XILINX®

XAPP992 (v8.0) September 21, 2010

FIFO Generator Migration Guide

Summary

The FIFO Generator Migration Guide provides step-by-step instructions for migrating existing designs containing instances of legacy FIFO cores (Synchronous FIFO v5.x and Asynchronous FIFO v6.x) to the latest version of the FIFO Generator.

Note: For all new designs, Xilinx recommends that you use the most recent version of the FIFO Generator core for your FIFO function requirements.

Migration Overview

The FIFO Generator Migration Kit, uses a perl script to automate the FIFO core migration process. The Migration Kit can be obtained at:

https://secure.xilinx.com/webreg/clickthrough.do?cid=151908&license=RefDesLicense

Differences from Legacy Cores

This section defines the feature differences between the older asynchronous and synchronous FIFO cores, and the latest version of the FIFO Generator core. Before migrating your existing designs, evaluate the differences, because they may affect the behavior of your current design. In some cases, you may need to modify your design to compensate for obsolete features.

The FIFO Generator core can generate synchronous and asynchronous FIFOs, and can also leverage the built-in FIFOs in the Virtex®-6, Virtex-5 and Virtex-4 FPGA device families. In addition, the FIFO Generator core benefits from the use of the Block Memory Generator, which optimizes block memory resource utilization and enhances the overall performance of the FIFO Generator.

Core Compatibility

The FIFO Generator core is not backward-compatible with the Synchronous FIFO and Asynchronous FIFO cores in the following ways:

- FIFO Generator uses different port names. For example, the names of the reset and handshaking flags are different.
- The XCO files for the previous generation memory cores are NOT compatible with the FIFO Generator.
- Behavioral differences between Synchronous FIFO cores generated by FIFO Generator and those generated by Synchronous FIFO include the following:
  - The maximum data count width for FIFO Generator is one bit less than that for Synchronous FIFO and the behavior of DATA_COUNT is different. For example, for the Synchronous FIFO, when the FIFO depth is 256, the maximum data count width is 9 bits. However, for FIFO Generator, the maximum data count width is only 8 bits.
    When the FIFO is full:
    DATA_COUNT = "1 0000 0000" in Synchronous FIFO
    DATA_COUNT = "0000 0000" in FIFO Generator
  - In this case, when using the maximum data count width, you can connect the full flag as the most significant bit of DATA_COUNT to make the FIFO Generator backward-compatible.

- ♦ If both data count widths are equal, the behavior of DATA COUNT is different. When the FIFO is full, DATA_COUNT is "11111111" in Synchronous FIFO and DATA_COUNT is "00000000" in FIFO Generator. In this case, the FIFO Generator is not backward compatible.
- Behavioral differences between Asynchronous FIFO cores generated by FIFO Generator and those generated by Asynchronous FIFO include the following:
  - ♦ The falling edge of the EMPTY and ALMOST_EMPTY flags may occur one clock cycle later in the FIFO Generator in response to a write operation.
  - ♦ The falling edge of the FULL and ALMOST_FULL flags may occur one clock cycle later in the FIFO Generator in response to a read operation.
  - ♦ WR_COUNT takes longer to respond to a read, and RD_COUNT takes longer to respond to a write in FIFO Generator.
  - ♦ Reset requirements differ between the two cores.
  - ♦ The default FULL reset value differs between the two cores—set the FULL reset value to ’1’ for backward-compatibility.

Differences in port names and XCO parameters for legacy cores (Synchronous and Asynchronous FIFOs) are defined in “Converting the XCO File,” page 4 and “Modifying the Instantiations of the Old Core,” page 18.

Obsolete Features for Legacy Cores

Create RPM

The FIFO Generator core does not support Relationally Placed Macros (RPMs), which were supported in Asynchronous FIFO core.

Migrating a Design

The Migration Kit provides a Perl script to help automate the process of converting existing Synchronous or Asynchronous FIFO cores to the latest FIFO Generator core version.

The migration script automates all the manual steps, from converting the XCO file to modifying the instantiation of the old core. In addition, this script can be used to only automate specific steps, making it also useful when following the “Manual Migration Process,” page 4.

Migration Script

Note: ISE® software v12.2 and later require a CGP file when CORE Generator is run in command line mode. This version of the Migration Script comes with a sample CGP file (`coregen.cgp`) which the user can modify according to their requirements. The modified CGP file should be kept in the user directory where the XCO file and instantiation files are located and must be named `coregen.cgp` to work with the migration script.

If you can provide a complete set of original design files (XCO files and instantiation template files), the migration script (`fifo_migrate.pl`) completely and seamlessly automates the migration process by executing the following steps:

1. Converts old XCO files to new format XCO files.
2. Converts instantiations of old cores to new core instantiations including changing port names.
3. Generates new netlist(s) by calling the CORE Generator™ software with the new XCO file.

About the Migration Script

The migration script, `fifo_migrate.pl`, can operate on various inputs and create a variety of outputs based on user-specified command line options.

When using the script as part of the standard, fully-automated flow, you supply the script with either of these two file types or both:

- Old XCO core configuration files (created by the GUI when the FIFO core was generated)
- HDL source file(s) containing the core instantiations (VHDL or Verilog)

From the script options, choose one or more of the following migration steps. All selected steps are automatically performed by the script.

- Old FIFO XCO files to FIFO Generator v7.2 XCO files (use -x option).
- Generate the new netlists and convert the instantiations of the Old FIFO cores in your HDL source code to latest FIFO Generator core instances by running the CORE Generator software (-x and -m options).

The script modifies and overwrites all input files so that the external project files and scripts do not need to be updated with new file names or locations. Although the script also automatically generates a backup of all files it modifies, it is strongly recommended that you create a backup of all project files before running the migration script.

Output Products

Depending on the chosen command line option, the script overwrites the input XCO files, modifies the input HDL files, and optionally generates FIFO Generator netlists (in the same location as the XCO files).

