MICROPROCESSADORES II

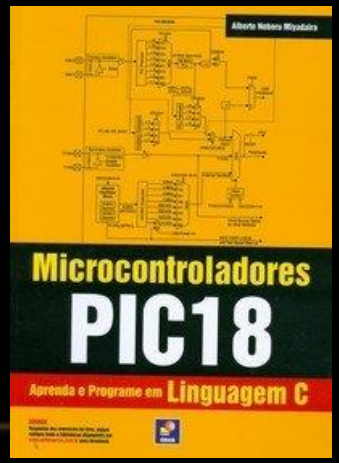
Professor: Patric Janner Marques

Aula : Conversor analógico digital e Comparadores analógicos

Referência bibliográfica

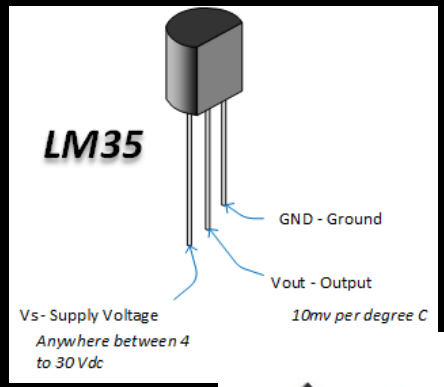
- Para aula de hoje:
  - Salvador: seção 1.9.4 (bem básico).
  - Miyadaira: Cap. 13; Cap. 14.

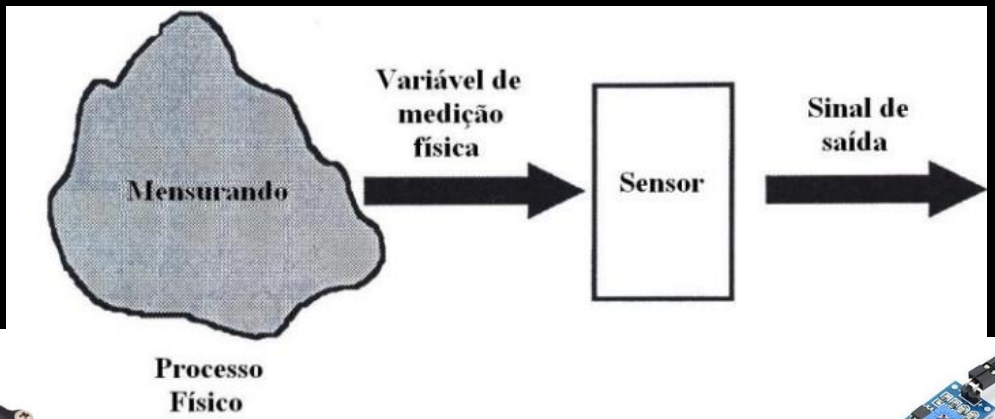
- **Cuidado:** Miyadaira utiliza compilador C18.
- Lembre-se que a plib não funciona mais!!

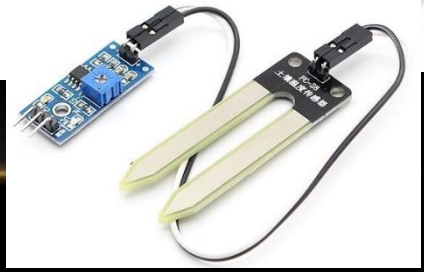


Conversor A/D

- De uma forma geral, as grandezas físicas que se pode observar raramente são de natureza elétrica. O primeiro passo para trazer as informações que elas possuem para os sistemas microprocessados é o de transformar essas grandezas em sinais elétricos, e para isto utiliza-se sensores e transdutores.

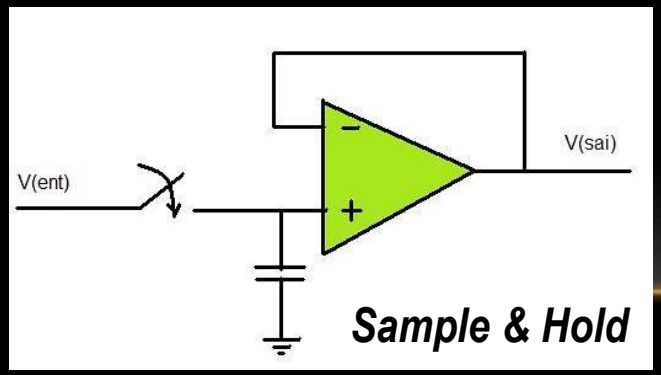


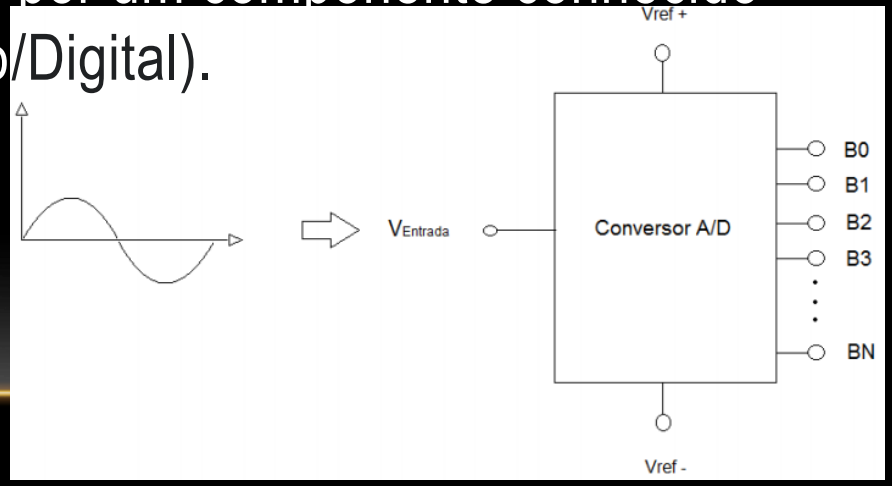




Conversor A/D

- Mesmo obtendo as informações das grandezas como sinais elétricos, sua natureza ainda é analógica e contínua no tempo;
- Para essas grandezas serem processadas pelo microcontrolador é necessário realizar mais uma transformação do sinal analógico para um sinal digital;
- Essa transformação é realizada por um componente conhecido como Conversor A/D (Analógico/Digital).



