Axcelerator Family Memory Blocks

Introduction

As FPGA applications become more complex, designers' expectations of FPGA capabilities grow, making embedded memory blocks more and more essential in FPGA designs. The Axcelerator family FPGAs contain from four (in the AX125) to 64 (in the AX2000) blocks of embedded RAM/FIFO. Each block is a 4.5k variable-aspect-ratio dual-port RAM. The variable aspect ratio implies that each block can be configured independently in both depth and width. The allowable variable aspects are 128x36, 256x18, 512x9, 1kx4, 2kx2 or 4kx1. Moreover, the memory blocks can be cascaded in both width and depth to build larger blocks.

The RAM blocks have one read port and one write port, allowing simultaneous read and write operations. Furthermore, the read and write port widths are specified independently, allowing data bus conversion. The read and write operations are synchronous using independent clocks. The read access time from a RAM block is typically 2.2ns.

RAM blocks in the Axcelerator family can also operate in ROM emulation mode. To establish ROM functionality, the RAM blocks can be serially loaded via JTAG circuitry before the device is functional.

Axcelerator memory blocks are located on the left side of each core tile. Therefore, the number of available memory blocks in each device depends on the number of core tiles. For example, in an AX125 device with a single core tile, the number of available memory blocks is 4, while the AX500, with four core tiles, has 16 memory blocks. Note that the AX250 is an exception since the core tile size is different from the rest of the family, accommodating three memory blocks.

Embedded Memory Block Architecture

For a better understanding of the different features of the Axcelerator family memory blocks and their functionality, the user needs to understand the architecture of each individual block.

The memory blocks of the Axcelerator family can be configured as either RAM or FIFO. Since the configuration for each is different, we will discuss RAM and FIFO blocks separately even though each is implemented using the same memory block.

RAM Block Architecture

As shown in Figure 1, the architecture of the Axcelerator family RAM block consists of a memory core and associated peripheral blocks. The signals are described in the subsections that follow.

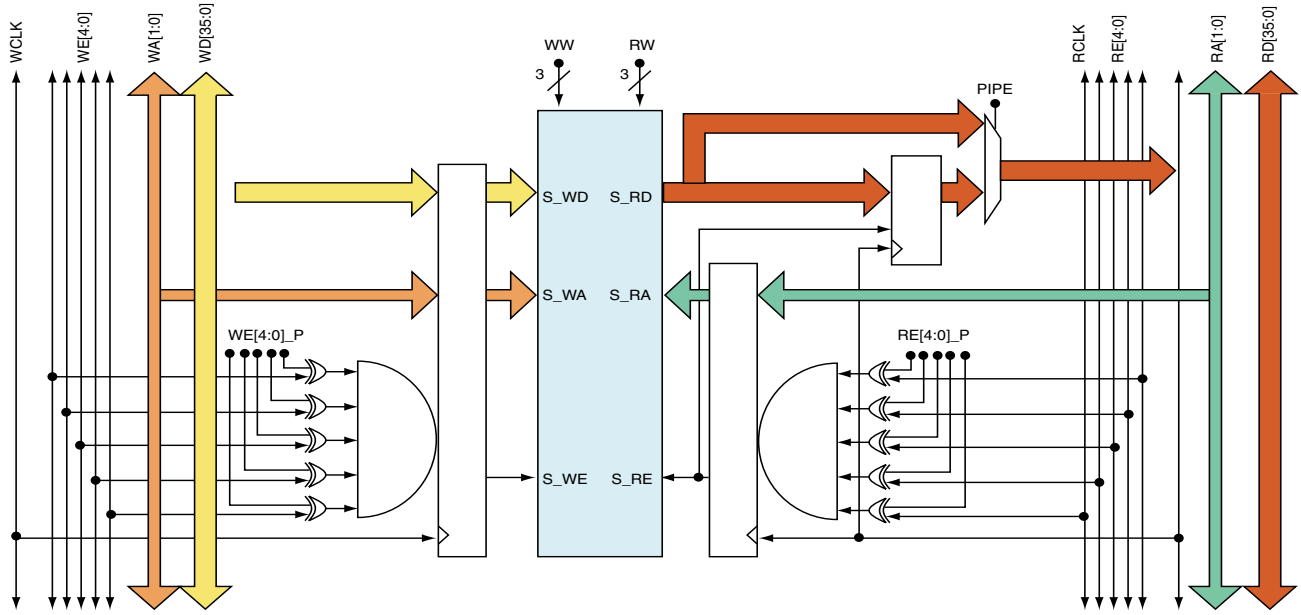

Figure 1 • The Axcelerator Family Embedded RAM Block

© 2002 Actel Corporation

RD and WD

RD and WD are 36-bit busses carrying the read and write data, respectively. These busses are linked to the adjacent memory blocks to allow cascading.

