PIC18F4550: TEMPORIZADORES

Prof. Matheus Ribeiro

1. INTRODUÇÃO

O PIC18F4550 possui 4 módulos temporizadores, denominados TIMER0, TIMER1, TIMER2 e TIMER3. Neste documento são apresentadas as características básicas de funcionamento, configuração e utilização dos temporizadores do PIC18F4550. O compilador base utilizado nos códigos apresentados é o MPLAB XC8.

2. MÓDULO TIMER0

O módulo Timer0 do PIC18F4550 permite operações de leitura e escrita, podendo operar como temporizador ou contador de 8 ou 16 bits. Possui um divisor de clock (PRESCALER) dedicado e programável e possibilidade de acionamento de interrupção por término de contagem.

O registrador de 8 bits T0CON é responsável pela configuração do módulo Timer0.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0

TMR0ON	1 = habilita o contador 0 = para o contador
T08BIT	*1 = contador de 8 bits (0 a 255)* 0 = contador de 16 bits (0 a 65.535)
T0CS	1 = fonte de *clock* proveniente do pino T0CKI 0 = fonte de *clock* proveniente do oscilador interno do microcontrolador
T0SE	Ignorado se T0CS = 0 1 = incremento na borda de descida do pino T0CKI 0 = incremento na borda de subida do pino T0CKI
PSA	1 = PRESCALER não utilizado 0 = PRESCALER habilitado de acordo com a configuração dos bits T0PS2:T0PS0
T0PS2:T0PS0	Ignorado se PSA = 1 111 = divide o *clock* na razão 1:256 110 = divide o *clock* na razão 1:128 101 = divide o *clock* na razão 1:64 100 = divide o *clock* na razão 1:32 011 = divide o *clock* na razão 1:16 010 = divide o *clock* na razão 1:8 001 = divide o *clock* na razão 1:4 000 = divide o *clock* na razão 1:2

2.1. Utilização do Timer0 no modo temporizador

Cálculo da frequência de clock do temporizador:

- A base de clock do Timer0 é o tempo de um ciclo de instrução (Tcy), ou seja, Fosc/4. Portanto, se utilizamos um cristal de 4MHz como oscilador externo (Fosc=4MHz), a frequência de clock do Timer1 será de 1MHz (Tcy = 1us).
- Desta forma, se trabalhamos com o contador ajustado para 8 bits, temos a temporização máxima de 256*1us = 256us. Para 16 bits, temos a temporização máxima de 65.536*1us = 65,536ms.

Uso do PRESCALER:

- O PRESCALER é um divisor de frequência responsável por reduzir os pulsos de clock gerados para o contador do Timer0 de acordo com a configuração dos bits T0PS2:T0PS0 do registrador T0CON.
- Exemplo: Se temos o clock do Timer0 igual a 1MHz e o PRESCALER foi configurado para a razão de 1:256, teremos o incremento do contador sendo realizado com a frequência de, aproximadamente, 3,9KHz. Neste caso, a temporização máxima para um contador de 16 bits será de 256*65.536*1us ≈ 16,78s.

Sinalização de término da contagem:

- O término de contagem ocorre no overflow do contador, que ocorre na transição 0xFF → 0x00 para o contador de 8 bits ou na transição 0xFFFF → 0x0000 para o contador de 16 bits.
- Para sinalizar o término da contagem, o sinalizador de overflow TMR0IF do registrador INTCON e atualizado para nível lógico alto.
- Para reiniciar a contagem o sinalizador de overflow deve ser limpo: INTCONbits.TMR0IF = 0.

Uso dos registradores TMR0L e TMR0H:

- O valor atual do contador pode ser lido ou redefinido por meio dos registradores TMR0L e TMR0H.
- Se o contador foi configurado para 8 bits, apenas o TMR0L é utilizado.
- Se o contador foi configurado para 16 bits, TMR0L corresponde a parte menos significativa do número e TMR0H corresponde a parte mais significativa. Para fazer a leitura do Timer em 16 bits é necessário fazer primeiro a leitura da parte baixa e depois da parte alta do resultado, caso contrário ocorrerá erro de leitura:

```
VALOR = TMR0L;
VALOR = VALOR + (TMR0H*256);
```