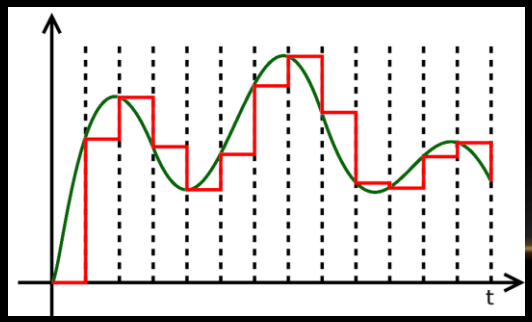


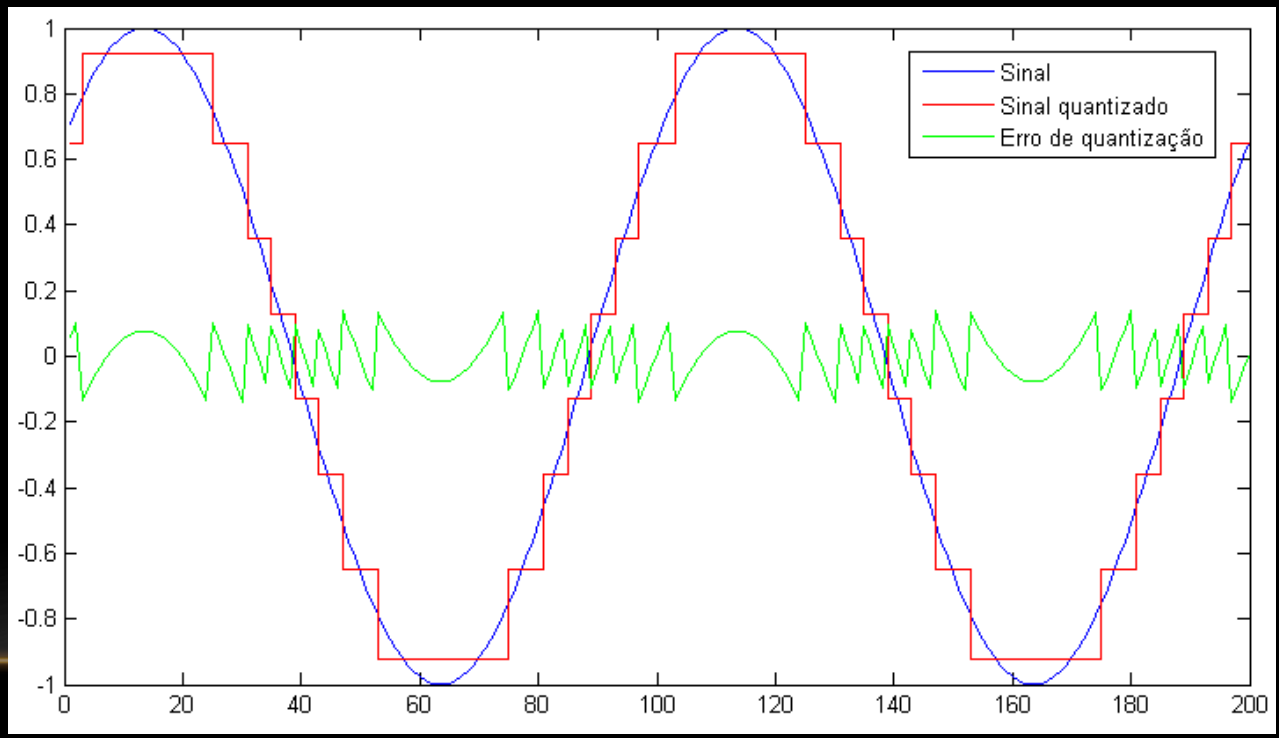
Conversor A/D

- Sendo assim, um conversor A/D transforma um sinal analógico, contínuo no tempo, em um sinal amostrado (ou digital), ou seja, discreto no tempo e quantizado dentro de um número finito de valores inteiros.

Observando:

- Amplitude
- Tempo





Conversor A/D – Na Amplitude

- A resolução do conversor se dá pelo seu número de bits (n), podendo ser 8, 10, 12, 16 entre outros, e a faixa de tensão elétrica a ser convertida, respeitando a equação:

$$Resolução = \frac{Faixa}{2^n - 1} = \frac{5 - 0\ V}{2^{10} - 1} = \frac{5}{1023} = 4{,}8875 mV/bit$$

Exemplo:

A/D de 8 bits
+ Vref = Vdd = 5V
- Vref = Vss = 0V
Resolução de 19,61 mV/bit
1,3 V = ?

5 V → 255
1,3V → X

X = 51

A/D de 10 bits
+ Vref = Vdd = 5V
- Vref = Vss = 0V
Resolução de 4,88 mV/bit
1,3 V = ?

5 V → 1023
1,3V → X

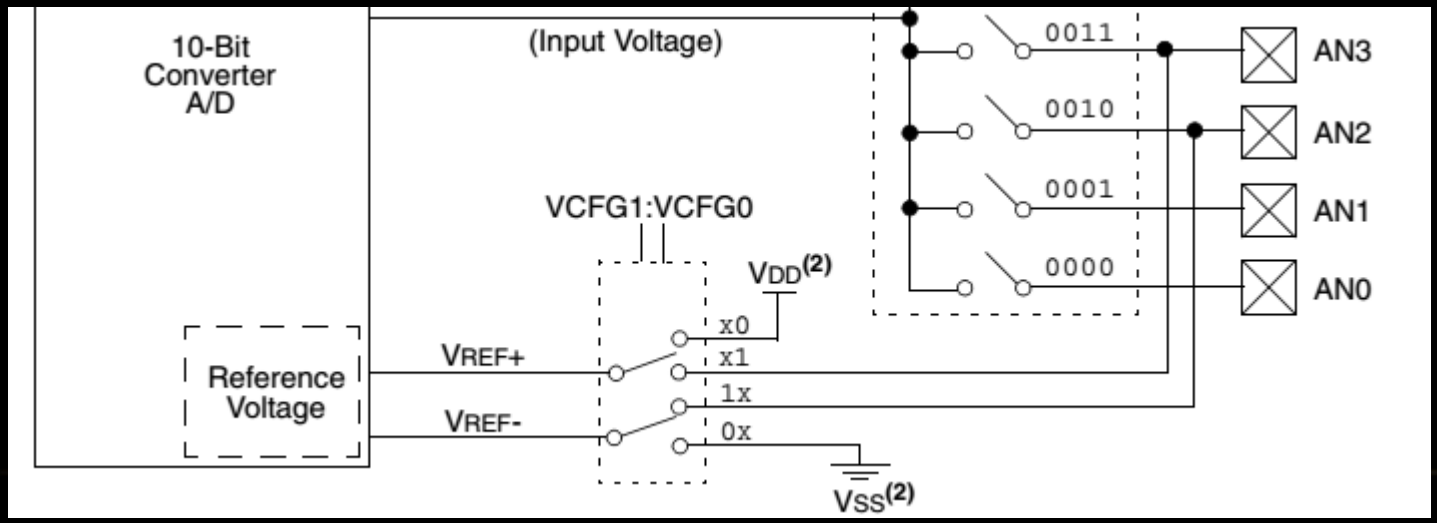
X = 265,9 ≃ 266

Conversor A/D no PIC18F4550

- No PIC18F4550 tem-se 13 ADCs multiplexados (AN0 a AN12);
- Detalhes na página 265 do *datasheet*;
- A resolução deles é de 10 bits;
- A tensão de referência (faixa de tensão) é configurável (próximo slide).
- Importante:
  - Se as entradas AN8 a AN12 forem utilizadas como ADC, a diretiva #pragma config PBADEN = ON deve ser usada;
  - Os pinos utilizados como ADC devem ser configurados como entrada pelo registrador TRIS.

