

# Discrete Input/Output TPU Function (DIO)

by Charles Melear

## 1 Functional Overview

The discrete input/output (DIO) function allows the user to configure a time processor unit (TPU) channel as an input or output. As an input, the channel can be read at any time or sampled at a periodic rate. As an output, the channel can be driven high or low upon command by the CPU.

A parameter RAM location, PIN\_LEVEL, is used to record the 16 most recent states of the TPU channel pin. The programmer may choose one of the four following conditions to update the parameter: 1) when a transition (positive, negative, or either) occurs, 2) when the CPU makes a request to read the logical value driving the pin, 3) when the CPU makes a request to drive the pin to a specified logical value or 4) at a periodic rate specified in the MATCH\_RATE register.

## 2 Detailed Description

The DIO function allows a TPU channel pin to emulate a discrete input or output pin. As an input, the pin can be read either on command or at a periodic rate. As an output, the pin can be driven high or low on command. The DIO function can be used in the following ways:

1. A TPU channel pin can be configured as an output and programmed to update on a host service request. An HSR %10 outputs a low level onto the pin; an HSR %01 outputs a high level. Both types of requests update bit 15 of PIN\_LEVEL with the pin level driven on the pin. Each time a host service request is issued to write a new value to the pin, the contents of the PIN\_LEVEL register are shifted to the right by one bit and the new level on the TPU channel is written into bit 15 of that register. Note that if 16 consecutive commands were issued to drive a TPU channel high, the PIN\_LEVEL register would contain \$FFFF. Likewise, if 16 consecutive commands were issued to write a TPU channel alternately high and low, the PIN\_LEVEL register would contain \$5555. The host sequence bits are not used for HSRs of %10 and %01.
2. A TPU channel can be configured as an input and programmed to update on positive transitions only, negative transitions only, or all transitions. This mode is entered by setting the host sequence bits to %00 and issuing an HSR %11 (initialization).

Transition mode is normally used with the PAC field set to detect either transition. After 16 selected transitions occur, the PIN\_LEVEL register contains \$FFFF (positive edges only), \$0000 (negative edges only) or \$5555 or \$AAAA (both positive and negative edges). The update of the PIN\_LEVEL register occurs any time a selected transition occurs. The time of the update is not recorded.

3. A TPU channel can be configured as an input and programmed to update at match rate by setting the host sequence bits to %01 and then issuing an HSR %11. The match rate is specified in TPU clock cycles; either timer count register 1 (TCR1) or timer count register 2 (TCR2) can be selected. At the rate specified in the MATCH\_RATE register, the contents of the associated PIN\_LEVEL register are shifted to the right by one place and the logic level of the TPU channel is loaded into bit 15 of the PIN\_LEVEL register. Match mode operation is normally used with the input pin configured by PAC to ignore transitions.

Match mode always uses TCR1 to set up the first compare interval during initialization even if TCR2 is specified in the TBS field of the channel control register. Subsequent matches occur at the TCR2 rate as specified by MATCH\_RATE.



4. A TPU channel can be configured as an input and programmed to update on a host service request. This mode is entered by setting the host sequence bits to %00 and issuing an HSR %11 (initialization). To read the pin, set the host sequence bits to %10 and then issue an HSR %11 (initialization). Each time the appropriate HSR is issued the contents of the associated PIN\_LEVEL register are shifted to the right by one place and the logic level of the TPU channel is loaded into bit 15 of the PIN\_LEVEL register.

If a host service request for initialization coincides with a scheduled request by a match or transition, the host request receives priority and the other request is ignored. HSR %11 should not be used when the host sequence bits are also %11, as errors result.

### 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. DIO function code size is:

$$12 \mu \text{ instructions} + 7 \text{ entries} = \mathbf{19 \text{ long words}}$$

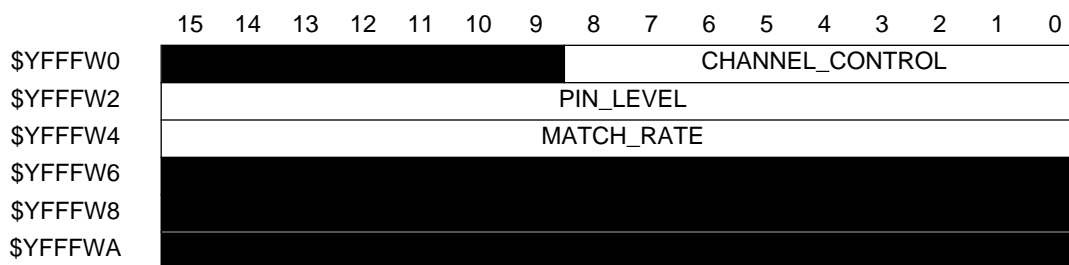
### 4 DIO Function Parameters

This section provides detailed descriptions of discrete input/output function parameters stored in channel parameter RAM. **Figure 1** shows TPU parameter RAM address mapping. **Figure 2** shows the parameter RAM assignment used by the DIO 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 1 TPU Channel Parameter RAM CPU Address Map**



W = Channel number

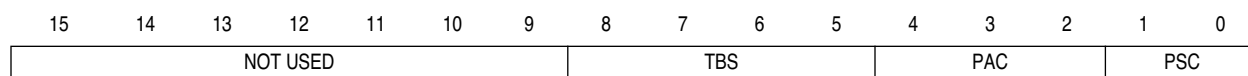
Parameter Write Access:

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

**Figure 2 DIO Function Parameter RAM Assignment**

#### 4.1 CHANNEL\_CONTROL

The CPU should write CHANNEL\_CONTROL prior to issuing an initialization HSR. The CHANNEL\_CONTROL parameter configures the PSC, PAC, and TBS fields. The PSC field is not used by the DIO function and should be set to “no change”. The PAC field specifies the pin logic response for a timer channel input. PAC should be set to “detect either edge” for transition mode and to “no transitions detected” for match mode. For discrete input, the TBS field selects the time base to be used for MATCH\_RATE comparisons. (TCR1 is recommended.) For discrete output, this field is a “don't care”. The following table defines the allowable data for this parameter.



**Table 1 DIO CHANNEL\_CONTROL Options**

TBS	PAC	PSC	Action	
8 7 6 5	4 3 2	1 0	Input	Output
		1 1	—	Do Not Force
	0 0 0		Do Not Detect Transition	—
	0 0 1		Detect Rising Edge	—
	0 1 0		Detect Falling Edge	—
	0 1 1		Detect Either Edge	—
	1 x x		Do Not Change PAC	Do Not Change PAC
<b>0 0 x x</b>			<b>Input Channel</b>	—
0 0 0 0			Capture TCR1, Match TCR1	—
0 0 0 1			Capture TCR1, Match TCR2	—
0 0 1 0			Capture TCR2, Match TCR1	—
0 0 1 1			Capture TCR2, Match TCR2	—
1 x x x			Do Not Change TBS	Do Not Change TBS

## 4.2 PIN\_LEVEL

PIN\_LEVEL indicates the 16 most recent pin values, with the most recent pin value contained in bit 15 and the least recent pin value contained in bit 0. The TPU writes this parameter when a specified transition occurs, or a match or host request to read the pin state occurs, depending on the mode of operation. When updated, the 16 most recent pin values are shifted right by one and the most recent pin value is placed into bit 15. The pin value contained in bit 0 before the right shift is lost after the data is shifted right by one.

## 4.3 MATCH\_RATE

MATCH\_RATE indicates the rate, expressed in cycles of the selected TCR, at which the pin value is recorded in PIN\_LEVEL when the channel is executing match mode operation.

## 5 Host Interface to Function

This section provides information concerning the TPU host interface to the DIO function. **Figure 3** 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 3 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] — 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[15:0]	Action Taken
00	Transition mode: record pin on transition
01	Match mode: record pin at MATCH_RATE
10	Record pin state on HSR%11
11	Indeterminate operation

**HSRR[0:1] — 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[15:0]	Initialization
00	No host service (reset condition)
01	Force high output
10	Force low output
11	Initialize as per host sequence bits

**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[15:0]	Channel Priority
00	Disabled
01	Low
10	Middle
11	High

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

For discrete input, the host CPU initializes the channel by:

1. Writing parameters CHANNEL\_CONTROL and MATCH\_RATE;
2. Writing host sequence bits to configure transition or match mode, as desired;
3. Writing host service request bits to request initialization (%11).

