Introduction

This application note will help developers quickly implement proof-of-concept designs using the **KXTJ3** and **KX003** tri-axial accelerometers. Please refer to the KXTJ3/KX003 product specifications 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 KXTJ3/KX003 accelerometers.

Circuit Schematic

This section shows recommended wiring for the KXTJ3/KX003 based on proven operation of the part. Specific applications may require modifications from these recommendations. Please refer to the corresponding product specification for all pin descriptions.

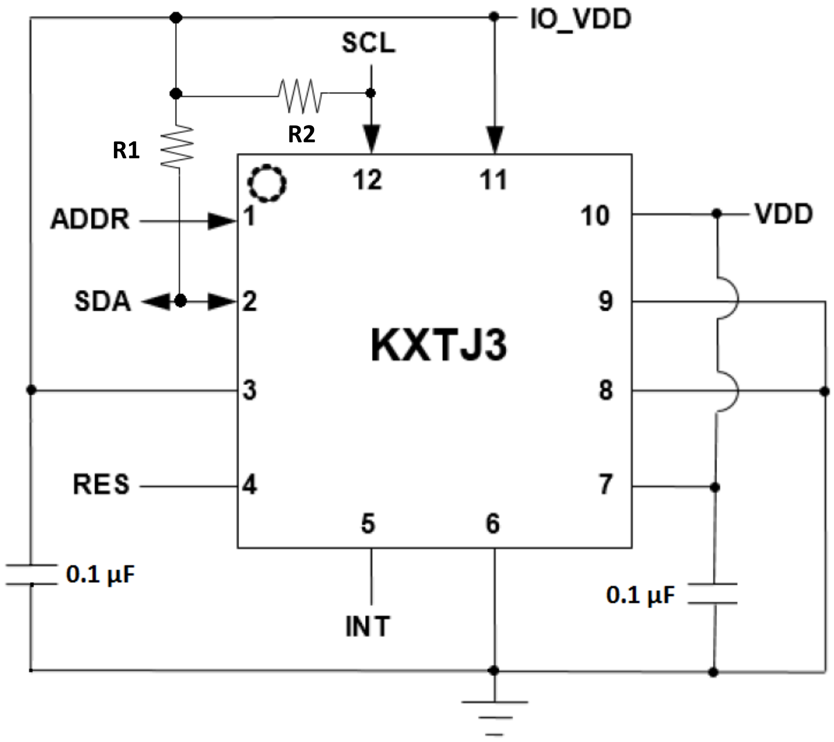

Figure 1: KXTJ3 Application Schematic

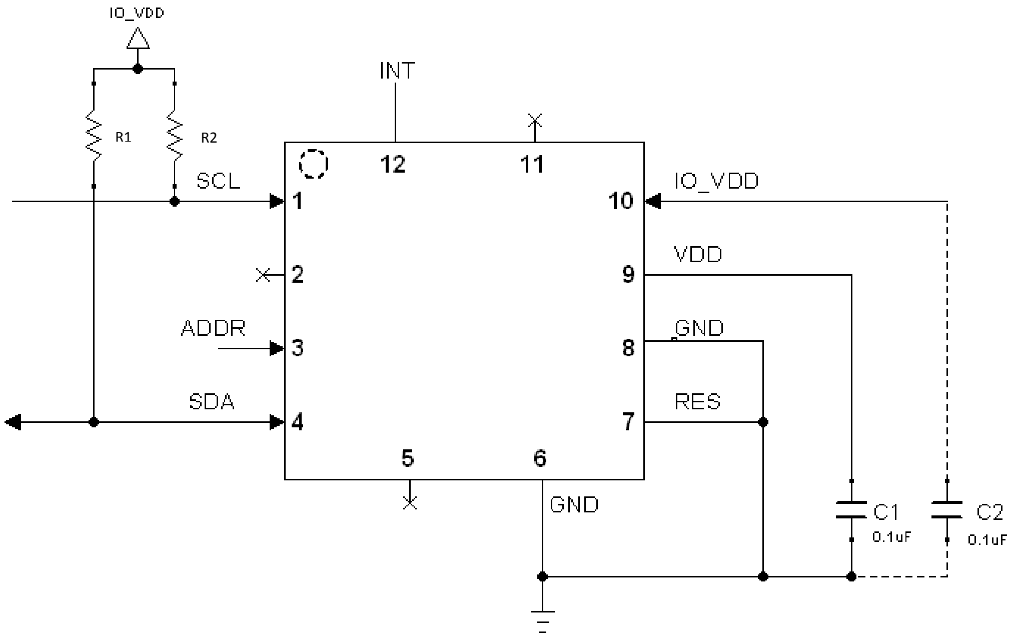

Figure 2: KX003 Application Schematic

Choose appropriate R1 and R2 for I2C communication

Quick Start Implementation

Here we present several basic ways to initialize the part's accelerometer and gyroscope. These can vary based on desired operation, but generally the initial operations a developer wants to do are: 1) read back acceleration data asynchronously, 2) read back acceleration data when next data set is ready via interrupt, 3) wake-up detection. 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. Asynchronous Read Back Acceleration Data

This example enables the accelerometer to start outputting sensor data that can be read from the output registers.

a) Write 0x00 (Reset Value) to Control Register 1 (CTRL_REG1) to set the accelerometer in disabled mode (this step must be taken prior to making any register changes).

