ANEXO

A continuación se presentan los diferente modelos utilizados para la simulación de la interfaz, así como el programa de cálculo del OTA. En primer lugar se presenta el modelo del acelerómetro torsional, posteriormente el de la capacidad MOS, y por último el programa.

A.1 Modelo de Acelerómetro Torsional

```
------------------------------------------------------------------
--HDL-A model of a pendulum accelerometer                       --
--                                                              --
--      ----x1-- __----- x2 -----                               --
--      ________/ /______________                               --
--     /        x0      /       /                               --
--    /_______  _______/_______/ ancho                          --
--           /_/                                                --
--                                                              --
--      This model was written by: Jose Maria GOMEZ CAMA        --
--      e-mail: chema@europa.fae.ub.es                          --
--                                                              --
------------------------------------------------------------------

ENTITY aceltor IS
        GENERIC (       ro,      -- Density of the material
                        ancho,  -- Is the width of the pendulum
                        alto,    -- Is the height of the pendulum
                        x2,      -- Is the lenght of the bigger arm
                        x1,      -- Is the lenght of the smaller arm
                        x0,      -- Is half the distance between plates
                                 -- that are on the pendulum
                        dist,    -- Is the distance between the capacitor
                                 -- plates without any force aplaid
                        k,       -- Is the torsion constant
                        psi,       -- Dumping factor

                        Voffset -- Is the voltage for the minimum
                                -- capacitance value
```

```
                        : analog);

        COUPLING (      alfa,   -- Is the angular position
                        cl,     -- Large arm capacitance
                        cs,     -- Small arm capacitance
                        mvl,    -- Electrostatic torque in the large arm
                        mvs     -- Electrostatic torque in the small arm
                        : analog);
        PIN (   ebl,    -- First node of the large arm capacitor
                ebs,    -- First node of the small arm capacitor
                ecl,    -- Second node of the large arm capacitor
                ecs,    -- Second node of the small arm capacitor
                a       -- Applaied acceleration, this could be mechanical.
                : electrical);
END ENTITY aceltor;

ARCHITECTURE bhv OF aceltor IS

        STATE ibl, ibs, ico, vbl, vbs, acel: analog;
        CONSTANT ii, b, mg, m0, m1, c0, c1, alfa0, alfa1, e0, c: analog;
        STATE theta, alfat, alfaddt, alfad2dt: analog;
        STATE it, bt: analog;
        STATE thetaddt, thetad2dt: analog;
BEGIN
        RELATION
                PROCEDURAL FOR INIT =>

                        ro      := 2.329e3;     --Kg/m3
                        ancho   := 1.4e-3;      --m
                        alto    := 10.0e-6;     --m
                        x2      := 2.0e-3;      --m
                        x1      := 1.4e-3;      --m
                        x0      := 14.0e-6;     --m
                        dist    := 8.7e-6;      --m
                        k       := 5.0e-6;      --Nm/rad
                        psi     := 102.18;
                        Voffset := -0.5;        --V

                        e0      := 8.85e-12;    --F/m

                        ii      := ro*ancho*alto*(x2**3+x1**3)/3.0;
                        b       := 2.0*psi*sqrt(k*ii);
                        mg      := ro*ancho*alto*(x2**2-x1**2)/2.0;
                        c       := dist/x2;

-- This part linearizes the capacitor and torque equations to avoid
-- a 0/0 singularity.

