Implementing a FIFO Buffer with a Dual-Port Memory and a CPLD or FPGA

AN4033

Introduction

Dual-Port memories enable communication and sharing of data between different systems. The flexibility of a Dual-Port allows it to function as a First-In First-Out (FIFO) buffer using the method described below, provided that external logic is available. This logic can be designed into an FPGA or CPLD that is already present within the system. The FIFO can be simple with few features to reduce the gate count, or a feature-rich design. Examples include multiple queues, partitioning the DP to be part FIFO and part DP memory, and data processing.

This application note discusses how an FPGA or CPLD and Dual-Port memory together can act as a FIFO. This combination is capable of incorporating many features that are not included in current standalone FIFOs, as well as the standard FIFO features. The designer can choose which of these features to implement to customize the FIFO for the system requirements. This method can be expanded for multiple queues as well, and an example implementation is shown in Figure 1.

Large FIFO densities can be achieved with this method with the Cypress Dual-Ports mentioned below in Table 1. For designs limited by available I/O pins, Dual-Ports with burst counters can operate without their address inputs connected, as well as run at speeds up to 167 MHz.

Throughout this application note the FPGA or CPLD will be referred to as the Logic Device, or LD. The Dual-Port memory will be referred to as the DP.

Figure 1. An FPGA or CPLD and one DP RAM acting to implement multiple FIFO queues. The DP is segmented to buffer data for FIFOs of varying size and data direction. It can also use a segment for normal DP operation, as shown.

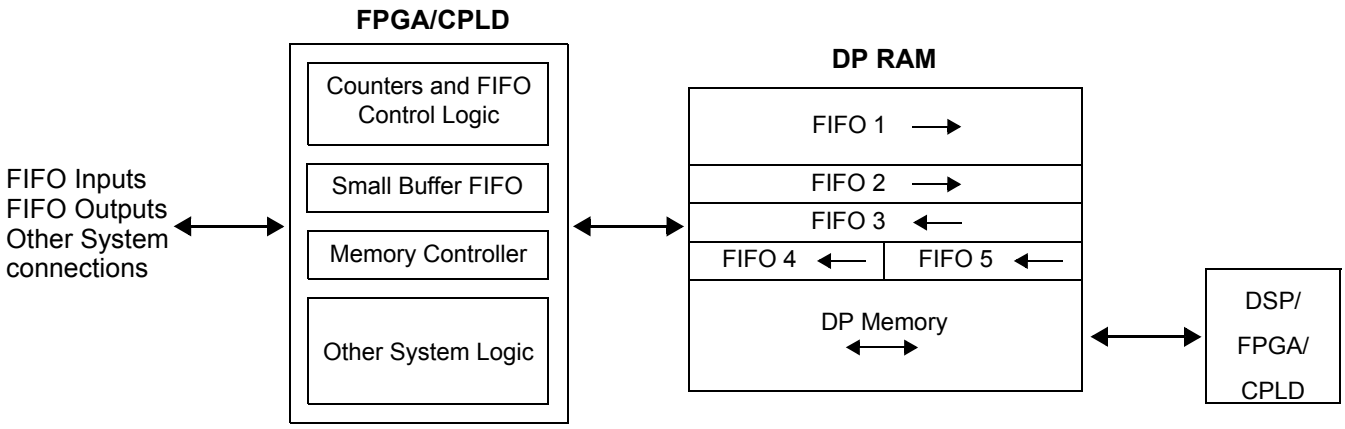


Table 1. Cypress High Density Dual-Port Memories

Part Number	Density (Mbit)	Configuration	Max. Speed (MHz)	Burst Counter
CY7C0852V	4	128K x 36	167	X
CY7C0832V	4	256K x 18	167	X
CY7C0853V	9	256K x 36	133	
CYD18S72V	18	256K x 72	133	
Cypress offers a full Dual-Port Memory portfolio. This is only a sample.				

