Interfacing with ANT
General Purpose
Chipsets and Modules
Copyright Information and Usage Notice

This information disclosed herein is the exclusive property of Dynastream Innovations 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 Dynastream Innovations 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 Dynastream Innovations Inc. unless such commitment is expressly given in a covering document.

The Dynastream Innovations 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 Dynastream 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 Dynastream 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 Dynastream was negligent regarding the design or manufacture of the Product.

©2007 Dynastream Innovations Inc. All Rights Reserved.
# TABLE OF CONTENTS

1 INTRODUCTION ........................................................................................................................................... 4

2 SYNCHRONOUS SERIAL INTERFACE ........................................................................................................ 9
  2.1 DESCRIPTION........................................................................................................................................... 9
  2.2 INTERCONNECT ...................................................................................................................................... 9
  2.3 PORT SELECT (PORTSEL) .................................................................................................................... 9
  2.4 SPEED SELECT (BR1, BR2, BR3) ............................................................................................................ 9
  2.5 SUSPEND MODE CONTROL (SUSPEND) ............................................................................................... 9
  2.6 SYNCHRONOUS PORT CONTROL (RTS) .............................................................................................. 9
  2.7 SLEEP ENABLE (SLEEP) ....................................................................................................................... 9
  2.8 LINK LAYER PROTOCOL ....................................................................................................................... 9
    2.8.1 Characteristics ................................................................................................................................ 10
    2.8.2 Message Structure .......................................................................................................................... 10
    2.8.3 Message Details ............................................................................................................................. 11
  2.9 ANT MESSAGES...................................................................................................................................... 12

3 ASYNCHRONOUS SERIAL INTERFACE ................................................................................................... 5
  3.1 DESCRIPTION....................................................................................................................................... 5
  3.2 INTERCONNECT ..................................................................................................................................... 5
  3.3 PORT SELECT (PORTSEL) .................................................................................................................... 5
  3.4 SPEED SELECT (BR1, BR2, BR3) ............................................................................................................ 5
  3.5 SUSPEND MODE CONTROL (SUSPEND) ............................................................................................... 5
  3.6 ASYNCHRONOUS PORT CONTROL (RTS) ............................................................................................ 5
  3.7 SLEEP ENABLE (SLEEP) ....................................................................................................................... 5
  3.8 LINK LAYER PROTOCOL ....................................................................................................................... 5
    3.7.1 Characteristics ................................................................................................................................ 5
    3.7.2 Message Structure .......................................................................................................................... 5
    3.7.3 Message Details ............................................................................................................................. 5
  3.9 ASYNCHRONOUS MESSAGING WITH BYTE FLOW CONTROL ............................................................... 5
  3.9.1 Characteristics ................................................................................................................................ 5
  3.9.2 Message Structure .......................................................................................................................... 5
  3.9.3 Message Details ............................................................................................................................. 5
  3.9.4 USING AN EPSON MCU AS A HOST CONTROLLER ......................................................................... 5

4 INTRODUCTION ........................................................................................................................................... 4
1 Introduction

ANT is a 2.4GHz bidirectional wireless sensor network communications technology optimized for transferring low-data rate, low-latency data between multiple ANT-enabled devices. The ultra-low power consumption of ANT guarantees an extended battery life even from low-capacity supplies such as a coin cell battery, enabling use in heart rate monitors, bicycle computers and wrist watches. ANT’s small size and low implementation cost allow effortless integration into the tiny form factor of wrist watches, PDAs and mobile phones.

ANT provides carefree handling of the Physical, Network and Transport OSI layers. In addition, it incorporates key low-level security features that form the foundation for user-defined sophisticated network security implementations. ANT ensures adequate user control while considerably lightening computational burden in providing a simple yet effective wireless networking solution.

