# Synchronized Pulse-Width Modulation (SPWM)

By Kevin Anderson

## 1 Functional Overview

The synchronized pulse-width modulation function generates a pulse-width modulated (PWM) waveform in which the CPU can change the period or high time at any time. When synchronized to a time function on a second channel, SPWM low-to-high transitions have a time relationship to transitions on the second channel.

## 2 Detailed Description

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

The following definitions are used in the SPWM descriptions.

Synchronization means that a time relationship exists between waveforms occurring on different channels.

A link means that a signal (link service request) has been sent from a source channel (linking channel) to a destination channel (linked channel).

Links are one way to accomplish synchronization. Synchronization can also occur in a repetitive fashion independent of a link signal. The time function and its various modes of operation determine the method used to perform synchronization.

SPWM has three modes of operation: mode 0, mode 1, and mode 2.

Mode 0 is indicated by host sequence bits 00. A repetitive waveform is generated in which the rising transition is calculated by adding PERIOD to the last rising transition time (LASTRISE). Calculation of the fall time is made by adding HIGH_TIME to LASTRISE. A channel operating in mode 0 can synchronize to another channel by receiving a link from the other channel. Upon receiving a link, a mode 0 channel calculates the next rising transition by adding a reference time, pointed to by REF_ADDR1, to the values of DELAY and PERIOD. DELAY and PERIOD are not changed.

Mode 1 is indicated by host sequence bits 01. The rising transition is calculated by adding a reference time, pointed to by REF_ADDR1, to DELAY. The reference time pointed to by REF_ADDR1 is saved in REF_VALUE. The falling transition is calculated by adding a second reference time, pointed to by REF_ADDR2, to DELAY and HIGH_TIME. A mode 1 PWM repetitively synchronizes each rising edge to a referenced time on another channel.

Mode 2 is indicated by host sequence bits 10. On each rising transition, a channel using mode 2 generates a link to a sequential block of up to eight channels to allow for synchronization. All other operation is identical to mode 0.

A channel operating in mode 0 or mode 1 can be linked, but has no capability to generate links. A channel operating in mode 2 can both generate and receive links. Therefore, a channel operating in mode 2

can link to channels operating in mode 0, mode 1, or mode 2. Only channels operating in mode 0 and mode 2 are programmed to resynchronize when linked. A channel operating in mode 1 synchronizes in a repetitive fashion independent of a link signal and performs an immediate update (described in more detail below) when linked.

In all modes, if the channel interrupt enable bit is set, an interrupt request is asserted for each low-to-high transition, the time of the transition is moved to LASTRISE, and calculation of the fall time and rise time is made, with the calculated rise time stored in NEXTRISE.

When using the three SPWM modes to generate synchronized PWM waveforms, mode 0 and mode 2 channels should have REF_ADDR1 pointing to LASTRISE or some other reference value on the synchronizing channel. Channels operating in mode 1 should have REF_ADDR1 pointing to NEXTRISE and REF_ADDR2 pointing to LASTRISE, both on the synchronizing channel. Figure 1 shows how the SPWM modes operate together.

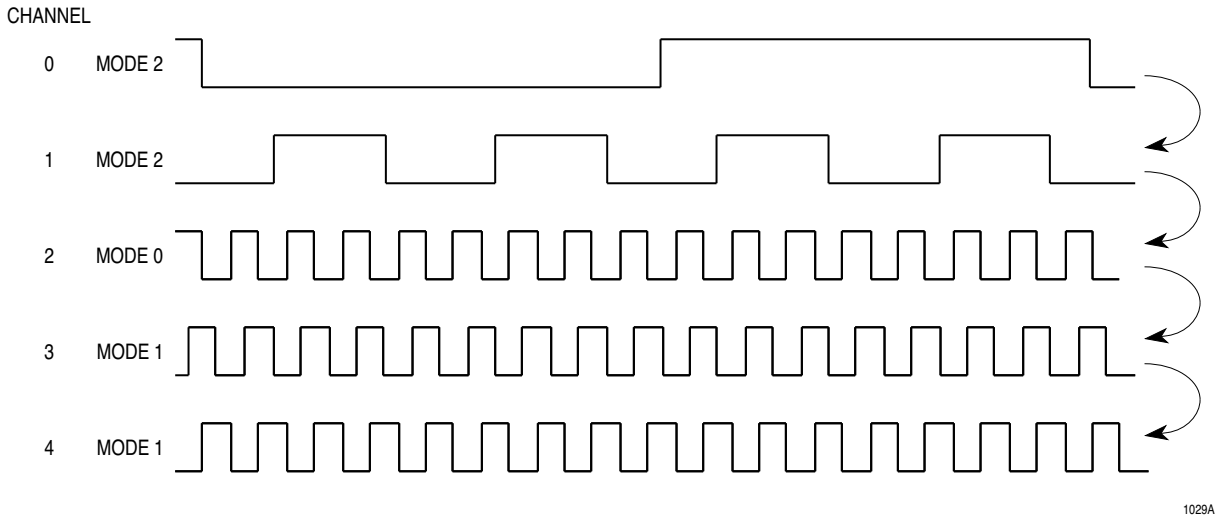



**Figure 1 SPWM Waveforms**

Channel 0 (using mode 2) links to channel 1 on each rising transition. Channel 1 (using mode 2) synchronizes to channel 0 when linked, and links to channel 2 on each rising transition. Channel 2 (using mode 0) synchronizes to channel 1 when linked. Channels 3 and 4 (using mode 1) synchronize to channels 2 and 3, respectively, on each of their rising edges. Each channel, with the exception of channel 0, synchronizes as specified by parameter DELAY; the rising edge of the synchronizing channel is delayed from the specified transition time by the value in DELAY. The DELAY parameter for channels 1 through 4 in the example is set for a 90° phase delay. Synchronized channels must be assigned to the same priority level in ascending numerical order so that the channel with the lower channel number performs its function before the channel that synchronizes to it.

In general, if the period of synchronized PWMs is not equal, modes 0 or 2 should be used. If the period is equal, mode 1 should be used. An immediate update is provided in mode 1, where the new delay is used during the immediate host service state, or when linked by another channel. In immediate mode, NEXTRISE is calculated using REF_VALUE added to DELAY. If immediate mode is not selected, the new delay is calculated as previously described for mode 1.

A detailed description of the SPWM algorithm, including a state diagram, is provided for reference at the end of this document.

