STM32 for Motor Control Applications

Features and benefits
Plan

- PWM generation
- Speed / position feedback
- Multi timer configuration
- Analog to Digital converter
- Current sensing and ADC synchronization
- Field Oriented Control
- Motor Control Tools
Advanced timer Features overview

STM32 for Motor Control

Features and benefits

STM32 Releasing your creativity
High-resolution PWM generation

- Motor Control Timer clock
  - Can be 2x the APB bus frequency
  - Max input clock is 72MHz to provide 13.8ns edge resolution (12-bit @16kHz edge-aligned PWM)

- Edge or center-aligned patterns

- Double-update mode (cf next slide)
  - No loss of resolution in center-aligned mode
  - Done thanks to an additional interrupt per PWM cycle or DMA transfers
Double Update Mode

An Update event (U) during the Overflow of the PWM counter improves duty cycle resolution.
PWM main Interrupt Service Routine

- So-called U (Update) event
  - Synchronously transfers all preload into active registers
    - 3 (4) compares for duty cycles
      - Preload mechanism can be disabled if needed
    - 1 Auto Reload for PWM switching period
      - allows changing on-the-fly the PWM frequency while maintaining duty cycles
    - PWM clock pre scaler

- Adjustable U event rate
  - programmable through a 8-bit repetition counter
  - Allows to choose Overflow/Underflow or both for update
Repetition Counter

- **Double update REP=0**
- **Single update OVF REP=1**
- **Single update UDF REP=1**
- **REP=2**
- **REP=3**

PWM counter
Other interrupts and DMA

- Other interrupt sources available on PWM timer
- Each Compare match (up or down counting selectable) or capture
- Trigger events
- Emergency Stop
- Some events are also mapped on the DMA controller
PWM’s DMA burst transfer

- Allows to update several registers with a single DMA event
- Efficient use of DMA (a single stream is required)

<table>
<thead>
<tr>
<th>RAM</th>
<th>Registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>OC1t0</td>
<td>OC1</td>
</tr>
<tr>
<td>OC2t0</td>
<td>OC2</td>
</tr>
<tr>
<td>OC3t0</td>
<td>OC3</td>
</tr>
<tr>
<td>OC1t1</td>
<td>Virtual Register</td>
</tr>
<tr>
<td>OC2t1</td>
<td></td>
</tr>
<tr>
<td>OC3t1</td>
<td></td>
</tr>
<tr>
<td>OC1t2</td>
<td></td>
</tr>
<tr>
<td>OC2t2</td>
<td></td>
</tr>
<tr>
<td>OC3t2</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
PWM outputs management

- Programmable hardware deadtime generation
- 8-bit register with 13.8ns max resolution at 72MHz (from 0 to 14µs, non-linear)

![Diagram of PWM outputs management]

- Individually selectable polarity selection
- Dedicated emergency stop input
  - Shuts down the 6 PWM outputs and issues an interrupt
  - **Asynchronous** operation (operates without clock source)
### Versatile PWM redirection circuitry 1/2

#### Table 40: Output Control of Complementary OCx and OCxN Channels

<table>
<thead>
<tr>
<th>Control Bits</th>
<th>Output State</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOE bit</td>
<td>OCx output state</td>
</tr>
<tr>
<td>OSS bit</td>
<td>OSSR bit</td>
</tr>
<tr>
<td>x</td>
<td>0</td>
</tr>
<tr>
<td>x</td>
<td>0</td>
</tr>
<tr>
<td>x</td>
<td>0</td>
</tr>
<tr>
<td>x</td>
<td>0</td>
</tr>
<tr>
<td>x</td>
<td>1</td>
</tr>
<tr>
<td>x</td>
<td>1</td>
</tr>
<tr>
<td>x</td>
<td>1</td>
</tr>
<tr>
<td>x</td>
<td>1</td>
</tr>
</tbody>
</table>

- **PWM timer used as a GP timer**
- **Motor Control (sinewave)**
- **Motor Control (6-steps)**
- **Motor Control (sinewave)**
- **Outputs disconnected from I/O ports**
- **All PWMs OFF (low Z for safe stop)**

