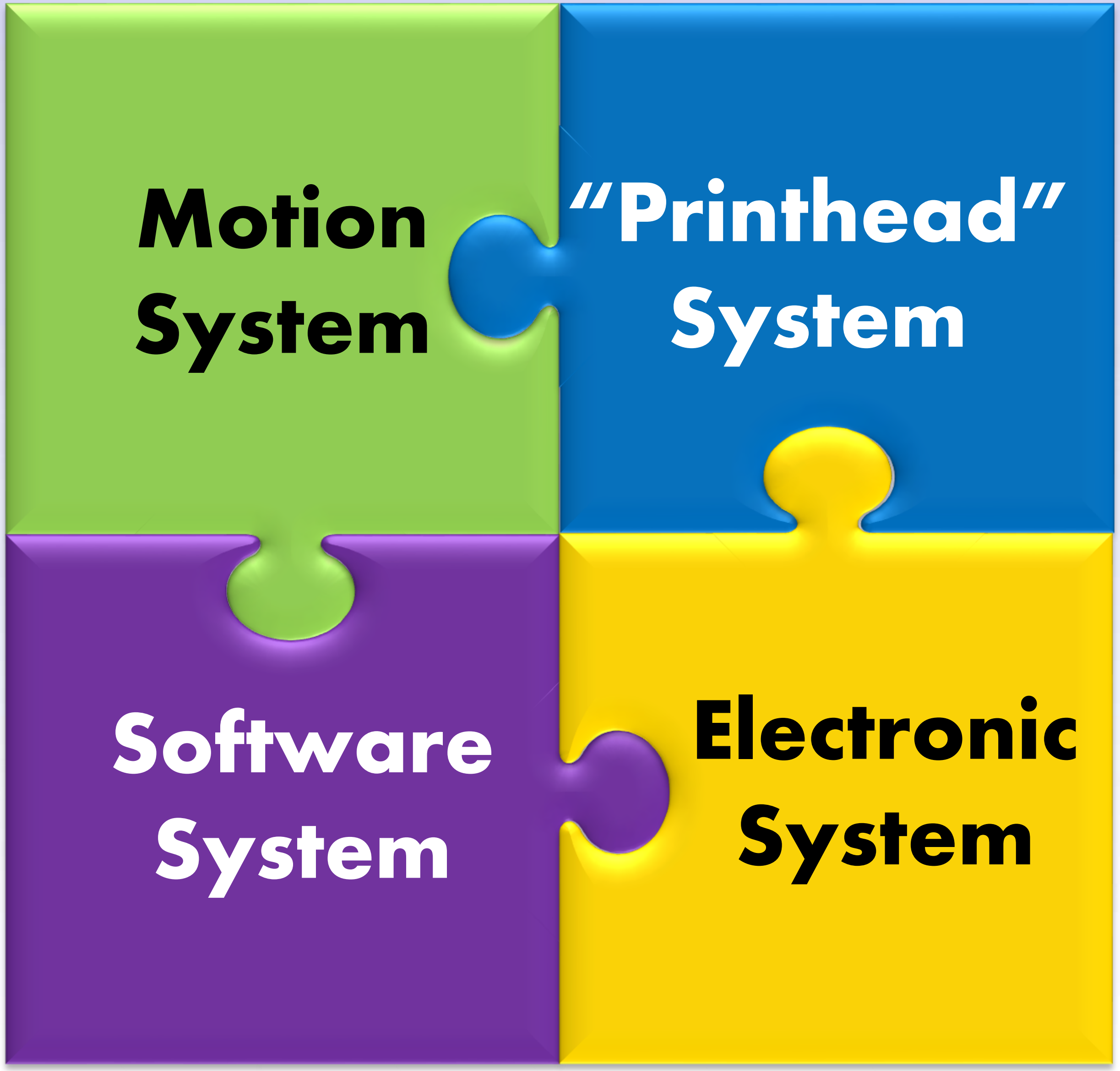
AM Machine

AM Machine

Motion System

Electronic System

- Subsystems – Modular design



Motion System

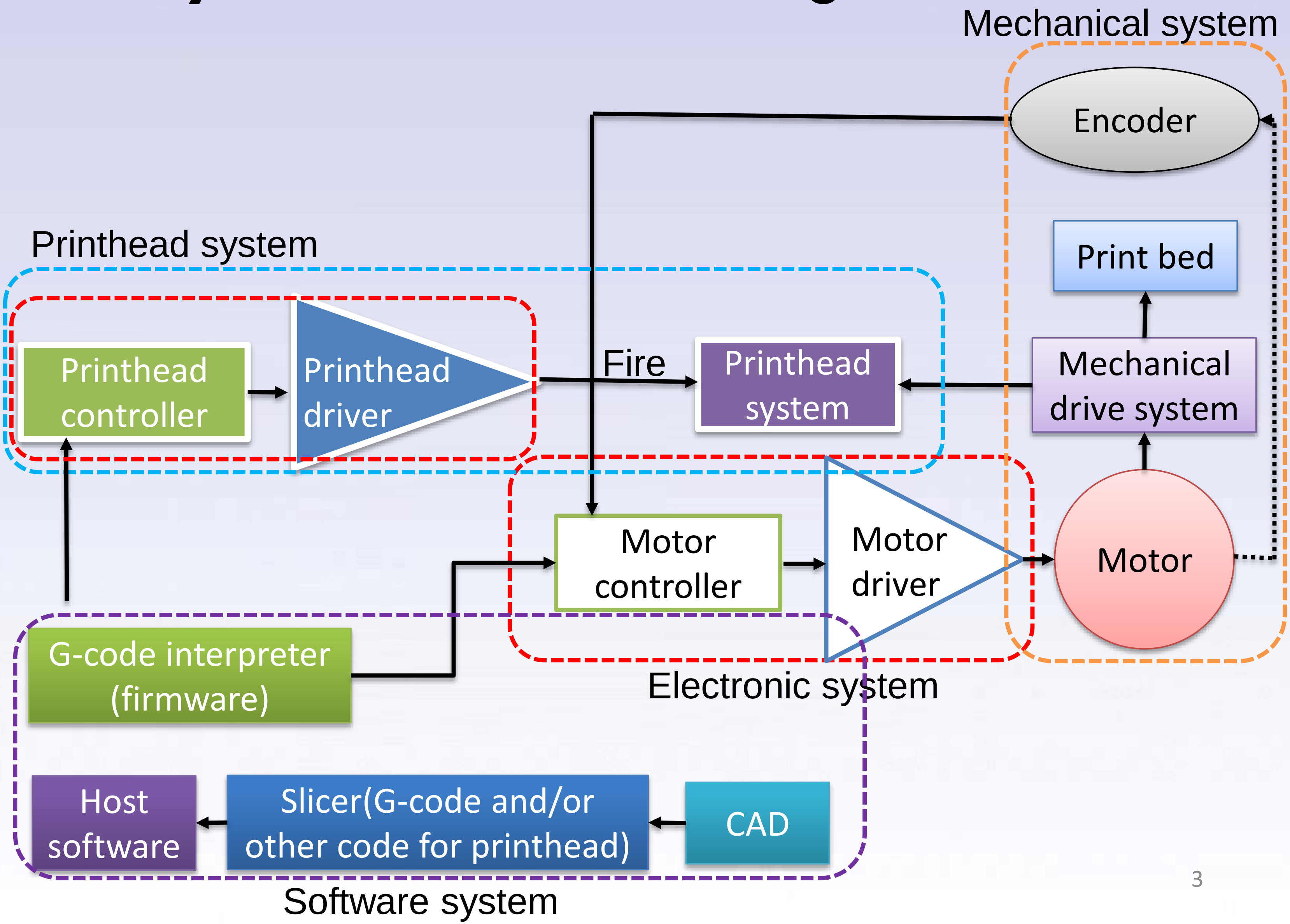
"Printhead" System

Software System

Electronic System

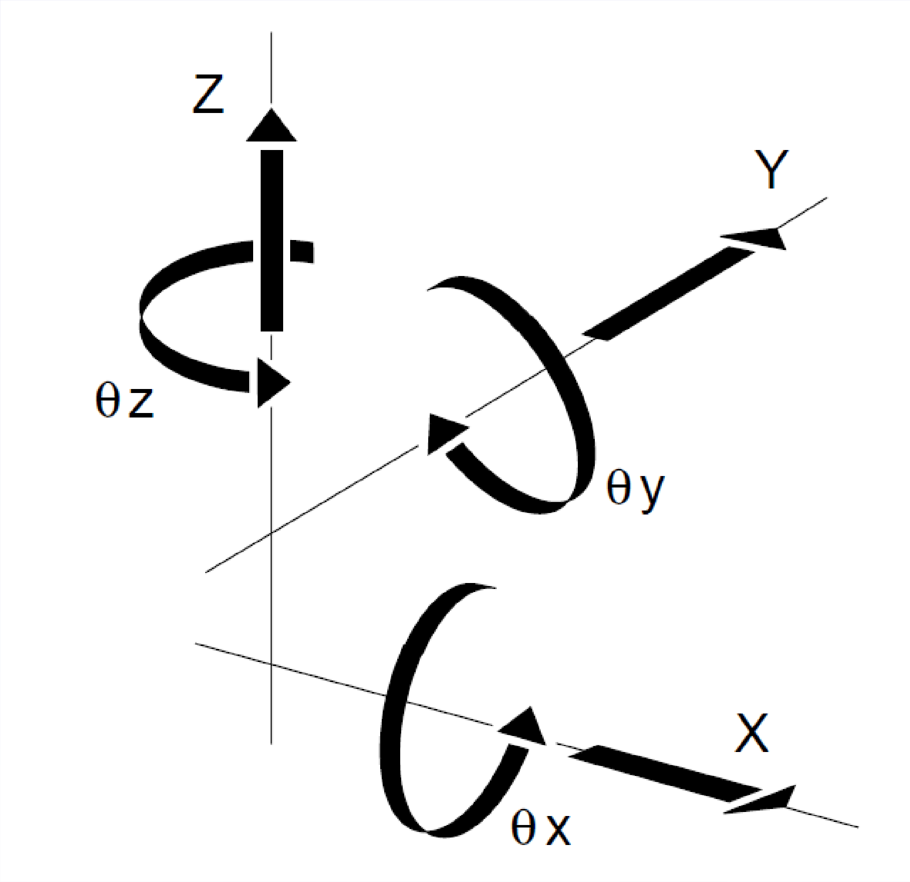
AM Machine

Subsystems – Modular design



Motion system

- Objective



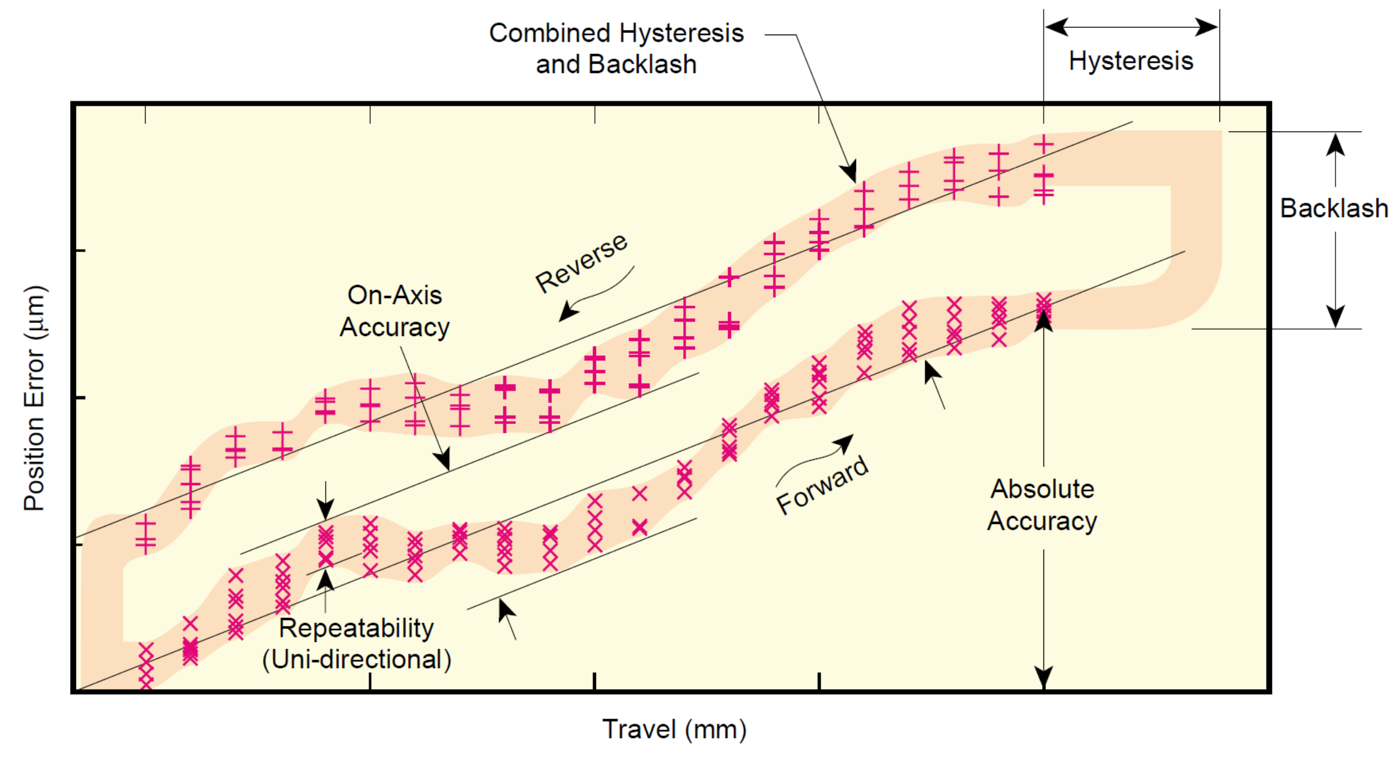
Six degrees of freedom for a motion system

- Position
- Velocity
- Acceleration

Motion system

Important concepts

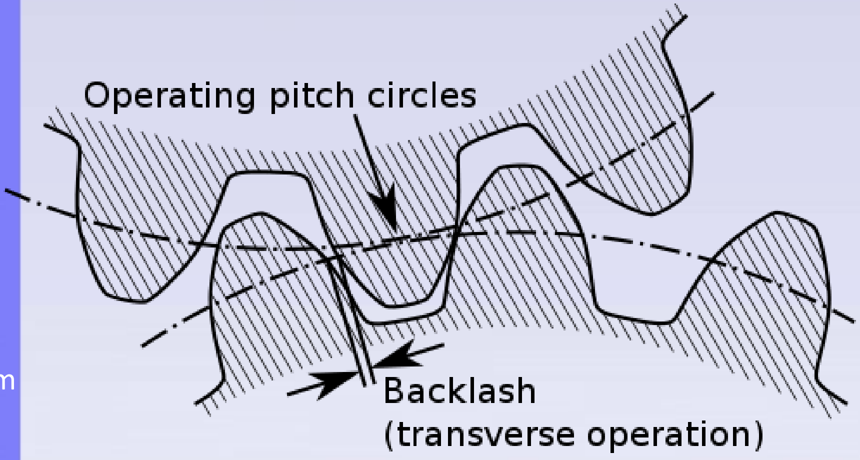
- **Resolution**: smallest increment that a motion system can detect
- **Minimum Increment Motion**: smallest motion than can be delivered
- **Accuracy**: maximum expected difference between ideal and actual

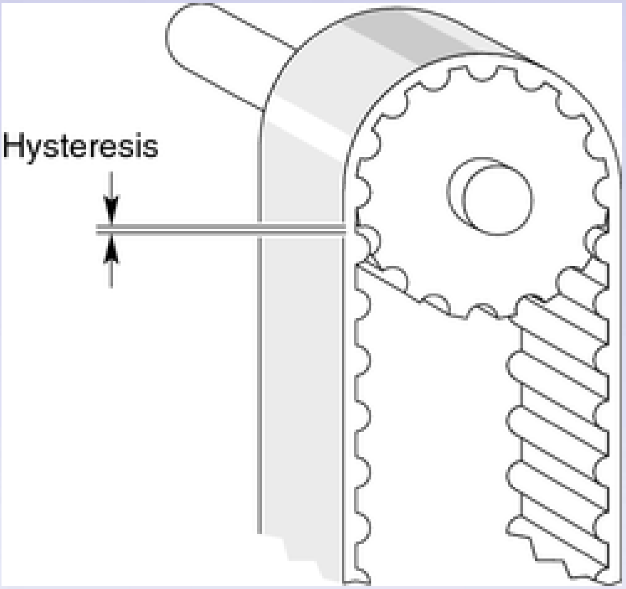


- **Absolute accuracy**;
- **On-Axis accuracy**;
- **Repeatability**;

Motion system

Important concepts





Backlash:

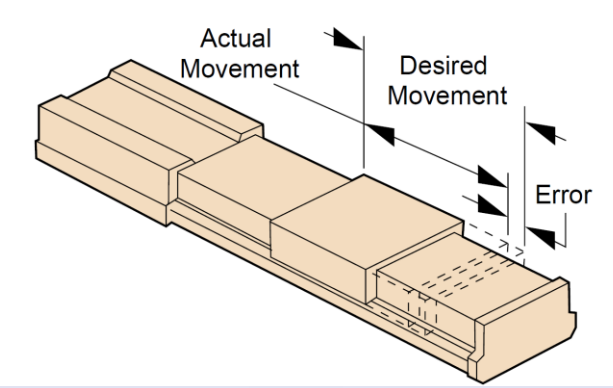
- The maximum magnitude of an input that produces no measurable output upon reversing direction.
- Caused by insufficient preloads or gaps between components.
- Relatively repeatable and can often be compensated

Hysteresis:

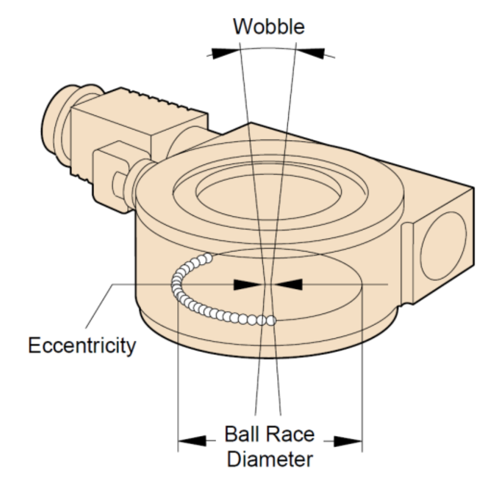
- The difference in the absolute position of an object for a given commanded input upon reversing direction.
- Caused by accumulated elastic forces.

