Add the text in 12.3.3 Constant function calls

SystemVerilog adds the following additional restrictions on constant functions:

- A constant function shall not have output, inout, or ref arguments.
- A void function shall not be a constant function.
- A constant function shall not contain a statement that directly schedules an event to execute after the function has returned.
- An import `"DPI"` function (see 12.5) shall not be a constant function.
- A constant function may have default argument values (see 12.4.3), but any such default argument value shall be a constant expression.
- A constant function shall not have any `fork/join`, `fork/join_any`, `fork/join_none` statements, or task enables.

Add a new section 12.3.4 Background processes spawned by function calls

.

12.3.4 Background processes spawned by function calls

Functions shall execute with no delay. Thus, a process calling a function shall return immediately. SystemVerilog allows any statement that does not block inside of a function. Specifically, non-blocking assignments, procedural continuous assignments, event triggers, clocking drives, and fork/join blocks are allowed inside a function.

A function that schedules an event to mature after that function returns shall be illegal in any context in which a side effect is disallowed or in any context other than procedural code originating in an initial or always block.

A task enable or any blocking statement shall be illegal inside a function unless that statement is within a `fork/join_none` block (See 11.6.1).

Note to editor: 11.6.1 was added by mantis 1615