RA and WA

RA and WA are 12-bit read and write address busses, respectively. A width of 12 allows addressing the largest configuration in depth (4kx1). For RAM configurations with less depth the unused LSBs of these busses should be grounded. For example in a 2kx2 RAM configuration, RA[0] and WA[0] are grounded.

RE and WE

There are two 5-bit independent enable controls: WE for write data and RE for read data. Read and write functions can be performed if the enable input to the memory core is active. The enable signals have independent polarity control. One of the five enable bits is used as a regular enable signal to the memory core. The other four bits are used as extra address lines to implement larger memory configurations by cascading memory blocks. Therefore, up to 16 Axcelerator family RAM blocks can be cascaded; this is the maximum number of memory blocks in a column. The embedded control logic handles assigning the proper values to the control signals for cascading purposes.

RCLK and WCLK

RCLK and WCLK are read and write clocks, respectively. The read and write operations can be carried out independently either on the positive or negative edge of the clock (user-selectable). RCLK and WCLK signals are explicitly routed to each RAM block.

RW and WW

RW and WW are 3-bit busses used for RAM data width configuration. As mentioned earlier, the memory blocks can be configured with different aspect ratios. These ratios are determined by RW and WW for read and write ports, respectively. Table 1 indicates the possible values for WW and the resulting configurations. The RW bus is similar to WW.

***Table 1** • Word Width Configuration for RAM*

WW	Write Word Width	WA LSB to be Grounded
000	4kx1	None
001	2kx2	WA[0]
010	1kx4	WA[1:0]
011	512x9	WA[2:0]
100	256x18	WA[3:0]
101	128x36	WA[4:0]
11x	Reserved	N/A

PIPE

To have a pipelined read output, the PIPE signal should be set high. As shown in Figure 1 on page 1, the read output of the memory core is input to a MUX. By setting PIPE high, the output will be registered prior to the RD bus. Therefore, in a pipelined RAM block, two active edges of RCLK are required to have the read data available on the RD bus. Hence, when the PIPE is on, there is no memory access time involved in the read cycle. Therefore, RAM blocks can operate in two read modules: pipelined and transparent. The mode selection is accessible to the user in the ACTgen macro builder.

After power up the initial values in the RAM cells are unknown. The initialization of the RAM can be done by writing to the memory cells or loading the RAM blocks through JTAG ports. Refer to Actel's web site for future application notes regarding details on the ROM Emulation mode in Axcelerator devices.

FIFO Block Architecture

The implementation of the FIFO block is slightly different from a RAM. Figure 2 on page 3 shows the architecture of the Axcelerator family synchronous FIFO blocks.

The FIFO block components shown in Figure 2 on page 3 are provided with every RAM block and are controlled by the five programmable bits of DEPTH. The LSB of DEPTH enables the FIFO logic, and the upper four bits determine the total depth of the FIFO blocks. Therefore, the FIFO blocks, similar to the RAM blocks, can be cascaded up to a maximum of 16 in a core-tile column. This will be discussed in detail later in the "Cascading Memory Blocks" section on page 4. It can be seen from Figure 2 on page 3 that the read address is pipelined in two stages with the write clock (WCLK) and the write address is pipelined in two stages with the read clock (RCLK). This has been done to avoid metastability problems of the addresses.

The functionality of the signals that are also used in a RAM block implementation, shown in Figure 1 on page 1, was described in the previous section. Here we discuss the signals that are unique to the FIFO block controller.

FREN and FWEN

FREN and FWEN signals can separately enable the synchronous FIFO read and write operations, respectively, by overriding the RE[4] and WE[4] signals. As can be seen in Figure 2 on page 3, RE[4] and WE[4] are used as the write and read enables to the RAM block. Read and write enable have independent polarity control and are user-selectable.

