ANEXO 4

Código Matlab interfaz library

Veamos las funcione que han sido necesarias para generar la interfaz library y dotarla de funcionalidad. Las funciones son:

- **Library.m**

```
% Script that has got the code that controls the behaviour of the GUI

function [varargout] = library(varargin)
% LIBRARY M-file for library.fig
%      LIBRARY, by itself, creates a new LIBRARY or raises the existing
%      singleton*.
%
%      H = LIBRARY returns the handle to a new LIBRARY or the handle to
%      the existing singleton*.
%
%      LIBRARY('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in LIBRARY.M with the given input arguments.
%
%      LIBRARY('Property','Value',...) creates a new LIBRARY or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before library_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to library_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help library

% Last Modified by GUIDE v2.5 20-Oct-2005 12:01:10

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @library_OpeningFcn, ...
                   'gui_OutputFcn',  @library_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before library is made visible.
function library_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to library (see VARARGIN)

global m1; % Value menu Converters
global m2; % Value menu Blocks
global m3; % Value menu Components
global m4; % Value menu Types
global menu_comp; % Dinamic menu Components
global menu_type; % Dinamic manu Types
global find; % Activate the description
global open; % Open the selected archive

% Iniatilize some values of global variables
[m1,m2,m3,m4,find,open] = var_global(handles);

handles.m1=m1;
handles.m2=m2;
handles.m3=m3;
handles.m4=m4;
handles.find=find;
handles.open=open;


% Choose default command line output for library
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes library wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = library_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on selection change in menu_conv.
function menu_conv_Callback(hObject, eventdata, handles)
% hObject    handle to menu_conv (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns menu_conv contents as cell array
%        contents{get(hObject,'Value')} returns selected item from menu_conv

global valconv;
```

```
global list_comp;
global list_type;

valconv = get(hObject,'Value');
list_comp = set(handles.menu_comp,'String');
list_type = set(handles.menu_type,'String');
m2 = 1;
m3 = 1;
m4 = 1;
set(handles.menu_bloc,'Value',1);
set(handles.menu_type,'Value',1);
set(handles.menu_comp,'Value',1);

switch valconv
case 1
   m1 = 1; % TD Sigma-Delta Converters
   list_comp = 'Library Components|Integrators|Comparators|Quantizers|All models|Others...';
   list_type = ' -';
   set(handles.menu_comp,'String',list_comp);
   set(handles.menu_type,'String',list_type);
case 2
   m1 = 2; % TC Sigma-Delta Converters
   list_comp = ' -';
   list_type = ' -';
   set(handles.menu_comp,'String',list_comp);
   set(handles.menu_type,'String',list_type);
case 3
   m1 = 3; % Pipeline Converters
   list_comp = 'Library Components|ADC_Stage|DAC_Stage|SHA_Stage|Stage complete|Stage
last|Digital correction|All models|Others...';
   list_type = ' -';
   set(handles.menu_comp,'String',list_comp);
   set(handles.menu_type,'String',list_type);
case 4
   m1 = 4; % Flash Converters
   list_comp = 'Library Components|Reference Generator|Comparators|Encoder|All
models|Others...';
   list_type = ' -';
   set(handles.menu_comp,'String',list_comp);
   set(handles.menu_type,'String',list_type);
end

handles.list_comp = list_comp;
handles.list_type = list_type;
handles.m1 = m1;
handles.m2 = m2;
handles.m3 = m3;
handles.m4 = m4;
guidata(hObject, handles);



% --- Executes during object creation, after setting all properties.
function menu_conv_CreateFcn(hObject, eventdata, handles)
% hObject    handle to menu_conv (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
```

```
%      See ISPC and COMPUTER.
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% --- Executes on selection change in menu_bloc.
function menu_bloc_Callback(hObject, eventdata, handles)
% hObject    handle to menu_bloc (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns menu_bloc contents as cell array
%       contents{get(hObject,'Value')} returns selected item from menu_bloc

global valbloq;
global list_comp;
global list_type;
global m1;

valbloq = get(hObject,'Value');
m1 = handles.m1;
m3 = 1;
m4 = 1;
set(handles.menu_type,'Value',1);
set(handles.menu_comp,'Value',1);

switch valbloq
case 1
   m2 = 1;
   % Case 1-1: Ideal Blocks
   if (m1 == 1)
      list_comp        =        'Library        Components|Integrators|Comparators|Quantizers|All
models|Others...';
      list_type = ' -';
      set(handles.menu_comp,'String',list_comp);
      set(handles.menu_type,'String',list_type);
   end
   % Case 2-1: Ideal Blocks

   % Case 3-1: Ideal Blocks
   if (m1 == 3)
      list_comp        =        'Library        Components|ADC_Stage|DAC_Stage|SHA_Stage|Stage
complete|Stage last|Digital correction|All models|Others...';
      list_type = ' -';
      set(handles.menu_comp,'String',list_comp);
      set(handles.menu_type,'String',list_type);
   end
   % Case 4-1: Ideal Blocks
   if (m1 == 4)
      list_comp    =    'Library    Components|Reference    Generator|Comparators|Encoder|All
models|Others...';
      list_type = ' -';
      set(handles.menu_comp,'String',list_comp);
      set(handles.menu_type,'String',list_type);
   end
case 2
```

```
  m2 = 2;
  % Case 1-2: No-ideal Blocks
  if (m1 == 1)
    list_comp = 'Library Components|Integrator|Comparator|Jitter Noise|All
models|Others...';
    list_type = ' -';
    set(handles.menu_type,'String',list_type);
    set(handles.menu_comp,'String',list_comp);
  end
  % Case 2-2: No-ideal Blocks

  % Case 3-2: No-ideal Blocks
  if (m1 == 3)
    list_comp = 'Library Components|DAC_Stage|Stage complete|Jitter Noise|All
models|Others...';
    list_type = ' -';
    set(handles.menu_type,'String',list_type);
    set(handles.menu_comp,'String',list_comp);
  end
  % Case 4-2: No-ideal Blocks
  if (m1 == 4)
    list_comp = 'Library Components|Reference Generator|Comparators|All models|Others...';
    list_type = ' -';
    set(handles.menu_type,'String',list_type);
    set(handles.menu_comp,'String',list_comp);
  end
case 3
  m2 = 3;
  % Case 1-3: Examples
  if (m1 == 1)
    list_comp = ' -|Own examples';
    list_type = 'Library Examples|TD Sigma Delta_LP_2order_ideal_simple|TD Sigma
Delta_LP_2order_real_simple|TD Sigma Delta_LP_2order_ideal_cascade|TD Sigma
Delta_LP_2order_real_cascade|All examples...';
    set(handles.menu_type,'String',list_type);
    set(handles.menu_comp,'String',list_comp);
  end
  % Case 2-3: Examples

  % Case 3-3: Examples
  if (m1 == 3)
    list_comp = ' -|Own examples';
    list_type='Library
Examples|Pipeline_ideal_10b|Pipeline_gain_10b|Pipeline_mismatch_10b|Pipeline_SR_BW_
10b|Pipeline_voffset_10b|Pipeline_jitter_10b|All examples...';
    set(handles.menu_type,'String',list_type);
    set(handles.menu_comp,'String',list_comp);
  end
  % Case 4-3: Examples
  if (m1 == 4)
    list_comp = ' -|Own examples';
    list_type = 'Library Examples|Flash_offset_relay|All examples...';
    set(handles.menu_type,'String',list_type);
    set(handles.menu_comp,'String',list_comp);
  end
end

handles.list_comp = list_comp;
handles.list_type = list_type;
```

```
handles.m2 = m2;
handles.m3 = m3;
handles.m4 = m4;
guidata(hObject, handles);



% --- Executes during object creation, after setting all properties.
function menu_bloc_CreateFcn(hObject, eventdata, handles)
% hObject    handle to menu_bloc (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end



% --- Executes on selection change in menu_comp.
