Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Филиал ФГАО ВО «ЮУрГУ (НИУ)» в г. Миассе
Факультет «Электротехнический»
Кафедра «Автоматика»

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой
_____С.С. Голощапов
_____2018 г.

Имитатор датчика температуры с интерфейсом CANopen

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ– 27.03.04.2018.340 ПЗ ВКР

Руководитель проекта
ст. преп. кафедры АиУ
_____Е.А.Канашев
_____2018г

Автор проекта
студент группы МиЭТ-482
_____Е.Д.Пояркова
_____2018 г.

Нормоконтролер
кафедры АиУ
_____Т.А. Барбасова
_____2018г.

Миасс 2018

АННОТАЦИЯ

Пояркова Е.Д. Имитатор датчика температуры с интерфейсом CANopen. – Миасс:филиал «ФГАО ВО ЮУрГУ (НИУ)», Эт; 2018, 74 с. 29 ил., библиогр. список – 51 наим., 2 прил., 1 лист схем ф. А3, 4 листов схемы ф. А4

В рамках выпускной квалификационной работы представлена разработка программного обеспечения для имитатора датчика температуры. На основе требований технического задания был произведен выбор оборудования для необходимой работоспособности.

Работа содержит в себе разработанную структурную схему устройства и электрическую принципиальную схему. Описан алгоритм работы программы.

Результатом работы является получение устройства, которое способно имитировать датчик температуры с помощью интерфейса CANopen.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 7
1 ПРОМЫШЛЕННЫЕ СЕТИ И ИНТЕРФЕЙСЫ 9
1.1 Общие сведения о промышленных сетях 9
1.2 CAN(Controller Area Network)-протокол 11
1.3 CANopen 17
1.3.1 Сведения о CANopen 17
1.3.2 Прикладной уровень: CANopen 18
1.3.3 Электронные спецификации устройств CANopen 21
2 РАЗРАБОТКА СТРУКТУРНОЙ СХЕМЫ И ВЫБОР ОБОРУДОВАНИЯ 24
2.1 Структурная схема 24
2.2 Выбор и анализ оборудования 24
2.2.1 Процессорный модуль 24
2.2.2 Блок питания 26
2.2.3 Элемент управления 27
2.2.4 CAN оборудование 29
2.2.5 Блок индикации 34
2.2.6. Отладочная плата 37
3 ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ CANOPEN 43
3.1 Стек протокола CANopen подчиненного устройства 43
3.1.1 Библиотека CANopenNode 43
3.1.2 Библиотека CANFestival 45
3.2 Программное обеспечение CANopen верхнего уровня 46
4 ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДАТЧИКА ТЕМПЕРАТУРЫ 48
4.1 Алгоритм работы имитатора датчика температуры 48
4.2 Имитация датчика температуры 48
4.3 Среда разработки 53
4.4 Проектирование программного обеспечения 55
ЗАКЛЮЧЕНИЕ 61
БИБЛИОГРАФИЧЕСКИЙ СПИСОК 62
ПРИЛОЖЕНИЕ А 66
ПРИЛОЖЕНИЕ Б 67

ВВЕДЕНИЕ

В автоматизированных системах обмен информацией между устройствами, в общем случае происходит при помощи промышленной сети («полевой шины», Fieldbus) [1].

Существуют отличия между промышленными сетями и офисными. В отличие от офисных сетей промышленные сети должны обеспечивать следующие свойства:

– особое конструктивное исполнение, которое должно обеспечивать защиту от влаги, ударов, пыли;

– обеспечивать устойчивость от воздействия электромагнитных помех;

– прочность кабеля, изоляции, элементов крепления должна быть повышенной;

– передача данных осуществляется с высокой надежностью;

– работы с длинными линиями [2].

В настоящее время промышленные сети широко распространены в различных отраслях промышленности. Существует уже более 50 различных типов промышленных сетей (Profibus, FDDI, Ethernet, Modbus, CANopen и другие). Каждый отдельный протокол был разработан с учетом особенностей технических систем и конкретных производств, они обеспечивают надежную коммуникацию и высокую точность управления. Из этого множества различных промышленных сетей лишь часть получила широкое распространение. Большая часть АСУ ТП в России использует сети Modbus и Profibus. В последнее время вырос интерес к сетям на основе DeviceNet и CANopen. Самой распространенной в мире считается открытая промышленная сеть Profibus. Чаще всего она используется для управления технологическим процессом, в крупных агрегатах для сборки, различных механизмов для транспортировки деталей и материалов. Когда осуществляется передача коротких сообщений, накладные расходы увеличиваются. Так же Profibus имеет более высокую стоимость по сравнению с

другими промышленными сетями. Самой простой и самой дешевой промышленной шиной является ASI, она очень просто конфигурируется. ASI наиболее часто используется для подключения фотоэлементов, датчиков приближения, индикаторов в различных системах упаковочных линий. Шина ASI была разработана для применения в небольших системах, где имеются устройства ввода/вывода.

Недорогой и надежной основой для некоторых распространенных промышленных сетей является CAN. Такая сеть имеет большую степень надежности, она может хорошо функционировать в очень сложных условиях. Вероятность отказа у CAN очень мала, менее одного случая в столетие.

Необходимо реализовать устройство, которое будет способно имитировать датчик температуры с использованием интерфейса CANopen. Для этого следует определиться с выбором оборудования, с помощью которого оно будет реализовано. Предстоит собрать структурную схему готового устройства, описать алгоритмическую часть и основное содержание кода вместе с используемыми функциями. Промышленное оборудование с использованием промышленных протоколов стоит дорого, необходимо построить простой аналог, который бы работал по протоколу CANopen.

1 ПРОМЫШЛЕННЫЕ СЕТИ И ИНТЕРФЕЙСЫ

1.1 Общие сведения о промышленных сетях

Процесс автоматизации промышленных производств развивается все более ускоряющимися темпами: увеличивается количество «интеллектуальных» оконечных устройств, растет число вовлеченных в процессы контроля и управления технологическим процессом вычислительных систем на базе микроконтроллеров [2]. Из-за этого в значительной мере возросла роль данных, которые собираются на всех уровнях АСУ ТП. Потребители информации повышают требования в части надежности получаемых данных, объема и скорости, следовательно, вопрос в обеспечении коммуникаций становится высокоприоритетным.

Промышленная сеть включает в себя комплекс оборудования и программного обеспечения, которые должны обеспечивать обмен информацией между несколькими устройствами. Такую сеть считают основой для построения распределенной системы сбора данных и управления.

Промышленная сеть соединяется с её компонентами при помощи интерфейсов. Сетевой интерфейс – физическая или (и) логическая граница между устройством и средой передачи информации. Чаще всего такой границей является набор электронных компонентов и связанного с ним программного обеспечения. При сильных изменениях внутренней структуры устройств или программного обеспечения интерфейс остается неизменным, что позволяет выделить интерфейс в составе оборудования [3].

При выборе интерфейса обычно смотрят на пропускную способность и максимальную длину подключаемого кабеля. В промышленной автоматизации наибольшее распространение получили последовательные интерфейсы RS-485, RS-232, RS 422, Ethernet, CAN, HART, AS-интерфейс.

Для того чтобы мог происходить обмен информацией, устройства, которые взаимодействуют между собой должны иметь один и тот же протокол обмена.

Протоколом можно назвать набор правил, который управляет обменом информации. Он назначает синтаксис и семантику сообщений, синхронизацию и состояние при коммуникации, операции управления. Устройства в промышленных сетях взаимодействуют при помощи модели клиент-сервер. В такой модели взаимодействуют два объекта, сервер выполняет действия, которые запрашивает клиент. Такая модель удобна для передачи данных, которые могут появляться периодически или в определенные промежутки времени. В такой модели взаимодействия выделяют устройство, которое другим подчиненным устройством. Устройство, которое проявляет инициативу при обмене, называют ведущим, главным или мастером. Устройство, отвечающее на запросы отправляемые мастером, называют ведомым или подчиненным [3]. Ведомое устройство не может начать коммуникацию первым, чтобы оно начало коммуникацию ведущее устройство должно отправить ему запрос, только после этого появится ответ на запрос. В сеть могут входить несколько ведущих устройств, такие сети называются многомастерными.

Каждая сеть может иметь топологию кольца, звезды, шины или смешанную. Топология звезда применятся довольно редко в промышленной автоматизации. Кольцо применятся в основном для передачи маркера в многомастерных сетях. Общепринятой топологией является шина.

Так как главной функцией сети можно считать соединение разного рода оборудования, проблема стандартизации приобрела особое значение. Поэтому в начале 80-х годах Международная организация по стандартизации ISO вместе с другими организациями сформулировала и приняла модель взаимодействия открытых систем OSI [4]. В модели OSI используется 7 уровней, при этом декомпозиция сделана таким образом, что только соседние уровни могут осуществлять взаимодействие между собой. Данный подход обеспечивает возможность решать задачи взаимодействия систем для всех уровней отдельно. Семь уровней модели OSI: 1 – физический, 2 – канальный (передачи данных), 3 – сетевой, 4 – транспортный, 5 – сеансовый, 6 – уровень представления, 7 – прикладной. Эта модель описывает лишь средства, которые реализуются

операционной системой, аппаратурой и сетевыми утилитами, но не описывает средства взаимодействия между приложениями.

Открытие промышленные сети – это сети, на которые распространяются международные стандарты промышленных сетей [2]. Открытые системы показывают в соответствие с особенными требованиями всех потребителей. Благодаря использованию принципов таких систем интеграция продукции различных производителей в одну сеть решается без особых проблем.

1.2 CAN(Controller Area Network)-протокол

CAN – протокол разрабатывался для использования в автомобильной электронике фирмой Robert Bosch GmbH. Этот протокол обладает повышенной надежностью и помехоустойчивостью. Так же есть особые возможности:

– все узлы могут получать сообщения с синхронизацией по времени;

– гибкость конфигурации;

– режим мультимастер;

– распознавание и передача сигналов о наличии ошибок;

– используется на витой паре на большие расстояния, до 1 км;

– разделяет случайные ошибки и постоянный отказ, какого либо узла, есть возможность отключить дефектный узел;

– осуществление передачи сообщений со сбоем автоматически при получении возможности еще одного доступа к шине.

Протокол передачи данных CAN описывает только первые два уровня OSI – физический и канальный уровень. В начале 90-х годов компания Bosch разработала спецификацию для этого протокола, и на основе него было реализовано большое число полнофункциональных сетей, таких как SDS, CANopen, DeviceNet и прочие. CAN интерфейс настольно распространен, что почти у каждого крупного производителя микроконтроллеров есть изделие с

данным интерфейсом. К главным достоинствам, которые определили такую большую популярность этого протокола для разработчиков можно отнести:

– большая скорость (до 1 Мбит/с);
– в сети могут размещаться несколько ведомых устройств;
– метод доступа CSMA/CA;
– стабильная система обнаружения и исправления ошибок.

CSMA/CA содержит в себе минимально возможную задержку передачи информации с эффективным арбитражем ситуаций, когда несколько узлов начинают передавать данные одновременно [2].

Физический уровень определяется стандартом ISO 11898. Используется дифференциальное подключение приемопередатчиков, которое обеспечивает подавление синфазной помехи. Уровень сигнала организовывается на 1/3 от уровня напряжения питания, вдобавок напряжение питания не регламентируется жестко. На рисунке 1.1 приведены типичные значения при значении напряжения питания +5В. В CAN-сети логическая единица всегда доминирует над логическим нулем. Из-за этого в стандарте CAN используется понятие доминирующее состояние линии для обозначения состояния линии с током и понятие рецессивное состояние как противоположное доминирующему.

Рисунок 1.1 – Электрические характеристики при напряжении питания +5В

Каждая сеть может иметь топологию кольца, звезды, шины или смешанную. Топология звезда применятся довольно редко в промышленной автоматизации. Кольцо применятся в основном для передачи маркера в многомастерных сетях. Общепринятой топологией является шина.

У CAN-сети ни один из узлов не обладает адресом. В этой сети используется другой метод передачи сообщений, он называется широковещательной передачей. Смысл его заключается в том, что сообщения отправляются всем узлам, но в его составе есть идентификатор, благодаря которому описан смысл отправленных данных. Благодаря этому идентификатору любой узел может получить это сообщение, если оно требуется данному устройству для функционирования. Узел принимает сообщение, если его идентификатор прошел через фильтр сообщений, который имеется на каждом узле.

CAN-сеть гарантирует, что все сообщения будут приняты любыми из узлов в одно и то же время или не будут приняты ни одним из них. Это обеспечивается широковещательным режимом передачи и использованием метода подтверждения приема сообщений. Если в сети никто не передает сообщения, то есть сеть на данный момент является свободной, каждый из узлов может начать передавать сообщение, но при этом любое сообщение обладает своим приоритетом при получении доступа к шине. Именно из-за этого передача может осуществляться только от одного устройства, которое обладает сообщением с самым большим приоритетом.