Basic Implementation

The main idea behind this solution is to use a small buffer FIFO within the LD to absorb the latency intrinsic to the DP. As shown in Figure 2, input data (DIN) is written into both the buffer FIFO and the DP on each rising edge of the write clock, WCLK. This continues until the buffer FIFO is full, at which point input data is only written into the DP. On the rising edge of the read clock, RCLK, data is read out of the buffer FIFO to DOUT. If the buffer FIFO has become full at any point, its input side becomes controlled by RCLK and incoming data is fed by the output data port of the DP. The DP continues to feed data to the buffer FIFO synchronous to RCLK until the buffer becomes full or the DP read address catches up to the DP write address, which can happen when the FIFO is close to being empty. A summary of this behavior is shown in Table 2. Control of the input to the buffer FIFO does not return to WCLK and DIN until the buffer FIFO has emptied, which means the overall FIFO is empty as well. On Master or Partial Reset, control of the input to the buffer FIFO returns to WCLK and DIN as well.

The buffer FIFO within the LD is a standard component of vendors' library of parameterized modules (LPM). To instantiate a FIFO in Cypress' Delta39K™ series of CPLDs, the CY_FIFO component is called within the Verilog or VHDL code and is supported by Cypress' *Warp* software. See the Cypress application note entitled *Using FIFOs in Delta39K™ CPLDs* for more information. FPGAs from Xilinx and CPLDs from Altera have FIFO components as well. There are many resources on Xilinx's web site, such as the application notes *XAPP258: FIFOs Using Virtex-II Block RAM* and *XAPP131: 170 MHz FIFOs Using the Virtex Block SelectRAM+ Feature* and the LogiCORE Asynchronous FIFO v5.1. Altera CPLDs use the Dual-Clock FIFO Megafunction. More information can be found in the *Single- and Dual-Clock FIFO Megafunctions* user guide on Altera's web site.

The FIFOs mentioned above produce Full and Empty flags, which are essential to the operation of the LD's buffer FIFO. The buffer FIFO's Full flag determines when DIN will stop being written to the buffer and is also used to control incoming data from the output port of the DP. The Empty flag is used as the Empty flag for the overall FIFO and controls which data enters the buffer FIFO.

An example design shown in Figure 3 illustrates how this method is used to create a FIFO for video data buffering purposes, while allowing the DP to connect and transfer data between the FPGA and a DSP. The top half of the figure shows the logical data flow of the design, while the bottom half shows the actual implementation. The connections to the DP are time division multiplexed in order to prevent contention

