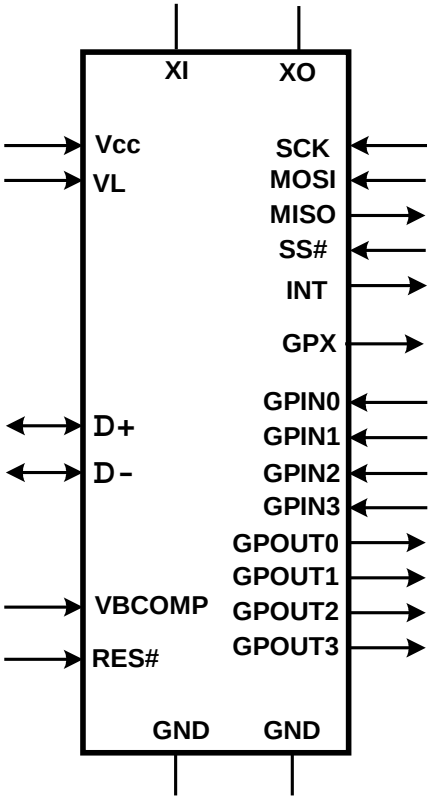
MAX3420E

SPI 接口的 USB 外设控制器

编程指南



欲了解更多关于 MAX3420E 的信息，请访问 http://www.maxim-ic.com.cn/max3420e.
欲了解更多关于 USB 及 Maxim 公司 USB 产品的信息，请访问 http://www.maxim-ic.com.cn/usb.

Rev. Sep 28, 2005

寄存器映射表

位名用正常字体表示，寄存器名用斜体字表示。
每个突出显示的单元都链接到一个对应的寄存器或位说明页。点击浏览器左箭头可返回本页。本文末尾的索引也链接到相关的说明页。

Reg	Name	b7	b6	b5	b4	b3	b2	b1	b0	acc
R0	*EP0FIFO*	b7	b6	b5	b4	b3	b2	b1	b0	RSC
R1	*EP1OUTFIFO*	b7	b6	b5	b4	b3	b2	b1	b0	RSC
R2	*EP2INFIFO*	b7	b6	b5	b4	b3	b2	b1	b0	RSC
R3	*EP3INFIFO*	b7	b6	b5	b4	b3	b2	b1	b0	RSC
R4	*SUDFIFO*	b7	b6	b5	b4	b3	b2	b1	b0	RSC
R5	*EP0BC*	0	b6	b5	b4	b3	b2	b1	b0	RSC
R6	*EP1OUTBC*	0	b6	b5	b4	b3	b2	b1	b0	RSC
R7	*EP2INBC*	0	b6	b5	b4	b3	b2	b1	b0	RSC
R8	*EP3INBC*	0	b6	b5	b4	b3	b2	b1	b0	RSC
R9	**EPSTALLS**	0	ACKSTAT	STLSTAT	STLEP3IN	STLEP2IN	STLEP1OUT	STLEP0OUT	STLEP0IN	RSC
R10	**CLRTOGS**	EP3DISAB	EP2DISAB	EP1DISAB	CTGEP3IN	CTGEP2IN	CTGEP1OUT	0	0	RSC
R11	**EPIRQ**	0	0	SUDAVIRQ	IN3BAVIRQ	IN2BAVIRQ	OUT1DAVIRQ	OUT0DAVIRQ	IN0BAVIRQ	RC
R12	**EPIEN**	0	0	ACKSTAT	IN3BAVIE	IN2BAVIE	OUT1DAVIE	OUT0DAVIE	IN0BAVIE	RSC
R13	**USBIRQ**	URESDNIRQ	VBUSIRQ	NOVBUSIRQ	SUSPIRQ	URESIRQ	BUSACTIRQ	RWUDNIRQ	OSCOKIRQ	RC
R14	**USBIEN**	URESDNIE	VBUSIE	NOVBUSIE	SUSPIE	URESIE	BUSACTIE	RWUDNIE	OSCOKIE	RSC
R15	**USBCTL**	HOSCSTEN	VBGATE	CHIPRES	PWRDOWN	CONNECT	SIGRWU	0	0	RSC
R16	**CPUCTL**	0	0	0	0	0	0	0	IE	RSC
R17	**PINCTL**	EP3INAK	EP2INAK	EP0INAK	FDUPSPI	INTLEVEL	POSINT	GPXB	GPXA	RSC
R18	**REVISION**	0	0	0	0	Rev3	Rev2	Rev1	Rev0	R
R19	*FNADDR*	0	b6	b5	b4	b3	b2	b1	b0	R
R20	**IOPINS**	GPIN3	GPIN2	GPIN1	GPIN0	GPOUT3	GPOUT2	GPOUT1	GPOUT0	RSC

注：acc（访问）列表示 CPU 访问寄存器的方式。R=读，RC=读或清除，RSC=读、置位或清除。

访问 MAX3420E 的内部寄存器

SPI™主控制器通过读、写 MAX3420E 的 21 个内部寄存器 R0-R20 对其进行控制。SPI 主控制器通过触发 MAX3420E 的 SS#引脚（从器件选择，低有效）启动每次寄存器访问，并由时钟打入八位 SPI 命令字节。命令字节格式如图 1 所示。

b7	b6	b5	b4	b3	b2	b1	b0
Reg4	Reg3	Reg2	Reg1	Reg0	0	DIR 1=wr 0=rd	ACKSTAT

图 1. SPI 命令字节。所有 SPI 传输都是先发送 bit 7。

Reg4:Reg0 设置寄存器地址，有效值为 0-20。大于 20 的值被 MAX3420E 忽略。方向位（DIR）设置数据传输的方向。ACKSTAT 是一个 USB 控制位（R9 中的 bit 6）的拷贝。由于 ACKSTAT 是一个频繁使用的寄存器位，在控制字节中提供一个拷贝以便于快速访问。

发送完命令字节之后，SPI 主控制器按 DIR 位规定的方向传输一个或多个字节。保持 SS#为低，SPI 主控制器每发出八个 SCLK 脉冲可完成一个字节的传输。所有字节传输完成后，SPI 主控制器撤销 SS#（驱动为高），终止传输过程。

发送完命令字节后也可以截止 SPI 访问。这一特性允许 SPI 主控制器仅仅设置 ACKSTAT 位，而无需完成整个 SPI 访问。

注意：ACKSTAT 位告诉 MAX3420E，SPI 主控制器已完成对 USB Control 传输的处理。这使 MAX3420E 应答下一个 Control 传输的 STATUS 阶段。

用 SPI 命令字节置位 ACKSTAT 位时，SPI 主控制器将寄存器区设为空值（不会被使用），设置 DIR 位为读操作（例如，读 Revision 寄存器 R18），并设置 ACKSTAT 位。然后，SPI 主控制器发出 SS#信号，输入 8 个命令位，然后撤销 SS#信号，终止 SPI 周期。这是设置 ACKSTAT 的最快的方法。

MAX3420E 有两种寄存器类型，FIFO 和控制寄存器。根据寄存器类型的不同，反复读或写寄存器会产生不同的效果。

寄存器 R0-R4 用来访问内部 FIFO。在用命令字节选择了寄存器号 R0-R4 之后，SPI 主控制器在 SPI 传输期间通过重复读或写操作来加载或卸载连续的 FIFO 字节。例如，如需从 SUDFIFO 读 8 个字节，SPI 主控制器可执行以下步骤：

1. 设置 SS#=0。
2. 发送 00100000。这条命令字节选择 R4（SUDFIFO）用于读操作（DIR=0）。
3. 发 8 个 SCLK 脉冲，读入一个数据字节，每个 SCLK 上升沿输入一位。
4. 重复步骤 3 七次，输入并存储 8 个字节。
5. 设置 SS#=1。

寄存器 R5-R20 为控制寄存器。如果 SPI 主控制器在同一个 SPI 传输周期（SS#为低）中反复读或写 R5-R20，每个字节的读或写操作将自动增加寄存器地址。这样无需写入新的命令字节来设置每次新寄存器的地址，就可以连续读或写寄存器。寄存器地址连续增加，直至到达最后一个寄存器 R20，此时寄存器地址将"粘着"于 R20。这个特性使μP 可以快速通过 R20 访问 IO 引脚。例如，为了向 GPIO 引脚发送预存波形，SPI 主控制器可以发命令字节 10100010（写 R20），然后向 R20 发送多个数据字节输出波形。

ACKSTAT

含义： 应答 CONTROL 传输的 STATUS 阶段

位置： EPSTALLS.6

置位： CPU 在其完成对 CONTROL 传输请求的服务后设置该位。这将令 SIE 向当前 CONTROL 传输的 STATUS 阶段发送 ACK 握手包。在 CPU 设置该位之前，SIE 以 NAK 握手包响应 CONTROL 传输的 STATUS 阶段。

