Introduction to Robotics

Big Robots

Machine Learning

Big Dog

Small Robots

Medical Robots

Military Robots

Multivariate Calculus

Mobile Robots

AI

Deep Learning

PSO

Humanoid Robots

Jumping Robots

Localization

Drones

AlphaGo Zero

Kinematics

Big Brother

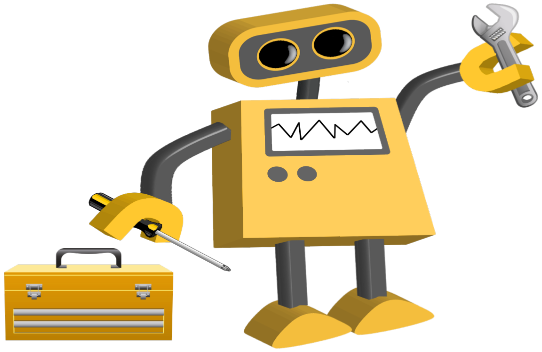
Wheeled Rbots

Legged Robots

Bayesian Probability

Welcome

Introduction to Robotics



Instructor

Tom Poliquin

tpoliquin@lumenetix.com

Course Overview

History of Robotics

Embedded Systems

Sensors

Actuators

Two Wheeled Robot

C Code / Hardware Interfaces

FSMs / Etc.

Projects

Introduction

Name

Where you work

What you expect from class

If awakened what language

What is this class ?

Quick overview of Robotics

Quick introduction to Embedded Systems

Basic real time programming paradigms

O)verview of basic sensors and actuators

What this class is not ...

No Deep Learning

No Kinematics Theory

No Baysean Decision Making

No High Level Abstractions

No Multivariate Calculus

Class Structure

Homework Review

Lecture

Quizes (?)

Homework Prep

---- Variable ---

Meta Info

Extremely Broad Topic

Potentially Overwhelming Prerequisites

Too Short

Too Long

Where are the Robots?

Behind the scenes

- Warehouse
- Milking Machines
- Car Assy

Interacting with humans

- Driverless Cars
- Education
- Companion

Robotics Overview

What is a robot?

What robots do now

Bio-inspired robots

Humanoid robots

Swarms and evolution

Future

What is a robot?

Senses

Acts Purposely

Intelligent / Autonomous

Brief History

Term from RUR (1920)

Aristotle 320 BC (intelligent tool)

60 AD (3 wheeled vehicle)

16th Century Golem (Humanoid Myth)

DaVinci Cart

1495 Autonomous Humanoid

1948 Walters Electric Vehicle

Philosophy Break

What do we want robots to do?

Member of congress

Bomb Disposal

Policeman

Baseball Pitcher

Ariline Pilot

Robot Parts

Sensors

Actuators

Brain (Not always required *)

* Braitenberg Vehicles

Robot Functions

	Bio	Robot
Sensing	Eyes Ears Whiskers	Cameras Microphones Switches
Signalling	Voice Face	Loudspeaker Wifi
Moving	Legs	Motors
Manipulating	Hands	Motors
Energy	Stomach	Batteries

How to Classify

Complicated

See tables 2 and 3 in text

Autonomy

Remotely Operated (Is this really a robot?)

Perform preprogrammed mission (by human)

Automous (Cruise missle)

Roomba

Rodney Brooks

2002 (7 generations)

Senses Obstacles

Senses Edges

Senses Dust Bin Full

Senses Battery Discharged

Moves

Random Walk Inefficient (early versions)

Where is Intelligence?

Looks Intelligent

Really Intelligent

Increase Intelligence

Preprogram more behaviours

Design robot to learn and develop

World is difficult

Language

Culture

Current Robots

Assembly Line

- Stationary
- Rigid Environment
- How to Program
- E.G. Spot Welder

Current Robots

Fetch and Carry

- Mobile
- Directed Cooperation
- Can Use Localization and Mapping

Current Robots

Tele-Operated

- Need communication link
- Undersea Rovers
- UAV
- Surgical Robots

Current Robots

- Education
- Development
- Research

Biologically Inspired

Artificial Life

Behaviour Based
vs
Sense-Plan-Act

Learning

Uncanny Valley

Companions

Swarms

Termites

Ants

Cisco Routers

UAVs

Ethics

Self Driving Cars

Speedy (Asimov)

Humans Make Skynet

Future

Planteary Explorers

Replication

MicroBots

NanoBots (Grey Goo)

Language / Culture

Meet Your Robot

- Brain
- OptoSensors (Obviously Sensors)
- Motor (Actuator)
- LEDs (Actuator)
- Speaker (Actuator)

Embedded Systems

How are they different from normal programs ?

- Normal programs have limited I/O
- Not usually real time
- Run on Desktops / Laptops
- Concurrancy usually required

Embedded Systems

Care a lot about time

Have lots of different types of I/O

Often use polling techniques

Often use interrupts

Often use DMA

Time

Embedded Systems usually have one core

Must give the appearance of doing multiple things at the same time

Often must have minimum latency to events

Must be close to real time

Input / Output

Input / Output is quite varied,

- Individual Pins (Digital)
- Individual Pins (Analog)
- Peripherals
- Many protocols,

