Serial Protocol Documentation

MECHTRON 3K04 Assignment 2
Group 4
Andrew De Rango, Ali Hussin, Ethan Otteson,
Marco Tan, Rafael Toameh
Friday, November 29, 2024

Table of Contents

1 Scope

This document outlines a comprehensive overview of serial communication using UART on the FRDM-K64F platform. Our goal was to create a seamless way for our Simulink model to exchange data with a computer, which required careful consideration of details such as baudrate, packet structure, and data formatting. By designing a custom serial packet protocol, we aimed to ensure that data transfers happen smoothly and reliably, which is an essential component of any medical device. This document outlines the program's requirements, protocol design, and justification. Specific implementation and verification details are not included in this document and can be found in the specific Pacemaker Firmware or DCM documentation.

2 Terms and Definitions

UART - (Universal asynchronous receiver/transmitter) - Hardware communication protocol that allows for asynchronous serial communication between devices.

DCM - Device Controller-Monitor, see separate documentation for further details.

HeartFlow - Name of the application that operates as the DCM.

Pacemaker Firmware - All software that is embedded into the pacemaker microcontroller, see separate documentation for further details.

Electrogram - Term used for electrical activity of the heart recorded by the pacemaker and graphed. Also referred to as egram.

3 Requirements

The requirements for the serial protocol between the Pacemaker Firmware and the DCM are quite straightforward. The serial protocol shall describe a common message structure used by both platforms to ensure message interpretation is agreed upon. The protocol should describe the message length and baud rate used and the structure for all design-relevant messages. The protocol should provide a message structure used to change pacemaker parameters, send electrogram data, and communicate the pacemaker's serial number.

4 Protocol Design

The following sections describe the serial communication protocol created by our team for this project. First is a description and justification of each message type followed by diagrams illustrating the behavior of the protocol and then the specific structure of each message is discussed. It was decided to create multiple message types for the serial communication protocol because it would allow for easier expansion of functionality in the future

by adding a new message type. Additionally, it allows for the separation of concerns as each message type is separate and distinct. The various message types also allow for more efficient communication for instance if a command to toggle electrogram data sending does not also contain useless parameter data. Details on each of these modes are discussed below.

4.1 Description of Message Types

This protocol describes five message types used to communicate between the DCM and Pacemaker Firmware. Each message type varies slightly depending on whether it is being sent from the DCM to the Pacemaker Firmware or vice versa. The five message types are handshake, poll request, parameter change request, parameter confirmation, and toggle electrogram. Each of these messages is discussed below.

1. **Handshake (0x01)**
   We like to think of the handshake as the initial greeting when the pacemaker first connects to the Device Communication Manager (DCM). It's a way for the DCM to say, "Who's out there?" and for the pacemaker to respond, "Here's my ID." This step is essential because, just like we double-check names in introductions, the DCM needs to confirm it's talking to the right device and that the serial number is legitimate.
   - *How it works:*
     When a pacemaker is not connected the DCM sends a general broadcast message to all connected devices, essentially asking, "Who's on this network?" The pacemaker responds by sending a message containing its unique serial number. This lets the DCM know exactly who it's communicating with and that this device is ready for interaction.
   - *DCM's Message:*
     When the DCM is ready to connect to a device it will send out a handshake message to all devices connected to that computer. This message will contain a specific identifier for the DCM to limit the chance that the pacemaker will mistake any random device for the DCM. The DCM will be unable to send any other message types to the pacemaker until a successful handshake is complete in which the pacemaker returns a message containing its serial number which matches the logged-in user's registered serial number.
   - *Pacemaker's Message:*
     Upon receiving a handshake message from the DCM with the specific DCM identifier, the pacemaker will reply with a message of the same type containing the pacemaker's serial number. Before a successful handshake, the pacemaker will ignore any other messages it receives.
2. **Poll Requests (0x02)**
   Poll requests are like periodic check-ins to ensure the pacemaker is still connected and working as expected. Imagine a lifeguard on duty, checking in with each swimmer every few minutes—this is a similar idea. Every few seconds, the DCM sends a quick "still there?" message to the pacemaker to confirm it hasn't gone offline or encountered any issues.

