General Description
Signal Description
Software Configuration
Exposure Control
Flashlight Control
CMOS Sensor Implementation Guide
Other Issue

OV5620/ MI8130 General Description

Key Specifications

Array Size		2592 x 1944
Power Supply	Analog	2.6 ~ 3.0V
	Digital	1.3V ± 5%
	I/O	1.7 ~ 3.3V
Power Requirements	Active	TBD
	Standby	250 µA
Electronics Exposure		1 T_{LINE} to 1/F where F = frame rate
Shutter		Electronic rolling shutter, snapshot
Output Format		10-bit digital RGB Raw data
Lens Size		1/2.5
Lens Chief Ray Angle		12.5°
Input Clock		6 - 27 MHz
Maximum System Clock		48 MHz
Maximum Data Rate		48 MHz
Max Image Transfer Rate	Full	7.5 fps
	1.3Mpixel	30 fps
	D1MD	60 fps
	VGA	60 fps
	QVGA	120 fps
Sensitivity		TBD
S/N Ratio		TBD
Dynamic Range		TBD
Scan Mode		Progressive
Pixel Size		2.2 µm x 2.2 µm
Dark Current		TBD
Fixed Pattern Noise		TBD
Image Area		5.808 mm x 4.294 mm
Package Dimensions		14.22 mm x 14.22 mm

Table 1: Key Performance Parameters

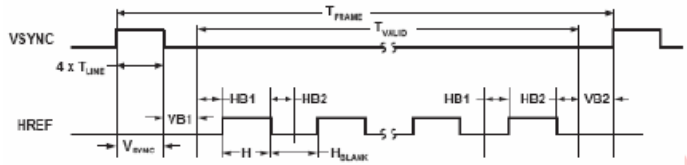
Parameter		Value
Optical format		1/2.5-inch (4:3)
Full resolution		3,264 x 2,448 pixels
Pixel size		1.75µm x 1.75µm
Chief ray angle		7.5 maximum
Color filter array		RGB Bayer pattern
Shutter type		Electronic rolling shutter (ERS) with global reset release (GRR)
Input clock frequency		6–48 MHz
Maximum data rate/ master clock		96 Mp/s
Frame rate	Full resolution	11 fps
	Video mode	30 fps
Supply voltage	Analog	2.5–3.1V (2.8V nominal)
	Digital	1.7V–1.9V (1.8 nominal)
	I/O	1.8V or 2.8V
	PLL	2.5V–2.9V (2.8V nominal)
ADC resolution		12-bit
Responsivity		0.3 V/lux-sec (at 550nm) (preliminary)
Dynamic range		70dB (preliminary)
SNR_{MAX}		38.9dB (preliminary)
Power consumption	Full resolution	<350mW (preliminary)
	Video mode	<200mW (preliminary)
	Standby	<10µW (preliminary)
Operating temperature		–30°C to +70°C (at junction)
Package		Bare die, 40-pin iLCC

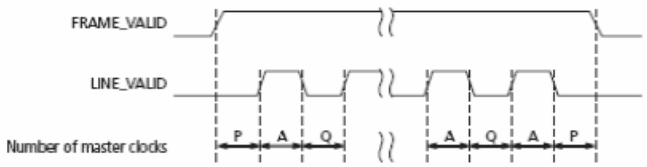
Signal Description

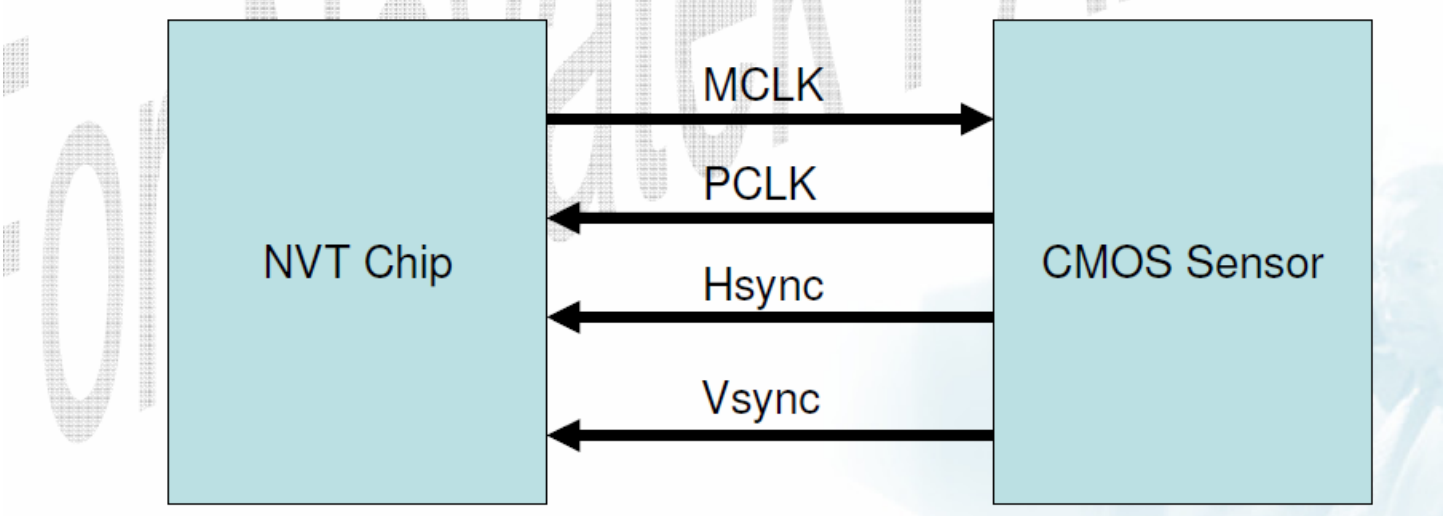
OV5620 Pin Description
1. PWDN (09): Power Down control, active high. (hardware standby)
2. RESET_B (11): Hardware reset, active low.
3. XVCLK(27): System clock input.
4. PCLK (40): Pixel clock output.
5. VSYNC (43): Vertical synchronization output.
6. HREF (44): Horizontal reference (data valid) output.
7. Y0:Y9 (13~16,34~39): Bit[0:9] of video output port.
8. SCL (45) / SDA (46): I2C clock/data.
9. FREX (10) / EXP_STB (12): Frame exposure control.

