Application Note

PAN1026 ETU

TOSHIBA TC35661 Chiron

Microcontroller software implementation example

Version 1.2, 2014-12-04

Contents

1 Introduction

The purpose of this document is to explain one way to use a microcontroller as a host for the PAN1026 without an RTOS to establish a Bluetooth connection with a PAN1026 ETU USB stick. If your controller is capable to run the FreeRTOS you may want to use the Toshiba API. In this case please refer to the links and resources available on the Panasonic website.

In the following instruction the ATMEL ATmega128L microcontroller is used, but you might feel free to use any other microcontroller with at least one UART interface. However, it is recommended to use a microcontroller with more than one UART interface initially, in order to monitor the communication between the microcontroller and the PAN1026.

The following instruction assumes that the required software EASYSPP is already installed on your computer and the initialization of two PAN1026 ETU USB sticks is completed. If this is not the case, please download the software (via resources and links as mentioned above) and refer to the TOSHIBA quick start guide to install the software and to initialize the PAN1026 ETU USB sticks.

The attached sourcecode is not, by any stretch of imagination, complete. It does only show how to initialize a PAN1026, establish a communication to a predefined device and transmit a test sentence, while sending debug information to a computer terminal. It has to be improved by the customer himself to develop more functionality and flexibility.

2 General

2.1 Feedback

Despite thorough testing, there may be unpredictable software/hardware problems or ambiguities in the guide. In that case, please write a short message to wireless@eu.panasonic.com and we will try to help or solve the issue.

2.2 Disclaimer

Description of hardware, software and other information in this document is intended to illustrate the functionality of the referred Panasonic product for demonstration purpose only.

Provided Example Source Code shall not be used or incorporated into any products or systems whose manufacture, use or sale is prohibited under any applicable laws or regulations.

Panasonic Industrial Devices Europe GmbH (PIDEU) provides Example Source Code on an “as is” basis without any right to technical support or updates and without warranty of any kind on a free of charge basis according to § 516 German Civil Law (BGB) including without limitation, any warranties or conditions of title, non – infringement, merchantability, or fitness for a particular purpose. Customer acknowledges that Example Source Code may bear defects and errors.

To the maximum extent allowable by Law PIDEU assumes no liability whatsoever including without limitation, indirect, consequential, special, or incidental damages or loss, including without limitation loss of profits, loss of opportunities, business interruption and loss of data.

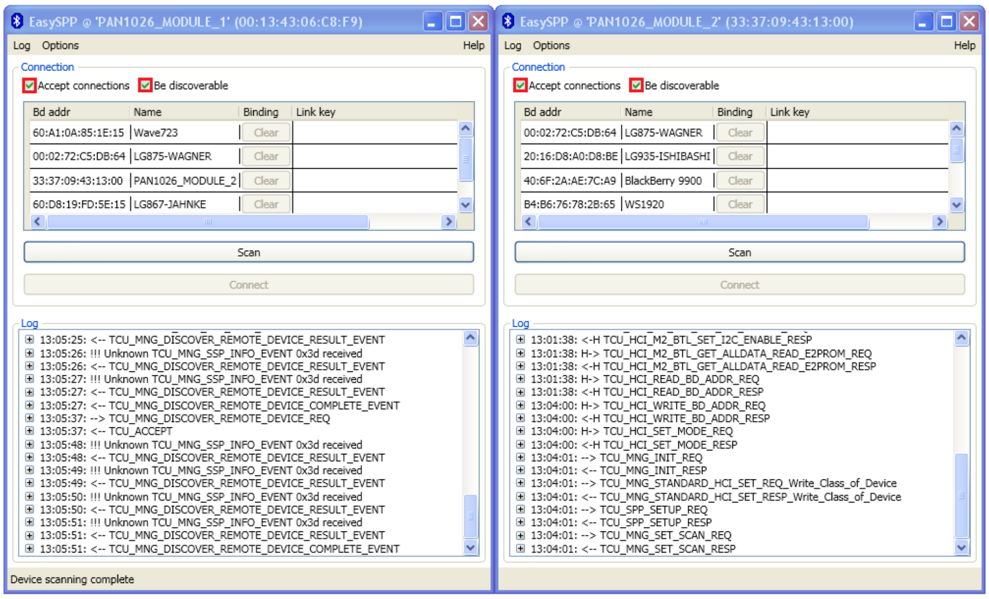
2.3 Document Status

The information contained herein is subject to change without notice.

3 Instruction

3.1 Getting the log files

Please launch the EasySPP software from the start menu and initialize two PAN1026 ETU USB sticks as it is described in the TOSHIBA TC35661 Chiron quick start guide. Furthermore, check the boxes “Accept connections” and “Be discoverable” for both devices.