The TPU then executes initialization and accepts an input transition type specified by the PAC field in CHANNEL\_CONTROL, or samples the state of the pin at the rate specified by MATCH\_RATE. The CPU should monitor the HSR register until the TPU clears the service request to %00 before changing any parameters or before issuing a new service request to this channel.

For discrete output, the host CPU initializes the channel by the following:

1. Writing %01 to the HSR bits; causing the TPU to output a high level to the pin; or,
2. Writing %10 to the HSR bits; causing the TPU to output a low level to the pin.

No other initialization is required.

For all modes of discrete input, once initialized, and discrete output, configuring the host sequence bits to %10 and issuing %11 to the HSR bits causes the TPU to read the pin level and to record the level read in bit 15 of PIN\_LEVEL. In the case of discrete output, the pin level read is the state of the output latch, and not the level of the pin at the pad. In all cases, whenever the pin level is recorded in bit 15 of PIN\_LEVEL, an interrupt is generated (if enabled).

Note that to switch from discrete output to discrete input, the host sequence bits must be configured for the proper mode of operation and initialization executed. To switch from discrete input to discrete output, no initialization is required; only the proper HSR must be initiated to force the proper output pin level.

## 7 Performance and Use of Function

### 7.1 Performance

Like all TPU functions, DIO 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. The more TPU channels are active, the more performance decreases. 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 the DIO state timing table below.

**Table 2 DIO State Timing**

State Number and Name	Max CPU Clock Cycles	RAM Accesses by TPU
S1 Init	18	4
S2 Match_PS	10	3
S3 Trans_PS_Low	4	2
S4 Trans_PS_High	4	2
S5 Low_Pin_Request	8	2
S6 High_Pin_Request	8	2

## 7.2 Changing Mode

The host sequence bits are used to select DIO 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.

## 8 Function Examples

The following examples give an indication of the capabilities of the DIO function. Each example includes a description of the example, a diagram of the initial parameter RAM content, a diagram of the output waveform, and a program listing.

### 8.1 Example A

#### 8.1.1 Description

This example sets up TPU channels 0, 5, 10, and 15 as outputs and causes them to go high or low by issuing the appropriate host service requests. Four square waves are generated at frequencies of  $f$ ,  $2f$ ,  $4f$ , and  $8f$ .

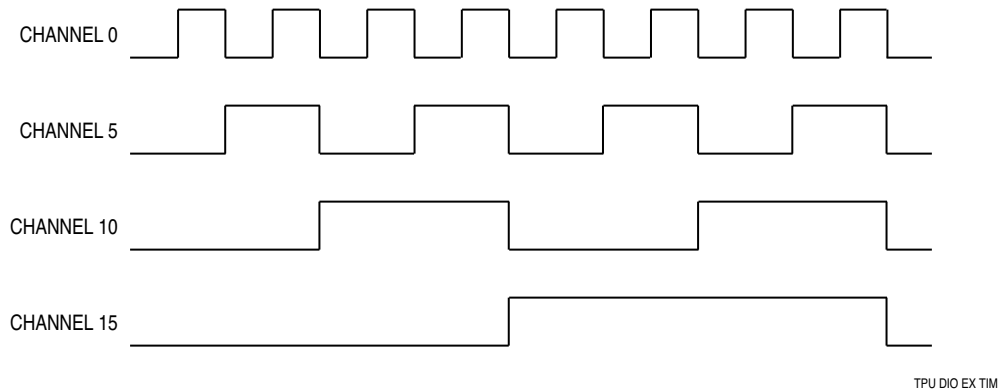
#### 8.1.2 Initialization

Configure the CHANNEL\_CONTROL register for channels 0, 5, 10, and 15 as follows:

**Table 3 DIO CHANNEL\_CONTROL Parameter**

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
\$YFFF00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 0
\$YFFF50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 5
\$YFFFA0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 10
\$YFFFF0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 15

### 8.1.3 Output Waveforms



### 8.1.4 Program Listing

```

1
2 * The following program uses TPU channels 0, 5, 10 and 15 in the Discrete
3 * Input/Output mode. The purpose of this example is to
4 * set up the channels as outputs and then cause them to go high or low
5 * by issuing the appropriate Host Service Requests. Four square waves will
6 * be generated at frequencies of f, 2f, 4f and 8f.
7
8 * This section of the program assigns names to the registers address.
9
00000000      10 TPUMCR equ $ffffe00          ;TPU Module Configuration Reg
00000000      11 TTCR equ $ffffe02
00000000      12 DSCR equ $ffffe04
00000000      13 DSSR equ $ffffe06
00000000      14 TICR equ $ffffe08          ;TPU Interrupt Config. Reg
00000000      15 CIER equ $ffffe0a        ;TPU Ch. Interrupt Enable Reg
00000000      16 CFSR0 equ $ffffe0c       ;TPU Ch. Function Select Reg 0
00000000      17 CFSR1 equ $ffffe0e       ;TPU Ch. Function Select Reg 1
00000000      18 CFSR2 equ $ffffe10       ;TPU Ch. Function Select Reg 2
00000000      19 CFSR3 equ $ffffe12       ;TPU Ch. Function Select Reg 3
00000000      20 HSQR0 equ $ffffe14       ;TPU Host Sequence Register 0
00000000      21 HSQR1 equ $ffffe16       ;TPU Host Sequence Register 1
00000000      22 HSRR0 equ $ffffe18       ;TPU Host Service Req. Reg 0
00000000      23 HSRR1 equ $ffffe1a       ;TPU Host Service Req. Reg 1
00000000      24 CPR0 equ $ffffe1c        ;TPU Channel Priority Reg 0
00000000      25 CPR1 equ $ffffe1e        ;TPU Channel Priority Reg 1
00000000      26 CISR equ $ffffe20        ;Channel Interrupt Status Reg
00000000      27 CH0_CNTL equ $fffff00     ;Channel 0 Control Reg
00000000      28 CH0_PINL equ $fffff02     ;Channel 0 Pin Level Reg
00000000      29 CH0_MATCH equ $fffff04    ;Channel 0 Match Rate Reg
00000000      30 CH5_CNTL equ $fffff50     ;Channel 5 Control Register
00000000      31 CH5_PINL equ $fffff52     ;Channel 5 Pin Level Register
00000000      32 CH5_MATCH equ $fffff54    ;Channel 5 Match Rate Register
00000000      33 CH10_CNTL equ $fffffa0    ;Channel 10 Control Register
00000000      34 CH10_PINL equ $fffffa2    ;Ch. 10 Pin Level Register
00000000      35 CH10_MATCH equ $fffffa4   ;Ch. 10 Match Rate Register
00000000      36 CH15_CNTL equ $ffffff0    ;Ch. 15 Control Register
00000000      37 CH15_PINL equ $ffffff2    ;Ch. 15 Pin Level Register
00000000      38 CH15_MATCH equ $ffffff4   ;Channel 15 Match Rate Register
39
00005000      40 org $5000                ;program origin
41 * This portion of the program initializes the TPU reg
41
00005000 33FC8000      42 init move.w #$8000,(CFSR0).1 ;init ch. 15 to DIO function
00FFFE0C
00005008 33FC0800      43      move.w #$0800,(CFSR1).1 ;init ch. 10 to DIO function
00FFFE0E

