Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«МЭИ»

Институт Радиотехники и электроники им. В.А. Котельникова

Кафедра Электроники и наноэлектроники

КУРСОВАЯ РАБОТА

по дисциплине «Схемотехника»

ТЕМА: Разработка устройства с передачей данных по протоколу LIN

|  |  |
| --- | --- |
| Студент гр. ЭР-07-22 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Сорокин М.С. |
| (подпись) |  |
| Кандидат технических наук, доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Воронцов В.А. |
| (подпись) |  |

Москва 2025

РЕФЕРАТ

Отчет 34 с., 30 рис., 3 табл., 4 источника.

Перечень ключевых слов: протокол, шина LIN, master, slave, break, sync, PID, D-триггер, J-K-триггер, тактовый генератор, управляемый буфер, мультиплексор, счетчик, компаратор, датчик дождя.

Объектом исследования в данной работе является электрическая цепь, состоящая из базовых логических элементов и, или, не и других элементов, состоящих из них, таких как: D-триггер, J-K-триггер, управляемый буфер, мультиплексор, счетчик, компаратор. Одним из нестандартных элементов является генератор случайных чисел, который заменяет значения с датчика дождя. Хоть это и относится к ЭВМ, но данное устройство не помогает в решении задачи, а используется исключительно для полной демонстрации работы шины LIN. Цель данной работы заключается в том, чтобы показать, как в машине приблизительно реализована передача информации для систем, не требующих высокой надежности, сложных сигналов и высокой скорости передачи. В данной работе мы рассмотрим это на примере одного устройства, как датчик дождя.

Оглавление

[РЕФЕРАТ 2](#_Toc198641930)

[ОПРЕДЕЛЕНИЯ 4](#_Toc198641931)

[ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ 6](#_Toc198641932)

[ЗАДАЧА 8](#_Toc198641933)

[ВВЕДЕНИЕ 9](#_Toc198641934)

[1 Необходимая информация для реализации протокола LIN 11](#_Toc198641935)

[1.1 Что такое шина LIN? Для чего и где она применяется? 11](#_Toc198641936)

[1.2 Устройство шины и протокола LIN 11](#_Toc198641937)

[1.3 Устройство для реализации протокола 13](#_Toc198641938)

[2 Реализация передачи данных от Master 13](#_Toc198641939)

[2.1 Master. Формирование поля Break 14](#_Toc198641940)

[2.2 Master. Формирование поля Sync 15](#_Toc198641941)

[2.3 Master. Формирование поля PID 15](#_Toc198641942)

[2.4 Slave. Считывание полей Break, Sync, PID 16](#_Toc198641943)

[2.5 Передача информации по шине 18](#_Toc198641944)

[3. Моделирование схемы 18](#_Toc198641945)

[3.1 Устройства, применяемые в работе. 18](#_Toc198641946)

[3.1.1 RS-триггер. 18](#_Toc198641947)

[3.1.2 D-триггер. 20](#_Toc198641948)

[3.1.3 Компаратор 20](#_Toc198641949)

[3.1.3 Счетчик 22](#_Toc198641950)

[3.1.5 Сдвиговый регистр с параллельным выводом 23](#_Toc198641951)

[3.1.6 Сдвиговый регистр с последовательным выводом 23](#_Toc198641952)

[3.2 Устройство Master. 24](#_Toc198641953)

[3.3 Устройство Slave. 26](#_Toc198641954)

[4. Итоговая работа схемы и хронограммы 28](#_Toc198641955)

[4.1 Общая схема 28](#_Toc198641956)

[4.2 Хронограмма протокола LIN 30](#_Toc198641957)

[ЗАКЛЮЧЕНИЕ 32](#_Toc198641958)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 33](#_Toc198641959)

ОПРЕДЕЛЕНИЯ

*Шина* *LIN* - это недорогой протокол последовательной передачи данных, который эффективно поддерживает удалённое применение в автомобильной сети. Он особенно хорошо подходит для мехатронных узлов в распределённых автомобильных приложениях, но в равной степени подходит и для промышленного применения. Он дополняет существующую сеть CAN, образуя иерархические сети внутри автомобилей.

*Тактовый генератор импульсов* - предназначен для синхронизации различных процессов в цифровых устройствах — ЭВМ, электронных часах, таймерах и других. Он вырабатывает электрические импульсы (обычно прямоугольной формы) заданной частоты, которая часто используется как эталонная — считая количество импульсов, можно, например, измерять временные интервалы.

***JK-триггер***— это один из стандартных типов триггеров в цифровой электронике, который используется для хранения одного бита информации. Он является более универсальным по сравнению с другими типами триггеров, такими как D-триггер и T-триггер. JK-триггер обладает двумя входами: J и K, и одним выходом, который может принимать два состояния: 0 и 1

*RS-триггер* — это устройство, которое в отсутствии входных сигналов может бесконечно долго находиться в одном из двух устойчивых состояний.

*D-триггер* — это триггер, который запоминает значение входного сигнала D при тактовом импульсе. Он имеет один вход D, на который подаётся сигнал, и тактовый вход C. При приходе тактового импульса Q принимает значение D.

*Компаратор* — устройство, сравнивающее величины аналоговых сигналов. Для аналоговых сигналов компаратор является логическим элементом с релейным режимом

*Счётчик числа импульсов —* устройство, на выходах которого получается двоичный или двоично-десятичный код, соответствующий числу поступивших импульсов. Счётчики могут строиться на двухступенчатых D-триггерах, T-триггерах и JK-триггерах.

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

*LIN -* Local Interconnect Network. Локальная межсетевая сеть.

*Frame Header* - заголовок кадра. То из чего состоит пакет с данными.

*Master –* ведущий. блок управления в шине LIN, который отправляет в шину заголовки пакетов. В зависимости от определенного бита заголовка подчиненные устройства понимают, что им требуется сделать

*Slave –* это принимающее устройство, которое получает команды от блока LIN Master и может только отвечать на них.

*Break –* Первая часть заголовка. это поле представляет из себя 13 нулевых битов подряд.

*Sync –* поле синхронизации. Этот байт имеет определенное значение - 0x55. Именно это число выбрано по той причине, что в двоичном виде оно представляет из себя чередующиеся нули и единицы - 0b01010101. При помощи этого поля устройства могут настроить свою скорость передачи данных.

*PID –* поле идентификатора. В нем зашифровано ID Slave устройства, состоящее из 6 битов, а последние два это биты четности.

Frame Data - Поле данных. Оно состоит из непосредственно байт данных (от 1-го до 8-ми байт) и контрольной суммы (1 байт)

RG – Регистр

T - Триггер

MB – Manage Box

CT2 – Счетчик двоичного кода

SW – переключатель “crcORout”

& – “И”

En – Enable

Clc – Clock

S – Set

R – Reset

D – Data

Res – Reset

RE – ReadEnable

Rec – Record

CRCs – CRCstart

TGm – TaktGen memory

CR – Check Record

Inp – Input

Out – Output

ЗАДАЧА

При выполнении курсовой работы надо:

1. Изучить предоставленную информацию от руководителя и разобраться с устройством и схемой работы шины LIN. Затем выбрать устройство, которое используется в машине, для реализации протокола LIN.
2. Определить то, как будет устроен протокол. Какие действия будет выполнять Master и из чего он будет состоять? Как и в каких ситуациях будет работать устройство Slave? Как будет происходить передача информации по одному проводу в обе стороны?
3. Смоделировать схему в предназначенной для этого программе с использованием элементов, которые были изучены на предмете текущего семестра по дисциплине «цифровая схемотехника». Показать работу созданных блоков.
4. Привести примеры работы полученного протокола в виде хронограммы. С помощью рисунков показать полную оформленную схему по ГОСТу.

