R01AN1653EG0100
Rev.1.00
Jan 06, 2014

RX210

Low cost solution for capacitive touch control and 7/14 segment LCD control

Introduction

Most hand held or wall mounted instruments require a display and some keys; these often form a large proportion of the cost of the product. The most economical displays are the glass LCD which are driven directly from the microcontroller and thus requires no display controller. The disadvantage is that the number of segments on the display is limited to the number of free I/O port lines available. The lowest cost keys are touch keys made of copper pads on the PCB and therefore have negligible cost. The disadvantage is that each key has to be the size of the finger pad and they are difficult to implement in a matrix format. However, for a small 7 segment or 14 segment displays with few keys, this method of operation would be economical.

This application note describes how to implement touch keys using just I/O ports and how to directly interface with the glass LCD using few I/O port lines. Two methods are described: Half Vcc method and Double Pulse method. The Half Vcc method requires two resistors on the COM line to generate ½Vcc voltage, whilst the double pulse method as the name suggests, requires twice the number of pulses which increases the burden on the microcontroller and consequently uses more power.

Target Device

RX210

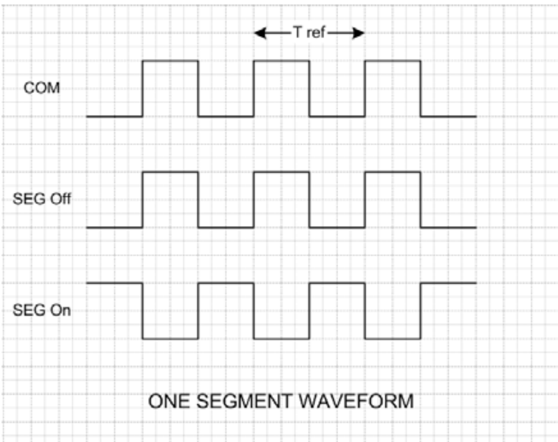
Contents

1. Display

1.1 Glass LCD Drive signals

Consider the simplest LCD display. It has one segment only. To operate the segment, it has to have a backplane, commonly referred to as the COM plane. The voltage is applied between the segment (SEG) and COM to switch ON the segment. To stop the buildup of DC voltage on the segment, the voltages are toggled so the effective DC level is 0, voltage across the segment is constant. The toggle rate is referred to as the refresh rate and it should be between 30 Hz and 75 Hz. The lower frequency may result in flickering, but the higher frequency may generate a 'ghosting' effect where the segment switching off may take longer time to turn off. The higher frequency may also use more power, so it is best to choose the frequency as low as possible, but without the display flickering.

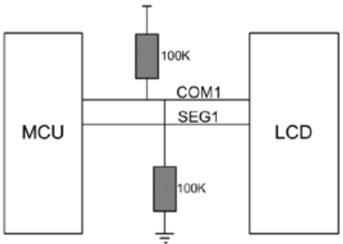
To switch OFF the segment, the same polarity voltage is applied to both the SEG and COM pins, and to switch ON the segment, opposite polarity is applied.