清除： 接收到 SETUP 令牌后，SIE 立即清除该位。

上电复位: ACKSTAT=0
芯片复位： ACKSTAT=0
总线复位： ACKSTAT=0
掉电: 只读

FYI: 设置 ACKSTAT 寄存器位的快捷方法是设置 SPI 命令字节的 bit 0。所有 Maxim 例程都使用了这一方法。

编程要点：

CPU 收到 Setup 数据就绪中断请求（SUDAVIRQ 位，第 66 页）时，向其写“1”清除 SUDAVIRQ，然后从 SUDFIFO 读取 8 个数据字节到存储器。然后 CPU 检查这 8 个字节以确定 USB 请求的种类。如果请求有误或者未知，CPU 设置 STLSTAT 位，以 STALL 握手包响应 STATUS 阶段（第 64 页）。

如果 CPU 识别该请求，它将处理该请求，完成处理后设置 ACKSTAT=1，通知 SIE 给 STATUS 阶段发送 ACK 握手包，终止 CONTROL 传输。在 CPU 响应或停止传输前，SIE 自动给 CONTROL 传输的状态阶段返回 NAK 握手包。

用来解释 8 字节 SETUP 数据的 C 程序通常为一个或多个 **case** 语句，它检查 SETUP 包内字节所有合法的组合。处理 STALL 的一个简便方法是使默认的 **case** 状态用来停止 CONTROL 传输（参见 STLSTAT 位，第 64 位）。

BUSACTIE

含义： 总线活动中断使能

位置： USBIEN.2

置位： CPU 置位该位以使能 BUSACT IRQ （第 2 页）。

清除： CPU 清除该位以禁止 BUSACT IRQ。

上电复位: BUSACTIE=0
芯片复位： BUSACTIE=0
总线复位： BUSACTIE=0
掉电: 只读

编程要点:

由于 USB 总线复位期间会清除大多数中断使能位，因此，作为 USB 总线复位处理的一部分，每次复位后都应调用一个打开所有中断使能位的初始化例程。

BUSACTIRQ

含义： 总线活动中断请求

位置： USBIRQ.2

置位： SIE 置位该位表示 USB 总线有活动。当 SIE 收到 SYNC（同步）域时，内部 BUSACT 信号被置位，该信号在 32 位长的 J 状态或 USB 总线复位期间复位。当内部 BUSACT 信号由 0 变 1 时 BUSACTIRQ 位被置位。

清除： CPU 向该位写“1”可清除该位。

上电复位： BUSACTIRQ=0
芯片复位： BUSACTIRQ=0
总线复位： BUSACTIRQ=0
掉电： 只读

CHIPRES

含义： 芯片复位

位置： USBCTL.5

置位： CPU 将该位置位以复位器件。其作用等效于将 RES#引脚拉低。

清除： CPU 清除该位可使芯片脱离复位状态。

上电复位: CHIPRES=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：

CPU 可以在置位该位后立即将它清除。

CONNECT

含义： 连接至 USB

位置： USBCTL.3

置位： CPU 将该位置位，连接 DPLUS 线和 V_{CC} 之间的 1500Ω 内部电阻。

清除： CPU 清除该位，断开 DPLUS 线和 V_{CC} 之间的 1500Ω 内部电阻。

上电复位: CONNECT=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：

CONNECT 位的工作和 VBGATE 位（76 页）的设置有关。如果 CONNECT=1 且 VBGATE=1，内部逻辑不连接上拉电阻，除非引脚 V_{BUS} 上检测到有效的 V_{BUS} 电压。如果 VBGATE=0，当 CONNECT=1 时，DPLUS 上拉电阻将被无条件连接。

只有上电复位操作可以清除 CONNECT 位。如果在工作期间通过 INT 引脚给 MAX3420E 施加外部复位，CONNECT 位状态仍然保持不变。这意味着如果器件被连接到 USB，当发出 RES#信号后，它将在整个复位期间和复位后（RES#=1）保持连接。

CTGEP1OUT

含义： 清除端点 1 OUT 数据触发。

位置： CLRTOGS.2

置位： CPU 将该位置位，清除 EP1-OUT 数据触发至 DATA0 状态。

清除： SIE 自动清除该位。

上电复位: CTGEP1OUT=0
芯片复位： CTGEP1OUT=0
总线复位： CTGEP1OUT=0
掉电: 只读

编程要点：

SIE 在芯片或 USB 总线复位期间自动清除所有数据触发。在以下两种情况下，CPU 通常需要单独清除各端点的数据触发：

- 主机发出 Set_Configuration 请求。
- 主机发出 Clear_Feature（端点停止）请求。

CTGEP2IN

含义： 清除端点 2 IN 数据触发。

位置： CLRTOGS.3

置位： CPU 将该位置位以清除 EP2-IN 数据触发至 DATA0 状态。

清除： SIE 自动清除该位。

上电复位： CTGEP2IN=0
芯片复位： CTGEP2IN=0
总线复位： CTGEP2IN=0
掉电： 只读

编程要点：

SIE 在芯片或 USB 总线复位期间自动清除全部数据触发。在以下两种情况下，CPU 通常需要单独清除各端点的数据触发：

- 主机发出 Set_Configuration 请求。
- 主机发出 Clear_Feature （端点停止）请求。

CTGEP3IN

含义： 清除端点 3 IN 数据触发。

位置： CLRTOGS.4

置位： CPU 将该位置位以清除 EP3-IN 数据触发至 DATA0 状态。

清除： SIE 自动清除该位。

上电复位: CTGEP3IN=0
芯片复位： CTGEP3IN=0
总线复位： CTGEP3IN=0
掉电: 只读

编程要点：

在芯片或 USB 总线复位期间 SIE 自动清除全部数据触发。在以下两种情况下，CPU 通常需要单独清除各端点的数据触发：

- 主机发出 Set_Configuration 请求。
- 主机发出 Clear_Feature （端点停止）请求。

EP0BC

含义： 端点 0 字节计数寄存器。由于 EP0 是双向端点，因此 IN 和 OUT 传输共享同一个 FIFO（EP0FIFO，第 10 页），寄存器的动作取决于传输方向。

位置： EP0BC[6:0]

写（IN）： 对于 IN 传输，CPU 在把数据装入 EP0FIFO 之后，将字节数写入该寄存器。有效值为 0-64。CPU 写寄存器后，SIE 配备给该端点，这样，当有对于该端点的下一个 IN 请求时，将以数据包而不是 NAK 包响应。

读（OUT）： 对于 OUT 传输，SIE 装载字节计数，指示 OUT 传输中接收的字节数。如果传输成功，SIE 以 ACKS 响应传输，更新字节计数寄存器，并置位 OUT0DAV 中断请求位（第 51 页）。

上电复位: EP0BC=0
芯片复位： EP0BC=0
总线复位： EP0BC=0
掉电: 不能读、写

编程要点：

Bit 7 没有作用，读数为 0。
CPU 写 EP0FIFO 来响应 CONTROL 传输的数据阶段。
CPU 读 EP0FIFO 来获得 CONTROL 传输的数据阶段。

EP0FIFO

含义： 端点 0 FIFO。该 64 字节的 FIFO 用于实现和双向端点 0 之间的 OUT 和 IN 传输。

位置： EP0FIFO[7:0]

写（IN）： 对于 IN 传输，CPU 把一系列字节写入该 FIFO 来填充 IN 数据。在 FIFO 填满数据包（0 到 64 个字节）之后，CPU 写字节数寄存器（第 9 页）以提供 IN 传输，并告知 SIE 在收到对端点 0 的 IN 包时需传输多少个字节。

读（OUT）： 对于 OUT 传输，SIE 用来自主机的 USB 数据填充 FIFO。如果 OUT 传输经过验证后是准确无误的，SIE 装载字节数寄存器（第 9 页），以表明 OUT 数据传输中接收到的字节数。对于成功传输，SIE 也会响应 OUT 传输以 ACKS 包，并置位 OUT0DAV 中断请求位（第 51 页）。

上电复位： EP0FIFO[7:0]=0
芯片复位： EP0FIFO[7:0]=0
总线复位： *不变*
掉电： 不能读、写

编程要点：

当发现出错时（CRC、位填充等等），SIE 可自动再试。这对 CPU 来说是不可见的——无中断标志或寄存器更新。

EP0INAK

含义： EP0-IN NAK

位置： PINCTL.5

置位： 当 EP0-IN 端点接收到 IN 请求并返回 NAK 握手时，SIE 将该位置位。

清除： CPU 向该位写 1 以清除该位。

上电复位： EP0IBN=0
芯片复位： EP0IBN=0
总线复位： EP0IBN=0
掉电： 读-写

编程要点：

可以查询该位以检查主机是否在请求 IN 数据，而 CPU 尚未装载并配置该端点。该位不包括在中断系统中。

注意：EP0INAK 位只是信息位。USB 设备固件通常不使用该位。

EP1DISAB

含义： 禁止端点 1-OUT

位置： CLRTOGS.5

置位： CPU 将该位置位以禁止到端点 1-OUT 的通信。

清除： CPU 清除该位以使能与端点 1-OUT 的通信。

上电复位： EP1DISAB=0
芯片复位： EP1DISAB=0
总线复位： EP1DISAB=0
掉电： 只读

编程要点：

禁用的端点不会响应任何通信。主机通常不会和在枚举期间未报告的端点通信，因此，它是一个针对错误的主机提供保护的“安全”位。

端点 0 没有禁止位，因为作为默认的 CONTROL 端点，它必须总是工作的。

EP1OUTBC

含义： 端点 1-OUT 字节数。

位置： EP1OUTBC[6:0]

