

U-FOC API. Версія 1.1

Загальна інформація

API дозволяє програмно керувати мотором, і налаштуваннями системи. Використовуючи API можна легко створити гнучку систему керування мотором, яка буде відповідати вимогам Вашого проєкту.

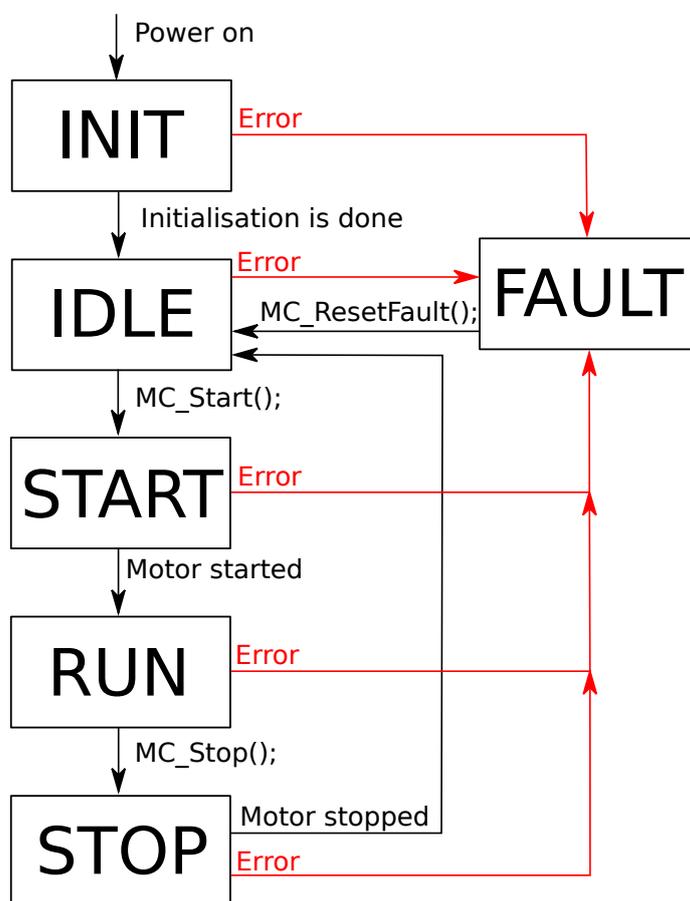
Керування мотором

uint8_t MC_GetState(void);

Повертає поточний стан системи:

MC_STATE_INIT	- Ініціалізація	int(0)
MC_STATE_IDLE	- Очікування, мотор зупинено	int(1)
MC_STATE_START	- Процес старту мотора	int(2)
MC_STATE_RUN	- Мотор обертається	int(3)
MC_STATE_STOP	- Процес зупинки мотора	int(4)
MC_STATE_FAULT	- Зупинка через помилку у роботі.	int(5)

Життєвий цикл статусів зображено на малюнку:



uint8_t MC_GetFault(void);

Повертає помилку яка виникла під час роботи, коли стан MC_STATE_FAULT:

MC_NO_FAULTS	- No errors	int(0)
MC_FOC_DURATION	- no used yet (reserved)	int(1)
MC_OVER_VOLT	- Over voltage	int(2)
MC_UNDER_VOLT	- Low voltage	int(3)
MC_OVER_TEMP	- High temperature	int(4)
MC_START_UP	- no used yet (reserved)	int(5)
MC_SPEED_FDBK	- Rotor position detection error	int(6)
MC_BREAK_IN	- Overcurrent protection tripped	int(7)
MC_SW_ERROR	- no used yet (reserved)	int(8)

void MC_ResetFault(void);

Скидає помилку. Намагається встановити FAULTS у стан MC_NO_FAULTS.

void MC_SetMode(uint8_t mode);

Встановлює режим роботи:

MC_MODE_TORQUE	- Torque mode	int(0)
MC_MODE_SPEED	- Velocity mode	int(1)

uint8_t MC_GetMode(void);

