MMA8451, 2, 3Q Single/Double and Directional Tap Detection

by: Kimberly Tuck
Applications Engineer

1.0 Introduction

Tapping is the way that users employ interacting with hand-held products that have an integrated accelerometer. For example, a single tap can be used to silence the sound; a double tap could be used to send an incoming call directly to voicemail. Directional tapping from the positive and negative directions of X, Y, and Z can be used in gaming for motion inputs or for changing pictures in slide show mode or as other generic functions. Tap as an input to the product device allows users to enjoy a more advanced and interactive user experience. The tap replaces the need for the mouse to point and click and select options.

The MMA8451, 2, 3Q has embedded single tap double and directional tap capabilities. Examples and explanations of how to configure the device for the embedded tap as well as the software algorithm for directional tap will be given in this application note. Note that in the data sheet the register names for tap are sometimes referred to as “pulse”. The signature of the tap signal from the accelerometer looks like a pulse, and that is where this reference derives. Pulse and double pulse are considered the same as tap and double tap in the literature.

1.1 Key Words

Accelerometer, Output Data Rate (ODR), Current Consumption, Tap, Directional Tap, Double Tap, Pulse, Double Pulse, Menu Selection, Threshold, Timing Window, Latency Timer, Second Tap, Sample Rate, Double Pulse Abort, Interrupt, FIFO, High Pass Filter (HPF), Low Pass Filter (LPF)

TABLE OF CONTENTS

1.2 Summary

A. A tap signature has a fast response time: sampling speed is important. The timing windows and conditions for detecting single and double tap are adjustable in the MMA8451, 2, 3Q to eliminate false taps

B. The signature of a tap is somewhat dependent on the user as well as on the mounting conditions of the accelerometer in the product.

C. The embedded single tap and the FIFO can be used for directional tap if a software routine is required in the MMA8451Q. The MMA8451, 2, 3Q has its own embedded directional information for single and double tap. Either way, the event is interrupt driven and the processor can go into a low power mode until the event occurs.

D. The MMA8451, 2, 3Q can be configured to detect a single tap only, a double tap only or both single and double taps to the same interrupt pin. It can also detect the direction of the tap.

E. The conditions for double tap detection and rejection are given in various scenario diagrams to assist the user to configure the tap feature appropriately.

F. The latch (**ELE** bit set) will hold the contents of the status register until the status register is read.

G. The X, Y, and Z tap thresholds for single and double tap can be changed in either active or standby mode.

2.0 MMA8451, 2, 3Q Consumer 3-axis Accelerometer 3 x 3 x 1 mm

The MMA8451, 2, 3Q has a selectable dynamic range of ±2g, ±4g, ±8g. The device has 8 different output data rates, selectable high pass filter cut-off frequencies, and high pass filtered data. The available resolution of the data and the embedded features is dependant on the specific device.

Note: The MMA8450Q has a different memory map and has a slightly different pin-out configuration.

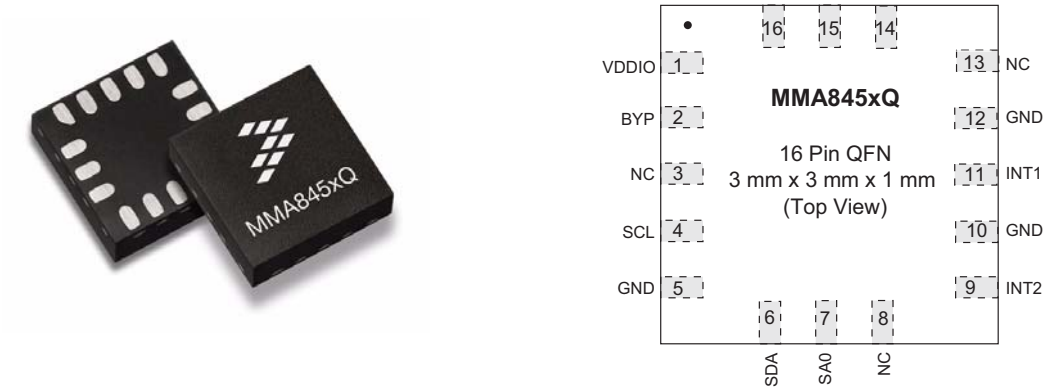

Figure 1. MMA8451, 2, 3Q Consumer 3-axis Accelerometer 3 x 3 x 1 mm

2.1 Output Data, Sample Rates and Dynamic Ranges of all Three Products

2.1.1 MMA8451Q

1. **14-bit data**
   2g (4096 counts/g = 0.25 mg/LSB) **4g** (2048 counts/g = 0.5 mg/LSB) **8g** (1024 counts/g = 1 mg/LSB)
2. **8-bit data**
   2g (64 counts/g = 15.6 mg/LSB) **4g** (32 counts/g = 31.25 mg/LSB) **8g** (16 counts/g = 62.5 mg/LSB)
3. **Embedded 32 sample FIFO (MMA8451Q)**

2.1.2 MMA8452Q

1. **12-bit data**
   2g (1024 counts/g = 1 mg/LSB) **4g** (512 counts/g = 2 mg/LSB) **8g** (256 counts/g = 3.9 mg/LSB)
2. **8-bit data**
   2g (64 counts/g = 15.6 mg/LSB) **4g** (32 counts/g = 31.25 mg/LSB) **8g** (16 counts/g = 62.5 mg/LSB)

2.1.3 MMA8453Q Note: No HPF Data

1. **10-bit data**
   2g (256 counts/g = 3.9 mg/LSB) **4g** (128 counts/g = 7.8 mg/LSB) **8g** (64 counts/g = 15.6 mg/LSB)
2. **8-bit data**
   2g (64 counts/g = 15.6 mg/LSB) **4g** (32 counts/g = 31.25 mg/LSB) **8g** (16 counts/g = 62.5 mg/LSB)

2.2 Application Notes for the MMA8451, 2, 3Q

The following is a list of all the application notes available for the MMA8451, 2, 3Q:

- **AN4068**, *Embedded Orientation Detection Using the MMA8451, 2, 3Q*
- **AN4069**, *Offset Calibration of the MMA8451, 2, 3Q*
- **AN4070**, *Motion and Freefall Detection Using the MMA8451, 2, 3Q*
- **AN4071**, *High Pass Filtered Data and Functions Using the MMA8451, 2,3Q*
- **AN4072, *MMA8451, 2, 3Q Single/Double and Directional Tap Detection***
- **AN4073**, *Using the 32 Sample First In First Out (FIFO) in the MMA8451Q*
- **AN4074**, *Auto-Wake/Sleep Using the MMA8451, 2, 3Q*
- **AN4075**, *How Many Bits are Enough? The Trade-off Between High Resolution and Low Power Using Oversampling Modes*
- **AN4076**, *Data Manipulation and Basic Settings of the MMA8451, 2, 3Q*

3.0 Tap Detection Applications Using the MMA8451, 2, 3Q Accelerometer

Figure 2 shows the signature of a tap event. This data was collected on an analog sensor at 2 kHz to show a very complete signature of the waveform. It is important to note that the tap signature from the Z data of the accelerometer can look somewhat different depending on the mechanics and mounting conditions of the sensor. To analyze the tap signature for writing algorithms, it is beneficial to mount the accelerometer under the conditions of the final product to do testing and analyze various similar waveforms. This data was collected by tapping on the back of our standard Sensor Toolbox Demo Boards. There are a few things to notice about the signature of the waveform:

- The time period of the entire event in this case lasts for about 10 ms.
- There is a sharp downward acceleration followed by an immediate rebound in a short time. Then the waveform settles.
- The embedded algorithms for single and double tap analyze the tap waveform comparing thresholds and timing conditions.

