Application Note

AN2370/D
Rev. 0, 10/2002

Using the Quadrature Decode TPU Function (QDEC) with the MPC500 Family

Jeff Loeliger
TECD

Freescale Semiconductor, Inc.

This TPU Programming note is intended to provide simple C interface routines to the quadrature decode TPU function (QDEC). [1] The routines are targeted for the MPC500 family of devices, but they should be easy to use with any device that has a TPU.

1 Functional Overview

The quadrature decode function is a TPU input function that uses two channels to decode a pair of out of phase signals in order to increment or decrement a (position) counter. It is particularly useful for decoding position and direction information from a slotted encoder in motion control systems, thus replacing expensive external solutions (see Figure 1).

[1]The information in this Programming Note is based on TPUPN20. It is intended to compliment the information found in that Programming Note.

© Freescale Semiconductor, Inc., 2004. All rights reserved.

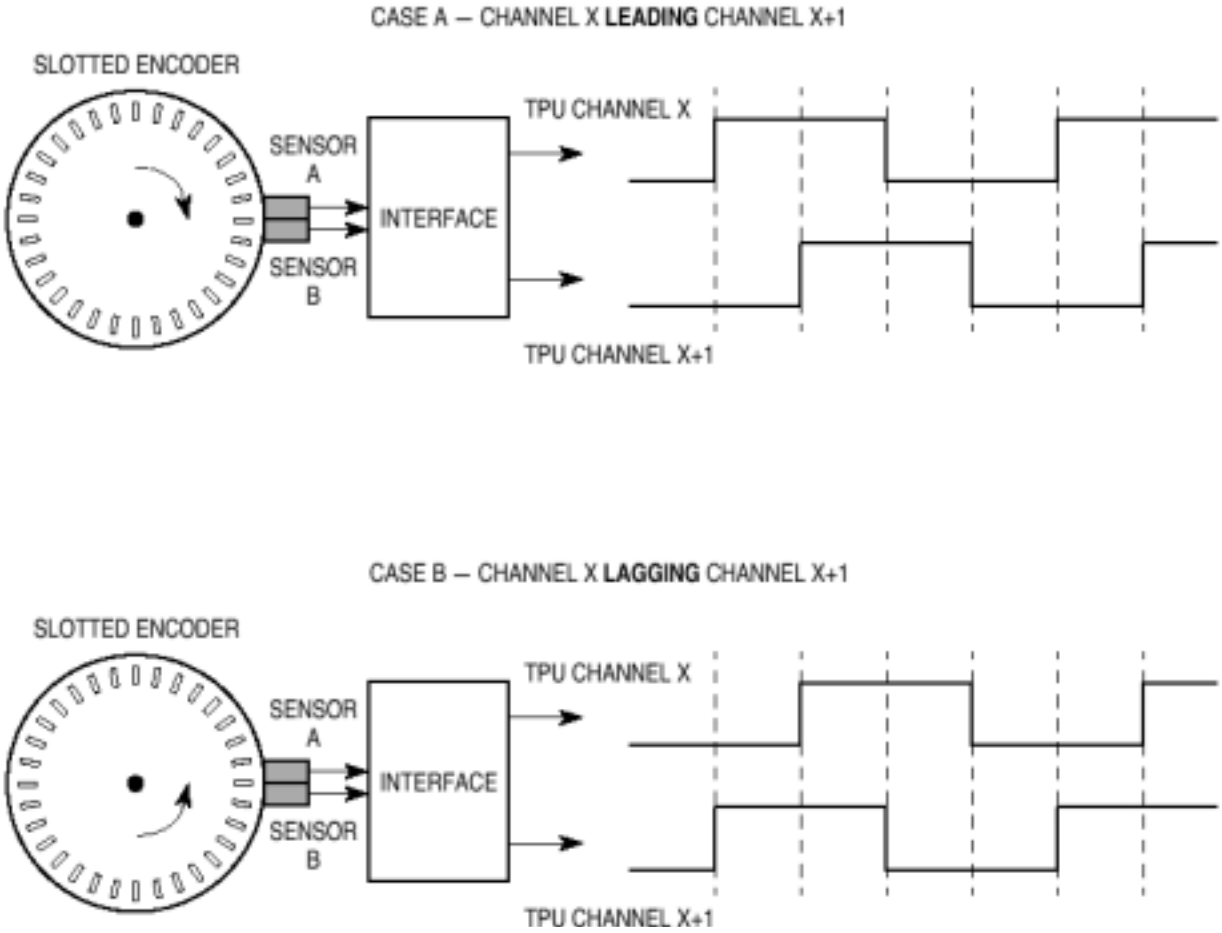

Figure 1. Lead/Lag Timing

2 Detailed Description

The QDEC function uses a pair of adjacent TPU channels to decode quadrature signals into a 16 bit counter in parameter RAM that is updated when a valid transition is detected on either one of the two inputs i.e. full “4x” every edge resolution is derived from the encoder signals. The counter is incremented or decremented depending on the lead/lag relationship of the two signals at the time of servicing the transition. The user can read the counter at any time using the *tpu_qdec_position* function. The counter is free running, overflowing to 0x0000 or under flowing to 0xFFFF depending on direction.

In systems where the counter may over or under flow, the user should ensure that the CPU reads the counter periodically, with the maximum period between reads being equivalent to 0x8000 counts at maximum signal frequency. Twos complement arithmetic can then be used by the CPU to maintain position and direction information.

When initialized the QDEC function is automatically configured so that the first edge on either channel will result in a counter update.

