PWM_X signal of eFlexPWM module

The eFlexPWM module is integrated in DSC family, Kinetis KV family and MPC56xx family, it is an important module in motor control application, switch mode power supply or the other energy control application.

The eFlexPWM module has 4 sub-modules: SM0, SM1, SM2, SM3, each sub-module can output 3 PWM signal: PWM_nA, PWM_nB and PWM_nX, the n is sub module index. The PWM_nA, PWM_nB can be independent or complementary, the PWM_nX is an extended PWM channel. For example, this is the eFlexPWMA module PWM signal output pins of MC56F84789:

GPIOE1 PWMA_0A

GPIOE0 PWMA_0B

GPIOG8 PWMB_0X PWMA_0X

GPIOE3 PWMA_1A

GPIOE2 PWMA_1B

GPIOG9 PWMB_1X PWMA_1X

GPIOE5 PWMA_2A

GPIOE4 PWMA_2B

GPIOG10 PWMB_2X PWMA_2X

GPIOE7 PWMA_3A

GPIOE6 PWMA_3B

GPIOF6 PWMA_3X PWMB_3X

It is a question what is function of PWM_nX, how to generate and control the timing of PWM_nX, what is the restriction of PWM_nX signal.

The documentation talks about the PWM_nX signal, give the register setting to set up the duty cycle and tricks to control the waveform of PWM_nX.

For the PWM_nA and PWM_nB, it is clear that PWM_SMn_VAL2 controls the rising edge of PWM_nA, PWM_SMn_VAL3 controls the falling edge of PWM_nA; that PWM_SMn_VAL4 controls the rising edge of PWM_nB, PWM_SMn_VAL5 controls the falling edge of PWM_nB. But for the PWM_nX, it is confused

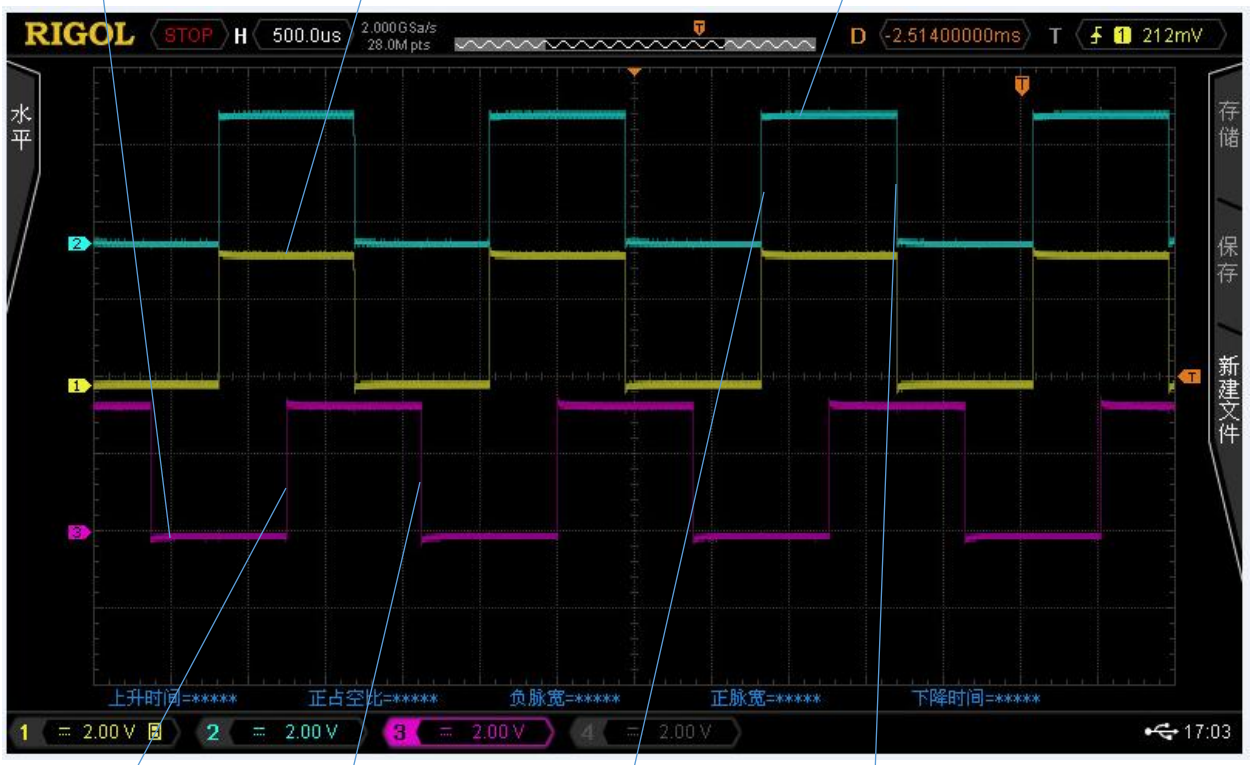
in RM, in fact the that PWM_SMn_VAL0 controls the rising edge of PWM_nX, PWM_SMn_VAL1 controls the falling edge of PWM_nX.