写: 在成功通过端点 1 接收到 OUT 传输之后，SIE 以 ACKS 响应传输，同时以接收到的字节数更新该寄存器，并置位 OUT1DAV 中断请求（第 51 页）。

读: CPU 在接收到 OUT1DAV 中断请求之后读该寄存器，以确定需从 EP1OUTFIFO（第 14 页）读取的字节数。

上电复位: EP1OUTBC=0
芯片复位： EP1OUTBC=0
总线复位： EP1OUTBC=0
掉电: 不能读、写

编程要点：

EP1OUT 是双缓冲端点，这意味着有两个 FIFO 和字节数寄存器。当 CPU 从一个 FIFO 读数据的同时，USB 数据可以被装入另一个 FIFO。这样改善了系统的带宽。OUT 端点的双缓冲工作方式的描述参见关于 OUT1DAVIRQ 位的讨论 （第 51 页）。

由于 OUT1DAVIRQ 逻辑对于双缓冲的适配，双缓冲对编程者来说是不可见的。例如，假定两个缓冲都是可用的，则 OUT1DAVIRQ=0。当有一个 OUT 包到达时，OUT1DAVIRQ 由 0 变 1，表明第一个包可用。对于单缓冲端点，在 CPU 释放 FIFO 以前，如果又有另一个包通过 EP1-OUT 到达，SIE 将发送 NAK 握手，表明端点无法接收数据。

然而，有了双缓冲端点，由于第二个缓冲可以接收数据，因此可以接收并以 ACK 应答第二个 OUT 包。如果在两个 FIFO 释放以前第三个 OUT 包到达，SIE 以 NAK 响应传输，表明两个 FIFO 都已满。

EP1OUTFIFO

含义： 端点 1-OUT 双缓冲 64 字节 FIFO

位置： EP1OUTFIFO[7:0]

写: SIE 用主机传输到端点 1-OUT 的数据填充 OUT FIFO。在成功收到 OUT 传输之后，SIE 发 ACK 应答传输，更新字节数寄存器（第 13 页），并发出 OUT1DAV 中断请求（第 51 页）。

读: 当 CPU 接收到 OUT1DAV 中断请求时，读取字节数寄存器以确定 FIFO 内字节的个数，然后读取相应字节数的数据。

上电复位: EP1OUTFIFO=0
芯片复位： EP1OUTFIFO=0
总线复位： *不变*
掉电: 不能读、写

编程要点：

EP1OUT 是双缓冲端点，这意味着有两个 FIFO 和字节数寄存器。当 CPU 从一个 FIFO 读出数据时，USB 数据可同时送入另外一个 FIFO。这样改善了系统的带宽。OUT 端点的双缓冲工作方式的描述参见关于 OUT1DAVIRQ 位的讨论 （第 53 页）。

由于 OUT1DAVIRQ 逻辑对于双缓冲的适配，使得双缓冲对编程者不可见。例如，假定两个缓冲都是可用的，则 OUT1DAVIRQ=0。当 OUT 包到达时，OUT1DAVIRQ 由 0 变 1，表明第一个包可用。对于单缓冲端点来说，在 CPU 释放 FIFO 以前，如果有另一个包通过 EP1-OUT 到达，SIE 将以 NAK 应答，表明端点无法接收数据。

然而，有了双缓冲端点后，由于第二个缓冲可以接收数据，因此可以接收并以 ACK 应答第二个 OUT 包。如果在两个 FIFO 释放以前，接收到第三个 OUT 包，那么 SIE 将以 NAK 应答传输，表明两个 FIFO 都是满的。

EP2DISAB

含义： 禁止端点 2

位置： CLRTOGS.6

置位： CPU 将该位置位以禁止同端点 2-IN 的通信。

清除： CPU 清除该位以使能同端点 2-IN 的通信。

上电复位： EP2DISAB=0
芯片复位： EP2DISAB=0
总线复位： EP2DISAB=0
掉电： 只读

编程要点：

禁止的端点不会对任何通信作出响应。主机通常不会和枚举期间未报告的端点通信，因此，该位可用于针对错误主机提供保护的“安全”位。

端点 0 没有禁止位，因为作为默认的 CONTROL 端点，它必须总是工作的。

EP2INAK

含义： 端点 2-IN NAK

位置： PINCTL.6

置位： 当 EP2-IN 端点接收到 IN 请求并且 SIE 返回 NAK 握手信号时，SIE 将该位置位。

清除： CPU 写 1 以清除该位。

上电复位： EP2INAK=0
芯片复位： EP2INAK=0
总线复位： EP2INAK=0
掉电： 读-写

编程要点：

可以查询该位以检查主机是否在请求 IN 数据，但由于 CPU 还未装载并配置端点，还未获得这些数据。该位不包括在中断系统中。

注意：EP2INAK 位只是信息位。USB 设备内的固件通常不使用该位。

EP2INBC

含义： 端点 2-IN 字节数寄存器

位置： EP2INBC[6:0]

写： CPU 将已装入 EP2INFIFO（第 18 页）的字节数装入该寄存器。为下一个 IN 传输配置端点。

读： SIE 发送 FIFO 中的数据，作为对 EP2-IN 主机请求的应答。

上电复位: EP2INBC=0
芯片复位： EP2INBC=0
总线复位： EP2INBC=0
掉电: 只读

EP2INFIFO

含义： 端点 2-IN FIFO （双缓冲 64 字节 FIFO）

位置： EP2INFIFO[7:0]

写： CPU 将数据字节载入该 FIFO，为向主机发送数据做准备。

读： SIE 通过 USB 发送这些字节，做为对 EP2-IN IN 请求的响应。

上电复位: EP2INFIFO=0
芯片复位： EP2INFIFO=0
总线复位： *不变*
掉电: 不能读、写

EP3DISAB

含义： 禁止端点 3

位置： CLRTOGS.7

置位： CPU 将该位置位以禁止同端点 3-IN 的通信。

清除： CPU 清除该位以使能同端点 3-IN 的通信。

上电复位: EP3DISAB=0
芯片复位： EP3DISAB=0
总线复位： EP3DISAB=0
掉电: 只读

编程要点：

被禁止的端点不会对任何通信作出响应。主机通常不会给在枚举期间未报告的端点发送信号，因此该位用于预防主机错误的“安全”位。

端点 0 没有禁止位，因为作为默认的 CONTROL 端点，它必须总是有效的。

EP3INAK

含义： 端点 3-IN NAK

位置： PINCTL.7

置位： 当 EP3-IN 端点接收到 IN 请求并且 SIE 返回 NAK 握手信号时，SIE 将该位置位。

清除： CPU 写 1 以清除该位

上电复位： EP3INAK=0
芯片复位： EP3INAK=0
总线复位： EP3INAK=0
掉电： 读-写

编程要点：

可以查询该位以检查主机是否在请求 IN 数据，而这些 IN 数据还未被 CPU 载入并配置端点。该位不包括在中断系统中。

注意：EP0INAK 位只是信息位。USB 设备固件通常不使用该位。

EP3INBC

含义： 端点 3-IN 字节数寄存器

位置： EP3INBC[6:0]

写： CPU 将已经装入 EP3INFIFO（第 22 页）的字节数装入该寄存器。为下一次 IN 传输做好准备。

读： SIE 发送 FIFO 中的数据，作为对 EP3-IN 主机请求的应答。

上电复位: EP3INBC=0
芯片复位： EP3INBC=0
总线复位： EP3INBC=0
掉电: 只读

EP3INFIFO

含义： 端点 3-IN FIFO，64 字节 FIFO

位置： EP3INFIFO[7:0]

写： CPU 将数据字节载入该 FIFO，为向主机发送数据做准备。

读： SIE 通过 USB 发送这些字节，做为对 EP3-IN IN 请求的响应。

上电复位: EP3INFIFO=0
芯片复位： EP3INFIFO=0
总线复位： *不变*
掉电: 不能读、写

FDUPSPI

含义： 全双工 SPI 端口工作模式

位置： PINCTL.4

置位： CPU 将该位置位，使 SPI 端口工作于全双工模式。

清除： SIE 清除该位，工作于半双工模式。

上电复位: FDUPSPI=0 （半双工）
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：

半双工 SPI

在半双工模式下（FDUPSPI=0），MOSI（主控制器输出，从机输入）引脚变为双向 IO 引脚，MISO（主控制器输入，从机输出）变为三态引脚。

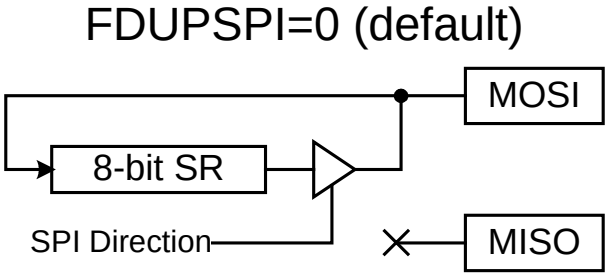


图 2. 半双工 SPI 接口

全双工 SPI

全双工模式（FDUPSPI=1）具有单独的 MOSI 和 MISO 引脚。该架构具有以下附加特性：每次传输第一个字节（命令字节）输入时，同时输出 8 位状态位，如图 3 所示。