Serial	SPI	
I2C	CAN	Parallel
DMX	USB	

Polling

Polling is a technique for reading sensors, or watching for events

- Simple
- Generally CPU intensive
- Difficult to achieve low latency

Interrupts

Interrupts are a hardware aid to handling events using a software callback

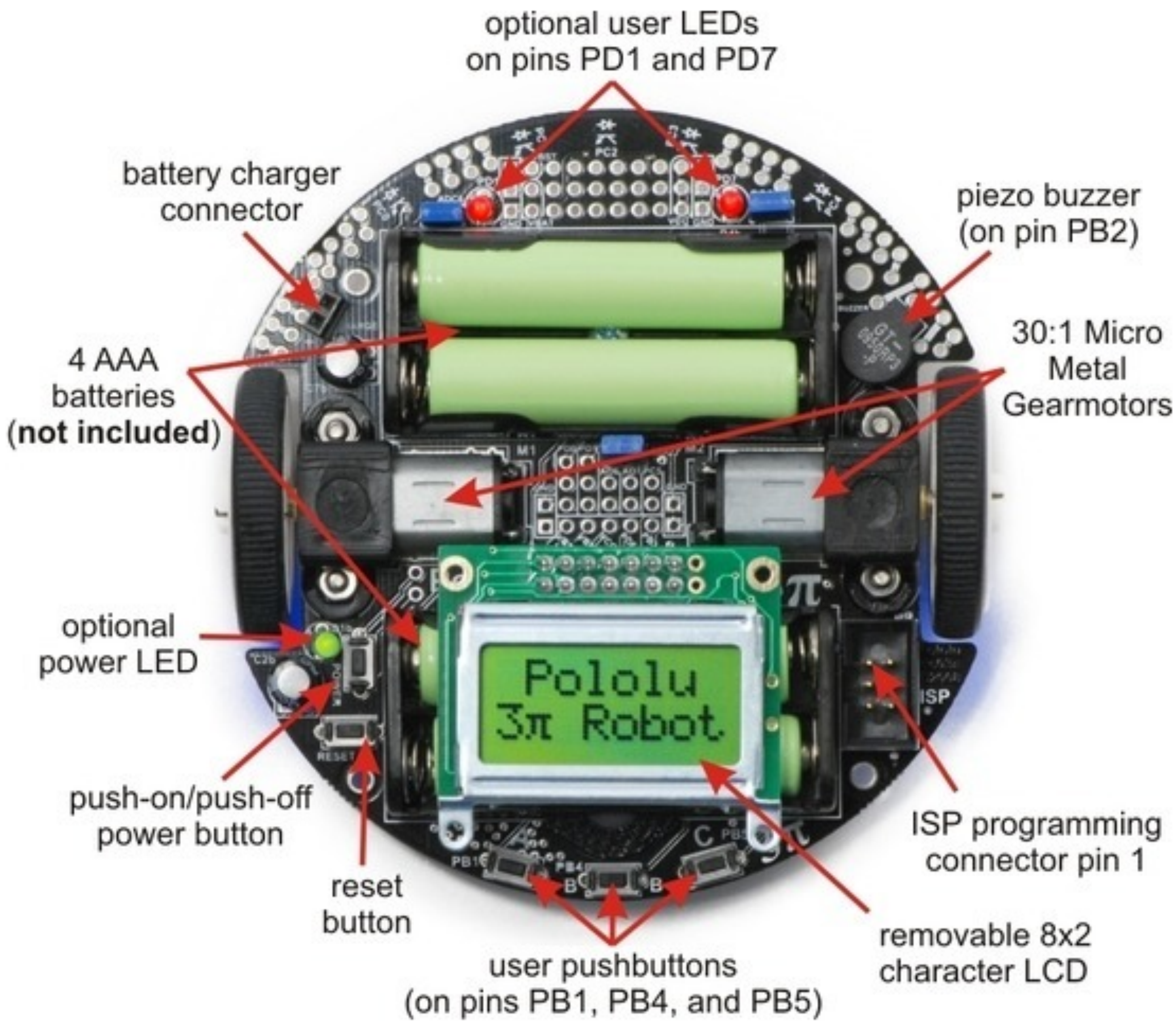
More complex than polling

Low latency

Low CPU impact

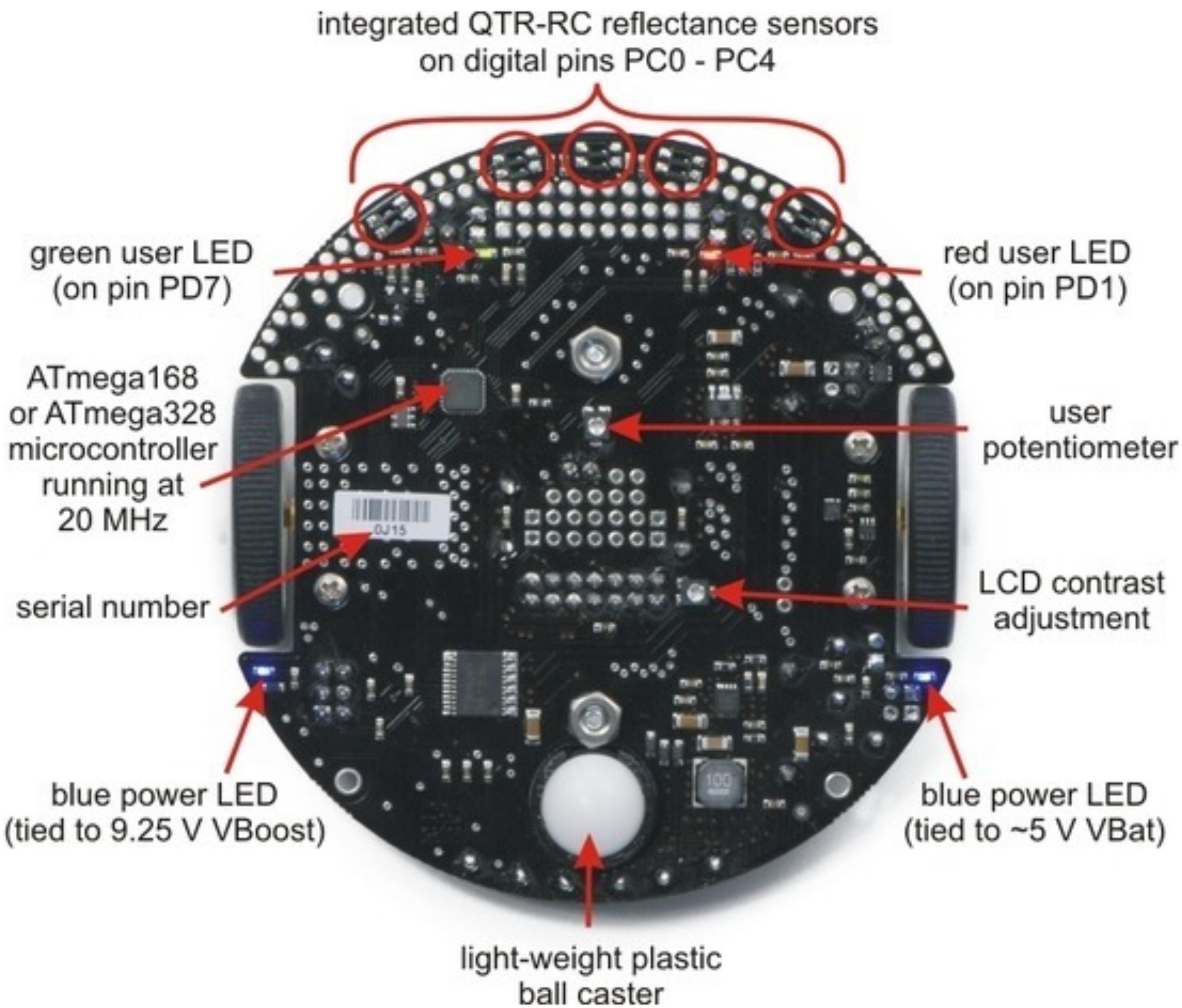
Potential concurrency issues

Meet Your Robot



General features of the Pololu 3pi robot, top view.