```



```

00005010 33FC0080      44      move.w #$0080,(CFSR2).l      ;init ch. 5 to DIO function
          00FFFE10
00005018 33FC0008      45      move.w #$0008,(CFSR3).l      ;init ch. 0 to DIO function
          00FFFE12
00005020 33FC0000      46      move.w #$0000,(HSQR0).l      ;ch. 15,10 pin level
          00FFFE14      ;reg. update on HSR = 11
00005028 33FC0000      47      move.w #$0000,(HSQR1).l      ;ch. 5,0 pin level
          00FFFE16      ;reg. update on HSR = 11
00005030 33FCC030      48      move.w #$c030,(CPR0).l      ;ch. 15,10 - hi priority
          00FFFE1C
00005038 33FC0C03      49      move.w #$0c03,(CPR1).l      ;ch. 5,0 - hi priority
          00FFFE1E
00005040 33FC0003      50      move.w #$0003,(CH15_CNTL).l ;ch. 15 - use TCR1
          00FFFFF0
00005048 33FC0003      51      move.w #$0003,(CH10_CNTL).l ;ch. 10 use TCR1
          00FFFFA0
00005050 33FC0003      52      move.w #$0003,(CH5_CNTL).l  ;ch. 5 use TCR1
          00FFFF50
00005058 33FC0003      53      move.w #$0003,(CH0_CNTL).l ;ch. 0 use TCR1
          00FFFF00
          54
55 * This portion of the program only initializes the DIO channels as outputs.
56 * No action will be taken on an external pin by executing the next two
57 * instructions.
          58
00005060 33FCC030      59      move.w #$c030,(HSRR0).l      ;HSR - init ch. 15,10
          00FFFE18
00005068 33FC0C03      60      move.w #$0c03,(HSRR1).l      ;HSR - init ch. 5,0
          00FFFE1A
61
62 * This portion of the program generates four square waves using software
63 * timing loops.
64
00005070 33FC8020      65 strt move.w #$8020,(HSRR0).l ;HSR - ch. 15-lo, c
          00FFFE18
00005078 33FC0802      65      move.w #$0802,(HSRR1).l      ;HSR ch 5-lo,ch 0-lo
          00FFFE1A
00005080 4EB90000      66      jsr wait
          518C
00005086 33FC0801      67      move.w #$0801,(HSRR1).l      ;HSR - ch. 0 - hi
          00FFFE1A
0000508E 4EB90000      68      jsr wait
          518C
00005094 33FC0402      69      move.w #$0402,(HSRR1).l      ;HSR ch. 5-hi, ch0-lo
          00FFFE1A
0000509C 4EB90000      70      jsr wait
          518C
000050A2 33FC0401      71      move.w #$0401,(HSRR1).l      ;HSR ch 5-hi, ch 0-hi
          00FFFE1A
000050AA 4EB90000      72      jsr wait
          518C
000050B0 33FC0802      73      move.w #$0802,(HSRR1).l      ;HSR ch 5-lo, ch 0-lo
          00FFFE1A
000050B8 33FC0010      74      move.w #$0010,(HSRR0).l      ;HSR - ch. 15 - lo, c
          00FFFE18
000050C0 4EB90000      74      jsr wait
          518C
000050C6 33FC0801      75      move.w #$0801,(HSRR1).l      ;HSR ch 5-lo, ch 0-hi
          00FFFE1A
000050CE 33FC8010      76      move.w #$8010,(HSRR0).l      ;HSR - ch. 15 - lo, c
          00FFFE18
000050D6 4EB90000      76      jsr wait
          518C
000050DC 33FC0402      77      move.w #$0402,(HSRR1).l      ;HSR ch 5-hi, ch 0-lo
          00FFFE1A
000050E4 33FC8010      78      move.w #$8010,(HSRR0).l      ;HSR - ch. 15 - lo, c

```

```

00FFFE18
000050EC 4EB90000 518C 78 jsr wait
000050F2 33FC0401 00FFFE1A 79 move.w #$0401,(HSRR1).1 ;HSR ch 5-hi, ch 0-hi
000050FA 33FC8010 00FFFE18 80 move.w #$8010,(HSRR0).1 ;HSR - ch. 15 - lo, c
00005102 4EB90000 518C 80 jsr wait
00005108 33FC0802 00FFFE1A 81 move.w #$0802,(HSRR1).1 ;HSR ch. 5-lo, ch 0-lo
00005110 33FC4020 00FFFE18 82 move.w #$4020,(HSRR0).1 ;HSR ch. 15-hi, c
00005118 4EB90000 518C 82 jsr wait
0000511E 33FC0801 00FFFE1A 83 move.w #$0801,(HSRR1).1 ;HSR ch. 5-lo, ch 0-hi
00005126 4EB90000 518C 84 jsr wait
0000512C 33FC0402 00FFFE1A 85 move.w #$0402,(HSRR1).1 ;HSR ch 5-hi, ch 0-lo
00005134 4EB90000 518C 86 jsr wait
0000513A 33FC0401 00FFFE1A 87 move.w #$0401,(HSRR1).1 ;HSR ch 5-hi, ch 0-hi
00005142 4EB90000 518C 88 jsr wait
00005148 33FC0802 00FFFE1A 89 move.w #$0802,(HSRR1).1 ;HSR-ch. 5-lo, ch. 0-lo
00005150 33FC4010 00FFFE18 90 move.w #$4010,(HSRR0).1 ;HSR - ch. 15 - hi, c
00005158 4EB90000 518C 90 jsr wait
0000515E 33FC0801 00FFFE1A 91 move.w #$0801,(HSRR1).1 ;HSR ch. 5-lo, ch 0-hi
00005166 4EB90000 518C 92 jsr wait
0000516C 33FC0402 00FFFE1A 93 move.w #$0402,(HSRR1).1 ;HSR ch 5-hi, ch 0-lo
00005174 4EB90000 518C 94 jsr wait
0000517A 33FC0401 00FFFE1A 95 move.w #$0401,(HSRR1).1 ;HSR ch 5-hi, ch 0-lo
00005182 4EB90000 518C 96 jsr wait
00005188 4EF85070 97 jmp strt
0000518C 203C0000 0FFF 98 wait move.l #$fff,d0 ;wait loop
00005192 90BC0000 0001 99 looplsub.l #$1,d0 ;wait loop
00005198 6600FFF8 100 bne loopl ;wait loop
0000519C 4E75 101 rts
102
103
104
105
106

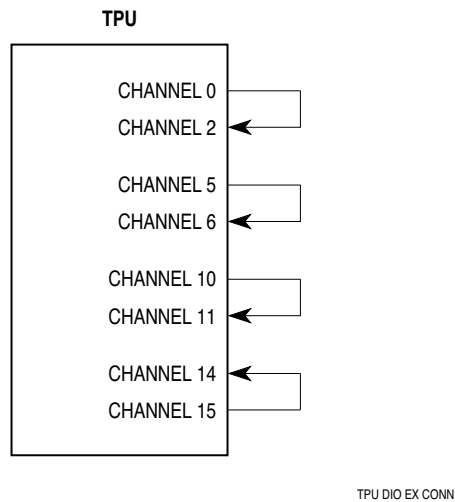
```

## 8.2 Example B

### 8.2.1 Description

This program uses TPU channels 0, 2, 5, 6, 10, 11, 14, and 15 with the discrete output function. The program sets up channels 0, 5, 10, and 15 as outputs and then causes them to go high or low by issuing the appropriate host service requests. Four square waves are generated at frequencies of  $f$ ,  $2f$ ,  $4f$ , and  $8f$ . In addition, TPU channels 2, 6, 11, and 14 are configured as inputs and update their PIN\_LEVEL registers on each transition. Channel 0 drives channel 2, channel 5 drives channel 6, channel 10 drives channel 11, and channel 16 drives channel 14. The PIN\_LEVEL registers of the input channels will all contain \$5555 or \$AAAA after 16 transitions have been detected on the slowest channel.

### 8.2.2 Hardware Configuration



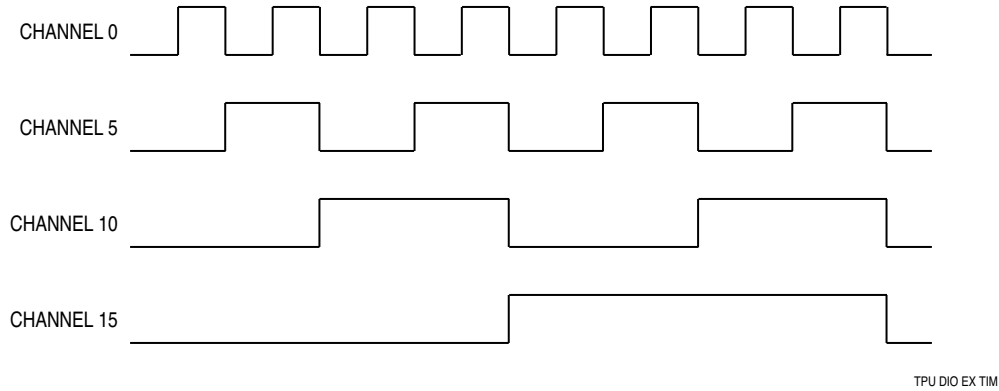
### 8.2.3 Initialization

Configure the CHANNEL\_CONTROL registers as follows:

**Table 4 DIO CHANNEL\_CONTROL Parameter**

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
\$YFFF00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 0
\$YFFF50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 5
\$YFFFA0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 10
\$YFFFF0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 15
\$YFFF20	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Channel 2
\$YFFF60	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Channel 6
\$YFFFB0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Channel 11
\$YFFFE0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Channel 14

