Using the DC Motor Control PWM eTPU Functions

Covers the MCF523x, MPC5500, and all eTPU-equipped Devices

by: Milan Brejl
System Application Engineer, Roznov Czech System Center
Michal Princ
System Application Engineer, Roznov Czech System Center

1 Introduction

The generation of PWM signals for DC motor-control applications with eTPU is provided by three eTPU functions:

- PWM - Master for DC motors (PWMMDC)
- PWM - Full range (PWMF)
- PWM - Commutated (PWMC)

The PWMMDC, PWMF, and PWMC eTPU functions are located in the DC motor control eTPU function set (set3). This eTPU application note is intended to provide simple C interface routines to the PWMMDC, PWMF, and PWMC eTPU functions. The routines are targeted at the MCF523x and MPC5500 families of devices, but they could be easily used with any device that has an eTPU.
2 Function Overview

The PWMMDC, PWMF, and PWMC functions enable the eTPU to generate PWM signals for driving a motor. The PWM Master (PWMMDC) function synchronizes and updates up to three PWM phases. The phases may be driven either by the PWM Full Range (PWMF) function that enables a full 0% to 100% duty-cycle range, or by the PWM Commutated (PWMC) function that enables switching the phase ON and OFF.

The PWMF and PWMC functions generate the PWM signals. The PWMMDC function controls the PWMF and PWMC functions, does not generate any drive signal, and can be executed even on an eTPU channel not connected to an output pin. If connected to an output pin, the PWMMDC function turns the pin high and low, so that the high-time identifies the period of time in which the PWMMDC execution is active. In this way, the PWMMDC function, as with many of the motor-control eTPU functions, supports checking eTPU latencies using an oscilloscope.

![Figure 1. Functionality of PWMMDC+PWMC](image-url)
3 Function Description

This chapter describes in detail all PWM signal generation features, as well as using the PWMMDC function with the PWMF or PWMC function.

3.1 Modulations

The PWMMDC function transforms input parameters into phase duty-cycles by modulation. The following modulations are provided by the PWMMDC function:

- **Unsigned Voltage** (VOLTAGE_UNSIGNED) - The input parameter is an applied motor voltage in the range (0,1). The voltage range (0,1) corresponds to the duty-cycle range (0%, 100%). Zero voltage corresponds to a 0% duty-cycle. All phases are controlled by the same duty-cycle, positional negated (see 3.2.4).

- **Signed Voltage** (VOLTAGE_SIGNED) - The input parameter is an applied motor voltage in the range (-1,1). The voltage range (-1,1) corresponds to the duty-cycle range (0%, 100%). Zero voltage corresponds to a 50% duty-cycle. All phases are controlled by the same duty-cycle, optionally negated (see 3.2.4)

- **No Modulation** (NO) - No input parameters are transformed into phase duty-cycles. The phase duty-cycles must be provided.
3.2 Phase Options

3.2.1 Phase Type: Single Channel or Complementary Pair

The phase type option applies equally for all phases. The phases can be represented by an eTPU channel generating the PWM signal, or by a pair of eTPU channels generating a complementary pair of top and bottom PWM signals with dead-time insertion. In this case, the complementary channel is always the channel next to the base channel. In the case of a single channel, the generated signal is equal to that generated by the base channel of a complementary pair.

![Figure 3. Examples of a PWM Signal Generated by a Single-channel Phase](image1)

![Figure 4. Examples of a PWM Signal Generated by a Complementary-pair Phase](image2)

3.2.2 Alignment: Edge-aligned or Center-aligned

The alignment option is common for all phases. The following figures depict examples of edge-aligned and center-aligned PWM signals.

![Figure 5. Examples of Edge-aligned PWM Signals](image3)
3.2.3 Polarity: Active High or Active Low

The polarity of each generated PWM signal is selectable. It can be either active-high polarity or active-low polarity.

3.2.4 Negate Duty: Transform Duty-Cycle Value to its Opposite

The negate duty option enables transforming the duty cycle value from the range 0% - 100% into the range 100% - 0%. If the negate duty option is set, the generated active-time is equal to the difference in the PWM period and the calculated active-time. The negate duty option is useful for driving a DC or BLDC motor.
3.2.5 Swap: Swap Dead-Time Insertion

Setting the swap option, together with changing the generated PWM signal polarities, enables swapping the base and the complementary channel.
### 3.2.6 Dead-Time Insertion

Figure 11 and Figure 12 depict the dead-time insertion effect.

**Figure 10. Effect of the Swap Dead-time Option**

- **Swap Dead-Time Option Cleared**
  - Base Channel
  - Complementary Channel

- **Swap Dead-Time Option Set**
  - Base Channel
  - Complementary Channel

- **Swap Dead-Time Option Set, Opposite Channel Polarities**
  - Base Channel
  - Complementary Channel

**Figure 11. Effect of Dead-Time Insertion if Swap Option is Cleared**

- Base Channel
  - Complementary Channel

**Figure 12. Effect of Dead-Time Insertion if Swap Option is Set**

- Base Channel
  - Complementary Channel
3.2.7 Activity: Channel Active or Channel Passive

Each PWMC phase channel can be initialized as active or passive. If a channel is passive, it can be commutated low or high using commutation commands OFF_LOW and OFF_HIGH, but the channel does not generate any PWM signal. A passive PWMC channel does not increase the overall eTPU busy-time. If a channel is active, it can generate a PWM signal and can be commutated on and off. See 3.5 for more information on commutation commands.

