RST0112R

DataSheet

1 Inch 1-Mega Pixel CMOS Image Sensor

V6.0

2026.03.06

应用

- 日夜兼容的监控
- 夜视设备
- 科学研究
- 无人机

特性

- 高灵敏度 12μm 像素
- 高信噪比
- 像素三转换增益
- 支持 HDR
  - Dual-CG combine
  - Dual-CG + LOFIC combine
  - PWL
- 支持黑电平校正
- 集成片上 OTP （4Kbit）
- 支持 OTP-DPC
- 高帧率下支持 1280x800 /1280x720 典型分辨率
- 支持 Updown、Mirror、任意尺寸裁窗口
- 支持 binning：
  - 2x2 color binning
  - 2x2 mono binning
- 支持外部触发曝光
- 支持外部同步，支持多图像传感器同步
- 支持闪光灯控制
- 支持 Rolling Global 曝光

关键指标（典型值）

- 产品料号：RST0112R
- 芯片尺寸：21.2mm × 15.2mm
- 分辨率：有效 1280x800（100 万像素）
- 像素尺寸：12μm×12μm
- 灵敏度：200V/lux•s
- 读出噪声：<0.65e@15.5x
- 满阱容量：65ke$^-$
- 信噪比：48dB
- 动态范围：68dB
  - 94dB@HDR mode
  - 120dB@LOFIC HDR mode
- 主时钟频率：典型 24MHz
- 帧率：30fps/60fps/120fps/180fps
- 曝光方式：Rolling、Rolling Global
- 量化深度：10Bit/12Bit
- 控制接口：I2C
- 输出的数据格式：

  10/12/16/24 BIT RAW
- 数据输出接口：
  - 1/2/4 Lane MIPI
  - 4 Lane SLVS
- 功耗：
  - 285mW @ 10bit 180fps
  - TBD @ 12bit 50fps

目录

图片索引

表格索引

1. 传感器简介

RST0112R 是一款高灵敏度、低噪声的 1 英寸 100 万像素格式 CMOS 图像传感器。 通过采用高灵敏的 12μm 像素，实现了出色的低光灵敏度、信噪比、满阱容量、量子效率和动态范围。提供了丰富的编程模式，可以方便地控制帧率、曝光时间、增益和其他参数。它还提供以下图像功能：高动态模式、镜像和翻转、选窗功能、黑电平校准、坏点校正、HDR 功能和黑太阳消除等。它支持高达 180fps @100 万像素格式的高帧率。具备这些特性的图像传感器，将为用户在微弱光环境下带来出色的成像体验。

1.1. 传感主要特点

- 高灵敏度 12μm 像素
- 支持 1280×800/1280×720 典型分辨率
- 支持 HDR
- 支持黑电平校正
- 支持坏点校正
- 支持 Upside-down, Mirror, 任意尺寸裁窗口
- 支持外部同步，支持双 Camera
- 支持 Raw10、Raw12、Raw16 和 Raw24
- 接口：MIPI&SLVS
- I2C 接口

1.2. 主要应用领域

- 日夜兼容的监控；
- 夜视设备；
- 科学研究；
- 无人机；

1.3. 芯片指标列表

表 1 - 1 芯片指标列表

参数	数值
产品料号	RST0112R
芯片尺寸	21.2mm×15.2mm
分辨率	有效 1280x800（100 万像素）
像素尺寸	12μm × 12μm
有效感光尺寸	15360μm×9600μm
灵敏度	200V/luxs
读出噪声	＜0.65e@15.5x
满阱容量	65ke-
信噪比	48dB
动态范围	68dB@normal mode 94dB@HDR Dual-CG mode 120dB@HDR Triple-CG mode
主时钟频率	典型 24MHz
帧频	30fps/60fps/120fps/180fps
曝光方式	rolling
量化深度	10Bit/12Bit
控制接口	I2C
数据输出接口	MIPI, SLVS

2. 传感器结构

2.1. 总体结构

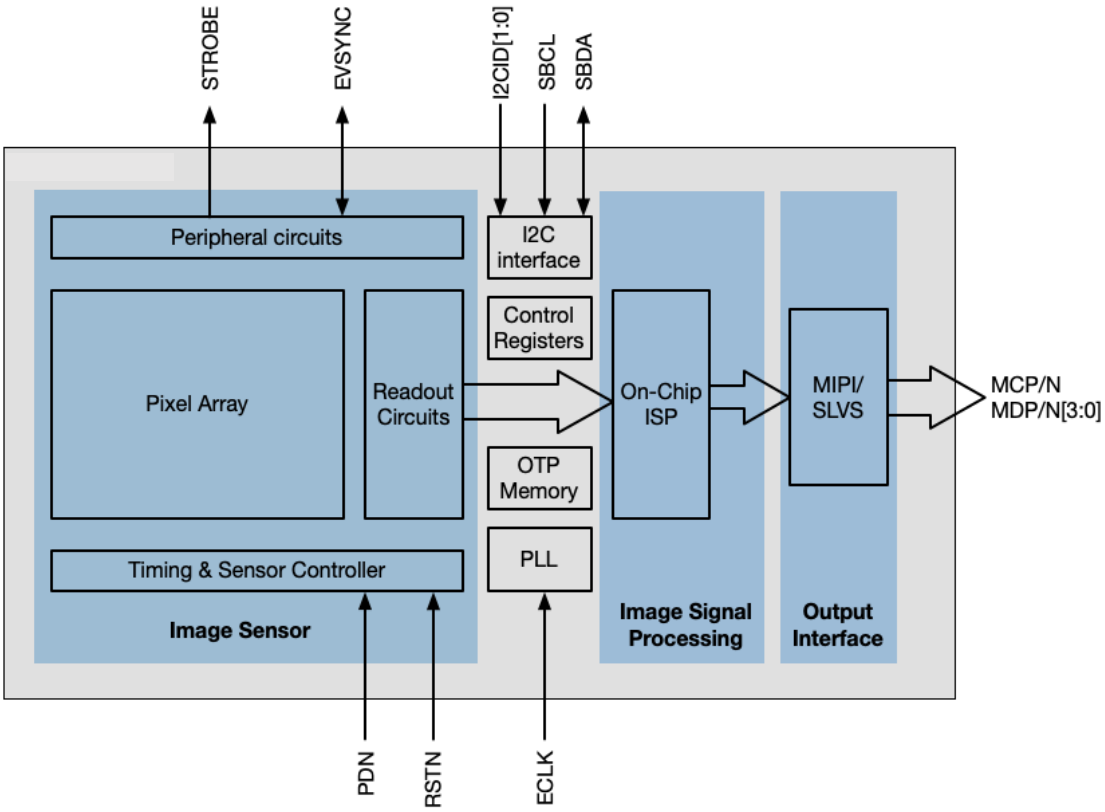


图 2 - 1 芯片整体结构

该芯片由像素阵列、读出电路、外围电路及数字电路组成，其中数字电路实现了时序控制、寄存器配置、OTP 读写、暗电平 / 坏点校正、HDR、PWL 压缩、binning、镜像翻转及 MIPI/SLVS 输出等核心功能。

2.2. 参考原理图设计

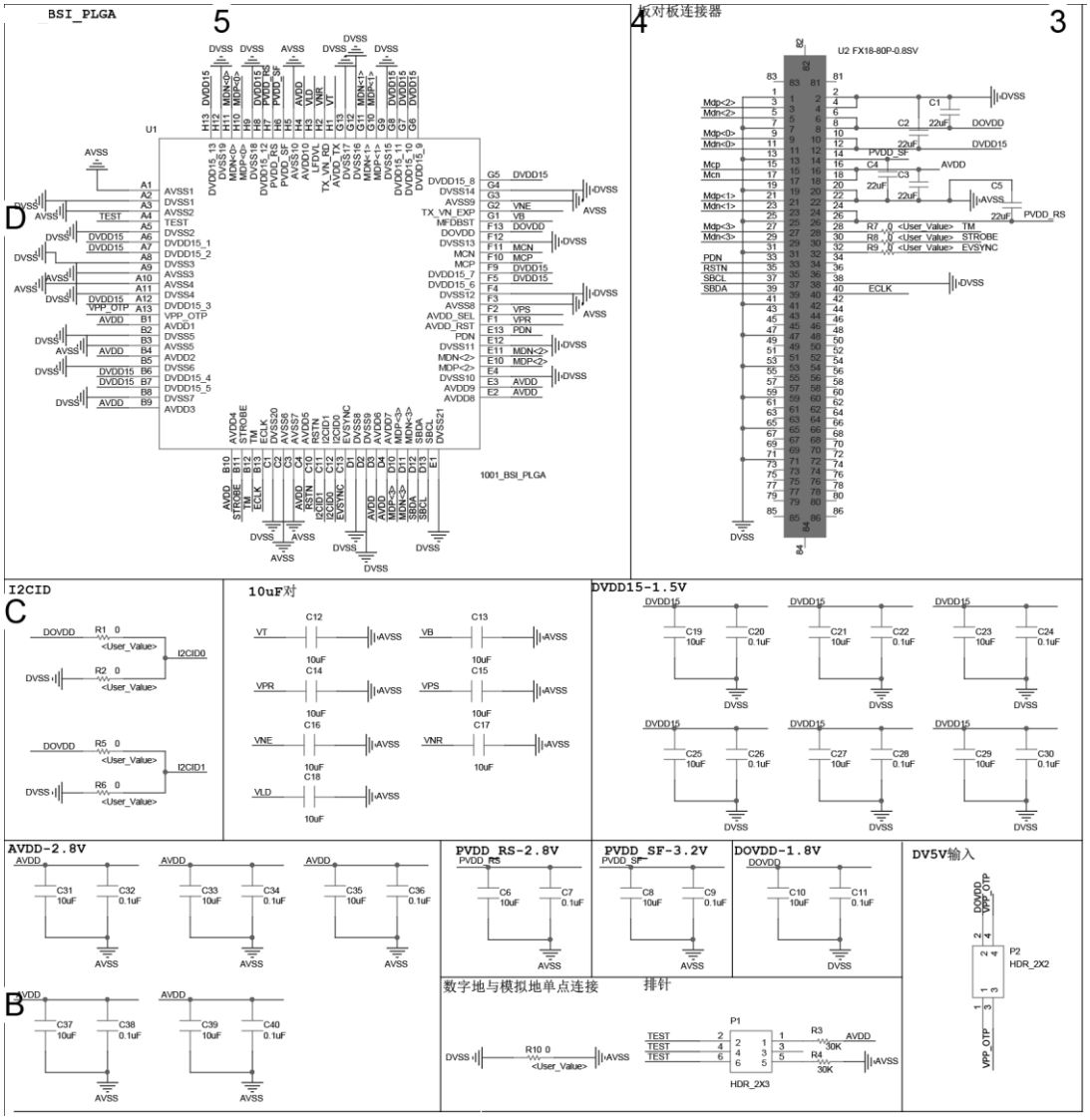


图 2 - 2 RST0112R_PLGA 参考原理图设计

2.3. 像素阵列

2.3.1. 像素阵列

有效像素区域是 1280×800，外侧有 4 个行 border 像素、8 个列 border 像素，如图 2 - 3。

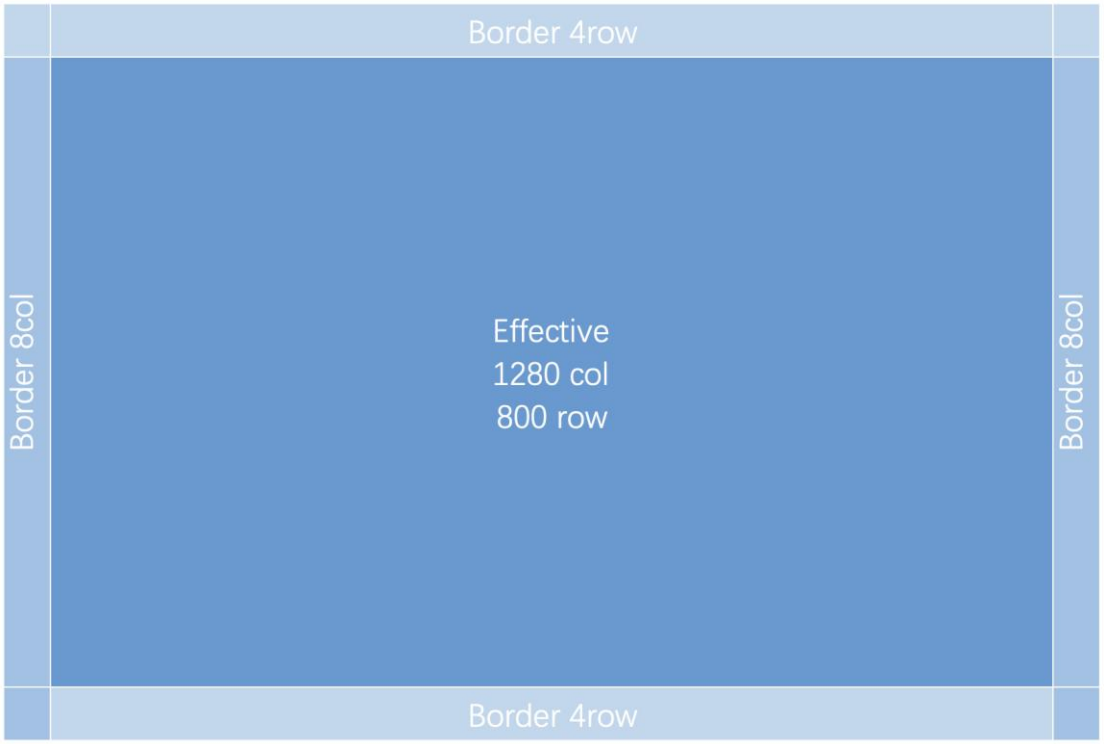

Border 4row
Border 8col
Effective
1280 col
800 row
Border 8col
Border 4row

图 2 - 3 像素阵列结构

2.4. 芯片封装信息

RST0112R 提供封装形式为 PLGA。产品型号为：RST0112R-M5A1，管壳信息如下。

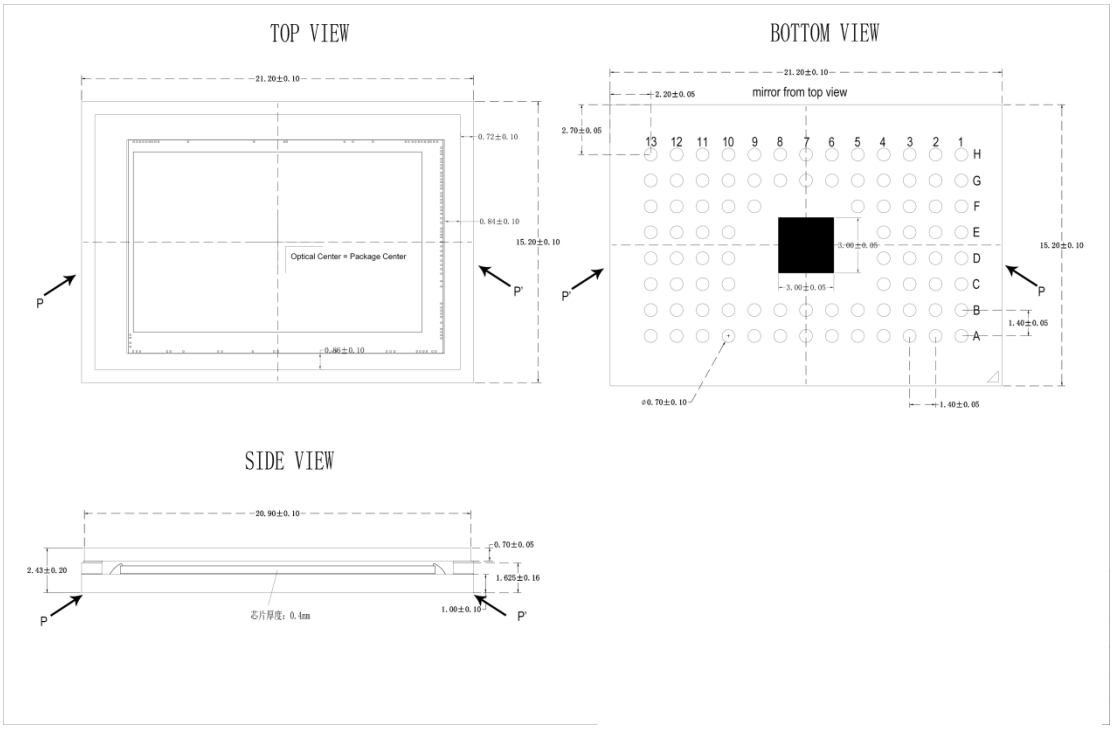


图 2 - 4 RST0112R-M5A1 的 PLGA 封装图

PLGA 管壳封装的实物图如下所示：

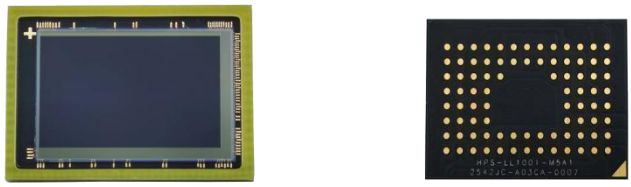


图 2 - 5 RST0112R-M5A1 的 PLGA 封装实物图

实物 3D 成像视图如下所示：

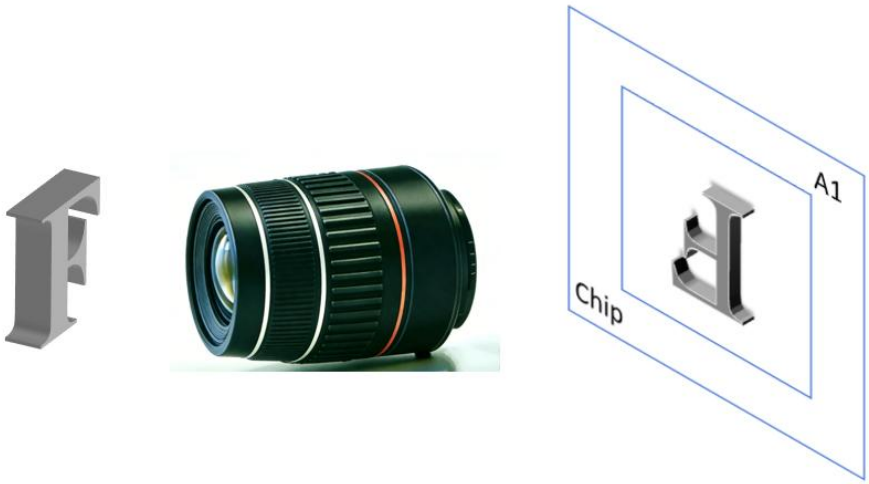


图 2 - 6 3D 成像视图

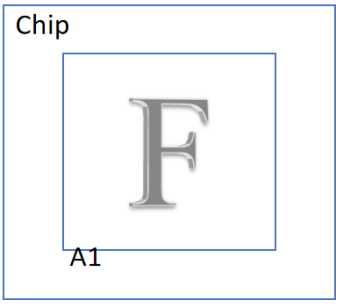


图 2 - 7 在 Chip 上成像图

图 2 - 8 拍摄实际效果图

2.5. 引脚说明

传感器芯片包含 99pin 引脚，使用 PLGA 封装后引脚映射关系和说明如下：

表 2 - 1 芯片 PLGA 引脚说明

PLGA 管脚序号	管脚名称	A/D	I/O/P	引脚说明
A1	AVSS	A	P	Analog ground
A2	DVSS	D	P	Digital ground
A3	AVSS	A	P	Analog ground
A4	TEST	A	I	Leave it floating.
A5	DVSS	D	P	Digital ground
A6	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
A7	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
A8	DVSS	D	P	Digital ground
A9	AVSS	A	P	Analog ground
A10	AVSS	A	P	Analog ground
A11	DVSS	D	P	Digital ground
A12	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
A13	VPPOTP	A	P	Provide DOVDD at regular working mode.
B1	AVDD	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
B2	DVSS	D	P	Digital ground
B3	AVSS	A	P	Analog ground
B4	AVDD	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
B5	DVSS	D	P	Digital ground
B6	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
B7	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
B8	DVSS	D	P	Digital ground
B9	AVDD	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
B10	AVDD	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
B11	STROBE	D	O	Output flash signal om this pin.This pin has internal pull down resistor Min 73.4K.
B12	TM	D	I	Leave it floating or tie to DVSS.
B13	ECLK	D	I	External clock. typical 24MHz, optional 6 to 36MHz.
C1	DVSS	D	P	Digital ground
C2	AVSS	A	P	Analog ground
C3	AVSS	A	P	Analog ground
C4	AVDD	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
C10	RSTN	D	I	Reset input. 0 to reset chip, 1 to work normally. This pin has internal pull down resistor Min 73.4K.
C11	I2CID1	D	I	I2C ID selection.This pin has internal pull down resistor Min 73.4K.

PLGA 管脚序号	管脚名称	A/D	I/O/P	引脚说明
C12	I2CID0	D	I	I2C ID selection.This pin has internal pull down resistor Min 73.4K.
C13	EVSYNC	D	I/O	External exposure control or continous imaging synchronous pin.This pin has internal pull down resistor Min 73.4K.
D1	DVSS	D	P	Digital ground
D2	DVSS	D	P	Digital ground
D3	AVDD	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
D4	AVDD	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
D10	MDP<3>	A	I/O	MIPI data3_p
D11	MDN<3>	A	I/O	MIPI data3_n
D12	SBDA	D	I/O	Data line for I2C bus.
D13	SBCL	D	I	Clock line for I2C bus.
E1	DVSS	D	P	Digital ground
E2	AVDD	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
E3	AVDD	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
E4	DVSS	D	P	Digital ground
E10	MDP<2>	A	I/O	MIPI data2_p
E11	MDN<2>	A	I/O	MIPI data2_n
E12	DVSS	D	P	Digital ground
E13	PDN	D	I	Power down input. 0 to power chip down, 1 to power up. This pin has internal pull down resistor Min 73.4K.
F1	VPR	A	I/O	IO for sensor internal voltage.Place 4.7uF cap to AVSS.
F2	VPS	A	I/O	IO for sensor internal voltage.Place 4.7uF cap to AVSS.
F3	AVSS	A	P	Analog ground
F4	DVSS	D	P	Digital ground
F5	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
F9	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
F10	MCP	A	I/O	MIPI clock_p
F11	MCN	A	I/O	MIPI clock_n
F12	DVSS	D	P	Digital ground
F13	DOVDD	D	P	IO power for digital signal, typical 1.8V.Place 2uF cap to DVSS.
G1	VB	A	I/O	IO for sensor internal voltage.Place 4.7uF cap to AVSS.

PLGA 管脚序号	管脚名称	A/D	I/O/P	引脚说明
G2	VNE	A	I/O	IO for sensor internal voltage.Place 4.7uF cap to AVSS.
G3	AVSS	A	P	Analog ground
G4	DVSS	D	P	Digital ground
G5	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
G6	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
G7	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
G8	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
G9	DVSS	D	P	Digital ground
G10	MDP<1>	A	I/O	MIPI data1_p
G11	MDN<1>	A	I/O	MIPI data1_n
G12	DVSS	D	P	Digital ground
G13	DVSS	D	P	Digital ground
H1	VT	A	I/O	IO for sensor internal voltage.Place 4.7uF cap to AVSS.
H2	VNR	A	I/O	IO for sensor internal voltage.Place 4.7uF cap to AVSS.
H3	VLD	A	P	IO for sensor internal voltage.Place 4.7uF cap to AVSS.
H4	AVDD	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
H5	AVSS	A	P	Analog ground
H6	PVDD_SF	A	P	Analog power, range is from 2.8V to 3.2V. Place 2uF cap to AVSS.
H7	PVDD_RS	A	P	Analog power, typical 2.8V. Place 2uF cap to AVSS.
H8	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.
H9	DVSS	D	P	Digital ground
H10	MDP<0>	A	I/O	MIPI data0_p
H11	MDN<0>	A	I/O	MIPI data0_n
H12	DVSS	D	P	Digital ground
H13	DVDD15	D	P	Digital core power, typical 1.5V. Place 2uF cap to DVSS.

表 2 - 2 PLGA 引脚合并电容组建议说明

管脚名称	A/D	I/O/P	可合并引脚序号
AVDD	A	P	B1,B4,B9,B10,C4,D3,D4,E2,E3,H4
DVDD15	A	P	A6,A7,A12,B6,B7,F5,F9,G5,G6,G7,G8,H8,H13

管脚名称	A/D	I/O/P	可合并引脚序号
DOVDD	A	P	F13

上述引脚可就近合并连接，只通过 1 组电容来进行滤波

2.6. 典型连接方式

下图是该传感器的典型引脚连接方式。

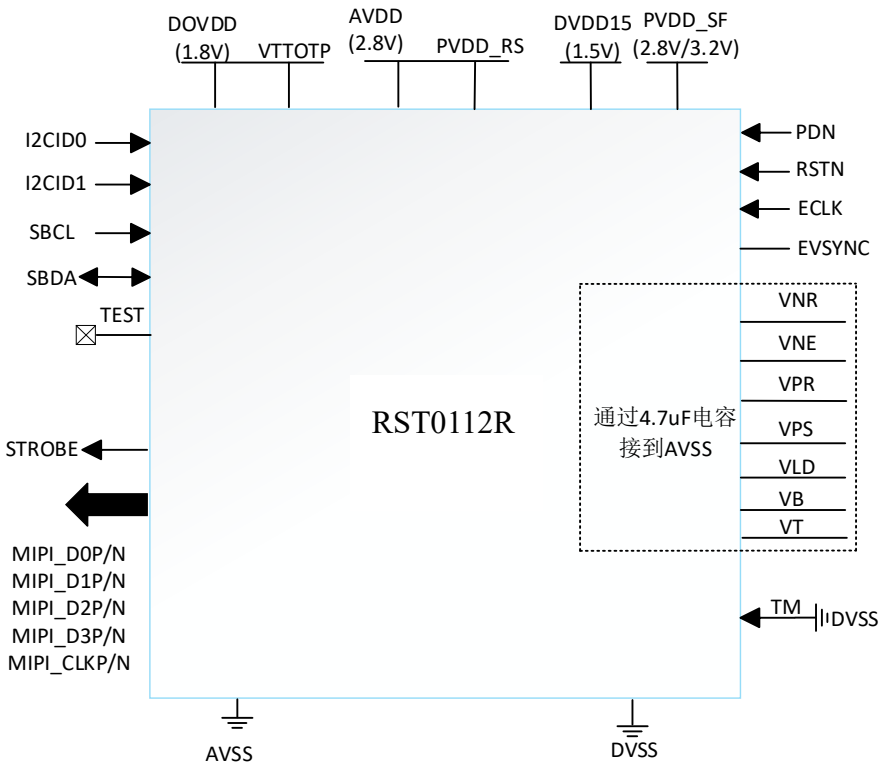


图 2 - 9 传感器引脚典型连接方式

电源需求：

（1）AVDD、PVDD_RS 和 PVDD_SF 是模拟电源，DVDD15、DOVDD、VPPOTP 是数字电源，其中 VPPOTP 连接到 DOVDD 的电源供电，因此外部需提供电源包括 AVDD（典型 2.8V）、PVDD_SF（典型 3.2V）、DVDD15（典型 1.5V）和 DOVDD（典型 1.8V）；

（2）模拟/数字电源需要分别对模拟地/数字地进行滤波，滤波电容使用≥2uF 的组合，滤波电容放置尽量靠近 sensor，对于同类电源引脚位置接近的情况，可以合并使用一个≥2uF 的电容；

（3）模拟电路易受噪声影响，需要使用低噪声 LDO 提供模拟电源，另外推荐将 PVDD_SF 独立 LDO 供电，方便电压调节以获得更优性能；

（4）为保证供电可靠性，条件允许情况下使用不小于 40mil 的线宽，不得低于 20mil，且尽量降低走线长度；

（5）确保模拟电源、地和数字电源地尽量隔离开；

（6）模拟地和数字地可在供电最源头位置进行单点连接；

信号需求：

（1）单端信号交互电平为 DOVDD 电压，典型为 1.8V；

（2）ECLK 的典型输入 24MHz；

（3）MIPI 差分走线阻抗设定为 100 欧姆，差分线全部做等长控制，同一对差分线长度差异<10mil，不同差分对之间长度差异<20mil，尽量在差分线外侧使用地线屏蔽，避免对其他线对的干扰；

（4）如果不需要外部同步功能，可以将 EVSYNC 做浮空处理；TEST 端可做浮空处理；TM 端可做浮空处理；

（4）I2CID 端口可使用下拉或上拉电阻将其电位拉低、拉高，从而选择 I2C 寄存器地址；

（5）VPR、VPS、VNE、VNR、VB、VT 和 VLD 端口需要分别连接对模拟地的≥4.7uF 电容；

其他需求：

（1）发热器件需要远离 sensor 放置，且尽量利用铺铜的电源、地层快速散热。

（2）钢网推荐厚度 0.12mm。

（3）SMT 贴片之前必须进行烘烤，具体细节如下：

- ◆ 常规烘烤温度为：120C-150C；
- ◆ 常规烘烤时间为：24h-48h。

2.7. 电气特性

表 2 - 3 绝对最大额定值（以上所有电压都是 to pad 电压）

项目	符号	绝对最大额定值	单位
模拟电源电压	V_{AVDD}	-0.3~4.0	V
I/O 电源电压	V_{DOVDD}	-0.3~4.0	V
数字电源电压	V_{DVDD}	-0.3~1.7	V
像素电源	V_{PVDD_RS},V_{PVDD_SF},	-0.3~4.0	V
I/O 输入电压	—	-0.3~DOVDD+0.3	V
I/O 输出电压	—	-0.3~DOVDD+0.3	V
工作温度	—	-45 ~ 70	℃

贮存温度	—	-45 ~ 80	℃

表 2 - 4 直流电气特性（以上所有电压都是 to pad 电压）

项目	符号	最小值	典型值	最大值	单位
电源					
模拟电源电压	VAVDD	2.7	2.8	2.9	V
像素电源	$V_{PVDD\_SF,}$	2.8	3.2	3.2	V
I/O 供电电压	VDOVDD	1.7	1.8	1.9	V
数字电源	VDVDD	1.4	1.5	1.6	V
电流（MIPI 模式，典型条件:AVDD=2.8V,DOVDD=1.8V,DVDD=1.5V,VPP=3V,4Lane MIPI full size@10Bit 180fps）					
模拟电源电流	IAVDD	—	20.5	—	mA
I/O 电源电流	IDOVDD	—	10.9	—	mA
数字电源电流	IDVDD	—	130.3	—	mA
像素电源电流	IVPP	—	3	—	mA
电流（MIPI 模式，典型条件：AVDD=2.8V，DOVDD=1.8V, DVDD=1.5V,VPP=3V,4Lane MIPI full size @12Bit 50fps）					
模拟电源电流	IAVDD	—	TBD	—	mA
I/O 电源电流	IDOVDD	—	TBD	—	mA
数字电源电流	IDVDD	—	TBD	—	mA
像素电源电流	IVPP	—	TBD	—	mA
数字输入（典型条件：AVDD=2.8V，DOVDD=1.8V）					
输入低电平	VIL	—	—	0.3×DOVDD	V
输入高电平	VIH	0.7×DOVDD	—	—	V
输入电容	CIN	—	—	20	pF
数字输出（25pF 标准负载）					
输入高电平	VOH	0.9×DOVDD	—	—	V
输入低电平	VOL	—	—	0.1×DOVDD	V
串行接口输入（SCL 和 SDA）					
输入低电平	VIL	-0.3	0	0.3×DOVDD	V
输入高电平	VIH	0.7×DOVDD	DOVDD	DOVDD+0.3	V

表 2 - 5 交流特性（TA=25°C，AVDD=2.8V，DOVDD=1.8V）

项目	符号	最小值	典型值	最大值	单位
交流参数					
软复位设置时间	—	—	—	1	ms
更改分辨率设置时间	—	—	—	1	ms
配置寄存器设置时间	—	—	—	300	ms
晶振和时钟输入					
输入时钟频率	FEXTCLK	6	—	40	MHz
输入时钟上升/下降时间	—	—	—	5	ns

2.8. Chief Ray Angle

CRA 为 0°

2.9. QE 曲线

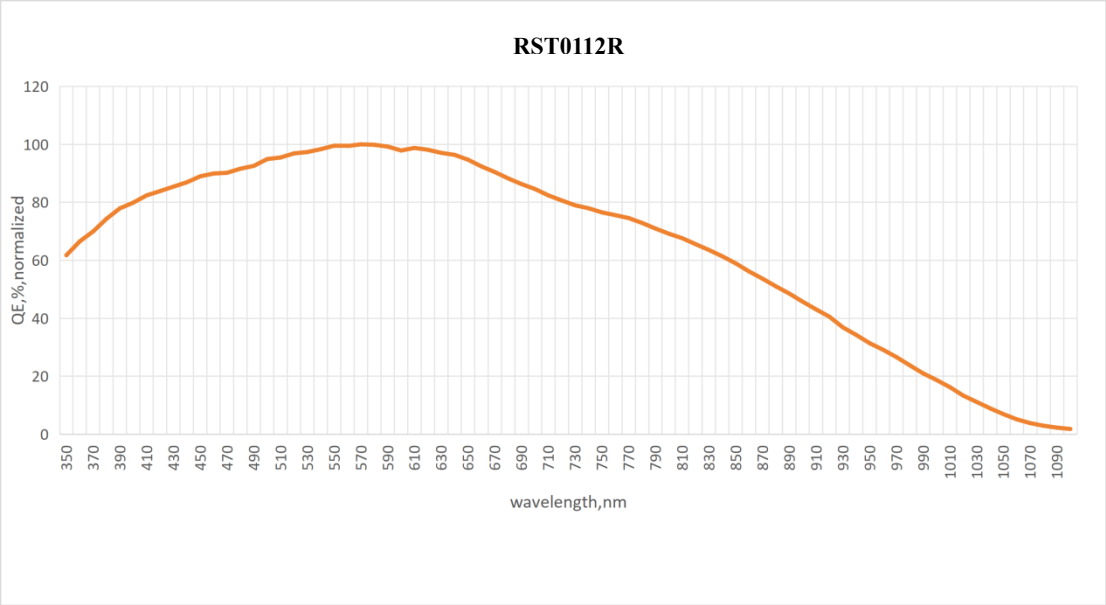


图 2 - 10 量子效率曲线图

3. 工作方式

3.1. 上电时序

芯片上电过程中电源和信号的顺序如下：

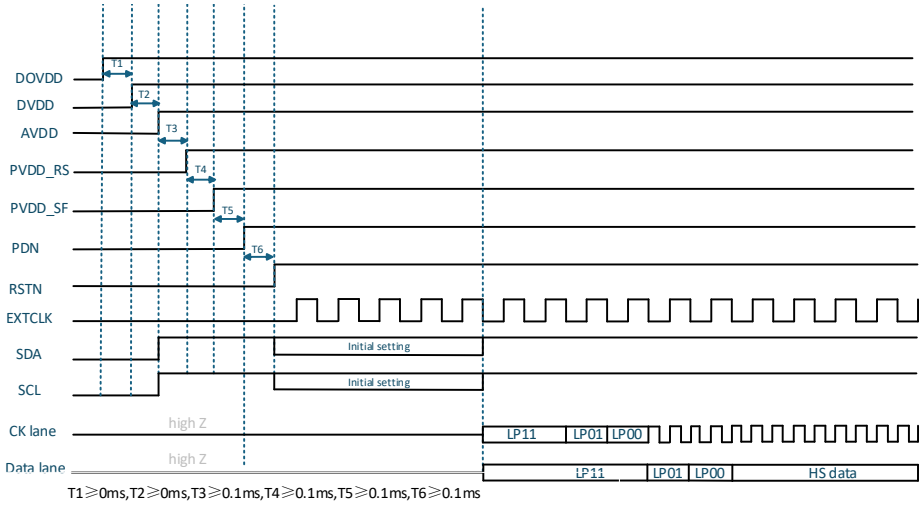


图 3 - 1 上电时序

3.2. I2C

RST0112R 的写入地址和读取地址可通过 i2cid0 和 i2cid1 端口来选择。

1）当 i2cid1 为 0，i2cid0 为 0 时，写入地址为 0x78，读取地址为 0x79；

2）当 i2cid1 为 0，i2cid0 为 1 时，写入地址为 0x7a，读取地址为 0x7b；

3）当 i2cid1 为 1，i2cid0 为 0 时，写入地址为 0x7c，读取地址为 0x7d；

4）当 i2cid1 为 1，i2cid0 为 1 时，写入地址为 0x7e，读取地址为 0x7f；

3.2.1. 信息类型

RST0112R 支持的信息类型如图 3 - 2 所示，包含 7bit 从机地址(slave address: {5’b0111_1,i2cid1,i2cid0})，16bit 子地址(sub address)和 8bit 数据(data)，重复起始条件不在图 3 - 2 中，但是在图 3 - 3、图 3 - 4 和图 3 - 5 中均有体现。

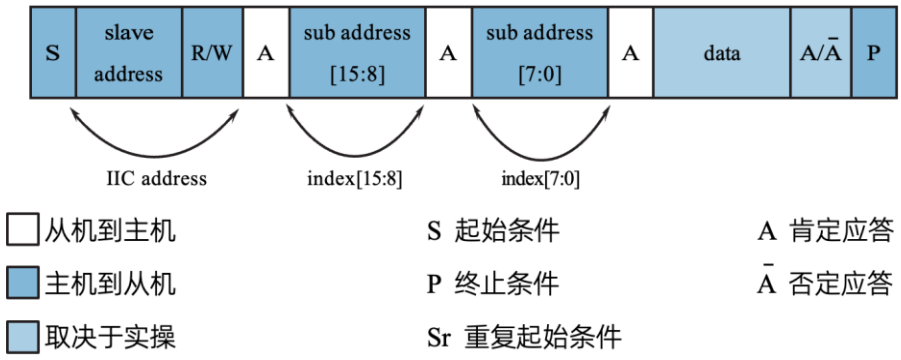


图 3 - 2 消息类型

3.2.2. 读/写操作

RST0112R 支持 4 种不同的读操作和 2 种不同的写操作

1）从任意位置单次读

2）从任意位置连续读

3）从当前位置单次读

4）从当前位置连续读

5）从任意位置单次写

6）从任意位置连续写

在一次读/写操作后，子地址会在芯片内部自动加 1。

在从任意位置单次读过程中，对于目标子地址，主机会先进行一次虚拟写操作，然后发送一个重复起始条件通过读操作再次对从机进行寻址。确认从机地址后，从机开始向 SDA 线输出数据，然后主机通过发送否定应答和终止条件来终止读操作，如图 3 - 3 所示。

