a

Single-Chip, DSP-Based High Performance Motor Controller

PRELIMINARY TECHNICAL INFORMATION **ADMC401**

FEATURES

26 MIPS Fixed Point DSP Core

- Single Cycle Instruction Execution (38.5 ns).
- External 14-bit Address and 24-bit Data Bus.
- ADSP-21xx Family Code Compatible.
- 16-bit Arithmetic & Logic Unit (ALU).
- Hardware Multiply & Accumulate Unit (MAC).
  - Single Cycle 16-bit x 16-bit Multiply & Accumulate into 40-bit Accumulator.
- 32-bit Shifter (Logical & Arithmetic).
- Multifunction Instructions.
- Single Cycle Context Switch.
- Powerful Program Sequencer:
  - Zero Overhead Looping.
  - Conditional Instruction Execution.
- Two Independent Data Address Generators.

Memory Configuration

- 2k x 24-bit Internal Program Memory RAM.
- 2k x 24-bit Internal Program Memory ROM.
- 1k x 16-bit Internal Data Memory RAM.
- Address & Data Buses permit External Memory Expansion.

High-Resolution Multichannel ADC System

- 12-bit Analog to Digital Converter.
- Pipeline Flash Architecture.
- 8 Dedicated Analog Inputs.
  - All 8 Channels Converted in <2 µs.
  - Channels Sampled at 120 ns Intervals.
  - 8 Dedicated, Memory-Mapped, 16-bit, 2's Complement Data Registers.
  - 4.0 V peak-peak Input Voltage Range.
- Ability to Synchronize Conversions to PWM.
- Internal or External Convert Start.
- Differential Non Linearity < 1 LSB.
- Integral Non Linearity < 2 LSB.
- Dedicated ADC Interrupt on end of Conversion.
- Operation from Internal or External Reference.

Voltage Reference

- Internal 2.5 V ± 1.0 % Voltage Reference.
- Dedicated VREF pin.

Three-Phase PWM Generation Subsystem

- 16-bit Dedicated PWM Generator.
- Edge Resolution to 38.5 ns.
- Programmable Dead Time.
- Programmable Minimum Pulse Width.
- Double Update Mode Allows Duty Cycle & Period Adjustment on Half Cycle Boundaries.
- Special Features for Brushless DC & Switched Reluctance Motors.
- Hardware Polarity Control.
- External Dedicated Asynchronous Shutdown Pin ($\overline{\text{PWMTRIP}}$).
- Additional Shutdown Pins in I/O System.
- Individual Enable/Disable of Each Output.
- High-Frequency Chopping Mode.
- Transparent Transition to Over-Modulation Range with Duty Cycles of 100%.
- Capability to Drive Optocouplers Directly (10 mA Sink & Source Capability).

Flexible Encoder Interface Subsystem

- Incremental Encoder Interface.
- Dedicated Interface (Quadrature and Index Signals).
- Alternative Frequency and Direction Inputs.
- Programmable Filtering of Encoder Input Signals.
- Glitch Detection.
- 16-bit Quadrature Counter.
- Input Encoder Signals to 3.25 MHz.
- Programmable MAXCNT Register.
- Two Inputs Permit Latching of Encoder Counter Value on External Triggers.
- Optional Hardware Reset of Counter.

Single North Marker Mode:
Permits Single Reset of Counter on Only First Zero Marker.
Status Bits to Read Encoder Inputs.
Companion Encoder Event System for Accuracy Enhancements at Low Speeds.
Associated EIU Loop Timer permits Regular, Programmable Updating of All Encoder and Event Timer Registers.
EIU Timer can also be used as General Purpose Timer if not linked to EIU block.

Peripheral I/O (PIO) Subsystem
12 Pin Digital I/O Port.
Bit Configurable as Input or Output.
Each Pin Configurable as Rising Edge, Falling Edge, High Level or Low Level Interrupt.
Hysteresis on all PIO Input Lines.
Four Dedicated PIO Interrupts on PIO0 to PIO3.
One Combined Interrupt for PIO4 to PIO11.
Each I/O Line Configurable as PWM Trip Source.

Two 8-bit Auxiliary PWM Outputs
Fixed 50.7 kHz Switching Frequency.
0 to 99.6 % Duty Cycle.

Event Timer Unit
Two Event Timer Channels (Capture)
Accurate Timing of Duty-Cycle, Period and Frequency.
Flexible Event Definition for Configuration.
Event Timer Readable On-the-Fly

Programmable Interrupt Controller
Manages Priority & Masking of Peripheral Interrupts.

16-Bit Watchdog Timer
Dedicated Clock Source Derived from CLKIN.

Internal Power-On Reset System
Monitors Supply Voltage during Operation.

Programmable 16-Bit Interval Timer with Prescaler

Two Double Buffered Synchronous Serial Ports
Variety of Boot Load Protocols via SPORT1:
Synchronous E^2PROM/SROM Booting.
UART Boot Loader with Autobaud.
Synchronous Master or Slave Boot Loader.
Debugger Interface via SPORT1:
UART Interface with Autobaud.
Synchronous Master or Slave Interface.

ROM Utilities
Full Debugger for Program Development.
Pre-programmed Math Functions:
SINE Function.
COSINE Function.
ARCTAN Function.
SQRT Function.
LOG10 and LN Functions.
RECIPROCAL Function.
Unsigned Single Precision Division.
UMASK - Limit Unsigned Value.
SMASK - Limit Signed Value.
FLTONE - Fixed-point (1.15) to two-word floating-point conversion.
FIXONE - Two-word floating-point to fixed-point (1.15) conversion.
FPA - Two-word floating-point addition.
FPS - Two-word floating-point subtraction.
FPM - Two-word floating-point multiplication.
FPD - Two-word floating-point division.
FPMACC - Floating-point multiply & accumulate.
PUT_VECTOR - Vector table modification.
Pre-programmed Motor Control Functions:
3-Phase to 2-Phase Transformation.
2-Phase to 3-Phase Transformation.
Forward Vector Rotation.
Reverse Vector Rotation.

Industrial Temperature Range -40°C to +85°C

Operating Voltage 5.0 V ± 5 %

Package: 144-Pin TQFP

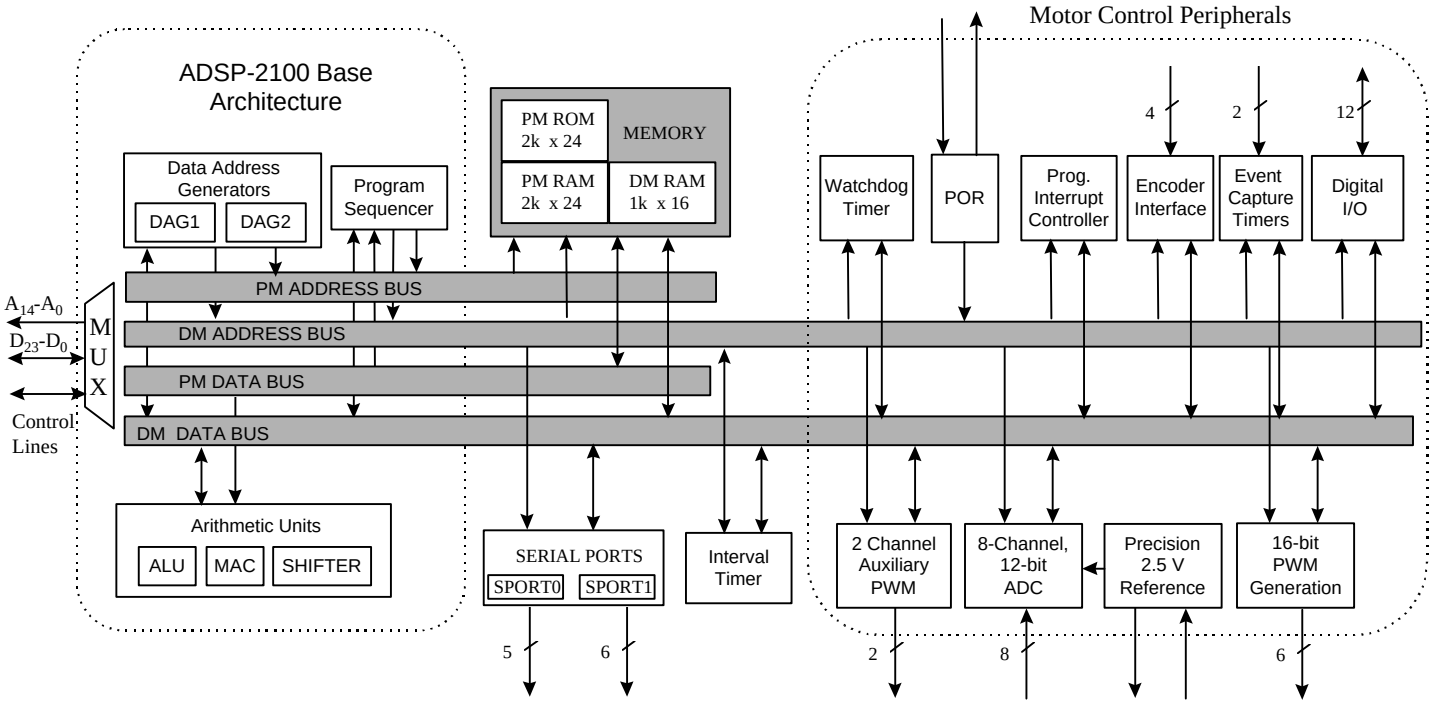


Figure 1: Functional Block Diagram of ADMC401.

ELECTRICAL CHARACTERISTICS (V_{DD} = AV_{DD} = 5 V ± 5 %, GND = AGND = 0 V, T_{AMB} = -40 °C to +85 °C, CLKIN = 13 MHz, unless otherwise specified)

Parameter		Test Conditions	Min	Max	Unit
V_{IH}	HI-Level Input Voltage[1,2]	@ V_{DD} = max	2.0		V
V_{IL}	LO-Level Input Voltage[1,2]	@ V_{DD} = min		0.8	V
V_{T+}	Positive-Going Threshold[3]	@ V_{DD} = min		TBD	V
V_{T-}	Negative-Going Threshold[3]	@ V_{DD} = min	0.8		V
ΔV_T	Hysteresis (V_{T+} - V_{T-})[3]	@ V_{DD} = min	TBD		mV
V_{OH}	HI-Level Output Voltage[1,4,6]	@ V_{DD} = min, I_{OH} = -1.0 mA	2.4		V
		@ V_{DD} = min, I_{OH} = -0.1mA	V_{DD} - 0.3		V
V_{OL}	LO-Level Output Voltage[1,4,6]	@ V_{DD} = min, I_{OL} = 2.0 mA		0.4	V
V_{OH}	HI-Level Output Voltage[5,6]	@ V_{DD} = min, I_{OH} = -10.0 mA	2.4		V
V_{OL}	LO-Level Output Voltage[5,6]	@ V_{DD} = min, I_{OL} = 10.0 mA		0.4	V
I_{IH}	HI-Level Input Current[7]	@ V_{DD} = max, V_{IN} = V_{DD} max		10	µA
I_{IH}	HI-Level Input Current[8]	@ V_{DD} = max, V_{IN} = V_{DD} max		100	µA
I_{IH}	HI-Level Input Current[9]	@ V_{DD} = max, V_{IN} = V_{DD} max		10	µA
I_{IL}	LO-Level Input Current[7]	@ V_{DD} = max, V_{IN} = 0V		10	µA
I_{IL}	LO-Level Input Current[8]	@ V_{DD} = max, V_{IN} = 0V		10	µA
I_{IL}	LO-Level Input Current[9]	@ V_{DD} = max, V_{IN} = 0V		100	µA
I_{OZH}	HI-Level Tristate Leakage Current[10]	@ V_{DD} = max, V_{IN} = V_{DD} max		10	µA
I_{OZL}	LO-Level Tristate Leakage Current[10]	@ V_{DD} = max, V_{IN} = 0V		10	µA
I_{DD}	Digital Supply Current (Idle)[11]	@ V_{DD} = max		TBD	mA
I_{DD}	Digital Supply Current (Dynamic)[12]	@ V_{DD} = max		TBD	mA
I_{DD}	Analog Supply Current (Disabled)[13]	@ V_{DD} = max		TBD	mA
I_{DD}	Analog Supply Current (ADC Only)[14]	@ V_{DD} = max		TBD	mA
I_{DD}	Analog Current (Reference + ADC)[15]	@ V_{DD} = max		TBD	mA

NOTES:

1. Bidirectional pins : D0-D23, RFS0, RFS1, TFS0, TFS1, SCLK0 and SCLK1.
2. Input only pins : $\overline{\text{PWMTRIP}}$, PWMPOL, $\overline{\text{PWMSR}}$, $\overline{\text{RESET}}$, EIA, EIB, EIZP, ETU0, ETU1, DR1A, DR1B, DR0, CLKIN, CONVST, MMAP, BMODE, $\overline{\text{BR}}$ and $\overline{\text{PWD}}$.
3. Programmable Input/Output Pins (PIO0 - PIO11) with Input Hysteresis.
4. Output pins : PWMSYNC, AUX0, AUX1, CLKOUT, DT0, DT1, $\overline{\text{BG}}$, $\overline{\text{BGH}}$, $\overline{\text{PMS}}$, $\overline{\text{DMS}}$, $\overline{\text{BMS}}$, $\overline{\text{RD}}$, $\overline{\text{WR}}$, PWDACK and A0-A13.
5. Output pins : AH, AL, BH, BL, CH and CL.
6. Although specified for TTL outputs, all ADMC401 outputs are CMOS compatible and will drive to V_{DD}-0.3V and GND+0.3V assuming no dc loads.
7. Input only pins $\overline{\text{RESET}}$, EIA, EIB, EIZP, ETU0, ETU1, DR1A, DR1B, DR0, CLKIN, CONVST, MMAP, BMODE, $\overline{\text{BR}}$ and $\overline{\text{PWD}}$.
8. Input pins with internal pull-down PIO0 - PIO11 and $\overline{\text{PWMTRIP}}$.

9. Input pins with internal pull-up, PWMPOL and $\overline{\text{PWMSR}}$.
10. Tri-statable pins : A0-A13, D0-D23, $\overline{\text{PMS}}$, $\overline{\text{DMS}}$, $\overline{\text{BMS}}$, $\overline{\text{RD}}$, $\overline{\text{WR}}$, DT0, DT1, RFS0, RFS1, TFS0, TFS1, SCLK0, SCLK1.
11. Idle refers to the ADMC401 state of operation during execution of the IDLE instruction. Deasserted pins are driven to V_{DD} or GND. Current reflects device operation with CLKOUT disabled.
12. Current reflects device operating with no output loads.
13. Current reflects device operating with both ADC and voltage reference powered down (ADCCTRL(4) = 1, ADCCTRL(3::2) = 0x0).
14. Current reflects device operating with ADC powered on and voltage reference powered down (ADCCTRL bit 4 = 1) and ADCCTRL (3::2) =0x10
15. Current reflects device operating with ADC and voltage reference powered on (ADCCTRL bit 4 = 0) and ADCCTRL (3::2) = 0x10.

ANALOG TO DIGITAL CONVERTER (V_{DD} = AV_{DD} = 5 V ± 5 %, GND = AGND = 0 V, T_{AMB} = -40 °C to +85 °C, CLKIN = 13 MHz, ADC clock = 8.66 MHz, V1 to V8 = 4.0 V_{pp}, V_{ref} = 2.5 V, unless otherwise specified)

Parameter		Test Conditions	Min	Typ	Max	Unit
ENOB	Effective Number of Bits	f_{in} = 1.0 kHz		12		bits
SNR	Signal to Noise Ratio	f_{in} = 1.0 kHz		74		dB
SNRD	Signal to Noise and Distortion	f_{in} = 1.0 kHz		70		dB
THD	Total Harmonic Distortion	f_{in} = 1.0 kHz		-80		dB
CTLK	Channel-Channel Crosstalk	f_{in} = 1.0 kHz on 7 channels, dc on other		-80		dB
CMRR	Common Mode Rejection Ratio			TBD		dB
PSRR	Power Supply Rejection Ratio			TBD		dB
$f_{s,max}$	Sample Frequency				TBD	kHz
t_{conv}	Total Conversion Time (8 Channels)				2.0	µs
t_{skew}	Time Skew Channel to Channel				120	ns
DNL	Differential Non-Linearity				± 0.5	LSB
INL	Integral Non-Linearity				± 1.0	LSB
	Zero Error (@ 25 °C)				TBD	ppm/°C
	Gain Error (@ 25 °C)				TBD	ppm/°C
VIN	Analog Input Voltage Range				4.0	V_{pp}
C_{in}	Input Capacitance			10		pF

VOLTAGE REFERENCE (V_{DD} = AV_{DD} = 5 V ± 5 %, GND = AGND = 0 V, T_{AMB} = -40 °C to +85 °C, CLKIN = 13 MHz, unless otherwise specified)

Parameter		Test Conditions	Min	Typ	Max	Unit
V_{REFOUT}	Internal Voltage Reference		2.475		2.525	V
	REFOUT Source Current				-1.0	mA
V_{REFIN}	REFIN Voltage Reference			2.5		V

R_{IN}	Reference Input Reference		5	kΩ

PULSEWIDTH MODULATOR

(V_{DD} = AV_{DD} = 5 V ± 5 %, GND = AGND = 0 V, T_{AMB} = -40 °C to +85 °C, CLKIN = 13 MHz, unless otherwise specified)

Parameter		Test Conditions	Min	Typ	Max	Unit
	Counter Resolution				16	bits
	Edge Resolution	Double Update Mode		38.5		ns
T_d	Programmable Dead Time		0 78.8			µs
	Dead Time Increments			77		ns
T_{min}	Programmable Minimum Pulsewidth		0		39.3	µs
	Minimum Pulsewidth Increments			38.5		ns
f_{PWM}	PWM Switching Frequency	16-bit Resolution	198			Hz
f_{PWM}	PWM Switching Frequency	8-bit Resolution			101.4	kHz
$T_{PWMSYNC}$	PWMSYNC Pulsewidth		38.5		6114	ns
	PWMSYNC Pulsewidth Increments			38.5		ns
f_{chop}	Gate Drive Chopping Frequency		25.39		6500	kHz

ENCODER INTERFACE

(V_{DD} = AV_{DD} = 5 V ± 5 %, GND = AGND = 0 V, T_{AMB} = -40 °C to +85 °C, CLKIN = 13 MHz, unless otherwise specified)

Parameter		Test Conditions	Min	Typ	Max	Unit
	Encoder Loop Timer Timeout Rate	EIUSCALE = 0x00 EIUPERIOD = 0x0001	38.5			ns
	Encoder Loop Timer Timeout Rate	EIUSCALE = 0xFF EIUPERIOD = 0xFFFF			0.65	s
T_{minenc}	Minimum Encoder Pulsewidth	EIUFILTER = 0x00		115.5		ns
T_{minenc}	Minimum Encoder Pulsewidth	EIUFILTER = 0x3F		7.39		µs
f_{encmax}	Maximum Encoder Rate	EIUFILTER = 0x00		3.25		MHz
f_{encmax}	Maximum Encoder Rate	EIUFILTER = 0x3F		67		kHz

AUXILIARY PWM OUTPUTS

(V_{DD} = AV_{DD} = 5 V ± 5 %, GND = AGND = 0 V, T_{AMB} = -40 °C to +85 °C, CLKIN = 13 MHz, unless otherwise specified)

Parameter		Test Conditions	Min	Typ	Max	Unit
	Resolution				8	bits
f_{aux}	Switching Frequency		50.78		13000	kHz

POWER ON RESET (V_{DD} = AV_{DD} = 5 V ± 5 %, GND = AGND = 0 V, T_{AMB} = -40 °C to +85 °C, CLKIN = 13 MHz, unless otherwise specified)

Parameter	Test Conditions	Min	Typ	Max	Unit
$V_{TH,POR}$ Power On Threshold Voltage		3.8		4.0	V
$V_{TH,RST}$ Reset Threshold Voltage		3.5		3.7	V
t_{RST} Reset Time			5.04		ms

ADMC401 - SPECIFICATIONS

RECOMMENDED OPERATING CONDITIONS

Parameter		B Grade		Unit
		Min	Max	
V_{DD}	Digital Supply Voltage	4.75	5.25	V
AV_{DD}	Analog Supply Voltage	4.75	5.25	V
T_{AMB}	Ambient Operating Temperature	-40	+85	°C

ABSOLUTE MAXIMUM RATINGS*

Supply Voltage..... 0.3 V to + 7V
Input Voltage.....-0.3 V to V_{DD} + 0.3V
Output Voltage Swing .....-0.3 V to V_{DD} + 0.3V
Operating Temperature Range (Ambient)..... -40°C to +85°C
Storage Temperature Range..... -65°C to +150°C
Lead Temperature (5 sec) ..... +280°C

*Stresses above those listed under absolute maximum ratings may cause permanent damage to the device. These are stresses only, and functional operation of the device at these or any other conditions above those indicated in the operational section of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

1. GENERAL DESCRIPTION

The ADMC401 is single-chip DSP-based controller, suitable for high performance control of ac induction motors (ACIM), permanent magnet synchronous motors (PMSM), brushless dc motors (BDCM) and switched reluctance (SR) motors in industrial applications. The ADMC401 integrates a 26 MIPS, fixed-point DSP core with a complete set of motor control peripherals that permits fast motor control in a highly-integrated environment.

The DSP core of the ADMC401 is the ADSP-2171 that is completely code compatible with the ADSP-21xx DSP family and combines three-computational units, data address generators and a program sequencer. The computational units comprise an ALU, a multiplier/accumulator (MAC) and a barrel shifter. The ADSP-2171 adds new instructions for bit manipulation, multiplication (x squared), biased rounding and global interrupt masking. In addition, two flexible double-buffered, bi-directional synchronous serial ports are included in the ADMC401.

The ADMC401 provides 2k x 24-bit program memory RAM, 2k x 24-bit program memory ROM and 1k x 16-bit data memory RAM. The program and data memory RAM can be boot loaded through the serial port from either a serial E^2PROM, through a UART connection (either from external host microprocessor or from the Motion Control Debugger) or via a synchronous serial interface from a host microprocessor. The program memory ROM includes a monitor that adds software debugging features through the serial port. In addition, a number of pre-programmed mathematical and motor control functions are included in the program memory ROM.

Additionally, the ADMC401 device adds significant external memory and peripheral expansion capabilities by making available the full address and data bus of the DSP core. This feature permits expansion of both external program and data memory and means that the DSP core can address up to 8 k x 24-bits of external program memory and up to 13 k x 16-bits of external data memory. In addition both internal and external program and data memory RAM can be boot loaded across the address and data buses from an external EPROM.

The motor control peripherals of the ADMC401 comprise a high-performance 8 channel, 12-bit sampling ADC system. In addition, a three-phase, 16-bit, center-based PWM generation unit can be used to produce high-accuracy PWM signals with minimal processor overhead. The ADMC401 also contains a flexible encoder interface unit for position sensor feedback, two auxiliary PWM outputs, 12 line of digital I/O, a two-channel event capture system, a 16-bit watchdog timer, two 16-bit interval timer (one of which can be linked to the encoder interface unit) and a programmable interrupt controller that manages all peripheral interrupts.

2. PIN LIST

Pin Group Name	# of Pins	Type	Function
A_{13} - A_0	14	O	Address Lines
D_{23} - D_0	24	I/O	Data Lines
$\overline{PMS}$, $\overline{DMS}$, $\overline{BMS}$	3	O	External Memory Select Lines
$\overline{RD}$, $\overline{WR}$	2	O	External Memory Read/Write Enable
MMAP	1	I	Memory Map Select
POR	1	O	Internal Power On Reset Output
$\overline{RESET}$	1	I	Processor Reset Input.
CLKOUT	1	O	Processor clock output
CLKIN, XTAL	2	I	External clock or quartz crystal input
$\overline{BR}$	1	I	Bus Request
$\overline{BG}$	1	O	Bus Grant
$\overline{BGH}$	1	O	Bus Hang Control
BMODE	1	I	Boot Mode Select
$\overline{PWD}$	1	I	Powerdown Pin
PWDACK	1	O	Powerdown Acknowledge
SPORT0	5	I/O	Serial Port 0 pins (TFS0, RFS0, DT0, DR0, SCLK0)
SPORT1	6	I/O	Serial Port 1 (TFS1/ $\overline{IRQ1}$, RFS1/ $\overline{IRQ0}$ / $\overline{SROM}$, DT1/FO, DR1A/FI, DR1B/FI, SCLK1)
VIN1 - VIN8	8	I	Analog Inputs
VINA	1	I	Analog Input to Sample & Hold Amplifier
AMUXOUT	1	O	Analog Output of Multiplexer
VREF	1	I/O	Reference Input Output
REFCOM	1	GND	Reference Common
CML	1	O	Common Mode Level (Mid-Supply)
CAPT,CAPB	2		Noise Reduction Pins
CONVST	1	I	External Convert Start
AH-CL	6	O	PWM outputs
$\overline{PWMTRIP}$	1	I	PWM trip signals
PWMPOL	1	I	PWM polarity pin
PWMSYNC	1	O	PWM synchronization pin
$\overline{PWMSR}$	1	I	PWM Switched Reluctance Mode Control
PIO0-PIO11	12	I/O	Digital I/O Port
ETU0, ETU1	2	I	Event Timer Inputs
AUX0-AUX1	2	O	Auxiliary PWM outputs
EIA,EIB,EIZP, EIS	4	I	Encoder interface pins and external EIU latch triggers
AVDD	2	SUP	Analog power supply
AGND	2	GND	Analog ground
VDD	7	SUP	Digital power supply
GND	19	GND	Digital ground

Table 1: Pin List of ADMC401 Device

3. DSP CORE ARCHITECTURE

3.1 DSP CORE FEATURES & OVERVIEW

The ADSP-2171 flexible architecture and comprehensive instruction set allow the processor to perform multiple operations in parallel. In one processor cycle (38.5 ns with a 13 MHz crystal) the DSP core can:

- generate the next program address
- fetch the next instruction
- perform one or two data moves
- update one or two data address pointers
- perform a computational operation

This all takes place while the processor continues to:

- receive and transmit through the serial ports
- decrement the interval timer
- generate PWM signals
- convert the ADC input signals
- operate the encoder interface units
- operate all other peripherals including the auxiliary PWM & event timer subsystem

The processor contains three independent computational units: the arithmetic & logic unit (ALU), the multiplier/accumulator (MAC) and the shifter. The computational units process 16-bit data directly and have provisions to support multi-precision computations. The ALU performs a standard set of arithmetic and logic operations; division primitives are also supported. The MAC performs single-cycle multiply, multiply/add, multiply/subtract operations with 40-bits of accumulation. The shifter performs logical and arithmetic shifts, normalization, denormalization, and derive exponent operations. The shifter can be used to implement efficiently numeric format control including floating-point representations. The internal result (R) bus directly connects the computational units so that the output of any unit may be the input of any unit on the next cycle.

A powerful program sequencer and two dedicated data address generators ensure efficient delivery of operands to these computational units. The sequencer supports conditional jumps, subroutine calls and returns in a single cycle. With internal loop counters and loop stacks, the ADMC401 executes looped code with zero overhead; no explicit jump instructions are required to maintain the loop.

Two data address generators (DAGs) provide addresses for simultaneous dual operand fetches from data memory and program memory. Each DAG maintains and updates four address pointers (I registers). Whenever the pointer is used to access data (indirect addressing), it is post-modified by the value in one of four modify (M) registers. A length value may be associated with each pointer (L registers) to implement automatic modulo addressing for circular buffers. The circular buffering feature is also used by the serial ports for automatic data transfers to and from on-chip memory. DAG1 generates only data memory addresses but provides an optional bit-reversal capability. DAG2 may generate either program or data memory addresses, but has no bit-reversal capability.

Efficient data transfer is achieved with the use of five internal buses:

- Program Memory Address (PMA) Bus
- Program Memory Data (PMD) Bus

- Data Memory Address (DMA) Bus
- Data Memory Data (DMD) Bus
- Result (R) Bus

Program memory can store both instructions and data, permitting the ADMC401 to fetch two operands in a single cycle, one from internal program memory and one from internal data memory. The ADMC401 can fetch an operand from on-chip program memory and the next instruction in the same cycle.

The ADMC401 writes data from its 16-bit registers to the 24-bit program memory using the PX register to provide the lower eight bits. When it reads data (not instructions) from 24-bit program memory to a 16-bit data register, the lower eight bits are placed in the PX register.

The ADMC401 can respond to a number of distinct DSP core and peripheral interrupts. The DSP core interrupts include serial port receive and transmit interrupts, timer interrupts, software interrupts and external interrupts. In addition, there is a master RESET signal. The motor control peripherals also produce interrupts to the DSP core.

The two serial ports (SPORTs) provide a complete synchronous serial interface with optional companding in hardware and a wide variety of framed and unframed data transmit and receive modes of operation. Each SPORT can generate an internal programmable serial clock or accept an external serial clock. Boot loading of both the program and data memory RAM of the ADMC401 can be through the serial port SPORT1. Alternatively the ADMC401 can be boot loaded from external parallel memory connected to the external address & date buses of the device.

