Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

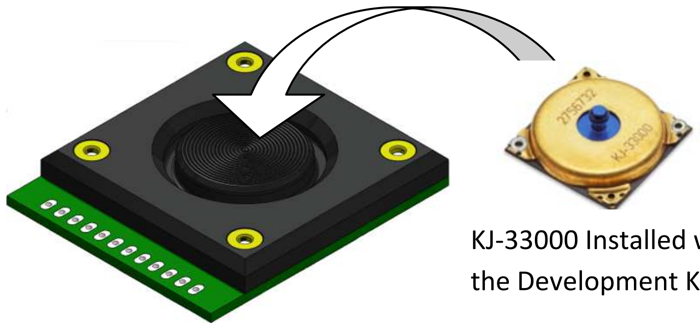
With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

Joystick I^{2}C Development Kit Programming and Application Note


KJ-33000 Installed within the Development Kit.

Knowles Acoustics

1151 Maplewood Drive

Itasca, IL 60143

1. DESCRIPTION AND APPLICATION

1.1. DESCRIPTION

The KJ-33000-002 is a developer's kit that includes the installed KJ-3300 Joystick. The Joystick is a digital joystick that communicates via an I^{2}C interface. This Application Note describes how to communicate with the Joystick.

1.2. APPLICATION

The Joystick is developed for hand held telecommunication and electronic devices. Refer to manufacturer's website (www.knowles.com) for current application notes.

2. I^{2}C Interface

The I^{2}C controller can operate up to 400kHz (fast mode). Seven-bit addressing is implemented. The preprogrammed I^{2}C address is 0x38. If the ADDR line is tied to V_{DD}, the device will respond to address 0x39.

A read initiated from the device returns two bytes. The first byte contains a left justified 4 bit value for the X axis while the second byte contains a left justified 4 bit value for the Y axis. The least significant 4 bits of both bytes are zero filled.

A write initiated to the device will send it into either Low Power Mode or Sleep Mode depending on the value of the least significant bit. A '1' corresponds to Low Power Mode while a '0' corresponds to Sleep Mode.

The open drain interrupt line is asserted (low level) on a change in either the X or Y value. No interrupts are generated when there are consecutive zeros on both axes.

3. Programming Examples

3.1. Communication to the Joystick

Communicating with the joystick using the I^{2}C bus, simply send the start condition, send the address of the device (0x38 or 0x39 based on the ADDR Pin) indicating a read operation. When an Idle condition exists, read a byte from the device (X displacement), issue an acknowledgement, then on the next idle, read another byte from the device (Y-displacement). Finally, issue a no-acknowledgement, wait for an idle condition and issue a stop condition.

Sample:

```
IdleI2C ();                 // Wait for an Idle Condition
StartI2C ();                // Issue a Start Condition
IdleI2C ();                 // Wait for an Idle Condition
address = 0x38 << 1 | 0x01; // Joystick Address Plus Read
ack = WriteI2C (address);   // Write the address plus read
IdleI2C ();                 // Wait for an Idle Condition
JoystickX = ReadI2C ();     // Read Joystick X displacement
AckI2C ();                  // Issue an Acknowledge Condition
IdleI2C ();                 // Wait for an Idle Condition
JoystickY = ReadI2C ();     // Read Joystick Y displacement
NotAckI2C ();               // Issue the No-Ack Condition
IdleI2C ();                 // Wait for an Idle Condition
StopI2C ();                 // Release the Bus
```

3.2. Lookup Tables

Lookup tables can be used to translate the displacement data to a more meaningful value. The X and Y data in the previous example will be found in the most significant 4 bits of the variables JoystickX and JoystickY. These values can be simply bit shifted right by 4 bits using the bitwise >> operator.

```
JoystickX = JoystickX >> 4;
JoystickY = JoystickY >> 4;
```

Each of the Joystick variables is now a number between 0 and 15 (4 bits). A lookup table consisting of a simple 16 byte array could be used to translate the value output by the joystick to a more relevant number. For example, the array:

```
char Xspeed[16] = {0,1,2,3,4,5,6,7,0,-1,-2,-3,-4,-5,-6,-7};
```

would be used to translate the actual output of the device to a real, displacement number.

Please note that due to the most significant bit indicating positive or negative displacement, there are 2 possibilities of zero in the joystick. 0b1000 is the same as 0b0000. The data in the array will then translate actual displacement to a number corresponding to the physical position of the stick.

Lookup tables could also be used to expand the 4 bit value to a signed 8 bit value. For example:

```
char Xspeed[16] = {0,15,31,47,63,79,95,111,0,-15,-31,-47,-63,-79,-95,-111};
```

linearly expands the value output by the joystick by indexing the JoystickX value in the array.

```
        LookupXSpeed = Xspeed[JoystickX];
```

3.3. Low Power Modes

There are two low power modes to the joystick; low power and sleep mode. To put the joystick into low power mode, a simple write to the joystick with the least significant bit being a one is required. To put the joystick into sleep mode, the least significant bit of the write must be a zero.

Sample:

```
IdleI2C ();                 // Wait for an Idle Condition
StartI2C ();                // Issue a Start Condition
IdleI2C ();                 // Wait for an Idle Condition
address = 0x38 << 1 | 0x00; // Joystick Address Plus Write
ack = WriteI2C (address);   // Write the address plus read
IdleI2C ();                 // Wait for an Idle Condition
ack = WriteI2C (0);         // Put the joystick into Low Pwr
NotAckI2C ();               // Issue the No-Ack Condition
IdleI2C ();                 // Wait for an Idle Condition
StopI2C ();                 // Release the Bus
```

This example will put the Joystick into sleep mode. The current consumption of the device will be at its minimum in this mode.

4. I^{2}C Demo Board

4.1. Clarification of Pins

For physical pin number, please see the product data sheet.

AV$_{DD}$ - Analog power supply : 3.3V typical

V$_{IO}$ – I/O power supply, 1.8V or connect to V$_{DD}$

Test – Connect to ground

ADDR – GND for address 0x38, V$_{DD}$ for address 0x39

SCL - I^{2}C clock line

Int – not necessary for operation, asserted at change of X or Y

SDA – I^{2}C data line

V$_{PP}$ – Programming power supply: Connect to VDD

V$_{SS}$ – Digital ground

V$_{DD}$ – Digital power supply: 3.3V typical

N/C - No connection

AV$_{SS}$ – Analog ground

Corner Pads – Connect to ground for best RF performance