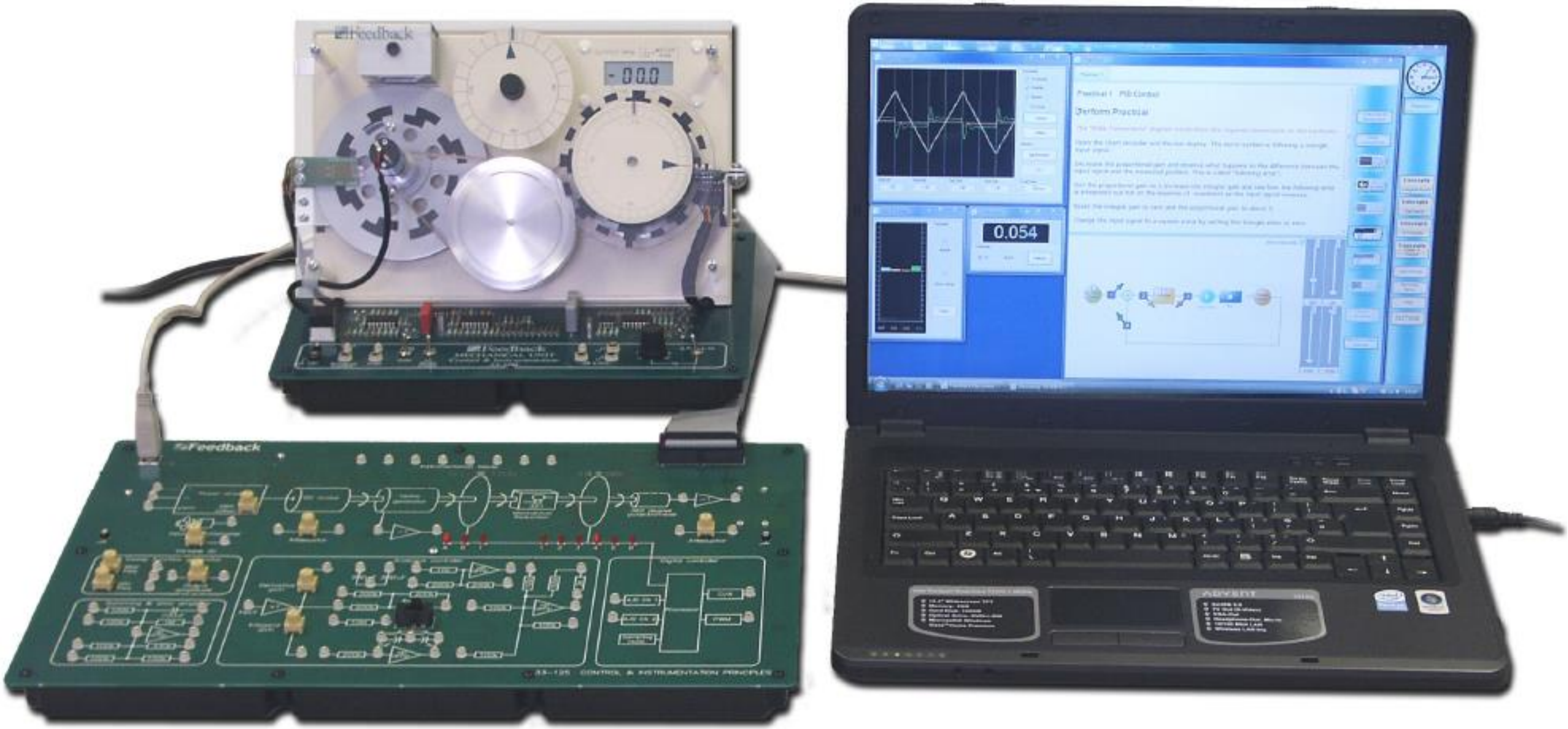
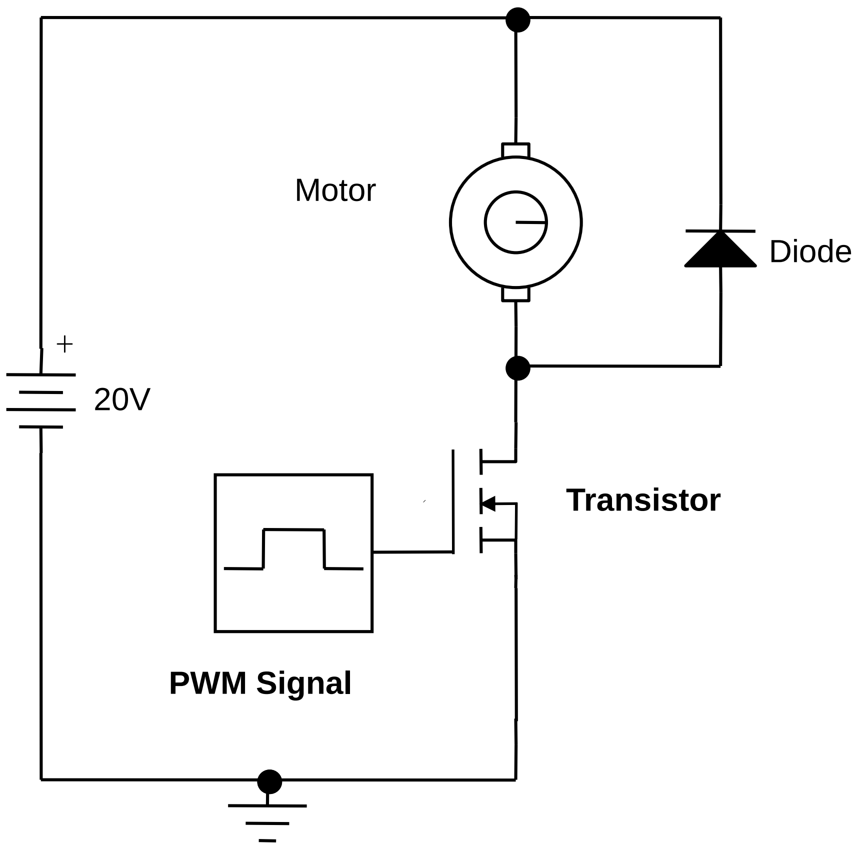
Speed Control of DC motor

Experimental Set-up

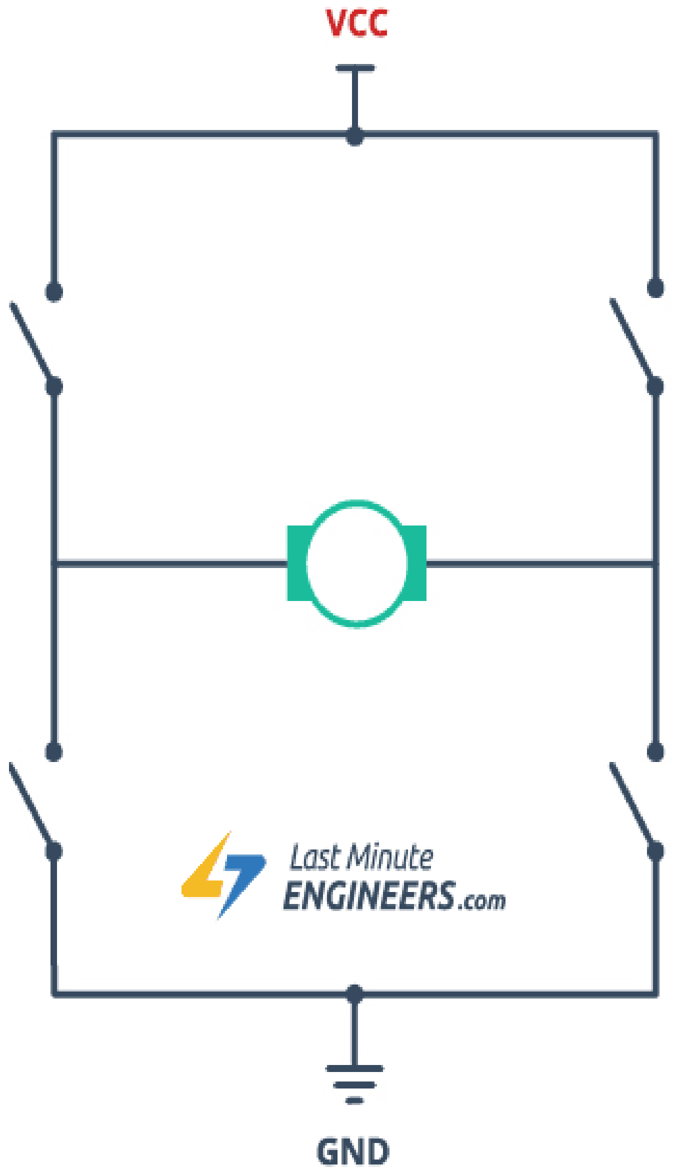


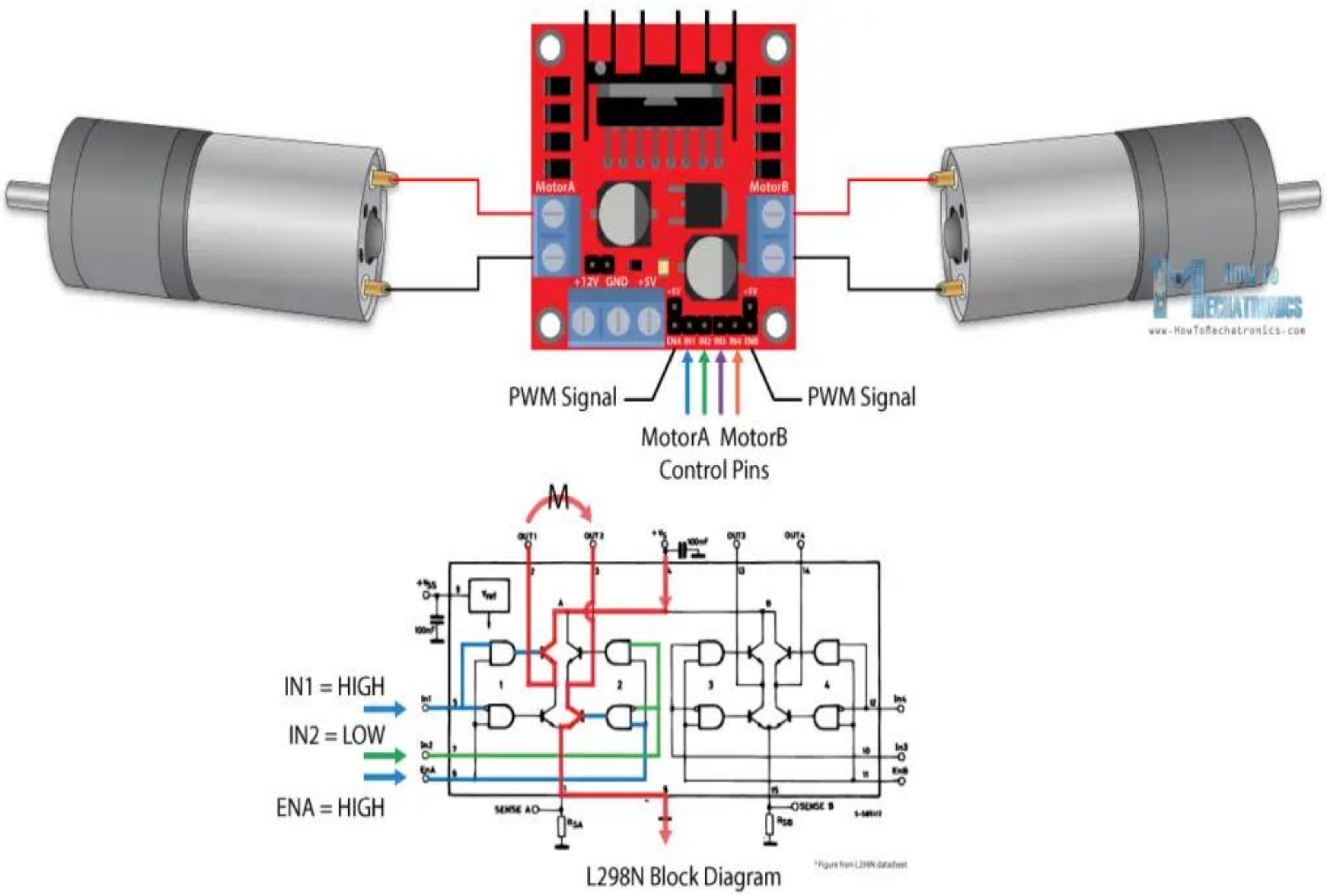
Motor Connection Diagram

- 15V DC Motor
- Motor speed controlled by the PWM signal which is generated by the Micro-Controller at one of its digital output port pin.
- The PWM signal essentially controls the On/Off state of the transistor, allowing the input current to flow through the motor or block it.