A programmable interval counter is also included in the DSP core and can be used to generate periodic interrupts. A 16-bit count register (TCOUNT) is decremented every *n* processor cycles, where *n-1* is a scaling value stored in the 8-bit TSCALE register. When the value of the counter reaches zero, an interrupt is generated and the count register is reloaded from a 16-bit period register (TPERIOD).

The ADMC401 instruction set provides flexible data moves and multifunction (one or two data moves with a computation) instructions. Each instruction is executed in a single 38.5 ns processor cycle (for a 13 MHz crystal). The ADMC401 assembly language uses an algebraic syntax for ease of coding and readability. A comprehensive set of development tools support program development.

3.2 SERIAL PORTS

The ADMC401 incorporates two complete synchronous serial ports (SPORT0 & SPORT1) for serial communications and multiprocessor communication. The following is a brief list of the capabilities of the ADMC401 SPORTs. Refer to the *ADSP-2100 Family User's Manual* for further details.

- SPORTs are bidirectional and have a separate, double-buffered transmit and receive section.
- SPORTs can use an external serial clock or generate their own serial clock internally.
- SPORTs have independent framing for the receive and transmit sections. Sections run in a frameless mode or with frame synchronization signals internally or externally generated. Frame synchronization signals are active high or inverted, with either of two pulse widths and timings.
- SPORTs support serial data word lengths from 3 to 16 bits and provide optional A-law and μ-law companding according to CCITT recommendation G.711.
- SPORT receive and transmit sections can generate unique interrupts on completing a data word transfer.
- SPORTs can receive and transmit an entire circular buffer of data with only one overhead cycle per data word. An interrupt is generated after a data buffer transfer.

- SPORT0 has a multichannel interface to selectively receive and transmit a 24 or 32 word, time-division multiplexed, serial bitstream.
- SPORT1 can be configured to have two external interrupts ($\overline{IRQ0}$ and $\overline{IRQ1}$), and the Flag In and Flag Out signals. The internally generated serial clock may still be used in this configuration.
- SPORT1 is the default input for program and data memory boot loading. The RFS1 pin can be configured internally to the ADMC401 as an SROM/E^2PROM reset signal.
- SPORT1 has two data receive pins (DR1A and DR1B). The DR1A pin is intended only for synchronous data receive from the external E^2PROM. The DR1B pin can be used as the data receive pin for a general purpose SPORT after booting or as the data receive pin for other boot load modes or as the UART/debugger interface. The DR1A and DR1B pins are internally multiplexed onto the one data receive pin of the SPORT. The particular data receive pin selected is determined by bit 4 of the MODECTRL register.

3.3 INTERRUPT OVERVIEW

The ADMC401 can respond to different interrupt sources, some of which are internal DSP core interrupts and others from the motor control peripherals. The DSP core interrupts include a:

- power up (or $\overline{RESET}$) interrupt
- a peripheral (or $\overline{IRQ2}$) interrupt
- a SPORT0 receive and a SPORT0 transmit interrupt
- a SPORT1 receive (or $\overline{IRQ0}$) and a SPORT1 transmit (or $\overline{IRQ1}$) interrupt
- two software interrupts
- an interval timer time out interrupt
- a powerdown interrupt

In addition, the motor control peripherals add other interrupts that include:

- a PWMSYNC interrupt
- an ADC end of conversion interrupt
- an encoder loop timer time out interrupt
- five peripheral input/output (PIO) interrupts
- an event timer interrupt
- an encoder count error interrupt
- a PWM trip interrupt

The interrupts are internally prioritized and individually maskable. All peripheral interrupts are multiplexed into the DSP core through the peripheral ($\overline{IRQ2}$) interrupt. The programmable interrupt controller (PIC) manages the prioritizing, masking and vector addressing of all peripheral interrupts.

3.4 MEMORY MAP

The ADMC401 has two distinct memory types; program memory and data memory (in addition to external boot memory). In general program memory contains user code and coefficients, while the data memory is used to store variables and data during program execution. Both program memory RAM and ROM is provided internally on the ADMC401. Internal program memory RAM is arranged as a 2k x 24-bit block starting at address 0x0000. A 2k x 24-bit block of internal

program memory ROM is located at address 0x0800. The 4k block of program memory address space starting at address 0x1000 is reserved. However, every effort will be made to make as much of this space as possible available as external program memory address space. Internal data memory is arranged as a 1k x 16-bit block starting at address 0x3800. The motor control peripherals are memory mapped into a region of the data memory space starting at 0x2000. The complete program and data memory maps are given in Table 2 and Table 3 respectively.

	Address Range	Function
Internal Program Memory RAM (2k)	0x0000-0x005F	Vector table
	0x0060-0x075F	User program space
	0x0760-0x07DF	Reserved by debugger
	0x07E0-0x07FF	Reserved by monitor
Internal Program Memory ROM (2k)	0x0800-0x0DF6	ROM monitor
	0x0DF7-0x0F4C	ROM utilities
	0x0F4D-0x0FFF	Reserved
Reserved Memory Space (4k)	0x1000 - 0x1FFF	Reserved (but some may become external program memory space)
External Program Memory (8k)	0x2000 - 0x3FFF	User Program Space (external)

Table 2: Program Memory Map of ADMC401.

	Address Range	Function
External Data Memory (8k)	0x0000-0x1FFF	External User Data Space
Memory Mapped Registers (1k)	0x2000-0x23FF	Reserved for Memory Mapped Peripheral Registers
External Data Memory (5k)	0x2400-0x3800	External User Data Space
Internal Data Memory RAM (1k)	0x3800-0x3B5F	Internal User Data Space
	0x3B60-0x3BFF	Reserved by Monitor
Reserved (1k)	0x3C00-0x3FFF	DSP Memory Mapped Registers & Reserved.

Table 3: Data Memory Map of ADMC401.

3.5 ROM CODE

The 2k x 24-bit block of program memory ROM starting at address 0x800 contains a monitor function that can be used to download and execute user programs via the serial port. In addition, the monitor function supports an interactive mode in which commands are received and processed from a host that is configured as a UART device. An example of such a host is the Windows-based *Motion Control Debugger* that is part of the software development system for the ADMC401. In the interactive mode, the host can access both the internal DSP & peripheral motor control registers of the ADMC401, read & write to both program & data memory, implement breakpoints and perform single-step operation as part of the program debugging cycle. In addition to the monitor function, the program memory ROM contains a number of useful mathematical

and motor control utilities that can be called as subroutines from the user code. A list of the functions and the associated program memory address in the ROM code is given in Table 4.

PM Address	Routine	Description
0x07F1	PER_RST	Peripheral Reset
0x0DED	UMASK	Limits unsigned value to given range
0x0DF4	PUT_VECTOR	Facilitates user setup of vector table
0x0E06	SMASK	Limits signed value to given range
0x0E26	ADMC_COS	Cosine function
0x0E2D	ADMC_SIN	Sine function
0x0E43	ARCTAN	Arctangent function
0x0E65	RECIPROCAL	Reciprocal (1/x) function
0x0E7B	SQRT	Square root function
0x0EB5	LN	Natural logarithm function
0x0EB8	LOG	Logarithm (base 10) function
0x0ED4	FLTONE	Fixed point to floating point conversion
0x0ED9	FIXONE	Floating point to fixed point conversion
0x0EDD	FPA	Floating point addition
0x0EEC	FPS	Floating point subtraction
0x0EFC	FPM	Floating point multiplication
0x0F05	FPD	Floating point division
0x0F26	FPMACC	Floating point multiply & accumulate
0x0F48	PARK	Forward & reverse Park transformation (vector rotation)
0x0F5C	REV_CLARK	Reverse Clark transformation (2->3 phase transformation)
0x0F72	FOR_CLARK	Forward Clark transformation (3->2 phase transformation)
0x0F80	COS64	64 point cosine table
0x0FC0	ONE_BY_X	16 point 1/x table
0x0FD0	SDIVQINT	Unsigned single precision division (Integer)
0x0FD9	SDIVQ	Unsigned single precision division (fractional)

Table 4: ROM Function descriptions

4. SYSTEM INTERFACE

4.1 CLOCK SIGNALS

The ADMC401 can be clocked by either a crystal or by a TTL-compatible clock signal. The CLKIN input cannot be halted, changed during operation or operated below the specified minimum frequency during normal operation. If an external clock is used, it should be a TTL-compatible signal running at half the instruction rate. The signal is connected to the CLKIN pin of the ADMC401. In this mode, with an external clock signal, the XTAL pin *must* be left unconnected. The ADMC401 uses an input clock with a frequency equal to half the instruction rate; a 13 MHz input clock yields a 38.5 ns processor cycle (which is equivalent to 26 MHz). Normally instructions are executed in a single processor cycle. All device timing is relative to the internal instruction rate, which is indicated by the CLKOUT signal when enabled.

Because the ADMC401 includes an on-chip oscillator circuit, an external crystal may be used instead of a clock source. The crystal should be connected across the CLKIN and XTAL pins. A parallel-resonant, fundamental frequency,

microprocessor-grade crystal should be used. A clock output signal (CLKOUT) is generated by the processor at the processor's cycle rate of twice the input frequency. The timing of the CLKIN and CLKOUT signals for the ADMC401 are shown in Figure 2. The period of the CLKIN signal is denoted by t_{CKI} and is equal to 76.9 ns for a 13 MHz CLKIN. The internal delay from rising edge of CLKIN to rising edge of CLKOUT is t_{CKOH}. The DSP instruction rate is t_{CK} (the period of the CLKOUT signal) which is equal to 38.5 ns for a 13 MHz CLKIN (corresponding to a 26 MIPS device). In addition, t_{CK} is the fundamental time increment of the motor control peripherals. Therefore, unless otherwise specified, the motor control peripherals are clocked at a rate equal to CLKOUT.

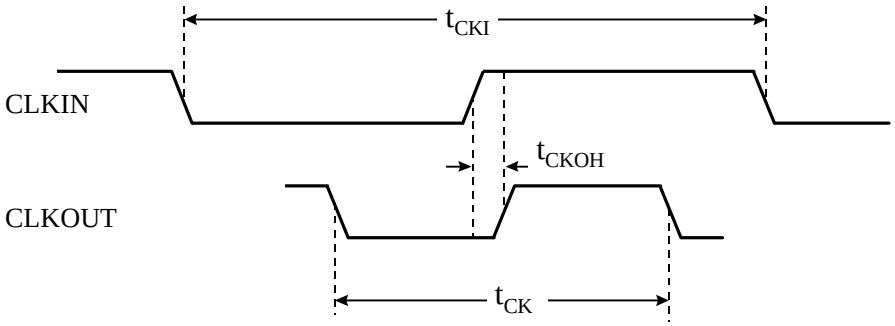


Figure 2: Clock Timing Signals for ADMC401.

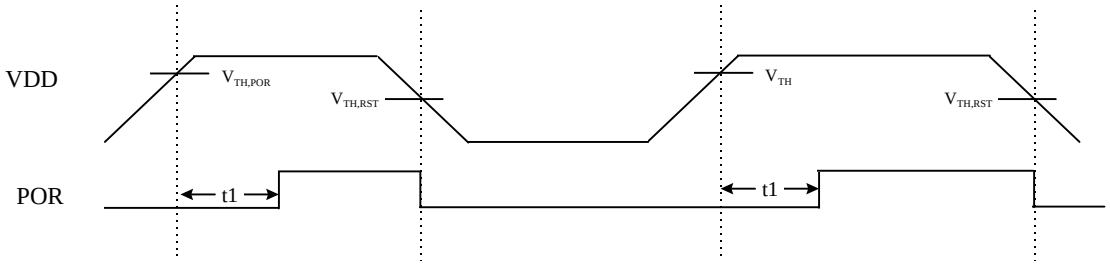
4.2 RESET AND POWER ON RESET (POR) CIRCUIT

The $\overline{\text{RESET}}$ pin initiates a complete hardware reset of the ADMC401 when pulled low. The $\overline{\text{RESET}}$ signal must be asserted when the device is powered up to assure proper initialization. The ADMC401 contains an integrated power-on reset (POR) circuit that provides an output reset signal, POR, from the ADMC401 on power up and if the power supply voltage falls below the threshold level. The ADMC401 may be reset from an external source using the $\overline{\text{RESET}}$ signal or alternatively the internal power on reset circuit may be used by connecting the POR pin to the $\overline{\text{RESET}}$ pin. During power up the $\overline{\text{RESET}}$ line must be activated for long enough to allow the DSP core's internal clock to stabilize. The power-up sequence is defined as the total time required for the crystal oscillator to stabilize after a valid V_{DD} is applied to the processor and for the internal phase locked loop (PLL) to lock onto the specific crystal frequency. A minimum of 2000 t_{CKI} cycles will ensure that the PLL has locked (this does not include the crystal oscillator start-up time).

The operation of the internal power on reset circuit is illustrated in Figure 3. On power up, the circuit maintains the POR pin low until it detects that the V_{DD} line has attained the threshold voltage, $V_{TH,POR}$, level. The V_{TH} value is nominally set at 3.9 V. As soon as the threshold voltage is attained the power on reset circuit enables a 17-bit counter that is clocked at the CLKOUT rate. While the counter is counting the POR pin is held low. When the counter overflows, after a time:

$$t_1 = 2^{17} \times t_{CK} = 2^{17} \times 38.5 \times 10^{-9} = 5.04\,\text{ms}$$

the POR pin is brought high and if the POR and $\overline{\text{RESET}}$ pins are connected, the device is brought out of reset.



$t1 = 2^{17} * t_{CK} = 5.04$ ms at 26 MHz CLKOUT

Nominal values: $V_{DD} = 5V$
$V_{TH,POR} = 3.9V$
$V_{TH,RST} = 3.6V$

Figure 3: Operation of power on reset (POR) circuit of ADMC401.

The internal power on reset circuit also acts as a power supply monitor and puts the POR pin at a LO level if it detects a voltage less than $V_{TH,RST}$ (nominally 3.6 V). The supply voltage must then exceed $V_{TH,POR}$ to initiate another power on reset sequence. The master reset ($\overline{\text{RESET}}$ = LO) sets all internal stack pointers to the empty stack condition, masks all interrupts, clears the MSTAT register and performs a full reset of all of the motor control peripherals. Following a power-up, it is possible to initiate a DSP core and motor control peripheral reset by simply pulling the $\overline{\text{RESET}}$ low. For these resets, there is no need to wait for PLL stabilization and the $\overline{\text{RESET}}$ signal must meet the minimum pulse width specification, t_{RSP}. To generate the external $\overline{\text{RESET}}$ signal, it is recommended to use either an RC circuit with a Schmitt trigger or a commercially available reset IC.

A full system reset of the ADMC401 resets the DSP core and all peripherals excluding the watchdog timer. A software controlled full peripheral reset (excluding the watchdog timer) is achieved by toggling the DSP FL2 flag with the following code segment:

```
PRESET:    SET FL2;
           TOGGLE FL2;
           TOGGLE FL2;
           RTS;
```

This code segment is available as a ROM function (PER_RST). A full DSP and peripheral reset will occur automatically on a watchdog trip.

4.3 EXTERNAL MEMORY INTERFACE

The ADMC401 can address 8k x 24-bits of external program memory (with possible expansion to 10k) and up to 13k x 16-bits of external data memory. The ADMC401 provides the address on a 14-bit address bus (A_{13} - A_0). Instructions or data are transferred across the 24-bit data bus (D_{23} - D_0) during program memory accesses. During data memory accesses, data is transferred on the 16 most significant bits (D_{23} - D_8) of the data bus. For a dual off-chip fetch, the data from program memory is read first, then the data memory data. The program memory select pin, $\overline{\text{PMS}}$, is activated during external program memory accesses and can be used as a chip select signal for the external memory devices. Similarly, for external data memory accesses, the $\overline{\text{DMS}}$ pin is activated.

Two control lines indicate the direction of the transfer. Memory read, $\overline{RD}$, is active low signaling a read from external memory and memory write, $\overline{WR}$, is active low signaling a write to external memory. Typically, the $\overline{PMS}$ line is connected to the $\overline{CE}$ (chip enable) of the external program memory and the $\overline{DMS}$ line is connected to the $\overline{CE}$ line of the external data memory. The $\overline{RD}$ line is connected to the $\overline{OE}$ (output enable) pin and the $\overline{WR}$ is connected to the $\overline{WE}$ (write enable) of both memories. On chip accesses (to internal program memory RAM and ROM) do not drive any of the external signals. The $\overline{PMS}$, $\overline{RD}$ and the $\overline{WR}$ lines remain high (deasserted) and the address and data buses are tri-stated. Similarly, internal accesses to data memory (including internal DM RAM and peripheral and DSP core memory mapped registers) also do not drive external signals and the $\overline{DMS}$, $\overline{RD}$ and the $\overline{WR}$ lines remain high (deasserted) and the address and data buses are also tri-stated.

The program memory interface can generate between 0 and 7 wait states for external memory devices. The program memory wait state field (PWAIT) in the System Control Register (bits 0 - 2) sets the number of wait states that are inserted. The wait states for external data memory accesses is defined by the Data Memory Wait State Control Register. All internal program memory, data memory and peripheral register accesses are zero wait state.

External peripherals can also be connected externally and memory mapped to the external memory space of the ADMC401. The 16 MSBs of the external data bus are connected internally to the 16 LSBs of the DMD. Therefore, the data lines D_{23} - D_8 should be used for 16-bit peripherals. Timing for external program and memory read and write operations are identical to those specified for the ADSP-2171 operating at 26 MHz CLKOUT. Refer to the datasheet for the ADSP-2171 for further details.

4.4 BOOT LOADING

4.4.1 Standalone Mode (MMAP=BMODE=1)

Boot loading of the ADMC401 may occur in a number of different ways and is determined by the state of both the MMAP and BMODE pins. If both MMAP and BMODE are tied to V_{DD} (HI) then the ADMC401 is placed in the so-called *standalone mode* and execution starts from internal program memory ROM at address 0x0800 following a power on or reset. This starts execution of the internal monitor function that first performs some initialization functions and copies a default interrupt vector table to addresses 0x0000 - 0x005B of program memory RAM. The monitor program next clears bit 4 of the MODECTRL register to connect the DR1A pin to the internal data receive port (DR1) of SPORT1. In addition, bit 5 of the MODECTRL register is set. This connects the FL1 port of the DSP core to the RFS1 / $\overline{SROM}$ pin to act as a reset for a serial memory device.

The monitor next attempts to boot load from an external SROM or E^{2}PROM on SPORT1 using the three wire connection of Figure 4. The monitor program first toggles the RFS1 / $\overline{SROM}$ pin of the ADMC401 to reset the serial memory device with the following code segment:

```
SROMRESET: SET FL1;
           TOGGLE FL1;
           TOGGLE FL1;
           RTS;
```

If an SROM or E^{2}PROM is connected to SPORT1, data is clocked synchronously into the ADMC401 at a rate of 1 Mb/s. Both internal and external program and data memory RAM can be loaded from the SROM/E^{2}PROM up to the available capacity of the serial memory device. After the complete boot load is complete, program execution begins at address 0x0060. This is where the first instruction of the user code should be placed.

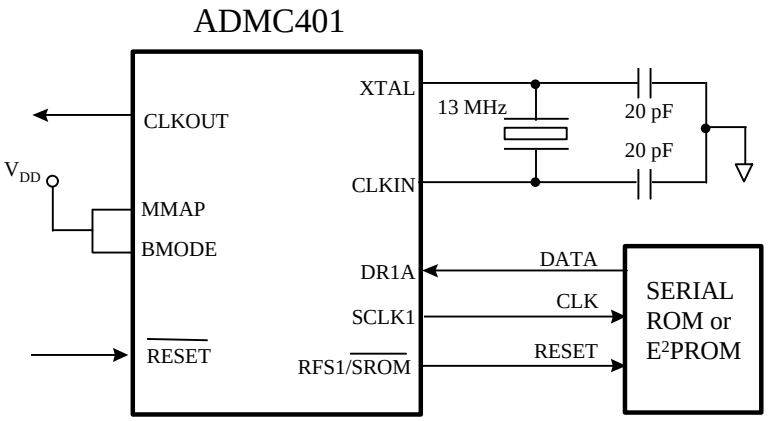


Figure 4: ADMC401 Basic System Configuration in Standalone Mode.

If boot loading from an E²PROM is unsuccessful, the monitor code reconfigures SPORT1 as a UART (setting both bits 4 and 5 of the MODECTRL register) and attempts to receive commands from an external device on this serial port using the DR1B pin. The monitor now waits for two bytes of information. These bytes are received asynchronously so that no clock is needed. The first bytes is the autobaud byte and it is used to calculate the baud rate at which data is being received. This is known as the autobaud feature. The ADMC401 will automatically lock onto the baud rate of the external device if it is sent a byte of 0x70. The maximum baud rate that the ADMC401 will lock onto is 300 kb/s for a 26 MHz CLKOUT.

The second byte of information received is the header byte that uniquely identifies to the monitor which type of interface it is connected to. There are six different interfaces supported on the ADMC401. These includes:

- A UART boot loader such as from a Motorola 68HC11 communicating over its Serial Communications Interface (SCI) port. Following the boot load, execution begins at address 0x060 of internal program memory RAM.
- A synchronous slave boot loader (the clock is external).
- A synchronous master boot loader (the ADMC401 provides the clock).
- A UART debugger interface such as the *Motion Control Debugger* from Analog Devices. The monitor then processes commands received from the debugger over the UART interface.
- A synchronous master debugger interface.
- A synchronous slave debugger interface.

4.4.2 EPROM Boot Mode (MMAP=BMODE=0)

If both the MMAP and BMODE pins are tied to GND, the ADMC401 operates in the so-called *EPROM Boot Mode*. In this mode the entire internal program memory, or any portion of it, can be loaded from an external source using a boot sequence over the memory interface. To allow boot loading from inexpensive EPROM devices, the processor loads data one byte at a time. The boot sequence can also be initiated after reset by software. Boot memory is organized into eight pages, each of which is 8k bytes long. Every fourth byte of a page is an *empty* byte except for the first one, which contains the page length. Each set of three bytes between successive empty bytes contains one 24-bit instruction to be loaded into the internal PM RAM of the DSP. The page length is read first and then bytes are loaded from the top of the page downwards. This causes shorter booting times for shorter pages. The length of the boot page is given as:

page length = (number of 24-bit PM words/8) - 1

That is a page length of 0 causes the boot address generator to generate byte addresses for 8 words which reside in 32 sequential EPROM locations.

Following a reset, if both MMAP and BMODE are LO, the boot sequence always boot loads page 0. After reset, boot loading can occur under program control from any one of up to 8 different boot pages. The boot page select field (BPAGE) in the memory mapped System Control Register specifies which boot page is to be loaded. To boot from a specific boot page, first set the BPAGE bits to the desired value and set the boot force bit (BFORCE) of the System Control Register to initiate a boot sequence.

The ADMC401 can boot its internal program memory from a single byte-wide CMOS EPROM such as the 27C64 or the 27C512. A low-cost commodity-grade EPROM with an industry-standard access time can be used. The number of wait states for the boot memory access is selected in the BWAIT field of the System Control Register. This field can be set to any value from 0 to 7 to set the number of wait states. The default value for the BWAIT field is 7 so that 7 wait states are inserted into the boot loading sequence.

Timing of the boot memory access is identical to that of external program memory or external data memory accesses, except that the active strobe is $\overline{\text{BMS}}$ rather than $\overline{\text{PMS}}$ or $\overline{\text{DMS}}$. To address eight pages of 8k bytes each, 16 address lines are needed. The least significant 14 bits are output on the 14-bit address bus (A_{13} to A_0) while the most significant two bits are output on the 2 MSBs of the data bus (D_{23} and D_{22}) during boot memory accesses. The data is read from the middle eight bits of the data bus (D_{15} to D_8).

4.4.3 External Memory Mode (BMODE=0, MMAP=1)

In this mode, with BMODE tied to GND and MMAP tied to V_{DD}, the ADMC401 is placed in external memory mode and there is no boot loading. The effect of this mode is that the internal 2k bank of program memory RAM is relocated from the bottom of memory (starting at address 0x0000) to the top of the program memory space (at address 0x3800). In this mode, program execution starts at external memory at address 0x0000.

4.4.4 HIP Booting (BMODE=1, MMAP=0)

This boot mode is not available on the ADMC401. It corresponds to boot loading over the host interface port (HIP) of the ADSP-2171 DSP core. Since the HIP is not available on the ADMC401, this combination of the BMODE and MMAP pins is illegal and the operation of the core is undefined in this state.

4.5 BUS REQUEST/GRANT

The ADMC401 can relinquish control of its data and address buses to an external device. The external device requests the bus by asserting (low) the bus request signal $\overline{\text{BR}}$. $\overline{\text{BR}}$ is an asynchronous input and if the ADMC401 is not performing an external access, it responds to the active $\overline{\text{BR}}$ input in the following processor cycle by:

- tri-stating the data and address buses and the $\overline{\text{PMS}}$, $\overline{\text{DMS}}$, $\overline{\text{BMS}}$, $\overline{\text{RD}}$ and $\overline{\text{WR}}$ output drivers
- asserting the bus grant $\overline{\text{BG}}$ signal, and
- halting program execution (unless Go Mode is enabled)

If Go Mode is enabled, the ADMC401 continues to execute instructions from its internal memory. It will not halt program execution until it encounters an instruction that requires an external access. If Go Mode is not enabled, the ADMC401

always halts before granting the bus. The processors internal state is not affected by granting the bus, and the serial ports remain active during a bus grant, whether or not the processor core halts.

If the ADMC401 is performing an external access when the $\overline{BR}$ signal is asserted, it will not grant the buses until the cycle after the access completes. The entire instruction does not need to be completed when the bus is granted. If a single instruction requires two external accesses, the bus will be granted between the two accesses. The second access is performed after $\overline{BR}$ is removed. When the $\overline{BR}$ input is released, the ADMC401 releases the $\overline{BG}$ signal, re-enables the output drivers and continues program execution from the point where it stopped. $\overline{BG}$ is always deasserted in the same cycle that the removal of $\overline{BR}$ is recognized. The bus request feature operates at all times, including when the ADMC401 is booting and when $\overline{RESET}$ is active. During $\overline{RESET}$, $\overline{BG}$ is asserted in the same cycle that $\overline{BR}$ is recognized. During booting, the bus is granted after the completion of loading of the current byte (including any wait states). Using the bus request during booting is one way to bring the booting operation under control of a host computer.

The ADMC401 has an additional output, Bus Grant Hung $\overline{BGH}$ which lets it operate in a multiprocessor system with a minimum number of wasted cycles. The $\overline{BGH}$ pin asserts when the ADMC401 is ready to execute an instruction but is stopped because the external bus is granted to another device. The other device can release the bus by deasserting bus request. Once the bus is released, the ADMC401 deasserts $\overline{BG}$ and $\overline{BGH}$ and executes the external access.

4.6 POWERDOWN MODES

The ADMC401 includes a powerdown feature that allows the device to enter a very low power dormant state through hardware or software control. In the powerdown mode:

- Internal clocks are disabled
- Processor registers and memory contents are maintained
- Ability to recover from powerdown in less than 100 CLKIN cycles
- Ability to disable internal oscillator when using crystal
- No need to shut down clock for lowest power when using external oscillator
- Interrupt support for *housekeeping* code before entering powerdown and after recovering from powerdown
- User selectable power-up context

4.6.1 Entering Powerdown

The powerdown sequence is initiated by applying a high-to-low transition on the $\overline{PWD}$ pin or by setting the powerdown force control bit (PDFORCE) of the SPORT1 autobuffer/powerdown control register. The DSP core then vectors to the non-maskable powerdown interrupt vector at address 0x002C. Care must be exercise to ensure that multiple powerdown interrupts do not occur or else stack overflow may result. The interrupt service routine at address 0x002C can be used to execute any number of housekeeping instructions prior to the processor entering the powerdown mode. Typically, this is used to configure the powerdown state, disable on-chip peripherals and clear pending interrupts. The DSP subsequently enters the powerdown mode when it executes the IDLE instruction (while $\overline{PWD}$ is asserted). The processor may take either one or two cycles to power down depending on internal clock states during execution of the IDLE instruction. All register and memory contents are maintained in powerdown. Also, all active outputs are held in whatever state they are in before going into powerdown. If an RTI instruction is executed before the IDLE instruction, the processor returns from the powerdown interrupt and the powerdown sequence is aborted.

4.6.2 Exiting Powerdown

The powerdown mode can be exited with the use of the $\overline{\text{PWD}}$ pin or with the $\overline{\text{RESET}}$ pin. There are also several user-selectable modes for startup from powerdown which specify a startup delay as well as specify the program flow after startup. This allows the program to resume from where it left off before powerdown or for the program context to be cleared. Applying a low-to-high transition on the $\overline{\text{PWD}}$ pin will take the processor out of powerdown. The amount of time it takes for the processor to come out of powerdown is controllable with the *delay startup from powerdown* control bit (XTALDELAY, bit 14 of the Powerdown Control Register). If this bit is cleared, no additional delay over the quick startup (100 cycles) is introduced. If this bit is set, a delay of 4096 cycles is introduced.