ВВЕДЕНИЕ

В данной курсовой работе будет реализована шина LIN и ее протокол передачи данных. Для начала нужно понять, что это такое, для чего, почему и где это используют?

Основная задача шины LIN заключается в том, чтобы передавать сигналы между двумя и большим количеством устройств в машине. Есть ведущее устройство так называемое «Master» и ведомое «Slave». Можно догадаться, что ведущее устройство главнее, поэтому оно управляет ведомым, которых может быть несколько в отличие от первого. Передача информации происходит по так называемой шине, которая на самом деле состоит всего лишь из одного провода, по которому могут передаваться сигналы 1 (это значение напряжения питания от автомобиля примерно 12 В) и 0 (когда питания уходит в землю и значение напряжения становится около 0 В). Сначала Master формирует свой запрос, а потом передает это Slave, который должен считать информацию, сверив ее со своим ID, так как ведущий может обращаться другому устройству, но подробнее об этом мы поговорим позже.

Можно задаться вопросом: «Есть же CAN шина. Она что, хуже?». На самом деле нет. Просто это немного разные технологии, которые в определенных условиях становятся лучше. Например, шина LIN используется в не таких важных электрических цепях, так как она менее надежна чем CAN, поэтому первое чаще всего встречается в такой электронике у машины, как: стеклоподъемники, фары, датчики давления в шинах, датчиках дождя и т.п. Еще одним недостатком LIN является скорость передачи данных, которая достаточно низкая в сравнении с ее конкурентом. Но зато есть большой плюс, благодаря которому используется данная технология это всего лишь один провод и дешевизна, которая помогает сэкономить.

Главной целью будет рассказать и показать функционал, устройство и логику работы данного протокола LIN. То, из чего он состоит, как формируется, считывается и передается. С помощью программы Logisim на простейших элементах, как «ИЛИ», «И», «НЕ» реализовать данный функционал отправки, обработки и возврата ответа, используя пройденный курс «Цифровой схемотехники».

Заключительным этапом работы станет считывание и проверка данных устройством Slave, и отправка полученных результатов с датчика дождя по шине LIN обратно. Значения датчика дождя будут браться от генератора случайных чисел, так как его реализация не входила в планы работы.

1 Необходимая информация для реализации протокола LIN

* 1. Что такое шина LIN? Для чего и где она применяется?

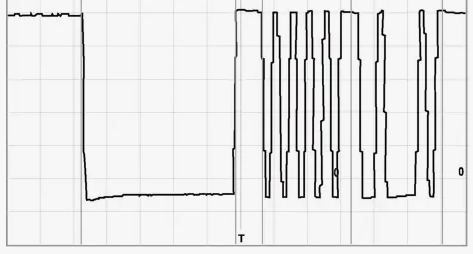
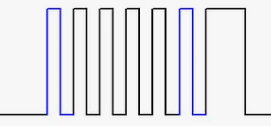
Это простой и недорогой последовательный протокол связи, разработанный для автомобильной электроники. Она используется там, где не требуется высокая скорость передачи данных, но важна низкая стоимость и простота реализации. LIN работает по принципу "ведущий-ведомый" (Master-Slave), где ведущее устройство управляет обменом данными, а ведомые отвечают только по запросу. Основное применение LIN — это связь между датчиками, исполнительными механизмами и управляющими блоками в автомобилях, например, для управления стеклоподъемниками, дверными замками, стеклоочистителями, зеркалами, подогревом сидений и другими второстепенными системами.

Главные плюсы LIN — это дешевизна (однопроводная шина вместо двухпроводной CAN), простота реализации и низкое энергопотребление. Она использует асинхронную передачу данных на скоростях от 2.4 до 20 кбит/с, что достаточно для многих простых задач. Однако у LIN есть и минусы: низкая скорость по сравнению с CAN или FlexRay, отсутствие встроенной защиты от помех (нет дифференциальной передачи, как в CAN) и ограниченная длина шины (обычно до 40 метров).

Несмотря на ограничения, LIN остается популярным решением для бюджетных и маломощных устройств в автомобилях, а также в некоторых промышленных и бытовых системах, где важна простота и низкая стоимость.

1.2 Устройство шины и протокола LIN

Шина LIN представляет собой однопроводную линию связи, которая использует напряжение бортовой сети автомобиля (обычно 12 В) для передачи данных. Физический уровень LIN основан на принципе открытого стока, когда линия не активна, она подтягивается резистором к напряжению питания (логическая "1"), а когда передается сигнал, ведущее устройство притягивает линию к земле (логический "0"). Это означает, что "1" — это высокий уровень (близкий к +12 В), а "0" — низкий (около 0 В). Как это выглядит на реальной осциллограмме показано на рисунке 1а, но мы будем моделировать нашу схему в программе Logisim, поэтому это будет выглядеть, как показано на рисунке 1б.

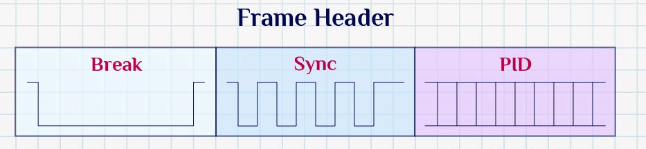
 

*Рисунок. 1а. Реалльная осцилограмма Рисунок.1б. Осцилограмма без учета помех*

На рисунке а можно заметить помехт, о которым мы поговорим немного позже. А идеальный сигнал, который мы и будем рассматривать в данной работе выглядит, как на рисунке 1б.

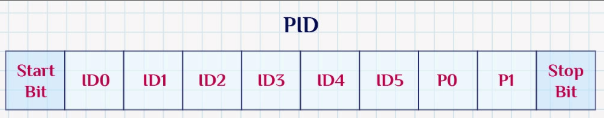
Протокол LIN работает по принципу ведущий-ведомый (Master-Slave), где ведущий узел (Master) инициирует все обмены данными, а ведомые (Slave) отвечают только на его запросы. Кадр данных в LIN состоит из трех частей: заголовка (Header), ответа (Response) и контрольной суммы (Checksum). Заголовок формируется ведущим устройством и содержит синхронизирующий байт, идентификатор сообщения (ID) и контрольный байт. В ответе ведомое устройство передает данные (1–8 байт), а контрольная сумма обеспечивает проверку целостности информации.

Заголовок пакета состоит из нескольких частей, изображенных на рисунке 2:



*Рисунок. 2. Заголовок пакета*

* **Поле Break** - это поле представляет из себя 13 нулевых битов подряд.
* **Поле Sync** - поле синхронизации. Этот байт имеет определенное значение - 0x55. Именно это число выбрано по той причине, что в двоичном виде оно представляет из себя чередующиеся нули и единицы - 0b01010101. При помощи этого поля устройства могут настроить свою скорость передачи данных.
* **Поле PID** - поле идентификатора. В нем зашифровано следующее:



*Рисунок 3. Поле PID*

Старт и стоп-биты здесь играют ту же роль, что и при передаче данных по UART, и используются для каждого из передаваемых по LIN байт.

Из битов ID0...ID5 складывается непосредственно значение идентификатора. А поскольку под это выделено только 6 битов, то значит диапазон значений идентификатора составляет от 0 до 0x3F (0b111111). При этом значения от 0x3C до 0x3F являются служебными.

