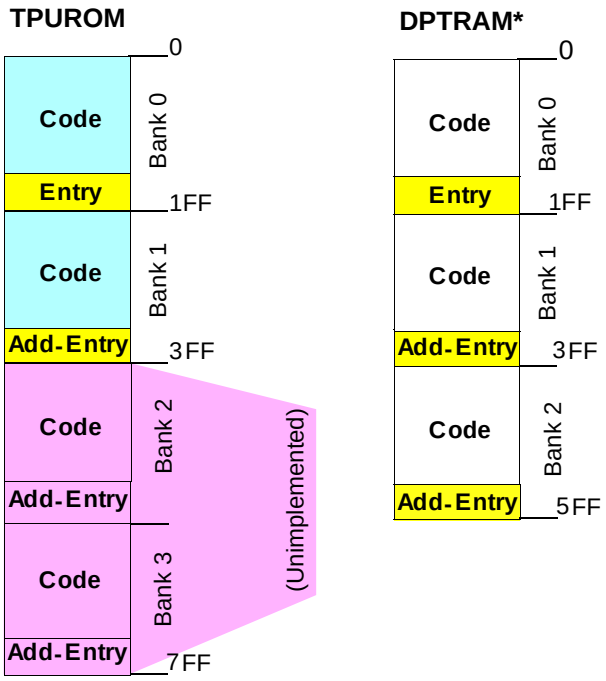
APPENDIX D
TPU ROM FUNCTIONS

The following pages provide brief descriptions of the pre-programmed functions in the TPU3. For detailed descriptions, refer to the programming note for the individual function. Motorola Programming Note TPUPN00/D, *Using the TPU Function Library and TPU Emulation Mode*, provides a list of available programming notes.

D.1 Overview

The TPU3 contains 4 Kbytes of microcode ROM. This appendix defines the functions that are in the standard ROM on the MPC555. The TPU3 can have up to 8 Kbytes of memory and a maximum of four entry tables (see **Figure D-1**).



*The DPTRAM is located at 0x30 2000 until it is switched to emulation mode. In emulation mode, the DPTRAM is accessible by the TPUs only.

Figure D-1 TPU3 Memory Map

The TPU3 can address up to 8 Kbytes of memory at any one time. It has 4 Kbytes of internal ROM, located in banks zero and one, and 6 Kbytes of dual ported SRAM (DPTRAM), located in banks zero, one and two. As only one type of memory can be used at a time, the TPU3 must either use the internal ROM or the SRAM. Functions from both memory types cannot be used in conjunction.

A new feature of the TPU3 microcode ROM is the existence of two entry tables in the 4 Kbytes of internal ROM. Each entry table has a set of sixteen functions and the user defines which of the two tables the TPU3 will be able to access. Only one table can be used at a time and functions from the two entry tables cannot be mixed. The default entry table is located in bank zero. This table is identical to the standard microcode ROM in the TPU2, so any CPU code written for the TPU2 will work unchanged on the TPU3. The TPU2 and TPU3 ROMs are different than the original TPU ROM. The functions in the default entry table in bank zero are listed in Table 1.

Table D-1 Bank 0 Functions

Function Number	Function Nickname	Function Name
$F	PTA	Programmable Time Accumulator
$E	QOM	Queued Output Match
$D	TSM	Table Stepper Motor
$C	FQM	Frequency Measurement
$B	UART	Universal Asynchronous Receiver/Transmitter
$A	NITC	New Input Capture/Input Transition Counter
9	COMM	Multiphase Motor Commutation
8	HALLD	Hall Effect Decode
7	MCPWM	Multi-Channel Pulse Width Modulation
6	FQD	Fast Quadrature Decode
5	PPWA	Period/Pulse Width Accumulator
4	OC	Output Compare
3	PWM	Pulse Width Modulation
2	DIO	Discrete Input/Output
1	SPWM	Synchronized Pulse Width Modulation
0	SIOP	Serial Input/output Port

The functions in the entry table in bank one are listed in Table 2.

Table D-2 Bank 1 Functions

Function Number	Function Nickname	Function Name
$F	PTA	Programmable Time Accumulator
$E	QOM	Queued Output Match
$D	TSM	Table Stepper Motor
$C	FQM	Frequency Measurement
$B	UART	Universal Asynchronous Receiver/Transmitter
$A	NITC	New Input Capture/Input Transition Counter
9	COMM	Multiphase Motor Commutation
8	HALLD	Hall Effect Decode
7	Reserved	
6	FQD	Fast Quadrature Decode
5	ID	Identification
4	OC	Output Compare
3	PWM	Pulse Width Modulation

Table D-2 Bank 1 Functions

Function Number	Function Nickname	Function Name
2	DIO	Discrete Input/Output
1	RWTPIN	Read/Write Timers and Pin
0	SIOP	Serial Input/output Port

The functions in the bank one entry table are identical to the bank zero entry table functions with three exceptions. Function 1, SPWM, has been replaced by RWTPIN. This is a function that allows the user to read and write to the TPU timebases and corresponding pin. Function 5, PPWA, is now an identification function in the second table. The microcode ROM revision number is provided by this function. Finally, Function 7, MCPWM, has been removed and left open for future use.

The CPU selects which entry table to use by setting the ETBANK field in the TPUMCR2 register. This register is write once after reset. Although one entry table is specified at start-up, it is possible, in some cases, to use functions from both tables without resetting the microcontroller. A customer may, for example, wish to use the ID function from bank one to verify the TPU microcode version but then use the MCPWM function from bank zero. As a customer will typically only run the ID function during system configuration, and not again after that, the bank one entry table can be changed to the bank zero entry table using the soft reset feature of the TPU3. The procedure should be:

1. Set ETBANK field in TPUMCR2 to %01 to select the entry table in bank one
2. Run the ID function
3. Stop the TPU3 by setting the STOP bit in the TPUMCR to one.
4. Reset the TPU3 by setting the SOFTRST bit in the TPUMCR2 register
5. Wait at least nine clocks
6. Clear the SOFTRST bit in the TPUMCR2 register

The TPU3 stays in reset until the CPU clears the SOFTRST bit. After the SOFTRST bit has been cleared the TPU3 will be reset and the entry table in bank zero will be selected by default. To select the bank zero entry table write %00 to the ETBANK field in TPUMCR 2. It is good practice always to initialize any write once register to ensure an incorrect value is not accidentally written.

The descriptions below document the functions listed in **Table D-1** (bank0) and **Table D-2** (bank1) of the TPU3 ROM module.

D.2 Programmable Time Accumulator (PTA)

PTA accumulates a 32-bit sum of the total high time, low time, or period of an input signal over a programmable number of periods or pulses. The period accumulation can start on a rising or falling edge. After the specified number of periods or pulses, the PTA generates an interrupt request.

From one to 255 period measurements can be accumulated before the TPU interrupts the CPU, providing instantaneous or average frequency measurement capability.

Figure D-2 shows all of the host interface areas for the PTA function.

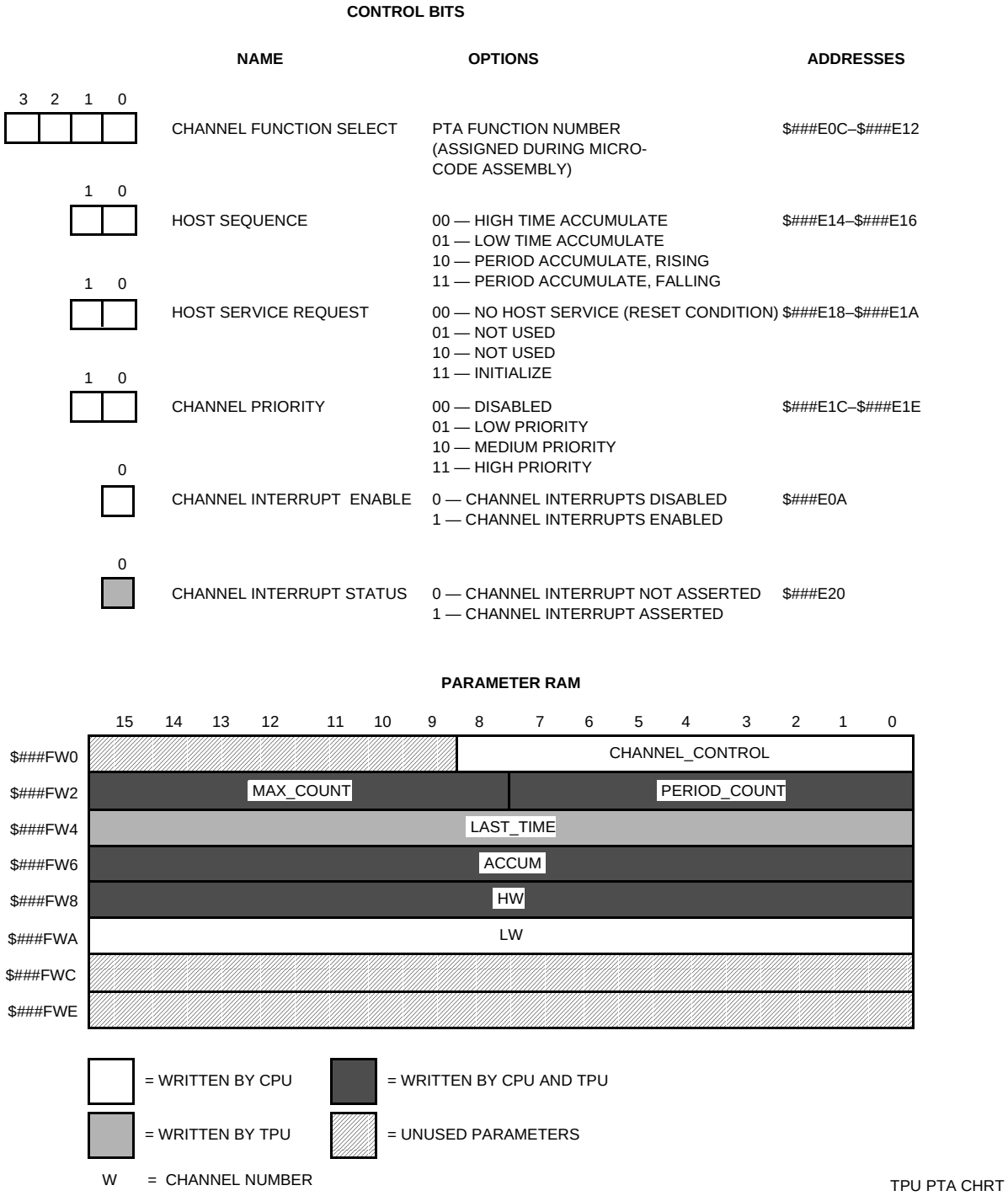


Figure D-2 PTA Parameters

D.3 Queued Output Match TPU Function (QOM)

QOM can generate single or multiple output match events from a table of offsets in parameter RAM. Loop modes allow complex pulse trains to be generated once, a specified number of times, or continuously. The function can be triggered by a link from another TPU channel. In addition, the reference time for the sequence of matches can be obtained from another channel. QOM can generate pulse-width modulated waveforms, including waveforms with high times of 0% or 100%. QOM also allows a TPU channel to be used as a discrete output pin.

Figure D-3 shows all of the host interface areas for the QOM function. The bit encodings shown in **Table D-3** describe the corresponding fields in parameter RAM.

Table D-3 QOM Bit Encoding

A	Timebase Selection
0	Use TCR1 as Timebase
1	Use TCR2 as Timebase

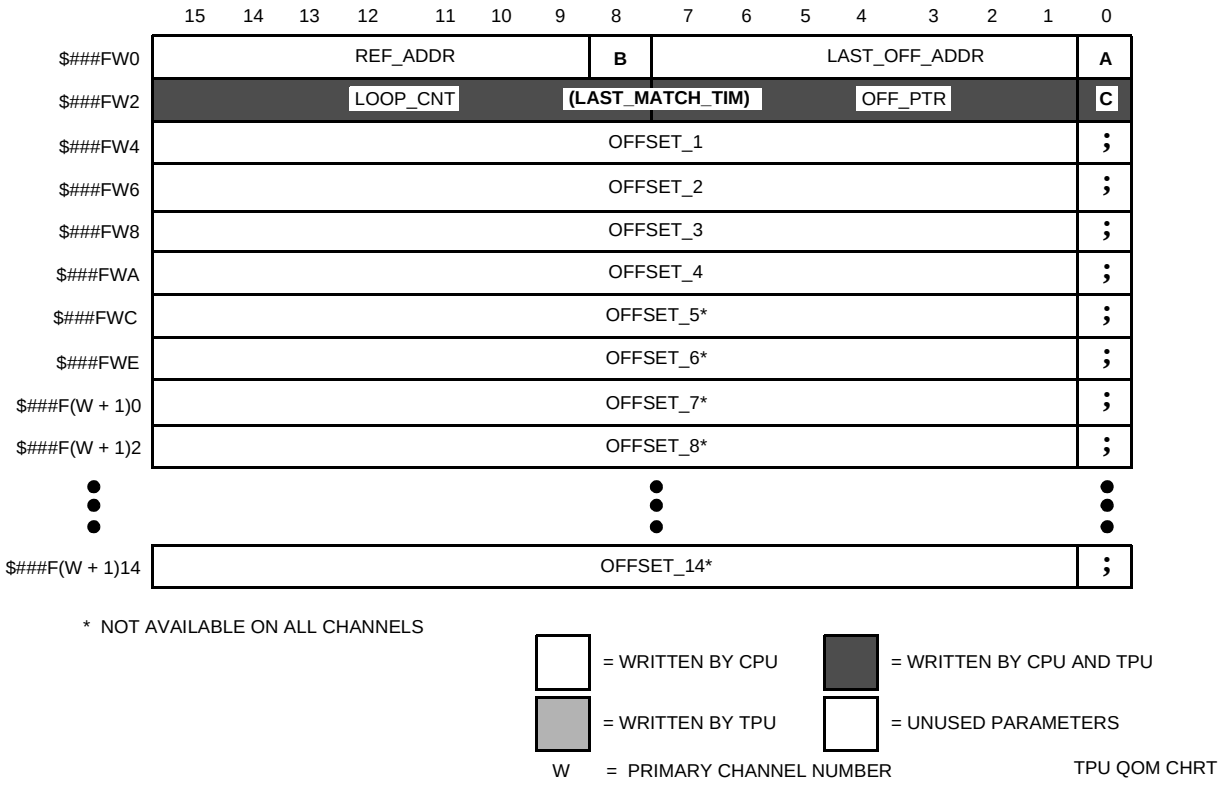
	Edge Selection
0	Falling Edge at Match
1	Rising Edge at Match

B:C	Reference for First Match
00	Immediate TCR Value
01	Last Event Time
10	Value Pointed to by REF_ADDR
11	Last Event Time

CONTROL BITS

3 2 1 0	NAME	OPTIONS	ADDRESSES
3 2 1 0	CHANNEL FUNCTION SELECT	QOM FUNCTION NUMBER (ASSIGNED DURING MICRO-CODE ASSEMBLY)	$###E0C–$###E12
1 0	HOST SEQUENCE	00 — SINGLE-SHOT MODE 01 — LOOP MODE 10 — CONTINUOUS MODE 11 — CONTINUOUS MODE	$###E14–$###E16
1 0	HOST SERVICE REQUEST	00 — NO HOST SERVICE (RESET CONDITION) 01 — INITIALIZE, NO PIN CHANGE 10 — INITIALIZE, PIN LOW 11 — INITIALIZE, PIN HIGH	$###E18–$###E1A
1 0	CHANNEL PRIORITY	00 — DISABLED 01 — LOW PRIORITY 10 — MEDIUM PRIORITY 11 — HIGH PRIORITY	$###E1C–$###E1E
0	CHANNEL INTERRUPT ENABLE	0 — CHANNEL INTERRUPTS DISABLED 1 — CHANNEL INTERRUPTS ENABLED	$###E0A
0	CHANNEL INTERRUPT STATUS	0 — CHANNEL INTERRUPT NOT ASSERTED 1 — CHANNEL INTERRUPT ASSERTED	$###E20

PARAMETER RAM



* NOT AVAILABLE ON ALL CHANNELS

= WRITTEN BY CPU

= WRITTEN BY CPU AND TPU

= WRITTEN BY TPU

= UNUSED PARAMETERS

W = PRIMARY CHANNEL NUMBER

TPU QOM CHRT

Figure D-3 QOM Parameters

D.4 Table Stepper Motor (TSM)

The TSM function provides for acceleration and deceleration control of a stepper motor with a programmable number of step rates up to 58. TSM uses a table in parameter RAM, rather than an algorithm, to define the stepper motor acceleration profile, allowing the user to fully define the profile. In addition, a slew rate parameter allows fine control of the terminal running speed of the motor independent of the acceleration table. The CPU need only write a desired position, and the TPU accelerates, slews, and decelerates the motor to the required position. Full and half step support is provided for two-phase motors. In addition, a slew rate parameter allows fine control of the terminal running speed of the motor independent of the acceleration table.

