COMMITTEE ACTION ITEM

---

* There are issues with analog primitives; having the default discipline of electrical in the language would not allow for using these blocks in mechanical & other systems easily. Some background
  - electrical was just defined as a starting point
  - it was agreed that there is a need to override the discipline of the analog primitive on a instance to instance basis
  - An option of having compile time directives to overcome this issue was discussed
    - a default analog discipline (for analog primitives)
    - a default digital discipline (for digital primitives)
    - default discipline

Need to address these items and others related in our list of open issues

Action Jon

Action Items from our list being addressed here are

#13 (806 / 8_3.6) Default Discipline
This applies to discrete disciplines only. Requires further definition (e.g. relationship with `reset_all).

#46 (910 / 9_33) Default_discipline only for digital?
Is *default_discipline only applicable to digital? If so then need to remove references to 'default_discipline electrical e.g. p3-20 of 2.0 LRM.*
Recommendation Resolve analog default_discipline (section 11.1) and then ensure that this section is in alignment.

#82 (939 / 9_9) Disciplines of analog primitives How to set and defaults
Analog primitives cannot via the language get a discipline defined so we need to specify a default value and a method for setting the disciplines. Recommendation Lots of possibilities, part of discipline resolution method, use OOMR declarations, default_analog_discipline (of course compiler directives suck in most of today's solutions)

#83 (939 / 9_30) Compatible disciplines to analog primitive
Certain primitives are not limited to electrical domain.

Primitives like sources, resistors, capacitors, ... can often be used in mechanical and other domains but if they are electrical then they will not be compatible.

Recommendation Analog primitives should default to domain continuous (neutral) and the discipline should resolve as follows

(THIS NEEDS MORE WORK!)

i. The discipline of defined primitives and behavioral blocks connected to each port
- If incompatible disciplines exists then error
ii. The global analog discipline
iii. The default discipline

The above assumes that the default discipline will be electrical for continuous domains and that there is a mechanism(s) for setting the global discipline and the discipline of specific ports. Disciplines of ports could be set via OOMR discipline declarations (mechanical top.I1.I2.R1.a)

where a is the port name.

Some of the basic requirements that I tried to deal with are

1) Compiler directive based solutions should be avoided. These are being moved away from and are not viable solutions in many of todays leading edge solutions. One of the key reasons is that the input is coming from libraries vs a single file stream which makes compiler directives less useful.

2) Defaults need to be defined and predictable

3) A high desire to use SPICE primitives in non-electrical domains (this implies that SPICE primitives must have a way to set or override disciplines)

4) Clarify the LRM better to answer the above issues

Here are the recommendations

1) The "default" discipline of analog "SPICE" primitives shall be electrical [#82 (939 / 9_9)]

2) The default discipline of digital primitives shall be set by the default_discipline compiler flag as defined in the LRM or a simulator (vendor) specific option to set the default_discipline directive. [#13 (806 / 8_3.6) & #46 (910 / 9_33)]

3) The compiler directive `default_discipline shall apply only to the discrete domain, it shall not effect nets of continuous domain. [#46 (910 / 9_33)]

4) The scope options on default discipline in section 3.6 / 11.1 / 3.7 should be eliminated. This is not listed in our list of items to cover but is impossible for simulators to support. The scope option is needed during compile phase yet scope is not known until the elaboration phase. We should eliminate the scope argument off of the default_discipline syntax and remove precedence 3 and 4 in section 3.7 and remove the word scope out of precedence 6.

5) Clearly state that the connect statement resolveto feature shall not be used to set the disciplines of simulator primitives but only for discipline resolution. It may be used to resolve between multiple analog compatible disciplines but not as a method to set the discipline of analog primitives.

(This may not be needed but in some of the questions on this topic and the resolveto rules I believe this was discussed)

6) The disciplines of SPICE primitives shall only use the default discipline if the discipline cannot be resolved via other methods. Methods shall include [#82 (939 / 9_9) & #83 (939 / 9_30)]

a) A specific method of setting of the discipline as an attribute on the component

b) Resolution based on other continuous modules attached to the net at the specific level of hierarchy

To address method (a) the following is provided

Goal Provide users the power of setting disciplines of the same primitive to different values in the same module (schematic). For example an electrical capacitor as well as a mechanical mass

using the internal SPICE capacitor primitive in the same module cannot be set differently without this feature. In Verilog-2001 they have added attributes which we could use thereby allowing vendors whom chose to not support this an legal out or we could just add a special property on any analog primitive. The following show several options

Option1

```
 resistor #(.r(1k)) (* integer port_discipline="electrical" ; *) r1 (node1, node2); // not needed as default
 resistor #(.r(1k)) (* integer port_discipline="rotational" ; *) r2 (node1, node2);
```