---

**STM32 for Motor Control**

**Features and benefits**

**STM32 Releasing your creativity**
Versatile PWM redirection circuitry 2/2

Example: 6-steps (or block commutated) drives

<table>
<thead>
<tr>
<th>Step</th>
<th>High</th>
<th>Low</th>
<th>OC1</th>
<th>OC1N</th>
<th>OC2</th>
<th>OC2N</th>
<th>OC3</th>
<th>OC3N</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>T1</td>
<td>T4</td>
<td>oc1ref</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>T1</td>
<td>T6</td>
<td>oc1ref</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>T3</td>
<td>T6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>T3</td>
<td>T2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>T5</td>
<td>T2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>T5</td>
<td>T4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Phase current

STM32 for Motor Control

Features and benefits
A break event can be generated by:
- The BRK input which has a programmable polarity and an enable bit BKE
- The Clock Security System

When a break occurs:
- The MOE bit (Main Output Enable) is cleared
- The break status flag is set and an interrupt request can be generated
- Each output channel is driven with the level programmed in the OISx bit
  - For instance all low side switches ON for PM motors in field weakening mode

Break applications:
- If the AOE is Reset, the MOE remains low until you write it to ‘1’ again
  - Normally used for security with break input connected to an alarm feedback from power stage, thermal sensors or any security components.
- If the AOE (Automatic Output Enable) bit is set, the MOE bit is automatically set again at the next update event UEV
  - Typically be used for cycle-by-cycle current regulation
  - Current regulation can also be performed using External trigger input (ETR)
Smoke inhibit protections

- Safety critical registers can be “locked”, to prevent power stage damages (software run-away,...)
  - Dead time, PWM outputs polarity, emergency input enable,...

- All target registers are read/write until lock activation (and then read-only if protected)
  - Once the two lock bits are written, they cannot be modified until next MCU reset (write-once bits)
  - 4 lock levels offer full灵活性 depending on the application (e.g. 6-steps vs sine)

- GPIO configuration can be locked to avoid having the PWM alternate function outputs reprogrammed as standard outputs
Debug feature

Motor control applications are usually tricky to debug using breakpoints:
- Standard breakpoints may damage the power stage
- Closed loop systems can hardly be stopped and re-started

A configuration bit allows to program the behavior of the PWM timer upon breakpoint match:
- Normal mode: the timer continues to operate normally
  - May be dangerous in some case since a constant duty cycle is applied to the inverter (interrupts not serviced)
- Safe mode: the timer is frozen and PWM outputs are shut down
  - Safe state for the inverter. The timer can still be re-started from where it stops.
Plan

- PWM generation
- Speed / position feedback
- Multi timer configuration
- Analog to Digital converter
- Current sensing and ADC synchronization
- Field Oriented Control
- Motor Control Tools
Speed Feedback

- Handled by the general purpose timers, with dedicated modes
  - These functions are available on all timers

- Hall sensors
  - Hall Sensor interface (XOR’ed inputs)

- Encoder
  - Encoder modes 1, 2 & 3 (2x, 4x)

- Tacho feedback
  - Clear on capture to measure exact period
TIM Block Diagram in encoder mode
Interfacing a TIM timer with an encoder

ENCODERS AND STM32 CONNECTION EXAMPLE:
- An incremental encoder can be connected directly to the MCU without external interface logic.
- The third encoder output which indicates the mechanical zero position (Z or index), may be connected to an external interrupt and trigger a counter reset.
Key encoder features

Programmable counting rate
- x4: normal mode, all edges active
  - a 1000 lines encoder will give 4000 counts per revolution
- x2: counts on input A (or B) only, but direction still determined with A and B
- “velocity mode”: encoder clock can be further prescaled if needed

Programmable encoder resolution
- When programming the autoreload register with the number of counts per revolution, the counter register directly holds the angle or the position
  - No need to do the difference vs previous counter value
- If set to 0xFFFF, can be made compatible with previous designs using a free-running counter

Possibility to generate one/multiple interrupts per revolution:
- once every 360°
- once 60°, 90°,... (depending on autoreload register setting)
Hall sensor Interface
Plan

- PWM generation
- Speed / position feedback
- Multi timer configuration
- Analog to Digital converter
- Current sensing and ADC synchronization
- Field Oriented Control
- Motor Control Tools
Timer Link system

- The three general purpose and the advanced timers are linked together and can be synchronized or chained, thanks to a Trigger output and several selectable trigger inputs.

