Introduction

This application note will help developers quickly implement proof-of-concept designs using the KXCJ9 tri-axis accelerometer. Please refer to the KXCJ9 data sheet for additional implementation guidelines. While Kionix strives to ensure that our accelerometers will meet design expectations by default, it is not possible to provide default settings to work in every environment. Depending on the intended application, it is very likely that some customization will be required in order to optimize performance. We hope the information provided here will help the developer get the most out of the KXCJ9.

Circuit Schematic

This section shows recommended wiring for the KXCJ9, based on proven operation of the part. Specific applications may require modifications from these recommendations. Please refer to the KXCJ9 Data Sheet for all pin descriptions.

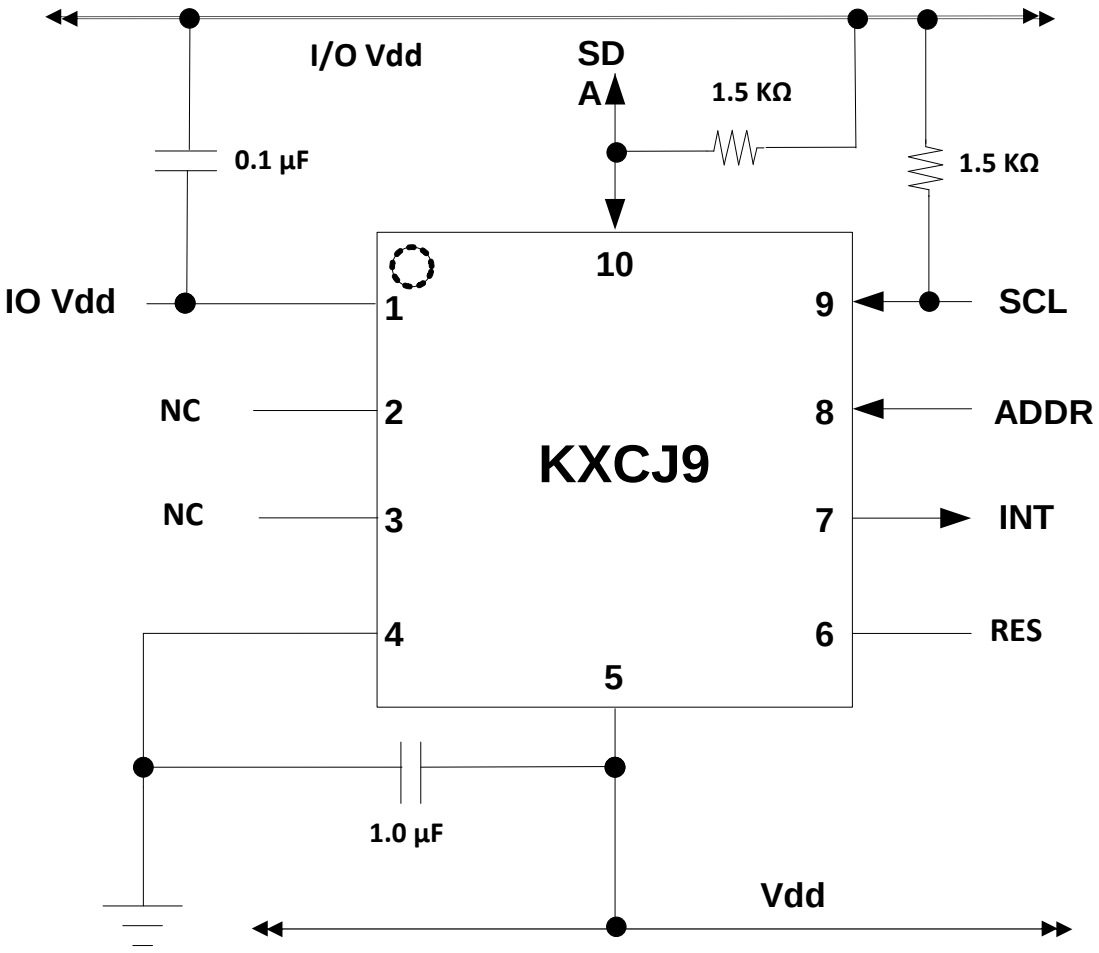

Figure 1. Application Schematic

36 Thornwood Dr. – Ithaca, NY 14850
tel: 607-257-1080 – fax: 607-257-1146
www.kionix.com - info@kionix.com

Quick Start Implementation

Here we present two basic ways to initialize the part. These can vary based on desired operation, but generally the initial operations a developer wants to do are: 1) read back acceleration data, and 2) activate the wake-up function. These cursory solutions are provided as a means for configuring the part to a known operational state. Note that these conditions just provide a starting point, and the values may vary as developers refine their application requirements.

1- Read Back Acceleration Data

- Write 0xC0 to Control Register 1 to assert PC1 (Power Control bit), set the G-range to +/-2g, and set the resolution to 12 bits.

Register Name	Address		Value	
	Hex	Binary	Hex	Binary
CTRL_REG1	**0x1B**	**0001 1011**	**0xC0**	**1100 0000**

- Acceleration data can now be read from the XOUT, YOUT, and ZOUT registers.

2- Activate Wake-Up Function

- Write 0x07 to Control Register 2 to set the ODR (output data rate) for wake up funtion to 100 Hz.
- Write 0x0A to Wakeup Timer to set the amount of time a motion must be present (100 mS) before a wake up is triggered. Here we assume that any axis or direction can trigger a wake up interrupt (INT_CTRL_REG2 default).