- *How it works:*
  The DCM sends out a poll request periodically, which acts as a quick check to ensure the same pacemaker is still connected. If several poll requests are sent by the DCM and not responded to, the DCM will assume the pacemaker has been disconnected. The pacemaker's only role in a poll request is to respond if prompted.
- *DCM's Message:*
  Following a successful handshake, the DCM will periodically send poll requests until the pacemaker has been disconnected or the user logs out. The DCM simply sends a message with the polling request type and an ID number to track the message. The pacemaker will read the message type and know to send a response.
- *Pacemaker's Message:*
  Upon receiving a message of type polling request from the DCM, the pacemaker will respond with the same message structure as the handshake reply. This means it will return the same message type and ID number that it received and its serial number. This allows the DCM to interpret the message as a type of polling request, associate it with the message it sent out based on the ID number, and ensure that the pacemaker that responded is the one it expected based on the serial number. If this reply is not received from the pacemaker or the serial number does not match the serial number provided during the handshake, the DCM will assume the original pacemaker has been disconnected and terminate the connection.

3. **Parameter and Mode Change Request (0x03)**
   This message type works along with the message type "Parameter and Mode Confirmation (0x04)" as a multistep process required to change the parameters or mode of the pacemaker. Adding defence in depth, to ensure the changes are correct before they are implemented into the operation of the pacemaker. It's a bit like adjusting the settings on a smartphone and being asked to review and confirm the changes before they are implemented. This message type is responsible for sending the new parameters from the DCM to the pacemaker and the pacemaker asking if they are correct.
   - *How it works:*
     Once a user submits a new mode and/or parameters, the DCM will send a message of this type to the pacemaker. The pacemaker will save these parameters without implementing them into the operation of the pacemaker and reply with its interpretation of the variables received. If the DCM can confirm these two sets of variables match then the process will move on to the next message type to get the parameters actually implemented. However, if these sets of variables differ then the DCM will resend this original message to avoid implementing unexpected parameters which may be unsafe. If the variable sets continue to not match this resending process will continue until a retry limit is reached at which the DCM should alert the user that there is an issue.
   - *DCM's Message:*
     Once the user submits a new mode and/or parameters, the DCM will send this

message type. This message will contain an ID number, message type, and all parameters submitted. The pacemaker will interpret this message and save each parameter to a temporary variable that is not implemented into the operation of the pacemaker.

- *Pacemaker's Message:*
  Receiving a message of this type from the DCM, the pacemaker will reply with its interpretation of the parameters. The message will include the message type and ID number that it received to allow the DCM to associate the reply with the initial request. Additionally, each parameter will be sent back to ensure the DCM and pacemaker agree on what the new parameters should be. Assuming this interpretation of the parameters is the same as the initial parameters sent, the DCM will move on to the next message type described to complete the process of changing the mode and/or parameters.

4. **Parameter and Mode Confirmation (0x04)**
   This message type works along with the message type "Parameter and Mode Change Request (0x03)" as a multistep process required to change the parameters or mode of the pacemaker. Adding defence in depth, to ensure the changes are correct before they are implemented into the operation of the pacemaker. It's a bit like adjusting the settings on a smartphone and being asked to review and confirm the changes before they are implemented. This message type is responsible for the parameters actually being implemented into the pacemaker's operation.
   - *How it works:*
     This message type acts as a review to ensure the Pacemaker Firmware and DCM agree on the new parameters before they are implemented. After the Pacemaker Firmware responds to the previous message type with its interpretation, the DCM sends a confirmation message to the pacemaker if the expected values were returned. The Pacemaker Firmware will then implement those values into its current behaviour and as a final check send the now-implemented parameters back to the DCM. If the values implemented were correct then the process of changing the variables is complete, however, if the DCM receives the final message and it contains different parameters than expected, then the whole process starting with the previous message type will be restarted.
   - *DCM's Message:*
     This message type will only be sent if the previous message type discussed was successful. The parameter confirmation message is simply an ID number and the message type. This will signal the pacemaker to implement the parameters stored in temporary variables to the actual operational variables and change the behaviour of the pacemaker.
   - *Pacemaker's Message:*
     Once the operational variables are modified, the pacemaker will send a reply message to the DCM containing the same ID number and message type received and the current set of parameters. This acts as a final safeguard so that the DCM can confirm that the correct parameters were implemented into the

pacemaker. If the parameters are as expected, the variable change process is complete. However, if they are different then the DCM will restart the whole process of changing the parameters and resend both involved message types to correct the issue.

