ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

**ЛАБОРАТОРНАЯ РАБОТА №4**

по дисциплине «Сети ЭВМ»

«Адресация IP версии 4. Статическая маршрутизация»

|  |  |
| --- | --- |
| Выполнил студент | Некто Н.Н. |
|  | Ф.И.О. |

|  |  |
| --- | --- |
| Группы | И\*-\*4\* |
|  |  |

|  |  |  |
| --- | --- | --- |
| Работу принял |  | доцент каф. ВС Перышкова Е.Н. |
|  | подпись |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Защищена |  | Оценка |  |
|  |  |  |  |

Новосибирск – 2020

**СОДЕРЖАНИЕ**

1. Задание на лабораторную работу.......................................................................3

2. Порядок выполнения работы.............................................................................5

3. Ответы на контрольные вопросы.....................................................................11

ПРИЛОЖЕНИЕ A. Исходный код задания №8..................................................23

# **ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ**

1. В сети, созданной Вами в лабораторных работах 1 и 2, измените конфигурацию канала, соединяющего маршрутизаторы офисов так, чтобы:

- Передача данных осуществлялась с применением алгоритма *PPP*;

- Доступ к каналу должен быть авторизованным с использованием алгоритма *CHAP*;

- Скорость передачи по каналу должна быть не более 128000 бит в секунду.

2. Разделите сеть Главного офиса на две виртуальные сети, объединив устройства так, как показано на рисунке 2.1.

3. Измените настройки сетевого оборудования так, чтобы в рамках выделенного диапазона адресов для сети Главного офиса были сформированы две логические подсети.

4. Сконфигурируйте маршрутизатор Главного офиса так, чтобы он обеспечивал связь между локальными сетями офиса.

5. Настройте маршрутизатор главного офиса так, чтобы появилась возможность передавать данные от серверов через их интерфейсы *FastEthernet* 0/1 (которые подключены к коммутатору, интегрированному в маршрутизатор). Эта сеть должна использовать протокол *IEEE 802.1Q*. В качестве номеров *VLAN* также должны использоваться 30 и 40.

6. Настройке локальную сеть дополнительного офиса так, чтобы в ней данные передавались кадрами размером 1290 октетов.

7. Объясните:

1) Почему после изменения сети в Главном офисе и корректного конфигурирования канала связи между маршрутизаторами не пришлось изменять настройки сети Дополнительного офиса для обеспечения связи между сетевыми узлами Главного офиса и Дополнительного офиса?

2) Могут ли интерфейсы серверов находиться в одном *VLAN*?

3) Почему при использовании кадров разной длинны данные передаются из сети дополнительного офиса в сеть главного офиса?

8. Напишите программу, реализующую расчет контрольной суммы для заданного файла. Имя файла задается как параметр для опции --*file*. Размер файла должен быть не менее 2 Мбайт. Содержание кодируемого файла роли не играет.

# **ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ**

**Задание 1.**

Для настройки авторизации по каналу между маршрутизаторами по протоколу *CHAP*, необходимо на обоих маршрутизаторах задать доменное имя устройств(*ip-domain-name)* и создать по 1-ому пользователю, причём имя пользователя должно совпадать с *ip-domain-name*, заданном на другом маршрутизаторе. У обоих пользователей пароли должны совпадать.

Маршрутизатор главного офиса:

|  |
| --- |
| MainOffice(config)#ip domain-name MainOffice |
| MainOffice(config)#username Office password cisco |
| MainOffice(config)#interface Serial <номер интерфейса> |
| MainOffice(config-if)#encapsulation ppp |
| MainOffice(config-if)#ppp authentication chap |
| MainOffice(config-if)#clock rate 128000 |

Маршрутизатор дополнительного офиса:

|  |
| --- |
| Office(config)#ip domain-name Office |
| Office(config)#username MainOffice password cisco |
| Office(config)#interface Serial <номер интерфейса> |
| Office(config-if)#encapsulation ppp |
| Office(config-if)#ppp authentication chap |
| Office(config-if)#clock rate 128000 |

