

ステートマシンエディタ

Active-HDL アプリケーションノート

ステートマシンエディタ概要

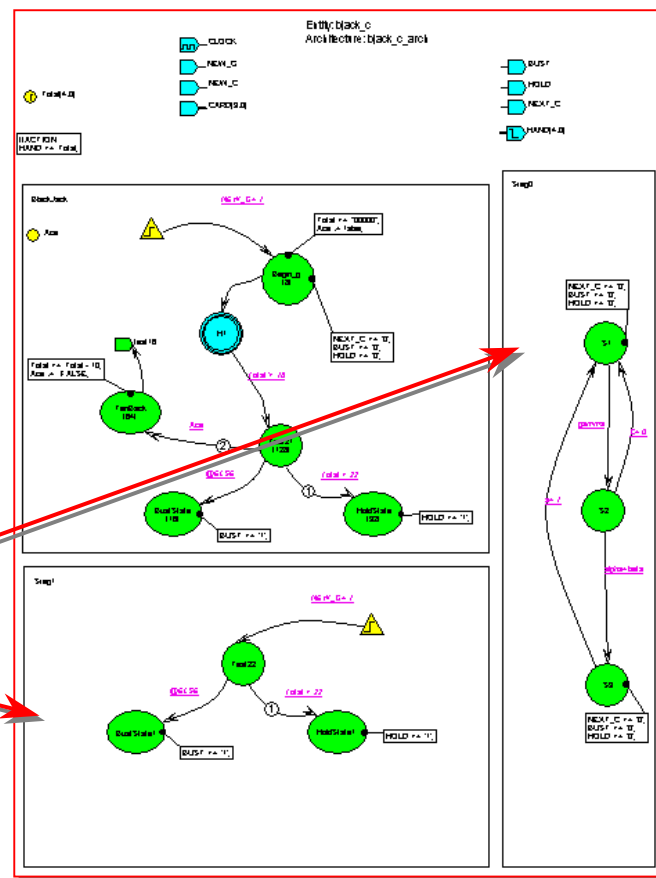
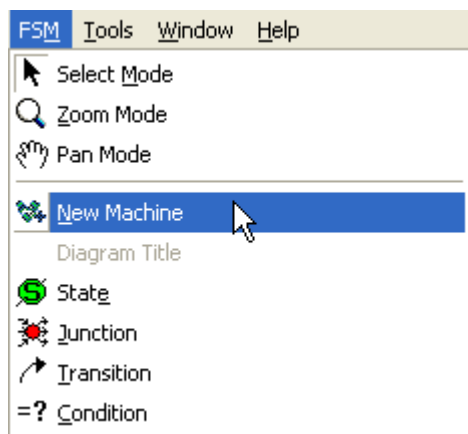
- Active-HDL のステートマシンエディタには下記機能があります。
 - 複数のステートマシン作成
 - マルチ・アーキテクチャ・サポート
 - 生成コード設定
 - ステートマシン用テストベンチ生成
 - ステートアクション編集ウィンドウ
 - 非同期マシン
 - マルチ・リセット・サポート
 - 遷移条件プライオリティ
 - ジャンクション・サポート
 - ステートマシンの階層化
 - ステートレジスタの観測
 - シンセシス・アトリビュート
 - レポートファイル生成

複数のステートマシン作成

- 同一画面内で複数のコンカレントなステートダイアグラムを使用したデザインユニットの記述が可能です。

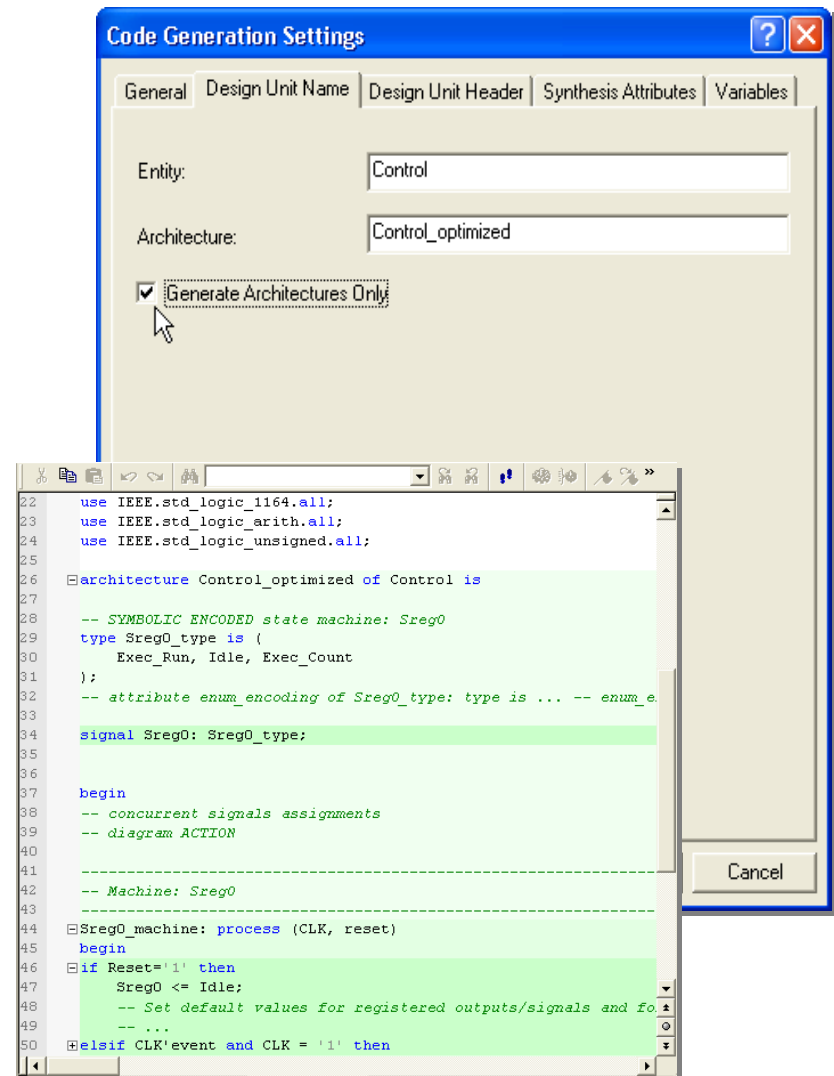
- 複数のステートダイアグラムの配置
 - メニュー | FSM | New Machine を選択

