



## **HxM Bluetooth API Guide**

## Document History

Version	Date	Description
1.1	16-05-2008	Initial version
1.2	14-07-2008	Added requirement for a security passkey to the security section.
1.3	28-08-2008	Added comms. packet information
1.4	19-01-2009	Added HyperTerminal Hardware Check
1.5	19-05-2010	Remove cadence
1.6	31-05-2010	Correct stride rollover value to 128
1.7	22-07-2010	Add Packet Logger Utility

## Document Notes

All numbers in this document are written in decimal, except hexadecimal numbers which are prefixed by '0x'. For example 5436 is decimal, while 0x5436 is hexadecimal.

## Contents

1.	REFERENCES.....	4
2.	ABBREVIATIONS.....	4
3.	INTRODUCTION.....	4
4.	SYSTEM OVERVIEW.....	5
4.1.	System Architecture .....	5
4.2.	Connecting to HxM Device .....	5
5.	HXM COMMUNICATIONS LINK.....	8
6.	COMMUNICATIONS PACKET STRUCTURE OVERVIEW .....	9
6.1.	Basic Message Format.....	9
6.1.1.	STX .....	9
6.1.2.	Msg ID .....	9
6.1.3.	DLC .....	9
6.1.4.	Data Payload .....	9
6.1.5.	CRC.....	10
6.1.6.	ETX .....	10
7.	STANDARD HXM DATA MESSAGE .....	11
7.1.	MSG:0x26 – Heart Rate, Speed & Distance packet .....	11
7.1.1.	Packing Format.....	12
8.	HARDWARE CHECK USING HYPER TERMINAL.....	14
9.	HXM PACKET LOGGER UTILITY .....	18
9.1.	Installation.....	18
9.1.1.	.NET 3.5 SP1.....	18
9.1.2.	HxM Packet Logger.....	18
9.2.	Operation.....	19
9.2.1.	RR Log data.....	21
9.2.2.	1 Hz Log Data.....	21

## 1. References

Ref #	ID	Description
[1]	9700.0028c	General Comms Link Specification.

## 2. Abbreviations

Abbreviation	Description
API	Application Programmer's Interface
ECCG	ElectroCardioGram
HRM	Heart Rate Monitor
HxM	Heart Rate / Speed and Distance Monitor
PC	Personal Computer
RTC	Real Time Clock
SPP	Serial Port Profile

## 3. Introduction

This document is intended as a user's guide for implementing an API to enable communications with a Zephyr Technology Bluetooth Heart Rate / Speed & Distance Monitor (HxM)

## 4. System Overview

### 4.1. System Architecture

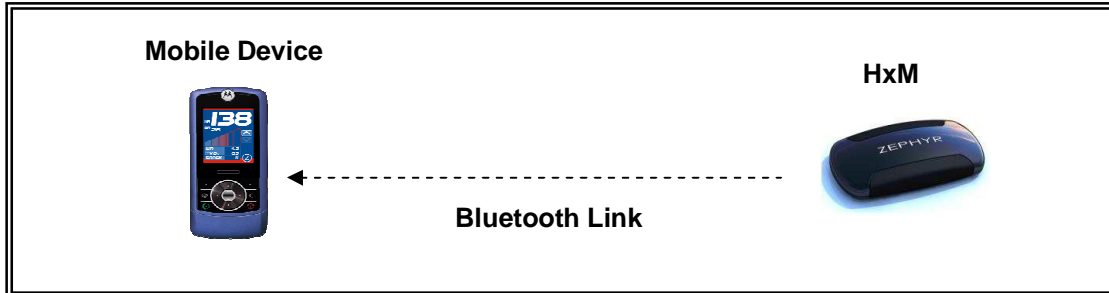


Figure 1 : HxM System Architecture

The diagram above shows that the Bluetooth HxM typically communicates with a mobile device over the Bluetooth link. The HxM only supports one link at a time and uses the Bluetooth **SPP** (Serial Port Profile) to communicate with other devices with the following low-level protocol:

- 115,200 baud
- 8 data bits
- 1 stop bit
- No parity

### 4.2. Connecting to HxM Device

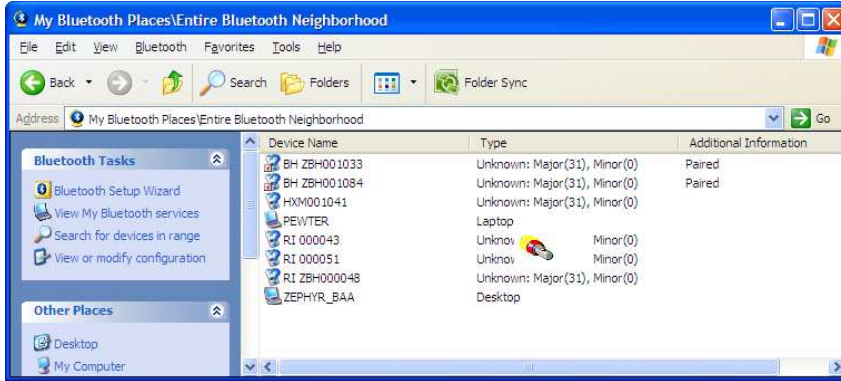
The device is discoverable. The discoverable name of the device is HXMXXXXXX where XXXXXX is the programmed serial number of the HxM device.

