====== pad lib ======

1. LV FLOW lib

{{http://vmcc.vicp.net:9090/tessent_v2023.1_doc/htmldocs/mgchelp.htm#context=ETAssembleReference&id=300&tag=idffd9a95b-1603-4bc3-8b72-c573dc94fddd|ETAssemble Tool Reference.Pad Library.Function}}

^ Function Type ^ Description ^ | forceDisable | Disables the output driver when it is set to 1. Usually, this pin is directly connected to the ForceDisable signal of the TAP. The forceDisable function does not support an active-low polarity or a pre-existing connection. | | fromPad, toPad | Input/Output connections from/to a pad and the core module. Typically, these connections are intercepted and the boundary scan cells are inserted. | | samplePad | Specifies that the pin is a test-only buffered copy of the fromPad output pin. While the fromPad pin always carries the functional data to the core, the samplePad pin reports the top-level pin state to the boundary-scan cells or to any other embedded test structure without disturbing the functional path timing. | | enableHigh, enableLow | Identify the active high or active low enable pins for bidirectional or output pads. | | fromSJEMux | Specifies the output of the SJEMux and provides feedback to the boundary-scan enable (EN) cell. | | toSJEMuxLow, toSJEMuxHigh | Specify the test enable signal from the boundary-scan EN cell. | | toSJIMux, fromSJIMux, toSJOMux, fromSJOMux | Specify the output/input on the pad cells that are connected to the Select JTAG Input (SJI) or output (SJO) multiplexer. | | selectJtagEnable | Specifies the connection for the SJEMux control signal, usually connected directly to the selectJtagOutput pin of the TAP, except when the pin is specified as an auxiliary output pin. | | selectJtagInput, selectJtagOutput | Specify the connection for the control signal, usually coming from the TAP controller, which controls the updating of the boundary-scan cell’s internal register. | | fromIO | Specifies the input connection from an input or inout top pin to its pad. If the pad description has this cell pin function but neither toIO nor padIO cell pin function, then the BSDL top pin direction will be forced to input even if the direction of the associated top pin is inout. | | toIO | Specifies the output connection from an output or inout top pin to its pad. If the pad description has this cell pin function but neither fromIO nor padIO cell pin function, then the BSDL top pin direction will be forced to output even if the direction of the associated top pin is inout. | | padIO | Specifies the input/output connection from a top pin to its pad. The top pin can be any direction. | | padIOInv | Specifies the input/output connection from the second top pin of a differential pin pair to its (differential) pad. | | fromIOInv | Specifies the input connection from the second top pin of a differential pin pair which direction is input or inout to its (differential) pad. | | toIOInv | Specifies the output connection from the second top pin of a differential pin pair whose direction is output or inout to its (differential) pad. | | parametric | Specifies that the cell pin is used for parametric test. | | DOT6-Related functions: | Group1: InitData, InitClk, TestData—This group is mandatory for all input and bidirectional AC pads. | | ::: | Group2: InitDataInv, TestDataInv—This group is mandatory for differential input and bidirectional AC pads. | | ::: | ACMode—This mode is optional. When this mode is present, ETAssemble adds the ACM subclass to the pad. | | ::: | For details, refer to the section Pad Library File in Support for IEEE 1149.6 Boundary Scan. |

{{:edastudy:pasted:20240409-090616.png?nolink}}

2. Tessent Cell Library

{{http://vmcc.vicp.net:9090/tessent_v2023.1_doc/htmldocs/mgchelp.htm#context=tessent_lib_user&id=81&tag=idf041d25c-1d03-4c91-ac27-40857c4b3f82|Attribute Descriptions}}

3. Tessent LV Flow Pad Library Mappings

{{http://vmcc.vicp.net:9090/tessent_v2023.1_doc/htmldocs/mgchelp.htm#context=tessent_lib_user&id=223&tag=idb753f227-607d-491d-96dc-a39cd40dd1bc|Tessent LV Flow Pad Library Mappings}}

^ Tessent LV Flow Syntax\ (pad.library) ^ New Tessent\ Syntax ^ ^ | forceDisable | pad_force_disable | | | fromPad | pad_from_pad | | | toPad | pad_to_pad | | | fromIO | pad_from_io | | | toIO | pad_to_io | | | padIO | pad_pad_io | Specifies the input/output connection from a top pin to its pad | | padIOInv | pad_pad_io_inv | | | fromIOInv | pad_from_io_inv | | | toIOInv | pad_to_io_inv | | | InitData | pad_init_data_dot6 | | | InitClk | pad_init_clock_dot6 | | | TestData | pad_test_data_dot6 | | | InitDataInv | pad_init_data_inv_dot6 | | | TestDataInv | pad_test_data_inv_dot6 | | | ACMode | pad_ac_mode_dot6 | | | enableHigh | pad_enable_high | Specifies the active high enable pins for bidirectional or output pads | | enableLow | pad_enable_low | Specifies the active high enable pins for bidirectional or output pads | | | pad_input_enable_high | | | | pad_input_enable_low | |