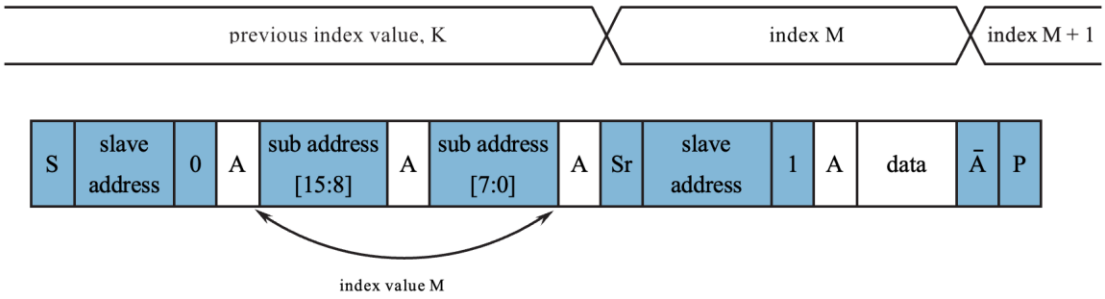


图 3 - 3 从任意位置单次读操作示意图

如果主机直接对从机寻址进行读操作，不进行虚拟写操作，那么从机会将上一次使用的子地址配置到 SDA 线上，然后主机会发送否定应答和终止条件来终止读操作，如图 3 - 4 所示，这样就完成了一次从当前位置单次读操作。

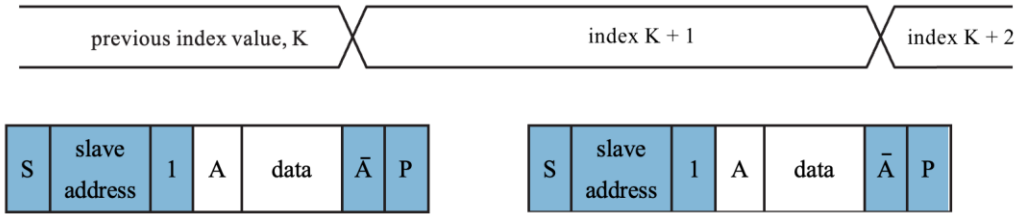


图 3 - 4 从当前位置单次读操作示意图

从任意位置连续读操作如**错误!未找到引用源。**所示。对于目标子地址，主机会进行一次虚拟写操作，得到从机的肯定应答后会发送一个重复起始条件然后通过读操作再次对从机进行寻址。如果主机在接收的数据后发送了肯定应答，它将作为从机的一个信号，表明读操作将在下一个子地址继续。当主机读完最后一个字节的数据时，会发送否定应答和终止条件来终止读操作。

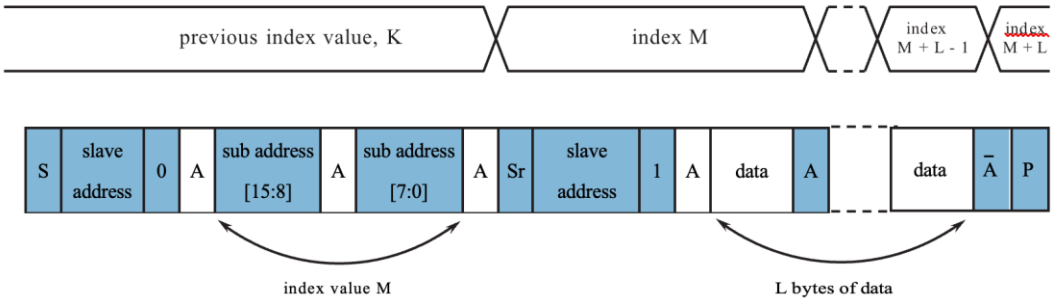


图 3 - 5 从任意位置连续读操作示意图

从当前位置连续读操作类似于从随机位置连续读。唯一的区别就是没有虚拟写地址的操作如**错误!未找到引用源。**所示。同样，主机会发送否定应答和终止条件来终止读操作。

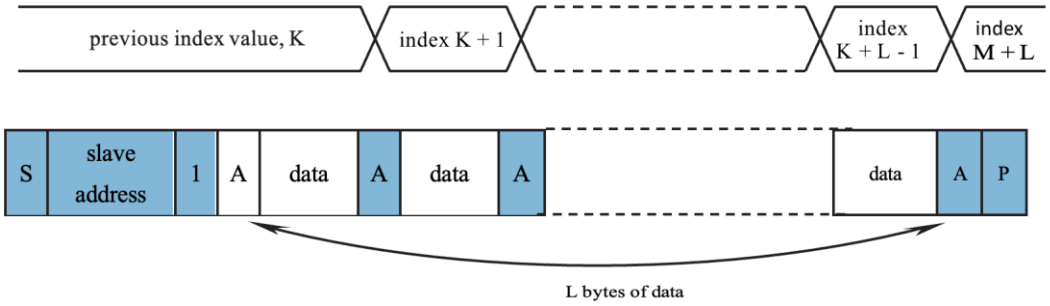


图 3 - 6 从当前位置连续读

任意位置的单次写操作如图 3 - 7 所示。主机对从机发送写操作，在从机确认后设置相应的子地址和数据，然后主机通过发送终止条件终止写操作。

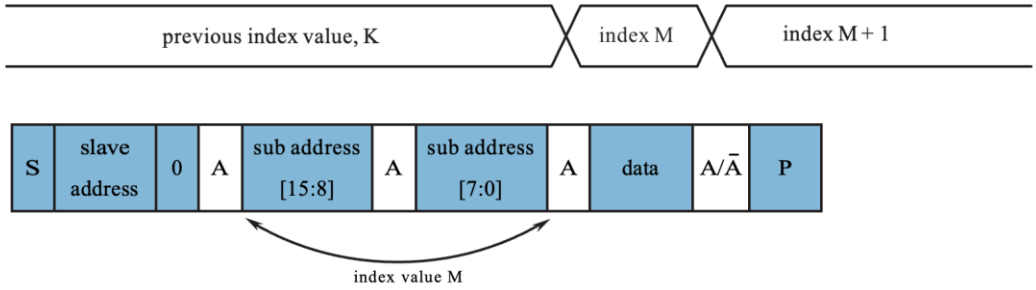


图 3 - 7 从任意位置单次写

连续写操作如图 3 - 8。在每字节数据后，从机会自动增加子地址。连续写操作通过主机通过发送终止条件终止。

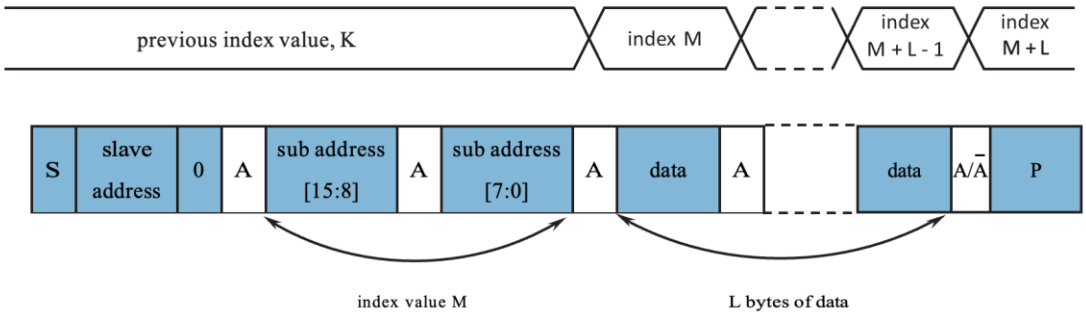


图 3 - 8 从任意位置连续写

3.2.3. I2C 时序

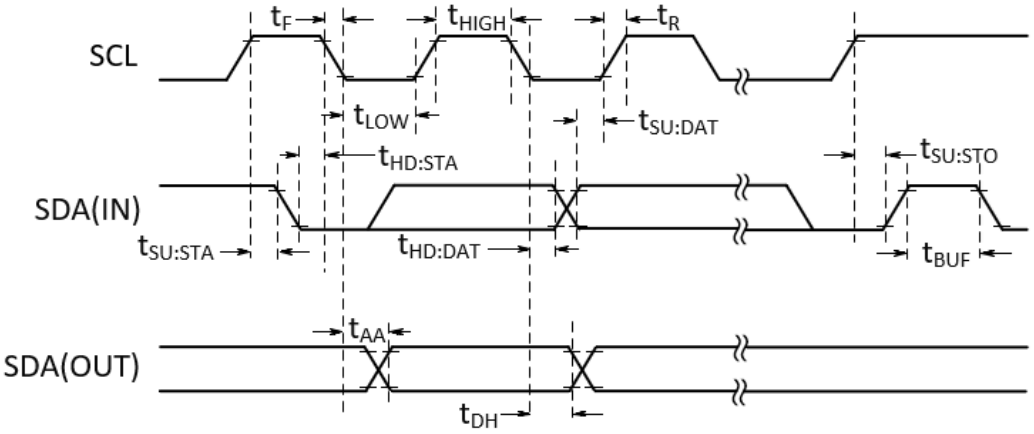


图 3 - 9 I2C 接口时序

表 3 - 1 I2C 接口时序特征

symbol	parameter	min	typ	max	unit
f_{SCL}	clock frequency			400	kHz
t_{LOW}	clock low period	1.3			μs
t_{HIGH}	clock high period	0.6			μs
t_{AA}	SCL low to data out valid	0.1		0.9	μs
t_{BUF}	bus free time before new start	1.3			μs
$t_{HD:STA}$	start condition hold time	0.6			μs
$t_{SU:STA}$	start condition setup time	0.6			μs
$t_{HD:DAT}$	data in hold time	0			μs
$t_{SU:DAT}$	data in setup time	0.1			μs
$t_{SU:STO}$	stop condition setup time	0.6			μs
t_R, t_F	I2C rise/fall times			0.3	μs
t_{DH}	data out hold time	0.05			μs

1）I2C 时序以 400kHz 模式为基础

2）在上升沿开始或下降沿结束时显示的定时测量为 30%

在上升/下降沿中间显示的定时测量为 50%

计时测量显示在上升沿的末端和/或下降沿的开始表示 70%

3.2.4. 寄存器表述方式说明

兼容标准 I2C 协议，采用 16bit 寄存器地址，8bit 数据位宽。

现就下文中寄存器的表述方式进行说明：

写寄存器对应的实际代码示例：

```
i2c_write(0x5000, 0x01); //对 0x5000 地址写 0x01
```

读寄存器对应的实际代码示例：

```
i2c_read(0x5000); //读 0x5000 地址内容
```

3.3. 帧时序

3.3.1. 正常曝光操作（rolling）

Rolling Shutter 是通过 Sensor 逐行曝光的方式实现的。在曝光开始的时候，Sensor 逐行扫描逐行进行曝光，直至所有像素点都被曝光。

在该项目中，快门操作和积分时间如下图所示，横轴为时间序列，纵轴为垂直地址。为了简化，快门和读出操作以行为单位进行记录。

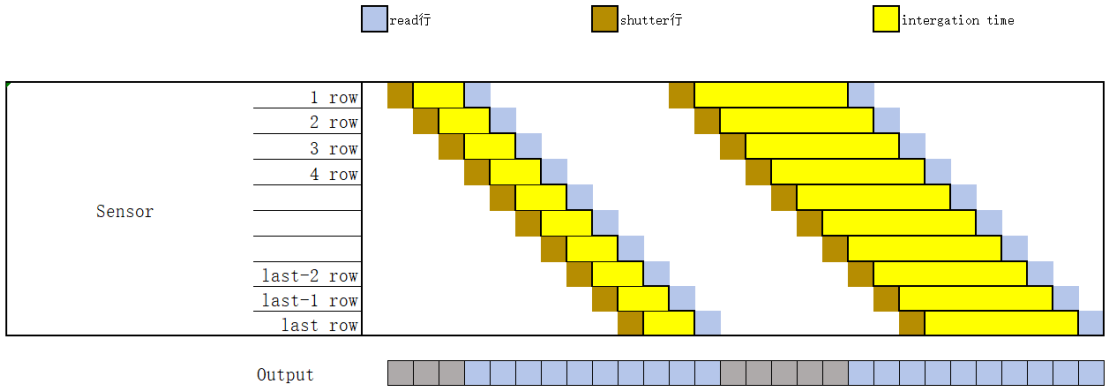


图 3 - 10 Image drawing of shutter operation

积分时间可以通过改变电子快门时间来控制。在电子快门设置中，积分时间由 TEXP 寄存器控制。

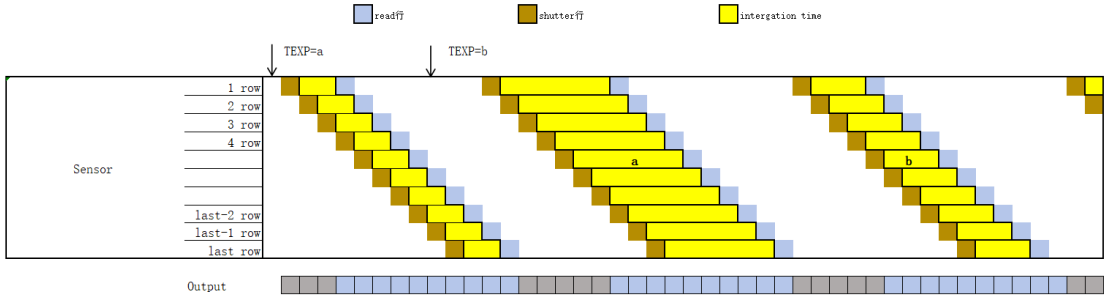


图 3 - 11 Image drawing of integration time control within a frame

内部曝光模式下，通过 TEXP（{0x3301,0x3302}）设置曝光时间，并需要通过 NewSetting（0x33fe=0x01）来使能生效。如图 3 - 12 所示，在第 N 帧写入的曝光时间，会在第 N+2 帧生效。

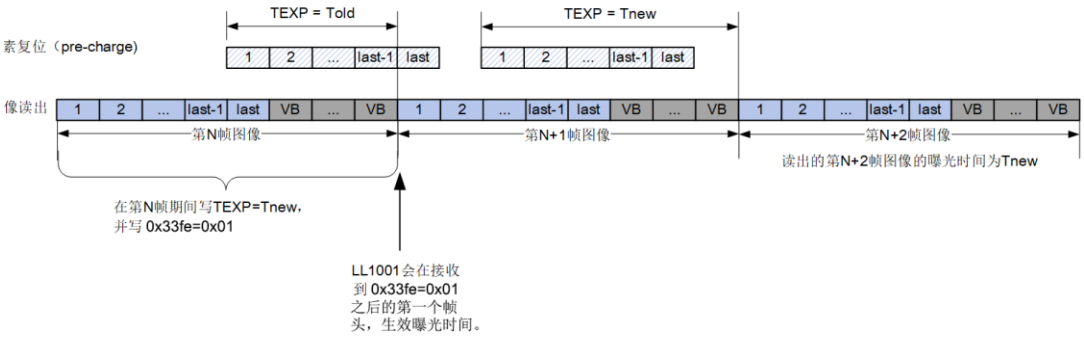


图 3 - 12 曝光生效机制

3.3.2. Rolling Global

Rolling Global Shutter 是通过快速 Shutter Sensor 所有行，随后在曝光结束后逐行量化、读出图像。配合可控光源，可达到接近全局曝光的性能。

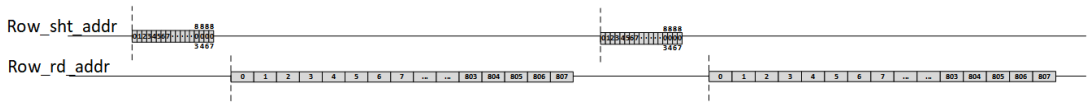


图 3 - 13 rolling global

3.4. 复位方式

该芯片复位包括通过寄存器的软复位和通过 IO 的硬复位。

- 硬复位

RSTN 信号下拉，此时全芯片数字电路全部重置且维持当前状态，直至 RSTN 信号上拉为 1。

- Softrst（软复位）

向 SOFTRST（0x0103）寄存器中写入 0x01，系统执行 1243.2ns 的软复位后自动恢复。此时内部产生的时钟、寄存器、sensor_ctrl 部分、Datapath 被复位。

- Restart

向 RESTART（0x0104）寄存器中写入 0x01，系统执行 1243.2ns 的复位后自动恢复。此时 sensor_ctrl、Datapath 被复位。

表 3 - 2 复位相关寄存器

Address	Register name	Default	R/W	Description
0x0103	SOFTRST	0x00	W	Bit[0]:softrst
0x0104	RESTART	0x00	W	Bit[0]:restart

3.5. 待机模式

通过将 PDN 引脚下拉置低电平，芯片进入 powerdown 待机模式，此模式下素有数字、模拟模块关闭，芯片功耗达到 43uW。

建议在此模式下同样将 RSTN 拉低。

为了回复正常，需先拉高 PDN、再拉高 RSTN，再通过 I2C 发送芯片初始化指令。

3.6. 成像模式

3.6.1. 普通模式

如需切换至普通模式需向寄存器 READ_MODE 写入切换，传感器工作在普通模式，具体配置如下。

表 3 - 3 linear 模式

Address	Register name	Default	R/W	Description
0x330d	READ_MODE	0x00	R/W	bit[0] : linear HCG bit[1] : linear MCG bit[2] : linear LCG(test mode)

3.6.2. HDR+LOFIC 模式

HDR 模式是指一个 pixel 下有两个增益：Middle Conversion Gain（MCG）和 High Conversion Gain (HCG)，而 HDR+LOFIC 模式是指一个 pixel 下有三个增益：MCG 和 HCG 和 LCG(Low Conversion Gain);

这两个模式下可以保证图像传感器能在极端弱光条件下捕获图像/视频而不牺牲高光条件下的性能。

在 HDR 模式下，HCG 和 MCG 拥有不同的像素转换增益，增益倍率约 17，具体倍率可通过 OTP 读出。

在 HDR+LOFIC 模式下，HCG、MCG、LCG 数据交替输出，每行先输出 HCG 数据再输出 MCG 最后输出 LCG 数据，不同数据在 MIPI 长包的包头中通过 virtual channel（VC 位）区分；MCG 时 VC 位输出寄存器 VIRTUAL_CHANNEL[1:0]，默认为 2'b00，HCG 时 VC 位输出 VIRTUAL_CHANNEL[3:2]，默认为 2'b01，LCG 时 VC 位输出 VIRTUAL_CHANNEL[5:4]，默认为 2'b10。

表 3 - 4 HDR+LOFIC 相关寄存器

Address	Register name	Default	R/W	Description
0x330d	READ_MODE	0x00	R/W	Bit[2:0]: 3'b011 HDR 3'b100 HDR+LOFIC
0x480b	VIRTUAL_CHANNEL	0x24	R/W	Bit[5:0]:virtual_channel

3.6.3. 低噪声模式

低噪声模式为相关多采样，不同于普通模式，低噪声模式是对信号进行多次采样；相关多采样模式有 2 次、4 次、8 次模式。

传感器工作到低噪声模式时，需要切换的寄存器主要为 CMS_NUM 具体如下表。

表 3 - 5 低噪声模式相关寄存器表

Address	Register name	Default	R/W	Description
0x330C	CMS_NUM	0x11	R/W	Bit[7:4]:HCG_cms_num Bit[3:0]:MCG_cms_num

3.7. 外部触发模式

3.7.1. 外部曝光

通常情况下 RST0112R 工作在内部曝光模式，即 RST0112R 按照通过 I2C 设定的曝光时间、帧时间，连续曝光、连续出图。

也可工作在外部曝光模式，即通过 Ext_exp 输入管脚，控制曝光时间和帧率。Ext_exp 的上升沿表征曝光开始，下降沿表征曝光结束。曝光结束后立即输出图像。每个 Ext_exp 脉冲触发一帧成像；如图 3 - 14 所示。

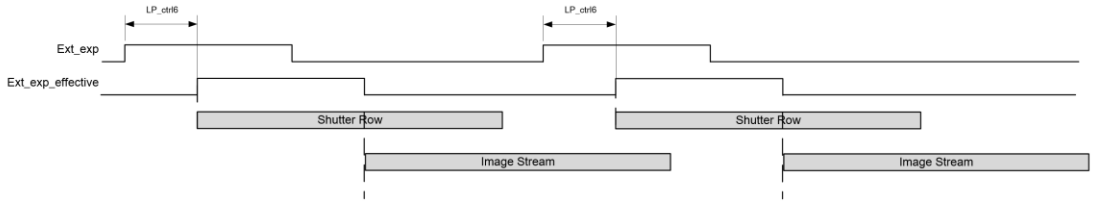


图 3 - 14 外部曝光示意图

外部曝光模式相关寄存器如表 3 - 6 所示；

表 3 - 6 外部曝光相关寄存器

Address	Register name	Default	R/W	Description
0x3D0B	ext_exp_filter	0x01	R/W	Bit[1:0]: ext_exp_filter
0x3D0C	ext_exp_offset	0x00	R/W	Bit[7:0]: ext_exp_offset
0x3D0D	r_ctrl0d	0x00	R/W	Bit[5:4]: exter_exp_mode Bit[0]: exter_exp_en
0x320a	LP_ctrl6	0x04	R/W	LP_ctrl6[15:8]
0x320b	LP_ctrl6	0x50	R/W	LP_ctrl6[7:0]

外部曝光设置方式：

1）设置 0x3D0D=0x01， 开启 Ext_exp_en；

2）设置 0x3D0B 设置 Ext_exp 的滤波等级

0x3D0B==0x00: 2 eclk

0x3D0B==0x01: 4 eclk

0x3D0B==0x02: 8 eclk

0x3D0B==0x03: 16 eclk

3）设置 LP_ctrl6，即 Ext_exp 到芯片内部生效的 Ext_exp_effective 之间的延迟；

4）设置 ext_exp_offset (0x3D0C)：

实际曝光时间 = Ext_exp 宽度 – ext_exp_offset

实际对应代码示例：

```
i2c_write(0x3D0D, 0x01); //开启 ext_exp_en
i2c_write(0x3D0B, 0x00); //设置 EVSYNC 作为外部曝光触发信号时的输入滤波等级为 2 eclk
i2c_write(0x3D0C, 0x00); //设置 ext_exp_offset=0
```

3.7.2. 外部同步

外部同步有两种工作模式：

Master Mode： FSYNC 作为输出。在帧头之后输出一个 FSYNC 脉冲，用做同步支持外部同步的 Slave CIS。 可微调功能如下：

1）输出 FSYNC 极性可调；

2）输出 FSYNC 脉冲的距离帧头的时间可调。（以行长为单位）；

3）输出 FSYNC 脉冲宽度可调（以 LL1001 row_clk 为单位，default 48MHz）；

4）每间隔 N 帧输出一个 FSYNC 脉冲，N 可调；

5）Debug 模式：FSYNC 输出内部 VSYNC，HSYNC；

Slave Mode：FSYNC 作为输入，在接收到有效的 FSYNC 之后，同步数据流输出；功能限制，以及可微调功能如下：

限制：帧时间（$S_{Frmtime}$）应尽量接近 FSYNC 的周期（ $M_{Frmtime}$ ），并且 $M_{Frmtime} >= S_{Frmtime}$ ；仅在内部曝光模式下生效。

1）输入 FSYNC 极性可调；

2）输入 FSYNC 可识别的最小宽度可调（防毛刺）；

3）FSYNC 同步位置可调；（以行长为单位）

4）主从帧时间误差 delta = $M_{Frmtime}$ - $S_{Frmtime}$ ，所能容忍的 delta 值可设置。Delta 值超过阈值的帧会被当做首次同步帧；

5）首次同步帧可选删除，且删除帧数可选；

6）作为从机，上电并初始化后，在首次接收到 FSYNC 之前，可选是否输出图像；

7）支持 Preshutter 模式： 若设置首次同步前不输出图像，则可进一步设置首次接收到的 FSYNC 为 Shutter 起始；

外部同步相关寄存器如表 3 - 7 所示：

表 3 - 7 外部同步相关寄存器

Address	Register name	Default	R/W	Description
0x3010	r_ctrl10	0x00	R/W	Bit[1]: STROBE_oe Bit[0]: FSYNC_oe
0x3D00	r_ctrl3d00	0x00	R/W	Bit[7:4]: FSYNC_out_sel Bit[0]: FSYNC_M_polarity
0x3D01	r_ctrl3d01	0x01	R/W	Bit[4]: FSYNC_S_polarity Bit[1:0]:FSYNC_S_filter
0x3D02	FSYNC_mode	0x00	R/W	Bit[1:0]: FSYNC_mode
0x3D03	FSYNC_M_width	0x0A	R/W	Bit[7:0]: FSYNC_M_width

0x3D04	FSYNC_M_interval	0x01	R/W	Bit[7:0]: FSYNC_M_interval
0x3D05	FSYNC_M_position	0x00	R/W	Bit[7:0]: FSYNC_M_position[15:8]
0x3D06	FSYNC_M_position	0x10	R/W	Bit[7:0]: FSYNC_M_position[7:0]
0x3D07	FSYNC_S_position	0x00	R/W	Bit[7:0]: FSYNC_S_position[15:8]
0x3D08	FSYNC_S_position	0x10	R/W	Bit[7:0]: FSYNC_S_position[7:0]
0x3D09	FSYNC_S_tol	0x03	R/W	Bit[7:0]: FSYNC_S_tol
0x3D0A	FSYNC_S_commc	0x21	R/W	Bit[7]: frm_out_bf_1st_sync Bit[6]: preshutter_mode_en Bit[5]: del_fsync_frm Bit[4]: single_mode_en Bit[3:0]: reserved

外部同步基本模式选择如图 3 - 15 所示：

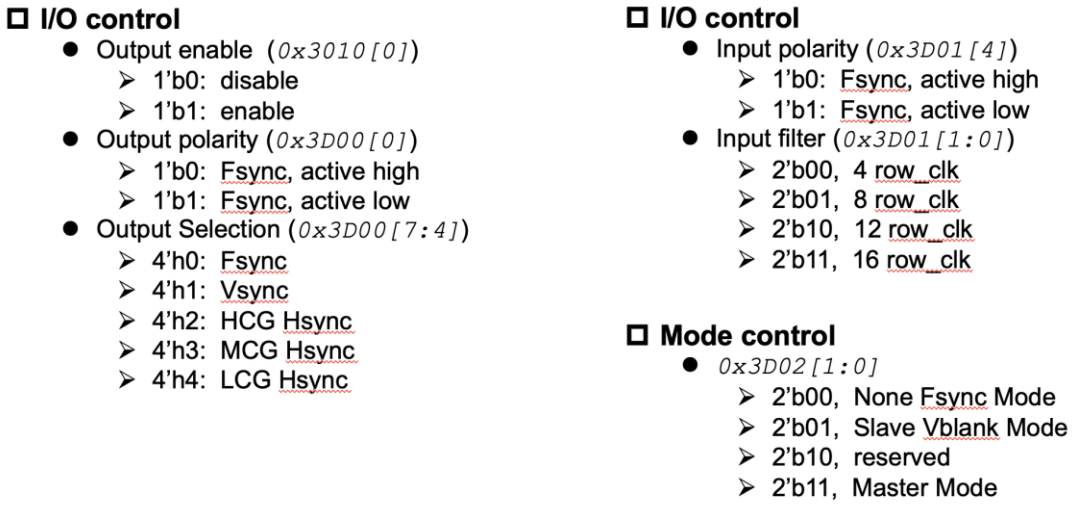


图 3 - 15 外部同步基本模式选择

外部同步 Master 模式，可以调整的设置有：

1）输出 FSYNC 的极性：FSYNC_M_polarity （0x3D00[0]）

2）输出 FSYNC 脉冲宽度：FSYNC_M_width （0x3D03[7:0]）

3）输出 FSYNC 的间隔：FSYNC_M_interval （0x3D04[7:0]）

4）输出 FSYNC 的相对位置：

FSYNC_M_position （{0x3D05[7:0],0x3D06[7:0]}）

具体设置的对应关系如图 3 - 16 所示：

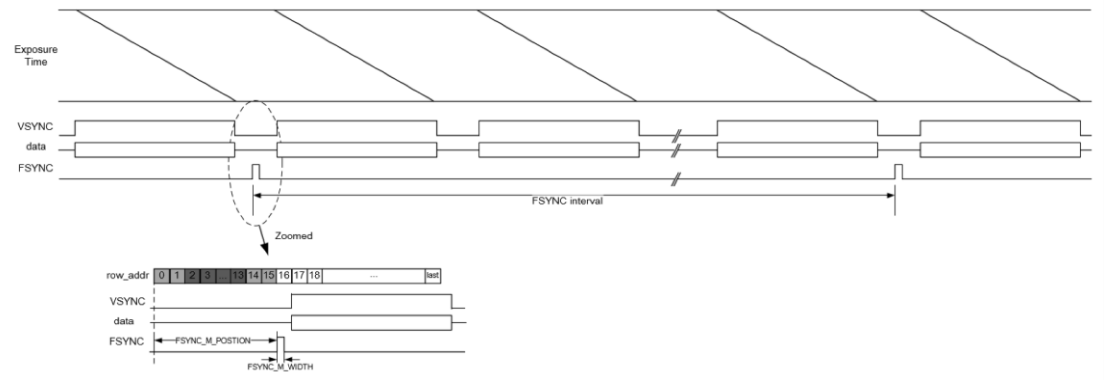


图 3 - 16 外部同步 Master 模式

外部同步 Slave 模式，可以调整的设置有：

1）输入 FSYNC 的有效极性： FSYNC_S_polarity （0x3D01[4]）

2）输入 FSYNC 滤波等级（最小脉冲宽度）：FSYNC_S_filter（0x3D01[1:0]）

0x3D01[1:0] == 2'b00: FSYNC 脉冲宽度小于 4 个 row_clk，会被忽略；

0x3D01[1:0] == 2'b01: FSYNC 脉冲宽度小于 8 个 row_clk，会被忽略；

0x3D01[1:0] == 2'b10: FSYNC 脉冲宽度小于 12 个 row_clk，会被忽略；

0x3D01[1:0] == 2'b11: FSYNC 脉冲宽度小于 16 个 row_clk，会被忽略；

5）输入 FSYNC 的相对位置：

FSYNC_S_position （{0x3D07[7:0],0x3D08[7:0]}）

6）主机和从机之间帧时间误差容忍：FSYNC_S_tol （0x3D09[7:0]）

主机从机帧时间(1/fps)误差超过 FSYNC_S_tol 会被判定为首次同步，进而触发删帧

7）行为控制：

> 0x3D0A[7]: frm_out_bf_1st_sync，

==1'b1, 接收到第一个 FSYNC 之前，不输出图像；

==1'b0, 接收到第一个 FSYNC 之前，正常输出图像；

>0x3D0A[6]: preshutter_mode_en,

==1'b0, FSYNC 正常同步

==1'b1,第一个 FSYNC 会被当做 Shutter 起始，第二个 FSYNC 开始正常同步；只在 0x3D0A[7]==1'b0 生效，且不会被判断为首次同步帧；

>0x3D0A[5]: del_fsync_frm

==1'b0,不删除首次同步帧

==1'b1,删除被判定为首次同步的帧，删除帧数由 0x3326[3:0]指定；

外部同步 Slave 模式，PreShutter 模式

1）FSYNC slave mode

0x3D02 = 0x01

2）首次接收有效 FSYNC 之前不出图

0x3D0A = 0xc0

3）首次接收到的有效 FSYNC 会被当做 Shutter 起始

4）按照 I2C 预先设置的曝光开始 Shutter

5）Master 应设置第 2 次 FSYNC 距离第 1 个 FSYNC 的时间为

TEXP+FSYNC_S_Position+2

6）自第 2 个 FSYNC 起，check FSYNC 与 FSYNC_S_position 之间的距离 delta，并调整 Vblank（+delta）。

例如图 3 - 17 所示，假设曝光时间设置为 5.5Trow，FSYNC_S_postion=3；首次 FSYNC 触发 Shutter 起始，之后会检测接收到的 FSYNC 与 FSYNC_S_POSITION 之间的差异，并在下一帧调整次差异；

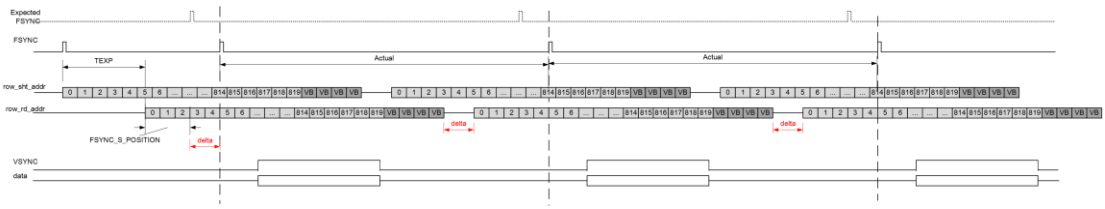


图 3 - 17 外部同步 Slave 模式（PreShutter mode）

Note：

1）为什么 VBLANK 不直接+2*delta？ —— 会导致每一帧 FSYNC 与帧头之间的距离不固定

2）RST0112R 仅会以整数行调整 VB，因此 FSYNC 距离帧头的距离会出现小于一行的抖动

3）FSYNC_S_POSITION 设置合适时，曝光时间不会出现误差；且对 delta 的容忍程度较高

外部同步 Slave 模式，Non-PreShutter 模式

1）设置为外部同步 Slave Non-PreShutter 模式

0x3D02 = 0x01;

0x3D0A[6] = 0;

2）可设置首次同步前不出图

3）自第 1 个 FSYNC 起， check FSYNC 与 FSYNC_S_position 之间的距离 delta，并调整 Vblank（+delta）

例如图 3 - 18 所示，假设 FSYNC_S_position=3；RST0112R 会检测接收到的 FSYNC 与 FSYNC_S_position 之间的差异，并在下一帧调整次差异；

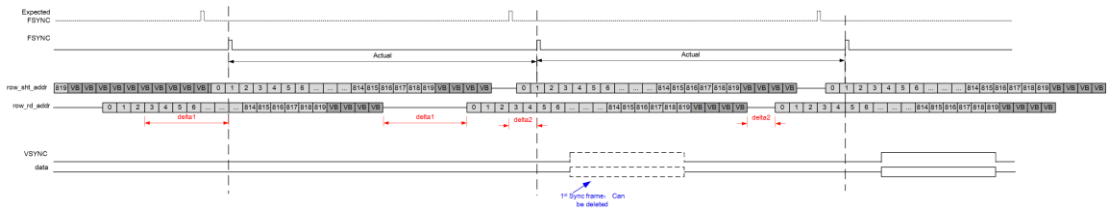


图 3 - 18 外部同步 Slave 模式（Non-PreShutter mode）

外部同步 Slave 模式下的 RST0112R 的曝光误差

在外部同步 Slave 模式下的曝光误差与 FSYNC_S_position 有关，若按图 3 - 19 所示设置 FSYNC_S_position 到推荐范围，曝光时间无误差，且容忍 delta 范围较宽；否则曝光时间的误差= delta，即 $M_{Frmtime}$ - $S_{Frmtime}$。

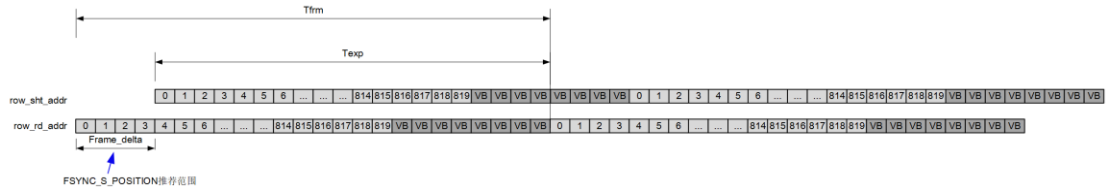


图 3 - 19 FSYNC_S_position 推荐设置范围

注意：

此处的 Frame_delta 由 0x3A1A 设置，意义为最大曝光时间与帧时间之间的关系：曝光时间 ＜ 1/fps – Frame_delta

外部同步 Slave 模式异常处理机制

首次同步之前：

如图 3 - 20 所示，可设置为首次同步前，按预先配置输出图像：

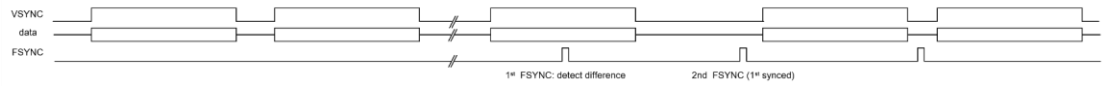


图 3 - 20 FSYNC 首次同步前正常输出图像

如图 3 - 21 所示，可设为首次同步前，不输出图像：

图 3 - 21 FSYNC 首次同步前不输出图像

首次同步时：

如图 3 - 22 所示，首次同步时，可删除首次同步之后的帧，帧数可配置；

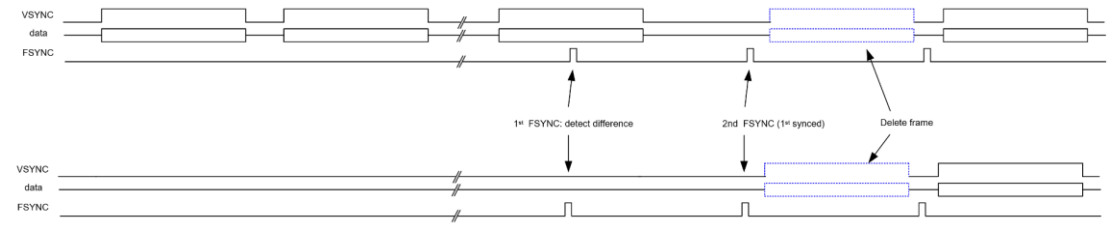


图 3 - 22 FSYNC 首次同步

同步中途，FSYNC 严重偏离期望位置（与期望位置偏差超过 FSYNC_S_tol）如图 3 - 23 所示，此时会被当做首次同步来处理：可删除首次同步之后的帧，之后正常同步；

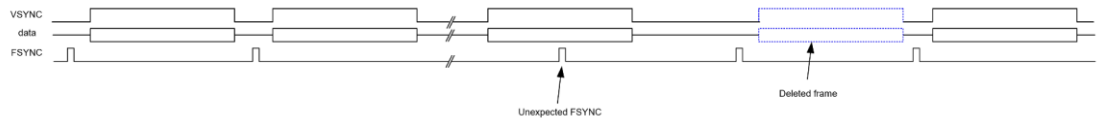


图 3 - 23 FSYNC 同步中途严重偏离期望位置

3.7.3. 闪光灯控制（strobe）

RST0112R 可通过 STROBE 管脚输出控制信号，作为闪光灯控制。与闪光灯控制相关的寄存器，如表 3 - 8 所示。

表 3 - 8 Strobe 相关寄存器

Address	Register name	Default	R/W	Description
0x3010	r_ctrl10	0x00	R/W	Bit[1]: STROBE_oe Bit[0]: FSYNC_oe
0x3D10	r_ctrl3d10		R/W	Bit[7]: Strobe_en Bit[6]: Strobe_polarity Bit[5:4]:Strobe_X_wid Bit[3]: reserved Bit[2:0] Strobe_mode
0x3D11	Strobe_dmy_lines		R/W	Bit[7:0]: Strobe_dmy_lines[15:8]
0x3D12	Strobe_dmy_lines		R/W	Bit[7:0]: Strobe_dmy_lines[7:0]
0x3D13	r_ctrl3d13		R/W	Bit[3]: Strobe_start_sel 1’b0: EndOfShutter 1’b1: EndOfStream Bit[1:0]: Strobe_frm_latency 2’b00: next frame 2’b01: 2 frames later 2’b10: 3 frames later 2’b11: 4 frames later
0x3D14	Strobe_row_latency		R/W	Bit[7:0] : Strobe_row_latency
0x3D15	Strobe_req		W	Bit[0] : Strobe_req[0]

