ANT+ Device Profile

HEART RATE MONITOR

ANT+ Managed Network Document
D00000693 Rev 1.13
Dynastream Innovations Inc.

P +1 403.932.9292 F +1 403.932.6521
228 River Avenue, Cochrane, Alberta, Canada T4C 2C1
thisisant.com

Revision History

Revision	Effective Date	Description
1.0	Feb 14, 2006	Created
1.02	Feb 28, 2006	Mark Reserved Fields
1.03	Oct 6, 2006	Modified Title of the document
1.04	Nov 27, 2006	Added future page note
1.05	Jan 15, 2007	Added measurement count value to data packet
1.06	Oct 2, 2007	Clarified page identification
1.07	Nov 15, 2007	Updated format, refined page definitions
1.10	May 2, 2008	Official Customer Release
1.11	July 17, 2008	Added data pages 1 - 4
1.12	Aug 27, 2008	Tech writer update/edit
1.13	Jan 28, 2011	Edited "Copyright Information and Usage Notice" section

Table of Contents

List of Tables

List of Figures

1 Overview of ANT+

The ANT+ managed networks are designed to promote interoperability of a new generation of sports and health devices. They combine the global 2.4GHz ANT RF products with common branding and an interface specification to allow manufacturers to build interoperable communicating sports and health equipment.

Interoperating equipment includes both sensors and receiver/display devices.

Examples of sensors include:

1. Speed and Distance Monitors
2. Heart Rate Monitors
3. Bike Speed Sensors
4. Bike Cadence Sensors
5. Combined Bike Speed and Cadence Sensors
6. Bike Power Sensors
7. Bike Component Sensors
8. Environment Sensors
9. Weight Scales
10. Fitness Equipment (Treadmills, Bike Trainers, Elliptical Trainers, etc)

Examples of receiver/display devices include:

1. Watches and Wrist-top Computers
2. Bike Computers
3. Cell Phones / PDAs
4. Fitness Equipment (Treadmills, Bike Trainers, Elliptical Trainers, etc)

ANT+ sensors use ANT 2.4GHz low power communication to transmit their data to the remote display devices. Sensor messages are embedded in ANT serial communications packets and have appropriate header and checksum information in the link layer.

The ANT+ sensor is designed to interface to an ANT receiver, which is embedded in the device that wishes to receive this sensor data. The information provided in this document assumes the user has knowledge of the ANT protocol, and is intended to be used in conjunction with the ANT Message Protocol and Usage document.

This document details the wireless communication link between the sensor and the receiving display device. The ANT+ sensor's typical use case, channel configuration, data message format, and implementation guidelines are detailed.

1.1 Interoperability

Interoperability of sensors within the ANT+ managed network is of paramount importance.

IMPORTANT: To have received this document you have agreed to and signed the ANT+ Managed Network license agreement and have received the ANT+ Managed Network key. By signing this agreement and receiving the ANT+ device profiles you are agreeing to implement and test your product to this specification in its entirety. You are also agreeing to implement only ANT+ defined messages on the ANT+ managed network. This is essential to maintaining interoperability of all devices on the ANT+ managed network.

2 Related Documents

Refer to current versions of the listed documents. To ensure you are using the current versions, check the website or contact your ANT+ representative.

1. ANT Message Protocol and Usage
2. ANT Reference Design User Manual

3 Typical Use Case of a Heart Rate Monitor

A heart rate monitor is a body-worn device that allows the wearer to measure his or her heart rate in real-time. Most heart rate monitors are worn around the chest and transmit heart rate data to watches or other display devices. However other types of monitors such as finger sensors, earlobe sensors, or the hand contact sensors on fitness machines may also use this device profile to transmit heart rate data. Similarly, the receiver need not be a watch but may be a cell phone, piece of fitness equipment, activity monitor, or other personal display device.

Figure 1 below illustrates how the heart rate monitor is typically used. The monitor transmits the user's heart rate information in the main data pages. Some device-specific information is transmitted at a slower rate in the background data pages.

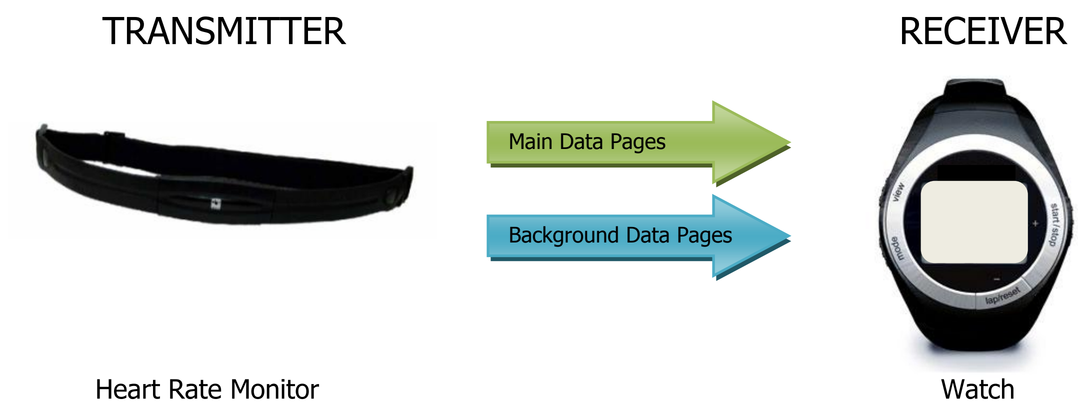

Figure 1: Standard Use Case of an ANT+ Heart Rate Monitor

3.1 Messages Transmitted from the ANT+ Heart Rate Monitor

It is important to note that all of the ANT+ heart rate monitor's data pages — main data pages and background data pages — transmit the user's current heart rate, current heart beat count, and the most recent heart beat event time. This message format ensures that the most recent critical heart rate information is sent with every message. This format also ensures that new heart rate monitors and receivers are backwards compatible with existing ANT+ heart rate monitors and receivers, which adhere to older versions of the ANT+ specifications.