In general, all the PWM signals are synchronized, but of course, it is application by application. If all the PWM signals are required to be synchronized(of course, this is the generic case), we use the master sync signal from SM0 to synchronize all the other sub-module, in detail, the PWM_SM0_VAL1 generates the master sync signal to synchronize the SM0/SM1/SM2/SM3, in the case, PWM_SM1_VAL1/ PWM_SM2_VAL1/ PWM_SM3_VAL1 are not used, they can be used to control the falling edge of the PWM_1X, PWM_2X, PWM_3X. The rising edge of PWM_1X, PWM_2X, PWM_3X are controlled by the PWM_SM1_VAL0/ PWM_SM2_VAL0/ PWM_SM3_VAL0.

For the PWM_0X, because the PWM_SM0_VAL1 register is used to control the period of all PWM signal(the period is PWM_SM0_VAL1- PWM_SM0_INIT), so the PWM_0X signal falling edge can NOT be controlled, in other words, the PWM_0X waveform is restricted.

In conclusion, the eFlexPWM module can generate 2*4+3=11 PWM signal with controllable edges, and one half-controllable PWM signal.

GPIOG8/PWMA_0X　　GPIOE1/PWM_0A　　GPIOG9/PWMA_1X



PWMA_SM0_VAL0　　PWMA_SM0_VAL1　　PWMA_SM1_VAL0　　PWMA_SM1_VAL1

0x0000　　0x1000/0xF000　　0xF800　　0x800

```
PWMA_SM0VAL0=0x0000;
PWMA_SM0VAL1=0x1000;        generate signal PWMA_0X yellow signal

PWMA_SM1VAL0=0xF800;
PWMA_SM1VAL1=0x800;         generate signal PWMA_1X blue signal
```

```
PWMA_SM0VAL2=0xF800;
PWMA_SM0VAL3=0x800;
```

