Cerne

Conhecimento para o Desenvolvimento

Manual de Programação

Índice

Capítulo 0 – Aspectos da PCL 1001

Capítulo 1 - Comandos e Funções da PCL1001

1.1 Acionando as saídas

1.2 Lendo as entradas

1.3 Comunicação Serial

1.4 Data e Hora

1.5 Atrasos

1.6 PWM

1.7 Acessando o Lcd

1.8 Velocímetro

1.9 Contador

1.10 Frequencímetro

1.11 Comparador

1.12 AD

1.13 DA

1.14 Cronômetro

1.15 Matemática

1.16 Jogo de Palavras

1.17 Periodímetro

1.18 Velocidade Angular

1.19 Memória EEPROM

1.20 Chamada de Telefone

1.21 Chamadas rotinas e saltos

1.22 Controle da Máquina

1.23 Variáveis

1.24 Constantes Químicas

1.25 Controle da Saída

Capítulo 2 – Controle de Fluxo

2.1 Laço WHILE

2.2 Laço REPEAT

2.3 Seleção de Casos

Capítulo 3 - Operadores Lógicos e Estruturas Condicionais Lógicas

3.1 Operadores

3.2 Estruturas Condicionais Lógicas

Capítulo 4 – Exemplos de Programação

Capítulo 0

Aspectos da PCL 1001

A PCL 1001 é um projeto nacional de baixo custo que agrega várias comandos que facilitam a programação de qualquer evento em uma linguagem de fácil entendimento chamada *AutoEasy.* Ao adquirir a PCL o desenvolvedor pode baixar uma licença da AutoEasy para desenvolvimento dos seus trabalhos.

Esta placa pode ser usada em diversos modos, como:

- Robótica Educacional;
- Mecatrônica;
- Telemetria;
- Controle de Processos Industriais;
- Automação Residencial;
- Controle de Aquários;
- Segurança;
- Sua imaginação...

Em seguida podemos apreciar os aspectos de hardware principais da PCL 1001:

- 6 entradas de contato seco;
- 5 saídas com controle de reversão;
- 1 saída de contato seco;
- Display LCD;
- Saída PWM;
- 2 saídas de Leds;
- AD Converter;
- DA Converter;
- RS-232 Full Duplex;
- Possibilidades de expansão de I/Os através da porta serial.
- Interface Paralela;
- Contas Matemáticas em 16 bits.

Capítulo 1

Comandos e Funções da PCL1001

1.1 Acionando as saídas

SET

Sintaxe:

set(parâmetro)

Descrição:

Este tipo de comando aciona o relê ou o led especificado em parâmetro. O parâmetro deve ser um número entre 1 e 6 ou led1 ou led2. Caso o parâmetro do relê seja “all”, todos os relês serão acionados.

Exemplo:

```
set(all)          ; Liga todos os relês
set(1)            ; Liga relê 1
set(2)            ; Liga relê 2
set(led1)                 ; Liga o Led1
```

INV

Sintaxe:

inv(parâmetro)

Descrição:

Este tipo de comando inverte o relê especificado em parâmetro. O parâmetro deve ser um número entre 1 e 5.

Exemplo:

```
inv(1)            ; Liga relê 1
inv (2)           ; Liga relê 2
```

CLR

Sintaxe:

clr(parâmetro)

Descrição:

Este tipo de comando desliga o relê ou o led especifícado em parâmetro. O parâmetro deve ser um número entre 1 e 6 ou led1 ou led2. Caso o parâmetro do relê seja “all”, todos os relês serão desligados.

Exemplo:

```
clr(all)        ; Desliga todos os relês
clr(1)          ; Desliga relê 1
clr(2)          ; Desliga relê 2
clr(led1)       ; Desliga o Led1
clr(led2)       ; Desliga o Led2
```

OSC

Sintaxe:

osc(parâmetro)

Descrição:

Este comando oscila o relê ou o led especifícado em parâmetro. O parâmetro deve ser um número entre 1 e 6 ou led1 ou led2.

Exemplo:

```
osc(1)          ; Oscila com frequência de 1 HZ o relê 1
osc(2)          ; Oscila com frequência de 1 HZ o relê 2
osc(led1)       ; Oscila com frequência de 1 HZ o led1
osc(led2)       ; Oscila com frequência de 1 HZ o led2
```

TOGGLE

Sintaxe:

toggle(parâmetro)

Descrição:

Efetua a inversão do estado atual da saída. Caso a saída esteja ativa, a mesma irá desligar e vice-versa.

Exemplo:

```
toggle(out(1))          ; caso a saída 1 esteja ligada, a mesma irá desligar
```

1.2 Lendo as entradas

IF SWITCH(número_da_entrada_digital) THEN

Sintaxe:

```
if switch(número_entrada_digital) then
                .
                .
                .
                comandos 1
                .
                .
                .

else
                .
                .
                .
                comandos 2
                .
                .
                .
end if
```

Descrição:

Testa a condição atual da entrada definida em número_entrada_digital. Se for verdadeiro, executa os comandos 1. Se for falso, executa os comandos 2. O número_entrada_digital deve ser um número entre 1 e 8.

Exemplo:

```
if switch(1) then                ;Testa entrada 1
        cld                      ;se verdadeiro, executa estes comandos...
        pause(1)
        disp(1)(Switch 1 fechado)
        txdata(Switch 1 fechado)
        pause(1)
else                             ;se falso executa o que está após else
        cld
        pause(1)
        disp(1)(Switch 1 aberto)
        txdata(Switch 1 aberto)
        pause(1)
end if
```

IF NOT SWITCH(número_da_entrada_digital) THEN

Sintaxe:

```
if not switch(número_entrada_digital) then
                .
                .
```

```
                .
                comandos 1
                .
                .
                .

        else
                .
                .
                .
                comandos 2
                .
                .
                .
        end if
```

Descrição:

Testa a condição atual da entrada definida em número_entrada_digital. Se for falso, executa os comandos 1. Se for verdadeiro, executa os comandos 2. O número_entrada_digital deve ser um número entre 1 e 8.

Exemplo:

```
if not switch(1) then                   ;Testa entrada 1
        cld                             ;se falso, executa estes comandos...
        pause(1)
        disp(1)(Switch 1 fechado)
        txdata(Switch 1 fechado)
        pause(1)
else                                    ;se verdadeiro executa o que está após else
        cld
        pause(1)
        disp(1)(Switch 1 aberto)
        txdata(Switch 1 aberto)
        pause(1)
end if
```

1.3 Comunicação Serial

PANEL

Descrição:

A PCL1001 pode comunicar-se com o painel PE01 externo conectado na porta serial. Para isso existe uma série de comandos que se corretamente seguidos farão o mesmo funcionar.

Exemplo

```
mode com with panel         ; prepara porta para funcionar no modo painel
mode panel date             ; prepara para funcionar como mostrador de data
```

ACESSANDO A PLACA DE EXPANSÃO

Descrição:

A PCL1003 pode ter as entradas e saídas, permitindo um total de 14 entradas e 14 saídas. Para isto, a placa de i/o deve estar conectada na rs232 da PCL. Os comandos para ligar/desligar relês são os mesmos assim como os comandos de teste de entrada digital.

Exemplo

```
set(14)  ; liga o relê 14
clr(14)   ; desliga o relê 14

if switch(7) then          ; se entrada 7 verdadeira...
```

TXPROG

Sintaxe:

txprog

Descrição:

Este comando envia pelo canal serial todo o programa gravado na memória EEPROM. Este comando é importante para debugação.

Exemplo:

```
txprog          ;envia pelo canal serial o programa residente na memória EEPROM.
```

ECHO DATA

Sintaxe:

echo data on ou off

Descrição:

Caso este comando esteja ativo, todos os caracteres recebidos pela serial da placa serão retornados pela mesma. O estado default é desligado.

TERMINAL

Sintaxe:

term (tempo)

Descrição:

Apresenta no display todos os caracteres recebidos do canal serial durante <tempo> em segundos. Casotempo seja 0, o comando term será executado infindavelmente.

BAUD RATE

Sintaxe:

baud(taxa de transmissão)

Descrição:

Permite alterar o taxa de transmissão da máquina. Apenas dois baud rates estão disponíveis, o de 9600 e19200 bps.

Exemplo:

```
baud(9600)      ; configura taxa de transmissão para 9600 bps (default)
baud(19200)     ; configura taxa de transmissão para 19200 bps
```

TXDATA BIN

Sintaxe:

txdata(bin$(variável))

Descrição:

Este comando envia pelo canal serial o conteúdo da variável em binário especificada em variável.

Exemplo:

```
a%=1000                 ; carrega a% com 1000
txdata(bin$(a%))        ; envia o mesmo pelo canal serial em binário
```

TXDATA HEX

Sintaxe:

txdata(hex$(variável))

Descrição:

Este comando envia pelo canal serial o conteúdo da variável em hexadecimal especificada em variável.

Exemplo:

```
a%=1000                 ; carrega a% com 1000
txdata(hex$(a%))        ; envia o mesmo pelo canal serial em hexadecimal
```

TXDATA OCT

Sintaxe:

txdata(oct$(variável))

Descrição:

Este comando envia pelo canal serial o conteúdo da variável em octal especificada em variável.

Exemplo:

```
a%=1000                 ; carrega a% com 1000
txdata(oct$(a%))        ; envia o mesmo pelo canal serial em octal
```

TXDATA DEC

Sintaxe:

txdata(variável)

Descrição:

Este comando envia pelo canal serial o conteúdo da variável em decimal especificada em variável.

Exemplo:

```
a%=1000                 ; carrega a% com 1000
txdata(a%)              ; envia o mesmo pelo canal serial em decimal
```

TXDATA STOPWATCH

Sintaxe:

txdata(stopwatch$)

Descrição:

Envia pelo canal serial o valor dos registradores de contagem do cronômetro.

Exemplo:

txdata(stopwatch$)

TXDATA(READ_MEMORY$)

Sintaxe:

txdata(read_memory$)

Descrição:

Envia pelo canal serial o caracter posicionado por addr_memory.

Exemplo:

addr_memory(100)
txdata(read_memory$)

TXDATA DATE$

Sintaxe:

txdata(date$)

Descrição:

Envia pelo canal serial a data corrente programada no dispositivo.

Exemplo:

txdata(date$) ; envia pelo canal serial a data corrente

TXDATA

Sintaxe:

txdata(string_de_caracteres)

Descrição:

Envia pelo canal serial os dados contidos em string_de_caracteres.

Exemplo:

txdata(Apice Tecnologia)

TXDATA_ASC

Sintaxe:

txdata_asc(código da tabela ASCII)

Descrição:

Envia pelo canal serial os dados contidos em código da tabela ASCII.

Exemplo:

txdata_asc(65,66,67,68,69,) ;transmite a string “ABCDE” pelo canal serial.

TXDATA TIME$

Sintaxe:

txdata(time$)

Descrição:

Envia pelo canal serial a hora corrente programada no dispositivo.

Exemplo:

txdata(time$) ; envia pelo canal serial a data corrente

TXDATA FREQ$

Sintaxe:

txdata(freq$)

Descrição:

Envia pelo canal serial o número de pulsos ocorridos em 1 segundo na entrada 6.

Exemplo:

```
label1 for novamente
input6 as frequecimeter

novamente

txdata(freq$)
goto novamente
```

RXDATA

Sintaxe:

```
if rxdata=”parâmetro” then
        txdata(Caractere recebido!)
end if
```

Descrição:

Verifica se há algum dado no buffer de recepção, e caso haja, faz o teste com o caractere recebido e com o parâmetro.

Exemplo:

```
if rxdata=”N” then
        txdata(Caractere recebido com sucesso!)
end if
```

1.4 Data e Hora

IF TIME$>

Sintaxe:

```
if time$>hora_minuto then
        .
        .
        comandos
        .
        .
end if
```

Descrição:

Este comando testa se hora_minuto são maiores que o do RTC da máquina. Caso seja, comandos serão executados. Este comando também pode funcionar como um despertador.

