Customizing Expert with New Functions Using LISA

Introduction

LISA (Language for Interfacing Silvaco Applications)is the scripting language used in conjunction with the layout editor Expert. LISA supports all main program flow controls such as conditional and loop statements, and provides many data types and operators. Xi-scripts is script language, developed as an extension of LISA by application-specific commands of Expert. Xi-scripts provide full control over layout objects and almost all of commands in Expert.

Overview of the New Functional Commands

The following new functions are now available:

- Automatic generation of Guard-Ring
- Automatic maximization of array placement
- Batch generation of bus-type text
- Automatic conversion to comb-shaped resistor (with use of PCELL library)

These new functions are designed to improve the efficiency of layout operations. A new function can be executed in two ways: typing command or with GUI.

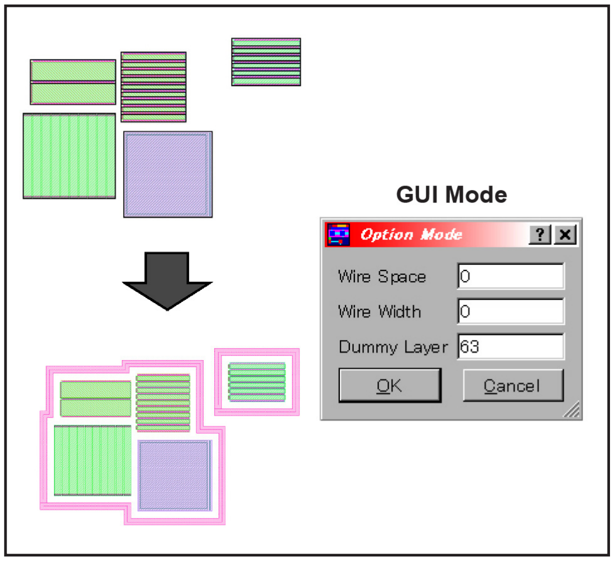
In the first case, when the command is executed with "/h" switch option, a help-dialog will pop up.

New custom commands can be used as general commands by the auto-run at a start-up. Also, if they are added to Expert's custom menu, they can be assigned a keyboard shortcuts.

Automatic Generation of Guard-Ring

This function generates enclosed wires around selected objects (Box/Polygon/Instance/Array) in the editing layer.

If there is no selected object, then a warning message is displayed and the generation will not occurr If the wire cannot be run through among the selected objects, they will be enclosed with the same wire. (See Figure1.)



Wire Space: 0
Wire Width: 0
Dummy Layer: 63

Figure 1. Example of generated Guard-Ring.

The specifiable variable values are spacing from selected object, wire width and dummy layer number.

Syntax : make_ring

XI > make_ring <space> <width> <layer number>

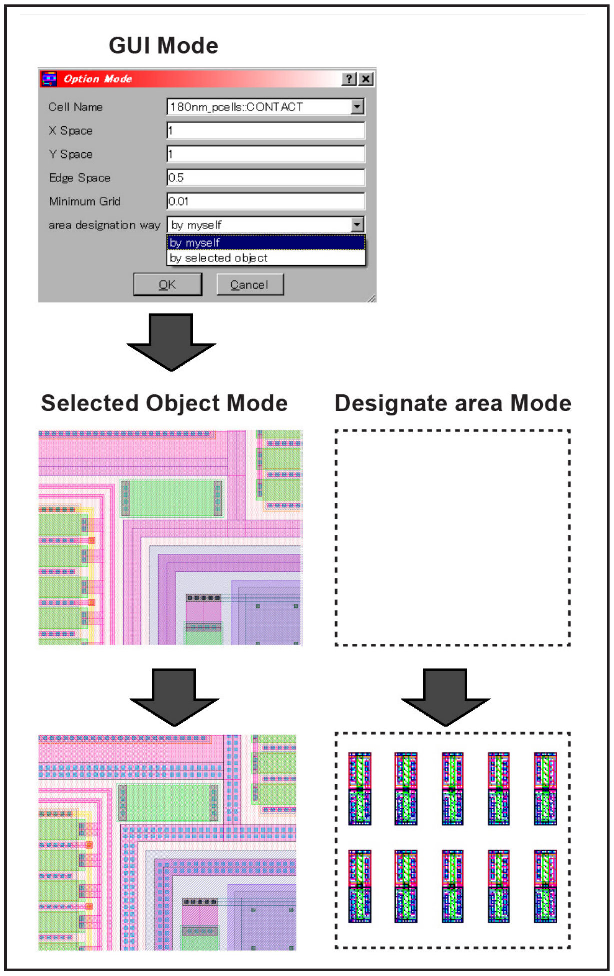
When the argument of either space or width (or both) is not specified, then GUI is displayed.