Option2

```
 resistor #(.r(1k), .port_discipline(electrical)) r1 (node1, node2);  // not needed as default
 resistor #(.r(1k), .port_discipline(rotational)) r2 (node1, node2);
```

Option 3 casting of the net expressions.

```
 resistor #(.r(1k)) r1 (electrical node1, electrical node2);  // not needed as default
 resistor #(.r(1k)) r2 (rotational node1, rotational node2);'
```

This had the added advantage that we could extend it to

```
 vcvs #(.gain(10) motor1 (electrical in, electrical gnd, rotational pos, rotational r_gnd);
```

The issue with option1 is that we need to clarify that this attribute is only applicable to built-in analog primitives and not a mechanism to override disciplines of modules. It is also verbose but on the other hand it is supported by the standard (I364-2001). If we desired at a later time this could be opened up to non primitives but would hold back for now.

The issue with option2 is that we are adding a new property to analog primitives that all simulators must know about whereas an attribute is ignored if you don't know about it. It is certainly an abuse of parameter passing and one we should avoid without good reason.

Option 3 is another variation that has some pro's and con's that could be discussed and might allow us to do further extensions. This might make netlisting a big more interesting but this can be discussed in committee.

My recommendation is to use the syntax of option 1 but I am open to option 2 or 3. In all cases we are not allowing the ability to have analog primitives with different disciplines at different ports but again this could be opened up at a later point.

An example of method (b) is provided below

```
module top();

blkA   i1 (a);
blkB   i2 (a);

endmodule

module blkA (a);
  output a;
  logic a;
  reg a;

module blkB (a);
   electrical gnd;
   ground gnd;
   blkC  i1 (a);

module blkC (a);
    output a;
     bicmosA a;
     analog
```

```
      initial a=1;              resistor #(.r(1k)) r1(a,gnd);          V(a) <+ 5;
endmodule                  endmodule                          endmodule
```

In the above example the discipline of the analog resistor primitive in blkB would resolve to discipline bicmosA. It would do this since no attribute was found but since blkC is connected to the same net (top.i2.a) and it has a compatible continuous discipline if you look at the vpiLoConn.

THE FOLLOWING ARE PROVIDED AS BACKGROUND TO THE DECISION PROCESS BUT ARE NOT RECOMMENDATIONS AND HAVE BEEN RULED OUT OF MY RECOMMENDATIONS.

1) Essentially do nothing, make disciplines for primitives hard coded and not changeable

Originally folks thought we could use wrappers around these primitives to map to other disciplines but further investigation showed that this would result in an error due to incompatible disciplines in most cases. Even so the primary value of SPICE primitives is to simulate electrical circuits so requiring for other applications where disciplines would not be compatible to create these primitives behaviorally is probably defendable. Thus this change would make SPICE primitives only usable with disciplines compatible with the electrical discipline.

This would essentially be covered and require by recommendations 1- 4 above with 5 and 6 not needed but with recommendation 1 being changed as follows

1) The discipline of SPICE primitives shall be electrical

The word default is no longer needed

2) Provide basic override ability of default via compiler directives

Electrical and Logic are just example disciplines (not hardcoded) yet if the user creates a different "base" discipline for analog there is no way to force blocks to use it. This provide a crude method to also address the above issue via directives although the language is moving away from these. If the user used electrical5v and electrical3v disciplines and they were in separate modules compiler directives could override the default discipline to correctly match the value. Problem here is that standard moving away from directives due to compiled mode simulators. Also this only works if a module has only one analog domain or if the user is manually creating the module/netlist.

```
`define default_analog_discipline electrical5v
module blocka ();
 .....
endmodule
`undefine default_analog_discipline

`define default_analog_discipline electrical3v
module blocka ();
 .....
endmodule
`undefine default_analog_discipline
```

<u>Additional things to do to support this change</u>

a) We need to create a new default_analog_discipline

b) We should also either create a new default_digital_discipline and retire the default_discipline OR limit the default_discipline to digital only. The later is my proposal since that is how it is primarily being used. (I doubt we will break one design by limiting it to only digital.)

With the above changes we would also still do the above recommendations 1-4 and again 5 and 6 are not needed, recommendation 6 is provided via the `deafault_analog_discipline method.

Required LRM changes for the proposed changes

red letters are addition
silver letters are deletions

3.6 Default discipline

Verilog-AMS HDL supports the **`default_discipline** compiler directive. This directive specifies a default discrete discipline to be applied to any discrete net which does not have an explicit discipline declaration. Its syntax is shown in Syntax 3-8.

