RAMPGEN_1CH

Texas Instruments – C2000 DSP System Applications Group

Table of contents

Table of Figures

Index of Tables

1 Overview

This module implements a ramp generator.

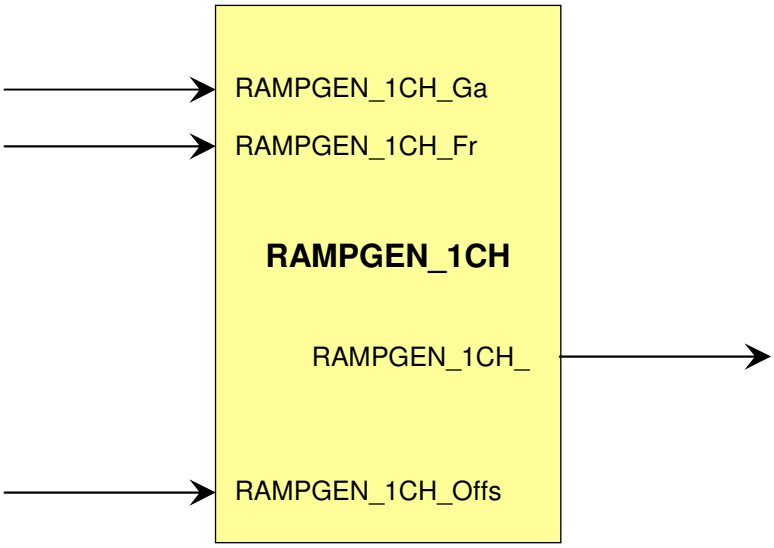

Figure 1. Single channel sine wave generator

2 Module Properties

This section describes module properties, such as compatible devices, components, invocation etc. The RAMPGEN_1CH module has the following dependencies:

Module	Dependency
CPU dependency	C28x
Device dependency	None (as long as the CPU dependency is satisfied)
Target application	Closed loop control.
Math format (precision)	32 bit fixed Q

Table 1. RAMPGEN_1CH module dependencies

The RAMPGEN_1CH module has the following components:

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

Table 2. RAMPGEN_1CH module components

The RAMPGEN_1CH module has the following miscellaneous properties:

Property name	Property value
Multiple instance support	Yes
Reentrant	No
Accessible from 'C' environment	Yes
Full configuration from 'C' environment	Yes
Input / Output connection	Pointer to signal net.

Table 3. RAMPGEN_1CH module miscellaneous properties

Component	Files
Macro source:	C:\tidcs\DPS_C280x\vXYZ[†]\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. RAMPGEN_1CH module component files

3 Module data definitions

3.1 Module inputs

Input name	Description	Data Format	Range
RAMPGEN_1CH_Gain	Amplitude control for the sine wave generator	Pointer to 16-bit fixed point input data	Q15: [-1, 1] or [-32768, 32767]
RAMPGEN_1CH _Freq	Frequency control for the sine wave generator	Pointer to 16-bit fixed point input data	Q15: [-1, 1] or [-32768, 32767]
RAMPGEN_1CH _Offset	Offset control for the sine wave generator	Pointer to 16-bit fixed point input data	Q15: [-1, 1] or [-32768, 32767]

[†] 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.

Table 5. RAMPGEN_1CH module inputs

3.2 Module output definitions

Output name	Description	Data Format	Range
RAMPGEN_1CH _Out	Sine wave output	Pointer to 16-bit fixed point output data	Q15: [-1, 1] or [-32768, 32767]

Table 6. RAMPGEN_1CH module outputs

4 Module API Description

This module has two executable code components, as described in Table 2. Each of these components is described in this section.

4.1 RAMPGEN_1CH_INIT

Function Name:	RAMPGEN_1CH_INIT
Prototype:	RAMPGEN_1CH_INIT nInstance
Return value:	None
Preconditions:	None

This function is the assembler initialization macro, and must be called 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.

- nInstance**:** Specifies the instance number of the current instance.
  - **Valid Range:** Limited only by available memory in the application.