5. **Electrogram Data (0x05)**
   Electrogram data is a real-time stream of the pacemaker-sensed electrical activity, providing us with insights into its status. When the DCM requests electrogram data, it's like flipping on a live stream that shows exactly what's happening inside the heart in real time. This data is essential for monitoring the pacemaker's function and can be vital for a clinical or technician.
   - *How it works:*
     The electrogram message type works like a toggle on the DCM side and is how the electrogram data is sent on the pacemaker side. The DCM can toggle on or off the pacemaker's electrographic data. This ensures that unnecessary data is not being sent when the DCM is not using it. When electrogram data is toggled on, the pacemaker will periodically send a message containing many samples of electrogram data to the pacemaker. If electrogram data is toggled off, the pacemaker will not send these messages.
   - *DCM's Message:*
     The electrogram message that the DCM sends to the pacemaker includes simply an ID number, message type, and whether it is toggled on or off. The pacemaker will then begin sending or stop sending electrogram data appropriately.
   - *Pacemaker's Message:*
     When electrogram data is toggled on, the pacemaker will sample the sensed atrial and ventricular activity and compute several of these samples into a message to send periodically to the DCM. This data can then be plotted on the DCM. When the electrogram is toggled off, the pacemaker will not send any electrogram messages.

4.2 Visual Description of Message Types

The following two diagrams (Figure 4.2.a & 4.2.b) provide a visual representation of the behaviour previously described for the serial protocol. Each message type and the expected responses are included.