Top Side



Labeled bottom view of the Pololu 3pi robot.

Bottom Side

Pin Functions

Pin Assignment Table Sorted by Function

Function	ATmegaxx8 Pin	Arduino Pin
free digital I/Os (x3) (remove PC5 jumper to free digital pin 19)	PD0, PD1, PC5	digital pins 0, 1, 19
free analog inputs (if you remove jumpers, x3)	PC5, ADC6, ADC7	analog inputs 5 – 7
motor 1 (left motor) control (A and B)	PD5 and PD6	digital pins 5 and 6
motor 2 (right motor) control (A and B)	PD3 and PB3	digital pins 3 and 11
QTR-RC reflectance sensors (left to right, x5)	PC0 – PC4	digital pins 14 – 18
red (left) user LED	PD1	digital pin 1
green (right) user LED	PD7	digital pin 7
user pushbuttons (left to right, x3)	PB1, PB4, and PB5	digital inputs 9, 12, and 13
buzzer	PB2	digital pin 10
LCD control (RS, R/W, E)	PD2, PB0, and PD4	digital pins 2, 8, and 4
LCD data (4-bit: DB4 – DB7)	PB1, PB4, PB5, and PD7	digital pins 9, 12, 13, and 7
reflectance sensor IR LED control (drive low to turn IR LEDs off)	PC5 (through jumper)	digital pin 19
user trimmer potentiometer	ADC7 (through jumper)	analog input 7
2/3rds of battery voltage	ADC6 (through jumper)	analog input 6
ICSP programming lines (x3)	PB3, PB4, PB5	digital pins 11, 12, and 13
reset pushbutton	PC6	reset
UART (RX and TX)	PD0 and PD1	digital pins 0 and 1
I2C/TWI	inaccessable to user	
SPI	inaccessable to user	

Which Pins Do What ?

Peripherals

Two kinds of Peripherals

Internal ....

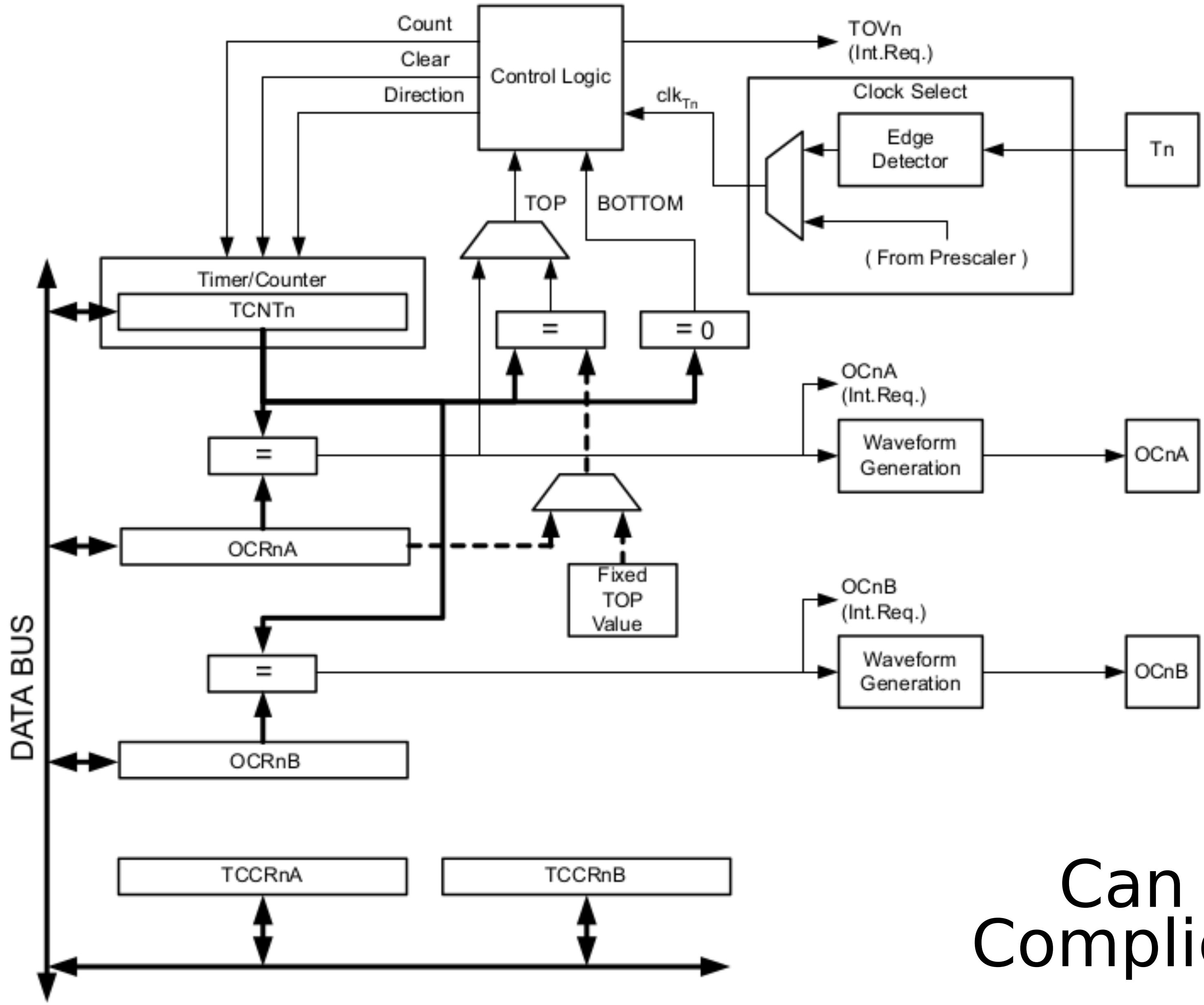
Timers EPROM Serial
SPI Watchdog GPIO
Power Mgmt Interrupt Ctllr

External ....

