Rápida Introdução a Projeto em VHDL:

Para o projeto de um circuito digital, o projetista deve descrever seu sistema em uma determinada linguagem, que pode ser gráfica, como um esquemático, ou textual, como em Spice, por exemplo, e em determinado nível de abstração, como interconexão de transistores individuais ou de portas lógicas. O processo de projeto emprega então ferramentas de software com uma série de etapas e algoritmos para completar ações de síntese e análise. Nas ações de síntese, informações são criadas ou detalhadas no projeto, e nas ações de análise, como em simulação lógica, por exemplo, o comportamento e a correção do projeto são verificados.

VHDL é uma linguagem de descrição de hardware criada para documentação, simulação e síntese de sistemas, abrangendo diversos níveis de abstração. A vantagem de usar VHDL (ou outras linguagens de descrição de hardware) para o projeto está justamente nessa maior abstração, deixando que a ferramenta faça as tarefas mais repetitivas, como a minimização lógica, por exemplo. Alguns recursos de VHDL usados para descrever e simular sistemas, como atrasos reais arbitrários, não podem ser sintetizados. A síntese de um circuito a partir de VHDL utiliza portanto um subconjunto da linguagem. Além disso, cada ferramenta de software tem determinadas capacidades, e pode tratar um subconjunto menor ou maior de construções. Mais importante que isso, entretanto, é que a descrição de hardware em um nível mais alto de abstração deverá usar determinados estilos, ou padrões, para que a ferramenta compreenda que tipo de hardware o projetista está descrevendo ou pode ser gerado para fazer o que foi descrito. Isso faz com que a sua descrição precise ser feita seguindo uma série de fórmulas ou receitas que descrevem bem determinados componentes.

Em VHDL a interface de cada componente é definida por uma "entidade", com portas de entrada, saída ou bidirecionais, que são seus pinos de entrada ou saída. A

implementação de um componente é definida por uma "arquitetura". Além dos sinais de entrada e saída, outros sinais internos podem ser declarados em cada arquitetura. O sinal representa basicamente um fio. Dentro de um arquitetura, uma operação típica é a atribuição de um sinal como função de outros sinais, como por exemplo: "signal_f <= a or b;", e todas as construções escritas são executadas em paralelo, pois é como o hardware se comporta. Mais especificamente, a semântica diz que os sinais que aparecem à direita de "<=" formam uma lista de sensibilidade desse assinalamento, e cada vez que há uma alteração em um desses sinais, também chamada de transição ou evento, esse assinalamento é reavaliado. Portanto, lembre-se de que a ordem na qual aparecem os assinalamentos na arquitetura é irrelevante, e você pode comparar isso com a posição na qual coloca os componentes em uma folha de esquemático: ela é irrelevante para o comportamento do circuito.

Vários circuitos podem ser feitos com variações dessa operação de assinalamento de sinais. Entretanto, para funcionalidades mais complexas, existe uma construção bastante especial chamada de "processo". O processo é conhecido como construção sequencial, mas tem duas faces um pouco distintas. Do ponto de vista externo, da arquitetura, ele serve apenas como uma forma mais genérica de especificar os valores finais para assinalamento de alguns sinais, mas dentro do qual se podem usar construções conhecidas de linguagens de programação, como if, while, chamadas de função ou variáveis intermediárias. Assim, embora esse processo seja parecido com um programa sequencial, que precisa de vários passos para sua execução, e seja avaliado sequencialmente, para o resto da arquitetura ele é semanticamente interpretado apenas como mais um assinalamento de sinais, sendo executado quase instantaneamente, ou como se fosse apenas uma porta lógica. Isso tem implicações muito importantes. Se você escrever uma construção do tipo for, sendo ela semanticamente executada toda em um único momento, ela

estará representando um for espacial, não temporal. Em outras palavras, uma construção for de, por exemplo, 0 a 31, irá gerar hardware paralelo para fazer 32 vezes as operações dentro do for ao mesmo tempo, e não uma após a outra, embora respeitando a ordem de dependência de dados, como por exemplo os bits de um ripple adder. Por essa razão um for não é uma construção muito frequente em VHDL, e quando aparecer, não estará representando um algoritmo.

A principal consequência da semântica de processos descrita acima é que você não deve usar um processo como se fosse um programa sequencial, mas em lugar disso deverá se acostumar com as fórmulas ou receitas já citadas, nas quais o processo vai especificar condições de assinalamento de sinais. Em resumo, um processo será sensível a uma lista de sinais, declarados na sua lista explícita de sensibilidade. Cada vez que houver um evento em um desses sinais, todo o processo é avaliado sequencialmente, mas no menor tempo possível, para determinar quais são os assinalamentos que serão feitos em outros sinais da arquitetura. Na prática, processos serão usados para representar registradores que dependem de sinais de relógio, e operadores, como um somador, uma ula ou um multiplexador, os quais você pode então descrever com construções de controle de fluxo.

VHDL também permite que sejam declaradas variáveis, assinaladas com ":=", e é comum usarmos variáveis dentro dos processos. Além disso, a linguagem tem tipos de dados e operações primitivas, alguns apropriados a sinais, como 'bit', outros mais planejados para variáveis, como 'integer', mas o mais frequente é usarmos o tipo de dado std_logic definido pelo padrão 1164, que fornece um sistema para simulação incluindo valores indefinido, alta impedância, fraco, etc. Com essas poucas informações vocês podem observar que existem muitas combinações possíveis entre sinal ou variável, tipo numérico como vetor de bits ou inteiro, assinalamento entre variáveis e sinais, sua declaração dentro e fora de processos, e esse é um dos muitos

pontos da linguagem onde precisamos escolher um padrão ou fórmula que funciona e é entendido pela ferramenta de síntese que usamos (no caso ALtera Max II). A percepção ou a escolha do que usar ficarão mais claras com o passar do tempo.