CoreAI v3.0

Handbook

Actel
POWER MATTERS

Actel Corporation, Mountain View, CA 94043

Printed in the United States of America

Part Number: 50200102-2

Release: February 2009

Trademarks

Table of Contents

Introduction

CoreAI (Analog Interface) allows for simple control of the analog peripherals within the Fusion device family. Control may be implemented with an internal or external microprocessor or microcontroller such as CoreABC, Core8051s, or Cortex-M1, etc., or with user-created custom logic within the FPGA fabric.

Key Features

CoreAI has the following features:

- Thin processor interface around Fusion AB (Analog Block) hard macro
- ADC conversions controlled by processor writes
- Internal logic interface for controlling the ACM (Analog Configuration MUX)
- Internal logic to divide clock for generating ACM clock
- Optional hardware-controlled inputs to directly control some AB functions
- Interrupt logic for various events (such as end of ADC conversions)

Core Overview

The industry-standard AMBA (Advanced Microcontroller Bus Architecture) APB (Advanced Peripheral Bus) slave interface is used as the primary control mechanism within CoreAI.

CoreAI instantiates the AB (Analog Block) macro (see Figure 1). The AB macro includes the ACM (Analog Configuration MUX) interface, Analog Quads, and RTC (Real-Time Counter). The ACM interface, within the AB macro, is used to control configuration of the Analog Quads and RTC in the Fusion device.

CoreAI generates the control signals used by the ACM, including its clock signal, which is generated by an internal clock divider. The ACM clock divider is used to ensure that the ACM interface is clocked at a frequency less than or equal to 10 MHz (see "ACM Interface" on page 36 for details). For more details on the silicon features of the AB macro, such as the Analog Quads, RTC, or ACM, refer to the Fusion datasheet (http://www.actel.com/products/fusion/docs.aspx).

Several aspects of CoreAI can be configured using top-level parameters (Verilog) or generics (VHDL). For a detailed description of the parameters/generics, refer to Table 3-1 on page 17. The CoreAI block diagram is shown in Figure 1. A typical application using CoreAI is shown in Figure 2.

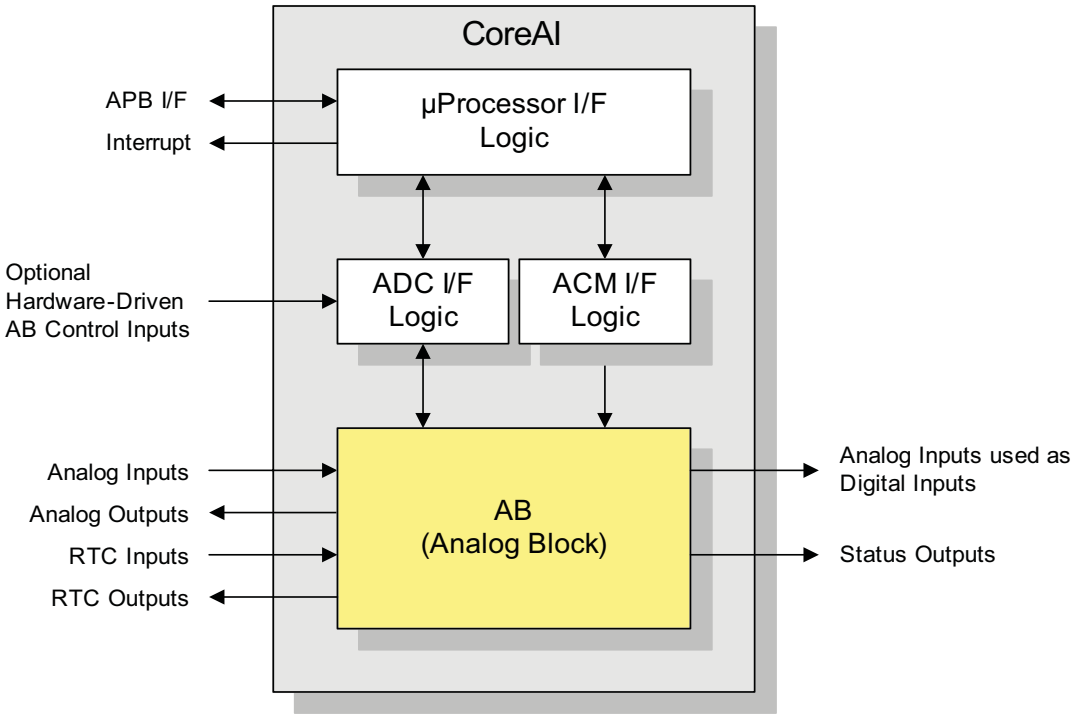

Figure 1 · Core AI Block Diagram

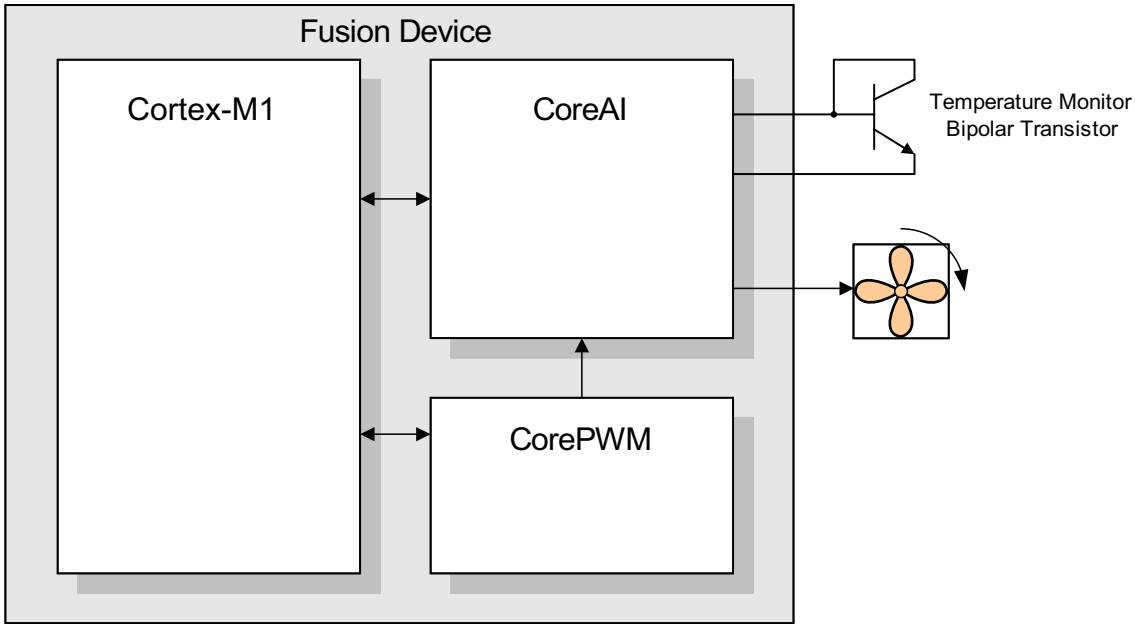

Figure 2 · Typical CoreAI Application

Supported Device Families

Fusion

Core Version

This handbook applies to CoreAI v3.0.

Supported Interfaces

CoreAI is available with the APB (Advanced Peripheral Bus) slave interface and must be connected to an APB master interface or to dedicated FPGA logic that implements an APB master interface.

Actel recommends that you use SmartDesign in the Libero® IDE Project Manager to connect and configure CoreAI in a processor-based system, using Cortex-M1, Core8051s or CoreABC.

Utilization and Performance

CoreAI has been implemented in the Actel Fusion device family. A summary of the data for CoreAI is listed in page 8 and Table 2. CoreAI can be used with any device in the Fusion family.

Table 1 · CoreAI Device Utilization and Performance (minimum configuration)

Family	Tiles			Utilization		Performance
	Sequential	Combinatorial	Total	Device		
Fusion	38	123	161	AFS090	7%	up to 100 MHz

Note: Data in this table were achieved using typical synthesis and layout settings. Top-level parameters/generics that differ from the default values were set as follows: PCLK_FREQUENCY = 40, CFG_VAREFSEL = 2, CFG_MODE = 32, CFG_TVC = 512, CFG_STC = 512, CFG_ACx = 512, CFG_ATx = 512, CFG_TMSTBINT = 2, CFG_GDx = 768, CFG_INTERRUPT = 3, APB_DWIDTH = 16.

Table 2 · CoreAI Device Utilization and Performance (maximum configuration)

Family	Tiles			Utilization		Performance
	Sequential	Combinatorial	Total	Device		
Fusion	103	208	317	AFS600	2%	up to 100 MHz

Note: Data in this table were achieved using typical synthesis and layout settings. Top-level parameters/generics that differ from the default values were set as follows: PCLK_FREQUENCY = 40, CFG_RTC = 1.

1 Functional Block Description

CoreAI, shown in Figure 1 on page 6, consists of the microprocessor interface logic, ACM interface logic, and ADC interface logic blocks. The microprocessor interface logic implements APB slave logic and generates a maskable interrupt. The ACM interface block writes configuration data into the AB macro to control Analog Quad and RTC settings. The ADC interface block sends control data to and receives status information from the ADC.

2 Tool Flows

Licenses

CoreAI is licensed in two ways. Depending on your license type, tool flow functionality may be limited.

Obfuscated

Complete RTL code is provided for the core, allowing the core to be instantiated within SmartDesign. Simulation, Synthesis and Layout can be performed within the Libero IDE. The RTL code for the core is obfuscated and some of the testbench source files are not provided; they are pre-compiled into the compiled simulation library instead.

RTL

Complete RTL source code is provided for the core and testbenches.

SmartDesign

CoreAI is available for download to the SmartDesign IP Catalog via the Libero IDE web repository. The parameters/generics of the core can be configured using the IP configuration GUI within SmartDesign as shown in Figure 2-1, Figure 2-2, and Figure 2-3.The parameters/generics of the core are fully described in "Parameters/Generics" on page 17 and cross references to them are shown next to the IP configuration screenshots in Figure 2-1, Figure 2-2, and Figure 2-3.

Note: **If you are using an AFS090 (or AFS090 variant) or AFS250 (or AFS250 variant), you must make the correct choice in the CoreAI IP configurator within SmartDesign via the Fusion Device selection box. For example, as shown in Figure 2-1, the device selected is AFS090. If you are not using either an AFS090 or AFS250 (or variants thereof), you must make the default choice of larger than AFS250 in the Fusion Device selection box.**

For information on using SmartDesign to instantiate, configure, connect, and generate cores, refer to the Libero IDE online help. For a detailed tutorial on DirectCore IP flow using SmartDesign, refer to Using DirectCore in Libero IDE.