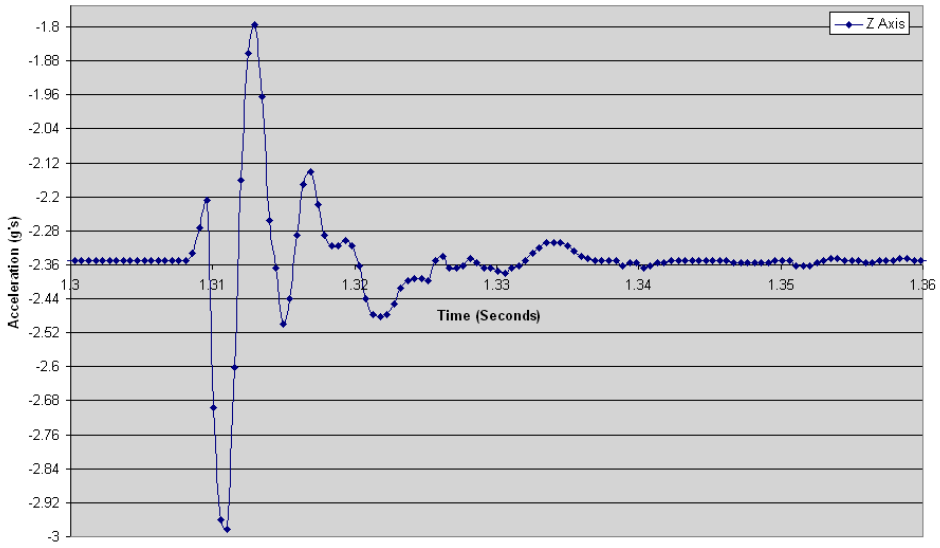

Figure 2. Tap Signature Z-axis Sample Rate 2 kHz

3.1 Single Tap

For a single tap event there are three key parameters to consider.

1. Time Window for the tap signature to cross a threshold and drop below it (**PULSE_TMLT)**
2. Threshold Value to Trigger the event (**PULSE_THSX, PULSE_THSY, PULSE_THSZ)**
3. Latency time to hold the event conditions (**PULSE_LTCY)**

Single tap is embedded in the design of the MMA8451, 2, 3Q. The tap event function must first be enabled to detect a single tap event with the corresponding axes enabled. A single tap event is configured by specifying two things; the threshold to be crossed and the time duration that it lasts. A single tap event is detected when the acceleration value is greater than the value set in the **PULSE_THSX** (Reg 0x23)**, PULSE_THSY** (Reg 0x24) and **PULSE_THSZ** (Reg 0x25) registers. The time period of the event must be shorter than the time set in the **PULSE_TMLT (Reg** 0x26) time limit register.

When configured for a single tap event, an interrupt is generated when the input acceleration on the selected axis exceeds the programmed threshold, and returns below it within a time window defined by **PULSE_TMLT**. If the **ELE** bit (bit 6) of the **PULSE_CFG (Reg** 0x21) register is not set, the interrupt is kept high for the duration of the Latency window **PULSE_LTCY** (Reg 0x27). The latency window is a user-definable period of delay after each pulse. This latency window applies either for single pulse or double pulse detection. If the **ELE** bit is set, the source register values will remain static until the **PULSE_SRC** (Reg 0x22) register is read.

Figure 3, shows a valid and invalid pulse condition.

A. Note in condition **(a)** the interrupt is asserted since the acceleration due to a pulse exceeds the specified acceleration threshold (value set in the **PULSE_THSX**) and crosses up and down before the specified Pulse Time Limit (value set in **PULSE_TMLT**) expires.

B. Note that in condition **(b)** the acceleration due to a pulse exceeds the specified acceleration threshold limit, but does not go below the threshold before the specified Pulse Time Limit expires. Therefore, this is an invalid pulse and the interrupt will not be triggered. Also note that the Latency is not shown for this example.

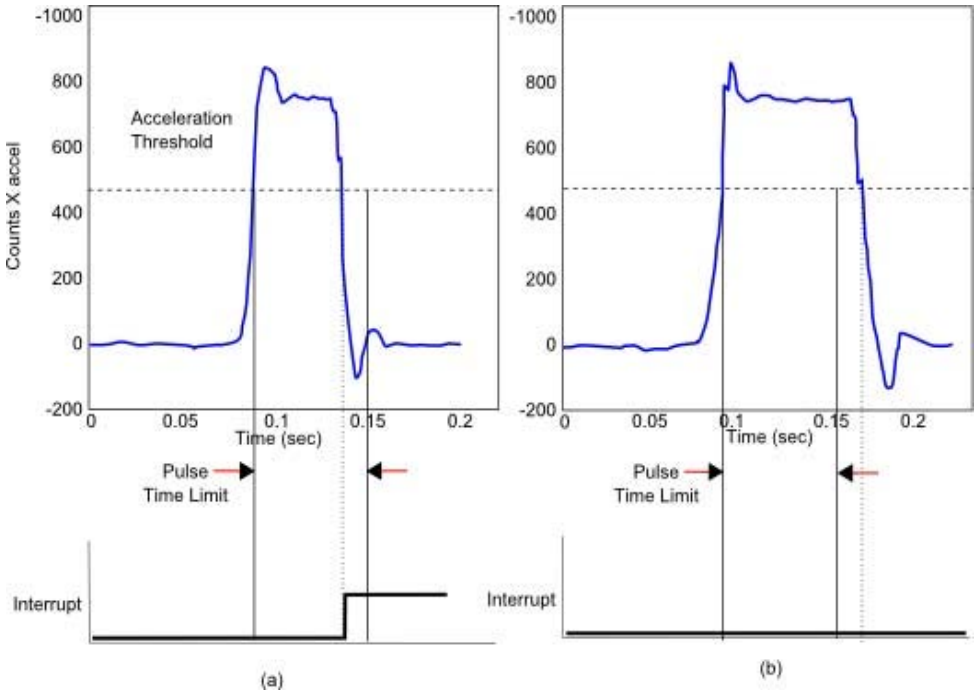

Figure 3. Single Tap Detection Conditions

3.2 Double Tap

A double tap event is detected when the device is configured for two single taps within defined time periods. The second time must occur after the time specified by the **PULSE_LTCY** (Reg 0x27) register, but also within the time limit of the **PULSE_WIND** (Reg 0x28) register.

