# ProASIC^PLUS® RAM/FIFO Blocks

## Introduction

The memory in the ProASIC^PLUS family provides great configuration flexibility. Unlike many other programmable logic devices, each ProASIC^PLUS block is designed and optimized as a two-port memory (1 read, 1 write). This provides up to 198 kbits of total memory for two-port and single-port usage. Each memory block can be configured as a FIFO or SRAM with independent selection of synchronous or asynchronous read and write ports. Additional characteristics include programmable flags as well as parity checking and generation. The SRAM/FIFO memory blocks perform at up to 150 MHz (typical) when operated individually. Each block contains a 256-word, 9-bit wide (1 read port, 1 write port) memory. The memory blocks may be combined to form wider memories or deeper memories. This provides bit widths of 9 (1 block), 18, 36, and 72, and depths of 256, 512, 768, and 1,024. Actel's SmartGen software provides a user-friendly graphical interface that enables users to quickly design memories containing these configurations.

## Embedded Memory Floorplan

The embedded memory is located across the top and bottom (except APA075 and APA150, which are top only) of the device in 256x9 blocks. Depending upon the device, up to 88 blocks are available to support a variety of memory configurations. Each block can be programmed as an independent memory or can be combined (using dedicated memory routing resources) to form larger, more complex memories. A single memory configuration cannot include blocks from both the top and bottom memory locations. The maximum number of blocks for a single configuration is up to 44 for the APA1000. See Table 1 for memory configuration details of all ProASIC^PLUS FPGAs.

*Table 1* • **Maximum Memory Configuration Sizes**

| Device | Bottom Number of 256x9 Blocks (organization) | Top Number of 256x9 Blocks (organization) | Wide Maximum Size | | Deep Maximum Size | |
|---|---|---|---|---|---|---|
| | | | D (bits) | W (bits) | D (bits) | W (bits) |
| APA075 | 0 | 12 (2x6) | 256 | 54 | 1,536 (6x256) | 18 (9x2) |
| APA150 | 0 | 16 (2x8) | 256 | 144 | 2,048 (8x256) | 18 (9x2) |
| APA300 | 16 (2x8) | 16 (2x8) | 256 | 144 | 2,048 (8x256) | 18 (9x2) |
| APA450 | 24 (2x12) | 24 (2x12) | 256 | 216 | 3,072 (12x256) | 18 (9x2) |
| APA600 | 28 (2x14) | 28 (2x14) | 256 | 252 | 3,584 (14x256) | 18 (9x2) |
| APA750 | 32 (2x16) | 32 (2x16) | 256 | 288 | 4,096 (16x256) | 18 (9x2) |
| APA1000 | 44 (2x22) | 44 (2x22) | 256 | 396 | 5,632 (22x256) | 18 (9x2) |

## Naming Convention for RAMs

RAM model names can have up to four parts:

- Base name indicating the type and size (RAM256X9)
- One character code designating the write port as asynchronous (A) or synchronous (S)
- One or two character code designating the read port as asynchronous (A), synchronous registered (SR), or synchronous transparent (ST)
- Optional one character code designating parity (P) generated

For example: RAM256X9SAP is a 256-word by 9-bit RAM with synchronous write and asynchronous read ports using the generate parity feature.

## Naming Convention for FIFOs

FIFO model names are the same as the RAM model name except for the base name FIFO.

For example: FIFO256X9SSRP, a 256-word by 9-bit FIFO with synchronous write and synchronous read ports (synchronous to separate clocks named RCLKS and WCLKS), has registered outputs and uses the generate parity feature.

## Basic Memory Configuration

Every memory on a chip may be configured independently as dual- or single-port SRAM or FIFO. This includes the possibility of synchronous and asynchronous SRAMs or FIFOs configured side-by-side. In all of these modes, a parity bit (ninth bit) can be checked or generated. The parity is checked while reading data and while writing data with the result of these checks shown in two independent signals. For the synchronous SRAMs and FIFOs, the memory can be configured in such a way that it acts like a transparent synchronous memory or like a synchronous memory with an output register for the data signals. Table 2 shows the different memory modes that can be configured. Odd or even parity can be selected.

