APPLICATION NOTE

Atmel AVR2068: RF4CE-HID QTouch Analyzer Target for AVR477

32-bit Atmel Microcontrollers

Features

- Integration of ZigBee® RF4CE and USB-HID Stacks on 32-bit Atmel® AVR® MCU
  - Atmel ATUC3A3256S
- QTouch® Analyzer Support
  - QDebug Protocol over USB-HID Interface
- Target Application for Atmel AVR477 Remote

Description

This application note demonstrates a ZigBee RF4CE Target that also incorporates a USB stack.

This application works in conjunction with the AVR477 Remote Control application [1]. In the demo scenario, the target application receives the QDebug data from the Remote over the air and then forwards it to QTouch Analyzer (on PC) using USB-HID interface.

Table of Contents

1. System Overview 3
2. Target Hardware 4
   - 2.2 Microcontroller 4
     - 2.2.1 RZ600 USB board 4
   - 2.3 RF transceiver 4
     - 2.3.1 RZ600 radio board 4
   - 2.4 Programming and debugging 4
3. Target Formware 5
   - 3.1 Atmel RF4Control – ZigBee RF4CE stack 5
     - 3.1.1 Target application 5
     - 3.1.2 Remote application 5
       - 3.1.2.1 System sequence 6
   - 3.2 Atmel USB device stack 7
     - 3.2.1 Stack features 7
   - 3.3 QDebug Protocol 7
     - 3.3.1 Message sequence chart 8
     - 3.3.2 USB descriptor configuration 8
     - 3.3.3 Protocol packet format 9
       - 3.3.3.1 Payload format 9
     - 3.3.4 QDebug commands 9
     - 3.3.5 QDebug commands description 10
       - 3.3.5.1 QT_CMD_SET_SUBS, set data subscription 10
       - 3.3.5.2 QT_SIGN_ON, Sign On 11
       - 3.3.5.3 QT_GLOBAL_CONFIG, Global Config 12
       - 3.3.5.4 QT_SENSOR_CONFIG, Sensor Config 12
       - 3.3.5.5 QT_SIGNALS, Signals 13
       - 3.3.5.6 QT_REFERENCES, References 14
       - 3.3.5.7 QT_DELTAS, Sensor Deltas 14
       - 3.3.5.8 QT_STATES, Sensor States 14
4. Quick Start Guide 16
   - 4.1 Hardware setup 16
     - 4.1.1 Tools required 16
     - 4.1.2 Remote 16
     - 4.1.3 Target 16
     - 4.1.4 QTouch Analyzer 16
   - 4.2 Quick start 17
   - 4.3 Display / demo scenarios 18
     - 4.3.1 RF4CE pairing 18
     - 4.3.2 Touch 18
     - 4.3.3 Proximity 18
     - 4.3.4 Accelerometer 18
     - 4.3.5 Sleep 19
     - 4.3.6 Wakeup 19
     - 4.3.7 Fault indication 19
     - 4.3.8 Start / Stop 19
     - 4.3.9 Atmel QTouch Analyzer Demo screen 20
5. References 21
6. Revision History 22

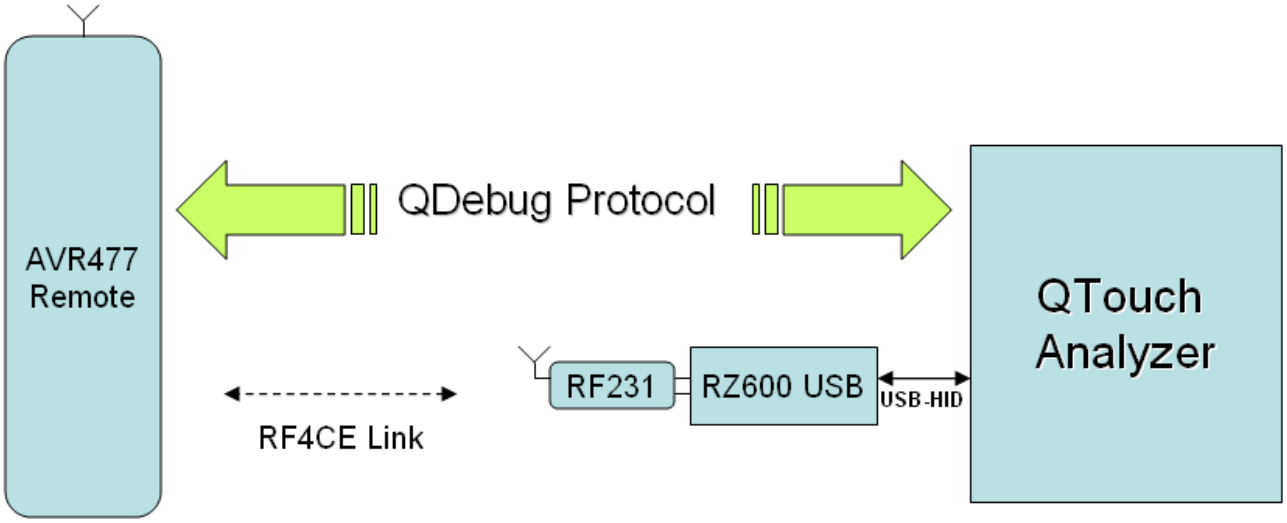
1. System Overview

This application note demonstrates the usage of the RZ600 -RF231module as a Target capable of receiving QDebug data over the air from the Atmel AVR477 remote control.The received QDebug data is then displayed on the Atmel QTouch Analyzer. The RZ600 module communicates with QTouch Analyzer using QDebug Protocol.The protocol description can be found in Section 3.3.

QDebug Protocol support allows:

- QDebug Subscription data transfer
- Start and Stop control mechanism of QDebug data from QTouch Remote Control
- Display of QDebug data to analyze the performance of QTouch Remote Control

Figure 1-1. System overview.



2. Target Hardware

The Atmel RZ600 Module serves as the target using the RZ600 USB board to connect to the PC and the RZ600-RF231 radio board to receive data over air.

The RZ600 USB board is connected to the RZ600 radio board using a 10-pin connector as shown in Table 2-1.

Please refer to the RZ600 Hardware Manual [6] for a complete hardware description.

Table 2-1. Radio frequency 10-pin header.

Pin	Name	Pin	Name
1	Reset	2	Misc
3	Interrupt	4	Sleep Transmit
5	Chip Select	6	MOSI
7	MISO	8	SCK
9	GND	10	V_{CC} (1.8V to 3.6V)

2.2 Microcontroller

The RZ600 USB board has an on-board 32-bit Atmel AVR MCU (ATUC3A3256S).

The microcontroller achieves exceptionally high data throughput by combining the multi-layered 32-bit AVR databus, 128kB on-chip SRAM, multi-channel peripheral, DMA controller, high-speed USB capability, 256kB internal FLASH memory and an AES crypto module.