When the HxM device is in a connected state it is not discoverable.

The following steps have to be undertaken to connect to a HxM device.

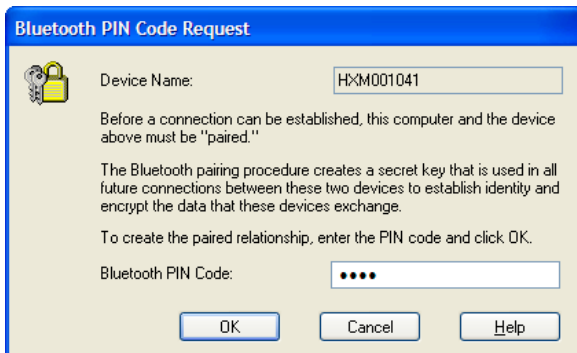
- 1) Activate the Bluetooth service of the device/computer wanting to connect to the HxM
- 2) Scan for Bluetooth devices in range
- 3) Pair with the HxM device found in range
- 4) Discover Services of Paired HxM
- 5) Connect to serial port of HxM device

The following screenshots show the above process with the standard Microsoft Bluetooth driver and Microsoft Windows XP Operating System.

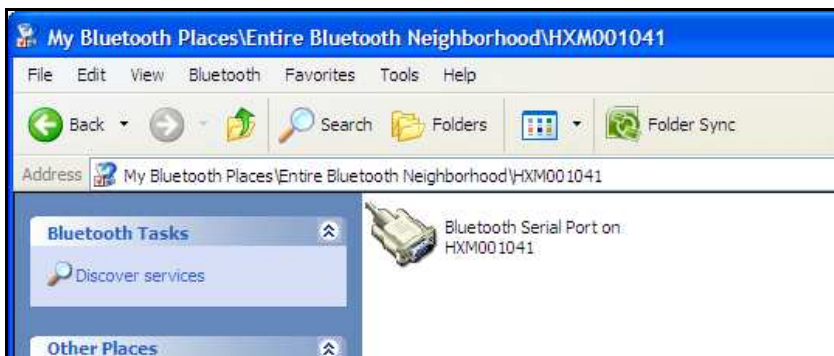


**Figure 2 : Scan for Bluetooth devices in range**

The device is discoverable. The discoverable name of the device is its serial number. To connect to the device a passkey must be used. The passkey is "1234", and cannot be changed.



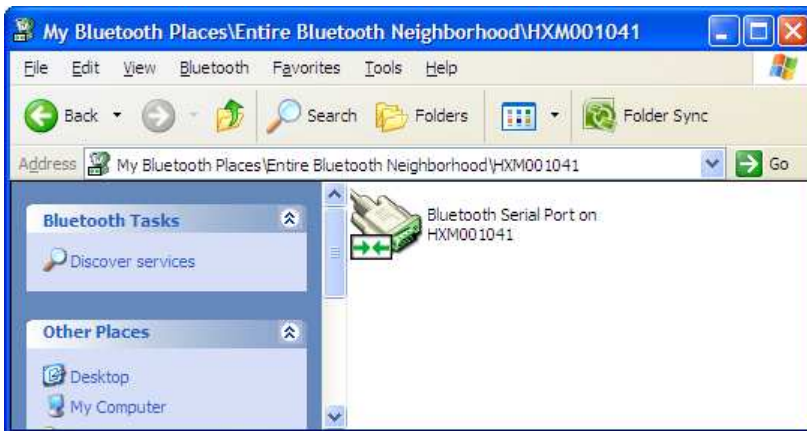
**Figure 3 : Pair with HxM and provide PIN**