generate signal PWMA_0A pink signal

```
int main(void)
{

	DisableWatchdog;
	CLOCK_init();
	GPIOEG_init();
	PWMA_init();
	//PWM_ISR_SETTING();
	//PWMX_Function();
	PWMA_MCTRL|=0x0100;  //enable the PWM module
	PWMA_MCTRL|=0x0200;  //enable the PWM module
	PWMA_MCTRL|=0x0400;  //enable the PWM module
	PWMA_MCTRL|=0x0800;  //enable the PWM module
	//asm(bfclr #300,sr); //enable interrupt
	for(;;)
	{}
	return(0);

}

void PWMA_init(void)
{
	//SM0 initialization

	PWMA_SM0INIT=0xF000;
	PWMA_SM0VAL0=0x0000;
	PWMA_SM0VAL1=0x1000;	//the modulo is 256
	PWMA_SM0VAL2=0xF800;  //75% duty cycle
	PWMA_SM0VAL3=0x800;
	PWMA_SM0VAL4=0xF800;  //25% duty cycle
	PWMA_SM0VAL5=0x800;
	PWMA_SM0CTRL2=0x0000; //complementary mode for PWM0A and PWM0B, IP bus clock
	PWMA_SM0CTRL=0x3400; //4 PWM opportunity, PWM clock=Fclk
	PWMA_SM0OCTRL=0x0000; //PWM does not inverter, PWM forced to logic 0 in fault
state
	PWMA_SM0TCTRL=0x0002; //generate PWMA0_TRIG1 signal, it will be routed to
trigger PWMB0_EXT_SYNC, PWMB1_EXT_SYNC, PWMB2_EXT_SYNC
	PWMA_SM0INTEN=0x0000; //disable all interrupt
	PWMA_SM0DISMAP0=0x0000; //Disable fault mask
	PWMA_SM0DISMAP1=0x0000; //Disable fault mask
	PWMA_SM0DTCNT0=0x0000; //dead time is set to 0
	PWMA_SM0DTCNT1=0x0000;
	PWMA_SM0CTRL|=0x04;

	//SM1 module initialization

	PWMA_SM1INIT=0xF000;
	PWMA_SM1VAL0=0xF800; //set the PWMA_1X duty cycle
	PWMA_SM1VAL1=0x800;	//the modulo is 256
	PWMA_SM1VAL2=0xF800;  //75% duty cycle
```

```
	PWMA_SM1VAL3=0x800;
	PWMA_SM1VAL4=0xF800;  //25% duty cycle
	PWMA_SM1VAL5=0x800;
	PWMA_SM1CTRL2=0x200; //complementary mode, IP bus clock, the INIT_SEL should
be 10, which means
	//that the SM0 synchronize thw SM1, PWMX_INIT is set as a test
	PWMA_SM1CTRL=0x3400; //4 PWM opportunity, PWM clock=Fclk
	PWMA_SM1OCTRL=0x0000; //PWM does not inverter, PWM forced to logic 0
	PWMA_SM1TCTRL=0x0000;
	PWMA_SM1INTEN=0x0000;
	PWMA_SM1DISMAP0=0x0000; //Disable fault mask
	PWMA_SM1DISMAP1=0x0000; //Disable fault mask
	PWMA_SM1DTCNT0=0x0000; //dead time is set to 0
	PWMA_SM1DTCNT1=0x0000;
	PWMA_SM1CAPTCTRLX|=0x40; //PWMA1_X output
	PWMA_OUTEN|=0x0FF0;  //enable PWM output
//	PWMA_FCTRL)=0xF000; //fault logic setting
	PWMA_SM1CTRL|=0x04;

		//test PWM2_X output signal
		//when the SM2 module of PWMA is synchronized by SM0 sync signal, the
PWM1_X/PWM2_X/PWM3_X can output arbitary
		//waveform without any restriction because of PWMA_SMxVAL1 register,
only PWMA_SM0VAL1 register is used to control the
		//PWMA frequency
		PWMA_SM2INIT=0xF000;
		PWMA_SM2VAL0=0x0000;
		PWMA_SM2VAL1=0x1000;	//the modulo is 256
		PWMA_SM2VAL2=0xF800;  //75% duty cycle
		PWMA_SM2VAL3=0x800;
		PWMA_SM2VAL4=0xF800;  //25% duty cycle
		PWMA_SM2VAL5=0x800;
		PWMA_SM2CTRL2=0x200; //complementary mode, IP bus clock, the INIT_SEL
should be 10, which means
			//that the SM0 synchronize the SM2 module
		PWMA_SM2CTRL=0x3400; //4 PWM opportunity, PWM clock=Fclk
		PWMA_SM2OCTRL=0x0000; //PWM does not inverter, PWM forced to logic 0
		PWMA_SM2TCTRL=0x0000;
		PWMA_SM2INTEN=0x0000;
		PWMA_SM2DISMAP0=0x0000; //Disable fault mask
		PWMA_SM2DISMAP1=0x0000; //Disable fault mask
		PWMA_SM2DTCNT0=0x0000; //dead time is set to 0
		PWMA_SM2DTCNT1=0x0000;
		PWMA_SM2CTRL|=0x04;
		//PWMA global register setting
		PWMA_OUTEN|=0x0FF0;  //enable PWM output
		PWMA_MASK=0x0000;  //disable PWM mask
		PWMA_SWCOUT=0x0000;  //determine dead time logic
		//	PWMA_FCTRL)=0xF000; //fault logic setting
		PWMA_MCTRL|=0x0007; //must use the instruction, otherwise, the counter
will disorder, IPOL is cleared, PWM23 manipulate the duty cycle
		PWMA_OUTEN|=0x03; // enable PWMA_1X signal
	return;
}
```

Conclusiuon:

each eFlexPWM has 4 sub-modules:SM0, SM1,SM2, SM3, each sub-module can output three PWM signals:PWM_nA, PWM_nB, PWM_nX. In the condition that all the PWM signal are synchronized, user can get 11 PWM signal with rising/falling edge controllable completely for PWM_0A, PWM_0B; PWM_1A, PWM_1B, PWM_1X; PWM_2A, PWM_2B, PWM_2X; PWM_3A, PWM_3B, PWM_3X; But the PWM_0X falling edge can not be controlled, in other words, it is half-controllable.