Motion system

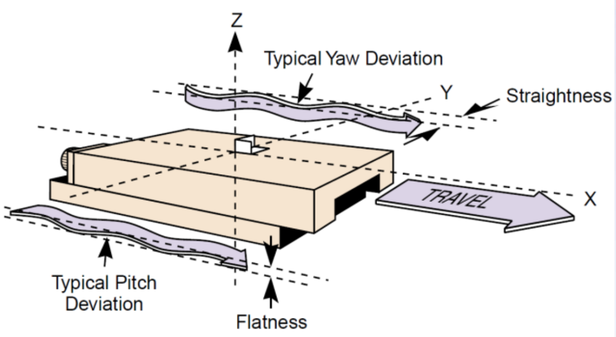
Important concepts



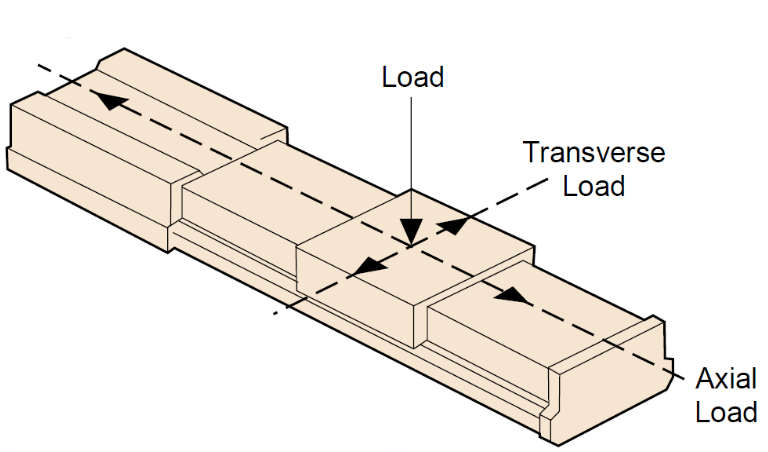
Linear Error



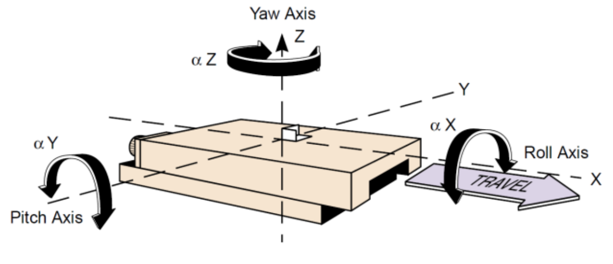
Eccentricity and Wobble



Runout: off-axis error



Load capacity: static and dynamic



Tilt and Wobble

Pictures courtesy: Newport

Motion system

Mechanical stage design

Materials

Material properties

- Stiffness
- Thermal expansion
- Thermal conductivity
- Material instability (creep)

Common stage materials

- Aluminum: light
- Steel: strong
- Brass: resistance to creep
- Granite: hard, no internal stress

Bearings – reduce friction

Ball bearing: point contact

Crossed roller bearing: line contact
Higher load capacity, stiffness, cost

Motion system

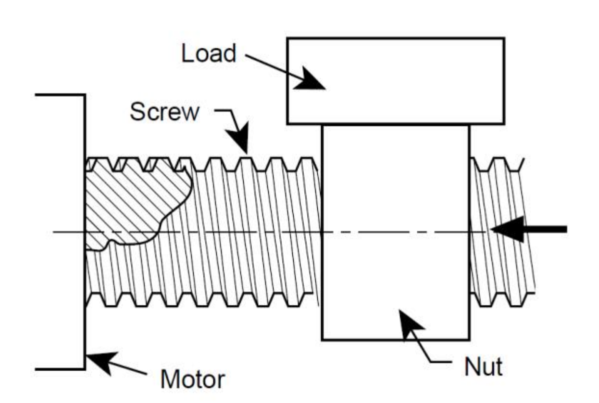
AM Machine

Motion System

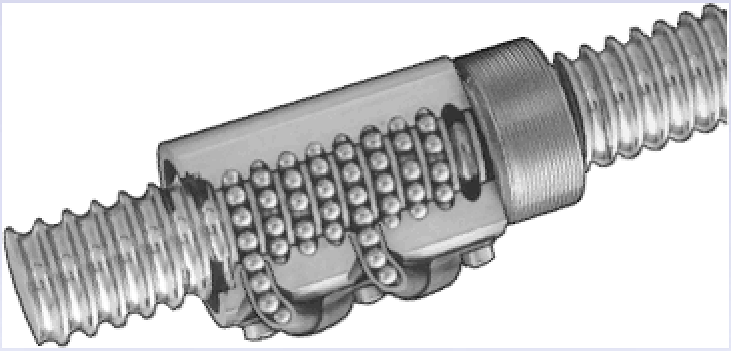
Electronic System

Mechanical stage design

- Drive systems – converts motor motion to desired motion

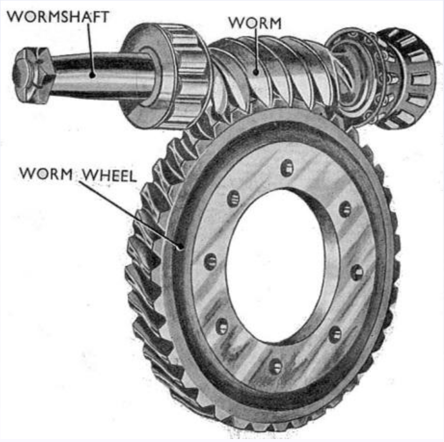


http://justinketterer.com/2010/05/12/mechatronics-the-major-subsystems/

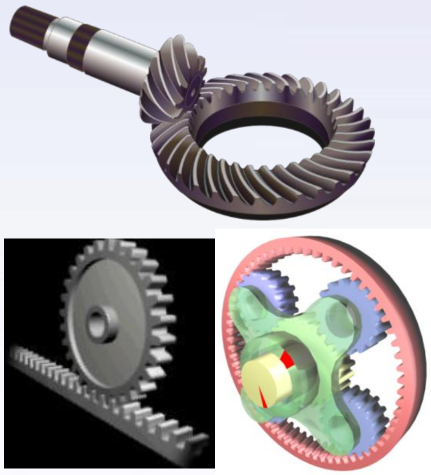


Torque ratio
Velocity ratio

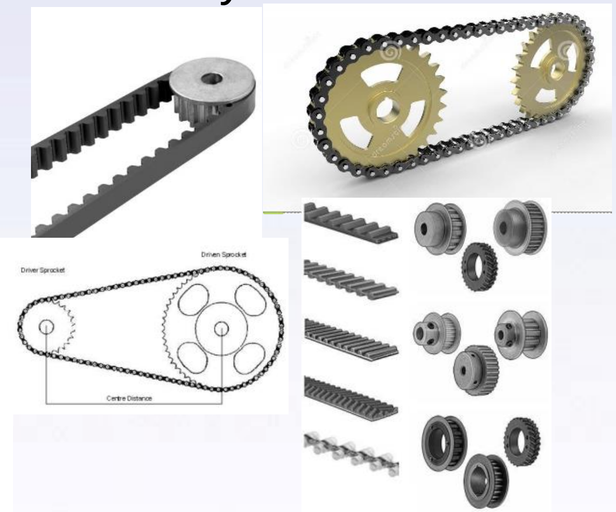
Lead screw: low cost, self-lock Ball screw: higher efficiency and cost



Worm drive: rotation to rotation, high velocity ratio



Gearbox: transmit motion



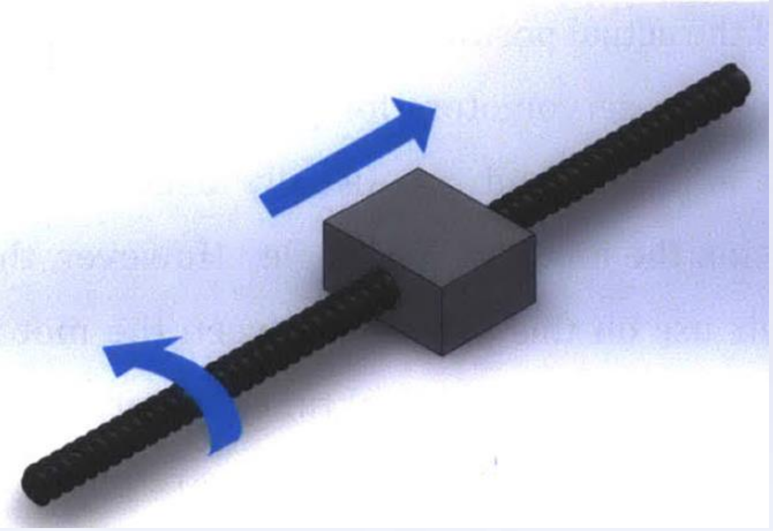
Belt, chain, sprocket drive

Motion system

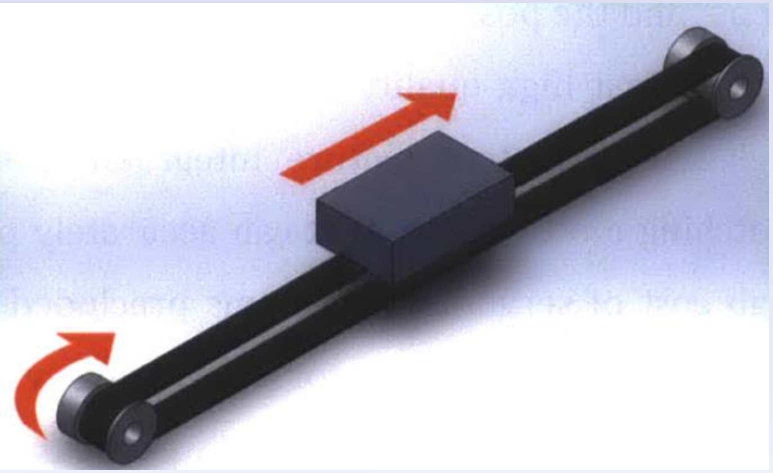
Mechanical stage design

- **Drive systems – converts motor motion to desired motion**

Two common drive systems for low cost 3D printers



Screw drive: more stiff, slower, more accurate



Belt drive: less stiff and accurate (backlash and hysteresis, but much faster. Need to be properly tensioned for better performance