Соперничество за доступ к шине осуществляется следующим образом. Если два или большее количество устройств понимают, что линия на данный момент свободна и начинают передачу сообщений одновременно, то возникает конфликт, который решается с помощью побитового сравнения идентификатора передаваемого сообщения с ситуацией на линии. В ходе арбитража (улаживание конфликта) все устройства сравнивают логический уровень передаваемого бита с логическим уровнем на шине. Если получается, что уровни оказываются одинаковыми, то устройства продолжают отправлять следующий бит идентификатора. Если приемник устройства отображает, что на данный момент на шине доминирующий уровень, а передатчик в то же самое время показывает рецессивный уровень, то устройство в этот же момент перестает передачу этого сообщения. Такой механизм гарантирует, что время и информация не будут потеряны.

Для большой точности передачи данных протокол предполагает особые методы обнаружения ошибок, сигнализации и самоконтроля, которые реализуются в каждом из узлов сети. Если какой либо из узлов находит ошибки, то такое сообщения находятся в том узле, где она была найдена и помечается флагом. Такое сообщение не передается и происходит повторная передача автоматически. Промежуток времени между моментом нахождения ошибки до начала следующей попытки передачи равно длительности 31-го бита, если не обнаруживается новая ошибка.

Протокол CAN использует в качестве электрического соединения витую пару. И этот кабель в сети должен обладать общим (третьим) проводом, так же необходимо наличие на обоих концах кабеля согласующих резисторов, а сопротивление выбирается равным волновому сопротивлению кабеля. У этого кабеля есть ограничение на максимальную длину, и она составляет 1 км. Но если необходимо иметь большую длину кабеля или количество узлов, зачастую применяют повторители интерфейсов, шлюзы или сетевые мосты.

Витая пара может использоваться экранированная или нет, это решение принимается исходя из электромагнитной обстановки. У передатчиков CAN есть режим, который управляет длительностью импульсов. Этот режим осуществляется при помощи заряда емкостей затворов выходных транзисторов от источников тока, причем величина тока задается внешним резистором [3]. При увеличении длительности фронта можно снизить требования к согласованию линии на низких частотах, так же сделать большую длину отводов и уменьшить излучение электромагнитных помех.

В CAN-протоколе определены типы кадров:

1) кадр данных – передает данные с передатчика на приемник;

2) кадр удаленного запроса – узел сети запрашивает передачу другим узлом кадра данных с конкретным идентификатором;

3) кадр перегрузки – служит гарантией задержки между передаваемыми кадрами, чтобы совершать управление потоком данных, то есть выдается узлом, который не готов к приему сообщения. Он состоит из двух полей: флага

перегрузки и поля разделителя. Существуют два условия, при совершении которых происходит передача кадра перегрузки, первый это перегрузка приемника, вследствие которого необходимо увеличить паузу получаемыми им кадрами и второй, это выявление доминантного бита, находящегося на месте первого и второго бита в поле перерыва между кадрами;

4) кадр ошибки – показывает, какой узел нашел ошибку шины или сети, это могут быть ошибки контрольной суммы, подтверждения, формата кадров, передачи бита или битстаффинга. Первое поле в кадре ошибки является флаг ошибки. Такое сообщение имеет самый большой приоритет, вследствие этого оно передается сразу, как происходит обнаружение ошибки и принимается всеми устройствами одновременно. Также в одно и то же время все устройства совершают удаление из свой памяти сообщение, которое содержит ошибку.

Рассмотрим подробнее кадр данных (рисунок 1.2).

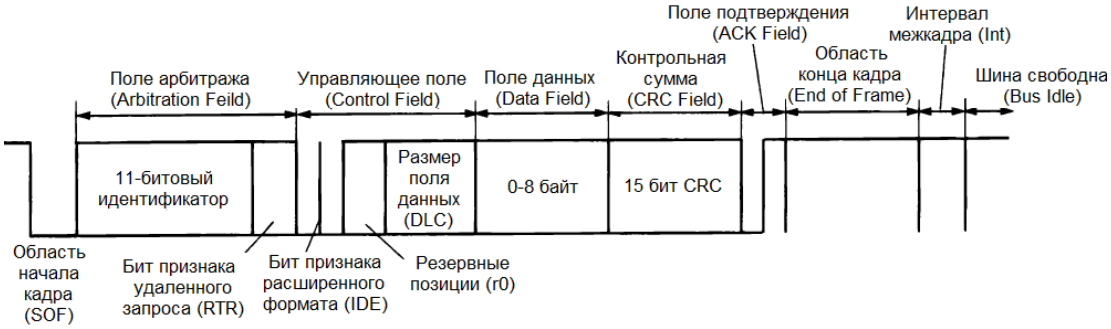

Рисунок 1.2 – Стандартный кадр данных

Он состоит из:

– стартового поля (SOF);
– поля арбитража (Arbitration Field);
– управляющего поля (Control Field);
– поля данных (Data Field);
– поля контрольной суммы (CRC);
– поля подтверждения (ACK Field);
– поля конца фрейма (EOF) [2].

Поле SOF располагается в начале кадра данных и кадра удаленного запроса. Это поле содержит один доминирующий бит.

Поле арбитража включает в себя 11-битовый идентификатор RTR-бит, который показывает, является ли данных кадр кадром данных или удаленным кадром. Идентификатор необходим для адресации сообщений и применяется механизм арбитража.

Управляющее поле кадра данных включает в себя 6 битов, 4 бита из которых определяют поле Data Length Code, которое показывает количество байт данных, передающихся в поле данных, 2 оставшихся бита зарезервированы для последующей редакции протокола.

Поле данных включает в себя передаваемые данные, притом количество байтов, которые необходимо передать, указываются в управляющем поле и не превышает 8. Поле CRC реализуем механизм избыточного контроля по четности передаваемых данных.

Поле подтверждения данных включают в себя участки ACK Slot и ACK Delimiter и делают следующую функцию: передающий узел отправляет по одному рецессивному биту на каждом из участков, а приемник, если он получил сообщение без сбоев, устанавливает на линии доминирующий бит в поле ACK Slot [2].

Поле конца кадра содержится в кадре данных и удаленном кадре и включает в себя 7 рецессивных битов.

Кадр удаленного запроса такой же по структуре, как и кадр данных, но не содержит в себе поля данных, а кадр ошибок и кадр перегрузки содержать два поля: в первом находятся флажки ошибок и служебная информация. А второе поле является разграничителем и имеет 8 рецессивных битов.

1.3 CANopen

1.3.1 Сведения о CANopen

CANopen можно назвать одним из самых распространенных приложений для прикладного уровня. Оно является приложением, которое поддерживает ассоциация CiA. У CiA существует профиль устройства (CiA DS 40x), который делает проще соединение модулей различных производителей в одну сеть, так же определяют минимально обязательный набор свойств модулей, который обеспечивает правильную работу системы на базовом уровне.

Изначально CANopen был разработан для сетей управления движущимися механизмами в системах промышленной автоматизации. Однако со временем его начали применять в медицине, в системе автоматизации зданий, морской электроники. К примеру, в морской электронике CANopen отвечает за пожарную сигнализацию, грузовую сигнализацию, навигационную систему и другие системы.

На рисунке 1.3 приведена архитектура протокола CANopen в соответствии с моделью OSI.

Два нижних уровня описываются стандартом CAN (ISO 11898, CAN SPECIFICATION 2.0 A/B). CANopen включает в себя особые правила битового квантования, также описывает 3 рекомендованных типа соединения: 9-контактный DSUB (DIN 411652), 5-контактный круглый Mini (ANSI/B93.55M1981), 5-контактное открытое клеммное соединение [2].

В сети CANopen предусмотрены 8 уровней скорости передачи данных: 1Мбит/с, 800, 500, 250, 152 ,50 20, и 10 кбит/с. Скорость 20 кбит/с считается обязательной для всех модулей.

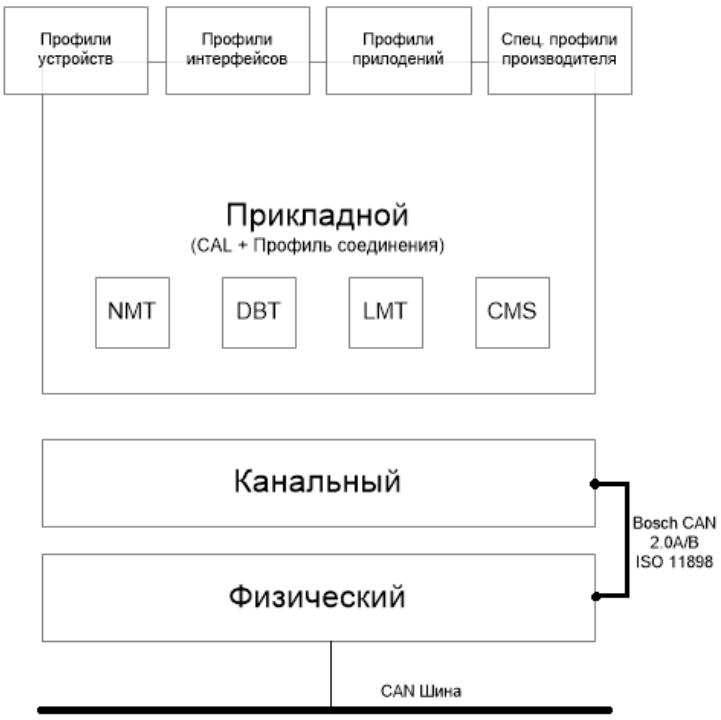

Рисунок 1.3 – Архитектура протокола CANopen

1.3.2 Прикладной уровень: CANopen

Прикладной уровень модели OSI осуществляет интерфейс между сетью и программным приложением, которое взаимодействует с похожими приложениями в других устройствах сети. На прикладном уровне осуществляется механизм синхронизации между устройствами.

Так как протокол CAN описывает только два первых уровня, то это привело к появлению множества несовместимых между собой протоколов прикладного уровня. Самыми распространенными протоколами этого уровня являются CANopen и DeviceNet. Для более простого применения этого стандарта ввели несколько присущих только CANopen понятий. У прикладного уровня есть функциональные возможности, и они делятся между сервисами (элементы услуг). Программные приложение осуществляют общение между собой с помощью вызова необходимых сервисов прикладного уровня. Сервисы же в свою очередь осуществляют обмен данными с одноранговыми сервисами с помощью CAN-сети

через определенный протокол. Данный протокол описывается в спецификации протокола сервиса.

Для взаимодействия программных приложений с прикладным уровнем с помощью языковой конструкции ввели понятие сервисного примитива. CANopen присущи 4 разных примитива:

1) запрос, осуществляемый приложением к прикладному уровню, который публикуется приложением для вызова сервиса;

2) индикация, которая публикуется прикладным уровнем для приложений, чтобы осведомить о внутренних событиях, которые обнаружил прикладной уровень или чтобы сообщить, что сервис запрошен;

3) ответ, который публикуется приложением для прикладного уровня, чтобы дать ответ на ранее полученную индикацию;

4) подтверждение, которое публикуется прикладным уровнем для приложения, чтобы сообщить об итоге ранее переданного запроса.

Устройство в сети CANopen представляется состоящим из 3 частей:

1) коммуникационный интерфейс (к шине CAN) и программный протокол обмена;

2) словарь объектов;

3) интерфейс к устройствам ввода-вывода и прикладная программа [3].

Коммуникационный интерфейс и программный протокол предоставляют сервис по подаче и получению через сеть коммуникационных объектов. В словаре объектов описываются типы данных, прикладные и коммуникационные объекты, которые используются на устройстве для обмена с помощью интерфейса к устройствам ввода вывода. Прикладная программа необходима для обеспечения внутреннего управления функциями устройства, а так же интерфейсом к устройствам ввода-вывода.

Словарь объектов считается самой значимой частью устройств CANopen. Объекты представляют собой профили устройств, типы данных, регистры ошибок. В объектном словаре каждый элемент адресуется 16-битным индексом.

Могут использоваться такие типы данных: Float, Time, Boolean, Date, Integer, UnsignedN, которые обладают общепринятым смыслом.

Главными составляющими прикладного уровня являются четыре основных сервисных элемента: CMS, LMT, NMT и DBT. CANopen обладает модулями на прикладном уровне, которые могут обмениваться между собой объектами-сообщениями – COB (Communication Object) которые включают в себя один или несколько CAN-кадров. Есть всего 4 типа таких объектов:

1) объекты данных процесса – Process Data Object (PDO);
2) объекты сервисных данных – Service Data Object (SDO);
3) объекты специальных функций – Special Function Objects;
4) объекты сетевого управления – Network Management Object [2].