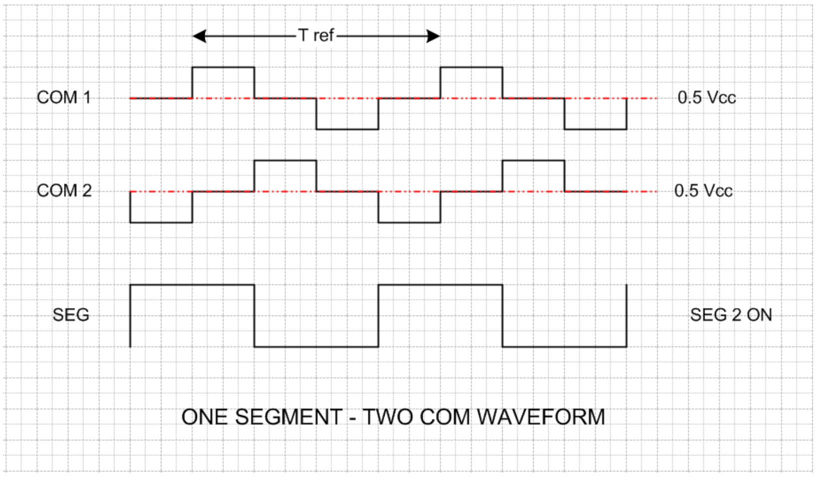


1.2 Half Vcc Method

A two segment display can be arranged in two ways: two SEG pins with one COM pin, or one SEG pin and two COM pins. In the first method, the COM pin is toggled, and the SEG pins are set to opposite polarity to switch on or the same polarity to switch off. This is the simplest method but would require N + 1 pins for N segments. The second method is difficult to implement as there are two back planes, and they have to be operated in multiplex mode. The number of phases depends on the number of back planes. In this example there would be two phases. In phase one, the COM1 pin is set high and SEG pin is either high or low depending if seg1 is ON or OFF. The COM2 pin is held at ½Vcc., the COM1 pin is then held low with the SEG pin following it. In the next phase, the COM1 pin is held at ½Vcc whilst the COM2 pin is toggled with the SEG pin either in phase or out of phase. This method is referred to as the Half VCC method. In this example there is no advantage as both methods require three pins, but when the number of segments is increased, the advantage is apparent. As each pin is driven by I/O port lines, the number of port lines required for N segments is (N/C) + C, where C is number of COM planes. So for 128 segments, the design would require 129 pins in method 1, whilst only 36 pins in method 2 with 4 COM planes.

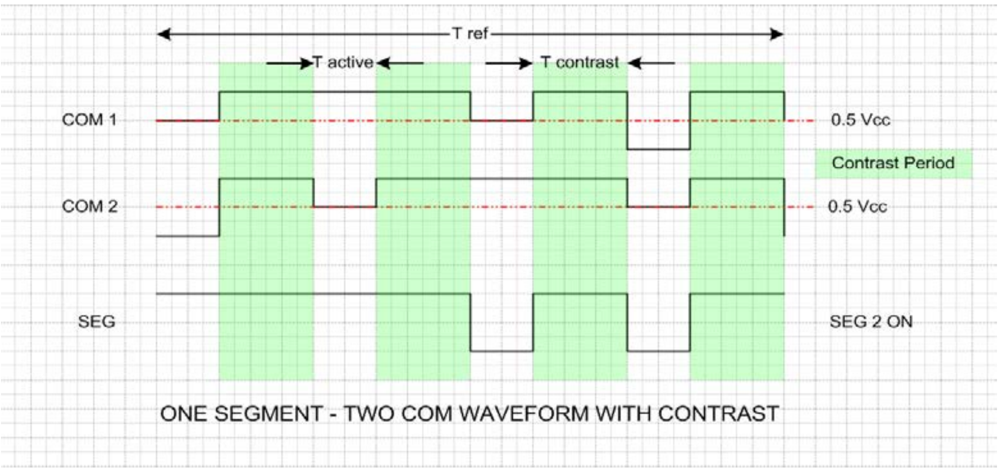
½Vcc voltage is obtained by connecting two equal resistors (100K) on the COM signal and then programming the I/O port as input.





1.3 Contrast

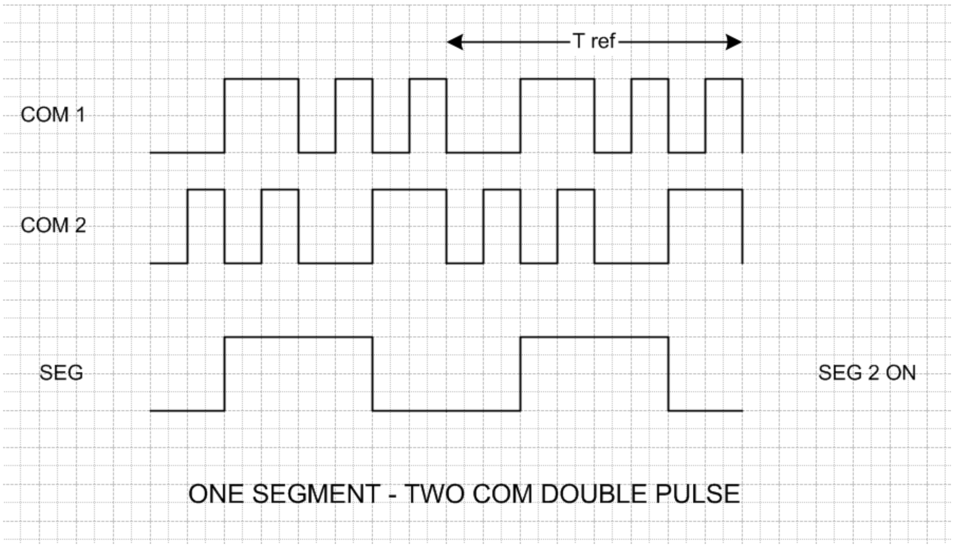
If the above design is actually implemented, most probably the segment would stay ON all the time. The reason is that it has some DC voltage applied from ½Vcc. To lower the DC bias, some delay has to be introduced. This delay acts as a contrast. The delay period can then be varied to increase or decrease the contrast level. The delay period is added after every pulse where both the segment and the COM lines are held at Vcc.