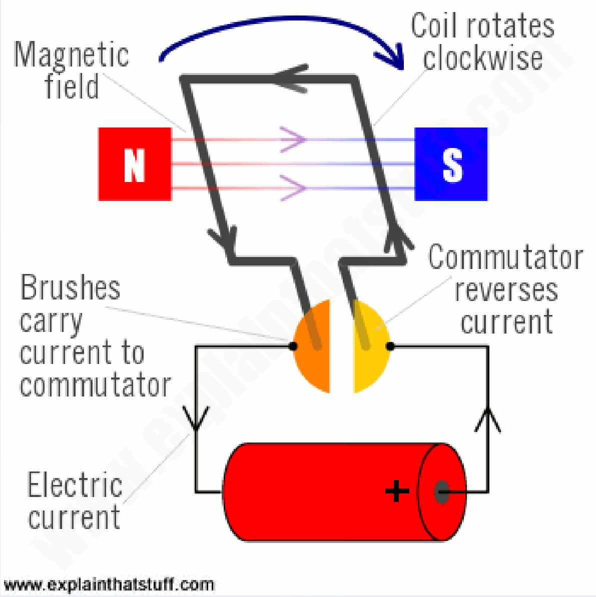
Motion system

AM Machine

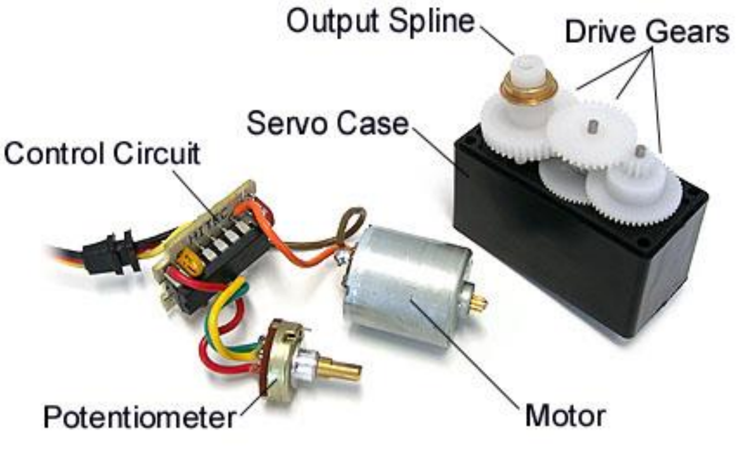
Motion System

Electronic System

Motors – produce motion

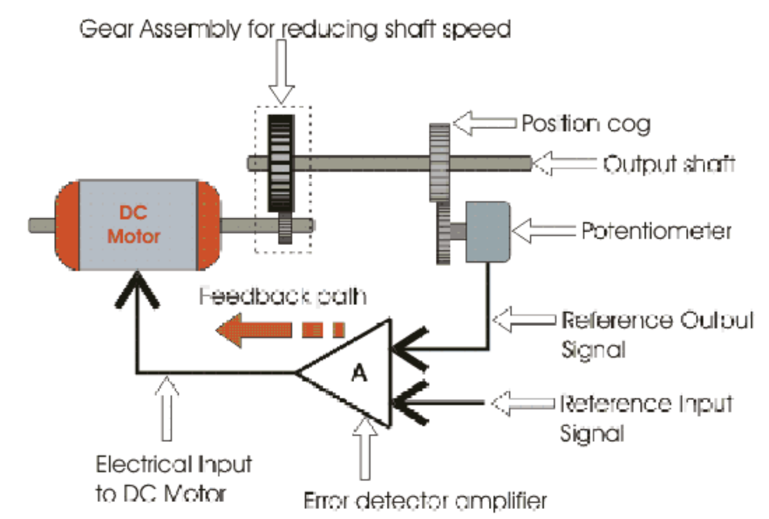


DC motor: continuous motion



Servo motor: high torque, slow, with gears

A DC motor, can be commonly found in toys, apply DC voltage, rotate fast



Feedback control with positiometer

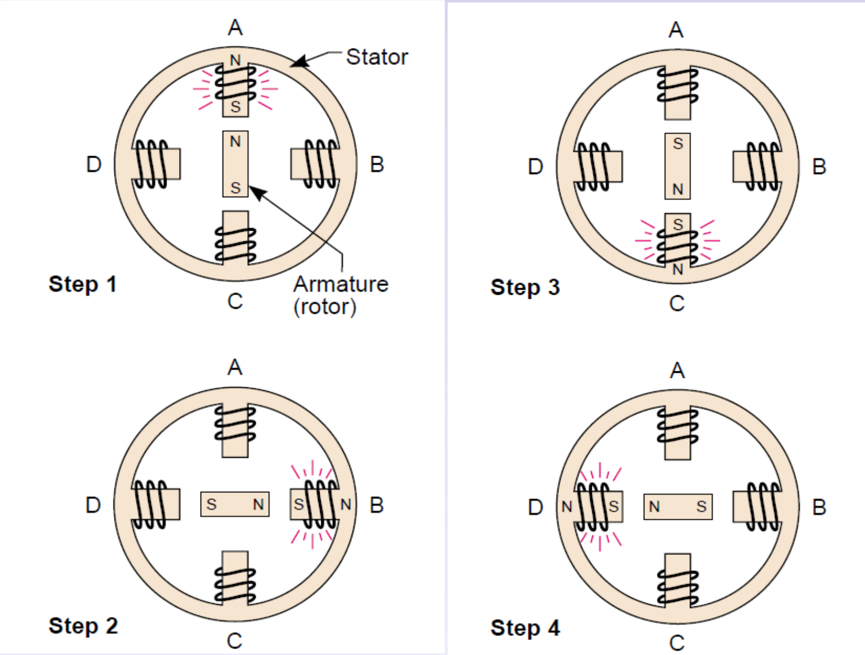
Motion system

AM Machine

Motion System

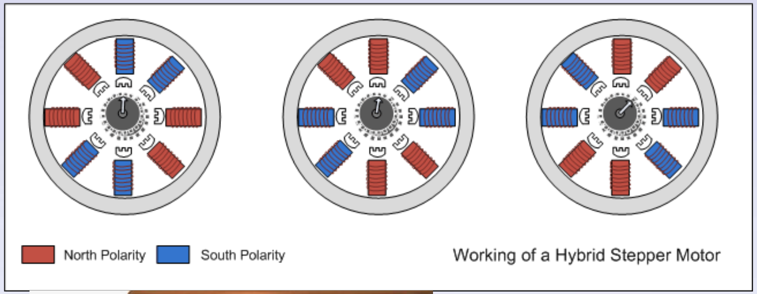
Electronic System

Motors – produce motion

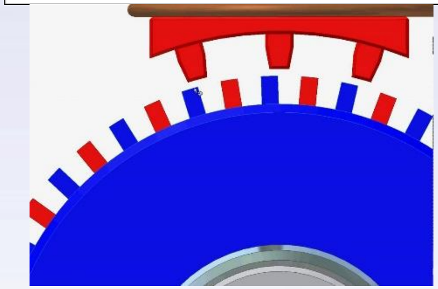


Permanent magnet (PM) stepper motor: stepped motion

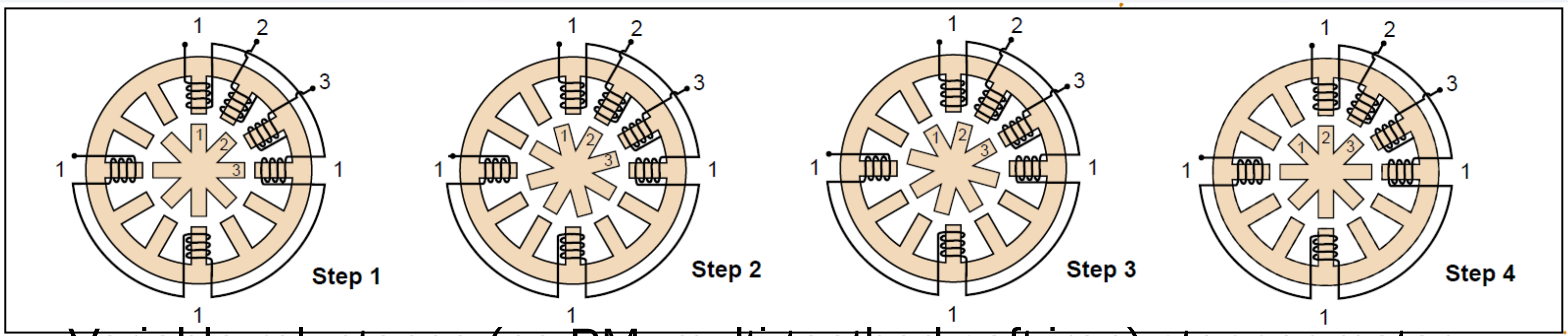
Stepper motor



Working of a Hybrid Stepper Motor



Hybrid stepper motor: highest performance



Variable reluctance (no PM, multi-toothed soft iron) stepper motor

Motion system

Motors – produce motion

	DC Motor	Servo Motor	Stepper Motor
Motion and torque	Very fast, continuous rotation, low torque	Fast, high torque, accurate position feedback control	Very high torque at low speed, stepped motion
Control	Very easy, connect with DC voltage	Difficult to set up with PWM tuning	Easy, counting steps for position control
Application	Car wheels, fans, etc.	robotic arm, rudder control	Position control apps, 3D printers
Pros	Very cheap, very easy to find and use	Very capable and high power, high torque and speed	Accurate position control, easy to find and use, high torque
Cons	Low torque, open loop, no position control	Expensive, difficult to find and use	Low torque at high speed, may skip steps at high load, may vibrate & resonate

Motion system

AM Machine

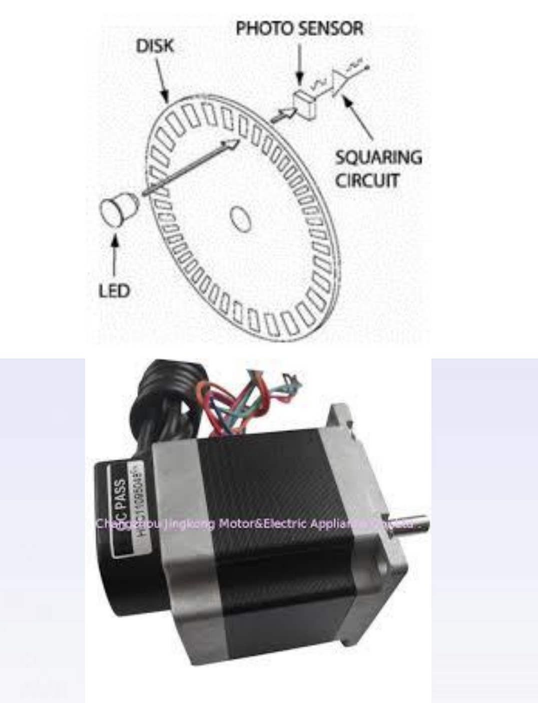
Motion System

Electronic System

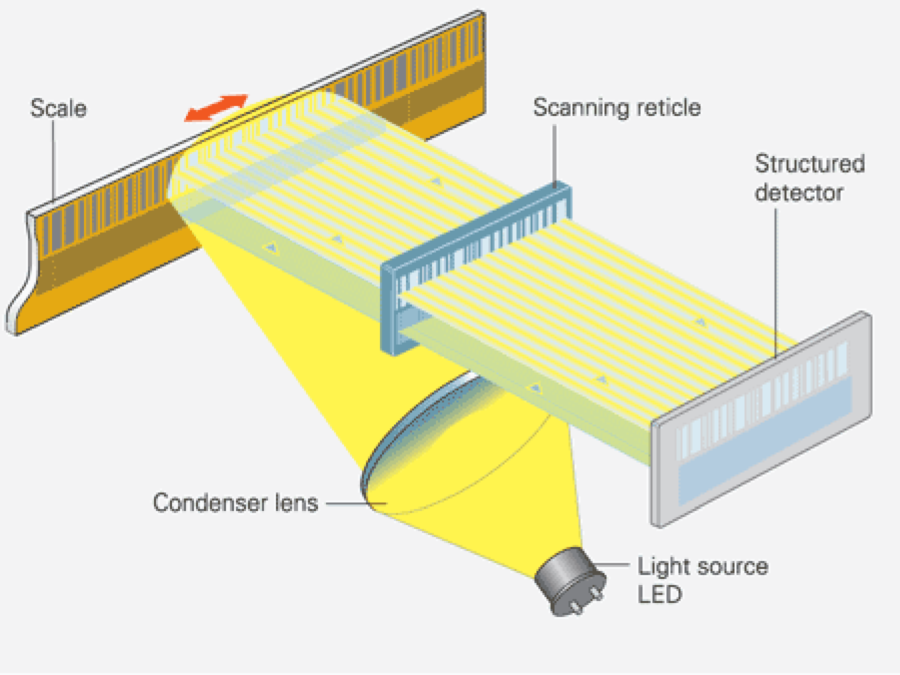
Motion feedback control

Principle: direct measurement BETTER than indirect measurement

- **Encoders for position**



Optical rotary encoder



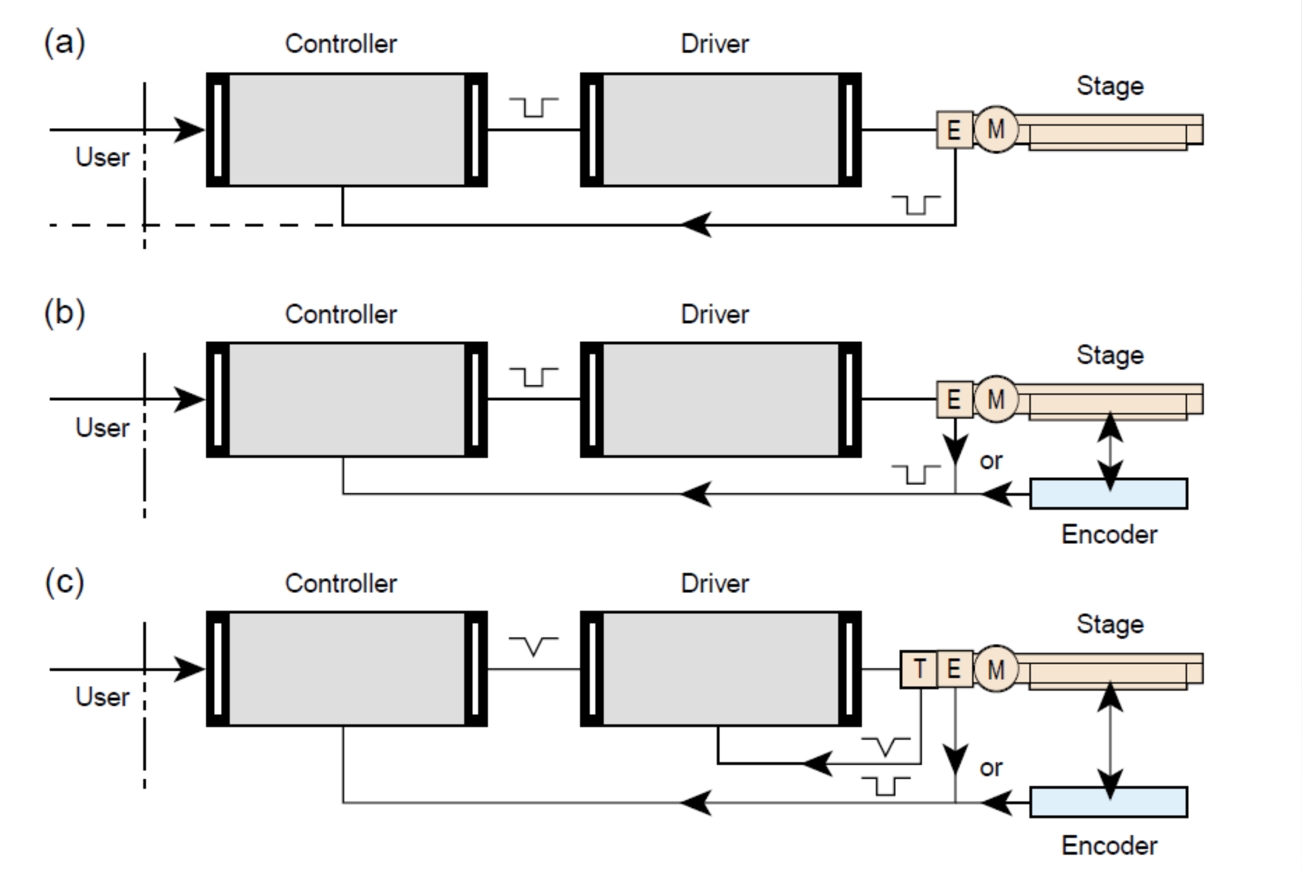
Optical linear encoder

- **Optical VS magnetic: environment**
- **Incremental VS absolute: cost, restart**

Motion system

Motion feedback control

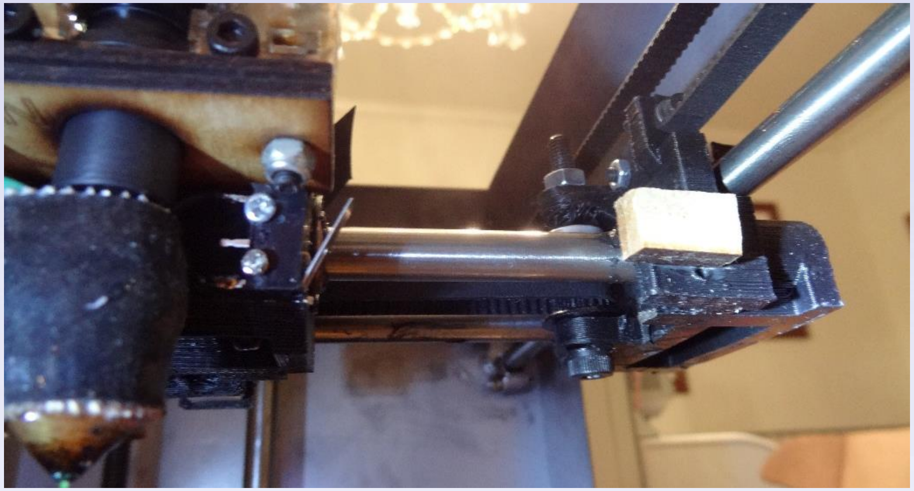
Tachometer for velocity



(a). Indirect rotary encoder; (b). Direct linear encoder; (c). Direct linear encoder for position and tachometer for velocity

Motion system

- **Motion feedback control**
  - **Limit switches**

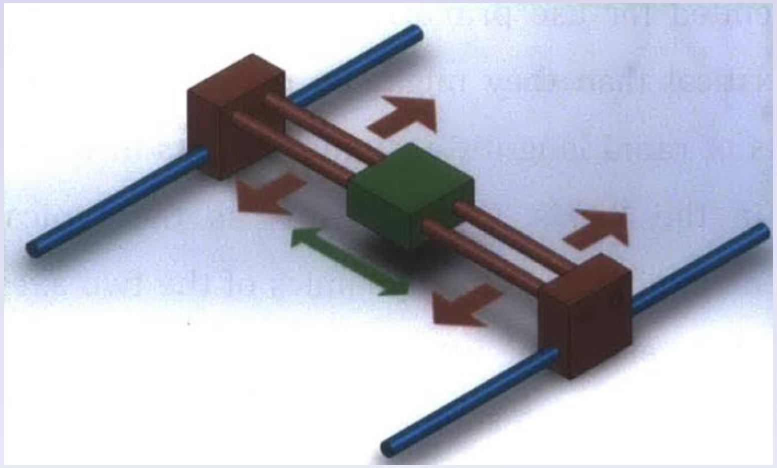


Limit Switches (can be mechanical or optical)

