

1 AS5040 扩展的 OTP 编程功能

本应用笔记介绍了AS5040的各种编程方法，并提供了C样例代码。本文不仅阐述了OTP寄存器的通用永久性编程方法，同时介绍了如何对OTP进行一次或多次非永久性“软写入”。

欲了解AS5040的更多详细信息，请参考AS5040数据资料。

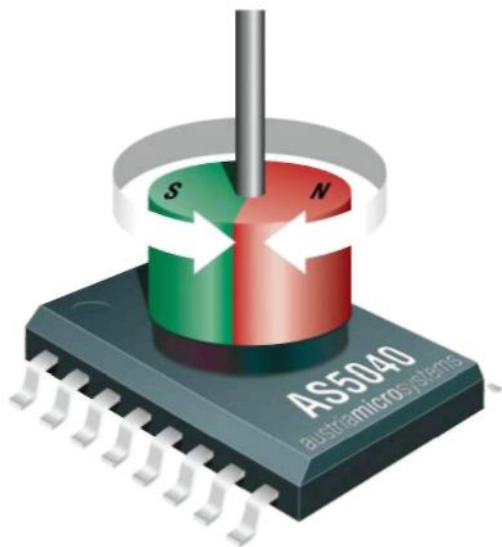


图1: AS5040和磁铁的典型配置

2 对 AS5040 编程

2.1 硬件接线

要进行OTP编程，需要3个信号：Prog、CSn和CLK（见图3）。要读取角位以进行零位编程时，还需要DO信号。AS5040能在3.3V或5V两种模式下进行编程。图2显示了在3.3V模式下，如何正确连接AS5040演示板。

上电后，在Prog = 高且CLK = 低的情况下，CSn的上升沿将启用AS5040编程操作。16位配置数据必须通过Prog引脚串行移入OTP寄存器。第1位(CCW)后面跟随零位数据（MSB在前）和增量模式设置，具体设置见表1。数据在CLK的上升沿必须是有效的(见图3)。

当数据写入OTP寄存器后，通过将引脚Prog的电压提高到编程电压V_{PROG}，可实现永久性编程。必须提供16个CLK脉冲（I_{PROG}）来对熔丝编程（见图4）。若要退出编程模式，必须通过一个上电复位过程来复位芯片。下一次上电后，已编程的数据将投入使用。

注意：在编程过程中，编程电流的转变会由于连接电缆存在电感而产生高压尖峰脉冲。为了避免这些尖峰脉冲以及其可能对IC造成的损害，连接电缆必须尽量短，特别是信号Prog和VSS。V_{PROG}开关晶体管与引脚Prog之间的最大线长（见图2）不应超过50mm（2英寸）。为了抑制可能产生的电压尖峰脉冲，应当在靠近引脚Prog和VSS的位置安装一个10nF陶瓷电容。只有在编程操作时需要此电容，正常工作情况下并不需要。

时钟时序 t_{clk} 必须选择适当的速率，以确保信号Prog在CLK的上升沿可保持稳定（见图3）。此外，编程电压应当采用1个10μF电容进行缓冲，并且安装位置应该靠近开关晶体管。此电容在编程过程中有助于提供峰值电流。引脚Prog处规定的编程电压为7.3–7.5V。为了补偿V_{PROG}开关晶体管两端的压降，所施加的编程电压可以略微高一些（7.5–8.0V，见图2）。