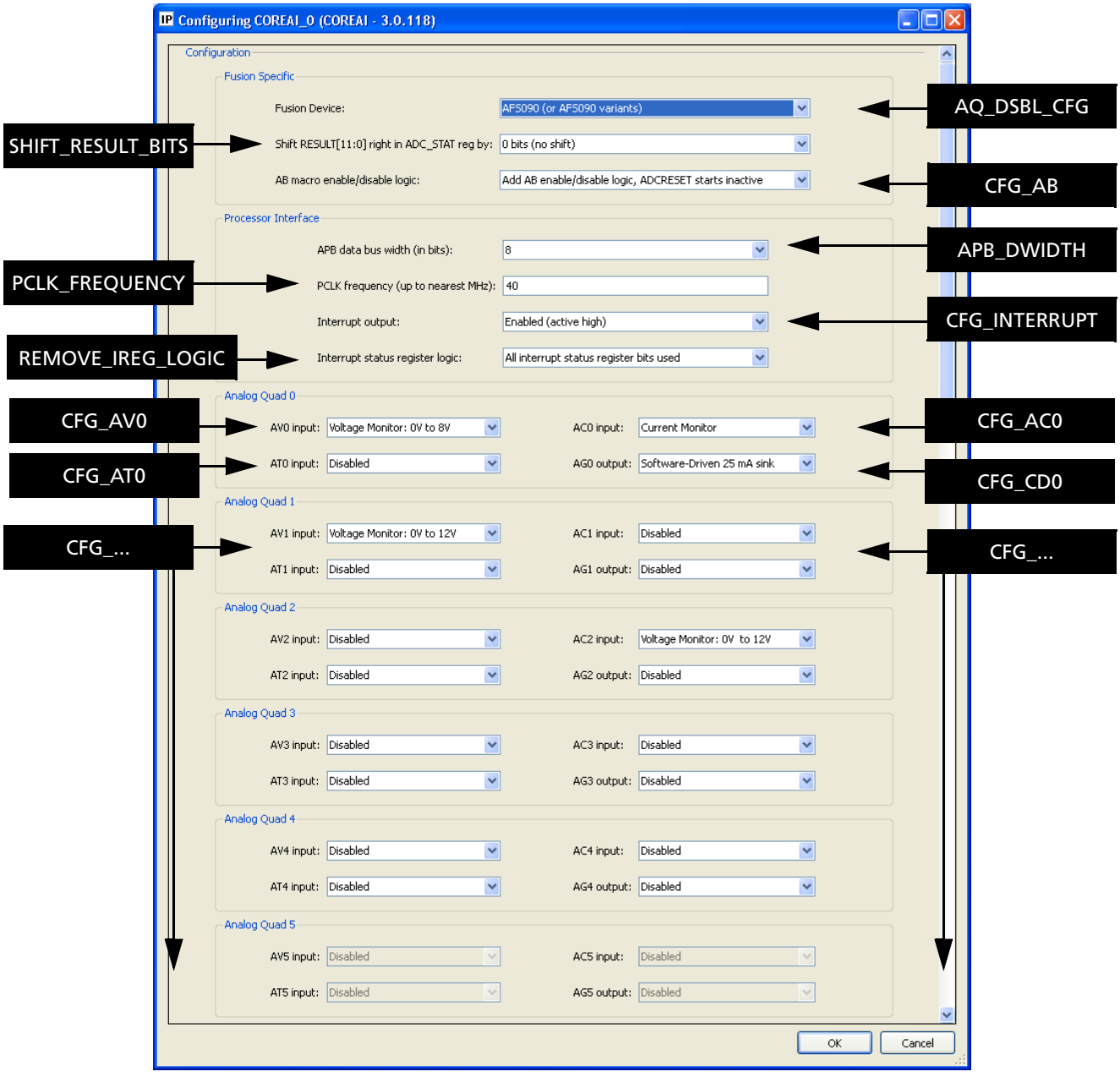

Figure 2-1 · CoreAI Configuration within SmartDesign

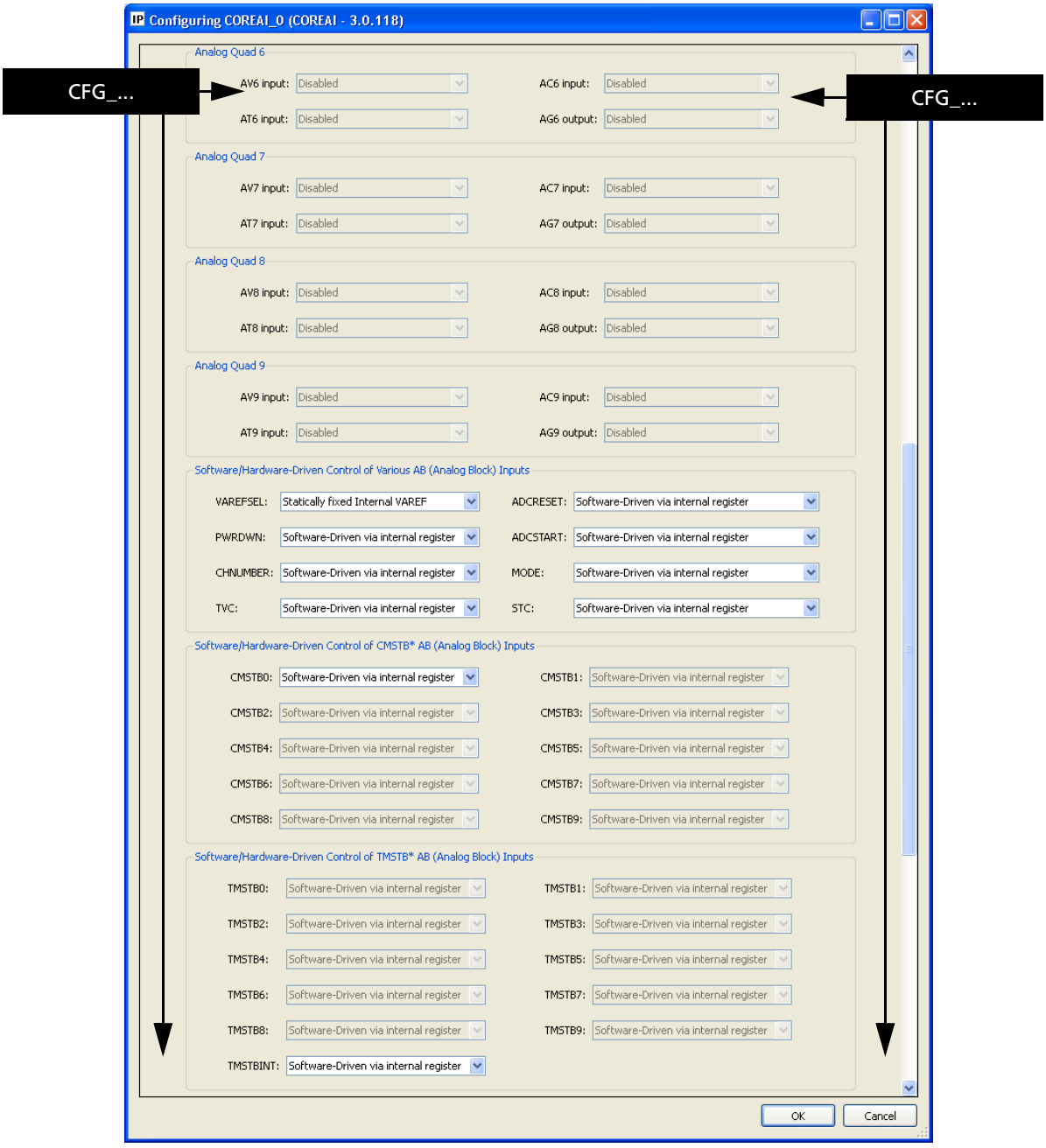

Figure 2-2 · CoreAI Configuration within SmartDesign (continued)

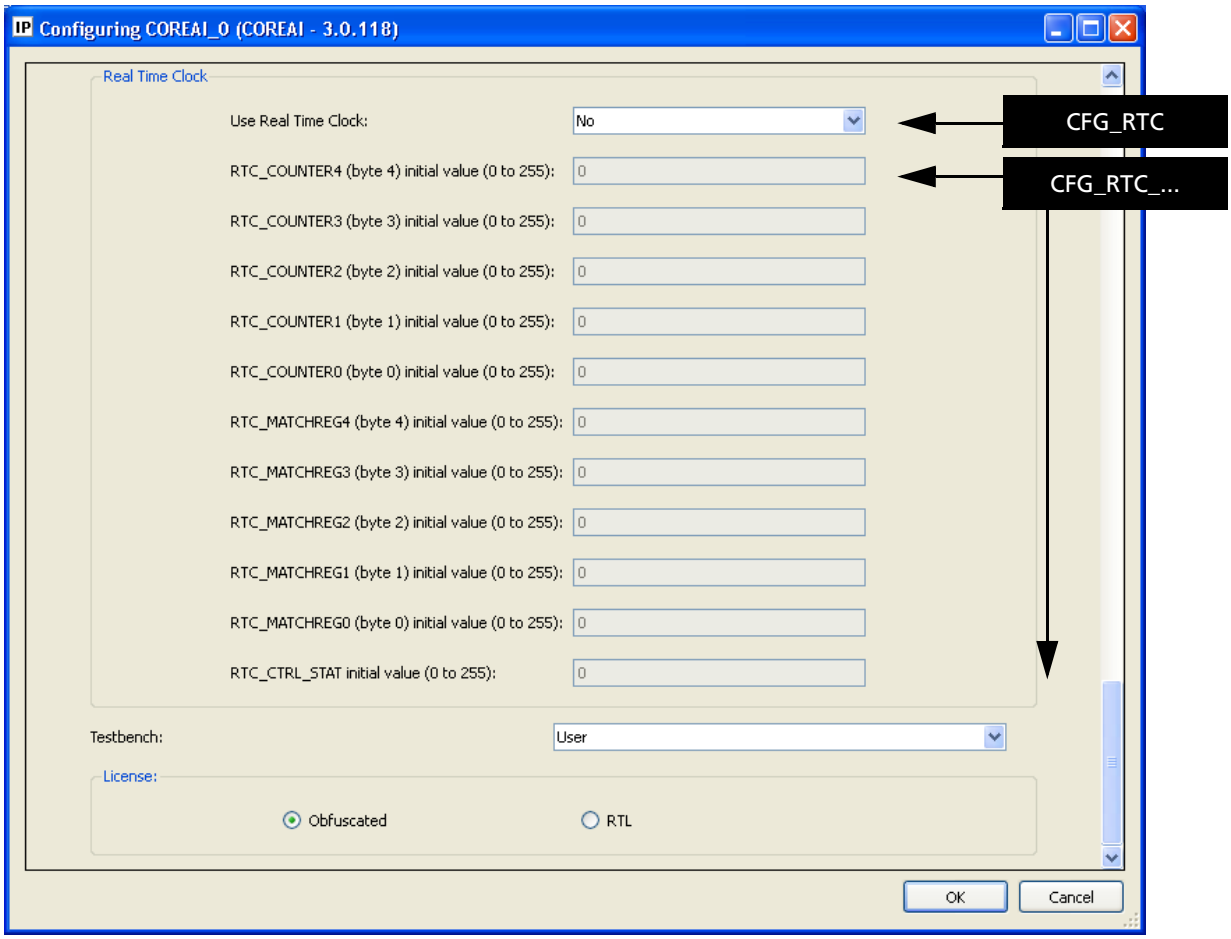

Figure 2-3 · CoreAI Configuration within SmartDesign (continued)

Simulation Flows

To run simulations, select the user testbench in SmartDesign through the CoreAI IP configuration GUI. Generate the design in SmartDesign. The appropriate test bench files are now installed.

To run the testbenches, set the design root to the CoreAI instantiation in the Libero IDE File Manager and click the Simulation icon in the Project Flow tab. This invokes Model*Sim*® and automatically runs the simulation.

Synthesis in Libero IDE

To run Synthesis on the core with parameters set in SmartDesign, set the design root appropriately and click the Synthesis icon in the Project Manager. The Synthesis window appears, displaying the Synplicity project. Set Synplicity to use the Verilog 2001 standard if Verilog is being used. To perform synthesis, click the Run icon.

Place-and-Route in Libero IDE

After setting the design root appropriately and running Synthesis, click the **Layout** icon in the Project Manager to invoke Designer. CoreAI requires no special place-and-route settings.

3 Interface Description

CoreAI is available with an APB slave interface and is easily connected to an APB bus on the SmartDesign Canvas.

Parameters/Generics

CoreAI has parameters (Verilog) and generics (VHDL), described in Table 3-1. All parameters and generics are integer types.

Table 3-1 · CoreAI Parameter/Generic Description

Name	Valid Range	Description
FAMILY	17	Only the Fusion family is supported for this core.
AQ_DSBL_CFG	0 to 2	Disable Analog Quads based on Fusion device 0 - Disable use of Analog Quads 5 to 9 (AFS090 or AFS090 variants) 1 - Disable use of Analog Quads 6 to 9 (AFS250 or AFS250 variants) 2 - Do not disable use of any of the Analog Quads from 0 to 9 (default, used with devices larger than the AFS250)
APB_DWIDTH	8, 16, or 32	Set this to the width of the APB data bus. Default is 8-bit width.
PCLK_FREQUENCY	1 to 100	Set this to the operating frequency in MHz of the PCLK input to the nearest integer, rounded up. CoreAI needs to know this information to set the internal clock divider correctly to communicate with the ACM interface within the AB macro. For example, if the actual PCLK input frequency is 19.15 MHz, PCLK_FREQUENCY should be set to 20; if the actual PCLK input frequency is exactly 19.0 MHz, PCLK_FREQUENCY should be set to 19.
SHIFT_RESULT_BITS	0, 2, or 4	This parameter/generic shifts the lower 12 bits in the ADC Status Register 11:0 bits 0 - ADC Status Register 11:0 connected to AB macro outputs RESULT[11:0] (default, not shifted) 2 - ADC Status Register bits 11:10 tied low and bits 9:0 connected to AB macro outputs RESULT[11:2] 4 - ADC Status Register bits 11:8 tied low and bits 7:0 connected to AB macro outputs RESULT[11:4]
CFG_AB	0, 1, or 2	AB Macro Enable Configuration 0 - Legacy mode - no clock enable or associated logic for AB macro 1 - Add clock enable logic for AB macro, ADCRESET starts inactive (default) 2 - Add clock enable logic for AB macro, ADCRESET starts active
CFG_INTERRUPT	0 to 3	Interrupt Output Configuration 0 - Interrupt Active-high (default) 1 - Interrupt Active-low 2 - Interrupt disabled (statically tied low) 3 - Interrupt disabled (statically tied high)

Table 3-1 · CoreAI Parameter/Generic Description (continued)