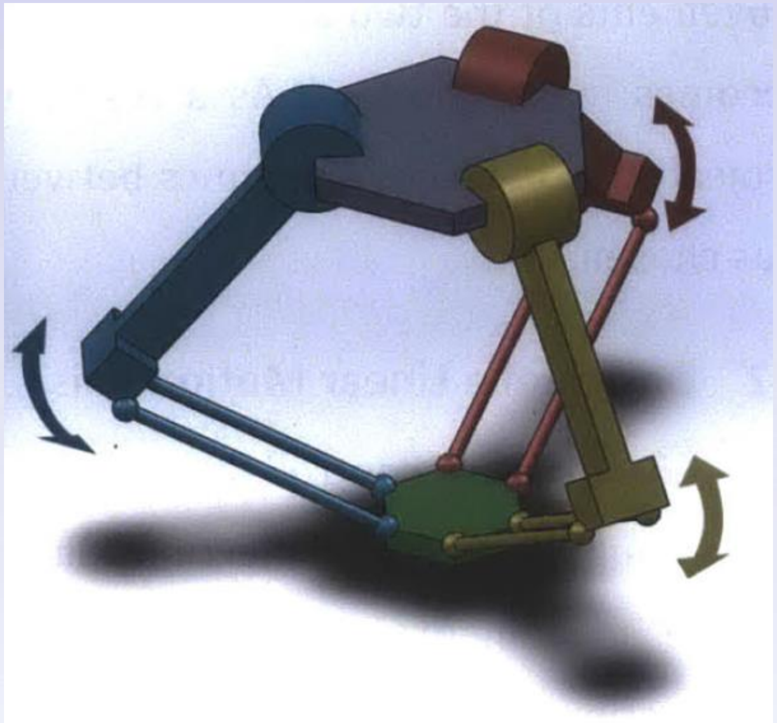
Motion system

Arrangement of axes

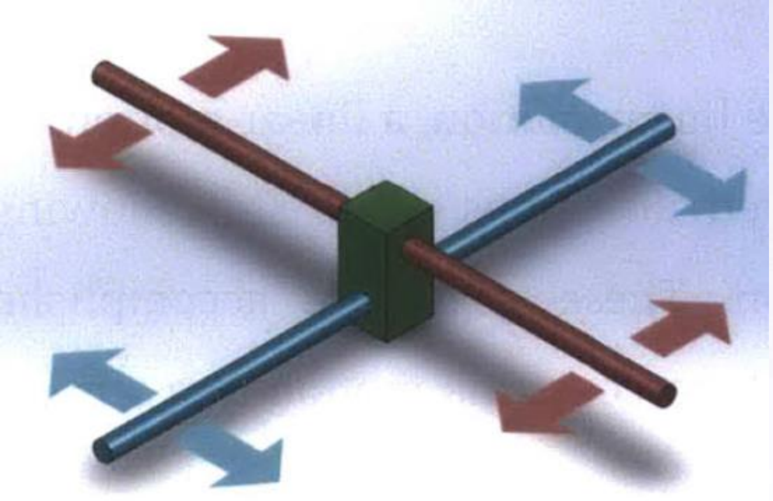
Pictures credit: Justin Lan@MIT



Serial axes arrangement



Delta robot configuration



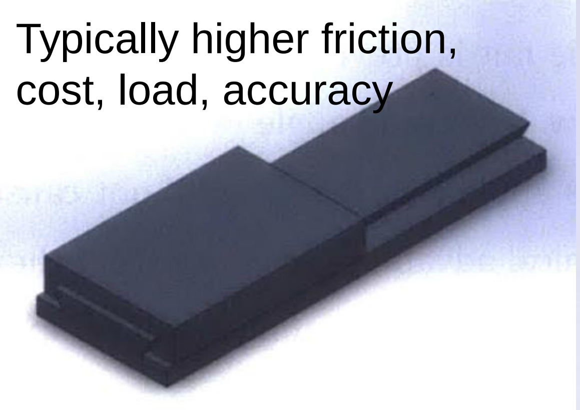
Crossed axes arrangement

Factors

- **Moving mass (load)**
- **Space**
- **Motion symmetry**

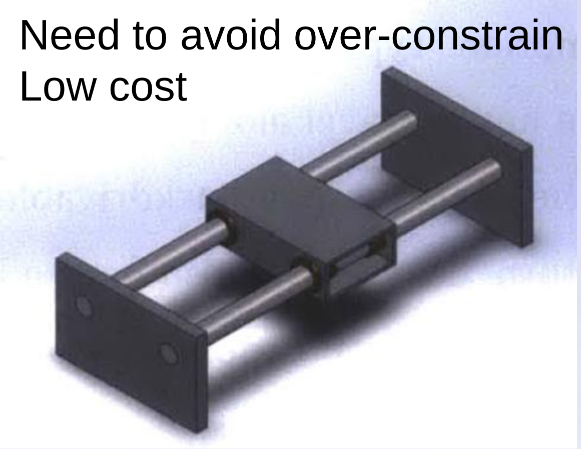
Motion system

Linear guides – need to constrain 5 DOFs



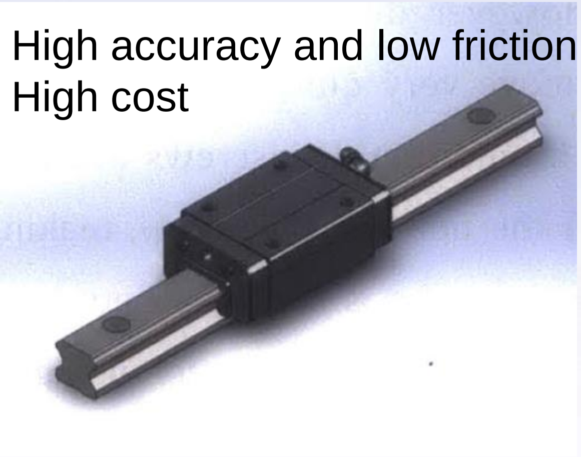
Typically higher friction, cost, load, accuracy

Dovetail rail



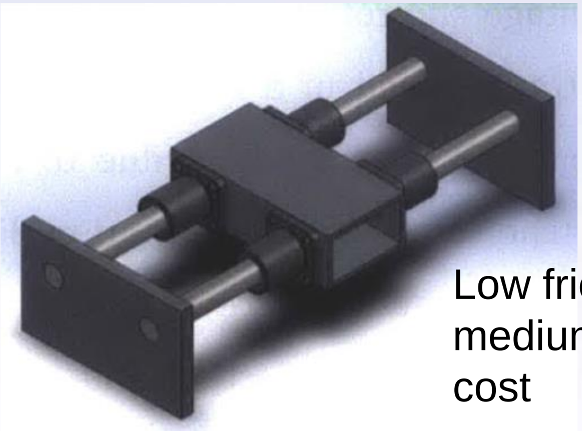
Need to avoid over-constrain
Low cost

Twin round rail with bushing



High accuracy and low friction
High cost

Dovetail with bearings



Low friction
medium load and
cost

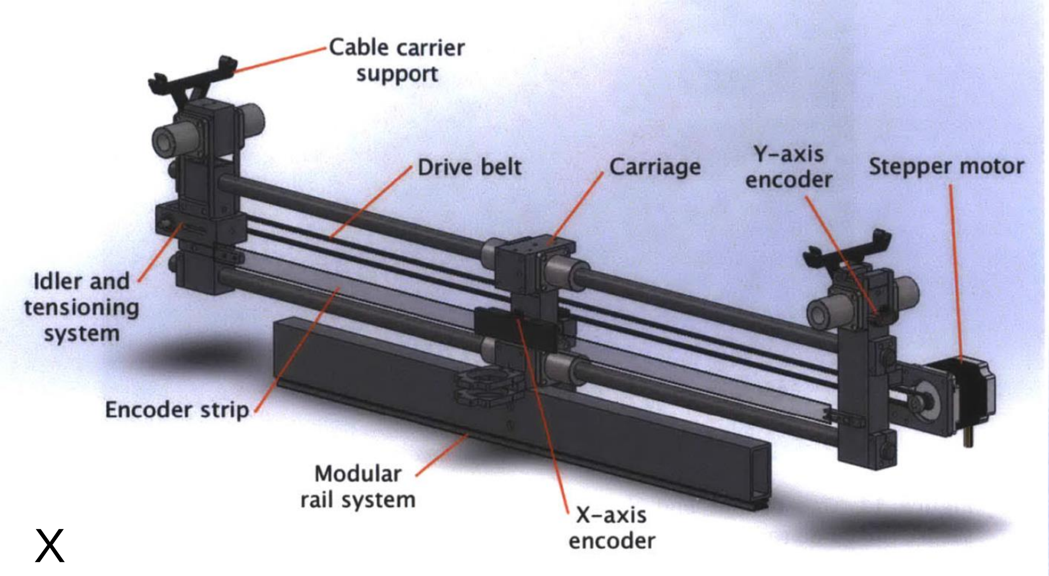
Twin-rail with ball bearings

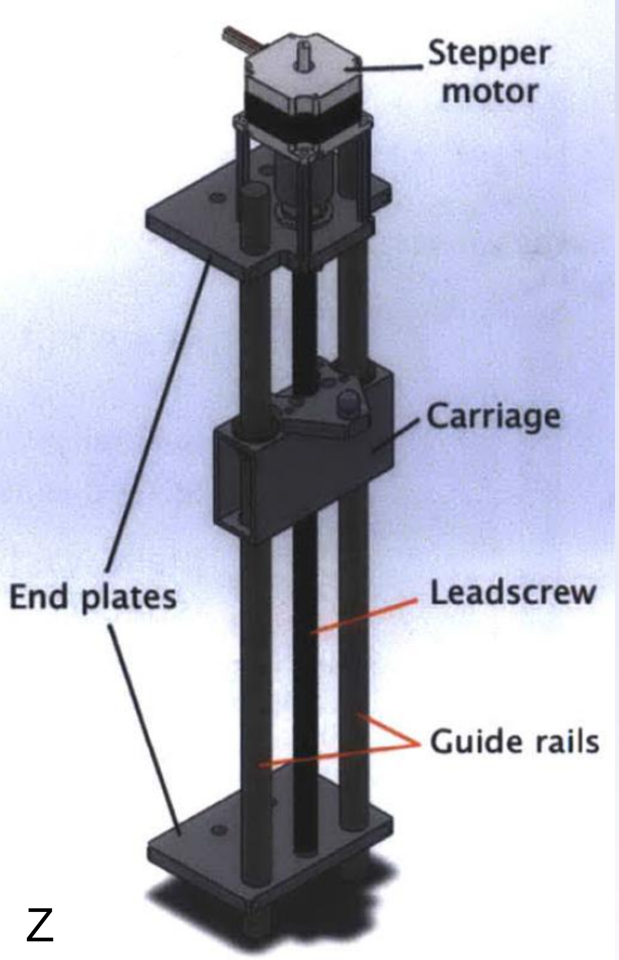
- **Friction (can use ball or roller)**
- **Cost (difficulty of mating and manufacturing)**
- **Load and accuracy**

