Freescale MQX RTOS Example Guide

This document explains the Multidrop example, what to expect from the example and a brief introduction to the API used.

MULTIDROP MODE

This mode is a UART’s feature. In the mode, a master can transmit an address character followed by a block of data characters targeted for one of up to 256 slave stations.

When the master sends an address character, the slave receiver notifies its respective CPU then compares the received address to its station address and enables its receiver if it wishes to receive the subsequent data characters or block of data from the master station. Unaddressed slave stations continue monitoring the data stream. Data fields in the data stream are separated by an address character. After a slave receives a block of data, its CPU disables the receiver and repeats the process

IOCTL COMMAND FOR MULTIDROP MODE

- ***IO_IOCTL_SERIAL_SET_PARITY***:
  Depending on parameter, the command is used to configure a master to send address *(IO_SERIAL_PARITY_MULTI_ADDRESS)* or data *(IO_SERIAL_PARITY_MULTI_DATA)* bytes. To configure a slave in multi-drop mode, two parameter values are accepted.
- ***IO_IOCTL_SERIAL_SET_ADDRESS_DEVICE***
  The command is used to set slave station address.
- ***IO_IOCTL_SERIAL_GET_ADDRESS_DEVICE***
  The command is used to get slave station address.

The example

This section describes the multi-drop mode example application. The example shows how to work with the multi-drop and how to use it.

The example consists of two different projects:

- The master project runs the MQX task on the first board. This task will transfer a block of data to slave, it first sends out an address byte to identify the target. Next byte is the size of block data. The remained bytes are data.

- The slave project runs another MQX task and on another boards. Only the target-addressed slave will receive data. The others will ignore the incoming data.

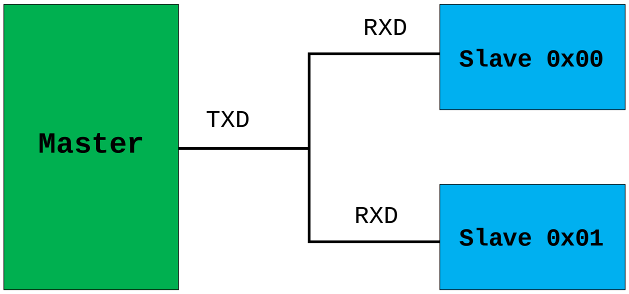
Running the example

A multidrop system contains a master and one or some slave boards. Each time master sends data to only one slave.

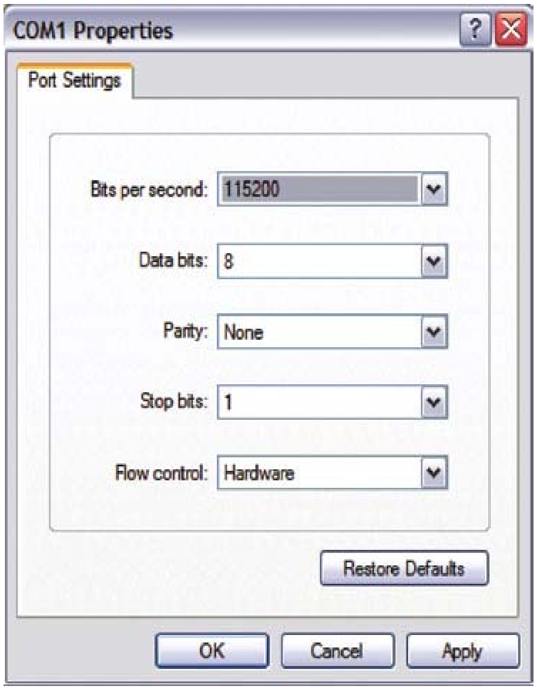
The example uses a master and two slaves.

Before building MQX libraries it is necessary to define BSP_SERIAL_MULTIDROP_DEVICE by enable BSP_ENABLE_TTYx or BSP_ENABLE_ITTYx in user_config.h depend on using board.

Typically, before building project slave it is necessary to redefine SLAVE_ADDRESS in slave.c, as well. Different slaves could not use the same address. In this example, the default SLAVE_ADDRESS is 0x00, and it is used for first slave, so please redefine SLAVE_ADDRESS is 0x01 for second slave.