3.3 Full Duty-Cycle Range and Minimum Pulse Width

Both the PWMF and PWMC phases may limit the minimum pulse width of the generated PWM signals. The minimum pulse width is an adjustable parameter that is common for all phases.

Furthermore, the PWMF function allows for generation of 0% and 100% duty-cycle PWM signals, meaning generating no pulse. The PWMMDC function generates a 0% or 100% duty-cycle if the calculated pulse width is less than half of the specified minimum pulse width.

3.4 Normal and Half-Cycle Reload

The update of the generated PWM signals can be once per several PWM periods or each PWM period in the normal update mode, and up to twice per PWM period in half-cycle update mode. Near the end of each PWM period, the PWMMDC function checks for an update. If new input values are available, they are applied in the next PWM period. Furthermore, in half-cycle update mode, the check is also performed near the end of the first half of each PWM period. If new input values are available, they are applied from the second half of the PWM period. The half-cycle update mode can only be used with center-aligned PWM alignment.

![Figure 13. Normal Update](image-url)
### 3.5 Commutations

When the PWMC function is used, the phases can be commutated. The commutation of a phase means changing a commutation state for each of the generated PWM signals. The commutation states are as follows:

- PWM generation is ON, the generated signal polarity is active-high
- PWM generation is ON, the generated signal polarity is active-low
- PWM generation is OFF, the pin polarity is low
- PWM generation is OFF, the pin polarity is high

At the same time, the commutation can also change the phase options: negate duty-cycle and swap dead-time insertion.

A commutation takes place immediately after a 32-bit commutation command is executed. The commutation command can be executed by another eTPU function (for example, hall decoder) or by the CPU, and the commutation command consists of the following fields:

- Channel number of the base channel
- New base channel commutation state:
  - ON_ACTIVE_HIGH
  - ON_ACTIVE_LOW
  - OFF_LOW
  - OFF_HIGH
- New complementary channel commutation state. If the complementary channel is not used (the phase type is single channel), set:
  - NOT_USED
Using the DC Motor Control PWM eTPU Functions, Rev. 1

3.6 Example Configurations

The PWMMDC, PWMF, and PWMC functions have been designed to provide as much flexibility as possible in the generation of the PWM signals that drive a motor. This flexibility may allow the PWMMDC, PWMF, and PWMC functions to meet the needs of an unusual drive scheme. However, since the primary purpose of the PWMMDC, PWMF, and PWMC functions is to drive motors in a conventional manner, the typical configurations are depicted in Figures 15 to 18.

**Figure 15. DC Motor - Unipolar Drive**

**Phase A**

Base Channel

Complementary Channel

**Phase B**

Base Channel

Complementary Channel

**PWMMDC Settings**

- **PhaseA_negate_duty:** DUTY_POS
- **PhaseB_negate_duty:** DUTY_NEG
- **Modulation:** VOLTAGE_SIGNED
- **Update:** NORMAL
- **Alignment:** CENTER_ALIGNED
- **Phases_type:** FULL_RANGE_COMPL_PAIRS
- **Swap:** NO_SWAP
- **Base_ch_polarity:** BASE_ACTIVE_HIGH
- **Compl_ch_polarity:** COMPL_ACTIVE_LOW
**Function Description**

**Figure 16. BLDC Motor - Unipolar 2-Quadrant Only Drive**

- **Phase A**
  - Base Channel
  - Complementary Channel

- **Phase B**
  - Base Channel
  - Complementary Channel

- **Phase C**
  - Base Channel
  - Complementary Channel

**PWMMDC Settings**
- PhaseA_negate_duty: DUTY_POS
- PhaseB_negate_duty: DUTY_POS
- PhaseC_negate_duty: DUTY_POS
- Modulation: VOLTAGE_UNSIGNED
- Update: NORMAL
- Alignment: EDGE_ALIGNED
- Phases_type: COMMUTATED_COMPL_PAIRS
- Swap: NO_SWAP
- Base_ch_polarity: BASE_ACTIVE_HIGH
- Compl_ch_polarity: COMPL_ACTIVE_LOW

**Commutation Commands Set**
- Base Channels: ON_ACTIVE_HIGH/OFF_LOW
- Compl_channels: OFF_LOW/OFF_HIGH
- Phase Options: –
Using the DC Motor Control PWM eTPU Functions, Rev. 1

Figure 17. BLDC Motor - Bipolar 4-Quadrant Drive
3.7 Interrupts

The PWMMDC function periodically generates an interrupt service request to the CPU. The generation of interrupt service requests depends on the type of update: one interrupt per PWM period is generated in the normal update mode; two interrupts per PWM period are generated in the half-cycle update mode.

3.8 Performance

Like all eTPU functions, the PWMMDC, PWF, and PWMC function performance in an application is, to some extent, dependent on the service time (latency) of other active eTPU channels. This is due to the operational nature of the scheduler. The performance limits of the PWMMDC function, with PWF or PWMC phases, can be expressed by these parameters:

- **minimum update_time**
  
  *update_time* is the time necessary to perform the update of all PWF or PWMC channels for the next PWM period. New input values must be set to the eTPU at least *update_time* before the next
Function Description

period, in order that they are applied in the next PWM period. The minimum update-time increases if another eTPU function may interfere into the update.

- **maximum eTPU busy-time during one period**
  This value, compared to the period value, determines the proportional load on the eTPU engine caused by the PWMMDC, and PWMF or PWMC functions.

Table 1 lists the performance limits if a single PWMMDC function with several PWMF phases is in use. Table 2 lists the performance limits if a single PWMMDC function with several PWMC phases is in use.