If a dummy layer number is not specified, "63" is used as the default value.

Automatic Maximization of Array Placement

This function maximizes the number of arrays in specified area. It can be specified in two ways: click any two points (default) or put inside area of selected objects.

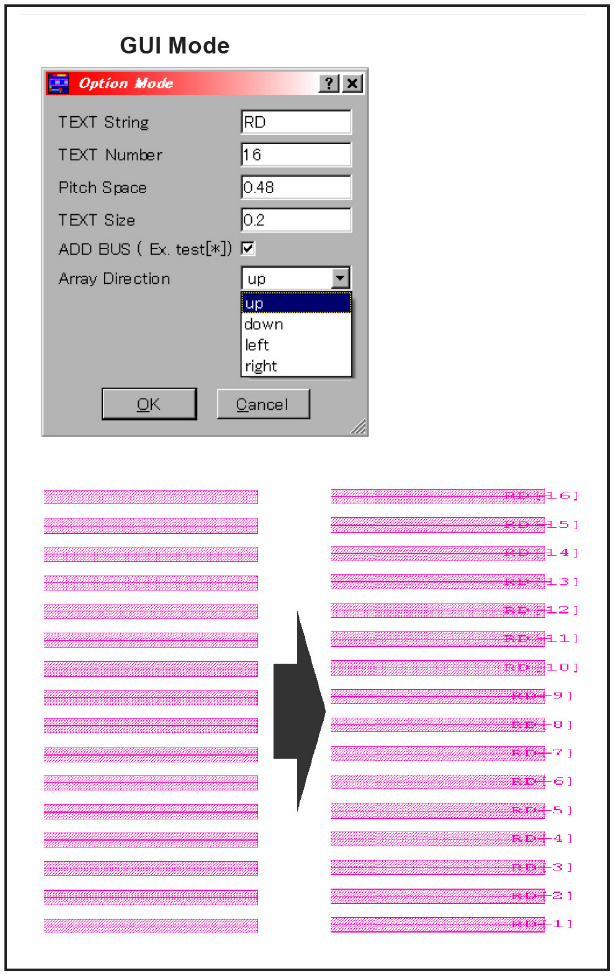
In the latter case, if there is no selected object, then a warning message is displayed and the execution will not occur.



Cell Name	180nm_pcells::CONTACT
X Space	1
Y Space	1
Edge Space	0.5
Minimum Grid	0.01
area designation way	by myself

by myself
by selected object

Figure 2. Example of maximized array placement.



TEXT String	RD
TEXT Number	16
Pitch Space	0.48
TEXT Size	0.2
ADD BUS (Ex. test[*])	✓
Array Direction	up

up
down
left
right

Figure 3. Example of generated bus description texts.

Other specifiable variable values are place-cell-name, array spacing (X-axis and Y-axis), spacing from area edge and minimum grid.

Syntax : max_array

XI > max_array <cell name> <X space> <Y space> <edge space> <minimum grid>

The argument occupies its position in sequence from the front. As switch options, "/d" means to specify by mouse, and "/s" means to specify in selected objects.

If a cell name is not present or specified, then GUI is displayed.

Batch Generation of Bus-Type Text

This function generates bus-type text, so its advantage will be given when it is applied to bit-line, such as memory block.

Bus description is formed as "***[number]", and bus description is selectable to be used or not. The default value is "be used".