Figure 2. FIFO Made from FPGA/CPLD and DP

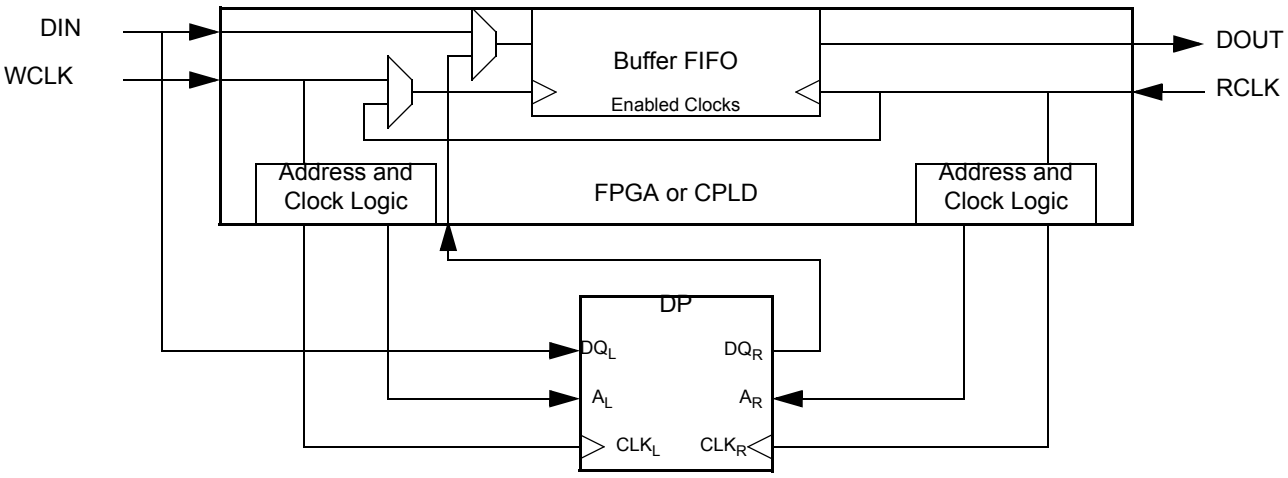


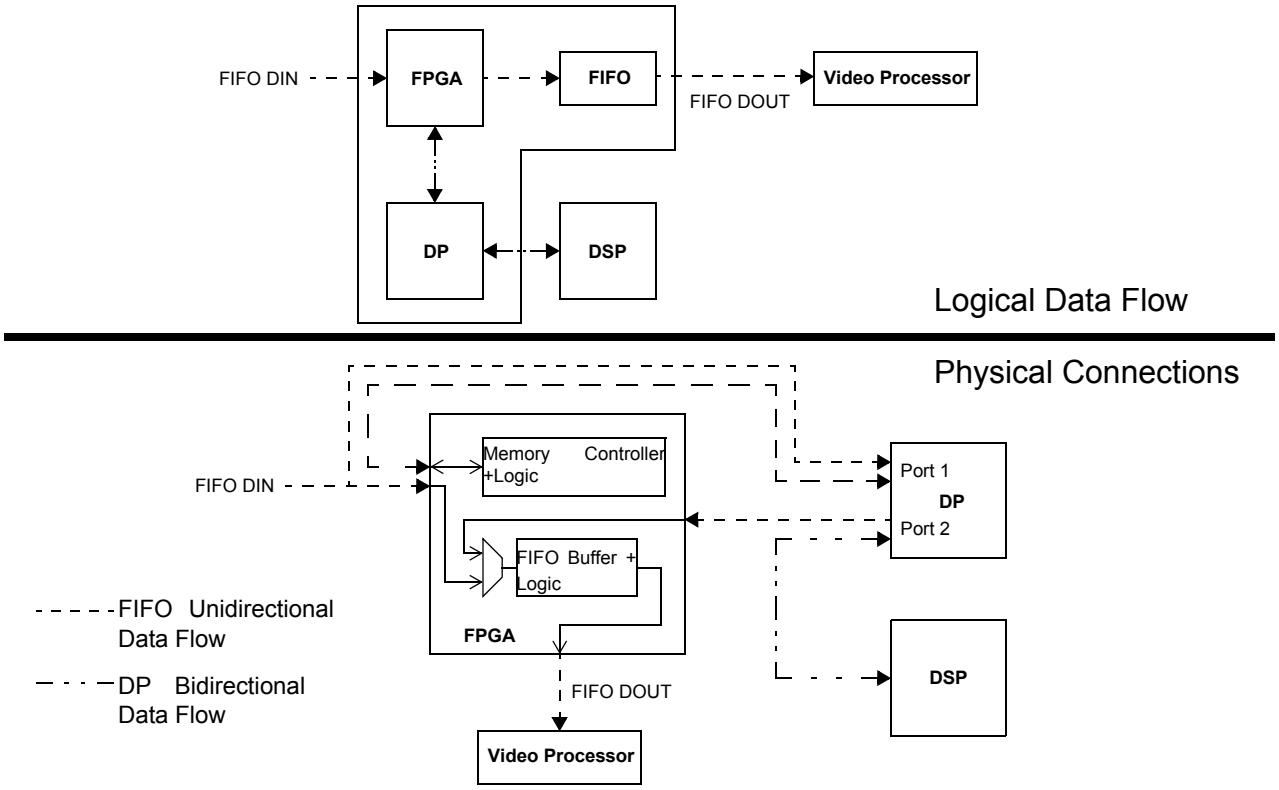
Table 2. Description of Interaction Between Buffer FIFO and DP

Clock Transition	Buffer FIFO Full in Past?	Data Source	Data Destination
WCLK LOW-to-HIGH	No	DIN	Buffer FIFO, DQ_L of DP
WCLK LOW-to-HIGH	Yes	DIN	DQ_L of DP
RCLK LOW-to-HIGH[1]	No	–	–
RCLK LOW-to-HIGH[1]	Yes	DQ_R of DP	Buffer FIFO

Note
1. On RCLK LOW-to-HIGH transition, data is always taken from the Buffer FIFO and presented at DOUT, unless the FIFO is empty.

Figure 3. The FPGA and DP act as a FIFO for the video data stream. The DP transfers and stores data between the FPGA and DSP as well

The FPGA and DP implement the FIFO. Three separate components are replaced by two.



Status Flags

All FIFOs have Full ($\overline{FF}$) and Empty ($\overline{EF}$) flags, as well as programmable Almost Full ($\overline{AF}$) and Almost Empty ($\overline{AE}$) flags. The Half Full ($\overline{HF}$) flag is usually optional. These flags are generated by a combination of counters that keep track of the number of writes and reads and logic that compares these counters and prevents metastability. The counters for the $\overline{AF}$ and $\overline{AE}$ flags are preloaded with a value less than half the size of the FIFO and a value greater than half the size of the FIFO respectively.

