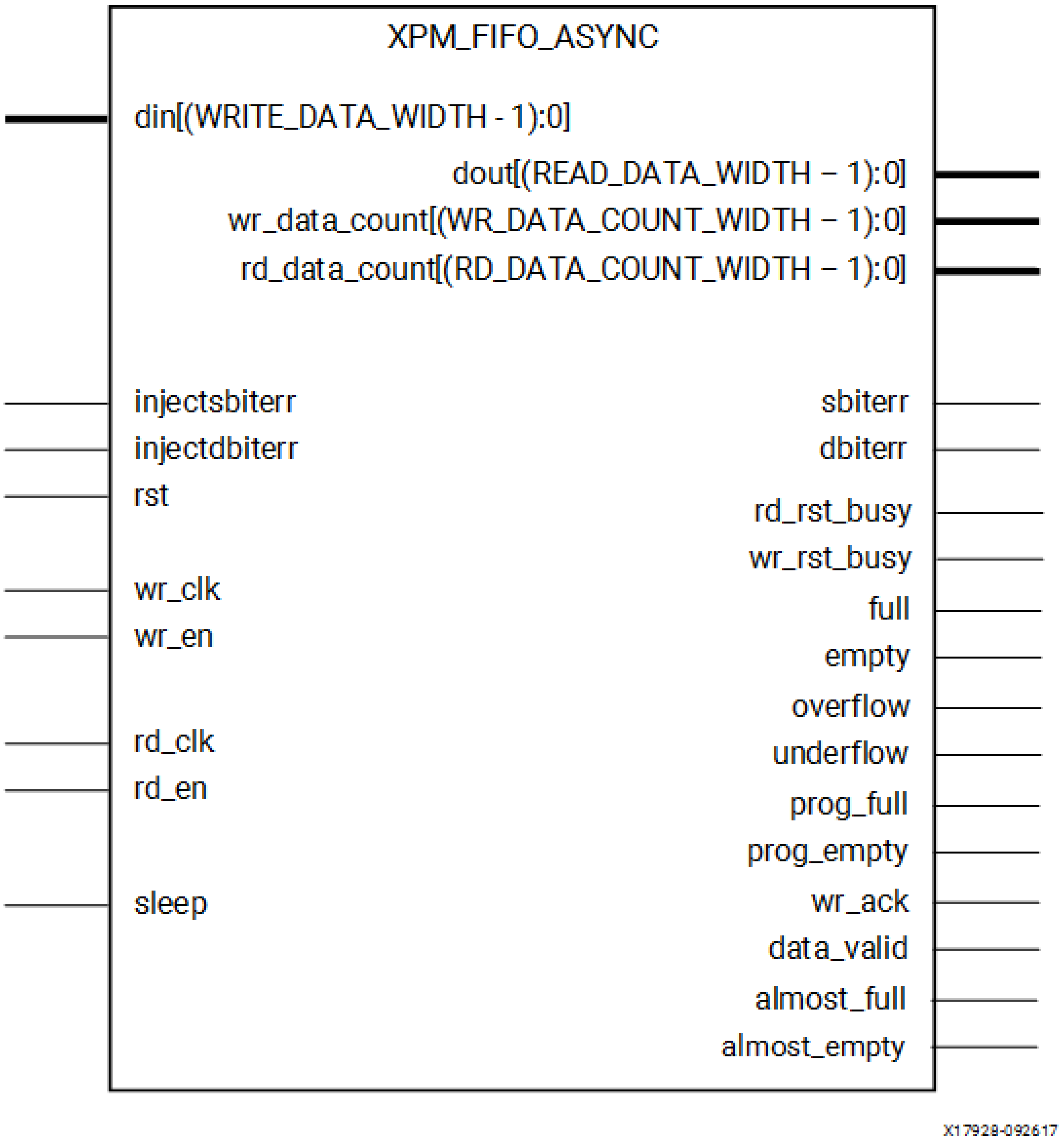
Vivado Design Suite 7 Series FPGA and Zynq 7000 SoC Libraries Guide (UG953)

XPM_FIFO_ASYNC

XPM_FIFO_ASYNC

Parameterized Macro: Asynchronous FIFO

- MACRO_GROUP: XPM
- MACRO_SUBGROUP: XPM_FIFO



Introduction

This macro is used to instantiate an asynchronous FIFO.
The following describes the basic write and read operation of an XPM_FIFO instance. It does not distinguish between FIFO types, clock domain or read mode.

- After a user issues a reset, the user should wait until the busy signals go low before issuing another reset.
- All synchronous signals are sensitive to the rising edge of wr_clk/rd_clk, which is assumed to be a buffered and toggling clock signal behaving according to target device and FIFO/memory primitive requirements.
- A write operation is performed when the FIFO is not full and wr_en is asserted on each wr_clk cycle.
- A read operation is performed when the FIFO is not empty and rd_en is asserted on each rd_clk cycle.
- The number of clock cycles required for XPM FIFO to react to dout, full, and empty changes depends on the CLOCK_DOMAIN, READ_MODE, and FIFO_READ_LATENCY settings.
    - It can take more than one rd_clk cycle to deassert empty due to write operation (wr_en = 1).
    - It can take more than one rd_clk cycle to present the read data on dout port upon assertion of rd_en.
    - It may take more than one wr_clk cycle to deassert full due to read operation (rd_en = 1).
- All write operations are gated by the value of wr_en and full on the initiating wr_clk cycle.
- All read operations are gated by the value of rd_en and empty on the initiating rd_clk cycle.
- Undriven or unknown values provided on module inputs will produce undefined output port behavior.
- wr_en/rd_en should not be toggled when reset (rst) or wr_rst_busy or rd_rst_busy is asserted.
- Assertion/deassertion of prog_full happens only when full is deasserted.
- Assertion/deassertion of prog_empty happens only when empty is deasserted.

Note: In an asynchronous FIFO (that is, two independent clocks), the RELATED_CLOCKS attribute should be set to TRUE only if both the wr_clk and rd_clk are generated from the same source.