H-bridge and LM298N motor driver





Sensors for Mechatronics System

- Internal sensors (Dead-reckoning)
  - Speed measurement – Tacho-generator, encoder
  - Current measurement

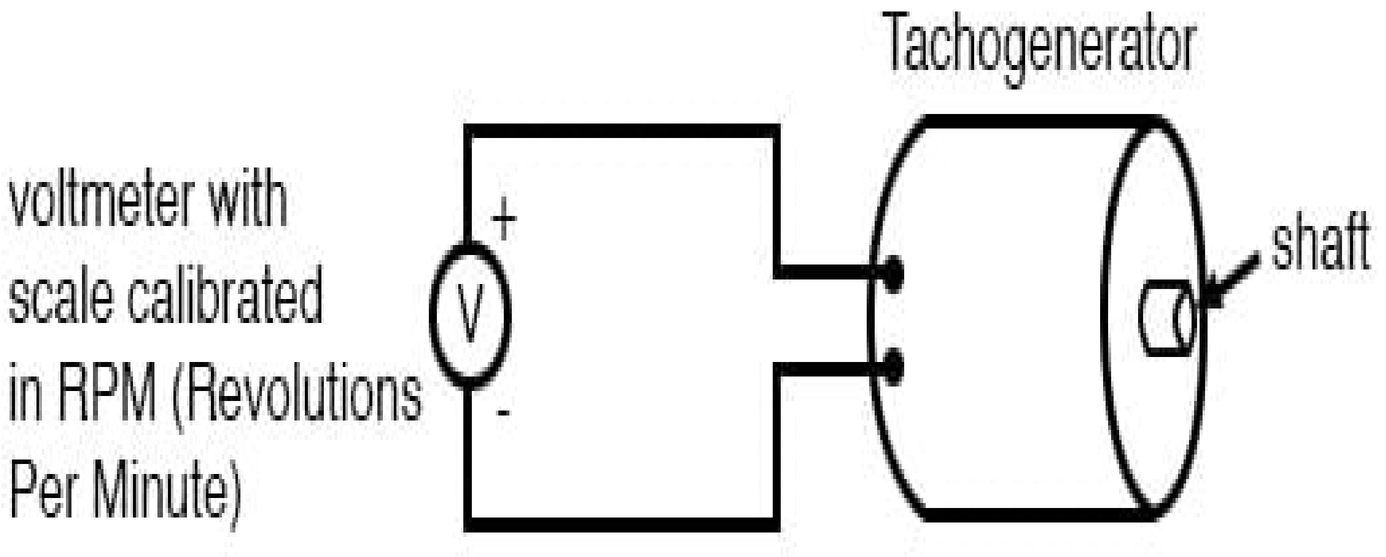
– Wheel Encoders, Odometry
– Accelerometers (spring-mounted masses)
– Gyroscopes (spinning mass, laser light)
– Compasses, inclinometers (earth magnetic field, gravity)

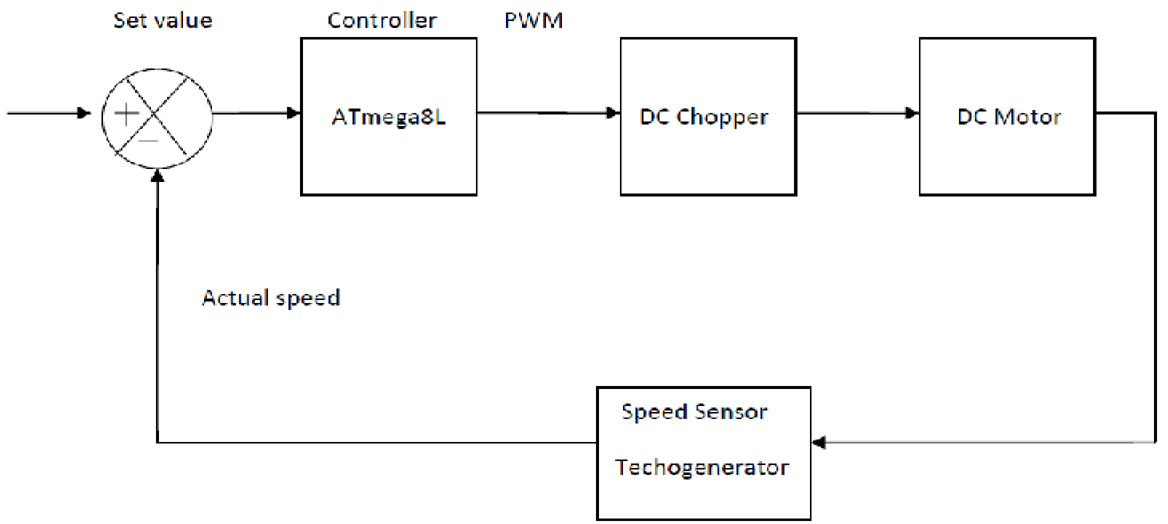
- Proximity sensors

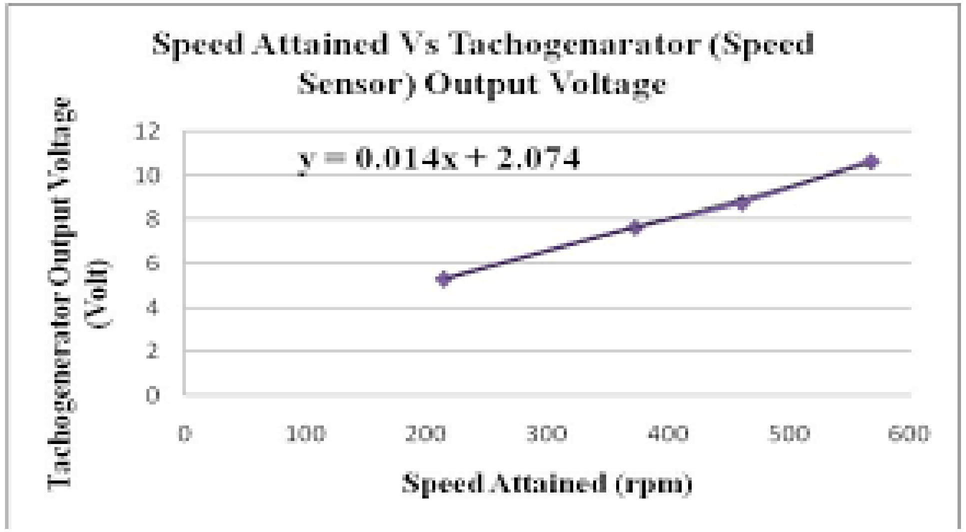
– Ultrasound
– Sonar (time of flight) [SOund Navigation And Ranging]
– Radar (phase and frequency) [RAdio Detection And Ranging]
– LIDAR, Laser range-finders (triangulation, ToF, phase) [LIght Detection And Ranging]
– Infrared (intensity)