The script creates a `./fifo_migrate_bak_filename(xco/v/vhd)` backup directory in which a copy of all files modified by the migration process are placed. It also generates a restore script in this directory, `restore_files.pl`, to allow you to restore the original files if necessary.

Using the Migration Script

To start the Migration Script, type the commands specific to your environment at the command prompt.

Linux

```
<path to script>/fifo_migrate.pl -x -m <HDL file(s)> <xco file>
```

Windows

```
xilperl <path to script>\fifo_migrate.pl -x -m <HDL file(s)> <xco file>
```

You must use at least one of the following options in the command string:

- **-x**. Creates XCO output files needed by the CORE Generator software to generate the core. Requires an input of the older version XCO file.
- **-m**. Calls the CORE Generator software to generate FIFO Generator netlists necessary to synthesize the design. This option must be used in tandem with the -x option. It modifies the HDL source files containing the core instantiations, converting the older instantiations and adding compatibility code. Requires either XCO or HDL file or both.

While using -x along with -m, the user can input only one XCO file, but one or more HDL files(s) that correspond(s) to the input XCO file. The modification of the HDL file will be according to the input XCO file. In addition, the user has to input the respective XCO file for modification of HDL files containing legacy version of instantiation(s).

`<xco file(s)>` is a list of one or more core configuration files corresponding to old FIFO cores which are to be converted to latest FIFO Generator core. These files can be referenced from directories other than the working directory.

To reverse the changes made by the script, go to the backup directory at `./fifo_migrate_bak_filename(xco/v/vhd)` and then run perl script `restore_files.pl`.

Usage Examples

```
fifo_migrate.pl -x -m my_design.v my_core.xco
```

1. Creates a FIFO Generator version of `my_core.xco`.
2. Modifies the instantiations of my_core in `my_design.v`.
3. Runs CORE Generator software to generate the FIFO Generator version of `my_core.ngc` netlist file.

```
fifo_migrate.pl -x my_mem_core.xco
```

1. Creates a FIFO Generator version of `my_mem_core.xco`.
2. Script prompts the user to input a valid FIFO Generator version on execution.

Manual Migration Process

This section provides the instructions for the manual migration of an existing design to a FIFO Generator core. A summary of the required steps are provided below, followed by specific step-by-step instructions.

1. Convert the XCO file.

   The XCO file is used by the CORE Generator to determine a core's configuration. The format for the FIFO Generator core differs from the Synchronous and Asynchronous FIFO cores.

   Note: If you plan to generate a new FIFO Generator core via the CORE Generator GUI, skip this step.
2. Generate the FIFO Generator core.
3. Modify the instantiations of the old core. As the final step in the migration, you must update all instantiations of the old cores in your HDL source code to reference the new core. This includes changing the port names, as explained in "Modifying the Instantiations of the Old Core," page 18. "Differences from Legacy Cores," page 1 discusses whether design modifications are needed to compensate for obsolete features.

Converting the XCO File

Table 1 and Table 2 define the mapping between the XCO file parameters for the Synchronous and Asynchronous FIFO cores and the XCO for the latest FIFO Generator core. You may need to change your design to compensate for obsolete features. See "Core Compatibility," page 1 for information about difference between the old and new cores.

In addition to changes in the parameter section of the XCO file, you must change the core name specified in the XCO. Update the core name and version from the old core to the new core.

For example, for the legacy Synchronous FIFO core, change the line:

```
SELECT Synchronous_FIFO family Xilinx,_Inc. 5.0
```

to:

```
SELECT FIFO_Generator family Xilinx,_Inc. 7.2
```

For the legacy Asynchronous FIFO core, change the line:

```
SELECT Asynchronous_FIFO family Xilinx,_Inc. 6.1
```

to:

```
SELECT FIFO_Generator family Xilinx,_Inc. 7.2
```

Table 1: **XCO Parameter Mapping: Synchronous FIFO Cores**

Synchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Synchronous FIFO)
component_name	component_name	No change required

memory_type	fifo_implementation	Common_Clock_Block_RAM = block_memory
		Common_Clock_Distributed_RAM = distributed_memory

data_width	input_data_width
	output_data_width
		input_data_width = data_width
		output_data_width = data_width

fifo_depth	input_depth
	output_depth
		input_depth = fifo_depth
		output_depth = fifo_depth

write_acknowledge_flag	write_acknowledge_flag	No change required
write_acknowledge_sense	write_acknowledge_sense	No change required
write_error_flag	overflow_flag	Direct replacement
write_error_sense	overflow_sense	Direct replacement
read_acknowledge_flag	valid_flag	Direct replacement
read_acknowledge_sense	valid_sense	Direct replacement
read_error_flag	underflow_flag	Direct replacement
read_error_sense	underflow_sense	Direct replacement
data_count	data_count	No change required
data_count_width	data_count_width	No change required
-	interface_type	Add this parameter and set the value to Native
-	reset_pin	Add this parameter and set the value to True
-	reset_type	Add this parameter and set the value to Synchronous_Reset
-	use_dout_reset	True (for backward compatibility)
-	almost_empty_flag	Add this parameter and set the value to False
-	almost_full_flag	Add this parameter and set the value to False
-	disable_timing_violations	Add this parameter and set the value to False
-	dout_reset_value	Add this parameter and set the value to 0
-	empty_threshold_assert_value	Add this parameter and set the value to 2
-	empty_threshold_negate_value	Add this parameter and set the value to 3
-	enable_ecc	Add this parameter and set the value to False
-	enable_reset_synchronization	Add this parameter and set the value to False
-	full_flags_reset_value	1 (for backward compatibility)
-	full_threshold_assert_value	Add this parameter and set the value to 4
-	full_threshold_negate_value	Add this parameter and set the value to 3
-	inject_sbit_error	Add this parameter and set the value to False
-	inject_dbit_error	Add this parameter and set the value to False