注: 複数のステートダイアグラムを配置するにはあらかじめエディタ画面に余白を設けておく必要があります。



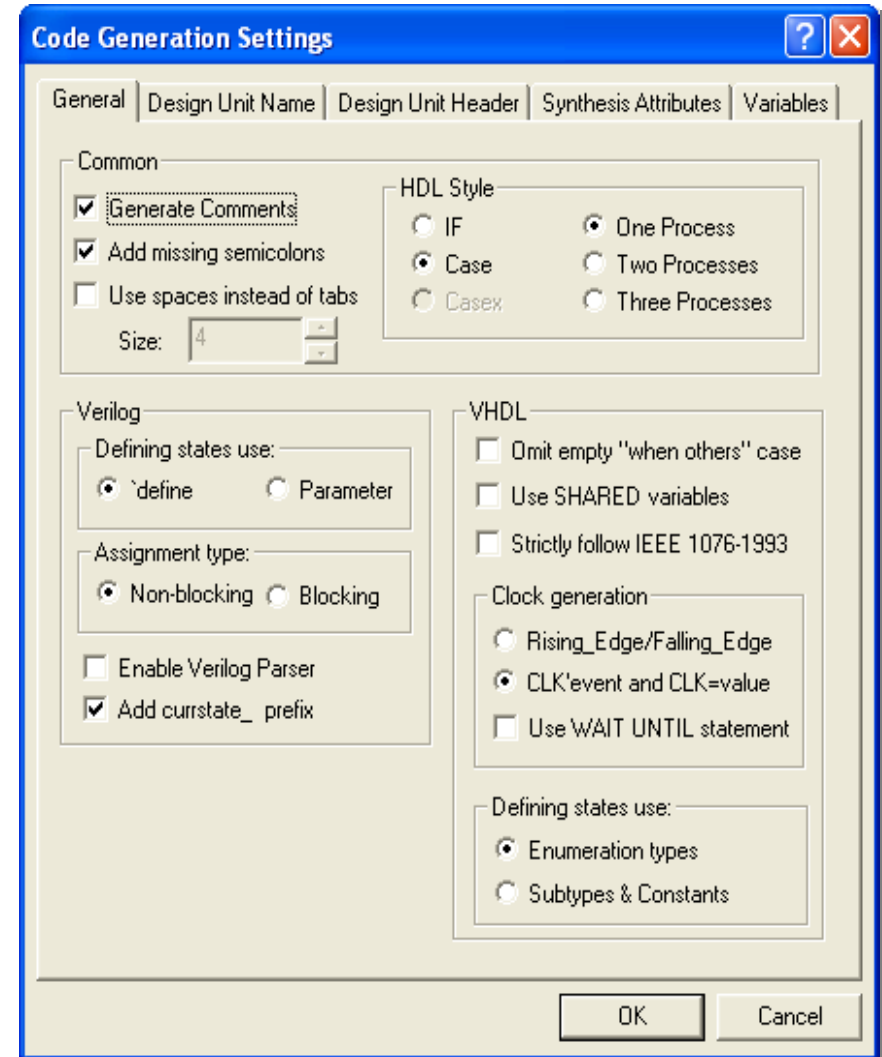
マルチ・アーキテクチャ・サポート

- VHDL の場合、アーキテクチャ部のみを生成することができます。
- 同一のエンティティに複数のアーキテクチャを配置することができます。
- アーキテクチャ部のみの生成方法
 - メニュー | FSM | Code Generation Settings を選択
 - Design Unit Name タブを選択
 - Generate Architectures Only にチェック