Exemplo:

```
if time$>06_50 then                  ; se passou das 6h50mim então...
        set(1)                       ; liga o relê 1
        disp(1)(E hora de acordar!)  ; apresenta mensagem no display
end if
```

IF TIME$<

Sintaxe:

```
if time$<hora_minuto then
        .
        .
```

```
        comandos
        .
        .
end if
```

Descrição:

Este comando testa se hora_minuto são menores do que o do RTC da máquina. Caso seja, comandos serão executados. Este comando também pode funcionar como um despertador.

Exemplo:

```
if time$<06_50 then         ; se não passou das 6h50mim então...
        clr(1)              ; desliga o relê 1
        disp(1)(Pode dormir...)  ; apresenta mensagem no display
end if
```

IF DATE$

Sintaxe:

```
if date$=data de teste then
```

Descrição:

Verifica se a data da máquina é igual a data de teste.

Exemplo

```
 if date$="10_01"  then    ; hoje é 10 de Janeiro?
      Set(3)               ; sim, então liga o relê 3
 end if
```

AJUSTE DE DATA E HORA

Sintaxe:

```
time$="20_57_00"
date$="21_01_05"
```

Descrição:

Ajusta via programa a data e hora do sistema.

DISP DATE$

Sintaxe:

disp(número da linha)(date$)

Descrição:

Apresenta no display a data corrente programada no dispositivo.

Exemplo:

```
disp(1)(date$ )          ; escreve na linha 1 do display  a data corrente
disp(2)(date$)           ; escreve na linha 2 do display  a data corrente
```

DISP TIME$

Sintaxe:

disp(número da linha)(time$)

Descrição:

Apresenta no display a hora corrente programada no dispositivo.

Exemplo:

```
disp(1)(time$ )          ; escreve na linha 1 do display a hora corrente
disp(2)(time$)           ; escreve na linha 2 do display a hora corrente
```

IF TIME$

Sintaxe:

```
if time$=hora_minuto then
        .
        .
        comandos
        .
        .
end if
```

Descrição:

Este comando testa se hora_minuto são iguais ao do RTC da máquina. Caso seja, os comandos são executados. Este comando também pode funcionar como um despertador.

Exemplo:

```
if time$=20_50 then
        set(1)
        disp(1)(E hora de acordar!)
end if
```

DISP DATE$

Sintaxe:

disp(número da linha)(date$)

Descrição:

Apresenta no display a data corrente programada no dispositivo.

Exemplo:

```
disp(1)(date$ )          ; escreve na linha 1 do display  a data corrente
disp(2)(date$)           ; escreve na linha 2 do display  a data corrente
```

1.26 Atrasos

DELAY_MS(VARIÁVEL)

Sintaxe:

variável=valor
delay_ms(variável)

Descrição:

O comando delay_ms conta um tempo em milisegundos em função do parâmetro carregado na variável.

Exemplo

```
a%=1000          ; atribui 1000 a a%
delay_ms(a%)     ; conta 1000 ms
```

DELAY_SEG(VARIÁVEL)

Sintaxe:

variável=valor
delay_seg(variável)

Descrição:

O comando delay_seg conta um tempo em segundos em função do parâmetro carregado na variável.

Exemplo

```
a%=10                ; atribui 10 a a%
delay_seg(a%)        ; conta 10 segundos
```

DELAY_MS

Sintaxe:

delay_ms(tempo)

Descrição:

Este comando conta um tempo em milisegundos especificado em tempo.

Exemplo:

```
set(1)                  ; liga o relê 1
delay_ms(1000)                  ; aguarda 1000 ms
clr(1)                  ; desliga o relê 1
```

DELAY_SEG

Sintaxe:

delay_seg(tempo)

Descrição:

Realiza a contagem de um tempo em segundos especificado em tempo.

Exemplo:

```
set(1)                  ; liga o relê 1
delay_seg(10)           ; aguarda 10 seg
clr(1)                  ; desliga o relê 1
```

1.17 PWM

INC PWM

Sintaxe:

inc pwm

Descrição:

Incrementa o *duty cycle* da saída PWM. A frequência do PWM é de 1kHz.

Exemplo:

```
inc pwm                    ; incrementa o ciclo ativo da saída
```

DEC PWM

Sintaxe:

```
dec pwm
```

Descrição:

Decrementa o *duty cycle* da saída PWM. A frequência do PWM é de 1kHz.

Exemplo:

```
dec pwm                    ; decrementa o ciclo ativo da saída
```

SET PWM

Sintaxe:

```
set pwm
```

Descrição:

Este comando coloca o *duty cycle* em 100%.

Exemplo:

```
set pwm                    ; põe o duty cycle em 100%
```

CLR PWM

Sintaxe:

```
clr pwm
```

Descrição:

Este comando coloca o *duty cycle* em 0%.

Exemplo:

```
clr pwm                    ; põe o duty cycle em 0%
```

PWM

Sintaxe:

pwm(duty cycle)

Descrição:

Ajusta o *duty clycle* da saída PWM. O *duty cycle* deve estar entre 0 até 100.

Exemplo:

```
pwm(50)                    ; ajusta o PWM para 50%
```

1.7 Acessando o LCD

CLD

Sintaxe:

cld

Descrição:

Limpa o display.

Exemplo:

```
cld        ;Limpa as duas linhas do display
```

DISP

Sintaxe:

disp(número_da_linha)(string_de_caracteres)

Descrição:

Apresenta em número_da_linha a string_de_caracteres.

Exemplo:

```
cld                        ;Limpa display
disp(1)(PCL1003)           ; Apresenta na linha 1 do display “PCL1003”
disp(2)(A placa inteligente)   ; Apresenta na linha 2 do display “A placa inteligente”
```

DISP DECIMAL

Sintaxe:

disp(número da linha)(ad_read$(decimal))

Descrição:

Este comando apresenta em número da linha a tensão de entrada do convesor analógico em decimal.

Exemplo:

```
disp(1)(ad_read$(decimal))
```

DISP VOLTS

Sintaxe:

disp(número da linha)(ad_read$(voltage))

Descrição:

Apresental através do número da linha a tensão de entrada do convesor analógico.

Exemplo:

```
disp(1)(ad_read$(voltage))
```

DISP FREQ$

Sintaxe:

disp(número da linha)(freq$)

Descrição:

Apresenta o número de pulsos que ocorreram em 1 segundo na entrada 6.

Exemplo:

```
label1 for novamente
input6 as frequecimeter

novamente

disp(1)(freq$)
goto novamente
```

DISP MONTH

Sintaxe:

disp(número da linha)(month$)

Descrição:

Apresenta em número da linha o mês corrente do rtc da placa.

Exemplo:

```
disp(1)(month$)        ; apresenta na linha 1 do display o mês corrente
```

DISP SPEED

Sintaxe:

disp(número da linha)(speed$)

Descrição:

Apresenta em número da linha, a última medição de velocidade realizada. Para que uma nova medição seja feita, o comando on speed deve ser utilizado novamente.

DISP(VARIÁVEL)

Sintaxe:

disp(número da linha)(variável%)

Descrição:

Mostra em número da linha do display o valor corrente da variável.

Exemplo

```
a%=100                    ;carrega a% com 100 decimal
disp(1)(a%)               ;apresenta conteúdo da variável
```

DISP BIN$

Sintaxe:

disp(número da linha)(bin$(variável))

Descrição:

Mostra em número da linha do display o valor corrente da variável em binário.

Exemplo

```
a%=1000                   ;carrega a% com 1000 decimal
disp(1)(bin$(a%))         ;apresenta conteúdo da variável em binário
```

DISP HEX$

Sintaxe:

disp(número da linha)(hex$(variável))

Descrição:

Mostra em número da linha do display o valor corrente da variável em hexadecimal.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
disp(1)(hex$(a%))       ;apresenta conteúdo da variável em hexa
```

DISP OCT$

Sintaxe:

disp(número da linha)(oct$(variável))

Descrição:

Mostra em número da linha do display o valor corrente da variável em octal.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
disp(1)(oct$(a%))       ;apresenta conteúdo da variável em octal
```

OSC DISP

Sintaxe:

osc(disp)

Descrição:

Este comando faz com que a mensagem que fique oscilando no display a uma frequencia de 1Hz.

NO OSC DISP

Sintaxe:

no osc(disp)

Descrição:

Este comando faz com que o display saia do modo oscilante.

DISP PULSE COUNTER

Sintaxe:

disp(número da linha)(pulse_counter$)

Descrição:

Este comando permite que seja apresentado em númer da linha o valor corrente da contagem de pulsos externa. O parâmetro número da linha deve ser 1 ou 2.

Exemplo:

```
disp(1)(pulse_counter$)                    ; apresenta na linha 1 do display o valor
                                           ; corrente da contagem de pulsos
```

DISP STOPWATCH

Sintaxe:

disp(número da linha)(stopwatch$)

Descrição:

Apresenta o estado do contador no display. O número da linha determina em qual linha esta será apresentada.

Exemplo:

```
begin stopwatch
disp(1)(stopwatch$)
```

ROTATE DISPLAY TO LEFT

Sintaxe:

rotate display to left

Descrição:

Executa uma rotação para a esquerda com a mensagem que está apresentada no display.

Exemplo:

```
disp(1)(  *** PCL 1003 *** )      ; escreve no display “ *** PCL 1003 *** )
rotate display to left            ; faz uma rotação para a esquerda
```

ROTATE DISPLAY TO RIGHT

Sintaxe:

rotate display to right

Descrição:

Executa uma rotação para a direita com a mensagem que está apresentada no display.

Exemplo:

```
disp(1)(  *** PCL 1001 *** )      ; escreve no display " *** PCL 1001 *** )
rotate display to right           ; faz uma rotação para a direita
```

1.8 Velocímetro

ON SPEED

Sintaxe:

on speed

Descrição:

Ativa o medidor de velocidade externa da máquina. As entradas 1 e 2 funcionam de modo a captar a diferença entre os tempos. O espaço deve estar previamente definido em space_default.

Exemplo

```
 space_default=100       ; diferença entre os sensores é de 100 m
 on speed                ; habilita a medição de velocidade
 disp(1)(speed$)         ; apresenta o resultado da medição
```

SPACE DEFAULT

Sintaxe:

space_default

Descrição:

Este comando determina o espaço que há entre os sensores. O valor máximo de space_deafult é 255 metros.

IF SPEED$=

Sintaxe:

if sped$=velocidade then

Descrição:

Verifica se a velocidade é igual á especificada.

Exemplo

```
if speed$=30 then          ; velocidade é igual a 30 m/s?
    set(1)                 ; sim, então liga o relê 1
end if
```

IF SPEED$>

Sintaxe:

if sped$>velocidade then

Descrição:

Verifica se a velocidade é maior que á especificada.

Exemplo

```
if speed$>30 then          ; velocidade é maior que 30 m/s?
    set(1)                 ; sim, então liga o relê 1
end if
```

IF SPEED$<

Sintaxe:

if sped$<velocidade then

Descrição:

Verifica se a velocidade é menor que á especificada.

Exemplo

```
if speed$<30 then          ; velocidade é menor que 30 m/s?
    set(1)                 ; sim, então liga o relê 1
end if
```

1.9 Contador

INPUT6 AS COUNTER

Sintaxe:

input6 as counter

Descrição:

Promove o funcionamento da entrada 6 como um contador de pulsos.

Exemplo:

input6 as counter

COUNTER

Sintaxe:

counter(valor)

Descrição:

Este tipo de comando permite que seja contado n pulsos especificados por valor. O valor deve ser um número entre 0 e 65535.

Exemplo:

```
label1 for again        ; cria label chamado again
input6 as counter       ; faz com que a entrada 6 funcione como contador
counter(1000)           ; ajusta para contar 1000 pulsos

again

if counter then         ; quando a contagem se encerrar...
    set(1)              ; processa esses comandos
    delay_ms(1000)
    clr(1)
    delay_ms(1000)
end if
goto again              ; salta para again
```

CLEAR PULSE COUNTER

Sintaxe:

clear pulse_counter$

Descrição:

Este comando limpa os registradores internos de contagem de pulsos externo.