function menu_comp_Callback(hObject, eventdata, handles)
% hObject    handle to menu_comp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns menu_comp contents as cell array
%       contents{get(hObject,'Value')} returns selected item from menu_comp

global valcomp;
global m2;
global list_type;

valcomp = get(hObject,'Value');
m1= handles.m1;
m2= handles.m2;
m3 = 1;
m4 = 1;
set(handles.menu_type,'Value',1);

if m2 == 3  % Examples
   switch valcomp
      case 1
         m3 = 1;
         switch m1
            case 1
               % Examples TD Sigma Delta
               list_type = 'Library Examples|TD Sigma Delta_LP_2order_ideal_simple|TD Sigma
Delta_LP_2order_real_simple|TD     Sigma     Delta_LP_2order_ideal_cascade|TD     Sigma
Delta_LP_2order_real_cascade|All examples...';
               set(handles.menu_type,'String',list_type);
            case 2
               % Examples TC Sigma Delta
            case 3
               % Examples Pipeline
               list_type                                    =                                    'Library
Examples|Pipeline_ideal_10b|Pipeline_gain_10b|Pipeline_mismatch_10b|Pipeline_SR_10b|P
```

```
ipeline_voffset_10b|Pipeline_jitter_10b|All examples...';
            set(handles.menu_type,'String',list_type);
          case 4
            % Examples Flash
            list_type = 'Library Examples|Flash_offset_relay|All examples...';
            set(handles.menu_type,'String',list_type);
        end
      case 2
        m3 = 2;
        list_type = ' -';
        set(handles.menu_type,'String',list_type);
  end
else
   switch valcomp
      case 1
        m3 = 1;
        % Case 1-2-1: Library Components
        if (m1 == 1|m1 == 2|m1 == 3|m1 == 4) & (m2 == 1|m2 == 2)
          list_type = ' -';
          set(handles.menu_type,'String',list_type);
        end
      case 2
        m3 = 2;
        % Case 1-2-2: TD Sigma Delta & No-ideal Blocks & Integrator
        if (m1 == 1) & (m2 == 2)
          list_type = 'Switch thermal noise|Operational amplifier noise|Operational amplifier
finite gain|Operational amplifier SR+GBW|Operational amplifier saturation voltages|All no-
idealities together';
          set(handles.menu_type,'String',list_type);
        end
        % Case 3-1-2: Pipeline & Ideal & ADC
        if (m1 == 3) & (m2 == 1)
          list_type = ' -';
          set(handles.menu_type,'String',list_type);
        end
        % Case 3-2-2: Pipeline & No-Ideal & DAC
        if (m1 == 3) & (m2 == 2)
          list_type = 'Offset reference voltage';
          set(handles.menu_type,'String',list_type);
        end
        % Case 4-2-2: Flash & No-ideal & Reference Generator
        if (m1 == 4) & (m2 == 2)
          list_type = 'Offset reference generator';
          set(handles.menu_type,'String',list_type);
        end
      case 3
        m3 = 3;
        % Case 1-2-3: TD Sigma Delta & No-ideal Blocks & Comparator
        if (m1 == 1) & (m2 == 2)
          list_type = 'Hysteresis|Offset';
          set(handles.menu_type,'String',list_type);
        end
        % Case 3-1-3: Pipeline & Ideal & DAC
        if (m1 == 3) & (m2 == 1)
          list_type = ' -';
          set(handles.menu_type,'String',list_type);
        end
        % Case 3-2-3: Pipeline & No-Ideal & Stage complete
        if (m1 == 3) & (m2 == 2)
```

```
        list_type = 'Stage RSD Finite gain|Stage RSD Mismatch|Stage RSD SR and BW|Stage
RSD Offset|Stage RSD Jitter noise';
        set(handles.menu_type,'String',list_type);
      end
      % Case 4-1-3: Flash & Ideal & Comparators
      if (m1 == 4) & (m2 == 1)
        list_type = 'Comparator without Ts|Comparator with Ts';
        set(handles.menu_type,'String',list_type);
      end
      % Case 4-2-3: Flash & No-Ideal & Comparators
      if (m1 == 4) & (m2 == 2)
        list_type = 'Offset|Offset Relay|Offset Relay Pole';
        set(handles.menu_type,'String',list_type);
      end
    case 4
      m3 = 4;
      % Case 1-1-4: TD Sigma Delta & No-ideal Blocks & Noise Jitter
      if (m1 == 1) & (m2 == 2)
        list_type = 'Clock Jitter';
        set(handles.menu_type,'String',list_type);
      end
      % Case 3-1-4: Pipeline & Ideal & SHA
      if (m1 == 3) & (m2 == 1)
        list_type = ' -';
        set(handles.menu_type,'String',list_type);
      end
      % Case 3-2-4: Pipeline & No-Ideal & Jitter
      if (m1 == 3) & (m2 == 2)
        list_type = ' -';
        set(handles.menu_type,'String',list_type);
      end
      % Case 4-1-4: Flash & Ideal & Encoder
      if (m1 == 4) & (m2 == 1)
        list_type = ' -';
        set(handles.menu_type,'String',list_type);
      end
      % Case 4-2-4: Flash & No-ideal & All
      if (m1 == 4) & (m2 == 2)
        list_type = ' -';
        set(handles.menu_type,'String',list_type);
      end
    case 5
      m3 = 5;
      % Case 1-2-5: TD Digma Delta & Ideal and No-ideal Blocks & All models
      if (m1 == 1) & (m2 == 1|m2 == 2)
        list_type = ' -';
        set(handles.menu_type,'String',list_type);
      end
      % Case 3-1-5: Pipeline & Ideal & Stage complete
      if (m1 == 3) & (m2 == 1)
        list_type = ' -';
        set(handles.menu_type,'String',list_type);
      end
      % Case 3-2-5: Pipeline & No-Ideal & All
      if (m1 == 3) & (m2 == 2)
        list_type = ' -';
        set(handles.menu_type,'String',list_type);
      end
      % Case 4-2-5: Flash & Ideal/No-ideal & All/Others
```

```
        if (m1 == 4) & (m2 == 1|m2 == 2)
          list_type = ' -';
          set(handles.menu_type,'String',list_type);
        end
      case 6
        m3 = 6;
        % Case 1-2-5: TD Digma Delta & Ideal and No-ideal Blocks & Others
        if (m1 == 1) & (m2 == 1|m2 == 2)
          list_type = ' -';
          set(handles.menu_type,'String',list_type);
        end
        % Case 3-2-6: Pipeline & No-Ideal & Others
        if (m1 == 3) & (m2 == 2)
          list_type = ' -';
          set(handles.menu_type,'String',list_type);
        end
        % Case 3-2-6: Flash & No-Ideal & Others
        if (m1 == 4) & (m2 == 2)
          list_type = ' -';
          set(handles.menu_type,'String',list_type);
        end
      case 7
        m3 = 7;
        % Case 3-1-7: Pipeline & All ideal models
        if (m1 == 3) & (m2 == 1)
          list_type = ' -';
                    set(handles.menu_type,'String',list_type);
        end
        % Case 3-1-7: Digital correction
        if (m1 == 3) & (m2 == 1)
          list_type = 'RSD_Digital correction to 3 bits|RSD_Digital correction to 4
bits|RSD_Digital correction to 5 bits|RSD_Digital correction to 6 bits|RSD_Digital correction
to 7 bits|RSD_Digital correction to 8 bits|RSD_Digital correction to 9 bits|RSD_Digital
correction to 10 bits|RSD_Digital correction to 11 bits|RSD_Digital correction to 12
bits|RSD_Digital correction to 13 bits|RSD_Digital correction to 14 bits|RSD_Digital correction
to 15 bits|Generic Dig correction with 2 stages|Generic Dig correction with 3 stages|Generic Dig
correction with 4 stages|All digital correction and others...';
          set(handles.menu_type,'String',list_type);
        end
      case 8
        m3 = 8;
        % Case 3-1-8: Pipeline & All models
        if (m1 == 3) & (m2 == 1)
          list_type = ' -';
                    set(handles.menu_type,'String',list_type);
        end
      case 9
        m3 = 9;
        % Case 3-1-9: Pipeline & Others...
        if (m1 == 3) & (m2 == 1)
          list_type = ' -';
                    set(handles.menu_type,'String',list_type);
        end
    end
end

handles.list_type = list_type;
handles.m4=m4;
handles.m3=m3;
```

```
guidata(hObject, handles);



% --- Executes during object creation, after setting all properties.
function menu_comp_CreateFcn(hObject, eventdata, handles)
% hObject    handle to menu_comp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end



% --- Executes on selection change in menu_type.
function menu_type_Callback(hObject, eventdata, handles)
% hObject    handle to menu_type (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = get(hObject,'String') returns menu_type contents as cell array
%        contents{get(hObject,'Value')} returns selected item from menu_type

global valtype;
valtype = get(hObject,'Value');
m2=handles.m2;

switch valtype
   case valtype
      m4 = valtype;
end
handles.m4=m4;
guidata(hObject, handles);



% --- Executes during object creation, after setting all properties.
function menu_type_CreateFcn(hObject, eventdata, handles)
% hObject    handle to menu_type (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end



% --- Executes on button press in button_find.
function button_find_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to button_find (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global find;
find = 0;
m1=handles.m1;
m2=handles.m2;
m3=handles.m3;
m4=handles.m4;
button_state = get(hObject,'Value');

if button_state == get(hObject,'Max')
   % toggle button is pressed
   find = 1;
elseif button_state == get(hObject,'Min')
   % toggle button is not pressed
   find = 0;
end

description(m1,m2,m3,m4,find,handles);

handles.find=find;
guidata(hObject, handles);



% --- Executes on button press in button_open.