Timing Diagrams

Figure: Reset Behavior

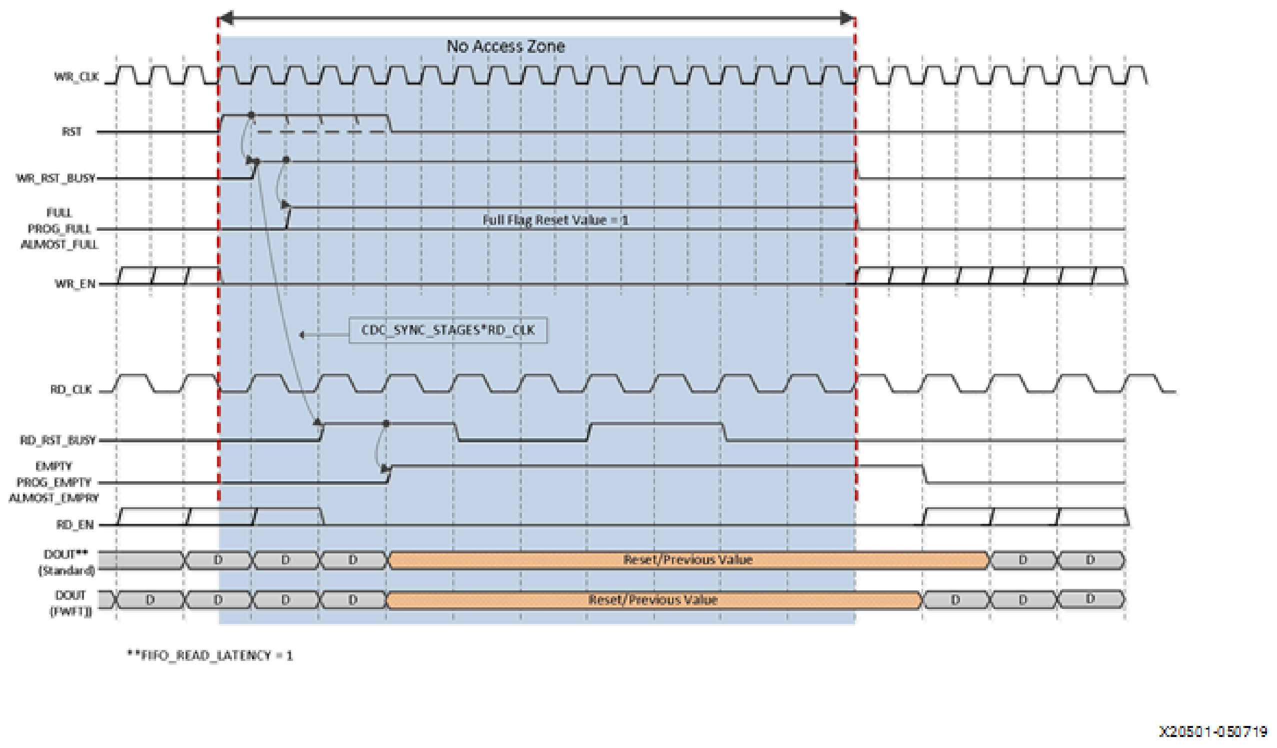


Figure: Standard Write Operation.

FIFO_WRITE_DEPTH=16, PROG_FULL_THRESH=6

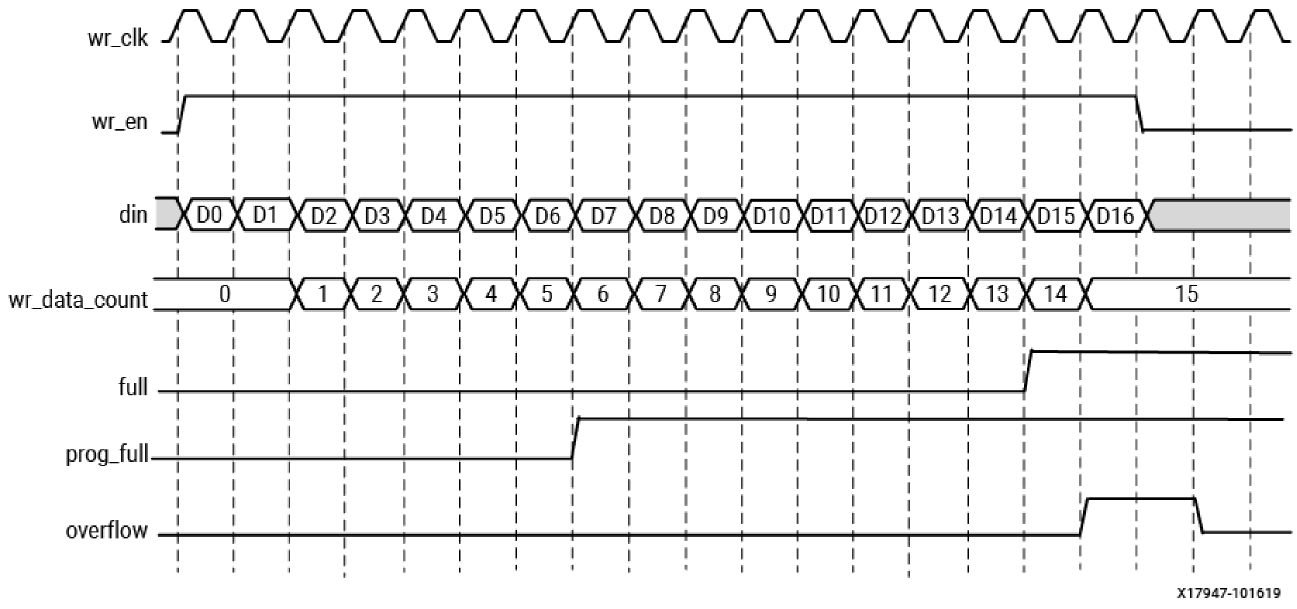


Figure: Standard Read Operation.

FIFO_WRITE_DEPTH=16, PROG_EMPTY_THRESH=3,
FIFO_READ_LATENCY=1

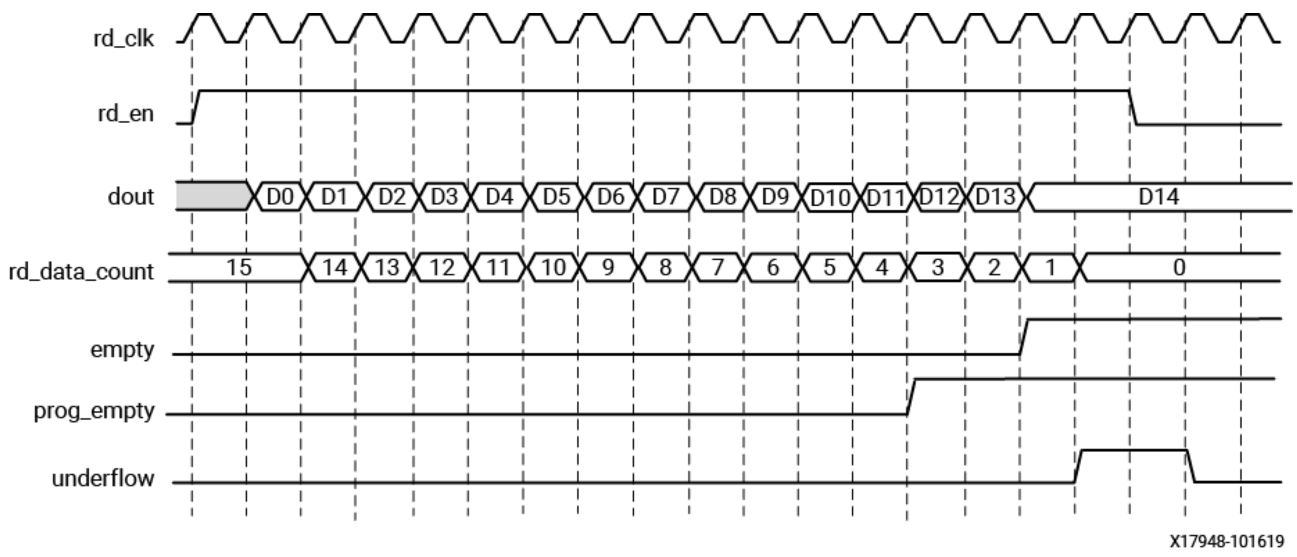


Figure: Standard Read Operation.