This makes it an ideal platform for IEEE® 802.15.4 based wireless applications.

2.2.1 RZ600 USB board

The RZ600 USB board has two LEDs for indication purposes, UART header, RF header and JTAG header for programming and debugging.

The RZ600 USB board has two clock sources (12MHz and 32.768kHz crystals), the reference application uses the 12MHz crystal as input to PLL for obtaining reference clock to run the USB and RF4CE stacks.

2.3 RF transceiver

The Atmel AT86RF231 is a low-power 2.4GHz transceiver designed for IEEE 802.15.4, ZigBee, RF4CE applications.It has an extended feature set supporting external Front-end control, antenna diversity and high-data rate support.

2.3.1 RZ600 radio board

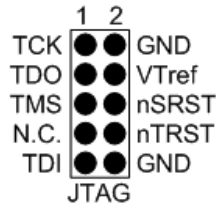
The AT86RF231 radio board has an RF header that can be used to connect the radio board to the RZ600 USB board as shown in Table 2-1.

A 16MHz crystal is used to generate the High-frequency RF Carrier used by the AT86RF231.

2.4 Programming and debugging

The RZ600 USB board has a 10-pin JTAG header that can be used to program and debug the MCU using standard Atmel tools.

Figure 2-2. JTAG header.



3. Target Formware

3.1 Atmel RF4Control – ZigBee RF4CE stack

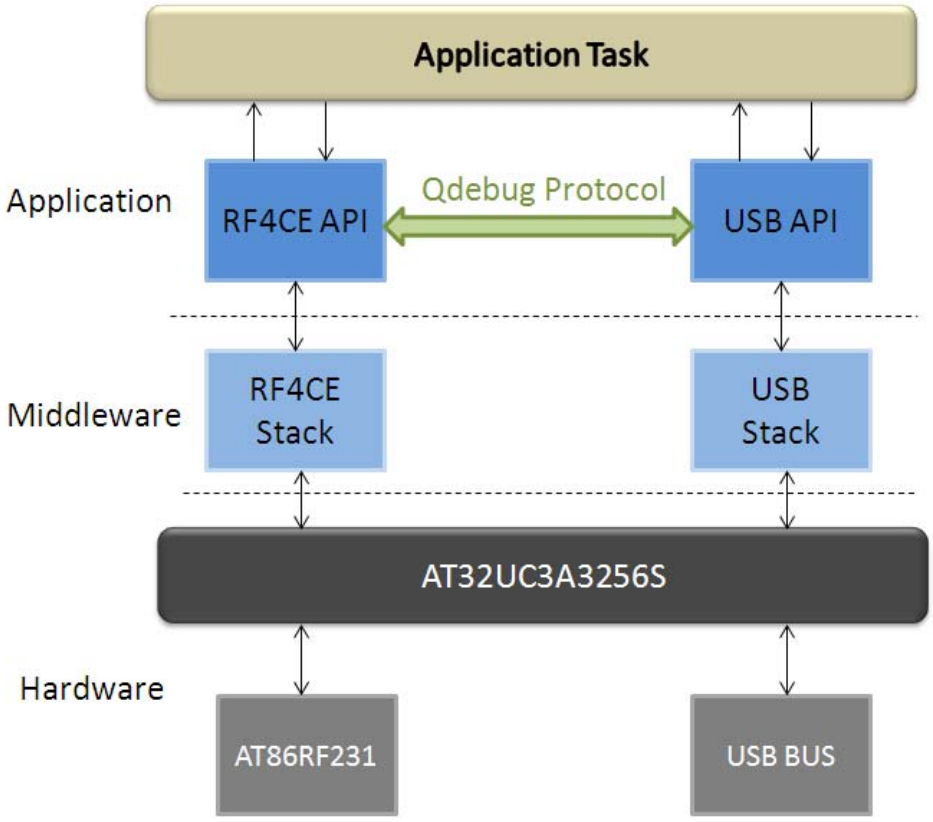
3.1.1 Target application

The reference application implementation for the RZ600 integrates the USB HID Driver and QDebug Protocol with the RF4CE library to perform the following actions:

- Receive QDebug data in RF4CE payload from AVR477 remote over air and forward it to QTouch Analyzer via USB-HID in QDebug Protocol format
- Receive commands from QTouch Analyzer in QDebug format and send it over air to AVR477 remote in RF4CE packet format

The application architecture for the Target reference application is shown in Figure 3-1.

Figure 3-1. Application architecture.



3.1.2 Remote application

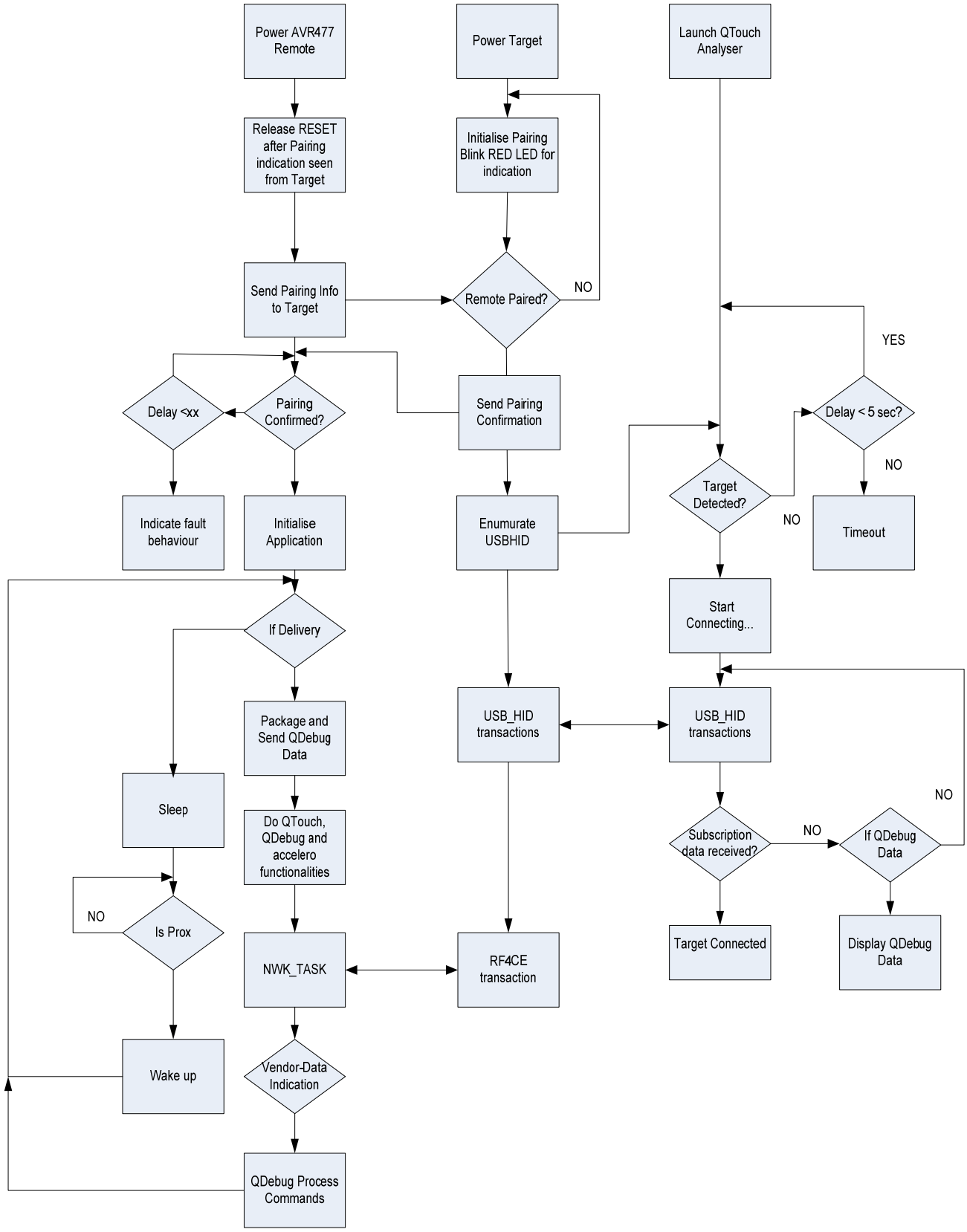
AVR477 implements the Atmel RF4Control - Touch remote control application on the Atmel ATmega128RFA1 SoC, integrating the RF4CE library and Capacitive Touch (32-Channel QMatrix library) implementation.

