AN3308
应用笔记

LIS3DH：MEMS 数字输出运动传感器
超低功耗高性能 3 轴 “nano” 加速度计

前言

本文档介绍了以 LGA 封装提供的低压 3 轴数字量输出线性 MEMS 加速度计。

LIS3DH 是属于 “nano” 系列的超低功耗高性能 3 轴线性加速度计，具有数字 I^2C、 SPI 串行接口标准输出。

器件具有超低功耗工作模式，可实现高级节能、智能睡眠唤醒以及恢复睡眠功能。

LIS3DH 具有 ±2g/±4g/±8g/±16g 的动态用户可选满量程，并能通过 1 Hz 到 5 kHz 的输出数据速率测量加速度。

器件可配置为通过独立的惯性唤醒 / 自由落体事件以及通过器件自身的位置生成中断信号。中断发生器的阈值和时序可由终端用户动态设定。

也可通过可自动编程的睡眠唤醒和恢复睡眠功能提高节能效率。

LIS3DH 集成了 32 级先进先出 (FIFO) 缓冲区供用户存储数据，从而可减少主机处理器的干预。

LIS3DH 采用纤薄的小型塑料平面网格阵列封装 (LGA)，可确保在更大的温度范围 （-40 °C 至 +85 °C）内正常工作。

SMD 封装的尺寸和重量都超小，非常适用于手持式便携应用 （比如手机和 PDA）或者任何需要减小封装大小和重量的其他应用。

目录

表格索引

图片索引

1 寄存器表

表 1. 寄存器表

寄存器名	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	位 0
STATUS_REG_AUX	07h	321OR	3OR	2OR	1OR	321DA	3DA	2DA	1DA
OUT_ADC1_L	08h	A1D7	A1D6	A1D5	A1D4	A1D3	A1D2	A1D1	A1D0
OUT_ADC1_H	09h	A1D15	A1D14	A1D13	A1D12	A1D11	A1D10	A1D9	A1D8
OUT_ADC2_L	0Ah	A2D7	A2D6	A2D5	A2D4	A2D3	A2D2	A2D1	A2D0
OUT_ADC2_H	0Bh	A2D15	A2D14	A2D13	A2D12	A2D11	A2D10	A2D9	A2D8
OUT_ADC3_L	0Ch	A3D7	A3D6	A3D5	A3D4	A3D3	A3D2	A3D1	A3D0
OUT_ADC3_H	0Dh	3A3D15	C2	A3D13	A3D12	A3D11	A3D10	A3D9	A3D8
INT_COUNTER_REG	0Eh	C7	C6	C5	C4	C3	C2	C1	C0
WHO_AM_I	0Fh	0	0	1	1	0	0	1	1
TEMP_CFG_REG	1Fh	ADC_PD	TEMP_EN	0	0	0	0	0	0
CTRL_REG1	20h	ODR3	ODR2	ODR1	ODR0	LPen	Zen	Yen	Xen
CTRL_REG2	21h	HPM1	HPM0	HPCF2	HPCF1	FDS	HPCLICK	HPIS2	HPIS1
CTRL_REG3	22h	I1_CLICK	I1_AOI1	0	I1_DRDY1	I1_DRDY2	I1_WTM	I1_OVERRUN	-
CTRL_REG4	23h	BDU	BLE	FS1	FS0	HR	ST1	ST0	SIM
CTRL_REG5	24h	BOOT	FIFO_EN	-	-	LIR_INT1	D4D_INT1	0	0
CTRL_REG6	25h	I2_CLICKen	I2_INT1	0	BOOT_I1	0	-	H_LACTIVE	-
参考	26h	REF7	REF6	REF5	REF4	REF3	REF2	REF1	REF0
STATUS_REG2	27h	ZYXOR	ZOR	YOR	XOR	ZYXDA	ZDA	YDA	XDA
OUT_X_L	28h	XD7	XD6	XD5	XD4	XD3	XD2	XD1	XD0
OUT_X_H	29h	XD15	XD14	XD13	XD12	XD11	XD10	XD9	XD8
OUT_Y_L	2Ah	YD7	YD6	YD5	YD4	YD3	YD2	YD1	YD0
OUT_Y_H	2Bh	YD15	YD14	YD13	YD12	YD11	YD10	YD9	YD8

表 1. 寄存器表 （续）

寄存器名	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	位 0
OUT_Z_L	2Ch	ZD7	ZD6	ZD5	ZD4	ZD3	ZD2	ZD1	ZD0
OUT_Z_H	2Dh	ZD15	ZD14	ZD13	ZD12	ZD11	ZD10	ZD9	ZD8
FIFO_CTRL_REG	2E	FM1	FM0	TR	FTH4	FTH3	FTH2	FTH1	FTH0
FIFO_SRC_REG	2F	WTM	OVRN_FIFO	-	FSS4	FSS3	FSS2	FSS1	FSS0
INT1_CFG	30h	AOI	6D	ZHIE	ZLIE	YHIE	YLIE	XHIE	XLIE
INT1_SRC	31h	-	IA	ZH	ZL	YH	YL	XH	XL
INT1_THS	32h	0	THS6	THS5	THS4	THS3	THS2	THS1	THS0
INT1_DURATION	33h	0	D6	D5	D4	D3	D2	D1	D0
CLICK_CFG	38h	-		ZD	ZS	YD	YS	XD	XS
CLICK_SRC	39h	-	IA	DCLICK	SCLICK	符号	Z	y	x
CLICK_THS	3Ah	-	Ths6	Ths5	Ths4	Ths3	Ths2	Ths1	Ths0
TIME_LIMIT	3Bh	-	TLI6	TLI5	TLI4	TLI3	TLI2	TLI1	TLI0
TIME_LATENCY	3Ch	TLA7	TLA6	TLA5	TLA4	TLA3	TLA2	TLA1	TLA0
TIME_WINDOW	3Dh	TW7	TW6	TW5	TW4	TW3	TW2	TW1	TW0
ACT_THS	3Eh	-	ATHS6	ATHS5	ATHS4	ATHS3	ATHS2	ATHS1	ATHS0
INACT_DUR	3Fh	ADUR7	ADUR6	ADUR5	ADUR4	ADUR3	ADUR2	ADUR1	ADUR0

2 工作模式

LIS3DH 提供三种不同的工作模式，分别是掉电模式、正常模式和低功耗模式。正常模式可确保达到更高的分辨率，而低功耗模式可以进一步减少电流消耗。

施加电源后，LIS3DH 会执行 5 ms 的启动程序来加载修调参数。启动完成后，器件会自动配置为掉电模式。

参考 LIS3DH 数据手册， CTRL_REG1 的输出数据速率 (ODR) 和低功耗使能 (LPen) 位以及 CTRL_REG4 的 HR 位用于选择工作模式 （掉电模式、正常模式和低功耗模式）以及输出数据速率 （*表 2* 和*表 3*）。

表 2. 工作模式选择

工作模式	CTRL_REG1[3] （LPen 位）	CTRL_REG4[3] （HR 位）	BW [Hz]	导通时间 [ms]
低功耗模式	1	0	ODR/2	1
正常模式	0	1	ODR/9	7/ODR

表 3. 数据速率配置

ODR3	ODR2	ODR1	ODR0	功率模式选择
0	0	0	0	掉电模式
0	0	0	1	正常 / 低功耗模式 (1 Hz)
0	0	1	0	正常 / 低功耗模式 (10 Hz)
0	0	1	1	正常 / 低功耗模式 (25 Hz)
0	1	0	0	正常 / 低功耗模式 (50 Hz)
0	1	0	1	正常 / 低功耗模式 (100 Hz)
0	1	1	0	正常 / 低功耗模式 (200 Hz)
0	1	1	1	正常 / 低功耗模式 (400 Hz)
1	0	0	0	低功耗模式 (1.5 kHz)
1	0	0	1	正常 (1.250 kHz)/ 低功耗模式 (5 kHz)

表 4 列出了不同工作模式下的典型功耗值。

表 4. 功耗 (μA)

ODR(Hz)	正常模式 (mA)	低功耗模式 (mA)
掉电	0.4	0.4
1	2	2
10	4	3
25	6	4
50	11	6

表 4. 功耗 (μA) （续）

ODR(Hz)	正常模式 (mA)	低功耗模式 (mA)
100	20	10
200	38	18
400	73	36
1250	185	-
1600	-	99
5000	-	184

2.1 掉电模式

器件处于掉电模式时，器件内部的全部内部块几乎都会关闭，以最大限度地降低功耗。但数字接口 （I^2C 和 SPI）仍处于激活状态，以便与器件之间进行通信。配置寄存器内容会保留下来，输出数据寄存器不会更新，因此在进入掉电模式之前会保留上次在存储器中采样的数据。

2.2 正常模式

在正常模式下，会以通过 DR 位选择的数据速率 (ODR) 生成数据，数据会用于通过 CTRL_REG1 的 Zen、 Yen 和 Xen 位使能的轴。为已禁用的轴生成的数据为 00h。

数据中断生成有效，通过 INT1_CFG 寄存器进行配置。

2.3 低功耗模式

正常模式可确保达到更高的分辨率，而低功耗模式可以进一步减少电流消耗。

在低功耗模式下，会以通过 DR 位选择的数据速率 (ODR) 生成数据，数据会用于通过 CTRL_REG1 的 Zen、 Yen 和 Xen 位使能的轴。为已禁用的轴生成的数据为 00h。

数据中断生成有效，通过 INT1_CFG 寄存器进行配置。

2.4 切换模式时序

切换模式时间如*表 1* 所示。

表 5. 切换模式时序

起始模式	目标模式	导通时间 - TYP (ms)
掉电	低功耗	2/ODR
掉电	正常	7/ODR
正常	掉电	0
低功耗	掉电	0
低功耗	正常	7/ODR
正常	低功耗	2/ODR

3 启动序列

器件通电后，会自动将校准系数从嵌入式闪存下载到内部寄存器。启动程序完成后，也就是大约 5 毫秒后，器件会自动进入掉电模式。要导通器件并采集加速度数据，需要通过 CTRL_REG1 选择其中一种工作模式并至少使能其中一个轴。

可使用下列通用序列对器件进行配置：

1. 写 CTRL_REG1
2. 写 CTRL_REG2
3. 写 CTRL_REG3
4. 写 CTRL_REG4
5. 写 CTRL_REG5
6. 写 CTRL_REG6
7. 写入参考
8. 写 INT1_THS
9. 写 INT1_DUR
10. 写 INT1_CFG
11. 写 CTRL_REG5

3.1 读取加速度数据

3.1.1 使用状态寄存器

应对为器件提供的 STATUS_REG 进行轮询，以检查何时有可用的新数据集。读取程序如下：

1. 读取 STATUS_REG
2. 如果 STATUS_REG(3) = 0，则转至 1
3. 如果 STATUS_REG(7) = 1，说明一些数据已被重写
4. 读 OUTX_L
5. 读 OUTX_H
6. 读 OUTY_L
7. 读 OUTY_H
8. 读 OUTZ_L
9. 读 OUTZ_H
10. 数据处理
11. 转至 1

第 3 步中进行的检查可了解读取速率是否适合数据生产速率。如果一个或多个加速度样本已被新数据覆盖，由于读取速率过慢的原因， STATUS_REG 的 ZYXOR 位会置 1。

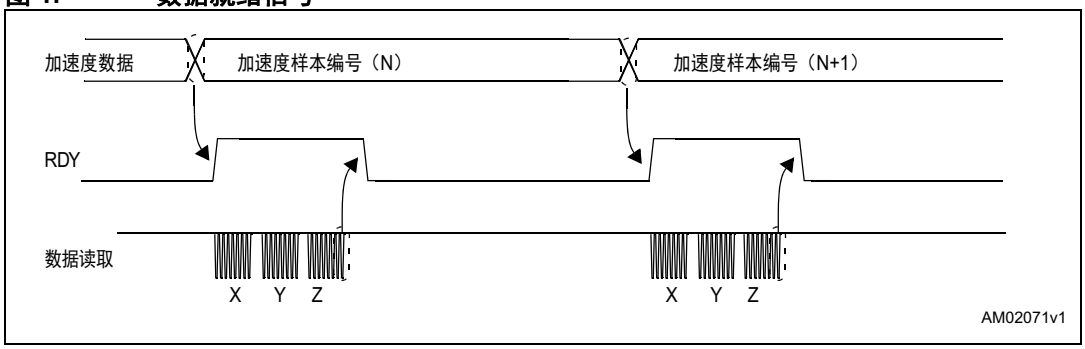
如果器件内存在的所有数据均已被读取，同时尚未生成新数据，上溢位会自动清零。

3.1.2 使用数据就绪 (DRY) 信号

可将器件配置为通过一个硬件信号决定何时有新的测量数据集可供读取。此信号由 STATUS_REG 的 XYZDA 位表示。可通过将 CTRL_REG3 的 I1_DRDY1 位置 1、通过 CTRL_REG6 的 H_LACTIVE 位将其极性设为低电平有效或高电平有效的方式将信号驱动到 INT1 引脚。

新的加速度数据集已生成并可供读取时，数据就绪信号会上升为 1。所有已使能通道的数据高位部分 （29h、 2Bh、 2Dh）均读取完毕后，中断会复位。

图 1. 数据就绪信号



3.1.3 使用块数据 (BDU) 功能

如果加速度数据的读取速度特别慢并且不能通过 STATUS_REG 中存在的 XYZDA 位或通过 RDY 信号进行同步 （或者不需要同步），则强烈建议将 CTRL_REG4 中的 BDU （块数据更新）位置 1。

此功能可避免读取与其它样本相关联的数值 （加速度数据的最高有效部分和最低有效部分）。特别是在 BDU 被激活的情况下，与每条通道相关联的数据寄存器始终会包含由器件生成的最新加速度数据，但如果发起了对给定寄存器对 （即 OUT_X_H 和 OUT_X_L、OUT_Y_H 和 OUT-Y_L、 OUT_Z_H 和 OUT_Z_L）的读取，读取数据的 MSB 和 LSB 部分之前，都会禁止刷新该寄存器对。

注： BDU 仅会确保已同时对 OUT_X(Y, Z)_L 和 OUT_X(X,Z)_H 进行采样。例如，如果读取速度过慢，可能会读取在 T1 采样的 X 和 Y 以及在 T2 采样的 Z。

3.2 了解加速度数据

测得的加速度数据会发送至 OUTX_H、 OUTX_L、 OUTY_H、 OUTY_L、 OUTZ_H 和 OUTZ_L 寄存器。这些寄存器分别包含作用于 X、 Y 和 Z 轴的加速度信号的最高有效部分和最低有效部分。

X (Y, Z) 通道的完整加速度数据是由 OUTX_H & OUTX_L（OUTY_H & OUTY_L、OUTZ_H & OUTZ_L）共同提供的，表示为 2 的补码。

3.2.1 数据对齐 (Data alignment)

加速度数据表示为 16 位数，向左对齐 （低四位无效）。

3.2.2 大端 / 小端选择

LIS3DH 允许交换加速度寄存器低位部分和高位部分的内容 （即交换 OUTX_H 与 OUTX_L 的内容），以便符合小端和大端数据表示法的要求。

“ 小端模式 ” 表示数字的低位字节存储在存储器的最低地址中，高位字节存储在最高地址中。（小端模式优先）。该模式相当于 CTRL_REG4 中的 BLE 位复位为 0 （默认配置）。

