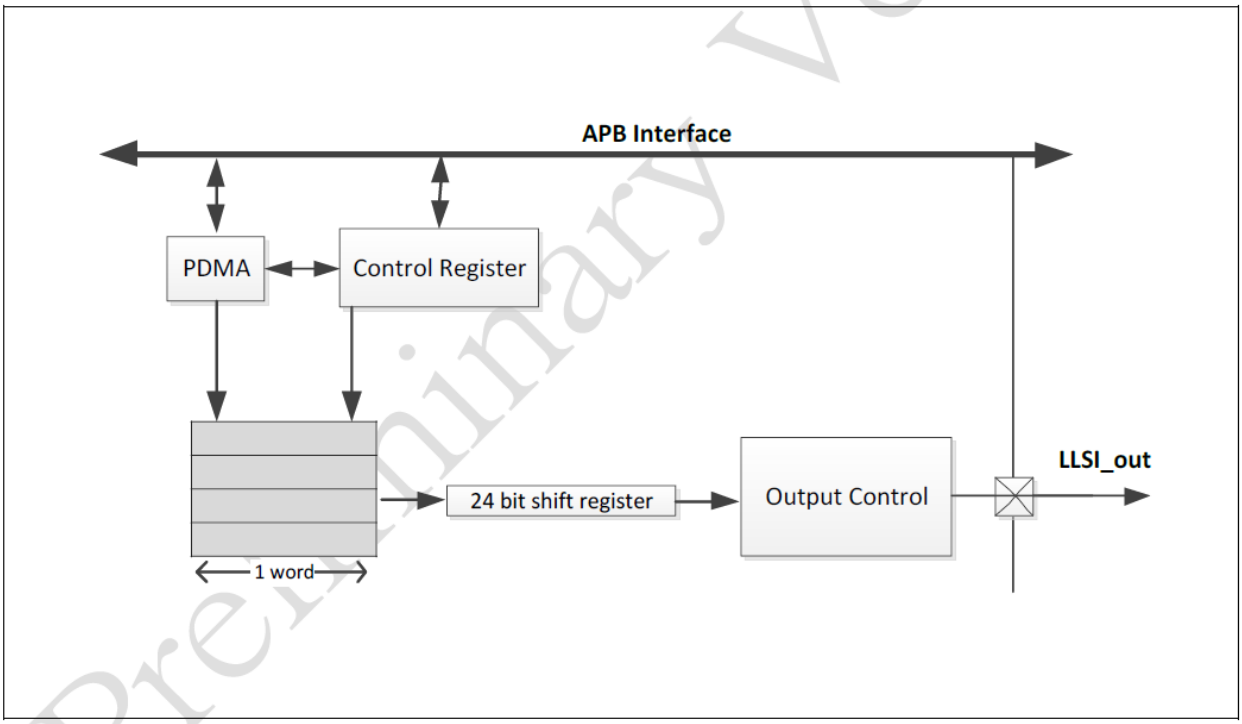
LLSI

LED Lighting Strip Interface

目录

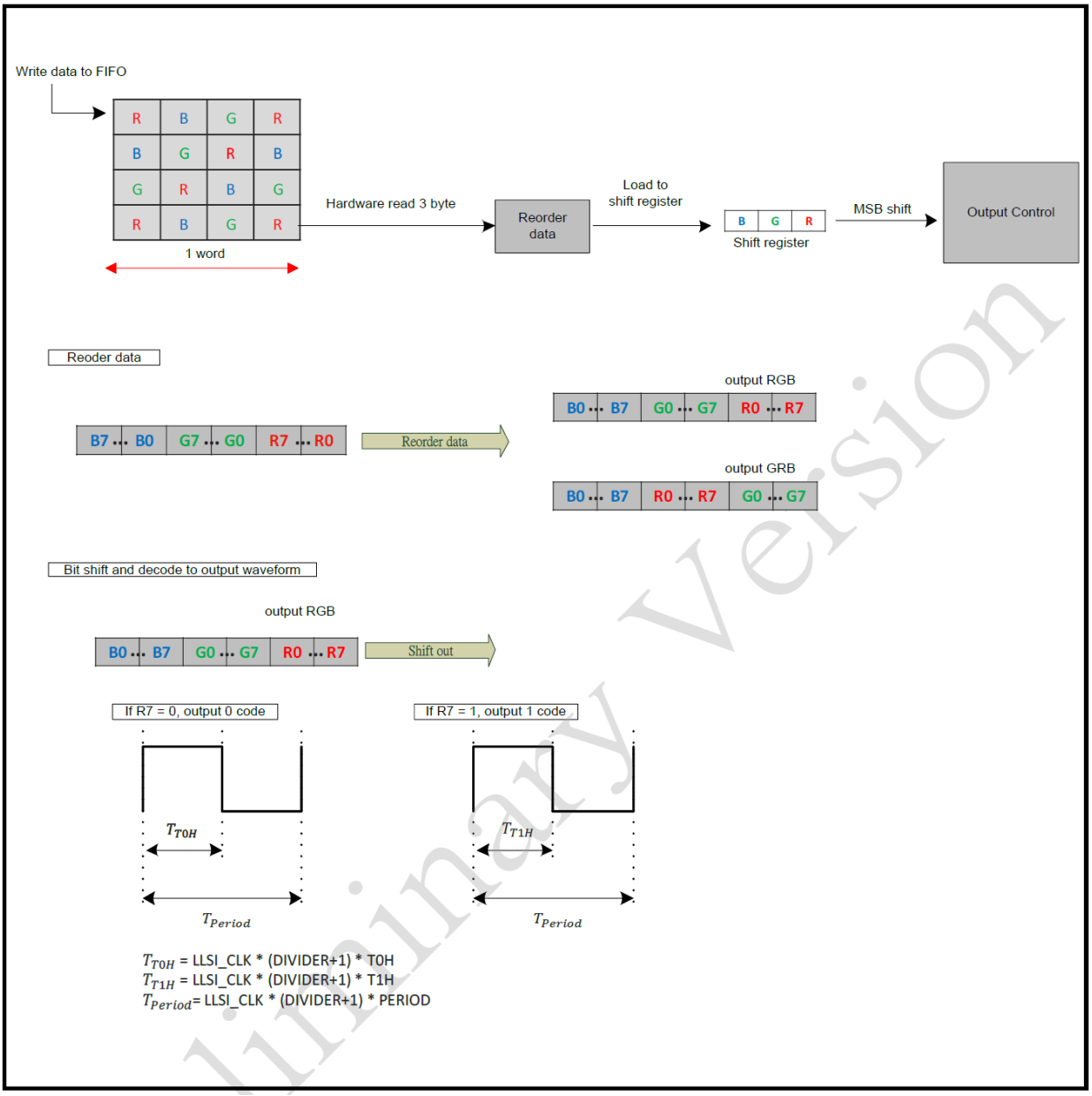
1 简介

LLSI 为 RGB 灯条控制器接口，每个通道可以用于将 RGB 数据来驱动含有数百个 LED 的单个灯条。最多包含 15 个 LLSI 通道。



2 主要特征

- 15 个 LLSI 通道，带空闲极性控制
- 每个 LLSI 通道包含 4 级深度的发送 FIFO
- 可配置的数据周期和编码
- 可配置的复位宽度
- 可配置的空闲电平
- 支持 RGB 和 GRB 格式
- *支持 RGBW 和 GRBW 格式*
- *支持自由数据格式*
- *支持自动发送附加数据*
- 支持 DMA 模式



