

Управление шаговым двигателем при помощи микроконтроллера Fujitsu MB90F598

Введение

Шаговые двигатели широко используются в принтерах, автоматических инструментах, приводах дисководов, автомобильных приборных панелях и других приложениях, требующих высокой точности позиционирования и микропроцессорного управления. Как известно, такое управление требует использования специальной логики и высокоточных драйверов, которые могут быть реализованы на дискретной элементной базе, что увеличивает сложность схемы и ее стоимость.

Для упрощения процесса разработки и уменьшения стоимости конечного изделия мы предлагаем использовать контроллер шагового двигателя, интегрированный в недорогой процессор Fujitsu MB90F598. Стоит отметить, что подобным контроллером обладают многие процессоры семейства F2MC-16LX, например MB90F428, что дает возможность пользователю выбрать контроллер с желаемой периферией.

Архитектура

Небольшие шаговые двигатели часто используются, например, в автомобильных приборных панелях (инструментальных кластерах) и выполняют там функции вращения стрелок спидометра, тахометра, указателя температуры охлаждающей жидкости и уровня топлива. При этом по сравнению с традиционно используемыми гальванометрическими системами отсутствует вибрация стрелки, увеличивается точность показаний. Один процессор MB90F598 способен обслуживать четыре независимых шаговых двигателя, которые подключаются непосредственно к процессору без дополнительных интерфейсных схем. Рассмотрим коротко принцип действия шаговых двигателей и их отличие от двигателей постоянного тока.

Шаговые двигатели: принцип действия и отличия от двигателей постоянного тока

Двигатели постоянного тока (ДПТ) начинают работать сразу, как только к ним будет приложено постоянное напряжение. Переключение направления тока через обмотки ротора осуществляется механическим коммутатором - коллектором. Постоянные магниты при этом расположены на статоре. Шаговый двигатель может быть рассмотрен как ДПТ без коммутатора. Обмотки его являются частью статора. На роторе расположен постоянный магнит или, для случаев с переменным магнитным сопротивлением, зубчатый блок из магнитомягкого материала. Все коммутации производятся внешними схемами. Обычно система мотор - контроллер разрабатывается так, чтобы была возможность вывода ротора в любую, фиксированную позицию, то есть система управляется по положению. Цикличность позиционирования ротора зависит от его геометрии.

Принято различать шаговые двигатели и серводвигатели. Принцип их действия во многом похож, и многие контроллеры могут работать с обоими типами. Основное отличие заключается в количестве шагов на цикл (один оборот ротора). Серводвигатели требуют наличия в системе управления аналоговой обратной связи, в качестве которой обычно используется потенциометр. Ток в этом случае обратно пропорционален разности желаемого и текущего положений. Шаговые двигатели преимущественно используются в системах без обратных связей, требующих небольших ускорений при движении.

Шаговые двигатели (ШД) делятся на две разновидности: двигатели с постоянными магнитами и двигатели с переменным магнитным сопротивлением (гибридные двигатели). С точки зрения контроллера отличие между ними отсутствует. Двигатели с постоянными магнитами обычно имеют две независимые обмотки, у которых может присутствовать или отсутствовать срединный отвод (см. рис.1).

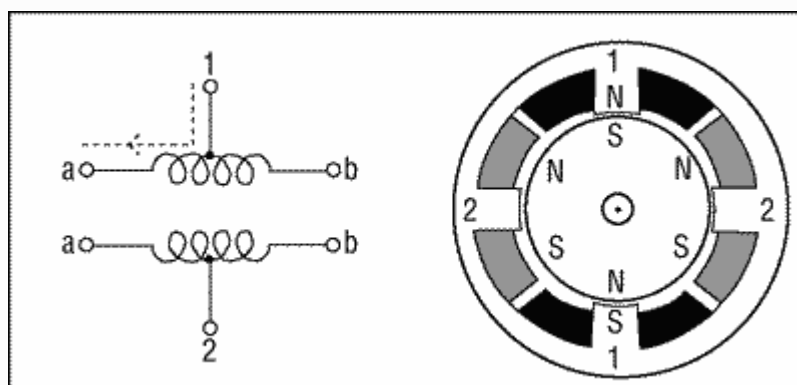


Рис.1. Униполярный ШД с постоянными магнитами.