支持 5 中闪光灯控制方式，基本可设置情况如图 3 - 24 所示：

- Strobe Output Enable: *STROBE_OE* (*0x3010[1]*)
- Strobe On: *STROBE_EN* (*0x3D10[7]*)
- Strobe polarity: *STROBE_POLARITRY* (*0x3D10[6]*),
  ==0 (default), active high; ==1, active low
- Strobe width in Xenon: *STROBE_X_WID* (*0x3D10[5:4]*)
- Strobe Mode: *STROBE_MODE* (*0x3D10[2:0]*)
  3'h0: Xenon flash Mode
  3'h1: LED1 Mode
  3'h2: LED2 Mode
  3'h3: LED3 Mode
  3'h4: LED4 Mode
  3'h5: Constant Mode

- Strobe Dummy lines: *STROBE_DMY_LINES* ({*0x3D11,0x3D12*}), 以Trow为单位
- Strobe start point sel: *STROBE_START_SEL* (*0x3D13[3]*)
  ==0 (default), endofshutter; ==1, endofstream
- Strobe frm latency: *STROBE_FRM_LATENCY* (*0x3D13[1:0]*), 以frame为单位
  2'b00: Strobe generated at next frame
  2'b01: Strobe generated 2 frames later
  2'b10: Strobe generated 3 frames later
  2'b11: Strobe generated 4 frames later

- Strobe row latency: *STROBE_ROW_LATENCY* (*0x3D14[7: 0]*), 以trow为单位
- Strobe Request: *STROBE_REQ* (*0x3D15[0]*)

图 3 - 24 LL1001 Strobe 基本设置

Xenon Flash （氙灯）模式

RST0112R 在接收到 Strobe_req（I2C 写 0x3D15=0x01→0x3D15=0x00）之后，会在第三帧（延迟帧数可通过 0x3D13[1:0]设置）输出 Strobe 脉冲。Strobe 脉冲的宽度由 0x3D10[5:4]设置，以行时间（H）为单位。如图 3 - 25 所示。

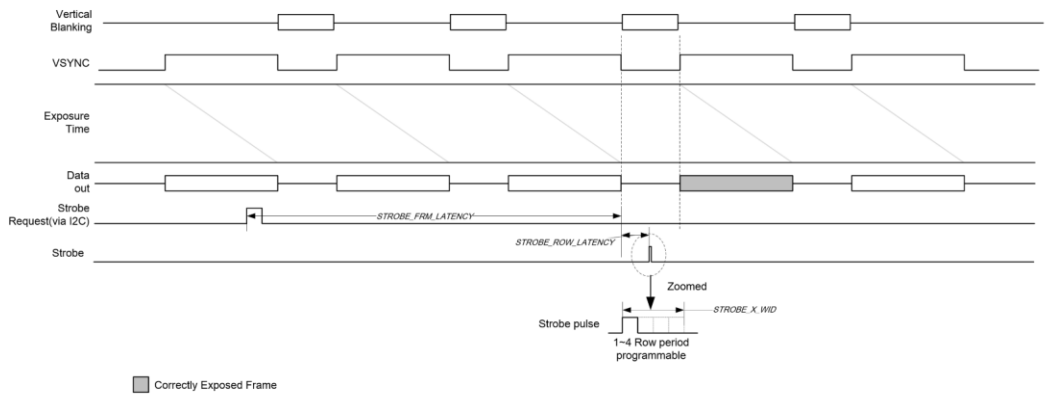


图 3 - 25 Strobe Xenon Flash（氙灯）模式

LED1 模式

在 LED1 模式下，会在 Strobe_req 上升沿一段时间之后，拉高 Strobe 管脚（极性可取反）。Strobe 管脚持续有效电平的时间由 Strobe_dmy_lines（{0x3D11,0x3D12}）设置，以行时间为单位。Strobe_dmy_lines 同时会增加曝光时间，以符合如图 3 - 26 所示要求。

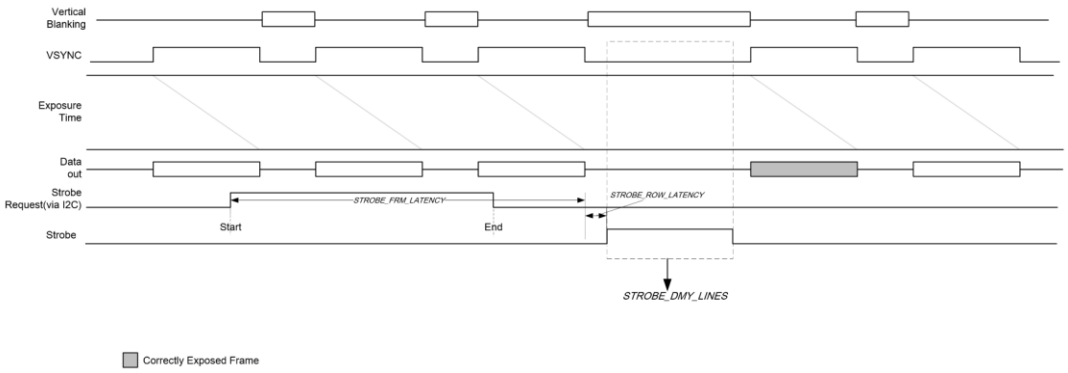


图 3 - 26 Strobe LED1 模式

LED2 模式

与 LED1 模式很类似，区别是在 LED2 模式下，如果 Strobe_req 始终未被写 0，即始终没有‘End’事件。RST0112R 会持续输出 Strobe 脉冲，如图 3 - 27 所示。

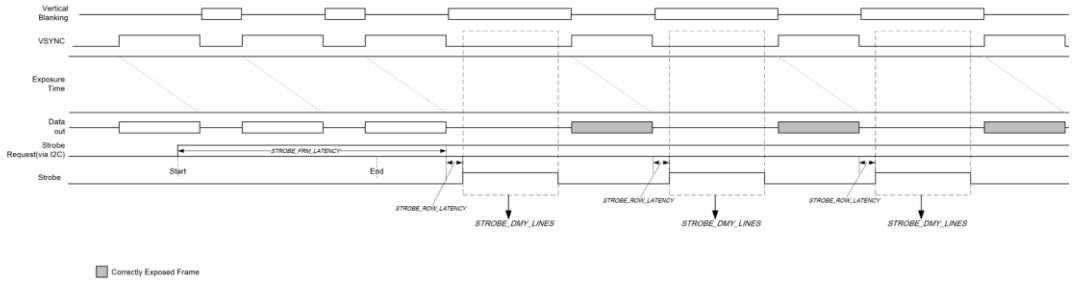


图 3 - 27 Strobe LED2 模式

LED3 模式

在 LED3 模式下，输出的 Strobe 信号会始终维持为高电平（极性可取反），直至 Strobe_req 被写 0。LED3 模式如图 3 - 28 所示。

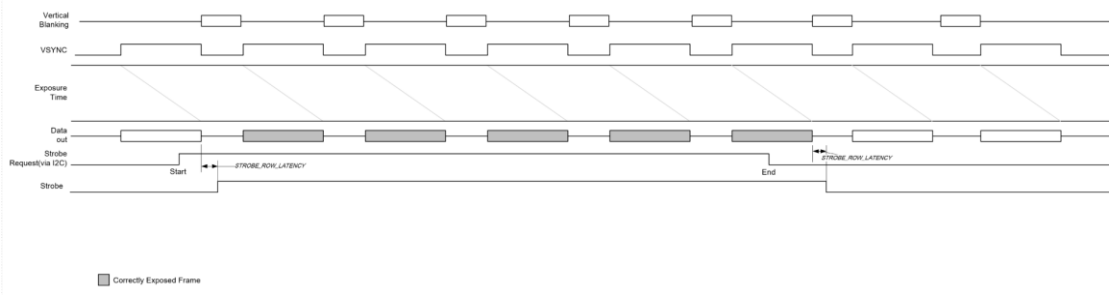


图 3 - 28 Strobe LED3 模式

LED4 模式

LED4 模式，用于与外部曝光配合，RST0112R 所有行 Shutter 完成之后，拉高 Strobe，并在 Exp_exp 管脚拉低时，拉低 Strobe。如图 3 - 29 所示。

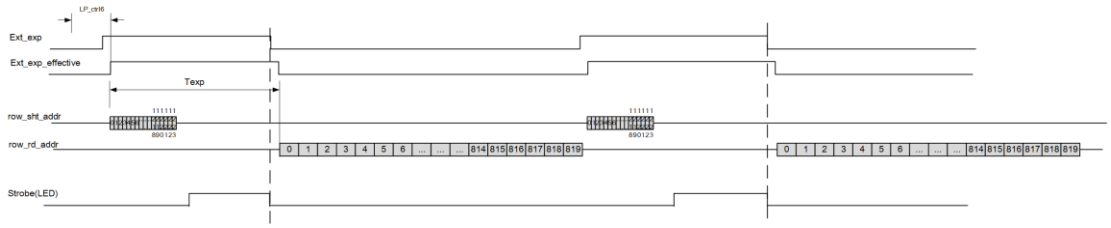


图 3 - 29 Strobe LED4 模式

Constant 模式

此模式下，通过 Strobe 管脚输出寄存器 0x3D15[0]的值。

4. 功能描述

4.1. 测试图形

传感器内置了测试图形(test pattern)，每个像素值约束如下图所示，其中测试模式可以利用寄存器 TEST_EN(寄存器地址:0x5000)写入 0x01 进入。

典型成像结果如下图。

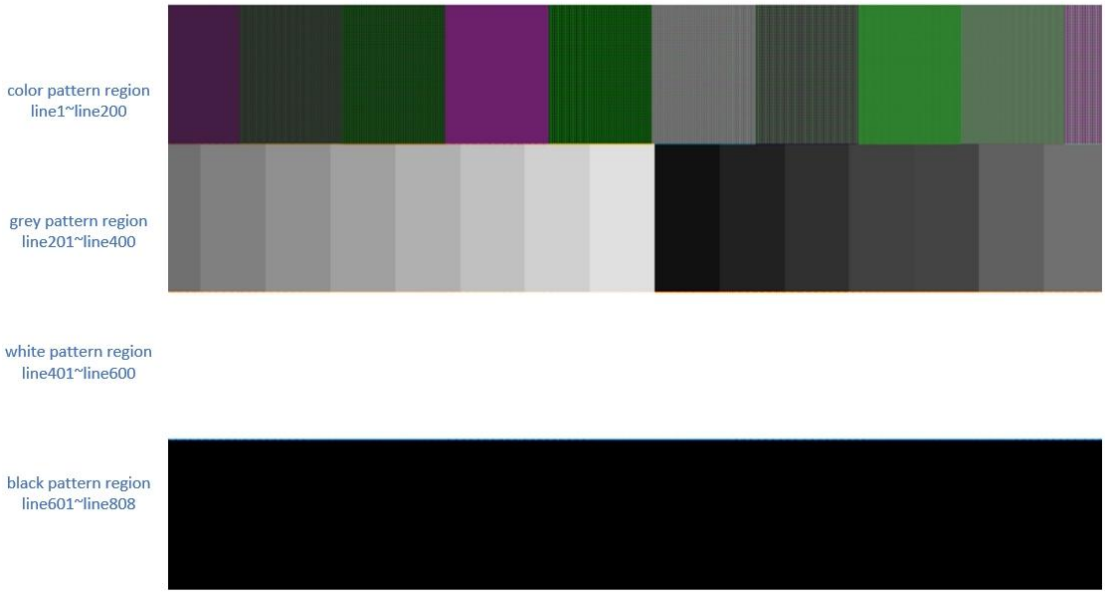


图 4 – 1 测试图形

表 4 - 1 测试图形寄存器

Address	Register name	Default	R/W	Description
0x5000	READC	0x00	R/W	Bit[0]:test_en

4.2. 内部测试信号

传感器包含一个 TEST 端口用于信号测试，通过 TEST_SEL_CODE 寄存器可以选择信号测试模式。

TEST_SEL_CODE[5:4]:

2'b00: enable voltage test input

2'b01: enable voltage test output

2'b10: enable digital test output

2'b11: enable current test output

4.2.1. 内部列电路测试

在 TEST_SEL_CODE[5:4]=2’b00 的情况下，测试尾列列电路读出。

4'd0: 外部 DAC 或电压源提供，供给尾列的 Bitline 输出节点的 sig 信号可选端。

4'd1: 外部 DAC 或电压源提供，供给尾列的 PGA 输出节点的 sig 信号可选端。

4.2.2. 内部模拟信号测试

在 TEST_SEL_CODE[5:4]=2’b01 的情况下，TEST 端口的模拟 buffer 工作，通过 TEST_SEL_CODE[3:0]选择要输出的模拟信号，具体如下。

4'd0: ramp 输出

4'd1: 最后侧列 PGA 输出

4'd2: 最右列 Bitline 输出

4'd3: Buffer_Vout_ramp_MCG

4'd4: Buffer_Vout_ramp_HCG

4'd5: Buffer_Vout_pix_MCG

4'd6: Buffer_Vout_pix_HCG

4'd7: Buffer_Vout_Vclamp_blacks

4'd8: Buffer_Vout_Vclamp_SIG

4'd9: Buffer_Vout_Vclamp_RST

4'd10: Buffer_Vout_Vclamp_PGA

4'd11: Buffer_Vout_RST_vlow

4'd12: Buffer_Vout_TCG_vlow

4'd13: Buffer_Vout_DCG_vlow

4.2.3. 内部数字信号测试

在 TEST_SEL_CODE[5:4]=2’b10 的情况下，TEST 端口的数字 buffer 工作，通过 TEST_SEL_CODE[3:0]选择要输出的数字信号，具体如下。

4'd0: MPLL 分频后的数字主时钟

4'd1: CPLL 分频后的数字主时钟

4'd2: 比较器输出，1.5V

4.2.4. 内部电流信号测试

在 TEST_SEL_CODE[5:4]=2’b11 的情况下，TEST 端口输出片内测试电流镜的电流，通过 TEST_SEL_CODE[3:0]选择要输出的电流信号，具体如下。

4'd0: comp 测试电流输出，典型情况下输出 30uA 电流，可连接 30-50k 对 AVDD 电阻测量；

4'd1: PGA 测试电流输出，典型情况下输出 30uA 电流，可连接 30-50k 对地电阻测量；

4'd2: pixbias 测试电流输出，典型情况下输出 30uA 电流，可连接 30-50k 对 AVDD 电阻测量；

4'd3: bandgap 测试电流输出，典型情况下输出 30uA 电流，可连接 30-50k 对地电阻测量；

4.3. 时钟调节

4.3.1. PLL

传感器包含一个锁相环（PLL）块，该块根据外部时钟输入生成所有必要的内部时钟。PLL 的内部功能块如下所示。

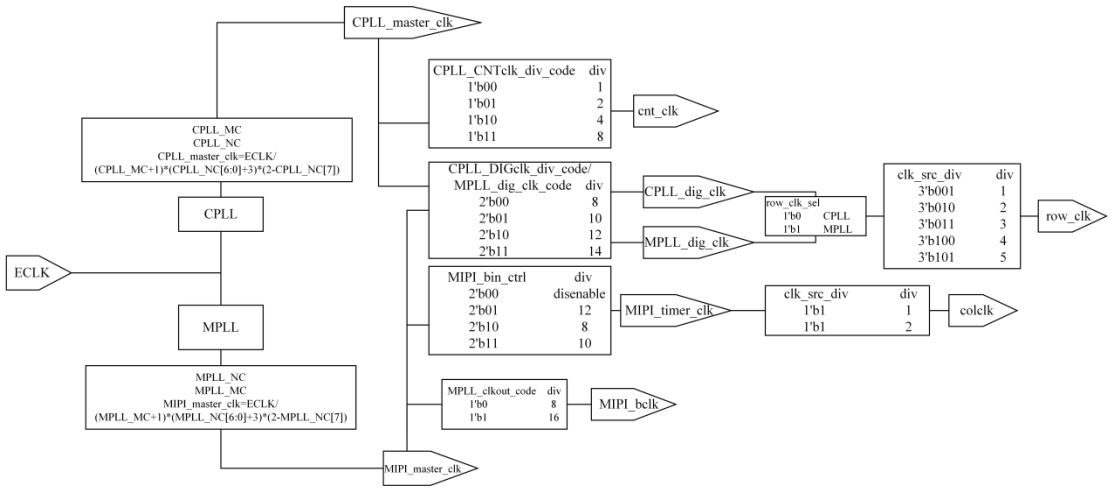


图 4 - 1 PLL 关系图

PLL 内部功能块寄存器的相关寄存器如下表.

表 4 - 2 PLL 相关寄存器

Address	Register name	Default	R/W	Description
0x0300	CPLL_CTRL00	0x07	R/W	Bit[2:0]:PLL_bias_code_15
0x0301	CPLL_CTRL01	0x00	R/W	Bit[2:0]:CPLL_MC_15
0x0302	CPLL_CTRL02	0x0a	R/W	Bit[7:0]:CPLL_NC_15
0x0303	CPLL_CTRL03	0x00	R/W	Bit[1:0]:CPLL_CNTclk_div_code_15
0x0304	CPLL_CTRL04	0x01	R/W	Bit[7:0]:CPLL_dctl_15
0x0305	CPLL_CTRL05	0x00	R/W	Bit[7:0]:CPLL_en_15
0x0306	CPLL_CTRL06	0x00	R/W	Bit[1:0]:CPLL_phase_15
0x0307	CPLL_CTRL07	0x00	R/W	Bit[1:0]:CPLL_DIGclk_div_code_15
0x0308	MPLL_CTRL00	0x00	R/W	Bit[2:0]:MPLL_bias_code_15
0x0309	MPLL_CTRL01	0x11	R/W	Bit[2:0]:MPLL_MC_15
0x030a	MPLL_CTRL02	0x07	R/W	Bit[7:0]:MPLL_NC_15
0x030b	MPLL_CTRL03	0x03	R/W	Bit[1:0]:MPLL_bias_code_15
0x030c	MPLL_CTRL04	0x00	R/W	Bit[7:0]:MPLL_bin_ctrl_15
0x030d	MPLL_CTRL05	0x00	R/W	Bit[7:0]:MPLL_en_15
0x030e	MPLL_CTRL06	0x00	R/W	Bit[1:0]:MPLL_phase_15
0x030f	MPLL_CTRL07	0x00	R/W	Bit[1:0]:MPLL_dig_clk_code_15
0x0310	PLL_CTRL00	0x00	R/W	Bit[7:5]:spi_clk_div Bit[4]:clk_src_div Bit[1:0]:dclk_div
0x0311	PLL_CTRL01	0x00	R/W	Bit[4]:row_clk_sel Bit[2:0]:row_clk_div
0x0312	PLL_CTRL02	0x44	R/W	Bit[6:4]:MPLL_pcp_clk_code_15 Bit[2:0]:MPLL_ncp_clk_code_15
0x0313	PLL_CTRL03	0x00	R/W	Bit[1:0]:MPLL_Bclk_phase_15
0x0314	PLL_CTRL04	0x00	R/W	Bit[7:0]:MPLL_clkout_code_15
0x0315	PLL_CTRL05	0x00	R/W	Bit[6]:dis_gat_bclk Bit[5]:dis_gat_dclk_hdr Bit[4]:dis_gat_dclk_otp_dpc Bit[3]:dis_gat_colclk Bit[2]:dis_gat_dclk Bit[1]:dis_gat_row_clk_sht Bit[0]:dis_gat_row_clk_rd

4.4. 帧率调节

RST0112R 采用逐行的方式执行曝光和读出，其帧长（帧率的倒数）以“行”为单位。如下图所示，帧率的控制方式有：

1）调整帧与帧之间的空行数 VBLANK 来调整帧长（推荐方式）

2）调整 RST0112R 一行的行长（非推荐方式）

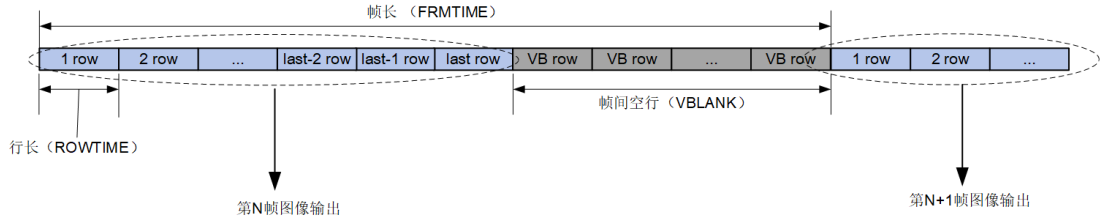


图 4 - 2 帧率示意图

注意：

1）行长 ROWTIME，可由寄存器（{0x3A0C,0x3A0D}）设置或读出；

2）行长以 mclkg 时钟周期为单位；

3）行长会影响各方面的性能表现，通常情况不推荐修改。若必须要修改，其设置值也应不小于我司所提供配置的对应设置；

4）帧长 FRMTIME 可由寄存器（{0x3A0E,0x3A0F}）读出；FRMTIME 为只读寄存器；

5）VBLANK 寄存器更改后，需要写 NewSetting（0x33fe=0x01）生效；

表 4 - 3 与帧率相关的寄存器

Address	Register name	Default	R/W	Description
0x3A10	VBLANK	0x00	R/W	Bit[7:0]:VBLANK[15:8]
0x3A11	VBLANK	0x01	R/W	Bit[7:0]:VBLANK[7:0]
0x3A0C	ROWTIME	0x02	R/W	Bit[4:0]:ROWTIME[12:8]
0x3A0D	ROWTIME	0x00	R/W	Bit[7:0]:ROWTIME[7:0] 以 mclkg 为单位的行时间
0x3A0E	FRMTIME	-	R	Bit[7:0]:FRMTIME[15:8]
0x3A0F	FRMTIME	-	R	Bit[7:0]:FRMTIME[7:0] 只读寄存器; 反应当前设置下的帧时间，以 ROWTIME 为单位:
0x33FE	NewSetting	0x01	W	在增益曝光时间等寄存器下发后向此寄存器写入 0x01，增益曝光等将在下下帧生效

0x33FD	update mode	0x00	W	0x01：自动刷新模式，无需 NewSetting 寄存器控制即可在下下帧生效增益曝光等操作; 0x00：手动刷新模式，在增益曝光时间等寄存器下发后向 0x33fe 寄存器写入 0x01，增益曝光等将在下下帧生效

4.5. 窗选

窗选通过定义 8 个参数，包括 x_addr_start、x_addr_end、y_addr_start、y_addr_end、x_output_offsize、x_output_size、y_output_offsize 和 y_output_size，其中前四个参数设置第一级模拟选窗，即曝光和量化的行列可选；后四个参数设置第二级数字选窗，在第一级的基础上再做一次数字窗选来确定最后输出的画幅尺寸，如图 4 - 3 所示。

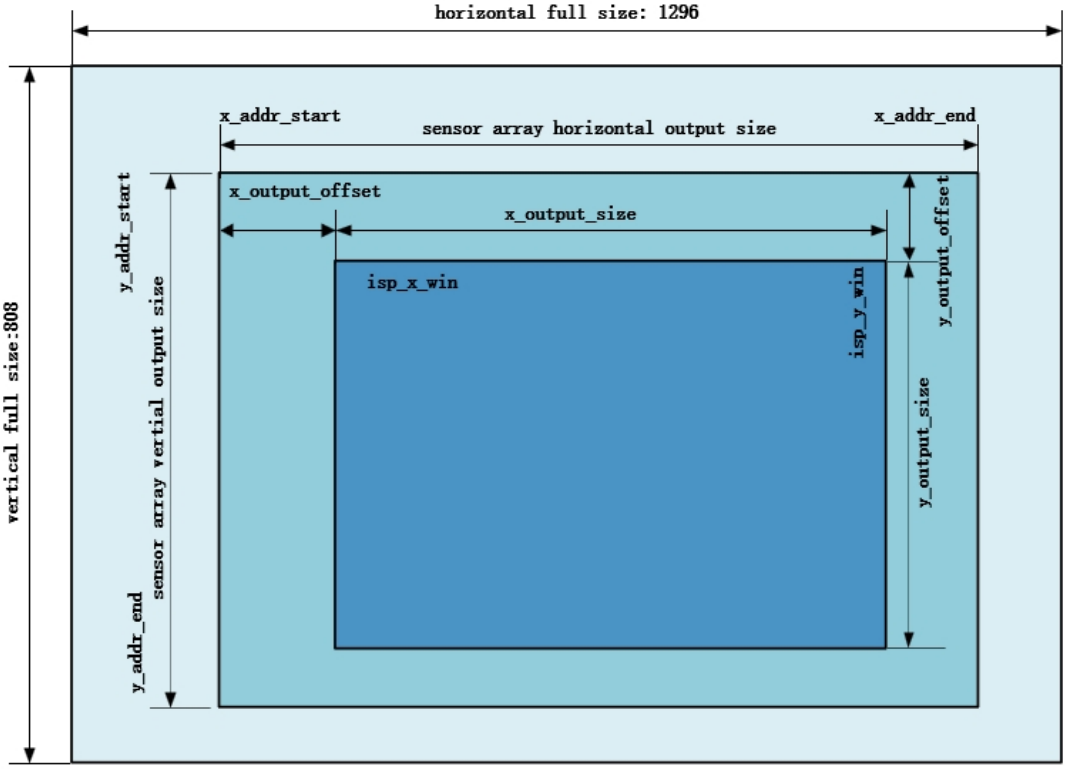


图 4 - 3 窗选示意图

通过正确设置参数，传感器内的部分阵列大小可以裁剪为可见区域，窗选功能不会冲突 FLIP 功能。

列级窗选：

一个列级码代表 4 个像素所以实际窗选范围为 0~323 对应着 0~1296 列

例：x_addr_start 写入 9’d0、x_addr_end 写入 9’d323、x_output_offset 写入 9’d2 和 x_output_size 写入 9’d320 时列级窗选为中间的 1280 列

行级窗选：

行级窗选范围为 12~819 对应 0~808 行

例：y_addr_start 写入 9’d32、y_addr_end 写入 9’d807、y_output_offset 写入 9’d4 和 y_output_size 写入 9’d768 时行级窗选为中间的 768 行

表 4 - 4 窗选寄存器表

Address	Register name	Default	R/W	Description
0x3A00	X_ADDR_START	0x00	R/W	Bit[0]:x_addr_start[8]
0x3A01	X_ADDR_START	0x00	R/W	Bit[7:0]:x_addr_start[7:0]
0x3A02	X_ADDR_END	0x00	R/W	Bit[0]:x_addr_end[8]
0x3A03	X_ADDR_END	0x01	R/W	Bit[7:0]:x_addr_end[7:0]
0x3A08	X_OUTPUT_SIZE	0x37	R/W	Bit[0]:x_output_size[8]
0x3A09	X_OUTPUT_SIZE	0x01	R/W	Bit[7:0]:x_output_size[7:0]
0x3A0A	Y_OUTPUT_SIZE	0x03	R/W	Bit[1:0]:y_output_size[9:8]
0x3A0B	Y_OUTPUT_SIZE	0x20	R/W	Bit[7:0]:y_output_size[7:0]
0x3A12	X_OUTPUT_OFFSET	0x00	R/W	Bit[0]:x_output_offset[8]
0x3A13	X_OUTPUT_OFFSET	0x02	R/W	Bit[7:0]:x_output_offset[7:0]
0x3A14	Y_OUTPUT_OFFSET	0x00	R/W	Bit[1:0]:y_output_offset[9:8]
0x3A15	Y_OUTPUT_OFFSET	0x04	R/W	Bit[7:0]:y_output_offset[7:0]

4.6. 图像翻转

RST0112R 提供镜像和翻转读出模式，可以分别水平和垂直反转传感器数据的读出顺序，如图 4 - 4 所示。

图 4 - 4 图像翻转示意图

传感器内置的图像翻转功能由 READC 寄存器(寄存器地址:0x3A1C)的值控制，主要有 UPDOWN 和 MIRROR 功能。

图像翻转寄存器使用：

1）寄存器写入 0x01：图像上下翻转（UPDOWN）

2）寄存器写入 0x02：图像左右翻转（MIRROR）

3）寄存器写入 0x03：图像同时上下翻转和左右翻转（MIRROR&UPDOWN）

表 4 - 5 图像翻转寄存器表

Address	Register name	Default	R/W	Description
0x3A1C	READC	0x00	R/W	Bit[1]:MIRROR Bit[0]:UPDOWN

传感器内置图像翻转输出首个像素色块颜色不变功能，由 FIRSTB_CTRL 寄存器（寄存器地址:0x5001）的值控制。

FIRSTB_CTRL 寄存器的使用：

1）写 0x01 时，MIRROR 时输出的首个像素色块和翻转前输出首个像素色块颜色一致

2）写 0x02 时：UPDOWN 时输出的首个像素色块和翻转前输出首个像素色块颜色一致

3）写 0x03 时：MIRROR&UPDOWN 翻转前后输出的首个像素色块颜色一致

表 4 - 6 FIRST BLUE 寄存器表

Address	Register name	Default	R/W	Description
0x5001	FIRSTB_CTRL	0x00	R/W	Bit[1]:firstB_en_row Bit[0]:firstB_en_col

4.7. 增益设置

4.7.1. 模拟增益设置

推荐设置 0x330E=0x2a，并通过 AGAIN_MCG、AGAIN_HCG、来调整模拟增益；增益更改后通过写 NewSetting（0x33fe=0x01）来生效。

其中 AGAIN_LCG 为 LCG 帧对应模拟增益，设定值默认为 0x10，即 1 倍增益，通常不做修改。

其中 AGAIN_MCG 为 MCG 帧对应模拟增益，MCG 总模拟增益：

$$RealAGain\_MCG = \frac{AGAIN\_MCG}{16}$$

其中 AGAIN_HCG 为 HCG 帧对应模拟增益，HCG 总模拟增益：

$$RealAGain\_HCG = \frac{AGAIN\_HCG}{16}$$

RST0112R 与模拟增益设置相关的寄存器如下表所示。

表 4 - 7 与模拟增益设置相关寄存器

Address	Register name	default	R/W	description
0x3314	AGAIN_MCG	0x00	R/W	Bit[12:8]:AGAIN_MCG[12:8]
0x3315	AGAIN_MCG	0x10	R/W	Bit[7:0]: AGAIN_MCG [7:0] 模拟增益=AGAIN_MCG /16, AGAIN_MCG =0x010 时，模拟增益=1x；AGAIN_MCG =0x011 时，模拟增益=17/16x，以此类推。
0x3310	AGAIN_HCG	0x00	R/W	Bit[12:8]:AGAIN_HCG[12:8]
0x3311	AGAIN_HCG	0x10	R/W	Bit[7:0]: AGAIN_HCG[7:0] 模拟增益=AGAIN_HCG/16, AGAIN_HCG=0x010 时，模拟增益=1x； AGAIN_HCG=0x011 时，模拟增益=17/16x，以此类推。
0x330E	RPC_MAP_IDX	0x00	W	Bit[5:4]:rpc_mapH_idx Bit[3:2]:rpc_mapM_idx Bit[1:0]:rpc_mapL_idx
0x330F	AGAIN_MODE	0x00	R/W	Bit[2:1]:rpc_mode, 2'b00: Again 2 frame delay 2'b01: Again 1 frame delay 2'b11: Again takes effect immediately. Bit[0]: rpc_en: 1'b1, Again set by RPC 1'b0, Again set directly.

实际对应代码示例：

```
i2c_write(0x330f, 0x00); //AGAIN 更新方式为两帧之后生效
i2c_write(0x3314, 0x01); //
i2c_write(0x3315, 0x00); //MCG 模拟增益设置为 16 倍
i2c_write(0x3310, 0x00); //
i2c_write(0x3311, 0x40); //HCG 模拟增益设置为 4 倍
i2c_write(0x33fe, 0x01); //NewSetting，以生效增益
```

RST0112R 理论最大模拟增益 62 倍，不同工作模式请咨询技术支持团队获取实际最大模拟增益以保证最优性能。

表 4 - 8 模拟增益映射表(十进制)

RealAGAIN_MCG/HCG	AGAIN_MCG/HCG	RealAGAIN_MCG/HCG	AGAIN_MCG/HCG	RealAGAIN_MCG/HCG	AGAIN_MCG/HCG
1	16	10.5	168	7	112
1.0625	17	11	176	7.25	116
1.125	18	11.5	184	7.5	120
1.1875	19	12	192	7.75	124
1.25	20	12.5	200	8	128
1.3125	21	13	208	8.5	136
1.375	22	13.5	216	9	144
1.4375	23	14	224	9.5	152
1.5	24	14.5	232	10	160
1.5625	25	1	16	10.5	168
1.625	26	1.0625	17	11	176
1.6875	27	1.125	18	11.5	184
1.75	28	1.1875	19	12	192
1.8125	29	1.25	20	12.5	200
1.875	30	1.3125	21	13	208
1.9375	31	1.375	22	13.5	216
2	32	1.4375	23	14	224
2.125	34	1.5	24	14.5	232
2.25	36	1.5625	25	15	240
2.375	38	1.625	26	15.5	248
2.5	40	1.6875	27	16	256
2.625	42	1.75	28	17	272
2.75	44	1.8125	29	18	288
2.875	46	1.875	30	19	304
3	48	1.9375	31	20	320
3.125	50	2	32	21	336
3.25	52	2.125	34	22	352
3.375	54	2.25	36	23	368
3.5	56	2.375	38	24	384
3.625	58	2.5	40	25	400
3.75	60	2.625	42	26	416
3.875	62	2.75	44	27	432
4	64	2.875	46	28	448
4.25	68	3	48	29	464
4.5	72	3.125	50	30	480
4.75	76	3.25	52	31	496
5	80	3.375	54	32	512
5.25	84	3.5	56	34	544
5.5	88	3.625	58	36	576
5.75	92	3.75	60	38	608
6	96	3.875	62	40	640

RealAGAIN_MCG/HCG	AGAIN_MCG/HCG	RealAGAIN_MCG/HCG	AGAIN_MCG/HCG	RealAGAIN_MCG/HCG	AGAIN_MCG/HCG
6.25	100	4	64	42	672
6.5	104	4.25	68	44	704
6.75	108	4.5	72	46	736
7	112	4.75	76	48	768
7.25	116	5	80	50	800
7.5	120	5.25	84	52	832
7.75	124	5.5	88	54	864
8	128	5.75	92	56	896
8.5	136	6	96	58	928
9	144	6.25	100	60	960
9.5	152	6.5	104	62	992
10	160	6.75	108		

4.7.2. 数字增益切换

HCG、MCG 和 LCG 各自有一个数字增益和 offset，当修改此增益时，需要刷新（NewSetting，0x33fe=0x01）。

Pixel = Dig_gain × Pix + offset[9:0];　//@offset[10]==0，offset 为正

Pixel = Dig_gain × Pix - offset[9:0];　//@offset[10]==1，offset 为负

数字增益相关寄存器如下表。

表 4 - 9 与模拟增益设置相关寄存器

Address	Register name	Default	R/W	Description
0x3312	DGAINH	0x00	R/W	Bit[13:8]:dGainH_pr[13:8]
0x3313	DGAINH	0x80	R/W	Bit[7:0]:dGainH_pr[7:0]
0x3316	DGAINM	0x00	R/W	Bit[13:8]:dGainM_pr[13:8]
0x3317	DGAINM	0x80	R/W	Bit[7:0]:dGainM_pr[7:0]
0x331A	DGAINL	0x00	R/W	Bit[13:8]:dGainL_pr[13:8]
0x331B	DGAINL	0x80	R/W	Bit[7:0]:dGainL_pr[7:0]
0x4023	OFFSET_CTRL0	0x00	R/W	Bit[2:0]:offset_fr0[10:8]，HCG
0x4024	OFFSET_CTRL0	0x00	R/W	Bit[7:0]:offset_fr0[7:0]，HCG
0x4025	OFFSET_CTRL1	0x00	R/W	Bit[2:0]:offset_fr1[10:8],MCG

0x4026	OFFSET_CTRL1	0x00	R/W	Bit[7:0]:offset_fr1[7:0],MCG
0x4027	OFFSET_CTRL2	0x00	R/W	Bit[2:0]:offset_fr210:8],LCG
0x4028	OFFSET_CTRL2	0x00	R/W	Bit[7:0]:offset_fr2[7:0],LCG

其中，offset 为有符号数，范围-256~255，写入时使用原码。

实际对应代码示例（HCG）：

```
i2c_write(0x3312, 0x00);
i2c_write(0x3313, 0x80); //数字增益设置为 1 倍
i2c_write(0x4023, 0x00);
i2c_write(0x4024, 0x40); //+64 offset
i2c_write(0x33fe, 0x01); //NewSetting，生效增益
```

推荐使用应用平台端的数字增益，来获得更精细化的数字增益调节，以及更方便的图像处理参数优化。

4.8. LCG/MCG/HCG 切换

LCG/MCG/HCG 的互相切换主要由寄存器 READ_MODE 写入切换，并且需要刷新（RESTART 寄存器写入 0x02）。

表 4 - 10 LCG/MCG/HCG 切换寄存器表

Address	Register name	Default	R/W	Description
0x330d	READ_MODE	0x00	R/W	bit[2:0]: 3’b000 HCG 3’b001 MCG 3’b010 LCG