FIFO_WRITE_DEPTH=16, PROG_EMPTY_THRESH=3, FIFO_READ_LATENCY=3

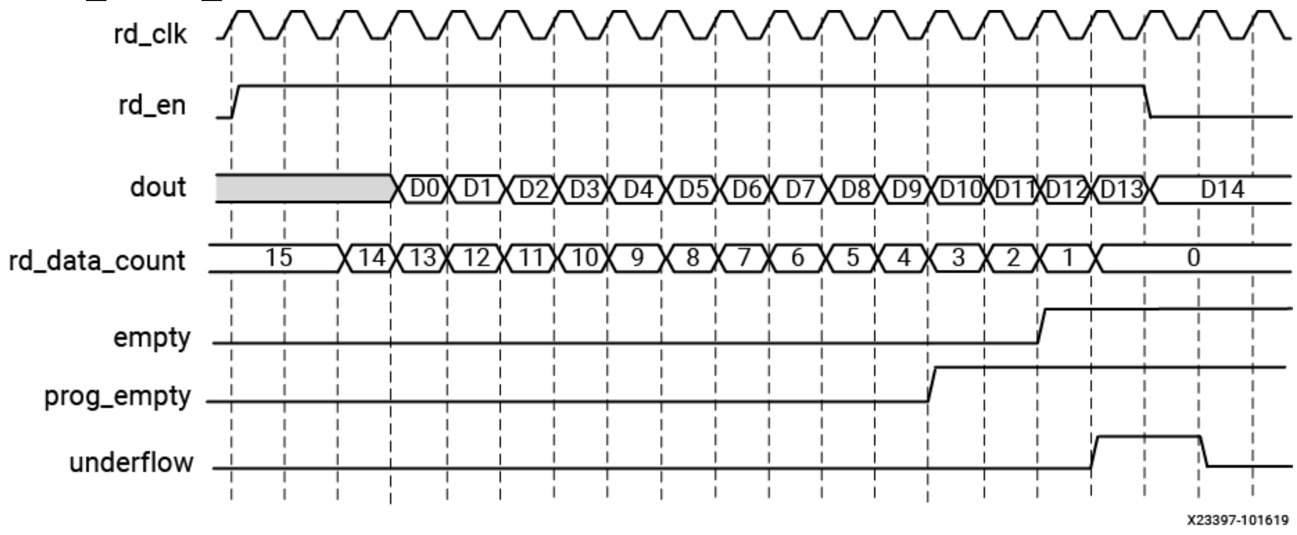


Figure: Write Operation.

READ_MODE=FWFT, FIFO_WRITE_DEPTH=16, PROG_FULL_THRESH=7

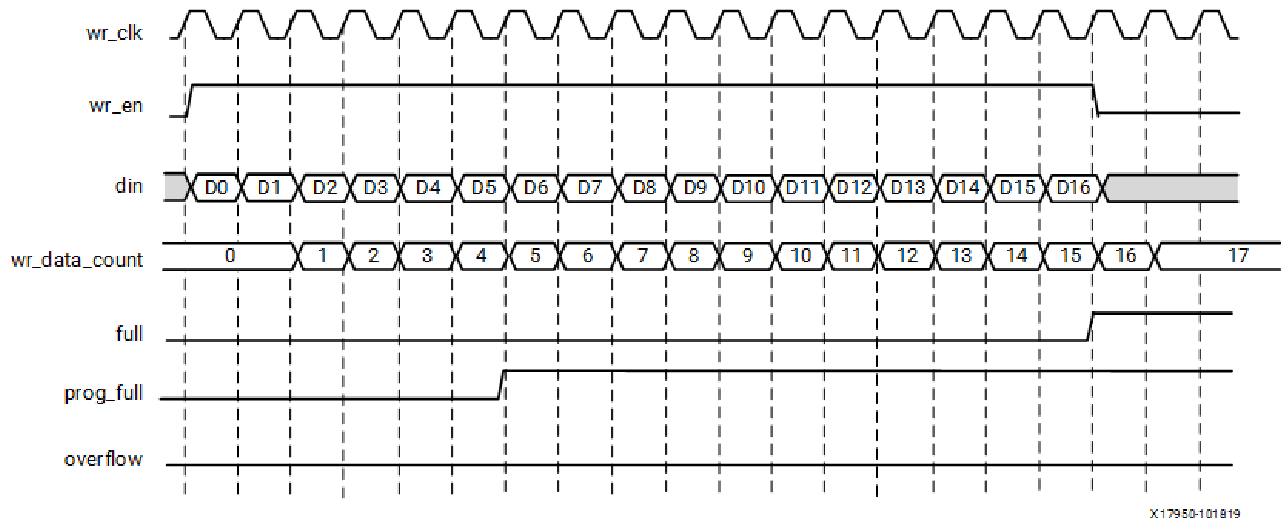


Figure: Read Operation.

READ_MODE=FWFT, FIFO_WRITE_DEPTH=16, PROG_EMPTY_THRESH=5

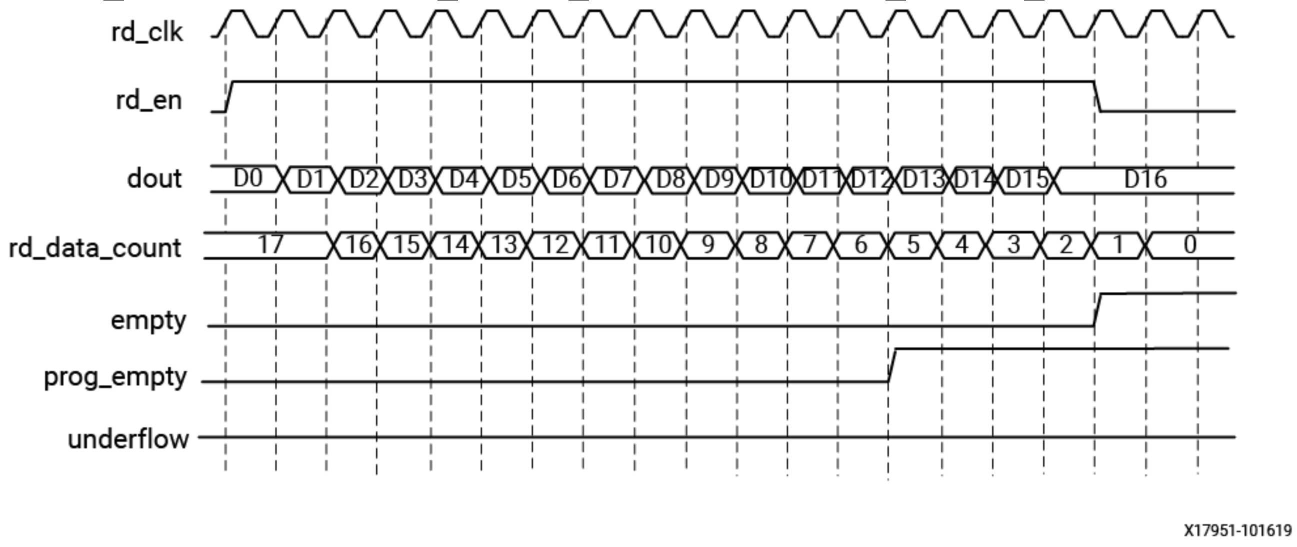


Figure: Standard Write Operation with Empty Deassertion.