Table 1: **XCO Parameter Mapping: Synchronous FIFO Cores *(Cont'd)***

Synchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Synchronous FIFO)
-	performance_options	Add this parameter and set the value to Standard_FIFO
-	programmable_empty_type	Add this parameter and set the value to No_Programmable_Empty_Threshold
-	programmable_full_type	Add this parameter and set the value to No_Programmable_Full_Threshold
-	read_clock_frequency	Add this parameter and set the value to 1
-	read_data_count	Add this parameter and set the value to False
-	read_data_count_width	Add this parameter and set the value to 1
-	use_embedded_registers	Add this parameter and set the value to False
-	use_extra_logic	Add this parameter and set the value to False
-	write_clock_frequency	Add this parameter and set the value to 1
-	write_data_count	Add this parameter and set the value to False
-	write_data_count_width	Add this parameter and set the value to 1
-	axi_type	Add this parameter and set the value to AXI4_Stream
-	enable_write_channel	Add this parameter and set the value to False
-	enable_read_channel	Add this parameter and set the value to False
-	clock_type_axi	Add this parameter and set the value to Common_Clock
-	use_clock_enable	Add this parameter and set the value to False
-	id_width	Add this parameter and set the value to 1
-	axi_address_width	Add this parameter and set the value to 1
-	axi_data_width	Add this parameter and set the value to 8
-	enable_awuser	Add this parameter and set the value to False
-	enable_wuser	Add this parameter and set the value to False
-	enable_buser	Add this parameter and set the value to False
-	enable_aruser	Add this parameter and set the value to False
-	enable_ruser	Add this parameter and set the value to False
-	enable_tuser	Add this parameter and set the value to False
-	awuser_width	Add this parameter and set the value to 1
-	wuser_width	Add this parameter and set the value to 1
-	buser_width	Add this parameter and set the value to 1
-	aruser_width	Add this parameter and set the value to 1
-	ruser_width	Add this parameter and set the value to 1
-	tuser_width	Add this parameter and set the value to 1
-	enable_tdata	Add this parameter and set the value to False
-	enable_tdest	Add this parameter and set the value to False
-	enable_tid	Add this parameter and set the value to False

Table 1: **XCO Parameter Mapping: Synchronous FIFO Cores *(Cont'd)***

Synchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Synchronous FIFO)
-	enable_tkeep	Add this parameter and set the value to False
-	enable_tlast	Add this parameter and set the value to False
-	enable_tready	Add this parameter and set the value to False
-	enable_tstrobe	Add this parameter and set the value to False
-	tdata_width	Add this parameter and set the value to 8
-	tdest_width	Add this parameter and set the value to 1
-	tid_width	Add this parameter and set the value to 1
-	tkeep_width	Add this parameter and set the value to 1
-	tstrb_width	Add this parameter and set the value to 1
-	axis_type	Add this parameter and set the value to FIFO
-	wach_type	Add this parameter and set the value to FIFO
-	wdch_type	Add this parameter and set the value to FIFO
-	wrch_type	Add this parameter and set the value to FIFO
-	rach_type	Add this parameter and set the value to FIFO
-	rdch_type	Add this parameter and set the value to FIFO
-	fifo_implementation_type_axis	Add this parameter and set the value to Common_Clock_Block_RAM
-	fifo_implementation_type_rach	Add this parameter and set the value to Common_Clock_Distributed_RAM
-	fifo_implementation_type_rdch	Add this parameter and set the value to Common_Clock_Block_RAM
-	fifo_implementation_type_wach	Add this parameter and set the value to Common_Clock_Distributed_RAM
-	fifo_implementation_type_wdch	Add this parameter and set the value to Common_Clock_Block_RAM
-	fifo_implementation_type_wrch	Add this parameter and set the value to Common_Clock_Distributed_RAM
-	fifo_application_type_axis	Add this parameter and set the value to Data_FIFO
-	fifo_application_type_rach	Add this parameter and set the value to Data_FIFO
-	fifo_application_type_rdch	Add this parameter and set the value to Data_FIFO
-	fifo_application_type_wach	Add this parameter and set the value to Data_FIFO
-	fifo_application_type_wdch	Add this parameter and set the value to Data_FIFO
-	fifo_application_type_wrch	Add this parameter and set the value to Data_FIFO
-	enable_ecc_axis	Add this parameter and set the value to False
-	enable_ecc_rach	Add this parameter and set the value to False
-	enable_ecc_rdch	Add this parameter and set the value to False
-	enable_ecc_wach	Add this parameter and set the value to False
-	enable_ecc_wdch	Add this parameter and set the value to False
-	enable_ecc_wrch	Add this parameter and set the value to False