### Table 1.
Performance Limits of Various Configurations of PWMMDC with PWMF Phases

<table>
<thead>
<tr>
<th>num_phases</th>
<th>configuration</th>
<th>modulation</th>
<th>update</th>
<th>phases_type</th>
<th>min update_time</th>
<th>max eTPU busy time during one period</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VOLTAGE_UNSIGNED</td>
<td>NORMAL</td>
<td>SINGLE_CHANNELS</td>
<td></td>
<td>304</td>
<td>340</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_UNSIGNED</td>
<td>NORMAL</td>
<td>COMPL_PAIRS</td>
<td></td>
<td>406</td>
<td>476</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_UNSIGNED</td>
<td>HALF_CYCLE</td>
<td>SINGLE_CHANNELS</td>
<td></td>
<td>316</td>
<td>618</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_UNSIGNED</td>
<td>HALF_CYCLE</td>
<td>COMPL_PAIRS</td>
<td></td>
<td>422</td>
<td>830</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_SIGNED</td>
<td>NORMAL</td>
<td>SINGLE_CHANNELS</td>
<td></td>
<td>304</td>
<td>340</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_SIGNED</td>
<td>NORMAL</td>
<td>COMPL_PAIRS</td>
<td></td>
<td>406</td>
<td>476</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_SIGNED</td>
<td>HALF_CYCLE</td>
<td>SINGLE_CHANNELS</td>
<td></td>
<td>316</td>
<td>618</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_SIGNED</td>
<td>HALF_CYCLE</td>
<td>COMPL_PAIRS</td>
<td></td>
<td>422</td>
<td>830</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_UNSIGNED</td>
<td>NORMAL</td>
<td>SINGLE_CHANNELS</td>
<td></td>
<td>724</td>
<td>828</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_UNSIGNED</td>
<td>NORMAL</td>
<td>COMPL_PAIRS</td>
<td></td>
<td>1030</td>
<td>1236</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_UNSIGNED</td>
<td>HALF_CYCLE</td>
<td>SINGLE_CHANNELS</td>
<td></td>
<td>752</td>
<td>1458</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_UNSIGNED</td>
<td>HALF_CYCLE</td>
<td>COMPL_PAIRS</td>
<td></td>
<td>1070</td>
<td>2094</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_SIGNED</td>
<td>NORMAL</td>
<td>SINGLE_CHANNELS</td>
<td></td>
<td>724</td>
<td>828</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_SIGNED</td>
<td>NORMAL</td>
<td>COMPL_PAIRS</td>
<td></td>
<td>1030</td>
<td>1236</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_Signed</td>
<td>HALF_CYCLE</td>
<td>SINGLE_CHANNELS</td>
<td></td>
<td>752</td>
<td>1458</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_Signed</td>
<td>HALF_CYCLE</td>
<td>COMPL_PAIRS</td>
<td></td>
<td>1070</td>
<td>2094</td>
</tr>
<tr>
<td>3</td>
<td>NO</td>
<td>NORMAL</td>
<td>SINGLE_CHANNELS</td>
<td></td>
<td>712</td>
<td>812</td>
</tr>
<tr>
<td>3</td>
<td>NO</td>
<td>NORMAL</td>
<td>COMPL_PAIRS</td>
<td></td>
<td>1018</td>
<td>1220</td>
</tr>
<tr>
<td>3</td>
<td>NO</td>
<td>HALF_CYCLE</td>
<td>SINGLE_CHANNELS</td>
<td></td>
<td>740</td>
<td>1428</td>
</tr>
<tr>
<td>3</td>
<td>NO</td>
<td>HALF_CYCLE</td>
<td>COMPL_PAIRS</td>
<td></td>
<td>1058</td>
<td>2064</td>
</tr>
</tbody>
</table>
Table 2. Performance Limits of Various Configurations of PWMMDC with PWMC Phases