The heart rate monitor is able to transmit two different main data pages. Main data pages are sent at a rate of approximately 4Hz. The choice of main data page and how often each main data page is sent are selected by the manufacturer but must meet the 4Hz data transmission rate requirement. Refer to section 5.2.2 for a more detailed description of the main data pages.

There are three different background data pages that the heart rate monitor can transmit. Two of these pages are manufacturer information and are required to be sent by each heart rate monitor. A background page is sent every 65th message. A further discussion of background data pages is found in section 5.2.1.

Figure 2 shows the different main data pages and background data pages that can be sent from the heart rate monitor. This figure highlights how each data page contains the most recent heart rate information, shown by the box at the start of each arrow.

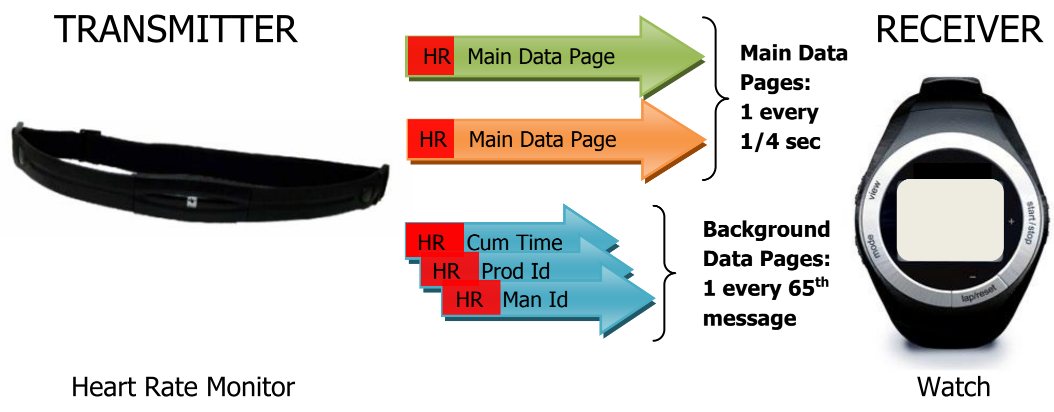

Figure 2: Timing of ANT+ Heart Rate Message Formats

3.2 ANT+ Heart Rate Monitor Receiver Implementation

It is important that the receiver is able to decode all of the data pages that can be sent from an ANT+ heart rate monitor. This document gives code examples showing how to implement code so that the receiver will be compatible with all types of ANT+ heart rate monitors regardless of the data pages a specific implementation of a heat rate monitor supports. The ANT+ heart rate monitor receiver shall implement capabilities to decode all ANT+ heart rate monitor data pages as outlined in this document.

4 Channel Configuration

The channel configuration parameters of the heart rate monitor and all other ANT-enabled devices are defined by the ANT protocol. Refer to the ANT Message Protocol and Usage document for definitions of the various channel parameters.

4.1 Receiver Channel Configuration

A device used to receive data from an ANT+ heart rate monitor must configure an ANT channel with the parameters listed in Table 1.

Parameter	Value	Comment
Channel Type	Receive (0x00)	The heart rate sensor is a transmit channel device; therefore the display or storage device must be configured as the receiver.
Network Key	ANT+ Managed Network Key	The ANT+ Managed Network Key is governed by the ANT+ Managed Network licensing agreement.
RF Channel	57	Channel 57 is used for the ANT+ heart rate monitor.
Transmission Type	0 for pairing	The transmission type must be set to 0 for a pairing search. Once the transmission type is learned, the receiving device should remember the type for future searches. To be future compatible, any returned transmission type is valid. Future versions of this spec may allow additional bits to be set in the transmission type.
Device Type	120 (0x78)	The device type shall be set to 120 (0x78) when searching to pair to an ANT+ heart rate monitor. Please see the ANT Message Protocol and Usage document for more details.
Device Number	1 – 65535 0 for searching	The transmitting sensor contains a 16-bit number that uniquely identifies its transmissions. Set the Device Number parameter to zero to allow wildcard matching. Once the device number is learned, the receiving device should remember the number for future searches. Please see the ANT Message Protocol and Usage document for more details.
Message Period	8070 counts	Data is transmitted from the heart rate monitor every 8070/32768 seconds (approximately 4.06Hz); however the receive rate can be set lower if required (refer to section 4.1.1).
Search Timeout	(Default = 30 seconds)	The default search timeout is set to 30 seconds in the ANT protocol. This timeout is implementation specific and can be set by the designer to the appropriate value for the system.