The interface between ANT and the Host application has been designed with the utmost simplicity in mind so that ANT can be easily and quickly implemented into new devices and applications. The encapsulation of wireless protocol complexity within the ANT chipsets vastly reduces the burden on the application host controller, allowing a low-cost 4-bit or 8-bit microcontroller to establish and maintain complex wireless networks. Data transfers can be scheduled in a deterministic or ad-hoc fashion. A burst mode allows for the efficient transfer of large amounts of stored data to and from a PC or other computing device. ANT aggressively balances functionality, cost, size and power consumption within the constraints of a wireless sensor network. Typical applications include sensor integration, tagging systems, remote monitoring etc.

The intent of this document is to detail the interface requirements between an application microcontroller and the ANT products. It provides insight into both interface signals and physical layer data formats.

A complete description of the ANT Message protocol is found in ANT Message Protocol and Usage document.
2 Asynchronous Serial Interface

2.1 Description
The Host MCU and ANT may communicate using the asynchronous mode of the serial interface. The connection diagram is shown below in Figure 2-1. Asynchronous mode is selected by the PORTSEL input being tied low.

Refer to Section 3 for details on the alternative synchronous mode.

2.2 Interconnect
The asynchronous serial interface between ANT and the Host MCU is shown below.

![Figure 2-1: Asynchronous Mode Connections](image)

Please note that all UART communication settings are for one start bit, one stop bit, 8 bits of data and no parity. Data is sent and received LSBit first.

2.3 Port Select (PORTSEL)
The PORTSEL signal should be tied low for asynchronous serial mode.

2.4 Speed Select (BR1, BR2, BR3)
The baud rate of the asynchronous communication between the Host and ANT is controlled by the speed select signals BR1, BR2 and BR3. Not all of these inputs are available on all ANT products. Please refer to the datasheets for products of interest for more information.

The table below shows the relationship between the states of the speed select signals and the corresponding baud rates.

<table>
<thead>
<tr>
<th>BR3*</th>
<th>BR2</th>
<th>BR1</th>
<th>Baud Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4800</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>19200</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>38400</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>50000</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1200</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2400</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>9600</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>57600</td>
</tr>
</tbody>
</table>

* Not available on all ANT devices. It is assumed to be of value 0 when not available.
Note that baud rate may have a significant impact on system current consumption. Refer to the Electrical Specifications section of the ANT product of interest for current consumption figures.

2.5 SUSPEND mode control (SUSPEND)

The assertion of the SUSPEND signal will cause ANT to terminate all RF and serial port activity and power down. This will happen immediately, regardless of the state of the ANT system. This signal provides support for use in USB applications, where USB devices are required to quickly enter a low-power state through hardware control.

Entering and exiting from the SUSPEND mode require the use of the SLEEP signal, in addition to the SUSPEND signal. The assertion of SUSPEND is only recognized if SLEEP is also asserted at the time. De-assertion of the SLEEP signal is the only method for exiting from SUSPEND mode, as shown in Figure 2-2. Following exit, all previous transactions and configurations will be lost – ANT will be in its power-up state.

![Figure 2-2: SUSPEND signal usage](image)

2.6 Asynchronous Port Control (RTS)

When ANT is configured in asynchronous mode, a full duplex asynchronous serial port is provided with flow control for data transmission from the Host to ANT. The flow control is performed by the RTS signal, which conforms to standard hardware flow control CMOS signal levels. The signal may therefore be attached to a PC serial port (with use of an RS-232 level shifter), or to any other RS-232 device. The RTS signal is de-asserted for approximately 50 µs after each correctly formatted message has been received. This RTS signal duration is independent of the baud rate. Incorrect messages or partial messages are not acknowledged.

When ANT raises the RTS signal high, the Host MCU may not send any more data until the RTS signal is lowered again. There is no flow control for data being transmitted from ANT to the Host controller, and therefore the Host controller must be able to receive data at any time.
2.7 SLEEP ENABLE (SLEEP)

The SLEEP input signal allows ANT to sleep when the serial port is not required, helping conserve power. This control mechanism is illustrated below in Figure 2-3.

This signal is essential for power savings in the nRF24AP1, but has less of an effect for the dual chip solutions.

![Diagram of ANT Sleep control](image)

If the SLEEP signal is not used, then it must be tied low. In this configuration, the ANT system will never sleep and will always be ready to receive data. The SUSPEND functionality cannot be used if the SLEEP signal is not used.