4.9. 曝光时间

该传感器具有可变电子快门功能，可以以行为单位控制积分时间。此外，该传感器执行滚动快门操作，其中对每一行依次执行电子快门和读出操作。

4.9.1. 曝光时间小于一帧

1) shutter_rolling

曝光一行的时间计算（Row_Time）：

Row_Time =ROWTIME/row_clk

row_clk:时钟的频率，其典型值为 48Mhz。

ROWTIME:行时间定义寄存器地址。

曝光时间计算（Exposure_Time）：

Exposure_Time =TEXP * Row_Time

TEXP ：积分时间定义寄存器地址。

4.9.2. 曝光时间大于一帧

1) shutter_rolling

Shutter_rolling 曝光时间的上限值为 FRMTIME - FRMTIME_DELTA，当 TEXP 超过上限值时有两种情况：

A:TIMING_CTRL3（地址：0x3a1b）[0]=1 时，帧率不变，曝光时间为 FRMTIME - FRMTIME_DELTA；

B:TIMING_CTRL3(地址:0x3a1b)[0]=0 时，曝光时间不变，帧率变为曝光时间+ FRMTIME_DELTA；

当曝光时间大于上限值且当 TEXP 和 Row_time 达到上限值的情况下，可以通过增加 VBLANK 行来改变帧率，以此获得较长的曝光时间。

Exposure_Time = (TEXP+VBLANK)*Row_Time

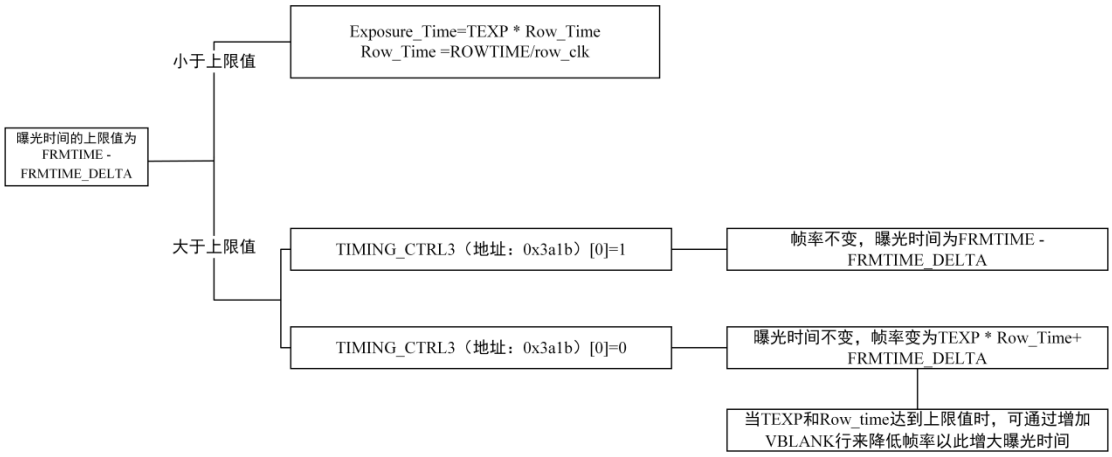


图 4 - 5 曝光时间流程

2)shutter_rolling_global

Shutter_rolling_global 曝光时间的上限值为 VBLANK - FRMTIME_DELTA，当 TEXP 超过上限值时有两种情况：

A:TIMING_CTRL3（地址：0x3a1b）[0]=1 时，帧率不变，曝光时间为 VBLANK - FRMTIME_DELTA；

B:TIMING_CTRL3(地址:0x3a1b)[0]=0 时，曝光时间不变，帧率变为曝光时间+ FRMTIME+FRMTIME_DELTA；

FRMTIME=(y_addr_end-y_addr_start)+25

表 4 - 11 曝光时间相关寄存器

Address	Register name	Default	R/W	Description
0x3301	TEXP	0x00	R/W	Bit[15:8]:texp[15:8]
0x3301	TEXP	0x00	R/W	Bit[15:8]:texp[15:8]
0x3302	TEXP	0x01	R/W	Bit[7:0]:texp[7:0]
0x3303	TEXP_CLK	0x00	R/W	Bit[15:8]:texp_clk[15:8]
0x3304	TEXP_CLK	0x00	R/W	Bit[7:0]:texp_clk[7:0]
0x3A0C	ROWTIME	0x02	R/W	Bit[4:0]:rowtime[12:8]
0x3A0D	ROWTIME	0x00	R/W	Bit[7:0]:rowtime[7:0]
0x3A10	VBLANK	0x00	R/W	Bit[7:0]:vblank[15:8]
0x3A11	VBLANK	0x01	R/W	Bit[7:0]:vblank[7:0]
0x3A1A	FRMTIME_DELTA	0x03	R/W	Bit[3:0]:frmtime_delta
0x3A1B	TIMING_CTRL3	0x00	R/W	Bit[4]:RollingGlobalEnBit[0]:fpship
0x3A23	ROWTIME_SHT	0x02	R/W	Bit[12:8]:rowtime_sht[12:8]=RollingGlobalEn?r2_ctrl23[12:8]:rowtime[12:8];
0x3A24	ROWTIME_SHT	0x00	R/W	Bit[7:0]:rowtime_sht[7:0]=RollingGlobalEn?r2_ctrl23[7:0]:rowtime[7:0];
0x3908	P3_GLOBAL	0x00	R/W	bit[0] : p3_global[8]
0x3909	P3_GLOBAL	0x05	R/W	bit[7:0]: p3_global[7:0]
0x3912	P4_GLOBAL	0x00	R/W	bit[0] : p4_global[8]

0x3913	P4_GLOBAL	0x30	R/W	bit[7:0]: p4_global[7:0]
0x3914	P5_GLOBAL	0x00	R/W	bit[0] : p5_global[8]
0x3915	P5_GLOBAL	0x05	R/W	bit[7:0]: p5_global[7:0]
0x3924	P6_GLOBAL	0x00	R/W	bit[0] : p6_global[8]
0x3925	P6_GLOBAL	0x05	R/W	bit[7:0]: p6_global[7:0]
0x393E	P7_GLOBAL	0x00	R/W	bit[0] : p7_global[8]
0x393F	P7_GLOBAL	0x60	R/W	bit[7:0]: p7_global[7:0]
0x3B40	P8_GLOABL	0x00	R/W	bit[0] : p8_global[8]
0x3B41	P8_GLOABL	0x60	R/W	bit[7:0]: p8_global[7:0]
0x3B42	P9_GLOABL	0x00	R/W	bit[0] : p9_global[8]
0x3B43	P9_GLOABL	0x05	R/W	bit[7:0]: p9_global[7:0]
0x3A04	Y_ADDR_START_REG	0x43	R/W	Bit[1:0]:y_addr_start_reg[9:8]
0x3A05	Y_ADDR_START_REG	0x00	R/W	Bit[7:0]:y_addr_start_reg[7:0]
0x3A06	Y_ADDR_end_reg	0x10	R/W	Bit[1:0]:y_addr_end_reg[9:8]
0x3A07	Y_ADDR_end_reg	0x03	R/W	Bit[7:0]:y_addr_end_reg[7:0]

4.10. 暗电平校正

RST0112R 暗电平校正功能，以确保曝光时间、曝光模式、读出模式、数据模式、数字增益或模拟增益变化时，能较为精确和稳定地补偿暗电流。后续寄存器列表中，fr0 代表 HCG，fr1 代表 MCG，fr2 代表 LCG。

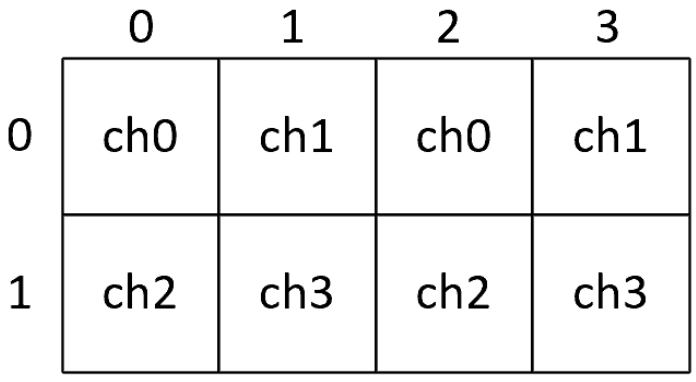


图 4 - 6 BLC 视野中的像素阵列

表 4 - 12 暗电平校正总开关

Address	Register name	Default	R/W	Description
0x4000	BLC_CTRL00	0x00	R/W	Bit[0]:blc_en

4.10.1. 暗区坏点校正

在统计 OB 值之前，先对暗区像素进行坏点校正。若目标点与其数据通道内前后各两个值极其本身共五个值的中值之差，超出阈值，则将目标点的值设为中值，否则不变。阈值可设寄存器为 BLC_MF_THRES，13bit 无符号整数。

表 4 - 13 暗区坏点校正相关寄存器列表

Address	Register name	Default	R/W	Description
0x4010	BLC_MF_THRES	0x08	R/W	Bit[4:0]:mf_abs_differ[12:8]
0x4011	BLC_MF_THRES	0x00	R/W	Bit[7:0]:mf_abs_differ[7:0]

4.10.2. OB 统计公式

OB 统计公式为：OB = K×(OB1 - OB2)+ OB2 + blc_rpc×Again

OB 为统计公式计算得到的统计值，分别统计 3 种像素 CG(HCG/MCG/LCG)下的 4 个颜色通道的 OB 值，16bit 有符号数，其中 5bit 为小数。限定其可设上限，寄存器为 BLC_DC_LIMIT，10bit 无符号整数。

OB1 为 BLC 行统计值，OB2 为 offset 行统计值，均为 21bit 有符号数，其中 7bit 为小数。

寄存器 BLC_CTRL02 控制公式中 OB2 的值：

1）BLC_CTRL02=1 时公式中 OB2 为统计得到的值；

2）BLC_CTRL02=0 时公式中 OB2 的值为 0。

K 是校正系数，可分别设置 3 种像素 CG 下（HCG/MCG/LCG）4 个颜色通道的 K 值，共 12 种，9bit 无符号数，其中 8bit 为小数。

blc_rpc 为模拟增益校正系数，可分别设置三种像素 CG 下（HCG/MCG/LCG）的值，为 8bit 有符号数，其中 6bit 为小数。

Again 则是根据模拟增益划分的档位，不同的 CG 对应不同的档位。

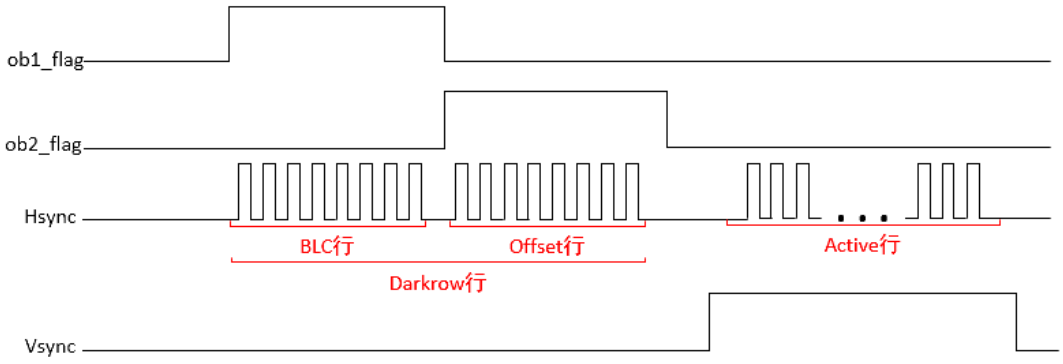


图 4 - 7 暗电平校正相关行示意图

表 4 - 14 OB 统计公式相关寄存器列表

Address	Register name	Default	R/W	Description
0x4012	BLC_CTRL02	0x00	R/W	Bit[0]:ob2_en
0x4020	BLC_RPC_HCG	0x00	R/W	Bit[7:0]:blc_rpc_fr0[7:0]
0x4021	BLC_RPC_MCG	0x00	R/W	Bit[7:0]:blc_rpc_fr1[7:0]
0x4022	BLC_RPC_LCG	0x00	R/W	Bit[7:0]:blc_rpc_fr2[7:0]
0x4029	BLC_DC_LIMIT	0x03	R/W	Bit[1:0]:blc_dc_limit[9:8]
0x402a	BLC_DC_LIMIT	0xff	R/W	Bit[7:0]:blc_dc_limit[7:0]
0x4030	BLC_K_CH0_HCG	0x01	R/W	Bit[0]:k_ch0_fr0[8]
0x4031	BLC_K_CH0_HCG	0x00	R/W	Bit[7:0]:k_ch0_fr0[7:0]
0x4032	BLC_K_CH1_HCG	0x01	R/W	Bit[0]:k_ch1_fr0[8]
0x4033	BLC_K_CH1_HCG	0x00	R/W	Bit[7:0]:k_ch1_fr0[7:0]
0x4034	BLC_K_CH2_HCG	0x01	R/W	Bit[0]:k_ch2_fr0[8]
0x4035	BLC_K_CH2_HCG	0x00	R/W	Bit[7:0]:k_ch2_fr0[7:0]
0x4036	BLC_K_CH3_HCG	0x01	R/W	Bit[0]:k_ch3_fr0[8]
0x4037	BLC_K_CH3_HCG	0x00	R/W	Bit[7:0]:k_ch3_fr0[7:0]
0x4040	BLC_K_CH0_MCG	0x01	R/W	Bit[0]:k_ch0_fr1[8]
0x4041	BLC_K_CH0_MCG	0x00	R/W	Bit[7:0]:k_ch0_fr1[7:0]
0x4042	BLC_K_CH1_MCG	0x01	R/W	Bit[0]:k_ch1_fr1[8]
0x4043	BLC_K_CH1_MCG	0x00	R/W	Bit[7:0]:k_ch1_fr1[7:0]
0x4044	BLC_K_CH2_MCG	0x01	R/W	Bit[0]:k_ch2_fr1[8]
0x4045	BLC_K_CH2_MCG	0x00	R/W	Bit[7:0]:k_ch2_fr1[7:0]
0x4046	BLC_K_CH3_MCG	0x01	R/W	Bit[0]:k_ch3_fr1[8]
0x4047	BLC_K_CH3_MCG	0x00	R/W	Bit[7:0]:k_ch3_fr1[7:0]
0x4050	BLC_K_CH0_LCG	0x01	R/W	Bit[0]:k_ch0_fr2[8]
0x4051	BLC_K_CH0_LCG	0x00	R/W	Bit[7:0]:k_ch0_fr2[7:0]
0x4052	BLC_K_CH1_LCG	0x01	R/W	Bit[0]:k_ch1_fr2[8]
0x4053	BLC_K_CH1_LCG	0x00	R/W	Bit[7:0]:k_ch1_fr2[7:0]
0x4054	BLC_K_CH2_LCG	0x01	R/W	Bit[0]:k_ch2_fr2[8]
0x4055	BLC_K_CH2_LCG	0x00	R/W	Bit[7:0]:k_ch2_fr2[7:0]
0x4056	BLC_K_CH3_LCG	0x01	R/W	Bit[0]:k_ch3_fr2[8]

0x4057	BLC_K_CH3_LCG	0x00	R/W	Bit[7:0]:k_ch3_fr2[7:0]

4.10.3. OB 统计区域

OB1 和 OB2 的统计区域相同。行列选择是独立的。

列选：默认统计输入至 BLC 模块的，然后前后各去 8 列的数据。BLC_AREA_SELECT[4]=1 时，可以选以 4 列为基本单位的连续列。

行选：暗区有 16 行，输入 BLC 模块的始终为中间八行，默认该 8 行全部统计，BLC_AREA_SELECT[0]=1 时，可选该 8 行中的任意连续行。

表 4 - 15 OB 统计区域相关寄存器列表

Address	Register name	Default	R/W	Description
0x4002	BLC_AREA_SELECT	0x00	R/W	Bit[4]:ob_hsel Bit[0]:ob_csel
0x4003	BLC_AREA_ROW_START	0x00	R/W	Bit[2:0]:ob_hstart
0x4004	BLC_AREA_ROW_END	0x07	R/W	Bit[2:0]:ob_hend
0x4005	BLC_AREA_COL_START	0x00	R/W	Bit[0]:ob_cstart[8]
0x4006	BLC_AREA_COL_START	0x00	R/W	Bit[7:0]:ob_cstart[7:0]
0x4007	BLC_AREA_COL_END	0x01	R/W	Bit[0]:ob_cend[8]
0x4008	BLC_AREA_COL_END	0x43	R/W	Bit[7:0]:ob_cend[7:0]

4.10.4. OB 迭代

多帧迭代会将统计公式得到的 OB 统计值即 $OB_{st\_current}$ 进行迭代，得到 OB 迭代值 OB_{st_new}，OB_{st_old} 为上一帧迭代结果。

有 1/4/8 帧三种迭代模式可选，BLC_CTRL03 默认为 0，1 帧迭代；为 2 时 4 帧迭代；为 3 时 8 帧迭代。

1）1 帧代表当前帧权重为 1: $OB_{st\_new}=OB_{st\_current}$

2）4 帧代表当前帧权重为 1/4：$OB_{st\_new}=OB_{st\_current}\times 0.25+OB_{st\_old}\times 0.75$

3）8 帧代表当前帧权重为 1/8：OB_{st_new} =$OB_{st\_current}$×0.125+OB_{st_old}×0.875

满足以下两个条件之一，OB_{st_new} 就会由当前帧的统计值开始重新迭代，即公式中，OB_{st_old}=0。两个条件均可由寄存器控制单独开启或者关闭，默认为 1 打开。

1）HCG/MCG/LCG 下任一颜色通道的当前帧的统计值与上一帧的迭代值的差大于设置值时进行重新迭代，即 abs($OB_{st\_current}$-OB_{st_old})>设置值（8bit 无符号整数），由 TRIGGER_SWITCH[6]控制。

2）迭代模式变更也会重新迭代，由 TRIGGER_SWITCH[7]控制。

表 4 - 16 OB 迭代相关寄存器列表

Address	Register name	Default	R/W	Description
0x4013	BLC_CTRL03	0x00	R/W	Bit[1:0]:a_ma
0x4017	TRIGGER_SWITCH	0xff	R/W	Bit[7]:abs_ob_current&old Bit[6]: a_ma change
0x4070	BLC_CTRL18	0x32	R/W	Bit[7:0]:ob_ch0_fr0_st
0x4071	BLC_CTRL19	0x32	R/W	Bit[7:0]:ob_ch1_fr0_st
0x4072	BLC_CTRL20	0x32	R/W	Bit[7:0]:ob_ch2_fr0_st
0x4073	BLC_CTRL21	0x32	R/W	Bit[7:0]:ob_ch3_fr0_st
0x4074	BLC_CTRL22	0x32	R/W	Bit[7:0]:ob_ch0_fr1_st
0x4075	BLC_CTRL23	0x32	R/W	Bit[7:0]:ob_ch1_fr1_st
0x4076	BLC_CTRL24	0x32	R/W	Bit[7:0]:ob_ch2_fr1_st
0x4077	BLC_CTRL25	0x32	R/W	Bit[7:0]:ob_ch3_fr1_st
0x4078	BLC_CTRL26	0x32	R/W	Bit[7:0]:ob_ch0_fr2_st
0x4079	BLC_CTRL27	0x32	R/W	Bit[7:0]:ob_ch1_fr2_st
0x407a	BLC_CTRL28	0x32	R/W	Bit[7:0]:ob_ch2_fr2_st
0x407b	BLC_CTRL29	0x32	R/W	Bit[7:0]:ob_ch3_fr2_st

4.10.5. OB 更新

满足 OB 值更新的触发条件时会采用新的迭代值作为有效值（OB_{st_new}➡OB_{eff}）。延迟几帧更新和额外连续更新几帧可分别由寄存器 OB_FRAME_DLY 和寄存器 OB_FRAME_SUC 控制。

Freeze 功能：由寄存器 BLC_CTRL04 控制，为 1 时开启，会冻结当前有效值。

持续更新功能：由寄存器 BLC_CTRL05 控制，为 1 时开启，开启时无需触发条件一直更新，同时也可控制延迟几帧生效。

触发条件如下（各个条件可单独由寄存器控制，默认为 1，打开）：

1）数字增益变化：

3 种 global_gain 和 3 种 mwb_gain 任一变化，由 TRIGGER_SWITCH[1]控制。

2）模拟增益变化：

3 种像素 CG 下任一模拟增益变化，由 TRIGGER_SWITCH[4]控制。

3）读出数据模式变化

由 TRIGGER_SWITCH[2]控制。

4）数据模式变化

由 TRIGGER_SWITCH[3]控制。

5）曝光时间变化或曝光模式变化

由 TRIGGER_SWITCH[5]控制。

6）通道 mean 值变化

3 种像素 CG 下任一颜色通道的 abs(OB_{st_new}-OB_{eff})>设置值（8bit 无符号整数），由 TRIGGER_SWITCH[0]控制。

表 4 - 17 OB 更新有关寄存器列表

Address	Register name	Default	R/W	Description
0x4014	BLC_CTRL04	0x00	R/W	Bit[0]:freeze
0x4015	BLC_CTRL05	0x00	R/W	Bit[0]:ob_en
0x4017	TRIGGER_SWITCH	0xff	R/W	Bit[5]:exp_change Bit[4]:rpc_change Bit[3]:data_mode_change Bit[2]:read_mode_change Bit[1]:dig_gain_change Bit[0]:abs_ob_new_eff_change
0x4018	OB_FRAME_DLY	0x00	R/W	Bit[7:0]:ob_frame_dly
0x4019	OB_FRAME_SUC	0x00	R/W	Bit[7:0]:ob_frame_suc
0x4060	BLC_CTRL06	0x32	R/W	Bit[7:0]:ob_ch0_fr0_ef
0x4061	BLC_CTRL07	0x32	R/W	Bit[7:0]:ob_ch1_fr0_ef
0x4062	BLC_CTRL08	0x32	R/W	Bit[7:0]:ob_ch2_fr0_ef
0x4063	BLC_CTRL09	0x32	R/W	Bit[7:0]:ob_ch3_fr0_ef
0x4064	BLC_CTRL10	0x32	R/W	Bit[7:0]:ob_ch0_fr1_ef
0x4065	BLC_CTRL11	0x32	R/W	Bit[7:0]:ob_ch1_fr1_ef
0x4066	BLC_CTRL12	0x32	R/W	Bit[7:0]:ob_ch2_fr1_ef
0x4067	BLC_CTRL13	0x32	R/W	Bit[7:0]:ob_ch3_fr1_ef
0x4068	BLC_CTRL14	0x32	R/W	Bit[7:0]:ob_ch0_fr2_ef
0x4069	BLC_CTRL15	0x32	R/W	Bit[7:0]:ob_ch1_fr2_ef
0x406a	BLC_CTRL16	0x32	R/W	Bit[7:0]:ob_ch2_fr2_ef
0x406b	BLC_CTRL17	0x32	R/W	Bit[7:0]:ob_ch3_fr2_ef

4.10.6. 有效区域像素校正

有效区域像素 pixel 的校正输出公式为:

$$Dout=Dig\_gain\times(Din-OB_{use}+random1)+offset+random2$$

Din 为有效区域输入像素值，14bit 有符号整数

Dout 为校正后的输出值，会根据数据模式钳位到最大位宽，同时负数会被修正为 0 或者 1，由寄存器 DATA_LOW_LIMIT 控制，默认为 0。

Dig_gain 是校正增益乘以 global_gain 再乘以 mwb_gain 而来，最终保留为 20Bit 有符号正数，其中 10bit 为小数。

1） 两种校正增益为 gain_correctH 和 gain_correctM。对应 HCG 和 MCG，为 8bit 无符号数，其中 7bit 为小数。

2） 3 种 global_gain 分别对 3 种像素 CG，14bit 无符号数，其中 7bit 为小数。

3） 4 种 mwb_gain 对应 3 种颜色通道，8bit 无符号数，其中 7bit 为小数。

offset 为补偿项，不同 CG 对应不同的 offset，11bit 有符号数。offset[10]=0 时＋offset[9:0]，offset[10]=1 时-offset[9:0]。

random1 和 random2 为随机小数项，均为 7bit 小数。random 功能可由寄存器 RANDOM_SELECT 控制，默认为 0 关闭，为 1 时 random1 单独开启，为 2 时，random2 单独开启。

OB_{use}为最终应用的值，BLC_CTRL00 默认为 0，此时 OB_{use}=0。BLC_CTRL00 为 1 时，如果 BLC_CTRL01 为 0，OB_{use}=OB_{eff}；如果 BLC_CTRL01 为 1，OB_{use}为所在像素 CG 下的 4 个颜色通道 OB_{eff}的平均值

表 4 - 18 像素校正相关寄存器列表

Address	Register name	Default	R/W	Description
0x4000	BLC_CTRL00	0x00	R/W	Bit[0]:blc_en
0x4001	BLC_CTRL01	0x00	R/W	Bit[4]:fixed_color
0x4016	RANDOM_SELECT	0x00	R/W	Bit[1:0]:rand_en
0x401a	DATA_LOW_LIMIT	0x00	R/W	Bit[0]:data_low_limit
0x4023	OFFSET_CTRL0	0x00	R/W	Bit[2:0]:offset_fr0[10:8]
0x4024	OFFSET_CTRL0	0x00	R/W	Bit[7:0]:offset_fr0[7:0]
0x4025	OFFSET_CTRL1	0x00	R/W	Bit[2:0]:offset_fr1[10:8]
0x4026	OFFSET_CTRL1	0x00	R/W	Bit[7:0]:offset_fr1[7:0]
0x4027	OFFSET_CTRL2	0x00	R/W	Bit[2:0]:offset_fr210:8]

0x4028	OFFSET_CTRL2	0x00	R/W	Bit[7:0]:offset_fr2[7:0]

4.10.7. OB 值读取

3 种像素 CG 下的 4 种颜色通道的 OB_{use} 可以被直接读取，同时可以通过不同配置读取不同种类的 OB 值

1）想读取 OB1，在默认配置基础上，令 BLC_CTRL00 为 1，BLC_CTRL05 为 1 即可。

2）想读取 OB2，在默认配置基础上，令 BLC_CTRL00 为 1，BLC_CTRAL02 为 1，BLC_CTRL05 为 1，且令 K 值为 0 即可。

3）想读取 $OB_{st\_current}$，仅需保证 BLC_CTRL00 为 1，BLC_CTRAL01 为 0，BLC_CTRAL03 为 0，BLC_CTRL05 为 1。

4）想读取 OB_{st_new}，仅需保证 BLC_CTRL00 为 1，BLC_CTRAL01 为 0，BLC_CTRL05 为 1。

想读取 OB_{eff}，仅需保证 BLC_CTRL00 为 1，BLC_CTRAL01 为 0。

表 4 - 19 OB 值读取相关寄存器

Address	Register name	Default	R/W	Description
0x4000	BLC_CTRL00	0x00	R/W	Bit[0]:blc_en
0x4001	BLC_CTRL01	0x00	R/W	Bit[4]:fixed_color
0x4012	BLC_CTRL02	0x00	R/W	Bit[0]:ob2_en
0x4013	BLC_CTRL03	0x00	R/W	Bit[1:0]:a_ma
0x4015	BLC_CTRL05	0x00	R/W	Bit[0]:ob_en
0x4030	BLC_K_CH0_HCG	0x01	R/W	Bit[0]:k_ch0_fr0[8]
0x4031	BLC_K_CH0_HCG	0x00	R/W	Bit[7:0]:k_ch0_fr0[7:0]
0x4032	BLC_K_CH1_HCG	0x01	R/W	Bit[0]:k_ch1_fr0[8]
0x4033	BLC_K_CH1_HCG	0x00	R/W	Bit[7:0]:k_ch1_fr0[7:0]
0x4034	BLC_K_CH2_HCG	0x01	R/W	Bit[0]:k_ch2_fr0[8]
0x4035	BLC_K_CH2_HCG	0x00	R/W	Bit[7:0]:k_ch2_fr0[7:0]
0x4036	BLC_K_CH3_HCG	0x01	R/W	Bit[0]:k_ch3_fr0[8]
0x4037	BLC_K_CH3_HCG	0x00	R/W	Bit[7:0]:k_ch3_fr0[7:0]
0x4040	BLC_K_CH0_MCG	0x01	R/W	Bit[0]:k_ch0_fr1[8]
0x4041	BLC_K_CH0_MCG	0x00	R/W	Bit[7:0]:k_ch0_fr1[7:0]
0x4042	BLC_K_CH1_MCG	0x01	R/W	Bit[0]:k_ch1_fr1[8]
0x4043	BLC_K_CH1_MCG	0x00	R/W	Bit[7:0]:k_ch1_fr1[7:0]
0x4044	BLC_K_CH2_MCG	0x01	R/W	Bit[0]:k_ch2_fr1[8]
0x4045	BLC_K_CH2_MCG	0x00	R/W	Bit[7:0]:k_ch2_fr1[7:0]
0x4046	BLC_K_CH3_MCG	0x01	R/W	Bit[0]:k_ch3_fr1[8]

0x4047	BLC_K_CH3_MCG	0x00	R/W	Bit[7:0]:k_ch3_fr1[7:0]
0x4050	BLC_K_CH0_LCG	0x01	R/W	Bit[0]:k_ch0_fr2[8]
0x4051	BLC_K_CH0_LCG	0x00	R/W	Bit[7:0]:k_ch0_fr2[7:0]
0x4052	BLC_K_CH1_LCG	0x01	R/W	Bit[0]:k_ch1_fr2[8]
0x4053	BLC_K_CH1_LCG	0x00	R/W	Bit[7:0]:k_ch1_fr2[7:0]
0x4054	BLC_K_CH2_LCG	0x01	R/W	Bit[0]:k_ch2_fr2[8]
0x4055	BLC_K_CH2_LCG	0x00	R/W	Bit[7:0]:k_ch2_fr2[7:0]
0x4056	BLC_K_CH3_LCG	0x01	R/W	Bit[0]:k_ch3_fr2[8]
0x4057	BLC_K_CH3_LCG	0x00	R/W	Bit[7:0]:k_ch3_fr2[7:0]

4.10.8. BLC 与 Darkrow 输出

Darkrow 输出时需要将寄存器 TIMING_CTRL1[7]和 TIMING_CTRL1[6]写入 1，并且关闭 BLC 功能，令 BLC_CTRL00 为 0。

当选择 Darkrow 输出时，必须关闭 BLC 功能。

表 4 - 20 Darkrow 输出相关寄存器列表

Address	Register name	Default	R/W	Description
0x4000	BLC_CTRL00	0x00	R/W	Bit[0]:blc_en
0x3327	TIMING_CTRL1	0x24	R/W	Bit[7]:darkrow_mode Bit[6]:darkrow_oe

4.11. OTP

RST0112R 提供最多 256 bytes 一次性编程的存储空间（One-Time Programmable memory，后文简称 OTP），用于存储芯片生产信息和坏像素信息。内部集成一个专用 256 bytes 的缓冲区，用于暂时存储即将写入 OTP 的信息，或者存储从 OTP 中读出的信息。如图 4 - 8 所示这个 256 bytes 的专用缓存，对应的地址空间为 0x6000~0x60FF。

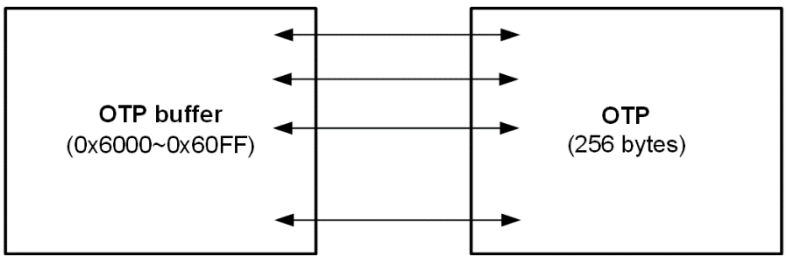


图 4 - 8 256bytes OTP 专用缓存地址空间

OTP 烧写、读出支持 4 种模式：单 byte 烧写，单 byte 读出，批量烧写、批量读出。

4.11.1. OTP 控制模块相关寄存器

表 4 - 21 OTP 控制模块相关寄存器

Address	Register Name	Default	R/W	Description
0x3E00	ADDR_PGM	0x00	R/W	Bit[7:0]: Single Program address(0x00~0xff)
0x3E01	DATA_PGM	0x00	R/W	Bit[7:0]: Single Program data
0x3E02	ADDR_READ	0x00	R/W	Bit[7:0]: Single Load(Read) address(0x00~0xff)
0x3E03	OTP_CTRL	0x00	R/W	Bit[2]: otp_dm Bit[1]: start_read Bit[0]: start_pgm
0x3E04	OTP_P0	0x01	R/W	Bit[7:0]: otp_p0
0x3E05	OTP_P1	0x02	R/W	Bit[7:0]: otp_p1
0x3E06	OTP_P2	0x13	R/W	Bit[4:0]: otp_p2[12:8]
0x3E07	OTP_P2	0xB0	R/W	Bit[7:0]: otp_p2[7:8]
0x3E08	OTP_P3	0x02	R/W	Bit[7:0]: otp_p3
0x3E09	OTP_P4	0x02	R/W	Bit[7:0]: otp_p4
0x3E0A	OTP_P5	0x0C	R/W	Bit[7:0]: otp_p5
0x3E0B	OTP_P6	0x02	R/W	Bit[7:0]: otp_p6
0x3E0C	OTP_BUSY	0x00	R	Bit[0]: OTP is busy
0x3E0D	DATA_READ	0x00	R	Bit[7:0] : Single Load(Read) data
0x3E10	OTP_CTRL_BATCH	0x00	R/W	Bit[2]: batch_pgm_bist Bit[1]: start_batch_read Bit[0]: start_batch_pgm
0x3E11	OTP_BATCH_START	0x00	R/W	Bit[7:0]: otp batch program/load start address(0x00~0xff)
0x3E12	OTP_BATCH_END	0xff	R/W	Bit[7:0]: otp batch program/load end address(0x00~0xff)
0x3E13	OTP_DPC_START	0x10	R/W	Bit[7:0]: otp dpc cluster info start address(0x00~0xff)
0x3E14	OTP_DPC_END	0xff	R/W	Bit[7:0]: otp dpc cluster info end address(0x00~0xff)
0x3E15	OTP_BATCH_GAP	0x10	R/W	Bit[7:0]: otp batch program/load gap distance
0x3E16	OTP_RATIO_ADDR	0x0C	R/W	Bit[7:0]: CG ratio content start addr
0x3E17	OTP_HML_RATIO_REG_EN	0x00	R/W	Bit[4]: otp_HML_ratio_reg_en Bit[0]: otp_HL_ratio_reg[16]
0x3E18	OTP_HL_RATIO_REG	0x00	R/W	Bit[7:0] : otp_HL_ratio_reg[15:8]
0x3E19	OTP_HL_RATIO_REG	0x00	R/W	Bit[7:0] : otp_HL_ratio_reg[7:0]
0x3E1A	OTP_HM_RATIO_REG	0x00	R/W	Bit[3:0] : otp_HM_ratio_reg[11:8]

0x3E1B	OTP_HM_RATIO_REG	0x00	R/W	Bit[7:0] : otp_HM_ratio_reg[7:0]

4.11.2. OTP 单 byte 烧写

OTP 的烧写有特定的时序要求，该时序由内置 OTP 控制器自动产生。典型的烧写时序由图 4 - 9 所示，时序的调整寄存器由表 4 - 22 所示；

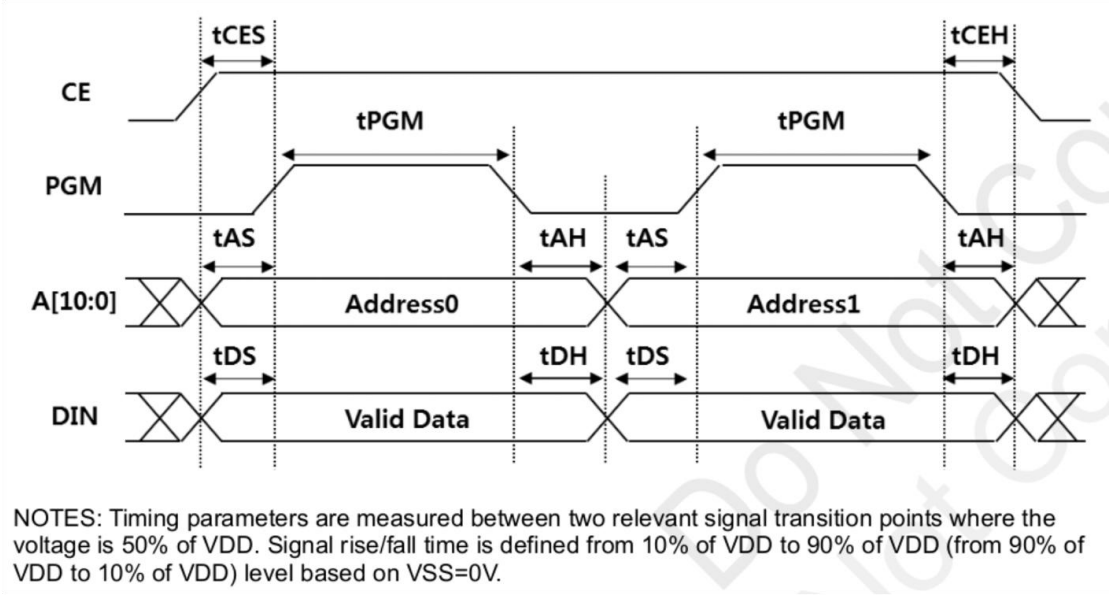


图 4 - 9 OTP 烧录时序

表 4 - 22 OTP 烧录时序调整寄存器

AC Parameter	Symbol	Spec Min	Spec typ	Spec Max	Reg	Reg unit	Reg time
Program Time	tPGM	200 us	210 us	220 us	otp_p2 (0x3E06,0x3E07)	eclk	210 us
CE Setup Time	tCES	50 ns	-	-	otp_p1 (0x3E05)	eclk	83 ns
CE Hold Time	tCEH	50 ns	-	-	otp_p1 (0x3E05)	eclk	83 ns
Address Setup Time	tAS	50 ns	-	-	otp_p1 (0x3E05)	eclk	83 ns
Address Hold Time	tAH	50 ns	-	-	otp_p3 (0x3E08)	eclk	83 ns
Data Setup Time	tDS	50 ns	-	-	otp_p1 (0x3E05)	eclk	83 ns

单 byte 烧写步骤：

1）确定 OTP_BUSY （寄存器地址为 0x3E0C）==0x00 (OTP not busy)

2）向寄存器 ADDR_PGM（寄存器地址为：0x3E00）写入需要写入数据的 OTP 地址；向寄存器 DATA_PGM（寄存器地址为：0x3E01）写入需要写入 OTP 的数据

3）向寄存器 OTP_CTRL（寄存器地址为：0x3E03）写入 0x01

4）向寄存器 OTP_CTRL（寄存器地址为：0x3E03）写入 0x00

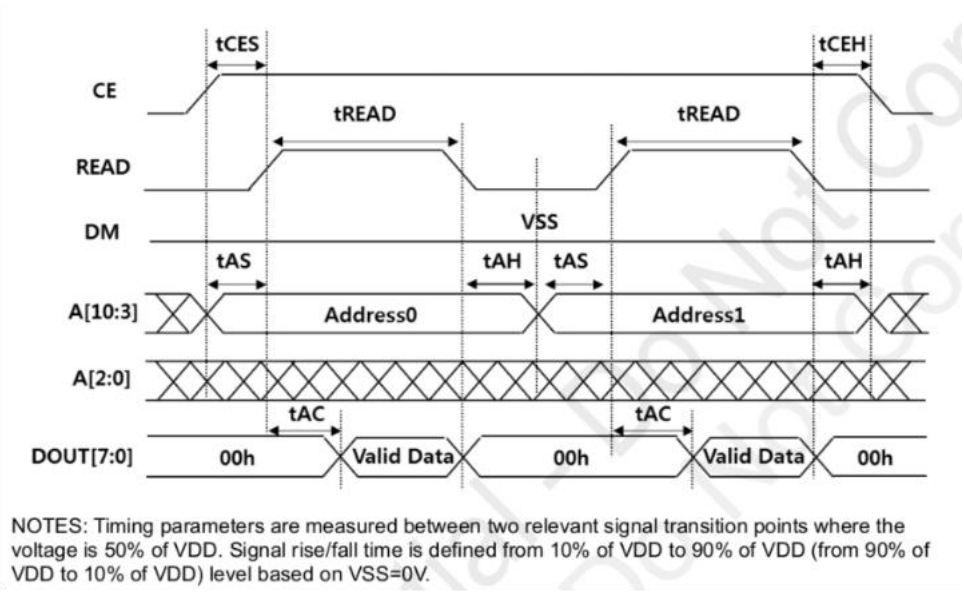
5）读 OTP_BUSY==0x00 (OTP not busy)时写入完成，可以开始下一次写入或读出

单 byte 烧写示例： (往 OTP 地址 0x41 中写入 0xAA)

```
i2c_read(0x3E0C);          //Read OTP BUSY
i2c_write(0x3E00, 0x41);  //Address of OTP
i2c_write(0x3E01, 0xAA);  //data to be programmed
i2c_write(0x3E03, 0x01);  //program enable
;delay 2ms
i2c_write(0x3E03, 0x00);  //program disable
```

4.11.3. OTP 单 byte 读出

OTP 的读出（load）有特定的时序要求，该时序由内置 OTP 控制器自动产生。典型的读出时序由图 4 - 10 所示，时序的调整寄存器由表 4 - 23 所示；



NOTES: Timing parameters are measured between two relevant signal transition points where the voltage is 50% of VDD. Signal rise/fall time is defined from 10% of VDD to 90% of VDD (from 90% of VDD to 10% of VDD) level based on VSS=0V.

