Draft 3 - page 8 - Section 3.6 - Enumerations

Proposal: Expand the first paragraph as follows:

An enumerated type has one of a set of named values. In the following example, "light1" and "light2" are defined to be variables of the anonymous (unnamed) enumerated type that includes the three members: "red", "yellow" and "green."

The second example was confusing until I saw the comment "// silver=4, gold=5"

What happens in the following example?:

enum (a=3, b=7, c) alphabet; // what is the value of c? Does c=8?

enum (a=0, b=7, c, d=8) alphabet; // now what is the value of c? Is this a syntax error?

Proposal: for the third example, change the comment to:

```
// silver=4'h4, gold=4'h5 (all are 4 bits wide)
enum {bronze=4'h3, silver, gold} medal4;
```

Proposal: permit enum to have a range to set the common size.

Proposed wording: Add the following before the paragraph starting with "The type name can be given ...," just before the typedef example.

Adding a constant range to the enum declaration can be used to set the size of the type. If any of the enum members are defined with a different sized constant, this shall be a syntax error.

```
// Error in the bronze and gold member declarations
enum [3:0] {bronze=5'h13, silver, gold=3'h5} medal4;

// Correct declaration - bronze and gold sizes are redundant
enum [3:0] {bronze=4'h13, silver, gold=4'h5} medal4;
```

Should enums also be allowed to be signed?

Note: the beauty of the above enum capabilities is that the enum values can be initially or always omitted, and only added if and when more designer control is required. I like it!

Regards - Cliff Cummings