12 Channel Capacitive Touch Key Controller (over 25 keys)

HX612

Datasheet

Revision 1.3

Table of Content

1. Description

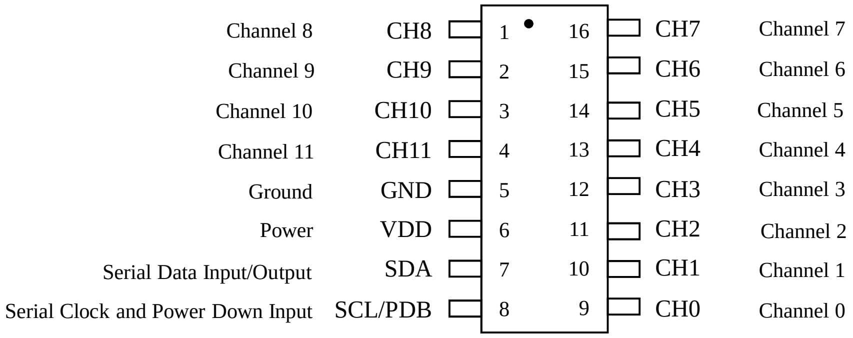
Based on Avia Semiconductor's patented technology, HX612 has 12 input channels, capable of controlling more than 25 keys in a 2 dimensional key array arrangement. It has a large controllable sensitivity/gain range, controlled by internal registers, independent for each input channels. This enables on chip sensitivity tuning for each channel to achieve optimal sensitivity matching of the channels, without any external component.

Patented dynamic analog chopping filter technology, high performance sigma-delta analog-to-digital converter, combined with adaptive digital filtering and the advanced drift tracking algorithms provide the HX612 with superior performance under severe RF and switching power supply interferences, temperature and humidity variations, static charge/discharge, and wet keyboard surface conditions.

2. Features

- 12 input channels, each can be turned on or off independently.
- Programmable sensitivity control for each channel, ensuring sensitivity matching between channels without any external components.
- Self-adaptive calibration; calibration time is less than 7.5mS; response to new touch key event quickly, even when key is touched during power up.
- 3 detection mode: quick touch, slow(long time) touch and single/array mixed mode.
- 3 touch key event interruption options for MCU interruption: falling edge, low voltage or rising edge.
- 2 operation modes: full speed and scan mode. In scan mode, 12-channel key scan time can be programmed to 25~800mS. The switch time between the 2 operation modes can be programmed to 1~7S.
- Low power consumption; current is less than 25uA in 50mS scan mode (@VDD=3.3V, any key wakeup).
- 2 wire serial interface communication: SCL and SDA.
- Operation voltage: 2.9~3.6V.
- Operation temperature: -40~+80℃.
- SOP-16L package.

3. Pin Description



SOP-16L Package

Pin #	Name	Function	Description
9~16, 1~4	CH0~CH11	Analog Input	12 input channels
5	GND	Ground	Ground
6	VDD	Power	Power supply: 2.9 ~ 3.6V
7	SDA	Digital I/O	Serial data input and output
8	SCL/PDB	Digital Input	Serial clock and power down control (low active) input

4. Key Electrical Characteristics

TA=25℃, VDD=3.3V

Parameter	Symbol	Test Condition	MIN	TYP	MAX	UNIT
Operation voltage	VDD		2.9	3.3	3.6	V
Full speed operation current	I_{NORMAL}	12 channels, full speed		1.8		mA
Scan mode operation current	I_{SCAN1}	12 channels, 50mS scanning		25		uA
	I_{SCAN2}	Channel 0 only, 50mS scanning		5		uA
Maximum input capacitance	C_{MAX}			60		pF
Minimum detectable capacitance	C_{MIN}			0.01		pF

5. Functional Description

In HX612, key events are detected by detecting the capacitance changes for each input channel. The capacitance changes are converted to digital by HX612's internal capacitance-to-digital converter. MCU obtains each channel's input capacitance change by reading the corresponding channel's *raw* and *base* data. The capacitance change is *delta*=*raw*-*base*, which represents the signal. *Raw* and *base* are signed 8 bit binary data.

There are various noise and interference sources in the detection system. The detected *raw* data inevitably includes noise and interferences. For the same input capacitance, the *raw* data will fluctuate. The degree of fluctuation is measured by peak-to-peak *raw* noise, or raw_{pp_noise}. The detection system's signal-to-noise ratio can be characterized by the ratio of $delta/raw_{pp\_noise}$. Increasing this ratio will increase the performance of the capacitive touch key controller system. Hence, one of the most important objectives for the system design is to optimize the ratio of $delta/raw_{pp\_noise}$.

The detection method and many register settings in HX612 are related to *raw*, *base*, *delta* and raw_{pp_noise} parameters.