3 引脚

LLSI 引脚为 LLSI1_TXD~LLSI15_TXD，具体引脚分配见相应型号的 MCU。

4 LLSI 地址

LLSI 挂在 APB 总线上，其控制寄存器地址范围为 0x4001_7400~0x4001_77FF，大小为 1KB。

5 LLSI 控制位/寄存器

5.1 模块控制位

RCC_APB2ENR[29]: LLSIEN
RCC_APB2RSTR[29]: LLSIRST

5.2 LLSI 通用寄存器

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	PR	DATPR[15:0]																RSTPR[15:0]															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	DUTYR	DUTY1[15:0]																DUTY0[15:0]															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	COMCR	RGBWF	RGBF	EXTDEN	EXTDBITS[4:0]					IDLEL		FREEF	FREEDBITS[4:0]					PSC[15:0]															
	Reset Value	0	0	0	1	1	1	1	1	0		0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10*(n)	CRn																	TXFIFORST			TXFTIE	TXFEIE	TXFNFIE	TCIE	TXRSTIE		TXFTCFG[1]	TXFTCFG[0]			DMAEN	TXRSTEN	EN
	Reset Value																	0			0	0	0	0	0		0	0			0	0	0
0x14*(n)	DRn	DAT[31:0]																															
	Reset Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x18*(n)	EXTDRn	EXTDAT[31:0]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C*(n)	ISRn																				TXFT	TXFE	TXFNF	TC	TXRSTIF								BSY
	Reset Value																				0	0	0	0	0								0

注：

1. *所有寄存器只能字写。*
2. *n=1~15*

位说明

Bits	R/W	Description
FREEDBITS[4:0]	R/W	**Free format Data Bits** 0~7 = 8 bits 8~31 = 9~32 bits ***仅在 FREEF=1 时有效***
FREEF	R/W	**Free Format of RGB data** 0 = RGB/RGBW format 1 = Free Format
IDLEL	R/W	**Idle Level** 总线空闲时的电平值。
EXTDBITS[4:0]	R/W	**Extra Data bits** 0~31 = 1~32 bits
EXTDEN	R/W	**Extra Data Enable**
RGBF	R/W	**RGB Format** 0 = RGB/RGBW 1 = GRB/GRBW ***仅在 FREEF=0 时有效***
RGBWF	R/W	**RGBW Format** 0 = RGB (3 bytes) 1 = RGBW (4 bytes) ***仅在 FREEF=0 时有效***
EN	R/W	**Enable**
TXRSTEN	R/W/HC	**Tx Reset Enable** 如果该位置 1，当 FIFO 为空及移位寄存器发送完毕，将发送 RESET 命令。 ***该位在 TC 置 1 时采样***
DMAEN	R/W	**DMA Enable** 如果该位置 1，当 FIFO 不满（TXFNF=1）时将产生 DMA 请求。
TXFTCFG[1:0]	R/W	**TXFIFO threshold configuration TXFIFO 空余阈值** 0 = 空余 1 个字 1 = 空余 2 个字 2 = 空余 3 个字 3 = 全空余
TXRSTIE	R/W	**TX Reset Completed Interrupt Enable**
TCIE	R/W	**TX Completed Interrupt Enable**
TXFNFIE	R/W	**TX FIFO Not Full Interrupt Enable**
TXFEIE	R/W	**TX FIFO Empty Interrupt Enable**
TXFTIE	R/W	**TX FIFO reaches Threshold Interrupt Enable**
TXFIFORST	W	**TX FIFO Reset** 同步 FIFO 指针；清除 FIFO

		该位写 1 会立即清除 FIFO 及指针，而不管当前是否正在发送，但不会中止当前移位寄存器中的数据发送
RSTPR[15:0]	R/W	**Reset Period**
DATPR[15:0]	R/W	**Data Period**
DUTY1[15:0]	R/W	**Duty of Data bit '1'**
DUTY0[15:0]	R/W	**Duty of Data bit '0'**
PSC[15:0]	R/W	**Clock Prescaler** LLSI_CLK = PCLK2/(PSC+1)
DAT[31:0]	W	**Data Write to FIFO** *仅允许字写入*
EXTDAT[31:0]	W	**Extra Data Reg** *数据右对齐，从 EXTDBITS 指定位开始发送*
TXRSTIF	R/W1C	**TX Reset completed Interrupt Flag** 由硬件置 1，写 1 清 0
TC	R/W1C	**TX Completed 发送完成标志** 由硬件置 1，写 1 清 0
TXFNF	R	**TX FIFO Not Full Flag** TX FIFO 不满时为 1
TXFE	R	**TX FIFO Empty Flag** TXFIFO 为空时为 1
TXFT	R	**TXFIFO Threshold flag** TXFIFO 空余空间>TXFTCFG 的配置时为 1
BSY	R	**TX busy** 1 = 正在发送数据或复位

