SECTION 8
QUEUED ANALOG-TO-DIGITAL CONVERTER MODULE

This section is an overview of the queued analog-to-digital converter (QADC) module. Refer to the ***QADC Reference Manual*** (QADCRM/AD) for a comprehensive discussion of QADC capabilities.

8.1 General

The QADC consists of an analog front-end and a digital control subsystem, which includes an intermodule bus (IMB) interface block. Refer to **Figure 8-1**.

The analog section includes input pins, an analog multiplexer, and two sample and hold analog circuits. The analog conversion is performed by the digital-to-analog converter (DAC) resistor-capacitor array and a high-gain comparator.

The digital control section contains the conversion sequencing logic, channel selection logic, and a successive approximation register (SAR). Also included are the periodic/interval timer, control and status registers, the conversion command word (CCW) table RAM, and the result word table RAM.

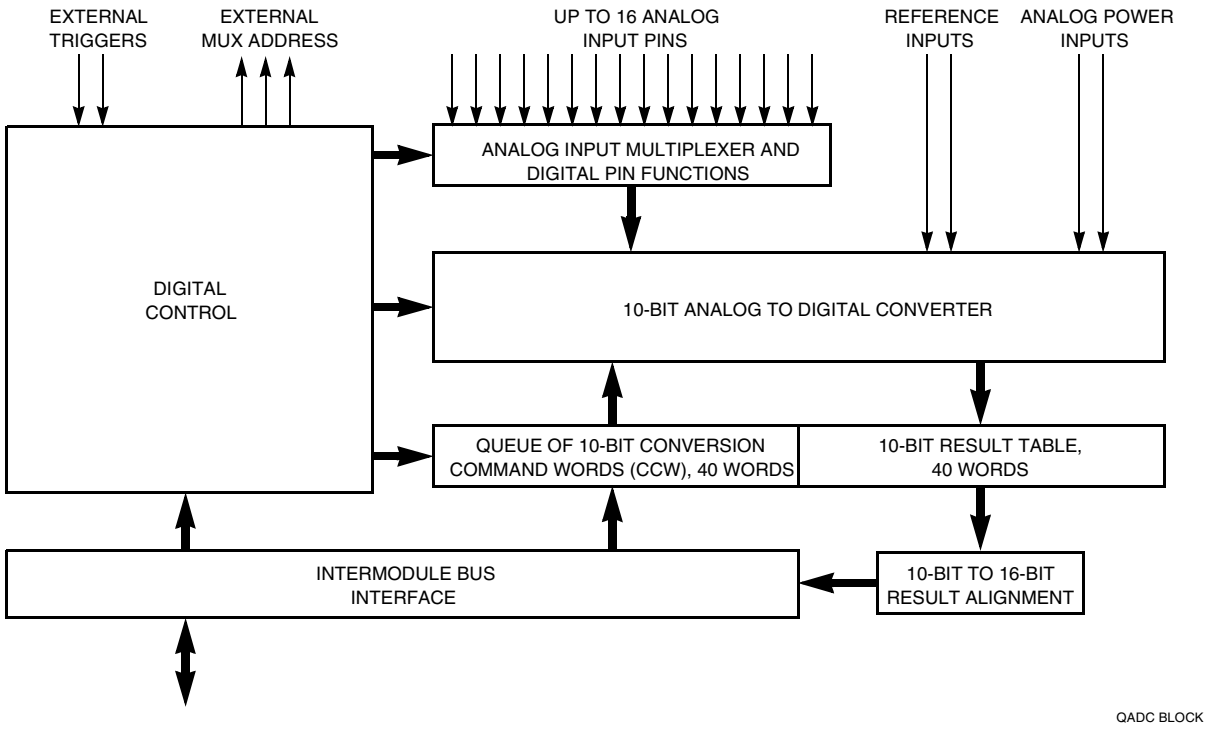


Figure 8-1 QADC Block Diagram

8.2 QADC Address Map

The QADC occupies 512 bytes of address space. Nine words are control, port, and status registers, 40 words are the CCW table, and 120 words are the result word table because 40 result registers can be read in three data alignment formats. The remaining words are reserved for expansion. Refer to **D.5 QADC Module** for information concerning the QADC address map.

8.3 QADC Registers

The QADC has three global registers for configuring module operation: the module configuration register (QADCMCR), the interrupt register (QADCINT), and a test register (QADCTEST). The global registers are always defined to be in supervisor data space. The CPU32 allows software to establish the global registers in supervisor data space and the remaining registers and tables in user space.

All QADC analog channel/port pins that are not used for analog input channels can be used as digital port pins. Port values are read/written by accessing the port A and B data registers (PORTQA and PORTQB). Port A pins are specified as inputs or outputs by programming the port data direction register (DDRQA). Port B is an input only port.

The four remaining control registers configure the operation of the queuing mechanism, and provide a means of monitoring the operation of the QADC. Control register 0 (QACR0) contains hardware configuration information. Control register 1 (QACR1) is associated with queue 1, and control register 2 (QACR2) is associated with queue 2. The status register (QASR) provides visibility on the status of each queue and the particular conversion that is in progress.

Following the register block in the address map is the CCW table. There are 40 words to hold the desired analog conversion sequences. Each CCW is a 16-bit word, with ten implemented bits in four fields. Refer to **D.5.8 Conversion Command Word Table** for more information.

The final block of address space belongs to the result word table, which appears in three places in the memory map. Each result word table location holds one 10-bit conversion value. The software selects one of three data formats, which map the 10-bit result onto the 16-bit data bus by reading the address which produces the desired alignment. The first address block presents the result data in right justified format, the second block is presented in left justified signed format, and the third is presented in left justified unsigned format. Refer to **D.5.9 Result Word Table** for more information.

8.4 QADC Pin Functions

The QADC uses a maximum of 21 external pins. There are 16 channel/port pins that can support up to 41 channels when external multiplexing is used (including internal channels). All of the channel pins can also be used as general-purpose digital port pins.

In addition, there are also two analog reference pins, two analog submodule power pins, and one V_{SS} pin for the open drain output drivers on port A.

The QADC allows external trigger inputs and the multiplexer outputs to be combined onto some of the channel pins. All of the channel pins are used for at least two functions, depending on the modes in use.

The following paragraphs describe QADC pin functions. **Figure 8-2** shows the QADC module pins.

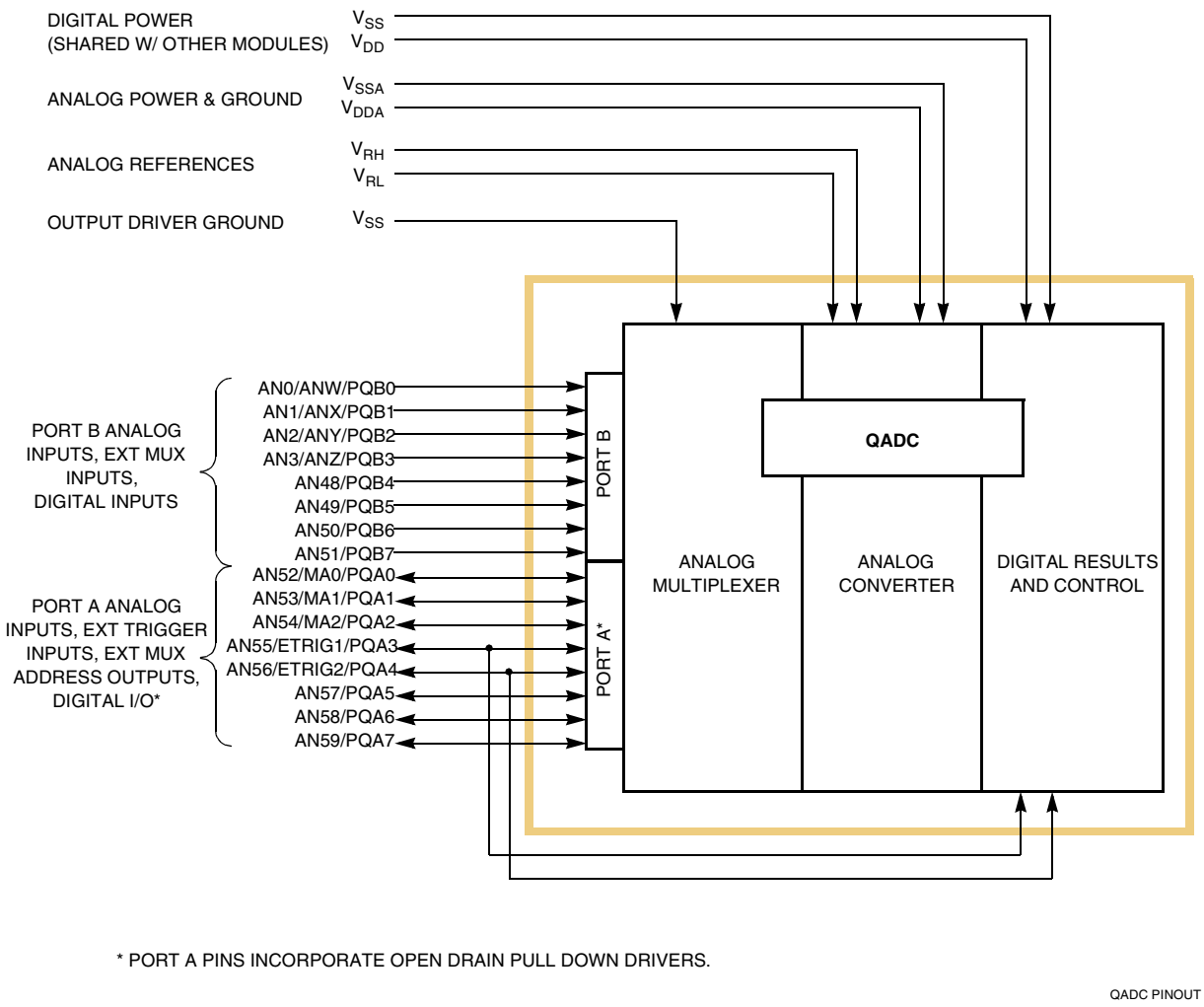


Figure 8-2 QADC Input and Output Signals

8.4.1 Port A Pin Functions

The eight port A pins can be used as analog inputs, or as a bidirectional 8-bit digital input/output port. Refer to the following paragraphs for more information.

8.4.1.1 Port A Analog Input Pins

When used as analog inputs, the eight port A pins are referred to as AN[59:52]. Due to the digital output drivers associated with port A, the analog characteristics of port A

are different from those of port B. All of the analog signal input pins may be used for at least one other purpose.

8.4.1.2 Port A Digital Input/Output Pins

Port A pins are referred to as PQA[7:0] when used as a bidirectional 8-bit digital input/output port. These eight pins may be used for general-purpose digital input signals or digital open drain pull-down output signals.

Port A pins are connected to a digital input synchronizer during reads and may be used as general purpose digital inputs.

