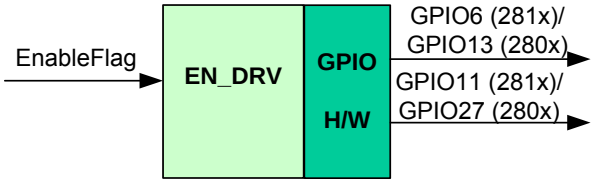
EN_DRV	*Enable PWM Driver*

Description

This module allows user to enable or disable six PWM signals on DMC1500 inverter board, using GPIO pins of 28xx devices.



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: x281x or x280x

C Version File Names: f281x_en.c, f281x_en.h (for x281x)
f280x_en.c, f280x_en.h (for x280x)

IQmath library files for C: N/A

Item	C version	Comments
Code Size□ (x281x/x280x)	40/42 words	
Data RAM	0 words•	
xDAIS ready	No	
XDAIS component	No	IALG layer not implemented
Multiple instances	Yes	
Reentrancy	Yes	

• Each pre-initialized DRIVE structure consumes 5 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 DRIVE object is defined by following structure definition

```
typedef struct { Uint16 EnableFlag;     // Input: Enable/Disable (1/0) flag (Q0)
                 void (*init)();        // Pointer to the init function
                 void (*update)();      // Pointer to the update function
               } DRIVE ;

typedef DRIVE *DRIVE_handle;
```

Item	Name	Description	Format	Range(Hex)
Inputs	EnableFlag	Enable flag Enable = 1, Disable = 0	Q0	0 or 1
Outputs	GPIO6,GPIO11 (x281x) GPIO13,GPIO27 (x280x)	General-purpose digital I/O configured as output on 28xx device	N/A	0-3.3 v

Special Constants and Data types

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

DRIVE_handle
User defined Data type of pointer to DRIVE module

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

Methods

void F281X_EV1_DRIVE_Init(DRIVE *);
void F281X_EV1_DRIVE_Update(DRIVE *);

void F280X_DRIVE_Init(DRIVE *);
void F280X_DRIVE_Update(DRIVE *);

This default definition of the object implements two methods – the initialization and the runtime update function for DRIVE generation. This is implemented by means of a function pointer, and the initializer sets this to F281X_EV1_DRIVE_Init and F281X_EV1_DRIVE_Update functions for x281x or F280X_DRIVE_Init and F280X_DRIVE_Update functions for x280x. The argument to this function is the address of the DRIVE object.

Module Usage

Instantiation
The following example instances one DRIVE object
DRIVE drv1;

Initialization
To Instance pre-initialized objects
DRIVE drv1 = DRIVE_DEFAULTS;

Invoking the computation function
drv1.init(&drv1);
drv1.update(&drv1);

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

```
main()
{

        drv1.init(&drv1);                    // Call init function for drv1

}

void interrupt periodic_interrupt_isr()
{

        drv1.EnableFlag = EnableFlag;
        drv1.update(&drv1);                  // Call update function for drv1

}
```