If the device is configured for double tap detection, an interrupt is generated after both taps are recognized. The recognition of the second tap occurs only if the event satisfies the constraints defined by the Pulse Latency, Pulse Time Limit and Double Pulse Abort. In particular, after the first tap has been recognized, the second tap detection procedure is delayed for an interval defined by the Latency register. This means that after the first tap has been recognized, the second tap detection procedure will start only if the input acceleration exceeds the threshold after the Pulse Latency window but before the Pulse Window has expired or if the acceleration is still above the threshold after the Latency has expired.

Once the second pulse detection procedure is initiated, the second pulse will be recognized with the same rule as the first: the acceleration must return below the threshold before the Time Limit has expired. Appropriately defining the Latency window is important to avoid unwanted pulses due to spurious bouncing of the input signal.

Figure 4 illustrates a single pulse event **(a)** and a double pulse event **(b)**. The device is able to distinguish between **(a)** and **(b)** by changing the settings of the **PULSE_CFG** register from single to double pulse recognition.

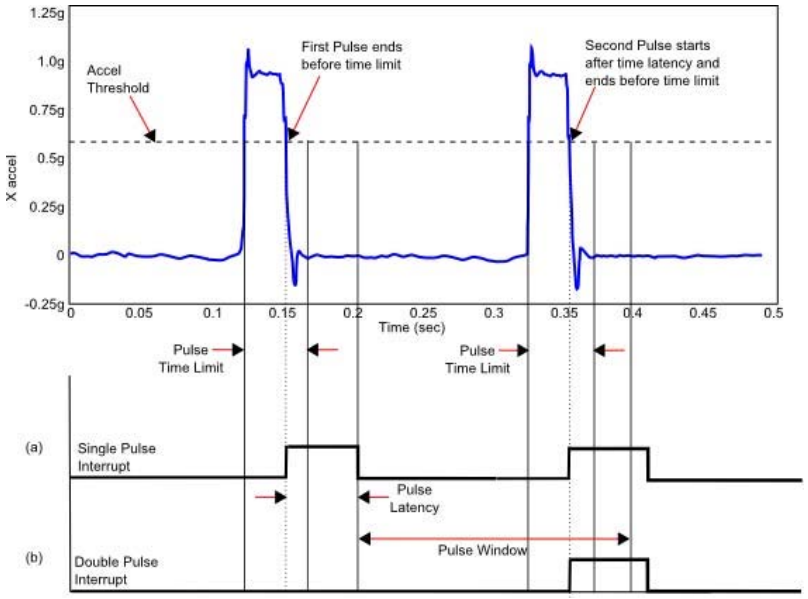

Figure 4. Single and Double Pulse Recognition

3.3 Various Scenarios Using Double Pulse Abort Conditions

The next image series illustrates various scenarios using the double pulse abort condition. Please refer to Tables 5, 6, 7, 8 and 9.

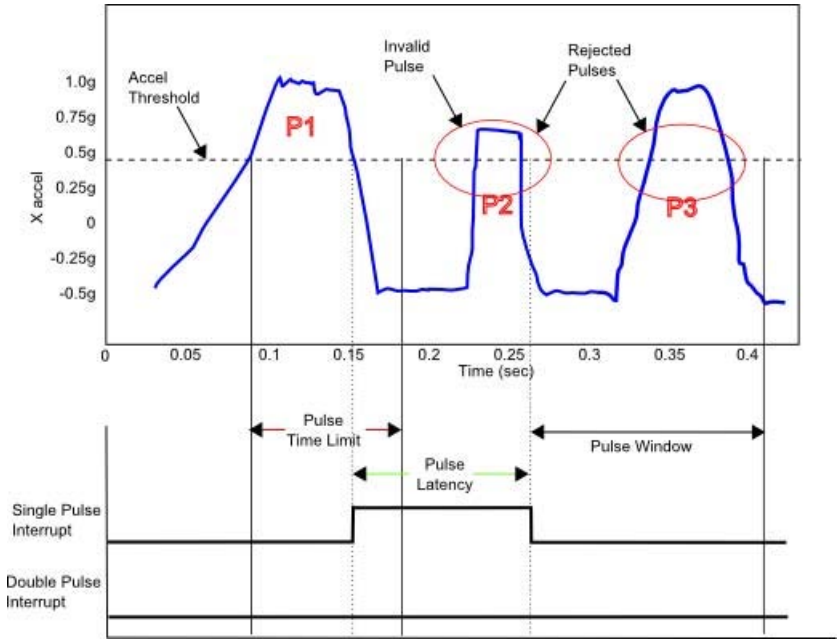

Figure 5. Double Pulse Aborted Due to Invalid Pulse During the Pulse Latency Phase (DPA = 1)

Note: In Figure 5, the Pulse P2 is rejected because the start of pulse P2 was detected during the PULSE_LTCY period.

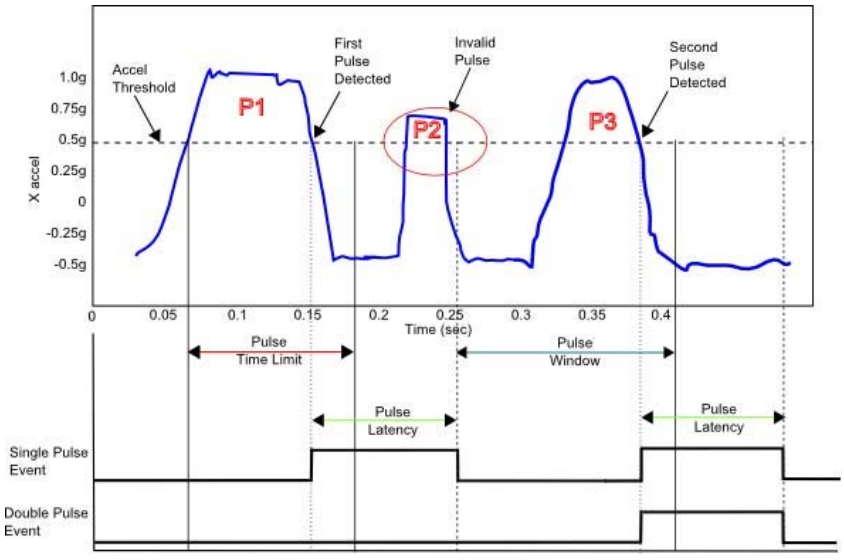

Figure 6. Double Pulse Detected Irrespective of the Invalid Pulse Status (DPA = 0)

Note: In Figure 6, Pulse P1 is detected as a valid single pulse, P2 is ignored because it occurs during the time period specified by the **PULSE_LTCY** register; but pulse P3 is detected as double pulse because the **DPA** bit is set to 0 and pulse P2 was ignored.