Биполярные шаговые двигатели с постоянными магнитами и гибридные двигатели сконструированы более просто, чем униполярные двигатели, обмотки в них не имеют центрального отвода (см рис.2).

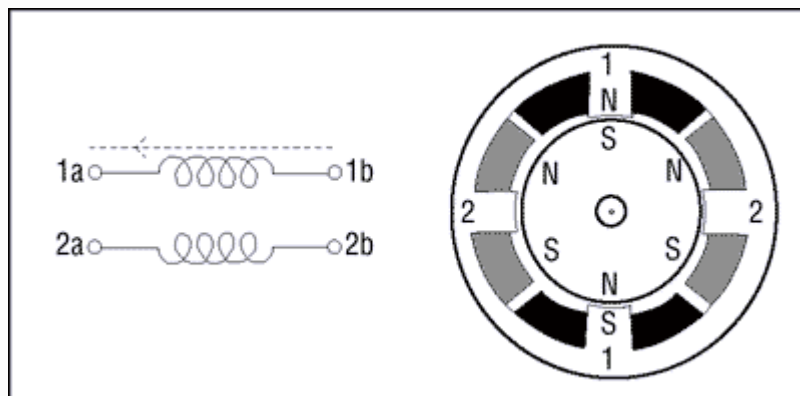


Рис.2. Биполярный и гибридный ШД.

За это упрощение приходится платить более сложным реверсированием полярности каждой пары полюсов мотора.

Шаговые двигатели имеют широкий диапазон угловых разрешений. Более грубые моторы обычно вращаются на 90° за шаг, в то время как прецизионные двигатели могут иметь разрешение $1,8^\circ$ или $0,72^\circ$ на шаг. Если контроллер позволяет, то возможно использование полушагового режима или режима с более мелким дроблением шага (микрошаговый режим), при этом на обмотки подаются дробные значения напряжений, зачастую формируемые при помощи ШИМ - модуляции.

Если в процессе управления используется возбуждение только одной обмотки в любой момент времени, то ротор будет поворачиваться на фиксированный угол, который будет удерживаться пока внешний момент не превысит момента удержания двигателя в точке равновесия.

Для правильного управления биполярным шаговым двигателем необходима электрическая схема, которая должна выполнять функции старта, стопа, реверса и изменения скорости. Шаговый двигатель транслирует последовательность цифровых переключений в движение. "Вращающееся" магнитное поле обеспечивается соответствующими переключениями напряжений на обмотках. Вслед за этим полем будет вращаться ротор, соединенный посредством редуктора с выходным валом двигателя.

Каждая серия содержит высокопроизводительные компоненты, отвечающие все возрастающим требованиям к характеристикам современных электронных применений.

Схема управления для биполярного шагового двигателя требует наличия мостовой схемы для каждой обмотки. Эта схема позволит независимо менять полярность напряжения на каждой обмотке. На рис.3 показана последовательность управления для режима с единичным шагом.



Рис.3. Управляющая последовательность для режима с единичным шагом.

На рис.4 показана последовательность для полушагового управления.

Индекс	1a	1b	2a	2b
1	+	-	-	-
2	+	+	-	-
3	-	+	-	-
4	-	+	+	-
5	-	-	+	-
6	-	-	+	+
7	-	-	-	+
8	+	-	-	+
9	+	-	-	-
10	+	+	-	-
11	-	+	-	-
12	-	+	+	-
13	-	-	+	-
14	-	-	+	+
15	-	-	-	+
16	+	-	-	+

Рис.4. Управляющая последовательность для режима с половинным шагом.

Блок управления шаговым двигателем в контроллере MB90F598

Блок управления шаговым двигателем в контроллере MB90F598 включает в себя драйверы, селектор и два ШИМ - генератора. Четыре драйвера имеют мостовые схемы, рассчитанные на высокий ток и подключаемые непосредственно к обмоткам двигателя. Комбинация ШИМ - генератора и селектора обеспечивает управление двигателем. Механизм синхронизации позволяет синхронизировать работу обоих ШИМ - генераторов. Назначение пинов процессора показано в таблице 1.