Motors LEDs Buttons
OptoSensors IMU etc.

Timers

Figure 19-1. 8-bit Timer/Counter Block Diagram



Can be Complicated

Watchdog

Used to prevent hangups

Countdown timer activated

Code must set a register periodically
to keep counter from expiring

It watchdog timer expires
usually performs soft reset

Interrupts

16.1. Interrupt Vectors in ATmega328/P

Table 16-1. Reset and Interrupt Vectors in ATmega328/P

Vector No	Program Address(2)	Source	Interrupts definition
1	0x0000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x0002	INT0	External Interrupt Request 0
3	0x0004	INT1	External Interrupt Request 0
4	0x0006	PCINT0	Pin Change Interrupt Request 0
5	0x0008	PCINT1	Pin Change Interrupt Request 1
6	0x000A	PCINT2	Pin Change Interrupt Request 2
7	0x000C	WDT	Watchdog Time-out Interrupt
8	0x000E	TIMER2_COMPA	Timer/Counter2 Compare Match A
9	0x0010	TIMER2_COMPB	Timer/Coutner2 Compare Match B
10	0x0012	TIMER2_OVF	Timer/Counter2 Overflow
11	0x0014	TIMER1_CAPT	Timer/Counter1 Capture Event
12	0x0016	TIMER1_COMPA	Timer/Counter1 Compare Match A
13	0x0018	TIMER1_COMPB	Timer/Coutner1 Compare Match B
14	0x001A	TIMER1_OVF	Timer/Counter1 Overflow
15	0x001C	TIMER0_COMPA	Timer/Counter0 Compare Match A
16	0x001E	TIMER0_COMPB	Timer/Coutner0 Compare Match B
17	0x0020	TIMER0_OVF	Timer/Counter0 Overflow
18	0x0022	SPI STC	SPI Serial Transfer Complete
19	0x0024	USART_RX	USART Rx Complete
20	0x0026	USART_UDRE	USART Data Register Empty
21	0x0028	USART_TX	USART Tx Complete
22	0x002A	ADC	ADC Conversion Complete
23	0x002C	EE READY	EEPROM Ready
24	0x002E	ANALOG COMP	Analog Comparator

Special fixed addresses

Firmware

High Level Drivers

- Easy to do Easy Things
- Sometimes buggy
- Sometimes difficult to integrate
- Timing Conflicts
- Peripheral Conflicts
  - Pins
  - Interrupts
  - Timers
  - CPU Cycle

Firmware

Low Level Drivers

- Hard to do Easy Things
- Less code; Less bugs.
- Close to bare metal
- Simpler to integrate
- Full control of,
  - Interrupts
  - Timers
  - CPU utilization
  - Timing

Modules

Follow good programming practices

Small, single purpose modules

''c' and 'h' files

'exports' and prototypes

Multitasking

Cooperative

- Periodic task execution
- Don't hog the CPU

Preemptive

- External entity determines when a task runs

Homegrown

- Difficult to get right in large projects

RTOS

Bit Constants

TMREG = 0x05
TMREG = 0b00000101

Update all bits

TMREG |= 0x04 Updates only bit 2 (1)

TMREG &=0x04 Updates only bit 2 (0)

Better

```
#define LAUNCH_MISSLES 5
TMREG |= (1 << LAUNCH_MISSLES)
TMREG &= ~(1 << LAUNCH_MISSLES)
```

Which one launches the missles ?

Timer Overview

Timers are complicated

Probably the most comple peripheral

Timers ....

- Provide precise timing
- Generate periodic interrupts
- Toggle output pins on schedule
- Generate complex PWM signals
- Count events
- Interrupt on pin changes
- Simplify communication routines

Timer Overview

Polling is bad

It eats up CPU cycles

Doing many things with a polling framework makes the code complex

Timers help eliminate polling through the use of interrupts

Electronics

Instant Electronics Course ......