The SLEEP and RTS signals only affect the data being transferred from the Host MCU to ANT. ANT will send data to the Host, when available, regardless of the state of these two signals.

**NOTE:** The RTS signal is raised by ANT after the last byte of a message has been received, and ANT will therefore lose any bytes that were sent, or in the process of being sent, before the RTS signal is acted upon by the Host MCU, and the transmission is halted. To avoid this problem, either the messages need to be spaced apart by the Host MCU or 0-pad bytes need to be added to the end of each message being transmitted to handle whatever byte pipeline is in place. For example, when considering PC communication, two 0-bytes must be appended to every message, since PCs interpret CTS at the hardware level rather than the software level. ANT will discard 0-pad bytes received. This issue usually occurs only when using burst transfers from the Host to ANT and high data rates are expected.

2.8 Link Layer Protocol

2.8.1 Characteristics

The ANT interface protocol has the following characteristics:

- Binary protocol
- Packets are of variable length
- Each packet contains an 8-bit Checksum
- Asynchronous data is transmitted with 1 start, 8 data bits, 1 stop bit and no parity, with standard CMOS level signaling
- Full duplex serial port
### 2.8.2 Message Structure

ANT and the Host MCU communicate by transmitting messages to each other. Each message is formatted as shown below.

<table>
<thead>
<tr>
<th>SYNC</th>
<th>LENGTH</th>
<th>ID</th>
<th>DATA_1</th>
<th>DATA_2</th>
<th>...</th>
<th>DATA_N</th>
<th>CHECKSUM</th>
<th>Opt. Zero Pad1</th>
<th>Opt. Zero Pad2</th>
</tr>
</thead>
</table>

Each variable-length message is sent starting with the SYNC byte and ending with the CHECKSUM. Bytes are sent LSB first.

### 2.8.3 Message Details

<table>
<thead>
<tr>
<th>Byte #</th>
<th>Bit #</th>
<th>Name</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7:0</td>
<td>SYNC</td>
<td>1 Byte</td>
<td>Fixed SYNC field = 10100100 (MSB:LSB)</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>LENGTH</td>
<td>1 Byte</td>
<td>Number of data bytes in the message</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>ID</td>
<td>1 Byte</td>
<td>Data type identifier: 0 : Invalid 1..255 : Valid data type ID</td>
</tr>
<tr>
<td>3..N+2</td>
<td>-</td>
<td>DATA_1 ... DATA_N</td>
<td>N Bytes</td>
<td>Message data bytes</td>
</tr>
<tr>
<td>N+3</td>
<td>-</td>
<td>CHECKSUM</td>
<td>1 Byte</td>
<td>XOR of all previous bytes (including SYNC)</td>
</tr>
<tr>
<td>N+4, N+5</td>
<td>-</td>
<td>Optional Zero PAD Bytes</td>
<td>1or2 Bytes</td>
<td>Zero PAD bytes may be required in conjunction with flow control when doing BURST transfers.</td>
</tr>
</tbody>
</table>

The following is an example of how to encode/decode an ANT serial message.

ANT_OpenChannel(1) -> SerialData (0xA4, 0x01, 0x4B, 0x01, 0xEF)

The details of the contents of this example serial message are shown in the table below.

<table>
<thead>
<tr>
<th>Byte #</th>
<th>Name</th>
<th>Length</th>
<th>Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SYNC</td>
<td>1 Byte</td>
<td>0xA4</td>
<td>SYNC is always 0xA4</td>
</tr>
<tr>
<td>1</td>
<td>LENGTH</td>
<td>1 Byte</td>
<td>0x01</td>
<td>Number of Data bytes in this message = 1</td>
</tr>
<tr>
<td>2</td>
<td>ID</td>
<td>1 Byte</td>
<td>0x4B</td>
<td>ANT_OpenChannel message ID is 0x4B</td>
</tr>
<tr>
<td>3</td>
<td>DATA_1</td>
<td>1 Byte</td>
<td>0x01</td>
<td>There is 1 Data Byte in this message: This byte is Channel #. It has been set to Channel = 1</td>
</tr>
<tr>
<td>4</td>
<td>CHECKSUM</td>
<td>1 Byte</td>
<td>0xEF</td>
<td>0xA4 xor 0x01 xor 0x4B xor 0x01 = 0xEF</td>
</tr>
</tbody>
</table>