图 4 - 10 OTP 读出时序

表 4 - 23 OTP 读出时序调整寄存器

AC Parameter	Symbol	Spec Min	Spec typ	Spec Max	Reg	Reg unit	Reg time
Read Time	tREAD	440 ns	450 ns	-	otp_p5 (0x3E0A)	eclk	750 ns
Access Time	tAC	-	-	400 ns	-	eclk	-

Read Setup Time	tCES	50 ns	-	-	otp_p4 (0x3E09)	eclk	83 ns
Read Hold Time	tCEH	50 ns	-	-	otp_p4 (0x3E09)	eclk	83 ns
Address Setup Time	tAS	50 ns	-	-	otp_p4 (0x3E09)	eclk	83 ns

单 byte 读出步骤：

1） 确定 OTP_BUSY （寄存器地址为 0x3E0C）==0x00 (OTP not busy)

2） 向寄存器 ADDR_READ（寄存器地址为：0x3E02）写入需要读出数据的 OTP 地址

3） 向寄存器 OTP_CTRL（寄存器地址为：0x3E03）写入 0x02

4） 向寄存器 OTP_CTRL（寄存器地址为：0x3E03）写入 0x00

5） 在 OTP_BUSY==0(OTP not busy) 时读出完成，读出寄存器 DOUT_OTP_CONTROLLER（寄存器地址为：0x3E0D）

单 byte 读出示例：(从 OTP 地址 0x42 中读出先前烧录的信息)

```
i2c_read(0x3E0C);          //Read OTP BUSY
i2c_write(0x3E02, 0x42);  //Address of OTP
i2c_write(0x3E03, 0x02);  //Load enable
;delay 2ms
i2c_write(0x3E03, 0x00);  //program disable
i2c_read(0x3E0D);          //
```

4.11.4. OTP 批量 byte 烧写（batch program）

除单 byte 烧写之外，RST0112R 支持通过 OTP 缓存，进行多 byte 批量烧写。

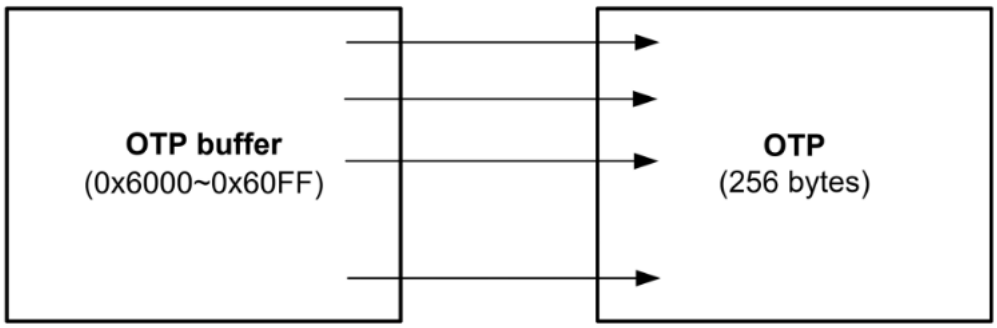


图 4 - 11 OTP 批量烧写

批量多 byte 烧写步骤：

1） 确定 OTP_BUSY （寄存器地址为 0x3E0C）==0x00 (OTP not busy)

2） 写入 OTP 批量烧写/读出的起始地址

3） 写入 OTP 批量烧写/读出的终止地址

4） 向 OTP 缓存对应地址中写入想要烧录的内容

5） 开启批量烧录

6） 关闭批量烧录

批量多 byte 烧写示例：（向 OTP 地址范围 0x10~ 0xCF 中烧录）

```
i2c_read(0x3E0C);          //Read OTP BUSY
i2c_write(0x3E11, 0x10);  //Start address of OTP
i2c_write(0x3E12, 0xCF);  //End address of OTP
i2c_write(0x6010, 0xA3);  //OTP buffer data
i2c_write(0x6011, 0x30);  //OTP buffer data
i2c_write(0x6012, 0x99);  //OTP buffer data
…
i2c_write(0x60CF, 0x11);  //OTP buffer data
i2c_write(0x3E10, 0x01);  //batch program enable
i2c_write(0x3E10, 0x00);  //batch program disable
;delay 450ms               //Delay for programming
```

4.11.5. OTP 批量 byte 读出（batch load）

除单 byte 读出之外，RST0112R 支持通过 OTP 缓存批量读出 OTP 中所有内容，且无需指定批量读出的地址范围。

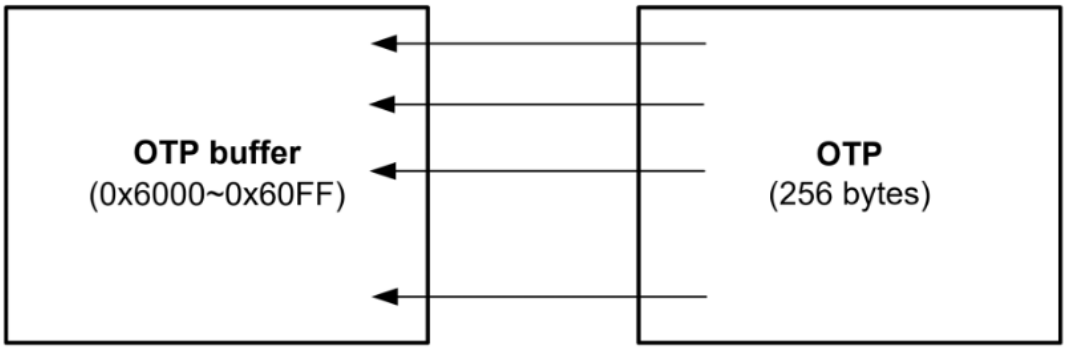


图 4 - 12 OTP 批量读出

批量多 byte 读出步骤：

1） 确定 OTP_BUSY （寄存器地址为 0x3E0C）==0x00 (OTP not busy)

2） 开启批量读出使能

3） 关闭批量读出使能

4） 使用 OTP 缓存地址读出 OTP 内容

批量多 byte 读出示例：

```
i2c_read(0x3E0C);          //Read OTP BUSY
i2c_write(0x3E10, 0x02);  //batch read enable
i2c_write(0x3E10, 0x00);  //batch read disable
;delay 10ms
i2c_read(0x6010);          //Read from OTP buffer
i2c_read(0x6011);          //Read from OTP buffer
i2c_read(0x6012);          //Read from OTP buffer
…
i2c_read(0x60CF);          //Read from OTP buffer
```

4.11.6. 带自测试的 OTP 批量 byte 烧写（batch program with BIST）

当开启 OTP BIST 功能时，OTP 多 byte 批量烧写之后，RT0112R 会自动读出 OTP 中内容，并与 OTP 缓存做对比，以判断 OTP 是否被烧录成功；

批量多 byte 烧写步骤：

1） 确定 OTP_BUSY （寄存器地址为 0x3E0C）==0x00 (OTP not busy)

2） 写入 OTP 批量烧写的起始地址

3） 写入 OTP 批量烧写的终止地址

4） 向 OTP 缓存对应地址中写入想要烧录的内容

5） 开启批量烧录，同时开启 BIST 使能

批量多 byte 烧写（BIST on）示例：（向 OTP 地址范围 0x10~ 0xCF 中烧录）

```
i2c_read(0x3E0C);          //Read OTP BUSY
i2c_write(0x3E11, 0x10);  //Start address of OTP
i2c_write(0x3E12, 0xCF);  //End address of OTP
i2c_write(0x6010, 0xA3);  //OTP buffer data
i2c_write(0x6011, 0x30);  //OTP buffer data
```

```
i2c_write(0x6012, 0x99);  //OTP buffer data
…
i2c_write(0x60CF, 0x11);  //OTP buffer data
i2c_write(0x3E10, 0x05);  //batch program enable
i2c_write(0x3E10, 0x00);  //batch program disable
;delay 450ms               //Delay for programming
i2c_read(0x3E10);          //Read BIST result
```

```
Note!
0x3E10[4]:  BIST done
0x3E10[5]:  BIST Error flag
```

上例中，如果读 0x3E10 返回值为 0x15，则表示 OTP 烧录成功，且 BIST 完成，无错误；如果读 0x3E10 返回值为 0x35, 则表示 OTP 烧录成功，且 BIST 完成，有错误；若返回值 0x3E10[4]==0，则表示 BIST 进行中，未完成；

4.12. OTP_DPC

RST0112R 内置坏像素消除模块，坏像素的信息在 CP 阶段存储于 OTP 中，在开启 DPC 功能时，会自动消除 OTP 中存储的坏像素。

RST0112R 可消除的类型如图 4 - 13 所示，可以为单坏像素，最多支持 4 x 4 像素。

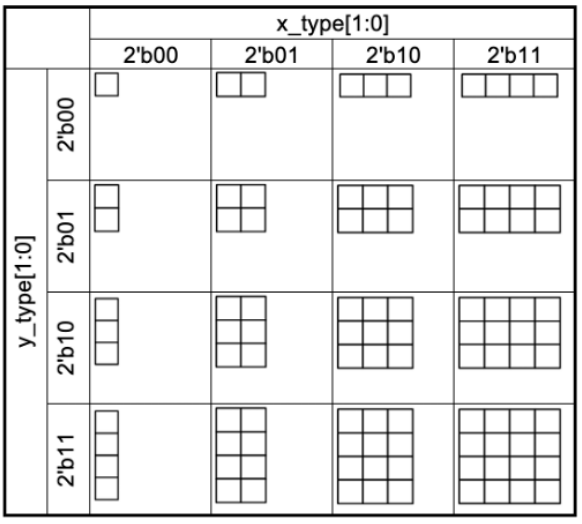


图 4 - 13 可消除的坏点类型

byte0							
b7	b6	b5	b4	b3	b2	b1	b0
-	x_type[1]	x_type[0]	y_type[1]	y_type[0]	x_org[10]	x_org[9]	x_org[8]
byte1							
b7	b6	b5	b4	b3	b2	b1	b0
x_org[1]	x_org[1]	x_org[1]	x_org[1]	x_org[1]	x_org[1]	x_org[1]	x_org[0]
byte2							
b7	b6	b5	b4	b3	b2	b1	b0
y_org[9]	y_org[8]	y_org[7]	y_org[6]	y_org[5]	y_org[4]	y_org[3]	y_org[2]
byte3							
b7	b6	b5	b4	b3	b2	b1	b0
y_org[1]	y_org[0]	-	-	-	-	-	-

图 4 - 14 坏点信息存储方式

一个坏点信息，需要 4bytes 的 OTP 存储空间，如图 4 - 14 所示。其中

1） x_type[1:0], y_type[1:0]用于指定坏点的 x（列）和 y（行）方向的尺寸；

2） x_org[10:0], y_org[9:0]分别指定坏点左上角像素的绝对 x（列）、y（行）坐标；绝对坐标是指在 Normal 模式下（no Flip，no Mirror），该像素在包括 darkrow 在内的、可寻址的所有像素阵列中的坐标。坐标以 0 为起始点。

4.12.1. OTP_DPC 模块相关寄存器

表 4 - 24 OTP_DPC 模块相关寄存器

Address	Register name	default	R/W	description
0x5100	otp_dpc_en	0x00	R/W	Bit[2]: otp_dpc_en_LCG Bit[1]: otp_dpc_en_MCG Bit[0]: otp_dpc_en_HCG
0x5104	r_ctrl04	0x00	R/W	Bit[4]: man_inc_en Bit[3]: disable_mf, disable mirror/flip Bit[2]: disable_offset, ==1, offset set by register; ==0, offset set automatically. Bit[1]: mirror_otp Bit[0]: disable_bin

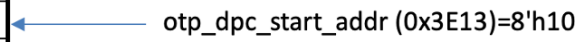
0x5105	r_ctrl05	0x68	R/W	Bit[7]: sub_bin_method_en Bit[6:5]: recov_method Bit[4]: fixed_replace Bit[3]: fixed_ptn Bit[2]: flip_otp Bit[1]: expo_en Bit[0]: gain_en
0x5106	expo_constraint_HCG	0x00	R/W	Bit[6:0]: expo_constraint_HCG[14:8]
0x5107	expo_constraint_HCG	0x00	R/W	Bit[7:0]: expo_constraint_HCG[7:0]
0x5108	gain_constraint_HCG	0x07	R/W	Bit[7:0]: gain_constraint_HCG[7:0]
0x5109	r_ctrl09	0x48	R/W	Bit[7]: xy_end_sel Bit[6]: vsync_rst_en Bit[4]: thre_en Bit[3:0]: thre_HCG
0x510A	man_x_even_inc	0x01	R/W	Bit[4:0]: man_x_even_inc
0x510B	man_x_odd_inc	0x01	R/W	Bit[4:0]: man_x_odd_inc
0x510C	man_y_even_inc	0x01	R/W	Bit[4:0]: man_y_even_inc
0x510D	man_y_odd_inc	0x01	R/W	Bit[4:0]: man_y_odd_inc
0x5110	x_offset_eff	0x00	R	Bit[7:0] x_offset_eff[IH_SW-1:8]
0x5111	x_offset_eff	0x00	R	Bit[7:0] x_offset_eff[7:0]
0x5112	y_offset_eff	0x00	R	Bit[7:0] y_offset_eff[IV_SW-1:8]
0x5113	y_offset_eff	0x00	R	Bit[7:0] y_offset_eff[7:0]
0x5114	man_x_even_inc_eff	0x00	R	Bit[4:0]: man_x_even_inc_eff
0x5115	man_x_odd_inc_eff	0x00	R	Bit[4:0]: man_x_odd_inc_eff
0x5116	man_y_even_inc_eff	0x00	R	Bit[4:0]: man_y_even_inc_eff
0x5117	man_y_odd_inc_eff	0x00	R	Bit[4:0]: man_y_odd_inc_eff
0x5120	man_x_offset	0x00	R/W	Bit[4:0]: man_x_offset[12:8]
0x5121	man_x_offset	0x00	R/W	Bit[7:0]: man_x_offset[7:0]
0x5122	man_y_offset	0x00	R/W	Bit[4:0]: man_y_offset[12:8]
0x5123	man_y_offset	0x00	R/W	Bit[7:0]: man_y_offset[7:0]
0x5126	r_ctrl26	0x00	R/W	Bit[7]: y_bin_man_en Bit[6]: y_bin4_man Bit[5]: y_bin3_man Bit[4]: y_bin2_man Bit[3]: x_bin_man_en Bit[2]: x_bin4_man Bit[1]: x_bin3_man Bit[0]: x_bin2_man
0x5128	expo_constraint_MCG	0x00	R/W	Bit[6:0]: expo_constraint_MCG[14:8]
0x5129	expo_constraint_MCG	0x00	R/W	Bit[7:0]: expo_constraint_MCG[7:0]

0x512A	gain_constraint_MCG	0x07	R/W	Bit[7:0]: gain_constraint_MCG[7:0]
0x512B	thre_MCG	0x08	R/W	Bit[3:0]: thre_MCG[3:0]
0x512C	expo_constraint_LCG	0x00	R/W	Bit[6:0]: expo_constraint_LCG[14:8]
0x512D	expo_constraint_LCG	0x00	R/W	Bit[7:0]: expo_constraint_LCG[7:0]
0x512E	gain_constraint_LCG	0x07	R/W	Bit[7:0]: gain_constraint_LCG[7:0]
0x512F	thre_LCG	0x08	R/W	Bit[3:0]: thre_LCG[3:0]

4.12.2. OTP_DPC 模块使用说明

坏点信息存储于 OTP 的空间地址范围，由寄存器 0x3E13，0x3E14 指定，默认范围 0x10~0xff，共计 240bytes，即支持最多 60 个坏点信息；需要注意的是，0x3E13 指定了第一个坏点信息的 1st byte，0x3E14 指定了最后一个坏点信息的最 last byte，如图所示：

otp_addr	I2C addr	Note
0x00	0x6000	CP_info_byte0
0x01	0x6001	CP_info_byte1
…		
0x0E	0x600E	CP_info_bytee
0x0F	0x600F	CP_info_bytef
0x10	0x6010	cluster_byte0
0x11	0x6011	cluster_byte1
0x12	0x6012	cluster_byte2
0x13	0x6013	cluster_byte3
…		
0xFC	0x60FC	cluster_byte0
0xFD	0x60FD	cluster_byte1
0xFE	0x60FE	cluster_byte2
0xFF	0x60FF	cluster_byte3



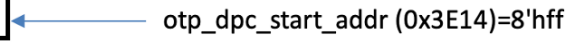


图 4 - 15 坏点信息的存储方式

OTP DPC 典型使用步骤：

1） CP 阶段，获取需要消除的坏点信息（x_type, y_type, x_org, y_org），按照图 4 - 14 的方式，先写入 OTP 缓存（0x6000~0x60FF）；

2） 写 0x3E13，0x3E14，分别指定第一个坏点信息的 1st byte，以及最后一个坏点信息的 last byte；

3） 写 0x5100，开启对应帧的坏点消除；

4） 根据消除效果，确认第 1）步中的写入 OTP 缓存的信息正确无误后，批量烧录 OTP 缓存数据到 OTP 中；

注意：

1） 模块算法限制，同一行中最多两个坏点！

2） OTP 中坏点的存储顺序，以 y_org 递增的方式存储

4.13. HDR_combine&PWL

RST0112R 一次曝光可输出高、中、低三帧不同亮度的图像，即 HCG、MCG、LCG。三帧图像的曝光同时发生，因此三帧为同一时刻不同亮度的图像。可通过 MIPI/LVDS 输出三帧图像（按行交替输出），同时内部集成了 HDR_combine 模块，也可选择输出 DCG 图像（H，M 两帧合并）， 或者 TCG 图像（H、M、L 三帧合并图像）。

4.13.1. HDR_combine 模块相关寄存器

表 4 - 25 HDR_combine 模型相关寄存器

Address	Register name	Default	R/W	Description
0x5180	HDR_r_ctrl00	0x00	R/W	Bit[4]:HM_correction_en Bit[1]:TCG_combine_en Bit[0]:DCG_combine_en
0x5181	HM_expect	0x00	R/W	Bit[3:0]:HM_expect[11:8]
0x5182	HM_expect	0x00	R/W	Bit[7:0]:HM_expect[7:0]
0x5183	HDR_MBIST_en	0x00	R/W	Bit[0]:HDR_MBIST_en
0x5184	TCG_shift	0x0B	R/W	Bit[4:0]:TCG_shift[4:0]
0x5185	thres_H_DCG	0xF0	R/W	Bit[7:0]:thres_H_DCG[7:0]
0x5186	thres_L_DCG	0x60	R/W	Bit[7:0]:thres_L_DCG[7:0]
0x5187	thres_H_TCG	0xBB	R/W	Bit[7:0]:thres_H_TCG[7:0]
0x5188	thres_L_TCG	0x4B	R/W	Bit[7:0]:thres_L_TCG[7:0]
0x5189	shift1	0x00	R/W	Bit[4:0]:shift1[4:0]
0x518A	shift2	0x03	R/W	Bit[4:0]:shift2[4:0]
0x518B	shift3	0x06	R/W	Bit[4:0]:shift3[4:0]
0x518C	shift4	0x09	R/W	Bit[4:0]:shift4[4:0]
0x518D	shift5	0x0C	R/W	Bit[4:0]:shift5[4:0]
0x518E	shift6	0x0E	R/W	Bit[4:0]:shift6[4:0]
0x518F	offset2	0x02	R/W	Bit[7:0]:offset2[15:8]
0x5190	offset2	0x56	R/W	Bit[7:0]:offset2[7:0]
0x5191	offset3	0x04	R/W	Bit[7:0]:offset3[15:8]
0x5192	offset3	0xF7	R/W	Bit[7:0]:offset3[7:0]
0x5193	offset4	0x07	R/W	Bit[7:0]:offset4[15:8]
0x5194	offset4	0xA2	R/W	Bit[7:0]:offset4[7:0]
0x5195	offset5	0x0A	R/W	Bit[7:0]:offset5[15:8]

0x5196	offset5	0x4E	R/W	Bit[7:0]:offset5[7:0]
0x5197	offset6	0x0C	R/W	Bit[7:0]:offset6[15:8]
0x5198	offset6	0x98	R/W	Bit[7:0]:offset6[7:0]
0x5199	thres1	0x00	R/W	Bit[7:0]:thres1[23:16]
0x519A	thres1	0x02	R/W	Bit[7:0]:thres1[15:8]
0x519B	thres1	0xAC	R/W	Bit[7:0]:thres1[7:0]
0x519C	thres2	0x00	R/W	Bit[7:0]:thres2[23:16]
0x519D	thres2	0x18	R/W	Bit[7:0]:thres2[15:8]
0x519E	thres2	0x0C	R/W	Bit[7:0]:thres2[7:0]
0x519F	thres3	0x00	R/W	Bit[7:0]:thres3[23:16]
0x51A0	thres3	0xC3	R/W	Bit[7:0]:thres3[15:8]
0x51A1	thres3	0x0C	R/W	Bit[7:0]:thres3[7:0]
0x51A2	thres4	0x06	R/W	Bit[7:0]:thres4[23:16]
0x51A3	thres4	0x1B	R/W	Bit[7:0]:thres4[15:8]
0x51A4	thres4	0x0C	R/W	Bit[7:0]:thres4[7:0]
0x51A5	thres5	0x30	R/W	Bit[7:0]:thres5[23:16]
0x51A6	thres5	0xDB	R/W	Bit[7:0]:thres5[15:8]
0x51A7	thres5	0x0C	R/W	Bit[7:0]:thres5[7:0]
0x51A8	offset_bl	0x00	R/W	Bit[6:0]:offset_bl[22:16]
0x51A9	offset_bl	0x00	R/W	Bit[7:0]:offset_bl[15:8]
0x51AA	offset_bl	0x00	R/W	Bit[7:0]:offset_bl[7:0]
0x51B0	PWL_mode	0x02	R/W	Bit[3:0]:PWL_mode[3:0]
0x51B1	thres_L_clamp	0xFF	R/W	Bit[7:0]:thres_L_clamp

4.13.2. HCG/MCG，HCG/LCG ratio storage

像素在 HCG 模式和 MCG 模式下的 CG 之比 otp_HM_ratio[11:0]，以及像素在 HCG 模式和 LCG 模式下的 CG 之比 otp_HL_ratio[15:0]，需要在 CP 阶段测量并计算后存入 OTP。

otp_HM_ratio[11:0] = HCG/MCG， 其 中 otp_HM_ratio[11:6] 为 整 数 ，otp_HM_ratio[5:0]为小数；

otp_HL_ratio[17:0] = HCG/LCG， 其 中 otp_HL_ratio[17:6] 为 整 数 ，otp_HL_ratio[5:0]为小数

存储的 OTP 起始地址由寄存器 0x3E16 指定，默认为 0x0C，对应的存储方式如图 4 - 16 所示。

otp_addr	I2C addr	Note
0x00	0x6000	CP_info_byte0
0x01	0x6001	CP_info_byte1
…		
0x0C	0x600C	**{otp_HL_ratio[16],otp_HM_ratio[11:8]}**
0x0D	0x600D	**otp_HM_ratio[7:0]**
0x0E	0x600E	**otp_HL_ratio[15:8]**
0x0F	0x600F	**otp_HL_ratio[7:0]**
0x10	0x6010	cluster_byte2
0x11	0x6011	cluster_byte3
…		
0xFC	0x60FC	cluster_byte0
0xFD	0x60FD	cluster_byte1
0xFE	0x60FE	cluster_byte2
0xFF	0x60FF	cluster_byte3

otp_ratio_addr (0x3E16)=8'h0C

otp_dpc_start_addr (0x3E13)=8'h10

otp_dpc_start_addr (0x3E14)=8'hff

图 4 - 16 CG ratio in OTP address

otp_HM_ratio，otp_HL_ratio 会被用于 HDR_combine；

4.13.3. DCG combine

芯片配置成 DCG 模式——同时输出 HCG，MCG 两帧图像。此时可配置 HDR_combine 模块工作在 DCG combine 模式。合并后的图像，可选择直接输出，或通过内置 PWL 模块，缩减位宽后输出；

DCG combine 工作过程简述如下：（内部参考）

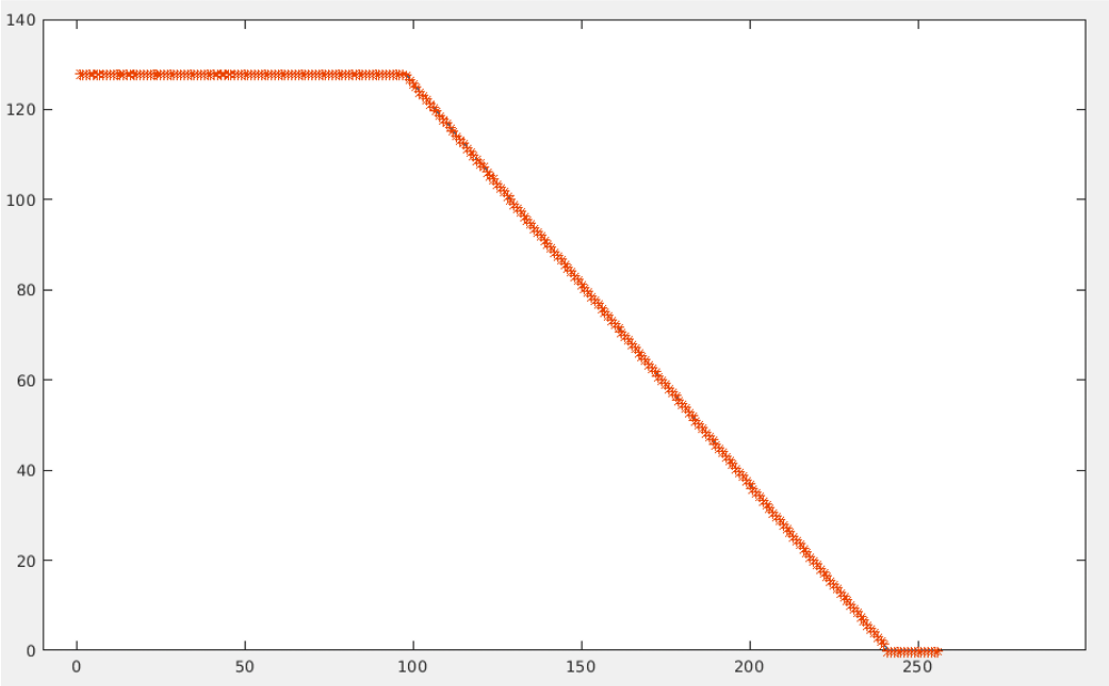
1) 关闭 BLC 中的 offset——HCG，MCG 以 0 为起点；
2) 自动计算曝光时间差异、模拟增益差异对 CG_ratio 的影响：

$$Ratio1 = \frac{TEXP \times T_{row} + TEXP_{clk} + adj_H}{TEXP \times T_{row} + TEXP_{clk} + adj_M} \times \frac{Again_H}{Again_M} \times otp\_HM\_ratio$$

3) 自动计算融合权重 Weight。取 HCG 的高 8Bit 作为 HW，以 HW 作为索引按如下方式计算 Weight，其中 Hth 由寄存器 0x5185 设置， Lth 由寄存器 0x5186 设置。

```
if(HW <= Lth)
    Weight = 128;
elseif(HW >= Hth)
    Weight = 0;
else
    Weight = (Hth-HW)*128/(Hth-Lth);
end
```

权重 Weight 曲线大致为：



4) MCG 乘以 Ratio1：$LR = MCG \times Ratio1$

5) 按权重 Weight，加权平均 HCG 和 LR：

$$DCG_{combine} = \frac{LR \times (128 - Weight) + HCG \times Weight}{128}$$

DCG combine 模式开启步骤：（发布资料）

1）曝光、模拟读出配置成 DCG 模式，同时产生 HCG、MCG 图像。

2）写 0x5180=0x01，开启 DCG_combine_en。

3）必要时调整 0x5185，0x5186，以调整合并的权重曲线。

4.13.4. TCG combine

芯片配置成 DCG 模式——同时输出 HCG，MCG 两帧图像。此时可配置 HDR_combine 模块工作在 DCG combine 模式。合并后的图像，可选择直接输出，或通过内置 PWL 模块，缩减位宽后输出；

TCG combine 工作过程简述如下：（内部参考）

1) 关闭 BLC 中的 offset——HCG，MCG，LCG 以 0 为起点；

2) 自动计算曝光时间差异、模拟增益差异对 CG_ratio 的影响：

$$Ratio1 = \frac{TEXP \times T_{row} + TEXP_{clk} + adj_H}{TEXP \times T_{row} + TEXP_{clk} + adj_M} \times \frac{Again_H}{Again_M} \times otp\_HM\_ratio$$

$$Ratio2 = \frac{TEXP \times T_{row} + TEXP_{clk} + adj_H}{TEXP \times T_{row} + TEXP_{clk} + adj_L} \times \frac{Again_H}{Again_L} \times otp\_HL\_ratio$$

3) 按 4.14.2 方式，融合 HCG，MCG 作为 DCG 图像；
4) 融合 DCG 和 LCG。取 DCG 高 8Bit 作为 HW，以 HW 作为索引按如下方式计算 Weight，其中 Hth 由寄存器 0x5187 设置，Lth 由寄存器 0x5188 设置。

```
if(HW <= Lth)
    Weight = 128;
elseif(HW >= Hth)
    Weight = 0;
else
    Weight = (Hth-HW)*128/(Hth-Lth);
end
```

5) LCG 乘以 Ratio2：$LR = LCG \times Ratio2$
6) 按权重 Weight，加权平均 HCG 和 LR：

$$TCG_{combine} = \frac{LR \times (128 - Weight) + DCG \times Weight}{128}$$

DCG combine 模式开启步骤：（发布资料）

1）曝光、模拟读出配置成 TCG 模式，同时产生 HCG、MCG、LCG 图像。

2）写 0x5180=0x02，开启 TCG_combine_en。

3）必要时调整 0x5185，0x5186，0x5187，0x5188 以调整合并的权重曲线。

4.13.5. PWL 压缩

RST0112R 内置 PWL 压缩模块，可缩减 HDR_combine 之后的数据位宽。通过设置如下寄存器：

offset2（{0x518F,0x5190}），

offset3（{0x5191,0x5192}），

offset4（{0x5193,0x5194}），

offset5（{0x5195,0x5196}），

offset6（{0x5197,0x5198}），

shift1（0x5189），shift2（0x518A），shift3（0x518B），shift4（0x518C），shift5（0x518D），shift6（0x518E），

以及 thres1（{0x5199,0x519A,0x519B}），

thres2（{0x519C,0x519D,0x519E}）

thres3（{0x519F,0x51A0,0x51A1}）

thres4（{0x51A2,0x51A3,0x51A4}）

thres5（{0x51A5,0x51A6,0x51A7}）

构建如图 4 - 17 所示的分段函数：

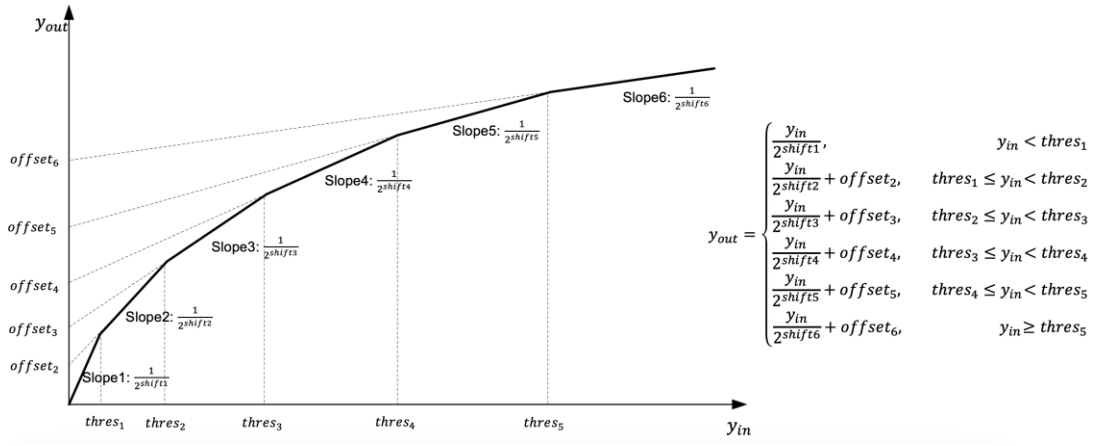


图 4 - 17 PWL 分段函数及其曲线

PWL 支持如下几种模式：

24Bit 直出 ： 0x51B0 = 0x00

24Bit→16Bit ： 0x51B0 = 0x01

24Bit→12Bit ： 0x51B0 = 0x02

24Bit→10Bit ： 0x51B0 = 0x03

16Bit 直出 ： 0x51B0 = 0x04

16Bit→12Bit ： 0x51B0 = 0x05

16Bit→10Bit ： 0x51B0 = 0x06

PWL 24Bit→16Bit 示例：

1） 输出范围为 16Bit，这里将输出范围平均分为 6 段，每段码值Δ≈10924

2） 确定每段斜率 shift1~shift6

3） 6 段线段须连续，因此依次计算出 thres1 → offset2 → thres2 → offset3 → thres3 → offset4 →t hres4 → offset5 → thres5 → offset6

shift	offset	thres
0x00		0x002AAC
0x01	0x1556	0x008004
0x02	0x3557	0x012AB4
0x04	0x6D58	0x03D574
0x08	0xA6DA	0x2E8174
0x0A	0xC9BB	

图 4 - 18 PWL 24Bit→16Bit 设置示例

PWL 24 → 16bit

$\Delta = 10924$

Δ	$\frac{y_{in}}{2^0},$	$y_{in} < \Delta$
2Δ	$\frac{y_{in}}{2^1} + \frac{\Delta}{2},$	$\Delta \le y_{in} < \Delta \times 3$
3Δ	$\frac{y_{in}}{2^2} + \frac{5\Delta}{4},$	$\Delta \times 3 \le y_{in} < \Delta \times 7$
4Δ	$\frac{y_{in}}{2^4} + \frac{41\Delta}{16},$	$\Delta \times 7 \le y_{in} < \Delta \times 23$
5Δ	$\frac{y_{in}}{2^8} + \frac{348\Delta}{89},$	$\Delta \times 23 \le y_{in} < \Delta \times 279$
~6Δ	$\frac{y_{in}}{2^{10}} + \frac{52\Delta}{11},$	$\Delta \times 279 \le y_{in}$

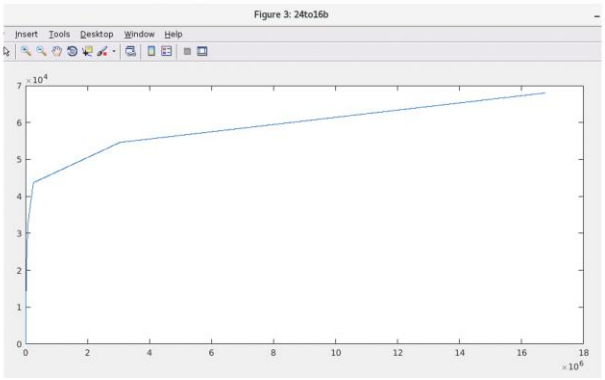


图 4 - 19 PWL 24Bit→16Bit 示例曲线

PWL 24Bit→12Bit 示例：

1） 输出范围为 12Bit，这里将输出范围平均分为 6 段，每段码值Δ≈684

2） 确定每段斜率 shift1~shift6

3） 6 段线段须连续，因此依次计算出 thres1 → offset2 → thres2 → offset3 → thres3 → offset4 →t hres4 → offset5 → thres5 → offset6

shift	offset	thres
0x00		0x0002AC
0x03	0x0256	0x00180C
0x06	0x04F7	0x00C30C
0x09	0x07A2	0x061B0C
0x0C	0x0A4E	0x30DB0C
0x0E	0x0C98	

图 4 - 20 PWL 24Bit→12Bit 设置示例

PWL 24 → 12bit

$\Delta = 684$

Δ	$\frac{y_{in}}{2^0},$	$y_{in} < \Delta$
2Δ	$\frac{y_{in}}{2^3} + \frac{7\Delta}{8},$	$\Delta \le y_{in} < \Delta \times 9$
3Δ	$\frac{y_{in}}{2^6} + \frac{119\Delta}{64},$	$\Delta \times 9 \le y_{in} < \Delta \times 73$
4Δ	$\frac{y_{in}}{2^9} + \frac{20\Delta}{7},$	$\Delta \times 73 \le y_{in} < \Delta \times 584$
5Δ	$\frac{y_{in}}{2^{12}} + \frac{27\Delta}{7},$	$\Delta \times 584 \le y_{in} < \Delta \times 4681$
~6Δ	$\frac{y_{in}}{2^{14}} + \frac{33\Delta}{7},$	$\Delta \times 4681 \le y_{in}$

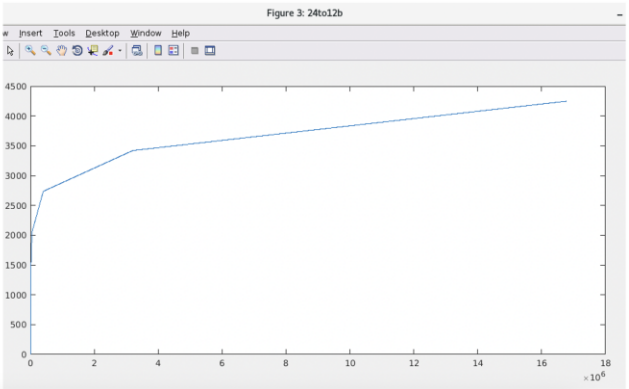


图 4 - 21 PWL 24Bit→12Bit 示例曲线

PWL 24Bit→10Bit 示例：

1） 输出范围为 10Bit，这里将输出范围平均分为 6 段，每段码值Δ≈171

2） 确定每段斜率 shift1~shift6

3） 6 段线段须连续，因此依次计算出 thres1 → offset2 → thres2 → offset3 → thres3 → offset4 →t hres4 → offset5 → thres5 → offset6

shift	offset	thres
0x00		0x0000AB
0x03	0x0095	0x000603
0x06	0x013D	0x0030C3
0x09	0x01E8	0x0186C3
0x0C	0x0293	0x0C36C3
0x10	0x034A	

图 4 - 22 PWL 24Bit→10Bit 设置示例

PWL 24 → 10bit

$\Delta = 171$

Δ	$\frac{y_{in}}{2^0}$,	$y_{in} < \Delta$
2Δ	$\frac{y_{in}}{2^3} + \frac{7\Delta}{8}$,	$\Delta \le y_{in} < \Delta\times 9$
3Δ	$\frac{y_{in}}{2^6} + \frac{119\Delta}{64}$,	$\Delta\times 9 \le y_{in} < \Delta\times 73$
4Δ	$\frac{y_{in}}{2^9} + \frac{20\Delta}{7}$,	$\Delta\times 73 \le y_{in} < \Delta\times 585$
5Δ	$\frac{y_{in}}{2^{12}} + \frac{27\Delta}{7}$,	$\Delta\times 585 \le y_{in} < \Delta\times 4681$
$\sim 6\Delta$	$\frac{y_{in}}{2^{16}} + \frac{69\Delta}{14}$,	$\Delta\times 4681 \le y_{in}$

