

oct 03, 14 1:26

transducer.vhd.latin1

Page 1/3

```

--
-- ADVance MS model for electromechanical harvesting_resonator adms_vhdlams_ESYCOM_juine2
007
--
LIBRARY DISCIPLINES;
USE DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
library IEEE;
use IEEE.MATH_REAL.all;

ENTITY electrostatic_transducer IS
    GENERIC(
        -- Parameters for gap closing architecture
        d:real:=30.0e-6;
        S:real:=1.0e-4;
        -- Parameters for area overlap transducer: Ct=C0+alpha*x
        C0:real:=150.0e-12; -- the capacitance at rest (x=0)
        alpha:real:=1.0e-6; -- the coefficient
        -- Parameters of the resonator
        d_stoppers:real:=0.9e-6; -- stopper position
        masse:real:=50.0e-6;
        k:real:=150.0;
        mu:real:=2.16e-3;
        -- initial conditions (not used in this model)
        Q0:real:=0.0; --initial charge
        x0:real:=0.0; --initial position
        v0:real:=0.0 --initial velocity
    );

    -- terminals : C1 et C2 are the capacitor's electrodes
    -- Mp et Mm are the mechanical terminal
    -- the voltage between them represent the external force
    -- Their polarity is important (the direction of the force)
    -- pos: the x-coordinate of the transducer, (output)
    -- capa: the instantaneous transducer capacity
    -- pos and capa are output terminals for measurement purposes
    PORT (
        TERMINAL C1      : Electrical ;
        TERMINAL C2      : Electrical ;
        TERMINAL Mp      : Electrical ;
        TERMINAL Mm      : Electrical;
        TERMINAL pos     : Electrical;
        TERMINAL capa     : Electrical
    );
END;

ARCHITECTURE gap_closing_transducer OF electrostatic_transducer IS

-- The function giving the relation between the transducer capacitance, the displacement x
-- and the parameters of the transducer
function transducer_capacitance_gap_closing(x:real; d:real;S:real) return real is
variable capa:real;
begin
    capa:=8.85e-12*S/(d-x);
return capa;
end;

-- The function giving the spatial gradient of the transducer capacitance (derivative on x)
function grad_transducer_capacitance_gap_closing(x:real; d:real;S:real) return real is
variable capa:real;
begin
    capa:=8.85e-12*S/(d-x)**2.0;
return capa;
end;

quantity charge : real:=0.0;
quantity dCvar_dx : real;

```

oct 03, 14 1:26

transducer.vhd.latin1

Page 2/3

```

quantity uC across iC through C1 to C2; -- the capacitor voltage
quantity ext_force across v through Mp to Mm; -- the external force applied on the mass
quantity x across ix through pos to electrical_ground; -- position of the mobile electrode of transducer
quantity Cvar across ica through capa to electrical_ground; -- the transducer capacitance
quantity F_transd:real; -- transducer's force
quantity F_spring:real; -- spring's force
quantity F_damper:real; -- damper's force
quantity a:real; -- acceleration of the mobile electrode of transducer

-- to add stoppers: uncomment these 3 lines
-- quantity F_stoppers:real; -- position of the mobile electrode of transducer
-- constant mu_stoppers:real:=1.0e3;
-- constant k_stoppers :real:=1.0e6;

BEGIN

-- mechanical side model : a force generator.
-- However, the position must be known for the force calculation
-- We put here the relations between the acceleration, the velocity
-- and the position
v==x'dot;
a==v'dot;
-- then we calculate the capacitance gradient at x
dCvar_dx==grad_transducer_capacitance_gap_closing(x, d, S);

-- We calculate the forces generated by the transducer,
-- the damper and the spring
F_transd==uC*uC*dCvar_dx/2.0;
F_damper== -mu*v;
F_spring== -k*x;

-- We write the 2nd Newtonian equation
-- without stoppers:
ext_force+F_transd+F_spring+F_damper==masse*a;

-- with stoppers
--- ext_force+F_transd+F_spring+F_damper+F_stoppers==masse*a;
--- if (X'above(d-d_stoppers)) use
--- F_stoppers== -mu_stoppers*v - k_stoppers*(x-(d-d_stoppers));
--- else
--- F_stoppers==0.0;
--- end use;

-- Electrical side model : a simple capacitor
-- first we calculate the capacitance value at x
Cvar==transducer_capacitance_gap_closing(x, d, S);
-- then the charge
ic==charge'dot;
-- then the capacitor voltage
charge==Cvar*Uc;

--- to add stoppers : uncomment this line
-- break on X'above(d-d_stoppers);
-- that's all...
END;

-- the principle is the same for the area overlap trasnducer
ARCHITECTURE area_overlap_transducer OF electrostatic_transducer IS

-- The function giving the relation between the transducer capacitance, the displacement
x
-- and the parameters of the transducer
function transducer_area_overlap(x:real; C0:real;alpha:real) return real is

```

oct 03, 14 1:26

transducer.vhd.latin1

Page 3/3

```

variable capa:real;
begin
    capa:=C0+alpha*x;
return capa;
end;

-- The function giving the spatial gradient of the transducer capacitance (derivative on x)
function grad_transducer_area_overlap(x:real; C0:real;alpha:real) return real is
variable capa:real;
begin
    capa:=alpha;
return capa;
end;

quantity charge : real:=0.0;
quantity dCvar_dx : real;
quantity uC across iC through C1 to C2; -- the capacitor voltage
quantity ext_force across v through Mp to Mm; -- the external force applied on the mass
quantity x across ix through pos to electrical_ground; -- position of the mobile electrode of transducer
quantity Cvar across icapa through capa to electrical_ground; -- the transducer capacitance
quantity F_transd:real; -- transducer's force
quantity F_spring:real; -- spring's force
quantity F_damper:real; -- damper's force
quantity a:real; -- acceleration of the mobile electrode of transducer

BEGIN

-- mechanical side model : a force generator.
-- However, the position must be known for the force calculation
-- We put here the relations between the acceleration, the velocity
-- and the position
v==x' dot;
a==v' dot;
-- then we calculate the capacitance gradient at x
dCvar_dx==grad_transducer_area_overlap(x, C0, alpha);

-- We calculate the forces generated by the transducer,
-- the damper and the spring
F_transd==uC*uC*dCvar_dx/2.0;
F_damper== -mu*v;
F_spring== -k*x;

-- We write the 2nd Newtonian equation
ext_force+F_transd+F_spring+F_damper==masse*a;

-- Electrical side model : a simple capacitor
-- first we calculate the capacitance value at x
Cvar==transducer_area_overlap(x, C0, alpha);
-- then the charge
ic==charge' dot;
-- then the capacitor voltage
charge==Cvar*Uc;

END;
```