При передачи данных можно использовать два различных механизма – на основе SDO и с использованием PDO. Объекты PDO и SDO необходимы для передачи данных.

SDO дают возможность модулям посылать друг другу данные любого объема в ацикличном низко приоритетном режиме. Чаще всего такой тип обмена применяется для конфигурации устройств или настройки формата PDO. Каждое устройство, которое включается в сеть CANopen, обязано поддерживать SDO-обмен.

PDO используется для синхронной или асинхронной скорости передачи не более 8 байтов, что соответствует длине поля данных кадра CAN). PDO обладает более высоким приоритетом, чем SDO, и чаще всего, его применяют для пересылки данных в режиме реального времени. У PDO можно выделить два типа объектов. Благодаря первому осуществляется передача данных, а второй необходим для приема данных.

Для выполнения особых задач, также обусловленных режимом реального времени, существуют объекты специальных функций:

1) синхронизация – Synchronization Object (SYNC) – необходима для запуска синхронных процессов;

2) объект штампа времени – Time Stamp Object – включается в себя значение абсолютного времени;

3) объект аварии – Emergency Object (EMCY) – необходим для отправки кодов ошибок модулей.

Объекты сетевого управления включают в себя сообщения сервисов DBT (сообщения распределения идентификаторов), LMT (Сообщения управления уровнями), NMT (сообщения управления сетью). Администрирует сеть NMT-мастер, который совершает инициализацию устройства, совершает контроль ошибок, производит их периодическую перекличку при помощи PDO-сообщений, чтобы выявить узлы, которые находятся отключенными от шины по счетчику ошибок или в нерабочем состоянии вследствие физического отсутствия. Запуск NMT-узла представлен в приложении А.

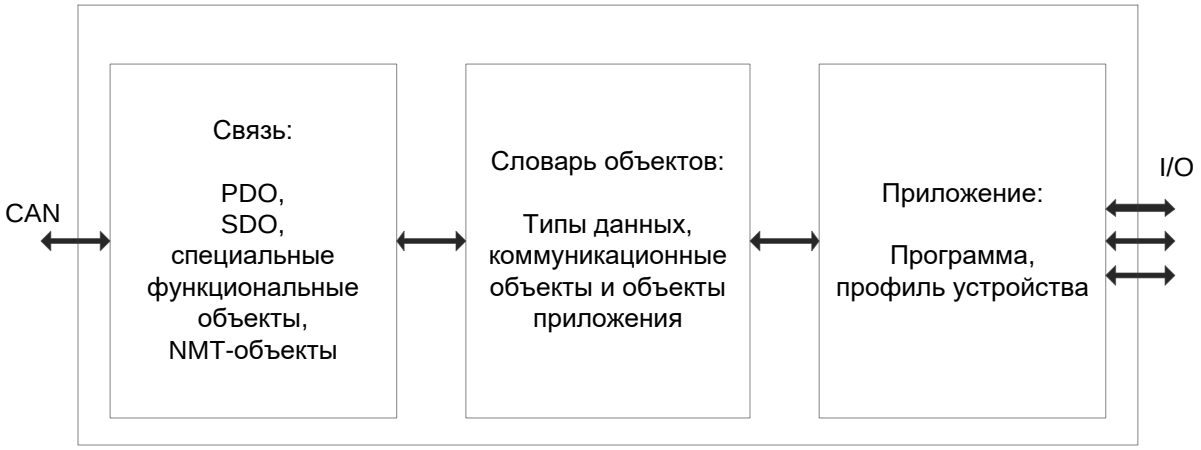

Рисунок 1.4 – Модель устройства с интерфейсом CANopen

1.3.3 Электронные спецификации устройств CANopen

Так как устройства, которые используются в сети программируемые, перед тем, как их включить в сеть требуется задать параметры, которые необходимы для их коммуникации с сетью и дальнейшего функционирования. CANopen предоставляет для этого стандартизованный метод. Этот метод подразумевает, что есть электронное описание устройств в текстовом формате, для отработки

которого необходим только несложный компилятор. Словарь объектов описывает группы доступных объектов и распространяется производителями CANopen-устройств в виде Electronic Data Sheet (текстового файла *.eds). CANopen описывает формат EDS (Electronic Data Sheet – электронный список параметров), описывающий конфигурацию и параметры устройств используемых в сети, так же в это описание входят контроллеры с модульной архитектурой.

EDS поставляются и поддерживаются производителями устройств. А если такого нет, то EDS содержится как настройка по умолчанию и является общей для определенного класса устройств, к примеру, датчик температуры, датчик давления. EDS представляет собой текстовый файл, который использует ASCII-коды (является набором символов по стандарту ISO 646). В таком файле длина строки равна 255 символов, необходимо, чтобы каждая строка оканчивалась символом CR или LF.

Файл EDS включает в себя информацию о самом файле, то есть его имя, дату создания, версию EDS, описание и другое. Так же в таком файле есть общая информация об устройстве, это имя производителя, имя устройства, номер версии, функции устройства. Есть еще конфигурационные параметры, это типы данных, тип устройства, длительность цикла обмена. Структура EDS файла полностью описана в стандарте [5].

На данный момент доступен стенд в одной из лабораторий университета, у которого есть шина CAN. Данный стенд помогает ознакомиться с работой протокола CAN, на нем есть всё необходимое для отработки знаний протокола по дисциплине «Промышленные сети и системы связи». Но при работе на нем было выявлено, что он не соответствует спецификации CANopen, принятой в промышленной автоматике. Поэтому возникла идея реализовать имитатор датчика температуры с интерфейсом CANopen, который бы соответствовал стандартам.

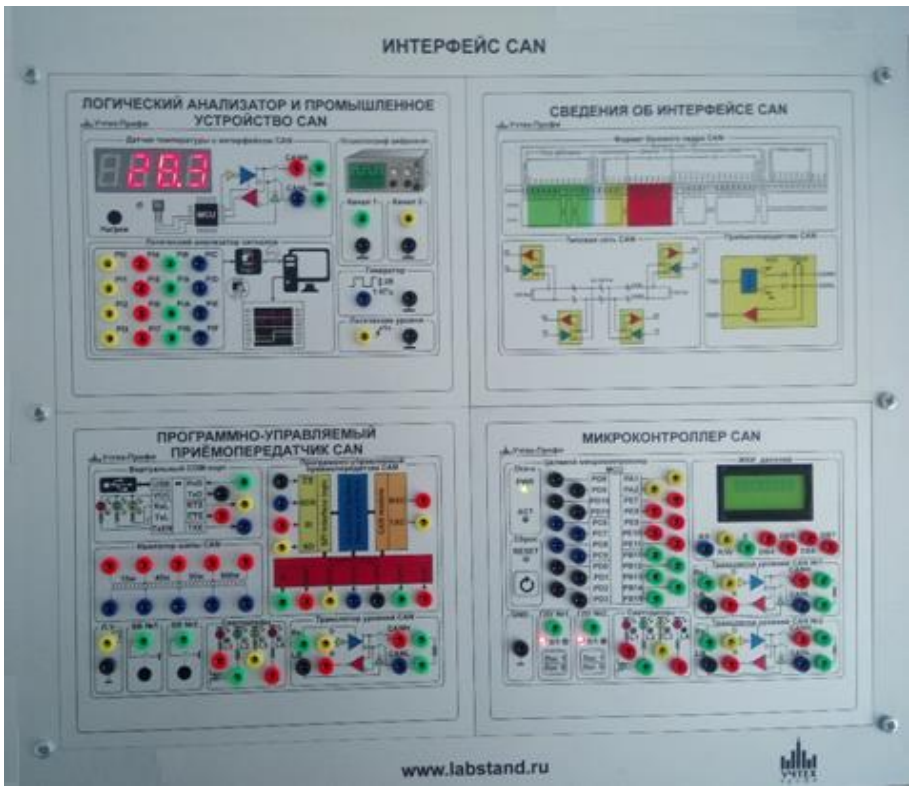

Рисунок 1.5 – Стенд CAN

Промышленное оборудование с использованием промышленных протоколов стоит достаточно дорого, поэтому можно попробовать построить дешевый аналог, который позволил бы отрабатывать навыки по работе с промышленным протоколом CANopen. Для успешного освоения необходим лабораторный практикум.

2 РАЗРАБОТКА СТРУКТУРНОЙ СХЕМЫ И ВЫБОР ОБОРУДОВАНИЯ

2.1 Структурная схема

Проанализировав элементы, которые необходимы для организации работы лабораторной установки, была составлена структурная схема.

Для корректной работы установки нужен процессорный модуль, который будет совершать обработку управляющих сигналов. Также необходим блок питания, который обеспечивает подачу электроэнергии для осуществления работы. Элементы управления необходимы для подачи управляющих сигналов, в качестве такого элемента выступает кнопка. Чтобы устройство могло работать с сетью CAN, нужен драйвер интерфейса CAN. Для того, чтобы наглядно видеть результат проделанной работы нужен блок индикации, в нашем случае это экран.

Разработанная структурная схема представлена в документе 270304.2018.340.02.01 С1.

На основе этой структурной схемы выберем необходимое оборудование.

2.2 Выбор и анализ оборудования

Произведем анализ возможного варианта воплощения структурной схемы в реальные элементы. В связи с этим произведем исследования. Попробуем реализовать каждый элемент отдельным оборудованием, проанализируем получившееся устройство.

2.2.1 Процессорный модуль

В качестве узла «Процессорный модуль» можно рассмотреть микроконтроллер STM32F407VGT6 [13]. STM32F407VGT6 является 32-битным ARM-микроконтроллером семейства Cortex-M4.

Характеристики STM32F407VGT6 представлены в таблице 2.2.

Таблица 2.2 – Характеристики STM32F407VGT6

Наименование параметра	Значение
Частота, МГц	168
Память программ, Мбайт	1
ОЗУ (RAM), кбайт	196
Ядро	ARM Cortex-M4
Количество входов/выходов	82
Напряжение питания, В	1,8…3,6
Корпус	LQFP100

Изображение STM32F407VGT6 можно увидеть на рисунке 2.2.

Рисунок 2.2 – Микроконтроллер STM32F407VGT6

Так же в качестве узла «Процессорный модуль» можно использовать микроконтроллер STM32F429ZIT6U [57]. STM32F429ZIT6U является 32-битным ARM-микроконтроллером семейства Cortex-M4.

Характеристики STM32F429ZIT6U представлены в таблице 2.3.

Таблица 2.3 – Характеристики STM32F429ZIT6U

Наименование параметра	Значение
Частота, МГц	168
Память программ, кбайт	152
ОЗУ (RAM), кбайт	192
Ядро	ARM Cortex-M4
Количество входов/выходов	82
Напряжение питания, В	1,8…3,6
Корпус	LQFP100

Изображение STM32F429ZIT6U можно увидеть на рисунке 2.3.

Рисунок 2.3 – Микроконтроллер STM32F429ZIT6U

2.2.2 Блок питания

В качестве узла «Блок питания» можно рассмотреть Chieftec GPE-600S 600W [14]. В блоке питания Eco сочетается максимальная эффективность и экологичность, в том числе обладает максимальной гибкостью в повседневной жизни. Изображение блока питания можно увидеть на рисунку 2.4.

Рисунок 2.4 – Блока питания Chieftec GPE-600S 600W

Характеристики блока питания представлены в таблице 2.4

Таблица 2.4 – Характеристики Chieftec GPE-600S 600W

Наименование параметра	Значение
Суммарная мощность, Вт	600
Ток по линии +12 В, А	45
Ток по линии +5 В, А	22
Ток дежурного источника (+5 В Standby)	2,5
Диапазон входного напряжения сети, В	230

2.2.3 Элемент управления

В качестве узла «Элемент управления» можно рассмотреть кнопку MJ-DS430 250V 1.0A OFF-(ON), без фиксации [15]. Кнопка изображена на рисунке 2.3.

Характеристики блока питания представлены в таблице 2.5.

Таблица 2.5 – Характеристики MJ-DS430 250V 1.0A OFF-(ON)

Наименование параметра	Значение
Номинальный ток, А	1
Номинальное напряжение, В	250
Сопротивление контактов, мОм	50
Сопротивление изоляции, МОм	100

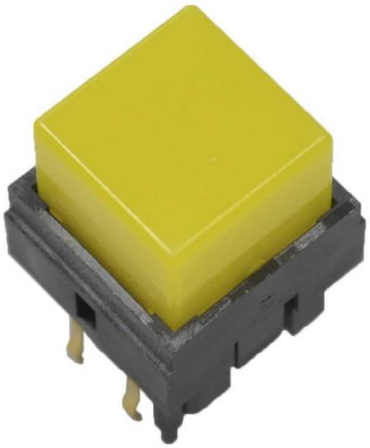

Рисунок 2.5 – Кнопка MJ-DS430 250V 1.0A OFF-(ON) желтая

Так же в качестве узла «Элемент управления» можно рассмотреть кнопку KLS7-TS6601-7.0-180 (TC-0104) [5]. Кнопка изображена на рисунке 2.6.

