

# Proton+ PIC18 A/D Converter Macros

## Table of Contents

<b>Introduction .....</b>	<b>2</b>
<b>A/D Converter Macros .....</b>	<b>2</b>
<b>Version vs. Devices .....</b>	<b>2</b>
<b>BusyADC .....</b>	<b>3</b>
<b>CloseADC .....</b>	<b>3</b>
<b>ConvertADC .....</b>	<b>4</b>
<b>OpenADC .....</b>	<b>5</b>
OpenADC (ADC_V1) .....	5
OpenADC (ADC_V2) .....	7
OpenADC (ADC_V3, 4, 5, 6) .....	9
OpenADC (ADC_V7, V7_1) .....	12
OpenADC (ADC_V8, 9) .....	15
OpenADC (ADC_V10) .....	18
OpenADC (ADC_V11) .....	20
OpenADC (ADC_V12) .....	23
<b>ReadADC .....</b>	<b>25</b>
<b>SetChanADC .....</b>	<b>26</b>
<b>SelChanConvADC .....</b>	<b>27</b>
<b>Macro definitions .....</b>	<b>28</b>
<b>Example use of the A/D Converter Macros: .....</b>	<b>28</b>

# Proton+ PIC18 ADC Macros

## Introduction

The ADC peripheral on 18F devices is supported with the following macros. The macros are a mixture of compiler types and preprocessor types, and can be found in "Includes\Sources\ADC.inc"

## A/D Converter Macros

Macro	Description
<b>BusyADC</b>	Is A/D converter currently performing a conversion?
<b>CloseADC</b>	Disable the A/D converter.
<b>ConvertADC</b>	Start an A/D conversion.
<b>OpenADC</b>	Configure the A/D converter.
<b>ReadADC</b>	Read the results of an A/D conversion.
<b>SetChanADC</b>	Select A/D channel to be used.
<b>SelChanConvADC</b>	Select A/D channel to be used and start an A/D conversion.

Based on the different control registers, configuration bits and their positions in the control register, all PIC18 devices are divided into the following different versions. Wherever required, separate macros have been designed to support these versions, so before calling the macros care has to be taken to know the version of the configured device and to use the appropriate macro with the correct number of arguments. For ADC\_V11 user has to configure the I/O registers.

Below is the table to find the ADC version for the selected device:

## Version vs. Devices

Version name	Device number
ADC_V1	18F242, 18F252, 18F442, 18F452, 18F248 18F258, 18F448, 18F458, 18F2439, 18F2539, 18F4439, 18F4539
ADC_V2	18F6620, 18F6720, 18F8620, 18F8720, 18F6520, 18F8520
ADC_V3	18F1220, 18F1320
ADC_V4	18F1230, 18F1330
ADC_V5	18F2220, 18F2320, 18F4220, 18F4320, 18F2420, 18F2520, 18F4420, 18F4520, 18F2423, 18F2523, 18F4423, 18F4523, 18F2455, 18F2550, 18F4455, 18F4550, 18F2410, 18F2510, 18F2515, 18F2610, 18F4410, 18F4510, 18F4515, 18F4610, 18F2525, 18F2620, 18F4525, 18F4620, 18F6310, 18F6410 18F8310, 18F8410, 18F6390, 18F6490, 18F8390, 18F8490, 18F6527, 18F6622, 18F6627, 18F6722, 18F8527, 18F8622, 18F8627, 18F8722, 18F6585, 18F6680, 18F8585, 18F8680, 18F6525, 18F6621, 18F8525, 18F8621, 18F2450, 18F4450, 18F2480, 18F2580, 18F4480, 18F4580, 18F2585, 18F2680, 18F4585, 18F4680, 18F2682, 18F2685, 18F4682, 18F4685, 18F2221, 18F2321, 18F4221, 18F4321
ADC_V6	18F24J10, 18F25J10, 18F44J10, 18F45J10, 18F63J11, 18F64J11, 18F65J11, 18F83J11, 18F84J11, 18F85J11, 18F63J90, 18F64J90, 18F65J90, 18F83J90, 18F84J90, 18F85J90, 18F65J10, 18F65J15, 18F66J10, 18F66J15, 18F67J10, 18F85J10, 18F85J15, 18F86J10, 18F86J15, 18F87J10, 18F66J60, 18F66J65, 18F67J60, 18F86J60, 18F86J65, 18F87J60, 18F96J60, 18F96J65, 18F97J60
ADC_V7	18F4331, 18F4431
ADC_V7_1	18F2331, 18F2431
ADC_V8	18F23K20, 18F24K20, 18F25K20, 18F43K20, 18F44K20, 18F45K20
ADC_V9	18F66J11, 18F66J16, 18F67J11, 18F86J11, 18F86J16 18F87J11, 18F65J50, 18F66J50, 18F66J55, 18F67J50, 18F85J50, 18F86J50, 18F86J55, 18F87J50
ADC_V10	18F13K50, 18LF13K50, 18F14K50, 18LF14K50, 18F13K22, 18F14K22, 18LF13K22, 18LF14K22
ADC_V11	18F24J50, 18F25J50, 18F26J50, 18F44J50, 18F45J50, 18F46J50, 18F24J11, 18F25J11, 18F26J11, 18F44J11, 18F45J11, 18F46J11, 18F24J50, 18F25J50, 18F26J50, 18F44J50, 18F45J50, 18LF46J50, 18LF24J11, 18LF25J11, 18LF26J11, 18LF44J11, 18LF45J11, 18LF46J11
ADC_V12	18F66J90, 18F67J90, 18F86J90, 18F87J90

# Proton+ PIC18 ADC Macros

## BusyADC

**Function:** Is the A/D Converter currently performing a conversion?

**Include:** `adc.inc`

**Prototype:** `Var = BusyADC( )`

**Remarks:** This macro indicates if the A/D peripheral is in the process of converting a value.

**Return Value:** 1 if the A/D peripheral is performing a conversion.  
0 if the A/D peripheral isn't performing a conversion.