## 3 Function Code Size

Total TPU function code size determines what combination of functions can fit into a given ROM or emulation memory microcode space. SPWM function code size is:

46 μ instructions + 8 entries = **54 long words**

## 4 Function Parameters

This section provides detailed descriptions of function parameters stored in channel parameter RAM. **Figure 2** shows TPU parameter RAM address mapping. **Figure 3** shows the parameter RAM assignment used by the function. In the diagrams, Y = M111, where M is the value of the module mapping bit (MM) in the system integration module configuration register (Y = $7 or $F).

| Channel Number | Base Address | Parameter Address | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | $YFFF## | 00 | 02 | 04 | 06 | 08 | 0A | — | — |
| 1 | $YFFF## | 10 | 12 | 14 | 16 | 18 | 1A | — | — |
| 2 | $YFFF## | 20 | 22 | 24 | 26 | 28 | 2A | — | — |
| 3 | $YFFF## | 30 | 32 | 34 | 36 | 38 | 3A | — | — |
| 4 | $YFFF## | 40 | 42 | 44 | 46 | 48 | 4A | — | — |
| 5 | $YFFF## | 50 | 52 | 54 | 56 | 58 | 5A | — | — |
| 6 | $YFFF## | 60 | 62 | 64 | 66 | 68 | 6A | — | — |
| 7 | $YFFF## | 70 | 72 | 74 | 76 | 78 | 7A | — | — |
| 8 | $YFFF## | 80 | 82 | 84 | 86 | 88 | 8A | — | — |
| 9 | $YFFF## | 90 | 92 | 94 | 96 | 98 | 9A | — | — |
| 10 | $YFFF## | A0 | A2 | A4 | A6 | A8 | AA | — | — |
| 11 | $YFFF## | B0 | B2 | B4 | B6 | B8 | BA | — | — |
| 12 | $YFFF## | C0 | C2 | C4 | C6 | C8 | CA | — | — |
| 13 | $YFFF## | D0 | D2 | D4 | D6 | D8 | DA | — | — |
| 14 | $YFFF## | E0 | E2 | E4 | E6 | E8 | EA | EC | EE |
| 15 | $YFFF## | F0 | F2 | F4 | F6 | F8 | FA | FC | FE |

— = Not Implemented (reads as $00)

**Figure 2 TPU Channel Parameter RAM CPU Address Map**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFFW0 | LASTRISE | | | | | | | | CHANNEL_CONTROL* | | | | | | | |
| $YFFFW2 | NEXTRISE | | | | | | | | | | | | | | | |
| $YFFFW4 | HIGH_TIME | | | | | | | | | | | | | | | |
| $YFFFW6 | PERIOD | | | | | | | | | | | | | | | |
| $YFFFW8 | | | | | | | | | REF_ADDR1 | | | | | | | |
| $YFFFWA | DELAY | | | | | | | | | | | | | | | |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFFW0 | LASTRISE | | | | | | | | CHANNEL_CONTROL* | | | | | | | |
| $YFFFW2 | NEXTRISE | | | | | | | | | | | | | | | |
| $YFFFW4 | HIGH_TIME | | | | | | | | | | | | | | | |
| $YFFFW6 | PERIOD | | | | | | | | | | | | | | | |
| $YFFFW8 | REF_ADDR1 | | | | | | | | REF_ADDR2 | | | | | | | |
| $YFFFWA | REF_VALUE | | | | | | | | | | | | | | | |

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFFW0 | LASTRISE | | | | | | | | CHANNEL_CONTROL* | | | | | | | |
| $YFFFW2 | NEXTRISE | | | | | | | | | | | | | | | |
| $YFFFW4 | HIGH_TIME | | | | | | | | | | | | | | | |
| $YFFFW6 | PERIOD | | | | | | | | | | | | | | | |
| $YFFFW8 | START_LINK_CHANNEL | | | | | LINK_CHANNEL_COUNT | | | REF_ADDR1 | | | | | | | |
| $YFFFWA | DELAY | | | | | | | | | | | | | | | |

Y = Channel number

*Once the channel is initialized, the channel control bits become the low nine bits of the LASTRISE parameter written by the TPU.

Parameter Write Access:

- Written by CPU
- Written by TPU
- Written by CPU and TPU
- Unused parameters

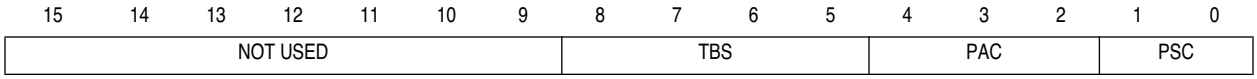
**Figure 3 Parameter RAM Assignment**

### 4.1 CHANNEL_CONTROL

CHANNEL_CONTROL shares the same location with LASTRISE and contains the PSC, PAC, and TBS fields. The PSC field forces the output level of the pin directly without affecting the PAC latches, or forces the output level to the state specified by the PAC latches. Normally, PSC is set to force the pin low to allow for synchronization of all channels before any pulses are generated. The SPWM function does not use the PAC field; it uses direct control by the microcode. The TBS field configures a channel pin as input or output and configures the time base for output match/input capture events.

**NOTE**

This channel must be configured as an output because the SPWM function is indeterminate when programmed as an input.




CHANNEL_CONTROL must be written by the CPU before initialization. The table below defines the allowable data for this parameter.

**Table 1 SPWM CHANNEL_CONTROL Options**

| TBS | PAC | PSC | Action | |
|---|---|---|---|---|
| 8 7 6 5 | 4 3 2 | 1 0 | Input | Output |
| | | 0 0<br>0 1<br>1 0<br>1 1 | —<br>—<br>—<br>— | Force Pin as Specified by PAC<br>Latches<br>Force Pin High<br>Force Pin Low<br>Do Not Force Any State |
| | 1 x x | | Do Not Change PAC | Do Not Change PAC |
| 0 1 x x<br>0 1 0 0<br>0 1 1 1<br>1 1 x x | | | —<br>—<br>—<br>Do Not Change PAC | Output Channel<br>Capture TCR1, Compare TCR1<br>Capture TCR2, Compare TCR2<br>Do Not Change TBS |

### 4.2 LASTRISE

LASTRISE is the time of the previous low-to-high transition. When executing state *Init,* LASTRISE is set by the TPU to the value of the TCR specified in CHANNEL_CONTROL. When SPWM is executing, the TPU updates LASTRISE at the beginning of each pulse to contain the time of the last low-to-high transition.

### 4.3 NEXTRISE

NEXTRISE is the current calculated rise time that is always calculated at the beginning of the pulse (on the low-to-high transition). The TPU updates this parameter, as shown in the following relationships.

**Table 2 NEXTRISE Calculation for Modes 0 and 2**

| Condition | NEXTRISE Calculation |
|---|---|
| Not linked [on low to high transition] | NEXTRISE = LASTRISE + PERIOD |
| Linked [after pin goes low] | NEXTRISE = (REF_ADDR1)[1] + DELAY + PERIOD |
| Always [on low to high transition] | Falling edge = LASTRISE + HIGH_TIME |

NOTES
1. Parentheses indicate the value pointed to by the specified address.

**Table 3 NEXTRISE Calculation for Mode 1**

| Condition | NEXTRISE Calculation |
|---|---|
| Normal update [on low to high transition] | NEXTRISE = (REF_ADDR1)[1] + DELAY |
| Immediate update [link or host request] | NEXTRISE = REF_VALUE + DELAY |
| Always [on low to high transition] | Falling edge = (REF_ADDR2)[1] + DELAY + HIGH_TIME |

NOTES
1. Parentheses indicate the value pointed to by the specified address.

### 4.4 HIGH_TIME

HIGH_TIME, which is updated by the CPU, is the current pulse high time that may be updated at any time. The estimated best-case minimum value for HIGH_TIME is greater than 32 system clocks, assuming a single channel operating. When more than one channel is operating, the minimum value for HIGH_TIME is TPU configuration dependent; the variables involved are described in State Timing — SPWM Function.

In modes 0 and 2, HIGH_TIME must be less than or equal to $8000. In mode 1, (DELAY + HIGH_TIME) must be less than or equal to $8000. The user should calculate the case timing to ensure accurate execution of this function.

Since 0% and 100% PWMs are not supported in SPWM, HIGH_TIME should be updated to a value that is in the range 0 < HIGH_TIME < PERIOD. Also, SPWMs programmed for a duty cycle closer to 0% or 100% than TPU latency time allows will not occur as programmed.