<table>
<thead>
<tr>
<th>num_phases</th>
<th>modulation</th>
<th>update</th>
<th>phases_type</th>
<th>min update_time</th>
<th>max eTPU busy time during one period</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VOLTAGE_UNSIGNED</td>
<td>NORMAL</td>
<td>SINGLE_CHANNELS</td>
<td>270</td>
<td>290</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_UNSIGNED</td>
<td>NORMAL</td>
<td>COMPLPAIRS</td>
<td>336</td>
<td>378</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_UNSIGNED</td>
<td>HALF_CYCLE</td>
<td>SINGLE_CHANNELS</td>
<td>282</td>
<td>540</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_UNSIGNED</td>
<td>HALF_CYCLE</td>
<td>COMPLPAIRS</td>
<td>352</td>
<td>676</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_SIGNED</td>
<td>NORMAL</td>
<td>SINGLE_CHANNELS</td>
<td>270</td>
<td>290</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_SIGNED</td>
<td>NORMAL</td>
<td>COMPLPAIRS</td>
<td>336</td>
<td>378</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_SIGNED</td>
<td>HALF_CYCLE</td>
<td>SINGLE_CHANNELS</td>
<td>282</td>
<td>540</td>
</tr>
<tr>
<td>1</td>
<td>VOLTAGE_SIGNED</td>
<td>HALF_CYCLE</td>
<td>COMPLPAIRS</td>
<td>352</td>
<td>676</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_UNSIGNED</td>
<td>NORMAL</td>
<td>SINGLE_CHANNELS</td>
<td>618</td>
<td>682</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_UNSIGNED</td>
<td>NORMAL</td>
<td>COMPLPAIRS</td>
<td>816</td>
<td>946</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_UNSIGNED</td>
<td>HALF_CYCLE</td>
<td>SINGLE_CHANNELS</td>
<td>646</td>
<td>1228</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_UNSIGNED</td>
<td>HALF_CYCLE</td>
<td>COMPLPAIRS</td>
<td>856</td>
<td>1636</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_SIGNED</td>
<td>NORMAL</td>
<td>SINGLE_CHANNELS</td>
<td>618</td>
<td>682</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_SIGNED</td>
<td>NORMAL</td>
<td>COMPLPAIRS</td>
<td>816</td>
<td>946</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_SIGNED</td>
<td>HALF_CYCLE</td>
<td>SINGLE_CHANNELS</td>
<td>646</td>
<td>1228</td>
</tr>
<tr>
<td>3</td>
<td>VOLTAGE_SIGNED</td>
<td>HALF_CYCLE</td>
<td>COMPLPAIRS</td>
<td>856</td>
<td>1636</td>
</tr>
<tr>
<td>3</td>
<td>NO</td>
<td>NORMAL</td>
<td>SINGLE_CHANNELS</td>
<td>604</td>
<td>668</td>
</tr>
<tr>
<td>3</td>
<td>NO</td>
<td>NORMAL</td>
<td>COMPLPAIRS</td>
<td>802</td>
<td>932</td>
</tr>
<tr>
<td>3</td>
<td>NO</td>
<td>HALF_CYCLE</td>
<td>SINGLE_CHANNELS</td>
<td>632</td>
<td>1200</td>
</tr>
<tr>
<td>3</td>
<td>NO</td>
<td>HALF_CYCLE</td>
<td>COMPLPAIRS</td>
<td>842</td>
<td>1608</td>
</tr>
</tbody>
</table>

The eTPU module clock is equal to the CPU clock on MPC5500 devices, as well as the peripheral clock, (half of the CPU clock) on MCF523x devices. For example, the eTPU module clock is 132 MHz on a 132-MHz MPC5554, and one eTPU cycle takes 7.58ns, while the eTPU module clock is only 75 MHz on a 150-MHz MCF5235, and one eTPU cycle takes 13.33ns.

The performance is influenced by compiler efficiency. The above numbers, measured on the code compiled by eTPU compiler version 1.0.0.5, are given for guidance only and are subject to change. For up to date information, refer to the information provided in the particular eTPU function set release available from Freescale.
4 C Level API for Function

The following routines provide easy access for the application developer to the PWMMDC, PWF, and PWMC functions. Use of these functions eliminates the need to directly control the eTPU registers.

There are 15 functions added to the application PWMMDC programming interface (API). The routines can be found in the etpu_pwmmdc.h and etpu_pwmmdc.c files, which should be linked with the top level development file(s).

Figure 19 shows the PWMMDC API state flow and lists API functions which can be used in each of its states.

---

**Figure 19. PWMMDC API State Flow**

All PWMMDC API routines will be described in order and are listed below:

- **Initialization Functions:**
  ```c
  int32_t fs_etpu_pwmmdc_init_3ph( uint8_t channel,
                                  uint8_t priority,
                                  uint8_t phaseA_channel,
                                  uint8_t phaseA_negate_duty,
                                  uint8_t phaseB_channel,
                                  uint8_t phaseB_negate_duty,
                                  uint8_t phaseC_channel,
                                  uint8_t phaseC_negate_duty,
                                  uint8_t modulation,
  ```
C Level API for Function

Using the DC Motor Control PWM eTPU Functions, Rev. 1

Freescale Semiconductor
uint24_t start_offset,
uint24_t period,
uint24_t update_time,
uint24_t dead_time,
uint24_t min_pw)

• Change Operation Functions:

int32_t fs_etpu_pwmmdc_enable_3ph( uint8_t channel,
                                    uint8_t base_ch_polarity,
                                    uint8_t compl_ch_polarity)

int32_t fs_etpu_pwmmdc_enable_2ph( uint8_t channel,
                                    uint8_t base_ch_polarity,
                                    uint8_t compl_ch_polarity)

int32_t fs_etpu_pwmmdc_enable_1ph( uint8_t channel,
                                    uint8_t base_ch_polarity,
                                    uint8_t compl_ch_polarity)

int32_t fs_etpu_pwmmdc_disable_3ph( uint8_t channel,
                                     uint8_t base_ch_disable_pin_state,
                                     uint8_t compl_ch_disable_pin_state)

int32_t fs_etpu_pwmmdc_disable_2ph( uint8_t channel,
                                     uint8_t base_ch_disable_pin_state,
                                     uint8_t compl_ch_disable_pin_state)

int32_t fs_etpu_pwmmdc_disable_1ph( uint8_t channel,
                                     uint8_t base_ch_disable_pin_state,
                                     uint8_t compl_ch_disable_pin_state)

int32_t fs_etpu_pwmmdc_set_period( uint8_t channel,
                                     uint24_t period);

int32_t fs_etpu_pwmmdc_update_voltage_unsigned( uint8_t channel,
                                              uint24_t voltage)

int32_t fs_etpu_pwmmdc_update_voltage_signed( uint8_t channel,
                                             int24_t voltage)

int32_t fs_etpu_pwmmdc_update_duty_cycles( uint8_t channel,
                                          int24_t dutyA,
                                          int24_t dutyB,
                                          int24_t dutyC)
int32_t fs_etpu_pwmmdc_commutate( uint8_t phase_channel,
          uint32_t commut_cmd)

• Value Return Functions:
  
  uint32_t fs_etpu_pwmmdc_commut_cmd( uint8_t phase_channel,
          uint8_t base_ch_cmd,
          uint8_t compl_ch_cmd,
          uint8_t negate_duty,
          uint8_t swap)

4.1 Initialization Function

4.1.1 int32_t fs_etpu_pwmmdc_init_1ph(...),
  int32_t fs_etpu_pwmmdc_init_2ph(...),
  int32_t fs_etpu_pwmmdc_init_3ph(...)

