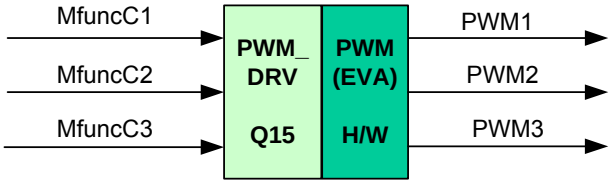
FC_PWM_DRV	*Full Compare PWM Driver*

Description

This module uses the duty ratio information and calculates the compare values for generating PWM outputs. The compare values are used in the full compare unit in 281x event manager (EVA) or EPWM unit in 280x. This also allows PWM period modulation.



Availability

This 16-bit module is available in one interface format:

1) The C interface version

Module Properties

Type: Target Dependent, Application Independent

Target Devices: 28x Fixed Point or Piccolo

C Version File Names: f2803xpwm.c, f2803xpwm.h (for x2803x)

IQmath library files for C: N/A

Item	C version	Comments
Code Size	90/257 words	
Data RAM	0 words•	
xDAIS ready	No	
XDAIS component	No	IALG layer not implemented
Multiple instances	Yes	
Reentrancy	Yes	

• Each pre-initialized PWMGEN structure consumes 9 words in the data memory

□ Code size mentioned here is the size of the ***init()*** and ***update()*** functions

C Interface

Object Definition

The structure of PWMGEN object is defined by following structure definition

```
typedef struct {  Uint16 PeriodMax;     // Parameter: PWM Half-Period in CPU clock cycles (Q0)
                  int16 MfuncPeriod;    // Input: Period scaler (Q15)
                  int16 MfuncC1;        // Input: PWM 1 Duty cycle ratio (Q15)
                  int16 MfuncC2;        // Input: PWM 2 Duty cycle ratio (Q15)
                  int16 MfuncC3;        // Input: PWM 3 Duty cycle ratio (Q15)
                  void (*init)();       // Pointer to the init function
                  void (*update)();     // Pointer to the update function
               } PWMGEN;


typedef PWMGEN *PWMGEN_handle;
```

Item	Name	Description	Format	Range(Hex)
Inputs	MfuncCx (x=1,2,3)	PWM duty cycle ratio	Q15	8000-7FFF
	MfuncPeriod	Period scaler	Q15	8000-7FFF
Outputs	PWMx (x=1,2,3,4,5,6)	Output signals from the 6 PWM pins in EVA on the x2812eZdsp.	N/A	0-3.3 V
PWMGEN parameter	PeriodMax	PWM Half-Period in CPU clock cycles	Q0	8000-7FFF

Special Constants and Data types

PWMGEN
The module definition is created as a data type. This makes it convenient to instance an interface to the PWMGEN driver. To create multiple instances of the module simply declare variables of type PWMGEN.

PWMGEN_handle
User defined Data type of pointer to PWMGEN module

PWMGEN_DEFAULTS
Structure symbolic constant to initialize PWMGEN module. This provides the initial values to the terminal variables as well as method pointers.

Methods

void F281X_EV1_PWM_Init(PWMGEN *);
void F281X_EV1_PWM_Update(PWMGEN *);

void F280X_PWM_Init(PWMGEN *);
void F280X_PWM_Update(PWMGEN *);

This default definition of the object implements two methods – the initialization and the runtime compute function for PWMGEN generation. This is implemented by means of a function pointer, and the initializer sets this to F281X_EV1_PWM_Init and F281X_EV1_PWM_Update functions for x281x or F280X_PWM_Init and F280X_PWM_Update functions for x280x. The argument to this function is the address of the PWMGEN object.

Module Usage

Instantiation
The following example instances one PWMGEN object
PWMGEN pwm1;

Initialization
To Instance pre-initialized object
PWMGEN pwm1 = PWMGEN_DEFAULTS;

Invoking the computation function
pwm1.init(&pwm1);
pwm1.update(&pwm1);

Example

The following pseudo code provides the information about the module usage.

```
main()
{

        pwm1.PeriodMax = 3000;          // PWM frequency = 10 kHz, clock = 60 MHz
        pwm1.init(&pwm1);               // Call init function for pwm1

}

void interrupt periodic_interrupt_isr()
{

        pwm1.MfuncC1 = (int)_IQtoIQ15(svgen_dq1.Ta);   // svgen_dq1.Ta is in GLOBAL_Q
        pwm1.MfuncC2 = (int)_IQtoIQ15(svgen_dq1.Tb);   // svgen_dq1.Tb is in GLOBAL_Q
        pwm1.MfuncC3 = (int)_IQtoIQ15(svgen_dq1.Tc);   // svgen_dq1.Tc is in GLOBAL_Q
        pwm1.update(&pwm1);                            // Call update function for pwm1


}
```