The context for exiting powerdown is set by bit 12 (PUCR) of the Powerdown Control Register. If this bit is cleared, the processor will continue to execute instructions following the IDLE instruction following the low-to-high transition on the $\overline{\text{PWD}}$ pin. When the RTI instruction is encountered in the interrupt service routine for the powerdown, operation is returned to the main routine. If the PUCR bit is set for a clear context, the processor resumes operation from powerdown by clearing the PC, STATUS, LOOP and CNTR registers. The IMASK and ASTAT registers are cleared and the SSTAT goes to 0x55. The processor starts execution at address 0x0000.

Active output pins retain their states during powerdown. In addition, interrupts are latched and can be services if the ADMC401 exits powerdown with PUCR = 0. It is possible to clock data into or out of the serial ports during powerdown by supplying an external serial clock. Data clocked into the ADMC401 will remain in the RX registers. These activities cause additional power consumption.

If $\overline{\text{RESET}}$ is activated while the ADMC401 is in the powerdown mode, the device is reset and instructions are executed and the device is boot loaded according to the settings of MMAP and BMODE pins. When exiting powerdown with $\overline{\text{RESET}}$, the XTALDELAY control bit is ignored.

4.6.3 Startup Time After Powerdown

The time required to exit the powerdown state depends on whether an internal or external oscillator is used, and the method used to exit powerdown. When the ADMC401 is in powerdown, the external clock signal (assuming the device is driven by an external TTL/CMOS clock) is ignored if the XTALDIS bit (bit 15 of the Powerdown Control Register) is set. It is therefore not necessary to stop the external clock since no power is wasted while this external clock is running. After the processor comes out of powerdown by either the $\overline{\text{PWD}}$ or $\overline{\text{RESET}}$ pins, it will begin executing after a maximum startup time of 100 CLKIN cycles as long as the clock oscillator is stable and at the same frequency as before powerdown.

If the external clock is unstable when the ADMC401 exits powerdown, then the XTALDELAY control bit can be used to insert an additional 4096 cycle delay into the startup time. This delay can only be inserted when the ADMC401 is brought out of powerdown by the $\overline{\text{PWD}}$ pin.

If the processor is taken out of powerdown by the $\overline{\text{RESET}}$ line and the clock is stable and at the same frequency as before powerdown, the $\overline{\text{RESET}}$ need only be held for 5 cycles.

If using an external crystal and the internal oscillator of the ADMC401, the startup sequence is somewhat different in that a trade-off must be made between a fast startup time and minimal power consumption during powerdown. If a fast startup is desired, then both the XTALDELAY and XTALDIS bits must be cleared so that the oscillator will continue to operate (and consume power) during powerdown and no startup delay time is inserted. In this configuration, the ADMC401 will begin execution at least 100 CLKIN cycles after the low-to-high transition at the $\overline{\text{PWD}}$ pin. If lowest possible power consumption possible is required, then both the XTALDELAY and XTALDIS bits must be set before entering powerdown (with the IDLE instruction). If the $\overline{\text{RESET}}$ line is used to exit powerdown and the clock has been stopped (XTALDIS is set), then the

$\overline{\text{RESET}}$ line must be held low for 1000 CLKIN cycles plus the time required for the PLL to lock and the crystal oscillator to stabilize (2000 CLKIN cycles). If the clock is running during powerdown, the $\overline{\text{RESET}}$ need only be asserted for 5 cycles.

4.6.4 The PWDACK Pin

The PWDACK pin is an output that indicates when the ADMC401 is in the powerdown mode. This pin is driven high by the processor when it has powered down. It is driven low after the processor has completed the power-up sequence. A low level on the PWDACK pin also indicates that there is a valid CLKOUT signal and that instruction execution has begun.

4.6.5 Using Powerdown as a Non-Maskable Interrupt

The powerdown interrupt is never masked. It is possible to use this interrupt for other purposes, if desired. The ADMC401 does not go into powerdown until the IDLE instruction is executed. If an RTI is executed instead the processor returns from the powerdown interrupt routine and the powerdown sequence is aborted.

5.0 THE ANALOG TO DIGITAL CONVERSION SYSTEM

5.1 OVERVIEW

The ADMC401 contains a fast, high-accuracy, multiple-input analog to digital conversion system that is needed for accurate measurement of currents, voltages and other signals in high performance motor control systems. A functional block diagram of the entire ADC system is shown in Figure 5.

The ADC system consists of an 8-to-1 line multiplexer, a sample and hold amplifier and a 12-bit pipeline flash analog to digital converter. Up to eight analog inputs may be applied at the pins VIN1 to VIN8. The output of the internal analog multiplexer is available at the AMUXOUT pin. It is necessary to connect a high quality unity gain non-inverting amplifier between the AMUXOUT and VINA pins to provide sufficient drive to the sample and hold amplifier. In normal operation, the ADC system converts the eight analog input signals sequentially on receipt of a convert start command. The convert start trigger may be internally or externally generated. As the conversion for a given analog input channel is completed, the corresponding digital number is written to a dedicated 16-bit, 2's complement, left-aligned register that is memory mapped to the data memory space of the DSP core. The ADC may use either an internally generated 2.5 V reference voltage or an externally supplied reference voltage level at the VREF pin. When selected, the internally generated reference voltage is also made available at the VREF pin for use by external circuitry. Operation of the ADC system is controlled by the ADCCTRL register. Another register, ADCTEST, may be used to enable a manual channel select feature that permits a single channel only to be selected for conversion on the occurrence of a convert start command.

5.2 CONVERT START COMMAND

The analog to digital conversion process (consisting of the sequential conversion of all eight analog inputs or the channel selected in the ADCTEST register) of the ADMC401 may be started by either an internal or an external command. Bit 0 of the ADCCTRL register determines whether internal or external convert start mode is enabled. If bit 0 of the ADCCTRL register is cleared, internal convert start mode is selected and the ADC conversion process is started on the rising edge of the PWMSYNC signal. Therefore, in this internal convert start mode, there is one conversion process per PWM period when the single update mode is enabled in the PWM subsystem. In double update PWM mode, there are two complete conversion processes per PWM period.

If bit 0 of the ADCCTRL register is set, external convert start mode is selected. In this mode, the conversion process is started on the occurrence of a rising edge on the CONVST pin. The start of conversion could be placed under software control by connecting one of the programmable input/output lines to the CONVST pin and generating a rising edge by

writing to the PIODATA register. By default, following a power on or reset, bit 0 of the ADCCTRL register is cleared so that internal convert start mode is selected.

5.3 ADC CHANNEL SELECT, ADCTEST REGISTER

The ADCTEST register can be used to overwrite the normal sequential conversion process of the ADC. Instead a single channel can be selected by programming the address into ADCTEST(2::0). For example, writing ADCTEST(2::0) = 0x00 selects analog input VIN1 for conversion, ADCTEST(2::0) = 0x001 selects analog input VIN2 etc. Bit 3 of the ADCTEST register controls the mode of the conversion. If ADCTEST(3) is set, the single channel convert mode is selected and the channel to be converted is read from ADCTEST(2::0). If ADCTEST(3) is cleared the normal sequence of eight channels converted is selected. The default value is that ADCTEST(3) is cleared to select the eight channel sequence mode.

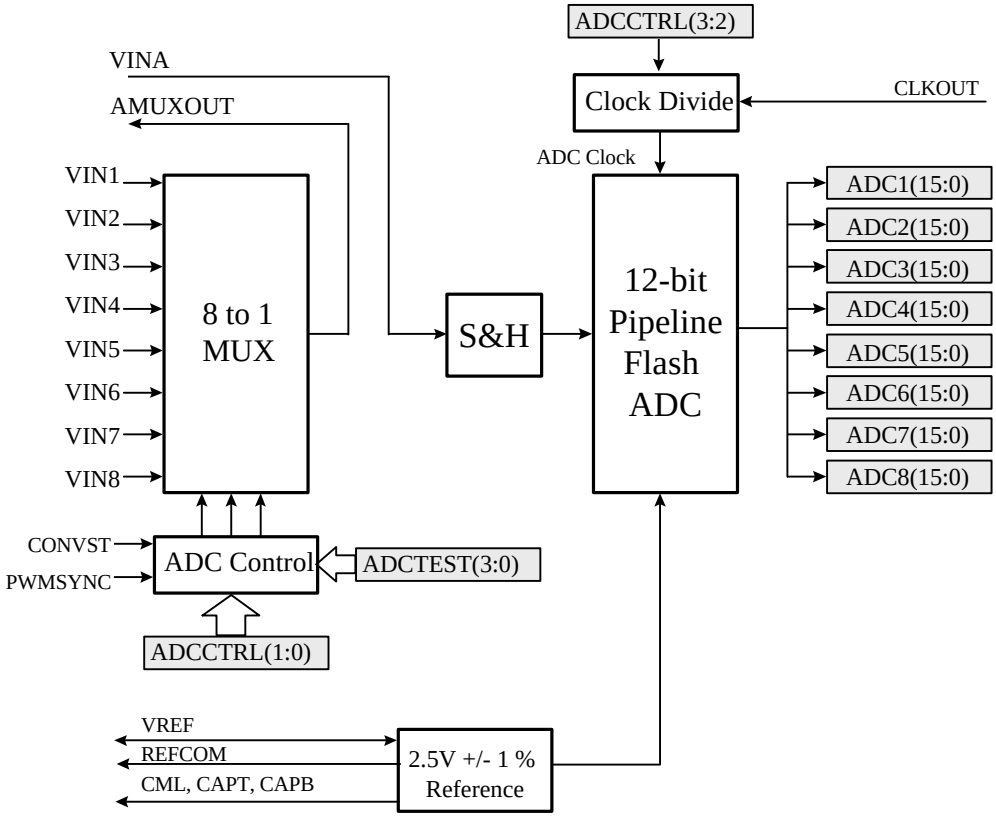


Figure 5: Functional block diagram of the ADC system of the ADMC401.

5.4 ADC TIMING AND CLOCK SIGNALS

The ADC consists of a three stage pipeline flash architecture and is clocked at an integer fraction of the DSP instruction rate. All of the timing of the ADC system is regulated by this clock signal and it determines the total conversion time as well as the skew between sampling of successive input channels. The ADC clock rate is controlled by bits 2 and 3 of the ADCCTRL register. The period of the ADC clock, $t_{CK,ADC}$ is related to the DSP CLKOUT period by:

$$t_{CK,ADC} = (N+1) \times t_{CK}$$

where N is the 2 bit value written to ADCCTRL(3::2). Clearing both bits 2 and 3 of the ADCCTRL register disables the ADC clock signal, effectively powering down the ADC system. By default, following a power on or reset, bit 2 of the ADCCTRL register is cleared and bit 3 is set so that the ADC clock frequency is a third of the CLKOUT rate (or 8.67 MHz for a 26 MHz CLKOUT). This corresponds to a $t_{CK,ADC}$ of approximately 115 ns.

The timing of the ADC conversion process is shown in Figure 6 where the convert start command may be derived either internally or externally, as described previously. At the rising edge of the convert start signal, the analog voltage on the VIN1 pin is automatically fed through the multiplexer and sampled by the sample and hold amplifier. It takes one ADC clock cycle to successfully select an analog input channel and lock onto it with the sample & hold amplifier. It takes a further three and a half clock cycles to complete the conversion of the sampled voltage. Therefore, four and a half clock cycles ($4.5t_{CK,ADC}$) after the sample instant the digital output word is latched to the corresponding ADC1 register. However, because of the pipeline nature of the ADC, it is possible to sample the second analog input, VIN2, after only one clock cycle. Therefore the time skew between samples of successive analog inputs is only one clock period, $t_{CK,ADC}$. Therefore, in the second clock period, the sampled V_2 goes through the first stage of the ADC pipeline while the sample of VIN3 goes through the second stage.

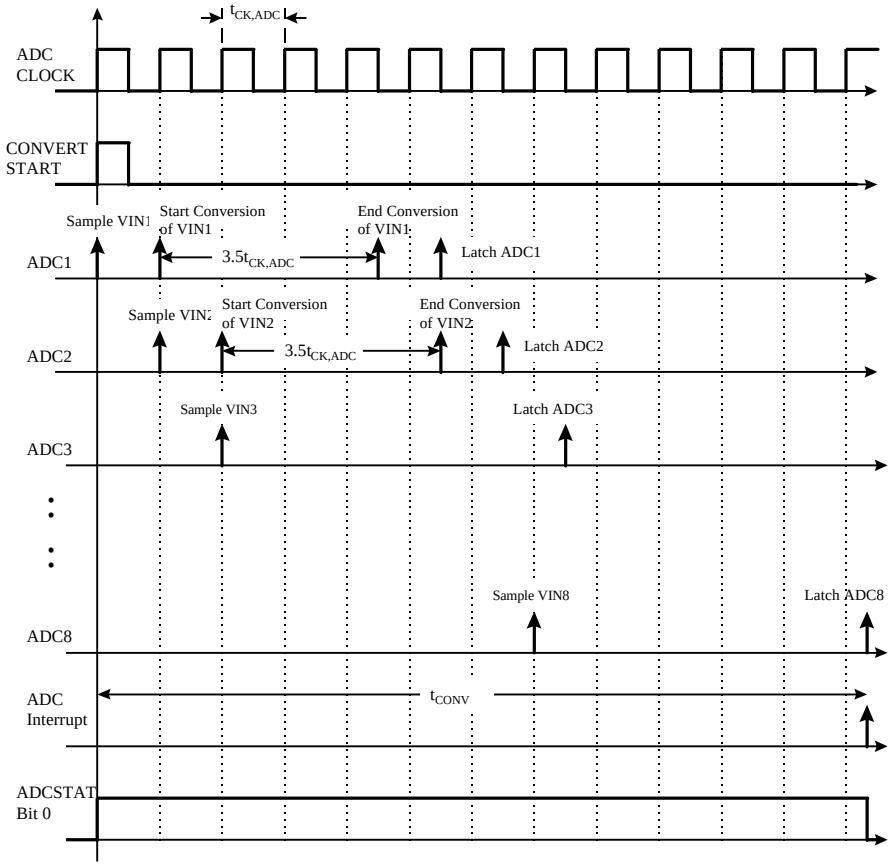


Figure 6: Representation of timing of ADC conversion process.

Clearly, this pipelined sampling and conversion process continues until all eight channels have been sequentially sampled, converted and the corresponding digital word written to the appropriate register. The input channels are always converted in

the same order and all channels are converted at each convert start trigger. As a result, it can be seen that the entire conversion time for the eight channels is equal to seven time skews plus the time required to convert the final channel or:

$$t_{CONV} = 7 \times t_{SKEW} + 4.5 \times t_{CK,ADC}$$
$$= 11.5 \times t_{CK,ADC}$$

so that with an 8.67 MHz ADC clock rate (default value at 26 MHz CLKOUT) the eight analog inputs are converted in slightly more than 1.32 μs. Any additional delays in latching the data into the appropriate registers and performing any other tasks must be such that at most 2 μs after the convert start signal, all data in the registers ADC1 to ADC8 is valid.

5.5 ADC STATUS AND INTERRUPTS

At the end of the conversion process as illustrated in Figure 6, an ADC interrupt is generated that is applied to the peripheral interrupt controller (PIC) block. This interrupt signals that the conversion of all eight channels has been completed and that the data in the eight ADC registers (ADC1 to ADC8) is now valid. This interrupt can be masked by the PICMASK register as part of the PIC control and operation. A one bit read only ADCSTAT register is used to give the operational status of the ADC. While the ADC is converting this status bit is set. Reading this register does not alter the contents of the register. This register can be used in a polling routine to wait until the end of conversion. Following the completion of the conversion process and the generation of the ADC interrupt, this status bit is cleared, automatically. It is possible (particularly when using external convert start mode) that a second convert start trigger may be applied while the ADC is still converting. In this case, the present conversion process continues as normal and the additional convert start signal is ignored. Therefore, the onus is on the user of the ADC block to ensure that multiple external convert start signal not be applied to the device as otherwise convert start triggers will be missed.

5.6 VOLTAGE REFERENCES

The ADMC401 contains an internal 2.5V ± 1% voltage reference that may be used by the ADC to eliminate the need for an external voltage reference. The output of this voltage reference is made available at the VREF pin when internal reference is selected and can be used by external biasing and scaling circuitry, if required. To prevent excessive loading of this internal reference, it should be buffered in a suitable unity gain, non-inverting stage prior to use by external circuitry. The internal voltage reference is selected by clearing bit 1 of the ADCCTRL register. Alternatively, if the required accuracy or thermal drift characteristics of the application require a more accurate reference, it may be connected at the VREF pin. Use of this external voltage reference for the ADC conversion is selected by setting bit 1 of the ADCCTRL register. The default condition following power up or reset is that bit 1 of the ADCCTRL register is cleared and the internal voltage reference is selected.

5.7 ADC TRANSFER CHARACTERISTICS

The analog input voltages on input pins have a nominal input voltage range of 4.0 V_{pp}. In normal operation a 2.5V reference voltage is used and the input voltages may be in the range 0.5V to 4.5V. Consequently, the transfer characteristics of each of the ADC channels may be written:

$$ADCn = \begin{cases} 0x7FF0 & \textit{if } VINn = 4.5V \\ 0x0000 & \textit{if } VINn = 2.5V \\ 0x8000 & \textit{if } VINn = 0.5V \end{cases}$$

where n (1 to 8) is the channel number. Therefore, the 12-bit data from the ADC is stored in 2's complement, left-aligned form in the 16-bit data registers. In normal operation (analog input voltage is in range), the least four significant bits of ADC1 to ADC8 are always cleared. The ADC also provides an out of range detector that is set when the input signal of a given channel exceeds the allowable input voltage range. When this condition is detected for a given analog input, the least significant bit (bit 0) of the corresponding register is set. Consequently, the condition of the analog input signal for a given channel may be established by examination of both the MSB and LSB of the associated ADC data register, according to Table 5.

MSB	LSB	Input Voltage Is
0	0	In Range, 2.5 V ≤ VIN_n ≤ 4.5 V
1	0	In Range, 0.5 V ≤ VIN_n < 2.5 V
0	1	Over Range, VIN_n > 4.5 V
1	0	Under Range, VIN_n < 0.5 V

Table 5: Out of range indication for analog input voltages.

5.8 ADC REGISTERS

The structure of the ADC registers is described in Figure 7.

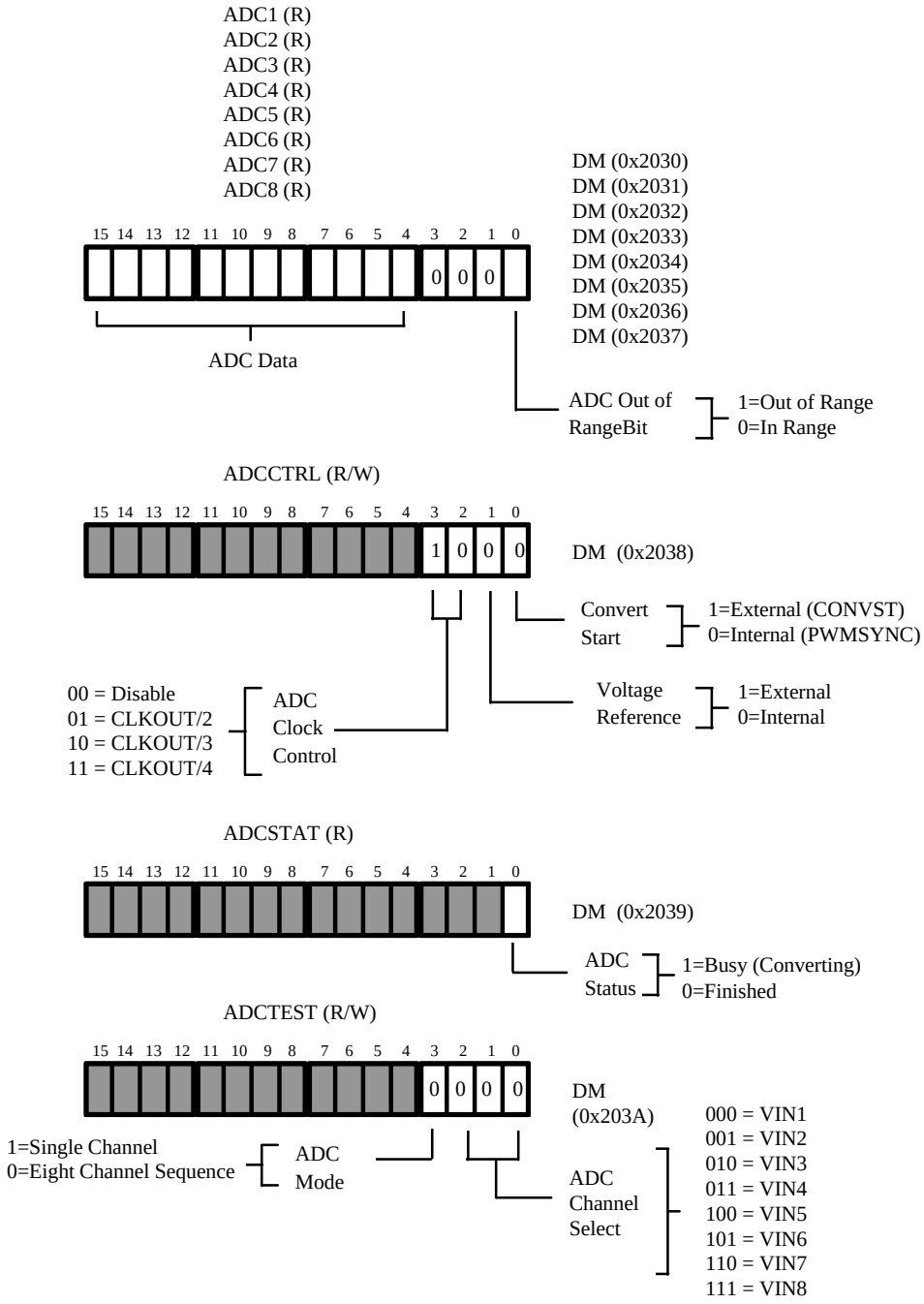


Figure 7: Structure of ADC Registers of ADMC401 (shaded bits are unused, reset values included, where defined).

6. THE PWM CONTROLLER.

6.1 OVERVIEW

The PWM generator block of the ADMC401 is a flexible, programmable, three-phase PWM waveform generator that can be programmed to generate the required switching patterns to drive a three-phase voltage source inverter for ac induction (ACIM) or permanent magnet synchronous (PMSM) motor control. In addition, the PWM block contains special functions that considerably simplify the generation of the required PWM switching patterns for control of the electronically commutated motor (ECM) or brushless dc motor (BDCM). A special mode for switched reluctance motors (SRM) is enabled by tying a dedicated pin, $\overline{\text{PWMSR}}$ to GND.

The PWM generator produces three pairs of PWM signals on the six PWM output pins (AH, AL, BH, BL, CH and CL). The six PWM output signals consist of three high-side drive signals (AH, BH and CH) and three low-side drive signals (AL, BL and CL). The polarity of the generated PWM signals may be programmed by the PWMPOL pin, so that either active HI or active LO PWM patterns can be produced by the ADMC401. The switching frequency, dead time and minimum pulse widths of the generated PWM patterns are programmable using respectively the PWMTM, PWMDT and PWMPD registers. In addition, three duty-cycle control registers (PWMCHA, PWMCHB and PWMCHC) directly control the duty cycles of the three-pairs of PWM signals.

Each of the six PWM output signals can be enabled or disabled by separate output enable bits of the PWMSEG register. In addition, three control bits of the PWMSEG register permit crossover of the two signals of a PWM pair for easy control of ECM or BDCM. In crossover mode, the PWM signal destined for the high-side switch is diverted to the complementary low-side output and the signal destined for the low-side switch is diverted to the corresponding high-side output signal. In addition to ease of use of the PWM controller for ECM or BDCM, this crossover mode can also be used to transition the PWM signals into the over-modulation range with relative ease.

In many applications, there is a need to provide an isolation barrier in the gate-drive circuits that turn on the power devices of the inverter. In general, there are two common isolation techniques, optical isolation using opto-couplers and transformer isolation using pulse transformers. The PWM controller of the ADMC401 permits mixing of the output PWM signals with a high-frequency chopping signal to permit easy interface to such pulse transformers. The features of this gate-drive chopping mode can be controlled by the PWMGATE register. There is an 8-bit value within the PWMGATE register that directly controls the chopping frequency. In addition, high-frequency chopping can be independently enabled for the high-side and the low-side outputs using separate control bits in the PWMGATE register.

The PWM generator is capable of operating in two distinct modes, single update mode or double update mode. In single update mode the duty cycle values are programmable only once per PWM period, so that the resultant PWM patterns are symmetrical about the mid-point of the PWM period. In the double update mode, a second updating of the PWM registers is implemented at the mid-point of the PWM period. In this mode, it is possible to produce asymmetrical PWM patterns, that produce lower harmonic distortion in three-phase PWM inverters. This technique also permits closed loop controllers to change the average voltage applied to the machine windings at a faster rate and so permits faster closed loop bandwidths to be achieved. The operating mode of the PWM block (single or double update mode) is selected by a control bit in MODECTRL register.

The PWM generator of the ADMC401 also provides an output pulse on the PWMSYNC pin that is synchronized to the PWM switching frequency. In single update mode a PWMSYNC pulse is produced at the start of each PWM period. In double update mode, an additional PWMSYNC pulse is produced at the mid-point of each PWM period. The width of the PWMSYNC pulse is programmable through the PWMSYNCWT register.

The PWM signals produced by the ADMC401 can be shut-off in a number of different ways. Firstly, there is a dedicated asynchronous PWM shutdown pin, $\overline{\text{PWMTRIP}}$, that, when brought LO, instantaneously places all six PWM outputs in the OFF state (as determined by the state of the PWMPOL pin). In addition, each of the PIO lines of the ADMC401 (PIO0 to PIO11) can be configured to act as an additional PWM shutdown. By setting the appropriate bit in the PIOPWM register, the corresponding PIO line acts as an asynchronous PWM shutdown source in a manner identical to the $\overline{\text{PWMTRIP}}$ pin. These two hardware shutdown mechanisms are asynchronous so that the associated PWM disable circuitry does not go through any clocked logic, thereby ensuring correct PWM shutdown even in the event of a loss of the DSP clock. In addition to the hardware shutdown features, the PWM system may be shutdown in software by writing to the PWMSWT register.

Status information about the PWM system of the ADMC401 is available to the user in the SYSSTAT register. In particular, the state of the $\overline{\text{PWMTRIP}}$ and PWMPOL pins is available, as well as status bits that indicates whether operation is in the first half or the second half of the PWM period and whether switched reluctance mode is enabled or disabled.

A functional block diagram of the PWM controller is shown in Figure 8. The generation of the six output PWM signals on pins AH to CL is controlled by four important blocks:

- The Three-Phase PWM Timing Unit, which is the core of the PWM controller, generates three pairs of complemented and dead time adjusted center based PWM signals.
- The Output Control Unit allows the redirection of the outputs of the Three-Phase Timing Unit for each channel to either the high-side or the low-side output. In addition, the Output Control Unit allows individual enabling/disabling of each of the six PWM output signals.
- The Gate Drive Unit provides the correct polarity output PWM signals based on the state of the PWMPOL pin. The Gate Drive Unit also permits the generation of the high-frequency chopping frequency and its subsequent mixing with the PWM signals.
- The PWM Shutdown Controller takes care of the various PWM shutdown modes (via the $\overline{\text{PWMTRIP}}$ pin, the PIO lines or the PWMSWT register) and generates the correct RESET signal for the Timing Unit.

The PWM controller is driven by a clock at the same frequency as the DSP instruction rate, CLKOUT and is capable of generating two interrupts to the DSP core. One interrupt is generated on the occurrence of rising edge on the PWMSYNC pulse and the other is generated on the occurrence of any PWM shutdown action.

6.2 THREE-PHASE TIMING UNIT

The 16-bit three-phase timing unit is the core of the PWM controller and produces three pairs of pulsewidth modulated signals with high resolution and minimal processor overhead. The outputs of this timing unit are active LO such that a low level is interpreted as a command to turn ON the associated power device. There are four main configuration registers (PWMTM, PWMDT, PWMPD and PWMSYNCWT) that determine the fundamental characteristics of the PWM outputs. In addition, the operating mode of the PWM (single or double update mode) is selected by bit 6 of the MODECTRL register. These registers in conjunction with the three 16-bit duty cycle registers (PWMCHA, PWMCHB and PWMCHC) control the output of the three-phase timing unit.