## 8.2.4 Output Waveforms



## 8.2.5 Program Listing

```

1
2 * The following program uses TPU channels 0, 2, 5, 6, 10, 11, 14 and 15 in the
3 * Discrete Input/Output mode. This program sets up channels 0, 5, 10 and 15
4 * as outputs and then cause them to go high or low
5 * by issuing the appropriate Host Service Requests. Four square waves will
6 * be generated at frequencies of f, 2f, 4f and 8f. TPU channels 2, 6,
7 * 11 and 14 will be configured as inputs and will update their PIN_LEVEL
8 * Registers on each transition. Channel 0 will drive channel 2, channel 5
9 * will drive channel 6, channel 10 will drive channel 11 and channel 15 will
10 * drive channel 14. The PIN_LEVEL Registers of the input channels should all
11 * contain $5555 or $AAAA after 16 transitions have been detected on the
12 * slowest channel.
13
14 * This section of the program assigns register names to the reg. address.
15
00000000      16 TPUMCR equ $ffffe00          ;TPU Module Config. Reg
00000000      17 TTCR equ $ffffe02
00000000      18 DSCR equ $ffffe04
00000000      19 DSSR equ $ffffe06
00000000      20 TICR equ $ffffe08          ;TPU Interrupt Config. Reg
00000000      21 CIER equ $ffffe0a        ;TPU Channel Interrupt Enable Reg
00000000      22 CFSR0 equ $ffffe0c       ;TPU Ch. Function Select Reg 0
00000000      23 CFSR1 equ $ffffe0e       ;TPU Ch. Function Select Reg 1
00000000      24 CFSR2 equ $ffffe10       ;TPU Ch. Function Select Reg 2
00000000      25 CFSR3 equ $ffffe12       ;TPU Ch. Function Select Reg 3
00000000      26 HSQR0 equ $ffffe14       ;TPU Host Sequence Reg 0
00000000      27 HSQR1 equ $ffffe16       ;TPU Host Sequence Reg 1
00000000      28 HSRR0 equ $ffffe18       ;TPU Host Svc. Request Reg 0
00000000      29 HSRR1 equ $ffffe1a       ;TPU Host Svc Request Reg 1
00000000      30 CPR0 equ $ffffe1c        ;TPU Channel Priority Reg 0
00000000      31 CPR1 equ $ffffe1e        ;TPU Channel Priority Reg 1
00000000      32 CISR equ $ffffe20        ;Channel Int. Status Reg
00000000      33 CH0_CNTL equ $fffff00     ;Channel 0 Control Register
00000000      34 CH0_PINL equ $fffff02     ;Channel 0 Pin Level Reg
00000000      35 CH0_MATCH equ $fffff04    ;Channel 0 Match Rate Reg
00000000      36 CH2_CNTL equ $fffff20     ;Channel 2 Control Reg
00000000      37 CH2_PINL equ $fffff22     ;Channel 2 Pin Level Reg
00000000      38 CH2_MATCH equ $fffff24    ;Channel 2 Match Rate Reg
00000000      39 CH5_CNTL equ $fffff50     ;Channel 5 Control Reg
00000000      40 CH5_PINL equ $fffff52     ;Channel 5 Pin Level Reg
00000000      41 CH5_MATCH equ $fffff54    ;Channel 5 Match Rate Reg
00000000      42 CH6_CNTL equ $fffff60     ;Channel 6 Control Reg
00000000      43 CH6_PINL equ $fffff62     ;Channel 6 Pin Level Reg
00000000      44 CH6_MATCH equ $fffff64    ;Channel 6 Match Rate Reg
00000000      45 CH10_CNTL equ $fffffa0    ;Channel 10 Control Reg
00000000      46 CH10_PINL equ $fffffa2    ;Channel 10 Pin Level Reg

```

```

00000000      47 CH10_MATCH equ $ffffa4      ;Channel 10 Match Rate Reg
00000000      48 CH11_CNTL equ $ffffb0     ;Channel 11 Control Reg
00000000      49 CH11_PINL equ $ffffb2     ;Channel 11 Pin Level Reg
00000000      50 CH11_MATCH equ $ffffb4     ;Channel 11 Match Rate Reg
00000000      51 CH14_CNTL equ $ffffe0     ;Channel 14 Control Reg
00000000      52 CH14_PINL equ $ffffe2     ;Channel 14 Pin Level Reg
00000000      53 CH14_MATCH equ $ffffe4     ;Channel 14 Match Rate Reg
00000000      54 CH15_CNTL equ $fffff0     ;Channel 15 Control Reg
00000000      55 CH15_PINL equ $fffff2     ;Channel 15 Pin Level Reg
00000000      56 CH15_MATCH equ $fffff4     ;Channel 15 Match Rate Reg
57
00005000      58      org $5000                ;program origin
59 * This portion of the program initializes the TPU registers
59
00005000 33FC8800      60 init move.w #$8800,(CFSR0).l      ;init ch 15,14 to DIO
      00FFFE0C
00005008 33FC8800      61      move.w #$8800,(CFSR1).l      ;init ch 10,11 to DIO
      00FFFE0E
00005010 33FC0880      62      move.w #$0880,(CFSR2).l      ;init ch 5,6 to DIO
      00FFFE10
00005018 33FC0808      63      move.w #$0808,(CFSR3).l      ;init ch 0,2 to DIO
      00FFFE12
00005020 33FC0000      64      move.w #$0000,(HSQR0).l      ;ch. 15,14,11,10 pin level
      00FFFE14      ;reg. update on transition
00005028 33FC0000      65      move.w #$0000,(HSQR1).l      ;ch.5,6,2,0 pin level
      00FFFE16      ;reg. update on transition
00005030 33FC0F0F      66      move.w #$f0f0,(CPR0).l          ;ch. 15,14,10,11 have
      00FFFE1C      ;high priority
00005038 33FC3C33      67      move.w #$3c33,(CPR1).l          ;ch. 5,6 and 2,0 have
      00FFFE1E      ;high priority
00005040 33FC0003      68      move.w #$0003,(CH15_CNTL).l ;ch. 15 use TCR1
      00FFFFF0
00005048 33FC000F      69      move.w #$000f,(CH14_CNTL).l ;ch. 14 use TCR1
      00FFFFE0      ;detect both edges
00005050 33FC000F      70      move.w #$000f,(CH11_CNTL).l ;ch. 11 use TCR1
      00FFFFB0      ;detect both edges
00005058 33FC0003      71      move.w #$0003,(CH10_CNTL).l ;ch. 10 use TCR1
      00FFFFA0
00005060 33FC000F      72      move.w #$000f,(CH6_CNTL).l ;ch. 6 use TCR1
      00FFFF60      ;detect both edges
00005068 33FC0003      73      move.w #$0003,(CH5_CNTL).l ;ch. 5 use TCR1
      00FFFF50
00005070 33FC000F      74      move.w #$000f,(CH2_CNTL).l ;ch. 2 use TCR1
      00FFFF20      ;detect both edges
00005078 33FC0003      75      move.w #$0003,(CH0_CNTL).l ;ch. 0 use TCR1
      00FFFF00
76
77 * This portion of the program initializes the DIO channels as inputs and
78 * outputs, as required. No action
79 * will be taken on an external pin by executing the next two instructions.
80
00005080 33FC0F0F      81      move.w #$f0f0,(HSRR0).l      ;HSR-ch 15,14,11,10
      00FFFE18
00005088 33FC3C33      82      move.w #$3c33,(HSRR1).l      ;HSR-ch.5,6,2,0
      00FFFE1A
83
84 * This portion of the program generates four square waves using software
85 * timing loops on channels 0, 5, 10 and 15. The transitions from these
86 * channels are recorded on channels 2, 6, 11 and 14, respectively.
87
00005090 33FC8020      88 strt move.w #$8020,(HSRR0).l      ;HSR - ch. 15 - lo, c
      00FFFE18
00005098 33FC0802      88      move.w #$0802,(HSRR1).l      ;HSR-ch 5-lo, ch 0-lo
      00FFFE1A
000050A0 4EB90000      89      jsr wait
      51AC

