STM32 БЩПС Двигун

Посібник бібліотеки

Ця бібліотека була розроблена та випробувана на STM32G431C6 (плата B-G431B-ESC1), який керує двигуном 42BLDC-24 (24 В, 4000 об/хв, із трьома датчиками Холла). Через пошкодження з'єднання на платі один із сигналів датчика Холла став недоступним. Замість заміни електропривода, бібліотеку було доповнено для витримування відмови одного датчика Холла, що дозволило зберегти безперервну роботу.
Завдяки цій відмовостійкій побудові, ви можете керувати своїм безщітковим двигуном постійного струму, використовуючи:

- Шестиступеневе жорстке перемикання
- Шестиступеневе м'яке перемикання

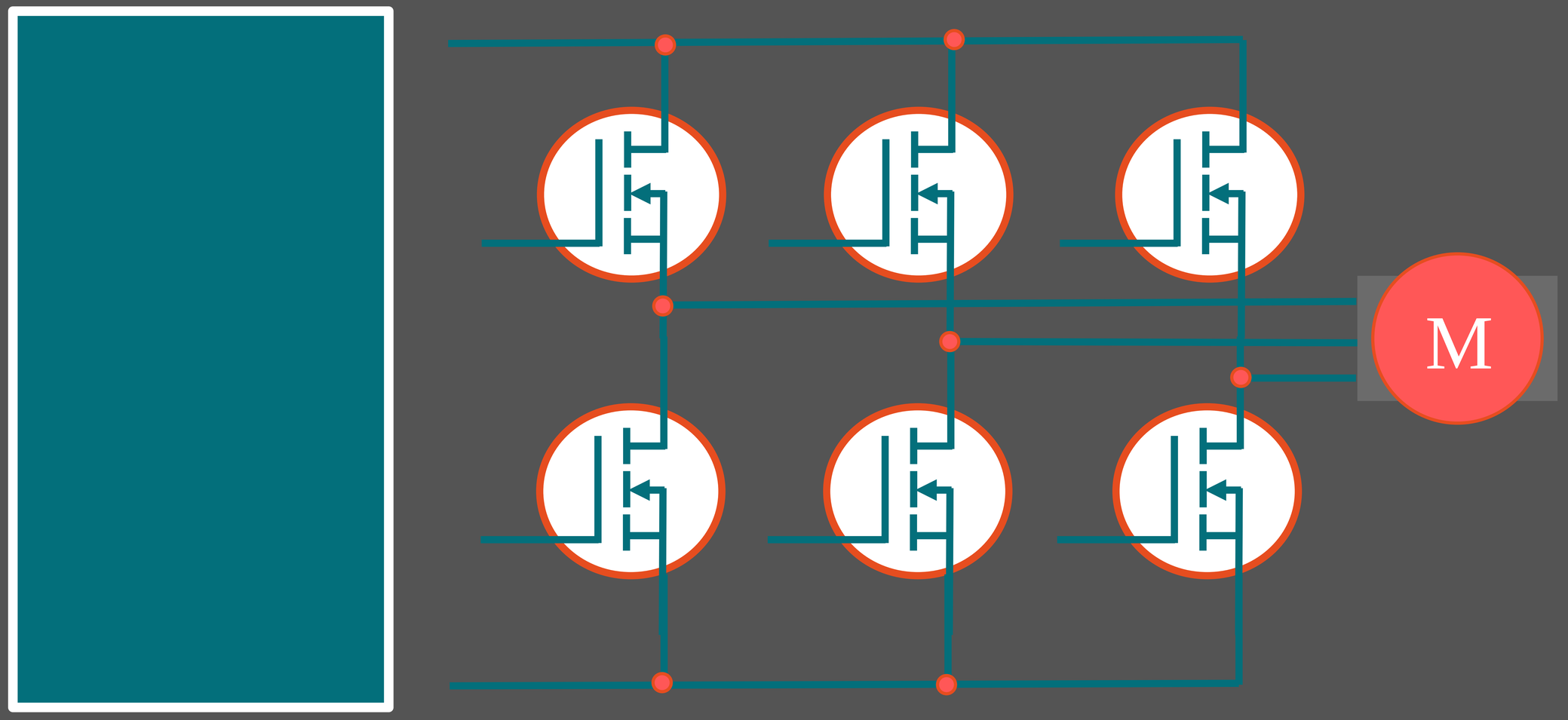
Обидва способи використовують ту саму основну шестиступеневу послідовність, але відрізняються способом подачі ШІМ під час фаз вмикання/вимикання, обираючи між простотою та корисністю.

