Table of contents

1 Overview....................................................................................................................... 2
2 Module Properties.............................................................................................................. 2
3 Module data definitions ...................................................................................................... 3
  3.1 Module inputs .................................................................................................................. 3
  3.2 Module output definitions ............................................................................................... 3
  3.3 Module coefficient configuration ..................................................................................... 4
4 Module API Description...................................................................................................... 5
  4.1 CNTL_2P2Z_INIT ............................................................................................................ 5
  4.2 CNTL_2P2Z.................................................................................................................... 6
5 Usage Example................................................................................................................... 7
6 Detailed description......................................................................................................... 8

Table of Figures

Figure 1. Two pole two zero controller module ................................................................. 2

Index of Tables

Table 1. CNTL_2P2Z module dependencies ...................................................................... 2
Table 2. CNTL_2P2Z module components ....................................................................... 2
Table 3. CNTL_2P2Z module miscellaneous properties .................................................. 3
Table 4. CNTL_2P2Z module component files .................................................................. 3
Table 5. CNTL_2P2Z module inputs ................................................................................ 3
Table 6. CNTL_2P2Z module outputs ............................................................................... 3
Table 7. CNTL_2P2Z module coefficient formats ............................................................. 4
1 Overview

This module implements a 2 pole – 2 zero controller, by implementing the second order difference equation shown in equation (2) below.

\[
\frac{U(z)}{E(z)} = \frac{B_2 z^2 + B_1 z + B_0}{-A_2 z^2 - A_1 z + 1}
\]  \hspace{1cm} (1)

\[
U(n) = A_1 \cdot U(n-1) + A_2 \cdot U(n-2) + B_0 \cdot E(n) + B_1 \cdot E(n-1) + B_2 \cdot E(n-2) \hspace{1cm} (2)
\]

2 Module Properties

This section describes module properties, such as compatible devices, components, invocation etc. The CNTL_2P2Z module has the following dependencies:

<table>
<thead>
<tr>
<th>Module</th>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU dependency</td>
<td>C28x</td>
</tr>
<tr>
<td>Device dependency</td>
<td>None (as long as the CPU dependency is satisfied)</td>
</tr>
<tr>
<td>Target application</td>
<td>Closed loop control.</td>
</tr>
<tr>
<td>Math format (precision)</td>
<td>32 bit fixed Q</td>
</tr>
</tbody>
</table>

Table 1. CNTL_2P2Z module dependencies

The CNTL_2P2Z module has the following components:

<table>
<thead>
<tr>
<th>Component</th>
<th>Present</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-based initialization</td>
<td>No</td>
</tr>
<tr>
<td>ASM interrupt initialization</td>
<td>Yes</td>
</tr>
<tr>
<td>ASM runtime macro</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 2. CNTL_2P2Z module components
The CNTL_2P2Z module has the following miscellaneous properties:

<table>
<thead>
<tr>
<th>Property name</th>
<th>Property value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiple instance support</td>
<td>Yes</td>
</tr>
<tr>
<td>Reentrant</td>
<td>No</td>
</tr>
<tr>
<td>Accessible from ‘C’ environment</td>
<td>Yes</td>
</tr>
<tr>
<td>Full configuration from ‘C’</td>
<td>Yes</td>
</tr>
<tr>
<td>environment</td>
<td></td>
</tr>
<tr>
<td>Input / Output connection</td>
<td>Pointer to signal net.</td>
</tr>
</tbody>
</table>

*Table 3. CNTL_2P2Z module miscellaneous properties*

**Component files**

C:\tidcs\DPS_C280x\Vxyz\lib\dpslib280x\include\ControlLawMacro.h†

*Table 4. CNTL_2P2Z module component files*

3 Module data definitions

3.1 Module inputs

<table>
<thead>
<tr>
<th>Input name</th>
<th>Description</th>
<th>Data Format</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ref</td>
<td>Reference Set point</td>
<td>Pointer to 16-bit fixed</td>
<td>Q15: [-1, 1]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>point input data</td>
<td>or [-32768, 32767]</td>
</tr>
<tr>
<td>Fdbk</td>
<td>Feedback, used to calculate the</td>
<td>Pointer to 16-bit fixed</td>
<td>Q15: [-1, 1]</td>
</tr>
<tr>
<td></td>
<td>error term</td>
<td>point input data</td>
<td>or [-32768, 32767]</td>
</tr>
</tbody>
</table>

*Table 5. CNTL_2P2Z module inputs*

3.2 Module output definitions

<table>
<thead>
<tr>
<th>Output name</th>
<th>Description</th>
<th>Data Format</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Out</td>
<td>Controller output</td>
<td>Pointer to 16-bit fixed</td>
<td>Q15: [-1, 1]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>point output data</td>
<td>or [-32768, 32767]</td>
</tr>
</tbody>
</table>

*Table 6. CNTL_2P2Z module outputs*

† 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.
3.3 Module coefficient configuration

The controller coefficients $A_1$, $A_2$, $B_0$, $B_1$ and $B_2$ in equation (1) above are specified in a record in memory. All coefficients are 32 bits wide, and are in a Q format as described below. The coefficient formats are shown in Table 7 below.