### 2.9 ANT Messages

Refer to ANT Message Protocol and Usage document for details on the different types of messages and overall ANT protocol description.
3 Synchronous Serial Interface

3.1 Description
This section details the synchronous serial interface between ANT and a Host MCU. This mode is selected by connecting the PORTSEL input high.

Please refer to Section 2 for details on the alternative asynchronous mode.

When operating in synchronous mode careful attention to reset behavior is required to prevent inadvertent deadlock conditions between ANT and the Host MCU. Please see Section 3.11 for more details on this subject.

In synchronous mode, ANT uses a half-duplex synchronous master serial interface with message flow control. The Host must be configured as a synchronous slave. The interface is meant to accommodate either a hardware synchronous slave port or a simple I/O control on the Host MCU. The Host MCU retains full control of the message flow and can halt incoming messages as required.

3.2 Interconnect
The synchronous serial interface between ANT and the Host MCU is shown below.

![Figure 3-1: Synchronous Mode Connections](image)

3.3 Port Select (PORTSEL)
The PORTSEL signal should be connected to logic high for synchronous serial mode.

3.4 Flow Control Select (SFLOW)
The Flow Control Select signal is used to configure the synchronous serial port for either Byte or Bit flow control.

<table>
<thead>
<tr>
<th>SFLOW</th>
<th>Flow Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Byte Flow Control</td>
</tr>
<tr>
<td>1</td>
<td>Bit Flow Control</td>
</tr>
</tbody>
</table>
Please note that Byte flow control assumes that the Host contains synchronous communication hardware which can be configured for synchronous slave communication. Bit flow control can be used when all serial lines are implemented in software on the Host MCU. The differences between byte and bit flow control are detailed in the sections below.

3.5 Operating Mechanism
A basic description of the communications mechanism follows.

- The synchronous serial port provided by ANT is a half duplex synchronous master, with full flow control in both directions of communication.
- Flow control of data transmitted to the Host MCU is controlled by the SRDY signal, and flow control of data transmitted to ANT is controlled by the master SCLK signal.
- By default, the Host is in receive mode and ANT is in transmit mode. In this state, ANT will forward all incoming radio messages to the Host as they become available. The Host uses the SRDY flow control to signal its readiness for incoming messages.
- If the Host MCU wishes to send a message to ANT (for example to open a communications channel), the Host indicates it wishes to enter into transmit mode by asserting the SMSGRDY signal.
- SRDY must be asserted for communication to begin.
- In either receive or transmit mode, ANT always transmits the first byte of information output from SOUT, which is clocked with the SCLK signal (see Section on Electrical Specifications for details of clock frequency). The LSBit of this byte indicates the direction of future bytes (0 : Message Receive, ANT → Host; 1: Message Transmit, Host → ANT)
- If the Host MCU is in receive mode (default), additional message bytes will be transmitted the same way as the first byte, from ANT → Host MCU.
- If the Host MCU is in transmit mode, it must output its data to the ANT SIN input at the clock rate provided by the ANT SCLK signal.
- Data is transmitted least-significant-bit first

3.6 Power Up / Power Down
ANT will automatically place itself into deep sleep mode when all radio channels are closed and there is no activity on the SMSGRDY input signal. The Host MCU should ensure these conditions during times that the ANT radio is not required in order to maximize product battery life.

Upon every power up, the host must apply the Synchronous Reset sequence as show in Section 3.11.

3.7 Link Layer Protocol

3.7.1 Characteristics
The ANT interface protocol has the following characteristics:

- Binary protocol
- Packets are of variable length
- Each packet contains an 8-bit Checksum
- Data is transmitted LSB first

3.7.2 Message Structure
ANT and the Host MCU communicate by transmitting messages to each other. Each message is formatted as shown below.