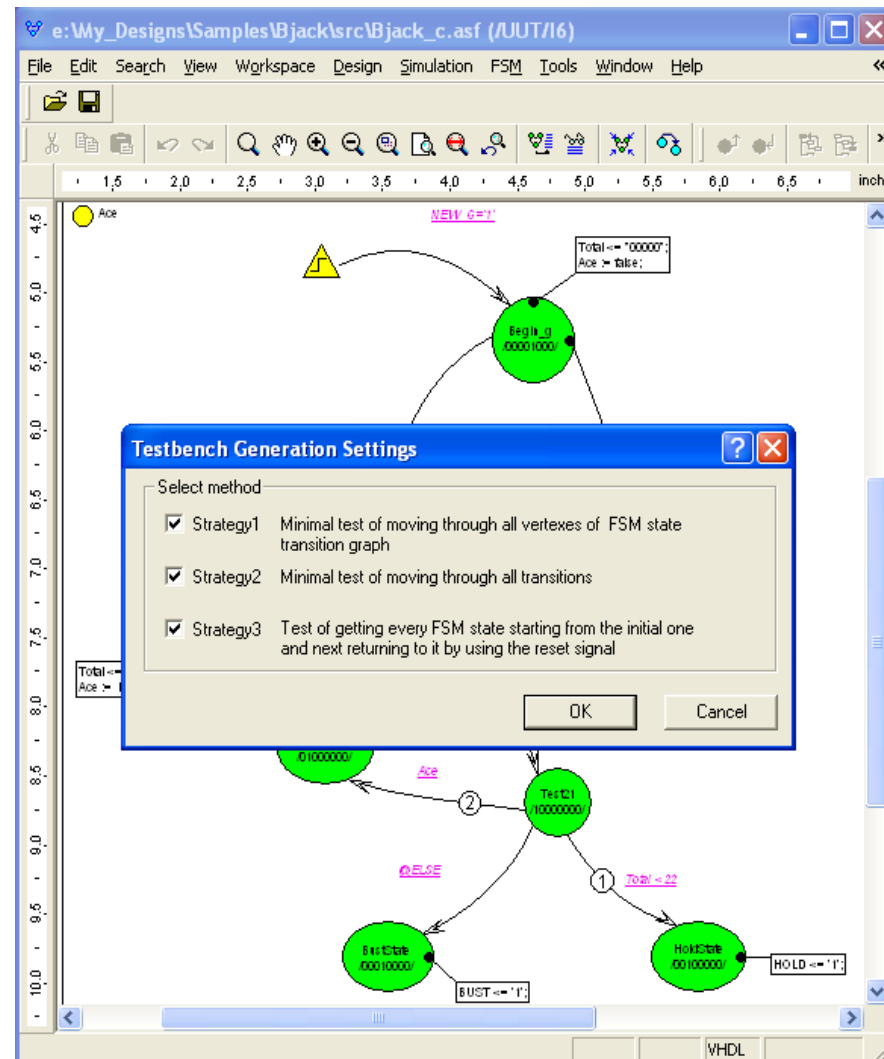
生成コード設定

- Code Generation Settings ウィンドウで生成する HDL コードの設定ができます。
- General タブ
 - HDL Style
 - If 文か case 文の選択
 - プロセスの選択
 - Verilog
 - アサインメントの選択
 - VHDL
 - クロック生成の選択



