Freescale Semiconductor
Application Note

Document Number: AN4381
Rev. 0, 12/2011

Configuring the FlexTimer for Position and Speed Measurement with an Encoder

by: Matus Plachy
System Application Engineer,
Freescale Czech System Center
Rožnov pod Radhoštem, Czech Republic

1 Introduction

Electronically controlled 3-phase permanent-magnet synchronous motors (PMSM) are becoming more and more popular in a variety of industrial and appliance motor-control applications, thanks to features such as high efficiency, reliability, and power density. When high dynamic performance is required, the vector control approach of the PMS motor is used. The rotor position information is critical for successfully performing the vector control algorithm. Quadrature encoders are widely used in industrial motor control applications as precise rotor position sensors mounted directly on the motor shaft. This application note describes the procedure for configuring the FlexTimer module for decoding these quadrature encoder signals for position and speed measurement. It also gives an example of the position and speed calculation.

The FlexTimer is a complex, general-purpose timer module that also possesses special features dedicated to a motor control application. Besides decoding the

Contents

1 Introduction 1
2 FlexTimer overview 2
2.1 FlexTimer features. . . . 2
3 Quadrature incremental encoder. . . . 3
4 FTM initialization for rotor position measurement using a quadrature encoder 4
5 Position and speed calculation example 5
5.1 Alignment 5
5.2 Position calculation 5
5.3 Speed calculation 6
6 Conclusion. . . . 9
7 References 9
8 Acronyms and abbreviated terms 9

quadrature encoder signals, it can be configured for generation of the six-signal pulse-width modulation (PWM) to control 3-phase electric motors. This process is described in AN3729, “Using FlexTimer in ACIM/PMSM Motor Control Applications.” The FlexTimer module is a component of a peripheral set of some Freescale products, such as the high-end S08, ColdFire V1, and Kinetis families. Some devices have more FlexTimer modules implemented on the chip.

2 FlexTimer overview

The FlexTimer module (FTM) is a two to eight channel timer that supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The FTM time reference is a 16-bit counter that can be used as an unsigned or signed counter.

2.1 FlexTimer features

The FTM features include:

- Selectable source clock
  - — Source clock can be the system clock, the fixed frequency clock, or an external clock
  - — Fixed frequency clock is an additional clock input to allow the selection of an on-chip clock source other than the system clock
  - — Selecting an external clock connects the FTM clock to a chip level input pin, therefore allowing it to synchronize the FTM counter with an off-chip clock source
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- 16-bit counter
  - — It can be a free-running counter or a counter with an initial and final value
  - — The counting can be up or up-down
- Each channel can be configured for input capture, output compare, or edge-aligned PWM mode
- Input capture mode
  - — In input capture mode, the capture can occur on rising edges, falling edges, or both edges
  - — In input capture mode, an input filter can be selected for some channels
- Output compare mode, in which the output signal can be set, cleared, or toggled on match
- All channels configurable for center-aligned PWM mode
- Ability to combine each pair of channels to generate a PWM signal (with independent control of both edges of the PWM signal)
- Ability of FTM channels to operate as pairs with equal outputs, pairs with complementary outputs, or independent channels (with independent outputs)
- Deadtime insertion available for each complementary pair
- Generation of triggers (match trigger)
- Software control of PWM outputs
- Up to 4 fault inputs for global fault control
- Configurable polarity for each channel

- The generation of an interrupt per channel
- The generation of an interrupt when the counter overflows
- The generation of an interrupt when the fault condition is detected
- Synchronized loading of write-buffered FTM registers
- Write protection for critical registers
- Backwards compatible with TPM
- Testing of input captures for a stuck at zero and one conditions
- Dual-edge capture for pulse- and period-width measurement
- Quadrature decoder with input filters, relative position counting, and interrupt on position count or capture of position count on an external event

3 Quadrature incremental encoder

The quadrature incremental encoder is a position feedback device that provides incremental counts. The incremental encoder provides relative information, where the feedback signal is always referenced to a start position. This is different from absolute encoders, where each position is defined as a combination of *N* bits (binary or the more common Gray encoding), while the number of bits determines the resolution. Thus, the absolute position information is provided immediately after powering up the system. The information is usually passed over some serial communication interface to the controlling MCU. Because of the latency caused by decoding and encoding the position information, an absolute encoder for some applications (high-speed motion control) cannot be used. In such a case, an incremental encoder is the better choice, because the signals are directly processed by the controlling MCU's on-chip hardware.

