Features

- Atmel® QTouchADC design based on Atmel ATtiny20
  - Optimal hardware design
  - Customized firmware
- Six capacitive touch keys
- Analog voltage output on touch detection
- Buzzer indication on touch detection
- Power management
- TPI programming for ATtiny20
- 5V operation

Description

TV Control Touch Keyboard is designed to be drop-in replacement for an existing electromechanical (resistive ladder) type TV controller. This keyboard outputs respective analog voltage (DC output) on touch detection. The output voltage level is dependent upon the supply voltage to the MCU.

Figure 1. TV Control Touch Keyboard

It is based on Atmel QTouchADC technology. Although it is designed for TV control application, it can be used for any other application which requires up to six keys.
Table of Contents

1. Introduction ........................................................................................................... 3
   1.1 QTouchADC ........................................................................................................ 3
2. Hardware Design .................................................................................................... 4
   2.1 Block Diagram .................................................................................................... 4
   2.2 Buzzer Driver ..................................................................................................... 4
   2.3 Low Pass RC Filter ........................................................................................... 5
   2.4 TPI Programming ............................................................................................. 5
3. Analog Output ......................................................................................................... 6
   3.1 PWM Signal Generation .................................................................................... 6
   3.2 PWM Resolution ............................................................................................... 6
   3.3 D.C Voltage Output ........................................................................................... 7
   3.4 Noise Margin for PWM Output .......................................................................... 7
      3.4.2 Transient Analysis Model ........................................................................... 8
4. Pin Configuration .................................................................................................... 9
   4.1 QTouchADC Sensing ........................................................................................ 9
   4.2 Analog Output ................................................................................................... 9
   4.3 Buzzer ................................................................................................................. 9
5. Firmware ................................................................................................................ 10
   5.1 Application Code Flow ...................................................................................... 10
   5.2 Sensitivity Tuning in Firmware ......................................................................... 11
      5.2.1 Over-sampling ............................................................................................. 11
      5.2.2 Detect Threshold (NTHR) .......................................................................... 11
      5.2.3 Charge Share Delay (CSD) ......................................................................... 11
   5.3 Files in Firmware ............................................................................................... 11
6. Schematic Design ................................................................................................... 12
7. PCB Layout ............................................................................................................. 13
8. Component List ..................................................................................................... 14
9. Recommended Reading ......................................................................................... 14
10. Revision History .................................................................................................. 15
1. Introduction

1.1 QTouchADC

QTouchADC is the newest Atmel sensing algorithm that is implemented by oversampling a standard SAR ADC. The QTouch® devices charge a sense electrode of unknown capacitance to a known potential. The electrode is typically a copper area on a PCB (could be any conductive material, including ITO).

QTouchADC only requires 1-series resistor and 1-pin per channel. No other external components are required. Compared with standard QTouch technology, QTouchADC offers faster acquisition times with shorter burst lengths, resulting in lower power consumption.

The QTouchADC method works by sharing charge between the ADC's internal sample-and-hold capacitor and the sense electrode capacitance. In the initial state, an amount of charge is built-up on an untouched sensor to set the sensor's reference level. As the finger makes contact with the sensor, it couples with the electric field of the sensor. When a sensor is touched the effective capacitance of the sensor electrode increases. This affects the amount of charge shared between the electrodes. Further charge transfers (conversions) from that sensor result in an increased signal level compared to the untouched state. If the difference between the reference level and the signal level is greater than the user-determined threshold (your delta), a touch is detected.

Figure 1-1. QTouchADC Sensing

Detailed technical description of the QTouchADC technology can be found in the application note Atmel AVR3001: QTouchADC Measurement and Tuning.
2. **Hardware Design**

2.1 **Block Diagram**

The keypad consists of the following basic components:

1) Atmel ATtiny20 Microcontroller.
2) Six self capacitance sensor electrodes.
3) PWM based analog output using low pass RC filter.
4) Buzzer for audio indication.
5) 6-pin TPI Programming Header.

![Block Diagram of TV Control Touch Keyboard](image)

2.2 **Buzzer Driver**

The buzzer is driven using a transistor. The switching signal to the transistor is provided from the ATtiny20 device using hardware PWM. The buzzer is driven with 50% duty cycle at 4kHz for 80ms on touch detection.

![Buzzer Drive Circuit](image)
2.3 **Low Pass RC Filter**

Ideally an external DAC should be used to obtain analog signals from digital signal but they are relatively expensive. Another option is to use an R-2R ladder, but it is also difficult to construct and requires a lot of GPIOs.

