SYNCHRONOUS FIFO

IMPORTANCE NOTICE

This document is a specification and guideline developed for academic study purposes.

It is intended to support student learning and may not cover all corner cases or production-level design considerations.

Contents

List of Figures

List of Tables

1. Overview

1.1 FIFO description

FIFO (First In First Out) is a memory element in which the first data writen is the first data read.

Figure 1 show the block diagram of the synchronous FIFO (SFIFO).

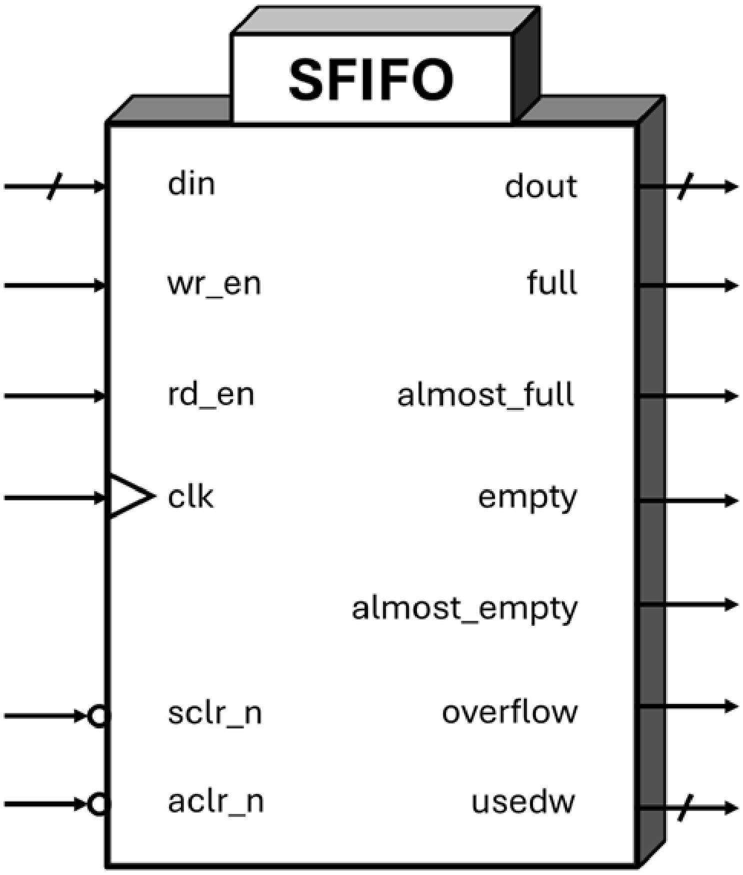

Figure 1: *Synchronous FIFO.*

1.2 FIFO Architecture

The synchronous FIFO is organized as a modular architecture that separates data storage from control logic. This separation improves design clarity, scalability, and ease of verification.

As shown in Figure 2, the FIFO is composed of two main functional blocks: a register-based memory block and a controller block. These blocks operate within a single clock domain and communicate through well-defined control and status signals to ensure correct FIFO operation.