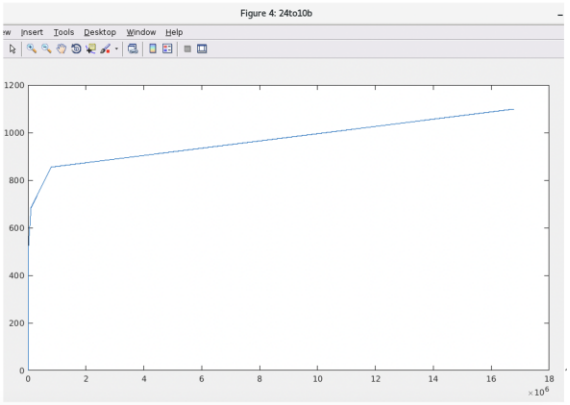


图 4 - 23 PWL 24Bit→10Bit 示例曲线

PWL 16Bit→12Bit 示例：

1） 输出范围为 12Bit，这里将输出范围平均分为 6 段，每段码值Δ≈684

2） 确定每段斜率 shift1~shift6

3） 6 段线段须连续，因此依次计算出 thres1 → offset2 → thres2 → offset3 → thres3 → offset4 →t hres4 → offset5 → thres5 → offset6

shift	offset	thres
0x00		0x0002AC
0x01	0x0156	0x000804
0x02	0x0357	0x0012B4
0x03	0x05AD	0x002814
0x04	0x082E	0x0052D4
0x06	0x0C10	

图 4 - 24 PWL 16Bit→12Bit 设置示例

PWL 16 → 12bit　　Δ= 684

Δ	$\frac{y_{in}}{2^0}$,	$y_{in} < \Delta$
2Δ	$\frac{y_{in}}{2^1}+\frac{\Delta}{2}$,	$\Delta \le y_{in} < \Delta\times3$
3Δ	$\frac{y_{in}}{2^2}+\frac{5\Delta}{4}$,	$\Delta\times3 \le y_{in} < \Delta\times7$
4Δ	$\frac{y_{in}}{2^3}+\frac{17\Delta}{8}$,	$\Delta\times7 \le y_{in} < \Delta\times15$
5Δ	$\frac{y_{in}}{2^4}+\frac{49\Delta}{16}$,	$\Delta\times15 \le y_{in} < \Delta\times31$
~6Δ	$\frac{y_{in}}{2^6}+\frac{289\Delta}{64}$,	$\Delta\times31 \le y_{in}$

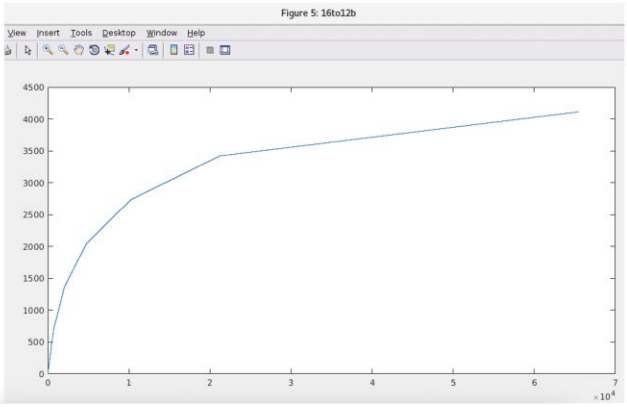


图 4 - 25 PWL 16Bit→12Bit 示例曲线

PWL 16Bit→10Bit 示例：

1） 输出范围为 10Bit，这里将输出范围平均分为 6 段，每段码值Δ≈171

2） 确定每段斜率 shift1~shift6

3） 6 段线段须连续，因此依次计算出 thres1 → offset2 → thres2 → offset3 → thres3 → offset4 →t hres4 → offset5 → thres5 → offset6

shift	offset	thres
0x00		0x0000AB
0x01	0x0055	0x000201
0x02	0x00D5	0x0004AD
0x04	0x01B6	0x000F5D
0x06	0x026E	0x003A1D
0x08	0x031C	

图 4 - 26 PWL 16Bit→10Bit 设置示例

PWL 16 → 10bit　　Δ= 171

Δ	$\frac{y_{in}}{2^0}$,	$y_{in} < \Delta$
2Δ	$\frac{y_{in}}{2^1}+\frac{\Delta}{2}$,	$\Delta \le y_{in} < \Delta\times3$
3Δ	$\frac{y_{in}}{2^2}+\frac{5\Delta}{4}$,	$\Delta\times3 \le y_{in} < \Delta\times7$
4Δ	$\frac{y_{in}}{2^4}+\frac{41\Delta}{16}$,	$\Delta\times7 \le y_{in} < \Delta\times23$
5Δ	$\frac{y_{in}}{2^6}+\frac{233\Delta}{64}$,	$\Delta\times23 \le y_{in} < \Delta\times87$
~6Δ	$\frac{y_{in}}{2^8}+\frac{233\Delta}{50}$,	$\Delta\times87 \le y_{in}$

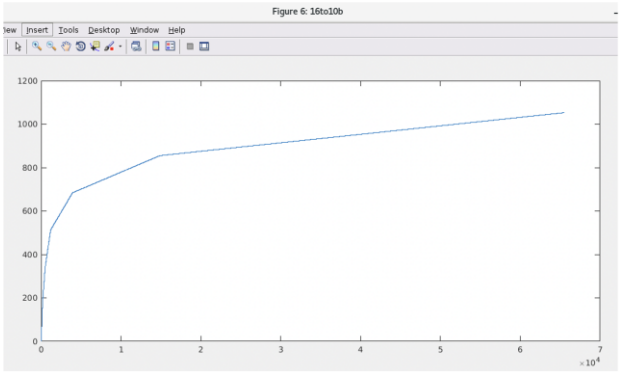


图 4 - 27 PWL 16Bit→10Bit 示例曲线

4.14. BINNING

RST0112R 支持 2x2binning 模式，在保持视场的同时可以提供较低分辨率输出。RST0112R 支持 2x2 color binning 和 2x2 mono binning 功能。

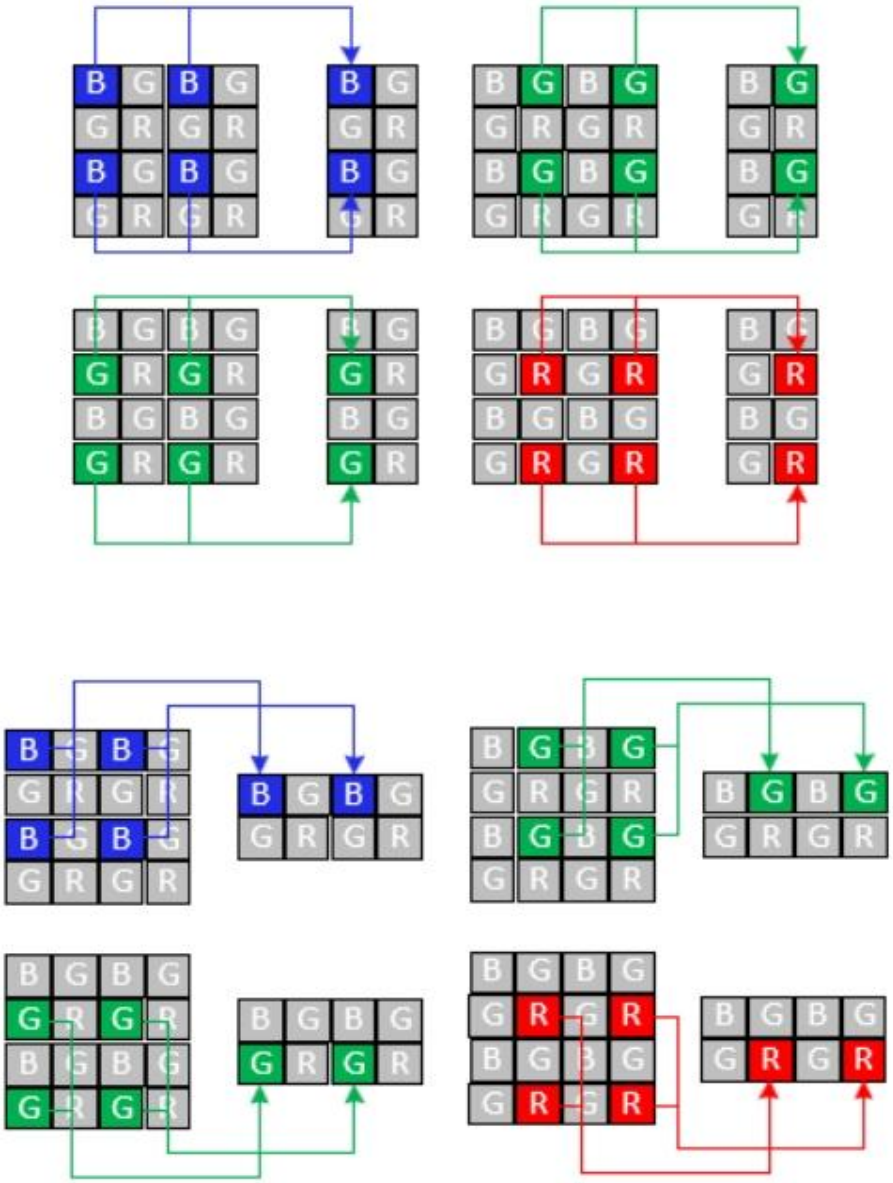


图 4 - 28 2x2 color binning 示意图

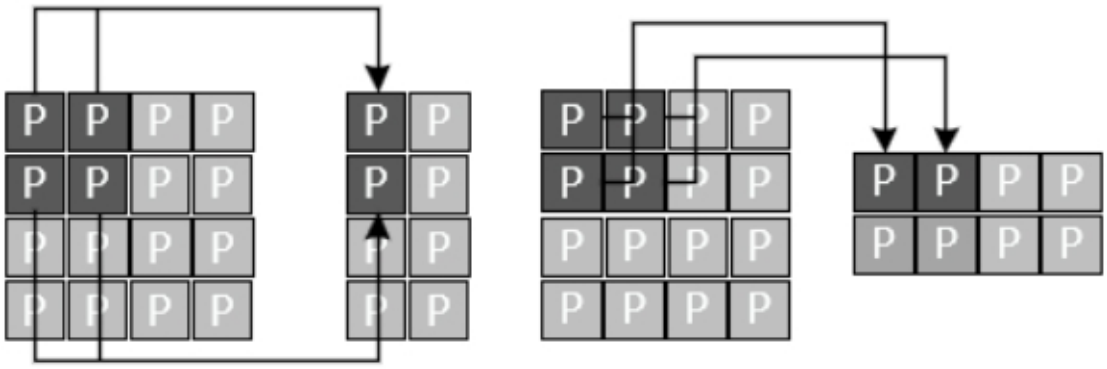


图 4 - 29 2x2 mono binning 示意图

Binning 寄存器的使用：

1) 2x2 color binning 在寄存器 0x3A21 和 0x3A22 同时写入 0x01 时开启，两个寄存器分别对应垂直和水平方向上的 color binning。
2) 2x2 mono binning 在寄存器 0x3A21 和 0x3A22 同时写入 0x02 时开启，两个寄存器分别对应垂直和水平方向上的 mono binning。

表 4 - 26 binning 寄存器表

Address	Register name	Default	R/W	Description
0x3A21	V_BIN2	0x00	R/W	Bit[1]:v_bin2_mono_en Bit[0]:v_bin2_color_en
0x3A22	H_BIN2	0x00	R/W	Bit[1]:h_bin2_mono_en Bit[0]:h_bin2_color_en

5. 读出接口

数据接口由寄存器 0x4830 设置，Bit[2]写入 0 并且 Bit[0]写入 1 数据接口为 MIPI，Bit[2]写入 1 并且 Bit[0]写入 0 数据接口为 SLVS。

表 5 - 1 数据接口设置寄存器表

Address	Register name	Default	R/W	Description
0x4830	MIPI_EN	0x00	R/W	Bit[2]:slvs_en Bit[1]:mipi_en_update_mode Bit[0]:mipi_en

5.1. MIPI

该芯片支持 1/2/4lane 的 MIPI 数据读出，根据输出位宽（10、12、16、24Bit）和输出通道数量不同，数据接口最高为 1Gbps/lane。

MIPI 接口连接关系如下：

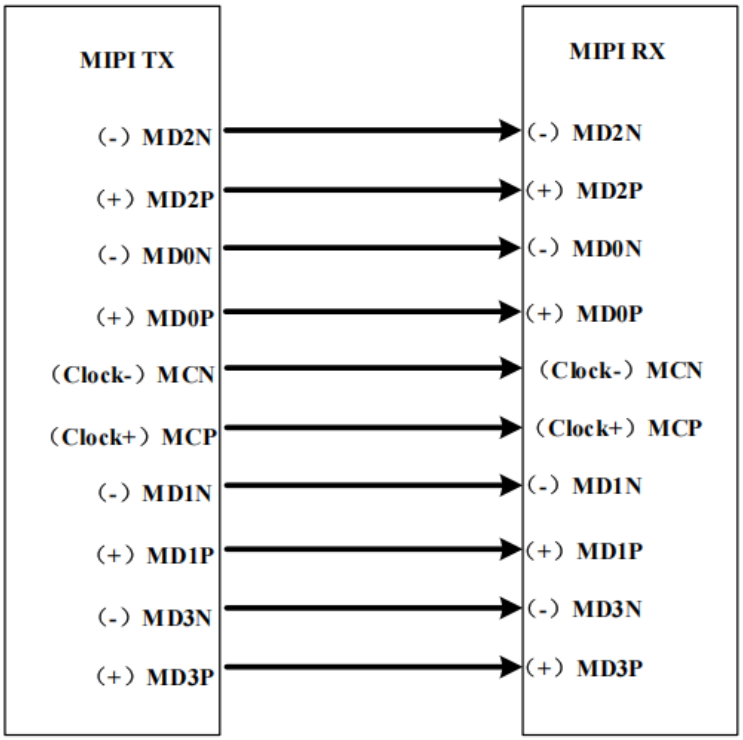


图 5 - 1 MIPI 接口连接方式

D-PHY 功能

D-PHY 将数据从 MIPI_TX 转换为与 MIPI 协议兼容的串行数据。详细的时间序列如下：

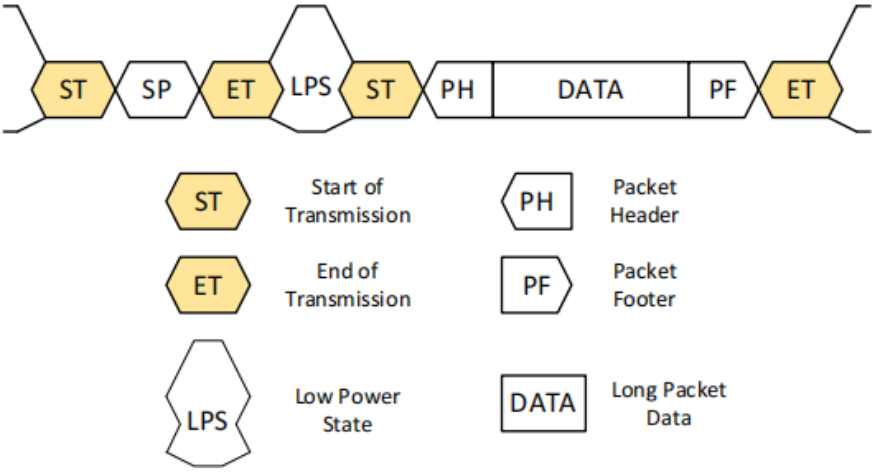


图 5 - 2 MIPI 底层数据包示意图

图 5 - 2 是 MIPI 底层数据包的简略示意图，其中分别展示了一个短数据包和长数据包的传输过程。图 5 - 3 展示了 MIPI 长、短数据包结构示意图。其中数据标识 DI(Data Identifier)用来区分不同的数据包类型。图 5 - 4 展示了 MIPI 工作在 1lane、2lane 和 4lane 模式下的数据包传输示意图。图 5 - 5 中，DI 包括两部分，分别是虚拟通道（VC）和数据类型（DT）。默认情况下，Sensor 给出的 MIPI 数据 VC 值都是 0，而 DT 值如表 5 - 3 所示。

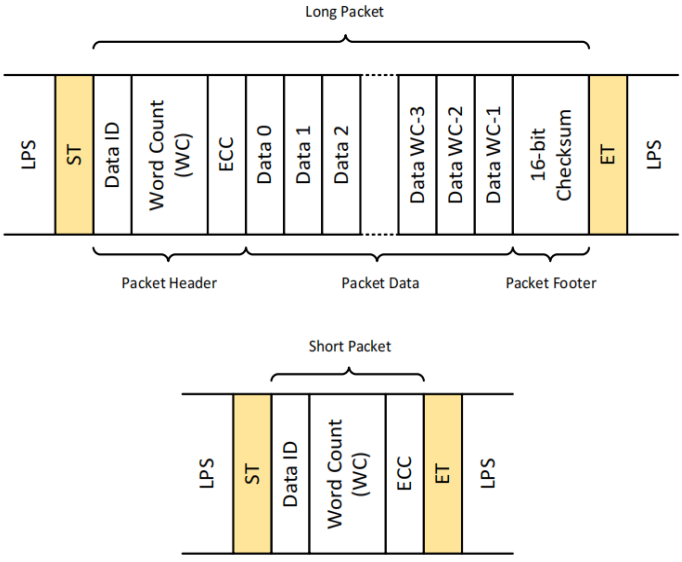


图 5 - 3 MIPI 底层数据包示意图

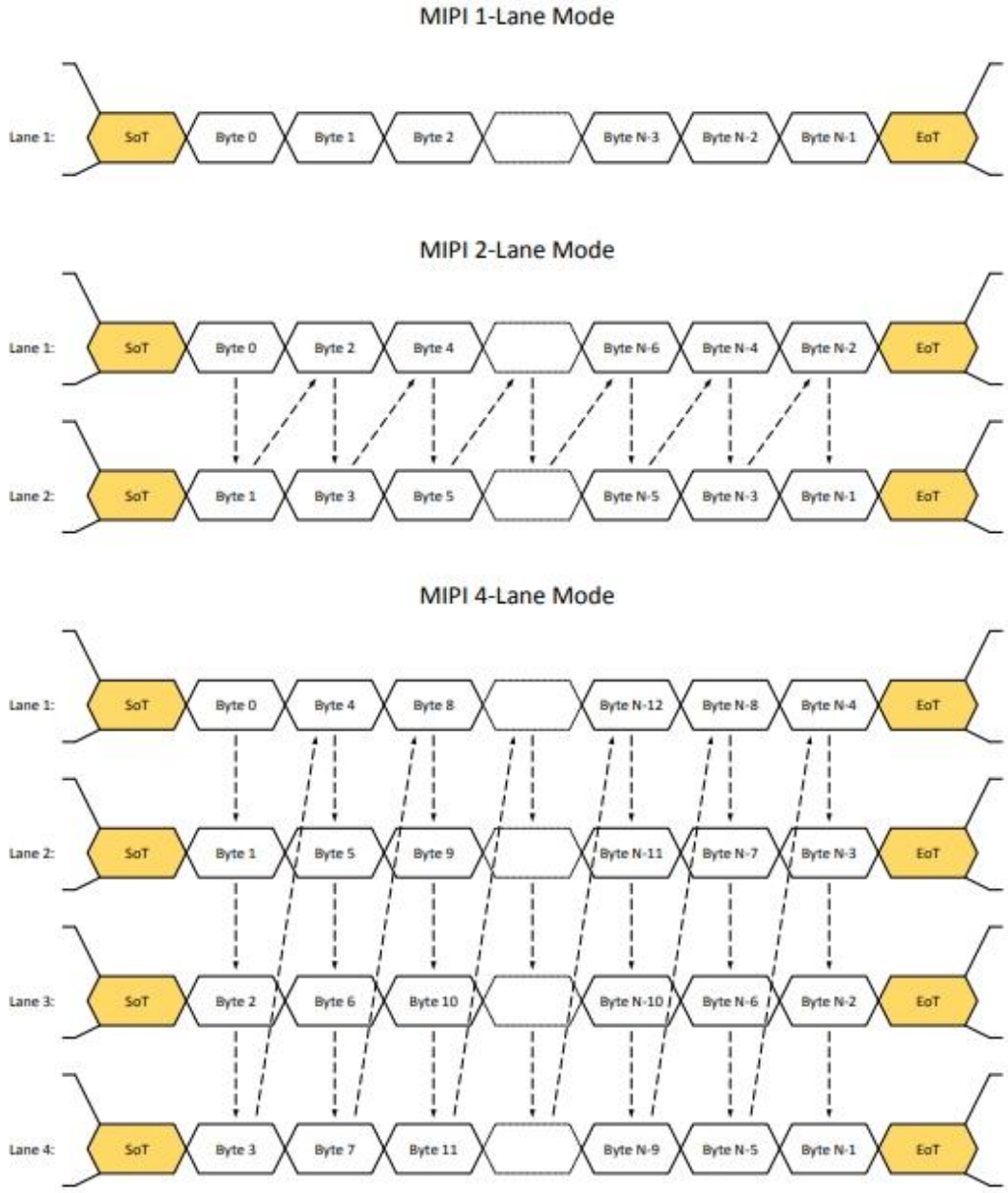


图 5 - 4 MIPI 1/2/4 Lane 模式数据包传输示意图

由 0x4824 寄存器控制 MIPI 的 Lane 数，写入 0x00 为 1Lane，写入 0x01 为 2Lane，写入 0x02 为 4Lane。

表 5 - 2 MIPI Lane 寄存器表

Address	Register name	Default	R/W	Description
0x4824	LANE_NUM	0x01	R/W	Bit[1:0]:Lane_num

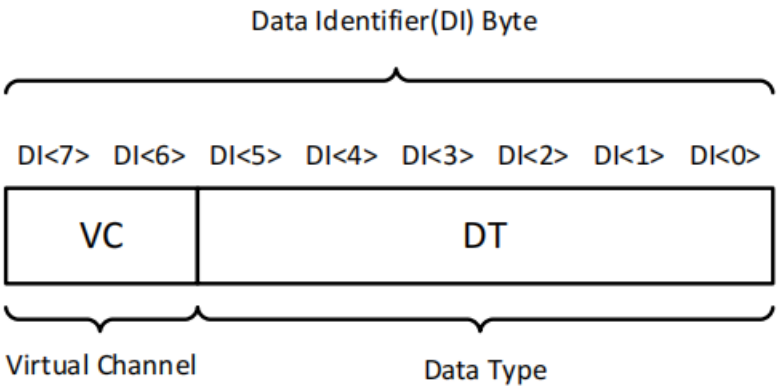


图 5 - 5 MIPI 数据包 DI 结构

表 5 - 3 MIPI 数据类型

DT	描述
6'h00	帧起始短包
6'h01	帧结束短包
6'h02	行起始短包
6'h03	行结束短包
6'h2b	10-Bit 模式下数据长包
6'h2c	12-Bit 模式下数据长包
6'h2e	16-Bit 模式下数据长包
6'h27	24-Bit 模式下数据长包

由寄存器 0x480e 设置 MIPI 数据类型，写入 0x2b 为 10bit 模式，写入 0x2c 为 12bit 模式，写入 0x2e 为 16bit 模式，写入 0x27 为 24bit 模式。

表 5 - 4 MIPI 数据类型寄存器表

Address	Register name	Default	R/W	Description
0x480e	DATA_ID	0x2b	R/W	Bit[7:0]:data_id

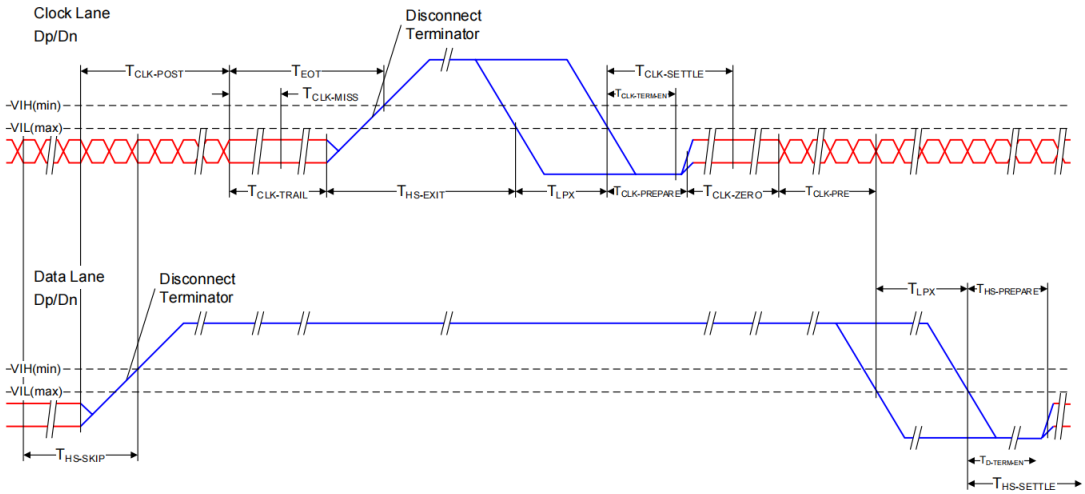


图 5 - 6 Switching the Clock Lane between Clock Transmission and Low-Power Mode

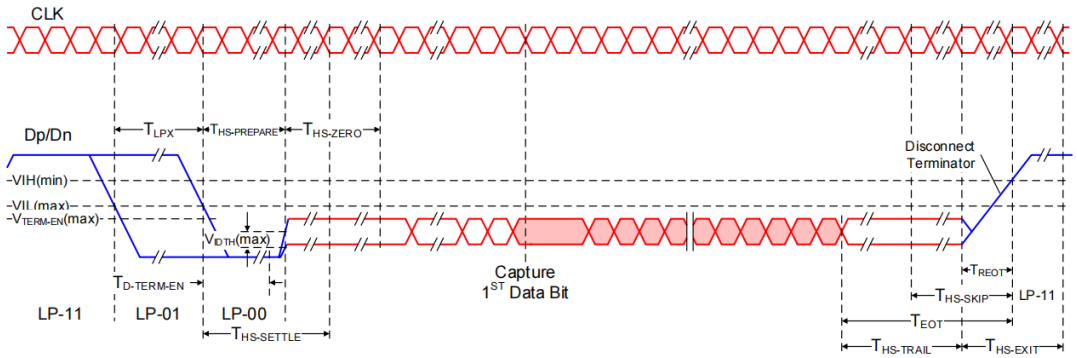


图 5 - 7 High-speed Data Transmission Bursts

表 5 - 5 MIPI 寄存器表

Address	Register name	Default	R/W	Description
0x4800	DC_TEST_LP_LK	0x3f	R/W	Bit[7:0]:dc_test_lp_lk
0x4801	DC_TEST_DATA_HS	0xff	R/W	Bit[7:0]:dc_test_data_hs
0x4802	ULP_MODE	0x00	R/W	Bit[0]:ulp_mode
0x4803	LS_MODE	0x00	R/W	Bit[0]:ls_mode
0x4804	HS_MODE	0x00	R/W	Bit[0]:hs_mode
0x4805	HS_MODE_VF	0x00	R/W	Bit[0]:hs_mode_vf
0x4806	INIT	0x00	R/W	Bit[0]:init
0x4807	TESTMODE	0x00	R/W	Bit[1:0]:testmode
0x4808	FRAME_END_DLY	0x01	R/W	Bit[7:0]:frame_end_dly[15:8]
0x4809	FRAME_EDN_DLY	0x65	R/W	Bit[7:0]:frame_end_dly[7:0]
0x480a	TX_SPEED_AREA_SEL	0x05	R/W	Bit[2:0]:tx_speed_area_sel
0x480b	VIRTUAL_CHANNEL	0x24	R/W	Bit[5:0]:virtual_channel

0x480c	MIPI_HYSNC_NUM	0x96	R/W	Bit[7:0]:mipi_hsync_num
0x480d	MIPI_LS_START_NUM	0x00	R/W	Bit[1:0]:mipi_ls_start_num
0x480f	HBLANK_MAX	0x01	R/W	Bit[4:0]:hblank_max[12:8]
0x4810	HBLANK_MAX	0x65	R/W	Bit[7:0]:hblank_max[7:0]
0x4811	H_SIZE_MIPI	0x05	R/W	Bit[3:0]:h_size_mipi[11:8]
0x4812	H_SIZE_MIPI	0x00	R/W	Bit[7:0]:h_size_mipi[7:0]
0x4813	V_SIZE_MIPI	0x03	R/W	Bit[2:0]:v_size_mipi[10:8]
0x4814	V_SIZE_MIPI	0x20	R/W	Bit[7:0]:v_size_mipi[7:0]
0x4815	R_LPX_DAT	0x07	R/W	Bit[3:0]:r_lpx_dat[3:0]
0x4816	R_HS_PREPARE	0x06	R/W	Bit[3:0]:r_hs_prepare[3:0]
0x4817	R_HS_ZERO	0x1a	R/W	Bit[4:0]:r_hs_zero[4:0]
0x4818	R_HS_TRAIL	0x0c	R/W	Bit[3:0]:r_hs_trail[3:0]
0x4819	R_EXIT	0x04	R/W	Bit[3:0]:r_exit[3:0]
0x481a	R_LPX_CK	0x07	R/W	Bit[3:0]:r_lpx_ck[3:0]
0x481b	R_CLK_PREPARE	0x05	R/W	Bit[3:0]:r_clk_prepare[3:0]
0x481c	R_CLK_ZERO	0x22	R/W	Bit[5:0]:r_clk_zero[5:0]
0x481d	R_CLK_POST	0x0e	R/W	Bit[4:0]:r_clk_post[4:0]
0x481e	R_CLK_TRAIL	0x07	R/W	Bit[3:0]:r_clk_trail[3:0]
0x481f	R_WAKEUP	0x00	R/W	Bit[1:0]:r_wakeup[17:16]
0x4820	R_WAKEUP	0x86	R/W	Bit[7:0]:r_wakeup[15:8]
0x4821	R_WAKEUP	0x88	R/W	Bit[7:0]:r_wakeup[7:0]
0x4822	R_INIT	0x0b	R/W	Bit[7:0]:r_init[15:8]
0x4823	R_INIT	0x40	R/W	Bit[7:0]:r_init[7:0]
0x4825	MIPI_DLANE_EN	0x00	R/W	Bit[2:0]:MIPI_dlane_en
0x4826	MIPI_CK_SKEW	0x00	R/W	Bit[1:0]:MIPI_ck_skew
0x4827	MIPI_PAUSE_CK	0x00	R/W	Bit[0]:MIPI_pause_ck
0x4828	MIPI_DA_SKEW	0x00	R/W	Bit[7:0]:MIPI_da_skew
0x4829	MIPI_HS_DRV	0x5a	R/W	Bit[7:0]:MIPI_hs_drv
0x482a	MIPI_HS_VREF	0x02	R/W	Bit[2:0]:MIPI_hs_vref[10:8]
0x482b	MIPI_HS_VREF	0x95	R/W	Bit[7:0]:MIPI_hs_vref[7:0]
0x482c	MIPI_LP_CTRL	0x04	R/W	Bit[2:0]:MIPI_lp_ctrl
0x482d	MIPI_LP_SST_EN	0x00	R/W	Bit[0]:MIPI_lp_sst_en
0x482e	MIPI_MPLL_BIAS_EN	0x00	R/W	Bit[0]:MIPI_MPLL_bias_en
0x482f	MIPI_PHY_EN	0x00	R/W	Bit[1]:MIPI_phy_en_LPBit[0]:MIPI_phy_en_Nor
0x4831	MIPI_EN_DLY	0x50	R/W	Bit[7:0]:mipi_en_dly[8:1] mipi_en_dly[0]=1'b0

5.2. SLVS

RST0112R 提供串行视频端口（SLVS），其数据接口与 MIPI 数据接口复用，通过寄存器控制选择输出 SLVS 格式数据。支持 4 data lane 来传输图像 10/12/16/24 Bit RAW 数据，默认先传输数据的 HSB 位。使用 SAV(Invalid)和 EAV(Invalid)标识消隐区的无效数据，使用 SAV(Valid)和 EAV(Valid)标识有效区像素数据，这种同步方式如图所示。

<table>
<tr><td>H.BLK</td><td rowspan="3">SAV
Invalid line</td><td>V.BLK</td><td rowspan="3">EAV
Invalid line</td><td>H.BLK</td></tr>
<tr><td>H.BLK</td><td>V.BLK</td><td>H.BLK</td></tr>
<tr><td>H.BLK</td><td>V.BLK</td><td>H.BLK</td></tr>
<tr><td>H.BLK</td><td rowspan="11">SAV
Valid line</td><td>Effective Pixel</td><td rowspan="11">EAV
Valid line</td><td>H.BLK</td></tr>
<tr><td>H.BLK</td><td>Effective Pixel</td><td>H.BLK</td></tr>
<tr><td>⋮
⋮</td><td>⋮
⋮</td><td>⋮
⋮</td></tr>
<tr><td>H.BLK</td><td>Effective Pixel</td><td>H.BLK</td></tr>
<tr><td>H.BLK</td><td>Effective Pixel</td><td>H.BLK</td></tr>
<tr><td>H.BLK</td><td>Effective Pixel</td><td>H.BLK</td></tr>
<tr><td>H.BLK</td><td>Effective Pixel</td><td>H.BLK</td></tr>
<tr><td>H.BLK</td><td>Effective Pixel</td><td>H.BLK</td></tr>
<tr><td>H.BLK</td><td rowspan="4">SAV
Invalid line</td><td>V.BLK</td><td rowspan="4">EAV
Invalid line</td><td>H.BLK</td></tr>
<tr><td>⋮</td><td>⋮</td><td>⋮</td></tr>
<tr><td>H.BLK</td><td>V.BLK</td><td>H.BLK</td></tr>
<tr><td>H.BLK</td><td>V.BLK</td><td>H.BLK</td></tr>
</table>

图 5 - 8 SLVS 同步方式

在 SLVS 传输模式中，行同步信号集成在数据流中，在数据流中的特殊码型 SAV 和 EAV 分别表示行的起始和结束，再由不同的 SAV 和 EAV 区分是否为有效行。在数据流中，SAV 和 EAV 由 4 个字段构成，每个字段的位宽与像素数据保持一致，前 3 个字段为固定的基准码字，根据第 4 个字段来区分帧/行的起始或结束。SLVS 同步码格式如表 5 - 6 所示。

表 5 - 6 SLVS 同步码格式表

Field	Bit Width	Sync code			
		SAV (Valid line)	EAV (Valid line)	SAV (Invalid line)	SAV (Valid line)
1st code	10Bit	3FFh	3FFh	3FFh	3FFh
	12Bit	FFFh	FFFh	FFFh	FFFh
	16Bit	FFFFh	FFFFh	FFFFh	FFFFh
	24Bit	FFFFFFh	FFFFFFh	FFFFFFh	FFFFFFh

2nd code	10Bit	000h	000h	000h	000h
	12Bit	000h	000h	000h	000h
	16Bit	0000h	0000h	0000h	0000h
	24Bit	000000h	000000h	000000h	000000h
3rd code	10Bit	000h	000h	000h	000h
	12Bit	000h	000h	000h	000h
	16Bit	0000h	0000h	0000h	0000h
	24Bit	000000h	000000h	000000h	000000h
4th code	10Bit	XXXh	XXXh	XXXh	XXXh
	12Bit	XXXh	XXXh	XXXh	XXXh
	16Bit	XXXXh	XXXXh	XXXXh	XXXXh
	24Bit	XXXXXXh	XXXXXXh	XXXXXXh	XXXXXXh

LVDS 同步码和像素数据在各个 Lane 上传输方式如图 5 - 9 所示， 图中 H 表示同步码，P 表示像素，H 和 P 的位宽与图像传感器输出单个像素的位宽一致。各个数据通道首先传输 4 个像素位宽的同步码，紧接着是像素数据。