{{:edastudy:pasted:20240409-091848.png?nolink}}

4. define a cell model

<code> model model_name (list_of_pins) (model_source = verilog_module; cell_type = { clock_gating_and | clock_gating_or |

```
buffer | inverter | or | and | nand | nor |     xor | mux |
clock_buffer | clock_inverter | clock_and | clock_or | clock_mux |
clock_dff | clock_shaper | dff | latch | synchronizer_cell |
scan_cell | pad | prohibited };
```

simulation_function = { clock_gating_and | clock_gating_or |

```
func_only_clock_gating_and | func_only_clock_gating_or |
clock_shaper_reset_active | clock_shaper_reset_inactive | buffer |
inverter | or | and | nand | nor | xor | mux | dff | latch |
 synchronizer_cell | scan_cell | tie0 | tie1 } ;
```

pad_ac; pad_nonjtag; pad_ac_lp_time = <time_in_seconds>; pad_ac_hp_time = <time_in_seconds>; pad_ac_hp_on_chip; bcell_lib_name = <bcell_lib_name> nonscan_model = model_name | model_name (list_of_pins); nonscan_equivalents = model_name1, modelname2 ...; scan_equivalents = model_name1, modelname2 ...; scan_length = integer; gated_out_scan_cell; mode(

```
pad_ac;
pad_ac_lp_time = <time_in_seconds>;
pad_ac_hp_time = <time_in_seconds>;
pad_ac_hp_on_chip [Yes | No];
input (pin_name) (
  [pad_to_pad] [pad_pad_io][pad_pad_io_inv] [pad_enable_high]
  [pad_enable_low] [pad_to_sje_mux_low] [pad_to_sje_mux_high]
  [pad_to_sji_mux] [pad_to_sjo_mux] [pad_select_jtag_enable]
  [pad_select_jtag_in] [pad_select_jtag_out]
```

```
  [pad_two_state_output_enable_high]
  [pad_two_state_output_enable_low]
  [pad_init_data_dot6] [pad_init_data_inv_dot6]
  [pad_init_posedge_clock_dot6] [pad_init_negedge_clock_dot6]
  [pad_init_enable_high_dot6] [pad_init_enable_low_dot6]
  [pad_ac_mode_dot6]
  [pad_data_inv]       [pad_diff_voltage] [pad_diff_current]
  [pad_from_io] [pad_from_io_inv] [pad_tied0] [pad_tied1]
  [pad_open])
) // end input section of mode
```

input (pin_name2) ()

```
...
input (pin_nameN) ()
```

output(pin_name) (

```
  [pad_from_pad] [pad_pad_io] [pad_pad_io_inv]
  [pad_from_sje_mux] [pad_from_sji_mux] [pad_from_sjo_mux]
  [pad_sample_pad]       [pad_data_inv] [pad_diff_voltage]
  [pad_diff_current] pad_open_drain] [pad_open_source]
  [pad_to_io] [pad_to_io_inv] [pad_pull0] [pad_pull1];
) // end output section of mode
output (pin_name2) ()
...
output (pin_nameN) ()
```

inout(pin_name) (

```
  [pad_pad_io] [pad_pad_io_inv][pad_diff_voltage]
  [pad_diff_current] [pad_open_drain]
  [pad_open_source][pad_pull0] [pad_pull1]
) // end inoutsection of mode
```

inout (pin_name2) ()

```
...
inout (pin_nameN) ()
```

)// end mode

// Completed model level statements, including pad modes if any. // Define model interface port directions, and attributes.

intern (intern_nodes) (intern_attributes) input (pin_names) (list_of_input_attributes) output (pin_names) (list_of_output_attributes) inout (pin_names) (list_of_inout_attributes)

// Define hardware simulation function using cell library // primitives and instances (

```
//hardware_statements, see Hardware Definitions
```

)

) // end model_name </code>

5. input pad

<code> model ipad (PAD, C) (
model_source = verilog_module; cell_type = pad;

input (PAD) (pad_from_io;) output (C) (pad_from_pad;) (
instance = buf02 ipad_inst (PAD, C);)
) // end model ipad </code>

带EN控制 <code> model ipad_ien (PAD, C, IEN) (
model_source = verilog_module; cell_type = pad;

input (PAD) (pad_from_io;) input (IEN) (pad_input_enable_high;) output (C) (pad_from_pad;) (
// Input a 0 (low voltage) to core when input disabled. primitive = _and mlc_and (PAD, IEN, C);)
) // end model ipad_ien </code>

差分PAD <code> model Diff_Pad (C,PADP,PADN) (cell_type = pad;

input (PADP) (pad_from_io; pad_diff_voltage) input (PADN) (pad_from_io_inv;) output (C) (
pad_from_pad)

(

primitive = _buf P1 (PADP, diff_net); primitive = _inv P2 (PADN, diff_net); primitive = _buf P3 (diff_net, C);