2.2. Utilização do Timer0 no modo contador

- Neste modo, a fonte de clock é externa através do pino RA4/T0CKI. Portanto, este pino deve ser configurado como uma entrada digital (TRISAbits.RA4 = 1).
- Também pode ser configurada a atualização para a borda de subida ou descida (T0CONbits.T0SE). Esta configuração pode ser utilizada para a adequação do contador a dispositivos “Normalmente Aberto” ou “Normalmente Fechado” (NA/NF).
- A atribuição dos valores iniciais da contagem, a utilização do PRESCALER e do sinalizador de término de contagem é feita da mesma forma que no modo temporizador.

2.3. Interrupção do Timer0

- Se a interrupção do Timer0 estiver habilitada (INTCONbits.GIE = 1 e INTCONbits.TMR0IE = 1) o programa é desviado para a Rotina de Tratamento de Interrupções sempre que ocorre a ativação do bit de sinalização de overflow TMR0IF.
- Antes de sair da Rotina de Tratamento de Interrupções o bit TMR0IF deve ser "limpo".
- No modo de economia de energia (Sleep mode) o Timer0 é desativado e, portanto, o microcontrolador não pode ser posto em pleno funcionamento a partir da interrupção do Timer0.

2.4. Utilização do Timer0

Modo Temporizador:

```
// Rotina de Inicialização
T0CON = 0x84;          // Timer0: ON, 16 bits, clock interno, PRESCALER ativo, 1:32
TMR0H = 0x85;          // Carrega valores iniciais do contador (0x85EE = 34.286)
TMR0L = 0xEE;
INTCONbits.TMR0IF = 0; // Limpa flag de overflow do Timer0
```

A partir das configurações iniciais do Timer0, podemos determinar o número de contagens para atingir o overflow como (CONT_MAX+1 – CONT_INICIAL)*PRESCALER, ou seja, (65.536 – 34.286)*32. Verificamos então que o overflow ocorre com 1.000.000 de contagens. Se utilizamos um cristal de 4MHz como oscilador principal temos 1TCY = 1us e, portanto, overflow com 1.000.000 x 1us = 1 segundo.

```
// Exemplo de rotina de utilização
while(!INTCONbits.TMR0IF);         // Aguarda atingir o overflow
LATDbits.LATD0 = !LATDbits.LATD0;  // Inverte o estado da saída
TMR0H = 0x85;                     // Recarrega os valores iniciais do contador
TMR0L = 0xEE;
INTCONbits.TMR0IF = 0;          // Limpa o sinalizador de overflow
```

Modo Contador:

```
// Rotina de Inicialização
T0CON = 0xF8;             // Timer0: ON, 8 bits, clock externo, descida, PRESCALER inativo
INTCONbits.TMR0IF = 0;    // Limpa flag de overflow do Timer0
TMR0L = 0xFC;             // Carrega valores iniciais do contador (0xFC = 252)
```

A partir das configurações iniciais do Timer0, podemos determinar o número de contagens para atingir o overflow como (CONT_MAX+1 – CONT_INICIAL)*PRESCALER, ou seja, (256 – 252)*1. Verificamos então que o overflow ocorre com 4 contagens.

```
// Exemplo de rotina de utilização
if(INTCONbits.TMR0IF)  // Teste se foi atingido o overflow
{
  LATDbits.LATD0 = !LATDbits.LATD0;  // Inverte o estado da saída
  INTCONbits.TMR0IF = 0;             // Limpa o sinalizador de overflow
  TMR0L = 0xFC;                      // Recarrega os valores iniciais do contador
}
```

3. MÓDULO TIMER1

O módulo Timer1 do PIC18F4550 permite operação como temporizador ou contador de 16 bits. Possui um divisor de clock (PRESCALER) dedicado e programável e possibilidade de acionamento de interrupção por término de contagem.