## CloseADC

**Function:** Disable the A/D Converter.

**Include:** `adc.inc`

**Prototype:** `CloseADC( )`

**Remarks:** This macro disables the A/D converter and A/D interrupt mechanism.

## Proton+ PIC18 ADC Macros

### ConvertADC

**Function:** Starts the A/D conversion process.

**Include:** `adc.inc`

**Prototype:** `ConvertADC( )`

**Remarks:** This macro starts an A/D conversion. The `BusyADC( )` macro or A/D interrupt may be used to detect completion of the conversion.

# Proton+ PIC18 ADC Macros

## OpenADC

### OpenADC (ADC\_V1)

For ADC\_V1

**Function:** Configure the A/D Converter.

**Include:** `adc.inc`

**Prototype:** `OpenADC(pConfig, pConfig2)`

**Arguments:** *PConfig*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

#### A/D clock source:

<code>ADC_FOSC_2</code>	<code>FOSC / 2</code>
<code>ADC_FOSC_4</code>	<code>FOSC / 4</code>
<code>ADC_FOSC_8</code>	<code>FOSC / 8</code>
<code>ADC_FOSC_16</code>	<code>FOSC / 16</code>
<code>ADC_FOSC_32</code>	<code>FOSC / 32</code>
<code>ADC_FOSC_64</code>	<code>FOSC / 64</code>
<code>ADC_FOSC_RC</code>	Internal RC Oscillator

#### A/D result justification:

<code>ADC_RIGHT_JUST</code>	Result in Least Significant bits
<code>ADC_LEFT_JUST</code>	Result in Most Significant bits

#### A/D voltage reference source:

<code>ADC_8ANA_0REF</code>	<code>VREF+ = VDD, VREF- = VSS, All analogue channels</code>
<code>ADC_7ANA_1REF</code>	<code>VREF+ = AN3, VREF- = VSS</code> All analogue channels except AN3
<code>ADC_6ANA_2REF</code>	<code>VREF+ = AN3, VREF- = AN2</code>
<code>ADC_6ANA_0REF</code>	<code>VREF+ = VDD, VREF- = VSS</code>
<code>ADC_5ANA_1REF</code>	<code>VREF+ = AN3, VREF- = VSS</code>
<code>ADC_5ANA_0REF</code>	<code>VREF+ = VDD, VREF- = VSS</code>
<code>ADC_4ANA_2REF</code>	<code>VREF+ = AN3, VREF- = AN2</code>
<code>ADC_4ANA_1REF</code>	<code>VREF+ = AN3, VREF- = VSS</code>
<code>ADC_3ANA_2REF</code>	<code>VREF+ = AN3, VREF- = AN2</code>
<code>ADC_3ANA_0REF</code>	<code>VREF+ = VDD, VREF- = VSS</code>
<code>ADC_2ANA_2REF</code>	<code>VREF+ = AN3, VREF- = AN2</code>
<code>ADC_2ANA_1REF</code>	<code>VREF+ = AN3,</code>
<code>ADC_1ANA_2REF</code>	<code>VREF+ = AN3, VREF- = AN2 (AN0 is analogue input)</code>
<code>ADC_1ANA_0REF</code>	<code>VREF+ = VDD, VREF- = VSS</code> AN0 is analogue input
<code>ADC_0ANA_0REF</code>	All digital I/O

# Proton+ PIC18 ADC Macros

## *pConfig2*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### Channel:

ADC_CH0	Channel 0
ADC_CH1	Channel 1
ADC_CH2	Channel 2
ADC_CH3	Channel 3
ADC_CH4	Channel 4
ADC_CH5	Channel 5
ADC_CH6	Channel 6
ADC_CH7	Channel 7

### A/D Interrupts:

ADC_INT_ON	Interrupts enabled
ADC_INT_OFF	Interrupts disabled

**Remarks:** This macro resets the A/D peripheral to the POR state and Configures the A/D-related Special Function Registers (SFRs) according to the options specified.

**Example:** `OpenADC(ADC_FOSC_32 & ADC_RIGHT_JUST & ADC_1ANA_0REF, ADC_CH0  
& ADC _INT_OFF)`

# Proton+ PIC18 ADC Macros

## OpenADC (ADC\_V2)

For ADC\_V2

**Function:** Configure the A/D Converter.

**Include:** `adc.inc`

**Prototype:** `OpenADC(pConfig, pConfig2)`

**Arguments:** *pConfig*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### A/D clock source:

ADC_FOSC_2	FOSC / 2
ADC_FOSC_4	FOSC / 4
ADC_FOSC_8	FOSC / 8
ADC_FOSC_16	FOSC / 16
ADC_FOSC_32	FOSC / 32
ADC_FOSC_64	FOSC / 64
ADC_FOSC_RC	Internal RC Oscillator

### A/D result justification:

ADC_RIGHT_JUST	Result in Least Significant bits
ADC_LEFT_JUST	Result in Most Significant bits

### A/D port Configuration:

ADC_0ANA	All digital	
ADC_1ANA	analogue:AN0	digital:AN1-AN15
ADC_2ANA	analogue:AN0-AN1	digital:AN2-AN15
ADC_3ANA	analogue:AN0-AN2	digital:AN3-AN15
ADC_4ANA	analogue:AN0-AN3	digital:AN4-AN15
ADC_5ANA	analogue:AN0-AN4	digital:AN5-AN15
ADC_6ANA	analogue:AN0-AN5	digital:AN6-AN15
ADC_7ANA	analogue:AN0-AN6	digital:AN7-AN15
ADC_8ANA	analogue:AN0-AN7	digital:AN8-AN15
ADC_9ANA	analogue:AN0-AN8	digital:AN9-AN15
ADC_10ANA	analogue:AN0-AN9	digital:AN10-AN15
ADC_11ANA	analogue:AN0-AN10	digital:AN11-AN15
ADC_12ANA	analogue:AN0-AN11	digital:AN12-AN15
ADC_13ANA	analogue:AN0-AN12	digital:AN13-AN15
ADC_14ANA	analogue:AN0-AN13	digital:AN14-AN15
ADC_15ANA	All analogue	

# Proton+ PIC18 ADC Macros

## *pConfig2*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### Channel:

ADC_CH0	Channel 0
ADC_CH1	Channel 1
ADC_CH2	Channel 2
ADC_CH3	Channel 3
ADC_CH4	Channel 4
ADC_CH5	Channel 5
ADC_CH6	Channel 6
ADC_CH7	Channel 7
ADC_CH8	Channel 8
ADC_CH9	Channel 9
ADC_CH10	Channel 10
ADC_CH11	Channel 11
ADC_CH12	Channel 12
ADC_CH13	Channel 13
ADC_CH14	Channel 14
ADC_CH15	Channel 15

