APPLICATION NOTE

Atmel AVR477: RF4Control – Touch Remote Control

8-bit Atmel Microcontroller

Features

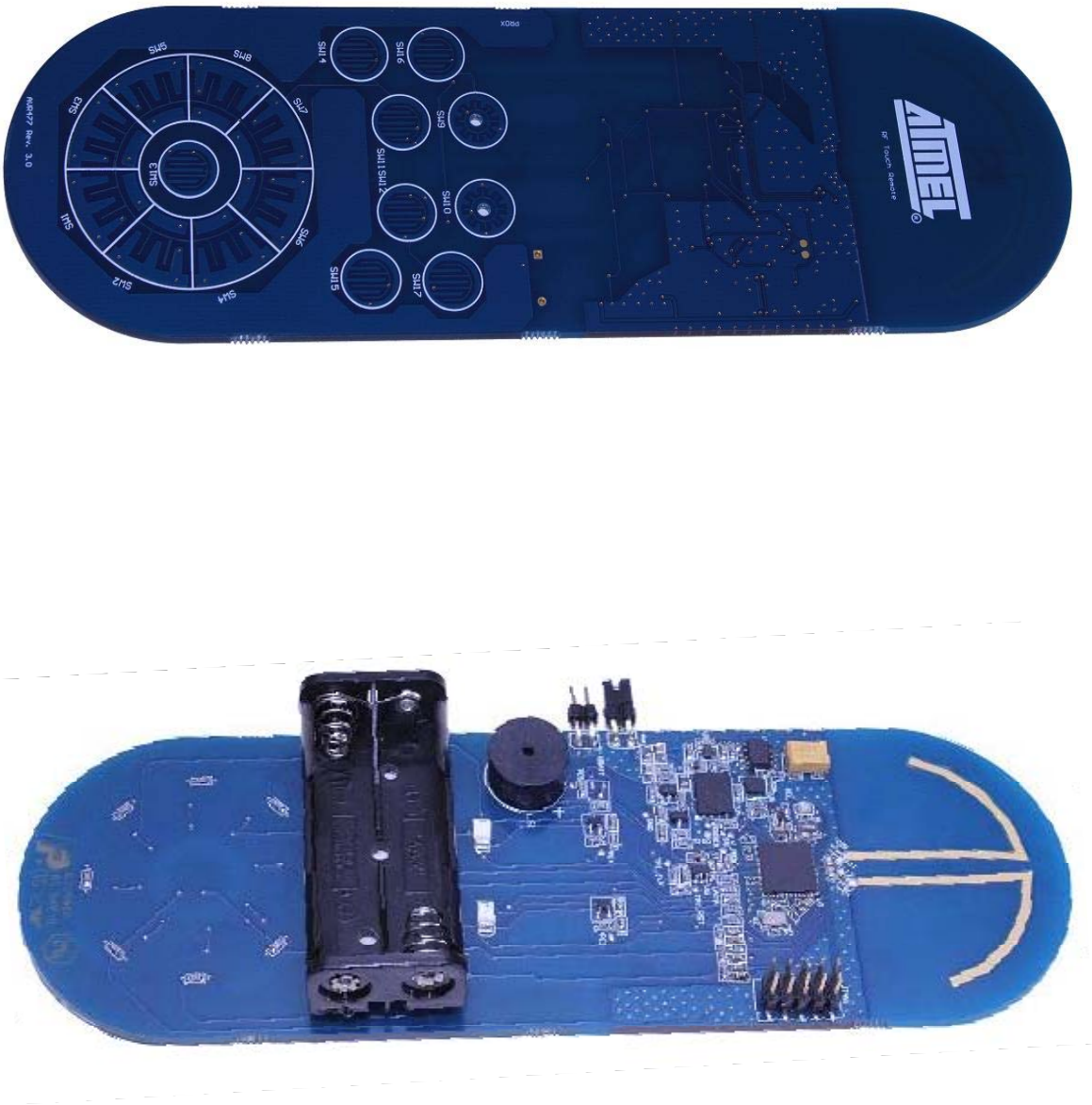
- 2.4GHz RF hardware with PCB antenna
- Atmel® ATmega128RFA1 centric design
- Nine capacitive touch (QMatrix) keys
- One wheel with 8-bit resolution
- Proximity sensing
- 3D accelerometer
- External EEPROM and DataFlash®
- TWI interface
- Buzzer control
- Backlight LED control
- Programming and debugging interfaces
- QDebug protocol support
- Target application platform
- Battery powered (2 × AAA)
- Provision for current measurement
- Provision for external regulator (Seiko)

Introduction

Zigbee® RF4Control provides a simple, robust, secure and low-cost communication network allowing wireless connectivity. Add to this, the best-in-class Capacitive Touch sensing technology for intuitive user interface from Atmel, the end product is an ideal remote controlling solution for numerous applications in consumer electronics appliances, multimedia/entertainment devices, home appliances, home automation, lighting control, etc.

RF4Control - Touch Remote Control is a complete Remote Control reference design centered around Atmel SOC ATmega128RFA1, featuring ZigBee RF4Control, Capacitive Touch and MEMS Sensors. This comprehensive application note details on the complete hardware and software implementation aspects involved in this design.

Figure 1. Atmel AVR477 remote control board.



For detailed information on Atmel Capacitive Touch (QMatrix) technology and associated products, please refer to http://www.atmel.com/products/touchsolutions.

For detailed information on RF4Control technology, please refer to the Atmel AVR®2102: RF4Control - User Guide [1].

Table of Contents

1. Remote Control board – hardware 4
   - 1.1 Microcontroller 5
   - 1.2 On-chip Radio Transceiver 5
   - 1.3 Clock sources 6
     - 1.3.1 Radio transceiver clock 6
     - 1.3.2 Microcontroller clock 6
   - 1.4 Antenna 7
   - 1.5 Capacitive touch sensors 7
     - 1.5.1 QMatrix electrode design 8
     - 1.5.2 Key 8
     - 1.5.3 Keys with LED with recess 9
     - 1.5.4 Wheel 9
     - 1.5.5 Proximity sensing 9
   - 1.6 Accelerometer 11
   - 1.7 LED / buzzer design 11
   - 1.8 External EEPROM 12
   - 1.9 External DataFlash 13
   - 1.10 Power Supply 13
     - 1.10.1 VDD 13
     - 1.10.2 Power supply considerations 13
   - 1.11 Test points 13
   - 1.12 Programming and debugging 14
     - 1.12.1 JTAG programming 14
     - 1.12.2 ISP Programming 14
     - 1.12.3 TPI programming 14
2. Terminal Target board – hardware 14
3. Atmel AVR477 – firmware 15
   - 3.2 Remote Control 17
   - 3.3 Terminal target 17
   - 3.4 ZRC Target example application 18
4. Quick start guide 20
   - 4.1 Hardware setup 20
     - 4.1.1 Tools required 20
     - 4.1.2 Remote 20
     - 4.1.3 Terminal target (RCB128RFA1) 20
     - 4.1.4 QTouch Analyzer 20
   - 4.2 Quick start 20
   - 4.3 Display / demo scenarios 21
     - 4.3.1 RF4CE pairing 21
     - 4.3.2 Touch 21
     - 4.3.3 Proximity 21
     - 4.3.4 Accelerometer 22
     - 4.3.5 Sleep 22
     - 4.3.6 Wakeup 22
     - 4.3.7 Fault indication 22
     - 4.3.8 Start / Stop 23
     - 4.3.9 Atmel QTouch Analyzer demo screen 23
5. References 24

Appendix A. Antenna characteristics 25

Appendix B. Revision history 29

1. Remote Control board – hardware

The Atmel ATmega128RFA1 is a SoC that integrates a powerful 8-bit AVR RISC Microcontroller, an IEEE® 802.15.4-compliant transceiver, and additional peripheral features. The built-in radio transceiver supports the worldwide accessible 2.4GHz ISM band.

Figure 1-1. System block diagram.

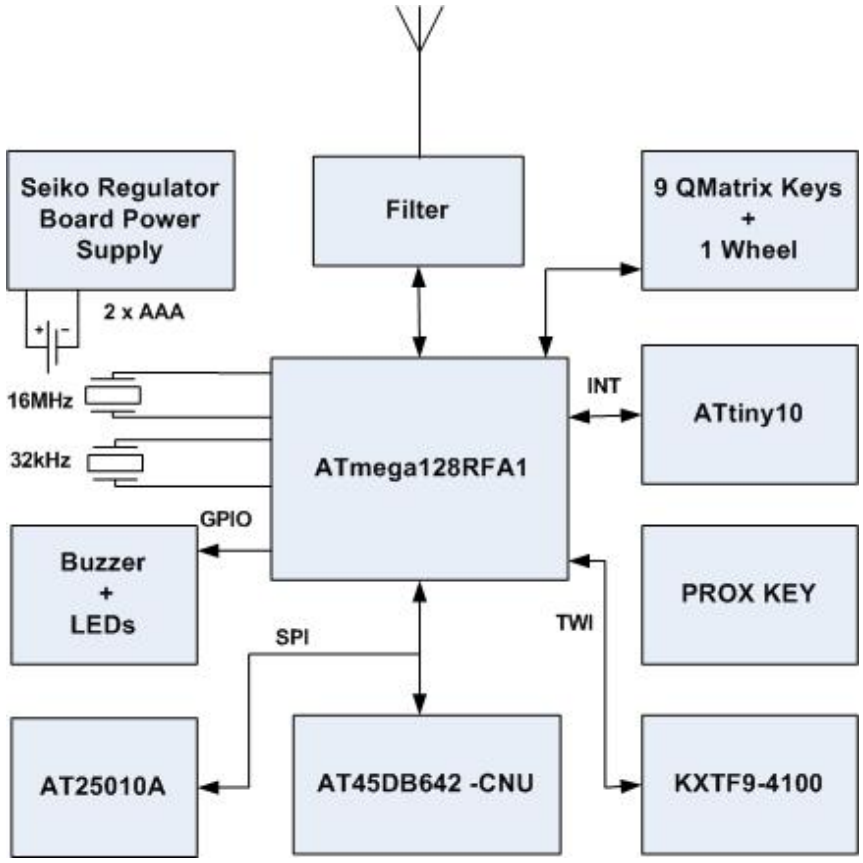


Figure 1-1 illustrates the Remote setup in general. It mainly consists of an Atmel ATmega128RFA1 and some peripheral circuitry. An external EEPROM is used to store MAC address and additional board information to avoid accidental data erasure during microcontroller firmware development.