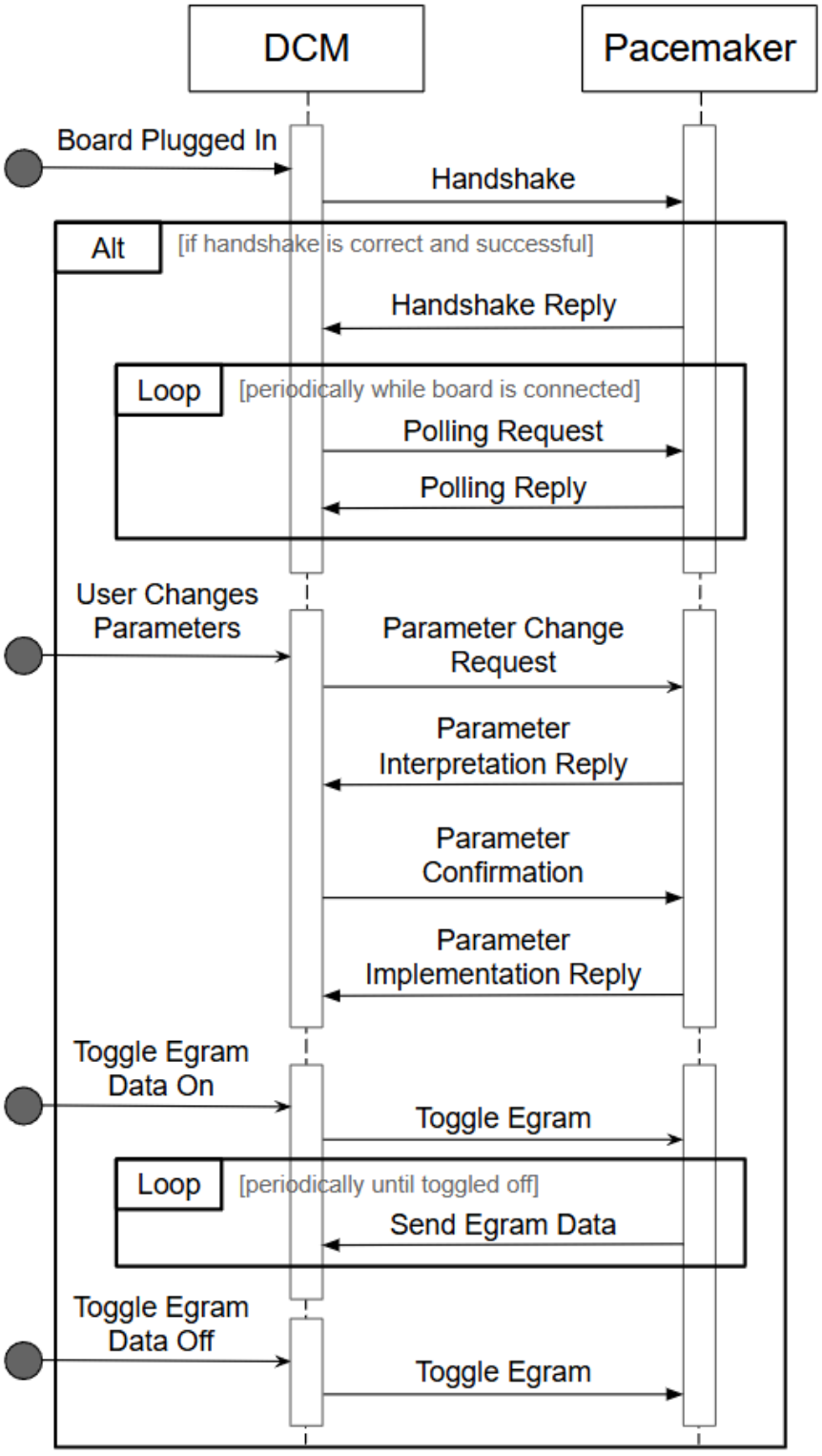

Figure 4.2.a: Sequence diagram for the serial communication protocol between the DCM and the Pacemaker Firmware. The diagram highlights that the handshake must be completed before any other communication can be done, the periodic nature of the polling requests and electrogram data, and the process required to change parameters.