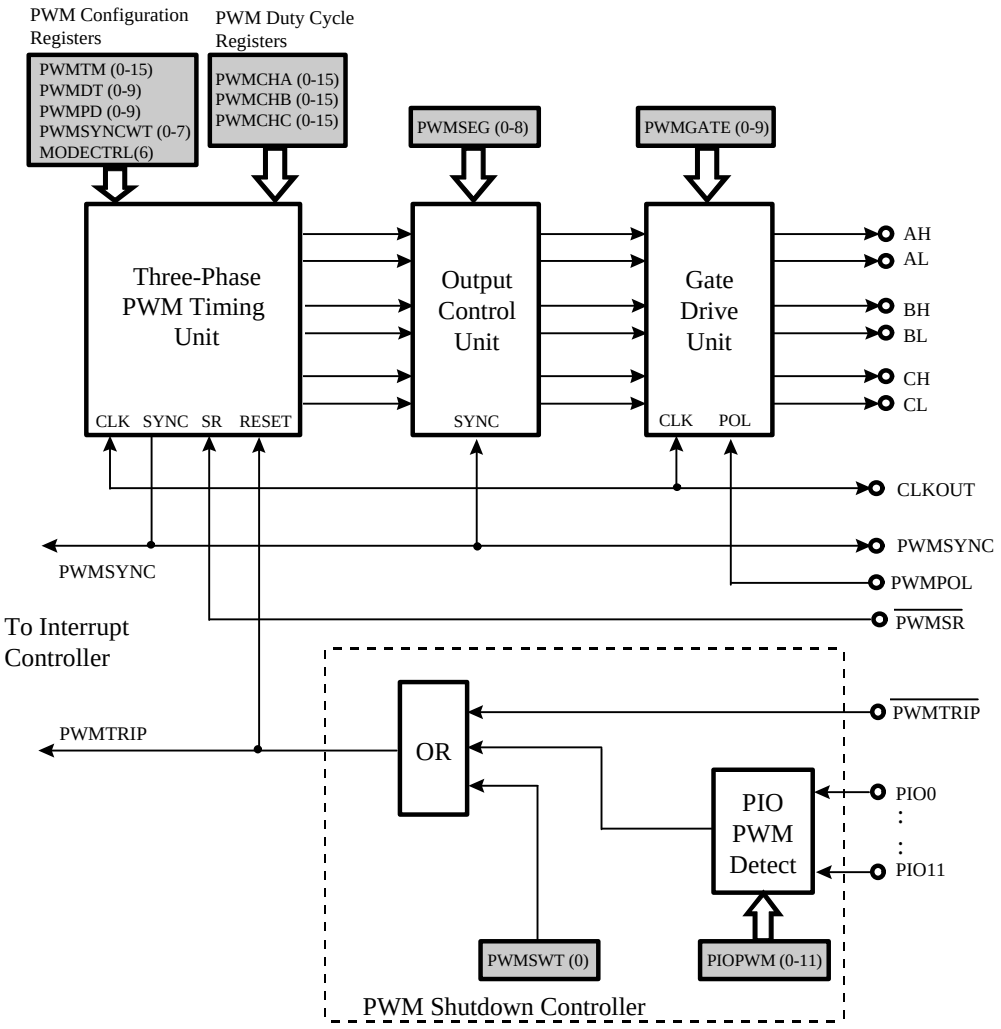


Figure 8: Overview of the PWM Controller of the ADMC401.

6.2.1 PWM Switching Frequency, PWMTM Register

The PWM switching frequency is controlled by the 16-bit read/write PWM period register, PWMTM. The fundamental timing unit of the PWM controller is t_{CK} (DSP instruction rate). Therefore, for a 26 MHz CLKOUT, the fundamental time increment is 38.5 ns. The value written to the PWMTM register is effectively the number of t_{CK} clock increments in half a PWM period. The required PWMTM value as a function of the desired PWM switching frequency (f_{PWM}) is given by:

$$\text{PWMTM} = \frac{f_{CLKOUT}}{2 \times f_{PWM}} = \frac{f_{CLKIN}}{f_{PWM}}$$

Therefore, the PWM switching period, T_s, can be written as:

$$T_s = 2 \times \text{PWMTM} \times t_{CK}$$

For example, for a 26 MHz CLKOUT and a desired PWM switching frequency of 10 kHz (T_s = 100 μs), the correct value to load into the PWMTM register is:

$$PWMTM = \frac{26 \times 10^6}{2 \times 10 \times 10^3} = 1300$$

The largest value that can be written to the 16-bit PWMTM register is 0xFFFF = 65,535 which corresponds to a minimum PWM switching frequency of:

$$f_{PWM,min} = \frac{26 \times 10^6}{2 \times 65{,}535} = 198 \text{ Hz}$$

6.2.2 PWM Switching Dead Time, PWMDT Register

The second important parameter that must be set up in the initial configuration of the PWM block is the switching dead time. This is a short delay time introduced between turning off one PWM signal (say AH) and turning on the complementary signal, AL. This short time delay is introduced to permit the power switch being turned off (AH in this case) to completely recover its blocking capability before the complementary switch is turned on. This time delay prevents a potentially destructive short-circuit condition from developing across the dc link capacitor of a typical voltage source inverter.

The dead time is controlled by the 10-bit, read/write PWMDT register. There is only one dead time register that controls the dead time inserted into the three pairs of PWM output signals. The dead time, T_d, is related to the value in the PWMDT register by:

$$T_d = PWMDT \times 2 \times t_{CK}$$

Therefore, a PWMDT value of 0x00A (= 10), introduces a 770 ns delay between the turn off on any PWM signal (say AH) and the turn on of its complementary signal (AL). The amount of the dead time can therefore be programmed in increments of $2t_{CK}$ (or 77 ns for a 26 MHz CLKOUT). The PWMDT register is a 10-bit register so that its maximum value is 0x3FF (= 1023) corresponding to a maximum programmed dead time of:

$$T_{d,max} = 1023 \times 2 \times t_{CK} = 1023 \times 2 \times 38.5 \times 10^{-9} = 78.77 \mu s$$

for a CLKOUT rate of 26 MHz. Obviously, the dead time can be programmed to be zero by writing 0 to the PWMDT register.

6.2.3 PWM Operating Mode, MODECTRL & SYSSTAT Registers

The PWM controller of the ADMC401 can operate in two distinct modes; single update mode and double update mode. The operating mode of the PWM controller is determined by the state of bit 6 of the MODECTRL register. If this bit is cleared the PWM operates in the single update mode. Setting bit 6 places the PWM in the double update mode. By default, following either a peripheral reset or power on, bit 6 of the MODECTRL register is cleared so that the default operating mode is single update mode.

In single update mode, a single PWMSYNC pulse is produced in each PWM period. The rising edge of this signal marks the start of a new PWM cycle and is used to latch new values from the PWM configuration registers (PWMTM, PWMDT, PWMPD and PWMSYNCWT) and the PWM duty cycle registers (PWMCHA, PWMCHB and PWMCHC) into the three-phase timing unit. In addition, the PWMSEG register is also latched into the output control unit on the rising edge of the PWMSYNC pulse. In effect, this means that the characteristics and resultant duty cycles of the PWM signals can be

updated only once per PWM period at the start of each cycle. The result is that PWM patterns that are symmetrical about the mid-point of the switching period are produced.

In double update mode, there is an additional PWMSYNC pulse produced at the mid-point of each PWM period. The rising edge of this new PWMSYNC pulse is again used to latch new values of the PWM configuration registers, duty cycle registers and the PWMSEG register. As a result it is possible to alter both the characteristics (switching frequency, dead time, minimum pulse width and PWMSYNC pulsewidth) as well as the output duty cycles at the mid-point of each PWM cycle. Consequently, it is possible to produce PWM switching patterns that are no longer symmetrical about the mid-point of the period (asymmetrical PWM patterns).

In the double update mode, it may be necessary to know whether operation at any point in time is in either the first half or the second half of the PWM cycle. This information is provided by bit 3 of the SYSSTAT register which is cleared during operation in the first half of each PWM period (between the rising edge of the original PWMSYNC pulse and the rising edge of the new PWMSYNC pulse introduced in double update mode). Bit 3 of the SYSSTAT register is set during operation in the second half of each PWM period. This status bit allows the user to make a determination of the particular half-cycle during implementation of the PWMSYNC interrupt service routine, if required.

The advantage of the double update mode is that lower harmonic voltages can be produced by the PWM process and faster control bandwidths are possible. However, for a given PWM switching frequency, the PWMSYNC pulses occur at twice the rate in the double update mode. Since, new duty cycle values must be computed in each PWMSYNC interrupt service routine, there is a larger computational burden on the DSP in the double update mode. Alternatively, the same PWM update rate may be maintained at half the switching frequency to give lower switching losses.

6.2.4 Width of the PWMSYNC pulse, PWMSYNCWT register.

The PWM controller of the ADMC401 produces an output PWM synchronization pulse at a rate equal to the PWM switching frequency in single update mode and at twice the PWM frequency in the double update mode. This pulse is available for external use at the PWMSYNC pin. The width of this PWMSYNC pulse is programmable by the 8-bit read/write PWMSYNCWT register. The width of the PWMSYNC pulse, $T_{PWMSYNC}$, is given by:

$$T_{PWMSYNC} = t_{CK} \times (PWMSYNCWT + 1)$$

so that the width of the pulse is programmable from t_{CK} to $256t_{CK}$ (corresponding to 38.5 ns to 9.856 μs for a CLKOUT rate of 26 MHz). Following a reset, the PWMSYNCWT register contains 0x27 (= 39) so that the default PWMSYNC width is 1.54 μs, again for a 26 MHz CLKOUT.

6.2.5 PWM Duty Cycles, PWMCHA, PWMCHB, PWMCHC Registers

The duty cycles of the six PWM output signals on pins AH to CL are controlled by the three 16-bit read/write duty cycle registers, PWMCHA, PWMCHB and PWMCHC. The integer value in the register PWMCHA controls the duty cycle of the signals on AH and AL, in PWMCHB control the duty cycle of the signals on BH and BL and in PWMCHC controls the duty cycle of the signals on CH and CL. The duty cycle registers are programmed in integer counts of the fundamental time unit, t_{CK}, and define the desired on-time of the high-side PWM signal produced by the three-phase timing unit over half the PWM period. The switching signals produced by the three-phase timing unit are also adjusted to incorporated the programmed dead time value in the PWMDT register. The three-phase timing unit produces active LO signals so that a LO level corresponds to a command to turn on the associated power device.

A typical pair of PWM outputs (in this case for AH and AL) from the timing unit are shown in Figure 9 for operation in single update mode. All illustrated time values indicate the integer value in the associated register and can be converted to time by simply multiplying by the fundamental time increment, t_{CK}. Firstly, it is noted that the switching patterns are perfectly symmetrical about the mid-point of the switching period in this single update mode since the same values of PWMCHA, PWMTM and PWMDT are used to define the signals in both half cycles of the period. It can be seen how the programmed duty cycles are adjusted to incorporate the desired dead time into the resultant pair of PWM signals. Clearly, the dead time is incorporated by moving the switching instants of both PWM signals (AH & AL) away from the instant set by the PWMCHA register. Both switching edges are moved by an equal amount (PWMDT*t_{CK}) to preserve the symmetrical output patterns. Also shown is the PWMSYNC pulse whose width is set by the PWMSYNCWT register and bit 3 of the SYSSTAT register that indicates whether operation is in the first or second half cycle of the PWM period.

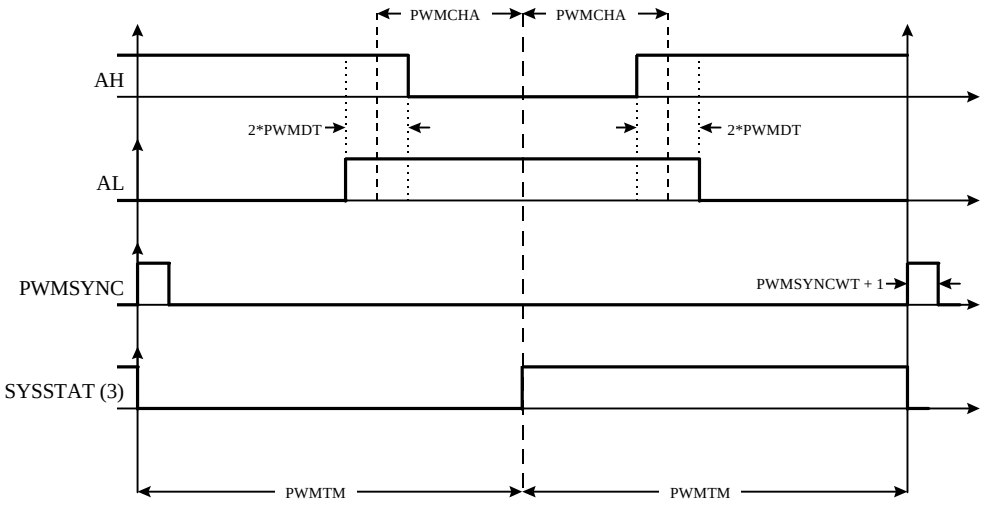


Figure 9: Typical PWM outputs of Three-Phase Timing Unit in Single Update Mode (active LO waveforms).

The resultant on-times of the PWM signals over the full PWM period (two half periods) produced by the PWM timing unit and illustrated in Figure 9 may be written as:

$$T_{AH} = 2 \times (\text{PWMCHA} - \text{PWMDT}) \times t_{CK}$$

$$T_{AL} = 2 \times (\text{PWMTM} - \text{PWMCHA} - \text{PWMDT}) \times t_{CK}$$

and the corresponding duty cycles are:

$$d_{AH} = \frac{T_{AH}}{Ts} = \frac{\text{PWMCHA} - \text{PWMDT}}{\text{PWMTM}}$$

$$d_{AL} = \frac{T_{AL}}{Ts} = \frac{\text{PWMTM} - \text{PWMCHA} - \text{PWMDT}}{\text{PWMTM}}$$

Obviously negative values of T_{AH} and T_{AL} are not permitted and the minimum permissible value is zero, corresponding to a 0% duty cycle. In a similar fashion, the maximum value is T_s, corresponding to a 100% duty cycle.

The output signals from the timing unit for operation in double update mode are shown in Figure 10. This illustrates a completely general case where the switching frequency, dead time and duty cycle are all changed in the second half of the PWM period. Of course, the same value for any or all of these quantities could be used in both halves of the PWM cycle. However, it can be seen that there is no guarantee that symmetrical PWM signal will be produced by the timing unit in this double update mode. Additionally, it is seen that the dead time is inserted into the PWM signals in the same way as in the single update mode.

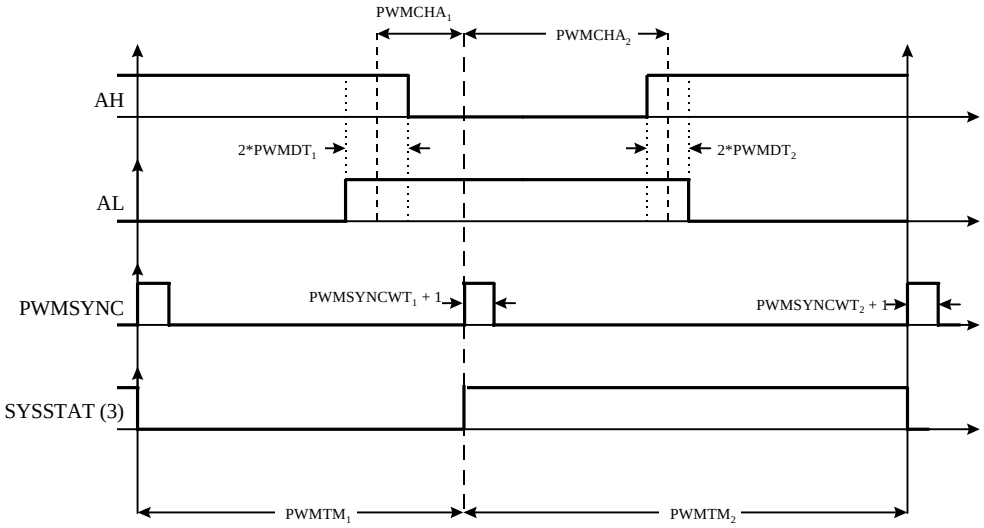


Figure 10: Typical PWM outputs of Three-Phase Timing Unit in Double Update Mode (active LO waveforms).

In general the on-times of the PWM signals over the full PWM period in double update mode can be defined as:

$$T_{AH} = \left(PWMCHA_1 + PWMCHA_2 - PWMDT_1 - PWMDT_2\right) \times t_{CK}$$

$$T_{AL} = \left(PWMTM_1 + PWMTM_2 - PWMCHA_1 - PWMCHA_2 - PWMDT_1 - PWMDT_2\right) \times t_{CK}$$

where the subscript 1 refers to the value of that register during the first half cycle and the subscript 2 refers to the value during the second half cycle. The corresponding duty cycles are:

$$d_{AH} = \frac{T_{AH}}{T_s} = \frac{\left(PWMCHA_1 + PWMCHA_2 - PWMDT_1 - PWMDT_2\right)}{\left(PWMTM_1 + PWMTM_2\right)}$$

$$d_{AL} = \frac{T_{AL}}{T_s} = \frac{\left(PWMTM_1 + PWMTM_2 - PWMCHA_1 - PWMCHA_2 - PWMDT_1 - PWMDT_2\right)}{\left(PWMTM_1 + PWMTM_2\right)}$$

since for the completely general case in double update mode, the switching period is given by:

$$T_s = \left(\mathrm{PWMTM}_1 + \mathrm{PWMTM}_2\right) \times t_{CK}$$

Again, the values of T_{AH} and T_{AL} are constrained to lie between zero and T_s. Similar PWM signals to those illustrated in Figure 9 and Figure 10 can be produced on the BH, BL, CH and CL outputs by programming the PWMCHB and PWMCHC registers in a manner identical to that described for PWMCHA.

6.2.6 Special Consideration for PWM Operation in Overmodulation

The PWM Timing Unit is capable of producing PWM signals with variable duty cycle values at the PWM output pins. At the extremities of the modulation process, both 0% and 100% modulation are possible. These two modes are termed **full OFF** and **full ON** respectively. In between, for other duty cycle values, the operation is termed **normal modulation**.

- **FULL ON:** The PWM for any pair of PWM signals is said to operate in FULL ON when the desired HI side output of the three-phase Timing Unit is in the ON state (LO) between successive PWMSYNC pulses. This state may be entered by virtue of the commanded duty cycle values (in conjunction with the PWMDT register) or by virtue of the correct operation of the pulse deletion circuit.
- **FULL OFF:** The PWM for any pair of PWM signals is said to operate in FULL OFF when the desired HI side output of the three-phase Timing Unit is in the OFF state (HI) between successive PWMSYNC pulses. This state may be entered by virtue of the commanded duty cycle values (in conjunction with the PWMDT register) or by virtue of the correct operation of the pulse deletion circuit.
- **NORMAL MODULATION:** The PWM for any pair of PWM signals is said to operate in normal modulation when the desired output duty cycle is other than 0% or 100% between successive PWMSYNC pulses.

There are certain situation when transitioning either into or out of either full ON or full OFF where it is necessary to insert additional dead time delays to prevent potential shoot through conditions in the inverter. The particular situation also depends on whether operation is in single or double update mode. In double update mode, it is also necessary to consider whether the PWM unit is transitioning from the first half cycle to the second half cycle or vica versa.

The insertion of the additional dead time into one of the PWM signals of a given pair during these transitions is only needed if otherwise both PWM signals would be required to toggle at the PWMSYNC boundary. The additional dead time delay is inserted into the PWM signal that is toggling into the ON state. In effect the turn ON of this signal is delayed by an amount 2*PWMDT*t_{CK} from the rising edge of PWMSYNC. After this delay, the PWM signal is allowed to turn ON provided, the desired output is still the ON state after the dead time delay.

Figure 11 illustrates two examples of such transitions where in Figure 11(a) when transitioning from normal modulation to full on at the half cycle boundary in double update mode, no special action is needed. However in Figure 11(b) when transitioning into full off at the same boundary, it can be seen that an additional dead time is necessary. Clearly, this inserted dead time is a little different to the normal dead time as it is impossible to move one of the switching events back in time as this would take it into the previous modulation cycle. Therefore, the entire dead time is inserted by delaying the turn on of the appropriate signal by the full amount. Table 6 summarizes the different possible transitions into and out of full ON and full OFF modes and the required actions. It can be seen that the transitions in double update mode at the full cycle boundary are identical to those of single update mode.

MODE	TRANSITION	OPERATION
DUM	Normal to Full ON at half cycle boundary	No action
DUM	Full ON to Normal at half cycle boundary	No action

DUM	Normal to Full OFF at half cycle boundary	Insert dead time to AL
DUM	Full OFF to normal at half cycle boundary	Insert dead time to AH
DUM/SUM	Normal to Full ON at full cycle boundary	Insert dead time to AH
DUM/SUM	Full ON to Normal at full cycle boundary	Insert dead time to AL
DUM/SUM	Normal to Full OFF at full cycle boundary	No action
DUM/SUM	Full OFF to normal at full cycle boundary	No action

Table 6: Summary of Operations needed due to transitions into and out of full ON and full OFF states of PWM controller.

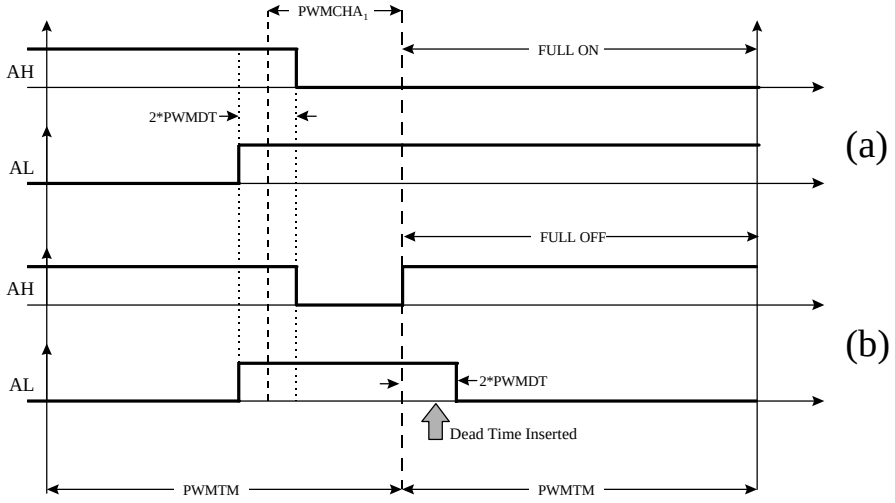


Figure 11: Examples of transitioning form normal modulation into either FULL ON or FULL OFF where it may be necessary to insert additional dead times (a) Transition from normal modulation to FULL ON at half cycle boundary in double update mode where no additional dead time is needed (b) Transition from normal modulation to FULL OFF at half cycle boundary in double update mode where additional dead time is inserted by the PWM controller of the ADMC401.

6.2.7 Minimum Pulse Width, PWMPD Register

In many power converter switching applications, it is desirable to eliminate PWM switching signals below a certain width. It takes a certain finite time to both turn on and turn off power semiconductor devices. Therefore, if the width of any of the PWM signals goes below some minimum value, it may be desirable to completely eliminate the PWM switching for that particular cycle. The allowable minimum pulsewidth for any of the six PWM outputs that can be produced by the PWM controller may be programmed using the 10-bit read/write PWMPD register. The minimum pulsewidth, T_{min} is programmed in increments of $2t_{CK}$ as:

$$T_{min} = 2 \times PWMPD \times t_{CK}$$

so that a PWMPD value of 0x00A defines a permissible minimum on-time of 0.77 μs for a 26 MHz CLKOUT. The operation of the minimum pulsewidth control ensures that the time from turning ON to turning OFF (or alternatively from turning OFF to turning ON) any PWM signal is never less than the T_{min} value as specified by the PWMPD register. If the PWM controller detects that the time between turning ON and turning OFF any one PWM signal (say AH) is less than T_{min},

the PWM pulse is deleted and the PWM signal remains completely OFF over the PWM period. The complementary signal, AL in this case, is then turned completely ON.

6.2.8 PWM Timer Operation

The internal operation of the PWM generation unit is controlled by the PWM timer that is clocked at the DSP instruction rate, with period t_{CK}. The operation of the PWM timer over one full PWM period is illustrated in Figure 12. It can be seen that during the first half cycle (SYSSTAT bit 3 is cleared), the PWM timer decrements from PWMTM to 0. At this point, the count direction changes and the timer continues to increment to the PWMTM value. Also shown in Figure 12 are the PWMSYNC pulses for operation in both single and double update modes. Clearly, an additional PWMSYNC pulse is generated at the mid-point of the PWM cycle in double update mode. Of course, the value of the PWMTM register could be altered at the mid-point in double update mode. In such a case, the duration of the second half period (SYSSTAT bit 3 is set) could be different to that of the first half cycle.

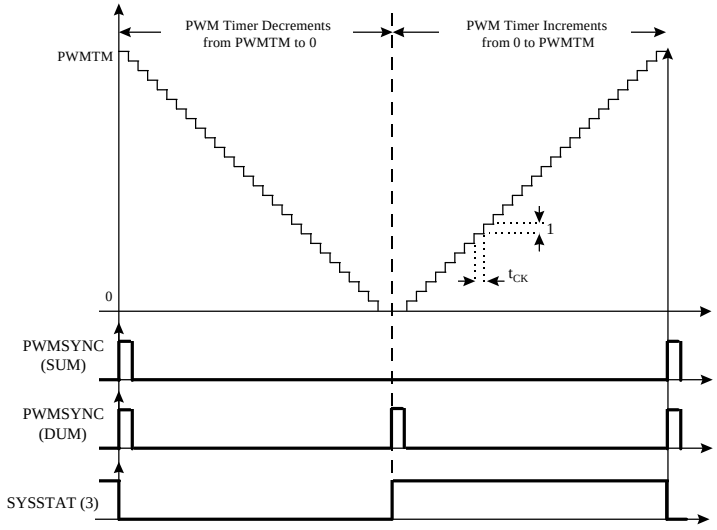


Figure 12: Operation of Internal PWM Timer of ADMC401.

6.2.9 Startup of PWM Controller

It is important to fully understand the operation of the PWM controller during startup. Following a power up or reset operation (either hardware of software), the PWM generation unit is placed in a **lockoff** state and all six PWM output signals (AH to CL) are placed in the OFF state (as defined by the PWMPOL pin). The first write operation to any of the PWMTM, PWMCHA, PWMCHB or PWMCHC registers will cause the PWM generation unit to transition to a **disable** state and the PWM timer begins to increment from zero. During this time all six PWM outputs are still in the OFF state and the PWMSYNC pulse is not produced.

The PWM generation unit does not enter the **normal** state until all four of the PWMTM, PWMCHA, PWMCHB and PWMCHC registers have been written to and the first load signal is generated. The load signal is generated each time the PWM timer reaches the value in the PWMTM register in single update mode. An additional load signal is generated in double update mode when the PWM timer reaches 0. The load signal is used to latch new values into each of the PWMTM, PWMCHA, PWMCHB, PWMCHC, PWMDT, PWMPD, PWMSEG and PWMSYNCWT registers.

The operation during startup will be somewhat different depending on whether or not the PWMTM register is the first PWM register that is written. If the PWMTM register is written first, the PWM timer will begin incrementing from zero to the PWMTM value. This actually corresponds to the second half cycle (SYSSTAT bit 3 is set) of the PWM as can be seen from Figure 12. During this time, the PWM unit is in the **disable** state and all six PWM outputs are OFF. Provided the PWMCHA, PWMCHB and PWMCHC registers have all been written to by the time the PWM timer reaches the PWMTM value, the first internal load signal is generated when the PWM timer reaches the PWMTM value. At this point the PWM generation unit enters the **normal** state and the new values are latched into the PWMCHA, PWMCHB and PWMCHC registers. However, no PWMSYNC pulse is generated at this first load signal. The values written to the PWMCHA, PWMCHB and PWMCHC registers (in conjunction with the other PWM registers such as the PWMDT, PWMPD, PWMSEG, PWMGATE registers) define the PWM outputs during the next PWM cycle (in single update mode) or the next half cycle (in double update mode). The appearance of the first PWMSYNC pulse (and associated interrupt) will occur at the end of this first PWM cycle in single update mode when the PWM timer again reaches the PWMTM value. The first PWMSYNC pulse and interrupt will occur at the mid-point of the first PWM cycle in double update mode when the PWM timer reaches zero. Therefore, in single update mode the first PWMSYNC pulse will occur one and a half PWM periods or:

$$T_{1,SUM} = 1.5 \times 2 \times \text{PWMTM} \times \text{t}_{\text{CK}}$$

after the initial write to the PWMTM register. In double update mode the first PWMSYNC pulse appears one full PWM period or:

$$T_{1,DUM} = 1.0 \times 2 \times \text{PWMTM} \times \text{t}_{\text{CK}}$$

after the initial write to the PWMTM register.

The startup operation of the PWM unit is illustrated in Figure 13 for single update mode. The first PWM register that is written to is the PWMTM register that begins the incrementing of the timer from 0. It can be seen from Figure 13 that both the AH and AL outputs (as well as the BH, BL, CH and CL outputs) are in the OFF state during this initial half period. Assuming an initial value of 0 is written to the PWMCHA register (and some initial value is written to the other two duty cycle registers) prior to the PWM timer reaching the PWMTM value, the AH output stays in the OFF state and the AL output goes to the full ON state for the entire next PWM period. Only at the end of this cycle is the first PWMSYNC pulse generated. Therefore, if the values in the PWMTM and duty cycle registers have not changed, the AL and AH signals will remain in the same state for the next PWM period.