                        alfa1           := c*1.0e-3;
                        alfa0           := c*-1.0e-3;
                        alfa            := 0.0;
                        theta           := 0.0;
--                        alfaddt        := 0.0;
--                        alfad2dt       := 0.0;

                        c1      := e0*ancho/alfa1*
                                        ln((dist+alfa1*x1)/
                                        (dist+alfa1*x0));
                        m1      := e0*ancho/alfa1**2*
                                        (dist*alfa1*(x0-x1)/
                                        (dist+alfa1*x0)/(dist+alfa1*x1)+
                                        ln((dist+alfa1*x1)/
                                        (dist+alfa1*x0)));
                        c0      := e0*ancho/alfa0*
                                        ln((dist+alfa0*x1)/
                                        (dist+alfa0*x0));
                        m0      := e0*ancho/alfa0**2*
                                        (dist*alfa0*(x0-x1)/
```

```
                                (dist+alfa0*x0)/(dist+alfa0*x1)+
                                ln((dist+alfa0*x1)/
                                (dist+alfa0*x0)));

                PROCEDURAL FOR AC, DC, TRANSIENT =>

                        acel := a.v;

                        vbl := [ebl, ecl].v - Voffset;
                        vbs := [ebs, ecs].v - Voffset;

                        alfa := c*th(theta);
                        alfat := alfa;
                        alfaddt := ddt(alfat);
                        alfad2dt := ddt(alfaddt);
--                        thetaddt := ddt(theta);
--                        thetad2dt := ddt(thetaddt);

-- We see if we are near to the 0 degrees. In this case we take the linear
-- aproximation

                        IF alfa < alfa1 and alfa > alfa0 THEN

                                cl := (c1-c0)/(alfa1-alfa0)*
                                        (alfa-alfa0) + c0;
                                mvl := -(m1-m0)/(alfa1-alfa0)*
                                        (alfa-alfa0) - m0;

                                cs := (c1-c0)/(alfa1-alfa0)*
                                        (-alfa-alfa0) + c0;
                                mvs := (m1-m0)/(alfa1-alfa0)*
                                        (-alfa-alfa0) + m0;

                        ELSE

                                cl := e0*ancho/alfa*
                                        ln((dist+alfa*x1)/
                                        (dist+alfa*x0));
                                mvl := -e0*ancho/alfa**2*
                                        (dist*alfa*(x0-x1)/
                                        (dist+alfa*x0)/(dist+alfa*x1)+
                                        ln((dist+alfa*x1)/
                                        (dist+alfa*x0)));

                                cs := e0*ancho/-alfa*
                                        ln((dist-alfa*x1)/
                                        (dist-alfa*x0));
                                mvs := e0*ancho/alfa**2*
                                        (dist*alfa*(x1-x0)/
                                        (dist-alfa*x0)/(dist-alfa*x1)+
                                        ln((dist-alfa*x1)/
                                        (dist-alfa*x0)));

                        END IF;

--We calculate the current for the capacitors.

                        ibl := cl*ddt(vbl);
                        ibs := cs*ddt(vbs);

                        [ebl, ecl].i %= ibl;
                        [ebs, ecs].i %= ibs;

--                        it := -2.0*c*ii*th(theta)/(ch(theta)**2);
--                        bt := c*b/(ch(theta)**2);

                EQUATION (theta)
                        FOR AC, DC, TRANSIENT =>
```

```
        --Equation for the alfa position.

                        ii*alfad2dt + b*alfaddt + k*alfat ==
                                mvl*vbl**2 + mvs*vbs**2 + mg*acel;

        END RELATION;
END ARCHITECTURE bhv;
```

A.2 Modelo de Capacidad MOS

```
ENTITY capmos IS
        GENERIC(        area,   -- Area en um^2
                        cox,    -- Capacidad del oxido en F
                        n,      -- Concentracion de impurezas cm^-3
                        vfb,    -- Tension de bandas planas
                        mos,    -- Tipo de substrato +1 pmos -1 nmos
                        t       -- Temperatura en K
                        :analog
                );

--        COUPLING(       vs,
--                        chf    -- Capacidad de alta frecuencia
--                        :analog
--                );

        PIN(            gate,   -- Port de puerta
                        bulk    -- Port de bulk
                        :electrical
                );
END ENTITY capmos;

ARCHITECTURE hf OF capmos IS
        constant        kb,     -- Constante de Boltzman
                        q,      -- Carga del electron
                        gap,    -- Gap del silicio
                        es,     -- Constante dielectrica del silicio
                        eox,    -- Constante dielectrica del oxido
                        ni,     -- Portadores intrinsecos