BEGIN PULSE COUNTER

Sintaxe:

begin pulse_counter$

Descrição:

Este comando habilita o funcionamento do contador de pulsos.

STOP PULSE COUNTER

Sintaxe:

stop pulse_counter$

Descrição:

Este comando desabilita o funcionamento do contador de pulsos.

INPUT6 AS PULSE_COUNTER

Sintaxe:

input6 as pulse_counter

Descrição:

Este comando faz com que a entrada 6 funcione como contador de pulsos.

IF PULSE_COUNTER$=

Sintaxe:

if pulse_counter$=parâmetro then

Descrição:

Este comando verifica se o contador de pulsos é igual a parâmetro. O parãmetro deve ser um número compreendido entre 0 e 65535 inclusive.

Exemplo:

```
if pulse_counter$=10000 then          ; se atingiu 10000 contagens então...
        txdata(10000 pulsos!)
        clear pulse_counter$
end if
```

IF PULSE_COUNTER$>

Sintaxe:

if pulse_counter$>parâmetro then

Descrição:

Este comando verifica se o contador de pulsos é maior que o parâmetro. O parâmetro deve ser um número compreendido entre 0 e 65535 inclusive.

Exemplo:

```
if pulse_counter$>5000 then          ; se passou 5000 contagens então...
        txdata(5000 pulsos!)
        clear pulse_counter$
end if
```

IF PULSE_COUNTER$<

Sintaxe:

```
if pulse_counter$<parâmetro then
```

Descrição:

Este comando verifica se o contador de pulsos é menor que o parâmetro. O parâmetro deve ser um número compreendido entre 0 e 65535 inclusive.

Exemplo:

```
if pulse_counter$<100 then           ; se é menor que 100 contagens então...
        txdata(100 pulsos!)
        clear pulse_counter$
end if
```

IF INP$=

Sintaxe:

```
if inp$=valor then
```

Descrição:

Este comando verifica se a entrada de dados corresponde ao valor especificado em valor.

Exemplo:

```
if inp$=1  then                      ;se a entrada 1 estiver fechada...
        txdata(100 pulsos!)          ;executa comandos
        clear pulse_counter$
end if
```

1.10 Frequencímetro

INPUT6 AS INPUT

Sintaxe:

input6 as input

Descrição:

Este tipo de comando permite que a entrada 6 funcione como uma entrada normal.

Exemplo:

input6 as input

INPUT6 AS FREQUECIMETER

Sintaxe:

input6 as frequecimeter

Descrição:

Este tipo de comando permite que a entrada 6 funcione como uma entrada de freqüencímetro.

Exemplo:

input6 as frequecimeter

RPM$

Sintaxe:

disp(número da linha)(rpm$)

Descrição:

Apresenta em número da linha o valor atual da medição do tacógrafo.

ALETAS

Sintaxe:

aletas_default= valor

Descrição:

É utilizado no comando RPM para medir frequencias externas. Na verdade, aletas é um divisor caso o encoder tenha mais de uma aleta. O valor default é 1 e o máximo permitido é 60.

1.11 Comparador

VREF

Sintaxe:

vref=tensão de referência

Descrição:

Este comando gera uma tensão para ser comparada com a tensão de entrada. Caso a tensão de referência seja maior que a tensão de entrada, a função comparator retorna ao 0. Caso contrário, retorna ao 1.

Exemplo:

```
vref=2.00
label1 for again

again

        if comarator then           ; se a tensão de entrada for maior que a de referência...
                set(1)              ; liga o relê 1
        end if

        if not comparator then      ; se a tensão de entrada for menor que a de referência...
                clr(1)              ; desliga o relê 1
        end if
```

1.12 AD

MAX VOLTAGE

Sintaxe:

max voltage tensão_máxima

Descrição:

Permite que sejam medidas e apresentadas no display tensões acima de 5V. Para isso um circuito condicionador de sinal deve ser utilizado. Deve-se lembrar que a tensão máxima de medida é de 65V e que o valor default é 5V.

Exemplo:

```
max voltage 50          ; permite que sejam medidas tensões até 50V
```

IF AD_READ$>

Sintaxe:

if ad_read$>tensão de referência then

Descrição:

Verifica se a tensão de entrada é maior que a tensão de referência. Caso este teste seja afirmativo, os comandos seguintes serão executados.

Exemplo:

```
if ad_read$>2.00 then          ; se a tensão de entrada for maior que 2 V
        set(1)                 ; então liga o relê 1
end if
```

IF AD_READ$<

Sintaxe:

if ad_read$<tensão de referência then

Descrição:

Verifica se a tensão de entrada é menor que a tensão de referência. Caso este teste seja afirmativo, os comandos seguintes serão executados.

Exemplo:

```
if ad_read$<2.00 then          ; se a tensão de entrada for menor que 2 V
        set(1)                 ; então liga o relê 1
end if
```

IF AD_READ$>=

Sintaxe:

if ad_read$>=tensão de referência then

Descrição:

Verifica se a tensão de entrada é maior ou igual a tensão de referência. Caso este teste seja afirmativo, os comandos seguintes serão executados.

Exemplo:

```
if ad_read$>=2.00 then         ; se a tensão de entrada for maior ou igual a2 V
        set(1)                 ; então liga o relê 1
end if
```

IF AD_READ$<=

Sintaxe:

if ad_read$<=tensão de referência then

Descrição:

Verifica se a tensão de entrada é menor ou igual a tensão de referência. Caso este teste seja afirmativo, os comandos seguintes serão executados.

Exemplo:

```
if ad_read$<=2.00 then          ; se a tensão de entrada for menor ou igual a 2 V
        set(1)                  ; então liga o relê 1
end if
```

IF AD_READ$=

Sintaxe:

if ad_read$=tensão de referência then

Descrição:

Verifica se a tensão de entrada é igual a tensão de referência. Caso este teste seja afirmativo, os comandos seguintes serão executados.

Exemplo:

```
if ad_read$=2.00 then           ; se a tensão de entrada for  igual a 2 V
        set(1)                  ; então liga o relê 1
end if
```

IF AD_READ$> AND AD_READ<

Sintaxe:

if ad_read$>tensão de referência mínima and ad_read$<tensão de referência máxima then

Descrição:

Este comando permite que o ad da máquina funcione como um comparador proporcionando, que, dentro de uma faixa de tensões o teste condicional seja positivo.

Exemplo:

```
        if ad_read$>2.00 and ad_read$<4.00 then     ; se a tensão medida for maior que 2.00 V e
menor 3.00
                set(1)                              ; então liga o relê 1
        end if
```

1.13 DA

OUT_VOLTAGE

Sintaxe:

out_voltage(tensão de saída)

Descrição:

Cria uma tensão de saída correspondente à tensão de saída. O valor de tensão de saída deve estar entre 0 e 5.

Exemplo:

```
out_voltage(3.2)
```

1.14 Cronômetro

BEGIN STOPWATCH

Sintaxe:

begin stopwatch

Descrição:

Inicia o funcionamento do cronômetro interno da máquina.

Exemplo:

```
begin stopwatch
disp(1)(stopwatch$)
```

STOP STOPWATCH

Sintaxe:

stop stopwatch

Descrição:

Este comando pára o funcionamento do cronômetro interno da máquina.

Exemplo:

```
stop stopwatch
disp(1)(stopwatch$)
```

CLEAR STOPWATCH

Sintaxe:

clear stopwatch

Descrição:

Executa a limpeza dos registradores de contagem de cronômetro.

Exemplo:

```
clear stopwatch
disp(2)(stopwatch$)
```

1.15 Matemática

SQUARE

Sintaxe:

```
variável %=valor                    ;carrega variável% com valor
variável%=square  variável%    ; tira a raiz quadrada.
```

Descrição:

A função square permite que se retire a raiz quadrada de um número qualquer até 65535. A mesma é armazenada em uma variável.

Exemplo

```
a%=100                    ; carrega a% com valor
a%=square a%              ; tira a raiz quadrada.
```

PERCENTAGE

Sintaxe:

```
variável%=valor           ;carrega a% com valor
variável%=variável%+ 20%  ;soma mais 20 % da própria variável.
```

Descrição:

A função percentage calcula a porcentagem de um número com base nele mesmo. O resultado é armazenada na própria variável.

Exemplo

```
a%=100                    ;carrega a% com 100 decimal
a%=a%+ 20%                ;soma mais 20 % da própria variável, ou seja, 120.
```

```
B%=1000                 ;carrega b% com 1000 decimal
b%=b% - 50 %            ;retira 50 porcento da mesma.
```

ADD

Sintaxe:

```
variável%=1
variável%=variável% + constante
```

Descrição:

A função soma uma constante a uma variável do sistema. O sistema possui oito variáveis de usuário.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
a%=a%+ 20               ;soma 20 ao conteúdo da variável
```

SUB

Sintaxe:

```
variável%=1
variável%=variável% - constante
```

Descrição:

A função subtrai uma constante de uma variável do sistema. O sistema possui oito variáveis de usuário.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
a%=a% - 50              ;subtrai 50 da variável
```

MUL

Sintaxe:

```
variável%=1
variável%=variável% * constante
```

Descrição:

Esta função multiplica uma constante e armazena em uma variável do sistema. O sistema possui oito variáveis de usuário.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
a%=a% - 50              ;subtrai 50 da variável
```

DIV

Sintaxe:

variável%=variável% / 2

Descrição:

A função divide uma constante por uma variável do sistema. O resultado é armazenado na própria variável.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
a%=a% /  50             ;subtrai 50 da variável
```

POW

Sintaxe:

variável%=variável% ^ 2

Descrição:

Eleva variável% pela constante e salva nela mesma. O sistema possui oito variáveis de usuário.

Exemplo

```
a%=10                   ;carrega a% com 10 decimal
a%=10%^2                ;eleva a segunda potência
```

IF VARIÁVEL=

Sintaxe:

if variável%=constante then

Descrição:

Verifica se a variável é igual à constante. O sistema possui oito variáveis de usuário.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
if a%=100 then          ;se a% igual a 100 então...
```

IF VARIÁVEL>

Sintaxe:

if variável%>constante then

Descrição:

Verifica se a variável é maior que à constante. O sistema possui oito variáveis de usuário.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
if a%>50 then           ;se variável maior que 50 então
```

IF VARIÁVEL<

Sintaxe:

if variável%<3 then

Descrição:

Verifica se a variável é menor que à constante.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
if a%<150 then          ; se for menor que 150 então...
```

HIPOT

Sintaxe:

a%=hipot(b%,c%)

Descrição:

Permite calcular a hipotenusa de um triângulo retângulo. A variável a% é o resultado do cálculo, ou seja, a hipotenusa e as variáveis b% e c% são os catetos.

Exemplo:

```
b%=6              ; atribui 6 a variável b%
c%=8              ; atribui 8 a variável c%
a%=hipot(b%,c%)   ; calcula a hipotenusa e armazena em a%
disp(1)(a%)       ; apresenta o resultado, ou seja, a hipotenusa
```

CIRCUNF

Sintaxe:

b%=raio
a@=circunf(b%)

Descrição:

Permite calcular a circunferência de um círculo. O raio deve estar previamente carregado em b%.

Exemplo:

```
 b%=2                ; atribui 2 a variável b%
a@=circunf(b%)       ; calcula a circunferencia e armazena em a@
disp(1)(a@)          ; apresenta o resultado de a@
```

AREA_CIRCUNF

Sintaxe:

b%=raio
a@=area_circunf(b%)

Descrição:

Permite calcular a área da circunferência de um círculo. O raio deve estar previamente carregado em b%.

Exemplo:

```
 b%=2                    ; atribui 2 a variável b%
a@=area_circunf(b%)      ; calcula a área da circunferencia e armazena em a@
disp(1)(a@)              ; apresenta o resultado de a@
```

CONSTANTES

```
pi=3.14          ; pi
e=2.72           ; exponencial
g=9.81           ; gravidade da terra
c=3.00           ; velocidade da luz
cte_g=6.67       ; constante gravitacional
Na=6.02          ; Número de Avogrado
R=8.31           ; Constante universal dos gases
c2=8.99          ; Relação massa-energia
&0=8.85          ; Constante de permissividade
u0=1.26          ; Constante de permeabilidade
h=6.63           ; Constante de Plank
k=1.38           ; Constante de Boltzmann
e=1.60           ; Carga elementar
me=9.11          ; Massa do elétron
mp=1.67          ; Massa do próton
```

```
mn=1.68          ; Massa do neutrôn
md=3.34          ; Massa dp dêuteron
a=5.29           ; Raio de Bohr
ub=9.27          ; Magnéton de Bohr
atm=1.01         ; Atmosfera da Terra
```

Descrição:

Iguala uma variável de duas casas decimais uma constante.

Exemplo

```
a@=pi      ; atribui a a@ pi, ou seja, 3.14
b@=e       ; atribui a b@ e, ou seja, 2.72
```

SIN(), COS(), TAN()...

Funções:

```
SIN        ; calcula o seno de um ângulo
COS        ; calcula o cosseno de um ângulo
TAN        ; calcula a tangente de um ângulo
SEC        ; calcula a secante de um ângulo
COSEC      ; calcula a cosecante de um ângulo
COTAN      ; calcula a cotangente de um ângulo
EXP        ; calcula a exponencial de um número
LOG        ; calcula o log na base 10 de uma constante
LN         ; calcula o log na base e
```

Descrição:

O parâmetro deve sempre vir em graus para que seja calculada na função acima. A mesma será armazenada em uma variável de casa decimal dupla.

Exemplo

```
a@=sin(30)     ; atribui a a@, o seno de 30 graus, ou seja, 0,5
b@=cos(90)     ; atribui a b@, o coseno de 90 graus, ou seja, 0.
```

FATORIAL

Sintaxe:

variável= constante !

Descrição:

A função fatorial retorna um número mediante o valor informado por constante. Constante deve ser um número entre 0 e 8 inclusive.

Exemplo

```
a%=0!          ; atribui o fatorial de 0 a a%
```

```
b%=1!              ; atribui o fatorial de 1 a b%
c%=5!              ; atribui o fatorial de 5 a c%
```

1.16 Jogo

GAME

Sintaxe:

game

Descrição:

Inicia o jogo de palavras existente na placa. Caso o jogador tenha achado a frase, a palavra game será positiva. Caso contrário negativa.

Exemplo

```
game          ; chama o jogo existente na placa

if not game then          ; se o jogador não encontrou a frase ...
    disp(1)(Voce perdeu...) ; apresenta mensagem que o jogo está perdido
end if

if game then
    disp(1)(Voce ganhou!)  ; apresenta mensagem que o jogo foi vencido
end if
```

1.17 Periodímetro

PERIODIMETER

Sintaxe:

disp(número da linha)(periodimeter$)

Descrição:

Esta funçao permite calcular o período da onda que chega através da entrada 6 do sistema. Não podemos esquecer que a entrada deve estar configurada como entrada de frequência.

Exemplo

```
input6 as frequecimeter       ; entrada 6 como entrada de pulsos
disp(1)(periodimeter$)        ; apresenta na linha 1 do display o período da onda
```

1.18 Velocidade Angular

ANG_SPEED

Sintaxe:

variável de duas casas decimais=ang_speed$

Descrição:

Calcula a velocidade angular e atribui a alguma variável. Para que este comando seja utilizado, a entrada 6 deve estar configurada como entrada para frequencímetro, para que satisfaça a equação W=2*pi*f.

Exemplo

```
input6 as frequecimeter      ; entrada 6 como entrada de pulsos
a@=ang_speed$                ; calcula a velocidade angular
```

1.19 Memória EEPROM

INC ADDR_MEMORY

Sintaxe:

inc addr_memory

Descrição:

Este comando incrementa o endereçamento da memória não volátil contida na máquina.

Exemplo:

```
inc addr_memory
write_memory(A)
```

DEC ADDR_MEMORY

Sintaxe:

dec addr_memory

Descrição:

Este comando decrementa o endereçamento da memória não volátil contida na máquina.

Exemplo:

```
dec addr_memory
write_memory(B)
```

WRITE_MEMORY

Sintaxe:

```
write_memory(caracter)
```

Descrição:

Este comando escreve na posição endereçada por addr_memory o caracter especificado pelo programa. Deve-se verificar que o comando escreve caracter e não strings.

Exemplo:

```
write_memory(J)
```

READ_MEMORY

Sintaxe:

```
if read_memory$=A then
        set(1)
end if
```

Descrição:

O comando verifica se na posição endereçada por add_memory existe o caracter usado para comparação. Caso exista, o comando é executado.

Exemplo:

```
addr_memory=100
if read_memory=A then
        txdata (Há o caracter A nesta posição de memória)
end if
```

1.20 Chamada de Telefone

CALLPHONE

Sintaxe:

```
call(número_do_telefone)
```

Descrição:

Efetua a geração de pulsos para a chamada de algum telefone.

Exemplo:

```
callphone(88316621)     ;liga para o telefone 88316621
```

1.21 Chamada a rotinas e saltos

LABEL

Sintaxe:

LABEL(N) FOR

Descrição:

Define um label para ser usado pelo comando GOTO. O Parâmetro N não pode exceder 8.

Exemplo:

```
label1 for Loop
label2 for Teste

        set(all)
        clr(all)
Loop

        disp(1)(Teste,,,)
        pause(1)
        cld
        pause(1)
        goto Loop
```

GOTO

Sintaxe:

goto LABEL

Descrição:

Volta para o LABEL especificado pelo comando label.

Exemplo:

```
        label Loop
        cld                     ;limpa o display
        pause(1)                        ;temporiza 1 segundo
        disp(1)(Aguarde . . .)  ;Escreve “Aguarde . . .” no display
        pause(1)                        ;temporiza 1 segundo
Loop:
        cld                     ;limpa o display
        clr(1)
```

```
pause(1)                    ;temporiza 1 segundo
set(1)
disp(1)(Rele 1 On)          ;Escreve “Aguarde . . .” no display
pause(1)                    ;temporiza 1 segundo
goto Loop                ;volta ao label Loop
```

CALL

Sintaxe:

call label identificador

Descrição:

Chama uma rotina definida pelo comando label.

Exemplo

```
 label1 for liga_rele        ; declara label liga_rele
  call liga_rele             ; chama liga_rele
 .
  .
programa do usuário
.
.
end                           ; fim do programa do usuário

liga_rele                   ; rotina para ligar o relê
     set(1)                 ; liga o relê 1
     set(2)                   ; liga o relê 2
     set(3)                   ; liga o relê 3
     return                   ; retorna da rotina
```

1.22 Controle da Máquina

STOP

Sintaxe:

stop

Descrição:

O comando cessa todo o processamento da máquina. O funcionamento só ocorre após um reset físico.

Exemplo:

```
if switch(1) then
        stop
```

```
end if
```

RESET

Sintaxe:

```
reset
```

Descrição:

Este tipo de comando resseta a máquina.

Exemplo:

```
if switch(1) then
        reset
end if
```

SHUTDOWN

Sintaxe:

```
shutdown on ou off
```

Descrição:

Caso o parâmetros seja on, a máquina desligará automaticamente após decorrido 1 minuto. Este comando é importante para economia de energia. O shutdown tanto pode estar on quanto off.

Exemplo:

```
shutdown on     ; após 1 minuto desliga a máquina
shutdown off    ; desabilita o comando shutdown
```

NO OPERATION

Sintaxe:

```
no operation
```

Descrição:

Gasta um ciclo de máquina sem fazer absolutamente nada.

Exemplo:

```
no operation    ; gasta um ciclo de máquina sem fazer absolutamente nada
```

1.23 Variáveis

GENERAL

Sintaxe:

general=variável inteira de 16 bits
variável inteira de 16 bits=general
general=general+variável inteira de 16 bits
general=general-variável inteira de 16 bits
general=general*variável inteira de 16 bits(só considera a parte baixa)
general=general/variável inteira de 16 bits(só considera a parte baixa)

Descrição:

É uma variável de acesso geral para todas as variáveis inteiras do sistema. Com esta variável podemos somar o conteúdos das variáveis, dentre outras operações.

Exemplo

```
; Neste exemplo iremos somar o conteúdo das variáveis a% e b%
; e salvar o resultado em a%

a%=1000       ; atribui 1000  a variável a%
b%=2000       ; atribui 2000 a variável b%
general=a%    ; general recebe a%
general=general+b%   ; soma o conteúdo de general mais b%
a%=general           ; a% recebe general
```

TIPOS DE VARIÁVEIS

Descrição:

O sistema possui:
- oito variáveis inteiras (0 á 65535);
- oito variáveis caracter;
- quatro variáveis com uma casa decimal (0 á 6553,5);
- quatro variáveis com duas casas decimais (0 á 655,35);

ASC$

Sintaxe:

Variável Inteira=asc$('a')

Descrição:

Atribui á alguma variável inteira o valor numérico da tabela ASCII do caracter.

Exemplo

```
a%=asc$('A')       ; atribui 65 a variável a%
```

CHR$

Sintaxe:

Variável Caracter=chr$(65)

Descrição:

Atribui á alguma variável caracter um valor até 127 da tabela ASCII.

Exemplo

```
a$=chr$(65)        ; atribui o caracter 'A' a variável a$
```

BIN$

Sintaxe:

Variável Inteira=bin$(valor binário)

Descrição:

Atribui á alguma variável inteira o valor binário do parâmetro valor binário.

Exemplo

```
a%=bin$(0000000011111111)    ; atribui 255 a variável a%
```

HEX$

Sintaxe:

Variável Inteira=hex$(valor hexadecimal)

Descrição:

Atribui á alguma variável inteira um valor em hexadecimal de 16 bits.

Exemplo

```
a%=hex$(FFFF)      ; atribui 65535 a variável a%
```

VARIÁVEL CARACTER

Sintaxe:

a$="caracter"

Descrição:

Atribui algum valor a uma variável caracter.

Exemplo

```
a$='a'       ; a$ recebe 'a'
```

NEW

Sintaxe:

new

Descrição:

Apaga a memória de programa.

CLEAR

Sintaxe:

clear

Descrição:

Apaga a memória de dados.

RANDOM

Sintaxe:

variável=random$

Descrição:

A função random$ gera um número aleatório de 8 bits que pode ser armazenado em uma variável.

Exemplo

```
a%=random$          ; a% recebe um valor randômico da função random$
```

FLAG

Sintaxe:

flag=valor lógico

Descrição:

Permite que sejam armazenados os valores lógicos para posteriores testes do programa. O total de flags são 4.

Exemplo:

```
if switch(1) then
        flag=1
end if
```

```
if switch(2) then
        flag=0
end if

.
.
.

if flag1 then
        disp(1)(Flag verdadeiro)
end if
```

1.24 Constantes Químicas

Z

Sintaxe:

variável inteira= z(elemento químico)

Descrição:

Atribui á alguma variável inteira de 16 bits o número atômico de algum elemento químico da tabela periódica.

Exemplo

```
a%=z(H)     ; a% recebe o número atômico de Hidrogênio
```

MASS

Sintaxe:

variável inteira= mass(elemento químico)

Descrição:

Atribui á alguma variável inteira de 16 bits a massa de algum elemento químico da tabela periódica.

Exemplo

```
a%=mass(H)     ; a% recebe massa de Hidrogênio
```

ELET

Sintaxe:

variável inteira= elet(elemento químico)

Descrição:

Atribui á alguma variável inteira de 16 bits a eletrosfera de algum elemento químico da tabela periódica.

Exemplo

```
a%=elet(H)     ; a% recebe a eletrosfera de Hidrogênio
```

SWAP

Sintaxe:

variável%=swap variável%

Descrição:

Este comando inverte a parte-alta da variável pela baixa e vice-versa.

Exemplo

```
a%=255                  ;carrega a% com 255 decimal
a%=swapf a%             ; inverte a parte-alta pela baixa
```

NOT

Sintaxe:

variável%=not variável%

Descrição:

Este comando inverte o estado de todos os bits da variável.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
a%=not a%               ; lógica not
```

1.25 Controle da Saída

MOV OUT,IN

Sintaxe:

mov out,in

Descrição:

Move o conteúdo da porta de entrada para a saída.