*Table 2* • **Basic Memory Configuration**

| Type | Write Access | Read Access | Parity | Library Cell Name |
|---|---|---|---|---|
| RAM | Asynchronous | Asynchronous | Checked | RAM256X9AA |
| RAM | Asynchronous | Asynchronous | Generated | RAM256X9AAP |
| RAM | Asynchronous | Synchronous Transparent | Checked | RAM256X9AST |
| RAM | Asynchronous | Synchronous Transparent | Generated | RAM256X9ASTP |
| RAM | Asynchronous | Synchronous Pipelined | Checked | RAM256X9ASR |
| RAM | Asynchronous | Synchronous Pipelined | Generated | RAM256X9ASRP |
| RAM | Synchronous | Asynchronous | Checked | RAM256X9SA |
| RAM | Synchronous | Asynchronous | Generated | RAM256X9SAP |
| RAM | Synchronous | Synchronous Transparent | Checked | RAM256X9SST |
| RAM | Synchronous | Synchronous Transparent | Generated | RAM256X9SSTP |
| RAM | Synchronous | Synchronous Pipelined | Checked | RAM256X9SSR |
| RAM | Asynchronous | Synchronous Pipelined | Generated | RAM256X9SSRP |
| FIFO | Asynchronous | Asynchronous | Checked | FIFO256X9AA |
| FIFO | Asynchronous | Asynchronous | Generated | FIFO 256X9AAP |
| FIFO | Asynchronous | Synchronous Transparent | Checked | FIFO 256X9AST |
| FIFO | Asynchronous | Synchronous Transparent | Generated | FIFO 256X9ASTP |
| FIFO | Asynchronous | Synchronous Pipelined | Checked | FIFO 256X9ASR |
| FIFO | Synchronous | Synchronous Pipelined | Generated | FIFO 256X9ASRP |
| FIFO | Synchronous | Asynchronous | Checked | FIFO 256X9SA |
| FIFO | Synchronous | Asynchronous | Generated | FIFO 256X9SAP |
| FIFO | Synchronous | Synchronous Transparent | Checked | FIFO 256X9SST |
| FIFO | Synchronous | Synchronous Transparent | Generated | FIFO 256X9SSTP |
| FIFO | Synchronous | Synchronous Pipelined | Checked | FIFO 256X9SSR |
| FIFO | Synchronous | Synchronous Pipelined | Generated | FIFO 256X9SSRP |

# Asynchronous SRAM

## RAM256X9AA, Asynchronous SRAM, Parity Check

The ports of the asynchronous SRAM with the parity bit checked are shown in Figure 1 and described in Table 3. The name of this cell in the library is RAM256X9AA. During a read or a write cycle, the corresponding error flag RPE for the read cycle or WPE for the write cycle is generated. The PARODD signal determines if the check is for even or odd parity. Odd parity is checked if the PARODD signal is HIGH and also if it is LOW. Refer to Table 4 for detail functionality.

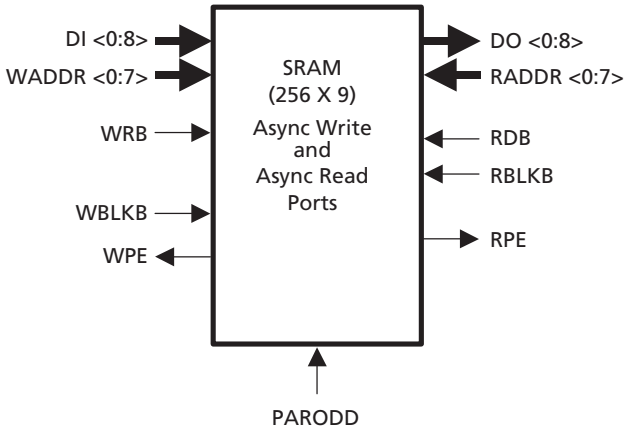


*Figure 1* • **RAM256X9AA**

*Table 3* • **Ports of the Asynchronous SRAM with Parity Bit Checked**

| Name | Direction | Function |
|---|---|---|
| DO8 down to DO0 | Out | Data output |
| WPE | Out | Write parity error |
| RPE | Out | Read parity error |
| WADDR down to WADDR0 | In | Write address |
| RADDR7 down to RADDR0 | In | Read address |
| DI8 down to DI0 | In | Data input |
| WRB | In | Write (bar = zero active) |
| RDB | In | Read (bar = zero active) |
| WBLKB | In | Write block (bar = zero active) |
| RBLKB | In | Read block (bar = zero active) |
| PARODD | In | Parity odd (when high) |

