



# G.FIT

## User Guide and Specification



## Copyright Information and Usage Notice

This information disclosed herein is the exclusive property of Garmin Canada Inc. No part of this publication may be reproduced or transmitted in any form or by any means including electronic storage, reproduction, execution, or transmission without the prior written consent of Garmin Canada Inc. The recipient of this document by its retention and use agrees to respect the copyright of the information contained herein.

The information contained in this document is subject to change without notice and should not be construed as a commitment by Garmin Canada Inc. unless such commitment is expressly given in a covering document.

The Garmin Canada Inc. ANT Products described by the information in this document are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Garmin product could create a situation where personal injury or death may occur. If you use the Products for such unintended and unauthorized applications, you do so at your own risk and you shall indemnify and hold Garmin and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Garmin was negligent regarding the design or manufacture of the Product.

©2019 Garmin Canada Inc. All Rights Reserved.

## Revision History

Revision	Effective Date	Description
1.0	June 2017	Initial Customer Sample Release.
1.1	September 2017	Customer Sampling Update
1.2	September 2017	Production Release
2.0	May 2018	Update serial interface with messaging for G.FIT 4.0.0 <ul style="list-style-type: none"> <li>• Add controllable features</li> <li>• Extension to custom events</li> <li>• Consistent resistance resolution</li> <li>• Product information configuration outside of FE metrics, with customizable device/manufacture strings</li> <li>• Add configuration option to disable HR pairing</li> <li>• Add rower instantaneous pace as a transmitted field over BLE</li> </ul> Additions to GFIT_SetDeviceID (0xE3) to include information on how the channel ID is constructed from the device ID
2.0_Alpha.05	May 2018	Format cleanup, G.FIT template. Spelling and phrasing changes in progress.
2.0_alpha.06	May 2018	Added section 4.7 – G.FIT custom service. Reworking spin down command payload table and descriptions Fixed all references.
2.0._alpha.07	May 2018	Fixed incorrect message sizes, missing/incorrect invalid values. Updated section 4.6 – Custom Events. Added section 6.1- ANT+ Manufacturer specific pages Add clearer resistance specification and calcs.
2.0_alpha.10	July 2018	Merge in changes from Tom and Mark.
2.0_alpha.11	November 2018	Updating company branding
2.0_alpha.13	June 2019	Updated HR Serial Message Format
2.0_alpha.14	June2019	Added more parameters to section 7.3.2 – gfit_fep_set_channel_configuration Added interaction details for new parameters and gfit_hrp_set_pairing_mode Added section 7.3.23 – gfit_fep_set_developer_options (0xDB) Added section 7.5.8 – gfit_event_ble_peripheral (0xBA)

## Table of Contents

<b>1</b>	<b>Support .....</b>	<b>11</b>
1.1	G.FIT technical references .....	11
<b>2</b>	<b>Use case .....</b>	<b>13</b>
<b>3</b>	<b>Development kit contents .....</b>	<b>15</b>
3.1	G.FIT libraries .....	15
3.2	G.FIT software development kit .....	15
3.2.1	Example SoC application (demo_gfit) .....	16
3.2.2	Fitness equipment console simulator PC application .....	16
3.2.3	Reference CIQ application .....	16
3.2.4	Reference iOS application .....	16
3.2.5	SimulANT+ .....	16
<b>4</b>	<b>G.FIT operation .....</b>	<b>17</b>
4.1	Fitness equipment (FE) states .....	17
4.1.1	G.FIT CONFIGURATION (OFF) .....	18
4.1.2	OFF .....	18
4.1.3	READY .....	18
4.1.4	IN USE .....	18
4.1.5	FINISHED (PAUSED) .....	19
4.2	Pairing heart rate monitors .....	20
4.2.1	Proximity pairing .....	20
4.2.2	Channel ID based pairing .....	21
4.3	G.FIT configuration .....	21
4.4	Event handling .....	22
4.4.1	Enabling events .....	23
4.4.2	Event responses .....	23
4.5	Custom events .....	26
<b>5</b>	<b>Connecting to G.FIT over BLE .....</b>	<b>27</b>
5.1	Interpreting resistance .....	28
5.2	G.FIT custom service .....	30
5.2.1	Custom control point characteristic .....	30
5.2.2	Custom data characteristic .....	31
5.2.3	Procedures .....	31
5.2.4	Setting the user and/or bicycle weight .....	31
5.2.5	Setting manufacturer specific data .....	33
5.2.6	Notification of updated manufacturer specific value .....	34
<b>6</b>	<b>Connecting to G.FIT over ANT+ FE-C .....</b>	<b>36</b>
6.1	Dual frequency FE-C .....	36
6.2	ANT+ manufacturer specific pages .....	36

6.2.1	Receiving custom events .....	36
6.2.2	Sending custom events .....	37
6.3	Using SimulANT+ to view transmitted data .....	37
6.4	Tracking FE-C channel with CIQ application .....	38
6.4.1	Installing CIQ application on a compatible Garmin watch .....	38
6.4.2	Opening the application .....	38
6.4.3	Connecting to the G.FIT ANT+ FE-C channel.....	38
6.4.4	Viewing live workout data.....	38
6.4.5	Saving user's workout session.....	38
6.4.6	Viewing workout data .....	38
6.5	FE-C Control Messages .....	38
<b>7</b>	<b>Serial interface .....</b>	<b>39</b>
7.1	Using extended serial messages .....	39
7.1.1	Commands (0xE2) .....	40
7.1.2	Requests (0xE1) .....	40
7.1.3	Responses/events (0xE0) .....	41
7.2	G.FIT response codes .....	41
7.3	Commands (0xE2).....	44
7.3.1	gfit_fep_set_state_ (0xD0) .....	44
7.3.2	gfit_fep_set_channel_configuration (0xD1).....	45
7.3.3	gfit_hrp_set_pairing_mode (0xD2) .....	47
7.3.4	gfit_hrp_set_inuse_scan_timeout (0xD3).....	48
7.3.5	gfit_hrp_set_pairing_proximity (0xD5) .....	49
7.3.6	gfit_hrp_pair_to_device (0xD6) .....	50
7.3.7	gfit_hrp_disconnect_device (0xD7) .....	52
7.3.8	gfit_enter_bootloader (0xD8).....	53
7.3.9	gfit_fep_set_inuse_adv_timeout (0xD9) .....	54
7.3.10	gfit_fep_set_equipment_type (0xE0) .....	55
7.3.11	gfit_fep_set_XX_data (0xE1).....	57
7.3.12	gfit_hrp_set_heart_rate (0xE2) .....	66
7.3.13	gfit_fep_set_device_id (0xE3) .....	67
7.3.14	fe_command_product_info (0xE4) .....	69
7.3.15	gfit_fep_send_command_update (0xE5) .....	71
7.3.16	fe_command_status_update (0xE6) .....	72
7.3.17	gfit_fep_abort_pending_commands (0xE7) .....	75
7.3.18	gfit_fep_set_max_resistance (0xE8) .....	76
7.3.19	gfit_fep_spin_down_ (0xE9) .....	77
7.3.20	gfit_fep_send_custom_event (0xB0) .....	85
7.3.21	gfit_fep_send_manufacturer_specific_page (0xB3) .....	86
7.3.22	gfit_fep_set_transmission_pattern (0xDA) .....	87

7.3.23	gfit_fep_set_developer_options (0xDB) .....	88
7.4	Requests (0xE1) .....	89
7.4.1	gfit_get_version_string (0xC0) .....	89
7.5	Events (0xE0).....	91
7.5.1	gfit_event_receive_heart_rate_ (0xB2) .....	91
7.5.2	Session command event (0xB4).....	95
7.5.3	gfit_event_set_target_ (0xB5) .....	96
7.5.4	gfit_event_set_simulation_parameters (0xB6) .....	97
7.5.5	gfit_event_set_user_data (0xB7) .....	98
7.5.6	Spin down Calibration Event (0xB8) .....	<b>Error! Bookmark not defined.</b>
7.5.7	gfit_event_entering_bootloader (0xB9) .....	100
7.5.8	gfit_event_ble_peripheral (0xBA) .....	100
7.5.9	gfit_event_custom (0xB1).....	101
7.6	Other ANT commands.....	102
7.6.1	Read ANT ID .....	102
7.6.2	RSSI calibration offset .....	103
<b>Appendix A - G.FIT SoC library.....</b>		<b>104</b>
A.1	Compiling the demo application .....	104
A.1.1	Library license key .....	104
A.1.2	Compiler compatibility.....	105
<b>Appendix B - Fitness equipment console simulator .....</b>		<b>106</b>
B.1	Introduction .....	106
B.2	Installing ANT USB interface board driver .....	106
B.3	Basic operation.....	108
B.3.1	Initial G.FIT setup .....	108
B.3.2	Turning on the simulated console.....	110
B.3.3	Pairing a heart rate monitor.....	110
B.3.4	Successfully paired heart rate monitor .....	111
B.3.5	Disconnecting a paired heart rate monitor .....	111
B.3.6	Starting a workout/session .....	111
B.3.7	Pausing/resuming a session .....	112
B.3.8	Ending a session .....	112
B.3.9	Turning off the simulated console .....	112
B.3.10	Enabling/disabling full screen mode.....	112
<b>8</b>	<b>Heading 1 - Template guidance.....</b>	<b>113</b>
8.1	Heading 2 .....	113
8.1.1	Heading 3.....	113
8.2	Headings use sentence style capitalization .....	113
8.2.1	Like this .....	113
8.2.2	Not Like This .....	113

8.3	Captions, tables, figures, and equations .....	113
8.4	Bullets and numbered text .....	114
8.5	Notes, bold text, italicized text, and document names .....	115
8.6	Watermarks .....	115
<b>9</b>	<b>Second 'Heading 1' heading .....</b>	<b>116</b>
<b>Appendix C - Sample appendix title .....</b>		<b>117</b>
C.1	Appendix H2.....	117
C.1.1	Appendix H3 .....	117
C.2	Table and Figure captions for use in Appendices .....	117

## List of Figures

Figure 2-1.	G.FIT use case illustration .....	14
Figure 4-1.	Fitness equipment state machine .....	17
Figure 4-2.	G.FIT pairing zones .....	21
Figure 4-3.	Order of G.FIT configuration commands .....	22
Figure 4-4.	Event handling (ANT) .....	24
Figure 4-5.	Event handling (BLE) .....	25
Figure 5-1.	Using BLE to set the user weight .....	33
Figure 5-2.	Using BLE to set a manufacturer specific value .....	34
Figure 5-3.	Notifying a display of a new manufacturer specific value .....	35
Figure 6-1.	Viewing received data pages on SimulANT+ .....	37
Figure 7-1.	Serial message general packet structure .....	39
Figure 7-2.	Extended serial message types .....	40
Figure 7-3.	Extended serial command .....	40
Figure 7-4.	Extended serial request .....	41
Figure 7-5.	Extended serial response .....	41
Figure 7-6.	Extended serial event .....	41
Figure 7-7.	Example: Setting FE metrics serial command (0xE1) .....	60
Figure 7-8.	Example: Successful spin down calibration process (ANT) .....	81
Figure 7-9.	Example: Successful spin down calibration process (BLE) .....	82
Figure 7-10.	Example: Cancelled spin down calibration process (ANT) .....	83
Figure 7-11.	Example: Cancelled spin down calibration process (BLE) .....	84
Figure 7-12.	Example: Requested spin down calibration process (BLE) .....	84
Figure 13.	Sequence diagram: proximity pairing mode hr events .....	92
Figure 14.	Sequence diagram: channel ID based pairing mode connection .....	93
Figure 7-15.	Sequence diagram: disconnecting from connected monitor device B .....	94

## List of Tables

Table 5-1.	Representations of resistance by context .....	28
------------	--	----

Table 5-2. Representations of resistance by data characteristic .....	29
Table 5-3. Example transmitted resistance values .....	29
Table 5-4. Service characteristics (primary service) .....	30
Table 5-5. Custom control point characteristic structure.....	30
Table 5-6. Op code and parameters .....	30
Table 5-7. G.FIT BLE response codes .....	30
Table 5-8. Custom data characteristic structure .....	31
Table 5-9. Op code and parameters .....	31
Table 6-1. G.FIT ANT+ manufacturer specific page format .....	36
Table 7-1. G.FIT response codes .....	41
Table 7-2. gfit_fep_set_state_ command message.....	44
Table 7-3. gfit_fep_set_state_ response message.....	45
Table 7-4. gfit_fep_set_channel_configuration command message .....	46
Table 7-5. gfit_fep_set_channel_configuration response message .....	46
Table 7-6. gfit_hrp_set_pairing_mode command message .....	47
Table 7-7. gfit_hrp_set_pairing_mode response message .....	47
Table 7-8. gfit_hrp_set_inuse_scan_timeout command message .....	48
Table 7-9. gfit_hrp_set_inuse_scan_timeout response message .....	48
Table 7-10. gfit_hrp_set_pairing_proximity command message .....	49
Table 7-11. gfit_hrp_set_pairing_proximity response message .....	49
Table 7-12. gfit_hrp_pair_to_device command message .....	50
Table 7-13. ANT and BLE channel IDs .....	50
Table 7-14. gfit_hrp_pair_to_device response message.....	51
Table 7-15. gfit_hrp_disconnect_device command message.....	52
Table 7-16. gfit_hrp_disconnect_device response message.....	52
Table 7-17. gfit_enter_bootloader command message.....	53
Table 7-18. gfit_enter_bootloader response message .....	53
Table 7-19. gfit_fep_set_inuse_adv_timeout command message .....	54
Table 7-20. gfit_fep_set_inuse_adv_timeout response message.....	54
Table 7-21. gfit_fep_set_equipment_type command message .....	55
Table 7-22. gfit_fep_set_equipment_type response message .....	56
Table 7-23. gfit_fep_set_XX_data command message .....	58
Table 7-24. gfit_fep_set_XX_data response message .....	58
Table 7-25. Field lengths .....	59
Table 7-26. Treadmill field IDs and field lengths .....	61
Table 7-27. Trainer/indoor bike field IDs and field lengths .....	62
Table 7-28. Elliptical/cross trainer field IDs and field lengths.....	63
Table 7-29. Rower field IDs and field lengths .....	64
Table 7-30. Step climber field IDs and field lengths .....	65
Table 7-31. gfit_hrp_set_heart_rate command message.....	66



Table 7-32. gfit_hrp_set_heart_rate response message.....	66
Table 7-33. Device ID to channel ID conversion .....	67
Table 7-34. gfit_fep_set_device_id command message .....	67
Table 7-35. gfit_fep_set_device_id response message .....	68
Table 7-36. fe_command_product_info command message .....	69
Table 7-37. fe_command_product_info response message.....	69
Table 7-38. Product information types and values .....	70
Table 7-39. gfit_fep_send_command_update command message .....	71
Table 7-40. gfit_fep_send_command_update response message .....	71
Table 7-41. fe_command_status_update command message.....	72
Table 7-42. fe_command_status_update response message.....	72
Table 7-43. Status type.....	73
Table 7-44. Status parameters .....	74
Table 7-45. gfit_fep_abort_pending_commands command message.....	75
Table 7-46. gfit_fep_abort_pending_commands response message.....	75
Table 7-47. gfit_fep_set_max_resistance command message .....	76
Table 7-48. gfit_fep_set_max_resistance response message .....	76
Table 7-49. gfit_fep_spin_down_ command message .....	77
Table 7-50. gfit_fep_spin_down_ payload.....	79
Table 7-51. gfit_fep_spin_down_ response message.....	80
Table 7-52. gfit_fep_send_custom_event command message.....	85
Table 7-53. gfit_fep_send_custom_event response message.....	85
Table 7-54. gfit_fep_send_manufacturer_specific_page command message.....	86
Table 7-55. gfit_fep_send_manufacturer_specific_page response message.....	86
Table 7-8. gfit_fep_set_transmission_pattern command message.....	87
Table 7-9. gfit_fep_set_transmission_pattern response message.....	87
Table 7-56. gfit_get_version_string request message.....	89
Table 7-57. gfit_get_version_string response message .....	89
Table 7-58. gfit_event_receive_heart_rate_ event message.....	91
Table 7-59. Session command event message.....	95
Table 7-60. gfit_event_set_target_ event message.....	96
Table 7-61. Target types .....	96
Table 7-62. gfit_event_set_simulation_parameters event message .....	97
Table 7-63. gfit_event_set_user_data event message .....	98
Table 7-64. Spin down calibration event message .....	99
Table 7-65. gfit_event_entering_bootloader event message .....	100
Table 7-66. gfit_event_custom event message.....	101
Table 7-67. Request to retrieve ANT ID.....	102
Table 7-68. ANT ID read response message.....	102
Table 7-69. RSSI calibration offset message .....	103

Table 7-70. RSSI calibration offset response message.....	103
Table 8-1. Table captions go over table .....	113

## List of Equations

Equation 5-1. Interpreting transmitted resistance values.....	29
Equation 7-1. Calculating gfit_fep_set_XX_data length.....	59
Equation 8-1. Deriving the scale factor .....	113

## 1 Support

The D52 ANT SoC module series (including G.FIT) uses the nRF52832 from Nordic Semiconductor. You can seek technical support from Nordic Semiconductor, [www.nordicsemi.com](http://www.nordicsemi.com). G.FIT application support can be sought from Garmin Canada via [www.thisisant.com](http://www.thisisant.com).

### 1.1 G.FIT technical references

Refer to current versions of the listed documents and software. To ensure you are using the current versions, check [www.thisisant.com](http://www.thisisant.com), [www.bluetooth.com](http://www.bluetooth.com), [www.nordicsemi.com](http://www.nordicsemi.com), or <http://infocenter.nordicsemi.com/index.jsp> or contact your Garmin Canada representative. User registration may be required.

#### Documents:

- a. G.FIT User Guide and Specification
- b. G.FIT Firmware Updater Application Note
- c. G.FIT and Premium Module Manufacturing Considerations Application Note
- d. nRF52832 Product Specification, Nordic Semiconductor
- e. nRF52 Series Compatibility Matrix, Nordic Semiconductor Infocenter
- f. nRF52832 Objective Product Specification, Nordic Semiconductor
- g. nRF52832 S332 SoftDevice Specification, Dynastream Innovations
- h. nRF52 Development Kit Documentation, Nordic Semiconductor Infocenter
- i. ANT SoC Module Starter Kit User Manual, Dynastream Innovations
- j. ANT Message Protocol and Usage, Dynastream Innovations
- k. Interfacing with ANT General Purpose Chipsets and Modules, Dynastream Innovations
- l. Application Note: Interpreting RF Radiation Patterns, Dynastream Innovations
- m. Bluetooth Fitness Machine Service/Profile, Bluetooth SIG
- n. Bluetooth Core Specification, Bluetooth SIG

#### Software:

- a. G.FIT SDK – simulators, G.FIT, iOS, Connect IQ demo code
- b. G.FIT Library – G.FIT Library files, S332 ANT/Bluetooth SoftDevice, Evaluation G.FIT Network Processor binary
- c. S332 nRF52832 SoftDevice, Garmin Canada
- d. nRF5 SDK, Nordic Semiconductor
- e. ANTwareII – a system testing and debugging tool, Dynastream Innovations
- f. SimulANT+

Design models (all apply to G.FIT modules):

- a. D52Q Altium library, Dynastream Innovations
- b. D52Q module STEP model, Dynastream Innovations
- c. D52M Altium library, Dynastream Innovations
- d. D52M module STEP model, Dynastream Innovations

## 2 Use case

G.FIT is a turnkey dual-protocol ANT/Bluetooth® low energy (BLE) solution for wireless fitness equipment and smart bike trainers, optimized for group training environments with 50+ fitness devices and multiple receivers.

As group fitness training grows in demand, many challenges to a good experience are presenting themselves:

- Gym and studio environments have lots of WiFi (and other) interference.
- Available heart rate monitors from multiple manufacturers and wireless standards.
- Gym users and instructors are mostly non-technical users.
- Gym patrons bring in their own personal devices (e.g. watches and phones).
- Gyms have many types of equipment from multiple manufacturers to maintain, many of them with different operating procedures.



With these challenges in mind, Garmin Canada Inc. has evolved individual fitness machines and smart bike trainers to incorporate wireless standards group fitness capabilities. The G.FIT solution is the result of that evolution, and solves the challenges of group fitness:

- G.FIT works around WiFi interference to ensure large groups of **50+ devices can connect** concurrently.
- G.FIT solves pairing and setup difficulties with dual protocol support for both **ANT+ and BLE heart rate monitors**.
- G.FIT makes it easy for non-technical users to pair by using **proximity pairing and list pairing** features.
- G.FIT solves personal device pairing by **re-broadcasting ANT+ or BLE heart rate data** and works with **Bluetooth FTMS (including Control Point) and the ANT+ FE-C Device Profile** so that people can record entire workout sessions on their own personal devices.
- G.FIT enables technology maintenance with both **built-in support for easy wireless updates and the addition of custom tailored features and controls** using the G.FIT SDK.

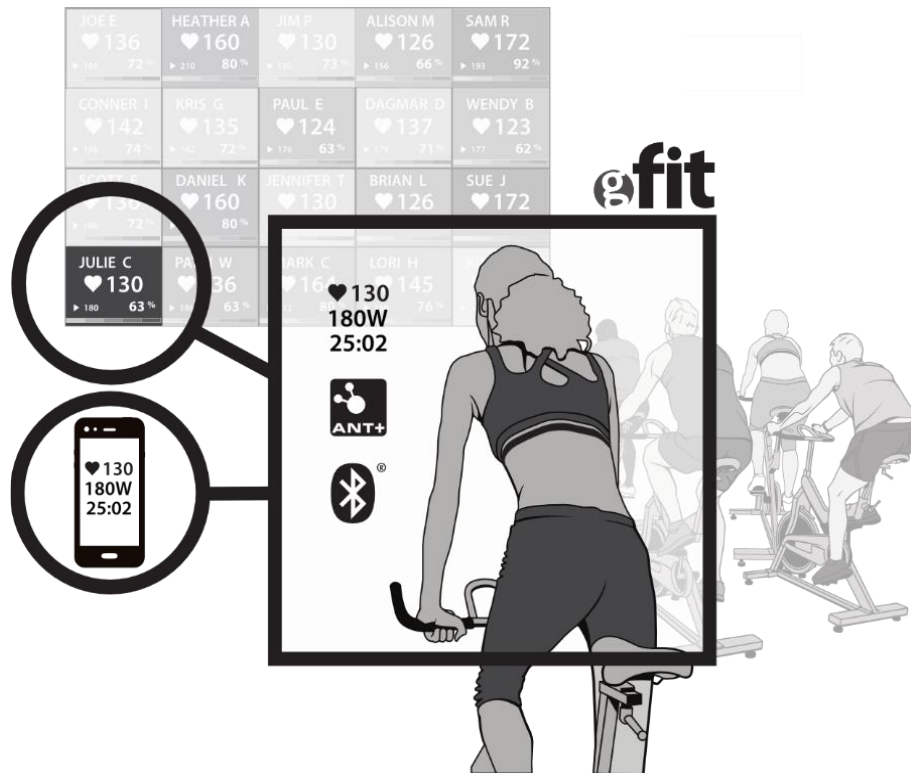


Figure 2-1. G.FIT use case illustration – fitness equipment



Figure 2-2 - G.FIT use case illustration - smart bike trainer

### 3 Development kit contents & software downloads

The D52 starter kit (D52DK2) contains all the necessary equipment needed to start G.FIT development. See [https://www.thisisant.com/developer/components/gfit/#873\\_tab](https://www.thisisant.com/developer/components/gfit/#873_tab).

- **G.FIT compatible module:** D52QSKM6IA-A Module
- **USB Interface Board:** The USB interface board is required so that the G.FIT module can be used with the provided PC application. See section B.2 for more details on installing the necessary drivers.
- **USB-m:** The USB-m is required so that the developer can view the data being transmitted by the G.FIT module. The USB-m can be used with SimulANT+ to connect to one of the G.FIT module's ANT+ FE-C channels.

G.FIT development is aided and supported by two download packages:

- **G.FIT libraries package:** a commercially licensed packages which contains static libraries, API headers, evaluation network processor and ANT/BLE S332 SoftDevice.
- **G.FIT software development kit (SDK):** a package licensed under the ANT+ Shared Source license that contains an example SoC application, FE simulator, and reference apps for Garmin Connect IQ and iOS.

The following sections provide a more detailed overview of downloaded G.FIT development collateral, which can all be obtained here: [https://www.thisisant.com/developer/components/gfit/#874\\_tab](https://www.thisisant.com/developer/components/gfit/#874_tab)

#### 3.1 G.FIT libraries

The content of the G.FIT libraries package is as follows:

- **G.FIT static library:** Pre-built library, compiled in GCC, that can be used by custom applications using G.FIT modules as an SoC. The static library, libgfit.a is available in the bin folder of the G.FIT\_Library zip file.
- **API headers:** These files, available in the inc\ folder of the G.FIT\_Library zip file, need to be included by your application to access the G.FIT static library, and provide documentation on the available API calls.
  - gfit\_interface.h: Function definitions and API documentation.
  - gfit\_defines.h: Constants, enumerations and data structures.
- **G.FIT Evaluation Network Processor:** Pre-built hex file for the G.FIT network processor intended for evaluation purposes. This hex file can be used only in D52 development modules (D52QSKM6IA-A), e.g., for use with the D52 ANT SoC Module Series Starter Kit (D52DK2).
- **ANT/BLE S332 SoftDevice**

#### 3.2 G.FIT software development kit

The G.FIT software development kit includes:

- Example SoC application (demo\_gfit)
- Fitness equipment simulator PC application
- Reference CIQ application
- Reference iOS application

- SimulANT+ (downloaded separately)

These are described in the following sections.

### **3.2.1 Example SoC application (demo\_gfit)**

Shows how to integrate the G.FIT library into an application. The demo implements a simple fitness equipment console with three buttons: Start (button A), Stop (button B) and Disconnect HR (button C). The demo shows how to initialize and configure G.FIT, trigger state transitions, and update data transmitted over the ANT+ FE-C and BLE FTMS profiles. Paired HR values and control commands are output over Segger RTT. The project file for the example application is located under apps\demo\_gfit\armgcc. See section A.1 for more details on compiling the demo.

### **3.2.2 Fitness equipment console simulator PC application**

The fitness equipment console simulator PC application simulates a fitness equipment console that uses G.FIT. See section Appendix B - Fitness equipment console simulator for more details.

### **3.2.3 Reference CIQ application**

Garmin's Connect IQ platform supports creating apps with custom ANT+ channels that can be used to connect Garmin's wearable and bike accessories with G.FIT enabled fitness equipment. If the user has a Connect IQ compatible Garmin watch/bike computer, they can track their workout and save that activity for viewing on Garmin Connect over the ANT+ FE-C channel. See section 6.4 for more details.

### **3.2.4 Reference iOS application**

iOS devices supporting BT v4.0 and above are compatible with G.FIT over a BLE connection using the FTMS (Fitness Machine Service), HRS (Heart Rate Service), and DIS (Device Information Service). An iOS Application (including source code) is available in the download package to provide an example for scanning, pairing and connecting with a G.FIT enabled fitness equipment device.

### **3.2.5 SimulANT+**

SimulANT+ can be used to connect to one of the G.FIT ANT+ FE-C channels. See section 6.3 for more details. SimulANT+ is available for download at <https://www.thisisant.com/developer/resources/downloads/>. Before downloading SimulANT+ you will need to create an account on <http://www.thisisant.com>.