**Задание 2-3.**

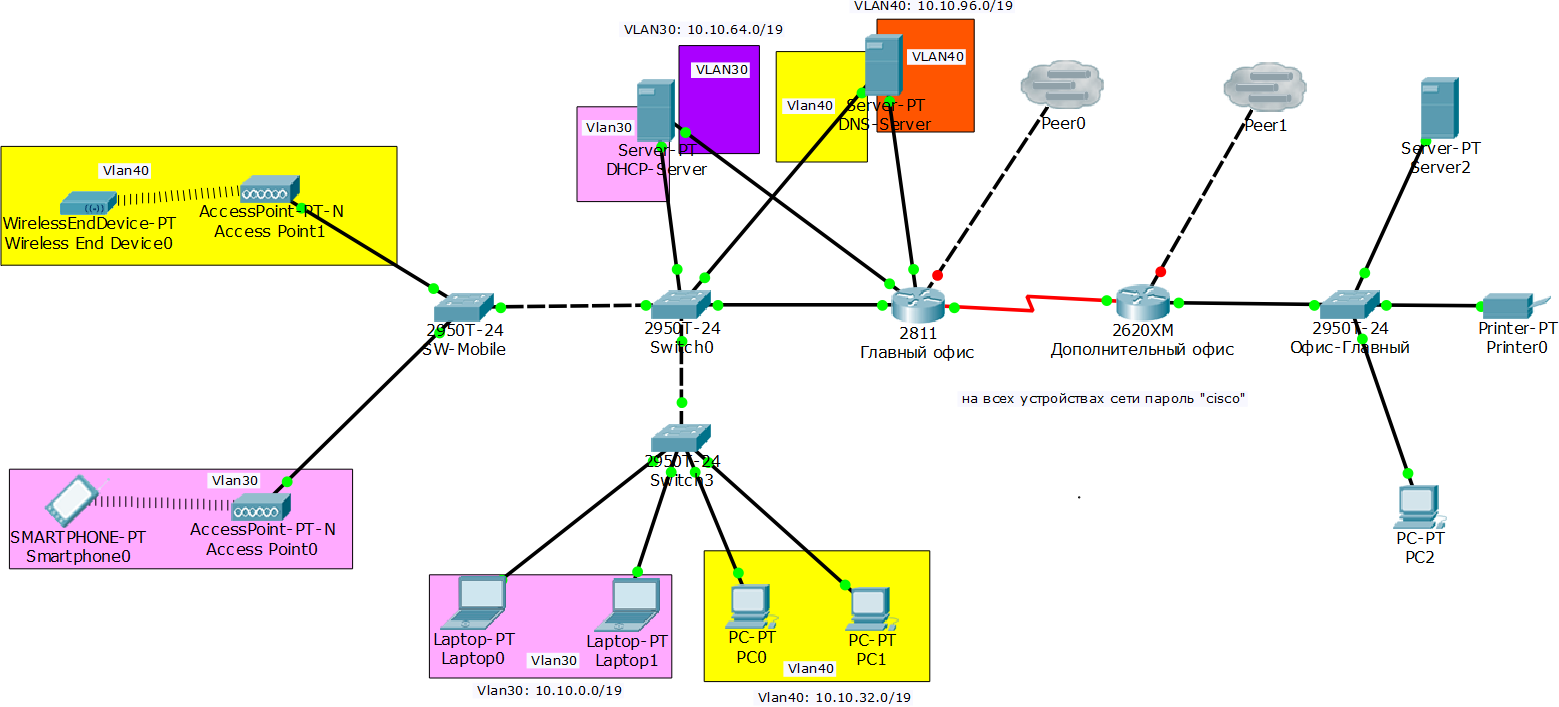


Рисунок 2.1 – Сеть, сконфигурированная в соответствии с заданием

Сеть главного офиса (10.10.0.0/18) делим на две подсети: 10.10.0.0/19 (для *Vlan30*) и 10.10.32.0/19 (для *Vlan40*). Это понадобится для настройки маршрутизации между виртуальными подсетями.

На трёх коммутаторах и маршрутизаторе главного офиса необходимо зайти в меню «*Config*→*SWITCHING*→*VLAN Database*», в поле «*VLAN Number*» ввести «30» и нажать «*Add*», затем в поле «*VLAN Number*» ввести «40» и нажать «*Add*». Имя *Vlan* (поле «*VLAN Name*») заполнять необязательно, имена *vlan*’ов будут заданы значениями по умолчанию («*VLAN30*» и «*VLAN40*»).