*Table 4* • **RAM256X9AA – Asynchronous Write and Read to the Same Address**

| Condition | Description |
|---|---|
| WB = '1' | Read occurs at the falling edge of RB (RB ↓) |
| WB = '0' | Read occurs at the rising edge of RB(RB ↑) |
| RB = '0' | Read occurs at the rising edge of WB (WB ↑) |
| WB = RB = '0' | Read occurs when RADDR or WADDR changes |
| RADDR! = WADDR | Read occurs when<br>(RDB = '0' and RADDR changes)<br>or falling edge of RDB (RDB ↓) |

## RAM256X9AAP, Asynchronous SRAM, Parity Generated

The ports of the library cell RAM256X9AAP are shown in Table 5. The main difference from the RAM256X9A is the WPE and RPE outputs. The data input still has nine signals although the ninth data input is internally used by the parity generator. The function of the remaining pins has not changed from the RAM256X9A library cell.

*Table 5* • **Ports of the Library Cell RAM256X9AAP**

| Name | Direction | Function |
|---|---|---|
| DO8 down to DO0 | Out | Data output |
| WADDR7 down to WADDR0 | In | Write address |
| RADDR7 down to RADDR0 | In | Read address |
| DI8 down to DI0 | In | Data input |
| WRB | In | Write (bar = zero active) |
| RDB | In | Read (bar = zero active) |
| WBLKB | In | Write block (bar = zero active) |
| RBLKB | In | Read block (bar = zero active) |
| PARODD | In | Parity odd (when high) |

## FIFO256X9AA, Asynchronous FIFO, Parity Check

The ports of the asynchronous FIFO with the parity bit checked are shown in Figure 2 on page 5 and described in Table 6 on page 5. The special signals for the FIFOs are FULL, EMPTY, EQTH, GEQTH, LGDEP[2:0], RESET, and LEVEL[7:0]. The FULL, EMPTY, and RESET signals are used as in every other standard FIFO. They indicate if the FIFO is completely empty or completely full, and RESET sets the read and the write pointer back to zero, although it does not erase the data in the FIFO. The EQTH and GEQTH signals are also available to monitor if the FIFO is filled less than, equal to, or more than a user-specified level. For this purpose, a LEVEL signal has to be provided to the FIFO. This LEVEL signal is eight bits deep, so every level of the 256-bit blocks can be monitored. The LEVEL signal is not allowed to be all '1' or all '0.' The EQTH and GEQTH signals are active HIGH. This means if the FIFO is less filled than the LEVEL signal specifies, both signals are LOW. If the FIFO is filled to the level, then both signals are HIGH. If it is filled higher than the level, the EQTH signal is LOW, and the GEQTH signal is HIGH. If a FIFO shallower than 256 bits is needed, this functionality can be created with the LGDEP[2:0] signal. This signal specifies the depth of the FIFO used in 2(LGDEP+1) steps. Therefore, it is possible to get FIFOs with a depth of 2, 4, 8, 16, 32, 64, 128, and 256 words.

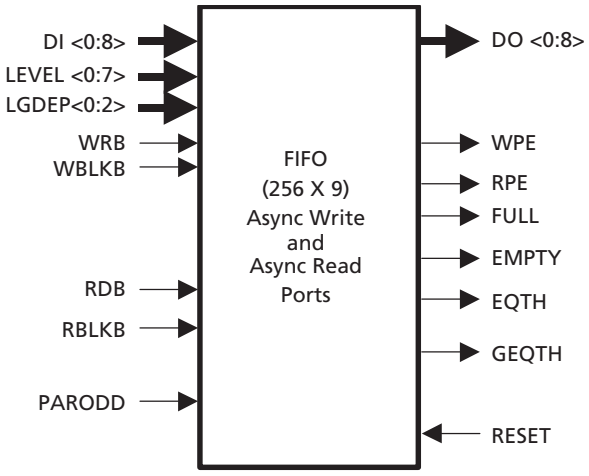


*Figure 2* • **FIFO256X9AA**

*Table 6* • **Ports of the FIFO256X9AA**

| Name | Direction | Function |
|---|---|---|
| DO8 down to DO0 | Out | Data output |
| FULL | Out | FIFO full flag |
| EMPTY | Out | FIFO empty flag |
| EQTH | Out | Filled to the level specified by LEVEL signal |
| GEQTH | Out | Filled to or over the LEVEL |
| WPE | Out | Write parity error |
| RPE | Out | Read parity error |
| LGDEP2 down to LGDEP0 | In | Defines DEPth of the FIFO |
| RESET | In | RESETs FIFO read and write pointer |
| LEVEL7 down to LEVEL0 | In | Specifies LEVEL of the EQTH and GEQTH |
| DI8 down to DI0 | In | Data input |
| WRB* | In | Write (bar = zero active) |
| RDB | In | Read (bar = zero active) |
| WBLKB | In | Write block (bar = zero active) |
| RBLKB | In | Read block (bar = zero active) |
| PARODD | In | Parity odd (when high) |