These routines are used to initialize the eTPU channels for the PWMMDC function and set the output
signal(s) to the inactive state, based on the base_ch_disable_pin_state and compl_ch_disable_pin_state
parameters. These functions, which differ in number of phases they initialize (1, 2, or 3), have the
following parameters:

• channel (uint8_t) - This is the PWMMDC channel number. This parameter should be assigned a
  value of 0-31 for ETPU_A, and 64-95 for ETPU_B.

• priority (uint8_t) - This is the priority to assign to the PWMMDC function. This parameter
  should be assigned a value of:
  — FS_ETPU_PRIORITY_HIGH,
  — FS_ETPU_PRIORITY_MIDDLE or
  — FS_ETPU_PRIORITY_LOW

• phaseA_channel (uint8_t) - This parameter determines phase A base channel number. This
  parameter should be assigned a value of 0-31 for ETPU_A, and 64-95 for ETPU_B.

• phaseA_negate_duty (uint8_t) - This parameter sets the negate duty-cycle option for phase A.
  This option can be changed in run-time by applying a commutation command. For more
  information, refer to 3.2.4. This parameter should be assigned a value of:
  — FS_ETPU_PWMMDC_DUTY_POS or
  — FS_ETPU_PWMMDC_DUTY_NEG

• phaseB_channel (uint8_t) - This parameter determines the phase B base channel number. This
  parameter should be assigned a value of 0-31 for ETPU_A, and 64-95 for ETPU_B.

• phaseB_negate_duty (uint8_t) - This parameter sets the negate duty-cycle option for phase B.
  This option can be changed in run-time by applying a commutation command. For more
  information, refer to 3.2.4.
This parameter should be assigned a value of:

- FS_ETPU_PWMMDC_DUTY_POS or
- FS_ETPU_PWMMDC_DUTY_NEG

- **phaseC_channel (uint8_t)** - This parameter determines Phase C base the channel number. This parameter should be assigned a value of 0-31 for ETPU_A, and 64-95 for ETPU_B.

- **phaseC_negate_duty (uint8_t)** - This parameter sets the negate duty-cycle option for phase C. This option can be changed in run-time by applying a commutation command. For more information, refer to 3.2.4. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_DUTY_POS or
  - FS_ETPU_PWMMDC_DUTY_NEG

- **modulation (uint8_t)** - This parameter determines the type of modulation. For more information, refer to 3.1. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_MOD_VOLTAGE_UNSIGNED,
  - FS_ETPU_PWMMDC_MOD_VOLTAGE_SIGNED or
  - FS_ETPU_PWMMDC_MOD_NO

- **update (uint8_t)** - This parameter determines the type of update. For more information, refer to 3.4. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_NORMAL or
  - FS_ETPU_PWMMDC_HALF_CYCLE

- **alignment (uint8_t)** - This parameter determines the type of PWM alignment. For more information, refer to 3.2.2. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_EDGE_ALIGNED or
  - FS_ETPU_PWMMDC_CENTER_ALIGNED

- **phases_type (uint8_t)** - This parameter determines the type of all the PWMF or PWMC phases. For more information, refer to 2 and 3.2.1. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_FULL_RANGE_SINGLE_CHANNELS or
  - FS_ETPU_PWMMDC_FULL_RANGE_COMPLPAIRS or
  - FS_ETPU_PWMMDC_COMMUTATED_SINGLE_CHANNELS or
  - FS_ETPU_PWMMDC_COMMUTATED_COMPLPAIRS

- **swap (uint8_t)** - This parameter sets the swap dead-time insertion option. For more information, refer to 3.2.5. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_NO_SWAP or
  - FS_ETPU_PWMMDC_SWAP

- **base_ch_disable_pin_state (uint8_t)** - This is the required output state of the base channel pin, after initialization. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_PIN_LOW or
  - FS_ETPU_PWMMDC_PIN_HIGH
• **compl_ch_disable_pin_state (uint8_t)** - This is the required output state of the complementary channel pin, after initialization. This parameter should be assigned a value of:
  — FS_ETPU_PWMMDC_PIN_LOW or
  — FS_ETPU_PWMMDC_PIN_HIGH

• **start_offset (uint24_t)** - This parameter is used to synchronize various eTPU functions that generate a signal. The first PWM period starts start_offset TCR1 clocks after initialization.

• **period (uint24_t)** - This parameter determines the PWM period as a number of TCR1 cycles.

• **update_time (uint24_t)** - This parameter determines the time that is necessary to perform the update of all PWM phases, as a number of TCR1 cycles.

• **dead_time (uint24_t)** - This parameter determines the dead-time as a number of TCR1 cycles.

• **min_pw (uint24_t)** - This parameter determines the minimum pulse width, as a number of TCR1 cycles.

### 4.2 Change Operation Functions

**4.2.1 int32_t fs_etpu_pwmmmdc_enable_1ph(...),
int32_t fs_etpu_pwmmmdc_enable_2ph(...),
int32_t fs_etpu_pwmmmdc_enable_3ph(....)**

These routines are used to enable PWM generation. They differ in number of phases which are enabled (1, 2 or 3). These functions have the following parameters:

• **channel (uint8_t)** - This is the PWMMDC channel number. This parameter must be assigned the same value as the channel parameter of the initialization function was assigned.