### A/D Interrupts:

ADC_INT_ON	Interrupts enabled
ADC_INT_OFF	Interrupts disabled

### A/D VREF+ and VREF- Configuration:

ADC_REF_VDD_VREFMINUS	VREF+ = VDD & VREF- = Ext.
ADC_REF_VREFPLUS_VREFMINUS	VREF+ = Ext. & VREF- = Ext.
ADC_REF_VREFPLUS_VSS	VREF+ = Ext. & VREF- = VSS
ADC_REF_VDD_VSS	VREF+ = VDD & VREF- = VSS

**Remarks:** This macro resets the ADC related registers to the POR state and then configures the clock, result format, voltage reference, port and channel.

**Example:** `OpenADC(ADC_FOSC_32 & ADC_RIGHT_JUST & ADC_14ANA, ADC_CH0)`



# Proton+ PIC18 ADC Macros

## OpenADC (ADC\_V3, 4, 5, 6)

For **ADC\_V3**, **ADC\_V4**, **ADC\_V5** and **ADC\_V6**

**Function:** Configure the A/D Converter.

**Include:** `adc.inc`

**Prototype:** `OpenADC(pConfig, pConfig2, pPortConfig)`

**Arguments:** *pConfig*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### A/D clock source:

<code>ADC_FOSC_2</code>	<code>FOSC / 2</code>
<code>ADC_FOSC_4</code>	<code>FOSC / 4</code>
<code>ADC_FOSC_8</code>	<code>FOSC / 8</code>
<code>ADC_FOSC_16</code>	<code>FOSC / 16</code>
<code>ADC_FOSC_32</code>	<code>FOSC / 32</code>
<code>ADC_FOSC_64</code>	<code>FOSC / 64</code>
<code>ADC_FOSC_RC</code>	Internal RC Oscillator

### A/D result justification:

<code>ADC_RIGHT_JUST</code>	Result in Least Significant bits
<code>ADC_LEFT_JUST</code>	Result in Most Significant bits

### A/D acquisition time select:

<code>ADC_0_TAD</code>	0 Tad
<code>ADC_2_TAD</code>	2 Tad
<code>ADC_4_TAD</code>	4 Tad
<code>ADC_6_TAD</code>	6 Tad
<code>ADC_8_TAD</code>	8 Tad
<code>ADC_12_TAD</code>	12 Tad
<code>ADC_16_TAD</code>	16 Tad
<code>ADC_20_TAD</code>	20 Tad

### *pConfig2*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### Channel:

For **ADC\_V3**:

<code>ADC_CH0</code>	Channel 0
<code>ADC_CH1</code>	Channel 1
<code>ADC_CH2</code>	Channel 2
<code>ADC_CH3</code>	Channel 3
<code>ADC_CH4</code>	Channel 4

## Proton+ PIC18 ADC Macros

ADC\_CH5      Channel 5  
ADC\_CH6      Channel 6

### For **ADC\_V4**:

ADC\_CH0      Channel 0  
ADC\_CH1      Channel 1  
ADC\_CH2      Channel 2  
ADC\_CH3      Channel 3

### For **ADC\_V5** and **ADC\_V6**:

ADC\_CH0      Channel 0  
ADC\_CH1      Channel 1  
ADC\_CH2      Channel 2  
ADC\_CH3      Channel 3  
ADC\_CH4      Channel 4  
ADC\_CH5      Channel 5  
ADC\_CH6      Channel 6  
ADC\_CH7      Channel 7  
ADC\_CH8      Channel 8  
ADC\_CH9      Channel 9  
ADC\_CH10      Channel 10  
ADC\_CH11      Channel 11  
ADC\_CH12      Channel 12  
ADC\_CH13      Channel 13  
ADC\_CH14      Channel 14  
ADC\_CH15      Channel 15

### **A/D Interrupts:**

ADC\_INT\_ON    Interrupts enabled  
ADC\_INT\_OFF   Interrupts disabled

### **A/D Vref+ and Vref- Configuration:**

ADC_REF_VDD_VREFMINUS	VREF+ = VDD & VREF- = Ext.
ADC_REF_VREFPLUS_VREFMINUS	VREF+ = Ext. & VREF- = Ext.
ADC_REF_VREFPLUS_VSS	VREF+ = Ext. & VREF- = VSS
ADC_REF_VDD_VSS	VREF+ = VDD & VREF- = VSS

### ***pPortConfig***

#### For **ADC\_V3**:

The value of pPortConfig can be any value from 0 to 127, few are defined below

ADC\_0ANA      All digital  
ADC\_1ANA      analogue:AN0  
ADC\_2ANA      analogue:AN0-AN1  
ADC\_3ANA      analogue:AN0-AN2  
ADC\_4ANA      analogue:AN0-AN3  
ADC\_5ANA      analogue:AN0-AN4  
ADC\_6ANA      analogue:AN0-AN5  
ADC\_7ANA      analogue:AN0-AN6

# Proton+ PIC18 ADC Macros

## For ADC\_V4:

The value of pPortConfig can be any value from 0 to 15, few are defined below

ADC_0ANA	All digital
ADC_1ANA	analogue:AN0
ADC_2ANA	analogue:AN0-AN1
ADC_3ANA	analogue:AN0-AN2
ADC_4ANA	analogue:AN0-AN3

## For ADC\_V5 and ADC\_V6:

ADC_0ANA	All digital	
ADC_1ANA	analogue:AN0	digital:AN1-AN15
ADC_2ANA	analogue:AN0-AN1	digital:AN2-AN15
ADC_3ANA	analogue:AN0-AN2	digital:AN3-AN15
ADC_4ANA	analogue:AN0-AN3	digital:AN4-AN15
ADC_5ANA	analogue:AN0-AN4	digital:AN5-AN15
ADC_6ANA	analogue:AN0-AN5	digital:AN6-AN15
ADC_7ANA	analogue:AN0-AN6	digital:AN7-AN15
ADC_8ANA	analogue:AN0-AN7	digital:AN8-AN15
ADC_9ANA	analogue:AN0-AN8	digital:AN9-AN15
ADC_10ANA	analogue:AN0-AN9	digital:AN10-AN15
ADC_11ANA	analogue:AN0-AN10	digital:AN11-AN15
ADC_12ANA	analogue:AN0-AN11	digital:AN12-AN15
ADC_13ANA	analogue:AN0-AN12	digital:AN13-AN15
ADC_14ANA	analogue:AN0-AN13	digital:AN14-AN15
ADC_15ANA	All analogue	

**Remarks:** This macro resets the A/D-related registers to the POR state and then Configures the clock, result format, voltage reference, port and channel.