FIFO_WRITE_DEPTH=16

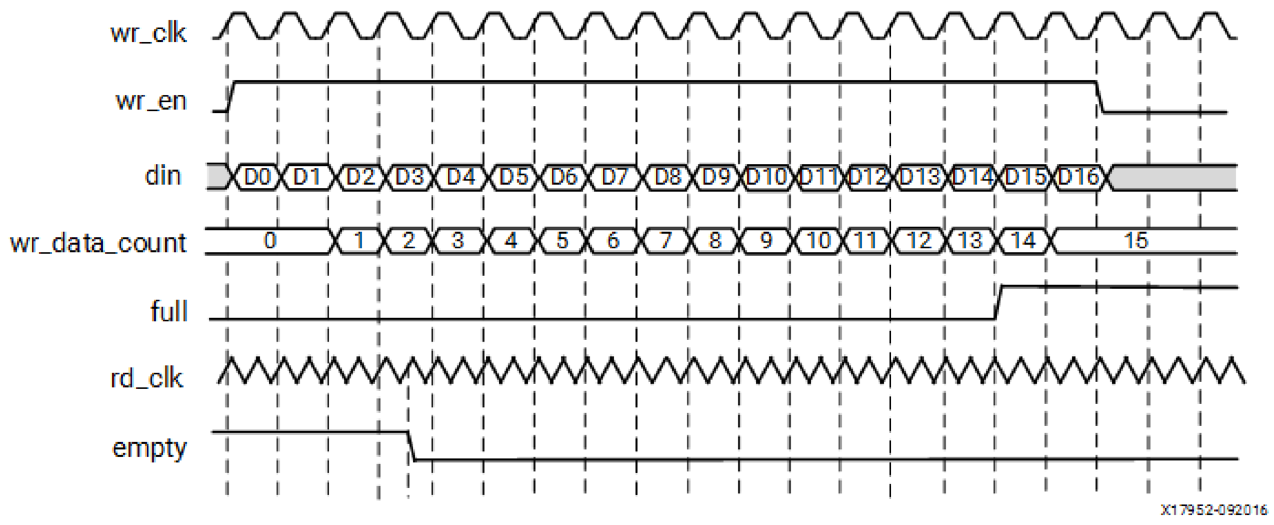
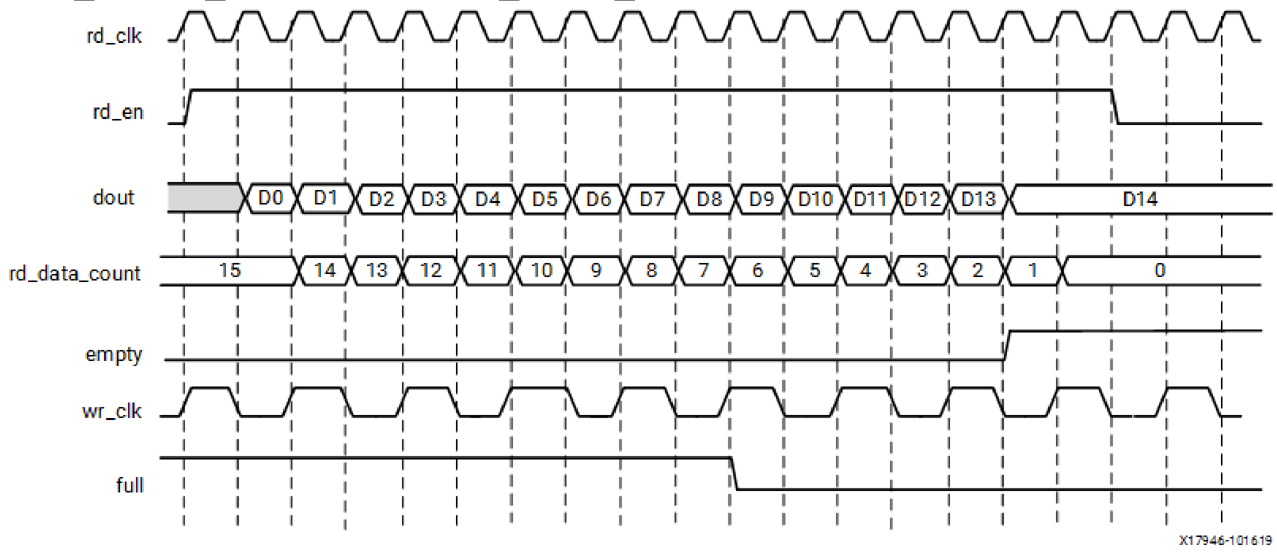


Figure: Standard Read Operation with Full Deassertion.

FIFO_WRITE_DEPTH=16, FIFO_READ_LATENCY=1



Latency

This section defines the latency in which different output signals of the FIFO are updated in response to read or write operations for standard read mode and FWFT read mode implementations.

Note: For LUTRAM, auto, mixed, and distributed RAM primitive types with FIFO_READ_LATENCY = 0 and READ_MODE = "std" will infer LUTRAM memory without a rd_clk. In this configuration, the dout port will toggle with respect to the posedge of the write clock only. Take care of this path as data is driven out of the LUT directly without any `rd_clk` domain flop.

The following table defines the write port flags update latency due to a write operation.

Table: Standard Read Mode — Write Port Flags Due to Write Operation

Signal	Latency (wr_clk)
full	0
almost_full	0
prog_full	2
wr_ack	1
overflow	0
wr_data_count	2

The following table defines the read port flags update latency due to a read operation.

Table: Standard Read Mode — Read Port Flags Due to Read Operation

Signal	Latency (rd_clk)
empty	0
almost_empty	0
prog_empty	1
data_valid	FIFO_READ_LATENCY
underflow	0
rd_data_count	2

The following table defines the write port flags update latency due to a read operation. N is the number of synchronization stages.

Table: Standard Read Mode — Write Port Flags Due to Read Operations

Signal	Latency

Signal	Latency
full	1 rd_clk + (N+2) wr_clk
almost_full	1 rd_clk + (N+3) wr_clk
prog_full	1 rd_clk + (N+2) wr_clk
wr_ack	N/A
overflow	N/A
wr_data_count	1 rd_clk + (N+2) wr_clk

The following table defines the read port flags update latency due to a write operation. N is the number of synchronization stages. In this example, N is 2.

Table: Standard Read Mode — Read Port Flags Due to Write Operation

Signal	Latency
empty	1 wr_clk + (N+2) rd_clk
almost_empty	1 wr_clk + (N+3) rd_clk
prog_empty	1 wr_clk + (N+3) rd_clk
data_valid	N/A
underflow	N/A
rd_data_count	1 wr_clk + (N+2) rd_clk

The following table defines the write port flags update latency due to a write operation.