The application collects the touch (QDebug) data from the Atmel QTouch library, packages it into the RF4CE frame format and sends it over the air to the RZ600 Target.

The entire system communication sequence is shown in Section 3.1.2.1.

Please refer to the Atmel AVR477: RF4Control – Touch Remote Control Application Note [1] for implementation details.

3.1.2.1 System sequence



3.2 Atmel USB device stack

The Target application uses the Device USB stack from Atmel AVR Software Framework (ASF) to implement a generic USB HID device operating with High Speed data rate.

3.2.1 Stack features

- USB 2.0 compliant and USB Chapter 9 certified
- Low Speed (1.5Mbit/s), Full Speed (12Mbit/s), High Speed (480Mbit/s) data rates
- Completely interrupt driven
- Smaller stack size
- USB DMA support increases speed performance
- Supports most USB classes and ready to use (HID, CDC, MSC, PHDC, AUDIO)

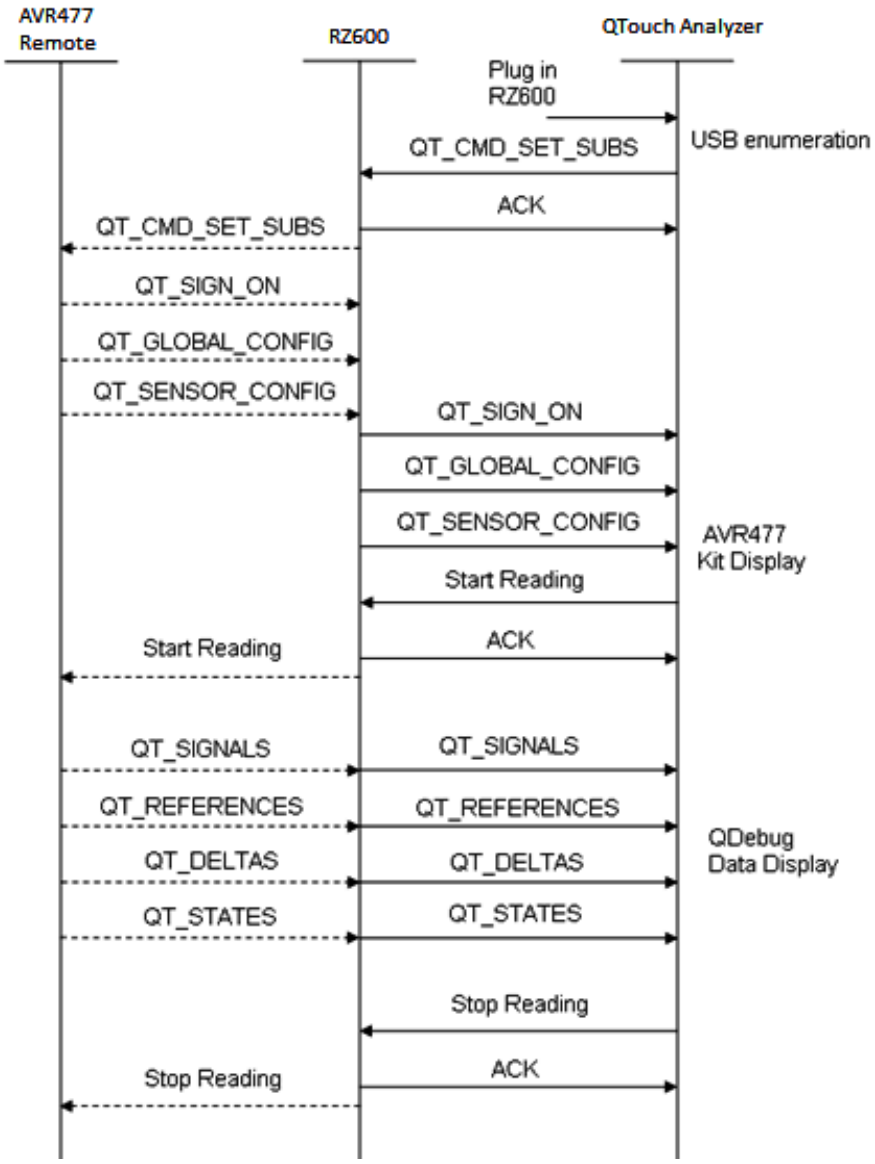
3.3 QDebug Protocol

The RZ600 Module uses the QDebug Protocol to communicate with the QTouch Analyzer.

This section requires a basic knowledge of the parameters involving in handling of Touch data from application using Atmel QTouch Library [7].

Section 3.3.1 shows the communication sequence in the demo scenario using the AVR477 Remote.

3.3.1 Message sequence chart



3.3.2 USB descriptor configuration

QTouch Analyzer expects the following configuration from the USB Device in order to detect the connected kit as a valid QDebug Interface.

The USB device descriptor values for certain parameters should be set as shown in Table 3-1.

Table 3-1. USB device descriptor parameters.

