CAPÍTULO 5:

MODELO DE RESISTENCIAS VARIABLES

5.1. Introducción

El modelo más preciso que es posible diseñar consiste en aquel capaz de reproducir fielmente las corrientes de pullup, pulldown, power_clamping y ground_clamping en función de la tensión de salida, la tensión de alimentación y el instante de tiempo en que nos encontremos, tal y como indica la información contenida en el fichero IBIS correspondiente.

Este diseño equivale por tanto a un modelo de resistencias variables, adecuado para simulaciones en que se tengan en cuenta reflexiones en la linea de transmisión a la que vaya conectada el driver.

5.2. Modelo utilizado

El circuito utilizado para este modelo se muestra en la siguiente figura:

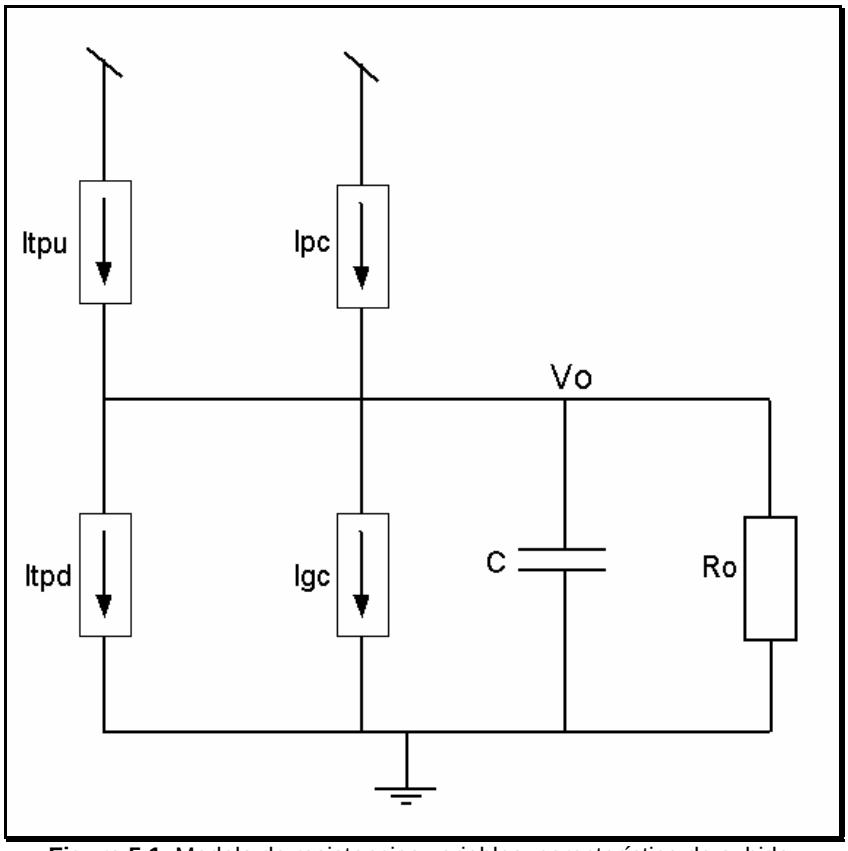

Figura 5.1: Modelo de resistencias variables, característica de subida.