Можливости бібліотеки

- Жорстке перемикання за шестиступеневою схемою
- М’яке перемикання за шестиступеневою схемою
- Стійкість до відмови одного сигналу датчика Холла
- ПІД-регулятор
- Захист від зависання та завад (шумів) завдяки затримці очікування та відслюванню перешкод

Пояснення принципу: Спрощення до пвмосту

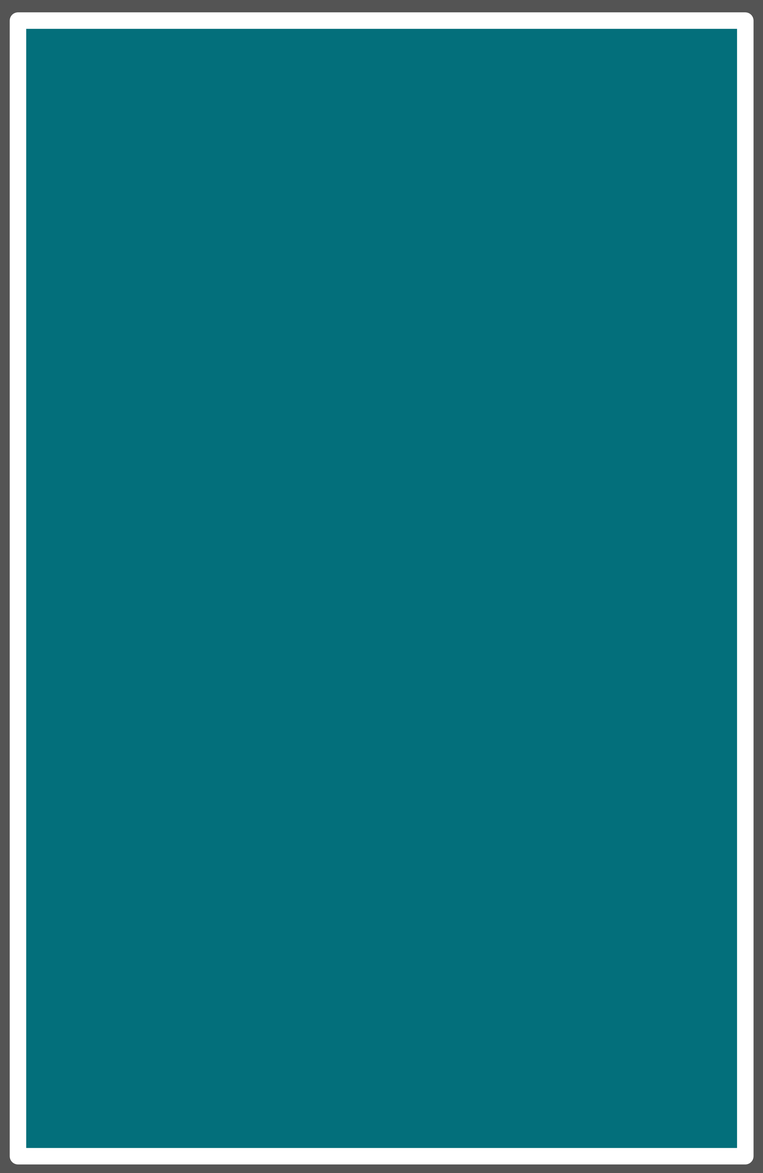
Щоб зробити шляхи струму зрозумілішими, кожен повний трифазний міст зводиться до відповідної пари активних пвмостів.