function button_open_Callback(hObject, eventdata, handles)
% hObject    handle to button_open (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global open;
global modelsimulated;
global load_ex;

open = 0;
load_ex = 0;
m1=handles.m1;
m2=handles.m2;
m3=handles.m3;
m4=handles.m4;

button_state = get(hObject,'Value');

if button_state == get(hObject,'Max')
   % toggle button is pressed
   open = 1;
elseif button_state == get(hObject,'Min')
   % toggle button is not pressed
   open = 0;
end

handles.open=open;
guidata(hObject, handles);

modelsimulated = archive(m1,m2,m3,m4,open,handles);
```

```
if
((m1==1)&(m2==3)&(m4~=1)&(m4~=5))|((m1==3)&(m2==3)&(m4~=1)&(m4~=8)|((m1==4)
&(m2==3)&(m4~=1)&(m4~=3)))
   load_ex =1;
   handles = state(m1,m3,m4);
   handles.model = modelsimulated;
   handles.load_ex = load_ex;
   handles.open=open;
   guidata(hObject, handles);
   GUIproyecto('file_open_model_Callback',hObject, eventdata, handles);
end
```

- **Description.m**

```
% Show a description of each type

function description(m1,m2,m3,m4,find,handles)

if find == 1
   switch m1
      case 1  % TD Sigma Delta
         switch m2
            case 1
               switch m3
                  case 1
                     switch m4
                        case 1
                           % Case 1-1-1-1:
                           set(handles.description,'String',char('','Library where you can find all
models of TD Sigma Delta: Ideal Blocks.'));
                     end
                  case 2
                     switch m4
                        case 1
                           % Case 1-1-2-1:
                           set(handles.description,'String',char('','Some models of Ideal integrators.'));
                     end
                  case 3
                     switch m4
                        case 1
                           % Case 1-1-3-1:
                           set(handles.description,'String',char('','Some model of Ideal comparators.'));
                     end
                  case 4
                     switch m4
                        case 1
                           % Case 1-1-4-1:
                           set(handles.description,'String',char('','Some model of Ideal quantizers.'));
                     end
                  case 5
                     switch m4
                        case 1
                           % Case 1-1-5-1:
                           set(handles.description,'String',char('','Simulink archive where you can find
all library models.'));
                     end
                  case 6
```

```
                switch m4
                  case 1
                    % Case 1-1-6-1:
                    set(handles.description,'String',char('','Simulink archive where you can put
other models.'));
                end

            end
          case 2
            switch m3
              case 1
                switch m4
                  case 1
                    % Case 1-2-1-1:
                    set(handles.description,'String',char('','Library where you can find all
models of TD Sigma Delta: No-ideal Blocks.'));
                end
              case 2
                switch m4
                  case 1
                    % Case 1-2-2-1:
                    set(handles.description,'String',char('','Model to simulate the switch thermal
noise of SC integrator.'));
                  case 2
                    % Case 1-2-2-2:
                    set(handles.description,'String',char('','Model to simulate the operational
amplifier noise.'));
                  case 3
                    % Case 1-2-2-3:
                    set(handles.description,'String',char('','Model to simulate the operational
finite gain.'));
                  case 4
                    % Case 1-2-2-4:
                    set(handles.description,'String',char('','Model to simulate the operational
slew-rate and gain-bandwith product.'));
                  case 5
                    % Case 1-2-2-5:
                    set(handles.description,'String',char('','Model to simulate the operational
amplifier saturation voltages.'));
                  case 6
                    % Case 1-2-2-6:
                    set(handles.description,'String',char('','Model to simulate all no-idealities in
the same integrator.'));
                end
              case 3
                switch m4
                  case 1
                    % Case 1-2-3-1:
                    set(handles.description,'String',char('','Model to simulate the hysteresis of
comparators.'));
                  case 2
                    % Case 1-2-3-2:
                    set(handles.description,'String',char('','Model to simulate the offset of
comparators.'));
                end
              case 4
                switch m4
                  case 1
                    % Case 1-2-4-1:
```

```
                    set(handles.description,'String',char('','Model to simulate the clock jitter. It
block should be put it at the input.'));
                end
            case 5
                switch m4
                    case 1
                        % Case 1-2-5-1:
                        set(handles.description,'String',char('','Simulink archive where you can find
all library models.'));
                end
            case 6
                switch m4
                    case 1
                        % Case 1-2-6-1:
                        set(handles.description,'String',char('','Simulink archive where you can put
other models.'));
                end
        end
    case 3
        switch m3
            case 1
                switch m4
                    case 1
                        % Case 1-3-1-1:
                        set(handles.description,'String',char('','Library where you can find all
examples of TD Sigma Delta'));
                    case 2
                        % Case 1-3-1-2:
                        set(handles.description,'String',char('Example:','  - Time Discrete Sigma
Delta','  - Low-Pass','  - 2º order','  - Simple','  - Ideal components'));
                    case 3
                        % Case 1-3-1-3:
                        set(handles.description,'String',char('Example:','  - Time Discrete Sigma
Delta','  - Low-Pass','  - 2º order','  - Simple','  - Some real components'));
                    case 4
                        % Case 1-3-1-4:
                        set(handles.description,'String',char('Example:','  - Time Discrete Sigma
Delta','  - ADSL+ application','  - Low-Pass','  - 2º order','  - Cascade','  - Ideal components'));
                    case 5
                        % Case 1-3-1-5:
                        set(handles.description,'String',char('Example:','  - Time Discrete Sigma
Delta','  - ADSL+ application','  - Low-Pass','  - 2º order','  - Cascade','  - Some real
components'));
                    case 6
                        % Case 1-3-1-6:
                        set(handles.description,'String',char('','Simulink archive where you can find
all examples.'));
                end
            case 2
                switch m4
                    case 1
                        % Case 1-3-2-1:
                        set(handles.description,'String',char('','Simulink archive where you can put
other examples.'));
                end
        end
    end
case 2  % TC Sigma Delta
    switch m2
```

```
        case 1
          switch m3
            case 1
              switch m4
                case 1
                  % Case 2-1-1-1:
              end
          end
        case 2
          switch m3
            case 1
              switch m4
                case 1
                  % Case 2-2-1-1:
              end
          end
        case 3
          switch m3
            case 1
              switch m4
                case 1
                  % Case 2-3-1-1:
              end
          end
      end
    case 3  % Pipeline
      switch m2
        case 1
          switch m3
            case 1
              switch m4
                case 1
                  % Case 3-1-1-1:
                  set(handles.description,'String',char('','Library where you can find all models of Pipeline: Ideal Blocks.'));
              end
            case 2
              switch m4
                case 1
                  % Case 3-1-2-1:
                  set(handles.description,'String',char('','Ideal model of analog to digital converter stage.','','It is usually referred to as a redundant signed digit (RSD) converter.','',' Also you can find a generic model.'));
              end
            case 3
              switch m4
                case 1
                  % Case 3-1-3-1:
                  set(handles.description,'String',char('','Ideal model of digital to analog converter stage.','','It is usually referred to as a redundant signed digit (RSD) converter.','',' Also you can find a generic model.'));
              end
            case 4
              switch m4
                case 1
                  % Case 3-1-4-1:
                  set(handles.description,'String',char('','Ideal model to simulate the sample and hold. Also, this block includes the residue gain in the stage.','',' Also you can find a generic model.'));
```

```
                end
            case 5
                switch m4
                    case 1
                        % Case 3-1-5-1:
                        set(handles.description,'String',char('','Model that conteins a complete stage: ADC, DCA y SHA','','ADC: analog to digital converter','DCA: digital to analog converter','SHA: residue gain, sample and hold'));
                end
            case 6
                switch m4
                    case 1
                        % Case 3-1-6-1:
                        set(handles.description,'String',char('','Model that conteins a last stage: only ADC','','ADC: analog to digital converter'));
                end
            case 7
                switch m4
                    case 1
                        % Case 3-1-7-1:
                        set(handles.description,'String',char('','Model to make the digital correction pipeline to 3 bits','','Valid for RSD stages.'));
                    case 2
                        % Case 3-1-7-2:
                        set(handles.description,'String',char('','Model to make the digital correction pipeline to 4 bits','','Valid for RSD stages.'));
                    case 3
                        % Case 3-1-7-3:
                        set(handles.description,'String',char('','Model to make the digital correction pipeline to 5 bits','','Valid for RSD stages.'));
                    case 4
                        % Case 3-1-7-4:
                        set(handles.description,'String',char('','Model to make the digital correction pipeline to 6 bits','','Valid for RSD stages.'));
                    case 5
                        % Case 3-1-7-5:
                        set(handles.description,'String',char('','Model to make the digital correction pipeline to 7 bits','','Valid for RSD stages.'));
                    case 6
                        % Case 3-1-7-6:
                        set(handles.description,'String',char('','Model to make the digital correction pipeline to 8 bits','','Valid for RSD stages.'));
                    case 7
                        % Case 3-1-7-7:
                        set(handles.description,'String',char('','Model to make the digital correction pipeline to 9 bits','','Valid for RSD stages.'));
                    case 8
                        % Case 3-1-7-8:
                        set(handles.description,'String',char('','Model to make the digital correction pipeline to 10 bits','','Valid for RSD stages.'));
                    case 9
                        % Case 3-1-7-9:
                        set(handles.description,'String',char('','Model to make the digital correction pipeline to 11 bits','','Valid for RSD stages.'));
                    case 10
                        % Case 3-1-7-10:
                        set(handles.description,'String',char('','Model to make the digital correction pipeline to 12 bits','','Valid for RSD stages.'));
                    case 11
```

```
                    % Case 3-1-7-11:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 13 bits','','Valid for RSD stages.'));
                case 12
                    % Case 3-1-7-12:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 14 bits','','Valid for RSD stages.'));
                case 13
                    % Case 3-1-7-13:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 15 bits','','Valid for RSD stages.'));
                case 14
                    % Case 3-1-7-14:
                    set(handles.description,'String',char('','Model to make the generic correction pipeline with 2 stages','','You could get 2*(N-1)+1 bits where N represents the number of bits of each stage.'));
                case 15
                    % Case 3-1-7-15:
                    set(handles.description,'String',char('','Model to make the generic correction pipeline with 3 stages','','You could get 3*(N-1)+1 bits where N represents the number of bits of each stage.','','Also, there is a model with different output bits in each stage.'));
                case 16
                    % Case 3-1-7-16:
                    set(handles.description,'String',char('','Model to make the generic correction pipeline with 4 stages','','You could get 4*(N-1)+1 bits where N represents the number of bits of each stage.','','Also, there is a model with different output bits in each stage.'));
                case 17
                    % Case 3-1-7-17:
                    set(handles.description,'String',char('','Models to make the digital correction pipeline in the range between 3 and 15 bits and others','','Valid for RSD and generic stages.','','In this simulink archive you can save other models that you create.'));
                end
            case 8
                switch m4
                    case 1
                    % Case 3-1-8-1:
                    set(handles.description,'String',char('','Simulink archive where you can find all models.'));
                end
            case 9
                switch m4
                    case 1
                    % Case 3-1-9-1:
                    set(handles.description,'String',char('','Simulink archive where you can put other models.'));
                end
        end
    case 2
        switch m3
            case 1
                switch m4
                    case 1
                    % Case 3-2-1-1:
                    set(handles.description,'String',char('','Library where you can find all models of Pipeline: No-ideal Blocks.'));
                end
            case 2
                switch m4
                    case 1
```

```
                    % Case 3-2-2-1:
                    set(handles.description,'String',char('','Model to simulate the ADC with offset in reference voltage.'));
                end
            case 3
                switch m4
                    case 1
                    % Case 3-2-3-1:
                    set(handles.description,'String',char('','Model to simulate the complete stage with finite gain.','','Valid to RSD stages.'));
                    case 2
                    % Case 3-2-3-2:
                    set(handles.description,'String',char('','Model to simulate the complete stage with mismatch between capacitors.','','Valid to RSD stages.'));
                    case 3
                    % Case 3-2-3-3:
                    set(handles.description,'String',char('','Model to simulate the complete stage with SR and BW.','','Valid to RSD stages.'));
                    case 4
                    % Case 3-2-3-4:
                    set(handles.description,'String',char('','Model to simulate the complete stage with offset in reference voltage.','','Valid to RSD stages.'));
                    case 5
                    % Case 3-2-3-5:
                    set(handles.description,'String',char('','Model to simulate the complete stage with jitter noise.','','Put it at the input','','Valid to RSD stages.'));
                end
            case 4
                switch m4
                    case 1
                    % Case 3-2-4-1:
                    set(handles.description,'String',char('','Model to simulate the jitter noise.','','You have to put this block at the input.'));
                end
            case 5
                switch m4
                    case 1
                    % Case 3-2-5-1:
                    set(handles.description,'String',char('','Simulink archive where you can find all models.'));
                end
            case 6
                switch m4
                    case 1
                    % Case 3-2-6-1:
                    set(handles.description,'String',char('','Simulink archive where you can put other models.'));
                end
        end
    case 3
        switch m3
            case 1
                switch m4
                    case 1
                    % Case 1-3-1-1:
                    set(handles.description,'String',char('','Library where you can find all examples of Pipeline'));
                    case 2
                    % Case 1-3-1-2:
```

```
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Ideal components'));
                case 3
                    % Case 3-3-1-3:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Real components',' - Finite Gain'));
                case 4
                    % Case 3-3-1-4:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Real components',' - Mismatch'));
                case 5
                    % Case 3-3-1-5:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Real components',' - Slew-rate and bandwidth'));
                case 6
                    % Case 3-3-1-6:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Real components',' - Offset in reference voltage'));
                case 7
                    % Case 3-3-1-7:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Real components',' - Jitter noise'));
                case 8
                    % Case 3-3-1-8:
                    set(handles.description,'String',char('','Simulink archive where you can find
all examples'));
            end
        case 2
            switch m4
                case 1
                    % Case 3-3-2-1:
                    set(handles.description,'String',char('','Simulink archive where you can put
other examples'));
            end
    end
end
    case 4 % Flash
    switch m2
        case 1
            switch m3
                case 1
                    switch m4
                        case 1
                            % Case 3-1-1-1: Library
                            set(handles.description,'String',char('','Library where you can find all
models of Flash: Ideal Blocks.'));
                    end
                case 2
                    switch m4
                        case 1
                            % Case 4-1-2-1:
                            set(handles.description,'String',char('','Ideal model to generate reference
voltages.','','Its output are inputs of comparators'));
                    end
                case 3
                    switch m4
                        case 1
                            % Case 4-1-3-1:
                            set(handles.description,'String',char('','Ideal model of comparator.'));
```

```
                case 2
                  % Case 4-1-3-2:
                  set(handles.description,'String',char('','Ideal model of comparator with
sample and hold at the input.'));
              end
            case 4
              switch m4
                case 1
                  % Case 4-1-4-1:
                  set(handles.description,'String',char('','Ideal encoder of 63 inputs.'));
              end
            case 5
              switch m4
                case 1
                  % Case 4-1-5-1:
                  set(handles.description,'String',char('','Simulink archive where you can find
all ideal models'));
              end
            case 6
              switch m4
                case 1
                  % Case 4-1-6-1:
                  set(handles.description,'String',char('','Simulink archive where you can put
other ideal models'));
              end
          end
        case 2
          switch m3
            case 1
              switch m4
                case 1
                  % Case 4-2-1-1: Library
                  set(handles.description,'String',char('','Library where you can find all
models of Flash: No-Ideal Blocks.'));
              end
            case 2
              switch m4
                case 1
                  % Case 4-2-2-1:
                  set(handles.description,'String',char('','Real model to generate reference
voltages. Included offet','','Its output are inputs of comparators'));
                end
            case 3
              switch m4
                case 1
                  % Case 4-1-3-1:
                  set(handles.description,'String',char('','Real model of comparator.'));
                case 2
                  % Case 4-2-3-2:
                  set(handles.description,'String',char('','Real model of comparator with relay
and offset.'));
                case 3
                  % Case 4-2-3-3:
                  set(handles.description,'String',char('','Real model of comparator with relay,
offset and pole effect.'));
              end
            case 4
              switch m4
                case 1
```

```
                    % Case 4-2-4-1:
                    set(handles.description,'String',char('','Simulink archive where you can find
all real models'));
                end
            case 5
                switch m4
                    case 1
                    % Case 4-2-5-1:
                    set(handles.description,'String',char('','Simulink archive where you can put
other real models'));
                end
        end
    case 3
        switch m3
            case 1
                switch m4
                    case 1
                    % Case 4-3-1-1:Library
                    set(handles.description,'String',char('','Library where you can find all
models of Flash: Examples.'));
                    case 2
                    % Case 4-3-1-2:
                    set(handles.description,'String',char('','Architecture Flash with:','' -
Generator reference',' - Bank of 63 comparators',' - Converter digital to thermometer',' - Binary
Logic converter'));
                    case 3
                    % Case 4-3-1-3:
                    set(handles.description,'String',char('','Simulink archive where you can find
all examples'));
                end
            case 2
                switch m4
                    case 1
                    % Case 4-3-2-1:
                    set(handles.description,'String',char('','Simulink archive where you can put
other examples'));
                end
        end
    end
  end
else
  % Don't push any find button
end
```

- **Archive.m**

```
% Function to open the selected archive. Also, if the user doesn't push the
% find button,it will appear the text with the description block.

function modelsimulated = archive(m1,m2,m3,m4,open,handles)

modelsimulated = '-';
if open == 1
  switch m1
    case 1
      switch m2
        case 1
          switch m3
            case 1
              switch m4
                case 1
                  % Case 1-1-1-1:
                  set(handles.description,'String',char('','Library where you can find all models of TD Sigma Delta: Ideal Blocks'));
                  arch_library_comp(m1,m2);
              end
            case 2
              switch m4
                case 1
                  % Case 1-1-2-1:
                  set(handles.description,'String',char('','Some models of Ideal integrators'));
                  arch_model_comp(m1,m2,m3,m4);
              end
            case 3
              switch m4
                case 1
                  % Case 1-1-3-1:
                  set(handles.description,'String',char('','Some model of Ideal comparators'));
                  arch_model_comp(m1,m2,m3,m4);
              end
            case 4
              switch m4
                case 1
                  % Case 1-1-4-1:
                  set(handles.description,'String',char('','Some model of Ideal quantizers'));
                  arch_model_comp(m1,m2,m3,m4);
              end
            case 5
              switch m4
                case 1
                  % Case 1-1-5-1:
                  set(handles.description,'String',char('','Simulink archive where you can find all library models'));
                  arch_model_comp(m1,m2,m3,m4);
              end
            case 6
              switch m4
                case 1
                  % Case 1-1-6-1:
                  set(handles.description,'String',char('','Simulink archive where you can put other models.'));
```

```
                    arch_model_comp(m1,m2,m3,m4);
                end
            end
        case 2
            switch m3
                case 1
                    switch m4
                        case 1
                            % Case 1-2-1-1:
                            set(handles.description,'String',char('','Library where you can find all
models of TD Sigma Delta: No-ideal Blocks'));
                            arch_library_comp(m1,m2);
                    end
                case 2
                    switch m4
                        case 1
                            % Case 1-2-2-1:
                            set(handles.description,'String',char('','Model to simulate the switch thermal
noise of SC integrator'));
                            arch_model_comp(m1,m2,m3,m4);
                        case 2
                            % Case 1-2-2-2:
                            set(handles.description,'String',char('','Model to simulate the operational
amplifier noise'));
                            arch_model_comp(m1,m2,m3,m4);
                        case 3
                            % Case 1-2-2-3:
                            set(handles.description,'String',char('','Model to simulate the operational
finite gain'));
                            arch_model_comp(m1,m2,m3,m4);
                        case 4
                            % Case 1-2-2-4:
                            set(handles.description,'String',char('','Model to simulate the operational
slew-rate and gain-bandwith product'));
                            arch_model_comp(m1,m2,m3,m4);
                        case 5
                            % Case 1-2-2-5:
                            set(handles.description,'String',char('','Model to simulate the operational
amplifier saturation voltages'));
                            arch_model_comp(m1,m2,m3,m4);
                        case 6
                            % Case 1-2-2-6:
                            set(handles.description,'String',char('','Model to simulate all no-idealities in
the same integrator'));
                            arch_model_comp(m1,m2,m3,m4);
                    end
                case 3
                    switch m4
                        case 1
                            % Case 1-2-3-1:
                            set(handles.description,'String',char('','Model to simulate the hysteresis of
comparators'));
                            arch_model_comp(m1,m2,m3,m4);
                        case 2
                            % Case 1-2-3-2:
                            set(handles.description,'String',char('','Model to simulate the offset of
comparators'));
                            arch_model_comp(m1,m2,m3,m4);
                    end
```

```
            case 4
              switch m4
                case 1
                  % Case 1-2-4-1:
                  set(handles.description,'String',char('','Model to simulate the clock jitter. It
block should be put it at the input'));
                  arch_model_comp(m1,m2,m3,m4);
              end
            case 5
              switch m4
                case 1
                  % Case 1-2-5-1:
                  set(handles.description,'String',char('','Simulink archive where you can find
all models'));
                  arch_model_comp(m1,m2,m3,m4);
              end
            case 6
              switch m4
                case 1
                  % Case 1-2-6-1:
                  set(handles.description,'String',char('','Simulink archive where you can put
other models.'));
                  arch_model_comp(m1,m2,m3,m4);
              end
          end
        case 3
          switch m3
            case 1
              switch m4
                case 1
                  % Case 1-3-1-1:
                  set(handles.description,'String',char('','Library where you can find all
examples of TD Sigma Delta'));
                  modelsimulated = arch_library_comp(m1,m2);
                case 2
                  % Case 1-3-1-2:
                  set(handles.description,'String',char('Example:','  - Time Discrete Sigma
Delta','  - Low-Pass','  - 2º order','  - Simple','  - Ideal components'));
                  modelsimulated = arch_model_comp(m1,m2,m3,m4);
                case 3
                  % Case 1-3-1-3:
                  set(handles.description,'String',char('Example:','  - Time Discrete Sigma
Delta','  - Low-Pass','  - 2º order','  - Simple','  - Some real components'));
                  modelsimulated = arch_model_comp(m1,m2,m3,m4);
                case 4
                  % Case 1-3-1-4:
                  set(handles.description,'String',char('Example:','  - Time Discrete Sigma
Delta','  - ADSL+ application','  - Low-Pass','  - 2º order','  - Cascade','  - Ideal components'));
                  modelsimulated = arch_model_comp(m1,m2,m3,m4);
                case 5
                  % Case 1-3-1-5:
                  set(handles.description,'String',char('Example:','  - Time Discrete Sigma
Delta','  - ADSL+ application','  - Low-Pass','  - 2º order','  - Cascade','  - Some real
components'));
                  modelsimulated = arch_model_comp(m1,m2,m3,m4);
                case 6
                  % Case 1-3-1-6:
                  set(handles.description,'String',char('','Simulink archive where you can find
all examples'));
```

```
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                end
              case 2
                switch m4
                  case 1
                    % Case 1-3-2-1:
                    set(handles.description,'String',char('','Simulink archive where you can put other examples'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                end
            end
        end
      case 2  % TC Sigma Delta
        switch m2
          case 1
            switch m3
              case 1
                switch m4
                  case 1
                    % Case 2-1-1-1:
                end
            end
          case 2
            switch m3
              case 1
                switch m4
                  case 1
                    % Case 2-2-1-1:
                end
            end
          case 3
            switch m3
              case 1
                switch m4
                  case 1
                    % Case 2-3-1-1:
                end
            end
        end
      case 3  % Pipeline
        switch m2
          case 1
            switch m3
              case 1
                switch m4
                  case 1
                    % Case 3-1-1-1:
                    set(handles.description,'String',char('','Library where you can find all models of Pipeline: Ideal Blocks.'));
                    arch_library_comp(m1,m2);
                end
              case 2
                switch m4
                  case 1
                    % Case 3-1-2-1:
                    set(handles.description,'String',char('','Ideal model of analog to digital converter stage.','','It is usually referred to as a redundant signed digit (RSD) converter.','',' Also you can find a generic model.'));
                    arch_model_comp(m1,m2,m3,m4);
```

```
            end
        case 3
            switch m4
                case 1
                    % Case 3-1-3-1:
                    set(handles.description,'String',char('','Ideal model of digital to analog converter stage.','','It is usually referred to as a redundant signed digit (RSD) converter.','' Also you can find a generic model.'));
                    arch_model_comp(m1,m2,m3,m4);
            end
        case 4
            switch m4
                case 1
                    % Case 3-1-4-1:
                    set(handles.description,'String',char('','Ideal model to simulate the sample and hold. Also, this block includes the residue gain in the stage.','' Also you can find a generic model.'));
                    arch_model_comp(m1,m2,m3,m4);
            end
        case 5
            switch m4
                case 1
                    % Case 3-1-5-1:
                    set(handles.description,'String',char('','Model that conteins a complete stage: ADC, DCA y SHA','','ADC: analog to digital converter','DCA: digital to analog converter','SHA: residue gain, sample and hold'));
                    arch_model_comp(m1,m2,m3,m4);
            end
        case 6
            switch m4
                case 1
                    % Case 3-1-6-1:
                    set(handles.description,'String',char('','Model that conteins a last stage: only ADC','','ADC: analog to digital converter'));
                    arch_model_comp(m1,m2,m3,m4);
            end
        case 7
            switch m4
                case 1
                    % Case 3-1-7-1:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 3 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 2
                    % Case 3-1-7-2:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 4 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 3
                    % Case 3-1-7-3:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 5 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 4
                    % Case 3-1-7-4:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 6 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 5
```

```
                    % Case 3-1-7-5:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 7 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 6
                    % Case 3-1-7-6:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 8 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 7
                    % Case 3-1-7-7:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 9 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 8
                    % Case 3-1-7-8:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 10 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 9
                    % Case 3-1-7-9:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 11 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 10
                    % Case 3-1-7-10:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 12 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 11
                    % Case 3-1-7-11:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 13 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 12
                    % Case 3-1-7-12:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 14 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 13
                    % Case 3-1-7-13:
                    set(handles.description,'String',char('','Model to make the digital correction pipeline to 15 bits','','Valid for RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 14
                    % Case 3-1-7-14:
                    set(handles.description,'String',char('','Model to make the generic correction pipeline with 2 stages','','You could get 2*(N-1)+1 bits where N represents the number of bits of each stage.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 15
                    % Case 3-1-7-15:
                    set(handles.description,'String',char('','Model to make the generic correction pipeline with 3 stages','','You could get 3*(N-1)+1 bits where N represents the number of bits of each stage.','','Also, there is a model with different output bits in each stage.'));
                    arch_model_comp(m1,m2,m3,m4);
                case 16
                    % Case 3-1-7-16:
                    set(handles.description,'String',char('','Model to make the generic correction
```

```
pipeline with 4 stages','','You could get 4*(N-1)+1 bits where N represents the number of bits of each stage.','','Also, there is a model with different output bits in each stage.'));
                    arch_model_comp(m1,m2,m3,m4);
                  case 17
                    % Case 3-1-7-17:
                    set(handles.description,'String',char('','Models to make the digital correction pipeline in the range between 3 and 15 bits and others','','Valid for RSD and generic stages.','','In this simulink archive you can save other models that you create.'));
                    arch_model_comp(m1,m2,m3,m4);
                end
              case 8
                switch m4
                  case 1
                    % Case 3-1-8-1:
                    set(handles.description,'String',char('','Simulink archive where you can find all models.'));
                    arch_model_comp(m1,m2,m3,m4);
                end
              case 9
                switch m4
                  case 1
                    % Case 3-1-9-1:
                    set(handles.description,'String',char('','Simulink archive where you can put other models.'));
                    arch_model_comp(m1,m2,m3,m4);
                end
            end
          case 2
            switch m3
              case 1
                switch m4
                  case 1
                    % Case 3-2-1-1:
                    set(handles.description,'String',char('','Library where you can find all models of Pipeline: No-ideal Blocks.'));
                    arch_library_comp(m1,m2);
                end
              case 2
                switch m4
                  case 1
                    % Case 3-2-2-1:
                    set(handles.description,'String',char('','Model to simulate the ADC with offset in reference voltage.'));
                    arch_model_comp(m1,m2,m3,m4);
                end
              case 3
                switch m4
                  case 1
                    % Case 3-2-3-1:
                    set(handles.description,'String',char('','Model to simulate the complete stage with finite gain.','','Valid to RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                  case 2
                    % Case 3-2-3-2:
                    set(handles.description,'String',char('','Model to simulate the complete stage with mismatch between capacitors.','','Valid to RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                  case 3
                    % Case 3-2-3-3:
```

```
                    set(handles.description,'String',char('','Model to simulate the complete stage
with SR and BW.','','Valid to RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                  case 4
                    % Case 3-2-3-4:
                    set(handles.description,'String',char('','Model to simulate the complete stage
with offset in reference voltage.','','Valid to RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                  case 5
                    % Case 3-2-3-5:
                    set(handles.description,'String',char('','Model to simulate the complete stage
with jitter noise.','','Put it at the input','','Valid to RSD stages.'));
                    arch_model_comp(m1,m2,m3,m4);
                end
              case 4
                switch m4
                  case 1
                    % Case 3-2-4-1:
                    set(handles.description,'String',char('','Model to simulate the jitter