## 4 G.FIT operation

### 4.1 Fitness equipment (FE) states

G.FIT has been designed so that it complies with the states defined in the ANT+ Fitness Equipment Device Profile. The ANT+ Fitness Equipment Device Profile defines four states that the fitness equipment can be in. Any FE device using G.FIT shall move between these states in response to user interaction as shown in Figure 4-1 below. These state transitions are completed using the `gfit_fep_set_state_` command (section 7.3.1).

When G.FIT is power cycled, it will enter a configuration state. In this configuration stage the application shall then configure G.FIT for the type of fitness equipment and desired fitness equipment behaviour. To move out of configuration mode, mode to state READY. When the fitness equipment is not used it can remain in a low power state, the OFF state. Typically, user activity such as pedaling will wake up the FE and activate the user interface. This would be an example of when to make the transition to the READY state. Some FE may never sleep and would always default to the READY state when not in use.

From READY, a button press or sustained activity begins a workout session and puts the FE into the IN USE state. At the end of the workout a button press or user inactivity may end the session and put the FE into the FINISHED state. From FINISHED, another button press or elapsed time with no activity will return the FE to the READY or OFF state.

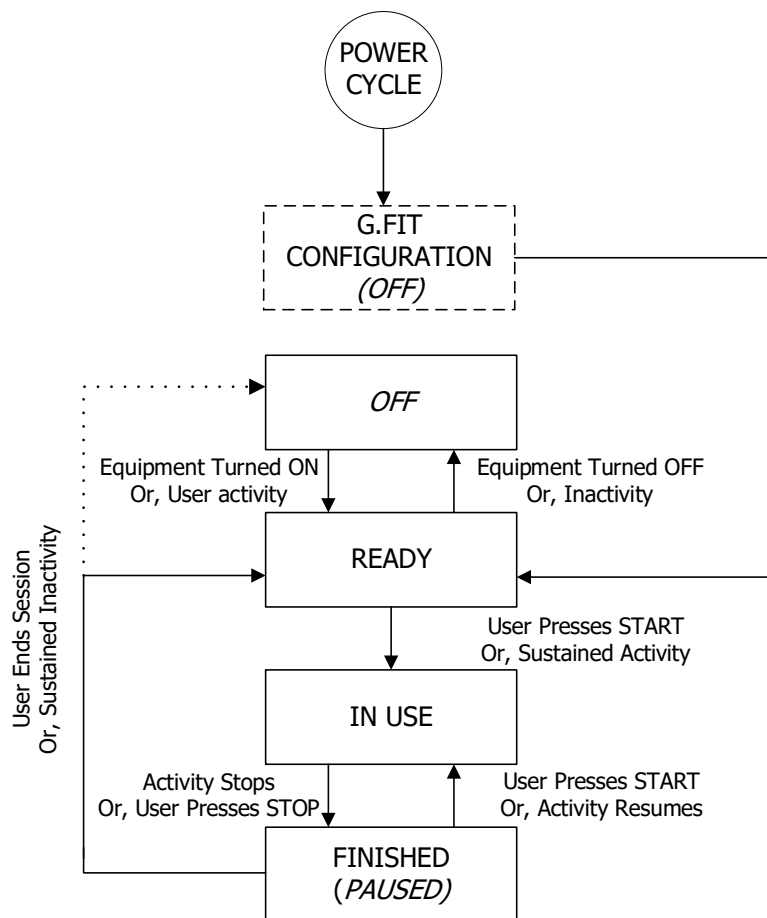


Figure 4-1. Fitness equipment state machine

#### **4.1.1 G.FIT CONFIGURATION (OFF)**

Each time G.FIT is power cycled, its settings are reset and it enters a special OFF state; during this initial OFF state, G.FIT is in configuration mode. In this initial OFF state, the application MCU configures G.FIT to meet the desired behaviour for the given fitness equipment and required use case. All commands that are used to configure G.FIT must be sent in this state. There are some minimum requirements that G.FIT requires before transitioning from OFF to READY, see section 4.3 for the minimum G.FIT configuration sequence.

#### **4.1.2 OFF**

This state is used to conserve power and it is recommended that G.FIT be put into this state after sustained user inactivity.

##### **4.1.2.1 Heart rate receive channels**

Both ANT+ and BLE heart rate scan channels are closed.

##### **4.1.2.2 Fitness equipment transmit channels**

Both the ANT+ FE-C channel and BLE FTMS peripheral transmit channels are closed.

##### **4.1.2.3 Transitioning from OFF**

When transitioning from OFF to READY, the developer should ensure that all channels have finished closing (see section [4.1.2.1](#), and section [4.1.2.2](#)) before making the transition.

#### **4.1.3 READY**

This state should be entered when there is initial user activity or when a user has recently ended a session. This state opens scanning channels that search for BLE and ANT+ heart rate monitors, allowing the user to connect their heart rate monitors as they prepare for a workout. Since G.FIT is scanning during this time, avoid running additional scans on other channels or doing flash writes. It also turns on the ANT+ FE-C channel and the BLE FTMS peripheral which transmits data describing the user's workout.

##### **4.1.3.1 Heart rate receive channels**

ANT+ and BLE heart rate scan channels are opened and will remain open while in state READY. If a heart rate monitor is paired to G.FIT, a dedicated channel is opened to establish a connection with that heart rate monitor, both ANT+ and BLE heart rate scan channels will remain open and scan for heart rates when connected to a heart rate monitor.

##### **4.1.3.2 Fitness equipment transmit channels**

The ANT+ FE-C channel is opened and begins transmitting and BLE FTMS peripheral begins advertising. Both channels will remain open and continue transmitting while in state READY. If a display is paired to G.FIT while in state IN USE, a connection is established with the display and the BLE FTMS peripheral stops advertising.

#### **4.1.4 IN USE**

This state should be entered when the user has sustained activity or has indicated a desire to start a workout session by pressing the start button.

##### **4.1.4.1 Heart rate receive channels**

The heart rate receive channels have the following behaviour depending on the user interaction.

###### **4.1.4.1.1 No heart rate monitor connected**

If G.FIT moves to state IN USE and there are no heart rate monitors connected, then the ANT+ and BLE heart rate scan channels will remain open for 30 seconds (by default, this can be adjusted using HR\_SetInUseScanTimeout, see section 7.3.4). After 30 seconds the ANT+ and BLE heart rate scan channels

will close preventing any heart rate monitors from pairing to the G.FIT. Avoid running additional scans on other channels or doing flash writes while G.FIT is still scanning.

#### **4.1.4.1.2 Heart rate monitor paired while IN USE**

If a heart rate monitor is paired to the G.FIT module while in state IN USE, a dedicated channel is opened to establish a connection with that heart rate monitor. After the connection is established to the heart rate monitor, both ANT+ and BLE heart rate scan channels will close immediately.

#### **4.1.4.1.3 Heart rate monitor paired while READY**

If a heart rate monitor was paired to the G.FIT while it was in state READY, when the G.FIT transitions to state IN USE both ANT+ and BLE heart rate scan channels will close immediately.

#### **4.1.4.2 Fitness equipment transmit channels**

The ANT+ FE-C channel remains open and continues transmitting while in state IN USE. The BLE FTMS peripheral has the following behaviour depending on the user's interactions.

##### **4.1.4.2.1 No display connected to peripheral**

If G.FIT moves to state IN USE and there is no display connected to the BLE FTMS peripheral, the peripheral will continue to advertise for 30 seconds (by default, this can be adjusted using `HR_SetInUseAdvertisingTimeout`, see section 7.3.9). After 30 seconds the BLE FTMS peripheral will stop advertising preventing any displays from pairing to G.FIT.

##### **4.1.4.2.2 Display connected while IN USE**

If a display is paired to G.FIT while in state IN USE, a connection is established with the display and the peripheral stops advertising.

##### **4.1.4.2.3 Display connected while READY**

If a display is paired to G.FIT while in state READY, the BLE FTMS peripheral will not advertise in state IN USE.

#### **4.1.5 FINISHED (PAUSED)**

This state should be entered when a user presses stop or the user's activity stops. Sustained inactivity should cause G.FIT to transition state OFF or READY.

#### **4.1.5.1 Heart rate receive channels**

The heart rate receive channels have the following behaviour depending on the user interaction.

##### **4.1.5.1.1 No heart rate monitor connected**

If G.FIT moves to state FINISHED while there are no heart rate monitors connected, and both the ANT+ and BLE heart rate scan channels are still open (the time spend in state IN USE was less than the `InUseScanTimeout`), the `InUseScanTimeout` will reset and the scan channels will remain open for the timeout duration.

##### **4.1.5.1.2 Heart rate monitor paired**

If a heart rate monitor was paired to the G.FIT when moving to state FINISHED, both ANT+ and BLE heart rate scan channels will remain closed. The G.FIT will maintain the connection to the heart rate monitor while in state FINISHED.

#### **4.1.5.2 Fitness equipment transmit channels**

The ANT+ FE-C channel remains open and continues transmitting while in state FINISHED. The BLE FTMS peripheral has the following behaviour depending on the user's interactions.

#### **4.1.5.2.1 No display connected to peripheral**

If G.FIT moves to state FINISHED while there is no display connected and the BLE FTMS peripheral is still advertising (the time spend in state IN USE was less than the InUseAdvertisingTimeout), the InUseAdvertisingTimeout will reset and the peripheral will continue to advertise for the timeout duration.

#### **4.1.5.2.2 Display connected**

If a display is paired to G.FIT when moving to state FINISHED, the BLE FTMS peripheral will not advertise in state FINISHED.

### **4.2 Pairing heart rate monitors**

G.FIT provides two methods of connecting to heart rate monitors to best fit the required use case:

- Proximity pairing allows the user to pair a heart rate monitor by physically moving the heart rate monitor to within the 'pairing zone' of the G.FIT module.
- Channel ID based pairing gives the application the flexibility to select which heart rate monitor to pair with; allowing the user to manually select the desired heart rate monitor by scrolling through all available heart rate monitors seen by G.FIT.

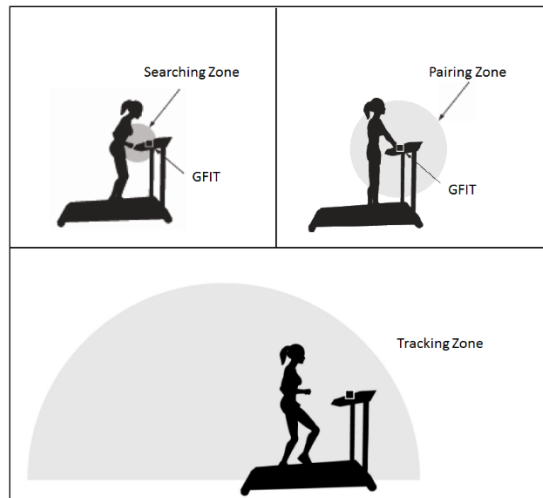
#### **4.2.1 Proximity pairing**

G.FIT will pair with a heart rate monitor automatically based on the proximity to the G.FIT module. In order to successfully pair with a heart rate monitor, the received signal strength received from the heart rate monitor must maintain a specified signal strength during each phase of the connection process. There are three phases in the connection process: searching, pairing, and tracking. See Figure 4-2 for typical zone sizes.

**Searching:** This phase of the connection process searches for possible heart rate monitor to connect to. It has the strictest RSSI threshold so that a heart rate monitor must be brought in close proximity to the G.FIT module. Once a heart rate monitor is detected in the searching phase it then advances to the Pairing phase.

**Pairing:** This phase of the connection process ensures that the heart rate monitor found in the searching phase maintains a close proximity to the G.FIT module. This helps prevent accidental connections to other heart rate monitor. This threshold is less strict than searching but should be defined so that the user must keep the heart rate monitor in the pairing zone during the initial connection process. Once G.FIT receives the sufficient number or packets from the heart rate monitor within the defined signal strength, a connection is established and the heart rate monitor advances to the tracking phase.

**Tracking:** The tracking phase is used to maintain a connection to the heart rate monitor while the user is using the fitness equipment.



**Figure 4-2. G.FIT pairing zones**

#### 4.2.1.1 RSSI thresholds

G.FIT provides default RSSI thresholds that should work in most use cases. However, it is **highly recommended** that the settings be adjusted to account for difference in module installation, equipment type and desired pairing experience. The application should define RSSI values to have the following behaviour:

**Searching:** The minimum RSSI before heart rate monitor is considered for pairing. Values below this threshold will not be considered in the pairing algorithm.

**Pairing:** The signal strength required for G.FIT to pair to the heart rate monitor. The heart rate monitor must maintain a signal strength greater than pairing for the duration of the pairing algorithm in order to successfully pair to G.FIT.

The RSSI values shall be set so that **Pairing < Searching**.

#### 4.2.2 Channel ID based pairing

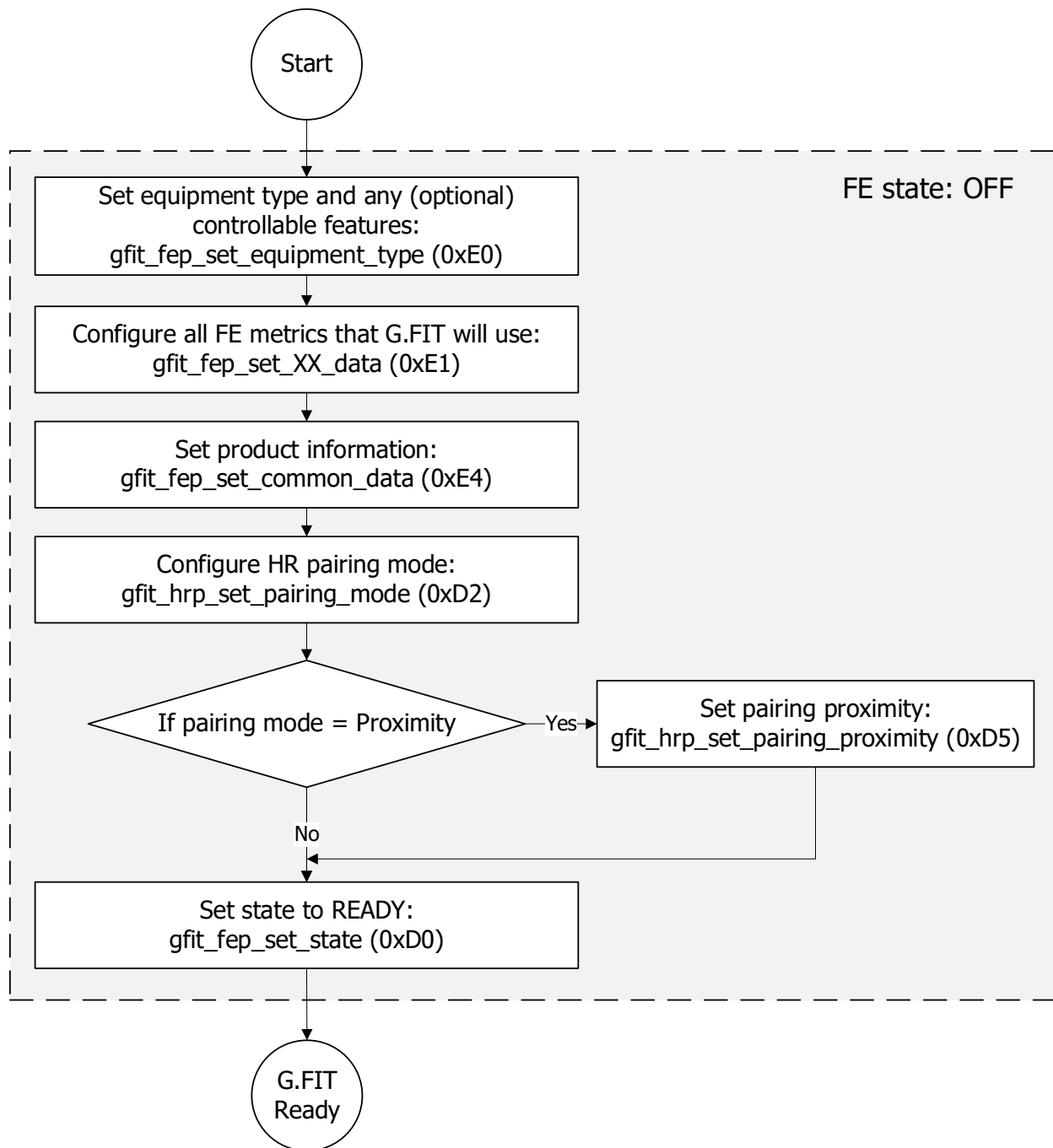
In channel ID pairing, G.FIT generates HR events (see section 7.5.1) for every heart rate message received on the ANT and BLE scan channels, and provides a method of connecting to the desired heart rate monitor using the `gfit_hrp_pair_to_device` serial command (see section 7.3.6). This gives the application the flexibility to pair to any heart rate monitor within range of the G.FIT module. This method of pairing can be used to scan for heart rate monitors in the area and allow the user to select their own heart rate monitor from a list of devices on a screen.

Refer to section 7.5.1 for sequence diagrams of HR Events in Proximity and Channel ID based pairing modes.

**Note:** Upon receiving a heart rate message on the ANT+ scan channel or the ANT+ paired channel, G.FIT will include a calibrated RSSI value in the generated HR event. This will occur in both proximity and channel ID based pairing modes.

### 4.3 G.FIT configuration

G.FIT needs to be configured while in the initial OFF state. Figure 4-3 shows the order in which G.FIT should be configured.



**Figure 4-3. Order of G.FIT configuration commands**

#### 4.4 Event handling

G.FIT will process specific messages received over ANT and BLE and will generate protocol agnostic events that can be processed by the application. Refer to section 0 for available G.FIT events.

#### 4.4.1 Enabling events

In order to generate a specific event, G.FIT must be configured as described in the 'Depends on' section for each event. For example, to enable HR events (described in 7.5.1), HR pairing must be enabled (see section 7.3.3).

If a particular event is disabled, when messages corresponding to that event are received over ANT and BLE, G.FIT will automatically generate the appropriate response for each protocol to indicate that particular feature is not supported. In the case of ANT, the message will be ignored. In the case of BLE (FTMS or G.FIT custom event), a response indicating 'Op Code Not Supported' will be sent to the display.

#### 4.4.2 Event responses

Most events require the application to issue a response, which indicates to a display whether the command was handled successfully or not. If the application receives an event (see section 0) where the 'Requires a Response' section indicates that a response is required, the application **shall** send a response using the `gfit_fep_send_command_update` (0xE5) command at any time after receiving the events. Sending the response is essential to completing the procedure when the events are triggered via BLE and failing to send a response will result in not being able to initiate new events via BLE until a response is sent.

If a particular controllable feature is enabled, all related events must be handled by the application and operations not supported by the fitness equipment **shall** be rejected. For example, if custom events are enabled, but the application only supports one particular op code, the event handler should process the supported op code, and then send a `gfit_fep_send_command_update` with Command Response = Pass. If an unsupported op code is received, the event handler should send a `gfit_fep_send_command_update` with Command Response = Fail.

No event handling timeouts are implemented by G.FIT – if the application fails to perform the requested action in a timely manner, the application is responsible for implementing a timeout and sending a failure response when the timeout expires.

The response is transmitted to the display through the mechanism defined for each protocol; a response is only transmitted on the protocol where the event originated (i.e., if an event is triggered via an ANT message, the response for it will only be transmitted via ANT). For ANT, command responses are sent through requestable data page 71 (Command Status), as shown in Figure 4-4. For BLE, command responses are sent through an indication for the Fitness Machine Control Point characteristic, using the Response Code op code, as shown in Figure 4-5. As can be seen in Figure 4-4 and Figure 4-5, application handling of the event and its response is the same regardless of which protocol the event originated in.

Optionally, after responding to an event, the fitness equipment can use the `fe_command_status_update` (0xE6) command to send an update with the new status. For example, as show in Figure 4-4 and Figure 4-5, after receiving a Set Target Event and accepting the command, the application can send a status update to indicate to the display the new target setting. For ANT+, status updates are available through requestable data pages. For BLE, status updates are sent through a notification for the Fitness Machine Status characteristic. Status updates can also be triggered independently from an event, e.g., if a setting is configured through the UI of the fitness equipment.

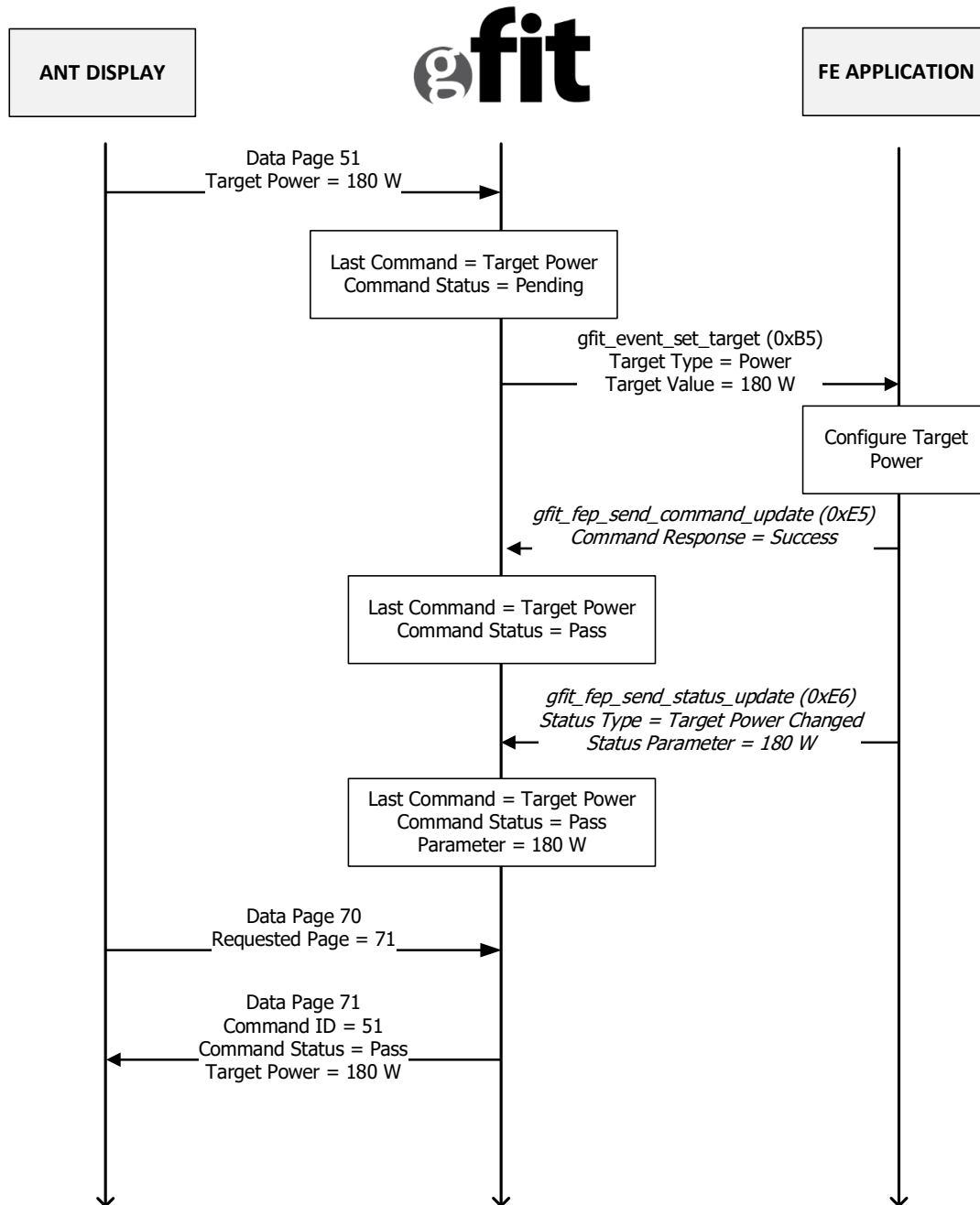


Figure 4-4. Event handling (ANT)



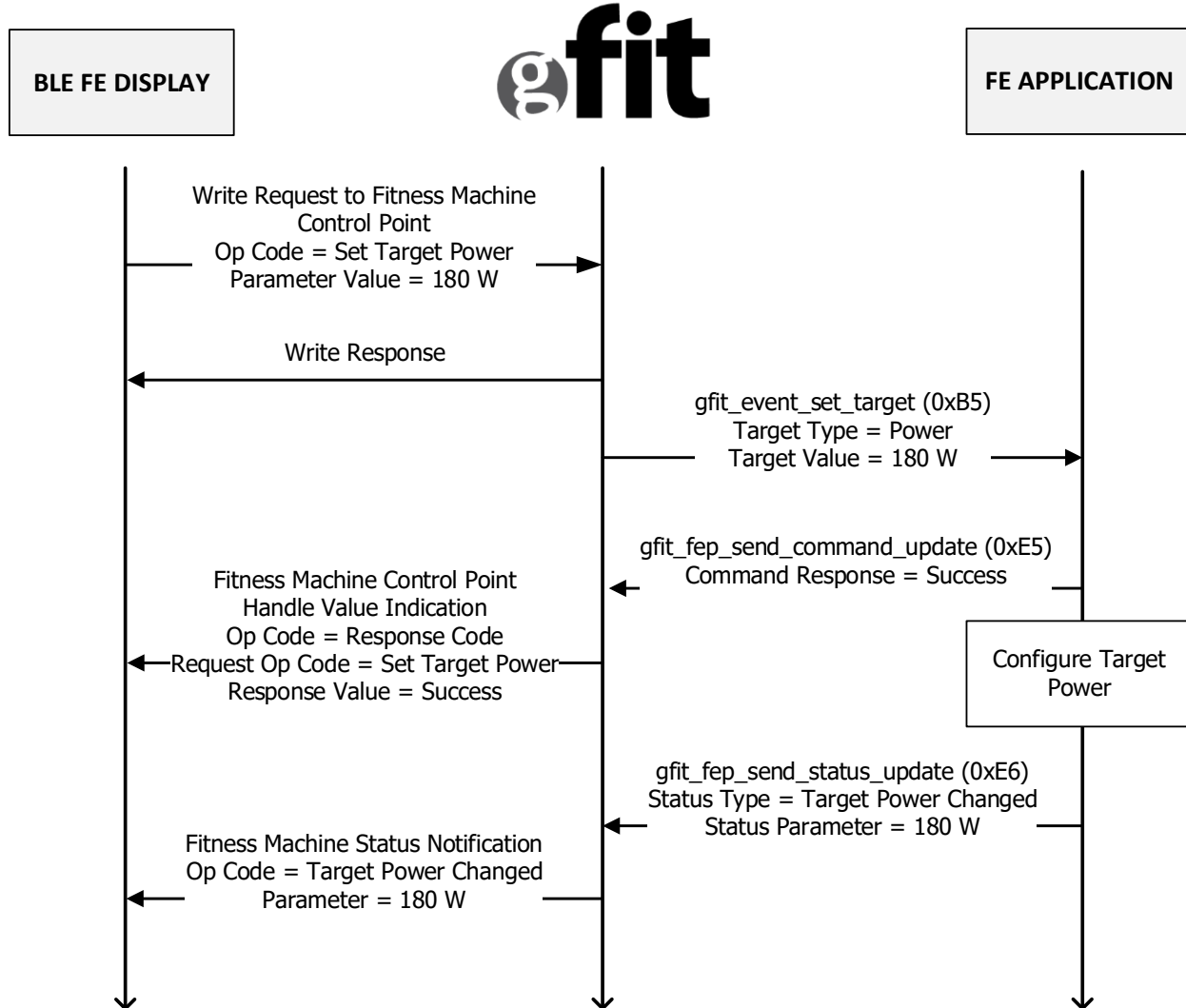


Figure 4-5. Event handling (BLE)

## 4.5 Custom events

G.FIT allows bidirectional custom message exchange between the fitness equipment and display over both ANT+ (through manufacturer specific pages, section 6.2) and BLE (through the G.FIT Custom Service, section 0).

