

Контроллеры телескопа МАСТЕР-2

Версия 2

В.Корнилов, ГАИШ

11 февраля 2011 г.

Краткое описание дополнительных узлов телескопа

Телескоп МАСТЕР-2 представляет собой 2 оптические трубы системы Гамильтона (диаметр входного зрачка 400 мм при относительном отверстии F/2.5), установленных параллельно на монтировке NTM500 по разные стороны полярной оси. Конструкция узла подвески труб предусматривает возможность работы в двух положениях: 1) оптические оси труб строго коллинеарны и 2) оптические оси смещены относительно исходного положения на $\approx 1^\circ$ по часовому углу на запад для западной трубы и на восток для восточной трубы.

Переключение положений труб производится линейным актюатором фирмы HIWIN LAS3-1-1-50-24 или фирмы SKF САНВ-10-В4А-050-192-ААА-Р00-000. В качестве датчика обратной связи используется вмонтированный в актюатор потенциометр 10К. Контроллер актюатора расположен в корпусе, закрепленном на западной интерфейсной пластине монтировки. В этом же корпусе расположен вторичный источник питания +5 В 400 мА для питания всех микроконтроллеров телескопа. Также корпус используется в качестве разветвителя линии RS485 и питания между остальными контроллерами.

На передних концах оптических труб телескопа (в районе входного корректора) закреплены контроллеры узлов фокусировки. В первом варианте, установленном в Коуровской обсерватории, движение узла линзы поля осуществляется мотор-редуктором 1516T012SR + 15AK 152:1 фирмы Faulhaber, в дальнейшем планируется везде использовать мотор-редуктор с энкодером 1516A012SR IE2-16 + 15AK-809:1.

Контроллер переключения фильтров вмонтирован непосредственно в фотометрические блоки, установленные в прямых фокусах оптических труб, и несущие приемные ПЗС камеры. Этот контроллер осуществляет ввод/вывод в пучок 4-х разных светофильтров.

Еще одним компонентом является контроллер 12-футового купола Astrohaven, установленный вместо штатного и предназначенный для управления южным и северным приводами открывания/закрывания створок. В приводах применены однофазные моторы переменного тока компании Leeson: модель С6С17NK19D, каталожный номер 114165.00. Основные дан-

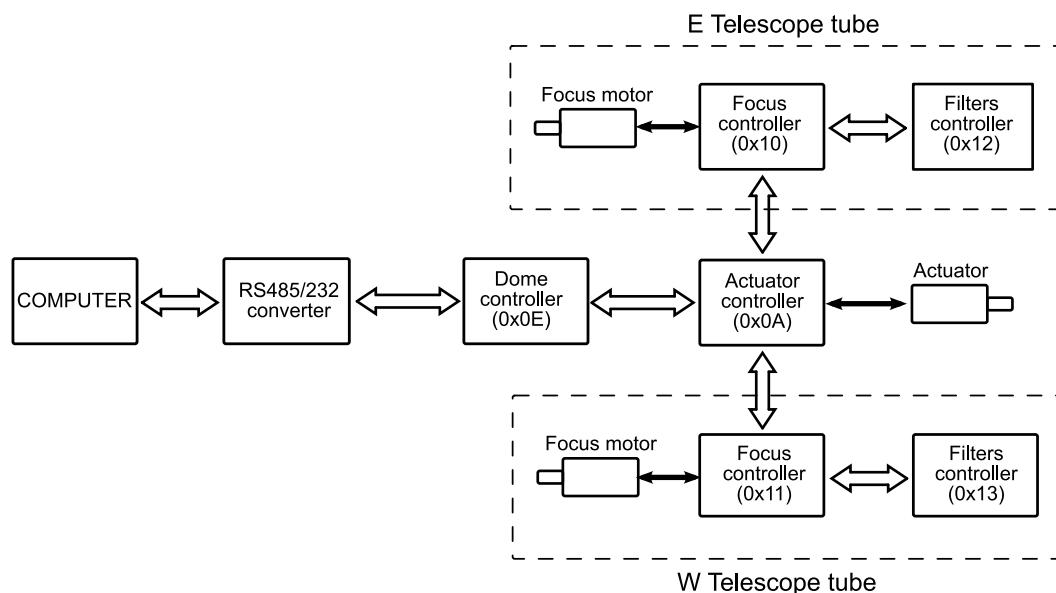


Рис. 1: Структурная схема дополнительной электроники телескопа МАСТЕР-2. Пустые стрелки — соединения DB9M/DB9F, сплошные — соединения через разъемы PC7

ные мотора приведены в приложении. Редуктор привода 1:60 обеспечивает момент на валу барабана 80 Нм.

Последние элементы системы управления — это блок питания DR-75-24 (+24 В 3.1 А) фирмы Mean Well, размещенный на стойке в помещении или в ящике под куполом и конвертер RS485/RS232, непосредственно подключенный к COM-порту управляющего компьютера.

Подключение контроллеров и других модулей

Контроллеры запрограммированы на скорость обмена 115Кбит/с с тем, чтобы можно было использовать конвертер RS485/COM. Принципиальная схема соединений между отдельными модулями показана на Рис. 1. Для передачи данных по линии RS485 используется простой компактный протокол, описанный, например, в http://dragon.sai.msu.ru/mass/download/doc/new_details.pdf. Этот протокол подразумевает адресацию 32 модулей в диапазоне адресов 0x00 - 0x1F, что согласуется с нагрузочной способностью используемых приемо-передатчиков ADM1485 компании Analog Devices. Контроллеры системы имеют следующие 16-ричные адреса (они, конечно, при необходимости могут быть изменены)

- Контроллер купола — 0x0E
- Контроллер коллимации оптических труб — 0x0A
- Контроллер фокусировки западной трубы — 0x11
- Контроллер фокусировки восточной — 0x10
- Контроллер фильтров западной трубы — 0x12
- Контроллер фильтров восточной трубы — 0x13

Все соединения между отдельными модулями выполнены на разъемах DB9F (входные) и DB9M (выходные) кабелем SFTP-4 (4 экранированные многожильные пары). Распайка разъемов и кабелей указана в Таблице 1

Таблица 1: Распайка кабелей и разъемов DB9

Сигнал	Контакт разъема	Цвет провода
RS485 линия А	1	зелено-белый
RS485 линия В	2	зеленый
RS485 общий	5	оранжево-белый + экран
RS485 питание +5	9	оранжевый
Питание +24	8	коричневый + коричнево-белый
Питание общий	7	синий + сине-белый

Источник питания +24 В подсоединяется на первом разъеме длинного сегмента кабеля отдельным шнуром $2 \times 0.5\text{мм}^2$ без распайки на контакты 7 и 8. Этот разъем непосредственно подключается к конвертеру RS485/RS232.

Для подключения линейного актюатора и фокусируемых моторов применяются разъемы РС7. Их распайка следующая:

Таблица 2: Распайка разъемов PC7

Сигнал	Контакт разъема актюатора	Цвет провода актюатора		Контакт разъема фокусировки
		HIWIN	SKF	
Мотор +	1	белый	черный	1
Мотор –	2	черный	красный	2
Общий	3	синий	белый	3
Сигнал с потенциометра	4	желтый	фиолетовый	—
Питание +5	5	красный	желтый	5
Сигнал энкодера А	—			6
Сигнал энкодера В	—			7

Базовый набор команд

Это общие для всех модулей команды. В приведенных ниже командах, так же как и в последующих разделах, адрес модуля опущен. Реально он должен предварять любую команду.