Table 1: ANT Channel Configuration for Receiving Heart Rate Data

4.1.1 Message Period

The message period is set up so that the display device can receive data at the full rate (~4.06 Hz) or at one half or one quarter of this rate; data can be received four times per second, twice per second, or once per second. The developer sets the message period count to receive data at one of the allowable receive rates:

- 8070 counts (~4.06 Hz, 4 messages/second)
- 16140 counts (~2.03 Hz, 2 messages/second)
- 32280 counts (~1.02 Hz, 1 message/second)

The minimum receive rate allowed is 32280 counts (~1.02 Hz).

The longer the count (i.e. lower receive rate) the more power is conserved by the receiver but a trade off is made for the latency of the data as it is being updated at a slower rate. The implementation of the receiving message rate by the display device is chosen by the developer.

As heart rates are typically greater than 1Hz (60bpm), the most robust solution to receive every heart beat timestamp for receiver applications that require R-R interval timing is to receive at 8070 counts (~4.06Hz). For applications on the receiver that display only the filtered computed heart rate (see section 5.3.2) a receive rate of 32280 counts (~1.02Hz) may be sufficient.

The new paging scheme of the heart rate monitor data allows for different pages of data to be sent. To incorporate receivers set at a slower receive rate the page toggle bit changes every 4^{th} message to ensure that all receivers see this toggle bit. For more information on the page toggle bit see section 5.3.1.

4.1.2 Receiver Channel Code Example

Figure 3 shows a code example of how to establish a receive channel using the channel parameters described in Table 1. This example uses the definitions in file ANT_DLL.h for understanding and clarity as these definitions are less involved than those for an embedded device. For more details on this code and examples of how to implement PC software please see the reference code provided with the ANT DLL.

```
/*******************************************************************************************************
 * These are the steps to set up an ANT channel to receive ANT+ heart rate monitor data.  The function
 * calls used here are based on the ANT PC Interface Functions.
 * *****************************************************************************************************/

#include "ant_dll.h"

void ConfigureRxChannel( UCHAR ucTransType, USHORT usDeviceNum )
{
   UCHAR ucNetNum     = 0x00;             //use network number 0
   UCHAR ucChanNum    = 0x00;             //assign channel 0
   UCHAR ucChanType   = 0x00;             //Receive or Slave channel
   UCHAR ucDeviceType = 0x78;             //Set the device type to 0x78 specific for the HRM
   UCHAR ucRF         = 0x39;             //Set the RF frequency to channel 57 - 2.457GHz
   UCHAR ucSearchTime = 0x0C;             //Set the search time to be 30 seconds (set in 2.5s increments)
   //substitute the ANT+ managed network key here
   UCHAR aucNetKey[8] = {0x__, 0x__, 0x__, 0x__, 0x__, 0x__, 0x__, 0x__};

   USHORT usMessagePeriod = 8070;         //Set the message period to 8070 counts specific for the HRM

   // Calls to set up the channel and open it.
   // Each call waits for an acknowledgement from ANT before continuing with the next call
   ANT_SetNetworkKey(ucNetNum, aucNetKey);
   if (!WaitAck(MESG_NETWORK_KEY_ID, MESSAGE_TIMEOUT))
      printf("Failed Setting the Network Key.\n");

   ANT_AssignChannel(ucChanNum, ucChanType, ucNetNum);
   if (!WaitAck(MESG_ASSIGN_CHANNEL_ID, MESSAGE_TIMEOUT))
      printf("Failed Assigning the Channel.\n");

   ANT_SetChannelId(ucChanNum, usDeviceNum, ucDeviceType, ucTransType);
   if (!WaitAck(MESG_CHANNEL_ID_ID, MESSAGE_TIMEOUT))
      printf("Failed Setting the Channel ID.\n");

   ANT_SetChannelPeriod(ucChanNum, usMessagePeriod);
   if (!WaitAck(MESG_CHANNEL_MESG_PERIOD_ID, MESSAGE_TIMEOUT))
      printf("Failed Setting the Message Period.\n");

   ANT_SetChannelRFFreq(ucChanNum, ucRF);
   if (!WaitAck(MESG_CHANNEL_RADIO_FREQ_ID, MESSAGE_TIMEOUT))
      printf("Failed Setting the Radio Frequency.\n");

   ANT_OpenChannel(ucChanNum);
   if (!WaitAck(MESG_OPEN_CHANNEL_ID, MESSAGE_TIMEOUT))
      printf("Failed Opening the Channel.\n");
}
```

Figure 3: Code Example of Receiver Channel Configuration

To search for a new heart rate monitor the ucTransType parameter should use a value of 0x00 for pairing and the usDeviceNum should use a value of 0x00 to allow for the wildcard search to take place. For more details on wildcard searching refer to the ANT Message Protocol and Usage document.