Since the two QDEC channels (which must always be adjacent) operate differently, the one with the lower channel number shall be referred to as the 'primary' channel and the other the 'secondary' channel.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

2.1 Time Stamp

The QDEC function also provides a time stamp referenced to TCR1 for every valid signal edge and the ability for the host CPU to obtain a current TCR1 value. These features allow position and speed interpolation by the host CPU between quadrature edges at very slow count rates.

2.2 Discrete Input / Transition Counter

A single channel programmed to run QDEC could be used as a digital input pin with a transition counter.

2.3 QDEC Routines

Rather than controlling the TPU registers directly, the QDEC routines in this TPU Programming Note may be used to provide a simple and easy interface. There are 4 routines for controlling the QDEC function in 2 files (tpu_qdec.h and tpu_qdec.c). The tpu_qdec.h file should be included in any files that use the routines. This files contains the function prototypes and useful #defines. Each of the following tpu_qdec.c routines is described in detail in the following sections:

- Initialization functions
  - void tpu_qdec_init(struct TPU3_tag *tpu, UINT8 channel, UINT8 priority, INT16 init_position);
  - void tpu_qdec_init_trans_count(struct TPU3_tag *tpu, UINT8 channel, UINT8 priority);
- Normal functions
  - INT16 tpu_qdec_position(struct TPU3_tag *tpu, UINT8 channel);
- Misc. functions
  - void tpu_qdec_data(struct TPU3_tag *tpu, UINT8 channel, INT16 *tcr1, INT16 *edge, INT16 *primary_pin, INT16 *secondary_pin);

2.3.1 void tpu_qdec_init

This function is used to initialize a pair of channels to run the QDEC function. This function has 4 parameters:

- *tpu - This is a pointer to the TPU3 module to use. It is of type TPU3_tag which is defined in m_tpu3.h
- channel - This is the channel number of the primary QDEC channel. The next channel is used as the secondary.
- priority - This is the priority to assign to both channels. This parameter should be assigned a value of: TPU_PRIORITY_HIGH, TPU_PRIORITY_MIDDLE or TPU_PRIORITY_LOW. The TPU priorities are defined in mpc500_utils.h.
- init_position - This is the starting position count.