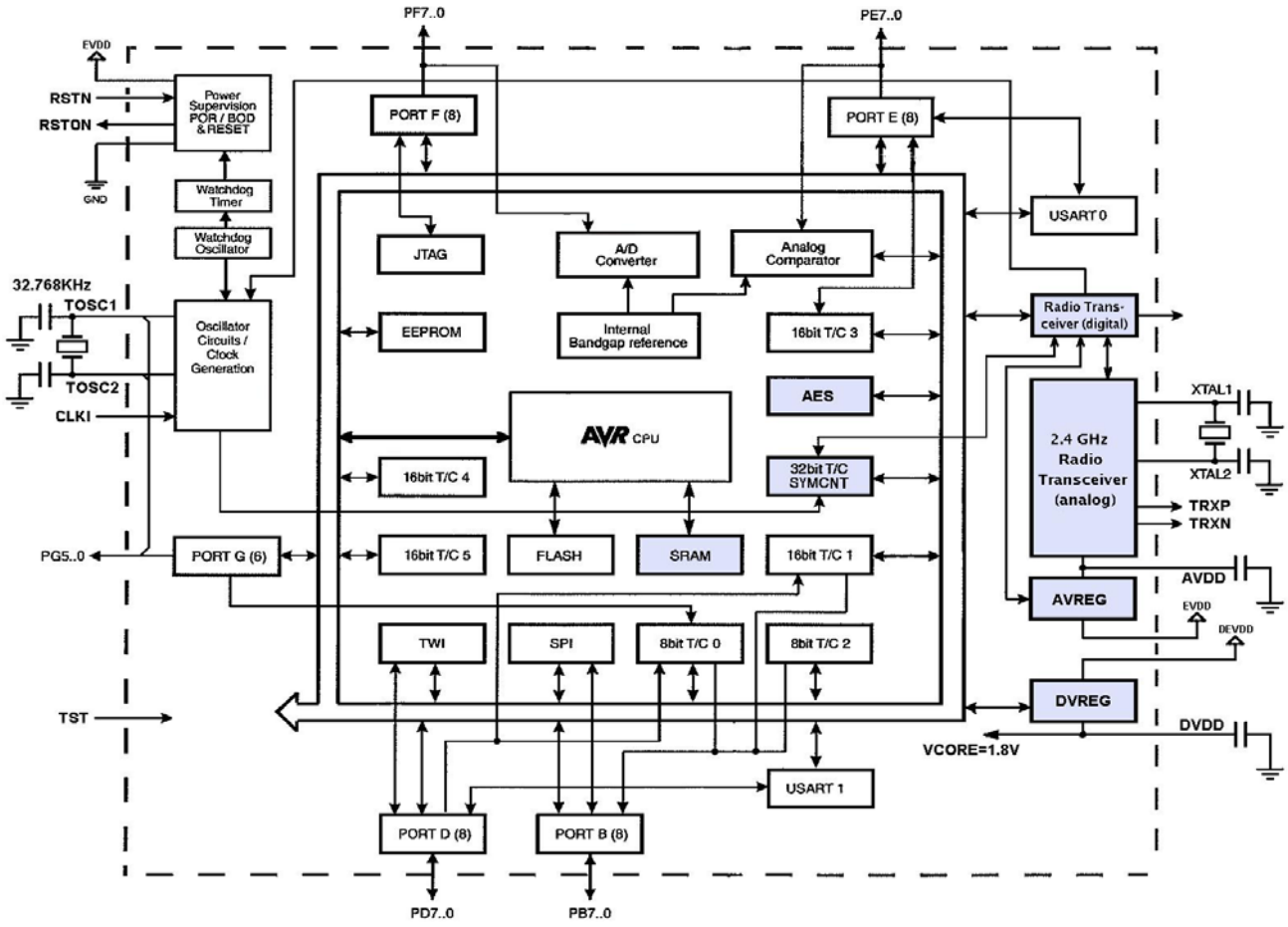
The radio transceiver incorporates MAC hardware accelerators to handle all RF modulation/demodulation, signal processing, frame reception and transmission actions. Further information about the radio transceiver and the microcontroller are provided in the Atmel ATmega128RFA1 datasheet [2] on the Atmel website. The RF front-end implementation was kept minimal by using a PCB antenna. A dipole antenna designed to match ATmega128RFA1 is used.

The system has nine touch keys and a wheel for user interface based on the QMatrix technology patented by Atmel. This board is used to detect the key touch or wheel position and communicate the corresponding key/wheel data to the target via the RF4Control. Proximity and accelerometer functionalities are implemented by using the Atmel ATtiny10 device and Kionix KXTF9 respectively. The backlight LEDs and Buzzer are controlled by the application for various user indications such as key touch, proximity, sleep and fault indications. This board also incorporates the Adjacent Key Suppression® (AKS®) technology patented by Atmel, which allows only the dominant key to register a touch and suppresses unintended touch.

Other features include; external EEPROM, external data Flash, 32kHz crystal, 16MHz transceiver crystal, JTAG programming header and Jumper provision for current measurement. Control for switching (ON and OFF) accelerometer and proximity controller for better power consumption. Provision for external Seiko make 1.8V voltage regulator. All components are placed on one side of PCB to demonstrate a low-cost manufacturing solution.

1.1 Microcontroller

Figure 1-2. Atmel ATmega128RFA1 block diagram.



The ATmega128RFA1 integrates a low-power; 8-bit microcontroller based on the AVR enhanced RISC architecture. It is derived from the Atmel ATmega1281 Microcontroller and the Atmel AT86RF231 Radio Transceiver. The 128kB non-volatile flash program memory and 16KB internal SRAM, supported by a rich set of peripheral units, makes it suitable for a full-function sensor network node.

By executing powerful instructions in a single clock cycle, the device achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

The microcontroller is capable of operating as a PAN-coordinator, a full-function device (FFD) or a reduced function device (RFD), as defined by IEEE 802.15.4. However, this hardware is not limited to these, and can be programmed to operate in other standards or ISM applications too.

The ATmega128RFA1 is designed to operate at full 16MHz speed over the complete supply voltage range from 1.8V to 3.6V.

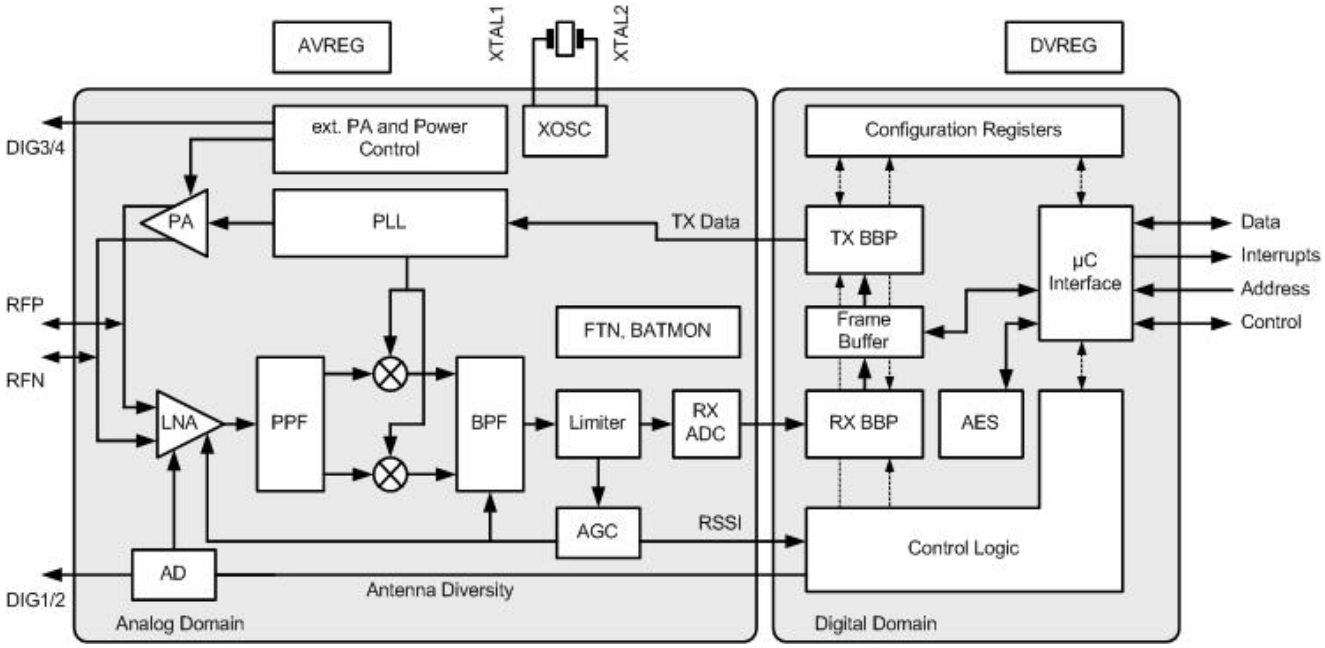
1.2 On-chip Radio Transceiver

Besides an 8-bit Atmel AVR Microcontroller, the ATmega128RFA1 also integrates an IEEE 802.15.4-compliant Radio Transceiver. RF and baseband critical components are integrated to transmit and receive signals according to IEEE 802.15.4 or proprietary ISM data rates.

It comprises a complex peripheral component containing the analog radio, digital modulation and demodulation including time and frequency synchronization and data buffering. The number of external components for the transceiver operation is minimized so that only the antenna, the crystal and decoupling capacitors are required. The bidirectional differential antenna pins (RFP, RFN) are used for transmission and reception, thus no external antenna switch is needed.

The transceiver block diagram of the Atmel ATmega128RFA1 is shown in Figure 1-3.

Figure 1-3. Transceiver block diagram.



1.3 Clock sources

1.3.1 Radio transceiver clock

The integrated radio transceiver is clocked by a highly accurate 16MHz reference crystal. Operating the node according to IEEE 802.15.4, the reference frequency deviation must be within ±40ppm. The absolute clock frequency is mainly determined by the external load capacitance of the crystal, which depends on the crystal type and is given in its datasheet.

The radio transceiver reference crystal, Q2, must be isolated from fast switching digital signals and surrounded by a grounded guard trace to minimize disturbance of the oscillation.

The Atmel AVR477 uses a SIWARD SX4025 crystal with two load capacitors. To compensate for fabrication and environment variations, the frequency can be tuned with the transceiver register, XOSC_CTRL (0x12).

By setting the fuses accordingly, the microcontroller can also be clocked by the 16MHz radio reference crystal.

1.3.2 Microcontroller clock

The ATmega128RFA1 provides several clock source options for the internal microcontroller:

- 16MHz calibrated internal RC oscillator
- 128kHz internal RC oscillator
- 16MHz radio reference crystal

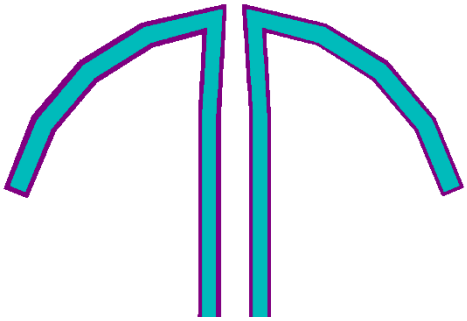
The calibrated internal RC oscillator, pre-scaled to 8MHz, is used as the default clock. It is recommended to use the MAC symbol counter clocked from the 16MHz radio reference crystal as a reference to calibrate the RC oscillator for higher accuracy.

A 32kHz crystal is connected to the Atmel ATmega128RFA1 pins (17-TOSC2; 18-TOSC1) to be used as a low-power, real-time clock. This time base can also run in sleep mode and create timer-based system wake-up events.

1.4 Antenna

The antenna used in this design is a bent dipole structure and is calculated to match 100Ω at feed point.

Figure 1-4. Dipole antenna.

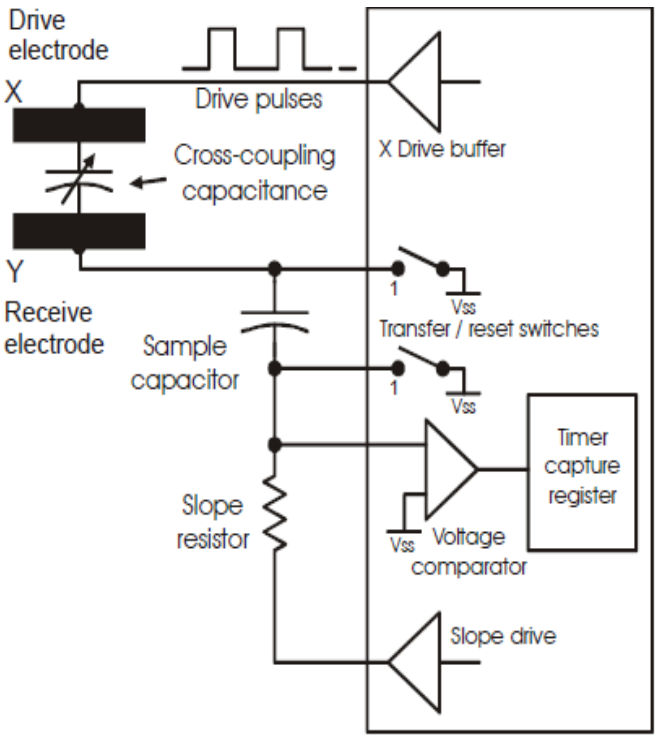


Further details on the antenna can be obtained from the report available in Appendix A.

