Module Document: AC_LINE_RECT

Texas Instruments – C2000 DSP System Applications Group

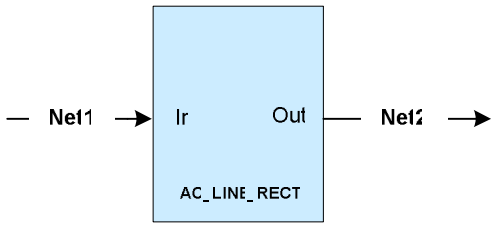
Contents

Figures

Tables

1 AC_LINE_RECT module documentation

This software module performs a numerical rectification function from a bipolar input signal. An input signal which is represented in Q15, representing for example a sine wave is bipolar in nature, i.e. it can have a positive or negative sign. This module converts such a signal to a unipolar format, with a positive sign.



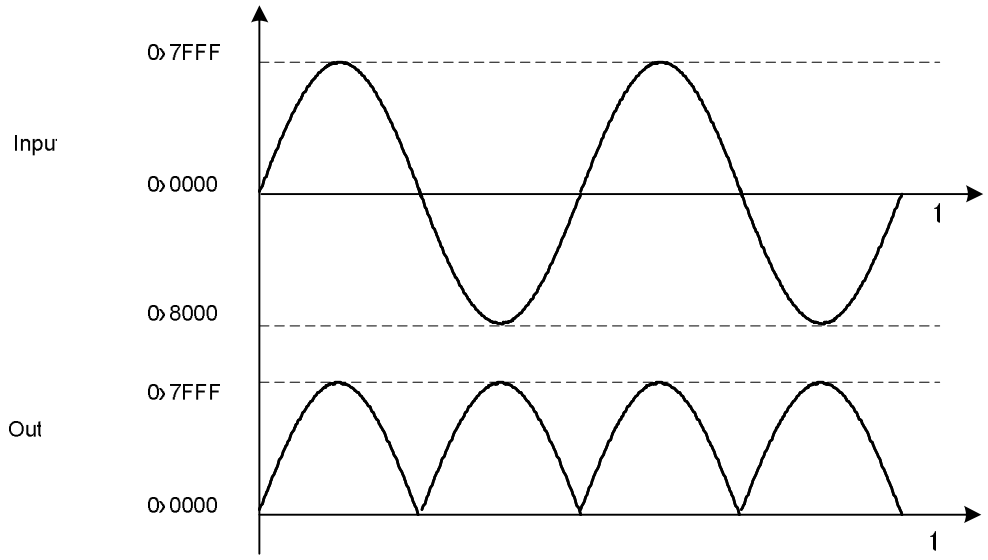


Figure 1. Input and output waveforms for the AC_LINE_RECT module

1.1 Module Properties

This section describes module properties, such as compatible devices, components, invocation etc. The AC_LINE_RECT module has the following dependencies:

Module	Dependency
CPU dependency	C28x
Device dependency	Any 28x device

Table 1. AC_LINE_RECT module dependencies

The AC_LINE_RECT module has the following components:

Component	Present
C-based initialization	No
ASM interrupt initialization	Yes
ASM runtime macro	Yes

Table 2 AC_LINE_RECT module components

The AC_LINE_RECT module has the following miscellaneous properties:

Property name	Property value
Multiple instance support	Yes
Reentrant	Yes
Accessible from 'C' environment	Yes
Full configuration from 'C' environment	Yes
Input / Output connection	Pointer to signal net.

Table 3. AC_LINE_RECT module miscellaneous properties

Component	Files
Macro source:	C:\tidcs\DPS_C280x\vXYZ[1]\dplib280x\dplib280x.inc
C Interface:	C:\tidcs\DPS_C280x\vXYZ\dplib280x\dplib280x.h
C source	C:\tidcs\DPS_C280x\vXYZ\dplib280x\dplib280x.src
Object file archive	C:\tidcs\DPS_C280x\vXYZ\dplib280x\dplib280x.lib

Table 4. AC_LINE_RECT module component files

[1] The xyz represents the version number directory level. For instance, a 1.00 release would have v100 in its directory path, and v210 would indicate a release 2.10.