Повертає поточний режим роботи:

MC_MODE_TORQUE	- Torque mode	int(0)
MC_MODE_SPEED	- Velocity mode	int(1)

void MC_Start(void);

Запускає мотор та режим стабілізації заданих показників у заданому режимі.

void MC_Stop(void);

Зупиняє мотор.

int32_t MC_GetTorqueTask();

Повертає заданий раніше момент.

void MC_SetTorque(int32_t torque);

Здає момент мотора. Не запускає мотор, не перемикає режим.

int32_t MC_GetSpeedTask();

Повертає задану раніше швидкість обертання мотора.

void MC_SetSpeed(int32_t speed);

Задає оберти мотора у обертах за хвилину. Не запускає мотор, не перемикає режим.

int32_t MC_GetRPM();

Повертає оберти мотора у обертах за хвилину.

void MC_SetTorqueRamp(int32_t torque, int32_t time);

Запускає зміну моменту від поточного до вказаного за вказаний час у мілісекундах. Встановлює режим роботи MC_MODE_TORQUE.

void MC_SetSpeedRamp(int32_t speed, int32_t time);

Запускає зміну обертів від поточного до вказаного за вказаний час у мілісекундах. Встановлює режим роботи MC_MODE_SPEED.

void MC_StopRamp();

Зупиняє роботу MC_SetTorqueRamp, MC_SetSpeedRamp.

void MC_ProtocolTimeoutEnable();

Вмикає відключення мотора, якщо не приходять пакти протоколу *U-FOC Protocol* по шині CAN або по порту UART. Ініціалізує вимикання двигуна на випадок обриву зв'язку з керівним приладом.

void MC_ProtocolTimeoutDisable();

Вимикає відстеження пактів протоколу *U-FOC Protocol* по шині CAN або по порту UART. Цю функцію необхідно виконати перед використанням API, інакше двигун буде зупинятися через те, що не надходять пакети по шині CAN або по порту UART.

void MC_SetProtocolTimeout(uint16_t mSec);

Встановлює час таймаута у мілісекундах для відстеження пактів протоколу *U-FOC Protocol* по шині CAN або по порту UART.

void MC_SetCurrentProtect(uint16_t amps);

Встановлює рівень спрацювання захисту по струму в амперах. Ця функція доступна не для всіх плат.

void MC_PID_Ia_Set (float32_t Kp, float32_t Ki, float32_t Kd);

Встановлює коефіцієнти PID регулятора кута вектора струма.

void MC_PID_Ir_Set (float32_t Kp, float32_t Ki, float32_t Kd);

Встановлює коефіцієнти PID регулятора радіуса вектора струма.

void MC_PID_Speed_Set (float32_t Kp, float32_t Ki, float32_t Kd);

Встановлює коефіцієнти PID регулятора швидкості обертання двигуна.

Доступ до налаштувань

Для прямого доступу до структури налаштувань у кодї програми треба оголосити змінну:

```
extern volatile SettingsStruct Settings;
```

Після чого можна змінювати налаштування, наприклад таким чином.

```
Settings.PID_SpeedKp = 0.25;  
Settings.PID_SpeedKi = 0.1;
```

Структура SettingsStruct:

```
typedef struct  
{  
    uint8_t DevID;  
  
    uint8_t RotorPolePairs;  
    int16_t HallAngleOffset;  
    int16_t VoltageAngleOffset;  
  
    int8_t Revers;  
  
    uint16_t RpmMax;  
    uint16_t VoltageMax;  
    uint16_t VoltageMin;  
    uint16_t BrakeVoltageOn;  
    uint16_t BrakeVoltageOff;  
    int16_t TemperatureMax;
```

```
uint16_t CurrentMax;
uint16_t CurrentBRK;
uint16_t TorqueMax;
uint16_t PowerMax;

uint16_t FilterIa;
uint16_t FilterIr;
uint16_t FilterSpeed;

float32_t PID_IaKp;
float32_t PID_IaKi;

float32_t PID_IrKp;
float32_t PID_IrKi;

float32_t PID_SpeedKp;
float32_t PID_SpeedKi;
float32_t PID_SpeedKd;

uint32_t ADC0_0;
uint32_t ADC0_1;
uint32_t ADC0_2;
int32_t ADCNoise;

uint16_t ProtocolTimeout;
uint8_t ProtocolTimeoutEnable;

uint8_t ModulationType;
} SettingsStruct;
```

