ISE

HDL Options

The following properties apply to the Synthesize process using the Xilinx® Synthesis Technology (XST) synthesis tool.

- **FSM Encoding Algorithm**

  Sets the fsm_encoding constraint which determines the finite state machine coding technique to be used.

  Select an option from the drop-down list.
  - **Auto**

    Selects the needed optimization algorithms during the synthesis process.
  - **One-Hot**

    Ensures that an individual state register is dedicated to one state. Only one flip-flop is active, or hot, at any one time. One-hot encoding is very appropriate with most FPGA targets where a large number of flip-flops are available. It is also a good alternative when trying to optimize speed or to reduce power dissipation.
  - **Compact**

    Minimizes the number of state variables and flip-flops. This technique is based on hypercube immersion. Compact encoding is appropriate when trying to optimize area.
  - **Sequential**

    Consists of identifying long paths and applying successive radix two codes to the states on these paths. Next state equations are minimized.
  - **Gray**

    Guarantees that only one state variable switches between two consecutive states. It is appropriate for controllers exhibiting long paths without branching. In addition, this coding technique minimizes hazards and glitches. Very good results can be obtained when implementing the state register with T or JK flip-flops.
  - **Johnson**

    Much like the Gray option, shows benefits with state machines containing long paths with no branching.
  - **User**

    The synthesis tool uses the encoding defined in the source file.
  - **Speed1**

    Speed1 encoding is oriented for speed optimization. The number of bits for state register depends on each FSM, but in general case it is greater than the number of FSM states.
  - **None**

    Disables automatic FSM extraction.

  By default, this property is set to Auto.
- **Safe Implementation**

  Specifies whether or not a safe implementation of a finite state machine description is used. When set to Yes, any illegal state will send the state machine back to a recovery state to return to normal operation. When set to No, the logic required for recovery from an illegal state is not implemented.

  By default, this property is set to No. This option is available when FSM Encoding Algorithm is set to anything other than None.

  For more information, refer to the safe_recovery_state constraint in the XST User Guide.
- **Case Implementation Style**

  Specifies how case statements are used during synthesis.

  You can select one of four options in the drop-down list box.
  - **None**

    No case statement directives are used during synthesis.
  - **Full**

    A Verilog metacomment used to indicate that all possible selector values have been expressed in a case, casex or casez statement. Values that are not expressed are simply considered as never being reached during normal circuit operation, and the directive prevents XST from creating additional hardware for those conditions.
  - **Parallel**

A Verilog metacomment used to force a case statement to be synthesized as a parallel multiplexer and prevents the case statement from being transformed into a prioritized if/elseif cascade.
    - **Full-Parallel**

      Both the parallel_case and full-case directives are applied to a case selector.

  For FPGAs, this property is set to None by default.

  For CPLDs, this property is set to Full-Parallel by default.
- **FSM Style (Advanced) (FPGA only)**

  Specifies whether to map the FSM LUTs or Block RAM. By default, this property is set to LUT.
- **RAM Extraction (FPGA only)**

  Specifies whether or not to use a RAM macro inference. By default, this property is set to True (checkbox is checked), and RAM inference is enabled.
- **RAM Style (FPGA only)**

  Specifies the way in which the macrogenerator implements the RAM macros.

  Note This property is available only when the RAM Extraction property is set to True (checkbox is checked).

  You can select one of the following three options:
    - **Auto**

      XST determines the best implementation for each macro.
    - **Distributed**

      Implements RAM as Distributed RAM.
    - **Block**

      Implements RAM as Block RAM.

  By default, this property is set to Auto.
- **ROM Extraction (FPGA only)**

  Specifies whether or not to use a ROM macro inference. By default, this property is set to True (checkbox is checked), and ROM inference is enabled.
- **ROM Style (FPGA only)**

  Specifies the way in which the macrogenerator implements the ROM macros.

  Note This property is available only when the ROM Extraction property is set to True (checkbox is checked).

  You can select one of the following three options:
    - **Auto**

      XST determines the best implementation for each macro.
    - **Distributed**

      Implements ROM as Distributed ROM.
    - **Block**

      Implements ROM as Block ROM.

  If the ROM Extraction property is set to False (checkbox is blank), then the ROM Style property is disabled and is not written to the command line.
- **Automatic BRAM Packing (Advanced) (FPGA only)**

  Specifies whether or not XST will try to pack two small single-port BRAMs into a single BRAM primitive, as dual-port BRAM. BRAMs to be packed together only if they are at the same hierarchical level in the design. By default, this property is set to No (checkbox is blank).
- **Mux Extraction**

  Specifies whether or not to use a multiplexer macro inference. You can select between Yes, No and Force. Force preserves all multiplexers during optimization process and does not optimize them with the rest of the design logic. By default, this property is set to Yes.
- **Mux Style (FPGA only)**

  Specifies the way in which the macrogenerator implements the multiplexer macros.