Flags

The FIFO block has 4 flags: FULL, EMPTY, AFULL, and AEMPTY. The computation of the FULL flag is done in the

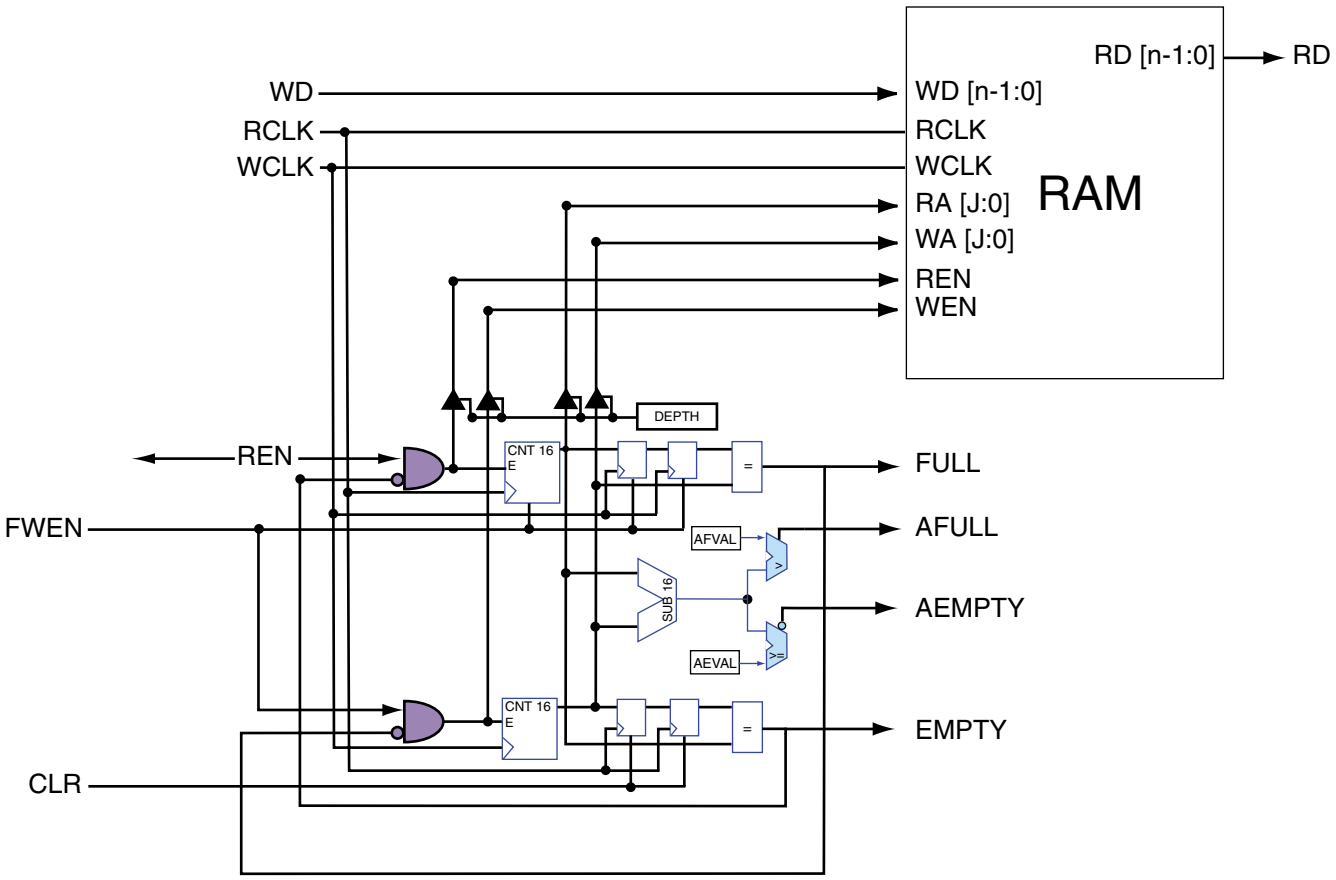

***Figure 2** • The Synchronous FIFO Block in Axcelerator Family Devices*

WCLK domain by registering the two previous values of the read address. If the current write address is equal to the registered value of the read address, the FULL flag will be asserted. The FULL flag disables further write operations by turning off WE[4]. Similarly the EMPTY flag computation is done in the RCLK domain by registering the two previous values of the write address. The assertion of the EMPTY flag disables further read operations by turning off RE[4]. Figure 2 indicates that the AFULL flag value is determined simply by comparing the write address with a predetermined value (AFVAL[7:0]). Similarly the result of comparison of the read address with the AEVAL[7:0] will determine the AEMPTY flag's value. The values for the almost empty and almost full flags can be set dynamically (i.e., during device operation). This will be discussed later when we explain the macros created by ACTgen. If not used, AEVAL and AFVAL should be grounded.