Table 1: **XCO Parameter Mapping: Synchronous FIFO Cores *(Cont'd)***

Synchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Synchronous FIFO)
-	inject_sbit_error_axis	Add this parameter and set the value to False
-	inject_sbit_error_rach	Add this parameter and set the value to False
-	inject_sbit_error_rdch	Add this parameter and set the value to False
-	inject_sbit_error_wach	Add this parameter and set the value to False
-	inject_sbit_error_wdch	Add this parameter and set the value to False
-	inject_sbit_error_wrch	Add this parameter and set the value to False
-	inject_dbit_error_axis	Add this parameter and set the value to False
-	inject_dbit_error_rach	Add this parameter and set the value to False
-	inject_dbit_error_rdch	Add this parameter and set the value to False
-	inject_dbit_error_wach	Add this parameter and set the value to False
-	inject_dbit_error_wdch	Add this parameter and set the value to False
-	inject_dbit_error_wrch	Add this parameter and set the value to False
-	input_depth_axis	Add this parameter and set the value to 1024
-	input_depth_rach	Add this parameter and set the value to 16
-	input_depth_rdch	Add this parameter and set the value to 1024
-	input_depth_wach	Add this parameter and set the value to 16
-	input_depth_wdch	Add this parameter and set the value to 1024
-	input_depth_wrch	Add this parameter and set the value to 16
-	enable_data_counts_axis	Add this parameter and set the value to False
-	enable_data_counts_rach	Add this parameter and set the value to False
-	enable_data_counts_rdch	Add this parameter and set the value to False
-	enable_data_counts_wach	Add this parameter and set the value to False
-	enable_data_counts_wdch	Add this parameter and set the value to False
-	enable_data_counts_wrch	Add this parameter and set the value to False
-	enable_prog_flags_axis	Add this parameter and set the value to False
-	enable_prog_flags_rach	Add this parameter and set the value to False
-	enable_prog_flags_rdch	Add this parameter and set the value to False
-	enable_prog_flags_wach	Add this parameter and set the value to False
-	enable_prog_flags_wdch	Add this parameter and set the value to False
-	enable_prog_flags_wrch	Add this parameter and set the value to False
-	programmable_full_type_axis	Add this parameter and set the value to Full
-	programmable_full_type_rach	Add this parameter and set the value to Full
-	programmable_full_type_rdch	Add this parameter and set the value to Full
-	programmable_full_type_wach	Add this parameter and set the value to Full
-	programmable_full_type_wdch	Add this parameter and set the value to Full
-	programmable_full_type_wrch	Add this parameter and set the value to Full

Table 1: **XCO Parameter Mapping: Synchronous FIFO Cores *(Cont'd)***

Synchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Synchronous FIFO)
-	full_threshold_assert_value_axis	Add this parameter and set the value to 5
-	full_threshold_assert_value_rach	Add this parameter and set the value to 5
-	full_threshold_assert_value_rdch	Add this parameter and set the value to 5
-	full_threshold_assert_value_wach	Add this parameter and set the value to 5
-	full_threshold_assert_value_wdch	Add this parameter and set the value to 5
-	full_threshold_assert_value_wrch	Add this parameter and set the value to 5
-	programmable_empty_type_axis	Add this parameter and set the value to Empty
-	programmable_empty_type_rach	Add this parameter and set the value to Empty
-	programmable_empty_type_rdch	Add this parameter and set the value to Empty
-	programmable_empty_type_wach	Add this parameter and set the value to Empty
-	programmable_empty_type_wdch	Add this parameter and set the value to Empty
-	programmable_empty_type_wrch	Add this parameter and set the value to Empty
-	empty_threshold_assert_value_axis	Add this parameter and set the value to 5
-	empty_threshold_assert_value_rach	Add this parameter and set the value to 4
-	empty_threshold_assert_value_rdch	Add this parameter and set the value to 4
-	empty_threshold_assert_value_wach	Add this parameter and set the value to 4
-	empty_threshold_assert_value_wdch	Add this parameter and set the value to 4
-	empty_threshold_assert_value_wrch	Add this parameter and set the value to 4
-	underflow_flag_axi	Add this parameter and set the value to False
-	underflow_sense_axi	Add this parameter and set the value to Active_High
-	overflow_flag_axi	Add this parameter and set the value to False
-	overflow_sense_axi	Add this parameter and set the value to Active_High
-	disable_timing_violations_axi	Add this parameter and set the value to False

Table 2: **XCO Parameter Mapping: Asynchronous FIFO Cores**

Asynchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Asynchronous FIFO)
component_name	component_name	No change required

memory_type	fifo_implementation	Independent_Clocks_Block_RAM = block
		Independent_Clocks_Distributed_RAM = distributed

data_width	input_data_width
	output_data_width
		input_data_width = input_data_width
		output_data_width = input_data_width

fifo_depth	input_depth
	output_depth
		input_depth = fifo_depth + 1
		output_depth = fifo_depth + 1

almost_empty_flag	almost_empty_flag	No change required
almost_full_flag	almost_full_flag	No change required
write_acknowledge	write_acknowledge_flag	Direct replacement

Table 2: **XCO Parameter Mapping: Asynchronous FIFO Cores *(Cont'd)***

Asynchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Asynchronous FIFO)
write_acknowledge_sense	write_acknowledge_sense	No change required
write_error	overflow_flag	Direct replacement
write_error_sense	overflow_sense	Direct replacement
read_acknowledge	valid_flag	Direct replacement
read_acknowledge_sense	valid_sense	Direct replacement
read_error	underflow_flag	Direct replacement
read_error_sense	underflow_sense	Direct replacement
write_count	write_data_count	Direct replacement
write_count_width	write_data_count_width	Direct replacement
read_count	read_data_count	Direct replacement
read_count_width	read_data_count_width	Direct replacement
create_rpm	-	Not supported
-	interface_type	Add this parameter and set the value to Native
-	reset_pin	Add this parameter and set the value to True
-	reset_type	Add this parameter and set the value to Asynchronous_Reset
-	use_dout_reset	True (for backward compatibility)
-	data_count	Add this parameter and set the value to False
-	data_count_width	Add this parameter and set the value to 1
-	almost_empty_flag	Add this parameter and set the value to False
-	almost_full_flag	Add this parameter and set the value to False
-	disable_timing_violations	Add this parameter and set the value to False
-	dout_reset_value	Add this parameter and set the value to 0
-	empty_threshold_assert_value	Add this parameter and set the value to 2
-	empty_threshold_negate_value	Add this parameter and set the value to 3
-	enable_ecc	Add this parameter and set the value to False
-	enable_reset_synchronization	Add this parameter and set the value to False
-	full_flags_reset_value	1 (for backward compatibility)
-	full_threshold_assert_value	Add this parameter and set the value to 4
-	full_threshold_negate_value	Add this parameter and set the value to 3
-	inject_sbit_error	Add this parameter and set the value to False
-	inject_dbit_error	Add this parameter and set the value to False
-	performance_options	Add this parameter and set the value to Standard_FIFO
-	programmable_empty_type	Add this parameter and set the value to No_Programmable_Empty_Threshold
-	programmable_full_type	Add this parameter and set the value to No_Programmable_Full_Threshold