- Write 0x04 to Wakeup Threshold to set the motion threshold to 0.250 g.
- Write 0x38 to Interrupt Control Register 1 to reflect the wakeup interrupt on a physical pin. The interrupt pin is active high and in pulse mode (a single pulse is transmitted when motion interrupt occurs)
- Write 0xC2 to Control Register 1 to assert PC1 (Power Control bit), set the G-range to +/-2g, set the resolution to 12 bits, and enable the wake-up function.
-

Register Name	Address		Value	
	Hex	Binary	Hex	Binary
CTRL_REG2	**0x1D**	**0001 1101**	**0x07**	**0000 0111**
WAKEUP_TIMER	**0x29**	**0010 1001**	**0x0A**	**0000 1010**
WAKEUP_THRESHOLD	**0x6A**	**0110 1010**	**0x04**	**0000 0100**
INT_CTRL_REG1	**0x1E**	**0001 1110**	**0x38**	**0011 1000**
CTRL_REG1	**0x1B**	**0001 1011**	**0xC2**	**1100 0010**

- Changes to wakeup state will now be reflected in bit 4 of STATUS_REG (INT bit), bit 1 of INT_SRC_REG1 (WUFS bit), and also on the physical interrupt pin.

Timing Requirements

There are several timing requirements that developers should keep in mind when working with the KXCJ9.

I²C Clock - The I²C Clock can be up to **3.4MHz**.

Power Up to Communication - After the part is powered up, it takes **30ms** before it is ready for I²C communication.

Enable to Valid Outputs - The timing from PC1 bit in Control Register 1 being asserted to valid acceleration outputs is dependent on ODR. Please refer to the table below for start-up timings.

Parameters		Units	Min	Typical	Max
Start Up Time	RES = 0	ms		2	
	RES = 1, ODR = 0.781Hz			1200	
	RES = 1, ODR = 12.5Hz			81	
	RES = 1, ODR = 50Hz			21	
	RES = 1, ODR = 200Hz			6	
	RES = 1, ODR = 800Hz			2.5	
	RES = 1, ODR = 1600Hz			2	

Accelerometer Placement and Orientation

Placement – It is important to note that the placement of the accelerometer within the target device can have a significant effect on tap/double-tap direction resolution. If tap detection is desired, the part should be placed as far away from the edges of the device housing as possible, with the ideal location being at the target device's center of mass.