Table: FWFT Read Mode — Write Port Flags Due to Write Operation

Signal	Latency
full	2
almost_full	1
prog_full	0

Signal	Latency
wr_ack	1
overflow	2
wr_data_count	2

The following table defines the read port flags update latency due to a read operation.

Table: FWFT Read Mode — Read Port Flags Due to Read Operation

Signal	Latency
empty	2
almost_empty	2
prog_empty	3
data_valid	0
underflow	2
rd_data_count	2

The following table defines the write port flags update latency due to a read operation. N is the number of synchronization stages.

Table: FWFT Read Mode — Write Port Flags Due to Read Operation

Signal	Latency
full	1 rd_clk + (N+3) wr_clk
almost_full	1 rd_clk + (N+4) wr_clk
prog_full	1 rd_clk + (N+5) wr_clk
wr_ack	N/A
overflow	N/A
wr_data_count	1 rd_clk + (N+3) wr_clk

The following table defines the read port flags update latency due to a write operation. N is the number of synchronization stages. In this example, N is 2.

Table: FWFT Read Mode — Read Port Flags Due to Write Operation

Signal	Latency
empty	1 wr_clk + (N+4) rd_clk
almost_empty	1 wr_clk + (N+4) rd_clk
prog_empty	1 wr_clk + (N+3) rd_clk
data_valid	1 wr_clk + (N+4) rd_clk
underflow	N/A
rd_data_count	1 wr_clk + (N+4) rd_clk

Port Descriptions

Port	Direction	Width	Domain	Sense	Handling if Unused	Function
almost_empty	Output	1	rd_clk	LEVEL_HIGH	DoNotCare	Almost Empty : When asserted, this signal indicates that only one more read can be performed before the FIFO goes to empty.
almost_full	Output	1	wr_clk	LEVEL_HIGH	DoNotCare	Almost Full: When asserted, this signal indicates that only one more write can be performed before the FIFO is full.
data_valid	Output	1	rd_clk	LEVEL_HIGH	DoNotCare	Read Data Valid: When asserted, this signal indicates that valid data is

Port	Direction	Width	Domain	Sense	Handling if Unused	Function
						available on the output bus (dout).
dbiterr	Output	1	rd_clk	LEVEL_HIGH	DoNotCare	Double Bit Error: Indicates that the ECC decoder detected a double-bit error and data in the FIFO core is corrupted.
din	Input	WRITE_DATA_WIDTH	wr_clk	NA	Active	Write Data: The input data bus used when writing the FIFO.
dout	Output	READ_DATA_WIDTH	rd_clk	NA	Active	Read Data: The output data bus is driven when reading the FIFO.
empty	Output	1	rd_clk	LEVEL_HIGH	Active	Empty Flag: When asserted, this signal indicates that the FIFO is empty. Read requests are ignored when the FIFO is empty, initiating a read while empty is not destructive to the FIFO.
full	Output	1	wr_clk	LEVEL_HIGH	Active	Full Flag: When asserted, this signal indicates that the FIFO is full. Write requests are ignored when the FIFO is full, initiating a write when the FIFO is full is not destructive to the contents of the FIFO.
injectdbiterr	Input	1	wr_clk	LEVEL_HIGH	0	Double Bit Error Injection: Injects a double bit error if

Port	Direction	Width	Domain	Sense	Handling if Unused	Function
						the ECC feature is used on block RAMs or UltraRAM macros.
injectsbiterr	Input	1	wr_clk	LEVEL_HIGH	0	Single Bit Error Injection: Injects a single bit error if the ECC feature is used on block RAMs or UltraRAM macros.
overflow	Output	1	wr_clk	LEVEL_HIGH	DoNotCare	Overflow: This signal indicates that a write request (wren) during the prior clock cycle was rejected, because the FIFO is full. Overflowing the FIFO is not destructive to the contents of the FIFO.
prog_empty	Output	1	rd_clk	LEVEL_HIGH	DoNotCare	Programmable Empty: This signal is asserted when the number of words in the FIFO is less than or equal to the programmable empty threshold value. It is de-asserted when the number of words in the FIFO exceeds the programmable empty threshold value.
prog_full	Output	1	wr_clk	LEVEL_HIGH	DoNotCare	Programmable Full: This signal is asserted when the number of words in the FIFO is greater than or equal to the programmable full threshold value.

Port	Direction	Width	Domain	Sense	Handling if Unused	Function
						It is de-asserted when the number of words in the FIFO is less than the programmable full threshold value.
rd_clk	Input	1	NA	EDGE_RISING	Active	Read clock: Used for read operation. rd_clk must be a free running clock.
rd_data_count	Output	RD_DATA_COUNT_WIDTH	rd_clk	NA	DoNotCare	Read Data Count: This bus indicates the number of words read from the FIFO.
rd_en	Input	1	rd_clk	LEVEL_HIGH	Active	Read Enable: If the FIFO is not empty, asserting this signal causes data (on dout) to be read from the FIFO. • Must be held active-low when rd_rst_busy is active high.
rd_rst_busy	Output	1	rd_clk	LEVEL_HIGH	Active	Read Reset Busy: Active-High indicator that the FIFO read domain is currently in a reset state.
rst	Input	1	wr_clk	LEVEL_HIGH	Active	Reset: Must be synchronous to wr_clk. The clock(s) can be unstable at the time of applying reset, but reset must be released only after the clock(s) is/are stable.

Port	Direction	Width	Domain	Sense	Handling if Unused	Function
sbiterr	Output	1	rd_clk	LEVEL_HIGH	DoNotCare	Single Bit Error: Indicates that the ECC decoder detected and fixed a single-bit error.
sleep	Input	1	NA	LEVEL_HIGH	0	Dynamic power saving: If sleep is High, the memory/fifo block is in power saving mode.
underflow	Output	1	rd_clk	LEVEL_HIGH	DoNotCare	Underflow: Indicates that the read request (rd_en) during the previous clock cycle was rejected because the FIFO is empty. Under flowing the FIFO is not destructive to the FIFO.
wr_ack	Output	1	wr_clk	LEVEL_HIGH	DoNotCare	Write Acknowledge: This signal indicates that a write request (wr_en) during the prior clock cycle is succeeded.
wr_clk	Input	1	NA	EDGE_RISING	Active	Write clock: Used for write operation. wr_clk must be a free running clock.
wr_data_count	Output	WR_DATA_COUNT_WIDTH	wr_clk	NA	DoNotCare	Write Data Count: This bus indicates the number of words written into the FIFO.
wr_en	Input	1	wr_clk	LEVEL_HIGH	Active	Write Enable: If the FIFO is not full, asserting this signal causes data (on din) to be written to the FIFO.

Port	Direction	Width	Domain	Sense	Handling if Unused	Function
						• Must be held active-low when rst or wr_rst_busy is active high.
wr_rst_busy	Output	1	wr_clk	LEVEL_HIGH	Active-High	Write Reset Busy: Active-High indicator that the FIFO write domain is currently in a reset state.

