Контроллеры монтировки для камеры сверх-широкого поля (Nanomount)

В.Корнилов, ГАИШ

29 августа 2007 г.

Описание электроники монтировки

Монтировка вилочного типа имеет приводы на обеих осях, предназначенные для наведения камеры на требуемый участок неба, привод полярной оси служит также для ведения с часовой скоростью.

Привод по часовому углу состоит из биполярного шагового двигателя типа FL39 (200 шагов на оборот, сопротивление обмотки 14Ω), вспомогательного редуктора 1:4.5 и основного фрикционного редуктора с отношением 1:45. На полярной оси установлен емкостный абсолютный датчик угла. Двигатель привода установлен на основной плите и закрыт корпусом. Электроника привода смонтирована рядом с двигателем.

Привод оси склонения состоит из биполярного шагового двигателя типа FL39 (200 шагов на оборот, сопротивление обмотки 14Ω) и червячного редуктора 1:100. Двигатель привода установлен на пере вилки. На оси склонения с со стороны второго пера установлен емкостный абсолютный датчик угла. Электроника привода смонтирована под основанием вилки.

Монтировка питается напряжением 24 В, ток менее 1 А.

Электроника монтировки состоит из следующих модулей и узлов:

- Контроллер датчика угла полярной оси (адрес 0x08)
- Датчик угла полярной оси (ротор и статор)
- Контроллер датчика угла оси склонения (адрес 0x09)
- Датчик угла оси склонения (ротор и статор)
- Контроллер шагового двигателя привода по часовому углу (адрес 0x10)
- Контроллер шагового двигателя привода по склонению (адрес 0x11)
- Преобразователь DC/DC +24 В на +5 В.

Подключение контроллеров и других модулей

Контролеры запрограммированы на скорость обмена 115Кбит/с с тем, чтобы можно было использовать конвертер RS485/COM. Принципиальная схема соединений между отдельными модулями и разъемами показана на Рис. 1

Общее описание контроллера шагового двигателя

Контроллер предназначен для работы с двухфазными биполярными шаговыми моторами с сопротивлением обмоток 10 – 100 Ω. Базовыми элементами контроллера являются микроконтроллер ATMega8 фирмы Atmel и драйвер PBL3777 фирмы Ericcson.

Главные функции микроконтроллера:

- поддержка асинхронного полудуплексного обмена с управляющим компьютером;
- поддержка временно́й шкалы работы двигателя;
- выдача управляющих сигналов на драйверы двигателей;
- выдача кода управления током для обеспечения микрошагового режима;

Внешний вид, расположение элементов и подключение внешних сигналов представлены на Рис. 2

Ток через обмотки двигателя регулируется резисторами в соответствии с формулой:

$$I_w = 0.1 U_{ref} / R_s$$

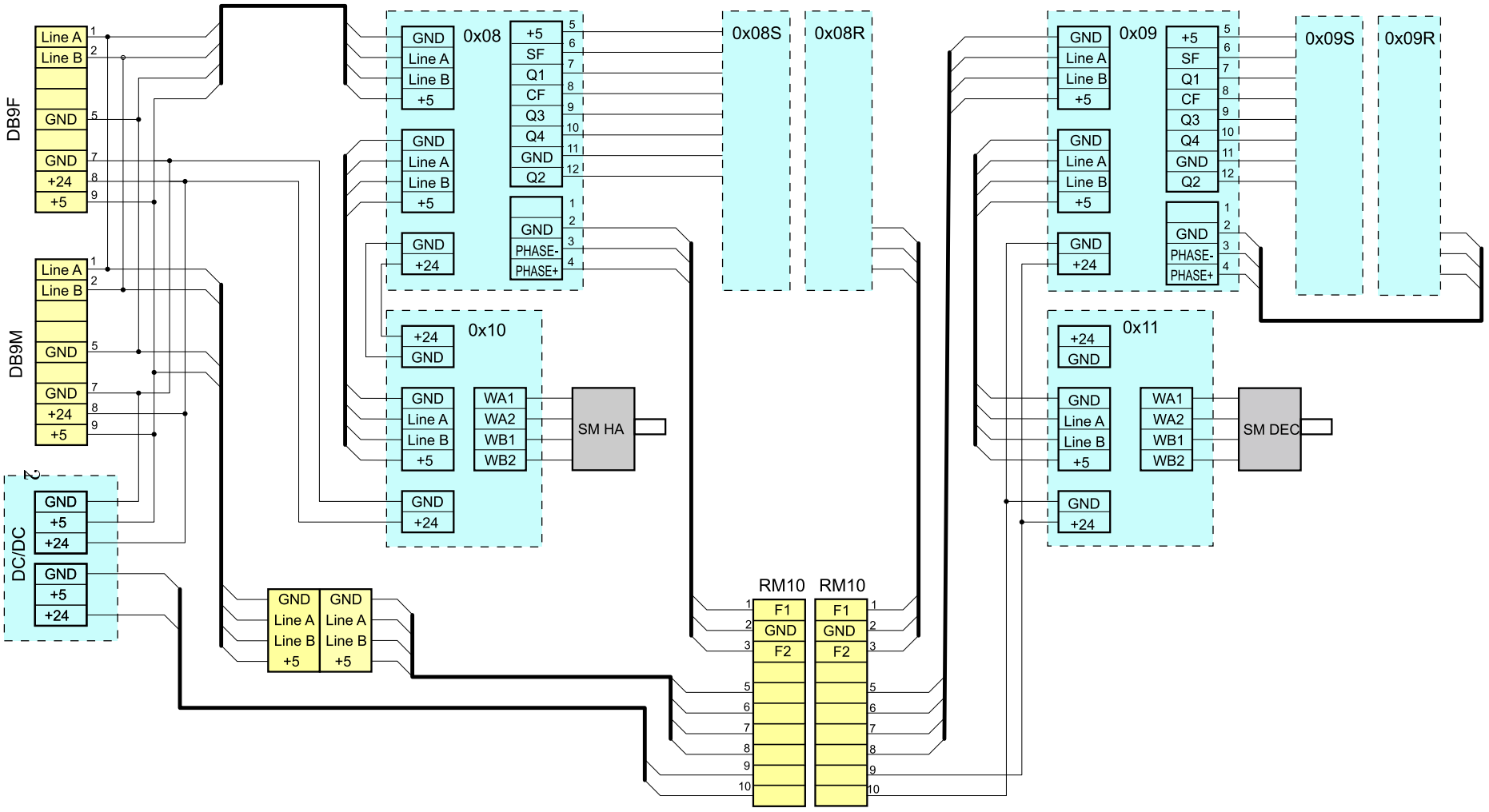

Рис. 1: Принципиальная схема соединений модулей. Модули обозначены их адресами. RM10 — разъем соединяющий подвижную часть монтировки (вилку), на схеме справа, с неподвижной (основанием) — слева.

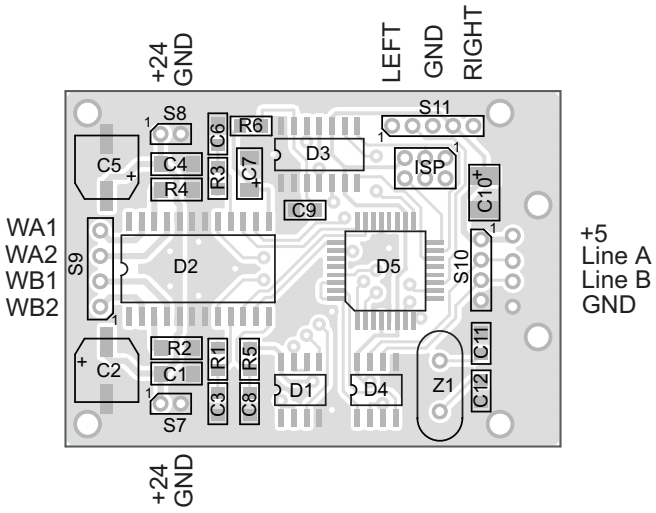

Рис. 2: Расположение элементов на плате контроллера шагового двигателя.

где U_{ref} — опорное напряжение, задаваемое резистором $R6$ в пределах от 2.5 до 4 В, $R_s = R2 = R4$, рекомендуемое значение $0.5 - 1.0\Omega$, установлено сопротивление 0.65Ω, что обеспечивает ток через двигатели в 0.6 А. Полная потребляемая мощность (двигатель+контроллер) составляет 6 Вт.

Следует иметь ввиду, что, поскольку для драйвере PBL3777 не рекомендуется заполнение больше 50%, то максимальный возможный ток определяется из соотношения

$$MaxI_w = 0.5U_D/R_w,$$

где U_D — питающее напряжение (24 В), R_w — сопротивление обмотки шагового мотора (14Ω). Это соотношение приводит к максимальному току 0.86 А.

Алгоритм работы контроллера шагового двигателя

В этом разделе приводятся соотношения между обычными величинами, характеризующими вращение двигателя и внутренними величинами, используемыми для управления микроконтроллером. Описывается версия микрокода контроллера предназначенного для работы в монтировке `Motor_nano.asm`, в котором реализовано как движение с постоянной скоростью (режим слежения), так и движение с заданным ускорением/замедлением для наведения.

Число шагов (микрошагов) на один оборот вала $N = N_m N_f$, где N_m – паспортное число шагов на оборот двигателя (число целых шагов), а N_f – число микрошагов на один шаг. В конфигурации для монтировки $N_m = 200$, а $N_f = 32$.

Текущее положения вала (абсолютная позиция) измеряется в шагах относительно принятого нуль-пункта. Полный диапазон абсолютной позиции равен ± 8388607 (3-х байтовое число). Относительное перемещение задается контроллеру в шагах также в виде знакового 3-х байтового числа.

Скорость движения V в об./с, $r = NV$ в шаг/с. В случае равномерного движения контроллеру передается величина T_c длительности шага в тактах таймера движения:

$$T_c = \frac{1}{\tau V N} \quad , \qquad 0 < T_c < 2^{16} \tag{1}$$

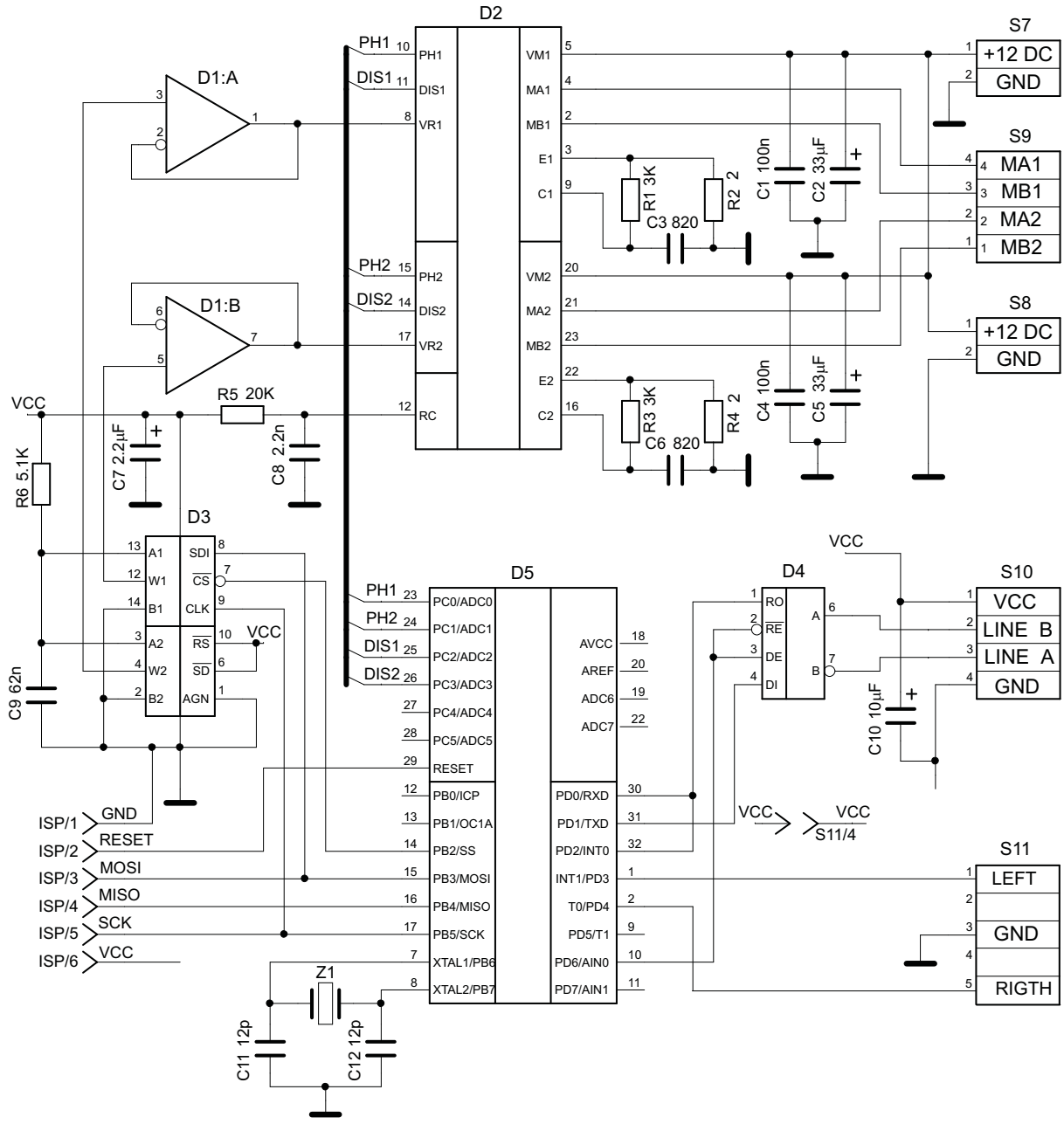

Рис. 3: Принципиальная схема контроллера шагового двигателя.

где τ – длительность такта таймера движения. Эта величина составляет $4.34\mu s$ при тактовой частоте микроконтроллера 14.7456 MHz.

Текущее состояние контроллера отражается в его регистре состояния `STATUS`, доступного только для чтения. Формат регистра состояния для контроллера шагового двигателя следующий:

`Бит 0` – `NEW_STEP`, выполнен новый шаг, но датчики не проверены;
`Бит 1` – `NEW_CYCLE`, цикл тайм-аута обмена;
`Бит 4` – `CLOCKWISE`, направление вращения мотора;
`Бит 5` – `TRACKING`, включен режим слежения;
`Бит 6` – `FORWARD`, текущее направление вращения мотора;
`Бит 7` – `MOTION`, показывает движение мотора.

Состояние датчиков (концевые выключатели) отражается в регистре `SENSOR`, также доступного только для чтения:

`Бит 1` – `LEFT`, левый концевик замкнут. Блокирует движение против часовой стрелки;
`Бит 2` – `RIGHT`, правый концевой датчик замкнут. Блокирует движение по часовой стрелке;

В рабочей конфигурации использование концевых выключателей не предполагается.

Команды контроллера шагового двигателя

В монтировке используются 2 модуля контроллеров шагового двигателя. Как указано выше, им установлены следующие адреса: **0x10** и **0x11**. Все базовые команды (см. например, описание контроллеров для ЗТЭ) реализованы. Набор функциональных команд представляет собой подмножество набора функциональных команд контроллера шагового двигателя для ЗТЭ (микрокод `Motor_tzm.asm`). Краткое описание функциональных команд приведено ниже:

- Останов вращения в режиме слежения (tracking mode):
  `TRACK_STOP (0x80) <- ACY`
  сбрасывает флаг `TRACKING`. Не действует на быстрое движение.

- Запуск вращения по часовой стрелке в режиме слежения:
  `TRACK_CW (0x81) <- ACY | ACW`

- Старт вращения против часовой стрелки в режиме слежения:
  `TRACK_CCW (0x82) <- ACY | ACW`
  Эта и предыдущая команды возвращают сигнал `ACW` если контроллер занят в режиме быстрого движения или установлены флаги `LEFT` или `RIGHT` соответственно.

- Проверка завершения быстрого движения:
  `TEST_FAST (0x83) <- ACY | ACW`
  Сигнал ожидания `ACW` возвращается если движение не завершено.

- Запуск быстрого движения:
  `RUN_FAST (0x84) <- ACY | ACW`
  Сигнал ожидания `ACW` возвращается если предыдущее быстрое движение не завершено или установлены флаги `LEFT` или `RIGHT` в зависимости от направления движения.

- Останов быстрого движения:
  `STOP_FAST (0x86) <- ACY | ACW`
  Сигнал ожидания `ACW` возвращается если быстрое движение уже завершено. Остановка происходит так же равнозамедленно, как и при запланированном завершении.

- Очистка значения абсолютной позиции мотора:
  `CLEAN_ABS (0x88) <- ACY`
  Младшие биты `position`, указывающие такт внутри цикла шагового двигателя не сбрасываются.