Figure D-4 and **Figure D-5** show all of the host interface areas for the TSM function when operating in master and slave mode, respectively.

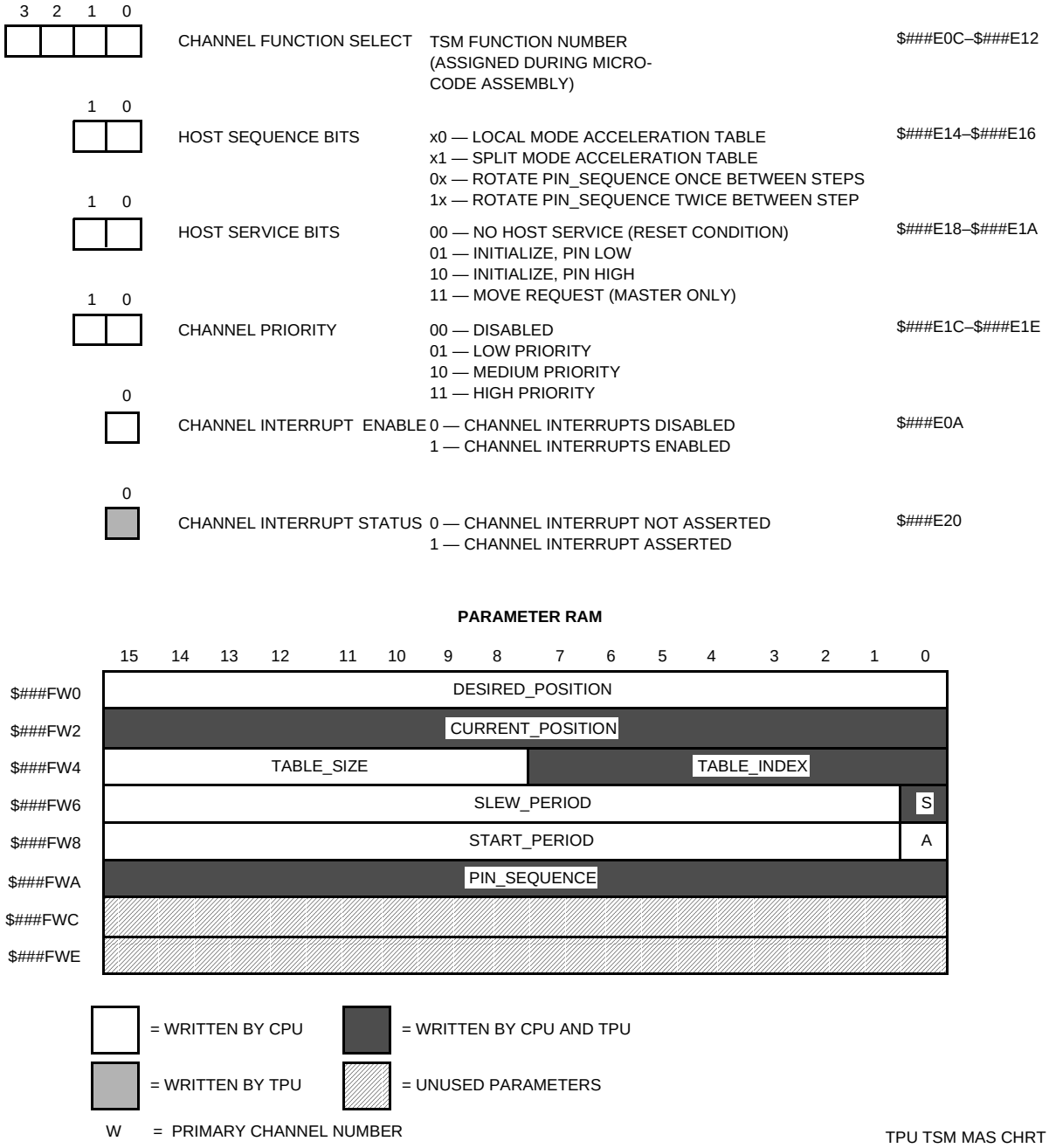


Figure D-4 TSM Parameters — Master Mode

CONTROL BITS

	NAME	OPTIONS	ADDRESSES
3 2 1 0	CHANNEL FUNCTION SELECT	TSM FUNCTION NUMBER (ASSIGNED DURING MICRO-CODE ASSEMBLY)	$###E0C–$###E12
1 0	HOST SEQUENCE BITS	x0 — LOCAL MODE ACCELERATION TABLE x1 — SPLIT MODE ACCELERATION TABLE 0x — ROTATE PIN_SEQUENCE ONCE BETWEEN STEPS 1x — ROTATE PIN_SEQUENCE TWICE BETWEEN STEP	$###E14–$###E16
1 0	HOST SERVICE BITS	00 — NO HOST SERVICE (RESET CONDITION) 01 — INITIALIZE, PIN LOW 10 — INITIALIZE, PIN HIGH 11 — MOVE REQUEST (MASTER ONLY)	$###E18–$###E1A
1 0	CHANNEL PRIORITY	00 — DISABLED 01 — LOW PRIORITY 10 — MEDIUM PRIORITY 11 — HIGH PRIORITY	$###E1C–$###E1E
0	CHANNEL INTERRUPT ENABLE	0 — CHANNEL INTERRUPTS DISABLED 1 — CHANNEL INTERRUPTS ENABLED	$###E0A
0	CHANNEL INTERRUPT STATUS	0 — CHANNEL INTERRUPT NOT ASSERTED 1 — CHANNEL INTERRUPT ASSERTED	$###E20

PARAMETER RAM

	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
$###F(W + 1)0	ACCEL_RATIO_2	ACCEL_RATIO_1
$###F(W + 1)2	ACCEL_RATIO_4	ACCEL_RATIO_3
$###F(W + 1)4	ACCEL_RATIO_6	ACCEL_RATIO_5
$###F(W + 1)6	ACCEL_RATIO_8	ACCEL_RATIO_7
$###F(W + 1)8	ACCEL_RATIO_10	ACCEL_RATIO_9
$###F(W + 1)A	ACCEL_RATIO_12	ACCEL_RATIO_11
$###F(W + 1)C *	ACCEL_RATIO_14 *	ACCEL_RATIO_13 *
⋮	⋮	⋮
$###F(W + 3)A *	ACCEL_RATIO_36 *	ACCEL_RATIO_35 *

* OPTIONAL ADDITIONAL PARAMETERS NOT AVAILABLE IN ALL CASES. REFER TO MOTOROLA PROGRAMMING NOTE TPUPN04 FOR DETAILS.

= WRITTEN BY CPU

= WRITTEN BY CPU AND TPU

= WRITTEN BY TPU

= UNUSED PARAMETERS

W = MASTER CHANNEL NUMBER

TPU TSM SLV

Figure D-5 TSM Parameters — Slave Mode

D.5 Frequency Measurement (FQM)

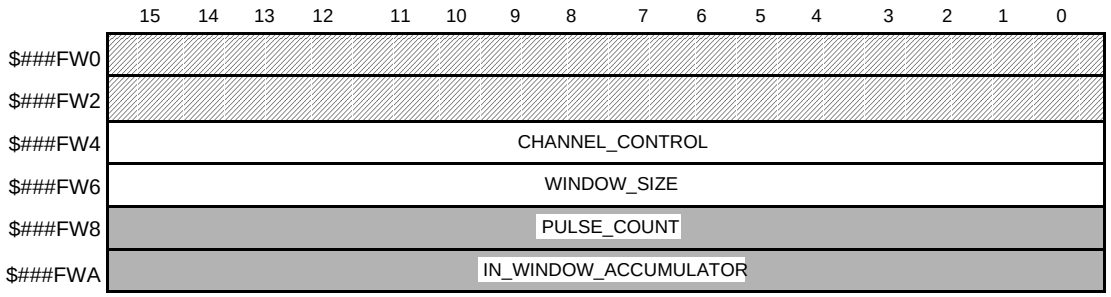
FQM counts the number of input pulses to a TPU channel during a user-defined window period. The function has single shot and continuous modes. No pulses are lost between sample windows in continuous mode. The user selects whether to detect pulses on the rising or falling edge. This function is intended for high speed measurement; measurement of slow pulses with noise rejection can be made with PTA.

Figure D-6 shows all of the host interface areas for the FQM function.

CONTROL BITS

	NAME	OPTIONS	ADDRESSES
0	CHANNEL INTERRUPT ENABLE	0 — CHANNEL INTERRUPTS DISABLED 1 — CHANNEL INTERRUPTS ENABLED	$###E0A
3 2 1 0	CHANNEL FUNCTION SELECT	xxxx — FQD FUNCTION NUMBER (ASSIGNED DURING MICROCODE ASSEMBLY)	$###E0C–$###E12
1 0	HOST SEQUENCE BITS	00 — BEGIN WITH FALLING EDGE, SINGLE-SHOT MODE 01 — BEGIN WITH FALLING EDGE, CONTINUOUS MODE 10 — BEGIN WITH RISING EDGE, SINGLE-SHOT MODE 11 — BEGIN WITH RISING EDGE, CONTINUOUS MODE	$###E14–$###E16
1 0	HOST SERVICE BITS	00 — NO HOST SERVICE (RESET CONDITION) 01 — NOT USED 10 — INITIALIZE 11 — NOT USED	$###E18–$###E1A
1 0	CHANNEL PRIORITY	00 — DISABLED 01 — LOW PRIORITY 10 — MEDIUM PRIORITY 11 — HIGH PRIORITY	$###E1C–$###E1E
0	CHANNEL INTERRUPT STATUS	0 — CHANNEL INTERRUPT NOT ASSERTED 1 — CHANNEL INTERRUPT ASSERTED	$###E20

PARAMETER RAM



= WRITTEN BY CPU

= WRITTEN BY CPU AND TPU

= WRITTEN BY TPU

= UNUSED PARAMETERS

W = PRIMARY CHANNEL NUMBER

TPU FQM CHRT

Figure D-6 FQM Parameters

D.6 Universal Asynchronous Receiver/Transmitter (UART)

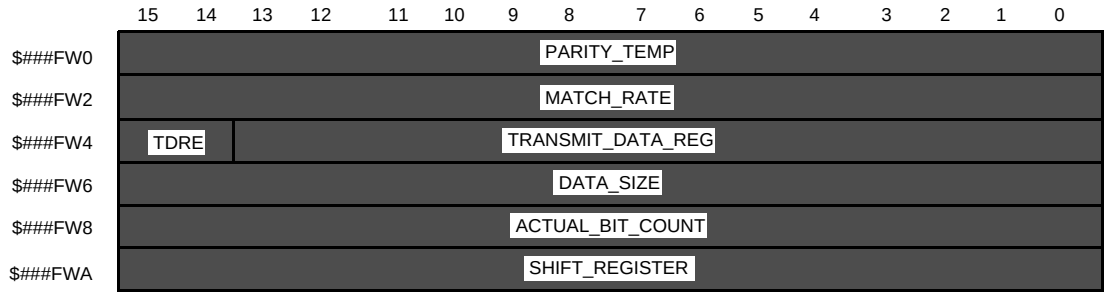
The UART function uses one or two TPU channels to provide asynchronous communications. Data word length is programmable from one to 14 bits. The function supports detection or generation of even, odd, and no parity. Baud rate is freely programmable and can be higher than 100 Kbaud. Eight bi-directional UART channels running in excess of 9600 baud could be implemented on the TPU.

Figure D-7 and **Figure D-4** show all of the host interface areas for the UART function in transmitting and receiving modes, respectively.