Exemplo:

mov out,in

COMPLEMENT

Sintaxe:

complement(out)

Descrição:

Modifica o estado atual de todas as 6 saídas dispostas na placa PCL 1003.

Exemplo:

complement(out) ;modifica todas as saídas. O que estava ligado agora está desligado e vice-versa.

LOAD OUT

Sintaxe:

load out,(parâmetro)

Descrição:

Carrega o valor especificado em parâmetro.

Exemplo:

load out,63 ;liga todos os relês
load out,1 ;liga somente o relê 1

DEC

Sintaxe:

dec out

Descrição:

Decrementa de uma unidade a saída.

Exemplo:

```
dec out          ;decrementa o valor anterior de 1.
```

INC

Sintaxe:

```
inc out
```

Descrição:

Incrementa de uma unidade a saída.

Exemplo:

```
inc out          ;decrementa o valor anterior de 1.
```

OUT

Sintaxe:

```
OUT(número_da_saída)
```

Descrição:

Verifica o estado atual de alguma saída informada em número_da_saída. Número_da_saída deve ser um número entre 1 e 6.

Exemplo:

```
if out(1) then next else skip
        disp(1)(Rele 1 esta)
        disp(2)(Ligado.)
end if

if out(1) then
        disp(1)(Rele 1 esta)
        disp(2)(Ligado)
else
        disp(1)(Rele 1 esta)
        disp(2)(Desligado)
end if
```

LED

Sintaxe:

```
SET(número_do_led)
CLR(número_do_led)
```

Descrição:

Liga (SET) ou desliga (CLR) o led especificado em número_do_led. Número_do_led pode ser 1 ou 2.

Exemplo:

```
if switch(1) then                    ;se switch(1) acionado...
        set(led1)                    ;autoriza a passagem no acesso.
        clr(led2)
        cld
        disp(1)(Switch 1 on)
        disp(2)(Acesso Autorizado)
        pause(1)
else                                 ;senão....
        set(led2)                    ;nega a passagem no acesso.
        clr(led1)
        cld
        disp(1)(Switch 1 off)
        disp(2)(Acesso Negado)
        pause(1)
end if
```

IF OUT$=

Sintaxe:

if out$=valor then

Descrição:

Este comando verifica se a saída de dados corresponde ao valor especificado em valor.

Exemplo:

```
if out$=1  then                      ;se a saída 1 estiver fechada...
        txdata(100 pulsos!)          ;executa comandos
        clear pulse_counter$
end if
```

Capítulo 2

Controle de Fluxo

2.1 Laço WHILE

DO - WHILE SWITCH

Sintaxe:

```
do
        .
        .
        comandos
        .
        .
while switch(número_da_entrada_digital)
```

Descrição:

Testa switch(número_da_entrada_digital) após executar, ao menos uma vez, o que está após de "do". Enquanto, o switch(número_da_entrada_digital) for verdadeiro, volta a executar a partir de "do", se a condição for falsa passa a executar o que está após o comando while.

Exemplo:

```
do
        txdata(Estrutura posfixada)
        cld
```

```
        set(le1)
        set(all)
        pause(4)
        clr(all)
while switch(1)
```

DO - WHILE NOT SWITCH

Sintaxe:

```
do
        .
        .
        comandos
        .
        .
while switch not (número_da_entrada_digital)
```

Descrição:

Testa switch(número_da_entrada_digital) após executar, ao menos uma vez, o que está após de “do”. Enquanto, o switch(número_da_entrada_digital) for falso, volta a executar a partir de “do”, se a condição for verdadeira passa a executar o que está após o comando while.

Exemplo:

```
do
        txdata(Estrutura posfixada)
        cld
        led(1)(1)
        set(all)
        pause(4)
        clr(all)
while not switch(1)
```

WHILE SWITCH...DO

Sintaxe:

```
while switch(número_da_entrada_digital) do
        .
        .
        comandos
        .
        .
loop
```

Descrição:

Testa primeiro o switch(número_da_entrada_digital). E quando verdadeiro, executa os comandos.

Exemplo:

```
while switch(8) do
        set(1)
        cld
        disp(1)(Rele 1 on)
        delay_seg(6)
        clr(1)
        cld
        disp(1)(Rele 1 off)
        pause(6)
loop
```

WHILE NOT SWITCH...DO

Sintaxe:

```
while switch(número_da_entrada_digital) do
        .
        .
        comandos
        .
        .
loop
```

Descrição:

Testa primeiro o switch(número_da_entrada_digital). E quando falso, executa os comandos.

Exemplo:

```
while not switch(8) do
        set(1)
        cld
        disp(1)(Rele 1 on)
        pause(6)
        clr(1)
        cld
        disp(1)(Rele 1 off)
        pause(6)
loop
```

WHILE ... LOOP

Sintaxe:

```
while variável do

        comandos

loop
```

Descrição:

Enquanto a variável for diferente de 0, comandos serão executados.-

Exemplo

```
 a%=10
while a% do
      txdata(PCL 1002)
      a%=a%-1
 loop
```

2.2 Laço REPEAT

REPEAT...LOOP FOR REPEAT

Sintaxe:

```
repeat(número_de_repetições)
        .
        .
        comandos
        .
        .
loop for repeat
```

Descrição:

Repete os comandos pelo número de vezes especificado em número_de_repetições. O número_de_repetições deve ser maior que zero e menor ou igual a 65535.

Exemplo:

```
repeat(5)
        set(1)
        pause(1)
        clr(1)                                  ;liga e desliga o relé 1 5 vezes.
        pause(1)
        if switch(2) then next else skip    ;se switch(2) acionado
                cld                             ;sai do repeat
                pause(1)
                disp(1)(Saindo do repeat)
                pause(1)
                exit repeat
        end if
        if switch(3) then next else skip
                restore                         ;se switch(3) volta para repeat
        end if
        set(all)
        pause(1)
        clr(all)
        pause(1)
loop for repeat
```

REPEAT...LOOP REPEAT

Sintaxe:

```
repeat(variável)
        comandos
loop repeat
```

Descrição:

O repeat permite que uma variável seja utilizada para o seu funcionamento. Para isso, siga o exemplo abaixo.

Exemplo:

```
a%=100                          ;carrega a% com 100 decimal
repeat(a%)                      ; executa estes comandos 100 vezes
        set(1)
        delay_ms(1500)
        clr(1)
        delay_ms(1500)
loop repeat
```

2.3 Seleção de Casos

SELECT CASE

Sintaxe:

```
select case variável
        case 1
                comandos 1
                .
                .
                break
        case n
                comandos n
                .
                .
                break
        else select
                comandos 3
                .
                break
        end select
```

Descrição:

Faz uma seleção de casos para teste. Caso um seja verdadeiro, os comandos entre o case e o break serão executados. Esta estrutura funciona para variáveis caracter, variáveis inteiras e para a recepção de dados.

Exemplo

```
a%=100
select case variável
        case 100
                txdata(a%)
                break
        case 200
                txdata(b%)
                break
        else select
                txdata(Nenhum e igual)
end select
```

Capítulo 3

Operadores Lógicos e Estruturas Condicionais Lógicas

3.1 Operadores

AND

Sintaxe:

variável%=variável% and constante

Descrição:

Este comando realiza uma lógica and entre a variável e a cconstante.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
a%=a% and 1             ; lógica e com 1
```

OR

Sintaxe:

variável%=variável% or 1

Descrição:

Este comando realiza uma operação or entre à variável e uma contante.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
a%=a% or  1             ; lógica or com 1
```

XOR

Sintaxe:

variável%=variável% xor constante

Descrição:

Este comando realiza uma operação xor entre a variável e a constante.

Exemplo

```
a%=100                  ;carrega a% com 100 decimal
a%=a% xor 1             ; lógica xor com 1
```

3.2 Estruturas Condicionais Lógicas

IF AND WITH...

Sintaxe:

if and with switch(número_entrada_digital 1), switch(número_entrada_digital 2), switch(número_entrada_digital n), then

.
.
.
comandos 1
.
.
.

else

.
.
.
comandos 2
.
.
.

end if

Descrição:

Operação and com as entradas digitais. O parâmetro n, não pode passar de 6 já que este é a última entrada digital. Se a operação and com as entradas for verdadeira, os comandos 1 serão executados, caso contrário, os comandos 2 serão executados.

Exemplo:

```
if and with switch(1),switch(2), then   ;Lógica e com switch(1) e switch(2)
    cld                                 ;caso verdadeiro, executa estes comandos...
    pause(1)
```

```
        disp(1)(Switch 1 fechado)
        txdata(Switch 1 fechado)
        pause(1)
else                                        ;caso falso executa o que está após else
        cld
        pause(1)
        disp(1)(Switch 1 aberto)
        txdata(Switch 1 aberto)
        pause(1)
end if
```

IF OR WITH...

Sintaxe:

```
        if or with switch(número_entrada_digital 1), switch(número_entrada_digital 2),
switch(número_entrada_digital n), then
                        .
                        .
                        .
                        comandos 1
                        .
                        .
                        .

        else
                        .
                        .
                        .
                        comandos 2
                        .
                        .
                        .
        end if
```

Descrição:

Operação or com as entradas digitais. O parâmetro n, não pode passar de 6 já que este é a última entrada digital. Se a operação or com as entradas for verdadeira, os comandos 1 serão executados, caso contrário, os comandos 2 serão executados.

Exemplo:

```
if or with switch(1),switch(2), then        ;Lógica ou com switch(1) e switch(2)
        cld                                 ;se verdadeiro, executa estes comandos...
        pause(1)
        disp(1)(Switch 1 fechado)
        txdata(Switch 1 fechado)
        pause(1)
else                                        ;caso falso executa o que está após else
        cld
        pause(1)
        disp(1)(Switch 1 aberto)
        txdata(Switch 1 aberto)
        pause(1)
```

end if

IF NOT AND WITH...

Sintaxe:

if not and with switch(número_entrada_digital 1), switch(número_entrada_digital 2), switch(número_entrada_digital n), then

.
.
.
comandos 1
.
.
.

else

.
.
.
comandos 2
.
.
.

end if

Descrição:

Operação and com as entradas digitais. O parâmetro n, não pode passar de 6 já que este é a última entrada digital. Se a operação and com as entradas for verdadeira, os comandos 1 serão executados, caso contrário, os comandos 2 serão executados.

Exemplo:

```
if and with switch(1),switch(2), then      ;Lógica e com switch(1) e switch(2)
        cld                                ;caso verdadeiro, executa estes comandos...
        pause(1)
        disp(1)(Switch 1 fechado)
        txdata(Switch 1 fechado)
        pause(1)
else                                       ;caso falso executa o que está após else
        cld
        pause(1)
        disp(1)(Switch 1 aberto)
        txdata(Switch 1 aberto)
        pause(1)
end if
```

IF NOT OR WITH...

Sintaxe:

if or with switch(número_entrada_digital 1), switch(número_entrada_digital 2), switch(número_entrada_digital n), then

```
                    .
                    .
                    .
                    comandos 1
                    .
                    .
                    .

        else
                    .
                    .
                    .
                    comandos 2
                    .
                    .
                    .
        end if
```

Descrição:

Operação or com as entradas digitais. O parâmetro n, não pode passar de 6 já que este é a última entrada digital. Se a operação or com as entradas, for verdadeira, os comandos 1 serão executados, caso contrário, os comandos 2 serão executados.