**Figure 4 : Discover services of HxM device**



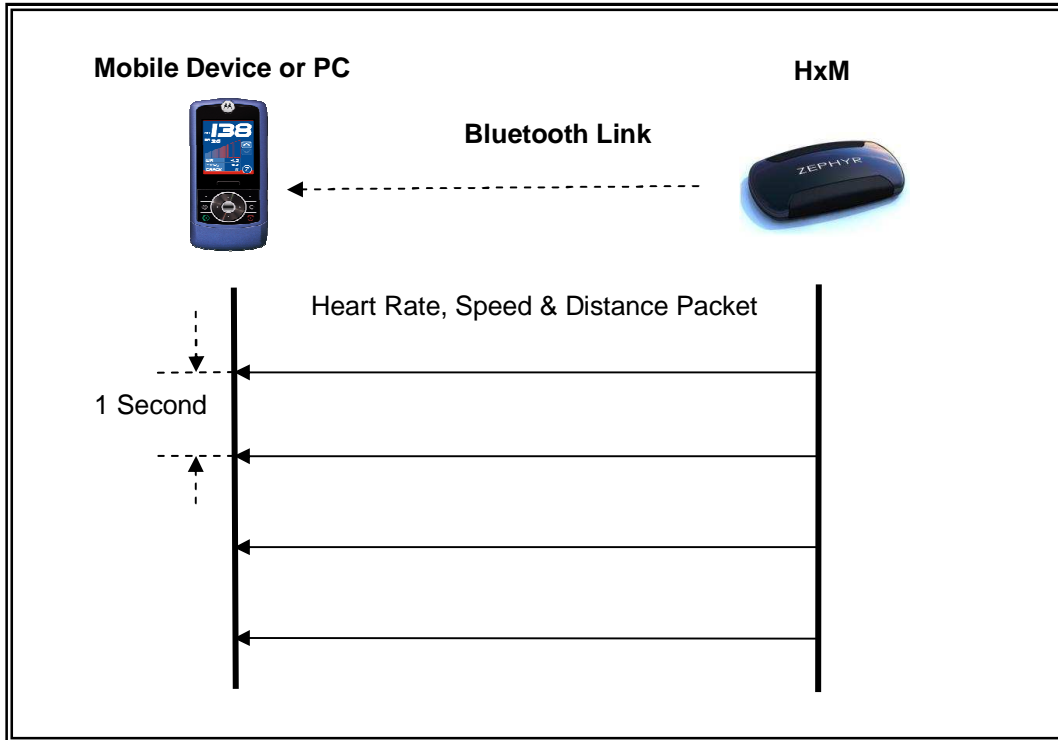
**Figure 5 : Connect to Bluetooth HxM serial port**



**Figure 6 : Connected to Bluetooth HxM serial port**

## 5. HxM Communications Link

Once a Bluetooth connection has been established with the HxM, the data packets can be received by the connecting device. The following sequence diagram shows an example session:



**Figure 7 : Example link session**

The diagram shows that after the mobile device has connected to the HRM it receives a data packet from the HxM at 1 second intervals.

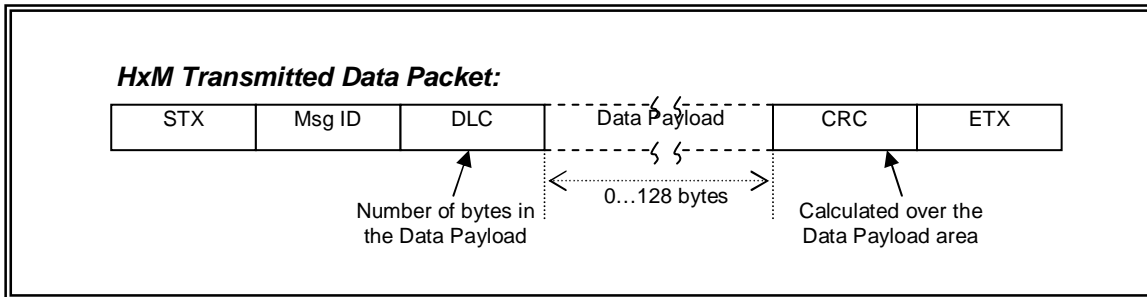
The Heart Rate, Speed & Distance message cannot be disabled. The device will not respond to or acknowledge.



## 6. Communications Packet Structure Overview

The serial communication between the HxM device and the paired device operates in a simplex mode. The HxM device transmits the HxM data packet and does not process any received data.

### 6.1. Basic Message Format



**Figure 8 : Basic Message Format**

#### 6.1.1. STX

The STX field (Start of Text) is a standard ASCII control character (0x02) and denotes the start of the message. Although the protocol does not guarantee this value will not appear again within a message, it gives some delimiting to the message and therefore a start character to search for when receiving data.

#### 6.1.2. Msg ID

The Message ID uniquely identifies each message type and is in binary format. For the HxM the message ID is 0x26 for the standard HxM data packet.

#### 6.1.3. DLC

The Data Length Code is used to specify the number of bytes within the data payload field of the message. Valid values range between zero and 128 (inclusive). For the standard HxM data packet the DLC has a value of 0x37.

#### 6.1.4. Data Payload

This field contains the actual data sent between the local and remote units and can contain anywhere between zero and 128 bytes of data. The number of bytes in this field is dictated by the DLC field.