To display the result, please turn on three terminal applications (for master and two slaves) on your PC and configure them as following: 115200 baud, 8 data bits, 1 stop bit, no parity and no flow control.



Bits per second: 115200

Data bits: 8

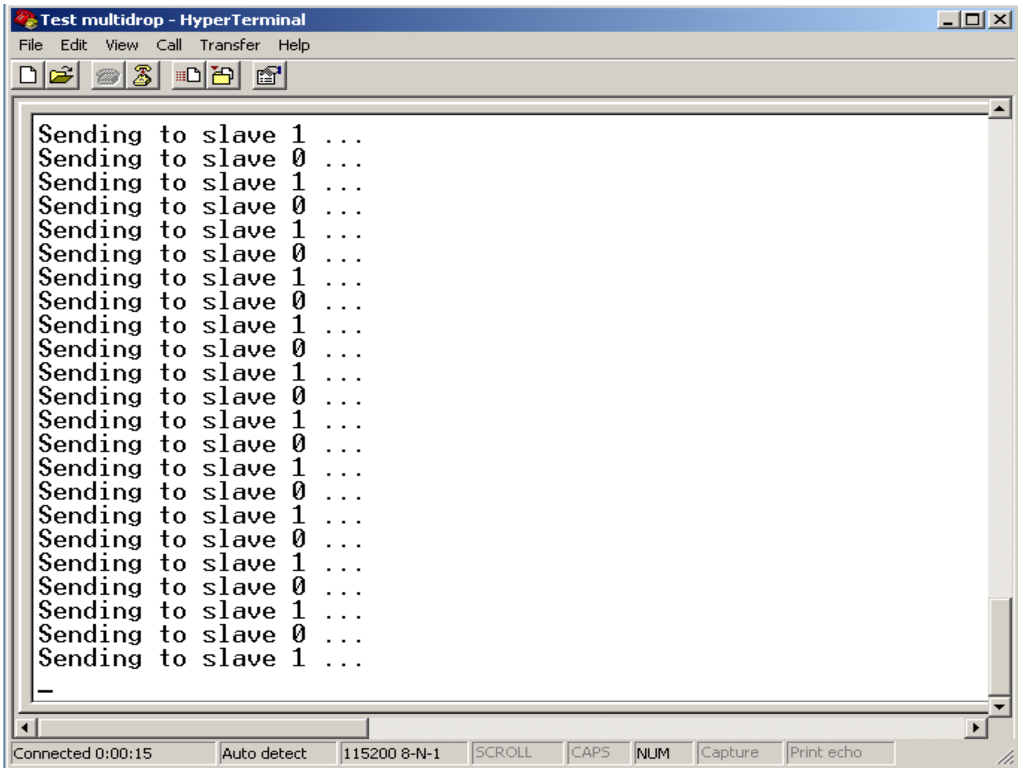
Parity: None

Stop bits: 1

Flow control: Hardware

