Well Promitiy and STI Stress Effect Parameters Extraction in Guardian LPE

Introduction

Beginning from 90 nm technology process and deeper, well proximity and shallow trench isolation (STI) stress effects become more significant in alteration of MOS device characteristics. Accurate post layout SPICE simulation can't be done without taking these effects into account. New SPICE device models, like BSIM4, have parameters for the simulation of well proximity and STI stress effects.

Simucad's Guardian LPE tool provides special functions for well proximity and STI stress effects parameter calculation that can be used in post layout SPICE simulation. There are two functions that can be used in LISA scripts for generic devices for this purpose: device_enclosure_vector and device_enclosure.

The first function is simpler, but is faster than the second one. The function that should be selected depending on the particular needs of the user.

Enclosure Vector

Well proximity and stress effects parameter calculations can be done via enclosure vector. Guardian LPE provides a pair of LISA functions that permits the calculation of the enclosure vector, and has the following syntax:

1. enclosure_vector = device_enclosure_vector(<layer_name>, max_distance)

2. device_set_enclosure_property(enclosure_vector)

The first function (device_enclosure_vector) calculates the enclosure vector for specified measurement layer <layer_name> over device seed shape. The device seed shape must satisfy the following conditions:

- be rectangular;
- be completely overlapped by measurement layer polygon;
- two opposite sides of the seed shape must coincide with edges on measurement layer.

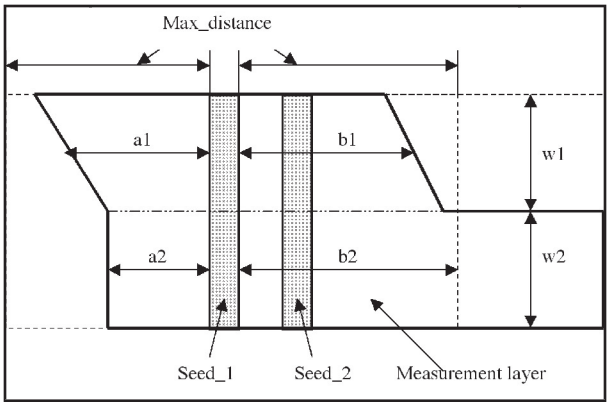

Figure 1. Enclosure vector for device shape Seed_1. Measurement layer is split into two trapezoids, so enclosure vector has two triplets (a1, b1, w1) and (a2, b2, w2). Value b2 is defined by Max_distance. The device shape Seed_2 will have another enclosure vector.

The measurement layer polygon is split into trapezoids with sides that are parallel to the coincident edges of the layer and device seed shape. Each trapezoid provides three values a, b, and w for the enclosure vector element (Figure 1). The max_distance parameter defines the size of the search window and so defines the maximum value for a and b.

The number of elements in the enclosure vector corresponds to the number of trapezoids and can be obtained via the SIZE field of the enclosure vector (enclosure_vector.size).

As mentioned above, the enclosure vector element has three fields a, b, and w, and access to these fields can be obtained in the following way: enclosure_vector[i].a, enclosure_vector[i].b, enclosure_vector[i].w, where i is the element index.

If the measured layer covers several device seed shapes the enclosure vector is calculated for each seed shape separately. During the enclosure vector calculation all other device seed shapes are ignored. If the device seed shape doesn't satisfied the required conditions, the a, b, and w values are set to -1.

The second function (device_set_enclosure_property) is used to output the enclosure vector to a netlist in the following format:

sa1=... sb1=... sw1=... [sa2=... sb2=... sw2=... [...]]

Example

The following example shows how to use the enclosure vector functions in a user defined LISA procedure:

```
define procedure NGATE
do begin
  W = 0.0;
  L = 0.0;
  PD = 0.0;
  PS = 0.0;
  AS = 0.0;
  AD = 0.0;

  AREA = devicearea("ngate", "");
  W1 = device_perimeter(RELBUTTING,
"ngate", "S");
  W2 = device_perimeter(RELBUTTING,
"ngate", "D");
  W = (W1 + W2) / 2;
  IF (W NEQ 0.0) THEN (L = AREA / W);

  AD = devicearea("D", "");
  AS = devicearea("S", "");
  PD = device_perimeter(RELNONE, "D",
"");
  PS = device_perimeter(RELNONE, "S",
"");

!!! deviceenclosurevector !!!
  ENCLVEC = deviceenclosurevector("act",
1000.0);
!!!

  deviceset_property("L", L);
  deviceset_property("W", W);
  deviceset_property("PD", PD);
  deviceset_property("PS", PS);
  deviceset_property("AD", AD);
  deviceset_property("AS", AS);

!!! devicesetenclosure_property !!!
  devicesetenclosure_property(ENCLVEC);
!!!

end;
```