noise.','','You have to put this block at the input.'));
                    arch_model_comp(m1,m2,m3,m4);
                end
              case 5
                switch m4
                  case 1
                    % Case 3-2-5-1:
                    set(handles.description,'String',char('','Simulink archive where you can find
all models.'));
                    arch_model_comp(m1,m2,m3,m4);
                end
              case 6
                switch m4
                  case 1
                    % Case 3-2-6-1:
                    set(handles.description,'String',char('','Simulink archive where you can put
other models.'));
                    arch_model_comp(m1,m2,m3,m4);
                end
            end
          case 3
            switch m3
              case 1
                switch m4
                  case 1
                    % Case 1-3-1-1:
                    set(handles.description,'String',char('','Library where you can find all
examples of Pipeline'));
                    modelsimulated = arch_library_comp(m1,m2);
                  case 2
                    % Case 1-3-1-2:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Ideal components'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                  case 3
                    % Case 3-3-1-3:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Real components',' - Finite Gain'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                  case 4
```

```
                    % Case 3-3-1-4:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Real components',' - Mismatch'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                  case 5
                    % Case 3-3-1-5:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Real components',' - Slew-rate and bandwidth'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                  case 6
                    % Case 3-3-1-6:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Real components',' - Offset in reference voltage'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                  case 7
                    % Case 3-3-1-7:
                    set(handles.description,'String',char('Example:',' - Pipeline',' - 10 bits',' -
Real components',' - Jitter noise'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                  case 8
                    % Case 3-3-1-8:
                    set(handles.description,'String',char('','Simulink archive where you can find
all examples'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                end
              case 2
                switch m4
                  case 1
                    % Case 3-3-2-1:
                    set(handles.description,'String',char('','Simulink archive where you can put
other examples'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                end
            end
        end
    case 4 % Flash
      switch m2
        case 1
          switch m3
            case 1
              switch m4
                case 1
                  % Case 3-1-1-1: Library
                  set(handles.description,'String',char('','Library where you can find all
models of Flash: Ideal Blocks.'));
                  arch_library_comp(m1,m2);
                end
              case 2
                switch m4
                  case 1
                    % Case 4-1-2-1:
                    set(handles.description,'String',char('','Ideal model to generate reference
voltages.','','Its output are inputs of comparators'));
                    arch_model_comp(m1,m2,m3,m4);
                end
              case 3
                switch m4
                  case 1
                    % Case 4-1-3-1:
```

```
                    set(handles.description,'String',char('','Ideal model of comparator.'));
                    arch_model_comp(m1,m2,m3,m4);
                  case 2
                    % Case 4-1-3-2:
                    set(handles.description,'String',char('','Ideal model of comparator with
sample and hold at the input.'));
                    arch_model_comp(m1,m2,m3,m4);
                end
              case 4
                switch m4
                  case 1
                    % Case 4-1-4-1:
                    set(handles.description,'String',char('','Ideal encoder of 63 inputs.'));
                    arch_model_comp(m1,m2,m3,m4);
                end
              case 5
                switch m4
                  case 1
                    % Case 4-1-5-1:
                    set(handles.description,'String',char('','Simulink archive where you can find
all ideal models'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                end
              case 6
                switch m4
                  case 1
                    % Case 4-1-6-1:
                    set(handles.description,'String',char('','Simulink archive where you can put
other ideal models'));
                    modelsimulated = arch_model_comp(m1,m2,m3,m4);
                end
            end
          case 2
            switch m3
              case 1
                switch m4
                  case 1
                    % Case 4-2-1-1: Library
                    set(handles.description,'String',char('','Library where you can find all
models of Flash: No-Ideal Blocks.'));
                    arch_library_comp(m1,m2);
                end
              case 2
                switch m4
                  case 1
                    % Case 4-2-2-1:
                    set(handles.description,'String',char('','Real model to generate reference
voltages. Included offet','','Its output are inputs of comparators'));
                    arch_model_comp(m1,m2,m3,m4);
                  end
              case 3
                switch m4
                  case 1
                    % Case 4-1-3-1:
                    set(handles.description,'String',char('','Real model of comparator.'));
                    arch_model_comp(m1,m2,m3,m4);
                  case 2
                    % Case 4-2-3-2:
                    set(handles.description,'String',char('','Real model of comparator with relay
```

```
and offset.'));
                        arch_model_comp(m1,m2,m3,m4);
                    case 3
                        % Case 4-2-3-3:
                        set(handles.description,'String',char('','Real model of comparator with relay,
offset and pole effect.'));
                        arch_model_comp(m1,m2,m3,m4);
                end
            case 4
                switch m4
                    case 1
                        % Case 4-2-4-1:
                        set(handles.description,'String',char('','Simulink archive where you can find
all real models'));
                        modelsimulated = arch_model_comp(m1,m2,m3,m4);
                end
            case 5
                switch m4
                    case 1
                        % Case 4-2-5-1:
                        set(handles.description,'String',char('','Simulink archive where you can put
other real models'));
                        modelsimulated = arch_model_comp(m1,m2,m3,m4);
                end
        end
    case 3
        switch m3
            case 1
                switch m4
                    case 1
                        % Case 4-3-1-1:Library
                        set(handles.description,'String',char('','Library where you can find all
models of Flash: Examples.'));
                        arch_library_comp(m1,m2);
                    case 2
                        % Case 4-3-1-2:
                        set(handles.description,'String',char('','Architecture Flash with:','' -
Generator reference',' - Bank of 63 comparators',' - Converter digital to thermometer',' - Binary
Logic converter'));
                        modelsimulated = arch_model_comp(m1,m2,m3,m4);
                    case 3
                        % Case 4-3-1-3:
                        set(handles.description,'String',char('','Simulink archive where you can find
all examples'));
                        modelsimulated = arch_model_comp(m1,m2,m3,m4);
                end
            case 2
                switch m4
                    case 1
                        % Case 4-3-2-1:
                        set(handles.description,'String',char('','Simulink archive where you can put
other examples'));
                        modelsimulated = arch_model_comp(m1,m2,m3,m4);
                end
        end
    end
  end
else
  % Don't push any open button
```

```
end
```

- **Arch_library_comp.m**

```
% Open a windows with all components.

function modelsimulated = arch_library_comp(m1,m2)

global modelsimulated;

diract=pwd;
switch m1
  case 1
    % Library of Discrete Time Sigma Delta
    switch m2
      case 1
        cd 'Theory Model/Discrete Time Sigma Delta/Ideal Blocks';
      case 2
        cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
      case 3
        cd 'Theory Model/Discrete Time Sigma Delta/Examples';
    end
  case 2
    % Library of Continuous Time Sigma Delta
    switch m2
      case 1
        cd 'Theory Model/Continuous Time Sigma Delta/Ideal Blocks';
      case 2
        cd 'Theory Model/Continuous Time Sigma Delta/No-ideal Blocks';
      case 3
        cd 'Theory Model/Continuous Time Sigma Delta/Examples';
    end
  case 3
    % Library of Pipeline
    switch m2
      case 1
        cd 'Theory Model/Pipeline/Ideal Blocks';
      case 2
        cd 'Theory Model/Pipeline/No-ideal Blocks';
      case 3
        cd 'Theory Model/Pipeline/Examples';
    end
  case 4
    % Library of Flash
    switch m2
      case 1
        cd 'Theory Model/Flash/Ideal Blocks';
      case 2
        cd 'Theory Model/Flash/No-ideal Blocks';
      case 3
        cd 'Theory Model/Flash/Examples';
    end
end

dirlib=pwd;
path(path,dirlib);

% Open a dialog box for selecting the file to be opened,
% starting in Library directory
```

```
[filename, pathname] = uigetfile( ...
   {'*.mdl', 'All Model and Library Files (*.mdl)'; ...
      '*.*','All Files (*.*)'}, ...
   'Library: Open your model');
cd(diract);
% If "Cancel" is selected then return
if isequal([filename,pathname],[0,0])
   return
% Otherwise construct the fullfilename and Check and load the file.
else
   File = fullfile(pathname,filename);
   open_system(File);
end

modelsimulated = File;
```

- **Arch_model_comp.m**

```
% Open a selected model.

function modelsimulated = arch_model_comp(m1,m2,m3,m4)

diract=pwd;

switch m1
   case 1
      switch m2
         case 1
            switch m3
               % TDSD: Ideal Blocks
               case 1
                  switch m4
                     case 1
                        % Case 1-1-1-1: Library
                  end
               case 2
                  switch m4
                     case 1
                        % Case 1-1-2-1:
                        cd 'Theory Model/Discrete Time Sigma Delta/Ideal Blocks';
                        filename = 'integrator_ideal';
                  end
               case 3
                  switch m4
                     case 1
                        % Case 1-1-3-1:
                        cd 'Theory Model/Discrete Time Sigma Delta/Ideal Blocks';
                        filename = 'comparator_ideal';
                  end
               case 4
```

```
        switch m4
          case 1
            % Case 1-1-4-1:
            cd 'Theory Model/Discrete Time Sigma Delta/Ideal Blocks';
            filename = 'quantizer_ideal';
        end
      case 5
        switch m4
          case 1
            % Case 1-1-5-1:
            cd 'Theory Model/Discrete Time Sigma Delta/Ideal Blocks';
            filename = 'tdsigmadelta_ideal_all';
        end
      case 6
        switch m4
          case 1
            % Case 1-1-6-1:
            cd 'Theory Model/Discrete Time Sigma Delta/Ideal Blocks';
            filename = 'tdsigmadelta_ideal_others';
        end
    end
  case 2
    switch m3
      % TDSD: No-ideal Blocks
      case 1
        switch m4
          case 1
            % Case 1-2-1-1: Library
        end
      case 2
        switch m4
          case 1
            % Case 1-2-2-1:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'integr_real_ktc';
          case 2
            % Case 1-2-2-2:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'integr_real_op';
          case 3
            % Case 1-2-2-3:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'integr_real_gainfinite';
          case 4
            % Case 1-2-2-4:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'integr_real_sr_gbw';
          case 5
            % Case 1-2-2-5:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'integr_real_sat';
          case 6
            % Case 1-2-2-6:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'integr_real_complete';
        end
      case 3
        switch m4
          case 1
```

```
            % Case 1-2-3-1:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'comp_real_hysteresis';
          case 2
            % Case 1-2-3-2:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'comp_real_offset';
        end
      case 4
        switch m4
          case 1
            % Case 1-2-4-1:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'jitter';
        end
      case 5
        switch m4
          case 1