1.5 Capacitive touch sensors

All the capacitive touch sensors (Keys and Wheel) are implemented using the Atmel QMatrix acquisition method. The QMatrix acquisition method detects touch using a scanned, passive matrix of electrode sets to achieve a large number of touch keys driven by a single chip. This configuration leads to high pin count efficiency in the sensing chip and allows for a small number of connections between the chip and the key matrix. QMatrix circuit offers very good signal-to-noise ratios, high level of immunity to moisture films, high level of temperature stability, low power characteristics and small IC package sizes for a given key count. For these reasons, QMatrix circuits are highly valued for automotive and consumer electronics.

Figure 1-5. QMatrix circuit diagram.



The basic benefits of the Atmel QMatrix technology can be summarized as:

- High key-count to chip pin-count efficiency
- Compatible with inexpensive 1-sided PCB materials
- Temperature stability
- Naturally moisture suppressing
- Highly immune to external RF fields
- Low-power modes
- Can be used to make clear touch screens

Lighted and back lighted keys are always of interest to designers. LEDs can be placed in the middle of QMatrix keys simply by creating an opening in the PCB and working the electrodes around a surface mount LED and its pads.

The general concept of capacitive touch sensing using QMatrix is illustrated below.

1.5.1 QMatrix electrode design

Capacitive touch sensing relies on sensing an unknown capacitance relative to a known reference capacitance (sampling capacitance). If the capacitance measured is large before touch, the added capacitance by touch (approx. 2 - 5pF) will not be detected. The reference capacitance (sampling capacitance) has to be larger than the sense capacitance (touch electrode capacitance) to get a reliable signal difference for touch and no touch. Typical values for the sense capacitance are 2 – 50nF.

QMatrix uses a pair of sensing electrodes for each channel. One is an emitting electrode into which a charge consisting of logic pulses is driven in burst mode. The other is a receive electrode that couples to the emitter via the overlying panel dielectric. When a finger touches the panel the field coupling is changed, and touch is detected.

For this application a small touch electrode is used to easily fit in a light switch.

To adjust the sensitivity of the sensors, the sampling resistor (Rsmp), can be changed. Increasing the Rsmp will prolong the sample capacitor, Cs, discharge time and hence the discharge slope will be shallower providing better resolution. Alternatively the Burst length of the sensors can be adjusted individually to increase/decrease the sensitivity. The burst and thus the count from the touch circuit on the electrode can be seen with an oscilloscope.

The sensors are designed using the Planar construction method (single layer design). In this method, both the X and Y electrodes and the sensor traces are fabricated on the same plane of the insulating substrate. Field propagation relies heavily on the overlying panel, since most of the coupling field flow is horizontal. The X and Y electrodes are designed to optimize the distribution of the electric field that is coupling through the overlying panel, to maximize the touch sensitivity.

1.5.2 Key

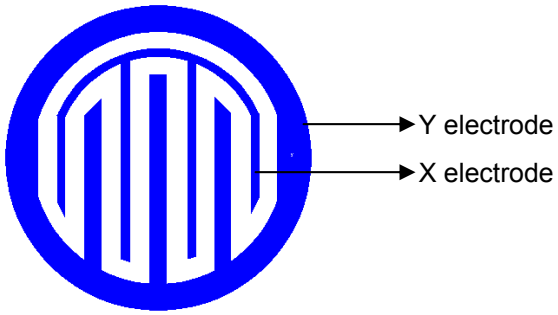
The X and Y electrodes are generally interdigitated, that is they form interlocking fingers. Typically the X electrode surrounds the Y electrode to contain the field with in the two. The key has a diameter of 9mm.

The Y electrode should use the thinnest possible (0.1mm to 0.5mm) for the fingers, as this minimizes the possibility of noise coupling to the sensor during touch.

For the X electrodes, wider electrodes are generally preferred as they tend to act as partial shields for the Y traces. The width of the X fingers should be calculated from thickness of the overlying panel (T). Generally, the function T/2 is used.

As with the width of the X fingers, the spacing between the X and Y electrodes should be T/2.

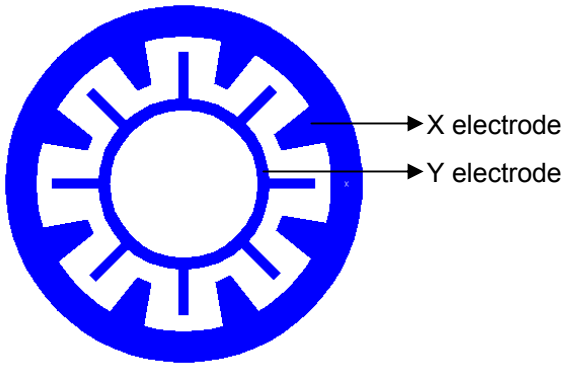
Figure 1-6. QMatrix key.



1.5.3 Keys with LED with recess

The recess is provided for maximum light dispersion underneath the keypad.

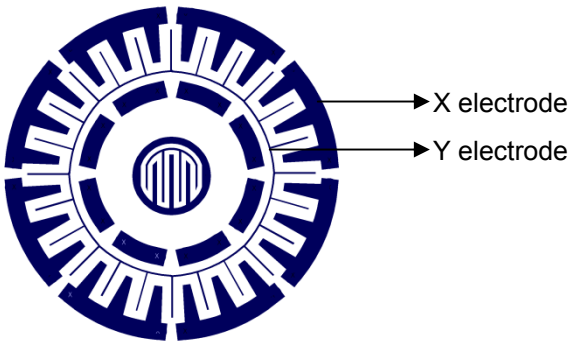
Figure 1-7. QMatrix key with LED and recess.



1.5.4 Wheel

The array of the keys is arranged in a circular form and there is no end border. To increase the diameter of the sensor, extra segments are created using resistive dividers on the X-lines. The Wheel is connected to four X-Lines and one Y-Line. It has eight resistively interpolated segments for supporting the bigger size of the key. The Wheel has an outer diameter of 38mm and inner diameter of 16mm. The Y electrode is shielded within the X electrode for better coupling and confinement of the electric field. The design and spacing requirements are similar to the key.

Figure 1-8. QMatrix Wheel with keys in centre.



1.5.5 Proximity sensing

A proximity sensor is used to wake the Remote Control from deep sleep as the user approaches it and picks it up by hand.

The Atmel ATtiny10 is used for proximity sensing. The proximity sensor is designed to cover the entire bottom half of the board.

For increasing the sensitivity one can increase the sensor area as well as the sampling capacitor (CS4) and vice a versa.

Provision to control the proximity sensor is provided in the hardware. The proximity sensor is switched ON just before the main MCU goes to sleep and switched OFF after wakeup, thereby reducing the power consumption.

Interrupt (PD2) from the poximity sensor is used to wake up the Atmel ATmega128RFA1 from sleep.

The ATtiny10 can be programmed via TPI, which can be wired up from the test points provided.

Figure 1-9. Proximity key.

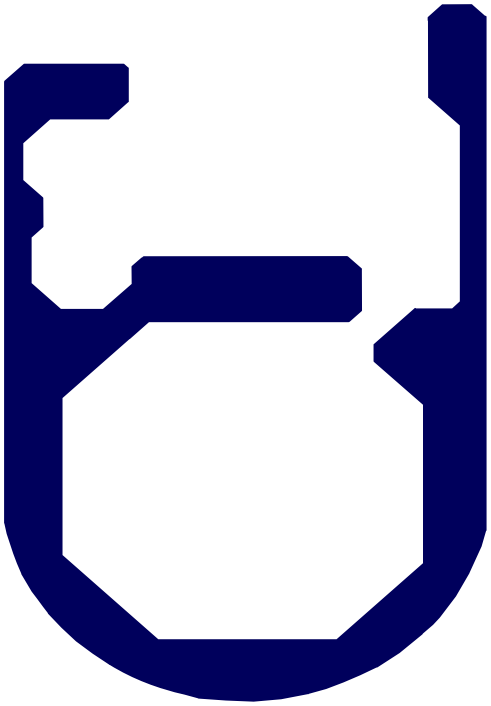
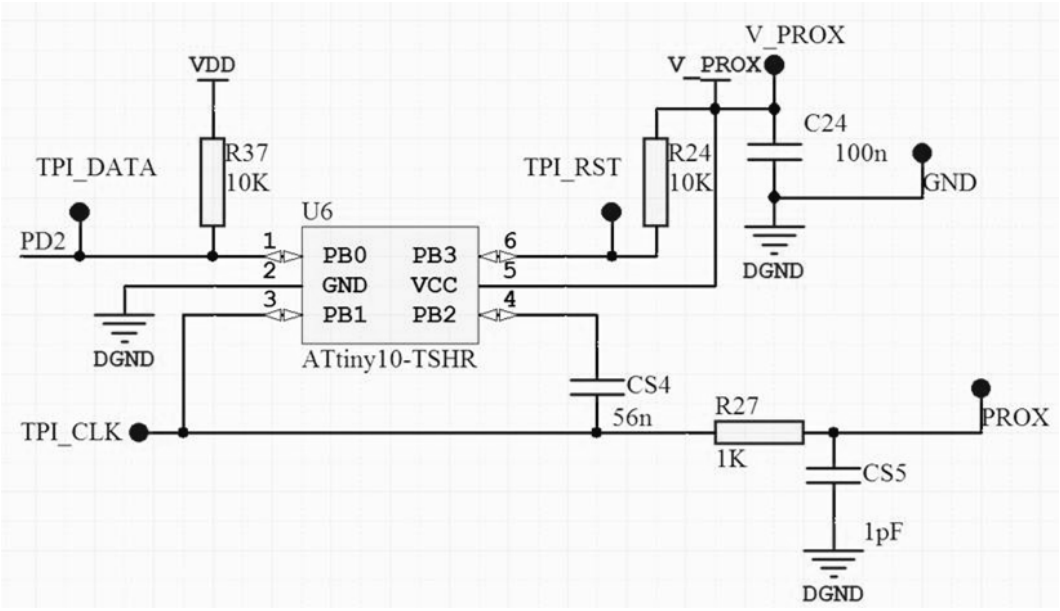


Figure 1-10. ATtiny10 as proximity sensor.



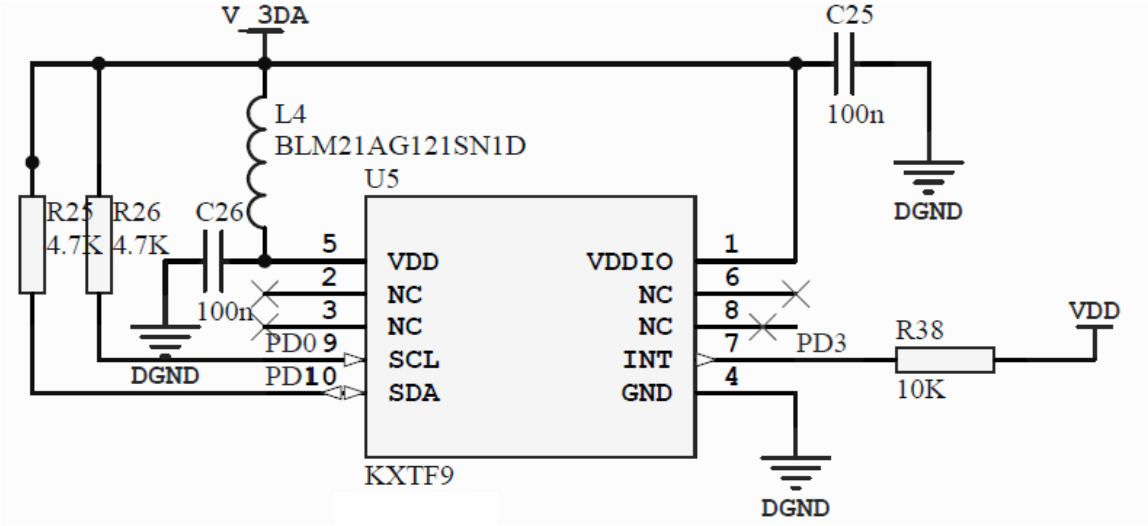
1.6 Accelerometer

The KXTF9 is a tri-axis ±2g, ±4g or ±8g silicon micro machined accelerometer with integrated orientation, tap/double tap and activity detecting algorithms. The sense element is fabricated using Kionix's proprietary plasma micromachining process technology. Acceleration sensing is based on the principle of a differential capacitance arising from acceleration-induced motion of the sense element, which further utilizes common mode cancellation to decrease errors from process variation, temperature and environmental stress.

