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 ........................................................................................................................3
  4.1 ADC3SYNCCnf ................................................................................................................................3
  4.2 ADC3SYNC_DRV_INIT ...................................................................................................................5
  4.3 ADC3SYNC_DRV ............................................................................................................................5
5 Usage Example: ...................................................................................................................................7

Table of Figures

Figure 1. Synchronous conversion, three channel ADC Driver ...............................................................2
Figure 2. Connecting the three channel ADC driver ....................................................................................7

Index of Tables

Table 1. ADC3SYNC_DRV module dependencies ......................................................................................2
Table 2. ADC3SYNC_DRV module components ..........................................................................................2
Table 3. ADC3SYNC_DRV module miscellaneous properties ...................................................................2
Table 4. ADC3SYNC_DRV module component files .................................................................................3
Table 5. ADC3SYNC_DRV module inputs ..................................................................................................3
Table 6. ADC3SYNC_DRV module outputs ...............................................................................................3
Table 7. ADC3SYNC_DRV channel selection .............................................................................................4
1 Overview

This software module controls the ADC on the F/C280x devices. It sets up the ADC in synchronous mode at CNT_zero event of the time-base of ePWM1, to convert three channels, and to make it available in a variable in the application.

2 Module Properties

This section describes module properties, such as compatible devices, components, invocation etc. The ADC3SYNC_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. ADC3SYNC_DRV module dependencies

The ADC3SYNC_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. ADC3SYNC_DRV module components

The ADC3SYNC_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. ADC3SYNC_DRV module miscellaneous properties
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>
</tr>
</thead>
<tbody>
<tr>
<td>ADCINAm</td>
<td>F280x/C280x ADC input pin</td>
<td>Analog sense value</td>
<td>See device datasheet for electrical specifications.</td>
</tr>
<tr>
<td>ADCINAn</td>
<td>F280x/C280x ADC input pin</td>
<td>Analog sense value</td>
<td>See device datasheet for electrical specifications.</td>
</tr>
<tr>
<td>ADCINAO</td>
<td>F280x/C280x ADC input pin</td>
<td>Analog sense value</td>
<td>See device datasheet for electrical specifications.</td>
</tr>
</tbody>
</table>

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>ADC3CH_Rslt</td>
<td>Driver output pointer</td>
<td>Pointer to 16-bit fixed point output data</td>
<td>Q15: [-1, 1] or [-32768, 32767]</td>
</tr>
</tbody>
</table>

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 ADC3SYNC_Cnf

Function Name: ADC3SYNC_Cnf
Prototype: void ADC3SYNC_Cnf(int mCh, int nCh, int oCh, int AcqWidth);
Return value: None.
Preconditions: The following preconditions must be satisfied:
The ADC module clock must be enabled in the PCLKCR0 register.

† 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.
The ADC3SYNCCnf function is called from the C environment, and performs driver configuration, including configuration of ADC registers.

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

- **nCh**: Channel to be converted to output 1.
  
  **Valid Range**: 0-15, corresponding to ADC channels A0-A7 and B0-B7. The channel selection is shown in Table 7 below.

- **oCh**: Channel to be converted to output 1.
  
  **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. ADC3SYNC_DRV channel selection*

**Example**: Call the ADC3SYNCCnf function to initialize EPWM1 module.

```
// Set up conversion of channels 0,1,2 with acq prescaler at 2 clocks
ADC3SYNCCnf(0,1,2,2);
```
4.2 ADC3SYNC_DRV_INIT

Function Name: ADC3SYNC_DRV_INIT
Prototype: ADC3SYNC_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 ADC3CH_DRV_INIT to initialize the ADC3CH_DRV module.

`;--------------------------------------------------------
`; ISR Initialisation
`;--------------------------------------------------------
/_ISR_Init: ADC3SYNC_DRV_INIT
LRETR

4.3 ADC3SYNC_DRV

Function Name: ADC3SYNC_DRV
Prototype: ADC3SYNC_DRV
Return value: None.
Preconditions: The following preconditions must be satisfied:
- The ADC module clock must be enabled in the PCLKCR0 register.
- C language init routine must be called.
- The ISR initialization macro ADC3SYNC_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 hardware output. The output is written to the ADC3CH_Rslt variable.

Example: Call the ADC3SYNC_DRV in an assembler ISR

`;--------------------------------------------------------
`; Runtime interrupt service routine
`;--------------------------------------------------------
/_ISR_Run: CONTEXT_SAVE ; call macro
ADC3SYNC_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:

![Diagram of three-channel ADC driver](image)

Figure 2. Connecting the three channel ADC driver

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

```c
ADC3SYNCConf(CH_Am_NO,CH_An_NO,CH_Ao_NO,ACQ_PRESCALER_SETTING);
```

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

```asm
; Instantiate the init macro
ADC3SYNC_DRV_INIT
```

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

```asm
; "call" the main macro
ADC3SYNC_DRV
```

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

```c
int16 AdcValue[3];
; Note AdcValue can be simply an global integer array variable
```

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

```c
// ADC3SYNC_DRV terminal pointers, external references
extern int16 *ADC3CH_Rslt;
```

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

```c
// ADC3SYNC_DRV connections
ADC3CH_Rslt = &AdcValue[0];
// 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.
```