5.1 Serial Interface

5.1.1 Timing Diagram

HX612 uses 2-wire serial interface: the serial clock input SCL; and the serial data input and output SDA. Both SCL and SDA have no internal pull up resisters. Data on the SDA bus is MSB first and LSB last, high represents data 1.

HX612 will read (latch in) the input data from SDA at SCL rising edge. Output data is latched to SDA pin at SCL falling edge. SCL is also used as power down control input. When MCU set the SCL line low for longer than 128uS (full speed mode) or 32mS (scan mode), HX612 will be reset and enter power down mode (less than 1uA current). All internal registers will be reset to the default values.

When SDA goes from high to low (falling edge), while SCL is maintained at high, it indicates the START condition. When SDA goes from low to high (rising edge), while SCL is maintained at high, it indicates the STOP condition. There are two basic serial interface operations: MCU Write and MCU Read.

MCU Write operation includes the following steps: 1) MCU starts the interface by pulling SDA line from high to low (falling edge), while SCL is high; 2) MCU sends 6 bit address ADDR[5:0] to SDA; 3) HX612 sends an ACK signal (high) to SDA; 4) MCU sends 8 bit data DATA[7:0] to SDA; 5) HX612 sends another ACK signal to SDA; 6) MCU sends the stop signal to stop the serial communication; as shown in Figure 1.

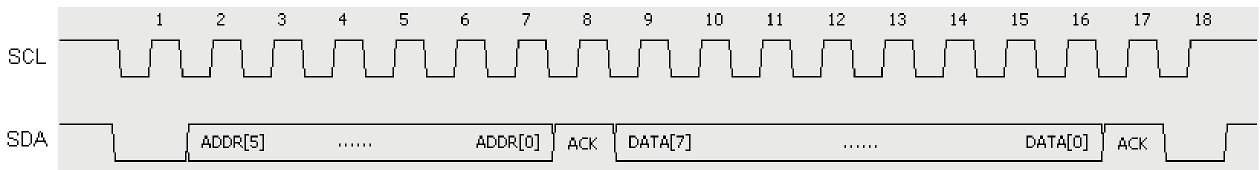


Figure 1 MCU Write Operation

Note: When HX612 is in scan mode, the time between 2 Write operations must be longer than the set value of SCAN_PERIOD register.

MCU Read operation includes the following steps: 1) HX612 starts the interface by pulling SDA line from high to low (falling edge), while SCL is high; 2) MCU sends a read indication signal 1 (high) to SDA; 3) HX612 outputs 16 bit data DATA[15:0] to SDA; 4) HX612 sends the stop signal to stop the serial communication; as shown in Figure 2.

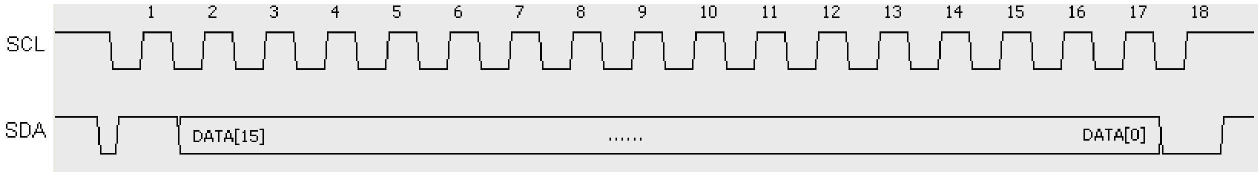


Figure 2 MCU Read Operation

5.1.2 Serial Interface Reset

When MCU sends a Read the Register command to HX612, or HX612 generates a touch key event interrupt, HX612 will pull down SDA line to notify MCU to read data. MCU needs to send SCL clock pulses to read the SDA data within 128uS. Otherwise SDA will go back to high, and the data will be lost.

If the time duration for SCL=1 is longer than 128us, that is, there is no SCL negative pulse for longer than 128uS, the serial interface will be reset. If this was a Write operation, HX612 will ignore this operation. No internal registers will be written.

5.1.3 Initialization and Reset

When HX612 is in full speed mode, if MCU pull down SCL line for longer than 128uS, it will reset HX612. When HX612 is in scan mode, pull down SCL for longer than 32mS will reset the chip. Since HX612 doesn't have power up reset operation, it is be necessary to pull down SCL for longer than 32mS to reset HX612 as initialization operation after MCU power up.

Figure 3 shows the process for MCU initialization of HX612.