Номер пина	Название пина	Функция
54 - 57	PWM1P0 PWM1M0 PWM2P0 PWM2M0	Выходы для нулевого канала управления ШД.
59 - 62	PWM1P1 PWM1M1 PWM2P1 PWM2M1	Выходы для первого канала управления ШД.
64 - 67	PWM1P2 PWM1M2 PWM2P2 PWM2M2	Выходы для второго канала управления ШД
69 - 72	PWM1P3 PWM1M3 PWM2P3 PWM2M3	Выходы для третьего канала управления ШД
58, 68	DVCC	"+" моста
53, 63, 73	DVSS	"-" моста

Схема выходного пина показана на рис.5.

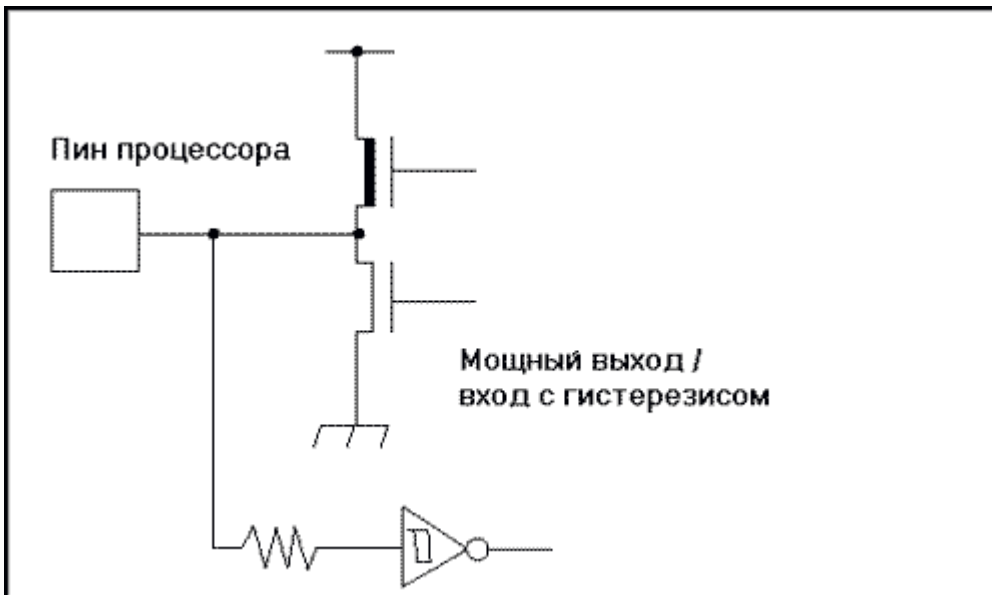


Рис.5. Схема выходного пина.

На рис.6 показана структурная схема одного канала блока управления шаговым двигателем.