The code example in Figure 4 assumes that the ANT receiver is already receiving information from a new device that it used wildcard parameters to find. When the code is implemented the receiver finds the device number and transmission type parameters of the channel.

```
/*****************************************************************************************************************
 * These are the steps to get the ANT channel parameters from an active ANT channel and save the device
 * number and transmission type to variables to be used for later pairing.
 * ****************************************************************************************************************/
#include "ant_dll.h"

static void Main_ProcessANTEvent( void )
{
   UCHAR *aucRxBuf = ANTInterface_GetPendingEvent();
   ... // existing code
   switch (aucRxBuf[BUFFER_INDEX_MESG_ID] )  // switch the current event that needs processing
   {
      ... // existing code
      case MESG_BROADCAST_DATA_ID:
      case MESG_ACKNOWLEDGED_DATA_ID:
      case MESG_BURST_DATA_ID:
      {
         ... // existing code
         if ( usDeviceNum == 0 )
         {
            // To find the ANT channel parameters of the found channel the following call is made
            ANT_RequestMessage ( ucChanNum, MESG_CHANNEL_ID_ID );
         }
         ... // continue existing code
      }
      case MESG_CHANNEL_ID_ID:
      {
         // This is the message response from ANT to the ANT_RequestMessage() function.
         // The channel parameters Device Number and Transmission Type are saved to the variables
         usDeviceNum = aucRxBuf[1];
         usDeviceNum |= aucRxBuf[2] << 8;
         ucTransType = aucRxBuf[4];
         break;
      }
      ... // continue existing code
   }
}
```

Figure 4: Code Example of Finding and Saving Channel Parameters

4.2 Transmitter Channel Configuration

The ANT+ heart rate monitor shall establish its ANT channel as shown in Table 2.

Parameter	**Value**	Comment
Channel Type	Transmit (0x10)	Within the ANT protocol the transmit channel type (0x10) allows for bi-directional communication channels and utilizes the interference avoidance techniques and other features inherent to the ANT protocol.
Network Key	ANT+ Managed Network Key	The ANT+ Managed Network Key is governed by the ANT+ Managed Network licensing agreement.
RF Channel	57	Channel 57 is used for the ANT+ heart rate monitor.
Transmission Type	1 (0x01)	ANT+ devices follow the transmission type definition as outlined in the ANT protocol.
Device Type	120 (0x78)	The device shall transmit its device type as 120 (0x78). Please see the ANT Message Protocol and Usage document for more details.
Device Number	1-65535	This is a two byte field that allows for a unique identification of a given heart rate monitor. It is imperative that the implementation allow for a unique device number to be assigned to a given device. NOTE: The device number for the transmitting sensor cannot be set to 0x0000.
Message Period	8070 counts	Data is transmitted every 8070/32768 seconds (approximately 4.06 Hz).

Table 2: ANT Channel Configuration for Transmitting Heart Rate Information

4.2.1 Channel Type

The transmit channel type (0x10) is used instead of a transmit only channel type (0x50) for a number of reasons. The most compelling reason to use the transmit channel (0x10) is that this channel uses the interference avoidance features inherent to the ANT protocol. A transmit only channel (0x50) does not use these features and is much more susceptible to interference from other 2.4GHz sources including other ANT and ANT+ devices.

4.2.2 Device Number

The device number needs to be as unique as possible across production units. An example of achieving this specification is to use the lowest two bytes of the serial number of the device for the device number of the ANT channel parameter.

The device number of the heart rate monitor shall not be 0x0000. Be careful if the device number is derived from the lower 16-bits of a larger serial number that multiples of 0x10000 (65536) do not cause the device number to be set to 0.

Data page 2 has been created specifically to allow for the resolution of a four byte serial number. This page provides the upper two bytes of the serial number and assumes the lower two bytes are used as the device number in the ANT channel parameters. Please refer to section 5.3.4 for details.

4.2.3 Transmitter Channel Code Example

Figure 5 below shows a code example of how to establish a transmit channel using the channel parameters described in Table 2. This example uses the definitions in file ANT_DLL.h for understanding and clarity. For more details on this code and examples of how to implement PC software please see the reference code provided with the ANT DLL.

```
/**************************************************************************************************************
 * These are the steps to set up an ANT channel to transmit ANT+ heart rate monitor data.  The function
 * calls used here are based on the ANT PC Interface Functions.
 * ************************************************************************************************************/
#include "ant_dll.h"

void ConfigureTxChannel( USHORT usDeviceNum )
{
   UCHAR ucNetNum     = 0x00;             //use network number 0
   UCHAR ucChanNum    = 0x01;             //assign channel 1
   UCHAR ucChanType   = 0x10;             //Transmit or Master channel
   UCHAR ucDeviceType = 0x78;             //Set the device type to 0x78 specific for the HRM
   UCHAR ucTransType  = 0x01;             //Set the transmission type to 1 specific for the HRM
   UCHAR ucRF         = 0x39;             //Set the RF frequency to channel 57 (2.457GHz)
   UCHAR ucTxPower    = 0x03;             //Set the Tx Power to 0dbm
   //substitute the ANT+ managed network key here
   UCHAR aucNetKey[8] = {0x__, 0x__, 0x__, 0x__, 0x__, 0x__, 0x__, 0x__};
   USHORT usMessagePeriod = 8070;         //Set the message period to 8070 counts specific for the HRM

   // Calls to set up the channel and open it.
   // Each command call waits for an acknowledgement from ANT before continuing to send the next command
   ANT_SetNetworkKey(ucNetNum, aucNetKey);
   if (!WaitAck(MESG_NETWORK_KEY_ID, MESSAGE_TIMEOUT))
      printf("Failed Setting the Network Key.\n");

   ANT_AssignChannel(ucChanNum, ucChanType, ucNetNum);
   if (!WaitAck(MESG_ASSIGN_CHANNEL_ID, MESSAGE_TIMEOUT))
      printf("Failed Assigning the Channel.\n");

   ANT_SetChannelId(ucChanNum, usDeviceNum, ucDeviceType, ucTransType);
   if (!WaitAck(MESG_CHANNEL_ID_ID, MESSAGE_TIMEOUT))
      printf("Failed Setting the Channel ID.\n");

   ANT_SetChannelPeriod(ucChanNum, usMessagePeriod);
   if (!WaitAck(MESG_CHANNEL_MESG_PERIOD_ID, MESSAGE_TIMEOUT))
      printf("Failed Setting the Message Period.\n");

   ANT_SetChannelRFFreq(ucChanNum, ucRF);
   if (!WaitAck(MESG_CHANNEL_RADIO_FREQ_ID, MESSAGE_TIMEOUT))
      printf("Failed Setting the Radio Frequency.\n");

   ANT_OpenChannel(ucChanNum);
   if (!WaitAck(MESG_OPEN_CHANNEL_ID, MESSAGE_TIMEOUT))
      printf("Failed Opening the Channel.\n");
}
```