Name	Valid Range	Description
REMOVE_IREG_LOGIC	0 or 1	Remove interrupt register logic 0 - All bits of the Interrupt Status Register are used (default) 1 - bits 6, 3:0 of the Interrupt Status Register are tied low
CFG_ADCRESET	0 or 1	ADC Reset Configuration 0 - Software-controlled by internal register (default) 1 - Hardware-controlled by HD_ADCRESET input Note that if the ADCRESET is configured as being hardware-controlled by the HD_ADCRESET input, it is the user's responsibility to activate and de-activate the ADC reset condition to the AB block, whereas the de-activation of the software-controlled reset condition is automatically handled within CoreAI (Refer to Table 10: ADC Control Register 1 for description of the software-controlled ADCRESET bit).
CFG_PWRDWN	0 or 1	ADC Power Down Configuration 0 - Software-controlled by internal register (default) 1 - Hardware-controlled by HD_PWRDWN input When hardware-controlled, the ADC will be powered up normally if the HD_PWRDWN input is logic 0 and will be powered down if the HD_PWRDWN input is logic 1.
CFG_VAREFSEL	0 to 3	ADC Reference Voltage Selection Configuration 0 - Software-controlled by internal register (default) 1 - Hardware-controlled by HD_VAREFSEL input 2 - Statically fixed at logic 0 to select internal VAREF as output 3 - Statically fixed at logic 1 to select external VAREF as input (Refer to Table 10: ADC Control Register 1 for description of the software-controlled VAREFSEL bit)

Table 3-1 · CoreAI Parameter/Generic Description (continued)

Name	Valid Range	Description
CFG_MODE	0, 16, 32 to 34, 36 to 38, 40 to 42, 44 to 46	ADC Mode Selection Configuration (hexadecimal values shown) This parameter/generic controls the connection to the MODE[3:0] input pins of the AB macro (controls ADC resolution, etc.). 0x00 - Software-controlled by internal register (default) 0x10 - Hardware-controlled by HD_MODE[3:0] inputs 0x20 - Statically fixed to 0x0 (ADC 10-bit mode) 0x21 - Statically fixed to 0x1 (ADC 12-bit mode) 0x22 - Statically fixed to 0x2 (ADC 8-bit mode) 0x24 - Statically fixed to 0x4 (ADC 10-bit mode without internal power-down after conversion) 0x25 - Statically fixed to 0x5 (ADC 12-bit mode without internal power-down after conversion) 0x26 - Statically fixed to 0x6 (ADC 8-bit mode without internal power-down after conversion) 0x28 - Statically fixed to 0x8 (ADC 10-bit mode without internal calibration) 0x29 - Statically fixed to 0x9 (ADC 12-bit mode without internal calibration) 0x2A - Statically fixed to 0xA (ADC 8-bit mode without internal calibration) 0x2C - Statically fixed to 0xC (ADC 10-bit mode without internal calibration and without internal power-down after conversion) 0x2D - Statically fixed to 0xD (ADC 12-bit mode without internal calibration and without internal power-down after conversion) 0x2E - Statically fixed to 0xE (ADC 8-bit mode without internal calibration and without internal power-down after conversion)
CFG_TVC	0, 256, 512 to 767	ADC Clock Divider Configuration (hexadecimal values shown) This parameter/generic controls the connection to the TVC[7:0] input pins of the AB macro (controls ADC internal clock-divider that divides the PCLK frequency to generate the internal ADC clock). 0x000 - Software-controlled by internal register (default) 0x100 - Hardware-controlled by HD_TVC[7:0] inputs 0x200 - Statically fixed to 0x00 (PCLK/4) 0x201 - Statically fixed to 0x01 (PCLK/8) ... 0x2FE - Statically fixed to 0xFE (PCLK/1020) 0x2FF - Statically fixed to 0xFF (PCLK/1024)

Table 3-1 · CoreAI Parameter/Generic Description (continued)

Name	Valid Range	Description
CFG_STC	0, 256, 512 to 767	ADC Sample Time Control Configuration (hexadecimal values shown) This parameter/generic controls the connection to the STC[7:0] input pins of the AB macro (controls ADC sample time control). 0x000 - Software-controlled by internal register (default) 0x100 - Hardware-controlled by HD_STC[7:0] inputs 0x200 - Statically fixed to 0x00 (2 ADC clock periods) 0x201 - Statically fixed to 0x01 (3 ADC clock periods) ... 0x2FE - Statically fixed to 0xFE (256 ADC clock periods) 0x2FF - Statically fixed to 0xFF (257 ADC clock periods)
CFG_ADCSTART	0 or 1	ADC Start Conversion Configuration 0 - Software-controlled by internal register (default) 1 - Hardware-controlled by HD_ADCSTART input
CFG_CHNUMBER	0 or 1	ADC Channel Number Control Configuration 0 - Software-controlled by internal registers (default) 1 - Hardware-controlled by HD_CHNUMBER[4:0] inputs
CFG_CMSTB9, CFG_CMSTB8, CFG_CMSTB7, CFG_CMSTB6, CFG_CMSTB5, CFG_CMSTB4, CFG_CMSTB3, CFG_CMSTB2, CFG_CMSTB1, CFG_CMSTB0	0 or 1	Current Monitor Strobes Configuration These parameters/generics control the connection to the CMSTB9 down to CMSTB0 input pins, respectively, of the AB macro (control current monitor strobes). 0 - Software-controlled by internal register (default) 1 - Hardware-controlled by HD_CMSTBx input For example, if CFG_CMSTB3 is set to 1, the HD_CMSTB3 input will control the CMSTB3 input pin of the AB macro, instead of software-controlled writes to bit 3 of ADC Control Register 3.
CFG_TMSTBINT	0 to 2	Internal Temperature Monitor Strobe Configuration This parameter/generic controls the connection to the TMSTBINT input pin of the AB macro (controls internal temperature monitor strobe). 0 - Software-controlled by internal register (default) 1 - Hardware-controlled by HD_TMSTBINT input 2 - Disabled (tied to logic 0 internally)

Table 3-1 · CoreAI Parameter/Generic Description (continued)

Name	Valid Range	Description
CFG_TMSTB9, CFG_TMSTB8, CFG_TMSTB7, CFG_TMSTB6, CFG_TMSTB5, CFG_TMSTB4, CFG_TMSTB3, CFG_TMSTB2, CFG_TMSTB1, CFG_TMSTB0	0 or 1	Temperature Monitor Strobes Configuration These parameters/generics control the connection to the TMSTB9 down to TMSTB0 input pins, respectively, of the AB macro (control temperature monitor strobes). 0 - Software-controlled by internal register (default) 1 - Hardware-controlled by HD_TMSTBx input For example, if CFG_TMSTB6 is set to 1, the HD_TMSTB6 input will control the TMSTB6 input pin of the AB macro, instead of software-controlled writes to bit 6 of ADC Control Register 4.
CFG_AV9,CFG_AV8, CFG_AV7,CFG_AV6, CFG_AV5,CFG_AV4, CFG_AV3,CFG_AV2, CFG_AV1,CFG_AV0	0 to 1,023	Configure AVx Inputs (hexadecimal values shown) Each of these parameters/generics is used to configure the AV9 down to AV0 inputs, respectively, connected to the AB macro. The lower 8 bits of each of these 10-bit integers are reserved for generating firmware settings for the target processor used with CoreAI and are ignored by the CoreAI hardware. The upper 2 bits are used to create the settings for each AVx input. 0x000 to 0x0FF - Voltage Monitor (default) 0x100 to 0x1FF - Digital Input 0x200 to 0x2FF - Reserved (unused) 0x300 to 0x3FF - Disabled For example, if CFG_AV5 is set to the value 0x000, the AV5 input will be used as an analog voltage monitor input. If CFG_AV5 is set to the value 0x100, the AV5 input will be used as a digital input, and the buffered DAVOUT5 output would be connected to the user's own logic. If CFG_AV5 is set to the value 0x300, the AV5 input will be disabled; in this case, the CoreAI AV5 input will not be used and the AV5 input of the AB macro will be hardwired to logic 0 within CoreAI.
CFG_AC9,CFG_AC8, CFG_AC7,CFG_AC6, CFG_AC5,CFG_AC4, CFG_AC3,CFG_AC2, CFG_AC1,CFG_AC0	0 to 1,023	Configure ACx Inputs (hexadecimal values shown) Each of these parameters/generics is used to configure the AC9 down to AC0 inputs, respectively, connected to the AB macro. The lower 8 bits of each of these 10-bit integers are reserved for generating firmware settings for the target processor used with CoreAI and are ignored by the CoreAI hardware. The upper 2 bits are used to create the settings for each ACx input. 0x000 to 0x0FF - Current Monitor (default) 0x100 to 0x1FF - Digital Input 0x200 to 0x2FF - Voltage Monitor 0x300 to 0x3FF - Disabled For example, if CFG_AC7 is set to the value 0x000, the AC7 input will be used as an analog current monitor input. If CFG_AC7 is set to the value 0x100, the AC7 input will be used as a digital input, and the buffered DACOUT7 output would be connected to the user's own logic. If CFG_AC7 is set to the value 0x200, the AC7 input will be used as an analog voltage monitor input. If CFG_AC7 is set to the value 0x300, the AC7 input will be disabled; in this case, the CoreAI AC7 input will not be used and the AC7 input of the AB macro will be hardwired to logic 0 within CoreAI.

Table 3-1 · CoreAI Parameter/Generic Description (continued)

Name	Valid Range	Description
CFG_AT9,CFG_AT8, CFG_AT7,CFG_AT6, CFG_AT5,CFG_AT4, CFG_AT3,CFG_AT2, CFG_AT1,CFG_AT0	0 to 1,023	Configure ATx Inputs (hexadecimal values shown) Each of these parameters/generics is used to configure the AT9 down to AT0 inputs, respectively, connected to the AB macro. The lower 8 bits of each of these 10-bit integers are reserved for generating firmware settings for the target processor used with CoreAI and are ignored by the CoreAI hardware. The upper 2 bits are used to create the settings for each ATx input. 0x000 to 0x0FF - Temperature Monitor (default) 0x100 to 0x1FF - Digital Input 0x200 to 0x2FF - Voltage Monitor 0x300 to 0x3FF - Disabled For example, if CFG_AT2 is set to the value 0x000, the AT2 input will be used as an analog temperature monitor input. If CFG_AT2 is set to the value 0x100, the AT2 input will be used as a digital input, and the buffered DATOUT2 output would be connected to the user's own logic. If CFG_AT2 is set to the value 0x200, the AT2 input will be used as an analog voltage monitor input. If CFG_AT2 is set to the value 0x300, the AT2 input will be disabled; in this case, the CoreAI AT2 input will not be used and the AT2 input of the AB macro will be hardwired to logic 0 within CoreAI.
CFG_GD9,CFG_GD8, CFG_GD7,CFG_GD6, CFG_GD5,CFG_GD4,CFG _GD3,CFG_GD2, CFG_GD1,CFG_GD0	0 to 1,023	Configure GDONx Gate Driver Control Inputs and AGx Gate Driver Outputs (hexadecimal values shown) Each of these parameters/generics is used to configure the GDON9 down to GDON0 inputs, respectively, connected to the AB macro, and the AG9 down to AG0 outputs, respectively, that are connected to the AB macro. The lower 8 bits of each of these 10-bit integers are reserved for generating firmware settings for the target processor used with CoreAI and are ignored by the CoreAI hardware. The upper 2 bits are used to create the settings for each GDONx input and AGx output. 0x000 to 0x0FF - GDONx/AGx Software-controlled by internal register (default) 0x100 to 0x1FF - GDONx/AGx Hardware-controlled by HD_GDONx input 0x200 to 0x2FF - Reserved (unused) 0x300 to 0x3FF - Disabled (GDONx and AGx) For example, if CFG_GD1 is set to the value 0x00, the GDON1 input and AG1 gate-driver output will be controlled by a software-controlled register within CoreAI (ADC Control Register 5). If CFG_GD1 is set to the value 0x100, the HD_GDON1 input (connected to the user's own logic) will be used to directly control the GDON1 input and AG1 gate-driver output. If CFG_GD1 is set to the value 0x300, the GDON1 input and AG1 output will be disabled; in this case, the CoreAI AG1 output will not be used and the GDON1 input of the AB macro will be hardwired to logic 0 within CoreAI.

Table 3-1 · CoreAI Parameter/Generic Description (continued)

Name	Valid Range	Description
CFG_RTC	0 to 2	Configure use of RTC This parameter/generic controls whether or not the RTC is used 0 - Not used (default) 1 - Used, the RTCPSMMATCH output is exposed on the SmartDesign canvas. You must connect a Crystal Oscillator component and a Voltage Regulator component to your design from the Libero IDE Catalog in the Fusion Peripherals category. 2 - Used, the RTCPSMMATCH output is not exposed on the SmartDesign canvas. You must connect a Crystal Oscillator component to your design from the Libero IDE Catalog in the Fusion Peripherals category. Consult the Libero IDE online help for information on how to connect Fusion peripherals to your design on the SmartDesign canvas.
CFG_RTC_COUNTER4, CFG_RTC_COUNTER3, CFG_RTC_COUNTER2, CFG_RTC_COUNTER1, CFG_RTC_COUNTER0	0 to 255	Configure ACM RTC Counter Registers Each of these parameters/generics is used to configure the RTC COUNTER4 down to COUNTER0 ACM registers, respectively, within the AB macro (see Table 18). Each of these 8-bit integers are reserved for generating firmware settings for the target processor used with CoreAI and are ignored by the CoreAI hardware.
CFG_RTC_MATCHREG4, CFG_RTC_MATCHREG3, CFG_RTC_MATCHREG2, CFG_RTC_MATCHREG1, CFG_RTC_MATCHREG0	0 to 255	Configure ACM RTC Match Registers Each of these parameters/generics is used to configure the RTC MATCHREG4 down to MATCHREG0 ACM registers, respectively, within the AB macro (see Table 18). Each of these 8-bit integers are reserved for generating firmware settings for the target processor used with CoreAI and are ignored by the CoreAI hardware.
CFG_RTC_CTRL_STAT	0 to 255	Configure ACM RTC Control/Status Register This parameter/generic is used to configure the RTC CTRL_STAT ACM register within the AB macro (see Table 18). This 8-bit integer is reserved for generating firmware settings for the target processor used with CoreAI and is ignored by the CoreAI hardware.