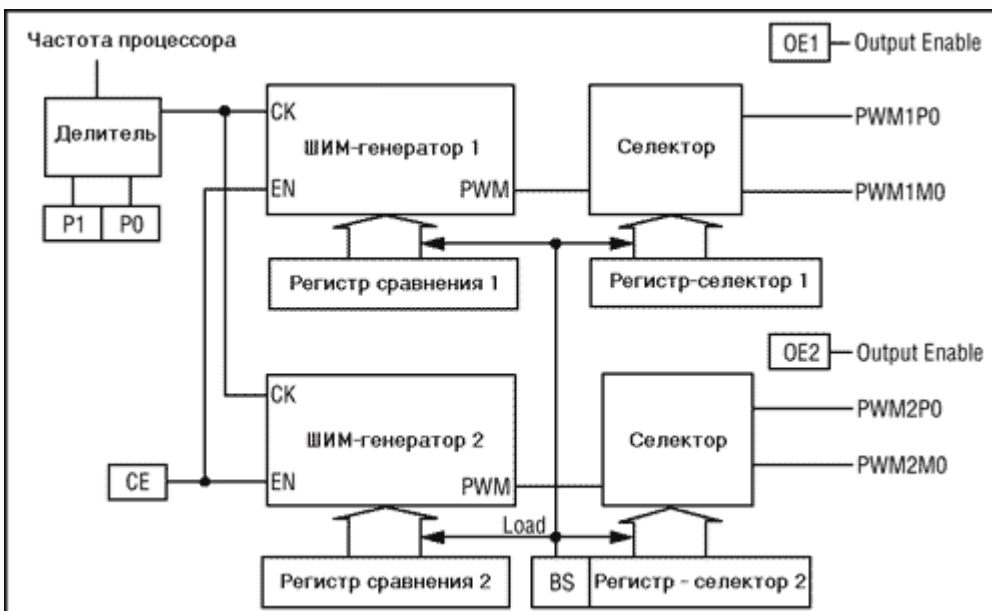


Рис.6. Блок-схема канала управления ШД.

Рассмотрим назначение и структуру регистров, ассоциируемых с модулем управления шаговым двигателем.

(1) **Управляющий регистр ШИМ 0**

PWM Control 0 Register
Address: 00005Eh

Bit Number	7	6	5	4	3	2	1	0
Bit Name	OE2	OE1	P1	P0	CE	—	—	TST
Read/Write	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	—	—	(R/W)
Initial Value	(0)	(0)	(0)	(0)	(0)	—	—	(0)

PWC0

Рис.7. Регистр управления.

[бит 7] OE2: Разрешение выхода.

Когда этот бит установлен в "1", внешние пины PWM2P0 и PWM2M0 подключаются к выходам ШИМ. В противном случае они являются портами общего назначения.

[бит 6] OE1: То же самое для пинов PWM1P0 и PWM1M0.

[биты 5 и 4] P1, P0: Выбор рабочей частоты ШИМ. Биты специфицируют (в соответствии с табл.2) входную частоту для ШИМ-генераторов.

Таблица 2

P1	P0	Входная частота для ШИМ - генераторов
0	0	Основная частота процессора
0	1	Основная частота процессора / 2
1	0	Основная частота процессора / 4
1	1	Основная частота процессора / 8

[бит 3] CE: Разрешение счета.

Этот бит разрешает работу ШИМ - генератора. Когда он устанавливается в "1", ШИМ -генератор запускается. Отметим, что второй генератор PWM2 запускается через один машинный цикл после первого. Это сделано чтобы снизить шумы переключений в выходных драйверах.

[бит 0] TST: Тестовый бит.

Этот бит предназначен для тестирования процессора. В приложениях пользователя он всегда должен быть установлен в 0.

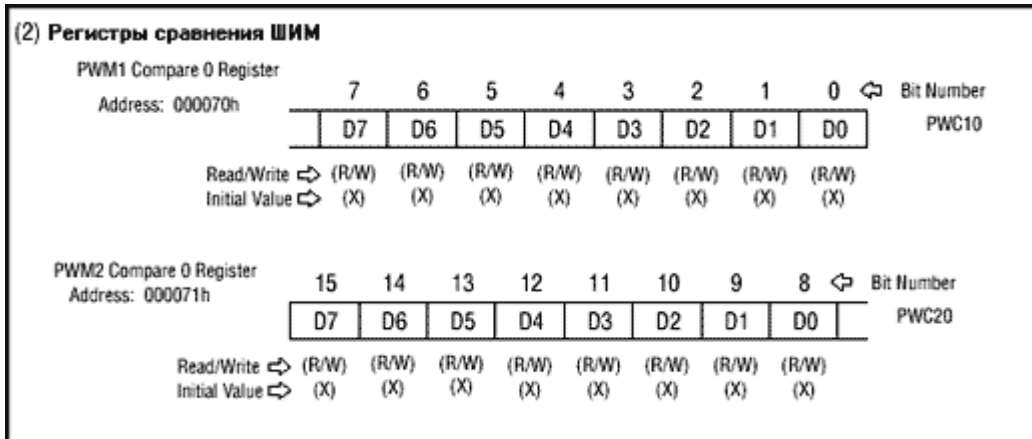


Рис.8. Регистры сравнения.

Содержимое этих двух 8-разрядных регистров определяет ширину импульсов (коэффициент ШИМ - модуляции), как показано на рис.9. Значение 00h соответствует заполнению 0%, а значение FFh - значению заполнения 99%.