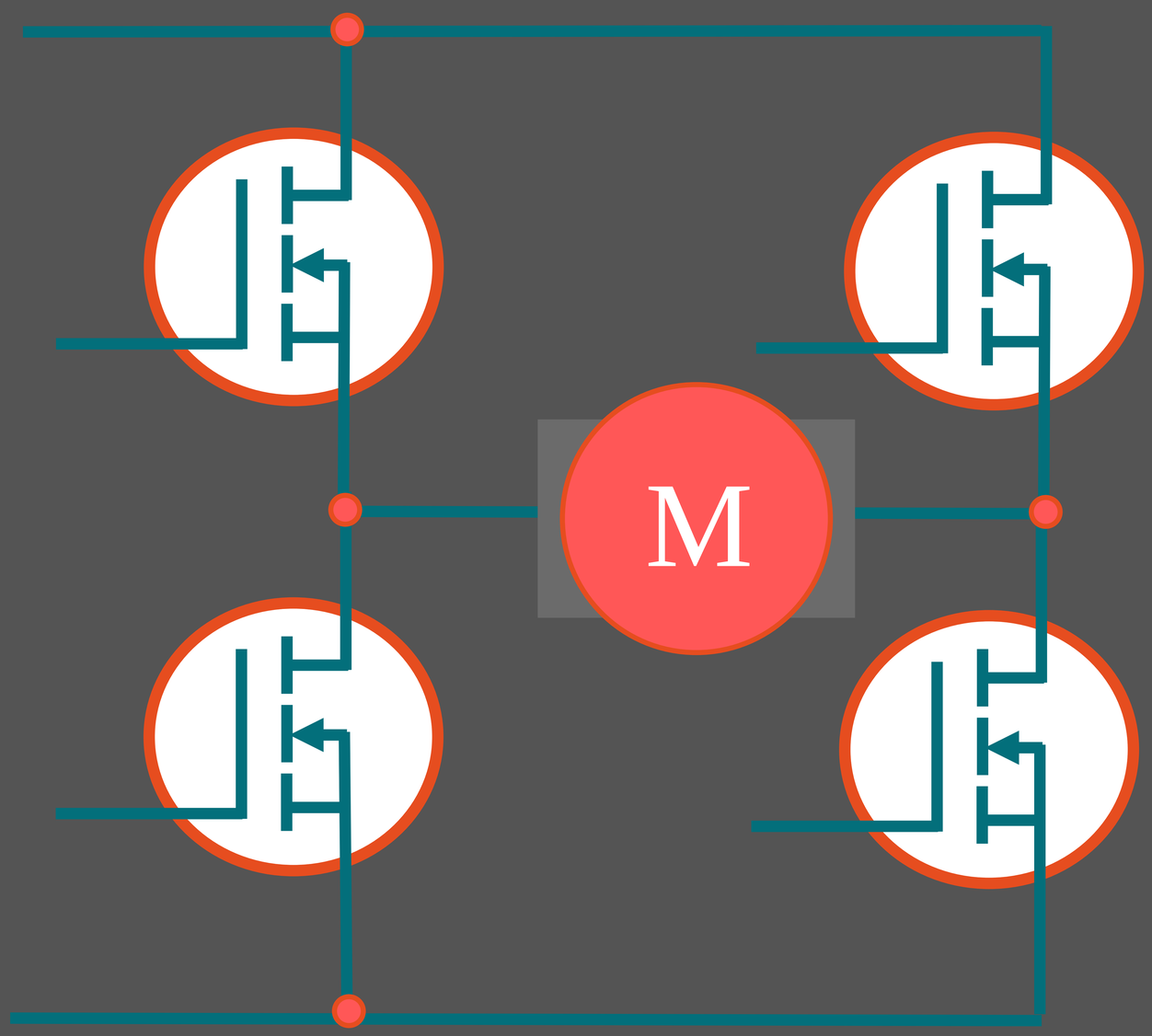


У будь-який заданий момент роботи проводять лише два транзистори:

- Верхній (високобічний) транзистор фази X
- Нижній (низькобічний) транзистор фази Y

Усі інші складові залишаються вимкненими, тому для розуміння втрат на перемикання та шляхів протікання струму достатньо моделі з двома елементами.





Жорстке перемикання

- Керування: ШІМ на верхньому ключі фази X та ШІМ на нижньому ключі фази Y (з однаковим робочим циклом)

- Шлях струму під час вмикання: Верхній ключ X → обмотка двигуна → нижній ключ Y
- Шлях струму під час вимикання: Вільне обертання через зворотні діоди нижнього ключа Y та верхнього ключа X

Втрати: Два падіння напруги на двох діодах під час кожного інтервалу вимкнення. Втрати на перемикання на обох транзисторах.

М’яке перемикання

- Керування: ШІМ на верхньому ключі фази X; постійне увімкнення (логічна 1) на нижньому ключі фази Y

- Шлях струму під час вмикання: Верхній ключ X → обмотка двигуна → нижній ключ Y
- Шлях струму під час вимикання: Вільне обертання через провідність нижнього ключа Y та один зворотний діод (верхнього X або нижнього)

Втрати: Одне падіння напруги на діоді під час інтервалу вимкнення. Втрати на перемикання на одному транзисторі. Через що вища ефективність.

Увага!!!

Усі канали передбачають керування із активним рівнем «високо».

Налаштування таймерів

- TIM1 (або TIM8): створює доповнювальні ШІМ-сигнали на трьох каналах для шестиступеневого керування.
- TIM2 (або будь-який інший): задає частоту перемикання ступенів та вимірює проміжки часу в мікросекундах.

Відповідний таймер (TIM2) можна задати у файлі timer.c.

Налаштування ПІД-регулятора

Задане значення ПІД-регулятора пов’язане з періодом перезавантаження таймера TIM2 (або будь-якого іншого), що визначає інтервал між ступенями. За потреби змінюйте ці значення для досягнення бажаної поведінки в pid.c.

Ідея відмовостійкости датчика Холла полягає в тому, що не має значення, який саме датчик вийшов з ладу; єдине, що потрібно — це виявити {1, 1, 1}, і незалежно від того, який датчик пошкоджений, третій та шостий крок (якщо вважати {1, 1, 1} за первий) не будуть помічені мікроконтролером. Тому ми відтворюємо кінець кроку, просто припиняючи крок після того, як пройде той самий час, за який попередній крок був виконаний.

Звичайно, щоб уникнути обставин при швидкому розгоні, коли кожен наступний крок набагато коротший за попередній, ми додаємо перериванння за допомогою датчика Холла разом з перериванням за часом під час кроків, які я називаю «повторними» кроками.

Приклад Використання

```
#include "SIXSTEPHeader/MAL_declaration.h"

int main(void)
{

  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM1_Init();
  MX_TIM2_Init();
  /* USER CODE BEGIN 2 */

  HAL_TIM_Base_Start(&htim2);


  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
   MAL_Run(HardSwitching, &htim1, function); // Виконання шести кроків

    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```