The main Atmel ATmega128RFA1 MCU communicates with the KXTF9 via the TWI interface for configuring the accelerometer and receiving the accelerometer data. The physical interrupt pin of the accelerometer is connected to the port pin PD3 of the ATmega128RFA1, and can be used as an interrupt source for the ATmega128RFA1.

Provision to switch ON/OFF the accelerometer is provided in hardware for better power management.

Figure 1-11. KXTF9 – 4100 as accelerometer.



1.7 LED / buzzer design

If LEDs are close (less than 4mm away) to capacitive sensors, their change in capacitance between the on and off states should be considered. It is also necessary to consider the changes in the nature of their drive circuitry.

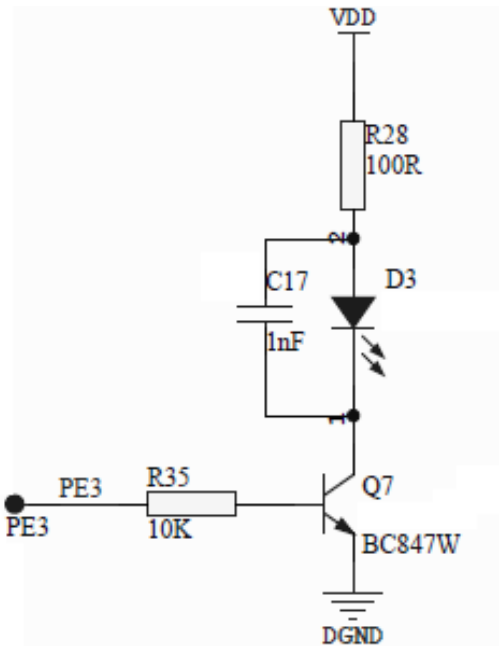
LEDs that are judged as close must be bypassed with a capacitor that has a typical value of 1nF.

Note: The bypass capacitor does not need to be physically close to the LED itself. Even if the capacitor is several centimeters away form the LED, it will serve the purpose. This may aid layouts that are tight for space around the sensors.

The backlight LEDs are within the key, hence the capacitors are provided. The LEDs are driven by transistors and shows the placement of the bypass capacitor for the LED.

The buzzer can also be driven by the transistor and can be driven by external source.

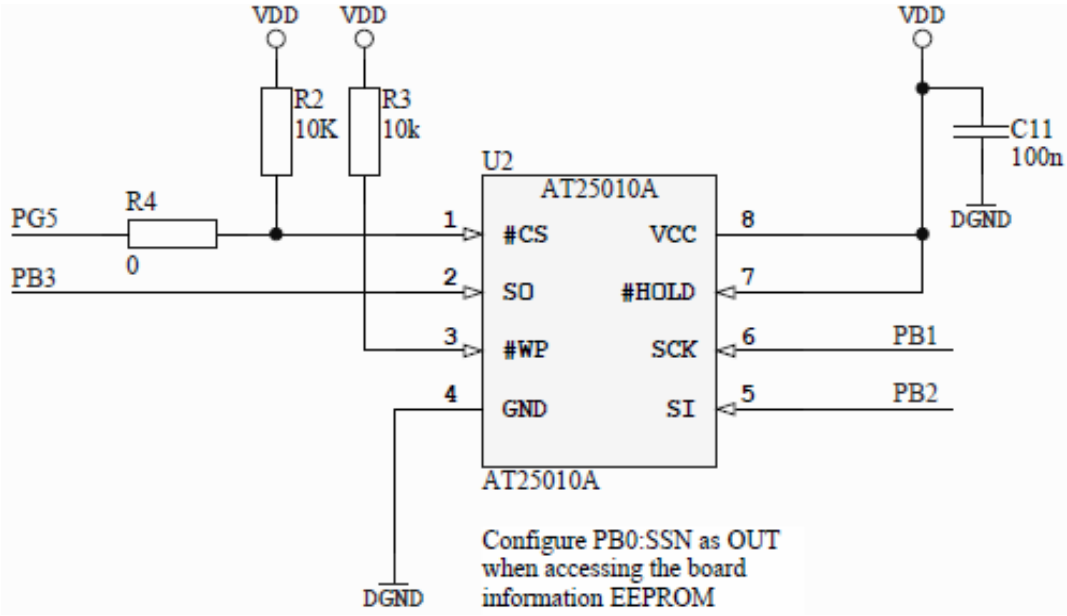
Figure 1-12. Bypass capacitor for LED.



1.8 External EEPROM

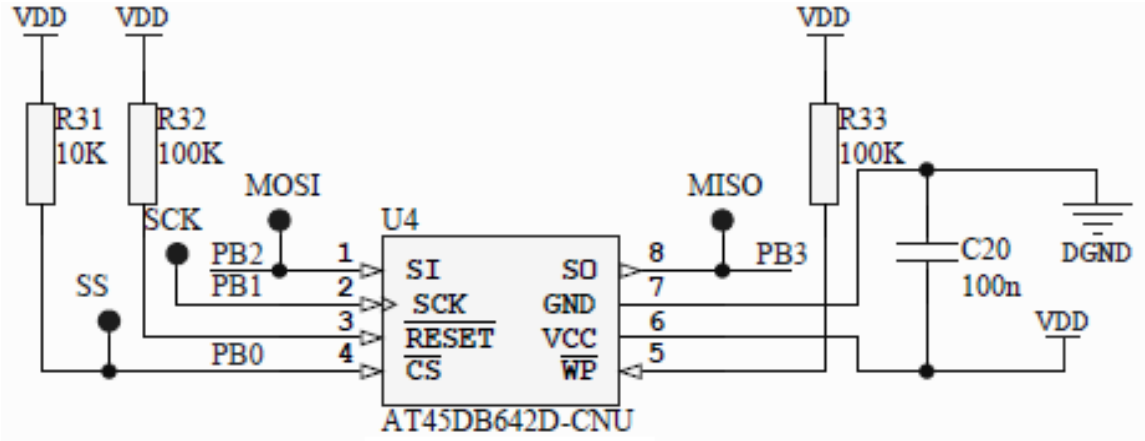
An Atmel AT25010M EEPROM with 128 × 8-bit organization is used because of its small package, low voltage and low-power operation. The EEPROM is connected via an SPI interface to the Atmel ATmega128RFA1. Accessing the EEPROM requires PG5 set to logic low. External EEPROM can be used to store the 64-bit MAC ID and any other user data.

Figure 1-13. EEPROM access decoding logic.



1.9 External DataFlash

Figure 1-14. DataFlash connectivity.



The serial-interface (SPI) sequential access flash memory Atmel AT45DB642D-CNU is part of the Atmel AVR477 design. This can be used for storing a flash image during over-the-air firmware upgrade or for any other user application needs.

1.10 Power Supply

1.10.1 VDD

The remote control hardware can be powered by two AAA batteries or external power supply as well. If external power supply is used, the supply voltage should be within 3.3V. While using an external power supply the considerations as mentioned in Section 1.10.2, should be taken into account.

The Atmel ATmega128RFA1 has an internal 1.8V regulator; the user can either choose to use the internal regulator itself or the external one. The hardware design supports the option to use external (Seiko make) regulator.

1.10.2 Power supply considerations

If the external power supply varies slowly with the temperature, the QMatrix acquisition methodology will track and compensate for these changes automatically with only minor changes in sensitivity. If the supply voltage fluctuates or shifts quickly, the drift compensation mechanism will not be able to keep up, causing sensitivity anomalies or false detections.

A single 0.1µF ceramic bypass capacitor, with short traces, should be placed very close to the supply pins.

It is recommended that the supply ripple and noise be not more than ±25mV.

1.11 Test points

There are 12 test points available on the board for testing, programming and debugging purpose.

Table 1-1. Test points.

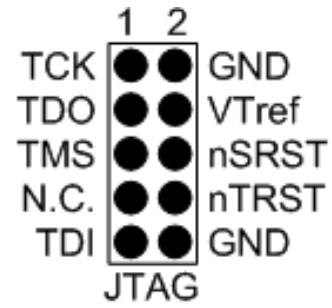
PROX	SPI	TWI	TPI
GND	MISO	SCL	TPI_CLK
PROX	MOSI	SDA	TPI_DATA
V_PROX	SCK		TPI_RST
	SS		

1.12 Programming and debugging

1.12.1 JTAG programming

The Atmel ATmega128RFA1 can be programmed through the JTAG interface. This is accessible through a programming header available on the board.

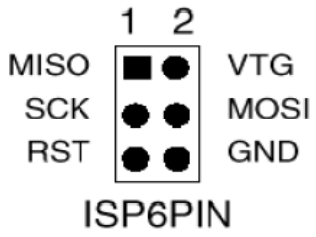
Figure 1-15. JTAG programming header.



1.12.2 ISP Programming

The ATmega128RFA1 can be programmed via ISP. Test points are provided on the board to wire up for ISP.

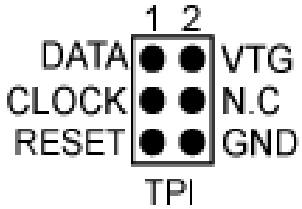
Figure 1-16. ISP connections.



1.12.3 TPI programming

The Atmel ATtiny10 device can be programmed via TPI. Test points are provided on the board to wire up for TPI.

Figure 1-17. TPI connections.



2. Terminal Target board – hardware

The Atmel RCB128RFA1 along with the STB can be used as terminal target with the Remote control for complete application setup.

Terminal target application is run on the RCB128RFA1 with the STB which can be interfaced with the Atmel QT™600-USB board. In this case, the ATmega128RFA1 controls the reset of the Atmel AT90USB device on the QT600-USB board. The following connections should be made for QDebug data transfer and Reset control.

The RCB128RFA1 and the STB are parts of the Atmel RF4CE-EK [4] kit, while the QT600-USB board [5] is a part of the Atmel QT600 kit.

Table 2-1. Connectors.

STB Connector	QT600 Connector
SDA	J300-1
SCL	J300-2
RSTN	J100-6

Note: The Atmel RZ600 Module (RZ600 USB Board + RF231 Radio Board) can also be used as Terminal Target. Please refer Application note AVR2068:RF4CE-HID QTouch Analyzer for AVR477 [6]

3. Atmel AVR477 – firmware

The AVR477 implements the Atmel RF4Control - Touch remote control application on the Atmel ATmega128RFA1 SoC, integrating the RF4Control 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 RF4Control frame format and sends it over the air. The AVR477 firmware also includes the terminal target application that transfers the received data to the Atmel QTouch® Studio. The application architecture for the AVR477 implementation is described Figure 3-1.

The firmware is available for download along with this application note. The firmware is fully documented using Doxygen documentation. OTAU functionality is available as an individual application. More details on OTAU functionality can be obtained from the Atmel AVR2102: RF4Control - User Guide.

Figure 3-1. AVR477 architecture.