Care should be taken when initializing TPU channels. The TPU’s behavior may be unpredictable if a channel is reconfigured while it is running. The channels should be stopped before then are configured. This is done by setting the channel’s priority to disabled. If the channel is currently being serviced when the priority is set to disable it will continue to service the channel until the state ends. To make sure the channel is not being service you need to wait for the longest state execution time after disabling the channel. All channels are disabled out of reset so the channels can be configured immediately from reset.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

The *tpu_qdec_init* function attempts to wait between the disabling of the channels before it starts configuring them, however the actual execution speed of the code will be depend on the specific system. If you are not configuring the channels from reset, then ideally it is best to have the functions disabled before calling this function. TPU channels can be disabled by using the *tpu_disable* function in the mpc500_utils.c file. For example, disabling channels 5 & 6 is done like this:

```
tpu_disable(tpu, 5);
tpu_disable(tpu,6);
```

2.3.2 void tpu_qdec_init_trans_count

This function is used to initialize a single channel to run in discrete input/transition count feature of the QDEC function. When configured in this mode, the pin state will be updated when each transition is serviced to contain a value representing the latest pin level and the position count will count the number of transitions on the pin (positive and negative). The *tpu_qdec_data* function can be used to get the pin state. This function has 3 parameters:

- *tpu - This is a pointer to the TPU3 module to use. It is of type TPU3_tag which is defined in m_tpu3.h
- channel - This is the channel number of the primary QDEC channel.
- priority - This is the priority to assign to both channels. This parameter should be assigned a value of: TPU_PRIORITY_HIGH, TPU_PRIORITY_MIDDLE or TPU_PRIORITY_LOW. The TPU priorities are defined in mpc500_utils.h.

As described in *tpu_qdec_init* it is best if the channel is disable and not running before this initialization routine is called. The *tpu_qdec_data* function will return the current state of both of pins, however because only one channel is used in this mode, the state of the second channel will not return useful data.

2.3.3 INT16 tpu_qdec_position

This function returns the current position count of the QDEC channels. This is the 16 bit counter that is the primary output of the QDEC function. This function has 2 parameters:

- *tpu - This is a pointer to the TPU3 module to use. It is of type TPU3_tag which is defined in m_tpu3.h
- channel - This is the channel number of the primary QDEC channel.

The return value is the current position count value.

2.3.4 void tpu_qdec_data

This function returns the data parameters associated with the QDEC channels. This is function can be used to get the current state of the pins or the data could be used to interpolate the position on very slow input signals. This function has 6 parameters:

- *tpu - This is a pointer to the TPU3 module to use. It is of type TPU3_tag which is defined in m_tpu3.h
- channel - This is the channel number of the primary QDEC channel.
- *tcr1 - The current value of the TCR1 timebase.
- *edge - The last edge time.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

- *primary_pin - The current state of the primary channel, this will be TPU_QDEC_PIN_HIGH or TPU_QDEC_PIN_LOW.
- *secondary_pin - The current state of the primary channel, this will be TPU_QDEC_PIN_HIGH or TPU_QDEC_PIN_LOW.

WARNING

In order for this function to return the current TCR1 value, it must request a TPU host service and wait for the service to be completed. If the TPU channel was accidentally disabled or the TPU is stuck in an endless loop then this function WILL NEVER RETURN.

The value 0x8000 is used to represent a pin high level, and 0x0000 to represent a pin low level.

3 Quadrature Decode Examples

The following examples show configuration of the quadrature decode function for both quadrature decode and as an input pin with transition counter. Each example is a C program that shows how to configure and use the QDEC interface routines.

3.1 Example 1

3.1.1 Description

This is a simple program to get a basic quadrature input signal system running. It configure channels 0 and 1 on TPU A to run QDEC. The initial position should be 0x0000. The channel should be scheduled as high priority.

The program then runs in an infinite loop reading back the current position based on the quadrature signal on TPU A channels 0 and 1.

3.1.2 Program

```
/*************************************************************************/
/* FILE NAME: tpu_qdec_example1.c             COPYRIGHT (c)  2002 */
/* VERSION: 1.0                                   All Rights Reserved     */
/*                                                                        */
/* DESCRIPTION: This sample program show a simple example of a program    */
/* that updates of the global "position" variable from an external         */
/* quadrature input signal. The signal is connected to TPU A on channels  */
/* 0 and 1.                                                               */
/* The program is targeted for the MPC555 but should work on any MPC500   */
/* device with a TPU. For other devices the setup routines will also need */
/* to be changed.                                                         */
/*========================================================================*/
/* HISTORY           ORIGINAL AUTHOR: Jeff Loeliger                       */
/* REV      AUTHOR      DATE       DESCRIPTION OF CHANGE                  */
```


Freescale Semiconductor, Inc.