Exemplo:

```
if or with switch(1),switch(2), then    ;Lógica ou com switch(1) e switch(2)
        cld                             ;se verdadeiro, executa estes comandos...
        pause(1)
        disp(1)(Switch 1 fechado)
        txdata(Switch 1 fechado)
        pause(1)
else                                    ;caso falso executa o que está após else
        cld
        pause(1)
        disp(1)(Switch 1 aberto)
        txdata(Switch 1 aberto)
        pause(1)
end if
```

Capítulo 4

Exemplos de Programação

Neste capítulo iremos estudar alguns exeplos de programação com a placa PLC 1001. Observe que em cada comando, há uma linha de comentários para facilitar o entendimento da mesma.

Exemplo 1 - Oscilando uma saída de relê

Este programa ficará ligando e desligando um relê interminávelmente. O tempo entre o ligar e desligar é de 1 segundo.

```
10    ;       Programa para Oscilar a Saída de um relê
20    ;              Autor: Cerne Tecnologia
30    ;               Data: 28/01/2005
40
50
60    set(1)                  ; liga a saída 1
70    delay_ms(1000)          ; conta um tempo de 1 segundo
80    clr(1)                  ; desliga a saída 1
90    delay_ms(1000)          ; conta um tempo de 1 segundo
100   goto 0                  ; volta para o início do comando
```

Exemplo 2 - Botão e Led

Este programa irá acender o led1 da placa assim que a entrada 1 ficar verdadeira.

```
10    ;              Programa Botão e Led
20    ;              Autor: Cerne Tecnologia
30    ;                 Data: 28/01/2005
40
50    if switch(1) then             ; entrada 1 está ativa?
60          set(led1)               ; sim, então liga led1
70    end if                        ; fim do se
80
90    if not switch(1) then         ; entrada 1 está aberta?
100         clr(led1)               ; sim, então desliga led1
110   end if                        ; fim do se
120
130   goto 0                        ; volta para o início
```

Exemplo 3 - Mensagem no display

Este programa irá apresentar uma mensagem no display LCD da placa

```
10    ;           Programa Mensagem no Display
20    ;              Autor: Cerne Tecnologia
30    ;               Data: 28/01/2005
40
50    disp(1)(PCL 1003)         ; mostra na linha 1 "PCL 1003"
60    disp(2)(Projeto Nacional) ; mostra na linha 2 "Projeto...
```

Exemplo 4 - Mensagem no display e Entradas

Este programa irá apresentar uma mensagem no display LCD caso a entrada 3 seja acionada.

```
10    ; Programa que apresenta mensagem se chave estiver on
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50    if switch(5) then
60          cld
70          disp(1)(Chave 5 on!)
80          delay_ms(1000)
30          cld
90    end if
100
110   goto 0
```

Exemplo 5 - Rotacionando uma mensagem no display

Este exemplo apresenta uma mensagem no display e depois fica rotacionando a mesma.

```
10    ; Programa para rotacionar mensagens no display
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50    label1 for novamente          ; define um label
60
70    disp(1)(PCL 1003)       ; mostra na linha 1 a mensagem
90    novamente
100         rotate display to right
110         delay_ms(500)
120         rotate display to right
130         delay_ms(500)
140         rotate display to right
150         delay_ms(500)
160         rotate display to left
170         delay_ms(500)
180         rotate display to left
190         delay_ms(500)
200         rotate display to left
210         delay_ms(500)
220         goto novamente
```

Exemplo 6 - Transmissão de Dados pelo Canal Serial

Este programa irá transmitir dados pelo canal serial continuamente a uma taxa de 9600 bps.

```
10    ;     Programa de Transmissão de Dados pelo Canal Serial
20    ;                Autor: Cerne Tecnologia
30    ;                 Data: 28/01/2005
40
50          txdata(PCL 1003)        ; transmite "PCL 1003"
60          delay_ms(1000)          ; espera 1000 ms
70          goto 0                  ; volta para 0
```

Exemplo 7 - Recepção de Dados pelo Canal Serial

Este programa irá receber dados do canal serial e dependendo do caracter que receber irá ligar/desligar um dos 2 relês da placa e mostrar uma mensagem no display.

```
10    ; Programa de Recepção de Dados
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50
60    if rxdata$="A" then      ; caracter recebido é o      "A"?
70          set(1)             ; sim, então liga o relê 1
80          disp(1)(Rele 1 On) ; mostra mensagem na linha  1
90    end if                        ; fim do se
100
110   if rxdata$="B" then       ; caracter recebido é o "B"?
120         clr(1)              ; sim, então desliga o relê 1
130         disp(1)(Rele 1 Off) ; mostra mensagem na linha 1
140   end if                        ; fim do se
170
150   if rxdata$="C" then      ; caracter recebido é o "C"?
180         set(2)             ; sim, então liga o relê 2
190         disp(1)(Rele 2 On) ; mostra mensagem na linha 1
200   end if                  ; fim do se
210
220   if rxdata$="D" then     ; caracter recebido é o "D"?
230         clr(2)             ; sim, então desliga o relê 2
240         disp(1)(Rele 2 Off) ; mostra mensagem na linha 1
250   end if                  ; fim do se
260
270         goto 0            ; volta para início do programa
160
```

Exemplo 8 - Utilizando a Data e Hora da Placa

A PCL 1003 possui internamente um relógio de data e hora que permite ao usuário controlar a data e hora do dia. Este relógio de alta precisão já tem tratamento para ano bissexto.

Neste próximo exemplo iremos apresentar a data e hora no display. Quando a hora passar um valor, um dos relês irá acionar para liga por-exemplo uma sirene.

```
10    ;         Programa de Controle de Data e Hora
20    ;              Autor: Cerne Tecnologia
30    ;                Data: 28/01/05
40
50          disp(1)(date$)    ; mostra na linha 1 a data
60          disp(2)(time$)    ; mostra na linha 2 a hora
70
80    if time$>"12_00" then   ; se passou do meio-dia então...
90          set(1)            ; liga o relê 1
100   end if                  ; fim do se
110
120   goto 0                  ; volta para início do programa
```

Exemplo 9 - Utilizando o Cronômetro

Neste exemplo veremos um exemplo para medir um determinado intervalo de tempo com o comando stopwatch$.

```
10    ; Programa para medir um determindao intervalo de tempo
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/05
40
50    if switch(1) then       ; entrada 1 está fechada?
60          clear stopwatch   ; sim, então limpa o    cronômetro
70    end if                  ; fim do se
80
90    if switch(2) then       ; entrada 2 está fechada?
100         begin stopwatch   ; sim, então inicia o timer   2
110   end if
120
130   if switch(3) then       ; entrda 3 está fechada?
140         stop stopwatch    ; sim, então pára o     cronômetro
150   end if                        ; fim do se
160
170   disp(1)(Medicao do Tempo); mostra mensagem no  display
180   disp(2)(stopwatch$)     ; mostra na linha 2 a   medição
190   goto 0                  ; volta para o início
```

Exemplo 10 - Implementando um controle de semáforo

Neste exemplo iremos implementar um controle de semáforo.

```
10 cld                          ; limpa o display
20 clr(all)                     ; desliga todos os relés
disp(1)(Semaforo)               ; escreve no mensagem
disp(2)(Verde)                  ; no display
set(1)                          ; liga o relé 1
txdata(Semaforo Verde Acionado) ; envia para o canal serial
delay_seg(10)                   ; Espera 10 s
clr(1)                          ;desliga o semáforo verde
cld                             ;limpa o display
disp(1)(Semaforo)               ;escreve mensagem
disp(2)(Amarelo)
set(2)                          ;liga o relê 2
txdata(Semaforo Amarelo Acionado)   ; transmite dados
delay_seg(2)                        ;aguarda 2 segundos
clr(2)                              ;desliga o relé 2
cld                                 ;limpam lcd
disp(1)(Semaforo)                   ; mostra mensagem
disp(2)(Vermelho)                   ; no display
set(3)                              ; liga o relê 3
txdata(Semaforo Vermelho Acionado) ; transmite dados
delay_seg(5)                        ; espera 5 segundos
goto 0                              ; volta para o início
```

Exemplo 11 - Medindo Velocidades Externas

Este exemplo demonstra como deve-se proceder para medir uma velocidade externa.

```
10    ; Programa de Medição de Velocidade Externa
```

```
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/05
40
50    space_default=200 ; distância entre os sensores é de 100m
120   cld                ; limpa o display
60    disp(1)(Medindo...) ; mostra mensagem
70    on speed          ; habilita a medição de velocidade
80    cld               ; limpa o lcd
90    disp(1)(speed$); mostra o valor da medição de velocidade
100   delay_ms(2000)          ; aguarda 2 segundos
130   goto 0                  ; volta para o início do programa
```

Exemplo 12 - Contador de Pulsos

Este exemplo irá demonstrar os passos necessários para a contagem de pulsos externos.

```
10    ; Programa para medição de pulsos externos
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50    input6 as pulse_counter$; config. entrda 6 como contador
60    label1 for again         ; define label1
100   begin pulse_counter$
190
120   again
110
70    disp(1)(pulse_counter$) ;apresenta o contador de pulsos
80
170   if switch(1) then             ; entrada 1 é verdadeira?
130         clear pulse_counter$    ; sim, então limpa contador
140   end if                        ; fim do se
150
160   if switch(2) then             ; entrada 2 é verdadeira?
200         stop pulse_counter$   ; sim, então pára a contagem
210   end if                        ; fim do se
220
230   goto again                    ; salta para again
```

Exemplo 13 - Frequêncímetro

Esta placa vêm equipada com um medidor de frequência externo. Vejamos um exemplo deste fantástico módulo.

```
10    ; Programa para medição de frequências externas
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50    input6 as frequecimeter ; config. entrada6 como
                        ; frequencímetro
60    label1 for again        ; define label 1
70
80    again
90
100         disp(1)(freq$)    ; apresenta a frequencia medida
110         goto again        ; salta para again
```

Exemplo 14 - Módulo Contador

Este módulo se parece um pouco com o contador de pulsos, diferindo que neste a recarga de dados é automática além de a informação de contagem encerrada ser informada por meio de flags.

```
10    ; Programa Contador de Pulsos externo com recarga
; automática
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50    input6 as counter
60    counter(1000)
70    label1 for again
80
90    again
100         if counter then
110               disp(1)(Contou 200 pulsos!)
120               delay_ms(2000)
130               cld
140         end if
150         goto again
```

Exemplo 15 - Chamando um Telefone

Este exemplo demonstra como deve ser feito para se fazer uma chamada de algum telefone pela linha telefônica. Neste caso, toda vez que a entrada 4 for pressionada haverá uma chamada de telefone via o relê 1.

```
10    ; Programa para demonstrar o comando para chamar um
; telefone
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50    if switch(4) then            ; chave 4 está pressionada?
60          disp(1)(Chamando Telefone)
100         callphone(88316621)    ; sim, então chama telefone
120         cld
70    end if
80
90    goto 0
```

Exemplo 16 - Implementando um Jogo de Palavras

A PCL1003 vem equipada com um jogo educativo de palavras. Esta pode ser acessada através do comando game. Caso o jogador tenha ganho a partida, o comando game será positivo, caso contrário negativo.

```
10    ; Programa para demonstrar a utilização do jogo de
; palavras
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50    disp(1)(Boa Partida!)        ; apresenta mensagem
60    delay_ms(2000)               ; espera 2 segundos
```

```
70    game                          ; chama o jogo
80
100   if not game then              ; se jogador perdeu...
110         cld                     ; limpa o display
120         disp(1)(Voce perdeu...) ; apresenta mensagem
130         delay_ms(2000)          ; espera 2 segundos
140   end if                        ; fim do se
150
160   if game then                  ; se jogador ganhou...
170         cld                     ; limpa o display
180         disp(1)(Voce Ganhou!)   ; apresenta mensagem
190         delay_ms(3000)          ; espera 2 segundos
200   end if                        ; fim do se
210
240   cld                           ; limpa o lcd
250   disp(1)(Deseja Continuar?)    ; apresenta mensagem
260
220   do                            ; faça
230         no operation            ; sem operação
270   while not switch(4)           ; enquanto a chave 4
                                ; estiver aberta
280   goto 0                        ; salta para o endereço 0
```