Descriptor variable	Value
idVendor	USB_VID_ATMEL(0x03EB)
idProduct	HID_QTOUCH_DEBUG_PID(0x211F)
bMaxPacketSize0	USB_DEVICE_EP_CTRL_SIZE(0x40)

3.3.3 Protocol packet format

The QDebug protocol uses the packet format shown in Table 3-2 for communication with QTouch Analyzer.

Table 3-2. QDebug protocol packet format.

Packet field	Variable	Size (in bytes)	Description
Header	protocol_id	1	QDEBUG_PROTOCOL_ID(0x01)
	spare_byte1	1	Spare
	spare_byte2	1	Used for indicating type of packet (1)
	packetCount	1	Indicates total number of QDebug data packets for a particular QDebug Command Indicates the packet number (2)
	packetNumber	1	
Payload	payload	60	Includes Checksum (1 byte)

Notes: 1. QTouch Analyzer supports two types of packets, namely:
- QDebug Data packet with spare_byte2 set to 0x00
- QDebug Ack packet with spare_byte2 set to 0x40

2. If the QDebug data size exceeds payload size, it is sent as several packets.This is limited by the size of the USB endpoint buffer (USB_DEVICE_EP_CTRL_SIZE - 64 bytes).

3.3.3.1 Payload format

The QDebug Protocol specifies the payload format shown in Table 3-3.

Table 3-3. QDebug payload format.

Structure	Variable	Description
Byte[0]	Message Start Byte	QT_MESSAGE_START (0x01)
Byte[1]	Packet length, High byte	Length of the package. Maximum length is 270. This includes packet length bytes and checksum and excludes QT_MESSAGE_START
Byte[2]	Packet length, Low byte	
Byte[3]	Sequence number	High nibble is incremented for every touch measurement Low nibble is incremented for each package
Byte[4]	Payload byte 1	Payload, N bytes, N is the size of the payload frame
….	….	
Byte[N+4]	Payload byte N	
Byte[N+5]	Checksum	This value is the checksum of the complete package (excluding message start byte)

3.3.4 QDebug commands

The QDebug Payload may contain information relevant to a particular command ID depending on the direction of the packet (From QTouch Analyzer to Target or from Target to QTouch Analyzer).

The set of command IDs for QTouch Analyzer commands are shown in Table 3-4 and Target commands in Table 3-5.

Table 3-4. QTouch Analyzer commands, PC → Target → Remote.

Command ID	Description	Send condition
QT_CMD_SET_SUBS	Set data subscription	Set up subscription of Touch data on AVR477 Remote. This is a bit field described in Section 3.3.5.1
QT_CMD_SET_GLOBAL_CONFIG	Set Touch global structure	Complete configuration structure of the Touch libraries (writable values only)
QT_CMD_SET_SENSOR_CONFIG	Set Touch channel structure	Channel specific configuration structure of the Touch libraries (writable values only) for one single channel

Table 3-5. Target commands, Remote → Target → PC.

Command ID	Description	Description
QT_SIGN_ON	Sign on	Initialization configuration structure(static)
QT_GLOBAL_CONFIG	Global Config	Global configuration structure(changeable)
QT_SENSOR_CONFIG	Channel Config per Sensor	Sensor config structure
QT_SIGNALS	Signals	Continuous (at each timer tick)
QT_REFERENCES	References	When changed, or requested by PC-host
QT_DELTAS	Sensor Delta	Continuous if enabled by host
QT_STATES	Sensor States	Sensors ON/OFF + rotor/slider position, sent when changed, or requested by PC-host

3.3.5 QDebug commands description

3.3.5.1 QT_CMD_SET_SUBS, set data subscription

Table 3-6. Set subscription packet structure.

Command ID	Description	Description
QT_CMD_SET_SUBS	1 byte	0x11
Once	2 bytes	Bitmask
Changed	2 bytes	Bitmask
Always	2 bytes	Bitmask

QTouch Analyzer can subscribe to the different data by setting the corresponding bit in the bitmask to 1.

Once: When a bit is set in the Once field, the corresponding QDebug parameters will be sent from AVR477 Remote to QTouch Analyzer one time only.

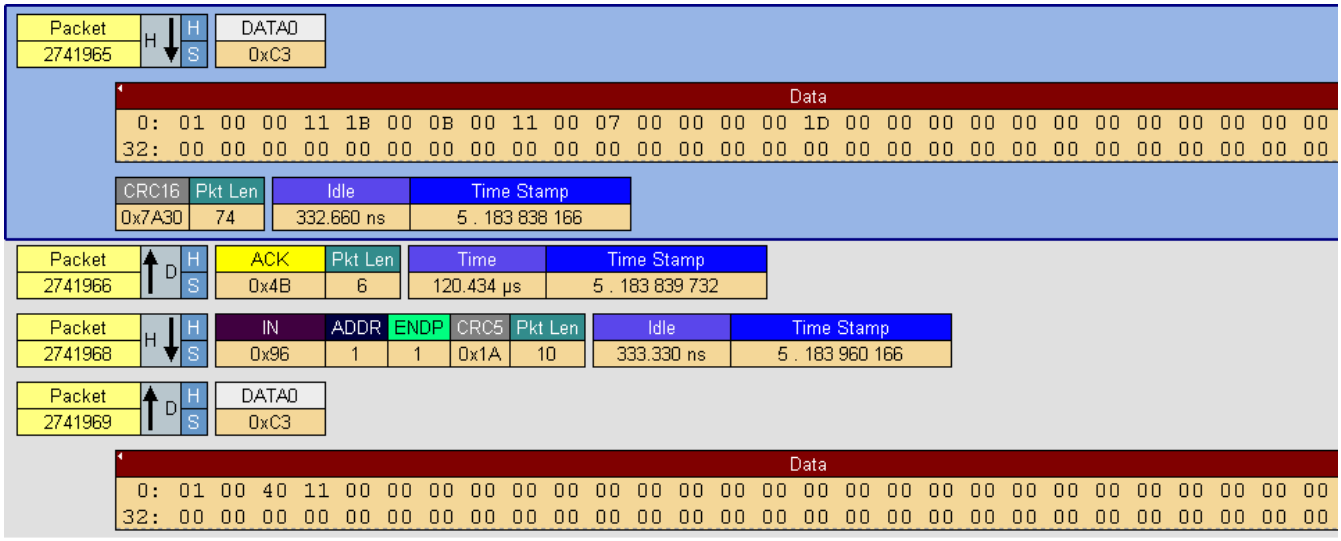
Changed: When a bit is set in the Changed field, the corresponding QDebug data will be sent from AVR477 Remote to QTouch Analyzer every time the parameters change.

Always: When a bit is set in the Always field, the corresponding QDebug data will be sent from AVR477 Remote to QTouch Analyzer for every touch measurement.