Example 2

```
/* ---   -----------  ---------    --------------------                 */
/* 1.0   J. Loeliger  11/Aug/02    Initial version of function.         */
/************************************************************************/

#include "mpc555.h"     /* Define all of the MPC555 registers, this needs to */
                        /* changed if other MPC500 devices are used.         */
#include "mpc500.c"     /* Configuration routines for MPC555 EVB, will need */
                        /* to be changed if other hardware is used.         */
#include "mpc500_util.h"    /* Utility routines for using MPC500 devices */
#include "tpu_qdec.h"    /* TPU QDEC functions */

#define ENCODER1 tpua,0  /* ENCODER1 is attached to TPU A channel 0 */

INT16 position;     /* global position value for quadrature input signal */

void main ()
{
     struct TPU3_tag *tpua = &TPU_A;   /* pointer for TPU routines */

     setup_mpc500(40);       /*Setup device and programm PLL to 40MHz*/

    /* Initialize quadrature input function with:   */
    /*    -ENCODER1                                 */
    /*    -Initial count value of 0x0000              */
    /*    -Schedule as high priority in the TPU     */
    tpu_qdec_init( ENCODER1 , TPU_PRIORITY_HIGH, 0x0000);

    while (1) {
         position = tpu_qdec_position ( ENCODER1 );
    }
}
```

3.2 Example 2

3.2.1 Description

This program shows how to use the QDEC function in the discrete input/transition count mode. A single channel is initialized and the number of input transition is monitored. The program also shows how to read back the other information from the QDEC function.

Channel 12 is configured for discrete input/transition count mode and scheduled as low priority.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

3.2.2 Program

```
/*************************************************************************/
/* FILE NAME: tpu_qdec_exmaple2.c             COPYRIGHT (c)  2002 */
/* VERSION: 1.0                                   All Rights Reserved     */
/*                                                                        */
/* DESCRIPTION: This sample program show a simple example of a program    */
/* that uses the discrete input/transition counter feature of the QDEC    */
/* function. The signal is connected to channel 12 on TPU B. The number of*/
/* transitions and other data is continually read from the channel.      */
/* The program is targeted for the MPC555 but should work on any MPC500   */
/* device with a TPU. For other devices the setup routines will also need */
/* to be changed.                                                         */
/*========================================================================*/
/* HISTORY           ORIGINAL AUTHOR: Jeff Loeliger                       */
/* REV      AUTHOR      DATE       DESCRIPTION OF CHANGE                  */
/* ---   -----------  ---------    --------------------                   */
/* 1.0   J. Loeliger  11/Aug/02    Initial version of function.           */
/*************************************************************************/

#include "mpc555.h"     /* Define all of the MPC555 registers, this needs to */
                        /* changed if other MPC500 devices are used.         */
#include "mpc500.c"     /* Configuration routines for MPC555 EVB, will need */
                        /* to be changed if other hardware is used.         */
#include "mpc500_util.h"    /* Utility routines for using MPC500 devices */
#include "tpu_qdec.h"   /* TPU QDEC functions */

INT16 count;     /* global transition counter */

void main ()
{
     struct TPU3_tag *tpub = &TPU_B;   /* pointer for TPU routines */
     INT16 tcr1;          /* current TCR1 value */
     INT16 edge;          /* last edge time */
     INT16 primary_pin;   /* current pin state */
     INT16 junk;          /* this variable is not used and will return junk */

     setup_mpc500(40);       /*Setup device and program PLL to 40MHz*/

    /* Initialize transition count feature with:   */
    /*    -Input signal on TPU B channel 12         */
```


Freescale Semiconductor, Inc.

```
    /*    -Schedule as low priority in the TPU      */
    tpu_qdec_init_trans_count( tpub, 12 , TPU_PRIORITY_LOW);

    while (1) {
         count = tpu_qdec_position ( tpub, 12 );

         tpu_qdec_data( tpub, 12 , &tcr1, &edge, &primary_pin, &junk);
    }
}
```

3.3 Function State Timing

When calculating the worst case latency for the TPU the execution time of each state of the TPU is need.