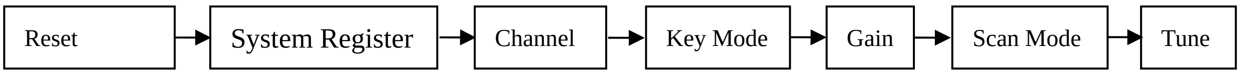


Figure 3 Procedures for HX612 initialization

```
void    Initial_HX612(void)
{
        // Reset HX612
        HX_SCL = 1;
        HX_SCL = 0;
        SDA_OT;
        HX_SDA = 0;
        Delayms(50);
        HX_SCL = 1;
        HX_SDA = 1;

        // System Registers
        HX61x_I2C_Write(0x19,0x08);
        HX61x_I2C_Write(0x1b,0x0a);
        HX61x_I2C_Write(0x1c,0x02);
```

```
HX61x_I2C_Write(0x1d,0x04);
HX61x_I2C_Write(0x1e,0x04);

// Channel On/Off
HX61x_I2C_Write(0x27,0x6f);     // CH8~CH11
HX61x_I2C_Write(0x28,0xff);     // CH0~CH7

// Maximum Key Touch Time
HX61x_I2C_Write(0x1f,0x04);     // [7:0] tm_key: 512ms*t

// Key Mode
HX61x_I2C_Write(0x23,0x00);     // [4]     key_long_en
                                // [3:0] key_long_time: 512ms*(1~15)
HX61x_I2C_Write(0x24,0x00);     // [7]     key_mix_en
                                // [6:4] key_mix_time: 16ms*(1~15)
HX61x_I2C_Write(0x26,0x00);     // Disable Key Interrupt

// Corse Gain
HX61x_I2C_Write(0x29,0x24);     // [5:4] Corse Gain1: 00~11: 2x/4x/6x/8x;
                                // [3:0] Corse Gain2: 0000~1000: 256x/128x/64x/32x/16x/8x/4x/2x/1x
// SENSEn
HX61x_I2C_Write(0x01,60);       //40%~70%*delta
HX61x_I2C_Write(0x02,60);
HX61x_I2C_Write(0x03,40);
HX61x_I2C_Write(0x04,40);
HX61x_I2C_Write(0x05,45);
HX61x_I2C_Write(0x06,45);
HX61x_I2C_Write(0x07,50);
HX61x_I2C_Write(0x08,55);
HX61x_I2C_Write(0x09,50);
HX61x_I2C_Write(0x0a,45);
HX61x_I2C_Write(0x0b,45);
HX61x_I2C_Write(0x0c,40);

HX61x_I2C_Write(0x0d,42);       //70%*SENSEn
HX61x_I2C_Write(0x0e,42);
HX61x_I2C_Write(0x0f,28);
HX61x_I2C_Write(0x10,28);
HX61x_I2C_Write(0x11,32);
HX61x_I2C_Write(0x12,32);
HX61x_I2C_Write(0x13,35);
```

```
    HX61x_I2C_Write(0x14,38);
    HX61x_I2C_Write(0x15,35);
    HX61x_I2C_Write(0x16,32);
    HX61x_I2C_Write(0x17,32);
    HX61x_I2C_Write(0x18,28);

    // Scan Mode
    HX61x_I2C_Write(0x20,0x60);   // 3s + 25ms
    HX61x_I2C_Write(0x21,40);     // 1000ms/25ms=40

    // Tune
    HX61x_I2C_Write(0x2a,0x01);   // [0] tune
}
```

5.2 Registers

Address	Register name	Bit	Default	Functions/Required Settings
01H~0CH	SENSEn	[7:0]	45	Channel 0~11 Sensitivity Register, n represent channel number, suggest setting: 40~80
0DH~18H	*System Parameter*	*[7:0]*	*40*	*Set to: 0.7* SENSEn*
19H	*System Parameter*	*[7:0]*	*4*	*Set to: 8*
1BH	*System Parameter*	*[7:0]*	*4*	*Set to: 10*
1CH	*System Parameter*	*[4:0]*	*4*	*Set to: 2*
1DH	*System Parameter*	*[4:0]*	*16*	*Set to: 4*
1EH	*System Parameter*	*[4:0]*	*8*	*Set to: 4*
1FH	MAX_TIME_KEY	[4:0]	8	Maximum key touch time 00001~11111：512ms*(1~31)
20H	SCAN_FULL_TIME	[7:5]	3	Scan mode full speed work time 001~111：1024ms*(1~7)
	System Parameter	*[4:3]*	*2'b00*	*Set to: 0*
	SCAN_PERIOD	[2:0]	3'b111	Scan mode scan period 000~101:25ms/50ms/100ms/200ms/400ms/800ms; 111:full-speed
21H	SCAN_RENEW_BASE_CNT	[6:0]	8	Set to: 1000~2000ms/scan period
23H	KEY_LONG_EN	[4]	1	Long time key press enable 0：Off；1：On
	KEY_LONG_TIME	[3:0]	4	Long time key press time 0001~1111：512ms*(1~15)
24H	KEY_MIX_EN	[7]	1	Mixed key function enable 0：Off；1：On
	KEY_MIX_TIME	[6:4]	4	Mixed key function monitor time 001~111：16ms*(1~7)