The software has to generate twice the number of pulses for the same period of time and change the time period depending on active or contrast period. The delay period can be achieved with two timers, or as this application shows, with one timer.

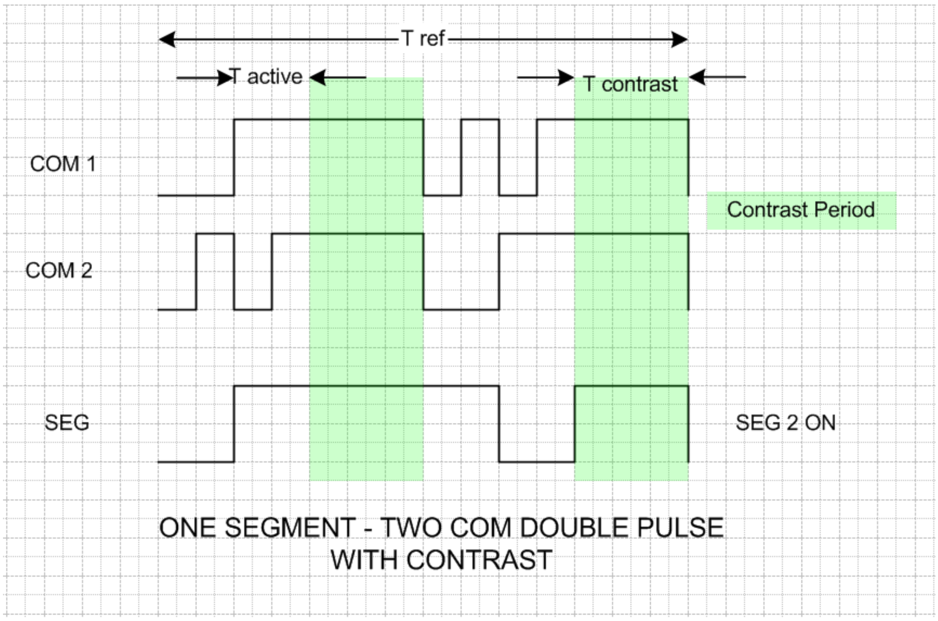
1.4 Double Pulse method

It is not necessary to generate ½Vcc. The double pulse method requires a pair of pulses going low and high to emulate ½Vcc. Whenever there is ½Vcc, replace it with high and low pulses. The advantage is that the resistors are no longer required and the port lines do not have to be set in input mode. The disadvantage is that it requires twice the number of pulses, thus placing a higher demand on the processor and consequently requiring more power. In this application, the order of pulses is different. COM1 goes low and high followed by COM2 going low and high instead of both COM lines going high in turn and then low. Both orders should function normally.



1.5 Double Pulse Method with Contrast

Contrast also needs to be added in this method. However, to reduce the number of pulses, the delay is added after every pair of pulses. Once again, this can be realised using only one timer.