The incremental encoders used in the motion-control applications are usually based on optical technology. The light emitted by an LED passes through the slots in metal discs and is detected by phototransistors. The output signals for an encoder with 1024 pulses are shown in Figure 1.

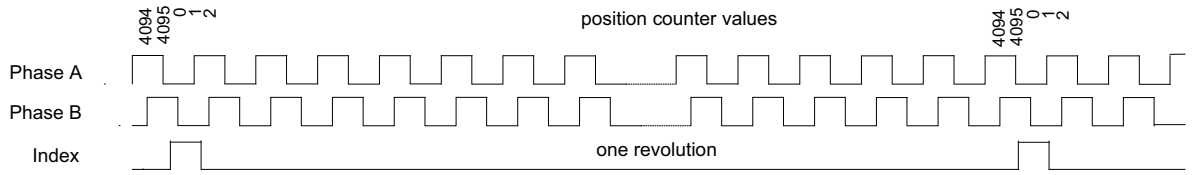

Figure 1. Quadrature encoder output signals

There are three output signals. The Phase A and Phase B signals consist of a series of pulses, the phase shifted by 90° (therefore the term *quadrature* is used). The third signal, here called "Index," provides the absolute position information. In motion control, it is used to check the pulse counting consistency. This means that after each revolution, the value of the counted pulses is captured and compared against the defined value. If there is a difference detected, the control algorithm then has to perform the position offset compensation. A possible loss of pulse should be detected and handled by the motor control application. The deviation in the real and measured values of the rotor position will influence the generation of the vector of the stator magnetic flux. An incrementing deviation during a long-term run of the motor yields significant torque drop, resulting in even a complete stop of the motor.

There are also encoders with an index signal that have more pulses per one revolution, or even two index signals.

The quadrature decoder in the MCU counts rising and falling edges of both the phase signals. Therefore, although the specification of the encoder quotes, for example, 1024 pulses per revolution, this value is multiplied by 4. So the internal quadrature decoder counter counts 4096 edges, as is shown in Figure 1. The internal logic of the quadrature decoder evaluates the direction of the rotation by increasing or decreasing the counter value.

4 FTM initialization for rotor position measurement using a quadrature encoder

The example of FTM initialization shown below is part of the motor control application described in DRM128, *PMSM Vector Control with Encoder on Kinetis* and is specific to the Kinetis K40 in a 144-pin package. Therefore, the module clock enabling and input pin configuration can be different on the other MCU families. There are three FTM modules on the Kinetis K40 MCU: one 8-channel module dedicated to generating the 6-channel PWM for running the 3-phase PMSM, and two 2-channel modules, one of them configured for decoding the quadrature encoder signals. The third FTM module is unused. Of course, the FTM with two channels allows the use of only the Phase A and Phase B signals to decode the position information. The processing of the index signal together with the phase signals can only be done in an FTM with three or more channels.

This document focuses below on the use of an FTM with two channels for position detection and the speed calculation. The capturing of the position count on an external event (index signal) has to be processed by software associated with the interrupt.

The external signals are routed to the input pins reflecting the hardware solution that is built on the Tower System modules (TWR-MK40 and TWR-MC-LV3PH), as it is described in DRM128. Next, it is assumed that the encoder has 1024 pulses. Please refer to the “Quadrature Decoder Mode” section of the *Kinetis K40 Sub-Family Reference Manual* for more details, as there are more options on setting the quadrature decoding mode based on the available encoder signals, as well as the method of counting-up and -down the edges.

The following is an initialization example for the Kinetis MK40X256VMD100:

```
    //enable the clock for FTM1
SIM_SCGC6 |= SIM_SCGC6_FTM1_MASK;
    //enable the counter
FTM1_MODE |= FTM_MODE_FTMEN_MASK;
    //enable the counter to run in the BDM mode
FTM1_CONF |= FTM_CONF_BDMMODE(3);
    //load the Modulo register and counter initial value
FTM1_MOD = 4095;
FTM1_CNTIN = 0;
    //configuring FTM for quadrature mode
FTM1_QDCTRL |= FTM_QDCTRL_QUADEN_MASK;
    // start the timer clock, source is the external clock
FTM1_SC |= FTM_SC_CLKS(3);
    //configuring the input pins:
PORTA_PCR8 = PORT_PCR_MUX(6); // FTM1 CH0
PORTA_PCR9 = PORT_PCR_MUX(6); // FTM1 CH1
```

5 Position and speed calculation example

The example of position and speed calculation is performed with the numbers represented in Q1.31 two-s complement, signed fractional format. So all physical quantities were scaled to the < –1,1) interval, where *–1* corresponds in the MCU to 2^{32} and *1* to $2^{31} – 1$.