Each port A pin is configured as an input or output by programming the port data direction register (DDRQA). Digital input signal states are read from the PORTQA data register when DDRQA specifies that the pins are inputs. Digital data in PORTQA is driven onto the port A pins when the corresponding bits in DDRQA specify outputs. Refer to **D.5.5 Port Data Direction Register** for more information. Since the outputs are open drain drivers (so as to minimize the effects to the analog function of the pins), external pull-up resistors must be used when port A pins are used to drive another device.

8.4.2 Port B Pin Functions

The eight port B pins can be used as analog inputs, or as an 8-bit digital input only port. Refer to the following paragraphs for more information.

8.4.2.1 Port B Analog Input Pins

When used as analog inputs, the eight port B pins are referred to as AN[51:48]/AN[3:0]. Since port B functions as analog and digital input only, the analog characteristics are different from those of port A. Refer to **APPENDIX A ELECTRICAL CHARACTERISTICS** for more information on analog signal characteristics. All of the analog signal input pins may be used for at least one other purpose.

8.4.2.2 Port B Digital Input Pins

Port B pins are referred to as PQB[7:0] when used as an 8-bit digital input only port. In addition to functioning as analog input pins, the port B pins are also connected to the input of a synchronizer during reads and may be used as general-purpose digital inputs.

Since port B pins are input only, there is no associated data direction register. Digital input signal states are read from the PORTQB data register. Refer to **D.5.5 Port Data Direction Register** for more information.

8.4.3 External Trigger Input Pins

The QADC has two external trigger pins (ETRIG[2:1]). The external trigger pins share two multifunction port A pins (PQA[4:3]), which are normally used as analog channel input pins. Each of the two external trigger pins is associated with one of the scan queues. When a queue is in external trigger mode, the corresponding external trigger

pin is configured as a digital input and the software programmed input/output direction for that pin is ignored. Refer to **D.5.5 Port Data Direction Register** for more information.

8.4.4 Multiplexed Address Output Pins

In non-multiplexed mode, the 16 channel pins are connected to an internal multiplexer which routes the analog signals into the A/D converter.

In externally multiplexed mode, the QADC allows automatic channel selection through up to four external 1-of-8 multiplexer chips. The QADC provides a 3-bit multiplexed address output to the external mux chips to allow selection of one of eight inputs. The multiplexed address output signals MA[2:0] can be used as multiplex address output bits or as general-purpose I/O.

MA[2:0] are used as the address inputs for up to four 1-of-8 multiplexer chips (for example, the MC14051 and the MC74HC4051). Since MA[2:0] are digital outputs in multiplexed mode, the software programmed input/output direction for these pins in DDRQA is ignored.

8.4.5 Multiplexed Analog Input Pins

In externally multiplexed mode, four of the port B pins are redefined to each represent a group of eight input channels. Refer to **Table 8-1**.

The analog output of each external multiplexer chip is connected to one of the AN[w, x, y, z] inputs in order to convert a channel selected by the MA[2:0] multiplexed address outputs.

Table 8-1 Multiplexed Analog Input Channels

Multiplexed Analog Input	Channels
ANw	Even numbered channels from 0 to 14
ANx	Odd numbered channels from 1 to 15
ANy	Even channels from 16 to 30
ANz	Odd channels from 17 to 31

8.4.6 Voltage Reference Pins

V_{RH} and V_{RL} are the dedicated input pins for the high and low reference voltages. Separating the reference inputs from the power supply pins allows for additional external filtering, which increases reference voltage precision and stability, and subsequently contributes to a higher degree of conversion accuracy. Refer to **Table A-11** and **Table A-12** for more information.

8.4.7 Dedicated Analog Supply Pins

V_{DDA} and V_{SSA} pins supply power to the analog subsystems of the QADC module. Dedicated power is required to isolate the sensitive analog circuitry from the normal levels of noise present on the digital power supply. Refer to **Table A-11** and **Table A-12** for more information.

8.4.8 External Digital Supply Pin

Each port A pin includes a digital open drain output driver, an analog input signal path, and a digital input synchronizer. The V_{SS} pin provides the ground level for the drivers on the port A pins. Since the QADC output pins have open drain type drivers, a dedicated V_{DD} pin is not needed.

8.4.9 Digital Supply Pins

V_{DD} and V_{SS} provide the power for the digital portions of the QADC, and for all other digital MCU modules.

8.5 QADC Bus Interface

The QADC can respond to byte, word, and long word accesses, however, coherency is not provided for accesses that require more than one bus cycle.

For example, if a long word read of two consecutive result registers is initiated, the QADC could change one of the result registers between the bus cycles required for each register read. All read and write accesses that require more than one 16-bit access to complete occur as two or more independent bus cycles.

Normal reads from and writes to the QADC require two clock cycles. However, if the CPU32 tries to access locations that are also accessible to the QADC while the QADC is accessing them, the bus cycle will require additional clock cycles. The QADC may insert from one to four wait states in the process of a CPU32 read from or write to such a location.

8.6 Module Configuration

The QADC module configuration register (QADCMCR) defines freeze and stop mode operation, supervisor space access, and interrupt arbitration priority. Unimplemented bits read zero and writes have no effect. QADCMCR is typically written once when software initializes the QADC, and not changed thereafter. Refer to **D.5.1 QADC Module Configuration Register** for register and bit descriptions.

8.6.1 Low-Power Stop Mode

When the STOP bit in QADCMCR is set, the clock signal to the A/D converter is disabled, effectively turning off the analog circuitry. This results in a static, low power consumption, idle condition. Low-power stop mode aborts any conversion sequence in progress. Because the bias currents to the analog circuits are turned off in low-power stop mode, the QADC requires some recovery time (t_{SR} in **APPENDIX A ELECTRICAL CHARACTERISTICS**) to stabilize the analog circuits after the STOP bit is cleared.

In the low-power stop mode, QADCMCR, the interrupt register (QADCINT), and the test register (QADCTEST) are not reset and fully accessible. The data direction register (DDRQA) and port data registers (PORTQA and PORTQB) are not reset and are read-only accessible. Control register 0 (QACR0), control register 1 (QACR1), control register 2 (QACR2), and status register (QASR) are reset and are read-only accessi-

ble. The CCW table and result table are not reset and not accessible. In addition, the QADC clock (QCLK) and the periodic/interval timer are held in reset during low-power stop mode.

If the STOP bit is clear, low-power stop mode is disabled. Refer to **D.5.1 QADC Module Configuration Register** for more information.

8.6.2 Freeze Mode

The QADC enters freeze mode when background debug mode is enabled and a breakpoint is processed. This is indicated by assertion of the FREEZE line on the IMB. The FRZ bit in QADCMCR determines whether or not the QADC responds to an IMB FREEZE assertion. Freeze mode is useful when debugging an application.

When the IMB FREEZE line is asserted and the FRZ bit is set, the QADC finishes any conversion in progress and then freezes. Depending on when the FREEZE is asserted, there are three possible queue freeze scenarios:

- When a queue is not executing, the QADC freezes immediately.
- When a queue is executing, the QADC completes the current conversion and then freezes.
- If during the execution of the current conversion, the queue operating mode for the active queue is changed, or a queue 2 abort occurs, the QADC freezes immediately.

When the QADC enters the freeze mode while a queue is active, the current CCW location of the queue pointer is saved.

In freeze mode, the analog logic is held in reset and is not clocked. Although QCLK is unaffected, the periodic/interval timer is held in reset. External trigger events that occur during freeze mode are not recorded. The CPU32 may continue to access all QADC registers, the CCW table, and the result table. Although the QADC saves a pointer to the next CCW in the current queue, software can force the QADC to execute a different CCW by writing new queue operating modes before normal operation resumes. The QADC looks at the queue operating modes, the current queue pointer, and any pending trigger events to decide which CCW to execute.

If the FRZ bit is clear, assertion of the IMB FREEZE line is ignored. Refer to **D.5.1 QADC Module Configuration Register** for more information.

8.6.3 Supervisor/Unrestricted Address Space

The QADC memory map is divided into two segments: supervisor-only data space and assignable data space. Access to supervisor-only data space is permitted only when the CPU32 is operating in supervisor mode. Assignable data space can have either restricted to supervisor-only data space access or unrestricted supervisor and user data space accesses. The SUPV bit in QADCMCR designates the assignable space as supervisor or unrestricted.

Attempts to read supervisor-only data space when the CPU32 is not in supervisor mode causes a value of $0000 to be returned. Attempts to read assignable data space

when the CPU32 is not in supervisor mode and when the space is programmed as supervisor space, causes a value of $FFFF to be returned. Attempts to write supervisor-only or supervisor-assigned data space when the CPU32 is in user mode has no effect.

The supervisor-only data space segment contains the QADC global registers, which include QADCMCR, QADCTEST, and QADCINT. The supervisor/unrestricted space designation for the CCW table, the result word table, and the remaining QADC registers is programmable. Refer to **D.5.1 QADC Module Configuration Register** for more information.

8.6.4 Interrupt Arbitration Priority

Each module that can request interrupts, including the QADC, has an interrupt arbitration number (IARB) field in its module configuration register. Each IARB field must have a different non-zero value. During an interrupt acknowledge cycle, IARB permits arbitration among simultaneous interrupts of the same priority level.

The reset value of IARB in the QADCMCR is $0. Initialization software must set the IARB field to a non-zero value in order for QADC interrupts to be arbitrated. Refer to **D.5.1 QADC Module Configuration Register** for more information.

8.7 Test Register

The QADC test register (QADCTEST) is used only during factory testing of the MCU.

8.8 General-Purpose I/O Port Operation

QADC port pins, when used as general-purpose input, are conditioned by a synchronizer with an enable feature. The synchronizer is not enabled until the QADC decodes an IMB bus cycle which addresses the port data register to minimize the high-current effect of mid-level signals on the inputs used for analog signals. Digital input signals must meet the input low voltage (V_{IL}) or input high voltage (V_{IH}) specifications in **APPENDIX A ELECTRICAL CHARACTERISTICS**. If an analog input pin does not meet the digital input pin specifications when a digital port read operation occurs, an indeterminate state is read.

During a port data register read, the actual value of the pin is reported when its corresponding bit in the data direction register defines the pin to be an input (port A only). When the data direction bit specifies the pin to be an output, the content of the port data register is read. By reading the latch which drives the output pin, software instructions that read data, modify it, and write the result, like bit manipulation instructions, work correctly.

There are two special cases to consider for digital I/O port operation. When the MUX (externally multiplexed) bit is set in QACR0, the data direction register settings are ignored for the bits corresponding to PQA[2:0], the three multiplexed address MA[2:0] output pins. The MA[2:0] pins are forced to be digital outputs, regardless of the data direction setting, and the multiplexed address outputs are driven. The data returned

during a port data register read is the value of the multiplexed address latches which drive MA[2:0], regardless of the data direction setting.

Similarly, when an external trigger queue operating mode is selected, the data direction register setting for the corresponding pins, PQA3 and/or PQA4, is ignored. The port pins are forced to be digital inputs for ETRIG1 and/or ETRIG2. The data read during a port data register read is the actual value of the pin, regardless of the data direction register setting.

