Improving Brushless DC Motor Control for Better Efficiency and Performance in Industrial, White and Brown Goods
Introduction

To compete in a market increasingly driven by efficiency concerns, vendors of industrial machines and household appliances must make best use of high-efficiency brushless DC (BLDC) motors. Today’s microcontrollers offer the compute power and specialized features needed to leverage BLDC motors while minimizing implementation costs. The most useful microcontroller features are specialized timers and on-chip CPU accelerators that eliminate the need for an external digital signal processor (DSP) — even in sensorless control applications.

Taking advantage of these features with BLDC motors helps meet government regulations for power efficiency and qualify for ratings (such as Energy Star) that boost sales. BLDC motors are ideal for meeting these goals with a number of physical and electrical advantages:

- By eliminating brushes, BLDC motors last longer and run quieter than comparable previous-generation brushed DC motors.
- BLDC motors can deliver higher torque than their brushed counterparts.
- In addition to being more reliable, quieter, and stronger, BLDC motors can run faster than brushed DC motors.
- BLDC motors can be sealed (e.g. for dishwasher pumps) or even run directly in liquids.

Thanks to these characteristics, BLDC motors’ market share has been growing significantly in automotive and industrial applications. The latter include electric vehicles, low-end and medium-range industrial drives, office automation equipment, HVAC actuators and fans, vending and cash machines. Many hard disk drives use a BLDC motor to turn the spindle. In the home, BLDC motors are used in washing machines, dishwashers, refrigerators and air conditioners.

BLDC motor control

Controlling a BLDC motor requires information about the rotor position and speed. Getting this information from three Hall sensors (or encoders) on the motor shaft is straightforward. Each Hall sensor outputs a high level for 180° of an electrical rotation, and a low level for the other 180°. The three sensors have a 60° relative offset from each other. This arrangement divides a rotation into six phases (3-bit code). The microcontroller reads the 3-bit code and determines the position of the rotor (Figure 1). Commutation in the right direction is calculated by reading consecutive Hall sensor positions.

Though Hall sensors simplify BLDC motor control, they add cost as well as extra wires to the motor that are susceptible to corrosion and can act as a source of EMI. Hence, many applications are moving away from sensor-based control.

The most popular sensorless method for detecting rotor position is to measure the back-EMF in the non-conducting phase of the BLDC motor winding. Using this measurement to calculate the zero crossing of each phase is CPU intensive and thus requires a higher-performance microcontroller than is needed for the sensor-based method. However, the necessary performance is readily available from today’s microcontrollers and is widely useful in products for improving overall efficiency and implementing product features such as a sophisticated user interface.

Along with higher compute power, the microcontroller needs to provide a versatile timer, referred as a multifunctional timer in Fujitsu microcontrollers. As mentioned earlier, an arithmetic accelerator also proves to be quite useful. The rest of this paper looks at these and other microcontroller features that are useful for implementing the BLDC motor control algorithm outlined in Figure 2 on the following page.
Multifunction timer

The multifunction timer (MFT) in a Fujitsu microcontroller includes a timer, input capture, output compare, a programmable pulse generator (PPG), and a waveform generator (Figure 3). These features are used for controlling BLDC motors as follows:

- Free-running timer — Using an up/up-down counter and control registers, this timer generates the pulse-width modulation (PWM) switching frequency for the isolated-gate bipolar transistors (IGBTs) that actually drive power to the motor.
- Output compare unit — This unit determines the duty cycle of each PWM cycle using compare registers, compare output latch, and compare control registers. An interrupt is generated when the values of the free-running timer and the compare register match.
- Input capture — This function consists of four independent external input pins that feed capture registers and capture control registers. When the function detects an edge of the input signal from the external pin, the value of the free-running timer is stored in the capture register and an interrupt is generated simultaneously. When using the sensor-based BLDC control method, input capture can be used for reading the Hall sensor input or encoder input.
- Waveform generator — This unit outputs PWM signals to drive six IGBT inverter power switches. The waveform generator provides a dead-time timer, which generates the non-overlap signal required to ensure that the positive and negative drive transistors never turn on simultaneously.
Application design using multifunction timer

A washing machine serves as a good example of a product that usually employs a three-phase BLDC motor that is controlled using the Hall sensor method or sensorless control (Figure 4). Both of these methods require use of specialized timers synchronized with other microcontroller peripherals such as A/D converters (ADCs) and a fail-safe unit.

The multifunction timer integrates all the necessary analog and digital functionality for the washing machine. The free-running timer can generate a wide range of carrier frequencies for creating the PWM waveform for motor control, while the MFT’s analog interface handles the control tasks associated with Hall sensors or sensorless measurements.

