

Bluetooth Serial Port Adapter Optimization

Bluetooth Serial Port Adapter Optimization

For the third version connectBlue serial port adapter products, there are some additional AT commands regarding optimization of the module.

This document describes how to optimize:

- Throughput
- Latency
- Power Consumption
- Range
- Connection & Discovery Time
- WLAN Co-existence

The modules this document is valid for are cB-OEMSPA310i, cB-OEMSPA311i/x, cB-OEMSPA312i/x, cB-OEMSPA331i/x and cB-OEMSPA332i/x.

For details of AT commands described in this documents, see the "Serial Port Adapter AT Commands" specification.

Bluetooth Serial Port Adapter Optimization

Table of content

1 Throughput3

2 Latency.....4

3 Power Consumption5

4 Range6

5 Connection & Discovery Time.....7

6 WLAN Co-existence8

Bluetooth Serial Port Adapter Optimization

1 Throughput

The data transmission throughput between two Bluetooth modules depends very much on the following factors.

- Radio Environment
- Distance between modules
- Selected packet types (DM1, DM3, DM5, DH1, DH3, DH5)
- Application protocol

In DM packets there is a 10-byte forward error correction included and the end number (1,3, or 5) is the packet length in slots. Hence, DH5 packets can contain the most application data since there is no forward error correction and it is the largest packet type consisting of 5 slots.

A good link using the default configuration normally leads to both sides using DH5 packets. This is the packet type that can contain the most application data and therefore it has the highest throughput. Since both sides use 5-slot packets it is also a *symmetrical link*. The theoretical maximum bit rate is then 433,9 kbits/s in each direction. Please note that this includes some protocol header data and the theoretical maximum application throughput will be a little bit lower.

If one side is limited to 1-slot packets, at the same time as the other side is using 5-slot packets the *link will be asymmetrical*. The highest maximum throughput is then 723,2 kbits/s for the side using DH5 packets and 57,6 kbits/s for the side using DH1 packets. Again the theoretical application throughput will be slightly lower since packets over air contain protocol header data.

There are some configuration options available as AT commands to affect the *packet type selection*. Normally (and by default), the module will automatically select packet types using a bit error rate algorithm. For UART baud rates of up to 460,8 kbits/s the default configuration is, more or less, all that is needed. However, for a baud rate of 921,6 kbits/s, the throughput can be increased slightly by forcing the connection to be asymmetrical. This means higher throughput in one direction at the cost of lower in the other. The *Link Policy* AT command can be used to force the receiving side to use only 1-slot packets (AT*AML P=1,0,1).

If the *application protocol* transmits packets with little data, the throughput will go down. The CPU will have to handle many small packets that will increase the CPU load, and there is a lot of Bluetooth protocol header data compared to application data in packets over air. Hence, to get a high throughput make sure that data packets transmitted on the UART are large.

Finally, if the radio link is poor there will be many retransmissions, which will bring down the throughput. Poor radio environment, bad antenna placement, or a long range can cause the link to get poor.

Bluetooth Serial Port Adapter Optimization

2 Latency

The latency consists of several different parts. First data must go through the serial line, then the CPU, over air, CPU on receiving side and finally the serial line on the receiving side.

For a 10 bytes packet that is transmitted using a baud rate of 57,6 kbits/s this typically means ~2 ms on the serial line on the transmitting side and another ~2 ms on the receiving side (not including CPU time and over air). Also note that there is a 3,5 byte receive timeout included to trigger the UART.

A very rough estimate on the CPU time is ~2-4 ms on each side. However, in addition to this, by default, the Bluetooth stack polls for data every 40 slots. Since each slot is 0,625 ms this means an additional delay of 0-25 ms. The *Link Policy* AT command can be used to force a shorter poll time. The Quality of Service (QoS) Bluetooth mechanism is used to change the poll time. It is recommended that the link policy for QoS in combination with DM1 packets be used.