)) </code>

6. OD OS psd

属性pad_open_drain与pad_open_source生成出来的rtl电路区别: \ 仅在输出Z态的控制逻辑不一样：

- OD输出Z态时需要把数据输入force为1
- OS输出Z态时需要把数据输入force为0

{{:edastudy:bscan:pasted:20240616-082324.png?nolink}}

在数据输入pin上加入pad_data_inv与不加的区别□\ OD情况□\
{{:edastudy:bscan:pasted:20240616-083305.png?nolink}}

OS情况□\ {{:edastudy:bscan:pasted:20240616-083351.png?nolink}}

6.1 output only 开漏 pad

只能驱动输出0， 数据输入1时，输出0， 数据输入0时高阻

{{:edastudy:pasted:20240614-103957.png?nolink}}

<code> model od_pad (DO, PAD) (model_source = verilog_module; cell_type = pad; mode(

```
 pin (DO) ( pad_to_pad; pad_data_inv;)
 pin (PAD) (pad_pad_io; pad_pull1; pad_open_source;)
```

) input (DO) ( ) inout (PAD) ( ) (

```
 primitive = _pull mlc_pull_1 (1, PAD);
```

```
 primitive = _tsh mlc_tsh_1 (0, DO, PAD);
```

)) </code>

verilog mode <code verilog> module od_pad (DO, PAD); input DO; inout PAD;

pullup(PAD);

assign PAD = (DO == 1’b1) ? 1’b0 : 1’bz;

endmodule </code>

6.2 od pull1 pad

只能驱动输出0， 数据输入0时，输出0， 数据输入1时高阻

{{:edastudy:pasted:20240614-112250.png?nolink}}

<code>

model od_pad (DI, DO, PAD) (model_source = verilog_module; cell_type = pad;

mode(

```
pin (PAD) (pad_pad_io; pad_pull1; pad_open_drain;)
pin (DO) (pad_to_pad; )
pin (DI) (pad_from_pad; )
```

)

input (DO) ( ) inout (PAD) ( ) output (DI) ( ) (

```
 primitive = _buf mlc_buf_1 (PAD, DI);
 primitive = _pull mlc_pull_1 (1, PAD);
```

```
 primitive = _inv   mlc_inv_1  (D0, DOinv);
 primitive = _tsh mlc_tsh_1 (D0, DOinv, PAD);
```

)) </code>

<code verilog>

module od_pad (DI, DO, PAD); output DI; input DO; inout PAD;

pullup(PAD);

assign PAD = (DO == 1’b0) ? 1’b0 : 1’bz; assign DI = PAD;

endmodule </code>

6.3 od inv pull1 pad

只能驱动输出0， 数据输入1时， 输出0， 数据输入0时高阻

{{:edastudy:pasted:20240614-124125.png?nolink}}

<code> model od_pad_inv (DI, DO, PAD) (model_source = verilog_module; cell_type = pad;

mode(

```
pin (PAD) (pad_pad_io; pad_pull1; pad_open_source;)
pin (DO) (pad_to_pad; pad_data_inv;)
pin (DI) (pad_from_pad; )
```

)

input (DO) ( ) inout (PAD) ( ) output (DI) ( ) (

```
 primitive = _buf mlc_buf_1 (PAD, DI);
 primitive = _pull mlc_pull_1 (1, PAD);
```

```
 primitive = _inv   mlc_inv_1  (D0, DOinv);
 primitive = _tsh mlc_tsh_1 (D0inv, DO, PAD);
```

)) </code>

<code verilog> module od_pad_inv (DI, DO, PAD); output DI; input DO; inout PAD;

pullup(PAD);

assign PAD = (DO == 1’b1) ? 1’b0 : 1’bz; assign DI = PAD; endmodule </code>

6.4 od inv pad

只能驱动输出0， 数据输入1时， 输出0， 数据输入0时高阻 \ 这种仿真会出错，因为pad_open_drain输出Z时会把DO force为1，但电路输出为0，比较不过。 \ 这时候可以把生成出来的电路换成OS生成出来的电路，仿真可以PASS

<code> model od_pad_inv (DI, DO, PAD) (model_source = verilog_module; cell_type = pad; mode(

```
pin (PAD) (pad_pad_io; pad_open_drain;)
pin (DO) (pad_to_pad; pad_data_inv;)
pin (DI) (pad_from_pad; )
```

) input (DO) ( ) inout (PAD) ( ) output (DI) ( ) (

```
 primitive = _buf mlc_buf_1 (PAD, DI);
```

```
 primitive = _inv   mlc_inv_1  (D0, DOinv);
 primitive = _tsh mlc_tsh_1 (D0inv, DO, PAD);
```

)) </code>

<code verilog> module od_pad_inv (DI, DO, PAD); output DI; input DO; inout PAD;

assign PAD = (DO == 1’b1) ? 1’b0 : 1’bz; assign DI = PAD; endmodule </code>