Orientation – While it is recommended to align the accelerometer's axes with those of the target device, it will sometimes be desirable or necessary to alter the part's orientation with respect to the device housing. Rotating about the Z axis at intervals of 90 degrees or about the X or Y axes at intervals of 180 degrees should not impact functionality. However, it is highly recommended that the device is not rotated 90 or 270 degrees about the X or Y axes. Due to the asymmetrical nature of the tilt position function, altering the orientation of the Z axis in this manner can cause incorrect screen rotation direction resolution.

Interrupt Configuration

The physical interrupt has 8 possible configurations, based on two states for each of the three customizable variables located in Interrupt Control Register 1:

Latched/Pulsed (IEL – bit 3 – 0x08)

0 – Latched mode – When an interrupt is triggered, it will remain active on the pin until cleared.

1 – Pulse mode – When an interrupt is triggered, it will cause a short (~50µs) pulse on the pin and clear itself.

Polarity (IEA – bit 4 – 0x10)

0 – Active Low – The interrupt pin will normally be HIGH, but will transition to LOW when an interrupt is triggered.

1 – Active High – The interrupt pin will normally be LOW, but will transition to HIGH when an interrupt is triggered.

Enable/Disable (IEN – bit 5 – 0x20)

0 – Disabled – Interrupt conditions will not be reflected on the physical interrupt pin.

1 – Enabled – Interrupt conditions will be reflected on the physical interrupt pin.

A Few Interrupt Tips

Read the Interrupt Release Register to Clear

In latched mode, the INT_REL register must be read in order to clear the physical interrupt pin. This will also clear the Interrupt Source Registers (1 and 2) and the INT bit (0x10) in the Status Register.

Microcontroller/GPIO Interrupt Handling –

GPIO configuration is based solely on the connected hardware. The KXCJ9 can be configured to issue interrupts depending on how the GPIO is programmed to catch them (if this is not the case, please contact your Kionix Sales Representative).

Generally, when an interrupt is triggered, the developer should take the following steps:

1- Disable GPIO interrupt
2- Clear GPIO interrupt and generate desired functionality
3- Enable GPIO interrupt

These steps should be taken without calling any I2C transactions if done in an interrupt context, because the operating system or kernel will not allow busy-waiting on an I/O operation during an interrupt service routine.

Interrupt Polling - If physical interrupts are not used, a polling mechanism can be devised, which checks the INT bit in STATUS_REG. This mechanism should then look at INT_SRC_REG1 to determine which caused the interrupt and what steps should be taken before clearing the interrupt source information by reading the INT_REL register.

Digital Engine Operation

Motion Detection

The Wake Up Function generates an interrupt when the part transitions from an inactive to an active state, as determined by the WUF_TIMER and WUF_THRESH register values. If the interrupt is configured in unlatched mode, it will be de-asserted when the part transitions from an active to an inactive state.

Axis Masking - It is possible to mask all wake-up events including the direction which occur on a particular axis (or axes). This is done with the 6 bits: XNWUE, XPWUE, YNWUE, YPWUE ZNWUE, and ZPWUE in Interrupt Control Register 1.

XNWU *- x negative (x-): 0 = disabled, 1 = enabled*
XPWU *- x positive (x+): 0 = disabled, 1 = enabled*
YNWU *- y negative (y-): 0 = disabled, 1 = enabled*
YPWU *- y positive (y+): 0 = disabled, 1 = enabled*
ZNWU *- z negative (z-): 0 = disabled, 1 = enabled*
ZPWU *- z positive (z+): 0 = disabled, 1 = enabled*

Timers and Thresholds

WUF (Wake Up Function) Timer – This timer establishes the number of ODR cycles that the acceleration on an unmasked axis must be above the WUF threshold before a wake up interrupt is triggered. Each count in this register equals one Motion Detection ODR cycle, as dictated by the OWUFA, OWUFB and OWUFC bits in Control Register 2.

WUF Threshold – This threshold determines how much acceleration is required in an unmasked axis in order to trigger a wake up interrupt that causes the part to transition from inactivity to activity.