An example netlist output would be:

```
MI2 d g 90 6 NMOS  sb2=2.3e-006
+sa2=1.2e-005 sw2=2.1e-006 sb1=2.3e-006
+sa1=1.5e-005 sw1=6e-006 AS=72.09P
AD=18.63P PS=34U PD=20.8U W=8.1U L=3.8U
MI1 90 5 s 6 NMOS  sb2=1.5e-005
+sa2=9e-007 sw2=2.1e-006 sb1=1.5e-005
+sa1=3.9e-006 sw1=6e-006 AS=25.29P
AD=72.09P PS=24U PD=34U W=8.1U L=2.2U
```

Well Proximity Enclosure Function

The well proximity enclosure function can be used for the calculation of well proximity and shallow trench isolation. The corresponding LISA function for Guardian LPE has the following syntax:

```
enclosurevector = deviceenclosure (direction,
            <baselayeror_pin>,
            <measlayeror_pin>,
            <orientlayeror_pin>,
            maxdistance)
```

where enclosure_vector is a vector of triplets (a, b, w), direction is the measurement orientation with respect to the coincident edges of the base layer <base_layer_or_pin> and orientation layer <orien_layer_or_pin>, and can have two values ORIEN_PERPENDICULAR and ORIEN_PARALLEL <meas_layer_or_pin> is the name of the measurement layer that encloses the base layer. See Figures 2 and 3 for more detaills.

The following are the function input arguments conditions .

The base layer can be the device layer, any pin or auxiliary layer, or pin name. The base layer object must be rectangular and completely overlapped by the measurement layer polygon.

The measurement layer can be any pin or auxiliary layer, or pin name. The measurement layer polygon must completely overlap the rectangular base layer object.

The orientation layer can be any pin or auxiliary layer, or pin name. The polygon(s) from the orientation layer must have at least one coincident edge with the base layer object. If there are several coincident edges then all such edges should be parallel each other.

The base, measurement and orientation layers must all be different layers.

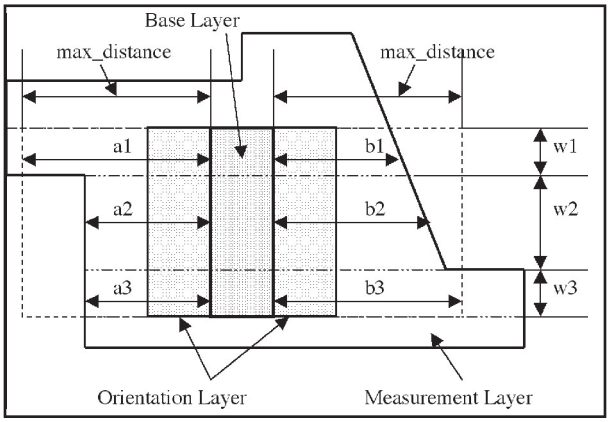

Figure 2. Enclosure function with ORIEN_PERPENDICULAR argument. Overlapped part of measurement layer is split into three trapezoids, so the output enclosure vector has three triplets (a1, b1, w1), (a2, b2, w2), and (a3, b3, w3). Values a1 and b3 are defined by max_distance.

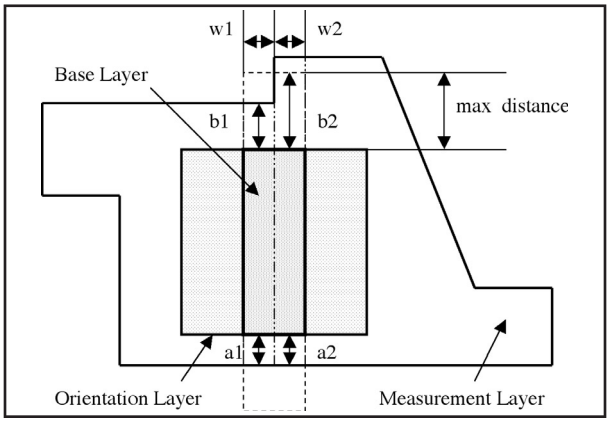

Figure 3. Enclosure function with ORIEN_PARALLEL argument. Overlapped part of measurement layer polygon is split into two trapezoids, so the output enclosure vector has two triplets (a1, b1, w1) and (a2, b2, w2). Value b2 is defined by max_distance.