The allowable values for AEVAL and AFVAL depend on the data width. Representing the data width by "w", Table 2 on page 4 lists the step size for those flags.

If the memory block is configured as FIFO, the read and write addresses are generated by two independent counters (designated as CNT16 in Figure 2). The outputs of the counters are buffered onto the address lines. EMPTY and FREN can disable the read address counter to prevent further read operations. Similarly, FULL and FRWN signals control the enable input of the write address counter.

When Axcelerator FIFO uses less address space than the address space defined by the configuration, the FULL flag will not assert at the intended Full. Therefore, AFULL should be configured to assert for the intended Full. For example, when the user requires a memory block as an 80x36 FIFO, the FIFO is configured as 128X36. The FULL flag will assert at 128 not 80. In this case, the WW is set to 101 (representing 128x38 configuration) and therefore, the user is able to set AFVAL to 80 (refer to Table 2 on page 4 for legal AFVAL values) and use it as a FULL flag.

DEPTH

The enable inputs of the buffers that are loading the address bus of the memory block are controlled by the DEPTH signal. DEPTH is five bits wide. The LSB of DEPTH is used to enable or disable the synchronous FIFO configuration. In other words, if the embedded memory blocks are configured as RAM, then the LSB of DEPTH is set

Table 2 • DEPTH Configuration for Cascaded FIFO

DEPTH	Cascaded Blocks	FULL	Address Bus WCNT/RCNT	AEVAL/AFVAL step size
00001	1	2^{12-w}	[15:w]	2^{8-w}
00011	2	2^{13-w}	[15:w]	2^{8-w}
00111	4	2^{14-w}	[15:w]	2^{8-w}
01111	8	2^{15-w}	[15:w]	2^{8-w}
11111	16	2^{16-w}	[15:w]	2^{8-w}

Note: *w is the data width in Table 1.*

to '0' and the address lines of the memory block are no longer loaded by the address counters. If the user wants to configure the memory block as FIFO, the LSB of DEPTH should be tied high. The four MSBs of DEPTH are used to cascade the FIFO blocks if required. Hence, no more than 16 FIFO blocks can be cascaded. Table 2 shows the possible values for DEPTH in the FIFO configuration.

Looking at Table 2, if the FIFO block is configured as a 512x9 block (i.e. w=011), then the minimum value for AEVAL and AFVAL is 2^{8-3}, which is equal to 32. Notice that the *next* possible value for those flags is 64.

CLR

The CLR input to the FIFO is an asynchronous active-high signal used to initialize the FIFO block. By asserting the CLR the read and write address point to '0' (the very first cell in the FIFO block), the EMPTY and AEMPTY flags are set to '1' while the FULL and AFULL flags are '0'. Once the CLR signal is high, the read output bits of the FIFO block are set to'0'. When CLR signal is set back to low, the write procedure starts at the next active edge of WCLK and the write address will advance from 0 to 1. Since the write address is registered in two stages, it takes 2 RCLK active edges for the new address value (e.g., 1) to propagate to the comparator and be piped to the RCLK domain. Hence, after 2 RCLK active edges, the EMPTY flag is deactivated and the data can be read at the third active edge of the RCLK. In other words, it takes three active edges of RCLK to read the data after releasing the CLR signal.

Cascading Memory Blocks

As mentioned in previous sections, the memory blocks can be cascaded to implement larger blocks. There are different control signals in RAM and FIFO blocks to control the cascading procedure. These signals are WE, RE , and DEPTH. However, the variable aspect ratio of memory blocks allows cascading in various configurations. The implementation of larger memory blocks can be categorized by two approaches: cascading in depth or width. In this section we will discuss the difference between them as well as implementation of the two configurations.

