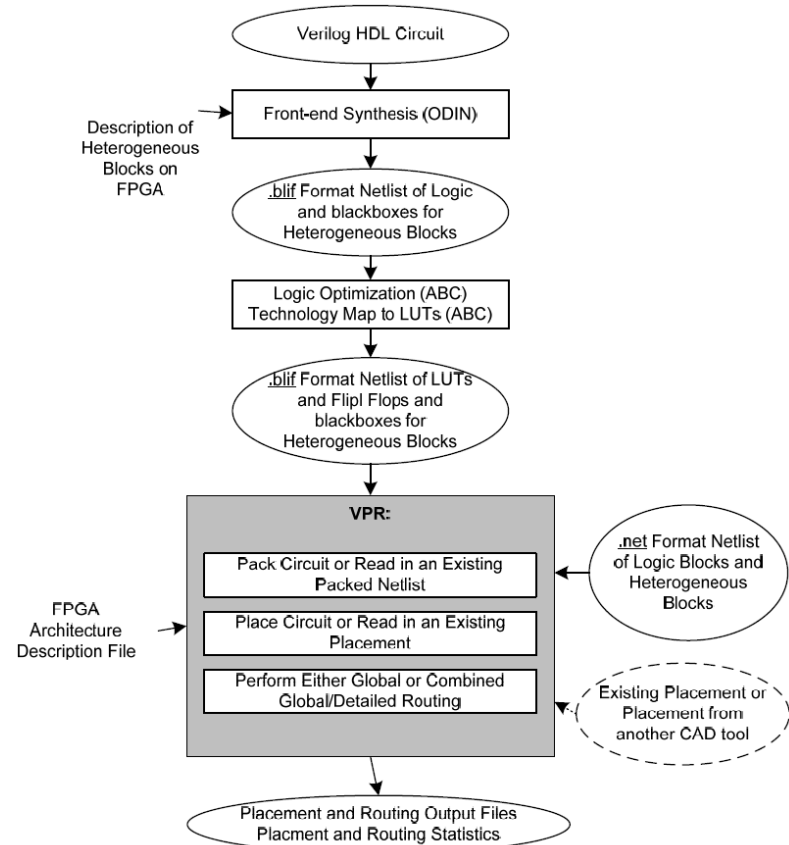


Partitioning Algorithm and Initial Results

Workflow

1. Synthesis
2. Optimisation
3. Partitioning
 1. Split
 2. Triplicate
 3. Join
 4. Flatten
4. Packing
5. Placing
6. Routing



CAD Design Flow
(VPR Manual)

Blif Format

- Text format.
- List of elements/nodes. Latches and Combinational Logic.
- Each node has text describing inputs, outputs, element.
- E.g (And Gate):
.names in1 in2 out

11 1

Algorithm

- 1. Read entire file into memory, and represent it as a graph.
- Each Model has a list of nodes and map of signalName->Signal
- Each signal points to its source and sinks
- Each node has a list (string) of inputs and outputs (+type, etc).
- Why do nodes have strings, which are looked up in a map to the signal, which points to the node? Why not have each node just point to the other node? Because of the way models are made and manipulated.

Algorithm 2

- 2. Traverse
- Start at one end, adding connected nodes to partition. Once partition reaches limit, write it out, remove those nodes from the network, and repeat.
- Specifically, start at outputs not inputs due to e.g. [IMAGE]
- 3. Adding to Partition
- Add to node collection
- Update Signals
- Recalculate critical path (max cost without cycles)

Algorithm 3

- model = network->MainModel
- FOREACH output in model
- q.Add(output->source)
- partition = EmptyModel
- WHILE node = q.pop
- IF visited(node)
- CONTINUE
- IF partition+node > limits
- WriteModelToNewFile(partition)
- partition = EmptyModel
- partition.Add(node)
- FOREACH input in node->inputs
- q.Add(input->source)
- WriteModelToNewFile(partition)
- Model::Add(node)
- nodeCollection.Add(node)
- UpdateSignals(node)
- inCost = MaxInputCost
- UpdateCosts(node, inCost)
- Model::UpdateCosts(node, inCost)
- nodeCost = inCost+InnateCost
- MarkVisited(node)
- FOREACH sink in node->sinks
- IF visited(sink)
- Cut(node, sink)
- ELSE
- UpdateCosts(sink, nodeCost)
- //Rename the signals, then after TMR we can rejoin them outside the partition
- Model::Cut(source, sink)
- source->output = "SpecialOut"+source->output
- sink->inputs[source->output] = "SpecialIn"source->output
- Model.AddInput("SpecialIn"source->output)
- Model.AddOutput("SpecialOut"+source->output)

Results

- Use very small partitions, to magnify effect.
- Routing is still the dominant contributor to time spent in workflow.
- Latency from 10%-80% increase.
- Area, number of elements, etc, all more or less triple.
- Number of nets on the critical path usually increases, sometimes decreases.