The state timings for the QDEC function are shown in Table 1. The states used by the C interface functions are shown in Table 2. State S3 is entered when there is a transition on the input pins.

Table 1. Quadrature Decode Function—State Timing

State Number & Name	Max CPU Clock Cycles	RAM accesses by TPU
S1 INIT_QDEC	12	3
S2 READ_TCR1_QDEC	2	1
S3 EDGE_QDEC	28	8

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

Table 2. QDEC Function State Usage

QDEC Function	State Uses
tpu_qdec_init	S1
tpu_qdec_init_trans_count	S1
tpu_qdec_position	none
tpu_qdec_data	S2

3.4 Function Code Size

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

21 µ instructions + 8 entries = 29 long words

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

4 Notes on Performance and Use of the QDEC Function

4.1 Performance

Like all TPU functions, the performance limit of the QECD function in a given application is dependent on the service time (latency) of other active TPU channels. This is due to the operational nature of the scheduler. Where a pair of QDEC channels is being used and no other TPU channels are active, the minimum time between count edges on the two channels is 42 CPU clock cycles. This is equivalent to a count rate of approximately 960,000 counts per second with a system clock speed of 40MHz.

When more TPU channels are active, this performance will be degraded; e.g. if two sets of encoder signals are decoded using four channels then the maximum count rate in each mode would be limited to approximately 480,000 counts per second. Use of other functions such as PWM will also have a degrading effect on this performance.

However the scheduler assures that the worst case latencies in any TPU application can be closely estimated, so it is recommended that the guidelines given in the TPU reference manual are used along with the figures given in the Quadrature Decode state timing table to perform an analysis on any proposed TPU application that appears to approach the performance limits of the TPU.

4.2 Accuracy

Since the TPU is essentially a software machine that takes time to respond to an input transition, there will always be a +/- 1 lsb uncertainty in the value of the position count in normal mode while the input signals are active.

These 'uncertainties' only apply while the external system (e.g. a motor) is active, after the system has been brought to a stop and the last transition has been serviced, the position count will be accurate.

4.3 Noise Immunity

Features in the hardware of the TPU and the microcode of the QDEC function protect, to a large extent, the counter from erroneous updates due to noise. All TPU input channels incorporate a digital filter, which rejects pulses of less than a programmable duration.

In addition to this protection, when servicing a transition, the QDEC function always checks the new pinstate against the previous pinstate from the last service and if they are equal then no action is taken. This protects against a noise pulse that is long enough to get through the digital filter, but not long enough to last from the actual transition time to the time that the TPU services the channel.

Despite these precautions, there may be situations where severe noise will result in erroneous updates of the counter e.g. noise on both channels simultaneously. Under these conditions it is recommended that additional external protection is added, such as schmitt trigger buffers and an additional analog or digital filter stage.

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

4.4 Use of QDEC with 3-Channel Encoders

Many shaft encoders supply three signals: two quadrature signals plus an index signal that generates a pulse once per revolution. This pulse usually has a fixed relationship to other system parameters and is used for alignment during startup.

These 3 signal encoders can be decoded when QDEC is used in conjunction with another TPU function called “New Input Transition Counter (NITC).” QDEC decodes the quadrature signals and the index pulse should be fed to the NITC channel. NITC allows any location in parameter RAM to be captured on a specified edge and the value presented to the CPU. In this case NITC would be configured to 'capture' the POSITION_COUNT parameter of QDEC.