Cascading in depth means that all the building blocks have the same read and write width as the required memory block, but the address bus width of the required large memory block is larger than the address bus width of each building block. For example, a 512x18 RAM block can be implemented by cascading two 256x18 RAM blocks, as shown in Figure 3.

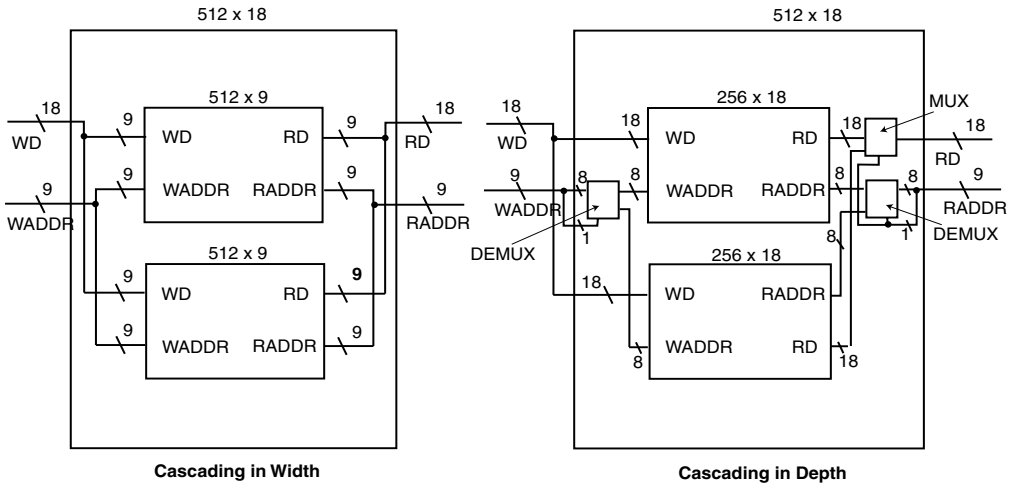

Figure 3 • Cascading Memory Blocks

On the other hand, with width cascading the width of the memory block address bus is equal to the address bus width of each building block. Each of the basic memory blocks is configured with the same depth as the required memory block. For example a 512x18 RAM block can be implemented with two 512x9 RAM blocks (Figure 3 on page 4). In order to build larger memory blocks, memory cores can be cascaded both in depth and width if required.

The preliminary characterization indicates that building large memory blocks by cascading in width shows better timing performance that the blocks implemented by cascading in depth. However, in some cases the *only* way to implement larger blocks is by cascading blocks in depth (e.g., implementation of an 8kx1). In those cases, the user should try to keep the number of the depth-cascaded blocks as low as possible to improve the timing performance.

ACTgen Macro Builder

Actel recommends that users generate the required macros with ACTgen software if possible. ACTgen generates and cascades memory blocks in a highly-efficient manner. In this section, we discuss the flow for generation of memory blocks in ACTgen and the associated macro features.

Figure 4 shows the ACTgen GUI of the RAM macro targeting the Axcelerator family.

The GUI has four major sections: device, pipeline, write, and read. By selecting pipelined RAM, the read data will be registered before being loaded onto the read data bus. The write and read depth and width are determined independently. However, the user should note that the following equation must hold:

Read depth * Read width = Write depth * Write width

ACTgen automatically cascades the memory blocks in the most efficient manner.