- AT*AMLP=10,0,1 on the master side (QoS for shortest poll and DM1 packets)
- AT*AMLP=9,0,1 on the slave side (DM1 packets)

The second parameter can be used to set a different poll time. Since polling often means higher power consumption it may not be a good idea to poll all the time. It is then possible to set a poll time shorter than the default (40) but not the shortest one (which is 2 for the above case).

Please note that the DM1 packets are used to get a more robust and deterministic behaviour (see Long Range Section). However, it can only contain 8 bytes of application data. That means if the application data packet contains 9 bytes, the delay will roughly be doubled since two packets over air is needed.

If application packets are bigger than 8 bytes a link policy using only QoS and not DM1 packets can be used (AT*AMLP=3,0,1). Since the module then controls packet types itself, a certain packet type is not guaranteed. It is still possible that DM1 packets will be used. However, for a good link it will use DH5 packets.

To make it even more interesting there is also another link policy to select QoS and DM packets (AT*AMLP=12,0,1). This will then be some kind of compromise between the above choices. The module may still select 5 slot packets but it will always use forward error correction, which makes it more robust and deterministic.

Bluetooth Serial Port Adapter Optimization

3 Power Consumption

For many applications it is important that the power consumption is kept low. Both when there is no connection and when there is a connection.

Apart from turning the power off there are a few configuration options to lower the power consumption.

For many applications it is possible to set the module in stop mode, which is the lowest power consumption mode. This is done using the *Power Mode* AT command (AT*AMPM=3,1). There are some restrictions on the behaviour of the module using this mode. Hence, it may not be feasible for all applications. For example, the DSR pin of the module must be activated some ~10 ms before transmitting data and inactivated to enable stop mode. Also, a remote peer may not be enabled with the “always connected” parameter set (AT*ADWDRP).

Furthermore, it is also possible to turn off the LED (if any) when the module enters stop mode. This configuration is set using the *Feature Mask* AT command (AT*AMWFM=1,1,1).

For modules not configured as servers, it is normally possible to make the module both *non-discoverable* (AT*AGDM=1,1) and *non-connectable* (AT*AGCM=1,1). This means that other modules will not find it doing inquiries and they cannot setup connections to that module (which is ok since it is a client only).

To lower the power consumption when there is an active connection, the *Link Policy* AT command can be used. Typically one of the *sniff mode* configurations is used. The longer the sniff period is, the lower the power consumption is. Of course this also means a longer reaction time. For example, a sniff period of 200 ms (AT*AMLPP=7,0,1) can delay a transmission for up to 200 ms depending on when the transmission is made. The second parameter of the *Link Policy* command for sniff configuration has a specific meaning. The default value of 0 means that sniff is always active. If set to 1 (AT*AMLPP=7,1,1), the module exits sniff mode when data needs to be transmitted. Sniff is activated again when there is no data to transmit for 1 second. *Please read the release notes regarding the use of sniff mode.* It is also possible to combine stop mode and sniff mode for the lowest possible power consumption. When doing this, thorough tests must be performed to guarantee robust behaviour.

If the modules are located close to each other it is also possible to decrease the maximum output power for transmission using the *Max Output Power* AT command (AT*AMMP=128,1). By default the value is set to 255, which always means max output power the module can achieve.

Bluetooth Serial Port Adapter Optimization

4 Range

There are some configuration options that may increase the range of operation.

By default configuration, the module automatically controls what packet types to use. Normally it starts with DM1 packets (1 slot, forward error correction) and if the link is good it will start using DH5 (5 slots, no forward error correction) to increase the throughput. Hence, for a poor connection, the module should use DM1 packets. Since DM1 packets are short packets with 10-bytes of forward error correction, the chances to receive (or receive and successfully correct) packets increases. This means less re-transmissions and a more robust link regarding range and poor radio environment. However, the control of packets is quite slow which means that it may use DH5 some time before using DM1 when a link goes from good to bad. Therefore, it may be better to manually configure the module to use only DM1 packets even if the link is temporarily good. This configuration is done using the *Link Policy* AT command (AT*AMLPR=9,0,1). The cost of this configuration is lower maximum throughput. For baud rates up to 115,2 kbits/s it is ok, but for higher baud rates, throughput is not increased.