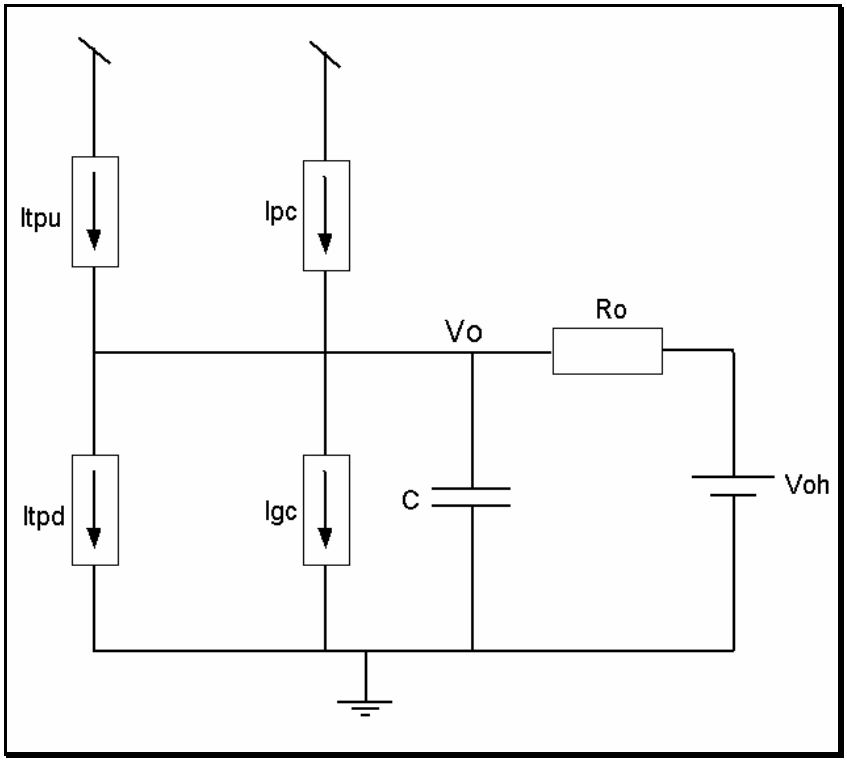

Figura 5.2: Modelo de resistencias variables, característica de bajada.

Se observan cuatro fuentes de corriente, que tienen como objeto adaptar la información disponible en un fichero IBIS:

- **Ipc**: Control de la corriente de *power_clamping*. Modela la corriente que absorve el circuito mediante sus estructuras limitadoras, cuando a la salida el voltaje supera su rango de funcionamiento normal.

  Constituye una fuente de corriente dependiente de tensión (VCCS), donde la tensión de control es la diferencia entre la alimentación del circuito y el voltaje de salida. La corriente asociada a cada voltaje se obtiene del fichero IBIS, a partir de la palabra reservada [POWER_clamp].

- **Igc**: Control de la corriente de *ground_clamping*. Modela la corriente que absorve el circuito mediante sus estructuras limitadoras, cuando a la salida el voltaje es inferior a su rango de funcionamiento normal.

Constituye una fuente de corriente dependiente de tensión (VCCS), donde la tensión de control es la diferencia entre el voltaje de salida y la referencia de tierra del circuito. La corriente asociada a cada voltaje se obtiene del fichero IBIS, a partir de la palabra reservada [GND_clamp].

- **Itpu**: Corriente total de pullup del driver, se calcula como el producto de dos corrientes, Ipu e Ikpu.

  - **Ipu** es una fuente de corriente dependiente de la diferencia de tensión Vcc-Vout, (VCCS). La corriente asociada a cada voltaje se obtiene a partir de la palabra reservada [Pullup].

  - **Ikpu** es una fuente de corriente que varia básicamente entre los valores cero y uno, y que actúa como multiplicador de Ipu. Su objetivo es modelar el comportamiento dinámico del driver, esto es, conseguir que las curvas de subida y bajada del circuito (dependientes del tiempo) correspondan con la información descrita en las palabras reservadas [Rising Waveform] y [Falling Waveform]. El cálculo de esta corriente será descrito en el siguiente apartado.

- **Itpd**: Corriente total de pulldown del circuito. Se calcula como el producto de dos corrientes, Ipd e Ikpd.

  - **Ipd** es una fuente de corriente dependiente de la tensión Vout, (VCCS). La corriente asociada a cada voltaje se obtiene a partir de la palabra reservada [Pulldown] del fichero IBIS correspondiente.

  - **Ikpd** es una fuente de corriente que varia básicamente entre los valores cero y uno, y que actúa como multiplicador de Ipd. Su objetivo, al igual que Ikpu, es modelar el comportamiento dinámico del driver.