            % Case 1-2-5-1:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'tdsigmadelta_noideal_all';
        end
      case 6
        switch m4
          case 1
            % Case 1-2-6-1:
            cd 'Theory Model/Discrete Time Sigma Delta/No-ideal Blocks';
            filename = 'tdsigmadelta_noideal_others';
        end
    end
  case 3
    switch m3
      % TDSD: Examples
      case 1
        switch m4
          case 1
            % Case 1-3-1-1: Library
          case 2
            % Case 1-3-1-2:
            cd 'Theory Model/Discrete Time Sigma Delta/Examples';
            filename = 'tdsigmadelta_lp_2_ideal_simple.mdl';
          case 3
            % Case 1-3-1-3:
            cd 'Theory Model/Discrete Time Sigma Delta/Examples';
            filename = 'tdsigmadelta_lp_2_real_simple.mdl';
          case 4
            % Case 1-3-1-4:
            cd 'Theory Model/Discrete Time Sigma Delta/Examples';
            filename = 'tdsigmadelta_lp_2_ideal_cascade.mdl';
          case 5
            % Case 1-3-1-5:
            cd 'Theory Model/Discrete Time Sigma Delta/Examples';
            filename = 'tdsigmadelta_lp_2_real_cascade.mdl';
          case 6
            % Case 1-3-1-6:
            cd 'Theory Model/Discrete Time Sigma Delta/Examples';
            filename = 'tdsigmadelta_example_all';
        end
      case 2
```

```
            switch m4
              case 1
                % Case 1-3-2-1:
                cd 'Theory Model/Discrete Time Sigma Delta/Examples';
                filename = 'tdsigmadelta_example_others';
            end
        end
    end
case 2  % TC Sigma Delta
    switch m2
      case 1
        switch m3
          case 1
            switch m4
              case 1
                % Case 2-1-1-1:
            end
        end
      case 2
        switch m3
          case 1
            switch m4
              case 1
                % Case 2-2-1-1:
            end
        end
      case 3
          switch m3
            case 1
              switch m4
                case 1
                  % Case 2-3-1-1:
              end
          end
    end
case 3  % Pipeline
      switch m2
        case 1
          switch m3
            case 1
              switch m4
                case 1
                  % Case 4-1-1-1: Library
              end
            case 2
              switch m4
                case 1
                  % Case 3-1-2-1:
                  cd 'Theory Model/Pipeline/Ideal Blocks';
                  filename = 'ADC_ideal';
              end
            case 3
              switch m4
                case 1
                  % Case 3-1-3-1:
                  cd 'Theory Model/Pipeline/Ideal Blocks';
                  filename = 'DAC_ideal';
              end
            case 4
```

```
    switch m4
      case 1
        % Case 3-1-4-1:
        cd 'Theory Model/Pipeline/Ideal Blocks';
        filename = 'SHA_ideal';
    end
  case 5
    switch m4
      case 1
        % Case 3-1-5-1:
        cd 'Theory Model/Pipeline/Ideal Blocks';
        filename = 'stagecomplete_ideal';
    end
  case 6
    switch m4
      case 1
        % Case 3-1-6-1:
        cd 'Theory Model/Pipeline/Ideal Blocks';
        filename = 'stagelast_ideal';
    end
  case 7
    switch m4
      case 1
        % Case 3-1-7-1:
        cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
        filename = 'digcorrection_3b_rsd';
      case 2
        % Case 3-1-7-2:
        cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
        filename = 'digcorrection_4b_rsd';
      case 3
        % Case 3-1-7-3:
        cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
        filename = 'digcorrection_5b_rsd';
      case 4
        % Case 3-1-7-4:
        cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
        filename = 'digcorrection_6b_rsd';
      case 5
        % Case 3-1-7-5:
        cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
        filename = 'digcorrection_7b_rsd';
      case 6
        % Case 3-1-7-6:
        cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
        filename = 'digcorrection_8b_rsd';
      case 7
        % Case 3-1-7-7:
        cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
        filename = 'digcorrection_9b_rsd';
      case 8
        % Case 3-1-7-8:
        cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
        filename = 'digcorrection_10b_rsd';
      case 9
        % Case 3-1-7-9:
        cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
        filename = 'digcorrection_11b_rsd';
      case 10
```

```
                % Case 3-1-7-10:
                cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
                filename = 'digcorrection_12b_rsd';
              case 11
                % Case 3-1-7-11:
                cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
                filename = 'digcorrection_13b_rsd';
              case 12
                % Case 3-1-7-12:
                cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
                filename = 'digcorrection_14b_rsd';
              case 13
                % Case 3-1-7-13:
                cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
                filename = 'digcorrection_15b_rsd';
              case 14
                % Case 3-1-7-14:
                cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
                filename = 'digcorrection_2stg_gen';
              case 15
                % Case 3-1-7-15:
                cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
                filename = 'digcorrection_3stg_gen';
              case 16
                % Case 3-1-7-16:
                cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
                filename = 'digcorrection_4stg_gen';
              case 17
                % Case 3-1-7-17:
                cd 'Theory Model/Pipeline/Ideal Blocks/Digital Correction';
                filename = 'digcorrection_all';
            end
          case 8
            switch m4
              case 1
                % Case 3-1-8-1:
                cd 'Theory Model/Pipeline/Ideal Blocks';
                filename = 'pipeline_ideal_all';
            end
          case 9
            switch m4
              case 1
                % Case 3-1-9-1:
                cd 'Theory Model/Pipeline/Ideal Blocks';
                filename = 'pipeline_ideal_others';
            end
        end
      case 2
        switch m3
          case 1
            switch m4
              case 1
                % Case 3-2-1-1: Library
            end
          case 2
            switch m4
              case 1
                % Case 3-2-2-1:
                cd 'Theory Model/Pipeline/No-ideal Blocks';
```

```
            filename = 'dac_real_voffset';
          end
        case 3
          switch m4
            case 1
              % Case 3-2-3-1:
              cd 'Theory Model/Pipeline/No-ideal Blocks';
              filename = 'stage_real_finitegain';
            case 2
              % Case 3-2-3-2:
              cd 'Theory Model/Pipeline/No-ideal Blocks';
              filename = 'stage_real_mismatch';
            case 3
              % Case 3-2-3-3:
              cd 'Theory Model/Pipeline/No-ideal Blocks';
              filename = 'stage_real_sr_bw';
            case 4
              % Case 3-2-3-4:
              cd 'Theory Model/Pipeline/No-ideal Blocks';
              filename = 'stage_real_voffset';
            case 5
              % Case 3-2-3-5:
              cd 'Theory Model/Pipeline/No-ideal Blocks';
              filename = 'stage_real_complete';
          end
        case 4
          switch m4
            case 1
              % Case 3-2-4-1:
              cd 'Theory Model/Pipeline/No-ideal Blocks';
              filename = 'jitter';
          end
        case 5
          switch m4
            case 1
              % Case 3-2-5-1:
              cd 'Theory Model/Pipeline/No-ideal Blocks';
              filename = 'pipeline_noideal_all';
          end
        case 6
          switch m4
            case 1
              % Case 3-2-6-1:
              cd 'Theory Model/Pipeline/No-ideal Blocks';
              filename = 'pipeline_noideal_others';
          end
      end
    case 3
      switch m3
        case 1
          switch m4
            case 1
              % Case 1-3-1-1:Library
            case 2
              % Case 1-3-1-2:
              cd 'Theory Model/Pipeline/Examples';
              filename = 'pipeline_10b_ideal.mdl';
            case 3
              % Case 3-3-1-3:
```

```
                cd 'Theory Model/Pipeline/Examples';
                filename = 'pipeline_10b_finitegain.mdl';
              case 4
                % Case 3-3-1-4:
                cd 'Theory Model/Pipeline/Examples';
                filename = 'pipeline_10b_mismatch.mdl';
              case 5
                % Case 3-3-1-5:
                cd 'Theory Model/Pipeline/Examples';
                filename = 'pipeline_10b_sr_bw.mdl';
              case 6
                % Case 3-3-1-6:
                cd 'Theory Model/Pipeline/Examples';
                filename = 'pipeline_10b_voffset.mdl';
              case 7
                % Case 3-3-1-7:
                cd 'Theory Model/Pipeline/Examples';
                filename = 'pipeline_10b_jitter.mdl';
              case 8
                % Case 3-3-1-8:
                cd 'Theory Model/Pipeline/Examples';
                filename = 'pipeline_example_all.mdl';
            end
          case 2
            switch m4
              case 1
                % Case 3-3-2-1:
                cd 'Theory Model/Pipeline/Examples';
                filename = 'pipeline_example_others.mdl';
            end
        end
    end
case 4 % Flash
  switch m2
      case 1
        switch m3
          case 1
            switch m4
              case 1
                % Case 3-1-1-1: Library
            end
          case 2
            switch m4
              case 1
                % Case 4-1-2-1:
                cd 'Theory Model/Flash/Ideal Blocks';
                filename = 'ref_gen_63';
            end
          case 3
            switch m4
              case 1
                % Case 4-1-3-1:
                cd 'Theory Model/Flash/Ideal Blocks';
                filename = 'comp';
              case 2
                % Case 4-1-3-2:
                cd 'Theory Model/Flash/Ideal Blocks';
                filename = 'comp_ts';
            end
```

```
        case 4
          switch m4
            case 1
              % Case 4-1-4-1:
              cd 'Theory Model/Flash/Ideal Blocks';
              filename = 'encoder_63';
          end
        case 5
          switch m4
            case 1
              % Case 4-1-5-1:
              cd 'Theory Model/Flash/Ideal Blocks';
              filename = 'flash_ideal_all';
          end
        case 6
          switch m4
            case 1
              % Case 4-1-6-1:
              cd 'Theory Model/Flash/Ideal Blocks';
              filename = 'flash_ideal_others';
          end
      end
    case 2
      switch m3
        case 1
          switch m4
            case 1
              % Case 3-2-1-1: Library
          end
        case 2
          switch m4
            case 1
              % Case 3-2-2-1:
              cd 'Theory Model/Flash/No-ideal Blocks';
              filename = 'ref_gen_63_offset';
            end
        case 3
          switch m4
            case 1
              % Case 4-2-3-1:
              cd 'Theory Model/Flash/No-ideal Blocks';
              filename = 'comp_offset';
            case 2
              % Case 4-2-3-2:
              cd 'Theory Model/Flash/No-ideal Blocks';
              filename = 'comp_offset_relay';
            case 3
              % Case 4-2-3-3:
              cd 'Theory Model/Flash/No-ideal Blocks';
              filename = 'comp_offset_pole';
          end
        case 4
          switch m4
            case 1
              % Case 4-2-4-1:
              cd 'Theory Model/Flash/No-ideal Blocks';
              filename = 'flash_noideal_all';
          end
        case 5
```

```
                switch m4
                  case 1
                    % Case 4-2-5-1:
                    cd 'Theory Model/Flash/No-ideal Blocks';
                    filename = 'flash_noideal_others';
                end
            end
          case 3
            switch m3
              case 1
                switch m4
                  case 1
                    % Case 4-3-1-1:Library
                  case 2
                    % Case 4-3-1-2:
                    cd 'Theory Model/Flash/Examples';
                    filename = 'flash_offset_relay.mdl';
                  case 3
                    % Case 4-3-1-3:
                    cd 'Theory Model/Pipeline/Examples';
                    filename = 'flash_example_all.mdl';
                end
              case 2
                switch m4
                  case 1
                    % Case 4-3-2-1:
                    cd 'Theory Model/Pipeline/Examples';
                    filename = 'flash_example_others.mdl';
                end
            end
      end
end

dirnew=pwd;

File = fullfile(dirnew,filename);
modelsimulated = File;
open_system(File);

cd(diract);
```

- **State.m**

```
% Function to load the value of variables that we use it to simulate the
% model.