Here is an example to illustrate these concepts. Given a FIFO of depth 1,024, the read and write counters will be 11 bits wide. The MSB is used to determine whether the FIFO is full or empty when the 10 LSBs are equal. Assuming that both counters were reset to zero, if the ten LSBs are equal and the MSBs differ, then this indicates that 1,024 more writes than reads have occurred and the FIFO is full. If the MSBs are the same then the read counter has caught up to the write counter, and the FIFO is empty. There are two ways to perform the comparison and flag generation. The first is to subtract the 10 LSBs of one counter from the other, bitwise OR the result and use this result along with the XOR/XNOR of the MSBs to generate the empty and full flags. The second way is to directly decode the flags by XNORing the 10 LSBs, then bitwise AND the result and the XOR/XNOR of the MSBs.

The latter method should lead to faster results, since logic gates are implemented through lookup tables (LUTs) or programmable logic arrays (PLAs) while the subtract operation has carry chain delays that grow with the width of the counters. Table 3 shows examples where the write counter and read counter are equal and not equal. When equal, the XNORs produce all ones, and the FIFO is empty. When unequal, not all the XNOR outputs are ones, so the FIFO is not empty. Table 4 shows similar examples for the full flag. The difference is that the MSBs are XORed together, to produce a '1' when they are not equal.

Operation of the programmable $\overline{AF}$ and $\overline{AE}$ flags is best shown through an example as well. For instance, if the $\overline{AE}$ flag is to be active when the FIFO has 20 or less entries then the write counter will be initialized with all zeros, while the read counter will be initialized with 00000010100. Thus when the write counter equals the read counter the FIFO will have 20 entries, the $\overline{AE}$ flag logic will trip and cause $\overline{AE}$ to go inactive. Similarly, if the $\overline{AF}$ flag is to be active when the FIFO has 1,004 or more entries the write counter will be initialized to all zeros and the read counter will be initialized to 01111101100.

The flags generated by the methods described above are not valid outputs since they can lead to metastability. This is because the counters are synchronous to separate clocks,

which means the flag signal is not synchronous to either clock. Subsequently, more logic is required to synchronize the flags and reduce the probability of metastability to insignificant levels. One way of achieving this is through the use of asynchronous state machines (ASMs). A good reference for ASMs is the book *Logic Design Principles* by E.J. McCluskey. Two ASMs are needed for each flag, a set ASM and a reset ASM that operate an SR latch. Together they synchronize the FF and AF flags to the WCLK and the EF and AE flags to the RCLK. For example, the set ASM for the EF flag should have the following characteristics:

- The ASM will hold a steady state until the look-ahead empty flag (empty +1) goes active. After the look-ahead empty flag goes active, the ASM will pulse the output on the very next rising edge of the enabled read clock.
- The set output will remain active for the full duration of the enabled read clock, and will deactivate when the enabled read clock deactivates.
- In the event of a concurrent enabled read and enabled write, the set ASM will maintain the output.

These requirements are sufficient to completely specify a flow table for the ASM, which, with the techniques discussed in the reference, lead directly to the ASM logic. The reset ASM will be very similar. As mentioned above, a look-ahead empty flag is needed, which means the read counter has been initialized with the value 1.

Table 3. Empty Flag Example

	MSB	10 LSBs		MSB	10 LSBs
Write Counter	0	1001100101		0	1001100101
Read Counter	0	1001100101		0	0111011001
XNOR/XNOR Operation	1	1111111111		1	0001000011
Bitwise AND Operation	1 => Empty			0 => Not Empty	

Table 4. Full Flag Example

	MSB	10LSBs		MSB	10 LSBs
Write Counter	1	0001110110		0	1010011110
Read Counter	0	0001110110		0	0001110110
XOR/XNOR Operation	1	1111111111		0	0100010111
Bitwise AND Operation	1 => Full			0 => Not Full	

Pin Count and Usage

This section describes how the pins on the DP and LD are to be connected and indicates the final pin count for the LD. The first case assumes a DP with no burst counter capability, while the second case is for a DP with burst counters.