Example: Call the RAMPGEN_1CH_INIT to initialize ramp generator instance 1.

```
;-------------------------------------------------------------
; ISR Initialization
;-------------------------------------------------------------
_ISR_Init:  . . . . . .  ; Other init routines go in here

            RAMPGEN_1CH_INIT  1

            . . . . . .  ; Other init routines go in here

            LRETR
```

4.2 RAMPGEN_1CH_INIT

Function Name: RAMPGEN_1CH

Prototype: RAMPGEN_1CH nInstance

Return value: None.

Preconditions: The following preconditions must be satisfied:

- The ISR initialization macro RAMPGEN_1CH_INIT must be instanced in an assembler initialization routine, and must run prior to this instance of the controller routine.

This function is the assembler run time macro, and this creates code that runs the run time computation for the sine wave. This computes the sine wave output and outputs it to the variable pointed to by the output pointer.

- nInstance**:** Specifies which controller instance is computed.
  - **Valid Range:** Limited only by available memory in the application.

Example: Call the RAMPGEN_1CH in an assembler ISR

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

            RAMPGEN_1CH 1
;-------------------------------------------------------
EXIT_ISR: ;Interrupt management before exit
;-------------------------------------------------------
            MOVW  DP,#ETCLR1>>6
            MOV   @ETCLR1,#0x01     ; Clear EPWM1 Int flag

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

5 Usage Example:

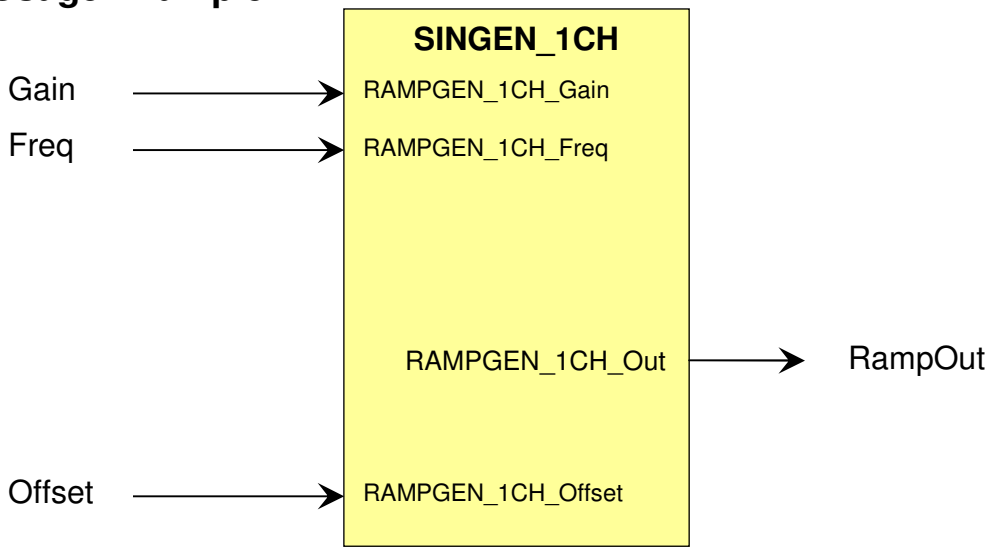

Figure 2. Single channel sine wave generator usage

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

```
; "call" the 1st instantiation of the init macro
     RAMPGEN_1CH_INIT         1
```

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

```
; "call" the main macro
     RAMPGEN_1CH         1
```

Step3. (optional) Declare "Signal Nets" to "connect" the module to in "C"

```
     int16 Gain, Freq, Offset, RampOut;
```

Step4. Declare the module "Terminal pointers" in "C"

```
// CNTL_2P2Z terminal external references for 1st instantiation
     extern int16   *RAMPGEN_1CH_Gain1, *RAMPGEN_1CH_Offset1;
     extern int16   *RAMPGEN_1CH_Freq1, *RAMPGEN_1CH_Out1;
```

Step5. "Connect" the module terminals to the Signal Nets in "C".

```
// CNTL_2P2Z(1) connections

     RAMPGEN_1CH_Gain1   = &Gain;
     RAMPGEN_1CH_Offset1 = &Freq;
     RAMPGEN_1CH_Freq1   = &Offset;
     RAMPGEN_1CH_Out1    = &RampOut;

// 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.
```