- Visual sensors: Cameras
- Satellite-based sensors: GPS

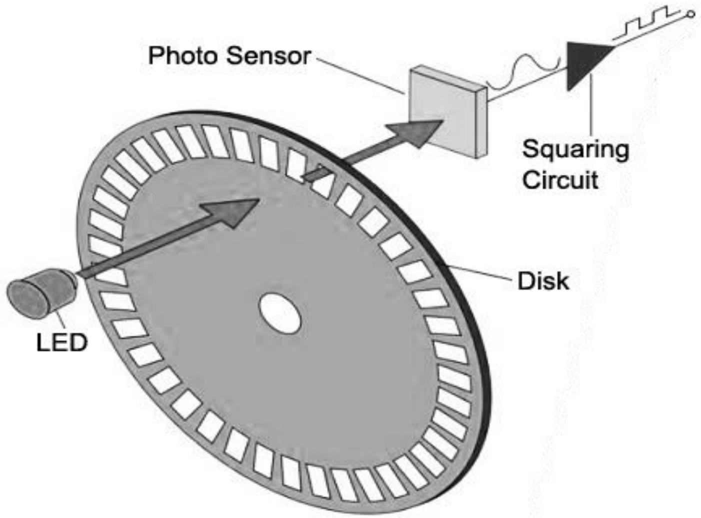
Tacho-generator

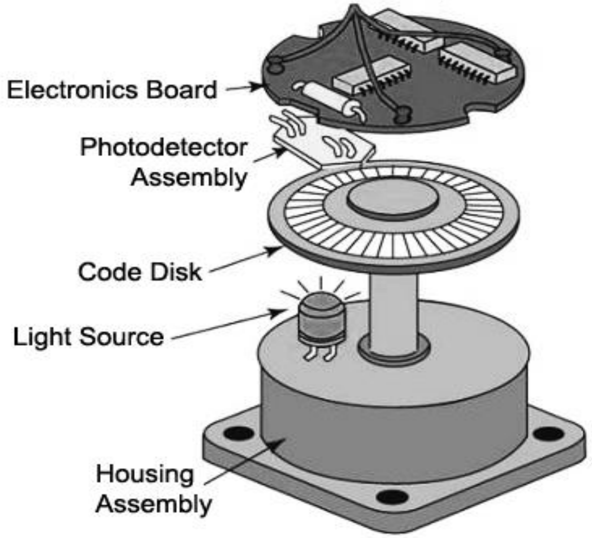


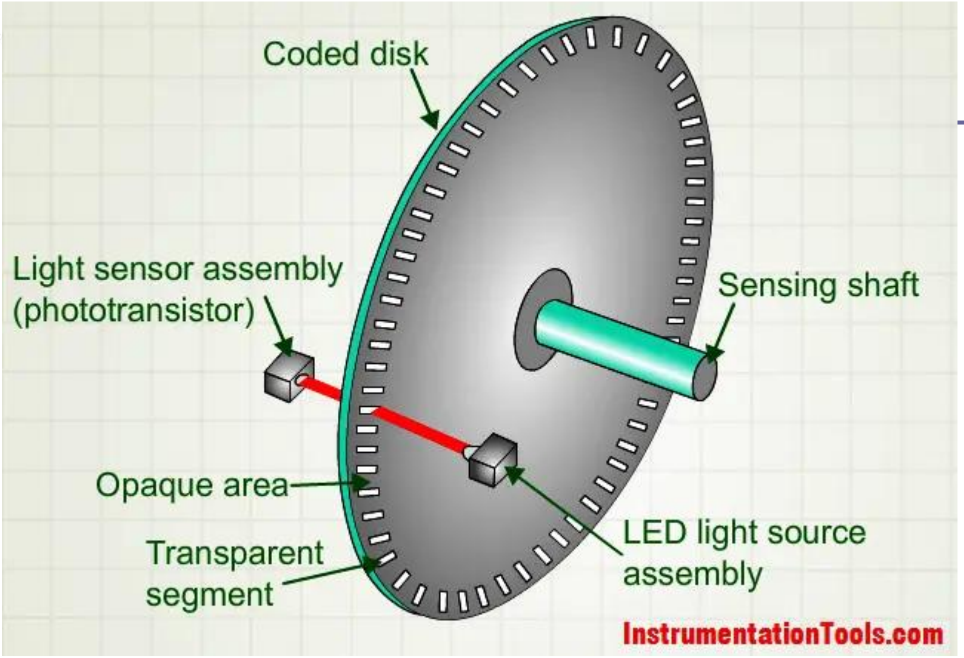


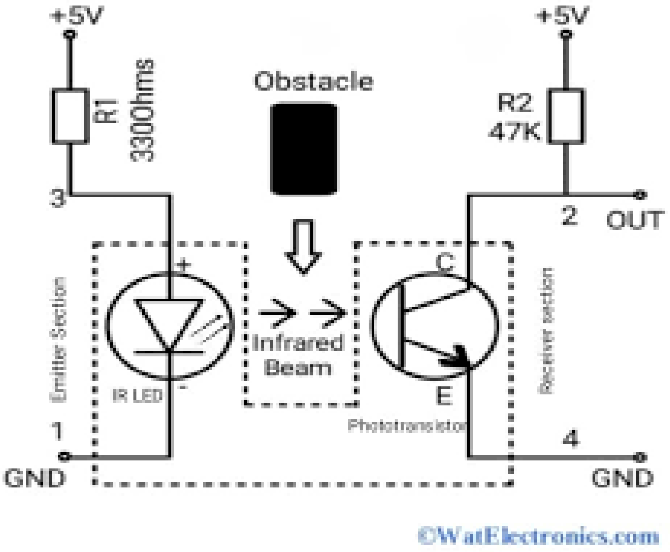


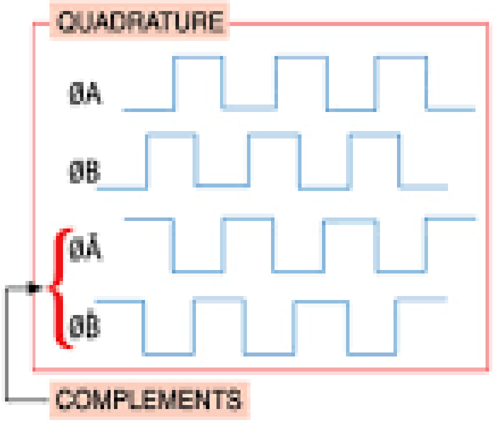
Encoder



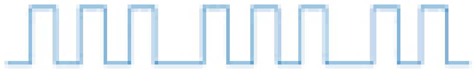






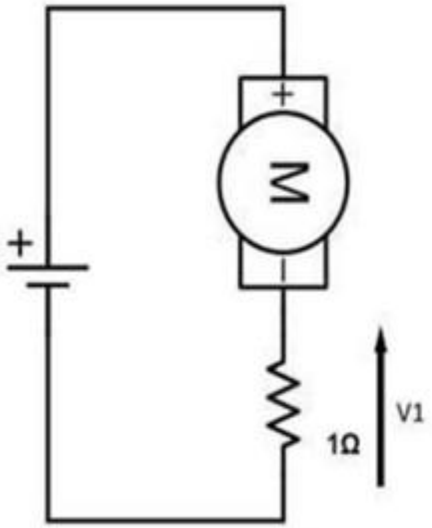


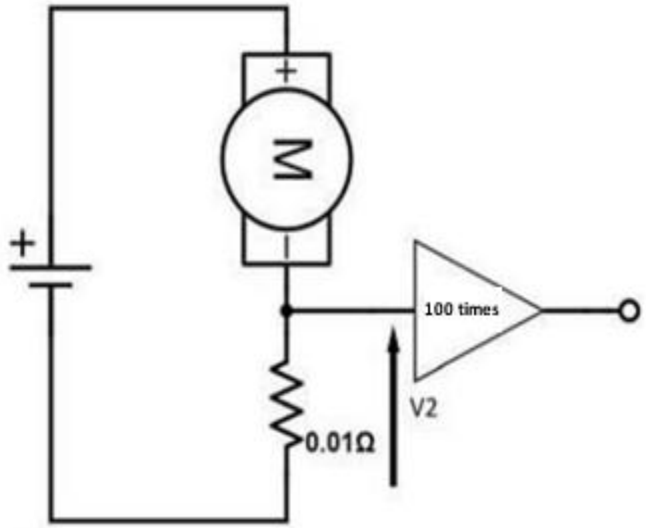
Incremental

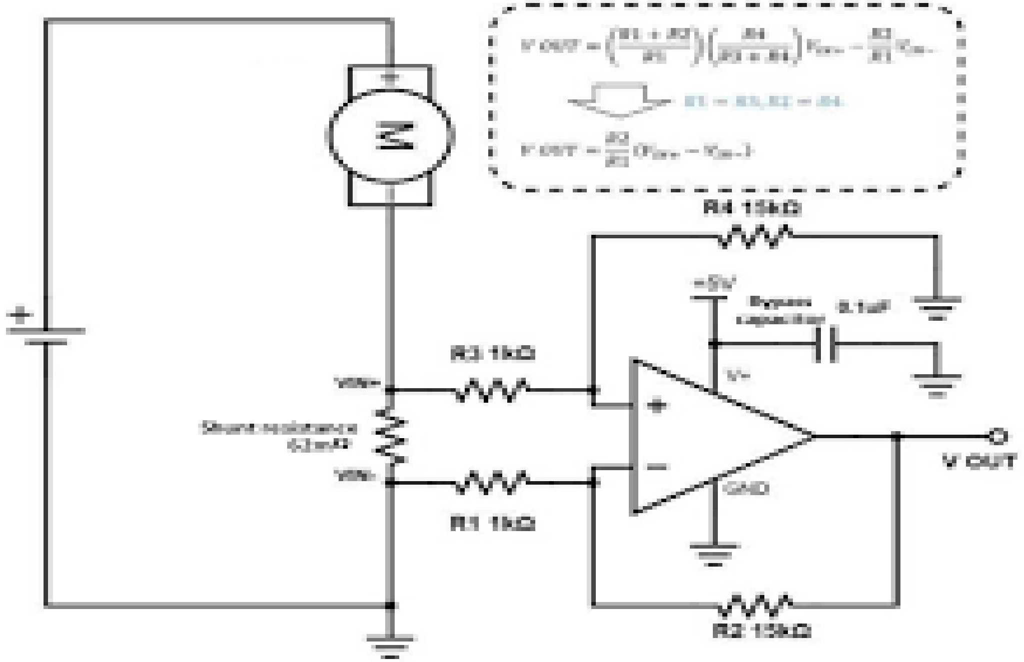


Absolute

Current measurement of DC motor







If you insert a 1Ω resistor and measure the voltage

I=V/R → I=V/1 → I=V

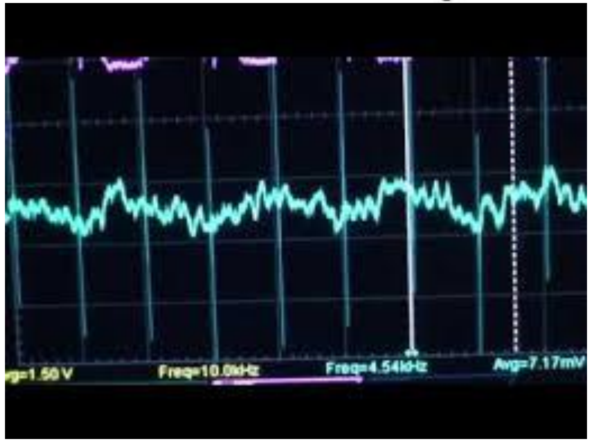
you can measure the voltage of the resistor as a current

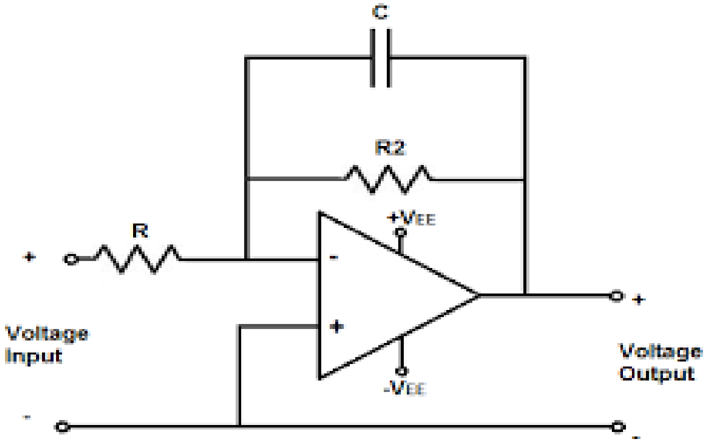
If you reduce the resistance to 1/100 and attach a 100 times amplifier

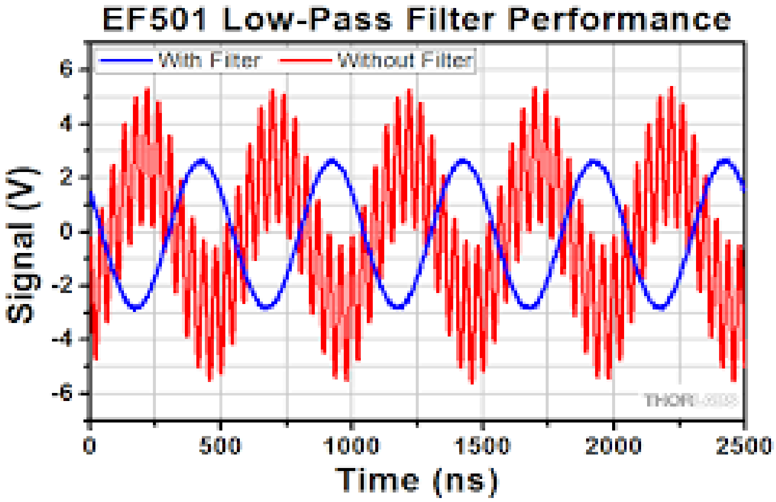
I=(V/(R/100))100→ I=V/1→ I=V

the current can be measured with minimal impact on the circuit

* It is assumed that the same current flows through the two circuits even if the resistance c

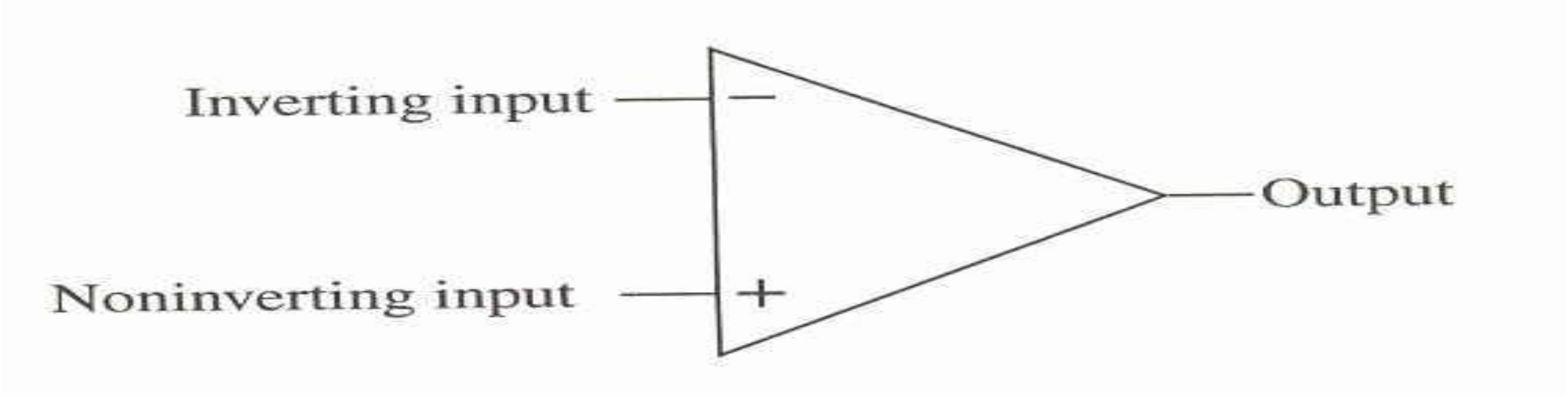






Processing of Sensor signals

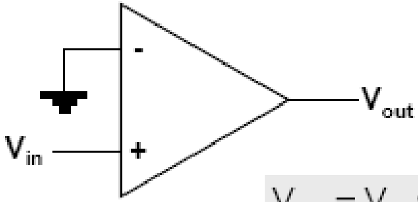
The operational amplifier (Op-amp)



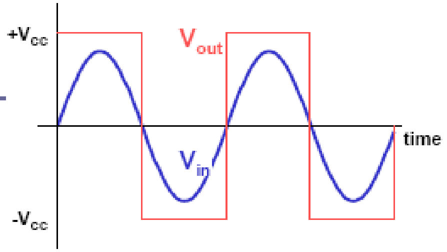
- **Differential voltage gain, Ad**: the amplification of the op-amp of the difference between the two inputs, also known as the **open loop gain**
- In a good amplifier the voltage gain should be as high as possible.
- Gains of 10^6 or higher are common.
- An ideal amplifier should have an infinite gain.

Basic Opamp Configuration

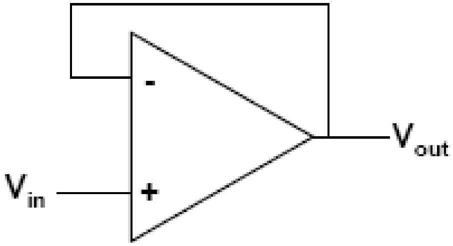
- Voltage Comparator
  - digitize input



$$V_{out} = V_{CC}\,\mathrm{sign}(V_{in})$$

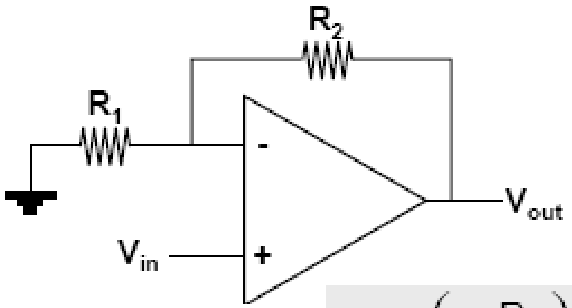


- Voltage Follower
  - buffer



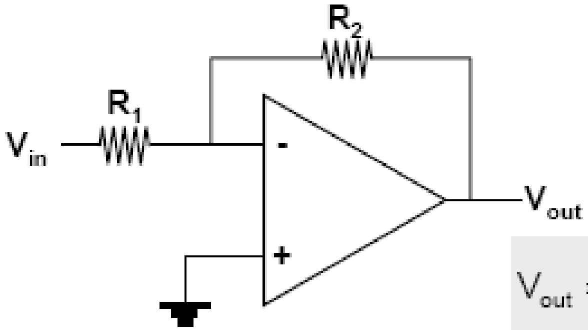
$$V_{out} = V_{in}$$

- Non-Inverting Amp



$$V_{out} = \left(1 + \frac{R_2}{R_1}\right) V_{in}$$

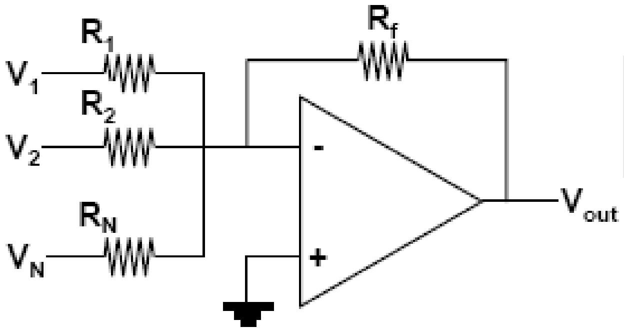
- *Inverting Amp*



$$V_{out} = -\frac{R_2}{R_1} V_{in}$$

More Opamp Configurations

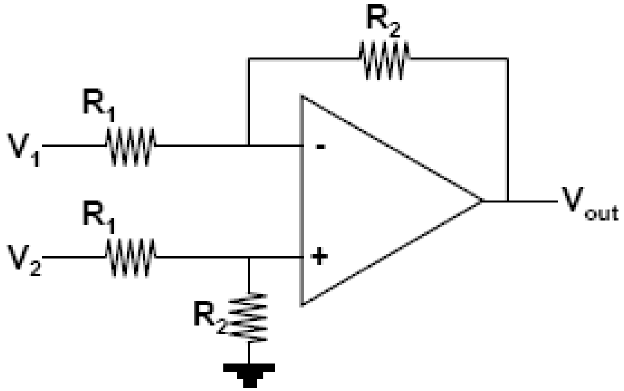
- Summing Amp



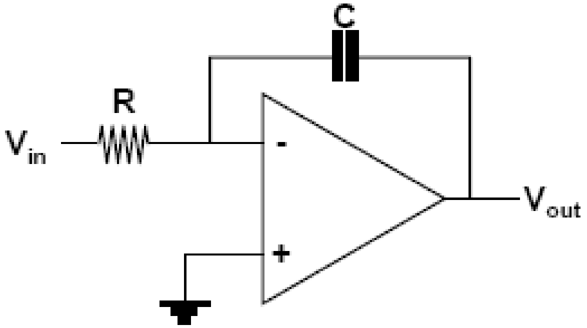
$$V_{out} = -\left(V_1 \frac{R_f}{R_1} + V_2 \frac{R_f}{R_2} + \cdots + V_N \frac{R_f}{R_N} \right)$$