Характеристики кнопки KLS7-TS6601-7.0 представлены в таблице 2.6.

Таблица 2.6 – Характеристики кнопки KLS7-TS6601-7.0

Наименование параметра	Значение
Номинальный ток, А	0,05
Номинальное напряжение, В	12
Высота, мм	7
Способ монтажа	В отверстия на плату

Рисунок 2.6 – Кнопка KLS7-TS6601-7.0

2.2.4 CAN оборудование

2.2.4.1 Адаптер USB – CAN/RS485/RS232 «Меркурий 221»

В качестве узла «Драйвер интерфейса CAN» можно рассмотреть Адаптер USB – CAN/RS485/RS232 «Меркурий 221» [16]. Адаптер USB – CAN/RS485/RS232 «Меркурий 221» изображен на рисунке 2.4.1.

«Меркурий 221» является преобразователем интерфейса USB в CAN/RS-232/RS485, и рассчитан для подсоединения к персональному компьютеру одного или нескольких электросчетчиков «Меркурий» со встроенными интерфейсами CAN/RS-485. После того, как будет установлен драйвер USB, устройство определяется как виртуальный порт COM. Характеристики адаптера USB – CAN/RS485/RS232 «Меркурий 221» представлены в таблице 2.4.1.

Таблица 2.4.1 – Характеристики Адаптера USB – CAN/RS485/RS232 «Меркурий 221»

Наименование параметра	Значение
Максимальная длина линии, м	1000
Внешнее питание интерфейсов приборов, В	(5…9) ± 10%
Рабочая температура, °C	0…+50
Разъём для подключения к RS-232	DB9
Максимальная скорость передачи, бод	1115200

Рисунок 2.4.1 – Адаптер USB – CAN/RS485/RS232 «Меркурий 221»

2.2.4.2 USB-CAN (OBD)

Так же в качестве узла «Драйвер интерфейса CAN» можно рассмотреть USB-CAN (OBD) [17]. USB-CAN (OBD) изображен на рисунке 2.4.2.

Рисунок 2.4.2 – USB-CAN (OBD)

USB-CAN (OBD) является 1-портовый преобразователем USB в CAN с гальванической изоляцией в металлическом корпусе. Есть режим прослушивания шины, имеется защита от импульсных помех.

Характеристики USB-CAN (OBD) представлены в таблице 2.4.2.

Таблица 2.4.2 – Характеристики USB-CAN (OBD)

Наименование параметра	Значение
Скорость CAN, Мбит/с	до 1
CAN контроллер	STM
CAN трансивер	MCP2551
Изолятор	ADUM1201
Разъём CAN	DB9

2.2.4.3 Преобразователь USB-to-CAN V2 IXXAT

Так же в качестве узла «Драйвер интерфейса CAN» можно рассмотреть USB-to-CAN V2 IXXAT [30]. USB-to-CAN V2 IXXAT изображен на рисунке 2.4.3.

Рисунок 2.4.3 – Преобразователь USB-to-CAN V2 IXXAT

USB-to-CAN V2 предоставляет пользователям существенные преимущества. Интерфейс USB-CAN V2 использует мощные аппаратные средства, которые позволяют обеспечить скорость соединения через USB 2.0 до 480 Мбит/с. К достоинствам можно отнести превосходную прочность, которая позволяет продлить срок эксплуатации устройства в условиях тяжелой окружающей среды, отличает простота в эксплуатации при конкурентной цене. Доступен режим прослушивания шины, имеется защита от импульсных помех.

Характеристики преобразователя USB-to-CAN V2 IXXAT блока питания представлены в таблице 2.4.3.

Таблица 2.4.3 – Характеристики преобразователя USB-to-CAN V2 IXXAT

Наименование параметра	Значение
Память RAM, кБайт	512
Память FLASH, кБайт	512
USB 2.0 Hi-Speed, Мбит/сек	480
Подключение к шине	2 × RJ45

2.2.4.4 Плата для подключения микроконтроллеров к CAN сети SN65HVD230 CAN Board

Так же в качестве узла «Драйвер интерфейса CAN» можно рассмотреть SN65HVD230 CAN Board [18]. SN65HVD230 CAN Board показан на рисунке 2.4.2.

Рисунок 2.4.4 – SN65HVD230 CAN Board

SN65HVD230 CAN Board является коммуникационной платой, которая позволяет подключать микроконтроллерные системы к CAN сети. Плата построена на базе чипа SN65HVD230. Характеристики SN65HVD230 CAN Board представлены в таблице 2.4.4.

Таблица 2.4.4 – Характеристики SN65HVD230 CAN Board

Наименование параметра	Значение
Рабочее напряжение, В	3,3
Тип выходного интерфейса	CAN
Тип входного интерфейса	MCU
Рабочая температура, °С	-40…+85

2.2.4.5 CAN SPI click MIKROE-986

Так же в качестве узла «Драйвер интерфейса CAN» можно рассмотреть CAN SPI click MIKROE-986 [19]. MIKROE-986 изображена на рисунке 2.4.5.

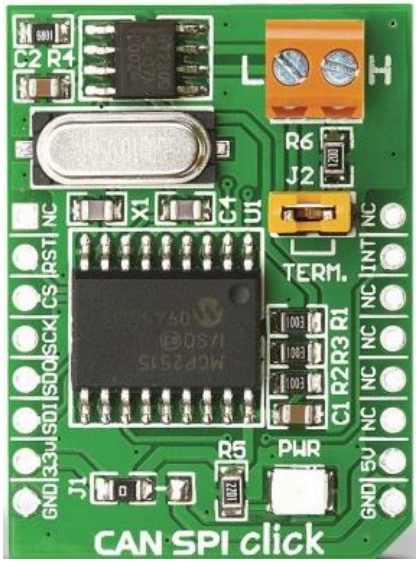

Рисунок 2.4.5 – CAN SPI click MIKROE-986

CAN SPI click MIKROE-986 обеспечивает последовательную связь физического уровня в соответствии со стандартом ISO 11898. Он необходим для деятельности в жестких условиях окружающей среды. Характеристики CAN SPI click MIKROE-986 представлены в таблице 2.4.5.

Таблица 2.4.5 – Характеристики CAN SPI click MIKROE-986

Наименование параметра	Значение
Напряжение питания, В	3,3
Скорость, Мб/с	до 1
Номер процессора	MCP2551
Потребление в режиме ожидания, мкА	370

2.2.5 Блок индикации

В качестве узла «Блок индикации» можно рассмотреть LCD дисплей 128x64 синий (ST7920) [20]. LCD дисплей 128x64 синий (ST7920) изображен на рисунке 2.5.1.

LCD дисплей 128x64 синий (ST7920) изображен на рисунке 2.5. LCD дисплей (ST7920) является графическим монохромным ЖК дисплеем с разрешением 128*64 точек.

Характеристики LCD дисплей 128x64 синий (ST7920) представлены в таблице 2.5.1

Таблица 2.5.1 – Характеристики LCD дисплей 128x64 синий (ST7920)

Наименование параметра	Значение
Разрешение	128*64
Цвет подцветки дисплея	Синий
Интерфейс	SPI, параллельный
Контроллер дисплея	ST7920
Напряжение питания модуля, В	4,5…5,5
Напряжение питания подсветки, В	3…3,6
Ток потребляемый модулем, мА	до 5
Ток потребляемый подцветкой, мА	до 60

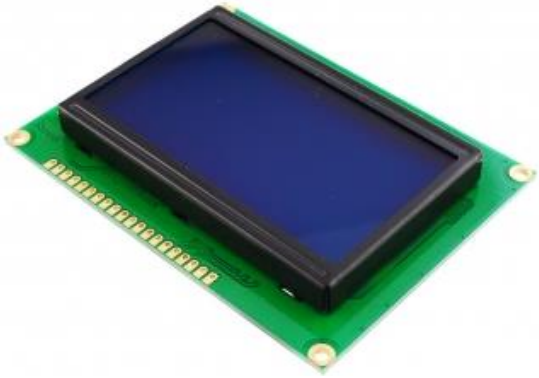

Рисунок 2.5.1 – LCD дисплей 128x64 синий (ST7920)

В качестве узла «Блок индикации» можно LCD дисплей 2004 (синий) [21]. LCD дисплей 2004 (синий) изображен на рисунке 2.5.2.

LCD дисплей 2004 (синий) – жидкокристаллический дисплей (Liquid Crystal Display) экран которого способен отображать одновременно до 80 символов (20 столбцов, 04 строки) [22].

Характеристики LCD дисплей 2004 (синий) представлены в таблице 2.5.2

Таблица 2.5.2 – Характеристики LCD дисплей 2004 (синий)

Наименование параметра	Значение
Чип	HD44780
Цвет подцветки дисплея	Синий
Интерфейс	Синхронный, 8-битный, параллельный
Напряжение питания, В	5
Рабочая температура, °С	-20…+70

Рисунок 2.5.2 – LCD дисплей 2004 (синий)

В качестве узла «Блок индикации» можно использовать 2,4" QVGA TFT LCD [26]. 2,4" QVGA TFT LCD изображен на рисунке 2.5.3. Графические возможности монитора значительны, их достаточно для создания изображения клавиатуры работающей благодаря сенсорным поверхностям. Характеристики 2.4" QVGA TFT LCD представлены в таблице 2.5.3

Таблица 2.5.3 – Характеристики 2,4" QVGA TFT LCD

Наименование параметра	Значение
Разрешение	240×320
Контроллер	SPFD5408
Интерфейс	8-битный
Напряжение питания, В	2,4…3,3
Рабочая температура, °С	-20…+70

Рисунок 2.5.3 –Дисплей 2,4" QVGA TFT LCD

В результате выбора компонентов было принято решение совместить несколько элементов структурной схемы в одно устройство. Таким устройством может послужить отладочная плата.

2.2.6. Отладочная плата

2.2.6.1 Отладочная плата STM32F429ZI-Discovery

Попробуем найти отладочную плату, в состав которой входило бы несколько элементов структурной схемы. В качестве такой платы рассмотрим STM32F429ZI-Discovery. Эта плата представлена на рисунке 2.6.1.

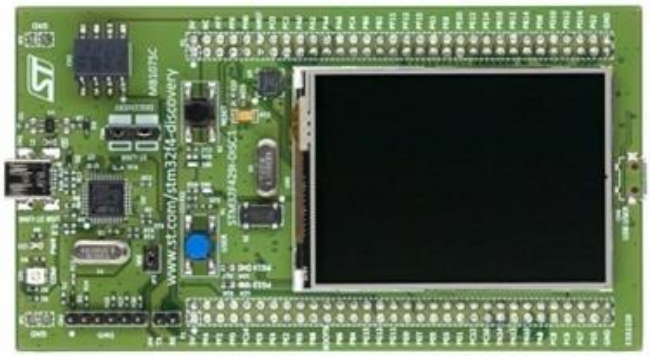

Рисунок 2.6.1 – STM32F429ZI-Discovery

Такая плата совмещает в себе несколько блоков структурной схемы, что делает готовое устройство более дешевым и простым в реализации. В этой отладочной плате присутствует блок индикации, то есть экран, есть в составе кнопка для управления, так же присутствует процессорный модуль и блок питания от USB.

Характеристики платы STM32F429ZI-Discovery приведены ниже:

– микроконтроллер STM32F429ZIT6, ARM Cortex-M4;
– оперативная память (RAM) 256 Кбайт;
– функции USB: виртуальный COM порт, отладочный порт;
– питание платы: через USB или от внешнего источника 3 или 5В;
– дисплей: 2,4" QVGA TFT LCD;
– разрядность шины данных 32 бита;
– две кнопки (пользовательская и сброс);
– ST MEMS 3-осевой цифровой гироскоп L3GD20 [23].

Для организации ввода значений воспользуемся кнопками, которые подключены по порты этой платы. Такая цена ниже, если собирать все элементы структурной схемы по отдельности. Это является ее главным преимуществом. Такая плата не включает в себя только блок интерфейса CAN.