Other specifiable variable values are text name, total number, pitch space, size and direction of the generation.

Syntax : text_array

XI > text_array <text name> <number> <pitch> <size>

The argument occupies its position in sequence from the front. As switch options, "/nb" means to ignore the bus description, "/b" or no specification means to apply the bus description.

As other switch options, the created direction must be specified, "/u" means upper, "/d" means lower, "l" means left and "r" means right. If a direction of the generation is not present, then GUI is displayed.

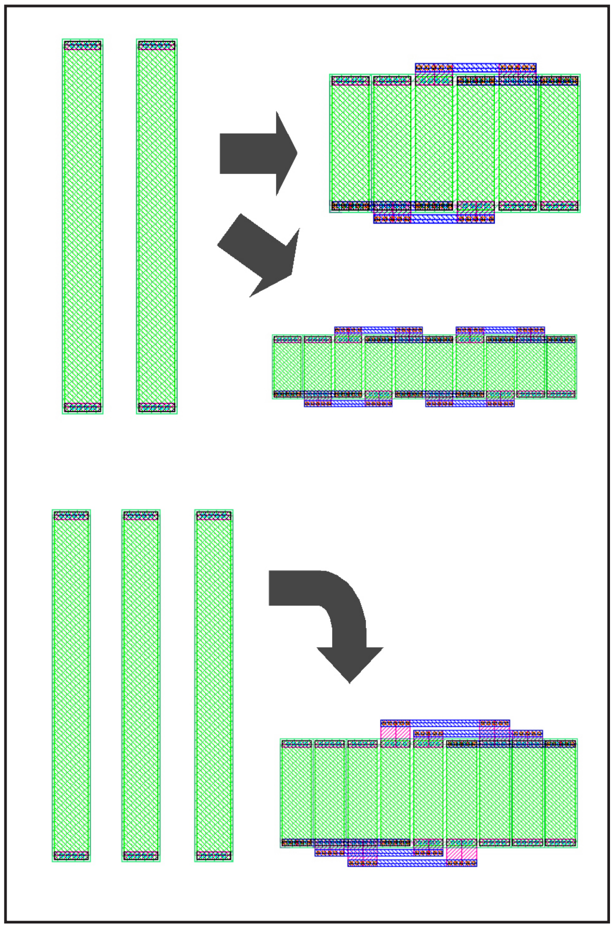


Figure 4. Example of converted resistor to comb-shaped.

Automatic Conversion to Comb-Shaped Resistor (with use of PCELL library)

Analog integrated circuits depend on device matching to obtain much of their precision and performance. Minimal matching can be obtained without much difficulty, and moderate matching can be reliably obtained using interdigitation to produce a common centroid layout.

This function creates automatic interdigitation for resistor Pcells with explicit set of parameters. Following parameter settings are required in a Pcell to work with this function:

Split number

Switch of conversion to comb-shaped

Number of object cell

Transform by position

The execution procedure is to select two or more target pcells and to execute a command. Matched resistors can be constructed from a single material, so only the instances of the same parameterized cell masters can be selected. Then, selected pcells are optimally modified in accordance with the selected cell number, the rotation position and so on.

```
Syntax : comb

XI > comb <split number>
```

An argument is split number only, and there is no switch option. If an argument is not present, “3” as the default number is used. GUI mode is not supported for this function.

Conclusion

Using custom LISA functions reduces design entry time and design rule violations by providing an advanced level of design automation to minimize tedious and repetitive layout tasks. Pcells make it possible to change the size, shape, or contents of each cell instance, without changing the original cell, raising the level of abstraction to the component level.

This application note provides custom script examples to speed up physical layout of custom mixed-signal and analog designs at the device, cell, and block levels. Sample scripts are provided in Appendix A, B C and D.

