for paper motor

```
                dcmotor_connect {
                        pwm_type = <DC_PWM_NORMAL>;
                        block_num = <5>;
                        enc_inputs = <SINGLE_A>;

                        pin_config = <   // PIN_FUNC      INVERT           PIN
                                DC_PIN_FUNC_SLP   DC_PIN_NO_INVERT DC_PIN_ONE
                                DC_PIN_FUNC_DIR   DC_PIN_NO_INVERT DC_PIN_ZERO
                                DC_PIN_FUNC_NC    DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                                DC_PIN_FUNC_ENCA DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                                DC_PIN_FUNC_PWM   DC_PIN_INVERT    DC_PIN_PWM0
                                DC_PIN_FUNC_NC    DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                                >;
                };
```

for mirror motor

```
                dcmotor_connect {
                        pwm_type = <DC_PWM_NORMAL>;
                        block_num = <4>;
                        enc_inputs = <SINGLE_A>;


                        pin_config = <   // PIN_FUNC      INVERT           PIN
                                DC_PIN_FUNC_NC    DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                                DC_PIN_FUNC_PWM   DC_PIN_INVERT    DC_PIN_PWM0
                                DC_PIN_FUNC_NC    DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                                DC_PIN_FUNC_ENCA DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                                DC_PIN_FUNC_NC    DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                                DC_PIN_FUNC_NC    DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                        >;
                };
```

The macros are defined as follow

```c
        #define DC_PIN_FUNC_NC      0       /**< Pin not connected      */
        #define DC_PIN_FUNC_PWM     1       /**< Pin used for PWM       */
        #define DC_PIN_FUNC_DIR     2       /**< Pin used for DIR (Normal only)    */
        #define DC_PIN_FUNC_SLP     3       /**< Pin used for sleep (Normal only)  */
        #define DC_PIN_FUNC_MODE    4       /**< Pin used for mode (Normal only)   */
        #define DC_PIN_FUNC_EN      5       /**< Pin used for enable (Phase only)  */
        #define DC_PIN_FUNC_ENCA    6       /**< Pin used for encoder A            */
        #define DC_PIN_FUNC_ENCB    7       /**< Pin used for encoder B            */
        #define DC_PIN_FUNC_DBG     8       /**< Pin used for debug signal         */

        #define DC_PIN_NO_INVERT    0       /**< Do not invert pin polarity        */
        #define DC_PIN_INVERT       1       /**< Invert pin polarity               */

        #define DC_PIN_NO_SIGNAL            0       /**< No output signal required      */
        #define DC_PIN_ZERO                 1       /**< Output a zero signal           */
        #define DC_PIN_ONE                  2       /**< Output a one signal            */
        #define DC_PIN_PWM0                 3       /**< Use the PWM signal as source   */
        #define DC_PIN_PWM1                 4       /**< Use the PWM signal as source   */
        #define DC_PIN_DBG_INT_ROW_SYNC     5       /**< Use Debug internal row sync    */
        #define DC_PIN_DBG_EXT_ROW_SYNC     6       /**< Use Debug external row sync    */

        #define DC_PWM_NORMAL       0       /**< PWM signal applied to PWM      */
        #define DC_PWM_PHASE        1       /**< PWM signal applied to PHASE    */

        #define AB_NORMAL   0       /**< Normal AB encoder */
        #define AB_SWAPPED  1       /**< Swapped AB encoder */
        #define SINGLE_A    2       /**< Single-ended encoded on A */
        #define SINGLE_B    3       /**< Single-ended encoded on B */
```

in driver/dcmotor/dcmotor-mod/dcmotor_priv.h

```
/**
 *  \brief Scan motor connections (Walter: 这个comment是错的)
 *
 *  Table used to specify how the 6 pins of the motor block are actually
 *  connected to the hardware.  A PWM_NORMAL connection will usually require
 *  DIR, PWM, SLP, and MODE; a PWM_PHASE connection only requires PWM and EN.
 *  Problem is these signals may be connected to the motor block in a variety of
 *  ways, the mech code needs to tell us.
 **/
typedef struct motor_connect_s {
        motor_pwm_out_t pwm_type;       /**< Type of PWM connection       */
        uint32_t        block_num;      /**< ASIC motor block number      */
        motor_enc_t     enc_inputs;     /**< encoder A/B inputs           */
        motor_pin_cfg_t pin_cfg[6];     /**< DC0 - DC5 pin configuration  */
} motor_connect_t;
```