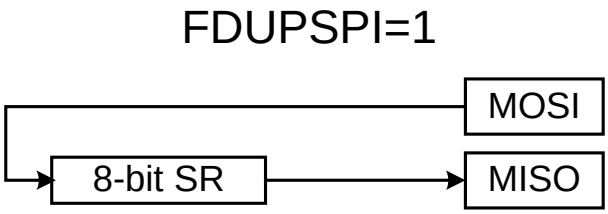


图 3. 全双工 SPI 接口

MISO bit	Signal	page
7	SUSPIRQ	**69**
6	URESIRQ	**72**
5	SUDAVIRQ	**66**
4	IN3BAVIRQ	**44**
3	IN2BAVIRQ	**42**
2	OUT1DAVIRQ	**53**
1	OUT0DAVIRQ	**51**
0	IN0BAVIRQ	**40**

图 4. 全双工模式：当命令字节输入时，这些 USB 状态位输出。

SPI 命令字节

在两种 SPI 模式下，进入 SPI 接口的第一个字节总是命令字节，用来设置寄存器地址、传送方向、以及直接设置 ACKSTAT 位。在整个 SPI 传输中，无论输入输出，位序均是 b7 在最前，b0 在最后。

MOSI bit	Signal
7	REG4
6	REG3
5	REG2
4	REG1
3	REG0
2	0
1	Direction （ 1=Wr, 0=Rd）
0	ACKSTAT （page 1）

图 5. SPI 命令字节

一个SPI周期从SPI主控制器驱动CS#为低开始，然后输入8个SPI时钟，在时钟上升沿读入如图5所示的指令字节内的数据。REG[4:0]设置寄存器地址，方向位确定该SPI周期的读写方向。ACKSTAT写入EPSTALLS 寄存器的相应位。如果FDUPSPI=1，在前8个SCK时钟期间图4中的数据同时输出到MISO引脚。

在命令字节之后，SPI主控制器又发出一组或多组时钟，每组时钟由8个SCLK脉冲组成,向MAX3420E输入或输出数据字节。只要CS#保持低电平，通过命令字节打入的寄存器地址始终有效。这样可以很方便地从端点FIFO中突发读写多个字节。例如，要把37个字节装入EP0FIFO，SPI主控制器写入命令字节00000010，选择R0（EP0FIFO）进行写操作（方向位为1） 。然后向SPI端口写入37个字节，最后驱动CS#为高完成SPI周期。

注意：MOSI和MISO数据在SCLK上升沿采样。数据在SCK下降沿发生变化。

当SPI主控制器驱动CS#返回高电平时SPI周期结束。

SPI 模式

SPI标准定义了4种时钟模式，表现为两种信号模式：CPOL（时钟极性）和CPHA（时钟相位），以（CPOL，CPHA）形式表示。实际上，要求正沿SCK并且要求MOSI数据先于第一个时钟正沿有效的接口可以不加调整地工作在模式（0,0）和（1,1）下。这一特性使MAX3420E无需模式引脚即可工作在模式（0,0）或（1,1）下。

下面的示波器曲线中微处理器和MAX3420E之间传输相同的数据。图6采用SPI模式（0,0），图7采用SPI模式（1,1）。 两者的区别在于SCLK信号的无效电平，模式（0,0）时为低，模式（1,1）时为高。两种模式下，在SCLK上升沿采样的MOSI和MISO数据是相同的。

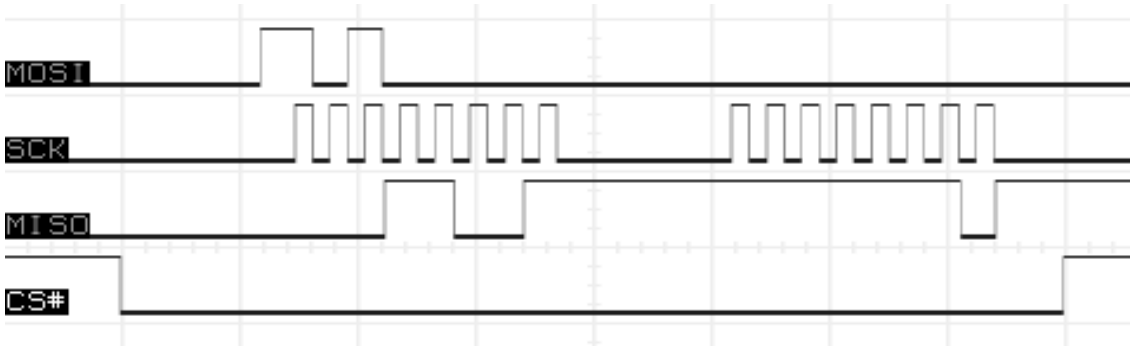


图 6. 工作在（0,0）模式下的 SPI 接口

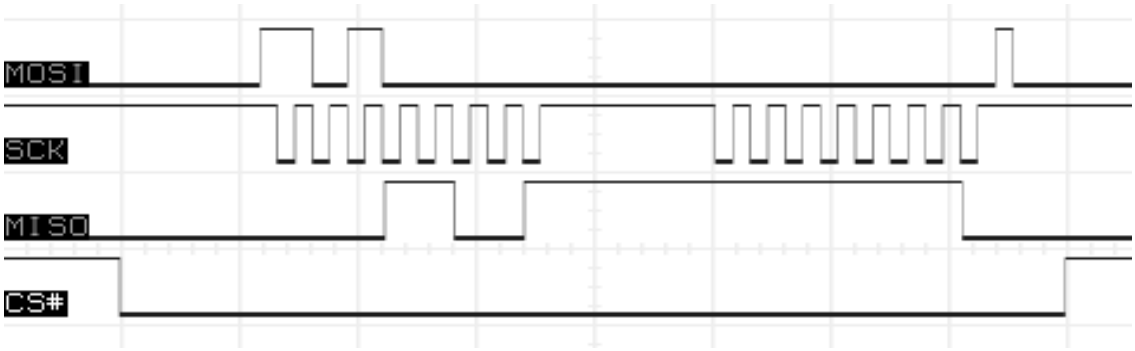


图 7. 工作在（1,1）模式下的 SPI 接口

FNADDR

含义： 主机分配给USB外设的功能地址。

位置： FNADDR[6:0]

置位： 主机发出Set_Address 请求后，SIE在接收到ACK握手后更新该寄存器。

清除： SIE 在芯片复位或 USB 总线复位期间清除该寄存器。

上电复位： FNADDR=0
芯片复位： FNADDR=0
总线复位： FNADDR=0
掉电： 只读

编程要点：

CPU 写该寄存器无效。

GPIN0

含义： 通用输入引脚 0

位置： IOPINS.4

置位： 该引脚指示 GPIN0 引脚的状态（以 V_L 为参照）。该引脚通过一个弱上拉电阻被拉高，因此该引脚悬空时为逻辑 1。

清除： 写该位无效

上电复位: 不定
芯片复位： 不定
总线复位： 不定
掉电: 只读

GPIN1

含义： 通用输入引脚 1

位置： IOPINS.5

置位： 该引脚指示 GPIN1 引脚的状态（以 V_L 为参照）。该引脚通过一个弱上拉电阻被拉高，因此该引脚悬空时为逻辑 1。

清除： 写该位无效。

上电复位: 不定
芯片复位： 不定
总线复位： 不定
掉电: 只读

GPIN2

含义： 通用输入引脚 2

位置： IOPINS.6

置位： 该引脚指示 GPIN2 引脚的状态（以 V_L 为参照）。该引脚通过一个弱上拉电阻被拉高，因此该引脚悬空时为逻辑 1。

清除： 写该位无效。

上电复位: 不定
芯片复位： 不定
总线复位： 不定
掉电: 只读

GPIN3

含义： 通用输入引脚 3

位置： IOPINS.7

置位： 该引脚指示 GPIN3 引脚的状态（以 V_L 为参照）。通过一个弱上拉电阻被拉高，因此该引脚悬空时为逻辑 1。

清除： 写该位无效。

上电复位: 不定
芯片复位： 不定
总线复位： 不定
掉电: 只读

GPOUT0

含义： 通用输出引脚 0

位置： IOPINS.0

置位： CPU 通过将该位置位来设置 GPOUT0 引脚为高（以 V_L 为参照）。CPU 也可以读该位。读该位可知道输出缓冲器之前的输出触发器的状态。因此，如果输出引脚驱动大的负载（如 LED），逻辑电平可能受影响，但 CPU 仍然能读取输出引脚的正确逻辑状态。