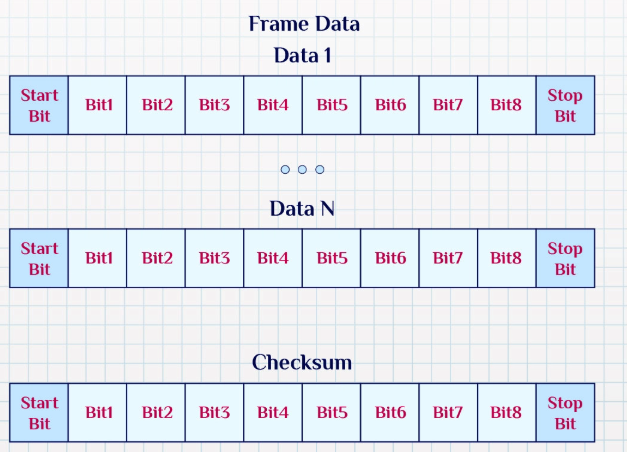
Итак, у нас остаются еще два бита четности, для них формула выглядит следующим образом:

P0=ID0⊕ID1⊕ID2⊕ID4 (1)

P1=!(ID1⊕ID3⊕ID4⊕ID5) (2)

И на этом все, заголовок пакета сформирован.

Поле данных в свою очередь состоит из непосредственно байт данных (от 1-го до 8-ми байт) и контрольной суммы (1 байт):



*Рисунок 4. Поле данных.*

1.3 Устройство для реализации протокола

Как было сказано выше, в машине шина LIN используется в различных второстепенных устройствах, которые не так важны по сравнению с двигателем, различными системами безопасности и т.п. Из всех устройств мое внимание больше всего привлек датчик дождя, так как мне было интересно узнать принцип его работы и то, как при попадании капель дождя на лобовое стекло, начинают работать дворники с различной скоростью.

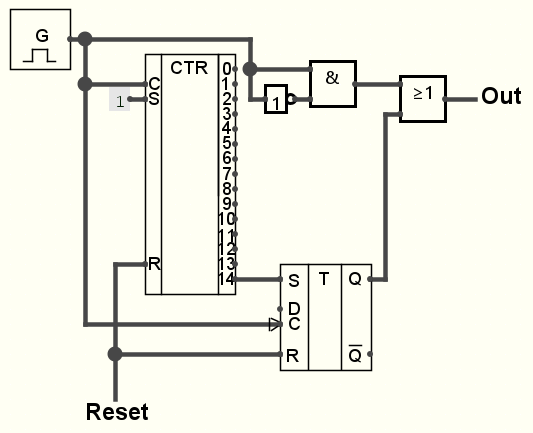
2 Реализация передачи данных от Master

Master (ведущий) — это главный управляющий модуль (например, блок управления кузовом, комбинация приборов или отдельный LIN-контроллер), который создает запросы к датчику и обрабатывает его ответы. Master решает, когда опрашивать датчик – он отправляет заголовок (Header) с идентификатором (ID), который говорит датчику дождя: "Пора передать данные". Master принимает ответ и принимает решение – например, включает дворники или регулирует их скорость.

2.1 Master. Формирование поля Break

Для того чтобы начать считывать сигнал, нужно знать, когда он начинается. Для этого и было придумано поле Break, состоящее всего лишь из одних тринадцати нулей. В различных реализациях шины LIN возможно использование двенадцати, либо четырнадцати нулей. Чаще всего в реализации UART используют именно тринадцать нулей. Такое количество обусловлено тем, что различные поля обычно состоят из 8 бит данных, а с учетом возможных старт и стоп битов это могут быть и 10 битов. Еще возможны различные перебои, помехи, которые могут нарушить передачу и распознавание сигнала, поэтому, чтобы не было путаницы и можно было отличить момент для начала передачи пакетов данных, решили принять 13 нулей подряд за начало передачи данных. Поэтому на данном этапе никаких особенностей нет. Будет просто использовать 13 нулей, как сигнал для устройства Slave о начале подачи информации.

Для реализации формирования данного поля нам понадобится счетчик на 15 значений, так как помимо 13 нулей для Break два нуля будут относиться уже к полю Sync. После счетчика стоит D-триггер, который срабатывает сразу же после счетчика. В это время параллельно через условие создается постоянный нуль при любых значениях тактового генератора, тем самым получается 15 нулей подряд. Вот как это выглядит:

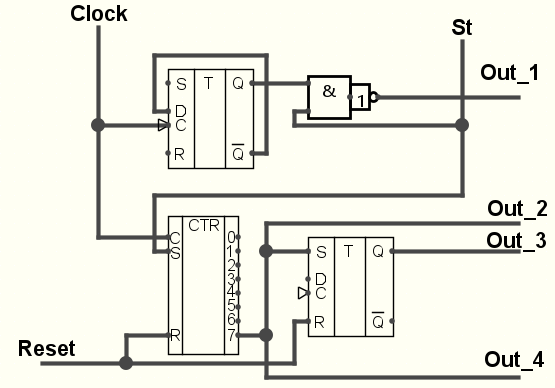


*Рис.5 Схема составления поля Break*

2.2 Master. Формирование поля Sync

Данное часть протокола отвечает за синхронизацию скорости передачи информации между Master и Slave. Оно состоит из 8 чередующихся нулей и единиц и двух дополнительных: старт (0) и стоп (1) биты. Начинается код с нуля, так как в обычном состоянии по шине идет 12 В (это логическая 1), поэтому, чтобы понять, когда начнется поле Sync, нужно притянуть питание к земле (логический 0). Возможен другой вариант, при котором постоянное значение на шине был нуль, и тогда старт бит начинался бы с 1, а стоп бит был 0. В данной работе я буду использовать первый вариант, где на шине постоянной будет единица. Поэтому итоговое поле Sync будет таким: 0010101011.

Чтобы составить данное поле, были использованы такие элементы: D-триггер, как делитель частоты, после которого шел элемент «И» с инверсным выходом, который соединен с D-триггером и выходом D-триггера с Break. Еще используется счетчик на 8, который как раз отсчитывает оставшиеся восемь битов для формирования поля.

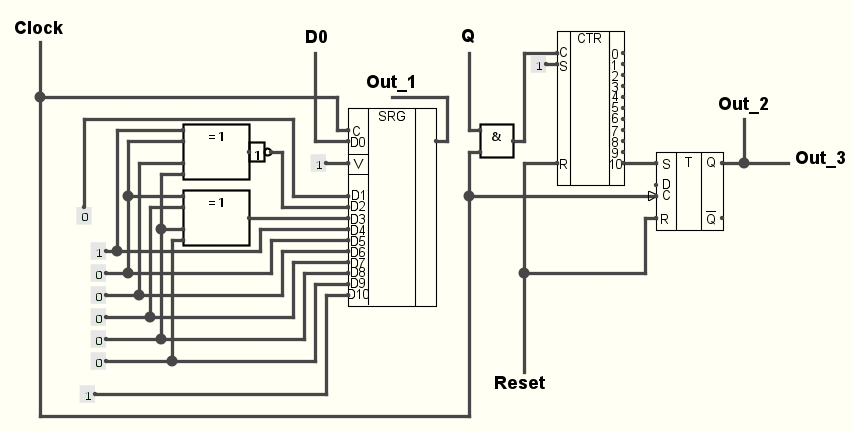


*Рис.6 Формирование поля Sync*

2.3 Master. Формирование поля PID

Состоит оно так же из 8 бит, по краям окруженное старт (0) и стоп (1) битами. Первые шесть бит отвечают за уникальный номер для определенного устройства, к которому захочет обратиться Master. В данной работе у нас будет только один Slave, что облегчит задачу не только в количестве идентификаторов, но и в размере схемы, но об этом поговорим чуть позже. Остается еще два бита, которые называются «биты четности». Они нужны для того, чтобы проверить правильность обращения к устройству Slave. Рассчитывается по формулам, которые были прописаны выше, 1 и 2.