2.2.6.2 Отладочная плата STM32F401C-Discovery

Если собирать все элементы по отдельности, то цена будет большой, попробуем собрать необходимые элементы более простым оборудованием, в состав которого будут входить сразу несколько блоков структурной схемы. Попробуем найти отладочную плату, в состав которой входило бы большее количество элементов. В качестве такой платы рассмотрим STM32F401C-Discovery. Эта плата представлена на рисунке 2.6.2.

Рисунок 2.6.2 STM32F401C-Discovery

Такая плата совмещает в себе несколько блоков структурной схемы, что делает готовое устройство более дешевым в реализации. В этой отладочной плате присутствует кнопка для управления, так же есть процессорный модуль и блок питания от USB.

Характеристики платы STM32F401C-Discovery приведены ниже:

– микроконтроллер STM32F401VCT6, 256 КБ Flash, 64 КБ RAM, корпус LQFP100;

– интегрированный ST-LINK/V2;

– напряжение питания +5 В;

– L3GD20 3-осевой гироскоп;

– LSM303DLHC акселерометр и магнетометр;

– MP45DT02 аудио датчик;

– CS43L22 аудио ЦАП;

– 8 светодиодов;

– 2 кнопки;

– USB OTG [28].

К её недостатку можно отнести отсутствие экрана, как на предыдущей плате.

2.2.6.3 Отладочная плата STM32F746G-Discovery

Рассмотрим другой пример отладочной платы, в состав которой входило бы большее количество элементов. В качестве такой платы рассмотрим STM32F746G-Discovery. Эта плата представлена на рисунке 2.6.3.

Рисунок 2.6.3 STM32F746G-Discovery

Такая плата совмещает в себе несколько блоков структурной схемы, что делает готовое устройство более дешевым в реализации. В этой отладочной плате присутствует блок индикации, то есть экран, есть в составе кнопка для управления, так же присутствует процессорный модуль и блок питания от USB.

Характеристики платы STM32F746G-Discovery приведены ниже:

– микроконтроллер STM32F769NIH6, ядро Arm Cortex-M7 + FPU, 2MB Flash/512+16+4KB RAM, USB OTG HS/FS, 28 com IF, LCD, DSI;

– встроенный программатор/отладчик ST-LINK/V2-1;

– USB ST-LINK с ренумерацией и тремя интерфейсами: виртуальный COM порт, устройство хранения, порт отладки;

– 4-дюймовый емкостной сенсорный LCD дисплей с MIPI DSI разъемом;

– SAI аудио кодек, с гнездом стерео гарнитуры, включая аналоговый вход микрофона и выход громкоговорителя;

– две кнопки: пользовательская и сброс;

– память 512-Мбит Quad-SPI Flash;

– 128-Мбит SDRAM;

– слот для MicroSD карт;

– разъем для подключения дочерней платы Wi-Fi или Ext-EEP;

– разъем Ethernet совместимый с IEEE-802.3-2002;

– пять способа питания платы [29].

Характеристики этой платы очень хороши, но нам достаточно и более простой платы. К её недостатку можно отнести высокую стоимость.

2.7 Метод ранжировки

Воспользуемся методом ранжировки для выбора оптимального оборудования.

В рассматриваемом методе необходимо произвести нумерацию всех необходимых критериев [32]. Результатом данной процедуры является нумерация всех критериев. Ранг каждого из критериев можно определить его номером, если на его месте в ряду отсутствуют какие-либо другие. Если же несколько неразличимых критериев располагаются на одном месте, то ранг каждого из них равен среднему арифметическому их новых номеров [31].

Рассматриваемые критерии: q1 – компактность, q2 – надежность, q3 – технические характеристики, q4 – стоимость, q5 – доступность.

Имеется ряд упорядоченных критериев q1, q2, … , q5:

q5; q4; q3; q1; q2.

Ранги критериев, вычисленные в соответствии с вышеуказанной процедурой, сведены в таблицу 2.7.1.

Таблица 2.7.1 – Ранги критериев

i	1	2	3	4	5
r_i	4	5	3	2	1

Переход от рангов к коэффициентам C_i производится на основе гипотезы о линейной зависимости между рангом и относительной ценностью критерия [32]. Чем выше ранг, тем менее важным является критерий и наоборот, чем ранг ниже, тем более важным является рассматриваемый критерий. Определение коэффициентов C_i для произвольного r ($1 \leq r_i \leq n$) производится в соответствии со следующей формулой:

$$C_i = 1 - \frac{r_i - 1}{n}, \qquad (2.1)$$

где r – ранг рассматриваемого критерия;

n – количество критериев.

Произведем расчет C_i по порядку.

$C_1 = 1 - \frac{4-1}{5} = 0{,}4$, $C_2 = 1 - \frac{5-1}{5} = 0{,}2$, $C_3 = 1 - \frac{3-1}{5} = 0{,}6$, $C_4 = 1 - \frac{2-1}{5} = 0{,}8$, $C_5 = 1 - \frac{1-1}{5} = 1$

Коэффициенты C_i сведены в таблицу 2.7.2.

Таблица 2.7.1 – Ранги критериев

i	1	2	3	4	5
C_i	0,4	0,2	0,6	0,8	1

Из таблицы следует, что самым важным критерием является доступность, далее идет стоимость и технические характеристики. Будем совершать выбор оборудования по двум основным критериям: доступность и стоимость.

Результатом работы данного раздела является структурная схема, которая необходима для выбора элементов для реализации поставленной задачи. При анализе каждого из узлов схемы было выбрано оборудование, которое удовлетворяло его нужды. При дальнейшем изучении возможного оборудования, было принято решение об объединении нескольких узлов структурной схемы с помощью одного элемента. Поэтому для прототипа устройства воспользуемся отладочной платой STM32F429ZI-Discovery, которая содержит несколько узлов: блок питания, управляющий элемент, блок индикации, процессорный модуль. Такой выбор устройства является оптимальным, так как в нем сочетается доступность, оптимальная стоимость и интеграция нескольких необходимых элементов в одном.

3 ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ CANOPEN

3.1 Стек протокола CANopen подчиненного устройства

Во 2 главе были выбраны элементы, с помощью которых можно реализовать устройство. Для того, чтобы все работало необходимо программное обеспечение.

Необходимо разработать программное обеспечение для микроконтроллера, чтобы он решал поставленную задачу.

Проведем сравнительный анализ. Написать программу для микроконтроллера для реализации интерфейса CANopen можно несколькими путями.

1) Самостоятельно с нуля. Этот путь является очень сложным и почти невозможным для одного человека. Ведь написать с нуля реализацию стека протокола очень сложно.

2) Можно воспользоваться готовыми библиотеками. Рассмотри три такие возможные библиотеки: CANopenNode, CANFestival, EMTAS.

Рассмотрим подробнее каждую из них. Опишем их преимущества и недостатки. Начнем рассмотрение по порядку.

3.1.1 Библиотека CANopenNode

CANopenNode является бесплатным стеком CANopen с открытым исходным кодом. Он написан на ANSI-C с применением объектно-ориентированного программирования [24]. Он может работать на разных микроконтроллерах, также и как отдельное приложение. В стеке есть функциональность master-устройств.

Переменные – для обмена (communication), устройства (device), пользовательские (custom) – организованы в CANopen OD (словаре объектов).

Реализованы функции:

– простой NMT master;

– NMT slave (запуск, остановка, сброс подчиненного устройства);

– контроль целостности сети по протоколу heartbeat (Heartbeat producer/consumer error control);

– связывание объектов PDO (PDO linking), динамическое отображение объектов OD (dynamic mapping) для быстрого обмена данными переменных процесса (process variables);

– ускоренная передача пакетов объектов SDO (expedited SDO), сегментированная передача SDO (segmented SDO) и блочная передача для сервисного доступа ко всем параметрам;

– SDO master;

– экстренные сообщения аварии (Emergency message);

– синхронизация (SYNC producer/consumer);

– энерго-независимое хранилище конфигурации (Non-volatile storage) [24].

Особенности CANopenNode.

1) CANopenNode изначально написан для 8/16 битных контроллеров. И если использовать 32-битный контроллер, то таймер может работать не очень адекватно. Ведь когда таймер переполняется, то он обнуляется, а это не будет происходить, так как он будет продолжать расти и механизм Heratbeat будет выдавать ошибку, что все узлы превысили время ожидания нового Heratbeat.

2) В механизме SDO только один контроллер может посылать с размером не более 8 байт, а если предполагается большее число контроллеров, то необходимо написать собственную настройку, которая бы использовала первые два байта PDO в виде управляющей информации. Это решение приводит к тому, что необходимо однозначно определять участников взаимодействия, что можно реализовать при помощи идентификаторов.

3) Настройка идентификаторов в рассмотренной библиотеке происходит в коде [25].

На рисунке 3.1 продемонстрирована диаграмма с типичной реализацией узла сети при помощи CANopenNode.

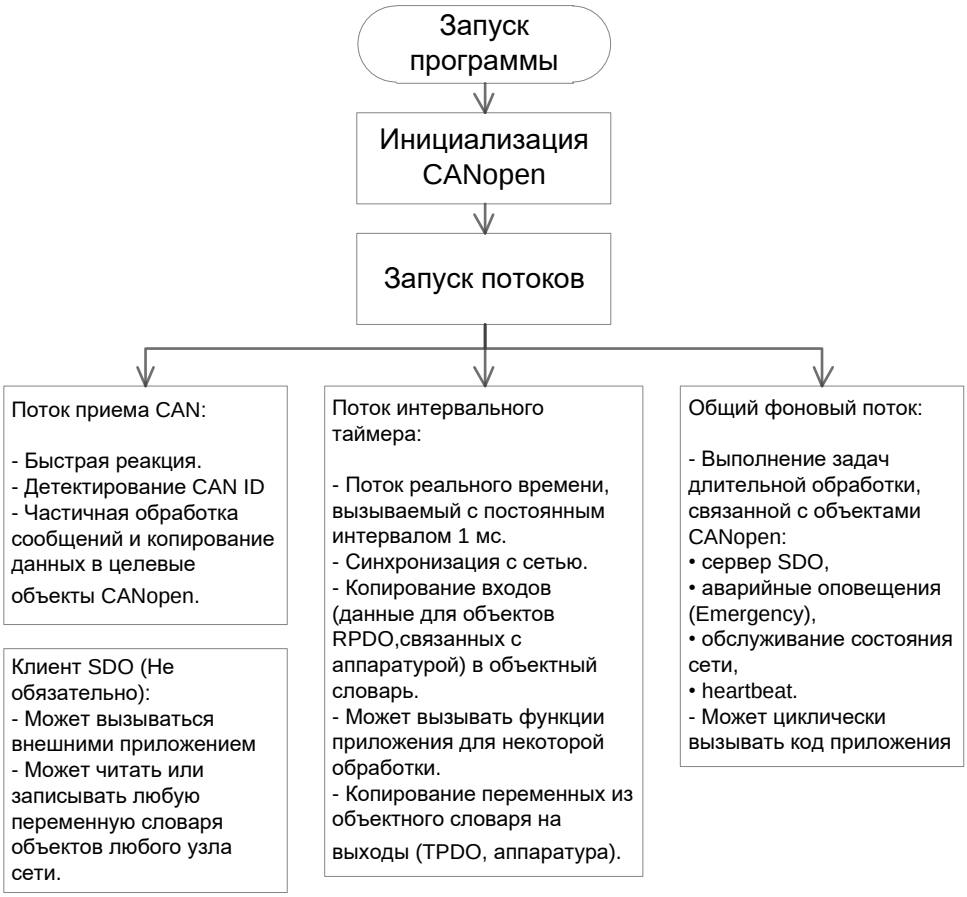

Рисунок 3.1 – Типичная реализация узла сети с применением CANopenNode

Библиотека CANopenNode в своем составе не имеет полный рабочий код для всех микроконтроллеров. Она является библиотекой со стеком и драйверами для нескольких различных микроконтроллеров. Получается, что библиотека CANopenNode может только использоваться как дополнительный подмодуль в проекте со специфической реализацией поддержки аппаратуры CAN и специфическим приложением [24].

3.1.2 Библиотека CANFestival

CANFestival является бесплатным инструментарием с открытым исходным кодом, он свободен от выплат при коммерческом использовании.

Он представляет рабочую среду разработки с помощью которой происходит создание подчиненных и главных устройств сети CANopen. На данный момент

CANFestival сфокусировал своё внимание на предоставлении независимой платформы ANSI-C стека CANopen, который можно реализовать на PC (Windows, Linux), межпроцессного обмена в реальном времени [27].

CANFestival поддерживает много платформ, что является её достоинством. Исходный код библиотеки CANFestival написан на C-ANSI. В CanFestival реализованы основные стандарты DS-301, DS-302, DS-305.