Ohms Law

Kirchoff's Law

Voltage Dividers

Capacitors

Inductors

Practical Electronics

Typical Capacitor Usage

- DC Blocking
- Filtering
- Delay

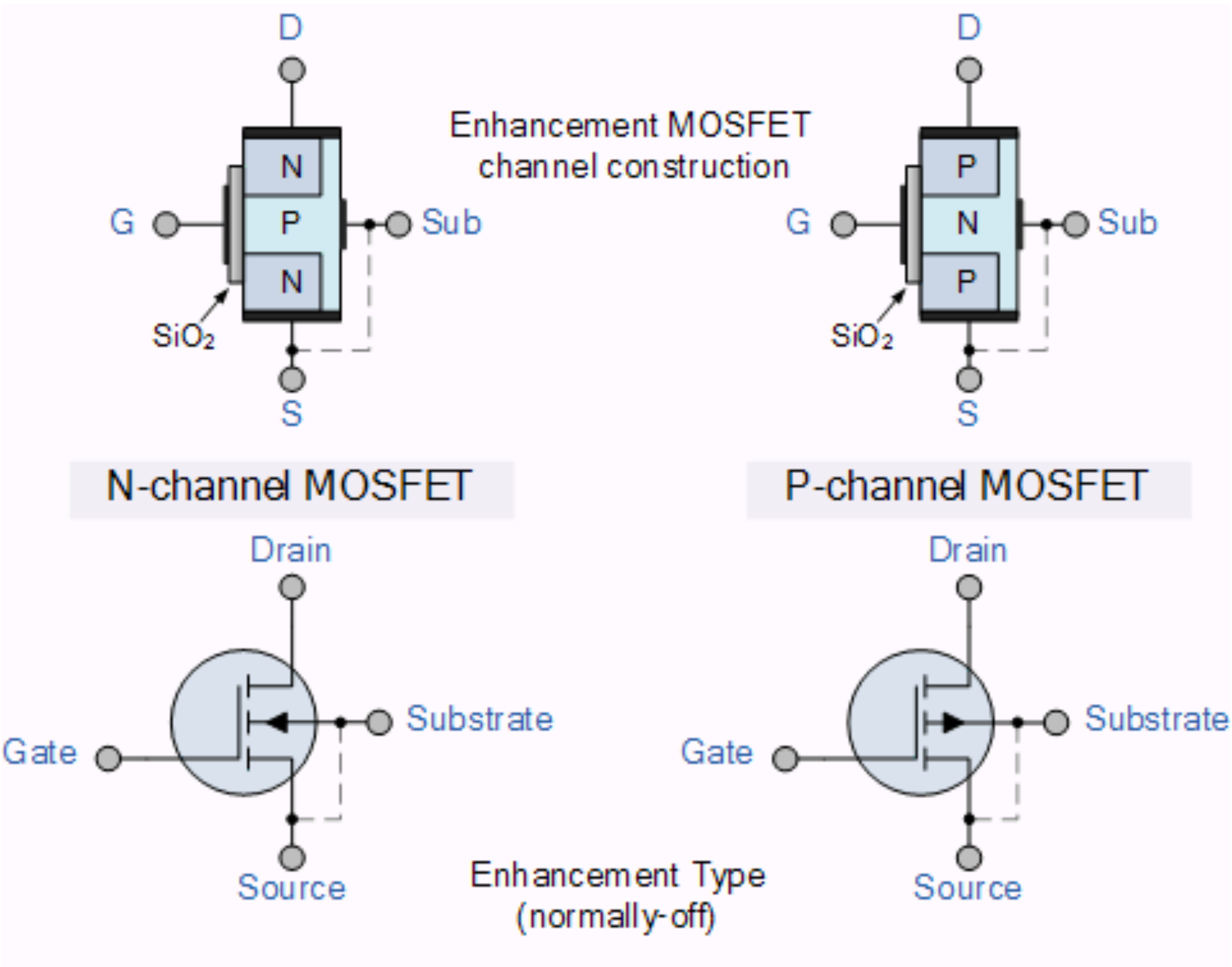
Typical Inductor Usage

- Noise Filter
- Switching Supplies

Typical FET Usage

- Power Driver
- Analog Switch

MOSFETs

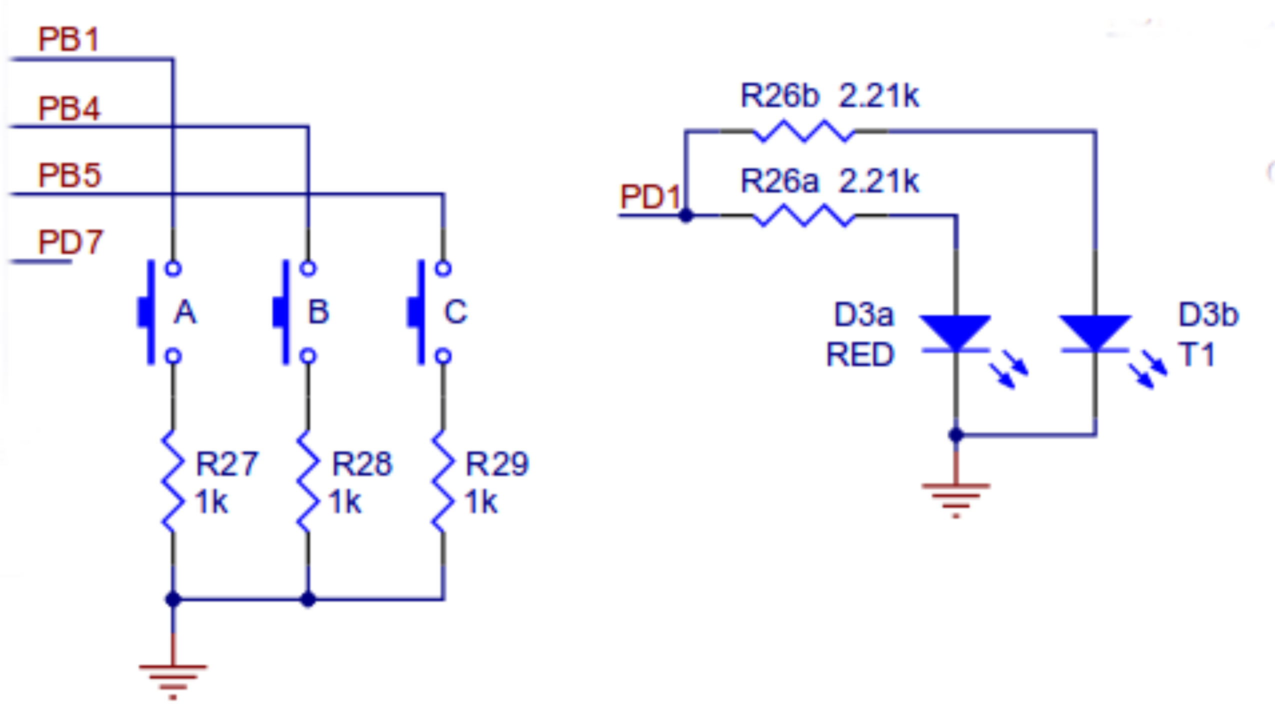


Used as

Drivers and Switches

LED / Button

Bad Practice



LED Specs (Typ)

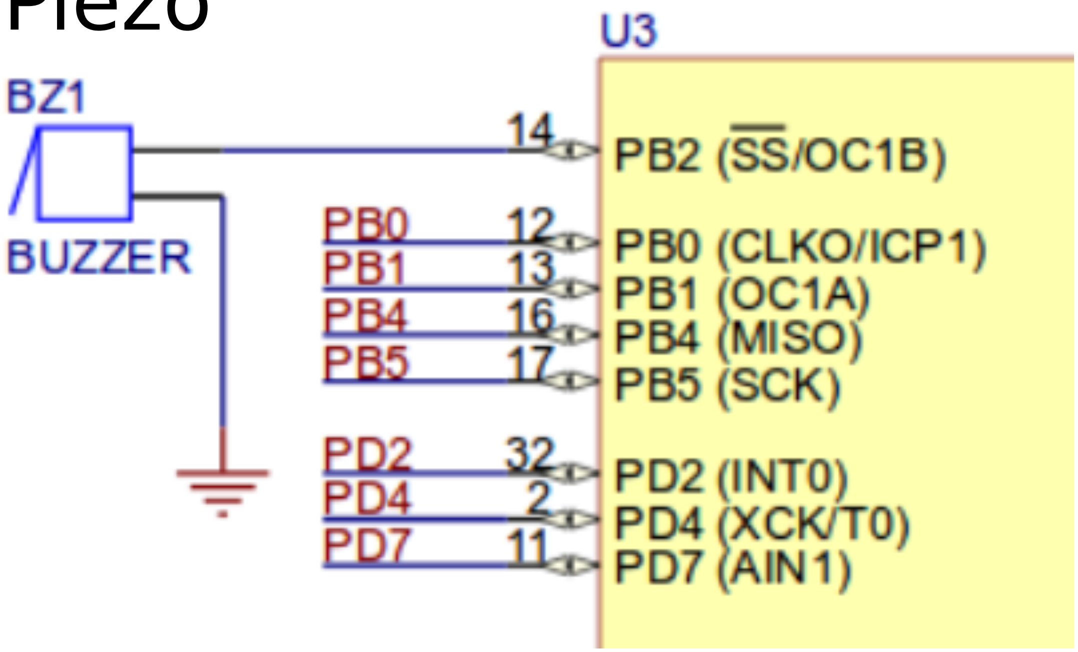
LEDs are current devices
(not voltage)

They are all different !