Pin Number	Name	Pin Type	Function/Description
01	DOVDD	Power	Power for I/O circuit (1.7V to 3.3V)
02	STROBE	Output	LED control output
03	RVDD	Power	Regulator power (2.8V)
04	VREF1	Analog	Internal reference - connect to ground using a 0.1 µF capacitor
05	VREF2	Analog	Internal reference - connect to ground using a 0.1 µF capacitor
06	NC	–	No connection
07	NC	–	No connection
08	NC	–	No connection
09	PWDN	Input (0)	Power down control, active high (hardware standby)
10	FREX	Input (0)	Frame exposure control 1
11	RESET_B	Input (1)	Hardware reset, active low
12	EXP_STB	Input (0)	Frame exposure control 2
13	Y0	Output	Bit[0] of video output port
14	Y1	Output	Bit[1] of video output port
15	Y2	Output	Bit[2] of video output port
16	Y3	Output	Bit[3] of video output port
17	NC	–	No connection
18	NC	–	No connection
19	NC	–	No connection
20	EVDD	Power	CCP2 power (2.8V)
21	CLK_P	Output	CCP2 positive clock output
22	CLK_N	Output	CCP2 negative clock output
23	DATA_P	Output	CCP2 interface positive data output
24	DATA_N	Output	CCP2 interface negative data output
25	EGND	Power	CCP2 ground
26	PVDD	Power	PLL power (2.8V)
27	XVCLK	Input	System clock input
28	DOGND	Power	Ground for I/O circuit
29	AVDD	Power	Analog power (2.8V)
30	DGND	Power	Digital ground
31	NC	–	No connection

Pin Number	Name	Pin Type	Function/Description
32	NC	–	No connection
33	DVDD	Power	Internal reference - connect to ground using a 0.1 µF capacitor or digital power (1.3V)
34	Y4	Output	Bit[4] of video output port
35	Y5	Output	Bit[5] of video output port
36	Y6	Output	Bit[6] of video output port
37	Y7	Output	Bit[7] of video output port
38	Y8	Output	Bit[8] of video output port
39	Y9	Output	Bit[9] of video output port
40	PCLK	Output	Pixel clock output
41	NC	–	No connection
42	NC	–	No connection
43	VSYNC	Output	Vertical synchronization output
44	HREF	Output	Horizontal reference (data valid) output
45	SCL	Input	I2C clock
46	SDA	I/O	I2C data
47	SVDD	Power	Analog power (2.8V)
48	AGND	Power	Analog ground



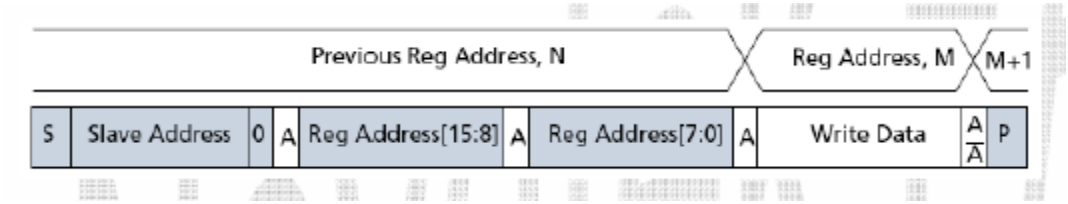




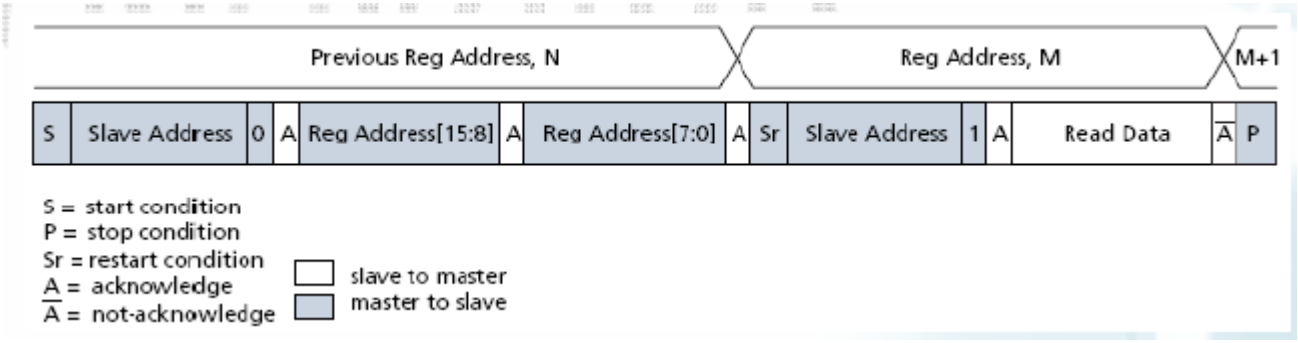
Software Configuration

Two-Wire Serial Register Interface (I2C/SPI)

1. Device Slave address (Parallel/MIPI/Hispi/CSI/LVDS)
2. Single WRITE from random address



3. Single READ to random address



Initial register

// ;XVCLK=24Mhz, SCLK=4x120Mhz, MIPI 640Mbps, DACCLK=240Mhz

```
  /*
     .width      = 4208,
     .height     = 3120,
     .hoffset    = 0,
     .voffset    = 0,
     .hts        = 9600/16,
     .vts        = 3328-6,
     .pclk       = (637*1000*1000+800*1000)/16,
     .mipi_bps    = 850*1000*1000,
     .fps_fixed  = 2,
     .bin_factor = 1,
  */
```