O registrador de 8 bits T1CON é responsável pela configuração do módulo Timer1.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON

RD16	1 = leitura ou escrita do Timer1 em uma operação de 16 bits 0 = leitura ou escrita do Timer1 em duas operações de 8 bits
T1RUN	1 = indica que o oscilador interno do Timer1 está sendo usado como fonte de *clock* 0 = indica que outra fonte de *clock* está sendo usada
T1CKPS1:T1CKPS0	Configura o divisor de clock (PRESCALER) do Timer1 11 = divide o *clock* na razão 1:8 10 = divide o *clock* na razão 1:4 01 = divide o *clock* na razão 1:2 00 = divide o *clock* na razão 1:1
T1OSCEN	Apenas para o modo de economia de energia 1 = habilita o oscilador interno do Timer1 0 = desliga o oscilador interno do Timer1
T1SYNC	Apenas para o modo contador 1 = não sincroniza a entrada de clock externo (modo assíncrono) 0 = sincroniza a entrada de clock externo (modo síncrono)
TMR1CS	1 = modo contador ou de economia de energia (clock proveniente do pino T13CKI ou do oscilador interno do Timer1) 0 = modo temporizador (clock proveniente do oscilador interno do microcontrolador)
TMR1ON	1 = habilita o contador 0 = para o contador

3.1. Utilização do Timer1 no modo temporizador

Cálculo da frequência de clock do temporizador:

- A base de clock do Timer1 é o tempo de um ciclo de instrução (Tcy), ou seja, Fosc/4. Portanto, se utilizamos um cristal de 4MHz como oscilador externo (Fosc=4MHz), a frequência de clock do Timer1 será de 1MHz (Tcy = 1us).
- O Timer1 opera somente em 16 bits e, portanto, temos a temporização máxima de 65.536*1us = 65,536ms.

Uso do PRESCALER:

- O PRESCALER é um divisor de frequência responsável por reduzir os pulsos de clock gerados para o contador do Timer1 de acordo com a configuração dos bits T1PS2:T1PS0 do registrador T1CON.
- Exemplo: Se temos o clock do Timer1 igual a 1MHz e o PRESCALER foi configurado para a razão de 1:8, teremos o incremento do contador sendo realizado com a frequência de, aproximadamente, 125KHz. Neste caso, a temporização máxima será de 8*65.536*1us = 524,288ms.

Sinalização de término da contagem:

- Para sinalizar o término da contagem, que ocorre na mudança FFFFh → 0000h, o sinalizador de overflow TMR1IF do registrador PIR1 é atualizado para nível lógico alto.
- Para reiniciar a contagem o sinalizador de overflow deve ser limpo: PIR1bits.TMR1IF = 0.

Uso dos registradores TMR1L e TMR1H:

- O valor atual do contador pode ser lido ou redefinido por meio dos registradores TMR1L e TMR1H, onde TMR1L corresponde a parte menos significativa do número e TMR1H corresponde a parte mais significativa.

3.2. Utilização do Timer1 no modo contador

- Neste modo, a fonte de clock é externa através do pino RC0/T1OSO/T13CKI. Portanto, este pino deve ser configurado como uma entrada digital (TRISCbits.RC0 = 1).
- A atribuição dos valores iniciais da contagem, a utilização do PRESCALER e do sinalizador de término de contagem é feita da mesma forma que no modo temporizador.

3.3. Oscilador interno do Timer1

- Neste modo, a fonte de clock é obtida através de um oscilador interno de baixo consumo ajustado para 32KHz, sendo necessário incluir ressonador (cristal + capacitores) entre os pinos RC0/T1OSO/T13CKI e RC1/OSI.
- Para utilizar este modo de operação, o bit T1OSCEN do registrador T1CON deve ser ativado. Como consequências esta ativação, os bits 1 e 0 do registrador TRISC são ignorados e os pinos RC1 e RC0 passam a ser entradas.
- Para uma correta operação, deve ser utilizado um atraso adequado para inicialização do Timer1.

3.4. Interrupção do Timer1

- Se a interrupção do Timer1 estiver habilitada (INTCONbits.GIE = 1 e PIE1bits.TMR1IE = 1) o programa é desviado para a Rotina de Tratamento de Interrupções sempre que ocorre a ativação do bit de sinalização de overflow TMR1IF.
- Antes de sair da Rotina de Tratamento de Interrupções o bit TMR1IF deve ser "limpo".