Основные характеристики:

1) NMT мастер и ведомый;
2) Heartbeat производитель и потребитель;
3) NodeGuard ведомое сообщение и основной мастер без трекинга;
4) служба SYNC;
5) поддерживает обмен SDO между сервером и несколькими клиентами;
6) PDO: возможно реализовать объекты TPDO и RPDO по отношению к типу передачи;
7) побитное изображение объектов на PDO;
8) служба EMCY, реализует отправление, прием и отслеживание объектов аварии;
9) типы данных: значения с разрядностью от 8 до 64 бит, строки фиксированной длины [27].

3.2 Программное обеспечение CANopen верхнего уровня

EMTAS CANopen DeviceDesigner является простым в использовании инструментом для быстрого и экономичного проектирования устройств CANopen. С помощью нескольких щелчков мыши объектный словарь устройства может быть создан на основе предопределенных профилей. Так же происходит настройка стека CANopen и драйвера CANopen с учетом характеристик устройства.

Основные моменты:

– быстрая разработка устройств с использованием стандартных, стандартизированных профилей;

– автоматическая генерация выходных данных из источников данных;

– согласованность объектного словаря, файла EDS и документации через общий источник;

– оптимальная и ресурсосберегающая конфигурация стека CANopen в соответствии с характеристиками устройства;

– легкая обработка с помощью входных масок для битовых кодированных значений.

В соответствии с профилями и вводом, объектный словарь создается в виде С файла исходного кода. Это интегрировано в приложение как интерфейс для стека протоколов. При соответствующей настройке можно получить доступ к содержанию словаря объектов непосредственно в виде С-переменной. В качестве альтернативы из приложения доступ к функциям возможен через index и subindex.

В соответствии с настройками и определениями объектного словаря создается файл конфигурации и инициализации в исходном коде языка Си. Это гарантирует, что только используемые сервисы стека CANopen будут скомпилированы и инициализированы.

CANopen DeviceDesigner является готовым продуктом, фирма EMTAS разработала её и продаёт. За её работоспособность отвечает фирма, все возможные ошибки, должны устраняться разработчиками. Она является готовой и полностью соответствует требованиям. К её недостаткам можно отнести ограниченную функциональность без оплаты. На данный момент это не является столь важным, так как без покупки функционирует 1 час, чего достаточно для целей, объявленных в ТЗ. Перезагрузка контроллера помогает решить проблему в использовании только 1 час и не приводит к последствиям. Её всегда можно произвести.

4 ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДАТЧИКА ТЕМПЕРАТУРЫ

4.1 Алгоритм работы имитатора датчика температуры

Необходимо описать работу датчика общими понятиями. Нужно, чтобы при нажатии на кнопку начала повышаться температура по рассчитанной математической модели. И при достижении определенного значения нагрев будет останавливаться. Каждое значение температуры передается в словарь объектов, которое пользователь может увидеть. Так же значение температуры должно отображаться пользователю. Пользователь может взаимодействовать с программой, чтобы смотреть значения параметров в словаре объектов, записывать их.

4.2 Имитация датчика температуры

Для описания нагревательного элемента воспользуемся математической моделью.

Математическое моделирование технических систем основывается на математических моделях. Математической моделью называется модель, записанная в форме математических соотношений, комплекс знаний, гипотез и представлений о происходящем процессе [38].

Вначале математическая модель записывается в виде алгебраических и дифференциальных уравнений, которые отображают динамические свойства объекта управления. Такие уравнения можно определить аналитически на основе законов физики, которые положены в основу работы объекта или экспериментально с применением методов пассивного или активного эксперимента.

Математическая модель нагревательного элемента определим на основе уравнения теплового баланса:

$$C_н \cdot m_н \cdot \frac{d\theta_н}{dt} = P_н - \Phi_{п.к}, \quad (4.1)$$

где $C_н$ – удельная теплоемкость нагревательного элемента;

$m_н$ – масса нагревательного элемента;

$\theta_н$ – температура нагревательного элемента;

$P_н$ – мощность нагревательного элемента;

$\Phi_{п.к}$ – поток теплоты, поступающий от нагревательного элемента к нагреваемому.

Поток теплоты, поступающий от нагревательного элемента к нагреваемому:

$$\Phi_{п.к} = \alpha_н \cdot F_н \cdot (\theta_н - \theta_к), \quad (4.2)$$

где $\alpha_н$ – коэффициент теплоотдачи нагревательного элемента;

$F_н$ – площадь поверхности нагревательного элемента;

$\theta_к$ – температура нагреваемого элемента.

С учетом формулы (4.2) и уравнение (4.1) после преобразований примет вид:

$$\frac{C_н \cdot m_н}{\alpha_н \cdot F_н} \cdot \frac{d\theta_н}{dt} + \theta_н = \frac{P_н}{\alpha_н \cdot F_н} + \theta_к. \quad (4.3)$$

Введем обозначения:

$$T_н = \frac{C_н \cdot m_н}{\alpha_н \cdot F_н}, \quad (4.4)$$

$$k_н = \frac{P_н}{\alpha_н \cdot F_н}. \quad (4.5)$$

И подставим их в уравнение (4.3). Получим окончательную запись:

$$T_н \cdot \frac{d\theta_н}{dt} + \theta_н = k_н \cdot P_н + \theta_к, \quad (4.6)$$

где $T_н$ – постоянная времени нагревательного элемента;

$k_н$ – коэффициент передачи нагревательного элемента.

Мощность нагревательного элемента определяется по формуле:

$$P_н = \frac{U_н^2}{R}, \tag{4.7}$$

где $U_н$ – напряжение на нагревательном элементе;

R – активное сопротивление нагревательного элемента.

Таким образом, динамика нагревательного элемента описана уравнениями (4.6) и (4.7).

Когда система начинает работать, в пределах малых отклонений будут изменяться все входные и выходные величины, так же и напряжение $U_н$ на нагревательном элементе и его мощность P соответственно относительно значений $U_{н0}$ и P_0 [38].

Для линеаризации уравнения (4.7) используем аналитический метод линеаризации, разложив его в ряд Тейлора, при $U=U_{н0}$.

$$P = P_0 + \left(\frac{dP}{dU}\right) \cdot \frac{(U_н - U_{н0})}{1!} + \left(\frac{d^2P}{dU^2}\right) \cdot \frac{(U_н - U_{н0})^2}{2!} + \\ + \left(\frac{d^3P}{dU^3}\right) \cdot \frac{(U_н - U_{н0})^3}{3!} + \cdots. \tag{4.8}$$

Не учитывая нелинейные члены ряда ввиду их малости, получаем:

$$P = P_0 + \left(\frac{dP}{dU}\right) \cdot (U_н - U_{н0}). \tag{4.9}$$

Так как $P - P_0 = \Delta P$, а $U_н - U_{н0} = \Delta U_н$, то линеаризованное уравнение в пределах малых отклонений мощности ΔP и напряжения $\Delta U_н$ запишем в следующем виде:

$$\Delta P = k_н \cdot \Delta U_н, \tag{4.10}$$

где $k_н$ – коэффициент усиления нагревательного элемента.

Чтобы определить $k_р$ возьмем производную от функции (4.7).

$$\frac{dP}{dU} = \frac{d(U_н^2/R)}{dU} = \frac{2U_н}{R}.$$

Подставим в это выражение $U_н = U_{н0}$, получаем:

$$k_н = \left(\frac{dP}{dU}\right)_{U_н = U_{н0}} = \frac{2U_{н0}}{R}. \quad (4.11)$$

При учете выражения (4.12) линеаризованное уравнение мощности нагревательного элемента (4.10) примет вид:

$$\Delta P = \frac{2U_{н0}}{R} \cdot \Delta U_н. \quad (4.12)$$

Зная, что линейные и линеаризованные математические модели в виде алгебраических и дифференциальных уравнений могут быть представлены соответствующим им передаточным функциям [38]. Для нашей системы передаточные функции на основе уравнений (4.6), (4.10) после их преобразования по Лапласу при нулевых начальных условиях принимает вид:

$$W(s) = \frac{\theta_н(s)}{P(s)} = \frac{k_н}{T_н \cdot s + 1}. \quad (4.13)$$

Уравнение (4.13) является передаточной функцией нагревателя по каналу мощности.

Рассчитаем $k_н$ и $T_н$для нашей конкретной задачи по формулам (4.4) и (4.5). Пусть значения параметров будут заданы числами: $\alpha_н = 0{,}85$, $F_н = 0{,}99$ $C_н = 500$ $m_н = 0{,}1$ $P_н = 100$.

$$T_н = \frac{C_н \cdot m_н}{\alpha_н \cdot F_н} = \frac{500 \cdot 0{,}1}{0{,}85 \cdot 0{,}99} = 60,$$

$$k_н = \frac{P_н}{\alpha_н \cdot F_н} = \frac{100}{0{,}85 \cdot 0{,}99} = 120.$$

Подставляем в уравнение (4.13) получившиеся значения и получаем математическую модель нашего нагревательного элемента.

$$W(s) = \frac{120}{60 \cdot s + 1}. \quad (14.14)$$

Переведем наше выражение из непрерывной области к цифровому виду. Построим конечно-разностное уравнение. Воспользуемся программой MATLAB для нахождения. Для того, чтобы записать выражение (14.3) в MATLAB, воспользуемся функцией tf. То есть запись будет выглядеть так: W1=tf(140,[60 1]).

Воспользуемся функцией zoh (zero oder hold), она предоставляет точное решение при условии постоянства входного сигнала между моментами квантования, так же дает задержку на такт [52]. Запись в MATLAB будет такой: Gz=c2d(W1, 0.5,'zoh'). В результате было получено выражение в пространстве состояний, оно приведено в выражении (14.15).

$$G(z) = \frac{1{,}162}{z - 0{,}9917}. \quad (14.15)$$

Запишем конечно-разностное уравнение.

$$y(k+1) = 0{,}9917 \cdot y(k) + 1{,}162 \cdot u(k) + \Delta T. \quad (14.16)$$

С помощью это уравнения моделируется изменение температуры на нагревательном элементе. Используем формулу (14.16) в написании кода программы.

4.3 Среда разработки

Для разработки проекта необходимо выбрать программу. Было рассмотрено несколько возможных вариантов для реализации.

1) Eclipse – является бесплатной интегрированной средой разработки модульных кроссплатформенных приложений. У этой среды разработки есть несколько особенностей:

– кроссплатформенность – работает под операционными системами Windows, Linux, Solaris и Mac OS X;

– можно программировать на многих языках, таких как Java, C и C++, PHP, Perl, Python, Cobol и других;

– имеет большой набор API для создания модулей;

– используется подход RCP (RichClientPlatform). Благодаря этому Eclipse является инструментом для создания практически любого клиентского программного обеспечения[53].

2) CooCoxIDE – представляет собой высокоинтегрированную программную среду, предназначенную для разработки кода. CooCox CoIDE можно назвать одной из самых простых и быстрых в плане установки и освоения программы, которая позволит начинающим пользователям разобраться в ней. При открытии программы необходимо выбрать отладочную плату, с которой будет продолжена работа, необходимо найти точное название ее, чтобы избежать ошибок в программе. Далее можно выбрать, откуда надо будет загружать библиотеки для более простой работы программы. Так же можно загрузить библиотеки из сторонних проектов, для этого необходимо в дереве проекта выбрать место загрузки файла.

Было выбрано программное обеспечение CooCoxIDE. Выберем его из-за простоты освоения программы. Этот программный продукт распространен из-за того, что распространяется свободно. У него есть небольшой недостаток, в его

составе сразу нет компилятора, но это исправимо при установке компилятора gcc. Поэтому программный продукт работает отлично с этим компонентом.

При написании программы вначале необходимо подключить библиотеки, которые будут использоваться в коде программы. К примеру, для использования кнопки в своей программе, необходимо подключить вначале библиотеку #include "stm32_ub_button.h". После того, как будут прописанные все используемые библиотеки, пишутся все переменные, которые будут использоваться, то есть произвести инициализацию переменных. Далее прописываются все функции.

Рассмотрим на конкретном примере разработанной программы. Вначале подключаем стандартные С-библиотеки: stdio.h, stdlib.h, string.h, getopt.h, system.h, image.h. Они необходимы для различных операций стандартного ввода и вывода. Содержат в себе функции, занимающиеся выделением памяти, контролем процесса выполнения программы, преобразованием типов и другие функции. Далее подключаем заголовки конкретных типов проекта: gen_define.h, co_canopen.h, co_401.h, usr_401.h, stm32_ub_button.h, stm32_ub_led.h, stm32f429i_discovery.h, stm32f4xx_gpio.h, stm32f4xx_rcc.h, stm32f4xx_usart.h. Они нужны для реализации конкретных задач технического задания. Разработка с поддержкой стандартов CANopen получается благодаря заголовочному файлу co_canopen.h. Действия с кнопкой возможны благодаря заголовочному файлу stm32_ub_button.h. Так же необходимо подключить ядро M4 CMSIS Core, который сократит время создания нового проекта. CMSIS представляет собой последовательные и простые интерфейсы для ядра, его периферии и операционных систем реального времени.