1. PLL setting
2. Control setting
3. Image processing setting
4. Reserved setting

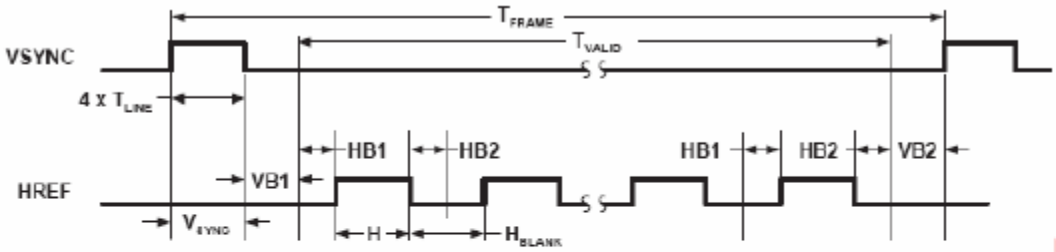
Preview/Capture Mode register

1. Vsync/Hsync length setting
2. Vertical/Horizontal output size and coordinate setting
3. Skipping/Binning ratio setting

Exposure register

1. Analog gain control register
2. Integration time register

OV5620 Mode setting



NOTES:

1. $T_{FRAME} = T_{VALID} + V_{BLANK}$
2. $H_{BLANK} = HB1 + HB2$
3. $T_{LINE} = H + H_{BLANK}$
4. $V_{BLANK} = VB1 + VB2 + V_{SYNC}$

Table 9 Control Parameters for Standard Resolution Output

Format	H_Size (pixels)	V_Size (pixels)	H_Bin	V_Bin	VB2	V_{SYNC}	VB1	HB1	HB2[a]	H_{BLANK}(pixels)	V_{BLANK} (T_{LINES})	Frame Rate[b] (fps)
5 Mpixel	2592	1944	1:1	1:1	0	4	20	192	468	660	24	7.5
1.3 Mpixel	1280	960	1:2	1:2	0	4	13	192	168	360	17	30
D1MD	864	600	1:3	1:3	0	4	13	192	244	436	17	60
QFMD[c]	1280	480	1:2	1:4	0	4	13	192	140	332	17	60
HF[d]	1280	240	1:2	1:8	0	4	13	192	88	280	17	120

Sensor mode：full size mode, binning mode, skip mode, crop mode

Analog gain control register

1. OV5620

Address (Hex)	Register Name	Default (Hex)	R/W	Description
00	GAIN	00	RW	AGC Gain Control Bit[7]: Reserved - must be set to '0' Bit[6:0]: Gain setting • Range: 1x to 16x Gain = (Bit[6]+1) x (Bit[5]+1) x (Bit[4]+1) x (1+Bit[3:0]/16) ***Note:*** *Set COM8[2] = 0 to disable AGC.*

2. MI8130

Reg. #	Bits	Default	Name	Frame Sync'd	Bad Frame
R12374 R0x3056	15:0	0x0234	green1_gain (RW)		
	15:12	X	Reserved		
	11:9	0x0001	Digital Gain Digital Gain. Legal values 1-7.	Y	N
	8:7	0x0000	Analog Gain Analog gain = (bit [8] + 1) * (bit [7] + 1) * initial gain.	Y	N
	6:0	0x0034	Initial Gain Initial gain = bits [6:0] * 1/32.	Y	N

$$gain = (<color>\_gain[8]+1)\times(<color>\_gain[7]+1)\times\frac{<color>\_gain[6:0]}{32}$$

Desired Gain	Recommended Gain Register Setting
1–1.969	0x0220–0x023F
2–7.9375	0x02A0–0x02FF
8–15.875	0x03C0–0x03FF