Es importante destacar y tener en cuenta que el sentido de las corrientes Ipu e Ipc definidas en los modelos es contrario al de la especificación IBIS.

5.3. Cálculo de Ikpu e Ikpd.

La función de estas corrientes es modelar la evolución de la salida del driver con respecto al tiempo. Una vez calculadas estas corrientes se sustituyen en el circuito completo, de forma que un análisis transitorio del mismo ha de generar a la salida el comportamiento esperado.

Para dar valores a estas corrientes es necesario tener una referencia de comportamiento deseado; la referencia utilizada será la relación tiempo-tensión de salida, descrita en las palabras reservadas:

a) [Rising Waveform], para la curva de subida.

b) [Falling Waveform], para la curva de bajada del driver.

Se busca la evolución de Ikpu e Ikpd con respecto al tiempo. Si se observa la estructura de los circuitos sobre los que se trabaja se puede comprobar que para cada tensión de salida Vout es posible conocer la intensidad asociada en las fuentes de corriente Ipc, Igc, Ipu e Ipd, así como la corriente por el condensador (Ic) y por la carga (Ix). De esta forma, aplicando la ley de Kirchhoff de las intensidades, quedarían como incógnitas por resolver los valores de Ikpu e Ikpd.

Si se asume una relación entre ambas corrientes se estará en condiciones de asociar a cada tensión de salida un valor para las citadas intensidades. Como además cada salida se asocia con un determinado instante de tiempo, quedan determinadas las relaciones Ikpu-t e Ikpd-t. Este será el método empleado.

Queda por especificar la relación entre Ikpu e Ikpd. Ya que Ikpu se asocia con la estructura de pullup del driver (activa cuando el circuito fuerza un nivel alto a la salida), e Ikpd se asocia con la estructura de pulldown (activa a nivel bajo), es razonable considerar una relación complementaria entre ambas corrientes, de forma que la suma de ambas sume uno. De esta forma se considerará la relación:

$$Ikpd = 1 - Ikpu$$

Por lo tanto, se verifica que:

$$Ipu \cdot Ikpu + Ipc = Ipd \cdot (1 - Ikpu) + Igc + Ic + Ix$$

Así pues:

$$Ikpu = \frac{Ipd + Igc + Ic + Ix - Ipc}{Ipu + Ipd}$$

5.4. Funciones en MATLAB para el cálculo de Ikpu e Ikp.

A partir de la información de un fichero IBIS, tal y como ha sido descrito en apartados anteriores, es posible conseguir todos los valores necesarios para proceder al cálculo de las corrientes Ikpu e Ikpd.

Dos funciones diferentes han sido creadas en MATLAB con este objetivo, según se trabaje con la característica de subida o con la de bajada del driver. La diferencia fundamental recae en el cálculo de la corriente a través de la carga de salida del circuito, que para la curva de subida es la resistencia Ro conectada a tierra, mientras que para la curva de bajada esta resistencia esta conectada a una fuente de alimentación de valor Voh. Por lo demás, y salvo el trabajar con el par de vectores [Tdesr, Vdesr] al pasar a nivel alto o el par [Tdesf, Vdesf] al pasar a nivel bajo, las funciones son idénticas.