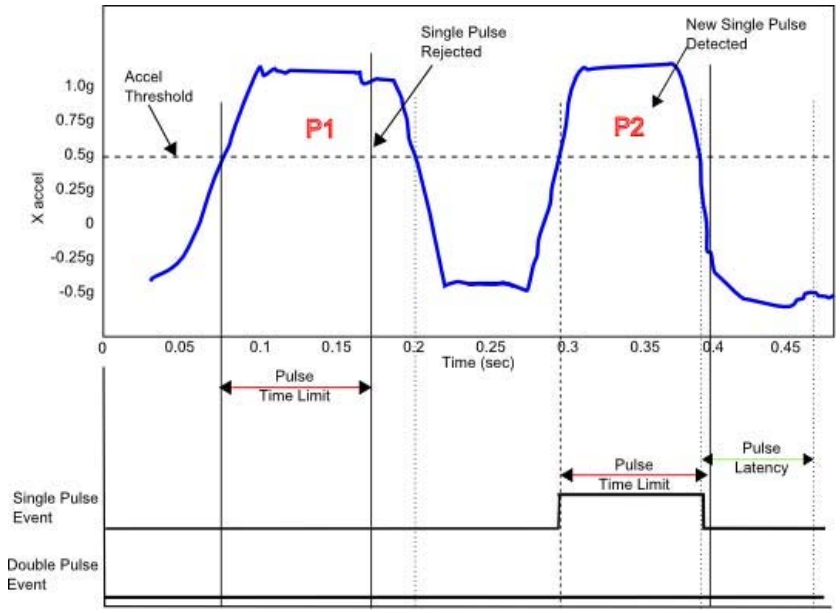

Figure 7. Double Pulse Rejected Due to Second Pulse Width Exceeding Pulse_TMLT Period (DPA = 1)

Note: In Figure 7, the second pulse is rejected because its pulse width is longer than the time period specified by the **PULSE_TMLT** register. After the **PULSE_WIND** period has expired, a new single pulse detection can begin.

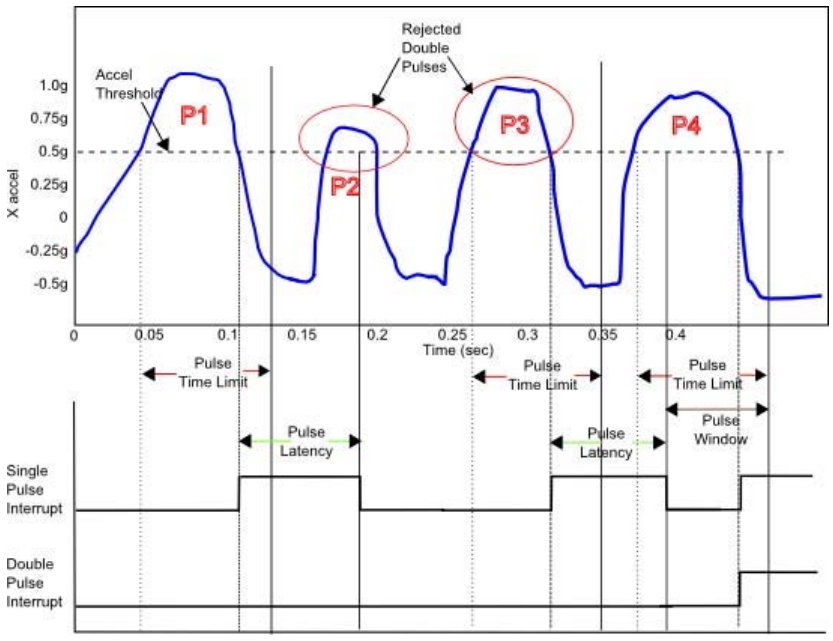

Figure 8. Double Pulse Rejected Due to Invalid Pulse During the Pulse Latency Phase (DPA = 1)

Note: In Figure 8, Pulse **P2** is rejected as a double pulse because it starts before the expiration of the time period specified by the **PULSE_LTCY** register. Pulse **P3** is detected as a new single pulse while pulse **P4** is detected as a single and double pulse.

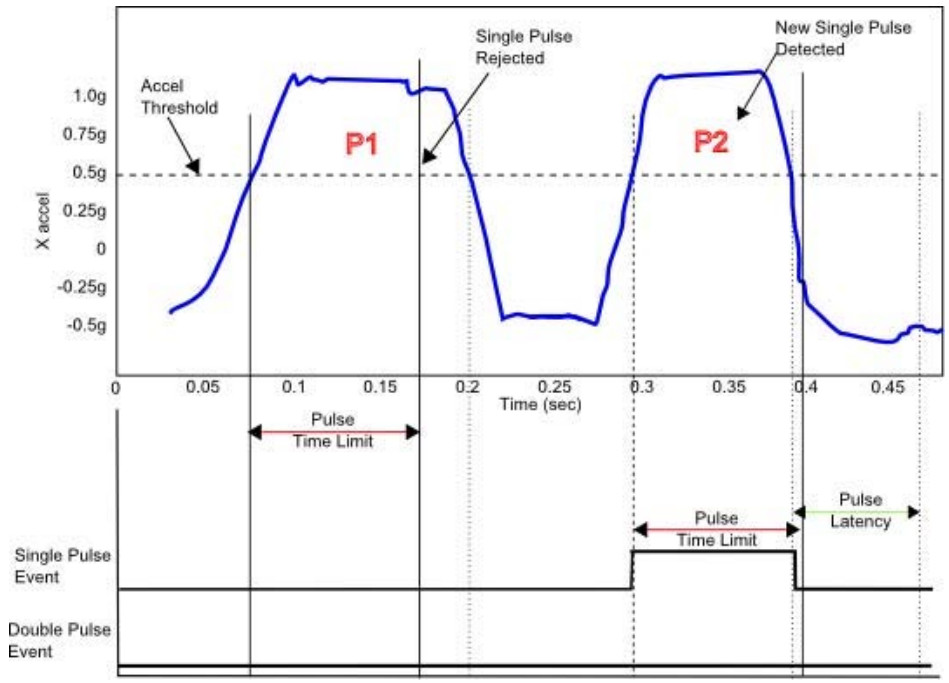

Figure 9. Single Pulse Rejected Due to First Pulse Width Exceeding Pulse_TMLT Period (DPA = 1)

Note: In Figure 9, Pulse P1 is rejected because its pulse width is longer than the time period specified by the **PULSE_TMLT** register.

