DR. BOB DAVIDOV

UDP обмен данными

Цель работы: изучение механизма передачи данных через UDP канал.

Задача работы: построение UDP канала для обмена данными.

Приборы и принадлежности: Два персональных компьютера, LAN – кабель, среда МатЛАБ, локальная сеть.

ОБЩИЕ СВЕДЕНИЯ

UDP (User Datagram Protocol) – один из протоколов транспортного уровня модели OSI. UDP протокол (из семейства TCP/IP) отвечает транспортному уровню только наличием порта. Протокол обеспечивает обмен датаграммами между приложениями, ограничивается контролем целостности данных в рамках одной датаграммы. В отличие TCP протокола, UDP не исключает возможности потери пакета целиком, или дублирования пакетов, нарушение порядка получения пакетов данных.

Модель OSI		
Тип данных	**Уровень (layer)**	**Функции**
Данные	7. Прикладной (application)	Доступ к сетевым службам
	6. Представления (presentation)	Представление и кодирование данных
	5. Сеансовый (session)	Управление сеансом связи
Сегменты	4. Транспортный (transport)	Прямая связь между конечными пунктами и надежность
Пакеты	3. Сетевой (network)	Определение маршрута и логическая адресация
Кадры	2. Канальный (data link)	Физическая адресация
Биты	1. Физический (physical)	Работа со средой передачи, сигналами и двоичными данными

Пакет расширения Instrument Control Toolbox среды МатЛАБ дает возможность осуществлять взаимодействие с измерительными приборами, такими как осциллографы и генераторы, непосредственно из среды MATLAB. Пакет позволяет взаимодействовать с оборудованием через широко распространенные протоколы, такие как GPIB, VISA, TCP/IP и UDP. Взаимодействие является двусторонним - можно как выводить данные из MATLAB, направляя их приборам, так и считывать данные для анализа и визуализации. Новые возможности:

- Поддержка драйверов оборудования, включая IVI, VXIplug&play, а также драйверов MATLAB, позволяющих взаимодействовать с приборами без необходимости изучения их специфических команд
- Новый графический интерфейс пользователя (tmtool) для обнаружения измерительного оборудования, его конфигурирования и взаимодействия с ним
- Средства разработки и тестирования драйверов оборудования
- Средства модификации драйверов IVI и VXIplug&play для включения в них MATLAB-процедур анализа данных

Полезные команды МатЛАБ.

Команда	Назначение
tcpipinfo = instrhwinfo('udp')	Имя и адрес компьютера
u=instrfindall u = instrfind('Type', 'udp')	Поиск и восстановление UDP и других объектов
u = udp('localhost',’’); u = udp('127.0.0.1',4012);	Создание UDP объекта По умолчанию назначается порт 9090
u = udp('localhost', port1, 'LocalPort', port2);	Создание UDP объекта для обмена с внешней средой
echoudp('state',port) echoudp('on’) echoudp('off) Пример. echoudp('on',4012);	Подключение к порту “Эхо” сервера. В этом режиме можно считывать данные переданные серверу “Эхо”.
fopen(u); fclose(u)	Подключение (отсоединение) объекта к серверу
fwrite(u,65:74) A = fread(u,10);	Запись и чтение данных
fprintf(u, '0.80'); data = fscanf(u, '%f')	Запись и чтение данных
fwrite(u, 1:250, 'int32'); data = fread(u, 250, 'int32');	Запись и чтение бинарных данных MATLAB Description 'uchar' unsigned character, 8 bits. 'schar' signed character, 8 bits. 'int8' integer, 8 bits. 'int16' integer, 16 bits. 'int32' integer, 32 bits.

	'uint8' unsigned integer, 8 bits. 'uint16' unsigned integer, 16 bits. 'uint32' unsigned integer, 32 bits. 'single' floating point, 32 bits. 'float32' floating point, 32 bits. 'double' floating point, 64 bits. 'float64' floating point, 64 bits. 'char' character, 8 bits (signed or unsigned). 'short' integer, 16 bits. 'int' integer, 32 bits. 'long' integer, 32 or 64 bits. 'ushort' unsigned integer, 16 bits. 'uint' unsigned integer, 32 bits. 'ulong' unsigned integer, 32 bits or 64 bits. 'float' floating point, 32 bits.

fprintf(u, '%s', 'Request Time');
fprintf(u, '%s\n', 'Request Time');
data = fscanf(u)
	Запись и чтение ASCII данных

u = udp(..., 'InputBufferSize', Val); u.InputBufferSize = Val	Изменение размера входного буфера. По умолчанию, размер буфера – 512 байт.
get(u,{'Name','RemoteHost', 'RemotePort','Type','LocalPort'}) get(u, 'Status') get(u, 'OutputBufferSize') get(u, 'InputBufferSize') get(u, 'ValuesSent') get(u, 'ValuesReceived') get(u, 'BytesAvailable')	Считывание и отображение UDP параметров
flushinput(u);	Удаление входных данных. Очистка буфера.