The analog functions include two independent, high-speed, 12-bit ADC (A/D converters) units (conversion time: 1.2μs) with a dedicated result register for every ADC channel. The ADC measurements are precisely synchronized to the PWM switching events, minimizing noise and allowing two ADC samples to be taken at the same time. The ADCs can be triggered by several MFT events, which can generate various waveform outputs including complementary PWM. The ADC’s flexibility is necessary for advanced motor control algorithms such as sensorless, which rely on precise knowledge of the motor’s phase currents. The ADC can be automatically triggered on compare-clear and zero-detect events of a selectable free-running timer (FRT). Also, separate, buffered, activation-compare registers automatically trigger the ADC at arbitrary points during the PWM period, independently for the up- and down-counting phase of symmetric PWM, if desired. The ADC converter also features a programmable conversion speed as per the operating voltage and frequency, for more design flexibility.

A refrigerator is a good example of an application that typically has two BLDC motors — both compressor and fan motors (Figure 5). The compressor system cools the refrigerator’s freezer compartment, while the fan circulates secondary cooling from the freezer to the refrigerator compartment.

This type of application can use a microcontroller with two MFT channels instead of using two separate microcontrollers. In addition to controlling the motors, the microcontroller also controls the various operational cycles of the refrigerating units. The microcontroller reads sensors such as a water-level sensor, capacity sensor, and temperature sensor and maintains the predetermined levels.

Accelerator for efficient, low-cost BLDC motor design

Sensorless control methods — also known as vector or field-oriented control (FOC) — require the microcontroller to perform complex algorithms and mathematical calculations. The conventional solutions are to use expensive high-performance digital signal processors or a very-high-performance microcontroller that can perform the necessary calculations at real-time speeds.

A third alternative is to use a relatively low-cost microcontroller that includes a built-in accelerator for mathematical calculations. Such a multiply-accumulate (MAC) unit performs the calculations much faster than the microcontroller’s CPU and operates independently while the...
CPU is busy with other system control tasks. In addition to handling motor-related calculations, this capability can be useful for sound processing and image-processing operations such as filtering, Fast Fourier Transforms (FFT), and discrete cosine transforms (DCTs).

In Fujitsu MB91260 series microcontrollers, for example, the accelerator executes fixed-point 16x16+40-bit product-sum operations in a single cycle. The accelerator includes two 16-bit data RAMs (64-word X-RAM and Y-RAM), as well as a 256-word command RAM. When a multiplication and addition (or multiplication) command is executed, the accelerator reads from each RAM area, performs the signed calculation and stores the result in a 40-bit accumulator. The unit can then extract the result by rounding from 40 to 16 bits.

As Figure 6 shows, the accelerator consists of multiple operation units connected by a bus (which appears on the left side of the figure). Along with calculations, the accelerator can perform variable monitoring and execute transfer and conditional branch instructions. In fact, when executing a multiplication and addition (or multiply) command, the accelerator can perform a transfer operation at the same time. Performing this transfer and the calculation in tandem makes it easy to perform delayed data processing with a digital filter.

In the case of BLDC motor control, the accelerator calculates variables such as new position estimation, angular velocity, motor rotation, and other correction and control calculations. The control algorithm is divided between the CPU and accelerator to achieve the real-time performance required for sensorless control (Figure 7) — without the expense of an external DSP.

```
unsigned short frt_count0 = 0x0000;
unsigned short frt_count1 = 0x0000;
int x, y, z1, z2, a, b, rot
void FRT_initialize(void)
{
    IO_ICR[27].byte = 0x10;
    IO_PORT3.IO_EPPR34.byte = 0x00;
    IO_FRTSEL.byte = 0x00;
    IO_TCCSH0.byte = 0x01;
    IO_TCCSL0.byte = 0x50;
    IO_CPCLR0 = 0x00FFFFFF;
    IO_TCDT0 = 0x00000000;
    a = cos(rot);
    b = sin(rot);
    z1 = a * x + b * y;
    
    Program for position sensing
    a = cos(rot);
    b = sin(rot);
    z1 = a * x + b * y;
```

Figure 6 — Multiply-accumulate accelerator in Fujitsu MB91260 series microcontrollers

Figure 7 — Dividing motor control between a microcontroller’s CPU and accelerator
Key features for BLDC motor control

Along with the multifunction timer and accelerator for motor control, Fujitsu microcontrollers include other special features that are appropriate for controlling BLDC motors using either sensor-based and sensorless control methods.