```

000050A6	33FC0001 00FFFE1A	90	move.w #\$0001,(HSRR1).1	;HSR - ch. 0 - hi
000050AE	4EB90000 51AC	91	jsr wait	
000050B4	33FC0402 00FFFE1A	92	move.w #\$0402,(HSRR1).1	;HSR-ch 5-hi, ch 0 lo
000050BC	4EB90000 51AC	93	jsr wait	
000050C2	33FC0001 00FFFE1A	94	move.w #\$0001,(HSRR1).1	;HSR-ch 5-hi, ch 0-hi
000050CA	4EB90000 51AC	95	jsr wait	
000050D0	33FC0802 00FFFE1A	96	move.w #\$0802,(HSRR1).1	;HSRch. 5-lo, ch 0-lo
000050D8	33FC0010 00FFFE18	97	move.w #\$0010,(HSRR0).1	;HSR - ch. 15 - lo, c
000050E0	4EB90000 51AC	97	jsr wait	
000050E6	33FC0001 00FFFE1A	98	move.w #\$0001,(HSRR1).1	;HSR-ch. 5-lo, ch 0-hi
		99		
000050EE	4EB90000 51AC	100	jsr wait	
000050F4	33FC0402 00FFFE1A	101	move.w #\$0402,(HSRR1).1	;HSR-ch 5-hi, ch 0-lo
		102		
000050FC	4EB90000 51AC	103	jsr wait	
00005102	33FC0001 00FFFE1A	104	move.w #\$0001,(HSRR1).1	;HSR-ch 5-hi, ch 0-hi
		105		
0000510A	4EB90000 51AC	106	jsr wait	
00005110	33FC0802 00FFFE1A	107	move.w #\$0802,(HSRR1).1	;HSR-ch. 5-lo, ch0-lo
00005118	33FC4020 00FFFE18	108	move.w #\$4020,(HSRR0).1	;HSR - ch. 15 - hi, c
00005120	4EB90000 51AC	108	jsr wait	
00005126	33FC0001 00FFFE1A	109	move.w #\$0001,(HSRR1).1	;HSR-ch. 5-lo, ch 0-hi
0000512E	4EB90000 51AC	110	jsr wait	
00005134	33FC0402 00FFFE1A	111	move.w #\$0402,(HSRR1).1	;HSR-ch 5-hi, ch0-lo
0000513C	4EB90000 51AC	112	jsr wait	
00005142	33FC0001 00FFFE1A	113	move.w #\$0001,(HSRR1).1	;HSR-ch 5-hi, ch 0-hi
0000514A	4EB90000 51AC	114	jsr wait	
00005150	33FC0802 00FFFE1A	115	move.w #\$0802,(HSRR1).1	;HSR-ch 5-lo, ch 0-lo
00005158	33FC0010 00FFFE18	116	move.w #\$0010,(HSRR0).1	;HSR-ch 15-hi, c
00005160	4EB90000 51AC	116	jsr wait	
00005166	33FC0001 00FFFE1A	117	move.w #\$0001,(HSRR1).1	;HSR-ch 5-lo, ch 0 hi
0000516E	4EB90000 51AC	118	jsr wait	
00005174	33FC0402 00FFFE1A	119	move.w #\$0402,(HSRR1).1	;HSR-ch 5-hi, ch 0-lo
0000517C	4EB90000 51AC	120	jsr wait	
00005182	33FC0001	121	move.w #\$0001,(HSRR1).1	;HSR-ch 5-hi, ch 0-lo

```

00FFFE1A
0000518A 4EB90000    122    jsr wait
          51AC
00005190 383900FF    123    move.w (CH2_PINL).l,d4    ;pin level 2 to D4
          FF22
00005196 3A3900FF    124    move.w (CH6_PINL).l,d5    ;pin level 5 to D5
          FF62
0000519C 3C3900FF    125    move.w (CH11_PINL).l,d6   ;pin level 11 to D6
          FFB2
000051A2 3E3900FF    126    move.w (CH14_PINL).l,d7   ;pin level 14 to D7
          FFE2
000051A8 4EF85090    127    jmp strt
000051AC 203C0000    128 wait move.l #$fff,d0    ;wait loop
          0FFF
000051B2 90BC0000    129 lop1 sub.l #$1,d0             ;wait loop
          0001
000051B8 6600FFF8    130    bne lop1                 ;wait loop
000051BC 4E75        131    rts
          132

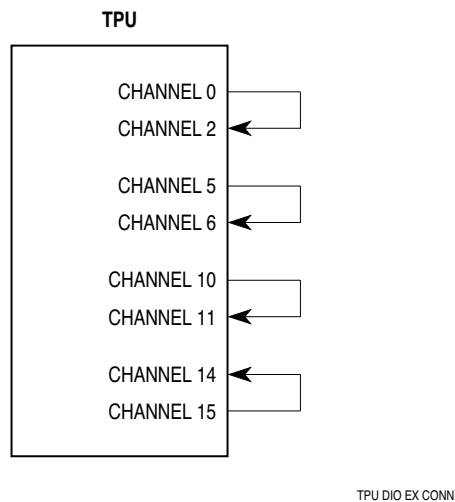
```

## 8.3 Example C

### 8.3.1 Description

This program uses TPU channels 0, 2, 5, 6, 10, 11, 14, and 15 with the discrete output function. The program sets up channels 0, 5, 10, and 15 as outputs and then causes them to go high or low by issuing the appropriate host service requests. Four square waves are generated at frequencies of  $f$ ,  $2f$ ,  $4f$ , and  $8f$ . In addition, TPU channels 2, 6, 11, and 14 are configured as inputs and programmed to update at match rate. Channel 0 drives channel 2, channel 5 drives channel 6, channel 10 drives channel 11, and channel 16 drives channel 14.

### 8.3.2 Hardware Configuration



### 8.3.3 Initialization

Configure the CHANNEL\_CONTROL and MATCH\_RATE registers as follows:

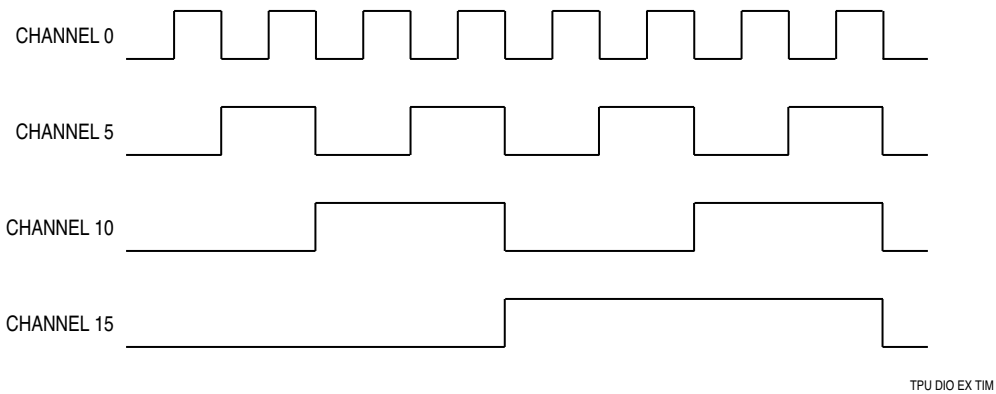
**Table 5 DIO CHANNEL\_CONTROL Parameter**

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
\$YFFF00	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 0
\$YFFF50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 5
\$YFFFA0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 10
\$YFFFF0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	Channel 15
\$YFFF20	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Channel 2
\$YFFF60	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Channel 6
\$YFFFB0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Channel 11
\$YFFFE0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Channel 14

**Table 6 DIO MATCH\_RATE Parameter**

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
\$YFFF24	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	Channel 2
\$YFFF64	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	Channel 6
\$YFFFB4	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	Channel 11
\$YFFFE4	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	Channel 14

**8.3.4 Output Waveforms**



**8.3.5 Program Listing**

```

1
2
3 * The following program uses TPU channels 0, 2, 5, 6, 10, 11, 14 and 15
4 * in the Discrete Input/Output mode.
5 *
6 * Channels 0, 5, 10 and 15 are set up as outputs and then caused to go
7 * high or low by issuing the appropriate Host Service Requests. Four
8 * square waves will be generated at frequencies of f, 2f, 4f and 8f.
9 * Channels 2, 6, 11 and 14 are set up in the Discrete Input/Output mode and
10 * programmed to Update at Match Rate. Channel 0 is connected to channel 2,
11 * channel 5 is connected to channel 6, channel 10 is connected to channel 11
12 * and channel 15 is connected to channel 14
13
14 * This section of the program assigns names to the register's address.
15
16 TPUMCR equ $fffe00           ;TPU Module Config.Reg
17 TTCR equ $fffe02           ;