Custom events can be used to transfer manufacturer specific information between the two devices. Custom events are transmitted over the same channel as FE data; care must be taken to not starve the FE data stream by sending manufacturer specific updates too frequently.

To send custom events, use the `gfit_fep_send_custom_event` message described in section 0.

## 5 Connecting to G.FIT over BLE

The following services and characteristics are available over BLE:

- Fitness Machine Service (FTMS)
  - UUID: 0x1826
  - Characteristics Present:
    - One of:
      - Treadmill Data (UUID: 0x2ACD)
      - Step Climber Data (UUID: 0x2ACF)
      - Cross Trainer Data (UUID: 0x2ACE)
      - Rower Data (UUID: 0x2AD1)
      - Indoor Bike Data (UUID: 0x2AD2)
    - Fitness Machine Features (UUID: 0x2ACC)
    - Resistance Level Range (UUID: 0x2AD6)
    - Power Range (UUID: 0x2AD8)
    - Inclination Range (UUID: 0x2AD5)
    - Heart Rate Range (UUID: 0x2AD7)
    - Control Point (UUID: 0x2AD9)
    - Fitness Machine Status (UUID: 0x2ADA)
- Device Information Service (DIS)
  - UUID: 0x180A
  - Characteristics Present:
    - Manufacturer Name String (UUID: 0x2A29)
    - Model Number String (UUID: 0x2A24)
    - Firmware Revision String (UUID: 0x2A26)
    - Software Revision String (UUID: 0x2A28)
    - Hardware revision (UUID: 0x2A27)
      - Only present if a value was set on the fitness equipment.
- Heart Rate Service (HRS) – only present if HR pairing is enabled
  - UUID: 0x180D
  - Heart Rate Measurement (UUID: 0x2A37)

- Custom G.FIT Service
  - UUID: 00001001C04266BA133590118F542C77 (not included in advertising data)
  - Characteristics Present:
    - Control Point (UUID: 8EC92001F3154F609FB8838830DAEA50)
    - Custom Data (UUID: 8EC92002F3154F609FB8838830DAEA50)

Unless otherwise noted above, all characteristics and services will be present including FTMS control point, FTMS range characteristics, etc. A controller connecting to G.FIT over BLE should check the Fitness Machine Features characteristic to determine what features are enabled before attempting to control the device. It is not possible for an application to add more BLE services or to add characteristics to the existing services.

The ranges configured in the range characteristics (UUID 0x2AD5 to 0x2AD8) have fixed values for all devices using G.FIT and cannot be modified. They indicate the allowable minimum and max values that can be written to the control point characteristic.

Controllers wishing to write to any control point characteristic must pair to G.FIT and subscribe to notifications from that characteristic before issuing any commands. See the control point section in the FTMS Service Specification documentation for more information. The Control Point characteristic for the Custom G.FIT Service follows the same requirements and operates in much the same way.

Most of the characteristics in these services support notifications, which can be enabled by a display by writing 1 to the Client Characteristic Configuration Descriptor (CCCD). Control point characteristics in FTMS and the Custom G.FIT service support indications, which can be enabled by writing 2 to the CCCD. G.FIT implements automatic handling of notifications and indications, including automatic retries for the latter.

All BLE services share a single connection, so it is not possible to send indications from two services at the same time.

BLE characteristics are generally little endian. Refer to the [FTMS Specification Document](#) from the Bluetooth SIG as well as the [Bluetooth SIG FTMS Assigned Numbers webpage](#) for specific information about endianness and data formatting.

## 5.1 Interpreting resistance

G.FIT uses units of 0.5% for all resistance values, as does ANT+. The resistance values transmitted using BLE FTMS use different resolutions, so this section explains how to interpret the values and provides examples.

**Table 5-1. Representations of resistance by context**

Context	Data type	Representation
G.FIT serial messages	sint16	Signed half percent
ANT+	int8	Unsigned half percent
BLE	sint16	Varies, see Table 5-2

The *Bluetooth Fitness Machine Service/Profile* specifies that resistance values should be transmitted as unitless values as described in Table 5-2. The range for all these values is transmitted as the 'supported resistance range'. Because multiple data characteristics can apply at the same time (for example indoor bike data, FTMS control point, and FTMS fitness machine status) and there is only one place to specify the range, G.FIT transmits the same range for all data characteristics: -2000 to +2000. This value is interpreted differently for each data characteristic based on their defined resolution, so the effective range varies.

**Table 5-2. Representations of resistance by data characteristic**

Data characteristic	ID	Representation	Effective range
Indoor bike data	0x2AD2	Unitless with a resolution of 1	-2000 to +2000
Rower data	0x2AD1	Unitless with a resolution of 1	-2000 to +2000
Cross trainer data	0x2ACE	Unitless with a resolution of 0.1	-200 to +200
FTMS control point Op Code: Set Target Resistance Level (Op Code: 0x04)	0x2AD9	Unitless with a resolution of 0.1	-200 to +200
FTMS control point supported resistance range	0x2AD6	Unitless with a resolution of 0.1	-200 to +200
FTMS fitness machine status Op Code: Target Resistance Level Changed (Op Code: 0x07)	0x2ADA	Unitless with a resolution of 0.1	-200 to +200

To obtain the resistance from the transmitted value, apply the following equation:

$$\frac{\text{resistance percentage}}{100} = \frac{\text{transmitted value} * \text{increment}}{\text{range max} * \text{range minimum increment}}$$

**Equation 5-1. Interpreting transmitted resistance values**

Where: Range max = 2000, Range minimum increment = 0.1, Increment = 0.1 or 1

Therefore

$$\text{transmitted value} = \frac{\text{resistance percentage} * \text{range max} * \text{range minimum increment}}{100 * \text{increment}}$$

$$\text{transmitted value} = \frac{\text{resistance percentage} * 200}{100 * \text{increment}}$$

Applying this equation to the cases of transmitting 50% resistance, and 0.5% resistance for different equipment types yields the results shown in Table 5-3.

**Table 5-3. Example transmitted resistance values**

Data characteristic	Increment	0.5%	50%
Indoor bike data	1	1	100
Rower data	1	1	100
Cross trainer data	0.1	10	1000
FTMS control point	0.1	10	1000
FTMS control point supported resistance range	0.1	10	1000
FTMS fitness machine status	0.1	10	1000

## 5.2 G.FIT custom service

The G.FIT Custom Service is a proprietary service used to send additional fitness equipment data not currently supported by FTMS, as well as enable exchange of manufacturer specific messages over BLE. The G.FIT Custom Service was modeled after FTMS.

Service declaration: Primary service, Custom UUID

**Table 5-4. Service characteristics (primary service)**

Characteristic name	Properties	Security permissions
Custom Control Point	Write, Indicate	Encryption
Custom Data	Notify	Encryption

### 5.2.1 Custom control point characteristic

The custom control point characteristic can be used by a BLE display to configure the user weight and/or bicycle weight used by G.FIT, as well as to send manufacturer specific data from the display to the fitness equipment.

**Table 5-5. Custom control point characteristic structure**

	Op code	Parameter
Octet order	N/A	Little-endian
Data type	uint8_t	uint_8_t[5]

**Table 5-6. Op code and parameters**

Op code	Definition	Parameter
0x00	Reserved for future use.	N/A
0x01	Set user weight.	User weight (uint32_t in 0.005kg)
0x02	Set bicycle weight.	Bicycle weight (uint16_t in 0.05kg)
0x03-0x7F	Reserved for future use.	N/A
0x80	Response code	See Table 5-7
0x81-0xDF	Reserved for future use.	N/A
0xE0-0xFF	Manufacturer specific op codes	Defined by manufacturer.

**Table 5-7. G.FIT BLE response codes**

Parameter	Type	Values
Request op code	uint8_t	Op code of command being responded to.
Response op code	uint8_t	0x00 Reserved for future use. 0x01 Success (command accepted through API gfit_fep_send_command_update). 0x02 Op code not supported (returned if a reserved op code is sent by the display). 0x03 Invalid parameter. 0x04 Operation failed (command rejected through API gfit_fep_send_command_update). 0x05 Not authorized (returned if an unencrypted connection is used).

### 5.2.2 Custom data characteristic

The custom data characteristic can be used to send the user weight and/or bicycle weight that is currently in use by G.FIT to a BLE display, as well as to send manufacturer specific data from the fitness equipment to the display.

**Table 5-8. Custom data characteristic structure**

	Op code	Parameter
Octet order	N/A	Little-endian
Data type	uint8_t	uint_8_t[5]

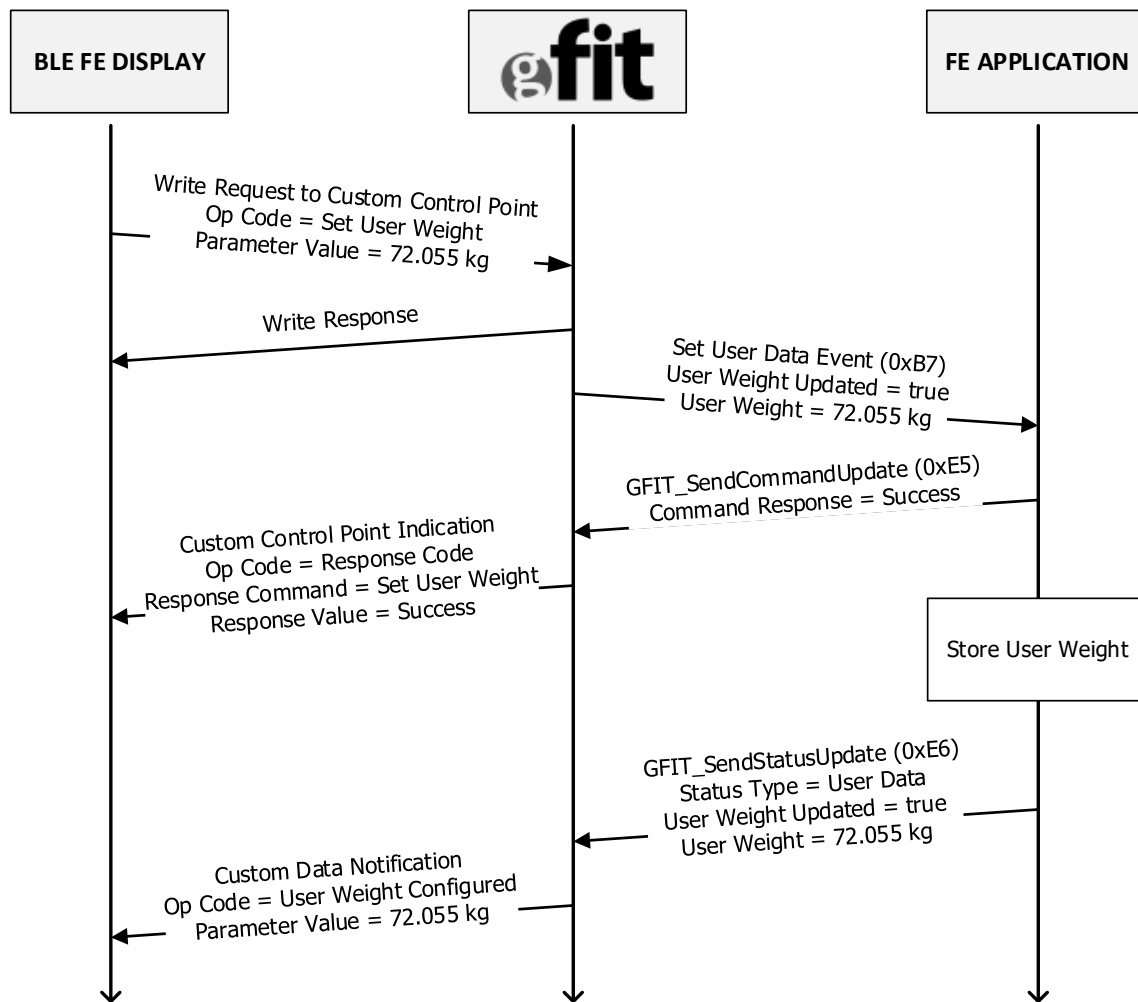
**Table 5-9. Op code and parameters**

Op code	Definition	Parameter
0x00	Reserved for future use.	N/A
0x01	Configured user weight.	Current user weight (uint32_t in 0.005kg)
0x02	Configured bicycle weight.	Current bicycle weight (uint16_t in 0.05kg)
0x03-0xDF	Reserved for future use.	N/A
0xE0-0xFF	Manufacturer specific op codes	Defined by manufacturer.

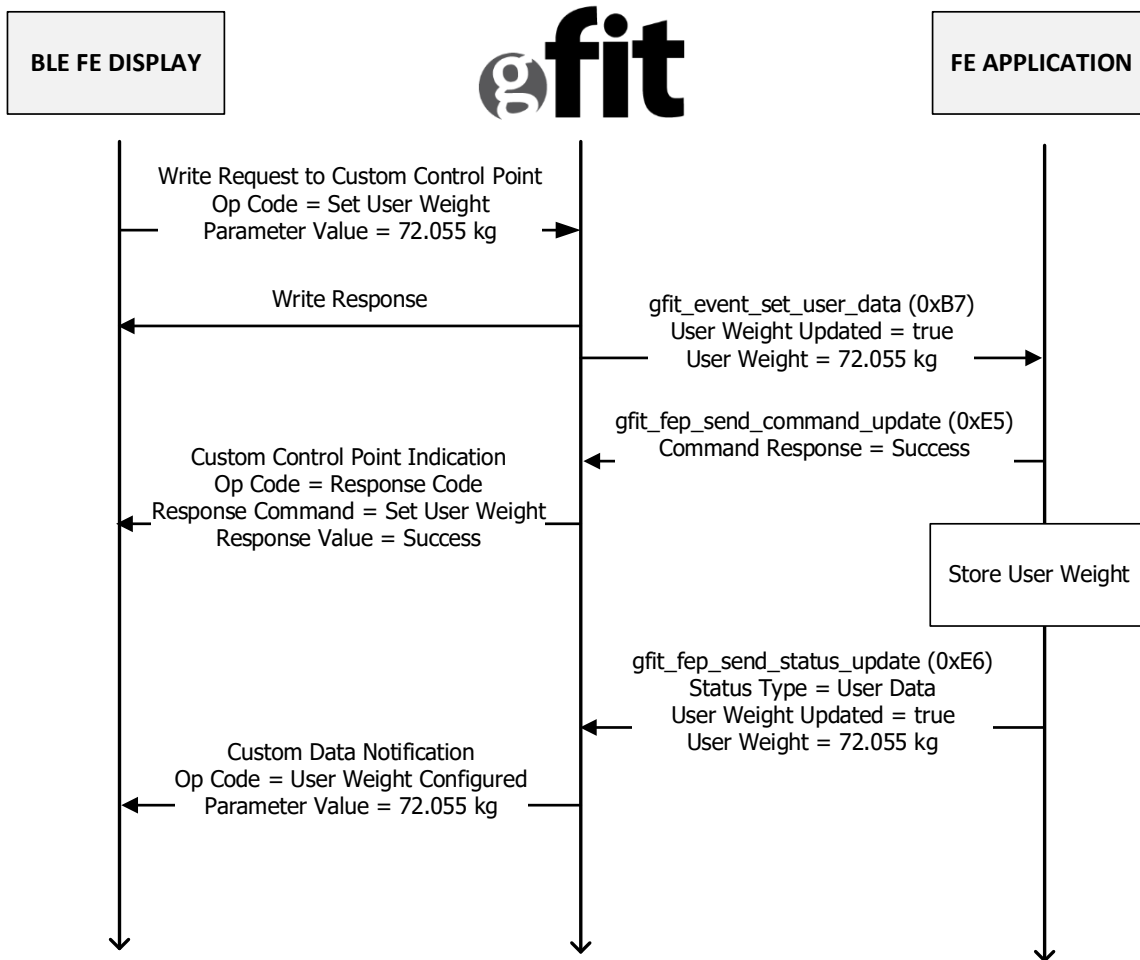
### 5.2.3 Procedures

#### 5.2.4 Setting the user and/or bicycle weight

The figure below shows the message flow that occurs when a BLE display is used to set the user weight on a G.FIT module. To set the bicycle weight, follow the same process, but include op code 0x02 for bicycle weight instead of 0x01 for user weight.





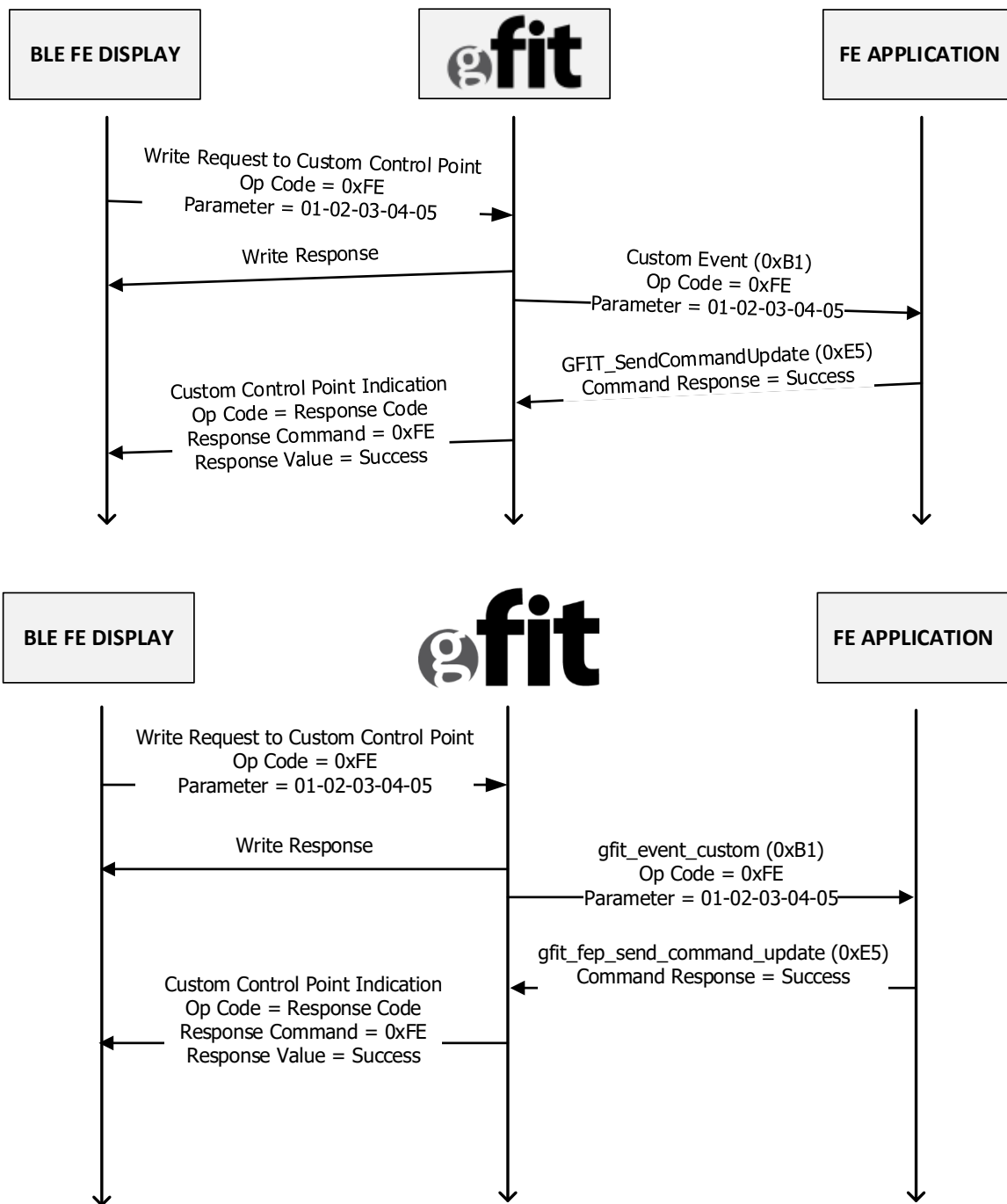


**Figure 5-1. Using BLE to set the user weight**

The `fe_command_status_update` command at the end of this process is optional. This step enables G.FIT to transmit the new user weight value on both the BLE channel (as a notification) and the ANT+ FE-C channel (as a requestable data page).

### 5.2.5 Setting manufacturer specific data

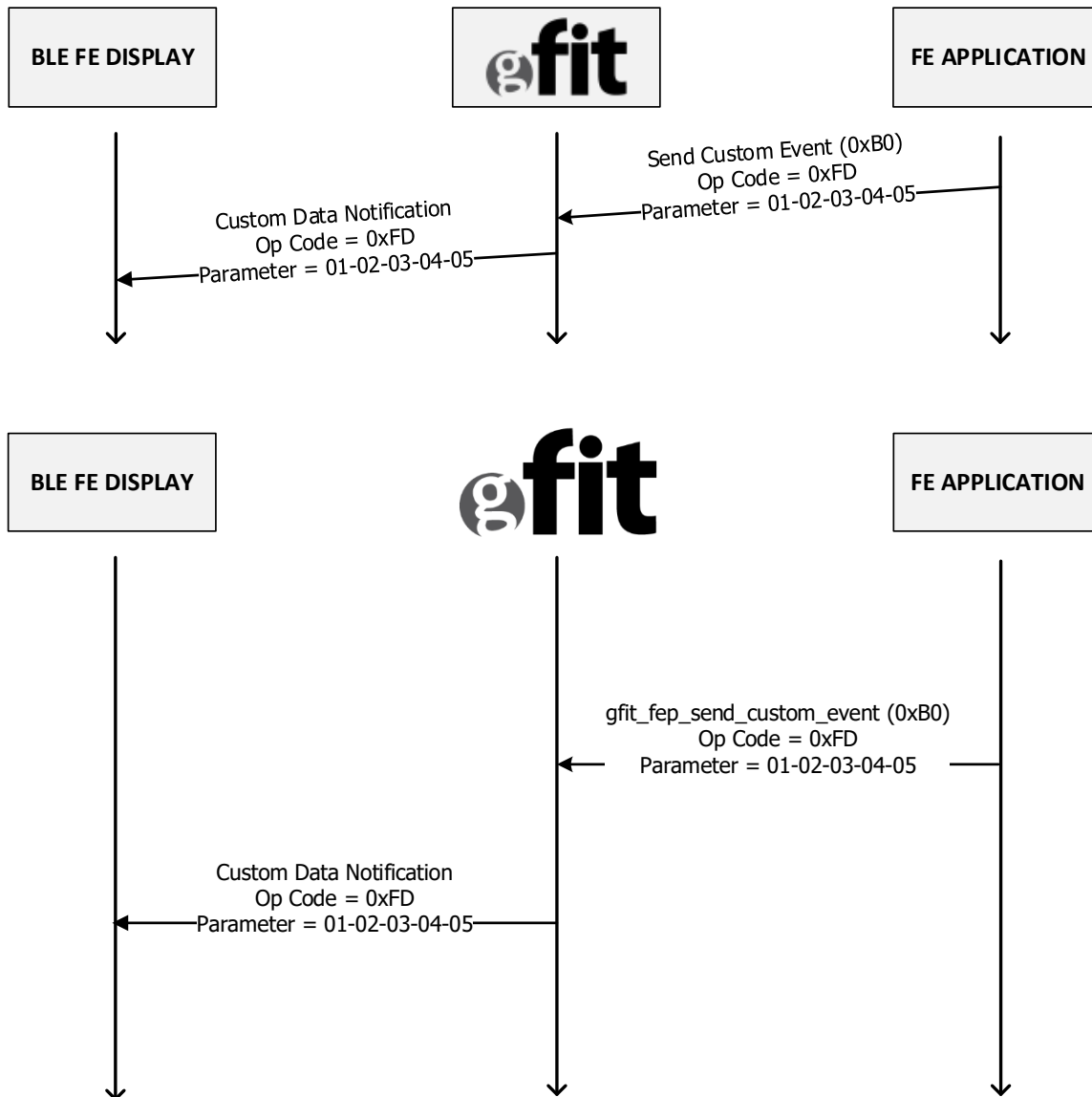
The figure below shows the message flow that occurs when a BLE display is used to set a manufacturer specific value on a G.FIT module.



**Figure 5-2. Using BLE to set a manufacturer specific value**

### **5.2.6 Notification of updated manufacturer specific value**

The figure below shows the fitness equipment application sending a serial command to the G.FIT library that causes G.FIT to send a notification to the BLE display containing the new manufacturer specific value. The same data is also sent on the ANT+ FE-C channel as a manufacturer specific page (see section 6.2).



**Figure 5-3. Notifying a display of a new manufacturer specific value**

## 6 Connecting to G.FIT over ANT+ FE-C

G.FIT implements the ANT+ FE-C device profile in its entirety, with the exception of auto zero calibration. The following data pages are not included in the regular page rotation, but can be requested by a display:

- Data Page 48 – Basic Resistance
- Data Page 49 – Target Power
- Data Page 50 – Wind Resistance
- Data Page 51 – Track Resistance
- Data Page 55- User Configuration

G.FIT uses ANT channels 9 to 14 inclusive. This leaves channels 0 to 8 inclusive for use by the application if required. Do not modify the network key for network 7 as this network is configured and used by the G.FIT library. Networks numbered 0 to 6 are available for use by the application if required.

### 6.1 Dual frequency FE-C

G.FIT works around WiFi interference to ensure large groups of 50+ devices can connect. To achieve this, G.FIT transmits ANT+ FE-C messages on 2 radio frequencies: the ANT+ radio frequency (2457MHz) and 2472MHz. The same information is always transmitted on both frequencies, to provide redundancy and reduce any gaps in the received data.

The second channel, 2472MHz, uses the same channel parameters as the ANT+ FE-C channel.

### 6.2 ANT+ manufacturer specific pages

G.FIT implements manufacturer specific pages following the format in Table 6-1.

**Table 6-1. G.FIT ANT+ manufacturer specific page format**

Byte #	Length	Description
0	1 byte	Op Code (Page Number) Value must be between 0xE0 – 0xFF.
1	2 bytes	Counter LSB
2		Counter MSB
3	5 bytes	Custom Data [0]
4		Custom Data [1]
5		Custom Data [2]
6		Custom Data [3]
7		Custom Data [4]

#### 6.2.1 Receiving custom events

When G.FIT receives a data page with page number in the manufacturer specific range (0xE0 – 0xFF), it will generate a Custom Event (0xB1), with op code set to the page number, and a 5-byte application defined parameter. The counter field **shall** be incremented by the display for each new message; the counter is used internally by G.FIT to differentiate between retries and consecutive commands.

Custom events require a response, using the `gfit_fep_send_command_update` (0xE5) command. The requestable command status page will include the op code as the Last Command ID, and the counter as the Parameter.

### 6.2.2 Sending custom events

The application can send a custom message to a display by using the `gfit_fep_send_custom_event` (0xB0) command. The counter will be automatically populated by G.FIT. The custom event will be transmitted as a broadcast message over ANT+ FE-C and retried 8 times.

### 6.3 Using SimulANT+ to view transmitted data

SimulANT+ can be used to view and parse the data being transmitted by G.FIT on the ANT+ frequency (2457MHz). SimulANT+ is available for download from <https://www.thisisant.com/developer/resources/downloads/>. Before downloading SimulANT+ you will need to create an account on <http://www.thisisant.com>. For details on using SimulANT+ see the user guide provided in the SimulANT+ download package.