Далее происходит инициализация переменных, задаются локальные определенные функции и локальные определенные переменные. По требованию ТЗ запишем скорость передачи CAN. Тогда протокол осуществляет передачу данных со скоростью 125 кбит/с, поэтому её надо прописать перед основной функцией. Далее прописывается аппаратный ускоритель для STM32F429 и прерывание системного таймера, и прочие дополнительные настройки.

После того, как будут подключены все библиотеки, заголовочные файлы и прописаны все функции и переменные, которые будут использоваться в коде программы, начинается разработки основного рабочего кода. Необходимо прописать инициализацию настройки кнопок, светодиодов и сообщений для вывода пользователю. Для работы с оборудованием CAN необходимо произвести инициализацию клиента CAN оборудования и дополнительную аппаратную инициализацию.

4.4 Проектирование программного обеспечения

Как было сказано в разделе 4.3 вначале подключаются необходимые библиотеки и заголовочные файлы. Далее прописываются переменные, которые будут использоваться, и функции, необходимые для правильной работы программы. По требованию ТЗ, протокол CAN осуществляет свою работу на скорости 125 кбит/с. Запишем переменную с такой скоростью передачи.

Так же необходимо учесть ошибки в инициализации стека протокола CANopen, в случае ошибки происходит оповещения пользователя о наличии ошибки. Одной из такой ошибок может быть ошибка инициализации библиотеки. Могут произойти ошибки инициализации регистра NMT, регистра SDO записи, чтения или проверки записи, регистра PDO и PDO события, самого стека протокола CAN, ошибка регистрации функций доступа.

В бесконечном цикле должны выполняться основные задачи, то есть по нажатию кнопки будет происходить моделирование по рассчитанной математической модели. Зададим u(k)=5, чтобы каждый шаг температура увеличивалась на 5°C. И будет отображать пользователю изменение температуры, показывая это на дисплее. Причем нагрев будет происходить до определенной температуры, которая рассчитана в математической модели, и она равна 120°C. Оповестим пользователя о достижении этой температуры, написав, что она является максимальной. Алгоритм работы программы представлен в документе 270304.2018.340.03.01 Д23.

На рисунке 4.1 представлены элементы для дальнейшего тестирования программы.

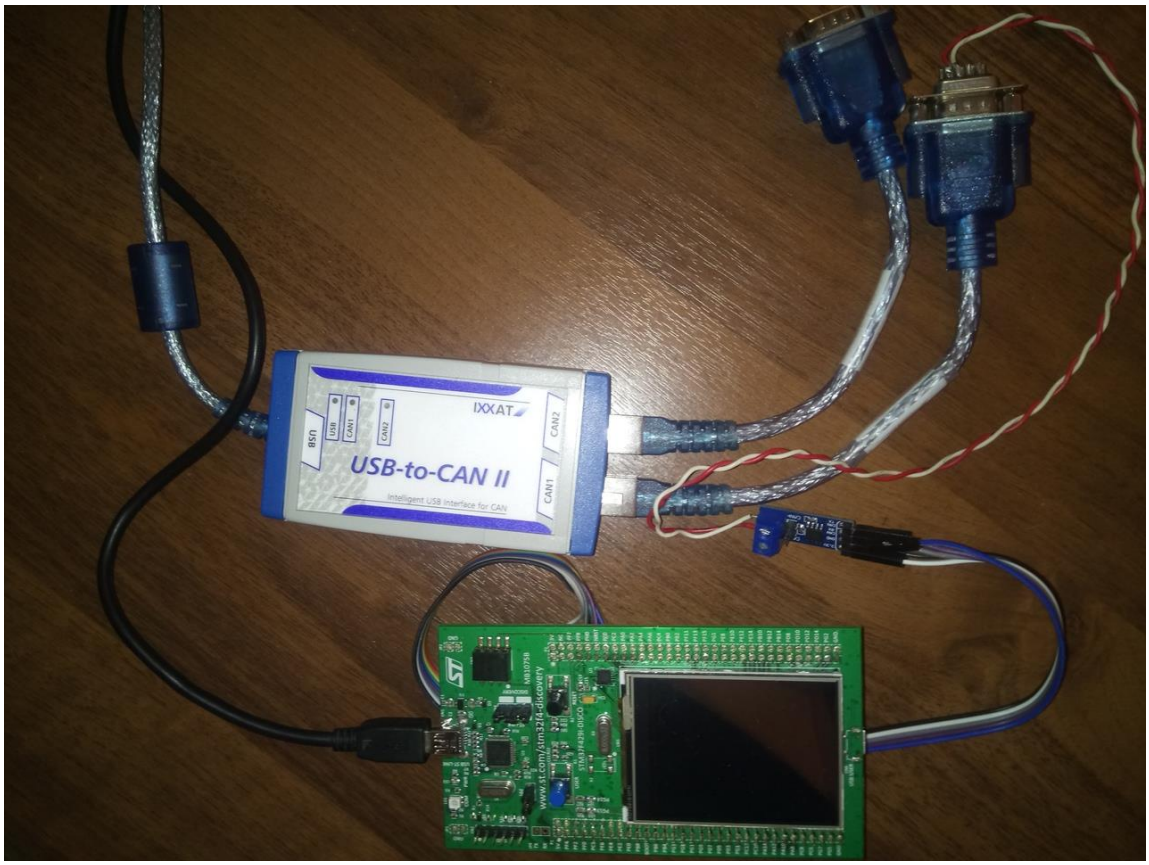

Рисунок 4.1 – Элементы для тестирования программы

Проведем тестирование программы в программе CooCoxIDE, в которой и происходила разработка её. Пронаблюдаем, что будет показывать в отладчике пользователю. На рисунке 4.3 изображен ответ программы на запуск. Пользователю отображается, что протокол CAN начал работу, с какой скоростью работает и в каком состоянии находится шина. В случае, если шина отключена, то это отобразится пользователю. Так же все ошибки в инициализации протокола CAN будут показаны пользователю. Драйвер интерфейса CAN подключен к плате на порты PD0 и PD1, что показано на электрической принципиальной схеме, которая представлена в документе 270304.2018.340.04.01 Э3. Кнопка, с помощью которой производиться управление, подключена на порт PA0, что так же отражена на электрической схеме.

Когда произошла компиляция кода программы, пользователю предоставляется возможность увидеть, что протокол CAN начал свою работу. Так же будет отображена дата отладки программного кода. Если произойдет ошибка в протоколе CAN, пользователю будет показана она. На рисунке 4.2 изображено сообщение пользователю поле загрузки программного кода в STM32F429 в окне отладчика.

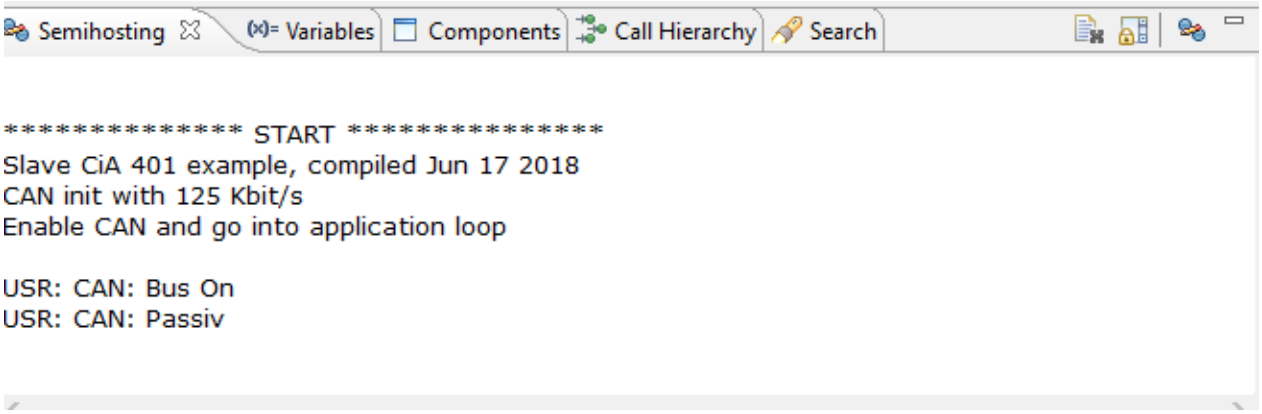

Рисунок 4.2 – Сообщение пользователю после загрузки программного кода

Рисунок 4.3 – Плата STM32F429ZI при загрузке кода

На рисунке 4.3. представлена сама плата STM32F429ZI после загрузки кода в неё. На ней можно наблюдать название проекта и окно, в котором будет

отображаться температура нашего нагревательного элемента. Начальное значение этой температуры равно 25°C и именно это значение отображается на дисплее.

При нажатии на кнопку, температура нагревательного элемента будет изменяться по рассчитанной формуле (14.16), которая была получена в главе 4.1.2. Именно по этому соотношению будет происходить моделирование температуры нашего нагревательного элемента. На рисунке 4.4 представлено отображение в окне отладчика изменение температуры. На этом рисунке видно, что при нажатии на кнопку происходит вывод пользователю температуры, которая сейчас промоделирована на нагревательном элементе. Можно наблюдать, что температуры выводится при изменении на каждые 5°C.

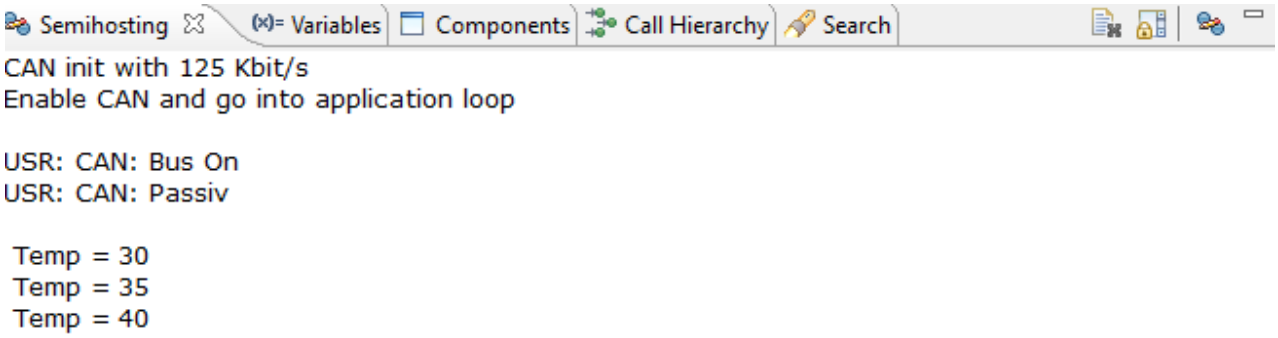

Рисунок 4.4 – Сообщение пользователю при нажатии кнопки

Рисунок 4.5 – Плата STM32F429ZI при нажатии на кнопку

На рисунке 4.5 представлена плата STM32F429ZI после нажатия на пользовательскую кнопку. На плате можно наблюдать название проекта и окно, в котором отображаться температура нашего нагревательного элемента. После нажатия кнопки произошло моделирование температуры, поэтому в окне отображения температуры значение температуры изменилось на другое значение с начального 25°C.

В программном продукте EMTAS CANopen DeviceDesigner посмотрим на нахождение нашей платы. На рисунке 4.6 показана программа EMTAS CANopen DeviceDesigner при подключении устройства. Как видно, наше устройство находится на 32 порту.

