Equivalent Placement Types

Problem

Problem to solve;

- In the Xilinx architecture there are SLICEL and SLICEMs. A SLICEL can be placed at either a SLICEM or SLICEL location.
- In VPR you also have IOs which can be placed in an IO_TOP, IO_LEFT, IO_RIGHT, IO_BOTTOM locations.

The issue is that the "tile type" is currently selected at clustering time rather than placement time.

Secondary Problem

The pb_type object is also overloaded in a number of ways.

Proposals

Replace "top level" pb_type objects with a "tile" object.

The "tile" object takes the on responsibility for describing what pb_types can be placed at a location and how they physically connect to the interconnect.

The "tile" object specifies which pb_types it can contain. Multiple "tile" objects can specify the same pb_types to contain.

The cluster still works at the level of pb_types but the placer and rr_graph generator all work at the tile level. This allows the placer to put valid pb_types in any valid tile.

The tilegrid now references tiles rather than pb_types.

Examples, CLBLM which can be either two SLICEL or one SLICEL and one SLICEM,

```
<tile name="CLBLM">
      <mode name="CLBLM">
            <capacity num="1" name="SLICEM" />
            <capacity num="1" name="SLICEL" />
      </mode>
      <mode name="CLBLL">
            <capacity num="2" name="SLICEL" />
      </mode>
</tile>
```

Example, of RAM block which can be either a two RAM18 or one RAM36

```
<tile name="RAM" width="1" height="4">
      <mode name="RAM18">
            <capacity num="2" name="RAM18" />
      </mode>
      <mode name="RAM36">
            <capacity num="1" name="RAM36" />
```

```
        </mode>
    </tile>
```

Example, two IO tiles which can both contain the IO pb_type but puts the pins along different edges;

```
    <tile name="IO_TOP">
        <capacity num="2" name="IOB" />
        <pinlocations pattern="custom">
            <>
        </pinlocations>
    </tile>
    <tile name="IO_BOTTOM">
        <capacity num="2" name="IOB" />
        <pinlocations pattern="custom">
            <>
        </pinlocations>
    </tile>
```

Example two tiles which contain the same CLB pb_type but have different routing channel connectivity;

```
    <tile name="CLB_FULL">
        <capacity num="1" name="CLB" />
        <fc in_type="frac" in_val="0.8" out_type="frac" out_val="0.6" />
    </tile>
    <tile name="CLB_PARTIAL">
        <!-- CLB in a location which prevents full connectivity to the routing graph -->
        <capacity num="1" name="CLB" />
        <fc in_type="frac" in_val="0.2" out_type="frac" out_val="0.2" />
    </tile>
```

These following properties move from a "top level only" <pb_type> to the <tile> tag.

capacity – The number of instances of this block type at each grid location
Default: 1

For example:

<tile name="IO" capacity="2"/>
...
</tile>

specifies there are two instances of the block type IO at each of its grid locations.

width – The width of the block type in grid tiles
Default: 1

height – The height of the block type in grid tiles
Default: 1

area – The logic area (in MWTA) of the block type
Default: from the <area> tag

The following tags are unique to the top level <pb_type> of a complex logic block. They describe how a complex block interfaces with the inter-block world.

<fc in_type="{frac|abs}" in_val="{int|float}" out_type="{frac|abs}" out_val="{int|float}">
<pinlocations pattern="{spread|perimeter|custom}">
<switchblock_locations...