相反， “ 大端模式 ” 表示数字的高位字节存储在存储器的最低地址中，低位字节存储在最高地址中。

3.2.3 加速度数据示例

表 6 提供的几个基本示例中，会在器件受给定加速度影响的情况下读取数据寄存器中的数据。表中列出的数值均假定器件已进行准确校准 （也就是没有偏移、没有增益误差），并且会真实显示 BLE 位的影响。

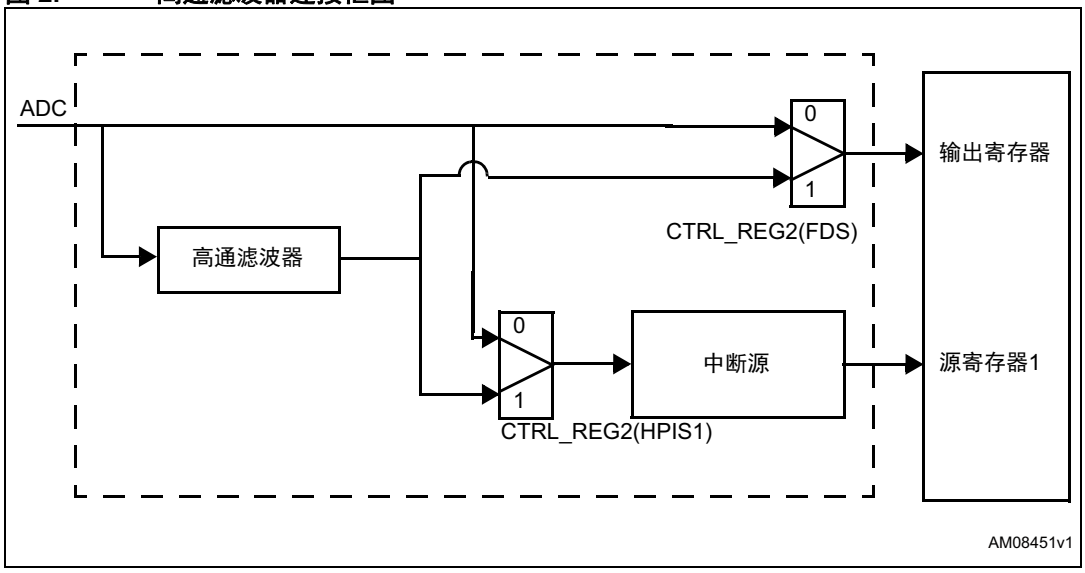
表 6. 输出数据寄存器内容与加速度对比 (FS = 2 *g*)

加速度值	BLE = 0		BLE = 1	
	寄存器地址			
	28h	29h	28h	29h
0 g	00h	00h	00h	00h
350 mg	E0h	15h	15h	E0h
1 g	00h	04h	04h	00h
-350 mg	20h	EAh	EAh	20h
-1g	00h	C0h	C0h	00h

4 高通滤波器

LIS3DH 提供的嵌入式高通滤波功能可轻松去除测得加速度的 DC 分量。如*图 2* 中所示，通过配置 CTRL_REG2 的 FDS、 HPen1 和 HPen2 位，可以将滤波器独立应用于输出数据和 / 或中断数据。这意味着可以在中断生成作用于未滤波数据的同时获得已进行滤波的数据。

图 2. 高通滤波器连接框图



4.1 滤波器配置

参照*表 7*，高通滤波器可具有两种工作模式：

表 7. 高通滤波器模式配置

HPM1	HPM0	高通滤波器模式
0	0	正常模式 （复位读取 HP_RESET_FILTER）
0	1	滤波参考信号
1	0	正常模式
1	1	中断事件自动复位

高通滤波器的带宽取决于所选 ODR 以及 CTRL_REG2 的 HPCFx 位的设置。高通滤波器截止频率 (f_t) 如*表 8* 中所示。

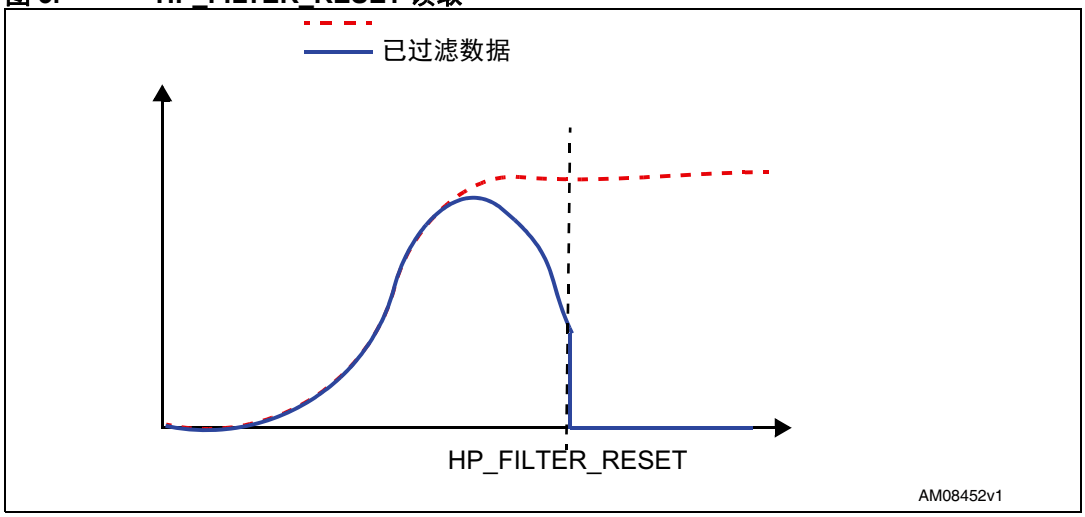
表 8. 低功耗模式 - 高通滤波器截止频率 [Hz]

HPC	f_t [Hz] @ 1Hz	f_t [Hz] @ 10Hz	f_t [Hz] @ 25Hz	f_t [Hz] @ 50Hz	f_t [Hz] @ 100Hz	f_t [Hz] @ 200Hz	f_t [Hz] @ 400Hz	f_t [Hz] @ 1.6 kHz	f_t [Hz] @ 5 kHz
00	0.02	0.2	0.5	1	2	4	8	32	100
01	0.008	0.08	0.2	0.5	1	2	4	16	50
10	0.004	0.04	0.1	0.2	0.5	1	2	8	25
11	0.002	0.02	0.05	0.1	0.2	0.5	1	4	12

4.1.1 正常模式

在该配置下，可复位高通滤波器，读取参考寄存器，从而可立即删除加速度的 DC 分量。

图 3. HP_FILTER_RESET 读取



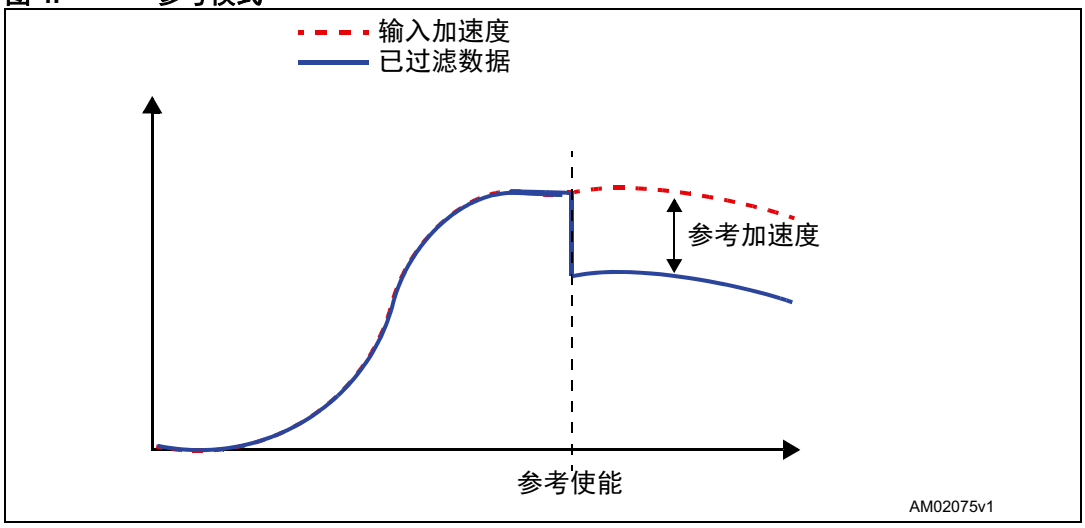
4.1.2 参考模式

在该配置下，输出数据会计算为输入加速度与参考寄存器内容之差。该寄存器表示为 2 的补码形式，这些 7 位寄存器的 1 LSB 的值取决于所选满量程 (*表 9*)。

表 9. 参考模式 LSB 值

满量程	参考模式 LSB 值 (mg)
2	~16
4	~31
8	~63

图 4. 参考模式

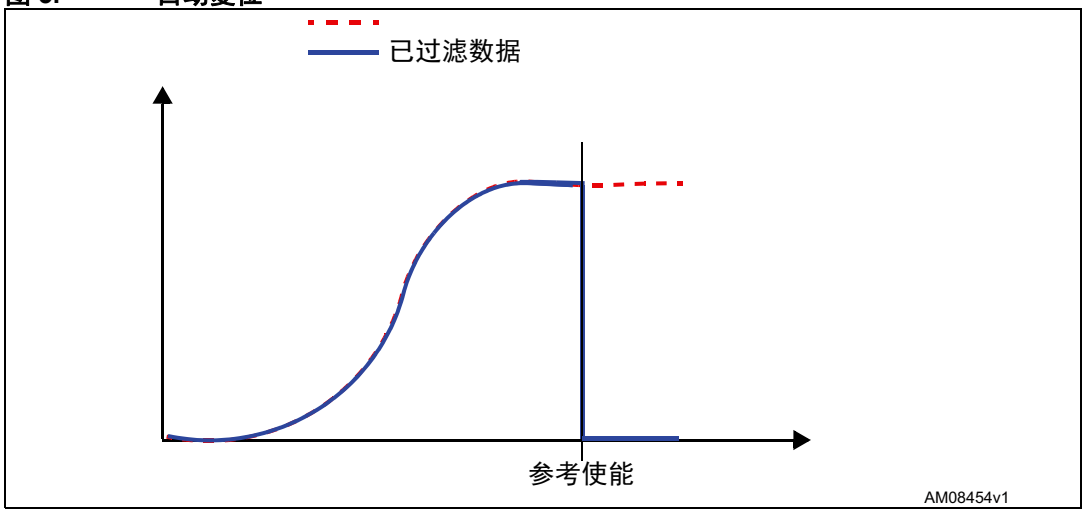


4.1.3 自动复位

在该配置下，发生配置的中断事件时，滤波器会自动复位。但会立即使用 HP_RESET 将滤波器置位。

注： *用于复位滤波器的 XYZ 数据集是中断后的数据集。*

图 5. 自动复位



5 中断生成

LIS3DH 中断信号可用作自由落体、唤醒、 6D 和 4D 定向检测以及点击检测。这些信号可驱动到两个中断引脚 （INT1 和 INT2）。

5.1 中断引脚配置

器件提供的两个引脚可激活为生成数据就绪信号或中断信号。引脚功能是通过 CTRL_REG3(22h) 和 CTRL_REG6(25h) 选择的。请参考*表 10* 和*表 11* 以及*图 6* 中提供的框图。

表 10. CTRL_REG3 寄存器

I1_CLICK	I1_INT1	0	I1_DRDY1	-	I1_WTM	I1_OVERRUN	-

表 11. CTRL_REG3 说明

I1_CLICK	INT1 上的 CLICK 中断。默认值 0。 （0：禁用； 1：启用）
I1_INT1	INT1 引脚上的中断发生器 1。默认值 0。 （0：禁用； 1：启用）
I1_DRDY1	INT1 上的 DRDY1 中断。默认值 0。 （0：禁用； 1：启用）
I1_WTM	INT1 上的 FIFO 水印中断。默认值 0。 （0：禁用； 1：启用）
I1_OVERRUN	INT1 上的 FIFO 上溢中断。默认值 0。 （0：禁用； 1：启用）

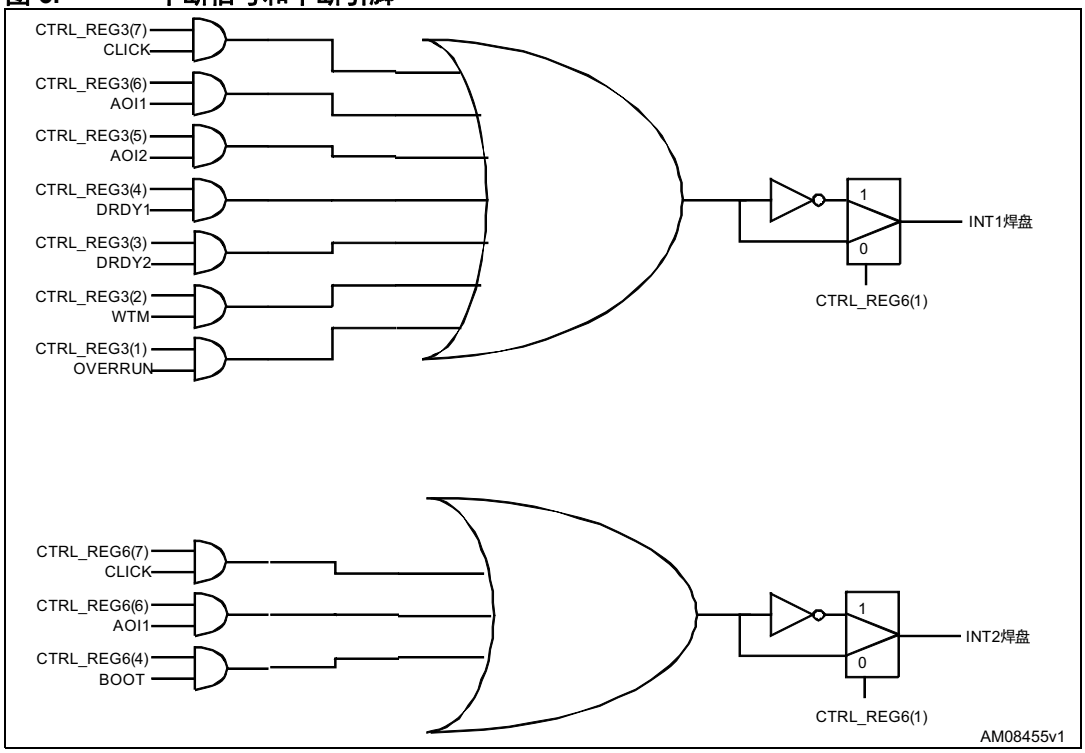
表 12. CTRL_REG6 寄存器

I2_CLICKen	I2_INT1	0	BOOT_I2	0	-	H_LACTIVE	-

表 13. CTRL_REG6 寄存器

I2_CLICKen	INT2 上的 CLICK 中断。默认值 0。 （0：禁用； 1：启用）
I2_INT2	INT2 引脚上的中断发生器 1。默认值 0。 （0：禁用； 1：启用）
BOOT_I2	INT2 上的 BOOT 状态。默认值 0。 （0：禁用； 1：启用）
HL_ACTIVE	0：中断高电平有效； 1：中断低电平有效