Figure 5: Code Example of Transmitter Channel Configuration

5 Message Payload Format

5.1 ANT+ Message Data Formats

All ANT messages have an 8 byte payload. For ANT+ messages, the first byte contains the data page number, and the remaining 7 bytes are used for sensor specific data. The 8 byte ANT+ message is referred to as a data page.

Byte #	Description	Length
0	Data Page Number	1 Bytes
1-7	Sensor Specific Data	7 Bytes

Table 3: ANT+ General Message Format

5.2 Data Page Types

Five different data pages are supported for the ANT+ heart rate monitor. These pages are divided into two distinct types of data. The first type is background information — data that is meant to be sent at a very slow update rate. The second type is main information that is sent for most of the data transmissions. Main data pages contain data that change quickly and need to be monitored.

5.2.1 Background Data Pages

The background data pages include pages 1, 2, and 3. These pages give information on cumulative operating time and manufacturer information. Background data pages 2 and 3 must be implemented. Page 1 is not required and its implementation is left to the discretion of the manufacturer.

5.2.1.1 Transmission Timing

A background message shall be sent every 65th message. This will allow the full manufacturer information and possibly the cumulative operating time to be transmitted at least once every 64.03 seconds.

5.2.2 Main Data Pages

The main data pages include page 0 and page 4. These pages are continuously sent from the heart rate monitor with the exception of every 65th message used by a background page. The choice of main data page is left to the discretion of the developer to implement. If interleaving of messages containing data pages 0 and 4 is desired the timing of this interleaving is decided by the developer.

5.2.3 Receiving Data Pages

An ANT+ receiver that wants to be compatible with the heart rate monitor device should implement all of the defined data pages in the device profile. This implementation is the only way that a receiver will be interoperable with existing and future ANT+ heart rate monitors.

5.3 Data Page Formats

The heart rate data format was the first defined ANT+ message format. This heart rate data format does not conform to the general ANT+ message definition. In order to add pages and maintain backwards compatibility, the following special rules apply to this message format:

1. The most significant bit of the Data Page Number is reserved for a Page Change Toggle Bit. This bit must be seen to toggle before the rest of the Data Page Number can be interpreted.
2. Bytes 1 – 3 are the only bytes that change definition in a page.
3. Bytes 4 – 7 have the same definition for every data page. This is the only data that can be interpreted before the Page Change Toggle Bit is seen to change.

5.3.1 Page Change Toggle Bit

The first byte of the heart rate data format comprises two data fields. Bits 0–6 determine the page number being used and identify the definition of the following three bytes.

The 7th bit or most significant bit (msb) is used for the page change toggle. The transmitter toggles the state of the toggle bit every fourth message (~1Hz) if the transmitter is using any of the page formats other than the page 0 data format. This allows the receiving/display unit to receive data from a heart rate sensor at a slower rate than 4Hz and still be able to observe the page change toggle bit to know that other data page formats are being used.

Figure 6 below shows how the toggle bit changes every fourth message. When the receiver sees the toggle bit change the new data pages can be decoded.

```
[00][FF][FF][FF][41][1E][D5][56]
[00][FF][FF][FF][61][3D][D6][56]
[00][FF][FF][FF][61][3D][D6][56]
[00][FF][FF][FF][61][3D][D6][56]
[80][FF][FF][FF][27][3E][D7][56]
[80][FF][FF][FF][27][3E][D7][56]
[80][FF][FF][FF][27][3E][D7][56]
[80][FF][FF][FF][27][3E][D7][56]
[00][FF][FF][FF][FA][3E][D8][56]
[00][FF][FF][FF][FA][3E][D8][56]
.
.
.
```

The receiver receives one of the messages in this set of four messages with the toggle bit set high. Now the new data pages can be decoded.

Figure 6: Example of Page Change Toggle Bit

5.3.2 Page 0 or Unknown Page Format

Page 0 allows for the heart rate monitor to send the measurement time of the last heart beat, the number of heart beats, and the computed heart rate. The computed heart rate is intended to be used directly by any display device without the need for any further calculations.