1.6 Quadruple LCD half Vcc method

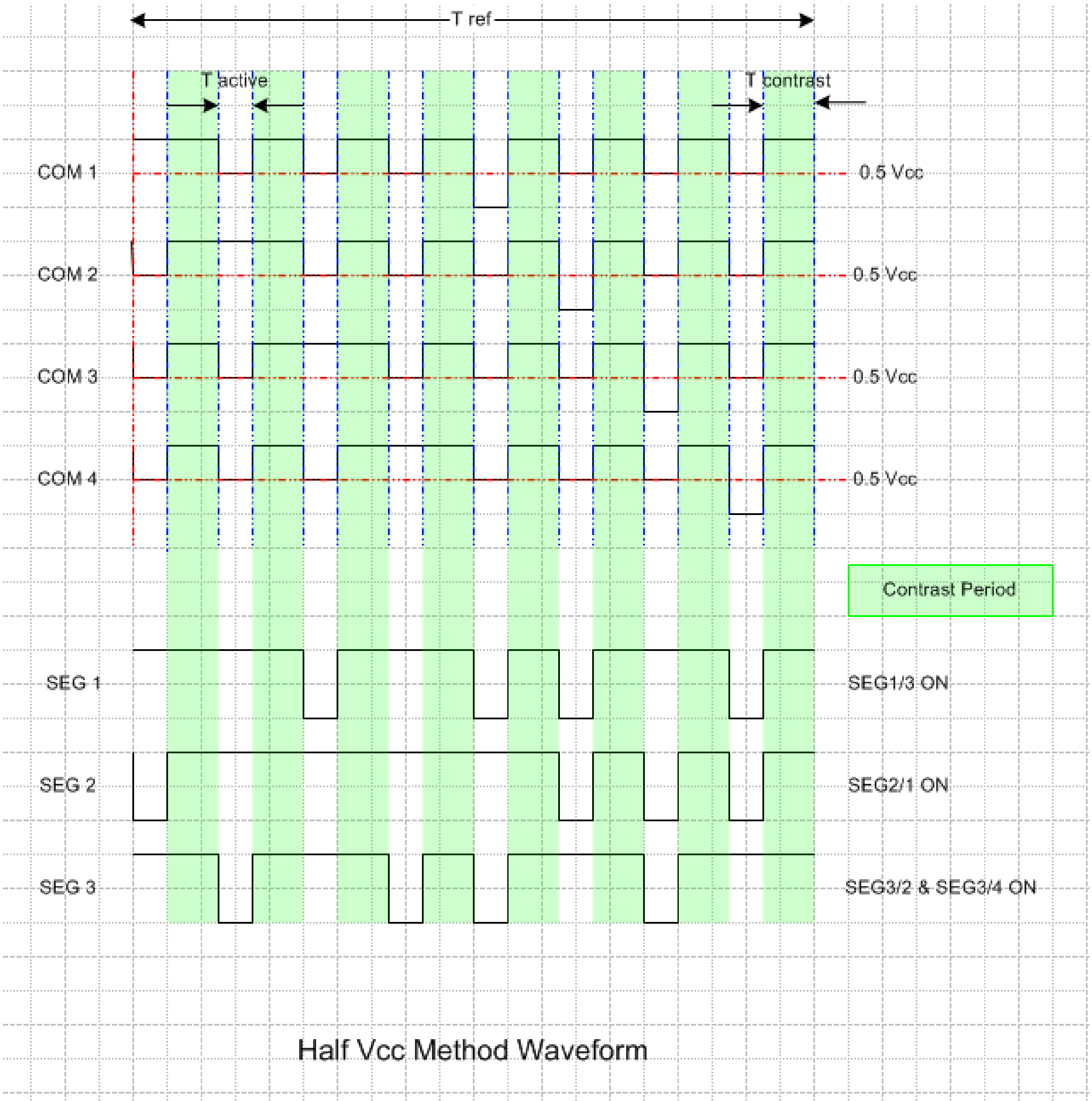
To drive larger displays more COM lines are required. In direct drive multiplex mode, more than four COM lines generate either flickering or ghosting effects. For higher numbers of COM lines, it is better to use a dedicated LCD controller which has the benefit of generating ⅓ Vcc, ½Vcc and ⅔Vcc voltage levels.

In this application, the display consists of 32 segment lines and 4 COM lines. The COM lines are multiplexed at the refresh rate of 30 Hz (T ref = 33.3 mS). Since each active period is followed by a contrast period, the time period for T active + T contrast = 33.3 ÷ 8 = 4.16 ms. Note: When the contrast period is increased, the active period is decreased such that the total time period remains the same. This ensures that the refresh rate is constant.

The contrast period of 33% (1.375 ms) seems to give reasonable result with this display. However, it may vary for different displays.

There are total of 16 states per refresh cycle, each state changes the waveform. Half of the states are for contrast and they can be combined together. For ease of understanding, this application adopts a 16 state loop.