	System Parameter	*[3:0]*	*4'b1100*	*Set to: 0*
26H	*System Parameter*	*[7:4]*	*4'b0000*	*Set to: 0*
	KEY_INT_EN	[3:0]	4'b1000	Key interrupt enable 1000: KEY_QUICK; 0100: KEY_LONG; 0010: KEY_MIXED; 0000: No interrupt ◆When interrupt，SDA will output the data selected by READ_SEL register.
27H	*System Parameter*	*[6:4]*	*3'b110*	*Set to: 6*
	CHANNEL_H_EN	[3:0]	4'b1111	Channel 11~8 enable 0：Off；1：On
28H	CHANNEL_L_EN	[7:0]	8'b11111111	Channel 7~0 enable (turn off ch0 last) 0：Off；1：On
29H	GAIN_ COARSE1	[5:4]	2'b00	Coarse Gain1 00~11：2x/4x/6x/8x
	GAIN_COARSE2	[3:0]	4'b1111	Coarse Gain2 0000~1xxx: 256x/128x/64x/32x/16x/8x/4x/2x/1x
2AH	*System Parameter*	*[1]*	*0*	*Set to: 0*
	TUNE	[0]	0	External calibration enable 1: enable calibration once; 0: disable
2BH	READ_SEL	[5:0]	6'b111111	MCU to read 16-bit data { MSBs, LSBs }: **10_0001**: { KEYSUM[3:0], KEY_MIXED[11:0] } **10_0010**: { KEYSUM[3:0], KEY_LONG[11:0] } **10_0011**: { KEYSUM[3:0], KEY_QUICK[11:0] } **01_0000~01_1011**: { *base*[7:0], *raw*[7:0] } of channels 0~11, read these data during system development or debug. **Note:** KEYSUM[3:0] is the sum of the numbers of 1's in KEY_MIXED[11:0] or KEY_LONG[11:0] or KEY_QUICK[11:0], indicating how many channels have detected with finger touch.

Note: All *System Parameters* in the table should be set to the required settings.

5.2.1 Gain and Sensitivity Registers

GAIN_COARSE1 and GAIN_COARSE2 are gain (has the same effects as sensitivity registers) for all 12 channels. SENSEn is sensitivity register for each channel, n is 0~11 representing channel numbers. By changing GAIN_COARSE1 and GAIN_COARSE2 settings, the *delta* value (*delta*=*raw*-*base*), which represents the capacitance change due to finger touch, for all channels can be changed for the same finger touch, in a given system.

To set GAIN_COARSE1 and GAIN_COARSE2 registers, set GAIN_COARSE2 to the minimum first and increasing GAIN_COARSE1. If GAIN_COARSE1 reaches the maximum and the *delta* value is still not large enough (recommended *delta* value is more than 60) for any channel, increase GAIN_COARSE2 by 1. Then, increasing GAIN_COARSE1 and check the *delta* values of all channels again. Repeat the above until the *delta* values for all channels are more than the desired value, for example, 60. SENSEn value should be set to 40%~70% of the *delta* value for each individual channel. If this percentage is too low, it will increase the system's sensitivity to noise and interferences. Increasing this percentage will reduce sensitivity. It's recommended to set SENSEn between 40 to 80. For example, if *delta0* to *delta11* are 120, 120, 80, 80, 90, 90, 100, 110, 100, 90, 90 and 80, then SENSE0 to SENSE11 should be set to 60, 60, 40, 40, 45, 45, 50, 55, 50, 45, 45 and 40 when choose 50% percentage.

5.2.2 Maximum Key Touch Time Register

Register MAX_TIME_KEY is the maximum key touch time register. When a channel detects a key event and it lasted for longer than MAX_TIME_KEY, HX612 will suppress the key press event and track/update the *base* register.

5.2.3 Detection Mode Register

◆ **Quick Touch Mode**

This is the normal operation mode. The channels detected with finger touch is set to 1 in register KEY_QUICK[11:0].

◆ **Slow (Long Touch Time) Mode**

KEY_LONG_EN and KEY_LONG_TIME are slow-mode control registers. When a key is pressed for longer time than KEY_LONG_TIME, but less than MAX_TIME_KEY, it will be considered as a valid key event. The channels detected with finger touch is set to 1 in register KEY_LONG[11:0].