Ensure that the USB-m is inserted into a USB port before the SimulANT+ application is opened. To view data being transmitted by G.FIT, select a Fitness Equipment Display simulator. Turn on the Fitness Equipment Display to start receiving data from G.FIT. Figure 6-1 shows the parsed data being received from G.FIT.

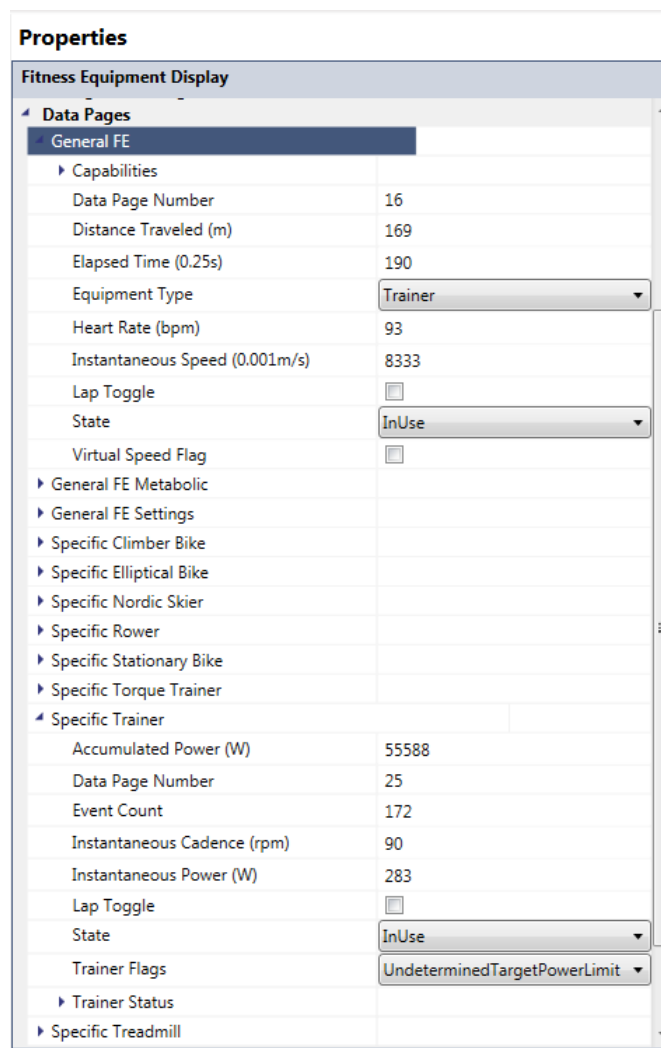


Figure 6-1. Viewing received data pages on SimulANT+

## **6.4 Tracking FE-C channel with CIQ application**

The CIQ application allows a user to connect to the G.FIT ANT+ FE-C channel. This allows the user to view their workout in real time on their watch. At the end of a workout, the user can save their session on their device. Using a CIQ application can allow the user to track their workout from the ANT+ FE-C channel and later upload that workout to Garmin Connect.

### ***6.4.1 Installing CIQ application on a compatible Garmin watch***

Connect the watch to a USB port on your computer. Select the appropriate PRG file for your specific device. Using a file browser place the PRG file into your device's GARMIN/APPS directory.

### ***6.4.2 Opening the application***

Using the devices menu, find and select the newly installed CIQ application named **G.FIT Display**.

### ***6.4.3 Connecting to the G.FIT ANT+ FE-C channel***

The CIQ application only searches for G.FIT modules that are in the READY state. If there is more than one G.FIT module in READY state than all available devices will be displayed in a list. Select the desired G.FIT module from the list.

### ***6.4.4 Viewing live workout data***

When G.FIT transitions from READY to IN USE the CIQ application detects that state transition and starts recording the user's workout. The user's workout is displayed on the watches screen. The user can pause/resume the workout timer by tapping the screen (on vivoactiveHR).

### ***6.4.5 Saving user's workout session***

If the back button is pressed the user has the option to save their data. Select yes to save the activity to the watch.

### ***6.4.6 Viewing workout data***

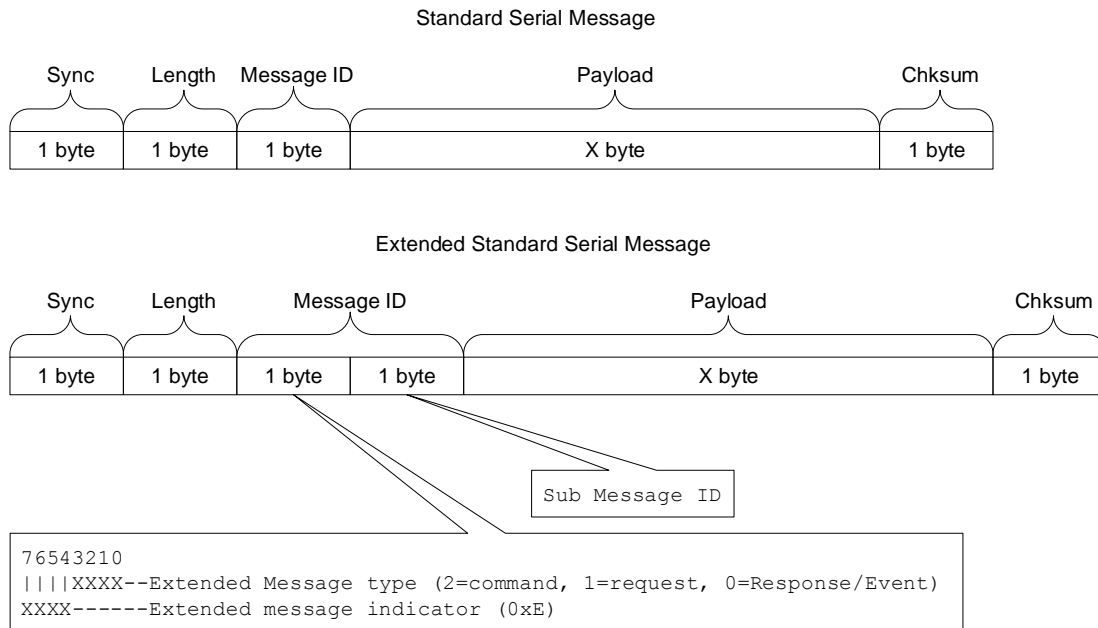
The app records FIT files during the workout. After a FIT file is saved, it can be found in the 'GARMIN\ACTIVITY' subdirectory on the watch. The 'Monkeygraph' section of the ConnectIQ Eclipse plugin can be used to view the workout data (see: <https://developer.garmin.com/connect-iq/programmers-guide/positioning-sensors/>). Alternatively, this data can be viewed by converting the FIT file to a CSV file using tools found in the 'FIT SDK' from [www.thisisant.com](http://www.thisisant.com). Once a Connect IQ app is written and uploaded as a beta version to Garmin Connect, the data can be viewed through the Garmin Connect Dashboard.

## **6.5 FE-C Control Messages**

Although multiple ANT displays may be connected to G.FIT at the same time to display broadcasted data only one can be used at a time to send FE-C control messages to the fitness equipment.

## 7 Serial interface

The extended serial message protocol is used to communicate between the G.FIT module and the application MCU. In addition to using 2-byte message IDs, the extended serial message protocol is more dynamic than the standard ANT serial message protocol. For all commands outside of G.FIT functionality, the standard serial message protocol still applies, as detailed in the *ANT Message Protocol and Usage* document.



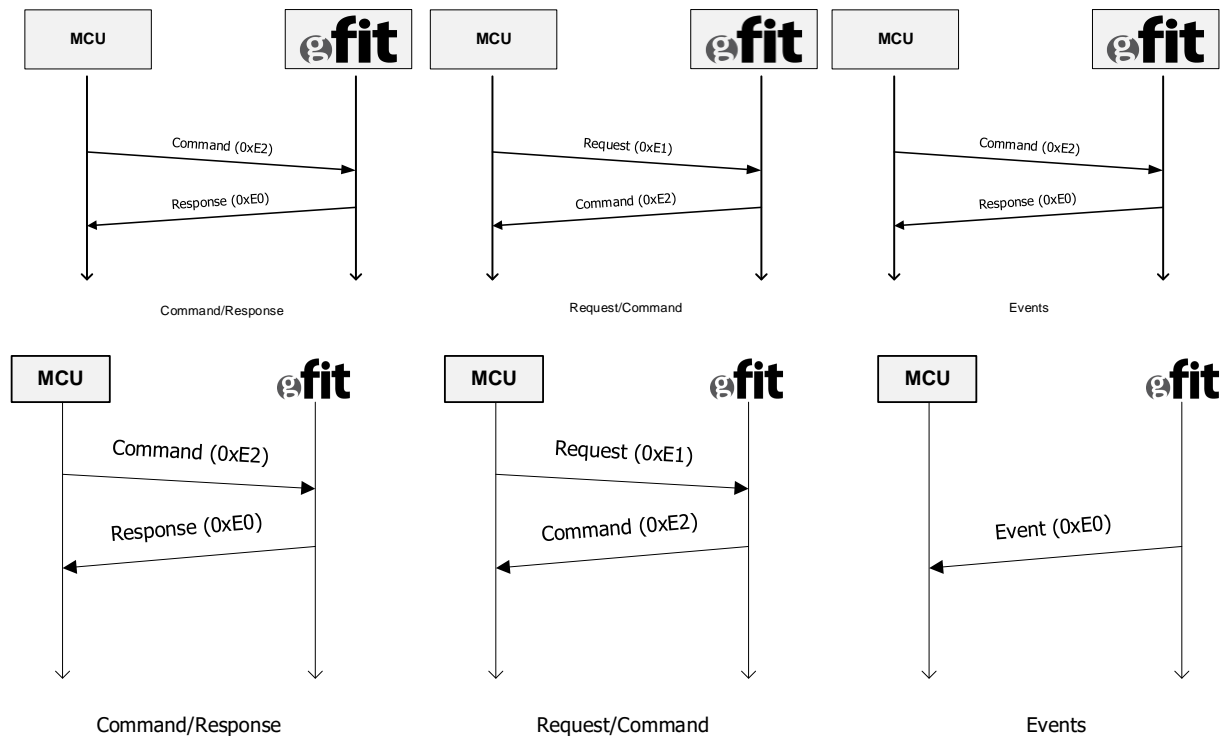
**Figure 7-1. Serial message general packet structure**

### 7.1 Using extended serial messages

The extended serial message protocol defines three types of messages – commands, requests and responses/events. Compared to a standard serial message, an extended serial message is indicated by setting the top nibble of the message ID to 0xE. The type of extended message is indicated in the following nibble (command = 0xE2, request = 0xE1 and response/event = 0xE0). The following byte is the sub-message ID, followed by the payload of the message.

The checksum for both a standard and an extended serial message is calculated as the XOR of all bytes including the sync byte. The length of a standard message includes the number of bytes in the payload of the message, whereas for extended serial messages the **length includes the payload + 1 byte** (to account for the sub-message ID). As with the standard serial message protocol, the use of optional padding bytes is also recommended for systems that are slow to react to hardware flow control (when using asynchronous serial interface).

In general, a command sent by the MCU to G.FIT elicits a response. A request elicits a command message and an event is an unsolicited message from G.FIT to the MCU. This behavior is illustrated in Figure 7-2.

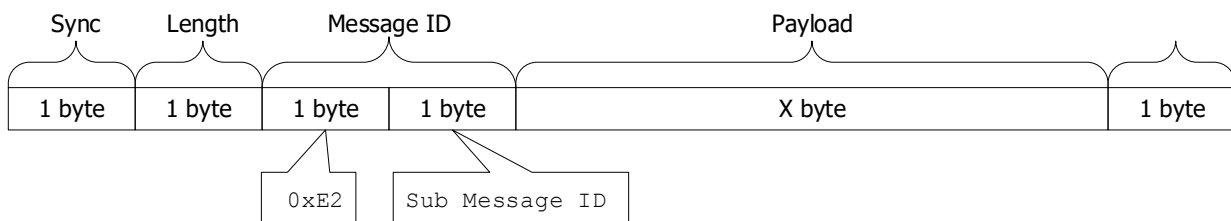


**Figure 7-2. Extended serial message types**

### 7.1.1 Commands (0xE2)

Extended serial commands are used to execute specific commands on G.FIT, or to provide information about the settings/state of G.FIT. When sent by the application MCU they elicit a response from G.FIT. If a message request is made by the application MCU, then G.FIT will send a command message in response.

The structure of a command is identical to that described by Figure 7-1, with the extended message type set to 2. The specific command to execute is indicated by the second byte of the message ID (sub-message ID). The payload is specific to each command and may be completely empty. Figure 7-3 shows an extended serial command.



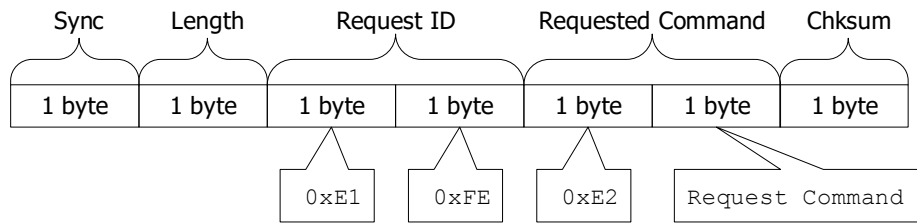
**Figure 7-3. Extended serial command**

### 7.1.2 Requests (0xE1)

Extended serial requests are used by the application MCU to request information from G.FIT. The information requested comes to the MCU in the form of a command.



The structure of a request message is shown in Figure 7-4. The first two bytes of the payload indicate which command is being requested.

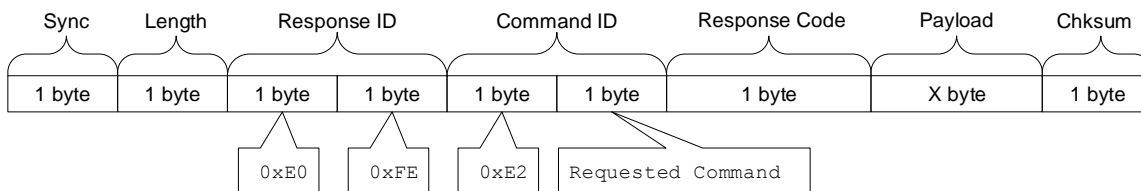


**Figure 7-4. Extended serial request**

### 7.1.3 Responses/events (0xE0)

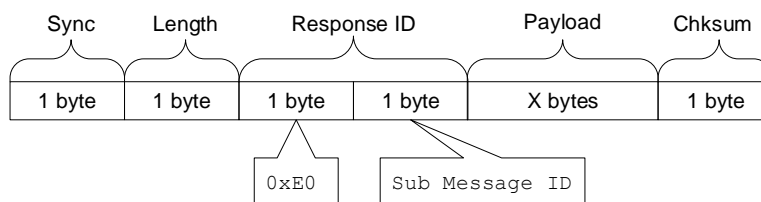
Responses are used to indicate the success or failure of a command. Events are a special form of response messages. They are unsolicited messages that come from G.FIT and indicate some type of information which the application MCU must handle.

The structure of a response message is shown in Figure 7-5. The payload of a response includes the ID of the message being responded to as well as the response code and associated payload.



**Figure 7-5. Extended serial response**

The structure of an event message is as shown in Figure 7-6. The sub-message ID indicates the type of event.



**Figure 7-6. Extended serial event**

## 7.2 G.FIT response codes

When a command is sent by the application MCU, G.FIT will respond with a response message. Within this response message is a response code. The response code informs the MCU of the success/failure of the command. Table 7-1 describes the meaning of each response code.

**Table 7-1. G.FIT response codes**

Code	Name	Description
0x00	GFIT_SUCCESS	The command performed successfully.
0x01	GFIT_ERROR INCORRECT_STATE	Many commands are only valid in specific states. This error is returned when command is called in a state where the command is not supported.
0x02	GFIT_ERROR INCORRECT_CONFIGURATION	Some commands require G.FIT to be configured appropriately before the command can properly perform its task. This error is returned if a command is called to perform a task that G.FIT is currently not configured to perform.
0x03	GFIT_ERROR_INVALID_COMMAND	The command sent to G.FIT was not recognized as a valid G.FIT command.
0x04	GFIT_ERROR INVALID_DEVICE_ID	G.FIT does not allow the ANT device ID to be set to zero. This error is returned if attempting to set a 4-byte device ID that has the two least significant bytes equal to zero.
0x05	GFIT_ERROR_INVALID_OP_CODE	This error is also returned if sending a custom event where the op code does not fall within the manufacturer specific range (greater than E0).
0x06	Reserved	This error code is not used and is reserved for future use.
0x07	GFIT_ERROR INVALID_PARAMETER	If a command is sent with an invalid parameter setting, then this error will be returned. Check which values are supported for the command that is being sent.
0x08	GFIT_ERROR NO_SENSOR	This error is returned if a disconnect command is sent and there is no wireless heart rate monitor to disconnect.
0x09	GFIT_ERROR MODULE_NOT_SUPPORTED	This error is returned if the G.FIT library is used on a module that was not intended to run the G.FIT library.
0x0A	GFIT_ERROR INVALID_LICENSE_KEY	This error is returned if using the G.FIT library and an incorrect licence key is used.
0x0B	GFIT_ERROR INVALID_FIELD_ID_FOR_EQUIPMENT_TYPE	Each fitness equipment has unique FE metrics specific to that type of equipment. If a field ID is being set that is not compatible for the configured fitness equipment type this error is returned. See section 7.3.11.2 for tables of valid field IDs for given fitness equipment types.
0x0C	GFIT_ERROR INCORRECT_FE_METRIC_PAYLOAD	This error is returned if G.FIT is unable to decode the gfit_fep_set_XX_data (0xE1) serial command. This could be caused by incorrectly sized fields, or invalid field IDs. See section 7.3.11 for more details on constructing the gfit_fep_set_XX_data (0xE1) serial command.
0x0D	GFIT_ERROR INVALID_EQUIPMENT_TYPE	This error is returned when attempting to configure G.FIT to a fitness equipment type that is not supported by G.FIT.

Code	Name	Description
0x0E	GFIT_ERROR EQUIPMENT_TYPE_NOT_SET	G.FIT needs to have an equipment type set before any FE metrics can be configured towards a given equipment type. This error is returned if attempting to move to READY or setting FE metrics without having an equipment type set.
0x0F	GFIT_ERROR NO_FIELD_IDS_CONFIGURED	This error occurs if you attempt to transition from OFF to READY without setting/configuring any FE metrics. FE metrics must be configured on G.FIT before the initial transition to READY to indicate which FE metrics are supported by the fitness equipment. See section 7.3.11 for more details.
0x10	GFIT_ERROR FIELD_ID_NOT_CONFIGURED	This error occurs when attempting to update a FE metric that was not configured on G.FIT. FE metrics must be configured before the initial transition from OFF to READY to indicate which FE metrics are supported by the fitness equipment. See section 7.3.11 for more details.
0x11	GFIT_ERROR INVALID_MFG_SPECIFIC_PAGE_NUMBER	This error is returned if sending a manufacturer page over ANT+ FE-C that does not fall within the manufacturer page range. Manufacturer pages should be greater than 0xE0.
0x12	GFIT_ERROR INVALID_LENGTH	If sending a command with an invalid length, then this error will be returned.
0x13	GFIT_ERROR INVALID_BIKE_POWER_USAGE	When equipment type is set to trainer, to comply with the ANT+ FE-C profile, G.FIT enforces that if either instantaneous power or accumulated power is updated, that both instantaneous power and accumulated power be included in the same gfit_fep_set_XX_data (0xE1) serial command.
0x14	GFIT_ERROR HR_SENSOR_PAired	This error is returned if attempting to set the HR via a command and there is already a wireless HR monitor connected, or if attempting to target pair to a HR monitor and there is already one connected.
0x15	GFIT_ERROR EQUIPMENT_TYPE_ALREADY_SET	The fitness equipment type should only be set once in a power cycle. If an attempt is made to change the equipment type, or if it is set twice then this error will be returned.
0x16	Reserved	Reserved for future use
0x17	GFIT_ERROR_INCORRECT_SPIN_DOWN_STATE	This error is returned if attempting a spin down operation in the incorrect spin down state (e.g., cancelling spin down calibration or sending a calibration status update when there is no calibration in progress).
0x18	GFIT_ERROR_NO_COMMAND_IN_PROGRESS	This error is returned if attempting to send a command update when there is no pending command.
0x19	GFIT_ERROR_INVALID_FE_STATUS_TYPE	This error is returned if attempting to send a status update with an invalid status type.

### 7.3 Commands (0xE2)

Each command is referenced by the related function in the G.FIT library and the associated sub-message ID. For example, for the command in section 7.3.2 `gfit_fep_set_channel_configuration` (0xD1), 'gfit\_fep\_set\_channel\_configuration' is the name of the function in the G.FIT library, and 0xD1 is the sub-message ID.

#### 7.3.1 `gfit_fep_set_state_` (0xD0)

Control Command: Commands the module to change states to the specified value. Only valid transitions are permitted. (See section 4.1 for more details).

It is recommended that the application waits a period greater than 200ms before performing consecutive state transitions (i.e. OFF → READY → OFF). This period allows the GFIT to perform state sensitive SoftDevice operations.

Valid State: OFF, READY, IN USE, FINISHED

Corresponding G.FIT library functions: The G.FIT library uses separate commands for each state transition: `gfit_fep_set_state_off`, `gfit_fep_set_state_ready`, `gfit_fep_set_state_inuse`, `gfit_fep_set_state_finished`. These can all be triggered by the serial command shown in Table 7-2.

**Table 7-2. `gfit_fep_set_state_` command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xD0 ( <code>gfit_fep_set_state_</code> )
4	Payload	1 byte	Set to represent the state transition: 0x01 – Transition to OFF 0x02 – Transition to READY 0x03 – Transition to IN USE 0x04 – Transition to PAUSED/FINISHED
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-3. gfit\_fep\_set\_state\_ response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xD0 (gfit_fep_set_state_)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Indicates current state if response code is GFIT_ERROR_INCORRECT_STATE: Indicates updated state if response code is GFIT_SUCCESS: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.2 gfit\_fep\_set\_channel\_configuration (0xD1)

Control Command: Determines which services are enabled.

Disabling ANT communication will disable transmission of fitness equipment data over FE-C as well as disable receiving ANT HR data. Similarly, disabling BLE peripheral communication will also disable FTMS and HRS.

Disabling FE-C or BLE peripheral communication does not affect which heart rate monitors G.FIT can receive data from.

Valid State: OFF

Default: All channels enabled

**Table 7-4. gfit\_fep\_set\_channel\_configuration command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xD1 (gfit_fep_set_channel_configuration)
4	Payload	1 byte	Bitfield representing the desired channel configuration. To enable each protocol for transmission, set the corresponding bit to 1. Bit 0 – ANT Communication Bit 1 – Dual frequency FE-C (ANT) Bit 2 – BLE peripheral (including FTMS, HRS, DIS and G.FIT custom service) Bit 3 – HRS (Heart Rate Service) Bit 4 – FTMS (Fitness Machine Service) Bit 5 – ANT HR Bit 6 – BLE HR
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-5. gfit\_fep\_set\_channel\_configuration response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xD1 (gfit_fep_set_channel_configuration)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x02 - GFIT_ERROR_INCORRECT_CONFIGURATION 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.3 gfit\_hrp\_set\_pairing\_mode (0xD2)

Configuration Command: Choose between automated proximity pairing or channel ID based pairing (where the application will choose to pair to a specific heart rate monitor using its ID), or to entirely disable HR pairing. If HR pairing is disabled, the heart rate receive channels will not be opened when transitioning to READY state, and HRS (Heart Rate Service) will not be included in the BLE peripheral. See section 4.2 for more details. Note that this command will be overwritten to disable HR pairing if both ANT and BLE HR data receiving are disabled.

Valid State(s): OFF

Default: Proximity Based

**Table 7-6. gfit\_hrp\_set\_pairing\_mode command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xD2 (gfit_hrp_set_pairing_mode)
4	Payload	1 byte	HR Pairing Mode: Set to 0x00 for proximity pairing (default setting). Set to 0x01 for channel ID based pairing. Set to 0x02 to disable HR pairing.
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-7. gfit\_hrp\_set\_pairing\_mode response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xD2 (gfit_hrp_set_pairing_mode)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.4 gfit\_hrp\_set\_inuse\_scan\_timeout (0xD3)

Configuration Command: Sets the scanning timeout for the scan channels once the module transitions to the IN USE state. After the timeout, the scan channels will close. Setting a timeout allows G.FIT to continue searching for a monitor while in state IN USE for the specified time. The pairing behavior during the search depends on the configured HR pairing mode. Avoid running additional scans on other channels or doing flash writes during this time.

Valid State: OFF

Default: 30 second in use scan timeout.

**Table 7-8. gfit\_hrp\_set\_inuse\_scan\_timeout command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xD3 (gfit_hrp_set_inuse_scan_timeout)
4	Payload	1 byte	Timeout Value: Set to 0 for no timeout (Scan channels stop when transitioning to IN USE state). Set to a value from 1-254 to set a timeout in units of 2.5 seconds Set to 255 for infinite timeout.
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-9. gfit\_hrp\_set\_inuse\_scan\_timeout response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xD3 (gfit_hrp_set_inuse_scan_timeout)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.



### 7.3.5 gfit\_hrp\_set\_pairing\_proximity (0xD5)

Configuration Command: Sets the RSSI pairing thresholds for the HR scan and proximity pairing. See section 4.2 and section 4.2.1.1 for more details on selecting RSSI values.

Required Configuration: HR Pairing Mode == Proximity.

Valid State: OFF

**Table 7-10. gfit\_hrp\_set\_pairing\_proximity command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x03
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xD5 (gfit_hrp_set_pairing_proximity)
4 – 5	Payload	2 bytes	Byte 4 - Search Threshold RSSI Value (dBm) Minimum RSSI before a heart rate monitor is considered for pairing. Values below the threshold are ignored by the pairing algorithm. Byte 5 - Pairing Threshold RSSI Value (dBm) Received signal strength required for G.FIT to pair to the heart rate monitor. G.FIT must continue to receive messages with received signal strength from the heart rate monitor greater than the pairing threshold value before G.FIT pairs to the monitor.
6	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-11. gfit\_hrp\_set\_pairing\_proximity response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xD5 (gfit_hrp_set_pairing_proximity)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x02 - GFIT_ERROR_INCORRECT_CONFIGURATION 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.6 gfit\_hrp\_pair\_to\_device (0xD6)

Control Command: Commands the module to only establish a channel with a specific device channel ID in the scan. Module must be set to use Channel ID based pairing mode.

Required Configuration: HR Pairing Mode == Channel ID.

Valid State: READY

**Table 7-12. gfit\_hrp\_pair\_to\_device command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x08
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xD6 (gfit_hrp_pair_to_device)
4 - 10	Payload	7 bytes	Byte 4 – indicates the protocol to use for pairing. Set to 0x00 to use ANT protocol Set to 0x01 to use BLE protocol Set bytes 5-10 as described in Table 7-13
11	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-13. ANT and BLE channel IDs**

Byte #	Length	If ANT protocol is used	If BLE protocol is used
5	1 byte	Set to device number LSB	Set to 6 Byte MAC Address LSB
6	1 byte	Set to device number MSB	...
7	1 byte	Set to transmission type	...
8	1 byte	Set to 0x00	...
9	1 byte	Set to 0x00	...
10	1 byte	Set to 0x00	Set to 6 Byte MAC Address MSB

The response to gfit\_hrp\_pair\_to\_device command will be returned as described in Table 7-14.