- Differential Amp

$$V_{out} = \frac{R_2}{R_1} \left(V_2 - V_1 \right)$$



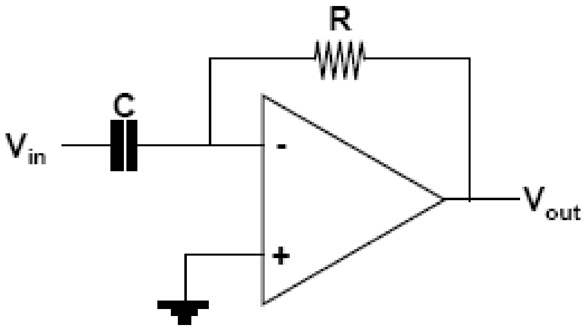
- Integrating Amp



$$V_{out} = -\frac{1}{j\omega CR} V_{in} = -\frac{1}{RC} \int V_{in} dt$$

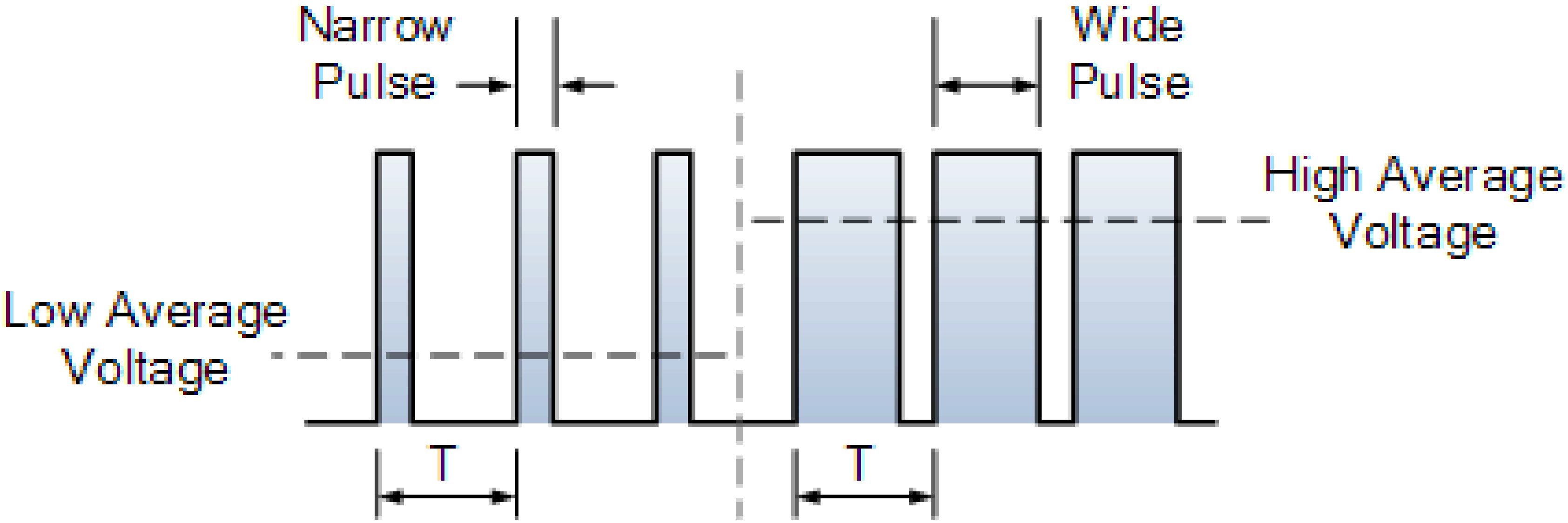
- Differentiating Amp

$$V_{out} = -\frac{R}{\frac{1}{j\omega C}} V_{in} = -RC \frac{dV_{in}}{dt}$$



Speed Control using Pulse Width Modulation (PWM)

- Used for ***efficient*** DC motor speed control.
- A PWM circuit works by generating a square wave with a variable on-to-off ratio.
- The average 'on' time may be varied from zero to 100%.
- A variable amount of voltage is applied across the motor, thus power is transferred to the load (motor).



Speed Control using Pulse Width Modulation (PWM)

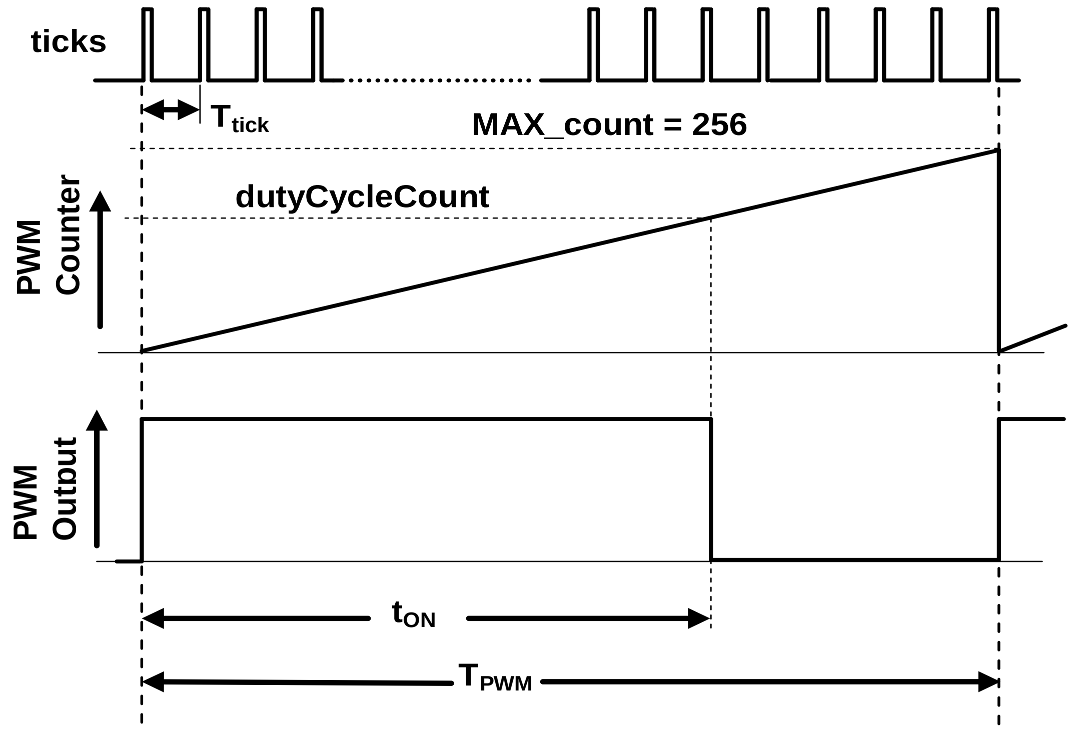
- The high value is held during a variable pulse width ***t*** during the fixed period ***T*** (Frequency = **1/*T***)
- The resulting waveform has a duty ratio (or duty cycle), defined as the ratio between the ON time and the period of the waveform, usually specified as a percentage.

$$duty ratio = \frac{t}{T} \times 100\%$$

- In PWM control, the voltage is switching rapidly across the armature
  - fixed period and amplitude (peak value)
  - variable duty cycle
- However, due to the motor inductance and resistance, the resulting current through the motor has a small fluctuation around an average value.
- As the duty cycle gets larger, the average voltage gets larger and the motor speed increases.

Generating PWM Signal

- **Use Timer 0 in Auto-reload mode so that it overflows at a regular interval and generates an interrupt (software *'tick'*)**
- **The *PWM_Counter* is incremented in the Timer 0 Interrupt Service Routine (ISR)**
- **When the *PWM_Counter* value exceeds *dutyCycleCount*, the *PWM_Output* is reset to 0**
- **When the *PWM_Counter* value exceeds *MAX_count*, the *PWM_Output* is set to 1 and *PWM_Counter* is re-initialized to 0**
- **Duty Cycle can be changed by altering the value of *dutyCycleCount***



Timer 0 ISR

```
void Timer0_ISR (void) interrupt 1
{
          TF0 = 0;   //-- clear TF0 overflow flag

          PWM_counter++;
          if (PWM_counter >= dutyCycleCount)
                    PWM_output = 0;

          if (PWM_counter >= MAX_Count)
          {
                    PWM_output = 1;
                    PWM_counter = 0;
          }
}
```

- 0 <= *dutyCycleCount* <= 256
- Resolution of the duty cycle is 1/256 (approx. 0.39%).
- PWM output is at one of the digital output port pin, example P0.4
- The PWM output pin must be configured in push-pull mode.

PWM Frequency: Timer 0 reload value

- For a desired PWM frequency, what should be the '*tick time*' (**Ttick**)?

$$T_{PWM} = T_{tick} \times 256$$

- If $\boldsymbol{T_{tick}}$ **= 10μs**, then

$$T_{PWM} = 10 \times 256 \ \mu s$$

$$f_{PWM} = \frac{1}{T_{PWM}} = \frac{1}{10 \times 256 \times 10^{-6}} \approx 390 \text{ Hz}$$

- Changing the value of $\boldsymbol{T_{tick}}$ will change the PWM frequency
- To produce an interrupt at every 10 μs, what should be the reload value of Timer 0?

PWM Frequency: Timer 0 reload value

- Using a System Clock of 22.1184 MHz-

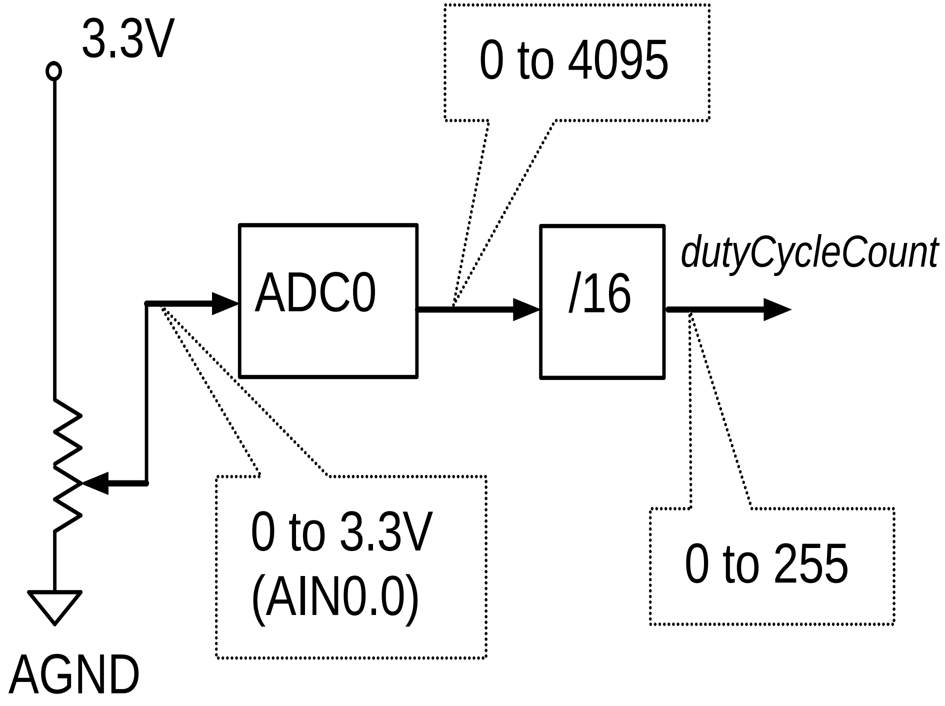
$$T_{sysclk} = \frac{1}{22118400 Hz} \approx 0.04521\ \mu s$$

$$\#\text{Sysclk pulses} = \frac{10 \mu s}{0.04521 \mu s} \approx 221$$

- Timer 0 reload value = 255 – 221 = 34

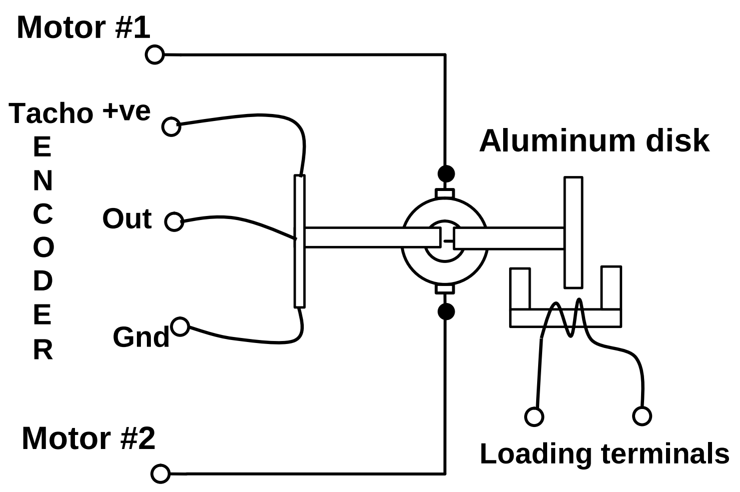
Varying the PWM Duty Cycle

- The PWM ON time is changed by changing the value of *dutyCycleCount*, which must be between 0 and 255.
- Use the potentiometer on the microcontroller development board which is connected to the Analog Input (AIN0.0) of ADC0.
- The ADC0 output is a 12-bit data (0 to 4095).
- In the program, read the ADC0 output and divide by 16.