Note: The contrast period cannot be set to 0, as it would be difficult to generate timer interrupts with 0 time period. In this application, the minimum contrast level is set at 10% and maximum level at 80%.



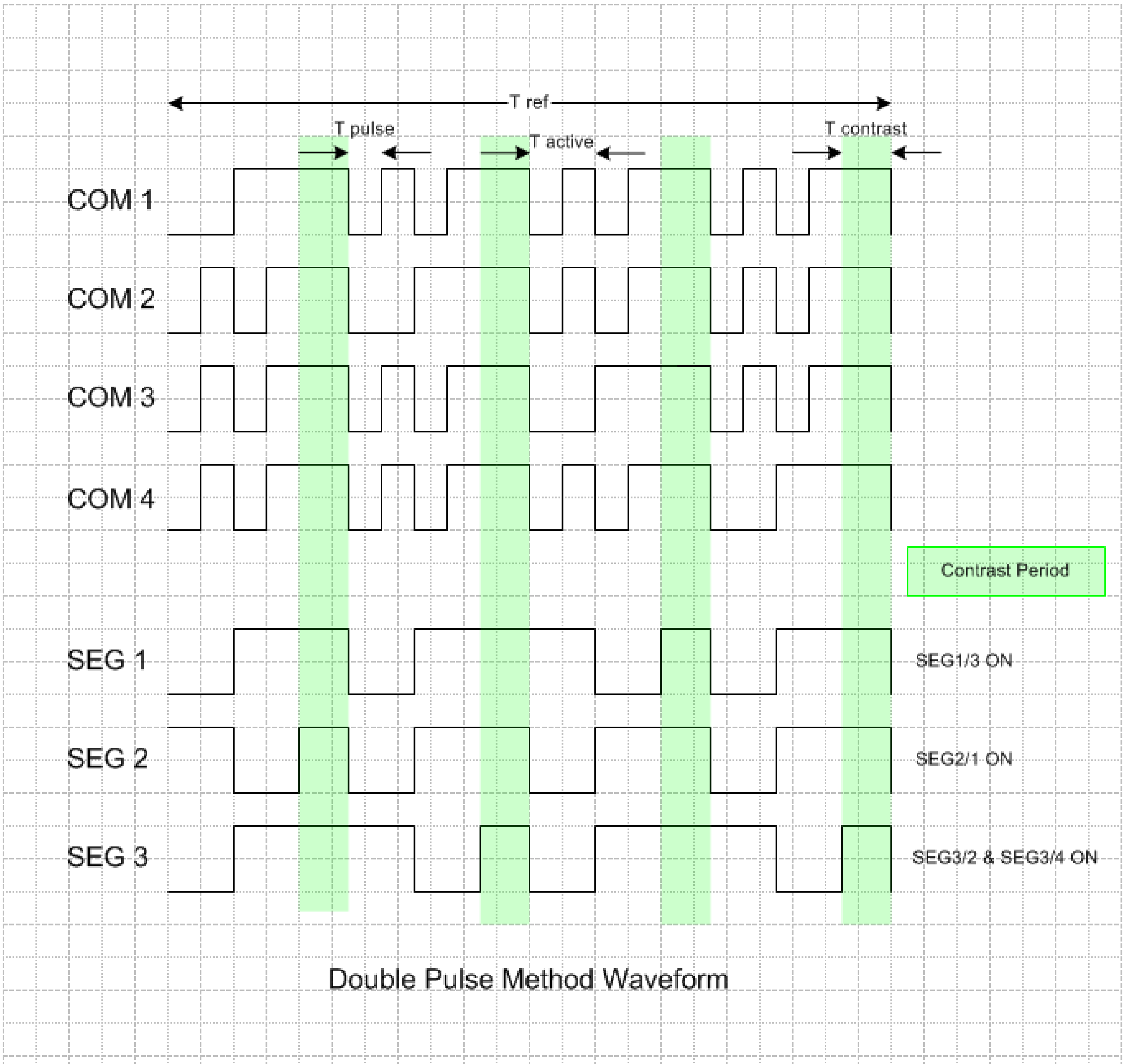
To examine the waveform on an ossciloscope, a trigger pulse has been added. This is available on TP2. A short pulse is generated everytime a new sequence of refresh has started. This is equivalent to the first state in the loop.