Parameter Dependencies and Precedence

Some of the parameter settings can be ignored or used depending on the values of other parameters. The effect of this can be seen when using the IP configurator GUI in SmartDesign: you will see that some of the fields are grayed out. This graying out behavior prevents some parameter values being used based on settings of other parameters. For example, if you are using an AFS090 device, and select this in the IP configurator, you will see that the configuration dialogs are grayed out for analog quads 5 through 9, since an AFS090 device only has analog quads from 0 to 4.

The parameter AQ_DSBL_CFG has higher precedence than the CFG_AVx, CFG_ACx, CFG_ATx, and CFG_GDx parameters. Also, the CFG_ACx parameters have higher precedence than the CFG_CMSTBx parameters, and the CFG_ATx parameters have higher precedence than the CFG_TMSTBx parameters.

For example, if the AQ_DSBL_CFG parameter has the value 0 (AFS090 device), then the CFG_AV5 to CFG_AV9 parameter settings, the CFG_AC5 to CFG_AC9 parameter settings, the CFG_AT5 to CFG_AT9 parameter settings, and the CFG_GD5 to CFG_GD9 parameter settings are ignored and effectively disabled, resulting in various IP configurator dialogs being grayed out. Likewise, since the CFG_AC5 to CFG_AC9 and CFG_AT5 to CFG_AT9 parameters are disabled in this case, this means the CFG_CMSTB5 to CFG_CMSTB9 and CFG_TMSTB5 to CFG_TMSTB9 parameters will also be disabled, since they are of lower precedence.

Ports

The port signals for the CoreAI macro are defined in Table 3-2 on page 24 and illustrated in Figure 3-1. CoreAI has from 194 to 243 I/O signals, depending on the APB_DWIDTH parameter.

Note that vector notation is used in Figure 3-1 for the AV, AC, AT, ATRETURN, HD_GDON, HD_CMSTB, HD_TMSTB, DAVOUT, DACOUT, DATOUT, and AG ports; however, these ports are actually split into individual single-bit ports, as described in Table 3-2. For example, there are five individual input ports, ATRETURN4, ATRETURN3, ATRETURN2, ATRETURN1, and ATRETURN0, rather than one vectored input port ATRETURN[4:0].

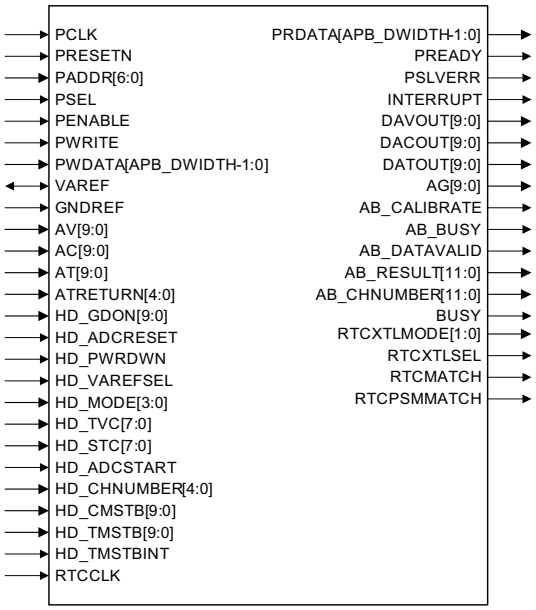

Figure 3-1 · CoreAI I/O Signal Diagram

Table 3-2 · CoreAI Signal Description

Name	Type	Description
APB Interface		
PCLK	Input	APB System Clock: reference clock for all internal logic. The frequency of this input clock signal must be set appropriately using the PCLK_FREQUENCY parameter/ generic in Table 3.
PRESETN	Input	APB active-low asynchronous reset
PADDR[6:0]	Input	APB address bus - This port is used to address internal CoreAI registers.
PSEL	Input	APB Slave Select - This signal selects CoreAI for reads or writes.
PENABLE	Input	APB Strobe - This signal indicates the second cycle of an APB transfer.
PWRITE	Input	APB Write/Read - If high, a write will occur when an APB transfer to CoreAI takes place; if low, a read from CoreAI will take place.

Table 3-2 · CoreAI Signal Description

Name	Type	Description
PWDATA[APB_DWIDTH-1:0]	Input	APB write data - This can be 8, 16, or 32 bits wide, depending on the setting of the APB_DWIDTH parameter.
PRDATA[APB_DWIDTH-1:0]	Output	APB read data - This can be 8, 16, or 32 bits wide, depending on the setting of the APB_DWIDTH parameter.
PREADY	Output	APB ready - This signal is not used by CoreAI and is tied high internally.
PSLVERR	Output	APB slave error - This signal is not used by CoreAI and is tied low internally.
INTERRUPT	Output	Microprocessor interrupt output - This interrupt signal is generated from 6 possible interrupt sources, each of which can be masked or enabled via the INTENABLE register. The polarity of this output is controlled via the CFG_INTERRUPT parameter/generic.
Analog Interface		
VAREF	Input or Output	Voltage reference - If using the internal voltage reference, this signal will be an output; if using an external voltage reference, this signal will be an input for this reference (see the CFG_VAREFSEL parameter/generic in Table 3-1 on page 17). This signal must always be brought to the top level of your design.
GNDREF	Input	Ground reference - If external voltage reference is used, this signal must be connected to the ground for the reference; otherwise this should be connected to digital ground (logic 0).
AV9,AV8,AV7,AV6,AV5,AV4,AV3,AV2,AV1,AV0	Input	Analog Voltage Monitor inputs - These signals correspond to the AVx voltage monitor inputs (AV9 through AV0) of the AB macro. Note: Unused AVx inputs need to be disabled with the CFG_AVx parameters/generics and connected to logic 0. Note that the setting of the AQ_DSBL_CFG parameter/generic affects which of these ports are available for use, depending on which Analog Quads are available (available Analog Quads vary for the smaller dies AFS090 and AFS250).
AC9,AC8,AC7,AC6,AC5,AC4,AC3,AC2,AC1,AC0	Input	Analog Current Monitor inputs - These signals correspond to the ACx current monitor inputs (AC9 through AC0) of the AB macro. Note: Unused ACx inputs need to be disabled with the CFG_ACx parameters/generics and connected to logic 0. Note that the setting of the AQ_DSBL_CFG parameter/generic affects which of these ports are available for use, depending on which Analog Quads are available (available Analog Quads vary for the smaller dies AFS090 and AFS250).
AT9,AT8,AT7, AT6,AT5, AT4,AT3, AT2,AT1,AT0	Input	Analog Temperature Monitor inputs - These signals correspond to the ATx temperature monitor inputs (AT9 through AT0) of the AB macro. Note: Unused ATx inputs need to be disabled with the CFG_ATx parameters/generics and connected to logic 0. Note that the setting of the AQ_DSBL_CFG parameter/generic affects which of these ports are available for use, depending on which Analog Quads are available (available Analog Quads vary for the smaller dies AFS090 and AFS250).
ATRETURN4,ATRETURN3,ATRETURN2,ATRETURN1, ATRETURN0	Input	Shared Analog Temperature Monitor Returns - These signals correspond to the shared returns for the temperature monitor inputs ATRETURN89 down to ATRETURN01, respectively, of the AB macro. Note that the setting of the AQ_DSBL_CFG parameter/generic affects which of these ports are available for use, depending on which Analog Quads are available (available Analog Quads vary for the smaller dies AFS090 and AFS250).

Table 3-2 · CoreAI Signal Description

Name	Type	Description
DAVOUT9,DAVOUT8, DAVOUT7,DAVOUT6, DAVOUT5,DAVOUT4, DAVOUT3,DAVOUT2, DAVOUT1,DAVOUT0	Output	Digital AV outputs - These signals correspond to the digital AV outputs (DAVOUT9 through DAVOUT0) of the AB macro. If any of the AVx inputs are configured as digital inputs rather than analog inputs, their corresponding buffered digital signals are put out on these ports.
DACOUT9,DACOUT8, DACOUT7,DACOUT6, DACOUT5,DACOUT4, DACOUT3,DACOUT2, DACOUT1,DACOUT0	Output	Digital AC outputs - These signals correspond to the digital AC outputs (DACOUT9 through DACOUT0) of the AB macro. If any of the ACx inputs are configured as digital inputs rather than analog inputs, their corresponding buffered digital signals are put out on these ports.
DATOUT9,DATOUT8, DATOUT7,DATOUT6, DATOUT5,DATOUT4, DATOUT3,DATOUT2, DATOUT1,DATOUT0	Output	Digital AT outputs - These signals correspond to the digital AT outputs (DATOUT9 through DATOUT0) of the AB macro. If any of the ATx inputs are configured as digital inputs rather than analog inputs, their corresponding buffered digital signals are put out on these ports.
HD_GDON9, HD_GDON8, HD_GDON7, HD_GDON6, HD_GDON5, HD_GDON4, HD_GDON3, HD_GDON2, HD_GDON1, HD_GDON0	Input	Hardware-Driven Gate Driver enables - These signals can control the corresponding GDONx gate-driver enable inputs (GDON9 through GDON0) of the AB macro if the CFG_GDx parameters/generics are set appropriately (refer to Parameters/Generics section). If unused, these inputs should be tied off to static logic 0 or logic 1 values.
AG9,AG8,AG7AG6, AG5,AG4,AG3,AG2, AG1,AG0	Output	Analog Gate Driver outputs - These signals correspond to the AGx gate driver outputs (AG9 through AG0) of the AB macro. Note: If unused, each of these gate driver outputs can be disabled via the CFG_GDx parameters/generics. Note that the setting of the AQ_DSBL_CFG parameter/generic affects which of these ports are available for use, depending upon which Analog Quads are available, which vary for the smaller die (AFS090 and AFS250).
HD_ADCRESET	Input	Hardware-Driven ADC Reset - This signal can optionally control the ADCRESET input of the AB macro if the CFG_ADCRESET parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, this input should be tied off to static logic 0 or logic 1.
HD_PWRDWN	Input	Hardware-Driven ADC Power Down - This signal can optionally control the PWRDWN input of the AB macro if the CFG_PWRDWN parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, this input should be tied off to static logic 0 or logic 1.
HD_VAREFSEL	Input	Hardware-Driven ADC Voltage Reference Select - This signal can optionally control the VAREFSEL input of the AB macro if the CFG_VAREFSEL parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, this input should be tied off to static logic 0 or logic 1.
HD_MODE[3:0]	Input	Hardware-Driven ADC Mode Control - These signals can optionally control the MODE[3:0] inputs of the AB macro if the CFG_MODE parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, these inputs should be tied off to static logic 0 or logic 1 values.