Bd addr	Name	Binding	Link key
60:A1:0A:85:1E:15	Wave723	Clear	
00:02:72:C5:DB:64	LG875-WAGNER	Clear	
33:37:09:43:13:00	PAN1026_MODULE_2	Clear	
60:D8:19:FD:5E:15	LG867-JAHNKE	Clear	

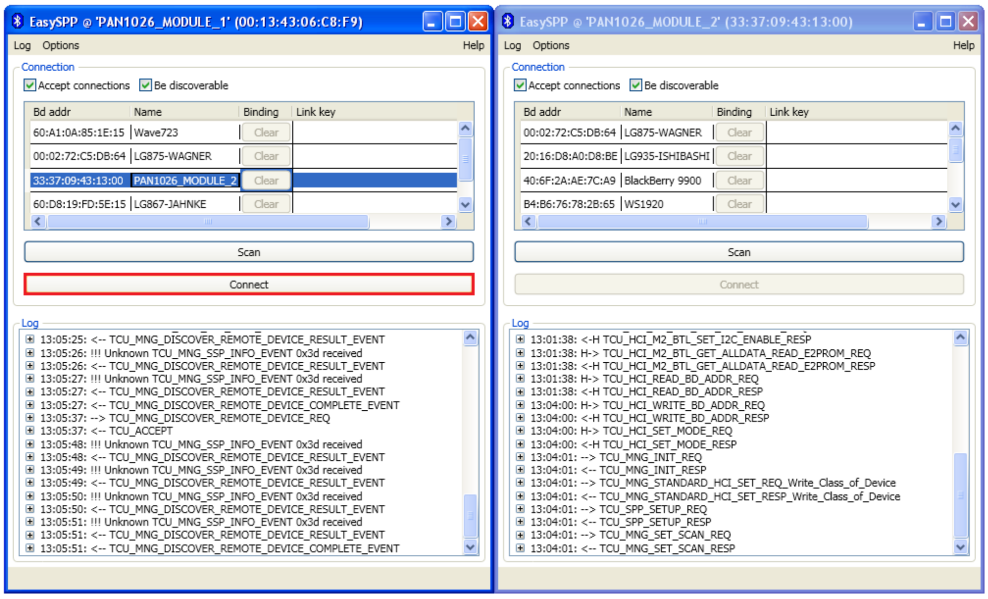
```
13:05:25: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:26: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:26: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:27: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:27: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:27: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_COMPLETE_EVENT
13:05:37: --> TCU_MNG_DISCOVER_REMOTE_DEVICE_REQ
13:05:37: <-- TCU_ACCEPT
13:05:48: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:48: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:49: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:49: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:50: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:50: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:51: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:51: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:51: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_COMPLETE_EVENT
```

Device scanning complete

Bd addr	Name	Binding	Link key
00:02:72:C5:DB:64	LG875-WAGNER	Clear	
20:16:D8:A0:D8:BE	LG935-ISHIBASHI	Clear	
40:6F:2A:AE:7C:A9	BlackBerry 9900	Clear	
B4:B6:76:78:2B:65	WS1920	Clear	

```
13:01:38: <-H TCU_HCI_M2_BTL_SET_I2C_ENABLE_RESP
13:01:38: H-> TCU_HCI_M2_BTL_GET_ALLDATA_READ_E2PROM_REQ
13:01:38: <-H TCU_HCI_M2_BTL_GET_ALLDATA_READ_E2PROM_RESP
13:01:38: H-> TCU_HCI_READ_BD_ADDR_REQ
13:01:38: <-H TCU_HCI_READ_BD_ADDR_RESP
13:04:00: H-> TCU_HCI_WRITE_BD_ADDR_REQ
13:04:00: <-H TCU_HCI_WRITE_BD_ADDR_RESP
13:04:00: H-> TCU_HCI_SET_MODE_REQ
13:04:00: <-H TCU_HCI_SET_MODE_RESP
13:04:01: --> TCU_MNG_INIT_REQ
13:04:01: <-- TCU_MNG_INIT_RESP
13:04:01: --> TCU_MNG_STANDARD_HCI_SET_REQ_Write_Class_of_Device
13:04:01: <-- TCU_MNG_STANDARD_HCI_SET_RESP_Write_Class_of_Device
13:04:01: --> TCU_SPP_SETUP_REQ
13:04:01: <-- TCU_SPP_SETUP_RESP
13:04:01: --> TCU_MNG_SET_SCAN_REQ
13:04:01: <-- TCU_MNG_SET_SCAN_RESP
```

After that, choose one module as host and establish a Bluetooth connection.