CONTROL BITS

	NAME	OPTIONS	ADDRESSES
3 2 1 0	CHANNEL FUNCTION SELECT	UART FUNCTION NUMBER (ASSIGNED DURING MICRO-CODE ASSEMBLY)	$###E0C–$###E12
1 0	HOST SEQUENCE BITS	00 — NO PARITY 01 — NO PARITY 10 — EVEN PARITY 11 — ODD PARITY	$###E14–$###E16
1 0	HOST SERVICE BITS	00 — NOT USED 01 — NOT USED 10 — TRANSMIT 11 — RECEIVE	$###E18–$###E1A
1 0	CHANNEL PRIORITY	00 — DISABLED 01 — LOW PRIORITY 10 — MEDIUM PRIORITY 11 — HIGH PRIORITY	$###E1C–$###E1E
0	CHANNEL INTERRUPT ENABLE	0 — CHANNEL INTERRUPTS DISABLED 1 — CHANNEL INTERRUPTS ENABLED	$###E0A
0	CHANNEL INTERRUPT STATUS	0 — CHANNEL INTERRUPT NOT ASSERTED 1 — CHANNEL INTERRUPT ASSERTED	$###E20

PARAMETER RAM (TRANSMITTER)



= WRITTEN BY CPU

= WRITTEN BY CPU AND TPU

= WRITTEN BY TPU

= UNUSED PARAMETERS

W = PRIMARY CHANNEL NUMBER

TPU UART TRANS

Figure D-7 UART Transmitter Parameters

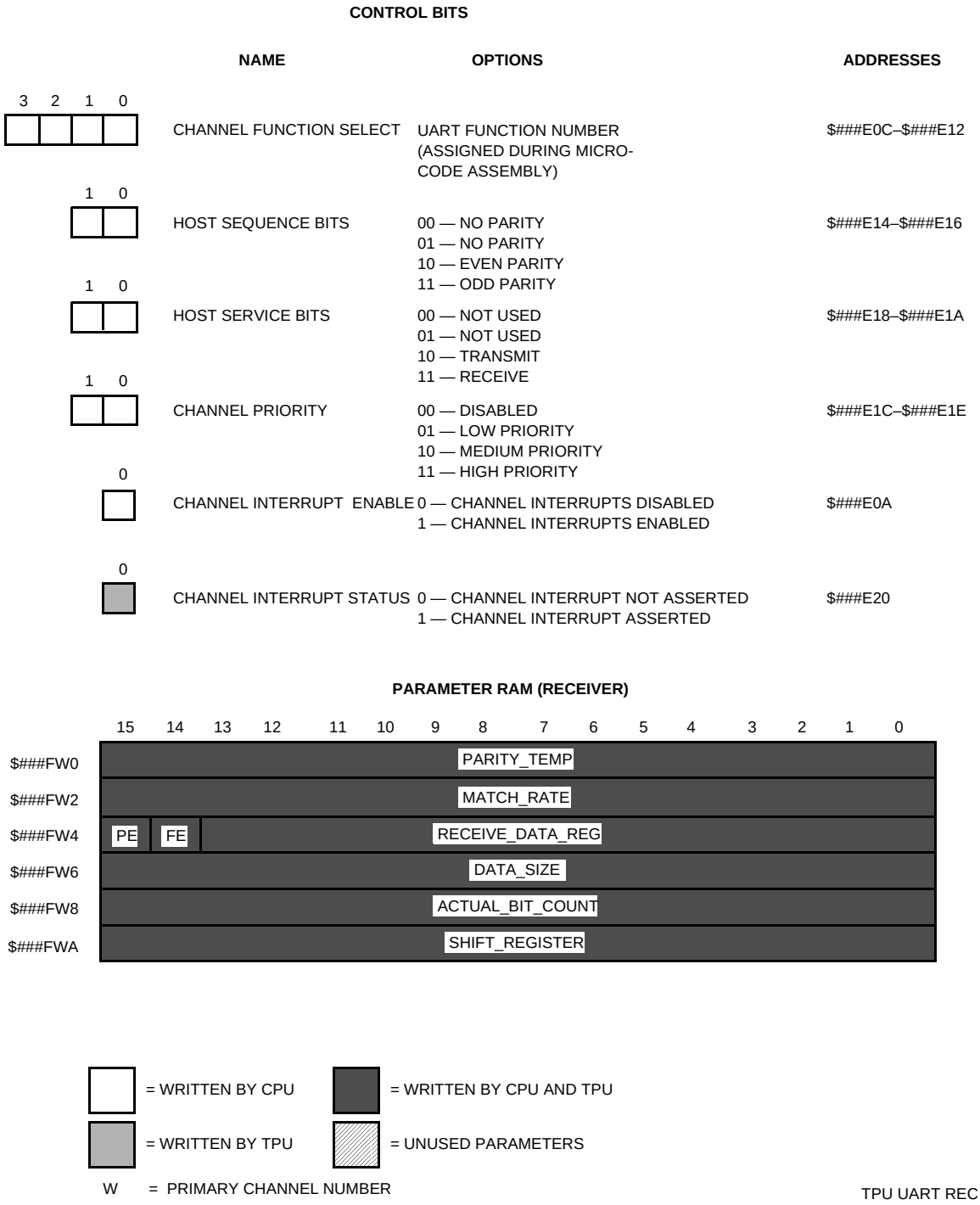


Figure D-8 UART Receiver Parameters

D.7 New Input Capture/Transition Counter (NITC)

Any channel of the TPU can capture the value of a specified TCR or any specified location in parameter RAM upon the occurrence of each transition or specified number of transitions, and then generate an interrupt request to notify the bus master. The times of the most recent two transitions are maintained in parameter RAM. A channel can perform input captures continually, or a channel can detect a single transition or specified number of transitions, ceasing channel activity until reinitialization. After each transition or specified number of transitions, the channel can generate a link to other channels.

Figure D-9 shows all of the host interface areas for the NITC function.

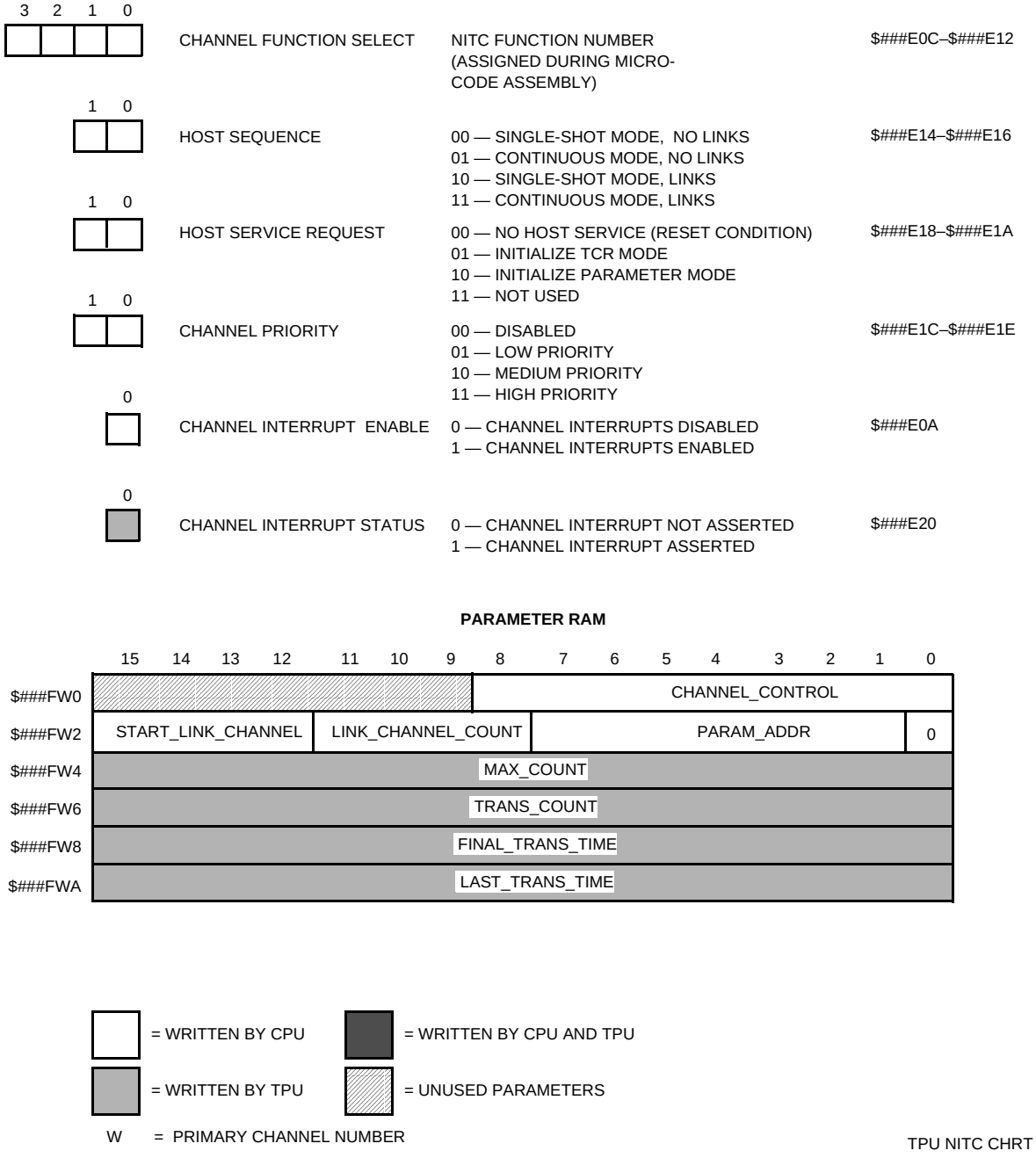


Figure D-9 NITC Parameters

D.8 Multiphase Motor Commutation (COMM)

The COMM function generates the phase commutation signals for a variety of brushless motors, including three-phase brushless direct current. It derives the commutation state directly from the position decoded in FQD, thus eliminating the need for hall effect sensors.

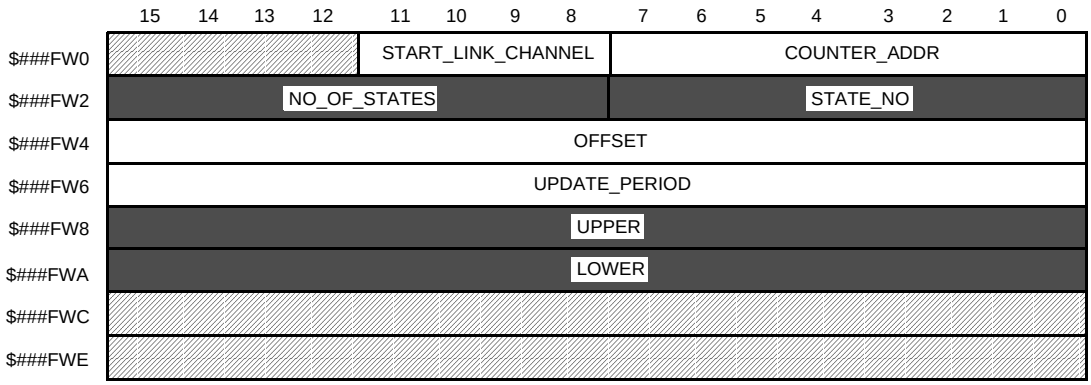
The state sequence is implemented as a user-configurable state machine, thus providing a flexible approach with other general applications. A CPU offset parameter is provided to allow all the switching angles to be advanced or retarded on the fly by the CPU. This feature is useful for torque maintenance at high speeds.

Figure D-10 and **Figure D-11** show all of the host interface areas for the COMM function.

CONTROL BITS

	NAME	OPTIONS	ADDRESSES
0	CHANNEL INTERRUPT ENABLE	0 — CHANNEL INTERRUPTS DISABLED 1 — CHANNEL INTERRUPTS ENABLED	$###E0A
3 2 1 0	CHANNEL FUNCTION SELECT	xxxx — FQD FUNCTION NUMBER (ASSIGNED DURING MICROCODE ASSEMBLY)	$###E0C–$###E12
1 0	HOST SEQUENCE	00 — SENSORLESS MATCH UPDATE MODE 01 — SENSORLESS MATCH UPDATE MODE 10 — SENSORLESS LINK UPDATE MODE 11 — SENSORED MODE	$###E14–$###E16
1 0	HOST SERVICE REQUEST	00 — NO HOST SERVICE (RESET CONDITION) 01 — NOT USED 10 — INITIALIZE OR FORCE STATE 11 — INITIALIZE OR FORCE IMMEDIATE STATE TEST	$###E18–$###E1A
1 0	CHANNEL PRIORITY	00 — DISABLED 01 — LOW PRIORITY 10 — MEDIUM PRIORITY 11 — HIGH PRIORITY	$###E1C–$###E1E
0	CHANNEL INTERRUPT STATUS	0 — CHANNEL INTERRUPT NOT ASSERTED 1 — CHANNEL INTERRUPT ASSERTED	$###E20

PARAMETER RAM



W = MASTER COMM CHANNEL NUMBER

TPU COMM CHRT 1

Figure D-10 COMM Parameters, Part 1 of 2

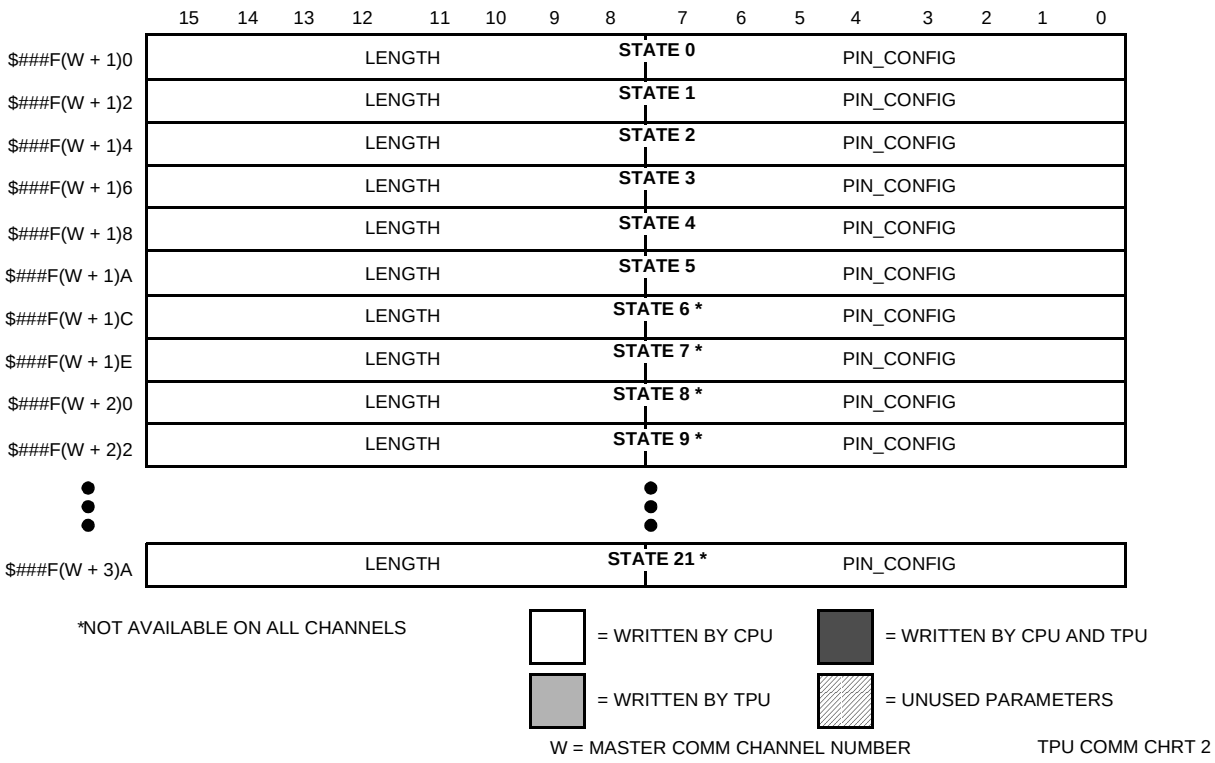


Figure D-11 COMM Parameters, Part 2 of 2

D.9 Hall Effect Decode (HALLD)

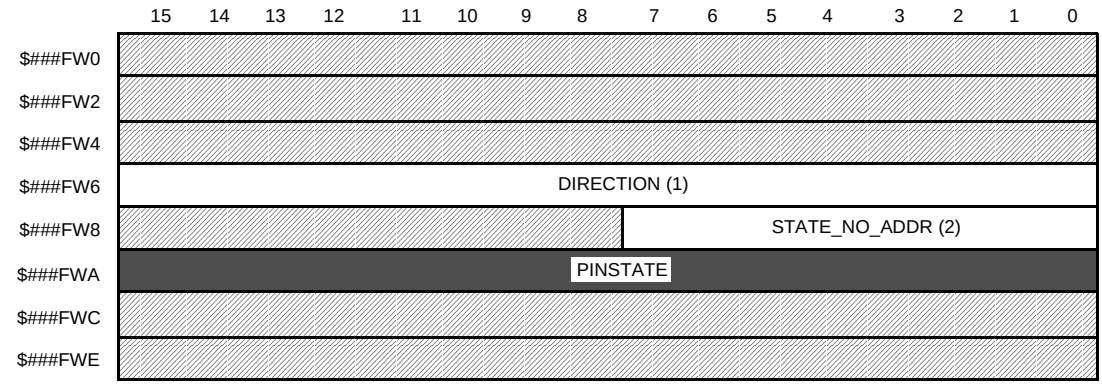
The HALLD function decodes the sensor signals from a brushless motor, along with a direction input from the CPU, into a state number. The function supports two- or three-sensor decoding. The decoded state number is written into a COMM channel, which outputs the required commutation drive signals. In addition to brushless motor applications, the function can have more general applications, such as decoding “option” switches.