Table 2: **XCO Parameter Mapping: Asynchronous FIFO Cores *(Cont'd)***

Asynchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Asynchronous FIFO)
-	read_clock_frequency	Add this parameter and set the value to 1
-	use_embedded_registers	Add this parameter and set the value to False
-	use_extra_logic	Add this parameter and set the value to False
-	write_clock_frequency	Add this parameter and set the value to 1
-	axi_type	Add this parameter and set the value to AXI4_Stream
-	enable_write_channel	Add this parameter and set the value to False
-	enable_read_channel	Add this parameter and set the value to False
-	clock_type_axi	Add this parameter and set the value to Common_Clock
-	use_clock_enable	Add this parameter and set the value to False
-	id_width	Add this parameter and set the value to 1
-	axi_address_width	Add this parameter and set the value to 1
-	axi_data_width	Add this parameter and set the value to 8
-	enable_awuser	Add this parameter and set the value to False
-	enable_wuser	Add this parameter and set the value to False
-	enable_buser	Add this parameter and set the value to False
-	enable_aruser	Add this parameter and set the value to False
-	enable_ruser	Add this parameter and set the value to False
-	enable_tuser	Add this parameter and set the value to False
-	awuser_width	Add this parameter and set the value to 1
-	wuser_width	Add this parameter and set the value to 1
-	buser_width	Add this parameter and set the value to 1
-	aruser_width	Add this parameter and set the value to 1
-	ruser_width	Add this parameter and set the value to 1
-	tuser_width	Add this parameter and set the value to 1
-	enable_tdata	Add this parameter and set the value to False
-	enable_tdest	Add this parameter and set the value to False
-	enable_tid	Add this parameter and set the value to False
-	enable_tkeep	Add this parameter and set the value to False
-	enable_tlast	Add this parameter and set the value to False
-	enable_tready	Add this parameter and set the value to False
-	enable_tstrobe	Add this parameter and set the value to False
-	tdata_width	Add this parameter and set the value to 8
-	tdest_width	Add this parameter and set the value to 1
-	tid_width	Add this parameter and set the value to 1
-	tkeep_width	Add this parameter and set the value to 1

Table 2: **XCO Parameter Mapping: Asynchronous FIFO Cores *(Cont'd)***

Asynchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Asynchronous FIFO)
-	tstrb_width	Add this parameter and set the value to 1
-	axis_type	Add this parameter and set the value to FIFO
-	wach_type	Add this parameter and set the value to FIFO
-	wdch_type	Add this parameter and set the value to FIFO
-	wrch_type	Add this parameter and set the value to FIFO
-	rach_type	Add this parameter and set the value to FIFO
-	rdch_type	Add this parameter and set the value to FIFO
-	fifo_implementation_type_axis	Add this parameter and set the value to Common_Clock_Block_RAM
-	fifo_implementation_type_rach	Add this parameter and set the value to Common_Clock_Distributed_RAM
-	fifo_implementation_type_rdch	Add this parameter and set the value to Common_Clock_Block_RAM
-	fifo_implementation_type_wach	Add this parameter and set the value to Common_Clock_Distributed_RAM
-	fifo_implementation_type_wdch	Add this parameter and set the value to Common_Clock_Block_RAM
-	fifo_implementation_type_wrch	Add this parameter and set the value to Common_Clock_Distributed_RAM
-	fifo_application_type_axis	Add this parameter and set the value to Data_FIFO
-	fifo_application_type_rach	Add this parameter and set the value to Data_FIFO
-	fifo_application_type_rdch	Add this parameter and set the value to Data_FIFO
-	fifo_application_type_wach	Add this parameter and set the value to Data_FIFO
-	fifo_application_type_wdch	Add this parameter and set the value to Data_FIFO
-	fifo_application_type_wrch	Add this parameter and set the value to Data_FIFO
-	enable_ecc_axis	Add this parameter and set the value to False
-	enable_ecc_rach	Add this parameter and set the value to False
-	enable_ecc_rdch	Add this parameter and set the value to False
-	enable_ecc_wach	Add this parameter and set the value to False
-	enable_ecc_wdch	Add this parameter and set the value to False
-	enable_ecc_wrch	Add this parameter and set the value to False
-	inject_sbit_error_axis	Add this parameter and set the value to False
-	inject_sbit_error_rach	Add this parameter and set the value to False
-	inject_sbit_error_rdch	Add this parameter and set the value to False
-	inject_sbit_error_wach	Add this parameter and set the value to False
-	inject_sbit_error_wdch	Add this parameter and set the value to False
-	inject_sbit_error_wrch	Add this parameter and set the value to False
-	inject_dbit_error_axis	Add this parameter and set the value to False
-	inject_dbit_error_rach	Add this parameter and set the value to False

