CoreAI

DirectCore

Product Summary

Intended Use

- Analog Interface Control Using a Microprocessor/ Microcontroller and an Actel Fusion™ Device
- Voltage, Current, and Temperature Monitoring Using a Microprocessor/Microcontroller and an Actel Fusion Device

Key Features

- ADC Conversions Controlled by MCU/MPU Writes
- AMBA APB Slave Interface (8- or 16-Bit Data Widths Supported)
- 14 Maskable Interrupt Sources
- Internal Clock Divider for Generating Analog Configuration MUX Clock
- Optional Read FIFO Stores up to 256 ADC Conversion Results
- Analog Configuration MUX Can Be Configured by SmartGen

Supported Families

- Fusion (including M7 devices)

Core Deliverables

- Evaluation Version
  - Compiled RTL Simulation Model Fully Supported in Actel Libero® Integrated Design Environment (IDE)
- Netlist Version
  - Structural Verilog and VHDL Netlists (with and without I/O Pads) Compatible with Actel Designer Software Place-and-Route Tool
  - Compiled RTL Simulation Model Fully Supported in Actel Libero IDE
- RTL Version
  - Verilog and VHDL Core Source Code
  - Core Synthesis Scripts
- Testbench (Verilog and VHDL)

Synthesis and Simulation Support

- Directly Supported within Actel Libero IDE and CoreConsole
- Synthesis: Synplicity®, Synopsys® (Design Compiler / FPGA Compiler / FPGA Express), Exemplar
- Simulation: OVI-Compliant Verilog Simulators and Vital-Compliant VHDL Simulators

Core Verification

- Comprehensive VHDL and Verilog Testbenches
- User Can Easily Modify User Testbench Using Existing Format to Add Custom Tests

Contents

General Description

CoreAI (Analog Interface) allows for simple control of the analog peripherals within the Fusion family of Actel devices. Control may be implemented with an internal or external microprocessor or microcontroller (such as Core8051 or CoreMP7), or with user-created custom logic within the FPGA fabric. 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, as shown in Figure 1 on page 2. 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 (refer to "ACM Interface" on page 18 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*.

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 4 on page 6.

The CoreAI block diagram is shown in Figure 1. A typical application using CoreAI is shown in Figure 2.

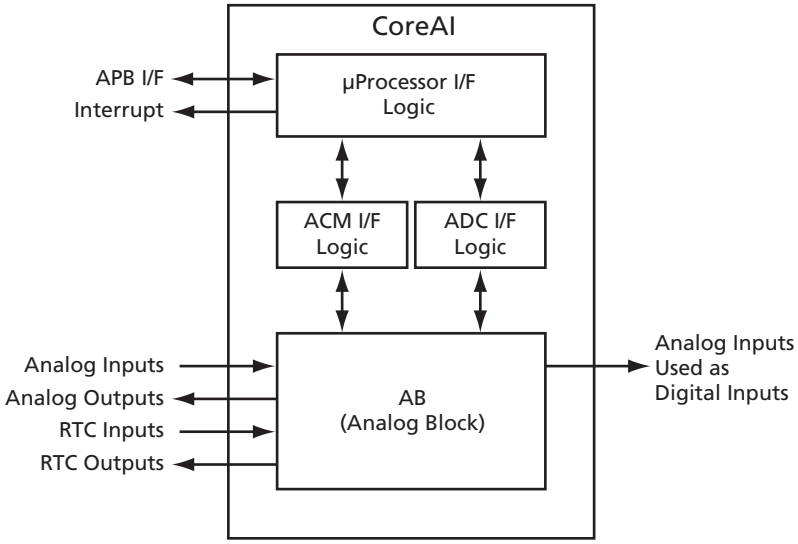


Figure 1 • **CoreAI Block Diagram**

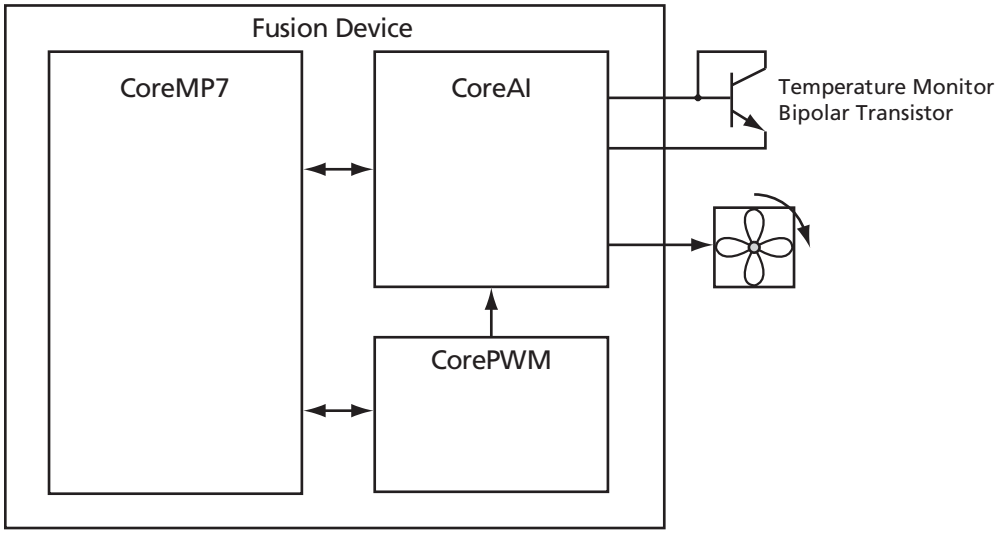


Figure 2 • **CoreAI Typical Application**

Functional Block Descriptions

CoreAI, shown in Figure 1 on page 2, 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.

CoreAI Device Requirements

CoreAI has been implemented in the Actel Fusion device family. A summary of the data for CoreAI is listed in Table 1 and Table 2.

Table 1 • **CoreAI Device Utilization and Performance (minimum configuration)**

Family	Cells or Tiles			Utilization		Performance
	Sequential	Combinatorial	Total	Device	Total	
Fusion	45	105	150	AFS090	7%	150 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: FIXED_VAREFSEL = 1, FIXED_VAREFSEL_VAL = 0, FIXED_MODE = 1, FIXED_MODE_VAL = 0, FIXED_TVC = 1, FIXED_TVC_VAL = 0, FIXED_STC = 1, FIXED_STC_VAL = 0, CFG_ACx = 512, CFG_ATx = 512, DISABLE_TMSTBINT = 1, CFG_GDx = 768, ACTLOW_INTERRUPT = 0, DISABLE_INTERRUPT = 1, APB_16BIT_DATA = 1.