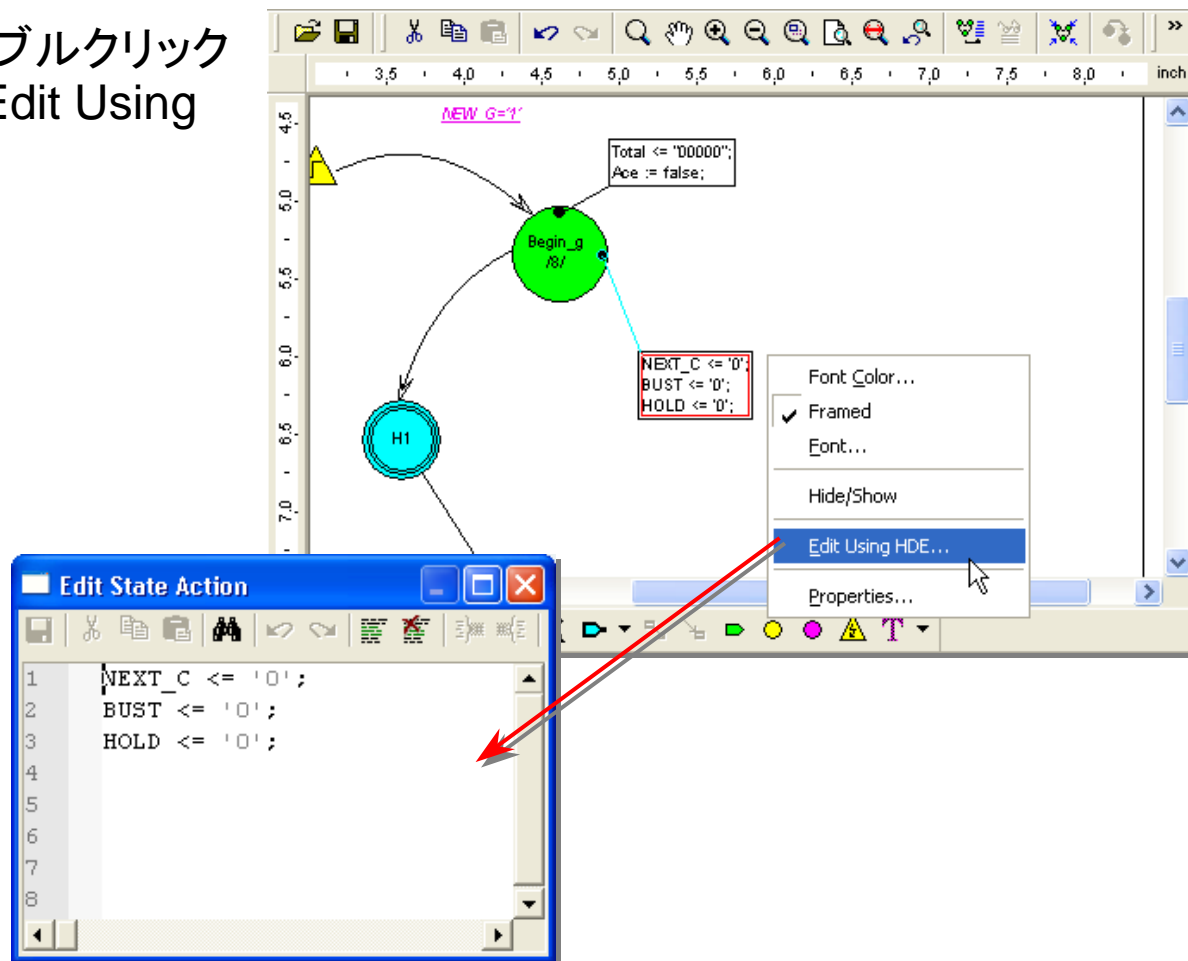
ステートマシン用テストベンチ生成

- 作成したステートマシン用のテストベンチを自動生成することができます。
- Testbench Generation Settings ウィンドウで3種類のテストベンチを選択/生成可能
 - メニュー | FSM | Testbench Generation Settings を選択
- テストベンチの生成
- メニュー | FSM | Generate Testbench を選択
 - デザインブラウザに control_TB というフォルダが生成されます。



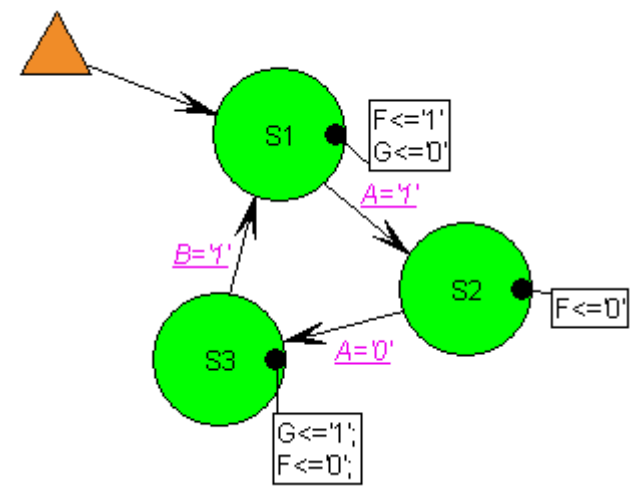
ステートアクション編集ウィンドウ

- ステートアクション記述用エディタをサポート
 - ステートアクションをダブルクリックまたは右クリックから Edit Using HDE を選択



非同期マシン

- 非同期ステートマシンの作成が可能です。
- その場合、Machine Properties ウィンドウ | General タブで適切な設定を行って下さい。
 - ステートマシンの空白部分を右クリックし、Properties を選択



Machine Properties

Name: Sreg0

Clock: [Dropdown]

Clock Enable: [Dropdown]

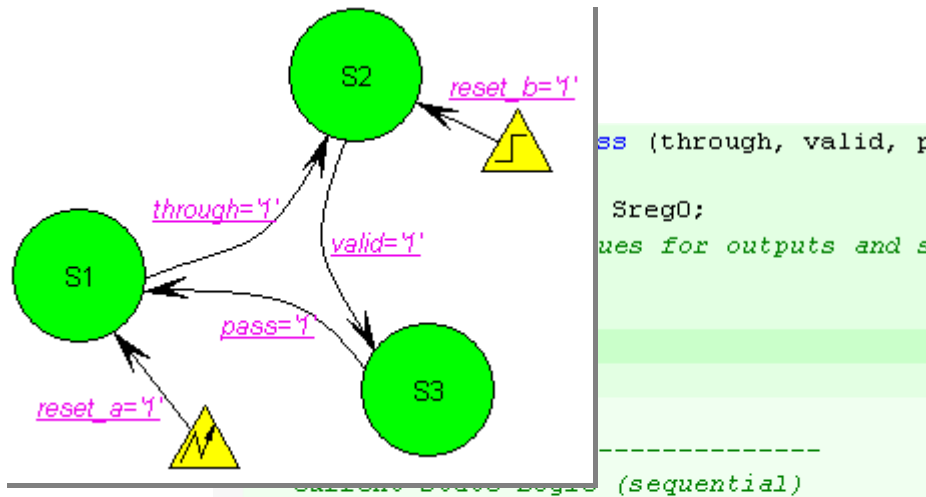
Encoding: Symbolic, Encoded: Binary/sequential