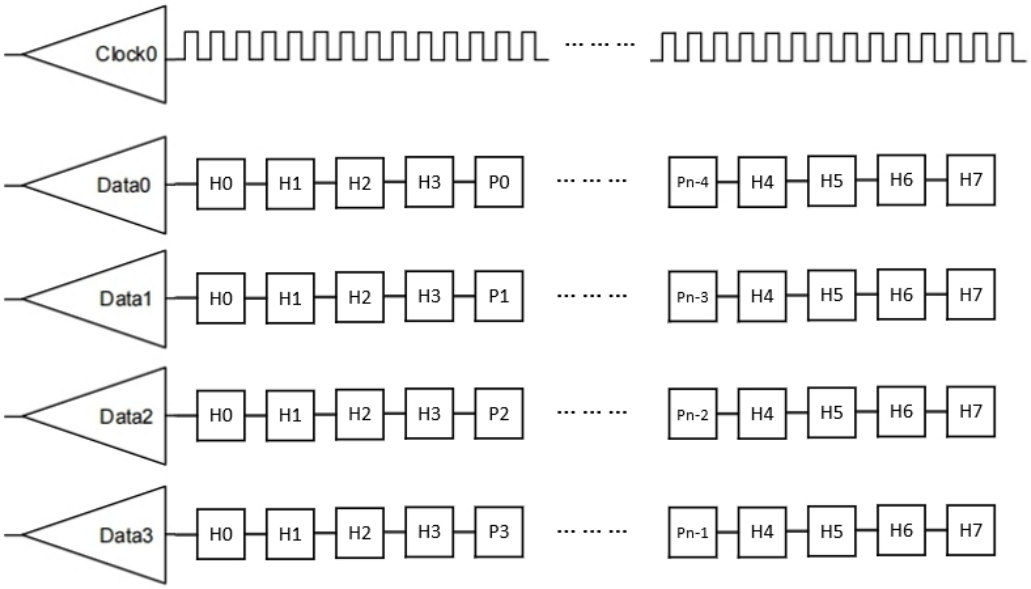


图 5 - 9 SLVS 同步码和图像传输方式

同步码和像素数据的传输是串行的。以 RAW12 为例，图像传感器输出单个像素点的时序如图 5 - 10 所示。

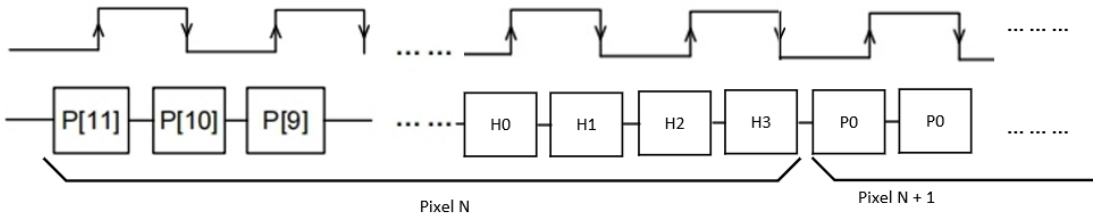


图 5 - 10 SLVS 单个像素点时序

SLVS 宽动态模式如下面两图所示。使用 SAV-EAV 作为同步方式，不同 CG 有独立的同步码，第 N 帧和第 N+1 帧的同步码也不同。

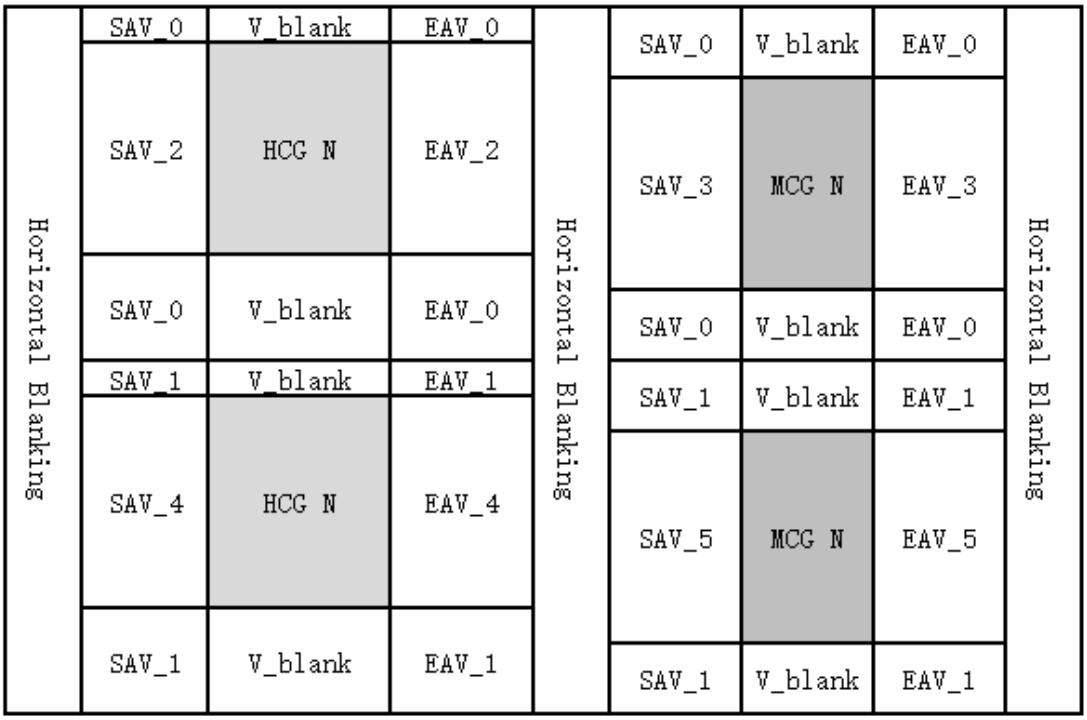


图 5 - 11 SLVS DualCG 宽动态模式

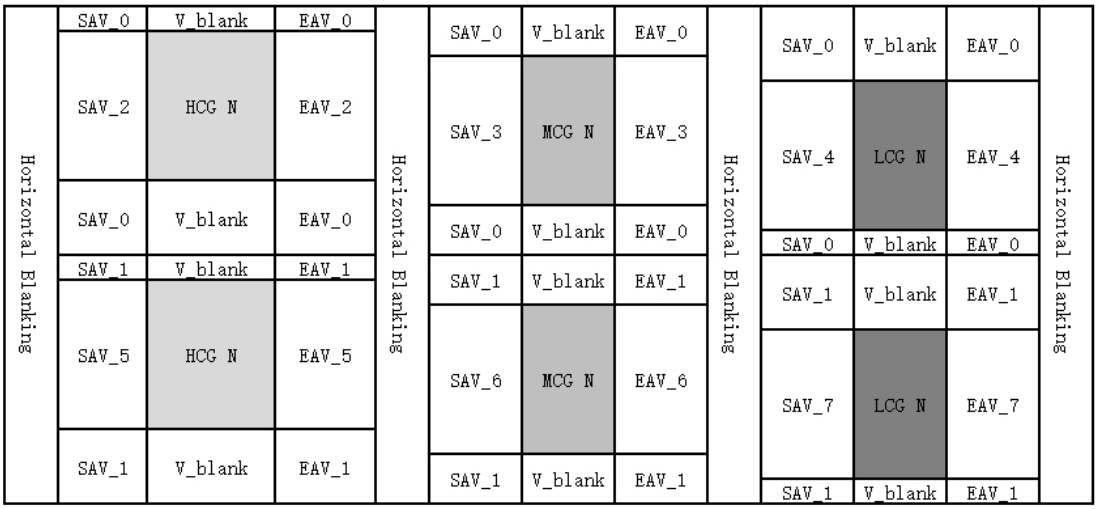


图 5 - 12 SLVS TripleCG 宽动态模式

表 5 - 7 SLVS 4 字段同步码寄存器表

Address	Register name	Default	R/W	Description
0x4700	SLVS_SAV0	0xab	R/W	Bit[7:0]:sav0
0x4701	SLVS_EAV0	0xb6	R/W	Bit[7:0]:eav0
0x4702	SLVS_SAV1	0xac	R/W	Bit[7:0]:sav1
0x4703	SLVS_EAV1	0xb7	R/W	Bit[7:0]:eav1
0x4704	SLVS_SAV2	0x80	R/W	Bit[7:0]:sav2
0x4705	SLVS_EAV2	0x9d	R/W	Bit[7:0]:eav2
0x4706	SLVS_SAV3	0x81	R/W	Bit[7:0]:sav3
0x4707	SLVS_EAV3	0x9e	R/W	Bit[7:0]:eav3
0x4708	SLVS_SAV4	0x82	R/W	Bit[7:0]:sav4
0x4709	SLVS_EAV4	0x9f	R/W	Bit[7:0]:eav4
0x470a	SLVS_SAV5	0x83	R/W	Bit[7:0]:sav5
0x470b	SLVS_EAV5	0xa0	R/W	Bit[7:0]:eav5
0x470c	SLVS_SAV6	0x84	R/W	Bit[7:0]:sav6
0x470d	SLVS_EAV6	0xa1	R/W	Bit[7:0]:eav6
0x470e	SLVS_SAV7	0x85	R/W	Bit[7:0]:sav7
0x470f	SLVS_SAV7	0xa2	R/W	Bit[7:0]:eav7

6. 典型应用实例（调试建议）

6.1. CG 和增益的动态切换策略

RST0112R 支持 HCG 和 MCG 两种像素级增益调节，其中 HCG 灵敏度高适用于弱光条件，MCG 灵敏度低适用于强光条件。在应用中可通过增益表映射来将 CG 作为传感器增益一部分进行动态切换，来适应不同光照环境。

推荐的增益表如下，表格内容较长，部分内容由公式和数值范围进行代替。下表中 SENSOR_GAIN 表示传感器输出的总的增益，包含模拟增益和 CG 两部分的贡献。

推荐表中 SENSOR_GAIN 取值范围 1~930，最大增益值随传感器工作模式会有变化，请咨询技术支持团队获取实际最大模拟增益设置以保证最优性能。

表 6 - 1 推荐的应用端增益表

SENSOR_GAIN	CG	RealAGAIN_MCG/HCG	AGAIN_MCG/HCG
1~13 SENSOR_GAIN=RealAGAIN_MCG	MCG	1~13 参考表 10-2 调节	参考表 10-2 调节
13.5	MCG	13.5	216
14	MCG	14	224
14.5	MCG	14.5	232
15	MCG	15	240
15.9375	HCG	1.0625	17
16.875	HCG	1.125	18
17.8125	HCG	1.1875	19
18.75	HCG	1.25	20
19.6875~930 SENSOR_GAIN=15x RealAGAIN_HCG	HCG	1.3125~62 参考表 10-2 调节	参考表 10-2 调节

6.2. 曝光和增益配置的生效时间

所有涉及 CG、增益、曝光时间的配置，需要在写完上述寄存器后统一写入一次 NewSetting（0x33fd=0x01）来让上述配置实际生效。

RST0112R 芯片内部工作机制决定，第 N 帧配置的曝光时间调节会在第 N+2 帧生效，默认增益·CG 的调节也是在第 N+2 帧生效，在平台中设置曝光调节延迟为 2；平台设定生效时间为 Delay 两帧。

7. 寄存器列表

7.1. SYS0_regf (0x01xx)

表 7 - 1 SYS0_regf

Address	Register name	Default	R/W	Description
0x0103	SOFTRST	0x00	W	Bit[0]:softrst
0x0104	RESTART	0x00	W	Bit[0]:restart

7.2. PLL_regf (0x03xx)

表 7 - 2 PLL_regf

Address	Register name	Default	R/W	Description
0x0300	CPLL_CTRL00	0x07	R/W	Bit[2:0]:PLL_bias_code_15
0x0301	CPLL_CTRL01	0x00	R/W	Bit[2:0]:CPLL_MC_15
0x0302	CPLL_CTRL02	0x0a	R/W	Bit[7:0]:CPLL_NC_15
0x0303	CPLL_CTRL03	0x00	R/W	Bit[1:0]:CPLL_CNTclk_div_code_15
0x0304	CPLL_CTRL04	0x01	R/W	Bit[7:0]:CPLL_dctl_15
0x0305	CPLL_CTRL05	0x00	R/W	Bit[7:0]:CPLL_en_15
0x0306	CPLL_CTRL06	0x00	R/W	Bit[1:0]:CPLL_phase_15
0x0307	CPLL_CTRL07	0x00	R/W	Bit[1:0]:CPLL_DIGclk_div_code_15
0x0308	MPLL_CTRL00	0x00	R/W	Bit[2:0]:MPLL_bias_code_15
0x0309	MPLL_CTRL01	0x11	R/W	Bit[2:0]:MPLL_MC_15
0x030a	MPLL_CTRL02	0x07	R/W	Bit[7:0]:MPLL_NC_15
0x030b	MPLL_CTRL03	0x03	R/W	Bit[1:0]:MPLL_bias_code_15
0x030c	MPLL_CTRL04	0x00	R/W	Bit[7:0]:MPLL_bin_ctrl_15
0x030d	MPLL_CTRL05	0x00	R/W	Bit[7:0]:MPLL_en_15
0x030e	MPLL_CTRL06	0x00	R/W	Bit[1:0]:MPLL_phase_15
0x030f	MPLL_CTRL07	0x00	R/W	Bit[1:0]:MPLL_dig_clk_code_15
0x0310	PLL_CTRL00	0x00	R/W	Bit[7:5]:spi_clk_div Bit[4]:clk_src_div Bit[1:0]:dclk_div
0x0311	PLL_CTRL01	0x00	R/W	Bit[4]:row_clk_sel Bit[2:0]:row_clk_div
0x0312	PLL_CTRL02	0x44	R/W	Bit[6:4]:MPLL_pcp_clk_code_15 Bit[2:0]:MPLL_ncp_clk_code_15
0x0313	PLL_CTRL03	0x00	R/W	Bit[1:0]:MPLL_Bclk_phase_15
0x0314	PLL_CTRL04	0x00	R/W	Bit[7:0]:MPLL_clkout_code_15
0x0315	PLL_CTRL05	0x00	R/W	Bit[6]:dis_gat_bclk Bit[5]:dis_gat_dclk_hdr

				Bit[4]:dis_gat_dclk_otp_dpc Bit[3]:dis_gat_colclk Bit[2]:dis_gat_dclk Bit[1]:dis_gat_row_clk_sht Bit[0]:dis_gat_row_clk_rd

7.3. SYSC_regf(0x30xx)

表 7 - 3 SYSC_regf

Address	Register name	Default	R/W	Description
0x3010	SYSC_CTRL0	0x00	R/W	Bit[1]:STROBE_oe Bit[0]:FSYNC_oe
0x3011	SYSC_CTRL1	0x00	R/W	Bit[1:0]:DS

7.4. PSM_regf (0x32xx)

表 7 - 4 PSM_regf

Address	Register name	Default	R/W	Description
0x3200	LP_CTRL0	0x03	R/W	Bit[2]:LP_pre_enable Bit[1]:LP_post_enable Bit[0]:LP_mid_enable
0x3201	LP_CTRL1	0x04	R/W	Bit[7:0]:LP_ctrl1
0x3202	LP_CTRL2	0x04	R/W	Bit[7:0]:LP_ctrl2
0x3203	LP_CTRL3	0x02	R/W	Bit[7:0]:LP_ctrl3
0x3204	LP_CTRL4	0x0a	R/W	Bit[7:0]:LP_ctrl4
0x3205	LP_CTRL5	0x05	R/W	Bit[7:0]:LP_ctrl5
0x3206	LP_EXP_TH	0x00	R/W	Bit[7:0]:LP_exp_th[15:8]
0x3207	LP_EXP_TH	0x0a	R/W	Bit[7:0]:LP_exp_th[7:0]
0x3208	LP_VB_TH	0x00	R/W	Bit[7:0]:LP_VB_th[15:8]
0x3209	LP_VB_TH	0x0f	R/W	Bit[7:0]:LP_VB_th[7:0]
0x320a	LP_CTRL6	0x04	R/W	Bit[7:0]:LP_ctrl6[15:8]
0x320b	LP_CTRL6	0x50	R/W	Bit[7:0]:LP_ctrl6[7:0]

7.5. TIMINGC_regf (0x33xx,0x3Axx)

表 7 - 5 TIMINGC_regf

Address	Register name	Default	R/W	Description
0x3301	TEXP_PR	0x00	R/W	Bit[15:8]:texp_pr[15:8]

0x3302	TEXP_PR	0x01	R/W	Bit[7:0]:texp_pr[7:0]
0x3303	TEXP_CLK_PR	0x00	R/W	Bit[15:8]:texp_clk_pr[15:8]
0x3304	TEXP_CLK_PR	0x00	R/W	Bit[7:0]:texp_clk_pr[7:0]
0x330B	DATA_MODE	0x00	R/W	Bit[2:0]:data_mode
0x330C	CMS_NUM	0x11	R/W	Bit[7:4]:HCG_cms_num Bit[3:0]:MCG_cms_num
0x330D	READ_MODE_PR	0x00	R/W	Bit[2:0]:read_mode_pr
0x330E	RPC_MAP_IDX	0x00	W	Bit[5:4]:rpc_mapH_idx Bit[3:2]:rpc_mapM_idx Bit[1:0]:rpc_mapL_idx
0x330F	RPC_MODE_PR	0x01	R/W	Bit[2:0]:rpc_mode_pr
0x3310	RPCH_PR	0x00	R/W	Bit[12:8]:rpcH_pr[12:8]
0x3311	RPCH_PR	0x10	R/W	Bit[7:0]:rpcH_pr[7:0]
0x3312	DGAINH	0x00	R/W	Bit[13:8]:dGainH_pr[13:8]
0x3313	DGAINH	0x80	R/W	Bit[7:0]:dGainH_pr[7:0]
0x3314	RPCM_PR	0x00	R/W	Bit[12:8]:rpcM_pr[12:8]
0x3315	RPCM_PR	0x10	R/W	Bit[7:0]:rpcM_pr[7:0]
0x3316	DGAINM	0x00	R/W	Bit[13:8]:dGainM_pr[13:8]
0x3317	DGAINM	0x80	R/W	Bit[7:0]:dGainM_pr[7:0]
0x3318	RPCL_PR	0x00	R/W	Bit[12:8]:rpcL_pr[12:8]
0x3319	RPCL_PR	0x10	R/W	Bit[7:0]:rpcL_pr[7:0]
0x331A	DGAINL	0x00	R/W	Bit[13:8]:dGainL_pr[13:8]
0x331B	DGAINL	0x80	R/W	Bit[7:0]:dGainL_pr[7:0]
0x331C	PGA_GAIN_GODEH_DIRECT	0x00	R/W	Bit[1:0]:PGA_gain_codeH_direct
0x331D	COL_GAIN_GODEH_DIRECT	0x00	R/W	Bit[2:0]:COL_gain_codeH_direct
0x331E	RAMP_GAIN_GODEH_DIRECT	0x00	R/W	Bit[5:0]:RAMP_gain_codeH_direct
0x331F	PGA_GAIN_GODEM_DIRECT	0x00	R/W	Bit[1:0]:PGA_gain_codeM_direct
0x3320	COL_GAIN_GODEM_DIRECT	0x00	R/W	Bit[2:0]:COL_gain_codeM_direct
0x3321	RAMP_GAIN_GODEM_DIRECT	0x00	R/W	Bit[5:0]:RAMP_gain_codeM_direct
0x3322	PGA_GAIN_GODEL_DIRECT	0x00	R/W	Bit[1:0]:PGA_gain_codeL_direct
0x3323	COL_GAIN_GODEL_DIRECT	0x00	R/W	Bit[2:0]:COL_gain_codeL_direct
0x3324	RAMP_GAIN_GODEL_DIRECT	0x00	R/W	Bit[5:0]:RAMP_gain_codeL_direct
0x3325	TIMING_CTRL0	0x07	R/W	Bit[7]:Again_test Bit[6]:LinearColSwitch_tmp

				Bit[5]:signal_manual Bit[4]:dummy_addr_en_sht Bit[1]:dummy_addr_en_rd Bit[0]:dummy_addr_en_hug
0x3326	DEL_FRM_MODE	0x51	R/W	Bit[7:0]:del_frm_mode
0x3327	TIMING_CTRL1	0x24	R/W	Bit[7]:darkrow_mode Bit[6]:darkrow_oe Bit[5]:double_shutter_en Bit[4]:sht_vb_hold Bit[3]:rd_vb_hold_rolling Bit[2]:rd_vb_hold_golbal
0x3330	ROW_BOOST_READ_IN_SEL_CTRL1	0x1F	R/W	Bit[4]:row_boost_read_in_sel1_rpc4 Bit[3]:row_boost_read_in_sel1_rpc3 Bit[2]:row_boost_read_in_sel1_rpc2 Bit[1]:row_boost_read_in_sel1_rpc1 Bit[0]:row_boost_read_in_sel1_rpc0
0x3331	ROW_BOOST_READ_IN_SEL_CTRL2	0x1F	R/W	Bit[4]:row_boost_read_in_sel2_rpc4 Bit[3]:row_boost_read_in_sel2_rpc3 Bit[2]:row_boost_read_in_sel2_rpc2 Bit[1]:row_boost_read_in_sel2_rpc1 Bit[0]:row_boost_read_in_sel2_rpc0
0x3332	PGA_BYPASS_RPC_CTRL0	0x00	R/W	Bit[4]:PGA_bypassH_rpc4 Bit[3]:PGA_bypassH_rpc3 Bit[2]:PGA_bypassH_rpc2 Bit[1]:PGA_bypassH_rpc1 Bit[0]:PGA_bypassH_rpc0
0x3333	PGA_BYPASS_RPC_CTRL1	0x1F	R/W	Bit[4]:PGA_bypassM_rpc4 Bit[3]:PGA_bypassM_rpc3 Bit[2]:PGA_bypassM_rpc2 Bit[1]:PGA_bypassM_rpc1 Bit[0]:PGA_bypassM_rpc0
0x3334	PGA_BYPASS_RPC_CTRL2	0x1F	R/W	Bit[4]:PGA_bypassL_rpc4 Bit[3]:PGA_bypassL_rpc3 Bit[2]:PGA_bypassL_rpc2 Bit[1]:PGA_bypassL_rpc1 Bit[0]:PGA_bypassL_rpc0
0x3335	PGA_LOAD_CODE_RPC_CTRL0	0x00	R/W	Bit[3:0]:PGA_load_code_rpc4
0x3336	PGA_LOAD_CODE_RPC_CTRL1	0x00	R/W	Bit[7:4]:PGA_load_code_rpc3 Bit[3:0]:PGA_load_code_rpc2
0x3337	PGA_LOAD_CODE_RPC_CTRL2	0x00	R/W	Bit[7:4]:PGA_load_code_rpc1 Bit[3:0]:PGA_load_code_rpc0
0x3338	COMP_BYPASS_CAS_SEL_RPC_CTRL0	0x00	R/W	Bit[5:4]:comp_bypass_cas_selH_rpc4 Bit[3:2]:comp_bypass_cas_selH_rpc3

				Bit[1:0]:comp_bypass_cas_selH_rpc2
0x3339	COMP_BYPASS_CAS_SEL_RPC_CTRL1	0x00	R/W	Bit[7:6]:comp_bypass_cas_selH_rpc1 Bit[5:4]:comp_bypass_cas_selH_rpc0 Bit[3:2]:comp_bypass_cas_selM_rpc4 Bit[1:0]:comp_bypass_cas_selM_rpc3
0x333A	COMP_BYPASS_CAS_SEL_RPC_CTRL2	0x00	R/W	Bit[7:6]:comp_bypass_cas_selM_rpc2 Bit[5:4]:comp_bypass_cas_selM_rpc1 Bit[3:2]:comp_bypass_cas_selM_rpc0 Bit[1:0]:comp_bypass_cas_selL_rpc4
0x333B	COMP_BYPASS_CAS_SEL_RPC_CTRL3	0x00	R/W	Bit[7:6]:comp_bypass_cas_selL_rpc3 Bit[5:4]:comp_bypass_cas_selL_rpc2 Bit[3:2]:comp_bypass_cas_selL_rpc1 Bit[1:0]:comp_bypass_cas_selL_rpc0
0x333C	COMP_CAPLOAD_RPC_CTRL0	0x00	R/W	Bit[5:4]:comp_caploadH_rpc4 Bit[3:2]:comp_caploadH_rpc3 Bit[1:0]:comp_caploadH_rpc2
0x333D	COMP_CAPLOAD_RPC_CTRL1	0x00	R/W	Bit[7:6]:comp_caploadH_rpc1 Bit[5:4]:comp_caploadH_rpc0 Bit[3:2]:comp_caploadM_rpc4 Bit[1:0]:comp_caploadM_rpc3
0x333E	COMP_CAPLOAD_RPC_CTRL2	0x00	R/W	Bit[7:6]:comp_caploadM_rpc2 Bit[5:4]:comp_caploadM_rpc1 Bit[3:2]:comp_caploadM_rpc0 Bit[1:0]:comp_caploadL_rpc4
0x333F	COMP_CAPLOAD_RPC_CTRL3	0x00	R/W	Bit[7:6]:comp_caploadL_rpc3 Bit[5:4]:comp_caploadL_rpc2 Bit[3:2]:comp_caploadL_rpc1 Bit[1:0]:comp_caploadL_rpc0
0x3340	COMP_PGADIR_SEL_RPC_CTRL0	0x7F	R/W	Bit[6]:comp_pgadir_selH_rpc4 Bit[5]:comp_pgadir_selH_rpc3 Bit[4]:comp_pgadir_selH_rpc2 Bit[3]:comp_pgadir_selH_rpc1 Bit[2]:comp_pgadir_selH_rpc0 Bit[1]:comp_pgadir_selM_rpc4 Bit[0]:comp_pgadir_selM_rpc3
0x3341	COMP_PGADIR_SEL_RPC_CTRL1	0xE0	R/W	Bit[7]:comp_pgadir_selM_rpc2 Bit[6]:comp_pgadir_selM_rpc1 Bit[5]:comp_pgadir_selM_rpc0 Bit[4]:comp_pgadir_selL_rpc4 Bit[3]:comp_pgadir_selL_rpc3 Bit[2]:comp_pgadir_selL_rpc2 Bit[1]:comp_pgadir_selL_rpc1 Bit[0]:comp_pgadir_selL_rpc0

0x3342	CNT_INPUT_SEL_MANNUAL	0x00	R/W	Bit[0]:cnt_input_sel_mannual
0x3345	DAC_OUT_SEL_RPC_CTRL0	0x7F	R/W	Bit[6]:DAC_out_selH_rpc4 Bit[5]:DAC_out_selH_rpc3 Bit[4]:DAC_out_selH_rpc2 Bit[3]:DAC_out_selH_rpc1 Bit[2]:DAC_out_selH_rpc0 Bit[1]:DAC_out_selM_rpc4 Bit[0]:DAC_out_selM_rpc3
0x3346	DAC_OUT_SEL_RPC_CTRL1	0xE0	R/W	Bit[7]:DAC_out_selM_rpc2 Bit[6]:DAC_out_selM_rpc1 Bit[5]:DAC_out_selM_rpc0 Bit[4]:DAC_out_selL_rpc4 Bit[3]:DAC_out_selL_rpc3 Bit[2]:DAC_out_selL_rpc2 Bit[1]:DAC_out_selL_rpc1 Bit[0]:DAC_out_selL_rpc0
0x3347	CLAMP_BLACKSUN_CODE_RPC_CTRL0	0x0D	R/W	Bit[3:0]:clamp_blacksun_codeH_rpc4
0x3348	CLAMP_BLACKSUN_CODE_RPC_CTRL1	0xDD	R/W	Bit[7:4]:clamp_blacksun_codeH_rpc3 Bit[3:0]:clamp_blacksun_codeH_rpc2
0x3349	CLAMP_BLACKSUN_CODE_RPC_CTRL2	0xDD	R/W	Bit[7:4]:clamp_blacksun_codeH_rpc1 Bit[3:0]:clamp_blacksun_codeH_rpc0
0x334A	CLAMP_BLACKSUN_CODE_RPC_CTRL3	0xDD	R/W	Bit[7:4]:clamp_blacksun_codeM_rpc4 Bit[3:0]:clamp_blacksun_codeM_rpc3
0x334B	CLAMP_BLACKSUN_CODE_RPC_CTRL4	0xDD	R/W	Bit[7:4]:clamp_blacksun_codeM_rpc2 Bit[3:0]:clamp_blacksun_codeM_rpc1
0x334C	CLAMP_BLACKSUN_CODE_RPC_CTRL5	0xDD	R/W	Bit[7:4]:clamp_blacksun_codeM_rpc0 Bit[3:0]:clamp_blacksun_codeL_rpc4
0x334D	CLAMP_BLACKSUN_CODE_RPC_CTRL6	0xDD	R/W	Bit[7:4]:clamp_blacksun_codeL_rpc3 Bit[3:0]:clamp_blacksun_codeL_rpc2
0x334E	CLAMP_BLACKSUN_CODE_RPC_CTRL7	0xDD	R/W	Bit[7:4]:clamp_blacksun_codeL_rpc1 Bit[3:0]:clamp_blacksun_codeL_rpc0
0x334F	CLAMP_SIG_CODE_RPC_CTRL0	0xDD	R/W	Bit[3:0]:clamp_SIG_codeH_rpc4
0x3350	CLAMP_SIG_CODE_RPC_CTRL1	0x55	R/W	Bit[7:4]:clamp_SIG_codeH_rpc3 Bit[3:0]:clamp_SIG_codeH_rpc2
0x3351	CLAMP_SIG_CODE_RPC_CTRL2	0x55	R/W	Bit[7:4]:clamp_SIG_codeH_rpc1 Bit[3:0]:clamp_SIG_codeH_rpc0
0x3352	CLAMP_SIG_CODE_RPC_CTRL3	0x55	R/W	Bit[7:4]:clamp_SIG_codeM_rpc4 Bit[3:0]:clamp_SIG_codeM_rpc3
0x3353	CLAMP_SIG_CODE_RPC_CTRL4	0x55	R/W	Bit[7:4]:clamp_SIG_codeM_rpc2 Bit[3:0]:clamp_SIG_codeM_rpc1

0x3354	CLAMP_SIG_CODE_RPC_CTRL5	0x55	R/W	Bit[7:4]:clamp_SIG_codeM_rpc0 Bit[3:0]:clamp_SIG_codeL_rpc4
0x3355	CLAMP_SIG_CODE_RPC_CTRL6	0x55	R/W	Bit[7:4]:clamp_SIG_codeL_rpc3 Bit[3:0]:clamp_SIG_codeL_rpc2
0x3356	CLAMP_SIG_CODE_RPC_CTRL7	0x55	R/W	Bit[7:4]:clamp_SIG_codeL_rpc1 Bit[3:0]:clamp_SIG_codeL_rpc0
0x3357	CLAMP_RST_CODE_RPC_CTRL0	0x0D	R/W	Bit[3:0]:clamp_RST_codeH_rpc4
0x3358	CLAMP_RST_CODE_RPC_CTRL1	0xDD	R/W	Bit[7:4]:clamp_RST_codeH_rpc3 Bit[3:0]:clamp_RST_codeH_rpc2
0x3359	CLAMP_RST_CODE_RPC_CTRL2	0xDD	R/W	Bit[7:4]:clamp_RST_codeH_rpc1 Bit[3:0]:clamp_RST_codeH_rpc0
0x335A	CLAMP_RST_CODE_RPC_CTRL3	0xDD	R/W	Bit[7:4]:clamp_RST_codeM_rpc4 Bit[3:0]:clamp_RST_codeM_rpc3
0x335B	CLAMP_RST_CODE_RPC_CTRL4	0xDD	R/W	Bit[7:4]:clamp_RST_codeM_rpc2 Bit[3:0]:clamp_RST_codeM_rpc1
0x335C	CLAMP_RST_CODE_RPC_CTRL5	0xDD	R/W	Bit[7:4]:clamp_RST_codeM_rpc0 Bit[3:0]:clamp_RST_codeL_rpc4
0x335D	CLAMP_RST_CODE_RPC_CTRL6	0xDD	R/W	Bit[7:4]:clamp_RST_codeL_rpc3 Bit[3:0]:clamp_RST_codeL_rpc2
0x335E	CLAMP_RST_CODE_RPC_CTRL7	0xDD	R/W	Bit[7:4]:clamp_RST_codeL_rpc1 Bit[3:0]:clamp_RST_codeL_rpc0
0x335F	PIX_CTRL0	0xBB	R/W	Bit[7]:pixH_PU Bit[6:4]:pixH_code Bit[3]:pixM_PU Bit[2:0]:pixM_code
0x3360	PIX_CTRL1	0x0B	R/W	Bit[3]:pixL_PU Bit[2:0]:pixL_code
0x3361	RAMP_CODE_RPC_CTRL0	0x0C	R/W	Bit[3:0]:rampH_code_rpc4
0x3362	RAMP_CODE_RPC_CTRL1	0xCC	R/W	Bit[7:4]:rampH_code_rpc3 Bit[3:0]:rampH_code_rpc2
0x3363	RAMP_CODE_RPC_CTRL2	0xCC	R/W	Bit[7:4]:rampH_code_rpc1 Bit[3:0]:rampH_code_rpc0
0x3364	RAMP_CODE_RPC_CTRL3	0xCC	R/W	Bit[7:4]:rampM_code_rpc4 Bit[3:0]:rampM_code_rpc3
0x3365	RAMP_CODE_RPC_CTRL4	0xCC	R/W	Bit[7:4]:rampM_code_rpc2 Bit[3:0]:rampM_code_rpc1
0x3366	RAMP_CODE_RPC_CTRL5	0xCC	R/W	Bit[7:4]:rampM_code_rpc0 Bit[3:0]:rampL_code_rpc4
0x3367	RAMP_CODE_RPC_CTRL6	0xCC	R/W	Bit[7:4]:rampL_code_rpc3 Bit[3:0]:rampL_code_rpc2
0x3368	RAMP_CODE_RPC_CTRL7	0xCC	R/W	Bit[7:4]:rampL_code_rpc1 Bit[3:0]:rampL_code_rpc0

0x3369	CLAMP_RST_PU_RPC_CTRL0	0x7F	R/W	Bit[6]:clamp_RST_PU_H_rpc4 Bit[5]:clamp_RST_PU_H_rpc3 Bit[4]:clamp_RST_PU_H_rpc2 Bit[3]:clamp_RST_PU_H_rpc1 Bit[2]:clamp_RST_PU_H_rpc0 Bit[1]:clamp_RST_PU_M_rpc4 Bit[0]:clamp_RST_PU_M_rpc3
0x336A	CLAMP_RST_PU_RPC_CTRL1	0xFF	R/W	Bit[7]:clamp_RST_PU_M_rpc2 Bit[6]:clamp_RST_PU_M_rpc1 Bit[5]:clamp_RST_PU_M_rpc0 Bit[4]:clamp_RST_PU_L_rpc4 Bit[3]:clamp_RST_PU_L_rpc3 Bit[2]:clamp_RST_PU_L_rpc2 Bit[1]:clamp_RST_PU_L_rpc1 Bit[0]:clamp_RST_PU_L_rpc0
0x336B	CLAMP_RST_PU_RPC_CTRL2	0xFF	R/W	Bit[6]:clamp_SIG_PD_H_rpc4 Bit[5]:clamp_SIG_PD_H_rpc3 Bit[4]:clamp_SIG_PD_H_rpc2 Bit[3]:clamp_SIG_PD_H_rpc1 Bit[2]:clamp_SIG_PD_H_rpc0 Bit[1]:clamp_SIG_PD_M_rpc4 Bit[0]:clamp_SIG_PD_M_rpc3
0x336C	CLAMP_RST_PU_RPC_CTRL3	0xFF	R/W	Bit[7]:clamp_SIG_PD_M_rpc2 Bit[6]:clamp_SIG_PD_M_rpc1 Bit[5]:clamp_SIG_PD_M_rpc0 Bit[4]:clamp_SIG_PD_L_rpc4 Bit[3]:clamp_SIG_PD_L_rpc3 Bit[2]:clamp_SIG_PD_L_rpc2 Bit[1]:clamp_SIG_PD_L_rpc1 Bit[0]:clamp_SIG_PD_L_rpc0
0x336D	DAC_OUT_SEL_RPC_CTRL0	0xFF	R/W	Bit[6]:DAC_out_selH_rpc4 Bit[5]:DAC_out_selH_rpc3 Bit[4]:DAC_out_selH_rpc2 Bit[3]:DAC_out_selH_rpc1 Bit[2]:DAC_out_selH_rpc0 Bit[1]:DAC_out_selM_rpc4 Bit[0]:DAC_out_selM_rpc3
0x336E	DAC_OUT_SEL_RPC_CTRL1	0xFF	R/W	Bit[7]:DAC_out_selM_rpc2 Bit[6]:DAC_out_selM_rpc1 Bit[5]:DAC_out_selM_rpc0 Bit[4]:DAC_out_selL_rpc4 Bit[3]:DAC_out_selL_rpc3 Bit[2]:DAC_out_selL_rpc2 Bit[1]:DAC_out_selL_rpc1

				Bit[0]:DAC_out_selL_rpc0
0x336F	RAMP_PD_RPC_CTRL0	0xFF	R/W	Bit[6]:rampH_PD_rpc4 Bit[5]:rampH_PD_rpc3 Bit[4]:rampH_PD_rpc2 Bit[3]:rampH_PD_rpc1 Bit[2]:rampH_PD_rpc0 Bit[1]:rampM_PD_rpc4 Bit[0]:rampM_PD_rpc3
0x3370	RAMP_PD_RPC_CTRL1	0xFF	R/W	Bit[7]:rampM_PD_rpc2 Bit[6]:rampM_PD_rpc1 Bit[5]:rampM_PD_rpc0 Bit[4]:rampL_PD_rpc4 Bit[3]:rampL_PD_rpc3 Bit[2]:rampL_PD_rpc2 Bit[1]:rampL_PD_rpc1 Bit[0]:rampL_PD_rpc0
0x3371	DAC_OUT_SEL_M	0xFF	R/W	Bit[7]:DAC_out_selH_m Bit[6]:DAC_out_selM_m Bit[5]:DAC_out_selM_m
0x3372	R_MWB_PR	0x80	R/W	Bit[7:0]:R_mwb_pr
0x3373	GR_MWB_PR	0x80	R/W	Bit[7:0]:Gr_mwb_pr
0x3374	B_MWB_PR	0x80	R/W	Bit[7:0]:B_mwb_pr
0x3375	GB_MWB_PR	0x80	R/W	Bit[7:0]:Gb_mwb_pr
0x33fb	TIMING_CTRL2	0x00	R/W	Bit[1]:NoSHT_1st_frm Bit[0]:startexp_pr
0x33fd	UPDATE_MODE	0x00	R/W	Bit[0]:update_mode
0x3A00	X_ADDR_START	0x00	R/W	Bit[0]:x_addr_start[8]
0x3A01	X_ADDR_START	0x00	R/W	Bit[7:0]:x_addr_start[7:0]
0x3A02	X_ADDR_end	0x00	R/W	Bit[0]:x_addr_end[8]
0x3A03	X_ADDR_end	0x01	R/W	Bit[7:0]:x_addr_end[7:0]
0x3A04	Y_ADDR_START_REG	0x43	R/W	Bit[1:0]:y_addr_start_reg[9:8]
0x3A05	Y_ADDR_START_REG	0x00	R/W	Bit[7:0]:y_addr_start_reg[7:0]
0x3A06	Y_ADDR_end_reg	0x10	R/W	Bit[1:0]:y_addr_end_reg[9:8]
0x3A07	Y_ADDR_end_reg	0x03	R/W	Bit[7:0]:y_addr_end_reg[7:0]
0x3A08	X_OUTPUT_SIZE	0x37	R/W	Bit[0]:x_output_size[8]
0x3A09	X_OUTPUT_SIZE	0x01	R/W	Bit[7:0]:x_output_size[7:0]
0x3A0A	Y_OUTPUT_SIZE	0x03	R/W	Bit[1:0]:y_output_size[9:8]
0x3A0B	Y_OUTPUT_SIZE	0x20	R/W	Bit[7:0]:y_output_size[7:0]
0x3A0C	ROWTIME	0x02	R/W	Bit[4:0]:rowtime[12:8]
0x3A0D	ROWTIME	0x00	R/W	Bit[7:0]:rowtime[7:0]
0x3A10	VBLANK	0x00	R/W	Bit[7:0]:vblank[15:8]