图 6. 中断信号和中断引脚



6 惯性中断

LIS3DH 可提供两个惯性中断信号，并可通过多种方式对这两个信号进行定制化。中断生成行为中涉及到的寄存器是 INT1_CFG、 INT1_THS 和 INT1_DURATION。

表 14. 中断模式配置

AOI	6D	中断模式
0	0	中断事件的 OR （或运算）组合
0	1	6 方向运动识别
1	0	中断事件的 AND （与运算）组合
1	1	6 方向位置识别

中断条件满足时，会生成中断信号，通过读取 INT1_SRC 寄存器，可以了解发生了什么情况。

6.1 时长

持续时间寄存器的内容会设置要识别的中断时间的最短持续时间。持续时间步数和最大值取决于选择的 ODR。

持续时间的测量单位为 N/ODR，其中， N 是持续时间寄存器的内容， ODR 是 50、 100、 400、 1000 Hz。

表 15. 正常模式下的持续时间 LSB 值

ODR (Hz)	持续时间 LSB 值 (ms)
1	1000
10	100
25	40
50	20
100	10
200	5
400	2.5
1000	1
1344	0.744
1620	0.617

6.2 门限

阈值寄存器定义了中断生成电路使用的参考加速度。这些 7 位寄存器的 1 LSB 的值取决于所选满量程 (*表 16*)。

表 16. 阈值 LSB 值

满量程	阈值 LSB 值 (mg)
2	~16
4	~31
8	~63
16	~125

6.3 自由落体和唤醒中断

LIS3DH 中断信号可当做自由落体和唤醒中断。中断条件满足时，会生成中断信号，通过读取 INT1_SRC 寄存器，可以了解发生了什么情况。

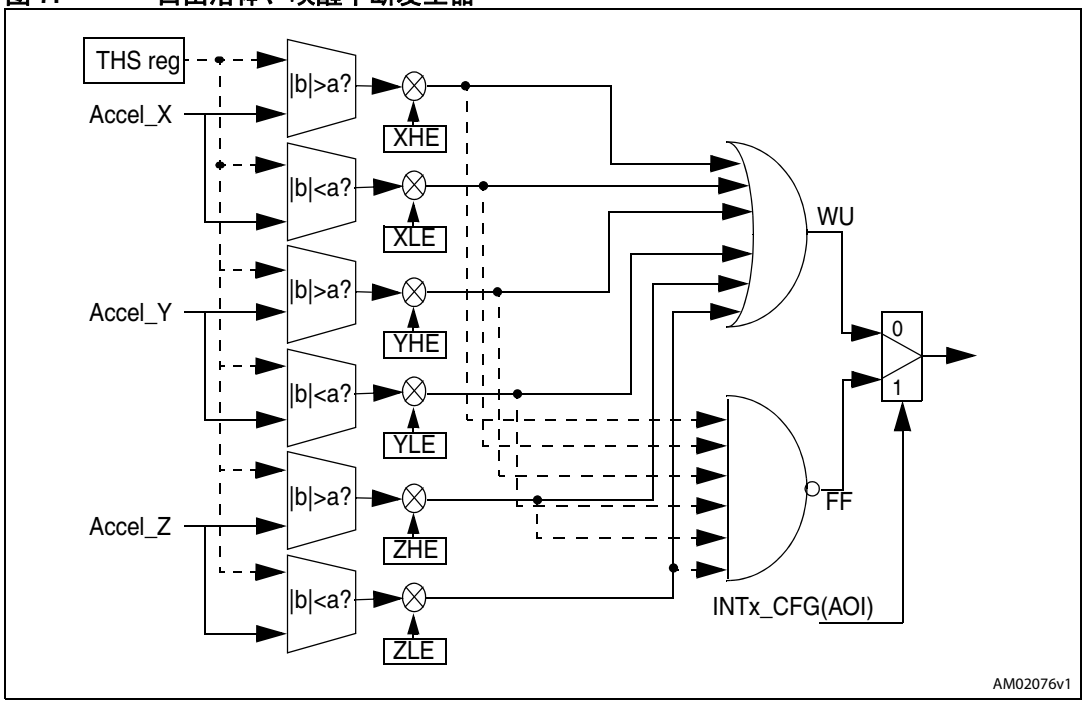
自由落体信号 (FF) 和唤醒信号 (WU) 中断生成块在*图 7* 中表示。

FF 或 WU 中断生成是通过 INT1_CFG 寄存器中的 AOI 位选择的。如果 AOI 位为 “0”，来自轴 （通过 INT1_CFG 寄存器使能）比较器的信号会输入到逻辑 OR 中。在这种情况下，当至少有一个已使能轴超出写入到 INT1_THS 寄存器模块中的阈值时，会生成中断。如果 AOI 位为 “1”，来自比较器的信号会进入 “NAND” 端口。在这种情况下，仅当所有已使能轴都超过写入到 INT1_THS 寄存器中的阈值时，才会生成中断信号。

CTRL_REG3 的 LIR1 位可用于决定是否必须锁存中断请求。如果 LIR1 位为 “0”，当中断条件满足时，中断信号会变为高电平，如果中断条件不再满足，中断信号会立即恢复低电平。否则，如果 LIR1 位为 “1”，如果中断条件适用，即使条件恢复为非中断状态，中断信号也会保持高电平，直至对 INT1_SRC 寄存器执行读取操作。

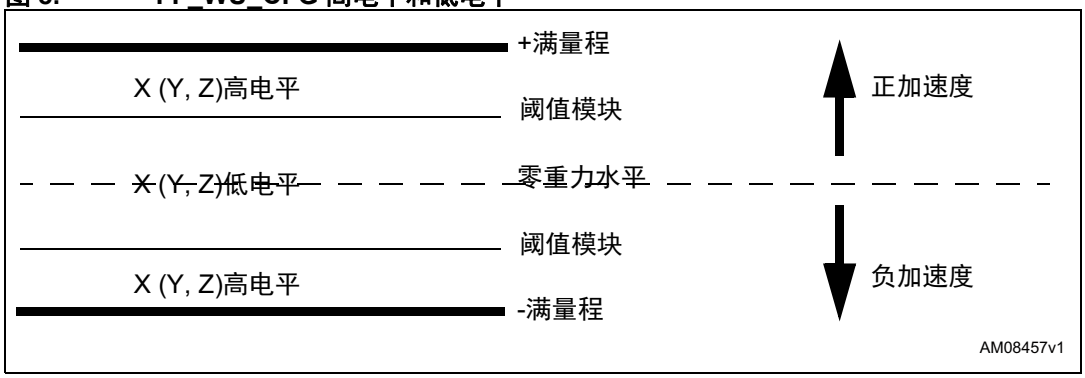
INT1_CFG 寄存器的 ZHIE、 ZLIE、 YHIE、 YLIE、 XHIE 和 HLIE 位可决定必须对哪一条轴执行中断决策、并可决定必须沿哪一方向传递阈值才能生成中断请求。

图 7. 自由落体、唤醒中断发生器



系统检测任何自由落体或惯性唤醒事件所使用的阈值模块是由 INT1_THS 寄存器定义的。阈值表示为 7 位无符号数字，并且绕零重力水平对称。如果 X (Y, Z) 通道的无符号加速度值大于 INT1_THS， XH (YH, ZH) 为真。同样，如果 X (Y, Z) 通道的无符号加速度值小于 INT1_THS， XL (YL, ZL) 低电平为真。更多详情，请参阅*图 8*。

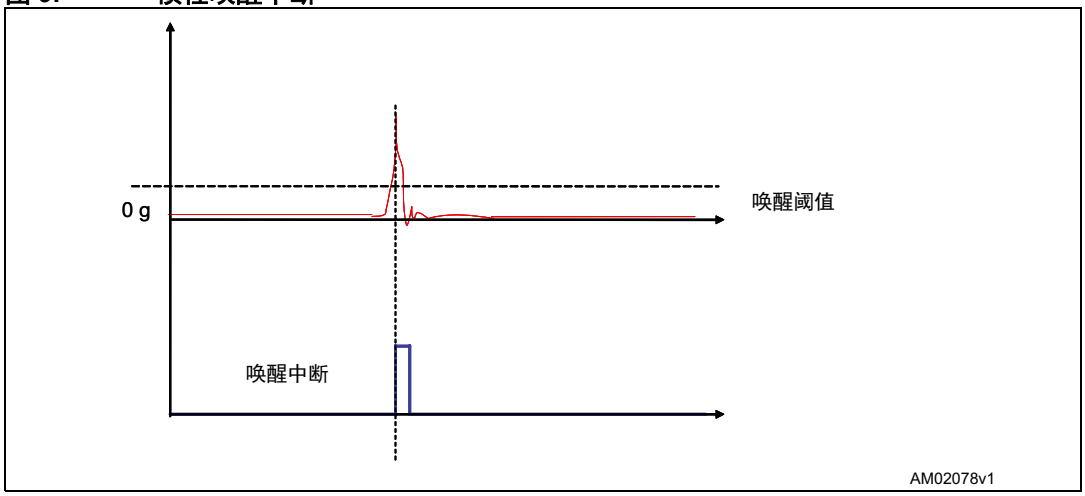
图 8. FF_WU_CFG 高电平和低电平



6.3.1 惯性唤醒

唤醒中断是指 INT1_CTRL 寄存器的特定配置，该配置允许在已配置轴的加速度超过定义的阈值 (*图 9*) 时生成中断。

图 9. 惯性唤醒中断



6.3.2 绕过高通滤波器

本段介绍的基本算法演示了惯性唤醒功能的实际应用。下列代码会将器件配置为可识别出沿 X 或 Y 轴方向的绝对加速度超过预设阈值 （本例中使用的阈值为 250 mg）的情况。触发中断的事件会锁存在设备内，会使用 INT1 引脚指示发生该事件。

1	将 A7h 写入 CTRL_REG1	// 启动传感器并使能 X、 Y 和 Z // ODR = 100 Hz
2	将 00h 写入 CTRL_REG2	// 高通滤波器已禁用
3	将 40h 写入 CTRL_REG3	// 中断被驱动到 INT1 焊盘
4	将 00h 写入 CTRL_REG4	// FS = 2 g
5	将 08h 写入 CTRL_REG5	// 中断已锁存
6	将 10h 写入 INT1_THS	// 阈值 = 250 mg
7	将 00h 写入 INT1_DURATION	// 持续时间 = 0
8	将 0Ah 写入 INT1_CFG	// 使能 XH 和 YH 中断生成
9	轮询 INT1 焊盘；如果 INT1=0，则转至 8	// 轮询等待唤醒事件的 // RDY/INT 引脚
10	读 INT1_SRC	// 返回触发了中断 // 的事件
11	（发生了唤醒事件；在此插入您的代码）	// 事件处理
12	转至 8	

6.3.3 使用高通滤波器

以下代码中的基本例程展示了对已进行高通滤波的数据执行的惯性唤醒功能的实际应用。器件被配置为可识别出施加到 X、 Y 或 Z 轴的加速度的高频分量超过预设阈值 （本例中使用的阈值为 250 mg）的情况。
触发中断的事件会锁存在设备内，会使用 INT1 引脚指示发生该事件。

1	将 A7h 写入 CTRL_REG1	// 启动传感器，使能 X、 Y 和 Z // ODR = 100 Hz
2	将 09h 写入 CTRL_REG2	// 高通滤波器由数据和中断 1 使能
3	将 40h 写入 CTRL_REG3	// 中断被驱动到 INT1 焊盘
4	将 00h 写入 CTRL_REG4	// FS = 2 *g*
5	将 08h 写入 CTRL_REG5	// 中断已锁存
6	将 10h 写入 INT1_THS	// 阈值 = 250 mg
7	将 00h 写入 INT1_DURATION	// 持续时间 = 0

8	读 HP_FILTER_RESET	// 进行虚拟读取，将高通滤波器强制设为
		// 当前加速度值
		// （也就是设置参考加速度 / 倾斜值）

9	将 2Ah 写入 INT1_CFG	// 配置所需唤醒事件
10	轮询 INT1 焊盘；如果 INT1=0，则转至 9	// 轮询等待唤醒事件的 // RDY/INT 引脚
11	（发生了唤醒事件；在此插入您的代码）	// 事件处理
12	读 INT1_SRC	// 返回触发了中断 // 中断并清除中断
13	（在此插入您的代码）	// 事件处理
14	转至 9	

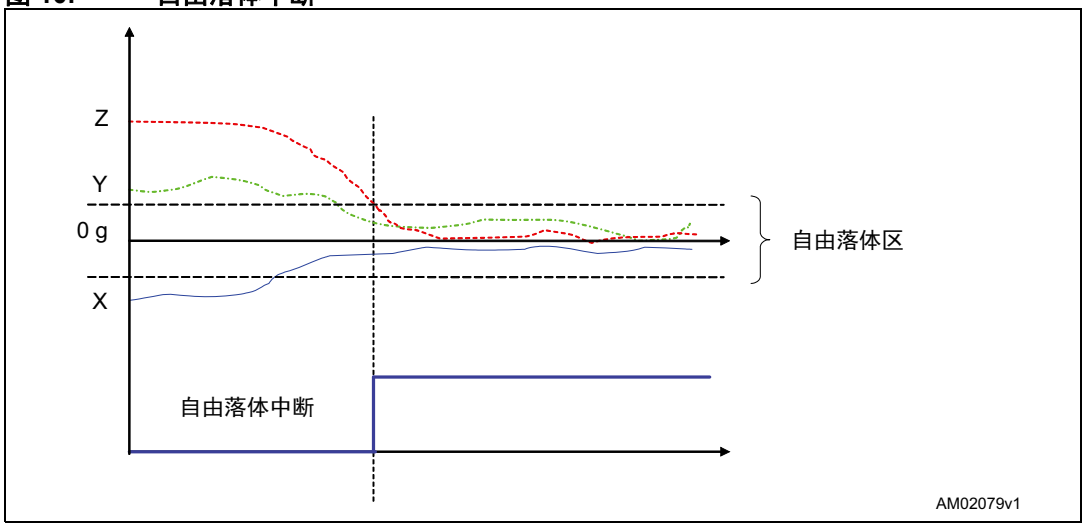
在第 8 步中，会对 HP_FILTER_RESET 寄存器执行虚拟读取，以便设置器件执行阈值比较时所参照的当前 / 参考加速度 / 倾斜状态。

可根据需要随时执行虚拟读取，以便将器件的方向 / 倾斜设为参考状态，无需等待滤波器稳定下来。

6.4 自由落体检测

自由落体检测是指利用 INT1_CTRL 寄存器的特定配置来识别器件是否在进行自由落体运动：沿所有轴测量的加速度均变为零。在实际情况下“自由落体区”定义为零重力水平附近，在该区域中，所有加速度都足够小，可生成中断 (*图 10*)。