***Note:*** **After FIFO reset, WRB needs an initial falling edge prior to any write actions.*

## FIFO256X9AAP, Asynchronous FIFO, Parity Bits Generated

The ports of the asynchronous FIFO with parity bit generated are described in Table 7. The only difference in the interface between the FIFO256X9AA and the FIFO256X9AAP are the missing parity read and write error flag pins 'WPE' and 'RPE.' In this configuration, the FIFO generates a parity bit from the eight lower bits and stores the result as the ninth bit of the word. The polarity of the parity is dependent on the 'PARODD' input signal. If the signal is 'HIGH,' the parity is generated to have an odd parity, and if 'LOW' the parity will be even. If there is a ninth bit passed to the input line of the FIFO, the input signal will be (ignored and) overwritten with the parity setting.

*Table 7* • **Ports of the FIFO256X9AAP**

| Name | Direction | Function |
|---|---|---|
| DO8 down to DO0 | Out | Data output |
| FULL | Out | FIFO full flag |
| EMPTY | Out | FIFO empty flag |
| EQTH | Out | Filled to the level specified by LEVEL signal |
| GEQTH | Out | Filled to or over the LEVEL |
| LGDEP2 down to LGDEP0 | In | Defines depth of the FIFO |
| RESET | In | Resets FIFO read and write pointer |
| LEVEL7 down to LEVEL0 | In | Specifies LEVEL of the EQTH and GEQTH |
| DI8 down to DI0 | In | Data Input |
| WRB | In | Write (zero active) |
| RDB | In | Read (zero active) |
| WBLKB | In | Write block (zero active) |
| RBLKB | In | Read block (zero active) |
| PARODD | In | Parity odd (when high) |

# Synchronous SRAMs

## RAM256X9AST, Synchronous, Transparent SRAM, Parity Checked

The ports of a RAM256X9AST are shown in Figure 3 on page 7 and listed in Table 8 on page 7. This memory cell is synchronous to the clock signal provided to the cell. The incoming data is stored in the memory on the rising edge of the clock signal. Since the memory is synchronous and transparent, the data out will be provided in the same clock cycle. For the pipelined or registered mode, the data out signals will be valid shortly after the next rising edge of the clock signal.

The RDBLK signal on all synchronous memory cells is used internally to determine if the memory is in transparent or pipeline mode. As previously described, all the other signals have the same functionality.

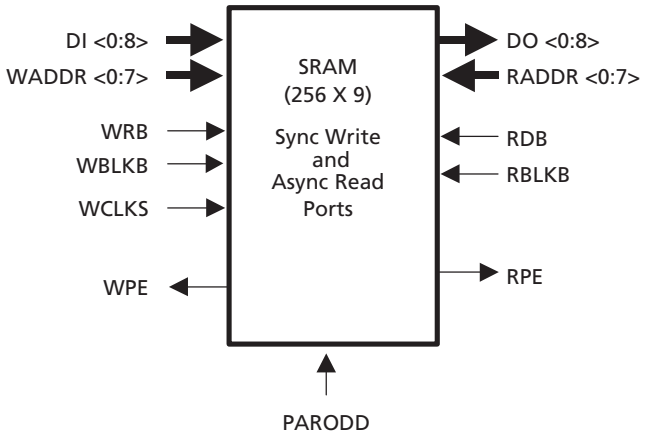


*Figure 3* • **RAM256X9AST**

*Table 8* • **Ports of the RAM256X9AST**

| Name | Direction | Function |
|---|---|---|
| DO8 down to DO0 | Out | Data output |
| WPE | Out | Write parity error |
| RPE | Out | Read parity error |
| WADDR7 down to WADDR0 | In | Write address |
| RADDR7 down to RADDR0 | In | Read address |
| DI8 down to DI0 | In | Data input |
| WCLKS | In | Write clocks |
| RCLKS | In | Read clocks |
| WRB | In | Write (zero active) |
| RDB | In | Read (zero active) |
| WBLKB | In | Write block (zero active) |
| PARODD | In | Parity odd (when high) |

## RAM256X9SSR, Synchronous, Registered SRAM, Parity Checked

The interface of this cell is exactly the same as that of the RAM256X9AST (Table 8). The difference is that output data is valid shortly after the next rising edge of the clock.