Tacho Output and Loading Mechanism

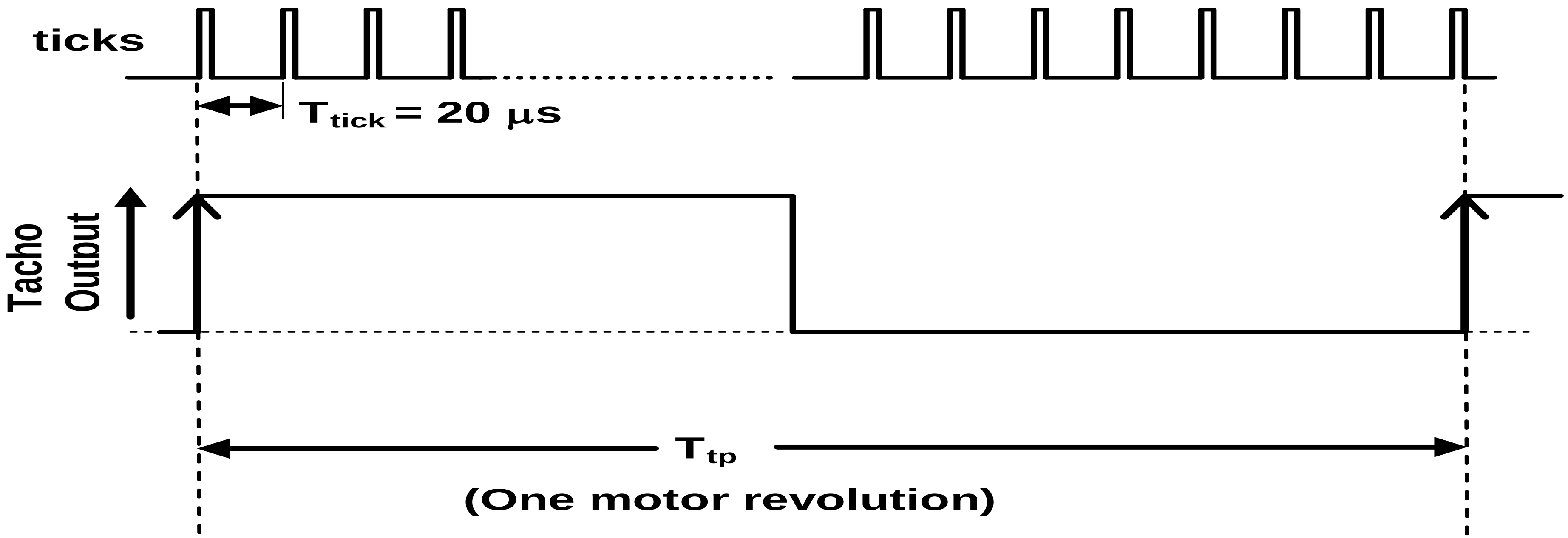
- For motor speed control, a reference speed is set (usually in rpm)
- The digital speed controller (implemented in software) employs control algorithms to maintain the actual motor speed as close as possible to the reference speed.
- The controller needs to know the actual motor speed so that if there is an error between the actual speed and the reference speed, it can take suitable corrective action to minimise the error in speed.



- **Tacho Encoder**
  - Hall-effect device.
  - Supply: +5V DC
  - Output is a square pulse of 50% duty cycle.
  - One pulse per motor revolution.
    - Slower the motor, larger the time period of the Tacho pulse.

Measuring Actual Motor Speed

- General principle for measuring actual motor speed-
  - For one Tacho pulse (say, between two rising edges) count the number of a fast-occurring ticks (overflows) of a programmed timer.



Measuring Actual Motor Speed

- Use Timer 1 in auto-reload mode to generate the software interrupts (ticks) at 20 μs interval.
- Calculate the Timer 1 reload value for a System Clock of 22.1184 MHz.
- **N** : number of ticks counted for one motor revolution.

$$Actual\ Motor\ Speed \propto \frac{1}{N}$$

$$Actual\ Motor\ Speed = \frac{K}{N}$$

where,

K is the proportionality constant

Calculating the value of *K*

$$Actual\ Motor\ Speed = \frac{K}{N}$$

$$Actual\ Motor\ Speed\ in\ RPS = f_{tp} = \frac{1}{T_{tp}}$$

$$Actual\ Motor\ Speed\ in\ RPM = \frac{60}{T_{tp}} = \frac{60}{N * T_{tick}}$$

$$\therefore \frac{K}{N} = \frac{60}{N * T_{tick}}$$

$$\therefore K = \frac{60}{T_{tick}}$$

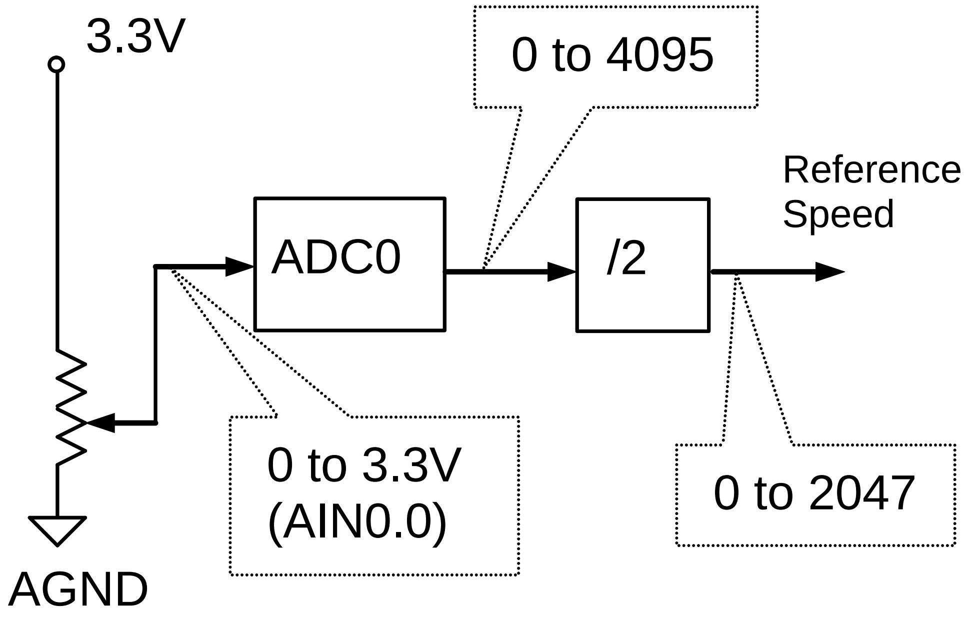
- If T_{tick} = 20 µs, K = 3,000,000

Counting *N* (number of ticks for one revolution)

- For every ***tick***, increment a counter (let us call it ***TACHO_counter***).
  - This can be done in the ISR of Timer 1.
- Use the Tacho encoder pulse to generate an external hardware interrupt at **/INT1** (P0.3)
- Count the number of ***tick***s between two successive Tacho interrupts; this is **N**.
- Implementation of Timer 1 and /INT1 ISRs are exercises which form part of your Lab Assignment

Setting Motor Reference Speed

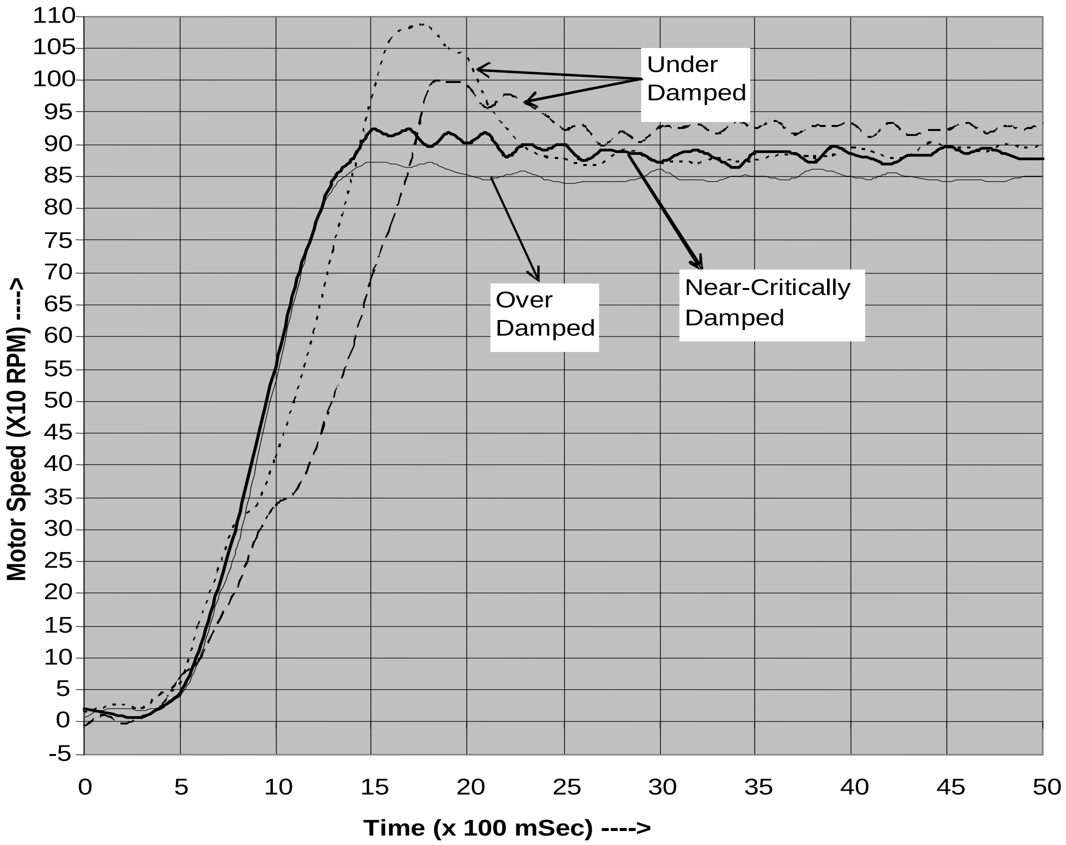
- In a digital speed controller, the controller computes the duty cycle of the PWM signal to make the motor run at the reference speed.
- We need to input the reference speed to the controller.
- This can be done by using the potentiometer on the expansion board which is connected to the Analog Input (AIN0.0) of ADC0.
- The digital output of the ADC0 can be a measure of the reference speed in RPM.
- You may want to clamp the Reference Speed to a range of 500 rpm to 2000 rpm.



Recording Transient behaviour of Motor

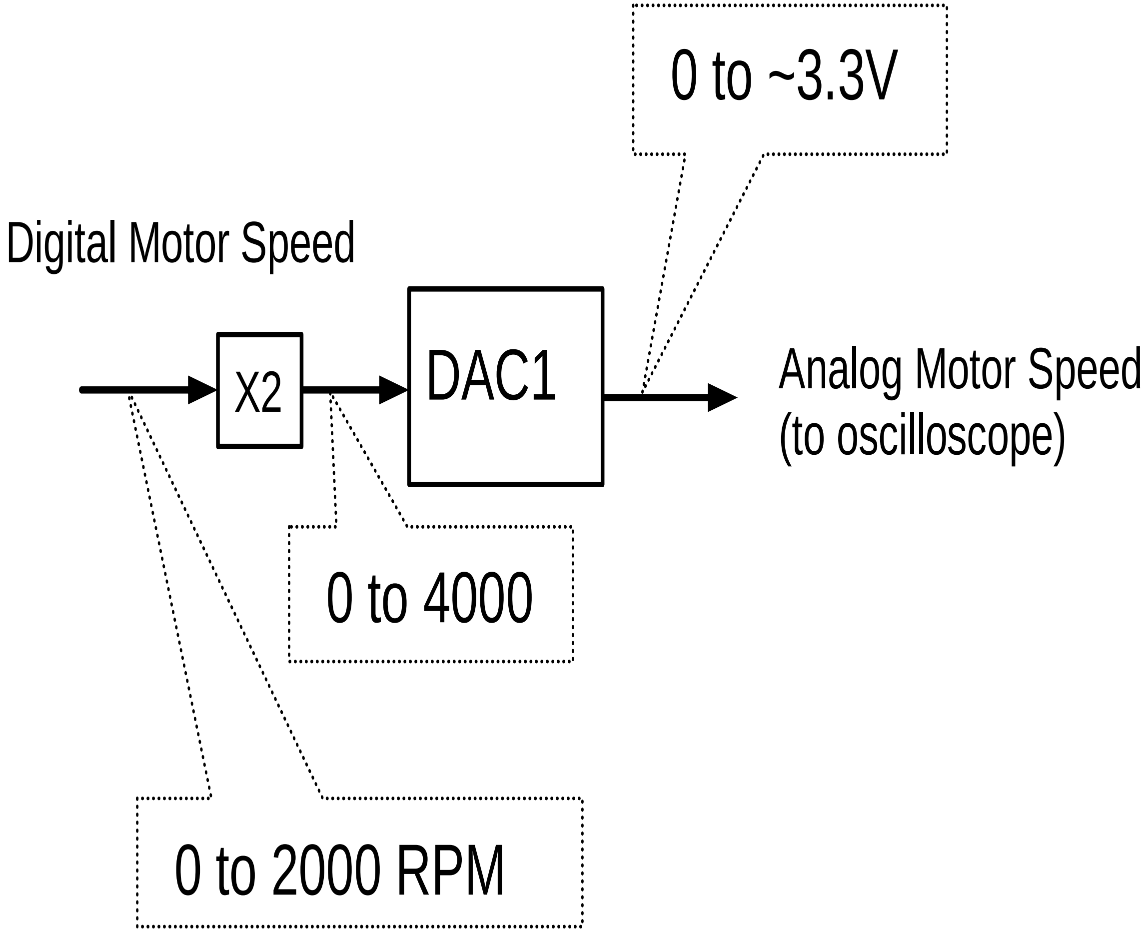
- The digital speed controller calculates the actual speed of the motor which can be displayed on the LCD.
- The actual motor speed is continuously changing, even when the system has reached a steady state and so called stabilised.