Table 2: **XCO Parameter Mapping: Asynchronous FIFO Cores *(Cont'd)***

Asynchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Asynchronous FIFO)
-	inject_dbit_error_rdch	Add this parameter and set the value to False
-	inject_dbit_error_wach	Add this parameter and set the value to False
-	inject_dbit_error_wdch	Add this parameter and set the value to False
-	inject_dbit_error_wrch	Add this parameter and set the value to False
-	input_depth_axis	Add this parameter and set the value to 1024
-	input_depth_rach	Add this parameter and set the value to 16
-	input_depth_rdch	Add this parameter and set the value to 1024
-	input_depth_wach	Add this parameter and set the value to 16
-	input_depth_wdch	Add this parameter and set the value to 1024
-	input_depth_wrch	Add this parameter and set the value to 16
-	enable_data_counts_axis	Add this parameter and set the value to False
-	enable_data_counts_rach	Add this parameter and set the value to False
-	enable_data_counts_rdch	Add this parameter and set the value to False
-	enable_data_counts_wach	Add this parameter and set the value to False
-	enable_data_counts_wdch	Add this parameter and set the value to False
-	enable_data_counts_wrch	Add this parameter and set the value to False
-	enable_prog_flags_axis	Add this parameter and set the value to False
-	enable_prog_flags_rach	Add this parameter and set the value to False
-	enable_prog_flags_rdch	Add this parameter and set the value to False
-	enable_prog_flags_wach	Add this parameter and set the value to False
-	enable_prog_flags_wdch	Add this parameter and set the value to False
-	enable_prog_flags_wrch	Add this parameter and set the value to False
-	programmable_full_type_axis	Add this parameter and set the value to Full
-	programmable_full_type_rach	Add this parameter and set the value to Full
-	programmable_full_type_rdch	Add this parameter and set the value to Full
-	programmable_full_type_wach	Add this parameter and set the value to Full
-	programmable_full_type_wdch	Add this parameter and set the value to Full
-	programmable_full_type_wrch	Add this parameter and set the value to Full
-	full_threshold_assert_value_axis	Add this parameter and set the value to 5
-	full_threshold_assert_value_rach	Add this parameter and set the value to 5
-	full_threshold_assert_value_rdch	Add this parameter and set the value to 5
-	full_threshold_assert_value_wach	Add this parameter and set the value to 5
-	full_threshold_assert_value_wdch	Add this parameter and set the value to 5
-	full_threshold_assert_value_wrch	Add this parameter and set the value to 5
-	programmable_empty_type_axis	Add this parameter and set the value to Empty
-	programmable_empty_type_rach	Add this parameter and set the value to Empty

Table 2: **XCO Parameter Mapping: Asynchronous FIFO Cores *(Cont'd)***

Asynchronous FIFO XCO Parameter	FIFO Generator XCO Parameter	Description of Conversion (FIFO Generator = Asynchronous FIFO)
-	programmable_empty_type_rdch	Add this parameter and set the value to Empty
-	programmable_empty_type_wach	Add this parameter and set the value to Empty
-	programmable_empty_type_wdch	Add this parameter and set the value to Empty
-	programmable_empty_type_wrch	Add this parameter and set the value to Empty
-	empty_threshold_assert_value_axis	Add this parameter and set the value to 5
-	empty_threshold_assert_value_rach	Add this parameter and set the value to 4
-	empty_threshold_assert_value_rdch	Add this parameter and set the value to 4
-	empty_threshold_assert_value_wach	Add this parameter and set the value to 4
-	empty_threshold_assert_value_wdch	Add this parameter and set the value to 4
-	empty_threshold_assert_value_wrch	Add this parameter and set the value to 4
-	underflow_flag_axi	Add this parameter and set the value to False
-	underflow_sense_axi	Add this parameter and set the value to Active_High
-	overflow_flag_axi	Add this parameter and set the value to False
-	overflow_sense_axi	Add this parameter and set the value to Active_High
-	disable_timing_violations_axi	Add this parameter and set the value to False

Generate the FIFO Generator Core

Generate a new FIFO Generator netlist. Note that you must have ISE 12.3 installed on your system. The newly generated netlist (NGC) file replaces the old netlist file.

There are two ways to generate a new FIFO Generator netlist: using the CORE Generator GUI, or by executing an updated XCO file.

Parameterizing the FIFO Generator GUI

With an existing project open, or after creating a new project, the FIFO Generator core is available through the CORE Generator GUI.

To open the FIFO Generator core, do the following:

1. Click View by Function (active by default), and then open Memories & Storage Elements > FIFOs.
2. Double-click FIFO Generator to display the main GUI screen.

Table 3 compares the GUI parameters between the Synchronous and Asynchronous FIFO cores and the FIFO Generator cores. This table helps you choose the appropriate options when creating a new core using the FIFO Generator GUI.

Table 3: **GUI Parameter Comparison**

Synchronous/ Asynchronous FIFO	**FIFO Generator**	**Functionality of GUI Parameter (FIFO Generator= Synchronous/Asynchronous FIFO)**
Block Memory Distributed Memory	• Common Clock (CLK), Block RAM • Common Clock (CLK), Distributed RAM • Common Clock (CLK), Shift Register • Common Clock (CLK), Built-in FIFO • Independent Clocks (RD_CLK, WR_CLK), Block RAM • Independent Clocks (RD_CLK, WR_CLK), Distributed RAM • Independent Clocks (RD_CLK, WR_CLK), Built-in FIFO	**Synchronous** • Common Clock (CLK), Block RAM=Block Memory • Common Clock (CLK), Distributed RAM=Distributed Memory **Asynchronous** • Independent Clocks (RD_CLK, WR_CLK), Block RAM=Block Memory • Independent Clocks (RD_CLK, WR_CLK), Distributed RAM=Distributed Memory
-	**Read Mode** • Standard FIFO • First-Word Fall-Through	Synchronous and Asynchronous FIFO Generators only implement Standard FIFO. First-word fall-through ensures that there is always a valid data in the output port when fifo is not empty. The first-word fall-through (FWFT) provides the ability to look ahead to the next word available from the FIFO without issuing a read operation.
-	**Built-in FIFO Options** • Read Clock Frequency • Write Clock Frequency	Set Read, Write Clock frequencies. This option is only available for built-in FIFOs.
Data Port Parameters • Input Data Width • FIFO Depth	**Data Port Parameters** • Write Width • Write Depth • Read Width • Read Depth	These parameters define the FIFO size. For a Synchronous FIFO generated by FIFO Generator, Read Width and Read Depth are not available. Write Width=Input Data Width. Write Depth=FIFO Depth. For an Asynchronous FIFO generated by FIFO Generator, Read Depth is automatically calculated from the other three parameters. Write Depth=FIFO Depth+1, Read Depth=FIFO Depth+1. For FIFO Generator, actual FIFO Write Depth=FIFO Depth=Write Depth-1; actual FIFO Read Depth=FIFO Depth=Read Depth-1.

