Synchronous and Asynchronous FIFO Designs

XAPP 051 September 17,1996 (Version 2.0)

Application Note by Peter Alfke

Summary

This application note describes RAM-based FIFO designs using the dual-port RAM in XC4000-Series devices. Synchronous designs with a common read/write clock are described, as well as asynchronous designs with independent read and write clocks. Emphasis is on the fast, efficient and reliable generation of the handshake signals FULL and EMPTY, which determine design performance.

Xilinx Family

XC4000E, XC4000L, XC4000EX, XC4000XL

Demonstrates

FIFO design techniques

Introduction

Many XC4000-Series designs use the distributed RAM feature to implement First-In-First-Out (FIFO) elastic buffers to form a bridge between subsystems with different clock rates and access requirements.

The non-synchronous nature of the single-port RAM in XC4000 confronts the designer with several challenges. Addresses must be multiplexed, independent read and write clocks must be synchronized, and access requests must be arbitrated.

The improved synchronous dual-port RAM (Select-RAM™) in the enhanced XC4000 Series devices, the XC4000E and XC4000EX, solves most of these problems. Since the basic RAM in each CLB has independent write and read addresses, there is no need to multiplex addresses and arbitrate their selection. The synchronous write mechanism simplifies write timing and contributes to much faster operation.

The FIFO design effort can now be concentrated on achieving high throughput and low cost, and on solving the fundamental timing problems created by asynchronous read and write clocks.

This application note describes several design examples: three different FIFO depths, each with either a common clock (synchronous operation) or with two unsynchronized clocks (asynchronous operation).

The first design is a 16 x 16 FIFO where the depth of the basic CLB-RAM is sufficient. This leads to a very fast and efficient implementation that can run at, or close to, the maximum write speed (70 MHz in an XC4000E-3 device), even for simultaneous read and write operations.

Table 1: FIFO Design Performance, XC4000E-3 Device

	No. of CLBs	Simultaneous Read and Write	
		Synch. Speed	Asynch. Speed
16 x 16 FIFO	23	65 MHz	50 MHz
32 x 8 FIFO	28	50 MHz	40 MHz
64 x 8 FIFO	48	50 MHz	40 MHz

The second design example is a 32 x 8 FIFO (32 deep, 8 bits wide) that requires input and output data multiplexing between two RAM banks. The address counters are longer than for the 16-deep FIFO, and the control logic generating FULL and EMPTY is more complex, with one additional layer of logic. This slows down operation, and 40 MHz simultaneous asynchronous read and write may be the maximum performance in an XC4000E-3 device.

The third design example is a 64 x 8 FIFO (64 deep, 8 bits wide) that requires input and output data multiplexing between four RAM banks.

Free-Running Read and Write Clocks

These designs assume free-running clocks, activated by their respective enable signals. Without a free-running Read Clock, the asynchronous designs would lock up with an active EMPTY(STRETCHED) output, which can only be terminated by a High level on Read Clock. If this clock is not free-running, the EMPTY(STRETCHED) output stops the external decision-making logic from making Read Clock go High. EMPTY(STRETCHED), therefore, stays active, even after data has been written into the FIFO. FULL(STRETCHED) would behave similarly without a free-running Write Clock.

Free-running clocks, activated by their respective enable signals, avoid these problems.

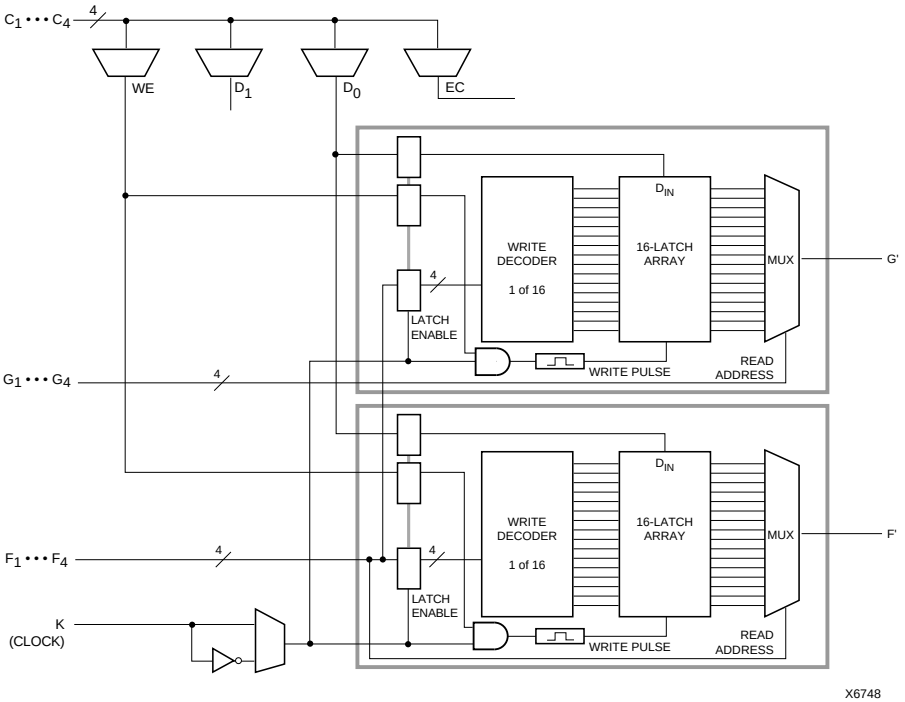


Figure 1: XC4000-Series Edge-Triggered RAM

16 x 16 FIFO with Common Clock

This example implements a 16 x 16 FIFO with a common read/write clock and individual read and write clock enables. This is the simplest and fastest design, since it avoids the more challenging issues of asynchronous clocking.