Register Name	Address	Value
CTRL_REG1	0x1B	0x00

b) Write 0xC0 to Control Register 1 (CNTRL_REG1) to assert PC1 (Power Control bit), set the G-range to +/-2g, and set the resolution to 12 bits.

Register Name	Address	Value
CTRL_REG1	0x1B	0xC0

c) Acceleration data can now be read asynchronously from the XOUT_L, XOUT_H, YOUT_L, YOUT_H, ZOUT_L, and ZOUT_H registers in 2's complement format.

2. Synchronous Hardware Interrupt Read Back Acceleration Data

This example configures the accelerometer to start outputting sensor data synchronously with the data ready interrupt on the physical interrupt pin (INT). When data is ready, data can be read from the output registers.

a) Write 0x00 (Reset Value) to Control Register 1 (CTRL_REG1) to set the accelerometer in disabled mode (this step must be taken prior to making any register changes).

Register Name	Address	Value
CTRL_REG1	0x1B	0x00

b) Read from the Interrupt Latch Release Register (INT_REL) to clear any outstanding interrupts. The actual read value can be ignored.

Register Name	Address	Value
INT_REL	0x1A	n/a

c) Write 0x30 to Interrupt Control Register 1 (INT_CTRL_REG1) to configure the hardware interrupt. For this example, we will enable the physical interrupt pin (INT) [*IEN[5]=1*], set the polarity to active high [*IEA[4]=1*], and latch until it is cleared by reading INT_REL (0x1A) [*IEL[3]=0*].

Register Name	Address	Value
INT_CTRL_REG1	0x1E	0x30

d) Write 0xE0 to Control Register 1 (CNTRL_REG1) to assert PC1 (Power Control bit), set the G-range to ±2g, set the resolution to 12 bits, and enable reporting of the availability of new acceleration data as an interrupt (DRDYE).

Register Name	Address	Value
CTRL_REG1	0x1B	0xE0

e) Acceleration data can now be read synchronously (when INT becomes active) from the XOUT_L, XOUT_H, YOUT_L, YOUT_H, ZOUT_L, and ZOUT_H registers in 2's complement format. The interrupt can be verified by reading the STATUS_REG register (*INT[4]=1*). The source can be verified by reading the INT_SOURCE1 register (*DRDY[4]=1*).

3. Activate Wake-Up Function

This example configures the sensor to utilize the embedded Wake-up from Sleep feature and configured to generate the interrupt on INT pin. The interrupt engine can be configured by the user to report when qualified changes detected by the acceleration occur, using the accelerometer. Optionally, the user has the ability to enable/disable specific accelerometer axes and specific directions, as well as to specify the delay time. An example use cause for the engine would be to detect motion on any axis to signal an event to wake up other device or system.

a) Write 0x00 (Reset Value) to Control Register 1 (CTRL_REG1) to set the accelerometer in disabled mode (this step must be taken prior to making any register changes).

Register Name	Address	Value
CTRL_REG1	0x1B	0x00

b) Read from the Interrupt Latch Release Register (INT_REL) to clear any outstanding interrupts.

Register Name	Address	Value
INT_REL	0x1A	n/a

c) Write 0x07 to Control Register 2 (CTRL_REG2) to set the ODR (Output Data Rate) for the wake up function to 100 Hz.

Register Name	Address	Value
CTRL_REG2	0x1D	0x07

d) Write 0x0A to Wake-Up Timer Register (WAKEUP_COUNTER) to set the amount of time a motion must be present (100 msec) before a wake up is triggered. Here was assume that any axis or direction can trigger a wake up interrupt (INT_CTRL_REG2 is set to default).

Register Name	Address	Value
WAKEUP_ COUNTER	0x29	0x0A

e) Write 0x08 to Wakeup Threshold Register (WAKEUP_THRESHOLD) to set the motion threshold to 0.5 g.

Note: For the KXTJ3/KX003, the equation for WAKEUP_THRESHOLD (counts) = Desired Threshold (g) x 256 (counts/g). If 0.5g is the desired threshold, the value would be 128 (0x80). Since the threshold value spans across 2 bytes and the high byte is at the lower address, the desired value must be shifted by 4 bits since it is offset starting at bit 4 (i.e. WAKEUP_THRESHOLD = (WUTH[11:4] | WUTH[3:0]) >> 4).

Register Name	Address	Value
WAKEUP_THRESHOLD	0x6A	0x08
WAKEUP_THRESHOLD	0x6B	0x00

f) Write 0x30 to Interrupt Control Register 1 (INT_CTRL_REG1) to configure the hardware interrupt. For this example, we will enable the physical interrupt pin (INT) [*IEN[5]=1*], set the polarity to active high [*IEA[4]=1*], and latch until it is cleared by reading INT_REL (0x1A) [*IEL[3]=0*].