R: Read, W: Write, HC: Hardware Clear; W1C: Write '1' Clear

6 LLSI 中断

单个通道的中断标志位与中断使能位相'与'，再所有通道相'或'，得到 LLSI 的中断。

LLSI 的中断向量如下：

Positio	Priority	Type of Priority	Acronym	Description	Address
F0:					
26	33	settable	SPI2/LLSI	SPI2/LLSI global interrupt	0x0000 00A8
F1/F3:					
96	103	settable	LLSI	LLSI global interrupt	0x0000 01C0

7 LLSI DMA 通道

CxS [3:0]	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
0000	TIM2_CH3	TIM2_UP TIM3_CH3	TIM3_CH4 TIM3_UP	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP	-	-
	-				TIM2_CH1	-	-
	-	-	-		TIM15_CH1 TIM15_UP TIM15_TRIG TIM15_COM	-	-
	ADC	-	TIM6_UP DAC_ Channel1	TIM7_UP DAC_ Channel2		-	-
	-	USART1_TX	USART1_RX	USART2_TX	USART2_ RX	USART3_ RX	USART3_TX
	LLSI1	LLSI2	LLSI3	LLSI4	LLSI5	USART4_ RX	USART4_TX
	-	SPI1_RX	SPI1_TX	SPI2_RX	SPI2_TX	LLSI6	LLSI7
	-	I2C1_TX	I2C1_RX	I2C2_TX	I2C2_RX	-	-
	-	TIM1_CH1	TIM1_CH2	-	TIM1_CH3	-	-
	-	-	TIM2_CH2	TIM2_CH4	-	-	-
	TIM17_CH1 TIM17_UP	-	TIM16_CH1 TIM16_UP	TIM3_CH1 TIM3_TRIG	-	-	-
0001	ADC	ADC	TIM6_UP DAC_ Channel1	TIM7_UP DAC_ Channel2	-	AES_IN	AES_OUT
0010	-	I2C1_TX	I2C1_RX	I2C2_TX	I2C2_RX	I2C1_TX	I2C1_RX
0011	-	SPI1_RX	SPI1_TX	SPI2_RX	SPI2_TX	SPI2_RX	SPI2_TX
0100	-	TIM1_CH1	TIM1_CH2	LLSI4	TIM1_CH3	TIM1_CH1 TIM1_CH2 TIM1_CH3	-
0101	-	-	TIM2_CH2	TIM2_CH4	LLSI5	LLSI6	TIM2_CH2 TIM2_CH4
0110	LLSI1	LLSI2	LLSI3	TIM3_CH1 TIM3_TRIG	-	TIM3_CH1 TIM3_TRIG	LLSI7
0111	TIM17_CH1 TIM17_UP	TIM17_CH1 TIM17_UP	TIM16_CH1 TIM16_UP	TIM16_CH1 TIM16_UP	-	TIM16_CH1 TIM16_UP	TIM17_CH1 TIM17_UP
1000	USART1_ RX	USART1_TX	USART1_RX	USART1_TX	USART1_ RX	USART1_ RX	USART1_TX
1001	USART2_ RX	USART2_TX	USART2_ RX	USART2_TX	USART2_ RX	USART2_ RX	USART2_TX
1010	USART3_ RX	USART3_TX	USART3_ RX	USART3_TX	USART3_ RX	USART3_ RX	USART3_TX
1011	USART4_ RX	USART4_TX	USART4_ RX	USART4_TX	USART4_ RX	USART4_ RX	USART4_TX
1100	USART5_ RX	USART5_TX	USART5_ RX	USART5_TX	USART5_ RX	USART5_ RX	USART5_TX
1101	USART6_ RX	USART6_TX	USART6_ RX	USART6_TX	USART6_ RX	USART6_ RX	USART6_TX

