Показать сообщение отдельно **Тема**: Обсуждение: Локальная сеть корвет

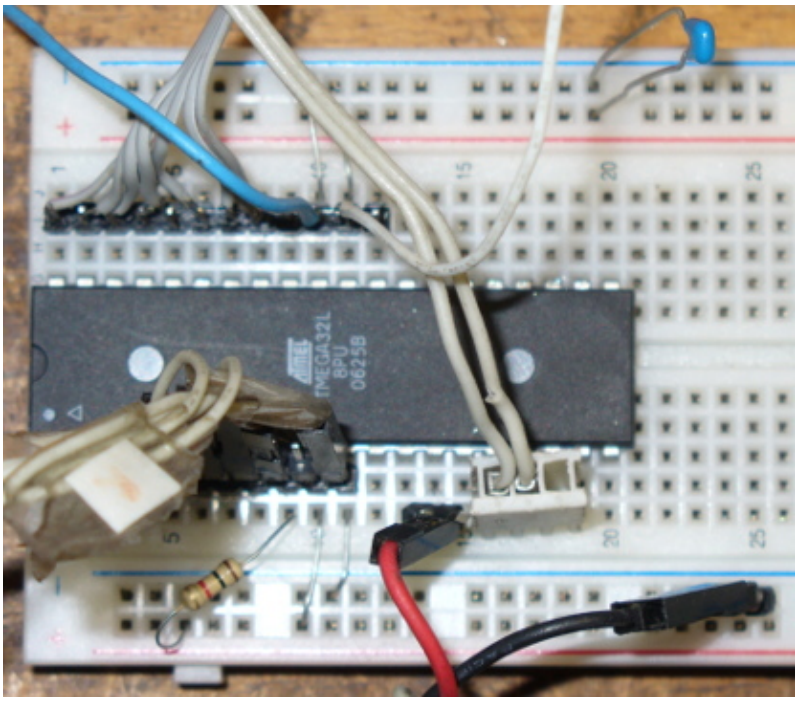
25.06.2014, 08:17 #**132**

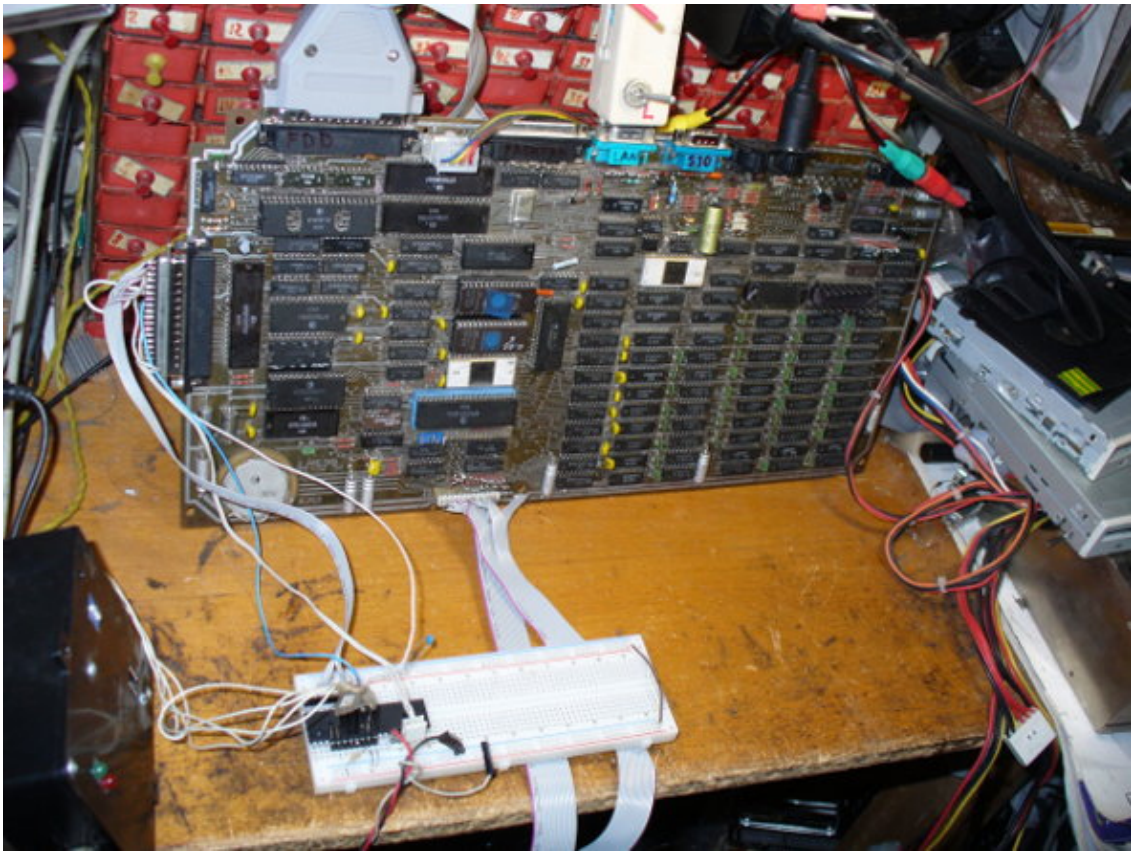
forth32
Member

Регистрация: 17.04.2011
Адрес: Санкт-Петербург
Сообщений: 66
Сказал(а)
Спасибо: 0
Поблагодарили 49 раз(а) в 24 сообщениях

Итак, рассказываю о первых опытах загрузки через боковой разъем корвета. Вначале предыстроия - о набитых по дороге шишках.

Я собрал макетку с Atmega32, подключил порт А меги к каналу А корвета, прерывание int0 меги - к сигналу Control, in1 - к каналу PB0 (0 бит адреса). То есть я пошел по пути, предложенному ESL - стробироваться единственным адресным битом. Получилась шина из 12 проводов (8 данных, 2 строба, земля, питание). Выглядит это так:





Написал программу для AVR, и небольшую тестовую программку для загрузки в корвет (она просто выводит строчку на экран и останавливает процессор). Запустил. И сразу огреб ошибку ОПТС - ОШИБКА ШИНЫ. Пришлось разбираться.

Во-первых, я решил просимулировать процесс загрузки. Еще раз спасибо ESL за сорцы его эмулятора. Я приделал к эмулятору загрузку с внешнего ПЗУ (добавил ключ -r в командную строку), и получил ту же самую Ошибку Шины! Уже в эмуляторе. Оказалось (я это разбирал в свое время в IDA, но уже забыл), что сумрачные гении, писавшие