## 6.1.5. CRC

An 8-bit CRC is used and has a polynomial of 0x8C (CRC-8), with the accumulator being initialised to zero before the CRC calculation begins. The following source code indicates the required implementation:

**crc:** The current CRC.

**Ch:** Value to add to the CRC calculation.

```
Void crc8PushByte(uint8_t *crc, uint8_t ch)
{
    uint8_t i;

    *crc = *crc ^ ch;
    for (i=0; i<8; i++)
    {
        if (*crc & 1)
        {
            *crc = (*crc >> 1) ^0x8C;
        }
        else
        {
            *crc = (*crc >> 1);
        }
    }
}
```

**pcrc:** Pointer to running CRC to update (set this to something before first call).

**Block:** Pointer to the block of data to push through.

**Count:** The number of bytes to push.

**Return:** The computed CRC (result also updated in pcrc if that is non-NULL).

```
uint8_t crc8PushBlock(uint8_t *pcrc, uint8_t *block, uint16_t count)
{
    uint8_t crc = pcrc ? *pcrc : 0;

    for (; count>0; --count,block++)
    {
        crc8PushByte(&crc, *block);
    }
    if (pcrc) *pcrc = crc;
    return crc;
}
```

## 6.1.6. ETX

The ETX field (End of Text) is a standard ASCII control character (0x03) and denotes the end of the message.

## 7. Standard HxM Data Message

### 7.1. MSG:0x26 – Heart Rate, Speed & Distance packet

This message contains the heart rate data, including the last 15 RR timestamps, and speed & distance data. The packet is transmitted periodically at 1Hz.

Byte/Bit	7	6	5	4	3	2	1	0	Field
0	STX								STX
1	0x26								Msg ID
2	55								DLC
3	Firmware ID								Payload
5	Firmware Version								
7	Hardware ID								
9	Hardware Version								
11	Battery Charge Indicator								
12	Heart Rate								
13	Heart Beat Number								
14	Heart Beat Timestamp #1 (Newest)								
16	Heart Beat Timestamp #2								
18	Heart Beat Timestamp #3								
20	Heart Beat Timestamp #4								
22	Heart Beat Timestamp #5								
24	Heart Beat Timestamp #6								
26	Heart Beat Timestamp #7								
28	Heart Beat Timestamp #8								
30	Heart Beat Timestamp #9								
32	Heart Beat Timestamp #10								
34	Heart Beat Timestamp #11								
36	Heart Beat Timestamp #12								
38	Heart Beat Timestamp #13								
40	Heart Beat Timestamp #14								
42	Heart Beat Timestamp #15 (Oldest)								
44	Reserved								
46	Reserved								
48	Reserved								
50	Distance								
52	Instantaneous speed								
54	Strides								
55	Reserved								
56	Reserved								
58	CRC								CRC
59	ETX								ETX

Table 7.1 *Message: Heart Rate, Speed & Distance Packet*

## 7.1.1. Packing Format

All bytes are little endian.

All 16 bit unsigned integers are sent least significant byte first.

### *Firmware ID & Version*

Zephyr Firmware releases are identified with identification codes of the following form “9500.xxxx.Vyz” where xxxx is a number identifying the firmware type (0000 - 9999), and y is the major version (1 – 9) and z is the minor version (A – Z, or a-z).

The firmware ID field represents xxxx, and the firmware version field is two ASCII bytes representing yz.

### *Hardware ID & Version*

Zephyr Hardware releases are identified with identification codes of the following form “9800.xxxx.Vyz” where xxxx is a number identifying the product (0000 - 9999), and y is the major version (1 – 9) and z is the minor version (A – Z, or a-z).

The hardware ID field represents xxxx, and the hardware version field is two ASCII bytes representing yz.

### *Battery Charge Indicator*

The battery charge indicator is an unsigned byte, representing the percentage of charge remaining. The valid range is 0 to 100%.

The unit will continue to run for a short time once the charge indication has reached 0%.

### *Heart Rate*

The heart rate is an unsigned byte, representing the heart rate in beats per minute.

The valid range is 30 to 240bpm, but the value will go to zero if no valid heart beat is detected within the timeout period.

### *Heart Beat Number*

The heart beat number is an unsigned byte, which is incremented each time a heart beat event is detected.

The valid range is 0 – 255. The count rolls over after 255.