**Example:** `OpenADC(ADC_FOSC_32 & ADC_RIGHT_JUST & ADC_12_TAD, _  
ADC_CH0 & ADC_REF_VDD_VSS & ADC_INT_OFF, _  
ADC_12ANA)`

# Proton+ PIC18 ADC Macros

## OpenADC (ADC\_V7, V7\_1)

For **ADC\_V7** and **ADC\_V7\_1**

**Function:** Configure the A/D Converter.

**Include:** `adc.inc`

**Prototype:** **OpenADC**(*pConfig1*, *pConfig2*, *pConfig3*)

**Arguments:** *pConfig1*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### A/D conversion type:

`ADC_CONV_CONTINUOUS`  
`ADC_CONV_SINGLE_SHOT`

### A/D conversion mode:

`ADC_MODE_MULTI_CH`  
`ADC_MODE_SINGLE_CH`

### A/D conversion sequence select:

<code>ADC_CONV_SEQ_SEQM1</code>	Sequence mode 1
<code>ADC_CONV_SEQ_SEQM2</code>	Sequence mode 2
<code>ADC_CONV_SEQ_STNM1</code>	Simultaneous mode 1
<code>ADC_CONV_SEQ_STNM2</code>	Simultaneous mode 2

### A/D result buffer depth Interrupt select control:

<code>INT_EACH_WR_BUF</code>	Interrupt when each is written to BUF
<code>INT_2_4_WR_BUF</code>	Interrupt at 2nd & 4th words are written to BUF
<code>INT_4_WR_BUF</code>	Interrupt at 4th word is written to BUF

### A/D Interrupts:

<code>ADC_INT_ON</code>	Interrupts enabled
<code>ADC_INT_OFF</code>	Interrupts disabled

### *pConfig2*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### A/D Vref+ and Vref- Configuration:

<code>ADC_REF_VDD_VREFMINUS</code>	VREF+ = VDD & VREF- = Ext.
<code>ADC_REF_VREFPLUS_VREFMINUS</code>	VREF+ = Ext. & VREF- = Ext.
<code>ADC_REF_VREFPLUS_VSS</code>	VREF+ = Ext. & VREF- = VSS
<code>ADC_REF_VDD_VSS</code>	VREF+ = VDD & VREF- = VSS

# Proton+ PIC18 ADC Macros

## A/D FIFO buffer control:

ADC\_FIFO\_EN  
ADC\_FIFO\_DIS

## A/D Trigger Source:

ADC_TRIG_EXT_INT0	External INT0 starts A/D
ADC_TRIG_TMR_5	TMR5 starts A/D
ADC_TRIG_INP_CAP	Input Capture starts A/D
ADC_TRIG_CCP2_COM	CCP2 Compare starts A/D
ADC_TRIG_PCPWM	PCPWM rising edge starts A/D

## PConfig3

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

## A/D result justification:

ADC_RIGHT_JUST	Result in Least Significant bits
ADC_LEFT_JUST	Result in Most Significant bits

## A/D acquisition time select:

ADC_0_TAD	0 Tad
ADC_2_TAD	2 Tad
ADC_4_TAD	4 Tad
ADC_6_TAD	6 Tad
ADC_8_TAD	8 Tad
ADC_10_TAD	10 Tad
ADC_12_TAD	12 Tad
ADC_16_TAD	16 Tad
ADC_20_TAD	20 Tad
ADC_24_TAD	24 Tad
ADC_28_TAD	28 Tad
ADC_32_TAD	32 Tad
ADC_36_TAD	36 Tad
ADC_40_TAD	40 Tad
ADC_48_TAD	48 Tad
ADC_64_TAD	64 Tad

## A/D clock source:

ADC_FOSC_2	FOSC / 2
ADC_FOSC_4	FOSC / 4
ADC_FOSC_8	FOSC / 8
ADC_FOSC_16	FOSC / 16
ADC_FOSC_32	FOSC / 32
ADC_FOSC_64	FOSC / 64
ADC_FOSC_RC	Internal RC Oscillator

# Proton+ PIC18 ADC Macros

## A/D Channel selection:

### Channel from group A

<code>ADC_CH_GRA_AN0 ( )</code>	select AN0 as analogue
<code>ADC_CH_GRA_AN4 ( )</code>	select AN4 as analogue
<code>ADC_CH_GRA_AN8 ( )</code>	select AN8 as analogue

### Channel from group B

<code>ADC_CH_GRB_AN1 ( )</code>	select AN1 as analogue
<code>ADC_CH_GRB_AN5 ( )</code>	select AN5 as analogue

### Channel from group C

<code>ADC_CH_GRC_AN2 ( )</code>	select AN2 as analogue
<code>ADC_CH_GRC_AN6 ( )</code>	select AN6 as analogue

### Channel from group D

<code>ADC_CH_GRD_AN3 ( )</code>	select AN3 as analogue
<code>ADC_CH_GRD_AN7 ( )</code>	select AN7 as analogue
<code>ALL_CH_DIGITAL ( )</code>	Make all Channels Digital

**Note:** AN5 to AN8 are not available in PIC18F2X31 devices.

**Remarks:** This macro Configures the clock, result format, voltage reference, conversion type, conversion mode, triggering sources, FIFO buffer control and conversion sequence select.