### 4.5 PERIOD

PERIOD, which is updated by the CPU, is the current PWM period, and is used by the TPU to calculate the next low-to-high transition time. The estimated best-case minimum value for PERIOD is greater than 50 system clocks, assuming a single channel operating. When more than one channel is operating, the minimum value for PERIOD is TPU configuration dependent; the variables involved are described in State Timing — SPWM Function.

In modes 0 and 2, (PERIOD + DELAY) must be less than or equal to $8000. The user should calculate the case timing to ensure accurate execution of this function.

### 4.6 DELAY

DELAY contains the count value from a reference time that indicates when the next rising transition will occur. If the reference time represents the rising transition time of a synchronizing PWM, the next rising transition on the pin is delayed as specified by DELAY. This parameter is written by the CPU at initialization and at any time during function operation.

DELAY can generally be set to the ranges shown in the following relationships:

| | |
|---|---|
| Modes 0 and 2 | HIGH_TIME < DELAY < (PERIOD – HIGH_TIME) |
| Mode 1 | 0 ≤ DELAY < Effective Period |

These values are applicable for any duty cycle.

### 4.7 REF_ADDR1

In modes 0 and 2, when a link is received, REF_ADDR1 points to a reference value to which DELAY and PERIOD are added, forming the next rising transition time. In mode 1, the reference value is added to DELAY, thus forming the rising transition time. This parameter is written by the CPU at initialization with the lowest eight bits of the referenced channel parameter address.

### 4.8 REF_ADDR2

In mode 1, REF_ADDR2 points to a reference value to which DELAY and HIGH_TIME are added, thus forming the falling transition time. This parameter is written by the CPU at initialization with the lowest eight bits of the referenced channel parameter address.

### 4.9 REF_VALUE

REF_VALUE is loaded during the *Trans_L_H_Mode1* state with the reference value pointed to by REF_ADDR1. This value is added to DELAY to calculate the new rise time for an immediate update request. If the new rise time has passed, the pin will be forced high immediately.

### 4.10 START_LINK_CHANNEL

START_LINK_CHANNEL contains the first channel of the link block. This parameter is written by the CPU at initialization and at any time during the function operation.

### 4.11 LINK_CHANNEL_COUNT

LINK_CHANNEL_COUNT determines the number of channels in the link block. This parameter is written by the CPU at initialization and at any time during the function operation. If this parameter is used, it must be greater than zero and less than or equal to eight: 0 < count ≤ 8. No check is performed by the TPU. If this number is out of range, the results are unpredictable.

## 5 Host Interface to Function

This section provides information concerning the TPU host interface to the function. **Figure 4** is a TPU address map. Detailed TPU register diagrams follow the figure. In the diagrams, Y = M111, where M is the value of the module mapping bit (MM) in the system integration module configuration register (Y = $7 or $F).

| Address | 15 8 | 7 0 |
|---|---|---|
| $YFFE00 | TPU MODULE CONFIGURATION REGISTER (TPUMCR) | |
| $YFFE02 | TEST CONFIGURATION REGISTER (TCR) | |
| $YFFE04 | DEVELOPMENT SUPPORT CONTROL REGISTER (DSCR) | |
| $YFFE06 | DEVELOPMENT SUPPORT STATUS REGISTER (DSSR) | |
| $YFFE08 | TPU INTERRUPT CONFIGURATION REGISTER (TICR) | |
| $YFFE0A | CHANNEL INTERRUPT ENABLE REGISTER (CIER) | |
| $YFFE0C | CHANNEL FUNCTION SELECTION REGISTER 0 (CFSR0) | |
| $YFFE0E | CHANNEL FUNCTION SELECTION REGISTER 1 (CFSR1) | |
| $YFFE10 | CHANNEL FUNCTION SELECTION REGISTER 2 (CFSR2) | |
| $YFFE12 | CHANNEL FUNCTION SELECTION REGISTER 3 (CFSR3) | |
| $YFFE14 | HOST SEQUENCE REGISTER 0 (HSQR0) | |
| $YFFE16 | HOST SEQUENCE REGISTER 1 (HSQR1) | |
| $YFFE18 | HOST SERVICE REQUEST REGISTER 0 (HSRR0) | |
| $YFFE1A | HOST SERVICE REQUEST REGISTER 1 (HSRR1) | |
| $YFFE1C | CHANNEL PRIORITY REGISTER 0 (CPR0) | |
| $YFFE1E | CHANNEL PRIORITY REGISTER 1 (CPR1) | |
| $YFFE20 | CHANNEL INTERRUPT STATUS REGISTER (CISR) | |
| $YFFE22 | LINK REGISTER (LR) | |
| $YFFE24 | SERVICE GRANT LATCH REGISTER (SGLR) | |
| $YFFE26 | DECODED CHANNEL NUMBER REGISTER (DCNR) | |

**Figure 4 TPU Address Map**

**CIER** — Channel Interrupt Enable Register **$YFFE0A**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH 15 | CH 14 | CH 13 | CH 12 | CH 11 | CH 10 | CH 9 | CH 8 | CH 7 | CH 6 | CH 5 | CH 4 | CH 3 | CH 2 | CH 1 | CH 0 |

| CH | Interrupt Enable |
|---|---|
| 0 | Channel interrupts disabled |
| 1 | Channel interrupts enabled |

**CFSR[0:3]** — Channel Function Select Registers **$YFFE0C – $YFFE12**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CFS (CH 15, 11, 7, 3) | | | | CFS (CH 14, 10, 6, 2) | | | | CFS (CH 13, 9, 5, 1) | | | | CFS (CH 12, 8, 4, 0) | | | |

CFS[4:0] — SPWM Function Number (Assigned during microcode assembly)

**HSQR[0:1]** — Host Sequence Registers **$YFFE14 – $YFFE16**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH 15, 7 | | CH 14, 6 | | CH 13, 5 | | CH 12, 4 | | CH 11, 3 | | CH 10, 2 | | CH 9, 1 | | CH 8, 0 | |

| CH | Operating Mode |
|---|---|
| 00 | Mode 0 |
| 01 | Mode 1 |
| 10 | Mode 2 |
| 11 | — |

**HSRR[1:0]** — Host Service Request Registers **$YFFE18 – $YFFE1A**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH 15, 7 | | CH 14, 6 | | CH 13, 5 | | CH 12, 4 | | CH 11, 3 | | CH 10, 2 | | CH 9, 1 | | CH 8, 0 | |

| CH | Initialization |
|---|---|
| 00 | No Host Service (Reset Condition) |
| 01 | Undefined |
| 10 | Initialize |
| 11 | Immediate Update (Mode 1) |

**CPR[1:0]** — Channel Priority Registers **$YFFE1C – $YFFE1E**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH 15, 7 | | CH 14, 6 | | CH 13, 5 | | CH 12, 4 | | CH 11, 3 | | CH 10, 2 | | CH 9, 1 | | CH 8, 0 | |

| CH | Channel Priority |
|---|---|
| 00 | Disabled |
| 01 | Low |
| 10 | Middle |
| 11 | High |

**CISR** — Channel Interrupt Status Register **$YFFE20**

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH 15 | CH 14 | CH 13 | CH 12 | CH 11 | CH 10 | CH 9 | CH 8 | CH 7 | CH 6 | CH 5 | CH 4 | CH 3 | CH 2 | CH 1 | CH 0 |

| CH | Interrupt Status |
|---|---|
| 0 | Channel interrupt not asserted |
| 1 | Channel interrupt asserted |

## 6 Function Configuration

The CPU configures the SPWM function as follows:

1. Writes parameters CHANNEL_CONTROL, HIGH_TIME, PERIOD, DELAY, REF_ADDR1, etc., to RAM.
2. Issues a host service request 10 for initialization.
3. Enables channel servicing by assigning a high, middle, or low priority.