Yellow	19 mcd	585 nm
	Vf = 2.1V	If = 20ma
Blue	If = 20ma	500 mcd
	Vf = 3.2V	468 nm
Green	If = 2ma	38 mcd
	Vf = 1.8V	570 nm

Buzzer

Hi Z Piezo

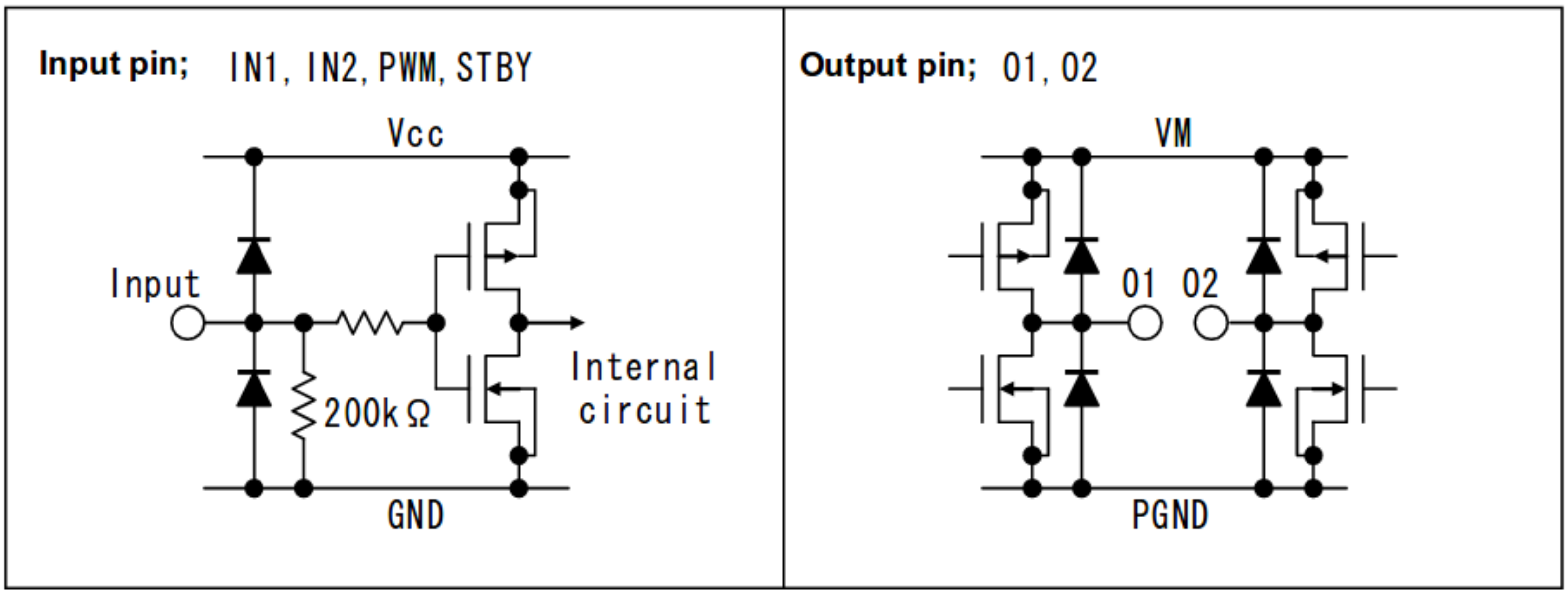


Motor

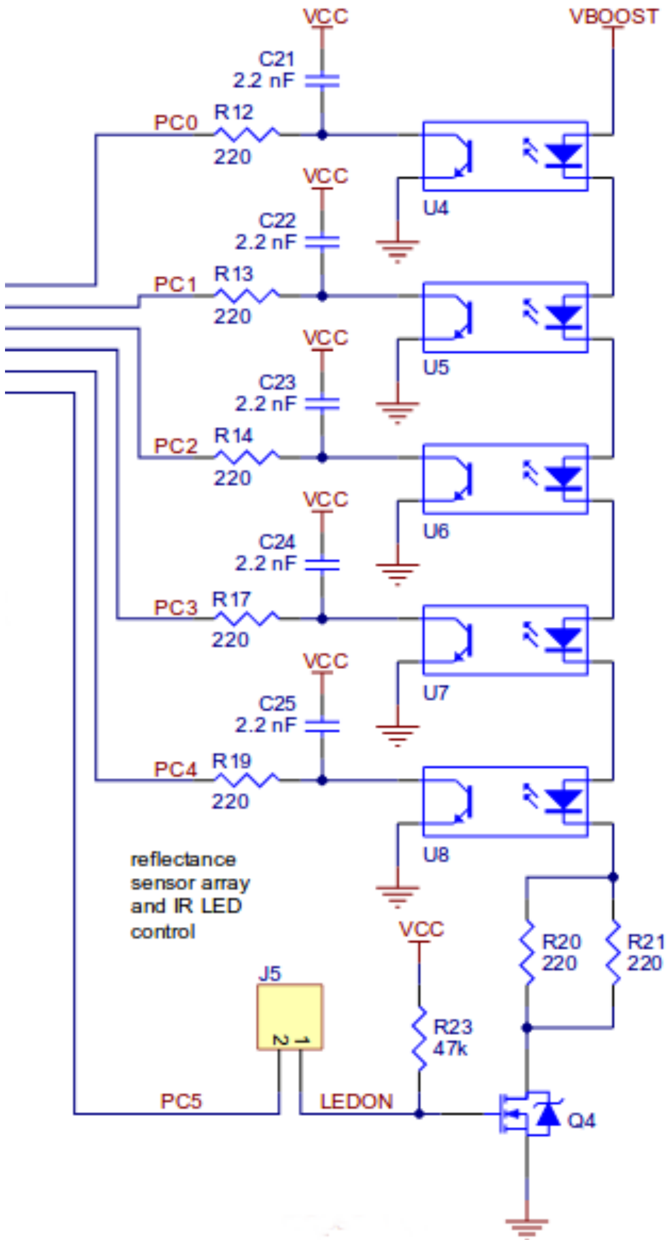
Gear ratio:	30:1
Free-running speed:	700 rpm
Free-running current:	60 mA
Stall torque:	6 oz·in
Stall current:	540 mA