8.8.1 Port Data Register

QADC ports A and B are accessed through two 8-bit port data registers (PORTQA and PORTQB). Port A pins are referred to as PQA[7:0] when used as an 8-bit input/output port. Port A can also be used for analog inputs AN[59:52], external trigger inputs ETRIG[2:1], and external multiplexer address outputs MA[2:0].

Port B pins are referred to as PQB[7:0] when used as an 8-bit input-only digital port. Port B can also be used for non-multiplexed AN[51:48]/AN[3:0] and multiplexed ANz, ANy, ANx, ANw analog inputs.

PORTQA and PORTQB are unaffected by reset. Refer to **D.5.4 Port A/B Data Register** for register and bit descriptions.

8.8.2 Port Data Direction Register

The port data direction register (DDRQA) is associated with the port A digital I/O pins. These bidirectional pins have somewhat higher leakage and capacitance specifications. Refer to **APPENDIX A ELECTRICAL CHARACTERISTICS** for more information.

Any bit in this register set to one configures the corresponding pin as an output. Any bit in this register cleared to zero configures the corresponding pin as an input. Software is responsible for ensuring that DDRQA bits are not set to one on pins used for analog inputs. When a DDRQA bit is set to one and the pin is selected for analog conversion, the voltage sampled is that of the output digital driver as influenced by the load.

NOTE

Caution should be exercised when mixing digital and analog inputs. This should be minimized as much as possible. Input pin rise and fall times should be as large as possible to minimize AC coupling effects.

Since port B is input-only, a data direction register is not needed. Read operations on the reserved bits in DDRQA return zeros, and writes have no effect. Refer to **D.5.5 Port Data Direction Register** for register and bit descriptions.

8.9 External Multiplexing Operation

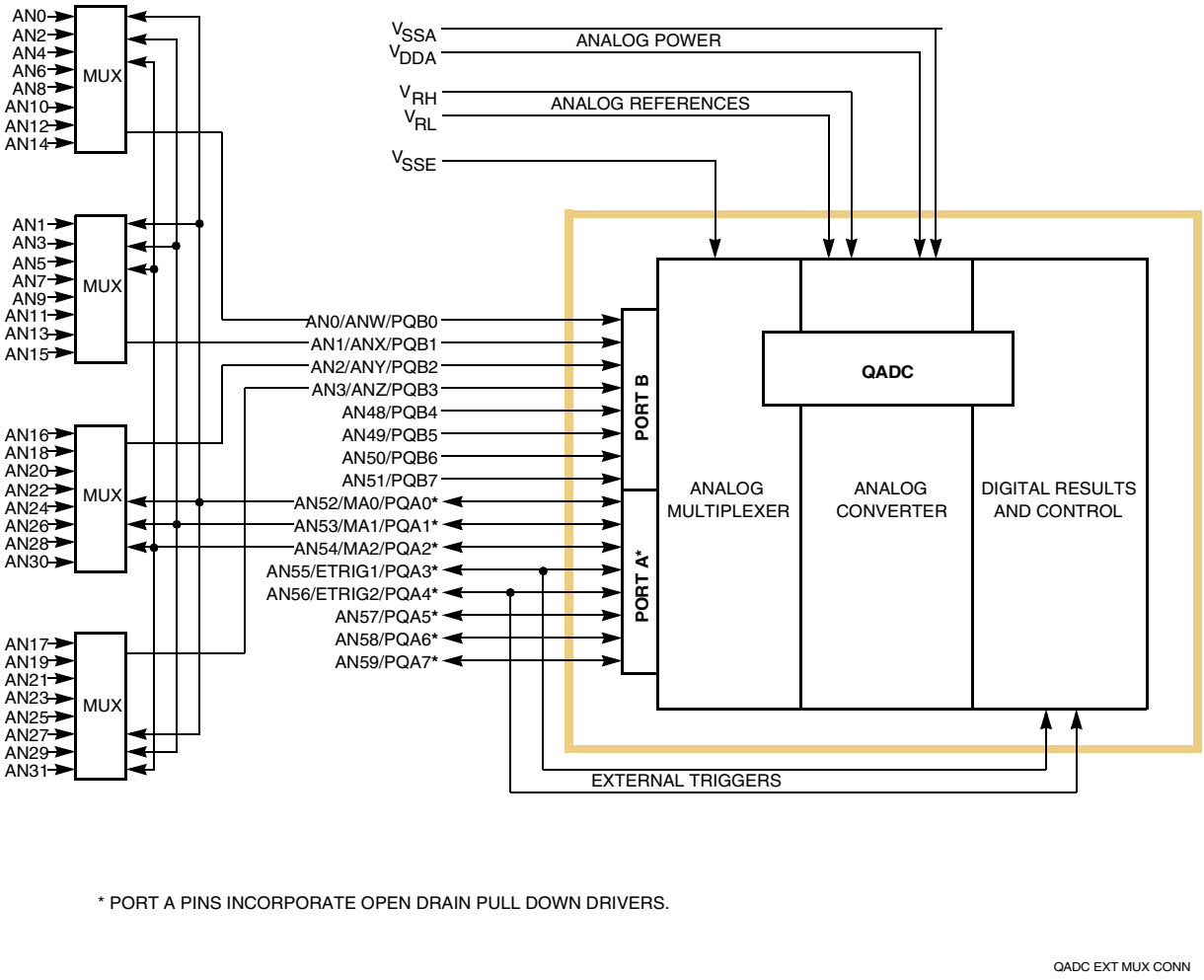
External multiplexers concentrate a number of analog signals onto a few inputs to the analog converter. This is helpful in applications that need to convert more analog sig-

nals than the A/D converter can normally provide. External multiplexing also puts the multiplexer closer to the signal source. This minimizes the number of analog signals that need to be shielded due to the close proximity of noisy, high speed digital signals near the MCU.

The QADC can use from one to four external multiplexers to expand the number of analog signals that may be converted. Up to 32 analog channels can be converted through external multiplexer selection. The externally multiplexed channels are automatically selected from the channel field of the conversion command word (CCW) table, the same as internally multiplexed channels.

All of the automatic queue features are available for externally and internally multiplexed channels. The software selects externally multiplexed mode by setting the MUX bit in QACR0.

Figure 8-3 shows the maximum configuration of four external multiplexers connected to the QADC. The external multiplexers select one of eight analog inputs and connect it to one analog output, which becomes an input to the QADC. The QADC provides three multiplexed address signals (MA[2:0]), to select one of eight inputs. These outputs are connected to all four multiplexers. The analog output of each multiplexer is each connected to one of four separate QADC inputs — ANw, ANx, ANy, and ANz.



* PORT A PINS INCORPORATE OPEN DRAIN PULL DOWN DRIVERS.

QADC EXT MUX CONN

Figure 8-3 Example of External Multiplexing

When the external multiplexed mode is selected, the QADC automatically creates the MA[2:0] open drain output signals from the channel number in each CCW. The QADC also converts the proper input channel (ANw, ANx, ANy, and ANz) by interpreting the CCW channel number. As a result, up to 32 externally multiplexed channels appear to the conversion queues as directly connected signals. Software simply puts the channel number of an externally multiplexed channel into a CCW.

Figure 8-3 shows that MA[2:0] may also be analog or digital input pins. When external multiplexing is selected, none of the MA[2:0] pins can be used for analog or digital inputs. They become multiplexed address outputs.

8.10 Analog Input Channels

The number of available analog channels varies, depending on whether or not external multiplexing is used. A maximum of 16 analog channels are supported by the internal multiplexing circuitry of the converter. **Table 8-2** shows the total number of analog input channels supported with zero to four external multiplexers.

Table 8-2 Analog Input Channels

Number of Analog Input Channels Available Directly Connected + External Multiplexed = Total Channels[1, 2]				
No External Mux Chips	**One External Mux Chip**	**Two External Mux Chips**	**Three External Mux Chips**	**Four External Mux Chips**
16	12 + 8 = 20	11 + 16 = 27	10 + 24 = 34	9 + 32 = 41

NOTES:
1. The above assumes that the external trigger inputs are shared with two analog input pins.
2. When external multiplexing is used, three input channels become multiplexed address outputs, and for each external multiplexer chip, one input channel becomes a multiplexed analog input.

8.11 Analog Subsystem

The QADC analog subsystem includes a front-end analog multiplexer, a digital to analog converter (DAC) array, a comparator, and a successive approximation register (SAR).

The analog subsystem path runs from the input pins through the input multiplexing circuitry, into the DAC array, and through the analog comparator. The output of the comparator feeds into the SAR and is considered the boundary between the analog and digital subsystems of the QADC.

Figure 8-4 shows a block diagram of the QADC analog submodule.