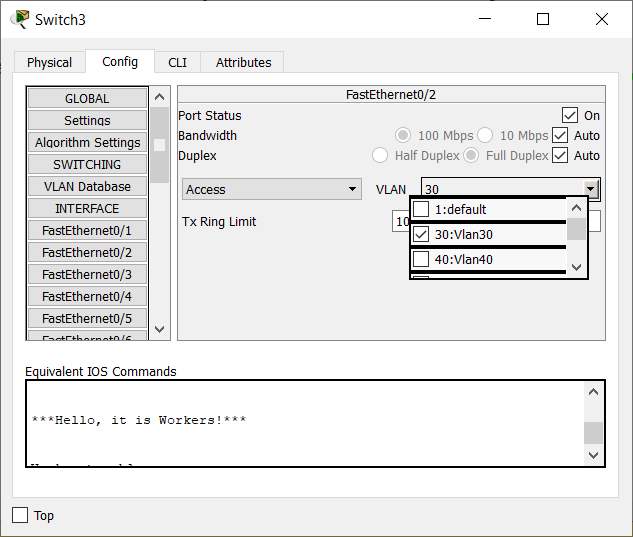
****

Рисунок 2.2 – Задаём режим работы порта

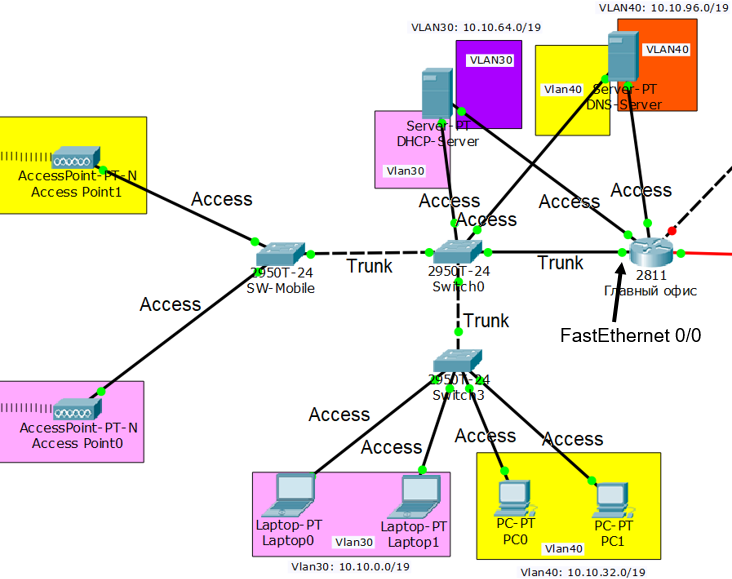
****

Рисунок 2.3 – Режимы работы портов коммутаторов

**Задание 4.**

Для того, чтобы обеспечить связь между локальными сетями главного офиса, необходимо на маршрутизаторе главного офиса перенастроить интерфейс, который соединяет маршрутизатор главного офиса с коммутатором *Switch0* (см. рисунок 2.3). На интерфейсе маршрутизатора создаём два подынтерфейса, задать формат передаваемого кадра *dot1q* и задаём им *IP*-адресы:

|  |
| --- |
| MainOffice(config)#interface fastEthernet 0/0.1 |
| MainOffice(config-subif)#encapsulation dot1q 30 (где 30 – номер vlan’а) |
| MainOffice(config-subif)#ip address 10.10.0.1 255.255.224.0 |
| MainOffice(config)#interface fastEthernet 0/0.2 |
| MainOffice(config-subif)#encapsulation dot1q 40 (где 40 – номер vlan’а) |
| MainOffice(config-subif)#ip address 10.10.32.1 255.255.224.0 |

**Задание 5.**

Сеть серверов главного офиса (10.10.64.0/18) разбиваем на две подсети: 10.10.64.0/19 и 10.10.96.0/19. На маршрутизаторе создаём два виртуальных интерфейса «*VLAN30*» и «*VLAN40*». Они будут играть роль шлюзов для устройств в локальных сетях.

|  |
| --- |
| MainOffice(config)# interface Vlan 30 |
| MainOffice(config-if)#ip address 10.10.64.1 255.255.224.0 |
| MainOffice(config-if)#no shutdown |
| MainOffice(config)# interface Vlan 40 |
| MainOffice(config-if)#ip address 10.10.96.1 255.255.224.0 |
| MainOffice(config-if)#no shutdown |

**Задание 6.**

В сети локального офиса расположен коммутатор «Офис Главный» (или «*OfficeMain*»). К нему присоединены все устройства, расположенные в сети дополнительного офиса, при этом используются порты *FastEthernet* с номерами от 0/1 до 0/10.

Соответственно, чтобы данные в локальной сети дополнительного офиса передавались кадрами размером 1290 октетов, необходимо ввести следующие команды:

|  |
| --- |
| OfficeMain(config)#interface range FastEthernet 0/1-10 |
| OfficeMain(config-if)#tx-ring-limit 1290 |

**Задание 7.**

1. Потому что 10.10.0.0/19 и 10.10.32.0/19 являются подсетями сети 10.10.0.0/18, и, если прописать статический маршрут в сеть 10.10.0.0/18, то он будет работать со всеми подсетями данной сети.

2. Не могут. Оба интерфейса могут входить в два разных *vlan*’а, имеющих одинаковый номер.

3. Коммутаторы передают данные в виде кадров длинной от 64 до 1522 байт. Для синхронизации приёмника и передатчика на физическом уровне, кадры дополняются вначале преамбулой (7 октетов со значением 0x55), разделителем начала кадра (октет со значением 0xD5) и в конце разделительным полем (12 октетов с пустым значением).

Поле «Длина кадра» (*L*, англ. *Length*) задает длину кадра. Если объём пересылаемых данных больше 1522, то указывается длинна кадра, и коммутатор может понять, что необходимо ожидать кадры оставшимися данными.

**Задание 8.**

Программа была написана на языке *Perl*. Для подсчёта контрольной суммы:

1. Генерируется таблица с 256 однобайтными константами
   1. Каждая ячейка инициализируется числом от 0 до 255
   2. Происходит побитовый сдвиг значения ячейки вправо на 1 бит. Если сдвинутый бит равен "1", то значение ячейки преобразуется операцией *XOR*, в качестве второго операнда используется образующий многочлен, равный *0xEDB88820*.
2. Хеш инициализируется значением 0хFFFFFFFF. Затем происходит побайтное преобразование входных данных по следующей формуле:

где *crc* – хеш, *table* – таблица констант, *data* – входные данные.