1.7 Quadruple LCD double pulse method

As stated in the section 1.5, in this method twice the number of pulses is generated, but there is no ½Vcc level.

The contrast period is added after every pair of pulses, so 2 X T active + T contrast = 33.3 ÷ 4 = 6.66 ms. With this method the best result is obtained when the contrast period is 50% (3.33 ms). For this setting the pulse period T pulse = 3.33 ÷ 4 = 832 us. This is considerable burden on the processor.

As before, the contrast period cannot be set to 0.



1.8 Software Setting

Before building the project, decide which method should be used. In the header file glass_lcd.h, comment out the method NOT required. There are other options regarding the touch keys, but they will be discussed later. Build the software and execute it.

```
/* Comment out the method not required */
#define DISPLAY_METHOD_HALF_VCC
//#define DISPLAY_METHOD_DOUBLE_PULSE
```

The display should show a message 'RENESAS – Press a key'. As there are only 8 characters, the message should scroll at the rate of two characters per second. Turn the pot (RV1) which acts as a contrast pot to set the ideal contrast level. Touch key 1 or 2 and the message will change to "KEY 1|2 PRESSED". The message will revert back to the original message when the finger is removed from the key. Press key 3 and the message will change to "KEY 3 PRESSED". When this key is released, the display will show the contrast value in percentage. Adjusting the contrast pot will change the value. Pressing key 3 will revert back to the original message.

1.9 Software Understanding

There are two main files that control the display. The primary one is glass_lcd.c which has all the display related functions, and the secondary one is adc_repeat.c which continuously reads the A/D value of the pot RV1 setting and adjusts the timer period.

The following analysis assumes that the Half Vcc method has been adopted.

1.9.1 Display Initialisation

Before using the display, it has to be initialised by executing the functions glass_lcd_init() and adc12_repeat_init().

The glass_lcd_init() function sets all the segment port lines as outputs, COM lines as inputs, sets the default contrast period and starts a period timer (CMT timer channel 3). The adc12_repeat_init() function sets the timer channel 1 to generate an interrupt every 1 second.

1.9.2 State Loop

Every time the period timer expires, it calls the glass_lcd_control() function. This function sets the segment and com lines according to the state it is implementing. After every state it sets the timer expiry time to either the active or contrast period. Once all the states are completed, it starts again from the first state. All the 16 states are executed with the T refresh period.

1.9.3 Contrast Period

The A/D timer calls callback_timer_adc12_repeat() every second. It reads the A/D value and converts it into a contrast percentage value. The zero value is equal to MIN_CONTRAST (10%) and the 4095 value is equal to MAX_CONTRAST (80%). It then calls glass_lcd_set_contrast() which calculates the periods based on refresh rate setting.

1.9.4 Refresh Rate

The refresh rate can be altered by changing the value of REFRESH_RATE_HZ in the header file glass_lcd.h. Values below 30 Hz may generate flickering and high values above 70 Hz may display ‘ghosting’. The default setting in this application is 30 Hz as it uses the least power without flickering.

1.9.5 Segment Arrangements

It would be desirable to arrange the segments such that SEG0 – SEG7 are on port A, SEG8 – SEG15 on port B etc. This would allow simple implementations of the patterns for numbers and characters. However, in most designs, the PCB layout plays an important role, and a good design will have short tracks between the port pins and the display. This means the segments are arranged in some random orders with respect to port lines. In this application, the display is nearest to ports PA, PB, PC, PD and PE, so these ports are utilised. The COM pins are nearest to port PC and to make all the PCB tracks flow optimally, PC3, PC3 PC6 and PC5 are used as COM I/O lines.

Each 16 segment digit utilises 4 SEG pins, each controlling 4 segments. This is very convenient, as the same character pattern can be used for all the digits. The patterns based on ASCII characters are specified in the file ascii_pattern.c. Note: Not all ASCII characters can be implemented in 16 segment format.

The table digits[DISPLAY_SIZE] (glass_lcd.c) specifies which port lines are utilised by the display digits. For example, digit 8 uses PD3, PD4,PE0 and PD7. The table port_mask[PORT_NUM] defines which port lines are actually used for segments. For example, Port C uses only PC1, PC2 and PC4 for segment lines.