motor_connect_t对应dts中dcmotor_connect node。

G2 SoC有7个scmotor block, as follow

in pegmatite.dtsi

```
        dcmotor@f8140000 {
                compatible = "mrvl,mv62x0-dcmotor-reg-b0";
                dcmotor_instance = <0>;
                reg = <0 0xf8140000 0 0x128>;
                interrupts = <0 182 4>;
        };

        dcmotor@f8141000 {
                compatible = "mrvl,mv62x0-dcmotor-reg-b0";
                dcmotor_instance = <1>;
                reg = <0 0xf8141000 0 0x128>;
                interrupts = <0 183 4>;
        };

        dcmotor@f8142000 {
                compatible = "mrvl,mv62x0-dcmotor-reg-b0";
                dcmotor_instance = <2>;
                reg = <0 0xf8142000 0 0x128>;
                interrupts = <0 184 4>;
        };

        dcmotor@f8143000 {
                compatible = "mrvl,mv62x0-dcmotor-reg-b0";
                dcmotor_instance = <3>;
                reg = <0 0xf8143000 0 0x128>;
                interrupts = <0 185 4>;
        };

        dcmotor@f8144000 {
                compatible = "mrvl,mv62x0-dcmotor-reg-b0";
                dcmotor_instance = <4>;
                reg = <0 0xf8144000 0 0x128>;
                interrupts = <0 186 4>;
        };

        dcmotor@f8145000 {
                compatible = "mrvl,mv62x0-dcmotor-reg-b0";
                dcmotor_instance = <5>;
                reg = <0 0xf8145000 0 0x128>;
                interrupts = <0 187 4>;
        };

        dcmotor@f8146000 {
                compatible = "mrvl,mv62x0-dcmotor-reg-b0";
                dcmotor_instance = <6>;
                reg = <0 0xf8146000 0 0x128>;
                interrupts = <0 188 4>;
        };
```

paper motor对应block_num = <5>;即dcmotor@f8145000 mirror motor对应block_num = <4>;即dcmotor@f8144000

使用的是single end encoder。

What is single end encoder ?

The format of pin definition

```
/**
 *  \brief Motor pin configuration
 *
 *  Defines the configuration for a single motor pin, only used as part of
 *  the motor_connect_t structure.
 **/
typedef struct motor_pin_cfg_s {
        motor_pin_func_t   func;      /**< Pin function                    */
        motor_pin_invert_t invert;    /**< Pin invert                      */
        motor_pin_signal_t signal;    /**< Output pin signal select        */
} motor_pin_cfg_t;
```

The 6 pins (pin0 to pin5) of paper motor的指定

```
                        pin_config = <  // PIN_FUNC      INVERT          PIN
                                DC_PIN_FUNC_SLP  DC_PIN_NO_INVERT DC_PIN_ONE
                                DC_PIN_FUNC_DIR  DC_PIN_NO_INVERT DC_PIN_ZERO
                                DC_PIN_FUNC_NC   DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                                DC_PIN_FUNC_ENCA DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                                DC_PIN_FUNC_PWM  DC_PIN_INVERT    DC_PIN_PWM0
                                DC_PIN_FUNC_NC   DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                                >;
```

pin2 and pin5没有接任何信号。

The 6 pins of mirror motor

```
                    pin_config = <  // PIN_FUNC      INVERT           PIN
                            DC_PIN_FUNC_NC   DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                            DC_PIN_FUNC_PWM  DC_PIN_INVERT    DC_PIN_PWM0
                            DC_PIN_FUNC_NC   DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                            DC_PIN_FUNC_ENCA DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                            DC_PIN_FUNC_NC   DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                            DC_PIN_FUNC_NC   DC_PIN_NO_INVERT DC_PIN_NO_SIGNAL
                    >;
```