Для проверки корректности реализации программы возьмём два файла: «*book.pdf*» и «*image.png*», первый размером больше 2 Мбайт, второй меньше. В соответствии с заданием, программа должна обработать только первый файл. Также проверим корректность нахождения контрольной суммы файла с помощью программы «*7-Zip*».

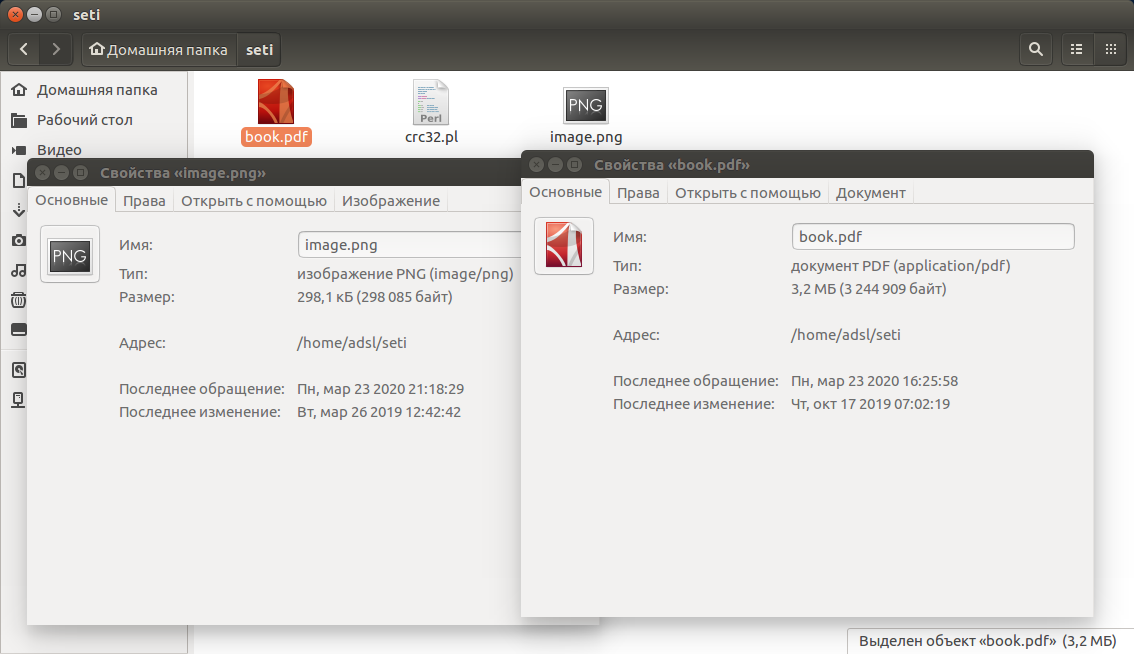


Рисунок 2.4 – Сведения о файлах

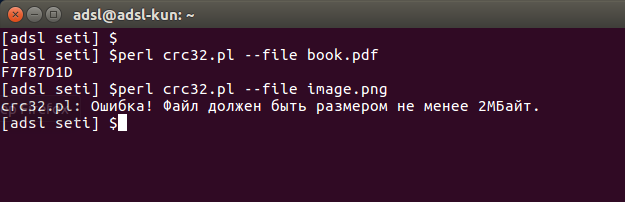


Рисунок 2.5 – Результат работы программы

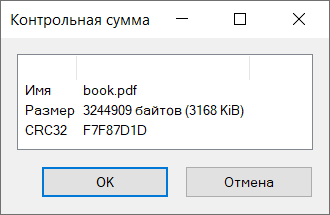


Рисунок 2.6 – Подсчёт контрольной суммы в «7-Zip»

**3 ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ**

**1. С какой целью разрабатывают форматы кадров?**

В локальных и глобальных сетях на канальном уровне используются различные протоколы и различные форматы кадров. В локальных сетях основным протоколом канального уровня является *Ethernet* и совместимые с ним. В глобальных соединениях "точка-точка" наиболее распространенным является протокол *Point-to-Point**Protocol* (*PPP)*. Формат кадров канального уровня практически одинаков для всех *Ethernet* совместимых технологий. Технология *Ethernet* предусматривает кадры четырех форматов, которые незначительно отличаются друг от друга.  Чтобы понять друг друга, отправитель и получатель должны использовать один и тот же тип кадров *Ethernet*. К счастью, кадры могут быть всего четырех разных форматов, и к тому же не сильно отличающихся друг от друга.

**2. Формат кадра по протоколу HDLC. Назначение полей.**

Протокол *HDLC* (англ. *Higher-level Data Link Control*) – это один из протоколов формирования последовательных каналов передачи информации в режиме точка-точка(и) или ведущий-ведомый. Кадры *HDLC* можно передавать, используя синхронные и асинхронные соединения. Формат кадра представлен на рисунке 3.1.