- Программный сброс:
`RESET (0x87) <- ACY,`
 Полностью инициализирует модуль. Также служит для перевода модуля из режима загрузчика (boot mode) в рабочий режим (application mode). В зависимости от программирования fuse битов микроконтроллера, после включения питания модуль может находиться в режиме загрузчика или может быть переведен в этот режим следующей командой.
- Переход в режим загрузчика для перепрограммирования рабочей микропрограммы.
`REPROGRAMMING (0xC0) <- ACY,`
 Подробнее — смотри описание утилиты `avrup`
- Чтение статусного регистра `STATUS` модуля. Конкретный смысл флагов зависит от модуля:
`GET_STATUS (0xE0) <- status`
- Чтение регистра датчиков `SENSOR` модуля:
`GET_SENSOR (0xE1) <- sensors`
- Запрос идентификации модуля:
`GET_IDENT (0xA2) <- id1 id2 id3 id4`
 Возвращает уникальную идентификацию модуля `id1 id2` и дату программирования кода `id3 id4`. Дата программирования кода обновляется при перепрограммировании при помощи утилиты `avrup`
- Запрос аппаратных констант модуля:
`GET_CONST (0xA3) <- const1 const2 const3 const4`
 Смысл констант `const1` и `const2` зависит от конкретного модуля, константы `const3 = low(freq)`, `const4 = high(freq)`, где `freq` — тактовая частота микроконтроллера в КГц.
- Загрузка контрольного ключа:
`LOAD_KEY (0x3E) low(key) high(key) <- ACY`
- Чтение контрольного ключа:
`READ_KEY (0xFE) <- low(key) high(key),`

Контрольный ключ используется управляющей программой для обнаружения ситуаций выключения питания или сброса модуля после перезапуска самой программы.

- Чтение произвольного участка памяти микроконтроллера:

`DUMP_MEMORY (0xAF) <- 16 bytes.`

Эта команда для отладочных целей или нестандартного использования модуля. Значение, загруженное в контрольный ключ, используется как начальный адрес памяти, включая регистры общего назначения, регистры ввода/вывода и собственно памяти (смотри описание AVR микроконтроллеров).

Команды низкого уровня можно подавать с управляющей машины используя утилиту `lmoх`. Если в системе все правильно: загружен драйвер `rs486com` или `rs486lpt` в зависимости от используемого порта и соответствующего конвертера, существует устройство `/dev/rs486com0` (или `/dev/rs486lpt0` для конвертера `parport/rs485`) установлены правильные права доступа (пользователь должен принадлежать к группе `uucp`), сам конвертер вставлен в соответствующий порт и подано напряжение питания на линию, появится приглашающая подсказка в виде символа `<`. В противном случае будет выдано сообщение об ошибке и утилита завершит работу.

После появления приглашения можно подать команду, на которую модуль обязательно должен ответить, например, сброс модуля актюатора будет выглядеть так:

```
<0a 87
```

```
Signal ACY
```

Запрос идентификации модуля контроллера купола

```
<0e a2
```

```
Data 4 bytes: 0e 07 0b 0d
```

Все числа шестнадцатеричные, регистр значения не имеет. Для выхода из утилиты `lmoх` достаточно нажать `ENTER`.

Контроллер линейного актюатора

Этот контроллер предназначен для переключения коллимации оптических труб. В силу ограниченности выполняемых функций его набор команд (микрокод `Master_interf.asm` и `Master_interf.inc`) редуцирован по сравнению с набором командами контроллера крыши, на базе которого настоящий модуль был разработан. Модулю установлен адрес **0x0A**.

Устройство не имеет в своем составе концевых выключателей. Вместо этого используется сигнал с потенциометра актюатора, линейно связанный с величиной выдвигания штока актюатора. При правильно установленных пределах движение штока осуществляется в диапазоне от нижнего `limit_low` до верхнего предела `limit_high` и обратно. Загружаемые значения должны быть получены непосредственным образом: при установке штока в желаемое положение команда чтения текущего положения потенциометра `GET_POSITION` возвращает значение, которое должно быть загружено на место нижнего или верхнего предела.

Модуль поддерживает все общие базовые команды компактного протокола и следующие функциональные команды:

- Запуск выдвижения штока актюатора
OPEN (0x80) <- ACY | ACW Сигнал ожидания ACW возвращается если движение не завершено или заблокировано.
- Запуск вдвижения штока актюатора
CLOSE (0x81) <- ACY | ACW Сигнал ожидания ACW возвращается если движение не завершено или заблокировано.
- Проверка завершения движения актюатора
TEST_MOTION (0x83) <- ACY | ACW
Сигнал ожидания ACW возвращается если движение не завершено.
- Аварийный останов актюатора
STOP_MOTION (0x85) <- ACY
- Разблокировка движения актюатора
UNLOCK (0x8C) <- ACY
- Блокировка движения актюатора
LOCK (0x8E) <- ACY
- Чтение текущего значения предельного времени движения uptime:
READ_TLIMIT (0xF2) <- low(uptime) high(uptime)
Значение uptime задается в 1/200 с.
- Установка значения предельного времени движения uptime:
LOAD_TLIMIT (0x32) low(uptime) high(uptime) <- ACY
- Чтение текущего значения нижнего предела limit_low:
READ_LOW (0xF4) <- low(limit_low) high(limit_low)
- Установка значения нижнего предела limit_low:
LOAD_LOW (0x34) low(limit_low) high(limit_low) <- ACY
- Чтение текущего значения верхнего предела limit_high:
READ_HIGH (0xF6) <- low(limit_high) high(limit_high)
- Установка значения верхнего предела limit_high:
LOAD_HIGH (0x36) low(limit_high) high(limit_high) <- ACY
- Чтение текущего положения потенциометра position:
GET_POSITION (0xF8) <- low(position) high(position)
Нижний и верхний пределы выражены в тех же самых условных единицах, что и позиция потенциометра (штока).
- Чтение текущей температуры temp:
READ_TEMP (0xFФ) <- low(temp) high(temp)
Температура T в градусах связана с получаемым значением temp соотношением $T = 250 \times \text{temp} / 256$.
- Чтение напряжения питания mainpower двигателя:
READ_UCC (0xFC) <- low(mainpower) high(mainpower)
Значение напряжения в Вольтах вычисляется по формуле $U_{cc} = 48 \times \text{mainpower} / 256$.

Текущее состояние контроллера отражается в его регистре состояния STATUS, доступного только для чтения. Его формат следующий:

- Бит 0 – NEW_CHECK, сигнал для проверки текущего положения;
- Бит 1 – TIME_OUT, основ предыдущего движения произошел по тайм-ауту;
- Бит 2 – NEW_DATA, сигнал для обработки данных с АЦП;

- Бит 4 – UNLOCK, блокировка движения, 1 – блокировка снята;
- Бит 5 – FORWARD, направление движения актюатора, 1 – шток втягивается;
- Бит 7 – MOTION, показывает движение актюатора.