Figure 1, taken from the XC4000 Series data sheet, shows the basic dual-port 16 x 1 RAM that can be implemented in each CLB.

Writing is synchronous. Write Enable, Write Address, and Write Data must meet the documented set-up time with respect to the Write Clock. Reading is asynchronous, controlled only by the Read Address. The two unused CLB flip-flops have uncommitted data inputs and Q outputs, but share the clock signal (although not the clock polarity) with the write port.

Figure 2 shows the basic block diagram of the 16 x 16 FIFO.

In the synchronous version of this design, read and write clock are identical. The 4-bit read counter and the 4-bit write counter, shown in Figure 3, are each implemented as two cascaded 2-bit Grey or Johnson counters. In a fully synchronous design, this choice is not mandatory, but it has advantages in the non-synchronous implementation. It is, however, mandatory that the upper two bits always stay constant for four consecutive counts. Four-bit Linear-Feedback-Shift-Register (LFSR) counters can, therefore, *not* be used, as will soon become apparent.

The two 4-bit counters address the RAM in the conventional way. Seen as a “black box”, the FIFO behaves like an elastic shift register: Input Data is accepted by the Write Clock when Write Enable was High during the set-up time before the active Write Clock edge. Output Data is always available at the output port, and is substituted by the next Data, after the Read Clock was enabled by a Read Enable High during the Read Clock set-up time.

FULL and EMPTY outputs must be interpreted by external logic to prevent a Write operation during FULL, or a Read operation during EMPTY.

Most of the design effort is spent on the control logic, shown in Figure 4, that detects the two abnormal conditions, FULL and EMPTY.

When the FIFO contains 16 words that have not yet been read, the FULL flag must be activated, and further write operations must be avoided.

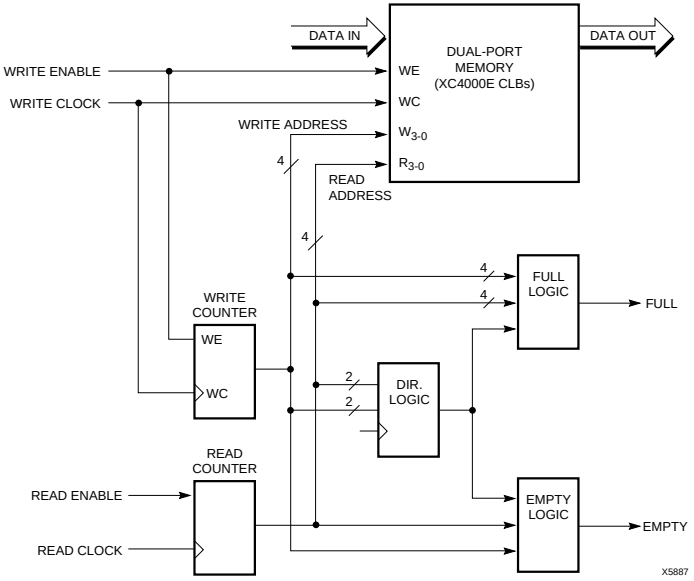
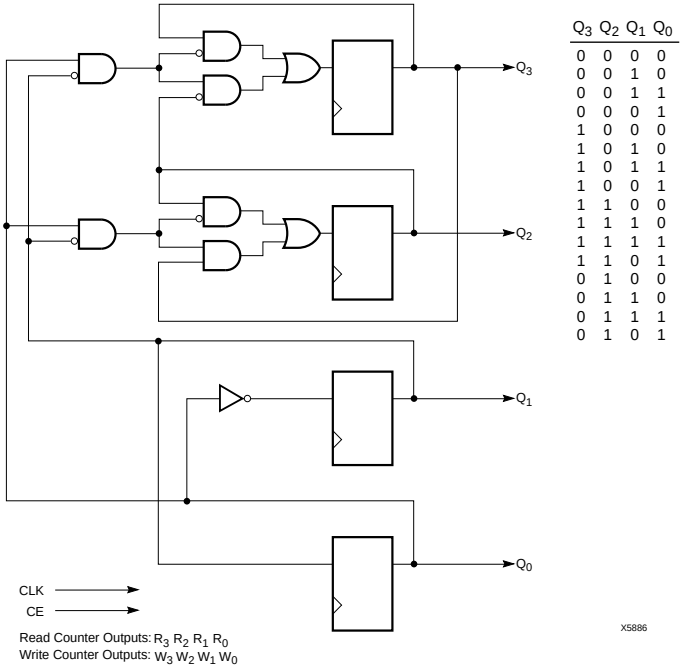