Byte	Description	Length	Value	Units	Rollover
0	Page Change Toggle	1 Bit (msb of Byte 0)	The transmitter must toggle this bit every 4 messages. The receiver may not interpret bytes 0-3 until it has seen this bit set to both a 0 and a 1.	N/A	N/A
0	Data Page Number	7 Bits (7 lsb of byte 0, mask 0x7F)	Data Page Number = 0x00	N/A	N/A
1	Reserved	3 Bytes	The transmitter shall set the value = 0xFF. The receiver shall not interpret this data at this time.	N/A	N/A
2	Reserved				
3	Reserved				
4	Heart Beat Event Time LSB	2 Bytes	Represents the time of the last valid heart beat event.	1/1024 second (s)	64s
5	Heart Beat Event Time MSB				
6	Heart Beat Count	1 Byte	A single byte value which increments with each heart beat event.	N/A	255 counts
7	Computed Heart Rate	1 Byte	Instantaneous heart rate. Invalid = 0x00 Valid Value = 1 – 255bpm. This value is intended to be displayed by the display device without further interpretation.	bpm	N/A

Table 4: Page 0 Heart Rate Data Format

5.3.2.1 Transmission Requirements

Page 0 is a main data page. Main data pages (page 0 or page 4) are continuously transmitted except for every 65th message; one background data page must be sent for every 64 main data pages that are transmitted.

5.3.3 Page 1 Format

Page 1 allows the receiver to determine the total time that the heart rate monitor has been active since the last battery change. The operating time increments by one count every two seconds, providing a maximum total time between rollovers of 33554432 seconds (9320 hours), which is greater than typical battery life. The operating time is reset when the battery is replaced.

Byte	Description	Length	Value	Units	Rollover
0	Page Change Toggle	1 Bit (msb of Byte 0)	Transmitter must toggle this bit every 4 messages. Receiver may not interpret bytes 0-3 until it has seen this bit set to both a 0 and a 1.	N/A	N/A
0	Data Page Number	7 Bits (7 lsb of byte 0, mask 0x7F)	Data Page Number = 0x01	N/A	N/A
1	Cumulative Operating Time (bits 0 - 7)	3 Bytes	Increments every 2 seconds and is reset on battery replacement	2 second intervals	33554430s
2	Cumulative Operating Time (bits 8 - 15)				
3	Cumulative Operating Time (bits 16 - 23)				
4	Heart Beat Event Time LSB	2 Bytes	Represents the time of the last valid heart beat event.	1/1024 second (s)	64s
5	Heart Beat Event Time MSB				
6	Heart Beat Count	1 Byte	A single byte value which increments with each heart beat event.	N/A	255 counts
7	Computed Heart Rate	1 Byte	Instantaneous heart rate. Invalid = 0x00 Valid Value = 1 – 255bpm. This value is intended to be displayed by the display device without further interpretation.	bpm	N/A

Table 5: Page 1 Heart Rate Data Format

5.3.3.1 Transmission Requirements

Page 1 is a background data page. A background page is sent every 65th message. The implementation of page 1 is left to the discretion of the manufacturer.

5.3.4 Page 2 Format

Page 2 allows the manufacturer to uniquely identify the heart rate monitor on the ANT+ network by setting the manufacturer identification field and by populating the serial number. Although the serial number allows for only two bytes of data, if it is used in conjunction with the device number a four byte serial number can be resolved.

Byte	Description	Length	Value	Units	Rollover
0	Page Change Toggle	1 Bit (msb of Byte 0)	The transmitter must toggle this bit every 4 messages. The receiver may not interpret bytes 0-3 until it has seen this bit set to both a 0 and a 1.	N/A	N/A
0	Data Page Number	7 Bits (7 lsb of byte 0, mask 0x7F)	Data Page Number = 0x02	N/A	N/A
1	Manufacturer ID	1 Byte	Contact the ANT+ Alliance if you wish to be added to this list as a heart rate sensor manufacturer.	N/A	N/A
2	Serial Number LSB	2 Bytes	This is the upper 16 bits of a 32 bit serial number.	N/A	N/A
3	Serial Number MSB				
4	Heart Beat Event Time LSB	2 Bytes	Represents the time of the last valid heart beat event.	1/1024 second (s)	64s
5	Heart Beat Event Time MSB				
6	Heart Beat Count	1 Byte	A single byte value which increments with each heart beat event.	N/A	255 counts
7	Computed Heart Rate	1 Byte	Instantaneous heart rate. Invalid = 0x00 Valid Value = 1 – 255bpm. This value is intended to be displayed by the display device without further interpretation.	bpm	N/A

Table 6: Page 2 Heart Rate Data Format

5.3.4.1 Transmission Requirements

Page 2 is a background data page. A background page is sent every 65th message. The implementation of page 2 is required by all manufacturers of ANT+ heart rate monitors.

5.3.4.2 Manufacturer ID

The list of manufacturer identification values is kept by the ANT+ Alliance. To obtain your unique manufacturer identification number please contact ANTAlliance@thisisant.com.

5.3.4.3 Serial Number Determination

The 16 bit device number allows a unique identification of the device in the RF domain, but cannot uniquely identify all manufactured heart rate monitors. When used in combination with the Manufacturer ID and the upper 16 bits of the serial number transmitted in this message, a unique identification of the heart rate monitor can be made.