• **base_ch_polarity (uint8_t)** - This parameter determines the polarity of the base channel. This parameter should be assigned a value of:
  — FS_ETPU_PWMMDC_ACTIVE_HIGH or
  — FS_ETPU_PWMMDC_ACTIVE_LOW

• **compl_ch_polarity (uint8_t)** - This parameter determines the polarity of the complementary channel. This parameter should be assigned a value of:
  — FS_ETPU_PWMMDC_ACTIVE_HIGH or
  — FS_ETPU_PWMMDC_ACTIVE_LOW

  This parameter applies only if phase_type is FS_ETPU_PWMMDC_COMPL_PAIRS.

These functions return 0 if the PWM phases were successfully enabled. In case the phase channels have any pending HSRs, the phases are not enabled and these functions should be called again later. In this case, a sum of pending HSR numbers is returned.
4.2.2 int32_t fs_etpu_pwmmmdc_disable_1ph(...),
        int32_t fs_etpu_pwmmmdc_disable_2ph(...),
        int32_t fs_etpu_pwmmmdc_disable_3ph(...)

These routines are used to disable generation of the PWM signal(s) and set the output signal(s) to the inactive state, based on the base_ch_disable_pin_state and compl_ch_disable_pin_state parameter. These routines differ in number of phases they disable (1, 2 or 3). These functions have the following parameters:

- **channel (uint8_t)** - This is the PWMMDC channel number. This parameter must be assigned the same value as the channel parameter of the initialization function was assigned.

- **base_ch_disable_pin_state (uint8_t)** - This is the required output state of the base channel pins, after disable of PWM generation. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_PIN_LOW or
  - FS_ETPU_PWMMDC_PIN_HIGH

- **compl_ch_disable_pin_state (uint8_t)** - This is the required output state of the complementary channel pins, after disable of PWM generation. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_PIN_LOW or
  - FS_ETPU_PWMMDC_PIN_HIGH

This parameter applies only if phase_type is FS_ETPU_PWMMDC_COMPL_PAIRS.

These functions return 0 if the PWM phases were successfully disabled. In case the phase channels have any pending HSRs, the phases are not disabled and this function should be called again later. In this case, a sum of pending HSR numbers is returned.

4.2.3 int32_t fs_etpu_pwmmmdc_set_period( uint8_t channel,
                                                   uint24_t period)

This function sets the PWM period. It has the following parameters:

- **channel (uint8_t)** - This is the PWMMDC channel number. This parameter must be assigned the same value as the channel parameter of the initialization function was assigned.

- **period (uint24_t)** - This parameter determines the PWM period as a number of TCR1 ticks. If the application uses PWM frequency in Hz, instead of PWM period in TCR1 ticks, one of the following expressions can be used instead of period:
  
  etpu_a_tcr1_freq/PWM_frequency
  etpu_b_tcr1_freq/PWM_frequency

If the new period value is set update_time before the start of the next PWM period, it is applied immediately from the next PWM period.
4.2.4 \texttt{int32_t fs_etpu_pwmmdc_update_voltage_unsigned( uint8_t channel, uint24_t voltage)}

This function sets the applied motor voltage in case of voltage-unsigned modulation (the \textit{modulation} parameter is set to \texttt{FS_ETPU_PWMMDC_MOD_VOLTAGE_UNSIGNED}). It has the following parameters:

- \textbf{channel (uint8_t)} - This is the PWMMDC channel number. This parameter must be assigned the same value as the channel parameter of the initialization function was assigned.

- \textbf{voltage (uint24_t)} - This parameter determines the applied motor voltage in the range \((0, 2^{24}-1)\). The voltage range \((0, 2^{24}-1)\) corresponds to the duty-cycle range \((0\%, 100\%)\).

This function returns 0 if the input voltage was successfully updated. In case the master channel has any pending HSR, the input voltage is not updated and this function should be called again later. In this case, the pending HSR number is returned.

4.2.5 \texttt{int32_t fs_etpu_pwmmdc_update_voltage_signed( uint8_t channel, int24_t voltage)}

This function sets the applied motor voltage in case of voltage-signed modulation (the \textit{modulation} parameter is set to \texttt{FS_ETPU_PWMMDC_MOD_VOLTAGE_SIGNED}). This function has the following parameters:

- \textbf{channel (uint8_t)} - This is the PWMMDC channel number. This parameter must be assigned the same value as the channel parameter of the initialization function was assigned.

- \textbf{voltage (int24_t)} - This parameter determines the applied motor voltage in the range \((-2^{23}, 2^{23}-1)\). The voltage range \((-2^{23}, 2^{23}-1)\) corresponds to the duty-cycle range \((0\%, 100\%)\).

This function returns 0 if the input voltage was successfully updated. In case the master channel has any pending HSR, the input voltage is not updated and this function should be called again later. In this case, the pending HSR number is returned.

4.2.6 \texttt{int32_t fs_etpu_pwmmdc_update_duty_cycles( uint8_t channel, int24_t dutyA, int24_t dutyB, int24_t dutyC)}

This function updates phase duty-cycles in case of no modulation (the \textit{modulation} parameter is set to \texttt{FS_ETPU_PWMMDC_MOD_NO}). It has the following parameters:

- \textbf{channel (uint8_t)} - This is the PWMMDC channel number. This parameter must be assigned the same value as the channel parameter of the initialization function was assigned.