Figure D-12 shows all of the host interface areas for the HALLD function.

CONTROL BITS

	NAME	OPTIONS	ADDRESSES
0	CHANNEL INTERRUPT ENABLE	0 — CHANNEL INTERRUPTS DISABLED 1 — CHANNEL INTERRUPTS ENABLED	$###E0A
3 2 1 0	CHANNEL FUNCTION SELECT	xxxx — FQD FUNCTION NUMBER (ASSIGNED DURING MICROCODE ASSEMBLY)	$###E0C–$###E12
1 0	HOST SEQUENCE	00 — CHANNEL A 01 — CHANNEL B 10 — CHANNEL B 11 — CHANNEL C (3-CHANNEL MODE ONLY)	$###E14–$###E16
1 0	HOST SERVICE REQUEST	00 — NO HOST SERVICE (RESET CONDITION) 01 — NOT USED 10 — INITIALIZE, 2-CHANNEL MODE 11 — INITIALIZE, 3-CHANNEL MODE	$###E18–$###E1A
1 0	CHANNEL PRIORITY	00 — DISABLED 01 — LOW PRIORITY 10 — MEDIUM PRIORITY 11 — HIGH PRIORITY	$###E1C–$###E1E
0	CHANNEL INTERRUPT STATUS	x — NOT USED	$###E20

PARAMETER RAM



= WRITTEN BY CPU

= WRITTEN BY CPU AND TPU

= WRITTEN BY TPU

= UNUSED PARAMETERS

W = CHANNEL NUMBER

NOTES:
1. CHANNEL A ONLY.
2. 1 CHANNEL ONLY (CHANNEL B IN 2-CHANNEL MODE, CHANNEL C IN 3-CHANNEL MODE).

TPU HALLD CHRT

Figure D-12 HALLD Parameters

D.10 Multichannel Pulse-Width Modulation (MCPWM)

MCPWM generates pulse-width modulated outputs with full 0% to 100% duty cycle range independent of other TPU activity. This capability requires two TPU channels plus an external gate for one PWM channel. (A simple one-channel PWM capability is supported by the QOM function.)

Multiple PWMs generated by MCPWM have two types of high time alignment: edge aligned and center aligned. Edge-aligned mode uses n + 1 TPU channels for n PWMs; center-aligned mode uses 2n + 1 channels. Center-aligned mode allows a user-defined “dead time” to be specified so that two PWMs can be used to drive an H-bridge without destructive current spikes. This feature is important for motor control applications.

Figure D-13 through **Figure D-18** show the host interface areas for the MCPWM function in each mode.