## RAM256X9ASTP, Synchronous, Transparent SRAM, Parity Generated

RAM256X9ASTP does not have the WPE and RPE output signals compared to the RAM256X9AST. The parity is generated internally while reading or writing the data. The lower eight bits are analyzed and the ninth bit is generated according to the PARODD flag. When PARODD is HIGH, parity is odd; when PARODD is LOW, parity is even. If there is a ninth bit written to the memory, it will be internally overwritten (ignored).

## RAM256X9SSRP, Synchronous, Registered SRAM, Parity Generated

The RAM256X9SSRP has the same interface as the RAM256X9ASTP. The difference is again the timing of the output data. The STP RAM supplies the data in the same clock cycle before the next rising edge of the clock, and the SRP will supply the data as if it was registered internally, shortly after the next rising edge of the clock.

# Synchronous FIFOs

## FIFO256X9AST, Synchronous, Transparent FIFO, Parity Checked

The FIFO256X9AST functionality is the same as the asynchronous FIFO except that the signals are stored and read out in relation to the clock signal. The data are stored with the rising edge of the clock, and the contents of the FIFO are shifted out in the same clock cycle. The ports of the synchronous FIFO with parity bit checked are shown in Figure 4 and described in Table 9.

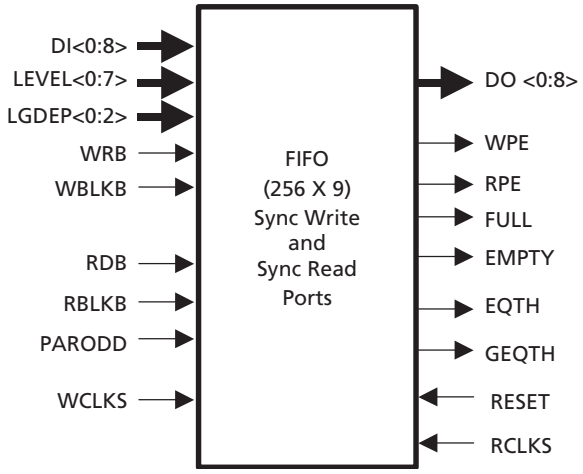


*Figure 4* • **FIFO256X9AST**

*Table 9* • **Ports of the FIFO256X9AST**

| Name | Direction | Function |
|---|---|---|
| DO8 down to DO0 | Out | Data output |
| FULL | Out | FIFO full flag |
| EMPTY | Out | FIFO empty flag |
| EQTH | Out | Filled to the level specified by LEVEL signal |
| GEQTH | Out | Filled to or over the LEVEL |
| WPE | Out | Write parity error |
| RPE | Out | Read parity error |
| WCLKS | In | Write clocks |
| RCLKS | In | Read clocks |
| LGDEP2 down to LGDEP0 | In | Defines depth of the FIFO |
| RESET | In | Resets FIFO read and write pointer |
| LEVEL7 down to LEVEL0 | In | Specifies LEVEL of the EQTH and GEQTH |
| DI8 down to DI0 | In | Data Input |
| WRB | In | Write (zero active) |
| RDB | In | Read (zero active) |
| WBLKB | In | Write block (zero active) |
| RBLKB | In | Read block (zero active) |
| PARODD | In | Parity odd (when high) |

As previously described, the functionality of the signals stays the same, but the timing of the signals changes.

### FIFO256X9ASR, Synchronous, Registered FIFO, Parity Checked

The interface is the same as the FIFO256X9ST; however, the timing for the outputs has changed. In registered mode, the data is provided after the next rising edge of the clock signal.

### FIFO256X9ASTP, Synchronous, Transparent FIFO, Parity Generated

No WPE and RPE signals can be found, because the parity is not checked while reading or writing the data. Instead, it is generated from the lower eight bits of the input data. The ninth bit will be generated corresponding to the PARODD signal for even (LOW PARODD signal) or odd (HIGH PARODD signal) parity in the resulting nine-bit word. If there is a ninth bit written to the FIFO, it is internally overwritten and the content is lost.