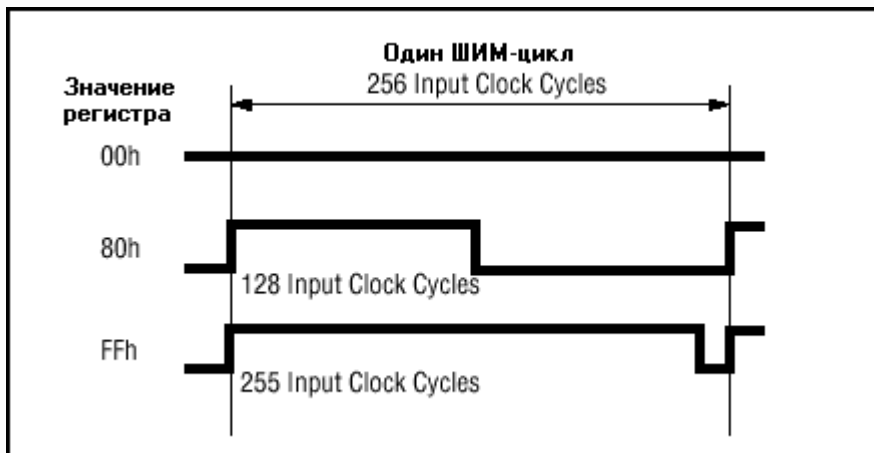


Рис.9. ШИМ - модуляция.

Для изменения значения коэффициента ШИМ - модуляции необходимо записать в регистры сравнения новое значение, после чего записать "1" в бит BS (Update bit) селекторного регистра, показанного на рис.10.



Рис.10. Селекторные регистры.

Селекторные регистры предназначены для переключения режимов работы выходов контроллера управления ШД. Рассмотрим назначение отдельных битов этих регистров.

[бит 14] BS: Бит модификации (Update bit).

Этот бит устанавливается для синхронных модификаций установок выходов ШИМ - генератора. Любая модификация в обоих регистрах сравнения или в регистрах - селекторах не будет выполнена, пока этот бит не будет установлен. При записи "1" в этот бит все изменения будут выполнены в конце текущего ШИМ - цикла. Бит автоматически обнуляется в начале следующего ШИМ - цикла.

[биты 13 - 11] P2 - P0: Эти биты выбирают режим для вывода PWM2P0.

[биты 10 - 8] M2 - M0: Эти биты выбирают режим для вывода PWM2M0.

[биты 5 - 3] P2 - P0: Эти биты выбирают режим для вывода PWM1P0.

[биты 2 - 0] M2 - M0: Эти биты выбирают режим для вывода PWM1M0.

В таблице 3 представлено соотношение между значениями этих битов и уровнями выходных сигналов.

Таблица 3

P2	P1	P0	PWMnPO
0	0	0	Низкий уровень
0	0	1	Высокий уровень
0	1	X	ШИМ - сигнал
1	X	X	Высокий импеданс
M2	M1	M0	PWMnMO
0	0	0	Низкий уровень
0	0	1	Высокий уровень
0	1	X	ШИМ - сигнал
1	X	X	Высокий импеданс

Таким образом, мы рассмотрели структуру блока управления шаговым двигателем и регистры, используемые для работы с этим блоком. Далее рассмотрим пример программы, реализующей движение ШД.

Изменяя размер шага в выходном сигнале можно управлять скоростью вращения шагового двигателя. Различные размеры шага могут быть запрограммированы путем загрузки соответствующих значений в регистры сравнения. При управлении ШД необходимо позаботиться о том, чтобы уменьшать скорость вращения при подходе к желаемой позиции. Это демонстрируется представленным ниже исходным кодом.

Программа пользователя должна установить процедуру обработки прерывания по переполнению перегружаемого таймера для изменения позиции шагового двигателя. В этом обработчике для управления внешними пинами должна осуществляться перезагрузка регистров сравнения и устанавливаться сигналы "OE" в регистре управления. Значение периода переполнения перегружаемого таймера выбирается так, чтобы обеспечить плавность движения выходного вала ШД.