◆ **Mixed Mode**

KEY_MIX_EN and KEY_MIX_TIME are mixed-mode key control registers. Mixed mode is suitable when a channel is used to control a single key and 2 channel controlled keys simultaneously. When a valid key event is detected in one channel, and another valid key event is detected in another channel within the time of KEY_MIX_TIME, it indicates that a key press event occurred in one of the 2 channel controlled keys, not the single channel controlled key. On the other hand, if only one channel has valid key event for longer time than KEY_MIX_TIME, it indicates a single channel controlled key event has occurred. The channels detected with finger touch is set to 1 in register KEY_MIX[11:0].

5.2.4 Key Interruption Register

There are 3 key interruption sources in HX612, corresponding to the 3 key detection modes. When key interruption is enabled, and a valid key press is detected, HX612 will pull down SDA line (set it to 0) to notify MCU. If MCU uses inquiry mode (KEY_INT_EN=0) to read channel registers, the SDA line will not be set to 0 when HX612 has valid key press event detection.

When there is a key press event interruption, the SDA line will output the corresponding register value determined by READ_SEL register. READ_SEL[1:0] default value is 11, corresponds to KEY_QUICK register. If MCU selects the slow mode (long touch time key) as the key event interruption source, and wants to read register KEY_LONG, MCU needs to set KEY_INT_EN=0100 and send read register command

HX612_I2C_Write(0x2b,0x32) to HX612 once to set READ_SEL[1:0]=10.

HX612 generates interruption by pulling down the SDA line for 128uS. If MCU doesn't send any SCL clock pulses to read data during the 128uS period, the SDA line will go back to high. To respond to HX612 interrupt, MCU can use SDA's falling edge, low voltage or rising edge as interruption.

5.2.5 Scan Mode Register

SCAN_FULL_TIME and SCAN_PERIOD are scan mode control registers. In scan mode, HX612 is in sleep status most of the time. It will check the channel status periodically in every SCAN_PERIOD time. When it detects any key event, it will return to full speed mode. In full speed mode, if it detects no key event for longer than SCAN_FULL_TIME time, it will return back to scan mode.

SCAN_RENEW_BASE_CNT is another scan mode control register. When HX612 stays in scan mode for longer than SCAN_RENEW_BASE_CNT* SCAN_PERIOD, the *base* will be updated. It is recommended to set SCAN_RENEW_BASE_CNT*SCAN_PERIOD between 1 to 2 seconds. For example, SCAN_PERIOD=25ms, SCAN_RENEW_BASE_CNT=40, and SCAN_RENEW_BASE_CNT*SCAN_PERIOD=1s.

5.2.6 Calibration Control Register

TUNE is a register for external calibration control. When MCU set TUNE=1, HX612 will execute calibration once. After calibration, TUNE will be reset to 0 automatically. It's suggested to run at least one calibration after MCU power up. Calibration time is 7.5mS.

5.3 Other Considerations for MCU Controller

5.3.1 MCU Enters Power Down Mode

When HX612 is in scan mode, it takes a maximum time of T_{SCAN_PERIOD} to respond to MCU, as shown in Figure 5.

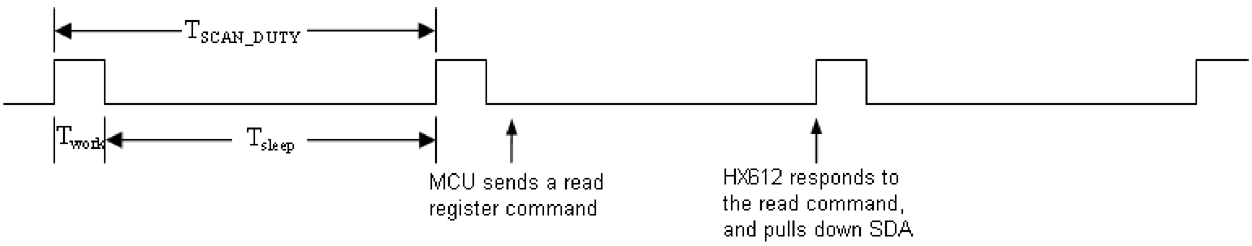


Figure 5 Response to MCU timing in scan mode

When MCU uses inquiry to communicate with HX612 and sends a Read Key Register command after HX612 enters the T_{sleep} period, HX612 serial interface can receive the read command from MCU, but it needs to wait until the next T_{work} period to pull down SDA line. Since MCU does not detect the SDA line going low after it sends the Read Key Register command, MCU many enter power down mode at this moment. When HX612 enters next T_{work} period, it will respond to the last MCU instruction and pull down SDA line to wake up MCU. This may cause MCU to wake up right after entering power down mode. To solve this problem, it's recommended that after MCU sends each command to HX612, it should wait for longer than

T_{SCAN_PERIOD}+128uS before MCU enters power down mode.