Machine: Synchronous, Asynchronous, Propagation Delay: 10 ns

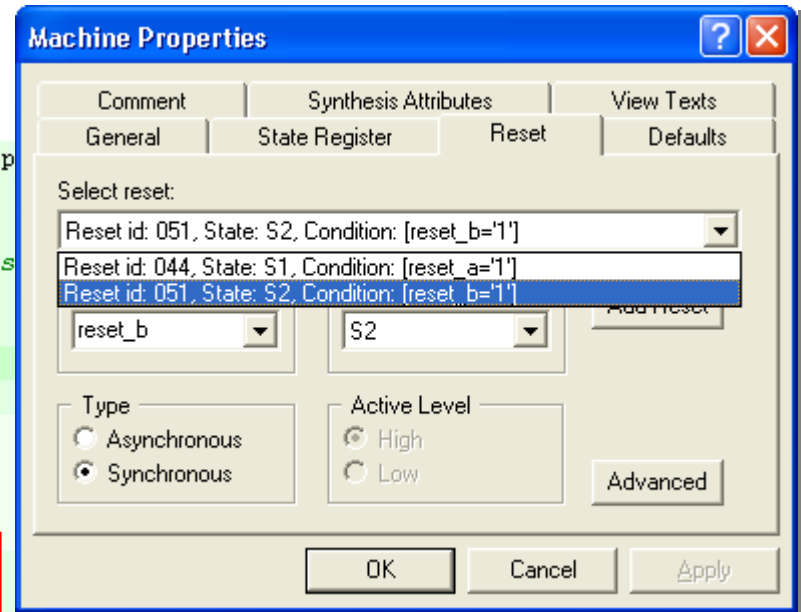
```
end
-- Current State Logic
Sreg0_CurrentState: process (NextState_Sreg0)
begin
    Sreg0 <= NextState_Sreg0 after 10 ns;
end process;
end Diagram1_arch;
```


マルチ・リセット・サポート

- 複数のリセットの設定ができます。
 - Machine Properties ウィンドウ | Reset タブ で設定

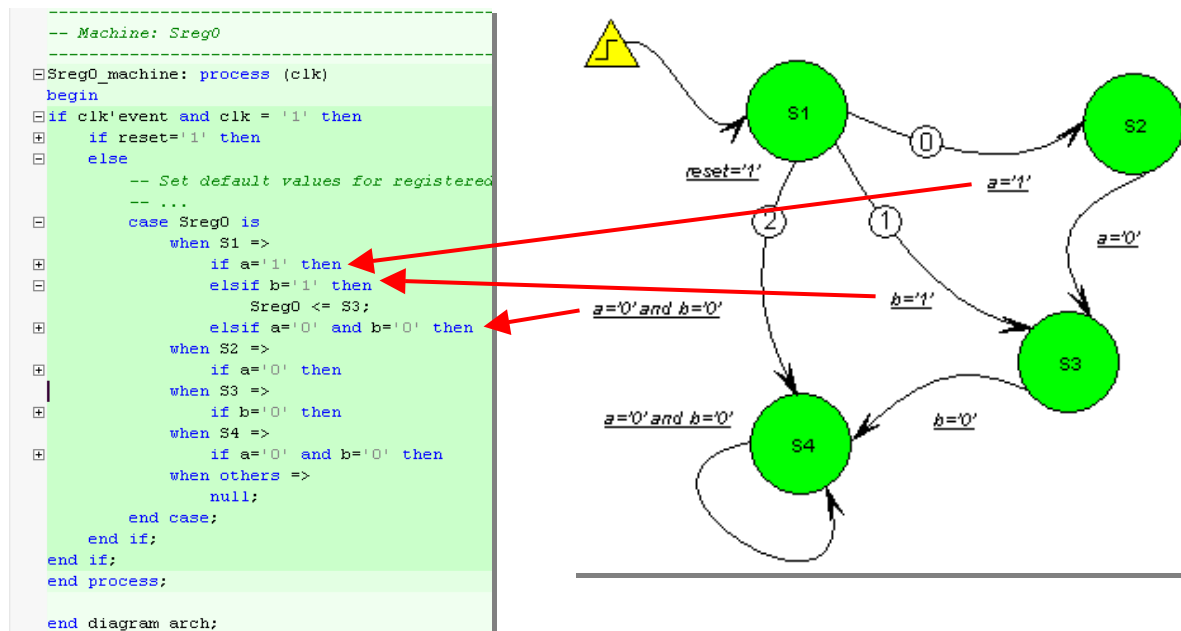


```
process (clk, reset_a)
begin
  if reset_a='1' then
    Sreg0 <= S1;
  elsif clk'event and clk = '1' then
    if reset_b='1' then
    else
    end if;
  end process;
```




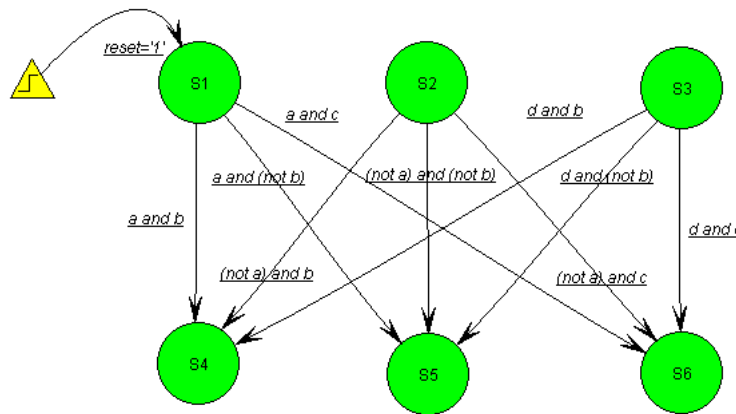
遷移条件プライオリティ

- Transition Auto Priority オプションを有効にすることで、ある状態から複数の遷移条件がある場合にそれらのプライオリティを自動的に割り当てることができます。2つ以上の条件が同時に発生する場合に、ステートマシンの行動のあいまいさを避けることができます。
- メニュー | Tools | Preferences を選択し、Preferences ウィンドウを起動
- Category でState Diagram Editor を選択し、右画面の Transition Auto Priority にチェック