A simple low cost technique is to use a RC filter to convert PWM signals to DC voltage. The Atmel ATtiny20 microcontroller generates output in PWM mode with varying duty cycle. A passive low pass filter is used to convert the digital PWM to analog voltage.

![RC Filter Circuit](image)

Low-pass filter circuit consists of a resistor in series with a load, and a capacitor in parallel with the load. The capacitor exhibits reactance and block low-frequency signals. At higher frequencies the reactance drops, and the capacitor effectively functions as a short circuit.

2.4 **TPI Programming**

The ATtiny20 supports TPI Programming. TPI is a very tiny programming interface for the newer TINY line of AVR®s with limited pins. Since the small sized tinyAVR®s lacks on-chip debugging circuit, the TPI protocol uses a new programming interface of three pins, in a half-duplex protocol.

![TPI Header](image)

A six pin TPI header is provided for the same. TPI programming can be done using AVRISP mkII and Atmel STK®600. When the device's RSTDSB pin is set only High Voltage Programming can be used to program the chip. Since the RESET line needs to be raised to +12V for programming, this is currently only supported by the newer STK600 programming board.
3. **Analog Output**

The board communicates the touch status to the host controller through ANO pin. The output is RC filtered PWM signal. There is a specific output voltage for each key.

PWM output will be generated during key press. In the idle state, when none of the touch keys are touched, the PWM output will be held high.

3.1 **PWM Signal Generation**

The PWM signal is generated using 16-bit Timer/Counter1 module of the Atmel ATtiny20. The Timer is configured to 16-bit Fast PWM mode. The System clock frequency in this case is 4MHz and clock prescaler has been disabled. The timer period is set to 34 counts for the PWM frequency of 117kHz.

![Figure 3-1. PWM Signal for ON/OFF Key](image)

3.2 **PWM Resolution**

The PWM resolution is dependent on many factors such as

- System clock frequency
- Waveform generation mode used
- Clock prescaler
- TOP value

Higher clock frequency will allow for larger number of clock ticks within the given time period required for a specified frequency in this case 117kHz. The waveform generation mode defines whether 8, 9, 10, or 16-bit PWM resolution can be achieved, the clock prescaler can divide the timer clock by a predetermined scale, and the TOP value determines the end of each counting cycle. The resolution of the PWM output can be calculated as follows:

\[
\text{Resolution} = \left(\frac{1}{34}\right) \times V_{cc}
\]
3.3 D.C Voltage Output

Table 3-1. Analog Output Values

<table>
<thead>
<tr>
<th>KEY</th>
<th>ANALOG OUTPUT (VCC = 5V)</th>
<th>DUTY CYCLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDLE STATE</td>
<td>5.0V</td>
<td>-</td>
</tr>
<tr>
<td>ON/OFF</td>
<td>2.7V</td>
<td>54.38%</td>
</tr>
<tr>
<td>MENU</td>
<td>3.4V</td>
<td>68.58%</td>
</tr>
<tr>
<td>PROG +</td>
<td>1.7V</td>
<td>34.25%</td>
</tr>
<tr>
<td>PROG -</td>
<td>3.9V</td>
<td>77.16%</td>
</tr>
<tr>
<td>VOL +</td>
<td>2.4V</td>
<td>48.56%</td>
</tr>
<tr>
<td>VOL -</td>
<td>4.4V</td>
<td>85.72%</td>
</tr>
<tr>
<td>TV/AV*</td>
<td>1.3V</td>
<td>25.69%</td>
</tr>
</tbody>
</table>

* PROG + and PROG – need to be touched together to activate the TV/AV function.

3.4 Noise Margin for PWM Output

Conversion from digital to analog would carry some ripple voltage in the output. The Low pass filter has to be designed to maintain the tolerable noise margin in the system.

The settings in the hardware and firmware have been done to have a noise margin below 0.02V. The theoretical ripple voltage of the system is 0.015154V.