Also, it is easier to keep a connection that is active than to setup a new connection. Hence, at a certain range it may be possible to send data on an active connection. However, if the connection is lost it may be difficult to set it up again. To setup connections more robustly, the *fast connect* scheme can be used (see Fast Connect & Discovery Section). Tests show that using the *fast connect* configuration (AT*AMWFM=1,2,1 on server) not only makes faster connections but also more robust connections. Hence, it will be easier to setup connections when the range is long or the radio environment is poor.

Bluetooth Serial Port Adapter Optimization

5 Connection & Discovery Time

In Bluetooth v2.0 there is something called interlaced page and inquiry scan. Basically this means that the module listens on more frequencies to quicker detect if a remote device is trying to setup a connection to it (page scan) and/or if a remote device is searching for Bluetooth devices (inquiry scan). Hence, there is nothing changed on the side that makes the connection attempt (page) or the inquiry.

To configure the module for *fast connect* (interlaces page scan) and/or *fast discovery* (interlaces inquiry scan), the *Feature Mask* AT command is used (AT*AMWFM=1,6,1). Please note that typically this is done on the server side and not client side. The cost is a slight increase in power consumption.

Bluetooth Serial Port Adapter Optimization

6 WLAN Co-existence

First of all there is the *Adaptive Frequency Hopping* (AFH) algorithm that is active by default. This means that the Bluetooth module will automatically remove frequencies that are blocked (by e.g. a WLAN network). Typically, this means that the frequencies that are used by the WLAN network will not be used. Normally it works very well. However, there are cases where this is not enough.

Problems with the AFH algorithm

- The AFH algorithm takes a few seconds before getting active. This means that for a few seconds after a Bluetooth connection is setup, the WLAN network may be affected.
- Frequencies previously removed are tested again after 30 seconds. Hence, if there is no traffic on the WLAN network for 30 seconds, the removed frequencies are added again which means that the WLAN network may again be affected (at least for a few seconds).
- The AFH algorithm is only used on an active Bluetooth connection. This means that for inquiries and connection attempts it is not used. Therefore, inquiries and connection attempts may still affect the WLAN network.

There are some configuration options available to handle some of the above problems.

There is a *Channel Map* AT command to manually exclude frequencies (AT*AMCM) that are used by the Bluetooth module. Hence, if the frequencies of the WLAN network are known (which is normally the case), these frequencies can be excluded using the channel map command and the Bluetooth module will not use them once the connection is active. The channel map then replaces the AFH algorithm. However, inquiries and connection attempts still uses all frequencies as for the AFH algorithm.

When a remote peer is defined (to setup a connection) it is possible to change the *connection timeout* (page timeout, default 5 seconds) to a much shorter time. We have made tests with as short timeout as 80 ms. The client will then only try to connect for 80 ms before giving up. Since a normal Bluetooth connection normally takes ~2 seconds this would fail unless the server is configured for *fast connect* (See Connection & Discovery Time section). During our tests, a client configured with a page timeout of 80 ms will succeed most of the times setting up a connection to a server configured for *fast connects*. Of course it will fail every now and again and this must be handled by the application. If using the "always connected" option for the *remote peer* AT command (AT*ADWDRP) it is also possible to change the default connection period from 10 seconds to a much lower one. A client that failed to setup a connection will then try again after a much shorter time. We have run tests with the client page timeout to 80 ms and the "always connected" period to 3 seconds. The affect on the WLAN network is then hardly measurable.

For the remaining problem with inquiries there are no really good solutions. Best is if you can avoid inquiries or at least decrease the maximum output power of the module (AT*AMMP) to not disturb the WLAN network.

Bluetooth Serial Port Adapter Optimization

It is possible that a longer distance between the Bluetooth module and the WLAN module is enough to remove any interference between the two technologies and the configuration options described above might then not be needed.