Design Entry Method

Instantiation	Yes
Inference	No
IP and IP Integrator Catalog	No

Available Attributes

Attribute	Type	Allowed Values	Default	Description
CASCADE_HEIGHT	DECIMAL	0 to 64	0	0- No Cascade Height, Allow Vivado Synthesis to choose. 1 or more - Vivado Synthesis sets the specified value as Cascade Height.
CDC_SYNC_STAGES	DECIMAL	2 to 8	2	Specifies the number of synchronization stages on the CDC path • Must be < 5 if FIFO_WRITE_DEPTH = 16
DOUT_RESET_VALUE	STRING	String	"0"	Reset value of read data path.
ECC_MODE	STRING	"no_ecc", "en_ecc"	"no_ecc"	• "no_ecc" - Disables ECC • "en_ecc" - Enables both ECC Encoder and Decoder

Attribute	Type	Allowed Values	Default	Description
				NOTE: ECC_MODE should be "no_ecc" if FIFO_MEMORY_TYPE is set to "auto". Violating this may result incorrect behavior.
EN_SIM_ASSERT_ERR	STRING	String	"warning"	• "warning" - Report warning message for FIFO overflow and underflow in simulation. • "error" - Report error message for FIFO overflow and underflow in simulation. • "fatal" - Report fatal message for FIFO overflow and underflow in simulation.
FIFO_MEMORY_TYPE	STRING	"auto", "block", "distributed"	"auto"	Designate the fifo memory primitive (resource type) to use. • "auto"- Allow Vivado Synthesis to choose • "block"- Block RAM FIFO • "distributed"- Distributed RAM FIFO NOTE: There may be a behavior mismatch if Block RAM or Ultra RAM specific features, like ECC or Asymmetry, are selected with FIFO_MEMORY_TYPE set to "auto".
FIFO_READ_LATENCY	DECIMAL	0 to 10	1	Number of output register stages in the read data path. • If READ_MODE = "fwft", then the only applicable value is 0.
FIFO_WRITE_DEPTH	DECIMAL	16 to 4194304	2048	Defines the FIFO Write Depth, must be power of two. • In standard READ_MODE, the effective depth = FIFO_WRITE_DEPTH-1 • In First-Word-Fall-Through READ_MODE, the effective depth = FIFO_WRITE_DEPTH+1 NOTE: The maximum FIFO size (width x depth) is limited to 150-Megabits.
FULL_RESET_VALUE	DECIMAL	0 to 1	0	Sets full, almost_full and prog_full to FULL_RESET_VALUE during reset

Attribute	Type	Allowed Values	Default	Description
PROG_EMPTY_THRESH	DECIMAL	3 to 4194301	10	Specifies the minimum number of read words in the FIFO at or below which prog_empty is asserted. • Min_Value = 3 + (READ_MODE_VAL*2) • Max_Value = (FIFO_WRITE_DEPTH-3) - (READ_MODE_VAL*2) If READ_MODE = "std", then READ_MODE_VAL = 0; Otherwise READ_MODE_VAL = 1. NOTE: The default threshold value is dependent on default FIFO_WRITE_DEPTH value. If FIFO_WRITE_DEPTH value is changed, ensure the threshold value is within the valid range though the programmable flags are not used.
PROG_FULL_THRESH	DECIMAL	5 to 4194301	10	Specifies the maximum number of write words in the FIFO at or above which prog_full is asserted. • Min_Value = 3 + (READ_MODE_VAL*2*(FIFO_WRITE_DEPTH/FIFO_READ_DEPTH))+CDC_SYNC_STAGES • Max_Value = (FIFO_WRITE_DEPTH-3) - (READ_MODE_VAL*2*(FIFO_WRITE_DEPTH/FIFO_READ_DEPTH)) If READ_MODE = "std", then READ_MODE_VAL = 0; Otherwise READ_MODE_VAL = 1. NOTE: The default threshold value is dependent on default FIFO_WRITE_DEPTH value. If FIFO_WRITE_DEPTH value is changed, ensure the threshold value is within the valid range though the programmable flags are not used.
RD_DATA_COUNT_WIDTH	DECIMAL	1 to 23	1	Specifies the width of rd_data_count. To reflect the correct value, the width should be log2(FIFO_READ_DEPTH)+1. • FIFO_READ_DEPTH = FIFO_WRITE_DEPTH*WRITE_DATA_WIDTH/READ_DATA_WIDTH
READ_DATA_WIDTH	DECIMAL	1 to 4096	32	Defines the width of the read data port, dout

Attribute	Type	Allowed Values	Default	Description
				• Write and read width aspect ratio must be 1:1, 1:2, 1:4, 1:8, 8:1, 4:1 and 2:1 • For example, if WRITE_DATA_WIDTH is 32, then the READ_DATA_WIDTH must be 32, 64,128, 256, 16, 8, 4. NOTE: • READ_DATA_WIDTH should be equal to WRITE_DATA_WIDTH if FIFO_MEMORY_TYPE is set to "auto". Violating this may result incorrect behavior. • The maximum FIFO size (width x depth) is limited to 150-Megabits.
READ_MODE	STRING	"std", "fwft"	"std"	• "std"- standard read mode • "fwft"- First-Word-Fall-Through read mode
RELATED_CLOCKS	DECIMAL	0 to 1	0	Specifies if the wr_clk and rd_clk are related having the same source but different clock ratios
SIM_ASSERT_CHK	DECIMAL	0 to 1	0	0- Disable simulation message reporting. Messages related to potential misuse will not be reported. 1- Enable simulation message reporting. Messages related to potential misuse will be reported.
USE_ADV_FEATURES	STRING	String	"0707"	Enables data_valid, almost_empty, rd_data_count, prog_empty, underflow, wr_ack, almost_full, wr_data_count, prog_full, overflow features.

Attribute	Type	Allowed Values	Default	Description
				• Setting USE_ADV_FEATURES[0] to 1 enables overflow flag; Default value of this bit is 1 • Setting USE_ADV_FEATURES[1] to 1 enables prog_full flag; Default value of this bit is 1 • Setting USE_ADV_FEATURES[2] to 1 enables wr_data_count; Default value of this bit is 1 • Setting USE_ADV_FEATURES[3] to 1 enables almost_full flag; Default value of this bit is 0 • Setting USE_ADV_FEATURES[4] to 1 enables wr_ack flag; Default value of this bit is 0 • Setting USE_ADV_FEATURES[8] to 1 enables underflow flag; Default value of this bit is 1 • Setting USE_ADV_FEATURES[9] to 1 enables prog_empty flag; Default value of this bit is 1 • Setting USE_ADV_FEATURES[10] to 1 enables rd_data_count; Default value of this bit is 1 • Setting USE_ADV_FEATURES[11] to 1 enables almost_empty flag; Default value of this bit is 0 • Setting USE_ADV_FEATURES[12] to 1 enables data_valid flag; Default value of this bit is 0
WAKEUP_TIME	DECIMAL	0 to 2	0	• 0 - Disable sleep • 2 - Use Sleep Pin NOTE: WAKEUP_TIME should be 0 if FIFO_MEMORY_TYPE is set to "auto". Violating this may result incorrect behavior.
WR_DATA_COUNT_WIDTH	DECIMAL	1 to 23	1	Specifies the width of wr_data_count. To reflect the correct value, the width should be log2(FIFO_WRITE_DEPTH)+1.
WRITE_DATA_WIDTH	DECIMAL	1 to 4096	32	Defines the width of the write data port, din • Write and read width aspect ratio must be 1:1, 1:2, 1:4, 1:8, 8:1, 4:1 and 2:1 • For example, if WRITE_DATA_WIDTH is 32, then the READ_DATA_WIDTH must be 32, 64,128, 256, 16, 8, 4.