- For TIM2:0, the input pins (TI1 and TI2) can also be used as triggers.
Synchronization Mode Configuration

- When in master mode, the trigger output can be:
  - Counter reset
  - Counter enable
  - Update event
  - Output Compare signal
- When configured as slave, the timer can work in the following modes:
  - Triggered
  - Gated
  - Reset
  - External clock
Example 1/3: chained timers

Cascade mode (for instance, chained time bases)
Examples 2/3: synchronized start

One Master several slaves
Examples 3/3: block commutation

A TIM timer handles Hall feedback and triggers an advanced timer for step commutation

**MASTER (TIM)**

**SLAVE (ADVANCED TIM)**

---

STM32 for Motor Control  
Features and benefits  
STM32 Releasing your creativity
Plan

- PWM generation
- Speed / position feedback
- Multi timer configuration
- Analog to Digital converter
- Current sensing and ADC synchronization
- Field Oriented Control
- Motor Control Tools
ADC Features (1/3)

- ADC conversion rate 1 MHz and 12-bit resolution
  - 107ns min sampling time
- ADC input range: $0 \leq \text{VIN} \leq VREF+$ (VREF+ only on LQFP100 package)
- Up to 18 multiplexed channels:
  - 16 external channels
  - 2 internal channels: Temperature sensor and voltage reference
- Interrupt generation
  - End of Conversion, End of Injected conversion, Analog watchdog
- DMA capability
  - On ADC1, but possibility to transfer also ADC2 result if sync’d
- Channels conversion groups:
  - Up to 16 channels regular group
  - Up to 4 channels injected group
ADC Injected Conversion

- Regular Scan mode
- Scan mode with injected conversion mode

Diagram:

- First channel Conversion
- Second channel Conversion
- Last channel Conversion
- First injected channel Conversion
- Second injected channel Conversion
- Last injected channel Conversion

Trigger → Interrupt
ADC Features (2/3)

- Analog Watchdog (1 channel or all regular or all injected)

- Sequencer-based scan mode
  - Any channel, any order (e.g. Ch3, Ch2, Ch11, Ch11, Ch3)
  - up to 16 regular conversion (transferred by DMA)
  - up to 4 injected conversion stored in internal registers

- Multiple trigger sources for both regular and injected conversion
  - Each group can be started by 6 events from the 4 timers (compare, over/underflow)
  - External event and software trig also available
ADC Features (3/3)

- Left or right Data alignment with inbuilt data coherency

- 4 offset compensation registers
  - Compensates external conditioning components offsets (such as Operational Amplifiers). Provides signed results if needed.

- Channel-by-channel programmable sampling time to be able to convert signals with various impedances
  - From 1µs (for Rin < 1.2KΩ) to 18µs (Rin < 350KΩ), 8 values
  - External voltage follower not mandatory when converting at 1MSps
ADC dual modes (1/2)

- Available in devices with two ADCs (Performance line)
- ADC1 and ADC 2 can work independently or coupled (master/slave)
- 8 ADC dual modes
ADC dual modes example (2/2)

Injected simultaneous mode on 4 injected channels

Sampling

Conversion

End of Injected Conversion on ADC1 and ADC2

Trigger for injected channels

Fast Interleaved mode on 1 regular channel in continuous conversion mode

Up to 2 MSps continuous data throughput (DMA-based)

Trigger for regular channels

7 ADCCLK cycles

STM32 for Motor Control Features and benefits

STM32 Releasing your creativity
Plan

- PWM generation
- Speed / position feedback
- Multi timer configuration
- Analog to Digital converter
- Current sensing and ADC synchronization
- Field Oriented Control
- Motor Control Tools
Isolated Current transducers 1/2

- Isolated / floating sensors
  - PWM synchronization is possible when all low sides and / or all high sides are ON ("crest" or "valley")
  - Two sensors are enough (currents are always readable)
  - The current can be read twice per PWM period
Isolated Current transducers 2/2