- **Hardware Watchdog** — The IEC60730 standard proposes "independent time-slot monitoring" to prevent any CPU malfunction. The watchdog timer, which is embedded in Fujitsu MCUs, takes care of this function. However, IEC60730 compliance requires that the watchdog be independent from other resources. For this reason, the Fujitsu motor MCU has two watchdogs: a regular software watchdog running on the main CPU clock, and a second hardware watchdog, using an independent internal oscillator. This ensures that at least one watchdog will be active in case of a crystal failure and can switch the main application to the internal CR oscillator.

- **Multi-Pulse Generator (MPG)** — The MPG is an alternative to the multifunction timer and can be used for controlling a second motor. The MPG supports operation modes suitable for inverter control, especially for three-phase BLDC motor in applications such as fans and compressors.

- **Dead-Time Generation** — To avoid a short between the motor windings, the MCU must insert a dead time between the PWM outputs. The Fujitsu MCU dead timer provides the flexibility to set a common dead time for all three motor phases or allows the setting of separate dead times for each of the upper/lower phases. Figure 8 shows an example of dead timer operation for controlling the upper and lower arm of a motor.

- **Over-current protection function** — The dead-time timer interrupt (DTTI) fault function in the MCU's motor peripheral allows for enabling/disabling driver signal outputs in case of emergency. Either a hardware or software trigger can inactivate the output. The output level can be fixed by three levels (high, low, or high impedance) when the DTTI function is enabled.

![Figure 8 — Dead time in the motor control sequence](image)
IEC60730 Self Test Library

To ensure safety, electrical control equipment for household and similar use must comply with the IEC60730 standard. This standard places a number of requirements on the MCU peripherals that support safety compliance and on code execution. Table 1 shows the basic self-test code requirement from the IEC60730 standard.

The IEC60730 test library includes the check for the CPU, system registers, interrupts handling, execution and monitoring, clock frequency monitoring, periodic memory test such as RAM and Flash, and other peripherals, such as GPIO and the communication interface.

To simplify application development, Fujitsu offers a self-test library whose functions meet the requirements of IEC60730. This self-test library can be used across all Fujitsu MCU platforms with minimal modification. Using this self-test library, users can rapidly develop control functions that comply with Software Class B.

### Table 1 — IEC60730 Safety requirements

<table>
<thead>
<tr>
<th>Components</th>
<th>Measures Applied</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Storage</td>
<td>Periodic self test using walkpat memory test</td>
</tr>
<tr>
<td></td>
<td>-After writing test pattern in memory area under test, invert bits of memory cells</td>
</tr>
<tr>
<td></td>
<td>one by one and check remaining data</td>
</tr>
<tr>
<td></td>
<td>Periodic CRC check</td>
</tr>
<tr>
<td></td>
<td>Word protection with redundancy bit</td>
</tr>
<tr>
<td></td>
<td>Redundant memory with software comparison</td>
</tr>
<tr>
<td>Code Execution</td>
<td>Periodic self test for equivalent class test</td>
</tr>
<tr>
<td></td>
<td>-Classify similar instructions in groups and test with input data</td>
</tr>
<tr>
<td></td>
<td>-e.g. move instruction, arithmetic instructions, …</td>
</tr>
<tr>
<td></td>
<td>Independent time slot and logical monitoring</td>
</tr>
<tr>
<td>Communication</td>
<td>Word protection with redundancy bit</td>
</tr>
<tr>
<td></td>
<td>Scheduled transmission for timing issue</td>
</tr>
<tr>
<td>Interrupt</td>
<td>Independent time slot and logical monitoring</td>
</tr>
<tr>
<td></td>
<td>-Check if no interrupt or too many interrupts</td>
</tr>
<tr>
<td>Clock</td>
<td>Frequency monitoring using independent fixed clock</td>
</tr>
<tr>
<td>IO Port</td>
<td>Input Comparison</td>
</tr>
<tr>
<td></td>
<td>Output verification</td>
</tr>
</tbody>
</table>

### Conclusion

BLDC motors are well suited to meet the efficiency and performance requirements of today’s household products and other motor applications. Whether the application employs position sensors or a sensorless control method, microcontrollers offer the ideal solution for controlling BLDC motors — especially with the use of versatile timer and safety-related peripherals. For sensorless control, on-chip arithmetic accelerators enable fine-tuned performance without the expense of an external DSP.

### For more information

For more information about Fujitsu microcontrollers, please go to [http://us.fujitsu.com/micro/mcu](http://us.fujitsu.com/micro/mcu), or address e-mail to inquiry@fma.fujitsu.com.