The TPU then executes initialization and asserts an interrupt if the interrupt enable bit is set. In the beginning of each period, new pulse parameters are calculated and interrupts are attempted. The CPU should monitor the host service request register (or the channel interrupt) until the TPU clears the host service request to 00 before changing any parameters or issuing a new service request to this channel.

The TPU stores the time of the last low-to-high transition in LASTRISE, which can be read by the CPU. The TPU calculates the pulse timing (next fall time, next rise time) in the beginning of the pulse, after generating a low-to-high transition. At this time, an interrupt is asserted if the interrupt enable bit is set. In order to change the SPWMs, the CPU writes HIGH_TIME, PERIOD, or DELAY to the parameter RAM.

Generally, new parameters are used on the next low-to-high transition. To cause an immediate update of the waveform in mode 1, the host CPU issues host service request 11. This request can be issued after any previous service request has been serviced, as indicated by the host service request bits of the channel being 00. After issuing either host service request 10 or 11, the CPU should wait for the host service request bits to be cleared by the TPU before changing any parameters or before issuing another service request to the channel.

## 7 Performance and Use of Function

### 7.1 Performance

Like all TPU functions, SPWM function performance in an application is to some extent dependent upon the service time (latency) of other active TPU channels. This is due to the operational nature of the scheduler. When a single SPWM channel is in use and no other TPU channels are active, the minimum time between successive matches is 40 CPU clocks. When more TPU channels are active, performance decreases. However, worst-case latency in any TPU application can be closely estimated. To analyze the performance of an application that appears to approach the limits of the TPU, use the guidelines given in the TPU reference manual and the information in State Timing — SPWM Function.

### 7.2 Changing Mode

The host sequence bits are used to select SPWM function operating mode. Change host sequence bit values only when the function is stopped or disabled (channel priority bits = %00). Disabling the channel before changing mode avoids conditions that cause indeterminate operation.

**Table 4 State Timing — SPWM Function**

| State Number and Name | Max CPU Clock Cycles | RAM Accesses by TPU |
|---|---|---|
| S1 *INIT*<br>Mode 0<br>Mode 1<br>Mode 2 | <br>24<br>32<br>30[1] | <br>4<br>5<br>4 |
| S2 *TRANS_L_H_MODE1* | 18 | 4 |
| S3 *TRANS_H_L_MODE1* | 2 | 1 |
| S4 *IMMED_LINK_MODE1* | 10 | 2 |
| S5 *TRANS_L_H_MODE0/2*<br>Mode 0<br>Mode 2 | <br>12<br>18[1] | <br>3<br>4 |
| S6 *TRANS_H_L_MODE0/2* | 14 | 4 |
| S7 *TRANS_L_H_LINK_MODE0/2*<br>Mode 0<br>Mode 2 | <br>14<br>20[1] | <br>3<br>4 |
| S8 *TRANS_H_L_LINK_MODE0/2* | 10 | 3 |
| S9 *LINK_MODE0/2* | 2 | 1 |

NOTES
1. Add two clocks for each channel linked.

### 7.3 Mode Usage

The three modes available in SPWM are designed to handle different synchronization conditions. Knowing the different cases that each mode handles is key to proper application of the function.

#### 7.3.1 Mode 0

This mode can generate a PWM signal on its own by proper initialization of the CHANNEL_CONTROL, PERIOD and HIGH_TIME parameter locations. However, using mode 0 in this manner is somewhat limited since SPWM cannot handle 0% and 100% duty cycles. The real power of mode 0 is that its generated output may be synchronized to an event on another channel. This event may come from a channel running SPWM mode 2 or any function which updates a reference time such as ITC.

When using mode 0 synchronized to another channel, the period must be less than or equal to the synchronizing channel. This is because the synchronizing channel causes a new low to high transition time to be scheduled on the linked channel for each of its own low to high transitions. The mode is normally used with the synchronizing channel period equal to an integer multiple of the mode 0 channel. If this is not the case there can be a one cycle change in period or high time when the channel desynchronizes.

When synchronized to a mode 2 channel, a mode 0 channel should be initialized according to these general guidelines.

Set REF1 to point to LASTRISE on the synchronizing channel.

Set the PSC field in CHANNEL_CONTROL to force the pin low. This allows the first rising edge to be scheduled by the synchronizing channel, which occurs at LASTRISE (of the synchronizing channel) + DELAY + PERIOD. Since PERIOD is part of this calculation, the first rising edge will not occur at the DELAY time offset from the synchronizing channel as one might expect, but rather at that time plus one period. However, if DELAY is set to a legal value and the linked channel period is an integer multiple of the synchronizing channel, all subsequent edges will appear as expected.

The legal value of the DELAY parameter is governed by the channel's period and high time. In general, the allowable values are:

HIGH_TIME < DELAY < (PERIOD – HIGH_TIME)

For example, if a mode 2 channel with a period of $800 synchronizes a mode 0 channel with a period of $80 and a hightime of $30, the allowable values for delay would be $FFD1 through $0050.

The minimum allowable value for PERIOD is greater than 50 system clocks, assuming a single channel operating. With more than one channel running, channel service latency must be taken into account to determine the minimum value. The maximum PERIOD value must satisfy the condition PERIOD + DELAY ≤ $8000.

The minimum allowable value for HIGH_TIME is greater than 32 system clocks, assuming a single channel operating. With more than one channel running, channel service latency once again determines the minimum value. Since 100% duty cycles are not supported, HIGH_TIME must be less than PERIOD minus the maximum channel service latency.

Synchronized channels must be assigned to the same priority in ascending numerical order with synchronizing channels having lower channel numbers than channels which they synchronize. Since lower channel numbers are serviced first within a priority level, this will insure that any synchronizing channel will be serviced (and referenced parameters updated) before any channel that it synchronizes.

### 7.3.2 Mode 1

Mode 1 is usually used for a synchronized channel whose period is the same as the synchronizing channel. The synchronizing channel may be running any SPWM mode. Note that there is no PERIOD parameter for this mode. Instead, the rise and fall edges are calculated from reference times on a synchronizing channel. These two reference times are pointed to by the parameters REF_ADDR1 and REF_ADDR2.

Mode 1 also provides for an immediate update by the CPU. This can be used by the CPU to schedule a new delay value for the synchronized channel on the current cycle. This update is scheduled by an HSR%11 on a mode 1 channel. If the synchronized rise time for this channel has not occurred during this period the new delay value will be used. If it has occurred, the new delay value will take effect in the next period. If the rise time has not occurred for this period and the new delay time called for has already passed, an immediate low to high transition will occur.

When synchronized to another channel, a mode 1 channel should be initialized according to these general guidelines.

REF_ADDR1 must point to NEXTRISE on the synchronizing channel. REF_ADDR1 is used to calculate the next rising edge. Setting REF_ADDR1 to NEXTRISE allows for very small accurate delay values between the rising edges of the two channels. In fact, by setting DELAY equal to zero the two rising edges will be coincident.

REF_ADDR2 must point to LASTRISE on the synchronizing channel. REF_ADDR2 is used to calculate the falling edge of the synchronized channel.