**Example:** `OpenADC(ADC_CONV_CONTINUOUS & ADC_MODE_MULTI_CH & ADC_CONV_SEQ_SEQM2, _  
ADC_REF_VDD_VSS & ADC_FIFO_DIS & ADC_TRIG_INP_CAP, _  
ADC_RIGHT_JUST & ADC_10_TAD & ADC_FOSC_8)`

# Proton+ PIC18 ADC Macros

## OpenADC (ADC\_V8, 9)

For **ADC\_V8** and **ADC\_V9**

**Function:** Configure the A/D Converter.

**Include:** `adc.inc`

**Prototype:** **OpenADC**(pConfig, pConfig2, pPortConfig)

**Arguments:** *pConfig*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### A/D clock source:

ADC_FOSC_2	FOSC / 2
ADC_FOSC_4	FOSC / 4
ADC_FOSC_8	FOSC / 8
ADC_FOSC_16	FOSC / 16
ADC_FOSC_32	FOSC / 32
ADC_FOSC_64	FOSC / 64
ADC_FOSC_RC	Internal RC Oscillator

### A/D result justification:

ADC_RIGHT_JUST	Result in Least Significant bits
ADC_LEFT_JUST	Result in Most Significant bits

### A/D acquisition time select:

ADC_0_TAD	0 Tad
ADC_2_TAD	2 Tad
ADC_4_TAD	4 Tad
ADC_6_TAD	6 Tad
ADC_8_TAD	8 Tad
ADC_12_TAD	12 Tad
ADC_16_TAD	16 Tad
ADC_20_TAD	20 Tad

### *pConfig2*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### Channel:

For **ADC\_V8**:

ADC_CH0	Channel 0
ADC_CH1	Channel 1
ADC_CH2	Channel 2
ADC_CH3	Channel 3
ADC_CH4	Channel 4

## Proton+ PIC18 ADC Macros

ADC_CH5	Channel 5
ADC_CH6	Channel 6
ADC_CH7	Channel 7
ADC_CH8	Channel 8
ADC_CH9	Channel 9
ADC_CH10	Channel 10
ADC_CH11	Channel 11
ADC_CH12	Channel 12

### For ADC\_V9:

ADC_CH0	Channel 0
ADC_CH1	Channel 1
ADC_CH2	Channel 2
ADC_CH3	Channel 3
ADC_CH4	Channel 4
ADC_CH5	<b>Unused Channel</b>
ADC_CH6	Channel 6
ADC_CH7	Channel 7
ADC_CH8	Channel 8
ADC_CH9	Channel 9
ADC_CH10	Channel 10
ADC_CH11	Channel 11
ADC_CH12	Channel 12
ADC_CH13	Channel 13
ADC_CH14	Channel 14
ADC_CH15	Channel 15

### A/D Interrupts:

ADC_INT_ON	Interrupts enabled
ADC_INT_OFF	Interrupts disabled

### A/D Vref+ and Vref- Configuration:

ADC_REF_VDD_VREFMINUS	VREF+ = VDD & VREF- = Ext.
ADC_REF_VREFPLUS_VREFMINUS	VREF+ = Ext. & VREF- = Ext.
ADC_REF_VREFPLUS_VSS	VREF+ = Ext. & VREF- = VSS
ADC_REF_VDD_VSS	VREF+ = VDD & VREF- = VSS



# Proton+ PIC18 ADC Macros

## *pPortConfig*

### For ADC\_V8:

The pPortConfig can have 8192 different combination, few are defined below

ADC_0ANA	All digital
ADC_1ANA	analogue:AN0
ADC_2ANA	analogue:AN0-AN1
ADC_3ANA	analogue:AN0-AN2
ADC_4ANA	analogue:AN0-AN3
ADC_5ANA	analogue:AN0-AN4
ADC_6ANA	analogue:AN0-AN5
ADC_7ANA	analogue:AN0-AN6
ADC_8ANA	analogue:AN0-AN7
ADC_9ANA	analogue:AN0-AN8
ADC_10ANA	analogue:AN0-AN9
ADC_11ANA	analogue:AN0-AN10
ADC_12ANA	analogue:AN0-AN11

### For ADC\_V9:

The pPortConfig can have 16384 different combinations, few are defined below

ADC_0ANA	All digital
ADC_1ANA	analogue:AN0
ADC_2ANA	analogue:AN0-AN1
ADC_3ANA	analogue:AN0-AN2
ADC_4ANA	analogue:AN0-AN3
ADC_5ANA	analogue:AN0-AN4
ADC_6ANA	analogue:AN0-AN6
ADC_7ANA	analogue:AN0-AN7
ADC_8ANA	analogue:AN0-AN8
ADC_9ANA	analogue:AN0-AN9
ADC_10ANA	analogue:AN0-AN10
ADC_11ANA	analogue:AN0-AN11
ADC_12ANA	analogue:AN0-AN12
ADC_13ANA	analogue:AN0-AN13
ADC_14ANA	analogue:AN0-AN14
ADC_15ANA	All analogue

**Remarks:** This macro resets the A/D-related registers to the POR state and then Configures the clock, result format, voltage reference, port and channel.

**Example:** `OpenADC(ADC_FOSC_32 & ADC_RIGHT_JUST & ADC_12_TAD, _  
ADC_CH0 & ADC_REF_VDD_VSS & ADC_INT_ON, _  
ADC_10ANA)`

# Proton+ PIC18 ADC Macros

## OpenADC (ADC\_V10)

For ADC\_V10

**Function:** Configure the A/D Converter.

**Include:** `adc.inc`

**Prototype:** `OpenADC(pConfig, pConfig2, pConfig3, pPortConfig)`

**Arguments:** *pConfig*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### A/D clock source:

<code>ADC_FOSC_2</code>	<code>FOSC / 2</code>
<code>ADC_FOSC_4</code>	<code>FOSC / 4</code>
<code>ADC_FOSC_8</code>	<code>FOSC / 8</code>
<code>ADC_FOSC_16</code>	<code>FOSC / 16</code>
<code>ADC_FOSC_32</code>	<code>FOSC / 32</code>
<code>ADC_FOSC_64</code>	<code>FOSC / 64</code>
<code>ADC_FOSC_RC</code>	Internal RC Oscillator

### A/D result justification:

<code>ADC_RIGHT_JUST</code>	Result in Least Significant bits
<code>ADC_LEFT_JUST</code>	Result in Most Significant bits