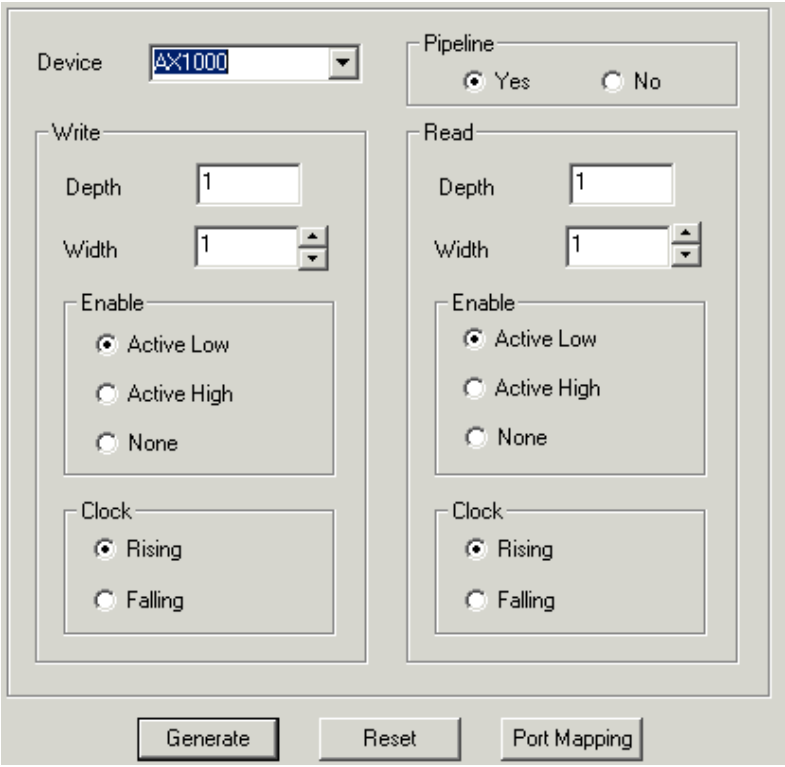

***Figure 4** • ACTgen GUI for Generating RAM Block*

Another useful feature of the ACTgen GUI is port mapping. By selecting this feature, the user can map the generated RAM block port names into his/her desired names. This facilitates using the memory blocks in larger designs. Figure 5 shows the GUI of the port mapping feature.

The ACTgen GUI for FIFO block generation is slightly different from RAM blocks. Figure 6 shows the GUI during FIFO block generation.