Figure 2: 16 x 16 FIFO Block Diagram



Q_3	Q_2	Q_1	Q_0
0	0	0	0
0	0	1	0
0	0	1	1
0	0	0	1
1	0	0	0
1	0	1	0
1	0	1	1
1	0	0	1
1	1	0	0
1	1	1	0
1	1	1	1
1	1	0	1
0	1	0	0
0	1	1	0
0	1	1	1
0	1	0	1

Figure 3: 16-Deep FIFO, Read Counter or Write Counter

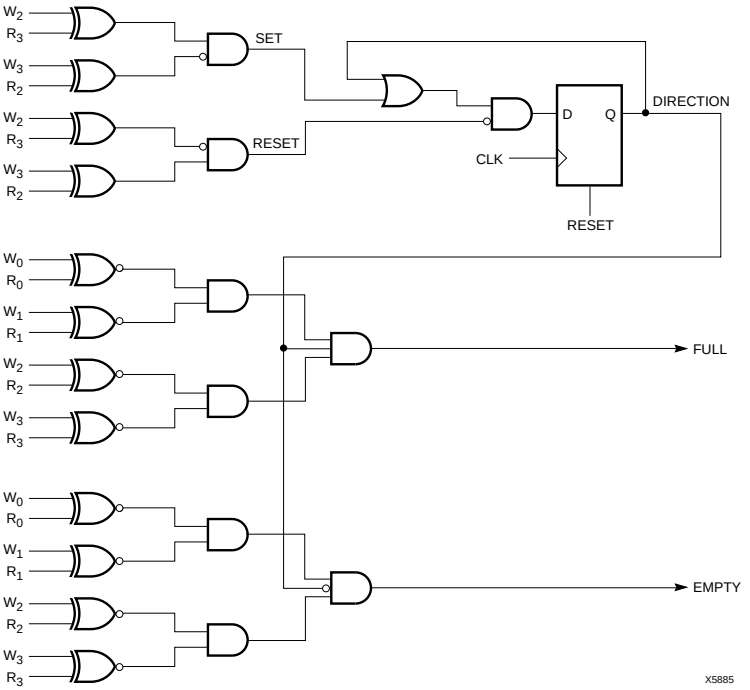


Figure 4: 16-Deep FIFO, Synchronous Control

When all words written into the FIFO have been read, the EMPTY flag must be activated, and further read operations must be avoided.

Unfortunately, the easily decoded signal for these two abnormal conditions is the same: read address is identical with write address. An additional signal must be created that distinguishes between the two very different conditions of FULL and EMPTY.

For this purpose, an auxiliary signal called DIRECTION is created to indicate whether the Write counter is about to catch up with the Read counter, or whether the Read counter is about to catch up with the Write counter. The two most significant bits of both counters are compared, since they indicate in which quadrant of the 16-position circular address space the present address resides.

These two most significant bits of both address counters together are used to address two 4-input look-up tables in parallel. The look-up tables (LUTs) decode the relative quadrant position of the two counters.

The 4-bit LUT address describes one of 16 possible conditions:

- Four addresses describe the situation where the write counter is in the quadrant immediately behind the read counter. This is decoded as a “possibly going full” condition, and sets the DIRECTION latch or flip-flop.
- Another four addresses describe the situation where the write counter is in the quadrant immediately ahead of the read counter. This is decoded as a “possibly going empty” condition, and it resets the DIRECTION latch or flip-flop.
- Four other addresses indicate that the two counters are in the same quadrant, and another four addresses indicate that the two counters are in opposite quadrants. These eight addresses provide no useful information about the relative address position, and thus do not affect DIRECTION.

Note that DIRECTION must start in the reset state when the FIFO is initiated with both counters at zero.

DIRECTION is thus established well before the actual FULL or EMPTY condition can occur. There will be at least four, and usually many more, consecutive set or reset inputs to the DIRECTION latch or flip-flop before it is being used to discriminate between FULL or EMPTY.

FULL goes active as a result of the write clock edge that writes data into the last available location. FULL goes inactive as a result of the first read clock that reads one word out of the previously full FIFO.

EMPTY goes active as a result of the read clock edge that reads the last available data from the FIFO. EMPTY goes inactive as a result of the first write clock that writes one word into the previously empty FIFO.

In a synchronous design, FULL and EMPTY are synchronous control signals, to be used appropriately by the logic external to the FIFO.

16 x 16 FIFO with Independent Clocks

This example adapts the 16 x 16 FIFO for use with independent read and write clocks.

An asynchronous design with separate and unrelated read and write clocks poses additional problems. The RAM array, the counters, and part of the control logic are unaffected, but the DIRECTION, FULL and EMPTY signals require additional attention, as shown in Figure 5. There is no common clock; DIRECTION therefore uses a latch, implemented as a combinatorial circuit in a separate CLB. Since the counter bits that determine DIRECTION are Grey-coded, with only one bit changing per transition, the decoding is guaranteed to be glitch-free. Initialization reset is provided by a flip-flop that is set by the first write operation.