```



```

00000000      18 DSCR equ $fffe04      ;
00000000      19 DSSR equ $fffe06      ;
00000000      20 TICR equ $fffe08      ;Interrupt Config Reg
00000000      21 CIER equ $fffe0a      ;Ch. Int. Enable Reg
00000000      22 CFSR0 equ $fffe0c      ;TPU Ch Function Select Reg 0
00000000      23 CFSR1 equ $fffe0e      ;TPU Ch Funct Select Reg 1
00000000      24 CFSR2 equ $fffe10      ;TPU Ch Funct Select Reg 2
00000000      25 CFSR3 equ $fffe12      ;TPU Ch Funct Select Reg 3
00000000      26 HSQR0 equ $fffe14      ;Host Sequence Reg 0
00000000      27 HSQR1 equ $fffe16      ;Host Sequence Reg 1
00000000      28 HSRR0 equ $fffe18      ;Host Service Req. Reg 0
00000000      29 HSRR1 equ $fffe1a      ;Host Service Req Reg 1
00000000      30 CPR0 equ $fffe1c      ;Ch. Priority Reg 0
00000000      31 CPR1 equ $fffe1e      ;Ch Priority Re 1
00000000      32 CISR equ $fffe20      ;Interrupt Status Reg
00000000      33 CH0_CNTL equ $ffff00      ;Ch. 0 Control Reg
00000000      34 CH0_PINL equ $ffff02      ;Ch. 0 Pin Level Reg
00000000      35 CH0_MATCH equ $ffff04      ;Ch 0 Match Rate Reg
00000000      36 CH2_CNTL equ $ffff20      ;Ch 2 Control Reg
00000000      37 CH2_PINL equ $ffff22      ;Ch 2 Pin Level Reg
00000000      38 CH2_MATCH equ $ffff24      ;Ch 2 Match Rate Reg
00000000      39 CH5_CNTL equ $ffff50      ;Ch 5 Control Reg
00000000      40 CH5_PINL equ $ffff52      ;Ch 5 Pin Level Reg
00000000      41 CH5_MATCH equ $ffff54      ;Ch 5 Match Rate Reg
00000000      42 CH6_CNTL equ $ffff60      ;Ch 6 Control Reg
00000000      43 CH6_PINL equ $ffff62      ;Ch 6 Pin Level Reg
00000000      44 CH6_MATCH equ $ffff64      ;Ch 6 Match Rate Reg
00000000      45 CH10_CNTL equ $ffffa0      ;Ch 10 Control Reg
00000000      46 CH10_PINL equ $ffffa2      ;Ch 10 Pin Level Reg
00000000      47 CH10_MATCH equ $ffffa4      ;Ch 10 Match Rate Reg
00000000      48 CH11_CNTL equ $ffffb0      ;Ch 11 Control Reg
00000000      49 CH11_PINL equ $ffffb2      ;Ch 11 Pin Level Reg
00000000      50 CH11_MATCH equ $ffffb4      ;Ch 11 Match Rate Reg
00000000      51 CH14_CNTL equ $ffffe0      ;Ch 14 Control Reg
00000000      52 CH14_PINL equ $ffffe2      ;Ch 14 Pin Level Reg
00000000      53 CH14_MATCH equ $ffffe4      ;Ch 14 Match Rate Reg
00000000      54 CH15_CNTL equ $fffff0      ;Ch 15 Control Reg
00000000      55 CH15_PINL equ $fffff2      ;Ch 15 Pin Level Reg
00000000      56 CH15_MATCH equ $fffff4      ;Ch 15 Match Rate Reg
00005000      57
00005000      58      org $5000      ;program origin
59 * This portion of the program initializes the TPU reg
00005000 33FC8800      60 init move.w #$8800,(CFSR0).1 ;init ch 1,5,14 to DIO
00FFFE0C
00005008 33FC8800      61      move.w #$8800,(CFSR1).1 ;init ch 10,11 to DIO
00FFFE0E
00005010 33FC0880      62      move.w #$0880,(CFSR2).1 ;init ch,6 to DIO
00FFFE10
00005018 33FC0808      63      move.w #$0808,(CFSR3).1 ;init ch 0,2 to DIO
00FFFE12
00005020 33FC1040      64      move.w #$1040,(HSQR0).1 ;ch. 15,10 pin level
00FFFE14      65 ;update on transition
;ch. 14,11 update at
;match rate
00005028 33FC1010      66      move.w #$1010,(HSQR1).1 ;ch. 5,0 pin level
00FFFE16      67 ;update on transition
;ch. 2 and 6 update at
;match rate
00005030 33FCF0F0      68      move.w #$f0f0,(CPR0).1 ;ch. 15,14,10,11 have
00FFFE1C ;high priority
00005038 33FC3C33      69      move.w #$3c33,(CPR1).1 ;ch. 5,6 and 2,0 have
00FFFE1E ;high priority
00005040 33FC0003      70      move.w #$0003,(CH15_CNTL).1;ch. 15 use TCR1
00FFFFF0
00005048 33FC000F      71      move.w #$000f,(CH14_CNTL).1;ch. 14 use TCR1

```

```

00FFFFE0                                ;detect both edges
00005050 33FC0900 72  move.w #$0900,(CH14_MATCH).l;ch.14 match-$900
00FFFFE4
00005058 33FC000F 73  move.w #$000f,(CH11_CNTL).l;ch. 11 use TCR1,
00FFFFB0                                ;detect both edges
00005060 33FC0900 74  move.w #$0900,(CH11_MATCH).l;ch. 11 match-$900
00FFFFB4
00005068 33FC0003 75  move.w #$0003,(CH10_CNTL).l ;ch. 10 use TCR1
00FFFFA0
00005070 33FC000F 76  move.w #$000f,(CH6_CNTL).l ;ch. 6 use TCR1
00FFFF60                                ;detect both edges
00005078 33FC0900 77  move.w #$0900,(CH6_MATCH).l ;ch. 6 match-$900
00FFFF64
00005080 33FC0003 78  move.w #$0003,(CH5_CNTL).l ;ch. 5 use TCR1
00FFFF50
00005088 33FC000F 79  move.w #$000f,(CH2_CNTL).l ;ch. 2 use TCR1
00FFFF20                                ;detect both edges
00005090 33FC0900 80  move.w #$0900,(CH2_MATCH).l ;ch. 2 match-$900
00FFFF24
00005098 33FC0003 81  move.w #$0003,(CH0_CNTL).l ;ch. 0 use TCR1
00FFFF00
82
83 * This portion of the program initializes the DIO channels as inputs and
84 * outputs as specified. No action
85 * will be taken on an external pin by executing the next two instructions.
86
000050A0 33FCF0F0 87  move.w #$f0f0,(HSRR0).l ;HSR to init ch.
00FFFFE8                                ;15,14,11,10
000050A8 33FC3C33 88  move.w #$3c33,(HSRR1).l ;HSR to init ch. 5,6,2,0
00FFFFE1A
89
90 * This portion of the program generates four square waves using software
91 * timing loops.
92
000050B0 33FC8020 93 strt move.w #$8020,(HSRR0).l ;HSR - ch. 15 - lo, c
00FFFFE18
000050B8 33FC0802 93  move.w #$0802,(HSRR1).l ;HSR-ch 5-lo,ch 0-lo
00FFFFE1A
000050C0 4EB85194 94  jsr wait
000050C4 33FC0001 95  move.w #$0001,(HSRR1).l ;HSR - ch. 0 - hi
00FFFFE1A
000050CC 4EB85194 96  jsr wait
000050D0 33FC0402 97  move.w #$0402,(HSRR1).l ;HSR-ch. 5-hi, ch 0-lo
00FFFFE1A
000050D8 4EB85194 98  jsr wait
000050DC 33FC0001 99  move.w #$0001,(HSRR1).l ;HSR-ch. 5-hi, ch - hi
00FFFFE1A
000050E4 4EB85194 100 jsr wait
000050E8 33FC0802 101 move.w #$0802,(HSRR1).l ;HSR-ch. 5-lo, ch 0 lo
00FFFFE1A
000050F0 33FC0010 102 move.w #$0010,(HSRR0).l ;HSR-ch. 15-lo, c
00FFFFE18
000050F8 4EB85194 102 jsr wait
000050FC 33FC0001 103 move.w #$0001,(HSRR1).l ;HSR-ch. 5-lo, ch 0-hi
00FFFFE1A
00005104 4EB85194 104 jsr wait
00005108 33FC0402 105 move.w #$0402,(HSRR1).l ;HSR-ch. 5-hi, ch 0-10
00FFFFE1A
00005110 4EB85194 106 jsr wait
00005114 33FC0001 107 move.w #$0001,(HSRR1).l ;HSR-ch. 5-hi, ch 0-hi
00FFFFE1A
0000511C 4EB85194 108 jsr wait
00005120 33FC0802 109 move.w #$0802,(HSRR1).l ;HSR-ch. 5-lo, ch 0-lo
00FFFFE1A
00005128 33FC4020 110 move.w #$4020,(HSRR0).l ;HSR-ch. 15-hi, c
00FFFFE18