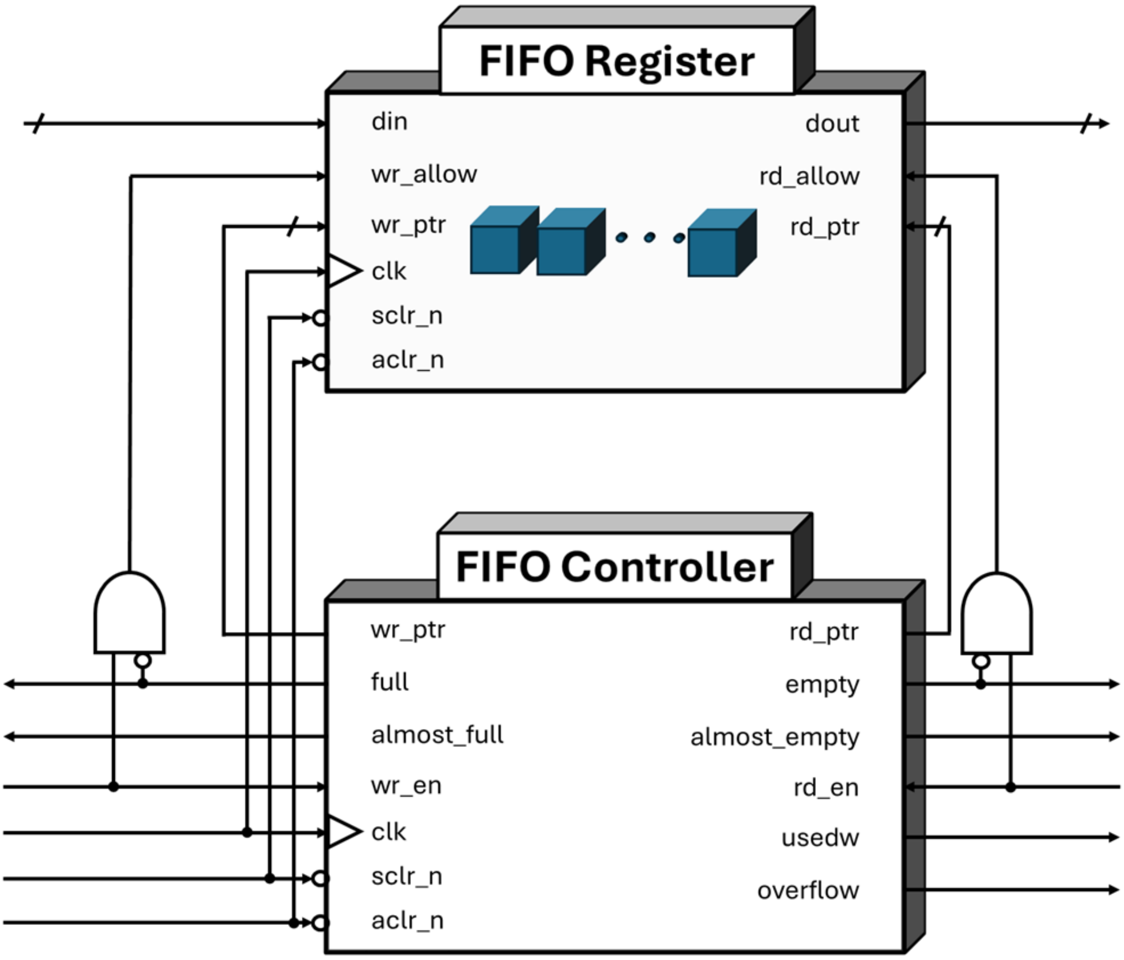

Figure 2: *Block diagram of synchronous FIFO*

The FIFO architecture consists of a register-based memory block and a controller module.

The controller is responsible for managing read/write pointers, status flags, and flow control, while the FIFO register stores the actual data words.

1.3 Key features

Synchronous FIFO with single clock domain

Supports simultaneous read and write operations

Full and empty status indicators

Almost full and almost empty flags

Used word count (usedw) output

Overflow detection

2. Configuration Parameters

Table 1: *FIFO Parameters*

Parameter	Type	Required	Description
DATA_WIDTH	integer	Yes	Width (in bits) of each data word stored in the FIFO memory.
DEPTH	integer	yes	Total number of data words that the FIFO can store.
AF_LEVEL	integer	No	Almost-full threshold. The almost_full flag is asserted when the number of stored words is greater than or equal to (DEPTH − AF_LEVEL).
AE_LEVEL	integer	No	Almost-empty threshold. The almost_empty flag is asserted when the number of stored words is less than or equal to AE_LEVEL.