- Чтение текущей позиции двигателя `position`:
  `GET_ABS (0xA0) <- byte1(position), byte2(position), byte3(position)`

- Загрузка величины перемещения `shift` для быстрого движения:
  `LOAD_SHIFT (0x03) byte1(shift), byte2(shift), byte3(shift) <- ACY`

- Чтение величины перемещения:
  `READ_SHIFT (0xA1) <- byte1(shift), byte2(shift), byte3(shift)`

- Установка скорости слежения `speed`:
  `LOAD_SPEED (0x32) low(speed), high(speed) <- ACY | ACW`
  Величина `speed` выражена в тактах таймера движения. Если в этот момент слежение выполняется, то изменение скорости произойдет на очередном шаге. Быстрое движение блокирует изменение скорости и других параметров (возвращается `ACW`)

- Чтение текущей скорости слежения `speed`:
  `READ_SPEED (0xF2) <- low(speed), high(speed)`

- Установка пути разгона при быстром движении `path`:
  `SET_PATH (0x34) low(path), high(path) <- ACY | ACW`

- Чтение установленного пути разгона:
  `READ_PATH (0xF4) <- low(path), high(path)`

- Установка ускорения `accel` разгона быстрого движения:
  `SET_ACCEL (0x36) low(accel), high(accel) <- ACY | ACW`

- Чтение ускорения разгона:
  `READ_ACCEL (0xF6) <- low(accel), high(accel)`

- Установка значения тайм-аута обмена `sleepdown` :
  `SET_SLEEP (0x3A) low(sleepdown), high(sleepdown) <- ACY | ACW`

- Чтение значения тайм-аута обмена `sleepdown` :
  `READ_SLEEP (0xFA) <- low(sleepdown), high(sleepdown)`

  Это значение выражено в 10 ms интервалах. По умалчанию установлено значение 60 сек.

Для примера, с использованием базовых функций обмена, находящихся в программном модуле `exchange.cpp`, функциональные команды могут быть вызваны следующим образом (в порядке их описания):

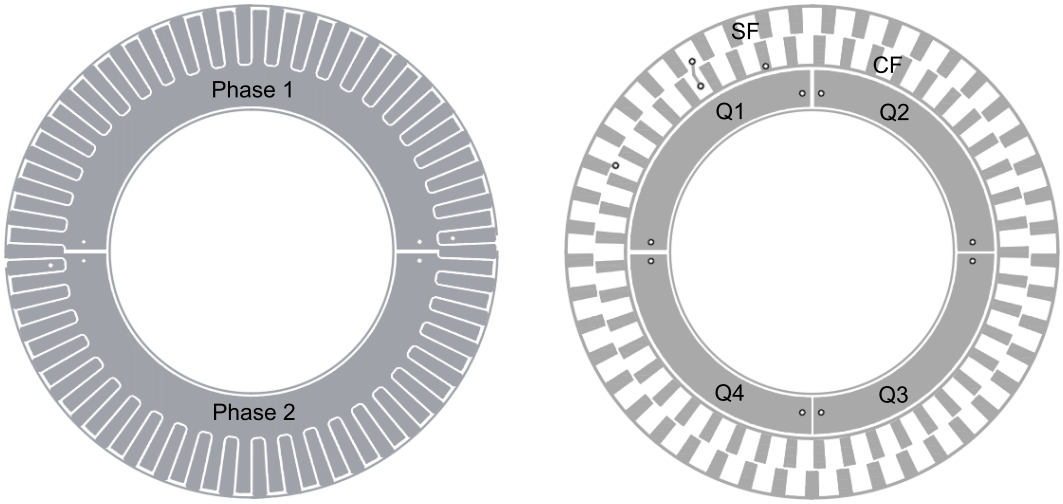

Рис. 4: Геометрия емкостного двух-отсчетного датчика угла. Слева — ротор датчика, справа — статор. Объяснения в тексте.

```
RS485::sendSimpleCommand( address, TRACK_STOP )
RS485::sendSimpleCommand( address, TRACK_CW )
RS485::sendSimpleCommand( address, TRACK_CCW )
RS485::sendSimpleCommand( address, TEST_FAST )
RS485::sendSimpleCommand( address, RUN_FAST )
RS485::sendSimpleCommand( address, STOP_FAST )
RS485::sendSimpleCommand( address, CLEAN_ABS )
RS485::askLong( address, GET_ABS, &(int)current_position )
RS485::sendData( address, LOAD_SHIFT, (unsigned char*)&shift )
RS485::askLong( address, READ_SHIFT, &shift )
RS485::sendWordCommand( address, LOAD_SPEED, (short)speed )
RS485::askWord( address, READ_SPEED, &(short)speed )
RS485::sendWordCommand( address, SET_PATH, (short)path )
RS485::askWord( address, READ_PATH, &(short)path )
RS485::sendWordCommand( address, SET_ACCEL, (short)accel )
RS485::askWord( address, READ_ACCEL, &(short)accel )
```

Команды контроллера датчика угла

В монтировке приемнены емкостные двух-отсчетные датчики угла. Их основой служат ротор и статор, выполненые на круглых печатных платах. Геометрия грубого и точного каналов приведена на Рис. 4. На две обкладки ротора подается с контроллера синусоидальный сигнал Phase 1 и противофазный ему сигнал Phase 2. С обкладок статора снимаются 4 сигнала грубого канала Q1, Q2, Q3 и Q4 и 2 сигнала точного канала — SF и сдвинутый относительно него на 1/4 периода сигнал CF.