Table 2 • **CoreAI Device Utilization and Performance (maximum configuration)**

Family	Cells or Tiles				Utilization		Performance
	Sequential	Combinatorial	Total	FIFO	Device	Total	
Fusion	130	330	460	1	AFS090	20%	133 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: ACM_CLK_DIV = 4, USE_RTC = 1, USE_RDFIFO = 1, USE_RDFIFO_AEVAL = 16, USE_RDFIFO_AFVAL = 240.

CoreAI Verification

The comprehensive simulation testbench verifies correct operation of the CoreAI macro.

The testbench applies several tests to the CoreAI macro, including the following:

- Voltage monitor, current monitor, and temperature monitor tests
- RTC tests
- Gate-driver control tests

Using the supplied testbench as a guide, the user can alter the verification of the core by adding custom tests or removing existing tests.

I/O Signal Descriptions

The port signals for the CoreAI macro are defined in Table 3 and illustrated in Figure 3. CoreAI has 120 I/O signals. Note that vector notation is used in Figure 3 for the AV, AC, AT, ATRETURN, DDGDON, DAVOUT, DACOUT, DATOUT, AG, and RTCXTLMODE ports; however, these ports are actually split into individual single-bit ports, as described in Table 3. For example, there are two individual output ports, RTCXTLMODE1 and RTCXTLMODE0, rather than one vectored output port RTCXTLMODE[1:0].

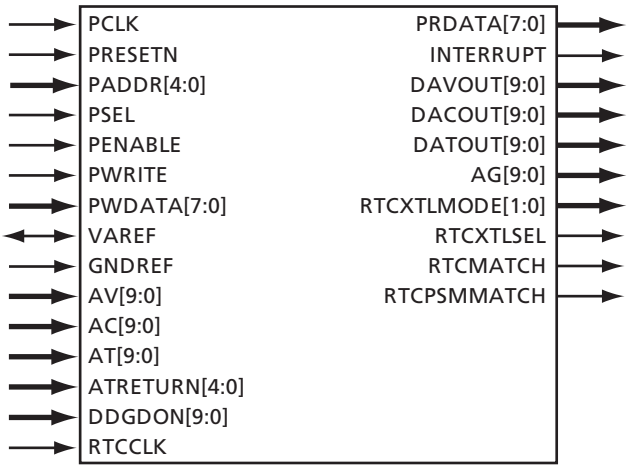


Figure 3 • **CoreAI I/O Signal Diagram**

Table 3 • **CoreAI I/O Signal Descriptions**

Name	Type	Description
APB Interface		
PCLK	Input	APB System Clock: reference clock for all internal logic
PRESETN	Input	APB active-low asynchronous reset
PADDR[4:0]	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.
PWDATA[15:0]	Input	APB write data – If the APB_16BIT_DATA parameter/generic is set to 1, all 16 bits are used; if the APB_16BIT_DATA parameter/generic is set to 0, only the lower 8 bits, PWDATA[7:0], are used (in this case, PWDATA[15:8] should be tied to static high or low values).
PRDATA[15:0]	Output	APB read data – If the APB_16BIT_DATA parameter/generic is set to 1, all 16 bits are used; if the APB_16BIT_DATA parameter/generic is set to 0, only the lower 8 bits, PRDATA[7:0], are used (in this case, PRDATA[15:8] can be left unconnected).
INTERRUPT	Output	Microprocessor interrupt output – This interrupt signal is generated from 14 possible interrupt sources, each of which can be masked or enabled via the INTENABLE register. The polarity of this output is controlled via the ACTLOW_INTERRUPT parameter/generic.

Note: All signals active high (logic 1) unless otherwise noted.

Table 3 • **CoreAI I/O Signal Descriptions (Continued)**

Name	Type	Description
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 FIXED_VAREFSEL and FIXED_VAREFSEL_VAL parameters/generics).
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.
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.
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.
ATRETURN4, ATRETURN3, ATRETURN2, ATRETURN1, ATRETURN0]	Input	Shared Analog Temperature Monitor Returns – These signals correspond to the shared returns for the temperature monitor inputs (ATRETURN89 through ATRETURN01) of the AB macro.
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.
DDGDON9, DDGDON8, DDGDON7, DDGDON6, DDGDON5, DDGDON4, DDGDON3, DDGDON2, DDGDON1, DDGDON0	Input	Direct Digital 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 "Parameter/Generic Descriptions" on page 6).
AG9, AG8, AG7 AG6, 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: All signals active high (logic 1) unless otherwise noted.

Table 3 • **CoreAI I/O Signal Descriptions (Continued)**

Name	Type	Description
RTCCLK	Input	RTC Clock input – If the RTC is used (via the USE_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.
RTCXTLMODE1, RTCXTLMODE0	Output	RTC XTLOSC Mode outputs – If the RTC is used (via the USE_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 USE_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 USE_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 VRPSM output – If the RTC is used (via the USE_RTC parameter/generic), this output port can be connected to the VRPSM pin of the internal voltage regulator to control regulator power-up; if the RTC is not used, this pin should be left unconnected.

Note: All signals active high (logic 1) unless otherwise noted.

Parameter/Generic Descriptions

CoreAI has parameters (Verilog) and generics (VHDL), described in Table 4, for configuring the RTL code. All parameters and generics are integer types.