Appendix A: Automatic Generation of Guard-Ring Sample Scripts

```
define command/replace “make_ring”;
define action
parameter wire_space
parameter wire_width
parameter dummy_layer
parameter Usage
do begin

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Usage !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
if(Usage EQL true) then begin
message_box(“XI> make_ring <option> <space> <width> <dummy_layer_number> \n”
&”\n”
&” <option> \n”
&”/h ... help \n”
&”\n”
&” <space> ... If you skip, GUI is popped up. \n”
&” <width> ... If you skip, GUI is popped up. \n”
&” <dummy_layer_number> ... If you skip, the default, 63 is used. \n”
,{});
end else begin

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! make “drop-down-list” of all cells. !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! (including library-cells, and except for the edited-cell. !!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  all_cells = get_library_cell_list(“”);
  a=0;
  loop begin
    a=a+1;
    if(&all_cells[a] EQL &get_edited_cell_name()) then (leave loop);
  end;
  seq_remove_ith(all_cells, a);
  seq_sort(all_cells);
  cells_buf = buffer_create();
  buffer_insert_at_end(cells_buf, &all_cells);

  all_library = get_activated_library_list();
  b=1;
  loop begin
    b=b+1;
    if(b GTR all_library.size) then (leave loop);
    library_cells=get_library_cell_list(all_library[b]);
```

```
      seq_sort(library_cells);
      c=0;
      loop begin
        c=c+1;
        if(c GTR library_cells.size) then (leave loop);
        library_cells[c]=&all_library[b] &”::” &library_cells[c];
      end;
      buffer_insert_at_end(cells_buf, &library_cells);
    end;
    loop begin
      if(cells_buf.at_end) then (leave loop);
      buffer_scan(cells_buf, “\n”);
      buffer_replace(cells_buf, cells_buf.index, cells_buf.index, “\t”);
      buffer_advance(cells_buf, 1);
    end;
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !!! make “drop-down-list” of all layers. !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !!! (Except for the Derive Layer, Datatype=/0 & GDSnumber>0. !!!!!!!!
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    layers = get_layer_list();
    c=1;
    loop begin
      if(c GTR layers.size) then (leave loop);
      if((get_gds_datatype(&layers[c]) NEQ 0) OR (get_gds_number(&layers[c]) LSS 1))
        then(seq_remove_ith(layers, c))
      else(c=c+1);
    end;
    layers_buf = buffer_create();
    buffer_insert_at_end(layers_buf, &layers);
    loop begin
      if(layers_buf.at_end) then (leave loop);
      buffer_scan(layers_buf, “\n”);
      buffer_replace(layers_buf, layers_buf.index, layers_buf.index, “\t”);
      buffer_advance(layers_buf, 1);
    end;
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !!! Search Selected Instance !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    selectInst = (find objects (SEARCH_ANY_OBJECT) /selected /visible /seq_output);
    if(selectInst.size EQL 0) then (message_box(“No objects selected”,{}))
    else begin
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !!! Create rule !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    if((wire_space*wire_width) EQL 0.0) then begin
      t1={ wire_space , “Wire Space”};
      t2={ wire_width , “Wire Width”};
      t3={ dummy_layer , “Dummy Layer”};
      tt={t1,t2,t3};
      tt1={“ Option Mode “};
      vars1 = (form create (tt) (tt1));

      if(&vars1.type NEQ “Null”) then begin !!! Form
        wire_space=vars1[1];
        wire_width=vars1[2];
        dummy_layer=vars1[3];
      end; !!!!!!! Form Cancel !!!!!!!
    end;
    if((wire_space*wire_width) GTR 0.0) then begin !!! Execution
      Flag = &layers[dummy_layer];
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Make Shape !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
      region mode /merge;
      i=0;
      loop begin
        i=i+1;
        if(i GTR selectInst.size) then (leave loop);
        inst = selectInst[i];
        inst_attr = string_to_lower(inst.shape);
        if((&inst_attr EQL “instance”) OR (&inst_attr EQL “array”)) then begin
          box (inst.bbox.xpos-(wire_space+wire_width/2)) (inst.bbox.ypos-(wire_space+wire_
          width/2)) (inst.bbox.xsize+2*(wire_space+wire_width/2)) (inst.bbox.ysize+2*(wire_
          space+wire_width/2)) /layer = (&Flag);
        end
      elseif((&inst_attr EQL “box”) OR (&inst_attr EQL “polygon”)) then begin
        select object (inst);
        copy layer (&Flag);
        resize selection (wire_space+wire_width/2) /over /grid;
        deselect all;
      end;
    end;
    deselect all;
    select layers (&Flag);
    merge selection; polygonize;
    region mode /normal;
    SearchCriteria = {search_criterion_create(OA_LAYER, (&Flag), EQ)};
    select_shape = (find objects (SEARCH_ANY_SHAPE) /criteria = (SearchCriteria) /selected /
    visible /seq_output);

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Action !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
      j=0;
      loop begin
        j=j+1;
        if(j GTR select_shape.size) then (leave loop);
        vertex = select_shape[j].coords;
        seq_add_last(vertex, (select_shape[j].coords)[1]);

        if(vertex[1].x LSS vertex[vertex.size-1].x) then begin
          vertex[-1].x = vertex[-1].x+wire_width;
          wire (vertex) /width = (wire_width) /extend /extjoint /layer = (get_edited_layer_
          name());
      end
    elseif(vertex[1].x GTR vertex[vertex.size-1].x) then begin
        vertex[-1].x = vertex[-1].x-wire_width;
        wire (vertex) /width = (wire_width) /extend /extjoint /layer = (get_edited_layer_
        name());
        end
      elseif(vertex[1].y LSS vertex[vertex.size-1].y) then begin
        vertex[-1].y = vertex[-1].y+wire_width;
        wire (vertex) /width = (wire_width) /extend /extjoint /layer = (get_edited_layer_
        name());
      end
    elseif(vertex[1].y GTR vertex[vertex.size-1].y) then begin
        vertex[-1].y = vertex[-1].y-wire_width;
        wire (vertex) /width = (wire_width) /extend /extjoint /layer = (get_edited_layer_
        name());
      end;

        end;
        select layers (&Flag);
        delete selection;
      end; !!!!!!! Execution Cancel !!!!!!!
    end; !!!!!!! Search Cancel !!!!!!!
  end; !!!!!!! Usage Cancel !!!!!!!
  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

  end; !!!!!!! define end !!!!!!!
  define argument wire_space /positional /coerce_to=(Float) /default=0.0 /is_default;
  define argument wire_width /positional /coerce_to=(Float) /default=0.0 /is_default;
  define argument dummy_layer /positional /coerce_to=(Integer) /default=63 /is_default;
  define argument h Usage /named /coerce_to=(Boolean) /value=true /is_default;
  set parameter default Usage false;
  complete command;
```