4.0 Configuring Registers for Single and Double Tap Detection

To utilize the single and/or double tap detection the following eight (8) registers must be configured.

1. Register 0x21: **PULSE_CFG** The Pulse Configuration Register
2. Register 0x22: **PULSE_SRC** Pulse Source Register
3. Register 0x23 - 0x25: **PULSE_THSX,Y,Z** Pulse Threshold for X, Y and Z Registers
4. Register 0x26: **PULSE_TMLT** Pulse Time Window 1 Register
5. Register 0x27: **PULSE_LTCY** Pulse Latency Timer Register
6. Register 0x28: **PULSE_WIND** Second Pulse Time Window Register

4.1 Register 0x21: PULSE_CFG Pulse Configuration Register

This register configures the event flag for the tap detection interrupt function for enabling/disabling single and double pulse on each of the axes. This register determines the following:

- Which axes will be involved
- Whether the event to detect is a single tap or a double tap
- Whether the status bits will be latched or not in the source register, Table 1 provides an example for enabling:
  - Single tap only
  - Double tap only or,
  - Both single and double tap by combining the two, setting Register 0x21 to 0x7F

Table 1. Register 0x21 PULSE_CFG Register (Read/Write) and Description

Tap Enable	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	DPA	ELE	ZDPEFE	ZSPEFE	YDPEFE	YSPEFE	XDPEFE	XSPEFE
Single Tap	0	1	0	1	0	1	0	1
Double Tap	0	1	1	0	1	0	1	0
Both S & D	0	1	1	1	1	1	1	1

4.1.1 Example Settings for Configuration of Single and/or Double Pulse Detection

Example X, Y and Z configured for Single Tap with Latch enabled

Code Example: IIC_RegWrite(0x21, 0x55);

Example X, Y and Z configured for Double Tap with Latch enabled

Code Example: IIC_RegWrite(0x21, 0x6A);

Example X, Y and Z configured for Single Tap and Double Tap with Latch enabled

Code Example: IIC_RegWrite(0x21, 0x7F);

4.2 Registers 0x23 - 0x25 PULSE_THSX, Y, Z Pulse Threshold for X, Y and Z Registers

The pulse threshold can be set separately for the X, Y, and Z axes. This allows for the same change in acceleration for all axes to be set regardless of the static orientation. The threshold values range from 0 to 127 (7 bits expressed as an absolute value) with steps of 0.063g/LSB at a fixed 8g acceleration range.

The **PULSE_THSX**, **PULSE_THSY** and **PULSE_THSZ** registers define the threshold for each of the axes. These are shown in Tables 2, 3, and 4.

Note: The thresholds can be changed in either standby or active mode. This is very useful since if there is a change in orientation the thresholds can all be changed immediately to compensate without interrupting the detection process.

Table 2. Register 0x23 PULSE_THSX Register (Read/Write)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	THSX6	THSX5	THSX4	THSX3	THSX2	THSX1	THSX0

Table 3. Register 0x24 PULSE_THSY Register (Read/Write)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	THSY6	THSY5	THSY4	THSY3	THSY2	THSY1	THSY0

Table 4. Register 0x25 PULSE_THSZ Register (Read/Write)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	THSZ6	THSZ5	THSZ4	THSZ3	THSZ2	THSZ1	THSZ0

4.2.1 Example: Set the Tap Detection Threshold for 2g on X, 2g on Y and 3g on Z

2g/0.063g/count = 32 counts
3g/0.063g/count = 48 counts

Code Example: IIC_RegWrite(0x23, 0x20); //Set X Threshold to 32 counts or 2g

Code Example: IIC_RegWrite(0x24, 0x20); //Set Y Threshold to 32 counts or 2g

Code Example: IIC_RegWrite(0x25, 0x0C); //Set Z Threshold to 48 counts or 3g

4.3 Register 0x26: PULSE_TMLT Pulse Time Window 1 Register

The byte **PULSE_TMLT** (bit fields defined as Tmlt7 through Tmlt0) define the maximum time interval that can elapse between the start of the acceleration on the selected axis exceeding the specified threshold and the end when the axes of acceleration must go back below the specified threshold.

Table 5. Register 0x26 PULSE_TMLT Register (Read/Write)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Tmlt7	Tmlt6	Tmlt5	Tmlt4	Tmlt3	Tmlt2	Tmlt1	Tmlt0

The minimum time step for the pulse time limit is defined in Table 6 and Table 7. The maximum time for a given ODR is the minimum time step multiplied by 255. The time steps available are dependent on the oversampling mode and whether the low pass filter option is enabled or not. The Pulse Low Pass Filter (LPF) is set in Register 0x0F. When the low pass filter is enabled the time step doubles. The filter should help eliminate additional ringing after the tap signature is detected.

Table 6. Time Step for PULSE Time Limit (Reg 0x0F) Pulse_LPF_EN = 1

ODR (Hz)	Max Time Range (s)				Time Step (ms)			
	Normal	LPLN	HighRes	LP	Normal	LPLN	HighRes	LP
800	0.319	0.319	0.319	0.319	1.25	1.25	1.25	1.25
400	0.638	0.638	0.638	0.638	2.5	2.5	2.5	2.5
200	1.28	1.28	0.638	1.28	5	5	2.5	5
100	2.55	2.55	0.638	2.55	10	10	2.5	10
50	5.1	5.1	0.638	5.1	20	20	2.5	20
12.5	5.1	20.4	0.638	20.4	20	80	2.5	80
6.25	5.1	20.4	0.638	40.8	20	80	2.5	160
1.56	5.1	20.4	0.638	40.8	20	80	2.5	160

With the Pulse LPF enabled, an ODR setting of 400 Hz with normal mode would result in a maximum pulse time limit of (2.5 ms * 255) = 638 ms.

Table 7. Time step for PULSE Time Limit (Reg 0x0F) Pulse_LPF_EN = 0

ODR (Hz)	Max Time Range (s)				Time Step (ms)			
	Normal	LPLN	HighRes	LP	Normal	LPLN	HighRes	LP
800	0.159	0.159	0.159	0.159	0.625	0.625	0.625	0.625
400	0.159	0.159	0.159	0.319	0.625	0.625	0.625	1.25
200	0.319	0.319	0.159	0.638	1.25	1.25	0.625	2.5
100	0.638	0.638	0.159	1.28	2.5	2.5	0.625	5
50	1.28	1.28	0.159	2.55	5	5	0.625	10
12.5	1.28	5.1	0.159	10.2	5	20	0.625	40
6.25	1.28	5.1	0.159	10.2	5	20	0.625	40
1.56	1.28	5.1	0.159	10.2	5	20	0.625	40

When the Pulse LPF is disabled an ODR setting of 400 Hz with normal mode would result in a maximum pulse time limit of (0.625 ms * 255) = 159 ms.