**Table 7-14. gfit\_hrp\_pair\_to\_device response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xD6 (gfit_hrp_pair_to_device)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x02 - GFIT_ERROR_INCORRECT_CONFIGURATION 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x12 - GFIT_ERROR_INVALID_LENGTH 0x14 - GFIT_ERROR_HR_SENSOR_PAIRED See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.7 gfit\_hrp\_disconnect\_device (0xD7)

Control Command: Commands the module to disconnect from an already paired/connected heart rate monitor.

Valid State: READY, IN USE, FINISHED

**Table 7-15. gfit\_hrp\_disconnect\_device command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x01
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xD7 (gfit_hrp_disconnect_device)
4	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-16. gfit\_hrp\_disconnect\_device response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xD7 (gfit_hrp_disconnect_device)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x08 - GFIT_ERROR_NO_SENSOR 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.8 gfit\_enter\_bootloader (0xD8)

Control Command: Restarts G.FIT and starts up the bootloader (G.FIT firmware updater) with the specified transport type. The transport type indicates whether the G.FIT firmware updater should expect serial or wireless communication (UART or BLE).

Valid State: This command can be issued in any state.

**Table 7-17. gfit\_enter\_bootloader command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xD8 (gfit_enter_bootloader)
4	Payload	1 byte	Transport Type: Set to 0x01 to update via BLE Set to 0x02 to update via UART
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

If the gfit\_enter\_bootloader command is successful, it will not return a GFIT\_SUCCESS response message. The module will reboot in bootloader mode. A response message will only be sent in the case of command failure.

**Table 7-18. gfit\_enter\_bootloader response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xD8 (gfit_enter_bootloader)
6	Response Code	1 byte	0x01 - GFIT_ERROR_INCORRECT_STATE 0x02 - GFIT_ERROR_INCORRECT_CONFIGURATION 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.9 gfit\_fep\_set\_inuse\_adv\_timeout (0xD9)

Configuration Command: Sets the advertising timeout for the peripheral once the module transitions to the IN USE state. After the timeout, the peripheral will stop advertising. Setting a timeout allows G.FIT to continue advertising so that the user can pair their display while in state IN USE for the specified time.

Valid State: OFF

Default: 30 second in use advertising timeout.

**Table 7-19. gfit\_fep\_set\_inuse\_adv\_timeout command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xD9 (gfit_fep_set_inuse_adv_timeout t)
4	Payload	1 byte	Timeout Value: Set to 0 for no timeout (Peripheral stops advertising when transitioning to IN USE state). Set to a value from 1-254 to set a timeout in units of 2.5 seconds Set to 255 for infinite timeout.
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-20. gfit\_fep\_set\_inuse\_adv\_timeout response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xD9 (gfit_fep_set_inuse_adv_timeout)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.10 gfit\_fep\_set\_equipment\_type (0xE0)

Configuration Command: Sets the FE equipment type. This updates which FE Specific Main Data Pages are transmitted on the ANT+ FE-C channel and which characteristics are enabled in BLE FTMS.

Optionally, specifies controllable features to be enabled. Enabling controllable features will configure the capabilities reported on the ANT+ FE-C channel and on BLE FTMS, and an event will be generated by G.FIT when a message corresponding to that controllable feature is received over ANT/BLE. Disabling a controllable feature will result in the feature being reported as not supported in the capabilities, and received commands related to that feature will be automatically rejected (on BLE FTMS) or ignored (on ANT+) by G.FIT.

Valid State: This configuration must be applied once in the OFF state, before the first transition to READY state (see section 4.1.1). Once the equipment type and optional controllable features are configured, a new configuration cannot be applied without power cycling G.FIT.

**Table 7-21. gfit\_fep\_set\_equipment\_type command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to: 0x02 – If not using optional controllable feature fields 0x04 – If using optional controllable features fields
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xE0 (gfit_fep_set_equipment_type)
4	Payload	1 byte	Set to represent the FE Type: 19 – Treadmill 20 – Elliptical 22 – Rower 23 – Climber 25 – Trainer/Indoor Bike
5	Controllable Features 1[Optional]	1 byte	Bitfield representing a set of controllable features. Set the corresponding bit to 1 to enable the feature. Bit 0 – Target Resistance (ANT+/FTMS) Bit 1 – Target Power (ANT+/FTMS) Bit 2 – Target Inclination (FTMS) Bit 3 – Targeted Distance (FTMS) Bit 4 – Target Heart Rate (FTMS) Bits 5-7 – Reserved for future use. Set to 0. Default: If the optional controllable features are not included, then all controllable features are disabled.
6	Controllable Features 2[Optional]	1 byte	Bitfield representing a set of controllable features. Set the corresponding bit to 1 to enable the feature. Bit 0 – Reset (FTMS) Bit 1 – Start, Stop, Pause (FTMS) Bit 2 – Bike Simulation (ANT+/FTMS/Custom G.FIT Service) Bit 3 – Spin Down Calibration (ANT+/FTMS) Bit 4 – Custom G.FIT Events (ANT+/Custom G.FIT Service) Bits 5-7 – Reserved for future use. Set to 0. Default: If the optional controllable features are not included, then all controllable features are disabled.
7	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Note:** If any controllable features are enabled, then both controllable features 1 and 2 must be included in the command.

As per the requirements of the ANT+ FE-C device profile, ANT+ controllable features can only be applied to fitness equipment type 'Trainer/Indoor Bike'. If controllable features are enabled for other fitness equipment types, control commands will be ignored when received over ANT+ but will be processed and will generate events when received over BLE.

If enabling Target Resistance or Bike Simulation for fitness equipment type 'Trainer/Indoor Bike', Target Power will also be enabled for ANT+, as this is a minimum requirement of the ANT+ FE-C device profile.

For the BLE FTMS Control Point characteristic, op code 0x00 (Request Control Point) will be automatically accepted if any of the controllable features supported over FTMS are enabled.

**Table 7-22. gfit\_fep\_set\_equipment\_type response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xE0 (gfit_fep_set_equipment_type)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x0D - GFIT_ERROR_INVALID_EQUIPMENT_TYPE 0x12 - GFIT_ERROR_INVALID_LENGTH 0x15 - GFIT_ERROR_EQUIPMENT_TYPE_ALREADY_SET See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.



### **7.3.11 *gfit\_fep\_set\_XX\_data (0xE1)***

When comparing the ANT+ FE-C profile and the BLE FTMS, there is quite a bit of variance in how the data is represented and transmitted over the air. More specifically, there are discrepancies between the units, ranges, and sizes of fields. G.FIT's interface for setting these fitness equipment metrics was designed to handle this complexity. As a result, the application is only required to enter a single G.FIT defined value that is compatible with both profiles. G.FIT then formats and sends the data accordingly for each profile, reducing implementation effort for the application layer.

Within this document 'FE metrics' refers to the data elements that are sent from the given type of fitness equipment such as speed, distance, power, etc.

Field IDs are used as a key to specify which FE metric the app is attempting to update on G.FIT. See section 7.3.11.2 for a list of field IDs that are applicable for a given fitness equipment type.

A field contains the value of the given FE metric that corresponds to the preceding field ID.

The `gfit_fep_set_XX_data` serial command has two purposes. When in the initial OFF state (described in section 4.1.1), `gfit_fep_set_XX_data` is used to configure G.FIT by specifying which FE metrics are supported by the fitness equipment. Once G.FIT moves to READY for the first time, it is no longer in configuration mode and the `gfit_fep_set_XX_data` serial command is used to set the values of the FE metrics, including any subsequent OFF states.

**Configuration Command (OFF):** `gfit_fep_set_XX_data` is used to configure which FE metrics G.FIT will support prior to initial use. This serial command must be sent once before the first transition from OFF to READY. This command is used to configure G.FIT and specify which FE metrics the fitness equipment supports. Configuring a FE metric before the initial OFF to READY transition ensures that the FE metric shows up as supported when transmitted over ANT+ FE-C and that it is included in the BLE FTMS. Any FE metrics not configured during first OFF (configuration) state, will be not be permitted to be updated in later states.

**Set Data Command (OFF, READY, IN USE, FINISHED):** Update the current fitness equipment metrics that G.FIT is reporting on the ANT+ FE-C channel and BLE FTMS.

**Valid State:** OFF, READY, IN USE, FINISHED

**Corresponding G.FIT library functions:** `gfit_fep_set_treadmill_data`, `gfit_fep_set_bike_data`, `gfit_fep_set_elliptical_data`, `gfit_fep_set_rower_data`, `gfit_fep_set_step_climber_data`

The G.FIT library uses separate commands for each equipment type. The set FE metrics serial message is used to update the values that are transmitted by G.FIT over ANT+ FE-C and BLE FTMS. Any number of FE metrics can be set in a single serial command. The serial message must be constructed as a series of pairs of a field ID followed by a field, so that G.FIT knows which FE metric the value (field) is associated with. See Figure 7-7 for an example of how this is completed.

For the configuration command, include the field ID of each supported field and set the value field to zero in each case. (This does not initialise the value to zero as the field is ignored but is important to maintain future compatibility.)

For the set data command, include the field ID and associated value field for each field where the value has changed. Field IDs with unchanged data may also be included if desired but are unnecessary.

ANT+ and BLE offer different units and ranges for each of the supported fields. Table 7-26 to Table 7-30 show the input units for G.FIT and the output units on both ANT+ and BLE. If an input value is provided that exceeds the supported range in either ANT+ or BLE, the value is capped to the largest value supported in the protocol with the smallest range and transmitted as is on the protocol with the largest range.

**Note:** While heart rate is considered an FE metric, it cannot be enabled or updated via the `gfit_fep_set_XX_data` serial command. The heart rate metric will automatically update if there is an external ANT+ or BLE heart rate monitor connected to G.FIT. If no external ANT+ or BLE heart rate monitor is connected to G.FIT, the `gfit_hrp_set_heart_rate` serial command can be used to manually set the heart rate metric (see section 7.3.12).

**Table 7-23. gfit\_fep\_set\_XX\_data command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Length is variable. See Equation 7-1 for calculating length.
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xE1 ( <code>gfit_fep_set_XX_data</code> )
4 – N	Payload	N bytes	The payload contains a set of key value pairs that determine which FE metrics will be set on G.FIT. The payload must start with a field ID followed by the dynamically sized field. If more than one FE metric needs to be set, you can continue placing field IDs followed by fields. See Figure 7-7 for an example of how to construct the payload.
N + 1	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-24. gfit\_fep\_set\_XX\_data response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xE1 ( <code>gfit_fep_set_XX_data</code> )
6	Response Code	1 byte	See Table 7-1 for descriptions. 0x00 - GFIT_SUCCESS 0x0B - GFIT_ERROR_INVALID_FIELD_ID_FOR_EQUIPMENT_TYPE 0x0C - GFIT_ERROR_INCORRECT_FE_METRIC_PAYLOAD 0x0E - GFIT_ERROR_EQUIPMENT_TYPE_NOT_SET 0x0F - GFIT_ERROR_NO_FIELD_IDS_CONFIGURED 0x10 - GFIT_ERROR_FIELD_ID_NOT_CONFIGURED 0x12 - GFIT_ERROR_INVALID_LENGTH 0x13 - GFIT_ERROR_INVALID_BIKE_POWER_USAGE
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.11.1 Details on constructing the gfit\_fep\_set\_XX\_data serial message

When the gfit\_fep\_set\_XX\_data payload (a series of pairs of a field ID followed by a field) is constructed, the fields must be represented using the appropriate length and encoded in little-endian. See Table 7-25 for the length of a field for a given field ID.

**Table 7-25. Field lengths**

Field ID used	Field length required
1 to 31	1
32 to 95	2
96 to 127	3

After the payload has been constructed, the length of the serial message can be calculated. The length of the serial message must include the Sub-Message ID, field IDs used, and the length of all fields combined. See Equation 7-1 for how to calculate the gfit\_fep\_set\_XX\_data serial message length.

$$\text{Total Field Length} = (\text{Number of 1-byte fields}) \times 1 + (\text{Number of 2-byte fields}) \times 2 + (\text{Number of 3-byte fields}) \times 3$$

$$\text{FE\_SetFEMetrics Length} = 1 + \text{Number of Updated FE Metrics} + \text{Total Field Length}$$

#### Equation 7-1. Calculating gfit\_fep\_set\_XX\_data length

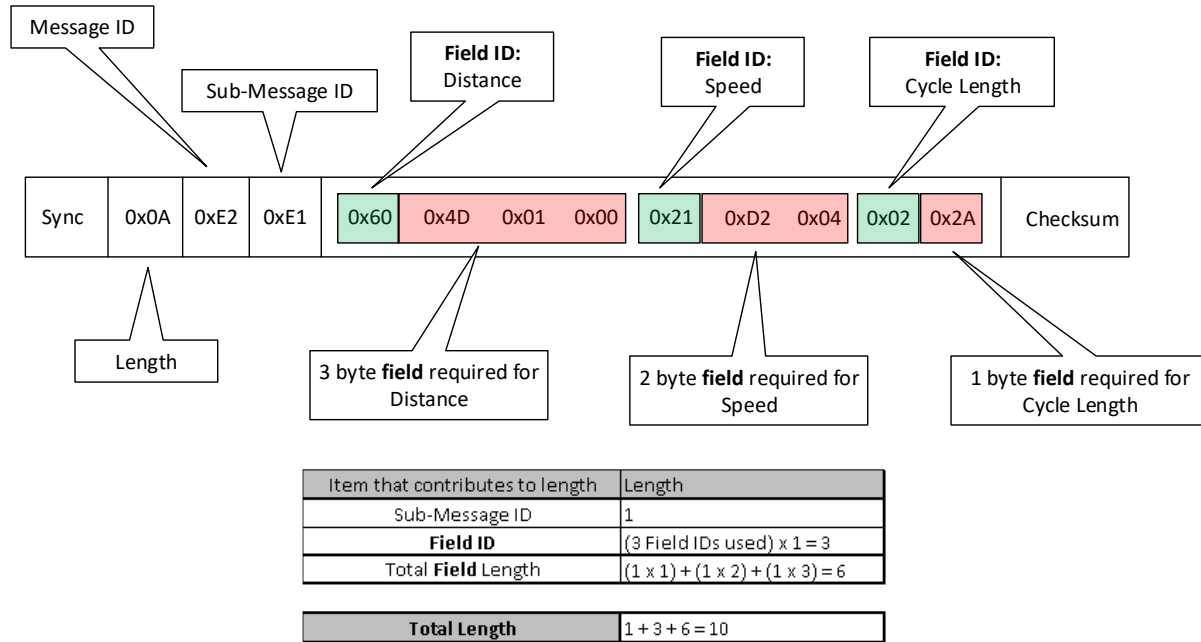
Figure 7-7 below shows an example of how to construct a gfit\_fep\_set\_XX\_data serial message. In this example 3 FE metrics are set in a single serial message. The message can be used to set any number of FE metrics, from 1 to all FE metrics for the given fitness equipment type.

**Set FE Metrics Example:**

Set Distance to 333

Set Speed to 1234

Set Cycle Length to 42

**Figure 7-7. Example: Setting FE metrics serial command (0xE1)**

**7.3.11.2 Fitness equipment type field IDs****Table 7-26. Treadmill field IDs and field lengths**

<b>Treadmill</b>	<b>FE metric</b>	<b>Input units (G.FIT)</b>	<b>Output units (ANT)</b>	<b>Output units (BLE)</b>	<b>Field ID</b>	<b>Field length (bytes)</b>	<b>Set value to invalid</b>
Treadmill Specific Data	Elapsed Time	Seconds	Quarter Seconds	Seconds	32 (0x20)	2	--
	Instantaneous Speed	Millimeters per Second	Millimeters per Second	Hundredth Kilometer per Hour	33 (0x21)	2	0xFFFF
	Instantaneous Cadence	Steps per Minute	Steps per Minute	--	35 (0x23)	2	0xFFFF
	Negative Vertical Distance	Tenth Meter	Tenth Meter	Tenth Meter	36 (0x24)	2	--
	Positive Vertical Distance	Tenth Meter	Tenth Meter	Tenth Meter	37 (0x25)	2	--
	Total Distance	Meters	Meters	Meters	96 (0x60)	3	--
Metabolic Data	Metabolic Equivalent	Hundredth MET	Hundredth MET	Tenth MET	44 (0x2C)	2	0xFFFF
	Caloric Burn Rate	Tenth kCal per Hour	Tenth kCal per Hour	kCal per Hour	45 (0x2D)	2	0xFFFF
	Calories Burned	kCal	kCal	kCal	46 (0x2E)	2	--
Settings	Cycle Length	Hundredth Meter	Hundredth Meter	--	2 (0x02)	1	0xFF
	Incline	Hundredth Percent	Hundredth Percent	Tenth Percent	47 (0x2F)	2 (signed)	0x7FFF

**Table 7-27. Trainer/indoor bike field IDs and field lengths**

Trainer/indoor bike	FE metric	Input units (G.FIT)	Output units (ANT)	Output units (BLE)	Field ID	Field length (bytes)	Set value to invalid
Bike Specific Data	Elapsed Time	Seconds	Quarter Seconds	Seconds	32 (0x20)	2	--
	Instantaneous Speed	Millimeters per Second	Millimeters per Second	Hundredth Kilometer per Hour	33 (0x21)	2	0xFFFF
	Instantaneous Cadence	Half Revolutions per Min	Revolutions per Min	Half Revolutions per Min	34 (0x22)	2	0xFFFF
	Instantaneous Power†	Watts	Watts (unsigned)	Watts	42 (0x2A)	2 (signed)	0x8000
	Accumulated Power†	Watts	Watts	--	43 (0x2B)	2	--
	Total Distance	Meters	Meters	Meters	96 (0x60)	3	--
Metabolic Data	Metabolic Equivalent	Hundredth MET	Hundredth MET	Tenth MET	44 (0x2C)	2	0xFFFF
	Caloric Burn Rate	Tenth kCal per Hour	Tenth kCal per Hour	kCal per Hour	45 (0x2D)	2	0xFFFF
	Calories Burned	kCal	kCal	kCal	46 (0x2E)	2	--
Settings	Resistance Level	Half Percent	Half Percent	Half Percent	51 (0x33)	2 (signed)	--
	Cycle Length	Hundredth Meter	Hundredth Meter	--	2 (0x02)	1	0xFF

† To comply with the ANT+ FE-C profile, G.FIT enforces that if either instantaneous power or accumulated power is updated, that both instantaneous power and accumulated power be included in the same gfit\_fep\_set\_XX\_data serial command. When updating either field ID 42 or 43, if both field IDs 42 and 43 are not updated in the same serial message, G.FIT will return error: GFIT\_ERROR\_INVALID\_BIKE\_POWER\_USAGE (0x13).

**Table 7-28. Elliptical/cross trainer field IDs and field lengths**

Elliptical/cross trainer	FE metric	Input units (G.FIT)	Output units (ANT)	Output units (BLE)	Field ID	Field length (bytes)	Set value to invalid
Elliptical Specific Data	Elapsed Time	Seconds	Quarter Seconds	Seconds	32 (0x20)	2	--
	Instantaneous Speed	Millimeters per Second	Millimeters per Second	Hundredth Kilometer per Hour	33 (0x21)	2	0xFFFF
	Instantaneous Cadence	Steps per Minute	Stride† (Steps/2) per Min	Steps per Minute	35 (0x23)	2	0xFFFF
	Positive Vertical Distance	Tenth Meter	Tenth Meter	Meter	37 (0x25)	2	--
	Stride† Count	Tenth Stride†	Stride†	Tenth Stride†	39 (0x27)	2	--
	Instantaneous Power	Watts	Watts (unsigned)	Watts	42 (0x2A)	2 (signed)	0x8000
	Total Distance	Meters	Meters	Meters	96 (0x60)	3	--
Metabolic Data	Metabolic Equivalent	Hundredth MET	Hundredth MET	Tenth MET	44 (0x2C)	2	0xFFFF
	Caloric Burn Rate	Tenth kCal per Hour	Tenth kCal per Hour	kCal per Hour	45 (0x2D)	2	0xFFFF
	Calories Burned	kCal	kCal	kCal	46 (0x2E)	2	--
Settings	Resistance Level	Half Percent	Half Percent	Twentieth Percent	51 (0x33)	2 (signed)	--
	Cycle Length	Hundredth Meter	Hundredth Meter	--	2 (0x02)	1	0xFF
	Incline	Hundredth Percent	Hundredth Percent	Tenth Percent	47 (0x2F)	2 (signed)	0x7FFF

† A stride is defined as a full revolution of the elliptical rear pedal disk/flywheel, which is the equivalent of two steps.

**Table 7-29. Rower field IDs and field lengths**

Rower	FE metric	Input units (G.FIT)	Output units (ANT)	Output units (BLE)	Field ID	Field length (bytes)	Set value to invalid
Rower Specific Data	Elapsed Time	Seconds	Quarter Seconds	Seconds	32 (0x20)	2	--
	Instantaneous Speed	Millimeters per Second	Millimeters per Second	Seconds per 500 m†	33 (0x21)	2	0xFFFF
	Instantaneous Cadence	Half Strokes per Min	Strokes per Min	Half Strokes per Min	34 (0x22)	2	0xFFFF
	Stroke Count	Strokes	Strokes	Strokes	41 (0x29)	2	--
	Instantaneous Power	Watts	Watts (unsigned)	Watts (signed)	42 (0x2A)	2 (signed)	0x8000
	Total Distance	Meters	Meters	Meters	96 (0x60)	3	--
Metabolic Data	Metabolic Equivalent	Hundredth MET	Hundredth MET	Tenth MET	44 (0x2C)	2	0xFFFF
	Caloric Burn Rate	Tenth kCal per Hour	Tenth kCal per Hour	kCal per Hour	45 (0x2D)	2	0xFFFF
	Calories Burned	kCal	kCal	kCal	46 (0x2E)	2	--
Settings	Resistance Level	Half Percent	Half Percent	Half Percent	51 (0x33)	2 (signed)	--
	Cycle Length	Hundredth Meter	Hundredth Meter	--	2 (0x02)	1	0xFF

† Transmitted over BLE as instantaneous pace



**Table 7-30. Step climber field IDs and field lengths**

Step climber	FE metric	Input units (G.FIT)	Output units (ANT)	Output units (BLE)	Field ID	Field length (bytes)	Set value to invalid
Step Climber Specific Data	Elapsed Time	Seconds	Quarter Seconds	Seconds	32 (0x20)	2	--
	Instantaneous Speed	Millimeters per Second	Millimeters per Second	Hundredth Kilometer per Hour	33 (0x21)	2	0xFFFF
	Instantaneous Cadence	Steps per Minute	Cycles <sup>†</sup> (Steps/2) per Minute	Steps per Minute	35 (0x23)	2	0xFFFF
	Positive Vertical Distance (Climber)	Meter	Meter <sup>‡</sup>	Meter	38 (0x26)	2	--
	Step Count	Steps	Cycle <sup>†</sup> (Steps/2)	Steps	40 (0x28)	2	--
	Instantaneous Power	Watts	Watts (unsigned)	--	42 (0x2A)	2 (signed)	0x8000
	Floors Climbed	Floors	--	Floors	50 (0x32)	2	--
Metabolic Data	Metabolic Equivalent	Hundredth MET	Hundredth MET	Tenth MET	44 (0x2C)	2	0xFFFF
	Caloric Burn Rate	Tenth kCal per Hour	Tenth kCal per Hour	kCal per Hour	45 (0x2D)	2	0xFFFF
	Calories Burned	kCal	kCal	kCal	46 (0x2E)	2	--
Settings	Cycle Length	Hundredth Meter	Hundredth Meter	--	2 (0x02)	1	0xFF

<sup>†</sup> A cycle is defined as a one step taken with each foot, resulting in 1 cycle = 2 steps climbed.

<sup>‡</sup> ANT+ FE-C climber specific page does not have vertical distance and will transmit inputted vertical distance as distance in data page 16.

### 7.3.12 gfit\_hrp\_set\_heart\_rate (0xE2)

Set Data Command: Update the current heart rate that the module is reporting on the ANT+ FE-C channel. This command will have no effect if there is an external ANT+ or BLE heart rate monitor connected to G.FIT.

Valid State: READY, IN USE, FINISHED

**Table 7-31. gfit\_hrp\_set\_heart\_rate command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xE2 (gfit_hrp_set_heart_rate)
4	Payload	1 byte	Set to represent the heart rate in bpm. 0x00 indicates invalid
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-32. gfit\_hrp\_set\_heart\_rate response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xE2 (gfit_hrp_set_heart_rate)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x12 - GFIT_ERROR_INVALID_LENGTH 0x14 - GFIT_ERROR_HR_SENSOR_PAIRED See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.13 gfit\_fep\_set\_device\_id (0xE3)

Configuration Command: Updates the G.FIT ANT+ FE-C device ID. This sets the 4-byte serial number transmitted in common page 81 and changes the ANT+ extended device number in the ANT channel ID. Setting the device ID does not affect the BLE M.A.C. address.

Note that the device ID is set by default to the preloaded ANT ID. The ANT ID is guaranteed to be unique. In most cases the device ID should not be set through this command manually.

G.FIT takes the preloaded ANT ID or the 4-byte device ID passed into this function and uses it to construct the ANT+ FE-C channel ID as shown in Table 7-33.

**Table 7-33. Device ID to channel ID conversion**

Device ID	ANT channel ID
Bits 0-15	Device number
Bits 16-19	Upper nibble of transmission type

Note that the lower nibble of transmission type is always 0x5 as per the ANT+ FE-C Device Profile.

Example: Assume the device ID being set is 0x76543210

The channel ID device number will be 0x3210. The channel ID transmission type will be 0x45

The device number of the fitness equipment shall not be 0x0000. Care should be taken to ensure that serial numbers that are multiples of 0x10000 (65536) are handled correctly so that the lower 16-bits of the device ID will not be equal to 0.

Valid State: OFF

**Table 7-34. gfit\_fep\_set\_device\_id command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x05
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xE3 (gfit_fep_set_device_id)
4	Payload	1 byte	Device number LSB
5		1 byte	...
6		1 byte	...
7		1 byte	Device number MSB
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-35. gfit\_fep\_set\_device\_id response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xE3 (gfit_fep_set_device_id)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x04 - GFIT_ERROR_INVALID_DEVICE_ID 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.14 fe\_command\_product\_info (0xE4)

Set Data Command: Update the product information transmitted over the ANT+ data pages 80 and 81 (known as common data pages), and over the BLE Device Information Service and advertising packets.

Valid State: OFF

**Table 7-36. fe\_command\_product\_info command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Variable - Set to 0x02 + Product Information Value Length. See Table 7-38.
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xE4 (fe_command_product_info)
4-(N-1)	Payload	Variable	Byte 4 – Product Information Type. See Table 7-38.
			Byte 5 – (N-1) – Product Information Value. See Table 7-38.
N	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-37. fe\_command\_product\_info response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xE4 (fe_command_product_info)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-38. Product information types and values**

Product information type		Product information value	
Value	Description	Length (bytes)	Description
0x00	Manufacturer ID (ANT+)	2	ANT+ Manufacturer ID (assigned by ANT+ Alliance)
0x01	Model Number (ANT+)	2	Model Number (defined by manufacturer)
0x02	Hardware Revision (ANT+/BLE)	1	Hardware revision (defined by manufacturer).
0x03	Software Revision (ANT+/BLE)	2	Byte 0 – Major Revision Byte 1 – Minor Revision  If not set, G.FIT will transmit the G.FIT version number in this field.
0x04	Firmware Revision (BLE)	2	Byte 0 – Major Revision Byte 1 – Minor Revision  If not set, G.FIT will transmit the G.FIT version number in this field.
0x05	Device Name (BLE)	Variable	Device Name string (If more than 7 characters are input, only the first 7 will be set.)  If not set, G.FIT will set the Device Name to 'GFIT'
0x06	Manufacturer String (BLE)	Variable	Manufacturer String (ASCII string, maximum length 35 characters).  If not set, G.FIT will set the Manufacturer String to 'DynastreamInnovationsInc'

**Note:** All multi-byte parameters are encoded as little-endian.

### 7.3.15 gfit\_fep\_send\_command\_update (0xE5)

Control Command: Sends an update to a display over ANT+/BLE to indicate the success or failure of the last command received, e.g., received a command to set a target from the display, reset session, etc. This command shall be used by the application every time an event requiring a response is received.