Состояние датчиков (или флагов состояния) отражается в регистре **SENSOR**, также доступного только для чтения:

- Бит 0 – CLOSE, флаг, блокирующий втягивание штока. Устанавливается когда **position** становится меньше **limit_low**;
- Бит 4 – OPEN, флаг, блокирующий выдвигание штока. Устанавливается когда **position** становится больше **limit_high**;

Контроллер мотора фокусировки

Этот контроллер предназначен для управлением DC мотором узла фокусировки оптической трубы. Микрокод модуля содержится в файлах **Master_focus.asm** и **Master_focus.inc**). Модулям установлены адреса **0x10** и **0x11**.

Модуль поддерживает все общие базовые команды компактного протокола и следующие функциональные команды:

- Останов двигателя
STOP (0x80) <- ACY | ACW
- Запуск движения в сторону трубы
MOVE_IN (0x81) <- ACY | ACW Сигнал ожидания **ACW** возвращается если движение не завершено.
- Запуск движения от трубы
MOVE_OUT (0x82) <- ACY | ACW Сигнал ожидания **ACW** возвращается если движение не завершено. Заметим, что в этих командах направление движения условное. Реальное направление будет зависеть от правильности распайки переходного разъема.
- Проверка завершения движения мотора
TEST_MOTION (0x83) <- ACY | ACW
Сигнал ожидания **ACW** возвращается если движение не завершено.
- Очистка счетчика положения фокуса
CLEAR (0x88) <- ACY | ACW
Сигнал ожидания **ACW** возвращается если уже происходит движение.
- Установка быстрой скорости движения фокуса
FAST (0x89) <- ACY | ACW
Сигнал ожидания **ACW** возвращается если уже происходит движение.
- Установка медленной скорости движения фокуса
FAST (0x8A) <- ACY | ACW
Сигнал ожидания **ACW** возвращается если уже происходит движение.
- Перемещение на величину сдвига **shift** со знаком:
SHIFT (0x52) low(shift) high(shift) <- ACY | ACW Сигнал ожидания **ACW** возвращается если уже происходит движение. Команда работает, если установлен фокусировочный мотор с энкодером. Тогда сдвиг **shift** и **position** измеряются в единицах 0.93 мкм.

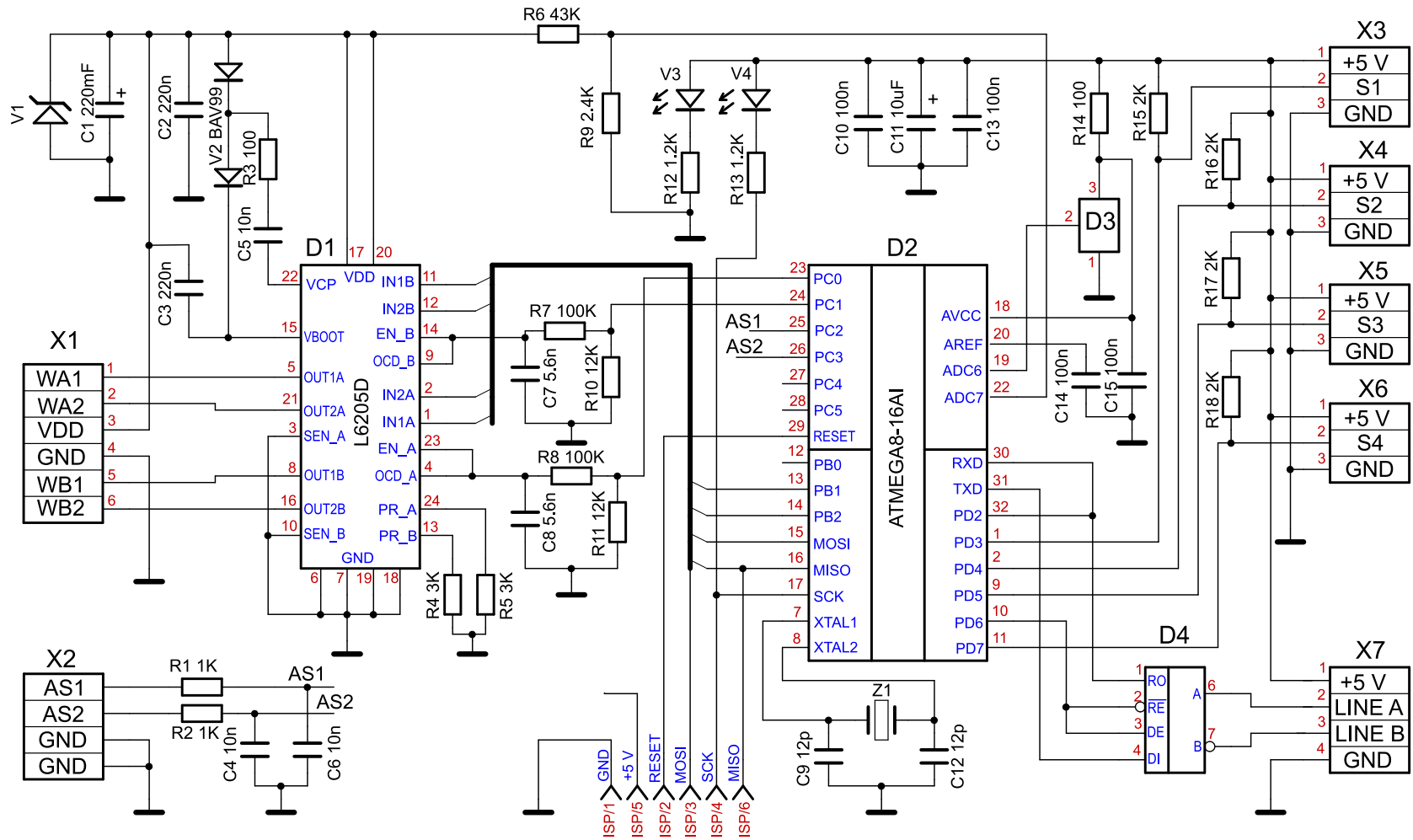


Рис. 2: Принципиальная схема контроллера актюатора.

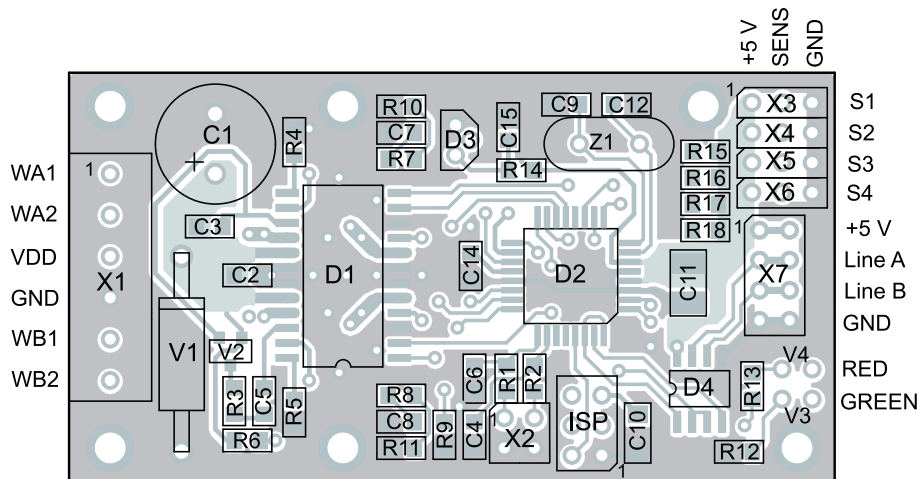


Рис. 3: Монтажная схема контроллера актюатора.