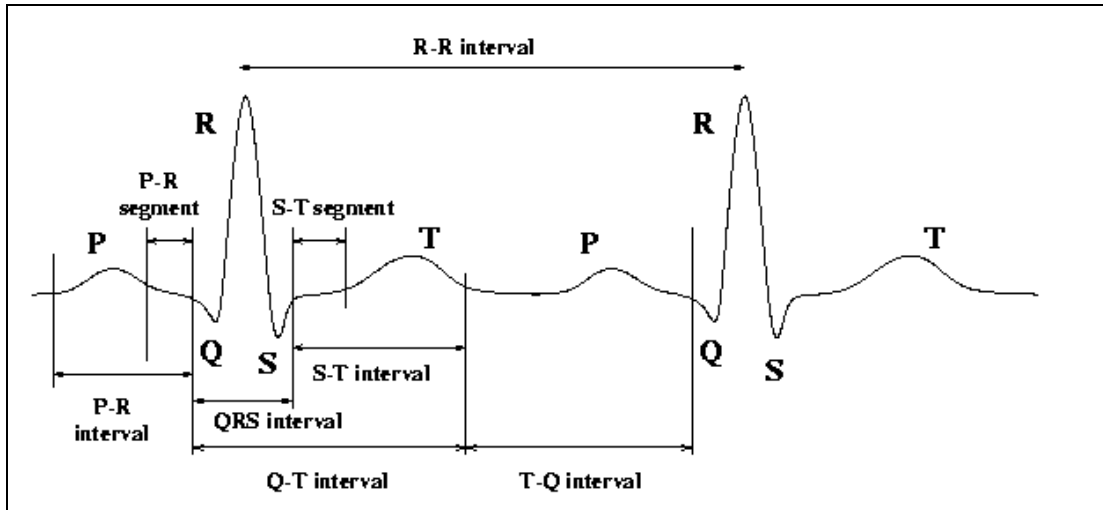
This enables the receiver to determine how many new heart beat timestamps are present in the received packet, even in the event of a dropped packet.

## Heart Beat Timestamp (1 to 15)

The heart beat timestamp is a 16 bit unsigned integer, representing the time at which the heart beat occurred, in milliseconds. (Valid range 0 – 65535ms)

The time used to populate the timestamp rolls over after 65535ms. This should be taken into consideration when calculating the heart beat interval (RR time).

The R to R data consists of a series of R-detection timestamps with 1 millisecond resolution. The internal clock rolls over at 65535 ms.



**Figure 9 : PQRST Waveform showing the R-R Interval.**

The R to R interval is the time between subsequent R peaks in an ECG waveform.

15 values are sent, such that at the fastest valid heart rate (240bpm) 3 consecutive packets can be dropped without loss of information. Timestamp #1 is the most recent heart beat and timestamp #15 is the oldest.

### *Distance*

The distance is a 16 bit unsigned integer, representing the cumulative distance travelled since the device was powered on, in 16ths of a meter (valid range 0 to 4095).

The distance travelled rolls over every 256m.

### *Instantaneous speed*

The Instantaneous speed is a 16 bit unsigned integer, representing the Instantaneous speed of the wearer in steps of 1/256m/s. The valid range is 0 – 4095 steps or 0 to 15.996m/s.

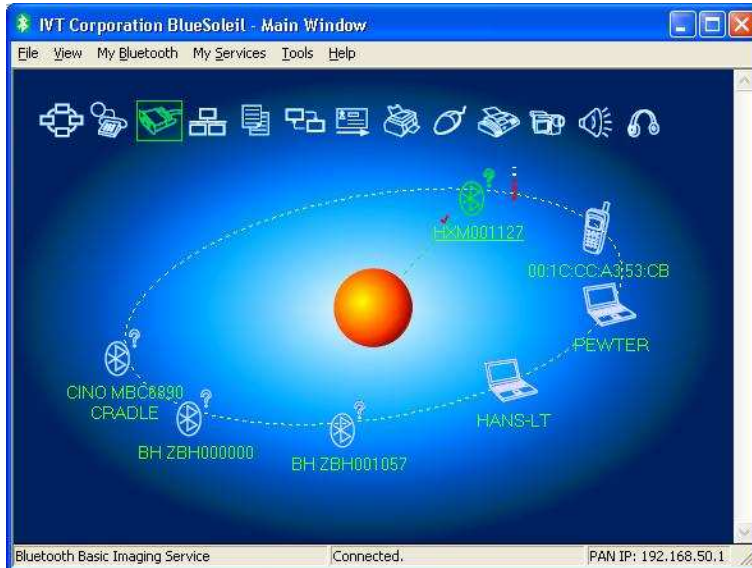
### *Strides*

The strides count is an unsigned byte, representing the number of strides since the unit was powered on. The valid range is 0 to 255 strides. The number of strides rolls over every 128 strides.