Figure 3-2. Representative View of Analog Voltage Signal after Filtering

The resistor (R) has been chosen to have a low value of 150Ω so as to minimize the drop in voltage after the filtering effect. Since the PWM frequency is in kHz range, the cut-off frequency is selected in Hz so as to provide significant attenuation to the high frequency signal.

\[ f = \frac{1}{2\pi RC} \]

The C value was calculated using the above formula after selecting 225Hz as the cut-off frequency.
3.4.2 Transient Analysis Model

\[ \text{PWM signal} \rightarrow R = 150 \Omega \rightarrow V_{\text{out}}(s) \rightarrow C = 4.7 \mu F \]

Table 3-2. Transient Analysis of the RC Low Pass Filter

<table>
<thead>
<tr>
<th>TRANSIENT ANALYSIS PARAMETER</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transfer function</td>
<td>( G(s) = \frac{1418.43^71631}{s + 1418.43^71631} )</td>
</tr>
<tr>
<td>Cut-frequency</td>
<td>( f_c = 225.75^169Hz )</td>
</tr>
<tr>
<td>Peak-to-peak ripple voltage</td>
<td>( \Delta V_{pk-pk} = 0.0151^54V )</td>
</tr>
<tr>
<td>Duty cycle = 50%</td>
<td></td>
</tr>
<tr>
<td>Settling time (0% to 90%)</td>
<td>( t_r = 0.0016233 \text{ sec} )</td>
</tr>
</tbody>
</table>
4. Pin Configuration

4.1 QTouchADC Sensing
The touch sensing is done using the internal ADC module of the microcontroller. Thus pins which are connected to the ADC module are connected to the sensor electrodes.

Table 4-1. Pin Configuration of Sense Pins

<table>
<thead>
<tr>
<th>FUNCTIONALITY</th>
<th>MCU PIN #</th>
<th>MCU PIN NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY0</td>
<td>12</td>
<td>PORTA1</td>
</tr>
<tr>
<td>KEY1</td>
<td>11</td>
<td>PORTA2</td>
</tr>
<tr>
<td>KEY2</td>
<td>10</td>
<td>PORTA3</td>
</tr>
<tr>
<td>KEY3</td>
<td>9</td>
<td>PORTA4</td>
</tr>
<tr>
<td>KEY4</td>
<td>8</td>
<td>PORTA5</td>
</tr>
<tr>
<td>KEY5</td>
<td>7</td>
<td>PORTA6</td>
</tr>
</tbody>
</table>

4.2 Analog Output
The analog output is generated by using the PWM module of the microcontroller.

Table 4-2. Pins Configuration for Analog Voltage Output

<table>
<thead>
<tr>
<th>FUNCTIONALITY</th>
<th>MCU PIN #</th>
<th>MCU PIN NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANALOG OUTPUT</td>
<td>5</td>
<td>PORTB2 (OC0A)</td>
</tr>
</tbody>
</table>

4.3 Buzzer
The buzzer is driven using PWM module of the microcontroller.

Table 4-3. Pin Configuration for Buzzer

<table>
<thead>
<tr>
<th>FUNCTIONALITY</th>
<th>MCU PIN #</th>
<th>MCU PIN NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUZZER</td>
<td>6</td>
<td>PORTA7 (OC0B)</td>
</tr>
</tbody>
</table>
5. **Firmware**

The firmware of the TV Control Touch Keyboard has two basic parts. The touch library which performs the capacitive touch measurements and the User Application Code which takes care of PWM generation and buzzer output based on the sensor states. Refer to Section 6.4 of *Atmel QTouch Library 5.0 User Guide* for details of firmware flow and configuration.

5.1 **Application Code Flow**

![Application Code Flow Diagram]

- **START**
- **Initialize System and Touch Library**
  - Timers are configured for PWM
  - ADC is configured for acquisition
  - Touch Parameters are initialized
- **Enable WDT for periodic wake-up**
  - Watch Dog Timer is configured in Interrupt mode to wake up device from sleep
- **Perform Touch Measurements**
  - `qt_measure_sensors();`
- **Is any key in Detect?**
  - YES: **Enable PWM Activate Buzzer**
  - NO: **Disable PWM**
  - **Enable Sleep**
  - **Watch Dog Timer Interrupt**
  - **Put the device in Low Power Mode**
5.2 Sensitivity Tuning in Firmware

In QTouchADC method does not use any external components. All sensitivity tuning parameters are to be adjusted in the firmware using the following parameters. Refer to the Section 2 of Atmel AVR3001: QTouchADC Measurement and Tuning for better understanding of the optimal settings for good sensitivity.

5.2.1 Over-sampling

Increasing the over-sampling increases the sensitivity of the sensors. This parameter is to be modified in the Linker Options in the IAR™ IDE.