Table 3-2 · CoreAI Signal Description

Name	Type	Description
HD_TVC[7:0]	Input	Hardware-Driven ADC Clock Divider Settings - These signals can optionally control the TVC[7:0] inputs of the AB macro if the CFG_TVC parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, these inputs should be tied off to static logic 0 or logic 1 values.
HD_STC[7:0]	Input	Hardware-Driven ADC Sample Time Control - These signals can optionally control the STC[7:0] inputs of the AB macro if the CFG_STC parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, these inputs should be tied off to static logic 0 or logic 1 values.
HD_ADCSTART	Input	Hardware-Driven ADC Start of Conversion - This signal can optionally control the ADCSTART input of the AB macro if the CFG_ADCSTART parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, this input should be tied off to static logic 0 or logic 1.
HD_CHNUMBER[4:0]	Input	Hardware-Driven ADC Channel Number Control - These signals can optionally control the CHNUMBER[4:0] inputs of the AB macro if the CFG_CHNUMBER parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, these inputs should be tied off to static logic 0 or logic 1 values.
HD_CMSTB9, HD_CMSTB8, HD_CMSTB7, HD_CMSTB6, HD_CMSTB5, HD_CMSTB4, HD_CMSTB3, HD_CMSTB2, HD_CMSTB1, HD_CMSTB0	Input	Hardware-Driven Current Monitor Strobes - These signals can optionally control the CMSTB9 down to CMSTB0 inputs of the AB macro, respectively, if the CFG_CHNUMBER parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, these inputs should be tied off to static logic 0 or logic 1 values.
HD_TMSTBINT	Input	Hardware-Driven Internal Temperature Monitor Strobe - This signal can optionally control the TMSTBINT input of the AB macro if the CFG_TMSTBINT parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, this input should be tied off to static logic 0 or logic 1.
HD_TMSTB9, HD_TMSTB8, HD_TMSTB7, HD_TMSTB6, HD_TMSTB5, HD_TMSTB4, HD_TMSTB3, HD_TMSTB2, HD_TMSTB1, HD_TMSTB0	Input	Hardware-Driven Temperature Monitor Strobes - These signals can optionally control the TMSTB9 down to TMSTB0 inputs of the AB macro, respectively, if the CFG_CHNUMBER parameter/generic is set appropriately (refer to Parameters/Generics section). If unused, these inputs should be tied off to static logic 0 or logic 1 values.
AB_CALIBRATE	Output	CALIBRATE Output from AB Macro - This direct output from the AB macro will be high during calibration after the ADC is reset. If unused, this port should be left unconnected.
AB_BUSY	Output	BUSY Output from AB Macro - This direct output from the AB macro will be high during an ADC conversion. If unused, this port should be left unconnected.
AB_DATAVALID	Output	DATAVALID Output from AB Macro - This direct output from the AB macro will be high after an ADC conversion has finished. If unused, this port should be left unconnected.
AB_RESULT[11:0]	Output	RESULT[11:0] Outputs from AB Macro - These direct outputs from the AB macro will contain the digital conversion data after an ADC conversion has finished. If unused, these ports should be left unconnected.

Table 3-2 · CoreAI Signal Description

Name	Type	Description
AB_CHNUMBER[4:0]	Output	CHNUMBER[4:0] Inputs to AB Macro - These signals to the AB macro contain the channel number for the current ADC conversion. These signals may be hardware-controlled or software-controlled depending on the setting of the CFG_CHNUMBER parameter/generic (refer to Parameters/Generics section). If unused, these ports should be left unconnected.
BUSY	Output	CoreAI Busy Output - This output will be high during any of the following events: ADC calibration, ADC busy converting data, CoreAI is performing a write procedure to the ACM, or CoreAI is performing a read procedure from the ACM.
RTC Interface		
RTCCLK	Input	RTC Clock input - If the RTC is used (via the CFG_RTC parameter/generic), this input must come from the internal crystal oscillator (XTLOSC) CLKOUT pin; if the RTC is not used, this pin should be tied low.
RTCXTLMODE[1:0]	Output	RTC XTLOSC Mode outputs - If the RTC is used (via the CFG_RTC parameter/ generic), these output ports must be connected to the internal crystal oscillator (XTLOSC) RTCMODE[1:0] pins; if the RTC is not used, these pins should be left unconnected.
RTCXTLSEL	Output	RTC XTLOSC Mode Selection output - If the RTC is used (via the CFG_RTC parameter/ generic), this output port must be connected to the internal crystal oscillator (XTLOSC) MODESEL pin; if the RTC is not used, this pin should be left unconnected.
RTCMATCH	Output	RTC Match output - If the RTC is used (via the CFG_RTC parameter/generic), this output port indicates that a match event has occurred and can be connected to other FPGA logic; if the RTC is not used, this pin should be left unconnected.
RTCPSMMATCH	Output	RTC Match output - If you use the RTC (via the CFT_RTC parameter/generic), this output port mimics the behavior of the RTCMATCH output, but can be used to connect to the VRPSM pin of the on-chip voltage regulator to optionally turn it on when a match occurs.

Analog Interfaces

CoreAI provides full access to the Fusion Analog interface; this interface is fully described in the Fusion datasheet, available on the Actel website.

4 Register Maps

APB Register Map

The internal register address map and reset values of each APB-accessible register for CoreAI are shown in Table 4-1 for 16-bit APB reads and writes (APB_DWIDTH=16), and in Table 4-2 for 8-bit APB reads and writes (APB_DWIDTH=8). Note that if the APB_DWIDTH parameter/generic is set to 32, the results are the same as for 16-bit mode, with the upper 16 bits of PWDATA[31:0] ignored on writes, and the upper 16 bits of PRDATA[31:0] returning logic 0 on reads.

Table 4-3 on page 31 through Table 4-13 on page 36 describes the various APB-accessible registers within CoreAI. Unless otherwise stated, each register can be read from or written to by an internal or external microprocessor/ microcontroller.

When reading from register bits that are write-only or unused (reserved), logic 0 will be returned. When writing to register bits that are read-only or unused (reserved), no action takes place.

The INTERRUPT output is generated as the logical OR of the interrupt enable bits (INTEN[6:0]) ANDed with the interrupt status bits (INT[6:0]), as shown in Figure 4-1.

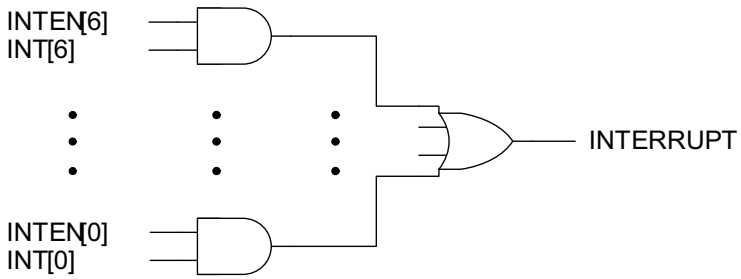

Figure 4-1 · CoreAI Interrupt Logic

Note: After an APB read of the interrupt status registers has been performed, each INT[6:0] bit will be cleared if its condition is no longer valid. If an APB read of the interrupt status registers has just occurred coincident with a pending interrupt condition, the interrupt condition will have priority in order to prevent a missed interrupt.

Table 4-1 · CoreAI Internal Register Address Map (16- or 32-bit Mode, APB_DWIDTH=16 or 32)

PADDR[6:0]	Type	Reset Value	Description
0x00	R/W	0x0004	ACM Control/Status Register
0x04	R/W	0x0000	ACM Address Register
0x08	R/W	0x0000	ACM Data Register
0x0C	R/W	0x0000	ADC Control Register 1
0x10	R/W	0x0000	ADC Control Register 2
0x14	R/W	0x0000	ADC Control Register 3
0x18	R/W	0x0000	ADC Control Register 4
0x1C	R/W	0x0000	ADC Control Register 5
0x20	R	0x0000	ADC Status Register

Table 4-1 · CoreAI Internal Register Address Map (16- or 32-bit Mode, APB_DWIDTH=16 or 32)

PADDR[6:0]	Type	Reset Value	Description
0x2C	R/W	0x0000	Interrupt Enable Register
0x30	R	0x0000	Interrupt Status Register

Table notes: Values shown in hexadecimal format; type designations: R = read only; R/W = read/write

Table 4-2 · CoreAI Internal Register Address Map (8-bit Mode, APB_DWIDTH=8)

PADDR[6:0]	Type	Reset Value	Description
0x00	R/W	0x04	ACM Control/Status Register (low-order bits 7:0)
0x04	R/W	0x00	ACM Control/Status Register (high-order bits 15:8)
0x08	R/W	0x00	ACM Address Register
0x10	R/W	0x00	ACM Data Register
0x18	R/W	0x00	ADC Control Register 1 (low-order bits 7:0)
0x1C	R/W	0x00	ADC Control Register 1 (high-order bits 15:8)
0x20	R/W	0x00	ADC Control Register 2 (low-order bits 7:0)
0x24	R/W	0x00	ADC Control Register 2 (high-order bits 15:8)
0x28	R/W	0x00	ADC Control Register 3 (low-order bits 7:0)
0x2C	R/W	0x00	ADC Control Register 3 (high-order bits 15:8)
0x30	R/W	0x00	ADC Control Register 4 (low-order bits 7:0)
0x34	R/W	0x00	ADC Control Register 4 (high-order bits 15:8)
0x38	R/W	0x00	ADC Control Register 5 (low-order bits 7:0)
0x3C	R/W	0x00	ADC Control Register 5 (high-order bits 15:8)
0x40	R	0x00	ADC Status Register (low-order bits 7:0)
0x44	R	0x00	ADC Status Register (high-order bits 15:8)
0x58	R/W	0x00	Interrupt Enable Register
0x60	R	0x00	Interrupt Status Register

table notes: Values shown in hexadecimal format; type designations: R = read only; R/W = read/write

Table 4-3 · ACM Control/Status Register

Bits	Name	Function
15:5	Reserved	Not used
4	ACMWRBUSY	ACM Write Cycle Busy (read-only) If 1, the ACM is busy writing data into the AB block.
3	ACMRDBUSY	ACM Read Cycle Busy (read-only) If 1, the ACM is busy reading data from the AB block.
2	ACMRESETBUSY	ACM Reset Cycle Busy (read-only) If 1, the ACM is busy being reset.
1	ACMRDSTART	ACM Read Start (write-only) 1 - The ACM starts a read cycle from the ACM address in the ACMADDR[7:0] bits of the ACM Address/Data register. Note that this write-only bit is active for one ACM clock cycle (self-clearing), and that the resulting busy signal from the ACM read taking place will be reflected in the ACMRDBUSY bit of this register. 0 - Normal (no operation or current ACM operation continues)
0	ACMRESET	ACM Reset (write-only) 1 - The ACM is put into a reset condition. Note that this write-only bit is active for only one ACM clock cycle (self-clearing), and that the resulting busy signal from the ACM reset taking place will be reflected in the ACMRESETBUSY bit of this register. 0 - Normal (no operation or current ACM operation continues)

Table 4-4 · ACM Address Register

Bits	Name	Function
15:8	Reserved	Not used
7:0	ACMADDR	ACM Address These bits are connected to the ACMADDR[7:0] port of the AB macro. (Refer to the Fusion datasheet for further information.)

Table 4-5 · ACM Data Register

Bits	Name	Function
15:8	Reserved	Not used
7:0	ACMDATA	ACM Data If this register is read from, the ACMRDATA[7:0] output port from the AB block is returned. If this register is written to, it will drive the ACMWDATA[7:0] input port of the AB block.

Table 4-6 · ADC Control Register 1

Bits	Name	Function
15:8	TVC	ADC Clock Divider (Refer to the Fusion datasheet for further information.)
7	ABENABLE	AB Macro Clock Enable 1 - Enable SYSCLK pin of the AB macro 0 - Disable the SYSCLK pin of the AB macro (default) Note that the behavior of this register depends on the CFG_AB parameter/generic settings.
6	ADCRESET	ADC Reset (write-only) 1 - The ADC is given an active high pulse (connected to the ADCRESET pin of the AB macro). Note that this write-only bit is active for only one PCLK clock cycle (self-clearing). 0 - Normal (no operation or current conversion continues)
5	PWRDWN	ADC Power Down 1 - The ADC is powered down. 0 - The ADC is powered up (normal operation).
4	VAREFSEL	ADC Voltage Reference Select 1 - Select external voltage reference (3.3 V max.) to be used (input on VAREF and GNDREF ports) 0 - Select internal voltage reference (2.56 V) to be used (output on VAREF port)
3:0	MODE	ADC Mode Selection The mode selection bits are used to select between 8-, 10-, and 12-bit ADC resolution. (Refer to the Fusion datasheet for further information.)

Table 4-7 · ADC Control Register 2

Bits	Name	Function
15:14	Reserved	Not used
13	ADCSTART	ADC Start Conversion (write-only) 1 - The ADC starts an analog-to-digital conversion on the selected channel. Note that this write-only bit is high for only one PCLK clock cycle (self-clearing). 0 - Normal (no operation or current conversion continues)
12:8	CHNUMBER	ADC Channel Number This 5-bit value selects one of 32 analog channels that are fed to the analog MUX within the AB macro. (Refer to the Fusion datasheet for further information.)
7:0	STC	ADC Sample Time Control (Refer to the Fusion datasheet for further information.)

Table 4-8 · ADC Control Register 3

Bits	Name	Function
15:10	Reserved	Not used
9:0	CMSTB	Current Monitor Strobes These bits are connected to the CMSTB9 down to CMSTB0 pins, respectively, of the AB macro. (Refer to the Fusion datasheet for further information.)

Table 4-9 · ADC Control Register 4

Bits	Name	Function
15:9	Reserved	Not used
10	TMSTBINT	Internal Temperature Monitor Strobe This bit is connected to the TMSTBINT pin of the AB macro. (Refer to the Fusion datasheet for further information.)
9:0	TMSTB	Temperature Monitor Strobes These bits are connected to the TMSTB9 down to TMSTB0 pins, respectively, of the AB macro. (Refer to the Fusion datasheet for further information.)