For more information on the calculations in the fractional format and variables scaling, see DRM105, *3-phase PMSM Vector Control using Quadrature Encoder on MCF51AC256*, and DRM102, *PMSM Vector Control with Quadrature Encoder on Kinetis.*

5.1 Alignment

Because the quadrature encoder is a relative position sensor, it is necessary to know the exact position of the rotor before the motor is started. One possible and easy method is to align the rotor to a predefined position. A DC voltage is applied on the stator winding, so that the stator and rotor magnetic fields become aligned to one axis and the rotor is moved to a known position. After the alignment, the FTM counter value is set to zero. The alignment process referenced in the vector control algorithm is described in more detail in DRM128 and DRM105.

5.2 Position calculation

There is a direct proportion between the number of pulses counted by the quadrature encoder and the position. This proportion can be described by a simple equation:

Eqn. 1

$$theta\_el\_frac = counter\_value \times position\_scale$$

Where:

theta_el_frac is the electrical position scaled to Q1.31 fractional format.

counter_value is the value of the FlexTimer internal counter `FTM1_CNT` that counts the edges of the input signals.

position_scale is the value by which the *counter_value* has to be multiplied in order to get the electrical position. The way to determine its value is described below.

First of all, it is necessary to understand that the quadrature encoder gives information on the mechanical position, while to the vector control algorithm requires the electrical position. The relationship is given by the following formula:

Eqn. 2

$$\theta_{el} = pp \times \theta_{mech}$$

Where:

θ_{el} [rad] is the electrical position.

pp [–] is the number of pole pairs of the motor.

θ_{mech} [rad] is the mechanical position, determined by the quadrature encoder.

It is obvious that the electrical position is changed *pp* – times faster than the mechanical position, and the same relationship also holds between the electrical and mechanical angular speeds. Figure 2 depicts the relationship of the mechanical position given by the encoder and the electrical position, considering the six-pole motor (*pp* = 3) and the quadrature encoder with 1024 pulses. The FTM counts 4 × 1024 edges, giving the total value of 4095 (the 4096^{th} match to zero) per one mechanical revolution.

Figure 2. Relationship between the electrical position in Q1.31 format and the FTM counter value

Furthermore, since the Q1.31 numeric format is used, the maximum position corresponds to the maximum number of the selected number format; that is, for $2^{31}–1(\sim \pi)$, the minimum position value is 2^{32} ($\sim -\pi$).

Considering the assumptions mentioned above, the value of *position_scale* can be calculated as:

Eqn. 3

$$position\_scale = \frac{2^{32}}{\frac{ENCODER\_PULSES \times 4}{pp}}$$

The term in the numerator represents the full range of the Q1.31 fractional numeric format from –1 to $1 – 2^{-31}$ that is expressed in the MCU as 2^{32}.

The position information is necessary for the calculation of the vector control algorithm. Therefore, the position value is calculated each time the fast (current) control loop is calculated. In the application described in DRM128, this is in the ADC1 conversion complete interrupt service routine (ISR).

5.3 Speed calculation

One of the easiest methods to calculate the speed is a position's derivative with respect to time. In a real-time application running on the MCU, the derivation is substituted by the difference in the value of the mechanical position, captured within a fixed time period. The precision of the speed value calculation

depends on the generated time base, because each small deviation in the time base is amplified by the difference calculation, and so noise is introduced to the speed value. The formula of the angular speed calculation is:

Eqn. 4

$$\omega_{el} = \frac{\theta_K - \theta_{K-1}}{T_{sample}}$$

Where:

ω_{el} [rad.s^{-1}] is the electrical angular speed.

θ_K [rad] is the actual electrical rotor position.

θ_{K-1} [rad] is the electrical rotor position in the previous step.

T_{sample} [s] is the sampling period, the time between when the θ_K and θ_{K-1} values are periodically captured.