## 8. Hardware Check Using Hyper Terminal

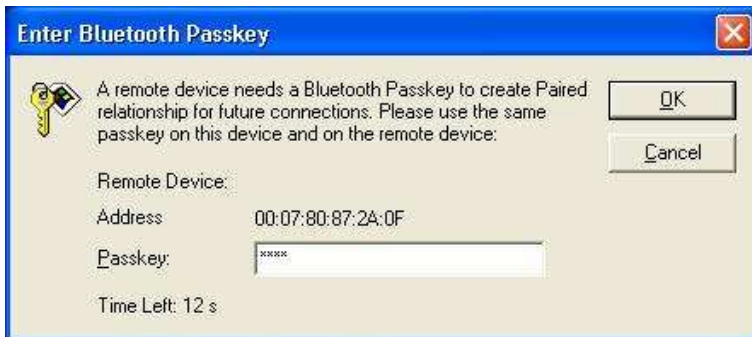
The following steps can be used to confirm output from the device using Hyper Terminal. In this example, the Bluetooth Utility used is IVT BlueSoleil in a desktop PC in conjunction with a USB Bluetooth dongle.

1. Start your Bluetooth utility and establish a connection with the HxM

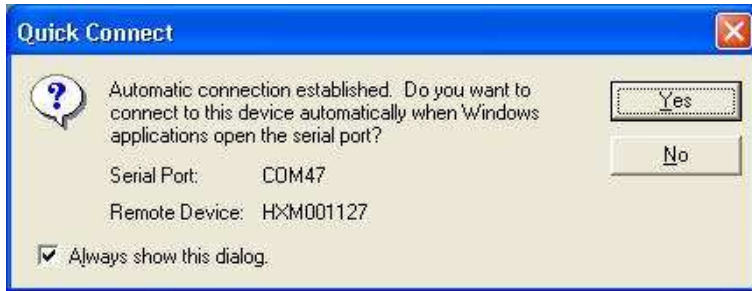


In the example above the device is HxM001127

2. Establish a Connection with the device using Passkey '1234'

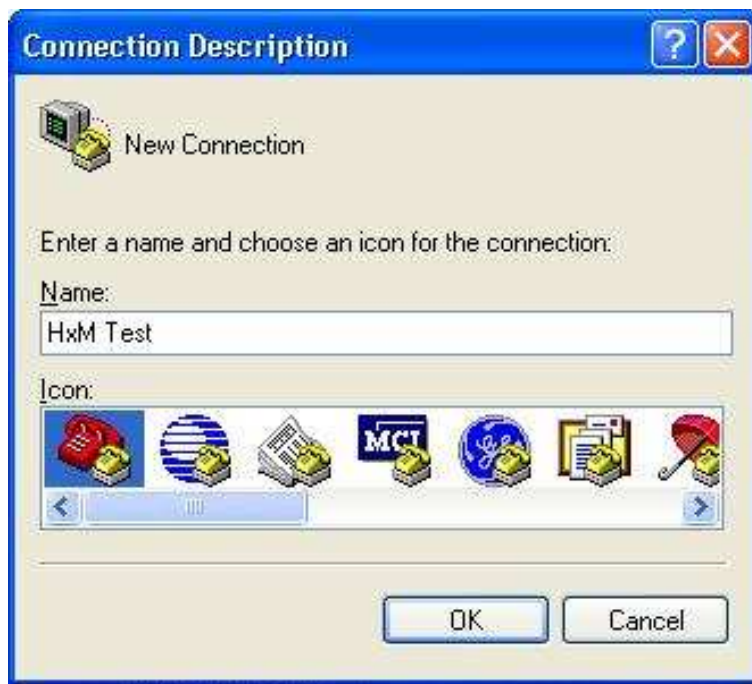


3. Determine which COM Port is being used:



The Serial Port Service is COM47 in this example

Use *Windows > Start > All Programs > Accessories > Communications > Hyper Terminal*

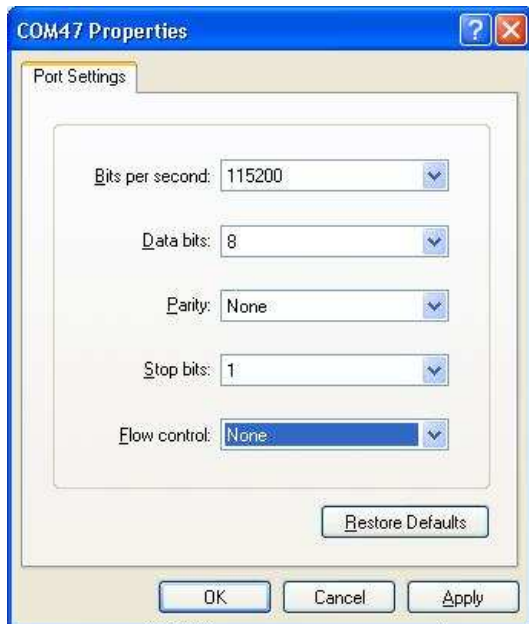


Enter 'HxM Test' or similar as a New Connection name.