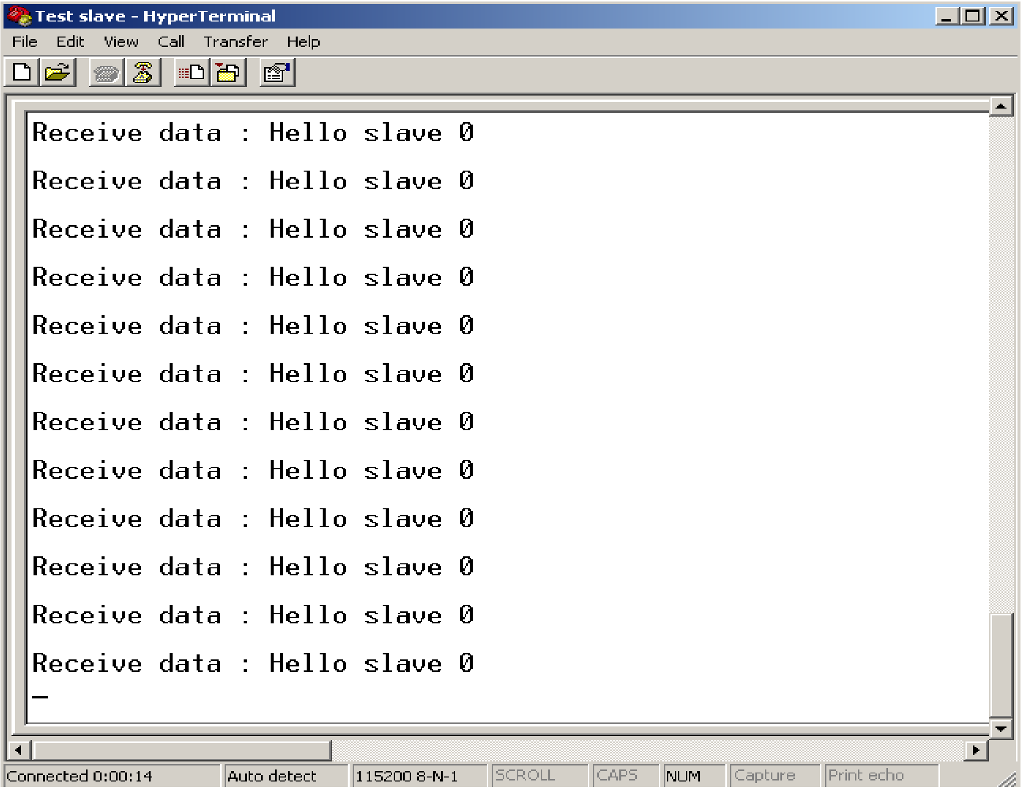
The following outputs can be seen on the serial console for individual boards.

Master board:



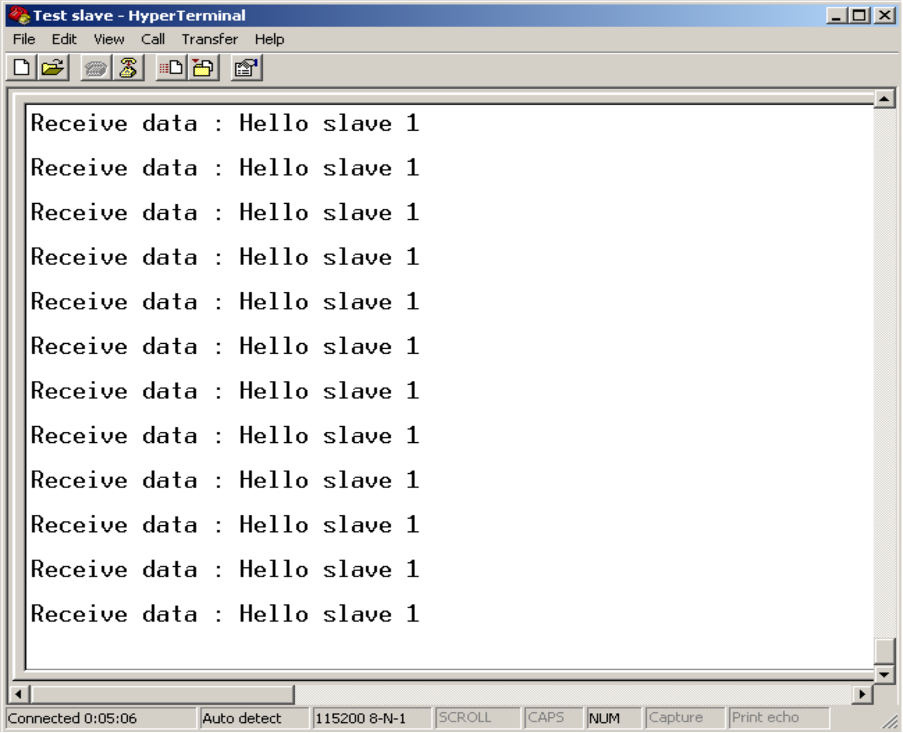
```
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
Sending to slave 0 ...
Sending to slave 1 ...
_
```

The First slave board (with address 0x00):