### A/D acquisition time select:

<code>ADC_0_TAD</code>	0 Tad
<code>ADC_2_TAD</code>	2 Tad
<code>ADC_4_TAD</code>	4 Tad
<code>ADC_6_TAD</code>	6 Tad
<code>ADC_8_TAD</code>	8 Tad
<code>ADC_12_TAD</code>	12 Tad
<code>ADC_16_TAD</code>	16 Tad
<code>ADC_20_TAD</code>	20 Tad

### *pConfig2*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### Channel:

<code>ADC_CH3</code>	Channel 3
<code>ADC_CH4</code>	Channel 4
<code>ADC_CH5</code>	Channel 5
<code>ADC_CH6</code>	Channel 6
<code>ADC_CH7</code>	Channel 7
<code>ADC_CH8</code>	Channel 8
<code>ADC_CH9</code>	Channel 9

# Proton+ PIC18 ADC Macros

ADC\_CH10      Channel 10  
ADC\_CH11      Channel 11

## Misc options:

TEMP\_REF\_BANDGAP      Temperature reference from BandGap  
DAC1                  DAC  
FVR1                  Fixed Voltage reference

## A/D Interrupts:

ADC\_INT\_ON      Interrupts enabled  
ADC\_INT\_OFF      Interrupts disabled

## *pConfig3*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

## A/D Vref+ and Vref- Configuration:

ADC\_REF\_VDD\_VDD      VREF+ = AVDD  
ADC\_REF\_VDD\_VREFPLUS      VREF+ = ext. source at VREF+  
ADC\_REF\_VDD\_FVREF      VREF+ = FVREF+  
ADC\_REF\_VDD\_VSS      VREF- = AVSS  
ADC\_REF\_VDD\_VREFMINUS      VREF- = ext. source at VREF-

## *pPortConfig*

The pPortConfig can have 16384 different combinations, few are defined below

ADC\_0ANA      All digital  
ADC\_1ANA      analogue:AN0  
ADC\_2ANA      analogue:AN0-AN1  
ADC\_3ANA      analogue:AN0-AN2  
ADC\_4ANA      analogue:AN0-AN3  
ADC\_5ANA      analogue:AN0-AN4  
ADC\_6ANA      analogue:AN0-AN6  
ADC\_7ANA      analogue:AN0-AN7  
ADC\_8ANA      analogue:AN0-AN8  
ADC\_9ANA      analogue:AN0-AN9  
ADC\_10ANA      analogue:AN0-AN10  
ADC\_11ANA      analogue:AN0-AN11  
ADC\_12ANA      analogue:AN0-AN12  
ADC\_13ANA      analogue:AN0-AN13  
ADC\_14ANA      analogue:AN0-AN14  
ADC\_15ANA      All analogue

**Remarks:**      This macro resets the A/D-related registers to the POR state and then Configures the clock, result format, voltage reference, port and channel.

**Example:**      `OpenADC(ADC_FOSC_32 & ADC_RIGHT_JUST & ADC_12_TAD, _  
                  ADC_CH0 & ADC_REF_VDD_VSS & ADC_INT_ON, _  
                  ADC_10ANA)`

# Proton+ PIC18 ADC Macros

## OpenADC (ADC\_V11)

For **ADC\_V11**

**Function:** Configure the A/D Converter.

**Include:** `adc.inc`

**Prototype:** **OpenADC**(*pConfig*, *pConfig2*, *pConfig3*)

**Arguments:** *pConfig*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### A/D clock source:

<code>ADC_FOSC_2</code>	<code>FOSC / 2</code>
<code>ADC_FOSC_4</code>	<code>FOSC / 4</code>
<code>ADC_FOSC_8</code>	<code>FOSC / 8</code>
<code>ADC_FOSC_16</code>	<code>FOSC / 16</code>
<code>ADC_FOSC_32</code>	<code>FOSC / 32</code>
<code>ADC_FOSC_64</code>	<code>FOSC / 64</code>
<code>ADC_FOSC_RC</code>	Internal RC Oscillator

### A/D result justification:

<code>ADC_RIGHT_JUST</code>	Result in Least Significant bits
<code>ADC_LEFT_JUST</code>	Result in Most Significant bits

### A/D acquisition time select:

<code>ADC_0_TAD</code>	0 Tad
<code>ADC_2_TAD</code>	2 Tad
<code>ADC_4_TAD</code>	4 Tad
<code>ADC_6_TAD</code>	6 Tad
<code>ADC_8_TAD</code>	8 Tad
<code>ADC_12_TAD</code>	12 Tad
<code>ADC_16_TAD</code>	16 Tad
<code>ADC_20_TAD</code>	20 Tad

### *pConfig2*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### Channel:

<code>ADC_CH0</code>	Channel 0
<code>ADC_CH1</code>	Channel 1
<code>ADC_CH2</code>	Channel 2
<code>ADC_CH3</code>	Channel 3
<code>ADC_CH4</code>	Channel 4
<code>ADC_CH5</code>	Channel 5

# Proton+ PIC18 ADC Macros

ADC_CH6	Channel 6
ADC_CH7	Channel 7
ADC_CH8	Channel 8
ADC_CH9	Channel 9
ADC_CH10	Channel 10
ADC_CH11	Channel 11
ADC_CH12	Channel 12
ADC_CH_CTMU	Channel 13
ADC_CH_VDDCORE	Channel 14
ADC_CH_VBG	Channel 15

## A/D Interrupts:

ADC_INT_ON	Interrupts enabled
ADC_INT_OFF	Interrupts disabled

## A/D Vref+ and Vref- Configuration:

ADC_REF_VDD_VREFMINUS	VREF+ = VDD & VREF- = Ext.
ADC_REF_VREFPLUS_VREFMINUS	VREF+ = Ext. & VREF- = Ext.
ADC_REF_VREFPLUS_VSS	VREF+ = Ext. & VREF- = VSS
ADC_REF_VDD_VSS	VREF+ = VDD & VREF- = VSS