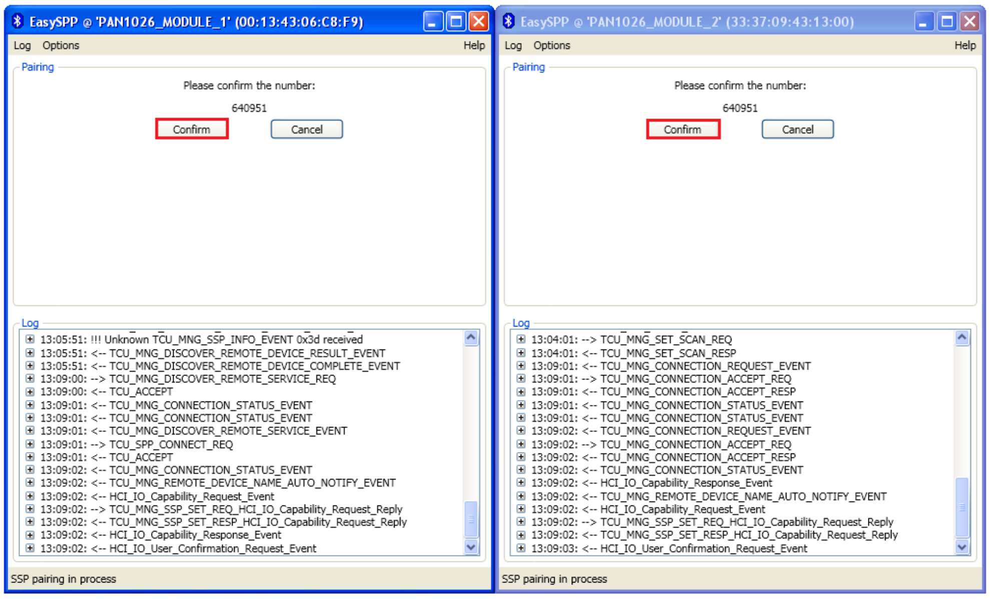
Bd addr	Name	Binding	Link key
60:A1:0A:85:1E:15	Wave723	Clear	
00:02:72:C5:DB:64	LG875-WAGNER	Clear	
33:37:09:43:13:00	PAN1026_MODULE_2	Clear	
60:D8:19:FD:5E:15	LG867-JAHNKE	Clear	

```
13:05:25: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:26: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:26: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:27: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:27: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:27: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_COMPLETE_EVENT
13:05:37: --> TCU_MNG_DISCOVER_REMOTE_DEVICE_REQ
13:05:37: <-- TCU_ACCEPT
13:05:48: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:48: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:49: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:49: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:50: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:50: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:51: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:51: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:51: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_COMPLETE_EVENT
```

Bd addr	Name	Binding	Link key
00:02:72:C5:DB:64	LG875-WAGNER	Clear	
20:16:D8:A0:D8:BE	LG935-ISHIBASHI	Clear	
40:6F:2A:AE:7C:A9	BlackBerry 9900	Clear	
B4:B6:76:78:2B:65	WS1920	Clear	

```
13:01:38: <-H TCU_HCI_M2_BTL_SET_I2C_ENABLE_RESP
13:01:38: H-> TCU_HCI_M2_BTL_GET_ALLDATA_READ_E2PROM_REQ
13:01:38: <-H TCU_HCI_M2_BTL_GET_ALLDATA_READ_E2PROM_RESP
13:01:38: H-> TCU_HCI_READ_BD_ADDR_REQ
13:01:38: <-H TCU_HCI_READ_BD_ADDR_RESP
13:04:00: H-> TCU_HCI_WRITE_BD_ADDR_REQ
13:04:00: <-H TCU_HCI_WRITE_BD_ADDR_RESP
13:04:00: H-> TCU_HCI_SET_MODE_REQ
13:04:00: <-H TCU_HCI_SET_MODE_RESP
13:04:01: --> TCU_MNG_INIT_REQ
13:04:01: <-- TCU_MNG_INIT_RESP
13:04:01: --> TCU_MNG_STANDARD_HCI_SET_REQ_Write_Class_of_Device
13:04:01: <-- TCU_MNG_STANDARD_HCI_SET_RESP_Write_Class_of_Device
13:04:01: --> TCU_SPP_SETUP_REQ
13:04:01: <-- TCU_SPP_SETUP_RESP
13:04:01: --> TCU_MNG_SET_SCAN_REQ
13:04:01: <-- TCU_MNG_SET_SCAN_RESP
```

Next, you will receive a Bluetooth pairing request on both devices. Please compare the passkeys and confirm the requests if the passkeys are the same. Otherwise decline the request.



