Mechanical/Electrical Systems

A principle attraction of Verilog-AMS, VHDL-AMS is the ability to model mixed nature systems such as mechanical/electrical systems.

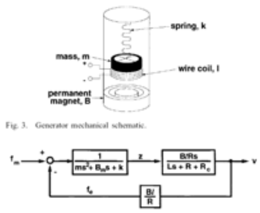
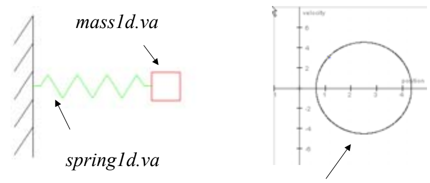


Fig. 3. Generator mechanical schematic.

How can we model this system?

Rajeevan Amirtharaja and Anantha Chandarakasan, "Self-Powered Signal Processing Using Vibration-Based Power Generation", IEEE J. of Solid-State Circuits, Vol. 33, No. 5, May 1998, pp. 687-695.

A Simple Spring System

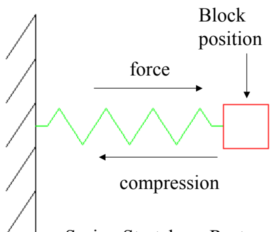


No damping, ideal spring

Plot of velocity vs. position

See http://www.myphysicslab.com

How To Model?



Force of spring moves block, need to determine new position of block (assume no other forces like friction or gravity, block is a point block (no width)

Spring Stretch = Rest_position – block position

F = k (Rest_position – block position)

Spring stiffness

Block Position (x)

F = m * a , m=mass of block, a = acceleration

velocity = dx/dt , where x = block position

acceleration = dv/dt = d^2x/dt, where v = velocity.

F = m * d^2x/dt , solve for position

$$x = \int\int F/m \, dt$$

kinematic, kinematic_v disciplines

```
// Position in meters
nature Position
 units = "m";
 access = Pos;
 ddt_nature = Velocity;
endnature

nature Force
 units = "n";
 access = F;
endnature

nature Velocity
 units = "m/s";
 access = Vel;
 ddt_nature = Acceleration;
 idt_nature = Position;
endnature
```

```
// Conservative disciplines

discipline kinematic
 potential Position;
 flow Force;
enddiscipline

discipline kinematic_v
 potential Velocity;
 flow Force;
enddiscipline
```


Model: spring1d.va

```
`include "constants.h"
`include "discipline.h"

module spring1d(n1,n2);
 inout n1,n2;
 kinematic n1,n2;

 parameter real k = 3 from (0:inf);
// spring constant given in n/m

 parameter real l = 2.5 from (0:inf);
// stretch value of string

 // coordinate system - X = 0, string is unstretched

 analog

   F(n1,n2) <+ k*(Pos(n1,n2)- l);

endmodule
```

Force in Newtons

1 Newton = force to accelerate 1 Kg at 1 m/s

Model: mass1d.va

```
// one dimensional mechanical mass model

module mass1d(n);
 inout n;|
 kinematic n;
 parameter real m = 25 from (0:inf), // mass given in Kg
               init_vel = 0,    // initial speed in m/s
               init_pos = 0;   // initial position in m

 real speed;

 analog
 begin

   speed = idt( F(n)/m ,init_vel);
   Pos(n1,n2) <+ idt(speed,init_pos);

   end

endmodule
```


Model: Velocity Monitor

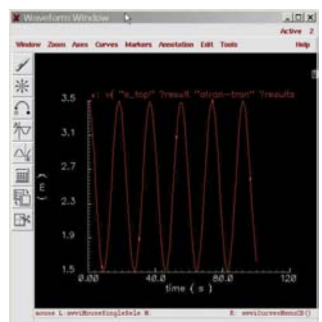
```
`include "constants.h"
`include "discipline.h"

//velocity monitor

module velmon(p,v);
 inout p,v;
 kinematic p;
 kinematic_v v;

 // find velocity

 analog

   Vel(v) <+ ddt(Pos(p));

endmodule
```


Testbench (tb.sp)

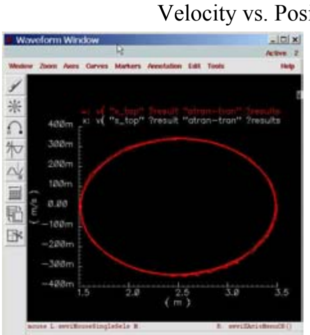
```
//

simulator lang=spectre

global 0 gnd
aliasGnd (gnd 0 ) vsource type=dc dc=0

parameters  simstop=100

ahdl_include "mass1d.va"
ahdl_include "spring1d.va"
ahdl_include "velmon.va"


u1 (s_top  gnd) spring1d k=3 l=2.5;
u2 (s_top ) mass1d init_pos= 3.5;
u3 (s_top v_top) velmon;

atran tran step=0.001 stop=simstop
```

Time in seconds

Position vs. Time



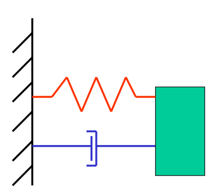
Velocity vs. Position

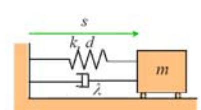


Uses 'Axes' menu to change X/Y axis variables after both curves are plotted

(choose independent variable option after both curves are plotted)

Spring with Dashpot (damper)





F = λ dx/dt

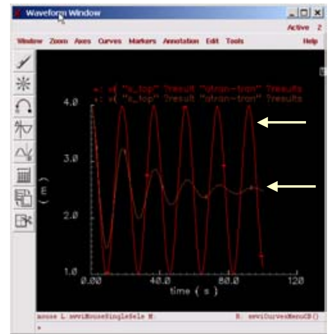
Where λ is damping coefficient

http://www.engin.brown/courses/en4 (dynamics and vibrations)