Exemplo 17 - Atribuindo valores e constantes as variáveis

Neste exemplo veremos como deve se proceder para atribuir um valor ou uma constantes a algumas variáveis dos sitema como as inteiras, as de duas casas decimais e uma casa decimal.

```
10    ; Programa de atribuição de valores as variáveis
20    ; Autor: Cerne Tecnologia
30    ; Data: 29/01/2005
40
50    a@=pi     ; atribui a constante pi a variável
60    b@=e      ; atribui a constante e(exponencial) a variável
70    c@=g      ; atribui a constante g(gravidade) a variável    80

90    disp(1)(Variavel a@ ) ; apresenta mensagem
100   disp(2)(a@)           ; apresenta conteúdo da variável
110   delay_ms(2000)        ; espera tempo
120
130   disp(1)(Variavel b@ ) ; apresenta mensagem
140   disp(2)(b@)           ; apresenta conteúdo da variável
150   delay_ms(2000)        ; espera tempo
160
170   disp(1)(Variável c@ )  ; apresenta mensagem
190   disp(2)(c@)            ; apresenta conteúdo da variável
```

Exemplo 18 - Calculando a hipotenusa de um triângulo

Neste exemplo iremos utilizar um pouco as funções matemáticas quea PCL nos oferece. Começemos pelo cálculo da hipotenusa.

```
10    ; Programa para Calcular a hipotenusa de ângulo
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50    ; Atenção!!!
60    ; a hipotenusa é o a% enquanto os catetos
```

```
70    ; são o b% e o c%.
80
90    b%=6              ; carrega um cateto
100   c%=10             ; carrega outro cateto
110   a%=hipot(b%,c%)   ; calcula a hipotenusa
120   disp(1)(a%)       ; apresenta o valor calculado
```

Exemplo 19 - Calculando o Perímetro e a Área de um Círculo

Vamos agora demonstrar as diretrizes para calcular o Perímetro e a Área de umcírculo.

```
10    ; Programa para Cálculo de Perímetro e Área de um Círculo
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50    ; Atenção!!!
60    ; O raio deve estar previamente
70    ; carregado em b%!
80
90    b%=3                    ; raio igual á 3
100   a@=circunf(b%)          ; calcula a circunferência
110   disp(1)(a@)             ; apresenta resultado
120   a@=area_circunf(b%)     ; calcula a área
130   disp(2)(a@)             ; apresenta resultado
```

Exemplo 20 - Calculando o Fatorial de um Número

Neste tópico iremos calcular o fatorial de um número. O máximo suportado pela placa é até o fatorial de 8.

```
10    ; Programa para Cálculo de Fatorial
20    ; Autor: Cerne Tecnologia
30    ; Data: 28/01/2005
40
50    a%=0!
60    disp(1)(a%)
70    delay_ms(1000)
80
90    b%=1!
100   disp(1)(b%)
110   delay_ms(1000)
120
130   c%=4!
140   disp(1)(c%)
150   delay_ms(1000)
```

Exemplo 21 - Soma, Subtração, Produto e Divisão com constantes

Neste exemplo iremos somar valores constante as variáveis.

```
10    ; Programa que demonstra as quatro operações básicas
20    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
50    a%=1000                 ; atribui 1000 á variável
60    disp(1)(Antes da soma)  ; apresenta mensagem
70    disp(2)(a%)             ; apresenta variável
```

```
80    delay_ms(2000)          ; espera um tempo
90
100   a%=a%+1500              ; soma 1500 a variável
110   disp(1)(Depois da soma) ; apresenta mensagem
120   disp(2)(a%)             ; apresenta variável
130   delay_ms(2000)          ; espera um tempo
140
150   a%=a%-200               ; subtrai 200 da variável
170   disp(1)(Apos Subtracao) ; apresenta mensagem
180   disp(2)(a%)             ; apresenta variável
190   delay_ms(2000)          ; espera tempo
200
210   a%=a%*4                 ; multiplica o último valor por 4
220   disp(1)(Apos Multipli)  ; apresenta mensagem
230   disp(2)(a%)             ; apresenta variável
240   delay_ms(2000)          ; espera tempo
260
250   a%=a%/2                 ; divide a% por 2
270   disp(1)(Apos Divisao)   ; apresenta mensagem
280   disp(2)(a%)             ; apresenta variável
```

Exemplo 22 - Soma, Subtração, Produto e Divisão com Variáveis

Agora iremos fazer as mesmas operações acima, porém entre as vairáveis.

```
10; Programa que demonstras as quatro operações com Variáveis
20; Autor: Cerne Tecnologia
30; Data: 29/01/2005
40
50    a%=1000                 ; atribui a variável a% 1000
60    general=a%        ; a variável geral é igual a 1000
70    b%=200                  ; atribui 200 a variável b%
80    general=general+b%      ; soma general mais b%
90    a%=general        ; a% recebe general
100   disp(1)(a%)       ; o resultado é 1200
110   delay_ms(2000)          ; aguarda 2 segundos
120
130   c%=500                  ; atribui 500 a variável c%
140   general=a%        ; general recebe a%
150   general=general-c%      ; subtrai general menos c%
160   a%=general        ; a% recebe general
170   disp(1)(a%)       ; o resultado é 700
180   delay_ms(2000)          ; aguarda 2 segundos
190
200   d%=4              ; atribui 4 a variável d%
210   general=a%        ; general recebe a%
220   general=general * d%    ; multiplica general * d%
230   a%=general        ; a% recebe general
240   disp(1)(a%)       ; o resultado é 2800
250   delay_ms(2000)          ; aguarda 2 segundos
260
270   e%=10             ; atribui 10 a variável
280   general=a%        ; general recebe a%
290   general=general / e%    ; divide general por e%
300   a%=general        ; a% recebe general
310   disp(1)(a%)       ; o resultado é 280
```

Exemplo 23 - Calculando a Porcetagem

Este exemplo permitirá que se calcule a porcentagem de uma variável mais um valor dedinido por uma constante.

```
10    ; Programa para Calculo de Porcentagem
20    ; Autor: Cerne Tecnologia
30    ; Data: 29/01/2005
40
50    a%=1000           ; atribui 1000 a a%
60    a%=a%+40%   ; a% é igual a 1400
70    disp(1)(a%) ; apresenta o valor do cálculo
80    delay_ms(1000)    ; aguarda tempo
90    b%=10000    ; atribui 10000 a a%
100   b%=b%+70%   ; a% é igual a 17000
110   disp(1)(b%) ; apresenta o valor do cálculo
```

Exemplo 24 - Tirando a Raiz de um número

Este exemplo irá calcular a raiz de um número e apresentar no display após o cálculo.

```
10    ; Programa para retirar a raiz quadrada de um número
20    ; Autor: Cerne Tecnologia
30    ; Data: 29/01/2005
40
50    a%=100            ; atribui 100 a a%
60    a%=square a%      ; calcula a raiz e salva em a%
70    disp(1)(a%) ; apresenta o resultado
```

Exemplo 25 - Calculando a Potência de um Número

Neste exemplo iremos calcular a potência de um número mediante ao valor atribuído á variável e elevaremos o mesmo á uma constante.

```
10    ; Programa para Calculo de potência
20    ; Autor: Cerne Tecnologia
30    ; Data: 29/01/2005
40
50    a%=2        ; atribui 2 a variável
60    a%=a% ^ 20  ; eleva o número a 10
70    disp(1)(a%) ; o resultado é 1024
```

Exemplo 26 - Testes condicionais com variáveis

Neste exemplo iremos fazer testes com os valores armazenados nas variáveis.

```
10    ; Programa de Teste Condicional com Variáveis
20    ; Autor: Cerne Tecnologia
30    ; Data: 29/01/2005
40
50    f%=101            ; atribui 100 a variável
60
70    if f%<100 then    ; se for menor que 100...
80          disp(1)(E menor que 100) ; apresenta mensagem
90          delay_ms(1000)           ; espera 1 segundo
100   end if                         ; fim do se
110
120   if f%>100 then    ; se for maior que 100...
130         disp(1)(E maior que 100) ; apresenta mensagem
```

```
140         delay_ms(1000)           ; espera 1 segundo
150   end if                         ; fim do se
160
170   if f%=100 then    ; se for igual a 100...
180         disp(1)(E igual a 100)   ; apresenta mensagem
190         delay_ms(1000)           ; espera 1 segundo
200   end if                         ; fim do se
```

Exemplo 27 - Seleção de Casos com Variáveis

Neste exemplo veremos como devemos proceder para fazermos uma seleção de casos com as variáveis.

```
10    ; Programa para demonstrar o uso do Select Case
20    ; Autor: Cerne Tecnologia
30    ; Data: 29/01/2005
40
50    d%=5000                  ; atribui 10000 a d%
60
70    select case d%           ; seleção de casos com d%
80
90          case 1000    ; caso seja 1000...
100
110               disp(1)(E 1000)  ; apresenta mensagem
120               delay_ms(1000)     ; aguarda tempo
130               break         ; fim do case
140
150         case 5000          ; caso seja 5000...
160
170               disp(1)(E 5000)  ; apresenta mensagem
180               delay_ms(1000)     ; aguarda tempo
190               break         ; fim do case
200
210         case 10000         ; caso seja 10000...
220
230               disp(1)(E 10000) ; apresenta mensagem
240               delay_ms(1000)     ; aguarda tempo
250               break         ; fim do case
260
270         else select        ; caso não seja nenhum...
280
290               disp(1)(Nao encontrado) ;apresenta mensagem
300               break
310
320   end select
330
```

Exemplo 28 - Operações Lógicas com Variáveis

Neste exemplo veremos como aplicar as operações lógicas nas variáveis inteiras do sistema.

```
10    ; Programa para demonstrar as operações lógicas
20    ; com as variáveis do sistema
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
```

```
50
60    a%=65535          ; atribui valor a variável
70    disp(1)(a%)       ; apresenta o conteúdo da variável
80    delay_ms(1000)    ; espera 2 segundos
90    a%=a% and 255     ; lógica and com constante
100   disp(1)(a%)       ; apresenta conteúdo da variável
110   delay_ms(1000)    ; espera 2 segundos
140
150   a%=1              ; atribui constante a variável
160   a%=a% or 2        ; lógica or com constante
170   disp(1)(a%)       ; apresenta conteúdo da variável
180   delay_ms(1000)    ; aguarda tempo
190
200   a%=255            ; atribui constante a variável
210   a%=swap a%        ; inverte parte baixa pela alta
220   disp(1)(a%)       ; apresenta conteúdo da variável
230   delay_ms(1000)    ; aguarda tempo
240
250   a%=1              ; atribui constante a variável
260   a%=not a%         ; inverte o estado dos bits
270   disp(1)(a%)       ; apresenta conteúdo da variável
280   delay_ms(1000)    ; aguarda tempo
290
300   a%=100            ; atribui constante a variável
310   a%=a% xor 1       ; lógica xor com 1
320   disp(1)(a%)       ; apresenta conteúdo da variável
330   delay_ms(1000)    ; aguarda tempo
```

Exemplo 29 - Atribuição nas Variáveis Caracter

Neste tópico vamos ver como é feita a atribuição de valores nas variáveis caracter.

```
10    ; Programa de atribuição de
20    ; Valores a variável caracter
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    a$='p'                ; atribui caracter a a$
70    b$='c'                ; atribui caracter a b$
80    c$='l'                ; atribui caracter a c$
110   disp(1)(a$)           ; apresenta a$
120   delay_ms(1000)        ; espera 1 segundo
130   disp(1)(b$)           ; apresenta b$
140   delay_ms(1000)        ; espera 1 segundo
150   disp(1)(c$)           ; apresenta c$
160   delay_ms(1000)        ; espera 1 segundo
```