## *pConfig3*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

## *pPortConfig*

ADC_0ANA	All digital	
ADC_1ANA	analogue:AN0	digital:AN1-AN13
ADC_2ANA	analogue:AN0-AN1	digital:AN2-AN13
ADC_3ANA	analogue:AN0-AN2	digital:AN3-AN13
ADC_4ANA	analogue:AN0-AN3	digital:AN4-AN13
ADC_5ANA	analogue:AN0-AN4	digital:AN5-AN13
ADC_6ANA	analogue:AN0-AN5	digital:AN6-AN13
ADC_7ANA	analogue:AN0-AN6	digital:AN7-AN13
ADC_8ANA	analogue:AN0-AN7	digital:AN8-AN13
ADC_9ANA	analogue:AN0-AN8	digital:AN9-AN13
ADC_10ANA	analogue:AN0-AN9	digital:AN10-AN13
ADC_11ANA	analogue:AN0-AN10	digital:AN11-AN13
ADC_12ANA	analogue:AN0-AN11	digital:AN12-AN13
ADC_13ANA	analogue:AN0-AN12	

## *Band Gap selection*

ADC_VBG_ON	VBG output enabled
ADC_VBG_OFF	VBG output disabled
ADC_VBG2_ON	VBG/2 output enabled
ADC_VBG2_OFF	VBG/2 output disabled

## Proton+ PIC18 ADC Macros

**Remarks:** This macro resets the A/D-related registers to the POR state and then Configures the clock, result format, voltage reference, port and channel.

**Example:** `OpenADC(ADC_FOSC_32 & ADC_RIGHT_JUST & ADC_12_TAD, _  
ADC_CH0 & ADC_REF_VDD_VSS & ADC_INT_OFF, ADC_12ANA)`

# Proton+ PIC18 ADC Macros

## ***OpenADC (ADC\_V12)***

For **ADC\_V12**

**Function:** Configure the A/D Converter.

**Include:** `adc.inc`

**Prototype:** **OpenADC**(*pConfig*, *pConfig2*, *pConfig3*)

**Arguments:** *pConfig*

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### **A/D clock source:**

<code>ADC_FOSC_2</code>	<code>FOSC / 2</code>
<code>ADC_FOSC_4</code>	<code>FOSC / 4</code>
<code>ADC_FOSC_8</code>	<code>FOSC / 8</code>
<code>ADC_FOSC_16</code>	<code>FOSC / 16</code>
<code>ADC_FOSC_32</code>	<code>FOSC / 32</code>
<code>ADC_FOSC_64</code>	<code>FOSC / 64</code>
<code>ADC_FOSC_RC</code>	Internal RC Oscillator

### **A/D result justification:**

<code>ADC_RIGHT_JUST</code>	Result in Least Significant bits
<code>ADC_LEFT_JUST</code>	Result in Most Significant bits

### **A/D acquisition time select:**

<code>ADC_0_TAD</code>	0 Tad
<code>ADC_2_TAD</code>	2 Tad
<code>ADC_4_TAD</code>	4 Tad
<code>ADC_6_TAD</code>	6 Tad
<code>ADC_8_TAD</code>	8 Tad
<code>ADC_12_TAD</code>	12 Tad
<code>ADC_16_TAD</code>	16 Tad
<code>ADC_20_TAD</code>	20 Tad

### ***pConfig2***

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

### **Channel:**

<code>ADC_CH0</code>	Channel 0
<code>ADC_CH1</code>	Channel 1
<code>ADC_CH2</code>	Channel 2
<code>ADC_CH3</code>	Channel 3
<code>ADC_CH4</code>	Channel 4
<code>ADC_CH5</code>	Channel 5

# Proton+ PIC18 ADC Macros

ADC_CH6	Channel 6
ADC_CH7	Channel 7
ADC_CH8	Channel 8
ADC_CH9	Channel 9
ADC_CH10	Channel 10
ADC_CH11	Channel 11

## A/D Interrupts:

ADC_INT_ON	Interrupts enabled
ADC_INT_OFF	Interrupts disabled

## A/D Vref+ and Vref- Configuration:

ADC_REF_VDD_VREFMINUS	VREF+ = VDD & VREF- = Ext.
ADC_REF_VREFPLUS_VREFMINUS	VREF+ = Ext. & VREF- = Ext.
ADC_REF_VREFPLUS_VSS	VREF+ = Ext. & VREF- = VSS
ADC_REF_VDD_VSS	VREF+ = VDD & VREF- = VSS

## pConfig3

A bitmask that is created by performing a bitwise AND operation ('&'), as shown in the example at the end of this document, with a value from each of the categories listed below. These values are defined in the file `adcdefs.inc`.

## Special Trigger Configuration:

ADC_TRIG_CTMU	Special Trigger from CTMU
ADC_TRIG_CCP2	Special Trigger from CCP2

## A/D port Configuration:

ADC_0ANA	All digital	
ADC_1ANA	analogue:AN0	digital:AN1-AN15
ADC_2ANA	analogue:AN0-AN1	digital:AN2-AN15
ADC_3ANA	analogue:AN0-AN2	digital:AN3-AN15
ADC_4ANA	analogue:AN0-AN3	digital:AN4-AN15
ADC_5ANA	analogue:AN0-AN4	digital:AN5-AN15
ADC_6ANA	analogue:AN0-AN5	digital:AN6-AN15
ADC_7ANA	analogue:AN0-AN6	digital:AN7-AN15
ADC_8ANA	analogue:AN0-AN7	digital:AN8-AN15
ADC_9ANA	analogue:AN0-AN8	digital:AN9-AN15
ADC_10ANA	analogue:AN0-AN9	digital:AN10-AN15
ADC_11ANA	analogue:AN0-AN10	digital:AN11-AN15

**Remarks:** This macro resets the A/D-related registers to the POR state and then Configures the clock, result format, voltage reference, port, special trigger and channel.

**Example:** `OpenADC(ADC_FOSC_32 & ADC_RIGHT_JUST & ADC_12_TAD, _  
ADC_CH0 & ADC_REF_VDD_VSS & ADC_INT_OFF, _  
ADC_TRIG_CTMU & ADC_1ANA)`



# Proton+ PIC18 ADC Macros

## ReadADC

<b>Function:</b>	Read the result of an A/D conversion.
<b>Include:</b>	<code>adc.inc</code>
<b>Prototype:</b>	<code>Var = ReadADC( )</code>
<b>Remarks:</b>	This macro reads the Word (10 or 12 bit) result of an A/D conversion.
<b>Return Value:</b>	This macro returns the Word (10 or 12-bit) result of the A/D conversion. Based on the Configuration of the A/D converter (e.g., using the <code>OpenADC( )</code> macro).