Conversor A/D no PIC18F4550

- V_{ref-} = Pino 4 VREF- e V_{ref+} = Pino 5 VREF+
- V_{ref-} = Pino 4 VREF- e V_{ref+} = VCC
- V_{ref-} = VSS e V_{ref+} = Pino 5 VREF+
- V_{ref-} = VSS e V_{ref+} = VCC



Conversor A/D no PIC18F4550

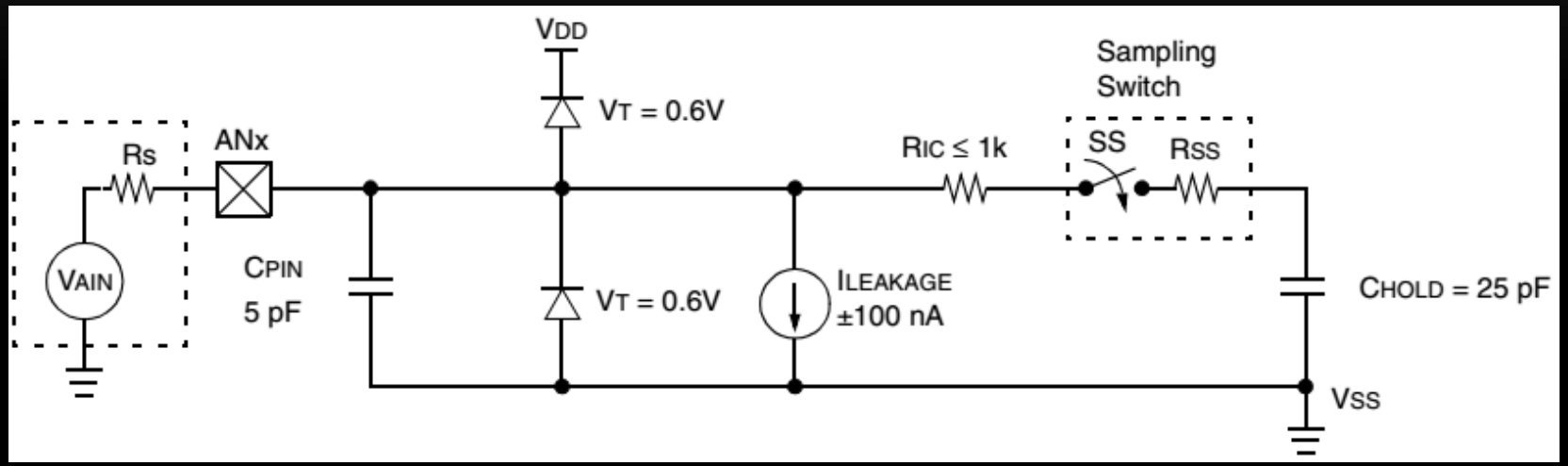
TABLE 28-28: A/D CONVERTER CHARACTERISTICS: PIC18F2455/2550/4455/4550 (INDUSTRIAL)
PIC18LF2455/2550/4455/4550 (INDUSTRIAL)

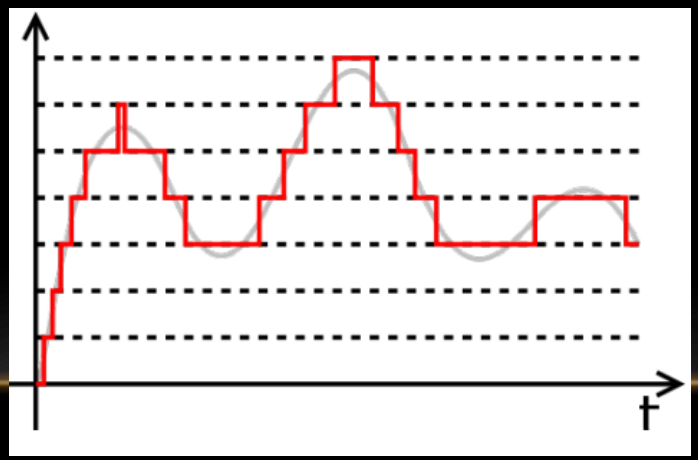
Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
A01	NR	Resolution	—	—	10	bit	ΔVREF ≥ 3.0V
A03	EIL	Integral Linearity Error	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A04	EDL	Differential Linearity Error	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A06	EOFF	Offset Error	—	—	<±1.5	LSb	ΔVREF ≥ 3.0V
A07	EGN	Gain Error	—	—	<±1	LSb	ΔVREF ≥ 3.0V
A10	—	Monotonicity	Guaranteed(1)			—	VSS ≤ VAIN ≤ VREF
A20	ΔVREF	Reference Voltage Range (VREFH – VREFL) Reference Voltage Range (VREFH – VREFL)	1.8 3	— —	— —	V V	VDD < 3.0V VDD ≥ 3.0V
A21	VREFH	Reference Voltage High	VSS	—	VREFH	V	
A22	VREFL	Reference Voltage Low	VSS – 0.3V	—	VDD – 3.0V	V	
A25	VAIN	Analog Input Voltage	VREFL	—	VREFH	V	
A30	ZAIN	Recommended Impedance of Analog Voltage Source	—	—	2.5	kΩ	
A50	IREF	VREF Input Current(2) VREF Input Current(2)	— —	— —	5 150	μA μA	During VAIN acquisition. During A/D conversion cycle.

Conversor A/D – No tempo

- A amostragem feita no microcontrolador é realizada pelo processo *Sample and Hold* (amostragem e retenção);
- O conversor A/D possui um capacitor interno, chamado de C_{Hold} (25pF), que é ligado ao canal analógico selecionado durante a amostragem do sinal;
- Assim, ele é carregado com a tensão presente na entrada e quando um processo de conversão tem início, o capacitor é desligado do canal selecionado, mantendo assim a tensão anteriormente presente na entrada;
- Desta forma, mesmo que a tensão na entrada sofra pequenas variações, estas não afetarão a conversão que agora está em andamento internamente;
- O processo de conversão da tensão armazenada no capacitor é feito pelo método de aproximações sucessivas.