Table 4 • **CoreAI Parameters/Generics Descriptions**

Name	Valid Range	Description
APB_16BIT_DATA	0 or 1	Set this to 1 if the APB reads and writes are using all 16-bits of the PRDATA[15:0] and PWDATA[15:0] ports, or to 0 if 8-bit APB reads and writes are done on the PRDATA[15:0] and PWDATA[15:0] ports. If 8-bit APB reads and writes are used, only the lower eight bits of each port are used, i.e., PRDATA[7:0] and PWDATA[7:0]; PWDATA[15:8] should be tied statically high or low and PRDATA[15:8] should be left unconnected in this case. Note that for 16-bit reads and writes, the internal address map shown in Table 9 on page 11 will concatenate adjacent bytes since PADDR[0] will be ignored; in this case, each byte that is addressed by PADDR[0] = 0 will correspond to the LSB and each byte that is addressed by PADDR[0] = 1 will correspond to the MSB of each 16-bit word. The default value of 0 uses 8-bit APB reads and writes.
ACTLOW_INTERRUPT	0 or 1	Interrupt Active Low: Set this to 1 if the INTERRUPT output polarity is active low, or to 0 if the INTERRUPT output polarity is active high. The default value of 0 is active high.
DISABLE_INTERRUPT	0 or 1	Disable Interrupt: Set this to 1 if the INTERRUPT output will not be used, in which case the INTERRUPT output will be fixed at the inactive polarity chosen by ACTLOW_INTERRUPT, i.e., INTERRUPT = ACTLOW_INTERRUPT; set this to 0 if the INTERRUPT output is used. The default value of 0 uses the INTERRUPT output.

Note: *Invalid values are 3, 7, 11, and 15 (refer to the *Fusion datasheet*).

Table 4 • **CoreAI Parameters/Generics Descriptions (Continued)**

Name	Valid Range	Description
ACM_CLK_DIV	1 to 4	Set this to the division value to generate the ACM clock, which has a maximum frequency of 10 MHz. The actual value will be the frequency of PCLK divided by ($2^{ACM\_CLK\_DIV}$); valid values are 1–4. The default value of 1 sets the ACM clock at half the PCLK frequency. **Note:** It is the user's responsibility to ensure that the ACM clock has a maximum operating frequency of 10 MHz.
USE_RTC	0 or 1	Use RTC: Set this to 1 if the RTC is to be used, or to 0 if the RTC is not to be used. The default value of 0 does not use the RTC.
USE_RDFIFO	0 or 1	Use Read FIFO: Set this to 1 if using an internal data FIFO for storing up to 256 ADC conversion results for later reading by a microprocessor, or to 0 if not using the FIFO. The default value of 0 does not use a read FIFO.
USE_RDFIFO_AEVAL	0 to 255	Read FIFO Almost Empty Value: Set this 8-bit integer to the fixed almost empty value of the read FIFO. This value determines when the almost empty flag output from the read FIFO becomes active; it represents the number of 16-bit ADC conversion result words that are left to be read before the read FIFO will be empty. **Note:** This is only used if USE_RDFIFO = 1.
USE_RDFIFO_AFVAL	0 to 255	Read FIFO Almost Full Value: Set this 8-bit integer to the fixed almost full value of the read FIFO. This value determines when the almost full flag output from the read FIFO becomes active; it represents the number of 16-bit ADC conversion result words that are left to be written before the read FIFO will be full. **Note:** This is only used if USE_RDFIFO = 1.
FIXED_VAREFSEL	0 or 1	Set this to 1 if the VAREFSEL input pin of the AB macro is to be fixed, or to 0 to allow register-controlled VAREFSEL setting (saves some logic tiles for a fixed system). The default value of 0 does not have a fixed VAREFSEL setting.
FIXED_VAREFSEL_VAL	0 or 1	Fixed VAREFSEL Value: Set this to a 1-bit constant value (set in higher-level wrapper). Set to 1 to select external VAREF as input, 0 to select internal VAREF as output. **Note:** This is only used if FIXED_VAREFSEL = 1.
FIXED_MODE	0 or 1	Set this to 1 if the MODE[3:0] pins of the AB macro are a fixed constant, or to 0 to allow register-controlled MODE[3:0] setting (saves some logic tiles for a fixed system). The default of 0 does not have a fixed MODE[3:0] setting.
FIXED_MODE_VAL	0 to 2, 4 to 6, 8 to 10, 12 to 14*	Fixed MODE Value: Set this to the integer value of the 4-bit fixed constant connected to the MODE[3:0] input pins of the AB macro (controls ADC resolution, etc.). **Note:** This is only used if FIXED_MODE = 1.
FIXED_TVC	0 or 1	Set this to 1 if the TVC[7:0] pins of the AB macro are a fixed constant, or to 0 to allow register-controlled TVC[7:0] setting (saves some logic tiles for a fixed system). The default value of 0 does not have a fixed TVC[7:0] setting.
FIXED_TVC_VAL	0 to 255	Fixed TVC Value: Set this to the integer value of the 8-bit fixed constant connected to the TVC[7:0] inputs of the AB macro. **Note:** This is only used if FIXED_TVC = 1.
FIXED_STC	0 or 1	Set this to 1 if the STC[7:0] pins of the AB macro are a fixed constant, or to 0 to allow register-controlled STC[7:0] setting (saves some logic tiles for a fixed system). The default value of 0 does not have a fixed STC[7:0] setting.

Note: *Invalid values are 3, 7, 11, and 15 (refer to the *Fusion datasheet*).

Table 4 • **CoreAI Parameters/Generics Descriptions (Continued)**

Name	Valid Range	Description
FIXED_STC_VAL	0 to 255	Fixed STC Value: Set this to the integer value of the 8-bit fixed constant connected to the STC[7:0] inputs of the AB macro. **Note:** This is only used if FIXED_STC = 1.
DISABLE_TMSTBINT	0 or 1	Disable Internal Temperature Monitor Strobe: Set this 1-bit integer to disable the internal temperature monitor strobes. If this bit is set to 1, the TMSTBINT input is disabled; if it is set to 0, the TMSTBINT input is register-controlled. The default value of 0 allows the internal temperature monitor strobe to be controlled via register settings.
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: Set each of these 10-bit integers to configure the AV9 through AV0 inputs 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 AVx input, shown in Table 5 on page 9. For example, if CFG_AV5 is set to the binary value 0000000000 (decimal 0), the AV5 input will be used as an analog voltage monitor input. If CFG_AV5 is set to the binary value 0100000000 (decimal 256), 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 binary value 1100000000 (decimal 768), 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: Set each of these 10-bit integers to configure the AC9 through AC0 inputs 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 ACx input, shown in Table 6 on page 9. For example, if CFG_AC7 is set to the binary value 0000000000 (decimal 0), the AC7 input will be used as an analog current monitor input. If CFG_AC7 is set to the binary value 0100000000 (decimal 256), 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 binary value 1000000000 (decimal 512), the AC7 input will be used as an analog voltage monitor input. If CFG_AC7 is set to the binary value 1100000000 (decimal 768), 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.