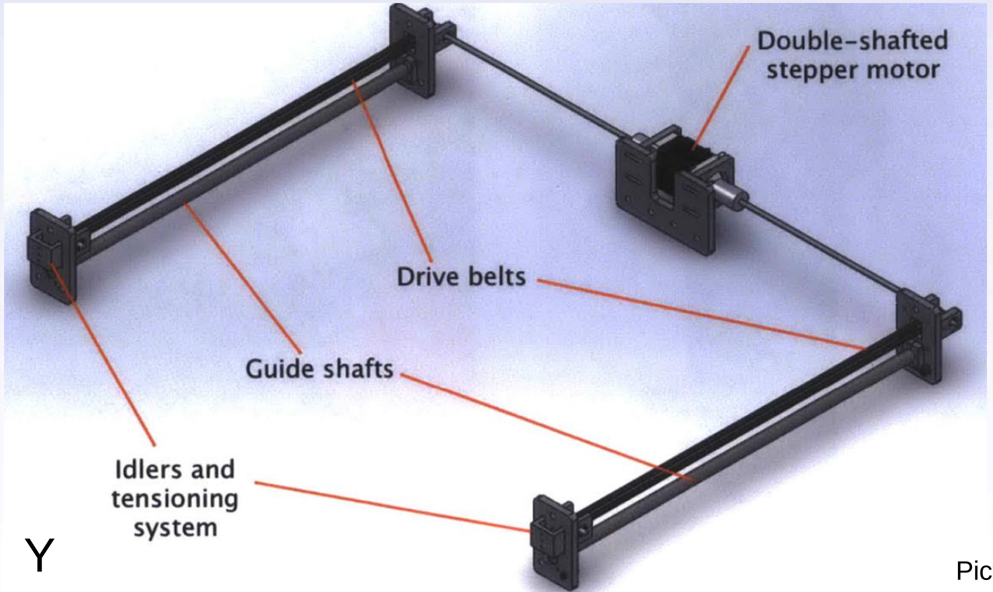
Pictures credit: Justin Lan@MIT

Motion system

Example design of XYZ axes



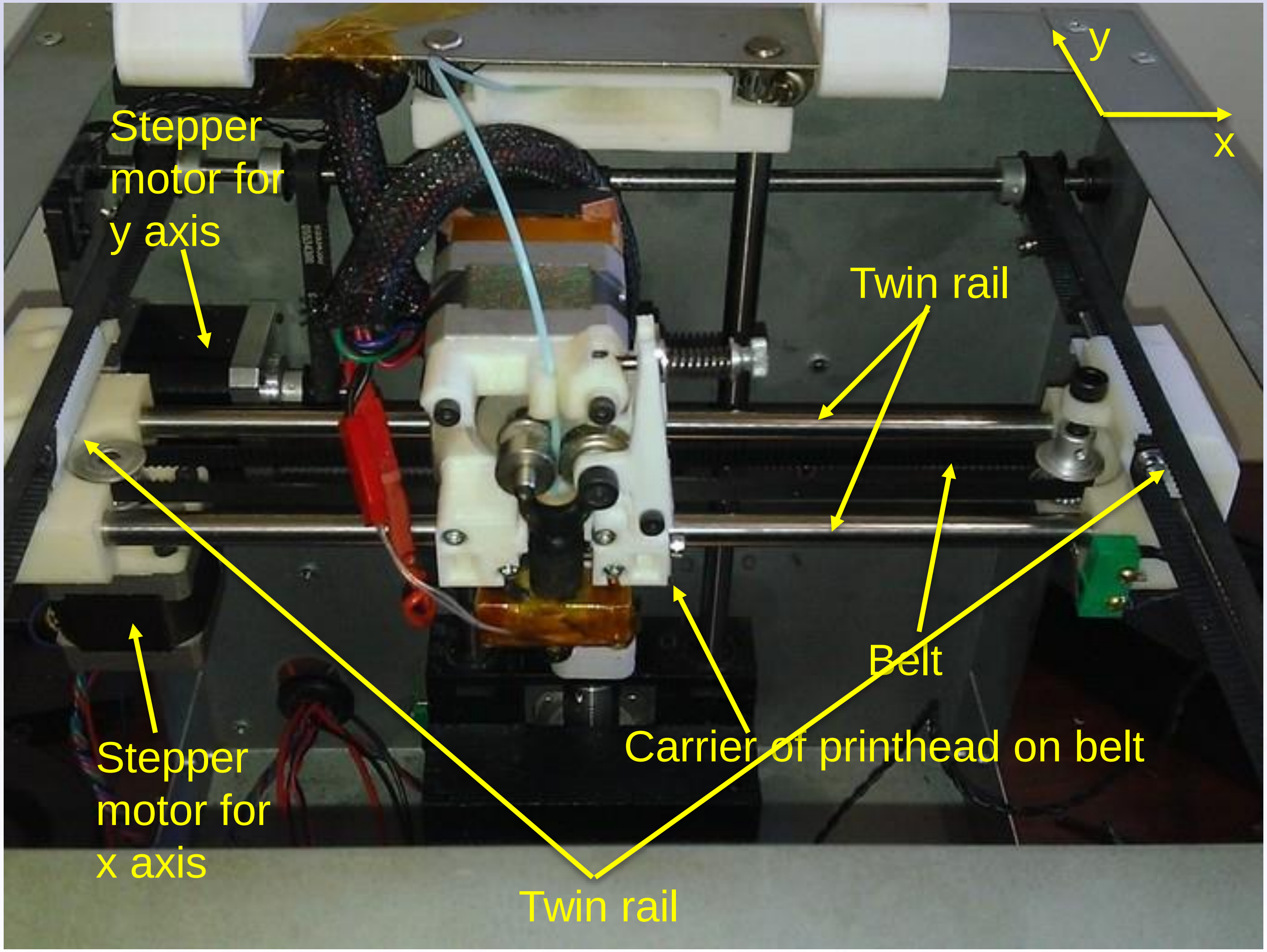




Pictures credit: Justin Lan@MIT

Motion system

Example of mechanical stage in Solidoodle

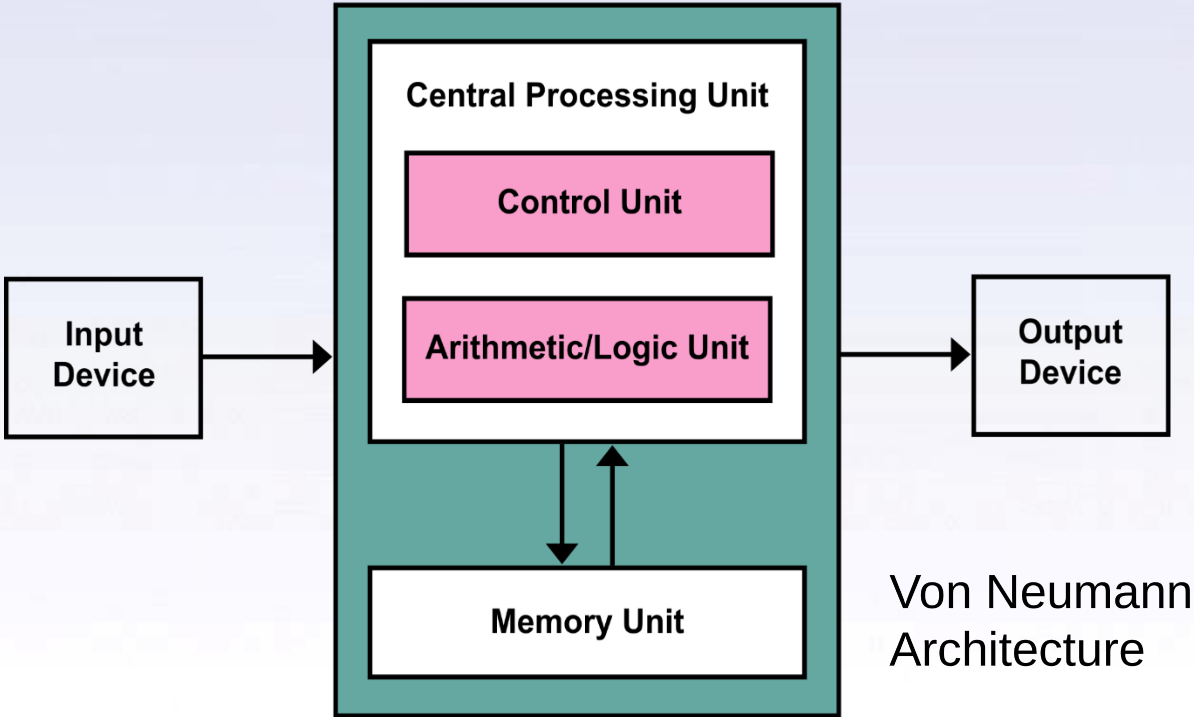


UNIVERSITY OF ARKANSAS

Electronic system

AM³ Lab
Advanced Manufacturing | Modeling | Materials

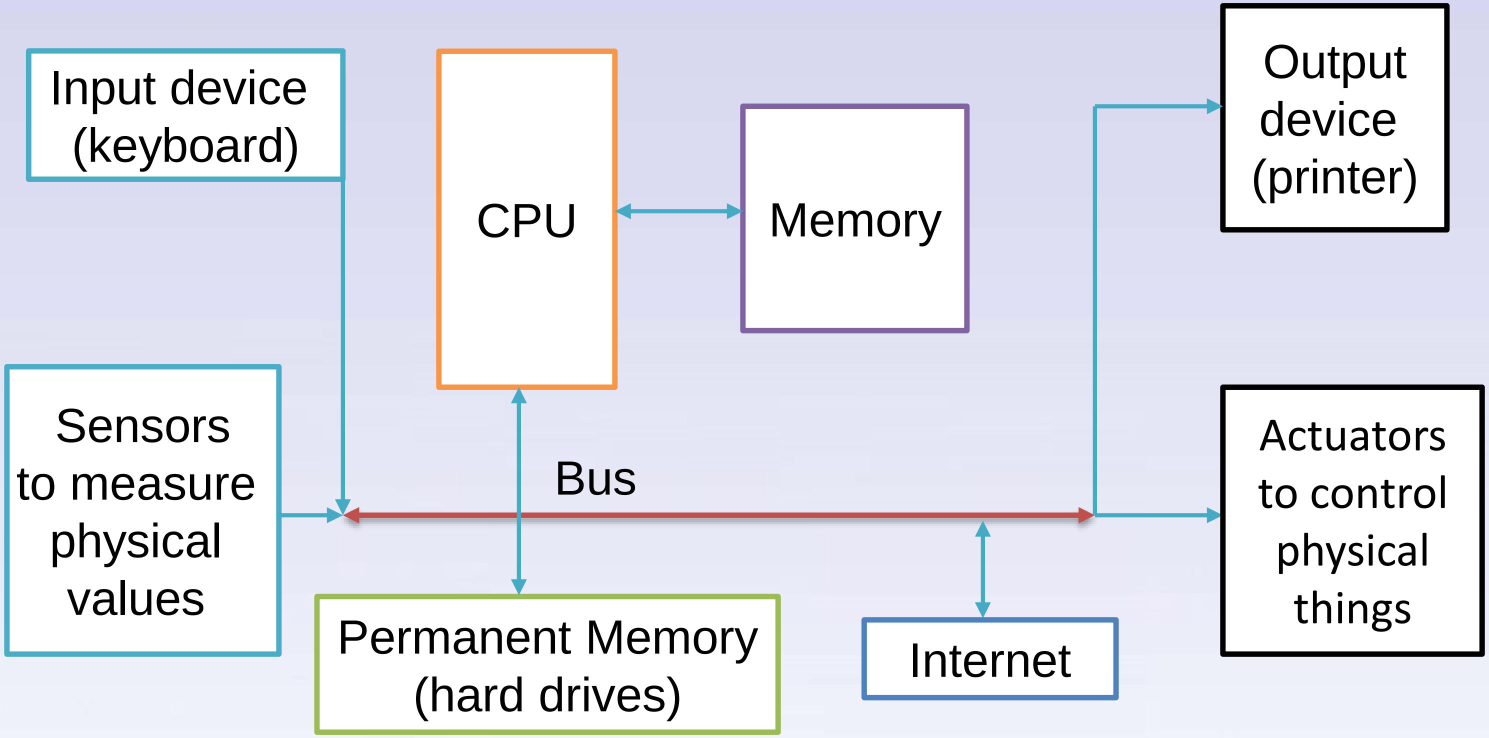
- Digital fabrication → Digital control



Von Neumann Architecture

Electronic system

- Digital fabrication → Digital control

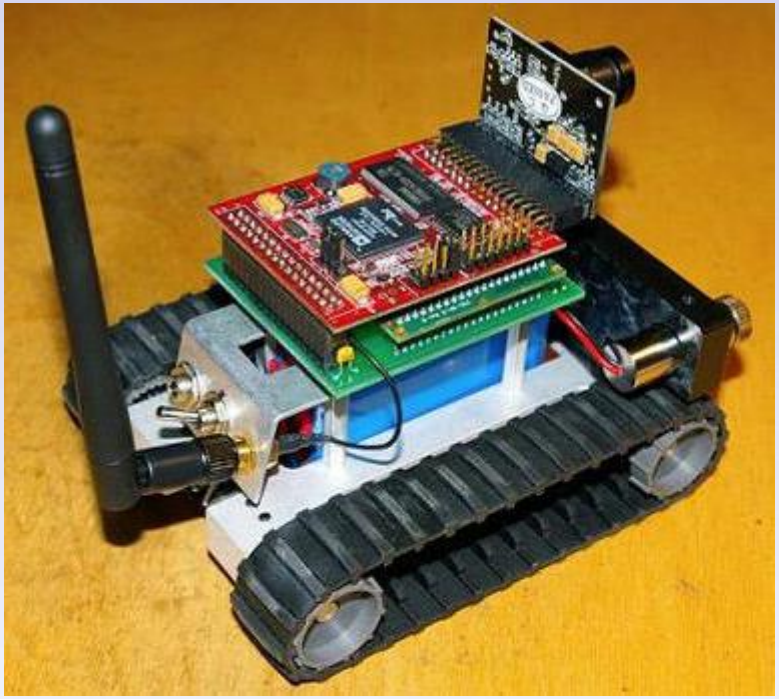

Input device (keyboard)
CPU
Memory
Output device (printer)
Sensors to measure physical values
Bus
Actuators to control physical things
Permanent Memory (hard drives)
Internet

One problem

- CPU and the physical world don't speak the same language (digital VS analogue; discrete VS continuous)

Electronic system

- Digital fabrication → Digital control



Computers are too big and heavy

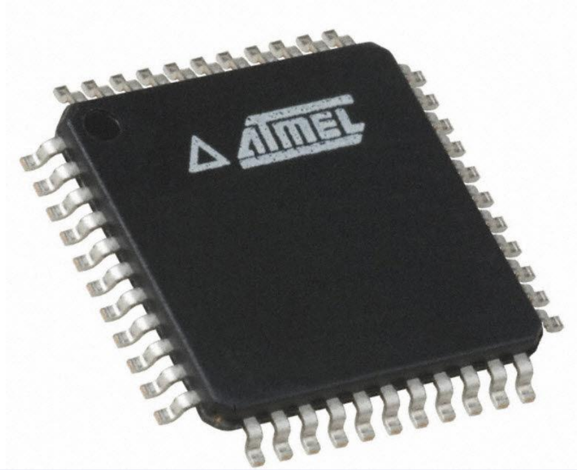
Not enough power from computer

More problems

- Mobility
- Insufficient power output

Electronic system

Solution – Embedded system



Microcontroller: a small computer on a single integrated circuit containing a processor core, memory, and programmable I/O peripherals, and even DACs for some

ARM Cortex-M cores
Atmel AVR (8-bit), AVR32 (32-bit), and AT91SAM (32-bit)
Intel 8051
STMicroelectronics STM8 (8-bit), ST10 (16-bit) and STM32 (32-bit)
Texas Instruments TI MSP430 (16-bit) C2000 (32-bit)
Toshiba TLCS-870 (8-bit/16-bit).

More see: http://en.wikipedia.org/wiki/List_of_common_microcontrollers

- **Solve the mobility issue by high integration**
- **For some with DAC, solve the D to A issue**

Electronic system

AM Machine

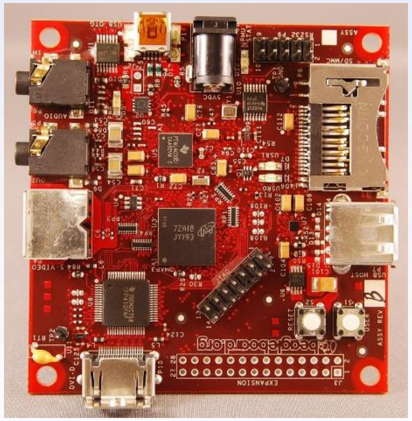
Motion System

Electronic System

- **Solution – Embedded system**
  - **Still need circuit board to connect – single board computer**

Arduino board (~10s MHz): No OS, but very easy, huge user base

Raspberry Pi (~100s MHz): more powerful, can run Linux, more complex

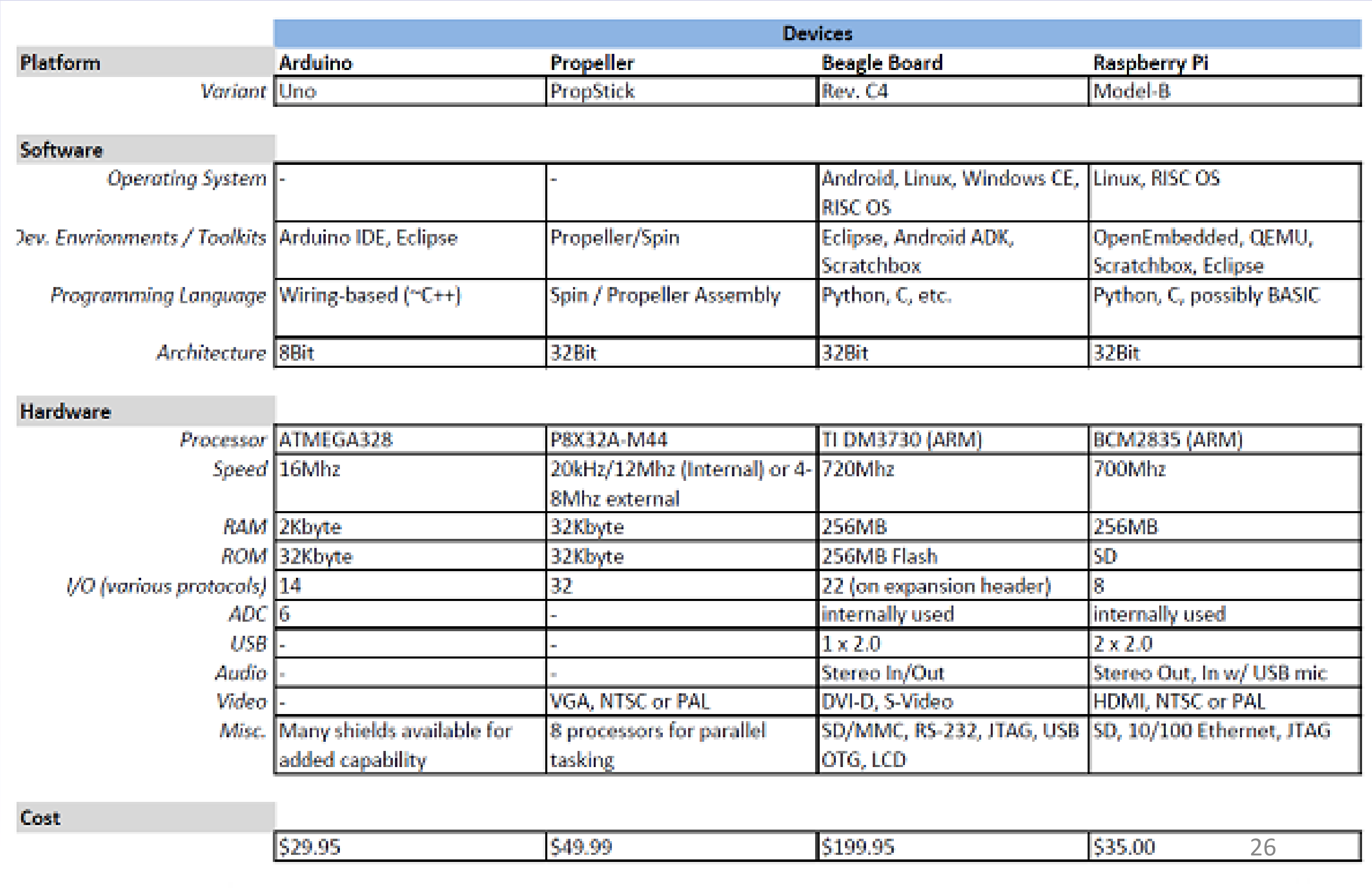


BeagleBoard(~10s MHz): by TI

And many others. Comparison see: http://en.wikipedia.org/wiki/Comparison_of_single-board_computers

Electronic system

- Solution – Embedded system
  - Still need circuit board to connect – single board computer



	Devices			
Platform	**Arduino**	**Propeller**	**Beagle Board**	**Raspberry Pi**
Variant	Uno	PropStick	Rev. C4	Model-B
Software				
Operating System	-	-	Android, Linux, Windows CE, RISC OS	Linux, RISC OS
Dev. Envrionments / Toolkits	Arduino IDE, Eclipse	Propeller/Spin	Eclipse, Android ADK, Scratchbox	OpenEmbedded, QEMU, Scratchbox, Eclipse
Programming Language	Wiring-based (~C++)	Spin / Propeller Assembly	Python, C, etc.	Python, C, possibly BASIC
Architecture	8Bit	32Bit	32Bit	32Bit
Hardware				
Processor	ATMEGA328	P8X32A-M44	TI DM3730 (ARM)	BCM2835 (ARM)
Speed	16Mhz	20kHz/12Mhz (Internal) or 4-8Mhz external	720Mhz	700Mhz
RAM	2Kbyte	32Kbyte	256MB	256MB
ROM	32Kbyte	32Kbyte	256MB Flash	SD
I/O (various protocols)	14	32	22 (on expansion header)	8
ADC	6	-	internally used	internally used
USB	-	-	1 x 2.0	2 x 2.0
Audio	-	-	Stereo In/Out	Stereo Out, In w/ USB mic
Video	-	VGA, NTSC or PAL	DVI-D, S-Video	HDMI, NTSC or PAL
Misc.	Many shields available for added capability	8 processors for parallel tasking	SD/MMC, RS-232, JTAG, USB OTG, LCD	SD, 10/100 Ethernet, JTAG
Cost				
	$29.95	$49.99	$199.95	$35.00

Electronic system

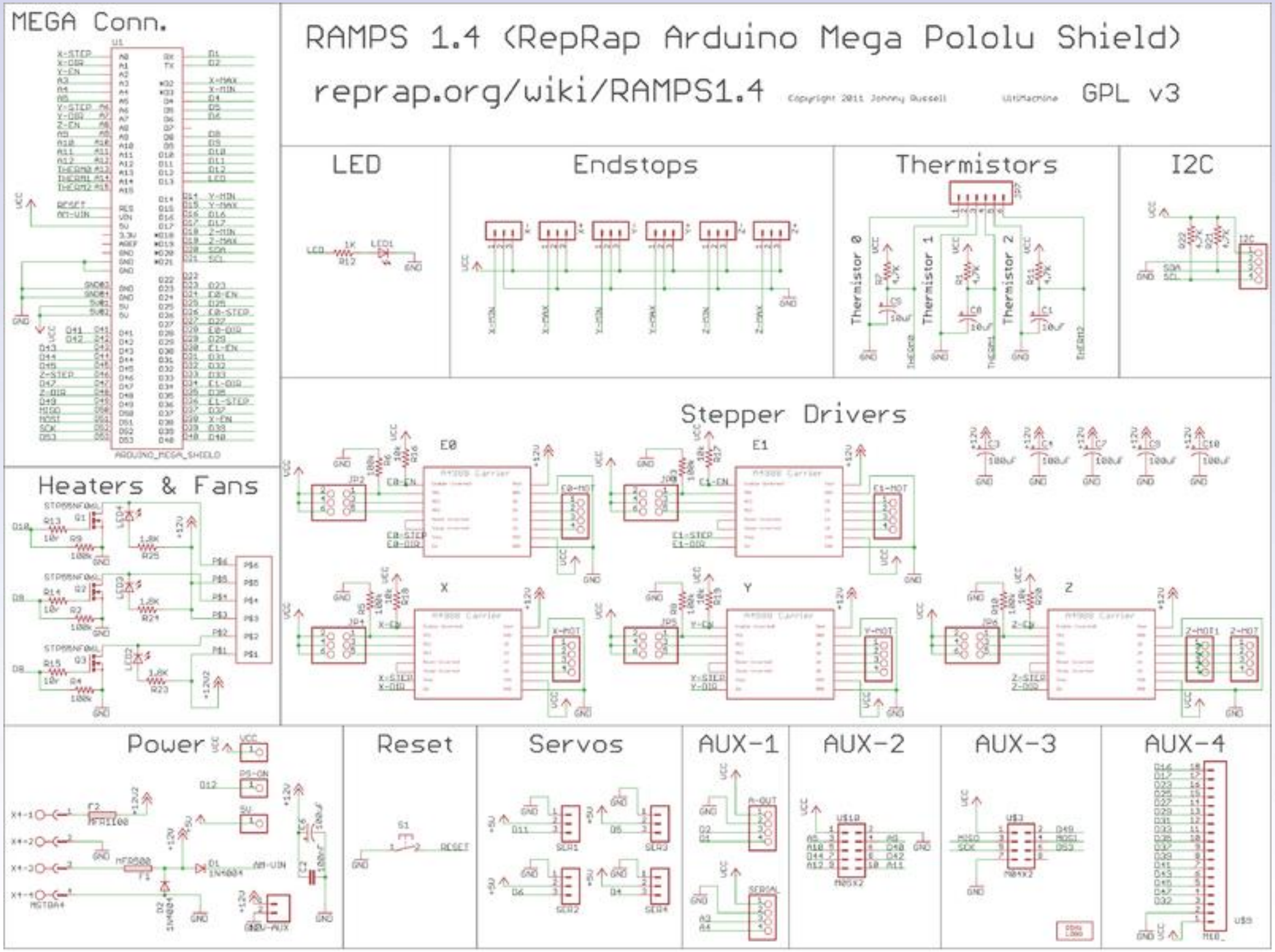
AM Machine

Motion System

Electronic System

Solution – Embedded system

- Extended capability – shields (e.g., RAMPs for RepRap 3D printer)