Troubleshooting

All Interrupt Issues

- Make sure the KXCJ9 is configured to issue interrupt signals in the way that your GPIO is programmed to handle them.
- An oscilloscope on the physical interrupt pin can be a valuable tool to confirm physical interrupt operation.
- Double check the Wake Up Mask bits in Interrupt Control Register 2.
- The WUF Timer is based on the respective Output Data Rates, so make sure the correct cycle time is used when calculating the expected timer length (please refer to the KXCJ9 product specification).

WUF (Wake Up Function) Interrupt Not Working

- Make sure that the WUF engine is enabled (WUFE bit in Control Register 1).
- Try altering the threshold requirements to achieve desired operation. If the part is waking up too easily, try increasing the threshold. If the interrupt is not firing at all, the threshold may be set too high.
- Try shortening the timer requirements, and make sure the acceleration on an unmasked axis is above the threshold until the expiration of the WUF Timer.

Accelerometer USB Development Kit

Kionix offers an Accelerometer USB Development Kit that can be used to quickly begin the development of applications and firmware that incorporate Kionix accelerometers including the KXCJ9. The Development Kit provides a common interface to Kionix evaluation boards. For additional information regarding the development kit please refer to Kionix Application Firmware Development Kit users manual. Here is a brief description of the applications and utilities supported by the development kit –

SensorScope

This application allows the user to monitor data coming from the attached sensor. This data can be saved to a file or viewed in real time. With only two verification steps, the application will display a series of graphs representing acceleration with respect to time for each axis. This data can be used to measure the noise of the accelerometer by using the following steps:

- Place the evaluation board on a flat surface in the desired orientation.
- To change the application settings, select *Settings* from the *Edit* menu. On this menu the following settings can be changed:
  - Sampling Rate - The rate at which the software queries the accelerometer for axis data.
  - Realtime Interval - The amount of data the software will buffer and display in real time.

- Select the capture button. The application will begin to capture data immediately. Captured data is written to a file, and will not be viewable until after the capture has finished. The status bar is used to notify the user of a capture in progress.
- The application will continue to collect data until the user clicks the *Stop* button, or the resulting capture file has exceeded the file size limits (~1Gigabyte). We recommend collecting the data for at least 120 seconds.
- Captured data will be saved as a list of comma-separated values (.csv). Each entry in the list is comprised of a time, followed by the raw count for each axis (x, y, and z respectively).
- Select *Save* or *Save As* from the *File* menu to save the file.
- Open the saved file using Excel. Calculate the average of the samples. This gives the noise of the accelerometer in raw counts.

SensorCalc

This application allows the user to test and calculate the zero-g offset and sensitivity parameters of the accelerometer. Once the accelerometer is properly placed relative to the Earth's gravity, simple mouse clicks initiate a series of test sequences that result in the display of raw-count data.

SensorMap

This application allows the user to read and write to specific registers of the accelerometer. The registers and their values are all displayed simultaneously on one color-coded grid.

Motion

This utility allows the user to test whether the accelerometer has detected motion on any of the axes. Options are available to change the resolution (8-bit or 12-bit), output data rate, high-pass filter settings, WUF_THRESH and WUF_TIMER register values, or mask any of the axes.

The Kionix Advantage

Kionix technology provides for X, Y, and Z-axis sensing on a single, silicon chip. One accelerometer can be used to enable a variety of simultaneous features including, but not limited to:

Hard Disk Drive protection
Vibration analysis
Tilt screen navigation
Sports modeling
Theft, man-down, accident alarm
Image stability, screen orientation & scrolling
Computer pointer
Navigation, mapping
Game playing
Automatic sleep mode

Theory of Operation

Kionix MEMS linear tri-axis accelerometers function on the principle of differential capacitance. Acceleration causes displacement of a silicon structure resulting in a change in capacitance. A signal-conditioning CMOS technology ASIC detects and transforms changes in capacitance into an analog output voltage, which is proportional to acceleration. These outputs can then be sent to a micro-controller for integration into various applications.

For product summaries, specifications, and schematics, please refer to the Kionix MEMS accelerometer product sheets at http://www.kionix.com/sensors/accelerometer-products.php