Set the PSC field in CHANNEL_CONTROL to force the pin low. This allows the first rising edge to be scheduled by the synchronizing channel which occurs at NEXTRISE (of the synchronizing channel) + DELAY. One artifact of using the NEXTRISE value is that the earliest that the first synchronized rising edge can occur after initialization is one period of the synchronizing channel. This may be remedied by initializing the synchronizing channel to start with the pin state low so that there are no rising edges during its first period time.

The DELAY parameter legal values are

$$0 \leq \text{DELAY} < \text{Effective Period}$$

The maximum value of DELAY must also satisfy the condition HIGH_TIME + DELAY ≤ $8000.

The minimum allowable value for HIGH_TIME is greater than 32 system clocks, assuming a single channel operating. With more than one channel running, channel service latency determines the minimum value. Since 100% duty cycles are not supported, HIGH_TIME must be less than the effective period minus the maximum channel service latency. The maximum value of HIGH_TIME must also satisfy the condition HIGH_TIME + DELAY ≤ $8000.

Synchronized channels must be assigned to the same priority in ascending numerical order with synchronizing channels having lower channel numbers than channels which they synchronize. Since lower channel numbers are serviced first within a priority level, this insures that any synchronizing channel is serviced (and referenced parameters updated) before any channel that it synchronizes.

### 7.3.3 Mode 2

Mode 2 is the most general purpose of the three modes. It has the operational nature of mode 0 with the addition that it generates synchronizing signals to other channels. It is the only mode in SPWM with this capability.

Mode 2 synchronizes channels by generating a link to the channel. Up to eight consecutive channel numbers may be linked by the channel. Two parameter fields are used for this purpose. The LINK_CHANNEL_COUNT specifies the number of channels between one and eight. The START_LINK_CHANNEL field holds the channel number of the first linked channel. For example, if the application called for TPU Channel 0 running mode 2 to link to TPU channels 5 through 8 running mode 0, LINK_CHANNEL_COUNT field on Channel 0 would be written to $04 and the START_LINK_CHANNEL field would be written to $05.

Notice that zero is not a legal value for LINK_CHANNEL_COUNT. Thus, a mode 2 channel must link at least one other channel. If no links are needed mode 0 should be used.

If a mode 2 channel will be linked to, the CHANNEL_CONTROL parameter should be initialized to force the pin low at initialization. In this way the first rising edge will be scheduled by the linking channel. If a mode 2 channel will only generate links, then the required start-up conditions dictate the initial state of the pin. In most cases CHANNEL_CONTROL is set to force the pin high if a mode 0 or 2 channel is being linked, and low if a mode 1 channel is being synchronized.

All other mode 2 parameters follow the same rules as described in mode 0.

## 8 SPWM Examples

The following examples give an indication of the capabilities of the SPWM function. Each example includes a description of the example, a diagram of the initial parameter RAM content, initial control bit settings, and a diagram of the output waveform.

### 8.1 Example A

### 8.2 Description

Generate a 50% duty cycle PWM signal with a period of $80 on channel 1 synchronized to a 25% duty cycle PWM signal on channel 0 with a period of $200. Delay the rising edge of channel 1 by 180 degrees from channel 0. Use TCR1 for the time base.

### 8.3 Initialization

Disable channels 0 and 1 by clearing channel priority bits CPR1[0:3]. Initialize the parameter RAM for each channel as shown below. Select the SPWM function by programming the function number ($7 for the standard mask) into CFSR3[0:7]. Select mode 2 for channel 0 and mode 0 for channel 1 by programming 10 into HSQR1[0:1] and 00 into HSQR1[2:3]. Request an INIT host service request for both channels by programming $A into HSRR1[0:3]. Finally, enable service by selecting low, middle, or high priority for the channels in CPR1[0:3].

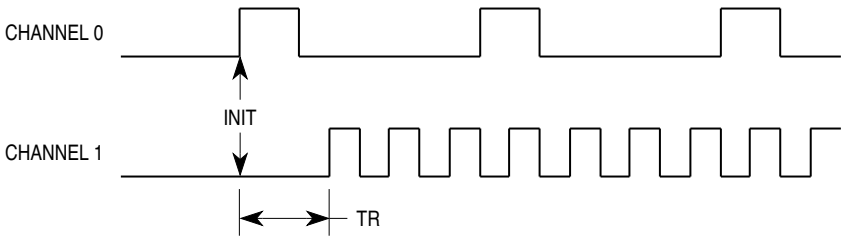
**Table 5 SPWM Channel 0 Parameter RAM**

| | 15 | | | | | | | 8 | | | | | | | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFF00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | CH_CNTL |
| $YFFF02 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | NEXT_RISE |
| $YFFF04 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | HIGH_TIME |
| $YFFF06 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PERIOD |
| $YFFF08 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | LINK/REF1 |
| $YFFF0A | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DELAY |

**Table 6 SPWM Channel 1 Parameter RAM**

| | 15 | | | | | | | 8 | | | | | | | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFF10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | CH_CNTL |
| $YFFF12 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | NEXT_RISE |
| $YFFF14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | HIGH_TIME |
| $YFFF16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PERIOD |
| $YFFF18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | REF_ADDR1 |
| $YFFF1A | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | DELAY |

#### 8.3.1 Output Waveforms




TR = LASTRISE + PERIOD + DELAY

TPU PN19 EXA TIM

## 8.4 Example B

### 8.4.1 Description

Generate a PWM signal with a period of $400 and a high time of $80 on channel 1 synchronized to a 50% duty cycle PWM signal on channel 0 with a period of $400. Delay the rising edge of channel 1 by $100 TCR1 clock ticks from channel 0.

### 8.4.2 Initialization

Disable channels 0 and 1 by clearing the channel priority bits (CPR1[0:3]). Initialize the parameter RAM for each channel as shown below. Select the SPWM function by programming the function number ($7 for the standard mask) into CFSR3[0:7]. Select mode 2 for channel 0 and mode 1 for channel 1 by programming 10 into HSQR1[0:1] and 01 into HSQR1[2:3]. Request an INIT host service request for both channels by programming $A into HSRR1[0:3]. Finally, enable service by selecting low, middle, or high priority for the channels in CPR1[0:3].

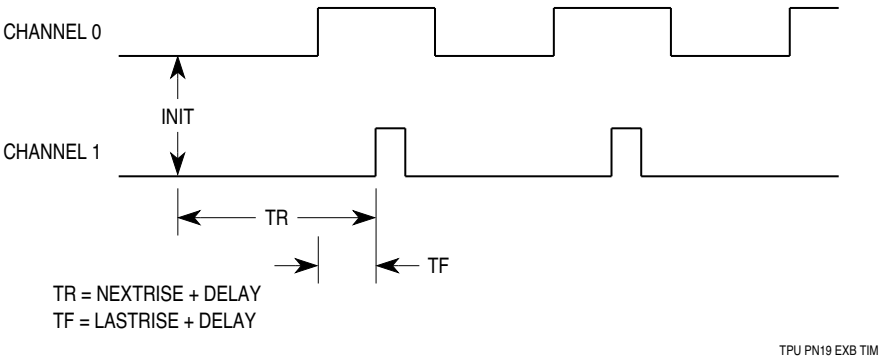
**Table 7 SPWM Channel 0 Parameter RAM**

| | 15 | | | | | | | 8 | | | | | | | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFF00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | CH_CNTL |
| $YFFF02 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | NEXT_RISE |
| $YFFF04 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | HIGH_TIME |
| $YFFF06 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PERIOD |
| $YFFF08 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | LINK/REF1 |
| $YFFF0A | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DELAY |

**Table 8 SPWM Channel 1 Parameter RAM**

| | 15 | | | | | | | 8 | | | | | | | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFF10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | CH_CNTL |
| $YFFF12 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | NEXT_RISE |
| $YFFF14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | HIGH_TIME |
| $YFFF16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DELAY |
| $YFFF18 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | REF1/REF2 |
| $YFFF1A | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | REF_VALUE |

### 8.4.3 Output Waveforms



### 8.5 Example C

### 8.6 Description

Program the TPU to generate the SPWM waveforms shown in **Figure 1**. Use a period of $2000 TCR1 counts for channel 0.

