Using ProASIC3/E FIFO for Generating Periodic Waveforms

Actel ProASIC3/E FPGAs contain embedded memory blocks that can be used as either RAM or FIFO. These memory blocks also include a dedicated FIFO controller to generate internal addresses and external flag logic (FULL, EMPTY, AFULL, AEMPTY).

In addition to the conventional flags, data, and address ports, ProASIC3/E FIFO blocks include two input pins: FSTOP and ESTOP. FSTOP is an active high signal used to stop the FIFO write-address counter when the FIFO is full (FULL flag is high). Similarly, the active high ESTOP signal stops the FIFO read counter from counting once the FIFO is empty (i.e., EMPTY flag goes high). When ESTOP is low, the read counter will roll over to the top (i.e. address '0') and start counting. If the ESTOP input is low, when the FIFO counter hits the bottom of the FIFO, the EMPTY flag becomes active for only one clock cycle. On the next cycle, when the read counter rolls over to the top, the flag is cleared.

One of the applications of ESTOP is to write into the FIFO once, then read the contents over and over. An example of this application is generating a periodic pattern/waveform out of the FIFO and the FPGA.

FIFO Initialization

As mentioned earlier, the FIFO blocks need be written once (initialized) with the data before the periodic reading starts. The initialization data can be provided to the FIFO by internal logic, embedded FROM (internal nonvolatile Flash memory cells) or external storage devices. Refer to the *ProASIC3/E FlashROM (FROM) application note* for more information.

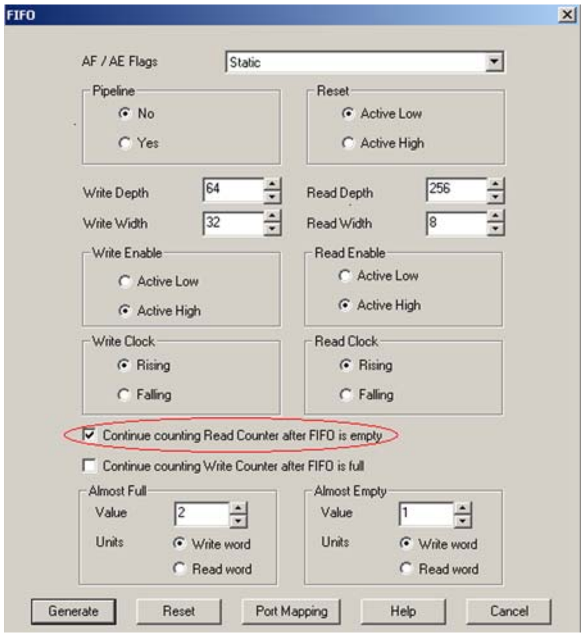
ProASIC3/E memory blocks feature variable-aspect ratio capability. Using this feature, the read and write ports can have different widths (and consequently depths). This may be useful in many applications, including the initialization of the FIFO blocks. For example, if each sample point of the desired waveform consists of eight bits (8-bit read width), the write port can be configured as 32-bit wide. This enables the user to enter four data points into the FIFO in one write clock cycle.

As a reference, the Sine Table on the Actel website contains a 1024x8 memory content for half-cycle of sine wave.

Creating a FIFO Block

The FIFO macro, FIFO4K18, can be instantiated directly from the ProASIC3/E macro library. After instantiation of the macro, designers must bind all the ports of the FIFO4K18 macro to the rest of the design or terminate them to appropriate logic based on the desired configuration and/or functionality of the FIFO block. This approach can be tedious and prone to errors. Actel recommends that designers use ACTgen core generator, integrated in the Actel Libero^{TM} Integrated Design Environment (IDE) software, to create FIFO blocks. ACTgen instantiates the FIFO macro from the library and connects all the ports appropriately based on user's entry. Refer to the *ProASIC3/E Macro Library Guide* for more information.

To create a FIFO with disabled ESTOP, users should check the **Continue counting Read Counter after FIFO is empty option** as shown in Figure 1 on page 2.


FIFO

AF / AE Flags: Static

Pipeline: [✓ No] Yes

Reset: [✓ Active Low] Active High

Write Depth: 64

Write Width: 32

Read Depth: 256

Read Width: 8

Write Enable: Active Low [✓ Active High]

Read Enable: Active Low [✓ Active High]

Write Clock: [✓ Rising] Falling

Read Clock: [✓ Rising] Falling

[✓] Continue counting Read Counter after FIFO is empty

[ ] Continue counting Write Counter after FIFO is full

Almost Full: Value 2; Units [✓ Write word] Read word

Almost Empty: Value 1; Units [✓ Write word] Read word

Figure 1 • **ACTgen GUI During ProASIC3/E FIFO Generation**

Waveform Generator Design

Two essential building blocks of a complete waveform generator design are the initialization block and FIFO. As mentioned in the "FIFO Initialization" section on page 1, there are various solutions to initialize the FIFO block. Figure 2 illustrates the block diagram for a waveform generator design when an external nonvolatile memory is used to initialize ProASIC3/E FIFO.