2.1.1 AS5040 OTP 寄存器的内容

CCW 逆时针位

ccw = 0 – 角度值沿顺时针方向增加

ccw = 1 – 角度值沿逆时针方向增加

Z [9:0] 可编程零位/Index位置

Indx Index脉冲宽度选择：1LSB / 3LSB

Div1, Div0 增量输出的分频器设置

Md1, Md0 增量输出模式选择

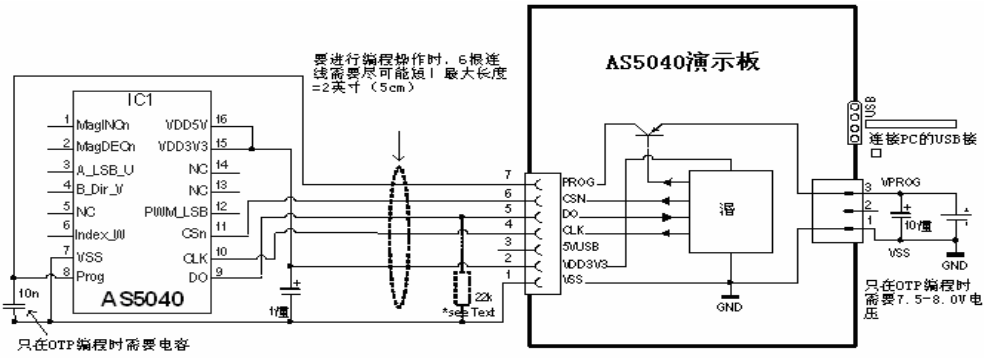


图2: AS5040的OTP编程连接方式（包括AS5040演示板）

*) 如果使用较长的电缆，推荐在DO线上接一个22k-56kΩ的下拉电阻，以释放数据传输间的信号，且不会产生静态电压。

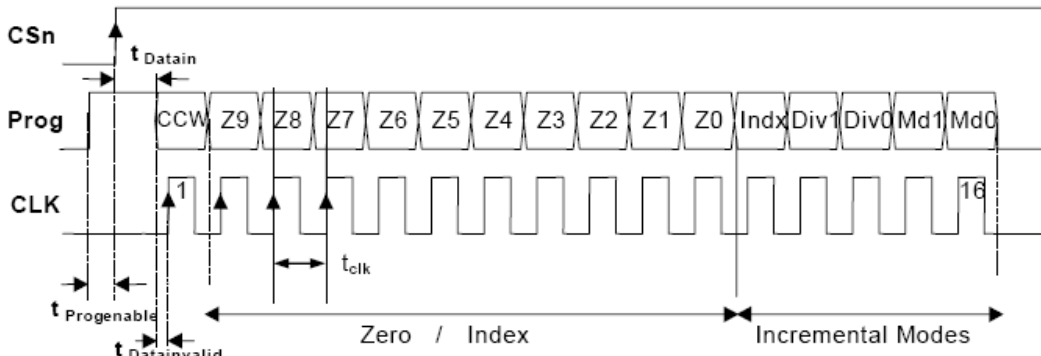
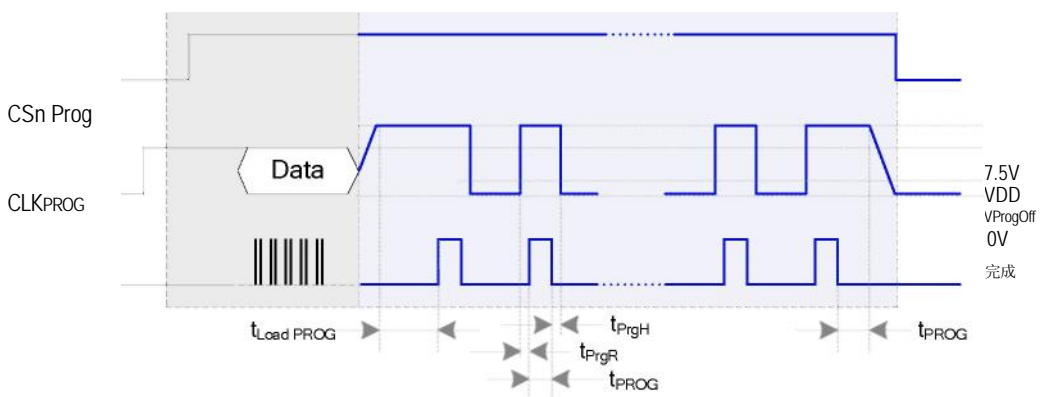


图3: 编程访问-写入数据（图4的一部分）



2.2 增量模式编程

提供三种不同的增量输出模式。

模式: Md1=0 / Md0=1将AS5040设置为正交模式。

模式: Md1=1 / Md0=0将AS5040设置为步/方向模式。

在这两种模式下, 均可以使用OTP分频器位Div1和Div0将增量分辨率从10位降至9、8或7位 (参见下面的表1))。

模式: Md1=1 / Md0=1将AS5040设置为无刷直流电动机换向模式, 引脚12 (PWM_LSB) 提供额外的LSB增量信号。

为了能够编程所有位, 缺省工厂设置的所有位 = 0。这种模式对应模式1:0 (正交A/B, 1LSB的index宽度, 256ppr)。

缺省情况下, 磁铁沿顺时针方向 (俯视) 旋转时, 绝

对角度输出值增大。设置CCW位 (见图3) 可以取反指定的方向, 例如磁铁安装在IC的下方时:

CCW = 0 - 角度值沿顺时针方向增加;
 CCW = 1 - 角度值沿逆时针方向增加。

缺省情况下, 零位/index位置脉冲的宽度为1个LSB。通过设置OTP寄存器的Index位, 脉冲宽度可增加到3个LSB。

还可为无刷直流电动机控制提供其它编程选项 (换向模式)。

Md1 = Md0 = 1可将增量输出引脚3、4和6设置为3相换向信号。Div1为双磁极 (Div1=0) 或四磁极 (Div1=1) 定子定义每圈的脉冲数。

此外, 引脚12提供LSB信号 (LSB信号取代PWM信号), 从而可实现高达10,000rpm的高速测量。

模式	OTP-模式-寄存器-位					引脚号				脉冲/转 ppr	分辨率 位
	Md1	Md0	Div1	Div0	Index	3	4	6	12		
缺省 (Mode0.0)	0	1	0*	0*	0*	A	B	1LSB	PWM 10位	2x256	10
quadAB-Mode1.0	0	1	0	0	0			1LSB			
quadAB-Mode1.1	0	1	0	0	1			3LSBs			
quadAB-Mode1.2	0	1	0	1	0			1LSB		2x128	9
quadAB-Mode1.3	0	1	0	1	1			3LSBs			
quadAB-Mode1.4	0	1	1	0	0			1LSB			
quadAB-Mode1.5	0	1	1	0	1			3LSBs		2x64	8
quadAB-Mode1.6	0	1	1	1	0			1LSB			
quadAB-Mode1.7	0	1	1	1	1			3LSBs			
Step/Dir-Mode2.0	1	0	0	0	0	LSB	Dir	1LSB	PWM 10位	512	10
Step/Dir-Mode2.1	1	0	0	0	1			3LSBs			
Step/Dir -Mode2.2	1	0	0	1	0			1LSB		256	9
Step/Dir -Mode2.3	1	0	0	1	1			3LSBs			
Step/Dir -Mode2.4	1	0	1	0	0			1LSB		128	8
Step/Dir -Mode2.5	1	0	1	0	1			3LSBs			
Step/Dir -Mode2.6	1	0	1	1	0			1LSB		64	7
Step/Dir -Mode2.7	1	0	1	1	1			3LSBs			
Commutation-Mode3.0	1	1	0	0	0	U(0°)	V(120°)	W(240°)	LSB	3 x 1	10
Commutation-Mode3.1	1	1	0	1	0						9
Commutation-Mode3.2	1	1	1	0	0	U' (0°, 180°)	V' (60°, 240°)	W' (120°, 300°)	LSB	2 x 3	10
Commutation-Mode3.3	1	1	1	1	0						9

表1: 一次性可编程 (OTP) 寄存器选项

*注意: Div1、Div0和Index在模式0:0下不能编程。

2.3 零位编程

零位编程是一种能够简化系统装配的OTP选择，磁铁不必手工调节至机械零位。一旦装配完成后，可以通过软件匹配机械和电气零位。整圈内的任何位置均可被永久性指定为新的零位/索引位置。

进行零位编程时，磁铁可以转至机械零位（例如，旋转开关的“关闭”位置），并读取实际的角度数值。

将此数值写入OTP寄存器的Z9:Z0位（参见图3），并如第2节所述进行编程设定。

这个新的绝对零位也是增量输出模式新的索引脉冲位置。

注意：零位数值也可以在编程前进行修改，例如，要与机械零位成180°角度（半圈）的位置设置为电气零位，只需在机械零位的读数上增加512，然后将得到的新数值编程至OTP寄存器。

2.4 模拟回读模式

非易失性编程（OTP）采用了芯片内置的齐纳二极管，这种二极管在承受规定的逆向电流后可永久性地保持在低阻状态。

编程过程成功与否取决于编程过程中所提供的电流大小（最高达130mA）。这一电流必须由外部电压源来提供。如果此电压源不能提供足够的功率，齐纳二极管有可能无法正确编程。为了验证编程位是否被正确编程，可以读取对应每一个齐纳二极管的模拟电平，以确定特定位是否被正确编程。

为了将AS5040置为模拟回读模式，必须如图5所示在引脚CSn、Prog和CLK上施加一个数字序列。该引脚的数字信号电平取决于供电配置（3.3V或5V）。

CSn（OutpEN）的第2个上升沿将使引脚Prog变为数字输出，而且必须撤消施加在引脚Prog上的逻辑高电平信号，以避免输出发生冲突（图5的灰色区域）。

随后CSn的下降沿将使引脚Prog改变成模拟输出，并提供一个基准电压 V_{ref} ，必须保存这个基准电压，以作为计算后续已编程和未编程OTP位的基准。完成这一步骤后，CLK的每一个上升沿将输出顺序与编程过程相反的第1位数据（参见图5: Md0-MD1-Div0, Div1-Indx-Z0-Z9, ccw）。

在模拟回读期间，应当移除引脚Prog处的电容（见图2）以实现较快的读出速率。

每一位测量的模拟电压必须减去前面所测得的 V_{ref} ，结果数值可给出被编程位的质量指示：<100mV的读数表明位编程正确，>1V的读数表明位编程不正确。

位于100mV至1V之间的读数表明该位存在错误，在上电读取OTP数据时，上述错误可能导致读取到未定义的数字值。

在第16个时钟（读取位“ccw”后）之后，必须通过断电来复位芯片。

注意：建议不要通过反复执行OTP编程操作来“清除”出错的熔丝。应该丢弃含有出错熔丝的芯片。

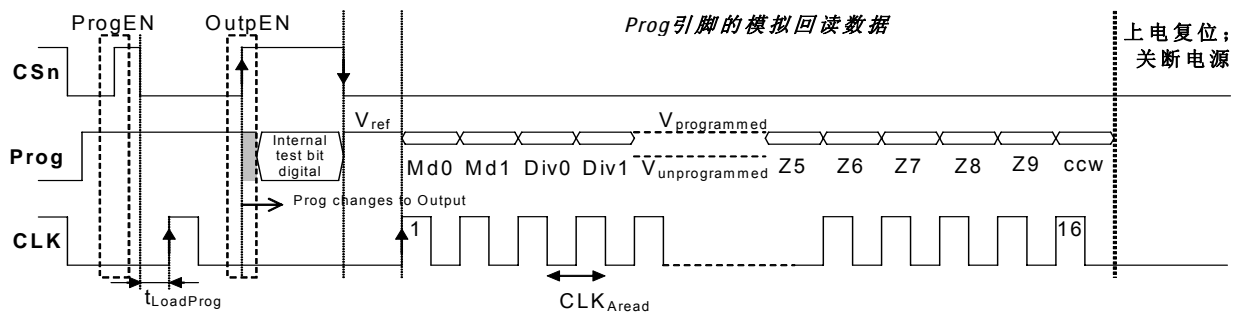


图 5: 模拟读取16位OTP寄存器的用户设置

2.5 同步串行接口 (SSI)