图 10. 自由落体中断



本段介绍了使用自由落体检测的基础知识。还特别介绍了以下将器件配置为检测自由落体事件并发信号指示此类事件的软件例程：

1	将 A7h 写入 CTRL_REG1	// 启动传感器，使能 X、 Y 和 Z // ODR = 100 Hz
2	将 00h 写入 CTRL_REG2	// 高通滤波器已禁用
3	将 40h 写入 CTRL_REG3	// 中断被驱动到 INT1 焊盘
4	将 00h 写入 CTRL_REG4	// FS = 2 g
5	将 08h 写入 CTRL_REG5	// 中断已锁存
6	将 16h 写入 INT1_THS	// 将自由落体阈值设为 350 mg
7	将 03h 写入 INT1_DURATION	// 设置最短事件持续时间
8	将 95h 写入 INT1_CFG	// 配置自由落体识别
9	轮询 INT1 焊盘；如果 INT1=0，则转至 10	// 轮询等待自由落体事件的 INT1 引脚
10	（发生了自由落体事件；在此插入您的代码）	// 事件处理
11	读取 INT1_SRC 寄存器	// 清除中断请求
12	转至 9	

示例代码利用设定为 350 mg 的阈值进行自由落体识别，并通过硬件信号 INT1 通知自由落体事件。在第 7 步， INT1_DURATION 寄存器像这样进行了配置，从而可忽略短于 3/DR = 3/100 ~= 30 ms 的事件，以避免假检测现象的发生。

发生自由落体事件后，对 INT1_SRC 寄存器的读取操作会清空请求，器件会准备好识别其他事件。

7 6D/4D 定向检测

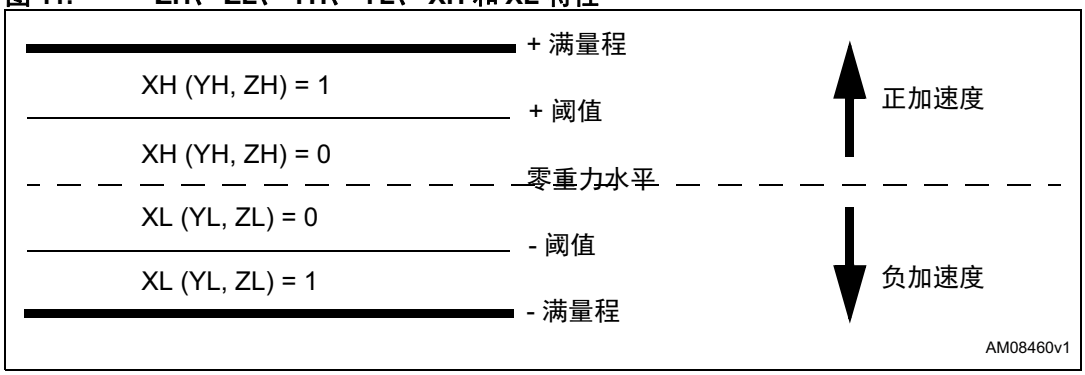
LIS3DH 的高级功能可检测器件在空间中的方向，从而可轻松为手持设备实施节能程序，并可实现图像自动旋转。

7.1 6D 定向检测

6D 定向方向功能可通过 INT1_CFG 寄存器的 AOI 和 6D 位启用。配置为实现 6D 功能时，INT1_SRC 的 ZH、ZL、YH、YL、XH 和 XL 位会提供关于加速度值以及加速度信号的相关信息，当加速度值大于阈值时，会生成中断。更具体地说：

- 当感应到的加速度大于正方向阈值时， ZH (YH, XH) 为 1
- 当感应到的加速度大于负方向阈值时， ZL (YL, XL) 为 1

图 11. ZH、 ZL、 YH、 YL、 XH 和 XL 特性



6D 方向功能有两种可行的配置：

- 6D 运动识别：在该配置下，当器件从一个方向 （已知或未知方向）移动到另一已知方向时，会生成中断。中断的有效持续时间仅为 1/ODR
- 6D 位置识别：在该配置下，当器件在已知方向处于稳定状态时，会生成中断。只要位置保持不变 （*图 12*, (a) 和 (b)），中断就有效。

参考*图 12*， 6D 运动行显示了当器件配置为在 X 和 Y 轴上实现 6D 运动识别 (INT1_CFG = 0x4Ah) 时中断的行为，而 6D 位置行则显示了当器件配置为在 X 和 Y 轴上实现 6D 位置识别 (INT1_CFG = 0xCAh) 时中断的行为。 INT1_THS 设为 0x21。

参考*图 13*，器件配置为在 X、 Y 和 Z 轴上实现 6D 位置功能。*表 17* 显示的是每个位置的 INT1_SRC 寄存器的内容。

图 12. 6D 运动与 6D 位置对比

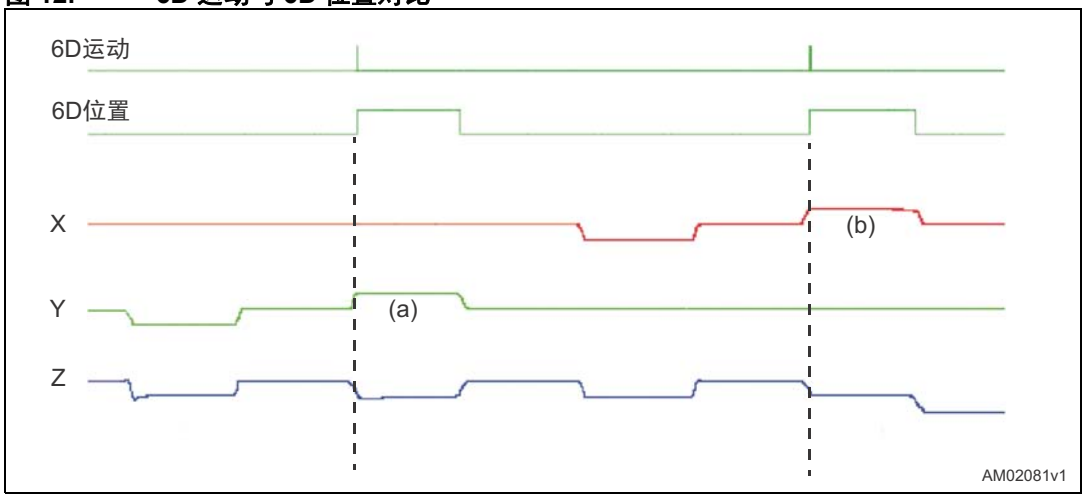


图 13. 6D 识别位置

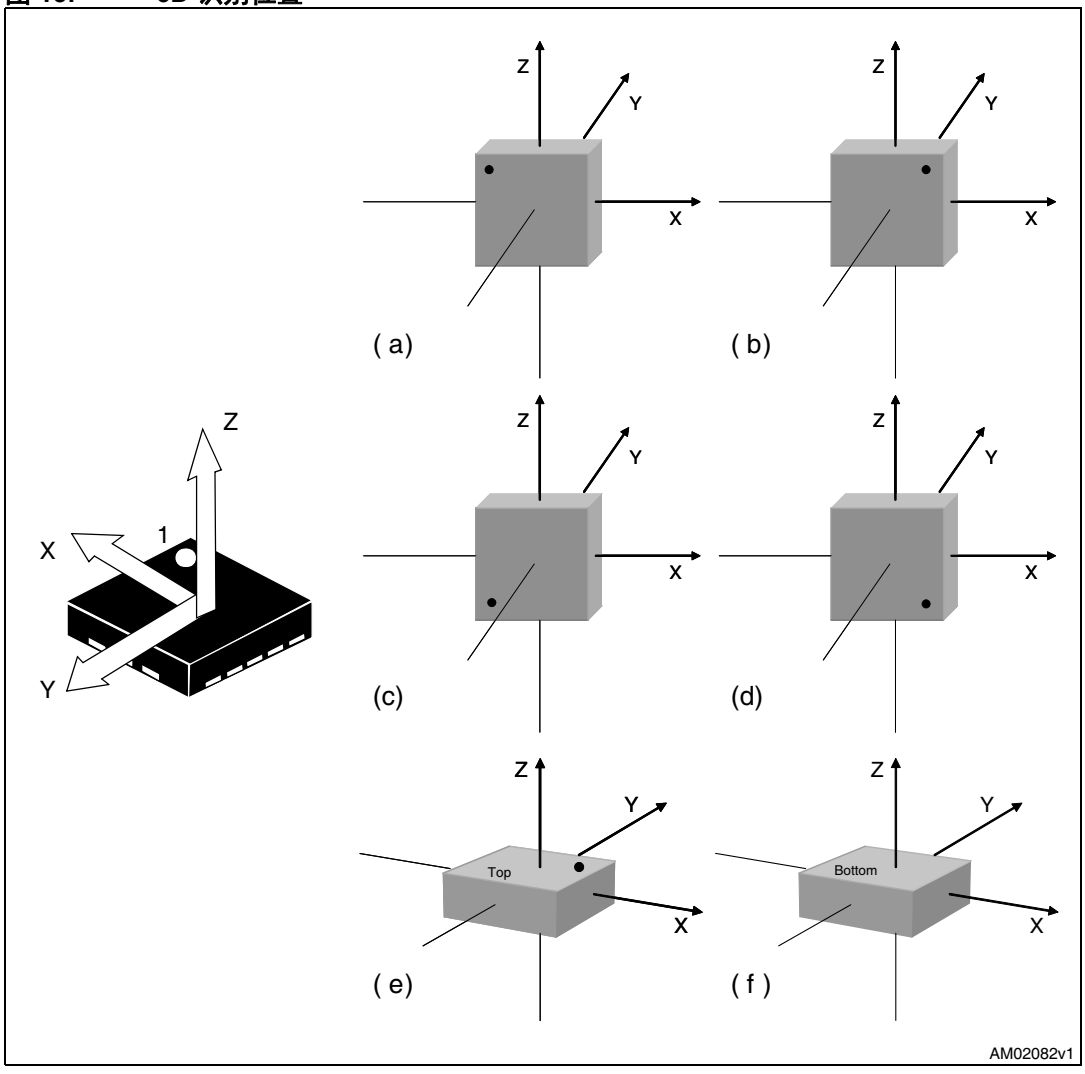


表 17. 6D 位置中的 INT1_SRC 寄存器

用例	IA	ZH	ZL	YH	YL	XH	XL
(a)	1	0	0	0	1	0	0
(b)	1	0	0	0	0	1	0
(c)	1	0	0	0	0	0	1
(d)	1	0	0	1	0	0	0
(e)	1	1	0	0	0	0	0
(f)	1	0	1	0	0	0	0

7.2 4D 方向

4D 方向功能是 6D 方向功能的子集，专为在手持设备中实施而定义。当 INT1_CFG 的 6D 位设为 1 时，可通过将 CTRL_REG5 的 D4D_INT1 位置 1 来启用此功能。在该配置下， Z 轴位置检测会禁用，从而可将位置识别缩小为*表 17* 的用例 (a)、 (b)、 (c) 和 (d)。

8 单击和双击识别

借助 LIS3DH 中的单击和双击识别功能，无需载入软件便可创建人机界面。器件可配置为沿任意方向点击时在专用引脚上输出中断信号。

如果传感器受到单个输入刺激，则会在惯性引脚 INT1 和 / 或 INT2 上生成中断请求。更先进的功能可在识别到两次输入刺激 （两个事件的间隔时间可通过程序设定）时生成中断请求，从而可实现类似鼠标按键的功能。

此功能完全可通过用户编程进行设定，用户可在程序中通过*第 8.3 节*中介绍的专用寄存器组设定刺激的预期幅度和时间。

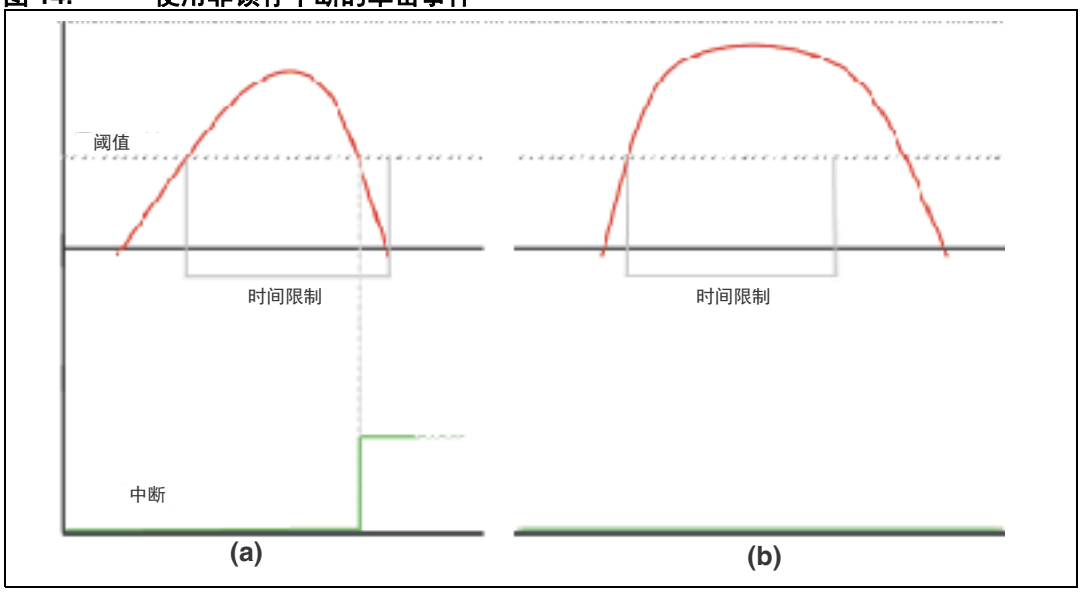
单击和双击识别分别以选定的输出数据速率工作。

8.1 单击

如果器件配置为实现单击事件识别，如果所选通道上的输入加速度超过设定的阈值、并且在由 TimeLimit 寄存器定义的时窗内恢复至阈值以下，则会生成中断，

如果 TAP_CFG 寄存器的 LIR 位未置 1，中断会保持高电平，并会在延迟窗口的持续时间内一直处于高电平状态。如果 LIR 位已置 1，中断会保持高电平，直至 TAP_SRC 寄存器被读取。

图 14. 使用非锁存中断的单击事件



在*图 14*(a) 中，已识别出点击事件，而在*图 14*(b) 中还未识别出点击，因为加速度在 TimeLimit 到期后降至阈值以下。

8.2 双击

如果器件配置为实现双击事件检测，当第一次点击后识别出第二次点击时，会生成中断。仅当事件满足延迟寄存器和窗口寄存器定义的规则时，才会识别第二次点击。

特别是在已识别了第一次点击后，第二次点击检测程序会延迟进行，延迟的时间间隔是延迟寄存器定义的 这意味着在识别出第一次点击后，仅当输入加速度在延迟窗口后、但在窗口到期 (*图 15 (a)*) 前超过阈值、或者加速度在延迟到期 (*图 16 (b)*) 后超过阈值的情况下才会开始执行第二次点击检测程序。

