Лабораторная работа № 14

# Управление шаговым электродвигателем

Цель работы:

1. Ознакомление с принципом работы шагового электродвигателя.
2. Реализация простых алгоритмов управления шаговым электродвига- телем на основе микроконтроллера ATmega8535.

Оборудование и программное обеспечение:

1. Лабораторный макет MK8535 в составе: модуль базовый МБ8535 +

модуль сменный МС07.

1. Кабель соединительный для программирования МК.
2. Персональный компьютер.
3. Интегрированная среда программирования CodeVisionAVR.

# Описание лабораторного макета

Функциональная схема лабораторного макета приведена на рис. 87. В работе используется сменный модуль МС07, содержащий 4-фазный ШД (на схеме – SM1) с углом поворота 7,5 град./шаг (48 шагов/оборот).

Токи в фазные обмотки поступают через транзисторные ключи VT1…VT4, которые управляются сигналами с выходов 0..3 порта A микроконтроллера (МК). Для управления поворотом ШД программа МК должна выводить в порт A последовательность двоичных комбинаций («единицы» включают фазные токи). Светодиоды VD5…VD8 показы- вают, какие из фазных обмоток включены. Диоды VD1…VD4 предна- значены для подавления индуктивных выбросов напряжения на фазных обмотках ШД.

На макете имеются две кнопки S1 и S2, подключенные к входам INT0 и INT1 МК. При нажатии на кнопки на входы поступают сигналы низкого уровня.

# Подготовка к работе

Рассмотрим и выполним примеры, демонстрирующие управление ШД в режимах полного шага и половинного шага.

Создадим проект в CodeVisionAVR и зададим нужный режим ра- боты микроконтроллера. Линии 0…3 порта A настроить на вывод – они

управляют обмотками ШД. Линию 0 порта D настроить на ввод – к ней подключен выход датчика начального положения ротора ШД.

На вкладке **External IRQ** разрешить прерывания INT0 и INT1

и выбрать для них режим **Falling Edge**.

Рис. 87. Функциональная схема лабораторного макета

# Полношаговый режим ШД

Чтобы привести ротор ШД во вращение, нужно последовательно, с заданным интервалом времени, подавать токи в фазные обмотки в со- ответствии с диаграммой на рис. 84. Для этого целесообразно разме- стить массив двоичных комбинаций, где единицы соответствуют фаз-

ным токам, в памяти МК и, затем, циклически выбирать их и подавать на входы ключей:

# unsigned char Ph4[4] = {0x01, 0x02, 0x04, 0x08};

По нажатию кнопки S1, связанной с входом INT0, индекс массива будет увеличиваться, и ротор ШД будет поворачиваться на один шаг (7,5º) в прямом направлении. По нажатию кнопки S2 (INT1) происходит уменьшение индекса и ротор, соответственно, поворачивается в обрат- ном направлении.

Ниже приведен пример программы для МК ATmega8535. В нем не показаны те строки, сформированные **CodeWizardAVR**, которые не относятся непосредственно к данному примеру. Кодовые комбинации для полношагового режима помещены в массив **Ph4[]**. Функция **FullStep()** реализует поворот ротора ШД один шаг. Направление задается параметром **dir**. Задержка 100 мс нужна, чтобы ротор успел повернуться (если выполняется серия шагов).

**#include <mega8535.h> #include <delay.h>**

**unsigned char Ph4[4] = {0x01, 0x02, 0x04, 0x08}; unsigned char index = 0;**

**#define ForWd 0**

**#define BakWd 1**

**void FullStep(unsigned char dir)**

**{**

# if (dir == 0)

**{**

# if (index < 3) index++; else index = 0;

**}**

# else

**{**

# if (index > 0) index--; else index = 3;

**}**

# PORTA = Ph4[index]; delay\_ms(100);

**}**

# // External Interrupt 0 service routine interrupt [EXT\_INT0] void ext\_int0\_isr(void)

**{**

# #asm(«cli») FullStep(ForWd); #asm(«sei»)

**}**

# // External Interrupt 1 service routine interrupt [EXT\_INT1] void ext\_int1\_isr(void)

**{**

# #asm(«cli») FullStep(BakWd); #asm(«sei»)

**}**

# void main(void)

**{**

**PORTA=0x00; DDRA=0x0F; // Port A init. PORTD=0x00; DDRD=0x00; // Port D initi.**

**// External Interrupt(s) initialization**

**// INT0: On. INT0 Mode: Low level**

**// INT1: On. INT1 Mode: Low level**

**// INT2: Off GICR|=0xC0; MCUCR=0x00; MCUCSR=0x00; GIFR=0xC0;**

**PORTA = Ph4[index]; // Включить ток обмоток**

**delay\_ms(100);**

**#asm(«sei») // Разрешить прерывания**

**while (1) {};**

**}**

# Полушаговый режим ШД

В этом режиме, за счет перехвата фаз, угловой шаг ротора состав- ляет 3,75º. Массив фазных кодов **Ph8[]** содержит 8 элементов:

# unsigned char Ph8[8]=

**{0x01,0x03,0x02,0x06,0x04,0x0C,0x08,0x09};**

Шаг ротора реализует функция **HalfStep()**. В остальном при- мер не отличается от предыдущего.

