**BC/NW 2017 № 1 (30):6.2**

**СПОСОБ КОНТРОЛЯ КАЧЕСТВА ЦИФРОВЫХ КАНАЛОВ ПЕРЕДАЧИ ДАННЫХ**

Прохоров А.М

В данной работе рассматривается модель распределенной территориальной сети автоматической системы управления (АСУ) реального масштаба времени (РМВ), использующая цифровые каналы связи. В данных системах необходима надежная связь с минимально возможной задержкой. Цифровые каналы связи перспективных АСУ РМВ имеют большие преимущества по сравнению с аналоговыми. Передача данных по цифровым каналам связи позволяет:

- обеспечить лучшую помехозащищенность;

- повысить живучесть территориальных АСУ, создавая дополнительные обходные маршруты;

- в разы уменьшить задержки доведения данных.

Структура цифрового канала связи представлена на рис. 1. Канал связи передает поток E1 со скоростью передачи данных 2048 Кб/с, состоящий из 30 информационных и 2 служебных каналов (по 64 Кб/с на канал).



**Рис. 1.** Структура цифрового канала связи

Цифровые каналы связи в перспективных АСУ обеспечивают прямое взаимодействие между центральными вычислительными комплексами (ЦВК) взаимодействующих комплексов средств автоматизации (КСА).

Для обеспечения высокой надежности функционирования АСУ применяется резервирование каналов. В настоящее время в АСУ РМВ используется резервирование (дублирование) каналов связи. Обычно применяется один основной проводной канал связи и один резервный (обычно радиорелейный канал). В перспективных территориальных АСУ специального назначения для обеспечения высокой надежности и живучести предполагается дополнительно использовать в качестве третьего и четвертого каналов обходные маршруты. В системе передачи данных территориальной АСУ необходимо иметь информацию о текущем состоянии каналов передачи данных, чтобы своевременно принимать решение о переходе на резервный канал и**/**или принимать необходимые меры к восстановлению качественной работы каналов связи. Для этого необходим алгоритм контроля состояния канала. Одним из таких алгоритмов является алгоритм «скользящее окно» [1]. Суть скользящего окна заключается в том, что каждый из n подряд следующих выделенных интервалов имеет собственный весовой коэффициент Kn. При накоплении определённого количества данных для нового интервала данные самого старого из интервалов удаляются, так что при вычислении скользящего среднего всегда учитываются результаты только n последних интервалов.

Результат Kкач по методу взвешенного скользящего среднего вычисляется по формуле:

Kкач = k1 (N1ош / N1 общ) + k2 (N2 ош / N2 общ) + …+ kn (Nn ош / Nn общ) (1)

где kn– весовой коэффициент интервала, Nn ош – количество ошибочно принятых кодограмм в интервале n, а Nn общ– общее количество принятых кодограмм в интервале n.

Если N1 общ = N2 общ = Nn общ= Nобщ, формула преобразуется в

Kкач = k1 (N1ош / Nобщ) + k2 (N2 ош / Nобщ) + …+ kn (Nn ош / Nобщ) (2)

Прежде, чем переходить к принципам программной реализации, необходимо рассмотреть вопросы, связанные с передачей данных по сети.

С точки зрения контроля состояния каналов необходимо рассмотреть канальный и транспортный уровни эталонной модели взаимодействия открытых систем, т.к. на этих уровнях может применяться восстановление данных, что приведет к невозможности оценки качества канала на уровне пользователя.

Канальный уровень может предоставлять различные сервисы [2]. Их набор может быть разным в разных протоколах.

1. Сервис без подтверждений, без установки соединения.

2. Сервис с подтверждениями, без установки соединения.

3. Сервис с подтверждениями, с установкой соединения.

В сервисе без подтверждений и без установки соединения никакие соединения заранее не устанавливаются и не разрываются после передачи кадров, и если какой-либо кадр теряется из-за помех в линии, то на канальном уровне не предпринимается никаких попыток восстановить его.

В следующем сервисе с подтверждениями и без установки соединения также не устанавливаются заранее соединения, но получение каждого кадра подтверждается. Таким образом, отправитель знает, дошел ли кадр до пункта назначения или потерялся.

Наиболее сложным сервисом, предоставляемым канальным уровнем, является ориентированный на соединение сервис с подтверждениями. При использовании данного метода источник и приемник, прежде чем передать друг другу данные, устанавливают соединение. Каждый посылаемый кадр нумеруется, а канальный уровень гарантирует, что каждый посланный кадр действительно принят на другой стороне канала связи.

Для рассматриваемой модели предпочтительнее всего будет сервис без подтверждений и без установки соединений. Данный класс сервисов применяется в линиях связи реального времени, в которых восстановление кадра увеличит задержки, что приводит к приему устаревших данных. Кроме того, восстановление кадров на канальном уровне не позволяет видеть ошибки в данных программному обеспечению, которое работает на самом верхнем уровне эталонной модели.