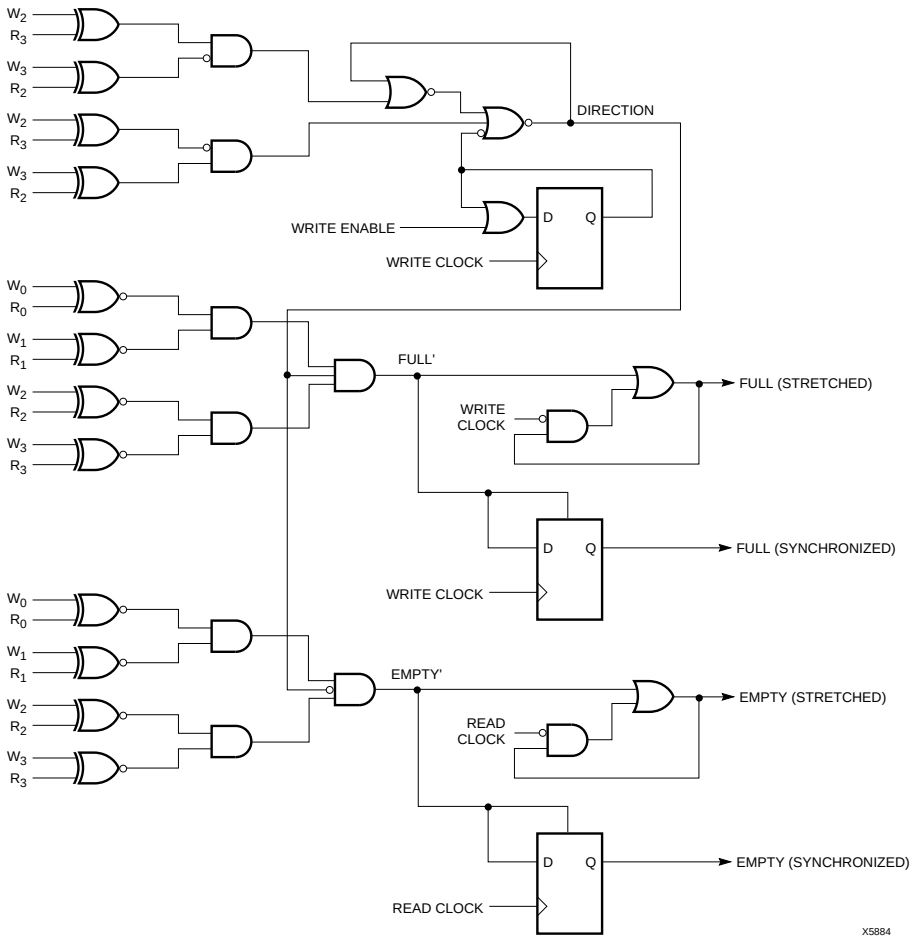


Figure 5: 16-Deep FIFO, Asynchronous Control

FULL goes active as a synchronous response to the write clock, but FULL goes inactive as a result of the read clock, asynchronous to the write clock. Since FULL is used only by the write control logic, there is no synchronization problem when FULL goes active, but there is when FULL goes inactive. The easiest solution is to stretch the FULL signal such that it cannot go inactive during the half-period of the write clock immediately preceding the active clock edge.

Figure 5 shows this circuit, assuming a rising clock edge. The FULL output can go inactive only while the write clock is High, which gives the write logic enough set-up time to the following rising clock edge.

There is still the possibility of metastable confusion if the full condition goes inactive in a very narrow timing window when the latch is about to latch up, i.e.-right after the falling edge of Write Clock. In most cases, this metastable disturbance will have settled well before the next rising clock edge, but if the user is concerned about this low-probability risk, FULL can be stretched by a complete write clock period, which will reduce the likelihood of metastable failure to an insignificant level.

Figure 5 also shows this alternate design, using a flip-flop clocked by the Write Clock, generating a synchronous falling edge of the FULL signal.

EMPTY is generated by a similar circuit that prevents EMPTY from going inactive while the read clock is Low, or synchronizes it with the other edge of the READ clock.

FULL and EMPTY flags are delayed by an extra level of logic, compared to the fully synchronous design. The stretcher circuit adds 2 ns, the alternate flip-flop adds 4 ns, thus reducing peak performance to 58 or 50 MHz, respectively.

As mentioned on the first page of this application note, Read Clock must be free-running, activated by Read Enable.

32 x 8 FIFO

This example implements a 32 x 8 FIFO with independent read and write clocks.

Since each CLB can only implement a 16 x 1 dual-port RAM, the 32-deep FIFO uses two memory banks. Longlines distribute data to and from the RAMs. For the write port, WE gates a 1-of-2 decoder. For the read port, TBUFs are used to multiplex the data onto Longlines, as shown in Figure 6.

The 5-bit read and write address counters each consist of a 3-bit (8 Linear-Feedback-Shift-Register (LFSR) counter followed by a 2-bit Grey or Johnson counter. (See Figure 7.) The latter drives the 1-of-2 decoders selecting between memory banks. The DIRECTION detector decodes the quadrant information and generates set and reset signals for the DIRECTION flip-flop or latch. When both counters are identical, either a FULL or EMPTY output flag is generated, depending on the state of DIRECTION.