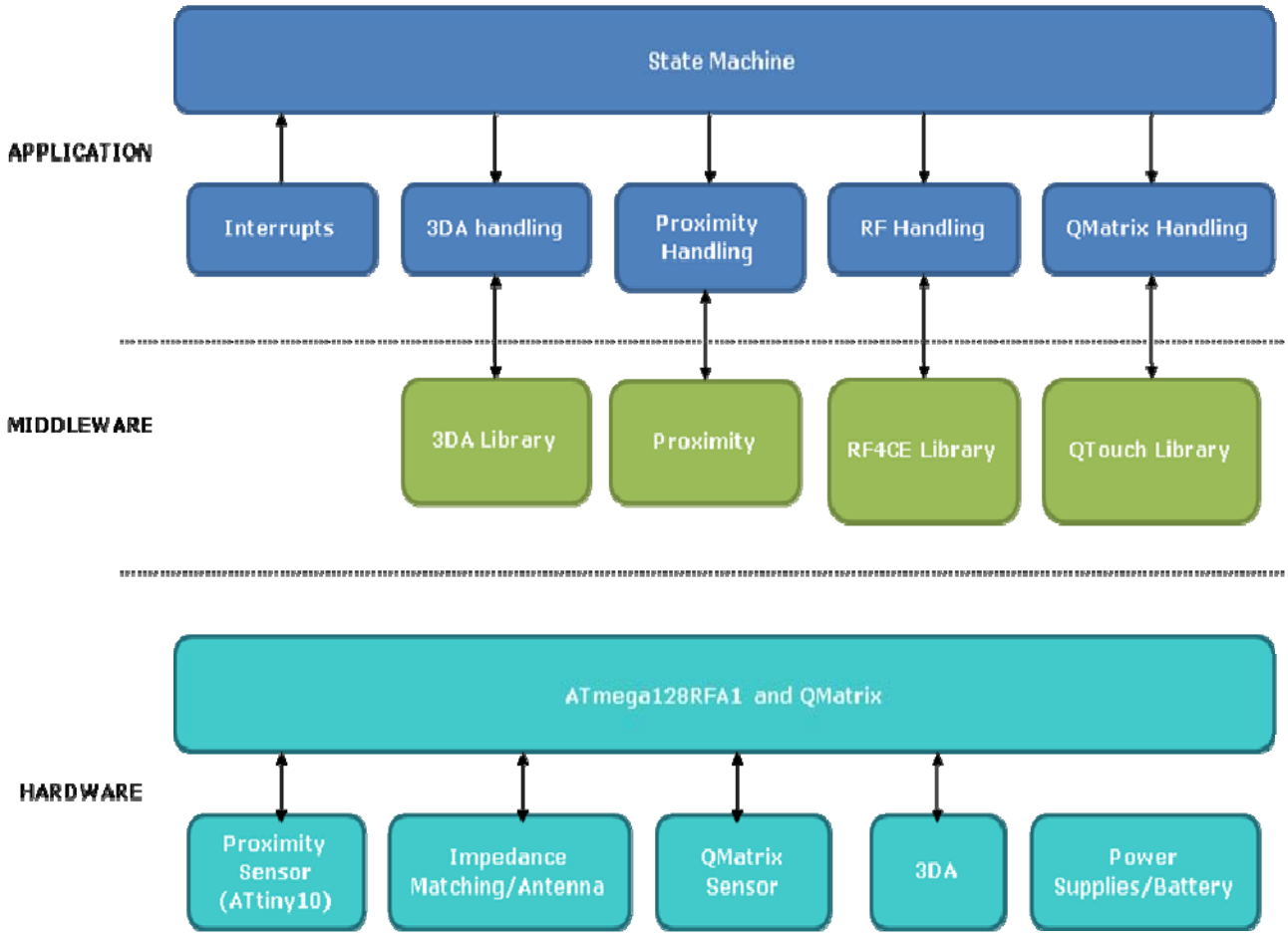
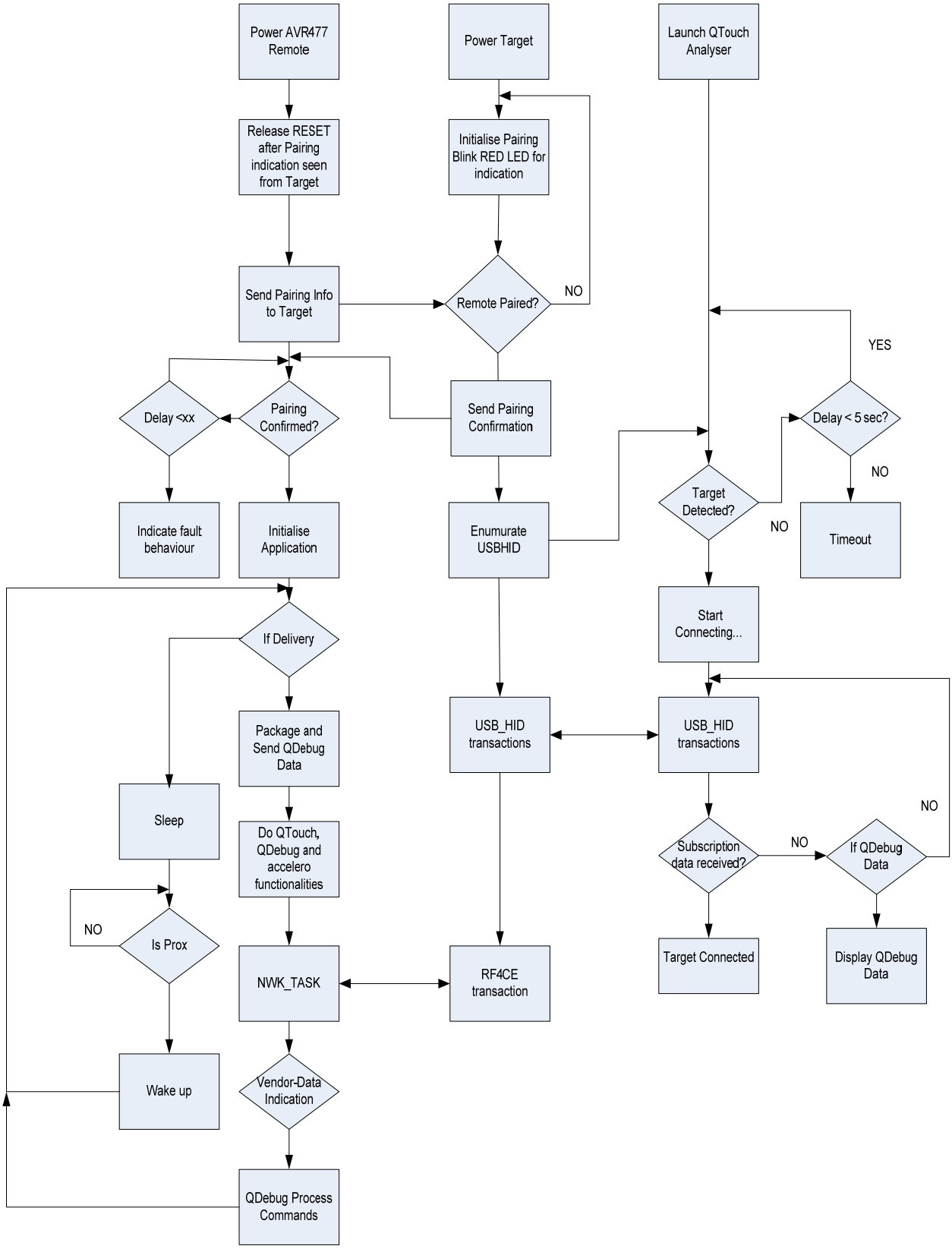


Figure 3-2 describes the flow (interactions) between the Remote, Target and Atmel QTouch Analyzer.

Figure 3-2. AVR477 program flow.



3.2 Remote Control

The remote firmware integrates the Atmel QTouch Library [3] with the Atmel RF4Control application. The Single Button Controller application is taken as reference for the remote control implementation. More details on the stack and standard applications can be obtained from the Atmel AVR2102: RF4Control - User Guide [1].

The Atmel QTouch (QMatrix) Library is used to sense nine Capacitive Touch (QMatrix) keys and one Wheel/Rotor with 8-bit resolution (256 steps). The Atmel QDebug is enabled in the application to collect the sensor data, which are transferred via the Atmel RF4Control to the target, which in turn transmits to the Atmel QTouch Analyzer for display.

The remote also implements TWI driver for accessing accelerometer data from the external 3DA accelerometer. For display and demonstration purpose of this accelerometer data, the application configures six unused channels in QTouch and changes the Reference and Signal values based on the accelerometer data read via TWI to display the current Accelero position.

The firmware residing in the Atmel ATtiny10 device is generated from the ATtiny10 Single Key Configurator tool. A proximity sensor is used as a wakeup source in this application thereby saving power consumption when the remote is idle and there is no touch operation. On proximity, there is a level interrupt triggered and LED indication is provided. The ATtiny10 device normally sleeps and is awake only just before the main MCU goes to sleep. Disabling the proximity during the normal operation saves power.

The Remote control application sends the QDebug data to the target in the RF4Control frames. When any QDebug command from the QTouch Analyzer via Target is received, it will be processed and necessary actions will be taken.

Any fault behaviour from the stack will be indicated by LEDs blinking and any further handling is out of scope of the reference application.

The remote firmware can be upgraded over-the-air and a separate application (ZRC_Target and ZRC_Serial_Interface) is provided for this purpose. Separate area within the internal flash is used for storing the firmware received over the air.

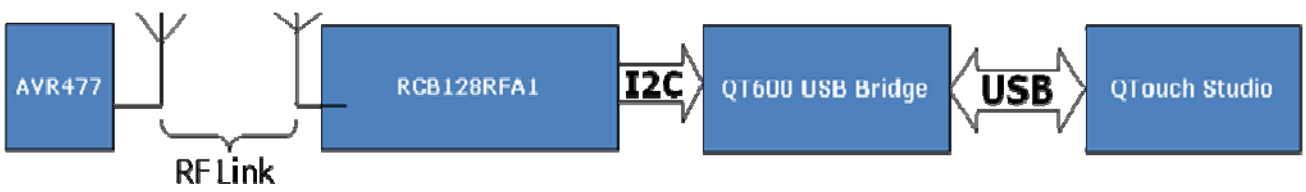
Buzzer indication is provided on every QTouch key status change this means on every touch and every release there will be a beep from the buzzer.

3.3 Terminal target

This target mainly receives the QDebug data transferred from remote via RF4Control and communicates it to QT600-USB board via TWI which in-turn communicates the same through its proprietary (QDebug) protocol to QTouch Analyzer. The hardware setup required for this implementation is mentioned in Chapter 2 RCB128RFA1 with Sensor Terminal Board (STB) and QT600-USB board.

The implementation is similar to the RZ600 target implementation with respect to the bridge functionality. After pairing with remote is done the target resets the MCU on the QT600-USB board to start connecting to the QTouch Analyzer.

Figure 3-3. AVR477 <-> RCB128RFA1 and QT600-USB board.



Note: The Atmel RZ600 Module (RZ600 USB Board + RF231 Radio Board) can also be used as Terminal Target. Refer to the Application note AVR2068:RF4CE-HID QTouch Analyzer for AVR477 [6] for implementation details.

3.4 ZRC Target example application

The ZRC Target application is a basic DOS application in which an option menu similar to the one implemented using a HyperTerminal window is implemented in the DOS shell. Over-the-air firmware upgrade option is supported via its extended menu options.

Firmware Over-The-Air (FOTA) upgrade menu option is used to perform a remote firmware upgrade of the controller node. First, the new image data file is transmitted via the serial link from the host PC to the client target node and then, via the wireless link to the controller node to perform the remote firmware upgrade.

The application can be built using a GCC compiler such as the one provided by MinGW (link below). To build the application execute the make command in the ZRC_Target directory.

http://www.mingw.org or http://sourceforge.net/projects/mingw/

Once the DOS application is up and running a serial connection to the Sensor Terminal Board can be established by opening a serial communication port.

The ZRC Target application uses a dedicated firmware module to control the serial communication between the host and the client. This serial interface handles all data exchange between the DOS shell host and the client hardware as displaying the options menu and issuing commands to the client. This interface is also capable to transmit large data files during a remote firmware upgrade via the client to controller node.

Since the implementation of the ZRC Target is very basic, a new firmware upgrade image for the controller must be pre-compiled and must use new_firmware.hex as the file name and placed in the ZRC_Target directory.

The example communication sequence between the target and the controller to download the new firmware image to a controller are shown in Figure 3-4.