4. Select the COM Port from the pulldown list and click **OK**

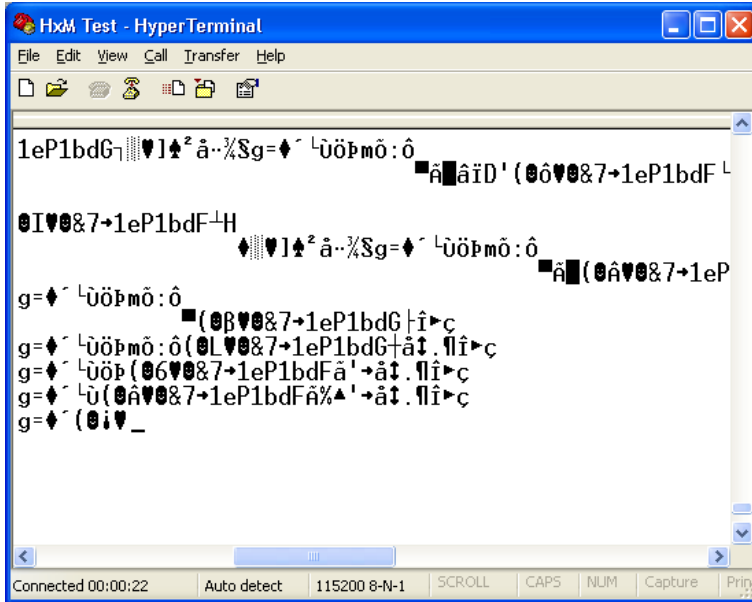


5. Set the Connection Properties as shown





6. The Hyper Terminal Window should now show output from the device



Note that the device outputs binary data. Hyper Terminal displays this as ASCII.

Notes:

1. The device has wear-detect circuitry which switches on data output. If no data is observed, moisten the sensors pads with water.

Excessively dry skin may not have sufficient conductivity to turn the device on.

## 9. HxM Packet Logger Utility

A PC Tool is included as part of the SDK to allow you to receive and record data from a PC with a Bluetooth Interface. This can be a native PC Bluetooth card, or a Bluetooth USB dongle.

### 9.1. *Installation*

#### 9.1.1. .NET 3.5 SP1

This utilities requires that Microsoft® .NET3.5SP1 be installed on your PC for it to operate (this framework is embed in Windows 7).

To install, browse the CD to locate

***.. \BT HxM SDK\Microsoft\DotNetFX35SP1\dotNetFx35setup.exe***

And double-click to install the software. Follow the Microsoft wizard instructions.

#### 9.1.2. HxM Packet Logger

To install, browse the CD and locate

***.. \BT HxM SDK\BT HxM Packet Logger\setup.exe***

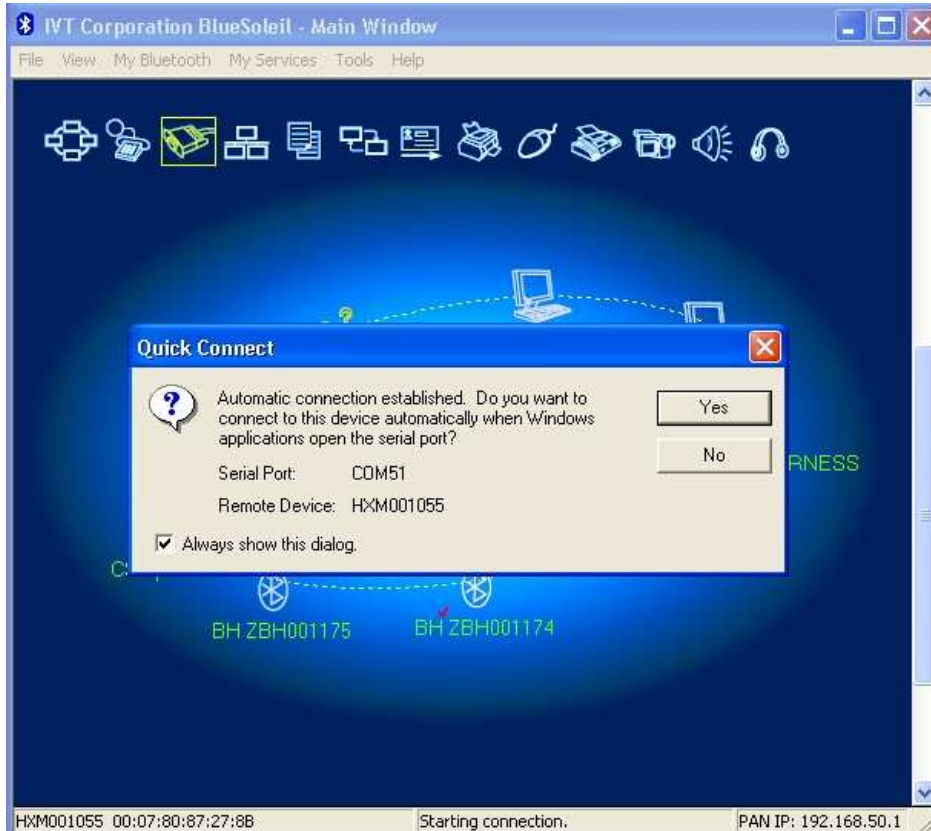
And follow the installation wizard.