Please confirm the number:

640951

```
13:05:51: !!! Unknown TCU_MNG_SSP_INFO_EVENT 0x3d received
13:05:51: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_RESULT_EVENT
13:05:51: <-- TCU_MNG_DISCOVER_REMOTE_DEVICE_COMPLETE_EVENT
13:09:00: --> TCU_MNG_DISCOVER_REMOTE_SERVICE_REQ
13:09:00: <-- TCU_ACCEPT
13:09:01: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:01: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:01: <-- TCU_MNG_DISCOVER_REMOTE_SERVICE_EVENT
13:09:01: --> TCU_SPP_CONNECT_REQ
13:09:01: <-- TCU_ACCEPT
13:09:02: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:02: <-- TCU_MNG_REMOTE_DEVICE_NAME_AUTO_NOTIFY_EVENT
13:09:02: <-- HCI_IO_Capability_Request_Event
13:09:02: --> TCU_MNG_SSP_SET_REQ_HCI_IO_Capability_Request_Reply
13:09:02: <-- TCU_MNG_SSP_SET_RESP_HCI_IO_Capability_Request_Reply
13:09:02: <-- HCI_IO_Capability_Response_Event
13:09:02: <-- HCI_IO_User_Confirmation_Request_Event
```

SSP pairing in process

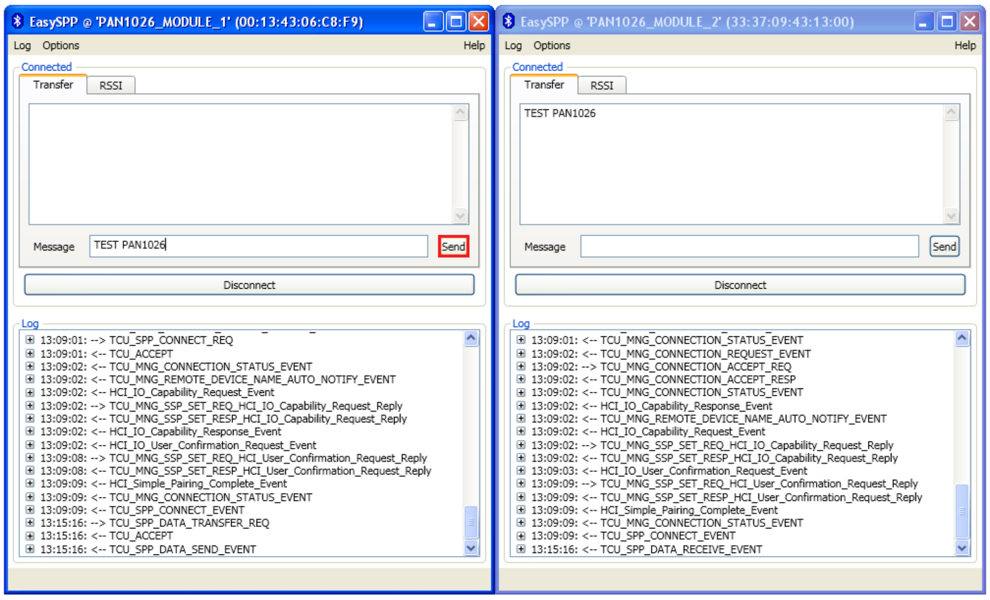
Please confirm the number:

640951

```
13:04:01: --> TCU_MNG_SET_SCAN_REQ
13:04:01: <-- TCU_MNG_SET_SCAN_RESP
13:09:01: <-- TCU_MNG_CONNECTION_REQUEST_EVENT
13:09:01: --> TCU_MNG_CONNECTION_ACCEPT_REQ
13:09:01: <-- TCU_MNG_CONNECTION_ACCEPT_RESP
13:09:01: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:01: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:02: <-- TCU_MNG_CONNECTION_REQUEST_EVENT
13:09:02: --> TCU_MNG_CONNECTION_ACCEPT_REQ
13:09:02: <-- TCU_MNG_CONNECTION_ACCEPT_RESP
13:09:02: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:02: <-- HCI_IO_Capability_Response_Event
13:09:02: <-- TCU_MNG_REMOTE_DEVICE_NAME_AUTO_NOTIFY_EVENT
13:09:02: <-- HCI_IO_Capability_Request_Event
13:09:02: --> TCU_MNG_SSP_SET_REQ_HCI_IO_Capability_Request_Reply
13:09:02: <-- TCU_MNG_SSP_SET_RESP_HCI_IO_Capability_Request_Reply
13:09:03: <-- HCI_IO_User_Confirmation_Request_Event
```

SSP pairing in process

Now transmit a sign or a simple sentence like it is shown in the following figure.