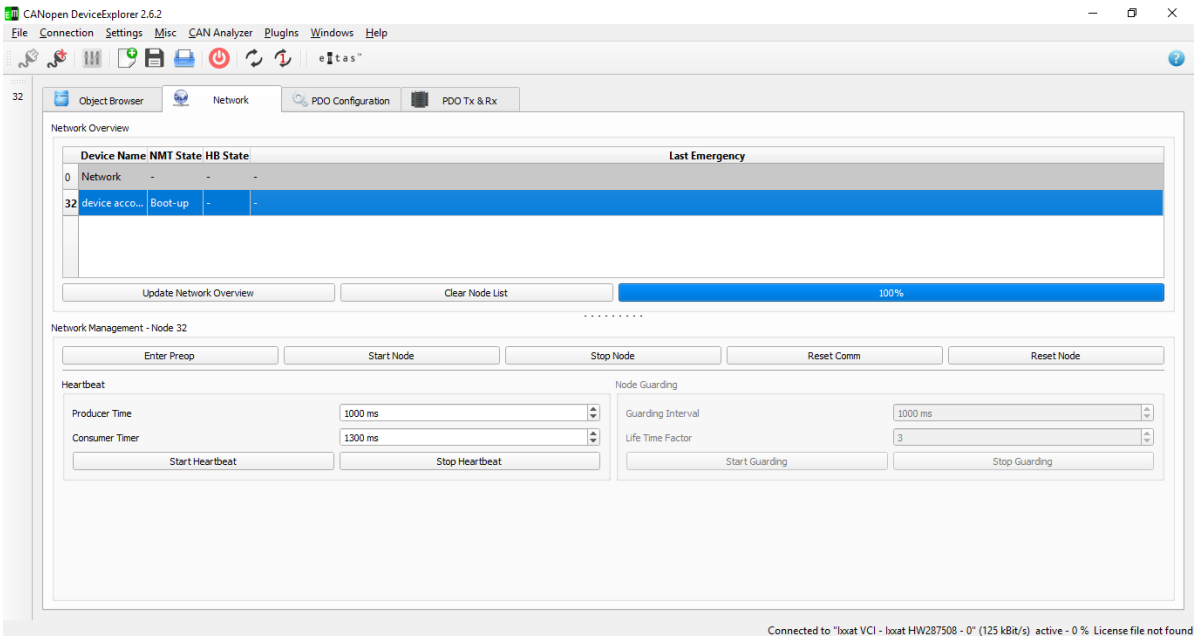

Рисунок 4.6 – EMTAS DeviceDesigner при обнаружении устройства

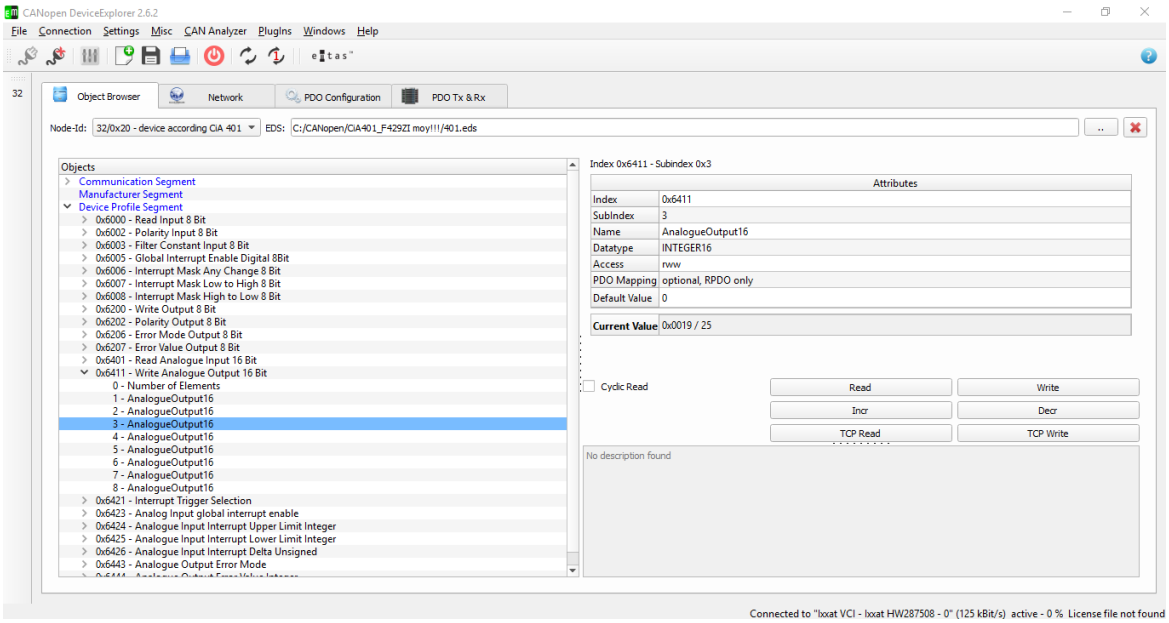

Рисунок 4.7 – Словарь объектов в EMTAS CANopen DeviceDesigner

Для дальнейшей работы необходимо загрузить EDS файл, то есть словарь объектов в наше устройство. После загрузки пользователю предоставляется возможность увидеть словарь объектов. На рисунке 4.7 изображен EDS файл. При просмотре словаря объектов можно анализировать значения, которые находятся в нем, мы можем считывать значения и записывать их.

ЗАКЛЮЧЕНИЕ

В рамках выпускной квалификационной работы был проведен комплексный анализ различный устройств для реализации имитатора датчика температуры, включающий в себя анализ этих элементов, их характеристик. Была разработана структурная схема устройства. На основе требований ТЗ был произведен выбор оптимального оборудования для необходимой работоспособности.

Работа содержит в себе описание алгоритмов работы программы. По алгоритму работы реализован прототип программного обеспечения на отладочной плате.

Результатом проделанной работы является прототип устройства, которое имитирует датчик температуры с интерфейсом CANopen.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Cucej, Z. Industrial networks. Electronic in Marine / Z. Cucej. – London: Springer-Verlag, 2004. – 109 p.

2 Кангин, В.В. Аппаратные и программные средства систем управления. Промышленные сети и контроллеры: учебное пособие / В.В. Кангин. – М.: БИНОМ. Лабораторные занятия, – 2010. – 418 с.

3 Денисенко, В.В. Компьютерное управление технологическим процессом, экспериментом, оборудованием / В.В. Денисенко – М.: Горячая линия – Телеком. – 2009. – 608 с.

4 Таненбаум, Э. Компьютерные сети. 4-е изд. / Э. Таненбаум. – СПб.: Питер, 2003. – 992 с.

5 Кнопка ТС-0104 – https://www.chipdip.ru/product/kls7-ts6601-7.0-180-tc-0104.

6 ГОСТ 19.701-90. ЕСПД. Схемы алгоритмов, программ, данных и систем. – М.: Изд-во стандартов, 1992. – 22 с.

7 ГОСТ 19.003-80. ЕСПД. Схемы алгоритмов и программ. Обозначение условные графические. – М.: Изд-во стандартов, 1981. – 9 с.

8 ГОСТ 2.701-2008. ЕСКД. Схемы. Виды и типы. Общие требования к выполнению. – М.: Изд-во стандартов, 2009. – 16 с.

9 ГОСТ 19.002-80. ЕСПД. Схемы алгоритмов и программ. Правила выполнения. – М.: Изд-во стандартов, 1981. – 9 с.

10 Bosch, R. GmbH: CAN Specification ver. 2.0 / R. Bosch. – London: Springer-Verlag, 1991. – 72 p.

11 Третьяков, С.А. CAN на пороге нового столетия / С.А.Третьяков. // Электроника. – 1999. – № 2. – С. 19-30.

12 Третьяков, С.А. CAN – локальная сеть контроллеров / С.А. Третьяков. // Электроника. – 1998. – №10. – С. 16-20.

13 Микроконтроллер STM32F407VGT6 – https://mcucpu.ru/index.php /ucontrollers/otladochnye-platy/123-otladochnaya-plata-stm32f4-discovery.

14 Блок питания Chieftec Eco 600W (GPE-600S) – http://www.hardwareluxx.ru/index.php/artikel/hardware/netzteile/38502.html.

15 Кнопка MJ-DS430 250V 1.0A OFF-(ON) желтая без фиксации – http://www.vsesamplus.ru/products/knopka-mj-ds430-250v-10a-off-on-zheltaya-bez-fiksatsii.

16 Адаптер USB-CAN/RS485/RS232 «Меркурий 221» – http://roomelectro.ru/801701/.

17 USB-CAN (OBD) – https://www.chipdip.ru/product/usb-can-obd?from=visited_product.

18 Плата для подключения микроконтроллеров к CAN сети, SN65HVD230 CAN Board – https://www.chipdip.ru/product/sn65hvd230-can-board.

19 CAN SPI click MIKROE-986 – https://www.mikroe.com/can-spi-33v-click.

20 LCD дисплей 128x64 синий (ST7920) – http://www.2150692.ru/tovary/display/lcd12864-blue.

21 LCD дисплей 2004 (синий) – http://www.2150692.ru/product/view/15/264.

22 LCD дисплей 2004 (синий) – http://iarduino.ru/shop/Displei/lcd2004-simvolnyy-displey-golubaya-podsvetka-20x4.html.

23 Отладочная плата STM32F429I-Discovery – http://data-ru.ru/shopp/index.php?route=product/product&product_id=7322.

24 CANopenNode – http://microsin.net/programming/arm/canopennode.html.

25 Использование сети CAN и стека CANopen – https://habr.com/post/144435/.

26 Дисплей 2.4" QVGA TFT LCD – http://arduino-kit.ru/catalog/id/displey-2.4-tft-240h320-sensornyiy-%2B-microsd.

27 CanFestival – http://microsin.net/programming/arm/canfestival.html.

28 Отладочная плата STM32F401C-Discovery – https://www.chipdip.ru/product/stm32f401c-disco-2.

29 Отладочная плата STM32F746G-Discovery – https://www.chipdip.ru /product/stm32f769i-disco.

30 Преобразователь USB-to-CAN V2 IXXAT – http://avtprom.ru/ news/2014/02/03/interfeis-ixxat-usb-can.

31 Голенищев, Э.П. Информационное обеспечение систем управления. / Э.П. Голенищев, И.В. Клименко. – Ростов н/Д.: «Феникс», 2003. – 352 с.

32 Денисов, А.А. Теория больших систем управления: Учеб. пособие для вузов. / А.А. Денисов, Д.Н. Колесников. СПб.: Энергоиздат, Ленингр. отд-ние, 1982. – 332 с.

33 Карташов Б.А. Практикум по автоматике. Математическое моделирование автоматического регулирования. / Б.А. Карташов. – М.: КолосС, 2004. – 184 с.

34 Бородин И.Ф. Автоматизация технологических процессов. / И.Ф. Бородин, А.А. Рысс. – М.: КолосС, 1996. – 352 с.

35 Бородин И.Ф. Автоматизация технологических процессов. / И.Ф. Бородин, Ю.А. Судник. – М.: КолосС, 2003. – 343 с.

36 Олифер В.Г. Руководство по технологии объединения сетей. / В.Г. Олифер, Н.А. Олифер; пер с англ. Б.И. Копылова. – М.: Издательский дом «Вильямс», 2002. – 1040 с.

37 Грибанов М.В. CAL-CAN протокол прикладного уровня для индустрии приложений / М.В. Грибанов, Д.П. Калачев, С.А. Третьяков // МКА. – 1999. №4. – С. 70-79.

38 Гусев, С. Краткий экскурс в историю промышленных сетей / С. Гусев. // СТА. – 2000. №4. –С. 78-84.

39 Карпенко Е. Возможности CAN-протокола / Е. Карпенко. // СТА. – 1998. №4. – С. 16-20.

40 Кругляк К. Промышленные сети: цели и средства / К. Кругляк //СТА. –2002. №4. – С. 6-17.

41 Щербаков А. Протоколы прикладного уровня CAN-сетей / А. Щербаков. //СТА. –1999. №3. –С. 6-15.

42 Олифер В.Г. Основы сетей передачи данных. / В.Г. Олифер, Н.А. Олифер. – М.: ИНТУ-ИТ.РУ, 2003 – 248 с.

43 Бесекерский, В.А. Теория систем автоматического регулирования: учеб. пособие / В.А. Бесекерский, Е.П. Попов. – М.: Наука, 1975.– 768 с.

44 Дейт, К. Введение в системы баз данных / К. Дейт. – М.: Наука, 1980. – 60 с.

45 Can (controller area networks) протокол – http://shkolnie.ru/informatika/74515/index.html.

46 Франка П. С++: учебный курс. / П. Франка. – СПб.: Питер, 2003. – 521 с.

47 Шпак Ю.А. Программирование на языке С для AVR и PIC микроконтроллеров./ Ю.А. Шпак – К.: «МК-Пресс», 2006. – 400 с.

48 Бесекерский, В.А. Теория систем автоматического управления: учеб. пособие / В.А. Бесекерский. – СПб.: Профессия, 2007.– 752 с.

49 ГОСТ 2.105-95 ЕСКД. Общие требования к текстовым документам– М.: Стандартинформ, 2011.– 31 с.

50 Z преобразование – http://matlab.exponenta.ru/forum/matlab-matlab-f1/preobrazovanie-t4121.html.

51 Казарин, С.А. Среда разработки Java-приложений Eclipse: учебное пособие / С.А. Казарин. – Москва, 2008. – 77 с.

ПРИЛОЖЕНИЕ А

Схема запуска NMT-узла

ПРИЛОЖЕНИЕ Б

Схемы и алгоритмы.