Case 1: No Burst Counters

The LD requires two enable signals that control if data can be written to or read from the FIFO. The input data, DIN, is connected to the LD and the DP ($DQ_{0\text{-}nL}$), while DOUT is an LD output. However, the output data from the DP ($DQ_{0\text{-}nR}$) must enter the LD's buffer FIFO input. Both of the DP's address ports ($A_{0\text{-}mL,\ 0\text{-}mR}$) are controlled by the LD. The clock signals enter the LD, which are gated and sent to the DP ($CLK_{L,\ R}$). Assuming no expansion is used, the DP chip enable signals are tied to enable the DP. The byte select inputs ($\overline{B}_{0\text{-}xL,\ 0\text{-}xR}$) and the output enable pins ($\overline{OE}_{L,\ R}$) are tied low to allow reading and writing to all the data pins. The left read/write enable input ($R/\overline{W}_L$) is tied low (for writing), and the right ($R/\overline{W}_R$) high (for reading). Master Reset ($\overline{MRST}$) is controlled by the LD as well.

Total LD pin count consumed by FIFO: 3n + 2m + 13

where n is the data bus width and m is the address bus width.

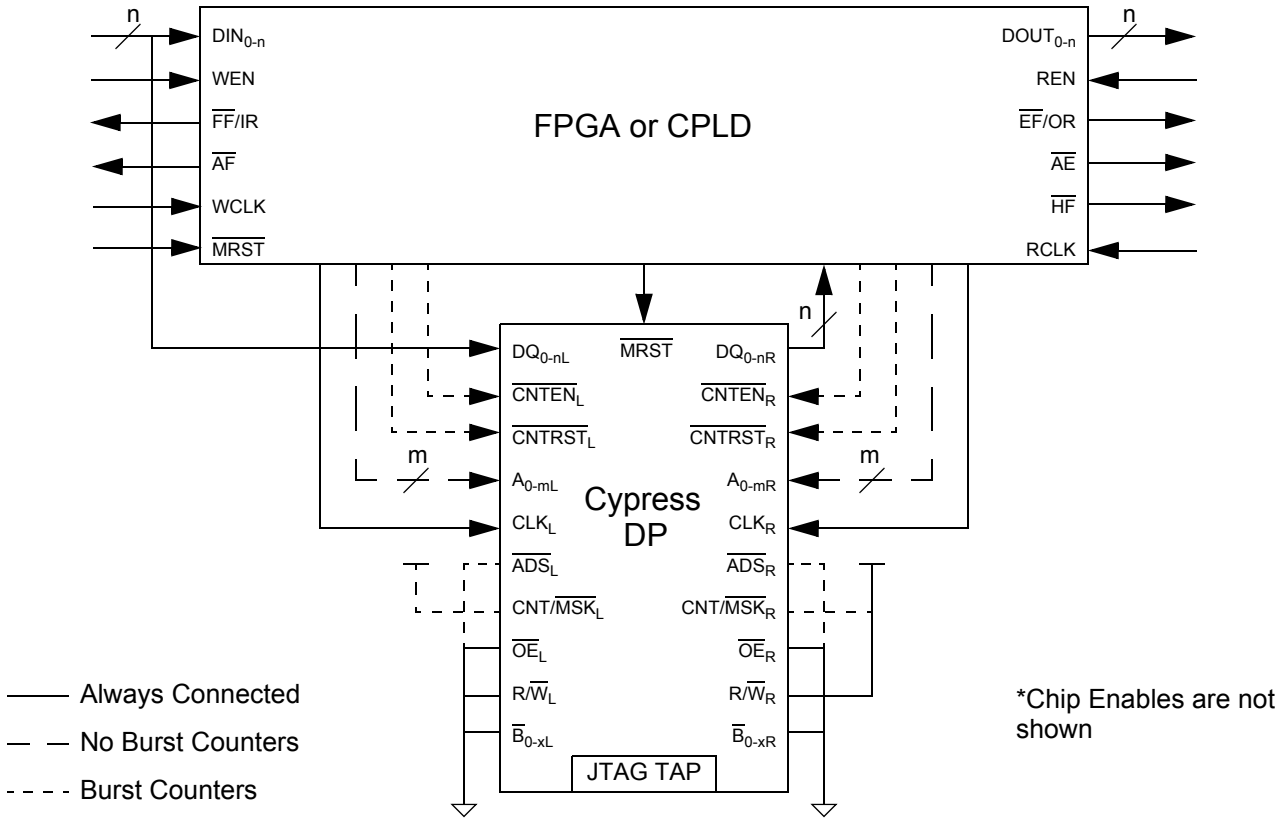
Case 2: Burst Counters

The DP connections are the same as in case 1 except the address inputs are now disconnected and tied either high or low, and new control signals are added. The counter enable inputs ($\overline{CNTEN}_{L,\ R}$) and counter reset inputs ($\overline{CNTRST}_{L,\ R}$) are controlled by the LD. The address strobe inputs ($\overline{ADS}_{L,\ R}$) are tied high to make sure the address lines are not polled. The address counter mask register enable inputs ($CNT/\overline{MSK}_{L,\ R}$) are tied high to prevent access to the counter mask registers.

Total LD pin count consumed by FIFO: 3n + 17

for a savings of 2m – 4 pins by using the burst counters. Figure 4 shows the connections between the LD and the DP, as well as the connections that are different between the two cases described above.

Figure 4. LD and DP Connections and Signal Names



Resource Estimation

Based on the DP FIFO solution described above, an estimate of the resources required for each type of LD is made. The estimate takes into account all the counters and comparators needed for the internal flag generation logic, the ASM logic for the external flags, and the buffer FIFO logic. For the Delta39K CPLD, approximately 600 macrocells and one channel memory block will be utilized. The solution will use about 300 logic slices and one RAM Block for Xilinx FPGAs, while Altera CPLDs will use about 600 logic elements (LEs) and one embedded systems block (ESB).

LD Speed Considerations