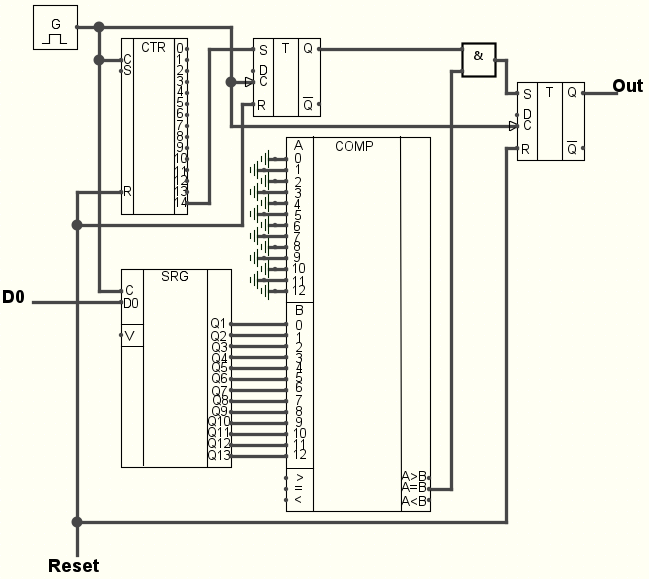
В реализации данной схемы нам помогут: два элемента «исключающее или», сдвиговый регистр с параллельным входом и последовательным выходом на 10 бит, счетчик на 11, условие «и» и триггер.



*Рис.7 Формирование поля PID*

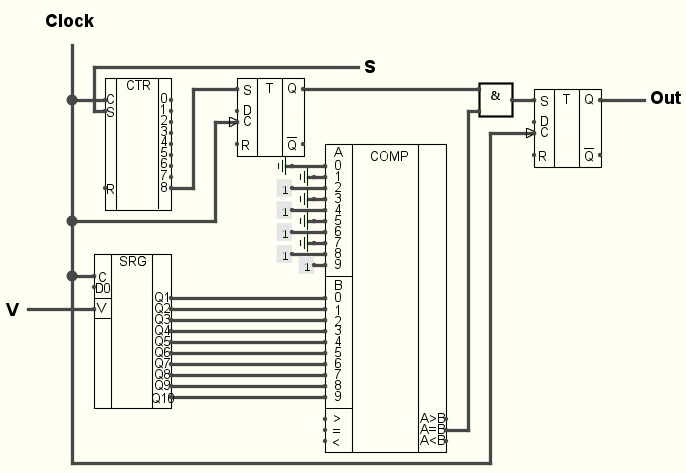
2.4 Slave. Считывание полей Break, Sync, PID

Устройство Master формирует пакет данных и отправляет его устройству Slave, которое в свое же время должно считать эти данные. Я реализовал это с использованием сдвигового регистра на 13 бит с последовательным входом и параллельным выходом соответственно. После регистра стоит компаратор на сравнение тринадцатибитного числа из нулей с таким же числом. Параллельно с этим работает счетчик на 15 значений. При выполнении двух условий одновременно сработает D-триггер и считается сигнал. На рисунке ниже показана данная схема.



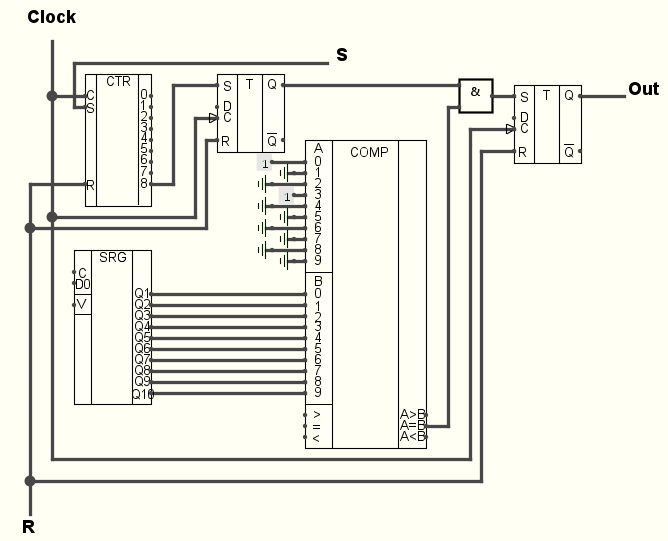
*Рис.8 Считывание поля Break*

Поле Sync работает по аналогичному принципу. Отличие заключается только в ограничениях элементов, где счетчик работает до 9, а сдвиговый регистр считывает 10 значений.



*Рис.9 Считывание поля Sync*

Остается проверить последнее поле - PID. И делается это по тому же принципу, как и оба прошлых поля. Отличий от Sync практически нет, кроме сравнения числа на компараторе. Тут у нас получится число 0x09, то есть в двоичной системе счисления это будет – 0000001001. Вот как это выглядит на схеме:



*Рис.10 Считывание поля PID*

2.5 Передача информации по шине

На первый взгляд может показаться, что тут нет ничего сложного, нужно просто соединить провода и все заработает. На самом деле проблемы возникают с тем, что нельзя просто соединить три провода, так как при разных значениях они будут приводить к ошибке, что теперь и кажется логичным. К счастью, с данной проблемой помог справиться буфер. Это такое устройство, которое передает сигнал только при подаче на его вход единицы, иначе, сигнал попросту не будет проходить. Используется два буфера для работы шины LIN. Первый обеспечивает передачу информации от Master-а, а второй наоборот – передачу от Slave.

3. Моделирование схемы

3.1 Устройства, применяемые в работе

В данной части отчета рассмотрим схемы различных элементов, реализованных на простейших элементах, как «И», «ИЛИ», «НЕ», которые применялись в моей работе. Начнем с триггеров.

3.1.1 RS-триггер

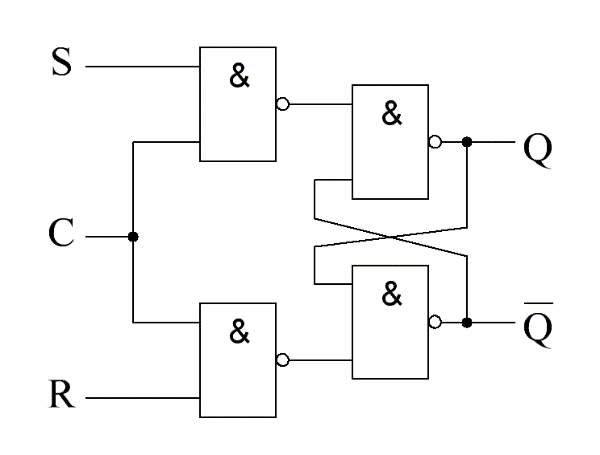


Рисунок 11 – Схема синхронного -триггера

Одной из простейших последовательных схем является ‑триггер (от англ. *Reset* и *Set*), построенного, как показано на рисунке 10, на элементах И-НЕ. У -триггера есть два входа – и и два выхода и [1]. В таблице 1 приведена его таблица истинности, а на рисунке 11 его условно графическое отображение.