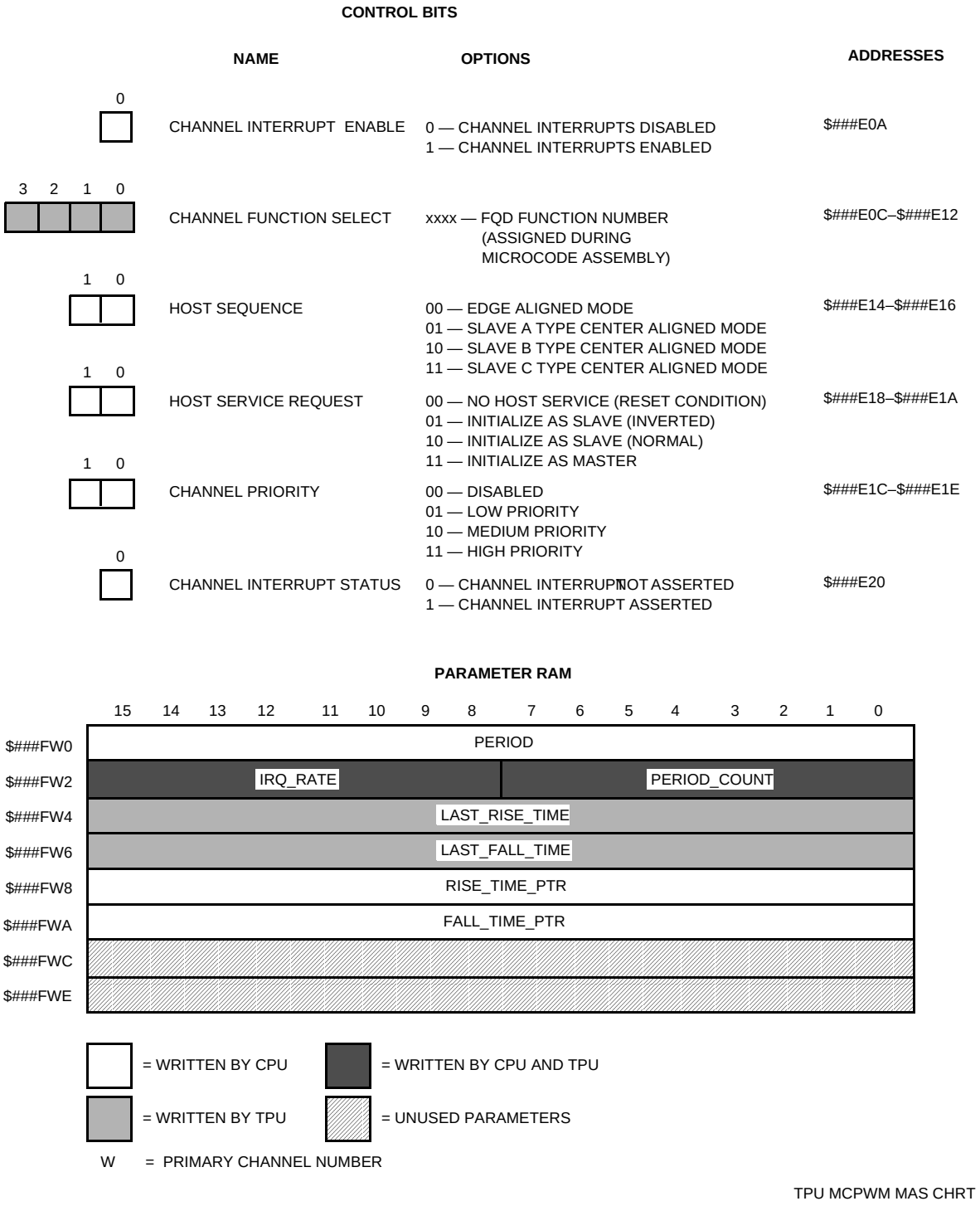


Figure D-13 MCPWM Parameters — Master Mode

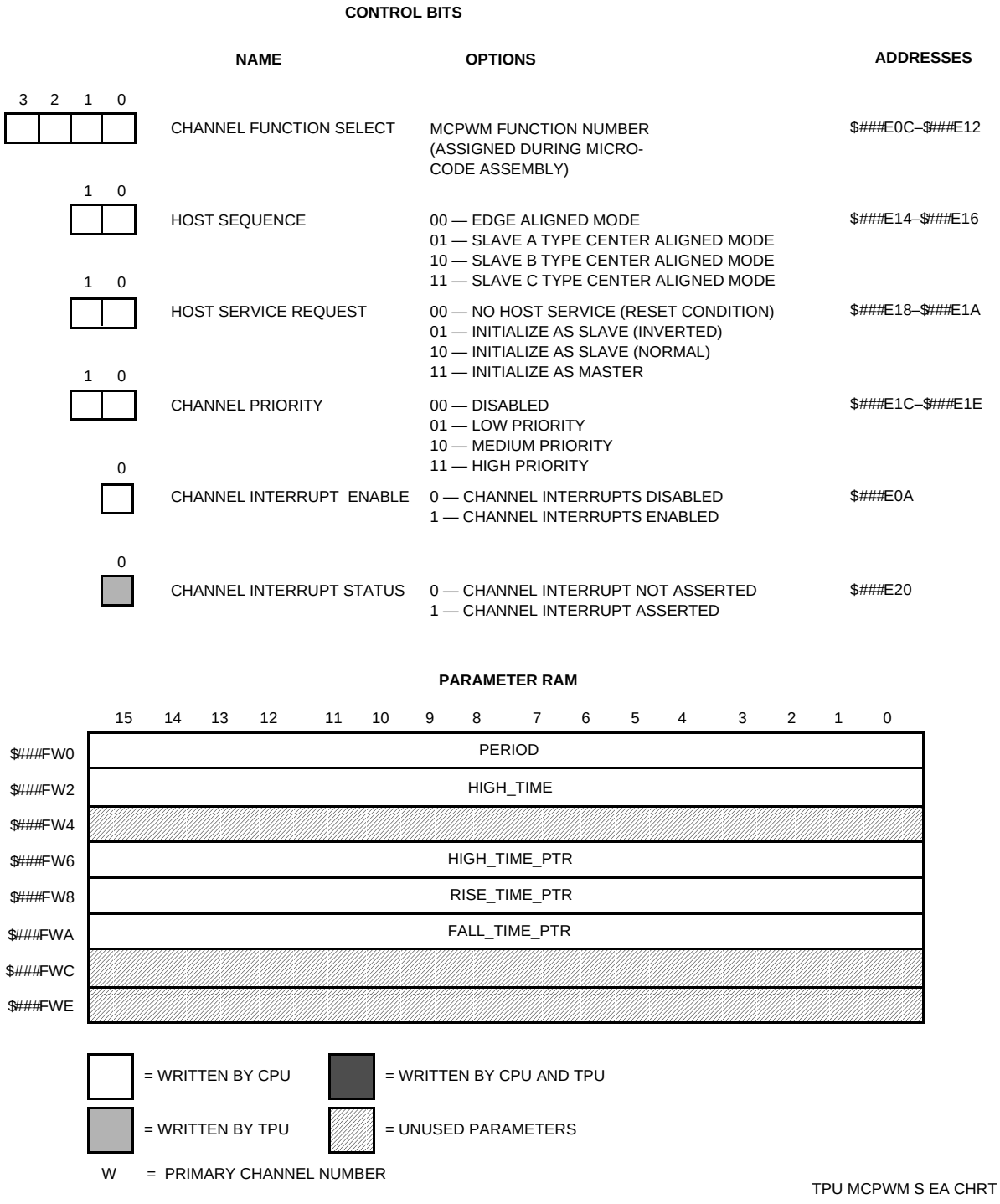


Figure D-14 MCPWM Parameters — Slave Edge-Aligned Mode

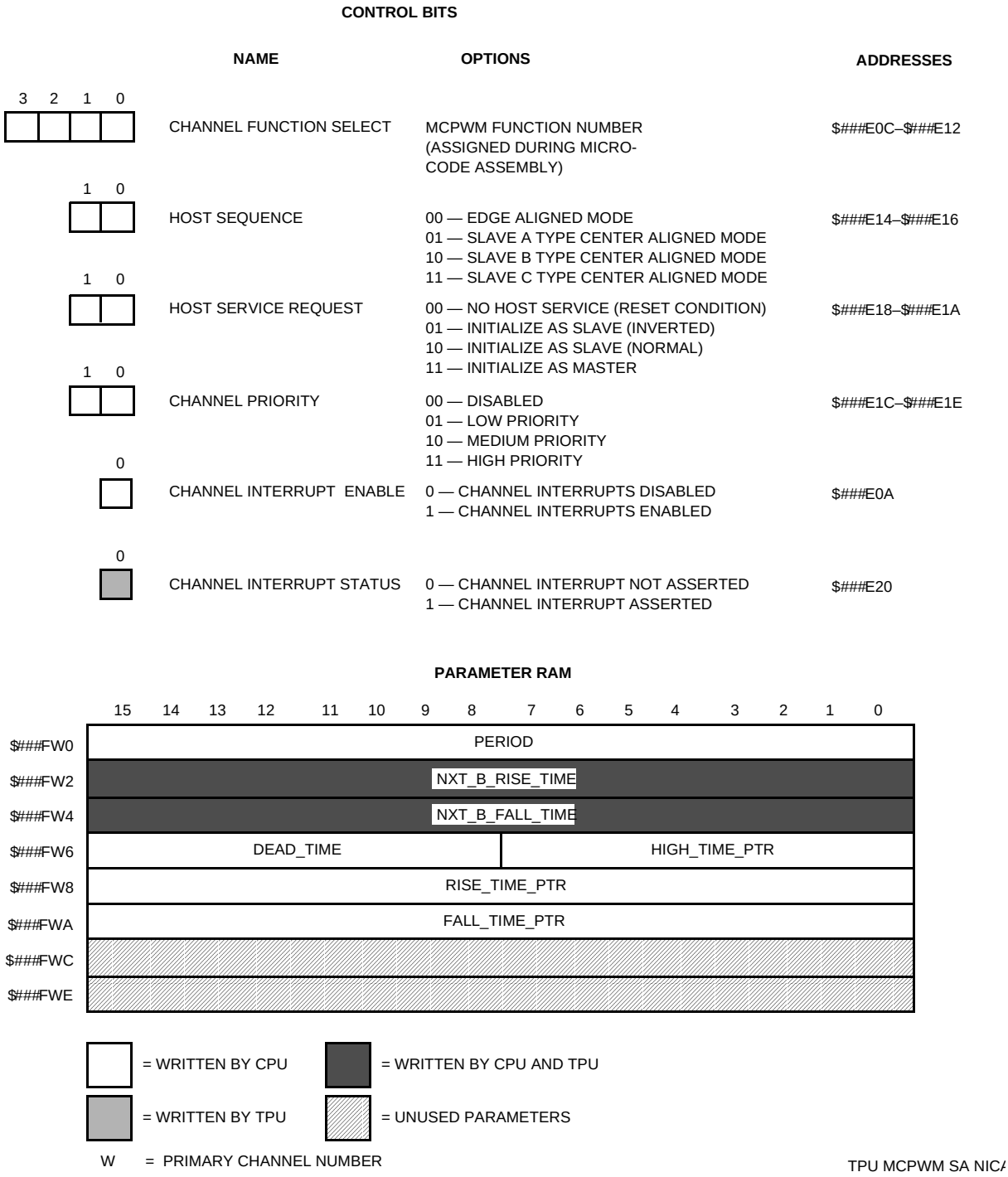


Figure D-15 MCPWM Parameters — Slave Ch A Non-Inverted Center-Aligned Mode

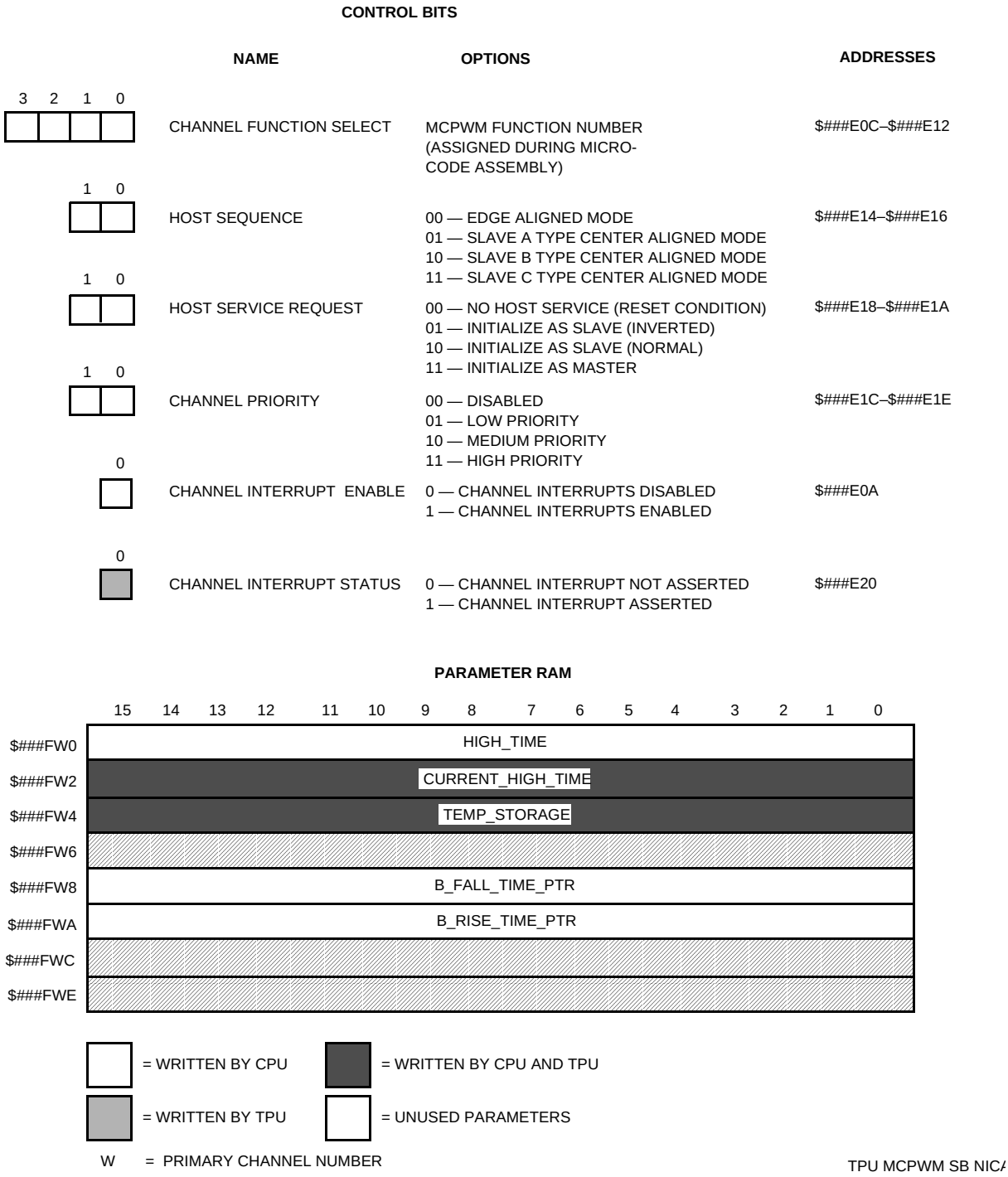


Figure D-16 MCPWM Parameters — Slave Ch B Non-Inverted Center-Aligned Mode

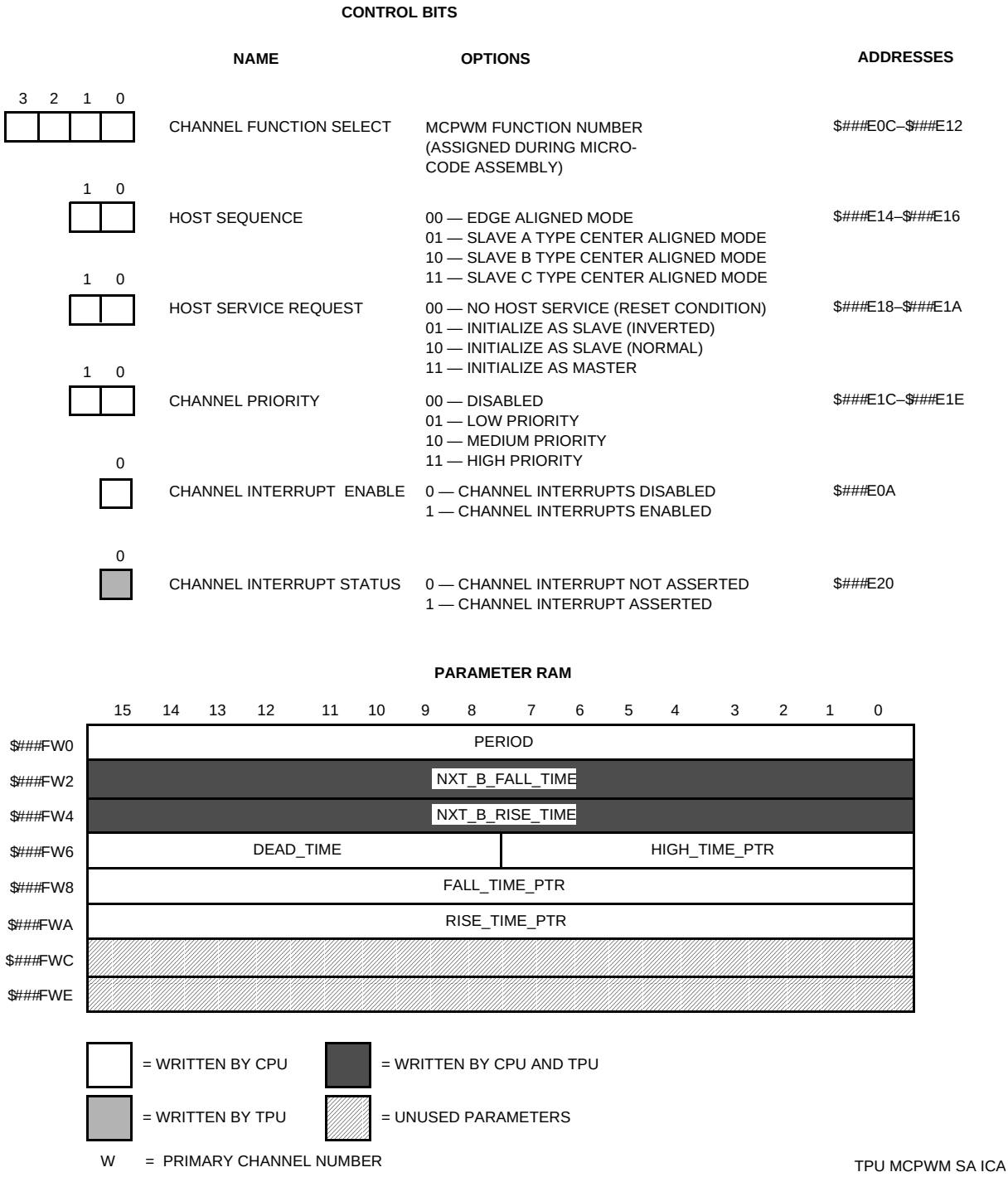


Figure D-17 MCPWM Parameters — Slave Ch A Inverted Center-Aligned Mode

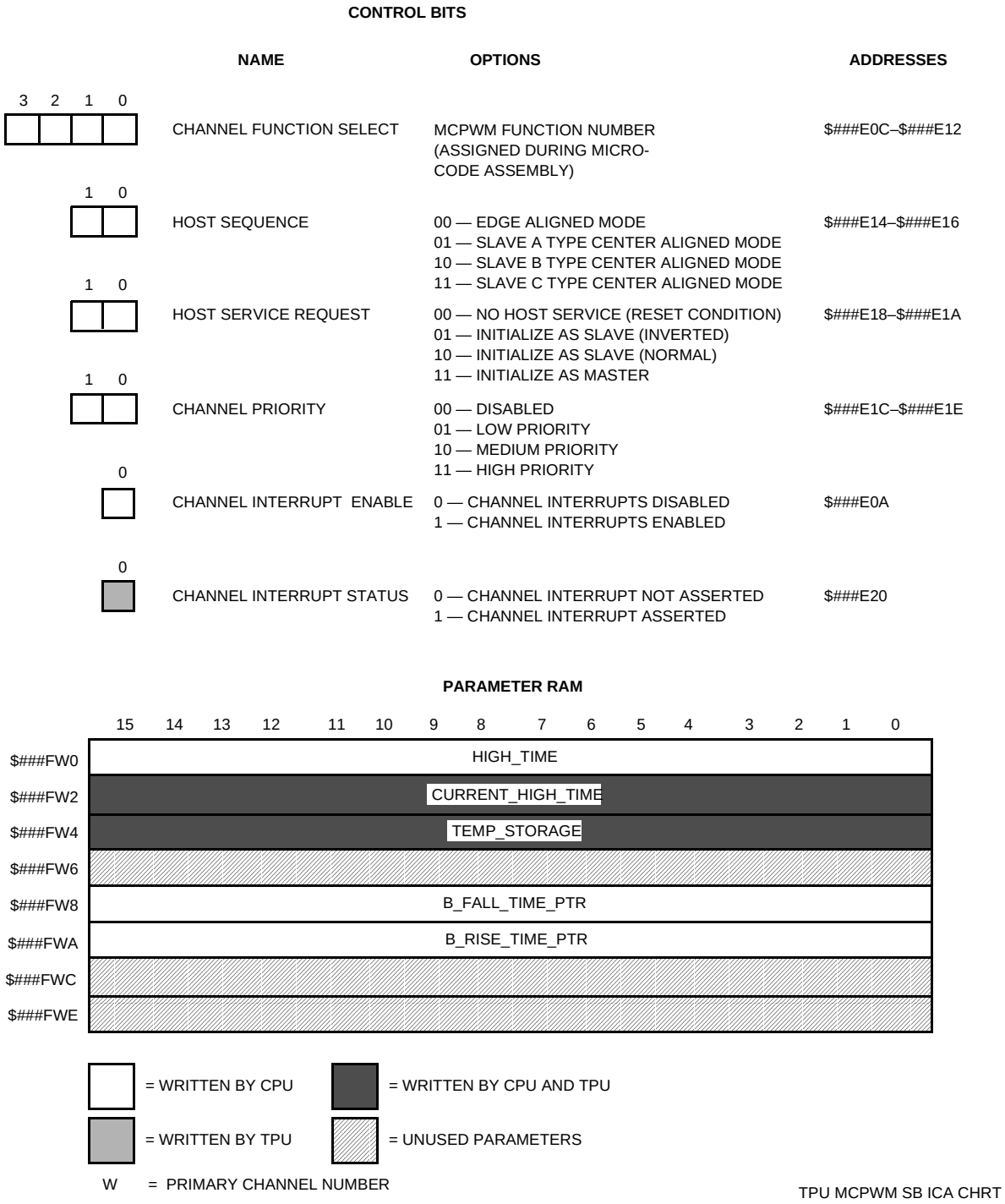


Figure D-18 MCPWM Parameters — Slave Ch B Inverted Center-Aligned Mode

D.11 Fast Quadrature Decode TPU Function (FQD)

FQD is a position feedback function for motor control. It decodes the two signals from a slotted encoder to provide the CPU with a 16-bit free-running position counter. FQD incorporates a “speed switch” which disables one of the channels at high speed, allowing faster signals to be decoded. A time stamp is provided on every counter update to allow position interpolation and better velocity determination at low speed or when low resolution encoders are used. The third index channel provided by some encoders is handled by the ITC function.

Figure D-19 and **Figure D-20** show the host interface areas for the FQD function for primary and secondary channels, respectively.