- Чтение текущего значения предельного времени движения `uptime`:
`READ_TLIMIT (0xF2) <- low(uptime) high(uptime)`
Значение `uptime` задается в $1/200$ с.
- Установка значения предельного времени движения `uptime`:
`LOAD_TLIMIT (0x32) low(uptime) high(uptime) <- ACY`
- Чтение текущего положения `position`:
`READ_HIGH (0xF6) <- low(position) high(position)`
- Чтение текущей температуры `temp`:
`READ_TEMP (0xF8) <- low(temp) high(temp)`
Температура T в градусах связана с получаемым значением `temp` соотношением $T = 250 \times temp / 256$.
- Чтение напряжения питания `mainpower` двигателя:
`READ_UCC (0xFA) <- low(mainpower) high(mainpower)`
Значение напряжения в Вольтах вычисляется по формуле $U_{cc} = 48 \times mainpower / 256$.

Текущее состояние контроллера отражается в его регистре состояния `STATUS`, доступного только для чтения. Его формат следующий:

- Бит 0 – `NEW_CHECK`, сигнал для проверки текущего положения;
- Бит 1 – `TIME_OUT`, останов предыдущего движения произошел по тайм-ауту;
- Бит 2 – `NEW_DATA`, сигнал для обработки данных с АЦП;
- Бит 5 – `SLOW`, установлена медленная скорость движения мотора;
- Бит 6 – `FORWARD`, направление движения мотора;
- Бит 7 – `MOTION`, показывает движение мотора.

Состояние датчиков (или флагов состояния) отражается в регистре `SENSOR`, также доступного только для чтения:

- Бит 0 – `EXTER`, резервный флаг для внутреннего концевика;
- Бит 1 – `INTER`, резервный флаг для внешнего концевика;

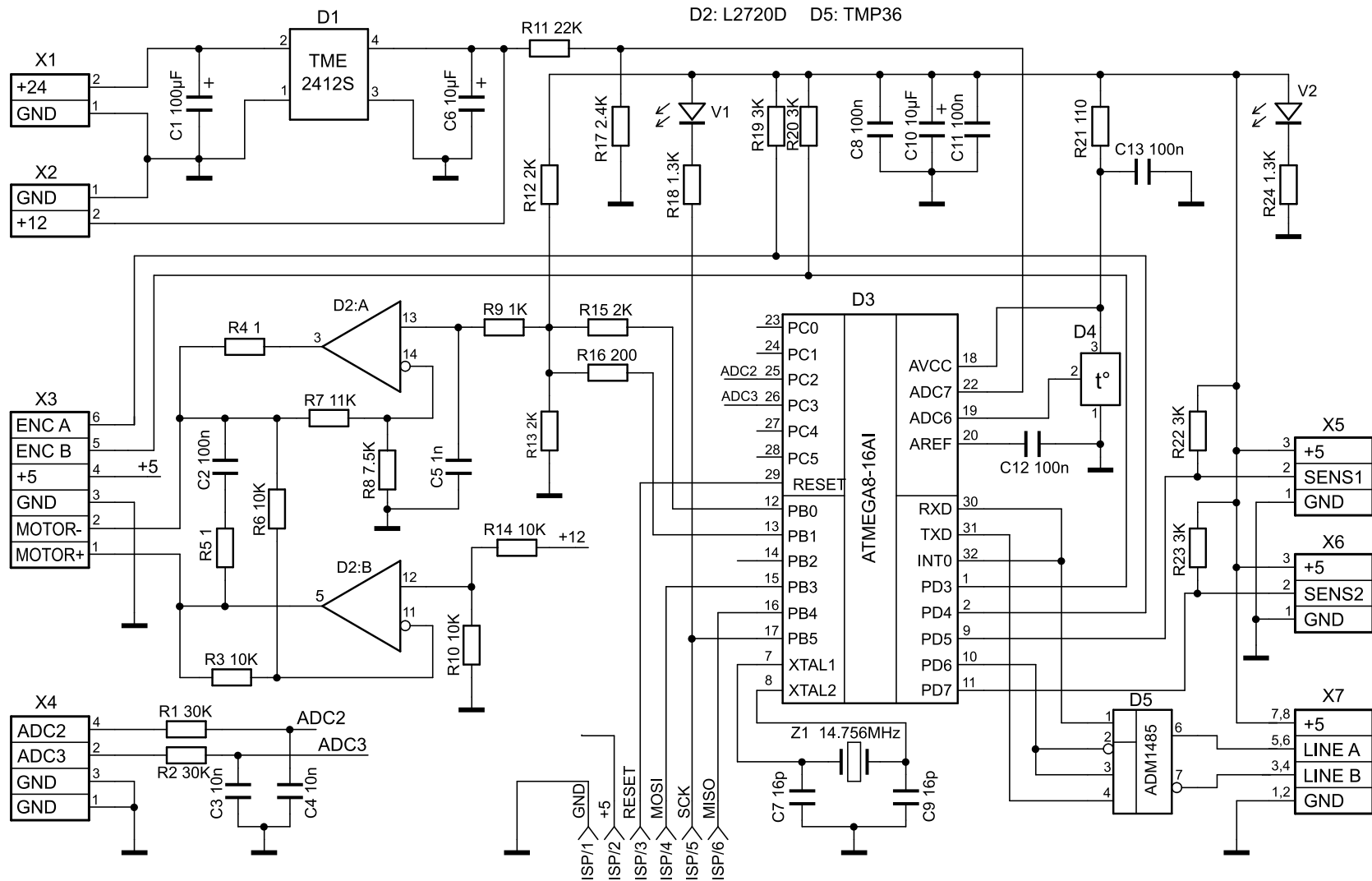


Рис. 4: Принципиальная схема контроллера мотора фокусировки.

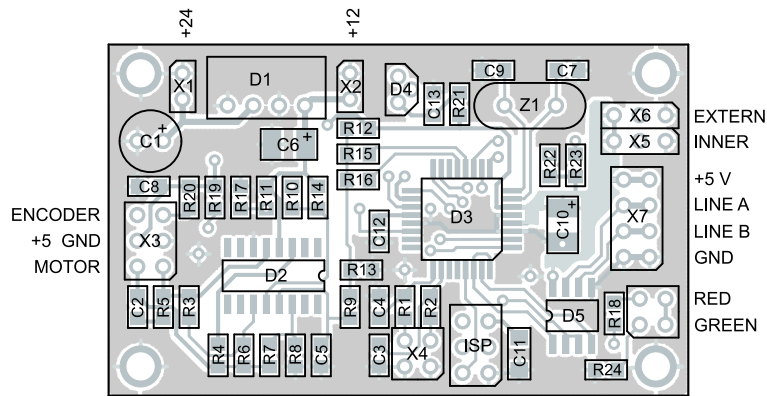


Рис. 5: Расположение элементов и штыревых разъемов на плате контроллера мотора фокусировки.

Контроллер переключения фильтров

Этот контроллер предназначен для управлением DC моторами фильтров. Расположен непосредственно в фотометре. Микрокод модуля содержится в файлах `Master_filter.asm` и `Master_filter.inc`). Модулям установлены адреса `0x12` и `0x13`.