Valid State: READY, IN USE, FINISHED

**Table 7-39. gfit\_fep\_send\_command\_update command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xE5 (gfit_fep_send_command_update)
4	Payload	1 byte	Set to represent the status update 0x00 – Command Response Success 0x01 – Command Response Fail
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-40. gfit\_fep\_send\_command\_update response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xE5 (gfit_fep_send_command_update)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x12 - GFIT_ERROR_INVALID_LENGTH 0x14 - GFIT_ERROR_NO_COMMAND_IN_PROGRESS See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.16 fe\_command\_status\_update (0xE6)

Control Command: Sends an update to a display over ANT+/BLE to indicate a status change after receiving a command or due to user interaction, e.g., reset, a target has been updated (resistance, incline, etc.). Sending this control command to G.FIT will update the Fitness Machine Status characteristic over FTMS or G.FIT Custom Service, depending on the status type, and will cache the content of requestable data pages 48, 49, 40, 41 or 55 (depending on the status type), as well as update the last processed command on requestable page 71 over ANT+.

Valid State: READY, IN USE, FINISHED

**Table 7-41. fe\_command\_status\_update command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Variable. ---Set to 0x02 + Status Parameter Length. See Table 7-43.
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xE6 (fe_command_status_update)
4-(N-1)	Payload	Variable	Byte 4 – Status Type. See Table 7-43. Bytes 5-(N-1) - Status Parameter. See Table 7-43 and Table 7-44.
N	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-42. fe\_command\_status\_update response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xE6 (fe_command_status_update)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x12 - GFIT_ERROR_INVALID_LENGTH 0x19 - GFIT_ERROR_INVALID_FE_STATUS_TYPE See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.



**Table 7-43. Status type**

Status type		Bytes 5 to (N-1)
Value	Description	Length
0x00	Reset (BLE)	0 bytes
0x01	Paused by user (BLE)	0 bytes
0x02	Stopped by user (BLE)	0 bytes
0x03	Stopped by safety key (BLE)	0 bytes
0x04	Started or resumed by user (BLE)	0 bytes
0x05	Target incline changed (ANT+/BLE)	2 bytes
0x06	Target resistance level changed (ANT+/BLE)	2 bytes
0x07	Target power changed (ANT+/BLE)	2 bytes
0x08	Target heart rate changed (BLE)	1 byte
0x09	Targeted distance changed (BLE)	3 bytes
0x0A	Simulation parameters changed (ANT+/BLE)	10 bytes
0x0B	User data changed (ANT+/BLE)	11 bytes

**Table 7-44. Status parameters**

Value	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10
0x00 – 0x04											
0x05	Target Inclination (0.01%, signed)										
0x06	Target Resistance Level (0.5%, signed)										
0x07	Target Power (0.25W, signed)										
0x08	Target Heart Rate (bpm, unsigned)										
0x09	Targeted Distance (m, unsigned)										
0x0A	Wind Resistance Coefficient (0.01 kg/m, unsigned) Invalid = 0xFF	Wind Speed (mm/s, signed) Invalid = 0x7FFFFFFF				Drafting Factor (0.01 unitless, unsigned) Invalid = 0xFF	Grade (0.01 %, signed) Invalid = 0x7FFF		Coefficient of Rolling Resistance (5x10-5, unitless, unsigned) Invalid = 0xFFFF		
0x0B	User Weight (0.005 kg, unsigned) Invalid = 0xFFFFFFFF				Bicycle Weight (0.05 kg, unsigned) Invalid = 0xFFFF		Bicycle Wheel Circumference (0.1 mm, unsigned) Invalid = 0xFFFFFFFF				Gear Ratio (0.03, unitless, unsigned) Invalid = 0x00

**Note:** All multi-byte parameters are encoded as little-endian.

### 7.3.17 gfit\_fep\_abort\_pending\_commands (0xE7)

Control Command: This command can be used to abort any pending non-acknowledged events requiring a response.

When a pending command is aborted using gfit\_fep\_abort\_pending\_commands:

ANT+: The command status on requestable data page 71 is set to 'Rejected'

BLE: Response sent over Fitness Machine Control Point with Response Value = 'Op Code Not Supported'

Valid State: READY, IN USE, FINISHED

**Table 7-45. gfit\_fep\_abort\_pending\_commands command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x01
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xE7 (gfit_fep_abort_pending_commands)
4	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-46. gfit\_fep\_abort\_pending\_commands response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xE7 (gfit_fep_abort_pending_commands)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x12 - GFIT_ERROR_INVALID_LENGTH 0x14 - GFIT_ERROR_NO_COMMAND_IN_PROGRESS See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.18 gfit\_fep\_set\_max\_resistance (0xE8)

Set Data Command: Configures the maximum applicable resistance in Newtons for the fitness equipment, transmitted only on ANT+ (data page 54). The equipment type must be configured before setting the maximum applicable resistance. This command is currently only supported by the 'Trainer/Indoor Bike' equipment type. This command has no effect over BLE.

Valid State: OFF, READY, IN USE, FINISHED

**Table 7-47. gfit\_fep\_set\_max\_resistance command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x03
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xE8 (gfit_fep_set_max_resistance)
4	Payload	2 bytes	Max Resistance (Newtons) LSB
5			Max Resistance (Newtons) MSB
6	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-48. gfit\_fep\_set\_max\_resistance response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xE8 (gfit_fep_set_max_resistance)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x12 - GFIT_ERROR_INVALID_LENGTH 0x0D - GFIT_ERROR_INVALID_EQUIPMENT_TYPE 0x0E - GFIT_ERROR_EQUIPMENT_TYPE_NOT_SET See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.19 gfit\_fep\_spin\_down\_ (0xE9)

Spin Down Calibration Command: This command should be sent to G.FIT after receiving a start spin down event, to indicate the fitness equipment will initiate the calibration process. This command is currently only supported by the equipment type 'Trainer/Indoor Bike'.

Valid State: READY, IN USE, FINISHED

Depends on Controllable Feature: Spin Down Calibration

Corresponding G.FIT library functions: The G.FIT library uses separate commands to execute the spin down serial message: gfit\_fep\_spin\_down\_start, gfit\_fep\_spin\_down\_set\_status, gfit\_fep\_spin\_down\_cancel, gfit\_fep\_spin\_down\_complete, gfit\_fep\_spin\_down\_request\_calibration, gfit\_fep\_spin\_down\_cancel\_request\_calibration, gfit\_fep\_spin\_down\_reject\_ignore\_request.

**Table 7-49. gfit\_fep\_spin\_down\_ command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Variable. Set to 0x02 + Command Parameter Length
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xE9 (gfit_fep_spin_down_)
4-(N-1)	Payload	Variable	Byte 4 – Command Type. See Table 7-50. Bytes 5-(N-1) – Command Parameter. See Table 7-50.
N	Checksum	1 byte	XOR of all previous bytes including the sync byte.

The payload of the command varies depending on the command type as detailed in

Table 7-50. The available command types are:

- **Start:** Send this after receiving a start spin down calibration event, to indicate the fitness equipment will initiate the calibration process
- **Set Status:** Send this when the target speed is reached (speed status OK). It can also be sent any time after spin down calibration has started but before it completes, to update values reported over ANT and BLE. This command **shall** be sent after the target speed has been reached (speed status OK).
- **Cancel:** This can be sent to G.FIT after receiving a start spin down event to indicate that the spin down calibration will not be attempted at this time. It can also be used any time after calibration has started to indicate that the spin down calibration has failed and should be aborted. For example, this can be used by the application to implement the following scenarios: timeout; speed up after the spin down timer has started; temperature conditions not unsuitable for calibration; user cancelled the operation.
- **Complete:** Send this when the spin down calibration process. Completing the spin down procedure will cancel has completed successfully. This cancels any pending calibration requests initiated through the 'Request calibration' command type (0x04).
- **Request calibration:** Send this to request that the display device start a spin down calibration. The request will be automatically cancelled upon a successful calibration procedure; or can be cancelled by the application by sending the 'Cancel request calibration' command type (0x05).
  - If a BLE display is not currently connected when this command is used, the request will be sent at the start of every new BLE connection until the request is cancelled.
  - Over ANT+, the 'Resistance calibration required' flag is set on data page 25 until the request is cancelled (see the *ANT+ Fitness Equipment Device Profile* for details).
- **Cancel Request Calibration:** Send this to cancel requests from the fitness equipment for the display to initiate calibration. This command can be used after receiving a spin down 'ignore' event to acknowledge the event and to stop the fitness equipment from indicating that it requires calibration.
- **Reject Ignore Request:** Send this to reject a request from a BLE display to ignore calibration requests (for example, when the fitness equipment is requesting to be calibrated, and the user chooses to ignore the calibration request on their BLE display, but the fitness equipment desires to continue to indicate to displays that it requires calibration).

**Table 7-50. gfit\_fep\_spin\_down\_payload**

Command type	Command parameter				
	Length (bytes)	Byte 0	Byte 1	Byte 2	Byte 3
0x00 Start	4	Target Speed Low (mm/s) Invalid = 0xFFFF		Target Speed High (mm/s) Invalid = 0xFFFF	
0x01 Set status	3	Current Temperature, in degrees with a -25C offset (e.g., 0x10 = -17C) 0xFF indicates invalid	Target Spin Down Time (ms) 0xFF indicates invalid	Bits 0-1: Temperature Status 0x00 – N/A 0x01 – Too Low 0x02 – OK 0x03 – Too High  Bits 2-3: Speed Status 0x00 – N/A 0x01 – Too Low 0x02 – OK 0x03 – Reserved for future use.  Bits 4- 7: Reserved for future use. Set to 0.	
0x02 Cancel	0				
0x03 Complete	4	Spin Down Time (ms) 0xFFFFFFFF denotes invalid This field is only transmitted over ANT+			
0x04 Request calibration	0				
0x05 Cancel calibration request	0				
0x06 Reject ignore request	0				

**Note:** All multi-byte parameters are encoded as Little Endian

**Table 7-51. gfit\_fep\_spin\_down\_ response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xE9 (gfit_fep_spin_down_)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x12 - GFIT_ERROR_INVALID_LENGTH 0x17 - GFIT_ERROR_INCORRECT_SPIN_DOWN_STATE See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.19.1 Spin down calibration process

The message flow diagrams in Figure 7-8 and Figure 7-9 show examples of successful spin down calibration messaging when communicating with ANT and BLE displays. The examples show the scenario where the FE application indicates that calibration is required prior to the user initiating calibration. (The user can also initiate calibration at any time without waiting for the application to request it.)

Figure 7-10 and Figure 7-11 show the messaging that occurs if the FE application cancels an in-progress calibration. Figure 7-12 details a possible interaction where the FE App indicates that it needs calibration, the BLE display tries to cancel it but the cancelling fails. (This situation does not apply to ANT displays.)



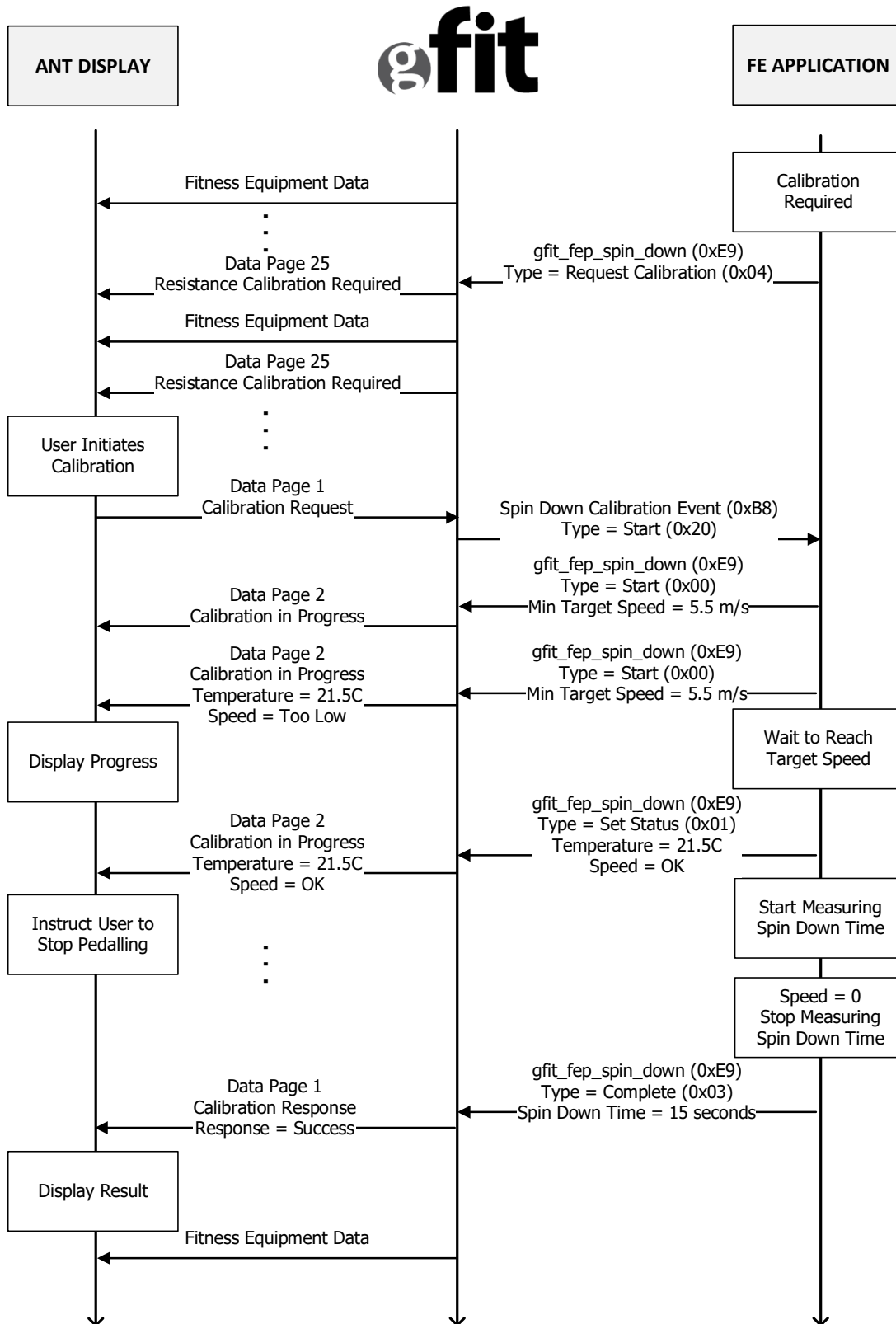
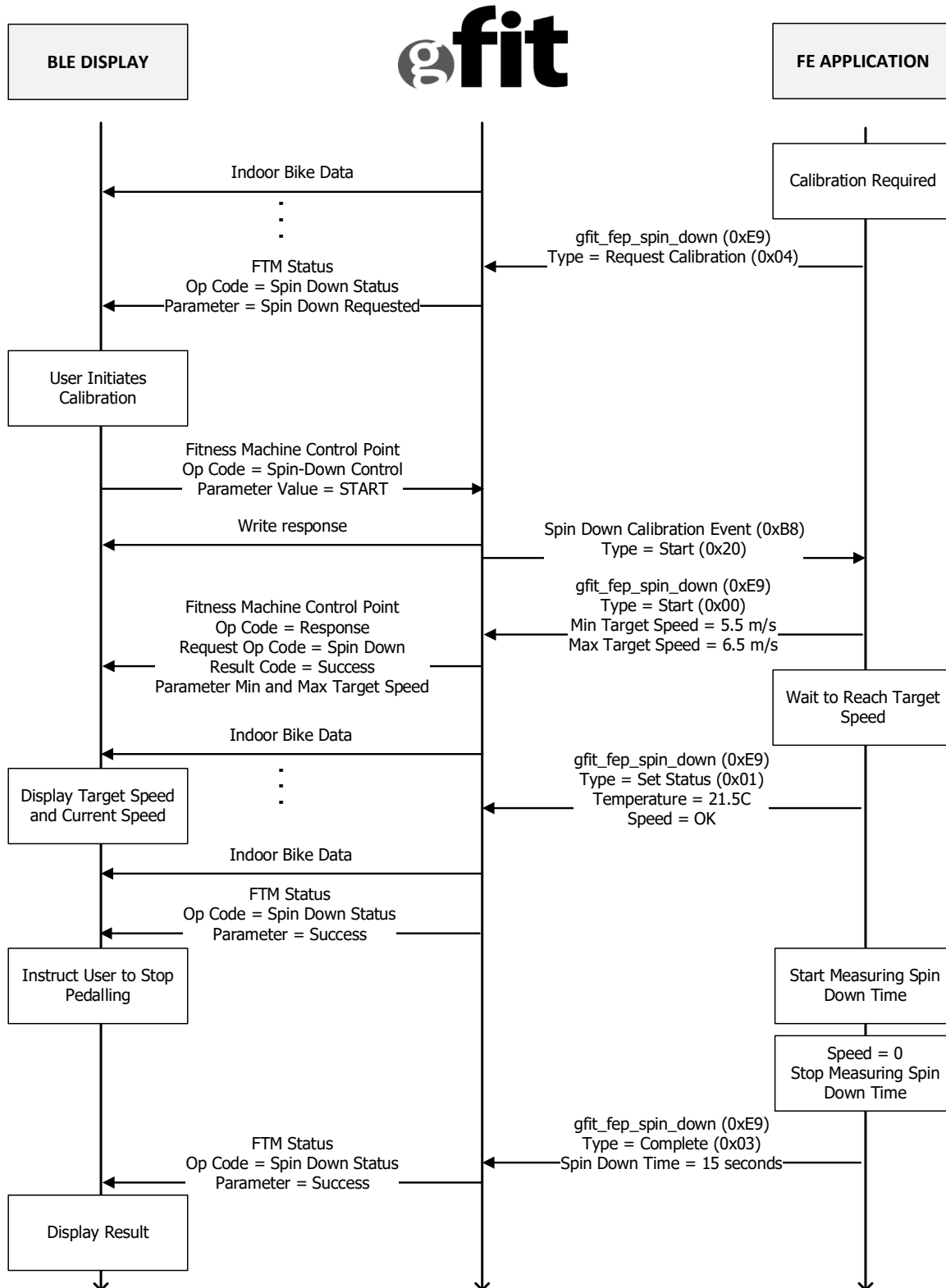
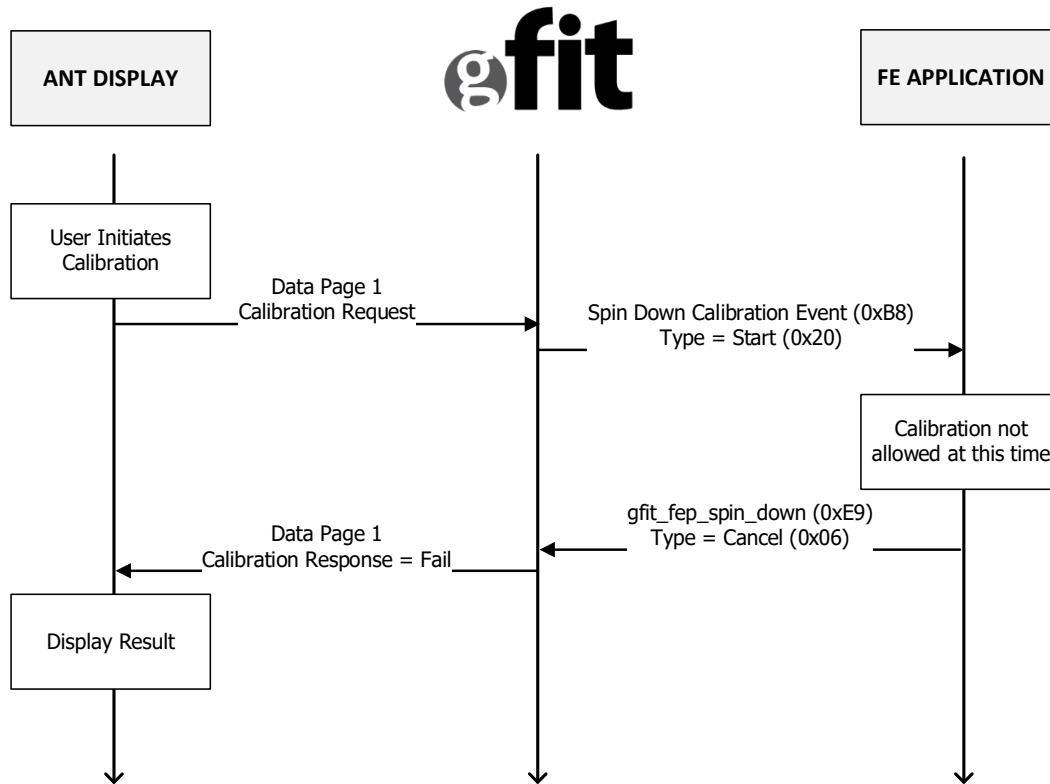


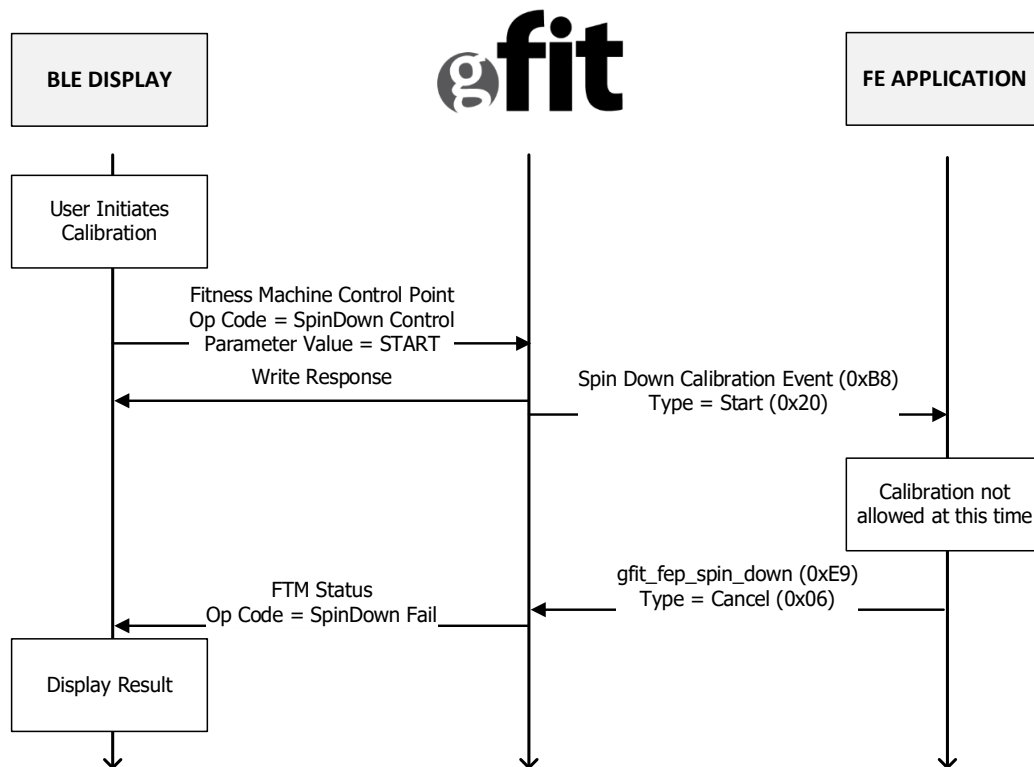
Figure 7-8. Example: Successful spin down calibration process (ANT)

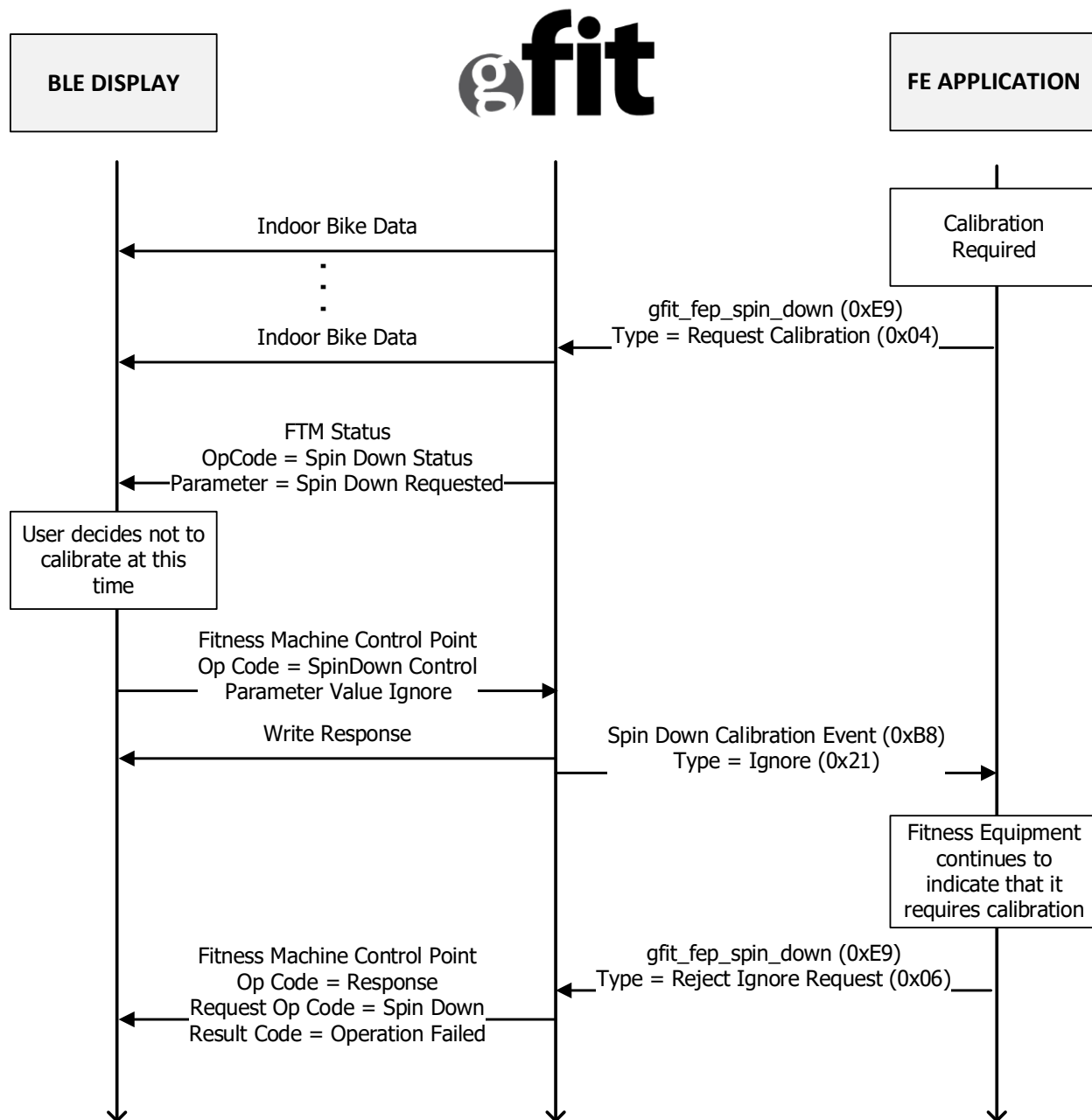


**Figure 7-9. Example: Successful spin down calibration process (BLE)**



**Figure 7-10. Example: Cancelled spin down calibration process (ANT)**



**Figure 7-11. Example: Cancelled spin down calibration process (BLE)****Figure 7-12. Example: Requested spin down calibration process (BLE)**

### 7.3.20 gfit\_fep\_send\_custom\_event (0xB0)

Command: Instruct the module to send an event to the display over the ANT+ FE-C channel and over BLE, using the G.FIT Custom Service.