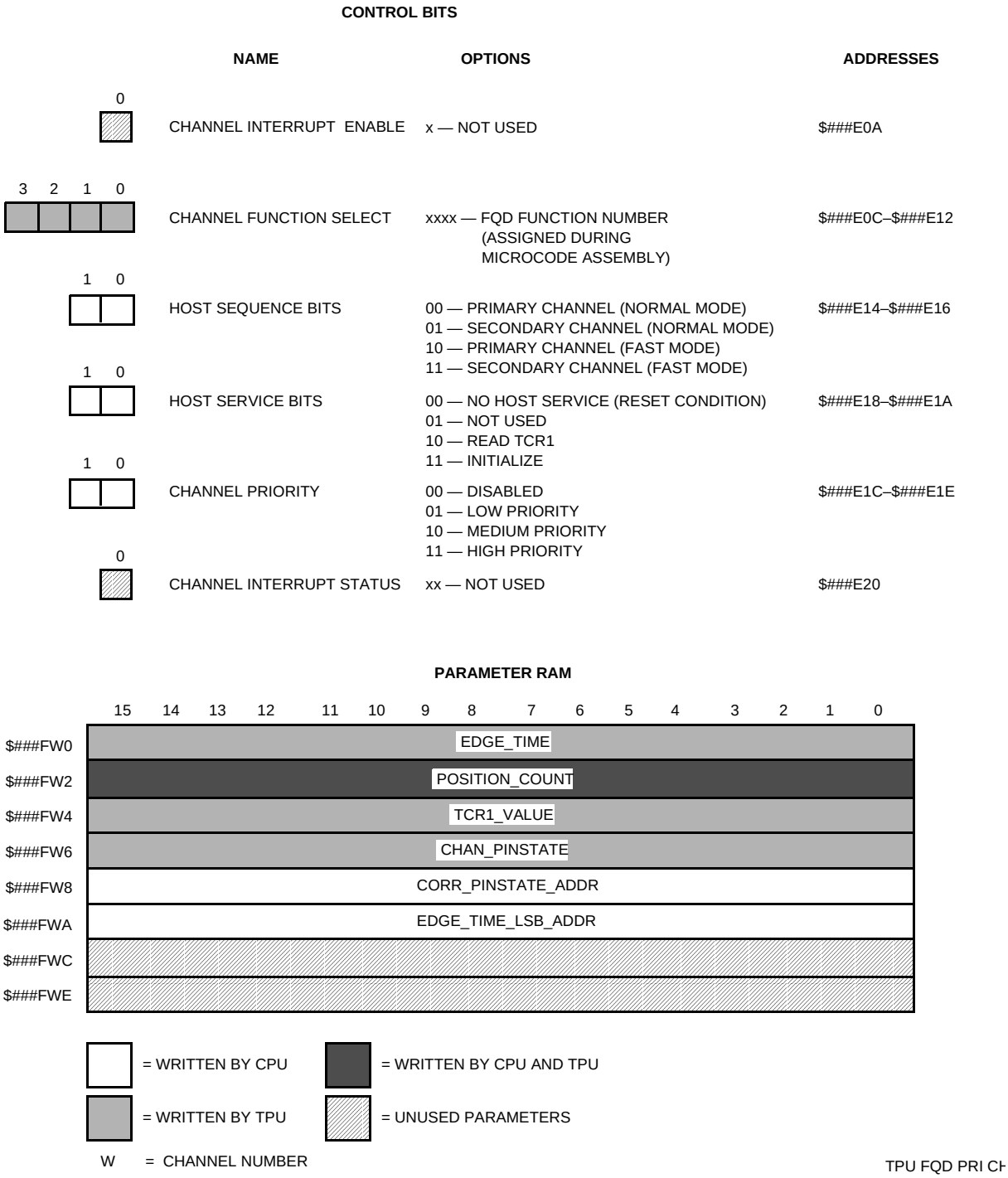


Figure D-19 FQD Parameters — Primary Channel

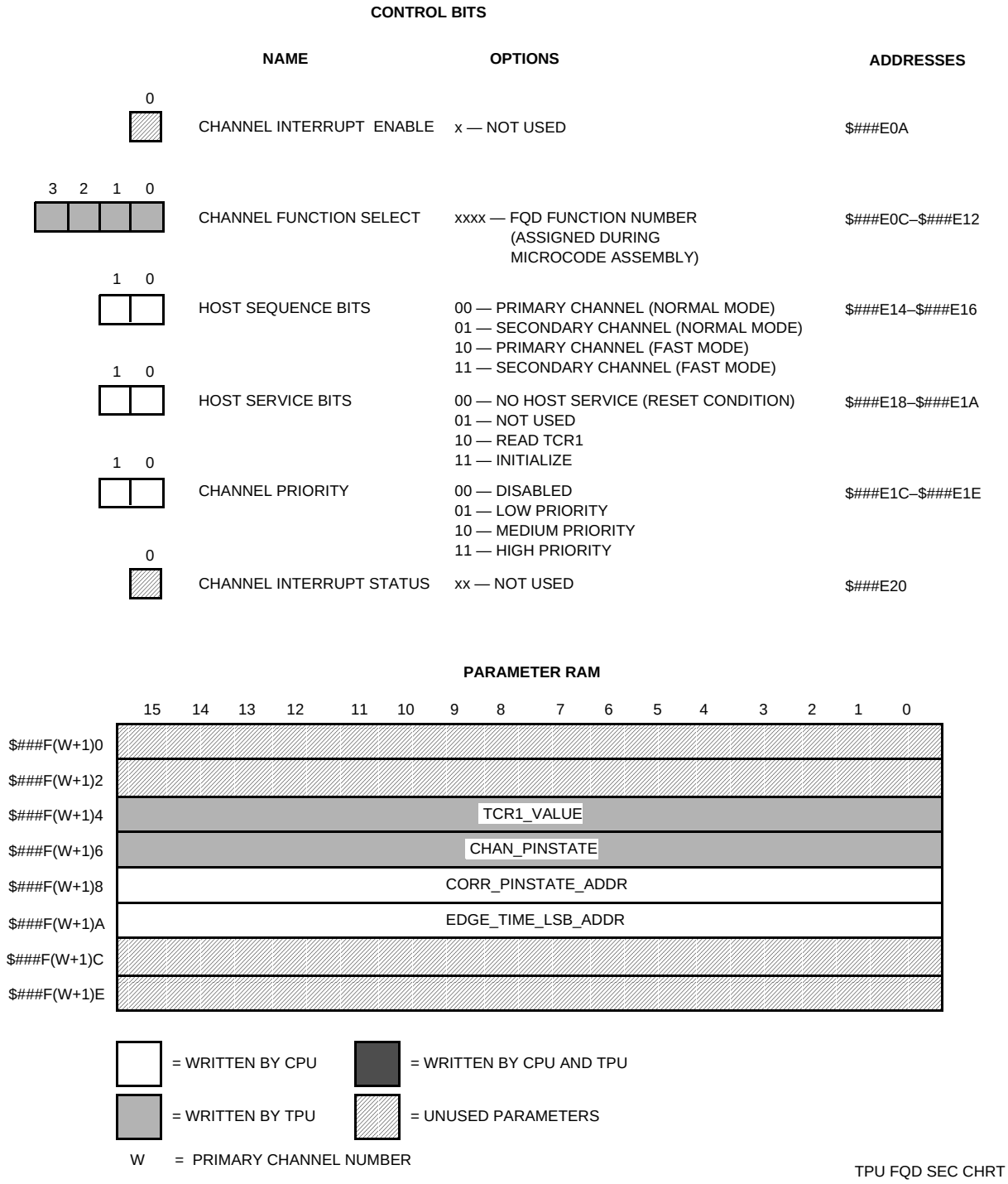


Figure D-20 FQD Parameters — Secondary Channel

D.12 Period/Pulse-Width Accumulator (PPWA)

The period/pulse-width accumulator (PPWA) algorithm accumulates a 16-bit or 24-bit sum of either the period or the pulse width of an input signal over a programmable number of periods or pulses (from one to 255). After an accumulation period, the algorithm can generate a link to a sequential block of up to eight channels. The user specifies a starting channel of the block and number of channels within the block. Generation of links depends on the mode of operation.

Any channel can be used to measure an accumulated number of periods of an input signal. A maximum of 24 bits can be used for the accumulation parameter. From one to 255 period measurements can be made and summed with the previous measurement(s) before the TPU interrupts the CPU, allowing instantaneous or average frequency measurement, and the latest complete accumulation (over the programmed number of periods).

The pulse width (high-time portion) of an input signal can be measured (up to 24 bits) and added to a previous measurement over a programmable number of periods (one to 255). This provides an instantaneous or average pulse-width measurement capability, allowing the latest complete accumulation (over the specified number of periods) to always be available in a parameter.

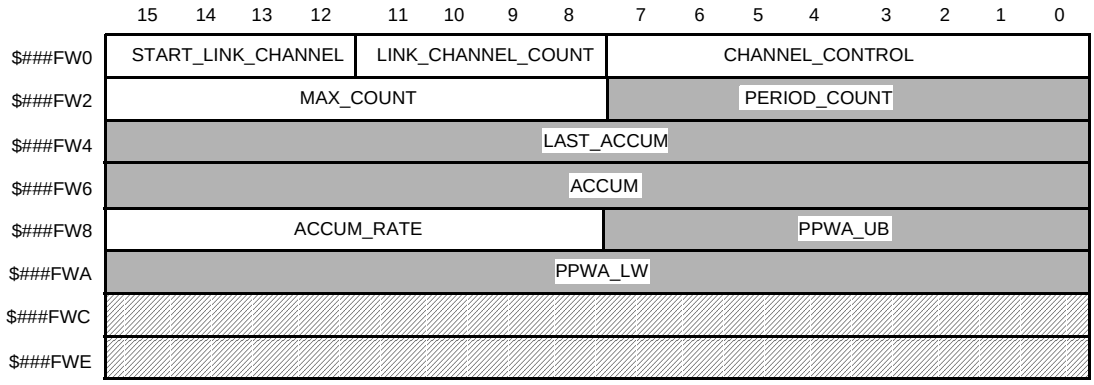
By using the output compare function in conjunction with PPWA, an output signal can be generated that is proportional to a specified input signal. The ratio of the input and output frequency is programmable. One or more output signals with different frequencies, yet proportional and synchronized to a single input signal, can be generated on separate channels.

Figure D-21 shows the host interface areas and parameter RAM for the PPWA function.

CONTROL BITS

	NAME	OPTIONS	ADDRESSES
3 2 1 0	CHANNEL FUNCTION SELECT	PPWA FUNCTION NUMBER (ASSIGNED DURING MICRO-CODE ASSEMBLY)	$###E0C–$###E12
1 0	CHANNEL PRIORITY	00 — CHANNEL DISABLED 01 — LOW PRIORITY 10 — MEDIUM PRIORITY 11 — HIGH PRIORITY	$###E1C–$###E1E
1 0	HOST SEQUENCE BITS	00 — ACCUMULATE 24-BIT PERIODS, NO LINKS 01 — ACCUMULATE 16-BIT PERIODS, LINKS 10 — ACCUMULATE 24-BIT PULSE WIDTHS, NO LINKS 11 — ACCUMULATE 16-BIT PULSE WIDTHS, LINKS	$###E14–$###E16
1 0	HOST SERVICE BITS	00 — NOT USED 01 — NOT USED 10 — INITIALIZE 11 — NOT USED	$###E18–$###E1A
0	INTERRUPT ENABLE	0 — INTERRUPT NOT ASSERTED 1 — INTERRUPT ASSERTED	$###E0A
0	INTERRUPT STATUS		$###E20

PARAMETER RAM



= WRITTEN BY CPU

= WRITTEN BY CPU AND TPU

= WRITTEN BY TPU

= UNUSED PARAMETERS

W = CHANNEL NUMBER

NOTES:
1. THE TPU DOES NOT CHECK THE VALUE OF LINK_CHANNEL_COUNT. IF THIS PARAMETER IS **NOT** > 0 AND ≤ 8, RESULTS ARE UNPREDICTABLE.
2. MAX_COUNT MAY BE WRITTEN AT ANY TIME BY THE HOST CPU, BUT IF THE VALUE WRITTEN IS ≤ PERIOD_COUNT, A PERIOD OR PULSE-WIDTH ACCUMULATION IS TERMINATED. IF THIS HAPPENS, THE NUMBER OF PERIODS OVER WHICH THE ACCUMULATION IS DONE WILL NOT CORRESPOND TO MAX_COUNT.

1049A

Figure D-21 PPWA Parameters

D.13 Output Compare (OC)

The output compare (OC) function generates a rising edge, falling edge, or a toggle of the previous edge in one of three ways:

1. Immediately upon CPU initiation, thereby generating a pulse with a length equal to a programmable delay time
2. At a programmable delay time from a user-specified time
3. Continuously. Upon receiving a link from a channel, OC references, without CPU interaction, a specifiable period and calculates an offset:

$$\text{Offset} = \text{Period} \times \text{Ratio}$$

where RATIO is a parameter supplied by the user.

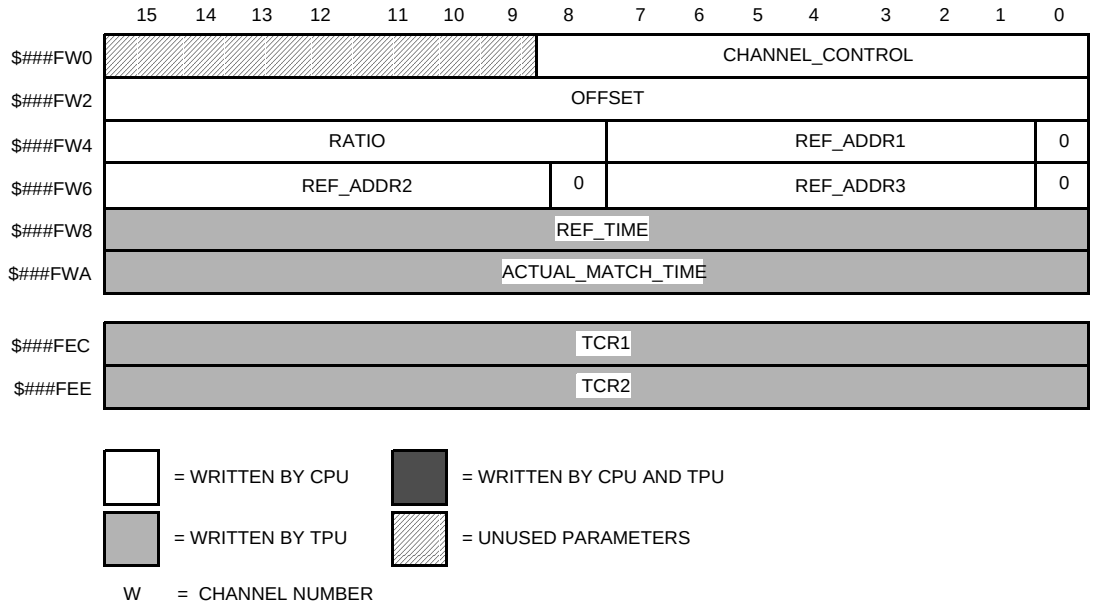
This algorithm generates a 50% duty-cycle continuous square wave with each high/low time equal to the calculated OFFSET. Due to offset calculation, there is an initial link time before continuous pulse generation begins.

Figure D-22 shows the host interface areas and parameter RAM for the OC function.

CONTROL BITS

	NAME	OPTIONS	ADDRESSES
3 2 1 0	CHANNEL FUNCTION SELECT	OC FUNCTION NUMBER (ASSIGNED DURING MICRO-CODE ASSEMBLY)	$###E0C–$###E12
1 0	CHANNEL PRIORITY	00 — CHANNEL DISABLED 01 — LOW PRIORITY 10 — MEDIUM PRIORITY 11 — HIGH PRIORITY	$###E1C–$###E1E
1 0	HOST SEQUENCE BITS	0x — MATCHES AND PULSES SCHEDULED 1x — ONLY READ TCR1, TCR2	$###E14–$###E16
1 0	HOST SERVICE BITS	00 — NO HOST SERVICE REQUEST 01 — HOST-INITIATED PULSE 10 — NOT USED 11 — INITIALIZE, CONTINUOUS PULSES	$###E18–$###E1A
0	INTERRUPT ENABLE	0 — INTERRUPT NOT ASSERTED 1 — INTERRUPT ASSERTED	$###E0A
0	INTERRUPT STATUS		$###E20

PARAMETER RAM



1024A

Figure D-22 OC Parameters

D.14 Pulse-Width Modulation (PWM)

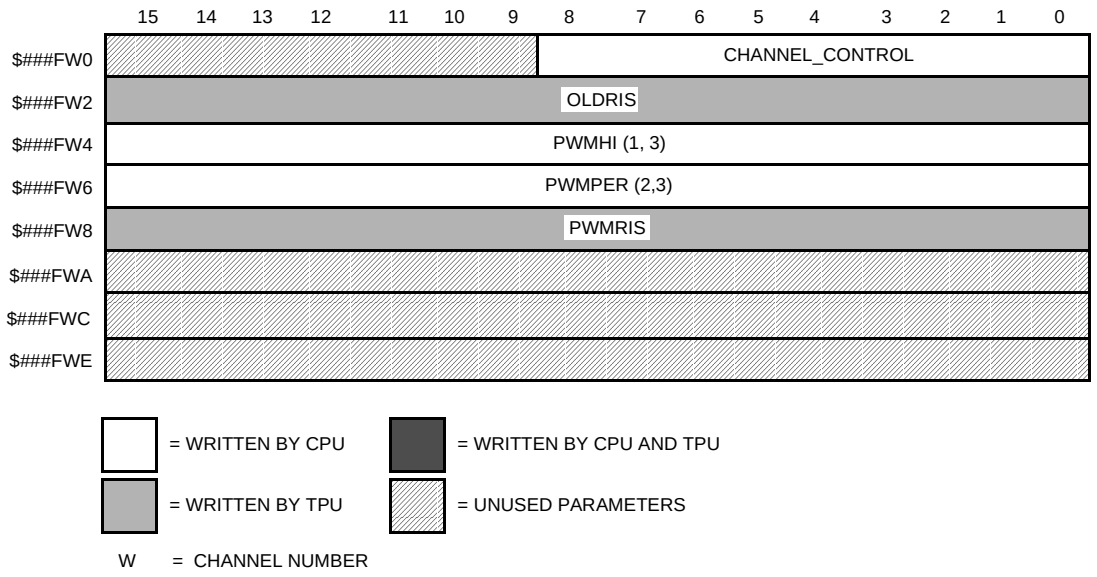
The TPU can generate a pulse-width modulation (PWM) waveform with any duty cycle from zero to 100% (within the resolution and latency capability of the TPU). To define the PWM, the CPU provides one parameter that indicates the period and another parameter that indicates the high time. Updates to one or both of these parameters can direct the waveform change to take effect immediately, or coherently beginning at the next low-to-high transition of the pin.