Модуль поддерживает все общие базовые команды компактного протокола и следующие функциональные команды:

- Вывод всех фильтров с оптической оси
`REMOVE (0x80) <- ACY | ACW`
- Установка фильтра 1 на оптическую ось
`INSERT1 (0x81) <- ACY | ACW` Сигнал ожидания `ACW` возвращается если движение не завершено.
- Установка фильтра 2 на оптическую ось
`INSERT2 (0x82) <- ACY | ACW` Сигнал ожидания `ACW` возвращается если движение не завершено.
- Установка фильтра 3 на оптическую ось
`INSERT3 (0x83) <- ACY | ACW` Сигнал ожидания `ACW` возвращается если движение не завершено.
- Установка фильтра 4 на оптическую ось
`INSERT1 (0x84) <- ACY | ACW` Сигнал ожидания `ACW` возвращается если движение не завершено. Перед исполнением команд установки находящийся на оси фильтр будет автоматически выведен.
- Аварийный останов, полностью сбрасывает ток с двигателя
`STOP (0x85) <- ACY | ACW`
- Проверка завершения движения мотора
`TEST_MOTION (0x86) <- ACY | ACW`
Сигнал ожидания `ACW` возвращается если движение не завершено.
- Чтение текущего значения времени переключения `uptime`:
`READ_TLIMIT (0xF2) <- low(uptime) high(uptime)`
Значение `uptime` задается в $1/200$ с.

- Установка значения переключения `uptime`:
`LOAD_TLIMIT (0x32) low(uptime) high(uptime) <- ACY`
 В отличие от других модулей, после истечения этого времени ток с моторов не сбрасывается, а только устанавливается флаг, сигнализирующий, что фильтр установлен на оптическую ось.
- Чтение текущей температуры `temp`:
`READ_TEMP (0xF8) <- low(temp) high(temp)`
 Температура T в градусах связана с получаемым значением `temp` соотношением $T = 250 \times \text{temp} / 256$.
- Чтение напряжения питания `mainpower` двигателя:
`READ_UCC (0xFA) <- low(mainpower) high(mainpower)`
 Значение напряжения в Вольтах вычисляется по формуле $U_{cc} = 48 \times \text{mainpower} / 256$.

Как обычно, текущее состояние контроллера отражается в его регистре состояния `STATUS`, доступного только для чтения. Его формат следующий:

- Бит 0 – `NEW_CHECK`, сигнал для проверки текущего положения;
- Бит 1 – `TIME_OUT`, основ предыдущего движения произошел по тайм-ауту;
- Бит 2 – `INSERT`, активен во время фазы введения фильтра;
- Бит 3 – `REMOVE`, активен во время фазы выведения фильтра;
- Бит 4 – `FILTER4`, активен во время установки фильтра 4;
- Бит 5 – `FILTER3`, активен во время установки фильтра 3;
- Бит 6 – `FILTER2`, активен во время установки фильтра 2;
- Бит 7 – `FILTER1`, активен во время установки фильтра 1.

Состояние датчиков (или флагов состояния) отражается в регистре `SENSOR`, также доступного только для чтения:

- Бит 0 – `USET4`, флаг устанавливается если фильтр 4 полностью убран;
- Бит 1 – `USET3`, флаг устанавливается если фильтр 3 полностью убран;
- Бит 2 – `USET2`, флаг устанавливается если фильтр 2 полностью убран;
- Бит 3 – `USET1`, флаг устанавливается если фильтр 1 полностью убран;
- Бит 4 – `SET4`, флаг устанавливается если фильтр 4 уже находится на оси;
- Бит 5 – `SET3`, флаг устанавливается если фильтр 3 уже находится на оси;
- Бит 6 – `SET2`, флаг устанавливается если фильтр 2 уже находится на оси;
- Бит 7 – `SET1`, флаг устанавливается если фильтр 1 уже находится на оси;

Контроллер купола *Astrohaven*

Контроллер предназначен для управления движением моторов переменного тока южного и северного приводов купола. Он также обеспечивает ручное включение движения моторов в прямом (закрывание) и обратном (открывание) направлениях, для чего на крышке корпуса установлены 4 кнопки: открывание сектора А, закрывание сектора А, открывание сектора В, закрывание сектора В. Программное и ручное управление блокируют друг-друга до полного завершения движения створок.

Микрокод модуля содержится в файлах `Master_dome.asm` и `Master_dome.inc`. Модулю установлен адрес `0x0E`. Контроллер поддерживает все общие базовые команды компактного протокола и следующие функциональные команды:

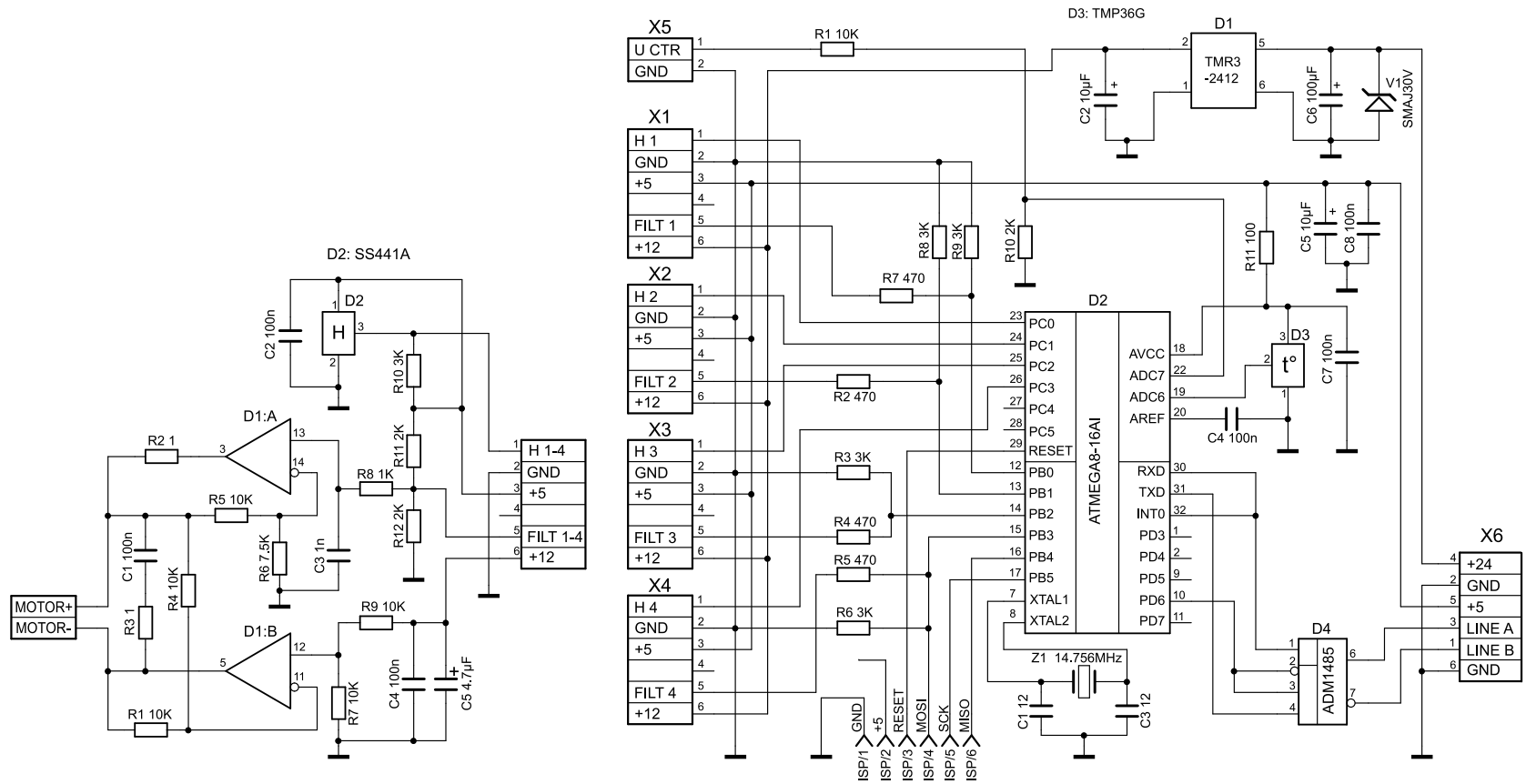


Рис. 6: Принципиальная схема контроллера мотора фокусировки. Показана цифровая часть контроллера и 1 из 4 аналоговых драйверов моторов

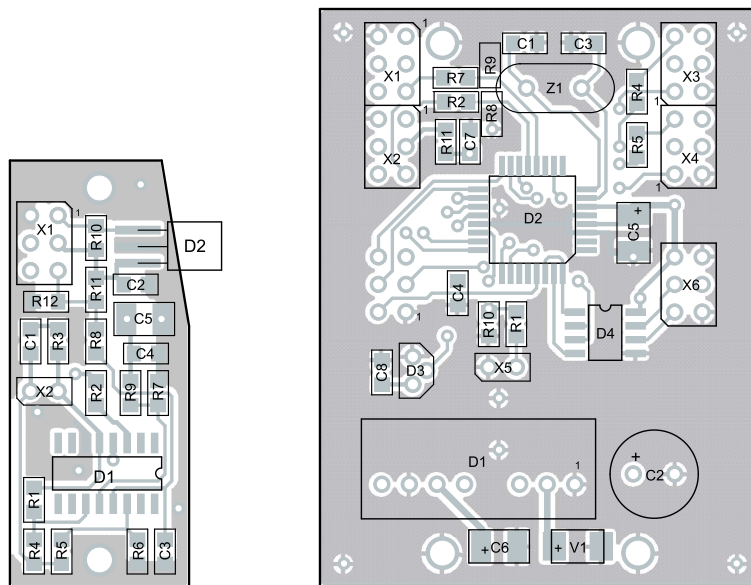


Рис. 7: Расположение элементов и штыревых разъемов на плате контроллера переключения фильтров. Слева — 1 плата драйвера мотора.

- Запуск открывания сектора А купола
A_OPEN (0x80) <- АСУ | АСВ Сигнал ожидания АСВ возвращается если выполняется движение или команда заблокирована.
- Запуск закрывания сектора А купола
A_CLOSE (0x81) <- АСУ | АСВ Сигнал ожидания АСВ возвращается если выполняется движение или команда заблокирована.
- Запуск открывания сектора В купола
V_OPEN (0x82) <- АСУ | АСВ Сигнал ожидания АСВ возвращается если выполняется движение или команда заблокирована.
- Запуск закрывания сектора В купола
V_CLOSE (0x83) <- АСУ | АСВ Сигнал ожидания АСВ возвращается если выполняется движение или команда заблокирована.
- Проверка завершения движения секторов
TEST_MOTION (0x84) <- АСУ | АСВ
Сигнал ожидания АСВ возвращается если движение еще не завершено.
- Аварийный останов движения
STOP_MOTION (0x85) <- АСУ
- Подмотка ремней сектора А если сегменты купола заблокированы
A_BACK (0x8A) <- АСУ | АСВ Сигнал ожидания АСВ возвращается если выполняется движение, команда заблокирована или ремень натянут.
- Подмотка ремней сектора В если сегменты купола заблокированы
V_BACK (0x8B) <- АСУ | АСВ Сигнал ожидания АСВ возвращается если выполняется движение, команда заблокирована или ремень натянут.
- Разблокировка движения моторов купола
UNLOCK (0x8C) <- АСУ
- Блокировка движения моторов купола
LOCK (0x8E) <- АСУ

- Чтение текущего значения предельного времени движения `uptime`:
`READ_TLIMIT (0xF2) <- low(uptime) high(uptime)`
Значение `uptime` задается в 1/100 с. По умолчанию установлено значение 15 с.
- Установка значения предельного времени движения `uptime`:
`SET_TLIMIT (0x32) low(uptime) high(uptime) <- ACY`
- Чтение значения тайм-аута обмена `sleepdown` :
`READ_SLEEP (0xF4) <- low(sleepdown), high(sleepdown)` Это значение выражено в 10 ms интервалах. По умолчанию установлено значение 150 сек.
- Установка значения тайм-аута обмена `sleepdown` :
`SET_SLEEP (0x34) low(sleepdown), high(sleepdown) <- ACY`
- Чтение значения задержки разблокировки кнопок `release` :
`READ_DELAY (0xF6) <- low(release), high(release)` Это значение выражено в 10 ms интервалах. По умолчанию установлено значение 0.5 сек.
- Установка значения задержки разблокировки кнопок `release` :
`SET_DELAY (0x36) low(release), high(release) <- ACY`
- Чтение текущего значения предельного тока `max_current`:
`READ_MAX_CURRENT (0xE8) <- low(max_current)`
- Установка значения предельного тока `max_current`:
`SET_MAX_CURRENT (0x28) low(max_current) <- ACY` Значение тока задается по формуле $\text{max_current} = 4.6 \times I$.
- Чтение текущего значения тока мотора `position`:
`GET_CURRENT (0xFA) <- low(current) high(current)`
Значение тока в Амперах получается по формуле $I = 0.22 \times \text{current} / 256$.

Текущее состояние контроллера отражается в его регистре состояния `STATUS`, доступного только для чтения. Его формат следующий:

- Бит 0 – `OVERCURRENT`, ошибка: превышение максимального тока мотора;
- Бит 1 – `NO_AC_POWER`, ошибка: отсутствие напряжения 220 В на контроллере;
- Бит 2 – `BELT_ERR`, ошибка: при движении ослаб один из ремней;
- Бит 3 – `TIME_OUT`, остановка мотора произошла не по датчикам, а по времени;
- Бит 4 – `UNLOCK`, флаг блокировки внешних команд движения;
- Бит 5 – `BYPASS`, флаг команды подмотки ремней, очищается после завершения команды;
- Бит 6 – `B_MOVE`, показывает движения мотора сектора В;
- Бит 7 – `A_MOVE`, показывает движение мотора сектора А.