A desktop shortcut will be installed.

## 9.2. Operation

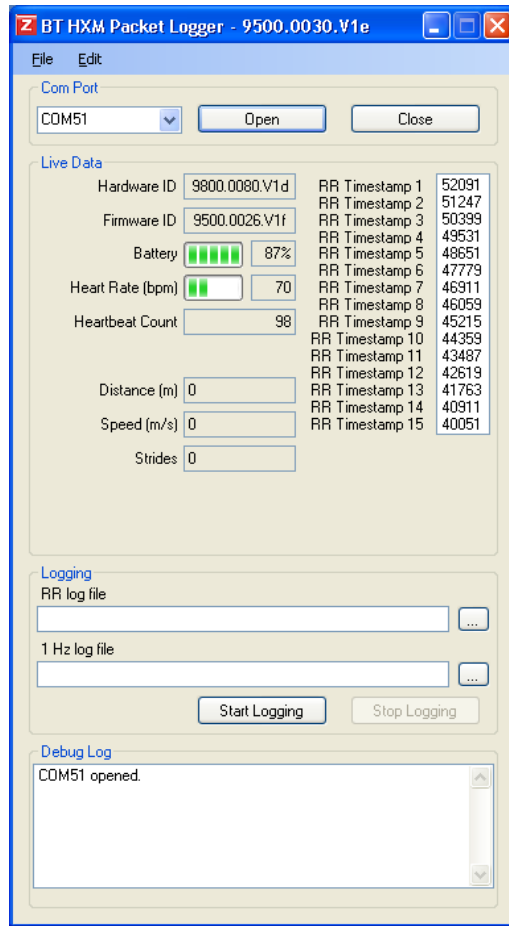
In order to use the Packet Logger, you must determine which COM Port is receiving the HxM data. Wear the device and moisten the sensor pads to power it on.


Use the Bluetooth Interface of your PC, or that supplied with your Bluetooth USB dongle, to display the Properties of the connection to the device:

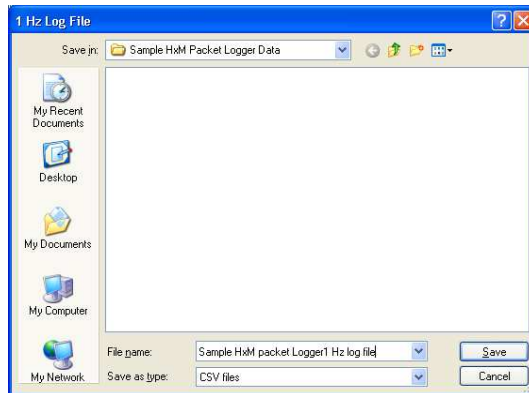
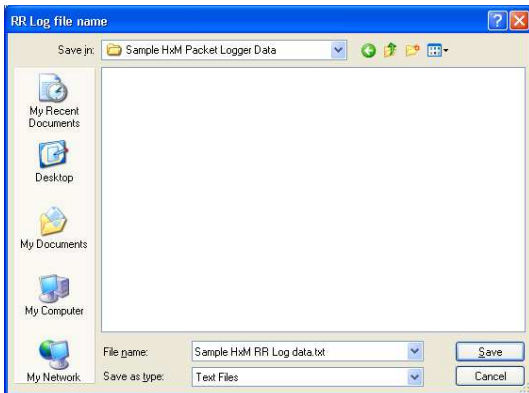


Note the COM Port number which has been allocated to the connection.

Start the Packet Logger utility, enter the COM Port number, and click *Open* to start the tool:



To record data, click the browse buttons , find a suitable location and enter the name of the file you want recorded. Do this for both the RR Log (text) file and the 1Hz Log (.csv) file



Click the *Start Logging* button to create the log files. Click *Stop Logging* when required.

Note: Each time the *Start Logging* button is clicked subsequently, the utility will overwrite any data already in the file displayed in the log file path. A new file must be created prior to clicking the *Start Logging* button if you don't want to over-write data already recorded.

## 9.2.1. RR Log data

The text file contains heart rate RR intervals, measured in milliseconds. There is no timestamp information.

## 9.2.2. 1 Hz Log Data

The .csv file contains columns corresponding to lines 3 – 56 in Table 7.1