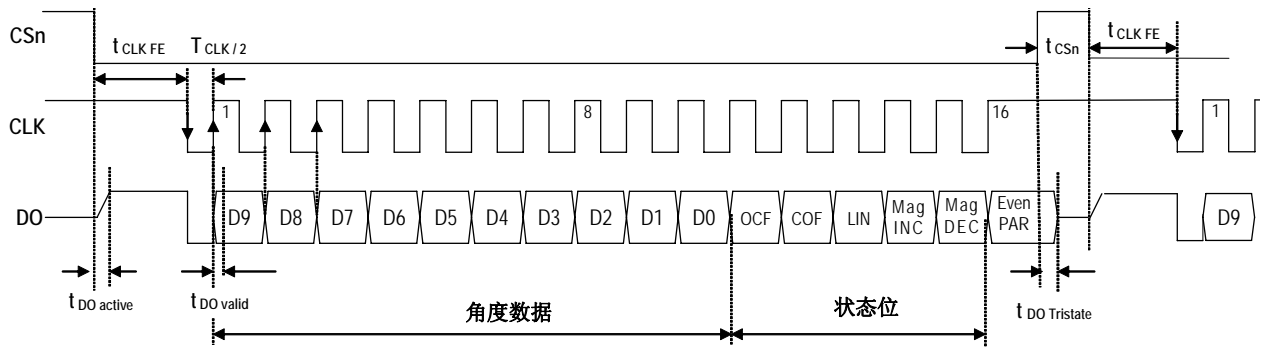


图6: 同步串行接口与绝对角度数据

如果CSn变为逻辑低，数据输出（DO）将从高阻（三态）变为逻辑高，并将启动读取操作。

- 经过最短时间 $t_{CLK\ FE}$ 之后，数据在CLK的第1个下降沿锁存至输出移位寄存器。
- 每个后续的CLK上升沿将移出1位数据。
- 串行字包含16位，前10位是角度信息D[9:0]，后6位包含系统信息，涉及数据的有效性，诸如OCF、COF、LIN、奇偶性和磁场状态（增强/减弱）等。
- 通过CSn处的逻辑“高”脉冲启动下一次测量，CSn的最小持续时间为 t_{CSn} 。

2.5.1 数据内容

D9:D0 绝对值角度数据（首先同步输出MSB）。

OCF（偏差补偿完成），逻辑高电平表示偏差补偿算法执行完毕。为了实现快速启动，可以通过外部微控制器查询此位。一旦此位置位，则表明AS5040已经完成启动，并且数据有效（参见表3）。

COF（Cordic溢出），逻辑高电平表示CORDIC单元出现了超范围的错误。此位置位时，D9:D0的数据无效。绝对值输出保留最后1个有效的角度值。

通过将磁铁位置调整至X-Y-Z容限以内，可以消除这种报警故障。

LIN（线性度报警），逻辑高电平表示输入磁场产生了严重的输出线性度问题。此位置位时，D9:D0的数据仍然可以使用，但可能包含无效数据。通过将磁铁位置调整至X-Y-Z容限以内，可以消除此项警告。

MagINC（磁场增强），磁铁被推向靠近IC的位置时，会导致磁场强度增大，该位将变成逻辑高。

MagDEC（磁场减弱），磁铁被拉至远离IC的位置时，会导致磁场强度减小，该位将变成逻辑高。

这两个信号均为逻辑高时，表明磁场强度超出了所允许的范围（参见表2）。

Mag INC	Mag DEC	说明
0	0	没有间距变化 磁场输入正常（在范围内）
0	1	间距增加，磁铁离开芯片：拉功能
1	0	间距缩短，磁铁靠近芯片：推功能
1	1	磁场输入无效 - 超出范围：过强、过弱（磁铁缺失）

表2: 磁场强度变化指示

注意：引脚1和引脚2（MagINCn, MagDECn）均为低电平有效的漏极开路输出，并要求外接上拉电阻。如果磁场强度处于范围之内，则两个输出均关闭。

这两个引脚也可以组合使用同一个上拉电阻。在这种情况下，只有当磁场强度处于正常范围时信号才会是高电平。所有其它情况下信号均为低电平（参见表2）。

偶数奇偶校验位用于第1至15位（D9-D0、OCF、COF、LIN、MagINCn和MagDECn）的传输错误检测。

绝对值角度输出总是设置为10位分辨率。将磁铁安装在芯片上方时，缺省情况下，角度数值沿顺时针方向增加。

当状态位具有下列配置时，数据D9:D0有效：

OCF	COF	LIN	Mag	Mag	奇偶校验
			INC	DEC	
1	0	0	0	0	1:15位的偶校验和
			0	1	
			1	0	

表3: 状态位输出

绝对值角度位置的采样速率为10kHz (0.1ms)。这样可以在0.1秒内读取360度范围的全部1024个位置= 9.76Hz (~10Hz)，而不会错失任何位置。将10Hz乘以60，结果得出对应的最大旋转速率为600 rpm。