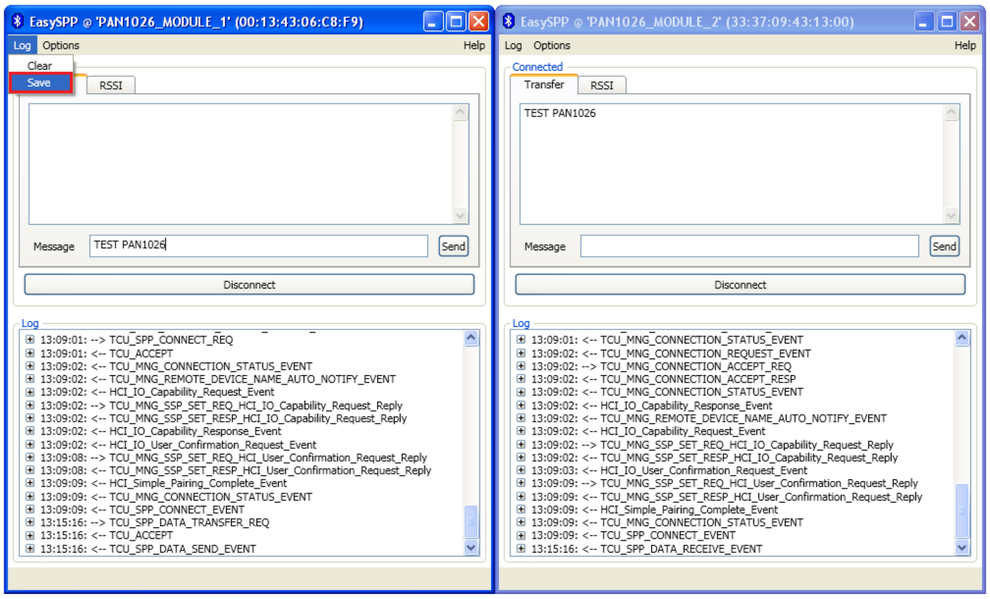
Message: TEST PAN1026

```
13:09:01: --> TCU_SPP_CONNECT_REQ
13:09:01: <-- TCU_ACCEPT
13:09:02: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:02: <-- TCU_MNG_REMOTE_DEVICE_NAME_AUTO_NOTIFY_EVENT
13:09:02: <-- HCI_IO_Capability_Request_Event
13:09:02: --> TCU_MNG_SSP_SET_REQ_HCI_IO_Capability_Request_Reply
13:09:02: <-- TCU_MNG_SSP_SET_RESP_HCI_IO_Capability_Request_Reply
13:09:02: <-- HCI_IO_Capability_Response_Event
13:09:02: <-- HCI_IO_User_Confirmation_Request_Event
13:09:08: --> TCU_MNG_SSP_SET_REQ_HCI_User_Confirmation_Request_Reply
13:09:08: <-- TCU_MNG_SSP_SET_RESP_HCI_User_Confirmation_Request_Reply
13:09:09: <-- HCI_Simple_Pairing_Complete_Event
13:09:09: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:09: <-- TCU_SPP_CONNECT_EVENT
13:15:16: --> TCU_SPP_DATA_TRANSFER_REQ
13:15:16: <-- TCU_ACCEPT
13:15:16: <-- TCU_SPP_DATA_SEND_EVENT
```

TEST PAN1026

```
13:09:01: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:02: <-- TCU_MNG_CONNECTION_REQUEST_EVENT
13:09:02: --> TCU_MNG_CONNECTION_ACCEPT_REQ
13:09:02: <-- TCU_MNG_CONNECTION_ACCEPT_RESP
13:09:02: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:02: <-- HCI_IO_Capability_Response_Event
13:09:02: <-- TCU_MNG_REMOTE_DEVICE_NAME_AUTO_NOTIFY_EVENT
13:09:02: <-- HCI_IO_Capability_Request_Event
13:09:02: --> TCU_MNG_SSP_SET_REQ_HCI_IO_Capability_Request_Reply
13:09:02: <-- TCU_MNG_SSP_SET_RESP_HCI_IO_Capability_Request_Reply
13:09:03: <-- HCI_IO_User_Confirmation_Request_Event
13:09:09: --> TCU_MNG_SSP_SET_REQ_HCI_User_Confirmation_Request_Reply
13:09:09: <-- TCU_MNG_SSP_SET_RESP_HCI_User_Confirmation_Request_Reply
13:09:09: <-- HCI_Simple_Pairing_Complete_Event
13:09:09: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:09: <-- TCU_SPP_CONNECT_EVENT
13:15:16: <-- TCU_SPP_DATA_RECEIVE_EVENT
```

Please save the log files of both devices as a base for the following sourcecode.


EasySPP @ 'PAN1026_MODULE_1' (00:13:43:06:C8:F9)
Log
Options
Help
Clear
Save
RSSI
Message
Send
Disconnect
Log
EasySPP @ 'PAN1026_MODULE_2' (33:37:09:43:13:00)
Connected
Transfer
RSSI
Message
Send
Disconnect
Log