ОПТС, додумались тестировать шину путем записи битовых образцов в канал А того самого ВВ55, который выведен на боковой разъем. Предварительно переключив этот канал в режим вывода. Они решили, что это хорошая идея - срать в порт, к которому неизвестно что подключено и который они же потом используют для ввода. В конце концов, если уж так хочется, есть же порт данных принтера, к которому точно ничего, кроме принтера, подключено быть не может. А боковой разъем они же сами позиционировали как универсальный для радиолюбителей (судя по статьям в том же журнале радио). За такое я бы яйца отрывал, причем медленно.

Дело в том, что ВВ55 обладает интересной особенностью. При чтении порта он всегда выдает текущее состояние своих ножек. Независимо, работает канал на ввод или на вывод. В результате, если порт работает на вывод, в канал записана 1, а внешний сигнал подтянул ножку к земле - из канала прочитается 0. Отсюда и ошибка шины - моя бедная мега просто перетянула к земле некоторые биты порта. Оказалось, что переключать порт меги на вывод можно только после того, как сигнал Control станет 1. До этого порт должен быть в режиме ввода, чтобы не мешать ОПТС развлекаться с тестом шины. Проблема в том, что этот самый сигнал Control в момент включения питания может несколько раз дернуться вверх-вниз, и это все надо отслеживать.