每两个角度位置读取一次，可以达到最高1200 rpm的旋转速率。

因此，增加旋转速率会减少每圈读取的绝对角度位置的数量。无论所读取的旋转速率或位置数量是多少，绝对角度数值总是采用10位最高分辨率。

由于采用了插补器，增量输出不受旋转速率限制的影响。增量输出信号可以应用在旋转速率高达10000rpm的高速应用中，而不会丢失脉冲。

2.6 SSI 读取软件实现

```
void readSSI(BYTE* buffer, BYTE num_bits){
    BYTE current_byte = 0;
    BYTE current_bit = 0;
    BYTE temp = 0;

    CLEAR_CSN(); wait(10); // set CSN=0
    CLEAR_CLK(); wait(10);

    temp = 0;
    for(current_bit = num_bits; current_bit; current_bit--){
        SET_CLK();
        temp += (VAL_D0) ? 1 : 0;
        CLEAR_CLK();
        wait(1);
        if(((current_bit - 1) & 0x07) == 0) {
            buffer[current_bit >> 3] = temp;
            temp = 0;
        }
        temp <<= 1;
    }
    SET_CLK();
    SET_CSN();
}
```

3 OTP 寄存器的非永久性写入

在工作过程中，也能以一种非永久性的方式（软写入）写入OTP寄存器。当然，在芯片断电时这种非永久性设置即会丢失。上电后，芯片会根据OTP寄存器的设定值进行设置。这种设置能以两种方式进行临时性改写：

- 单次软写入，在AS5040上电后只进行一次OTP设置改写。
- 反复软写入，在AS5040上电后进行多次OTP设置改写。

3.1 单次 OTP 软写入

如果上电后AS5040的OTP寄存器只进行一次修改，可以采用如图7所示的简化软写入操作过程。除了CSn、Prog和CLK在第16个时钟之后都变为低并且不在引脚Prog上施加编程电压以外，这种写入过程与OTP编程过程的第一部分（图3）完全相同。这一过程完成后，可以紧接着使用CSn、CLK和DO信号进行常规的角度数据读取操作（图6）。

需要注意，这个操作在AS5040上电后只能执行一次。如果要重复这一操作以实现其它设置，必须首先通过供电循环进行一次上电复位。如果不断电就进行重复软写入操作，将会改写工厂设置并导致芯片停止工作。当然，这样的错误状态或不正确的软写入操作不会导致AS5040永久性损坏。要消除错误状态，只需给芯片断电并重启芯片即可。

3.2 OTP 写入软件实现

```

void writeOTP(BYTE* buffer, BYTE num_bits)
{
    BYTE *current_byte;
    BYTE current_bit = 0;
    BYTE temp = 0;

    current_byte = buffer + ((num_bits-1)>>3); //8 ;
    temp = *current_byte;
    if(num_bits % 8){
        temp <<= 8 - (num_bits % 8);
    }

    OUT_PROG_IN();
    CLEAR_PROG_IN();

    //-- Init OTP-Write
    CLEAR_CSN();
    wait(500);
    CLEAR_CLK();
    wait(500);

    //-- ProgEN
    SET_PROG_OUT();
    wait(500);

    SET_CSN();
    wait(500);

    //-- send OTP Data
    for(current_bit = num_bits; current_bit; current_bit--)
    {
        if(temp & 0x80)
            {SET_PROG_OUT();
            wait(600);
            }
        else
            {CLEAR_PROG_OUT();
            wait(600);
            }
    }

    SET_CLK();
    wait(300); // delay

    CLEAR_CLK();
    wait(50);

    temp <<= 1;
    if(((current_bit-1) & 0x07) == 0)
    {
        temp = *((-current_byte);
    }
}
// END OTP-Write
CLEAR_PROG_OUT(); // set PROG_OUT=0
wait(600);

CLEAR_CSN(); //set CSN=0
wait(600);

SET_CSN(); //set CSN=1
wait(600);

SET_CLK();
wait(100);

INP_PROG_IN();
CLEAR_PROG_IN();
}
    
```