<table>
<thead>
<tr>
<th>Sync R/W</th>
<th>MSG LENGTH</th>
<th>MSG ID</th>
<th>DATA_1</th>
<th>DATA_2</th>
<th>......</th>
<th>DATA_N</th>
<th>CHECKSUM</th>
</tr>
</thead>
</table>

### 3.7.3 Message Details

<table>
<thead>
<tr>
<th>Byte #</th>
<th>Bit #</th>
<th>Description</th>
<th>Length</th>
<th>Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7:1</td>
<td>SYNC</td>
<td>7 bits</td>
<td></td>
<td>Fixed SYNC field = 1010010 (MSB:LSB)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>R/W</td>
<td>1 bit</td>
<td></td>
<td>0 : Write (Message ANT → Host)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 : Read (Message Host → ANT)</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>LENGTH</td>
<td>1 Byte</td>
<td></td>
<td>Number of data bytes in the message (Length should be between 1 and 9)</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>ID</td>
<td>1 Byte</td>
<td></td>
<td>Data type identifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 : Invalid</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1..255 : Valid data type ID</td>
</tr>
<tr>
<td>3..N+2</td>
<td>-</td>
<td>DATA_1 ... DATA_N</td>
<td>N Bytes</td>
<td></td>
<td>Message data bytes (There may be between 1 and 9 data bytes)</td>
</tr>
</tbody>
</table>

The following is an example of how to encode a message to send from the Host to ANT.

ANT_OpenChannel(1)

✈ SerialData (0xA5) // 0xA5 is read indicating that the Host may send a message to ANT

SerialData(0x01, 0x4B, 0x01, 0xEE) // The Host can then send the 4-byte message to ANT

<table>
<thead>
<tr>
<th>Byte #</th>
<th>Name</th>
<th>Length</th>
<th>Direction</th>
<th>Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SYNC</td>
<td>1 Byte</td>
<td>ANT→Host</td>
<td>0xA5</td>
<td>SYNC is 0xA5 for a Host→ANT transaction</td>
</tr>
<tr>
<td>1</td>
<td>LENGTH</td>
<td>1 Byte</td>
<td>ANT→ANT</td>
<td>0x01</td>
<td>Number of data bytes in this message = 1</td>
</tr>
<tr>
<td>2</td>
<td>ID</td>
<td>1 Byte</td>
<td>ANT→ANT</td>
<td>0x4B</td>
<td>ANT_OpenChannel message ID is 0x4B</td>
</tr>
<tr>
<td>3</td>
<td>DATA_1</td>
<td>1 Byte</td>
<td>ANT→ANT</td>
<td>0x01</td>
<td>There is 1 Data Byte in this message: This byte is Channel #: It has been set to Channel = 1</td>
</tr>
<tr>
<td>4</td>
<td>CHECKSUM</td>
<td>1 Byte</td>
<td>ANT→ANT</td>
<td>0xEE</td>
<td>0xA5 xor 0x01 xor 0x4B xor 0x01 = 0xEE</td>
</tr>
</tbody>
</table>

The following is an example of how the Host would decode a message received from ANT.

✈ SerialData (0xA4, 0x02, 0x52, 0x01, 0x03, 0xF6) // The Host receives 6-byte message