CxS[3:0]	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
0000	none				
0001	-	-	TIM6_UP DAC_Channel1	TIM7_UP DAC_Channel2	ADC
0010	I2C2_TX	I2C2_RX	-	-	-
0011	-	-	SPI1_RX	SPI1_TX	-
0100	-	-	-	-	-
0101	-	-	-	-	-
0110	LLSI1	LLSI2	LLSI3	LLSI4	LLSI5
0111	LLSI6	LLSI7	LLSI8	LLSI9	LLSI10
1000	USART1_TX	USART1_RX	USART1_RX	USART1_TX	USART1_TX
1001	USART2_TX	USART2_RX	USART2_RX	USART2_TX	USART2_TX
1010	USART3_TX	USART3_RX	USART3_RX	USART3_TX	USART3_TX
1011	USART4_TX	USART4_RX	USART4_RX	USART4_TX	USART4_TX
1100	USART5_TX	USART5_RX	USART5_RX	USART5_TX	USART5_TX
1101	USART6_TX	USART6_RX	USART6_RX	USART6_TX	USART6_TX
1110	USART7_TX	USART7_RX	USART7_RX	USART7_TX	USART7_TX
1111	USART8_TX	USART8_RX	USART8_RX	USART8_TX	USART8_TX

8 LLSI 时钟

LLSI 模块的输入时钟为 APB2 时钟 PCLK2。其位定时的时钟 LSSI_CLK 为 PCLK2 分频后得到（PCLK2/(PSC+1)）。

9 LLSI 编程

对 LLSI 编程比较简单，将相关时序设置好，对 DAT 寄存器写入数据就可以发送出去了，但需留意 FIFO 操作及 Reset 的发送。

9.1 FIFO 操作

9.1.1 RGB 数据格式

RGB 数据按 24 位组织的，但 FIFO 是 32 位宽，其与 RGB 数据的对应是从 R 开始，并按字节依次对应 R、G、B、R、G…。但一帧数据（两次 Reset 之间的数据定义为一帧）里最后一个 RGB 数据，不一定刚好位于 FIFO 的 Bit[31:24]（最高字节），而下一帧数据的写入，会重新从 Bit[7:0]对应 R，因此

FIFO 读指针可能对应不上新的数据。为了解决此问题，在发送完成（TC 置 1）且 TXRSTEN=1 时，会将 FIFO 读、写指针清零来实现同步，MCU 可以在 TC 置 1 后写入下一帧数据。
MCU 也可以在需要时，将 TXFIFORST 位写 1 来同步指针，然后开始写入 FIFO。

注：

- *如果需要提前停止数据发送，采用以下 2 种方式：*
  - ✧ *使用将 EN 清 0 的方式。EN 清 0 会同时复位 FIFO 指针、打断当前正在移位的数据。然后将 EN 写 1，再向 FIFO 写入新的数据*
  - ✧ *将 FIFORST 写 1，再清 0，等待 TC 为 1（此处要求必须有数据正在发送），再向 FIFO 写入新的数据*

  前种方式立即停止数据发送，后种方式会等待当前移位寄存器的数据发送完毕。
- *发送数据过程中，如果 FIFO 未及时更新数据，发送会暂停，输出维持在 IDLEL 指示的电平*

9.1.2 RGBW 数据格式

对于 RGBW 数据，刚好是 1 个字（4 个字节），因此，采用字写 FIFO 时，不会出现某个字包含有效及无效数据。但在硬件设计上，FIFO 指针仍然和 RGB 数据格式时一样，采用同样的 FIFO 指针管理方式。

9.1.3 自由数据格式

当采用自由数据格式时，对 FIFO 数据的存放格式不作定义，硬件上只是将 FIFO 数据按照从高位至低位的顺序，从 FREEDBITS 指定的位开始依次发送。
对于长度为 1 个字节（8 位）、2 个字节（9~16 位）、3 个字节（17~24 位）、4 个字节（25~32 位）举例如下：