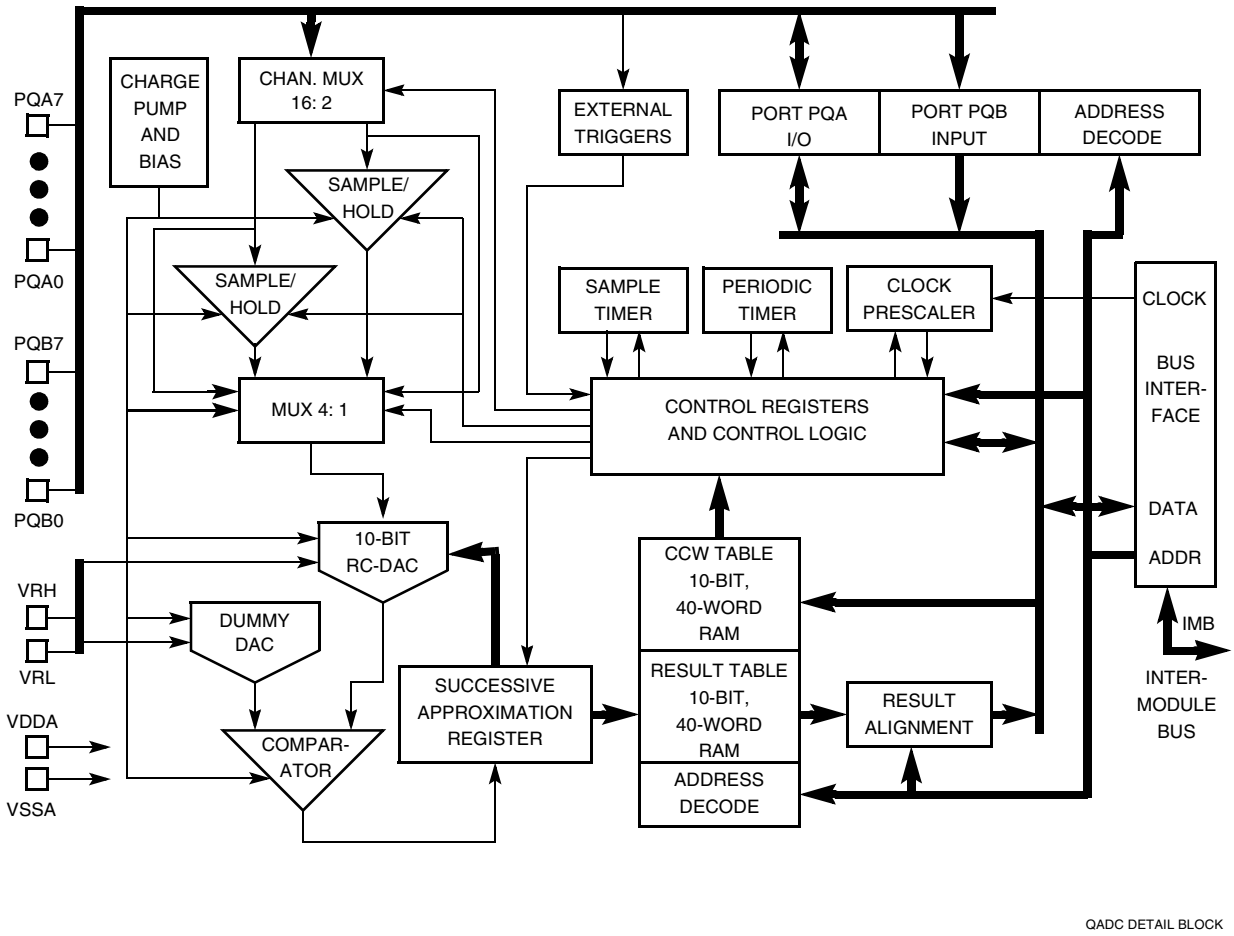


Figure 8-4 QADC Module Block Diagram

8.11.1 Conversion Cycle Times

Total conversion time is made up of initial sample time, transfer time, final sample time, and resolution time. Initial sample time refers to the time during which the selected input channel is connected to the sample capacitor at the input of the sample buffer amplifier. During the transfer period, the sample capacitor is disconnected from the multiplexer, and the stored voltage is buffered and transferred to the RC DAC array. During the final sampling period, the sample capacitor and amplifier are bypassed, and the multiplexer input charges the RC DAC array directly. During the resolution period, the voltage in the RC DAC array is converted to a digital value and stored in the SAR.

Initial sample time is fixed at two QCLKs and the transfer time at four QCLKs. Final sample time can be 2, 4, 8, or 16 ADC clock cycles, depending on the value of the IST field in the CCW. Resolution time is ten cycles.

Transfer and resolution require a minimum of 18 QCLK clocks (8.6 µs with a 2.1 MHz QCLK). If the maximum final sample time period of 16 QCLKs is selected, the total conversion time is 15.2 µs with a 2.1 MHz QCLK.

Figure 8-5 illustrates the timing for conversions. This diagram assumes a final sampling period of two QCLKs.

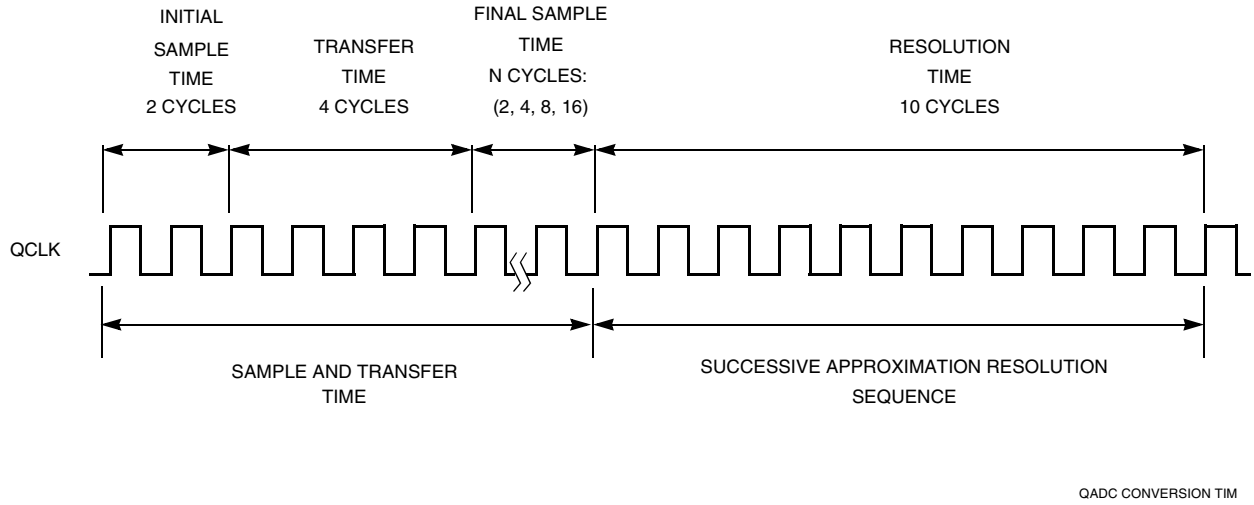


Figure 8-5 Conversion Timing

8.11.1.1 Amplifier Bypass Mode Conversion Timing

If the amplifier bypass mode is enabled for a conversion by setting the amplifier bypass (BYP) bit in the CCW, the timing changes to that shown in **Figure 8-6**. The initial sample time and the transfer time are eliminated, reducing the potential conversion time by six QCLKs. However, due to internal RC effects, a minimum final sample time of four QCLKs must be allowed. This results in a savings of four QCLKs. When using the bypass mode, the external circuit should be of low source impedance, typically less than 10 kΩ. Also, the loading effects of the external circuitry by the QADC need to be considered, since the benefits of the sample amplifier are not present.

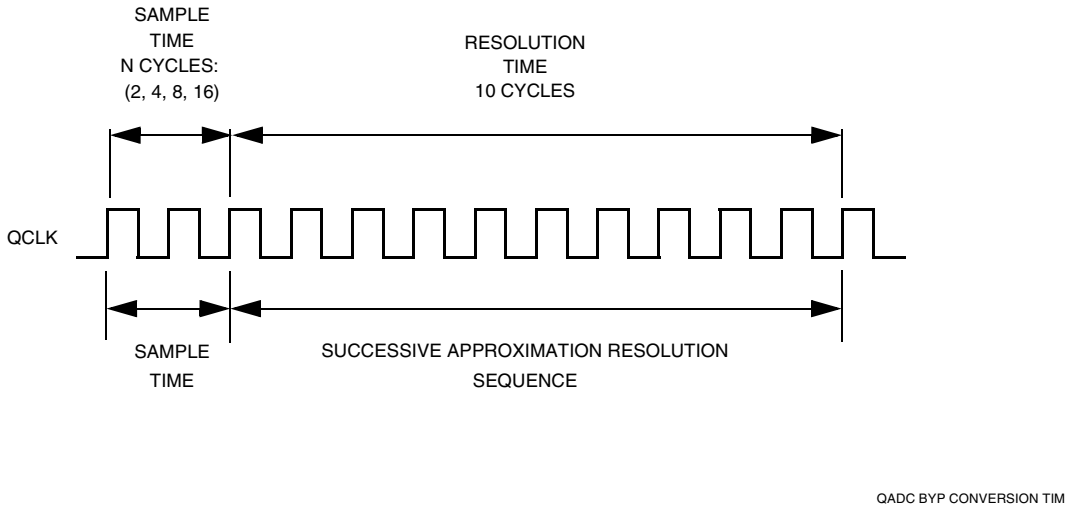


Figure 8-6 Bypass Mode Conversion Timing

8.11.2 Front-End Analog Multiplexer

The internal multiplexer selects one of the 16 analog input pins or one of three special internal reference channels for conversion. The following are the three special channels:

- V_{RH} — Reference Voltage High
- V_{RL} — Reference Voltage Low
- $V_{DDA}/2$ — Mid-Analog Supply Voltage

The selected input is connected to one side of the DAC capacitor array. The other side of the DAC array is connected to the comparator input. The multiplexer also includes positive and negative stress protection circuitry, which prevents other channels from affecting the current conversion when voltage levels are applied to the other channels. Refer to **APPENDIX A ELECTRICAL CHARACTERISTICS** for specific voltage level limits.

8.11.3 Digital to Analog Converter Array

The digital to analog converter (DAC) array consists of binary-weighted capacitors and a resistor-divider chain. The array serves two purposes:

- The array holds the sampled input voltage during conversion.
- The resistor-capacitor array provides the mechanism for the successive approximation A/D conversion.

Resolution begins with the MSB and works down to the LSB. The switching sequence is controlled by the digital logic.

8.11.4 Comparator

The comparator is used during the approximation process to sense whether the digitally selected arrangement of the DAC array produces a voltage level higher or lower

than the sampled input. The comparator output feeds into the SAR which accumulates the A/D conversion result sequentially, starting with the MSB.

8.11.5 Successive Approximation Register

The input of the successive approximation register (SAR) is connected to the comparator output. The SAR sequentially receives the conversion value one bit at a time, starting with the MSB. After accumulating the ten bits of the conversion result, the SAR data is transferred to the appropriate result location, where it may be read by user software.

8.12 Digital Control Subsystem

The digital control subsystem includes conversion sequencing logic, channel selection logic, the clock and periodic/interval timer, control and status registers, the conversion command word table RAM, and the result word table RAM.

The central element for control of the QADC conversions is the 40-entry conversion command word (CCW) table. Each CCW specifies the conversion of one input channel. Depending on the application, one or two queues can be established in the CCW table. A queue is a scan sequence of one or more input channels. By using a pause mechanism, subqueues can be created in the two queues. Each queue can be operated using several different scan modes. The scan modes for queue 1 and queue 2 are programmed in QACR1 and QACR2. Once a queue has been started by a trigger event (any of the ways to cause the QADC to begin executing the CCWs in a queue or subqueue), the QADC performs a sequence of conversions and places the results in the result word table.

8.12.1 Queue Priority

Queue 1 has execution priority over queue 2 execution. **Table 8-3** shows the conditions under which queue 1 asserts its priority:

Table 8-3 Queue 1 Priority Assertion

Queue State	Result
Inactive	A trigger event for queue 1 or queue 2 causes the corresponding queue execution to begin.
Queue 1 active/trigger event occurs for queue 2	Queue 2 cannot begin execution until queue 1 reaches completion or the paused state. The status register records the trigger event by reporting the queue 2 status as trigger pending. Additional trigger events for queue 2, which occur before execution can begin, are recorded as trigger overruns.
Queue 2 active/trigger event occurs for queue 1	The current queue 2 conversion is aborted. The status register reports the queue 2 status as suspended. Any trigger events occurring for queue 2 while queue 2 is suspended are recorded as trigger overruns. Once queue 1 reaches the completion or the paused state, queue 2 begins executing again. The programming of the resume bit in QACR2 determines which CCW is executed in queue 2.
Simultaneous trigger events occur for queue 1 and queue 2	Queue 1 begins execution and the queue 2 status is changed to trigger pending.
Subqueues paused	The pause feature can be used to divide queue 1 and/or queue 2 into multiple subqueues. A subqueue is defined by setting the pause bit in the last CCW of the subqueue.

Figure 8-7 shows the CCW format and an example of using pause to create subqueues. Queue 1 is shown with four CCWs in each subqueue and queue 2 has two CCWs in each subqueue.