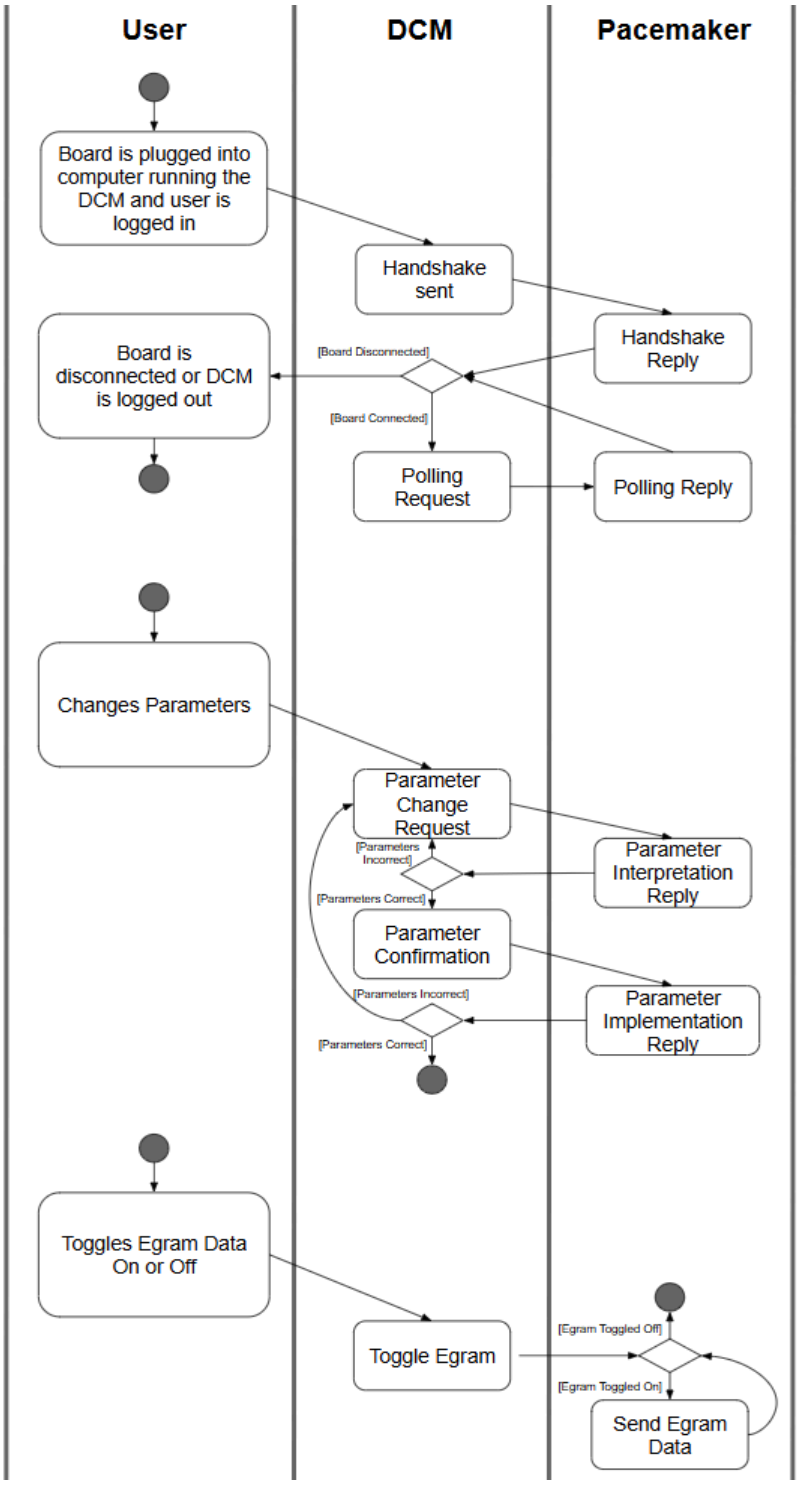

Figure 4.2.b: Activity diagram for the serial communication protocol between the DCM and the Pacemaker Firmware. The diagram distinguishes whether the user, DCM, or pacemaker completes the activity. The diagram illustrates the behaviour for the three user inputs which involve serial communication; the handshake and polling requests, changing parameters, and toggling electrogram data. Note that changing parameters and toggling electrogram data both require a successful handshake before they can be executed.

4.3 Specific Structure of Message Types

The specific structure of each message type is shown in the following tables and discussed. Justification of data type choice is also provided for each section. All messages that are part of this serial communication protocol operate at a baud rate of 115200 and have a message length of 82 bytes. Each message begins with a two-byte header containing the message ID number, which is used by the DCM to track responses to messages it sends to the pacemaker, and the message type which is used by both the DCM and pacemaker to know which one of these tables it should use to interpret the message.

4.3.1 Handshake (0x01)

The following table pertains to the handshake message structure sent from the DCM to the pacemaker. The message ID is a number between 0-255 which the DCM assigns to a message to associate it with its reply. The DCM should never have two open message conversations with the same ID. The message type is 0x01 in hexadecimal. The validation bytes spell out “HeartFlow” and are used as the DCM’s unique identifier to prevent the pacemaker from handshaking with other devices. Any unused bytes are set to zero and left empty.

Table 4.3.1.a: Handshake Structure for Message Sent from the DCM to the Pacemaker.

Byte Index	**Category**	**Name**	**Datatype**
0	Header	Message ID	uint8 (1 byte)
1	Header	Message Type	uint8 (1 byte)
2	Body	SW Validation Character 1	uint8 (1 byte)
3	Body	SW Validation Character 2	uint8 (1 byte)
4	Body	SW Validation Character 3	uint8 (1 byte)
5	Body	SW Validation Character 4	uint8 (1 byte)
6	Body	SW Validation Character 5	uint8 (1 byte)
7	Body	SW Validation Character 6	uint8 (1 byte)
8	Body	SW Validation Character 7	uint8 (1 byte)
9	Body	SW Validation Character 8	uint8 (1 byte)
10	Body	SW Validation Character 9	uint8 (1 byte)
11-81	Body	Unused	N/A