A continuación son mostradas ambas funciones:

```
function [Ikpur,Ikpdr ] = Ikp_r(Voh,Ipu,Ipc,Ipd,Igc,Vreference,Vclamping,Vdesr,Tdesr,Ro,C)
    % Function used to calculate automatically the currents Ikpur and Ikpdr, necessary for the
    % nonlinear model. Developed by Mario Palma.
    % Voh: Power supply.
    % Ipu: Pullup current, taken from the IBIS file.
    % Ipc: Power clamping current, taken from the IBIS file.
    % Ipd: Pulldown current, taken from the IBIS file.
    % Igc: Ground clamping current, taken from the IBIS file.
    % Vreference: Voltage values associated with Ipd and Ipu, taken from the IBIS file.
    % Vclamping: Voltage values associated with Ipc and Igc, taken from the IBIS file.
    % Vdesr: Output voltage that it is required, taken from [Rising Waveform].
    % Tdesr: Time associated with Vdesr, taken from [Rising Waveform].
    % Ro: Output load, taken from the IBIS file.
    % C: Ccomp, taken from the IBIS file.

    % If the output voltage over a load is known, it is possible to calculate
    % the current flowing through it:
    Ix=Vdesr/Ro;

    % To calculate the capacitor current is necessary to solve the capacitor
    % equation:
    ic=C*(diff(Vdesr)./diff(Tdesr));% Contains one less element than the original arrays.
    ic=[ic; 0];                     % At the last instant the current through
                                    % the capacitor is considered zero.

    % It necessary to calculate the currents associated to the output voltage
    % for each current source:
    Ipui=(interp1(Vreference,Ipu,(Voh-Vdesr)));
    Ipdi=(interp1(Vreference,Ipd,Vdesr));

    Ipci=(interp1(Vclamping,Ipc,(Voh-Vdesr)));
    Igci=(interp1(Vclamping,Igc,Vdesr));

    % Now the k-vectors are created:
    Ikpur=(Ipdi+Igci+ic+Ix-Ipci)./(Ipui+Ipdi);
    Ikpdr=1-Ikpur;
    plot(Tdesr,Ikpur,Tdesr,Ikpdr,':'); grid; axis; zoom;
    xlabel('time');ylabel('Ikp_r currents');
    end
```

```
function [Ikpuf,IkpdfIkp_f(Voh,Ipu,Ipc,Ipd,Igc,Vreference,Vclamping,Vdesf,Tdesf,Ro,C)
    % Function used to calculate automatically the currents Ikpuf and Ikpdf, necessary
    % for the nonlinear model
    % Voh: Power supply.
    % Ipu: Pullup current, taken from the IBIS file.
    % Ipc: Power clamping current, taken from the IBIS file.
    % Ipd: Pulldown current, taken from the IBIS file.
    % Igc: Ground clamping current, taken from the IBIS file.
    % Vreference: Voltage values associated with Ipd and Ipu, taken from the IBIS file.
    % Vclamping: Voltage values associated with Ipc and Igc, taken from the IBIS file.
    % Vdesf: Output voltage that it is required, taken from [Rising Waveform].
    % Tdesf: Time associated with Vdesf, taken from [Falling Waveform].
    % Ro: Output load, taken from the IBIS file.
    % C: Ccomp, taken from the IBIS file.

    % If the output voltage over a load is known, it is possible to calculate
    % the current flowing through it:
    Ix=(Vdesf-Voh)/Ro;   % This line changes if it is compared with the function Kp_r.m

    % To calculate the capacitor current is necessary to solve the capacitor
    % equation:
    ic=C*(diff(Vdesf)./diff(Tdesf));% Contains one less element than the original arrays.
    ic=[ic; 0];                     % At the last instant the current through
                                    % the capacitor is considered zero.

    % It necessary to calculate the currents associated to the output voltage
    % for each current source:
    Ipui=interp1(Vreference,Ipu,(Voh-Vdesf));  % The arrays are trasposed to fix
    Ipdi=interp1(Vreference,Ipd,Vdesf);        % the dimension of Vreference.

    Ipci=interp1(Vclamping,Ipc,(Voh-Vdesf));
    Igci=interp1(Vclamping,Igc,Vdesf);

    % Now the k-vectors are created:
    Ikpuf=(Ipdi+Igci+ic+Ix-Ipci)./(Ipui+Ipdi);
    Ikpdf=1-Ikpuf;
    plot(Tdesf,Ikpuf,Tdesf,Ikpdf,':'); grid; axis; zoom;
    xlabel('time');ylabel('k-coefficients');
    end
```

5.5. Ejemplo. Resultados con el modelo DR-2.

La precisión en los resultados de estas funciones va a depender del número de puntos utilizados, principalmente en los vectores Tdes_ y Vdes_.