Bitmask:
Bit 0: QT_SIGN_ON
Bit 1: QT_GLOBAL_CONFIG
Bit 2: QT_SENSOR_CONFIG
Bit 3: QT_SIGNALS
Bit 4: QT_REFERENCES
Bit 5: QT_DELTAS
Bit 6: QT_STATES

Example: Figure 3-2 shows an example transfer of subscription set request from QTouch Analyzer to Target and the Ack packet sent by Target to QTouch Analyzer.

Figure 3-2. Example subscription and ack packets.



Note: The other QTouch Analyzer commands are not used in the reference application and their description is out of the scope of this document.

3.3.5.2 QT_SIGN_ON, Sign On

Table 3-7. Sign On Packet Structure.

Field	Size	Value	Description
QT_SIGN_ON	1 byte	0x21	Command ID
Project-id	2 bytes	0x01	AVR477 Board ID
Interface	1 byte	0x06	USB HID (1)
Protocol type	1 byte	0x01	1: Current Protocol 2: Future use
Protocol version	1 byte	0x02	Protocol Version
Library type	1 byte	0x01	0 = QTouch 1= QMatrix
Library version	2 bytes		Touch library version
Library variant	2 bytes	Not used	Port combinations + timing. (If 0,not available)
Num channels	1 byte	32 channels	N is the number of sensing channels in AVR477 Remote

Note: 1. QTouch Analyzer does not display Interface type for HID devices in Kit information.

3.3.5.3 QT_GLOBAL_CONFIG, Global Config

Table 3-8. Global Config Packet Structure.

Field	Size	Value	Description
QT_GLOBAL_CONFIG	1 byte	0x22	Command ID
Global config struct	7 bytes		(Sensor recalibration threshold) recal_threshold_t qt_recal_threshold (Sensor detect integration (DI) limit) uint8_t qt_di (Sensor drift hold time) uint8_t qt_drift_hold_time (Sensor maximum on duration) uint8_t qt_max_on_duration (Sensor negative drift rate) uint8_t qt_neg_drift_rate (Sensor positive drift rate) uint8_t qt_pos_drift_rate (Positive recalibration delay) uint8_t qt_pos_recal_delay
qt_measurement_period_msec	2 bytes	15	Time interval in milliseconds between each qt_measure_sensors() call in the user application (If 0,not supported)
TICKS_PER_MS	2 bytes	15	The number of timer ticks that makes one millisecond (If 0,not supported)
Timing settings	1 byte	0x00	NA

3.3.5.4 QT_SENSOR_CONFIG, Sensor Config

Table 3-9. Sensor Config Packet Structure.

Field	Size	Value	Description
QT_SENSOR_CONFIG	1 byte		Command ID
K or KRS	1 byte		0= Keys only 1= Keys, Rotors and Sliders
Per sensor configuration	K: N*3 KRS: N*4 (1)		(Sensor detection threshold) uint8_t threshold uint8_t type_aks_pos_hyst (2) uint8_t from_channel If KRS: uint8_t to_channel;

Notes: 1. N is the number of channels.

2. aks_pos_hyst holds the sensor type, AKS® group, positive recal flag, and hysteresis value.

   Bitmask

   Bits 7, 6: sensor type:

   00: reserved

   01: key

   10: rotor

   11: slider

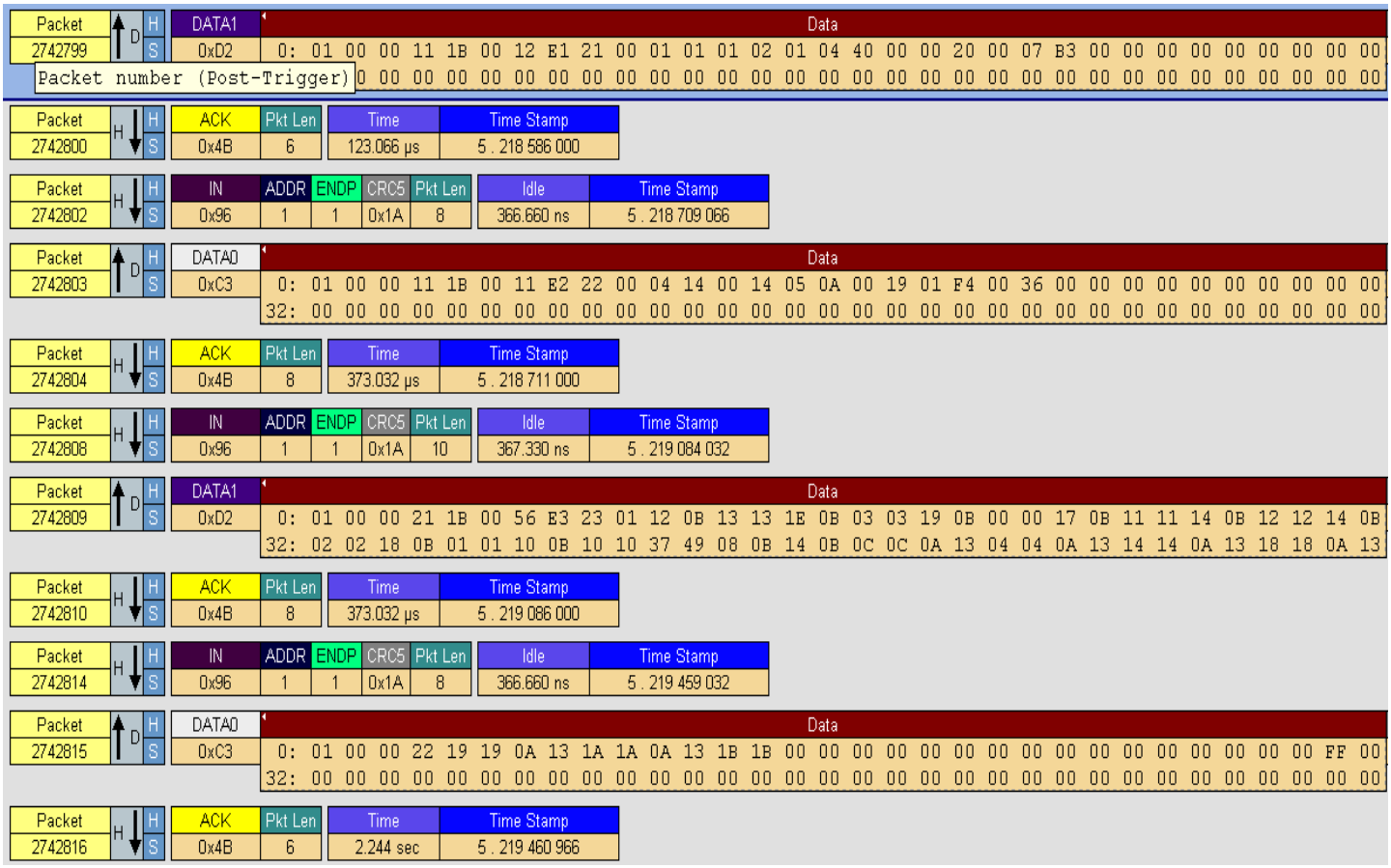
   Bits 5, 4, 3: AKS group

   Bit 2 : positive recal flag

   Bits 1, 0: hysteresis

Example: Figure 3-3 shows an example transfer of subscription data sent from AVR477 remote over air to Target and transferred over USB HID to QTouch Analyzer.