### FIFO256X9ASRP, Synchronous, Registered FIFO, Parity Generated

No WPE and RPE signals exist, because the parity is not checked while reading or writing the data, but it is generated from the lower eight bits of the input data. The ninth bit will be generated corresponding to the PARODD signal (HIGH for odd parity, LOW for even parity). If there is a ninth bit written to the FIFO, it will be internally overwritten (ignored).

## Deep and Wide, Synchronous RAMs and FIFOs

To obtain memories that are deeper or wider than the default 256x9 block, two or more RAM or FIFO cells must be used. Actel's SmartGen software facilitates building wider and deeper memories for optimal memory usage. SmartGen automatically generates the "glue" logic and a placement file for this logic when memories that are wider or deeper than the default size are selected. The following sections are for users who wish to understand the interconnecting logic better or wish to implement this logic by hand.

### Wider Memories

If a RAM with more than a nine-bit width is needed, two or more RAM cells of the same kind, e.g. RAM256X9SSR, must be connected together. The functionality of a RAM256x18SSR is given, if all the input signals with the same names are connected to the same input nets except for the DO[17:0] signals. Additionally, the parity error flags must be considered. If they are used, the question is if the parity is for the whole word of 16 bits plus parity, or if there will be two parity bits needed, one for each eight-bit word. If the parity has to be checked for every eight-bit word, then no additional logic is needed. If parity is needed on 16 bits, then additional logic will be needed.

## Deep Memories

### Deep RAMs

To create deep memories, slightly more user input is required compared to wide memories. For example, with the RAM512x9AST block, two RAM256X9 blocks will be needed to create the deep RAM. The most significant bit of the write address is the block select signal. Since just one of the blocks is active on a write cycle, this signal should be directly connected to the WBLKB signal of one RAM block, and it should be inverted before connecting to the WBLKB signal of the other block.

Multiplexors are used to choose which RAM cell provides the output data on a given read cycle. The most significant bit of the read address should be assigned to the select signal on the multiplexors. When more than two RAM blocks need to be connected, the multiplexor can be arranged in a ripple form or as a tree (Figure 5 on page 10). For timing reasons, the memory has a row of multiplexors directly connected to

their DO signals. These multiplexors are called DMUX. To enable the backend tool to place-and-route DMUX cells properly, the select signals for the DMUX cells have to be fed through the RAM blocks with which they are associated. The select signal is connected to the DIS pin of the RAM256X9 block and then a net has to be created between the DOS pin of the same RAM block and the select inputs of the DMUX cells. These connections are illustrated in Figure 5. If the RPE or WPE signals are used, they must be multiplexed. No DMUX cells are available for these signals. Figure 5 and Figure 6 show the basic block diagrams.