- Sample Response

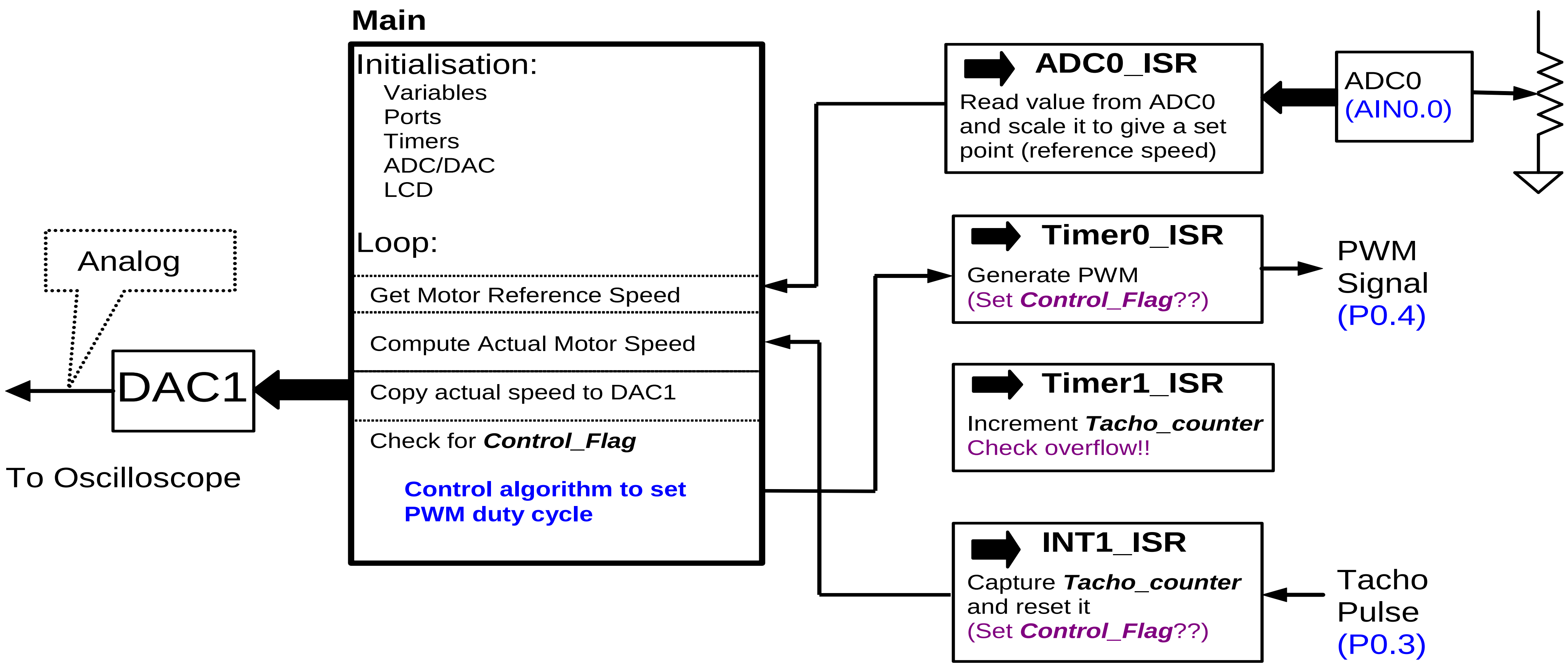


Displaying Actual Motor Speed as an analog voltage on oscilloscope

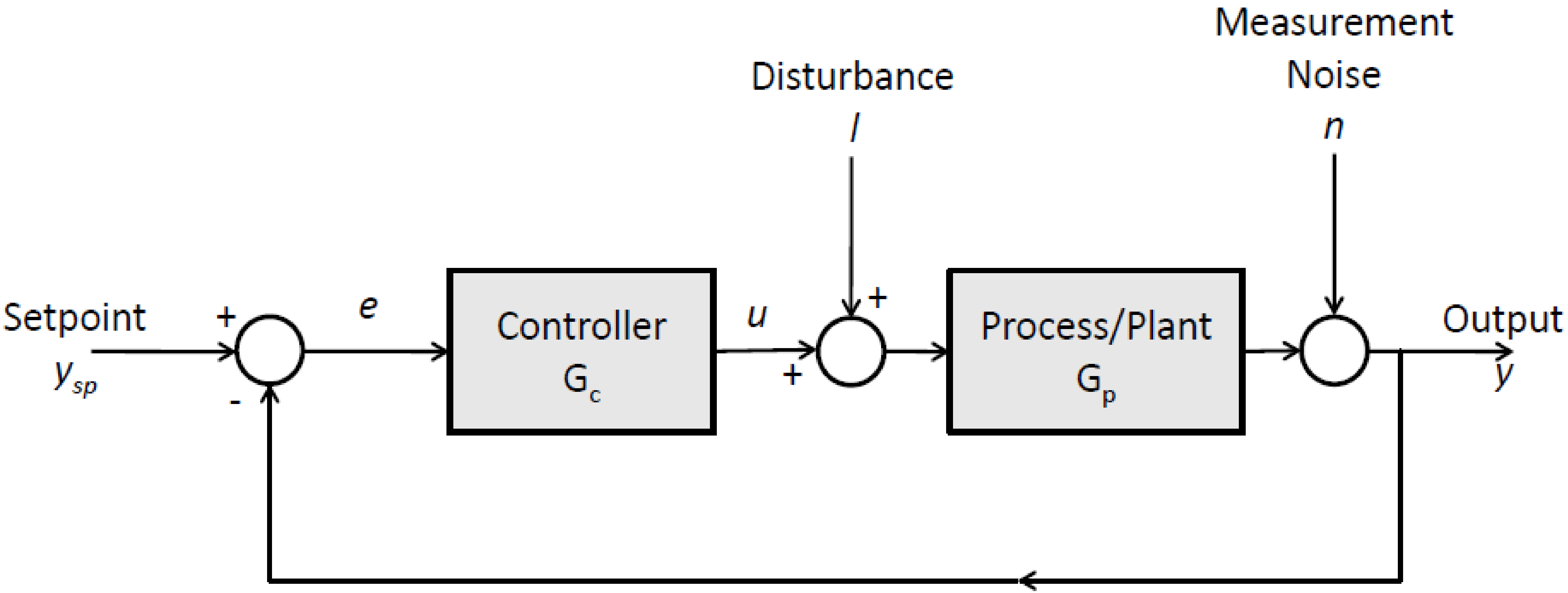
- Send the actual motor speed (which is expected to be in the range of 0 to 2000 RPM) to the DAC1.
- The DAC1 analog output will be a measure of the actual motor speed.
- Since it is a 12-bit DAC, the input can be in the range of 0 to 4095. Hence the actual motor speed may be multiplied by 2 before presenting at the DAC1.



Program Structure



Feedback Control System



Classical Controller Structure

- A particular structure commonly used is PID

  PID stands for: **P** (*Proportional*)
  I (*Integral*)
  D (*Derivative*)

- These controllers have proven to be robust and useful in may different applications – used in over 90% of industrial control systems

Transfer Function

$$G_c(s) = K_p + \frac{K_I}{s} + K_D s$$

One pole at zero
Two zeros anywhere

proportional

integral

derivative

$$u(t) = K_p e(t) + K_I \int e(t)dt + K_D \frac{de(t)}{dt}$$

Alternative form:

$$u(t) = K_c \left[e(t) + \frac{1}{T_i} \int_0^t e(\tau)\, d\tau + T_d \frac{de(t)}{dt} \right]$$

Variations

Controller	PID Gain	Transfer Function
Proportional	$K_I = 0$, $K_D = 0$	K_p
Proportional plus Integral	$K_D = 0$	$K_p + \frac{K_I}{s}$
Proportional plus Derivative	$K_I = 0$	$K_p + K_D s$

Effects of PID Gains on Step Input

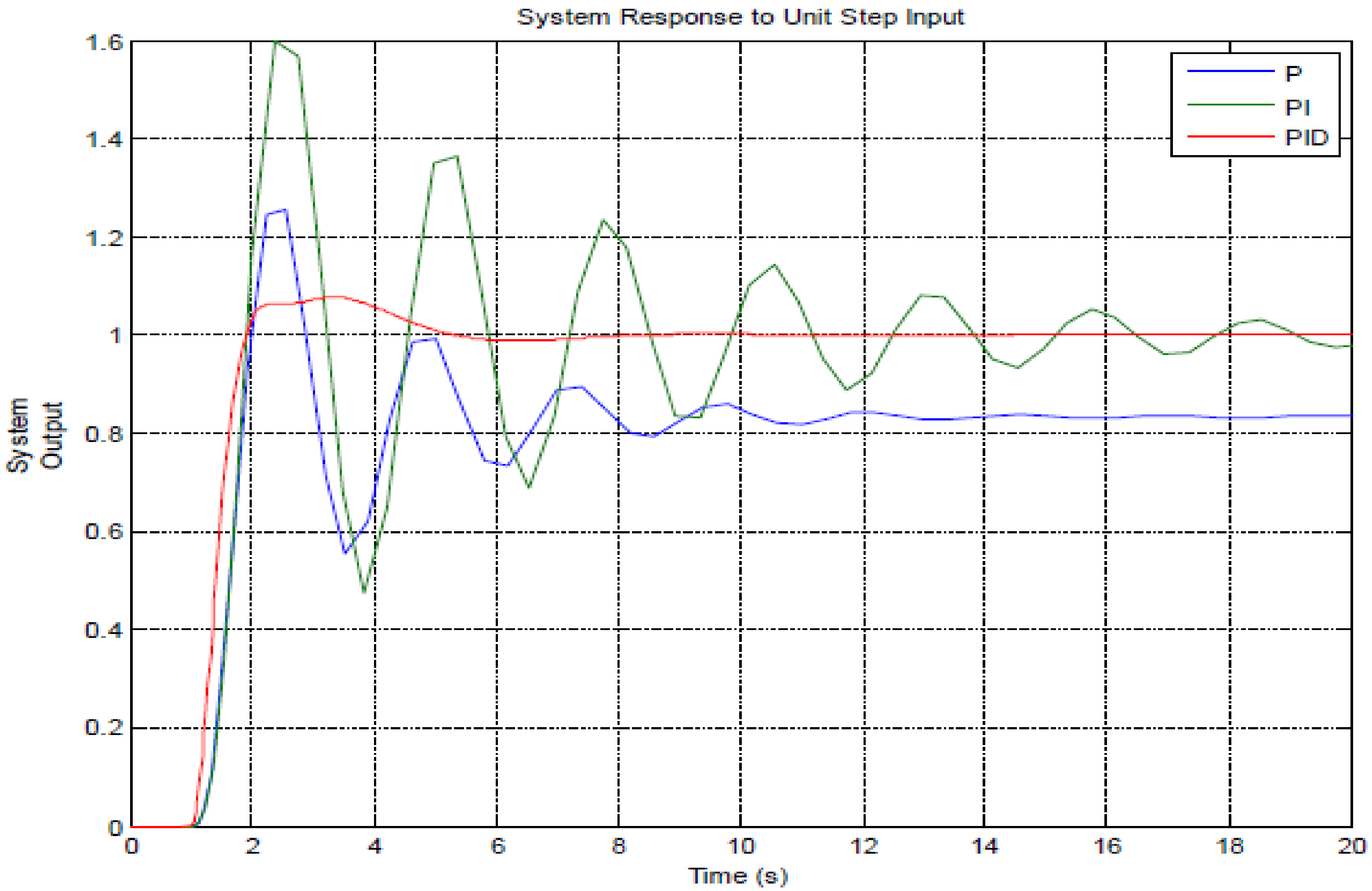
PID Gain	Percent Overshoot	Settling Time	Steady State Error
Increasing K_p	Increases	Minimal Impact	Decreases
Increasing K_I	Increases	Increases	Zero error
Increasing K_D	Decreases	Decreases	No impact

Design of a PID controller involves determining suitable values of the PID gains -- a process called PID tuning

Example

- Consider a feedback control system with H=1 and
$$G(s) = \frac{10}{s^3 + 6s^2 + 11s + 16}$$
- Specifications:
  - Zero steady state error
  - Settling time within 5s
  - Rise time within 2s
  - Minimal overshoot

Comparison of Controllers



P: $K_p = 3$

PI: $K_p = 2.7$
$K_I = 1.8$

PID: $K_p = 2$
$K_I = 2.2$
$K_D = 1.2$

PID Tuning

- A variety of methods are available
- Manual tuning – trial and error
- Ziegler-Nichols tuning method
  - Developed by John Ziegler and Nathaniel Nicols in 1942
  - Based on certain forms of the process model
  - Process model does not have to be precisely known
  - Will not work for all processes