  Note This property is available when the Mux Extraction property is set to Yes or Force.

  You can select one of the following three options:
    - **Auto**

      XST determines the best implementation for each considered macro.
    - **MUXF**

      Is based on Virtex® and Spartan® series MuxF5/F6/F7/F8 resources.
    - **MUXCY**

      Is based on Virtex and Spartan series MuxCY resources.

By default, this property is set to Auto.

- **Decoder Extraction (FPGA only)**

  Specifies whether or not to use a decoder macro inference. By default, this property is set to True (checkbox is checked).
- **Priority Encoder Extraction (FPGA only)**

  Specifies whether or not to use a priority encoder macro inference. You can select between Yes, No and Force. Force forces XST to extract the macro or does not optimize it with the rest of the design logic. By default, this property is set to Yes.
- **Shift Register Extraction (FPGA only)**

  Specifies whether or not to use a shift register macro inference. By default, this property is set to True (checkbox is checked), and a shift register macro is inferred.
- **Logical Shifter Extraction (FPGA only)**

  Specifies whether or not to infer a logical shifter macro. By default, this property is set to True (checkbox is checked).
- **XOR Collapsing (FPGA only)**

  Specifies whether or not cascaded XORs are collapsed into a single XOR. By default, this property is set to True (checkbox is checked), and cascaded XORs are collapsed.
- **Resource Sharing**

  Specifies whether or not to share arithmetic operator resources. By default, this property is set to True (checkbox is checked).
- **Multiplier Style**

  This property is available for Virtex®-II, Virtex-II Pro, Virtex-II Pro X, Spartan®-3, Spartan-3E and Spartan-3A only.

  Specifies how the macrogenerator implements the multiplier macros.

  Select an option from the drop-down list.
  - **Auto**

    XST looks for the best implementation for each considered macro.
  - **Block**

    This implementation style uses block multiplier resources available in the Virtex-II and Virtex-II Pro devices.
  - **LUT**

    This implementation style uses LUT resources available in the Virtex devices.
  - **Pipe_LUT**

    A combination of LUTs and registers are used for this implementation style. It is valid only if the multiplication function is registered one or more times allowing XST to evenly distribute the registers throughout the multiplication function.

  By default, the property is set to AUTO.
- **Use DSP48 (Virtex-4 only)**

  Specifies whether or not DSP48 blocks are utilized for Virtex-4 designs.

  Select an option from the drop-down list.
  - **Auto**

    XST examines the benefits of placing these macros in DSP48 blocks, and then determines the most efficient implementation.
  - **Yes**

    XST places all macros in the DSP48 blocks whenever possible. This option enables you to see how many DSP48 blocks are used for a compiled submodule.
  - **No**

    XST uses standard FPGA resources for these macros.

  By default, this property is set to Auto.
- **Use DSP Block (Virtex-5 and Spartan-3A D only)**

  Specifies whether or not DSP blocks are utilized for Virtex-5 designs.

  Select an option from the drop-down list.
  - **Auto**

    XST examines the benefits of placing these macros in DSP blocks, and then determines the most efficient implementation.
  - **Yes**

    XST places all macros in the DSP blocks whenever possible. This option enables you to see how many DSP blocks are used for a compiled submodule.
  - **No**

    XST uses standard FPGA resources for these macros.

  By default, this property is set to Auto.

- **Asynchronous to Synchronous (Advanced) (FPGA only)**

  Specifies whether or not asynchronous Set/Reset signals will be replaced by synchronous signals throughout the design. If selected, registers can be absorbed by DSP blocks or BRAMs. Since XST will be allowed to merge more registers into dedicated resources, your quality of results will improve. This feature may also have a positive iMPACT on power optimization.

  Caution Replacing Asynchronous Set/Reset signals by Synchronous signals makes the generated NGC netlist NOT equivalent to the initial RTL description. You must ensure that the synthesized design satisfies the initial specification. XST will inform you with this message:

  ```
  WARNING: You have requested that asynchronous control signals of sequential
  elements be treated as if they were synchronous. If you haven’t done so yet,
  please carefully review the related documentation material. If you have opted
  to asynchronously control flip-flop initialization, this feature allows you to
  better explore the possibilities offered by the Xilinx solution without having
  to go through a painful rewriting effort. However, be well aware that the
  synthesis result, while providing you with a good way to assess final device
  usage and design performance, is not functionally equivalent to your HDL
  description. As a result, you will not be able to validate your design by
  comparison of pre-synthesis and post-synthesis simulation results. Please also
  note that in general we strongly recommend synchronous flip-flop
  initialization.
  ```

  By default, this property is set to No (checkbox is blank).