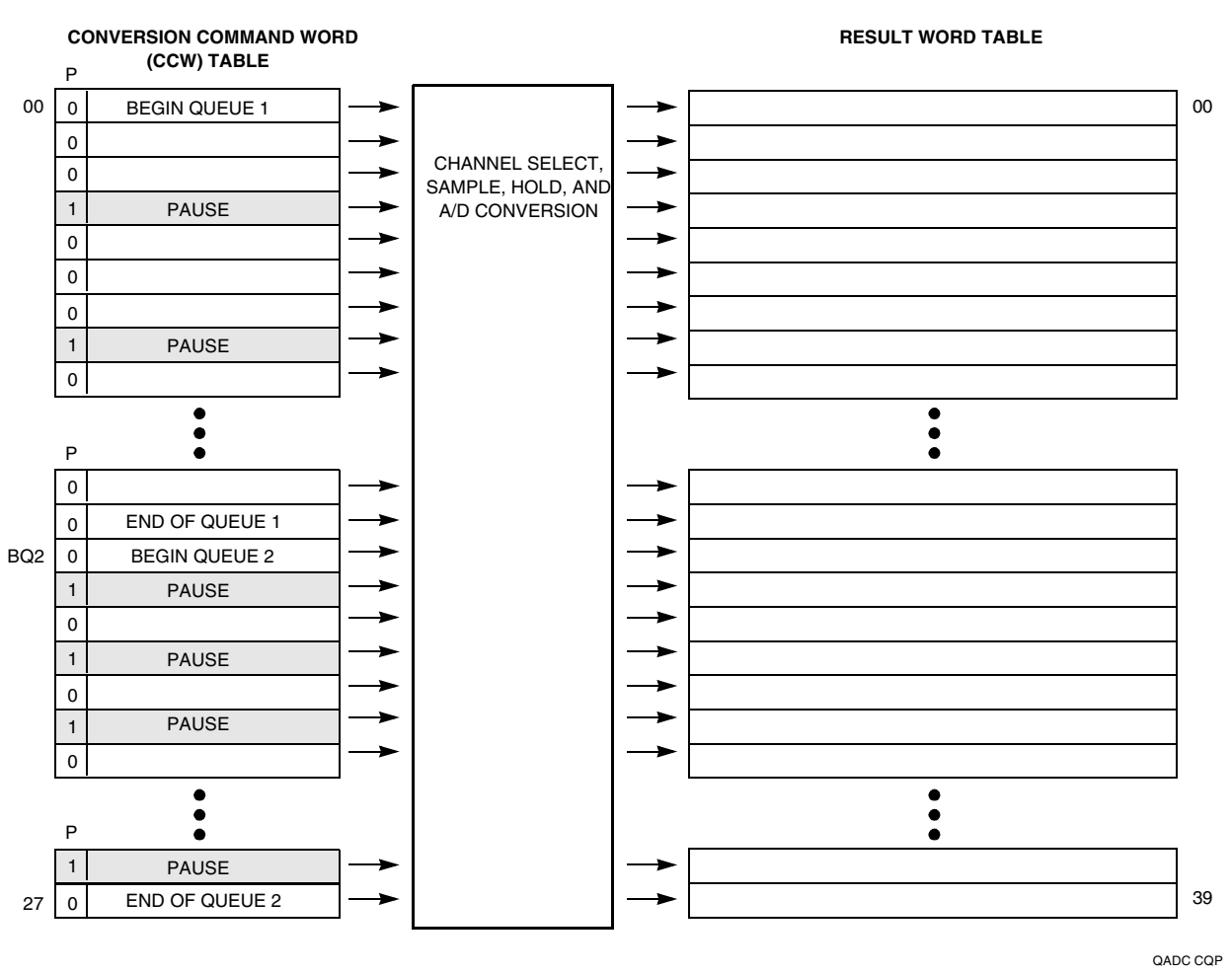


Figure 8-7 QADC Queue Operation with Pause

The queue operating mode selected for queue 1 determines what type of trigger event causes the execution of each of the subqueues within queue 1. Similarly, the queue operating mode for queue 2 determines the type of trigger event required to execute each of the subqueues within queue 2.

The choice of single-scan or continuous-scan applies to the full queue, and is not applied to each subqueue. Once a subqueue is initiated, each CCW is executed sequentially until the last CCW in the subqueue is executed and the pause state is entered. Execution can only continue with the next CCW, which is the beginning of the next subqueue. A subqueue cannot be executed a second time before the overall queue execution has been completed.

Trigger events which occur during the execution of a subqueue are ignored, except that the trigger overrun flag is set. When continuous-scan mode is selected, a trigger event occurring after the completion of the last subqueue (after the queue completion flag is set), causes execution to continue with the first subqueue, starting with the first CCW in the queue.

When the QADC encounters a CCW with the pause bit set, the queue enters the paused state after completing the conversion specified in the CCW with the pause bit. The pause flag is set and a pause software interrupt may optionally be issued. The status of the queue is shown to be paused, indicating completion of a subqueue. The QADC then waits for another trigger event to again begin execution of the next subqueue.

8.12.2 Queue Boundary Conditions

A queue boundary condition occurs when one or more of the queue operating parameters is configured in a way that will inhibit queue execution. One such boundary condition is when the first CCW in a queue specified channel 63, the end-of-queue (EOQ) code. In this case, the queue becomes active and the first CCW is read. The EOQ code is recognized, the completion flag is set, and the queue becomes idle. A conversion is not performed.

A similar situation occurs when BQ2 (beginning of queue 2 pointer) is set beyond the end of the CCW table (between $28 and $3F) and a trigger event occurs for queue 2. The EOQ condition is recognized immediately, the completion flag is set, and the queue becomes idle. A conversion is not performed.

The QADC behaves the same way when BQ2 is set to CCW0 and a trigger event occurs for queue 1. After reading CCW0, the EOQ condition is recognized, the completion flag is set, and the queue becomes idle. A conversion is not performed.

Multiple EOQ conditions may be recognized simultaneously, but the QADC will not behave differently. One example is when BQ2 is set to CCW0, CCW0 contains the EOQ code, and a trigger event occurs for queue 1. The QADC will read CCW0 and recognize the queue 1 trigger event, detecting both as EOQ conditions. The completion flag will be set and queue 1 will become idle.

Boundary conditions also exist for combinations of pause and end-of-queue. One case is when a pause bit is in one CCW and an end-of-queue condition is in the next CCW. The conversion specified by the CCW with the pause bit set completes normally. The pause flag is set. However, since the end-of-queue condition is recognized, the completion flag is also set and the queue status becomes idle, not paused. Examples of this situation include:

- The pause bit is set in CCW5 and the channel 63 (EOQ) code is in CCW6.
- The pause bit is set in CCW27.
- During queue 1 operation, the pause bit is set in CCW14 and BQ2 points to CCW15.

Another pause and end-of-queue boundary condition occurs when the pause and an end-of-queue condition occur in the same CCW. Both the pause and end-of-queue conditions are recognized simultaneously. The end-of-queue condition has precedence so a conversion is not performed for the CCW and the pause flag is not set. The QADC sets the completion flag and the queue status becomes idle. Examples of this situation are:

- The pause bit is set in CCW0A and EOQ is programmed into CCW0A.
- During queue 1 operation, the pause bit is set in CCW20, which is also BQ2.

8.12.3 Scan Modes

The QADC queuing mechanism provides several methods for automatically scanning input channels. In single-scan mode, a single pass through a sequence of conversions defined by a queue is performed. In continuous-scan mode, multiple passes through a sequence of conversions defined by a queue are executed. The following paragraphs describe the disabled/reserved, single-scan, and continuous-scan operations.

8.12.3.1 Disabled Mode and Reserved Mode

When the disabled mode or a reserved mode is selected, the queue is not active.

NOTE

Do not use a reserved mode. Unspecified operations may result.

Trigger events cannot initiate queue execution. When both queue 1 and queue 2 are disabled, no wait states will be inserted by the QADC for accesses to the CCW and result word tables. When both queues are disabled, it is safe to change the QADC clock prescaler values.

8.12.3.2 Single-Scan Modes

When application software requires execution of a single pass through a sequence of conversions defined by a queue, a single-scan queue operating mode is selected.

In all single-scan queue operating modes, software must enable a queue for execution by writing the single-scan enable bit to one in the queue's control register. The single-scan enable bits, SSE1 and SSE2, are provided for queue 1 and queue 2, respectively.

Until the single-scan enable bit is set, any trigger events for that queue are ignored. The single-scan enable bit may be set to one during the write cycle that selects the single-scan queue operating mode. The single-scan enable bit can be written as a one or a zero but is always read as a zero.

After the single-scan enable bit is set, a trigger event causes the QADC to begin execution with the first CCW in the queue. The single-scan enable bit remains set until the queue scan is complete; the QADC then clears the single-scan enable bit to zero. If the single-scan enable bit is written to one or zero before the queue scan is complete, the queue is not affected. However, if software changes the queue operating mode, the new queue operating mode and the value of the single-scan enable bit are recognized immediately. The current conversion is aborted and the new queue operating mode takes effect.

By properly programming the MQ1 field in QACR1 or the MQ2 field in QACR2, the following modes can be selected for queue 1 and/or 2:

- Software initiated single-scan mode

— Software can initiate the execution of a scan sequence for queue 1 or 2 by selecting this mode, and setting the single-scan enable bit in QACR1 or QACR2. A trigger event is generated internally and the QADC immediately begins execution of the first CCW in the queue. If a pause is encountered, queue execution ceases momentarily while another trigger event is generated internally, and then execution continues. While the time to internally generate and act on a trigger event is very short, software can momentarily read the status conditions, indicating that the queue is paused.

— The QADC automatically performs the conversions in the queue until an end-of-queue condition is encountered. The queue remains idle until software again sets the single-scan enable bit. The trigger overrun flag is never set while in this mode.

- External trigger rising or falling edge single-scan mode
  — This mode is a variation of the external trigger continuous-scan mode. It is available for both queue 1 and queue 2. Software programs the external trigger to be either a rising or a falling edge. Software must also set the single-scan enable bit for the queue in order for the scan to take place. The first external trigger edge causes the queue to be executed one time. Each CCW is read and the indicated conversions are performed until an end-of-queue condition is encountered. After the queue scan is complete, the QADC clears the single-scan enable bit. Software may set the single-scan enable bit again to allow another scan of the queue to be initiated by the next external trigger edge.
- Interval timer single-scan mode
  — In addition to the above modes, queue 2 can also be programmed for the interval timer single-scan mode. The queue operating mode for queue 2 is selected by the MQ2 field in QACR2.
  — When this mode is selected and software sets the single-scan enable bit in QACR2, the periodic/interval timer begins counting. The timer interval can range from 2^7 to 2^{17} QCLK cycles in binary multiples. When the time interval expires, a trigger event is generated internally to start the queue. The timer is reloaded and begins counting again. Meanwhile, the QADC begins execution with the first CCW in queue 2.
  — The QADC automatically performs the conversions in the queue until a pause or an end-of-queue condition is encountered. When a pause is encountered, queue execution stops until the timer interval expires again; queue execution then continues. When an end of queue condition is encountered, the timer is held in reset and the single-scan enable bit is cleared.
  — Software may set the single-scan enable bit again, allowing another scan of the queue to be initiated by the interval timer. The interval timer generates a trigger event whenever the time interval elapses. The trigger event may cause queue execution to continue following a pause, or may be considered a trigger overrun if the queue is currently executing.