清除： CPU 清除该位以将 GPOUT0 引脚状态设置为 0。

上电复位: GPOUT0=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

GPOUT1

含义： 通用输出引脚 1

位置： IOPINS.1

置位： CPU 通过将该位置位来设置 GPOUT1 引脚为高（以 V_L 为参照）。CPU 也可以读该位。读该位可知道输出缓冲器之前的输出触发器的状态。因此，如果输出引脚驱动大负载（如 LED），逻辑电平可能会受到影响，但 CPU 仍然能读取输出引脚的正确逻辑状态。

清除： CPU 清除该位以将 GPOUT1 引脚状态设置为 0。

上电复位: GPOUT1=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

GPOUT2

含义： 通用输出引脚 2

位置： IOPINS.2

置位： CPU 通过将该位置位来设置 GPOUT2 引脚为高（以 V_L 为参照）。CPU 也可以读该位。读该位可知道位于输出缓冲器之前的输出触发器的状态。因此，如果输出引脚驱动大负载（如 LED），逻辑电平可能会受到影响，但 CPU 仍然能读取输出引脚的正确逻辑状态。

清除： CPU 清除该位以将 GPOUT2 引脚状态设置为 0。

上电复位： GPOUT2=0
芯片复位： *不变*
总线复位： *不变*
掉电： 读-写

GPOUT3

含义： 通用输出引脚 3

位置： IOPINS.3

置位： CPU 通过将该位置位来设置 GPOUT3 引脚为高（参考 V_L）。CPU 也可以读该位。读该位可知道位于输出缓冲器之前的输出触发器的状态。因此，如果输出引脚驱动大负载（如 LED），逻辑电平可能会受到影响，但 CPU 仍然能读取输出引脚的正确逻辑状态。

清除： CPU 清除该位以将 GPOUT3 引脚状态设置为 0。

上电复位: GPOUT3=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

GPXA

含义： GPXB:GPXA 两个位用来确定 GPX 引脚的输出信号。

位置： PINCTL.0

置位： CPU 将该位置位。

清除： CPU 清除该位。

上电复位: GPXA=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：

GPXB	GPXA	GPX Pin
0	0	OPERATE （POR 的反）
0	1	VBUS 检测
1	0	BUSACT
1	1	SOF （SOF 包到达时从 0 跳到 1，50% 占空比）

GPXB

含义： GPXB:GPXA 两个位确定 GPX 引脚的输出信号。

位置： PINCTL.1

置位： CPU 将该位置位。

清除： CPU 清除该位。

上电复位: GPXB=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：

GPXB	GPXA	GPX Pin
0	0	OPERATE （POR 的反）
0	1	VBUS 检测
1	0	BUSACT
1	1	SOF （SOF 包到达时从 0 跳到 1，50% 占空比）

HOSCSTEN

含义： 使能主机启动振荡器。当 MAX3420E 处于掉电模式（PWRDOWN=1）时起作用。

位置： USBCTL.7

置位： CPU 置位该位时，当 USB DPLUS 产生由 1 变 0 的跳变时（主机恢复信号）启动片上振荡器。

清除： CPU 清除该位时，当 USB 主机恢复总线信号时芯片保持低功耗状态（禁止振荡器启动）。

上电复位: HOSCSTEN=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：

一旦 CPU 将 MAX3420E 置于掉电状态（PWRDOWN=1，第 55 页），有 3 种方法可以重新启动 MAX3420E 振荡器：

1. SPI 主控制器设置 PWRDOWN=0（也可以通过芯片复位实现）。
2. SPI 主控制器设置 SIGRESUME=1（ 第 58 页）。
3. USB 主机恢复总线活动。在低功耗模式下（振荡器关闭），MAX3420E 通过 DPLUS 数据线上的低电平来检测此状态。

当 MAX3420E 用于自供电外设时，HOSCSTEN 位用于上述第三种情况。假定用户将处于掉电模式的自供电外设插入已关机的 PC。DPLUS 上的逻辑低（ 由主 PC 根集线器上的 DPLUS 下拉电阻造成）看起来就像 USB RESUME 信号，这样一来通常会重新启动振荡器并唤醒芯片。但是在这种情况下，在 PC 打开 V_{BUS} 激活 USB 端口以前，芯片不应该通电或者为 DPLUS 电阻供电。

因此在自供电系统中，为使 USB 处于挂起状态，CPU 应设置 PWRDOWN=1、HOSCSTEN=0。这使系统进入掉电状态，并且 DPLUS 线上的 1 到 0 跳变不唤醒芯片。

对于自供电外设，将 GPXB-A 引脚（第 36 页）设置为 01，可以方便地检测主机 PC 是否上电并打开 VBUS。该设置把 V_{BUS} 比较器输出连接至 GPX 引脚。然后，通过 GPX 引脚 CPU 便可直接检测 V_{BUS}。未连接时，CPU 应该设置 CONNECT=0（第 5 页），然后在检测到有效的 USB 连接之后，才把它设置为 1。

IE

含义： 中断使能

位置： CPUCTL.0

置位： CPU 将该位置位以激活 INT 输出引脚。INT 输出引脚的特性可以通过 INTLEVEL （第 45 页）及 POSINT （第 54 页）位编程设置。

清除： CPU 清除该位禁止 INT 输出引脚。IE=0 时 INT#引脚的状态处于无效状态（电平模式时开路，负沿时为高，正沿时为低）。

上电复位: IE=0
芯片复位： IE=0
总线复位： *不变*
掉电: 只读

IN0BAVIE

含义： 端点 0 IN 缓冲器可用中断使能

位置： EPIEN.0

置位： CPU 将该位置位以使能 IN0BAV 中断请求（第 40 页）

清除： CPU 清除该位以禁止 IN0BAV 中断请求。

上电复位: IN0BAVIE=0
芯片复位： IN0BAVIE=0
总线复位： IN0BAVIE=0
掉电: 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

IN0BAVIRQ

含义： 端点 0 IN 缓冲器可用中断请求

位置： EPIRQ.0

置位： SIE 接收到指向端点 0 的 IN 令牌、发送 EP0FIFO （第 10 页）中的数据并收到来自主机的 ACK 握手后将该位置位。这表明 EP0FIFO 可以再次被 CPU 装载数据。

清除： CPU 通过向该位写“1”，或写字节数寄存器 EP0BC （第 9 页）的方法来复位该位。

上电复位: IN0BAVIRQ=1
芯片复位： IN0BAVIRQ=1
总线复位： IN0BAVIRQ=1
掉电: 只读

IN2BAVIE

含义： 端点 2 IN 缓冲器可用中断使能

位置： EPIEN.3

置位： CPU 将该位置位以使能 IN2BAV 中断请求（第 42 页）

清除： CPU 清除该位以禁止 IN2BAV 中断请求。

上电复位: IN2BAVIE=0
芯片复位： IN2BAVIE=0
总线复位： IN2BAVIE=0
掉电: 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

IN2BAVIRQ

含义： 端点 2 IN 缓冲器可用中断请求

位置： EPIRQ.3

置位： SIE 接收到指向端点 0 的 IN 令牌、发送 EP2INFIFO （第 10 页）中的数据并收到来自主机的 ACK 握手后将该位置位。这表明 EP2INFIFO 可以再次被 CPU 装载数据。

清除： CPU 通过写字节数寄存器 EP2INBC （第 17 页）的方法来复位该位。

上电复位： IN2BAVIRQ=1
芯片复位： IN2BAVIRQ=1
总线复位： IN2BAVIRQ=1
掉电： 只读

编程要点：

EP2IN 是双缓冲端点，这意味着有两个 FIFO 和字节数寄存器。当 CPU 向一个 IN FIFO 压入数据时，双缓冲允许 USB 数据同时从另外一个 IN FIFO 弹出。这样改善了系统的带宽。

IN2BAVIRQ 逻辑使双缓冲对于编程者透明。例如，假定两个缓冲均可用，则 IN2BAVIRQ =1。CPU 把 N 字节数据写入 EP2INFIFO，然后向 EP2INBC 寄存器写入 N 以配置传输。对于单缓冲的 IN 端点来说，在主机发送 IN 令牌给端点并接受 IN 数据包之前不会发生任何操作。但是对于双缓冲，由于有第二个缓冲器供 CPU 装载数据，IN2BAVIRQ 位在无效后可立即变为有效，就好象第一个包已经被发送并被主机接收。

双缓冲也意味着在 MAX3420E 复位之后， IN2BAVIRQ 位保持为 1，直到两次加载 EP2INBC 寄存器。

IN3BAVIE

含义： 端点 3 IN 缓冲器可用中断使能

位置： EPIEN.4

置位： CPU 将该位置位以使能 IN3BAV 中断请求（第 44 页）。

清除： CPU 清除该位以禁止 IN3BAV 中断请求。

上电复位： IN3BAVIE=0
芯片复位： IN3BAVIE=0
总线复位： IN3BAVIE=0
掉电： 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