TEST PAN1026

```
13:09:01: --> TCU_SPP_CONNECT_REQ
13:09:01: <-- TCU_ACCEPT
13:09:02: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:02: <-- TCU_MNG_REMOTE_DEVICE_NAME_AUTO_NOTIFY_EVENT
13:09:02: <-- HCI_IO_Capability_Request_Event
13:09:02: --> TCU_MNG_SSP_SET_REQ_HCI_IO_Capability_Request_Reply
13:09:02: <-- TCU_MNG_SSP_SET_RESP_HCI_IO_Capability_Request_Reply
13:09:02: <-- HCI_IO_Capability_Response_Event
13:09:02: <-- HCI_IO_User_Confirmation_Request_Event
13:09:08: --> TCU_MNG_SSP_SET_REQ_HCI_User_Confirmation_Request_Reply
13:09:08: <-- TCU_MNG_SSP_SET_RESP_HCI_User_Confirmation_Request_Reply
13:09:09: <-- HCI_Simple_Pairing_Complete_Event
13:09:09: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:09: <-- TCU_SPP_CONNECT_EVENT
13:15:16: --> TCU_SPP_DATA_TRANSFER_REQ
13:15:16: <-- TCU_ACCEPT
13:15:16: <-- TCU_SPP_DATA_SEND_EVENT
```

TEST PAN1026

```
13:09:01: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:02: <-- TCU_MNG_CONNECTION_REQUEST_EVENT
13:09:02: --> TCU_MNG_CONNECTION_ACCEPT_REQ
13:09:02: <-- TCU_MNG_CONNECTION_ACCEPT_RESP
13:09:02: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:02: <-- HCI_IO_Capability_Response_Event
13:09:02: <-- TCU_MNG_REMOTE_DEVICE_NAME_AUTO_NOTIFY_EVENT
13:09:02: <-- HCI_IO_Capability_Request_Event
13:09:02: --> TCU_MNG_SSP_SET_REQ_HCI_IO_Capability_Request_Reply
13:09:02: <-- TCU_MNG_SSP_SET_RESP_HCI_IO_Capability_Request_Reply
13:09:03: <-- HCI_IO_User_Confirmation_Request_Event
13:09:09: --> TCU_MNG_SSP_SET_REQ_HCI_User_Confirmation_Request_Reply
13:09:09: <-- TCU_MNG_SSP_SET_RESP_HCI_User_Confirmation_Request_Reply
13:09:09: <-- HCI_Simple_Pairing_Complete_Event
13:09:09: <-- TCU_MNG_CONNECTION_STATUS_EVENT
13:09:09: <-- TCU_SPP_CONNECT_EVENT
13:15:16: <-- TCU_SPP_DATA_RECEIVE_EVENT
```

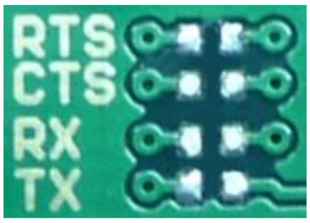
Note that the BD-Address and the name of the modules are variable and that they can be changed individually in the sourcecode.

3.2 Preparing the Hardware

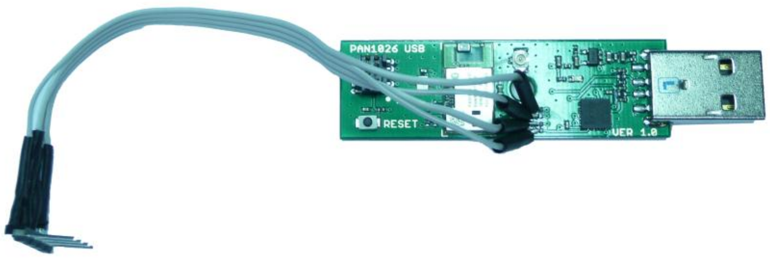
To have the best opportunities to verify the software step by step, it is recommended to prepare the test hardware first as follows.

The Resistors R5, R6, R7 and R8 (please refer to the USB Evaluation Kit wiring diagram in the PAN1026_ApplicationNote.pdf) on the PAN1026 USB stick PCB are 0Ω resistors for the wired connection between the FT232RQ chip and the TC35661.

Please unsolder these resistors to establish a wired connection between the PAN1026 and the microcontroller of your choice.



It is now possible to solder wires directly to the free pads that are right next to the Bluetooth module. The hardware flow control (RTS, CTS) is optional and can be left open if your controller does not support hardware flow control.



Another option is to scrape the solder resist from the circle pads, drill holes, put wires through them and solder the wires on top of the PCB.

Link the microcontroller UART interface with the PAN1026 UART pins on the prepared USB stick, which has to be plugged into a USB slot for power supply.

Please mind an appropriate system clock frequency for your microcontroller system to avoid malfunction. Your microcontroller datasheet provides appropriate oscillator frequencies to minimize the error of the UART.

3.3 PAN1026 Documentation

If you take a closer look into the previously saved log files, you will realize that all data have a timestamp, a sending direction and a name, which indicates whether the command is a request or a response. In the last line of all commands you will find the transmitted or received data sequence.