Exposure Control

Exposure control for Preview/Capture mode switch

1. Different binning/skipping mode compensation
2. Different Integration time calculation

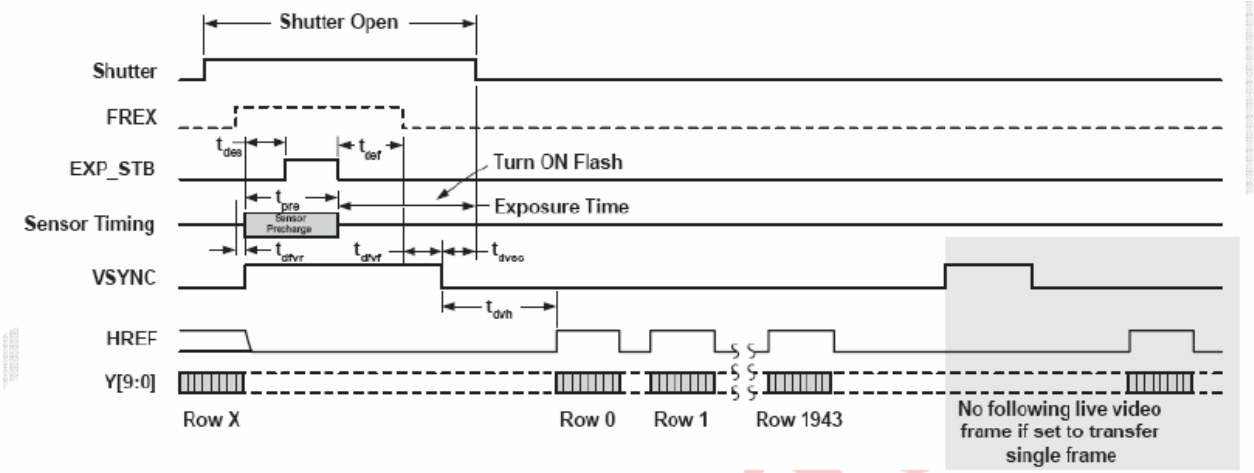
-- Hsync period

-- pixel clock

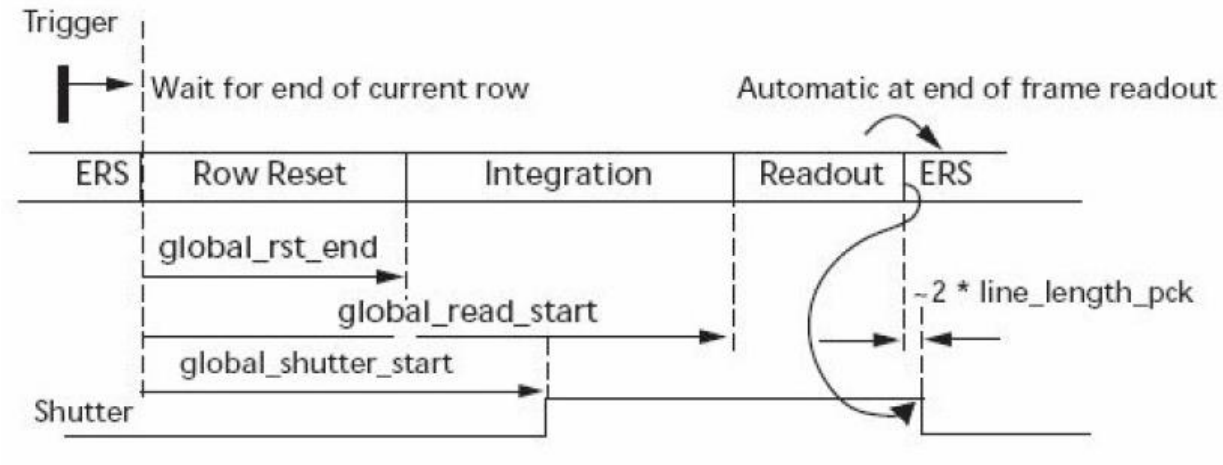
3.exposure mode(global shutter[CCD/CMOS] and rolling shutter[CMOS])