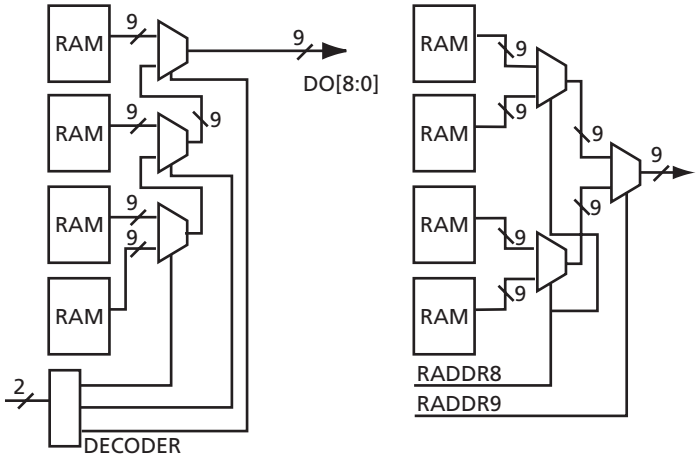


*Figure 5* • **How to Connect a DMUX Cell Select Pin**

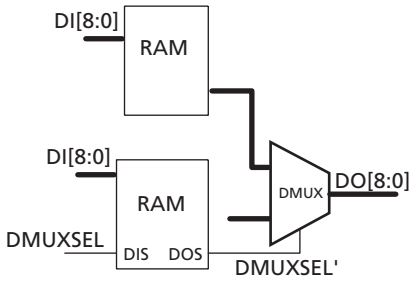


*Figure 6* • **Deep RAM Block Diagram**

## Deep FIFOs

Deep FIFOs are slightly more difficult to build than deep RAMs. This is because of the missing address signals. There has to be a block select generated internally for both the read and the write cycle. To guarantee proper functionality, the FIFOs must be filled up equally. That means that the block select for the read cycle is shifted with every read access from one FIFO to the next. When a read is asserted to the FIFO, the RDSHIFT register is enabled by the delayed version of the RDB. The next FIFO will be enabled for reading on the falling edge of the clock. As this occurs, the data DMUX is also switched to the next FIFO's DOUT bus on the following rising clock edge. Because of this switching behavior, the data will no longer be valid once the RCLK toggles, regardless of the state of the RDB signal. This effect is shown in Figure 9 on page 12. The write block select is shifted with every write cycle. Two shift registers are needed to build this functionality (Figure 7 on page 11). The shift registers are reset with the RESET signal. It is essential that both shift registers start with the same output vector.

The use of DMUX cells is the same as in the deep RAM design, therefore we get a basic block diagram for deep FIFOs, as shown in Figure 8 on page 11. An EMPTY signal or a FULL signal is generated for the deep FIFO if all FIFO blocks are full or empty. These signals must be connected to an AND gate to generate the correct functionality.