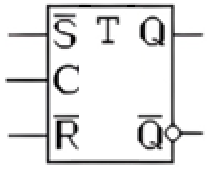


Рисунок 12 – Условно-графическое отображение -триггера

Таблица 1 – Таблица истинности синхронного -триггера

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| 0 | X | X | 0 | 0 |
| 0 | X | X | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | \* |
| 1 | 1 | 1 | 1 | \* |

Теперь смоделируем его и рассмотрим его временные диаграммы (рисунок 12). На полученном графике видно, что в начальный момент времени, когда на оба входа не поступал сигнал, триггер находится в неопределенном состоянии, что приводит к выводу ошибки. После подачи сигнала на вход триггер определяется, и ошибка исчезает.

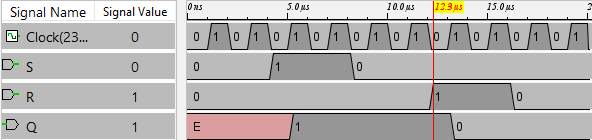


Рисунок 13 – Временная диаграмма-триггера

3.1.2 D-триггер.

-триггером называется триггер с одним информационным входом, работающий так, что сигнал на выходе после переключения равен сигналу на входе до переключения «1».

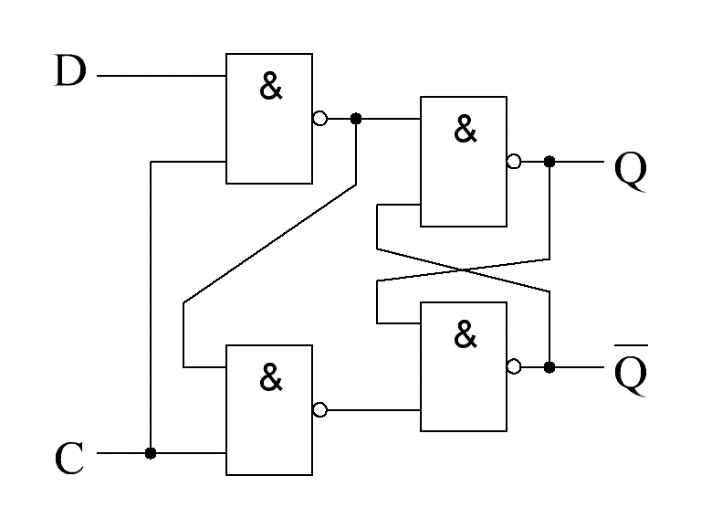


Рисунок 14 – Схема синхронного -триггера

В схемах он отображается как на рисунке 15.

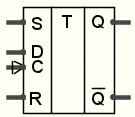


Рисунок 15 – Условно-графическое отображение -триггера

На полученных диаграммах видно, что в начальный момент триггер не определен, но сразу же после первого такта устанавливается в нуль. При подаче на вход сигнала триггер его устанавливает, но после его исчезновения он сразу переходит в состояние устойчивого нуля.

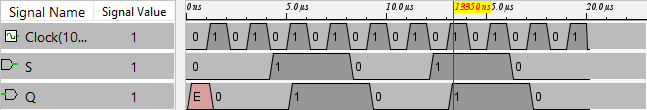


Рисунок 16 – Временная диаграмма-триггера

3.1.3 Компаратор

Компаратор - комбинационная схема, реализующая сравнение двух операндов, представленных двоичными кодами. На рисунке 21 представлена схема компаратора на 8 бит.

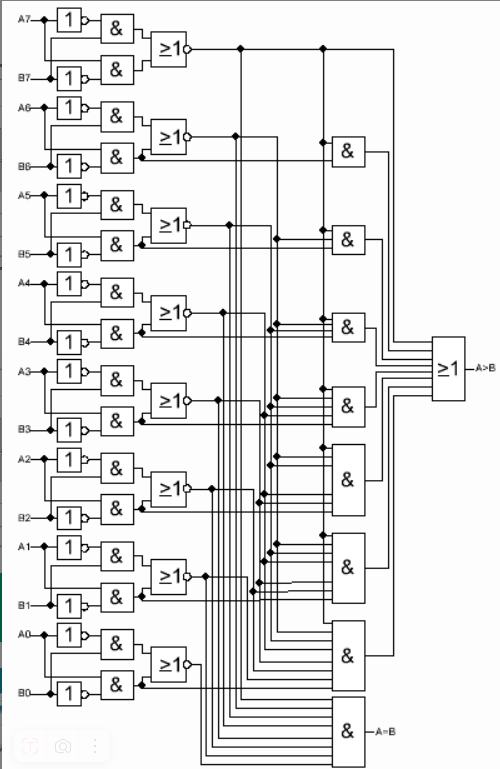


Рисунок 17 – Схема компаратора на 8 бит

В данном блоке сравниваются 2 параллельно пришедших 8-ми битных числа, результатом которого будет либо равенство, либо разница (больше или меньше) значений А и В.

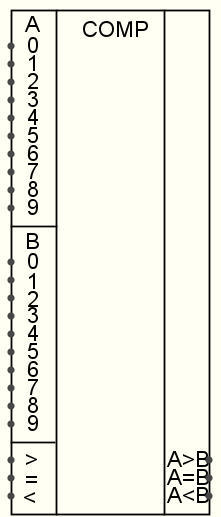


Рисунок 18 – Условно-графическое отображение компаратора на 10 бит

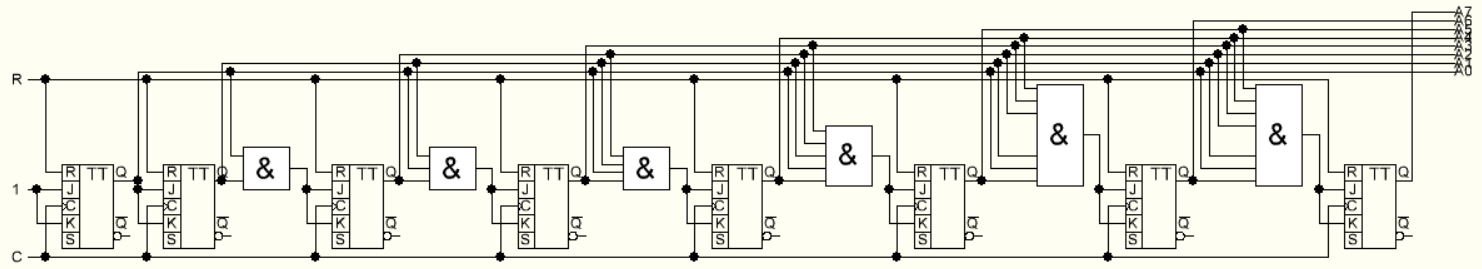
В данной курсовой работе будут использоваться компаратор и 13 бит, но я показал схему на 10, так как она просто влезает в отчет.

3.1.3 Счетчик

Счётчик — это устройство, которое используется в цифровой электронике для подсчёта тактовых импульсов.

В данной работе он реализован достаточно интересно. С помощью JK-триггеров и компаратора на сравнение. При установке различных чисел на компаратор задается определенное число для срабатывания счетчика и изменением количества триггеров, можно менять значения в степени двойки.

Вот как это выглядит на схеме.



