Freescale Semiconductor
Application Note

AN2850
Rev. 0, 06/2005

Using the General Purpose Input/Output (GPIO) eTPU Function

by: Ken Terry
MCD

The enhanced time processor unite (eTPU) general purpose input/output (GPIO) application note describes a set of simple C interface routines to the GPIO eTPU function. The routines can be used on any device that has an eTPU. The example code provided in this document was written for the MPC5554 device. This application note should be read in conjunction with application note “AN2864 - General C Functions for the eTPU”.

1 Function Overview

The GPIO functions allow the user to configure an eTPU channel as an input or output.

As an input, the selected eTPU pin can be read periodically, as a result of a CPU command, or whenever a transition occurs on a pin. When an eTPU

channel is configured to read the pin periodically, the period is determined by either of the eTPU timer counter registers (TCR1 or TCR2).

As an output, the pin can be driven either to a logic high or a logic low level, as a result of a CPU command.

When a channel is configured as an input, the eTPU records the last 24 levels, or edge transitions in a parameter held in its assigned parameter RAM.

2 Detailed Description

The functions support four modes of operation:

- Output mode
- Input on transition mode
- Input periodic mode
- Input immediate mode

In output mode, the selected eTPU channel can be set to drive a logic high or logic low level as a result of a CPU command. The output state is controlled by three API functions, allowing the eTPU pin to be driven high, low, or to a logic level determined by a function input parameter.

In input on transition mode, the selected eTPU channel records transitions on its corresponding input pin. The channel can be configured to record rising edge, falling edge, or rising and falling edge transitions. Whenever an input transition occurs, the eTPU will generate both a DMA and an interrupt request.

In input periodic mode, the channel samples and records the pin state at regular intervals. The interval period is determined by a variable, which defines a number of timer counter register (TCR) counts. Either time base, TCR1 or TCR2, can be used. The interval period is dependent therefore on the selected time base, the time base prescaler value, the number of TCR counts and the system frequency. Whenever a channel pin is sampled in this mode, the eTPU generates both a DMA and an interrupt request.

In input immediate mode, the channel pin is sampled immediately on request. On completion of the sampling process, the eTPU generates both a DMA and an interrupt request.

For all input modes, a record of the sampled pin value is stored in the low order 24 bits of a 32 bit variable - PINSTATE. Each time the input pin is sampled, the PINSTATE variable is shifted left and the latest input pin value is stored in the least significant bit (lsb) position of the variable. For output modes, the variable will contain a record of the transmitted values.

3 Performance

The GPIO eTPU function is very simple and straightforward. Because of the way in which the eTPU scheduler operates, the performance of the GPIO function is dependent on how many other eTPU channels are active and the assigned priority of the channels. The GPIO output functions can be executed in a single eTPU cycle. The maximum number of cycles required to process an input transition

will be dependent on the compiler, and the user should refer to the accompanying release notes for the actual cycle requirements for input periodic mode and input immediate modes.

4 C Level API for the eTPU GPIO Function

The following routines have been developed to allow easy implementation of the GPIO eTPU functions. They provide a simple C level API interface, allowing the user to initialize and control the eTPU channels running the GPIO functions, without the need to directly access any of the eTPU registers. The GPIO API consists of eight separate functions. The functions and the function definitions are found in files etpu_gpio.c and etpu_gpio.h, available from Freescale.

The following is a description of each of the functions.

4.1 Initialization

```
void fs_etpu_gpio_init(uint8_t channel, uint8_t priority)
```

This is the initialization function for an eTPU channel assigned the GPIO function. The function has the following parameters:

- Channel – this is the channel number (0 – 31 for eTPU_A and 64 – 95 for eTPU_B).
- Priority - this is the channel priority and should be assigned one of the following symbolic constants, which are defined in the utilities file etpu_utils.h:

FS_ETPU_PRIORITY_HIGH

FS_ETPU_PRIORITY_MIDDLE

FS_ETPU_PRIORITY_LOW

FS_ETPU_PRIORITY_DISABLED

4.2 Output Functions

```
void fs_etpu_gpio_output_high(uint8_t channel)
```

This function sets the pin associated with the channel to a logic high. It has the following single parameter:

- Channel – this is the channel number (0 – 31 for eTPU_A and 64 – 95 for eTPU_B).

```
void fs_etpu_gpio_output_low(uint8_t channel)
```

This function sets the pin associated with the channel to a logic low. It has the following single parameter:

- Channel – this is the channel number (0 – 31 for eTPU_A and 64 – 95 for eTPU_B).

```
void fs_etpu_gpio_output(uint8_t channel, uint8_t level)
```

This function sets the pin associated with the channel to a logic level defined by an input parameter. It has the following parameters:

- Channel – this is the channel number (0 – 31 for eTPU_A and 64 – 95 for eTPU_B).
- Level – this defines the logic level to which the associated output pin should be set. It should be assigned one of the following symbolic constants - defined in fs_etpu_gpio.h:

FS_ETPU_OP_LOW

FS_ETPU_OP_HIGH

4.3 Input Functions – Input Transition Mode

```
void fs_etpu_gpio_cfg_input_trans (uint8_t channel, uint8_t mode)
```

This function configures the eTPU GPIO channel for input transition mode. In this mode, the channel will detect input transitions on the associated channel pin. On each detected transition, the PINSTATE parameter in the eTPU parameter RAM is shifted left by one bit, and the logical state of the associated input pin is recorded in the lsb of PINSTATE. The function has two parameters:

- Channel – this is the channel number (0 – 31 for eTPU_A and 64 – 95 for eTPU_B).
- Mode – determines the type of transition detected by the channel. This parameter should be assigned one of the following symbolic constants – defined in etpu_gpio.h:

FS_ETPU_GPIO_INPUT_RISING

FS_ETPU_GPIO_INPUT_FALLING

FS_ETPU_GPIO_INPUT_EITHER

4.4 Input Functions – Input Periodic Mode

```
void fs_etpu_gpio_cfg_input_periodic (uint8_t channel, uint8_t
timebase,uint32_t rate)
```

This function configures the eTPU GPIO channel for input periodic mode The state of the associated input pin is sampled periodically in this mode. Each time the pin is sampled, the PINSTATE parameter in the eTPU parameter RAM is shifted left by one bit and the logical state of the associated input pin is recorded in the lsb of PINSTATE.

The function has three parameters:

- Channel – this is the channel number (0 – 31 for eTPU_A and 64 – 95 for eTPU_B).
- Timebase – determines whether TCR1 or TCR2 is used as timebase and should be assigned a value of ETPU_TCR1 or ETPU_TCR2.
- Rate – this is a 32-bit value, of which the lower 24 bits are written to the selected timer counter register to define the sampling period.

4.5 Input Functions – Input Immediate Mode

```
void fs_etpu_gpio_input_immed (uint8_t channel)
```

This function updates the PINSTATE parameter immediately, according to the input pin level. The PINSTATE parameter is shifted left by one bit, and the logical state of the associated input pin is recorded in the lsb of PINSTATE.

The function has a single parameter:

- Channel – this is the channel number (0 – 31 for eTPU_A and 64 – 95 for eTPU_B).

4.6 Return Pin State History Function

```
uint32_t fs_etpu_gpio_pin_history (uint8_t channel)
```

This function returns a 32-bit uint value containing the PINSTATE parameter held in the eTPU parameter RAM.

The function has a single parameter:

- Channel – this is the channel number (0 – 31 for eTPU_A and 64 – 95 for eTPU_B).

5 Examples of Function Use

Example code illustrating how to use the GPIO Function is available from Freescale. This code is designed to work with the MPC5554 EVB, but with some minor modifications, can work with any embedded microcontroller with an eTPU.

The software comprises a simple test routine that sets up two eTPU channels (4 & 8) and assigns them as GPIO. The software is contained in two files: gpio_example.c and gpio_example.h.

The main() routine is found in gpio_example.c. This routine initializes the MPC5554 device for 128 MHz CPU operation. The eTPU is initialized according to information contained in the my_etpu_config struct, which is defined in the file gpio_example.h.

For the test routine to work correctly, the device pins corresponding to eTPU channels 4 and 8 need to be connected together.

ETPU channel 4 (designated GPIO0) is configured as a GPIO output, and channel 8 (designated GPIO1) is configured as an input. As GPIO0 is toggled, the various input functions on GPIO1 are exercised.

6 Summary

This application note provides the user with a description of the API functions used to implement the GPIO eTPU micro-code function. The example code described in this document is developed for the MPC5554; however these simple C API functions provide a general purpose I/O capability that can be easily implemented on any MCU with an eTPU.

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
LDCForFreescaleSeminconductor@hibbertgroup.com

AN2850
Rev. 0
06/2005