fclose(u);
delete(u);
clear u
echoudp('off')
	Удаление UDP объекта

Характеристики UDP канала:

- Переданные числа с дробной частью принимаются целыми числами.
- Приемник принимает переданные отрицательные числа нулями.
- Диапазон передаваемых чисел 0 .. 255
- Скорость чтения порта ~ 1 000 байт/с
- Скорость записи в порт ~ 10 000 символов/сек байт/с
- 'fread' функция ограничена чтением 512 байт

Список параметров UDP объекта

```
>> get(u1)
   ByteOrder = bigEndian
```

BytesAvailable = 0
BytesAvailableFcn =
BytesAvailableFcnCount = 48
BytesAvailableFcnMode = terminator
BytesToOutput = 0
ErrorFcn =
InputBufferSize = 60000
Name = UDP-192.168.1.147
ObjectVisibility = on
OutputBufferSize = 512
OutputEmptyFcn =
RecordDetail = compact
RecordMode = overwrite
RecordName = record.txt
RecordStatus = off
Status = open
Tag =
Timeout = 300
TimerFcn =
TimerPeriod = 1
TransferStatus = idle
Type = udp
UserData = []
ValuesReceived = 0
ValuesSent = 0

UDP specific properties:
DatagramAddress =
DatagramPort = []
DatagramReceivedFcn =
DatagramTerminateMode = on
InputDatagramPacketSize = 8192
LocalHost =
LocalPort = 9000
LocalPortMode = manual
OutputDatagramPacketSize = 8192
ReadAsyncMode = continuous
RemoteHost = 192.168.1.147
RemotePort = 9001
Terminator = LF

ПРИМЕРЫ ПОЛУЧЕНИЯ ПРОВЕРЕННЫХ РЕЗУЛЬТАТОВ И ВАРИАНТЫ ДЛЯ САМОКОНТРОЛЯ

Задание 1. Работа udp соединения в режиме Эхо

1. Создайте UDP объект с портом 9090
   ```
   >> u=udp('localhost', 9090)
   ```
   или
   ```
   >> u=udp('localhost') % по умолчанию - связь с портом 9090
   ```
   или
   ```
   >> u=udp('192.168.0.223', 9090); % IP адрес  -  не статический
   ```

   Внимание! проверяйте состояние UDP объекта на каждом шаге выполнения задания.
2. Откройте объект
   ```
   >> fopen(u)
   ```
3. Запишите данные в объект, например,
   ```
   >> fwrite(u, 1:10);
   ```
4. Выполните команду чтение данных:
   ```
   >> A=fread(u, 10);
   ```
   Сообщение МатЛАБ:
   ```
   Warning: The specified amount of data was not returned within the Timeout
            period.
   ```
   Почему массив A - пустой?
5. Переведите UDP объект на работу в режиме Эхо.
   ```
   >> echoudp('on',9090)
   ```
6. Опять запишите данные в объект, например,
   ```
   >> fwrite(u, 1:10);
   ```
7. Выполните команду чтение данных:
   ```
   >> A=fread(u, 10);
   ```
   Чему равен параметр объекта "BytesAvailable". Почему?
8. Запишите следующие данные,
   ```
   >> fwrite(u, 3:2:10);
   ```
9. Удалите структуру UDP объекта из памяти Workspace
   ```
   >> clear u
   ```
10. Восстановите объект следующей командой
    ```
    >> u=instrfindall
    ```
    или
    ```
    >> u = instrfind('Type', 'udp')
    ```
11. Считайте данные
    ```
    >> A=fread(u, 4);
    ```
12. Закройте UDP объект
    ```
    >> fclose (u)
    ```

Задание 2. Построение UDP канала между двумя средами на одном компьютере

1. В <u>первой среде МатЛАБ</u> создайте UDP объект с удаленным портом.
   ```
   >> u=udp('localhost', 9090,'LocalPort', 9091);
   Другие примеры:
   ```
   u=udp('192.168.0.223', 9090); % (not static IP)

u=udp('localhost'); в этом случае номер порта (по умолчанию) - 9090

2. Откройте объект
```
>>fopen(u)
```
3. Создайте UDP объект во второй среде МатЛАБ .
```
>> u=udp('localhost', 9091,'LocalPort', 9090);
```
4. Откройте объект
```
>>fopen(u)
```
5. Проверьте параметры структуры, используя команду >>UDP.
```
 Read/Write State
    TransferStatus:     idle
    BytesAvailable:     0
    ValuesReceived:     0
    ValuesSent:         0
```
6. Из первой среды передайте следующие значения в UDP порт
```
>>fwrite(u, 1:10);
```
7. Проверьте состояние порта во второй среде
```
 Read/Write State
    TransferStatus:     idle
    BytesAvailable:     10
    ValuesReceived:     0
    ValuesSent:         0
```
8. Во второй среде считайте данные из UDP порта
```
>>a = fread (u, 10);
```
9. Проверьте состояние порта во второй среде.
```
  Read/Write State
    TransferStatus:     idle
    BytesAvailable:     0
    ValuesReceived:     10
    ValuesSent:         0
```
10. На принимаемом компьютере запустите циклический опрос состояния UDP объекта. Выполнять побайтное чтение и отображение данных поступающих в UDP объекте.
```
while (true)
    b = get(u, 'BytesAvailable');
    if b > 0
        %disp(sprintf('%d ',fread (u, b)))
        disp(sprintf('%d', str2num (dec2bin (fread (u, 1)))))
    end
end
```
11. На первом компьютере запустите программу формирования SIN сигнала и передачи его в UDP канал.
```
for i = 0:1000;
   fwrite(u, 100 + round (100 * sin(2 * pi * i /100)));
   %pause (0.001);
end
```
Наблюдайте за передачей данных.