Note: *Invalid values are 3, 7, 11, and 15 (refer to the *Fusion datasheet*).

Table 4 • **CoreAI Parameters/Generics Descriptions (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: Set each of these 10-bit integers to configure the AT9 through AT0 inputs 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 ATx input, shown in Table 7 on page 10. For example, if CFG_AT2 is set to the binary value 0000000000 (decimal 0), the AT2 input will be used as an analog temperature monitor input. If CFG_AT2 is set to the binary value 0100000000 (decimal 256), 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 binary value 1000000000 (decimal 512), the AT2 input will be used as an analog voltage monitor input. If CFG_AT2 is set to the binary value 1100000000 (decimal 768), 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 Gate Drivers: Set each of these 10-bit integers to configure the GDON9 through GDON0 inputs and the AG9 through AG0 outputs 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 shown in Table 8 on page 10. For example, if CFG_GD1 is set to the binary value 0000000000 (decimal 0), the 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 binary value 0100000000 (decimal 256), the DDGDON1 input (connected to the user's own logic) will be used to directly control the AG1 gate-driver output. If CFG_GD1 is set to the binary value 1100000000 (decimal 768), the 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.

Note: *Invalid values are 3, 7, 11, and 15 (refer to the *Fusion datasheet*).

Table 5 • **Settings for AVx Inputs**

CFG_AVx Bit-Position Values ('X' indicates "don't care")										AVx Input Usage
9	8	7	6	5	4	3	2	1	0	
0	0	X	X	X	X	X	X	X	X	Voltage Monitor (default)
0	1	X	X	X	X	X	X	X	X	Digital Input
1	0	X	X	X	X	X	X	X	X	Reserved (unused)
1	1	X	X	X	X	X	X	X	X	Disabled

Table 6 • **Settings for ACx Inputs**

CFG_ACx Bit-Position Values ('X' indicates "don't care")										ACx Input Usage
9	8	7	6	5	4	3	2	1	0	
0	0	X	X	X	X	X	X	X	X	Current Monitor (default)
0	1	X	X	X	X	X	X	X	X	Digital Input
1	0	X	X	X	X	X	X	X	X	Voltage Monitor
1	1	X	X	X	X	X	X	X	X	Disabled

Table 7 • **Settings for ATx Inputs**

CFG_ATx Bit-Position Values ('X' indicates "don't care")										ATx Input Usage
9	8	7	6	5	4	3	2	1	0	
0	0	X	X	X	X	X	X	X	X	Temperature Monitor (default)
0	1	X	X	X	X	X	X	X	X	Digital Input
1	0	X	X	X	X	X	X	X	X	Voltage Monitor
1	1	X	X	X	X	X	X	X	X	Disabled

Table 8 • **Settings for Gate Driver Usage**

CFG_GDx Bit-Position Values ('X' indicates "don't care")										Gate Driver Usage
9	8	7	6	5	4	3	2	1	0	
0	0	X	X	X	X	X	X	X	X	Register controlled (default)
1	0	X	X	X	X	X	X	X	X	Gate Driver (AGx) controlled by DDGDONx input
1	0	X	X	X	X	X	X	X	X	Reserved (unused)
1	1	X	X	X	X	X	X	X	X	Disabled (GDONx and AGx)

Internal CoreAI Registers

The internal register address map and reset values of each APB-accessible register for CoreAI are shown in Table 9 on page 11. In the case of 8-bit APB reads and writes (APB_16BIT_DATA = 0), each PADDR[4:0] address is significant and contains the byte listed in the far right column; however, when 16-bit APB reads and writes are used (APB_16BIT_DATA = 1), PADDR[4:1] addresses contain one 16-bit word, where each two adjacent rows in Table 9 on page 11 contain the low-order and high-order bytes, consecutively (PADDR[0] is unused in this case).

Table 10 on page 12 through Table 31 on page 17 describe 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. All registers are listed assuming 8-bit APB reads and writes; if 16-bit APB reads and writes are used, add 8 to each item in the "Bits" column for high-order bytes (Table 11 on page 12, Table 15 on page 13, Table 17 on page 13, Table 19 on page 14, Table 21 on page 14, Table 23 on page 14, Table 25 on page 15, Table 29 on page 16, and Table 31 on page 17).

When reading from register bits that are write-only or unused (reserved), a 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[14:0]) ANDed with the interrupt status bits (INT[14:0]), as shown in Figure 4.

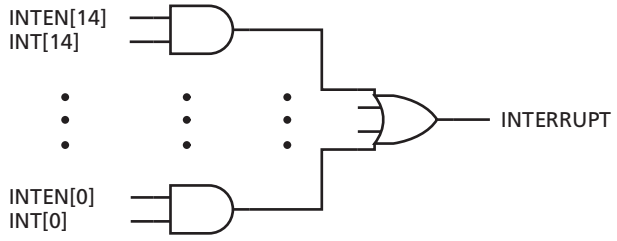


Figure 4 • **CoreAI Interrupt Logic**

Note that after an APB read of the interrupt status registers has been performed, each INT[14:0] 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 9 • **CoreAI Internal Register Address Map***

PADDR[4:0]	Type	Reset Value	Brief Description
0x00	R/W	0x04	ACM Control/Status Register (low-order)
0x01	R/W	0x00	ACM Control/Status Register (high-order)
0x02	R/W	0x00	ACM Address Register
0x03	–	0x00	Reserved
0x04	R/W	0x00	ACM Data Register
0x05	–	0x00	Reserved
0x06	R/W	0x00	ADC Control Register 1 (low-order)
0x07	R/W	0x00	ADC Control Register 1 (high-order)
0x08	R/W	0x00	ADC Control Register 2 (low-order)
0x09	R/W	0x00	ADC Control Register 2 (high-order)
0x0a	R/W	0x00	ADC Control Register 3 (low-order)
0x0b	R/W	0x00	ADC Control Register 3 (high-order)
0x0c	R/W	0x00	ADC Control Register 4 (low-order)
0x0d	R/W	0x00	ADC Control Register 4 (high-order)
0x0e	R/W	0x00	ADC Control Register 5 (low-order)
0x0f	R/W	0x00	ADC Control Register 5 (high-order)
0x10	R	0x00	ADC Status Register (low-order)
0x11	R	0x00	ADC Status Register (high-order)
0x12	R	0x00	Read FIFO Data Output
0x13	–	0x00	Reserved
0x14	R	0x0c	Read FIFO Status
0x15	–	0x00	Reserved
0x16	R/W	0x00	Interrupt Enable Register (low-order)
0x17	R/W	0x00	Interrupt Enable Register (high-order)
0x18	R	0x00	Interrupt Status Register (low-order)
0x19	R	0x00	Interrupt Status Register (high-order)