The following table pertains to the handshake reply message structure sent from the pacemaker to the DCM. The message ID and message type are identical to the ID and type of

the message that initiated the reply. The pacemaker serial number is the unique identifier of that pacemaker which allows the DCM to ensure it is connected to the correct pacemaker. Using a uint16 is appropriate for the serial number as it allows for 65,536 different pacemakers to be created. If the expected production numbers increased in the future this would need to be changed. Any unused bytes are set to zero and left empty.

Table 4.3.1.b: Handshake Structure for Message Sent from the Pacemaker to the DCM.

Byte Index	Category	Name	Datatype
0	Header	Message ID	uint8 (1 byte)
1	Header	Message Type	uint8 (1 byte)
2-3	Body	Pacemaker Serial Number	uint16 (2 bytes)
4-81	Body	Unused	N/A

4.3.2 Poll Requests (0x02)

The following table pertains to the poll request message structure sent from the DCM to the pacemaker. The message ID is a number between 0-255 which the DCM assigns to a message to associate it with its reply. The DCM should never have two open message conversations with the same ID. The message type is 0x02 in hexadecimal. Any unused bytes are set to zero and left empty.

Table 4.3.2.a: Polling Structure for Message Sent from the DCM to the Pacemaker.

Byte Index	Category	Name	Datatype
0	Header	Message ID	uint8 (1 byte)
1	Header	Message Type	uint8 (1 byte)
2-81	Body	Unused	N/A

The following table pertains to the poll request-reply message structure sent from the pacemaker to the DCM. This message has an identical structure to the handshake reply in Table 4.3.1.b except the message type echoed back will be 0x02 rather than 0x01. The message ID and message type are identical to the ID and type of the message that initiated the reply. The pacemaker serial number is the unique identifier of that pacemaker which allows the DCM to ensure it is connected to the correct pacemaker. Using a uint16 is appropriate for the serial number as it allows for 65,536 different pacemakers to be created. If the expected production numbers increased in the future this would need to be changed. Any unused bytes are set to zero and left empty.

Table 4.3.2.b: Polling Structure for Message Sent from the Pacemaker to the DCM.

Byte Index	Category	Name	Datatype
0	Header	Message ID	uint8 (1 byte)
1	Header	Message Type	uint8 (1 byte)
2-3	Body	Pacemaker Serial Number	uint16 (2 bytes)
4-81	Body	Unused	N/A

4.3.3 Parameters and Mode Change Request (0x03)

The following table pertains to the parameter and mode change request message structure sent from the DCM to the pacemaker, and from the pacemaker to the DCM. Importantly, the 0x03 message structure is entirely independent of the direction of the message. The message ID is a number between 0-255 which the DCM assigns to a message to associate it with its reply. The DCM should never have two open message conversations with the same ID. The message type is 0x03 in hexadecimal. The pacemaker will echo back the same message ID and type supplied in the DCM's message. The pacemaker mode has assigned numeric values discussed in the Pacemaker Firmware documentation for each distinct mode. The mode can be represented by a uint8 as there are far less than 256 modes. The lower and upper rate limits can be represented by uint8 because they are positive integer values less than 255. This is the same reason that the pulse widths, rate factor, reaction time, and recovery time are represented by uint8 (refer to the Pacemaker Firmware documentation as the bounds for reaction and recovery time were modified from the provided requirements). The refractory periods and AV delay had to be represented by uint16 as although they are positive integers, they can have values greater than 255. The amplitudes and sensitivities were represented by single floats as although they are decimals they do not require the precision of a double float. Finally, the activity threshold is represented as an integer between 1-7 which can be seen in the DCM and Pacemaker Firmware specific documentation. Any unused bytes are set to zero and left empty.

Table 4.3.3.a: Parameter Change Message Structure for DCM and Pacemaker.