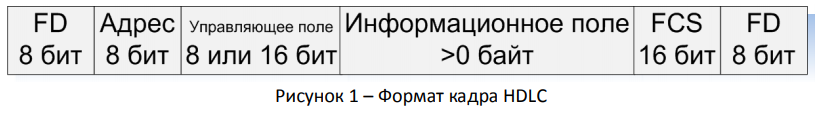


Рисунок 3.1 – Формат кадра HDLC

Поля *FD* (англ. *Frame Delimiter*) являются разделителями кадров и используются для синхронизации передатчика и приемника(ов). Это поле всегда содержит заданную последовательность бит - 0x7E. Уникальность значения поля гарантируется использованием специального механизма передачи кадра: «битстаффинга» в синхронных соединениях и «байтстаффинга» в асинхронных. Битстаффинг — вставка бита 0 после 5 подряд идущих битов, содержащих 1. В байтстаффинге используется escape-последовательность, то есть байт 0x7E в середине кадра заменяется последовательностью байтов (0x7D, 0x5E), а байт (0x7D) — последовательностью байтов (0x7D, 0x5D).

В поле «Адрес» содержится номер устройства-получателя кадра. В режиме «точка-точка» это поле используется для указания направления передачи кадра (от источника к приемнику или наоборот).

Управляющее поле занимает 1 или 2 байта. Его структура зависит от типа передаваемого кадра. Тип кадра определяется первыми битами управляющего поля: 0 — информационный, 10 — управляющий, 11 — ненумерованный тип.

Поле «Контрольная сумма кадра» (*FCS*, англ. *Frame Check Sequence*) содержит контрольную сумму кадра, с помощью которой приемник удостоверяется в неизменности передаваемых данных.

**3. Протокол PPP. Формат кадра. Назначение полей.**

Протокол *PPP* (англ. *Point-to-Point Protocol*) позволяет организовать автоматическое согласование параметров канала, авторизацию доступа к каналу передачи данных и мультиплексирование данных от нескольких протоколов сетевого уровня. Формат кадра по протоколу *PPP* представлен на рисунке 3.2.

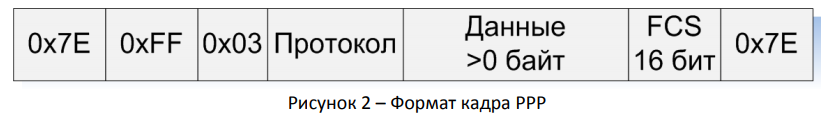


Рисунок 3.2 – Формат кадра PPP

При передаче данных по протоколу PPP поле адреса содержит 0xFF, а управляющее поле 0x03. Информационное поле разделено на две части: Протокол и Данные. В поле «Протокол» указывается номер протокола сетевого уровня, данные для которого передаются по каналу.

**4. Протоколы авторизации PAP и СHAP.**

Для авторизации участников взаимодействия по последовательному каналу, реализованному на основе протокола *PPP*, используются протоколы: *PAP* (англ. *Password Authentication Protocol*), *CHAP* (англ. *Challenge Handshake Authentication Protocol*) и другие.

Протокол *PAP* является самым простым и незащищенным (см. рисунок 3.3). После настройки физического соединения сторона-клиент (кто инициировал создание канала) отправляет в открытом виде пару «пользователь пароль». Сторона-сервер проверяет полученное значение и либо авторизует соединение, либо отказывает в этом. Обратим внимание, что имя пользователя и его пароль передаются по каналу связи в открытом виде, что является существенным недостатком. Этого недостатка лишен протокол *CHAP*, который использует три стадии авторизации. На первом шаге сторона-сервер отправляет стороне-клиенту некоторую случайно сгенерированную строку и свое имя. Используя имя сервера, клиент определяет пароль. Полученную строку и пароль сервера сторона-клиент формирует по алгоритму *MD5* хеш-строку, которую отправляет серверу. Сервер выполняет аналогичные действия и сверяет результат. Если результаты совпали, то он авторизует канал. В результате пароль пользователя не передается по каналу связи.

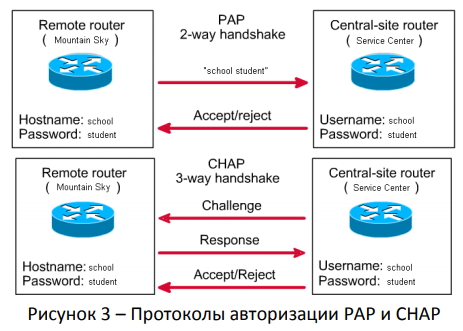


Рисунок 3.3 – Протоколы авторизации PAP и CHAP

**5. Форматы кадров стандарта Ethernet.**

Существует несколько форматов кадра (см. рисунок 3.4). Формат кадра *Ethernet DIX* разработан родоначальниками применения стандарта *Ethernet* – компаниями *Digital Equipment Corp* совместно с компаниями *Intel* и *Xerox* и был предложен сообществу *IEEE* для утверждения его в качестве стандарта. Однако сообществом *IEEE* принят немного иной стандарт, включающий разделение канального уровня на два подуровня: доступ к среде и управления каналом. Кроме того, компания *Novell* предложила свою версию стандарта. В результате на практике стали использоваться четыре разных формата фрейма. Кадр *Ethernet SNAP* (*SubNetwork Access Protocol* — протокол доступа к подсетям) представляет собой расширение кадра *802.3/LLC* за счет введения дополнительного заголовка.