При рассмотрении транспортного уровня нужно отметить, что транспортные протоколы в некоторых отношениях напоминают протоколы канального уровня. На транспортном уровне нашли применение два основных протокола – UDP (User Datagram Protocol) и TCP (Transmission Control Protocol).

UDP аналогичен сервису без подтверждений и без установления соединения на канальном уровне: данный протокол не занимается управлением потоком данных, контролем перегрузки, повторной передачей после приема испорченного сегмента. Все это перекладывается на пользовательские процессы.

TCP аналогичен сервису с подтверждениями и с установлением соединения на канальном уровне: он устанавливает соединение, управляет потоком данных, упорядочивает данные и т.д.

Для рассматриваемой модели будет предпочтительнее использование протокола UDP, т.к это позволит обнаруживать ошибки на пользовательском уровне. Кроме того, ему требуется меньшее количество байтов служебной нагрузки и, в итоге, меньше времени для передачи данных, чем протоколу TCP. Например, если целью соединения было отправить запрос, занимающий один сегмент, и получить ответ, также занимающий один сегмент, то при использовании TCP будет задействовано восемь сегментов. Если же используется UDP, произойдет обмен только двумя сегментами: запрос и ответ.

Теперь возникает вопрос: как считать количество ошибок с целью оценки качества канала? Можно использовать следующий подход. С некоторой периодичностью передавать фиксированное число N кодограмм (если количество информационных кодограмм будет меньше N, то необходимо добавить контрольные кодограммы до числа N). Приемная сторона будет знать, что она должна получить N кодограмм и в случае, если получено менее N кодограмм, то можно легко посчитать количество потерянных кодограмм данных и, следовательно, оценить качество канала.

В итоге получаем. На передающей стороне:

N = I + K (3)

где N – общее число кодограмм, I – информационные кодограммы, K – контрольные кодограммы (K ϵ [0;N]).

На приемной стороне:

Nош = N – Nпрн (4)

где Nош – число потерянных кодограмм, N – общее число кодограмм, Nпрн – число принятых кодограмм.

Таким образом, можно применить метод «скользящего окна» для оценки состояния канала связи.

При разработке программного обеспечения необходимо рассмотреть сетевые сокеты. Сокеты – механизм взаимодействия процессов, причем сетевые сокеты позволяют обмениваться данными процессам как на одном хосте, так и процессам на разных хостах.

Сокеты характеризуются тремя атрибутами: доменом, типом и протоколом [3].

Домены задают рабочую среду, которую будет использовать соединение сокетов. Самый популярный домен сокетов – AF\_INET, применяемый во многих локальных сетях и в Интернете.

Существует два типа сокетов: потоковые и дейтаграммные. Потоковые сокеты обеспечивают соединение, представляющее собой последовательный и надежный двунаправленный поток байтов. Потоковые сокеты реализованы в домене AF\_INET соединениями на базе протоколов TCP/IP. В отличие от потоковых дейтаграммные сокеты не устанавливают и не поддерживают соединение. Дейтаграммные сокеты также реализованы в домене AF\_INET, но с помощью соединений UDP/IP.

Если низкоуровневый механизм передачи данных позволяет применять несколько протоколов, предоставляющих сокет требуемого типа, можно выбрать конкретный протокол или сокет. Обычно на более низком уровне применяется один протокол, поэтому он выбирается как протокол по умолчанию.

При разработке программного обеспечения на передающей стороне необходимо применять алгоритм, изображенный на рис. 2. На приемной стороне необходимо применять алгоритм, изображенный на рис. 3.

|  |  |
| --- | --- |
| **Рис. 2.** Алгоритм на передающей стороне | **Рис. 3.** Алгоритм на приемной стороне |

**Заключение**

Необходимо отметить, что данный способ организации обмена данными позволяет работать с минимальными задержками, что требуется в системах реального масштаба времени, одновременно с этим производя контроль качества канала, что необходимо для обеспечения надежности системы.

**Литература**

1. Патент на изобретение № 2598807«Способ контроля качества каналов передачи данных», Савватеев В.С., Нимира М.Г.

2. Таненбаум, Э. Компьютерные сети / Э.Таненбаум, Д.Уэзеролл. – СПб.: Питер, 2012. – 960 с.

3. Мэтью, Н. Основы программирования в Linux / Н.Мэтью, Р.Стоунс. – СПб.: БХВ-Петербург, 2009. – 896 с.

4. Стивенс, У.Р. UNIX: разработка сетевых приложений / У.Р.Стивенс. – СПб.: Питер, 2003. – 1088 с.