БГУИР

Кафедра ЭВМ

Лабораторная работа №5

Детерминированные методы доступа к моноканалу

Выполнил Проверил:

Студент группы 750502 Глоба А.А

Никанов И.В.

Минск 2019

**Основные теоретические сведения:**

Если случайные методы уместно использовать при шинной топологии, применительно к которой четко выражена возможность возникновения коллизий, то детерминированные методы хорошо «ложатся» на кольцевую топологию.

Концептуальная разница между случайными и детерминированными методами заключается в том, возникает ли случайность

при «обращении» станции к моноканалу.

Кольцо можно рассматривать как своеобразный моноканал, один такт работы которого представляет собой полный либо

частичный «обход» кадром всех станций. Более подробно типичную кольцевую топологию можно представить следующим

образом.

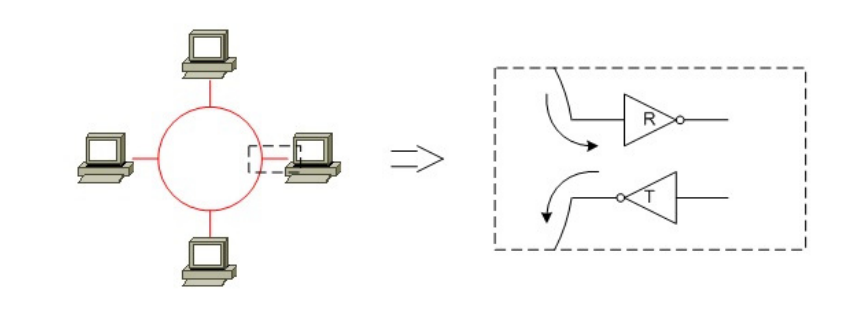


Рис. 1. Цепи передатчика и приемника типичной кольцевой топологии

1. ФОРМАТЫ КАДРОВ TOKEN RING И НАЗНАЧЕНИЕ ОСНОВНЫХ ПОЛЕЙ

В стандарте предусмотрены четыре вида передаваемых последовательностей:

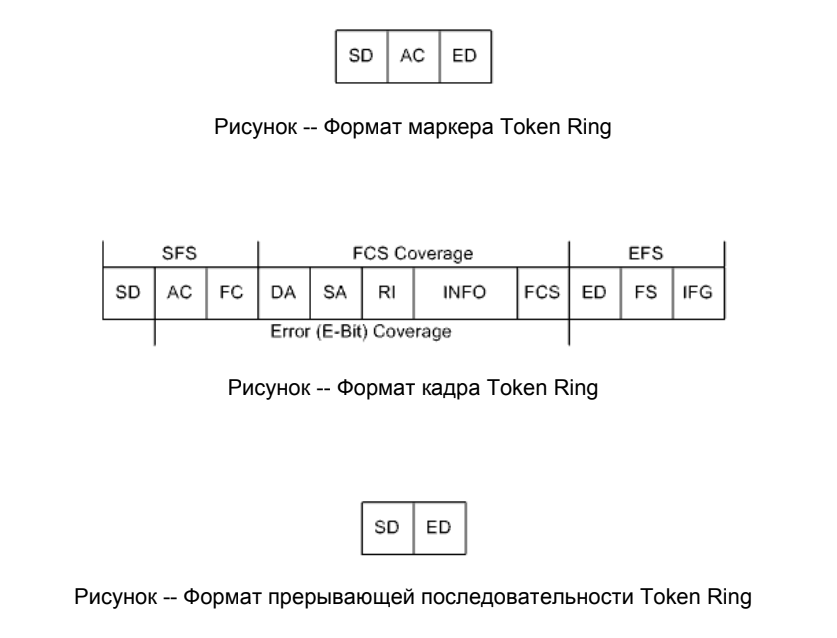
1. Token -- маркер.

2. Frame -- кадр.

3. Abort Sequence -- прерывающая последовательность.

4. Fill -- заполняющая последовательность.

Каждая из станций в любое время должна распознавать (и различать) маркеры, кадры и специальные последовательности.



Поля:

1. SD (Starting Delimiter) -- начальный разделитель.

2. AC (Access Control) -- контроль доступа.

3. FC (Frame Control) -- контроль кадра.

4. DA (Destination Address) -- адрес назначения.

5. SA (Source Address) -- адрес источника.

6. RI (Routing Information) -- информация о маршрутизации (может отсутствовать).

7. INFO (Information) -- данные (могут отсутствовать).

8. FCS (Frame Check Sequence) -- контрольная сумма.

9. ED (Ending Delimiter) -- конечный разделитель.

10. FS (Frame Status) -- состояние кадра.

11. IFG (InterFrame Gap) -- межкадровый интервал.

С точки зрения алгоритма контроля доступа наибольший интерес представляет одноименное поле, а также поле состояния кадра.