Table 4-10 · ADC Control Register 5

Bits	Name	Function
15:10	Reserved	Not used
9:0	GDON	Gate Driver Enables These bits are connected to the GDON9 down to GDON0 pins, respectively, of the AB macro. Note that the CFG_GDx parameters/generics settings affect whether or not these register bits are used. (Refer to the Fusion datasheet for further information.)

Table 4-11 · ADC Status Register

Bits	Name	Function
15	CALIBRATE	ADC Calibrate 1 - The ADC is busy performing its calibration. 0 - The ADC is calibrated. (Refer to the Fusion datasheet for further information.)
14	SAMPLE	ADC Sample 1 - The ADC is sampling the selected analog input. 0 - Normal (no operation or the ADC has finished the sampling phase) (Refer to the Fusion datasheet for further information.)
13	BUSY	ADC Busy 1 - The ADC is busy performing an analog-to-digital conversion. 0 - The ADC is not busy. (Refer to the Fusion datasheet for further information.)

Table 4-11 · ADC Status Register

Bits	Name	Function
12	DATAVALID	ADC Data Valid 1 - The ADC contains valid data from an analog-to-digital conversion on the RESULT[11:0] outputs. 0 - Normal (no operation or current conversion continues) (Refer to the Fusion datasheet for further information.)
11:0	RESULT	ADC Result These bits come from the RESULT[11:0] bits of the AB macro. (Refer to the Fusion datasheet for further information.) Note that the actual assignments of these 12 status bits will change according to the value of the SHIFT_RESULT_BITS parameter/generic. If SHIFT_RESULT_BITS=0, these 12 bits will be the RESULT[11:0] bits from the AB macro (default); if SHIFT_RESULT_BITS=2, these 12 bits will be "00", RESULT[11:2] (suitable for reading results when the ADC is set for 10-bit resolution); if SHIFT_RESULT_BITS=4, these 12 bits will be "0000", RESULT[11:4] (suitable for reading results when the ADC is set for 8-bit resolution, and CoreAI is setup for 8-bit APB mode).

Table 4-12 · Interrupt Enable Register

Bits	Name	Function
15:7	Reserved	Not used
6:0	INTEN	Interrupt Enables Each of these bits is ANDed with each of the bits in the Interrupt Status Register to contribute to the ORed INTERRUPT output. To mask the contribution of the corresponding bit in the Interrupt Status Register, set that bit to 0; to enable the contribution, set that bit to 1. Note that the Interrupt Status Register bits only mask what appears on the INTERRUPT output and that the Interrupt Status Register bits will always be active to reflect current interrupt source conditions unless the REMOVE_IREG_LOGIC parameter/generic is set to 1. If REMOVE_IREG_LOGIC is set to 1, bits 6 and 3 down to 0 of this register will be permanently tied to logic 0. See the Interrupt Status Register description for details on which bits are disabled when the REMOVE_IREG_LOGIC parameter/generic is set to 1.

Table 4-13 · Interrupt Status Register

Bits	Name	Function
15:7	Reserved	Not used
6	INT6	RTC Match Rising Edge If 1, the RTCMATCH output has transitioned high, indicating that the RTC has reached the desired count. Note that if the REMOVE_IREG_LOGIC parameter/generic is set to 1, this bit will be permanently tied to logic 0.
5	INT5	ACM Read Done If 1, an ACM read cycle has completed and valid data can be read from the ACM DATA register.
4	INT4	ACM Write Done If 1, an ACM write cycle has completed and valid data has been written into the AB macro.
3	INT3	DATAVALID Rising Edge If 1, the DATAVALID signal from the AB block has transitioned high, indicating that valid converted data from the ADC is available on the RESULT[11:0] output port from the AB block. Note that if the REMOVE_IREG_LOGIC parameter/generic is set to 1, this bit will be permanently tied to logic 0.
2	INT2	BUSY Rising Edge If 1, the BUSY signal from the AB block has transitioned high, indicating that the ADC has begun an analog-to-digital conversion and is now busy performing that conversion. Note that if the REMOVE_IREG_LOGIC parameter/generic is set to 1, this bit will be permanently tied to logic 0.
1	INT1	CALIBRATE Falling Edge If 1, the CALIBRATE signal from the AB block has transitioned low, indicating that the ADC has finished its calibration procedure. Note that if the REMOVE_IREG_LOGIC parameter/generic is set to 1, this bit will be permanently tied to logic 0.
0	INT0	CALIBRATE Rising Edge If 1, the CALIBRATE signal from the AB block has transitioned high, indicating that the ADC has started its calibration procedure. Note that if the REMOVE_IREG_LOGIC parameter/generic is set to 1, this bit will be permanently tied to logic 0.

ACM Interface

The ACM interface is used to configure the analog quads and RTC within the AB macro. Various features of the analog quads need to be set prior to correct operation of the ADC, including the pre-scaler circuits in each of the AV, AC, and AT analog input ports that will be accessed in the user's design. The SmartDesign IP configuration GUI can be used to configure the various pre-scaler settings for each quad, as well as the functions used (voltage monitor, current monitor, temperature monitor, gate-driver output driver strengths, etc.). Consult the Fusion datasheet for details on how each pre-scaler should be set, relative to the design-specific voltage, current, or temperature ranges used. The internal address map of the ACM is shown in Table 4-14 on page 37.

Note that the ACM must operate with a clock frequency that is less than or equal to 10 MHz; to achieve this requirement, the user must set the frequency of the PCLK input correctly via the PCLK_FREQUENCY parameter/generic (refer to the PCLK_FREQUENCY parameter/generic in the Parameters/Generics section, and note that the value entered must be rounded up to the nearest integer). An internal ACM clock divider circuit uses the value set in the

PCLK_FREQUENCY parameter/generic in order to correctly derive a frequency less than or equal to 10 MHz for the ACM clock. For example, if the system clock frequency (PCLK) is 39.25 MHz, PCLK_FREQUENCY should be set to 40. ACM reads and writes are synchronized to the internally generated ACM clock. Since the ACM clock is operating at a lower frequency than the system clock (PCLK) used by CoreAI, various status and interrupt status registers have been implemented to indicate when ACM read and write accesses are busy or completed (see Table 4-3 on page 31 and Table 4-13 on page 36).

Table 4-14 · ACM Address Map for Configuring Analog Quads and RTC

ACMADDR[7:0]	Name	Description	Associated Peripheral
0x00	-	Reserved	Unused
Analog Quad 0			
0x01	AQ0	Byte 0 (AV0 control)	Analog Quad
0x02	AQ0	Byte 1 (AC0 control)	Analog Quad
0x03	AQ0	Byte 2 (AG0 control)	Analog Quad
0x04	AQ0	Byte 3 (AT0 control)	Analog Quad
Analog Quad 1			
0x01	AQ0	Byte 0 (AV0 control)	Analog Quad
0x02	AQ0	Byte 1 (AC0 control)	Analog Quad
0x03	AQ0	Byte 2 (AG0 control)	Analog Quad
0x04	AQ0	Byte 3 (AT0 control)	Analog Quad
Analog Quad 1			
0x05	AQ1	Byte 0 (AV1 control)	Analog Quad
0x06	AQ1	Byte 1 (AC1 control)	Analog Quad
0x07	AQ1	Byte 2 (AG1 control)	Analog Quad
0x08	AQ1	Byte 3 (AT1 control)	Analog Quad
…			
Analog Quad 9			
0x25	AQ9	Byte 0 (AV9 control)	Analog Quad
0x26	AQ9	Byte 1 (AC9 control)	Analog Quad
0x27	AQ9	Byte 2 (AG9 control)	Analog Quad
0x28	AQ9	Byte 3 (AT9 control)	Analog Quad
Real-Time Counter			
0x40	COUNTER0	Counter bits 7:0	RTC
0x41	COUNTER1	Counter bits 15:8	RTC
0x42	COUNTER2	Counter bits 23:16	RTC
0x43	COUNTER3	Counter bits 31:24	RTC

Table 4-14 · ACM Address Map for Configuring Analog Quads and RTC

ACMADDR[7:0]	Name	Description	Associated Peripheral
0x44	COUNTER4	Counter bits 39:32	RTC
0x48	MATCHREG0	Match register bits 7:0	RTC
0x49	MATCHREG1	Match register bits 15:8	RTC
0x4A	MATCHREG2	Match register bits 23:16	RTC
0x4B	MATCHREG3	Match register bits 31:24	RTC
0x4C	MATCHREG4	Match register bits 39:32	RTC
0x50	MATCHBITS0	Individual match bits 7:0	RTC
0x51	MATCHBITS1	Individual match bits 15:8	RTC
0x52	MATCHBITS2	Individual match bits 23:16	RTC
0x53	MATCHBITS3	Individual match bits 31:24	RTC
0x54	MATCHBITS4	Individual match bits 39:32	RTC
0x58	CTRL_STAT	Control (write) / Status (read) register	RTC
0x59	TEST_REG	Test register	RTC

Note: Table values shown in hexadecimal format.

ACM Reads

An example of reading data from the ACM is illustrated in Figure 4-2 on page 38. The steps for reading information from the AB block via the ACM are:

1. Read the ACM Control/Status register to make sure that CoreAI is not busy processing an ACM read or write.
2. Write the desired ACM address to the ACM Address register.
3. Write a logic 1 to the ACMRDSTART bit of the ACM Control/Status register.
4. Keep reading (polling) the ACM Control/Status register until CoreAI is not busy processing the ACM read. Alternatively, the BUSY output of CoreAI may be monitored during this operation.
5. Read the ACM Data register to read the byte of ACM data that is now available from the AB block.

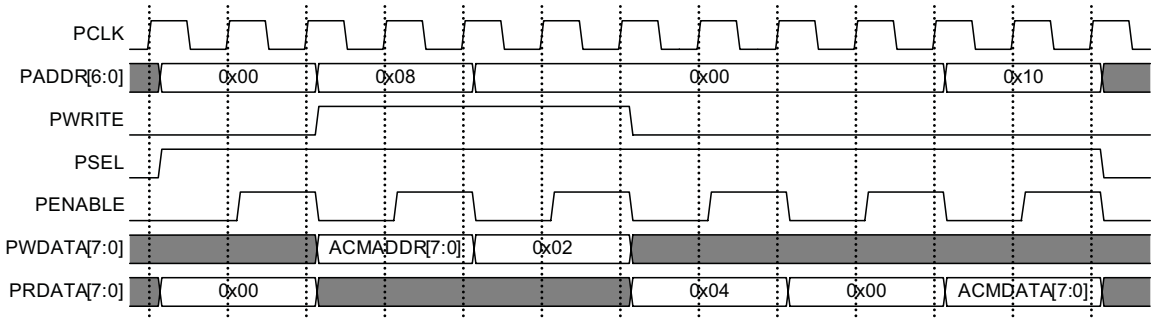

Figure 4-2 · ACM Read Procedure (8-bit APB reads shown)

ACM Writes

An example of writing data into the ACM is illustrated in Figure 4-3. The steps for writing information into the AB block via the ACM are:

1. Read the ACM Control/Status register to make sure that CoreAI is not busy processing an ACM read or write.
2. Write the desired ACM address to the ACM Address register.
3. Write the desired ACM data to the ACM Data register. At this point, internal logic within CoreAI begins the process of transferring the contents of the ACM Data register to the ACM location that is addressed by the ACM Address register. Since the act of writing data to the ACM Data register starts an ACM write procedure, no equivalent to the ACMRDSTART bit, such as an "ACMWRSTART" bit is required.
4. Keep reading (polling) the ACM Control/Status register until CoreAI is not busy processing the ACM write. Alternatively, the BUSY output of CoreAI may be monitored during this operation.

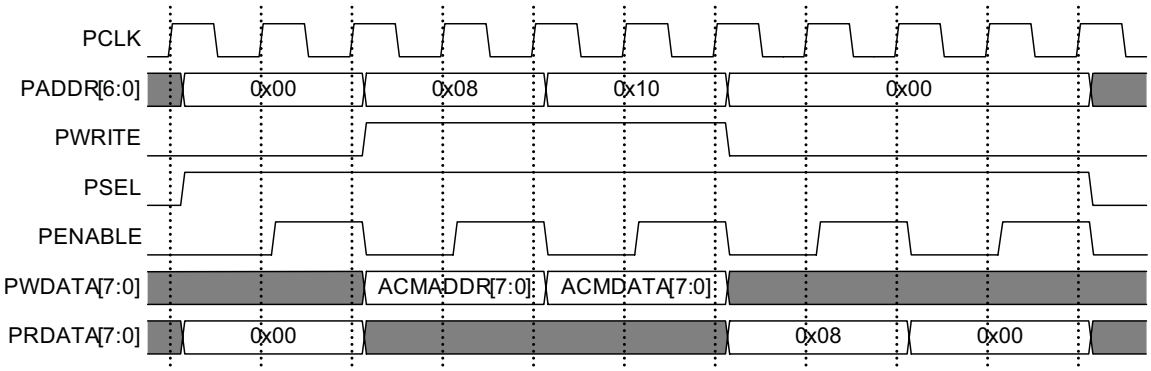

Figure 4-3 · ACM Write Procedure (8-bit APB writes shown)

RTC Access

For detailed connection and operation of the 40-bit real-time counter block within the AB macro, refer to the Fusion datasheet. The SmartGen software or CoreConsole IDP configuration GUI is used to configure the RTC. The internal registers of the RTC are accessed via the ACM interface address map listed in Table 4-14 on page 37. The primary control and status of the RTC is accomplished via writes and reads of the RTC CTRL_STAT (Control Status) register at ACM address 0x58 (decimal 88) that is listed in Table 4-14. The bit descriptions of the RTC Control/Status register are listed in Table 4-15.