The measurement layer polygon is split into trapezoids, with sides that are parallel to the coincident edges of the base layer rectangle and orientation layer polygon for ORIEN_PARALLEL direction argument or are perpendicular to the coincide edges for ORIEN_PERPENDICULAR direction argument. Each trapezoid provides three values a, b, and w for the enclosure vector element (Figure 2, Figure 3). The max_distance parameter defines the size of the search window and so defines the maximum value for a and b.

There is a additional function (device_set_named_enclosure_property) which is used to output the enclosure vector with a special name to the netlist:

```
device_set_named_enclosure_property(<name>,
enclosure_vector)
```

where <name> is a string variable.

The output will have the following format:

```
<name>a1=...<name>b1=...<name>w1=...[<name>
a2=...<name>b2=...<name>w2=...[..]]
```

Example

The following example shows how to use the enclosure functions in a user defined LISA procedure:

```
define procedure PGATE
do begin
  W = 0.0;
  L = 0.0;
  PD = 0.0;
  PS = 0.0;
  AS = 0.0;
  AD = 0.0;
  AREA = device_area("pgate", "");
  W1 = device_perimeter(REL_BUTTING,
"pgate", "S");
  W2 = device_perimeter(REL_BUTTING,
"pgate", "D");
!  W = (W1 + W2) / 2;
  IF (W NEQ 0.0) THEN (L = AREA / W);

  AD = device_area("D", "");
  AS = device_area("S", "");
  PD = device_perimeter(REL_NONE, "D",
"");
  PS = device_perimeter(REL_NONE, "S",
"");

  c = device_count("D");
  net_name = device_pin_net("S");

  device_set_property("L", L);
  device_set_property("W", W);
  device_set_property("PD", PD);
  device_set_property("PS", PS);
  device_set_property("AD", AD);
  device_set_property("AS", AS);

! PERP and PARA are enclosure vectors
  PERP = device_enclosure(ORIEN_PERPENDIC-
ULAR, "pgate", "nwell", "S", 1000.0);
  PARA = device_enclosure(ORIEN_PARALLEL,
"pgate", "nwell", "S", 1000.0);

  N_PERP = PROX_PERP.SIZE;
  E_PERP = 0.0;
  I = 0;
```

```
  LOOP
   BEGIN
     I = I + 1;
     EPERP = EPERP + (PROXPERP[I].a +
PROXPERP[I].b) * PROXPERP[I].w;
     IF (I EQL NPERP) THEN (LEAVE LOOP);
   END;
   EPERP = EPERP / L / 2;

  NPARA = PROXPARA.SIZE;
  EPARA = 0.0;
  I = 0;
  LOOP
   BEGIN
     I = I + 1;
     EPARA = EPARA + (PROXPARA[I].a +
PROXPARA[I].b) * PROXPARA[I].w;
     IF (I EQL NPARA) THEN (LEAVE LOOP);
   END;
   EPARA = EPARA / W / 2;

  device_set_named_enclosure_
property("PERP", PERP);
  device_set_named_enclosure_
property("PARA", PARA);
  device_set_property("EPERP", EPERP);
  device_set_property("EPARA", EPARA);

end;
```

An example netlist output would be:

```
MI1 2 G 1 NW PMOS  EPARA=9.61832e-007
+EPERP=2.98389e-005 PARAb1=3e-006
+PARAa1=4e-006 PARAw1=3.6e 006
+PERPb1=8.4e-006 PERPa1=8e-006
PERPw1=1.31e-005
+sb1=5.5e-006 sa1=4.1e-006 sw1=1.31e-005
+AS=53.71P AD=72.05P PS=34.4U PD=37.2U
+W=13.1U L=3.6U
```

Conclusion

This application note describes the new functions for well proximity and STI stress effects parameter calculations in Guardian LPE. These features permit Guardian LPE to be successfully used for device layout parameter extraction for the latest (<90nm) CMOS technology processes.