0x3A11	VBLANK	0x01	R/W	Bit[7:0]:vblank[7:0]
0x3A12	X_OUTPUT_OFFSET	0x00	R/W	Bit[0]:x_output_offset[8]
0x3A13	X_OUTPUT_OFFSET	0x02	R/W	Bit[7:0]:x_output_offset[7:0]
0x3A14	Y_OUTPUT_OFFSET	0x00	R/W	Bit[1:0]:y_output_offset[9:8]
0x3A15	Y_OUTPUT_OFFSET	0x04	R/W	Bit[7:0]:y_output_offset[7:0]
0x3A16	X_ODD_INC	0x01	R/W	Bit[4:0]:x_odd_inc
0x3A17	X_EVEN_INC	0x01	R/W	Bit[4:0]:x_even_inc
0x3A18	Y_ODD_INC	0x01	R/W	Bit[4:0]:y_odd_inc
0x3A19	Y_EVEN_INC	0x01	R/W	Bit[4:0]:y_even_inc
0x3A1A	FRMTIME_DELTA	0x03	R/W	Bit[3:0]:frmtime_delta
0x3A1B	TIMING_CTRL3	0x00	R/W	Bit[4]:RollingGlobalEn Bit[0]:fpship
0x3A1C	READC	0x00	R/W	Bit[1]:mirror_pr Bit[0]:updown_pr
0x3A21	V_BIN2	0x00	R/W	Bit[1]:v_bin2_mono_en Bit[0]:v_bin2_color_en
0x3A22	H_BIN2	0x00	R/W	Bit[1]:h_bin2_mono_en Bit[0]:h_bin2_color_en
0x3A23	ROWTIME_SHT	0x02	R/W	Bit[4:0]:rowtime_sht[12:8]=RollingGlobalEn?r2_ctrl23[12:8]:rowtime[12:8];
0x3A24	ROWTIME_SHT	0x00	R/W	Bit[7:0]:rowtime_sht[7:0]=RollingGlobalEn?r2_ctrl23[7:0]:rowtime[7:0];
0x3A25	DAC_VOS_CODE_RPC_CTRL0	0x33	R/W	Bit[3:0]:DAC_VOS_codeH_rpc4
0x3A26	DAC_VOS_CODE_RPC_CTRL1	0x33	R/W	Bit[7:4]:DAC_VOS_codeH_rpc3 Bit[3:0]:DAC_VOS_codeH_rpc2
0x3A27	DAC_VOS_CODE_RPC_CTRL2	0x33	R/W	Bit[7:4]:DAC_VOS_codeH_rpc1 Bit[3:0]:DAC_VOS_codeH_rpc0
0x3A28	DAC_VOS_CODE_RPC_CTRL3	0x33	R/W	Bit[7:4]:DAC_VOS_codeM_rpc4 Bit[3:0]:DAC_VOS_codeM_rpc3
0x3A29	DAC_VOS_CODE_RPC_CTRL4	0x33	R/W	Bit[7:4]:DAC_VOS_codeM_rpc2 Bit[3:0]:DAC_VOS_codeM_rpc1
0x3A2A	DAC_VOS_CODE_RPC_CTRL5	0x33	R/W	Bit[7:4]:DAC_VOS_codeM_rpc0 Bit[3:0]:DAC_VOS_codeL_rpc4
0x3A2B	DAC_VOS_CODE_RPC_CTRL6	0x33	R/W	Bit[7:4]:DAC_VOS_codeL_rpc3 Bit[3:0]:DAC_VOS_codeL_rpc2
0x3A2C	DAC_VOS_CODE_RPC_CTRL7	0x33	R/W	Bit[7:4]:DAC_VOS_codeL_rpc1 Bit[3:0]:DAC_VOS_codeL_rpc0
0x3A2D	DAC_CODERANGE_RPC_CTRL0	0x44	R/W	Bit[3:0]:DAC_CoderangeH_rpc4
0x3A2E	DAC_CODERANGE_RPC_CTRL1	0x44	R/W	Bit[7:4]:DAC_CoderangeH_rpc3 Bit[3:0]:DAC_CoderangeH_rpc2

0x3A2F	DAC_CODERANGE_RPC_CTRL2	0x44	R/W	Bit[7:4]:DAC_CoderangeH_rpc1 Bit[3:0]:DAC_CoderangeH_rpc0
0x3A30	DAC_CODERANGE_RPC_CTRL3	0x44	R/W	Bit[7:4]:DAC_CoderangeM_rpc4 Bit[3:0]:DAC_CoderangeM_rpc3
0x3A31	DAC_CODERANGE_RPC_CTRL4	0x44	R/W	Bit[7:4]:DAC_CoderangeM_rpc2 Bit[3:0]:DAC_CoderangeM_rpc1
0x3A32	DAC_CODERANGE_RPC_CTRL5	0x44	R/W	Bit[7:4]:DAC_CoderangeM_rpc0 Bit[3:0]:DAC_CoderangeL_rpc4
0x3A33	DAC_CODERANGE_RPC_CTRL6	0x44	R/W	Bit[7:4]:DAC_CoderangeL_rpc3 Bit[3:0]:DAC_CoderangeL_rpc2
0x3A34	DAC_CODERANGE_RPC_CTRL7	0x44	R/W	Bit[7:4]:DAC_CoderangeL_rpc1 Bit[3:0]:DAC_CoderangeL_rpc0
0x3A35	DAC_CUT_RST_CODE_RPC_CTRL0	0x08	R/W	Bit[5:0]:DAC_CUT_rst_codeH_rpc4
0x3A36	DAC_CUT_RST_CODE_RPC_CTRL1	0x08	R/W	Bit[5:0]:DAC_CUT_rst_codeH_rpc3
0x3A37	DAC_CUT_RST_CODE_RPC_CTRL2	0x08	R/W	Bit[5:0]:DAC_CUT_rst_codeH_rpc2
0x3A38	DAC_CUT_RST_CODE_RPC_CTRL3	0x08	R/W	Bit[5:0]:DAC_CUT_rst_codeH_rpc1
0x3A39	DAC_CUT_RST_CODE_RPC_CTRL4	0x08	R/W	Bit[5:0]:DAC_CUT_rst_codeH_rpc0
0x3A3A	DAC_CUT_RST_CODE_RPC_CTRL5	0x08	R/W	Bit[5:0]:DAC_CUT_rst_codeM_rpc4
0x3A3B	DAC_CUT_RST_CODE_RPC_CTRL6	0x08	R/W	Bit[5:0]:DAC_CUT_rst_codeM_rpc3
0x3A3C	DAC_CUT_RST_CODE_RPC_CTRL7	0x08	R/W	Bit[5:0]:DAC_CUT_rst_codeM_rpc2
0x3A3D	DAC_CUT_RST_CODE_RPC_CTRL8	0x08	R/W	Bit[5:0]:DAC_CUT_rst_codeM_rpc1
0x3A3E	DAC_CUT_RST_CODE_RPC_CTRL9	0x08	R/W	Bit[5:0]:DAC_CUT_rst_codeM_rpc0
0x3A3F	DAC_CUT_RST_CODE_RPC_CTRL10	0x18	R/W	Bit[5:0]:DAC_CUT_rst_codeL_rpc4
0x3A40	DAC_CUT_RST_CODE_RPC_CTRL11	0x18	R/W	Bit[5:0]:DAC_CUT_rst_codeL_rpc3
0x3A41	DAC_CUT_RST_CODE_RPC_CTRL12	0x18	R/W	Bit[5:0]:DAC_CUT_rst_codeL_rpc2
0x3A42	DAC_CUT_RST_CODE_RPC_CTRL13	0x18	R/W	Bit[5:0]:DAC_CUT_rst_codeL_rpc1
0x3A43	DAC_CUT_RST_CODE_RPC_CTRL14	0x18	R/W	Bit[5:0]:DAC_CUT_rst_codeL_rpc0
0x3A44	DAC_CUT_SIG_CODE_RPC_CTRL0	0x18	R/W	Bit[5:0]:DAC_CUT_sig_codeH_rpc4

0x3A45	DAC_CUT_SIG_CODE_RPC_CTRL1	0x18	R/W	Bit[5:0]:DAC_CUT_sig_codeH_rpc3
0x3A46	DAC_CUT_SIG_CODE_RPC_CTRL2	0x18	R/W	Bit[5:0]:DAC_CUT_sig_codeH_rpc2
0x3A47	DAC_CUT_SIG_CODE_RPC_CTRL3	0x18	R/W	Bit[5:0]:DAC_CUT_sig_codeH_rpc1
0x3A48	DAC_CUT_SIG_CODE_RPC_CTRL4	0x18	R/W	Bit[5:0]:DAC_CUT_sig_codeH_rpc0
0x3A49	DAC_CUT_SIG_CODE_RPC_CTRL5	0x18	R/W	Bit[5:0]:DAC_CUT_sig_codeM_rpc4
0x3A4A	DAC_CUT_SIG_CODE_RPC_CTRL6	0x18	R/W	Bit[5:0]:DAC_CUT_sig_codeM_rpc3
0x3A4B	DAC_CUT_SIG_CODE_RPC_CTRL7	0x18	R/W	Bit[5:0]:DAC_CUT_sig_codeM_rpc2
0x3A4C	DAC_CUT_SIG_CODE_RPC_CTRL8	0x18	R/W	Bit[5:0]:DAC_CUT_sig_codeM_rpc1
0x3A4D	DAC_CUT_SIG_CODE_RPC_CTRL9	0x18	R/W	Bit[5:0]:DAC_CUT_sig_codeM_rpc0
0x3A4E	DAC_CUT_SIG_CODE_RPC_CTRL10	0x08	R/W	Bit[5:0]:DAC_CUT_sig_codeL_rpc4
0x3A4F	DAC_CUT_SIG_CODE_RPC_CTRL11	0x08	R/W	Bit[5:0]:DAC_CUT_sig_codeL_rpc3
0x3A50	DAC_CUT_SIG_CODE_RPC_CTRL12	0x08	R/W	Bit[5:0]:DAC_CUT_sig_codeL_rpc2
0x3A51	DAC_CUT_SIG_CODE_RPC_CTRL13	0x08	R/W	Bit[5:0]:DAC_CUT_sig_codeL_rpc1
0x3A52	DAC_CUT_SIG_CODE_RPC_CTRL14	0x08	R/W	Bit[5:0]:DAC_CUT_sig_codeL_rpc0

7.6. EXTERC_regf (0x3Dxx)

表 7 - 6 EXTERC_regf

Address	Register name	Default	R/W	Description
0x3D00	FSYNC_CTRL0	0x00	R/W	Bit[7:4]:FSYNC_M_polarity Bit[0]:FSYNC_out_sel
0x3D01	FSYNC_CTRL1	0x01	R/W	Bit[4]:FSYNC_S_polarity Bit[1:0]:FSYNC_S_filter
0x3D02	FSYNC_CTRL2	0x00	R/W	Bit[7:0]:FSYNC_mode
0x3D03	FSYNC_CTRL3	0x0A	R/W	Bit[7:0]:FSYNC_M_width
0x3D04	FSYNC_CTRL4	0x01	R/W	Bit[7:0]:FSYNC_M_interval
0x3D05	FSYNC_CTRL5	0x00	R/W	Bit[7:0]:FSYNC_M_position[15:8]
0x3D06	FSYNC_CTRL6	0x10	R/W	Bit[7:0]:FSYNC_M_position
0x3D07	FSYNC_CTRL7	0x00	R/W	Bit[7:0]:FSYNC_S_position[15:8]

0x3D08	FSYNC_CTRL8	0x10	R/W	Bit[7:0]:FSYNC_S_position
0x3D09	FSYNC_CTRL9	0x03	R/W	Bit[7:0]:FSYNC_S_tol
0x3D0A	FSYNC_CTRL10	0x21	R/W	Bit[7:0]:FSYNC_S_commc
0x3D0B	EXT_CTRL0	0x01	R/W	Bit[1:0]:ext_exp_flter
0x3D0C	EXT_CTRL1	0x00	R/W	Bit[7:0]:ext_exp_offset
0x3D0D	EXT_CTRL2	0x00	R/W	Bit[5:4]:exter_exp_mode Bit[0]:exter_exp_en
0x3D10	STROBE_CTRL0	0x00	R/W	Bit[7]:STROBE_en Bit[6]:STROBE_polarity Bit[5:4]:STROBE_X_wid Bit[2:0]:STROBE_mode
0x3D11	STROBE_CTRL1	0x00	R/W	Bit[7:0]:STROBE_dmy_lines[15:8]
0x3D12	STROBE_CTRL2	0x0A	R/W	Bit[7:0]:STROBE_dmy_lines
0x3D13	STROBE_CTRL3	0x01	R/W	Bit[3]:STROBE_start_sel Bit[1:0]:STROBE_frm_latency
0x3D14	STROBE_CTRL4	0x04	R/W	Bit[7:0]:STROBE_row_latency
0x3D15	STROBE_CTRL5	0x00	R/W	Bit[0]:STROBE_req

7.7. OTPC_regf (0x3Exx)

表 7 - 7 OTPC_regf

Address	Register name	Default	R/W	Description
0x3E00	addr_pgm	0x00	R/W	Bit[7:0]:SingleProgramaddress(0x00~0xff)
0x3E01	data_pgm	0x00	R/W	Bit[7:0]:SingleProgramdata
0x3E02	addr_read	0x00	R/W	Bit[7:0]:SingleLoad(Read)address(0x00~0xff)
0x3E03	otp_ctrl	0x00	R/W	Bit[2]:otp_dm Bit[1]:start_read Bit[0]:start_pgm
0x3E04	otp_p0	0x01	R/W	Bit[7:0]:otp_p0
0x3E05	otp_p1	0x02	R/W	Bit[7:0]:otp_p1
0x3E06	otp_p2	0x13	R/W	Bit[4:0]:otp_p2[12:8]
0x3E07	otp_p2	0xB0	R/W	Bit[7:0]:otp_p2[7:8]
0x3E08	otp_p3	0x02	R/W	Bit[7:0]:otp_p3
0x3E09	otp_p4	0x02	R/W	Bit[7:0]:otp_p4
0x3E0A	otp_p5	0x0C	R/W	Bit[7:0]:otp_p5
0x3E0B	otp_p6	0x02	R/W	Bit[7:0]:otp_p6
0x3E0C	otp_busy	0x00	R	Bit[0]:OTPisbusy
0x3E0D	data_read	0x00	R	Bit[7:0]:SingleLoad(Read)data
0x3E10	otp_ctrl_batch	0x00	R/W	Bit[2]:batch_pgm_bist Bit[1]:start_batch_read Bit[0]:start_batch_pgm
0x3E11	otp_batch_start	0x00	R/W	Bit[7:0]:otpbatchprogram/loadstartaddress(0x00~0xff)

0x3E12	otp_batch_end	0xff	R/W	Bit[7:0]:otpbatchprogram/loadendaddress(0x00~0xff)
0x3E13	otp_dpc_start	0x10	R/W	Bit[7:0]:otpdpcclusterinfostartaddress(0x00~0xff)
0x3E14	otp_dpc_end	0xff	R/W	Bit[7:0]:otpdpcclusterinfoendaddress(0x00~0xff)
0x3E15	otp_batch_gap	0x0c	R/W	Bit[7:0]:otpbatchprogram/loadgapdistance
0x3E16	otp_ratio_addr	0x00	R/W	Bit[7:0]:CGratiocontentstartaddr
0x3E17	otp_HML_ratio_reg_en	0x00	R/W	Bit[4]:otp_HML_ratio_reg_en Bit[0]:otp_HL_ratio_reg[16]
0x3E18	otp_HL_ratio_reg	0x00	R/W	Bit[7:0]:otp_HL_ratio_reg[15:8]
0x3E19	otp_HL_ratio_reg	0x00	R/W	Bit[7:0]:otp_HL_ratio_reg[7:0]
0x3E1A	otp_HM_ratio_reg	0x00	R/W	Bit[3:0]:otp_HM_ratio_reg[11:8]
0x3E1B	otp_HM_ratio_reg	0x00	R/W	Bit[7:0]:otp_HM_ratio_reg[7:0]

7.8. BLC_regf (0x40xx)

表 7 - 8 BLC_regf

Address	Register name	Default	R/W	Description
0x4000	BLC_CTRL00	0x00	R/W	Bit[0]:blc_en
0x4001	BLC_CTRL01	0x00	R/W	Bit[4]:fixed_color
0x4002	BLC_AREA_SELECT	0x00	R/W	Bit[4]:ob_hselBit[0]:ob_csel
0x4003	BLC_AREA_ROW_START	0x00	R/W	Bit[2:0]:ob_hstart
0x4004	BLC_AREA_ROW_END	0x07	R/W	Bit[2:0]:ob_hend
0x4005	BLC_AREA_COL_START	0x00	R/W	Bit[0]:ob_cstart[8]
0x4006	BLC_AREA_COL_START	0x00	R/W	Bit[7:0]:ob_cstart[7:0]
0x4007	BLC_AREA_COL_END	0x01	R/W	Bit[0]:ob_cend[8]
0x4008	BLC_AREA_COL_END	0x43	R/W	Bit[7:0]:ob_cend[7:0]
0x4010	BLC_MF_THRES	0x08	R/W	Bit[4:0]:mf_abs_differ[12:8]
0x4011	BLC_MF_THRES	0x00	R/W	Bit[7:0]:mf_abs_differ[7:0]
0x4012	BLC_CTRL02	0x00	R/W	Bit[0]:ob2_en
0x4013	BLC_CTRL03	0x00	R/W	Bit[1:0]:a_ma
0x4014	BLC_CTRL04	0x00	R/W	Bit[0]:freeze
0x4015	BLC_CTRL05	0x00	R/W	Bit[0]:ob_en
0x4016	RANDOM_SELECT	0x00	R/W	Bit[1:0]:rand_en
0x4017	TRIGGER_SWITCH	0xff	R/W	Bit[7:0]:trig_sw
0x4018	OB_FRAME_DLY	0x00	R/W	Bit[7:0]:ob_frame_dly
0x4019	OB_FRAME_SUC	0x00	R/W	Bit[7:0]:ob_frame_suc
0x401a	DATA_LOW_LIMIT	0x00	R/W	Bit[0]:data_low_limit

0x4020	BLC_RPC_HCG	0x00	R/W	Bit[7:0]:blc_rpc_fr0[7:0]
0x4021	BLC_RPC_MCG	0x00	R/W	Bit[7:0]:blc_rpc_fr1[7:0]
0x4022	BLC_RPC_LCG	0x00	R/W	Bit[7:0]:blc_rpc_fr2[7:0]
0x4023	OFFSET_CTRL0	0x00	R/W	Bit[2:0]:offset_fr0[10:8]
0x4024	OFFSET_CTRL0	0x00	R/W	Bit[7:0]:offset_fr0[7:0]
0x4025	OFFSET_CTRL1	0x00	R/W	Bit[2:0]:offset_fr1[10:8]
0x4026	OFFSET_CTRL1	0x00	R/W	Bit[7:0]:offset_fr1[7:0]
0x4027	OFFSET_CTRL2	0x00	R/W	Bit[2:0]:offset_fr210:8]
0x4028	OFFSET_CTRL2	0x00	R/W	Bit[7:0]:offset_fr2[7:0]
0x4029	BLC_DC_LIMIT	0x03	R/W	Bit[1:0]:blc_dc_limit[9:8]
0x402a	BLC_DC_LIMIT	0xff	R/W	Bit[7:0]:blc_dc_limit[7:0]
0x4030	BLC_K_CH0_HCG	0x01	R/W	Bit[0]:k_ch0_fr0[8]
0x4031	BLC_K_CH0_HCG	0x00	R/W	Bit[7:0]:k_ch0_fr0[7:0]
0x4032	BLC_K_CH1_HCG	0x01	R/W	Bit[0]:k_ch1_fr0[8]
0x4033	BLC_K_CH1_HCG	0x00	R/W	Bit[7:0]:k_ch1_fr0[7:0]
0x4034	BLC_K_CH2_HCG	0x01	R/W	Bit[0]:k_ch2_fr0[8]
0x4035	BLC_K_CH2_HCG	0x00	R/W	Bit[7:0]:k_ch2_fr0[7:0]
0x4036	BLC_K_CH3_HCG	0x01	R/W	Bit[0]:k_ch3_fr0[8]
0x4037	BLC_K_CH3_HCG	0x00	R/W	Bit[7:0]:k_ch3_fr0[7:0]
0x4040	BLC_K_CH0_MCG	0x01	R/W	Bit[0]:k_ch0_fr1[8]
0x4041	BLC_K_CH0_MCG	0x00	R/W	Bit[7:0]:k_ch0_fr1[7:0]
0x4042	BLC_K_CH1_MCG	0x01	R/W	Bit[0]:k_ch1_fr1[8]
0x4043	BLC_K_CH1_MCG	0x00	R/W	Bit[7:0]:k_ch1_fr1[7:0]
0x4044	BLC_K_CH2_MCG	0x01	R/W	Bit[0]:k_ch2_fr1[8]
0x4045	BLC_K_CH2_MCG	0x00	R/W	Bit[7:0]:k_ch2_fr1[7:0]
0x4046	BLC_K_CH3_MCG	0x01	R/W	Bit[0]:k_ch3_fr1[8]
0x4047	BLC_K_CH3_MCG	0x00	R/W	Bit[7:0]:k_ch3_fr1[7:0]
0x4050	BLC_K_CH0_LCG	0x01	R/W	Bit[0]:k_ch0_fr2[8]
0x4051	BLC_K_CH0_LCG	0x00	R/W	Bit[7:0]:k_ch0_fr2[7:0]
0x4052	BLC_K_CH1_LCG	0x01	R/W	Bit[0]:k_ch1_fr2[8]
0x4053	BLC_K_CH1_LCG	0x00	R/W	Bit[7:0]:k_ch1_fr2[7:0]
0x4054	BLC_K_CH2_LCG	0x01	R/W	Bit[0]:k_ch2_fr2[8]
0x4055	BLC_K_CH2_LCG	0x00	R/W	Bit[7:0]:k_ch2_fr2[7:0]
0x4056	BLC_K_CH3_LCG	0x01	R/W	Bit[0]:k_ch3_fr2[8]
0x4057	BLC_K_CH3_LCG	0x00	R/W	Bit[7:0]:k_ch3_fr2[7:0]
0x4060	BLC_CTRL06	0x32	R/W	Bit[7:0]:ob_ch0_fr0_ef
0x4061	BLC_CTRL07	0x32	R/W	Bit[7:0]:ob_ch1_fr0_ef
0x4062	BLC_CTRL08	0x32	R/W	Bit[7:0]:ob_ch2_fr0_ef
0x4063	BLC_CTRL09	0x32	R/W	Bit[7:0]:ob_ch3_fr0_ef
0x4064	BLC_CTRL10	0x32	R/W	Bit[7:0]:ob_ch0_fr1_ef
0x4065	BLC_CTRL11	0x32	R/W	Bit[7:0]:ob_ch1_fr1_ef
0x4066	BLC_CTRL12	0x32	R/W	Bit[7:0]:ob_ch2_fr1_ef
0x4067	BLC_CTRL13	0x32	R/W	Bit[7:0]:ob_ch3_fr1_ef

0x4068	BLC_CTRL14	0x32	R/W	Bit[7:0]:ob_ch0_fr2_ef
0x4069	BLC_CTRL15	0x32	R/W	Bit[7:0]:ob_ch1_fr2_ef
0x406a	BLC_CTRL16	0x32	R/W	Bit[7:0]:ob_ch2_fr2_ef
0x406b	BLC_CTRL17	0x32	R/W	Bit[7:0]:ob_ch3_fr2_ef
0x4070	BLC_CTRL18	0x32	R/W	Bit[7:0]:ob_ch0_fr0_st
0x4071	BLC_CTRL19	0x32	R/W	Bit[7:0]:ob_ch1_fr0_st
0x4072	BLC_CTRL20	0x32	R/W	Bit[7:0]:ob_ch2_fr0_st
0x4073	BLC_CTRL21	0x32	R/W	Bit[7:0]:ob_ch3_fr0_st
0x4074	BLC_CTRL22	0x32	R/W	Bit[7:0]:ob_ch0_fr1_st
0x4075	BLC_CTRL23	0x32	R/W	Bit[7:0]:ob_ch1_fr1_st
0x4076	BLC_CTRL24	0x32	R/W	Bit[7:0]:ob_ch2_fr1_st
0x4077	BLC_CTRL25	0x32	R/W	Bit[7:0]:ob_ch3_fr1_st
0x4078	BLC_CTRL26	0x32	R/W	Bit[7:0]:ob_ch0_fr2_st
0x4079	BLC_CTRL27	0x32	R/W	Bit[7:0]:ob_ch1_fr2_st
0x407a	BLC_CTRL28	0x32	R/W	Bit[7:0]:ob_ch2_fr2_st
0x407b	BLC_CTRL29	0x32	R/W	Bit[7:0]:ob_ch3_fr2_st
0x4080	TEXP_EXT_CTRL0	0x00	R/W	Bit[0]:texp_ext_en
0x4081	TEXP_EXT_CTRL1	0x00	R/W	Bit[4:0]:texp_ext[28:24]
0x4082	TEXP_EXT_CTRL2	0x00	R/W	Bit[7:0]:texp_ext[23:16]
0x4083	TEXP_EXT_CTRL3	0x00	R/W	Bit[7:0]:texp_ext[15:8]
0x4084	TEXP_EXT_CTRL4	0x01	R/W	Bit[7:0]:texp_ext[7:0]
0x4090	ADJ_CRTLH	0x00	R/W	Bit[4:0]:adjH[12:8]
0x4091	ADJ_CRTLH	0x00	R/W	Bit[7:0]:adjH[7:0]
0x4092	ADJ_CRTLM	0x00	R/W	Bit[4:0]:adjM[12:8]
0x4093	ADJ_CRTLM	0x00	R/W	Bit[7:0]:adjM[7:0]
0x4094	ADJ_CRTLL	0x00	R/W	Bit[4:0]:adjL[12:8]
0x4095	ADJ_CRTLL	0x00	R/W	Bit[7:0]:adjL[7:0]
0x40a0	BLC_OB_READ00		R	Bit[7:0]:ob_ch0_fr0_use[15:8]
0x40a1	BLC_OB_READ00		R	Bit[7:0]:ob_ch0_fr0_use[7:0]
0x40a2	BLC_OB_READ01		R	Bit[7:0]:ob_ch1_fr0_use[15:8]
0x40a3	BLC_OB_READ01		R	Bit[7:0]:ob_ch1_fr0_use[7:0]
0x40a4	BLC_OB_READ02		R	Bit[7:0]:ob_ch2_fr0_use[15:8]
0x40a5	BLC_OB_READ02		R	Bit[7:0]:ob_ch2_fr0_use[7:0]
0x40a6	BLC_OB_READ03		R	Bit[7:0]:ob_ch3_fr0_use[15:8]
0x40a7	BLC_OB_READ03		R	Bit[7:0]:ob_ch3_fr0_use[7:0]
0x40b0	BLC_OB_READ04		R	Bit[7:0]:ob_ch0_fr1_use[15:8]
0x40b1	BLC_OB_READ04		R	Bit[7:0]:ob_ch0_fr1_use[7:0]
0x40b2	BLC_OB_READ05		R	Bit[7:0]:ob_ch0_fr1_use[15:8]
0x40b3	BLC_OB_READ05		R	Bit[7:0]:ob_ch1_fr1_use[7:0]
0x40b4	BLC_OB_READ06		R	Bit[7:0]:ob_ch2_fr1_use[15:8]
0x40b5	BLC_OB_READ06		R	Bit[7:0]:ob_ch2_fr1_use[7:0]
0x40b6	BLC_OB_READ07		R	Bit[7:0]:ob_ch3_fr1_use[15:8]
0x40b7	BLC_OB_READ07		R	Bit[7:0]:ob_ch3_fr1_use[7:0]

0x40c0	BLC_OB_READ08		R	Bit[7:0]:ob_ch0_fr2_use[15:8]
0x40c1	BLC_OB_READ08		R	Bit[7:0]:ob_ch0_fr2_use[7:0]
0x40c2	BLC_OB_READ09		R	Bit[7:0]:ob_ch1_fr2_use[15:8]
0x40c3	BLC_OB_READ09		R	Bit[7:0]:ob_ch1_fr2_use[7:0]
0x40c4	BLC_OB_READ10		R	Bit[7:0]:ob_ch2_fr2_use[15:8]
0x40c5	BLC_OB_READ10		R	Bit[7:0]:ob_ch2_fr2_use[7:0]
0x40c6	BLC_OB_READ11		R	Bit[7:0]:ob_ch3_fr2_use[15:8]
0x40c7	BLC_OB_READ11		R	Bit[7:0]:ob_ch3_fr2_use[7:0]

7.9. SLVSC_regf (0x47xx)

表 7 - 9 SLVSC_regf

Address	Register name	Default	R/W	Description
0x4700	SLVS_SAV0	0xab	R/W	Bit[7:0]:sav0
0x4701	SLVS_EAV0	0xb6	R/W	Bit[7:0]:eav0
0x4702	SLVS_SAV1	0xac	R/W	Bit[7:0]:sav1
0x4703	SLVS_EAV1	0xb7	R/W	Bit[7:0]:eav1
0x4704	SLVS_SAV2	0x80	R/W	Bit[7:0]:sav2
0x4705	SLVS_EAV2	0x9d	R/W	Bit[7:0]:eav2
0x4706	SLVS_SAV3	0x81	R/W	Bit[7:0]:sav3
0x4707	SLVS_EAV3	0x9e	R/W	Bit[7:0]:eav3
0x4708	SLVS_SAV4	0x82	R/W	Bit[7:0]:sav4
0x4709	SLVS_EAV4	0x9f	R/W	Bit[7:0]:eav4
0x470a	SLVS_SAV5	0x83	R/W	Bit[7:0]:sav5
0x470b	SLVS_EAV5	0xa0	R/W	Bit[7:0]:eav5
0x470c	SLVS_SAV6	0x84	R/W	Bit[7:0]:sav6
0x470d	SLVS_EAV6	0xa1	R/W	Bit[7:0]:eav6
0x470e	SLVS_SAV7	0x85	R/W	Bit[7:0]:sav7
0x470f	SLVS_SAV7	0xa2	R/W	Bit[7:0]:eav7
0x4710	SLVS_LANE_NUM	0x02	R/W	Bit[1:0]:slvs_lane_num

7.10. MIPI_regf (0x48xx)

表 7 - 10 MIPI_regf

Address	Register name	Default	R/W	Description
0x4811	H_SIZE_MIPI	0x05	R/W	Bit[3:0]:h_size_mipi[11:8]
0x4812	H_SIZE_MIPI	0x00	R/W	Bit[7:0]:h_size_mipi[7:0]
0x4824	LANE_NUM	0x01	R/W	Bit[1:0]:Lane_num
0x4825	MIPI_DLANE_EN	0x00	R/W	Bit[2:0]:MIPI_dlane_en
0x4826	MIPI_CK_SKEW	0x00	R/W	Bit[1:0]:MIPI_ck_skew
0x482f	MIPI_PHY_EN	0x00	R/W	Bit[1]:MIPI_phy_en_LP Bit[0]:MIPI_phy_en_Nor

0x4830	MIPI_EN	0x00	R/W	Bit[2]:slvs_en_pr Bit[1]:mipi_en_update_mode Bit[0]:mipi_en_pr
0x4831	MIPI_EN_DLY	0x50	R/W	Bit[7:0]:mipi_en_dly[8:1] mipi_en_dly[0]=1'b0

7.11. ISPC_regf (0x50xx)

表 7 - 11 ISPC_regf

Address	Register name	Default	R/W	Description
0x5000	TEST_EN	0x00	R/W	Bit[0]:test_en
0x5001	FIRSTB_CTRL	0x00	R/W	Bit[1]:firstB_en_row Bit[0]:firstB_en_col
0x5002	DOUT_MODE	0x00	R/W	Bit[1:0]:dout_mode

7.12. OTP_DPC_D4_regf (0x5100~0x517f)

表 7 - 12 OTP_DPC_D4_regf

Address	Register name	Default	R/W	Description
0x5100	otp_dpc_en	0x00	R/W	Bit[2]:otp_dpc_en_LCG Bit[1]:otp_dpc_en_MCG Bit[0]:otp_dpc_en_HCG
0x5104	r_ctrl04	0x00	R/W	Bit[4]:man_inc_en Bit[3]:disable_mf, disable mirror/flip Bit[2]:disable_offset,==1,offset set by regsiter; ==0,offset set automatically. Bit[1]:mirror_otp Bit[0]:disable_bin
0x5105	r_ctrl05	0x68	R/W	Bit[7]:sub_bin_method_en Bit[6:5]:recov_method Bit[4]:fixed_replace Bit[3]:fixed_ptn Bit[2]:flip_otp Bit[1]:expo_en Bit[0]:gain_en
0x5106	expo_constraint_HCG	0x00	R/W	Bit[6:0]:expo_constraint_HCG[14:8]
0x5107	expo_constraint_HCG	0x00	R/W	Bit[7:0]:expo_constraint_HCG[7:0]
0x5108	gain_constraint_HCG	0x07	R/W	Bit[7:0]:gain_constraint_HCG[7:0]
0x5109	r_ctrl09	0x48	R/W	Bit[7]:xy_end_sel Bit[6]:vsync_rst_en Bit[4]:thre_en

				Bit[3:0]:thre_HCG
0x510A	man_x_even_inc	0x01	R/W	Bit[4:0]:man_x_even_inc
0x510B	man_x_odd_inc	0x01	R/W	Bit[4:0]:man_x_odd_inc
0x510C	man_y_even_inc	0x01	R/W	Bit[4:0]:man_y_even_inc
0x510D	man_y_odd_inc	0x01	R/W	Bit[4:0]:man_y_odd_inc
0x5110	x_offset_eff	0x00	R	Bit[7:0]x_offset_eff[IH_SW-1:8]
0x5111	x_offset_eff	0x00	R	Bit[7:0]x_offset_eff[7:0]
0x5112	y_offset_eff	0x00	R	Bit[7:0]y_offset_eff[IV_SW-1:8]
0x5113	y_offset_eff	0x00	R	Bit[7:0]y_offset_eff[7:0]
0x5114	man_x_even_inc_eff	0x00	R	Bit[4:0]:man_x_even_inc_eff
0x5115	man_x_odd_inc_eff	0x00	R	Bit[4:0]:man_x_odd_inc_eff
0x5116	man_y_even_inc_eff	0x00	R	Bit[4:0]:man_y_even_inc_eff
0x5117	man_y_odd_inc_eff	0x00	R	Bit[4:0]:man_y_odd_inc_eff
0x5120	man_x_offset	0x00	R/W	Bit[4:0]:man_x_offset[12:8]
0x5121	man_x_offset	0x00	R/W	Bit[7:0]:man_x_offset[7:0]
0x5122	man_y_offset	0x00	R/W	Bit[4:0]:man_y_offset[12:8]
0x5123	man_y_offset	0x00	R/W	Bit[7:0]:man_y_offset[7:0]
0x5126	r_ctrl26	0x00	R/W	Bit[7]:y_bin_man_en Bit[6]:y_bin4_man Bit[5]:y_bin3_man Bit[4]:y_bin2_man Bit[3]:x_bin_man_en Bit[2]:x_bin4_man Bit[1]:x_bin3_man Bit[0]:x_bin2_man
0x5128	expo_constraint_MCG	0x00	R/W	Bit[6:0]:expo_constraint_MCG[14:8]
0x5129	expo_constraint_MCG	0x00	R/W	Bit[7:0]:expo_constraint_MCG[7:0]
0x512A	gain_constraint_MCG	0x07	R/W	Bit[7:0]:gain_constraint_MCG[7:0]
0x512B	thre_MCG	0x08	R/W	Bit[3:0]:thre_MCG[3:0]
0x512C	expo_constraint_LCG	0x00	R/W	Bit[6:0]:expo_constraint_LCG[14:8]
0x512D	expo_constraint_LCG	0x00	R/W	Bit[7:0]:expo_constraint_LCG[7:0]
0x512E	gain_constraint_LCG	0x07	R/W	Bit[7:0]:gain_constraint_LCG[7:0]
0x512F	thre_LCG	0x08	R/W	Bit[3:0]:thre_LCG[3:0]

7.13. HDR_D4_regf (0x5180~0x51ff)

表 7 - 13 HDR_D4_regf

Address	Register name	Default	R/W	Description
0x5180	HDR_CTRL0	0x00	R/W	Bit[7:0]:r_ctrl00_o
0x5181	HDR_CTRL1	0x00	R/W	Bit[7:0]:r_ctrl01_o
0x5182	HDR_CTRL2	0x00	R/W	Bit[7:0]:r_ctrl02_o

0x5183	HDR_CTRL3	0x00	R/W	Bit[2]:MBIST_err Bit[1]:MBIST_done Bit[0]:r_ctrl03_o
0x5184	HDR_CTRL4	0x11	R/W	Bit[4:0]:r_ctrl04_o
0x5185	HDR_CTRL5	0xF0	R/W	Bit[7:0]:r_ctrl05_o
0x5186	HDR_CTRL6	0x60	R/W	Bit[7:0]:r_ctrl06_o
0x5187	HDR_CTRL7	0xBB	R/W	Bit[7:0]:r_ctrl07_o
0x5188	HDR_CTRL8	0x4B	R/W	Bit[7:0]:r_ctrl08_o
0x5189	HDR_CTRL9	0x00	R/W	Bit[4:0]:r_ctrl09_o
0x518A	HDR_CTRL10	0x03	R/W	Bit[4:0]:r_ctrl0a_o
0x518B	HDR_CTRL11	0x06	R/W	Bit[4:0]:r_ctrl0b_o
0x518C	HDR_CTRL12	0x09	R/W	Bit[4:0]:r_ctrl0c_o
0x518D	HDR_CTRL13	0x0C	R/W	Bit[4:0]:r_ctrl0d_o
0x518E	HDR_CTRL14	0x0E	R/W	Bit[4:0]:r_ctrl0e_o
0x518F	HDR_CTRL15	0x02	R/W	Bit[7:0]:r_ctrl0f_o
0x5190	HDR_CTRL16	0x56	R/W	Bit[7:0]:r_ctrl10_o
0x5191	HDR_CTRL17	0x04	R/W	Bit[7:0]:r_ctrl11_o
0x5192	HDR_CTRL18	0xF7	R/W	Bit[7:0]:r_ctrl12_o
0x5193	HDR_CTRL19	0x07	R/W	Bit[7:0]:r_ctrl13_o
0x5194	HDR_CTRL20	0xA2	R/W	Bit[7:0]:r_ctrl14_o
0x5195	HDR_CTRL21	0x0A	R/W	Bit[7:0]:r_ctrl15_o
0x5196	HDR_CTRL22	0x4E	R/W	Bit[7:0]:r_ctrl16_o
0x5197	HDR_CTRL23	0x0C	R/W	Bit[7:0]:r_ctrl17_o
0x5198	HDR_CTRL24	0x98	R/W	Bit[7:0]:r_ctrl18_o
0x5199	HDR_CTRL25	0x00	R/W	Bit[7:0]:r_ctrl19_o
0x519A	HDR_CTRL26	0x02	R/W	Bit[7:0]:r_ctrl1a_o
0x519B	HDR_CTRL27	0xAC	R/W	Bit[7:0]:r_ctrl1b_o
0x519C	HDR_CTRL28	0x00	R/W	Bit[7:0]:r_ctrl1c_o
0x519D	HDR_CTRL29	0x18	R/W	Bit[7:0]:r_ctrl1d_o
0x519E	HDR_CTRL30	0x0C	R/W	Bit[7:0]:r_ctrl1e_o
0x519F	HDR_CTRL31	0x00	R/W	Bit[7:0]:r_ctrl1f_o
0x51A0	HDR_CTRL32	0xC3	R/W	Bit[7:0]:r_ctrl20_o
0x51A1	HDR_CTRL33	0x0C	R/W	Bit[7:0]:r_ctrl21_o
0x51A2	HDR_CTRL34	0x06	R/W	Bit[7:0]:r_ctrl22_o
0x51A3	HDR_CTRL35	0x1B	R/W	Bit[7:0]:r_ctrl23_o
0x51A4	HDR_CTRL36	0x0C	R/W	Bit[7:0]:r_ctrl24_o
0x51A5	HDR_CTRL37	0x30	R/W	Bit[7:0]:r_ctrl25_o
0x51A6	HDR_CTRL38	0xDB	R/W	Bit[7:0]:r_ctrl26_o
0x51A7	HDR_CTRL39	0x0C	R/W	Bit[7:0]:r_ctrl27_o
0x51A8	HDR_CTRL40	0x00	R/W	Bit[6:0]:r_ctrl28_o
0x51A9	HDR_CTRL41	0x00	R/W	Bit[7:0]:r_ctrl29_o
0x51AA	HDR_CTRL42	0x00	R/W	Bit[7:0]:r_ctrl2a_o
0x51B0	HDR_CTRL43	0x02	R/W	Bit[3:0]:r_ctrl30_o

0x51B1	HDR_CTRL44	0xFF	R/W	Bit[7:0]:r_ctrl31_o

8. 文档修订历史

版本	修改内容说明	Owner/Date