Table of contents

1 Overview ........................................................................................................................................... 2
2 Module Properties ............................................................................................................................ 2
3 Module Input and Output Definitions ............................................................................................... 3
  3.1 Module inputs .......................................................................................................................... 3
  3.2 Module outputs ......................................................................................................................... 3
4 Module API Description .................................................................................................................. 4
  4.1 ADC1CONTConf .................................................................................................................... 4
  4.2 ADC1CH_DRV_INIT .............................................................................................................. 5
  4.3 ADC1CH_DRV ....................................................................................................................... 5
5 Usage Example .............................................................................................................................. 7
6 Detailed description ....................................................................................................................... 7

Table of Figures

Figure 1. Continuous conversion, one channel ADC Driver ................................................................. 2
Figure 2. Connecting the one channel ADC driver ................................................................................ 7

Index of Tables

Table 1. ADC1CONT_DRV module dependencies .............................................................................. 2
Table 2. ADC1CONT_DRV module components ................................................................................. 2
Table 3. ADC1CONT_DRV module miscellaneous properties ............................................................... 3
Table 4. ADC1CONT_DRV module component files ........................................................................... 3
Table 5. ADC1CONT_DRV module inputs ........................................................................................... 3
Table 6. ADC1CONT_DRV module outputs ......................................................................................... 3
Table 7. ADC1CONT_DRV channel selection ...................................................................................... 4
1 Overview
This software module controls the ADC on the F/C280x devices. It sets up the ADC in continuous conversion mode, to convert one channel, and to make it available in a variable in the application. The conversion result is unipolar.

![Diagram of ADC1CONT_DRV and ADC_A0](image)

Figure 1. Continuous conversion, one channel ADC Driver

2 Module Properties
This section describes module properties, such as compatible devices, components, invocation etc. The ADC1CONT_DRV 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>x2801 / x2806 / x2808 members only</td>
</tr>
</tbody>
</table>

Table 1. ADC1CONT_DRV module dependencies

The ADC1CONT_DRV module has the following components:

<table>
<thead>
<tr>
<th>Component</th>
<th>Present</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-based initialization</td>
<td>Yes</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. ADC1CONT_DRV module components
The ADC1CONT_DRV 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>No</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’ environment</td>
<td>Yes</td>
</tr>
<tr>
<td>Input / Output connection</td>
<td>Pointer to signal net.</td>
</tr>
</tbody>
</table>

*Table 3. ADC1CONT_DRV module miscellaneous properties*

**Component files**

C:\tidcs\DPS_C280x\Vxyz\lib\drvlib280x\src\ADC_1ChContDrvCnf.c†

C:\tidcs\DPS_C280x\Vxyz\lib\drvlib280x\include\ADC_DriverMacro.h†

*Table 4. ADC1CONT_DRV module component files*

## 3 Module Input and Output Definitions

### 3.1 Module inputs

<table>
<thead>
<tr>
<th>Input name</th>
<th>Description</th>
<th>Data Format</th>
<th>Range</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADCINAn</td>
<td>F280x/C280x ADC input pin</td>
<td>Analog sense value</td>
<td>See device datasheet for electrical specifications.</td>
<td></td>
</tr>
</tbody>
</table>

*Table 5. ADC1CONT_DRV module inputs*

### 3.2 Module outputs

<table>
<thead>
<tr>
<th>Output name</th>
<th>Description</th>
<th>Data Format</th>
<th>Output data Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC1CH_Rslt</td>
<td>Driver output pointer</td>
<td>Pointer to 16-bit fixed point output date</td>
<td>Q15: [-1, 1] or [-32768, 32767]</td>
</tr>
</tbody>
</table>

*Table 6. ADC1CONT_DRV 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.
4 Module API Description

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

4.1 ADC1CONTCnf

Function Name: ADC1CONTCnf

Prototype: void ADC1CONTCnf(int nChannel, int AcqWidth);

Return value: None.

Preconditions: The following preconditions must be satisfied:

- The ADC module clock must be enabled in the PCLKCR0 register.

The ADC1CONTCnf function is called from the C environment, and performs driver configuration, including configuration of ADC registers.

- **Valid Range:** 0-15, corresponding to ADC channels A0-A7 and B0-B7. The channel selection is shown in Table 7 below.