MCU should set SDA line to quasi-bi-directional mode or high resistance input mode before MCU enters power down mode.

6. Suggestions for PCB Layout

Touch key PCB keyboard is an integral part of the capacitive touch key system. All signals on the touch key PCB keyboard are small signals. Proper design of the PCB layout will critically affect the performance of the whole system.

6.1 Routings for Touch Keys

The routings from HX612 input channel pins to each touch key are necessary, but not the desirable part of the touch key. It's the parasitic of the touch key. Minimizing its effect is very important.

The routing lines should not be put on the same side as the touch pad. The PCB side of the touch pad area should only include the touch pads, surrounded by ground with more than 40mils spacing between pad and ground.

6.1.1 Routing Length and Width

Minimizing all channels' routing length and width is a good practice. Routing lengths for different channels are not required to be equal. Routing differences among the channels will increase sensitivity differences of the channels. The differences can be adjusted to the minimum through internal sensitivity/gain control registers. However, designing the routings for each channel to be similar in length and width will help reduce interferences from ground and common mode sources.

6.1.2 Routing Directions

Routing line spacing should be wide. Suggested routing line spacing is 20mils.

Routing line should avoid going near or under other touch pad, as shown in Figure 6.

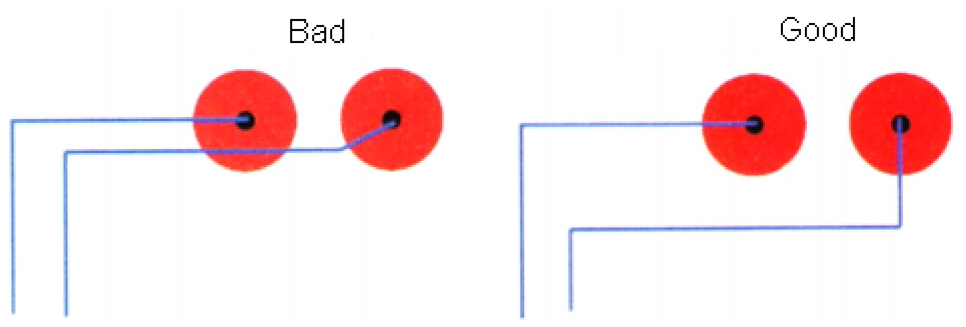


Figure 6 Routing for touch pad

Routing lines are small signal lines with high impedance levels. Hence, they are sensitive to interferences and noise coupling. They should not go near other large signal, large current and high frequency communication lines. These include power lines, especially switching power lines, I^2C and SPI serial communication lines. These lines should not be put on the touch key pad side of the PCB, and not running parallel to the touch key routing lines, as shown in Figure 7.

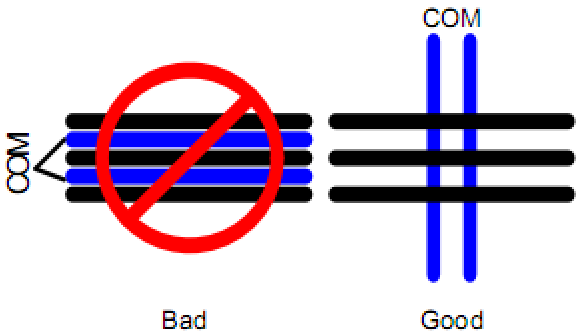


Figure 7.1 Routing on different PCB layer

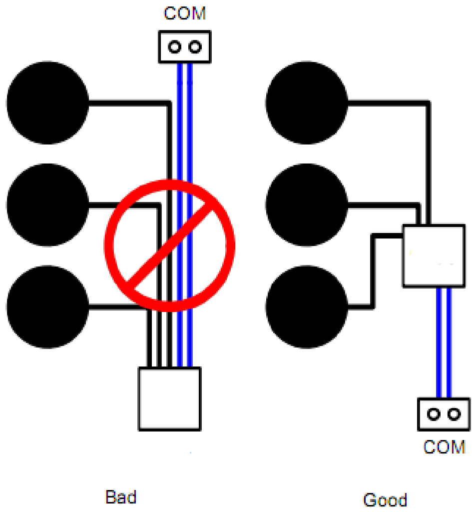


Figure 7.2 Separate them if on the same layer

6.2 Space Between Touch Key and Ground

The touch keys should be surrounded by Ground to reduce undesirable interferences. But the ground shielding should not be too close to the touch key in order to reduce the parasitic capacitance between channel inputs to ground. It's suggested to keep more than 40mils spacing.