第二次点击检测程序启动后，会采用与识别第一次点击时一样的程序识别第二次点击：加速度必须在 TimeLimit 到期前恢复为阈值以下。

请务必正确定义延迟窗口，以避免因输入信号虚假反弹出现不需要的点击。

图 15. 单击和双击识别

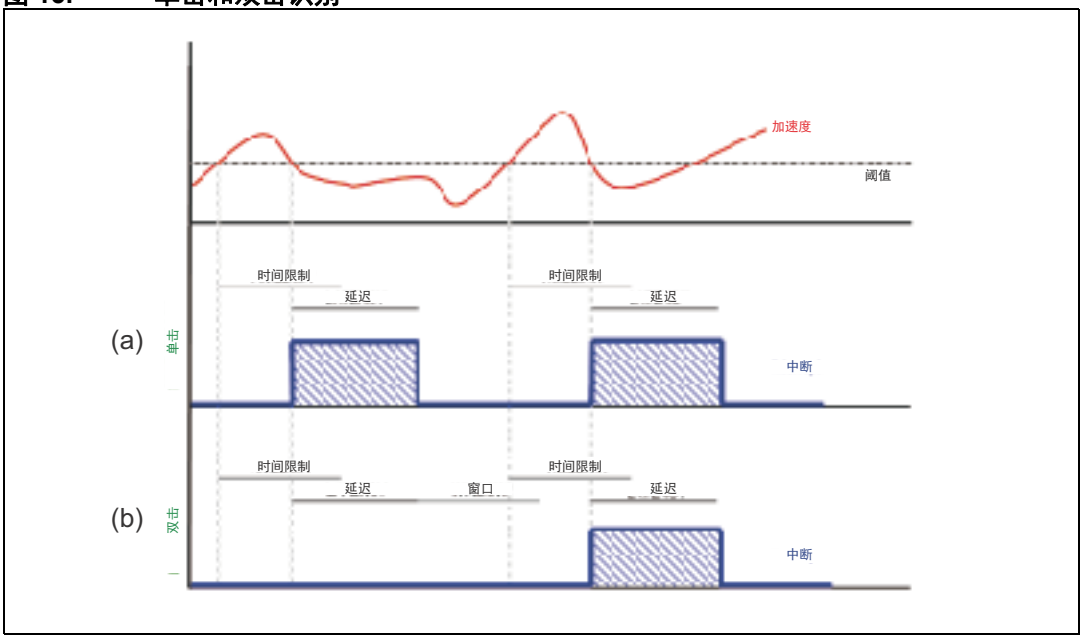
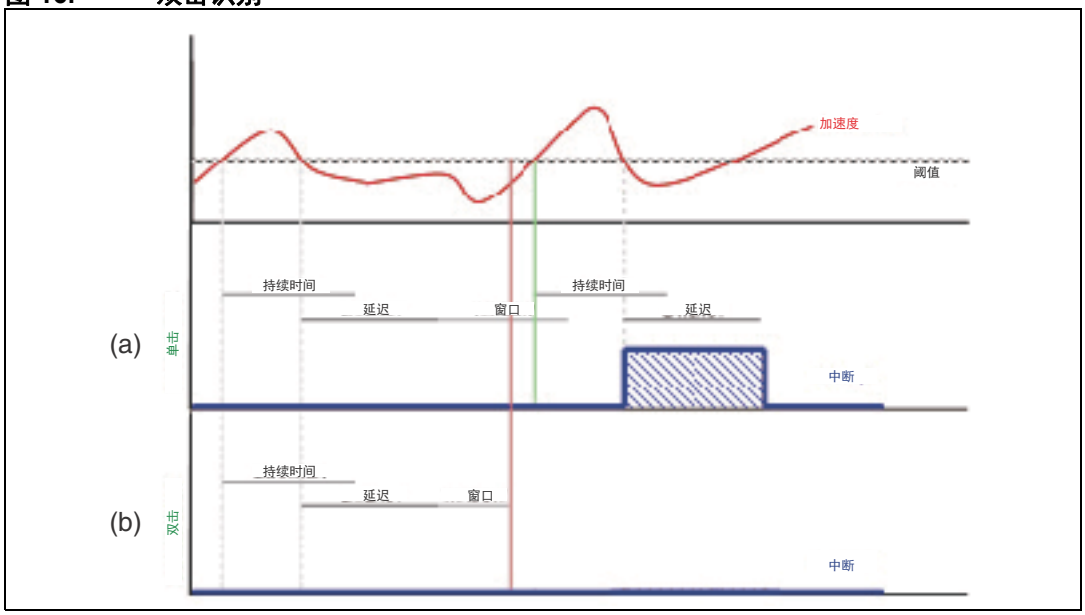


图 15 通过图形介绍了单击事件 (a) 和双击事件 (b)。器件能够通过将 TAP_CFG 寄存器的设置从单击识别改为双击识别的方式区分 (a) 和 (b)。

图 16. 双击识别



在*图 16*(a) 中，已正确识别出双击事件，而在*图 16*(b) 中还未生成中断，因为输入加速度在窗口时间间隔到期后超过阈值。

8.3 寄存器说明

8.3.1 TAP_CFG (38h)

表 18. TAP_CFG 寄存器

-	-	ZD	ZS	YD	YS	XD	XS

表 19. TAP_CFG 说明

ZD	在 Z 轴上启用中断双连击。默认值：0 （0：禁用中断请求； 1：对高于预设阈值的 测得加速度启用中断请求）
ZS	在 Z 轴上启用中断单连击。默认值：0 （0：禁用中断请求； 1：对高于预设阈值的 测得加速度启用中断请求）
YD	在 Y 轴上启用中断双连击。默认值：0 （0：禁用中断请求； 1：对高于预设阈值的 测得加速度启用中断请求）
YS	在 Y 轴上启用中断单连击。默认值：0 （0：禁用中断请求； 1：对高于预设阈值的 测得加速度启用中断请求）
XD	在 X 轴上启用中断双连击。默认值：0 （0：禁用中断请求； 1：对高于预设阈值的 测得加速度启用中断请求）
XS	在 X 轴上启用中断单连击。默认值：0 （0：禁用中断请求； 1：对高于预设阈值的 测得加速度启用中断请求）

表 20. 真值表

DZ / DY / DX	SZ / Y / X	点击输出
0	0	0
0	1	单击
1	0	双击
1	1	单击或双击

8.3.2 TAP_SRC (39h)

表 21. TAP_SRC 寄存器

-	IA	Dtap	Stap	符号	Z	Y	X

表 22. TAP_SRC 说明

-	-
IA	中断有效。默认值：0 （0：未生成中断； 1：已生成一个或多个中断）
Dtap	双连击使能。默认值：0 （0：双连击检测禁用， 1：双连接检测使能）
Stap	单连击使能。默认值：0 （0：单连击检测禁用， 1：单连接检测使能）
符号	连击符号。 0：正检测， 1：负检测
Z	Z 连击检测。默认值：0 （0：无中断， 1：发生了 Z 轴置为高电平的事件）
是	Y 连击检测。默认值：0 （0：无中断， 1：发生了 Y 轴置为高电平的事件）
X	X 连击检测。默认值：0 （0：无中断， 1：发生了 X 轴置为高电平的事件）

8.3.3 TAP_THS (3Ah)

表 23. TAP_THS 寄存器

-	THS6	THS5	THS4	THS3	THS2	THS1	THS0

表 24. TAP_SRC 说明

THS6-THS0	连击阈值。默认值：000 0000

1 LSB = 满量程 /128。

THS6 到 THS0 定义了系统启动点击检测程序所使用的阈值。阈值表示为 6 位无符号数。

8.3.4 TIME_LIMIT (3Bh)

表 25. TIME_LIMIT 寄存器

-	TLI6	TLI5	TLI4	TLI3	TLI2	TLI1	TLI0

表 26. TIME_LIMIT 寄存器

TLI7-TLI0	连击时间限制。默认值：000 0000

1 LSB = 1/ODR。

TLI7 到 TLI0 定义了启动点击检测程序 （选定通道上的加速度超过设定的阈值）与加速度变回阈值以下所经过的最大时间间隔。

8.3.5 TIME_LATENCY (3Ch)

表 27. TIME_LATENCY 寄存器

TLA7	TLA6	TLA5	TLA4	TLA3	TLA2	TLA1	TLA0

表 28. TIME_LATENCY 说明

TLA7-TLA0	连击时间延迟。默认值：000 0000

1 LSB = 1/ODR。

如果器件配置为实现双击检测， TLA7 到 TLA0 定义了在第一次点击检测后开始的时间间隔，在这段时间内会禁用点击检测程序。

8.3.6 时窗 (3Dh)

表 29. TIME_WINDOW 说明

TW7	TW6	TW5	TW4	TW3	TW2	TW1	TW0

表 30. TIME_LATENCY 说明

TW7-TW0	连击时窗

1 LSB = 1/ODR。

如果器件配置为实现双击检测， TW7 到 TW0 定义了延迟间隔结束后可经过的最大时间间隔，在这段时间内可开始点击检测程序。

8.3.7 CTRL_REG3 [中断 CTRL 寄存器] (22h)

表 31. CTRL_REG3 寄存器

I1_TAP	I1_INT1	-	I1_DRDY1	I1_DRDY2	I1_WTM	I1_OVERRUN	-

表 32. CTRL_REG3 说明

I1_TAP	INT1 上的点击中断。默认值 0。 （0：禁用； 1：启用）
I1_INT1	INT1 引脚上的中断发生器 1。默认值 0。 （0：禁用； 1：启用）
I1_DRDY1	INT1 上的 DRDY1 中断。默认值 0。 （0：禁用； 1：启用）
I1_DRDY2	INT1 上的 DRDY2 中断。默认值 0。 （0：禁用； 1：启用）
I1_WTM	INT1 上的 FIFO 水印中断。默认值 0。 （0：禁用； 1：启用）
I1_OVERRUN	INT1 上的 FIFO 上溢中断。默认值 0。 （0：禁用； 1：启用）

8.4 示例

下图显示的是不同条件下的点击中断生成情况。截图是在运行演示套件 GUI 接口的电脑上获取的，ODR 设为 400 Hz，满量程设为 4 g。LIS3DH 寄存器的内容已通过软件界面的专用面板进行修改，用户可在该面板中对嵌入了点击的功能的所有不同设置和特性进行评估。下例中，仅启用了 X 轴生成点击中断。

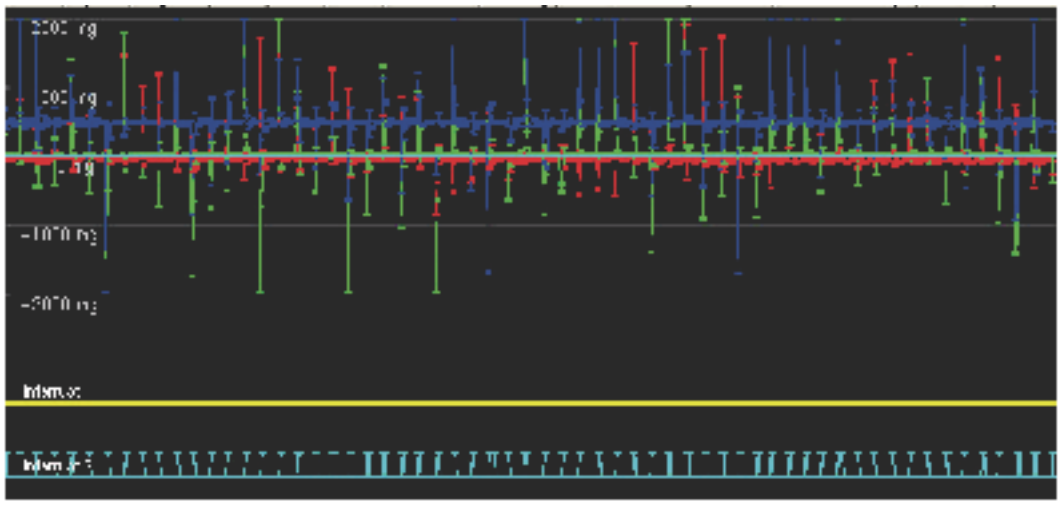
8.4.1 调整 TAP_TimeLimit

图 17 显示的是 TAP_TimeLimit = 01h (2.5 ms) 时执行的采集。使用该设置时，单击识别窗口较短，通常加速度不会及时恢复为阈值以下。

图 18 中显示的是 TAP_TimeLimit = 33h (127 ms) 时完成的采集。使用该设置时，单击识别窗口较长，事件更容易识别。

图 17. 短 TimeLimit

图 18. 长 TimeLimit



8.4.2 调整 TAP_Latency

图 19 显示的是 TAP_Latency =15h (52 ms) 时进行的采集。使用该设置时，器件几乎会将每个加速度尖峰识别为点击操作。

图 20 中显示的是 TAP_Latency = FFh (637 ms) 时执行的采集。使用该设置时，器件会将每两个尖峰中的一个识别为点击操作。

图 19. 短延迟

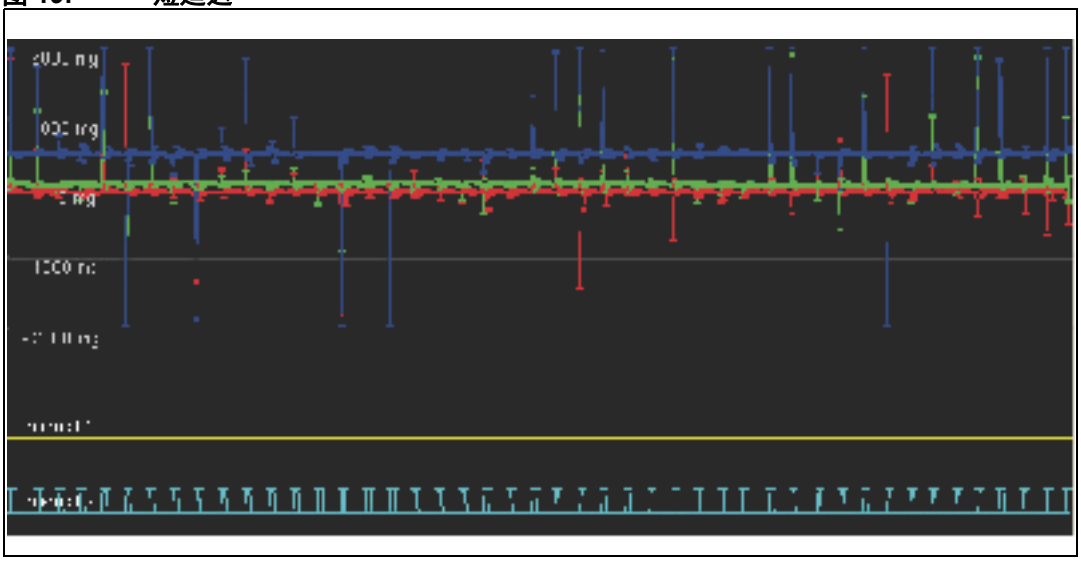
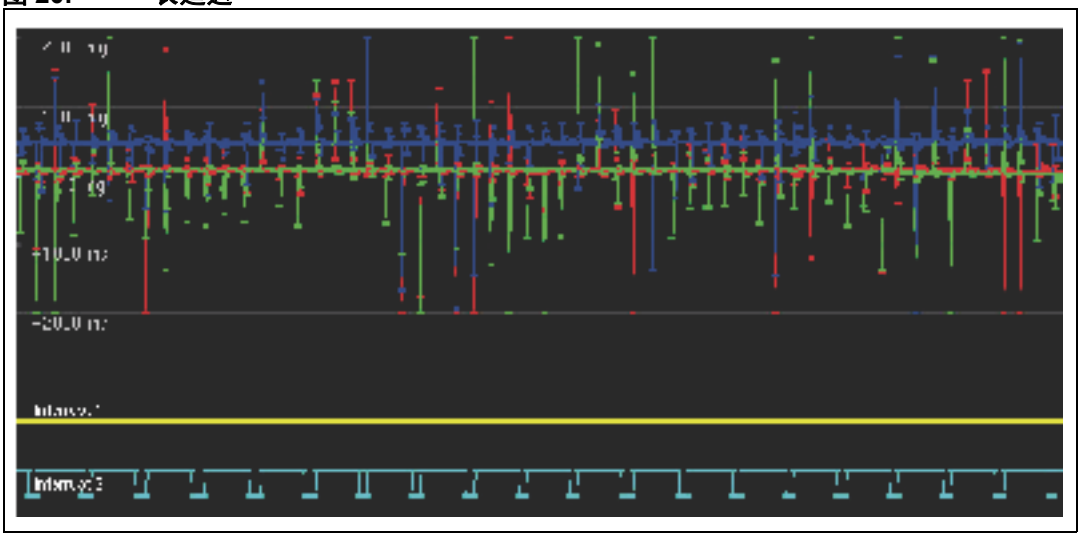