Byte Index	Category	Name	Datatype
0	Header	Message ID	uint8 (1 byte)
1	Header	Message Type	uint8 (1 byte)
2	Body	Pacemaker Mode	uint8 (1 byte)
3	Body	Lower Rate Limit	uint8 (1 byte)
4	Body	Upper Rate Limit	uint8 (1 byte)
5-6	Body	Atrial Refractory Period	uint16 (2 byte)

7-8	Body	Ventricular Refractory Period	uint16 (2 byte)
9	Body	Atrial Pulse Width	uint8 (1 byte)
10	Body	Ventricular Pulse Width	uint8 (1 byte)
11-14	Body	Atrial Amplitude	single (4 byte)
15-18	Body	Ventricular Amplitude	single (4 byte)
19-22	Body	Atrial Sensitivity	single (4 byte)
23-26	Body	Ventricular Sensitivity	single (4 byte)
27-28	Body	Atrioventricular Delay	uint16 (2 byte)
29	Body	Rate Factor	uint8 (1 byte)
30	Body	Activity Threshold	uint8 (1 byte)
31	Body	Reaction Time	uint8 (1 byte)
32	Body	Recovery Time	uint8 (1 byte)
33-81	Body	Unused	N/A

4.3.4 Parameters and Mode Confirmation (0x04)

The following table pertains to the parameter and mode confirmation message structure sent from the pacemaker to the DCM. The message ID is a number between 0-255 which the DCM assigns to a message to associate it with its reply. The DCM should never have two open message conversations with the same ID. The message type is 0x04 in hexadecimal. Any unused bytes are set to zero and left empty. Sending this message type is sufficient to let the pacemaker know that it is good to transfer the temporary parameters to operational parameters, no need for an additional byte.

Table 4.3.4.a: Confirmation Structure for Message Sent from the DCM to the Pacemaker.

Byte Index	**Category**	**Name**	**Datatype**
0	Header	Message ID	uint8 (1 byte)
1	Header	Message Type	uint8 (1 byte)
2-81	Body	Unused	N/A

The pacemaker will reply to the 0x04 message type with the same structure as seen in Table 4.3.3.a but with a message type of 0x04, and all of the parameters will be the actual

operational parameters rather than the temporary parameters sent for a 0x03 message. No additional table is provided.

4.3.5 Electrogram Data (0x05)

The following table pertains to the electrogram toggle message structure sent from the DCM to the pacemaker. The message ID is a number between 0-255 which the DCM assigns to a message to associate it with its reply. The DCM should never have two open message conversations with the same ID. The message type is 0x05 in hexadecimal. The electrogram toggle byte allows the same message structure to be used to toggle on and off the electrogram data transmission. If the electrogram toggle byte is zero, then the pacemaker will not send an electrogram but if it is any other value then electrogram data will be sent periodically. Any unused bytes are set to zero and left empty.

Table 4.3.5.a: Electrogram Structure for Message Sent from the DCM to the Pacemaker.

Byte Index	Category	Name	Datatype
0	Header	Message ID	uint8 (1 byte)
1	Header	Message Type	uint8 (1 byte)
2	Body	Electrogram Toggle	uint8 (1 byte)
3-81	Body	Unused	N/A

The following table pertains to the electrogram data message structure sent from the pacemaker to the DCM when electrogram data is toggled on. The message will be sent periodically every ten sampled time steps, note that the specific sampling rate is discussed in the Pacemaker Firmware documentation. The message ID is not required for electrogram data as it is not a direct response to any specific DCM message. The message type is 0x05 in hexadecimal. The sensed data was chosen to be represented as a single float as it provided enough precision to plot the electrogram charts within the DCM while minimizing the data size of the whole message as this is the longest message type used. The atrial and ventricular sensing both consist of ten samples in sequential order, one atrial and one ventricular sample is taken each time step.

Table 4.3.5.b: Electrogram Structure for Message Sent from the Pacemaker to the DCM.

Byte Index	Category	Name	Datatype
0	Header	Unused	N/A
1	Header	Message Type	uint8 (1 byte)
2-41	Body	Atrial Sensing	single (4 byte)
42-81	Body	Ventricular Sensing	single (4 byte)