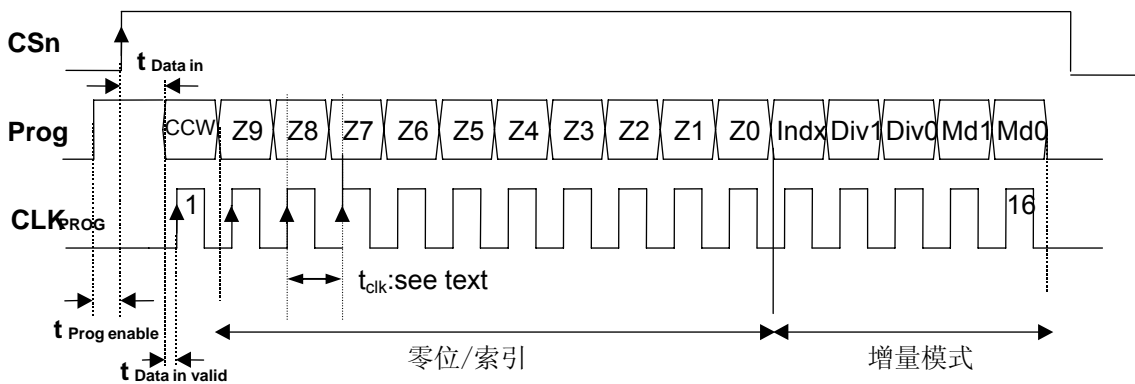


图7: 用户设置软写入的单个OTP写入操作

3.3 反复 OTP 软写入

如果上电后要多次改写AS5040的OTP寄存器，则必须使用下面阐述的软写入操作过程。该过程需要读取/写入OTP寄存器的所有32位。具体分为两部分：

- 16位用户设置
- 16位工厂设置

3.3.1 OTP 用户设置

OTP的用户设置部分包括第2.1.1节说明的各个位：CCW、零位和增量模式设置位。顾名思义，用户可以采用任何可能的组合设置来修改和编程这些位。

3.3.2 OTP 工厂设置

OTP的工厂设置部分包含若干调试位，可以在芯片的工厂测试过程中启用/禁用某些模块。一定不能以任何方式修改工厂设置！如果在软写入模式下不小心修改了工厂设置，可以通过上电复位将其清除。但如果在永久性OTP编程操作中修改了出厂设置，可能导致芯片报废。

因此建议在编程后要验证整个OTP寄存器。

注意，每颗芯片可能包含不同的工厂设置！

3.4 反复 OTP 软写入操作

为了重复执行软写入OTP寄存器的操作，必须读取/写入整个OTP，并需要正确终止OTP访问过程，以返回到常规的角度读取模式。

必须遵循以下操作步骤：

- 读取并保存32位OTP寄存器内容
- 根据需要修改16位用户设置。一定不要改动16位工厂设置！
- 将修改的设置写入整个32位OTP寄存器
- 根据需要可反复执行OTP写入

3.4.1 32 位 OTP 寄存器读取操作

整个OTP寄存器的读取操作如下面的图8所示。在Prog = 高且CLK = 低的情况下，CSn的上升沿将会启动该过程，后面跟随一个哑时钟周期(CLK #1)，此时Prog = 1且CSn = 0，然后在Prog = 1时又提供一个CSn上升沿。在CSn的第二个上升沿之后引脚Prog变为输出，驱动引脚Prog的控制器必须将其I/O状态由输出改为输入，以避免两个输出间产生冲突。Prog的状态是工厂设置的第1位(FS0)。每个后续的CLK上升沿将移出1位数据。紧随最后一个工厂设置位(FS15)，第17个时钟移出用户设置的第1位(ccw)。第33个时钟使Prog返回0。必须通过一个CSn脉冲来终止OTP读取操作过程，此时Prog和CLK均为低。

需注意，OTP读取操作总是读取“硬编程”的OTP内容。如果你通过“软写入”方式修改OTP，一个OTP读取操作将使OTP复位至硬编程的数值。因此，在一个OTP写入操作之后，不能用该读取操作来验证OTP的临时改写内容(3.4.2)。

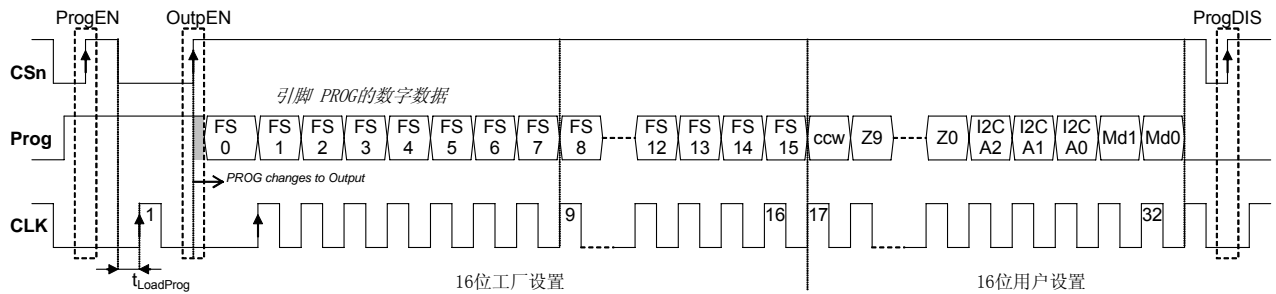


图8: 32位OTP寄存器读取操作过程

3.4.2 32位 OTP 寄存器写入操作

完整的OTP寄存器写入操作过程如图9所示。类似OTP读取操作，在Prog = 高且CLK = 低时，CSn的上升沿将启动该过程，但不在CSn上施加额外脉冲。编程数据必须加在Prog上，并随每个CLK的上升沿移入OTP。位流的顺序与OTP读取过程相同：

第1位是工厂设置的第0位(FS0)，接着是FS1……FS15。

在第17个时钟时，用户设置的第1位(ccw)被移入OTP寄存器。后面紧随零位数据(Z9:Z0)和增量模式编程位Indx、Div1、Div0、Md1和Md0。必须采用与终止OTP读取操作相同的方式终止OTP写入操作：Prog和CLK均为低时在CSn上产生一个脉冲。