3. Interface Description

***Table 2**: Input and Output Ports Description*

Port	Type	Required	Description
clk	Input	Yes	Clock signal. All FIFO operations are triggered on the rising edge of clk.
aclr_n sclr_n	Input	No	Asynchronous active-low clear (aclr_n). When asserted (0), FIFO status signals and internal counters are reset immediately, independent of the clock. Synchronous active-low clear (sclr_n). When asserted (0), FIFO status signals and internal counters are reset on the next rising edge of clk. Make sure either aclr_n or sclr_n port is enabled and included in the design to ensure the correct functionality of the FIFO.
wr_en	Input	Yes	Write request signal. When asserted, a write operation is requested. A write occurs only if FIFO is not full, or if a read occurs in the same cycle (read-while-write).
rd_en	Input	Yes	Read request signal. When asserted, a read operation is requested. A read occurs only if FIFO is not empty.
din	Input	Yes	Input data bus. Data to be written into FIFO. Must be stable when wr_en is asserted.
dout	Output	Yes	Output data bus. Shows data read from FIFO. Data is valid when a read operation is performed.
full	Output	No	When asserted, the FIFO is considered full (usedw == DEPTH). Do not perform write request operation when the fifo is full.
empty	Output	No	When asserted the FIFO is considered empty (usedw == 0) . Do not perform read request when the FIFO is empty.
almost_full	Output	No	Asserted when FIFO occupancy reaches a programmable near-full threshold (usedw >= DEPTH - AF_LEVE). Used as early warning.
almost_empty	Output	No	Asserted when FIFO occupancy reaches a programmable near-empty threshold (usedw <= AE_LEVEL). Used as early warning.
usedw	Output	No	FIFO occupancy counter. Indicates the number of valid data words currently stored in the FIFO (0 … DEPTH).
overflow	Output	No	Asserted when a write request occurs while FIFO is full and no read occurs in the same cycle. Indicates data loss.

Notes:

- Read and write enable signals (rd_en, wr_en) are sampled on the rising edge of clk.
- When both rd_en and wr_en are asserted in the same cycle, the FIFO allows simultaneous read and write operations if the FIFO is neither empty nor full.
- Status flags (full, empty, almost_full, almost_empty) reflect the FIFO state corresponding to the current cycle.

4. Reset Behavior

Table 3: *FIFO Reset Behavior*

Reset Signal	**Type**	**Active Level**	**Clock Dependency**	**Description**
aclr_n	Asynchronous reset	Active Low	Independent of clock	Immediately resets FIFO internal logic when asserted low.
sclr_n	Synchronous reset	Active Low	Applied on active clock edge	Resets FIFO internal logic on the rising edge of the clock when asserted low.

Table 4: *FIFO State After Reset*

Item	**Value After Reset**
Read pointer	Cleared to 0
Write pointer	Cleared to 0
Used word count (usedw)	0
FIFO empty flag	Asserted
FIFO full flag	Deasserted
Almost empty flag	Asserted
Almost full flag	Deasserted
Output data (dout)	Cleared (if implemented)

5. Functional Operation

***Table 5**: Write operation*

Input				Output	Sate
full	empty	wr_en	rd_en	wr_allow	
0	x	1	x	1	Write
1	x	x	0	0	Not write

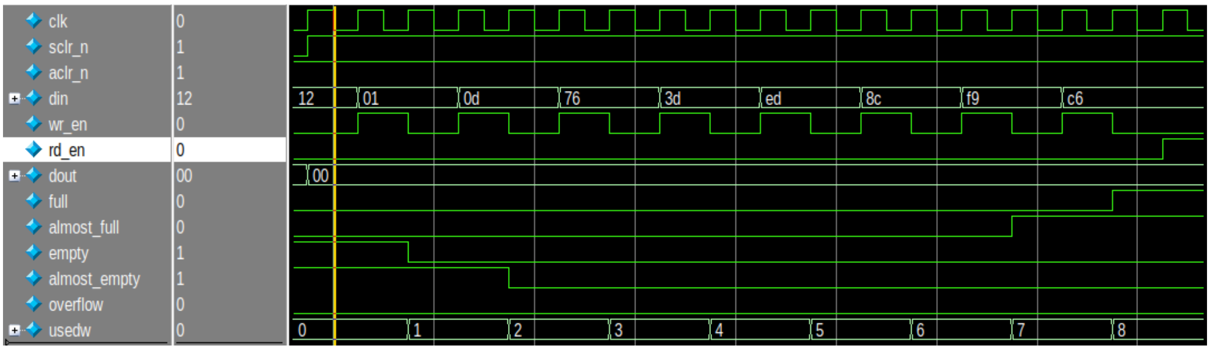

***Figure 3:** Write waveform*

***Table 6:** Read operationa*

Input				Output	Sate
full	empty	wr_en	rd_en	rd_allow	
x	0	x	1	1	Read
x	1	x	x	0	Not read

