Разработка средств автоматизации программирования устройств Интернета вещей на базе платформы SciVi

Лукьянов Александр Михайлович

2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Энергонезависимая память — особый вид запоминающих устройств, способный хранить данные при отсутствии электропитания. Такая память чаще всего используется для хранения данных, необходимых для инициализации устройства, и конфигурационных данных между его устройства. Такая задача особо остро стоит при работе с микроконтроллерами (миниатюрными компьютерами, обычно размещёнными в одной интегральной схеме). Это обусловлено, во-первых, уязвимостью таких устройств к перебоям электропитания и, во-вторых, особенностями условий их использования: устройства с микроконтроллерами обычно создаются для автономной работы, поэтому после временного отключения питания они должны самостоятельно восстанавливать своё прошлое состояние. В микроконтроллерах для решения этой задачи обычно используются электрически стираемые перепрограммируемые постоянные запоминающие устройства (ЭСППЗУ, англ. Electrically Erasable Programmable Read-Only Memory, EEPROM) — вид устройств энергонезависимой памяти, позволяющих электрическим импульсом стереть сохранённые данные, а затем, при необходимости, записать новые.

Микроконтроллеры, в частности, используются платформой научной визуализации и визуальной аналитики SciVi, разработанной сотрудниками Пермского государственного национального исследовательского университета [1]. В SciVi уже реализованно сохранение настроечной информации в EEPROM, однако сделано это за счёт стандартных средств. Их низкоуровневость и ограниченность не позволяет использовать EEPROM удобно и, главное, расширять его применение хранением новых данных.

В основе данной работы лежит поиск решения указанных проблем в использовании EEPROM микроконтроллеров.

Цель работы: разработать программный модуль с высокоуровневым интерфейсом, позволяющий удобно и без необходимости ручной настройки

сохранять и считывать информацию из EEPROM микроконтроллеров, в соответствии с требованиями платформы SciVi.

Объект исследования данной работы: использование энергонезависимой памяти. Предмет исследования: использование EEPROM микроконтроллеров для автоматизации их настройки.

Для достижения цели работы, были поставлены следующие задачи:

1. Составить требования к необходимому программному модулю.
2. Исследовать существующие средства для работы с энергонезависимой памятью и, в частности EEPROM микроконтроллеров.
3. При возможности выбрать одно из таких средств для использования в качестве основы разрабатываемого модуля.
4. Разработать программный модуль для работы с EEPROM микроконтроллеров, соответствующий всем поставленным требованиям.
5. Провести тестирование и отладку разработанного программного модуля.
6. Интегрировать разработанный модуль в платформу SciVi.

1 Анализ существующих решений

1.1 Постановка задачи

В платформе SciVi в основном применяются микроконтроллеры серии ESP8266, а для их программирования используются инструменты среды разработки Arduino IDE, позволяющие программировать микроконтроллеры, используя язык программирования C++. А так же специальное дополнение к этой среде для работы с ESP8266 [2], содержащее, в частности, набор “стандартных” библиотек.

Таким образом, необходимый программный модуль должен представлять собой библиотеку классов языка программирования C++, может использовать стандартный набор библиотек Arduino IDE и указанного дополнения к ней, выполнятся на микроконтроллерах серии ESP8266 и, по возможности, Arduino. Эта библиотека должна:

1. Предоставлять пользователю возможность сохранять и считывать данные из EEPROM микроконтроллера. При этом:
   1.1. Данные могут иметь произвольную структуру.
   1.2. Доступ к ним должен производиться по некоторым идентификаторам, уникальным для различных данных. Без необходимости ручных манипуляций с адресами EEPROM со стороны пользователя.
2. Автоматически определять факт наличия в EEPROM данных с заданным идентификатором, определять адрес для записи новых данных, сохранять в EEPROM метаданные о хранящих данных для их использования после перезапуска микроконтроллера.
3. Минимизировать количество операций записи в EEPROM, т.к. каждая ячейка такой памяти может быть перезаписана ограниченное количество раз (обычно производители гарантируют от 100.000 до 1.000.000 циклов перезаписи), после чего выходит из строя.

Ключевым требованием является полная автоматизация работы с адресами EEPROM, это необходимо для создания возможности использования EEPROM из различных независимых программных

модулей. В противном случае, этим модулям понадобилось бы каким-либо образом обмениваться информацией об используемых ими адресах для избежания чтения и записи разными модулями по одним и тем же адресам.