图 20. 长延迟



8.4.3 调整 TAP_Window

进行双击识别时，TAP_Latency + TAP_Window 定义了将两次连续点击识别为双击事件的最大间隔时间。固定延迟可避免信号虚假反弹，可以像在电脑上调整鼠标属性的“双击速度”设置一样调整 TAP_Window。

图 21 显示的是 TAP_Window = 42h (1065 ms) 时进行的采集。使用该设置时，加速度的两个连续峰值间隔越远，第二个峰值出现在窗口之外。

图 22 中显示的是 TAP_Window = FFh (637 ms) 时执行的采集。使用该设置时，器件会在第二个加速度峰值后正确生成双击中断。

图 21. 短窗口

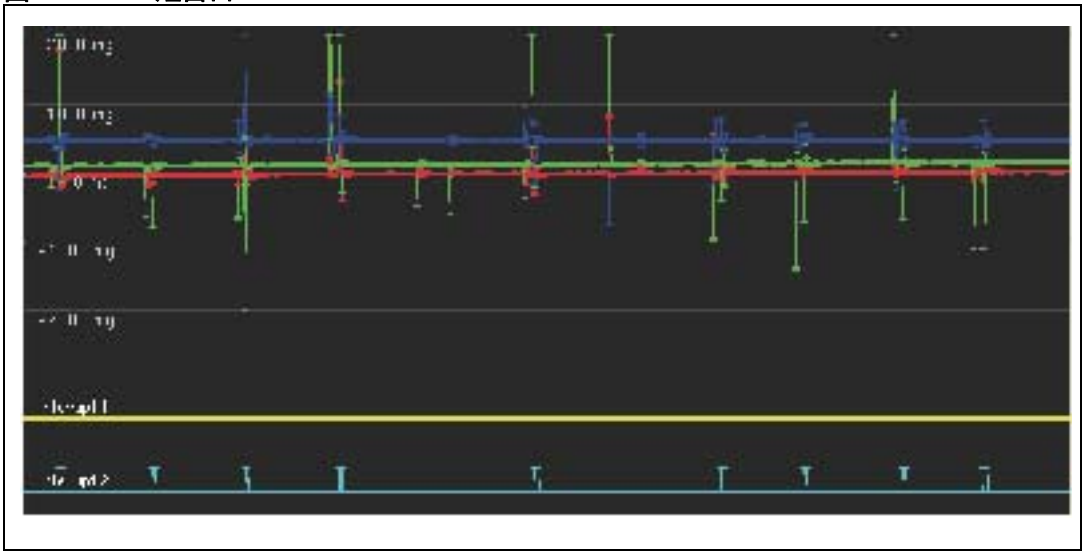
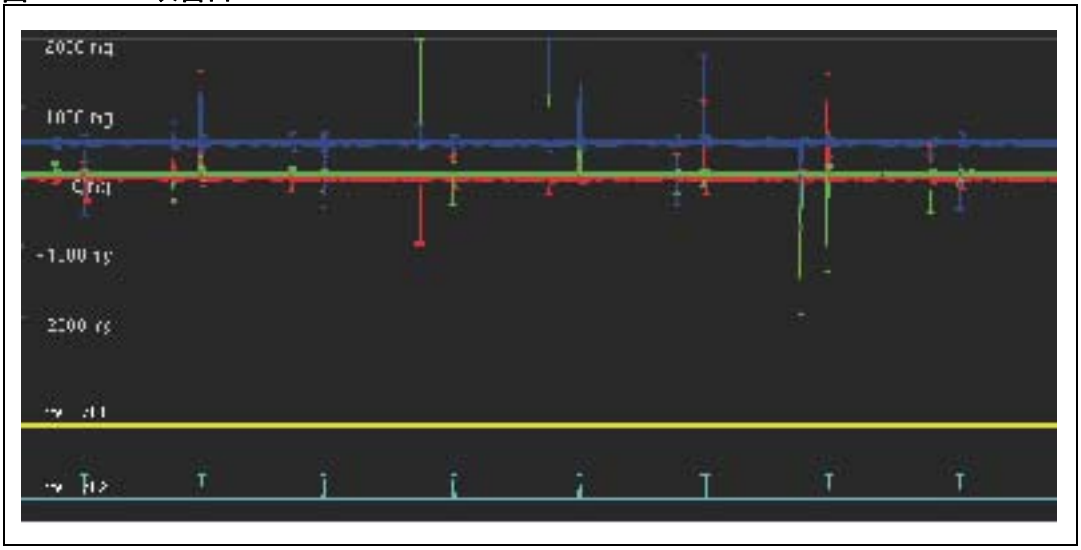


图 22. 长窗口



9 先进先出 (FIFO) 缓冲区

为了减少主机处理器交互并便于对事件识别数据进行后处理，LIS3DH 为三条输出通道 X、Y 和 Z 分别嵌入了先进先出 (FIFO) 缓冲区。

使用 FIFO 可使系统实现一致的节能效率，仅当需要时才会唤醒 FIFO，并会从 FIFO 批量输出重要数据。

FIFO 缓冲区可在四种不同模式下工作，各个模式可确保在应用开发过程中实现高度灵活性：Bypass 模式、 FIFO 模式、 Stream 模式和 Stream-to-FIFO 模式。

可使能可编程水印等级和 FIFO 上溢事件在 INT1 引脚上生成专用中断。

9.1 FIFO 描述

FIFO 缓冲区最多能为每条通道存储 32 个 10 位加速度采样；数据采用左对齐 16 位 2 的补码形式存储。

数据样本集合由 6 个字节（Xl、Xh、Yl、Yh、Zl 和 Zh）和组成，它们会以选定的输出数据速率 (ODR) 释放到 FIFO 中。

新样本集合会放在第一个空闲的 FIFO 位置中，缓冲区被占满后，新样本集合会覆盖最早的值。

表 33. FIFO 缓冲区完整表示 （存储了第 32 个样本集合）

输出寄存器	**0x28h**	**0x29h**	**0x2Ah**	**0x2Bh**	**0x2Ch**	**0x2Dh**
	Xl(0)	Xh(0)	Yl(0)	Yh(0)	Zl(0)	Zh(0)
FIFO 索引	**FIFO 样本集合**					
FIFO(0)	Xl(0)	Xh(0)	Yl(0)	Yh(0)	Zl(0)	Zh(0)
FIFO(1)	Xl(1)	Xh(1)	Yl(1)	Yh(1)	Zl(1)	Zh(1)
FIFO(2)	Xl(2)	Xh(2)	Yl(2)	Yh(2)	Zl(2)	Zh(2)
FIFO(3)	Xl(3)	Xh(3)	Yl(3)	Yh(3)	Zl(3)	Zh(3)
...	...	...	...	...	...	...
...	...	...	...	...	...	...
FIFO(30)	Xl(30)	Xh(30)	Yl(30)	Yh(30)	Zl(30)	Zh(30)
FIFO(31)	Xl(31)	Xh(31)	Yl(31)	Yh(31)	Zl(31)	Zh(31)

表 34. FIFO 上溢表示 （存储了第 33 个样本集合、丢弃了第 1 个样本）

输出寄存器	0x28h	0x29h	0x2Ah	0x2Bh	0x2Ch	0x2Dh
	Xl(1)	Xh(1)	Yl(1)	Yh(1)	Zl(1)	Zh(1)
FIFO 索引	样本集合					
FIFO(0)	Xl(1)	Xh(1)	Yl(1)	Yh(1)	Zl(1)	Zh(1)
FIFO(1)	Xl(2)	Xh(2)	Yl(2)	Yh(2)	Zl(2)	Zh(2)
FIFO(2)	Xl(3)	Xh(3)	Yl(3)	Yh(3)	Zl(3)	Zh(3)
FIFO(3)	Xl(4)	Xh(4)	Yl(4)	Yh(4)	Zl(4)	Zh(4)
...	...	...	...	...	...	...
...	...	...	...	...	...	...
FIFO(30)	Xl(31)	Xh(31)	Yl(31)	Yh(31)	Zl(31)	Zh(31)
FIFO(31)	Xl(32)	Xh(32)	Yl(32)	Yh(32)	Zl(32)	Zh(32)

表 33 表示的是存储了 32 个样本时 FIFO 已满的状态，而*表 34* 表示的是第 33 个样本插入到 FIFO 中、第 1 个样本被覆盖时的下一步。新的最早样本集合在输出寄存器中可用。

如果 FIFO 已使能，并且所处模式不是 Bypass 模式， LIS3DH 输出寄存器 （28h 到 2Dh）始终会包含最早的 FIFO 样本集合。

9.2 FIFO 寄存器

FIFO 缓冲区由三个不同的加速度计寄存器进行管理，其中两个寄存器可使能并配置 FIFO 特性，第三个寄存器会提供关于缓冲区状态的信息。

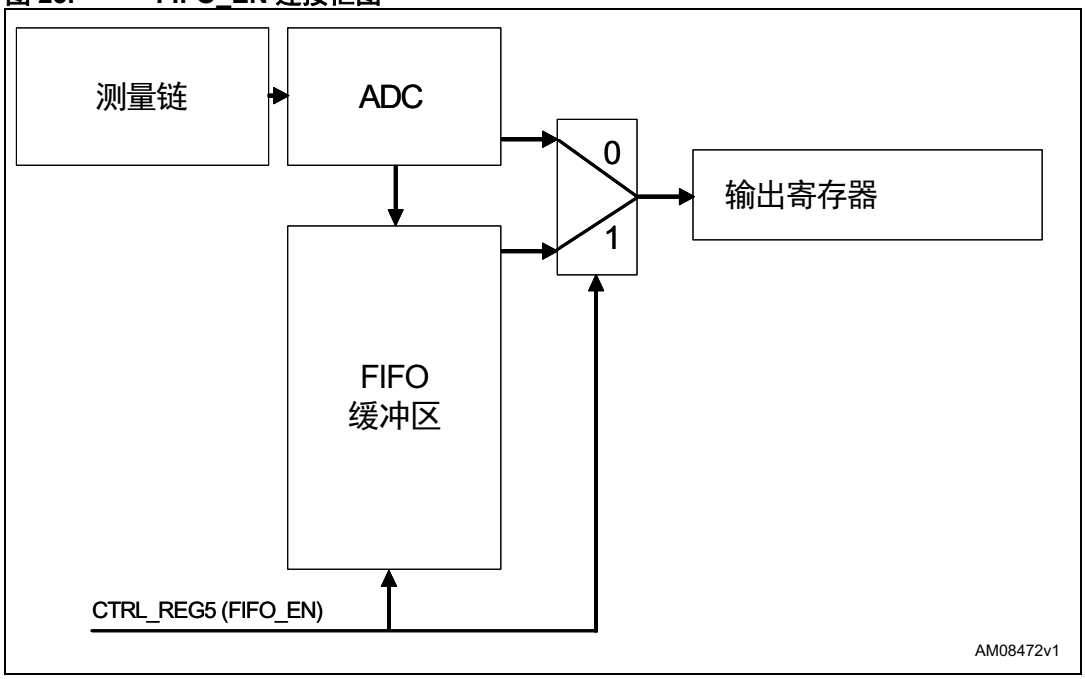
9.2.1 控制寄存器 5 (0x24)

CTRL_REG5 中的 FIFO_EN 位必须设为 1 才能使能内部的先进先出缓冲区；如果该位置 1，加速度计输出寄存器 （28h 到 2Dh）不会包含当前加速度计值，但始终会包含 FIFO 中存储的最早值。

表 35. CTRL_REG5 中的 FIFO 使能位

b7	b6	b5	b4	b3	b2	b1	b0
X	FIFO_EN	X	X	X	X	X	X

图 23. FIFO_EN 连接框图



9.2.2 FIFO 控制寄存器 (0x2E)

该寄存器专用于 FIFO 模式选择和水印配置。

表 36. FIFO_CTRL_REG

b7	b6	b5	b4	b3	b2	b1	b0
FM1	FM0	0	FTH4	FTH3	FTH2	FTH1	FTH0

FM[1:0] 位专用于定义 FIFO 缓冲区特性选择：

1. FM[1:0] = (0,0)：Bypass 模式
2. FM[1:0] = (0,1)：FIFO 模式
3. FM[1:0] = (1,0)：Stream 模式
4. FM[1:0] = (1,1)：Stream-to-FIFO 模式

用于激活 Stream-to-FIFO 模式的触发与已选 NT1_SRC 寄存器 IA 位的值相关，不取决于中断引脚值和极性。如果已选中断未驱动到中断引脚，也会生成触发。

FTH[4:0] 位用于定义水印等级；如果 FIFO 内容超过该值，FIFO 源寄存器中的 WTM 位会置“1”。

9.2.3 FIFO 源寄存器 (0x2F)

该寄存器每个 ODR 会更新一次，会提供关于 FIFO 缓冲区状态的信息。

表 37. FIFO_SRC_REG

b7	b6	b5	b4	b3	b2	b1	b0
WTM	OVRN	空	FSS4	FSS3	FSS2	FSS1	FSS0

- 如果 FIFO 内容超过水印等级， WTM 位会置 1。
- 如果FIFO缓冲区已满，OVRN位会置1，这意味着FIFO缓冲区包含32个未读样本。下一ODR 时，新样本集合会替换最早的 FIFO 值。第一个样本集合已被读取时， OVRN 位会复位。
- 当所有 FIFO 样本已被读取并且 FIFO 为空时， EMPTY 标志会置 1。
- FSS[4:0] 字段始终包含 FIFO 缓冲区中存储的当前未读样本数。FIFO 使能后，该值会以 ODR 频率增加，直至缓冲区已满，随后，每次从 FIFO 重新获取一个样本集合，该值都会减小。

寄存器内容会与 FIFO 写操作和读操作同步更新。

表 38. FIFO_SRC_REG 特性 （假定 FTH[4:0] = 15）

WTM	OVRN	空	FSS[4:1]	未读 FIFO 样本	时序
0	0	1	00000	0	t0
0	0	0	00001	1	t0 + 1/ODR
0	0	0	00010	2	t0 + 2/ODR
...	...	...	...	...	...
0	0	0	01111	15	t0 + 15/ODR
1	0	0	10000	16	t0 + 16/ODR
...	...	...	...	...	...
1	0	0	11110	30	t0 + 30/ODR
1	0	0	11111	31	t0 + 31/ODR
1	1	0	11111	32	t0 + 32/ODR