Once the tables are set, calling glass_lcd_write_char() will write the character patterns in the dynamic variable com_patterns[COM_NUM][PORT_NUM]. This variable holds the segment pattern corresponding to the current message. The ports are set by this pattern every active phase. When the COM line is in the low phase, the pattern is inversed. glass_lcd_write_segment_pattern() performs this function.

1.9.6 Displaying Message

Two additional functions are provided to display a string message (glass_lcd_write_str) or a number (glass_lcd_write_num). This application only uses glass_lcd_write_str(), but the number is provided for the user's benefit.

To clear the display, glass_lcd_clear() is called. This rapidly clears the pattern variables, thus no segments will be switched ON.

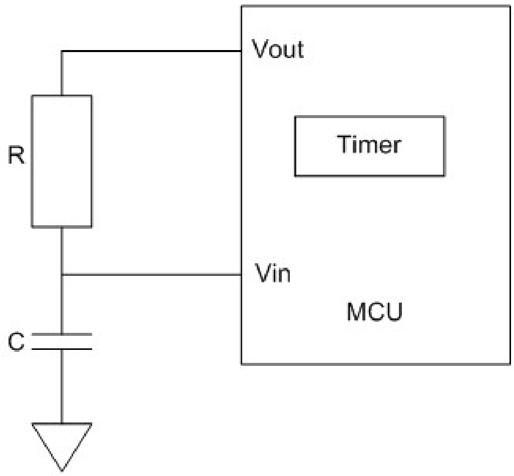
2. Touch Key

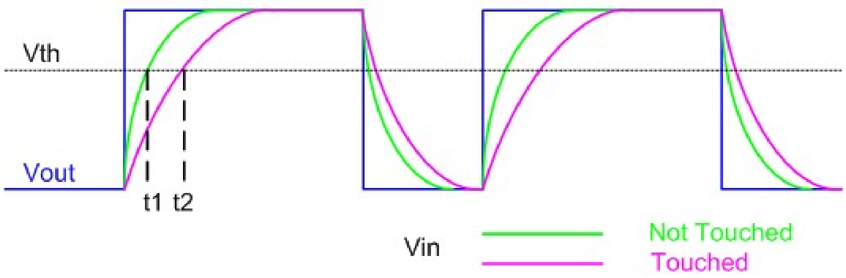
This application demonstrates the implementation of a touch key using an I/O port line. This is by far the cheapest method available to implement a key as it requires only a small PCB pad and two resistors. The aim is to measure the capacitive value of the pad, as the value increases when a finger is placed in its vicinity because the body acts as a small capacitor.

There are various ways of measuring the capacitive value, but by far the simplest method is to measure the rise time or the fall time. This application uses the rise time method to determine the state of the key.

2.1 RC rise time method

As the name suggest, this method relies on the increase in rise time when the key is touched. Vcc is applied to the resistor and the time is measured for it to reach the threshold level. The same is repeated with the key touched. Both the times are recorded for actual key checks. Although the reading of the timer is more accurate, it not practical as each key requires its own timer. In the actual design, only one key is connected to the port with a timer input facility to measure the touched and untouched time period. In this application, all three keys are connected to timer input so that their rise time can be analysed individually.