3.5. Utilização do Timer1

Modo Temporizador:

```
// Rotina de inicialização
T1CON = 0x01;           // Timer1: PRESCALER 1:1, clock Fosc/4
TMR1H = 0x00;
TMR1L = 0x00;           // Carrega valores iniciais do Timer1 (0x0000)
PIR1bits.TMR1IF = 0;    // Limpa flag de overflow do Timer1
```

```
// Rotina de utilização
while(!PIR1bits.TMR1IF);           // Aguarda atingir o overflow
LATDbits.LATD0 = !LATDbits.LATD0;  // Inverte o estado da saída
TMR1H = 0x00;
TMR1L = 0x00;                      // Recarrega os valores iniciais do contador
PIR1bits.TMR1IF = 0;              // Limpa o sinalizador de overflow
```

Modo Contador:

```
// Rotina de inicialização
T1CON = 0x03;           // Timer1: PRESCALER 1:1, clock externo
TMR1H = 0xFF;
TMR1L = 0xFE;           // Carrega valores iniciais do Timer1 (0xFFFD = 65.534)
PIR1bits.TMR1IF = 0;    // Limpa flag de overflow do Timer1

// Rotina de utilização
while(!PIR1bits.TMR1IF);           // Aguarda atingir o overflow
LATDbits.LATD0 = !LATDbits.LATD0;  // Inverte o estado da saída
TMR1H = 0xFF;
TMR1L = 0xFE;                      // Recarrega os valores iniciais do contador
PIR1bits.TMR1IF = 0;               // Limpa o sinalizador de overflow
```

4. MÓDULO TIMER2

O módulo Timer2 do PIC18F4550 permite operação como temporizador de 8 bits. Possui um divisor de clock (PRESCALER) na entrada e divisor de contagens completas na saída (POSTSCALER) programáveis e possibilidade de acionamento de interrupção por ocorrência de igualdade dos registradores TMR2 e PR2.

O registrador de 8 bits T2CON é responsável pela configuração do módulo Timer2.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0

T2OUTPS3:T2OUTPS0	Configuração do POSTSCALER 0000 = 1:1 0001 = 1:2 0010 = 1:3 0011 = 1:4 0100 = 1:5 0101 = 1:6 0110 = 1:7 0111 = 1:8 1000 = 1:9 1001 = 1:10 1010 = 1:11 1011 = 1:12 1100 = 1:13 1101 = 1:14 1110 = 1:15 1111 = 1:16
TMR2ON	1 = habilita o Timer2 0 = para o Timer2
T2CKPS1:T2CKPS0	Configuração do PRESCALER 00 = sem divisão 01 = divide o *clock* na razão 1:4 1x = divide o *clock* na razão 1:16

4.1. Utilização do Timer2

Cálculo da frequência de clock do temporizador:

- A base de clock do Timer2 é o tempo de um ciclo de instrução (Tcy), ou seja, Fosc/4. Portanto, se utilizamos um cristal de 4MHz como oscilador externo (Fosc=4MHz), a frequência de clock do Timer0 será de 1MHz (Tcy = 1us).
- O Timer2 tem tamanho fixo de 8 bits e, portanto, temos a temporização máxima de 256*1us = 256us.

Uso do PRESCALER:

- O PRESCALER é um divisor de frequência responsável por reduzir os pulsos de clock gerados para o contador do Timer2 de acordo com a configuração dos bits T2CKPS1:T2CKPS0 do registrado T2CON.

- Exemplo: Se temos o clock do Timer2 igual a 1MHz e o PRESCALER foi configurado para a razão de 1:16, teremos o incremento do contador sendo realizado com a frequência de, aproximadamente, 62,5KHz. Neste caso, a temporização máxima do contador será de 16*256*1us = 4,096ms.

Uso do POSTSCALER:

- O POSTSCALER é um circuito contador de 4 bits responsável por aumentar o número de "casamentos" entre os registradores TMR2 e PR2 necessários para a sinalização de término de temporização.
- Exemplo: Se temos o POSTSCALER configurado para 1:10, teremos a sinalização de término de contagem após 10 comparações positivas entre TMR2 e PR2.