- \textbf{dutyA (int24_t)} - This parameter determines the phase A duty-cycle in range \((-2^{23},2^{23}-1)\). The voltage range \((-2^{23}, 2^{23}-1)\) corresponds to the duty-cycle range \((0\%, 100\%)\).

- \textbf{dutyB (int24_t)} - This parameter determines the phase B duty-cycle in range \((-2^{23},2^{23}-1)\). The voltage range \((-2^{23}, 2^{23}-1)\) corresponds to the duty-cycle range \((0\%, 100\%)\).

- \textbf{dutyC (int24_t)} - This parameter determines the phase C duty-cycle in range \((-2^{23},2^{23}-1)\). The voltage range \((-2^{23}, 2^{23}-1)\) corresponds to the duty-cycle range \((0\%, 100\%)\).
This function returns 0 if the duty-cycles were successfully updated. In case the master channel has any pending HSR, the duty-cycles are not updated and this function should be called again later. In this case, the pending HSR number is returned.

### 4.2.7 int32_t fs_etpu_pwmmdc_commutate( uint8_t phase_channel, uint32_t commut_cmd)

This function commutate on PWMC phase. Commutation command can be safely applied only to phases of type FS_ETPU_PWMMDC_COMMUTATED_SINGLE_CHANNELS or FS_ETPU_PWMMDC_COMMUTATED_COMPL_PAIRS. This function has the following parameters:

- **phase_channel (uint8_t)** - This is a PWMMDC phase channel number. This parameter must be assigned the same value as a phaseX_channel parameter of the initialization function was assigned.
- **commut_cmd (uint32_t)** - This parameter is a 32-bit commutation command. This parameter could be created using the `fs_etpu_pwmmdc_commut_cmd( )` function. See 4.3.1 for more description.

### 4.3 Value Return Function

#### 4.3.1 uint32_t fs_etpu_pwmmdc_commut_cmd(...)

This function composes and returns a 32-bit commutation command word. This created commutation command can be used by the `fs_etpu_pwmmdc_commutate` function. This function has the following parameters:

- **phase_channel (uint8_t)** - This is the PWMC phase base channel number. This parameter should be assigned a value of 0-31 for ETPU_A, and 64-95 for ETPU_B.
- **base_ch_cmd (uint8_t)** - This parameter sets the base channel commutation byte. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_OFF_LOW or
  - FS_ETPU_PWMMDC_OFF_HIGH or
  - FS_ETPU_PWMMDC_ON_ACTIVE_HIGH or
  - FS_ETPU_PWMMDC_ON_ACTIVE_LOW
- **compl_ch_cmd (uint8_t)** - This parameter sets the complementary channel commutation byte. This parameter should be assigned a value of:
  - FS_ETPU_PWMMDC_COMPL_NOT_USED if phases_type was initialized to
    - FS_ETPU_PWMMDC_COMMUTATED_SINGLE_CHANNELS, or
  - FS_ETPU_PWMMDC_OFF_LOW or
  - FS_ETPU_PWMMDC_OFF_HIGH or
  - FS_ETPU_PWMMDC_ON_ACTIVE_HIGH or
  - FS_ETPU_PWMMDC_ON_ACTIVE_LOW
Example Use of Function

5.1 Demo Applications

The usage of the PWMMDC, PWMF and PWMC eTPU functions is demonstrated in the following applications:

- “3-Phase BLDC Motor with Speed Closed Loop, driven by eTPU on MCF523x,” AN2892.
- “Three 3-Phase BLDC Motors with Speed Closed Loop, driven by eTPU on MCF523x,” AN2948.
- “DC Motor with Speed and Current Closed Loop, driven by eTPU on MCF523x,” AN2955.
- “BLDC Motor with Quadrature Encoder and Speed Closed Loop, driven by eTPU on MCF523x,” AN2957.

For a detailed description of the demo applications refer to the mentioned application notes.

Example of PWM functions initialization, assignment of the PWMMDC, PWMC and PWMF functions to an eTPU channels and using of API functions are stated in the following paragraphs.

5.1.1 Function Calls: PWMMDC+PWMC

/* initialize PWMMDC eTPU function with 3 PWMC functions*/
err_code = fs_etpu_pwmmdc_init_3ph (PWMMDC0_MASTER,/* engine: A; channel: 7 */
FS_ETPU_PRIORITY_MIDDLE,/* priority: Middle */
PWMMDC0_PHASEA_BASE_CHANNEL,/* engine: A; channel: 8 */
FS_ETPU_PWMMDC_DUTY_POS,/* phaseA_negate_duty: do not negate duty-cycle */
PWMMDC0_PHASEB_BASE_CHANNEL,/* engine: A; channel: 10 */
FS_ETPU_PWMMDC_DUTY_POS,/* phaseB_negate_duty: do not negate duty-cycle */
Example Use of Function

PWMMDC0_PHASEC_BASE_CHANNEL,/* engine: A; channel: 12 */
FS_ETPU_PWMMDC_DUTY_POS,/* phaseC_negate_duty: do not negate duty-cycle */
FS_ETPU_PWMMDC_MOD_VOLTAGE_SIGNED,/* modulation: signed voltage */
FS_ETPU_PWMMDC_NORMAL,/* update: normal update */
FS_ETPU_PWMMDC_CENTER_ALIGNED,/* alignment: center-aligned */
FS_ETPU_PWMMDC_COMMUTATED_COMPL_PAIRS,/* phases_type: commutated compl. pairs */
FS_ETPU_PWMMDC_NO_SWAP,/* swap: do not swap dead-time insertion */
FS_ETPU_PWMMDC_PIN_LOW,/* base_ch_disable_pin_state: active high */
FS_ETPU_PWMMDC_PIN_LOW,/* compl_ch_disable_pin_state: active high */
10000, /* start_offset: 10000 */
etpu_a_tcr1_freq/16000,/* period: etpu_a_tcr1_freq/16000 */
750, /* update_time: 750 */
50, /* dead_time: 50 */
50); /* min_pw: 100 */