1.2 Module Input Definition

Input name	Description	Format	Range
AC_LINE_RECT_InX	Input	Pointer to 16-bit fixed point input data	Q15: [-1, 1) or [-32768, 32767]

(X is the instance number)

Table 5. AC_LINE_RECT terminal inputs

1.3 Module Output Definition

Output name	Description	Format	Range
AC_LINE_RECT_OutX	Output	Pointer to 16-bit fixed point input data	Q15: [0, 1) or [0, 32767]

(X is the instance number)

Table 6. AC_LINE_RECT terminal outputs

1.4 Module API Description: AC_LINE_RECT_INIT

Function Name: AC_LINE_RECT_INIT

Prototype: AC_LINE_RECT_INIT nInstance

Return value: None.

Preconditions: None

This function is the assembler initialization macro, and must be called prior to running the AC_LINE_RECT runtime macro, for proper operation of the runtime macro routine. This initialization routine must be executed as part of an assembler initialization routine. This macro routine declares variables, initializes variables to known values, and sets up constants for the runtime macro routines.

1. nInstance**:** Specifies which instance is initialized.

Example: Call the AC_LINE_RECT_INIT to initialize the module.

```
;-------------------------------------------------------
; ISR Initialization
;-------------------------------------------------------
_ISR_Init:  ...
                    ...
                    AC_LINE_RECT_INIT                    1

                    LRETR
```

1.5 Module API Description:AC_LINE_RECT

Function Name: AC_LINE_RECT

Prototype: AC_LINE_RECT nInstance

Return value: None.

Preconditions: The following preconditions must be satisfied:

- The ISR initialization macro AC_LINE_RECT_INIT must be instanced in an assembler initialization routine, and the assembler init routine must run prior to this routine.

This function is the assembler run time macro, and this creates code runs the AC line rectification function.

- nInstance**:** Specifies which instance is run.

Example: Call the AC_LINE_RECT in an assembler ISR

```
;-------------------------------------------------------
; Runtime interrupt service routine
;-------------------------------------------------------
_ISR_Run:           CONTEXT_SAVE              ;call macro

;-------------------------------------------------------
; Run the AC_LINE_RECT instance 1
;-------------------------------------------------------
                    AC_LINE_RECT 1
;-------------------------------------------------------
EXIT_ISR: ;Interrupt management before exit
;-------------------------------------------------------
                    MOVW        DP,#ETCLR1>>6
                    MOV         @ETCLR1,#0x01; Clear EPWM1 Int flag

;-------------------------------------------------------
; Restore context & return
;-------------------------------------------------------
                    CONTEXT_REST
                    IRET
```

1.6 Usage Example:

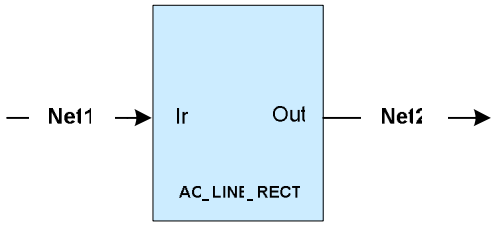


Figure 2. Connecting the AC_LINE_RECT module

Step1. Instantiate the INIT macro in assembly (this is one-time pass through code)

```
; Instantiate the init macro

       AC_LINE_RECT_INIT                      1
```

Step2. Instantiate the run time macro in assembly (this is usually looped or ISR code)

```
; “call” the runtime macro

       AC_LINE_RECT              1
```

Step3. Declare signal nets to which the module will be connected

```
int16   Net1, Net2;
```

Step3. Declare the terminal pointers in C

```
// AC_LINE_RECT terminal pointers, external references

extern int16           * AC_LINE_RECT_In1, *AC_LINE_RECT_Out1;
```

Step6. “Connect” the module terminals to the Signal Nets in “C”.

```
// AC_LINE_RECT connections
// Note this can be done once during init, or dynamically during
// run time operation, i.e. module connections can be
// re-configured to other Nets as required by the application.
       AC_LINE_RECT_In1          = &Net1;
       AC_LINE_RECT_Out1 = &Net2;
```