Ziegler Nicols PID Tuning

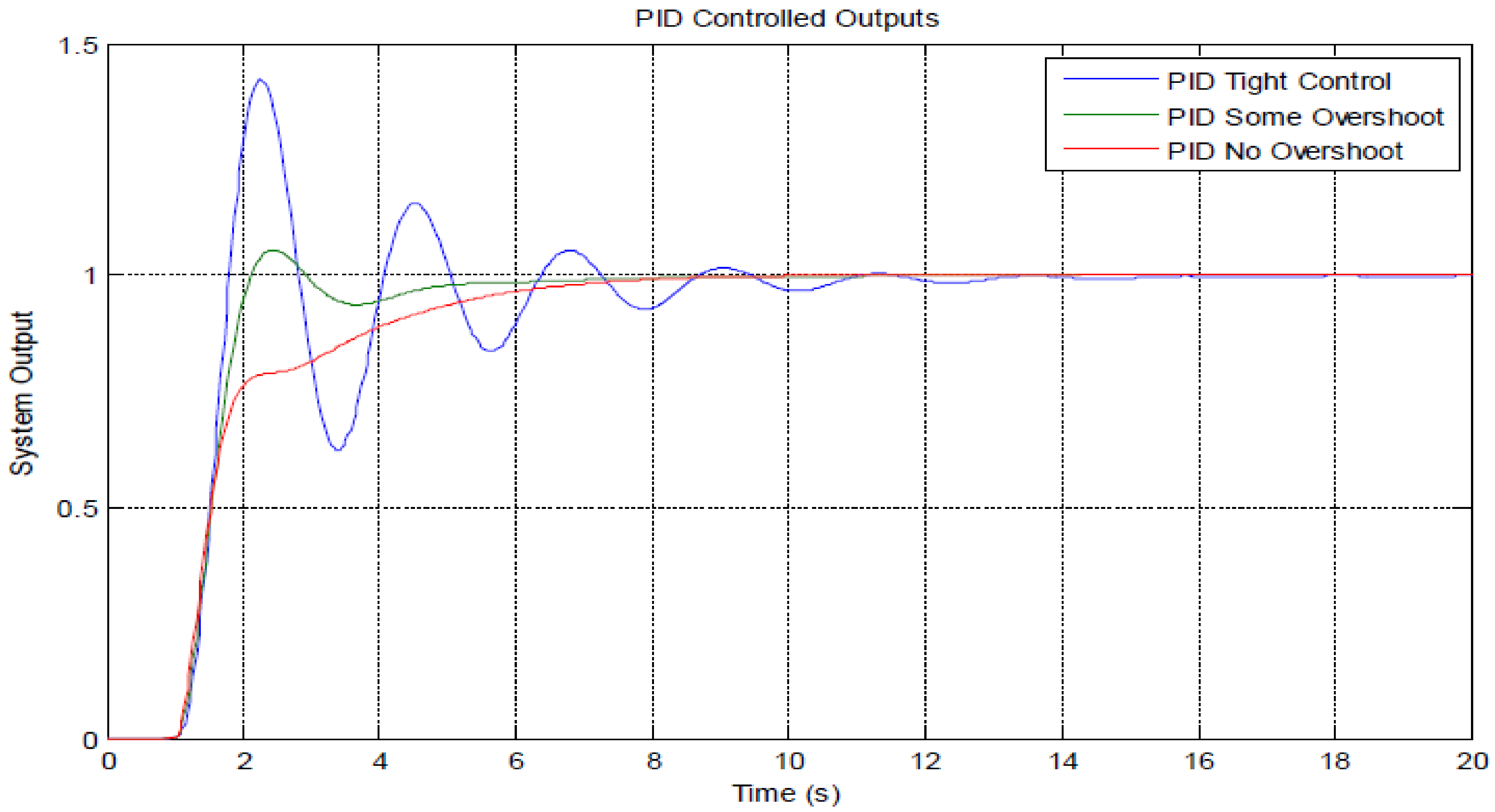
1. Set K_I and K_D to zero
2. Increase K_P until the system is at the boundary of instability (sustained oscillation)
3. The value of K_P at this point is called the ultimate gain K_U
4. The period of the oscillation is called the ultimate period T_U
5. Use the following table

Ziegler Nicols PID Tuning

Controller Type	K_P	K_I	K_D
P	$0.5\ K_U$		
PI	$0.45\ K_U$	$0.833\ T_U$	
PID tight control	$0.6\ K_U$	$0.5\ T_U$	$0.125\ T_U$
PID some overshoot	$0.33\ K_U$	$0.5\ T_U$	$0.33\ T_U$
PID no overshoot	$0.2\ K_U$	$0.3\ T_U$	$0.5\ T_U$

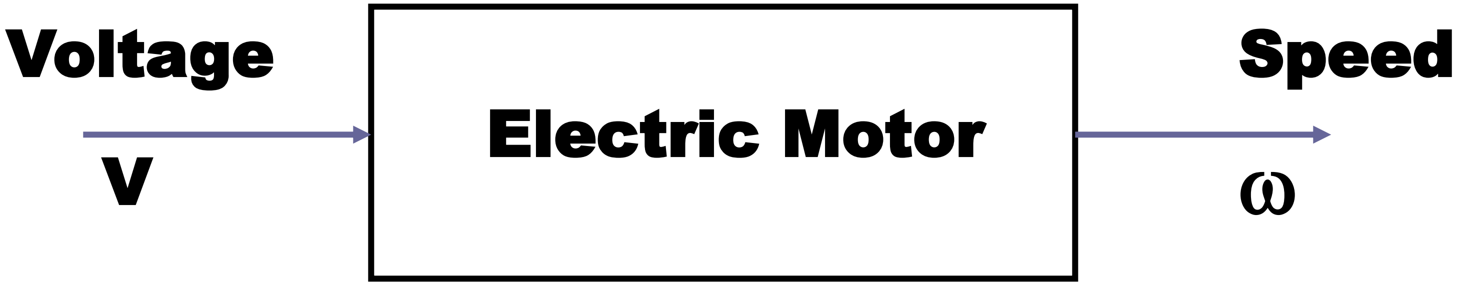
Additional fine tuning is usually required to obtain the best performance.

Ziegler Nichols Tuned Controllers



Mathematical models for systems

Mathematical models for systems is important to understand the behavior of them. The model is a replica of the relationships between the input and output or inputs and outputs. The actual relationship that exist between the input and output of a system have been replaced by mathematical expressions.