Valid State: READY, IN USE, FINISHED

**Table 7-52. gfit\_fep\_send\_custom\_event command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Variable Set to 0x08 if Number of Transmissions (Byte 10) is specified or 0x07 if not.
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xB0 (gfit_fep_send_custom_event)
4 – 9/10	Payload	6 bytes	Byte 4 – Application defined op code, in the range E0-FF (inclusive) Bytes 5 – 9 - Application defined parameter Bytes 10 – Optional: Number of Transmissions (if not specified default of 8 transmission will be used)
10/11	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-53. gfit\_fep\_send\_custom\_event response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xB0 (gfit_fep_send_custom_event)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x05 - GFIT_ERROR_INVALID_OP_CODE 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.21 gfit\_fep\_send\_manufacturer\_specific\_page (0xB3)

Command: Instruct the module to send a manufacturer specific over the ANT+ FE-C channel. The manufacturer specific page will replace the page to be transmitted; it is not recommended to insert manufacturer specific pages in a short rotation interval as that may affect the transmission pattern for ANT+ FE-C. This command should only be called once per channel period of the ANT+ FE-C channel; the application can use the EVENT\_TX as cue to send the manufacturer specific page, or send the command based on asynchronous events (e.g. button). Valid manufacturer specific pages numbers are 0xE0 to 0xFF.

Valid State: READY, IN USE, FINISHED

**Table 7-54. gfit\_fep\_send\_manufacturer\_specific\_page command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x09
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xB3 (gfit_fep_send_manufacturer_specific_page)
4	Payload	1 byte	Manufacturer specific page number
5-11		1 byte	Manufacturer specific data
12	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-55. gfit\_fep\_send\_manufacturer\_specific\_page response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xB3 (gfit_fep_send_manufacturer_specific_page)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x11 - GFIT_ERROR_INVALID_MFG_SPECIFIC_PAGE_NUMBER 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.22 gfit\_fep\_set\_transmission\_pattern (0xDA)

Configuration Command: This command will set the transmission pattern used when sending FE page data. There are currently two supported patterns:

- default: broadcasts the fe\_specific page at 2Hz
- fast: broadcasts the fe\_specific page at 3Hz

This command is only supported for the rower equipment type.

Valid State: OFF

**Table 7-56. gfit\_fep\_set\_transmission\_pattern command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xDA (gfit_fep_set_transmission_pattern)
4	Payload	1 byte	Pattern Value: 0x00 (pattern_default) - Broadcasts the fe_specific page at 2Hz 0x01 (pattern_fast) - broadcasts the fe_specific page at 3Hz
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-57. gfit\_fep\_set\_transmission\_pattern response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xDA (gfit_fep_set_transmission_pattern)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x12 - GFIT_ERROR_INVALID_LENGTH 0x0D - GFIT_ERROR_INVALID_EQUIPMENT_TYPE 0x0E - GFIT_ERROR_EQUIPMENT_TYPE_NOT_SET See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.3.23 gfit\_fep\_set\_developer\_options (0xDB)

Configuration Command: This command is used to change the security level of the FTMS Control Point Characteristic. FTMP (Fitness Machine Profile) specification requires the FTMS Control Point Characteristic to have security level 2 or higher. To maintain compliance with the FTMP specification, leave this as default (0x00) to keep security level 2. Set to 0x01 to use security level 1.

Note that this command can only be used in the OFF state before transitioning to the READY state for the first time.

Valid State: OFF

Default: Security Level 2

**Table 7-58. gfit\_fep\_set\_developer\_options command message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0xE2 (command)
3	Sub-Message ID	1 byte	Set to 0xDB (gfit_fep_set_developer_options)
4	Payload	1 byte	Set to change the security level 0x00 – Security Level 2 (default) 0x01 – Security Level 1
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-59. gfit\_fep\_set\_developer\_options response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x05
2	Response ID	2 bytes	0xE0 (response)
3			0xFE (G.FIT response)
4	Command ID	2 bytes	0xE2 (command)
5			0xDB (gfit_fep_set_developer_options)
6	Response Code	1 byte	0x00 - GFIT_SUCCESS 0x01 - GFIT_ERROR_INCORRECT_STATE 0x07 - GFIT_ERROR_INVALID_PARAMETER 0x12 - GFIT_ERROR_INVALID_LENGTH See Table 7-1 for descriptions.
7	Payload	1 byte	Current State: 0x00 – State is INVALID 0x01 – State is OFF 0x02 – State is READY 0x03 – State is IN USE 0x04 – State is PAUSED/FINISHED
8	Checksum	1 byte	XOR of all previous bytes including the sync byte.



## 7.4 Requests (0xE1)

Each request is referenced by the related function in the G.FIT library and the associated sub-message ID.

### 7.4.1 gfit\_get\_version\_string (0xC0)

To request the version number from G.FIT, use the request version number message.

**Table 7-60. gfit\_get\_version\_string request message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x03
2	Request ID	2 bytes	Set to 0xE1 (request)
3			Set to 0xFE
4	Command ID	2 bytes	Set to 0xE2 (command)
5			Set to 0xC0 (gfit_get_version_string)
6	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-61. gfit\_get\_version\_string response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x0B
2	Message ID	1 byte	0xE2 (command)
3	Sub-Message ID	1 byte	0xC0 (gfit_get_version_string)
4-6	Payload	3 bytes	Software version prefix
7		1 byte	Major version number
8		1 byte	Software version separator
9-10		2 bytes	Minor version number
11-13		3 bytes	Software version postfix
14	Checksum	1 byte	XOR of all previous bytes including the sync byte.

#### 7.4.1.1 Version Numbering

Version numbers in GFIT will be formatted as: [Major].[Minor][Build Version Letter][Build Version Number]. The version string can be retrieved from the library by calling `gfit_get_version_string()` API or it can be retrieved from the NP by issuing a serial request (0xE1) for the version string (0xC0).

##### **B - "Build"**

Example: 4.00B00

This is the default letter for all normal, full version builds.

B indicates that it will work on GFIT modules and has the appropriate license. Normally, the letter suffix will be followed by 00 but this may be incremented to indicate minor non-API breaking bug fixes.

##### **E - "Evaluation"**

Example: 4.00E00

This letter indicates that this is the evaluation version of the G.FIT Network Processor application. This network processor can be run on the SK (Starter Kit) modules only and only for evaluation purposes.

##### **A - "Alpha"**

Example: 4.00A07

This letter indicates that this is an Alpha version.

Alpha versions are occasionally sent to customers for evaluating new features. Note that a version number such as 4.00A07 means that this is the 8th alpha build before version 4.00.

I.E. features which appear in version 4.00A07 will appear in 4.00B00 when they are approved.

## 7.5 Events (0xE0)

Each event is referenced by the related definition in the G.FIT library and the associated sub-message ID.

### 7.5.1 *gfit\_event\_receive\_heart\_rate\_ (0xB2)*

Event: An event that is sent to the MCU when G.FIT receives HR data from the wireless heart rate monitor it is currently connected to, as well as scan packets for heart rate monitors in the vicinity.

Requires Response: No

Depends on HR pairing mode: Proximity/Channel ID

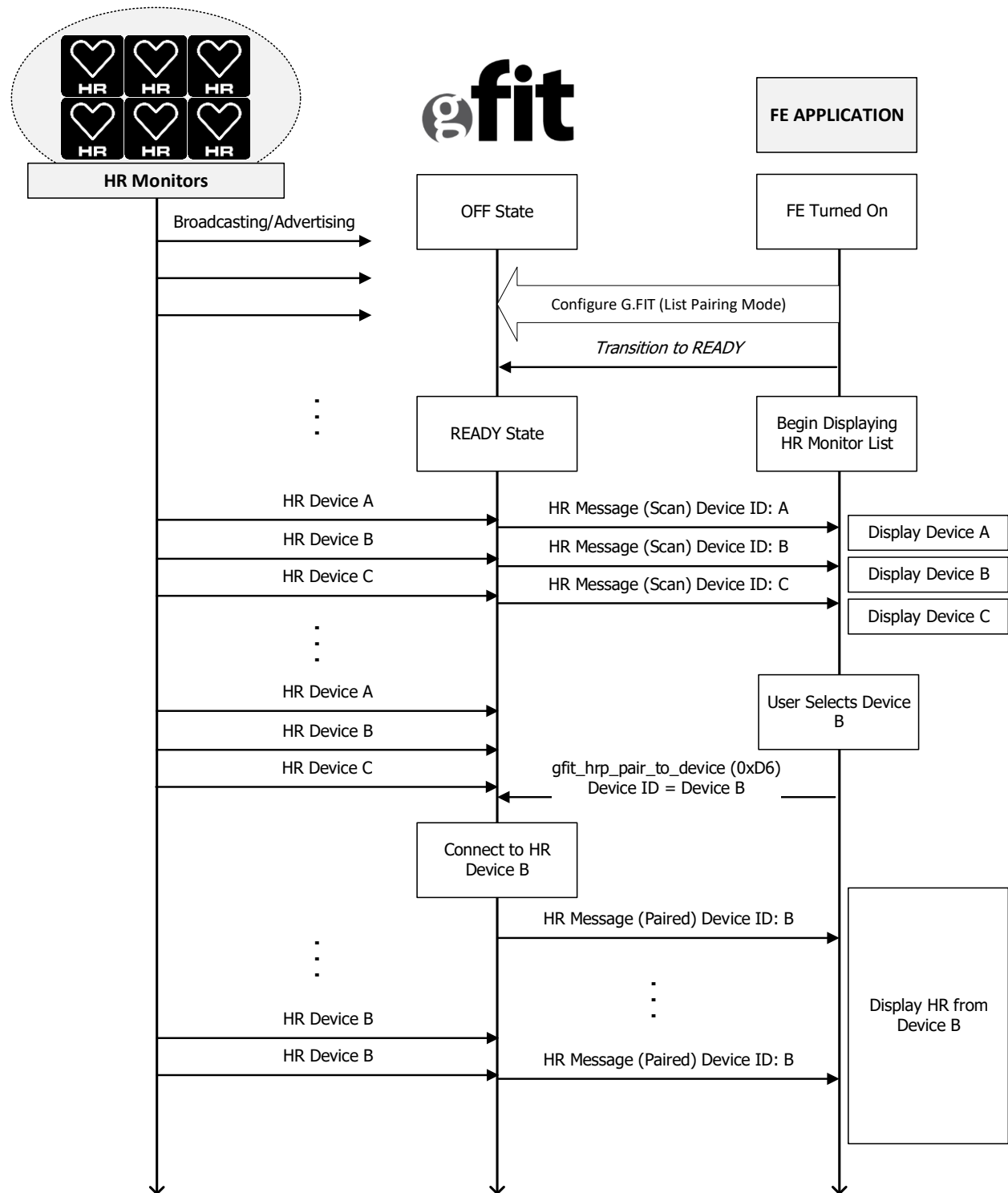
Corresponding G.FIT library functions: This serial message relates to `gfit_event_receive_heart_rate_scan_packet` and `gfit_event_receive_heart_rate_paired_packet` in the G.FIT library.

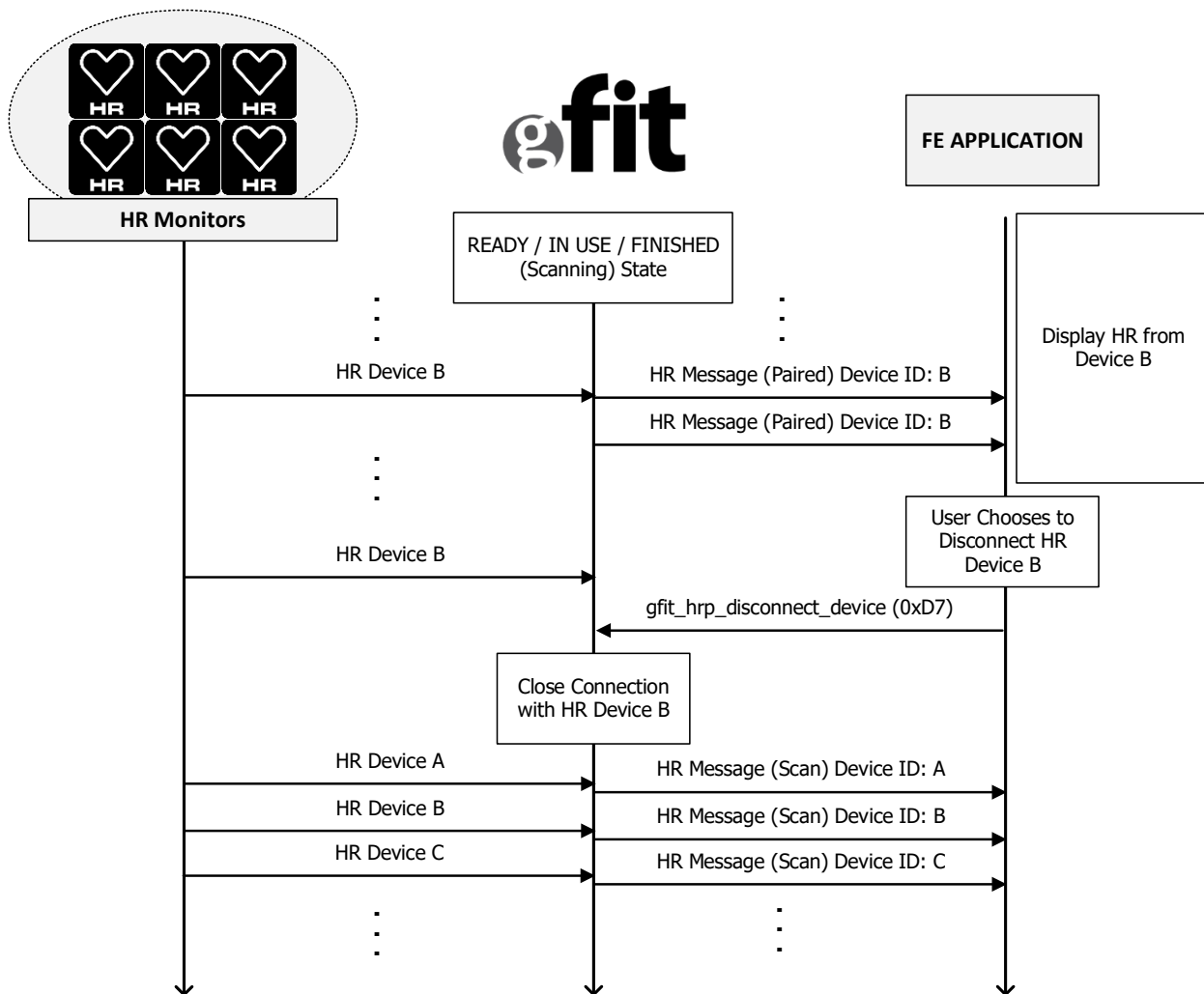
**Table 7-62. *gfit\_event\_receive\_heart\_rate\_ event message***

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Variable
2	Message ID	1 byte	0xE0 (event)
3	Sub-Message ID	1 byte	0xB2 ( <i>gfit_event_receive_heart_rate_</i> )
4	Flags	1 byte	Bit 0 – 0: Scan, 1: Pair Bit 1 – 0: ANT protocol, 1: BLE protocol Bits 2-7 – Reserved for future use (set to 0)
5	Heart Rate	1 byte	Heart rate in bpm. 0x00 indicates invalid
6	RSSI	1 byte	RSSI in dBm 0x7F indicates invalid
7	HRM ID	6 bytes	HRM ID LSB
8 – 11			...
12			HRM ID MSB
13	BLE Friendly Name Length	1 byte	Length of the BLE friendly name 0x00 indicates no name If the protocol is ANT, this will be set to 0x00
14 – N	BLE Friendly Name	N bytes	BLE friendly name (maximum of 20 bytes).
N + 1	Checksum	1 byte	XOR of all previous bytes including the sync byte.

#### 7.5.1.1.1 Proximity pairing mode



**7.5.1.1.2 List Pairing Mode****Figure 14. Sequence diagram: channel ID based pairing mode connection**



**Figure 7-15. Sequence diagram: disconnecting from connected monitor device B**

#### **7.5.1.1.3 Handling disconnected monitors**

It is recommended that the application maintains an internal timeout (recommended ~15 seconds) once paired to an HR monitor. The timeout should be restarted every time a paired HR event is received from G.FIT. If a paired monitor disconnects from G.FIT (for example, due to a battery outage, or the user moving out of proximity), this timeout allows the application to prompt the user that the monitor has been removed, and action is required. If the fitness equipment is still in a scanning state (READY, IN USE/FINISHED before scanning timeout), it should go back to the search/pairing screen to indicate that the monitor is no longer paired, and a new monitor may be paired.

### 7.5.2 Session command event (0xB4)

Event: An event that is sent to the MCU when G.FIT receives commands over BLE FTMS to reset, start/resume, stop and pause the current session. The application can use the session command events as a cue to change the G.FIT state (e.g., on receiving a start command, the application can instruct G.FIT to switch to IN USE state), or manage session data (e.g., reset timers and clear targets when a reset command is received).

Requires Response: Yes, see section 4.4.2.

Depends on Controllable Feature: Reset and/or Start/Stop/Pause

Corresponding G.FIT library functions: This serial message relates to gfit\_event\_reset, gfit\_event\_start\_or\_resume, gfit\_event\_stop and gfit\_event\_pause in the G.FIT library.

**Table 7-63. Session command event message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x02
2	Message ID	1 byte	0xE0 (event)
3	Sub-Message ID	1 byte	0xB4 (Session Command Event)
4	Command Type	1 byte	Indicates the type of session command 0x01 – Reset 0x02 – Start/Resume 0x03 – Stop 0x04 – Pause All other values are reserved for future use.
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.5.3 gfit\_event\_set\_target\_ (0xB5)

Event: An event that is sent to the MCU when G.FIT receives commands over ANT+ or BLE FTMS to set a target for the session.

Requires Response: Yes, see section 4.4.2.

Depends on Controllable Feature: Target Inclination/Target Resistance Level/Target Power/Target Heart Rate/Targeted Distance

Corresponding G.FIT library functions: gfit\_event\_set\_target\_inclination, gfit\_event\_set\_target\_resistance\_level, gfit\_event\_set\_target\_power, gfit\_event\_set\_target\_heart\_rate, and gfit\_event\_set\_targeted\_distance

**Table 7-64. gfit\_event\_set\_target\_ event message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x02 + Target Value Length
2	Message ID	1 byte	0xE0 (event)
3	Sub-Message ID	1 byte	0xB5 (gfit_event_set_target_)
4	Target Type	1 byte	Indicates the type of target. See Table 7-65.
5-(N-1)	Target Value	Variable	See Table 7-65
N	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-65. Target types**

Target type		Target value	
Value	Description	Length (bytes)	Description
0x10	Set Target Inclination	2	Target Inclination (0.01%, signed)
0x11	Set Target Resistance Level	2	Target Resistance Level (0.5%, signed)
0x12	Set Target Power	2	Target Power (0.25 W, signed)
0x13	Set Target Heart Rate	1	Target Heart Rate (bpm, unsigned)
0x14	Set Targeted Distance	4	Targeted Distance (m, unsigned)

**Note:** All multi-byte parameters are encoded as little-endian.



### 7.5.4 *gfit\_event\_set\_simulation\_parameters (0xB6)*

Event: An event that is sent to the MCU when G.FIT receives simulation parameters over ANT+ or BLE FTMS.

Requires Response: Yes, see section 4.4.2.

Depends on Controllable Feature: Bike Simulation

**Table 7-66. gfit\_event\_set\_simulation\_parameters event message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	11 bytes
2	Message ID	1 byte	0xE0 (event)
3	Sub-Message ID	1 byte	0xB6 (gfit_event_set_simulation_parameters)
4	Wind Resistance Coefficient	1 byte	Wind Resistance Coefficient (0.01 kg/m, unsigned) 0xFF indicates invalid
5-8	Wind Speed	4 bytes	Wind Speed (mm/s, signed) 0x7FFFFFFF indicates invalid †
9	Drafting Factor	1 byte	Drafting Factor (0.01 resolution, unsigned) 0xFF indicates invalid
10-11	Grade	2 bytes	Grade (0.01%, signed) 0x7FFF indicates invalid †
12-13	Coefficient of Rolling Resistance	2 bytes	Coefficient of Rolling Resistance (resolution 5x10 <sup>5</sup> , unsigned) 0xFFFF indicates invalid
14	Checksum	1 byte	XOR of all previous bytes including the sync byte.

† Wind speed and grade are both signed values, and therefore 0 is a valid value for both parameters. As such, neither parameter can be set to 0 by default and are instead initially set to 0x7FFFFFFF and 0x7FFF respectively.

### 7.5.5 *gfit\_event\_set\_user\_data (0xB7)*

Event: An event that is sent to the MCU when G.FIT receives user data over ANT+ or BLE (FTMS/G.FIT Custom Service), for use in simulation mode.

Requires Response: Yes, see section 4.4.2.

Depends on Controllable Feature: Bike Simulation

**Table 7-67. gfit\_event\_set\_user\_data event message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	12 bytes
2	Message ID	1 byte	0xE0 (event)
3	Sub-Message ID	1 byte	0xB7 (Set Simulation Parameters Event)
5-8	User Weight†	4 bytes	User Weight (5g resolution, unsigned) – ANT+/G.FIT Custom Service 0xFFFFFFFF indicates invalid
9-10	Bicycle Weight†	2 bytes	Bicycle Weight (50 g resolution, unsigned) – ANT+/G.FIT Custom Service 0xFFFF indicates invalid
11-14	Bicycle Wheel Circumference	4 bytes	Bicycle Wheel Circumference (0.1 mm, unsigned) – ANT+/BLE FTMS 0xFFFFFFFF indicates invalid
15	Gear Ratio	1 byte	Front:back gear ratio, from 0.03 (0x01) to 7.65 (0xFF) – ANT only 0x00 indicates invalid
14	Checksum	1 byte	XOR of all previous bytes including the sync byte.

† To be able to set user weight and/or bicycle weight, G.FIT must have the bike simulation controllable feature enabled.

### 7.5.6 Spin down Calibration Event (0xB8)

Event: An event that is sent to the MCU when G.FIT receives spin down calibration commands over ANT+ or BLE FTMS.

Requires Response: Yes, see section 7.3.19.

Depends on Controllable Feature: Spin Down Calibration

Corresponding G.FIT library functions: gfit\_event\_start\_spin\_down, gfit\_event\_ignore\_spin\_down, and gfit\_event\_cancel\_spin\_down

**Table 7-68. Spin down calibration event message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x02
2	Message ID	1 byte	0xE0 (event)
3	Sub-Message ID	1 byte	0xB8 (Spin Down Calibration Event)
4	Spin Down Event Type	1 byte	Indicates the type of spin down calibration event: 0x20 – Start (ANT+/FTMS) 0x21 – Ignore (FTMS) 0x22 – Cancel (ANT+/FTMS)
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Note:** All multi-byte parameters are encoded as little-endian.

### 7.5.7 gfit\_event\_entering\_bootloader (0xB9)

Event: An event that is sent to the MCU when G.FIT is about reset and enter bootloader mode, i.e., start the G.FIT firmware updater.

The event indicates which transport type will be used when the bootloader is started. G.FIT will reset immediately after this event.

Requires Response: No.

**Table 7-69. gfit\_event\_entering\_bootloader event message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	2 bytes
2	Message ID	1 byte	0xE0 (event)
3	Sub-Message ID	1 byte	0x 0xB9 (gfit_event_entering_bootloader)
4	Bootloader transport type	1 byte	0x00 – Invalid/Unknown 0x01 – BLE 0x02 – UART All other values are reserved for future use.
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.5.8 gfit\_event\_ble\_peripheral (0xBA)

Event: An event that is sent to the MCU when a BLE peripheral connect to or disconnects from G.FIT over BLE FTMS.

The event indicates whether a device connected to or disconnected from G.FIT.

Requires Response: No.

**Table 7-70. gfit\_event\_ble\_peripheral event message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	2 bytes
2	Message ID	1 byte	0xE0 (event)
3	Sub-Message ID	1 byte	0x 0xBA (gfit_event_ble_peripheral)
4	Bootloader transport type	1 byte	0x00 – BLE Peripheral Connected 0x01 – BLE Peripheral Disconnected
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.5.9 *gfit\_event\_custom (0xB1)*

Event: An event that is sent to the MCU when the display (e.g. training software, leaderboard application, bike computer) sends a custom message to G.FIT.

Requires Response: Yes, see section 4.4.2.

Depends on Controllable Feature: G.FIT Custom Events

**Table 7-71. gfit\_event\_custom event message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x07
2	Message ID	1 byte	0xE0 (event)
3	Sub-Message ID	1 byte	0xB1 (gfit_event_custom)
4	Op Code	1 byte	Application defined op code
5-9	Parameter	5 bytes	Application defined parameter
10	Checksum	1 byte	XOR of all previous bytes including the sync byte.

## 7.6 Other ANT commands

Standard ANT serial messages (detailed in the *ANT Message Protocol and Usage* document) can be used with G.FIT. This section highlights the most useful ones.

### 7.6.1 Read ANT ID

The ANT ID can be retrieved from the module using a serial command if the stock firmware is present. To do so, a Request Message (0x4D) must be sent. In the SoC library the ANT ID may be retrieved by reading the 32-bit value: NRF\_UICR->CUSTOMER[GFIT\_UICR\_CUST\_ANT\_ID\_OFFSET]

**Table 7-72. Request to retrieve ANT ID**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x02
2	Message ID	1 byte	Set to 0x4D
3	Channel	1 byte	Set to 0x01
4	Requested Message ID	1 byte	Set to 0x61
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

The response will arrive with Message ID 0x61 and can be parsed as:

**Table 7-73. ANT ID read response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x04
2	Message ID	1 byte	0x61
3	Payload	4 bytes	ANT ID[0]
4			ANT ID[1]
5			ANT ID[2]
6			ANT ID[3]
7	Checksum	1 byte	XOR of all previous bytes including the sync byte.

### 7.6.2 RSSI calibration offset

Used to obtain the 1-byte calibration offset from G.FIT. In the SoC library the ANT ID may be retrieved by reading the 32-bit value: NRF\_UICR->CUSTOMER[GFIT\_UICR\_CUST\_RSSI\_CAL\_OFFSET]

**Table 7-74. RSSI calibration offset message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	Set to 0x01
2	Message ID	1 byte	Set to 0xE4
3	Sub-Message ID	1 byte	Set to 0x02
4	Checksum	1 byte	XOR of all previous bytes including the sync byte.

**Table 7-75. RSSI calibration offset response message**

Byte #	Name	Length	Description
0	Sync	1 byte	Fixed value of 10100100 or 10100101 (Refer to the <i>Interfacing with ANT General Purpose Chipsets and Modules</i> document for details.)
1	Length	1 byte	0x02
2	Message ID	1 byte	0xE4
3	Sub-Message ID	1 byte	0x02
4	Payload	1 byte	Calibration Offset
5	Checksum	1 byte	XOR of all previous bytes including the sync byte.