### 8.7 Initialization

Disable channels 0 through 4 by clearing the channel priority bits (CPR1[0:9]). Initialize the parameter RAM for each channel as shown below. Select the SPWM function by programming the function number ($7 for the standard mask) into CFSR3[0:15] and CFSR2[0:3]. Select mode 2 for channels 0 and 1 by programming 10 into HSQR1[0:1] and HSQR1[2:3], mode 0 for channel 2 by programming 00 into HSQR1[4:5], and mode 1 for channels 3 and 4 by programming 01 into HSQR1[6:7] and HSQR1[8:9]. Request an INIT host service request for all channels by programming $2AA into HSRR1[0:9]. Finally, enable service by selecting a priority level for the channels in CPR1[0:9].

**Table 9 SPWM Channel 0 Parameter RAM**

| | 15 | | | | | | | 8 | | | | | | | | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFF00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | CH_CNTL |
| $YFFF02 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | NEXT_RISE |
| $YFFF04 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | HIGH_TIME |
| $YFFF06 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PERIOD |
| $YFFF08 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | LINK/REF1 |
| $YFFF0A | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DELAY |

**Table 10 SPWM Channel 1 Parameter RAM**

| | 15 | | | | | | | 8 | | | | | | | | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFF10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | CH_CNTL |
| $YFFF12 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | NEXT_RISE |
| $YFFF14 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | HIGH_TIME |
| $YFFF16 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PERIOD |
| $YFFF18 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | LINK/REF1 |
| $YFFF1A | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DELAY |

**Table 11 SPWM Channel 2 Parameter RAM**

| | 15 | | | | | | | 8 | | | | | | | | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFF20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | CH_CNTL |
| $YFFF22 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | NEXT_RISE |
| $YFFF24 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | HIGH_TIME |
| $YFFF26 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PERIOD |
| $YFFF28 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | REF_ADDR1 |
| $YFFF2A | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DELAY |

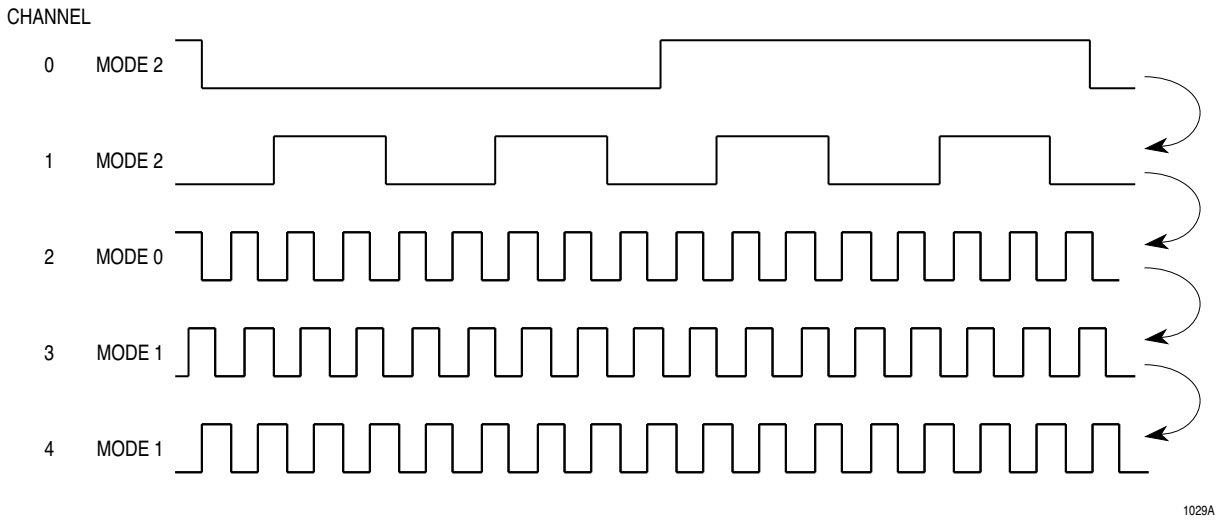
**Table 12 SPWM Channel 3 Parameter RAM**

| | 15 | | | | | | | | 8 | | | | | | | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFF30 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | CH_CNTL |
| $YFFF32 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | NEXT_RISE |
| $YFFF34 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | HIGH_TIME |
| $YFFF36 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DELAY |
| $YFFF38 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | REF1/REF2 |
| $YFFF3A | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | REF_VALUE |

**Table 13 SPWM Channel 4 Parameter RAM**

| | 15 | | | | | | | | 8 | | | | | | | 0 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $YFFF40 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | CH_CNTL |
| $YFFF42 | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | NEXT_RISE |
| $YFFF44 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | HIGH_TIME |
| $YFFF46 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DELAY |
| $YFFF48 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | REF1/REF2 |
| $YFFF4A | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | REF_VALUE |

### 8.7.1 Output Waveforms




## 9 Function Algorithm

The following description is provided as a guide only, to aid understanding of the function. The exact sequence of operations in microcode may be different from that shown, in order to optimize speed and code size. TPU microcode source listings for all functions in the TPU function library can be downloaded from the Freescale Freeware bulletin board. Refer to *Using the TPU Function Library and TPU Emulation Mode* (TPUPN00/D) for detailed instructions on downloading and compiling microcode.

The synchronized pulse-width modulation function consists of nine states, described below. For clarity, reference is made to internal flags in the following descriptions. These internal TPU control bits are not available to the user.

### 9.1 STATE1: Init

This state is entered as a result of HSR %10. It initializes the pulse parameters as well as the channel latches. The start time of the pulse is set to the current TCR time. The PSC field determines the setting of the pin after initialization. Flag0 is set to indicate mode 0/2 (flag0 = 0) or mode 1 (flag0 = 1).

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 10xxxx

Match Enable: Disable

```
Configure channel latches via CHANNEL_CONTROL
Negate MRL
Negate TDL
Clear flag0
Clear flag1
If (match TCR = TCR1) then {
    ERT = TCR1
}
Else {
    ERT = TCR2
}
If (sequence bit 0 = 0) then {
    Goto Trans_L_H_Mode 0/2 state
}
Else {
    Goto Trans_L_H_Mode 1 state
}
```

### 9.2 State 2: Trans_L_H_Mode1

This state is entered after a match occurs and a low-to-high transition results. In this state, the TPU sets the fall time of the pulse, calculates and stores the new rise time, stores the last rise time and reference value, and generates an interrupt.

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 001x11

Match Enable: Enable

```
Assert interrupt request
Set flag0
LASTRISE = ERT
REF_VALUE = (REF_ADDR1)
NEXTRISE = REF_VALUE + DELAY
ER = (REF_ADDR2) + DELAY + HIGH_TIME
Set PAC to high to low
Negate MRL
Negate LSR
```

### 9.3 State 3: Trans_H_L_Mode1

This state is entered after a match occurs and a high-to-low transition results. In this state, the TPU sets the rise time of the pulse using the value stored in state 2 or state 4.