Sinalização de término da contagem:

- Para sinalizar o término da contagem, que ocorre no overflow do POSTSCALER, o sinalizador de TMR2IF do registrador PIR1 e atualizado para nível lógico alto.
- Para reiniciar a contagem o sinalizador de overflow deve ser limpo: PIR1bits.TMR2IF = 0.

Uso dos registradores TMR2 e PR2:

- O valor atual do contador pode ser lido ou redefinido por meio do registrador TMR2.
- O valor atual do período o Timer2 pode ser lido ou redefinido por meio do registrador PR2.
- O registrador TMR2 começa a contagem em 0x00 e, se o registrador PR2 for ajustado para 0xFF, teremos o período máximo (contagem de 256).
- Quando o valor de TMR2 atinge o estabelecido para PR2, o registrador TMR2 é reconfigurado para 0x00 automaticamente.

4.2. Interrupção do Timer2

- Se a interrupção do Timer1 estiver habilitada (INTCONbits.GIE = 1 e PIE1bits.TMR2IE = 1) o programa é desviado para a Rotina de Tratamento de Interrupções sempre que ocorre a ativação do bit de sinalização de overflow TMR1IF.
- Antes de sair da Rotina de Tratamento de Interrupções o bit TMR1IF deve ser "limpo".

4.3. Utilização do Timer2

```
// Rotina de inicialização
T2CON = 0xFF;        // Timer2: POSTSCALE: 1:16, ON, PRESCALE 1:16
TMR2 = 0x00;         // Carrega valor inicial do Timer2
PR2 = 0xFF;          // Carrega valores iniciais do período do Timer2 (0xFF = período máximo)
PIR1bits.TMR2IF = 0; // Limpa flag de overflow do Timer2

// Rotina de utilização
while(!PIR1bits.TMR2IF);           // Aguarda overflow do Timer2
TMR2 = 0x00;                       // Redefine valor inicial do Timer2
PIR1bits.TMR2IF = 0;               // Limpa flag de overflow do Timer2
LATDbits.LATD0 = !LATDbits.LATD0;  // Inverte o estado da saída
```

5. MÓDULO TIMER3

O módulo Timer3 do PIC18F4550 permite operação como temporizador ou contador de 16 bits. Possui um divisor de clock (PRESCALER) dedicado e programável e possibilidade de acionamento de interrupção por término de contagem.

O registrador de 8 bits T3CON é responsável pela configuração do módulo Timer3.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON

RD16	1 = leitura ou escrita do Timer3 em uma operação de 16 bits 0 = leitura ou escrita do Timer3 em duas operações de 8 bits
T3CCP2:T3CCP1	1x = Timer3 é a fonte de *clock* para comparação/captura nos módulos CCPx e ECCPx 01 = Timer3 é a fonte de *clock* para a comparação/captura no módulo CCP2 Timer1 é a fonte de *clock* para a comparação/captura no módulo CCP1 00 = Timer1 é a fonte de *clock* para a comparação/captura nos módulos CCPx e ECCPx
T1CKPS1:T1CKPS0	Configura o divisor de clock (PRESCALE) do Timer3 11 = divide o *clock* na razão 1:8 10 = divide o *clock* na razão 1:4 01 = divide o *clock* na razão 1:2 00 = divide o *clock* na razão 1:1
$\overline{\text{T3SYNC}}$	Apenas para o modo contador 1 = não sincroniza a entrada de clock externo (modo assíncrono) 0 = sincroniza a entrada de clock externo (modo síncrono)
TMR3CS	1 = modo contador ou de economia de energia (clock proveniente do pino T13CKI ou do oscilador do Timer1) 0 = modo temporizador (clock proveniente do oscilador interno do microcontrolador)
TMR3ON	1 = habilita o Timer3 0 = para o Timer3

5.1. Utilização do Timer3

Cálculo da frequência de clock do temporizador:

- A base de clock do Timer3 é o tempo de um ciclo de instrução (Tcy), ou seja, Fosc/4. Portanto, se utilizamos um cristal de 4MHz como oscilador externo (Fosc=4MHz), a frequência de clock do Timer3 será de 1MHz (Tcy = 1us).
- O Timer3 opera somente em 16 bits e, portanto, temos a temporização máxima de 65.536*1us = 65,536ms.