Electronic system

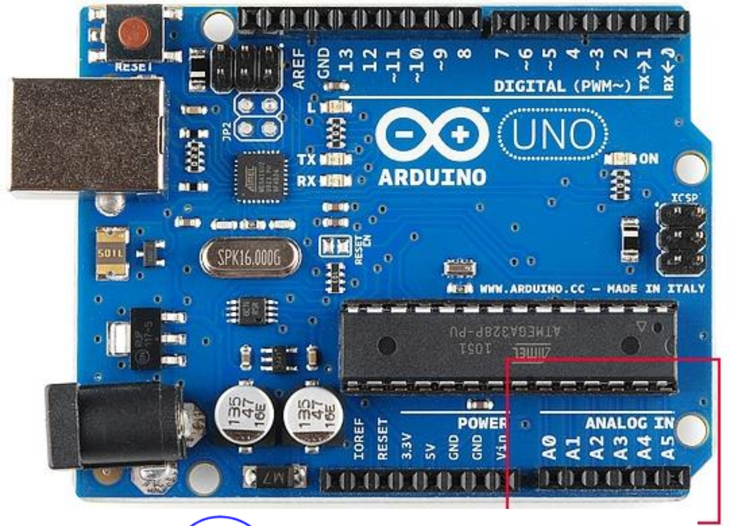
AM Machine

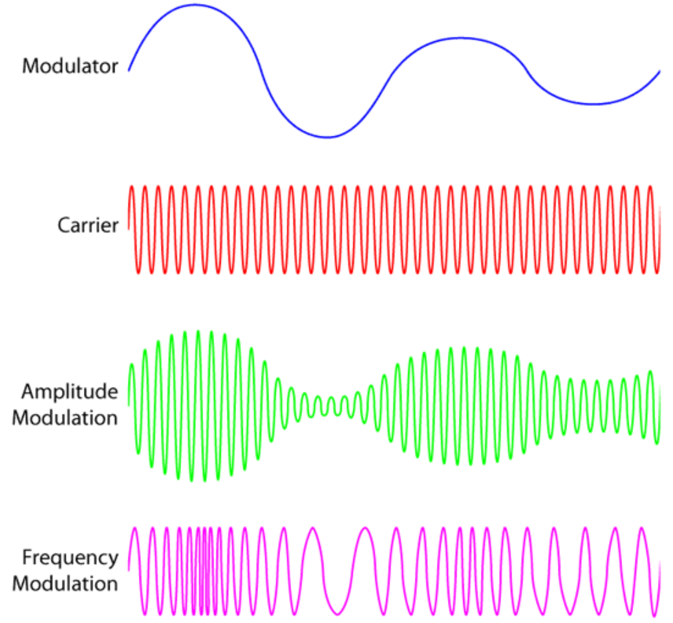
Motion System

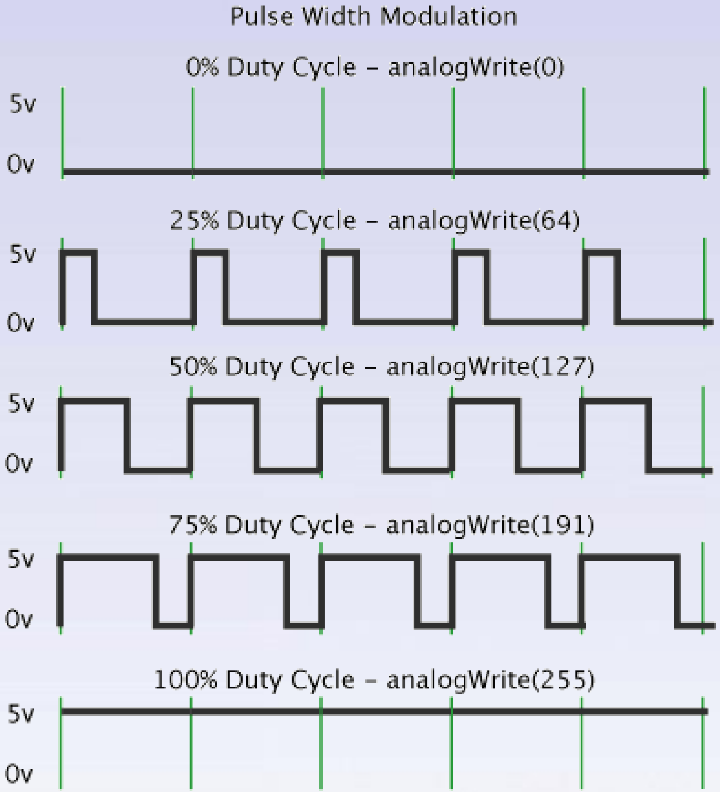
Electronic System

Solution – Embedded system

- Digital Analogue Converter (DACs)






Pulse Width Modulation
0% Duty Cycle – analogWrite(0)
25% Duty Cycle – analogWrite(64)
50% Duty Cycle – analogWrite(127)
75% Duty Cycle – analogWrite(191)
100% Duty Cycle – analogWrite(255)
5v
0v

Pulse Width Modulation (PWM) VS AM or FM

The Voltage/Operating Voltage (e.g., 5V) ratio is modulated to the percentage of ON time in a duty cycle

Electronic system

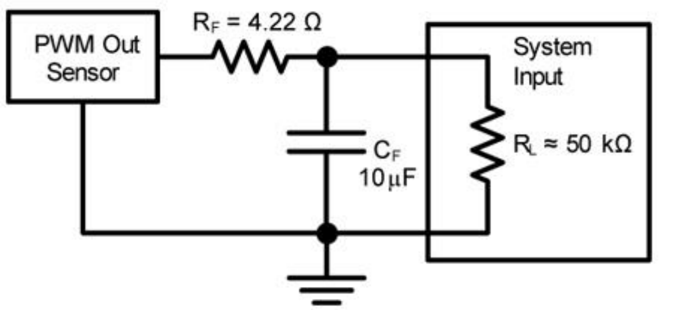
AM Machine

Motion System

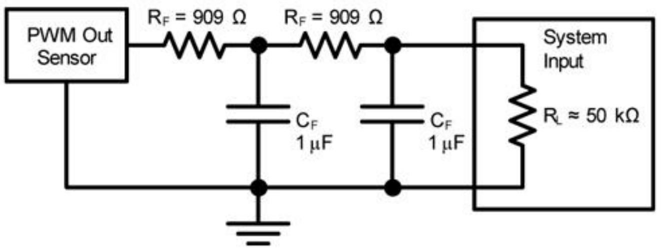
Electronic System

Solution – Embedded system

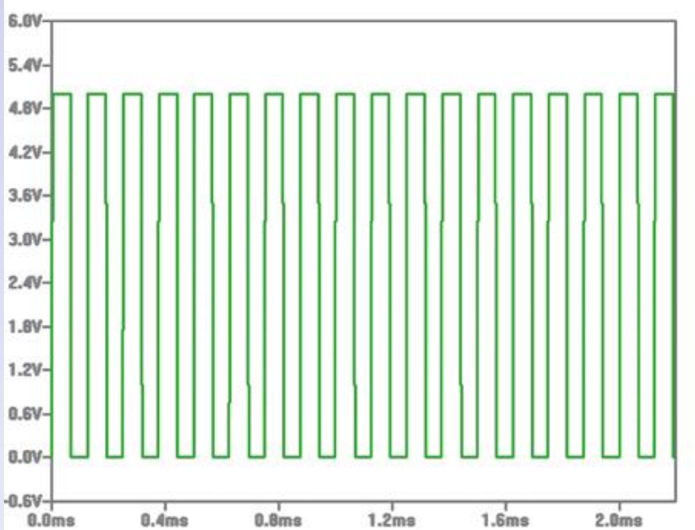
- Digital Analogue Converter (DACs)



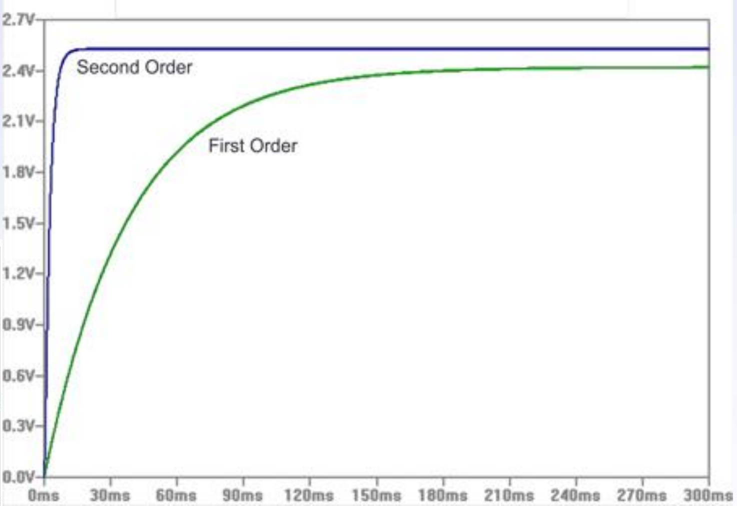
First order low pass filter



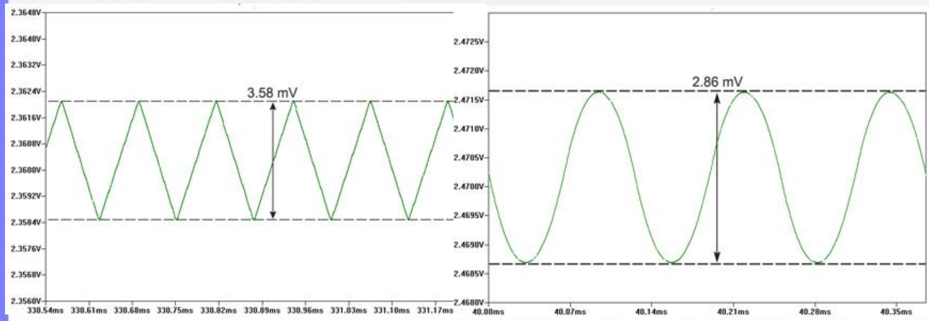
Second order low pass filter



PWM signal



Constant voltage analog signal



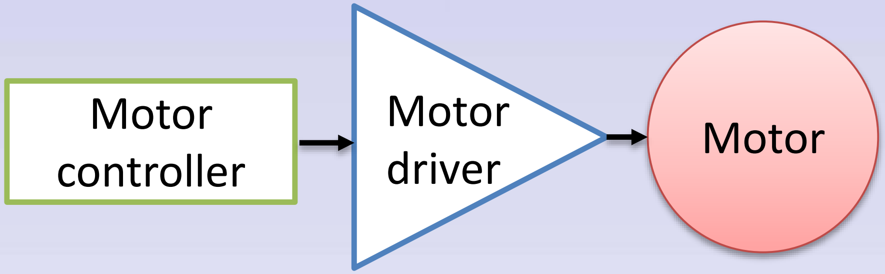
Ripple comparison between 1st and 2nd order LPF