Appendix B: Automatic Maximization of Array Placement Sample Scripts

```
define command/replace “max_array”;
define action
parameter cell_name
parameter x_space
parameter y_space
parameter edge_space
parameter min_grid
parameter how_to
parameter Usage
do begin

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Usage !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
if(Usage EQL true) then begin
message_box(“XI> max_array <option> <cell_name> <x_space> <y_space> <edge_space> <min_
grid> \n”
&”\n”
&” <option> If <cell_name> doesn’t exist, GUI is popped up. \n”
&”/d ... The placed area is decided by your designation area (default mode). \n”
&”/s ... The placed area is decided by selected objects area. \n”
&”/h ... help (this window) \n”
&”\n”
&”   <cell_name>   ... If you skip, GUI is popped up. \n”
&”   <x_space>     ... If you skip, the default, 0.2 is used. \n”
&”   <y_space>     ... If you skip, the default, 0.2 is used. \n”
&”   <edge_space>  ... If you skip, the default, 0.08 is used. \n”
&”   <min_grid>    ... If you skip, the default, 0.01 is used. \n”
,{});
end else begin

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! make “drop-down-list” of all cells. !!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! (including library-cells, and except for the edited-cell. !!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  all_cells = get_library_cell_list(“”);
  a=0;
  loop begin
    a=a+1;
    if(&all_cells[a] EQL &get_edited_cell_name()) then (leave loop);
  end;
  seq_remove_ith(all_cells, a);
  seq_sort(all_cells);
  buf_string = buffer_create();
  buffer_insert_at_end(buf_string, &all_cells);

  all_library = get_activated_library_list();
  b=1;
  loop begin
    b=b+1;
    if(b GTR all_library.size) then (leave loop);
    library_cells=get_library_cell_list(all_library[b]);
    seq_sort(library_cells);
    c=0;
    loop begin
      c=c+1;
      if(c GTR library_cells.size) then (leave loop);
      library_cells[c]=&all_library[b] &"::" &library_cells[c];
    end;
    buffer_insert_at_end(buf_string, &library_cells);
  end;
  loop begin
    if(buf_string.at_end) then (leave loop);
    buffer_scan(buf_string, "\n");
    buffer_replace(buf_string, buf_string.index, buf_string.index, "\t");
    buffer_advance(buf_string, 1);
  end;
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Placement rule !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  if(cell_exists(cell_name) EQL false) then begin
    display(&cell_name &" is not exist.");
    t1={ &buf_string , "Cell Name"};
    t2={ x_space , "X Space"};
    t3={ y_space , "Y Space"};
    t4={ edge_space , "Edge Space"};
    t5={ min_grid , "Minimum Grid"};
    t6={ "by myself\t"
      &"by selected object" , "area designation way"}; !!! How to designate area
    tt={t1,t2,t3,t4,t5,t6};
    tt1={" Option Modes "};
    vars1 = (form create (tt) (tt1));
    if(&vars1.type NEQ "Null") then begin !!! Form
      cell_name = vars1[1];
      x_space = vars1[2];
      y_space = vars1[3];
      edge_space= vars1[4];
      min_grid = vars1[5];
      how_to = vars1[6];
    end; !!!!!!! Form Cancel !!!!!!!
  end;
  if(cell_exists(cell_name) EQL true) then begin
    cell_bbox = get_cell_bbox(cell_name);