Далее я доработал эмулятор для отслеживания сигнала control, и заодно добавил туда сборку лога обращений к эмулируемой внешней ПЗУ. Оказалось, что с полследовательностью адресов мы не ошиблись - 4,5,6,7,0 и далее 3 раза по кольцу все 256 адресов. Заодно проверил свою тестовую программку - в эмуляторе она загрузилась и запустилась нормально.

Теперь, зная о подводных камнях, я исправил программу в AVR. Приходится отслеживать оба перепада сигнала Control. Если он стал 0 - немедленно переводим порт A в режим ввода и ждем. Если стал 1 - выставляем в порт A первый в цепочке байт (по адресу 4) и начинаем отслеживать перепады PB0, выставляя по порядку байты в порт A. Вот и весь процесс загрузки.
В результате я получил 2 варианта загрузчика для AVR - один использует прерывание от PB0 (типа Pinchange - по обоим перепадам сигнала), другой - программный опрос сигнала PB0. Работоспособны оба. Но программный опрос имеет, на мой взгляд, 2 преимущества - он короче на 14 байт и может работать с любым битом любого порта AVR, то есть освобождается ценное прерывание Int1. А в меге32 этих прерываний всего 3, не так уж и много. Прикладываю к посту архив с обоими вариантами загрузчика. exr.asm - вариант с прерываниями, exr-poll.asm - вариант с программным опросом. extrom.bin - загружаемый блок с тестовой программкой. Для тех, кто захочет повторить мой эксперимент (похоже, вряд ли такие найдутся, но а вдруг...) - надо выставить FUSE на тактирование от внутреннего генератора на 8 МГц, и включить детектор напряжения питания на 4.5 V - чтобы программа не сходила с ума в момент включения питания. Естественно, можно использовать любую мегу, не только 32, и даже тиньку. Только подправить программу на используемые порты.
256-байтовый блок для загрузки в корвет размещается в NVRAM меги с адреса 0. Его можно прошить отдельно с помощью того же avrdude с ключем -U nvram. То есть можно менять эти блоки, не трогая основную флеш-память AVR. При страте программа переносит этот блок в основную SRAM с адреса 100, и оттуда уже выдает данные в порт по мере необходимости.

Итак, загрузчик работает. Что дальше? Я вижу несколько этапов, которые придется пройти.

1. Допаять еще несколько проводов - от канала C BB55 к какому-нибудь порту меги. Это буду сигнал управления двухсторонним обменом в режиме 2.

2. Придумать протокол загрузки и написать вторичный загрузчик. Он должен будет перекинуть в корвет образ системы (сетевой BIOS + BDOS + CCP). Примерно как сейчас заливается по сети CP/N.SYS.

3. Приделать к меге какое-нибдуь хранилище данных - хотя бы ту же SD-карту, хоть мне и очень не хочется связываться с драйвером VFAT. Кстати, на первых порах можно и без фата обойтись - просто разбить SD на несколько фиксированных зон и прописать туда данные с помощью обычного dd. Проблема еще в том, что SD требует 3v уровней - придется мудрить с согласованием, а я это очень не люблю...

4. Ну и, наконец, придумать и реализовать протокол взаимодействия между сетевым биосом и программой в AVR. Это самая интересная, но при этом громоздкая часть работы. За основу можно взять исходные тексты стандартного биоса корвета, добавив туда перехват запросов READ и WRITE при обращении к эмулируемым дискам, и какой-нибудь командный протокол для монтирования образов KDI.

Вложения

extboot.7z (7.3 Кб, 2 просмотров)

Online

| IP Address

edit quote

Эти 2 пользователя(ей) сказали Спасибо forth32 за это полезное сообщение: Удалить вашу благодарность	esl (25.06.2014), eugeniusz (25.06.2014)