Register Name	**Address**	**Value**
INT_CTRL_REG1	0x1E	0x30

g) Write 0xC2 to Control Register 1 (CNTRL_REG1) 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**
CTRL_REG1	0x1B	0xC2

h) 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. Additionally, the axis and direction of the detected motion is reflected in INT_SOURCE2 register.

Timing Requirements

There are several timing requirements that developers should keep in mind when working with the KXTJ3/KX003.

I²C Clock - The I²C Clock can support Fast Mode up to **400 KHz** and High Speed mode up to **3.4 MHz**.

Interrupt Configuration

Physical Interrupt

There is one (1) available physical interrupt. It has FOUR (4) possible configurations (excluding enable/disable), based on two (2) states for polarity and two (2) states for latched/pulsed configuration for the Interrupt Pin Control Register 1 (INT_CTRL_REG1):

Enable/Disable (IEN[5])

- 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.

Polarity (IEA[4])

- 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.

Latched/Pulsed (IEL2[3])

- 0 – Latched mode – When an interrupt is triggered, it will remain active on the pin until cleared by reading INT_REL.
- 1 – Pulse mode – When an interrupt is triggered, it will cause a short (~0.03-0.05ms) pulse on the pin and clear itself.

Motion Detection

There is also an additional (1) interrupt control register (INT_CTRL_REG2) which controls which axis and direction of the detected motion can cause an interrupt.

XNWU[5]	– x negative (x-):	0 = disabled, 1 = enabled
XPWU[4]	– x positive (x-):	0 = disabled, 1 = enabled
YNWU[3]	– y negative (x-):	0 = disabled, 1 = enabled
YPWU[2]	– y positive (x-):	0 = disabled, 1 = enabled
ZNWU[1]	– z negative (x-):	0 = disabled, 1 = enabled
ZPWU[0]	– z positive (x-):	0 = disabled, 1 = enabled

Interrupts

CTRL_REG1 controls both interrupts (DRDYE and WUFE) on the KXTJ3/KX003. These bits are used to enable/disable the respective interrupt.

DRDYE[5]	– Enables/Disables new acceleration data as interrupt
WUFE[1]	– Enables/Disables the Wake-Up (motion detect) function

[By Default: All interrupts are not enabled]
- 0 – Disabled – Associated interrupt is disabled
- 1 – Enabled – Associated interrupt is enabled

A Few Interrupt Tips

Read the Interrupt Release Register to Clear

In latched mode, the INT1_REL registers must be read in order to clear the physical interrupt pin. This will also clear the Interrupt Source Registers and the particular INT bit in the Interrupt Source Register.

Microcontroller/GPIO Interrupt Handling

GPIO configuration is based solely on the connected hardware. The KXTJ3/KX003 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 digital communication 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 SOURCE_REG register. If reading acceleration data, the status bit is also cleared if data is read from the respective output registers. If using WUF, this mechanism should then look at INT_SOURCE2 to determine which direction caused the interrupt and what steps should be taken before clearing the interrupt source information by reading the INT_ REL register.

Wake-up Function Tips

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_THRESHOLD register values. If the interrupt is configured in unlatched mode (ULMODE[7]=1 in INT_CTRL_REG2), it will be de-asserted when the non-activity time required has expired before another wake-up interrupt can be set (NA_COUNTER).

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 CTRL_REG2.

WUF (Wake Up Function) Threshold

- This threshold determines how much acceleration is required in an un-masked 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 KXTJ3/KX003 is enabled and configured to issue interrupt signals in the way that your GPIO is programmed to handle them (INT_CTRL_REG1).
- An oscilloscope on the physical interrupt pin can be a valuable tool to confirm physical interrupt operation.
- Double check the appropriate interrupts are enabled in the CTRL_REG1 register

Accelerometer Data Ready Interrupt Not Working

- Make sure that the Accelerometer Data Ready interrupt is enabled (DRDYE) in the Control Register 1 (CTRL_REG1).

- Ensure the interrupt is not latched by reading the INT_REL register
- Make sure that the physical interrupt signal is enabled (INT_CTRL_REG1)

Wake-Up Interrupt Not Working

- Make sure that the Wake-Up interrupt is enabled (WUFE) in the Control Register 1 (CTRL_REG1).
- Ensure the interrupt is not latched by reading the INT_REL register
- Make sure that the physical interrupt signal is enabled (INT_CTRL_REG1)

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 arising from the acceleration induced motion. 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, which is proportional to acceleration, into an analog output voltage that is sent through an analog-to-digital converter. The acceleration data may be accessed through the output data registers by the micro-controller for use in various applications.

For product summaries, specifications, and schematics, please refer to the Kionix MEMS accelerometer product sheets at http://www.kionix.com/parametric/Accelerometers.