Como ejemplo, a continuación se muestran los resultados proporcionados por MATLAB al trabajar con el modelo DR-2, facilitado por Siemens:

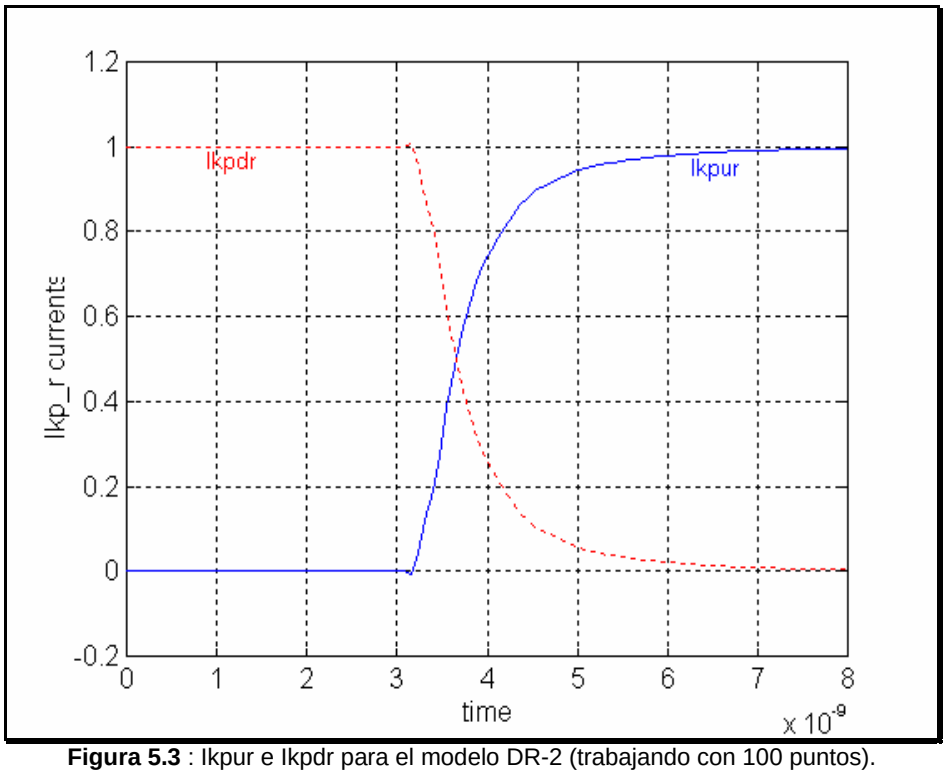

Figura 5.3 : Ikpur e Ikpdr para el modelo DR-2 (trabajando con 100 puntos).