Note: *Values shown in hexadecimal format. Type designations: "R" – read-only, "R/W" – read/write, "–" – Not used (to accommodate 16-bit APB access)

Table 10 • **ACM Control/Status Register (low-order)**

Bits	Name	Function
7:5	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 11 • **ACM Control/Status Register (high-order)**

Bits	Name	Function
7:0	Reserved	Not used

Table 12 • **ACM Address Register**

Bits	Name	Function
7:0	ACMADDR[7:0]	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 13 • **ACM Data Register**

Bits	Name	Function
7:0	ACMDATA[7:0]	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 14 • **ADC Control Register 1 (low-order)**

Bits	Name	Function
7	Reserved	Not used
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	PWRDN	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[3:0]	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 15 • **ADC Control Register 1 (high-order)**

Bits	Name	Function
7:0	TVC[7:0]	ADC Clock Divider (Refer to the *Fusion datasheet* for further information.)

Table 16 • **ADC Control Register 2 (low-order)**

Bits	Name	Function
7:0	STC[7:0]	ADC Sample Time Control (Refer to the *Fusion datasheet* for further information.)

Table 17 • **ADC Control Register 2 (high-order)**

Bits	Name	Function
7:6	Reserved	Not used
5	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)
4:0	CHNUMBER[4:0]	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.)

Table 18 • **ADC Control Register 3 (low-order)**

Bits	Name	Function
7:0	CMSTB[7:0]	Current Monitor Strobes These bits are connected to the CMSTB[7:0] pins of the AB macro. (Refer to the *Fusion datasheet* for further information.)

Table 19 • **ADC Control Register 3 (high-order)**

Bits	Name	Function
7:2	Reserved	Not used
1:0	CMSTB[9:8]	Current Monitor Strobes These bits are connected to the CMSTB9 and CMSTB8 pins of the AB macro. (Refer to the *Fusion datasheet* for further information.)

Table 20 • **ADC Control Register 4 (low-order)**

Bits	Name	Function
7:0	TMSTB[7:0]	Temperature Monitor Strobes These bits are connected to the TMSTB[7:0] pins of the AB macro. (Refer to the *Fusion datasheet* for further information.)

Table 21 • **ADC Control Register 4 (high-order)**

Bits	Name	Function
7:2	Reserved	Not used
2	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.)
1:0	TMSTB[9:8]	Temperature Monitor Strobes These bits are connected to the TMSTB9 and TMSTB8 pins of the AB macro. (Refer to the *Fusion datasheet* for further information.)

Table 22 • **ADC Control Register 5 (low-order)**

Bits	Name	Function
7:0	GDON[7:0]	Gate Driver Enables These bits are connected to the GDON[7:0] pins of the AB macro. Note that the CFG_GDx parameters/generics settings affect whether or not the GDON[7:0] register bits are used. (Refer to the *Fusion datasheet* for further information.)

Table 23 • **ADC Control Register 5 (high-order)**

Bits	Name	Function
7:2	Reserved	Not used
1:0	GDON[9:8]	Gate Driver Enables These bits are connected to the GDON9 and GDON8 pins of the AB macro. Note that the CFG_GDx parameters/generics settings affect whether or not the GDON[9:8] register bits are used. (Refer to the *Fusion datasheet* for further information.)

Table 24 • **ADC Status Register (low-order)**

Bits	Name	Function
7:0	RESULT[7:0]	ADC Result (low-order) These bits come from the RESULT[7:0] bits of the AB macro. (Refer to the *Fusion datasheet* for further information.)

Table 25 • **ADC Status Register (high-order)**

Bits	Name	Function
7	CALIBRATE	ADC Calibrate 1 – The ADC is busy performing its calibration. 0 – The ADC is calibrated. (Refer to the *Fusion datasheet* for further information.)
6	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.)
5	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.)
4	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.)
3:0	RESULT[11:8]	ADC Result (high-order) These bits come from the RESULT[11:8] bits of the AB macro. (Refer to the *Fusion datasheet* for further information.)

Table 26 • **Read FIFO Data Output Register**

Bits	Name	Function
7:0	DOUT[7:0]	Read FIFO Data Output If CoreAI is used in 8-bit APB read/write mode, the read FIFO must be read twice to retrieve one ADC sample of result data (two bytes); in this case, the low-order byte would be read first and the high-order byte second. Note also that in 8-bit APB read/write mode, the read FIFO will not become empty until the last high-order byte has been read from the FIFO, i.e., an even number of reads must occur. If CoreAI is used in 16-bit APB read/write mode, the read FIFO will be read as one 16-bit word at once (with only the lower 12 bits being significant, since the ADC result has 12 bits of resolution); in this case, DOUT[15:0] will appear on the PRDATA[15:0] output port, rather than DOUT[7:0] appearing on PRDATA[7:0] as in 8-bit mode.

Table 27 • **Read FIFO Status Register**

Bits	Name	Function
7:4	Reserved	Not used
3	AEMPTY	Read FIFO Almost Empty 1 – The read FIFO is almost empty (based on the value of the USE_RDFIFO_AEVAL parameter/ generic). 0 – The read FIFO is not almost empty.
2	EMPTY	Read FIFO Empty 1 – The read FIFO is empty. 0 – The read FIFO is not empty.
1	AFULL	Read FIFO Almost Full 1 – The read FIFO is almost full (based on the value of the USE_RDFIFO_AFVAL parameter/generic). 0 – The read FIFO is not almost full.
0	FULL	Read FIFO Full 1 – The read FIFO is full. 0 – The read FIFO is not full.