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 001001

Match Enable: Enable

```
ER = NEXTRISE
Set PAC to low to high
Negate MRL
```

### 9.4 State 4: Immed_Link_Mode1

This state is entered as a result of HSR %11 or a link. In this state the TPU calculates the new rise time of the pulse, and sets the rise time if the pin is low. Otherwise, the new rise time is set when a high-to-low transition occurs.

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 11xxxx

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 0001x1

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 001101

Match Enable: Disable

```
If (flag0 = 1) then {
    NEXTRISE = REF_VALUE + DELAY
    Negate LSR
    If (PS = 0) then {
        ER = NEXTRISE
        Set PAC to low to high
        Negate MRL
    }
}
```

### 9.5 State 5: Trans_L_H_Mode0/2

This state is entered after a match occurs and a low-to-high transition results. In this state, the TPU sets the fall time of the pulse, calculates and stores the new rise time, stores the last rise time, and generates an interrupt. A block of channels is also linked in mode 2.

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 001010

Match Enable: Enable

```
Assert interrupt request
Clear flag0
LASTRISE = ERT
NEXTRISE = LASTRISE + PERIOD
ER = LASTRISE + HIGH_TIME
Set PAC to high to low
Negate MRL
If (sequence bit 1 = 1) then {
    Link to block of channels
}
```

### 9.6 State 6: Trans_H_L_Mode0/2

This state is entered after a match occurs and a high-to-low transition results. In this state, the TPU sets the rise time of the pulse using the value stored in state 5. A new rise time is calculated if flag1 is asserted.

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 001000

Match Enable: Enable

```
If (Flag1 = 0) then {
    ER = NEXTRISE
    Set PAC to low to high
    Negate MRL
```

```
        Clear flag1
    }
    Else{
        Clear flag1
        NEXTRISE = (REF_ADDR1) + DELAY + PERIOD
        ER = NEXTRISE
        Set PAC to low to high
        Negate MRL
    }
```

### 9.7 State 7: Trans_L_H_Link_Mode0/2

This state is entered after a match occurs, a low-to-high transition results, and a link occurs. This state is identical to *Trans_L_H_Mode0/2* except that flag1 is also set.

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 001110

Match Enable: Enable

Negate LSR
Assert flag1
(*Trans_L_H_Link_Mode0/2* is executed)

### 9.8 State 8: Trans_H_L_Link_Mode0/2

This state is entered after a match occurs, a high-to-low transition results, and a link occurs. In this state, a new rise time is calculated and set by the TPU.

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 001100

Match Enable: Enable

Negate LSR
Clear flag1
NEXTRISE = (REF_ADDR1) + DELAY + PERIOD
ER = NEXTRISE
Set PAC to low to high
Negate MRL

### 9.9 State 9: Link_Mode0/2

This state is entered after a link. In this state, the TPU sets flag1 to cause a new rise time calculation in a subsequent state.

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 0001x0

Match Enable: Enable

Negate LSR
Assert flag1

The following table shows the SPWM state transitions listing the service request sources and channel conditions from current state to next state. **Figure 4** and **Figure 5** illustrate the flow of SPWM states, including the initialization and immediate update states.

**Table 14 SPWM State Transition Table**

| Current State | HSR | M/TSR | LSR | Pin | Flag0 | Next State |
|---|---|---|---|---|---|---|
| All States | 10 | — | — | — | — | S1 *Init* |
| | 11 | — | — | — | — | *S4 Immed_Link_Mode1* |
| S1 *Init* | 00 | 1 | 0 | 0 | 1 | S3 *Trans_H_L_Mode1* |
| | 00 | — | 1 | 0 | 1 | *S4 Immed_Link_Mode1* |
| | 00 | 1 | 0 | 0 | 0 | *S6 Trans_H_L_Mode0/2* |
| | 00 | 1 | 1 | 0 | 0 | S8 *Trans_H_L_Link_Mode0/2* |
| | 00 | 0 | 1 | 0 | 0 | S9 *Link_Mode0/2* |
| S2 *Trans_L_H_Mode1* | 00 | 1 | 0 | 0 | 1 | S3 *Trans_H_L_Mode1* |
| | 00 | 1 | 1 | 0 | 1 | *S4 Immed_Link_Mode1* |
| | 00 | 0 | 1 | 1 | 1 | S4 *Immed_Link_Mode1* |
| S3 *Trans_H_L_Mode1* | 00 | 1 | — | 1 | 1 | S2 *Trans_L_H_Mode1* |
| | 00 | 0 | 1 | 0 | 1 | *S4 Immed_Link_Mode1* |
| S4 *Immed_Link_Mode1* | 00 | 1 | — | 1 | 1 | S2 *Trans_L_H_Mode1* |
| | 00 | 1 | 0 | 0 | 1 | *S3 Trans_H_L_Mode1* |
| S5*Trans_L_H_Mode0/2* | 00 | 1 | 0 | 0 | 0 | S6*Trans_H_L_Mode0/2* |
| | 00 | 1 | 1 | 0 | 0 | *S8 Trans_H_L_Link_Mode0/2* |
| | 00 | 0 | 1 | 1 | 0 | S9 *Link_Mode0/2* |
| S6*Trans_H_L_Mode0/2* | 00 | 1 | 0 | 1 | 0 | S5*Trans_L_H_Mode0/2* |
| | 00 | 1 | 1 | 1 | 0 | *S7 Trans_L_H_Link_Mode0/2* |
| | 00 | 0 | 1 | 0 | 0 | S9 *Link_Mode0/2* |
| S7 *Trans_L_H_Link_Mode0/2* | 00 | 1 | 0 | 0 | 0 | S6*Trans_H_L_Mode0/2* |
| | 00 | 1 | 1 | 0 | 0 | *S8 Trans_H_L_Link_Mode0/2* |
| | 00 | 0 | 1 | — | 0 | S9 *Link_Mode0/2* |
| S8*Trans_H_L_Link_Mode0/2* | 00 | 1 | 0 | 1 | 0 | S5*Trans_L_H_Mode0/2* |
| | 00 | 1 | 1 | 1 | 0 | *S7 Trans_L_H_Link_Mode0/2* |
| | 00 | 0 | 1 | 0 | 0 | S9 *Link_Mode0/2* |
| S9 *Link_Mode0/2* | 00 | 1 | 0 | 1 | 0 | S5*Trans_L_H_Mode0/2* |
| | 00 | 1 | 0 | 0 | 0 | *S6 Trans_H_L_Mode0/2* |
| | 00 | 1 | 1 | 1 | 0 | S7 *Trans_L_H_Link_Mode0/2* |
| | 00 | 1 | 1 | 0 | 0 | S8 *Trans_H_L_Link_Mode0/2* |
| | 00 | 0 | 1 | — | 0 | S9 *Link_Mode0/2* |

NOTES:
1. Conditions not specified are “don't care”.
2. HSR = Host service request
   LSR = Link service request
   M/TSR = Either a match or transition (input capture) service request occurred (M/TSR = 1) or neither occurred (M/TSR = 0).