可通过配置 CTRL_REG3 使能水印标志和 FIFO 上溢事件，以便在 INT1 引脚上生成专用中断。

表 39. CTRL_REG3 (0x22)

b7	b6	b5	b4	b3	b2	b1	b0
X	X	X	X	X	I1_WTM	I1_OVRN	X

- I1_WTM 位会驱动 INT1 引脚上的水印标志 (WTM)。
- I1_OVRN 位会驱动 INT1 引脚上的上溢事件 (OVRN)。

如果两个位均置 “1”， INT1 引脚状态为两个信号的逻辑或组合。

9.3 FIFO 模式

LIS3DH FIFO 缓冲区可配置为在四种不同的模式下工作，可通过 FIFO_CTRL_REG 中的 FM[1:0] 字段选择工作模式。提供的配置可确保实现高度灵活性，并可增加应用开发中可使用的功能数。

Bypass、 FIFO、 Stream 和 Stream-to-FIFO 模式在下面几段中进行了介绍。

9.3.1 Bypass 模式

启用 Bypass 模式后， FIFO 不可运行：缓冲区内容会被清空、输出寄存器 （0x28 到 0x2D）会冻结为最后载入的值，在选择其他模式之前， FIFO 缓冲区会保持空白状态。

请按照以下步骤配置 Bypass 模式：

1. 将控制寄存器 5 (0x24) 中的 FIFO_En 位置 “1” 可使能 FIFO。执行完此操作后， FIFO 缓冲区会使能，但不会采集数据，数据寄存器冻结为上次加载的样本集合。
2. 将 FIFO 控制寄存器 (0x2E) 中的 FN[1:0] 字段设为 “00” 可激活 Bypass 模式。如果启用该模式， FIFO 源寄存器 (0x2F) 会前置设为等于 0x20。

必须使用 Bypass 模式才能在另一模式运行时停止并复位 FIFO 缓冲区。请注意，将 FIFO 缓冲区置于 Bypass 模式会清除整个缓冲区的内容。

9.3.2 FIFO 模式

在 FIFO 模式下，缓冲区会继续填入数据，直至填满为止 （存储 32 个样本集合），随后，缓冲区会停止采集数据， FIFO 内容保持不变，直至选择了另一模式。

请按照以下步骤配置 FIFO 模式：

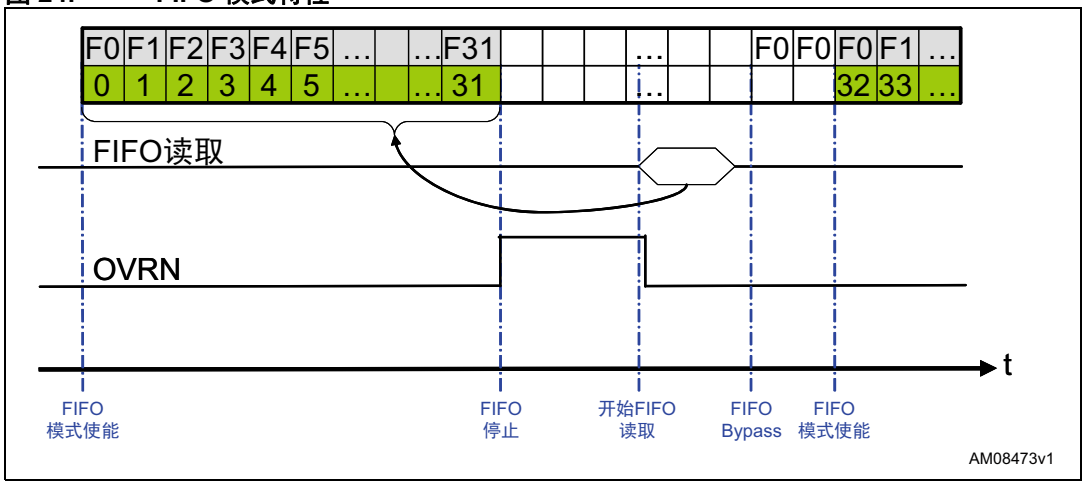
1. 将控制寄存器 5 (0x24) 中的 FIFO_En 位置 “1” 可使能 FIFO。执行完此操作后， FIFO 缓冲区会使能，但不会采集数据，数据寄存器冻结为上次加载的样本集合。
2. 将 FIFO 控制寄存器 (0x2E) 中的 FN[1:0] 字段设为 “01” 可激活 Bypass 模式。

选择该模式后， FIFO 会开始进行数据采集，源寄存器 (0x2F) 也会根据存储的样本数发生变化。此过程结束时，如果选择了控制寄存器 5 中的 I1_OVRN 位，源寄存器会设为 0xDF，OVRN 标志会生成中断。如果 OVRN 置为 “1”，可重新获取数据，获取数据时会从输出寄存器读取 32 个样本集合，如果应用要求的样本数较少，还可以根据 WTM 标志 （而不是 OVRN）重新获取数据。由于在 FIFO 模式下数据采集已停止，并且不存在覆盖已获取数据的风险，因此通信速度并不重要。重新启动 FIFO 模式之前，请务必在读取程序之后通过 Bypass 模式进行转换。

FIFO 模式应用提示如下：

1. 将 FIFO_En 置 1：使能 FIFO
2. 将 FM[1:0] 设为 (0,1)：使能 FIFO 模式
3. 等待 OVRN 或 WTM 中断
4. 从加速度计输出寄存器读取数据
5. 将 FM[1:0] 设为 (0,0)：使能 Bypass 模式
6. 重复第 2 点及后续步骤

图 24. FIFO 模式特性



如果使能了 FIFO 模式，缓冲区会开始采集数据，并会以所选输出数据速率填入全部 32 个位置 （从 F0 到 F31）。缓冲区已满后， OVRN 位会变为高电平，数据采集会永久停止；用户可随时读取 FIFO 内容，因为在选择 Bypass 模式之前， FIFO 缓冲区的内容保持不变。读取程序包括 32 个 6 字节样本集合 （共 192 字节），会从 FIFO 中存储的最早的样本 (F0) 开始获取数据。第一个样本集合已被读取时， OVRN 位会复位。 Bypass 模式设置会复位 FIFO 并允许用户再次使能 FIFO 模式。

9.3.3 Stream 模式

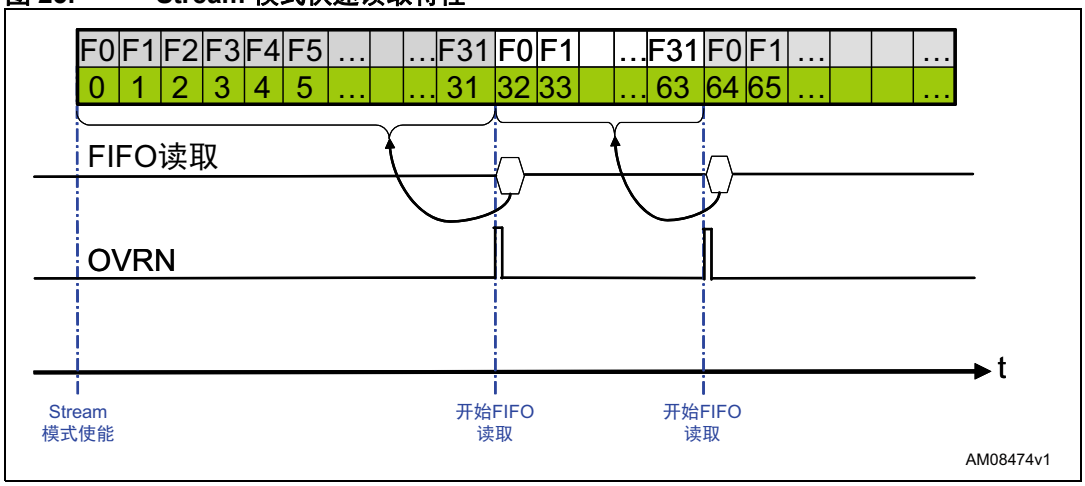
在 Stream 模式下， FIFO 会继续填入数据，如果缓冲区已满， FIFO 索引会从头开始使用，较早的数据会被当前数据替代。最早的值会继续被覆盖，直至读取操作释放了可用 FIFO 位置。为了使 FIFO 位置的释放速度快于获得新数据的速度，主机处理器的读取速度至关重要。 FM[1:0] Bypass 配置用于停止该模式。

请按照以下步骤配置 FIFO 模式：

1. 将控制寄存器 5 (0x24) 中的 FIFO_En 位置 “1” 可使能 FIFO。执行完此操作后， FIFO 缓冲区会使能，但不会采集数据，数据寄存器冻结为上次加载的样本集合。
2. 将 FIFO 控制寄存器 (0x2E) 中的 FN[1:0] 字段设为 “10” 可激活 Stream 模式。

如上所述，对于 FIFO 模式，如果 OVRN 置为 “1”，可重新获取数据，获取数据时会从输出寄存器读取 32 个样本集合，如果应用要求的样本数较少，还可以根据 WTM 标志重新获取数据。

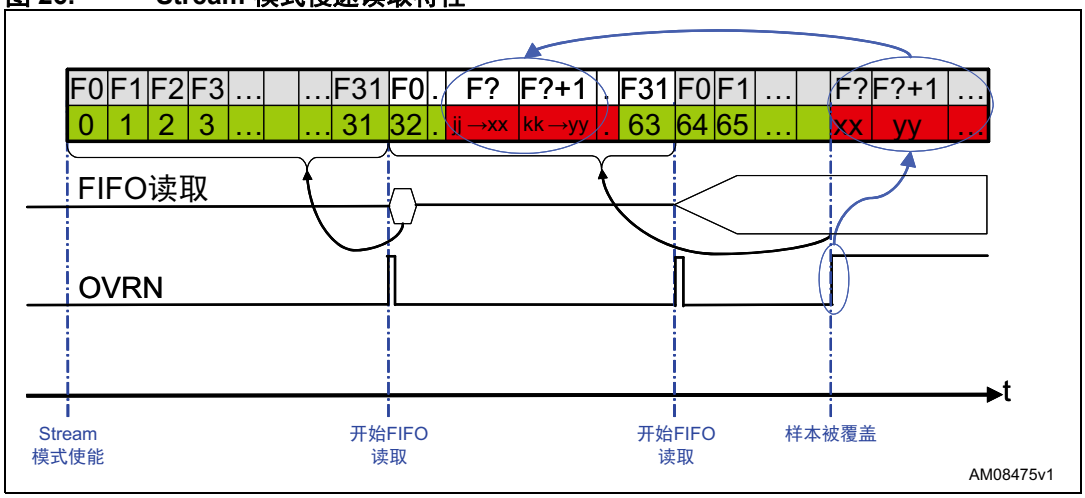
图 25. Stream 模式快速读取特性



在 Stream 模式下， FIFO 缓冲区会以选定的输出数据速率持续填入数据 （从 F0 到 F31）。缓冲区填满后， OVRN 标志会变为高电平，建议的解决方法是以快于 1*ODR 的速度读取所有 FIFO 样本 （192 个字节），以便释放 FIFO 位置供新的加速度样本使用。这样可避免数据丢失，并可减少主机处理器交互，从而可提高系统效率。如果读取过程的速度不够快，可观察到三种不同的情况：

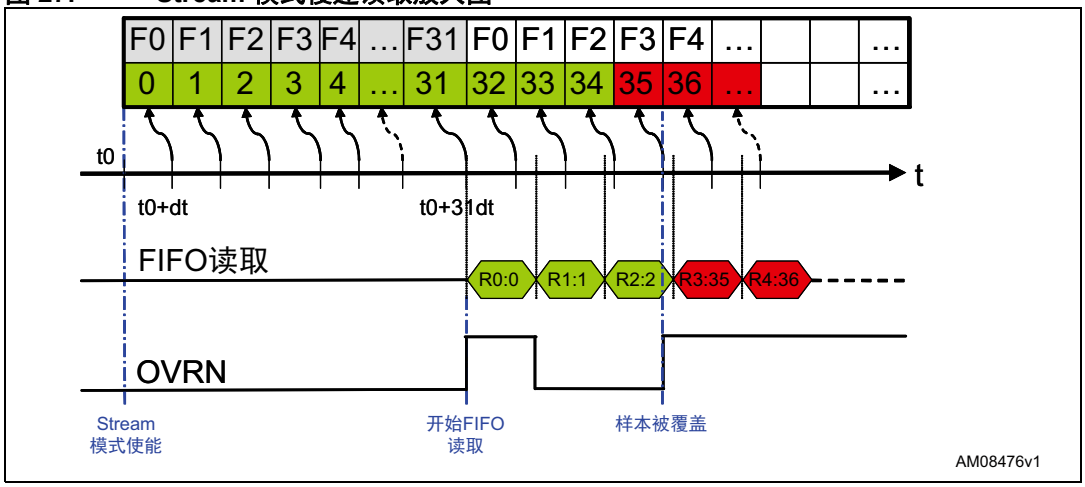
1. FIFO 样本集合 （6 字节）的读取速度快于 1*ODR：由于新数据生成前已释放了 FIFO 位置，因此可正确地重新获取数据。
2. FIFO 样本集合 （6 字节）的读取速度与 1*ODR 同步：由于新数据生成前已释放了 FIFO 位置，因此可正确地重新获取数据，但没有用到 FIFO 的优势。这种情况相当于在发生数据就绪中断时读取数据，与标准的加速度计读取相比，不会减少主机处理器的交互。
3. FIFO 样本集合 （6 字节）的读取速度比 1*ODR 慢：在这种情况下，一些数据会丢失，因为数据恢复的速度不够快，无法为新的加速度数据释放 FIFO 位置。*图 26*. 正确恢复的样本数与当前 ODR 与 FIFO 样本集合读取速率之差相关。

图 26. Stream 模式慢速读取特性



在*图 26* 中，由于读取速度慢，不会重新获取来自 “jj” 的数据，因为这些数据会被系统生成的新加速度计样本所代替。

图 27. Stream 模式慢速读取放大图



使能 Stream 模式后， FIFO 位置会在每个 ODR 时间帧结束时填入数据。 OVRN 标志置 “1” 后，必须立即开始读取过程，会在读取操作开始时从 FIFO 重新获取数据。读取命令发送到器件后，输出寄存器内容会移动到 SPI/I^{2}C 寄存器，当前最早的 FIFO 值会移入输出寄存器，以执行下一次读取操作。如果读取速度比 1*ODR 慢，新样本插入到寻址位置后，可从 FIFO 重新获取一些数据。在*图 27* 中， F3 索引刷新后会开始执行第四条读取命令，此命令会中断数据读取。 OVRN 标志告知用户该事件已发生。本例中读取了三个正确样本，正确恢复的样本数取决于 ODR 和 FIFO 样本集合读取时间帧之差。

9.3.4 Stream-to-FIFO 模式

此模式是上述 Stream 模式与 FIFO 模式的结合。在 Stream-to-FIFO 模式下， FIFO 缓冲区会在 Stream 模式下开始工作，并会在发生选定的中断后切换为 FIFO 模式。