The 32-bit serial number comprised of the upper serial number (most significant 16 bits) and the device number (least significant 16 bits) provides more than 4 billion serial numbers for each manufacturer ID. The manufacturer must ensure that this data is unique for each heart rate monitor produced.

It is important to note that the device ID must never be 0, therefore serial numbers that are integer multiples of 65536 must not be used. See section 4.2.2 for more details.

5.3.5 Page 3 Format

Page 3 allows the manufacturer to set and transmit hardware and software versions of the heart rate monitor as well as the model number.

Byte	Description	Length	Value	Units	Rollover
0	Page Change Toggle	1 Bit (msb of Byte 0)	The transmitter must toggle this bit every 4 messages. The receiver may not interpret bytes 0-3 until it has seen this bit set to both a 0 and a 1.	N/A	N/A
0	Data Page Number	7 Bits (7 lsb of byte 0, mask 0x7F)	Data Page Number = 0x03	N/A	N/A
1	Hardware Version	1 Byte	To be set by the manufacturer.	N/A	N/A
2	Software Version	1 Byte	To be set by the manufacturer.	N/A	N/A
3	Model Number	1 Byte	To be set by the manufacturer.	N/A	N/A
4	Heart Beat Event Time LSB	2 Bytes	Represents the time of the last valid heart beat event.	1/1024 second (s)	64s
5	Heart Beat Event Time MSB				
6	Heart Beat Count	1 Byte	A single byte value which increments with each heart beat event.	N/A	255 counts
7	Computed Heart Rate	1 Byte	Instantaneous heart rate. Invalid = 0x00 Valid Value = 1 – 255bpm. This value is intended to be displayed by the display device without further interpretation.	bpm	N/A

Table 7: Page 3 Heart Rate Data Format

5.3.5.1 Transmission Requirements

Page 3 is a background data page. A background page is sent every 65th message. The implementation of page 3 is required by all manufacturers of ANT+ heart rate monitors.

5.3.6 Page 4 Format

Page 4 allows the heart rate monitor to transmit the measured time of the previously measured heart beat. This format provides a level of redundancy in the transmitted data stream.

Byte	Description	Length	Value	Units	Rollover
0	Page Change Toggle	1 Bit (msb of Byte 0)	The transmitter must toggle this bit every 4 messages. The receiver may not interpret bytes 0-3 until it has seen this bit set to both a 0 and a 1.	N/A	N/A
0	Data Page Number	7 Bits (7 lsb of byte 0, mask 0x7F)	Data Page Number = 0x04	N/A	N/A
1	Manufacturer Specific	1 Byte	The receiver shall not interpret this field.	N/A	N/A
2	Previous Heart Beat Event Time LSB	2 Bytes	Represents the time of the previous valid heart beat event.	1/1024 second (s)	64s
3	Previous Heart Beat Event Time MSB				
4	Heart Beat Event Time LSB	2 Bytes	Represents the time of the last valid heart beat event.	1/1024 second (s)	64s
5	Heart Beat Event Time MSB				
6	Heart Beat Count	1 Byte	A single byte value which increments with each heart beat event.	N/A	255 counts
7	Computed Heart Rate	1 Byte	Instantaneous heart rate. Invalid = 0x00 Valid Value = 1 – 255bpm. This value is intended to be displayed by the display device without further interpretation.	bpm	N/A

Table 8: Page 4 Heart Rate Data Format

5.3.6.1 Transmission Requirements

Page 4 is a main data page. Main data pages (page 0 or page 4) are continuously transmitted except for every 65th message; one background data page must be sent for every 64 main data pages that are transmitted.

5.3.6.2 Manufacturer Specific Field

This field allows manufacturers to add 1 byte of data to this data page. The field is not interpreted by the receiver and can be used for debugging or any other purpose required by the manufacturer.

5.3.6.3 R-R Interval Measurements

This data page allows for R-R calculation based on the difference between the last heart beat event time and the previous heart beat event time. A sample calculation of the R-R measurement is shown below in Figure 7.

```
// previous existing code

if ((ucBeatCount – ucPreviousBeatCount) == 1)  // ensure that there is only one beat between time intervals
{
   usR_R_Inteval = usBeatTime – usPreviousBeatTime;  // subracting the event times gives the R-R interval

   //converting the timebase from 1/1024 of a second to milliseconds
   usR_R_Interval_ms = (((ULONG) usR_R_Interval) * (ULONG) 1000) / (ULONG) 1024;
}

ucPreviousBeatCount = ucBeatCount;  // update the previous beat count
usPreviousBeatTime = usBeatTime;    // update the previous beat count

// continue existing code
```

Figure 7: Code Example of R-R Interval Calculations

The R-R interval can be computed by subtracting (in USHORT form) the last event time from the previous event time (bytes 4|5 from bytes 2|3) after making sure that the event count has changed from the previous reported R-R interval.

Data page 4 has the advantage that an R-R value can be computed without receiving the immediately preceding event. However, code can be implemented to use other message types and calculate R-R intervals using the previous event.

5.3.7 Page 5 – 127 Formats

These pages are reserved for future use.