mirror motor是单方向旋转的，所以不需要direction控制。

DC_PIN_FUNC_PWM pin是output，用于给dc motor's driver,作为控制旋转速度的参数

DC_PIN_FUNC_ENCA pin是input，应该是motor转速的feedback。

---

in driver/dcmotor/dcmotor-mod/mtr6pin.c

```c
/**
 *  \brief Set the default motor configuration
 *
 *  Function to set the default motor configuration (MCFG).  The motor 'connection' map
 *  was provided by the mech code, our job here is to get things mapped correctly.
 *  This is done just once when a motor is created, we must be sure to leave the motor
 *  block disabled.
 *
 *  \param[in] motor_regs    Pointer to the motor registers
 *  \param[in] connect_table Pointer to a motor connection table
 **/
void set_dcmtr6pin_motorconfig_initialize(struct dcmotor_reg_driver_interface *reg_iface,
        const motor_connect_t *connect_table)
{
        uint32_t mcfg_val = 0;
        uint32_t invert   = 0;

        /* Make sure to start out with the motor block disabled */
        mcfg_val = DCMOTOR_BASE_MCFG_EN_REPLACE_VAL(mcfg_val, DCMTR6PIN_MCFG_DISABLE);         ①

        /* Build the invert mask based on the settings in the connections table.  Note
         * that I didn't put this into a loop so I could use the OBA macros (in case
         * things move around in later revs).
         * BTW: this code will only work when 'not inverted' = 0.
         */
        XASSERT(DCMTR6PIN_MCFG_NO_INVERT == 0, DCMTR6PIN_MCFG_NO_INVERT);
        invert |= (connect_get_inv_for_pin(connect_table, 0) << DCMTR6PIN_MCFG_INV_PIN0);         ②
        invert |= (connect_get_inv_for_pin(connect_table, 1) << DCMTR6PIN_MCFG_INV_PIN1);
        invert |= (connect_get_inv_for_pin(connect_table, 2) << DCMTR6PIN_MCFG_INV_PIN2);
        invert |= (connect_get_inv_for_pin(connect_table, 3) << DCMTR6PIN_MCFG_INV_PIN3);
        invert |= (connect_get_inv_for_pin(connect_table, 4) << DCMTR6PIN_MCFG_INV_PIN4);
        invert |= (connect_get_inv_for_pin(connect_table, 5) << DCMTR6PIN_MCFG_INV_PIN5);

        ASSERT(invert == (invert & (DCMOTOR_BASE_MCFG_INV_MASK >> DCMOTOR_BASE_MCFG_INV_SHIFT)));
        mcfg_val = DCMOTOR_BASE_MCFG_INV_REPLACE_VAL(mcfg_val, invert);

        /* Roll through all of the pin selects.  Note that I didn't put this into a loop
         * so I could use the OBA macros (in case things move around in later rev
s).
         */
        mcfg_val = reg_iface->get_rev0_major(reg_iface) == MOTOR_REV0_MAJOR_GR2 ?
            ③
            DCMOTOR_BASE_MCFG_PINSEL0_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 0)) :
            GS2_DCMOTOR_BASE_MCFG_PINSEL0_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 0));
        mcfg_val = reg_iface->get_rev0_major(reg_iface) == MOTOR_REV0_MAJOR_GR2 ?
            DCMOTOR_BASE_MCFG_PINSEL1_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 1)) :
            GS2_DCMOTOR_BASE_MCFG_PINSEL1_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 1));
        mcfg_val = reg_iface->get_rev0_major(reg_iface) == MOTOR_REV0_MAJOR_GR2 ?
            DCMOTOR_BASE_MCFG_PINSEL2_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 2)) :
            GS2_DCMOTOR_BASE_MCFG_PINSEL2_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 2));
        mcfg_val = reg_iface->get_rev0_major(reg_iface) == MOTOR_REV0_MAJOR_GR2 ?
            DCMOTOR_BASE_MCFG_PINSEL3_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 3)) :
            GS2_DCMOTOR_BASE_MCFG_PINSEL3_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 3));
        mcfg_val = reg_iface->get_rev0_major(reg_iface) == MOTOR_REV0_MAJOR_GR2 ?
            DCMOTOR_BASE_MCFG_PINSEL4_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 4)) :
            GS2_DCMOTOR_BASE_MCFG_PINSEL4_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 4));
        mcfg_val = reg_iface->get_rev0_major(reg_iface) == MOTOR_REV0_MAJOR_GR2 ?
            DCMOTOR_BASE_MCFG_PINSEL5_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 5)) :
            GS2_DCMOTOR_BASE_MCFG_PINSEL5_REPLACE_VAL(mcfg_val, connect_get_sig_for_pin(connect_table, 5));

        /* Make sure only valid bits are being set */
        ASSERT(reg_iface->get_rev0_major(reg_iface) == MOTOR_REV0_MAJOR_GR2 ?
            (mcfg_val == (mcfg_val & (DCMOTOR_BASE_MCFG_EN_MASK    |
                        DCMOTOR_BASE_MCFG_INV_MASK      |
                        DCMOTOR_BASE_MCFG_PINSEL0_MASK  |
                        DCMOTOR_BASE_MCFG_PINSEL1_MASK  |
                        DCMOTOR_BASE_MCFG_PINSEL2_MASK  |
                        DCMOTOR_BASE_MCFG_PINSEL3_MASK  |
                        DCMOTOR_BASE_MCFG_PINSEL4_MASK  |
                        DCMOTOR_BASE_MCFG_PINSEL5_MASK))) :
            (mcfg_val == (mcfg_val & (DCMOTOR_BASE_MCFG_EN_MASK    |
                        DCMOTOR_BASE_MCFG_INV_MASK      |
                        GS2_DCMOTOR_BASE_MCFG_PINSEL0_MASK  |
                        GS2_DCMOTOR_BASE_MCFG_PINSEL1_MASK  |
                        GS2_DCMOTOR_BASE_MCFG_PINSEL2_MASK  |
                        GS2_DCMOTOR_BASE_MCFG_PINSEL3_MASK  |
                        GS2_DCMOTOR_BASE_MCFG_PINSEL4_MASK  |
                        GS2_DCMOTOR_BASE_MCFG_PINSEL5_MASK))) );

        /* Set the actual HW MCFG register */
```

```
        reg_iface->set_cfg(reg_iface, mcfg_val);
}
```