Attribute Type Allowed Values Default	Description
NOTE: • WRITE_DATA_WIDTH should be equal to READ_DATA_WIDTH if FIFO_MEMORY_TYPE is set to "auto". Violating this may result incorrect behavior. • The maximum FIFO size (width x depth) is limited to 150-Megabits.	

VHDL Instantiation Template

```
-- xpm_fifo_async: Asynchronous FIFO
-- Xilinx Parameterized Macro, version 2025.1

xpm_fifo_async_inst : xpm_fifo_async
generic map (
   CASCADE_HEIGHT => 0,            -- DECIMAL
   CDC_SYNC_STAGES => 2,           -- DECIMAL
   DOUT_RESET_VALUE => "0",        -- String
   ECC_MODE => "no_ecc",           -- String
   EN_SIM_ASSERT_ERR => "warning", -- String
   FIFO_MEMORY_TYPE => "auto",     -- String
   FIFO_READ_LATENCY => 1,         -- DECIMAL
   FIFO_WRITE_DEPTH => 2048,       -- DECIMAL
   FULL_RESET_VALUE => 0,          -- DECIMAL
   PROG_EMPTY_THRESH => 10,        -- DECIMAL
   PROG_FULL_THRESH => 10,         -- DECIMAL
   RD_DATA_COUNT_WIDTH => 1,       -- DECIMAL
   READ_DATA_WIDTH => 32,          -- DECIMAL
   READ_MODE => "std",             -- String
   RELATED_CLOCKS => 0,            -- DECIMAL
   SIM_ASSERT_CHK => 0,            -- DECIMAL; 0=disable simulation
messages, 1=enable simulation messages
   USE_ADV_FEATURES => "0707",     -- String
   WAKEUP_TIME => 0,               -- DECIMAL
   WRITE_DATA_WIDTH => 32,         -- DECIMAL
   WR_DATA_COUNT_WIDTH => 1        -- DECIMAL
)
port map (
   almost_empty => almost_empty,   -- 1-bit output: Almost Empty :
When asserted, this signal indicates that only one more read can be
performed
                                   -- before the FIFO goes to
empty.

   almost_full => almost_full,     -- 1-bit output: Almost Full:
When asserted, this signal indicates that only one more write can
be performed
                                   -- before the FIFO is full.

   data_valid => data_valid,       -- 1-bit output: Read Data
Valid: When asserted, this signal indicates that valid data is
available on the
                                   -- output bus (dout).

   dbiterr => dbiterr,             -- 1-bit output: Double Bit
Error: Indicates that the ECC decoder detected a double-bit error
and data in the
                                   -- FIFO core is corrupted.

   dout => dout,                   -- READ_DATA_WIDTH-bit output:
Read Data: The output data bus is driven when reading the FIFO.
   empty => empty,                 -- 1-bit output: Empty Flag:
When asserted, this signal indicates that the FIFO is empty. Read
requests are
                                   -- ignored when the FIFO is
empty, initiating a read while empty is not destructive to the
FIFO.

   full => full,                   -- 1-bit output: Full Flag: When
asserted, this signal indicates that the FIFO is full. Write
requests are
                                   -- ignored when the FIFO is
full, initiating a write when the FIFO is full is not destructive
to the contents
                                   -- of the FIFO.

   overflow => overflow,           -- 1-bit output: Overflow: This
signal indicates that a write request (wren) during the prior clock
cycle was
                                   -- rejected, because the FIFO is
```

```
full. Overflowing the FIFO is not destructive to the contents of
the FIFO.

   prog_empty => prog_empty,       -- 1-bit output: Programmable
Empty: This signal is asserted when the number of words in the FIFO
is less than
                                   -- or equal to the programmable
empty threshold value. It is de-asserted when the number of words
in the FIFO
                                   -- exceeds the programmable
empty threshold value.

   prog_full => prog_full,         -- 1-bit output: Programmable
Full: This signal is asserted when the number of words in the FIFO
is greater
                                   -- than or equal to the
programmable full threshold value. It is de-asserted when the
number of words in the
                                   -- FIFO is less than the
programmable full threshold value.

   rd_data_count => rd_data_count, -- RD_DATA_COUNT_WIDTH-bit
output: Read Data Count: This bus indicates the number of words
read from the FIFO.
   rd_rst_busy => rd_rst_busy,     -- 1-bit output: Read Reset
Busy: Active-High indicator that the FIFO read domain is currently
in a reset state.
   sbiterr => sbiterr,             -- 1-bit output: Single Bit
Error: Indicates that the ECC decoder detected and fixed a single-
bit error.
   underflow => underflow,         -- 1-bit output: Underflow:
Indicates that the read request (rd_en) during the previous clock
cycle was
                                   -- rejected because the FIFO is
empty. Under flowing the FIFO is not destructive to the FIFO.

   wr_ack => wr_ack,               -- 1-bit output: Write
Acknowledge: This signal indicates that a write request (wr_en)
during the prior clock
                                   -- cycle is succeeded.

   wr_data_count => wr_data_count, -- WR_DATA_COUNT_WIDTH-bit
```

```
output: Write Data Count: This bus indicates the number of words
written into the
                                   -- FIFO.

   wr_rst_busy => wr_rst_busy,     -- 1-bit output: Write Reset
Busy: Active-High indicator that the FIFO write domain is currently
in a reset
                                   -- state.

   din => din,                     -- WRITE_DATA_WIDTH-bit input:
Write Data: The input data bus used when writing the FIFO.
   injectdbiterr => injectdbiterr, -- 1-bit input: Double Bit Error
Injection: Injects a double bit error if the ECC feature is used on
block RAMs
                                   -- or UltraRAM macros.

   injectsbiterr => injectsbiterr, -- 1-bit input: Single Bit Error
Injection: Injects a single bit error if the ECC feature is used on
block RAMs
                                   -- or UltraRAM macros.

   rd_clk => rd_clk,               -- 1-bit input: Read clock: Used
for read operation. rd_clk must be a free running clock.
   rd_en => rd_en,                 -- 1-bit input: Read Enable: If
the FIFO is not empty, asserting this signal causes data (on dout)
to be read
                                   -- from the FIFO. Must be held
active-low when rd_rst_busy is active high.

   rst => rst,                     -- 1-bit input: Reset: Must be
synchronous to wr_clk. The clock(s) can be unstable at the time of
applying
                                   -- reset, but reset must be
released only after the clock(s) is/are stable.

   sleep => sleep,                 -- 1-bit input: Dynamic power
saving: If sleep is High, the memory/fifo block is in power saving
mode.
   wr_clk => wr_clk,               -- 1-bit input: Write clock:
Used for write operation. wr_clk must be a free running clock.
   wr_en => wr_en                  -- 1-bit input: Write Enable: If
the FIFO is not full, asserting this signal causes data (on din) to
```

```
be written
                                   -- to the FIFO. Must be held
active-low when rst or wr_rst_busy is active high.

);

-- End of xpm_fifo_async_inst instantiation
```