12. Остановите программу чтения данных командой Ctrl/c.
13. Убедитесь, что переданные числа с дробной частью принимаются целыми числами.
14. Закройте все UDP объекты
```
 >> fclose (u)
```

Задание 3. Передача данных между двумя компьютерами по udp каналу.

1. Считайте имена компьютеров.

   Start > My Computer > Properties > Computer Name > Full computer name.

   Предположим, имя одного компьютера - 'doetom.dhpc’, а другого - 'doejohn.dhpc'

   'doetom.dhpc’ использует локальный порт 8844, а - 'doejohn.dhpc' – 8866.

   Внимание. Каждый хост взаимодействует с портом другого хоста через удаленный порт RemotePort. Локальный порт принадлежит хосту (компьютеру) на котором создан объект.

2. Создайте UDP объекты на каждом хосте

   u1 = udp('doetom.dhpc, 'RemotePort', 8866, 'LocalPort', 8844) на хосте doejohn.dhpc,

   u2 = udp('doejohn.dhpc', 'RemotePort', 8844, 'LocalPort', 8866) на хосте doejohn.dhpc

3. Соедините объекты. Выполните команды

   fopen(u1) и fopen(u2)

4. Передайте данные

   fprintf(u1, 'Ready for data transfer.')

5. Примите (считайте) данные на другом компьютере

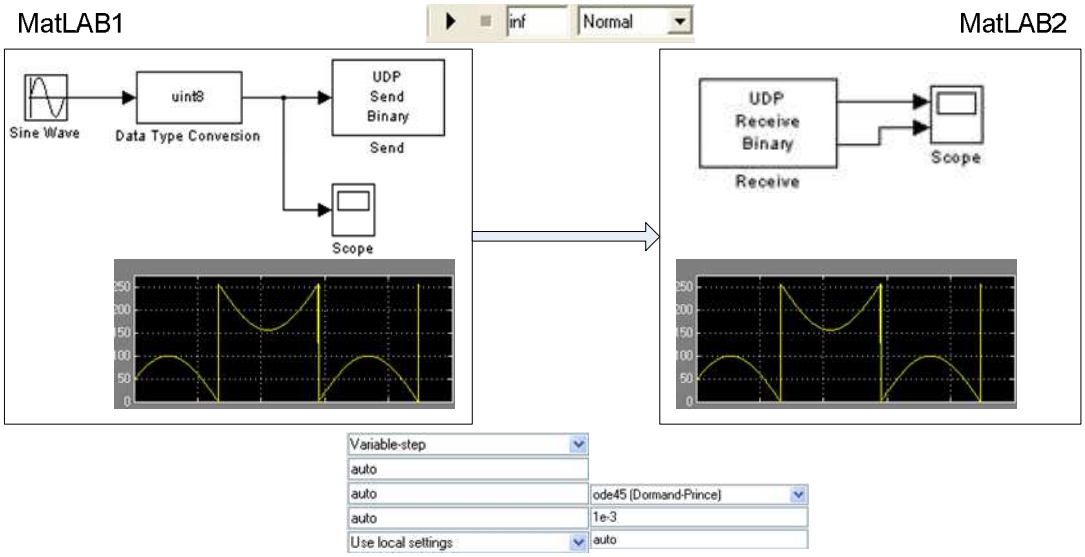
   fscanf(u2)

6. Разъедините и удалите UDP объекты. Пример дан для объекта u1. Повторите действия с объектом u2.
   fclose(u1)
   delete(u1)
   clear u1

Задание 4. Подключение Simulink к UDP каналу

1. Постройте модели генератора и приемника сигналов в Simulink

2. Используя блоки библиотеки xPC Target передайте данные через UDP канал от генератора одной модели приемнику другой модели, например,



Задание 4. UDP соединение “Simulink” - “m-файл”

1. Постройте соединение “Simulink модель” - “m-файл”
2. Запишите принимаемые m-файлом данные в workspace.
3. Сравните переданные и полученные данные.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Где целесообразно использовать UDP соединение?
2. Какова скорость передачи UDP канала?
3. Каков диапазон чисел передаваемых по UDP каналу?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. http://ru.wikipedia.org/wiki/Сетевая_модель_OSI