Следующий код иллюстрирует управление позицией ШД, путем установки требуемого параметра в управляющей программе.

```

// Управление ротором одного ШД
void TestStepperMotor_0 (void)
{
  unsigned long ctr;
  unsigned long ctr2;
  DDR4_D40 = 1;
  PDR4_P40 = 1;
  DDR4_D47 = 1;
  PDR4_P47 = 0;
  uiMaxSpeed = 1; // Управление скоростью
  for(ctr = 500000L; ctr; ctr--);
  uiRequiredPosition = MAX_STEPS;
  // MAX_STEPS - максимально
  // возможное количество шагов
  // двигателя в любом направлении.
  for(ctr = 700000L; ctr; ctr--);
  uiRequiredPosition = 0;
  while (uiCurrentPosition != uiRequiredPosition);

  for(ctr = 700000L; ctr; ctr--);
  uiRequiredPosition = 1000;
  while (uiCurrentPosition != uiRequiredPosition);
  for(ctr = 700000L; ctr; ctr--);
  uiRequiredPosition = 2000;
  while(uiCurrentPosition != uiRequiredPosition);
  for(ctr = 700000L; ctr; ctr--);
  uiRequiredPosition = 1500;
  while(uiCurrentPosition != uiRequiredPosition);
  for(ctr = 700000L; ctr; ctr--);
  uiRequiredPosition = 500;
  while(uiCurrentPosition != uiRequiredPosition);
  for(ctr = 700000L; ctr; ctr--);
  while (0); }

```

В пределах одного шага требуемая позиция (которая может быть получена от входных сенсоров) устанавливается переменной "uiRequiredPosition". Эта переменная модифицируется в обработчике прерывания перегружаемого таймера пока требуемая позиция не станет равной текущей позиции ротора. Наличие рассогласования между требуемой и текущей позициями означает, что вал двигателя еще не достиг требуемой позиции.

Селекторный регистр используется для того, чтобы установить нужный квадрант магнитного потока. Например для двухполюсного двухобмоточного ШД существует четыре различные позиции, в которых может находиться ротор. В зависимости от квадранта ротора программируются регистры сравнения и селекторные регистры, что обеспечивает необходимый поворот ротора.

В предлагаемом примере используется только один из четырех каналов управления ШД, именно к нему должен быть подключен двигатель. Четыре возможных квадранта могут быть запрограммированы путем установки соответствующих значений для битов P1 и M1 в селекторных регистрах PWS1 и PWS2. Программирование этих, взаимно исключаемых квадрантов показано в табл.4 и табл.5.

Таблица 4

PWS10_M1	PWS10_P1	PWS20_M1	PWS20_P1	Обмотка 2	Обмотка 1
0	1	0	1	+	+
0	1	1	0	-	+
1	0	1	0	-	-
1	0	0	1	+	-

Четыре квадранта и соответствующие им значения регистров представлены в табл.5.

Таблица 5

Квадрант	Двоичное значение (№,бит 1,4,9,12)	Шестнадцатеричное значение
PWS_QI	0001 0000 0001 0000	0x1010
PWS_QII	0000 0010 0001 0000	0x0210
PWS_QIII	0000 0010 0000 0010	0x0202
PWS_QIV	0001 0000 0000 0010	0x1002

Значение регистров сравнения определяет ширину импульса ШИМ. Для достижения сглаженности движения из 256 возможных значений этих регистров выделены 32 значения (по одному на каждый шаг), представленные следующей таблицей.

```

unsigned char LookupTable [32] = {
  0,13,25,37,50,62,74,86,98,109,120,131,142,152,162,1
  71,180,189,197,205,212,219,225,231,236,240,244,247,
  250,252,254,255
};

```

Следовательно, псевдо-код, программирующий сравнение и выбор для первого квадранта может выглядеть так.

```

TableIndex = PositionDesired & 0x1F; // 2 to the power 5 = 32 steps per quadrant
CompareRegister_1 = LookupTable [ TableIndex];
CompareRegister_2 = LookupTable [ 32 - TableIndex];
SelectRegister = PWS_QI; // 1'st Quadrant means coil_1 is Pos and coil_2 is Pos

```

Максимальная скорость движения определяется исходя из физических возможностей шагового двигателя. При этом скорость регулируется путем изменения размера шага. Более крупные шаги соответствуют большей скорости движения. Необходимо аккуратно устанавливать значение регистров сравнения так, чтобы обеспечить снижение скорости по мере приближения к желаемой позиции.