- Counter
- Compare 1
- Compare 2
- Low side 1
- Low side 2
- Phase current
- Average current
- ADC Sampling
Shunt based current sensing 1/2

- Using 3 shunts
- PWM synchronization mandatory
- Need to sample when all low side switches are ON, in the middle of PWM period (“crest”)
- Current cannot be read during 33% of sine period, but two phases at least are readable
- The third one can be reconstructed ($I_a = -I_b - I_c$)
Shunt based current sensing 2/2

Counter
Compare 1
Compare 2
Low side 1
Low side 2
Phase current
Average current
ADC input
(Phase 1)
ADC Sampling

0V

STM32 for Motor Control
ADC synchronization in STM32

- Done thanks to a synchronization unit embedded in the PWM timer.
- 2 options available:
  - Direct synchronization on PWM crest, valley, or both.
  - Delayed synchronization with the 4th Compare channel
- The ADC results can be then processed with an end of conversion interrupt or transferred by DMA.
Direct synchronization

- The PWM timer “update” signal triggers
- Simultaneous injected conversions on both ADCs
- No error due to sequential phase sampling
Plan

- PWM generation
- Speed / position feedback
- Multi timer configuration
- Analog to Digital converter
- Current sensing and ADC synchronization
- Field Oriented Control
- Motor Control Tools
Field oriented control (FOC): a quick overview

- Mathematical technique utilized for achieving decoupled control of the flux and torque in a three-phase machine (ACIM or PMSM)
- Stator current is decomposed into:
  - the magnetizing current $I_{ds}$, producing a magnetic field algebraically added to the one of the rotor
  - The quadrature current $I_{qs}$ which controls torque just like the armature current does in the DC motor
- Benefits (vs scalar control)
  - Precise and responsive speed control when the load changes
  - Optimized efficiency even during transient operation
  - Allows position control (through instantaneous torque control)
FOC in torque control (Open loop)

AC Mains

3 phase inverter

PMSM Motor ~

\[ i_{qs}^{*} \]

\[ i_{ds}^{*} \]

\[ v_{qs} \]

\[ v_{ds} \]

\[ v_{a'b'c} \]

\[ \theta_{r el} \]

\[ i_{qs} \]

\[ i_{ds} \]

\[ i_{abc} \]

\[ i_{a'b'} \]

\[ \theta_{r el} \]

PARK

REVERSE PARK & circle limitation

CALC SVPWM

CLARKE

CURRENT READING

DC Domain

AC Domain

STM32 for Motor Control

Features and benefits

STM32 Releasing your creativity
FOC in speed control (Closed loop)

AC Mains

3 phase inverter

PMSM Motor ~

v_{a,b,c}

v_{qs}

v_{ds}

v_{a'\beta'}

θ_{r el}

i_{qs}

i_{ds}

i_{abc}

i_{\alpha\beta}

ω_{r}

ω_{r}^*

i_{qs}^*

i_{ds}^*

Torque and flux controller

PID

REVERSE PARK & circle limitation

Calc SVPWM

Clarke

Current reading

Rotator speed/position feedback

DC Domain

AC Domain

STM32 for Motor Control

Features and benefits

STM32 Releasing your creativity
FOC computational requirements

- The complete vector control algorithm must be continuously recomputed, at a rate of between 1 and 10kHz (1ms down to 100µs loop time, depending on the final application).
- 16-bit is the minimum required accuracy for the main control variables, with a need for 32-bit calculations.
- The algorithm intensively requires math computations (trigonometric functions, PID regulator, real-time flux and torque estimation based on motor parameters).
A sensorless control method: the B-emf observer

In control theory a system is **observable**, if it is possible to fully reconstruct the system state from its output measurements.