Table 28 • **Interrupt Enable Register (low-order)**

Bits	Name	Function
7	Reserved	Not used
6:0	INTEN[6:0]	Low-Order Interrupt Enables Each of these bits is ANDed with each of the bits in the INT[6:0] register to contribute to the ORed INTERRUPT output. To mask the contribution of the corresponding bit in the INT[6:0] register, set that bit to 0; to enable the contribution, set that bit to 1. Note that the INTEN[6:0] bits only mask what appears on the INTERRUPT output and that the INT[6:0] register bits will always be active to reflect current interrupt source conditions.

Table 29 • **Interrupt Enable Register (high-order)**

Bits	Name	Function
7:0	INTEN[14:7]	High-Order Interrupt Enables Each of these bits is ANDed with each of the bits in the INT[14:7] register to contribute to the ORed INTERRUPT output. To mask the contribution of the corresponding bit in the INT[14:7] register, set that bit to 0; to enable the contribution, set that bit to 1. Note that the INTEN[14:7] bits only mask what appears on the INTERRUPT output and that the INT[14:7] register bits will always be active to reflect current interrupt source conditions.

Table 30 • **Interrupt Status Register (low-order)**

Bits	Name	Function
7	Reserved	Not used
6	INT[6]	RTC Match Rising Edge – If 1, the RTCMATCH output has transitioned high, indicating that the RTC has reached the desired count.
5	INT[5]	ACM Read Done – If 1, an ACM read cycle has completed and valid data can be read from the ACM DATA register.
4	INT[4]	ACM Write Done – If 1, an ACM write cycle has completed and valid data has been written into the AB macro.
3	INT[3]	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.
2	INT[2]	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.
1	INT[1]	CALIBRATE Falling Edge – If 1, the CALIBRATE signal from the AB block has transitioned low, indicating that the ADC has finished its calibration procedure.
0	INT[0]	CALIBRATE Rising Edge – If 1, the CALIBRATE signal from the AB block has transitioned high, indicating that the ADC has started its calibration procedure.

Table 31 • **Interrupt Status Register (high-order)**

Bits	Name	Function
7	INT[14]	Read FIFO Almost Empty Rising Edge – If 1, the Almost Empty flag from the read FIFO has transitioned high, indicating that it is almost empty.
6	INT[13]	Read FIFO Almost Empty Falling Edge – If 1, the Almost Empty flag from the read FIFO has transitioned low, indicating that it is no longer almost empty.
5	INT[12]	Read FIFO Empty Rising Edge – If 1, the Empty flag from the read FIFO has transitioned high, indicating that it is now empty.
4	INT[11]	Read FIFO Empty Falling Edge – If 1, the Empty flag from the read FIFO has transitioned low, indicating that it is no longer empty.
3	INT[10]	Read FIFO Almost Full Rising Edge – If 1, the Almost Full flag from the read FIFO has transitioned high, indicating that it is almost full.
2	INT[9]	Read FIFO Almost Full Falling Edge – If 1, the Almost Full flag from the read FIFO has transitioned low, indicating that it is no longer almost full.
1	INT[8]	Read FIFO Full Rising Edge – If 1, the Full flag from the read FIFO has transitioned high, indicating that it is now full.
0	INT[7]	Read FIFO Full Falling Edge – If 1, the Full flag from the read FIFO has transitioned low, indicating that it is no longer full.

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 SmartGen software, included with Libero IDE, 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 32.

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 clock-divider appropriately via the ACM_CLK_DIV parameter/generic. For example, if the system clock frequency (PCLK) is 50 MHz, ACM_CLK_DIV can be set to 3, which would generate an ACM clock frequency of $50/(2^3) = 6.25$ MHz.

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 10 on page 12 and Table 30 on page 17).

Table 32 • **ACM Address Map (Analog Quads and RTC)***

PADDR[4:0]	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			
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
Analog Quad 9			
0x25	AQ9	Byte 0 (AV9 control)	Analog Quad
0x26	AQ9	Byte 2 (AC9 control)	Analog Quad
0x27	AQ9	Byte 2 (AG9 control)	Analog Quad
0x28	AQ9	Byte 3 (AT9 control)	Analog Quad
0x29	–	Reserved	Unused
0x30	–	Reserved	Unused
…	…	…	Unused
0x3E	–	Reserved	Unused

Note: *Address values are given in hexadecimal format.

Table 32 • **ACM Address Map (Analog Quads and RTC)* (Continued)**

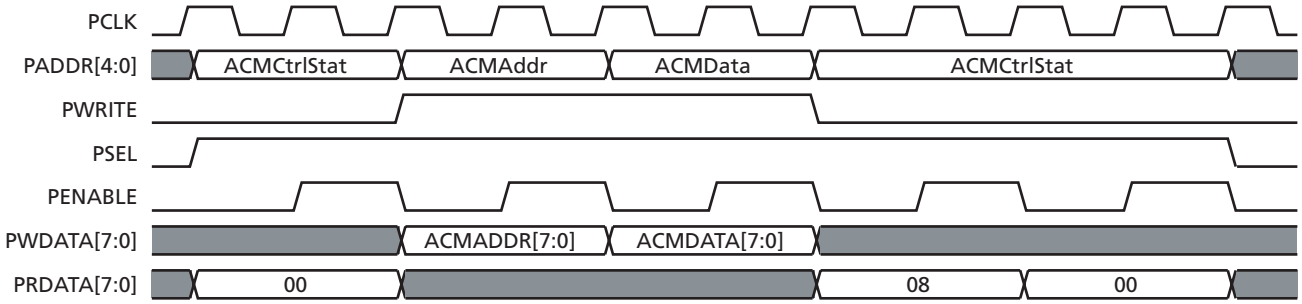
PADDR[4:0]	Name	Description	Associated Peripheral
Real-Time Counter			
0x3F	–	Reserved	Unused
0x40	COUNTER0	Counter bits 7:0	RTC
0x41	COUNTER1	Counter bits 15:8	RTC
0x42	COUNTER2	Counter bits 23:16	RTC
0x43	COUNTER3	Counter bits 31:24	RTC
0x44	COUNTER4	Counter bits 39:32	RTC
0x45	–	Reserved	Unused
0x46	–	Reserved	Unused
0x47	–	Reserved	Unused
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
0x4D	–	Reserved	Unused
0x4E	–	Reserved	Unused
0x4F	–	Reserved	Unused
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
0x55	–	Reserved	Unused
0x56	–	Reserved	Unused
0x57	–	Reserved	Unused
0x58	CTRL_STAT	Control (write) / Status (read) register bits 7:0	RTC
0x59	TEST_REG	Test register(s)	RTC
0x60	–	Reserved	Unused
0x61	–	Reserved	Unused
…	…	…	Unused
0xFF	–	Reserved	Unused

Note: *Address values are given in hexadecimal format.