Опис налаштувань:

DevID — Номер пристрою для протоколу *U-FOC Protocol*

RotorPolePairs — Кількість пар полюсів електродвигуна

HallAngleOffset — Зсув датчиків Холла. Умовні одиниці 1 — 1,875 градуси.

VoltageAngleOffset — Зсув датчиків вектора напруги у градусах. Використовується коли коефіцієнти ПІД регулятора куту струму встановлені в 0 і є потреба ручного коригування.

Revers — **MC_ROTATION_POSITIVE** / **MC_ROTATION_REVERSE** Вказує напрямок обертання валу мотора за замовчуванням (пряме чи зворотне).

RpmMax — Максимальні оберти мотора на валу, оберти/хвилину.

VoltageMax — Максимальна напруга живлення

VoltageMin — Мінімальна напруга живлення.

BrakeVoltageOn — Напруга при якій вмикається гальмівний резистор.

BrakeVoltageOff — Напруга при якій вимикається гальмівний резистор.

TemperatureMax — Максимальна температура.

CurrentMax — Максимальний струм.

CurrentBRK — Струм при якому спрацює захист від перенавантаження.

TorqueMax — Максимальний момент у умовних одиницях. Вказувати не потрібно, вираховується функцією **MC_Settings_Recalculate()**

PowerMax — Максимальна потужність.

FilterIa — Коефіцієнт фільтрації кута вектору струму

FilterIr — Коефіцієнт фільтрації модуля вектору струму

FilterSpeed — Коефіцієнт фільтрації швидкості обертання

PID_IaKp — Пропорційний коефіцієнт PI регулятора кута вектору струму

PID_IaKi — Інтеграційні коефіцієнт PI регулятора кута вектору струму

PID_IrKp — Пропорційний коефіцієнт PI регулятора радіуса вектору струму

PID_IrKi — Інтеграційні коефіцієнт PI регулятора радіуса вектору струму

PID_SpeedKp — Пропорційний коефіцієнт PI регулятора швидкості обертання

PID_SpeedKi — Інтеграційні коефіцієнт PI регулятора швидкості обертання

ADC0_0 — Показники ADC датчика струму при нульовому струмі.

ADC0_1 — Показники ADC датчика струму при нульовому струмі.

ADC0_2 — Показники ADC датчика струму при нульовому струмі.

ADCNoise — Шум ADC датчиків струму.

ProtocolTimeout — Час очікування пакетів по протоколу *U-FOC Protocol* (CAN-шина чи USART-порт).

ProtocolTimeoutEnable — Вмикання контролю надходження пакетів по протоколу *U-FOC Protocol* (CAN-шина чи USART-порт). Якщо увімкнено, і пакетів не надходило вказаний у ProtocolTimeout час, мотор вимикається.

ModulationType — тип модуляції. Доступні SPWM, THIPWM, SVPWM, DPWMMIN.

Коди:

1 - SPWM

2 - THIPWM

3 - SVPWM

4 - DPWMMIN

void MC_SaveSettings(void);

Записує налаштування у енергонезалежну пам'ять. **УВАГА!** Використовуйте цю функцію лише коли мотор зупинено!

void MC_ResetSettings(void);

Скидає налаштування до початкових. Але не записує у енергонезалежну пам'ять. **УВАГА!** Використовуйте цю функцію лише коли мотор зупинено!

void MC_LoadSettings(void);

Завантажує налаштування з енергонезалежної пам'яті. **УВАГА!** Використовуйте цю функцію лише коли мотор зупинено!