Следовательно, смещение, на которое перемещается ротор при движении из текущей позиции к желаемой, определяет скорость. Для достижения желаемой скорости можно использовать следующее условие.

```

If( (RequiredPosition_0-CurrentPosition_0) >
CurrentOffset_0) {
// We can speed up the motor
CurrentOffset ++;
}
.
.
.
CurrentPosition_0 = CurrentPosition_0 + CurrentOff-set;
.
.
ProgramCompareAndSelectRegisters ();

```

Подобным образом, уменьшая переменную "CurrentOffset" можно последовательно понижать значение скорости при достижении требуемой позиции или достижению максимальной скорости. Движение ротора может осуществляться в двух направлениях: по часовой стрелке и против. При смене направления движения необходимо обеспечить торможение и разгон ротора. Алгоритм работы программы показан на рис.11.

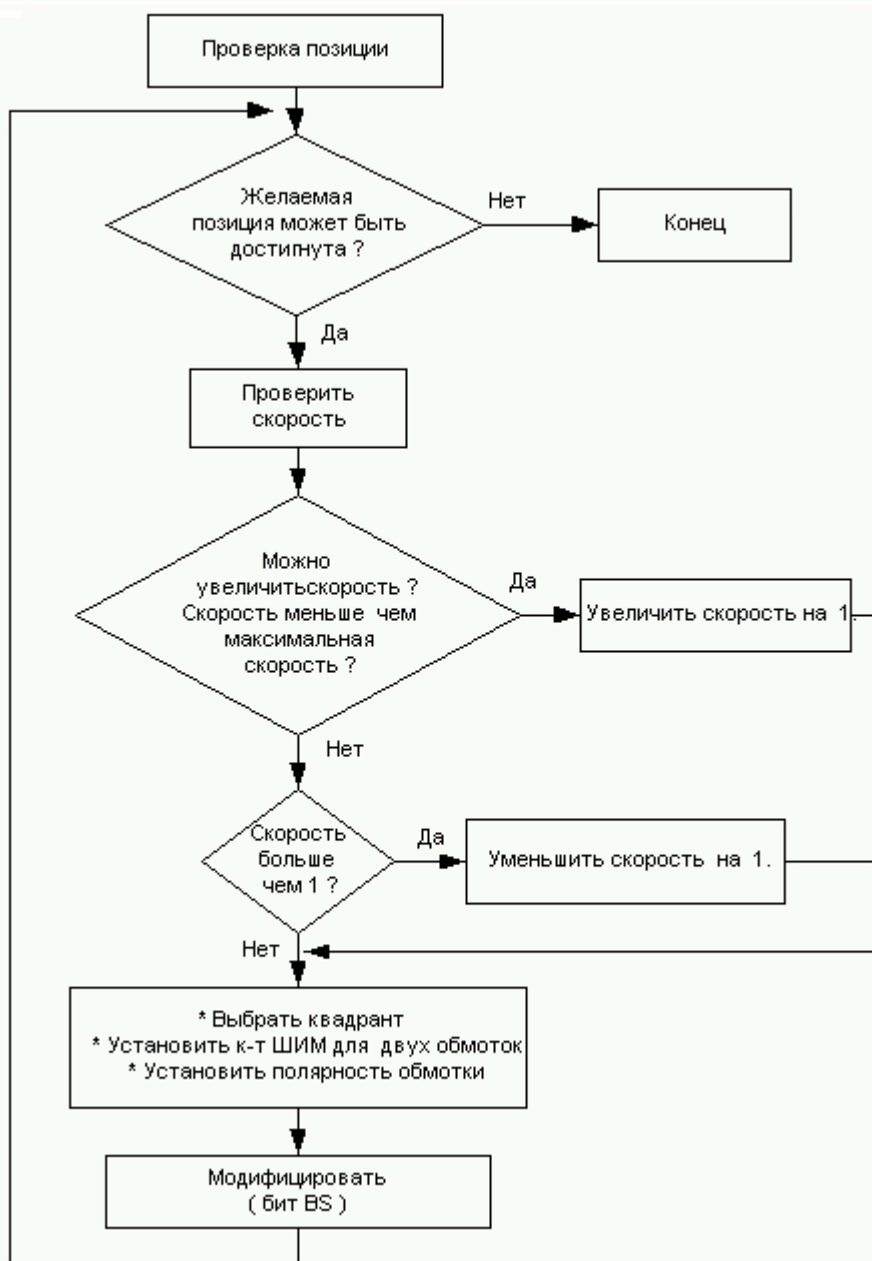


Рис.11. Алгоритм работы программы управления ШД.

Попробовать работу с шаговым двигателем можно используя конструктор FUJITSU, продаваемый в фирме КТЦ-МК (www.cec-mc.ru). Установленный в конструкторе контроллер MB90F598G позволяет подключить к нему до четырех шаговых двигателей. Конструктор содержит минимальные