В монтировке используются 2 модуля контроллеров датчиков угла. Как указано выше, им установлены следующие адреса: **0x08** и **0x09**. Все базовые команды (см. например, описание контроллеров для ЗТЭ) реализованы. Набор функциональных команд (микрокод `Capsin.asm` и `Capsin.inc`) содержит только одну команду:

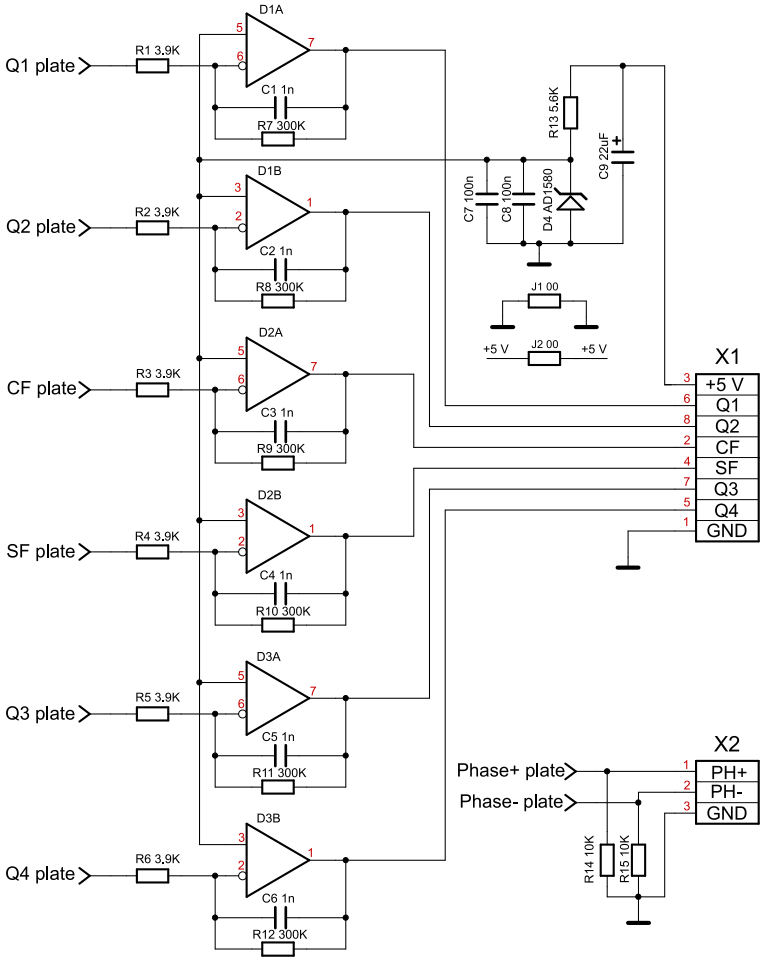

Рис. 5: Принципиальная схема статора и ротора датчика угла.

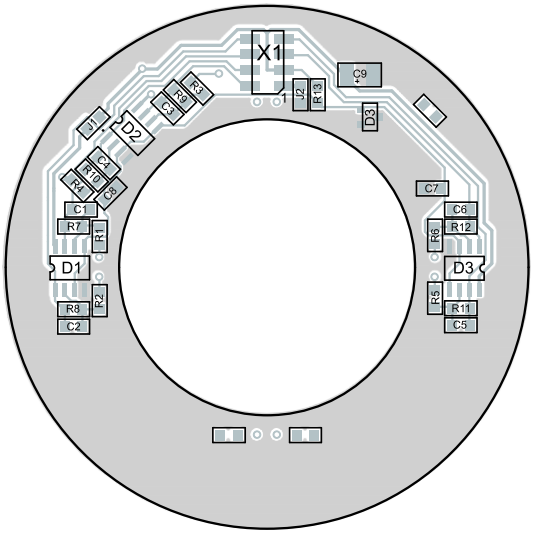

Рис. 6: Монтажная схема статора датчика угла со стороны компонент.