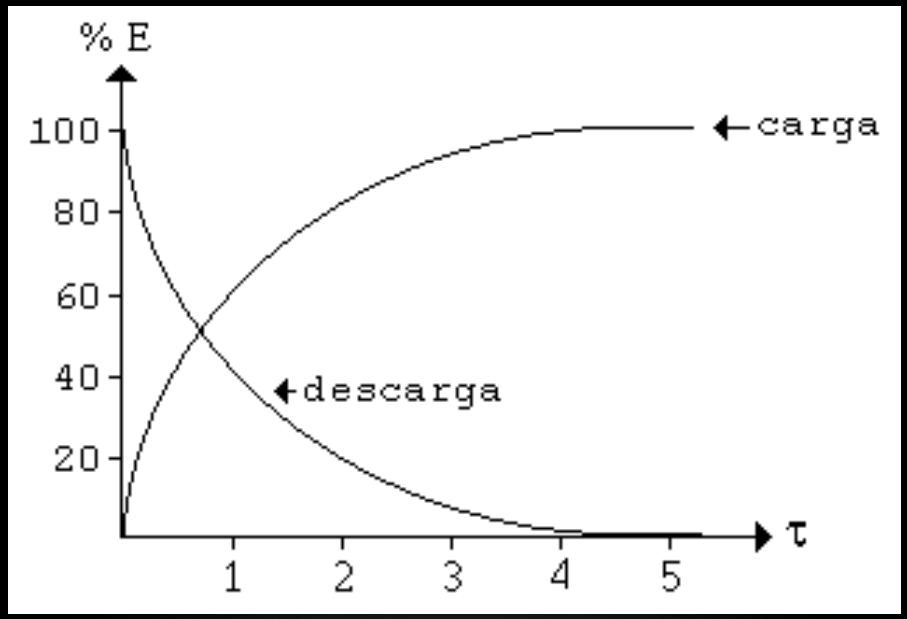
Conversor A/D – No tempo





Conversor A/D – No tempo

Mas o que o tempo de conversão tem haver com isso?!?!?



Isso lembra algo???

Conversor A/D – No tempo

- Após o tempo de carga do capacitor (THOLD) a entrada é desconectada e inicia-se o processo de conversão da tensão armazenada no capacitor;
- Logo, o tempo total de amostragem, que vai do instante em que o canal a ser amostrado é selecionado ao momento em que o resultado da conversão é armazenado nos registros de resultado do AD, é dado pela soma do tempo de aquisição com o tempo de conversão.
- O tempo de aquisição varia em função da temperatura, da tensão de alimentação e da resistência da fonte do sinal a ser amostrado, que pode ser equacionado da seguinte forma:

Conversor A/D – No tempo

$$T_{ACQ} = \text{Amplifier Settling Time} + \text{Holding Capacitor Charging Time} + \text{Temperature Coefficient}$$
$$= T_{AMP} + T_C + T_{COFF}$$

Tempo de estabilização do amplificador

$$T_{AMP} = 0.2\ \mu s$$

Tempo de carregamento do capacitor

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ \ln(1/2048)$$

Coeficiente de temperatura

$$T_{COFF} = (Temp - 25°C)(0.02\ \mu s/°C)$$

O coeficiente de temperatura somente é necessário para temperatura acima de 25ºC. Para valores abaixo TCOFF = 0 ms.

Conversor A/D – No tempo

- Para RS 2.5kOhm (o máximo recomendado) e temperatura de 85º C, tem-se:

$$T_{AMP} = 0.2\ \mu s$$

$$T_{COFF} = (Temp - 25°C)(0.02\ \mu s/°C)$$
$$(85°C - 25°C)(0.02\ \mu s/°C)$$
$$1.2\ \mu s$$

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ \ln(1/2048)\ \mu s$$
$$-(25\ pF)\ (1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega)\ \ln(0.0004883)\ \mu s$$
$$1.05\ \mu s$$

$$T_{ACQ} = 0.2\ \mu s + 1.05\ \mu s + 1.2\ \mu s$$
$$2.45\ \mu s$$

Conversor A/D no PIC18F4550

- Observação importante:

A resistência da fonte da tensão a ser convertida (Rs no circuito do slide 10) não deve ser superior a 2.5kΩ, caso contrário o capacitor CHOLD pode não ser completamente carregado quando se iniciar a conversão, resultando em uma medida incorreta.

Conversor A/D no PIC18F4550

- O T_{AD} é o tempo necessário para a conversão de 1 bit;
- E para a conversão de 10 bit do ADC é necessário aguardar 11 T_{AD} para a correta conversão;
- Sendo a fonte de *clock* do sistema do A/D dada pelo *firmware* desenvolvido, tomando como base o *clock* da CPU.

Deve-se respeitar os limites no PIC18F4550.....

Conversor A/D no PIC18F4550

TABLE 28-29: A/D CONVERSION REQUIREMENTS

Param No.	Symbol	Characteristic		Min	Max	Units	Conditions
130	TAD	A/D Clock Period	PIC18**F**XXXX	0.7	25.0(1)	µs	TOSC based, VREF ≥ 3.0V
			PIC18**LF**XXXX	1.4	25.0(1)	µs	VDD = 2.0V, TOSC based, VREF full range
			PIC18**F**XXXX	TBD	1	µs	A/D RC mode
			PIC18**LF**XXXX	TBD	3	µs	VDD = 2.0V, A/D RC mode
131	TCNV	Conversion Time (not including acquisition time)(2)		11	12	TAD	
132	TACQ	Acquisition Time(3)		1.4 TBD	— —	µs µs	-40°C to +85°C 0°C ≤ to ≤ +85°C
135	TSWC	Switching Time from Convert → Sample		—	**(Note 4)**		
137	TDIS	Discharge Time		0.2	—	µs	

Conversor A/D no PIC18F4550

Lembrando que 2 TOSC = FOSC/2

Modo automático

TABLE 21-1: TAD vs. DEVICE OPERATING FREQUENCIES

AD Clock Source (TAD)		Maximum Device Frequency	
Operation	ADCS2:ADCS0	PIC18FXXXX	PIC18LFXXXX(4)
2 TOSC	000	2.86 MHz	1.43 MHz
4 TOSC	100	5.71 MHz	2.86 MHz
8 TOSC	001	11.43 MHz	5.72 MHz
16 TOSC	101	22.86 MHz	11.43 MHz
32 TOSC	010	45.71 MHz	22.86 MHz
64 TOSC	110	48.0 MHz	45.71 MHz
RC(3)	x11	1.00 MHz(1)	1.00 MHz(2)