The startup operation of the PWM unit is illustrated in Figure 14 where again it is assumed that the PWMTM register is written first and begins the timer incrementing from 0. Again, the PWMCHA duty cycle register is written with an initial value of 0, so that when the PWM timer reaches the PWMTM value, the AH signal remains in the OFF state and the AL signal goes to fully ON for the next PWM half cycle. At the end of this half cycle, when the PWM timer reaches zero, the first PWMSYNC pulse appears. At this stage, assuming the PWM duty cycle registers have been updated, the PWM signals transition to the appropriate duty cycle outputs, as given by the new duty cycle register value and the contents of the dead time register, as illustrated in Figure 14. Again, it can be seen that because of the transition from full OFF to normal modulation at the first PWMSYNC pulse, an additional dead time is inserted into the AH output as dictated by the logic of Table 6.

If the PWMTM register is not the first PWM register written to but instead either PWMCHA, PWMCHB or PWMCHC is written first, then the operation is different to that described above. The act of writing to one of the duty cycle registers causes the PWM timer to begin incrementing prior to writing to the PWMTM register. The PWMTM register will thus contain its default reset value of 0. Therefore, the first load signal would not be generated until the PWM timer increments to 0 (corresponding to a counter overflow from 0xFFFF). Therefore, the PWMTM register will not be loaded with its initial

value until 65,535*t_{CK} (2.52 ms at 26 MHz CLKOUT) after the initial write to the first duty cycle register. Therefore, it is recommended that the PWM system be initialized by first writing to the PWMTM register and immediately following with initial writes to the three duty cycle registers and any other appropriate configuration registers.

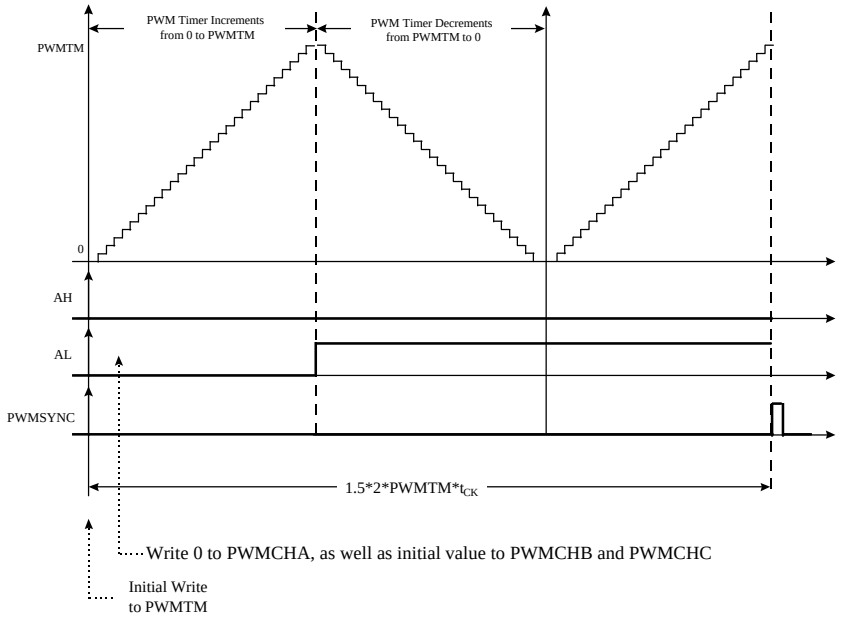


Figure 13: Startup of PWM Generation Unit in Single Update Mode (Active HI PWM).

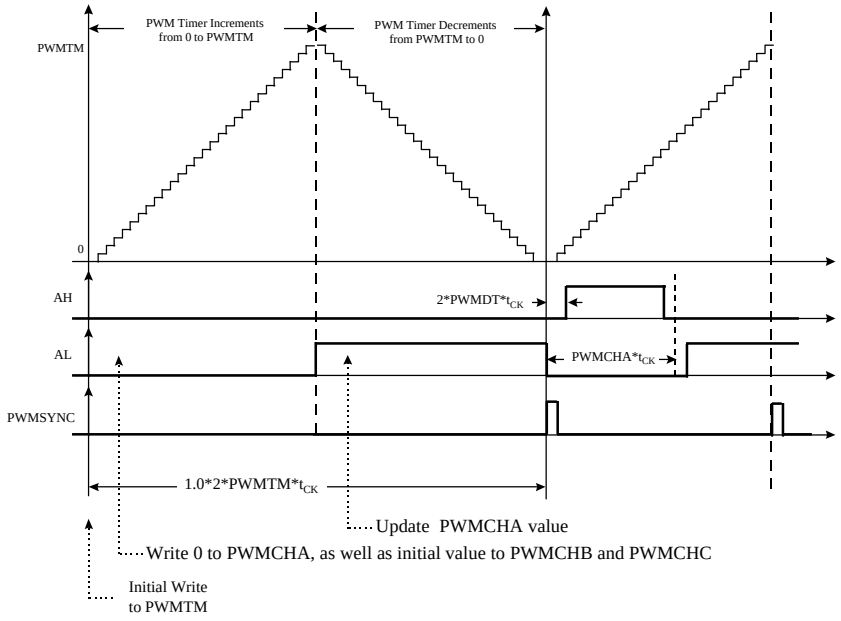


Figure 14: Startup of PWM Generation Unit in Double Update Mode (Active HI PWM).

6.2.10 Effective PWM Resolution

In single update mode, the same value of PWMCHA, PWMCHB and PWMCHC are used to define the on-times in both half cycles of the PWM period. As a result the effective resolution of the PWM generation process if $2t_{CK}$ (or 77 ns for a 26 MHz CLKOUT) since incrementing one of the duty cycle registers by 1 changes the resultant on-time of the associated PWM signals by t_{CK} in each half period (or $2t_{CK}$ for the full period).

In double update mode, improved resolution is possible since different values of the duty cycles registers are used to define the on-times in both the first and second halves of the PWM period. As a result, it is possible to adjust the on-time over the whole period in increments of t_{CK}. This corresponds to an effective PWM resolution of t_{CK} in double update mode (or 38.5 ns for a 26 MHz CLKOUT).

The achievable PWM switching frequency at a given PWM resolution is tabulated in Table 7.

Resolution (bits)	Single Update Mode PWM Frequency (kHz)	Double Update Mode PWM Frequency (kHz)
8	50.8	101.6
9	25.4	50.8
10	12.7	25.4
11	6.35	12.7
12	3.17	6.35

Table 7: Achievable PWM resolution in single and double update modes (CLKOUT = 26 MHz).

6.3 OUTPUT CONTROL UNIT, PWMSEG REGISTER

The operation of the Output Control Unit is controlled by the 9-bit read/write PWMSEG register that controls two distinct features that are directly useful in the control of ECM or BDCM.

6.3.1 Crossover Feature

The PWMSEG register contains three crossover bits; one for each pair of PWM outputs. Setting bit 8 of the PWMSEG register enables the crossover mode for the AH/AL pair of PWM signals, setting bit 7 enables crossover on the BH/BL pair of PWM signals and setting bit 6 enables crossover on the CH/CL pair of PWM signals. If crossover mode is enabled for any pair of PWM signals, the high-side PWM signal from the timing unit (AH say) is diverted to the associated low-side output of the Output Control Unit so that the signal will ultimately appear at the AL pin. Of course, the corresponding low-side output of the Timing Unit is also diverted to the complementary high-side output of the Output Control Unit so that the signal appears at the AH pin. Following a reset, the three crossover bits are cleared so that the crossover mode is disabled on all three pairs of PWM signals.

Care must be taken with the crossover feature to prevent potential shoot through conditions from developing in the power converter due to dead time violations. This situation is illustrated in Figure 15 where operation in double update mode is assumed with active LO PWM signals. The channel A duty cycle register is altered for the second half cycle and it is assumed that the crossover mode is enabled for the second half cycle. Figure 15 (a) illustrates the resultant PWM signals if the crossover mode is not enabled. Figure 15(b) shows the resultant signals if the AH and AL signals are simple swapped by setting the crossover bit for channel A prior to the start of the second half cycle. Clearly, this would cause both the AL and AH signals to switch simultaneously at the mid point of the PWM cycle. This corresponds to a simultaneous command to turn OFF the AH switch and turn ON the AL switch. This violates the required dead time criterion and can cause potentially destructive short-circuit conditions in the power inverter. As a result, the PWM unit of the ADMC401 detects this condition and inserts an additional dead time delay into the PWM signals as illustrated in Figure 15(c).

The PWM unit of the ADMC401 inserts this additional dead time into the appropriate signals if it detects a change of state in any of the crossover bits at successive PWMSYNC boundaries. In other words, if the PWM unit detects that the channel A crossover bit is set at one PWMSYNC boundary and cleared at the next one (or vica versa), it inserts a dead time into the PWM signal that is to be turned ON (either AL or AH). Of course, this may result in complete deletion of the PWM pulse if the duty cycle value is sufficiently small. Also, the PWM block monitors the channel B and channel C crossover bits and performs similar operations on the BH, BL and CH, CL signals, respectively. It is important to recognize that such an action is necessary only when the crossover bit is toggled. If the crossover bit remains set or cleared for many PWM cycles, the added dead time is not necessary as all dead time insertions are taken care of by the normal operation of the three-phase timing unit.

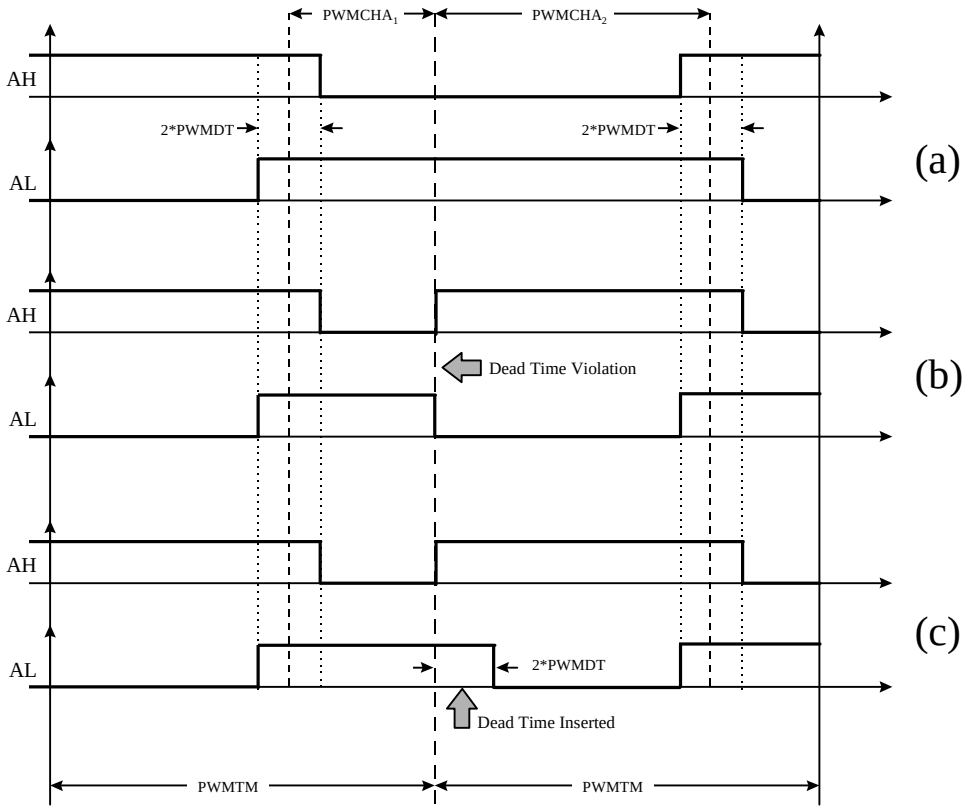


Figure 15: Insertion of Additional Dead Time due to Crossover Feature of PWM. (a) Initial programmed PWM signals without crossover featured enabled (double update mode, active LO). (b) Same PWM signals but with crossover mode enabled in second half cycle resulting in potential dead time violation at middle of PWM cycle (c) Resultant PWM output signals of ADMC401 with added dead time to prevent potential problem outlined in (b).

6.3.2 Output Enable Function

The PWMSEG register also contains six bits (bits 0 to 5) that can be used to individually enable or disable each of the six PWM outputs. The PWM signal of the AL pin is enabled by setting bit 5 of the PWMSEG register while bit 4 controls AH, bit 3 controls BL, bit 2 controls BH, bit 1 controls CL and bit 0 controls the CH output. If the associated bit of the PWMSEG register is set, then the corresponding PWM output is disabled irrespective of the value of the corresponding duty cycle register. This PWM output signal will remain in the OFF state as long as the corresponding enable/disable bit of

the PWMSEG register is set. This output enable function is implemented after the crossover function. Following a reset, all six enable bits of the PWMSEG register are cleared so that all PWM outputs are enabled by default. In a manner identical to the duty cycle registers, the PWMSEG is latched on the rising edge of the PWMSYNC signal so that changes to this register only become effective at the start of each PWM cycle in single update mode. In double update mode, the PWMSEG register can also be updated at the mid-point of the PWM cycle.

6.3.3 Brushless DC Motor (Electronically Commutated Motor) Control

In the control of an ECM only two inverter legs are switched at any time and often the high-side device in one leg must be switched ON at the same time as the low-side driver in a second leg. Therefore, by programming identical duty cycles values for two PWM channels (say PWMCHA = PWMCHB) and setting bit 7 of the PWMSEG register to crossover the BH/BL pair if PWM signals, it is possible to turn ON the high-side switch of phase A and the low-side switch of phase B at the same time. In the control of ECM, it is usual that the third inverter leg (phase C in this example) be disabled for a number of PWM cycles. This function is implemented by disabling both the CH and CL PWM outputs by setting bits 0 and 1 of the PWMSEG register. This situation is illustrated in Figure 16 where it can be seen that both the AH and BL signals are identical, since PWMCHA=PWMCHB and the crossover bit for phase B is set. In addition, the other four signals (AL, BH, CH and CL) have been disabled by setting the appropriate enable/disable bits of the PWMSEG register. For the situation illustrated in Figure 16, the appropriate value for the PWMSEG register is 0x00A7. In normal ECM operation, each inverter leg is disabled for certain periods of time, so that the PWMSEG register is changed based on the position of the rotor shaft (motor commutation).

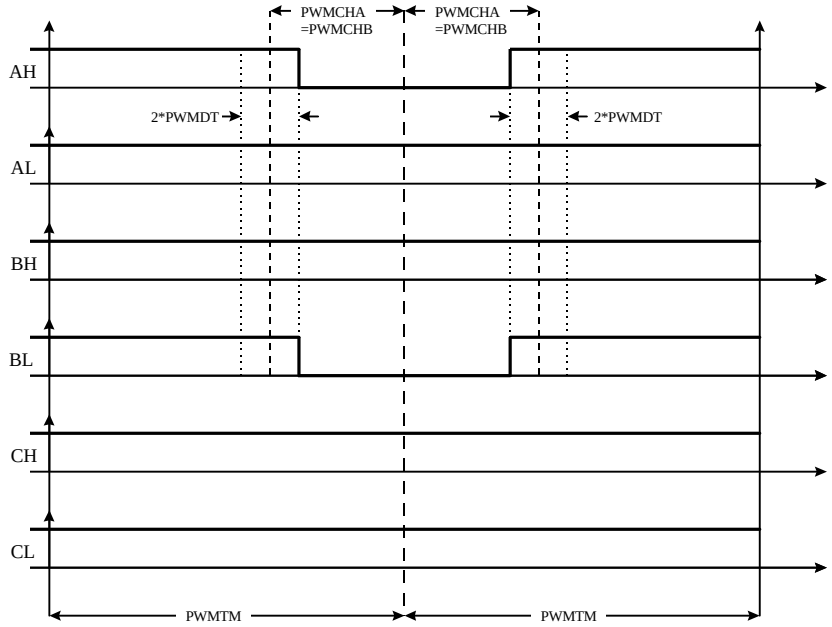


Figure 16: Example active LO PWM signals suitable for ECM control, PWMCHA=PWMCHB, crossover BH/BL pair and disable AL, BH, CH and CL outputs. Operation is in single update mode.

6.4 GATE DRIVE UNIT, PWMGATE REGISTER

6.4.1 High Frequency Chopping

The Gate Drive Unit of the PWM controller adds features which simplify the design of isolated gate drive circuits for PWM inverters. If a transformer coupled power device gate drive amplifier is used then the active PWM signal must be chopped at a high frequency. The 10-bit read/write PWMGATE register allows the programming of this high frequency chopping mode. The chopped active PWM signals may be required for the high-side drivers only, for the low-side drivers only or for both the high-side and low-side switches. Therefore, independent control of this mode for both high and low-side switches is included with two separate control bits in the PWMGATE register.

Typical PWM output signals with high-frequency chopping enabled on both high-side and low-side signals are shown in Figure 17. Chopping of the high-side PWM outputs (AH, BH and CH) is enabled by setting bit 8 of the PWMGATE register. Chopping of the low-side PWM outputs (AL, BL and CL) is enabled by setting bit 9 of the PWMGATE register. The high frequency chopping frequency is controlled by the 8-bit word (GDCLK) placed in bits 0 to 7 of the PWMGATE register. The period of this high frequency carrier is:

$$T_{chop} = [4 \times (GDCLK + 1)] \times t_{CK}$$

and the chopping frequency is therefore an integral subdivision of the CLKOUT frequency:

$$f_{chop} = \frac{f_{CLKOUT}}{[4 \times (GDCLK + 1)]}$$

The GDCLK value may range from 0 to 255, corresponding to a programmable chopping frequency rate from 25.4 kHz to 86.5 MHz for a 26 MHz CLKOUT rate. The gate drive features must be programmed before operation of the PWM controller and typically are not changed during normal operation of the PWM controller. Following a reset, all bits of the PWMGATE register are cleared so that high frequency chopping is disabled, by default.

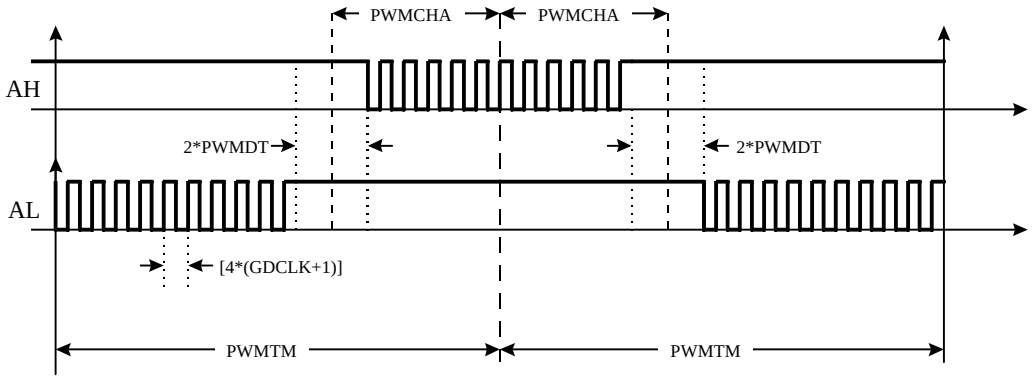


Figure 17: Typical active LO PWM signals with high-frequency gate chopping enabled on both high-side and low-side switches (GDCLK is integer equivalent of value in bits 0 to 7 of PWMGATE register.

6.4.1 PWM Polarity Control, PWMPOL Pin

The polarity of the PWM signals produced at the output pins AH to CL may be selected in hardware by the PWMPOL pin. Connecting the PWMPOL pin to DGND selects active LO PWM outputs, such that a LO level is interpreted as a command

to turn on the associated power device. Conversely, connecting V_{DD} to PWMPOL pin selects active HI PWM and the associated power devices are turned ON by a HI level at the PWM outputs. There is an internal pull-up on the PWMPOL pin, so that if this pin becomes disconnected (or is not connected), active HI PWM will be produced. The level on the PWMPOL pin may be read from bit 2 of the SYSSTAT register, where a zero indicated a measured LO level at the PWMPOL pin.

6.5 SWITCHED RELUCTANCE MODE

The PWM block of the ADMC401 contains a switched reluctance mode that is controlled by the state of the $\overline{\text{PWMSR}}$ pin. The switched reluctance (SR) mode is enabled by connecting the $\overline{\text{PWMSR}}$ pin to DGND. In this SR mode, the low-side PWM signals from the three-phase timing unit assume permanently ON states, independent of the value written to the duty-cycle registers. The duty cycles of the high-side PWM signals from the timing unit are still determined by the three duty cycle registers. Using the cross-over feature of the output control unit, it is possible to divert the permanently ON PWM signals to either the high-side or low-side outputs. This mode is necessary because in the typical power converter configuration for switched or variable reluctance motors, the motor winding is connected between the two power switches of a given inverter leg. Therefore, in order to build up current in the motor winding, it is necessary to turn on both switches at the same time. Typical active LO PWM signals during operation in SR mode are shown in Figure 18 for operation in double update mode. It is clear that the three low-side signals (AL, BL & CL) are permanently ON and the three-high-side signals are modulated in the usual manner so that the corresponding high-side power switches are switched between the ON and OFF states. The SR mode can **only** be enabled by connecting the $\overline{\text{PWMSR}}$ pin to GND. There is no software means by which this mode can be enabled. There is an internal pull-up resistor on the $\overline{\text{PWMSR}}$ pin so that if this pin is left unconnected or becomes disconnected the SR mode is disabled. Of course, the SR mode is disabled when the $\overline{\text{PWMSR}}$ pin is tied to V_{DD}. The state of this switched reluctance mode may be read from bit 4 of the SYSSTAT register. If the $\overline{\text{PWMSR}}$ pin is HI (such that the SR mode is disabled) bit 4 of the SYSSTAT register is cleared (indicating that the mode is disabled). Conversely, if the $\overline{\text{PWMSR}}$ pin is LO and SR mode is enabled, bit 4 of SYSSTAT is set.

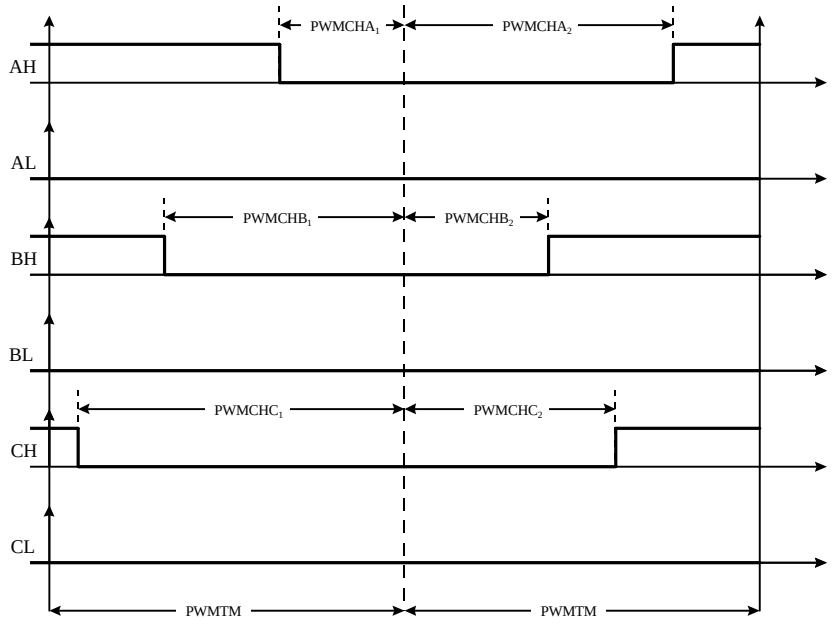


Figure 18: Active LO PWM signals in SR Mode (PWMPOL = $\overline{\text{PWMSR}}$ = DGND) for ADMC401 in double update mode. The signals from the three-phase timing unit are not crossed over (PWMSEG = 0) and the dead time is zero (PWMDT = 0).

6.6 PWM SHUTDOWN

In the event of external fault conditions, it is essential that the PWM system be instantaneously shutdown in a safe fashion. A falling edge on the $\overline{\text{PWMTRIP}}$ pin provides an instantaneous, asynchronous (independent of the DSP clock) shutdown of the PWM controller. All six PWM outputs are placed in the OFF state (as defined by the PWMPOL pin). In addition, the PWMSYNC pulse is disabled and the associated interrupt is stopped. The $\overline{\text{PWMTRIP}}$ pin has an internal pull-down resistor so that if the pin becomes unconnected the PWM will be disabled. The state of the $\overline{\text{PWMTRIP}}$ pin can be read from bit 0 of the SYSSTAT register.

The 12 PIO lines of the ADMC401 can also be configured to operate as PWM shutdown pins using the PIOPWM register. The 12-bit PIOPWM has a control bit for each PIO line (bit 0 control PIO0 etc.). Setting the control bit enables the corresponding PIO line as a PWM shutdown pin. A falling edge on the PIO line will then generate an instantaneous, asynchronous shutdown of the PWM system, in a manner identical to the $\overline{\text{PWMTRIP}}$ pin. On power-up and following a reset, all PIO lines are configured as inputs, have pull-downs and are programmed as PWM shutdown pins (PIOPWM = 0x0FFF) so that the PWM is shutdown. Correct operation of the PWM is not possible without first correctly configuring the PIO system.

In addition, it is possible to initiate a PWM shutdown in software by writing to the 1-bit read/write PWMSWT register. The act of writing to this register generates a PWM shutdown command in a manner identical to the $\overline{\text{PWMTRIP}}$ or PIO pins. It does not matter which value is written to the PWMSWT register. However, following a PWM shutdown, it is possible to read the PWMSWT register to determine if the shutdown was generated by hardware or software. Reading the PWMSWT register automatically clears its contents.
On the occurrence of a PWM shutdown command (either from the $\overline{\text{PWMTRIP}}$ pin, the PIO lines or the PWMSWT register), a PWMTRIP interrupt will be generated. In addition, the PWMSYNC pulse no longer appears at the output pin. However, internal operation of the PWM timer continues. Following a PWM shutdown, the PWM can only be re-enabled (in a PWMTRIP interrupt service routine, for example) by writing to all of the PWMTM, PWMCHA, PWMCHB and PWMCHC registers. Provided, the external fault has been cleared and the $\overline{\text{PWMTRIP}}$ or appropriate PIO lines have returned to a HI level, the PWM controller will restart in a manner identical to that following a power up, as described in Section 6.2.7.

6.7 PWM REGISTERS

The PWM registers are described in Figure 19.

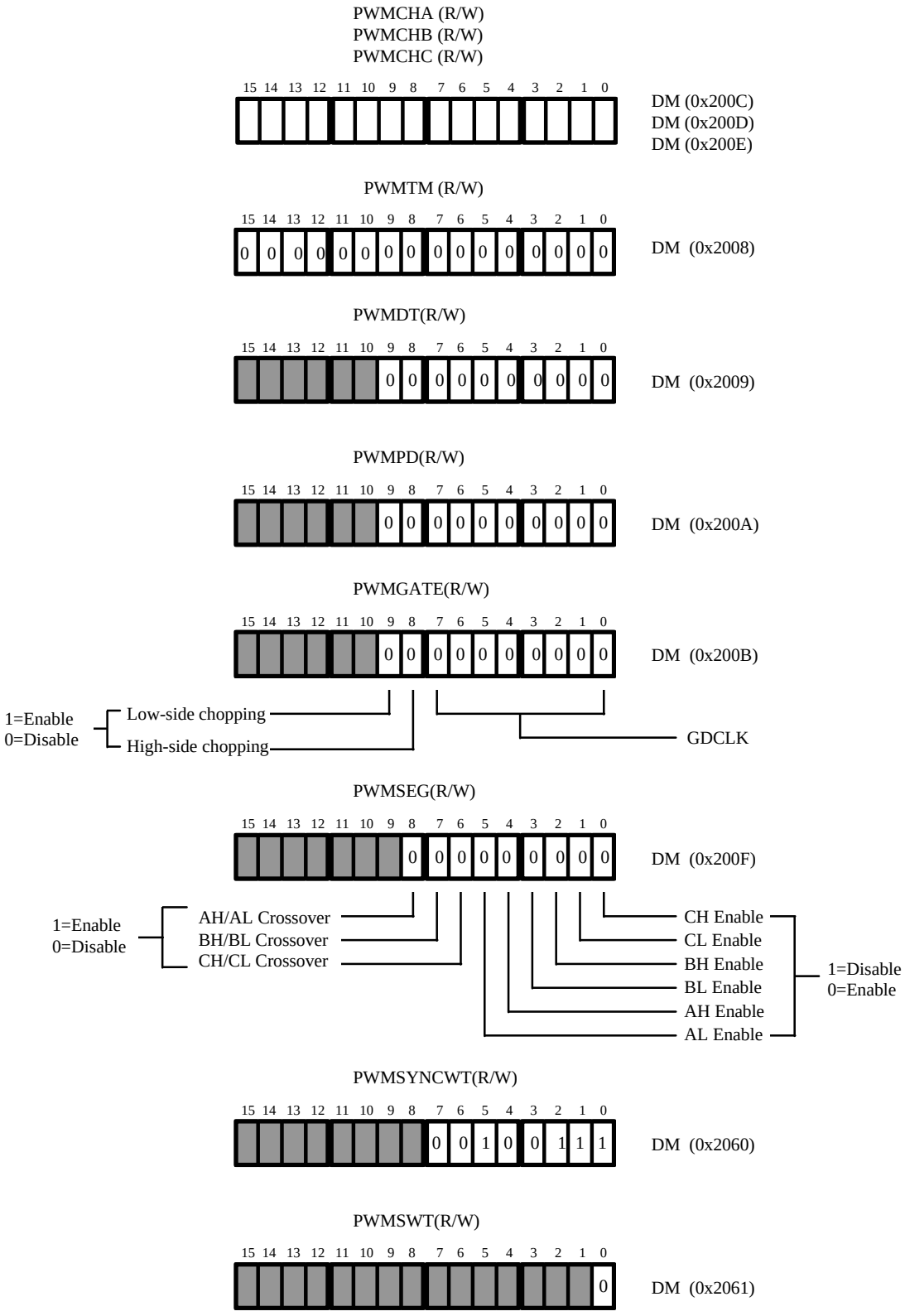


Figure 19: Structure of PWM Registers of ADMC401 (shaded bits are unused, reset values included, where defined).