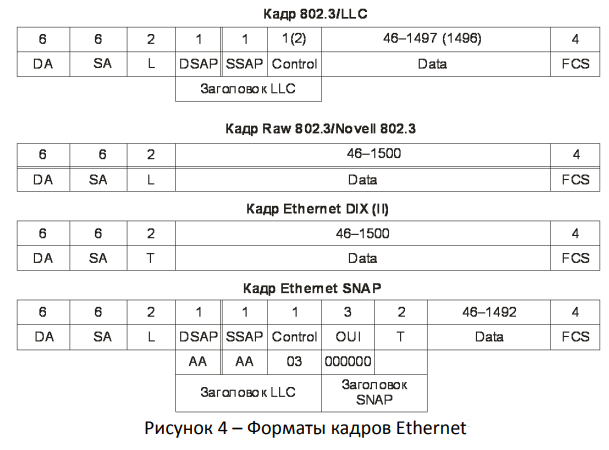


Рисунок 3.4 – Форматы кадров Ethernet

**6. Алгоритм автоматического определения формата кадра Ethernet.**

Сетевые адаптеры могут поддерживать все четыре формата кадров *Ethernet*. Для корректной передачи данных адаптер должен точно определить используемый тип кадра. Автоматически распознают тип кадра *Ethernet* с использованием значения полей кадров. Например, кадры *Ethernet* II легко отличить от других типов кадров по значению поля *L/T*: если оно больше 1500, значит, это поле является полем типа протокола (*T*), так как значения кодов протоколов выбраны так, что они всегда больше 1500. В свою очередь наличие поля T говорит о том, что это кадр *Ethernet* II, который единственный использует это поле в данной позиции кадра. Если кадр принадлежит к типу, отличному от *Ethernet DIX* (поле *L/T* имеет значение меньшее или равное 1500), то выполняется дальнейшая проверка по наличию или отсутствию полей *LLC*. Поля *LLC* могут отсутствовать только в том случае, если за полем длины идет 2-байтное поле, которое всегда заполняется единицами, что дает значение 0xFFFF. Ситуация, когда поля *DSAP* и *SSAP* одновременно содержат такие значения, возникнуть не может, поэтому наличие двух таких октетов говорит о том, что это кадр *Raw* 802.3. В остальных случаях дальнейший анализ проводится в зависимости от значений полей *DSAP* и *SSAP*. Если они равны 0xAA, то это кадр *Ethernet SNAP*, а если нет, то 802.3/*LLC*.

**7. Стандарт IEEE 802.1Q. Назначение. Пример применения.**

На практике часто возникает задача разделения устройств, подключенных к одному или нескольким коммутаторам на несколько непересекающихся локальных сетей. В случае, если используется только один коммутатор, то эта задача решается путем конфигурирования портов коммутатора, указав каждому порту к какой локальной сети он относится. Если же используется несколько коммутаторов (см. рисунок 3.5), то необходимо между коммутаторами помимо данных передавать информацию к какой локальной сети относится кадр. Для этого был разработан стандарт 802.1Q, называемый *VLAN* (англ. *Virtual Local Area Network*).

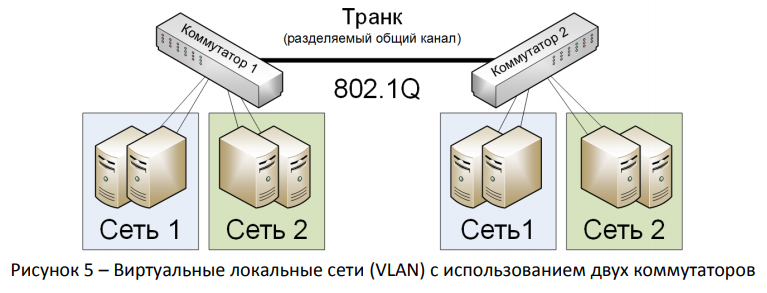


Рисунок 3.5 – Виртуальные локальные сети (VLAN) с использованием двух коммутаторов

**8. Конфигурирования последовательных интерфейсов на оборудовании CISCO.**

Формат кадра, используемый для передачи данных по каналу, определяется в режиме конфигурирования интерфейса, командой encapsulation (см. рисунок 3.6). Скорость передачи данных по последовательному каналу задается командой «*clock* *rate*». Параметры протокола *PPP* задаются одноименной командой. Параметр *authentication* задает используемый протокол аутентификации соединения (*PAP*, *CHAP* или оба). Имя пользователя и пароль, передаваемые при аутентификации по протоколу *PAP*, задаются параметром «*pap sent-username password*». Сторона-клиент использует для проверки полученных данных базу пользователей, формируемую командой «*username* *password*». В протоколе *CHAP* используется та же самая база пользователей, а в качестве имени, передаваемого на первом этапе, используется имя сетевого узла (см. рисунок 3.7).