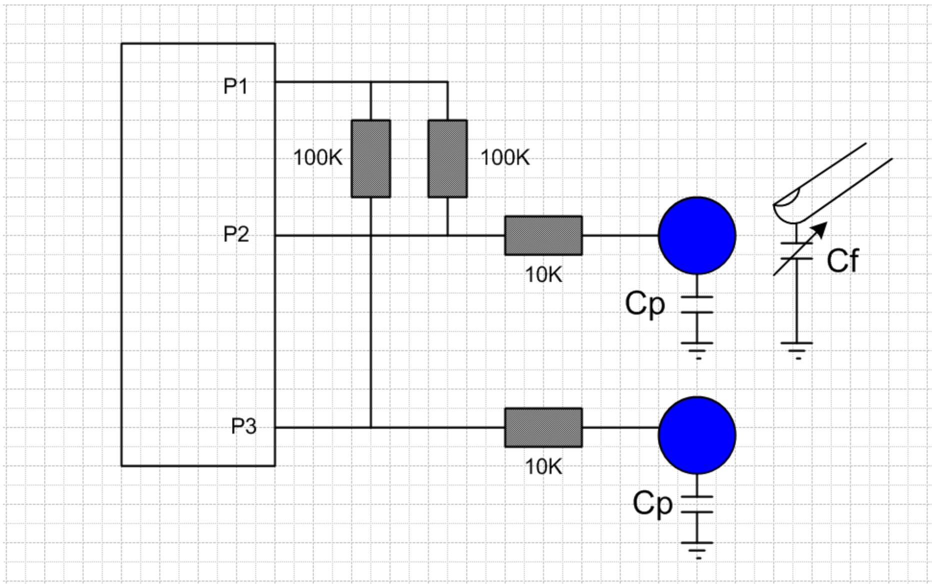




2.2 Reading a Key

Using the timer to read the rise time is only necessary to collect the period information. Once that is achieved, a more practical approach is used to read all the keys simultaneously. The keys are connected as shown below. At least one key port should have the timer input capture facility. Port P1 is set as output and all the key ports (P2, P3) are set as inputs. P1 is normally set low. A time period is selected which falls between the two time periods.

Port P1 is set high (Vcc) and the timer started with the above timer period. When the timer expires, all the key ports are read. If it is high, then the key is not touched, while low indicates it is. To increase the reliability, the keys are read a number of times over a period and the decision is made accordingly. In this application, keys are read individually in series every 1 ms, and if three consecutive samples are positive, then it is considered that the key is touched. Likewise, to ascertain that the key is released, 10 samples have to be negative.



To increase the efficiency of this method, the user can also measure the fall time by setting P1 low. The fall time is shorter and therefore not as reliable as rise time. Preferably, it should not be used on its own but should be complimented with the rise time.

2.3 Touch Key Software Setting

There are two files associated with touch keys: touch_key.c and touch_key_measure.c. File touch_key_measure.c is only required for measuring the rise time period, and is generally omitted in the final product. Before building the project, decide on the course of action. For normal key operation, comment out #define TOUCHKEY_MEASURE in main_touch_lcd.c.

The keys are scanned separately, as the middle key has a different rise time to the two outer keys as it is surrounded by two pads on either side, resulting in higher capacitance. In practice, if a large matrix of keys needs to be scanned, it can be performed in two parts: outer keys and inner keys.

2.4 Touch Key Software Functions

2.4.1 touch_key_init()

This function is called once before the touch keys are utilised. It sets the control port as output and all the keys' ports as inputs. It also resets each keys counter which is then incremented when the key is touched. Timer CMT ch2 is used for scanning the key and timer TMR ch3 is used for key capacitance to decay before the next measurement.

TKEY_CTRL is first set high to start the rise time. CMT ch2 is set with the time specified in the key_period[] array for the required key. When the timer expires, the key input is read and if the value is high, the key is not touched, and if low, it is touched.

TMR ch3 is then started in single shot mode to give 1 ms delay.

Note: Multi key presses are not entertained.

2.4.2 touch_key_in_key()

This function is synonymous to the standard InKey() function. It checks for any key press over a given time period. It returns the pressed key's or NO_KEY if none of the keys are pressed.

2.4.3 touch_key_get()

This function is synonymous to the standard GetKey() function. It waits forever until one of the keys is pressed.

2.4.4 touch_key_release()

This is a useful function to check if all the keys are released. It waits forever until none of the keys are pressed.

2.5 Measuring the rise time

A facility is available in this application to measure the rise time of the keys. Comment out #define TOUCHKEY_DEMO and include #define TOUCHKEY_MEASURE. Rebuild the project and execute it. The display will continuously show the rise time period of key 1. Touch the key, and the rise time will change. Note the readings. Adjust the pot RV1 to set the contrast level if the display is not clear.

Press SW2 on the RX210 board. The display will now show the rise time of key 2. Similarly, press SW3 on the RX210 board and the rise time of key 3 is displayed. Note down the readings and these can be set in touch_keys.c.

```
const uint16_t key_period[NUM_KEYS] = {400,500,350};
```

In the example above, the setting for key 1 is 400 clock pulses, key 2 is 500 clock pulses and key 3 is 350 clock pulses.

Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

Revision History

Rev.	Date	Description	
		Page	Summary
0.10	Nov 7 2012		Draft edition for review
0.20	Feb 8 2013		Review modifications
1.00	Jan 6 2014		First release

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 LanGao Rd., Putuo District, Shanghai, China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 3.0