Table 3: **GUI Parameter Comparison *(Cont'd)***

Synchronous/ Asynchronous FIFO	FIFO Generator	Functionality of GUI Parameter (FIFO Generator= Synchronous/Asynchronous FIFO)
-	**Implementation Options** • Enable ECC • Use Embedded Registers in BRAM or FIFO (when possible)	Enable Virtex-6, Virtex-5 and Virtex-4 FPGA-specific features. These options are only available for block RAM and built-in based FIFOs.
Asynchronous • Almost Full Flag • Almost Empty Flag	**Optional Flags** • Almost Full Flag • Almost Empty Flag	Available for all FIFOs except built-in FIFOs. These parameters control generation of ALMOST FULL and ALMOST EMPTY flags. These flags indicate that FIFO is almost full or almost empty.
Handshaking Options • Write Acknowledge Flag • Write Error Flag • Read Acknowledge Flag • Read Error Flag (Active High or Active Low)	**Handshaking Options** • Write Acknowledge Flag • Overflow Flag • Valid Flag (Read Acknowledge) • Underflow Flag (Read Error)	These parameters control generation of FIFO status flags.
-	**Initialization** • Reset Pin • Asynchronous Reset • Synchronous Reset ♦ Full Flags Reset Value • Use Dout Reset • Enable Reset Synchronization	When Reset Pin is selected, reset is generated in input ports. Otherwise, reset is always tied to zero. Asynchronous Reset is supported for both Asynchronous and Synchronous FIFOs while Synchronous is only available in Synchronous FIFO. Synchronous Reset behavior is equal to SINIT in Synchronous FIFO Generator. Full flags reset value determine the value of full flags (FULL, ALMOST_FULL, PROG_FULL) during asynchronous reset. Set the value to '1' for backward-compatibility. Use dout reset to determine if the DOUT output resets to a specified value when the reset signal is asserted. Set to true for backward compatibility. Enable Reset Synchronization determines the use of internal reset synchronization logic. Set to true for backward compatibility.

Table 3: **GUI Parameter Comparison *(Cont'd)***

Synchronous/ Asynchronous FIFO	FIFO Generator	Functionality of GUI Parameter (FIFO Generator= Synchronous/Asynchronous FIFO)
-	**Programmable Flags** • Programmable Full Type	Programmable Full Type provides the following options: • No_Programmable_Full_Threshold • Single_Programmable_Full_Threshold_Constant • Multiple_Programmable_Full_Threshold_Constants • Single_Programmable_Full_Threshold_Input_Port • Multiple_Programmable_Full_Threshold_Input_Ports
	- full_threshold_assert_value	- Full threshold assert value is used to set the upper threshold value for the programmable full flag.
	- full_threshold_negate_value	- Full Threshold Negate is used to set the lower threshold value for the programmable full flag.
	• Programmable Empty Type	Programmable Empty Type provides the following options: • No_Programmable_Empty_Threshold • Single_Programmable_Empty_Threshold_Constant • Multiple_Programmable_Empty_Threshold_Constants • Single_Programmable_Empty_Threshold_Input_Port • Multiple_Programmable_Empty_Threshold_Input_Ports
	- empty_threshold_assert_value	- Empty Threshold Assert is used to set the lower threshold value for the programmable empty flag.
	- empty_threshold_negate_value	- Empty Threshold Negate is used to set the upper threshold value for the programmable empty flag.
Synchronous • Data Count • Data Count Width **Asynchronous** • Write Count • Write Count Width • Read Count • Read Count Width	**Common Clock** • Data Count ♦ Data Count Width **Independent Clock** • Write Data Count ♦ Write Data Count Width • Read Data Count ♦ Read Data Count Width	These parameters control generation of DATA COUNT.

Table 3: **GUI Parameter Comparison *(Cont'd)***

Synchronous/ Asynchronous FIFO	FIFO Generator	Functionality of GUI Parameter (FIFO Generator= Synchronous/Asynchronous FIFO)
Asynchronous • Layout (Create RPM)	-	Enable to locate the FIFO according to design attributes. When this option is selected, the Asynchronous FIFO is generated using Relationally Placed Macros (RPMs).
-	**Error Injection** • Single Bit Error Injection • Double Bit Error Injection	Enable Virtex-6 FPGA-specific features. These options are only available for block RAM and built-in based FIFOs.

Modifying the Instantiations of the Old Core

For each memory core instantiation, do the following:

1. Change the name of the module. (Only necessary if component name of the core has changed.)
2. Change the port names. For port name conversions, see Tables 4 and 5.
3. Modify your design, if necessary, to compensate for obsolete features. See "Core Compatibility," page 1 for information about difference between old and new cores.

Table 4: **Port Name Mapping: Asynchronous FIFO Cores**

Legacy Asynchronous FIFO Core		New FIFO Generator Core		Conversion Description	Functionality
Port	**Availability**	**Port**	**Availability**	**Port**	**Availability**
DIN[N:0]	Available	DIN[N:0]	Available	Same	Available
WR_EN	Available	WR_EN	Available	Same	Available
WR_CLK	Available	WR_CLK	Available	Same	Available
RD_EN	Available	RD_EN	Available	Same	Available
RD_CLK	Available	RD_CLK	Available	Same	Available
AINIT	Available	RST	Optional	Asynchronous reset	Available
FULL	Available	FULL	Available	Same	Available
ALMOST_ FULL	Optional	ALMOST_ FULL	Available	Same	Available
-	-	PROG_FULL	Optional	-	Optional
-	-	PROG_FULL_ THRESH	Optional	-	Optional
-	-	PROG_FULL_ THRESH_ ASSERT	Optional	-	Optional
-	-	PROG_FULL_ THRESH_ NEGATE	Optional	-	Optional