The **state observer** is a system that provides an estimation of the internal state of the observed system given its input and output measurements.
Luenberger Observer for PMSM FOC

\[ u (v_q, v_d) \rightarrow B \rightarrow f \rightarrow C \rightarrow y (i_q, i_d) \]

\[ x (i_{qs}, i_{ds}, e_Q, e_D) \]

\[ \hat{x} (i_D, i_Q, e_Q, e_D) \rightarrow \hat{y} (i_q, i_d) \]
Getting the rotor angle

Once $\hat{e}_\alpha$ and $\hat{e}_\beta$ have been determined, rotor angle position could be simply computed by formula:

$$\theta_r = \arctg \frac{\hat{e}_\alpha}{\hat{e}_\beta}$$

And thus the speed could be computed as derivative of rotor angle

Rotor angle can also be estimated with a software PLL solution to avoid arctg function (low accuracy at 90°) and derivative (noise sensitive)
FOC algorithm execution time

With Cortex-M3 running at 72 MHz from embedded flash, fully code in C (optimized for speed)

For reference, Sensored FOC is ~17\(\mu\)s (vs 25\(\mu\)s with ARM7-based STR750, ie ~30% gain)
FOC software library memory footprint

Thanks to the high density of Thumb2 instruction set, the preliminary overall size of PMSM FOC software library is 26Kb.

Excluding only the LCD and Joystick management the overall code size falls to 16Kb.

Preliminary, Code optimized for speed.
Practical results

- 700W AC induction motor, sensored field oriented control
Plan

- PWM generation
- Speed / position feedback
- Multi timer configuration
- Analog to Digital converter
- Current sensing and ADC synchronization
- Field Oriented Control
- Motor Control Tools
STM32 Software library for PMSM FOC (1/2)

- Supported current sensing methodologies:
  - Isolated Current sensing
  - Three shunt resistors
  - Single shunt resistor (future development)

- Supported rotor position feedback:
  - Encoder
  - Hall sensors:
    - 60° and 120° placement
  - Sensorless
STM32 Software library for PMSM FOC (2/2)

- Developers' support
  - Progressive system development for guiding users during their own development
  - DAC functionality for tracing the most important software variables
  - User interface for real time tuning of PIDs and observer gains (via LCD and joystick)

- MISRA 2004 compliant

- Free of charge and open source

- Demonstration kit available e/o Oct 07
Progressive system development

- Guide user in his development, gradually take him towards sensorless solution
- Allows to avoid the inclusions of the code lines implementing not used functionalities

```c
#define ICS_SENSORS  // Current sensing by ICS (Isolated current sensors)
#define THREE_SHUNT  // Current sensing by Three Shunt resistors
#define ENCODER  // Position sensing by Incremental encoder
#define HALL_SENSORS  // Speed sensing by Hall sensors
#define NO_SPEED_SENSORS  // No speed sensors
#define VIEW_HALL_FEEDBACK
#define VIEW_ENCODER_FEEDBACK
#define Id_Iq_DIFFERENTIAL_TERM_ENABLED  // PI + Differential term for Id & Iq regulation
#define SPEED_DIFFERENTIAL_TERM_ENABLED  // PI + Differential term for speed regulation
#define FLUX_TORQUE_PIDs_TUNING
#define OBSERVER_GAIN_TUNING
#define DAC_FUNCTIONALITY
```
**DAC functionality**

- Implemented using two out of the four TIM3 output compare channels
- Allow the simultaneous monitoring of up to two software variables selectable in real-time using a dedicated menu
- Can be disabled simply commenting one code line
User interface

- Real time tuning of torque, flux and speed PID.s
- Observer gains tuning (in case of sensorless)
- Variation of target speed (speed control) or target torque and flux (torque control)
- Bus voltage and power stage temperature monitoring
- Selection of variables to put on output for DAC functionality implementation

### STM32 Motor Control
PMSM FOC ver 0.2

<table>
<thead>
<tr>
<th>Closed Loop</th>
<th>Speed (rpm)</th>
<th>Ref</th>
<th>Meas</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>02500</td>
<td></td>
<td>00000</td>
</tr>
</tbody>
</table>

Speed control mode

Move $\uparrow\downarrow$ Change
STM32 demonstration kit

- Three-phase power stage with shunt-based current reading
- Complete source files software libraries for 3-PH Induction and PMSM motors provided
- Brushless PM Motor with encoder included
STM32-MCKIT (alpha version)

- Brushless PM Motor with encoder
- Three-phase Power stage with shunt-based current reading
- Opto-isolated JTAG
- Evaluation board (final STM32 version has a color LCD)