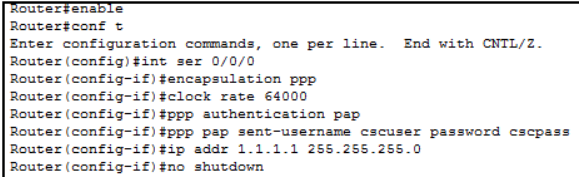
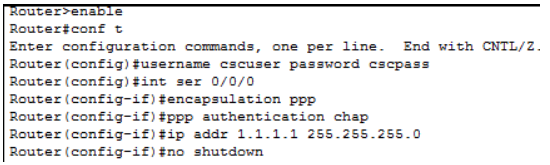


Рисунок 3.6 – Настройка протокола PAP на маршрутизаторе

 Рисунок 3.7 – Настройка протокола CHAP на маршрутизаторе

**9. Конфигурирование интерфейса Ethernet на оборудовании CISCO.**

Для интерфейсов, использующих стандарт *Ethernet* можно задать размер поля данных в передаваемых кадрах (см. рисунок 3.8). Делается это в режиме настройки интерфейса с помощью команды «*mtu*». Режим работы канала (дуплекс или симплекс) задается с помощью команды «*duplex*» (она также допускает использование режима автосогласования). Скорость работы канала задается командой «*speed*» (которая также допускает режим автосогласования).

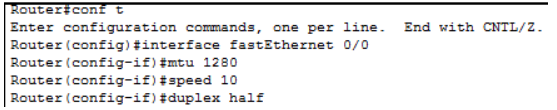


Рисунок 3.8 – Конфигурирования режима работы порта по стандарту Ethernet

**10. Настройка VLAN на оборудовании CISCO.**

Создание виртуальной локальной сети в коммутаторах *CISCO* начинается с её описания (см. рисунок 3.9). Делается это в привилегированном режиме с помощью команды «*Vlan* 2». В результате примера, приведенного на рисунке 3.6, в базе данных коммутатора будет создана виртуальная локальная сеть с номером 7 и ей дано имя «*nachalstvo*». Напомним, что всегда присутствует *VLAN* с номером 1 (имя «*default*»), в которую помещаются кадры, не имеющие тега виртуальной сети.

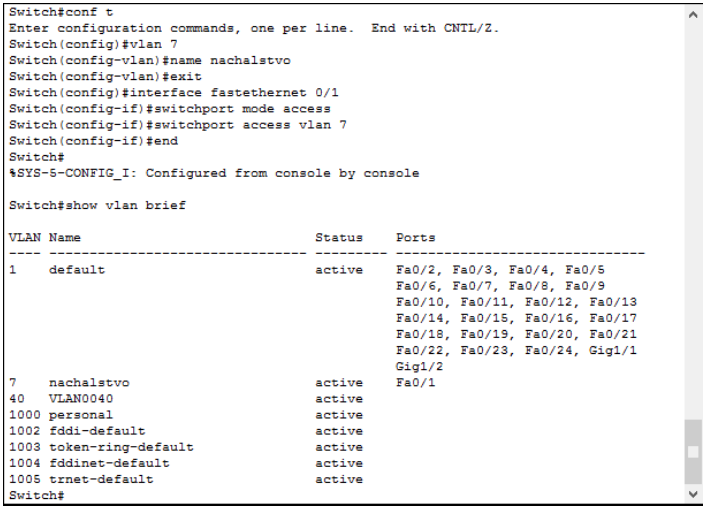


Рисунок 3.9 – Создание сети VLAN с номером 1 и настройка порта, входящего в эту сеть

**11. Реализация маршрутизации между VLAN.**

Каждому порту устройства задается режим работы (*access* или *trunk*) и указываются параметры виртуальной локальной сети. Делается это с помощью команды «*switchport*». По умолчанию все порты работает в режиме «*access*» и принадлежат *VLAN* с номером 1. Режим работы коммутационного порта задается командой «*switchport mode*». Режим «*trunk»* предполагает передачу кадров в формате 802.1*Q*. Режим «*access»* требует передачу и прием кадров без тега. При передаче кадра через порт тег убирается. При приеме кадра тег добавляется и пересчитывается контрольная сумма кадра. Указать к какой сети относится порт, работающий в режиме доступа, можно с помощью команды «*switchport access vlan*». Узнать к каким сетям относятся порты можно в привилегированном режиме с помощью команды «*show vlan brief*».

В режиме «*trunk»* порту необходимо указать кадры каких *VLAN* допустимо передавать через него и какой *VLAN* будет использовать нетегированные кадры. Допустимые *VLAN* указываются командой «*switchport trunk allowed vlan*». Номер локальной сети для не тегированного трафика задается командой «*switchport trunk native vlan*» (см. рисунок 3.10).

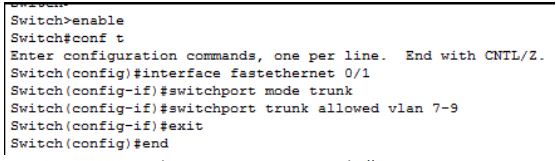


Рисунок 3.10 – Конфигурирование интерфейса в режиме «trunk*»*.