*Рис.19. Схема счетчика на 8*

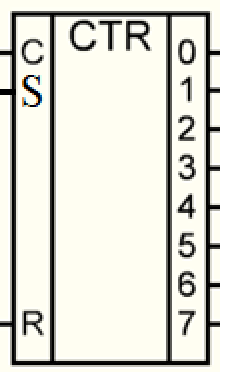
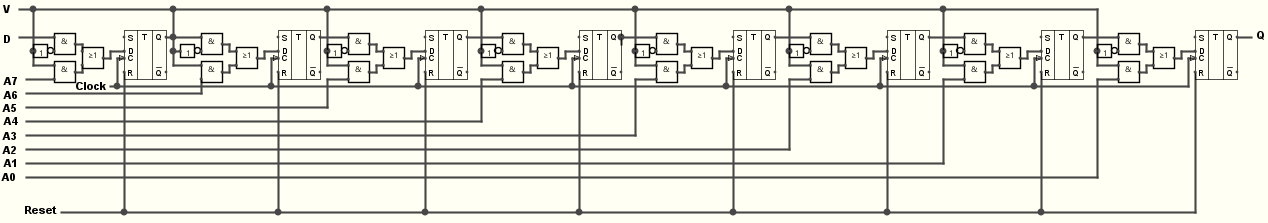


Рис. 20. Условно-графическое отображение счетчика на 8 бит

3.1.5 Сдвиговый регистр с последовательным выводом

Данное устройство реализовано на D-триггерах. Сколько триггеров столько значений можно будет записать параллельно. Вывод происходит по одному биту. Вот как это выглядит на примере SRG8\_out\_par:



*Рис 23. Схема сдвигового регистра с последовательным выводом на 8 бит*

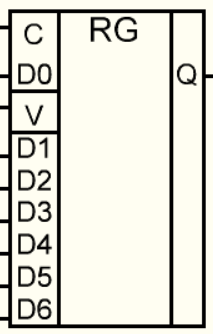


Рисунок 24 – Условно-графическое отображение сдвигового регистра с последовательным выводом на 8 бит

3.2 Устройство Master.

Данную часть лучше всего будет разделить на две – это ведущий и ведомый. Мы начнем с первого, так как это самая главная часть. По прошлому подразделу можно было понять принцип считывания Break, Sync, PID. Теперь я постараюсь немного короче объяснить эту часть и больше затронуть те моменты, которые не были озвучены.

Начнем с начала. Мы считываем 15 подряд идущих нулей с помощью счетчика, который при срабатывании активирует D-триггер и тогда запись прекращается, так как значения записываются через мультиплексор. Единица срабатывает на одном из элементов «или», который идет к А0.

Затем считывается поле Sync по похожей схеме. Только с помощью делителя частоты и условия «и» с инверсией, мы получаем чередование 0 и 1 до срабатывания счетчика с триггером и сигнал на мультиплексоре не становится равен 11, что говорит о считывании следующего поля.

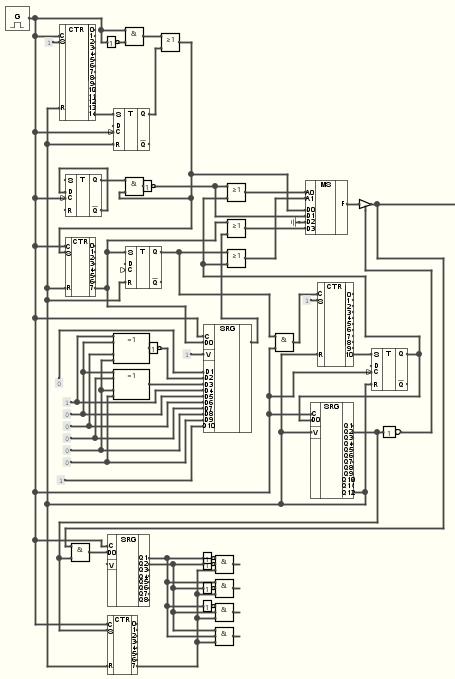
PID формируется с помощью двух элементов «исключающее или», где один из них с инвертором на выходе. Эти данные передаются на параллельный по записи сдвиговый регистр, выдавая по одному значению, пока счетчик не достигнет 11. После чего начнет отрабатывать SRG12, так как в это время будет ожидать передачи информации от Slave, а затем, после срабатывания третьего триггера данные перестанут поступать на шину LIN с помощью буфера, который до этого был включен. После того, как отработает последний триггер, то произойдет сброс всей схемы и все начнется с начала.

Ниже приведена таблица работы мультиплексора при разных значениях, подаваемых на A.

*Таблица 2. Работа мультиплексора при разных сигналах*

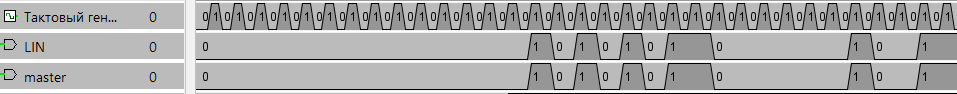
|  |  |  |
| --- | --- | --- |
| A0 | A1 | MUX |
| 0 | 0 | Break |
| 0 | 1 | Sync |
| 1 | 0 | - |
| 1 | 1 | PID |

На рисунке ниже изображена вся схема того, как устроен Master. Тут я еще не рассказал про самую нижнюю часть, которая отвечает за считывание информации со Slave устройства.



*Рис.25 Общая схема устройства Master*

Хронограмма для приведенной выше схемы будет выглядеть таким образом:

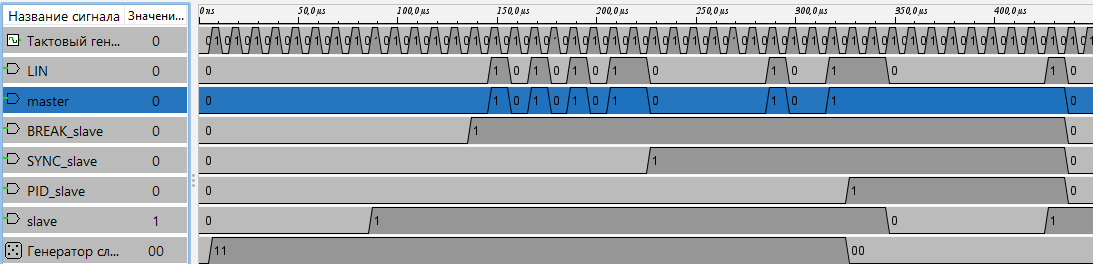


*Рис.26 Хронограмма для устройства Master*

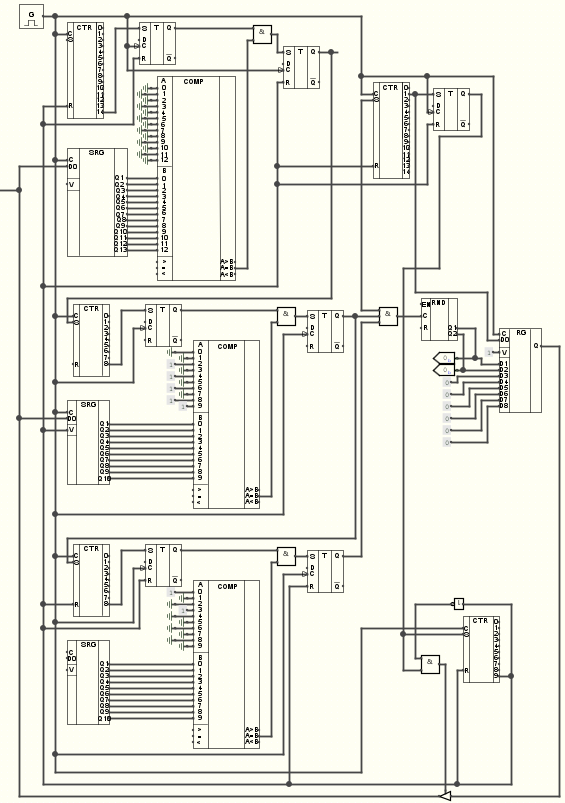
3.3 Устройство Slave.