Asynchronous control logic for a 32-deep FIFO is shown in Figure 8.

For a more detailed description, see the earlier 16 x 16 FIFO section.

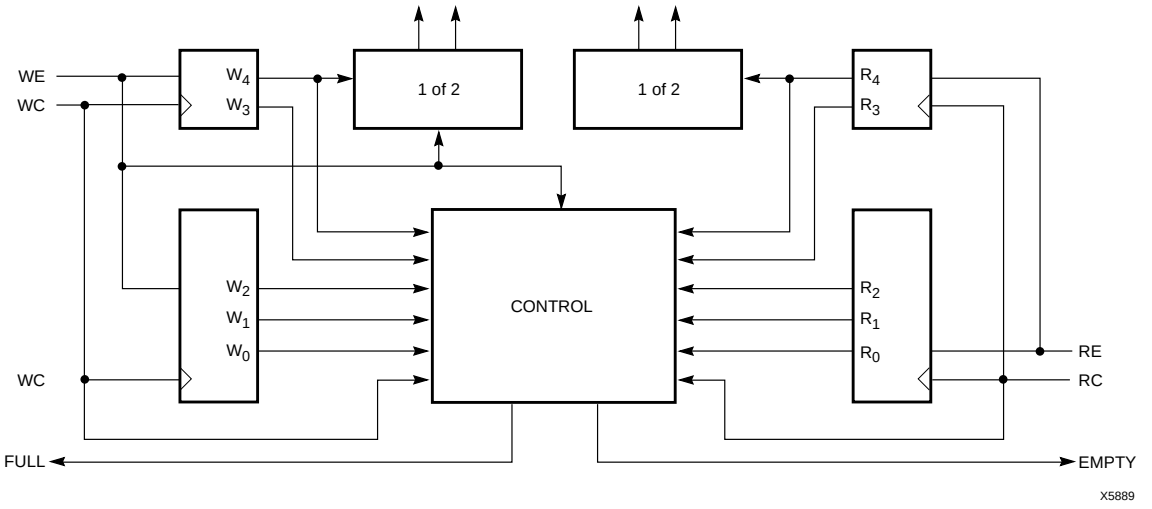


Figure 6: 32 x 8 FIFO Block Diagram

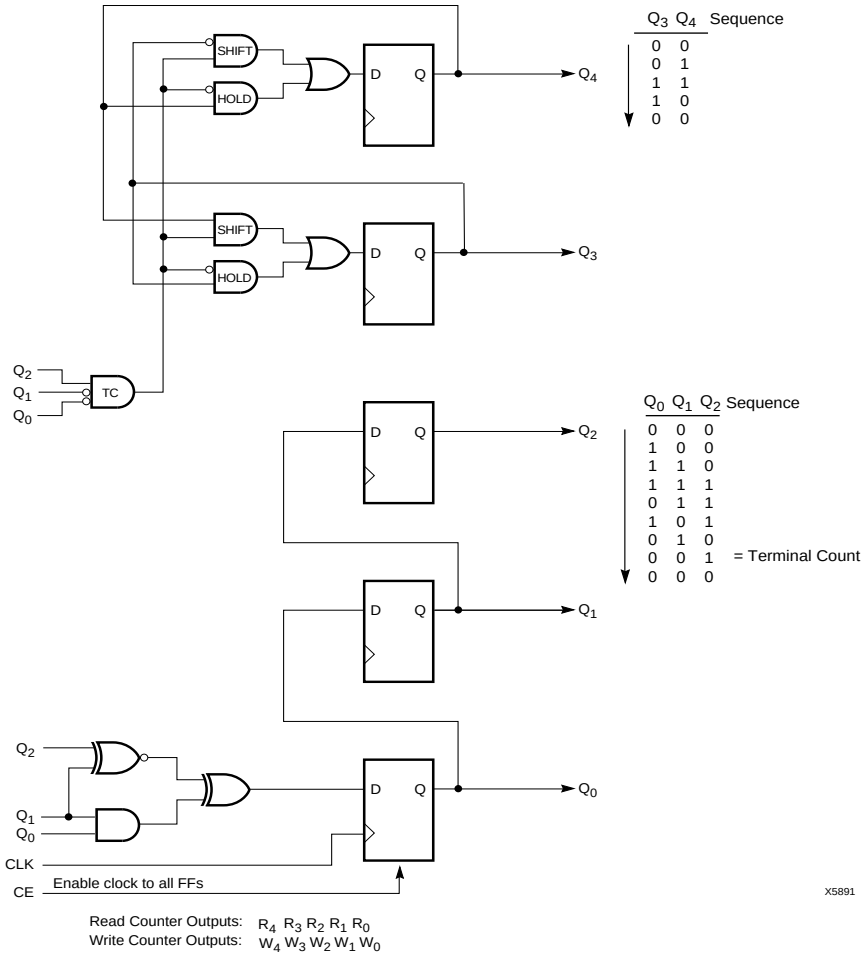


Figure 7: 32-Deep FIFO, Read Counter or Write Counter

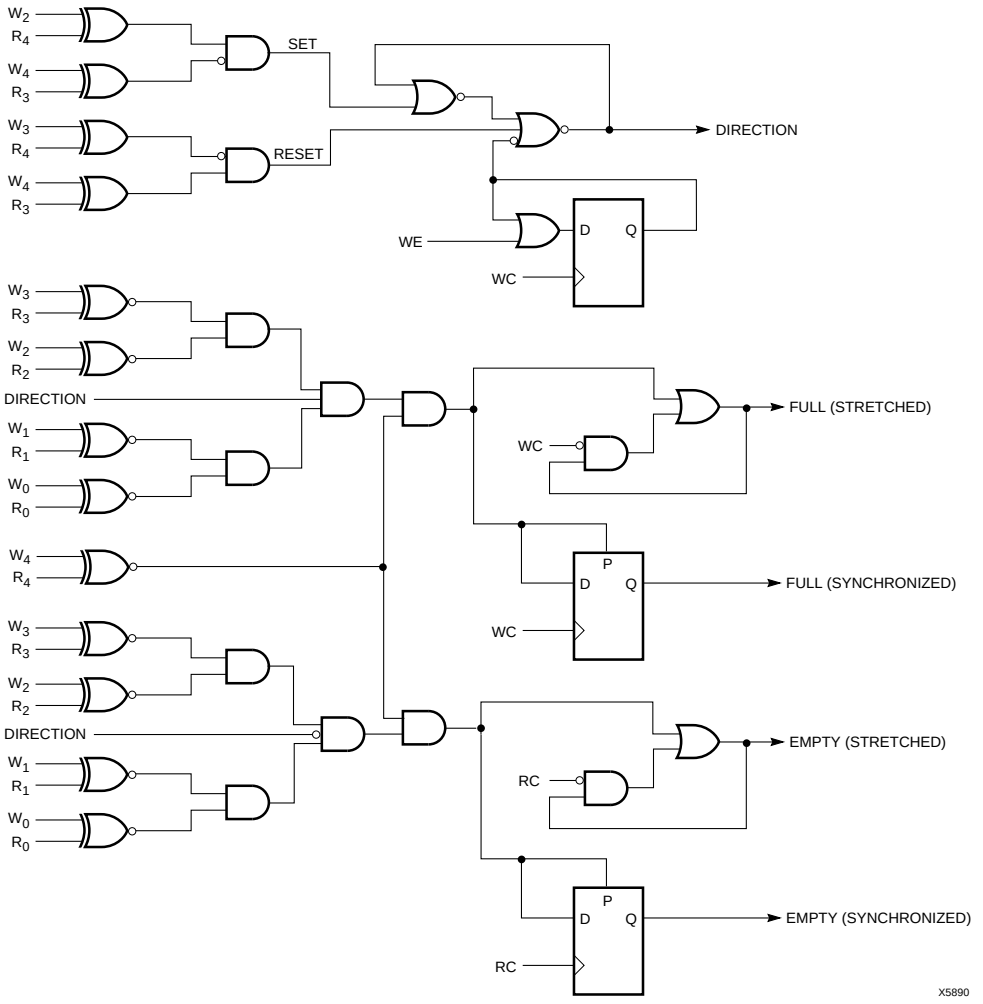


Figure 8: 32-Deep FIFO, Asynchronous Control (5 or 6 CLBs)

64 x 8 FIFO

This example implements a 64 x 8 FIFO with independent read and write clocks.

Since each CLB can only implement a 16 x 1 dual-port RAM, the 64-deep FIFO uses four memory banks. Longlines distribute data to and from the RAMs. For the write port, WE gates a 1-of-4 decoder. For the read port, TBUFs are used to multiplex the data onto Longlines, as shown in Figure 9.

The 6-bit read and write address counters each consist of a 4-bit ÷16 Linear-Feedback-Shift-Register (LFSR) counter followed by a 2-bit Grey or Johnson counter. (See Figure 10.) The latter drives the 1-of-4 decoders selecting between memory banks. The DIRECTION detector decodes the quadrant information and generates set and reset signals for the DIRECTION flip-flop or latch. When both counters are identical, either a FULL or EMPTY output flag is generated, depending on the state of DIRECTION.

Asynchronous control logic for a 64-deep FIFO is shown in Figure 11.

For a more detailed description, see the earlier 16 x 16 FIFO section.