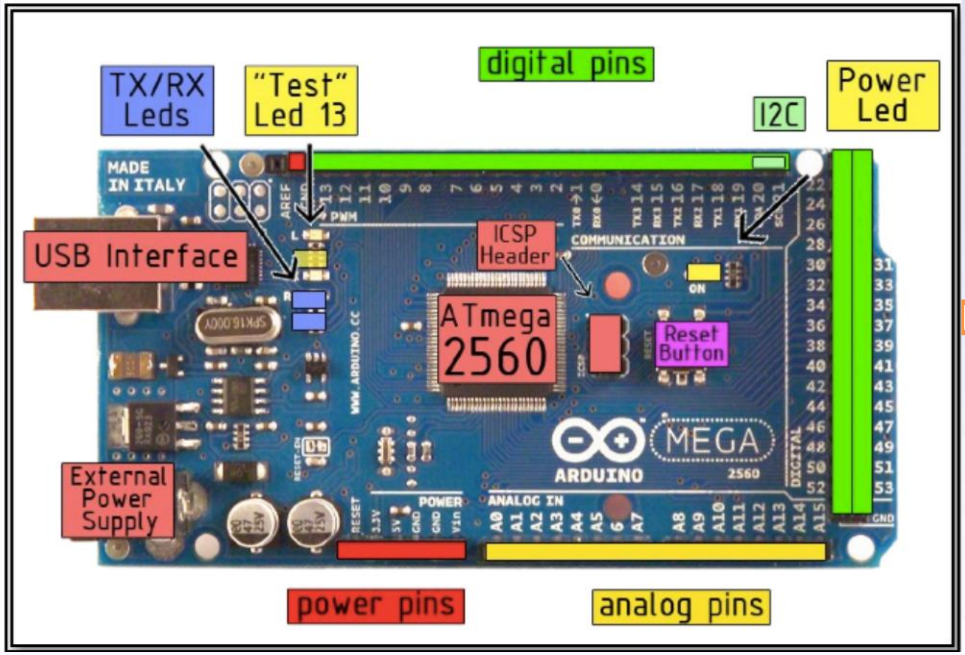
Electronic system

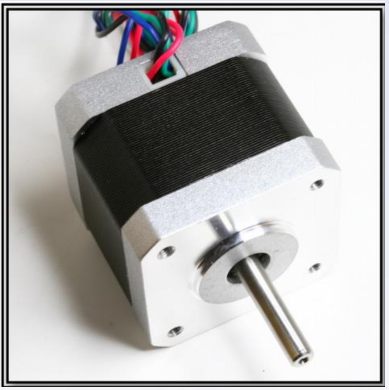
- Objective – drive the motors

Motor control

- Start / Stop
- Speed
- Torque
- Position







Stepper motor
Typically 3 – 5V; 1 – 1.5A

Motor controller with PWM signal output: Arduino
Output: 5V, <40mA

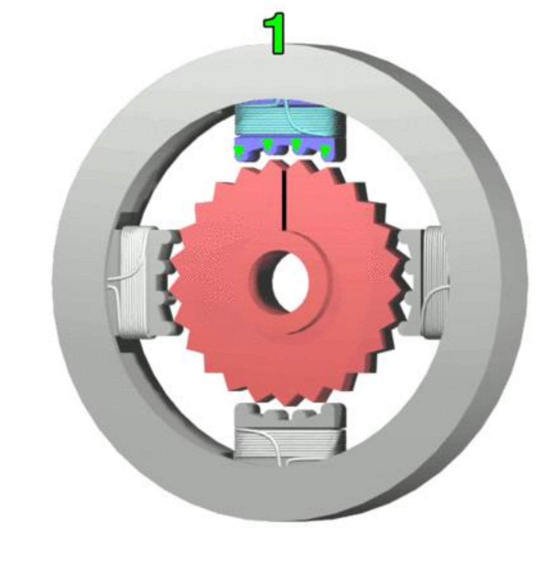
Electronic system

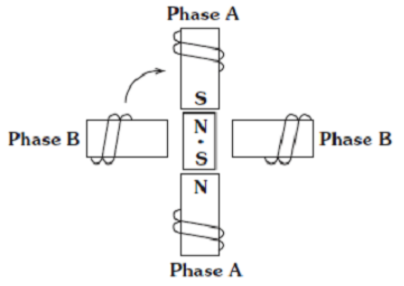
AM Machine

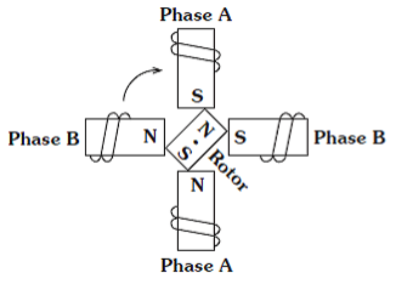
Motion System

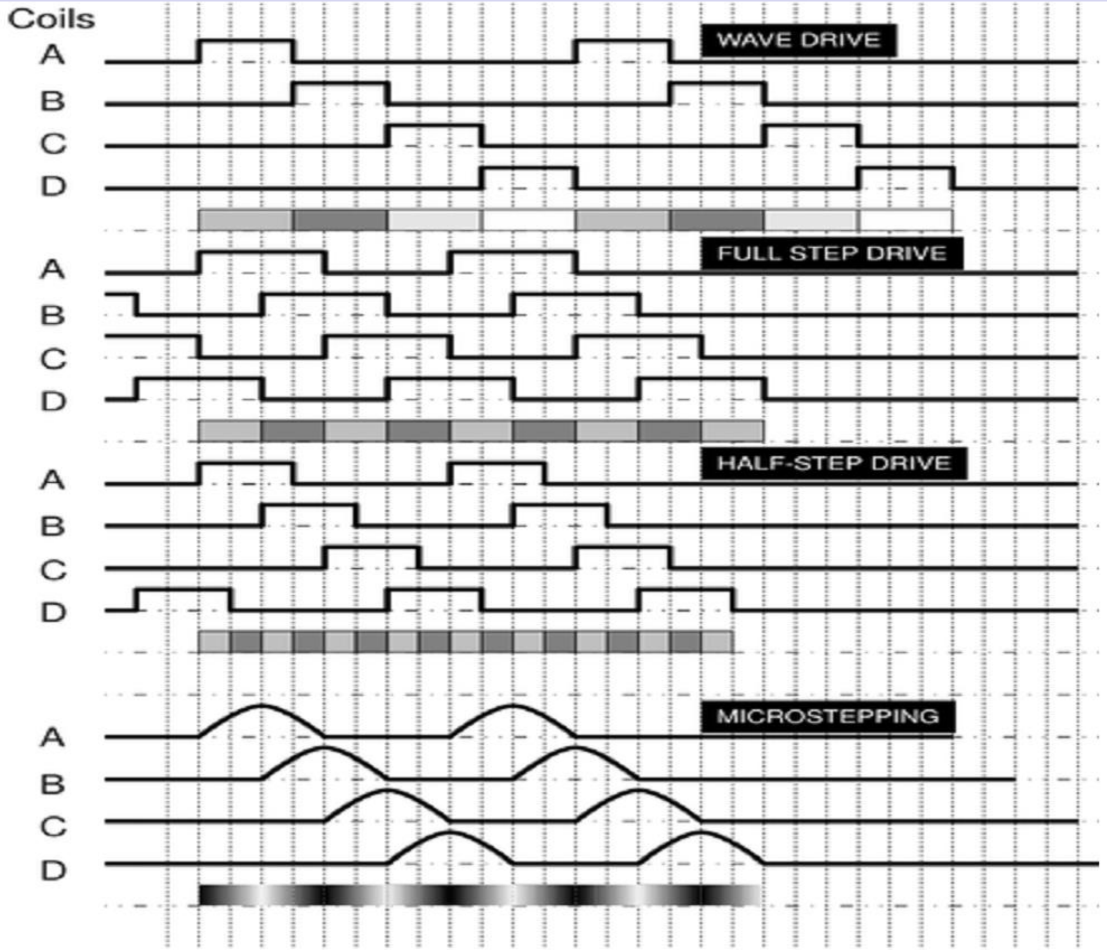
Electronic System

Stepper motor-drive modes





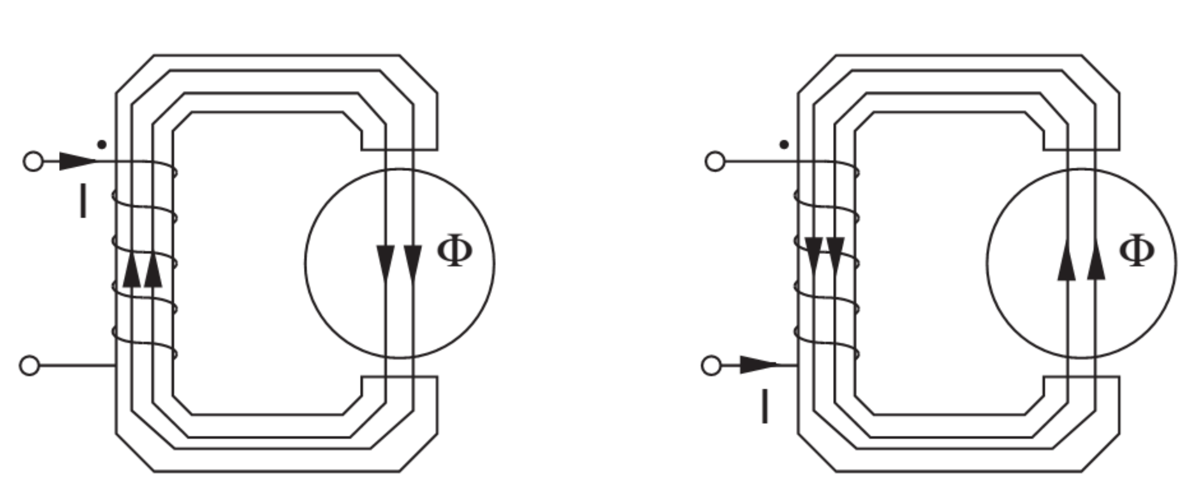




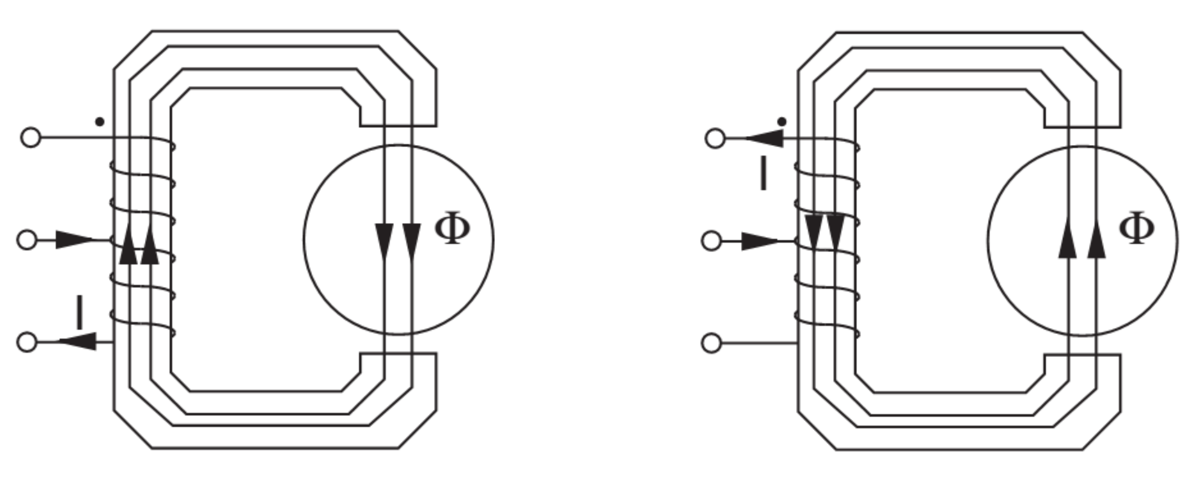
Drive modes: wave, full-step, half-step, microstepping
Vibration issue.

Electronic system

Stepper motor - types



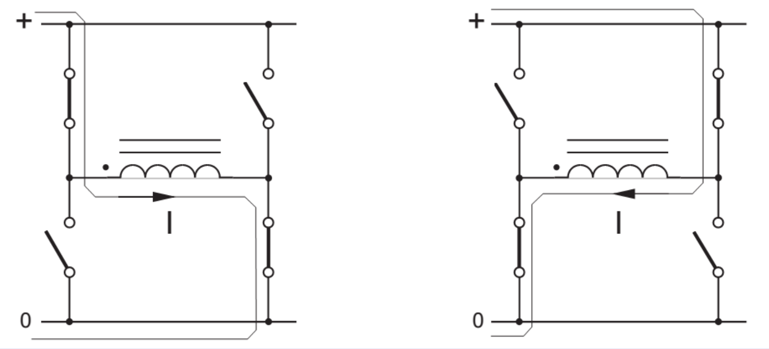
Bipolar – without center tap, larger torque



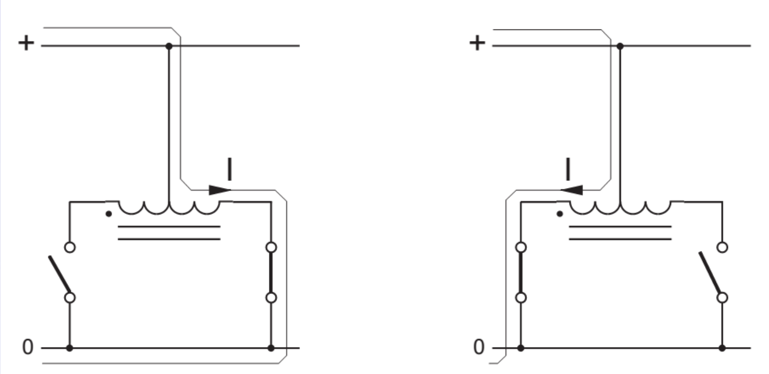
Unipolar – with center tap, only half of the coil is used to produce force, less power loss

Electronic system

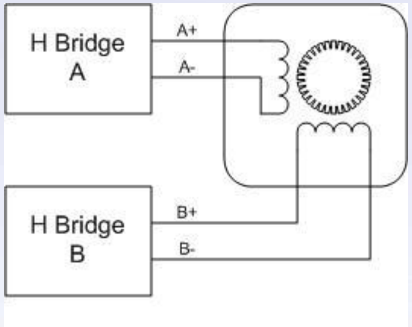
Stepper motor - types



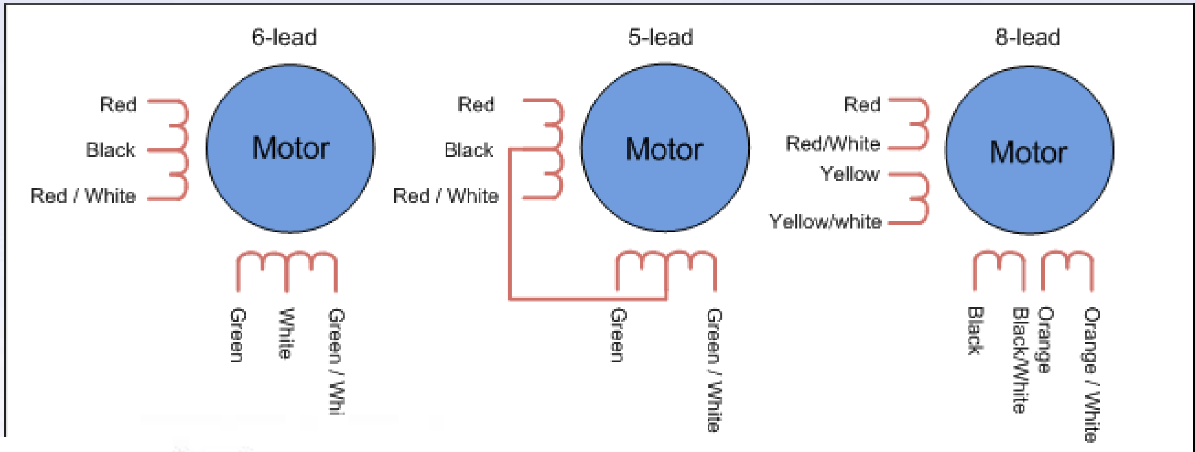
Bipolar drive circuit – H-bridge, more complicated, 4 switches per phase



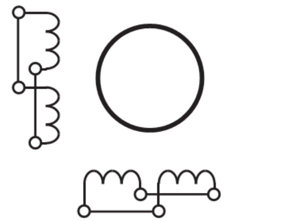
Unipolar drive circuit – 2 switches per phase



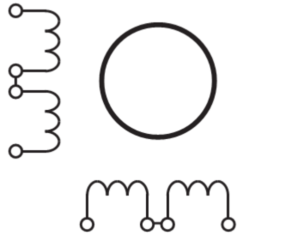
4 Wire Stepper Motor (Bipolar)



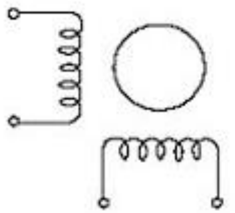
Unipolar Stepper Motor



Bipolar, parallel connection
4 windings / 8 leads



Bipolar, series connection
4 windings / 8 leads

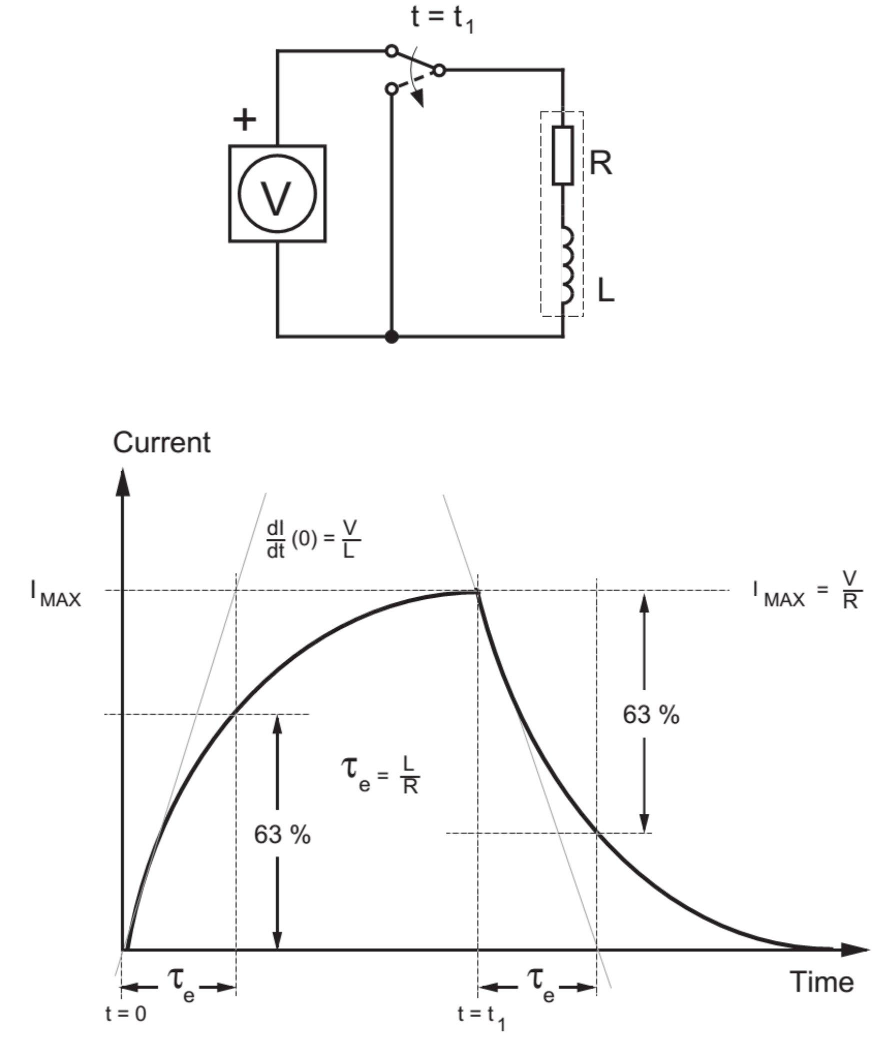


Bipolar-single winding 4 leads

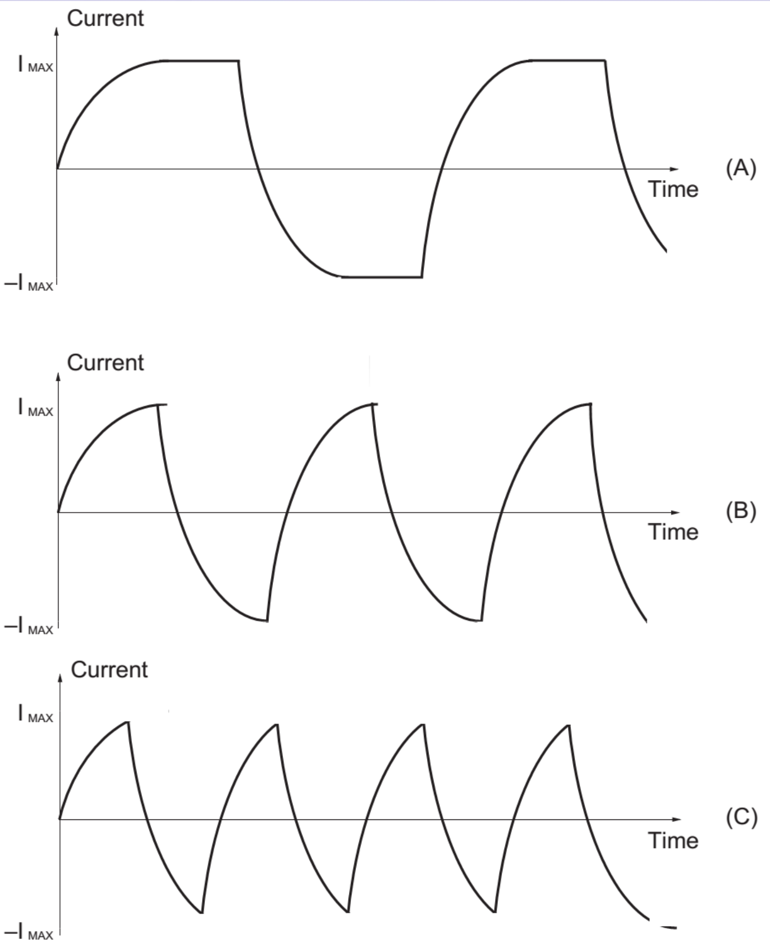
Count leads to distinguish unipolar and bipolar

Electronic system

Stepper motor - drive



Current wave in stepper motor windings with PWM voltage input



Current wave with input signal at different frequency, when above a certain f, current never reaches max