8.12.3.3 Continuous-Scan Modes

When application software requires execution of multiple passes through a sequence of conversions defined by a queue, a continuous-scan queue operating mode is selected.

When a queue is programmed for a continuous-scan mode, the single-scan enable bit in the queue control register does not have any meaning or effect. As soon as the queue operating mode is programmed, the selected trigger event can initiate queue execution.

In the case of the software initiated continuous-scan mode, the trigger event is generated internally and queue execution begins immediately. In the other continuous-scan queue operating modes, the selected trigger event must occur before the queue can start. A trigger overrun is recorded if a trigger event occurs during queue execution in the external trigger continuous-scan mode and the periodic timer continuous-scan mode. When a pause is encountered during a scan, another trigger event is required for queue execution to continue. Software involvement is not required for queue execution to continue from the paused state.

After queue execution is complete, the queue status is shown as idle. Since the continuous-scan queue operating modes allow an entire queue to be scanned multiple times, software involvement is not required for queue execution to continue from the idle state. The next trigger event causes queue execution to begin again, starting with the first CCW in the queue.

NOTE

It may not be possible to guarantee coherent samples when using the continuous-scan queue operating modes since the relationship between any two conversions may be variable due to programmable trigger events and queue priorities.

By programming the MQ1 field in QACR1 or the MQ2 field in QACR2, the following modes can be selected for queue 1 and/or 2:

- Software initiated continuous-scan mode
  - — When this mode is programmed, the trigger event is generated automatically by the QADC, and queue execution begins immediately. If a pause is encountered, queue execution ceases for two QCLKs, while another trigger event is generated internally; execution then continues. When the end-of-queue is reached, another internal trigger event is generated, and queue execution begins again from the beginning of the queue.
  - — While the time to internally generate and act on a trigger event is very short, software can momentarily read the status conditions, indicating that the queue is paused or idle. The trigger overrun flag is never set while in the software initiated continuous-scan mode.
  - — This mode keeps the result registers updated more frequently than any of the other queue operating modes. Software can always read the result table to get the latest converted value for each channel. The channels scanned are kept up to date by the QADC without software involvement.
  - — This mode may be chosen for either queue, but is normally used only with queue 2. When the software initiated continuous-scan mode is chosen for queue 1, that queue operates continuously and queue 2, being lower in priority, never gets executed. The short interval of time between a queue 1 pause and

the internally generated trigger event, or between a queue 1 completion and the subsequent trigger event is not sufficient to allow queue 2 execution to begin.

- External trigger rising or falling edge continuous-scan mode
  - — The QADC provides external trigger pins for both queues. When this mode is selected, a transition on the associated external trigger pin initiates queue execution. The external trigger is programmable, so that queue execution can begin on either a rising or a falling edge. Each CCW is read and the indicated conversions are performed until an end-of-queue condition is encountered. When the next external trigger edge is detected, queue execution begins again automatically. Software initialization is not needed between trigger events.
- Periodic timer continuous-scan mode
  - — In addition to the previous modes, queue 2 can also be programmed for the periodic timer continuous-scan mode, where a scan is initiated at a selectable time interval using the on-chip periodic/interval timer. The queue operating mode for queue 2 is selected by the MQ2 field in QACR2.
  - — The QADC includes a dedicated periodic/interval timer for initiating a scan sequence for queue 2 only. A programmable timer interval can be selected ranging from 2^7 to 2^{17} times the QCLK period in binary multiples.
  - — When this mode is selected, the timer begins counting. After the programmed interval elapses, the timer generated trigger event starts the queue. The timer is then reloaded and begins counting again. Meanwhile, the QADC automatically performs the conversions in the queue until an end-of-queue condition or a pause is encountered. When a pause is encountered, the QADC waits for the periodic interval to expire again, then continues with the queue. When an end-of-queue is encountered, the next trigger event causes queue execution to begin again with the first CCW in queue 2.
  - — The periodic timer generates a trigger event whenever the time interval elapses. The trigger event may cause queue execution to continue following a pause or queue completion, or may be considered a trigger overrun. As with all continuous-scan queue operating modes, software action is not needed between trigger events.
  - — If the queue completion interrupt is enabled when using this mode, software can read the analog results that have just been collected. Software can use this interrupt to obtain non-analog inputs as well, as part of a periodic look at all inputs.

8.12.4 QADC Clock (QCLK) Generation

Figure 8-8 is a block diagram of the clock subsystem. QCLK provides the timing for the A/D converter state machine which controls the timing of conversions. QCLK is also the input to a 17-stage binary divider which implements the periodic/interval timer. To obtain the specified analog conversion accuracy, the QCLK frequency (f_{QCLK}) must be within the tolerance specified in **Table A-13**.

Before using the QADC, software must initialize the prescaler with values that put QCLK within a specified range. Though most applications initialize the prescaler once and do not change it, write operations to the prescaler fields are permitted.

CAUTION

A change in the prescaler value while a conversion is in progress is likely to corrupt the conversion result. Therefore, any prescaler write operation should be done only when both queues are disabled.

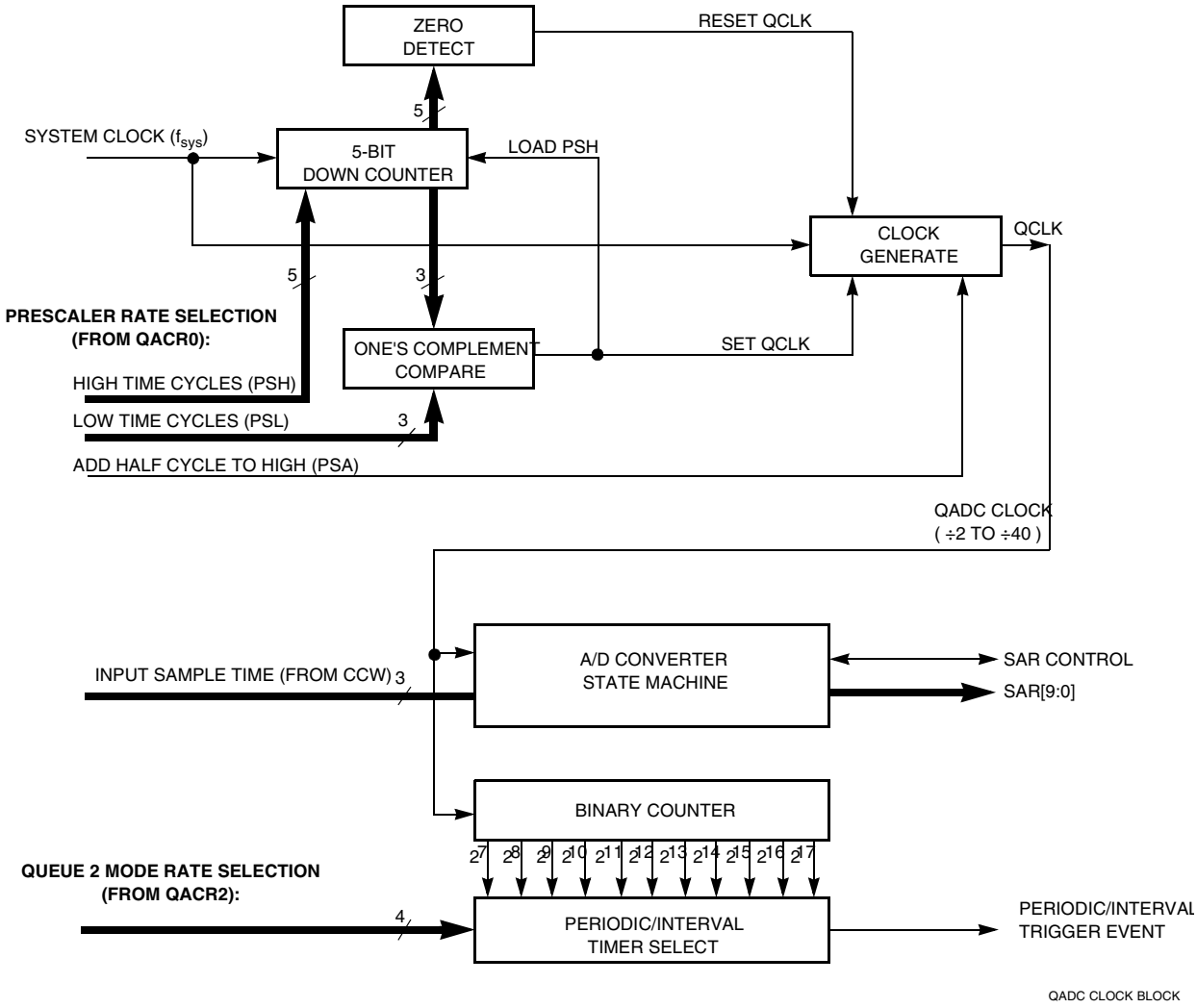


Figure 8-8 QADC Clock Subsystem Functions

To accommodate wide variations of the main MCU clock frequency f_{sys}, QCLK is generated by a programmable prescaler which divides the MCU system clock to a frequency within the specified QCLK tolerance range. The prescaler also allows the duty cycle of the QCLK waveform to be programmable.

The basic high phase of the QCLK waveform is selected with the PSH (prescaler clock high time) field in QACR0, and the basic low phase of QCLK with the PSL (prescaler clock low time) field. The duty cycle of QCLK can be further modified with the PSA (prescaler add a clock tick) bit in QACR0. The combination of the PSH and PSL parameters establishes the frequency of QCLK.

Figure 8-8 shows that the prescaler is essentially a variable pulse width signal generator. A 5-bit down counter, clocked at the system clock rate, is used to create both the high phase and the low phase of the QCLK signal. At the beginning of the high phase, the 5-bit counter is loaded with the 5-bit PSH value. When the zero detector finds that the high phase is finished, QCLK is reset. A 3-bit comparator looks for a one's complement match with the 3-bit PSL value, which is the end of the low phase of QCLK. The PSA bit allows the QCLK high-to-low transition to be delayed by a half cycle of the input clock.

The following sequence summarizes the process of determining what values are to be put into the prescaler fields in QACR0:

1. Choose the system clock frequency f_{sys}.
2. Choose first-try values for PSH, PSL, and PSA, then skip to step 4.
3. Choose different values for PSH, PSL, and PSA.
4. If the QCLK high time is less than t_{PSH} (QADC clock duty cycle – Minimum high phase time), return to step 3. Refer to **Table A-13** for more information on t_{PSH}. QCLK high time is determined by the following equation:

$$\text{QCLK high time (in ns)} = \frac{1000\ (1 + \text{PSH} + 0.5\ \text{PSA})}{f_{sys}\text{(in MHz)}}$$

   where PSH = 0 to 31 and PSA = 0 or 1.

