**Документация на систему водоснабжения**

**Проект «Чистая вода на даче»**

1. Никогда не запитывать китайские клоны Arduino Nano одновременно от USB и 12В источника. Полное уничтожение платы гарантировано!
2. Если при движении крана пропало электричество, то после возобновления подачи, он начнет открываться (см openValve, closeValve). Возможны проблемы, исправь логику.
3. Если при закрытии или открытии не сработает концевик, то в процедурах openValve и closeValve для этого есть счетчик, он крутиться до 50. С учетом delay(100) получаем, что в любом случае, через 5 секунд мотор будет остановлен.
4. Работа с дозирующим насосом, предпочтительный режим: установка в режим BATCH, режим контакта CLOSED и нужное количество ходов мембраны на 200 литров (50ход \* 10л \* 20 = 1000 ход/200л или 0.56мл/ход\*1000 = 560мл гипохлорида натрия на 200 литров). Когда на затвор полевого транзистора приходит 5В, контакты замыкаются и сразу же начинается дозация. Дополнительно (не использовать этот режим): если поставить режим контакта OPEN (в настройках насоса), то дозация будет выполняться при подаче LOW. Т.е. в первом случае насос ждет, пока замкнутся контакты, во втором – пока разомкнутся. На Dose контакте платы центральную жилу сигнального кабеля насоса подключить на +. Таким образом, штатный режим digitalWrite(x, LOW); Когда хочу дозировать: digitalWrite(x, HIGH); и опять в LOW.

Протоколы.

I2C

Сохранение состояния в EEPROM

Ведомое устройство

В ходе работы существуют две важных вещи: возобновление движения крана в нужное состояние (при отключении электричества), слежение за ошибками работы концевиков.

Для этого необходимо 4 крана \* 2 концевика + 4 желаемых состояния = 12 байт EEPROM памяти.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| V1\_IN | | | V1\_OUT | | | V2\_IN | | | V2\_OUT | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 4 | 7 | 8 | 9 | 10 | 11 |
| DS | OS | CS | DS | OS | CS | DS | OS | CS | DS | OS | CS |

DS — Desired State или желаемое состояние;

OS — Opened Switch или концевик открытия крана

CS — Closed Switch или концевик закрытия крана

При вызове функции openValve()/closeValve для любого экземпляра класса Valve, первым делом в EEPROM записывается желаемое состояние (OPENED или CLOSED) и лишь после этого отдается команда на перемещение. Ячейка для записи определяется по формуле:

*eeprom\_cell = 3 \* (valveNumber — 1)*

где valveNumber — порядковый номер крана от 1 до 4, определяемый константной V1(2)\_IN(OUT) (в скобках альтернативное значение) при вызове конструктора Valve.

При отключении питания и последующего его возобновления, в ходе инициализации контроллера для каждого экземпляра класса Valve вызывается функция restoreState, которая читает сохраненное в EEPROM желаемое состояние и в соответсвии с полученным значением до-открывает или до-закрывает кран.

Ошибками в ходе работы кранов по сути являются факты ненажатия на концевики. Чтобы за этим следить, после окончания движения крана в EEPROM пишется был нажат концевик (true) или нет (false). Попутно оператор return возвращает результат события выше, где оно сохраняется в массив.

Из массива регулярно считывааются значения функцией errorCheck и на основе этих значений встроенным в плату светодиодом сигнализируется номер крана, а затем неисправный концевик. Например, если у крана V2\_IN:

- неисправен концевик открытия, то будет три вспышки, пауза, одна вспышка.

- неисправен концевик закрытия: три вспышки, пауза, две вспышки

- неисправны оба: три вспышки, пауза, одна, пауза, две.

После длинная пауза, и если неисправен следующий кран, сигнализация повторяется.

При перезапуске контроллера состояние концевиков восстанавливаются из EEPROM, а пишутся, как сказано выше они по формуле:

*cell\_index = 3 \* (valveNumber — 1) + 1; //для open switch*

*cell\_index = 3 \* (valveNumber — 1) + 2; //для close switch*

Сторожевой таймер

Контроллеры зависают при пуске насосов. Электромагнитный импульс выводит их из строя до жесткой перезагрузки. Чтобы исключить непредвиденные последствия, был активирован сторожевой таймер. Загрузчик Arduino Nano по умолчанию не деактивирует сторожевой таймер, и оставляет его с предельно малым временем ожидания в 15 мс. Из-за этого система уходит в бесконечный цикл перезагрузок. Чтобы это вылечить, потребовалось заменить загрузчик в Nano. Для этого была взята свободная Arduino Nano и прошита из родной среды примером Arduino ISP. Получается программатор, который подключается к целевой плате. Затем средствами avrdude (avrdude –v –p m328p –P COMx –b 115200 –c avrisp (скорость может быть разной)) прошивается загрузчик optiboot\_atmega328p.hex, взятый с <https://github.com/Optiboot/optiboot> (здесь первичное описание, но оно оказалось нерабочим, <https://geektimes.ru/post/255800/> ) Optiboot также идет в пакете родных программ Ардуино штатно. Прошивка загрузчика через среду Ардуино с выбором Arduino ISP и указанием целевой платы не удалась. Походу, проблема в скоростях UART.

Затем программатор отцепляется, в файле Arduino/hardware/arduino /avr/boards.txt в 147-ой строке выставляется скорость nano.menu.cpu.atmega328.upload.speed=115200, поскольку Optiboot загрузчик теперь работает с другой скоростью. В среде Ардуино выбирается родной программатор Arduino AVRISP mkII и все шьется как и раньше.