- **AcqWidth:** Specifies the acquisition prescaler in clock cycles.

<table>
<thead>
<tr>
<th>Channel setting</th>
<th>Channel converted</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ADCINA0</td>
</tr>
<tr>
<td>1</td>
<td>ADCINA1</td>
</tr>
<tr>
<td>2</td>
<td>ADCINA2</td>
</tr>
<tr>
<td>3</td>
<td>ADCINA3</td>
</tr>
<tr>
<td>4</td>
<td>ADCINA4</td>
</tr>
<tr>
<td>5</td>
<td>ADCINA5</td>
</tr>
<tr>
<td>6</td>
<td>ADCINA6</td>
</tr>
<tr>
<td>7</td>
<td>ADCINA7</td>
</tr>
<tr>
<td>8</td>
<td>ADCINB0</td>
</tr>
<tr>
<td>9</td>
<td>ADCINB1</td>
</tr>
<tr>
<td>10</td>
<td>ADCINB2</td>
</tr>
<tr>
<td>11</td>
<td>ADCINB3</td>
</tr>
<tr>
<td>12</td>
<td>ADCINB4</td>
</tr>
<tr>
<td>13</td>
<td>ADCINB5</td>
</tr>
<tr>
<td>14</td>
<td>ADCINB6</td>
</tr>
<tr>
<td>15</td>
<td>ADCINB7</td>
</tr>
</tbody>
</table>

*Table 7. ADC1CONT_DRV channel selection*

Example: Call the ADC1CONTCnf function to initialize EPWM1 module.

```
//------------------------------------------------------------------
// Set up conversion of channel 0, with acq prescaler at 2 clocks
//------------------------------------------------------------------
ADC1CONTCnf(0, 2);
```
4.2 ADC1CH_DRV_INIT

Function Name: ADC1CH_DRV_INIT
Prototype: ADC1CH_DRV_INIT
Return value: None.
Preconditions: The following preconditions must be satisfied:
- The ADC module clock must be enabled in the PCLKCR0 register, and
- The C language init routine must be called.

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.

Example: Call the ADC1CH_DRV_INIT to initialize ADC1CH_DRV module.

```c
;---------------------------------------------------------
; ISR Initialisation
;---------------------------------------------------------
_ISR_Init: ADC1CH_DRV_INIT
LRETR
```

4.3 ADC1CH_DRV

Function Name: ADC1CH_DRV
Prototype: ADC1CH_DRV
Return value: None.
Preconditions: The following preconditions must be satisfied:
- The ADC module clock must be enabled in the PCLKCR0 register.
- The C language init routine must be called.
- The ISR initialization macro ADC1CH_DRV_INIT must be instanced and executed in an assembler initialization routine.

This function is the assembler run time macro, and this creates code that forms a bridge between software and the ADC register hardwar output. The output is written to the ADC1CH_Rslt variable.
Example: Call the ADC1CH_DRV in an assembler ISR

;------------------------------------------------------------------------
; Runtime interrupt service routine
;------------------------------------------------------------------------
_ISR_Run:   CONTEXT_SAVE        ; call macro
             ADC1CH_DRV
             . . .                   ; other code
;------------------------------------------------------------------------
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:

Usage Example:

Step1. Call the driver configuration function in C (this is one-time pass through code)

\[
\text{ADC1CONTcnf(CHANNEL\_NUMBER, ACQ\_PRESCALER\_SETTING);}\]

Step2. Instantiate the INIT macro in assembly (this is one-time pass through code)

; Instantiate the init macro

\[
\text{ADC1CH\_DRV\_INIT}\]

Step3. Instantiate the run time macro in assembly (this is usually looped or ISR code)

; “call” the main macro

\[
\text{ADC1CH\_DRV}\]

Step4. (optional) Declare “Signal Nets” to “connect” the module to in “C”

\[
\text{int16 \ AdcValue;}
\]

; Note AdcValue can be simply a global integer variable

Step5. Declare the module “Terminal pointers” in “C”

// ADC1CH\_DRV terminal pointers, external references

\[
\text{extern int16 \ &ADC1CH\_Rslt;}
\]

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

// ADC1CH\_DRV connections

\[
\text{ADC1CH\_Rslt = &AdcValue;}
\]

// 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.

Figure 2. Connecting the one channel ADC driver