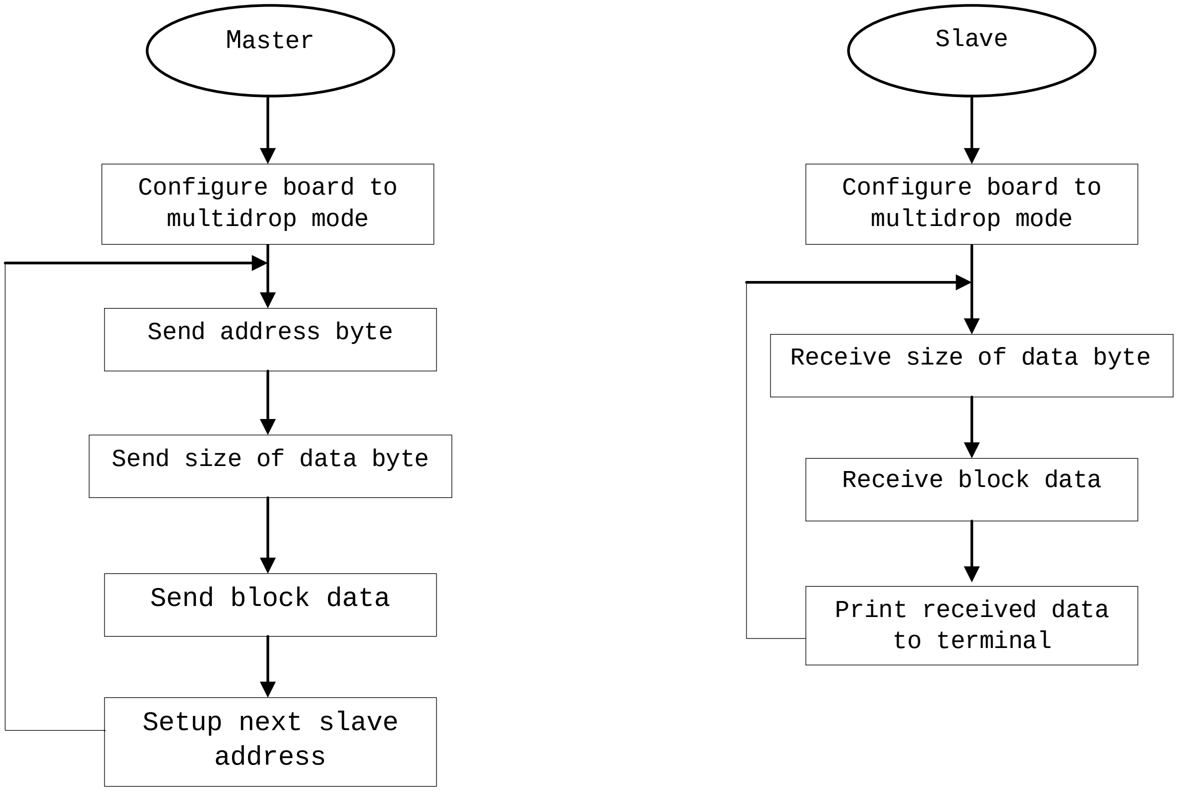
```
Receive data : Hello slave 0

Receive data : Hello slave 0

Receive data : Hello slave 0

Receive data : Hello slave 0

Receive data : Hello slave 0

Receive data : Hello slave 0

Receive data : Hello slave 0

Receive data : Hello slave 0

Receive data : Hello slave 0

Receive data : Hello slave 0

Receive data : Hello slave 0

Receive data : Hello slave 0
_
```

The second slave board (with address 0x01):



```
Receive data : Hello slave 1
Receive data : Hello slave 1
Receive data : Hello slave 1
Receive data : Hello slave 1
Receive data : Hello slave 1
Receive data : Hello slave 1
Receive data : Hello slave 1
Receive data : Hello slave 1
Receive data : Hello slave 1
Receive data : Hello slave 1
Receive data : Hello slave 1
Receive data : Hello slave 1
```

Explanation of the example


Master
Configure board to multidrop mode
Send address byte
Send size of data byte
Send block data
Setup next slave address
Slave
Configure board to multidrop mode
Receive size of data byte
Receive block data
Print received data to terminal