Frame (Global) exposure control with mechanical shutter

1. OV5620



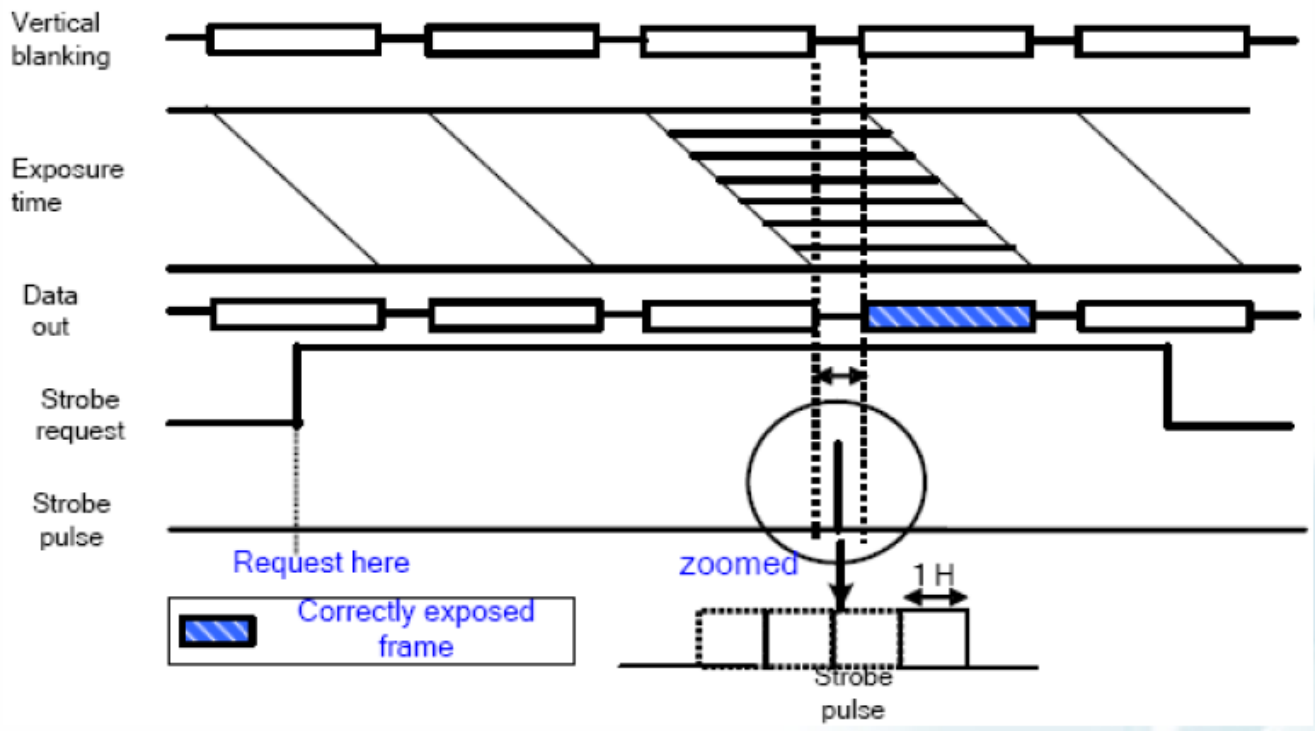
2. MI8130



Flashlight Control

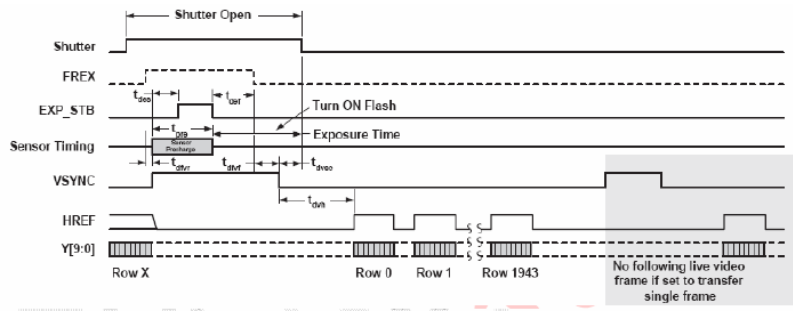
Electronic shutter mode

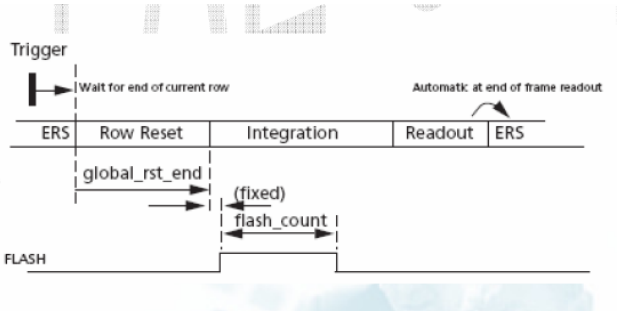
1. Integration time must be more than the line number of a full frame.
2. Flashlight should be triggered when all lines are integrating.



Frame (Global) exposure mode with mechanical shutter

1. Flashlight should be triggered during frame exposure.
2. CMOS sensor can control the flashlight trigger and pulse width.





CMOS Sensor Implementation Guide

Sensor Signal Checking
I2C Interface
Sensor Initial/Mode register setting
Sensor driver setting

Sensor Signal Checking
1. HW Power down pin
2. HW reset pin (power-up reset)
3. Vertical sync signal
4. Horizontal sync signal
5. External clock
6. Pixel clock
7. Data pin [0:9]

I2C Interface
1. I2C clock
2. I2C data
3. I2C device address

Sensor Initial/Mode register setting
1. Initial register setting
2. Preview mode setting (VGA 30 fps, PCLK, binning mode)
3. Capture mode setting (ERS/GRS control)

Sensor driver setting
1. Sensor Initial/Mode register setting
2. Sensor interface I2C command
3. SIE coordinate setting
4. Digital zoom table setting
5. Preview/Capture mode setting flow

Sensor Initial register setting

static SENSOR_CMD AR0238CSP_INI_REG[] = 《==目前基本不用，直接把所有 setting 放到 sensor mode

{

{0x301A, 2, {0x0001, 0x0000}},

…..

};