7. ENCODER INTERFACE UNIT.

7.1 OVERVIEW

The ADMC401 incorporates a powerful encoder interface block to incremental shaft encoders, that are often used for position feedback in high performance motion control systems. The functional block diagram of the entire encoder interface system of the ADMC401 is shown in Figure 20. The encoder interface unit (EIU) includes a 16-bit quadrature up/down counter, programmable input noise filtering of the encoder input signals and the zero markers, and has four dedicated pins on the ADMC401. The quadrature encoder signals are applied at the EIA and EIB pins. Alternatively, a frequency and direction set of inputs may be applied to the EIA and EIB pins. In addition, two zero marker/strobe inputs are provided on pins EIZP and EIS. These inputs may be used to latch the contents of the encoder quadrature counter into dedicated registers, EIZPLATCH and EISLATCH, on the occurrence of external events at the EIZP and EIS pins. These events may be programmed to be either rising edge only (latch event) or rising edge if the encoder is moving in the forward direction and falling edge if the encoder is moving in the reverse direction (software latched zero marker functionality). The encoder interface unit incorporates programmable noise filtering on the four encoder inputs to prevent spurious noise pulses from adversely affecting the operation of the quadrature counter. The encoder interface unit operates at a clock frequency equal to half of the DSP instruction rate, CLKOUT, so that the fundamental time increment is $2*t_{CK}$ (or 77 ns for a 26 MHz CLKOUT). The encoder interface unit operates correctly with encoder signals at frequencies of up to 3.25 MHz.

The EIU may be programmed to use the zero marker on EIZP to reset the quadrature encoder in hardware, if required. Alternatively, the zero marker can be ignored and the encoder quadrature counter is reset according to the contents of a maximum count register, EIUMAXCNT. There is also a “single north marker” mode available in which the encoder quadrature counter is reset only on the first zero marker pulse. Both modes are enabled by dedicated control bits in the EIU control register, EIUCTRL. A status bit is set in the EIUSTAT register on the first occurrence of the zero marker.

The encoder interface unit can also be made to implement some error checking functions. If the error checking mode is enabled, upon the occurrence of a zero pulse, the contents of the encoder counter register are compared with the expected value (0 or EIUMAXCNT depending on the direction of rotation). If an encoder count error is detected (say due to a disconnected encoder line), a status bit in the EIUSTAT register is set and an EIU count error interrupt is generated. An additional status bit is provided in the EIUSTAT register that indicates the initialization state of the EIU. Until the EIUMAXCNT register is written to, the EIU is not initialized. Three status bit in the EIUSTAT register can also be read to read the state of the three EIU pins, EIA, EIB and EIZP.

The Encoder Interface Unit of the ADMC401 contains a 16-bit loop timer that behaves in a manner similar to the programmable interval timer of the DSP core. The loop timer consist of a timer register, period register and scale register so that it can be programmed to time-out and reload at appropriate intervals. A control bit in the EIUCTRL register is used to enable/disable this loop timer. When this loop timer times out, an EIU loop timer timeout interrupt is generated. This interrupt could be used to control the timing of speed and position control loops in high-performance drives.

The encoder interface unit also includes a high-performance encoder event timer (EET) block that permits the accurate timing of successive events of the encoder inputs. The EET can be programmed to time the duration between up to 255 encoder pulses and can be used to enhance velocity estimation, particularly at low speeds of rotation. The information from the registers of the EET block can be latched in two ways. In one mode, the contents of the EIU quadrature count register, EIUCNT and all relevant EET registers (EETT and EETDELTAT) are latched when the EIU timer times-out. In the second mode, the act of reading the EIUCNT register also simultaneously latches the EET registers. The EET data latching mode is selected by a control bit in the EIUCTRL register.

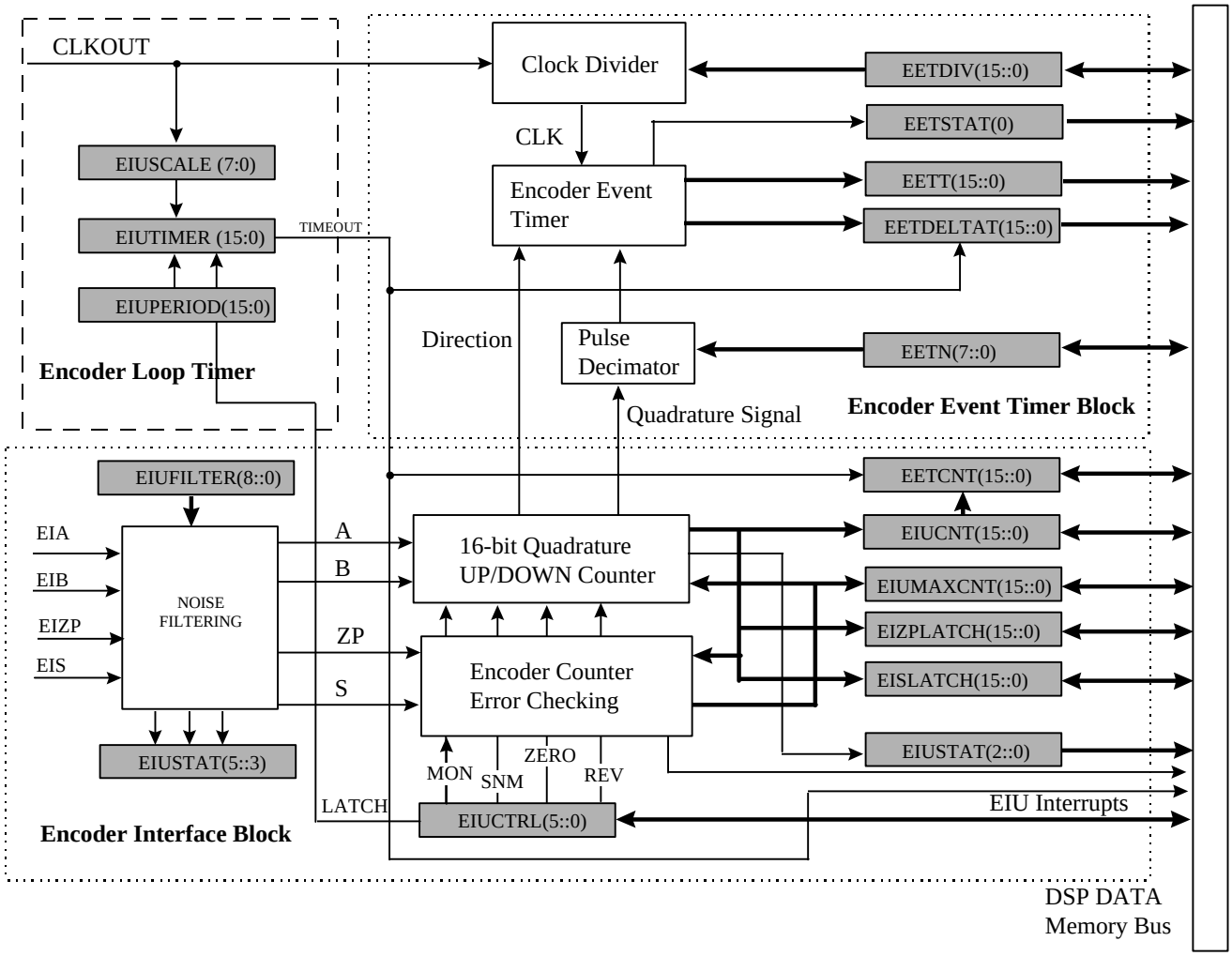


Figure 20: Configuration of Encoder Interface System of ADMC401**.**

7.2 ENCODER LOOP TIMER

The EIU contains a 16-bit loop timer that is structured in a manner similar to the interval timer of the DSP core (TCOUNT, TPERIOD and TSCALE registers). The corresponding registers of the encoder loop timer are the 16-bit read/write EIUTIMER and EIUPERIOD and the 8-bit read/write EIUSCALE register. The EIU loop timer is clocked at the CLKIN rate, t_{CKI}.

The EIU loop timer can be used to generate periodic interrupts based on multiples of the CLKIN cycle times. The EIU loop timer is enabled by setting bit 5 of the EIUCTRL register. When enabled, the 16-bit timer register (EIUTIMER) is decremented every N cycles, where N-1 is the scaling value stored in the 8-bit EIUSCALE register. When the value of the EIUTIMER register reaches zero and EIU loop timer time-out interrupt is generated and the EIUTIMER register is reloaded with the 16-bit value in the EIUPERIOD register. The scaling feature of this timer, provided by the EIUSCALE register, allows the 16-bit timer to generate periodic interrupts over a wide range of periods. For a 26 MHz CLKOUT rate (38.5 ns period), the timer can generate interrupts with periods of 77 ns up to 5.04 ms with a zero scale value (EIUSCALE = 0). When scaling is used, time periods can range up to 1.29 s. The EIU loop timer time-out interrupt can be masked in the PICMASK register.

7.3 ENCODER INTERFACE STRUCTURE & OPERATION

7.3.1 Introduction

The encoder interface section consists of 16-bit quadrature up/down counter, a 16-bit read/write EIUCNT register that allows the up/down counter to be read by the DSP. There is also a 16-bit read/write EIUMAXCNT register that must be written to initialize the encoder system. Until the EIUMAXCNT register has been written to, the encoder interface unit is not initialized and bit 2 of the EIUSTAT register is set. The contents of the EIUMAXCNT register are used in certain operating modes to reset the quadrature counter. The contents of the EIUMAXCNT register are also used for error checking of the EIU. Operation of the encoder interface is controlled by the read/write EIUCTRL register.

7.3.2 Input Noise Filtering of Encoder Signals

A functional block diagram of the input stages of the encoder interface is shown in Figure 21. The four encoder input signals (EIA, EIB, EIZP & EIS) are first synchronized to the CLKOUT rate in input synchronization buffers. This eliminates the asynchronous nature of real world encoder signals prior to use in the encoder interface unit logic. Subsequently, all three synchronized signals (EIAS, EIBS, EIZPS & EISS) are applied to programmable noise filtering circuits that can be programmed to reject pulses that are shorter than some suitable programmed value. The outputs of the filter stage are applied to the quadrature counter stage.

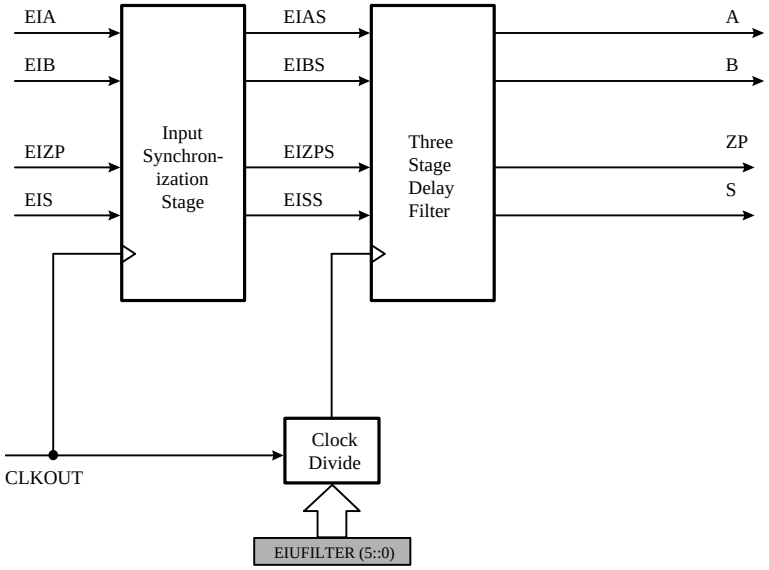


Figure 21: Functional Block Diagram of Inputs Stage (Synchronization and Noise Filtering of Encoder Interface).

Each of the four synchronized input signals (EIAS, EIBS, EIZPS and EISS) is applied to a three clock cycle delay filter such that the filtered output signals are not permitted to change until a stable value has been registered for three successive clock cycles. While the encoder signals are changing, the filter maintains the previous output value. The clock frequency used for the filter circuits is programmed by bits 0 to 5 of the EIUFILTER register. The 6-bit quantity written to bits 0 to 5 of the EIUFILTER register is used to divide the CLKOUT frequency and provide the clock source for the encoder noise filters. If the value written to bits 0 to 5 of the EIUFILTER register is N, the period of the clock source used in the encoder filters is $(N+1)*t_{CK}$. Therefore, the minimum pulse width that can be accepted on the encoder signals is three of these clock periods or:

$$T_{minenc} = 3 \times (N+1) \times t_{CK}$$

For example, writing a value of 3 to bits 0 to 5 of the EIUFILTER register, means that clock frequency used in the encoder filters is 6.5 MHz (for a CLKOUT rate of 26 MHz). In order to register as a stable value, the encoder input signals must be stable for three of these 6.5 MHz cycles (or 462 ns). Consequently, the smallest period that will be registered on the synchronized encoder inputs is 924 ns, corresponding to a maximum encoder rate of 1.08 MHz. In general, the maximum encoder rate that can be recognized is given by:

$$f_{ENCMAX} = \frac{f_{CLKOUT}}{6 \times (N+1)}$$

Operation of both the input synchronization logic and the noise filters is shown in Figure 22 for the default case where EIUFILTER(5::0) = 0x00 and the noise filters are clocked at CLKOUT.

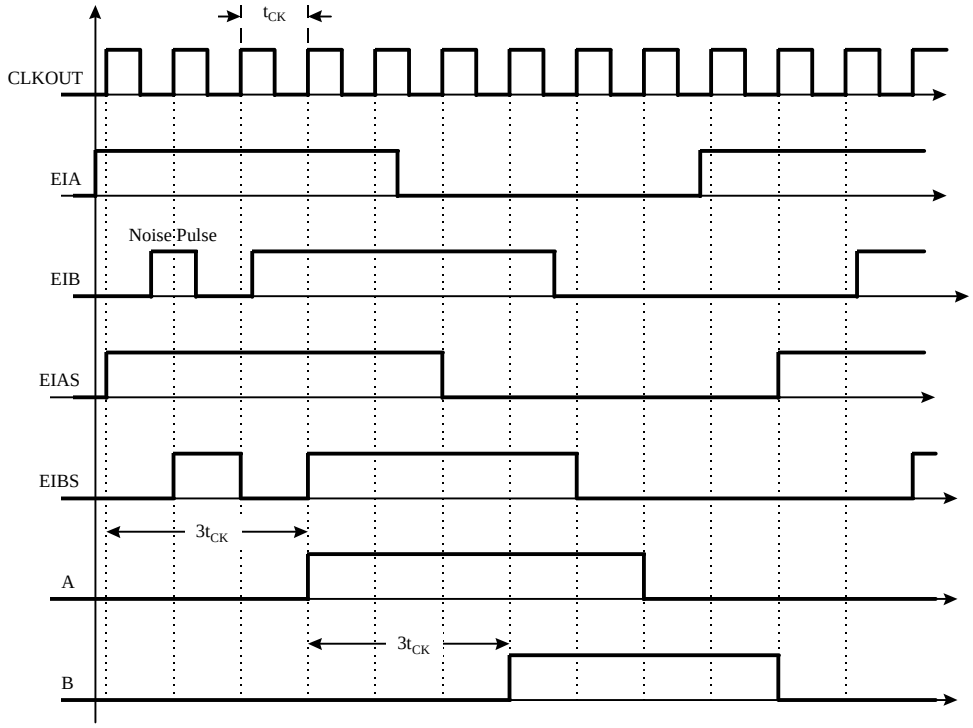


Figure 22: Operation of input synchronization and noise filters of encoder interface with EIUFILTER(5:0) = 0x00 such that the filters are operated at CLKOUT.

The default value for EIUFILTER(5::0) following a power on reset is 0x00 so that the EIU filters are clocked at the CLKOUT rate and minimal filtering is applied. There is a direct trade-off between the amount of filtering applied to the encoder inputs and the maximum possible encoder signal rate. In effect, the larger the value of EIUFILTER(5::0), the more filtering that is applied to the encoder signals so that, for a given number of encoder lines, the maximum speed of rotation is lower. The maximum encoder signal rate and minimum permissible encoder pulse width are tabulated in Table 8 for different values of EIUFILTER(5::0).

EIUFILTER (5::0)	Encoder Filter Clock Frequency (MHz)	Minimum Pulse Width (ns)	Maximum Encoder Rate (MHz)
0 (default)	26	115.5	4.3
1	13	231	2.1
2	8.667	346.5	1.44
3	6.5	462	1.08
4	5.2	577.5	0.865
:	:	:	:
63	0.406	7392	0.067

Table 8: Minimum permissible encoder pulse width and corresponding maximum encoder signal rate for different value of EIUFILTER(5:0) assuming a CLKOUT rate of 26 MHz.

The influence of the encoder filter can be on the zero marker signals (EIZPS and EISS) can be somewhat different that on the EIAS or EIBS signals, depending on the exact nature of the encoder. In common incremental encoders, the width of the zero marker can be equal to a quarter, a half or a full period of one of the quadrature signals (say EIA). Applying the three-stage delay filter to a zero marker whose width is either equal to half or a full quadrature pulse period does not change the achievable maximum encoder rate. However, the maximum possible encoder rate is changed if the three-stage filter is applied in the case where the width of the zero marker is equal to a quarter of the EIA or EIB period. In this case the influence of the three-stage delay filter is to effectively half the maximum encoder signal rate to that described above (or 1.625 MHz for a 26 MHz CLKOUT rate).

7.3.3 Encoder Quadrature Counter Operation

Following the input synchronization stages the A and B encoder signals (filtered outputs) go to a state machine that determines the appropriate actions for the quadrature counter, EIUCNT. Essentially, the state machine consists of dual flip-flops on each of the A and B inputs so that the logic can detect change of state events at each input, as illustrated in Figure 23. The action demanded by the state machine depends on the present state of both encoder inputs (NEW_A and NEW_B) as well as the state of both inputs in the previous clock cycle (LAST_A and LAST_B). Four possible outcomes are possible from the state machine:

- COUNT UP: The EIU Quadrature Counter is Incremented by 1.
- COUNT DOWN: The EIU Quadrature Counter is Decremented by 1.
- NO COUNT: No change of state event has been detected and no action is taken at the Quadrature Counter.
- ERROR: This is an illegal state (possibly occurs if the encoder signals exceed the maximum data rate). If this event is detected, the NO COUNT action is performed.

The actions taken by the Encoder Quadrature counter are summarized in Table 9.

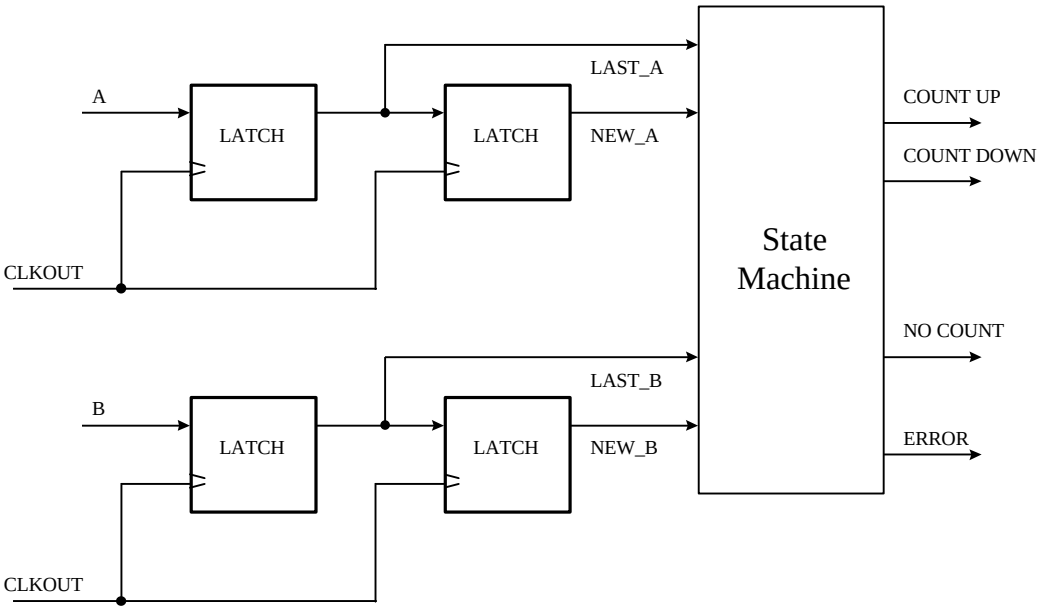


Figure 23: Structure of EIU Input State Machine.

NEW_A	LAST_A	NEW_B	LAST_B	ACTION
0	0	0	0	NO COUNT
0	0	0	1	COUNT UP
0	0	1	0	COUNT DOWN
0	0	1	1	NO COUNT
0	1	0	0	COUNT UP
0	1	0	1	ERROR
0	1	1	0	ERROR
0	1	1	1	COUNT DOWN
1	0	0	0	COUNT UP
1	0	0	1	ERROR
1	0	1	0	ERROR
1	0	1	1	COUNT DOWN
1	1	0	0	NO COUNT
1	1	0	1	COUNT DOWN
1	1	1	0	COUNT UP
1	1	1	1	NO COUNT

Table 9: Summary of Appropriate Quadrature Counter Operations.

7.3.4 Encoder Counter Direction

The direction of quadrature counting is determined by bit 0 (REV) of the EIUCTRL register. If the REV bit is cleared the signal at the EIA pin is fed to the A input to the quadrature counter and the EIB pin is fed to the B input. Thus, if the EIA-

encoder signal leads the EIB-signal (and therefore the A signal leads the B signal), the quadrature counter is incremented on each edge, as shown in Figure 24. This (A signal leads the B signal) is defined as the forward direction of motion. Setting bit 0 of the EIUCTRL register causes the signal at the EIA pin to be fed to the B input to the quadrature counter and the signal EIB becomes the B input to the quadrature counter. Therefore, if the EIA signal led the EIB signal at the pins of the ADMC401, the A input to the quadrature counter will now lag the B input. This will be recognized as rotation in the reverse direction and the counter will be decremented on each quadrature pulse. Following a reset, the REV bit is cleared.

As shown in Figure 20, the two encoder signals are used to derive a quadrature signal that is used, in conjunction with a direction bit, to increment or decrement the encoder counter and also the encoder event timer. The derivation of these signals results from the output of the state machine described in Table 9. The status of the direction signal is indicated at bit 1 of the EIUSTAT register. While the encoder counter is incrementing, bit 1 is set. Alternatively, when the encoder counter is decrementing, bit 1 of the EIUSTAT register is cleared.

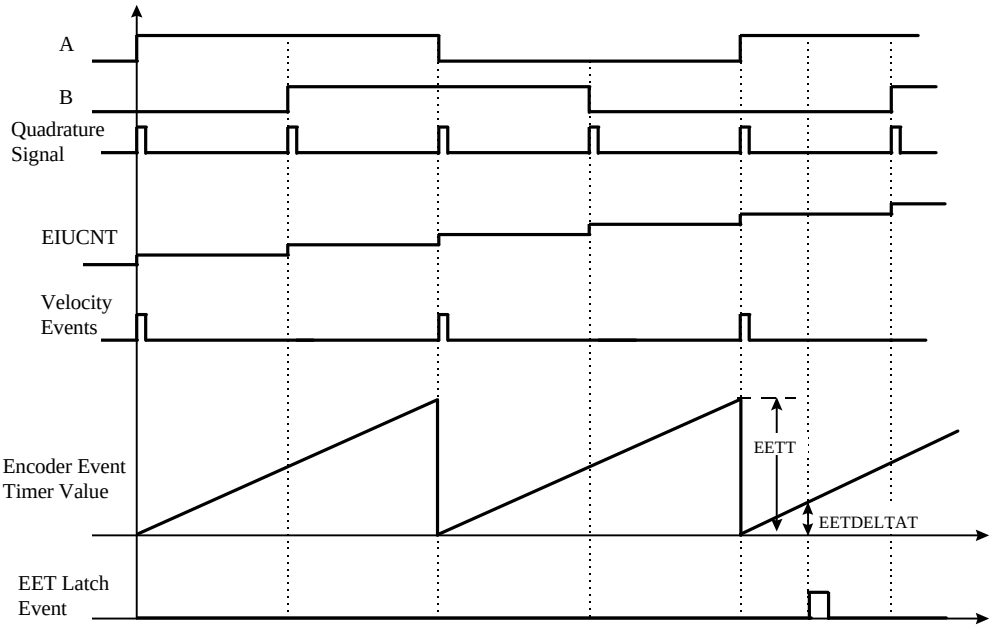


Figure 24: Operation of encoder interface unit and EET of ADMC401 in the forward direction with EETN = 2.

7.3.5 Alternative Frequency and Direction Inputs

Instead of the quadrature EIA and EIB encoder inputs, the encoder interface unit can also accept alternative Frequency and Direction Inputs. This mode is enabled by setting bit 6 of the EIUCTRL register. In this so-called *FD Mode*, the EIA input pin accepts a frequency signal and the EIB pin accepts the direction signal. The signal on these pins are subject to the same synchronization and filtering logic as described previously. However, in this mode, the operation of the state machine is bypassed and the quadrature counter is incremented or decremented on each rising edge of the signal on the EIA pin. If the EIB pin is HI, forward operation is assumed and the counter is incremented on each rising edge of the frequency signal on the EIA input. On the other hand, if the EIB pin is LO, reverse rotation is assumed and the quadrature counter is decremented by one at each rising edge of the signal on the EIA pin. On power-up or reset, bit 6 of the EIUCTRL register is cleared so that this mode is disabled by default.

7.3.6 Encoder Counter Reset

The ZERO bit (bit 1) of the EIUCTRL register determines if the encoder zero marker is used to hardware reset the up/down counter of the encoder interface. When bit 1 of the EIUCTRL register is set, the zero marker signal on EIZP is used to reset the up/down counter to zero (if moving in the forward direction) or to the value in the EIUMAXCNT register (if moving in the reverse direction). The reset operation takes place on the next quadrature pulse after the zero marker has been recognized. In order to ensure correct encoder counting (no missing or spurious codes) the logic in the encoder counter latches the conditions (appropriate encoder edge) at which the first reset is performed. Thereafter, irrespective of operating conditions the encoder reset operation is always aligned with the same encoder edge. For example, if the first reset operation occurs on the rising edge of B and the encoder is moving in the forward direction, then all subsequent reset operations are aligned with the rising edge of the B signal (while moving in the forward direction) and on the falling edge of B for rotation in the reverse direction. In order to account for zero marker signals of different widths, the zero marker will be recognized as the rising edge of the EIZP signal when moving in the forward direction. When moving in the reverse direction, the zero marker is recognized at the falling edge of the signal at the EIZP pin.

When the ZERO bit of the EIUCTRL register is cleared, the zero marker is not used to reset the counter. In this mode, the contents of the EIUMAXCNT register are used as the reset value for the up/down counter. For example, for an N-line incremental encoder, the appropriate value to write to the EIUMAXCNT register is 4N-1. Therefore, for a 1024 line encoder, a value of 0x0FFF (= 4095) would be written to the EIUMAXCNT register. However, since absolute position information is not available in this mode, due to the absence of the zero marker, the full 16-bit range of the quadrature counter may be employed by writing a value of 0xFFFF to the EIUMAXCNT register. Following a reset, the ZERO bit is cleared. **The value written to the EIUMAXCNT register must be in the form 4N - 1, where N is any integer**.

7.3.7 Latch/Freeze/Software Zero Marker Inputs

The encoder interface unit of the ADMC401 provides two marker signals, EIZP and EIS that are both filtered and synchronized in a manner identical to the other encoder signals to produce the ZP and S signals. ZP can be used as a hardware reset of the encoder counter. However, in many application a hardware reset of the counter may not be desirable because of disastrous effects that could occur due to incorrect resetting of the counter. Instead, the encoder counter can be programmed to operate in full 16-bit rollover mode, by clearing bit 1 of the EIUCTRL register and programming EIUMAXCNT to be 0xFFFF. In this case, the quadrature counter will use the full 16-bit range of the EIUCNT register.