IN3BAVIRQ

含义： EP3-IN 缓冲器可用中断

位置： EPIRQ.4

置位： SIE 在接收到指向端点 3 的 IN 令牌、发送 EP3INFIFO （第 22 页）中的数据并收到来自主机的 ACK 握手后将该位置位。这表明 EP3INFIFO 可以再次被 CPU 装载数据。

清除： CPU 通过写字节数寄存器 EP3INBC （第 21 页）清除该位。

上电复位： IN3BAVIRQ=1
芯片复位： IN3BAVIRQ=1
总线复位： IN3BAVIRQ=1
掉电： 只读

编程要点：

上电或复位时，IN FIFO 可用于接收 CPU 数据，所以这个位为 1。IN FIFO 缓冲器有效位是唯一默认为 1 的位。

INTLEVEL

含义： INT 输出引脚是电平有效的。

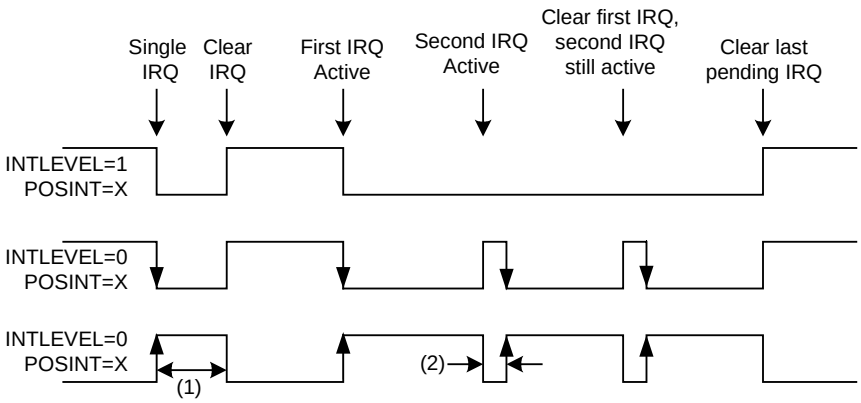
位置： PINCTL.3

置位： CPU 将该位置位，使 INT 输出电平信号。INTLEVEL=1 时输出引脚为低电平有效，漏极开路。INTLEVEL=1 时，系统必须接上拉电阻到 VL。

清除： CPU 清除该位以使 INT 引脚为边沿有效。当 INTLEVEL=0 时，POSINT 位（第 54 页）设置边沿极性。

上电复位: INTLEVEL=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：



(1) Width determined by clearing the IRQ. (2) Fixed at ~20 usec.

图 8. INT 引脚动作取决于 INTLEVEL 及 POSINT 位

图 8 所示的波形为不同 INTLEVEL 和 POSINT（第 54 页）位设置下的 INT 引脚的动作。在电平模式下（INTLEVEL=1），INT 引脚在有中断发生时保持低电平。

NOVBUSIE

含义： 无 V_{BUS} 中断使能。

位置： USBIEN.5

置位： 使能 NOVBUS 中断请求（第 47 页）。

清除： 禁止 NOVBUS 中断请求。

上电复位： NOVBUSIE=0
芯片复位： NOVBUSIE=0
总线复位： NOVBUSIE=0
掉电： 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

NOVBUSIRQ

含义： 无 V_{BUS} 中断请求。

位置： USBIRQ.5

置位： 当 V_{BUS} 引脚电平跌至低于 V_{BUS} 检测门限时，SIE 将该位置位。

清除： CPU 写“1”清除该位。

上电复位： NOVBUSIRQ=0
芯片复位： NOVBUSIRQ=0
总线复位： NOVBUSIRQ=0
掉电： 只读

编程要点：

对于自供电器件，通过 IRQ 位可以很容易检测到基于 MAX3420E 的 USB 外设是否同 USB 主机断开。

OSCOKIE

含义： 振荡器正常中断使能

位置： USBIEN.0

置位： CPU 将该位置位以使能 OSCOK 中断请求（第 49 页）。

清除： CPU 清除该位以禁止 OSCOK 中断请求。

上电复位： OSCOKIE=0
芯片复位： OSCOKIE=0
总线复位： OSCOKIE=0
掉电： 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

OSCOKIRQ

含义： 振荡器正常中断请求

位置： USBIRQ.0

置位： 内部 OSCOK 位表明内部 12MHz 振荡器稳定，芯片可正常工作。当 OSCOK 信号由 0 变 1 时，SIE 将 OSCOKIRQ 位置位，表明芯片准备好投入工作。

清除： CPU 写“1”以清除该位。

上电复位: OSCOKIRQ=0
芯片复位： OSCOKIRQ=0
总线复位： OSCOKIRQ=0
掉电: 只读

OUT0DAVIE

含义： 端点 0 OUT 数据可用中断使能

位置： EPIEN.1

置位： CPU 将该位置位以使能 OUT0DAV 中断请求（第 53 页）。

清除： CPU 将该位清除以禁止 OUT0DAV 中断请求。

上电复位: OUT0DAVIE=0
芯片复位： OUT0DAVIE=0
总线复位： OUT0DAVIE=0
掉电: 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

OUT0DAVIRQ

含义： 端点 0 OUT 数据可用中断请求

位置： EPIRQ.1

置位： 当 EP0 成功接收到（并以 ACK 应答）了一个 OUT 数据包时，SIE 将该位置位。

清除： CPU 写“1”将该位清除，为另一次传输配置端点。

上电复位: OUT0DAVIRQ=0
芯片复位： OUT0DAVIRQ=0
总线复位： OUT0DAVIRQ=0
掉电: 只读

OUT1DAVIE

含义： 端点 1 OUT 数据可用中断使能

位置： EPIEN.2

置位： CPU 将该位置位以使能 OUT1DAV 中断请求（第 51 页）。

清除： CPU 将该位清除以禁止 OUT1DAV 中断请求。

上电复位: OUT1DAVIE=0
芯片复位： OUT1DAVIE=0
总线复位： OUT1DAVIE=0
掉电: 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

OUT1DAVIRQ

含义： 端点 1 OUT 数据可用中断使能

位置： EPIRQ.2

置位： 当 EP1 成功接收到（并以 ACK 应答）了一个 OUT 数据包时，SIE 将该位置位。

清除： CPU 写“1”将该位清除，为另一次传输配置端点。

上电复位： OUT1DAVIRQ=0
芯片复位： OUT1DAVIRQ=0
总线复位： OUT1DAVIRQ=0
掉电： 只读

编程要点：

EP1OUT 是双缓冲端点，这意味着有两个 FIFO 和字节数寄存器。当 CPU 从一个 FIFO 读出数据时，允许 USB 数据同时压入另一个 FIFO。这样改善了系统的带宽。

OUT1DAVIRQ 逻辑使双缓冲对于编程者透明。例如，假定两个缓冲都是可用的，则 OUT1DAVIRQ=0。当 OUT 包到达时，OUT1DAVIRQ 由 0 变 1，表明第一个包可用。对于单缓冲的端点来说，在 CPU 释放 FIFO 以前，如果有另一个包通过 EP1-OUT 到达，SIE 将发送 NAK 握手信号，表明端点无法接收数据。

然而，对于双缓冲端点，由于第二个缓冲可以接收数据，因此可以接收并发 ACK 应答第二个 OUT 包。 如果在两个 FIFO 释放以前，接收到第三个 OUT 包，那么 SIE 发送 NAK 信号拒绝传输，表明两个 FIFO 都是满的。当 CPU 读完第一个 FIFO 并且清除了 OUT1DAVIRQ 位时（ 通过向该位写 1），它会立即由 0 变 1，表明第 2 个 FIFO 中的数据可用。

POSINT

含义： 设置 INT 输出引脚（如果设置为边沿输出）为上升沿有效。只有当 INTLEVEL=0 （第 45 页）时该位有效。

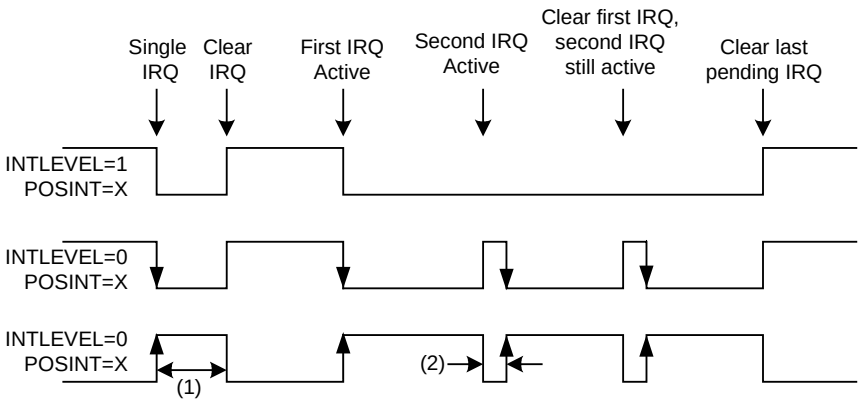
位置： PINCTL.2

置位： CPU 将该位置位，使 INT 引脚在有中断请求时由 0 变 1。

清除： CPU 将该位清除，使 INT 引脚在有中断请求时由 1 变 0。

上电复位: POSINT=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：



(1) Width determined by clearing the IRQ. (2) Fixed at ~20 usec.