Table 4: **Port Name Mapping: Asynchronous FIFO Cores *(Cont'd)***

Legacy Asynchronous FIFO Core		New FIFO Generator Core		Conversion Description	Functionality
Port	**Availability**	**Port**	**Availability**	**Port**	**Availability**
WR_COUNT [W:0]	Optional	WR_DATA_ COUNT[D:0]	Optional	Direct replacement	Optional
WR_ACK	Optional	WR_ACK	Optional	Same	Optional
WR_ERR	Optional	OVERFLOW	Optional	Direct replacement	Optional
DOUT[N:0]	Available	DOUT[M:0]	Available	Same	Optional
EMPTY	Available	EMPTY	Available	Same	Optional
ALMOST_ EMPTY	Optional	ALMOST_ EMPTY	Optional	Same	Optional
-	-	PROG_EMPTY	Optional	-	Optional
-	-	PROG_EMPTY_ THRESH	Optional	-	Optional
-	-	PROG_EMPTY_ THRESH_ ASSERT	Optional	-	Optional
-	-	PROG_EMPTY_ THRESH_ NEGATE	Optional	-	Optional
RD_COUNT [:R:0]	Optional	RD_DATA_ COUNT[C:0]	Optional	Direct replacement	Optional
RD_ACK	Optional	VALID	Optional	Direct replacement	Optional
RD_ERR	Optional	UNDERFLOW	Optional	Direct replacement	Optional
-	-	SBITERR	Optional	-	Optional
-	-	DBITERR	Optional	-	Optional
-	-	INJECTSBITERR	Optional	-	-
-	-	INJECTDBITERR	Optional	-	-

Table 5: **Port Name Mapping: Synchronous FIFO Cores**

Legacy Synchronous FIFO Core		New FIFO Generator Core		Conversion Description	Functionality
Port	**Availability**	**Port**	**Availability**	**Port**	**Availability**
DIN[N:0]	Available	DIN[N:0]	Available	Same	Available
WR_EN	Available	WR_EN	Available	Same	Available
CLK	Available	CLK	Available	Same	Available
RD_EN	Available	RD_EN	Available	Same	Available
SINIT	Available	SRST	Optional	Direct replacement	Available

Table 5: **Port Name Mapping: Synchronous FIFO Cores *(Cont'd)***

Legacy Synchronous FIFO Core		New FIFO Generator Core		Conversion Description	Functionality
Port	**Availability**	**Port**	**Availability**	**Port**	**Availability**
FULL	Available	FULL	Available	Same	Available
-	-	ALMOST_ FULL	Available	-	Available
-	-	PROG_FULL	Optional	-	Optional
-	-	PROG_FULL_ THRESH	Optional	-	Optional
-	-	PROG_FULL_ THRESH_ ASSERT	Optional	-	Optional
-	-	PROG_FULL_ THRESH_ NEGATE	Optional	-	Optional
DATA_ COUNT[C:0]	Optional	DATA_COUNT [D:0]	Optional	FIFO Generator only supports Data Count Width from 1 to LOG2(FIFO depth). Additional logic: If Data Count Width is equal to LOG2(FIFO Depth)+1, need to link FULL port to the most significant bit of the DATA_COUNT port. (DATA_COUNT[D:0] = {FULL, DATA_COUNT [C:0]})	Optional
WR_ACK	Optional	WR_ACK	Optional	Same	Optional
WR_ERR	Optional	OVERFLOW	Optional	Direct replacement	Optional
DOUT[N:0]	Available	DOUT[M:0]	Available	Same	Optional
EMPTY	Available	EMPTY	Available	Same	Optional
-	-	ALMOST_ EMPTY	Optional	-	Optional
-	-	PROG_ EMPTY	Optional	-	Optional

Table 5: **Port Name Mapping: Synchronous FIFO Cores *(Cont'd)***

Legacy Synchronous FIFO Core		New FIFO Generator Core		Conversion Description	Functionality
Port	**Availability**	**Port**	**Availability**	**Port**	**Availability**
-	-	PROG_ EMPTY_ THRESH	Optional	-	Optional
-	-	PROG_EMPTY_ THRESH_ ASSERT	Optional	-	Optional
-	-	PROG_EMPTY_ THRESH_ NEGATE	Optional	-	Optional
RD_ACK	Optional	VALID	Optional	Direct replacement	Optional
RD_ERR	Optional	UNDERFLOW	Optional	Direct replacement	Optional
-	-	SBITERR	Optional	-	Optional
-	-	DBITERR	Optional	-	Optional
-	-	INJECTSBITERR	Optional	-	-
-	-	INJECTDBITERR	Optional	-	-

References

1. DS317, *LogiCORE IP FIFO Generator Data Sheet*
2. UG175, *LogiCORE IP FIFO Generator User Guide*

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
5/29/07	1.0	Initial Xilinx release.
9/4/07	2.0	Updated for release 2.0
10/17/07	2.5	Updated for FIFO Generator core release v4.2.
3/24/08	3.0	Updated for FIFO Generator core release v4.3.
6/2/08	3.1	Updated mapping of Distributed RAM FIFO for Synchronous FIFO to Common Clock Shift-Register FIFO for FIFO Generator core. Updated for FIFO Generator core release v4.4.
4/24/09	4.0	Updated for FIFO Generator core v5.1.
6/24/09	4.5	Updated for FIFO Generator core v5.2. Added information about the FIFO Generator Migration Kit.
9/16/09	5.0	Updated for FIFO Generator core v5.3.
4/19/10	6.0	Updated for FIFO Generator core v6.1.
7/23/10	7.0	Updated for FIFO Generator core v6.2.
9/21/10	8.0	Updated for FIFO Generator core v7.2. Moved information about migrating FIFO Generator cores that are not Synchronous or Asynchronous to UG175, *LogiCORE IP FIFO Generator User Guide*.

Notice of Disclaimer