The scope of this directive is similar to the scope of the **`define** compiler directive. The default discipline is applied to all discrete signals without a discipline declaration which appear in the text stream following the use of the `‘default_discipline` directive until either the end of the text stream or another `‘default_discipline` directive with the same combination of qualifier and scope (if applicable) is found in the subsequent text. Therefore, more than one `‘default_discipline` directive can be in force simultaneously, provided each differs in scope, qualifier, or both.

In addition to `reset_all if this directive is used without a discipline name, it turns off all currently active default disciplines without setting a new default discipline. Subsequent signals without a discipline shall be associated with the empty discipline.

11.1 `default_discipline

`default_discipline** controls the `net_discipline` type created for implicit net declarations (see 3.4.6). It can be used only outside of module definitions. It affects all modules which follow the directive, even across source file boundaries. Multiple **`default_discipline directives are allowed. The latest occurrence of this directive in the source controls the discipline type of discrete nets which are implicitly declared. Syntax 11-1 shows the syntax for this directive.

3.6.1 or 3.4.3.4 (New Section to insert) **Disciplines of Primitives**

With internal primitives the discipline of the vpiLoConn to be used in discipline resolution during a mixed signal simulation must be known. For digital primitives the domain is discrete and thus the discipline is set via the default_discipline directive as it is for digital modules. For analog primitives, the discipline will be the defined by the attribute port_discipline on that instance. If no attribute is found then it will acquire the discipline of other compatible continuous disciplines connected to that net segment. If no disciplines are connected to that net then the default discipline is set to electrical. This is further described in section E.3.3

3.7 Discipline precedence

...
.
2. The local declaration of the net in the module to which it belongs, e.g.,

```
module example2;
electrical net;
```

endmodule

3. `default_discipline with qualifier and scope, e.g.,
**`default_discipline** `electrical` **trireg** `example1.instance5;`

4. `default_discipline with scope only, e.g.,
**`default_discipline** `electrical example1.instance5;`

35. `default_discipline` with qualifier only, e.g.,
`default_discipline** `electrical` **trireg;

46. `default_discipline` without qualifier or scope, e.g.,
**`default_discipline** `electrical;`

8.7.2 Discipline resolution connect statement

The discipline resolution connect statement specifies a single discipline to use during the discipline resolution process when multiple nets with compatible discipline are part of the same mixed net, as shown in Syntax 8-8.

Syntax 8-8—Syntax for connect configuration resolveto statements

(Syntax box)

where *discipline_list* is the list of compatible disciplines and *discipline_identifier* is the discipline to be used. Use of the resolveto statement cannot be used to force the discipline of an analog primitive to another value. (Not sure if this is needed but someone in my discussions proposed doing this. If someone tried to use these with another discipline they would fail for being incompatible.)

E.3 Preferred primitive, parameter, and port names

Table E.1 shows the required names for primitives, parameters, and ports which are otherwise unnamed in SPICE. For connection by order instead of by name, the ports and parameters shall be given in the order listed. The default discipline of the ports for these primitives shall be `electrical` and their descriptions shall be `inout`. (already in LRM)

E.3.3

In order to provide the ability to use analog primitive in any design, including mixed disciplines, the ability to override the default discipline is provided. The discipline of analog primitives will be resolved based instance specific attributes, the disciplines of other instances on the same net, or default to electrical if it cannot be determined.

The precedence for the discipline of analog primitives is as follows

1. A port_discipline attribute on the analog primitive
2. The resolution of the discipline
3. The default analog primitive of electrical

E.3.3.1 Setting the discipline of analog primitives

A new optional attribute shall be provided called "port_discipline" which shall have as a value the desired discipline for the analog primitive. It shall only apply to the instance it is attached. The value shall be of type string and the value must be a valid discipline of domain continuous. This

attribute shall only apply to analog primitives and for all other modules shall be ignored. The following provides an example of this attribute.

```
resistor #(.r(1k)) (* integer port_discipline="electrical" ; *) r1 (node1, node2); // not needed as default
resistor #(.r(1k)) (* integer port_discipline="rotational" ; *) r2 (node1, node2);
```

Attributes are defined in the IEEE 1364-2001 LRM and will not be described in this document.

E.3.3.2 Resolving the disciplines of analog primitives

If not attribute exists on the instance of a analog primitive then the discipline may be determined by the disciplines of other instances connected to the same net segment. The disciplines of the vpiLoConn of all other instances on the net segment shall be evaluated to determine if they are of domain continuous and compatible with each other. If they are then the discipline of the analog primitive shall be set to the same discipline. If they are not compatible then an error will occur as define is section 3.8 of this LRM. If there are no defined continuous disciplines defined on the net segment then the discipline shall default to electrical.