4.5 Listing 1

```
/*****************************************************************************/
/* FILE NAME: tpu_qdec.h                      COPYRIGHT (c)  2002 */
/* VERSION: 1.0                                   All Rights Reserved     */
/*                                                                           */
/* DESCRIPTION: This file defines the interface to the TPU QDEC functions */
/* and provides useful #defines.                                             */
/*                                                                           */
/*===========================================================================*/
/* HISTORY           ORIGINAL AUTHOR: Jeff Loeliger                          */
/* REV      AUTHOR      DATE       DESCRIPTION OF CHANGE                   */
/* ---   -----------  ---------    --------------------                    */
/* 1.0   J. Loeliger  11/Aug/02    Initial version of function.            */
/*****************************************************************************/
#ifndef _TPU_QDEC_H
#define _TPU_QDEC_H

#include "m_common.h"
#include "m_tpu3.h"

/* Define function number                            */
/* This might change with a custom set of functions */
#define TPU_FUNCTION_QDEC 0x6

/* Define HSR values */
#define TPU_QDEC_INIT 0x3
#define TPU_QDEC_READ_TCR1 0x2

/* Define HSQ values */
#define TPU_QDEC_PRIMARY_CHANNEL 0x0
```


For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

```
#define TPU_QDEC_SECONDARY_CHANNEL 0x1

/* Define pin state */
#define TPU_QDEC_PIN_HIGH 0x8000
#define TPU_QDEC_PIN_LOW  0x0000

/* Define parameter RAM locations */
#define TPU_QDEC_EDGE_TIME          0
#define TPU_QDEC_POSITION_COUNT     1
#define TPU_QDEC_TCR1_VALUE         2
#define TPU_QDEC_CHAN_PINSTATE      3
#define TPU_QDEC_CORR_PINSTATE_ADDR 4
#define TPU_QDEC_EDGE_TIME_LSB_ADDR 5

/* TPU QDEC function prototypes */
void tpu_qdec_init(struct TPU3_tag *tpu, UINT8 channel, UINT8 priority, \
                INT16 init_position);
void tpu_qdec_init_trans_count(struct TPU3_tag *tpu, UINT8 channel, \
                UINT8 priority);

INT16 tpu_qdec_position(struct TPU3_tag *tpu, UINT8 channel);

void tpu_qdec_data(struct TPU3_tag *tpu, UINT8 channel, INT16 *tcr1, \
                INT16 *edge, INT16 *primary_pin, INT16 *secondary_pin);

#endif /* ifndef _TPU_QDEC_H  */

/*********************************************************************
```

4.6 Listing 2

```
/*********************************************************************/
/* FILE NAME: tpu_qdec.c                     COPYRIGHT (c)  2002 */
/* VERSION: 1.0                                  All Rights Reserved     */
/*                                                                       */
/* DESCRIPTION: This file contains the TPU QDEC functions. These functions*/
/* allow you to completely control TPU channels running the QDEC function.*/
/* They provide a simple interface requiring the minimum amount of        */
/* configuration by the user.                                            */
/*=======================================================================*/
```


For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

Listing 2

```
/* HISTORY           ORIGINAL AUTHOR: Jeff Loeliger                       */
/* REV      AUTHOR      DATE       DESCRIPTION OF CHANGE                 */
/* ---   -----------  ---------    --------------------                  */
/* 1.0   J. Loeliger  11/Aug/02    Initial version of function.          */
/*************************************************************************/
#include "tpu_qdec.h"
#include "mpc500_util.h"

/*************************************************************************
FUNCTION      : tpu_qdec_init
PURPOSE       : To initialize a pair of channels to run the QDEC function.
INPUTS NOTES  : This function has 4 parameters:
                 *tpu - This is a pointer to the TPU3 module to use. It is of
                        type TPU3_tag which is defined in m_tpu3.h
                 channel - This is the channel number of the primary QDEC
                           channel. The next channel is used as the secondary.
                 priority - This is the priority to assign to both channels.
                            This parameter should be assigned a value of:
                            TPU_PRIORITY_HIGH, TPU_PRIORITY_MIDDLE or
                            TPU_PRIORITY_LOW.
                 init_position - This is the starting position.
RETURNS NOTES : none
WARNING       : The channels must be stopped before it is reconfigured. The
                function disables the channels but if they were currently
                being serviced it would continue. The delay for assigning the
                pram pointer may to enough but depends on system loading.
*************************************************************************/
void tpu_qdec_init(struct TPU3_tag *tpu, UINT8 channel, UINT8 priority, \
                INT16 init_position)
{
    struct TPU_param_tag *pram;
    UINT8 channel2;

    /* if primary channel is 15 then secondary channel should be 0 */
    channel2 = (channel + 1) & 0xF;

    /* diable channels so they can be configured safely */
    tpu_disable( tpu, channel);
    tpu_disable( tpu, channel2);

    /* select QDEC function for both channels */
```