Exemplo: Fosc= 20MHz, deve-se usar Fosc/16

$$TDA = \frac{1}{\frac{Fosc}{16}} = \frac{16}{20MHz} = 0.8\mu s$$

Conversor A/D no PIC18F4550

FIGURE 21-4: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 000, TACQ = 0)

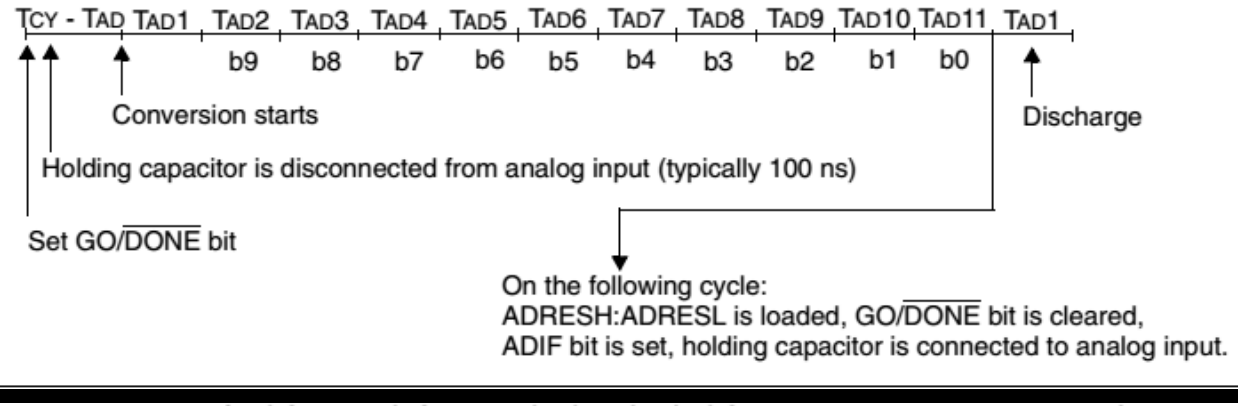
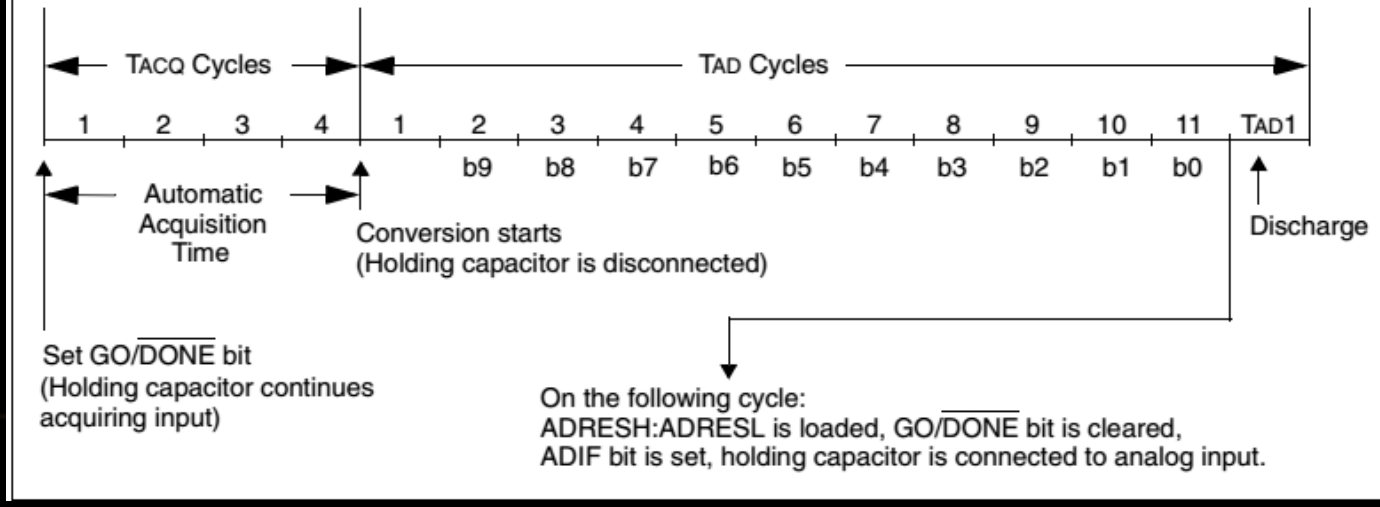


FIGURE 21-5: A/D CONVERSION TAD CYCLES (ACQT<2:0> = 010, TACQ = 4 TAD)



Conversor A/D - Programação

E a programação??????????

Temos que construir a nossa biblioteca para o ADC e adicionar no .h: #define ADC_V5

- Com base em “plib/adc.h”:
  - OpenADC(configurações) - Habilita e configura o módulo conversor;
    - Configurações: são as configurações do conversor separadas por um &.
  - SetChanADC(configuração) – Seleciona o canal analógico que será usado para efetuar a conversão.
    - Configuração: Nome do canal que será selecionado, por exemplo, ADC_CH0.
  - ConvertADC( ) – Inicia o processo de conversão do sinal analógico.

Conversor A/D - Programação

- BusyADC( ) - Verifica se o módulo conversor está em processo de conversão ou não. Se estiver ocupado (em processo de conversão) retorna 1, ou retorna 0 se estiver disponível para realizar uma nova conversão;
- ReadADC( ) – Retorna o valor que foi convertido pelo conversor;
- CloseADC( ) - Desabilita o conversor.

Configurações para OpenADC(configurações)

- Clock para conversão (Define T_{AD}):

ADC_FOSC_2 //A/D base de tempo de conversão é Fosc/2

ADC_FOSC_4 //A/D base de tempo de conversão é Fosc/4

ADC_FOSC_8 //A/D base de tempo de conversão é Fosc/8

ADC_FOSC_16 //A/D base de tempo de conversão é Fosc/16

ADC_FOSC_32 //A/D base de tempo de conversão é Fosc/32

ADC_FOSC_64 //A/D base de tempo de conversão é Fosc/64

ADC_FOSC_RC//A/D base de tempo de conversão é Internal RC OSC

Configurações para OpenADC(configurações)

- Formato do resultado:

ADC_RIGHT_JUST //Justificado a direita

ADC_LEFT_JUST //Justificado a esquerda

Quando se utiliza 10 bits:

ADC_RIGHT_JUST

Quando se utiliza menos que 10 bits, exemplo, somente 8 bits:

ADC_LEFT_JUST

Mais informações: <http://picguides.com/beginner/adc.php>

Configurações para OpenADC(configurações)

- Tempo de aquisição automático (Define T_{ACQ}):