①

MCFG (Motor Control Configuration)register的Enbale bit为0(DCMTR6PIN_MCFG_DISABLE),disable PWM drive

②

对paper motor而言

```
                          pin_config = <  // PIN_FUNC      INVERT            PIN
                                  DC_PIN_FUNC_SLP  DC_PIN_NO_INVERT DC_PIN_ONE
                                  DC_PIN_FUNC_DIR  DC_PIN_NO_INVERT DC_PIN_ZERO
                                  DC_PIN_FUNC_NC   DC_PIN_NO_INVERT DC_PIN_NO_SIGNA
L
                                  DC_PIN_FUNC_ENCA DC_PIN_NO_INVERT DC_PIN_NO_SIGNA
L
                                  DC_PIN_FUNC_PWM  DC_PIN_INVERT    DC_PIN_PWM0
                                  DC_PIN_FUNC_NC   DC_PIN_NO_INVERT DC_PIN_NO_SIGNA
L
                                  >;
```

根据"INVERT" field来设置MCFG register中"INV" (bit 24 to bit 29)。

不是太理解什么意思？

③

```
                          pin_config = <  // PIN_FUNC      INVERT            PIN
                                  DC_PIN_FUNC_SLP  DC_PIN_NO_INVERT DC_PIN_ONE
                                  DC_PIN_FUNC_DIR  DC_PIN_NO_INVERT DC_PIN_ZERO
                                  DC_PIN_FUNC_NC   DC_PIN_NO_INVERT DC_PIN_NO_SIGNA
L
                                  DC_PIN_FUNC_ENCA DC_PIN_NO_INVERT DC_PIN_NO_SIGNA
L
                                  DC_PIN_FUNC_PWM  DC_PIN_INVERT    DC_PIN_PWM0
                                  DC_PIN_FUNC_NC   DC_PIN_NO_INVERT DC_PIN_NO_SIGNA
L
                                  >;
```