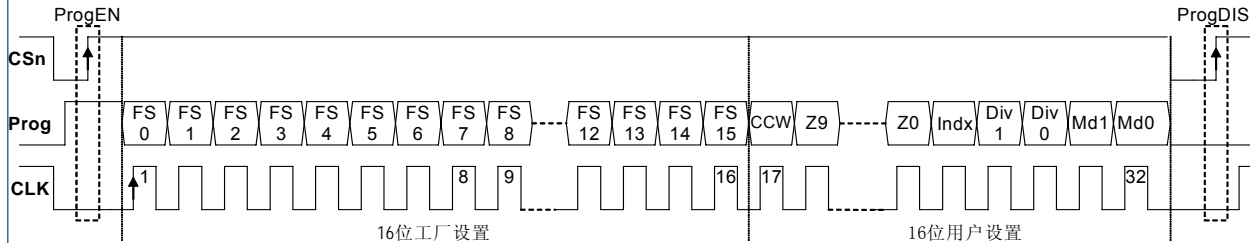


图9: 32位OTP寄存器写入操作过程

根据需要可以随意重复该写入操作，且无需在各次写入操作之间对芯片断电。

3.5 OTP 寄存器永久性写入操作及验证

只编程用户设置的简化OTP编程操作如图3和图4所示，它为编程AS5040提供了一种简单的方法，且无需考虑工厂设置的内容。

然而，为了确认编程用户设置时没有意外修改工厂设置，最好在编程结束后验证整个OTP寄存器的内容。具体操作过程如下：

- 读取并保存整个32位OTP寄存器的内容
- 根据需要修改16位用户设置。一定不要修改16位工厂设置。
- 如图10所示，将修改的设置写入整个32位OTP寄存器。建议将所有16位工厂设置(FS0...FS15)设为“0”，以确保所有这些位不被修改。
- $V_{PROG} = 7.3 - 7.5 V$ （见图10的中间部分）时CSn的上升沿将启动一个编程周期。在施加 V_{PROG} 编程电压的同时，提供33个长度为 $2\mu s$ $\pm 10\%$ 的CLK脉冲。这一步将对每个包含“1”的OTP位进行编程。包含“0”的OTP位不会被编程。
- 编程完成后，将Prog设为0并验证编程数据，如图10和图8所示。此外，你也可以采用32位OTP模拟回读方式（图11）来验证已编程和未编程熔丝的质量。

3.6 OTP 编程软件实现

```
void zappOTP(BYTE* buffer, BYTE num_bits)
{
    BYTE counter = 0;

    CLEAR_CSN(); //set CSN=0
    wait(10);

    CLEAR_CLK();
    wait(10);

    // Init Zapping Mode
    SET_V_ZAPP(); // set V_ZAPP = 1 wait(100);
    SET_CSN(); //set CSN=1 wait(50);

    // start zapping
    //num_bits = 32;
    for(counter = 0; counter <= num_bits; counter++)
    {
        SET_CLK(); //set CLK=1
        _nop_(); // wait for tzapp = 2us (typ.) High periode
        _nop_();
        _nop_();
        _nop_();
        _nop_();
        CLEAR_CLK(); //set CLK=0
        wait(1);

        CLEAR_V_ZAPP(); // set V_ZAPP = 1
        OUT_PROG_IN();
        CLEAR_PROG_IN();
        wait(80);
        INP_PROG_IN();
    }
}
```

```

_nop_();
_nop_();
_nop_();
_nop_();
SET_V_ZAPP(); // set V_ZAPP = 1
wait(1);
}

// End Zapping Mode CLEAR_V_ZAPP(); // set V_ZAPP = 0

OUT_PROG_IN(); CLEAR_PROG_IN();

wait(200);

NP_PROG_IN();
_nop_();
_nop_();
_nop_();
_nop_();
_nop_();

CLEAR_CSN(); //set CSN=1
wait(2);

SET_CSN(); //set CSN=1
wait(2);

CLEAR_CSN(); //set CSN=1
}
    
```

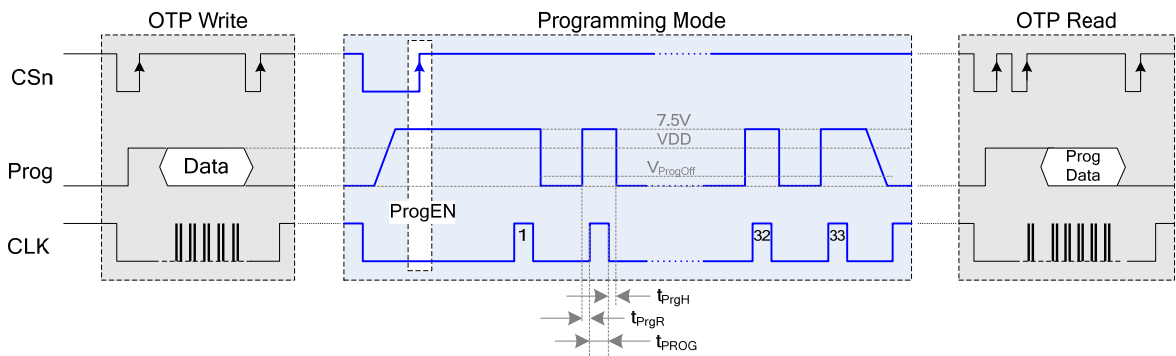


图10: 永久性编程和验证整个OTP寄存器

3.6.1 32 位 OTP 模拟读取

32位OTP模拟回读过程与16位用户设置模拟回读过程（见第2.4节）相同，除了需要提供32个时钟而不是16个时钟。请注意，与数字OTP读取操作（图8）相比，它的顺序恰好是反的。第1位是MD0，接下来是MD1、Div0、Div1、Z0……Z9、CCW、FS15……FS0。