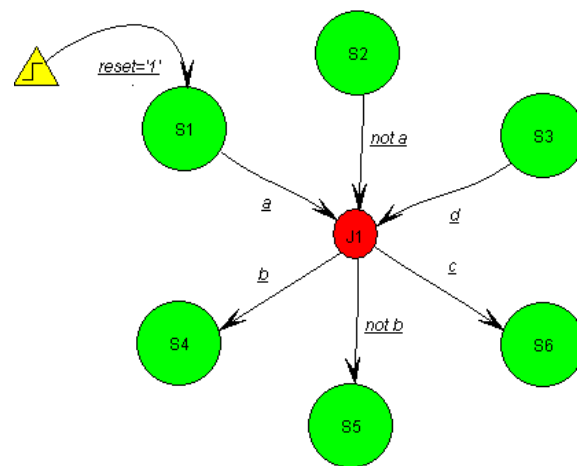
ジャンクション・サポート

- ジャンクションはステートマシン作成をシンプルに行う為のグラフィカルオブジェクトです。
- ジャンクションを使用することで右図上のステートマシンを下図のようにシンプルに作成することができます。
-  ジャンクションボタン




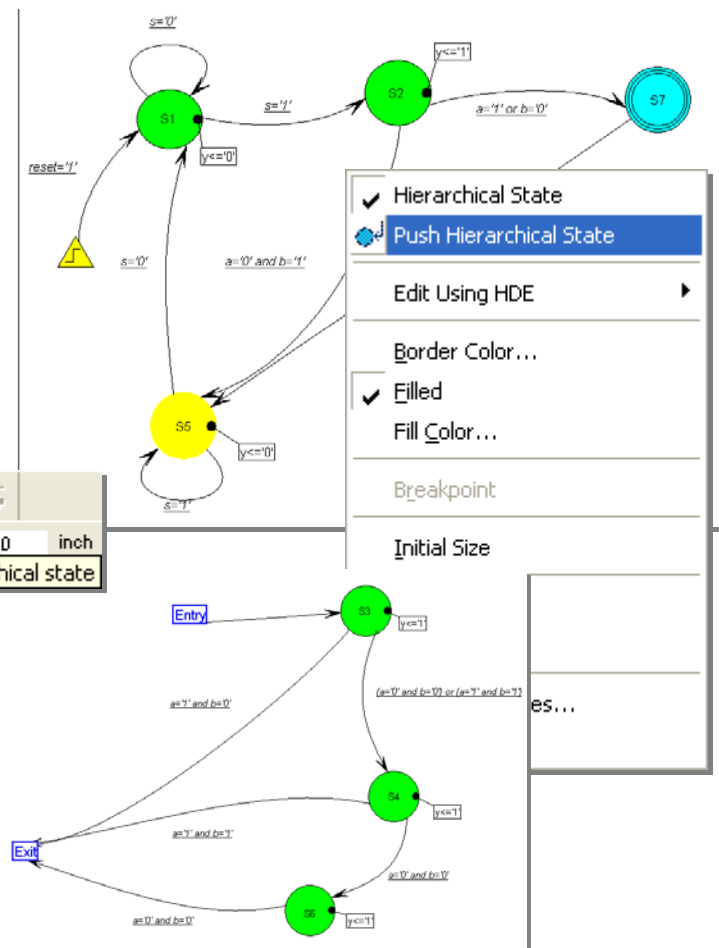
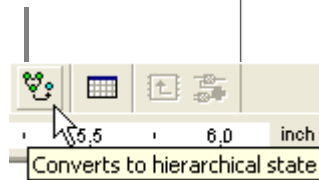
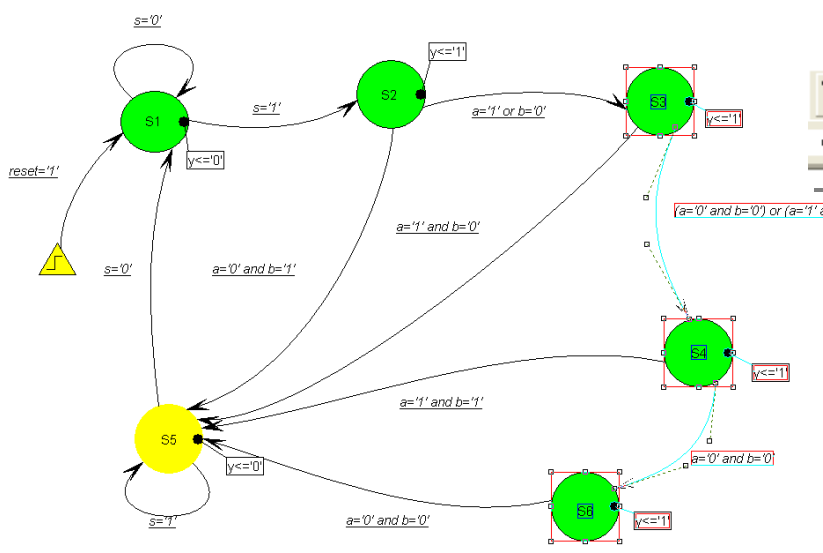
```
case Sreg0 is
  when S1 =>
    if a and (not b) then
    elsif a and c then
    elsif a and b then
  when S2 =>
    if (not a) and (not b) then
    elsif (not a) and c then
    elsif (not a) and b then
  when S3 =>
    if d and (not b) then
    elsif d and c then
    elsif d and b then
  when others =>
    null;
end case;
```

```
case Sreg0 is
  when S1 =>
    if (a) and (not b) then
    elsif (a) and (c) then
    elsif (a) and (b) then
  when S2 =>
    if (not a) and (not b) then
    elsif (not a) and (c) then
    elsif (not a) and (b) then
  when S3 =>
    if (d) and (not b) then
    elsif (d) and (c) then
    elsif (d) and (b) then
  when others =>
    null;
end case;
```