```

```

00005130 4EB85194      110      jsr wait
00005134 33FC0001      111      move.w #$0001,(HSRR1).l      ;HSR-ch. 5-lo, ch 0-hi
      00FFFE1A
0000513C 4EB85194      112      jsr wait
00005140 33FC0402      113      move.w #$0402,(HSRR1).l      ;HSR-ch. 5-hi, ch 0-lo
      00FFFE1A
00005148 4EB85194      114      jsr wait
0000514C 33FC0001      115      move.w #$0001,(HSRR1).l      ;HSR-ch 5-hi, ch 0-hi
      00FFFE1A
00005154 4EB85194      116      jsr wait
00005158 33FC0802      117      move.w #$0802,(HSRR1).l      ;HSR-ch. 5-lo, ch 0-lo
      00FFFE1A
00005160 33FC0010      118      move.w #$0010,(HSRR0).l      ;HSR - ch. 15 - hi, c
      00FFFE18
00005168 4EB85194      118      jsr wait
0000516C 33FC0001      119      move.w #$0001,(HSRR1).l      ;HSR-ch. 5-lo, ch 0-hi
      00FFFE1A
00005174 4EB85194      120      jsr wait
00005178 33FC0402      121      move.w #$0402,(HSRR1).l      ;HSR-ch. 5-hi, ch 0-lo
      00FFFE1A
00005180 4EB85194      122      jsr wait
00005184 33FC0001      123      move.w #$0001,(HSRR1).l      ;HSR-ch. 5-hi, ch 0-lo
      00FFFE1A
0000518C 4EB85194      124      jsr wait
00005190 4EF850B0      125      jmp strt
00005194 203C0000      126      waitmove.l #$fff,d0          ;wait loop
      0FFF
0000519A 90BC0000      127      lop1sub.l #$1,d0            ;wait loop
      0001
000051A0 6600FFF8      128      bne lop1                    ;wait loop
000051A4 383900FF      129      move.w ($ffff22).l,d4      ;ch 2 match rate reg FF22
      ;to D4
000051AA 3A3900FF      130      move.w ($ffff62).l,d5      ;ch 6 match rate reg FF62
      ; in D5
000051B0 3C3900FF      131      move.w ($ffffb2).l,d6      ;ch 11 match rate reg FFB2
      ;to D11
000051B6 3E3900FF      132      move.w ($ffffe2).l,d7      ;ch 14 match rate reg FFE2
      ;to D14
000051BC 4E75          133      rts
      134

```

## 9 Function State Descriptions

This section describes the states entered for each of the four DIO cases (request for initialization, update on match rate, update on transition, and set pin low or high). Refer to **10 Function Algorithm** for detailed descriptions of each state.

A host service request can be issued at any time and from any state. To begin, assume that the channel function select register, host sequence register, channel priority register and the channel parameter RAM have all been programmed. At this point the channel will be ready to receive its first host service request via the host service request register.

### 9.1 CASE 1: Request for Initialization

HSR = 11, INITIALIZATION  
 HSQ = 10, UPDATE ON HSR = 11

State 1 is entered. The channel pin is not configured as an output or an input; it is simply left in its current condition. The contents of the PIN\_LEVEL register are shifted to the right by one bit and the new pin state is recorded in bit 15 of the PIN\_LEVEL register. Out of reset, the TPU channel pins are inputs. In general, a TPU channel could be either an input or an output. The level that is written into bit 15 of the PIN\_LEVEL register will either be the level driven into the TPU channel if the channel is currently configured as an input or the level being driven by the TPU channel if the channel is configured as an output.

It is most important to recognize that for the case being discussed, i.e., the HSR bits = 11 (initialization) and the HSQ bits = 10 (mode 2), the S1 state is simply executed, exited, and no further action is taken. Realize that S2, S3 and S4 cannot be entered unless the algorithm is currently in S1. Therefore, the HSQ bits cannot be equal to %10 if a channel is to be configured as an input and sampled at the match rate or on each transition.

## 9.2 CASE 2: Update on MATCH\_RATE

HSR = 11, INITIALIZATION  
HSQ = 01, UPDATE ON MATCH RATE

State 1 is entered where the channel pin is configured as an input. The algorithm exits state 1 and goes to state 2. The algorithm will remain in state 2 until a new host service request is issued. While in state 2, the TPU channel pin, already configured as an input, is continually sampled at the periodic rate determined by the MATCH\_RATE register. Each time the TPU channel is sampled the contents of the PIN\_LEVEL register are shifted to the right by one bit and the new pin state is recorded in bit 15 of the PIN\_LEVEL register. The algorithm remains in state 2.

## 9.3 CASE 3: Update on Transition

HSR = 11, INITIALIZATION  
HSQ = 00, UPDATE ON TRANSITION

State 1 is entered where the channel pin is configured as in input. The algorithm exits state 1 and goes to either state 3 if the TPU channel pin is currently at a logic 0 or state 4 if the TPU channel pin is currently at a logic 1.

The DIO algorithm can be programmed to recognize positive edges only, negative edges only or both positive and negative edges. If negative edges only are selected, the algorithm goes to and remains in state 3. The PIN\_LEVEL register is updated with a logic 0 each time a negative transition occurs. If positive edges only are selected, the algorithm goes to and remains in state 4. The PIN\_LEVEL register is updated with a logic 1 each time a positive transition occurs. If both positive and negative edges are selected, either state 3 or state 4 is entered depending upon whether the next transition is negative or positive, respectively. After either state 3 or state 4 is entered, the algorithm alternates between the two states with each new transition.

## 9.4 CASE 4A: Set Pin Low

HSR = 10, SET PIN LOW  
HSQ = 10, UPDATE ON HSR = 11

State 5 is entered where the channel pin is configured as an output and the pin is driven to a logic 0. The contents of the PIN\_LEVEL register are shifted to the right by one bit and a logic 0 is recorded in bit 15 of the PIN\_LEVEL register. The algorithm remains in state 5 until a new host service request is issued.

## 9.5 CASE 4B: Set Pin High

HSR = 01, SET PIN HIGH  
HSQ = 10, UPDATE ON HSR = 11

State 6 is entered where the channel pin is configured as an output and the pin is driven to a logic 1. The contents of the PIN\_LEVEL register are shifted to the right by one bit and a logic 1 is recorded in bit 15 of the PIN\_LEVEL register. The algorithm remains in state 6 until a new host service request is issued.

## 10 Function Algorithm

The DIO time function consists of six states, described in the following paragraphs. 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 Motorola Freeware bulletin board. Refer to *Using the TPU Function Library and TPU Emulation Mode (TPUPN00/D)* for detailed instructions on downloading and compiling microcode.

### 10.1 State 1: *InitRecord\_PS*

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

Match Enable: Don't Care A.1.1 State 1 Init/Record\_PS

Summary:

This state is entered as a result of HSR %11. In this state the channel is either configured to perform modes 1 or 2 discrete input or, if host sequence equals 10, the channel is not configured and the pin state is recorded in bit 15 of PIN\_LEVEL. The previous pin states are shifted right by one, losing the least recent pin state in bit 0. An interrupt request is then generated. Flag0 is used internally to indicate one of two modes of operation when configured for discrete input. When clear, flag0 indicates transition mode operation; when set, it indicates match mode operation.

```
If host sequence bit 1 = 1 {
    Bit N replaced by bit N+1 of parameter PLV
    Bit 15 of parameter PLV gets pin state
    Assert interrupt request
}
Else {
    Configure the channel latches via CHANNEL_CONTROL
    Clear flag0
    ERT replaced by TCR1
    If host sequence bit 0 = 0 {
        Bit N replaced by bit N+1 of parameter PLV
        Bit 15 of parameter PLV gets pin state
        Assert interrupt request
    }
    Else {
        Generate match at ERT + MATCH_RATE
        Assert flag0
        Bit N replaced by bit N+1 of parameter PLV
        Bit 15 of parameter PLV gets pin state
        Assert interrupt request
    }
}
}
```

#### 10.1.1 State 2: *Match\_PS*

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 001xx1

Match Enable: Disable

Summary:

This state is entered due to a match when the channel is configured for discrete input match mode operation. In this state a new match time is scheduled by adding the last match time to MATCH\_RATE. Bit 15 of PIN\_LEVEL is updated with the pin level recorded at the time of service, and the previous pin states are shifted right by one, losing the least recent pin state contained in bit 0. An interrupt request is then generated.