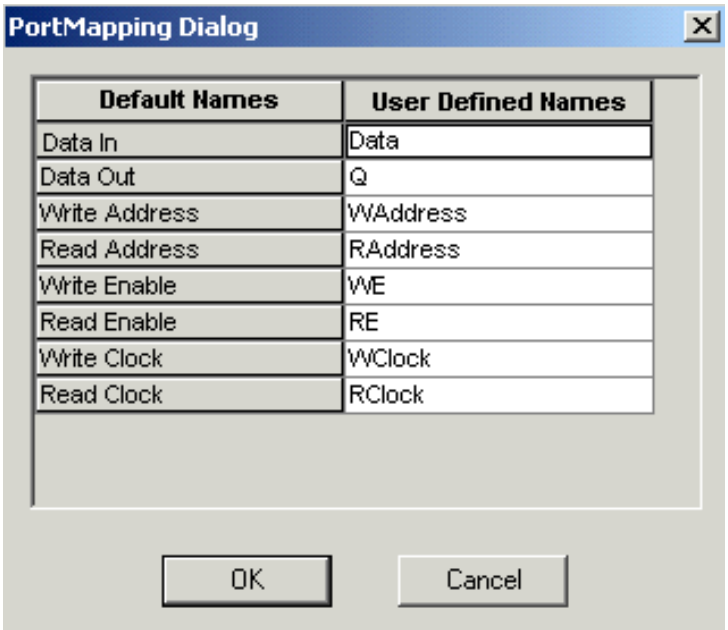

***Figure 5** • Port Mapping GUI*

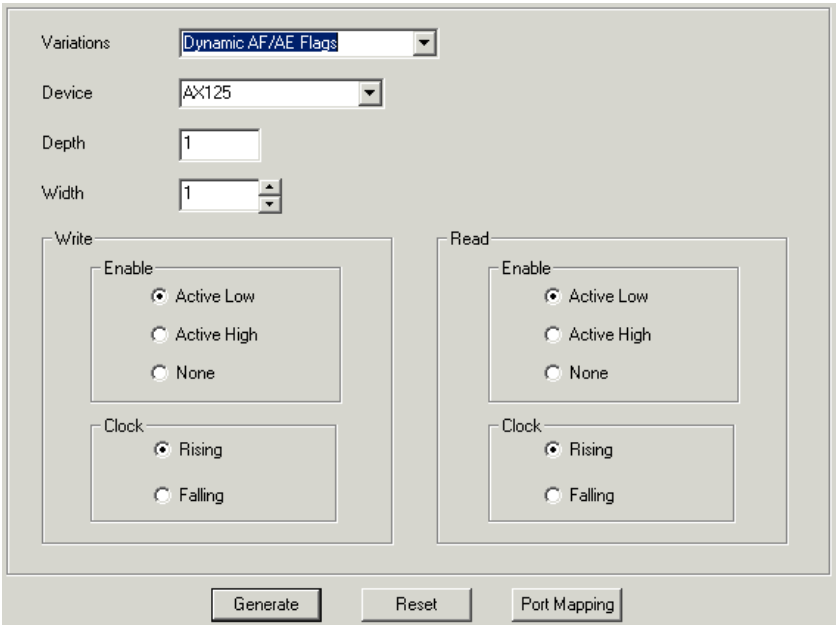

***Figure 6** • ACTgen GUI for FIFO Generation*

The Variations drop-down menu gives the option to select AEMPTY and AFULL flags. If the AFULL/AEMPTY flags are chosen as static, their values must be entered in the "Almost Full Value" and "Almost Empty Value" windows, considering the possible values mentioned in Table 2 on page 4. If the flags are chosen to be dynamic, then an 8-bit port for each flag will be available to the user in the generated macro. In dynamic mode, the values for the flags can be changed while the device is operational. A "None" option also exists under Variations; by selecting it, the AE and AF flags will be disabled.

The depth and width of the FIFO block must be entered in the respective boxes. The minimum values for the depth and the width of a FIFO block are '1.'

FIFO generation also supports a port mapping feature similar to the RAM block (Figure 7). Please note that on the port mapping list, AEMPTY and AFULL represent the values of the AEMPTY and AFULL flags, respectively, if dynamic configuration is chosen.

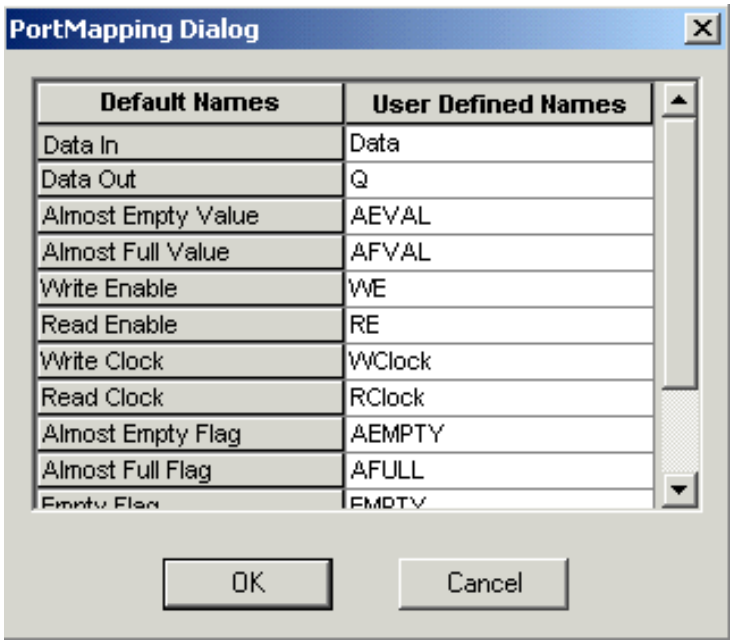

***Figure 7** • Port Mapping GUI in FIFO Generation*

Conclusion

The Axcelerator family devices provide very flexible memory blocks that make it easier to fit the memory blocks in a variety of designs. The memory blocks can run at very high speeds (the read time for a RAM block is 2.2ns). The Axcelerator devices use a single block that can be configured as either RAM or FIFO. The memory blocks support variable aspect ratios to support bus conversion and can be cascaded to build larger memories. Cascading in width results in better timing performance of the memory blocks. The ACTgen macro builder generates the required memory blocks in a highly-efficient, high-performance manner.

Actel and the Actel logo are registered trademarks of Actel Corporation.
All other trademarks are the property of their owners.

http://www.actel.com

Actel Europe Ltd.
Maxfli Court, Riverside Way
Camberley, Surrey GU15 3YL
United Kingdom
Tel: +44 (0)1276 401450
Fax: +44 (0)1276 401590

Actel Corporation
955 East Arques Avenue
Sunnyvale, California 94086
USA
Tel: (408) 739-1010
Fax: (408) 739-1540

Actel Asia-Pacific
EXOS Ebisu Bldg. 4F
1-24-14 Ebisu Shibuya-ku
Tokyo 150 Japan
Tel: +81-(0)3-3445-7671
Fax: +81-(0)3-3445-7668

5192694-3/5.02