The steps can be triggered from the ZRC Target menu. In general for the FOTA feature the communication between the target and the controller uses vendor-specific data frames.

To determine if a newer firmware needs to be downloaded to the controller, the target asks for the current firmware version using the firmware version request command. The target application uses the firmware version provided by the firmware version response frame to determine if a newer firmware is available for the controller. The ZRC Target application does not actually check the firmware version; it only provides the hooks to do so. The vendor data frames are sent from the target node using multi-channel mode. This ensures that the frames can be received when the controller enables its receiver during the active period of the power-save or duty-cycle mode.

The new firmware image is stored into the MCU flash. This is done by the self-programming method. Since self-programming fails if the supply voltage drops below its minimum, verifying the supply voltage is recommended. The supply voltage of the controller can be retrieved by sending the battery status request. The controller answers with the battery status response that contains the supply voltage in millivolts. Since during the firmware download the receiver of the controller is kept on, it is recommended to have a reasonable battery status to avoid that the voltage drops below the supported value for self-programming. See the device datasheet for further information about self-programming.

To accelerate the actual receiving of the new firmware image the controller receiver is enabled. To leave the power save mode, the target sends the Rx Enable request command.

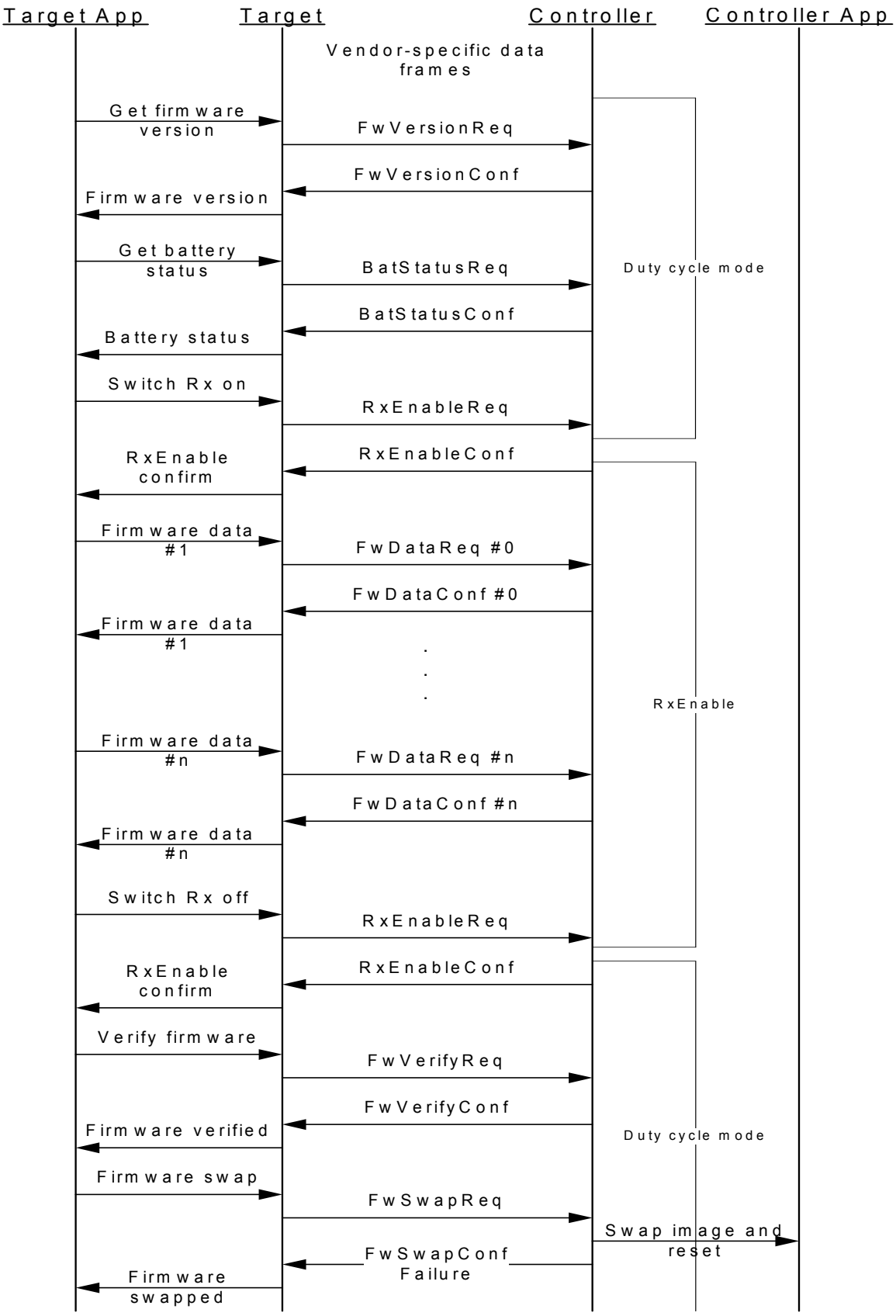
Now, the new firmware can be downloaded from the target to the controller. The target uses the firmware data request frames to send the data packets and the received data packets are stored to the flash of the controller. After that the controller sends a firmware data response command to indicate that it is ready for the next firmware packet.

Once the entire new firmware image is sent from the target to the controller, the target sets the controller to the power-save mode again by sending the Rx Enable request command.

The new firmware image can be verified by using the firmware verify request. If the firmware could be verified by the controller, the controller sends the firmware verify response. Methods (for example, signature check) how to verify that the correct firmware is received is out of scope of the example application. The application examples do not provide this feature.

Finally, the new image can be swapped with the application code. This is triggered from the target by sending the firmware swap request. The controller copies the entire image to the application code area. This requires about five seconds. After this duration the controller needs to be power cycled.

Figure 3-4. FOTA Implementation Example.



4. Quick start guide

4.1 Hardware setup

4.1.1 Tools required

- Atmel AVR Studio® 6.0
- Atmel QTouch Analyzer
- Atmel AVR JTAGICE mkII
- Atmel AVR477 Remote
- Atmel RCB128RFA1 + STB

4.1.2 Remote

- Hardware: AVR477 Remote
- Device: ATmega128RFA1
- Target Supply: 2 × AAA batteries
- Target Clock: Internal RC
- Hex: EXAMPLES_AVR477_REMOTE.hex

4.1.3 Terminal target (RCB128RFA1)

- Hardware: RCB128RFA1 +STB + QT600
- Device: ATmega128RFA1
- Target Supply: USB
- Target Clock: Internal RC
- Hex: EXAMPLES_AVR477_TARGET.hex

4.1.4 QTouch Analyzer

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

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 STB + QT600 board with EXAMPLES_AVR477_TARGET.hex

- Fuse settings on STB + QT600 board should be 0xFE 0x99 0x62
- 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 RCB + STB with the Atmel QT600 as mentioned in Chapter 2
- Wait for the AMBER LED in the STB to blink at least once
- 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 other LED in STB will turn on indicating completion of pairing after a finite delay (within 30 seconds)
- The Atmel QTouch Analyzer will now start connecting with the QT600 and the image of the AVR477 Remote will be displayed
- 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 RCB + STB with QT600 as mentioned in Chapter 2
- Wait for the AMBER LED in the STB to blink at least once
- Keep the remote in vertical position and then press and release the Reset switch located on the rear side of the remote
- The remote should beep once and the other LED in STB will turn on indicating completion of pairing after a finite delay (within 30 seconds)

4.3.2 Touch

- 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

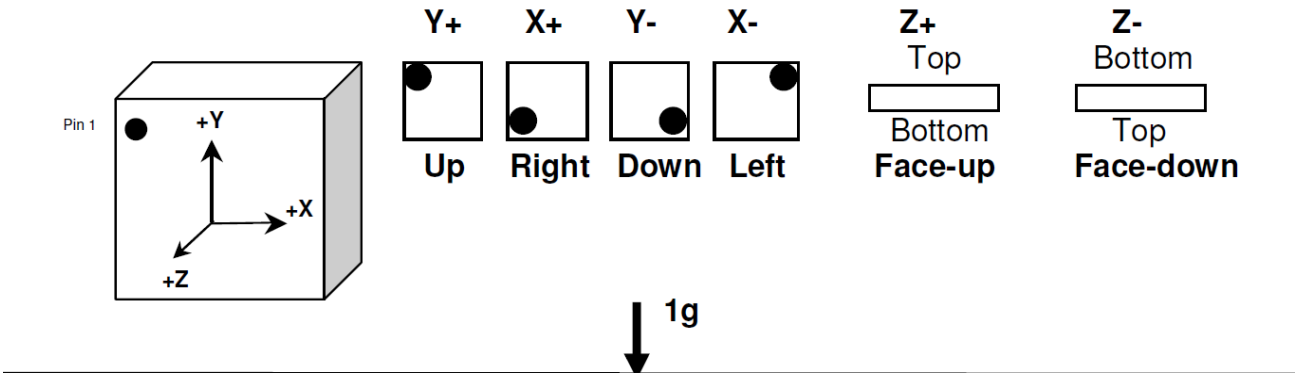
4.3.3 Proximity

- 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 resume and QTouch Analyzer will also start receiving the data

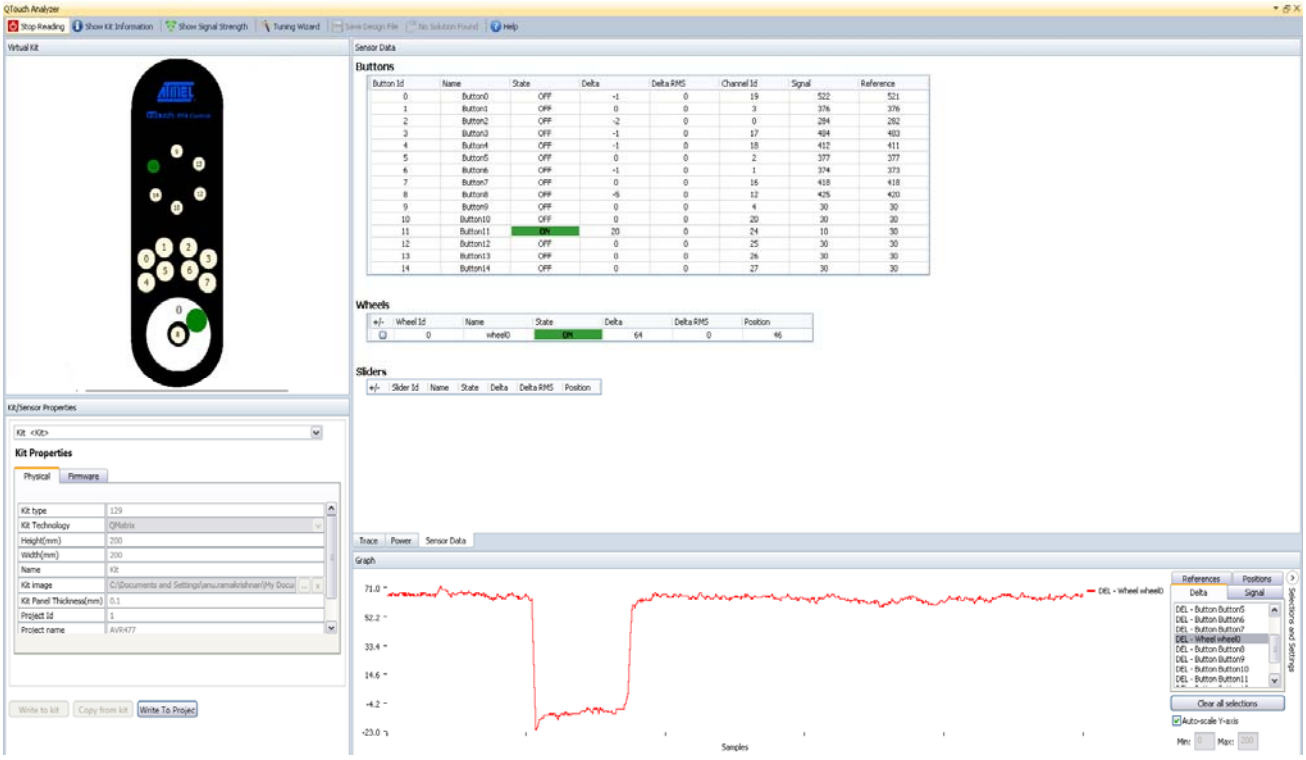
4.3.7 Fault indication

- Both the LEDs will toggle together on fault indication
- Plug out the Atmel QT600 from the USB port to break the communication, this will trigger fault indication
- 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 or not received by the target
- If the fault indication persists (for example, broken link), recovery from this situation would be to warm 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

4.3.9 Atmel QTouch Analyzer demo screen



5. References

1. Atmel AVR2102: RF4Control - User Guide: http://www.atmel.com/Images/doc8357.pdf.
2. Atmel ATmega128RFA1 datasheet: http://www.atmel.com/Images/doc8266.pdf.
3. Atmel QTouch Library User Guide: http://www.atmel.com/Images/doc8207.pdf.
4. Atmel RF4CE-EK kit: http://www.atmel.com/tools/RF4CE-EK.aspx.
5. Atmel QT600 kit: http://www.atmel.com/tools/QT600.aspx.
6. Atmel AVR2068: RF4CE-HID QTouch Analyzer Target for AVR477

Appendix A. Antenna characteristics

- Bend dipole structure
- Matching using a differential stripline
- Calculated to 100Ω at the feedpoint
- ABS radome included in calculation (effect of a housing)

Figure 5-2. Simulation layout with and without ABS radome.