Где:

1. P (Priority bits) -- текущий уровень приоритета.

2. T (Token bit) -- идентификатор маркера: 0 -- маркер, 1 -- кадр.

3. M (Monitor bit) -- бит монитора.

4. R (Reservation bits) -- запрашиваемый уровень приоритета.



Где:

1. A (Address-recognized bit) -- флаг распознания адреса (дублируется).

2. C (frame-Copied bit) -- флаг копирования кадра (дублируется).

3. r (reserved) -- зарезервировано.

**Код лабораторной работы**

**Файл Main.java:**

timer = new Timer();  
  
 stationForms.get(0).getStartButton().setOnAction((event) -> { //высставляем действие на кнопку старт  
 stationModel = new StationModel();  
 for (int i = 1; i <= 3; i++) {  
 int currentStation = i;  
  
 timer.schedule(new TimerTask() {  
 @Override  
 public void run() { //начать выполнять задачу с интервалом  
 System.out.println(currentStation); //выводим в консоль на какой станции сейчас маркер  
 try {  
 stationRoutine(stationForms.get(currentStation - 1), stationModel); //  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 },1000, 1000);  
  
 stationForms.get(0).getStartButton().setDisable(true);  
 }  
 });  
  
 stationForms.get(0).getTokenText().textProperty().addListener(observable -> sendPackage(stationForms.get(0)));//установить слушатель, на тот случай  
 stationForms.get(1).getTokenText().textProperty().addListener(observable -> sendPackage(stationForms.get(1)));//если на нас висит маркер  
 stationForms.get(2).getTokenText().textProperty().addListener(observable -> sendPackage(stationForms.get(2)));//и в поле ввода есть текст для отправки  
}  
  
public static void stationRoutine(StationForm stationForm, StationModel myStationModal) throws InterruptedException {  
 if (myStationModal.getControl() == 0) {  
 myStationModal.setSource(stationForm.getSourceAddress()); //выставляем параметры для станции  
 myStationModal.setDestination(stationForm.getDestinationAddress()); //и сидим здесь, пока нет текста  
 myStationModal.setMonitor(stationForm.getMainStation()); //установить управляющую станцию  
 } else { //как только появился текст  
 runOnUIThread(() -> {  
 if (myStationModal.getDestination() == stationForm.getSourceAddress()) { //если есть еще символы для отправки  
 String data = "";  
 data += (char) myStationModal.getData(); //получаем символ, который мы получили из ф-ии sendPackage  
 myStationModal.setStatus((byte) 1); //выставляем когда уже получили символ  
 stationForm.getOutputArea().appendText(data); //отображаем полученный символ на станции-приемнике  
 }  
 if (myStationModal.getStatus() == 1 && stationForm.getMainStation() == 1) { //заходим сюда, если до этого мы отправили символ  
 myStationModal.setControl((byte) 0); //выставляем на случай, если символ был последним  
 myStationModal.freeData(); //очищаем поле data для последующих данных  
 myStationModal.setStatus((byte) 0); //  
 }  
 });  
 }  
 runOnUIThread(() -> stationForm.getTokenText().setText(MARKER));  
 Thread.sleep(1000);  
 runOnUIThread(() -> stationForm.getTokenText().setText(""));  
}  
  
public void sendPackage(StationForm stationForm) {  
 if (!stationForm.getInputArea().getText().equals("") //если в поле ввода текста не пусто  
 && !stationForm.getDestination().equals("") //и у нас есть адрес куда нужно отправить  
 && stationModel.getControl() == 0 //и мы не заняты(т.е. не отправляем в текущий момент)  
 && stationModel != null) { //и станции созданы  
 if (stationForm.getTokenText().getText().equals("\*")) { //если маркер у нас  
 if (stationForm.getDestination().equals("3")) {  
 stationForm.setDestinationAddress(addresses[0]);  
 } //выставляем адрес отправки  
 if (stationForm.getDestination().equals("9")) {  
 stationForm.setDestinationAddress(addresses[1]);  
 } //выставляем адрес отправки  
 if (stationForm.getDestination().equals("27")) {  
 stationForm.setDestinationAddress(addresses[2]);  
 } //выставляем адрес отправки  
 stationModel.setControl((byte) 1); //показываем что теперь занято(чтобы другие не могли отправлять одновременно с нами)  
 stationModel.setDestination(stationForm.getDestinationAddress());//выставляем адрес приемника  
 stationModel.setSource(stationForm.getSourceAddress()); //выставляем адрес источника  
 stationModel.setMonitor(stationForm.getMainStation()); //выставляем станцию-монитор  
  
 String reduced = stationForm.getInputArea().getText().substring(1); //получаем новую подстроку(без 1-го символа) на станции-источнике  
 stationModel.setData(stationForm.getInputArea().getText().getBytes()[0]);//отправляем 1ый символ в data на станцию-приемник  
 stationForm.getInputArea().setText(reduced); //выставляем новую подстроку(без 1-го символа) на станции-источнике  
 }  
 }  
}