ACM Writes

An example of writing data into the ACM is illustrated in Figure 5. The steps for writing information into the AB block via the ACM are as follows:

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.
4. Keep reading the ACM Control/Status register until CoreAI is not busy processing the ACM write.



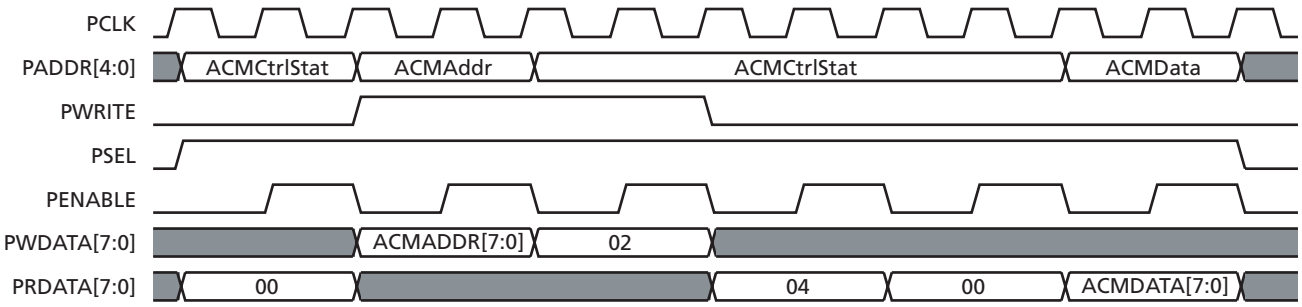
Note: 8-bit APB reads and writes are shown.

Figure 5 • **ACM Write Procedure**

ACM Reads

An example of reading data from the ACM is illustrated in Figure 6. The steps for reading information from the AB block via the ACM are as follows:

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 the ACM Control/Status register until CoreAI is not busy processing the ACM read.
5. Read the ACM Data register to read the byte of ACM data that is now available from the AB block.



Note: 8-bit APB reads and writes are shown.

Figure 6 • **ACM Read Procedure**

RTC Operation

For detailed data on the connection and operation of the 40-bit real-time counter block within the AB macro, refer to the *Fusion datasheet*.

The SmartGen software can be used to configure the RTC. The internal registers of the RTC are accessed via the ACM interface address map listed in Table 32 on page 18. Control of the RTC is primarily accomplished via writes to the RTC CTRL_STAT (Control/Status) register at ACM address 0x58 (88), which is listed in Table 33. Status information is primarily obtained by reading the same register.

Table 33 • **RTC Control/Status Register**

Bits	Name	Function
7	rtc_rst	RTC Reset: Writing logic 1 to this bit causes an RTC reset.[1] Writing logic 0 to this bit will allow synchronous removal of reset after two ACM_CLK cycles if VCC33UP = 1.[2]
6	cntr_en	Counter Enable: Writing logic 1 to this bit will enable the counter if the RTC is not in reset. After reset is removed and cntr_en = 1, 64 RTCCLK positive edges (one-half of the pre-scaler division factor) will elapse before the counter is incremented.[3] A 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.
5	vr_en_mat	Voltage Regulator Enable on Match: Writing logic 1 to this bit will allow the RTCMATCH output port to transition 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 kHz crystal operation, this should be set to "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 to this bit resets the 40-bit counter value to 0. Writing logic 1 allows the counter to count.
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 a 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 a logic 1 is written to this bit, only the RTC can be used to control the RTCXTLSEL and RTCMODE[1:0] inputs of the crystal oscillator. This bit must be set to 1 to allow the RTC counter to function if the 1.5 V supply is off.

Notes:

1. Reset of all RTC states (except this Control/Status register) occurs asynchronously if VCC33UP = 0 or CTRL_STAT bit 7 (rtc_rst) is written to 1.
2. Reset is removed synchronously after 2 rising edges of the ACM_CLK, following both VCC33UP = 1 and rtc_rst = 0.
3. 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.

ADC Operation

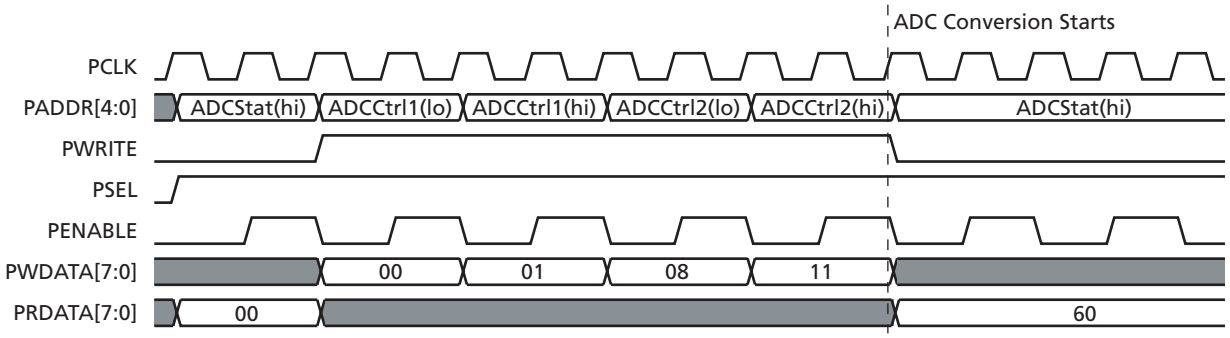
Control of the ADC within the AB macro in CoreAI is accomplished by APB reads and writes. After a power-up reset condition, the ADC will come out of its reset state and commence with its internal calibration sequence. When this calibration sequence has finished, the ADC will be ready to use for conversions.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

The user 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. The SmartGen software can be used to configure the analog quads and RTC.