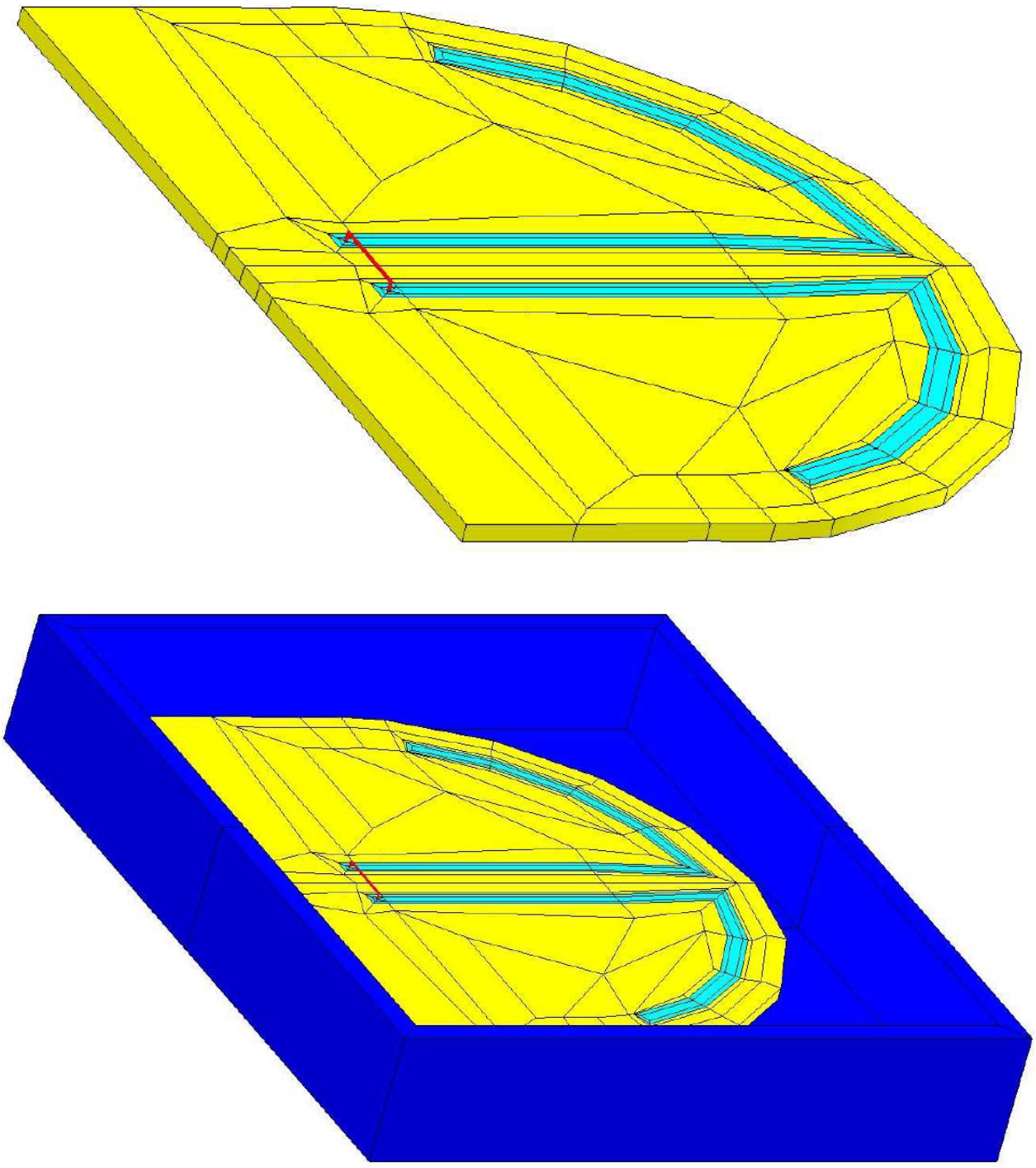


Figure 5-3. Calculated matching with ABS radome against 100Ω.

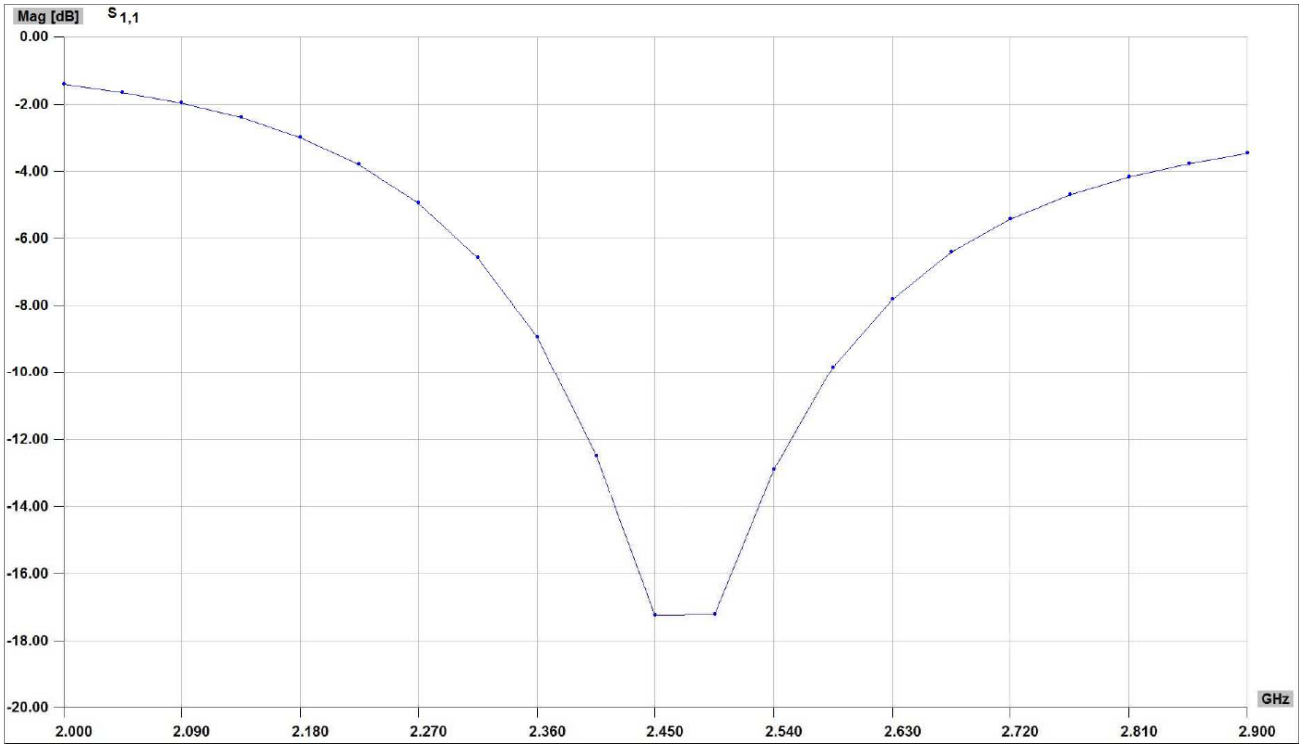


Figure 5-4. Measured matching with ABS radome against 100Ω.

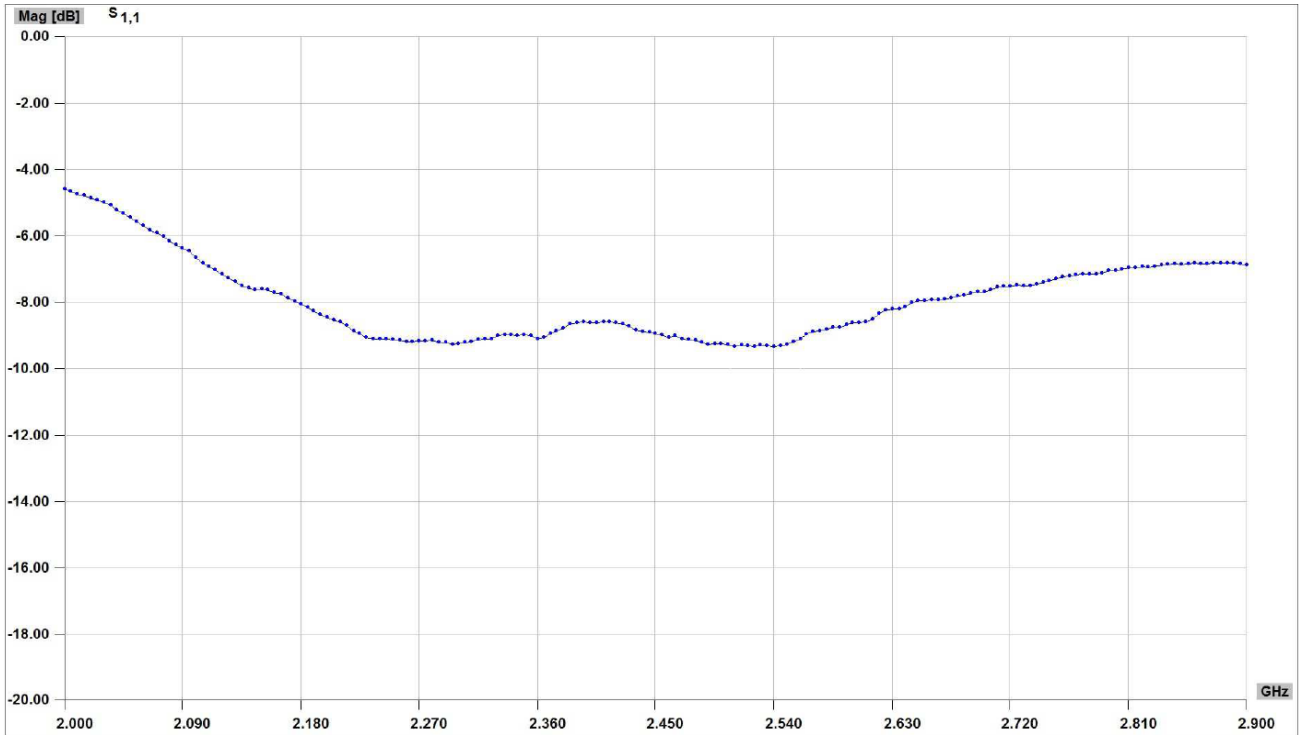


Figure 5-5. Calculated radiation patterns horizontal.