For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

Listing 2

```
    tpu_func( tpu, channel, TPU_FUNCTION_QDEC);
    tpu_func( tpu, channel2, TPU_FUNCTION_QDEC);

    /* Initialize parameter RAM                                          */
    /* -setup initial count in POSITION_COUNT                            */
    /* -initialize CORR_PINSTATE_ADDR and EDGE_TIME_LSB_ADDR in both channels*/
    tpu->PARM.R[channel][TPU_QDEC_POSITION_COUNT] = init_position;
    tpu->PARM.R[channel][TPU_QDEC_CORR_PINSTATE_ADDR] = (INT16) (((channel2) << 4) + 6);
    tpu->PARM.R[channel][TPU_QDEC_EDGE_TIME_LSB_ADDR] = (INT16) ((channel << 4) + 1);
    tpu->PARM.R[channel2][TPU_QDEC_CORR_PINSTATE_ADDR] = (INT16) ((channel << 4) + 6);
    tpu->PARM.R[channel2][TPU_QDEC_EDGE_TIME_LSB_ADDR] = (INT16) ((channel << 4) + 1);

    /* Configure the first channel as the primary channel and the following */
    /* channel as the secondary channel.                                 */
    tpu_hsq(tpu, channel, TPU_QDEC_PRIMARY_CHANNEL);
    tpu_hsq(tpu, channel2, TPU_QDEC_SECONDARY_CHANNEL);

    /* Initialize both channels */
    tpu_hsr(tpu, channel, TPU_QDEC_INIT);
    tpu_hsr(tpu, channel2, TPU_QDEC_INIT);

    /* Enable channels by assigning a priority to them. */
    /* Both channels MUST have the same priority.       */
    tpu_enable(tpu, channel, priority);
    tpu_enable(tpu, channel2, priority);
}

/*************************************************************************
FUNCTION      : tpu_qdec_init_trans_count
PURPOSE       : To initialize one channel as a discrete input/transition
                counter.
INPUTS NOTES  : This function has 3 parameters:
                  *tpu - This is a pointer to the TPU3 module to use. It is of
                         type TPU3_tag which is defined in m_tpu3.h
                  channel - This is the channel number of the primary QDEC
                            channel. The next channel is used as the secondary.
                  priority - This is the priority to assign to both channels
                             This parameter should be assigned a value of:
                             TPU_PRIORITY_HIGH, TPU_PRIORITY_MIDDLE or
                             TPU_PRIORITY_LOW.
RETURNS NOTES : none
```


For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

Listing 2

```
WARNING       : The channel must be stopped before it is reconfigured. The
                function disables the channel but if it is currently
                being serviced it would continue. The delay for assigning the
                pram pointer may to enough but depends on system loading.
*************************************************************************/
void tpu_qdec_init_trans_count(struct TPU3_tag *tpu, UINT8 channel, \
                UINT8 priority)
{
    struct TPU_param_tag *pram;

    /* diable channel so it can be configured safely */
    tpu_disable( tpu, channel);

    /* select QDEC function for both channels */
    tpu_func( tpu, channel, TPU_FUNCTION_QDEC);

    /* Initialize parameter RAM                              */
    /* -clear POSITION_COUNT to count transitions            */
    /* -initialize CORR_PINSTATE_ADDR and EDGE_TIME_LSB_ADDR */
    tpu->PARM.R[channel][TPU_QDEC_POSITION_COUNT] = 0;
    tpu->PARM.R[channel][TPU_QDEC_CORR_PINSTATE_ADDR] = (INT16) ((channel << 4) + 6);
    tpu->PARM.R[channel][TPU_QDEC_EDGE_TIME_LSB_ADDR] = (INT16) ((channel << 4) + 1);

    /* Configure the channel as a primary channel */
    tpu_hsq(tpu, channel, TPU_QDEC_PRIMARY_CHANNEL);

    /* Initialize the channel */
    tpu_hsr(tpu, channel, TPU_QDEC_INIT);

    /* Enable the channel by assigning a priority to it. */
    tpu_enable(tpu, channel, priority);
}

/*************************************************************************
FUNCTION      : tpu_qdec_position
PURPOSE       : This function returns the current postion count for the
                quqdrature input signal.
INPUTS NOTES  : This function has 2 parameters:
                 *tpu - This is a pointer to the TPU3 module to use. It is of
                        type TPU3_tag which is defined in m_tpu3.h
                 channel - This is the channel number of the primary QDEC
```