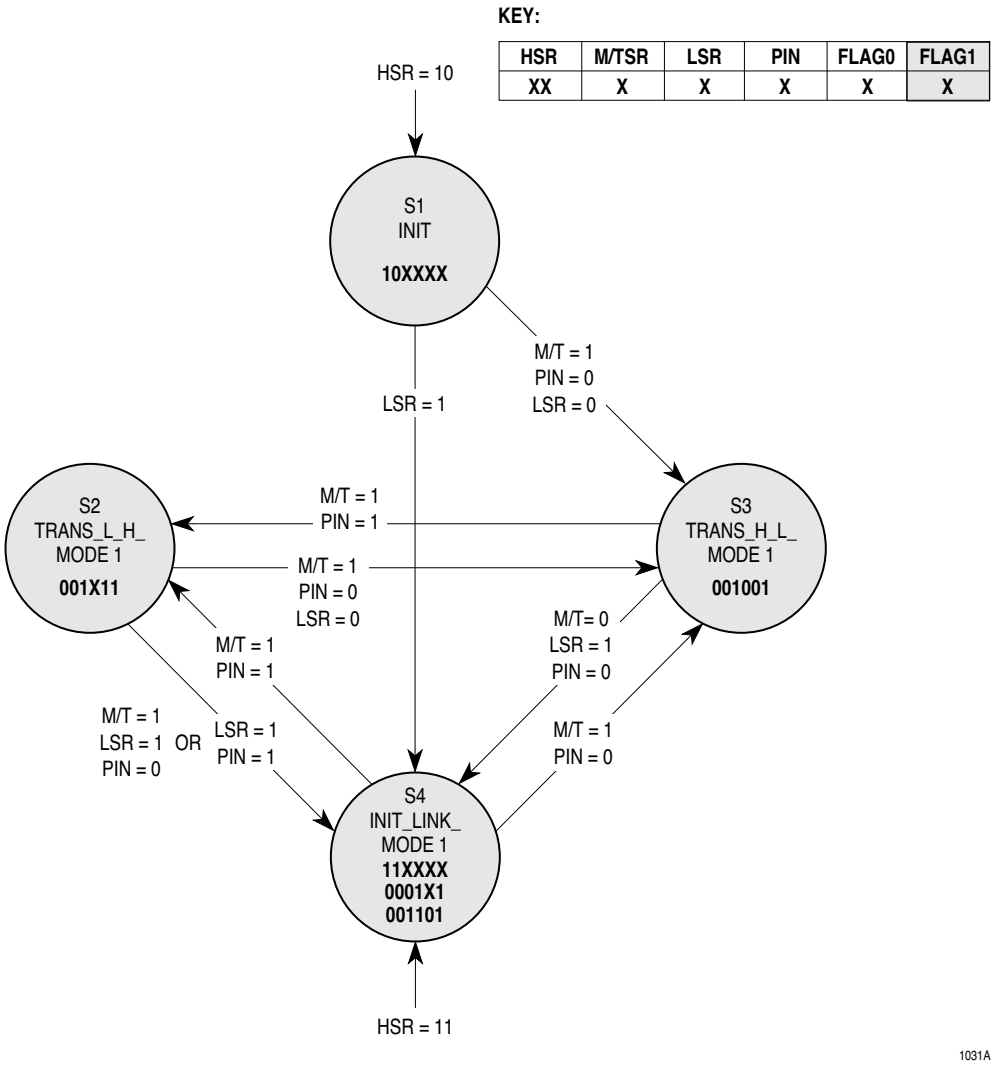



**Figure 5 SPWM State Flowchart, Mode 1**

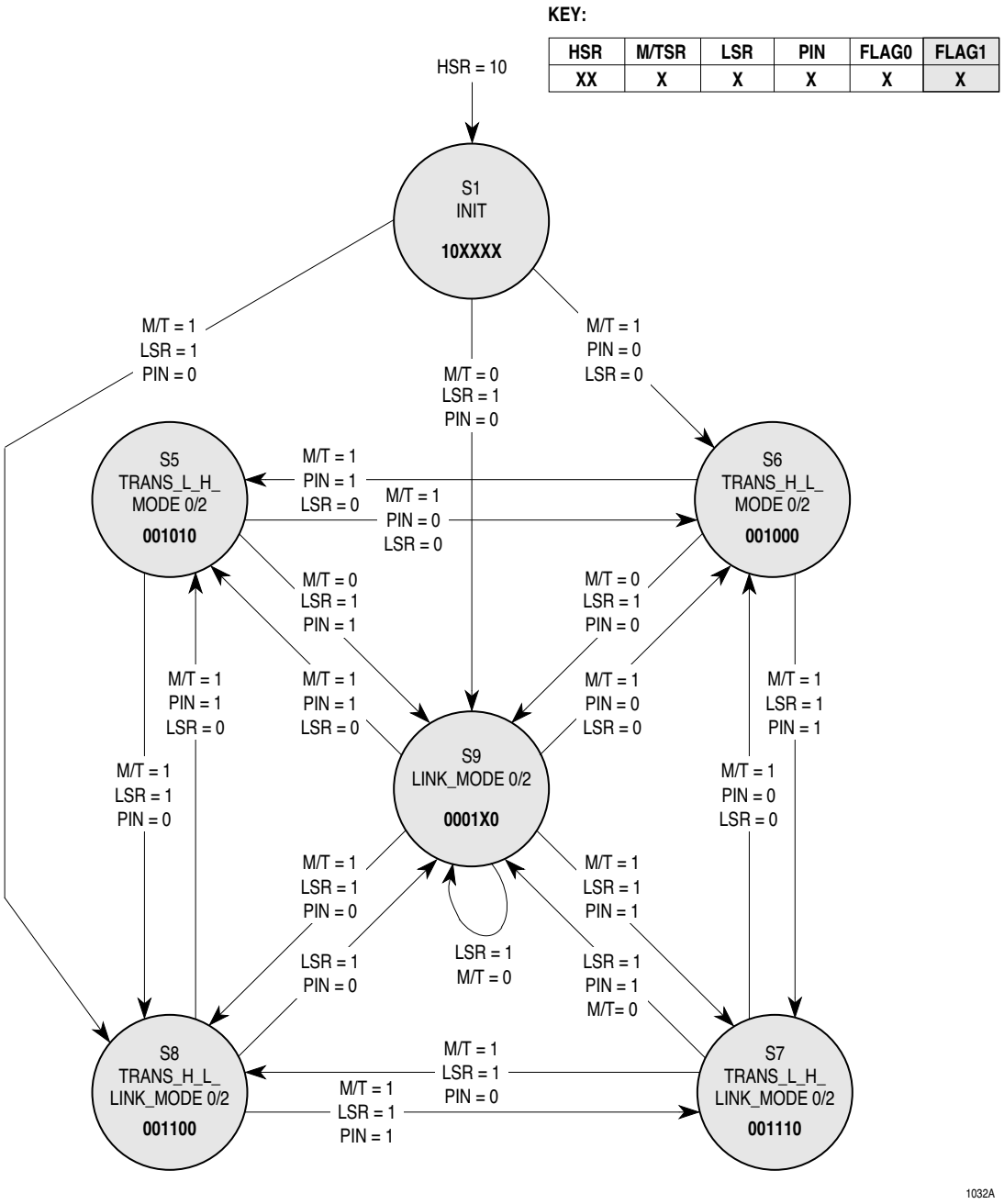



**Figure 6 SPWM State Flowchart, Modes 0 and 2**

NOTES

### *How to Reach Us:*

**Home Page:**
www.freescale.com

**E-mail:**
support@freescale.com

**USA/Europe or Locations Not Listed:**
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

**Europe, Middle East, and Africa:**
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

**Japan:**
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

**Asia/Pacific:**
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

***For Literature Requests Only:***
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com