Motor Driver

Standard H Driver



Optosensor



Tricky

..... but cheap

Mechanics

You want to be able to control your robot

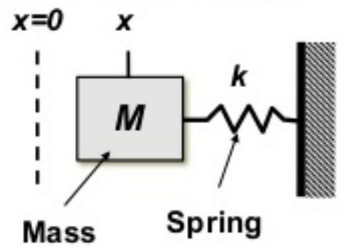
This requires a 'Control System'

System Identification makes this easier

System Identification

You need some equations of motion

Deriving the equation of motion from the energy



$$\frac{d}{dt}(T+U)=\frac{d}{dt}\left(\frac{1}{2}m\dot{x}^2+\frac{1}{2}kx^2\right)=0$$

$$\Rightarrow \dot{x}\left(m\ddot{x}+kx\right)=0$$

Since $\dot{x}$ cannot be zero for all time, then

$$m\ddot{x}+kx=0$$

College of Engineering College of Engineering 4/53
© Eng. Vib, 3rd Ed.

How do you get these?

System Identification

You need some basic physics

You need to decide between,

State Space
Laplace

For small systems ..

Laplace

For larger more complex systems ..

State Space

Model

Let's Pick Laplace

We make our model (some math)

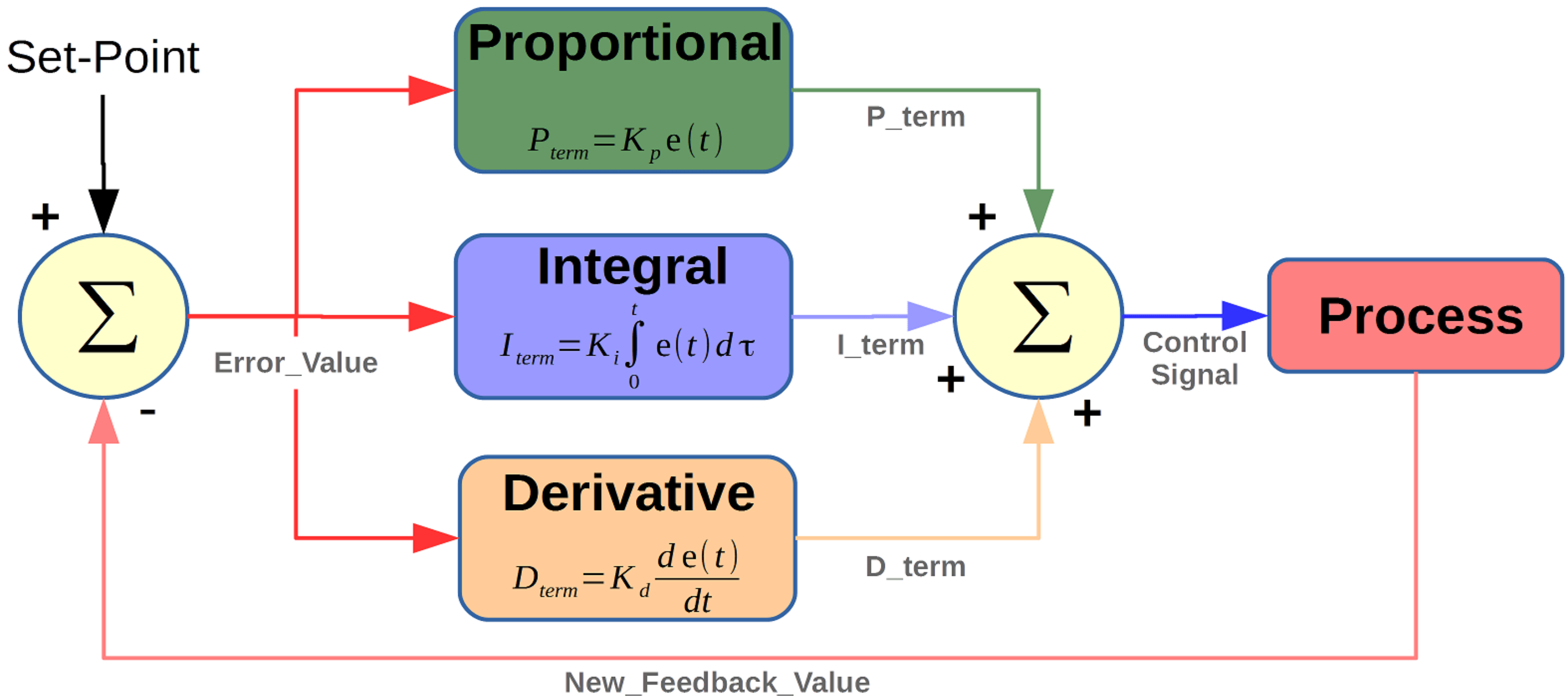
We decide what we want to do

Our goal is to make our
two wheeled robot to

move in a straight line
in a particular direction

PID

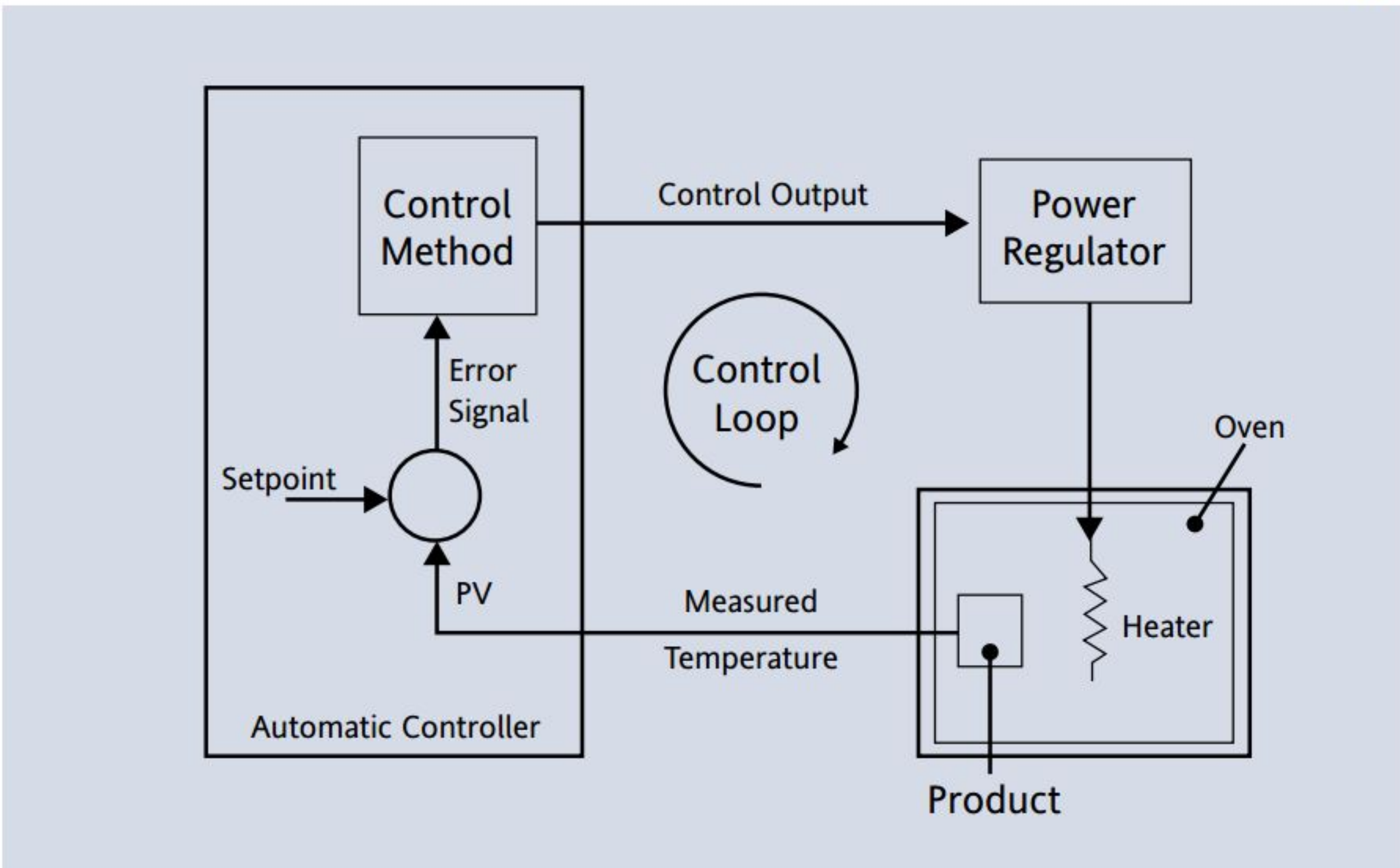
The simplest and most common control method for simple systems is a PID controller



PID Controller

A simple thermostat example

Automatic control loop

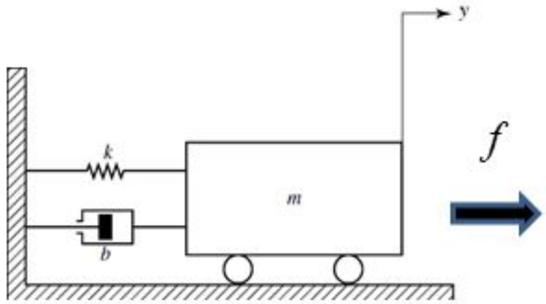