элементы, необходимые для работы контроллера, интерфейсные элементы, позволяющие подключиться к PC, а также макетное поле. Программа может быть подготовлена в интегрированной среде разработки SOFTUNE V3.0 и загружена в контроллер по последовательному порту непосредственно из персонального компьютера при помощи утилиты загрузки Flash Memory Writer. Все программное обеспечение является свободно распространяемым и может быть приобретено в фирме КТЦ-МК.

Для полноценной работы вам необходимы следующие компоненты:

1. Конструктор FUJITSU или фирменный конструктор FLASH-CAN2 Board, Part No. FLASHCAN2-100MP-M06
2. MB90F598 Микроконтроллер (входит в состав конструктора)
3. Шаговый двигатель.
4. Соединительные провода.
5. Кабель RS-232.
6. Компьютер с установленной SOFTUNE V3.0
7. Прикладная программа.
8. Программа загрузки контроллера.
9. Источник питания (5В, 0.5А).

Схема подсоединения одного шагового двигателя к контроллеру (конструктору), приведена на рис.12.

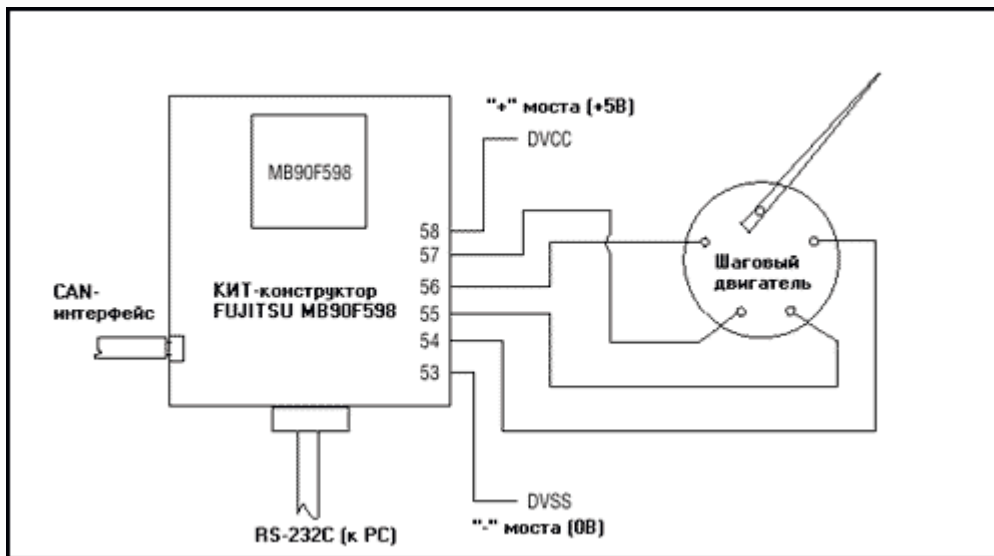


Рис.12. Соединение ШД и контроллера MB90F598.

Литература

1. MB90F598 Data Sheet.
2. MB90F598 Hardware Manual.

Разработчик
фирмы "КТЦ-МК"
Киселёв Д.В.
design@cec-mc.ru