# Proton+ PIC18 ADC Macros

## SetChanADC

This macro **not** supported with **ADC\_V7** and **ADC\_V7\_1**.

**Function:** Select the channel used as input to the A/D Converter.

**Include:** `adc.inc`

**Prototype:** `SetChanADC(pChannel)`

**Arguments:** *pChannel*

One of the following values (defined in `adcdefs.inc`):

<code>ADC_CH0</code>	Channel 0
<code>ADC_CH1</code>	Channel 1
<code>ADC_CH2</code>	Channel 2
<code>ADC_CH3</code>	Channel 3
<code>ADC_CH4</code>	Channel 4
<code>ADC_CH5</code>	Channel 5
<code>ADC_CH6</code>	Channel 6
<code>ADC_CH7</code>	Channel 7
<code>ADC_CH8</code>	Channel 8
<code>ADC_CH9</code>	Channel 9
<code>ADC_CH10</code>	Channel 10
<code>ADC_CH11</code>	Channel 11
<code>ADC_CH12</code>	Channel 12
<code>ADC_CH13</code>	Channel 13
<code>ADC_CH14</code>	Channel 14
<code>ADC_CH15</code>	Channel 15
<code>ADC_CH_CTMU</code>	Channel 13
<code>ADC_CH_VDDCORE</code>	Channel 14
<code>ADC_CH_VBG</code>	Channel 15

**Remarks:** Selects the pin that will be used as input to the A/D Converter.

**Example:** `SetChanADC(ADC_CH0)`

# Proton+ PIC18 ADC Macros

## SelChanConvADC

**Function:** Select the channel used as input to the A/D Converter and starts the A/D conversion process.

**Include:** `adc.inc`

**Prototype:** `SelChanConvADC(pChannel)`

**Arguments:** *pChannel*

One of the following values (defined in `adcdefs.inc`):

<code>ADC_CH0</code>	Channel 0
<code>ADC_CH1</code>	Channel 1
<code>ADC_CH2</code>	Channel 2
<code>ADC_CH3</code>	Channel 3
<code>ADC_CH4</code>	Channel 4
<code>ADC_CH5</code>	Channel 5
<code>ADC_CH6</code>	Channel 6
<code>ADC_CH7</code>	Channel 7
<code>ADC_CH8</code>	Channel 8
<code>ADC_CH9</code>	Channel 9
<code>ADC_CH10</code>	Channel 10
<code>ADC_CH11</code>	Channel 11
<code>ADC_CH12</code>	Channel 12
<code>ADC_CH13</code>	Channel 13
<code>ADC_CH14</code>	Channel 14
<code>ADC_CH15</code>	Channel 15
<code>ADC_CH_CTMU</code>	Channel CTMU
<code>ADC_CH_VDDCORE</code>	Channel VDDCore
<code>ADC_CH_VBG</code>	Channel VBG

**Remarks:** Selects the pin that will be used as input to the A/D converter. And starts an A/D conversion. The **BusyADC**( ) macro or A/D interrupt may be used to detect completion of the conversion.

**Example:** `SelChanConvADC(ADC_CH0)`

# Proton+ PIC18 ADC Macros

## Macro definitions

Macros	Description
ADC_INT_ENABLE( )	Enables A/D interrupt i.e. sets PEIE and ADIE bits.
ADC_INT_DISABLE( )	Disables the A/D interrupt.
ADC_SEVT_ENABLE( )	Enables the Special Event Trigger bit, this macro is available for <b>ADC_V4</b> only.
ADC_SEVT_DISABLE( )	Disables the Special Event Trigger bit, this macro is available for <b>ADC_V4</b> only.
ADC_CALIB( )	Enables the A/D calibration bit, this macro is available for <b>ADC_V6</b> , <b>ADC_V9</b> , <b>ADC_V11</b> and <b>ADC_V12</b> only.
ADC_NO_CALIB( )	Disables the A/D calibration bit, this macro is available for <b>ADC_V6</b> , <b>ADC_V9</b> , <b>ADC_V11</b> and <b>ADC_V12</b> only.
ADC_CH_GRA_AN0( )	Selects the adc channel, this macro is available for <b>ADC_V7</b> and <b>ADC_V7_1</b> only.
ADC_CH_GRA_AN4( )	Selects the adc channel, this macro is available for <b>ADC_V7</b> and <b>ADC_V7_1</b> only.
ADC_CH_GRB_AN1( )	Selects the adc channel, this macro is available for <b>ADC_V7</b> and <b>ADC_V7_1</b> only.
ADC_CH_GRC_AN2( )	Selects the adc channel, this macro is available for <b>ADC_V7</b> and <b>ADC_V7_1</b> only.
ADC_CH_GRD_AN3( )	Selects the adc channel, this macro is available for <b>ADC_V7</b> and <b>ADC_V7_1</b> only.
ADC_CH_GRB_AN5( )	Selects the adc channel, this macro is available for <b>ADC_V7</b> only.
ADC_CH_GRC_AN6( )	Selects the adc channel, this macro is available for <b>ADC_V7</b> only.
ADC_CH_GRD_AN7( )	Selects the adc channel, this macro is available for <b>ADC_V7</b> only.
ADC_CH_GRA_AN8( )	Selects the adc channel, this macro is available for <b>ADC_V7</b> only.
ALL_CH_DIGITAL( )	Selects the adc channel, this macro is available for <b>ADC_V7</b> only.

## Example use of the A/D Converter Macros:

```

Device = 18F452
Xtal = 4
Optimiser_Level = 3
Dead_Code_Remove = On
Include "adc.inc"           'Load the macros into the program

Dim Result as Word

' Configure A/D Convertor
OpenADC(ADC_FOSC_32 & ADC_RIGHT_JUST & ADC_12_TAD, _
        ADC_CH0 & ADC_REF_VDD_VSS & ADC_INT_OFF, _
        ADC_CH10)
DelayUs 2                    ' Delay for 2 MicroSeconds
ConvertADC()                 ' Start conversion
While BusyADC() = 1 : Wend   ' Wait for completion
Result = ReadADC()           ' Read result
CloseADC()                   ' Disable A/D converter

```