**#include <mega8535.h> #include <delay.h> unsigned char Ph8[8] =**

**{0x01,0x03,0x02,0x06,0x04,0x0C,0x08,0x09};**

**unsigned char index = 0; #define ForWd 0**

**#define BakWd 1**

**void HalfStep(unsigned char dir)**

**{**

# if (dir == ForWd)

**{**

# if (index < 7) index++; else index = 0;

**}**

# else

**{**

# if (index > 0) index--; else index = 7;

**}**

# PORTA = Ph8[index]; delay\_ms(100);

**}**

# // External Interrupt 0 service routine interrupt [EXT\_INT0] void ext\_int0\_isr(void)

**{**

# #asm(«cli») HalfStep(ForWd); #asm(«sei»)

**void main(void)**

**{**

**PORTA=0x00; DDRA=0x0F; // Port A init. PORTD=0x00; DDRD=0x00; // Port D init.**

**// External Interrupt(s) initialization**

**// INT0: On**

**// INT0 Mode: Low level**

**// INT1: On**

**// INT1 Mode: Low level**

**// INT2: Off GICR|=0xC0; MCUCR=0x00; MCUCSR=0x00; GIFR=0xC0;**

**PORTA = Ph8[index]; // Включить ток обмоток**

**delay\_ms(100);**

**#asm(«sei») // Разрешить прерывания**

**while (1) {};**

**}**

# Задание

1. Создать проекты и реализовать рассмотренные выше примеры для полношагового и полушагового режимов ШД.
2. Изменить код программы так, чтобы по нажатию кнопки RESET ротор ШД вначале установился в исходное угловое положение по сигналу датчика, а затем повернулся на 90º вперед в полношаговом режиме.
3. Условия п. 2 повторить для полушагового режима, ротор повернуть на 270º назад от исходного положения.

**}**

# // External Interrupt 1 service routine interrupt [EXT\_INT1] void ext\_int1\_isr(void)

**{**

# #asm(«cli») HalfStep(BakWd); #asm(«sei»)

**}**

# УПРАВЛЕНИЕ ЭЛЕКТРОДВИГАТЕЛЕМ ПОСТОЯННОГО ТОКА

Управление двигателем постоянного тока (ДПТ) сводится к подаче на него требуемого напряжения заданной полярности. Величина напря- жения задает скорость, а полярность – направление вращения. Типичная схема выходного каскада представлена на рис. 88. Видно, что схема пред- ставляет собой полный Н-мост.

Рис. 88. Выходной каскад схемы управления ДПТ

От схемы управления поступают команды F (Forward – вперёд) и R (Reverse – назад). При подаче этих команд изменяется полярность напряжения, прикладываемого к двигателю. Когда F = 1 и R = 0, ток течет по цепи: открытый транзистор VT4 → двигатель → транзистор VT3. Если F = 0 и R = 1, ток протекает по другой цепи: транзистор VT2 → двигатель → транзистор VT5. Заметим, что направление враще- ния двигателя – параметр относительный, поэтому здесь и далее выра- жения «вращение вперед (назад)» указывают два состояния двигателя с разными направлениями вращения.

Если F = R = 0, двигатель работает в режиме останова. При этом двигатель работает в условиях, близких к холостому ходу, то есть фак- тически вращается по инерции.

Наиболее быстро двигатель останавливается при F = R = 1 (режим торможения), когда он замкнут накоротко открытыми нижними транзи-

сторами VT3 и VT5, так как в это время запасенная в роторе кинетиче- ская энергия рассеивается на сопротивлении обмотки.

Рассмотренные режимы работы ДПТ сведены в табл. 35.

*Таблица 35*

Действие команд управления

|  |  |  |
| --- | --- | --- |
| Команды | Состояние | Действие |
| F | R | OUT+ | OUT– |
| 0 | 1 | Н | В | Вращение вперед |
| 1 | 0 | В | Н | Вращение назад |
| 0 | 0 | В | В | Останов |
| 1 | 1 | Н | Н | Торможение |

Как видно из построения схемы, приведенной на рис. 88, напря- жение, прикладываемое к двигателю, не может быть больше напряжения на входе управления VC (Voltage Control). Напряжение на этом входе не линейно, но монотонно связано с напряжением на двигателе, поэтому его используют для управления скоростью.

Диоды VD3…VD6 на рис. 88 служат для защиты транзисторов моста от индуктивных выбросов напряжения с обмоток двигателя.

# Управление скоростью вращения при помощи ШИМ

Достаточно высокая (по сравнению с быстродействием управля- ющего микроконтроллера) инерционность электродвигателей позволяет использовать для управления не аналоговый сигнал, от амплитуды кото- рого линейно зависит желаемая частота вращения, а ШИМ-сигнал, в котором соответствующим образом изменяется соотношение длитель- ности импульса и паузы. В сущности, изменение этого соотношения (соответственно и мощности, передаваемой на двигатель за период дей- ствия ШИМ-сигнала) эквивалентно изменению уровня аналогового напряжения управления. Преимущества ШИМ – это экономичность (транзисторы работают в ключевом режиме), и относительная простота реализации (не требуется ЦАП). Поэтому на вход VC (см. рис. 88) для управления скоростью вращения будем подавать ШИМ-сигнал от мик- роконтроллера.

Сглаживание импульсов ШИМ происходит за счет инерционно- сти ДПТ. Применение ФНЧ здесь не обязательно и даже нежелательно – на их последовательных элементах могут происходить потери мощности при больших токах.