/* Enable generation of PWM signals */
fs_etpu_pwmmdc_enable_3ph(PWMMDC0_MASTER,
    FS_ETPU_PWMMDC_ACTIVE_HIGH,
    FS_ETPU_PWMMDC_ACTIVE_LOW);

/* Update Applied Motor Voltage */
fs_etpu_pwmmdc_update_voltage_signed_int(PWMMDC0_MASTER,0x4CCCCC); //60%

/* Commutate phase A OFF */
comm_cmd = fs_etpu_pwmmdc_commut_cmd(PWMMDC0_PHASEA_BASE_CHANNEL,
    FS_ETPU_PWMMDC_OFF_LOW,
    FS_ETPU_PWMMDC_OFF_LOW,
    FS_ETPU_PWMMDC_DUTY_POS,
    FS_ETPU_PWMMDC_NO_SWAP);
fs_etpu_pwmmdc_commutate(PWMMDC0_PHASEA_BASE_CHANNEL, comm_cmd);

/* Disable generation of PWM signals */
5.1.2 Function Calls: PWMMDC+PWMF

/* initialize PWMMDC eTPU function with 2 PWMF functions */
err_code = fs_etpu_pwmmdc_init_2ph (PWMMDC0_MASTER, /*engine: A; channel: 7 * /
    FS_ETPU_PRIORITY_MIDDLE, /* priority: Middle */
    PWMMDC0_PHASEA_BASE_CHANNEL, /* engine: A; channel: 8 */
    FS_ETPU_PWMMDC_DUTY_POS, /* phaseA_negate_duty: do not negate duty-cycle */
    PWMMDC0_PHASEB_BASE_CHANNEL, /* engine: A; channel: 10 */
    FS_ETPU_PWMMDC_DUTY_NEG, /* phaseB_negate_duty: negate duty-cycle */
    FS_ETPU_PWMMDC_MOD_VOLTAGE_SIGNED, /* modulation: signed voltage */
    FS_ETPU_PWMMDC_NORMAL, /* update: normal update */
    FS_ETPU_PWMMDC_CENTER_ALIGNED, /* alignment: center-aligned */
    FS_ETPU_PWMMDC_FULL_RANGE_COMPL_PAIRS, /* phases_type: full range compl. pairs */
    FS_ETPU_PWMMDC_NO_SWAP, /* swap: do not swap dead-time insertion */
    FS_ETPU_PWMMDC_PIN_LOW, /* base_ch_disable_pin_state: pin low */
    FS_ETPU_PWMMDC_PIN_LOW, /* compl_ch_disable_pin_state: pin low */
    10000, /* start_offset: 10000 */
    etpu_a_tcr1_freq/16000, /* period: etpu_a_tcr1_freq/16000 */
    750, /* update_time: 750 */
    50, /* dead_time: 50 */
    100); /* min_pw: 100 */

/* Enable generation of PWM signals */
fs_etpu_pwmmdc_enable_2ph(PWMMDC0_MASTER, 
    FS_ETPU_PWMMDC_ACTIVE_HIGH, 
    FS_ETPU_PWMMDC_ACTIVE_LOW);

/* Update Applied Motor Voltage */
fs_etpu_pwmmdc_update_voltage_signed_int(PWMMDC0_MASTER,0x4CCCCC); //60%
Using the DC Motor Control PWM eTPU Functions, Rev. 1

Summary and Conclusions

/* Disable generation of PWM signals */

fs_etpu_pwmmdc_disable_2ph(PWMMDC0_MASTER,
                         FS_ETPU_PWMMDC_PIN_LOW,
                         FS_ETPU_PWMMDC_PIN_LOW);

6 Summary and Conclusions

This application note provides the user with a description of Motor Control PWM (PWMMDC, PWMF and PWMC) eTPU functions usage and examples. The simple C interface routines to the DC Motor Control PWM eTPU functions enable easy implementation of them in applications. The demo application is targeted at the MCF523x family of devices, but it could be easily reused with any device that has an eTPU.

7 References

2. “General C Functions for the eTPU,” AN2864.
3. “Using the DC Motor Control eTPU Function Set (set3),” AN2958.
4. Enhanced Time Processing Unit Reference Manual, ETPURM/D.
5. eTPU Graphical Configuration Tool, http://www.freescale.com/etpu, ETPUGCT.
6. “3-Phase BLDC Motor with Speed Closed Loop, driven by eTPU on MCF523x,” AN2892.
7. “Three 3-Phase BLDC Motors with Speed Closed Loop, driven by eTPU on MCF523x,” AN2948.
8. “DC Motor with Speed and Current Closed Loop, driven by eTPU on MCF523x,” AN2955.
9. “BLDC Motor with Quadrature Encoder and Speed Closed Loop, driven by eTPU on MCF523x” AN2957.
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 that 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™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.© Freescale Semiconductor, Inc. 2005. All rights reserved.

Document Number: AN2840
Rev. 1
04/2005