请按照以下步骤配置 Stream-to-FIFO 模式：

1. 使用寄存器 INT1_CFG (0x30) 配置所需中断发生器。
2. 根据配置的中断发生器配置 FIFO 控制寄存器 (0x2E) 中的 TRen 位：将 TRen 设为 “0” 可选择中断 1，将 TRen 设为 “1” 可选择中断 2。
3. 将控制寄存器 5 (0x24) 中的 FIFO_En 位置 “1” 可使能 FIFO。执行完此操作后， FIFO 缓冲区会使能，但不会采集数据，数据寄存器冻结为上次加载的样本集合。
4. 将 FIFO 控制寄存器 (0x2E) 中的 FN[1:0] 字段设为 “11” 可激活 Stream-to-FIFO 模式。

中断触发与 INT1_SRC 寄存器中的 IA 相关联，即使中断信号未驱动到中断焊盘，也会生成中断触发。如果 IA 和 OVRN 位均置 1，则会执行模式转换。 Stream-to-FIFO 模式易受触发电平影响，而不是是触发边沿的影响，这意味着如果 stream-to-FIFO 处于 FIFO 模式，并且中断条件消失， FIFO 缓冲区会因 IA 位变为零而恢复为 Stream 模式。建议锁存用作 FIFO 寄存器的中断信号，以免中断信号丢失。如果选定的中断被锁存，则需要读取寄存器 INT1_SRC 来清空 IA 位；读取后， IA 位会取 2*ODR 的值变低。

在 Stream 模式下， FIFO 缓冲区会继续填入数据，当缓冲区已满时， OVRN 位会置 1，后续样本会覆盖原样本。发生触发时，可观察到两种不同情况：

1. 如果 FIFO 缓冲区已满 (OVRN = “1”)，则会在触发后收到第一个样本时停止采集数据。FIFO 内容由触发事件前的 30 个样本、生成了中断的样本以及触发后的一个样本构成。
2. 如果 FIFO 未满 （初始瞬态），则会继续填入数据，直至填满为止 (OVRN = “1”)，之后，如果触发条件仍存在， FIFO 会停止采集数据。

图 28. Stream-to-FIFO 模式：中断未锁存

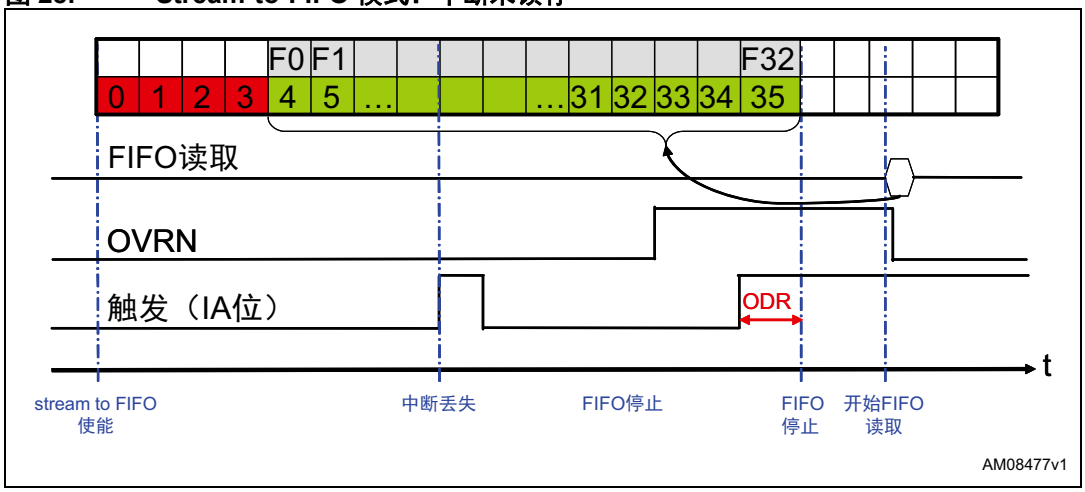
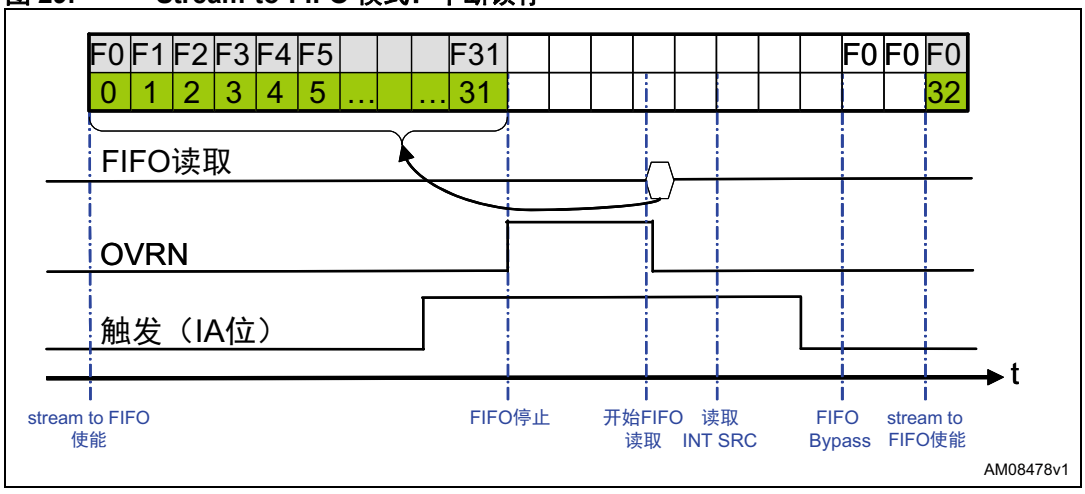


图 29. Stream-to-FIFO 模式：中断锁存



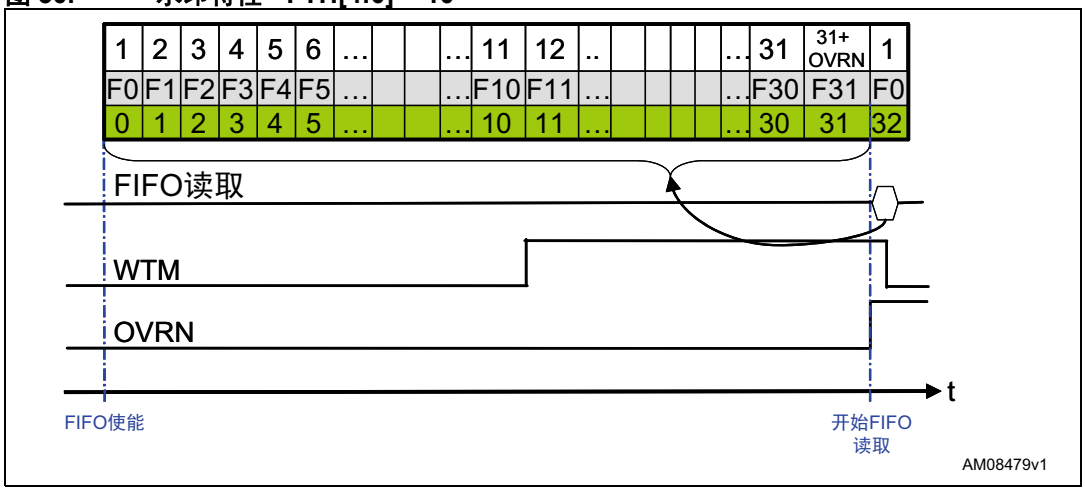
可使用 Stream-to-FIFO 来分析生成中断的样本历史；标准操作是在 FIFO 模式已触发、FIFO 缓冲区已满并停止时读取 FIFO 内容。

9.4 水印

水印是可用于生成特定中断的可配置标志，可用于确定 FIFO 缓冲区何时包含的样本数至少为定义为水印级别的数目。用户可使用 FIFO 控制寄存器中的 FTH[4:0] 字段选择所需级别（范围为 0 到 31），而 FIFO 源寄存器 FSS[4:0] 始终包含存储在 FIFO 中的样本数。

如果 FSS[4:0] 大于 FTH[4:0]， FIFO 源寄存器中的 WTM 位会置 1，相反，如果 FSS[4:0] 字段小于 FTH[4:0]， WTM 会置 0。 FSS[4:0] 会以 ODR 频率增加一步，每次由用户执行样本集合读取操作时， FSS[4:0] 会减小一步。

图 30. 水印特性 - FTH[4:0] = 10



在*图 30* 中，第一行指示 FSS[4:0] 值，第二行指示相对 FIFO 位置，最后一行显示增加的 FIFO 数据。假定 FTH[4:0] = 10，当第 11 个 FIFO 位置 (F10) 被填入数据时， WTM 标志会从 “0” 变为 “1”。*图 31* 显示了当 FIFO 内容小于 FTH[4:0] 时， WTM 标志变为低电平，这意味着第 9 个未读样本集合仍在 FIFO 中。

可将 CTRL_REG3 中的 I1_WTM 位置为高电平，以此使能水印标志 (WTM)，在 INT1 引脚上生成专用中断。

9.5 从 FIFO 重新获取数据

如果 FIFO 已使能，并且所处模式不是 Bypass 模式，读取输出寄存器 （28h 到 2Dh）会返回早的 FIFO 样本集合。

读取输出寄存器时，其内容会移至 SPI/I^2C 输出缓冲区。FIFO 位置最好上移一位，以便为接收的新样本释放空间，并使输出寄存器能够载入 FIFO 缓冲区中存储的当前最早的值。

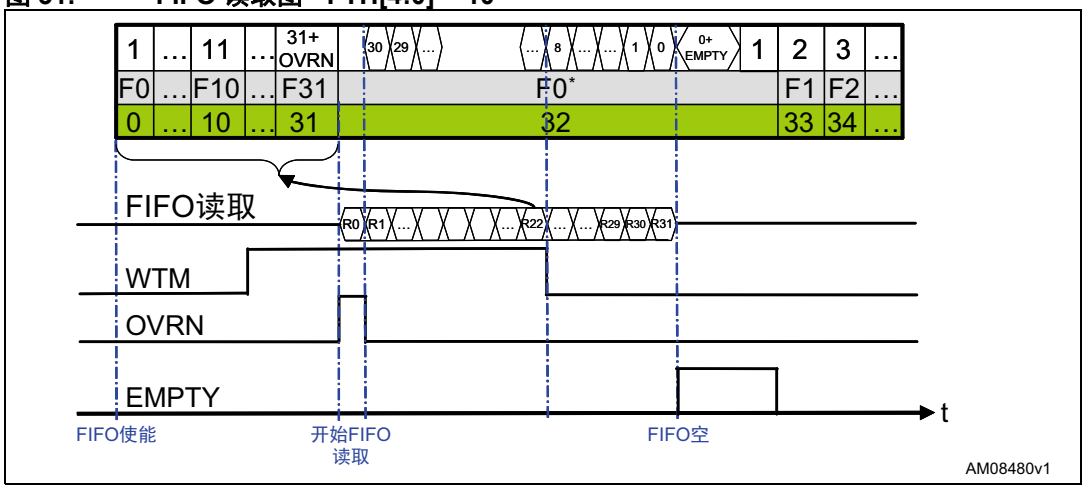
整个 FIFO 内容是通过对加速度计输出寄存器执行 32 次读取操作重新获取的， FIFO 缓冲区有新样本集合之前，所有其他的读取操作都会返回相同的最后值。

为了提高应用的灵活性，可使用每种读取字节组合从 FIFO 重新获取数据 （例如：196 次单字节读取， 32 次 6 字节读取， 1 次 196 字节的多字节读取等）。

建议以快于 1*ODR 的速度通过 196 字节多字节读取的方式来读取所有 FIFO 位置 （6 个输出寄存器乘以 32 个存储位置）。为了最大限度地减少主器件与从器件之间的通信，器件会自动更新读取地址；到达寄存器 0x2D 时，会回滚到 0x28。

为了避免数据丢失，必须按照可用的串行通信速率选择正确的 ODR。如果使用的是标准 I^2C 模式 （最大速率 100 kHz），单次样本集合读取会用时 830 µs，而整体 FIFO 下载约用时 17.57ms。I^2C 速度低于 SPI，大概需要 29 个时钟脉冲才能开始通信 （开始、从站地址、器件地址 + 写入地址、重新开始、器件地址 + 读取），另外还需要 9 个时钟脉冲才能读取每个字节。如果遵循该建议，完整 FIFO 读取的执行速度会快于 1*ODR，这意味着如果使用标准 I^2C，可选 ODR 必须低于 57 Hz。如果使用快速 I^2C 模式 （最大速率 400 kHz），可选 ODR 必须低于 228 Hz。

图 31. FIFO 读取图 - FTH[4:0] = 10



在*图 31* 中，“Rx” 表示 6 字节读取操作，“F0*” 代表单个 ODR 时隙（为清晰起见，示意图进行了放大）。

10 辅助 ADC

LIS3DH 配有的辅助 10 位 ADC 转换器在器件的 ADC1、 ADC2 和 ADC3 输入引脚上复用，从而可将陀螺仪模拟量输出等外部模拟信号转换为数字信号。随后可通过读取寄存器 OUT_ADC1_L、 OUT_ADC1_H、 OUT_ADC2_L、 OUT_ADC2_H、 OUT_ADC3_L 和 OUT_ADC3_H 读取器件中的数字化数据。转换状态会在 STATUS_AUX (07h) 寄存器中报告；辅助 ADC 数据准备就绪后， STATUS_AUX 寄存器的 321DA 位会置 1。

可通过将 CTRL_REG3 的 I1_DRY2 位置 1 的方式将 321DA 位的状态驱动到 INT1 焊盘。

将 TEMP_CFG_REG (1Fh) 的 ADC_PD 位置 1 会使能辅助 ADC。

10.1 温度传感器

LIS3DH 配有内部温度传感器。将 TEMP_CFG_REG 寄存器的 TEMP_EN 位置 1 可使能温度数据。如果使能了辅助 ADC 和温度传感器， ADC 的第三条通道会用于对温度传感器输出进行数字化处理，并可在 OUT_3_L 和 OUT_3_H 寄存器中以 2 的补码数据形式进行读取。

11 版本历史

表 40. 文档版本历史

日期	版本	变更
2011 年 1 月 14 日	1	初始版本。

表 41. 中文文档版本历史

日期	版本	变更
2017 年 5 月 10 日	1	中文初始版本。

重要通知 - 请仔细阅读

意法半导体公司及其子公司 （“ST”）保留随时对 ST 产品和 / 或本文档进行变更、更正、增强、修改和改进的权利，恕不另行通知。买方在订货之前应获取关于 ST 产品的最新信息。 ST 产品的销售依照订单确认时的相关 ST 销售条款。

买方自行负责对 ST 产品的选择和使用， ST 概不承担与应用协助或买方产品设计相关的任何责任。

ST 不对任何知识产权进行任何明示或默示的授权或许可。

转售的 ST 产品如有不同于此处提供的信息的规定，将导致 ST 针对该产品授予的任何保证失效。

ST 和 ST 徽标是 ST 的商标。所有其他产品或服务名称均为其各自所有者的财产。

本文档中的信息取代本文档所有早期版本中提供的信息。本文档的中文版本为英文版本的翻译件，仅供参考之用；若中文版本与英文版本有任何冲突或不一致，则以英文版本为准。