4.3.1 Example Settings for the Pulse Time Limit

Example: To set the Pulse Time Limit for 30 ms at 200 Hz ODR in Normal Mode with the LPF Enabled A. 30 ms/5 ms = 6 counts

Code Example: IIC_RegWrite(0x26, 0x06);

4.4 Register 0x27: PULSE_LTCY Pulse Latency Timer Register

The Pulse Latency Timer Register is the time duration that the tap event can be read from the source register to detect the X, Y, and Z for single pulse or double pulse events without the latch enabled. The duration of any specified latency time is valid each time a single or double pulse occurs. The latency time duration for every pulse is the value specified by Register 0x27 shown in Table 8.

Table 8. Register 0x27 PULSE_LTCY Register (Read/Write)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Ltcy7	Ltcy6	Ltcy5	Ltcy4	Ltcy3	Ltcy2	Ltcy1	Ltcy0

The Pulse Latency Register (bit fields defined as Ltcy7 through Ltcy0) defines the time interval that starts after the first pulse detection, from which point the pulse detection function ignores the start of new pulses.

The minimum time step for the pulse latency is defined in Table 9 and Table 10. The maximum time is the minimum time step at the ODR and Power Mode multiplied by 255. Notice that the timing is dependent on the oversampling mode.

Table 9. Time Step for PULSE Latency @ ODR and Power Mode (Reg 0x0F) Pulse_LPF_EN = 1

ODR (Hz)	Max Time Range (s)				Time Step (ms)			
	Normal	LPLN	HighRes	LP	Normal	LPLN	HighRes	LP
800	0.638	0.638	0.638	0.638	2.5	2.5	2.5	2.5
400	1.276	1.276	1.276	1.276	5	5	5	5
200	2.56	2.56	1.276	2.56	10	10	5	10
100	5.1	5.1	1.276	5.1	20	20	5	20
50	10.2	10.2	1.276	10.2	40	40	5	40
12.5	10.2	40.8	1.276	40.8	40	160	5	160
6.25	10.2	40.8	1.276	81.6	40	160	5	320
1.56	10.2	40.8	1.276	81.6	40	160	5	320

Table 10. Time Step for PULSE Latency @ ODR and Power Mode (Reg 0x0F) Pulse_LPF_EN = 0

ODR (Hz)	Max Time Range (s)				Time Step (ms)			
	Normal	LPLN	HighRes	LP	Normal	LPLN	HighRes	LP
800	0.318	0.318	0.318	0.318	1.25	1.25	1.25	1.25
400	0.318	0.318	0.318	0.638	1.25	1.25	1.25	2.5
200	0.638	0.638	0.318	1.276	2.5	2.5	1.25	5
100	1.276	1.276	0.318	2.56	5	5	1.25	10
50	2.56	2.56	0.318	5.1	10	10	1.25	20
12.5	2.56	10.2	0.318	20.4	10	40	1.25	80
6.25	2.56	10.2	0.318	20.4	10	40	1.25	80
1.56	2.56	10.2	0.318	20.4	10	40	1.25	80

4.4.1 Example Settings for Pulse Latency Timer

Set the Pulse Latency Timer to 200 ms, 200 Hz ODR Low Power Mode, LPF Not Enabled.

200 ms/5.0 ms = 40 counts

Code Example: IIC_RegWrite(0x27, 0x28);

4.5 Register 0x28: PULSE_WIND Second Pulse Time Window Register

Table 11. Register 0x28 PULSE_WIND Register (Read/Write)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Wind7	Wind6	Wind5	Wind4	Wind3	Wind2	Wind1	Wind0

The **PULSE_WIND** register with (bit fields Wind7 through Wind0) defines the maximum interval of time that can elapse after the end of the latency interval within which the start of the second pulse event must be detected (provided the device has been configured with double pulse detection enabled). The detected second pulse width must be shorter than the time limit constraints specified by the **PULSE_TMLT** register, but the double pulse need not cross the threshold within the time specified by the **PULSE_WIND** register. The minimum time step for the pulse window is defined in Table 8 and Table 9. The timing is the same as that of the latency timer. The maximum time is the time step at the ODR and oversampling mode multiplied by 255.

4.5.1 Example Settings for Pulse Window for Detecting a Double Tap

Set the Pulse window to 300 ms, 100 Hz ODR Low Power Mode, LPF Enabled

300 ms/20 ms = 15 counts

Code Example: IIC_RegWrite(0x28, 0x0F);

4.6 Register 0x22: PULSE_SRC Pulse Source Register

This register sets a flag for the appropriate bit if a double or single pulse event has occurred on any axis and the direction that it occurred. Note: The corresponding axis and event has to be enabled above in Register 0x21 for the event to be seen in the source register. The event active flag will remain high until the latency time specified in the **PULSE_LTCY** expires, unless the latch is enabled. If the latch is enabled the source register values will remain frozen until the source register is read.

Table 12. Register 0x22 PULSE_SRC Register (Read Only) and Description

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EA	AxZ	AxY	AxX	DPE	PolZ	PolY	PolX

5.0 Configuring the Tap Detection to an Interrupt Pin

In order to set up the system to route to a hardware interrupt pin, the System Interrupt (bit 3 in Reg 0x2D) must be enabled. The MMA8451, 2, 3Q allows for seven (7) separate types of interrupts. One (1) of these is reserved for tap.

Step 1: Enable the Interrupt in Register 0x2D.

Table 13. 0x2D CTRL_REG4 Register (Read/Write) and Description

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INT_EN_ASLP	INT_EN_FIFO	INT_EN_TRANS	INT_EN_LNDPRT	**INT_EN_PULSE**	INT_EN_FF_MT	—	INT_EN_DRDY

The **INT_EN_PULSE** interrupt enable bit allows the tap function to route its event detection flag to the interrupt controller of the system. The interrupt controller routes the enabled function to either the INT1 or INT2 pin. To enable the Pulse (Tap) block, set bit 3 in Register 0x2D as follows:

Code Example: IIC_RegWrite(0x2D, 0x08);

Step 2: Route the interrupt to INT1 or to INT2. This is done in register 0x2E.

Table 14. 0x2E CTRL_REG5 Register (Read/Write) and Description

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INT_CFG_ASLP	INT_CFG_FIFO	INT_CFG_TRANS	INT_CFG_LNDPRT	**INT_CFG_PULSE**	INT_CFG_FF_MT	—	INT_CFG_DRDY

Note: To route the Tap I to INT1 set bit 3 in register 0x2E. Clear bit 3 to route tap to INT2.

Code Example: IIC_RegWrite(0x2E,0x08); //Set Tap to INT1

5.1 Reading the System Interrupt Status Source Register

In the interrupt source register, the status of the various embedded functions can be determined. The bits that are set (logic ‘1’) indicate which function has asserted an interrupt; conversely, the bits that are cleared (logic ‘0’) indicate which function has not asserted or has de-asserted an interrupt. The interrupts are rising edge sensitive. The bits are set by a low to high transition and are cleared by reading the appropriate interrupt source register.