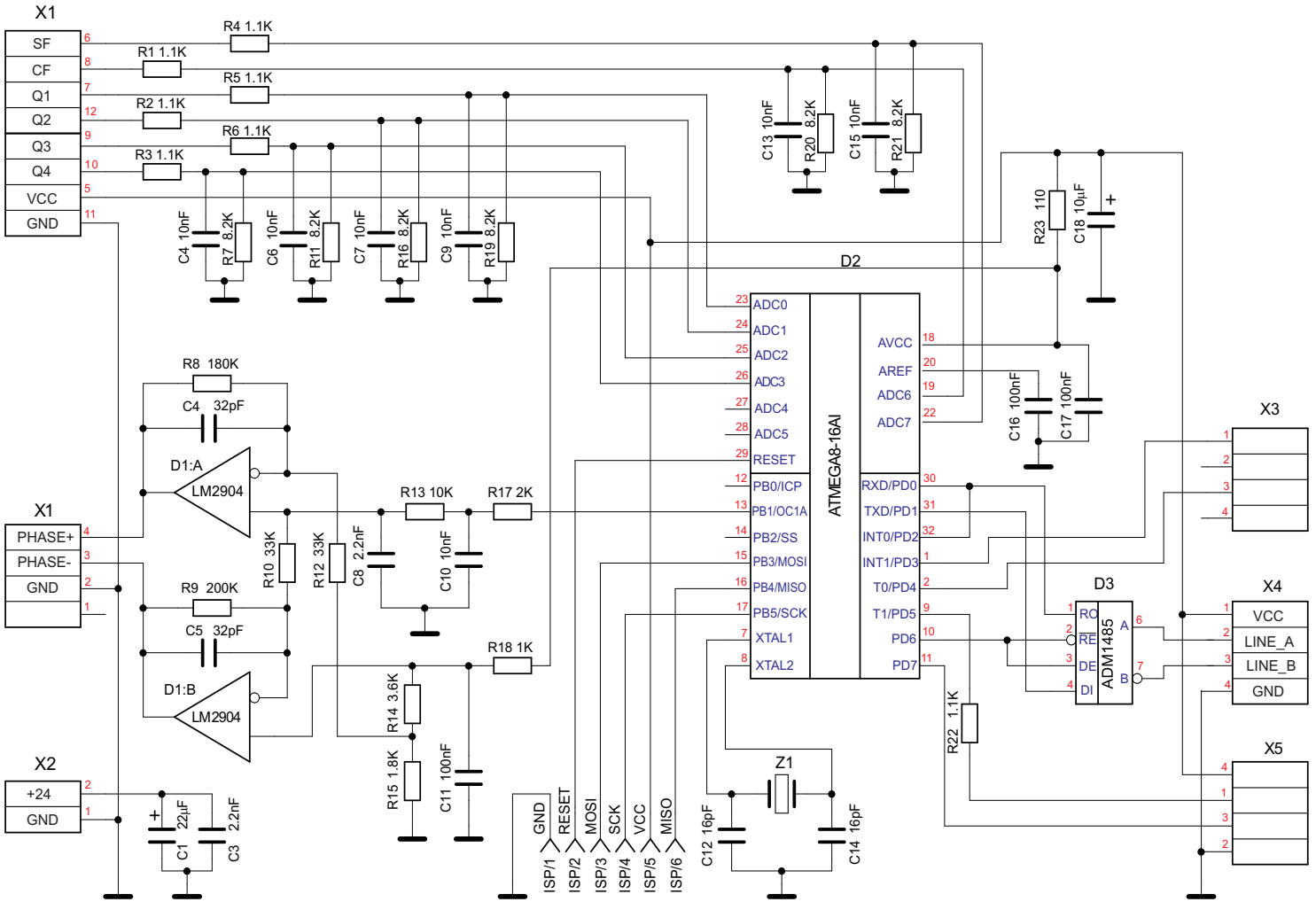

Рис. 7: Принципиальная схема контроллера датчика угла.

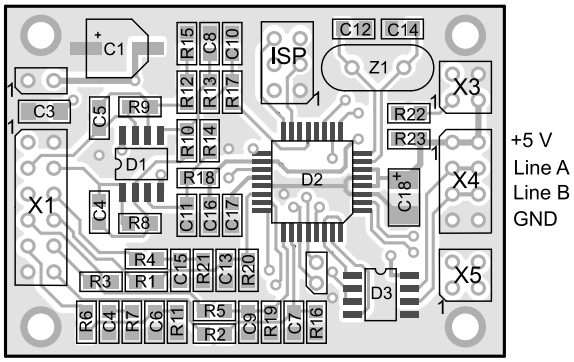

Рис. 8: Размещение компонент на печатной плате контроллера датчика угла.

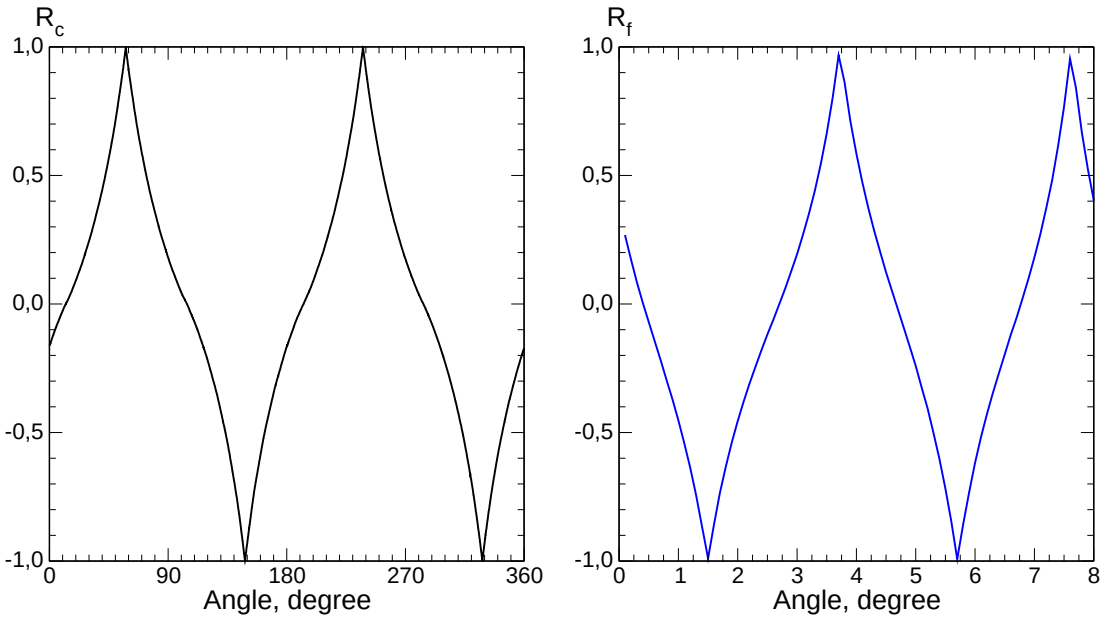

Рис. 9: Пример измеренной зависимости величины R_c грубого канала и R_f точного канала от угла поворота датчика.

- Чтение амплитуд 4-х грубых (Q1, Q2, Q3, Q4) и 2-х точных (SF, CF) отсчетов:
  `READ_AMPL (0xA0) <- word(Q1), word(Q2), word(Q3), word(Q4), word(SF), word(CF)`

С использованием базовых функций обмена, находящихся в программном модуле `exchange.cpp`, эта команда вызывается следующим образом:

```
RS485::askData( address, READ_AMPL, (unsigned char*)amplitudes ),
```

где `amplitudes` — массив из 6 величин типа `short`. Вызов возвращает правильное значение равное 6.

