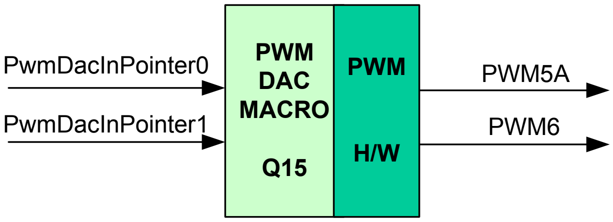
# PWMDAC DRV — *3-channel PWM DAC Driver*

**Description** This module converts any s/w variables with Q15 representation into the PWM signals in EPWM 5A & 6A for 2833x. Thus, it can be used to view the signal, represented by the variable, at the outputs of the PWM5A and PWM6A pins through the external low-pass filters.



**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:** x2833x Floating Point devices

**C Version File Names:** f2833xpwmdac.h (for x2833x)

**IQmath library files for C:** N/A

## C Interface

### Object Definition

The structure of PWMDAC object is defined by following structure definition

```
typedef struct {
          int16 *PwmDacInPointer0;    // Input: Pointer to source data output on PWMDAC ch 0
          int16 *PwmDacInPointer1;    // Input: Pointer to source data output on PWMDAC ch 1
          Uint16 PeriodMax;           // Parameter: PWMDAC half period in number of clocks (Q0)
          } PWMDAC;

typedef PWMDAC *PWMDAC_handle;
```

| Item | Name | Description | Format | Range(Hex) |
|---|---|---|---|---|
| **Inputs** | PwmDacInPointerx (x=0,1) | These input variables contain the addresses of the desired s/w variables. | N/A | N/A |
| **Outputs** | PWMx (x=5,6) | Output signals from the PWM 5-6 pins in | N/A | 0-3.3 V |
| **PWMDAC parameter** | PeriodMax | PWMDAC half period in number of clocks | Q0 | 8000-7FFF |

### Special Constants and Data types

**PWMDAC**

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

**PWMDAC_handle**

User defined Data type of pointer to PWMDAC module

**PWMDAC_DEFAULTS**

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

### Methods

**PWMDAC_INIT_MACRO (PWMDAC *);**
**PWMDAC_MACRO (PWMDAC *);**

This default definition of the object implements two methods – the initialization and the runtime compute macro for PWMDAC generation. This is implemented by means of a macro pointer, and the initializer sets this to PWMDAC_INIT_MACRO and PWMDAC_MACRO macros for x2833x. The argument to this macro is the address of the PWMDAC object.

## Module Usage

### Instantiation

The following example instances one PWMDAC object

```
PWMDAC  pwmdac1;
```

### Initialization

To Instance pre-initialized objects

```
PWMDAC pwmdac1 = PWMDAC_DEFAULTS;
```

### Invoking the computation macro

```
PWMDAC_INIT_MACRO(pwmdac1);
PWMDAC_MACRO(pwmdac1);
```

## Example

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

```
main()
{

        pwmdac1.PeriodMax = 2500;      // PWM frequency = 30 kHz, clock = 150 MHz
        pwmdac1.PwmDacInPointer0= &pwmdac_ch1_Q15;
        pwmdac1.PwmDacInPointer1= &pwmdac_ch2_Q15;
        PWMDAC_INIT_MACRO (pwmdac1);    // Call init macro for pwmdac1

}

void interrupt periodic_interrupt_isr()
{

        pwmdac_ch1_Q15 = (int)_IQtoIQ15(variable1_in_IQ);
        pwmdac_ch2_Q15 = (int)_IQtoIQ15(variable2_in_IQ);
        PWMDAC_MACRO(pwmdac1); // Call update macro for pwmdac1

}
```

## Technical Background

This module converts any s/w variables with Q15 representation into the PWM signals in 2833x devices. The EPWM5-6 will be used. Therefore, the external low-pass filters are necessary to view the actual signal waveforms as seen in Figure 1. The (1$^{st}$-order) RC low-pass filter can be simply used for filter out the high frequency component embedded in the actual low frequency signals. To select R and C values, its time constant can be expressed in term of cut-off frequency ($f_c$) as follow:

$$\tau = RC = \frac{1}{2\pi f_c} \qquad (1)$$

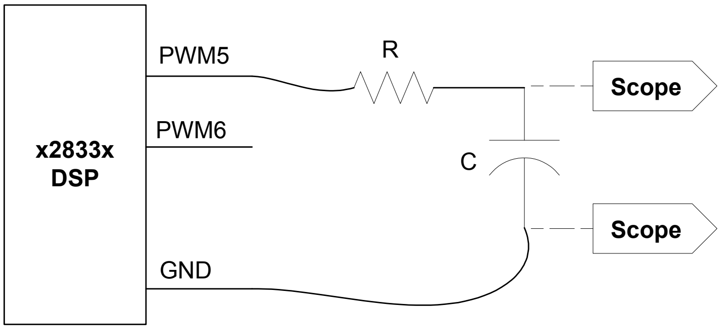


Figure 1: External RC low-lass filter connecting to PWM5 pin in x2833x DSP