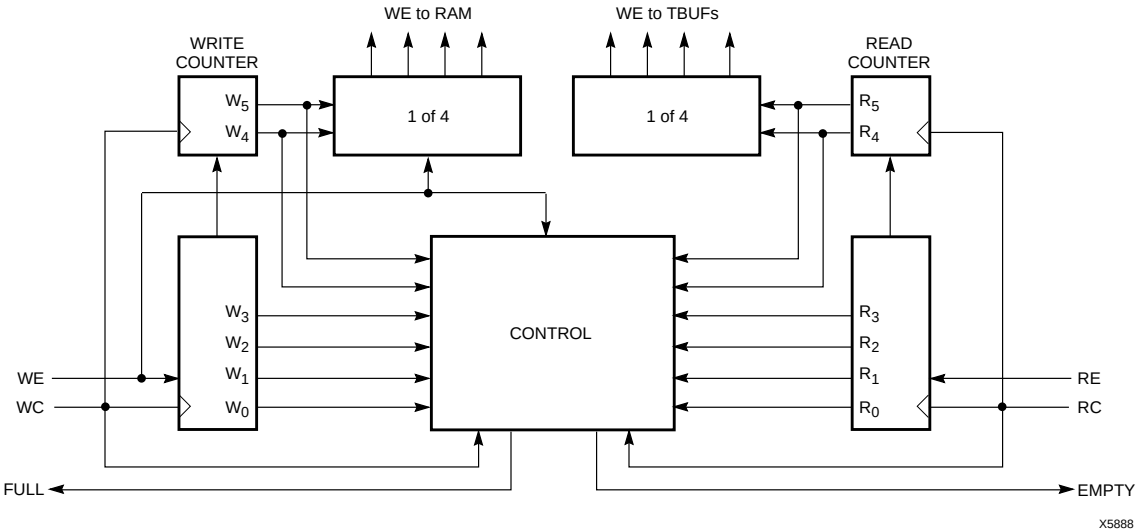
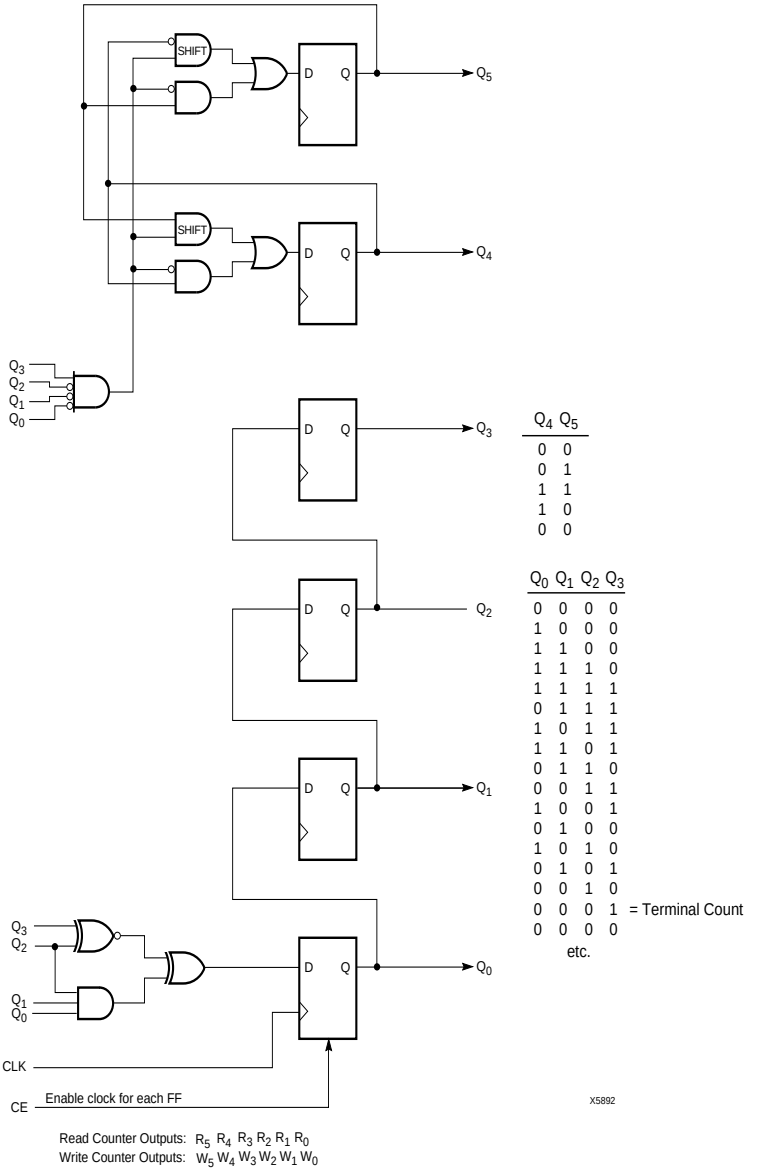


Figure 9: 64 x 8 FIFO Block Diagram



Q_4	Q_5
0	0
0	1
1	1
1	0
0	0

Q_0	Q_1	Q_2	Q_3	
0	0	0	0	
1	0	0	0	
1	1	0	0	
1	1	1	0	
1	1	1	1	
0	1	1	1	
1	0	1	1	
1	1	0	1	
0	1	1	0	
0	0	1	1	
1	0	0	1	
0	1	0	0	
1	0	1	0	
0	1	0	1	
0	0	1	0	
0	0	0	1	= Terminal Count
0	0	0	0	

etc.