Model: damper1d.va

```
`include "constants.h"
`include "discipline.h"

module damper1d(n1,n2);
 inout n1,n2;
 kinematic n1,n2;

 parameter real d = 1000 from (0:inf);
 // friction coefficient in n*s/m

 analog

   F(n1,n2) <+ d*ddt(Pos(n1,n2));

endmodule
```


Spring Simulation



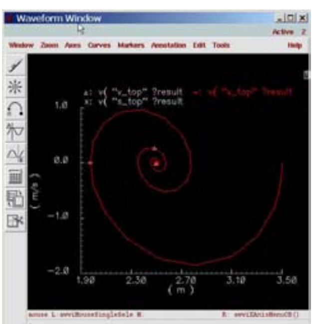
m=25
init_pos =3.5
k = 3
l = 2.5

Without damping

With damping

d=1.0

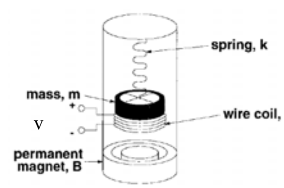
With Damping, Velocity vs. Position



m= 0.5
init_pos =3.5
k = 3
l = 2.5

Damping = 0.5

Conductor Moving in Magnetic Field



B magnetic flux density

l coil length

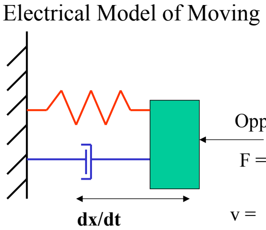
dx/dt coil velocity

i coil current

Faraday's Law of Magnetic induction (induced voltage):
v = B * ***l*** * (dx/dt)

Lorentz force law (electromotive force that opposes spring movement):
F = - B * ***l*** * i

Electrical Model of Moving Coil in Magnetic Field

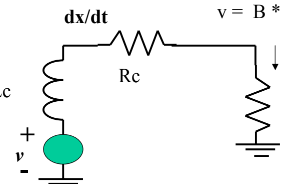


Opposing electromotive force

F = - B * ***l*** * i

dx/dt

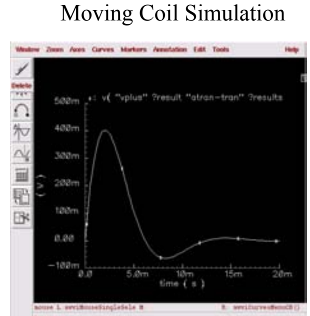
v = B * ***l*** * (dx/dt)



RL (load resistance)

Rc – coil self resistance
Lc – coil self inductance

Moving Coil Simulation



Induced voltage vs. Time

Spring length = 0.1m
Spring mass = 0.5 g
Spring K = 174
Spring length = 0.1m
Initial Deflection = +50%

Coil:

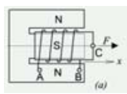
20 Turns, radius 0.025m

Magnet Flux density
.008 Telsa

Load Resistance = 10 ohms, ignore coil self-inductance, self-resistance.
Mechanical damping = 0.3

Voice Coils

- Previous example used external force to move coil through magnetic field
- Can also hold coil stationary in fixed magnetic field, and use alternating current to generate a force
  - This is called a voice coil, used in speakers. The coil can move within the housing to form a mechanical actuator, or the generated force can be use to move an external component.
  - Forms the basis of many magnetic actuators



Current applied through terminals A/B

From http://icosym-nt.cvut.cz/course/node52.html


```
template voice_coil p m pos1 pos2 = b, len, r, l
 electrical p,m # Electrical pins,
 translational_pos pos1, # Position of coil.
 pos2 # Reference position.
 number b = 300m, #Flux density across the air-gap (Teslas)
 len = 15, # Length of the wire in the coil (meters)
 r = 8, # Coil "DC" electrical resistance (Ohms)
 l = 1m # Coil inductance (H)
 {
 var i i # Current in the coil (A)
 var vel_mps vel # Velocity of coil, (meters/sec)
 val pos_m posn # Differential position (coil to ref, meters)
 val frc_N force # Electro-magnetic force, N
 val v v_bemf # Back EMF generated voltage
 values {
 # Define differential position
 posn = pos_m(pos1) - pos_m(pos2)
 # Electro-magnetic force generated by current in coil
 force = b*len*i
 # Calculate back EMF voltage due to coil motion
 v_bemf = b*len*vel
 }
 equations {
  i(p-> m) + = i
 frc_N(pos1-> pos2) += force
 i: v(p)-v(m) = r*i + d_by_dt(l*i) + v_bemf
 vel: vel = d_by_dt(posn)
   }
 }
```

Analogy Saber model of a voice coil model provided for reference

Saber is a proprietary AMS modeling language

Homework

- Duplicate the 2-spring simulation on www.myphysics.com. Turn in a screenshot of a 'awd' plot of block position 1 versus block position 2 for several cycles.
- Duplicate the moving coil simulation using the parameters given.
  - Ignore Rc, Lc of the coil
  - Write an EMF module that takes in spring position, and outputs induced magneto force as a damping force on the motion of the spring (this force is very small compared to the mechanical damping force).
  - If the mechanical damping force is removed, how long does it take the spring to damp to 25% of its maximum value?
  - Capture a screenshot that shows the induced voltage (with mechanical damping) as shown on the previous page. Also capture a screenshot that shows the spring being damped purely by the induced magneto force.

Interesting Links

- http://www.magnetsales.com/Design/Calc_filles/ConversionMag.asp (magnetic units of conversion)
- http://www.magnetsales.com/Design/Tools1.htm (Calculator for magnet parameters based on geometries, material)
- http://www.myphysicslab.com Applets for physics