static SENSOR_CMD AR0238CSP_REG_MODE1[]=

{

    //rev 0.1 //2016-06-08

    //Linear Initialization

    {0x301A, 2, {0x0001, 0x0000}},

```
…
};
Sensor Interface I2C command
static ER WriteReg_OV4689(SENSOR_ID Id, SENSOR_CMD *Cmd)
{
    Sensor_I2C_Lock(Sensor_I2C[Id], Sensor_I2C_SESSION[Id]);
    Sensor_I2CSet_Transmit(Sensor_I2C[Id], Sensor_WRITE_ID[Id], Cmd->uiAddr, Cmd->uiData[0],
I2CFMT_2B1B); // static SENSOR_CMD OV4689_MODE14[] ={       {0x0103, 1, {0x01, 0x00}}, …};
    Sensor_I2C_Unlock(Sensor_I2C[Id], Sensor_I2C_SESSION[Id]);
    return E_OK;
}

static ER ReadReg_OV4689(SENSOR_ID Id, SENSOR_CMD *Cmd)
{
    Sensor_I2C_Lock(Sensor_I2C[Id], Sensor_I2C_SESSION[Id]);
    Sensor_I2CSet_Receive(Sensor_I2C[Id], Sensor_WRITE_ID[Id], Sensor_READ_ID[Id], Cmd->uiAddr,
&(Cmd->uiData[0]), I2CFMT_2B1B);
    Sensor_I2C_Unlock(Sensor_I2C[Id], Sensor_I2C_SESSION[Id]);
    return E_OK;
}
SIE coordinate setting
//parallel
static SENSOR_SIGNAL HD_IMX322LQJ[MODE_MAX + 1] =
{
    {0, 0, 0, 0},
    {1, 2200, 170, 1920},
    {1, 1650, 178, 1280},
};

static SENSOR_SIGNAL VD_IMX322LQJ[MODE_MAX + 1] =
{
    {0, 0, 0, 0},
    {1,  1125,  32,  1080},
    {1,   750,  16,   720},
```

```
};
//MIPI(CSI)
static SENSOR_SIGNAL HD_SEN_IMX291M[MODE_MAX + 1] =
{
    {0,    0, 0, 0},
    {0, 4400, 0, 2200}, //mode 1 - 1080P30/12bit, 4Ch_MIPI
    {0, 2200, 0, 2200}, //mode 2 - 1080p60/ 12bit 4Ch_MIPI
};
static SENSOR_SIGNAL VD_SEN_IMX291M[MODE_MAX + 1] =
{
    {0,    0, 0, 0},
    {0, 1125, 0, 1125}, //mode 1
    {0, 1125, 0, 1125}, //mode 2
};
static SENSOR_SIGNAL HD_TRANS_IMX291M[MODE_MAX + 1] =
{
    {0, 0, 0, 0},
    {0, 0, 0, 1920}, //mode 1
    {0, 0, 0, 1920}, //mode 2
};
static SENSOR_SIGNAL VD_TRANS_IMX291M[MODE_MAX + 1] =
{
    {0, 0, 0, 0},
    {0, 0, 2, 1084}, //mode 1
    {0, 0, 2, 1084}, //mode 2
};
//LVDS
static UINT32 LVDS_Order_Map[LVDS_DATLANE_ID_MAX] =
{
    LVDS_PIXEL_ORDER_DL0, LVDS_PIXEL_ORDER_DL1, LVDS_PIXEL_ORDER_DL2,
LVDS_PIXEL_ORDER_DL3, LVDS_PIXEL_ORDER_DL4,
    LVDS_PIXEL_ORDER_DL5, LVDS_PIXEL_ORDER_DL6, LVDS_PIXEL_ORDER_DL7,
LVDS_PIXEL_ORDER_DL8, LVDS_PIXEL_ORDER_DL9
};
```

```
static UINT32 LVDS_ValidLane_Map[]=
{
LVDS_IN_VALID_D0,LVDS_IN_VALID_D1,LVDS_IN_VALID_D2,LVDS_IN_VALID_D3,LVDS_IN_VALID_D4,
LVDS_IN_VALID_D5,LVDS_IN_VALID_D6,LVDS_IN_VALID_D7,LVDS_IN_VALID_D8,LVDS_IN_VALID_D9
};

static SENSOR_LVDS LVDS_IMX078CQK[MODE_MAX + 1] =
{
    {//null
        {0, 0, 0, 0},
        {0, 0, 0, 0},
        0,
        0,
        0,
        0,
        0,
        {SEN_IGNORE, SEN_IGNORE, SEN_IGNORE, SEN_IGNORE, SEN_IGNORE, SEN_IGNORE,
SEN_IGNORE, SEN_IGNORE, SEN_IGNORE, SEN_IGNORE},
    },
    {//mode 0
        {8, 1170, 0, 0},
        {8, 3080, 0, 0},
        4168,
        3060,
        LVDS_DATLANE_8,
        LVDS_PIXDEPTH_12BIT,
        LVDS_DATAIN_BIT_ORDER_MSB,//??
        {0, 1, 2, 4, 5, 7, 8, 9, SEN_IGNORE, SEN_IGNORE},
    },
    {//mode 2
```

```
        {8, 462, 0, 0},
        {8, 2600, 0, 0},
        2084,
        1150,
        LVDS_DATLANE_4,
        LVDS_PIXDEPTH_12BIT,
        LVDS_DATAIN_BIT_ORDER_MSB,
        {2, 4, 5, 7, SEN_IGNORE, SEN_IGNORE, SEN_IGNORE, SEN_IGNORE, SEN_IGNORE,
SEN_IGNORE},
    },
};

static SENSOR_SIGNAL HD_SEN_IMX078CQK[MODE_MAX + 1] =
{
    {0,    0, 0, 0},
    {0, 1170, 0, 0}, //mode 0
    {0,  462, 0, 0}, //mode 2 //AE
};

static SENSOR_SIGNAL VD_SEN_IMX078CQK[MODE_MAX + 1] =
{
    {0,    0, 0, 0},
    {0, 3080, 0, 0}, //mode 0
    {0, 1300, 0, 0}, //mode 2