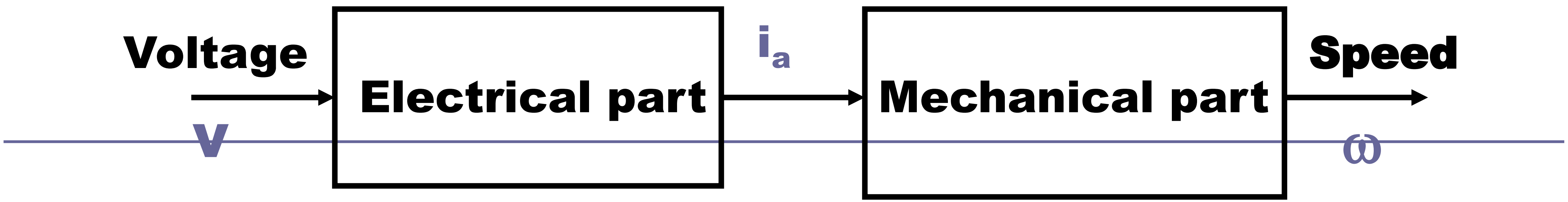


Approximately, $\omega \propto V$

Or, $\omega = GV$

Model >>

Dynamics of Electrical motor

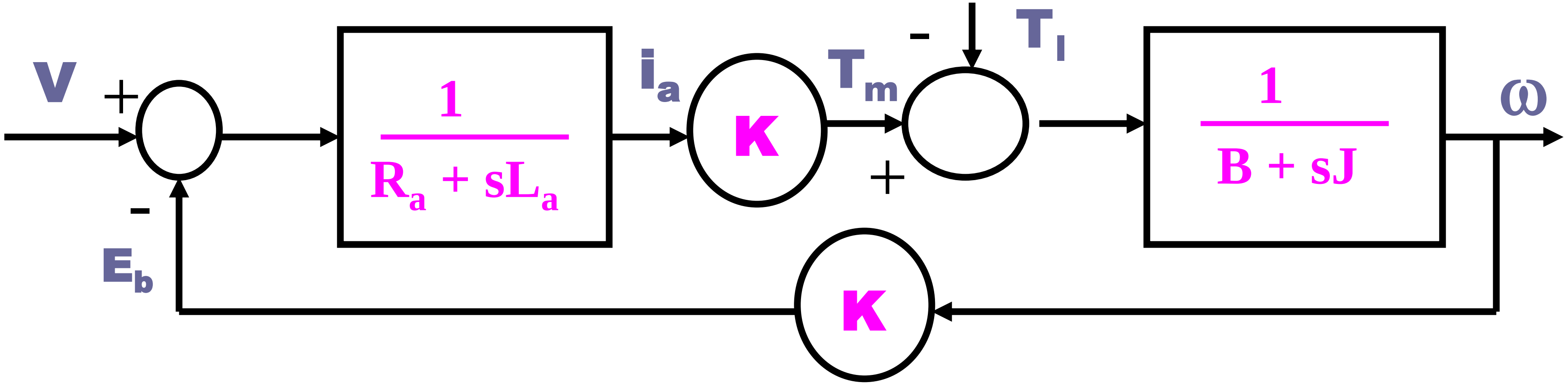


Electric circuit equation: $V = E_b + R_a i_a + L_a \, di_a/dt$

Back e.m.f. equation: $E_b = K\omega$

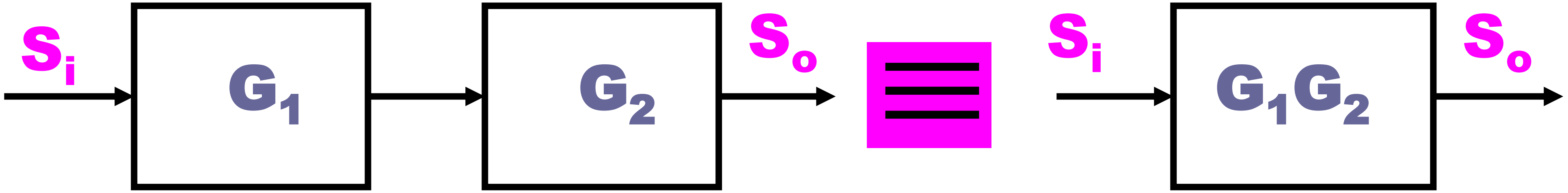
Torque equation: $T_m = K i_a$

Mechanical equation: $T_m = T_l + B\omega + J \, d\omega/dt$

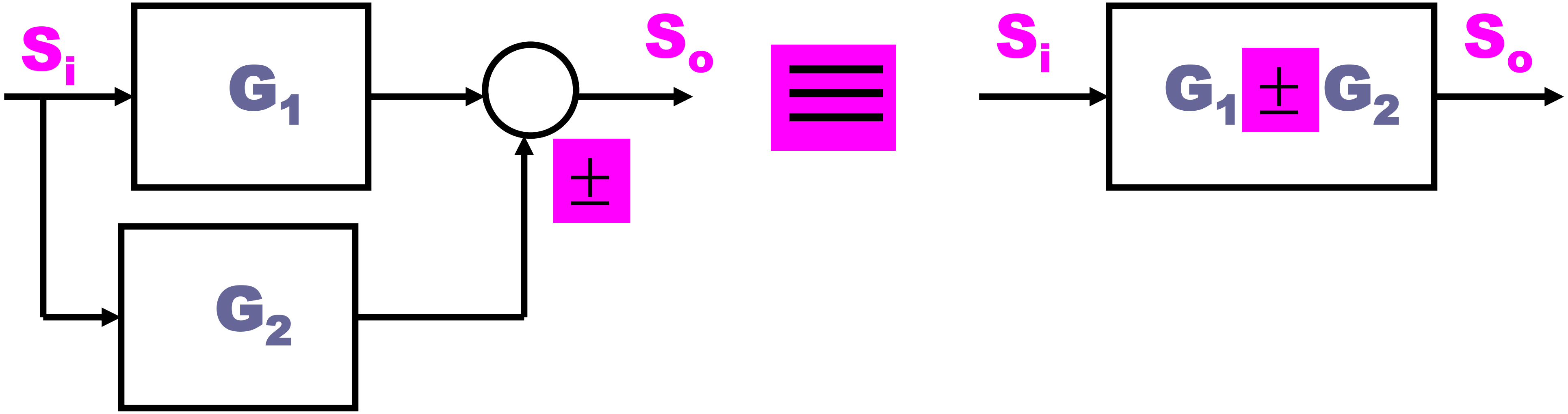


Operations on Blocks

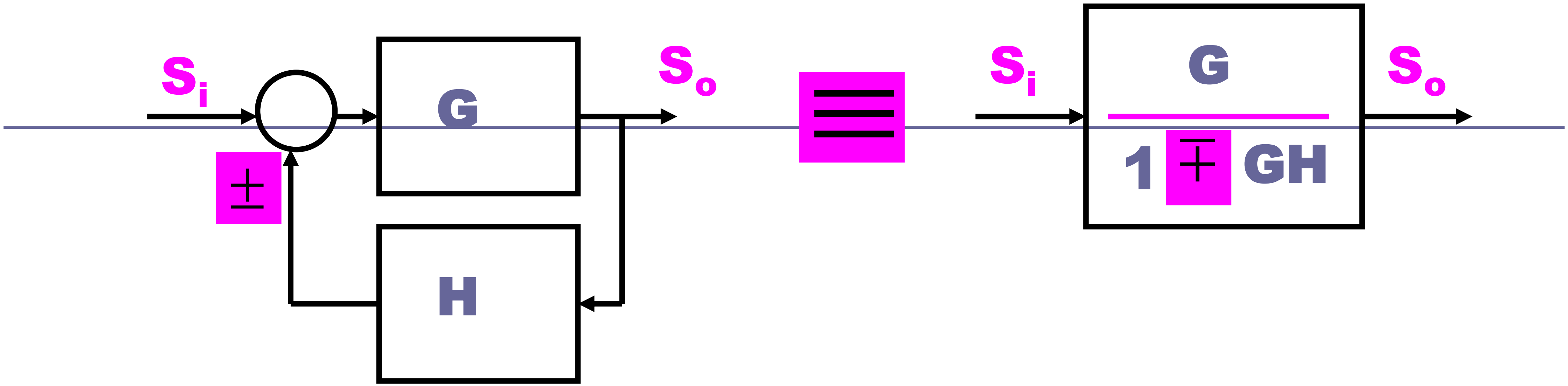
1. Cascading blocks



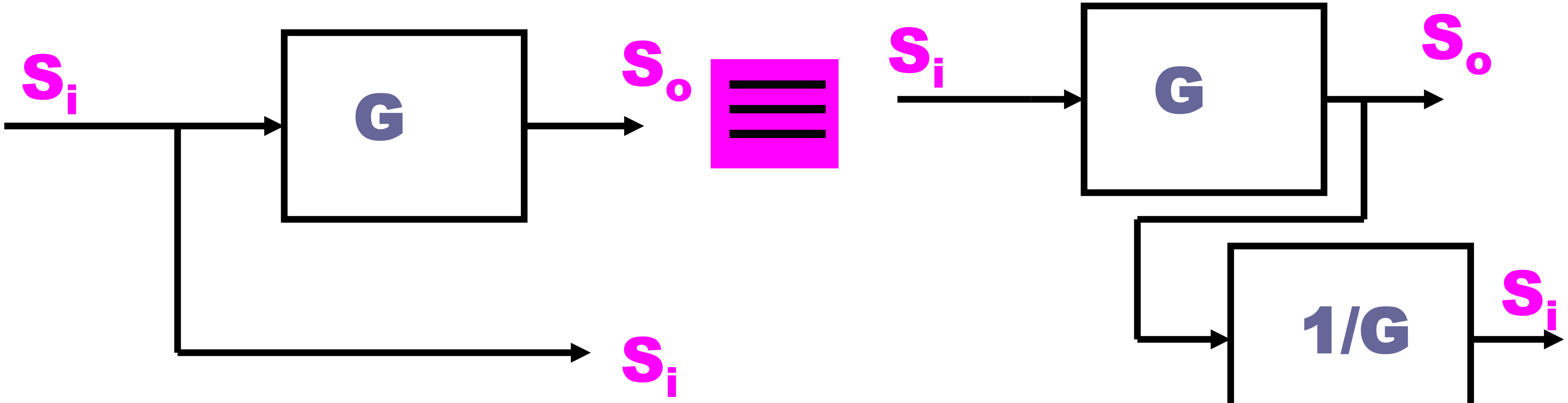
2. Elimination of a forward loop



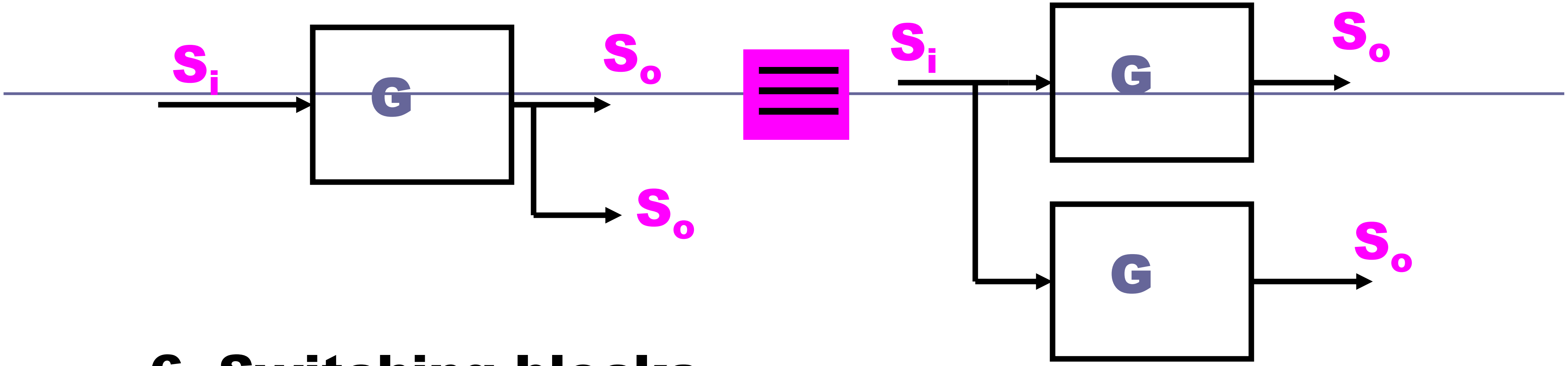
3. Elimination of a feedback loop



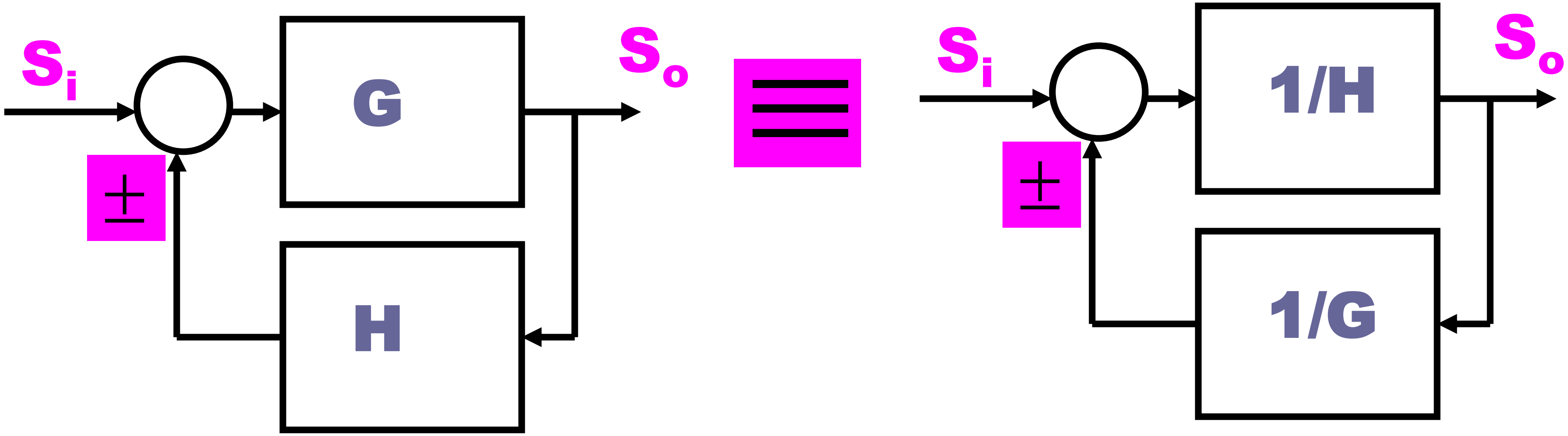
4. Moving a take-off point from an input to an output of a block



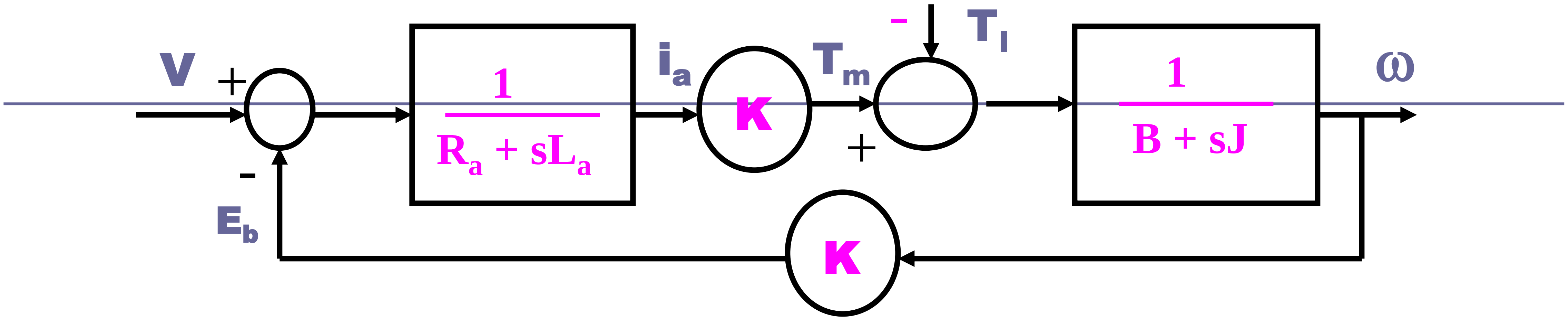
5. Moving a take-off point to a block ahead



6. Switching blocks



Transfer function of DC motor



$$\left.\frac{\omega}{V}\right|_{T_l=0} = \frac{G}{1+GH} = \frac{\dfrac{K}{(R_a+sL_a)(B+sJ)}}{1+\dfrac{K^2}{(R_a+sL_a)(B+sJ)}}$$

$$= \frac{K}{K^2 + R_a B(1+s\tau_a)(1+s\tau_m)}$$

$;\tau_a$ = L_a/R_a

τ_m= J/B

TRANSFER FUNCTION (T.F.)

T.F. which describes the dynamic behavior of the system is defined as the ratio of the Laplace transform of the output variable to the Laplace transform of the input variable, with all initial conditions assumed to be zero.

How to get T.F.?

1. Obtain the mathematical (differential) equation describing the relationship between input and output or inputs and outputs.
2. Usually the equations are dictated by certain laws or principles. For example, for mechanical system it is Newton's law of motion. For electrical system it is Kirchhoff's laws etc.

Modeling in State Space

Modern Control Theory (MCT) versus Conventional Control Theory (CCT)

1. MCT is applicable to Multiple-Input-Multiple-Output (MIMO) system, which may be linear or non-linear, time-invariant or time varying. CCT is only applicable to Linear Time Invariant (LTI) Single Input Single Output (SISO) system.

2. MCT is essentially a time domain approach while CCT is a complex frequency- domain approach.

A few definitions:

State: The state of a dynamic system is the smallest set of variables (called state variables) such that the knowledge of these variables at $t = t_o$, together with the knowledge of the input for $t >= t_o$, completely determines the behavior of the system for any time $t >= t_o$.

State variables: The state variables of a dynamic system are the variables making up the smallest set of variables that determine the state of the dynamic systems.

A set of n simultaneous, first order differential equations with n variables, where the n variables are to be solved are the state variables.

The state variables need not be physically measurable or observable quantities.

State vector: If n state variables are needed to completely describe the behavior of a given system, then these n state variables can be considered the n components of a vector x. Such a vector is called a state vector.

State space: The n-dimensional vector space whose coordinate axes consists of the x_1 axis, x_2 axis, … x_n axis where x_1, x_2 … x_n are state variables, is called a state space. Any state can be represented by a point in the state space.

State-space equations: Three types of variables are involved in the modeling of dynamic system: input variables, output variables and state variables.

Output equations: The algebraic equations that expresses the output variables of a system as linear combinations of the state variables and the inputs.

A system is represented in state space by the following equations:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

x = state vector, **y** = output vector, **u** = input vector, **A** = system matrix, **B** = input matrix, **C** = output matrix, **D** = feedforward matrix.

State-space representation of DC motor

$L_a \, di_a/dt = - R_a i_a - K\omega + V$

$J \, d\omega/dt = K i_a - B\omega - T_l$

$$\begin{bmatrix} \dfrac{di_a}{dt} \\ \dfrac{d\omega}{dt} \end{bmatrix} = \begin{bmatrix} -\dfrac{R_a}{L_a} & -\dfrac{K}{L_a} \\ \dfrac{K}{J} & -\dfrac{B}{J} \end{bmatrix} \begin{bmatrix} i_a \\ \omega \end{bmatrix} + \begin{bmatrix} \dfrac{1}{L_a} & 0 \\ 0 & -\dfrac{1}{J} \end{bmatrix} \begin{bmatrix} V \\ T_l \end{bmatrix}$$

$$y = \omega = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} i_a \\ \omega \end{bmatrix} + \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} V \\ T_l \end{bmatrix}$$

How to get T.F. From State-space model

$\dot{x} = Ax + Bu \qquad\qquad y = Cx + Du$

Taking Laplace transformation, we get

$$sX(s) = AX(s) + BU(s)$$
$$(sI - A)X(s) = BU(s)$$
$$\text{Or, } X(s) = (sI - A)^{-1}BU(s)$$

$$\begin{aligned} Y(s) &= CX(s) + DU(s) \\ &= [C(sI - A)^{-1}B + D]U(s) \end{aligned}$$

So, the T.F. is $$T.F. = \frac{Y(s)}{U(s)} = C(sI - A)^{-1}B + D$$

Linearity:

For a linear system the ***Principle of Superposition*** is valid

If **input** F_1 produces **response** x_1 and **Input** F_2 produces **response** x_2

Then **input** $(F_1 + F_2)$ must produce **response** $(x_1 + x_2)$.

Law of proportionality

If **input** F_1 produces **response** x_1 then **Input** aF_1 must produce a **response** ax_1

Most of the practical systems are non-linear.