The signals on ZP and S can be configured to latch the contents of the EIUCNT register into dedicated memory mapped registers (EIZPLATCH for ZP and EISLATCH for S) on the occurrence of definite events on these pins. The exact nature of the events are determined by bit 7 of the EIUCTRL register for ZP and bit 8 of the EIUCTRL register for S. If bit 7 of the EIUCTRL register is cleared, the contents of the EIUCNT register are latched to the EIZPLATCH register on the occurrence of a rising edge on the ZP signal. In this mode, the signals can be used to latch or freeze the EIUCNT contents on the occurrence of an external event such as that from limit switches or other triggers.

If bit 7 of the EIUCTRL register is set, then the EIUCNT contents are latched to the EIZPLATCH register on the rising edge of the ZP signal if the quadrature counter is incrementing (count up). If the quadrature counter is decrementing the EIUCNT contents are latched to the EIZPLATCH register on the falling edge of the ZP signal. In this mode, the action resembles that of a zero marker function and the encoder index pulse could be applied to EIZP. The advantage is that the EIUCNT register contents are latched at the appropriate zero marker inputs but the contents of the quadrature counter are no affected. Bit 8 of the EIUCTRL register defines the S events that cause the EIUCNT register to be latched to the EISLATCH register in an identical manner to that described for bit 7 above. EIZPLATCH and EISLATCH are 16-bit read only registers whose state is undefined on power up. On power-up or following a reset, both bit 7 and 8 of the EIUCTRL register are cleared so that freeze mode is enabled.

7.3.8 Single North Marker Mode

A further reset mode is available in the encoder interface unit called *Single North Marker Mode*. This mode is enabled by setting bit 2 (SNM) of the EIUCTRL register. For this mode to operate the ZERO bit (bit1) of the EIUCTRL register must also be set In this mode, the EIUCNT register is reset (to zero or EIUMAXCNT depending on direction) **only on the first occurrence** of the zero marker. Subsequently, the EIUCNT register is reset by the natural roll-over to zero or the value in the EIUMAXCNT register. Following a reset, this SNM bit is cleared. Bit 6 of the EIUSTAT register is used to signal the first occurrence of a zero marker. When the first zero marker has been recognized by the EIU, bit 6 of the EIUSTAT register is set.

7.3.9 Encoder Error Checking

Error checking in the EIU is enabled by setting bit 3 (MON) of the EIUCTRL register. The ZERO bit of the EIUCTRL register **must also be set** for error checking to be enabled. In this mode, the contents of the EIUCNT register are compared with the expected value (zero or EIUMAXCNT depending on direction) when the zero marker is detected. If a value other than the expected value is detected, an error condition is generated by setting bit 0 of the EIUSTAT register and triggering an EIU count error interrupt. This EIU count error interrupt is also managed and may be masked by the programmable interrupt controller (PIC) block. The encoder continues to count encoder edges after an error has been detected. Bit 0 of the EIUSTAT register is cleared on the occurrence of the next zero marker provided the error condition no longer exists and the EIUCNT register again matches the expected value. Following a reset, the MON bit is cleared.

7.3.10 Encoder Pin Status

There are three additional status bits provided in the EIUSTAT register that provide a measure of the state of the three EIU pins following the synchronization buffers (EIAS, EIBS and EIZPS signals). Bit 3 indicates the state of the EIAS signal, bit 4 indicates the state of the EIBS signal and bit 5 gives the state of the EIZPS signal. The value of these status bits read is not affected by any of the control bits in the EIUCTRL register.

7.4 ENCODER EVENT TIMER

7.4.1 Introduction & Overview

The encoder event timer block forms an integral part of the EIU of the ADMC401. The EET accurately times the duration between encoder events. The information provided by the EET may be used to make allowances for the asynchronous timing of encoder and DSP-reading events. As a result, more accurate computations of the position and velocity of the motor shaft may be performed.

The EET consists of a 16-bit encoder event timer, an encoder pulse decimator and a clock divider as shown in Figure 20. The EET clock frequency is selected by the 16-bit read/write EETDIV clock divide register, whose value divides the CLKIN frequency. The contents of the encoder event timer are incremented on each rising edge of the divided clock signal. An EETDIV value of zero gives the maximum divide value of 0x10000 (= 65,536), so that the clock frequency to the encoder event timer is at its minimum possible value.

The quadrature signal from the encoder interface unit is decimated at a rate determined by the 8-bit read/write EETN register. For example, writing a value of 2 to EETN, produces a pulse decimator output train at half the quadrature signal frequency, as shown in Figure 24. The rising edge of this decimated signal is termed a velocity event. Therefore, for an EETN value of 2, a velocity event occurs every two encoder edges, or on each edge of one of the encoder signals. An EETN value of 0 gives an effective pulse decimation value of 256.

On the occurrence of a velocity event, the contents of the encoder event timer are stored in an intermediate Interval Time register. Under normal operation, this register stores the elapsed time between successive velocity events. After, the timer value has been latched at the velocity event, the contents of the encoder event timer are reset to one.

7.4.2 Latching Data from the EET

When using the data from the Encoder Event Timer, it is important to latch a triplet set of data at the same instant in time. The three pieces of data are the contents of the encoder quadrature up/down counter, the stored value in the Interval Time register (giving the precise measured time between the last two velocity events) and the present value of the encoder event timer (giving an indication of how much time has passed since the last velocity event).

The data from the EET can be latched on the occurrence of two different events. The particular event is selected by bit 4 (EETLATCH) of the EIUCTRL register. Setting this EETLATCH bit causes the data to be latched on the time-out of the encoder loop timer (EIUTIMER). At that time, the contents of the encoder quadrature counter (EIUCNT) are latched to a 16-bit, read-only register EETCNT. In addition, the contents of the intermediate Interval Time register are latched to the EETT register and the contents of the encoder event timer are latched to the EETDELTAT register. The three registers, EETCNT, EETT and EETDELTAT then contain the desired triplet of position/speed data required for the control algorithm. In addition, if the time-out of the EIUTIMER is used to generate an EIU loop timer interrupt, the required data is automatically latched and waiting for execution of the interrupt service routine (which may be some time after the time-out instant if there are multiple interrupts in the system). By latching the EIUCNT register to the EETCNT, the user does not have to worry about changes in the EIUCNT register (due to additional encoder edges) prior to servicing of the EIU loop timer interrupt.

The other EET latch event is defined by clearing the EETLATCH bit of the EIUCTRL register. In this mode, whenever, the EIUCNT register is read by the DSP, the current value of the intermediate Interval Time register are latched to the EETT register and the contents of the encoder event timer are latched to the EETDELTAT register. The three registers, EIUCNT, EETT and EETDELTAT now contain the desired triplet of position/speed data required for the control algorithm. Note the difference from before in that the encoder count value is now available in the EIUCNT register.

It is important to realize that the EETT and EETDELTAT registers is only updated by either the time-out of the EIUTIMER register (if EETLATCH bit is set) or the act of reading the EIUCNT register (if the EETLATCH bit is cleared). Therefore if the EETLATCH bit is set, the act of reading the EIUCNT register will not update the EETT and EETDELTAT registers. Following a reset, bit 4 of the EIUCTRL is cleared.

7.4.3 EET Status Register

There is a 1-bit EETSTAT register that indicates whether or not an overflow of the EET has occurred. If the time between successive velocity events is sufficiently long, it is possible that the encoder event timer will overflow. When this condition is detected, bit 0 of the EETSTAT register is set and the EETT register is fixed at 0xFFFF. Reading the EETSTAT register clears the overflow bit and permits the EETT register to be updated at the next velocity event. If an encoder direction reversal is detected by the EIU, the encoder event timer is set to zero and the EETT register is set to its maximum 0xFFFF value. Subsequent velocity events will cause the EETT register to be updated with the correct value. If a value of 0xFFFF is read from the EETT register, bit 0 of the EETSTAT register can be read to determine whether an overflow or direction reversal condition exists. In the case of a direction reversal, the contents of the EETDELTAT register is valid, representing the time from the direction reversal to the instant at which the EIUCNT register is read. On reset the EETN, EETDIV, EETDELTAT and EETT registers are all cleared to zero. Whenever either the EETN or EETDIV registers are written to, the encoder event timer is reset to zero and the EETT register is set to zero.

7.5 EIU/EET INTERFACE, PINS AND REGISTERS

The EIU and EET registers are illustrated in Figure 25, Figure 25, and Figure 27.

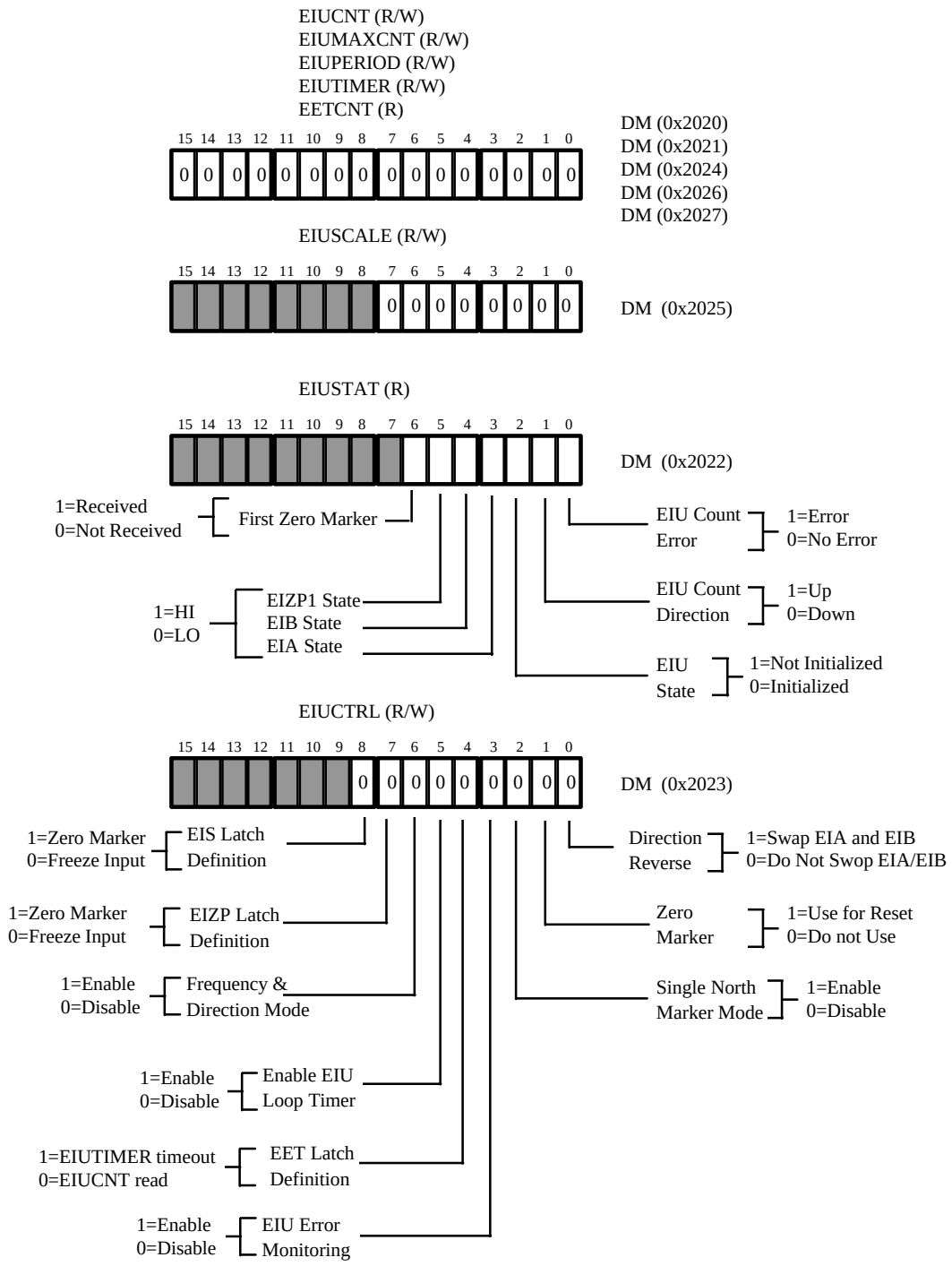


Figure 25: Structure of EIU registers of ADMC401 (shaded bits are unused, reset values included, where defined).

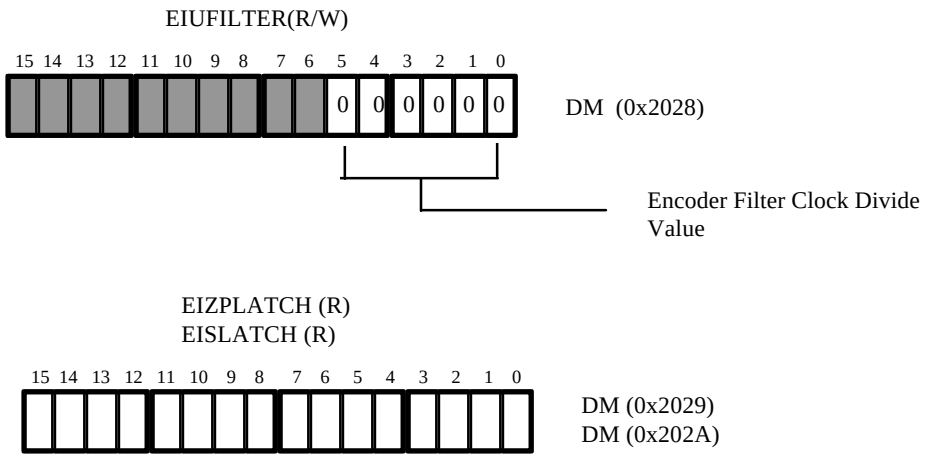


Figure 26: Structure of EIU registers of ADMC401 (shaded bits are unused, reset values included, where defined).

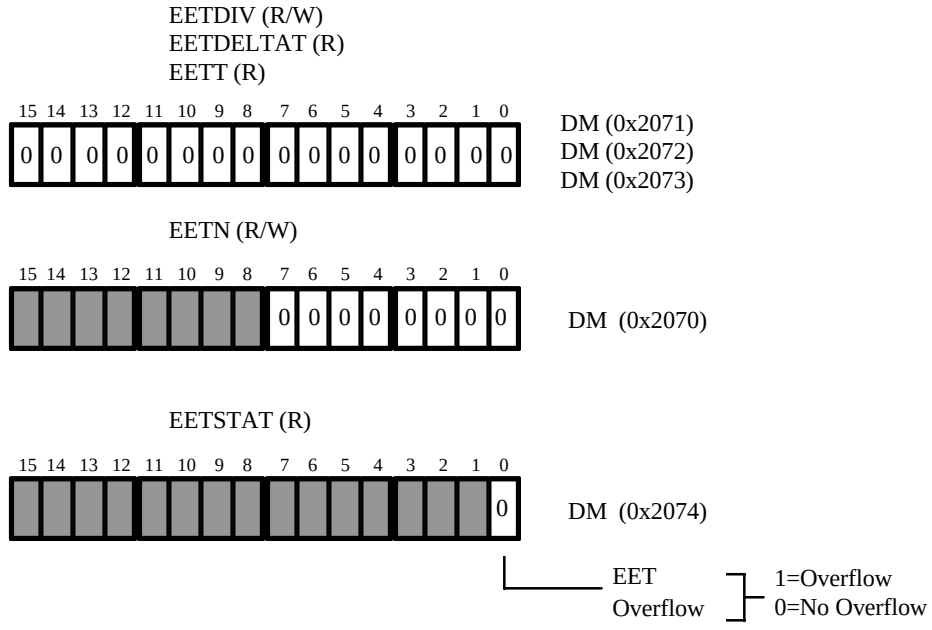


Figure 27: Structure of EET registers of ADMC401 (shaded bits are unused, reset values included, where defined).

8.0 PROGRAMMABLE DIGITAL INPUT/OUTPUT.

8.1 OVERVIEW

The ADMC401 has 12 programmable digital input/output pins called PIO0 to PIO11. Each pin may be individually configured as either an input or an output. An associated data register may be used to read data from pins configured as inputs and write data to pins configured as outputs. In addition, each I/O line may be configured as an interrupts source. Both edge (rising and falling) and level (high and low) interrupts may be detected. Four of the PIO lines (PIO0 to PIO3) have dedicated vector addresses in the interrupt table. The remaining eight interrupts (PIO4 to PIO11) are multiplexed into a single additional interrupt vector location. The PIOFLAG register is used to determine which line caused the interrupt. In addition, all PIO lines may be alternatively configured as PWM trip sources. The PIOPWM register has dedicated bits that may be used to enable this function on each PIO line. In this mode, a low level on any pins configured as a PWM trip source shuts down the PWM in a manner identical to the $\overline{\text{PWMTRIP}}$ pin.

8.2 PIO CONFIGURATION

Each of the 12 programmable input/output lines may be configured as either an input or an output by programming the appropriate bits of the PIODIR register. This 12-bit read/write register has one bit associated with each I/O line; bit 0 corresponds to PIO0 etc. Clearing a bit in the PIODIR register will configure the corresponding pin as an input line. Conversely, setting a bit configures the pin as an output pin. On reset, bits of the PIODIR register are cleared so that all 12 PIO pins are configured as inputs. In addition, all PIO lines are internally pulled down in the ADMC401 so that unconnected lines are seen as low-level inputs.

8.3 PIO DATA READING/WRITING

Associated with the PIO system is a data register, PIODATA, that also has a bit associated with each I/O line. Data written to the PIODATA register will appear on those pins configured as outputs. In addition, reading the PIODATA register will read the data from those pins configured as inputs.

8.4 PIO INTERRUPT GENERATION

Each of the 12 PIO lines may be configured as an interrupt source. Four of the PIO lines, PIO0 to PIO3, have dedicated interrupt vector locations while the remaining eight are multiplexed into an additional interrupt vector. The PIOINTEN enable function is used to enable or disable the interrupt mode on the PIO4 to PIO11 lines. The PICMASK register of the programmable interrupt controller is used to enable interrupts on the four dedicated PIO lines, PIO0 to PIO3.

Interrupts may be generated on either edge (rising or falling) or level (high or low) events by programming the appropriate bits of both the PIOMODE and PIOLEVEL registers. Both registers have a dedicated bit for each of the twelve PIO lines. Setting the appropriate bit of the PIOMODE register configures the interrupt as level sensitive while clearing the bit set the interrupt for edge sensitive. In level-sensitive mode (PIOMODE bit is 1), setting the corresponding bit in the PIOLEVEL register configures the interrupt as active high while clearing the bit in the PIOLEVEL register configures it for active low. In edge-sensitive mode (PIOMODE bit is 0), setting the corresponding bit of the PIOLEVEL register configures the interrupt for rising edge, while clearing the bit configures the interrupt for falling edge. On reset all PIO interrupts are disabled.

The four dedicated PIO interrupts from PIO0 to PIO3 have interrupt vector addresses at program memory addresses 0x44 for PIO0, 0x48 for PIO1, 0x4C for PIO2 and 0x050 for PIO3. In the event of an interrupt on PIO4 to PIO11, the corresponding bit of the PIOFLAG register is set and the general PIO interrupt is activated. This interrupt has a dedicated vector address at location 0x3C. In the interrupt service routine for this interrupt, the user must poll the PIOFLAG register to determine which of the PIO4 to PIO11 lines, that have interrupts enabled, caused the interrupt. Of course, if only one of the PIO4 to PIO11 lines has interrupts enabled, no polling is necessary.

PIO lines that are configured as outputs may also be used to generate interrupts. If, for example, one of the PIO lines is configured simultaneously as an output and as an interrupts source, writing the appropriate data sequence to that line will trigger an interrupt.

8.5 PIO AS PWM TRIP SOURCES

By setting the appropriate bits of the PIOPWM register, each of the twelve PIO lines can be configured as a PWM trip source. In this mode, a low-level on the PIO pin will cause a PWM trip that will disable all six PWM outputs on AH to CL. The disabling of the PWM is independent of the DSP clock, so that the PWM stage can be fully protected even in the event of a loss of clock signal to the DSP. In addition, a PWMTRIP interrupt will be generated when the PWM is reset. However, it is also possible to generate the normal PIO interrupts on the occurrence of a falling-edge on the PIO line. The advantage of this highly flexible structure for PWM shutdown is that multiple fault signals could be applied to the ADMC401 at different PIO lines. The occurrence of a falling-edge on any of them will instantaneously shut down the PWM. However, based on the particular PIO interrupt that is flagged, the user can easily determine the source of the trip. This permits the action of the interrupt service routines following a PWM trip to be tailored to the particular fault that occurred.

On reset, all PIO lines are configured as PWM trip sources. Because, all PIO lines are also configured as inputs and have internal pull-down resistors, any unconnected PIO lines will cause a PWM trip. Therefore, prior to using the PWM system of the ADMC401, it is imperative that the PIO stage be correctly configured for the particular application.

8.6 PIO REGISTERS

The configuration of all registers associated with the PIO system of the ADMC401 are shown in graphically in Figure 28. Each of the registers has a bit associated directly with one of the PIO lines. For example, bit 0 of all registers affects only the PIO0 line of the ADMC401.

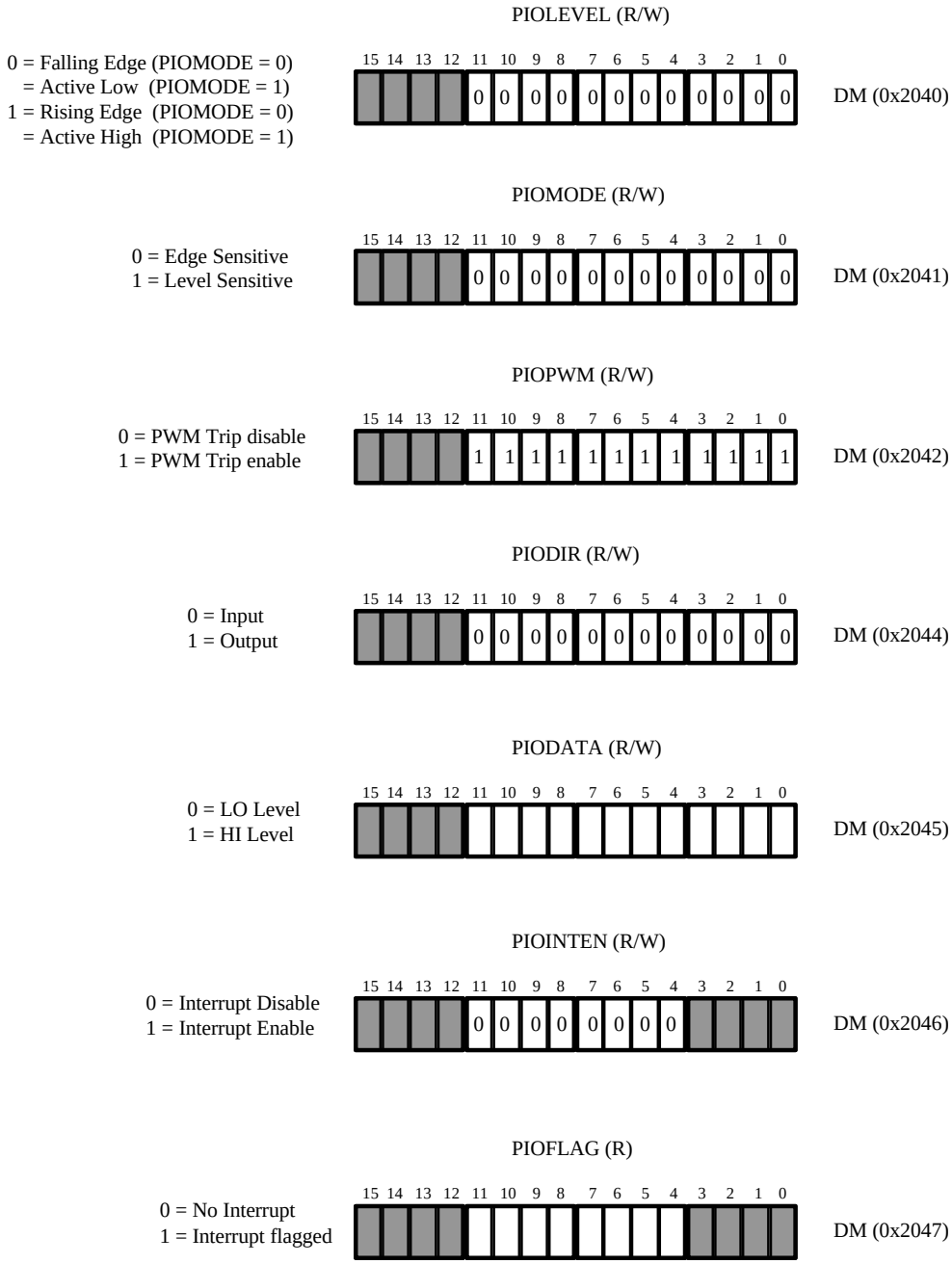


Figure 28: Structure of PIO Registers of ADMC401.

9.0 EVENT TIMERS

9.1 OVERVIEW

The ADMC401 contains a dual channel event timer unit (ETU) that may be used to accurately measure the elapsed time between defined instants on a particular channel. The ETU has two dedicated input pins, ETU0 and ETU1. The ETU system contains a set of 16-bit data registers that are used to store the value of the dedicated ETU timer on the occurrence of defined events on the input pins. A configuration register is used to define the nature of the events on each of the input pins. In addition, a control register is used to initiate event capture on the inputs. A status register may be read to determine the state of the two capture channels. A dedicated ETU interrupt may be generated upon completion of a capture sequence on either the ETU0 or ETU1 channels.

An event may be defined as either a rising or falling edge on the associated ETU0 and ETU1 inputs pins. Therefore, the ETU system can be used to compute the frequency, period, duty cycle or on-time of signals applied at the inputs. A block diagram of the ETU system of the ADMC401 is shown in Figure 29.

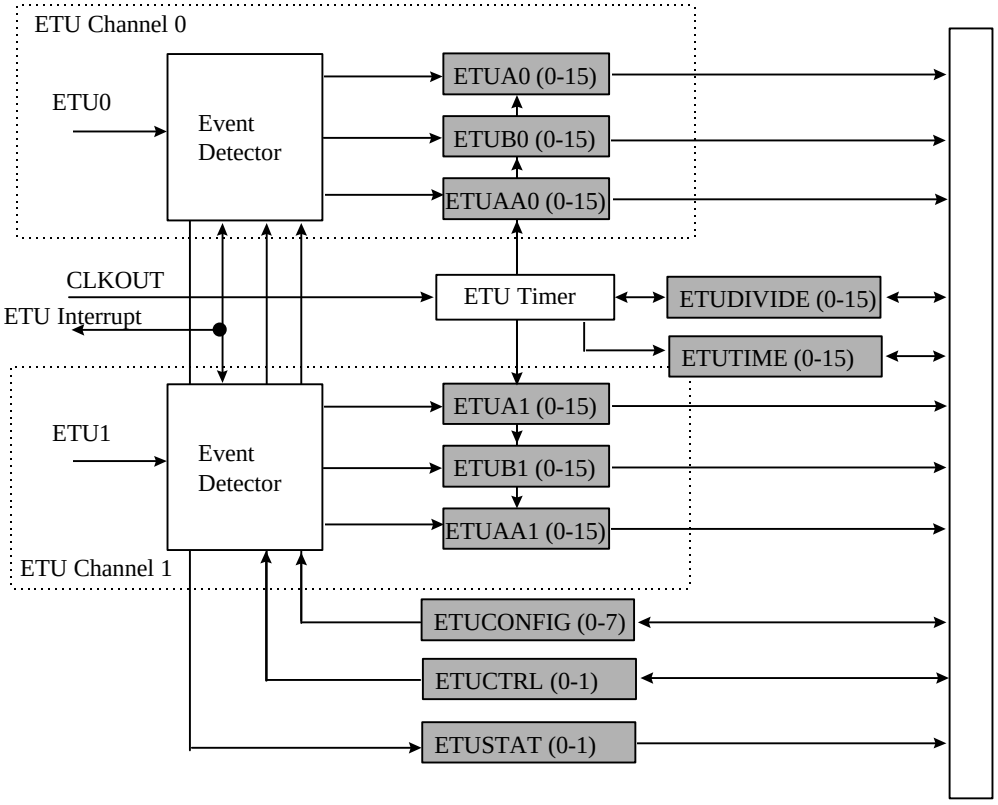


Figure 29: Functional Block Diagram of Event Timer Unit of ADMC401.

9.2 ETU EVENT DEFINITION

The ETU system of the ADMC401 contains a dedicated 16-bit timer whose clock frequency may be programmed using the ETUDIVIDE register. This register divides the CLKIN frequency to provide the clock signal for the ETU timer. The clock

frequency of the ETU timer may be expressed as CLKIN/ETUDIVIDE and is common to both channels. At any time, the contents of the ETU timer may be read in the 16-bit read only ETUTIME register.

