The Big Brain Theory -
Visualizing SoC Design & Verification Data

Gordon Allan, Mentor Graphics Corp.

Introduction

Collection, analysis and visualization of design and verification data is what we do, in our daily design/verification workflows, helped along by our favorite D/V methodologies and EDA tools. Successful teams get more out of the data they capture and how they use it, and avoid being overwhelmed by too much of the wrong data or the wrong kinds of processing. Recognizing that value, and amplifying it, is the next step to process maturity.

The biggest challenge we face is maximizing utilization of the best design/verification tool we possess - our engineering brains.

Their performance is limited by the rate we feed them with meaningful data - data at the right level of detail or abstraction to solve the problem at hand, and ideally data which is presented in a fluid, visual form to make the most of our parallel processing superpowers.

Advanced data visualization techniques are becoming standard practice in other industries and they can be leveraged in ours. We must pay heed to the challenges of managing 'Big Data' in our domain, so that it works for us rather than overwhelming us.

Data Management – Size, Shape, Dimensionality

Is there such a thing as Static Design Data?

Dynamic Data – the Temporal Dimension

Hierarchical Data – the Vertical Dimension

Data Across Time – the Historical Dimension

Data Visualization – Distance and Focus

Learn from the principles of centuries of scientific endeavor and the instruments used to present information in a meaningful way.

General principles to follow here are simple but often overlooked or only partially implemented as a favor to the user. These should always be implemented:

1. Where more detail is available below this data item, announce its existence and invite elaboration.
2. Where more context is available above this data item, announce its presence and invite exploration.
3. Where relationships or linkage exists between two or more data items, announce their existence and invite investigation.
4. Where membership of a class of item is denoted by an attribute, denote that clearly and invite filtering, searching, and association.

Data Filtering – Distractions and Indicators

There are bad distractions (requiring filtering) and good distractions (requiring smooth context switch and motion when navigating the data set). Some general principles in this area are worthy of mention and discussion:

1. When switching context vertically, horizontally, or via linkage, preserve sufficient context of origin to reinforce the relationship.
2. When elaborating detail on an item, retain sufficient context to reinforce the membership that detail has of the original item.
3. When ascending data hierarchy, retain sufficient depth of open detail to link the two levels of data with a degree of persistence.

Examples of Design data visualization of different kinds of data structures and recorded series:

The D3.js Data Visualization Library

Combines Web HTML, CSS, SVG (Scalable Vector Graphics) and JavaScript enabling compelling interactive visualization solutions.

```javascript
var data = [4, 8, 15, 16, 23, 42];
var width = 420;
barHeight = 20;
data(datum) {
  var x = d3.scale.linear()
    .domain([0, d3.max(data)])
    .range([0, width]);
  var chart = d3.selectAll("chart")
    .attr("height", barHeight * data.length);
  var bar = chart.selectAll("y")
    .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ");" });

  bar.append("rect")
    .attr("height", barHeight)
    .attr("width", width)
    .attr("height", barHeight * data[i])
    .attr("x", function(d) { return x(d) - barHeight / 2; })
    .attr("y", function(d) { return i * barHeight; })
    .attr("fill", "#75b1d4");

  bar.append("text")
    .attr("dy", ".35em")
    .attr("y", barHeight / 2)
    .attr("x", function(d) { return x(d) - barHeight / 2; })
    .text(function(d) { return d; });
};
```

Complex visualizations can be built from these idioms, the result is a very high performance, scalable, flexible implementation.

Data Filtering

Rich data sets can be combined and overlaid in a configurable manner including features for interaction, animation, hovering and adjustment by mouse or touch interfaces.

Examples of SoC data visualization using the D3.js library:

Some examples of SoC design visualization:

- Rich data sets can be combined and overlaid in a configurable manner including features for interaction, animation, hovering and adjustment by mouse or touch interfaces.

Contact Information

Mentor Graphics Corporation
46871 Bayside Parkway, Fremont, CA 94538

Gordon Allan
Product Manager & Architect,
Visualizer Debug

gordon_allan@mentor.com