Сигналы грубых отсчетов имеют трапецевидную форму с периодом равным 360° и последовательно смещенные на $\approx 90°$. Для дальнейших вычислений грубого угла используется минимизирующее влияние различного рода ошибок отношение двух линейно независимых комбинаций сигналов: $U1 = Q1 + Q2 - Q3 - Q4$ и $U2 = Q1 - Q2 - Q3 + Q4$. Эти сигналы имеют практически правильную треугольную форму.

Их отношение

$$R_c = \frac{U1}{U2}, \quad \text{если} \quad U1 < U2 \quad \text{или} \quad R = \frac{U2}{U1}, \quad \text{если} \quad U1 \geq U2$$

представляет собой нормированный сигнал из значения которого и вычисляется грубое значение угла поворота. Для определения правильной четверти требуется использование информации о знаках сигналов $U1$ и $U2$.

Для уточнения значения угла поворота используется отношение R_f сигналов $F1$ и $F2$ точных отсчетов, имеющих форму близкую к синусоидальной. Это отношение также слабочувствительно к изменениям внешних условий. Сигналы точного канала имеют период 8°, т.е. в 45 раз меньше чем период сигналов грубого канала. Измеренные значения отношений R_c и R_f приведены на Рис. 9. Обе кривые имеют сложную зависимость от угла, поэтому использование фиксированной математической модели дает большую погрешность, чем табличное описание, выполненное по результатам калибровки.

Контроллер мотора крыши и его команды

Это такой же контроллер как и тот, что используется в системе "МАСТЕР" для открывания главной крыши и "курятника". Соответственно, для работы можно использовать уже имеющиеся скрипты. Исходный микрокод несколько модифицирован, чтобы обеспечить автоматическое закрывание крыши в случае отсутствия обмена в течение интервала sleepdown. Набор команд (микрокод `Motor_kroof.asm` и `Motor_kroof.inc`) несколько отличается от набора команд контроллера зажимов тормозов телескопа ЗТЭ, модификацией которого является данный контроллер. Модулю установлен адрес **0x0E**.

На Рис. 11 приведена монтажная схема контроллера, где обозначены правильные подключения.

- Проверка завершения движения канала A:
  `TEST_MOTION (0x83) <- ACY | ACW`
  Сигнал ожидания `ACW` возвращается если движение не завершено.

- Проверка завершения движения канала B:
  `TEST_MOTION (0x84) <- ACY | ACW`
  Сигнал ожидания `ACW` возвращается если движение не завершено.

- Аварийный останов всех двигателей:
  `STOP_MOTION (0x85) <- ACY`

- Запуск вращения двигателя канала A в сторону концевого выключателя "зажато":
  `TIGHT_A (0x88) <- ACY | ACW`

- Запуск вращения двигателя канала A в сторону концевого выключателя "отжато":
  `LOOSE_A (0x89) <- ACY | ACW`

- Запуск вращения двигателя канала B в сторону концевого выключателя "зажато":
  `TIGHT_B (0x8A) <- ACY | ACW`

- Запуск вращения двигателя канала B в сторону концевого выключателя "отжато":
  `LOOSE_B (0x8B) <- ACY | ACW`

- Разблокировка запуска двигателя канала A:
  `LOCK_A (0x8C) <- ACY`
- Разблокировка запуска двигателя канала B:
  `LOCK_A (0x8D) <- ACY`
- Блокировка запуска двигателя канала A:
  `LOCK_A (0x8E) <- ACY`
- Блокировка запуска двигателя канала B:
  `LOCK_A (0x8F) <- ACY`
- Чтение текущего значения предельного времени движения `uptime`:
  `READ_TLIMIT (0xF2) <- low(uptime) high(uptime)`
- Установка значения предельного времени движения `uptime`:
  `LOAD_TLIMIT (0x32) low(uptime) high(uptime) <- ACY`
- Установка значения тайм-аута обмена `sleepdown` :
  `SET_SLEEP (0x36) low(sleepdown), high(sleepdown) <- ACY | ACW`
- Чтение значения тайм-аута обмена `sleepdown` :
  `READ_SLEEP (0xF6) <- low(sleepdown), high(sleepdown)` Это значение выражено в 5 ms интервалах. По умолчанию установлено значение 60 сек.
- Чтение текущей температуры `temp`:
  `READ_TEMP (0xEC) <- temp`
  Температура `T` в градусах связана с получаемым значением `temp` соотношением `T = 250×temp/256`.
- Чтение напряжения питания `mainpower` двигателя:
  `READ_UCC (0xED) <- mainpower`
  Значение напряжения в Вольтах вычисляется по формуле `Ucc = 48×mainpower/256`.

Текущее состояние контроллера отражается в его регистре состояния `STATUS`, доступного только для чтения. Формат регистра состояния для контроллера двигателя крыши следующий:

`Бит 0` – `NEW_CHECK`, сигнад для проверки концевых выключателей;
`Бит 1` – `TIME_OUT`, основ предыдущего движения произошел по тайм-ауту;
`Бит 2` – `UNLOCK_B`, блокировка движения мотора B, 1 — блокировка снята;
`Бит 3` – `UNLOCK_A`, блокировка движения мотора A, 1 — блокировка снята;
`Бит 4` – `FORWARD_B`, направление вращения мотора B;
`Бит 5` – `FORWARD_A`, направление вращения мотора A;
`Бит 6` – `MOTION_B`, показывает движение мотора канала B;
`Бит 7` – `MOTION_A`, показывает движение мотора канала A.