ADC Control

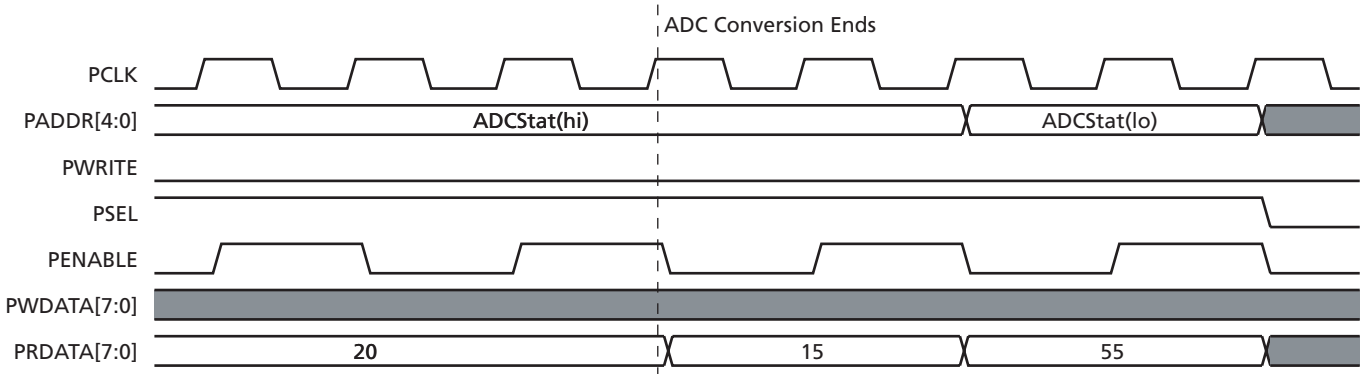
A typical analog-to-digital conversion is shown starting in Figure 7 and ending in Figure 8 on page 23. The steps for performing an ADC conversion are as follows:

1. Read the ADC Status register to ensure 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 and channel number to sample (1 of 32).
4. Set the Start Conversion bit to begin the conversion process.
5. If a current or temperature monitor operation is used on a corresponding AC or AT input pin, a write must be done to set the corresponding current 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 and temperature monitoring.
6. Keep reading the high-order ADC Status register until the ADC is not busy performing a conversion.
7. If a current or temperature monitor operation has completed using a corresponding AC or AT input pin, a write must be done to set the corresponding current or temperature monitor strobe low in ADC Control register 3 or ADC Control register 4, respectively.
8. Read the low-order ADC Status register to obtain the lower 8 bits of the resulting ADC conversion data (the upper 4 bits of conversion data have already been read from the high-order ADC Status register).



Note: 8-bit APB reads and writes are shown.

Figure 7 • **ADC Start of Conversion**



Note: 8-bit APB reads and writes are shown.

Figure 8 • **ADC End of Conversion**

ADC Result Read FIFO

An optional read FIFO, capable of holding up to 256 analog-to-digital conversions, is instantiated in CoreAI when the USE_RDFIFO parameter/generic is set to 1. The controlling element that is connected to CoreAI (internal or external microprocessor or microcontroller, or custom FPGA logic) can read resulting ADC conversions from this FIFO when desired. Resulting ADC conversion data is always written into the FIFO 12 bits at once, regardless of the ADC resolution (8-, 10-, or 12-bit), and is MSB-justified so that the resulting values are always of the same order of magnitude. The steps for reading resulting ADC conversion data from the read FIFO are as follows:

1. Read the Read FIFO Status register to check if there is data in the FIFO (not empty).
2. If the APB_16BIT_DATA parameter/generic is set to 0 (8-bit mode), read the Read FIFO Data Output register twice in a row to retrieve the lower 8 bits then the upper 4 bits of ADC conversion data. If the APB_16BIT_DATA parameter/generic is set to 1 (16-bit mode), read the Read FIFO Data Output register once to retrieve all 12 bits of ADC conversion data.
3. Repeat steps 1 and 2 until the Read FIFO is empty.

Note that although the ADC conversion results are stored as one 16-bit word, the four upper bits of this 16-bit word are reserved for possible future use. Since there is no channel information associated with each ADC conversion word, it is up to the user to maintain temporal coherency between conversions and resulting data.

Interrupt Logic

Although polling operations have been described to control the ACM and ADC, wherein status registers are periodically read (polled), the INTERRUPT output may be connected to a microprocessor or microcontroller to indicate similar events (e.g., ACM reads and writes completing, ADC calibration events, read FIFO events, etc.). In this case, interrupt service routines may be written in software to determine what actions, if any, should be taken for each of the interrupt events. Any or all interrupt events can be disabled by writing logic 0 values to the INTEN[14:0] register bits to mask the effect of each of the interrupt sources; it is the responsibility of the microprocessor/microcontroller software developer to determine what should or should not be masked.

Ordering Information

Order CoreAI through your local Actel sales representative. Use the following naming convention when ordering: CoreAI-XX, where XX is listed in Table 34.

Table 34 • **Ordering Codes**

Traditional (FTP) Delivery		**CoreConsole Delivery**	
EV	Evaluation version	Evaluations included with CoreConsole	
SN	Single use netlist	OC	Obfuscated RTL – 1 year license
AN	Multiple use netlist	OM	Obfuscated RTL – multiple use license
SR	Single use RTL	RC	RTL source – 1 year license
AR	Multiple use RTL	RM	RTL source – multiple use license
UR	Unrestricted technology (on request)		
UG	Upgrade		
M	Maintenance		

Datasheet Categories

In order to provide the latest information to designers, some datasheets are published before data has been fully characterized. Datasheets are designated as "Product Brief," "Advanced," and "Production." The definitions of these categories are as follows:

Product Brief

The product brief is a summarized version of an advanced or production datasheet containing general product information. This brief summarizes specific device and family information for unreleased products.

Advanced

This datasheet version contains initial estimated information based on simulation, other products, devices, or speed grades. This information can be used as estimates, but not for production.

Unmarked (production)

This datasheet version contains information that is considered to be final.

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

51700066-0/3.06