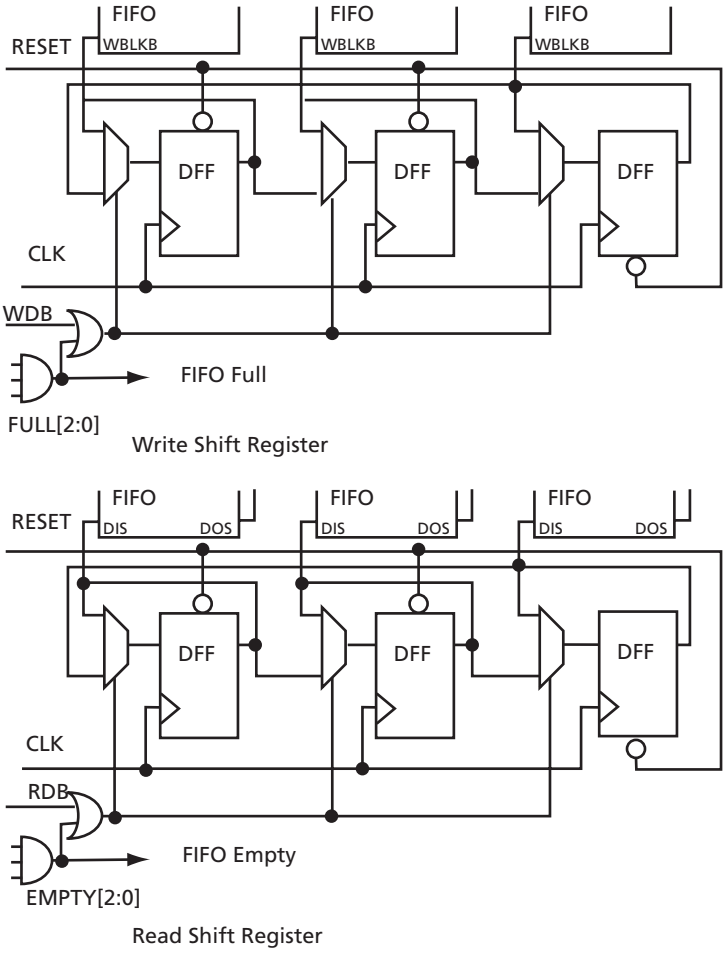


*Figure 7* • **Examples of Internal Shift Registers for Deep FIFOs**

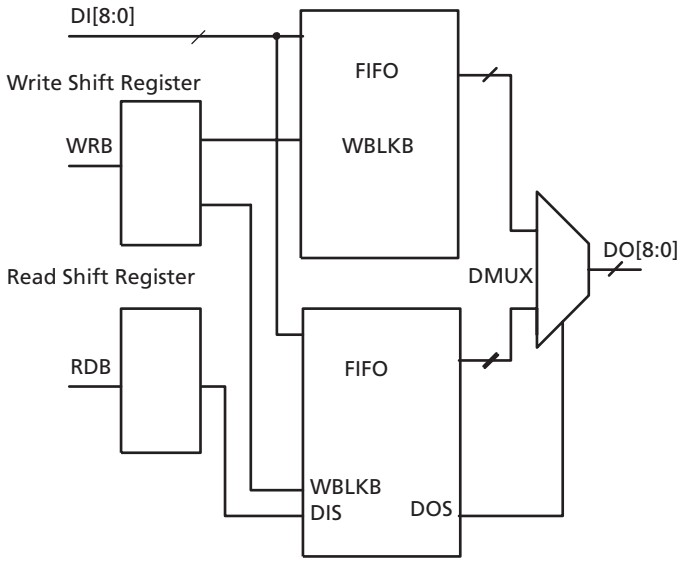


*Figure 8* • **Example of a Deep FIFO Block Diagram**

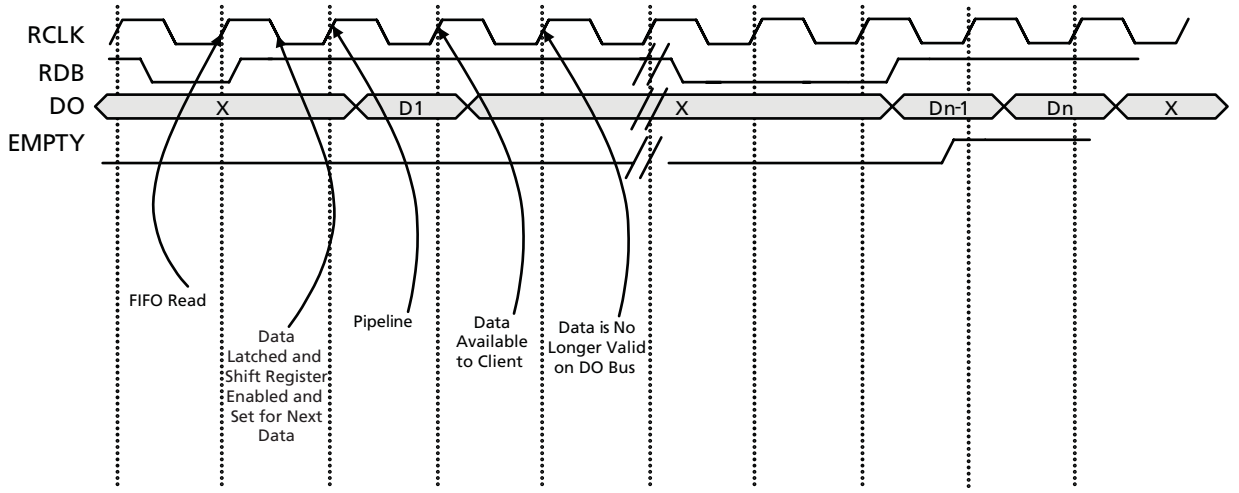


*Figure 9* • **Example of the Timing of a FIFO Read**

# Conclusion

The ProASIC^PLUS family of FPGAs provides very flexible memory blocks that can be implemented very quickly in any design. SmartGen generates and cascades memory blocks and defines the necessary RAM and FIFO logic. Actel recommends that users generate macros using the SmartGen core generator to facilitate this process.

# List of Changes

| Previous Version | Changes in Current Version 51900029-2/8.06* | Page |
|---|---|---|
| 51900029-1/2.06 | The "Deep FIFOs" section was updated to describe FIFO read timing. | 10 |
| | Figure 9 is new. | 12 |
| 51900029-0/6.03 | Table 3 was updated. | 3 |
| | Table 4 is new. | 3 |

***Note:*** **The part number is located on the last page of the document.*

www.actel.com

**Actel Corporation**

2061 Stierlin Court
Mountain View, CA
94043-4655 USA
**Phone** 650.318.4200
**Fax** 650.318.4600

**Actel Europe Ltd.**

Dunlop House, Riverside Way
Camberley, Surrey GU15 3YL
United Kingdom
**Phone** +44 (0) 1276 401 450
**Fax** +44 (0) 1276 401 490

**Actel Japan**
www.jp.actel.com

EXOS Ebisu Bldg. 4F
1-24-14 Ebisu Shibuya-ku
Tokyo 150 Japan
**Phone** +81.03.3445.7671
**Fax** +81.03.3445.7668

**Actel Hong Kong**
www.actel.com.cn

Suite 2114, Two Pacific Place
88 Queensway, Admiralty
Hong Kong
**Phone** +852 2185 6460
**Fax** +852 2185 6488

51900029-2/8.06