图 9. INT 引脚动作取决于 INTLEVEL 和 POSINT 位

如果 INTLEVEL=1 （第 45 页），设置该位不起作用。

每当有新的中断产生，或者当 SPI 主控制器清除了一个中断，而同时还有其他已登记的中断时，边沿敏感模式输出一个跳变沿。如图 9 所示，有效边沿脉冲的宽度各不相同。如果如图所示的第一个脉冲，只有一个中断被激活，则脉冲宽度取决于 SPI 主控制器需要多长时间清除中断位。如果产生新中断时还有其他已登记的中断，或者当 SPI 主控制器清除了一个中断时尚有其他已登记的中断，脉宽固定在大约 20 微秒。

PWRDOWN

含义： 使 MAX3420E 掉电。

位置： USBCTL.4

置位： CPU 将该位置位以使芯片进入低功耗状态，满足处于挂起状态的 USB 外设的低功耗要求。

清除： CPU 清除该位以使芯片脱离低功耗状态并恢复工作。

上电复位: PWRDOWN=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：

PWRDOWN 位由 0 到 1 的跳变触发掉电工作模式。因此，如果芯片被唤醒时 PWRDOWN=1（ 各种唤醒方法参见下文），器件不会立即再次掉电。*任何唤醒例程都应清除 PWRDOWN 位，以便为下次 0-1 跳变做好准备。*

CPU 通常设置 PWRDOWN=1 和 HOSCSTEN=1（第 37 页）来使 MAX3420E 进入掉电模式。当 CPU 设置 PWRDOWN=1 时，SIE 发生以下动作：

- 停止内部的 12MHz 振荡器。
- 监控 USB DPLUS 引脚，以检测总线活动。
- 监控 SPI 端口，以访问有限的一组寄存器（如 USBCTL）。

当处于低功耗状态时 （PWRDOWN=1），可以通过以下两种方式唤醒芯片：

1. *CPU 清除 PWRDOWN 位。*

这样会启动内部振荡器，一旦振荡器稳定，SIE 激活 OSCOK IRQ（第 49 页）。

2. *SIE 检测 DPLUS 是否激活，以及 HOSCSTEN 位是否为 1。*

自供电器件设置 HOSCSTEN=0 时的特例参见第 37 页的说明。

RWUDNIE

含义： 远端唤醒信号完成中断使能

位置： USBIEN.1

置位： CPU 将该位置位以使能 RWUDN 中断请求（第 57 页）。

清除： CPU 清除该位以禁止 RWUDN 中断请求。

上电复位： RWUDNIE=0
芯片复位： RWUDNIE=0
总线复位： RWUDNIE=0
掉电： 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

RWUDNIRQ

含义： 远端唤醒信号完成中断请求

位置： USBIRQ.1

置位： 在 RWU 信号（10ms 的 K 状态）结束后 SIE 将该位置位。

清除： CPU 写“1”清除该位。

上电复位: RWUDNIRQ=0
芯片复位： RWUDNIRQ=0
总线复位： RWUDNIRQ =0
掉电: 只读

编程要点：

CPU 设置位 SIGRWU=1 （第 58 页），发出远端唤醒信号。

SIGRWU

含义： 发出远端唤醒信号

位置： USBCTL.2

置位： CPU 将该位置位，向 USB 主机发出远端唤醒信号。

清除： CPU 清除该位以结束远端唤醒信号。

上电复位: SIGRWU=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：

SPI 主控制器将该位置位以向总线发出 USB 远端唤醒信号。当 SPI 主控制器设置 SIGRWU=1 时，MAX3420E 在等待 5ms 的 J 状态后，驱动总线进入 K 状态（D+为低，D-为高），并保持 10ms。10ms 之后，MAX3420E 停止驱动总线，并触发 RWUDNIRQ（第 57 页）中断位。只有当总线处于挂起状态时，SPI 主控制器才可设置 SIGRWU=1。

MAX3420E 处于掉电状态（PWRDOWN= 1，第 55 页）时， SPI 主控制器写 USBCTL 寄存器设置 SIGRWU=1 的同时，必须清除 PWRDOWN 位，重新启动内部振荡器。这样，在 SIE 重新启动振荡器并等待其稳定后，便可发出 RWU 信号。

SPI 主控制器设置 SIGRWU=1 并等待 SIE 发出 RWUDNIRQ（第 57 页）以表明唤醒信号结束。当产生 RWUDNIRQ 后，SPI 主控制器应设置 SIGRWU=0 以关闭信号。如果 SPI 主控制器在 10ms 的信号间隔内设置 SIGRWU=0，RWU 信号照样终止于 10ms。如果发送 RWUDNIRQ 中断信号时 SPI 主控制器未清除 SIGRWU 位，SIE 将持续以 5ms 悬空（J），10ms K 驱动总线。

FYI: 只有满足以下两个前提，USB 外设才发出远端唤醒信号：

1. 在枚举期间设备固件通知主机该设备可以发送远端唤醒信号。具体是，配置描述符的 bmAttributes 域的 bit 5 表明配置是否支持远端唤醒：1=支持，0=不支持。
2. 主机发出 Set_Feature 请求，并将特性选择域设置为“Device_Remote_Wakeup”。如果第一个条件不成立，即不支持远端唤醒，主机永远不会发送这个请求。

主机随后可以发出 Clear_Feature 请求并将选择域设置为“Device_Remote_Wakeup”以禁止远端唤醒信号。

STLEP0IN

含义： 返回 STALL 握手以响应对端点 0 的 IN 请求。

位置： EPSTALLS.0

置位： CPU 将该位置位以通知 SIE 返回 STALL 握手，作为对端点 0 的 IN 请求的响应。

清除： SETUP 令牌一到达 SIE 即清除该位。

上电复位： STLEP0IN=0
芯片复位： STLEP0IN=0
总线复位： STLEP0IN=0
掉电： 只读

编程要点：

CPU 发送 STALL 握手信号，表明端点 0 的请求非法或未知。

Endpoint 0 有 3 个 stall 位用于状态阶段，以及传输中可能会用的可选的 IN 及 OUT 数据阶段：

- STLSTAT （第 64 页）
- STLEP0IN
- STLEP0OUT （第 60 页）

如果需要停止一次 CONTROL 传输，两种数据阶段（如果有）和状态阶段都应收到 STALL 握手。由于下一个 SETUP 令牌到来时 3 个停止位均被清除，因此停止 CONTROL 传输的最好方法是将三个端点 0 停止位置位。这将完全中止主机发送的所有阶段的 CONTROL 传输。

STLEP0OUT

含义： 返回 STALL 握手以响应对端点 0 的 OUT 请求。

位置： EPSTALLS.1

置位： CPU 将该位置位以通知 SIE 返回 STALL 握手，作为对端点 0 的 OUT 请求的响应。

清除： SETUP 令牌一到达 SIE 即清除该位。

上电复位： STLEP0OUT=0
芯片复位： STLEP0OUT=0
总线复位： STLEP0OUT=0
掉电： 只读

编程要点：

CPU 发送 STALL 握手信号，表明对端点 0 的请求不合法或者未知。

Endpoint 0 有 3 个 stall 位用于状态阶段，以及传输中可能用到的 IN 及 OUT 数据阶段：

- STLSTAT （第 64 页）
- STLEP0IN （第 59 页）
- STLEP0OUT

如果需要停止一次 CONTROL 传输，数据阶段（如果有）和状态阶段都应接收到 STALL 握手信号。由于下一个 SETUP 令牌到来时 3 个停止位均被清除，因此停止 CONTROL 传输的最好的方法是将三个端点 0 停止位置位。这将完全中止主机发送的所有阶段的 CONTROL 传输。

STLEP1OUT

含义： 停止 EP1-OUT。返回 STALL 握手以响应对端点 1 的 OUT 请求。

位置： EPSTALLS.2

置位： CPU 将该位置位以通知 SIE 返回 STALL 握手，作为对端点 1 的 OUT 请求的响应。

清除： CPU 清除该位，使 EP1-OUT 返回正常的 ACK-NAK 模式。

上电复位: STLEP1OUT=0
芯片复位： STLEP1OUT=0
总线复位： STLEP1OUT=0
掉电: 只读

编程要点：

当接收到指向 EP1-OUT 的 Set_Feature(HALT) 命令时，CPU 将该位置位。
当接收到指向 EP1-OUT 的 Clear_Feature(HALT) 命令时，CPU 清除该位。

STLEP2IN

含义： 停止 EP2-IN。返回 STALL 握手以响应对端点 2 的 IN 请求。

位置： EPSTALLS.3

置位： CPU 将该位置位以通知 SIE 返回 STALL 握手，响应对端点 2 的 IN 请求。

清除： CPU 清除该位，使 EP2-IN 返回正常的 ACK-NAK 模式。

上电复位： STLEP2IN=0
芯片复位： STLEP2IN=0
总线复位： STLEP2IN=0
掉电: 只读

编程要点：

当接收到指向 EP2-IN 的 Set_Feature(HALT) 时，CPU 将该位置位。
当接收到指向 EP2-IN 的 Clear_Feature(HALT) 时，CPU 清除该位。