Uso do PRESCALER:

- O PRESCALER é um divisor de frequência responsável por reduzir os pulsos de clock gerados para o contador do Timer3 de acordo com a configuração dos bits T3CKPS1:T3CKPS0 do registrado T3CON.
- Exemplo: Se temos o clock do Timer3 igual a 1MHz e o PRESCALER foi configurado para a razão de 1:8, teremos o incremento do contador sendo realizado com a frequência de, aproximadamente, 125KHz. Neste caso, a temporização máxima será de 8*65.536*1us = 524,288ms.

Sinalização de término da contagem:

- Para sinalizar o término da contagem, que ocorre na mudança FFFFh → 0000h, o sinalizador de overflow TMR3IF do registrador PIR2 é atualizado para nível lógico alto.
- Para reiniciar a contagem o sinalizador de overflow deve ser limpo: PIR2bits.TMR3IF = 0.

Uso dos registradores TMR3L e TMR3H:

- O valor atual do contador pode ser lido ou redefinido por meio dos registradores TMR3L e TMR3H, onde TMR3L corresponde a parte menos significativa do número e TMR3H corresponde a parte mais significativa.

5.2. Utilização do Timer3 no modo contador

- Neste modo, a fonte de clock é externa através do pino RC0/T1OSO/T13CKI. Portanto, este pino deve ser configurado como uma entrada digital (TRISACbits.RC0 = 1).
- A atribuição dos valores iniciais da contagem e a utilização do PRESCALER e do sinalizador de término de contagem é feita da mesma forma que no modo temporizador.

5.3. Utilização do oscilador interno do Timer1

- Neste modo, a fonte de clock é obtida através de um oscilador interno de baixo consumo ajustado para 32KHz, sendo necessário incluir ressonador (cristal + capacitores) entre os pinos RC0/T1OSO/T13CKI e RC1/OSI.
- Para utilizar este modo de operação, o bit T1OSCEN do registrador T1CON deve ser ativado. Como consequências esta ativação, os bits 1 e 0 do registrador TRISC são ignorados e os pinos RC1 e RC0 passam a ser entradas.
- Para uma correta operação, deve ser utilizado um atraso adequado para inicialização do Timer1.

5.4. Interrupção do Timer3

- Se a interrupção do Timer3 estiver habilitada (INTCONbits.GIE = 1 e PIE2bits.TMR3IE = 1) o programa é desviado para a Rotina de Tratamento de Interrupções sempre que ocorre a ativação do bit de sinalização de overflow TMR3IF.
- Antes de sair da Rotina de Tratamento de Interrupções o bit TMR3IF deve ser "limpo".

5.5. Utilização do Timer3

Modo Temporizador:

```
// Rotina de Inicialização
T3CON = 0x01;           // Timer1: PRESCALER 1:1, clock Fosc/4
PIR2bits.TMR3IF = 0;    // Limpa flag de overflow do Timer3
TMR3H = 0x00;
TMR3L = 0x00;           // Carrega valores iniciais do Timer3 (0x0000)
```

```
// Exemplo de rotina de utilização
while(!PIR2bits.TMR3IF);           // Aguarda atingir o overflow
LATDbits.LATD0 = !LATDbits.LATD0;  // Inverte o estado da saída
TMR3H = 0x00;
TMR3L = 0x00;                      // Recarrega os valores iniciais do contador
PIR2bits.TMR3IF = 0;               // Limpa o sinalizador de overflow
```

Modo Contador:

```
// Rotina de Inicialização
T3CON = 0x03;             // Timer3: 8 bits, PRESCALER 1:1, clock externo
TMR3H = 0xFF;
TMR3L = 0xFD;       // Carrega valores iniciais do Timer3 (0xFFFD = 65.533)
PIR2bits.TMR3IF = 0;      // Limpa flag de overflow do Timer3

// Exemplo de rotina de utilização
while(!PIR2bits.TMR3IF);           // Aguarda atingir o overflow
TMR3H = 0xFF;
TMR3L = 0xFD;                      // Recarrega os valores iniciais do contador
PIR2bits.TMR3IF = 0;               // Limpa o sinalizador de overflow
LATDbits.LATD0 = !LATDbits.LATD0;  // Inverte o estado da saída
```