Состояние датчиков (или флагов состояния) отражается в регистре `SENSOR`, также доступного только для чтения:

- Бит 0 – `B_OPENED`, устанавливается, если замкнут датчик “открыто” сектора В;
- Бит 1 – `B_CLOSED`, устанавливается, если замкнут датчик “закрыто” сектора В;
- Бит 2 – `B_TOP_FLAG`, устанавливается, если замкнут датчик верхнего ремня сектора В;
- Бит 3 – `B_BOT_FLAG`, устанавливается, если замкнут датчик нижнего ремня сектора В;
- Бит 4 – `A_OPENED`, устанавливается, если замкнут датчик “открыто” сектора А;
- Бит 5 – `A_CLOSED`, устанавливается, если замкнут датчик “закрыто” сектора А;
- Бит 6 – `A_TOP_FLAG`, устанавливается, если замкнут датчик верхнего ремня сектора А;

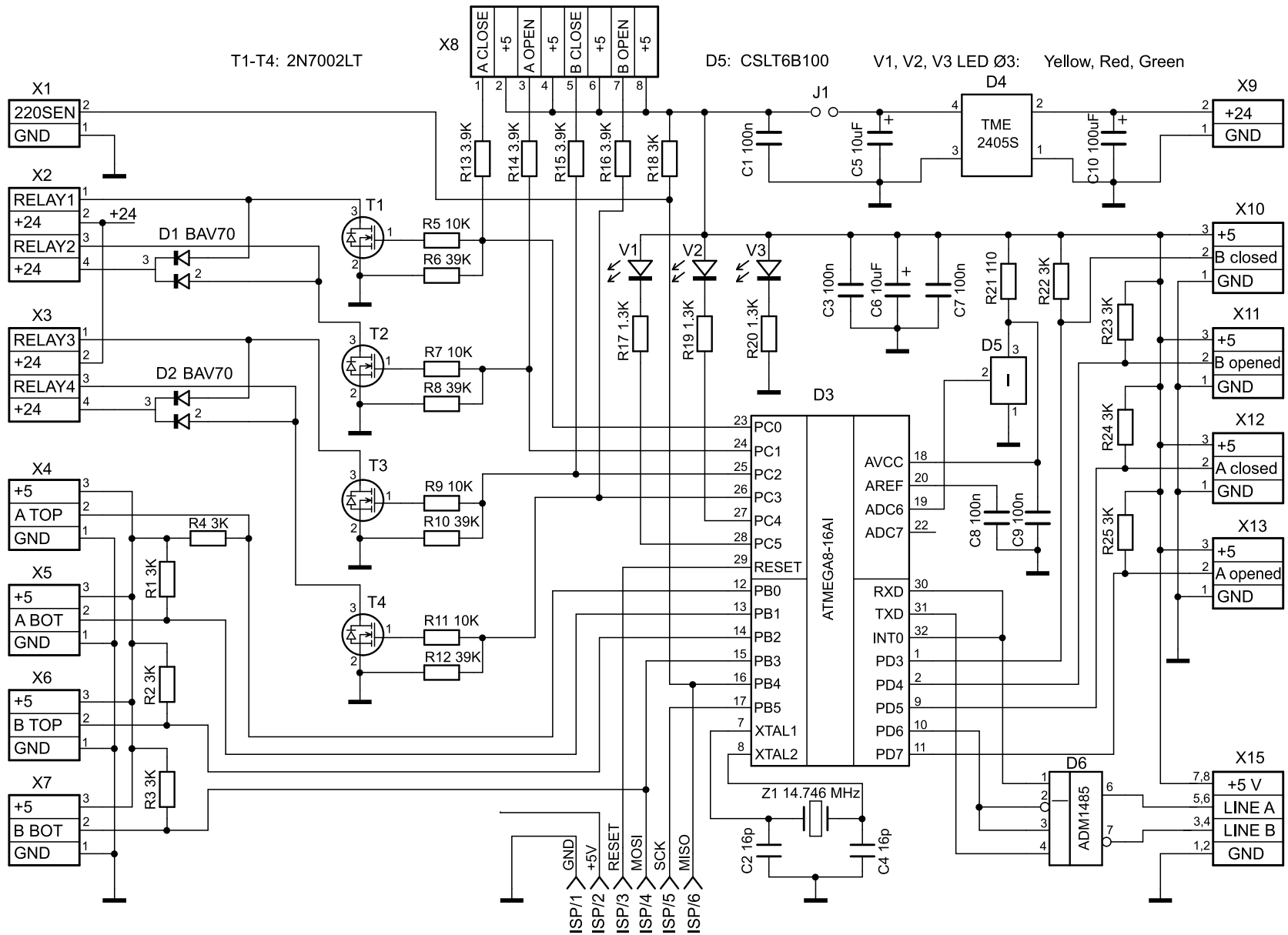


Рис. 8: Принципиальная схема цифровой части контроллера купола.