For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

Listing 2

```
                        channel. The next channel is used as the secondary.
RETURNS NOTES : The current position count.
*****************************************************************************/
INT16 tpu_qdec_position(struct TPU3_tag *tpu, UINT8 channel)
{
     return (tpu->PARM.R[channel][TPU_QDEC_POSITION_COUNT]);
}

/*****************************************************************************
FUNCTION      : tpu_qdec_data
PURPOSE       : To get the current TCR1 value and the time of the last edge.
INPUTS NOTES  : This function has 6 parameters:
                 *tpu - This is a pointer to the TPU3 module to use. It is of
                        type TPU3_tag which is defined in m_tpu3.h
                 channel - This is the channel number of the primary QDEC
                           channel. The next channel is used as the secondary.
                 *tcr1 - The current Value of the TCR1 timebase.
                 *edge - The last edge time.
                 *primary_pin - The current state of the primary channel, this
                                will be TPU_QDEC_PIN_HIGH or TPU_QDEC_PIN_LOW.
                 *secondary_pin - The current state of the primary channel, this
                                  will be TPU_QDEC_PIN_HIGH or TPU_QDEC_PIN_LOW.
RETURNS NOTES : none
**WARNING**   : If the channel is disabled or the TPU is not responding this
                function will NEVER EXIT!
*****************************************************************************/
void tpu_qdec_data(struct TPU3_tag *tpu, UINT8 channel, INT16 *tcr1, \
                INT16 *edge, INT16 *primary_pin, INT16 *secondary_pin)
{
    UINT8 channel2;

    /* if primary channel is 15 then secondary channel should be 0 */
    channel2 = (channel + 1) & 0xF;

     /* wait until the TPU channel has no pending HSRs */
     /* WARNING if the TPU does on service this request then this loop will */
     /* NEVER EXIT!                                                         */
     tpu_ready(tpu, channel);

     /* issue request to update TCR1 value */
     tpu_hsr(tpu, channel, TPU_QDEC_READ_TCR1);
```


For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

Listing 2

```
    /* read parameters */
    *edge = tpu->PARM.R[channel][TPU_QDEC_EDGE_TIME];
    *primary_pin=tpu->PARM.R[channel][TPU_QDEC_CHAN_PINSTATE];
    *secondary_pin=tpu->PARM.R[channel2][TPU_QDEC_CHAN_PINSTATE];

    /* wait until TCR1 value has been updated */
    /* WARNING if the TPU does on service this request then this loop will */
    /* NEVER EXIT!                                                         */
    tpu_ready(tpu, channel);

    *tcr1 = tpu->PARM.R[channel][TPU_QDEC_TCR1_VALUE];
}

/*********************************************************************
```

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

THIS PAGE INTENTIONALLY LEFT BLANK

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

THIS PAGE INTENTIONALLY LEFT BLANK

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

THIS PAGE INTENTIONALLY LEFT BLANK

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

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

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.
Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor 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 Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor 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 Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor 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 Freescale Semiconductor was negligent regarding the design or manufacture of the part.

freescale™
semiconductor

**For More Information On This Product,
Go to: www.freescale.com**