## Appendix A - G.FIT SoC library

### A.1 Compiling the demo application

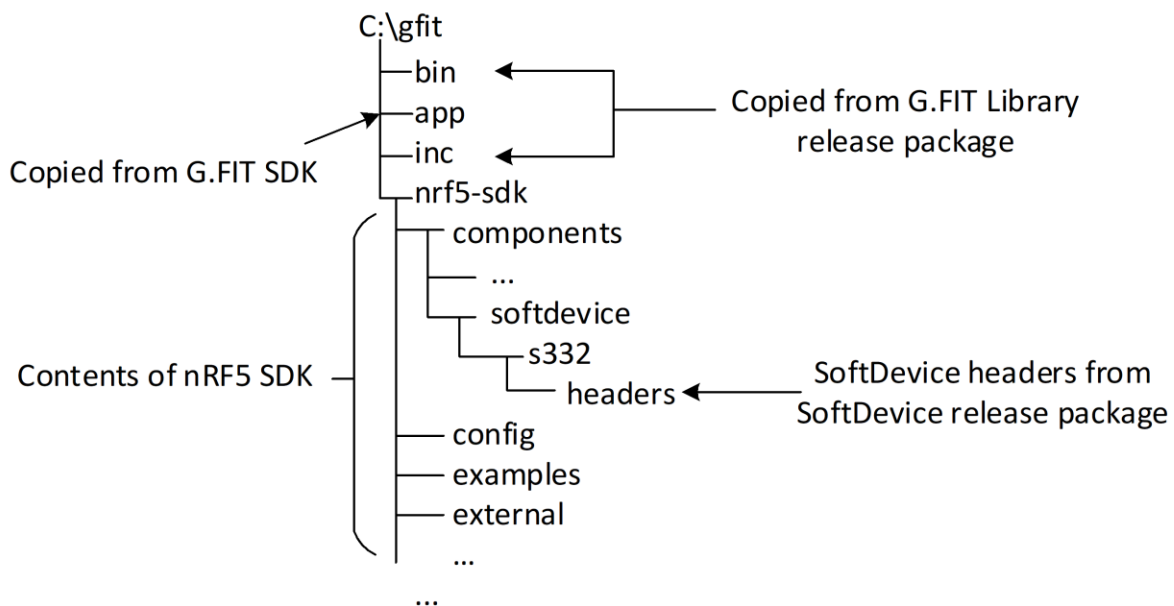
Unzip the contents of the G.FIT\_Library zip file inside the G.FIT Libraries Package into a directory on your hard drive (e.g. C:\gfit).

Download version 13.1 the Nordic nRF5 SDK for GCC from <http://developer.nordicsemi.com/>. Create a new directory called nrf5-sdk inside the gfit folder and install the Nordic nRF5 SDK in this directory.

Unzip the S332 SoftDevice package and copy the contents of the include directory directly into the components\softdevice\s332\headers directory of the nrf5-sdk.

Unzip the G.FIT SDK package and copy the app directory into the gfit directory.

The final directory structure should look like Figure A-1.



**Figure A-1. Final directory structure**

Build the demo\_gfit project. The first time you build the project, the compiler will point out two errors which serve as reminders that licenses must be obtained to use the S332 SoftDevice and the G.FIT libraries. Follow the instructions in the error messages to enable the evaluation keys and allow the project to build.

#### A.1.1 Library license key

The G.FIT library require a license key to operate. An evaluation key is available which will enable full functionality and is to be used for NON-COMMERCIAL USE ONLY. The license key required for the G.FIT library is different and independent from the license key G.FIT operates exclusively on Garmin G.FIT modules, once the Distribution Agreement for the G.FIT Module is completed, SoftDevice royalties will be waived for each instance of ANT SoftDevice used within a G.FIT module for commercial end product.

Further information about licensing can be found at: [www.thisisant.com/developer/ant/licensing](http://www.thisisant.com/developer/ant/licensing).

License validation can extend the library initialization time to up to 100 ms.



### ***A.1.2 Compiler compatibility***

The G.FIT static library is built with GNU ARM Embedded Toolchain version 4.9 2015q3, available at:

<https://launchpad.net/gcc-arm-embedded/4.9/4.9-2015-q3-update>.

It uses the settings in Table A-1. See example application Makefile for further details.

**Table A-1. GCC specific compiler options**

Option	Description
-mcpu=cortex-m4	Cortex-M4
-mfloat-abi=hard -mfpu=fpv4-sp-d16	The libraries are built using hardware floating point numbers. Application settings must match.
-mthumb	Thumb mode

## Appendix B - Fitness equipment console simulator

### B.1 Introduction

G.FIT Simulator is an application that demonstrates how G.FIT would operate for a given type of fitness equipment. As a test tool it allows developers to quickly configure the G.FIT module (D52QSKM6IA-A attached to a USB interface board) settings to see their effect.

### B.2 Installing ANT USB interface board driver

Before downloading the required software, you will need to create an account on <http://www.thisisant.com>

Download the ANT USB Interface Board Driver for Windows package from <http://www.thisisant.com/developer/resources/downloads> and extract the entire contents onto the hard drive.

**Note:** The ANT USB Interface Board drivers are unsigned. Systems that require signed drivers for installation (e.g. Windows 8, Windows 10) are required to boot with driver signature enforcement disabled to complete the installation process. This procedure requires restarting the computer and booting up in a special mode. This additional step can be found by conducting a Google search.

Run the USBXpressInstaller.exe file contained in the folder. Install the drivers in the desired location.

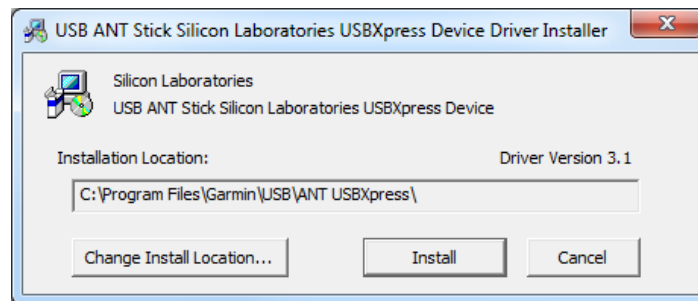


Figure B-1. Install USB interface driver

A warning message may appear that indicates that Windows can't verify the publisher of the driver software. Click 'Install this driver software anyway' to continue.

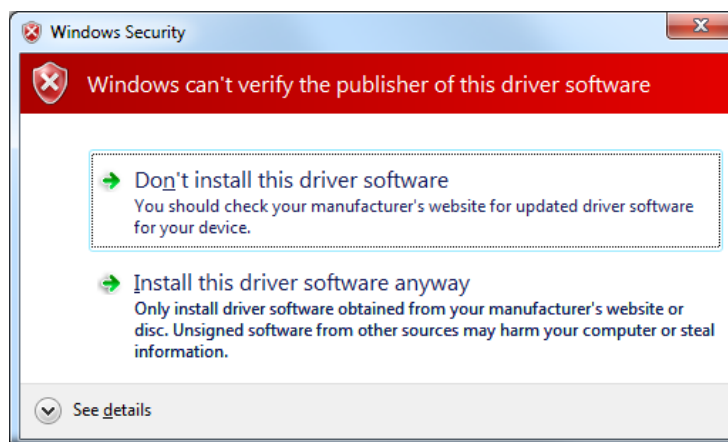
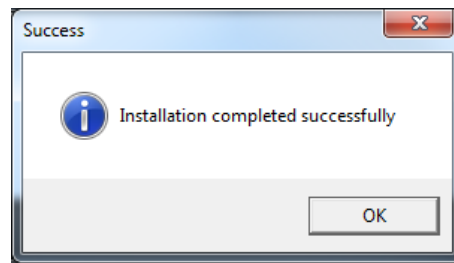


Figure B-2. Bypass publisher verification

A window will indicate the drivers have installed correctly.



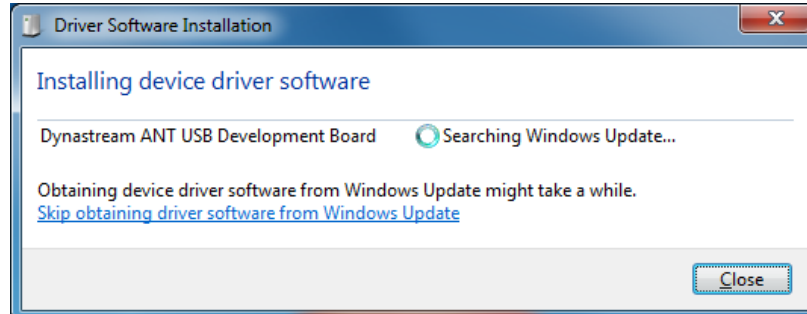
**Figure B-3. Driver installation success**

Connect the G.FIT module to the USB interface board and insert into a USB port.



**Figure B-4. G.FIT module mounted on a USB interface board**

The Driver Software Installation wizard should pop up and begin a search for drivers; the wizard will indicate the USB device is 'Ready to Use' when it detects the installed drivers on the PC.



**Figure B-5. Driver installation wizard**

### B.3 Basic operation

This section provides a high-level description of how to operate the fitness equipment console simulator. It may be useful to reference section 4.1 (Fitness Equipment States), when reading this basic operation section.

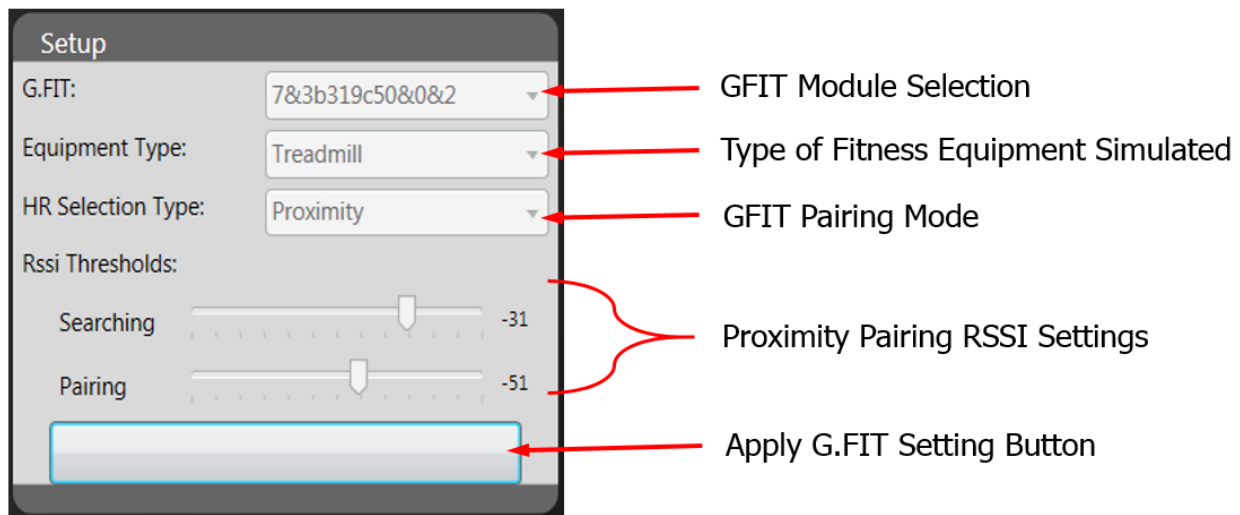
Start by opening the application (FitnessEquipmentSimulator.exe). Figure B-6 shows the user interface that is provided by the simulator. The menus on the left side of the application are used to modify settings, and to view the commands that are being sent to G.FIT. The main part of the application provides a simulated fitness equipment console implemented using a G.FIT module.



**Figure B-6. Overview of the fitness equipment console simulator**

#### B.3.1 Initial G.FIT setup

The Setup section contains the advanced settings related to G.FIT configuration. These settings must be adjusted before the simulated console is turned on. Figure B-7 shows the settings that must be set before the console main power is turned on. After each setting is selected the user must press the 'Apply G.FIT Setting Button' to complete that portion of the configuration.



**Figure B-7. Overview of initial G.FIT setup**

**B.3.1.1 G.FIT**

This shows which G.FIT module the simulator is connected to. If there is more than one G.FIT module connected to the computer, then select which module to use.

**B.3.1.2 Equipment type**

This setting determines which type of fitness equipment is simulated.

**B.3.1.3 HR selection type**

Select the pairing mode used to connect the heart rate monitor to G.FIT. See section 4.2 for more details on the best choice for the given use case.

**B.3.1.4 RSSI thresholds**

When pairing to a heart rate monitor using proximity mode, the simulator will pair with a heart rate monitor automatically based on the proximity to G.FIT. See section 4.2.1.1 for more details on selecting RSSI values.

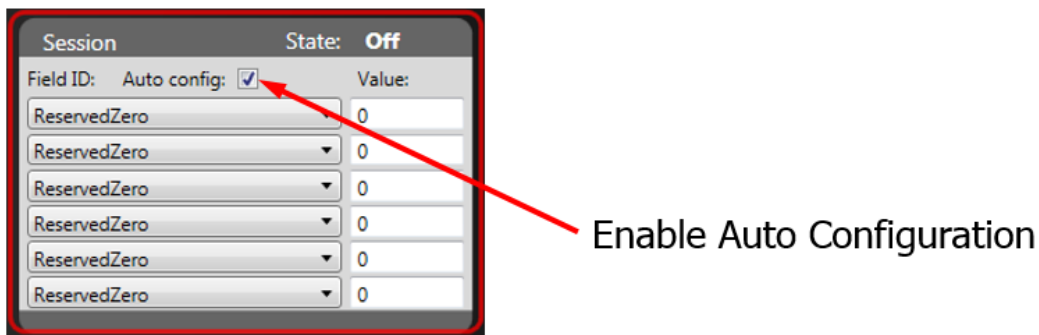
The RSSI values shall be set so that Pairing < Searching.

**B.3.1.5 Configuring FE metrics**

Specific field IDs will need to be selected appropriately depending on which fitness equipment type is set. For details on the setting FE metrics see section 7.3.11. For a list of field IDs for the given fitness equipment type see section 7.3.11.2.

**B.3.1.6 Auto configuration**

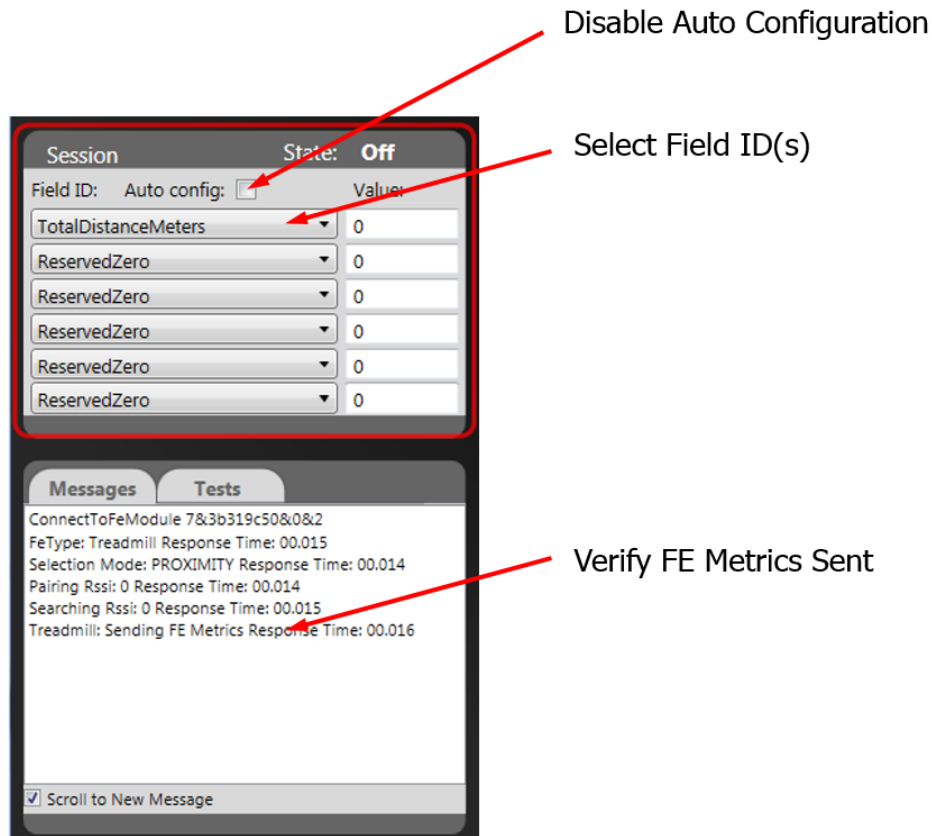
This provides the quickest and easiest method of configuring the FE metrics for the given type of fitness equipment. Ensure that the Auto config checkbox is selected as in Figure B-8. When selected, the simulator will configure every FE metric listed in the appropriate table (section 7.3.11.2) for the given fitness equipment type. Press the 'Apply G.FIT Setting Button' seen in Figure B-7 to complete G.FIT configuration.



**Figure B-8. Auto configuration of FE metrics**

**B.3.1.7 Manual configuration**

If you would like to simulate a type of fitness equipment that does not fully support all types of FE metrics, then manual configuration is required. Manual configuration allows the user to individually select the FE metrics that will be supported. To use manual configuration, ensure that the Auto config checkbox is deselected as in Figure B-9. Field IDs can then be selected to configure which FE metrics G.FIT will support. It is important to select the proper field IDs for the given type of fitness equipment (see section 7.3.11.2). When a field ID is selected, verify that it was sent properly and that there were no errors returned. When finished configuring the FE metrics manually, press the 'Apply G.FIT Setting Button' seen in Figure B-7 to complete G.FIT configuration.



**Figure B-9. Manual configuration of FE metrics**

### ***B.3.2 Turning on the simulated console***

To turn on the simulated console, press the red circle  button in the upper right section of the console. Turning on the console puts it into FE state READY. This has the following effects:

- Turns on the ANT+ FE-C channel
- Begins searching for ANT+ heart rate monitors
- Begins searching for BLE heart rate monitors
- Turns on the BLE FTMS peripheral

### ***B.3.3 Pairing a heart rate monitor***

Depending on the HR selection type value specified in the Setup section of the application, the pairing of a HR strap will be completed in one of two ways.

#### **B.3.3.1 Proximity pairing**

The simulated console will pair with a heart rate monitor automatically based on the proximity to G.FIT. See section 4.2.1 for more details.

#### **B.3.3.2 Pairing from a list of devices (channel ID based pairing)**

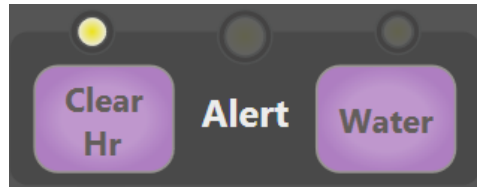
This allows the user to select which heart rate monitor to connect to via a user interface. In list mode the simulated console will display all the heart rate monitors in range of G.FIT and allow the user to select which heart rate monitor they want to pair to. See section 4.2.2 for more details.

### Selecting a heart rate monitor in list

When there is more than one heart rate monitor displayed on the screen, the user will have to select the one that they are using. Pressing the Up or Down buttons on the console will change which heart rate monitor is highlighted on the screen. Up to four heart rate monitors can be displayed on the screen at once. When the user has their heart rate monitor highlighted, they must press the Select button to connect to it.

#### **B.3.4 Successfully paired heart rate monitor**

After a heart rate monitor is paired, the connection is identified by the yellow light on the console as seen in Figure B-10.



**Figure B-10. Indicator for a connected heart rate monitor**

The current heart rate monitor data is displayed on the console display. The user is given information on their current heart rate, the type of heart rate monitor being used (ANT/BLE), the device ID, and the RSSI.

#### **B.3.5 Disconnecting a paired heart rate monitor**

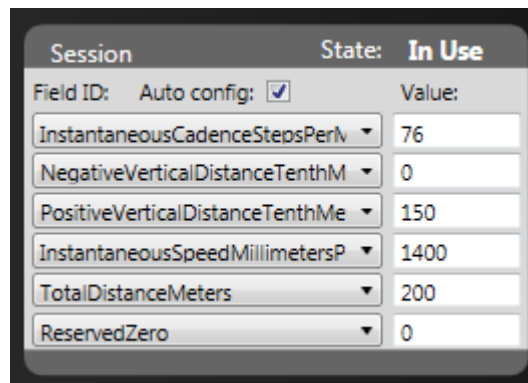
If the user wants to disconnect their heart rate monitor or possibly another heart rate monitor that was accidentally connected to, they can press the Clear Hr button. After a heart rate monitor has been disconnected the simulated console returns to the pairing screen.

#### **B.3.6 Starting a workout/session**

A session can be started with or without a heart rate monitor paired. To start a session, press the Start button. The state will change from READY to IN USE.

##### **B.3.6.1 Modifying FE metrics**

The simulated data that is sent over ANT+ FE-C channel and BLE FTMS peripheral can be modified at any time during the session. Simply change any of the values within the 'Session' section of the application. To update a FE metric, simply select the required field ID and adjust its value. The simulator will allow up to 6 values to be adjusted simultaneously. See Figure B-11 for an example of setting some FE metrics for a treadmill. When in state IN USE the simulator will automatically update the FE metric elapsed time to match the value that is shown in the simulated console in addition to the other FE metrics specified by the user.



**Figure B-11. Modifying FE metrics**

### **B.3.6.2 Sending commands to a leaderboard**

These are custom messages that are sent from the simulated console to the leaderboard and are not part of the ANT+ FE-C profile.

#### **Request water**

Press the Water button to trigger the water request. Press the Water button again to cancel the water request.

#### **Flip user's leaderboard tile**

Press the Flip button to send a tile flip command. This allows the user to flip their tile on the leaderboard to display more data.

### ***B.3.7 Pausing/resuming a session***

Active sessions (i.e. FE state IN USE) can be paused by pressing the Stop button once. Doing so pauses the session and puts the console into FE state FINISHED (PAUSED). To resume the session the user will press the Start button and the console will go back into FE state IN USE.

### ***B.3.8 Ending a session***


If the user wishes to end their session from:

- an active session (FE State IN USE) then the user will press the Stop button twice.
- a paused session (FE State FINISHED (PAUSED)) then the user will press the Stop button once.

Once the session has ended the console will go back into FE state READY.

If there was a connected heart rate monitor when the session was ended, that heart rate monitor will be disconnected.

### ***B.3.9 Turning off the simulated console***

To turn off the simulated console, press the red circle  button in the upper right section of the console. To successfully turn off the simulated console, G.FIT must be in FE state READY or FINISHED (PAUSED). Turning off the console puts it into FE state OFF. When in FE state OFF, G.FIT can be reconfigured as described in section 4.3.

### ***B.3.10 Enabling/disabling full screen mode***

To enter full screen, simply press:



To exit out of full screen mode, hover the mouse over the lower right section to make the exit full screen button appear, press this button to exit full screen.





## 8 Heading 1 - Template guidance

### 8.1 Heading 2

#### 8.1.1 Heading 3

#### 8.1.1.1 Heading 4

#### 8.1.1.1.1 Heading 5

### 8.2 Headings use sentence style capitalization

#### 8.2.1 Like this

#### 8.2.2 Not Like This

Text style is 'ANT Normal'.

### 8.3 Captions, tables, figures, and equations

Captions are now correctly formatted by default and should automatically use the 'Caption' style (Tahoma, 8.5pt, bold, centred, line spacing single, before 0pt, after 10pt). Use sentence style capitalization, and no period at the end.

Inserting a new table should automatically give you the correct shading and borders. Modifying tables by deleting or adding rows should also preserve the correct striping. To get the correct font formatting within the table, apply the 'ANT Table Header', 'ANT Table Text Left' and 'ANT Table Text Center' styles. If a table does not have the correct shading, apply the table style 'Striped Table'.

**Table 8-1. Table captions go over table**

Header	Header text	ANT Table text header
Table text left	Table text centre	Table text left Text should automatically center itself vertically.
		Autofit table to window – unless for very small tables

Equations are images (layout-> 'inline' and centre by applying the 'Caption' style to the equation). Equation captions go under the equation.

$$ScaleFactor = \frac{KnownDistance}{MeasuredDistance}$$

**Equation 8-1. Deriving the scale factor**

Images use 'inline' layout. Centre them by applying the 'Caption' style to the image. Captions go under the image. Note that images used to be 'top and bottom' format, but this caused a lot of confusion when editing as images can jump around and get lost depending on where their anchor is positioned on the page. Using inline formatting should remove this headache.



**Figure . All captions include chapter number**

## **8.4 Bullets and numbered text**

Bulleted text looks like this:

- Item one using style ANT Bullets 1
- Item two
  - Level 2 bullets using style ANT Bullets 2

Itemised lists look like this:

- e. Item 1 using style ANT List 1
- f. Item 2
  - i. Level 2 list using style ANT List 2
  - ii. So pretty

When you have a procedure you're writing instructions for, do this:

- 1) Use this style for instruction steps. The style is called ANT Instructions one and is based on the custom multi-level list 'Instruction steps'. As with all ANT styles you can find it in the style pane; or copy & paste from here.

Indented text ANT Indent 1, to match ANT Instructions 1. Use this if you need another paragraph for the same step in the instructions. The next paragraph will automatically be ANT Instructions 1, which you can change if needed.

- 2) Right click the number and select 'Restart at 1' if the list starts at a higher number.

- a) Instruction sub-steps look like this.

Indented text ANT Indent 2, to match ANT Instructions 2. Use this if you need another paragraph for the same sub-step in the instructions. The next paragraph will automatically be ANT Instructions 2, which you can change if needed.

If you need to include some sample code:

```
## Use style 'ANT Code Sample'
Lgrjalæjr
Sglsg
Hello World!
```

## 8.5 Notes, bold text, italicized text, and document names

Formatting can be used to communicate extra information about what you're saying. For example, **really important information that you don't want readers to miss** should be styled with 'Strong'.

**Note:** The 'Strong' style is also used to help people notice your notes 😊. Apply it to 'Note:' at the start of the line.

Don't italicize anything except for document names. For those use style 'Document Name', e.g., see the *ANT Message Protocol and Usage* document. Don't bother italicizing the docs listed in the related documents section.

## 8.6 Watermarks

To modify 'Preliminary – subject to change' watermarks, double click the header or footer area, then select the image and modify as desired.

## **9 Second 'Heading 1' heading**

Each top-level heading will automatically start on a new page. No need to add page breaks.

Appendices use the styles 'Appendix H1,2,3, or 4' as the heading levels.

## Appendix C - Sample appendix title

Use style Appendix H1 for top level headings.

### C.1 Appendix H2

Blah blah blah

#### C.1.1 Appendix H3

Blah

##### C.1.1.1 Appendix H4

Lbsfh

### C.2 Table and Figure captions for use in Appendices

Copy and paste this caption for tables in the appendix, then edit as needed. Select the caption and press F9 to update the numbering. Using this rather than the main body captions will make sure that the chapter number is correct.

#### Table C-1. Sample appendix table caption

Use this one for figures.

#### Figure C-1. Sample appendix figure caption

Getting the numbers to start again at 1 at the start of a new appendix section (e.g., A-1, B-1, C-1 rather than A-1, B-2, C-3) is done by inserting the two hidden fields that are at the end of the main section title. To make these visible so you can copy them in more easily to any appendix H1 heading that is missing them, select the row, and toggle field codes. They should look like this:

• Appendix C- Sample appendix title • SEQ-AppendixFigure \r0 \h \\* •  
MERGEFORMAT • SEQ-AppendixTable \r0 \h \\* • MERGEFORMAT •

#### Figure C-2. Field codes for resetting table and figure counts in appendices

Tables and figures in the appendices are not included in the Table of Figures or Table of Tables.