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! How to designate area !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    if(&how_to EQL “by selected object”) then begin
      base_obj = (find objects (SEARCH_ANY_OBJECT) /selected /seq_output);
      if(base_obj.size EQL 0) then begin
        placement_area = seq_create_predicted(0);
        message_box(“Please select one instance at least.”,{});
      end else begin
        duplicate selection 0 0;
      convert polygon(base_obj);
      tools cutbyvertex /regions /selected /regionvertices=4;

    base_obj = (find objects (SEARCH_ANY_OBJECT) /selected);
    placement_area = seq_create_predicted(base_obj.size);
    if(base_obj.size NEQ 0) then begin
      current_obj = base_obj.first;
      h=0;
      loop begin
        h=h+1;
        seq_add_last(placement_area , current_obj.bbox);
        if(h GEQ base_obj.size) then (leave loop);
        current_obj = base_obj.next;
      end;
      end;!!!!!!! Search End !!!!!!!
      undo;undo; !!! convert & cutbyvertex
    end;!!!!!!! Selected instance Check
    end;!!!!!!! Mode-1 End !!!!!!!
      if(&how_to EQL “by myself”) then begin
      placement_area = seq_create_predicted(1);
      designate_area = get_rect();
      if(&designate_area NEQ “nil”) then (seq_add_last(placement_area , designate_area));
  end;!!!!!!! Mode-2 End !!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Action !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    i=0;
    loop begin
      i=i+1;
      if(i GTR placement_area.size) then (leave loop);
      current_obj = placement_area[i];
      Nx=floor((current_obj.xsize+x_space-2*edge_space)/(cell_bbox.xsize+x_space));
      Ny=floor((current_obj.ysize+y_space-2*edge_space)/(cell_bbox.ysize+y_space));
      if(Nx*Ny NEQ 0) then begin
        Offset_x=round(((current_obj.xsize-(Nx*cell_bbox.xsize+(Nx-1)*x_space))/2), min_grid);
        !!!Snap to center
        Offset_y=round(((current_obj.ysize-(Ny*cell_bbox.ysize+(Ny-1)*y_space))/2), min_grid);
        !!! Snap to center
        xpos=current_obj.xpos+Offset_x-cell_bbox.xpos;
        ypos=current_obj.ypos+Offset_y-cell_bbox.ypos;
        array (xpos) (ypos) /cell = (&cell_name) /row=(Ny) /col=(Nx) /xdeltarow=0
        ydeltarow=(cell_bbox.ysize+y_space) /xdeltacol=(cell_bbox.xsize+x_space) /ydelta-
        col=0;
      end else (message_box("Warning: designated area is too small !!",{}));
    end;
  end; !!!!!!! Form Cancel !!!!!!!
end; !!!!!!! Usage Cancel !!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

end; !!!!!!! define end !!!!!!!
define argument cell_name /positional /coerce_to=(String) /default="" /is_default;
define argument x_space /positional /coerce_to=(Float) /default=0.2 /is_default;
define argument y_space /positional /coerce_to=(Float) /default=0.2 /is_default;
define argument edge_space /positional /coerce_to=(Float) /default=0.08 /is_default;
define argument min_grid /positional /coerce_to=(Float) /default=0.01 /is_default;
define argument d how_to /named /coerce_to=(String) /value="by myself" /is_default;
define argument s how_to /named /coerce_to=(String) /value="by selected object" /is_default;
set parameter default how_to "by myself";
define argument h Usage /named /coerce_to=(Boolean) /value=true /is_default;
set parameter default Usage false;
complete command;
```