STLEP3IN

含义： 返回 STALL 握手以响应对端点 3 的 IN 请求。

位置： EPSTALLS.4

置位： CPU 将该位置位以通知 SIE 返回 STALL 握手，响应端点 3 的 IN 请求。

清除： 当接收到指向 EP3-IN 的 Clear_Feature(Halt) 请求时，CPU 清除该位。

上电复位： STLEP3IN=0
芯片复位： STLEP3IN=0
总线复位： STLEP3IN=0
掉电： 只读

编程要点：

当接收到指向 EP3-IN 的 Set_Feature(HALT) 时，CPU 将该位置位。
当接收到指向 EP3-IN 的 Clear_Feature(HALT) 时，CPU 清除该位。

STLSTAT

含义： 返回 STALL 握手以响应 CONTROL 传输的 STATUS 阶段。

位置： EPSTALLS.5

置位： CPU 将该位置位以发送 STALL 握手，响应 CONTROL 传输的 status 阶段。在 CPU 应答（ACKSTAT 位=1）或停止（STLSTAT=1）传输之前，SIE 用 NAK 握手响应 CONTROL 传输的 status 阶段。

清除： SETUP 令牌一到达 SIE 即清除该位。

上电复位： STLSTAT=0
芯片复位： STLSTAT=0
总线复位： STLSTAT=0
掉电： 只读

编程要点：

Endpoint 0 有 3 个 stall 位用于状态阶段，以及传输中可能用到的 IN 及 OUT 数据阶段：

- STLSTAT
- STLEP0IN （第 59 页）
- STLEP0OUT （第 60 页）

如果需要停止一次 CONTROL 传输，数据阶段（如果有）和状态阶段都应接收到 STALL 握手信号。由于下一个 SETUP 令牌到来时 3 个停止位均被清除，因此停止 CONTROL 传输的最好的方法是将三个端点 0 停止位置位。这将完全中止主机发送的所有阶段的 CONTROL 传输。

SUDAVIE

含义： SETUP 数据可用中断使能

位置： EPIEN.5

置位： CPU 将该位置位以使能 SUDAV 中断请求（第 66 页）。

清除： CPU 清除该位以禁止 SUDAV 中断请求。

上电复位： SUDAVIE=0
芯片复位： SUDAVIE=0
总线复位： SUDAVIE=0
掉电： 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

SUDAVIRQ

含义： SETUP 数据可用中断请求

位置： EPIRQ.5

置位： 如果在 CONTROL 传输中正确接收到 8 个 SETUP 数据字节，SIE 将该位置位。

清除： CPU 写“1”清除该位。

上电复位: SUDAVIRQ=0
芯片复位： SUDAVIRQ=0
总线复位： SUDAVIRQ=0
掉电: 只读

编程要点：

该中断信号通知 CPU 从 SUDFIFO（第 67 页）读 8 个 SETUP 数据字节，并解释主机请求。

SUDFIFO

含义： 设置数据 FIFO

位置： SUDFIFO[7:0]

写： 当 SIE 从主机接收 CONTROL 传输时，会将 8 个 SETUP 数据字节复制进该 FIFO。如果数据证实准确无误，SIE 发送 SETUP 数据可用中断请求（第 66 页）。

读： CPU 对该寄存器进行 8 次读操作，以取回 8 个 SETUP 字节。

上电复位： SUDFIFO=0
芯片复位： SUDFIFO=0
总线复位： SUDFIFO=0
掉电： 不能读、写

编程要点：

MAX3420E 提供这个单独的 FIFO 是为了避免混淆 EP0FIFO（第 10 页）里的数据和命令字节。这样使编程者无须考虑 FIFO 中的数据类型，EP0FIFO（第 10 页）内总是数据，SUDFIFO 内总是命令。

SUSPIE

含义： SUSPEND 中断使能

位置： USBIEN.4

置位： CPU 将该位置位以使能 SUSPEND 中断请求（第 69 页）。

清除： CPU 清除该位以禁止 SUSPEND 中断请求。

上电复位: SUSPIE=0
芯片复位： SUSPIE=0
总线复位： SUSPIE=0
掉电: 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

SUSPIRQ

含义： SUSPEND 中断请求

位置： USBIRQ.4

置位： 当检测到 USB 挂起（总线 3ms 无流量）时，SIE 将该位置位。

清除： CPU 写“1”清除该位。

上电复位: SUSPIRQ=0
芯片复位： SUSPIRQ=0
总线复位： SUSPIRQ=0
掉电: 只读

编程要点：

CPU 通过关闭系统外设，并把 MAX3420E 置于低功耗状态（PWRDOWN=1，第 55 页）来响应该中断。

URESDNIE

含义： USB 总线复位完成中断使能

位置： USBIEN.7

置位： CPU 将该位置位以使能 URESDN IRQ（第 68 页）。

清除： CPU 清除该位以禁止 URESDN IRQ。

上电复位: URESDNIE=0
芯片复位： URESDNIE=0
总线复位： *不变*
掉电: 只读

编程要点：

这是 USB 总线复位期间不被清除的两个位之中的一个。另一个是 URESIE（第 72 页）。在总线复位期间，它们常被 CPU 使用。

URESDNIRQ

含义： USB 总线复位完成中断请求

位置： USBIRQ.7

置位： 当检测到 USB 总线复位结束时，SIE 将该位置位。

清除： CPU 写“1”清除该位。

上电复位: URESDNIRQ=0
芯片复位： URESDNIRQ=0
总线复位： *不变*
掉电: 只读

URESIE

含义： USB 复位中断使能

位置： USBIEN.3

置位： CPU 将该位置位以使能 USB 复位中断请求（第 73 页）。

清除： CPU 清除该位以禁止 USB 复位中断请求。

上电复位: URESIE=0
芯片复位： URESIE=0
总线复位： *不变*
掉电: 只读

编程要点：

这是在 USB 总线复位期间不被清除的两个位之中的一个。另一个是 URESDNIE（第 70 页）。在总线复位期间，它们常被 CPU 使用。

URESIRQ

含义： USB 总线复位中断请求

位置： USBIRQ.3

置位： 当检测到 USB 总线复位（至少 2.5μs 的 SE0 总线状态）时，SIE 将该位置位。

清除： CPU 写“1”清除该位。

上电复位: URESIRQ=0
芯片复位： URESIRQ=0
总线复位： URESIRQ=1
掉电: 只读

VBUSIE

含义： V_{BUS}检测中断使能

位置： USBIEN.6

置位： CPU 将该位置位以使能 VBUS 中断请求（第 74 页）。

清除： CPU 清除该位以禁止 VBUS 中断请求。

上电复位: VBUSIE=0
芯片复位： VBUSIE=0
总线复位： VBUSIE=0
掉电: 只读

编程要点：

由于 USB 复位期间会清除大多数中断使能位，因此在 USB 总线复位时，需要进行初始化，以置位中断使能位。

VBUSIRQ

含义： V_{BUS}存在中断请求

位置： USBIRQ.6

置位： 当 V_{BUS}比较器由 0 变 1 时 SIE 将该位置位。

清除： CPU 写“1”清除该位。

上电复位: VBUSIRQ=0
芯片复位： VBUSIRQ=0
总线复位： VBUSIRQ=0
掉电: 只读

编程要点：

对于基于 MAX3420E 的自供电外设，该中断位提供了一个检测外设是否连接到 USB 主机的简便手段。

VBGATE

含义： V_{BUS} 门控

位置： USBCTL.6

置位： CPU 将该位置位，以使 CONNECT 位的工作以 V_{BUS}s 引脚上存在 V_{BUS} 为条件。

清除： CPU 清除该位使 CONNECT 的工作与 V_{BUS} 是否有效无关。

上电复位： VBGATE=0
芯片复位： *不变*
总线复位： *不变*
掉电: 读-写

编程要点：

USB 标准规定：没有 V_{BUS} 的情况下全速 USB 器件不能为其 DPLUS 上拉电阻供电。这一规定主要针对那些自供电设备，这类设备具有自己的电源并且不通过引脚 V_{BUS} 直接为上拉电阻供电。只有当主机打开 V_{BUS} 以激活下游端口时，才允许外设为其上拉电阻供电来指示其已接入总线了。

利用 VBGATE 位可自动满足该项要求，这是通过根据 V_{BUS} 的状态对 CONNECT 位的工作加以门控实现的，如下表所示：

CONNECT	VBGATE	VBUS_DET	上拉
0	X	X	不连接
1	0	X	连接
1	1	0	不连接
1	1	1	连接

如果 VBGATE=1 且 CONNECT=1，则在 V_{BUS} 检测器指示 V_{BUS} 电压有效之前不会连接上拉电阻。

索引