Generate match at ERT + MATCH\_RATE  
Bit N replaced by bit N+1 of parameter PLV  
Bit 15 of parameter PLV gets pin state  
Negate MRL, TDL, LSR  
Assert interrupt request

#### 10.1.2 State 3: Trans\_PS\_Low

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 001x00

Match Enable: Don't Care

Summary:

This state is entered in transition mode only, after a transition is detected, when the pin is low at the time of service. Bit 15 of PIN\_LEVEL is updated with the pin level, and the previous pin states are shifted right by one, losing the least recent pin state contained in bit 0. An interrupt request is then generated.

Bit N replaced by bit N+1 of parameter PLV  
Bit 15 of parameter PLV gets pin state 0  
Negate MRL, TDL, LSR  
Assert interrupt request

#### 10.1.3 State 4: Trans\_PS\_High

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 001x10

Match Enable: Don't Care

Summary:

This state is entered in transition mode only, after a transition is detected, when the pin is high at the time of service. Bit 15 of PIN\_LEVEL is updated with the pin level, and the previous pin states are shifted right by one, losing the least recent pin state contained in bit 0. An interrupt request is then generated.

Bit N replaced by bit N+1 of parameter PLV  
Bit 15 of parameter PLV gets pin state 1  
Negate MRL, TDL, LSR  
Assert interrupt request

#### 10.1.4 State 5: Low\_Pin\_Request

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

Match Enable: Don't Care

Summary:

This state is entered due to HSR %10. In this state the pin becomes an output and is forced low, and the match and capture time bases are forced to TCR1. Bit 15 of PIN\_LEVEL is updated with the pin level, which is forced, and the previous pin states are shifted right by one, losing the least recent pin state contained in bit 0. An interrupt request is then generated.

Set pin low  
Bit N replaced by bit N+1 of parameter PLV  
Bit 15 of parameter PLV gets pin state 0  
Negate MRL, TDL, LSR  
Assert interrupt request

### 10.1.5 State 6: High\_Pin\_Request

Condition: HSR1, HSR0, M/TSR, LSR, Pin, Flag0 = 01xxxx

Match Enable: Don't Care

Summary:

This state is entered as a result of HSR %01. In this state the pin becomes an output and is forced high, and the match and capture time bases are forced to TCR1. Bit 15 of PIN\_LEVEL is updated with the pin level, which is forced, and the previous pin states are shifted right by one, losing the least recent pin state contained in bit 0. An interrupt request is then generated.

- Set pin high
- Bit N replaced by bit N+1 of parameter PLV
- Bit 15 of parameter PLV gets pin state 0
- Negate MRL, TDL, LSR
- Assert interrupt request

The following table shows the DIO transitions listing the service request sources and channel conditions from current state to next state. **Figure 4** illustrates the flow of DIO states.

**Table 7 DIO State Transition Table**

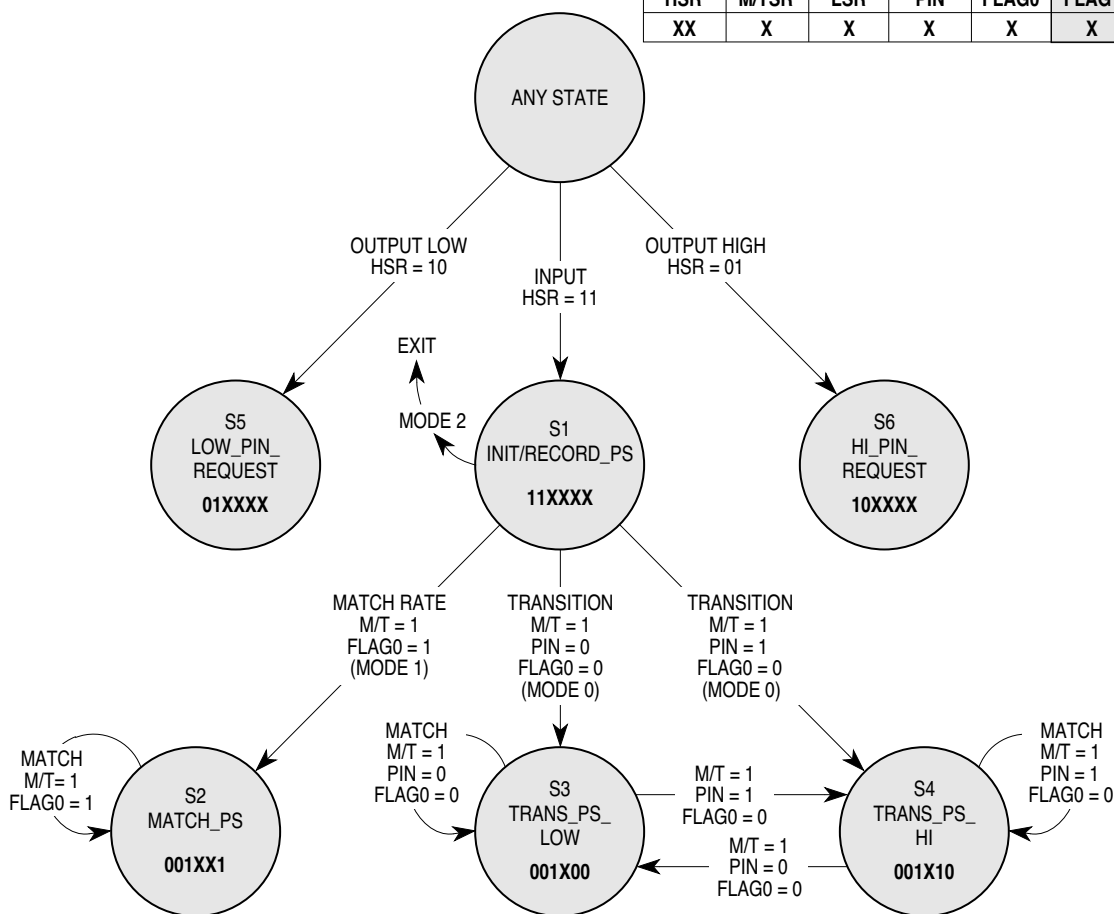
Current State	HSR	M/TSR	LSR	Pin	Flag0	Flag1	Next State
Any State	11	—	—	—	—	—	S1 Init/Record_PS
Any State	01	—	—	—	—	—	S6 High_Pin_Request
Any State	10	—	—	—	—	—	S5 Low_Pin_Request
S1 Init/Record_PS	00	1	—	—	1	—	S2 Match_PS
	00	1	—	0	0	—	S3 Trans_PS_Low
	00	1	—	1	0	—	S4 Trans_PS_High
S2 Match_PS	00	1	—	—	1	—	S2 Match_PS
S3 Trans_PS_Low	00	1	—	0	0	—	S3 Trans_PS_Low
	00	1	—	1	0	—	S4 Trans_PS_High
S4 Trans_PS_High	00	1	—	0	0	—	S3 Trans_PS_Low
	00	1	—	1	0	—	S4 Trans_PS_High
Unimplemented Conditions	00	0	1	x	—	—	—

NOTES:

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

KEY:

HSR	M/TSR	LSR	PIN	FLAG0	FLAG1
XX	X	X	X	X	X



1018A

Figure 4 DIO State Flowchart

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. MOTOROLA and ! are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

**How to reach us:**

**USA/EUROPE/Locations Not Listed:** Motorola Literature Distribution;

P.O. Box 5405, Denver Colorado 80217. 1-800-441-2447, (303) 675-2140

**Mfax™:** RMFAX0@email.sps.mot.com - TOUCHTONE (602) 244-6609, U.S. and Canada Only 1-800-774-1848

**INTERNET:** <http://Design-NET.com>

**JAPAN:** Nippon Motorola Ltd.; Tatsumi-SPD-JLDC,

6F Seibu-Butsuryu-Center, 3-14-2 Tatsumi Koto-Ku, Tokyo 135, Japan. 81-3-3521-8315

**ASIA PACIFIC:** Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,

51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852-26629298

Mfax is a trademark of Motorola, Inc.



**MOTOROLA**