                        beta,   -- Beta
                        ldi,    -- Longitud intrinseca de Debye
                        q0,
                        c0,
                        vf,     -- Potencial de Fermi
                        dox,    -- Anchura del oxido
                        areacm
                        :analog;

        state           fis,    -- Potencial en la superficie
                        vcal,   -- Tension de calculo
                        qsc,    -- Cargas en el semiconductor
                        v       -- Tension gate-bulk
                        :analog;

        variable        clf,    -- Capacidad de baja frequencia
                        vs,     -- Tension en la superficie
                        chf,    -- Capacidad de alta frecuencia
                        ccal,   -- Capacidad para calculo
                        fir,    -- Potencial se superficie real
                        fq,
                        fc,
                        signvs  -- Signo de vs
                        :analog;

BEGIN
        RELATION
                PROCEDURAL FOR INIT =>

                        area    := 625.0;       -- um^2
                        cox     := 228.0e-9;    -- F/cm^2
                        n       := 2.94e16;     -- cm^-3
                        vfb     := 0.0;         -- V
                        mos     := -1.0;        -- Substrato nmos
                        t       := 300.0;       -- K
```

```
                    kb      := 1.38e-23;
                    q       := 1.602e-19;
                    gap     := 1.12;        -- eV
                    es      := 1.04e-12;
                    eox     := 3.4e-13;
                    ni      := 1.0e10;

                    areacm  := area*1.0e-8;

                    beta    := q/(kb*t);
                    ldi     := sqrt(es*kb*t/(2.0*q**2*ni));
                    q0      := 2.0*sqrt(2.0)*q*ni*ldi;
                    c0      := q0*beta/2.0;
                    vf      := ln(ni/n);

--            PROCEDURAL FOR DC =>

                    -- Calculo de la capacidad para el caso de DC

--                    v := [gate, bulk].v;

--                    vs := fis*beta;

--                    if vs > 0.0 then signvs := 1.0;
--                    elsif vs < 0.0 then signvs := -1.0;
--                    else signvs := 0.0;
--                    end if;

--                    f := sqrt(-vs*sh(vf)-(ch(vf)-ch(vs+vf))) +
--                            1.0e-20;

--                    qsc := -q0*signvs*f;

--                    vcal := fis - qsc/cox;

--                    ccal := signvs*es/ldi*(sh(vs+vf)-sh(vf))/
--                            (sqrt(2.0)*f);

--                    c := cox*ccal/(cox+ccal)*areacm;

--                    [gate, bulk].i %= c*ddt(v);

                PROCEDURAL FOR DC, AC, TRANSIENT =>

                    -- Calculo de la capacidad para el caso de AC

                    v := [gate, bulk].v;

                    fir := fis - vfb;

                    vs := fis*beta;

                    if vs > 0.0 then signvs := 1.0;
                    elsif vs < 0.0 then signvs := -1.0;
                    else signvs := 0.0;
                    end if;

                    fq := sqrt(-vs*sh(vf)-(ch(vf)-ch(vs+vf))) +
                            1.0e-20;

                    qsc := -q0*signvs*fq;

                    vcal := fir - qsc/cox;

                    fc := sqrt(((vs-1.0)*exp(-vf)+
                            exp(-(vs+vf)))/2.0) +
                            1.0e-20;
```

```
--                          ccal := signvs*c0*(sh(vs+vf)-sh(vf))/
--                                  fq;

--                          clf := ccal/(ccal+cox)*(cox*areacm);

                          ccal := signvs*c0*exp(-vf)*(1.0-exp(-vs))/
                                  (2.0*fc);

                          chf := ccal/(ccal+cox)*(cox*areacm);

                          [gate, bulk].i %= chf*ddt(v);

                  EQUATION (fis) FOR AC, DC, TRANSIENT =>

                          -- Nos resuelve la ecuacion de la carga
                          -- en el semiconductor

                          v == mos*vcal;

        END RELATION;
END ARCHITECTURE hf;
```

A.3 Programa de Cálculo del OTA

```
#include <iostream.h>
#include <math.h>

int main()
{
	double sr, gbw, cl;
	double k1, k3, k5, k7, k9, k11, kib;
	double gm1, ib;
	double vtp = -1, vtn= 0.815;
	double bp = 38e-6, bn = 127.6e-6;
	double v5, v7, vib, vdd=5., vss=0.;

	cout << "Gain Band With: ";
	cin >> gbw;
	cout << "Slew Rate: ";
	cin >> sr;
	cout << "C load: ";
	cin >> cl;

	gm1 = gbw*2*M_PI*cl;
	ib = sr*cl;

	k1 = gm1/2./bp/(0.2);
	k3 = 2.*ib/bn/(0.5*0.5);
	v5 = 2.4-(0.5);
//	k5 = gm1/2./bn/(v5-vtn);
	k5 = 2.*ib/bn/((0.3)*(0.3));
	v7 = -2.4-(-0.5);
//	k7 = gm1/2./bp/(vtp-v7);
	k7 = 2.*ib/bp/((0.3)*(0.3));
	k9 = 2.*ib/bp/(0.5*0.5);
	k11 = 2.*ib/bp/(0.3*0.3);
	vib = (vdd-vss)-(-(-0.3+vtp)+0.5+vtn);
	kib = 2.*ib/bp/(vib*vib);

	cout << "k1 = " << k1 << '\n';
	cout << "k3 = " << k3 << '\n';
	cout << "k5 = " << k5 << '\n';
	cout << "k7 = " << k7 << '\n';
	cout << "k9 = " << k9 << '\n';
	cout << "k11 = " << k11 << '\n';
	cout << "kib = " << kib << '\n';
	cout << "Ib = " << ib << '\n';
}
```