};

static SENSOR_SIGNAL HD_TRANS_IMX078CQK[MODE_MAX + 1] =
{
    {0, 0, 0, 0},
    {0, 0, 96, 4168 - 96}, //mode 0
    {0, 0, 48, 2084 - 48}, //mode 2
};

static SENSOR_SIGNAL VD_TRANS_IMX078CQK[MODE_MAX + 1] =
```

```
{
    {0, 0, 0, 0},
    {0, 0, 16,3060 - 16}, //mode 0
    {0, 0, 4,1150 - 6}, //mode 2
};
typedef struct
{
    UINT32 Sync;            ///< sync
    UINT32 Period;          ///< period
    UINT32 DataStart;       ///< valid data start pos
    UINT32 DataSize;        ///< valid data size
} SENSOR_SIGNAL;

static SENSOR_MODE_INFO Mode_IMX078CQK[MODE_MAX + 1] =
{
      { //mode 0  // 4000 x 3000
        SENSOR_MODE_LINEAR, // sensor mode type(HDR or .....)
        288000000,// ///< SIE clock frequence Hz sie clock >=Pixel clock
        72000000///< MCLK frequence Hz
        SENSOR_STPIX_R, ///< Sensor start pixel
        SENSOR_FMT_POGRESSIVE, ///< Sensor data type
        {SENSOR_RATIO_4_3, 1000, 1000},   // 1000/1=1000; 1000/1=1000; ///< Sensor ratio information
        {{0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}, {0, 0, 0, 0}},///< Sensor OB
        200, ///< frame rate X 10
        72000000,//(72000000/2 = 36000000 in order to drop frame rate) for NT96650 ///< pixel clock Hz
        100, ///< binning ratio X 100
        621, ///< length from VD start to 1st active line(including OB), uint:us
        49641, ///< length from VD start to last active line(including OB), uint:us
        0,
        NULL,
        &HD_TRANS_IMX078CQK[1],
        &VD_TRANS_IMX078CQK[1],
        NULL,
```

```
        NULL,
        &HD_SEN_IMX078CQK[1],
        &VD_SEN_IMX078CQK[1],
        &LVDS_IMX078CQK[1],
        NULL
    }
};
typedef struct
{
    SENSOR_MODE_TYPE ModeType;      ///< sensor mode type(HDR or .....)
    UINT32 SIEFreq;                 ///< SIE clock frequence Hz
    UINT32 MCLKFreq;                ///< MCLK frequence Hz
    SENSOR_STPIX StPix;             ///< Sensor start pixel
    SENSOR_FMT Fmt;                 ///< Sensor data type
    SENSOR_IMG_RATIO Ratio;         ///< Sensor ratio information
    SENSOR_OB OB;                   ///< Sensor OB
    UINT32 FrameRate;               ///< frame rate X 10
    UINT32 Pclk;                    ///< pixel clock Hz
    UINT32 biningRatio;             ///< binning ratio X 100
    UINT32 StrLnT;                  ///< length from VD start to 1st active line(including OB), uint:us
    UINT32 EndLnT;                  ///< length from VD start to last active line(including OB), uint:us
    UINT32 TransDelyT;              ///< length from exposure end to start of data transmission, uint:us

    SENSOR_SEL_IMG_ID *SelImgId;    ///< sensor select frame information
    SENSOR_SIGNAL *TransHD;         ///< transfer HD signal
    SENSOR_SIGNAL *TransVD;         ///< transfer VD signal
    SENSOR_SIGNAL *Trans2HD;        ///< transfer HD signal (for HDR Sensor frame 2)
    SENSOR_SIGNAL *Trans2VD;        ///< transfer VD signal (for HDR Sensor frame 2)
    SENSOR_SIGNAL *SenHD;           ///< Sensor HD signal
    SENSOR_SIGNAL *SenVD;           ///< Sensor VD signal

    SENSOR_LVDS *LVDS;              ///< lvds information
```

```
    SENSOR_DVI *DVI;                ///< dvi information
} SENSOR_MODE_INFO;


Digital zoom table setting
const UINT32 VDOZOOM_INFOR_MODE_1_TABLE[20][DZOOM_ITEM_MAX] =
{
    // sie_in          sie_out        ipe_in          crop size
    {2688, 1508, 2112, 1188, 2112, 1188, 2688, 1508}, //1x
};
```