6 Implementation Guidelines – Computed Heart Rate

The heart rate monitor computes and filters the heart rate and transmits it in the computed heart rate field. This field is designed to be directly displayed. Further filtering can be added on the receiver if desired by monitoring the cumulative heart beat count. If the count does not increment over a period of time then the user may have removed the monitor, or the user may not be wearing the heart rate monitor correctly. In either case, if the count does not increment it is recommended that code be implemented to blank the heart rate display rather than displaying erroneous computed heart rate data.

6.1 Receiver Implementation Code Example

ANT devices are capable of transmitting data in one of three different formats as defined in the ANT protocol. For more information on these data types and how they are used please refer to the ANT Protocol Messaging and Usage document.

The display device should be able to accept all of the ANT data message types:

- Broadcast Data (Message ID = 0x4E)
- Acknowledged Data (Message ID = 0x4F)
- Burst Data (Message ID = 0x50)

The following example code shows the state machine necessary to handle the ANT message data types and shows a specific implantation for the ANT+ Heart Rate Monitor. This demonstrates the ability to use the defined toggle change bit and search for specific page numbers and has the ability to easily add new page definitions as they are required.

Many of the constant variables used in the below example come from the ANT Reference Designs. For more detailed information on this code example and for more embedded code examples please see the ANT Reference Design User Manual.

```
//previous existing code

switch (pucEventBuffer[BUFFER_INDEX_MESG_ID])  // switch the current event that needs processing
{                                            // pucEventBuffer holds the returned message from ANT
   case MESG_BROADCAST_DATA_ID:
   case MESG_ACKNOWLEDGED_DATA_ID:
   case MESG_BURST_DATA_ID:
   {
      UCHAR ucPage = pucEventBuffer[BUFFER_INDEX_MESG_DATA];
      switch(pucEventBuffer[BUFFER_INDEX_CHANNEL_NUM])
      {
         case CHANNEL_0:  // Channel 0 will be collecting HRM data
         {
            static UCHAR ucOldPage;
            static UCHAR ucState = INIT_PAGE;  // sets the state of the receiver - INIT, STD_PAGE, EXT_PAGE
            if (ucState == INIT_PAGE)
            {
               ucState = STD_PAGE;  // change the state to STD_PAGE and allow the checking of old and new pages
            }
            // decode with pages if the page byte or toggle bit has changed
            else if ( (ucPage != ucOldPage) || (ucState == EXT_PAGE) )
            {
               ucState = EXT_PAGE;  // set the state to use the extended page format

               switch (ucPage & ~TOGGLE_MASK) //check the new pages and remove the toggle bit, TOGGLE_MASK = 0x80
               {
                  case PAGE_1:
                  {
                     //decode the cumulative operating time
                     ulOperatingTime  =  pucEventBuffer[BUFFER_INDEX_MESG_DATA + 1];
                     ulOperatingTime  |= pucEventBuffer[BUFFER_INDEX_MESG_DATA + 2] << 8;
                     ulOperatingTime  |= pucEventBuffer[BUFFER_INDEX_MESG_DATA + 3] << 16;
                     ulOperatingTime = 2 * ulOperatingTime;
                     break;
                  }
                  case PAGE_2:
                  {
                     //decode the Manufacturer ID
                     ucManId = pucEventBuffer [BUFFER_INDEX_MESG_DATA + 1];
                     //decode the 4 byte serial number
                     ulSerialNumber =  usDeviceNum;
                     ulSerialNumber |= pucEventBuffer [BUFFER_INDEX_MESG_DATA + 2] << 16;
                     ulSerialNumber |= pucEventBuffer [BUFFER_INDEX_MESG_DATA + 3] << 24;
                     break;
                  }
                  case PAGE_3:
                  {
                     //decode HW version, SW version, and model number
                     ucHwVersion  = pucEventBuffer[BUFFER_INDEX_MESG_DATA + 1];
                     ucSwVersion  = pucEventBuffer[BUFFER_INDEX_MESG_DATA + 2];
                     ucModelNum   = pucEventBuffer[BUFFER_INDEX_MESG_DATA + 3];
                     break;
                  }
                  case PAGE_4:
                  {
                     //decode the previous heart beat measurement time
                     usPreviousBeat =  pucEventBuffer [BUFFER_INDEX_MESG_DATA + 2];
                     usPreviousBeat |= pucEventBuffer [BUFFER_INDEX_MESG_DATA + 3] << 8;
                     break;
                  }
               }
            }
         // decode the last four bytes of the HRM format, the first byte of this message is the channel number
         DecodeDefaultHRM( &pucEventBuffer[BUFFER_INDEX_MESG_DATA + 4] );
         ucOldPage = ucPage;
         }
      // continue code for all of the active channels
      }
   }
}
//code continues

/****************************************************************************************************
 * This function is used to decode the last 4 bytes of data from the HRM message.  The variables
 * usMeasurement Time, ucMeasurementCount, and ucComputedHeartRate have been defined previously.
 * ***********************************************************************************************/
void DecodeDefaultHRM(UCHAR* pucPayload)
{
   // decode the measurement time data (two bytes)
   usBeatTime   = pucPayload[0];
   usBeatTime  |= pucPayload[1] << 8;
   // decode the measurement count data
   ucBeatCount  = pucPayload[2];
   // decode the measurement count data
   ucComputedHeartRate = pucPayload[3];
}
```

Figure 8: Receiver Code Example of Decoding ANT Data Message Types