A reference design for initialization block in Figure 2 is presented in the *Embedded SRAM Initialization Using External Serial EEPROM application note*. The "Creating a FIFO Block" section on page 1 discusses the creation and instantiation of the FIFO block.

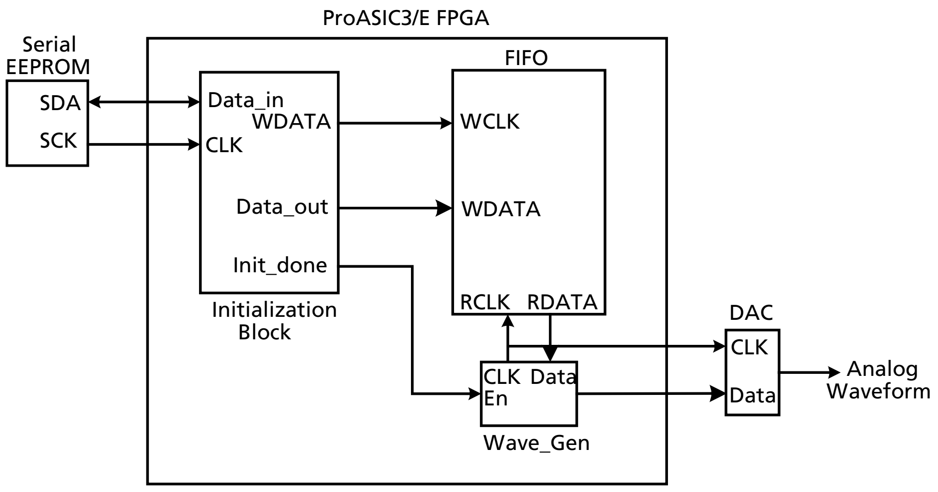


Figure 2 • **Block Diagram of the Waveform Generator Using External EEPROM**

The Wave_Gen block shown in Figure 2 on page 2 reads the waveform data points and sends it out of the FPGA to be converted into an analog waveform. The structure and functionality of Wave_Gen block depends entirely to the waveform type (sine wave, square wave, etc.) and the format of the data stored in the FIFO (two data points per row, half or full cycle, etc.). The Sine Table on the Actel website contains the initialization data for half of a cycle of a sine wave. If this table is used to initialize FIFO blocks, the contents of the FIFO should be read twice to generate a full cycle of periodic sine wave. The following is an example of the Wave_Gen block in VHDL that uses the half-cycle data, which is stored in the FIFO, and transmits a full sine wave data to the external D/A. The Wave_Gen design in this example, enabled at the end of initialization, reads through the FIFO block (half-cycle sine wave) and changes the sign (+/-) of the read data alternatively at the end of each half-cycle to generate a full-cycle sine wave.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity wave_gen is
   port( clock, reset_n, enable, empty: in std_logic;
         wave_in : in  std_logic_vector(7 downto 0);
         renb_fifo : out std_logic;
         wave_out: out std_logic_vector( 8 downto 0 ));
end;

architecture behave of wave_gen is
   type state_type is ( pos_cycle, neg_cycle);
   signal current_state, next_state: state_type;
   signal current_sign, next_sign: std_logic;
begin

   process( clock, reset_n )
   begin
      if reset_n = '0' then
         current_state <= pos_cycle;
         current_sign <= '0';
      elsif rising_edge( clock ) then
         if enable = '1' then
         if (empty = '1') then
            current_state <= next_state;
            current_sign <= next_sign;
            end if;
         end if;
      end if;
   end process;

process(current_state)
```

```
begin
   case current_state is
      when pos_cycle =>
         next_state <= neg_cycle;
         next_sign <= '1';
            when neg_cycle =>
         next_state <= pos_cycle;
         next_sign <= '0';
            when others =>
         next_state <= pos_cycle;
         next_sign <= '0';
   end case;
end process;

process(clock, reset_n)
begin
   if (reset_n = '0') then
      wave_out <= (others => '0');
   elsif rising_edge( clock ) then
      wave_out <= current_sign & wave_in;
   end if;
end process;

process (enable, reset_n)
begin
if (reset_n = '0' or enable = '0') then
   renb_fifo <= '1';
else
   renb_fifo <= '0';
end if;
end process;

end;
```

Related Documents

Application Notes

ProASIC3/E FlashROM (FROM)

www.actel.com/documents/PA3_E_FROM_AN.pdf

Embedded SRAM Initialization Using External Serial EEPROM

www.actel.com/documents/EmbeddedSRAMInit_AN.pdf

User's Guides

ProASIC3/E Macro Library Guide

www.actel.com/documents/pa3_libguide.pdf

Actel

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

51900080-0/01.05