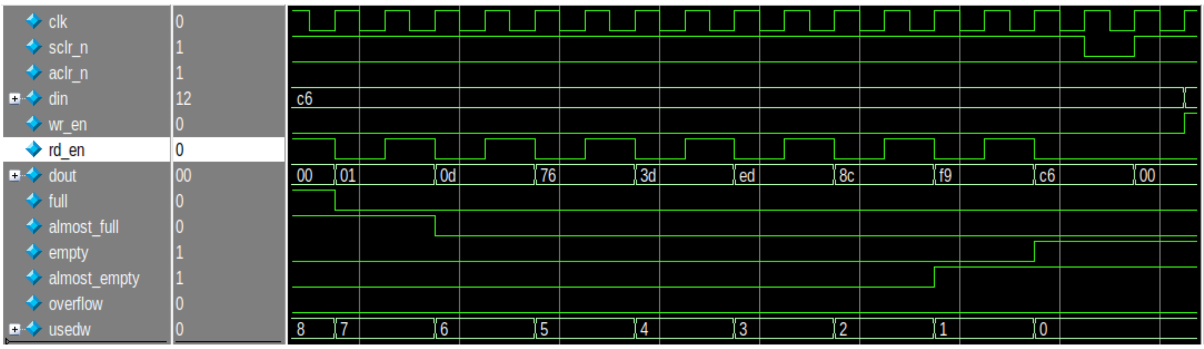

***Figure 4:** Read waveform*

Table 7: *Simultaneous Read/Write Operation*

full	empty	wr_en	rd_en	Behavior
0	0	1	1	Read and write occur simultaneously
1	x	x	1	Read allowed, write blocked
x	1	1	x	Write allowed, read blocked

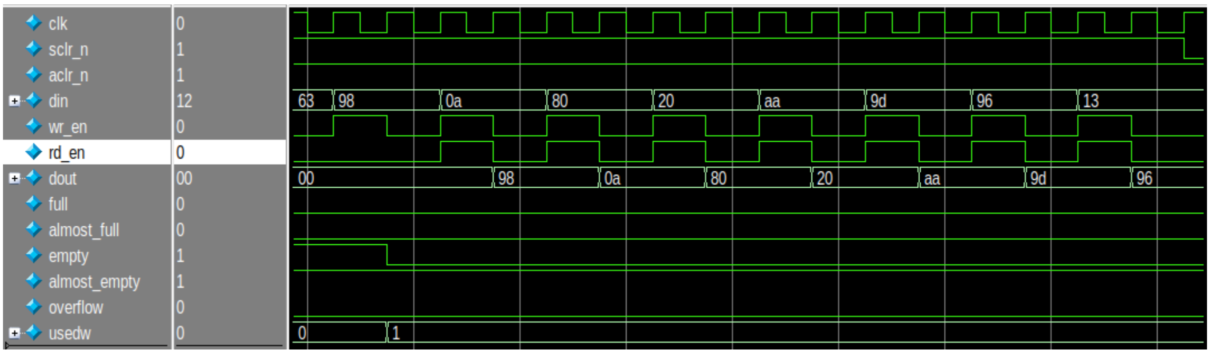

Figure 5: *Simultaneous Read/Write waveform*

Notes:

- The usedw counter is incremented on a successful write operation and decremented on a successful read operation.
- During simultaneous read and write operations, the usedw value remains unchanged.
- FIFO status flags are updated synchronously with the clock.