Verilog Instantiation Template

```
// xpm_fifo_async: Asynchronous FIFO
// Xilinx Parameterized Macro, version 2025.1

xpm_fifo_async #(
   .CASCADE_HEIGHT(0),           // DECIMAL
   .CDC_SYNC_STAGES(2),          // DECIMAL
   .DOUT_RESET_VALUE("0"),       // String
   .ECC_MODE("no_ecc"),          // String
   .EN_SIM_ASSERT_ERR("warning"), // String
   .FIFO_MEMORY_TYPE("auto"),    // String
   .FIFO_READ_LATENCY(1),        // DECIMAL
   .FIFO_WRITE_DEPTH(2048),      // DECIMAL
   .FULL_RESET_VALUE(0),         // DECIMAL
   .PROG_EMPTY_THRESH(10),       // DECIMAL
   .PROG_FULL_THRESH(10),        // DECIMAL
   .RD_DATA_COUNT_WIDTH(1),      // DECIMAL
   .READ_DATA_WIDTH(32),         // DECIMAL
   .READ_MODE("std"),            // String
   .RELATED_CLOCKS(0),           // DECIMAL
   .SIM_ASSERT_CHK(0),           // DECIMAL; 0=disable simulation
messages, 1=enable simulation messages
   .USE_ADV_FEATURES("0707"),    // String
   .WAKEUP_TIME(0),              // DECIMAL
   .WRITE_DATA_WIDTH(32),        // DECIMAL
   .WR_DATA_COUNT_WIDTH(1)       // DECIMAL
)
xpm_fifo_async_inst (
   .almost_empty(almost_empty),   // 1-bit output: Almost Empty :
When asserted, this signal indicates that only one more read can be
```

```
performed
                                  // before the FIFO goes to empty.

   .almost_full(almost_full),     // 1-bit output: Almost Full:
When asserted, this signal indicates that only one more write can
be performed
                                  // before the FIFO is full.

   .data_valid(data_valid),       // 1-bit output: Read Data Valid:
When asserted, this signal indicates that valid data is available
on the
                                  // output bus (dout).

   .dbiterr(dbiterr),             // 1-bit output: Double Bit
Error: Indicates that the ECC decoder detected a double-bit error
and data in the
                                  // FIFO core is corrupted.

   .dout(dout),                   // READ_DATA_WIDTH-bit output:
Read Data: The output data bus is driven when reading the FIFO.
   .empty(empty),                 // 1-bit output: Empty Flag: When
asserted, this signal indicates that the FIFO is empty. Read
requests are
                                  // ignored when the FIFO is
empty, initiating a read while empty is not destructive to the
FIFO.

   .full(full),                   // 1-bit output: Full Flag: When
asserted, this signal indicates that the FIFO is full. Write
requests are
                                  // ignored when the FIFO is full,
initiating a write when the FIFO is full is not destructive to the
contents of
                                  // the FIFO.

   .overflow(overflow),           // 1-bit output: Overflow: This
signal indicates that a write request (wren) during the prior clock
cycle was
                                  // rejected, because the FIFO is
full. Overflowing the FIFO is not destructive to the contents of
the FIFO.
```

```
   .prog_empty(prog_empty),       // 1-bit output: Programmable
Empty: This signal is asserted when the number of words in the FIFO
is less than
                                  // or equal to the programmable
empty threshold value. It is de-asserted when the number of words
in the FIFO
                                  // exceeds the programmable empty
threshold value.

   .prog_full(prog_full),         // 1-bit output: Programmable
Full: This signal is asserted when the number of words in the FIFO
is greater than
                                  // or equal to the programmable
full threshold value. It is de-asserted when the number of words in
the FIFO is
                                  // less than the programmable
full threshold value.

   .rd_data_count(rd_data_count), // RD_DATA_COUNT_WIDTH-bit
output: Read Data Count: This bus indicates the number of words
read from the FIFO.
   .rd_rst_busy(rd_rst_busy),     // 1-bit output: Read Reset Busy:
Active-High indicator that the FIFO read domain is currently in a
reset state.
   .sbiterr(sbiterr),             // 1-bit output: Single Bit
Error: Indicates that the ECC decoder detected and fixed a single-
bit error.
   .underflow(underflow),         // 1-bit output: Underflow:
Indicates that the read request (rd_en) during the previous clock
cycle was rejected
                                  // because the FIFO is empty.
Under flowing the FIFO is not destructive to the FIFO.

   .wr_ack(wr_ack),               // 1-bit output: Write
Acknowledge: This signal indicates that a write request (wr_en)
during the prior clock
                                  // cycle is succeeded.

   .wr_data_count(wr_data_count), // WR_DATA_COUNT_WIDTH-bit
output: Write Data Count: This bus indicates the number of words
written into the
                                  // FIFO.
   .wr_rst_busy(wr_rst_busy),     // 1-bit output: Write Reset
Busy: Active-High indicator that the FIFO write domain is currently
in a reset
                                  // state.

   .din(din),                     // WRITE_DATA_WIDTH-bit input:
Write Data: The input data bus used when writing the FIFO.
   .injectdbiterr(injectdbiterr), // 1-bit input: Double Bit Error
Injection: Injects a double bit error if the ECC feature is used on
block RAMs
                                  // or UltraRAM macros.

   .injectsbiterr(injectsbiterr), // 1-bit input: Single Bit Error
Injection: Injects a single bit error if the ECC feature is used on
block RAMs
                                  // or UltraRAM macros.

   .rd_clk(rd_clk),               // 1-bit input: Read clock: Used
for read operation. rd_clk must be a free running clock.
   .rd_en(rd_en),                 // 1-bit input: Read Enable: If
the FIFO is not empty, asserting this signal causes data (on dout)
to be read
                                  // from the FIFO. Must be held
active-low when rd_rst_busy is active high.

   .rst(rst),                     // 1-bit input: Reset: Must be
synchronous to wr_clk. The clock(s) can be unstable at the time of
applying
                                  // reset, but reset must be
released only after the clock(s) is/are stable.

   .sleep(sleep),                 // 1-bit input: Dynamic power
saving: If sleep is High, the memory/fifo block is in power saving
mode.
   .wr_clk(wr_clk),               // 1-bit input: Write clock: Used
for write operation. wr_clk must be a free running clock.
   .wr_en(wr_en)                  // 1-bit input: Write Enable: If
the FIFO is not full, asserting this signal causes data (on din) to
be written
                                  // to the FIFO. Must be held
active-low when rst or wr_rst_busy is active high.
```

```
);

// End of xpm_fifo_async_inst instantiation
```

Related Information

- *XPM FIFO Testbench File* (xpm-fifo-testbench.zip)