Appendix C: Batch generation of bus-type text Sample Scripts

```
define command/replace “text_array”;
define action
parameter Text_string
parameter Text_Number
parameter Pitch_Space
parameter Text_Size
parameter Bus_mode
parameter Direction_mode
parameter Usage
do begin

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Usage !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
if(Usage EQL true) then begin
message_box(“XI> text_array <option> <text_name> <number> <pitch> <size> \n”
&”\n”
&” <option> If you don’t add the directional option, GUI is popped up. \n”
&”/b ... bus-form (default mode) \n”
&”/nb ... cancel the bus-form \n”
&”/u ... up direction \n”
&”/d ... down direction \n”
&”/l ... left direction \n”
&”/r ... right direction \n”
&”/h ... help (this window) \n”
&”\n”
&”   <text_name>  ... If you skip, the default, \”test\” is used. \n”
&”   <number>     ... If you skip, the default, 1 is used. \n”
&”    <pitch>     ... If you skip, the default, 1.0 is used. \n”
&”   <size>       ... If you skip, the default, 1.0 is used. \n”
,{});
end else begin

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Attach the point you want to create !!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! message_box(“Please click on the first point.”,{});
  p1=get_point();
  if(&p1.type NEQ “Null”) then begin !!! Form-1

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Setting Option !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
  if(Direction_mode.size EQL 0) then begin
    s1={ &Text_string , “TEXT String”};
    s2={ Text_Number , “TEXT Number”};
    s3={ Pitch_Space , “Pitch Space”};
    s4={ Text_Size , “TEXT Size”};
    s5={ Bus_mode , “ADD BUS ( Ex. test[\*])”};
    s6={ “up\t”
      &”down\t”
      &”left\t”
      &”right” , “Array Direction”};
    ss={s1,s2,s3,s4,s5,s6};
    ss1={“ Option Mode “};
    vars1 = (form create (ss) (ss1));

    if(&vars1.type NEQ “Null”) then begin !!! Form-2
      Text_string=vars1[1];
      Text_Number=vars1[2];
      Pitch_Space=vars1[3];
      Text_Size=vars1[4];
      Bus_mode=vars1[5];
      Direction_mode=vars1[6];
    end; !!!!!!! Form-2 Cancel !!!!!!!
  end;
  if(Direction_mode.size GTR 0) then begin !!! Execution

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Action !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    j=0;
    Text_Slope=0;
    PText_string=&Text_string;
    loop begin
      j=j+1;
      if(j GTR Text_Number) then (leave loop);
      if(&Bus_mode EQL “True”) then(PText_string=(&Text_string&”[“&j&”]”));
      X=p1.x;
      Y=p1.y;
      if(&Direction_mode EQL “up”) then begin
        Y=p1.y + (j-1)*Pitch_Space;
      end
    elseif(&Direction_mode EQL “down”) then begin
        Y=p1.y - (j-1)*Pitch_Space;
      end
    elseif(&Direction_mode EQL “left”) then begin
        X=p1.x - (j-1)*Pitch_Space;
```

```
          Text_Slope=90;
        end
      elseif(&Direction_mode EQL "right") then begin
          X=p1.x + (j-1)*Pitch_Space;
          Text_Slope=90;
        end;
        text (X) (Y) (&PText_string) /layer = (get_edited_layer_name()) /slope=(Text_Slope)
        /MiddleCenter /height = (Text_Size) /width = (Text_Size);
        end;

    end; !!!!!!! Execution Cancel !!!!!!!
  end; !!!!!!! Form-1 Cancel !!!!!!!
end; !!!!!!! Usage Cancel !!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
end; !!!!!!! define end !!!!!!!
define argument Text_string /positional /coerce_to=(String) /default="test" /is_default;
define argument Text_Number /positional /coerce_to=(Integer) /default=1 /is_default;
define argument Pitch_Space /positional /coerce_to=(Float) /default=1.0 /is_default;
define argument Text_Size /positional /coerce_to=(Float) /default=1.0 /is_default;
define argument nb Bus_mode /named /coerce_to=(Boolean) /value=false /is_default;
define argument b Bus_mode /named /coerce_to=(Boolean) /value=true /is_default;
set parameter default Bus_mode true;
define argument u Direction_mode /named /coerce_to=(String) /value="up" /is_default;
define argument d Direction_mode /named /coerce_to=(String) /value="down" /is_default;
define argument l Direction_mode /named /coerce_to=(String) /value="left" /is_default;
define argument r Direction_mode /named /coerce_to=(String) /value="right" /is_default;
set parameter default Direction_mode "";
define argument h Usage /named /coerce_to=(Boolean) /value=true /is_default;
set parameter default Usage false;
complete command;
```