ADC_0_TAD //A/D tempo de aquisição é 0 TAD

ADC_2_TAD //A/D tempo de aquisição é 2 TAD

ADC_4_TAD //A/D tempo de aquisição é 4 TAD

ADC_6_TAD //A/D tempo de aquisição é 6 TAD

ADC_8_TAD //A/D tempo de aquisição é 8 TAD

ADC_12_TAD //A/D tempo de aquisição é 12 TAD

ADC_16_TAD //A/D tempo de aquisição é 16 TAD

ADC_20_TAD //A/D tempo de aquisição é 20 TAD

Configurações para OpenADC(configurações)

- Tempo Interrupção:

ADC_INT_ON //Habilita a interrupção

ADC_INT_OFF //Desabilita a interrupção

- Configuração do V_{REF}:

ADC_REF_VREFPLUS_VREFMINUS //Vref- = Pino 4 VREF- e Vref+ = Pino 5 VREF+

ADC_REF_VDD_VREFMINUS //Vref- = Pino 4 VREF- e Vref+ = VCC

ADC_REF_VREFPLUS_VSS //Vref- = VSS e Vref+ = Pino 5 VREF+

ADC_REF_VDD_VSS //Vref- = VSS e Vref+ = VCC

Configurações para OpenADC(configurações)

- Controle da porta do conversor:

ADC_0ANA //Todos pinos digitais

ADC_1ANA //AN0 analógico, o resto (AN1-AN15) digital

ADC_2ANA // AN0-AN1 analógicos, o resto (AN2-AN15) digital

ADC_3ANA // AN0-AN2 analógicos, o resto (AN3-AN15) digital

Assim sucessivamente até:

ADC_15ANA // Todos pinos analógicos

Conversor A/D – Programação – Exemplo

```
OpenADC(ADC_FOSC_64 &   // ADC_FOSC_64:      Clock de conversão do A/D igual a
                        // FAD = FOSC/64 = 48MHz/64 = 750kHz
                        // Desta forma, TAD=1/FAD = 1,33us.
    ADC_RIGHT_JUST &          // ADC_RIGHT_JUST:    Resultado da conversão ocupará os
                         // bits menos significativos dos registradores ADRESH e ADRESL
    ADC_2_TAD,              // ADC_2_TAD: Determina o tempo de aquisição
                        // Neste caso será igual a 2*TAD = 2*1,33us = 2,6us.
    ADC_CH0 &                    // ADC_CH0:  selecionar o canal no qual será realizada a
                                 // conversão inicial, neste caso o AN0.
    ADC_INT_OFF &       // ADC_INT_OFF: Desabilita a interrupção de término de conversão.
    ADC_REF_VDD_VSS ,    // ADC_VREFPLUS_VDD:    Determina o VDD (+5V) como tensão de
                                 //referência positiva (VREF+) e o VSS (0V)como tensão de
                                 //referência negativa (VREF-).
    ADC_5ANA);        // Configura os pinos AN0 a AN4 como Entradas Analógicas, o resto digital.
```

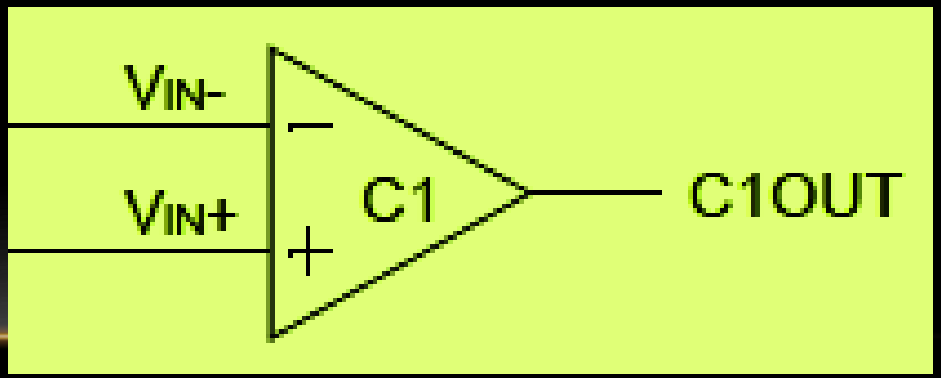
Conversor A/D – Programação

```
void main(void) {
  unsigned int valor;
  TRISB= 0;   // Setando toda a porta B como saída
  TRISA = 0b00000001; // Setando RA0 como entrada p/ ADC
  LATB=255;    // Inicializando os leds apagados
  OpenADC (ADC_FOSC_16 & //FAD = FOSC/16 = 20MHz/16 = 125kHz. Desta forma, TAD=1/FAD = 0,8us.
    ADC_RIGHT_JUST &
    ADC_4_TAD,                  //Tempo de conversão de uma palavra de 10-bits, neste caso será igual a 4*TAD = 4*0,8us = 3,2us.
    ADC_CH0 &
    ADC_INT_OFF &
    ADC_REF_VDD_VSS, //Determina o VDD (+5V) como tensão de referência positiva (VREF+) e o VSS (0V)como tensão de
                  //referência negativa (VREF-).
    ADC_1ANA);      // Será um canal no momento

  SetChanADC (ADC_CH0);   // Seleciona o canal para a conversão
  while(1)    {
    ConvertADC();
    while(BusyADC());
    valor = ReadADC();
    LATB = valor;
    } }
```

Comparadores Analógicos

- Quando se trabalha com microcontroladores, existe situações onde é necessária uma comparação rápida entre dois ou mais valores analógicos para que seja tomada alguma ação sem uma intervenção do programa principal;
- Isso poderia ser vantajoso se fosse feito de forma automática pelo microcontrolador ao invés de se ter módulos A/D para fazer este tipo de tarefa.