**12. Алгоритм циклического избыточного кодирования.**

При передаче данных принимающей стороне необходимо удостовериться, что данные получены в неизменном виде. Одним из подходов, используемых для решения это задачи, является использование дополнительного поля в кадре, в которые помещаются данные, вычисленные на основе передаваемых данных. В результате принимающая сторона, выполняя аналогичные вычисления сверяет результат и если он совпадает со значением поля, то считается, что данные переданы без изменения. Очевидно, что создание таких алгоритмов расчета контрольных сумм, которые бы давали заданный уровень гарантии определения изменения данных, является актуальной и нетривиальной задачей.

На практике широкое распространение получил алгоритм циклического избыточного кодирования (англ. *Cyclic Redundancy Code*). Алгоритм *CRC* базируется на свойствах деления с остатком двоичных многочленов. В качестве контрольной суммы используется остаток от деления многочлена, соответствующего входным данным, на некий фиксированный порождающий многочлен. В зависимости от вида порождающего многочлена и его длины, изменяется вероятность совпадения контрольных сумм для различных исходных данных и время контрольного суммирования.

Существует достаточно большое разнообразие порождающих многочленов для алгоритмов контрольного суммирования *CRC* – 8, 16, 32 И 64, подобранных на основе теории кодирования и многочисленных исследований.

В вычислительной технике оперировать с полиномами N–степени - неудобно и ресурсоемко. Поэтому полиномы заменяют бинарными последовательностями и вычисляют контрольную сумму, оперируя уже не с полиномами, а с бинарными данными.

Деление можно заменить повторением операций вычитания. Таким образом, вычитание полиномов сводится к операции «исключающего или» с элементами полинома, имеющими одну и ту же степень, а, следовательно, можно заменить вычитание полиномов на операцию «исключающие или», с сопоставленными им бинарными последовательностями. Пример определения контрольной суммы c использованием полинома *CRC*8 приведен на рисунке 3.11.

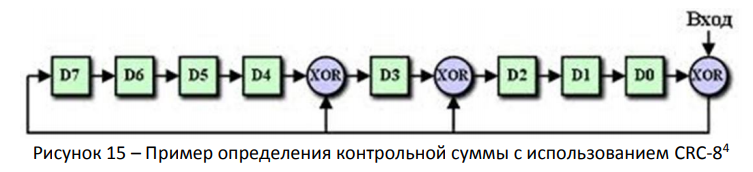


Рисунок 3.11 – Пример определения контрольной суммы с использованием CRC-8

**13. Таксономия алгоритмов циклического избыточного кодирования.**

Для описания алгоритмов циклического избыточного кодирования принято использовать таксономию:

* Name: название алгоритма;
* Width: степень используемого полинома (разрядность регистра результата);
* Poly: Порождающий полином (задается в виде бинарных значений коэффициентов);
* Init: Начальное значение регистра результата;
* Refin: Порядок формирования кодируемой последовательности. False — начиная со старшего значащего бита (MSB-first), или True — с младшего (LSB-first);
* RefOut: инвертируется ли порядок битов регистра перед выполнением операции XOR;
* XorOut: Значение, с которым выполняется операция XOR;
* Check: Проверочный результат расчета по алгоритму CRC для последовательности, формируемой из строки ASCII символов «123456789» (9 октетов). Поле не является обязательным.

**ПРИЛОЖЕНИЕ A. Исходный код задания №8**

use strict; #Подключаем

use lib qw(blib/lib lib); #

use Archive::Zip; #библиотеки

use FileHandle; #

#Основная программа

my $\_2MiB = 2097152; #2 Мегабайта в байтах

my $file\_name = $ARGV[1]; #Имя файла

my $file\_size = -s "$ARGV[1]"; #Получаем размер файла в байтах

if ($ARGV[0] eq "--file") { #Сравнение строк

if (-d $file\_name) { #Проверка, является ли $file\_name директорией

warn "$0: Ошибка! \"${file\_name}\": Это не файл, а директория.\n";

exit 10;

} else {

my $file = FileHandle->new(); #Объявление файла

if (!$file->open($file\_name, 'r')) { #Проверка, существует ли файл

warn "$0: Ошибка! Файл \"${file\_name}\" не существует.\n";

exit 20;

}

if ($file\_size < $\_2MiB) { #Проверка, весит ли файл меньше, чем 2 МБайта

warn "$0: Ошибка! Файл должен быть размером не менее 2МБайт.\n";

exit 30;

}

binmode($file); #Устанавливает дескриптор для чтения файла в текстовый режим

my $data; #Входные данные(содержимое файла)

my $crc = 0; #Контрольная сумма

#Получение контрольной суммы файла

while ($file->read($data, 32768)) {

$crc = Archive::Zip::computeCRC32($data, $crc);

}

my $file\_crc = sprintf("%X", $crc); #Преобразуем в 16-ричное число

printf("$file\_crc\n"); #Вывод результата на экран

exit 0;

}

} else {

warn "$0: Ошибка! Программе переданы некорректные параметры запуска.\nПример: perl crc32.pl --file \"имя\_файла\"\n";

exit 40;

}