<table>
<thead>
<tr>
<th>Coefficient</th>
<th>Description</th>
<th>Format</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>B2</td>
<td>Controller coefficient</td>
<td>32-bit fixed point</td>
<td>Q26: [-32, 31.99999]</td>
</tr>
<tr>
<td>B1</td>
<td>Controller coefficient</td>
<td>32-bit fixed point</td>
<td>Q26: [-32, 31.99999]</td>
</tr>
<tr>
<td>B0</td>
<td>Controller coefficient</td>
<td>32-bit fixed point</td>
<td>Q26: [-32, 31.99999]</td>
</tr>
<tr>
<td>A2</td>
<td>Controller coefficient</td>
<td>32-bit fixed point</td>
<td>Q26: [-32, 31.99999]</td>
</tr>
<tr>
<td>A1</td>
<td>Controller coefficient</td>
<td>32-bit fixed point</td>
<td>Q26: [-32, 31.99999]</td>
</tr>
<tr>
<td>MAX</td>
<td>Control output upper saturation bound</td>
<td>32-bit fixed point</td>
<td>Q24: [-128, 127.99999]</td>
</tr>
<tr>
<td>MIN</td>
<td>Control output lower saturation bound</td>
<td>32-bit fixed point</td>
<td>Q24: [-128, 127.99999]</td>
</tr>
</tbody>
</table>

Table 7. CNTL_2P2Z module coefficient formats

The coefficients must be declared in (for convenience) the same file which instantiates the macro CNTL_2P2Z_INIT. The order of the coefficients is significant, since the CNTL_2P2Z_INIT macro takes the base of this coefficient record as a parameter and all the coefficients are accessed relative to this address. Here is an example of how to set up coefficients. Each coefficient is declared as a '.long' which creates a 32-bit constant.

```assembly
.sect "CNTL_coeff"
DCDC_VLOOP_COEFF1:
.long 141465485 ; B2
.long -308700774 ; B1
.long 167772160 ; B0
.long -5368709 ; A2
.long 72477573 ; A1
.long 0x00FFFFFF ; MAX
.long 0x00000000 ; MIN
```
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 CNTL_2P2Z_INIT

Function Name: CNTL_2P2Z_INIT
Prototype: CNTL_2P2Z_INIT nInstance, address
Return value: None
Preconditions: None

This function is the assembler initialization macro, and must be called in addition to the C language initialization routine, 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 CNTL_2P2Z_INIT to initialize the two pole, two zero controller module.

```
_isr_Init: . . . . . ; Other init routines go in here
CNTL_2P2Z_INIT 1, DCDC_VLOOP_COEFF1
. . . . . ; Other init routines go in here
lretr
```
4.2 CNTL_2P2Z

Function Name: CNTL_2P2Z
Prototype: CNTL_2P2Z nInstance
Return value: None.
Preconditions: The following preconditions must be satisfied:

The ISR initialization macro CNTL_2P2Z_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 two pole – two zero controller. This computes the control output depending on the

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

Example: Call the CNTL_2P2Z in an assembler ISR

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

       CNTL_2P2Z  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:

Step 1. Instantiate the INIT macro in assembly (this is one-time pass through code)

PFC_ILOOP_COEFF:

```
.long -8760341 ; B2 Q26
.long 3265008 ; B1 Q26
.long 12025350 ; B0 Q26
.long 14900244 ; A2 Q26
.long 52208619 ; A1 Q26
.long 0x00FFFFFF ; MAX Q24
.long 0x00000000 ; MIN Q24
```

; “call” the 1st instantiation of the init macro

```
CNTL_2P2Z_INIT 1, PFC_ILOOP_COEFF
```

Step 2. Instantiate the run time macro in assembly (this is usually looped or ISR code)

; “call” the main macro

```
CNTL_2P2Z 1
```

Step 3. (optional) Declare “Signal Nets” to “connect” the module to in “C”

```
int Net1, Net2, Net3;
```

Step 4. Declare the module “Terminal pointers” in “C”

```
// CNTL_2P2Z terminal external references for 1st instantiation
extern int *CNTL_2P2Z_Ref1, *CNTL_2P2Z_Fdbk1, *CNTL_2P2Z_Out1;
```

Step 5. “Connect” the module terminals to the Signal Nets in “C”.

```
// CNTL_2P2Z(1) connections

CNTL_2P2Z_Ref1 = &Net1;
CNTL_2P2Z_Out1 = &Net2;
CNTL_2P2Z_Fdbk1 = &Net3;

// 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.
```