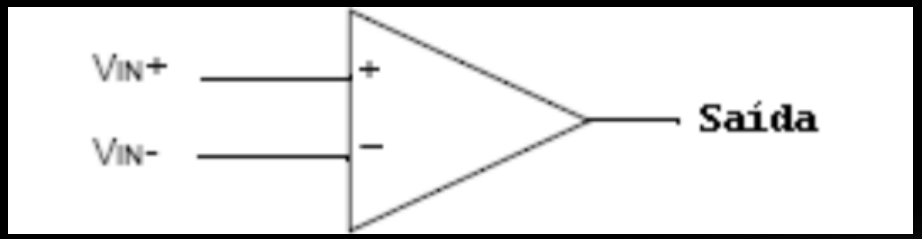


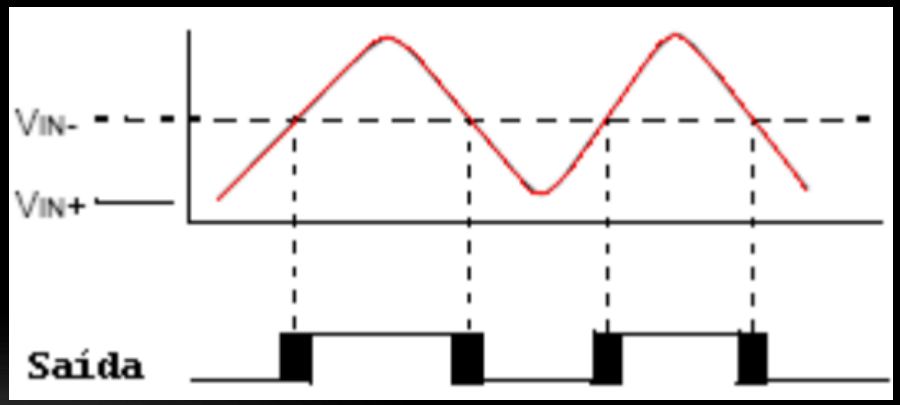
Comparadores Analógicos

- No microcontrolador PIC18F4550 tem-se um periférico que possui esta característica. **Página 273 do *Datasheet***;
- O modulo de comparação analógica contém dois comparadores que podem ser configurados de várias formas;
- As entradas podem ser pinos multiplexados de entrada da **Porta A** (RA0 até RA5), bem como podem ser referências de tensões obtidas dentro do microcontrolador;
- As saídas digitais podem ser obtidas com valores normais ou inversos, estando disponíveis na saída do módulo comparador ou ainda podem ser lidas através do registro de controle.

Comparadores Analógicos

- Quando a entrada analógica Vin+ é maior que a entrada analógica Vin-, a saída do comparador terá um valor de saída de nível lógico alto;
- Quando a entrada analógica em Vin- é maior que a entrada analógica Vin+, a saída do comparador possui sua saída com um nível lógico baixo.





As áreas em negrito da saída do comparador representam a incerteza do valor de saída devido aos *offsets* e o tempo de resposta das entradas.

Comparadores Analógicos

- O registrador para configurar os comparadores:

Registro CMCON:

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

Comparadores Analógicos

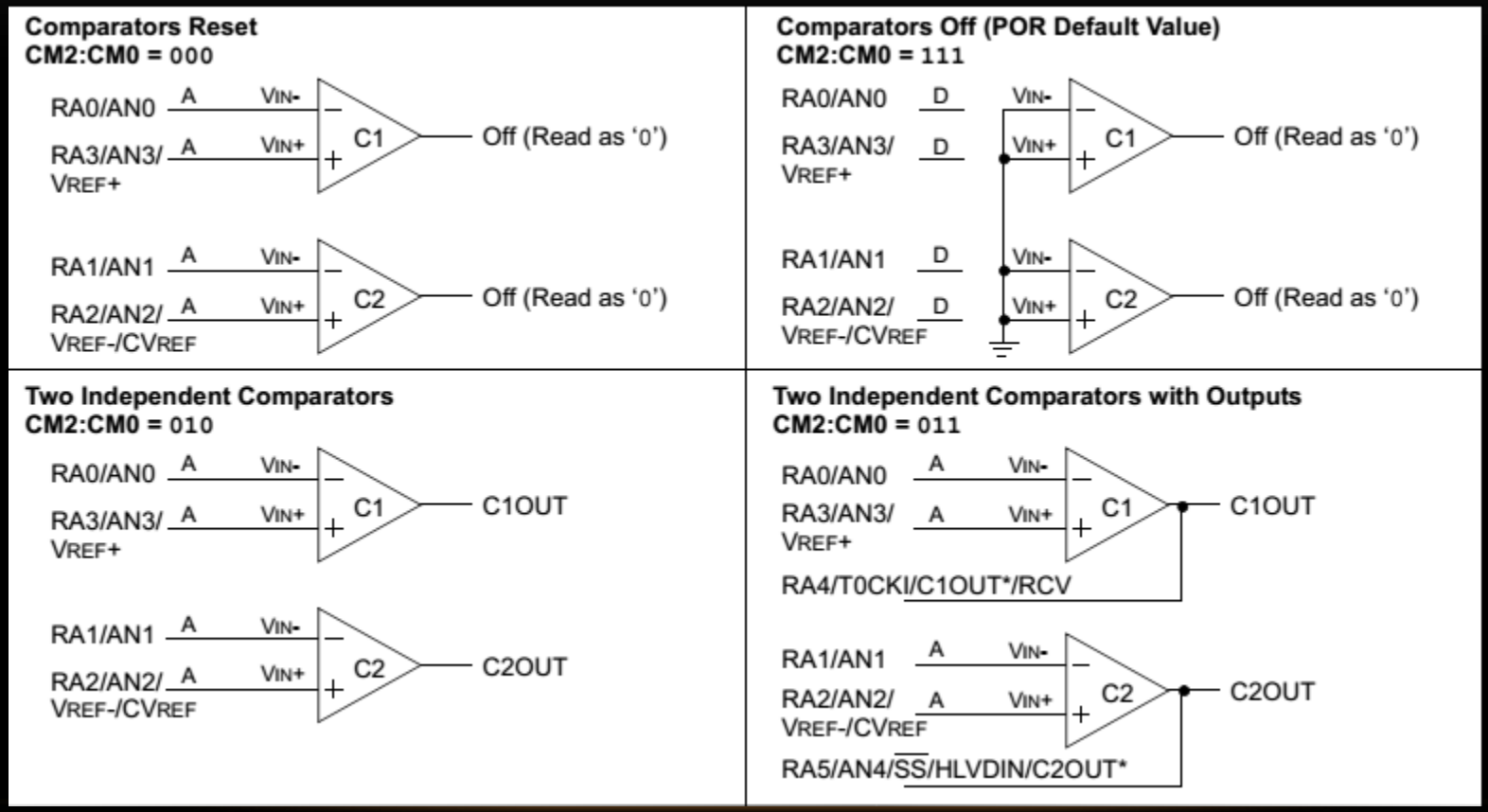
- C2OUT: Armazena o valor de saída do comparador 2, onde:
  - Quando C2INV for **zero**, temos o valor **um** quando o valor de entrada Vin+ do comparador 2 for maior que Vin- e **zero** quando o valor de entrada de Vin+ for menor que Vin-;
  - Quando C2INV for **um** temos exatamente a situação contrária da anterior.
- C1OUT: Armazena o valor de saída do comparador 1, onde:
  - Quando C1INV for **zero**, temos o valor **um** quando o valor de entrada Vin+ do comparador 1 for maior que Vin- e **zero** quando o valor de entrada de Vin+ for menor que Vin-;
  - Quando C1INV for **um** temos exatamente a situação contrária da anterior.