Figure D-23 shows the host interface areas and parameter RAM for the PWM function.

CONTROL BITS

	NAME	OPTIONS	ADDRESSES
3 2 1 0	CHANNEL FUNCTION SELECT	PWM FUNCTION NUMBER (ASSIGNED DURING MICRO-CODE ASSEMBLY)	$###E0C–$###E12
1 0	CHANNEL PRIORITY	00 — DISABLE 01 — LOW PRIORITY 10 — MEDIUM PRIORITY 11 — HIGH PRIORITY	$###E1C–$###E1E
1 0	HOST SEQUENCE BITS	xx — NOT USED	$###E14–$###E16
1 0	HOST SERVICE BITS	00 — NOT USED 01 — IMMEDIATE UPDATE OF PWM 10 — INITIALIZE 11 — NOT USED	$###E18–$###E1A
0	INTERRUPT ENABLE	0 — INTERRUPT NOT ASSERTED 1 — INTERRUPT ASSERTED	$###E0A
0	INTERRUPT STATUS		$###E20

PARAMETER RAM



NOTES:
1. BEST-CASE MINIMUM FOR PWMHI IS 32 SYSTEM CLOCK CYCLES.
2. BEST-CASE MINIMUM FOR PWMPER IS 48 SYSTEM CLOCK CYCLES.
3. PWMHI AND PWMPER MUST BE ACCESSED COHERENTLY.

1027A

Figure D-23 PWM Parameters

D.15 Discrete Input/Output (DIO)

The DIO function allows a TPU channel to be used as a digital I/O pin.

When a pin is used as a discrete input, a parameter indicates the current input level and the previous 15 levels of a pin. Bit 15, the most significant bit of the parameter, indicates the most recent state. Bit 14 indicates the next most recent state, and so on. The programmer can choose one of the three following conditions to update the parameter:

1. when a transition occurs
2. when the CPU makes a request, or
3. when a rate specified in another parameter is matched

When a pin is used as a discrete output, it is set high or low only upon request by the CPU.

Figure D-24 shows the host interface areas for the DIO function.

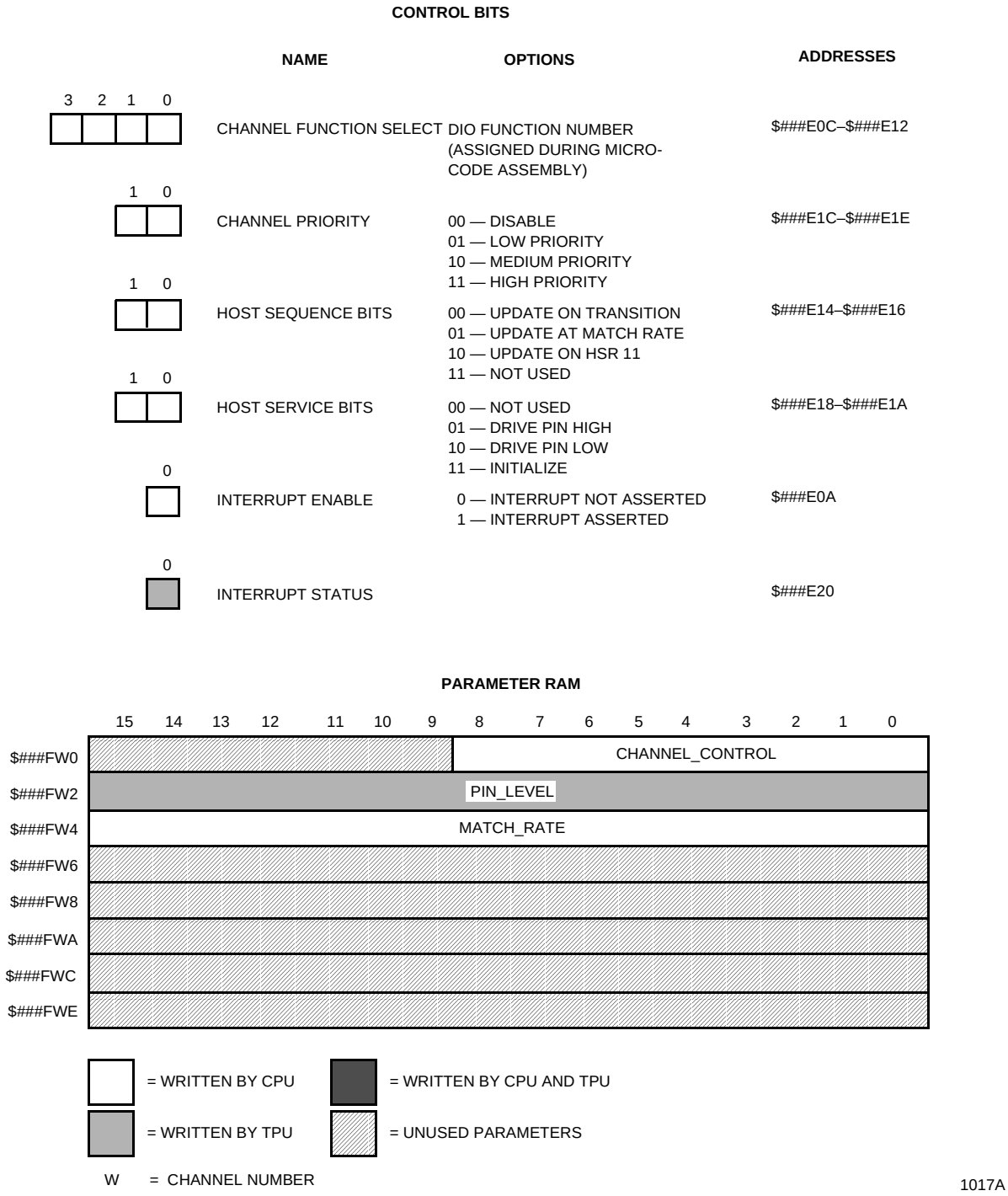


Figure D-24 DIO Parameters

D.16 Synchronized Pulse-Width Modulation (SPWM)

The SPWM function generates a pulse-width modulated waveform (PWM). The CPU can change the period or high time of the waveform at any time. Three different operating modes allow the function to maintain complex timing relationships between channels without CPU intervention.

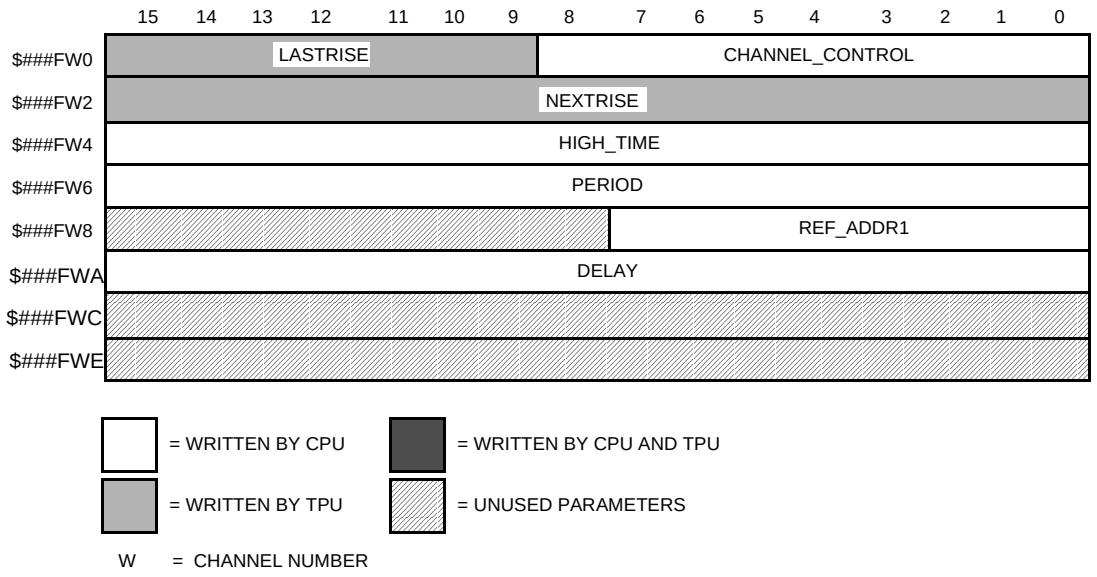
The SPWM output waveform duty cycle excludes 0% and 100%. If a PWM does not need to maintain a time relationship to another PWM, the PWM function should be used instead.

Figure D-25 and **Figure D-26** show all of the host interface areas for the SPWM function.

CONTROL BITS

	NAME	OPTIONS	ADDRESSES
3 2 1 0	CHANNEL FUNCTION SELECT	SPWM FUNCTION NUMBER (ASSIGNED DURING MICRO-CODE ASSEMBLY)	$###E0C–$###E12
1 0	CHANNEL PRIORITY	00 — DISABLE 01 — LOW PRIORITY 10 — MEDIUM PRIORITY 11 — HIGH PRIORITY	$###E1C–$###E1E
1 0	HOST SEQUENCE BITS	00 — MODE 0 01 — MODE 1 10 — MODE 2 11 — NOT USED	$###E14–$###E16
1 0	HOST SERVICE BITS	00 — NO HOST SERVICE REQUEST 01 — NOT USED 10 — INITIALIZE 11 — IMMEDIATE UPDATE (MODE 1)	$###E18–$###E1A
0	INTERRUPT ENABLE	0 — INTERRUPT NOT ASSERTED 1 — INTERRUPT ASSERTED	$###E0A
0	INTERRUPT STATUS		$###E20

PARAMETER RAM (MODE 0)



1030A-1

Figure D-25 SPWM Parameters, Part 1 of 2

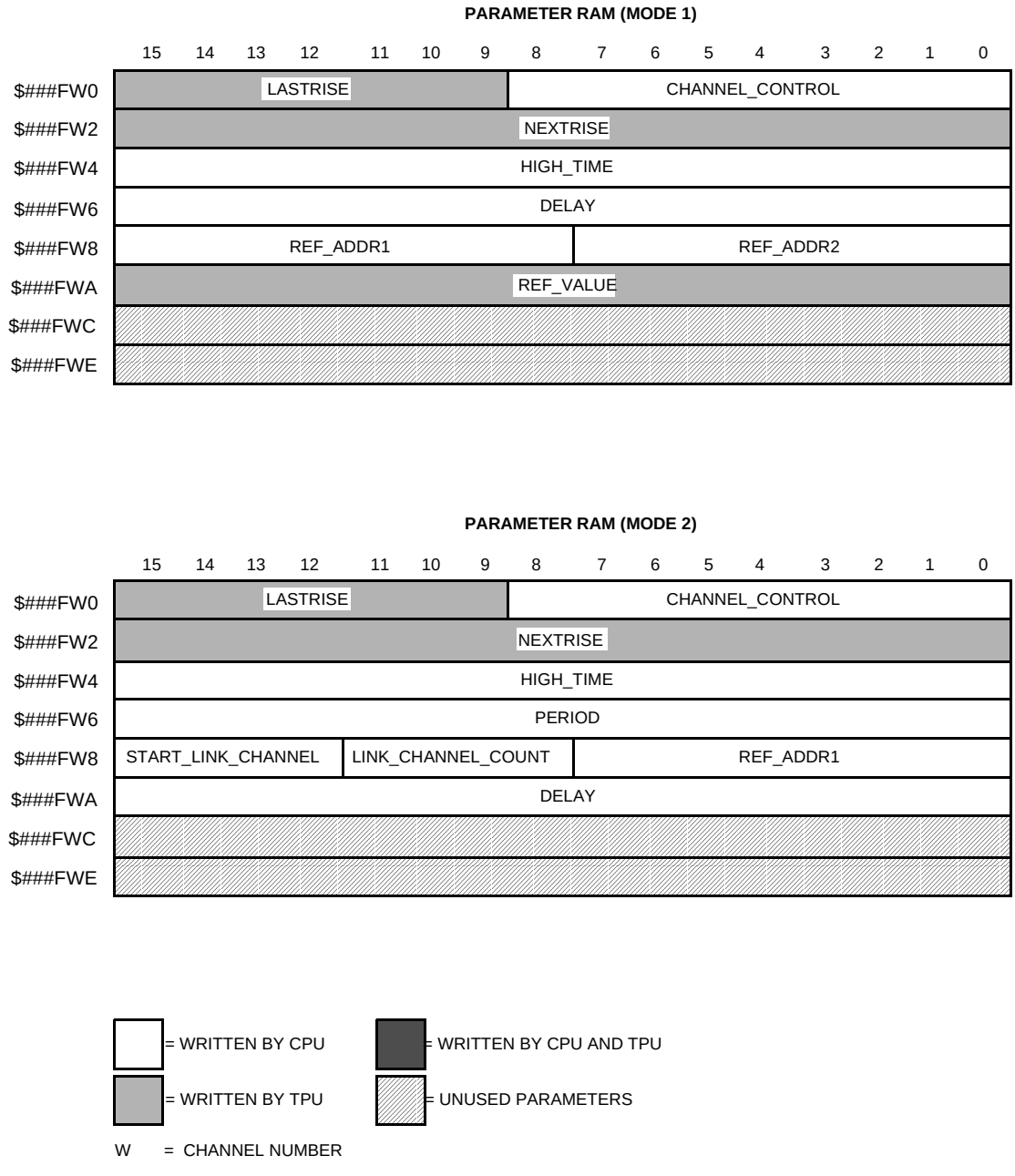


Figure D-26 SPWM Parameters, Part 2 of 2

D.17 Serial Input/Output Port (SIOP)

The serial input/output port (SIOP) TPU function uses two or three TPU channels to form a uni- or bi-directional synchronous serial port that can be used to communicate with a wide variety of devices. Features such as baud rate and transfer size are user programmable. The function can also produce a clock-only, when it uses just one channel.

The SIOP TPU function has been designed to closely resemble the SIOP hardware port found on some Motorola MCUs and can be used to add serial capabilities to a device without a serial port, or extend the capabilities of one with a hardware synchronous port.

SIOP operates in master mode (i.e., the TPU always generates the clock) and the following features are programmable by the user:

1. Choice of clock-only (one channel), clock + transmit (two channels), clock + receive (two channels) or clock + transmit + receive (three channels) operating modes
2. Baud rate period is freely programmable by the user over a 15-bit range of TCR1 counts
3. Selection of msb or lsb first shift direction
4. Variable transfer size from one to 16 bits
5. Clock polarity is programmable

When a transfer of data is complete the SIOP function notifies the host CPU by issuing an interrupt request. The arrangement of the multiple SIOP channels is fixed: the data out channel is the channel above the clock channel and the data in channel is the channel below the clock channel. In clock-only or uni-directional mode, the unused TPU channels are free to run other TPU functions. Two possible SIOP configurations are show in **Figure D-27**.