Table 15. 0x0C INT_SOURCE: System Interrupt Status Register (Read Only)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SRC_ASLP	SRC_FIFO	SRC_TRANS	SRC_LNDPRT	**SRC_PULSE**	SRC_FF_MT	—	SRC_DRDY

6.0 Details Configuring the MMA8451, 2, 3Q for Single and Double Tap

The registers of importance for configuring the MMA8451, 2, 3Q for tap detection are listed below in Table 16.

Table 16. Registers of Importance for Embedded Single and Double Tap Detection

Reg	Name	Definition	Bit 7	Bit 6	Bit 5	Bit4	Bit3	Bit 2	Bit 1	Bit0
0C	INT_SOURCE	Interrupt Status R	SRC_ASLP	SRC_FIFO	SRC_TRANS	SRC_LNDPRT	SRC_PULSE	SRC_FF_MT	—	SRC_DRDY
0F	HP_FILTER_CUTOFF	HP Filter Settings R/W	—	—	Pulse_HPF_Bypass	Pulse_LPF_EN	—	—	SEL1	SEL0
21	PULSE_CFG	Pulse Config R/W	DPA	ELE	ZDPEFE	ZSPEFE	YDPEFE	YSPEFE	XDPEFE	XSPEFE
22	PULSE_SRC	Pulse Source R	EA	AxZ	AxY	AxX	DPE	PolZ	PolY	PolX
23	PULSE_THSX	Pulse X Threshold R/W	0	THSX6	THSX5	THSX4	THSX3	THSX2	THSX1	THSX0
24	PULSE_THSY	Pulse Y Threshold R/W	0	THSY6	THSY5	THSY4	THSY3	THSY2	THSY1	THSY0
25	PULSE_THSZ	Pulse Z Threshold R/W	0	THSZ6	THSZ5	THSZ4	THSZ3	THSZ2	THSZ1	THSZ0
26	PULSE_TMLT	Pulse First Timer R/W	Tmlt7	Tmlt6	Tmlt5	Tmlt4	Tmlt3	Tmlt2	Tmlt1	Tmlt0
27	PULSE_LTCY	Pulse Latency R/W	Ltcy7	Ltcy6	Ltcy5	Ltcy4	Ltcy3	Ltcy2	Ltcy1	Ltcy0
28	PULSE_WIND	Pulse 2nd Window R/W	Wind7	Wind6	Wind5	Wind4	Wind3	Wind2	Wind1	Wind0
2A	CTRL_REG1	Control Reg1 R/W (Interrupt Enable Map)	ASLP_RATE1	ASLP_RATE0	DR2	DR1	DR0	LNOISE	F_READ	ACTIVE
2B	CTRL_REG2	Control Reg2 R/W	ST	RST	qq	SMODS1	SMODS0	SLPE	MODS1	MODS0
2D	CTRL_REG4	Control Reg4 R/W (Interrupt Enable Map)	INT_EN_ASLP	INT_EN_FIFO	INT_EN_TRANS	INT_EN_LNDPRT	INT_EN_PULSE	INT_EN_FF_MT	—	INT_EN_DRDY
2E	CTRL_REG5	Control Reg5 R/W (Interrupt Configuration)	INT_CFG_ASLP	INT_CFG_FIFO	INT_CFG_TRANS	INT_CFG_LNDPRT	INT_CFG_PULSE	INT_CFG_FF_MT	—	INT_CFG_DRDY

6.1 Example Single Tap Only: Normal Mode, No LPF, 400 Hz ODR

Step 1: To set up any configuration make sure to be in Standby Mode.

IIC_RegWrite(0x2A, 0x08); //400 Hz, Standby Mode

Step 2: Enable X and Y and Z Single Pulse

IIC_RegWrite(0x21, 0x15);

Step 3: Set Threshold 1.575g on X and 2.65g on Z

Note: Each step is 0.063g per count

1.575g/0.063g = 25 counts
2.65g/0.063g = 42 counts

IIC_RegWrite(0x23, 0x19); //Set X Threshold to 1.575g

IIC_RegWrite(0x24, 0x19); //Set Y Threshold to 1.575g

IIC_RegWrite(0x25, 0x2A); //Set Z Threshold to 2.65g

Step 4: Set Time Limit for Tap Detection to 50 ms, Normal Mode, No LPF

Data Rate 400 Hz, time step is 0.625 ms

50 ms/0.625 ms = 80 counts
IIC_RegWrite(0x26,0x50); //50 ms

Step 5: Set Latency Time to 300 ms

Data Rate 400 Hz, time step is 1.25 ms
300 ms/1.25 ms = 240 counts

IIC_RegWrite(0x27,0xF0); //300 ms

Step 6: Route INT1 to System Interrupt

IIC_RegWrite(0x2D, 0x08); //Enable Pulse Interrupt Block in System CTRL_REG4

IIC_RegWrite(0x2E, 0x08); //Route Pulse Interrupt Block to INT1 hardware Pin CTRL_REG5

Step 7: Put the device in Active Mode

CTRL_REG1_Data = IIC_RegRead(0x2A); //Read out the contents of the register

CTRL_REG1_Data| = 0x01; //Change the value in the register to Active Mode.

IIC_RegWrite(0x2A, CTRL_REG1_Data); //Write in the updated value to put the device in Active Mode

6.2 Double Tap Only: Low Power Mode, No LPF, 400 Hz ODR

Step 1: To set up any configuration make sure to be in Standby Mode.

IIC_RegWrite(0x2A, 0x08); //400 Hz, Standby Mode

Step 2: Enable X, Y and Z Double Pulse with DPA = 0 no double pulse abort

IIC_RegWrite(0x21, 0x2A);

Step 3: Set Threshold 3g on X and Y and 5g on Z

Note: Every step is 0.063g

3 g/0.063g = 48 counts
5g/0.063g = 79 counts

IIC_RegWrite(0x23, 0x30); //Set X Threshold to 3g

IIC_RegWrite(0x24, 0x30); //Set Y Threshold to 3g

IIC_RegWrite(0x25, 0x4F); //Set Z Threshold to 5g

Step 4: Set Time Limit for Tap Detection to 60 ms LP Mode

Note: 400 Hz ODR, Time step is 1.25 ms per step

60 ms/1.25 ms = 48 counts

IIC_RegWrite(0x26,0x30); //60 ms

Step 5: Set Latency Time to 200 ms

Note: 400 Hz ODR LPMode, Time step is 2.5 ms per step

00 ms/2.5 ms = 80 counts

IIC_RegWrite(0x27,0x50); //200 ms

Step 6: Set Time Window for second tap to 300 ms

Note: 400 Hz ODR LP Mode, Time step is 2.5 ms per step

300 ms/2.5 ms = 120 counts

IIC_RegWrite(0x28,0x78); //300 ms

Step 7: Route INT1 to System Interrupt

IIC_RegWrite(0x2D, 0x08); //Enable Pulse Interrupt in System CTRL_REG4

IIC_RegWrite(0x2E, 0x08); //Route Pulse Interrupt to INT1 hardware Pin CTRL_REG5

Step 8: Set the device to Active Mode

CTRL_REG1_Data = IIC_RegRead(0x2A); //Read out the contents of the register

CTRL_REG1_Data| = 0x01; //Change the value in the register to Active Mode.