Состояние датчиков (концевые выключатели) отражается в регистре `SENSOR`, также доступного только для чтения:

`Бит 0` – `A CLOSE`, концевик "Закрыто" канала A замкнут. Блокирует закрывание крыши;
`Бит 1` – `B CLOSE`, аналогично, в рабочей конфигурации не используется;
`Бит 4` – `A OPEN`, концевик "Открыто" канала A замкнут. Блокирует открывание крыши;
`Бит 5` – `B OPEN`, аналогично, в рабочей конфигурации не используется;

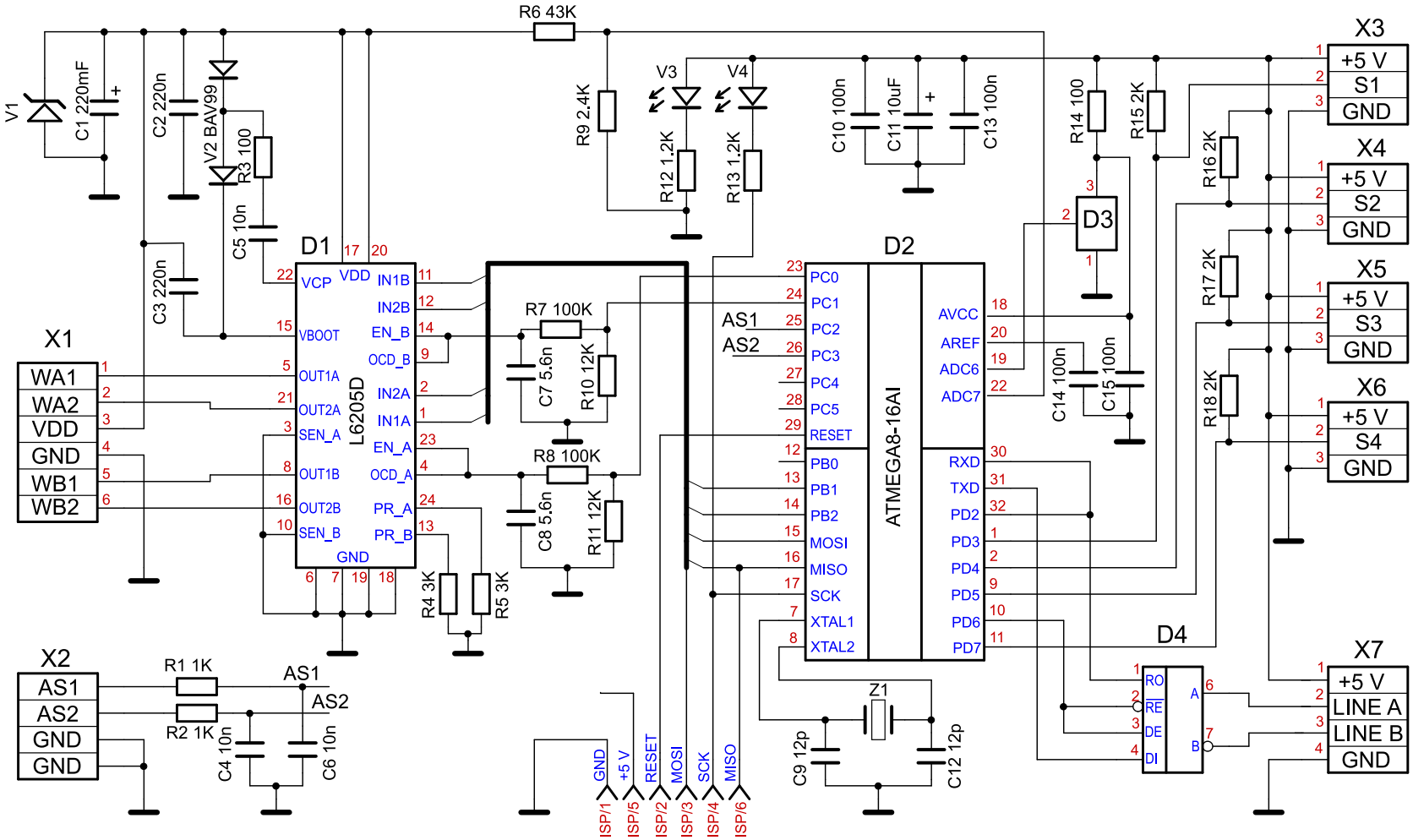

Рис. 10: Принципиальная схема контроллера двигателя постоянного тока для крыши.

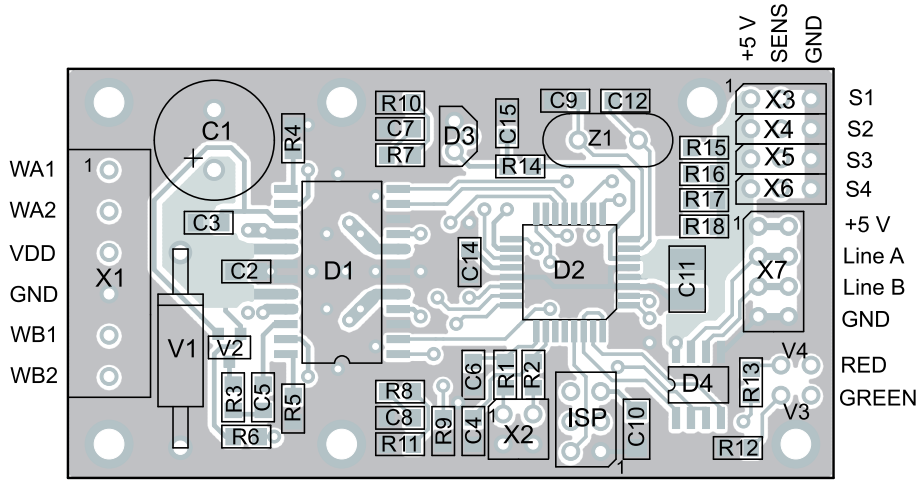

Рис. 11: Монтажная схема контроллера двигателя постоянного тока для крыши.