Figure 3-3. Example Subscription data packets.



3.3.5.5 QT_SIGNALS, Signals

Table 3-10. Signals Packet structure.

Field	Size	Value	Description
QT_SIGNALS	1 byte	0x24	Command ID
Signals Channel 0	2 bytes	0xffff	Touch Data(Signals) for channel 0
Signals Channel 1	2 bytes	0xffff	
Signals Channel 2	2 bytes	0xffff	
….			
Signals Channel N-1	2 bytes	0xffff	Touch Data(Signals) for channel N (1)

Note: 1. N is the number of channels.

3.3.5.6 QT_REFERENCES, References

Table 3-11. Signals Packet, Structure.

Field	Size	Value	Description
QT_REFERENCES	1 byte	0x25	Command ID
Reference Channel 0	2 bytes	0xffff	Touch Data(References) for channel 0
Reference Channel 1	2 bytes	0xffff	
Reference Channel 2	2 bytes	0xffff	
….			
Reference Channel N-1	2 bytes	0xffff	Touch Data(References) for channel N (1)

Note: 1. N is the number of channels.

3.3.5.7 QT_DELTAS, Sensor Deltas

Table 3-12. Deltas Packet Structure.

Field	Size	Value	Description
QT_DELTAS	1 byte	0x26	Command ID
Delta, Sensor 0	2 bytes	0xffff	Delta value for Sensor 0(Signed)
Delta, Sensor 1	2 bytes	0xffff	
Delta, Sensor 2	2 bytes	0xffff	
….			
Delta, Sensor M-1	2 bytes	0xffff	Delta value for Sensor M (1) (Signed)

Note: 1. M is the number of channels.

3.3.5.8 QT_STATES, Sensor States

Table 3-13. States Packet Structure.

Field	Size	Value	Description
QT_STATES	Byte	0x27	Command ID
Max number of channels	1 byte	N (1)	Max number of channels: QT_NUM_CHANNELS fixed as per chosen touch library
Max number of R/S	1 byte	M (2)	Max number of rotors and sliders: QT_MAX_NUM_ROTORS_SLIDERS fixed as per chosen touch library
Sensor States	X bytes	QT_NUM_SENSOR_STATE_BYTES	X = N/8 (rounded up value)
Rotor/slider positions	Y bytes	8	QT_MAX_NUM_ROTORS_SLIDERS

Notes: 1. N is the number of channels.
2. M is the number of sensors.

Example: Figure 3-4 through Figure 3-7 show an example transfer of QDebug data (QT_SIGNALS, QT_REFERENCES, QT_DELTAS, QT_STATES) sent from AVR477 remote over air to Target and transferred over USB HID to QTouch Analyzer.

Figure 3-4. Example QDebug data packet – QT_SIGNALS.

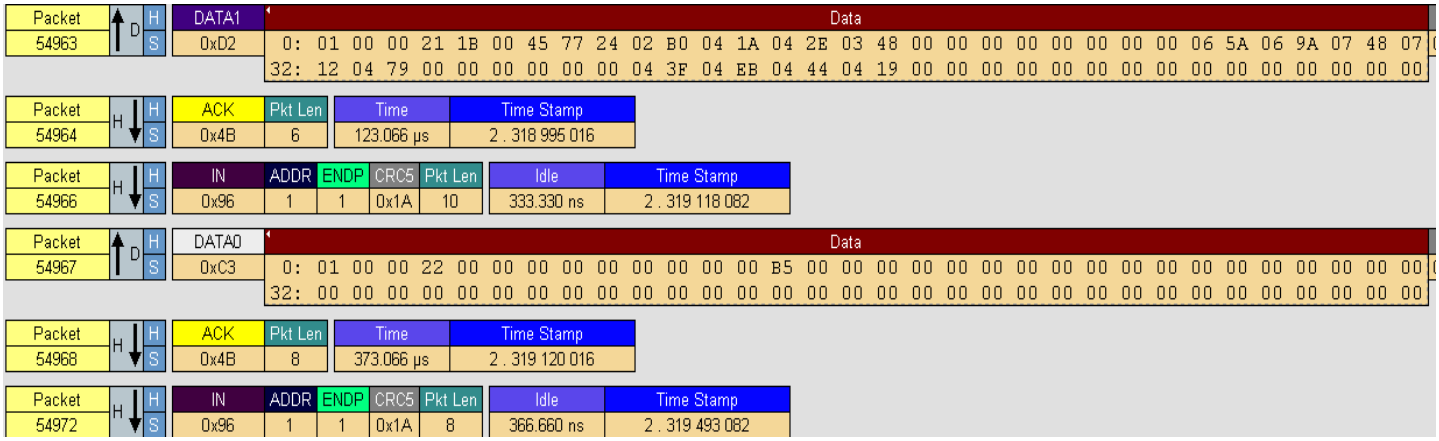


Figure 3-5. Example QDebug data packets – QT_REFERENCES.

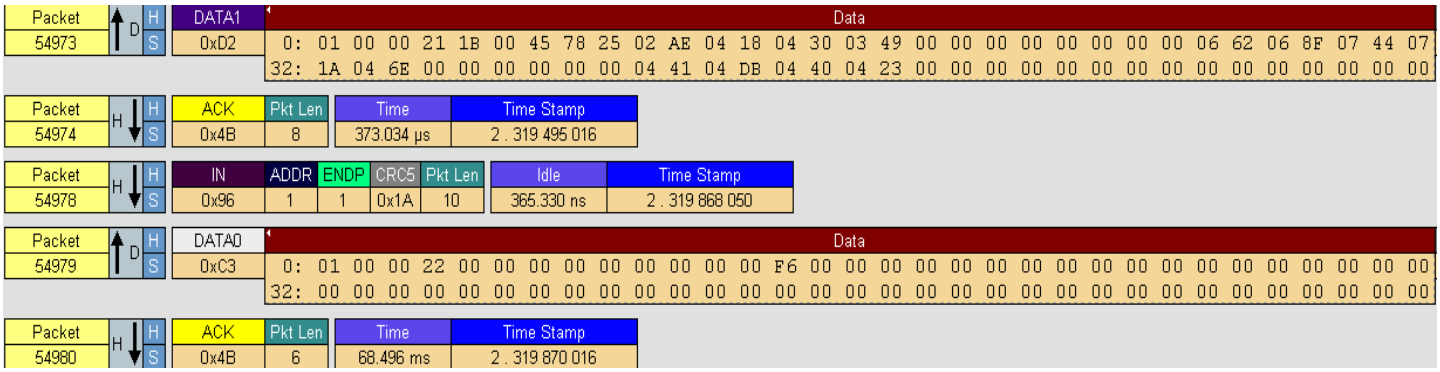


Figure 3-6. Example QDebug data packets – QT_DELTAS.