Appendix D: Automatic conversion to comb-shaped resistor (with use of PCELL library) Sample Scripts

```
define command/replace “comb”;
define action
parameter split_n
do begin
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!! Set Parameters !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
select_cells = (find objects (SEARCH_INSTANCE) /selected /seq_output);
if(select_cells.size LEQ 1) then (message_box(“Please select two instances at least.”,{}))
else begin
  X=select_cells[1].position.x;
  Y=select_cells[1].position.y;
  pitch=get_cell_bbox(select_cells[1].library&”::”&select_cells[1].cell).xsize;
! pitch=select_cells[1].bbox.xsize;
  i=0;
  loop begin
    i=i+1;
    deselect all;
    if(i GTR select_cells.size) then (leave loop);
    select object (select_cells[i]);
    j=0;
    loop begin
      j=j+1;
      if(j GTR select_cells[i].params.size) then (leave loop);
      if((select_cells[i].paramnames)[j] EQL “N”) then begin
        if((select_cells[i].rotation EQL 0) OR (select_cells[i].rotation EQL 180)) then begin
        modify instance /selected /xpos = (X+pitch*(i-1)) /ypos = (Y)
                                 /params = ({split_n,TRUE,select_cells.size,i})
                                 /parnames = ({“N”,”CommonCentroid”,”Common_N”,”Shape_Style”});
      leave loop;
    end
  elseif((select_cells[i].rotation EQL 90) OR (select_cells[i].rotation EQL 270)) then begin
      modify instance /selected /xpos = (X) /ypos = (Y+pitch*(i-1))
                          /params = ({split_n,TRUE,select_cells.size,i})
                          /parnames = ({“N”,”CommonCentroid”,”Common_N”,”Shape_Style”});
      leave loop;
    end;
  end;
end;!!!!!!! Parameter Loop
end;!!!!!!! Select Loop
end;!!!!!!! Selected instance Check
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
!!! Set Placement !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

end; !!!!!!! define end !!!!!!!
define argument split_n /positional /coerce_to=(Integer) /default=3 /is_default;
complete command;
```