Figure 10: 64-Deep FIFO, Read or Write Counter

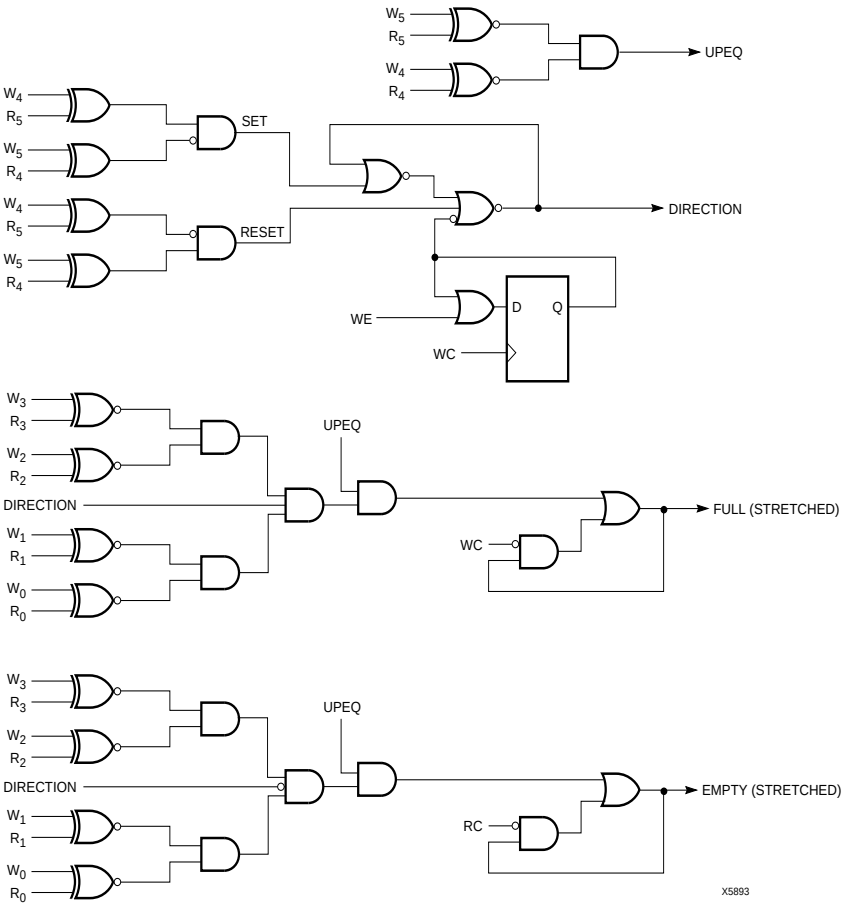


Figure 11: 64-Deep FIFO, Asynchronous Control

Conclusion

The synchronous dual-port Select-RAM mode available in the XC4000 Series makes it possible to incorporate fast and efficient FIFO designs running either with a common clock or with two asynchronous clocks. Four such designs are described in this application note:

- 16-deep FIFO with common clock
- 16-deep FIFO with asynchronous clocks
- 32-deep FIFO with asynchronous clocks
- 64-deep FIFO with asynchronous clocks.

The larger FIFOs with common clocks are not described, but their design is an obvious extension of the 16-deep synchronous design.

Limitations and Restrictions

WARNING: THESE ARE UNTESTED DESIGNS.

Xilinx, Inc. does not make any representation or warranty regarding these designs or any item based on these designs. Xilinx disclaims all express and implied warranties, including but not limited to the implied fitness of these designs for a particular purpose and freedom from infringement. Without limiting the generality of the foregoing, Xilinx does not make any warranty of any kind that any item developed based on these designs, or any portion of them, will not infringe any copyright, patent, trade secret or other intellectual property right of any person or entity in any country. It is the responsibility of the user to seek licenses for such intellectual property rights where applicable. Xilinx shall not be liable for any damages arising out of or in connection with the use of these designs including liability for lost profit, business interruption, or any other damages whatsoever.

XILINX® The Programmable Logic Company℠

Headquarters

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
U.S.A.

Tel: 1 (800) 255-7778
or 1 (408) 559-7778
Fax: 1 (800) 559-7114

Net: hotline@xilinx.com
Web: http://www.xilinx.com

North America

Irvine, California
(714) 727-0780

Englewood, Colorado
(303)220-7541

Sunnyvale, California
(408) 245-9850

Schaumburg, Illinois
(847) 605-1972

Nashua, New Hampshire
(603) 891-1098

Raleigh, North Carolina
(919) 846-3922

West Chester, Pennsylvania
(610) 430-3300

Dallas, Texas
(214) 960-1043

Europe

Xilinx Sarl
Jouy en Josas, France
Tel: (33) 1-34-63-01-01
Net: frhelp@xilinx.com

Xilinx GmbH
Aschheim, Germany
Tel: (49) 89-99-1549-01
Net: dlhelp@xilinx.com

Xilinx, Ltd.
Byfleet, United Kingdom
Tel: (44) 1-932-349401
Net: ukhelp@xilinx.com

Japan

Xilinx, K.K.
Tokyo, Japan
Tel: (03) 3297-9191

Asia Pacific

Xilinx Asia Pacific
Hong Kong
Tel: (852) 2424-5200
Net: hongkong@xilinx.com