Table 4-15 · RTC Control/Status Register

Bit	Name	Description
7	rtc_rst	RTC Reset: Writing logic 1 to this bit causes an RTC reset. Reset of all RTC states (except this Control/Status register) occurs asynchronously if VCC33UP = 0 or CTRL_STAT bit 7 (rtc_rst) is set to 1. Writing a logic 0 to this bit will allow synchronous deassertion of reset after 2 ACM_CLK cycles if VCC33UP = 1. Reset is removed synchronously after 2 rising edges of the ACM_CLK, following both VCC33UP = 1 and rtc_rst = 1.
6	cntr_en	Counter Enable: Writing logic 1 in this bit will enable the counter if the RTC is not in reset. It takes 64 RTCCLK positive edges (1/2 of the pre-scaler division factor), after reset is removed and cntr_en = 1, before the counter is incremented. Writing logic 0 in this bit resets the pre-scaler and therefore suspends incrementing the counter, but the counter is not reset. Before writing to the COUNTER registers, the counter must be disabled.

Table 4-15 · RTC Control/Status Register

Bit	Name	Description
5	vr_en_mat	Voltage Regulator Enable on Match: Writing logic 1 to this bit will allow the RTCMATCH output port to go to logic 1 when a match occurs between the 40-bit counter and the 40-bit match register. Logic 0 forces RTCMATCH to logic 0, to prevent enabling the voltage regulator from the RTC.
4:3	xt_mode[1:0]	Crystal Oscillator Mode: These bits control the RTCXTLMODE[1:0] output ports that are connected to the RTCMODE[1:0] input pins of the crystal oscillator pad. For 32.768 kHz crystal operation, this should be set to xt_mode[1:0] = "01".
2	rst_cnt_omat	Reset Counter on Match: Writing logic 1 to this bit allows the counter to clear itself when a match occurs. In this situation, the 40-bit counter clears on the next rising edge of the pre-scaled clock, approximately 4 ms after the match occurs (the pre-scaled clock toggles at a rate of 256 Hz, given a 32.768 kHz external crystal). Writing logic 0 to this bit allows the counter to increment indefinitely, while still allowing match events to occur.
1	rstb_cnt	Counter Reset: Writing logic 0 resets the 40-bit counter value to 0. Writing logic 1 allows the counter to count. Counter will first increment on the 64th rising edge of RTCCLK after all of the following are true: reset is removed, the rstb_cnt bit is set to 1, and the cntr_en bit is set to 1; it will then increment every 128 RTCCLK cycles
0	xtal_en	Crystal Oscillator Enable: This bit controls the RTCXTLSEL output port that is connected to the SELMODE input pin of the crystal oscillator. If logic 0 is written to this bit, only the FPGA fabric can be used to control the crystal oscillator EN and MODE[1:0] inputs. If logic 1 is written to this bit, the RTC takes control of the XTAL oscillator; for example, RTC mode bits (RTCXTLMODE[1:0]) configure the XTAL oscillator (not the FPGA mode bits). This bit must be set to 1 to allow the RTC counter to function if the 1.5 V supply is off. To enable sleep mode: set xtal_en to logic 0 so that the crystal is controlled from the FPGA EN signal then, when the FPGA is powered down, the EN signal from the FPGA will be logic 0, disabling the crystal oscillator.

5 ADC Operation

Control of the ADC within the AB macro in CoreAI is accomplished by APB reads and writes. After a power-up reset condition, the ADC will come out of its reset state and commence with its internal calibration sequence. When this calibration sequence has finished, the ADC will be ready to use for conversions. Read the INT[1:0] register bits to obtain status information relevant to the ADC calibration.

You must configure the analog quads appropriately via the ACM interface to match the required design-specific voltage, current, and temperature ranges prior to performing ADC conversions; failure to do so may result in damage to the Fusion device. You can use the Catalog in the Project Manager to configure the analog quads and RTC.

ADC Control

A typical ADC analog-to-digital conversion is shown starting in Figure 5-1 and ending in Figure 5-2. The steps for performing an ADC conversion are:

1. Read the ADC Status register to make sure that the ADC is not busy calibrating or performing a conversion.
2. Write the desired ADC settings to ADC Control register 1, including the voltage reference selection (internal or external), mode selection (ADC resolution), and clock divider settings.
3. Write the desired ADC settings to ADC Control register 2, including the sample time control, channel number to sample (1 of 32), and finally, set the start conversion bit to begin the conversion process.
4. If a current monitor or temperature monitor operation is used on a corresponding AC or AT input pin, respectively, a write must be done to set the corresponding current monitor strobe or temperature monitor strobe high in ADC Control register 3 or ADC Control register 4, respectively. Refer to the Fusion Datasheet for details on external component connection requirements for current monitoring and temperature monitoring. Alternatively, if the desired current or temperature monitor strobe is hardware-controlled (see the Parameters/Generics section), the HD_CMSTBx or HD_TMSTBx may be controlled by hardware external to CoreAI to set the strobe high.
5. Keep reading the high-order ADC Status register until the ADC is not busy performing a conversion. Alternatively, the BUSY output of CoreAI may be monitored during this operation.
6. If a current monitor or temperature monitor operation has completed using a corresponding AC or AT input pin, respectively, a write must be done to set the corresponding current monitor strobe or temperature monitor strobe low in ADC Control register 3 or ADC Control register 4, respectively. Alternatively, if the desired current or temperature monitor strobe is hardware-controlled (see the Parameters/Generics section), the HD_CMSTBx or HD_TMSTBx may be controlled by hardware external to CoreAI to set the strobe low.
7. For 8-bit APB mode (APB_DWIDTH=8), read the low-order ADC Status register to obtain the lower 8 bits of the ADC result conversion data (the upper 4 bits of conversion data have already been read from the high-order ADC Status register).

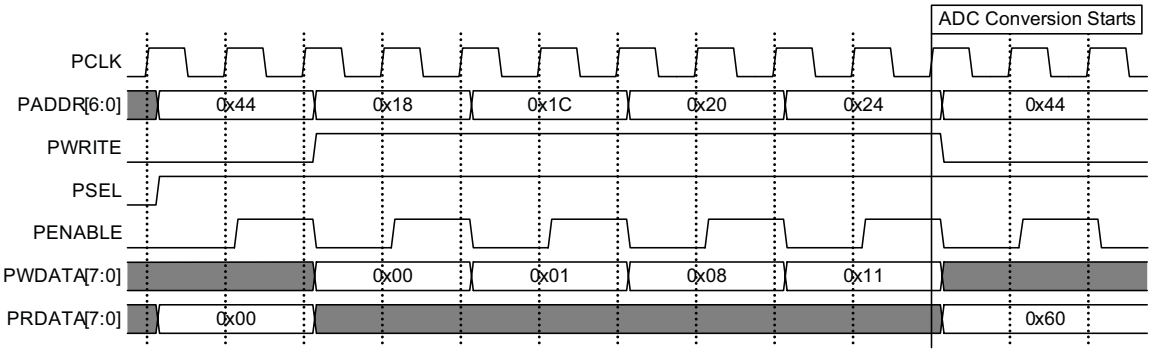

Figure 5-1 · ADC Start of Conversion

Note: 8-bit APB reads and writes shown, "hi" refers to bits 15:8, "lo" refers to bits 7:0.

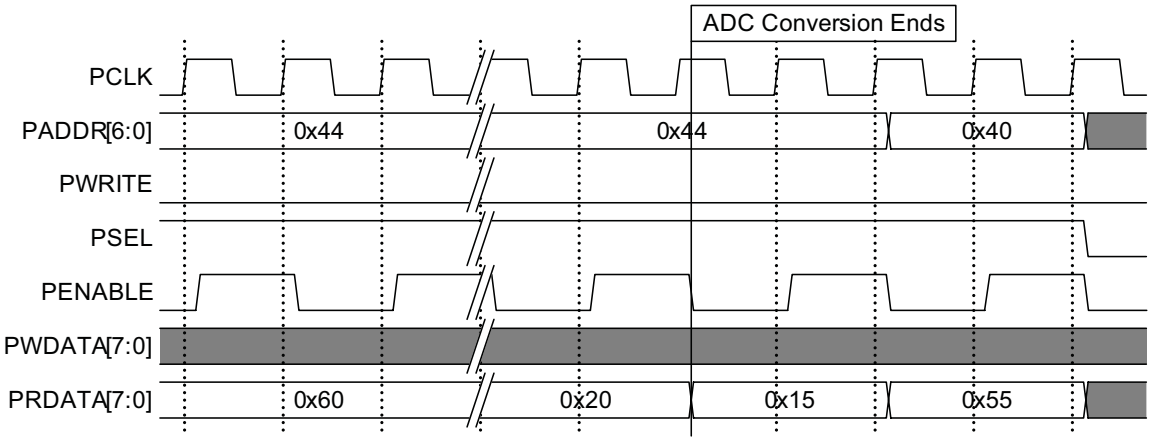

Figure 5-2 · ADC End of Conversion

Note: 8-bit APB reads and writes shown, "hi" refers to bits 15:8, "lo" refers to bits 7:0

Testbench Operation and Modification

Simple Application Testbench

An example user testbench is included with CoreAI. The user testbench is provided in pre-compiled Model*Sim* format and in RTL source code for all releases (obfuscated and RTL), for you to examine and modify to suit your needs. The source code for the user testbench is provided to ease the process of integrating the CoreAI macro into your design and verifying according to your own custom needs. A block diagram of the user testbench is shown in Figure 6-1.

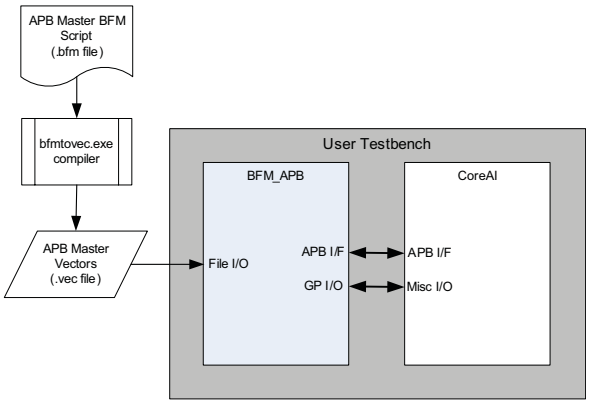

Figure 6-1 · CoreAI User Testbench

As shown in Figure 6-1, the user testbench instantiates an Actel DirectCore AMBA BFM (bus functional model) module to emulate an APB master that controls the operation of CoreAI via reads and writes to access internal registers.

A BFM ASCII script source file with comments, is included in the directory <proj>/simulation; where <proj> represents the path to your Libero IDE project. The BFM source file coreai_usertb_apb_master.bfm controls the APB master processor. This BFM source file is automatically recompiled each time the simulation is invoked from Libero IDE by the bfmtovec.exe executable, if running on a Windows platform, or by the bfmtovec.lin executable, if running on a Linux platform. The coreai_usertb_apb_master.vec vector file, created by the bfmtovec executable, is read in by the BFM module for simulation in Model*Sim*.

You may alter the BFM script, if desired. Refer to the Actel DirectCore AMBA BFM User's Guide for more information.

The source code for the user testbench and BFM script is available with the CoreAI obfuscated and RTL releases. A compiled Model*Sim* simulation library containing the BFM modules is available with the CoreAI obfuscated release. Obfuscated RTL versions of the BFM modules are available with the CoreAI RTL release.

System Operation

This chapter provides hints to ease the process of implementation and integration of CoreAI into your own design.

Using CoreAI with Cortex-M1

CoreAI can be used with Cortex-M1, Actel's soft IP version of the popular ARM® microprocessor that has been optimized for the M1 Fusion flash-based FPGA devices. To create a design using Cortex-M1 and CoreAI you should use SmartDesign. An example SmartDesign sub-system using Cortex-M1 and CoreAI is shown in Figure 7-1. Please refer to the Libero IDE online help for information on creating a Cortex-M1-based design using SmartDesign.

