PIC32 Analog to Digital Converter Interface Module

Documentation Rev. 0.1 10/27/20
Code Rev. 1.0.0 10/27/20

Purpose:

This module provides a set of interface functions to perform a simplified initialization of the A/D converter system on the PIC32MX170F256B microcontroller. The initialization function must be called before any use is made of the A/D converter. After the initialization is complete, reads of the analog input pins should be made using the `void ADC_MultiRead(uint32_t results[HowMany]))` function.

How to obtain:

TBD.

Usage:

To use the A/D library in your project, copy the `PIC32_AD_Lib.c` file into the in ProjectSource folder in your project and the `PIC32_AD_Lib.h` file into your ProjectHeaders folder, then add these two files into the Project window of your MPLAB X project in the *Source Files* and *Header Files* folders of the project window.
The file `PIC32_AD_Lib.c` must be included in any project wishing to use the functions provided by this module.
The header file `PIC32_AD_Lib.h` should be included in any module wishing to use the functions provided by this module.

Revision History:

October 27, 2020 First release of code and formal documentation. Based on c version 1.0 dates 10/27/20

Initialization

Function:

```
void ADC_ConfigAutoScan( uint16_t whichPins, uint8_t numPins)
```

Parameters:

`uint8_t whichPins spcifies which of the ANx pins will be converted a 1 in a bit position indicates that that ANx channel is to be converted e.g: to convert on AN0, set bit 0 to 1.`

`uint8_t numPins how many channels (pins) in the scan set.`

Returns:

`Nothing.`

Description:

Configures the A/D converter subsystem for auto-sampling on a set of pins for use in measuring analog inputs in the range of 0-3.3V. Once this function is called, the port pins configured for analog input can no longer be used for digital functions.

Notes:

Nothing more at this time

Usage:

```
ADC_ConfigAutoScan( (BIT4HI | BIT5HI), 2);
```

This call would initialize the A/D converter to convert on AN4 & AN5 (pins RB2 & RB3). It is up to you to be sure that the pins are configured as analog inputs (they default that way).

Analog Pin Read

Function:

```
void ADC_MultiRead(uint32_t *adcResults)
```

Parameters:

`a pointer to the first element of an array of type uint32_t with a number elements matching the number of channels initialized in the call to ADC_ConfigAutoScan().`
When writing your code, simply use the name of the results array as the parameter to this function. For this syntax to work, the results array needs to be declared as an array, even if it has only one element.

Returns:

Nothing

Description:

Reads the A/D conversion result (10 bits, 0-1023 corresponding to 0-3.3V) from the appropriate A/D converter register(s) and copies the values into the elements of the results array that was passed as the parameter.

Notes:

After this call, `adcResults[0]` will contain the conversion result from least significant ANx channel configured in `ADC_ConfigAutoScan()`. If more than one channel was configured, then the next most significant ANx results will be available in `adcResults[1], and so on`.

Usage:

```
// magic number used solely for clarity in the documentation. Do not do this in your code

uint32_t ConversionResults[3];

ADC_MultiRead(ConversionResults )
```

Assuming that the A/D converter had been initialized with `ADC_ConfigAutoScan(whichPins,3);` this call would read the analog values on the three ANx channels indicated in whichPins and copy the A/D conversion results into the first 3 elements of the ConversionResults array.