There are two events that are used to trigger the ETU, termed event A and event B. By setting the appropriate bits of the ETUCONFIG register, it is possible to define both events A and B as either rising or falling edges on the appropriate pin. For example, setting bit 0 of the ETUCONFIG register, defines event A of the ETU0 channel as a rising edge on the ETU0 pin. Similarly, setting bit 4 of the ETUCONFIG register defines event A of the ETU1 channel as a rising edge on the ETU1 pin. Event A defines the start of the event capture sequence. Associated with each ETU channel are three data registers, ETUA0, ETUB0 and ETUAA0 for ETU channel 0 and ETUA1, ETUB1 and ETUAA1 for ETU channel 1. These data registers store the ETU timer value on the occurrence of the first A event, the first B event and the second A event respectively. For example, for ETU channel 0, ETUA0 store the timer value on the first occurrence of event A on the ETU0 pin, ETUB0 stores the timer value on the first occurrence of event B on the ETU0 pin and ETUAA0 store the timer value on the second occurrence of event A on the ETU0 pin. Registers ETUA1, ETUB1 and ETUAA1 perform the same function for events on ETU channel 1.

9.3 ETU INTERRUPT GENERATION

The completion of the event capture sequence can be defined as either the occurrence of event B or the second occurrence of event A by setting the appropriate bits of the ETUCONFIG register. At the end of the capture sequence, the ETU generates an interrupt. For example, if bit 2 of the ETUCONFIG register is set, ETU channel 0 will generate an ETU interrupt on the occurrence of event B on the ETU0 pin. On the other hand, if bit 6 of the ETUCONFIG register is cleared, ETU channel 1 will generate an ETU interrupt on the occurrence of the second event A on the ETU1 pin. Both ETU channels generate the same interrupt to the DSP when capture is complete. If both ETU channels are used simultaneously, the ETUSTAT register can be polled to determine the status of both channels and determine which caused the interrupt. If capture on ETU channel 0 is complete, bit 0 of the ETUSTAT register is set. Similarly, if event capture on ETU channel 1 is complete, bit 1 of the ETUSTAT register is set. Reading the ETUSTAT register automatically clears all bits of the register.

9.4 ETU OPERATING MODES

The ETU channels of the ADMC401 can operate in two distinct modes; single shot and free-running. The particular mode may be selected for ETU channel 0 by programming bit 3 of the ETUCONFIG register and for ETU channel 1 by programming bit 7 of the ETUCONFIG register. Setting these bits puts the respective ETU channel in free-running mode while clearing the bits enables the single-shot mode. In single-shot mode, upon completion of the capture sequence and consequent generation of the interrupt, further event capture is disabled until the interrupt has been serviced and the appropriate bit of the ETUCTRL register has been set. Setting bit 0 of the ETUCTRL register restarts the capture for ETU channel 0, while bit 1 restarts capture for channel 1. In the free-running mode, the bits of the ETUCTRL register remain set and the ETU channel continues to capture following the generation of the interrupt.

9.5 ETU REGISTERS

The configuration of the ETU registers are summarized graphically in Figure 30.

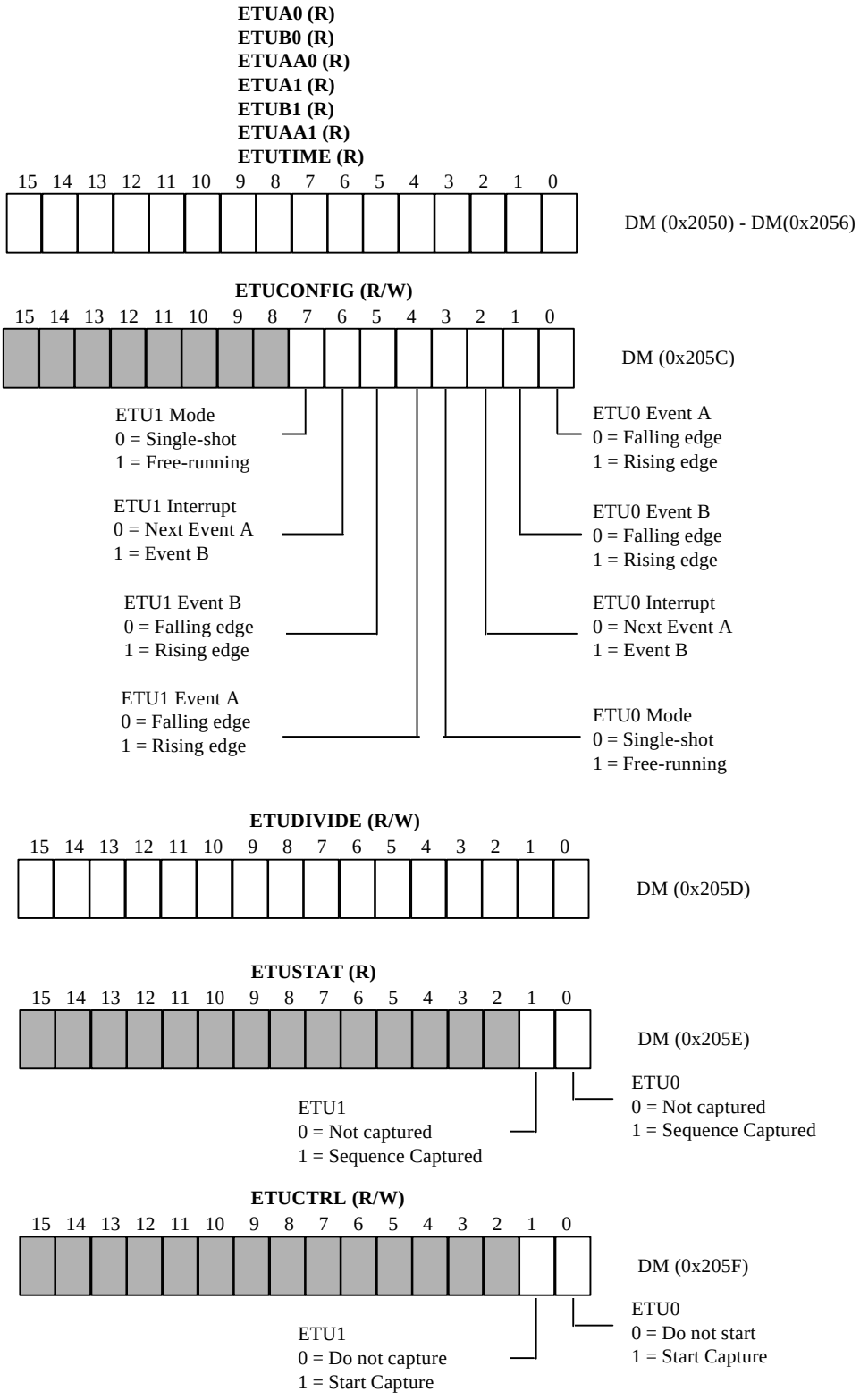


Figure 30: Configuration of ETU Registers.

10. AUXILIARY PWM TIMERS

The ADMC401 provides two auxiliary, fixed-frequency, variable duty cycle PWM outputs that may be used to drive auxiliary switching circuits in the motor control system. Alternatively, by adding appropriate filtering at the output these signals can be used as to provide a simple digital-to-analog converter. These output signals appear on the AUX0 and AUX1 pins and are controlled by the duty cycle registers, AUXTIM0 and AUXTIM1. The auxiliary PWM outputs operate at a fixed frequency that is CLKIN/256. This give an auxiliary PWM switching frequency of 50.7 kHz for a 13 MHz CLKIN. The output duty cycle at the auxiliary PWM output pin is controlled by comparing the 8-bit auxiliary PWM duty-cycle registers, AUXTIM0 and AUXTIM1 with the contents of a timer. The value written to these registers may range from 0 to 255 so that duty-cycles from 0 to 99.6% may be produced at the output pins. A simple filter at the output could then be used to produce a corresponding analog output from 0 to 4.98 V. The outputs of the two auxiliary PWM timer circuits are synchronized on their rising edges. When the auxiliary timer registers are written to the value becomes effective immediately. Therefore, if the value is smaller than the present timer value, the outputs go low immediately. The correct duty cycle appears for the subsequent auxiliary PWM period. On reset, the AUXTIM0 and AUXTIM1 registers are cleared so that no auxiliary PWM signals are produced and the AUX0 and AUX1 pins are low until these registers are programmed. The format of the AUXTIM0 and AUXTIM1 registers is shown in Figure 31.

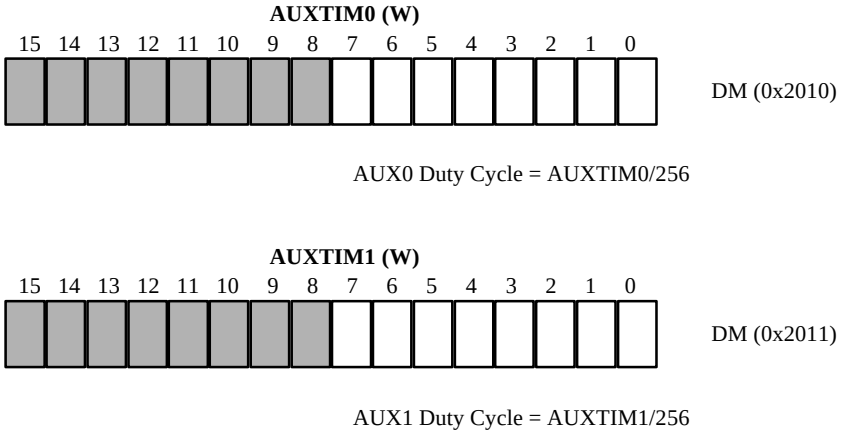


Figure 31: Configuration of auxiliary PWM timer registers

11 WATCHDOG TIMER

11.1 OVERVIEW

The watchdog timer is used as a protection mechanism against unintentional software events causing the DSP to become stuck in infinite loops. It can be used to cause a compete DSP and peripheral reset in the event of such a software error. The watchdog timer consists of a 16-bit timer that is clocked at the CLKIN rate, t_{CKI}. In fact, for increased ruggedness of the watchdog timer circuit, a separate clock signal is derived directly from the crystal or oscillator inputs just for the watchdog circuit.

The watchdog timer is disabled on power up. The watchdog timer is enabled by writing a *TIMEOUT* value to the WDTIMER register. This also resets the WDFLAG in the SYSSTAT register. Once the watchdog timer has been initialized, the contents of the timer are decremented at the CLKIN rate. In order to prevent a watchdog timer trip, it is

necessary to write again to the WDTIMER register. For all writes to the WDTIMER register (subsequent to the initial write), it is unimportant what value is written. The act of writing to the WDTIMER register automatically reloads the initial *TIMEOUT* value. If the watchdog timer is not re-written to after an interval:

$$T_{WDT} = WDTIMER \times t_{CKI}$$

the watchdog timer will decrement to zero and a watchdog trip will be generated. In this case a complete reset of the DSP core and motor control peripherals is initiated and bit 1 of the SYSSTAT register (WDFLAG) is set. Following the reset, the DSP core can determine if the reset was caused by a watchdog trip (and so take appropriate action) or if it is due to the normal power up or reset sequence. The watchdog timer remains disabled while the WDFLAG is set to prevent continuous watchdog trips. The watchdog timer can be restarted and the WDFLAG reset by writing a nonzero *TIMEOUT*' value to the WDTIMER register. The WDFLAG will be reset but the watchdog timer will remain disabled if 0x0000 is written to the WDTIMER register. The watchdog timer is reset by an asynchronous low input on the $\overline{\text{RESET}}$ line. The watchdog circuit is not reset by the general peripheral reset (by calling the PER_RST ROM utility or toggling the FL2 flag).

12 PROGRAMMABLE INTERRUPT CONTROLLER.

12.1 OVERVIEW

The ADMC401 uses the $\overline{\text{IRQ2}}$ pin of the DSP core to generate a peripheral interrupt. There are multiple sources of peripheral interrupts e.g. the ADC block, PIO block, EIU block, ETU block and PWM block. To avoid a software latency in determining the source of the interrupt a Programmable Interrupt Controller (PIC) is used. The PIC block essentially augments the existing ADSP2171 vector table with additional vector addresses, one address for each of the eleven new interrupt sources. With the occurrence of an interrupt from the peripheral blocks, the PIC block generates an address which points to the corresponding vector address in the DSP vector table. The PIC consists of an output register, PICVECTOR. The contents of this register contain a pointer to an entry in the DSP vector table. During normal operation the interrupt service routine (ISR) located at vector address 0x0004 (or the $\overline{\text{IRQ2}}$ /peripheral interrupt) jumps to the address pointed to by the PICVECTOR register.

The vector addresses between 0x00 and 0x2C are reserved for the DSP core interrupts. The vector table addresses from PM(0x30) to PM(0x58) are reserved for use by peripheral interrupt service routines. Each vector address occupies four addresses of PM. The size of the PICVECTOR from the DSP prospective is 16 bits, however only five of these bits are necessary to cover the range of addresses. The priority of the peripheral interrupts is fixed in hardware. The ISR at address PM(0x30) has the highest priority while the ISR at address PM(0x58) has the lowest.

In the case of multiple simultaneous interrupts, the PIC will load the PICVECTOR register with the interrupt that has the highest priority. In between reads of the PICVECTOR register (while the DSP is servicing other interrupts for example) the PICVECTOR is updated with the highest priority of any peripheral interrupts. This ensures that when the $\overline{\text{IRQ2}}$ is again re-asserted, the highest priority interrupt that occurred since the last reading of the PICVECTOR register is now waiting to be serviced.

Note that in normal operation the $\overline{\text{IRQ2}}$ ISR should disable interrupts and then jump to the address pointed to by the PICVECTOR register. The onus is on the subsequent peripheral ISR to re-enable interrupts early on in the ISR if desired. The PIC block will assert a new interrupt only after the PICVECTOR register has been read. The DSP will respond to the interrupt only after interrupts have been re-enabled on $\overline{\text{IRQ2}}$. When PICVECTOR is read, if another interrupt is pending in

the PIC, then the $\overline{IRQ2}$ line to the DSP remains LO and no edge will be seen. In order to catch all interrupts, $\overline{IRQ2}$ interrupts should be configured as level sensitive in the ICNTL register.

The four least significant PIO pins are assigned unique vector addresses. An interrupt on any of the remaining eight lines (PIO4 to PIO11) will trigger a separate fifth PIO interrupt that has its own vector address. The PIOFLAG register can be read to determine the exact source of this fifth interrupt. An 11-bit PICMASK register can be used to enable or disable any or all of the eleven peripheral interrupt sources.

12.2 PIC REGISTERS

The program memory addresses of the vector table reserved for the various peripheral interrupts are summarized in Table 10. The PICMASK register is described in Figure 32.

PM Address	Function
0x30	ADC end of conversion interrupt
0x34	PWMSYNC interrupt
0x38	EIU loop timer time out interrupt
0x3C	PIO4 to PIO11 interrupt
0x40	EIU counter error interrupt
0x44	ETU interrupt
0x48	PIO0 interrupt
0x4C	PIO1 interrupt
0x50	PIO2 interrupt
0x54	PIO3 interrupt
0x58	PWM trip interrupt

Table 10: Peripheral Interrupt Vector Address Table.

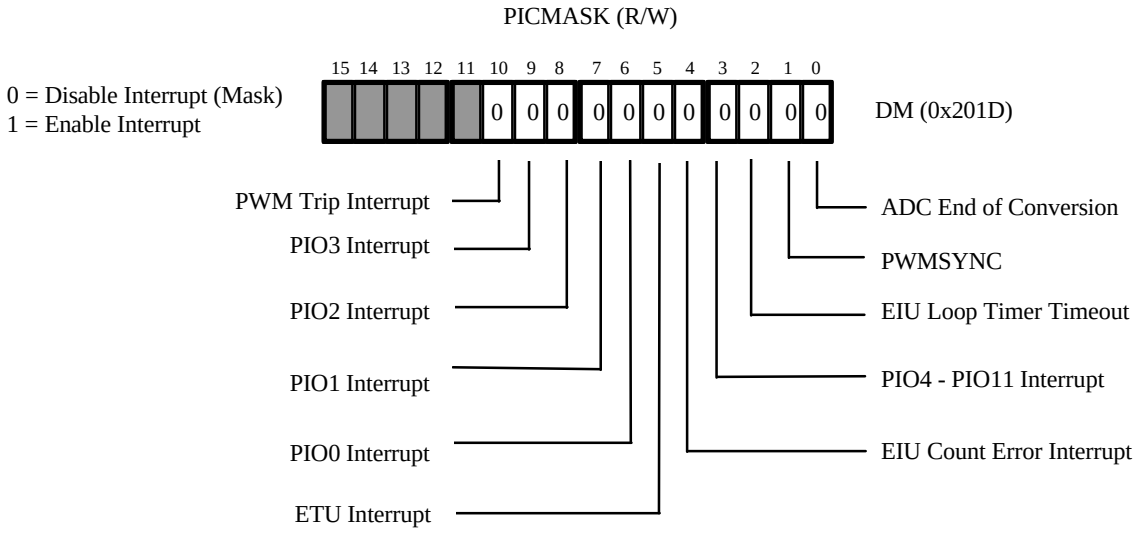


Figure 32: Structure of PICMASK Register of ADMC401.

13.0 SYSTEM CONTROLLER.

13.1 OVERVIEW

The system controller has a number of functions:

1. It decodes the DSP address bus and selects the appropriate peripheral registers.
2. It controls the SPORT1 multiplexer select lines.
3. It resets the peripherals and control registers on hardware, software or watchdog initiated resets.
4. It can be used to control the peripheral test modes.

13.2 DSP INTERFACE AND MEMORY MAP

All data transfer between the DSP core and the peripherals is controlled by the system controller. When the DSP core puts a valid peripheral address on the DM address bus, the system controller generates an active low chip select signal for the selected peripheral register. If the resister is a read only register data will be loaded onto the DSP DM data bus only on the rising edge of $\overline{RD}$. Conversely, if the resister is a write register, data will be loaded into the register only on the rising edge of $\overline{WR}$. The peripheral registers are right justified, i.e. the LSB of each register is connected to the LSB of the 16 bit DSP DM data bus. Any unused bits are connected to a logic zero. The ADMC401 peripheral registers are memory mapped to the DSP data address space, starting at address 0x2000. The peripheral registers are summarized in Table 11.

ADDRESS	NAME	TYPE	BITS	RESET VALUE	FUNCTION
0x2000 - 0x2007					Reserved
0x2008	PWMTM	R/W	15:0	0x0000	PWM period register
0x2009	PWMDT	R/W	9:0	0x0000	PWM deadtime register
0x200A	PWMPD	R/W	9:0	0x0000	PWM pulse deletion register
0x200B	PWMGATE	R/W	9:0	0x0000	PWM chopping control
0x200C	PWMCHA	R/W	15:0	undefined	PWM channel A duty cycle control
0x200D	PWMCHB	R/W	15:0	undefined	PWM channel B duty cycle control
0x200E	PWMCHC	R/W	15:0	undefined	PWM channel C duty cycle control
0x200F	PWMSEG	R/W	8:0	0x000	PWM crossover & output enable
0x2010	AUXTIM0	W	7:0	undefined	Aux. PWM channel 0 duty cycle
0x2011	AUXTIM1	W	7:0	undefined	Aux. PWM channel 1 duty cycle
0x2012 - 0x2014					Reserved
0x2015	MODECTRL	R/W	6:4	0x00	Mode control register
0x2016	SYSSTAT	R	4:0	undefined	System status register
0x2017					Reserved
0x2018	WDTIMER	R/W	15:0	undefined	Watchdog timer register
0x2019 - 0x201B					Reserved
0x201C	PICVECTOR	R	15:0	undefined	Peripheral interrupt address
0x201D	PICMASK	R/W	10:0	0x000	Peripheral interrupt mask register
0x201E - 0x201F					Reserved
0x2020	EIUCNT	R/W	15:0	0x0000	Position Count Value
0x2021	EIUMAXCNT	R/W	15:0	0x0000	Maximum EIUCNT Value
0x2022	EIUSTAT	R	6:0	undefined	EIU Status Register
0x2023	EIUCTRL	R/W	8:0	0x00	EIU Control Register

0x2024	EIUPERIOD	R/W	15:0	0x0000	EIU Loop Timer Period Register
0x2025	EIUSCALE	R/W	7:0	0x00	EIU Loop Timer Scale Register
0x2026	EIUTIMER	R/W	15:0	0x0000	EIU Loop Timer Register
0x2027	EETCNT	R	15:0	0x0000	Latched Copy of EIUCNT
0x2028	EIUFILTER	R/W	5:0	0x0	EIU Filter Control Register
0x2029	EIZPLATCH	R	15:0	undefined	ZP1 Latch Register
0x202A	EISLATCH	R	15:0	undefined	ZP2 Latch Register
0x202B - 0x202F					Reserved
0x2030	ADC1	R	15:0	undefined	ADC1 data register
0x2031	ADC2	R	15:0	undefined	ADC2 data register
0x2032	ADC3	R	15:0	undefined	ADC3 data register
0x2033	ADC4	R	15:0	undefined	ADC4 data register
0x2034	ADC5	R	15:0	undefined	ADC5 data register
0x2035	ADC6	R	15:0	undefined	ADC6 data register
0x2036	ADC7	R	15:0	undefined	ADC7 data register
0x2037	ADC8	R	15:0	undefined	ADC8 data register
0x2038	ADCCTRL	R/W	3:0	0x08	ADC control register
0x2039	ADCSTAT	R	0	undefined	ADC status register
0x203A	ADCTEST	R/W	3:0	0x0	ADC channel select register
0x203B - 0x203F					Reserved
0x2040	PIOLEVEL	R/W	11:0	0x0000	PIO interrupt select
0x2041	PIOMODE	R/W	11:0	0x0000	PIO interrupt edge/level select
0x2042	PIOPWM	R/W	11:0	0x0FFF	PIO PWMTRIP enable register
0x2043					Reserved
0x2044	PIODIR	R/W	11:0	0x0000	PIO direction control
0x2045	PIODATA	R/W	11:0	undefined	PIO data register
0x2046	PIOINTEN	R/W	11:4	0x0000	PIO interrupt enable
0x2047	PIOFLAG	R	11:4	undefined	PIO interrupt flag (PIO4 to PIO11)
0x2048 - 0x204F					Reserved
0x2050	ETUA0	R	15:0	undefined	Event A Capture - Channel 0
0x2051	ETUB0	R	15:0	undefined	Event B Capture - Channel 0
0x2052	ETUAA0	R	15:0	undefined	Event AA Capture - Channel 0
0x2053	ETUA1	R	15:0	undefined	Event A Capture - Channel 1
0x2054	ETUB1	R	15:0	undefined	Event B Capture - Channel 1
0x2055	ETUAA1	R	15:0	undefined	Event AA Capture - Channel 1
0x2056	ETUTIME	R	15:0	undefined	ETU Timer Value
0x205C	ETUCONFIG	R/W	7:0	0x00	ETU Configuration Register
0x205D	ETUDIVIDE	R/W	15:0	0x0000	ETU Clock Divide Register
0x205E	ETUSTAT	R	1:0	undefined	ETU Status Register
0x205F	ETUCTRL	R/W	1:0	0x0	ETU Control Register
0x2060	PWMSYNCWT	R/W	7:0	0x27	PWMSYNC width control
0x2061	PWMSWT	R/W	0	0x0	PWM software trip
0x2062 - 0x206F					Reserved
0x2070	EETN	R/W	7:0	0x00	EET Pulse Decimator Register
0x2071	EETDIV	R/W	15:0	0x0000	EET Clock Divider Register

0x2072	EETDELTAT	R	15:0	0x0000	EET Timer Period Register
0x2073	EETT	R	15:0	0x0000	EET Delta Timer Register
0x2074	EETSTAT	R	0	0x0	EET Status Register
0x2075 - 0x23FF					Reserved

Table 11: Peripheral Registers of ADMC401

13.3 MODECTRL REGISTER

The SPORT1 peripheral pins are configured using the MODECTRL register. Two bits of this register control the SPORT1 UART and DR1A/DR1B multiplexer. Setting the UARTEN bit connects DR1A to the RFS1 input which allows SPORT1 to be used as a UART port. The DR1SEL bit selects either pins DR1A or DR1B. The reset condition for all bits in this register is zero. Setting the UARTEN bit will also multiplex the FL1 pin on the ADSP2171 core to the RFS1 pin on the ADMC401. In this mode this pin is called $\overline{\text{SROM}}$ and is used to reset the SROM. The structure of the SPORT1 multiplex and control lines is shown in Figure 33.

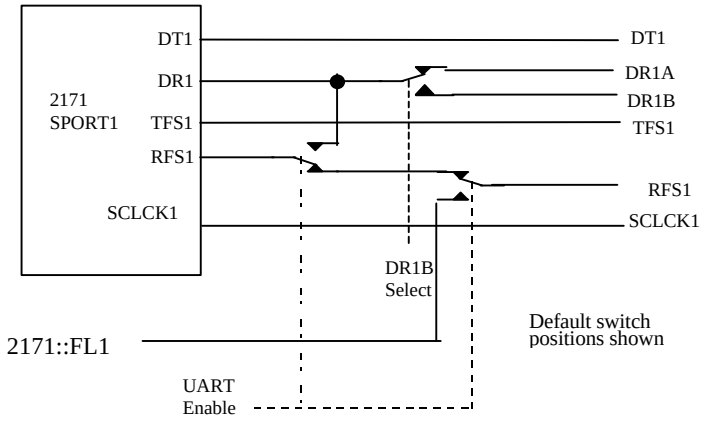


Figure 33: SPORT1 and $\overline{\text{SROM}}$ Multiplexer.

Additionally, setting bit 6 of the MODECTRL register places the PWM in the double update mode. Clearing the bit places the PWM in the single update mode. Bit 6 is cleared by default on power up or following a reset.

13.4 SYSSTAT REGISTER

The SYSSTAT register indicates various status information of the ADMC401 such as:

1. The status of the $\overline{\text{PWMTRIP}}$ pin.
2. The status of the watchdog flag
3. The status of the PWMPOL pin
4. The phase of the PWM
5. The status of the SR mode of the PWM

13.5 SYSTEM CONTROLLER INTERFACE & REGISTERS

The registers of the system control block are illustrated in Figure 34.

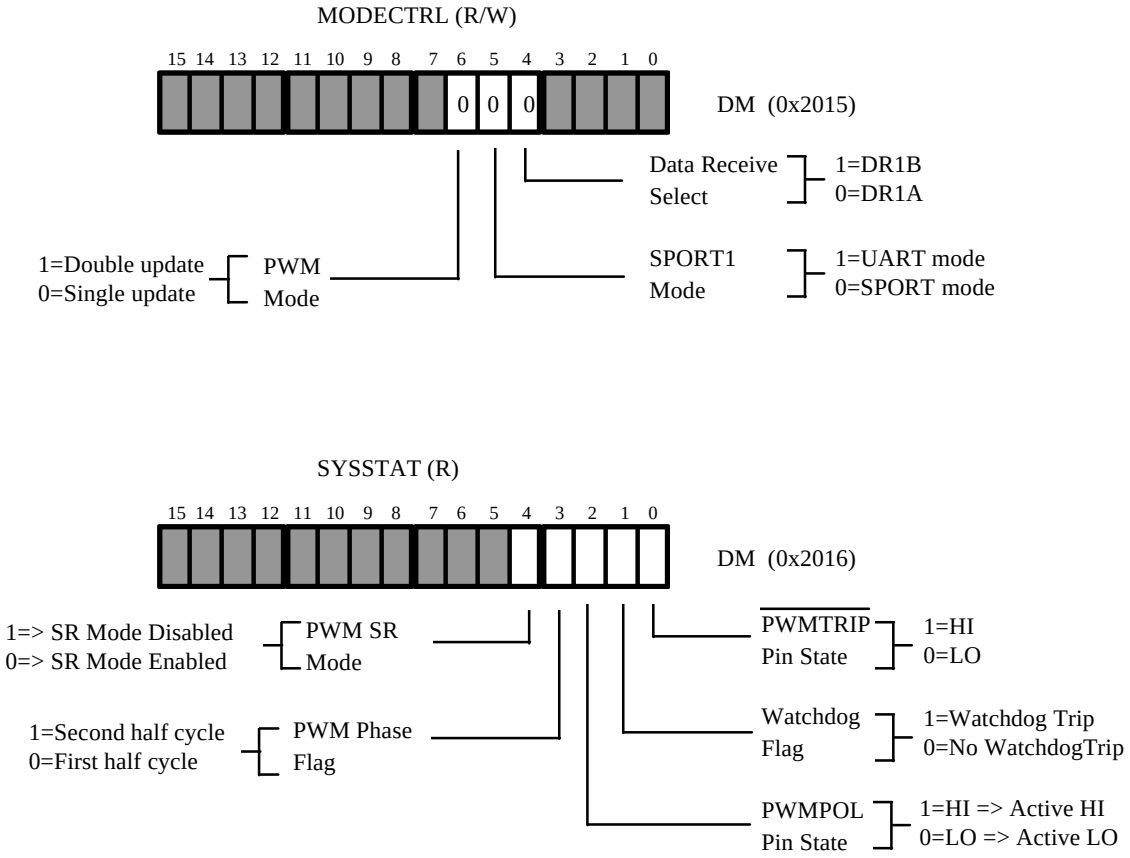


Figure 34: Configuration of MODECTRL and SYSSTAT registers of ADMC401.