После формирования сигнала, нужно его считать. Этим занимается ведомое устройство, которое, как мы разбирали раньше – по одному принципу считывает Break, Sync, PID с помощью SRG и CTR, которые работают параллельно друг с другом, а после, с помощью триггера и компаратора проходят условие «И» и включают другой триггер, который означает, что поле считалось. После этого сигнал идет работать уже с другой частью протокола. Когда сформируется последняя часть, то запустится счетчик до 1 из 16. Это сделано для того, чтобы за то время пока будет считываться сигнал с датчика дождя, счетчик не успел обнулиться, и схема не сломалась, а при срабатывании счетчика запустится RG8 в запись. Так же после формирования всех трех полей, выводы с них идут на элемент «И», который и запускает считывание данных с датчика дождя. Это реализуется с помощью генератора случайных чисел, так как мы не знаем когда может пойти дождь и создания датчика дождя не входило в задание. В это время, D-триггер после CTR1 держит 1, которая передает Slave в режим передачи данных с датчика дождя на LIN шину.

На рисунке ниже будет показана хронограмма и то, как это все выглядит на схеме.



*Рис.27 Хронограмма устройства Slave*



*Рис. 28. Схема устройства Slave*

4. Итоговая работа схемы и хронограммы

На данном шаге разберем схему до конца и расскажем то, чего не было на прошлых этапах, чтобы пройти путь от самого начала и до конца. Посмотрим на общий вид схемы и разберем хронограммы, которые она показывает.

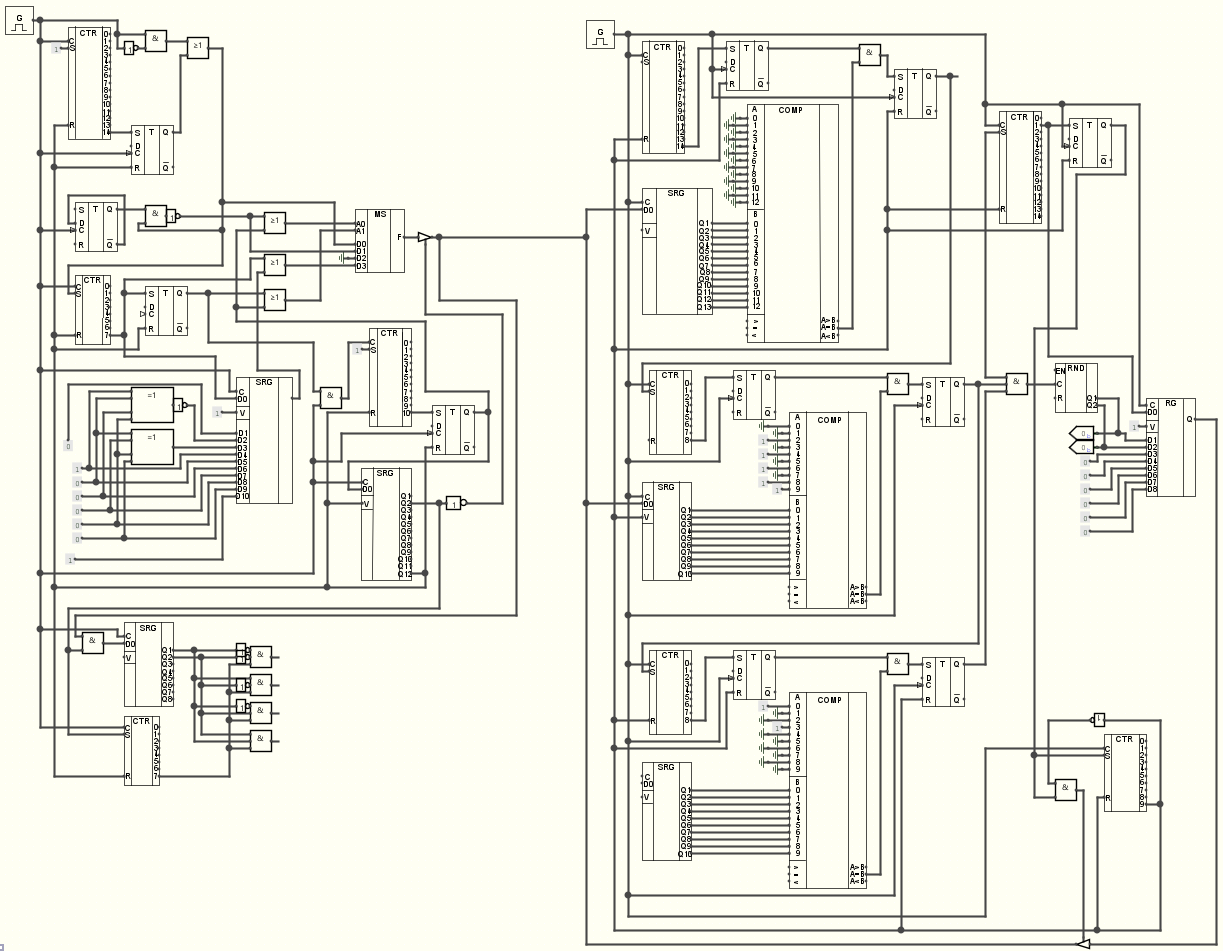
4.1 Общая схема

На прошлых этапах мы разобрали полностью составленный протокол для передачи данных и затем считали его ведомым устройством. Еще было рассказано, как происходит считывание данных с датчика дождя. Это реализовано с помощью генератора случайных чисел, который может выдать значения в два бита, которые показаны в таблице ниже.

Таблица 3. Сила дождя в зависимости от значений датчика

|  |  |
| --- | --- |
| Значения датчика дождя | Сила дождя |
| 00 | Дождя нет. Дворники не работают |
| 01 | Слабый дождь. Режим дворников 1 |
| 10 | Сильный дождь. Режим дворников 2 |
| 11 | Ливень. Режим дворников 3 |

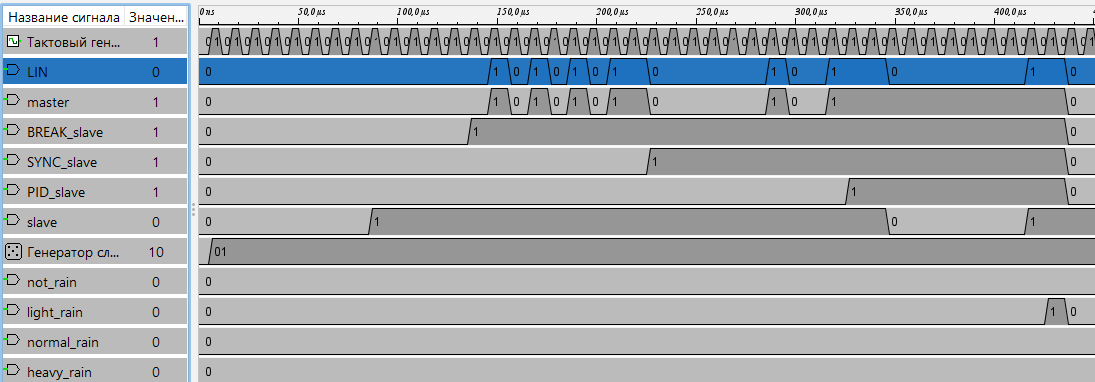
Slave, считав данные с датчика дождя, должен передать их по шине LIN Master-у, который управляет самими дворниками. Действие передачи реализуется так же с помощью сдвигового регистра, счетчика и буфера, который разрешает передать сигнал на ведущее устройство. Оно уже, так сказать, распаковывает данные с помощью того же регистра только на последовательный прием и параллельный вывод, где с помощью элементов «И» определяется число из представленной выше таблицы. В дальнейшем эта информация передается на сами дворники, а у нас это выводится на хронограмме. На этом объяснение всей схемы заканчивается. Теперь можно посмотреть на общий вид и на то, как это все работает.