5. If QCLK low time is less than t_{PSL} (QADC clock duty cycle – Minimum low phase time), return to step 3. Refer to **Table A-13** for more information on t_{PSL}. QCLK low time is determined by the following equation:

$$\text{QCLK low time (in ns)} = \frac{1000\ (1 + \text{PSL} - 0.5\ \text{PSA})}{f_{sys}\text{(in MHz)}}$$

   where PSL = 0 to 7 and PSA = 0 or 1.

6. Calculate the QCLK frequency (f_{QCLK}).

$$\frac{(f_{QCLK}\text{ (in MHz)} = 1000)}{\text{QCLK high time (in ns)} + \text{QCLK low time (in ns)}}$$

7. Choose the number of input sample cycles (2, 4, 8, or 16) for a typical input channel.
8. If the calculated conversion times are not sufficient, return to step 3. Conversion time is determined by the following equation:

$$\text{Conversion time (in } \mu\text{s)} = \frac{16 + \text{Number of input sample cycles}}{f_{QCLK}\text{(in MHz)}}$$

9. Code the selected PSH, PSL, and PSA values into the prescaler fields of QACR0.

Figure 8-9 and **Table 8-4** show examples of QCLK programmability. The examples include conversion times based on the following assumptions:

- f_{sys} = 20.97 MHz.
- Input sample time is as fast as possible (IST[1:0] = %00, 2 QCLK cycles).

Figure 8-9 and **Table 8-4** also show the conversion time calculated for a single conversion in a queue. For other MCU system clock frequencies and other input sample times, the same calculations can be made.

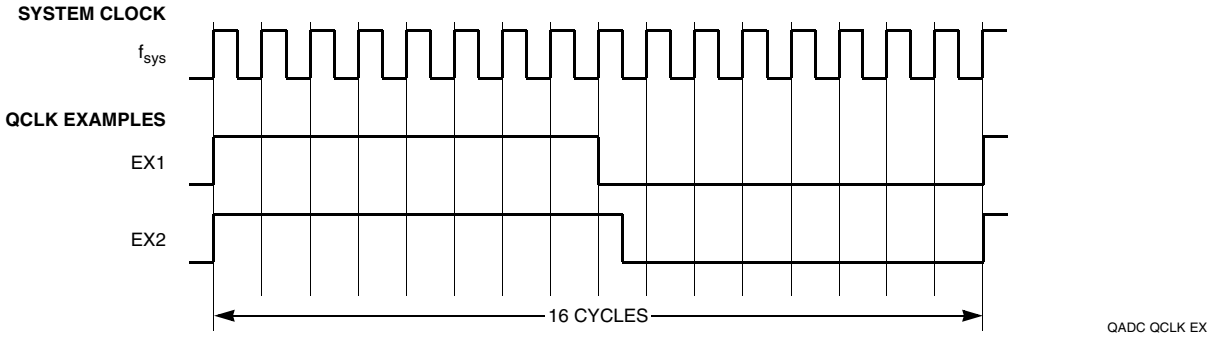


Figure 8-9 QADC Clock Programmability Examples

Table 8-4 QADC Clock Programmability

Control Register 0 Information				f_{sys} = 20.97 Input Sample Time (IST) = %00	
Example Number	PSH[4:0]	PSA	PSL[2:0]	QCLK (MHz)	Conversion Time (µs)
1	7	0	7	1.0	18.0
2	7	1	7	1.0	18.0

The MCU system clock frequency is the basis of QADC timing. The QADC requires that the system clock frequency be at least twice the QCLK frequency. Refer to **Table A-13** for information on the minimum and maximum allowable QCLK frequencies.

Example 1 in **Figure 8-9** shows that when PSH = 3, the QCLK remains high for four system clock cycles. It also shows that when PSL = 3, the QCLK remains low for four system clock cycles.

In order to tune QCLK for the fastest possible conversion time for any given system clock frequency, the QADC permits one more programmable control of the QCLK high and low time. The PSA bit in QACR0 allows the QCLK high phase to be stretched for a half cycle of the system clock, and correspondingly, the QCLK low phase is shortened by a half cycle of the system clock.

Example 2 in **Figure 8-9** is the same as Example 1, except that the PSA bit is set. The QCLK high phase has 4.5 system clock cycles; the QCLK low phase has 3.5 system clock cycles.

8.12.5 Periodic/Interval Timer

The QADC periodic/interval timer can be used to generate trigger events at programmable intervals to initiate scans of queue 2. The periodic/interval timer is held in reset under the following conditions:

- Queue 2 is programmed to any queue operating mode which does not use the periodic/interval timer
- Interval timer single-scan mode is selected, but the single-scan enable bit is cleared to zero
- IMB system reset or the master reset is asserted
- The QADC is placed in low-power stop mode with the STOP bit
- The IMB FREEZE line is asserted and the QADC FRZ bit is set to one

Two other conditions which cause a pulsed reset of the timer are:

- Rollover of the timer counter
- A queue operating mode change from one periodic/interval timer mode to another periodic/interval timer mode

During the low-power stop mode, the periodic/interval timer is held in reset. Since low-power stop mode initializes QACR2 to zero, a valid periodic or interval timer mode must be written to QACR2 when exiting low-power stop mode to release the timer from reset.

If the QADC FRZ bit is set to one and the IMB FREEZE line is asserted while a periodic or interval timer mode is selected, the timer is reset after the current conversion completes. When a periodic or interval timer mode has been enabled (the timer is counting), but a trigger event has not been issued, freeze mode takes effect immediately, and the timer is held in reset. When the IMB FREEZE line is negated, the timer starts counting from zero.

8.12.6 Control and Status Registers

The following paragraphs describe the control and status registers. The QADC has three control registers and one status register. All of the implemented control register fields can be read or written. Reserved locations read zero and writes have no effect. The control registers are typically written once when software initializes the QADC and are not changed afterwards. Refer to **D.5.6 QADC Control Registers** for register and bit descriptions.

8.12.6.1 Control Register 0 (QACR0)

Control register QACR0 establishes the QCLK with prescaler parameter fields and defines whether external multiplexing is enabled.

8.12.6.2 Control Register 1 (QACR1)

Control register QACR1 is the mode control register for queue 1. Applications software defines the operating mode for the queue, and may enable a completion and/or pause interrupt. The SSE1 bit may be written to one or zero but always reads zero.

8.12.6.3 Control Register 2 (QACR2)

Control register QACR2 is the mode control register for queue 2. Applications software defines the operating mode for the queue, and may enable a completion and/or pause interrupt. The SSE2 bit may be written to one or zero but always reads zero.

8.12.6.4 Status Register (QASR)

The status register QASR contains information about the state of each queue and the current A/D conversion. Except for the four flag bits (CF1, PF1, CF2, and PF2) and the two trigger overrun bits (TOR1 and TOR2), all of the status register fields contain read-only data. The four flag bits and the two trigger overrun bits are cleared by writing a zero to the bit after the bit was previously read as a one.

8.12.7 Conversion Command Word Table

The CCW table is a 40-word long, 10-bit wide RAM, which can be programmed to request conversions of one or more analog input channels. The entries in the CCW table are 10-bit conversion command words. The CCW table is written by software and is not modified by the QADC. Each CCW requests the conversion of an analog channel to a digital result. The CCW specifies the analog channel number, the input sample time, and whether the queue is to pause after the current CCW. Refer to **D.5.8 Conversion Command Word Table** for register and bit descriptions.

The ten implemented bits of the CCW word can be read and written. Unimplemented bits are read as zeros, and write operations have no effect. Each location in the CCW table corresponds to a location in the result word table. When a conversion is completed for a CCW entry, the 10-bit result is written in the corresponding result word entry.

The beginning of queue 1 is the first location in the CCW table. The first location of queue 2 is specified by the beginning of queue 2 pointer BQ2 in QACR2. To dedicate the entire CCW table to queue 1, queue 2 is disabled, and BQ2 is programmed to any value greater than 39. To dedicate the entire CCW table to queue 2, queue 1 is disabled, and BQ2 is specified as the first location in the CCW table.

Figure 8-10 illustrates the operation of the queue structure.

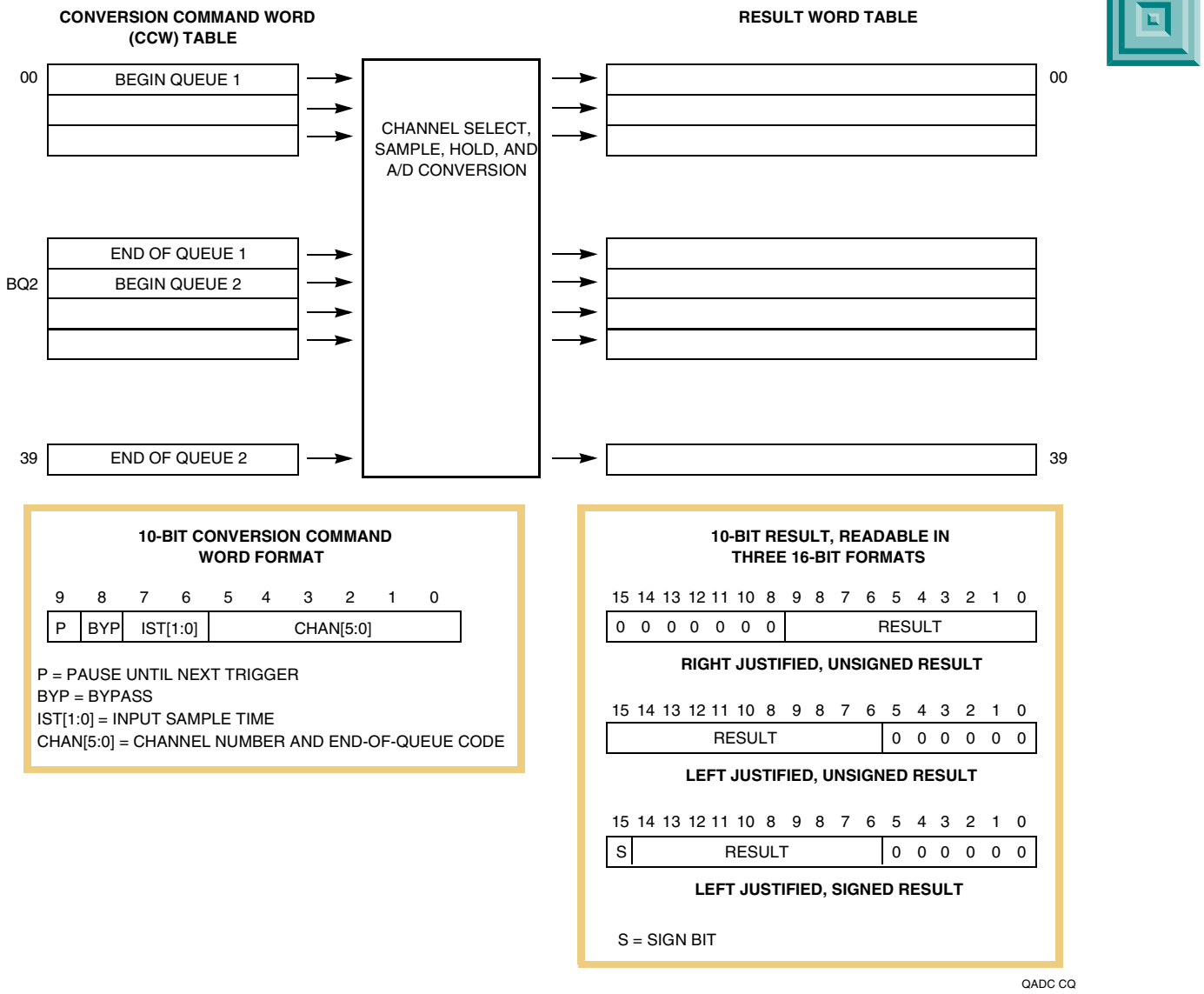


Figure 8-10 QADC Conversion Queue Operation

To prepare the QADC for a scan sequence, the desired channel conversions are written to the CCW table. Software establishes the criteria for initiating the queue execution by programming queue operating mode. The queue operating mode determines what type of trigger event initiates queue execution.