ステートマシンの階層化

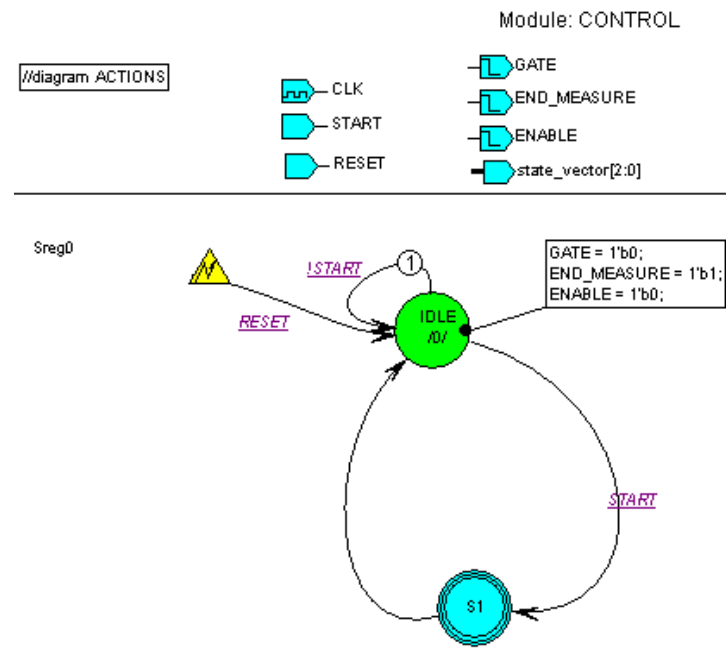
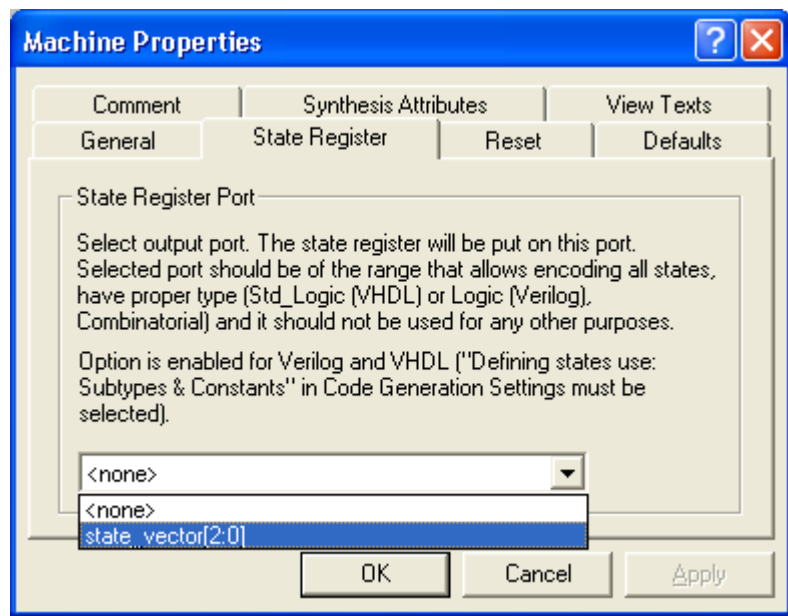
- 任意のステートを選択し、階層化することができます。
- 階層化したいステートを選択(下図では3つのステートを選択)
-  ボタンを押し、階層化。青色のステートで表示
- 下位階層の確認。青色のステートを右クリックし Push Hierarchical State を選択
- 上位階層に戻るには下位階層の空白部分を右クリックし、Pop Hierarchy を選択