In the application described in the DRM128 (running on the Kinetis K40), the time base is generated by the Periodic Interrupt Timer (PIT). The PIT is configured to generate an interrupt every 1 ms. In order to keep the required precision of the time base, it is necessary to assign the highest priority to the PIT interrupt. Even so, a small deviation (up to 6 machine cycles) is introduced to the time base because of the different latency between the interrupt request and entering the interrupt service routine when the interrupt request is generated from the application main or from another interrupt. This deviation is then balanced by implementing the moving average filter on the calculated angular speed value.

The calculation of the speed is performed in the PIT ISR. It is assumed that by implementing the same principle as used for the angular position calculation, the calculation of the electrical angular speed that is executed by the MCU is performed according to the following equation:

Eqn. 5

$$omega\_el\_frac = (theta\_el\_frac_K - theta\_el\_frac_{K-1}) \times omega\_scale$$

Where:

omega_el_frac is the mechanical angular speed in Q1.31 fractional format.

$theta\_el\_frac_K$ is the actual electrical position in Q1.31 fractional format.

$theta\_el\_frac_{K-1}$ is the electrical rotor position in the previous step, in Q1.31 fractional format.

Similar to the position calculation, the scaling is also required for the speed. However, as will be shown below, the determination of the *omega_scale* is not as easy as the calculation of the *position_scale* value.

The term *omega_scale* is calculated as:

Eqn. 6

$$omega\_scale = \frac{2^{32} \times \theta_{max}}{\omega_{max} \times T_{sample}}$$

Where:

θ_{max} [rad] is the maximal position (π).

ω_{max} [rad.s^{-1}] is the maximum electrical angular speed.

T_{sample} [s] is the time between when the position values are captured.

The maximum value of the speed reflects the maximum speed of the motor (considering also field-weakening, if implemented in the application) plus an additional margin (at least 10%) to prevent an overflow during the calculation.

The value of the *omega_scale* calculated according to Equation 6 exceeds the range of a 32-bit number. Therefore, an adjustment has to be made (a right shift by *N* bits to fit to the desired range), and so Equation 5 needs to be modified to:

Eqn. 7

$$omega\_el\_frac = (theta\_el\_frac_K - theta\_el\_frac_{K-1}) \times (omega\_scale \times 2^{-N}) \times 2^N$$

The number of bits (*N*) by which the value of the *omega_scale* needs to be shifted is calculated by the following two equations:

Eqn. 8

$$omega\_scale\_adj = \frac{log\left(\frac{\theta_{max}}{\omega_{max} \times T_{sample}}\right)}{log(2)}$$

Eqn. 9

$$N = omega\_scale\_shift = ceil(omega\_scale\_adj)$$

In the Q1.31 fractional format, the adjusted scale is then defined as:

Eqn. 10

$$omega\_scale\_adj\_frac = 2^{32} \times \left(\frac{\theta_{max}}{\omega_{max} \times T_{sample}}\right) \times 2^{omega\_scale\_shift}$$

After the arrangement, Equation 7 is then calculated in the MCU in two separate equations, for better lucidity with the help of a temporary variable:

Eqn. 11

$$tmp\_frac = (theta\_el\_frac_K - theta\_el\_frac_{K-1}) \times omega\_scale\_adj\_frac$$

Eqn. 12

$$omega\_el\_frac = tmp\_frac >> omega\_scale\_shift$$

6 Conclusion

This application note describes one of the easier possible methods of calculating of the position and speed from quadrature encoder signals. There are also other methods for the position and speed calculation, for example, a method based on the implementation of the angle tracking observer. These are described in other application notes.

7 References

Table 1 lists the documents to which this application note refers.

Table 1. References

Doc order number	Title	Availability
AN3729	Using FlexTimer in ACIM/PMSM Motor Control Applications	www.freescale.com
DRM102	PMSM Vector Control with Single-Shunt Current-Sensing Using MC56F8013/23	
DRM105	PM Sinusoidal Motor Vector Control with Quadrature Encoder	
DRM128	PMSM Vector Control with Quadrature Encoder on Kinetis	
varies	K40 Sub-Family Reference Manual	

8 Acronyms and abbreviated terms

Table 2 contains abbreviated terms used in this document.

Table 2. Acronyms and abbreviated terms

Term	Meaning
ADC	Analog-to-digital converter
FTM	FlexTimer module
ISR	Interrupt service routine
LED	Light emitting diode
PIT	Periodic interrupt timer
PMSM	Permanent magnet synchronous motor
PWM	Pulse-width modulation

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

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)
www.freescale.com/support

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 China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN4381
Rev. 0
12/2011

freescale™