8 bit data

	Bit 31			Bit 0
Word 3				D[7:0]
Word 2				D[7:0]
Word 1				D[7:0]
Word 0				D[7:0]

10 bit data

	Bit 31			Bit 0
			D[9:8]	D[7:0]
			D[9:8]	D[7:0]
			D[9:8]	D[7:0]
			D[9:8]	D[7:0]

20 bit data

Word 3		D[19:16]	D[15:8]	D[7:0]
Word 2		D[19:16]	D[15:8]	D[7:0]
Word 1		D[19:16]	D[15:8]	D[7:0]
Word 0		D[19:16]	D[15:8]	D[7:0]

28 bit data

D[27:14]	D[23:16]	D[15:8]	D[7:0]
D[27:14]	D[23:16]	D[15:8]	D[7:0]
D[27:14]	D[23:16]	D[15:8]	D[7:0]

数据发送时，将从蓝色背景的最高有效位开始发送。
9.1.1 节中的 FIFO 指针同步规则仍然适用。

9.2 Reset 发送

通过将 TXRSTEN 置 1，可以在所有数据（包括附加数据）完成后，自动发送 Reset 命令。

9.3 附加数据发送

部分 LED 驱动控制芯片除了接收显示数据（RGB 数据）外，还会附加控制数据，而且控制数据格式及长度与 RGB 数据不一致。如下图所示：

该帧数据中先是 N 个字节的 LED 数据，随后紧跟 20 位的 RGBW 增益数据。

为了简便的发送此类数据，允许软件在 RGB 数据发送完毕后，自动将附加数据寄存器（EXTDR）的数据发送。附加数据最长为 32 个位（由 EXTDBITS 设置）。
如果允许了附加数据发送，TC 置位会发生在附加数据发送完毕时刻，而不会在 RGB 数据发送完成时置 1。同样的，如果将 TXFRSTEN 置 1，Reset 命令会在附加数据发送完毕后才发送。

9.4 软件编程

LLSI 可以工作在 DMA 模式或纯软件操作，以下简单阐述编程方法。

- DMA 模式

1) 初始化 DMA：长度设置为 1 帧数据的存放长度（字长，非 LED 数据长度），源和目的都为字操作，源地址为 RGB 数据存放的起始地址，目的地址为 LLSI DAT 寄存器；开启 DMA 完成中断
2) 初始化 LLSI：设置数据格式及时钟分频信息、位、Reset 定时信息，设置 FIFO 阈值；开启 TC 中断
3) 启动 LLSI
4) 启动 DMA
5) 等待 LLSI TC 中断
   - 在 LLSI TC 中断中，更新 DMA 设置并重启 DMA，以使 LLSI 发送下一帧数据

- 纯软件模式

1) 初始化 LLSI：设置数据格式及时钟分频信息、位、Reset 定时信息，设置 FIFO 阈值；开启 TXFNF 中断、TC 中断
2) 设置好第一帧数据的源地址，RGB 字计数器清 0
3) 启动 LLSI
4) 等待中断
   - 在 TXFNF 中断中，如果 RGB 字计数器<数据总长度：
     Copy 一个源地址指向的 RGB 数据至 DAT 寄存器；源地址++；字计数器++
     否则：
     关闭 TXFNF 中断；更新下一帧数据的源地址，RGB 字计数器清 0
   - 在 TC 中断中，打开 TXFNF 中断

以上两种方式，如果需要发送附加数据，在初始化时将 EXTDEN 置 1，EXTDBITS 设定位数，待发数据写入 EXTDR。
如果需要发送 Reset，初始化时将 TXRSTEN 置 1。

10 版本历史

Date	Revision	Author	Changes
2022/9/6	0.10	Dick Hou	初版
2022/9/22	0.11		* 调整寄存器的位分配 * 增加 BSY 位 * 增加 RGBW 数据格式 * 增加自由数据格式 * 增加附加数据发送 * 修改 FIFORST 和 TXRSTEN 的说明
2022/9/23	0.12		* 调整自由格式的 FIFO 数据 * DR 只能字写

11 -