It is highly recommended to go through the log files step by step with the available documentation to get familiar with the commands and events. Therefore, the most important documents are the following:

1. PAN1026_Documentation Guide
2. TC35661APL_ROM501_Extension_HCI_E_18thSeptember2013
3. PAN1026_TC35661APL_ROM501_MNG_E_26thJuly2013_1
4. PAN1026_TC35661APL_ROM501_SPP_E_24thJune2013
5. PAN1026_TC35661APL_ROM501_MNG_SequenceChart
6. PAN1026_TC35661APL_ROM501_SPP_SequenceChart
7. TC35661APL_configuration_settings_1_0

Furthermore, it is recommended to comment the log files as good as possible to understand the structure and the sequence of the commands and events. Please find an example attached and note that currently unnecessary commands like patches and scanning procedures have been removed to ensure clarity for this simple example.

The remaining documents should be read beforehand to get an overview of the functionality of the PAN1026.

3.4 Writing the Source code

The idea for a quick start microcontroller program is to copy the commands/requests from the log files, store them in a header file (e.g. commands.h) and send these commands to the TC35661 after the previous command has been accepted successfully. Nevertheless, to obtain a sourcecode which is loose to the used module, it is necessary to keep a few commands variable as for instance the command TCU_HCI_WRITE_BD_ADDR_REQ.

The attached source code initializes a PAN1026, requests the pairing with another predefined PAN1026 ETU USB stick and transmits a test sentence.

Please follow the instructions below to program your source code step by step and refer to the attached source code example, the log files and the documentation to succeed.

3.4.1 UART Settings

The first step is to initialize the UART interface. The given example initializes the two following communications:

USART0:

Purpose:	Debugging (Outputs the exchanged data on USART1)
Direction:	Unidirectional (Controller, PC Terminal)
Baud Rate:	115200bps
Data Format:	8N1
Flow Control:	None

USART1:

Purpose:	Data exchange
Direction:	Bidirectional (Controller, PAN1026)
Baud Rate:	9600bps (The TC35661 default is 115200bps, but is programmable to other baud rates)
Data Format:	8N1
Flow Control:	None

After the UART initialization, it is necessary to compose functions which are able to transmit and receive strings with various lengths. It is possible to use interrupts for the communication as well.

3.4.2 First Communication

As a first step in the communication between the microcontroller and the PAN1026, transmit the software reset *TCU_HCI_RESET_REQ* with the data sequence *0x01 0x03 0x0c 0x00* and check whether the microcontroller receives the data sequence *0x04 0x0e 0x04 0x04 0x03 0x0c 0x00* which is called *TCU_HCI_RESET_RESP*. If this does not work the following points might be helpful:

- Check the data sequence transmission with a terminal on your computer
  (If possible use a second UART interface and a terminal to monitor the communication)
- Check the soldering points for appropriate contact
- Check the power supply of the USB stick
- Check the UART voltage, which has to operates at 3.3V
  (If your controller operates at another voltage you will need a levelshifter)

When the first communication does work exactly as expected, please go on with the next section.

3.4.3 PAN1026 Initialization

The next step is to initialize the PAN1026. To obtain a communication which is loose to the used PAN1026 and its responses, it is recommended to prepare an event handler which recognizes the responses and decides how to proceed. The attached source code provides an example of how to structure the different event handlers.

The attached source code uses the following ten commands to initialize the PAN1026:

- TCU_HCI_GET_FIRMWARE_VERSION_REQ
- TCU_HCI_M2_BTL_SET_I2C_ENABLE_REQ
- TCU_HCI_M2_BTL_EEPROM_WRITE_ENABLE_REQ
- TCU_HCI_M2_GENERAL_READ_EEPROM_REQ
- TCU_HCI_WRITE_BD_ADDR_REQ
- TCU_HCI_SET_MODE_REQ
- TCU_MNG_INIT_REQ
- TCU_MNG_STANDARD_HCI_SET_REQ_Write_Class_of_Device
- TCU_SPP_SETUP_REQ
- TCU_MNG_SET_SCAN_REQ

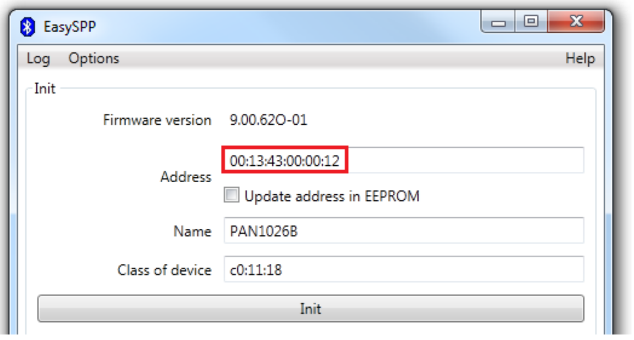
Please note, that not every single command is mandatory for the initialization.

The initialization is complete after the transmission of the *TCU_SPP_SETUP_REQ*. To discover the device with the second PAN1026 USB stick or another Bluetooth device it is necessary to add the *TCU_MNG_SET_SCAN_REQ*. Now you can check whether or not the device is discoverable

If the last step is done and the device is discoverable with any Bluetooth device, you are ready to establish a Bluetooth connection and transmit data.