function handles = state(m1,m3,m4)

global handles;

switch m1
  case 1
    switch m3
      % TDSD: Examples
      case 1
        switch m4
          case 1
            % Case 1-3-1-1: Library
          case 2
            % Case 1-3-1-2: tdsigmadelta_lp_2_ideal_simple
            handles.Ts = 1/(2*32*250); % Ts=1/fs=1/M*fb=1/(2*M*BW) con fb:frec de
Nyquist
            handles.FS = 2;
            handles.N = 8;
            handles.bandwidth = 250;
            handles.A = 0.5;
            handles.fsin = 70;
            handles.nsamples = 10;
            handles.minsignalband = 10;
              % No-idealities
            handles.AodB = 0;
            handles.sigmajitter =0;
            handles.SR = 0;
            handles.Tau = 0;
            handles.Eps = 0;
            handles.sigma = 0;
            handles.Aux1 = 0;
            handles.Aux2 = 0;
            handles.Aux3 = 0;
            handles.Aux4 = 0;
          case 3
            % Case 1-3-1-3: tdsigmadelta_lp_2_real_simple
            handles.Ts = 1/(2*32*250); % Ts=1/fs=1/M*fb=1/(2*M*BW) con fb:frec de
Nyquist
            handles.FS = 2;
            handles.N = 8;
            handles.bandwidth = 250;
            handles.A = 0.5;
            handles.fsin = 70;
            handles.nsamples = 10;
            handles.minsignalband = 10;
              % No-idealities
            handles.AodB = 22;
            handles.sigmajitter =30e-6;
            handles.SR = 1e5;
            handles.Tau = 1e-5;
            handles.Eps = 0;
            handles.sigma = 0;
            handles.Aux1 = 1e-12; % Cs
```

```
            handles.Aux2 = 0.1e-3; % Vn
            handles.Aux3 = 0.65; % Vmax/Vmin
            handles.Aux4 = 0;
        case 4
            % Case 1-3-1-4: tdsigmadelta_lp_2_ideal_cascade
            handles.Ts = 1/(2*32*250); % Ts=1/fs=1/M*fb=1/(2*M*BW) con fb:frec de
Nyquist
            handles.FS = 2;
            handles.N = 8;
            handles.bandwidth = 250;
            handles.A = 0.5;
            handles.fsin = 70;
            handles.nsamples = 10;
            handles.minsignalband = 10;
              % No-idealities
            handles.AodB = 0;
            handles.sigmajitter =0;
            handles.SR = 0;
            handles.Tau = 0;
            handles.Eps = 0;
            handles.sigma = 0;
            handles.Aux1 = 0;
            handles.Aux2 = 0;
            handles.Aux3 = 0;
            handles.Aux4 = 0;
        case 5
            % Case 1-3-1-5: tdsigmadelta_lp_2_real_cascade
            handles.Ts = 1/(2*32*250); % Ts=1/fs=1/M*fb=1/(2*M*BW) con fb:frec de
Nyquist
            handles.FS = 2;
            handles.N = 8;
            handles.bandwidth = 250;
            handles.A = 0.5;
            handles.fsin = 70;
            handles.nsamples = 10;
            handles.minsignalband = 10;
              % No-idealities
            handles.AodB = 22;
            handles.sigmajitter =30e-6;
            handles.SR = 1e5;
            handles.Tau = 1e-5;
            handles.Eps = 0;
            handles.sigma = 0;
            handles.Aux1 = 1e-12; % Cs
            handles.Aux2 = 0.1e-3; % Vn
            handles.Aux3 = 0.65; % Vmax/Vmin
            handles.Aux4 = 0;
        case 6
            % Case 1-3-1-6: All
            handles.Ts = 1/(2*32*250); % Ts=1/fs=1/M*fb=1/(2*M*BW) con fb:frec de
Nyquist
            handles.FS = 2;
            handles.N = 8;
            handles.bandwidth = 250;
            handles.A = 0.5;
            handles.fsin = 70;
            handles.nsamples = 10;
            handles.minsignalband = 10;
              % No-idealities
```

```
                handles.AodB = 0;
                handles.sigmajitter =0;
                handles.SR = 0;
                handles.Tau = 0;
                handles.Eps = 0;
                handles.sigma = 0;
                handles.Aux1 = 0;
                handles.Aux2 = 0;
                handles.Aux3 = 0;
                handles.Aux4 = 0;
            end
        case 2
            switch m4
                case 1
                    % Case 1-3-2-1: Others
            end
    end
case 2  % TC Sigma Delta

case 3  % Pipeline
    switch m3
        case 1
            switch m4
                case 1
                    % Case 1-3-1-1: Library
                case 2
                    % Case 1-3-1-2: pipeline_10b_ideal
                    handles.Ts  =  1/(2*160*2.2e6);  %  Ts=1/fs=1/M*fb=1/(2*M*BW)  con  fb:frec  de
Nyquist
                    handles.FS = 2;
                    handles.N=14;
                    handles.bandwidth=2.2e6;
                    handles.A=0.5;
                    handles.fsin=550.0123e3;
                    handles.nsamples=10;
                    handles.minsignalband=10;
                      % No-idealities
                    handles.AodB = 0;
                    handles.sigmajitter =0;
                    handles.SR = 0;
                    handles.Tau = 0;
                    handles.Eps = 0;
                    handles.sigma = 0;
                    handles.Aux1 = 0;
                    handles.Aux2 = 0;
                    handles.Aux3 = 0;
                    handles.Aux4 = 0;
                case 3
                    % Case 1-3-1-3: pipeline_10b_finitegain
                    handles.Ts  =  1/(2*160*2.2e6);  %  Ts=1/fs=1/M*fb=1/(2*M*BW)  con  fb:frec  de
Nyquist
                    handles.FS = 2;
                    handles.N=14;
                    handles.bandwidth=2.2e6;
                    handles.A=0.5;
                    handles.fsin=550.0123e3;
                    handles.nsamples=10;
                    handles.minsignalband=10;
                      % No-idealities
```

```
            handles.AodB = 20*log10(1500);
            handles.sigmajitter =0;
            handles.SR = 0;
            handles.Tau = 0;
            handles.Eps = 0;
            handles.sigma = 0;
            handles.Aux1 = 0;
            handles.Aux2 = 0;
            handles.Aux3 = 0;
            handles.Aux4 = 0;
        case 4
            % Case 1-3-1-4: pipeline_10b_mismatch
            handles.Ts = 1/(2*160*2.2e6); % Ts=1/fs=1/M*fb=1/(2*M*BW) con fb:frec de
Nyquist
            handles.FS = 2;
            handles.N=14;
            handles.bandwidth=2.2e6;
            handles.A=0.5;
            handles.fsin=550.0123e3;
            handles.nsamples=10;
            handles.minsignalband=10;
              % No-idealities
            handles.AodB = 20*log10(1500);
            handles.sigmajitter =0;
            handles.SR = 0;
            handles.Tau = 0;
            handles.Eps = 0;
            handles.sigma = 0;
            handles.Aux1 = 0;
            handles.Aux2 = 0;
            handles.Aux3 = 0;
            handles.Aux4 = 0;
        case 5
            % Case 1-3-1-5: pipeline_10b_sr_bw
            handles.Ts = 1/(2*160*2.2e6); % Ts=1/fs=1/M*fb=1/(2*M*BW) con fb:frec de
Nyquist
            handles.FS = 2;
            handles.N=14;
            handles.bandwidth=2.2e6;
            handles.A=0.5;
            handles.fsin=550.0123e3;
            handles.nsamples=10;
            handles.minsignalband=10;
              % No-idealities
            handles.AodB = 20*log10(1500);
            handles.sigmajitter =0;
            handles.SR = 0;
            handles.Tau = 0;
            handles.Eps = 0;
            handles.sigma = 0;
            handles.Aux1 = 0;
            handles.Aux2 = 0;
            handles.Aux3 = 0;
            handles.Aux4 = 0;
        case 6
            % Case 1-3-1-6: pipeline_10b_voffset
            handles.Ts = 1/(2*160*2.2e6); % Ts=1/fs=1/M*fb=1/(2*M*BW) con fb:frec de
Nyquist
            handles.FS = 2;
```

```
                handles.N=14;
                handles.bandwidth=2.2e6;
                handles.A=0.5;
                handles.fsin=550.0123e3;
                handles.nsamples=10;
                handles.minsignalband=10;
                  % No-idealities
                handles.AodB = 20*log10(1500);
                handles.sigmajitter =0;
                handles.SR = 0;
                handles.Tau = 0;
                handles.Eps = 0;
                handles.sigma = 0;
                handles.Aux1 = 0;
                handles.Aux2 = 0;
                handles.Aux3 = 0;
                handles.Aux4 = 0;
            case 7
                % Case 1-3-1-7: pipeline_10b_real
                handles.Ts = 1/(2*160*2.2e6); % Ts=1/fs=1/M*fb=1/(2*M*BW) con fb:frec de
Nyquist
                handles.FS = 2;
                handles.N=14;
                handles.bandwidth=2.2e6;
                handles.A=0.5;
                handles.fsin=550.0123e3;
                handles.nsamples=10;
                handles.minsignalband=10;
                  % No-idealities
                handles.AodB = 20*log10(1500);
                handles.sigmajitter =0;
                handles.SR = 0;
                handles.Tau = 0;
                handles.Eps = 0;
                handles.sigma = 0;
                handles.Aux1 = 0;
                handles.Aux2 = 0;
                handles.Aux3 = 0;
                handles.Aux4 = 0;
            case 8
                % Case 1-3-1-8: All
          end
        case 2
          switch m4
            case 1
                % Case 1-3-2-1: Others
          end
      end
    case 4
      switch m3
        % Flash: Examples
        case 1
          switch m4
            case 1
                % Case 4-3-1-1: Library
            case 2
                % Case 4-3-1-2: flash_offset_relay
                handles.Ts = 1/(2*32*250); % Ts=1/fs=1/M*fb=1/(2*M*BW) con fb:frec de
Nyquist
```

```
            handles.FS = 2;
            handles.N = 8;
            handles.bandwidth = 250;
            handles.A = 0.5;
            handles.fsin = 70;
            handles.nsamples = 10;
            handles.minsignalband = 10;
              % No-idealities
            handles.AodB = 0;
            handles.sigmajitter =0;
            handles.SR = 0;
            handles.Tau = 0;
            handles.Eps = 0;
            handles.sigma = 0;
            handles.Aux1 = 0;
            handles.Aux2 = 0;
            handles.Aux3 = 0;
            handles.Aux4 = 0;
        end
      case 2
        switch m4
          case 1
            % Case 1-3-2-1: Others
        end
    end
end
```