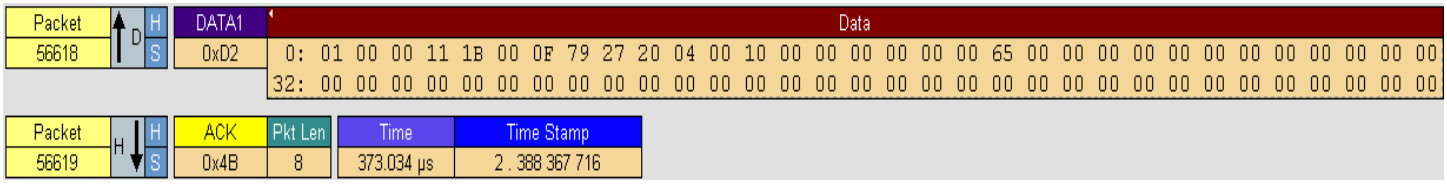
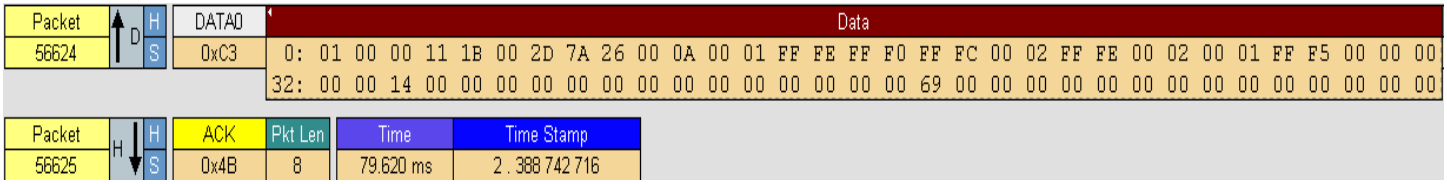


Figure 3-7. Example QDebug data packets – QT_STATES.



4. Quick Start Guide

4.1 Hardware setup

4.1.1 Tools required

- Atmel Studio 6
- Atmel QTouch Analyzer
- Atmel AVR JTAGICE mkII/JTAGICE3
- Atmel AVR477 Remote
- ATAVRRZ600(with AT86RF231 radio module)

4.1.2 Remote

- Hardware: AVR477 Remote
- Device: ATmega128RFA1
- Target Supply: 2 × AAA batteries
- Target Clock: Internal RC
- Hex: AVR477_Remote.hex

4.1.3 Target

- Hardware: ATAVRRZ600(with AT86RF231 radio module)
- Device: AT32UC3A3256S
- RF Transceiver: AT86RF231
- Target Supply: USB bus powered
- Target Clock: Internal RC
- Hex: AVR2068_Target.hex

4.1.4 QTouch Analyzer

- XML: UserBoardsMruLog.xml
- JPG: avr477.jpg
- Design file: a.qtdgn

For kit display of AVR477 Remote control in QTouch Analyzer, these files are necessary.They are available as part of the AVR2068 firmware package in the directory \AVR2068_Target\QTouch Analyzer.

4.2 Quick start

Figure 4-1. Keys and Wheel position.

- Mount two fully charged AAA batteries in the AVR477 Remote
- Program the AVR477 Remote Control Board with EXAMPLES_AVR477_REMOTE.hex
- Fuse settings on AVR477 Remote Control Board should be 0xFE 0x99 0xE2
- Program RZ600 with AVR2068_Target.hex
- Save 1-AVR477.jpg and a.qtdgn in \My Documents\QTouchComposer\UserBoards
- UserBoardsMruLog.xml already exists in this directory,edit this file such that the path matches the location of the a.qtdgn file
- For example, Add: <UserBoard ProjectId="1" ProjectPath="C:\Documents and Settings\username\My Documents\QTouchComposer\UserBoards\a.qtdgn" SolutionPath="" /> and replace username with local setting
- Launch QTouch Analyzer
- Connect RZ600 to a free USB port in the computer
- Wait for the red LED on the RZ600 MCU board to start blinking
- Keep the remote in vertical position and press and release the Reset switch located on the rear side of the remote
- The remote should beep once and the red LED on the RZ600 will turn off indicating completion of pairing after a finite delay (within 30 seconds)
- The USB device gets enumerated and Atmel QTouch Analyzer will now initiate connection to the Target
- QDebug Subscription data is transferred over air from AVR477 remote to Target and sent to QTouch Analyzer
- QTouch Analyzer is now in connected state and the AVR477 kit is displayed on QTouch Analyzer
- Press “Start Reading” button and go to the “Sensor Data” tab
- Six keys on the top of the picture are used to simulate the accelerometer status
- The rest of the keys and wheel are Touch sensors on the AVR477

4.3 Display / demo scenarios

4.3.1 RF4CE pairing

- Connect RZ600 to a free USB port in the computer
- Wait for the red LED on the RZ600 MCU board to start blinking
- Keep the remote in vertical position and press and release the Reset switch located on the rear side of the remote
- The remote should beep once and the red LED on the RZ600 will turn off indicating completion of pairing after a finite delay (within 30 seconds)

4.3.2 Touch

- On touching a key or wheel,QDebug data is transferred from AVR477 remote to Target and this is indicated via toggling of green LED status on RZ600 MCU board
- Any key or wheel touch will correspondingly turn the sensor GREEN in the AVR477 image in the QTouch Analyzer
- QDebug data (signals, references, states and deltas) are displayed in the Sensor View Control window --> Sensors
- Key touch and release will correspondingly be indicated by beep in the remote
- If there is no touch for an approximately 10 sec. interval, the remote goes to sleep for power saving and wakes up either on proximity sensing or on scheduled wake-up timer

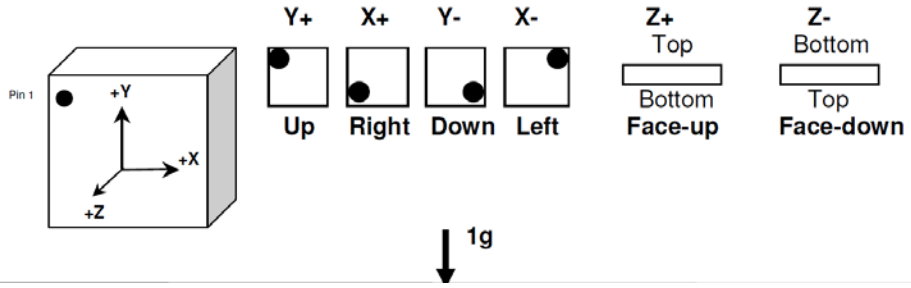
4.3.3 Proximity

- Proximity functionality is used only for wake-up from sleep. When MCU is awake proximity is disabled
- Picking up the remote in hand should be sensed and is indicated by turning ON both the LEDs in the center keys. These LEDs stays ON only for a short time even if held continuously in the hand

4.3.4 Accelerometer

- Only Tilt positions are simulated
- They are indicated by turning the corresponding sensor to GREEN color in the AVR477 image displayed in the QTouch Analyzer
- Sensors for Accelerometer tilt position for visualization in the QTouch Analyzer:
  - Sensor 9----------→ Z+
  - Sensor 10----------→ Z-
  - Sensor 11----------→ Y+
  - Sensor 12----------→ Y-
  - Sensor 13----------→ X-
  - Sensor 14----------→ X+

Figure 4-2. Accelerometers tilt positions.