ステータレジスタの観測

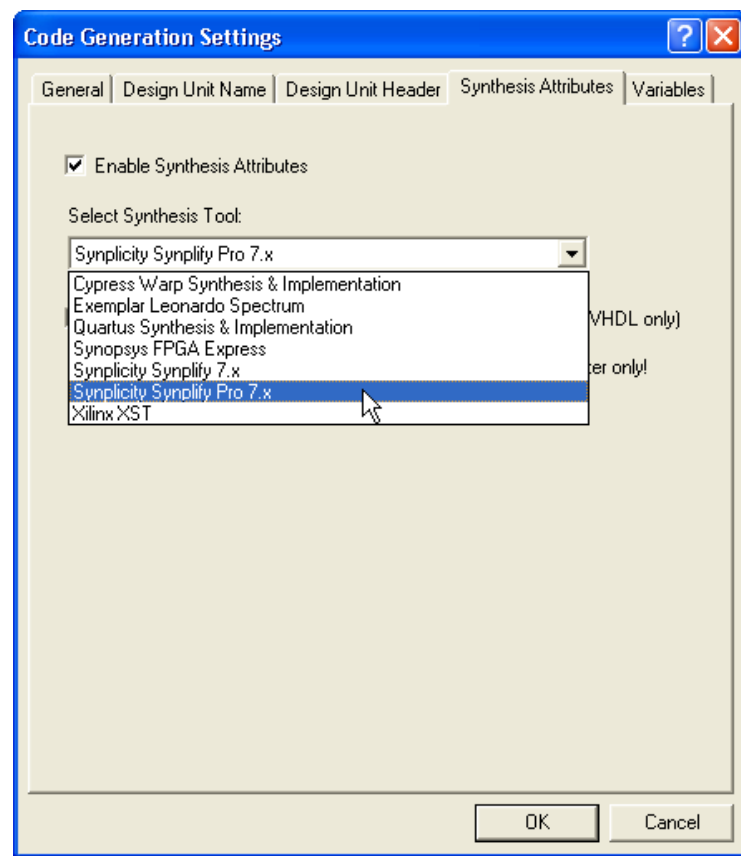
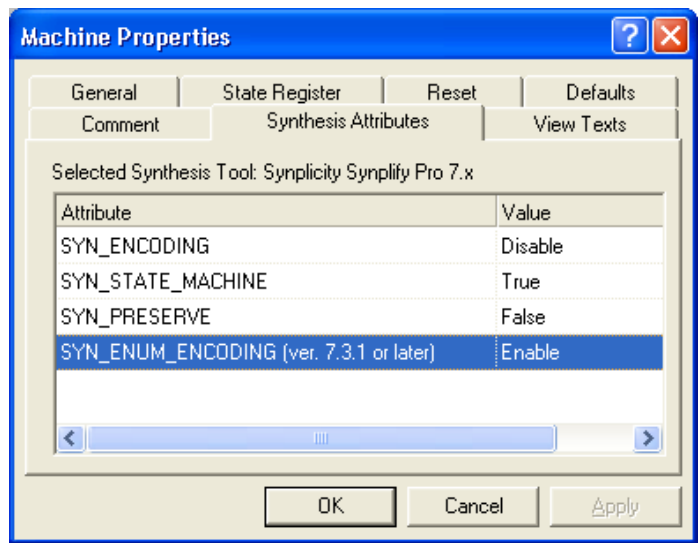
- ステータレジスタの内容をコンビネーショナル・アウトプット・ポートへ受け渡すことができます。
 - Machine Properties ウィンドウ | State Register タブでポートを選択

注: ステータレジスタの設定を行うにはCode Generation Settings ウィンドウで次の指定をしておく必要があります。Defining states use 欄 Verilog: 'define VHDL: Subtypes & Constants



シンセシス・アトリビュート

- HDL コード生成の際にシンセシス・アトリビュートを追加することができます。
- Code Generation Settings ウィンドウ | Synthesis Attributes タブで使用する論理合成ツールを選択
- Machine Properties ウィンドウ | Synthesis Attributes タブで適切な設定を行って下さい。



レポートファイル生成

- ステートマシンエディタには作成したステートマシンに関するレポートファイル生成機能があります。
- ヘッダー、ポート、階層情報等が生成されます。
- 生成方法
 - メニュー | FSM | ASF Repot を選択
 - コンソールに出力されたメッセージをダブルクリックして表示

ASF Report File - Bjack_c.asf

Created on Tue 2005-07-26 at 15:56:58

Source file: E:\My_Designs\Samples\Bjack\src\Bjack_c.asf

Target HDL: VHDL

Entity : bjack_c
Architecture: bjack_c_arch

Design Unit Header:

```
library IEEE;  
use IEEE.std_logic_1164.all;  
use IEEE.std_logic_arith.all;  
use IEEE.std_logic_unsigned.all;
```

Ports :

```
BUST (Output port, Combinatorial, Type: Std_Logic)  
CARD[3:0] (Input port, Type: Std_Logic)  
CLOCK (Input port, Clock signal, Type: Std_Logic)  
HAND[4:0] (Output port, Registered, Type: Std_Logic)  
HOLD (Output port, Combinatorial, Type: Std_Logic)  
NEW_C (Input port, Type: Std_Logic)  
NEW_G (Input port, Type: Std_Logic)  
NEXT_C (Output port, Combinatorial, Type: Std_Logic)
```

Signals:

```
Total[4:0] (Registered, Type: Std_Logic)
```

```
State Mac Console  
Machine t # Generating testbench using strategy 3...  
Clock Sig # Test quality for strategy 3 :  
Encoding: # States : 100.00 %  
Unsatisfi # Transitions : 38.46 %  
Illegal s # Test length : 11  
          # Time : 0.1 sec.  
Variables # FSM testbench generation completed. 0 error(s), 0 warning(s)  
Ace (Type # FSM: Creating ASF report for file: C:\My_Designs\Data\lab 4\lab 4\src\control.asf  
          # FSM: The results have been written to the log file. Double click on this line to view the log.  
Hierarchy Console  
State Machine: BlackJack  
|_ H1  
design flow bjack_c.asf asf report fil...
```