SFWR ENG 3DX4 Summary

Instructor: Dr. Lawford

Course: SFWR ENG 3DX4

*Math objects made using* [*MathType*](http://www.dessci.com/en/products/mathtype/)*; graphs made using* [*Winplot*](http://math.exeter.edu/rparris/winplot.html)*.*

Table of Contents

[Introduction to Systems 1](#_Toc410529597)

[Laplace 2](#_Toc410529598)

[Transfer Functions 3](#_Toc410529599)

[Electrical 3](#_Toc410529600)

[Component stuff 3](#_Toc410529601)

[Mesh Analysis 3](#_Toc410529602)

[Cramer’s Rule 4](#_Toc410529603)

[OP-Amps 4](#_Toc410529604)

[Mechanical 4](#_Toc410529605)

[Translational Systems 4](#_Toc410529606)

[Rotational Systems 5](#_Toc410529607)

[Non-/Linear Systems 5](#_Toc410529608)

[Block Diagrams 6](#_Toc410529609)

[State Space Equations 6](#_Toc410529610)

[Transfer Function -> State Space 7](#_Toc410529611)

Note: the following summaries may be useful:

* [SFWR ENG 2MX3](https://drive.google.com/open?id=0BxW61uJyyN8TTWx5d0gzQW9ZUzQ&authuser=0)
* [ENGINEER 3N03](https://docs.google.com/document/d/117z1qGbrDJJV9bx57CQ4SxEL8Ws8oL27bM7-NgFHNKU/edit)
* [TRON 3TA4](https://drive.google.com/file/d/0BxW61uJyyN8TLTR4UV9fYVdBeEU/view?usp=sharing)

I may review to clarify or correct, but mostly I will omit those things.

# Introduction to Systems

Systems can be represented by **block diagrams** to make it easier to marginalize the different parts of the systems.

# Laplace

Useful for…

Time begins when your signal begins



Initial conditions:

* *c*(0)

**Time domain** (*t*): variables are lower case, e.g. *f* (*t*)

**Frequency domain** (*s*): variables are upper case, e.g. *F* (*s*)

**Transfer function**:

When doing the inverse Laplace, it’s useful to break your fractions up so that you can

**Strictly Stable**: it will eventually get back to the initial position

**Marginally Stable**:

**Unstable**: it will progressively get worse

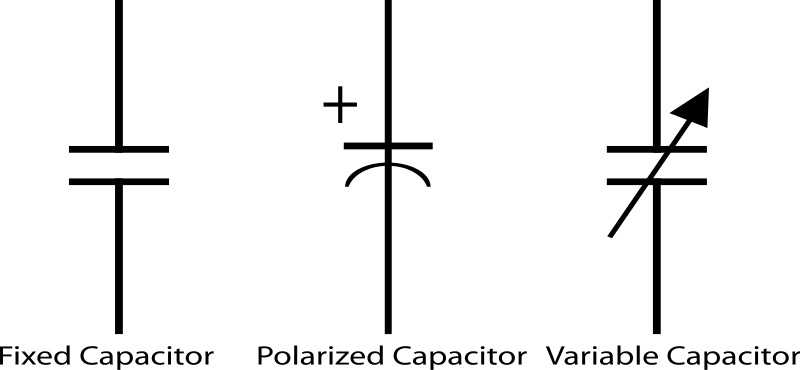


# Transfer Functions

## Electrical

### Component stuff



[](https://upload.wikimedia.org/wikipedia/commons/thumb/1/1c/Types_of_capacitor.svg/800px-Types_of_capacitor.svg.png)

**admittance**:







### Mesh Analysis

You cannot use Ohm’s law to find the current through a voltage source, so represent the current by isomething, like ix.

### Cramer’s Rule





### OP-Amps

## Mechanical

**Translational systems**:

**Rotational Systems**:

**Newton’s Second Law of Motion**: Σ *f = Ma*





### Translational Systems

#### Spring

Spring is like a capacitor

**Force displacement**:

#### Viscous Damper

Using viscous fluid to slow something down

Viscous Damper is like a resistor

**Force displacement**:

#### Mass

Mass is like a inductor

**Force displacement**:

### Rotational Systems

**Transducer**: anything that converts energy to electrical energy

**Transmitter**: long distances

Unstable systems have ∞ steady state error

**Steady-state error** [*e∞*]:



**Final value theorem**: finds steady state error



So and you’re given F(s), so just multiply by s and find the limit.

**Rise time** [*Tr*]: time between 10% and 90% of final value (*c*final)

**Peak time** [*Tp*]: time it takes to get to highest peak (*c*max)

**Settling time** [*Ts*]: how long it takes to get to the steady state within ±2%

**Percent overshoot** [%OS]: how much further is the peak from the final





# Non-/Linear Systems

* Op Amps are linear
* If you don’t have enough voltage, your motor magnets won’t have enough power to switch poles, so they require a minimum voltage

You can’t model non-linear systems, until you linearize it. To do this, we find the slope and approximate the equation of the line, using y=mx+b

**Proportional-Integral-Derivative (PID)**:

If your gears are vibrating, your PID is probably too high

# Block Diagrams

A way of representing a system

**Summing junction**: could be an X or +, but usually an X in this course

**Cascade**: subsystems in series are multiplied

**Parallel**: parallel subsystems have a *summing junction* at the end, so you just add everything together

**Feedback**: positive feedback is bad



Positive:

Negative: 

Simplification:

# State Space Equations

Yeah, you think you know them from 2MX3, but you don’t really know them. Apparently the ABCD variables actually have names.

* **System Matrix [A]**:
* **Input Matrix [B]**:
* **Output Matrix [C]**:
* **Feedforward Matrix [D]**:

# Transfer Function -> State Space

**Phase Variable Approach**:

The *n* state variables will consist of:

* *y*
* the derivatives of *y*