There are two components of the FIFO implementation within the LD that could be the critical delay paths. One is the buffer FIFO and the other is the internal flag logic. The buffer FIFO uses the internal RAM of the LD and other logic. For the Xilinx Virtex series, as shown in the Asynchronous FIFO LogiCORE data sheet, the speed for a 255 x 16 FIFO is given. Depending on the speed grade, the Virtex can operate the FIFO from 114 to 156 MHz, the Virtex-E from 178 to 196 MHz, and the Virtex-II up to 233 MHz. This FIFO is much deeper than what is needed for buffering purposes. A minimum depth 15 x 36 FIFO is better suited for this application, and would result in faster operating speeds.

The internal flag logic consists of wide adders, an XOR stage, and a wide AND gate, followed by the ASM logic and flag output logic. The output of the ASM is gated by a clock signal, so these paths are separated by a rising clock edge. Based on logic and path delay estimations, these paths should have comparable delays. For example, with the Altera APEX-II CPLD it is estimated that these paths can operate at speeds up to 200 MHz. Current CPLDs and FPGAs are efficient at wide data-bus operations, so increasing the width of the address bus for deeper DPs will have marginal effects on the operating speed.

Additional Features

Since the LD is programmable, there is virtually no limit to the features that can be implemented or the operations that can be performed on the data. For instance, packet information could be added to the data words, since all the data passes through the LD's buffer FIFO. If the full depth of the DP is not needed for the FIFO, it can be partitioned into segments.

Each of these segments can act as either memory for a FIFO or normal dual-ported memory. This gives the designer full flexibility in how to completely utilize the DP. The FIFO segments would require their own internal counters and flags, while the dual-port segments would need standard memory control logic.

The sections below describe more features that are standard in standalone FIFOs. However, these features are optional, and can be included at the designer's discretion. For more information refer to a FIFO data sheet.

AE/AF Flag Register Programming

There are three ways to program the $\overline{AE}$ and $\overline{AF}$ flag registers: parallel loading through the input port DIN, serial loading, or default values. These methods are simple to implement in the LD. It does require extra control pins however. A common solution is to use one pin that selects serial loading, and two dual-purpose pins. In serial mode one pin is an enable pin and the other is for the serial data. Otherwise, they select three default offsets or enable parallel loading through DIN.

- Three LD I/O pins

FIFO Master and Partial Reset

Each reset signal has its own pin. During Master Reset, all registers, counters, and programmable features are reset and reprogrammed. The DP's Master Reset ($\overline{MRST}$) is activated at this time as well. During Partial Reset, all counters are reset, but programmable registers such as the $\overline{AF}$ and $\overline{AE}$ flag registers are left untouched, as well as device settings such as bus-matching and endian select mode.