UNIVERSITY OF ARKANSAS

Electronic system

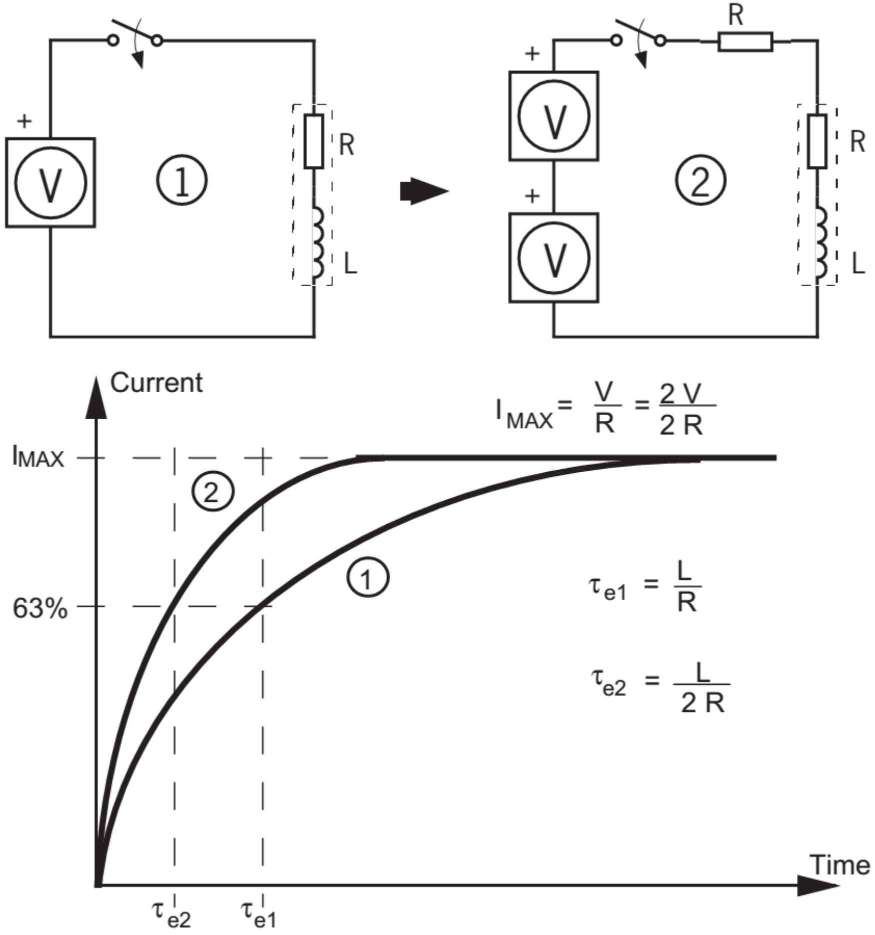
AM³ Lab
Advanced Manufacturing | Modeling | Materials

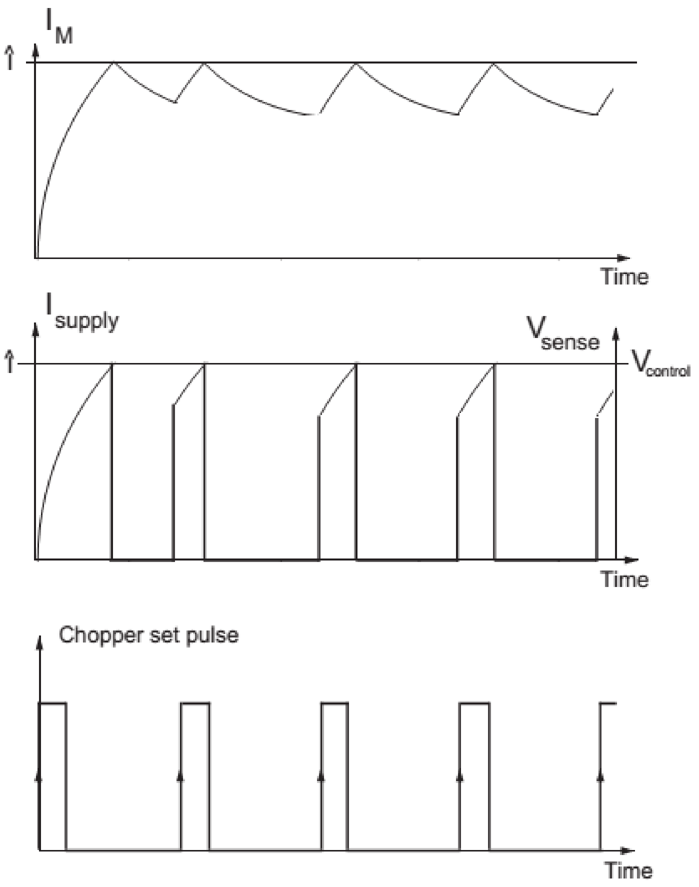
Stepper motor – drive – current control

AM Machine

Motion System

Electronic System


+
V
1
R
L
R
+
V
+
V
2
R
L
Current
$I_{MAX} = \frac{V}{R} = \frac{2V}{2R}$
IMAX
63%
2
1
$\tau_{e1} = \frac{L}{R}$
$\tau_{e2} = \frac{L}{2R}$
Time
τ_{e2}
τ_{e1}



Resistance limitation: Use a higher voltage and control the current with a resistance to speed up current build-up, more energy loss on resistance.

Chopper control: Use a very high voltage for fast current build-up and control the current by controlling the duty cycle. Optimal solution

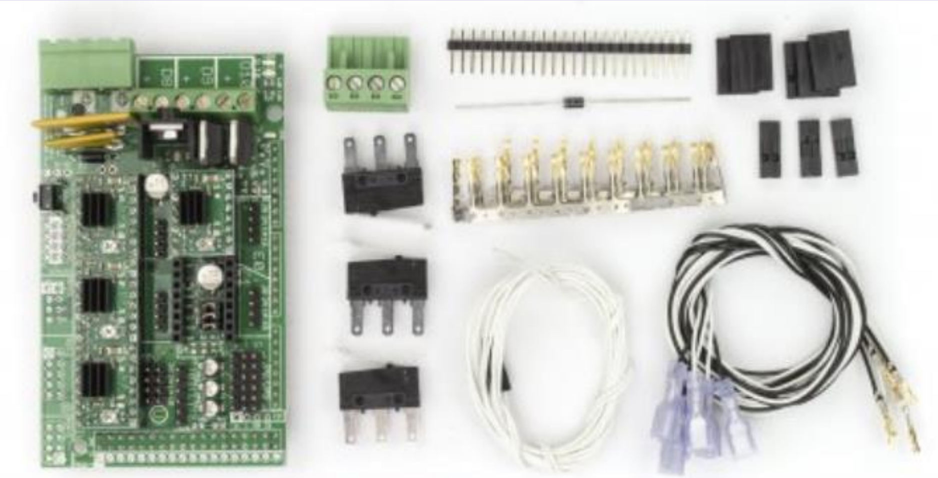
Electronic system

AM Machine

Motion System

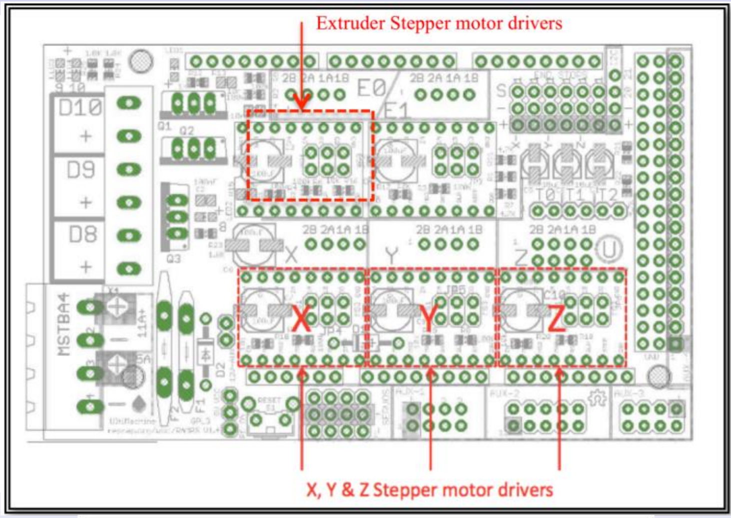
Electronic System

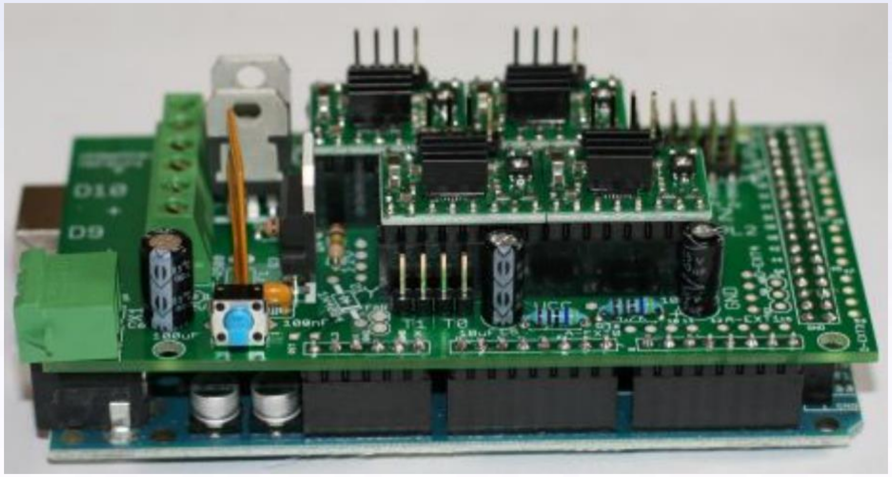
Stepper motor – drive



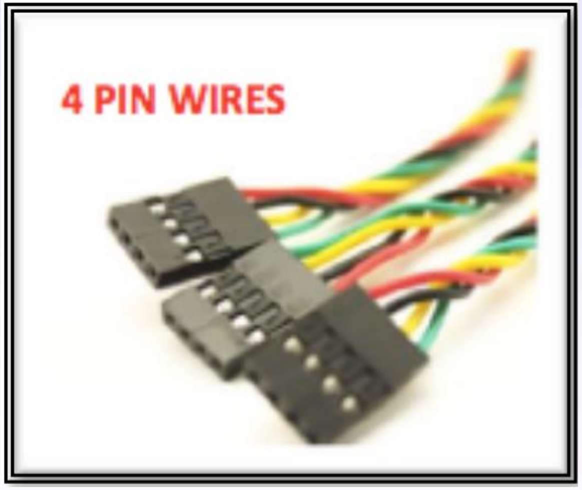
Shields: RAMPS 1.4 -- Pololu Shield provides the driver circuit with 11A fuse, power connection, chopper current control etc.for driving the stepper motor

RAMPS Schematic


Extruder Stepper motor drivers
D10
D9
D8
MSTBA4
E0
E1
X
Y
Z
X, Y & Z Stepper motor drivers



Plug it into Arduino

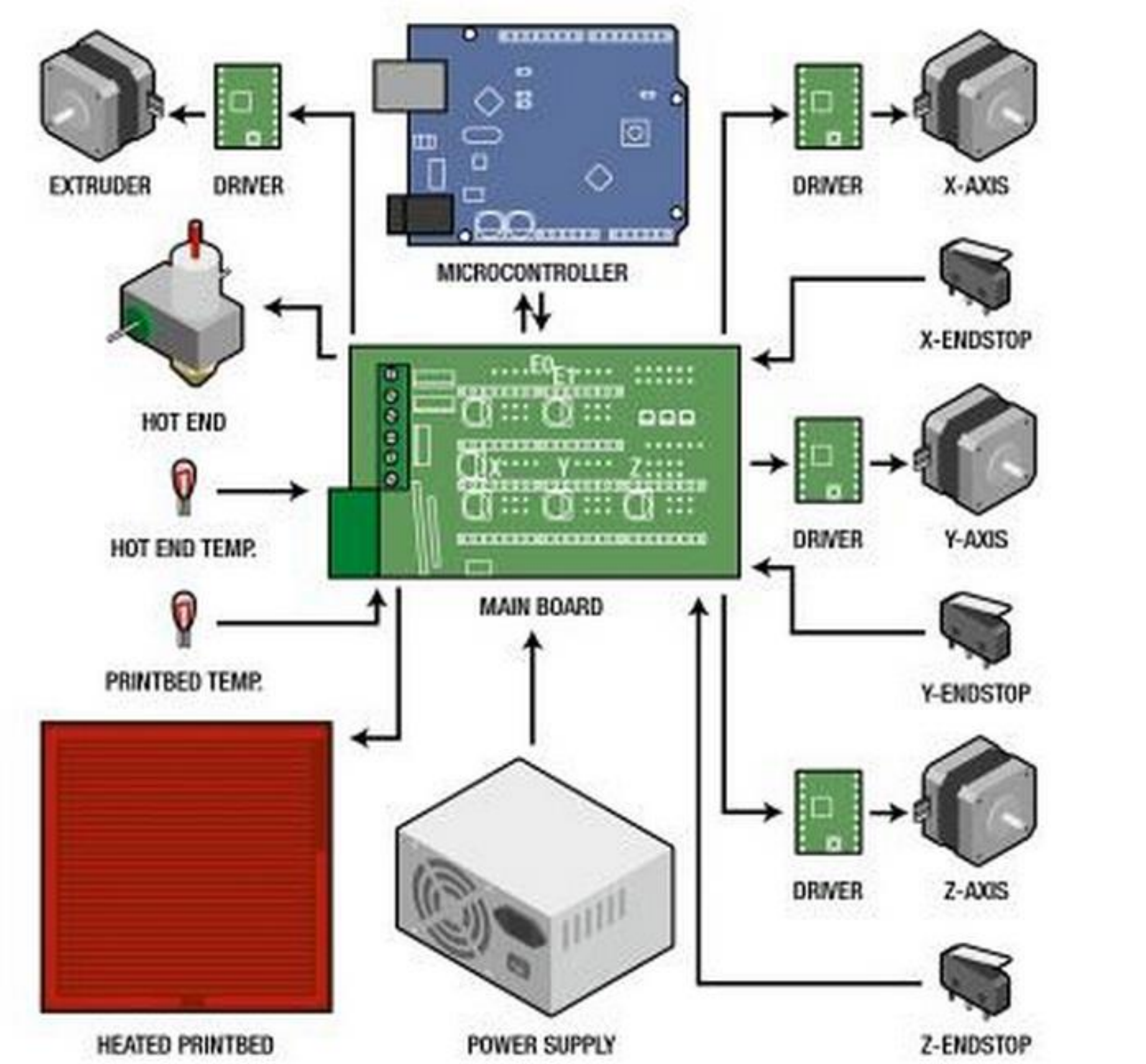


Plug in motor and power supply

Pic credits: Maram Sulimani@MSU

Electronic system

Stepper motor – drive



Pic credits: Practical 3D Printers by Brian Evans

Electronic system

AM Machine

Motion System

Electronic System

Stepper motor – use Arduino for PWM control

Use digitalWrite function

```
int myPin = 13;
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
// Approximately 10% duty cycle @ 1KHz
  delayMicroseconds(100);
  digitalWrite(13, LOW);
  delayMicroseconds(1000 - 100);
}
```

Use analogWrite function

```
// LED connected to digital pin 9
int ledPin = 9;
// potentiometer connected to analog
int analogPin = 3;   pin 3
// variable to store the read value

int val = 0;
void setup()
{
// sets the pin as output
  pinMode(ledPin, OUTPUT);
}

void loop()
{
// read the input pin
  val = analogRead(analogPin);
// analogRead values go from 0 to 1023,
// analogWrite values from 0 to 255
  analogWrite(ledPin, val / 4);
}
```

Electronic system

AM Machine

Motion System

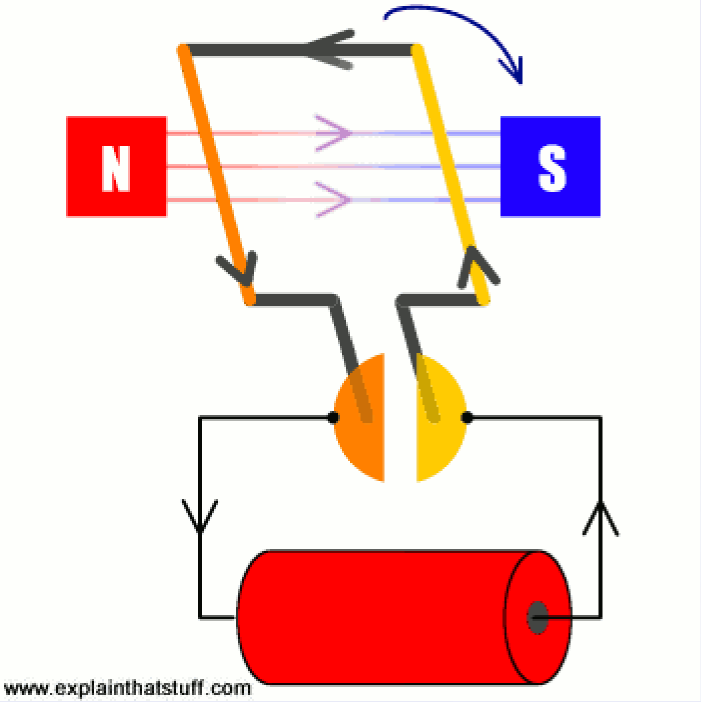
Electronic System

Stepper motor – use Arduino for PWM control

```
/*
Adafruit Arduino - Lesson 16. Stepper
*/

#include <Stepper.h>

int in1Pin = 12;
int in2Pin = 11;
int in3Pin = 10;
int in4Pin = 9;

Stepper motor(512, in1Pin, in2Pin, in3Pin,
in4Pin);

void setup()
{
pinMode(in1Pin, OUTPUT);
pinMode(in2Pin, OUTPUT);
pinMode(in3Pin, OUTPUT);
pinMode(in4Pin, OUTPUT);
```

```

// this line is for Leonardo's, it
delays the serial interface
// until the terminal window is
opened
while (!Serial);
Serial.begin(9600);
motor.setSpeed(20);
}

void loop()
{
if (Serial.available())
{
int steps = Serial.parseInt();
motor.step(steps);
}
}
```

THANK
YOU!



DC Motor animation