根据上面"PIN" field来设置MCFG中各个pin做什么用。

Enable 用到的dcmotor 4 and 5的pins。

in mv6220-ffc.dts

```
		dcmotor@f8144000 {
			compatible = "mrvl,mv62x0-dcmotor-reg-b0";
			dcmotor_instance = <4>;
			pinctrl-0 = <&mirror_motor_pins>;
			pinctrl-names = "default";
		};

		dcmotor@f8145000 {
			compatible = "mrvl,mv62x0-dcmotor-reg-b0";
			dcmotor_instance = <5>;
			pinctrl-0 = <&paper_motor_pins>;
			pinctrl-names = "default";
		};
```

```
		paper_motor_pins: pinmux_paper_motor_pins {
		pinctrl-single,pins = <
			0x9C 0x1	// BLDC_MTR_BRAKE
			0xA0 0x1	// nBLDC_MTR_DIR
			0xAC 0x1	// BLDC_MTR_EN
			0xA8 0x1	// nBLDC_MTR_READY
			>;
			pinctrl-single,bias-pulldown = <PD_OFF>;
			pinctrl-single,bias-pullup = <PU_OFF>;
		};

		mirror_motor_pins: pinmux_mirror_motor_pins {
		pinctrl-single,pins = <
			0x88 0x1	// PWM_DC_MTR3
			0x90 0x1	// MTR3_ENCDR
			>;
			pinctrl-single,bias-pulldown = <PD_OFF>;
			pinctrl-single,bias-pullup = <PU_OFF>;
		};
```

根据Gemstone2_PinMux.xls

IO Pad pin number	DCMTR5 pin when function = 1, signal	相对与IO PAD address offset
IO_PAD39	DCMTR5_P[0] (**DC_PIN_FUNC_SLP**) "1"	0x9C
IO_PAD40	DCMTR5_P[1] (**DC_PIN_FUNC_DIR**) "0"	0xA0
IO_PAD41	DCMTR5_P[2]	
IO_PAD42	DCMTR5_P[3] (**DC_PIN_FUNC_ENCA**) encoder A	0xA8
IO_PAD43	DCMTR5_P[4] (**DC_PIN_FUNC_PWM**) pwm0	0xAC
IO_PAD44	DCMTR5_P[5]	

IO Pad pin number	DCMTR4 pin when function = 1	相对与IO PAD address offset
IO_PAD33	DCMTR4_P[0]	
IO_PAD34	DCMTR4_P[1] (**DC_PIN_FUNC_PWM**) pwm0	0x88
IO_PAD35	DCMTR4_P[2]	
IO_PAD36	DCMTR4_P[3] (**DC_PIN_FUNC_ENCA**) encoder A	0x90
IO_PAD37	DCMTR4_P[4]	
IO_PAD38	DCMTR4_P[5]	

上面只有"encode A"的pin是input，用于输入motor的速度反馈信息。即该pin要配置成input。

in mtr6pin.c

```c
void set_dcmtr6pin_encoderconfig_initialize(struct dcmotor_reg_driver_interface *reg_iface,
        const motor_connect_t *connect_table,
        uint32_t                timebase_select)
{
......

 uint32_t inpin_cfg = reg_iface->get_input_pin_cfg(reg_iface);

......

        /* ENCA */
        pin = connect_get_pin_for_func(connect_table, DC_PIN_FUNC_ENCA);
        if (pin == INVALID_PIN_NUM) {
                dbg1(("%s: Must specify ENCA pin in motor connect table\n", __func__));
                XASSERT(0, DC_PIN_FUNC_ENCA);
        } else {
                inpin_cfg = DCMOTOR_BASE_IN_PIN_CFG_ENCA_SEL_REPLACE_VAL(inpin_cfg, pin);
        }

        /* ENCB */
        pin = connect_get_pin_for_func(connect_table, DC_PIN_FUNC_ENCB);
        if (pin == INVALID_PIN_NUM) {
                dbg1(("%s: Did you forget to specify ENCB pin in motor connect table\n", __func__));
        } else {
                inpin_cfg = DCMOTOR_BASE_IN_PIN_CFG_ENCB_SEL_REPLACE_VAL(inpin_cfg, pin);
        }

        /* Write the HW register */
        reg_iface->set_input_pin_cfg(reg_iface, inpin_cfg);

 ......
}
```

在"pin_config" array中搜索DC_PIN_FUNC_ENCA功能的pin,然后在Input Pin Configuration(IN_PIN_CFG) register中设置该encode A使用找到的pin number。