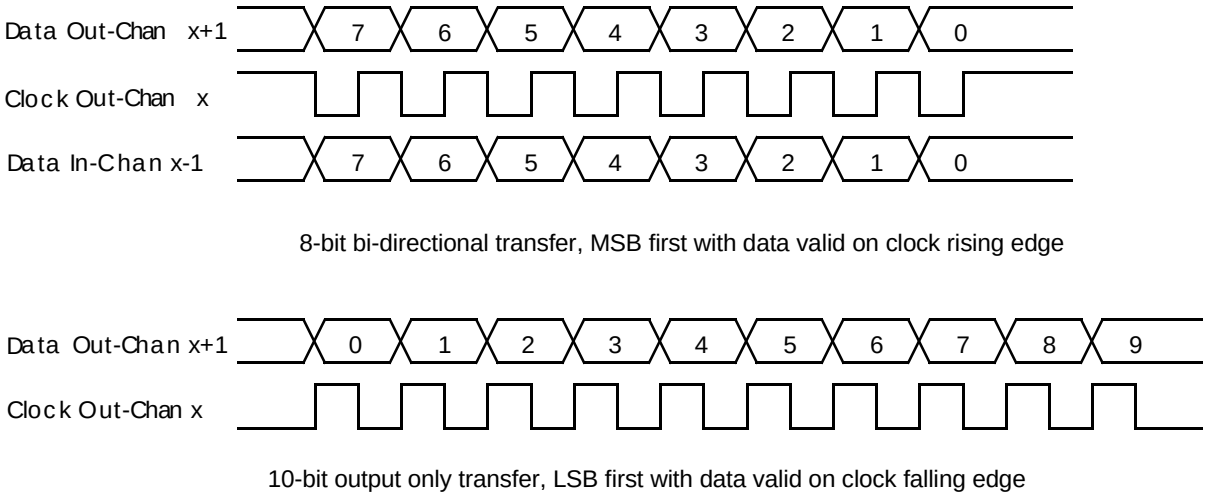
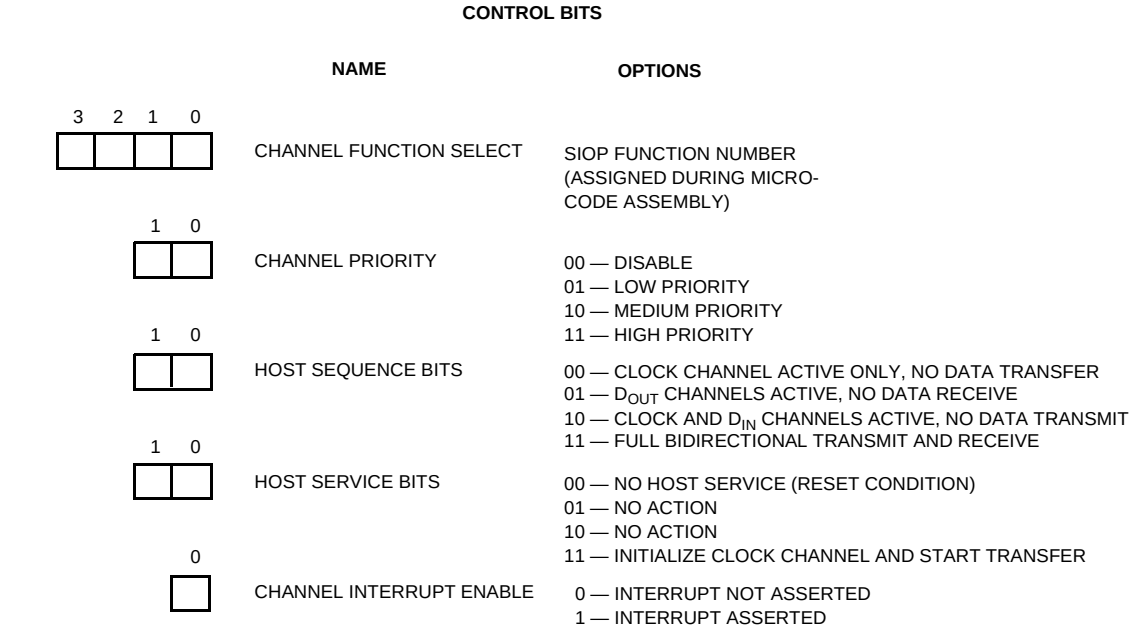


Figure D-27 Two Possible SIOP Configurations

D.17.1 Parameters

Figure D-28 shows the host interface areas and parameter RAM for the SIOP function. The following sections describe these parameters. Note that only the clock channel requires any programming by the user — the data in and out channels are entirely under TPU microcode control.



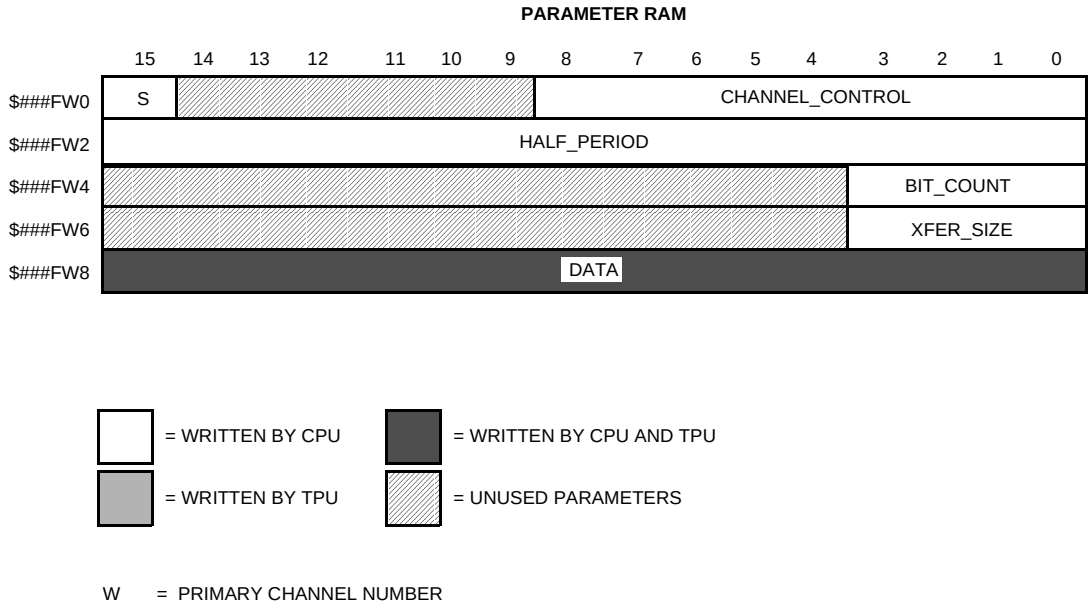


Figure D-28 SIOP Parameters

D.17.1.1 CHAN_CONTROL

This 9-bit CPU written parameter is used to setup the clock polarity for the SIOP data transfer. The valid values for CHAN_CONTROL for this function are given in the table below. CHAN_CONTROL must be written by the host prior to issuing the host service request (HSR) to initialize the function.

Table D-4 SIOP Function Valid CHAN_Control Options

CHAN_CONTROL[1] 8 7 6 5 4 3 2 1 0	Resulting Action
0 1 0 0 0 1 1 0 1	Data valid on clock Falling edge.
0 1 0 0 0 1 1 1 0	Data valid on clock Rising edge.

NOTES:
1. Other values of CHAN_CONTROL may result in indeterminate operation.

D.17.1.2 BIT_D

BIT_D is a CPU written bit that determines the direction of shift of the SIOP data. If BIT_D is zero then SIOP_DATA is right shifted (lsb first). If BIT_D is one then SIOP_DATA is left shifted (msb first).

D.17.1.3 HALF_PERIOD

This CPU-written parameter defines the baud rate of the SIOP function. The value contained in HALF_PERIOD is the number of TCR1 counts for a half SIOP clock period (e.g., for a 50 KHz baud rate, with a TCR1 period of 240 ns, the value [(1/50 KHz)/2]/240 ns = 42 should be written to HALF_PERIOD. The range for HALF_PERIOD is 1 to $8000, although the minimum value in practice will be limited by other system conditions. See notes on use and performance of SIOP function.

D.17.1.4 BIT_COUNT

This parameter is used by the TPU to count down the number bits remaining while a transfer is in progress. During the SIOP initialization state, BIT_COUNT is loaded with the value contained in XFER_SIZE. It is then decremented as the data is transferred and when it reaches zero, the transfer is complete and the TPU issues an interrupt request to the CPU.

D.17.1.5 XFER_SIZE

This CPU-written parameter determines the number of bits that make up a data transfer. During initialization, XFER_SIZE is copied into BIT_COUNT. XFER_SIZE is shown as a 5-bit parameter to match the maximum size of 16 bits in SIOP_DATA, although the TPU uses the whole word location. For normal use, XFER_SIZE should be in the range 1-to-16.

D.17.1.6 SIOP_DATA

This parameter is the data register for all SIOP transfers. Data is shifted out of one end of SIOP_DATA and shifted in at the other end, the shift direction being determined by the value of BIT_D. In output only mode, zero will be shifted into SIOP_DATA and

in input only mode, the data shifted out is ignored. In clock-only mode SIOP_DATA is still shifted. Note that no 'justifying' of SIOP_DATA is performed by the TPU, (e.g., if an 8-bit bi-directional transfer is made, shifting lsb first, then the bottom byte of SIOP_DATA will be shifted out and the input data will be shifted into the upper byte of SIOP_DATA).

Note

SIOP_DATA is not buffered. The CPU should only access it between completion of one transfer and the start of the next.

D.17.2 Host CPU Initialization of the SIOP Function

The CPU initializes the SIOP function by:

1. Disabling the channel by clearing the two channel priority bits
2. Selecting the SIOP function on the channel by writing the assigned SIOP function number to the function select bits
3. Writing CHAN_CONTROL in the clock channel parameter RAM
4. Writing HALF_PERIOD, BIT_D and XFER_SIZE in the clock channel parameter RAM to determine the speed, shift direction and size of the transfer
5. Writing SIOP_DATA if the data output is to be used
6. Selecting the required operating mode via the two host sequence bits
7. Issuing a host service request type %11
8. Enabling service by assigning H, M or L priority to the clock channel via the two channel priority bits

The TPU then starts the data transfer, and issues an interrupt request when the transfer is complete.

Once the function has been initialized, the CPU only needs to write SIOP_DATA with the new data and issue a HSR %11 to initiate a new transfer. In input or clock-only modes, just the HSR %11 is required.

D.17.3 SIOP Function Performance

Like all TPU functions, the performance limit of the SIOP function in a given application is dependent to some extent on the service time (latency) associated with other active TPU channels. This is due to the operational nature of the scheduler. Where two channels are being used for a uni-directional system, and no other TPU channels are active, the maximum baud rate is approximately 230 KHz at a bus speed of 16.77 MHz. A three-channel bi-directional system under the same conditions has a maximum baud rate of approximately 200 KHz. When more TPU channels are active, these performance figures will be degraded, however, the scheduler assures that the worst case latency in any TPU application can be closely approximated. It is recommended that the guidelines given in the TPU reference manual be used along with the information given in the SIOP state timing table to perform an analysis on any proposed TPU application that appears to approach the performance limits of the TPU.

Table D-5 SIOP State Timing[1]

State Number and Name	Max. CPU Clock Cycles	Number of RAM Accesses by TPU
S1 SIOP_INIT HSQ = X0 X1	 28 38	 7 7
S2 DATA_OUT HSQ = X0 X1	 14 24	 4 4
S3 DATA_IN HSQ = 0X 1X	 14 28	 4 6

NOTES:
1. Execution times do not include the time slot transition time (TST = 10 or 14 CPU clocks).

D.17.3.1 XFER_SIZE Greater than 16

XFER_SIZE is normally programmed to be in the range 1-to-16 to match the size of SIOP_DATA, and has thus been shown as a 5-bit value in the host interface diagram. However, the TPU actually uses all 16 bits of the XFER_SIZE parameter when loading BIT_COUNT. In some unusual circumstances this can be used to the user's advantage. If an input device is producing a data stream of greater than 16 bits then manipulation of XFER_SIZE will allow selective capturing of the data. In clock-only mode, the extended XFER_SIZE can be used to generate up to $FFFF clocks.

D.17.3.2 Data Positioning

As stated above, no 'justifying' of the data position in SIOP_DATA is performed by the TPU. This means that in the case of a byte transfer, the data output will be sourced from one byte and the data input will shift into the other byte. This rule holds for all data size options except 16 bits when the full SIOP_DATA register is used for both data output and input.

D.17.3.3 Data Timing

In the example given in **Figure D-29**, the data output transitions are shown as being completely synchronous with the relevant clock edge and it is assumed that the data input is latched exactly on the opposite clock edge. This is the simplest way to show the examples, but is not strictly true. Since the TPU is a multi-tasking system, and the data channels are manipulated directly by microcode software while servicing the clock edge, there is a finite delay between the relevant clock edge and the data-out being valid or the data-in being latched. This delay is equivalent to the latency in servicing the clock channel due to other TPU activity and is shown as 'Td' in the timing diagram. Td is the delay between the clock edge and the next output data being valid and also the delay between the opposite clock edge and the input data being read. For the vast majority of applications, the delay Td will not present a problem and can be ignored. Only for a system which heavily loads the TPU should the user calculate the worst case latency for the SIOP clock channel + actual SIOP service time (= Td) and ensure that the baud rate is chosen such that HALF_PERIOD - Td is not less that the minimum setup time of the receiving device. A transmitting device must also hold data valid for a minimum time of Td after the clock.

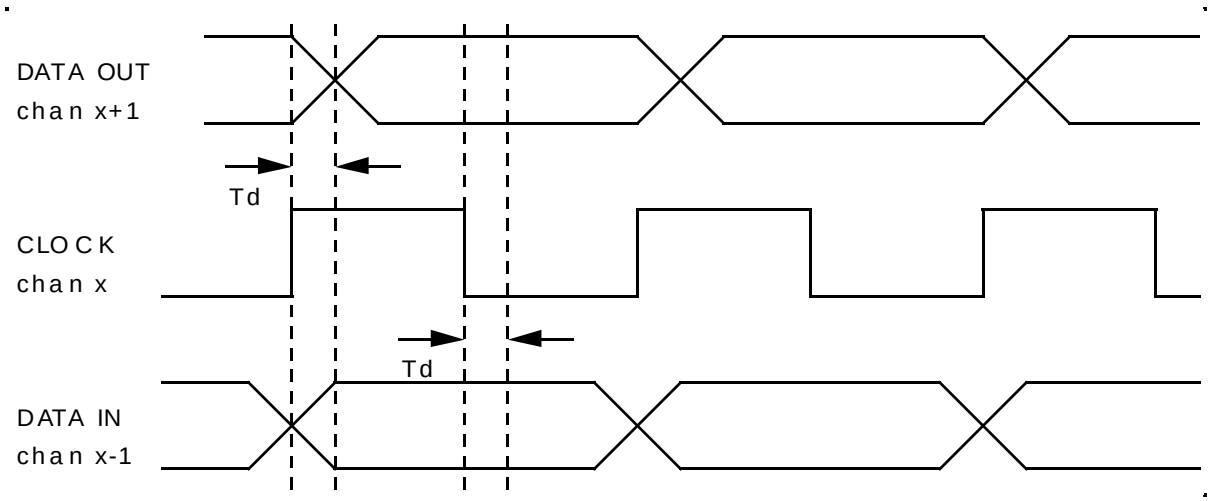


Figure D-29 SIOP Function Data Transition Example