IIC_RegWrite(0x2A, CTRL_REG1_Data); //Write in the updated value to put the device in Active Mode

6.3 Single Tap and Double Tap LP Mode, 200 Hz ODR

Step 1: Go to Standby Mode to change configuration settings.

IIC_RegWrite(0x2A,0x10); //200 Hz, Standby Mod**e**

Step 2: Enable X, Y, Z Single Pulse and X, Y and Z Double Pulse with DPA = 0 no double pulse abort

IIC_RegWrite(0x21, 0x3F);

Step 3: Set Threshold 2g on X and Y and 4g on Z

Note: Every step is 0.063g

2g / 0.063g = 32 counts
4g/ 0.063g = 64 counts

IIC_RegWrite(0x23, 0x20); //Set X Threshold to 2g

IIC_RegWrite(0x24, 0x20); //Set Y Threshold to 2g

IIC_RegWrite(0x25, 0x40); //Set Z Threshold to 4g

Step 4: Set Time Limit for Tap Detection to 60 ms (LP Mode, 200 Hz ODR, No LPF)

Note: 200 Hz ODR LP Mode, Time step is 2.5 ms per step

60 ms /2.5 ms = 24 counts

IIC_RegWrite(0x26,0x18); //60 ms

Step 5: Set Latency Timer to 200 ms

Note: 200 Hz ODR LP Mode, Time step is 5 ms per step

200 ms/ 5 ms = 40 counts

IIC_RegWrite(0x27,0x28); //200 ms

Step 6: Set Time Window for Second Tap to 300 ms

Note: 200 Hz ODR LP Mode, Time step is 5 ms per step

00 ms/5 ms = 60 counts

IIC_RegWrite(0x28,0x3C); //300 ms

Step 7: Route INT1 to System Interrupt

IIC_RegWrite(0x2D, 0x08); //Enable Pulse Interrupt Block in System CTRL_REG4

IIC_RegWrite(0x2E, 0x08); //Route Pulse Interrupt Block to INT1 hardware Pin CTRL_REG5

Step 8: Active Mode

CTRL_REG1_Data = IIC_RegRead(0x2A); //Read out the contents of the register

CTRL_REG1_Data| = 0x01; //Change the value in the register to Active Mode.

IIC_RegWrite(0x2A, CTRL_REG1_Data); //Write in the updated value to put the device in Active Mode

Step 9: Write an Interrupt Service Routine

```
Interrupt void isr_KBI (void)
{
      //clear the interrupt flag
      CLEAR_KBI_INTERRUPT;
      //Determine the source of the interrupt by first reading the system interrupt
      register
      Int_SourceSystem =IIC_RegRead(0x0C);
      //Set up Case Statement to service all the possible interrupts
      if((Int_SourceSystem&0x08)==0x08)
      {
            //Perform an Action since Pulse Flag has been Set
            //Read the Pulse Status Register to clear the system interrupt
            Int_SourcePulse=IIC_RegRead(0x22);
            //Can parse out the data here to perform a specific action
            //based on the flags set on each axis.
      }
}
```

7.0 Writing an Algorithm to Detect Directional Tap with the MMA8451Q

Directional tap detection has various challenges due to the mechanics and vibration and placement of the sensor. The orientation that the object is tapped may not be on the axis where the accelerometer is placed. This can cause challenges in designing a repeatable algorithm for directional tap on all 6 faces of a 3D object. The MMA8451, 2, 3Q does have a directional tap algorithm internally but by using the 32 sample FIFO and the embedded single tap detection the directional information can be extracted for a more flexible algorithm.

7.1 Sampling Rate Requirements for Tap Signatures

In order to detect the full signature of a tap a sample rate of 600 Hz or greater is typically required. The tap signature shown in Figure 2 was sampled at 2 kHz from an analog accelerometer to show the full signature of the tap. By using the embedded tap detection at 800 Hz with the FIFO, the directional information can be extracted without the need for continuous data polling. The tap is detected internally at 1600 Hz, while the directional information is analyzed at 800 Hz flushing the FIFO.

In many applications such as menu selections, typically there is only one or maybe two axes of importance. This simplifies the algorithm. For example, on a remote control to change the channel tapping in one axis in both directions is the only input. For simplified applications such as this the directional tap can work very well. There are larger challenges when all axes are enabled at the same time to reliably detect the axis of interest. There is often a lot of cross coupling due to vibrations and mounting of the accelerometer and also from the variation arising due to how different users tap.

7.2 Procedure for Directional Tap using the MMA8451Q

Step 1: Configure the accelerometer to be in 800 Hz 8g Mode, either Low Power Mode or Normal Mode depending on extra power savings required.

Step 2: Configure the FIFO to be in Trigger Mode so that the data continuously overwrites up to the watermark of 20 samples, allowing 12 samples after the tap event to store the rebound data. Refer to Freescale application note AN4073.

Step 3: Configure the embedded single pulse detection for detecting X, Y, Z single pulse with X, Y = 1.67g and Z = 3.19g. Set the Time Limit to 50 ms for the pulse and set the latency to 500 ms.

Step 4: Set up the FIFO Overflow Detection in the System Interrupt Detection to detect on INT1.

Step 5: Configure the MCU to stay in Sleep Mode until INT1 is asserted if this is desired for additional current savings.

Step 6: When INT1 is asserted while the MCU is in sleep mode, wake the MCU/processor

Step 7: Flush the FIFO data to clear the FIFO interrupt and store the data in a 32 sample buffer. Read the Source Register to determine which axes asserted the tap and clear the tap interrupt.

Step 8: Look at the Max and Min values of each axis whose flag was detected in the pulse source register.

Step 9: Determine the direction: If the first peak is greater in absolute value than the second peak, it is a positive tap, otherwise it is a negative tap.

Step 10: If more than one axis has triggered a valid tap, determine which event occurred first by analyzing the FIFO data.

Step 11: If the taps all occurred at the same time, the axis with the largest value calculating max-min is the axis the event occurred on.

For more in-depth details of developing a robust directional tap algorithm please contact Freescale for further assistance. There is a demo available in the Sensor Toolbox Software.

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

AN4072
Rev. 0
09/2010