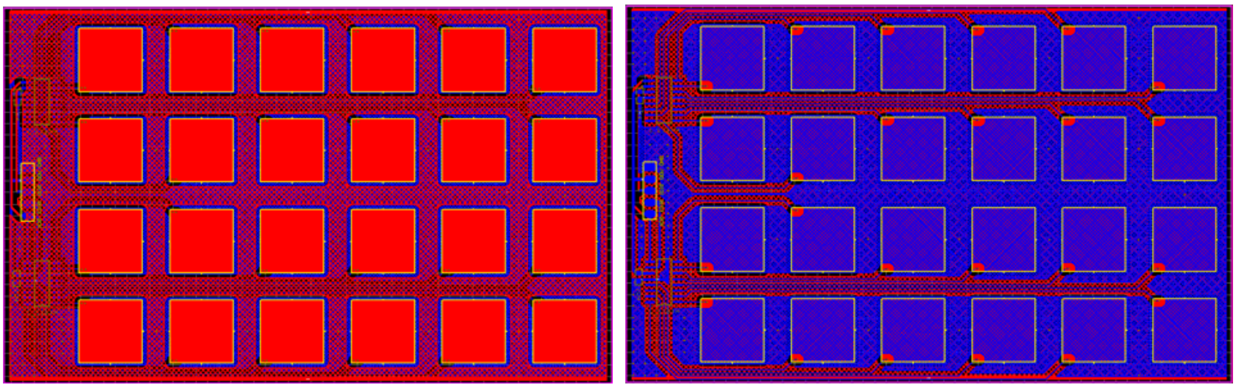
6.3 Space Between Touch Keys

When touch keys are very close to each other, one channel's touch key event can affect other channel's *delta* value significantly, possibly making false key event decision. Using the neighbor key suppression function will effectively eliminate the problem. But when the key sensitivities are set very high and the dielectric media on top of the touch key is relatively thick, there are still chances for making false decisions. It's suggested to keep this spacing for more than 40mils.

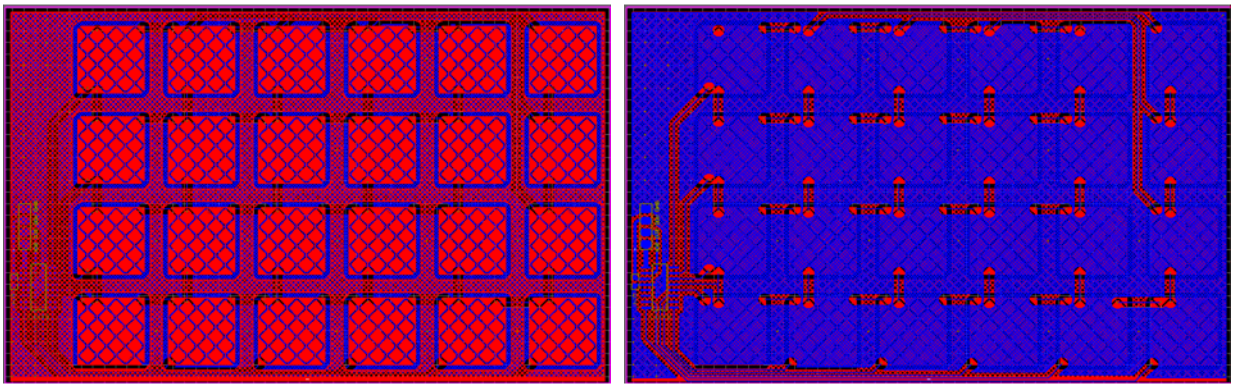
6.4 Ground Layout

It's unnecessary to pour the ground everywhere. It will actually increase the parasitic capacitances between the input channels to ground. It's recommended to use 40% grid type grounding for touch key PCB layer, angled at 45°. PCB layer for Channel routing and chip can use 60%~80% grounding, angled at 45°; keep the spacing as mentioned above for more than 40mils.