/* Default value: 1. Values: 1u, 4u and 16u */
#define DEF_QT_BURST_LENGTH (0x01u)

5.2.2 Detect Threshold (NTHR)

Increasing the detect threshold decreases the sensitivity. This parameter can be modified in the touch_config_tiny20.h file.

/* Sensor detection threshold value. Range: 1u to 255u. */
#define DEF_QT_SENSOR_0_THRESHOLD (6u)
#define DEF_QT_SENSOR_1_THRESHOLD (6u)
#define DEF_QT_SENSOR_2_THRESHOLD (6u)
#define DEF_QT_SENSOR_3_THRESHOLD (6u)
#define DEF_QT_SENSOR_4_THRESHOLD (6u)
#define DEF_QT_SENSOR_5_THRESHOLD (6u)

5.2.3 Charge Share Delay (CSD)

Increasing the charge share delay time allows for increased time for charge transfer between sensor electrode and sampling capacitor. For good sensitivity it is necessary to ensure that full charge transfer takes place. This parameter can be modified in the touch_config_tiny20.h file.

/* Range: 1, 2, 4, 8 or 10 internal System Clock cycles. CAUTION: Do NOT append 'u' or () in this case. */
#define DEF_QT_DELAY_CYCLES 1

5.3 Files in Firmware

The source code contains the following files:

<table>
<thead>
<tr>
<th>FILENAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>libtiny20-6qt-k-0rs.r90</td>
<td>Touch Library file for QTouchADC on Atmel ATtiny20</td>
</tr>
<tr>
<td>touch_api_tiny20.h</td>
<td>Touch Library API file</td>
</tr>
<tr>
<td>touch_config_tiny20.h</td>
<td>Configuration file for setting of tuning parameters</td>
</tr>
<tr>
<td>main.c</td>
<td>User application code</td>
</tr>
</tbody>
</table>
6. Schematic Design

Figure 6-1. Schematic of the Atmel ATtiny20 Based TV Control Touch Keyboard

Place resistors as close to uC.
7. PCB Layout

The design for the TV Control Touch Keyboard follows a dual layer construction. The sensors are on the top layer which is to be covered by a dielectric front panel. The bottom layer has the components and the necessary routing and ground loading. FR4 PCB substrate material is recommended for optimal performance.

Figure 7-1. PCB Layout - Top Layer

![PCB Layout - Top Layer](image1)

Figure 7-2. PCB Render – Top Layer

![PCB Render – Top Layer](image2)

Figure 7-3. Bottom Layer

![Bottom Layer](image3)

Figure 7-4. PCB Render – Bottom Layer

![PCB Render – Bottom Layer](image4)
8. Component List

<table>
<thead>
<tr>
<th>DESIGNATOR</th>
<th>COMPONENT TYPE</th>
<th>COMPONENT VALUE</th>
<th>FUNCTIONALITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1</td>
<td>MCU 14pin-SOIC</td>
<td>ATtiny20</td>
<td></td>
</tr>
<tr>
<td>R1, R2, R3, R4, R5, R6</td>
<td>SMD resistor (0603)</td>
<td>1k</td>
<td>Series resistor for Touch</td>
</tr>
<tr>
<td>R7</td>
<td>SMD resistor (0603)</td>
<td>100E</td>
<td>Buzzer drive resistor</td>
</tr>
<tr>
<td>R8</td>
<td>SMD resistor (0603)</td>
<td>10k</td>
<td>Transistor bias resistor</td>
</tr>
<tr>
<td>R9</td>
<td>SMD resistor (0603)</td>
<td>150E</td>
<td>Low Pass Filter (R)</td>
</tr>
<tr>
<td>C2</td>
<td>SMD capacitor (0603)</td>
<td>4.7μF</td>
<td>Low Pass Filter (C)</td>
</tr>
<tr>
<td>Q1</td>
<td>Transistor</td>
<td>BC847W</td>
<td>Buzzer Driver</td>
</tr>
<tr>
<td>LS1</td>
<td>Piezo Buzzer</td>
<td></td>
<td>Buzzer</td>
</tr>
</tbody>
</table>

9. Recommended Reading

- AVR3001: QTouchADC Measurement and Tuning
- Atmel QTtouch Library 5.0 User Guide
- QTAN0079: Buttons, Sliders and Wheels Touch Sensor Design Guide
10. Revision History

<table>
<thead>
<tr>
<th>Doc. Rev.</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>42208A</td>
<td>10/2013</td>
<td>Initial document release</td>
</tr>
</tbody>
</table>