4.3.5 Sleep

- Approximately 10 seconds without any key touch will put the remote into the sleep mode
- The LEDs will toggle alternatively indicating that the remote is entering the sleep mode
- No data will be communicated right from the time the LEDs start toggling, hence the QDebug data display in the Atmel QTouch Analyzer will stop
- Prox and Tilt position changes are not considered as key touch, hence even if the remote is held and moved (without touching any touch sensors) the remote may go into the sleep mode

4.3.6 Wakeup

- The remote can wake up either on proximity sensing or on the scheduled wake-up timer
- The accelerometer is switched OFF during the sleep mode and it is re-initialised after the remote wakes up. This can be visualized by the simulated accelerometer sensor display going to default facedown position just after wake-up
- Data communication will and QTouch Analyzer will also start receiving the data

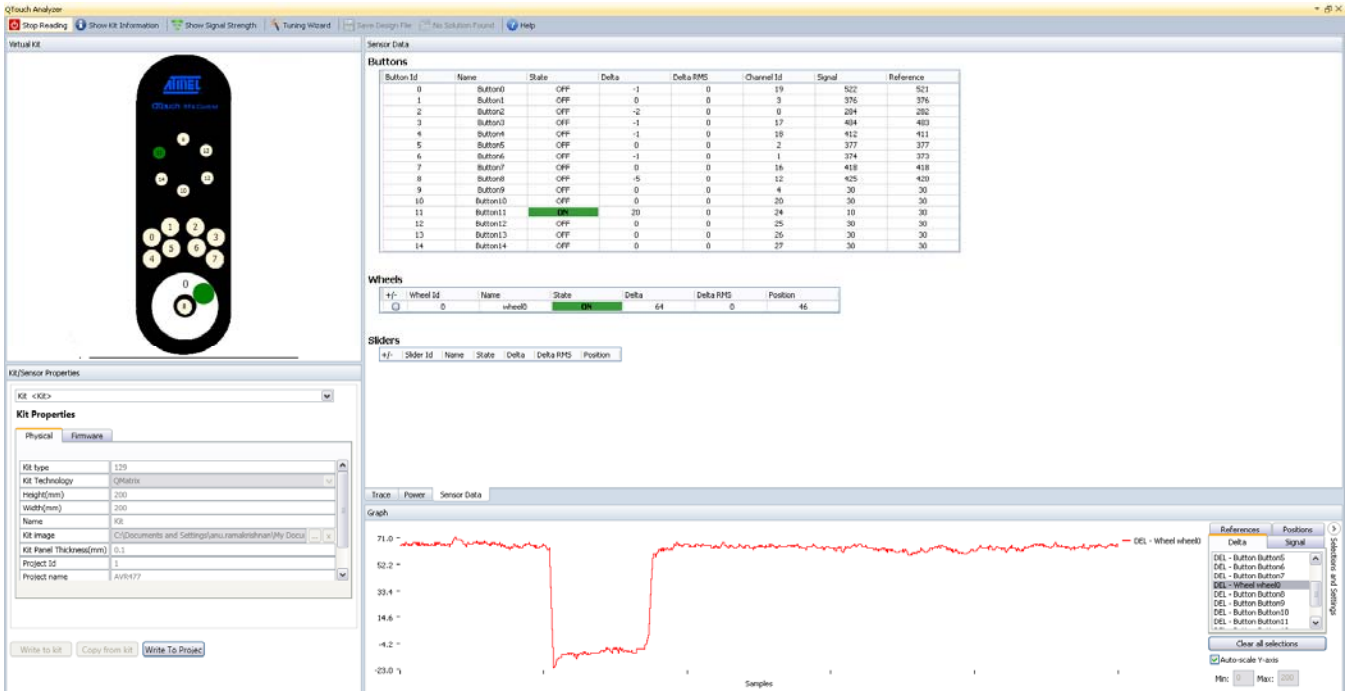
4.3.7 Fault indication

- Both the LEDs will toggle together on fault indication due to RF4Control Stack
- The red LED does not stop blinking if pairing is not performed or unsuccessful
- The green LED stops toggling when there is no reception of data over air from paired remote
- Plug out RZ600 from the USB port to break the communication, this will trigger fault indication on AVR477 remote
- The fault indication is common for all types of errors, and servicing of the fault indications is out of scope of this application note
- It is acceptable to receive fault indication once in a while, as this could happen because a packet is lost over the air
- If the fault indication persists (for example, broken link), recovery from this situation would be to restart the communication from the beginning by pairing

4.3.8 Start / Stop

- Start/Stop reading from the Atmel QTouch Analyzer will start and stop data reception from the Remote, and this can also be verified by the green LED toggling on the RZ600 MCU board

4.3.9 Atmel QTouch Analyzer Demo screen



5. References

[1]. AVR477: RF4Control – Touch Remote Control

[2]. Atmel AVR2102: RF4Control - User Guide: http://www.atmel.com/Images/doc8357.pdf

[3]. Atmel AVR4900: ASF - USB Device stack: http://www.atmel.com/Images/doc8360.pdf

[4]. Atmel AT32UC3A3256S datasheet: http://www.atmel.com/Images/doc32072.pdf

[5]. Atmel AT86RF231 datasheet: http://www.atmel.com/Images/doc8111.pdf

[6]. Atmel RZ600 kit: http://www.atmel.com/tools/RZ600.aspx

[7]. Atmel QTouch Library User Guide: http://www.atmel.com/Images/doc8207.pdf

6. Revision History

Doc. Rev.	Date	Comments
42036A	12/2012	Initial document release

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan G.K.
16F Shin-Osaki Kangyo Building
1-6-4 Osaki
Shinagawa-ku, Tokyo 141-0032
JAPAN
Tel: (+81)(3) 6417-0300
Fax: (+81)(3) 6417-0370