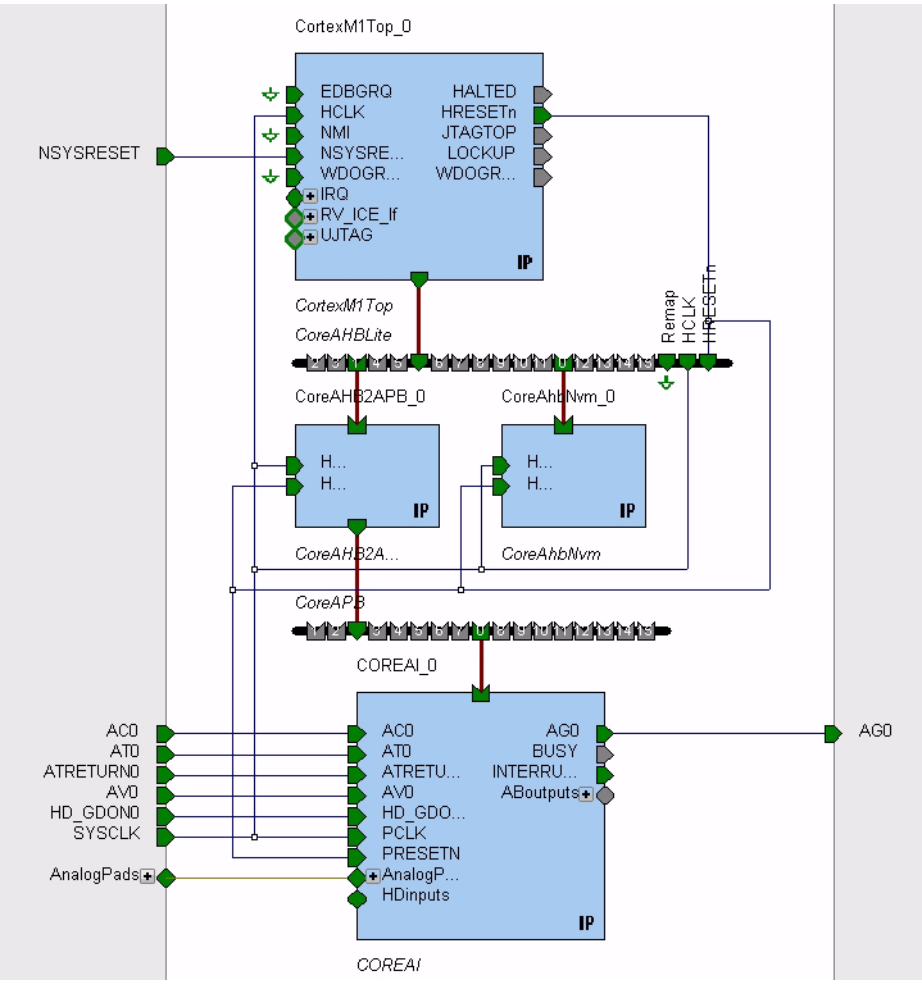

Figure 7-1 · Example System Using CoreMP7 and CoreAI

Using CoreAI with Core8051s

CoreAI can also be used with Core8051s. An example FPGA design using Core8051s and CoreAI is shown in Figure 7-2. For this example, the internal Flash memory is used for Core8051s program storage, and can be

programmed independently from the FPGA fabric by use of the FlashPro software and hardware (refer to FlashPro documentation for details on how to program the Flash memory within the Fusion devices).

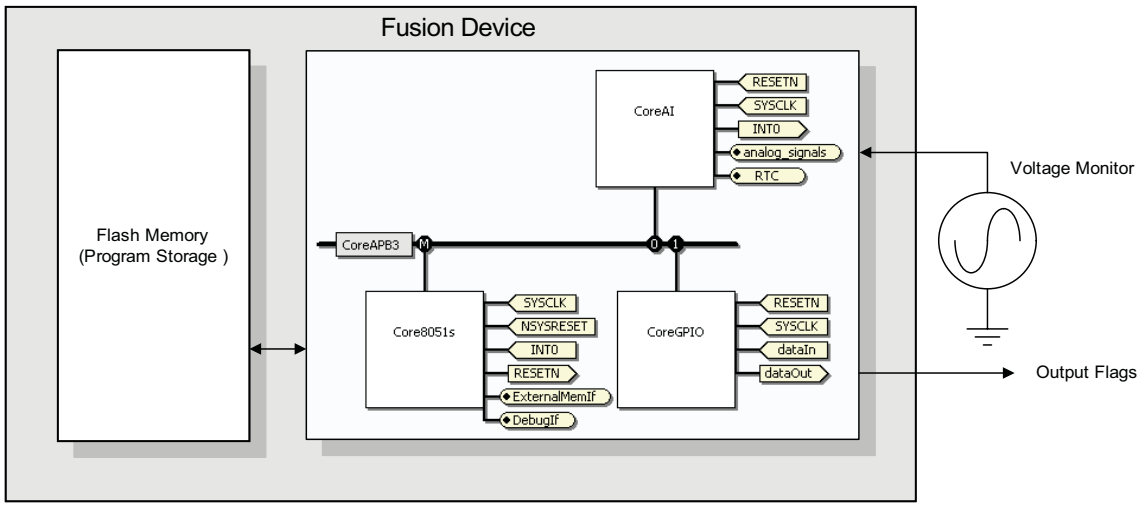

Figure 7-2 · Example System Using Core8051s and CoreAI

Using CoreAI with CoreABC

CoreAI can also be used with CoreABC. An example FPGA design using CoreABC and CoreAI is shown in Figure 7-3. CoreABC allows a simple set of APB read and write cycles that can be used to configure CoreAI and then to read and compare the Analog values to turn the digital outputs on and off.

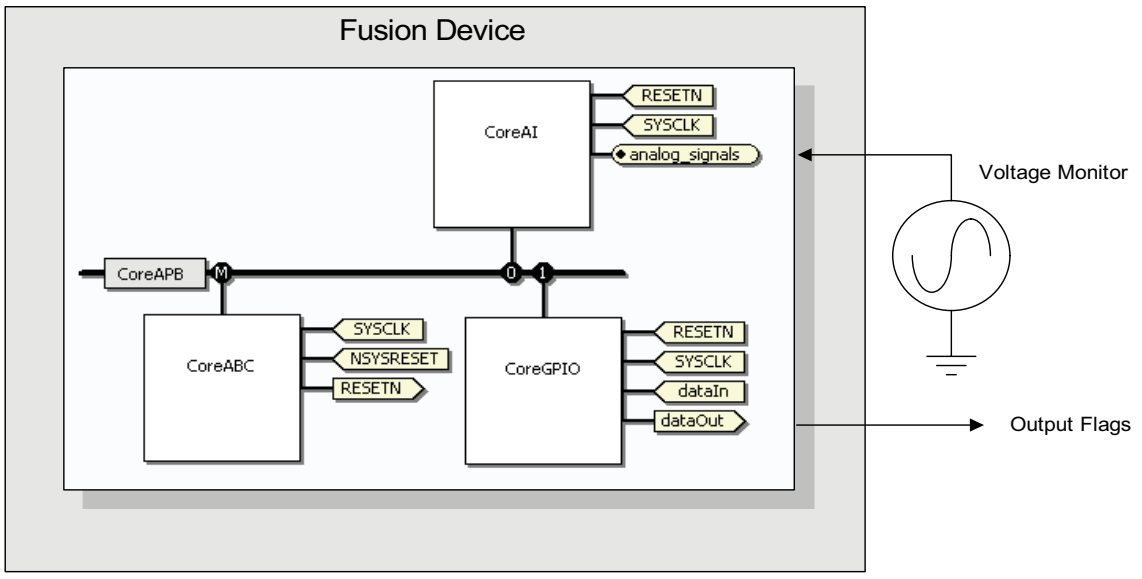

Figure 7-3 · Example System using CoreABC and CoreAI

Using CoreAI with ADC Results FIFO

CoreAI can be used with FIFO logic to store ADC results. An example FPGA design using Core8051s, CoreAI, and the user's own custom logic controlling a FIFO instantiated using the Libero IDE is shown in Figure 7-4. In this example design, a rising edge event on the AB_DATAVALID status signal is used to write valid data from the AB_RESULT[11:0] signals as well as the AB_CHNUMBER[4:0] channel selection signals into the FIFO.

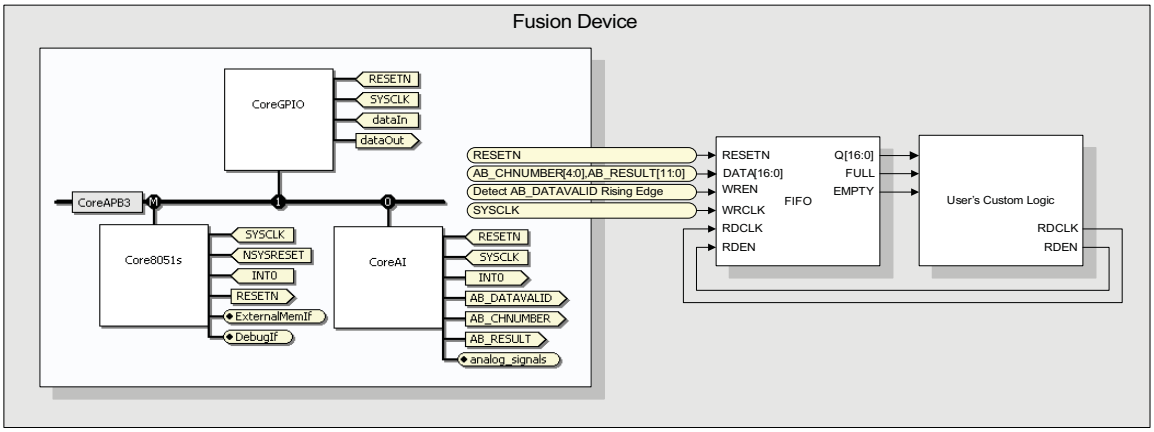

Figure 7-4 · Example Using CoreAI with FIFO Logic

8 Ordering Information

Ordering Codes

CoreAI can be ordered through your local Actel sales representative. Order using the number scheme: CoreAI-XX, where XX is listed in Table 8-1.

Table 8-1 · Ordering Codes

XX	Description
OM	RTL for Obfuscated RTL - multiple-use license
RM	RTL for RTL source - multiple-use license

Note: CoreAI-OM is included free with a Libero IDE license.

A

Product Support

Actel backs its products with various support services including Customer Service, a Customer Technical Support Center, a web site, an FTP site, electronic mail, and worldwide sales offices. This appendix contains information about contacting Actel and using these support services.

Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

From Northeast and North Central U.S.A., call **650.318.4480**
From Southeast and Southwest U.S.A., call **650. 318.4480**
From South Central U.S.A., call **650.318.4434**
From Northwest U.S.A., call **650.318.4434**
From Canada, call **650.318.4480**
From Europe, call **650.318.4252** or **+44 (0) 1276 401 500**
From Japan, call **650.318.4743**
From the rest of the world, call **650.318.4743**
Fax, from anywhere in the world **650.318.8044**

Actel Customer Technical Support Center

Actel staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions. The Customer Technical Support Center spends a great deal of time creating application notes and answers to FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

Actel Technical Support

Visit the Actel Customer Support website (www.actel.com/custsup/search.html) for more information and support. Many answers available on the searchable web resource include diagrams, illustrations, and links to other resources on the Actel web site.

Website

You can browse a variety of technical and non-technical information on Actel's home page, at www.actel.com.

Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center from 7:00 A.M. to 6:00 P.M., Pacific Time, Monday through Friday. Several ways of contacting the Center follow:

Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is tech@actel.com.

Phone

Our Technical Support Center answers all calls. The center retrieves information, such as your name, company name, phone number and your question, and then issues a case number. The Center then forwards the information to a queue where the first available application engineer receives the data and returns your call. The phone hours are from 7:00 A.M. to 6:00 P.M., Pacific Time, Monday through Friday. The Technical Support numbers are:

650.318.4460
800.262.1060

Customers needing assistance outside the US time zones can either contact technical support via email (tech@actel.com) or contact a local sales office. Sales office listings can be found at www.actel.com/contact/offices/index.html.

Index

Actel is the leader in low-power and mixed-signal FPGAs and offers the most comprehensive portfolio of system and power management solutions. Power Matters. Learn more at www.actel.com.

Actel Corporation *• 2061 Stierlin Court • Mountain View, CA 94043 • USA*
Phone 650.318.4200 • Fax 650.318.4600 • Customer Service: 650.318.1010 • Customer Applications Center: 800.262.1060

Actel Europe Ltd*. • River Court, Meadows Business Park • Station Approach, Blackwater • Camberley Surrey GU17 9AB • United Kingdom*
Phone +44 (0) 1276 609 300 • Fax +44 (0) 1276 607 540

Actel Japan *• EXOS Ebisu Building 4F • 1-24-14 Ebisu Shibuya-ku • Tokyo 150 • Japan*
Phone +81.03.3445.7671 • Fax +81.03.3445.7668 • http://jp.actel.com

Actel Hong Kong *• Room 2107, China Resources Building • 26 Harbour Road • Wanchai • Hong Kong*
Phone +852 2185 6460 • Fax +852 2185 6488 • www.actel.com.cn

50200102-2/X.XX