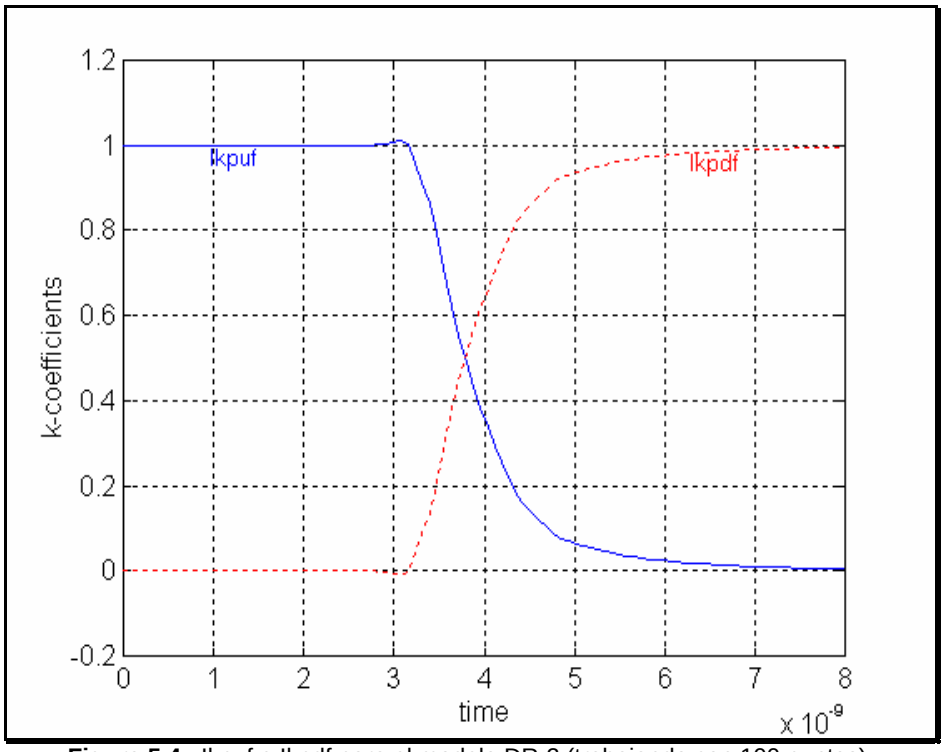

Figura 5.4 : lkpuf e lkpdf para el modelo DR-2 (trabajando con 100 puntos).

5.6. Implementación del modelo en HSPICE.

Para comprobar la operatividad del modelo, así como para poder compararlo posteriormente con el circuito real (último capítulo), el modelo va a ser implementado con HSPICE.

Es interesante destacar la forma en que han sido modeladas las distintas fuentes de corriente:

- **Ipu e Ipd**: Fuentes de corriente controladas por tensión (VCCS), definidas en nodos auxiliares. Se realiza un interpolado lineal entre los puntos definiendo la relación entre ellos como piecewise linear (pwl). En el modelo quedan de la siguiente forma:

```
Gpd    0    auxpd    VCCS    pwl(1)    out  0    [pares V,I]

Gpu    0    auxpu    VCCS    pwl(1)    pw   out  [pares V,I]
```

- **Ikpu_** : Fuente de corriente dependiente del tiempo, definida en un nodo auxiliar. De nuevo el interpolado entre puntos consecutivos es lineal. La información de la relación tiempo-intensidad es la salida generada por MATLAB. En el fichero de estímulo queda así:

```
Ikpu_    0    auxk  pwl    [pares t,Ikpu_]
```

- **Ikpd_**: Fuente de corriente dependiente del tiempo definida en un nodo auxiliar. Se calcula como 1-Ikpu_ de la siguiente forma:

```
Fkpdr    0    auxd  CCCS    poly(1) Vkpu_r    1    -1
```

- **Itpu**: Fuente de corriente resultado de multiplicar Ipu e Ikpu_. El producto[1] se realiza utilizando la opción poly(2) de HSPICE:

```
Ftpu  pw    out    CCCS    poly(2)    Vkpu_    Vpu   0 0 0 0 1
```

[1] Esta operación puede ser muy complicada si no se aprovechan correctamente las facilidades de HSPICE.

- **Itpd**: Fuente de corriente resultado de multiplicar Ipd e Ikpd_. El producto se realiza utilizando la opción poly(2) de HSPICE:

```
Ftpu    out  0    CCCS    poly(2)    Vkpd_    Vpd  0 0 0 0 1
```

Los ficheros completos HSPICE con los que se ha trabajado se encuentran listados en el **apéndice 3**.

Para facilitar en gran medida la generación de estos ficheros ha sido creado, con la ayuda de Antonio Alcaide Alias[2], un script en lenguaje PERL que construye a partir de un fichero de entrada IBIS la mayoria del fichero de estímulo para HSPICE. Este programa se encuentra listado en el **apéndice 4**.

[2] Antonio Alcaide Alias es Ingeniero Superior de Telecomunicación por la Universidad de Sevilla.