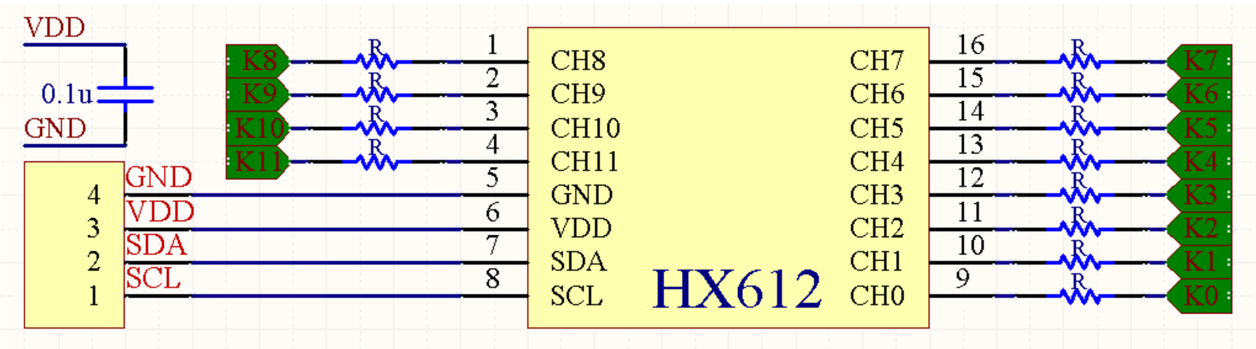
6.5 Single Channel Controlled Keys



6.6 Two Channel (Array) Controlled Keys



7. Application Circuit



Note：

1. Including a resistor R between the input channel and the touch pad will help reduce interferences from RF sources. The resistor should be near chip pins in layout. Its value depends on application; normally 10k is a good choice. It can also be removed completely with slight reduction in RF interference rejection.
2. The 0.1uF capacitor between VDD and GND should be put close to the chip pins in layout.
3. Unused channels can be left unconnected (floating).

8. Reference Driver (C)

```
#define   HX_SCL     P02                            // SCL: quasi-bidirection or push-pull output configuration
#define   HX_SDA     P03
#define   SDA_OT     P0M0 &= 0xf7; P0M1 |= 0x08     // SDA: push-pull output configuration (01)
#define   SDA_IN     P0M0 &= 0xf7; P0M1 &= 0xf7     // SDA: quasi-bidirection configuration (00)
```

8.1 MCU Write

```
void   HX612_I2C_Write(uchar addr,uchar dat)
{
    uchar    i;

    HX_SCL = 1;
    SDA_OT;
    HX_SDA = 1;
    HX_SDA = 0;
    HX_SCL = 0;
    HX_SCL = 1;
    for (i=0;i<6;i++)
    {
        HX_SCL = 0;
        HX_SDA = (bit)(addr&0x20);
        HX_SCL = 1;
        addr = addr<<1;
    }
    HX_SCL = 0;
    HX_SDA = 1;
    HX_SCL = 1;
    for (i=0;i<8;i++)
    {
        HX_SCL = 0;
        HX_SDA = (bit)(dat&0x80);
        HX_SCL = 1;
        dat = dat<<1;
    }
    HX_SCL = 0;
    HX_SDA = 1;
    HX_SCL = 1;
    HX_SCL = 0;
    HX_SDA = 0;
    HX_SCL = 1;
```

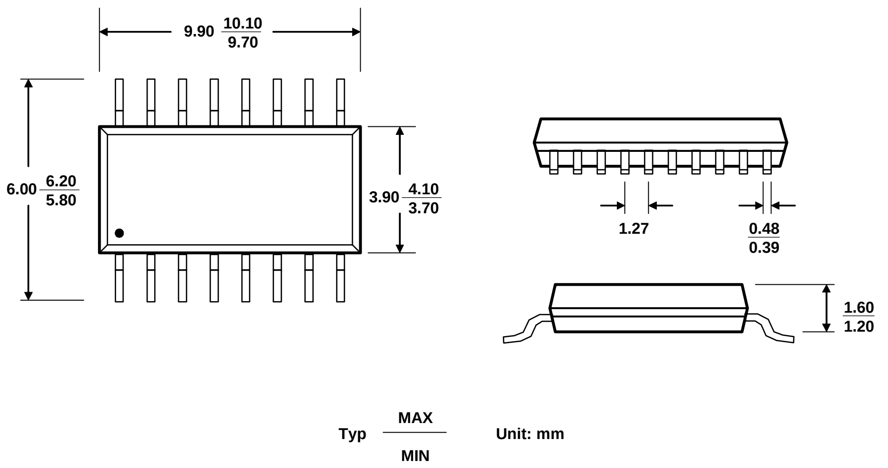
```
    HX_SDA = 1;
}
```

8.2 MCU Read

```
uint   HX612_I2C_Read(void)
{
    uchar    i;
    uint     RDat;

    HX_SCL = 1;
    SDA_IN;
    _nop_();_nop_();_nop_();      // Delay_5uS
    _nop_();_nop_();_nop_();
    RDat = 0;
    if (HX_SDA==0)                // inquiry mode
    {
        HX_SCL = 0;
        SDA_OT;
        HX_SDA = 1;
        HX_SCL = 1;
        SDA_IN;
        for (i=0;i<16;i++)
        {
            HX_SCL = 0;
            RDat = RDat<<1;
            HX_SCL = 1;
            if (HX_SDA==1)   RDat++;
        }
        HX_SCL = 0;
        _nop_();
        HX_SCL = 1;
        _nop_();
    }
    return (RDat);
}
```

9. Package Information



SOP-16L Package