Sensor Implementation SOP
1. Check HW pin signal
• PWDN, HW Reset, MCLK, PCLK
2. Check sensor working frequency
3. Check I2C command
• Slave address
• Address and data format
4. Sensor software reset
• Software reset -> Delay -> Action 《==Default sync output (HD, VD, PCLK)
5. Load sensor initial setting
6. Load sensor preview mode setting
• Check AGC, AEC setting
7. Check Active, Crop offset, Crop size
8. Load sensor capture mode setting

Sensor driver config files:

```
//sensor driver
IMX078CQK.c
IMX078CQK_param.c
IMX078CQK_param_Int.h
Makefile
SensorDrv.h // declare sensor /*type extern SENSOR_DRVTAB* Sensor_GetDrvTab_IMX078CQK(void);*/
MakeConfig.txt //add sensor driver
/*ifeq "$(SENSOR)" "CMOS_IMX078CQK"
  SENSOR_TYPE = _SENSORLIB_CMOS_IMX078CQK_
Endif*/
DxCamera_Sensor.c //control power and config PLL
#elif ((_SENSORLIB_ == _SENSORLIB_CMOS_IMX078CQK_) &&   \
    (_SENSORLIB2_ == _SENSORLIB2_OFF_) &&         \
```

```
    (_SENSORLIB3_ == _SENSORLIB3_OFF_) &&          \
    (_SENSORLIB4_ == _SENSORLIB4_OFF_))
static SENSOR_INIT_OBJ Sensor_GetObj(void)
{
    /*
    Sensor pin      LVDS pin
    0   A      ->   0
    1   B      ->   1
    2   C      ->   2
    3   D      ->   3
    4   E      ->   4
    5   F      ->   5
    6   G      ->   6
    7   H      ->   7
    8   I      ->   8
    9   J      ->   9
    */
    SENSOR_INIT_OBJ InitObj = {0};
    InitObj.CmdInfo.CmdType = SENSOR_CMD_SIF;
    InitObj.CmdInfo.INFO.SIF.busclk = 24000000;
    InitObj.CmdInfo.INFO.SIF.chanel = SIF_CH0;
    InitObj.CmdInfo.INFO.SIF.sen_d_s = 1;
    InitObj.CmdInfo.INFO.SIF.sen_h = 1;
    InitObj.CmdInfo.INFO.SIF.DMA_en = DISABLE;

    InitObj.ChgMclkEn = ENABLE; //notify sensor driver

    InitObj.Sen2LVDSPinMap[0] = 0;//LVDS order map, accorder to HW layout
    InitObj.Sen2LVDSPinMap[1] = 1;
    InitObj.Sen2LVDSPinMap[2] = 2;
    InitObj.Sen2LVDSPinMap[3] = 3;
    InitObj.Sen2LVDSPinMap[4] = 4;
    InitObj.Sen2LVDSPinMap[5] = 5;
```

```
    InitObj.Sen2LVDSPinMap[6] = 6;
    InitObj.Sen2LVDSPinMap[7] = 7;
    InitObj.Sen2LVDSPinMap[8] = 8;
    InitObj.Sen2LVDSPinMap[9] = 9;
    return InitObj;
}
static void SenPowerOn(void)
{
    gpio_setPin(GPIO_SENSOR_PWM2);  // 1.8 EN   // P_GPIO_37
    Delay_DelayMs(10);
    gpio_setPin(GPIO_SENSOR_PWM1);  // 1.5 EN   // P_GPIO_40
    Delay_DelayMs(10);
    gpio_setPin(GPIO_SENSOR_PWM0);  // 2.7 EN   // P_GPIO_36

    Delay_DelayMs(10);
    gpio_clearPin(GPIO_SENSOR_RESET);
    Delay_DelayMs(100);
    gpio_setPin(GPIO_SENSOR_RESET);

    if (pll_getPLLEn(PLL_ID_5) == DISABLE)
    {
        pll_setPLL(PLL_ID_5, 0x300000); //Mclk:  72000000 = 12000000* register/0x20000 //288MHz
        pll_setPLLEn(PLL_ID_5, ENABLE);
    }

    pll_selectClkSrc(PLL_CLK_SIEMCLK, PLL_CLKSRC_PLL5);
    pll_setClkFreq(PLL_CLK_SIEMCLK, 72000000);//72MHz //660 move to sensor dirver.
    pll_setClkEn(PLL_CLK_SIEMCLK, ENABLE);
}
PinmuxCfg.c //config interface
#elif ((_SENSORLIB_ == _SENSORLIB_CMOS_IMX078CQK_) && (_SENSORLIB2_ ==
_SENSORLIB2_OFF_) && (_SENSORLIB3_ == _SENSORLIB3_OFF_) && (_SENSORLIB4_ ==
_SENSORLIB4_OFF_))
```

```
    {PIN_FUNC_SENSOR,
PIN_SENSOR_CFG_LVDS|PIN_SENSOR_CFG_MCLK|PIN_SENSOR_CFG_LVDS_VDHD},  《==define mclk output and sensor signal type(master or slave), if define PIN_SENSOR_CFG_LVDS_VDHD for sensor slave, else sensor master
    {PIN_FUNC_SENSOR2,          PIN_SENSOR2_CFG_NONE},
    {PIN_FUNC_SENSOR3,          PIN_SENSOR3_CFG_NONE},
    {PIN_FUNC_SENSOR4,          PIN_SENSOR4_CFG_NONE},
    {PIN_FUNC_MIPI_LVDS,        PIN_MIPI_LVDS_CFG_CLK0 | PIN_MIPI_LVDS_CFG_DAT0 |
                                PIN_MIPI_LVDS_CFG_DAT1 | PIN_MIPI_LVDS_CFG_DAT2 |
                                PIN_MIPI_LVDS_CFG_DAT3 | PIN_MIPI_LVDS_CFG_DAT4 |
                                PIN_MIPI_LVDS_CFG_DAT5 | PIN_MIPI_LVDS_CFG_DAT6 |
                                PIN_MIPI_LVDS_CFG_DAT7 | PIN_MIPI_LVDS_CFG_DAT8 |
                                PIN_MIPI_LVDS_CFG_DAT9},
/*
static SENSOR_INFO g_pIMX078CQK_BASE_INFO =
{
        SENSOR_TYPE_CMOS,
        SENSOR_SIGNAL_SLAVE,
        SENSOR_DATA_LVDS,
        6460,
        4743,
        4000,
        3000,
          1,
        SENSOR_CMD_UNKNOWN,
        SENSOR_FPS_DEPEND_ON_EXPT,
        NULL,
};
*/
//sensor IO config
IOCfg.c
//init io
GPIO_INIT_OBJ uiGPIOMapInitTab[] = {
    {  GPIO_SENSOR_STANDBY,    GPIO_DIR_OUTPUT,    GPIO_SET_OUTPUT_HI,
PAD_PIN_NOT_EXIST      },
```

```
    { GPIO_SENSOR_RESET,      GPIO_DIR_OUTPUT,    GPIO_SET_OUTPUT_HI,
PAD_PIN_NOT_EXIST      },

    { GPIO_SENSOR_PWM2,       GPIO_DIR_OUTPUT,    GPIO_SET_OUTPUT_LOW,
PAD_PIN_NOT_EXIST     },

    { GPIO_SENSOR_PWM1,       GPIO_DIR_OUTPUT,    GPIO_SET_OUTPUT_LOW,
PAD_PIN_NOT_EXIST     },

    { GPIO_SENSOR_PWM0,       GPIO_DIR_OUTPUT,    GPIO_SET_OUTPUT_LOW,
PAD_PIN_NOT_EXIST     },
```

IOCfg.h

```
#define GPIO_SENSOR_PWM0          P_GPIO_40  /////2.8

#define GPIO_SENSOR_PWM1          P_GPIO_36   /////1.2

#define GPIO_SENSOR_PWM2          P_GPIO_37  /////1.8

#define GPIO_SENSOR_STANDBY        S_GPIO_8

#define GPIO_SENSOR_RESET          S_GPIO_7
```

Other Issue
Anti-Flicker
Image distortion (Rolling shutter effect)
Black level calibration issue (Color shift issue in long exposure time environment)
Color shading correction

Anti-Flicker

Image distortion (Rolling shutter effect)

Black level calibration issue

Color Shading

1>ERR:sie3_isr() FE-CHK: SIE IO ERR:000,00100!! ==》SIE 接收数据速度慢或是 sensor 传输速度太快

2>ERR:csi_isr() CSI ERROR! Sts0=0x00800003 Sts1=0x00000000==》SIE clock 不匹配或是 sensor output size 与 SIE in size 参数不匹配

3>ERR:FC_IPC_Dzoom() IFE H crop overflow 512(max = 511),有提供 Dzoom Table 工具，重新算 dzoom table 测试

4>ERR:Ctrl_Runtime_Chg() IPC_Chg_Load timeout

ERR:IPL_Ctrl_Runtime_Chg() command(0x00040000) error(2) 使用 ipl dumpT 0 提示 VD 为 0

ERR:rhe_isr() RHE direct buf err!

ERR:Ctrl_Runtime_Chg() IPC_Chg_Load timeout

ERR:IPL_Ctrl_Runtime_Chg() command(0x00040000) error(2)

==》sensor driver

5>提示 ERR:ICF_FC_SIE1_FLDEND_ISR() trigger1 interval shorter than process time 693 0 0!!

ERR:csi_isr() CSI ERROR! Sts0=0x00F00000 Sts1=0x400000000.==》修改 sensor setting

6>ERR:ICF_FC_TimerTrigProc() trigger interval shorter than process time!! ==》请确认硬件。

7>ERR:ime_close() State Machine Error ...

ERR:ime_close() Current State:3==》没有正常关 IME