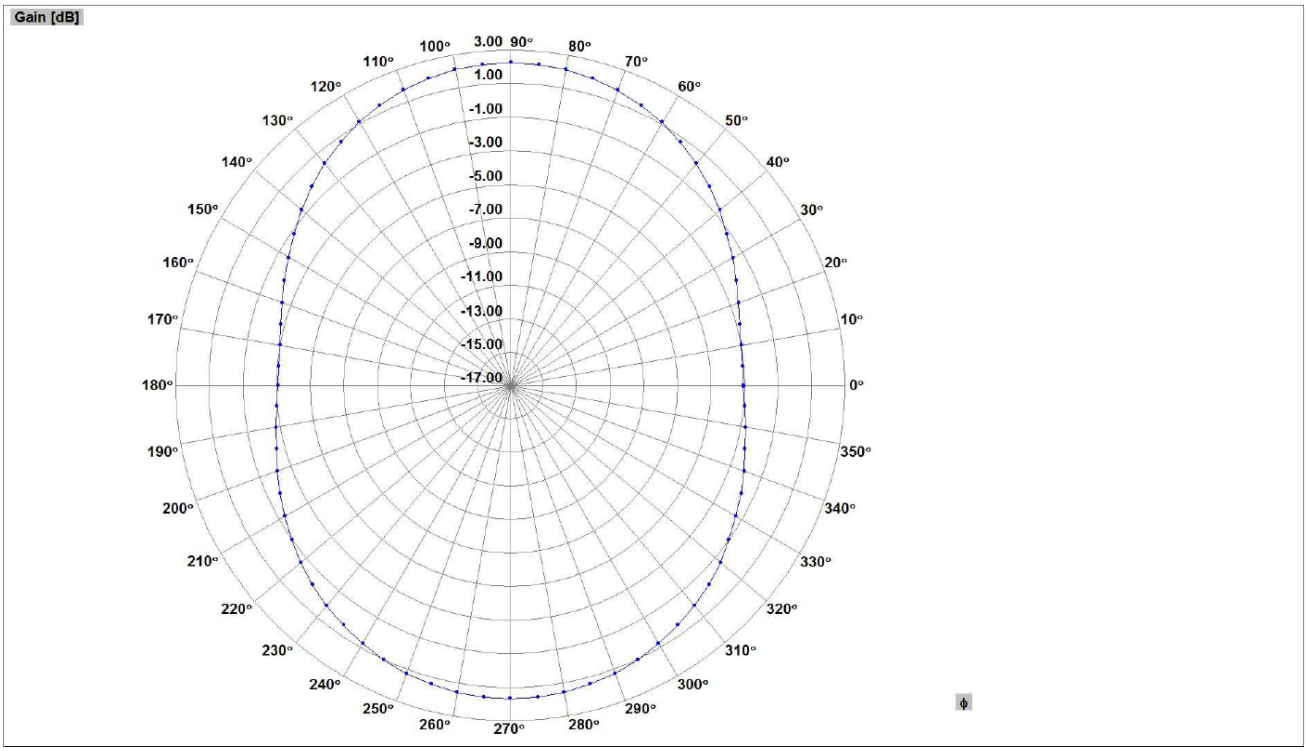


Figure 5-6. Calculated radiation patterns vertical.

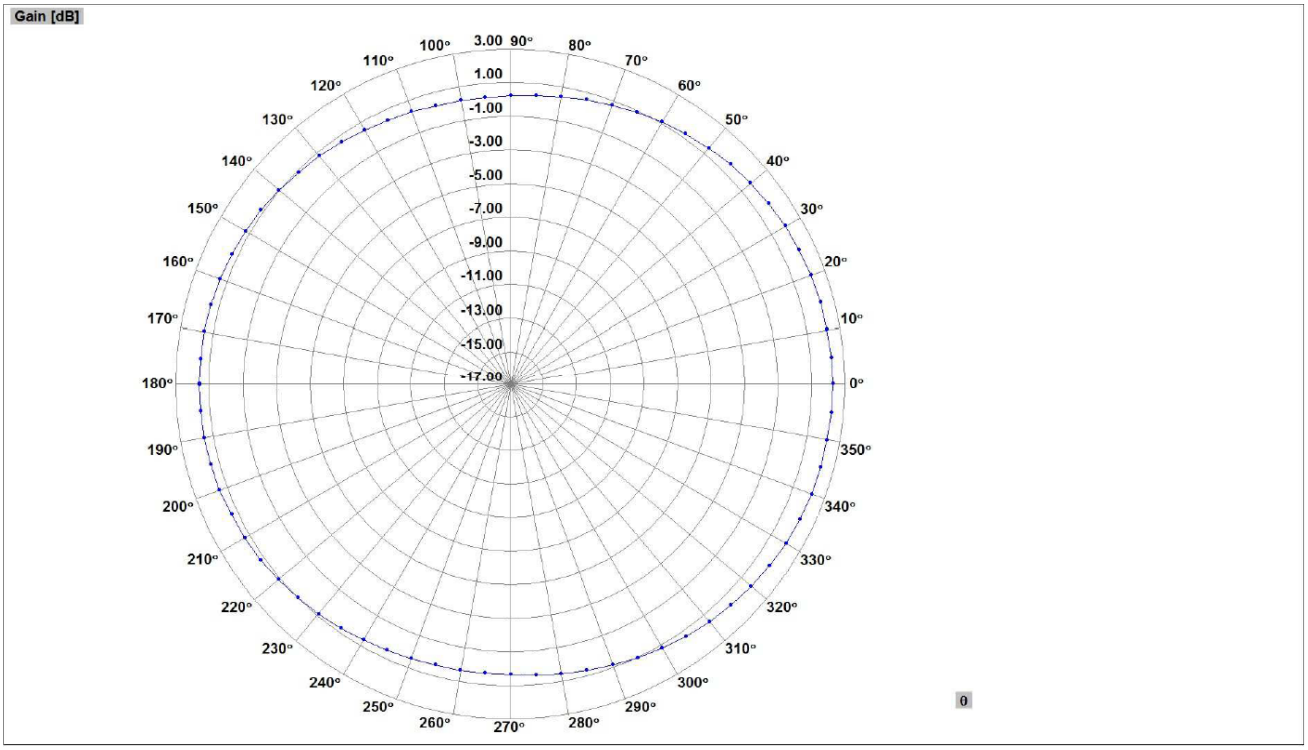


Figure 5-7. Measured radiation patterns horizontal.

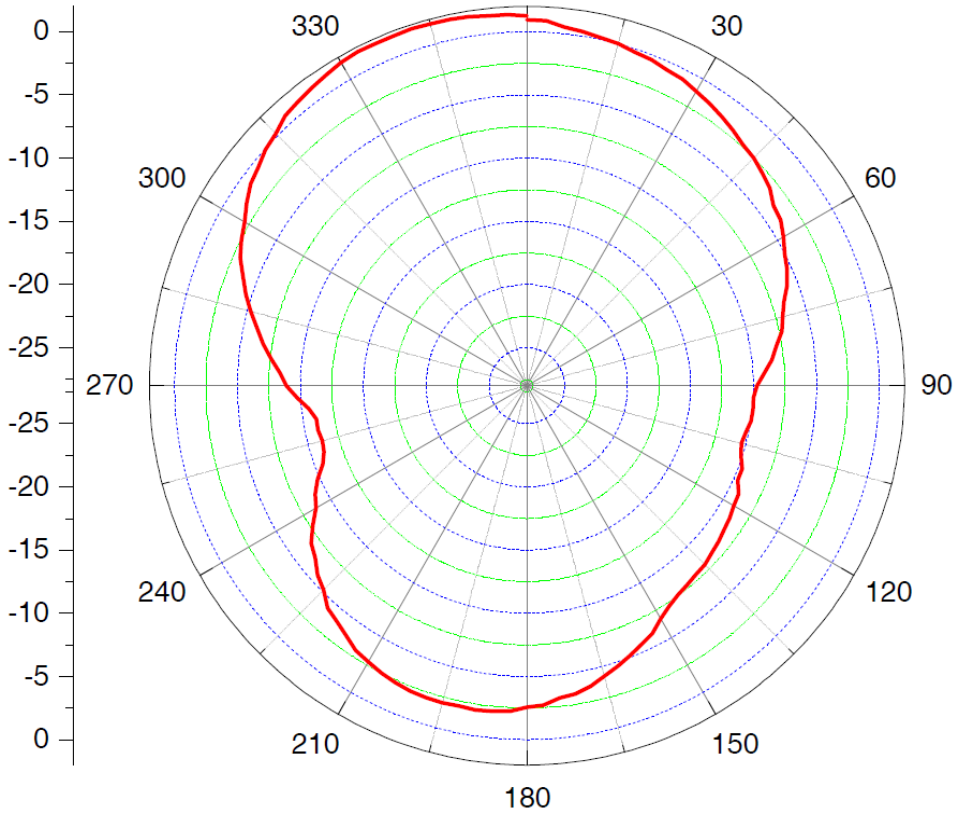
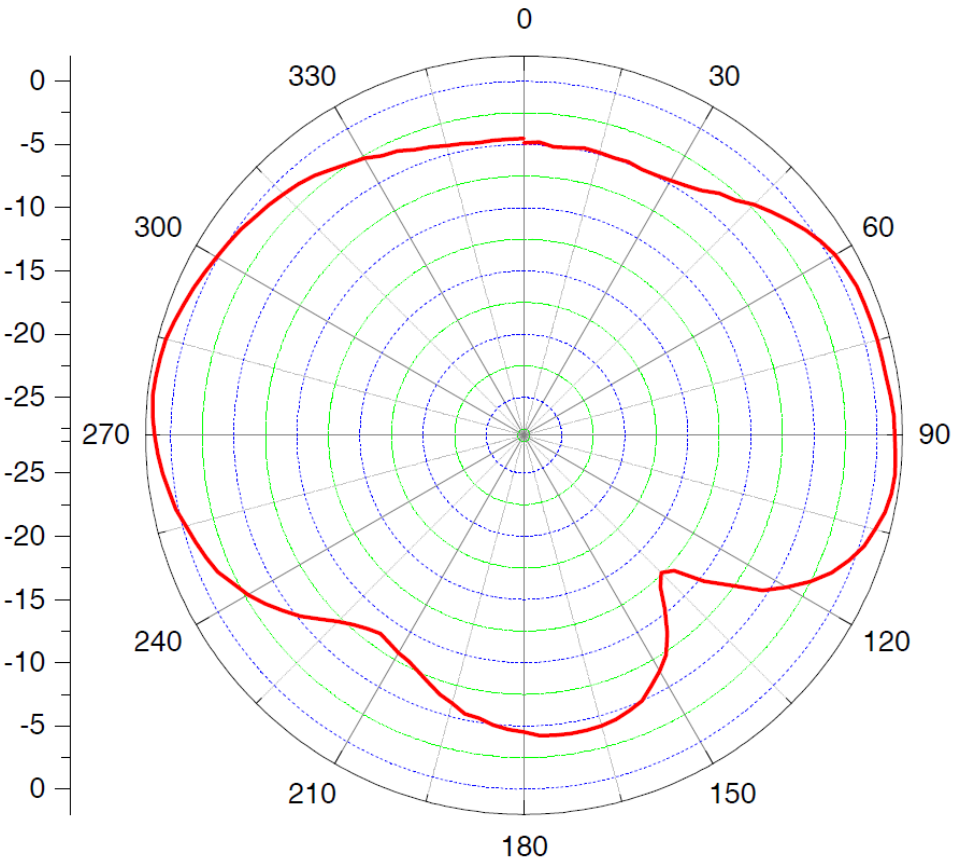


Figure 5-8. Measured radiation patterns vertical.



Appendix B. Revision history

Doc. Rev.	Date	Comments
42020A	12/2012	Initial document release.

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation
1600 Technology Drive
San Jose, CA 95110
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 Bldg.
1-6-4 Osaki, Shinagawa-ku
Tokyo 141-0032
JAPAN
Tel: (+81)(3) 6417-0300
Fax: (+81)(3) 6417-0370