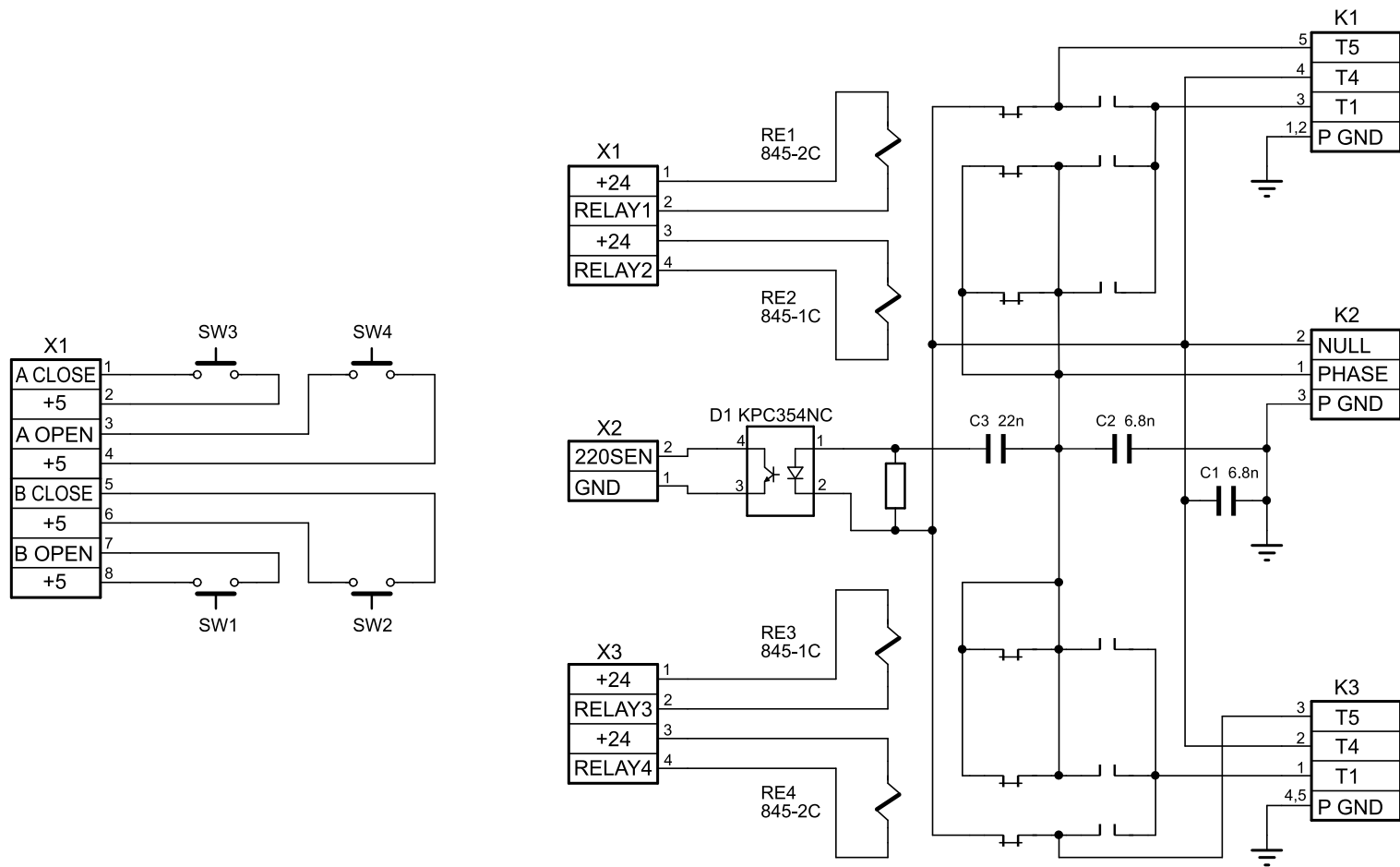


Рис. 9: Принципиальная схема подключения кнопок ручного управления (слева) и сильноточной части контроллера купола.

Бит 7 – A_BOT_FLAG, устанавливается, если замкнут датчик нижнего ремня сектора А;

Движение секторов может осуществляться только по очереди. При нормальных условиях продолжительность движения южного сектора составляет при закрывании 13.8 с со скоростью 8.3 см/с, при открывании — 13.6 с со скоростью 8.5 см/с. Продолжительность движения северного сектора составляет при закрывании 13.35 с со скоростью 8.25 см/с, при открывании — 13.15 с со скоростью 8.35 см/с.

При движении сегментов по внешним командам контролируются конечные выключатели, время движения, натяжение ремней (это существенно при открывании) и ток (существенно при закрывании). При удаленном управлении рекомендуется начинать движение с тайм-аутом около 1 с и программно контролировать отпускание концевиков “закрото” или “открыто”. Если сегменты действительно начали двигаться, эти датчики разомкнутся примерно через 0.5 с.

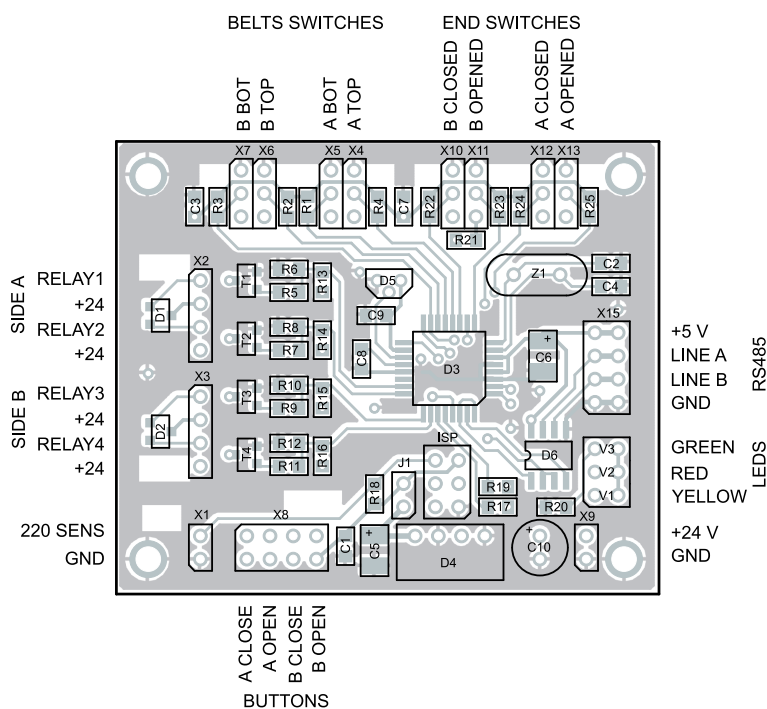


Рис. 10: Расположение элементов и штыревых разъемов на цифровой плате контроллера купола

При ручном управлении время движения не контролируется. Датчики ремней приводят к кратковременной остановке и последующему запуску, если кнопка остается нажатой. После включения напряжения питания +24 В контроллер находится в специальном начальном режиме. При этом возможно ручное управление, блокировка движения в случае срабатывания концевых выключателей. Не контролируется связь с компьютером. Поэтому можно спокойно работать со створками, хотя при длительной работе рекомендуется выключить контроллер совсем.

После первой же команды, полученной от компьютера (запуск управляющей программы или посылка отдельных команд с помощью утилиты `lmoх`), запускается таймер обмена, который через 150 с приведет к старту закрывания сектора А, а затем, еще через 150 с — и сектора В. **Будьте внимательны!**

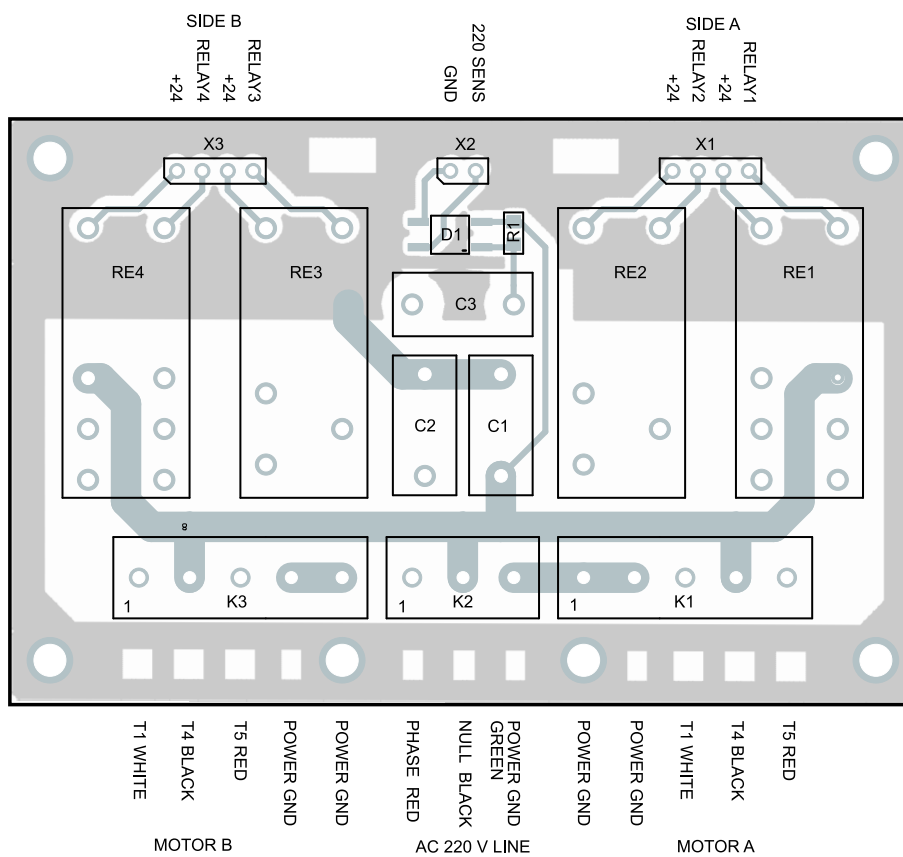


Рис. 11: Расположение элементов и штыревых разъемов на силовоточной плате контроллера купола