- One LD I/O pin for Partial Reset

Bus-Matching and Endian Select

Bus-Matching allows inputs or outputs with differing bus widths to use the FIFO to communicate with each other. Take, for example, the situation where a 36-bit wide bus needs to interface with a 9-bit wide bus. For every word written into the FIFO, four words would become available at the output. Endian Select is an extension to bus-matching. It allows the ordering of the words to change. Using the example from before, assume the 36-bit word ABCD (9-bit segments) has been written into the FIFO. In normal Big Endian mode the 9-bit words will be read out in this order: A, B, C, D. In Little Endian mode the words will be read out in the opposite order: D, C, B, A. See the data sheet for a FIFO with bus-matching and endian select for more details. These features require four control pins. One controls whether the FIFO is in Big Endian or Little Endian mode. Another controls whether bus-matching is enabled or not. The remaining two control which of the four combinations is being selected (for a x36 FIFO, with options for x18 and x9).

These features do change the way the data ports on the LD and DP are connected to each other. For instance, if DIN is 18-bits wide and the DP is 36-bits wide, then DIN is connected to the LD to buffer the first word (the 18 MSBs) as well as to the 18 LSBs of the DP's input port. The LD sends the buffered first word to the 18 MSBs of the DP's input port. The result is that the same number of LD I/Os are used, except for the extra control pins. However, if DOUT is not as wide as the DP width, then the total number of I/O pins used will decrease by the difference in widths.

- Four LD I/O pins

Retransmit/Mark and Retransmit

Retransmit is a mode where the FIFO read counter is reset to zero, and data is read out from that point forward. While in retransmit mode, the write counter cannot equal the read counter. This prevents the retransmit data from being overwritten. Upon leaving retransmit mode, operation resumes as normal. Mark and Retransmit expands upon the retransmit concept by allowing control over the memory position the read counter is reset to. A Mark input pin, when active, indicates the current read counter value is to be saved (marked). Upon retransmit, the counter is reset to this value. Mark and Retransmit requires the DP Address busses to be controlled by the LD as well as the Address Strobe inputs ($\overline{ADS}_{L,\ R}$). Two additional control inputs are needed, a retransmit pin and a mark pin.

- Four LD I/O pins (+ address overhead)

Mailbox Bypass Register

In situations where the two ports need to communicate data which is separate from the FIFO data stream or is time critical, mailbox bypass registers are a useful feature. Two registers are created in the LD that match the data bus widths for each port. If the DP has a mailbox feature, it cannot be used since two addresses within the memory array are used as the mailboxes, but the whole address space is needed for proper FIFO operation. Four extra I/O pins are required for this feature. Each port needs a read/write control pin that can act as a mailbox select as well. The remaining two pins are used to indicate that the mailbox is full. Data is prevented from being written to the mailbox until it has been read out by the other port. These signals need to be appropriately buffered within the LD to prevent metastability. See a DP data sheet for more information on how mailboxes function.

- Four LD I/O pins

First Word Fall Through (FWFT)

FWFT is a mode where data written to an empty FIFO appears at the output read port after a certain number of RCLK transitions, regardless of whether the port is enabled or not. Non-empty FIFO operation proceeds as normal. The $\overline{FF}$ and $\overline{EF}$ flags also take on slightly different meanings. The $\overline{FF}$ becomes Input Ready (IR), which indicates if there is space available for writing to the FIFO memory. The $\overline{EF}$ becomes Output Ready (OR), which indicates the presence of valid data on DOUT, ready to be read. For more information see the data sheet of a FIFO with FWFT mode. This fea-

ture requires one extra control pin that indicates whether the FIFO is in standard mode or FWFT mode.

- One LD I/O pin

Implementing all the features described above would require 19 LD I/O pins, in addition to the pin count described in the Pin Count and Usage section above.

Conclusion

The configurability of FPGAs and CPLDs allow Dual-Ported memory to be used as a FIFO buffer, which makes it a more versatile solution than a standalone FIFO. In addition, high speed FPGA/CPLDs have the capability to utilize most or all of the available DP bandwidth. These benefits make it an ideal solution for interprocessor communication and data-flow management.

Reference

E. J. McCluskey. *Logic Design Principles With Emphasis on Testable Semicustom Circuits*. Prentice Hall, Englewood Cliffs, NJ, 1986.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408-943-2600
Fax: 408-943-4730
http://www.cypress.com