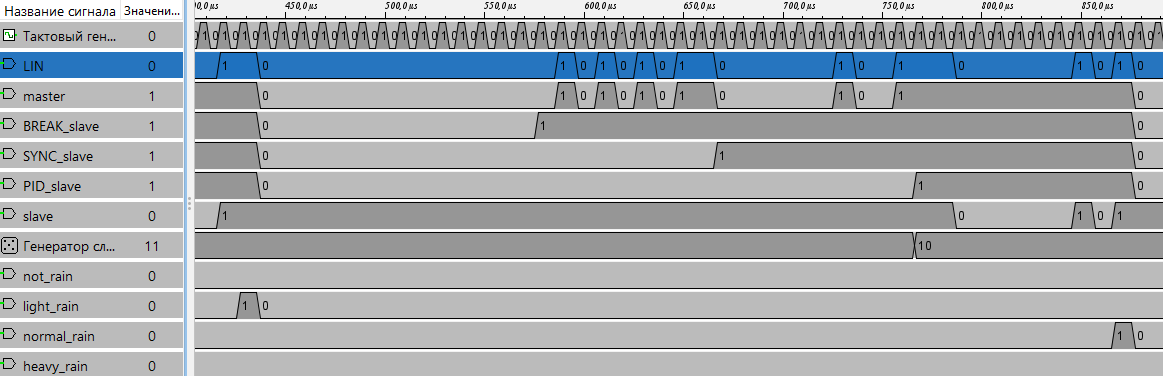
*Рис. 29. Общий вид схемы*

4.2 Хронограмма протокола LIN

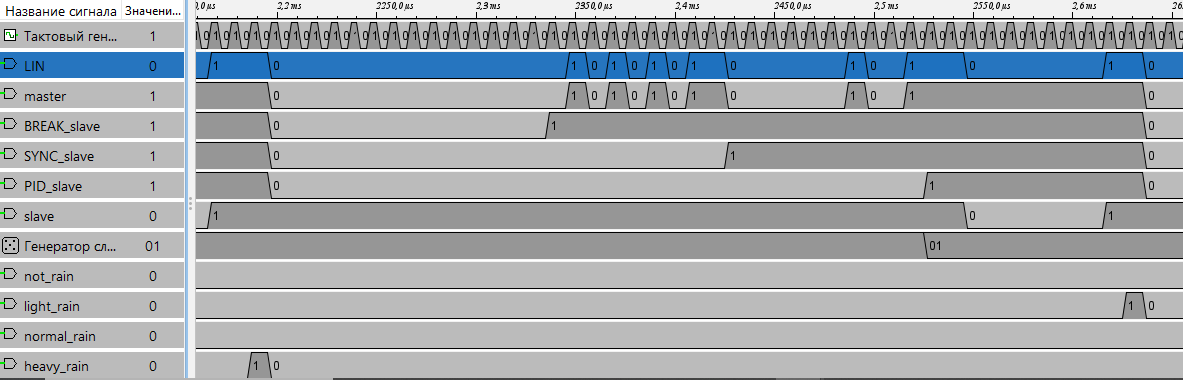
С помощью кнопки «моделировать» в программе Logisim, выбираем «такты включены», устанавливаем нужную нам частоту работы тактовых генераторов и включаем моделирование. Появляется картинка построения сигнала на различных контактах схемы, которые мы указали. Это помогает быстро и просто определять то, как работает схема.



*Рис. 30а. Хронограмма реализации протокола LIN*



*Рис. 30б. Хронограмма реализации протокола LIN*



*Рис 30в. Хронограмма реализации протокола LIN*

Начнем с первого рисунка. Это начало работы протокола. Сразу же обращаем внимание на строчку с шиной LIN и Master, которые по началу схожи. У них идет последовательность из 15 нулей, так как 13 первых это Break, а 2 оставшихся – Sync. Затем идут оставшиеся 8 бит данных поля и начинается формирование последнего поля PID, которое так же правильно записывается. После чего, ведущий ожидает передачи информации с датчика дождя и все сбрасывается. Дальше идут строчки, которые означают было принято ли поле или нет. Если стоит 1, то соответствующий сигнал считан, иначе – нет. Следующая строчка показывает значения Slave, который при 1 считывает информацию либо находится в спящем режиме. Когда Slave поймет по полю PID, что обращение идет к нему и получит весь сигнал, то ему нужно будет передать информацию от датчика дождя к Master, что он и делает. В зависимости сгенерированных значений может получиться разный сигнал, но первые 6 бит всегда будут нулями. Это можно заметить на хронограмме в строчках slave и LIN. Следующая строка, это датчик дождя (генератор случайных чисел), который каждый раз в определенный промежуток времени высылает информацию по дождю, которая передается на дворники. Это можно заметить по оставшимся ниже строчкам с различными уровнями дождя.

Остальные два рисунка показывают цикличность исправной работы протокола. Возможно изменение только значений с датчика дождя, так как мог начаться либо закончиться дождь через некоторое время.

ЗАКЛЮЧЕНИЕ

В данном курсовом задании был реализован протокол шины LIN для датчика дождя в программе Logisim. Основная цель была рассказать и показать функционал, устройство и логику работы данной протокола LIN. То, из чего он состоит, как формируется, считывается и передается. С помощью простейших элементов, как «И», «ИЛИ», «НЕ», были созданы другие более сложные: D-триггер, компаратор, счетчики на различные значения, сдвиговые регистры с последовательным и параллельным выходами, так же на разное количество бит. Данные элементы были использованы при создании протокола Master-ом всех полей, как Break, Sync, PID и их считывания с помощью устройства Slave, которое затем передавало данные обратно ведущему устройству для сообщения их дворникам. Схема была разработана с возможностью дальнейшего улучшения, например, добавления еще некоторых устройств Slave (стеклоподъемники, фары и т.п.), что в целом и подразумевалось для машины, чтобы сделать ее менее трудозатратной и компактной.

В результате все поставленные задачи выполнены. Схема была разработана с возможностью дальнейшего улучшения, например, добавления еще некоторых устройств Slave (стеклоподъемники, фары и т.п.), что в целом и подразумевалось для данного. Помимо этого, можно было добавить различные обработки ошибок, чтобы система была полностью автономна и могла справляться с экстренными при неправильной передаче полей или битов четности из-за различных помех.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 7.32-2017 «Структура и правила оформления». Структура пояснительных записок к отчетам по практике и выпускным квалификационным работам (ВКР) и требования к их оформлению.
2. Дэвид М. Харрис, Сара Л. Харрис Цифровая схемотехника и архитектура компьютера. / пер. с англ. Imagination Technologies. – М.: ДМКПресс
3. Единая система конструкторской документации. Обозначения условные, графические в схемах ГОСТ 2.743-91. В. В. Долгополов, канд. техн. наук; В. Ю. Гуленков, канд. техн. наук; С. С. Борушек, Л. Г. Юрганова, В. В. Гугнина URL: <https://ohranatruda.ru/upload/iblock/d1c/4294849507.pdf> (дата обращения 16.05.2025)
4. Протокол LIN. Полный обзор, описание, формат кадра. 04.08.2020 -URL: <https://microtechnics.ru/protokol-lin-polnyj-obzor-opisanie-format-kadra/> (дата обращения 12.03.2025)