✈ Channel_Status(1, 3) // Decodes into a channel status message

<table>
<thead>
<tr>
<th>Byte #</th>
<th>Name</th>
<th>Length</th>
<th>Direction</th>
<th>Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SYNC</td>
<td>1 Byte</td>
<td>ANT→Host</td>
<td>0xA4</td>
<td>SYNC is 0xA4 for an ANT→Host transaction</td>
</tr>
<tr>
<td>1</td>
<td>LENGTH</td>
<td>1 Byte</td>
<td>ANT→ANT</td>
<td>0x02</td>
<td>Number of data bytes in this message = 2</td>
</tr>
<tr>
<td>2</td>
<td>ID</td>
<td>1 Byte</td>
<td>ANT→ANT</td>
<td>0x52</td>
<td>Channel_Status Message is 0x52</td>
</tr>
<tr>
<td>3</td>
<td>DATA_1</td>
<td>1 Byte</td>
<td>ANT→ANT</td>
<td>0x01</td>
<td>There are 2 data bytes in this message: This byte is Channel #: Channel = 1</td>
</tr>
<tr>
<td>4</td>
<td>DATA_2</td>
<td>1 Byte</td>
<td>ANT→ANT</td>
<td>0x03</td>
<td>This byte is the status. Status = 3, which indicates the channel is tracking.</td>
</tr>
<tr>
<td>5</td>
<td>CHECKSUM</td>
<td>1 Byte</td>
<td>ANT→Host</td>
<td>0xF6</td>
<td>0xA5 xor 0x02 xor 0x52 xor 0x01 xor 0x03 = 0xF6</td>
</tr>
</tbody>
</table>
3.8 Synchronous Messaging with Byte Flow Control

Byte flow-control mode is used when a synchronous hardware serial port is available.

The Host MCU flow-control signal \texttt{SRDY} can either be implemented with a software controlled IO line, or in some cases may be controlled by the Host's hardware serial port (e.g. EPSON MCU USART support for \texttt{SRDY}).

Data bits change state on the falling edge of SCLK and are read on the rising edge of SCLK. This is true for transactions in either direction.

The first byte in the transaction sequence is always sent from ANT to the Host MCU. The first bit of the first byte dictates the direction for the remaining bytes in the transaction.

Shown below in Figure 3-2 to Figure 3-5 are examples of transactions between the Host and ANT in byte synchronous mode.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{ ANT -> Host Transaction (Hardware SRDY) }
\caption{Figure 3-2: ANT -> Host Transaction (Hardware \texttt{SRDY})}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{ Host -> ANT Transaction (Hardware SRDY) }
\caption{Figure 3-3: Host -> ANT Transaction (Hardware \texttt{SRDY})}
\end{figure}
3.9 Synchronous Messaging with Bit Flow Control

If no hardware serial port is available on the Host MCU, ANT can be still be controlled using bit flow control. Using this method, the serial lines are implemented with software controlled IO lines. All of the signaling at the message transaction level remains the same as above, but it becomes the following at the byte level.
It is important to note that the Host MCU will do all bit processing on the rising edge of the SCLK signal, with the exception when the byte is being transmitted from the Host MCU to ANT, where the first data bit will need to be asserted before the first clock edge. The final rising edge of the byte transaction will be the event to drive byte processing.

3.10 Serial Enable Control (ANT → Host)

The SEN signal, which is driven by ANT, will be asserted by ANT prior to message transmission. It can therefore be used as a serial port enable signal, which is useful in cases where the Host serial port requires hardware activation.
3.11 Synchronization

In order for the Host MCU to guarantee synchronization with ANT in startup conditions, a reset sequence must be applied to ANT. This is applicable to Synchronous mode communication only.

3.12 Using an Epson MCU as a Host controller

The interface has been designed to easily communicate with an EPSON microcontroller with a built-in USART. The EPSON should be configured in the following manner.
Figure 3-10: Example EPSON configuration for Byte Synchronous serial interface

For proper implementation of the above setup:

1. The GPIO connected to \( \text{SEN} \) must be configured as input.
2. The GPIO connected to \( \text{SMSGRDY} \) must be configured as output.
3. The EPSON USART must be configured as a synchronous slave.
4. Configure the \( \text{SRDY} \) pin to be controlled by the USART. Note that the register flag that causes the \( \text{SRDY} \) pin to go low while waiting for a new byte must not be cleared until the \( \text{SEN} \) signal is seen to go low from the ANT device. This is to avoid causing a synchronization condition as described in Section 3.11 above.

With the above setup, the EPSON MCU hardware USART will control the signaling on the \( \text{SRDY} \), \( \text{SIN} \), \( \text{SOUT} \) and \( \text{SCLK} \) pins while the firmware (on the MCU) will handle signals on GPIOs connected to \( \text{SEN} \) and \( \text{SMSGRDY} \).