3.4.4 Pairing and Data Exchange

To establish a connection to a predefined device it is necessary to include its BD-Address into the commands which are used for pairing and data transmission. The attached source code does that automatically, so that the BD-Address of the remote device does only have to be declared once (e.g. REMOTE_BDADDRESS in commands.h). You will find the remote BD-Address of the second PAN1026 USB stick in the address field of EasySPP as shown below.



Firmware version 9.00.620-01
Address 00:13:43:00:00:12
Update address in EEPROM
Name PAN1026B
Class of device c0:11:18

The source code uses the following three commands to establish a connection:

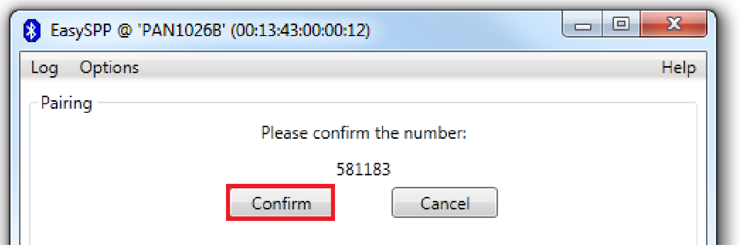
- TCU_SPP_CONNECT_REQ
- TCU_MNG_SSP_SET_REQ_HCI_IO_Capability_Request_Reply
- TCU_MNG_SSP_SET_REQ_HCI_User_Confirmation_Request_Reply

The command TCU_SPP_DATA_TRANSFER_REQ enables to transmit data. Please refer to the documentation to understand the structure of these commands and their events.

3.5 Functional Verification

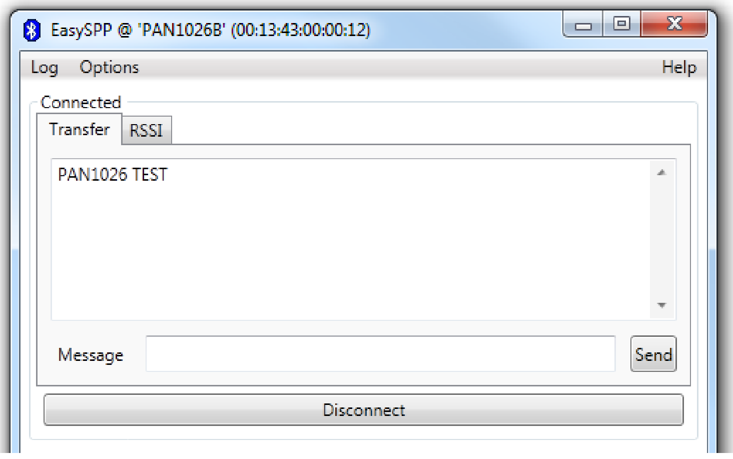
To check the last step you have to make sure that the second USB stick is initialized successfully and accepts connections.

Start your microcontroller and take a look at the EasySPP user interface. After a short time you will receive a connection request like it is shown in the following figure. Please confirm this request to complete the pairing process. If you do not confirm the request, it will time out and has to be repeated.



Please confirm the number:
581183

After you have confirmed the request, the microcontroller will send the test sentence which should appear in the EasySPP message box as shown below.



PAN1026 TEST

After all the previous hurdles have been cleared, the source code can be improved to a more interactive code, which suits your applications.

4 References

No.	Title	Source
1	TOSHIBA TC35661APL EasySPP QuickStartGuide	TC35661APL_EasySPP_quick_start_guide_v1_0_0_15.pdf
2	PAN1026 Overview Specification	TC35661SBG-501_E_rev100_Oct_2013_Overview_Specification.pdf
3	PAN1026 Product Specification	PAN1026_Datasheet.pdf
4	PAN1026 Design Guide	PAN1026_ApplicationNote.pdf
5	PAN1026 Documentation Guide	PAN1026_DocumentationGuide.pdf
6	Toshiba Chiron Configuration Settings	TC35661APL_configuration_settings_1_0.pdf
7	PAN1026 HCI Extension command set	TC35661APL_ROM501_Extension_HCI_E_18thSeptember2013.pdf
8	PAN 1026 MNG command set	PAN1026_TC35661APL_ROM501_MNG_E_26thJuly2013_1.pdf
9	PAN 1026 SPP command set	PAN1026_TC35661APL_ROM501_SPP_E_24thJune2013.pdf
10	PAN1026 MNG Sequence Chart	PAN1026_TC35661APL_ROM501_MNG_SequenceChart.pdf
11	PAN1026 SPP Sequence Chart	PAN1026_TC35661APL_ROM501_SPP_SequenceChart.pdf
12	Bluetooth Specification	Core_v4.1.pdf