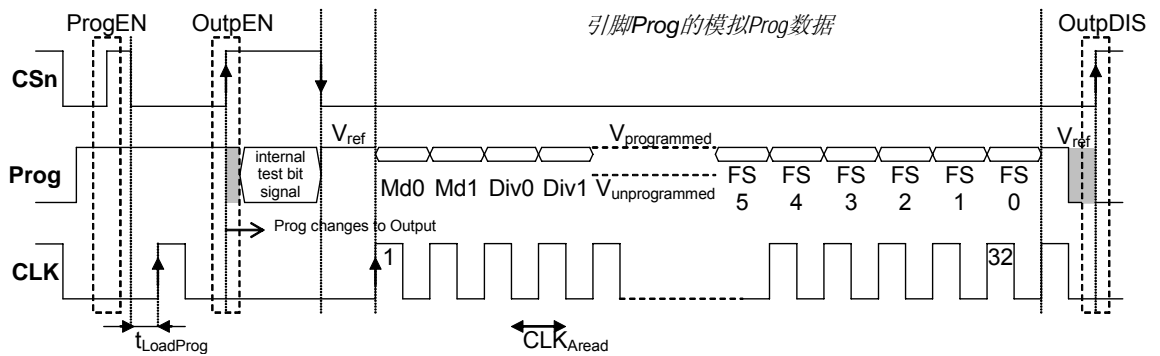


图11: 32位OTP模拟回读

4 反复执行永久性编程

你可以执行多次OTP“硬编程”操作（图4、图10），但你只能将“0”改为“1”，而不能通过硬编程将“1”改为“0”。如果你必须对一个已经编程的芯片重新执行永久性（“硬”）编程操作，你只能将那些需要从“0”改为“1”的位设为“1”。在重新编程的过程中，在OTP中已经包含“1”的位不能再设为“1”，否则会向已经编程的熔丝施加编程电流。

5 编程条件

表4列出了对AS5040编程时要求的参数。若需了解完整的工作参数列表，请参考AS5040数据资料。

（工作条件： $T_{amb} = -40$ 至 $+125^{\circ}\text{C}$ ， $V_{DD5V} = 3.0$ - 3.6V （3V工作电压） $V_{DD5V} = 4.5$ - 5.5V （5V工作电压），除非另有说明）

参数	符号	最小	典型	最大	单位	备注
编程启用时间	$t_{\text{Prog enable}}$	2			μs	Prog引脚上升沿和CSn上升沿之间的时间
开始写入数据	$t_{\text{Data in}}$	2			μs	
写入数据有效	$t_{\text{Data in valid}}$	250			ns	在CLK _{PROG} 上升沿写入数据
载入编程数据	$t_{\text{Load PROG}}$	3			μs	
CLK prog之前Vprog的上升时间	t_{PrgR}	0			μs	
CLK prog之后Vprog的持续时间	t_{PrgH}	0		5	μs	
写入数据-编程CLK _{PROG}	CLK _{PROG}			250	kHz	
CLK脉冲宽度	t_{PROG}	1.8	2	2.2	μs	在编程过程中：16个时钟周期
编程完后Vprog的保持时间	$t_{\text{PROG finished}}$	2			μs	下一次上电后，编程数据投入使用
编程电压	V_{PROG}	7.3	7.4	7.5	V	编程后必须关断
编程电压关断电平	V_{ProgOff}	0		1	V	编程电压必须下降至该电平
编程电流	I_{PROG}			130	mA	编程过程中
模拟读取CLK	CLK _{Aread}			100	kHz	模拟回读模式
已编程的齐纳电压（逻辑1）	$V_{\text{programmed}}$			100	mV	模拟回读模式下， $V_{\text{Ref}}-V_{\text{PROG}}$ （见第2.4节）
未编程的齐纳电压（逻辑0）	$V_{\text{unprogrammed}}$	1			V	

表4: 编程参数

6 修订历史

版本	日期	说明
1.2	2006年3月23日	加入C源代码
1.0	2005年11月2日 2005年10月27日	修订图1 最初修本本

7 联系方式

austriamicrosystems AG

A 8141 Schloss Premstätten, Austria

电话: +43 (0) 3136 500 0

传真: +43 (0) 3136 525 01

info@austriamicrosystems.com

8 版权说明

版权所有 © 2006, austriamicrosystems注册商标®。保留所有权利。本文材料未经版权所有人的事先书面批准, 不得进行复制、改编、合并、翻译、保存或使用。尽本公司所知, 本出版物中由austriamicrosystems AG提供的信息经确认为正确和准确。