Comparadores Analógicos

- C2INV e C1INV: Como já pôde ser notado, quando possuem valor **um** inverte o valor de saída do comparador C2 e C1 respectivamente e quando em **zero** não inverte;

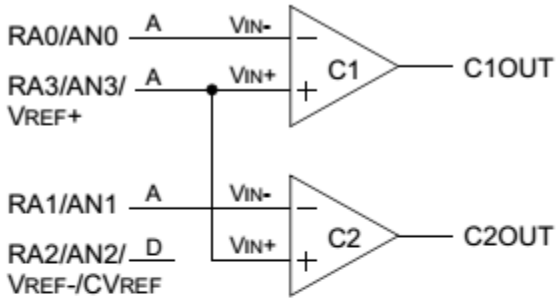
- CM2:CM1:CM0: Estes três registros são responsáveis pela seleção do modo de funcionamento dos dois comparadores, onde sua aplicação pode ser vista abaixo:

Comparadores Analógicos

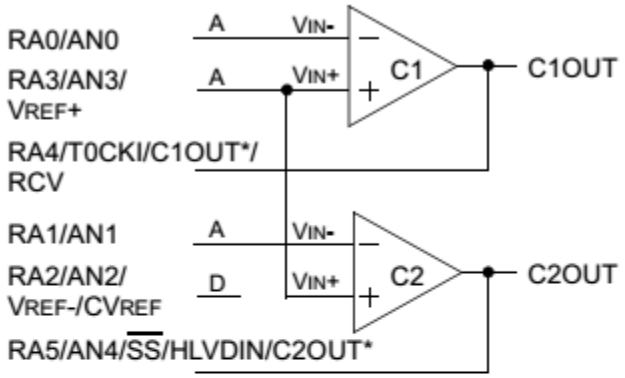


Comparadores Analógicos

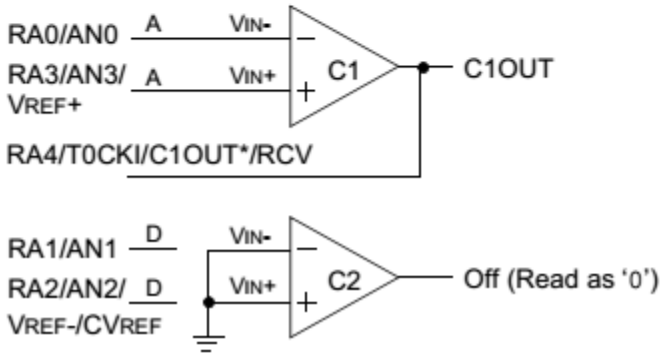
Two Common Reference Comparators
CM2:CM0 = 100



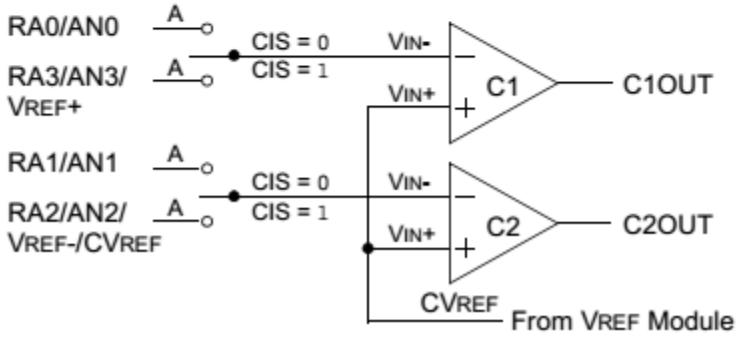
Two Common Reference Comparators with Outputs
CM2:CM0 = 101



One Independent Comparator with Output
CM2:CM0 = 001



Four Inputs Multiplexed to Two Comparators
CM2:CM0 = 110



Comparadores Analógicos

- Por fim, o registrador CIS serve para selecionar a que pinos estarão conectados os Vin- dos comparadores C1 e C2. Assim, quando usa-se CM2:CM1:CM0 = 110 tem-se:

  - Quando CIS for igual a **um**, C1 Vin- estará conectado a RA3/AN3/Vref+ e C2 conectado a RA2/AN2/Vref-/CVref
  - Quando CIS for igual a **zero**, C1 Vin- estará conectado a RA0/AN0 e C2 conectado a RA1/AN1.

Comparadores Analógicos

TABLE 22-1: REGISTERS ASSOCIATED WITH COMPARATOR MODULE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	55
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	55
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	53
PIR2	OSCFIF	CMIF	USBIF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	56
PIE2	OSCFIE	CMIE	USBIE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	56
IPR2	OSCFIP	CMIP	USBIP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	56
PORTA	—	RA6(1)	RA5	RA4	RA3	RA2	RA1	RA0	56
LATA	—	LATA6(1)	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	56
TRISA	—	TRISA6(1)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	56

Comparadores Analógicos – Programação

- Para facilitar um pouco, o compilar xc8 tem uma biblioteca para ajudar na configuração dos comparadores, **que não funciona mais!!**

“plib/ancomp.h”

Temos que construir a nossa biblioteca para o comparador analógico. Não realizada no exemplo postado.

void Open_ancomp(unsigned char config);

Onde config pode ser:

/// Opções de saída invertida

COMP_1_2_OP_INV // Saída 1 e 2 invertidas

COMP_1_OP_INV // Saída 1 invertida

COMP_2_OP_INV // Saída 2 invertida

COMP_OP_INV_NONE // Nenhuma saída invertida

Comparadores Analógicos – Programação

Onde config pode ser:

/// Opções de modo de operação

COMP_1_2_INDP // Comparadores independentes

COMP_1_2_INDP_OP // Comparadores independentes com saída externa

COMP_1_2_COMN_REF // Comparador com uma referência comum

COMP_1_2_COMN_REF_OP // Comparador com uma referência comum e saída externa

COMP_1_INDP_OP // Um comparador independente com saída externa

COMP_INT_REF_SAME_IP // 4 Entradas multiplexadas para 2 comparadores RA0 e RA1

COMP_INT_REF_MUX_IP // 4 Entradas multiplexadas para 2 comparadores RA3 e RA2

Comparadores Analógicos – Programação

Onde config pode ser:

/// Opções de interrupção

COMP_INT_EN // Comparador gera interrupção

COMP_INT_DIS // Comparador não gera interrupção

Comparadores Analógicos – Programação

Exemplo:

```
void main (void){
   TRISD = 0;
   LATD0 = 1;
   Open_ancomp(COMP_1_2_OP_INV&
            COMP_1_2_INDP&
            COMP_INT_DIS );
   while (1)   {
      if(C1OUT == 1){ // testa o bit do comparador 1
                LATD0 = 1;                    }
      else
                LATD0 = 0;
   }
}
```