Octave Example

MATLAB Simulation
Mass Spring Dashpot System

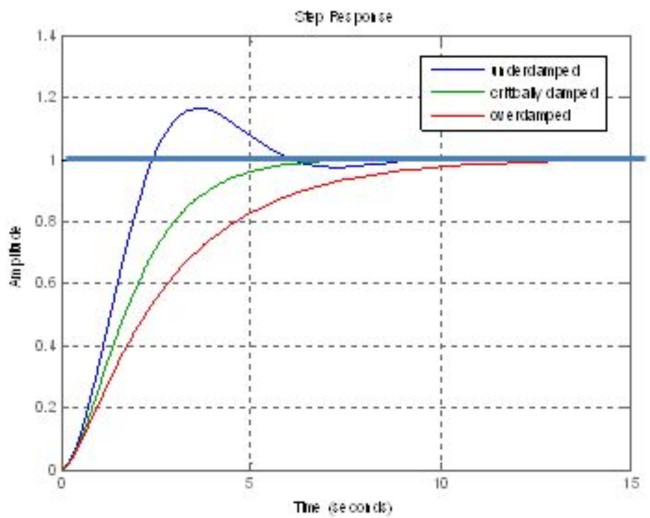
- Transfer function $G(S) = \frac{Y(s)}{F(s)} = \frac{1}{ms^2+bs+k}$
- m=1, k=1

$$\ddot{x} + 2\zeta\omega_0\dot{x} + \omega_0^2 x = 0.$$



- Case study
  - b=1 (underdamped ζ<1)
  - b=2 (critically damped ζ=1)
  - b=3 (over damped ζ>1)

```
num = 1
den = [1 b 1]
sys = tf(num, den)
step(sys)
```



Thomas Bayes (1701–1761)

Prior Probability

Likelihood of the evidence 'E' if the Hypothesis 'H' is true

$$P(H|E) = \frac{P(H) * P(E|H)}{P(E)}$$

Posterior Probability of 'H' given the evidence

Priori probability that the evidence itself is true

Bayes

Uses -

- Inference
- Statistics
- Probability
- Programming (New)
- Probabilistic Robotics
- Localization

Competitive Advantage

Bayes

Piano Example

Cancer Screening Example

Robotic Example

Particles

Bayes

Bayes in Robotics

Main use is Localization

We can't do a lot until
we know where we are.

Used with Particles

MDP

Markov Decision Process

Assumes next state only dependent on previous state

Tied to Bayes

Image Processing

Large Processing Power

Simple - Edge Detection

Complex - Deep Learning

Hard Problem

AI

Summary

What did we learn

Robotics is a gigantic field

Pick and Place
Industrial
Medical
Recovery
Companions
.....;.....