1.2 Стандартная библиотека

В стандартный набор библиотек Arduino IDE уже входит библиотека для работы с EEPROM [3]. Однако она предоставляет только простые функции, такие как: записать и считать бит по указанному адресу, позже в неё были также добавлены функции для чтения и записи данных произвольных типов, но так же только по явно указанному адресу. Очевидно, это делает стандартную библиотеку нарушающей все поставленные требования, однако её функции можно использовать в качестве низкоуровневого интерфейса EEPROM в разрабатываемом библиотеке. Кроме указанных выше, стандартная библиотека содержит обёртку вокруг функции записи, производящую физическую перезапись данных, только если они отличаются от хранящихся по указанному адресу в данный момент. В дальнейшем, практически во всех случаях, есть смысл использовать для записи именно эту функцию с целью уменьшения износа EEPROM.

1.3 Библиотека EEManager

Как и SciVi в данный момент, большая часть проектов, хранящих какие-либо данные в EEPROM, ограничиваются использованием стандартной библиотеки. И до недавнего времени в открытом доступе отсутствовали более высокоуровневые альтернативы. Однако не так давно появилась новая библиотека, преследующая те же цели - EEManager [4]. Она имеет открытый исходный код (опубликован под лицензией MIT [5]) и документацию на русском языке.

Эта библиотека также требует ручного манипулирования с адресами, однако имеет следующие преимущества:

- Реализован механизм отложенной записи: по-умолчанию данные записываются в EEPROM с заданной задержкой после последней команды на запись. Использование такого подхода имеет смысл в ситуациях, когда данные должны перезаписываться много раз

за короткий промежуток времени, в действительности же, с таким механизмом, данные в EEPROM будут записаны только в последний раз. В то же время этот механизм имеет значительный недостаток: если потеря питания произойдёт после команды записи, но до истечения задержи, новые данные записаны не будут. Это делает использование такого механизма оправданным только в устройствах, для которых гарантия записи не является обязательной и точность восстановления состояния после потери питания не представляет критической важности.

- Библиотека также реализует "механизм ключа первой записи". Вместе с каждым блоком данных в EEPROM хранится специальное однобайтовый ключ. При обращении к блоку данных пользователь указывает придуманный им ключ, который не должен изменяться от запуска к запуску, а из EEPROM считывается записанное значение ключа. Если они совпадают, значит необходимые данные уже записаны в EEPROM и их необходимо считать, иначе данные никогда не были записаны, тогда данные должны быть наоборот записаны.

Работа с данной библиотекой осуществляется следующим образом:

1. Создаётся переменная в энергозависимой памяти, значение которой необходимо хранить в EEPROM.
2. Создаётся специальный объект, описывающий блок EEPROM. При этом пользователь указывает переменную в энергозависимой памяти, значение которой необходимо хранить, и адрес в EEPROM, начиная с которого должна быть записана эта переменная.
3. С помощью механизма ключа первой записи либо в EEPROM записывается значение переменной по-умолчанию, либо наоборот: сохранённое в EEPROM значение считывается в переменную.
4. При необходимости текущее значение переменной записывается в EEPROM. Для этого у описанного выше объекта существует два метода: для немедленной записи и для запуска таймера записи с задержкой.

1.4 EEPROMWearLevel

2 Разработка библиотеки менеджера EEPROM

2.1 Требования к библиотеки

2.2 Выбор необходимых программных средств

2.3 Разработка структуры библиотеки

2.3.1 Разработка внешнего интерфейса библиотеки

2.3.2 Переменные

2.3.3 Разделы памяти

2.3.4 Менеджер памяти

2.4 Разработка библиотеки

2.5 Использование библиотеки

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Ryabinin K.*, *Chuprina S.* Adaptive Scientific Visualization System for Desktop Computers and Mobile Devices // Procedia Computer Science. — 2013. — Т. 18. — С. 722—731.
2. Arduino core for ESP8266 WiFi chip / I. Grokhotkov [и др.]. — URL: `https://github.com/esp8266/Arduino`.
3. *Arduino Software*. EEPROM Library. — URL: `https://docs.arduino.cc/learn/built-in-libraries/eeprom`.
4. *AlexGyver*. EEManager. — 2021. — URL: `https://github.com/GyverLibs/EEManager`.
5. *Massachusetts Institute of Technology*. MIT License. — 1988. — URL: `https://opensource.org/license/mit/`.

ПРИЛОЖЕНИЕ