A scan sequence may be initiated by the following trigger events:

- A software command
- Expiration of the periodic/interval timer
- An external trigger signal

Software also specifies whether the QADC is to perform a single pass through the queue or is to scan continuously. When a single-scan mode is selected, queue execu-

tion begins when software sets the single-scan enable bit. When a continuous-scan mode is selected, the queue remains active in the selected queue operating mode after the QADC completes each queue scan sequence.

During queue execution, the QADC reads each CCW from the active queue and executes conversions in four stages:

1. Initial sample
2. Transfer
3. Final sample
4. Resolution

During initial sample, the selected input channel is connected to the sample capacitor at the input of the sample buffer amplifier.

During the transfer period, the sample capacitor is disconnected from the multiplexer, and the stored voltage is buffered and transferred to the RC DAC array.

During the final sample period, the sample capacitor and amplifier are bypassed, and the multiplexer input charges the RC DAC array directly. Each CCW specifies a final input sample time of 2, 4, 8, or 16 QCLK cycles. When an analog-to-digital conversion is complete, the result is written to the corresponding location in the result word table. The QADC continues to sequentially execute each CCW in the queue until the end of the queue is detected or a pause bit is found in a CCW.

When the pause bit is set in the current CCW, the QADC stops execution of the queue until a new trigger event occurs. The pause status flag bit is set, which may generate an interrupt request to notify software that the queue has reached the pause state. When the next trigger event occurs, the paused state ends, and the QADC continues to execute each CCW in the queue until another pause is encountered or the end of the queue is detected.

An end-of-queue condition is indicated as follows:

- The CCW channel field is programmed with 63 ($3F) to specify the end of the queue.
- The end of queue 1 is implied by the beginning of queue 2, which is specified in the BQ2 field in QACR2.
- The physical end of the queue RAM space defines the end of either queue.

When any of the end-of-queue conditions is recognized, a queue completion flag is set, and if enabled, an interrupt request is generated. The following situations prematurely terminate queue execution:

- Since queue 1 is higher in priority than queue 2, when a trigger event occurs on queue 1 during queue 2 execution, the execution of queue 2 is suspended by aborting execution of the CCW in progress, and queue 1 execution begins. When queue 1 execution is complete, queue 2 conversions restart with the first CCW entry in queue 2 or the first CCW of the queue 2 subqueue being executed when queue 2 was suspended. Alternately, conversions can restart with the aborted queue 2 CCW entry. The resume RES bit in QACR2 allows software to select

where queue 2 begins after suspension. By choosing to re-execute all of the suspended queue 2 and subqueue CCWs, all of the samples are guaranteed to have been taken during the same scan pass. However, a high trigger event rate for queue 1 can prohibit the completion of queue 2. If this occurs, execution of queue 2 may begin with the aborted CCW entry.

- When a queue is disabled, any conversion taking place for that queue is aborted. Putting a queue into disabled mode does not power down the converter.
- When the operating mode of a queue is changed to another valid mode, any conversion taking place for that queue is aborted. The queue operating restarts at the beginning of the queue, once an appropriate trigger event occurs.
- When placed in low-power stop mode, the QADC aborts any conversion in progress.
- When the FRZ bit in the QADCMCR is set and the IMB FREEZE line is asserted, the QADC freezes at the end of the current conversion. When FREEZE is negated, the QADC resumes queue execution beginning with the next CCW entry.

8.12.8 Result Word Table

The result word table is a 40-word long, 10-bit wide RAM. The QADC writes a result word after completing an analog conversion specified by the corresponding CCW. The result word table can be read or written, but in normal operation, software reads the result word table to obtain analog conversions from the QADC. Unimplemented bits are read as zeros, and write operations have no effect. Refer to **D.5.9 Result Word Table** for register descriptions.

While there is only one result word table, the data can be accessed in three different alignment formats:

1. Right justified, with zeros in the higher order unused bits.
2. Left justified, with the most significant bit inverted to form a sign bit, and zeros in the unused lower order bits.
3. Left justified, with zeros in the unused lower order bits.

The left justified, signed format corresponds to a half-scale, offset binary, two's complement data format. The data is routed onto the IMB according to the selected format. The address used to access the table determines the data alignment format. All write operations to the result word table are right justified.

8.13 Interrupts

The QADC supports both polled and interrupt driven operation. Status bits in QASR reflect the operating condition of each queue and can optionally generate interrupts when enabled by the appropriate bits in QACR1 and/or QACR2.

8.13.1 Interrupt Sources

The QADC has four interrupt service sources, each of which is separately enabled. Each time the result is written for the last CCW in a queue, the completion flag for the corresponding queue is set, and when enabled, an interrupt request is generated. In

the same way, each time the result is written for a CCW with the pause bit set, the queue pause flag is set, and when enabled, an interrupt request is generated.

Table 8-5 displays the status flag and interrupt enable bits which correspond to queue 1 and queue 2 activity.

Table 8-5 QADC Status Flags and Interrupt Sources

Queue	Queue Activity	Status Flag	Interrupt Enable Bit
Queue 1	Result written for the last CCW in queue 1	CF1	CIE1
	Result written for a CCW with pause bit set in queue 1	PF1	PIE1
Queue 2	Result written for the last CCW in queue 2	CF2	CIE2
	Result written for a CCW with pause bit set in queue 2	PF2	PIE2

Both polled and interrupt-driven QADC operations require that status flags must be cleared after an event occurs. Flags are cleared by first reading QASR with the appropriate flag bits set to one, then writing zeros to the flags that are to be cleared. A flag can be cleared only if the flag was a logic one at the time the register was read by the CPU. If a new event occurs between the time that the register is read and the time that it is written, the associated flag is not cleared.

8.13.2 Interrupt Register

The QADC interrupt register QADCINT specifies the priority level of QADC interrupt requests and the upper six bits of the vector number provided during an interrupt acknowledge cycle.

The values contained in the IRLQ1 and IRLQ2 fields in QADCINT determine the priority of QADC interrupt service requests. A value of %000 in either field disables the interrupts associated with that field. The interrupt levels for queue 1 and queue 2 may be different.

The IVB[7:2] bits specify the upper six bits of each QADC interrupt vector number. IVB[1:0] have fixed assignments for each of the four QADC interrupt sources. Refer to **8.13.3 Interrupt Vectors** for more information.

8.13.3 Interrupt Vectors

When the QADC is the only module with an interrupt request pending at the level being acknowledged, or when the QADC IARB value is higher than that of other modules with requests pending at the acknowledged IRQ level, the QADC responds to the interrupt acknowledge cycle with an 8-bit interrupt vector number. The CPU32 uses the vector number to calculate a displacement into the exception vector table, then uses the vector at that location to jump to an interrupt service routine.

The interrupt vector base field IVB[7:2] specifies the six high-order bits of the 8-bit interrupt vector number, and the QADC provides two low-order bits which correspond to one of the four QADC interrupt sources.

Figure 8-11 shows the format of the interrupt vector, and lists the binary coding of the two low-order bits for the four QADC interrupt sources.

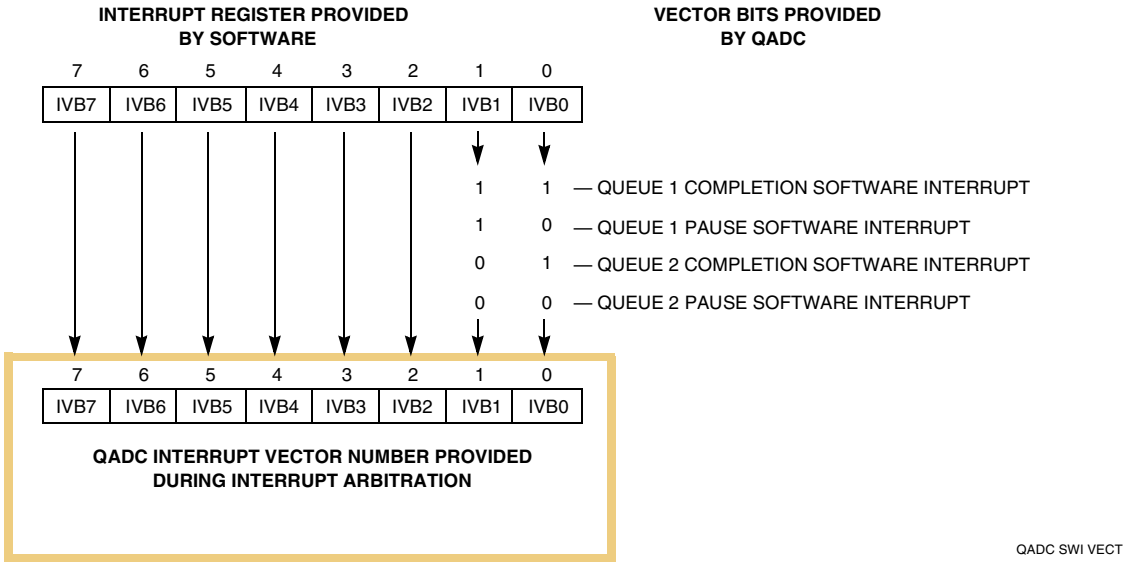


Figure 8-11 QADC Interrupt Vector Format

The IVB field has a reset value of $0F, which corresponds to the uninitialized interrupt exception vector.

8.13.4 Initializing the QADC for Interrupt Driven Operation

The following steps are required to ensure proper operation of QADC interrupts:

1. Assign the QADC a unique non-zero IARB value. The IARB field is located in QADCMCR. The lowest priority IARB value is %0001, and the highest priority IARB value is %1111.
2. Set the interrupt request levels for queue 1 and queue 2 in the IRLQ1 and IRLQ2 fields in QADCINT. Level %001 is the lowest priority interrupt request, and level %111 is the highest priority request.
3. Set the six high-order bits of the eight-bit IVB field in QADCINT. The QADC provides the two low-order vector bits to identify one of four QADC interrupt requests. The vector number for each QADC interrupt source corresponds to a specific vector in the exception vector table. Each vector in the exception vector table points to the beginning address of an exception handler routine.