Exemplo 30 - Transmissão de Variáveis Caracter pelo Canal Serial

Neste exemplo iremos abordar a transmissão das variáveis caracter pelo canal serial.

```
10    ; Programa para demonstrar a transmissão
20    ; de variáveis caracter pelo canal serial.
30
40    a$='b'      ; carrega variável caracter
50    txdata(a$)  ; transmite a mesma pelo canal serial
60
70    b$='A'      ; carrega variável caracter
80    txdata(b$)  ; transmite a mesma pelo canal serial
```

Exemplo 31 - Testes condicionais com as Variáveis Caracter

Neste exemplo vamos abordar o teste condicional com a variável caracter.

```
10    ; Programa para demonstrar
20    ; testes condicionais com as variáveis caracter
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    e$='v'
70
80    if e$='m' then
90          disp(1)(Variavel igual)
100         disp(2)(a 'm'           )
110         delay_ms(2000)
120   end if
```

Exemplo 32 - Seleção de casos com as Variáveis Caracter

Neste exemplo vamos abordar a seleção de casos com a variável caracter.

```
10    ; Programa para demonstrar o uso da seleção
20    ; de casos com a variável caracter
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    a$='z'                  ; atribui caracter
70
80    select case a$          ; seleção de casos de a$
90          case 'm'    ; caso 'm'
100               set(1)      ; liga relê 1
110               break ; fim do case
120         case 'o'    ; caso 'o'
130               set(2)      ; liga o relê 2
140               break ; fim do case
150         case 'z'    ; caso 'z'
160               set(3)      ; liga o relê 3
170               break ; fim do case
180   end select        ; fim de select
```

Exemplo 33 - Utilizando as funções CHR$ e ASC$

Neste exemplo vamos utilizar as funções CHR$ e ASC$ para atribuir valores as variáveis que não se encontram no teclado.

```
10    ; Programa de demonstração da
20    ; utilização das funções CHR$ e ASC$
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    a%=asc$('A')      ; atribui o valor ASCII do 'A' em a%
70    disp(1)(a%) ; apresenta no display
80
90    a$=chr$(48) ; atribui o '0' a a$
100   disp(2)(a$) ; apresenta no display
```

Exemplo 34 -Escrevendo na memória não volátel

O sistema possui 256 bytes de de memória de dados não voláteis, ou seja, após o sistema ter sido desligado, os dados não serão perdidos. Neste primeiro exemplo com estas memórias, iremos escrever alguns caracteres em posições específicas da memória.

```
10    ; Programa exemplo para escrita
20    ; de dados na memória não volátel
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    addr_memory=100         ; aponta para o endereço 100
70    write_memory(a)         ; escreve o caracter neste
80                      ; endereço
```

Exemplo 35 - Lendo a memória não volátel

Agora iremos ler na mesma posição em que foi feita a escrita e compararmos se os valores estão corretos.

```
10    ; Programa exemplo para
20    ; demonstrar a leitura da memória
100   ; Autor: Cerne Tecnologia
110   ; Data: 29/01/2005
30
40    addr_memory=100         ; aponta para o endereço
50
60    if read_memory$='a' then ; caracter lido é o 'a'?
70          disp(1)(Foi escrito o A!) ; sim, então apresenta
                                ; mensagem
90          txdata(read_memory$)      ; envia dado lido
80    end if                          ; fim do se
```

Exemplo 36 - Comparador

Iremos utilizar nesta parte do manual, o comparador de tensão existente na placa. Para isso, devemos informar ao sistema qual será a tensão de referência utilizando o comando Vref. A tensão de referência deve ser um valor entre 0 e 5 VCC.

```
10    ; Programa exemplo para demonstrar
20    ; a utilização do comparador de tensão
30
40    Vref=2.5    ; define a tensão de referência em 2.2V
50    label1 for again ; define um label
70
80    again
90       if comparator then ; se a tensão de entrada é maior...
60          disp(1)(Tensao de Entrada) ; apresenta mensagem
100         disp(2)(E maior!)    ; apresenta mensagem
190      set(1)                  ; liga o relê 1
110   end if                     ; fim do se
120
130   if not comparator then ;se a tensão de entrada é menor...
140         disp(1)(Tensao de Entrada) ; apresenta mensagem
150         disp(2)(E menor!)    ; apresenta mensagem
200         clr(1)                    ; desliga o relê 1
160   end if                     ; fim do se
170
180      goto again              ; salta para again
```

Exemplo 37 - Leitura de Tensão externa

Agora iremos ler uma tensão externa e verificar se ela ultrapassou um valor. Caso tenha ultrapassado, iremos transmitir uma mensagem pelo canal serial, ligar um relê e mostrar uma mensagem.

```
10    ; Programa de Leitura de Tensão externa
20    ; Autor: Cerne Tecnologia
30    ; Data: 29/01/2005
40
50    if ad_read$>3.00 then   ; se a tensão de entrada for
                              ; maior que 3 Volts...
60
70          set(1)            ; liga o relê 1
80          disp(1)(Maior que 3 V) ; mostra mensagem
90          txdata(Passou 3 Volts) ; transmite dados
100
110   end if                  ; fim do se
120
130   goto 0                  ; volta para o início
```

Exemplo 38 - Comparador de Janela

Este fantástico comando permite que verifiquemos se a tensão de entrada está entre dois limites. Caso estyeja, o teste condcional será verdadeiro e os comandos subsequentesserão executados.

```
10    ; Programa exemplo para
20    ; demonstrar a utilização do
30    ; comparador de janela
40
50    if ad_read$>2.00 and ad_read$<4.00 then  ; se a medição
60                                             ; estiver nessa faixa...
70          toggle(out(1))                     ; inverte o estado do relê1
80          txdata(Esta na faixa)              ; transmite dados
90
100   end if                                   ; fim do se
110
120   goto 0                                   ; volta para o início
```

Exemplo 39 - Controle por PWM

A PCL 1003 vem com um módulo PWM para controle externo. O PWM pode ser usado por-exemplo para controlar a velocidade de um motor. O duty-cycle é ajustável por software.

```
10    ; Programa para controle do PWM
20    ; Autor: Cerne Tecnologia
30    ; Data: 29/01/2005
40
50    if switch(1) then  ; se chave 1 pressionada...
60          pwm(0)         ; duty-cycle é 0
70    end if               ; fim do se
80
90    if switch(2) then  ; se chave 2 pressionada...
100         pwm(25)        ; duty-cycle é 25%
110   end if               ; fim do se
```

```
120
130   if switch(3) then  ; se chave 3 pressionada...
140         pwm(50)        ; duty-cycle é 50%
150   end if               ; fim do se
160
170   if switch(4) then  ; se chave 4 pressionada...
180         pwm(75)        ; duty-cycle é 75%
190   end if               ; fim do se
200
210   if switch(5) then  ; se chave 5 pressionada...
220         pwm(100)   ; duty-cycle é 100%
230   end if               ; fim do se
240
250   goto 0               ; volta para o início
```

Exemplo 40 - Saída de Tensão

A PCL ainda tem uma saída de tensão de 0 á 5V ajustável por software. Vejamos um exemplo deste módulo.

```
10    ; Programa exemplo para demonstrar
20    ; a saída de tensão externa
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    if rxdata$='A' then     ; se recebeu o 'A'...
70          out_voltage(0)    ; saída de 0V
80    end if                  ; fim do se
90
100   if rxdata$='B' then       ; se recebeu o 'B'...
110         out_voltage(1.21) ; saída de 1.21 V
120   end if                    ; fim do se
130
140   if rxdata$='C' then       ; se recebeu o 'C'...
150         out_voltage(2.45) ; saída de 2.45 V
160   end if                    ; fim do se
170
180   if rxdata$='D' then       ; se recebeu o 'D'...
190         out_voltage(4.10) ; saída de 4.1 V
210   end if
220
230   goto 0
```

Exemplo 41 - Desligando a placa automaticamente

O Comando Shutdown permite que a placa seja desligada passado algum tempo fornceido em segundos. Vejamos uma aplicação.

```
10    ; Programa para desligar a placa
20    ; após um determinado horário
30
40    disp(1)(time$)          ; apresenta  a hora
50
60    if time$>"22_00" then   ; se passou das 22:00...
70          shutdown on ; programa para desligar em 1 minuto
80    end if                  ; fim do se
90
100   goto 0                  ; volta para início
```

Exemplo 42 - Apresentando o mês corrente

A PCL 1003 possui internamente um calendário do mês vigente e este pode ser apresentado no display.

```
10    ; Programa para apresentar
20    ; o mês corrente
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    disp(1)(time$)          ; apresenta a hora
70    disp(2)(date$)          ; apresenta a data
80    disp(2)(month$)         ; apresenta o mês
90    goto 0                  ; volta para o início
```

Exemplo 43 - Utilizando o Terminal

Neste exemplo aprenderemos a utilizar o comando Terminal. Este fantástico comando permite que os dados digitados no teclado o seu pc sejam enviados ao display da PCL.

```
10    ; Programa para demonstrar a
20    ; utilização do comando terminal
70    ; Autor: Cerne Tecnologia
80    ; Data: 29/01/2005
30
40    term 20     ; fica no comando term por 10 segundos
50    cld   ; limpa o display
60    disp(1)(Fim do Processo) ; apresenta "Fim do Processo"
```

Exemplo 44 - Chamando Rotinas

Agora iremos chamar aprender a chamar rotinas com a AutoEasy. As rotinas diminuem o espço da memória de programa, pois podem ser utilizadas em comum por vários processos no programa.

```
10    ; Programa exemplo para demonstrar
20    ; a utilização das chamadas de programa
40    ; Autor: Cerne Tecnologia
30    ; Data: 29/01/2005
50
60    label1 for limpa_display    ; define label1
70    label2 for mostra_tela      ; define label2
80    label3 for transmite_dados  ; define label3
90
100   call mostra_tela      ; chama rotina
110   call transmite_dados        ; chama rotina
120   call limpa_display          ; chama rotina
130   delay_ms(1000)              ; espera 1 segundo
140   call mostra_tela      ; chama rotina
150   call transmite_dados        ; chama rotina
160   call limpa_display          ; chama rotina
170   end                   ; fim
180
190   mostra_tela
```

```
200         disp(1)(Teste do Call)
210         return                ; retorno de sub-rotina
220
230   transmite_dados
240         txdata(Teste do Call)
250         return                ; retorno de sub-rotina
260
270   limpa_display
280         delay_ms(1000)
290         cld
320         return                ; retorno de sub-rotina
```

Exemplo 45 - Apresentando o conteúdo de uma variável em outras bases

A PCL pode apresentar e transmitir o conteúdo de uma variável em: binário, decimal, octal e hexadecimal. Vamos acompanhar nesnte exemplo como isto é possível

```
10    ; Programa para apresentar
20    ; o conteúdo da variável em outras bases
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    a%=100            ; carrega variável com 100
70    disp(1)(Valor em decimal) ; apresenta mensagem
80    disp(2)(a%) ; apresenta conteúdo em decimal
90    delay_ms(2000)    ; aguarda 2 segundos
100   cld         ; limpa o display
110   disp(1)(Valor em binario) ; apresenta mensagem
120   disp(2)(bin$(a%))  ; apresenta conteúdo em binário
130   delay_ms(2000)       ; aguarda 2 segundos
140   cld            ; limpa o display
150   disp(1)(Valor em Hexadecimal) ; apresenta mensagem
160   disp(2)(hex$(a%)) ; apresenta conteúdo em hexa
170   delay_ms(2000)      ; aguarda 2 segundos
180   cld           ; limpa o display
190   disp(1)(Valor em octal)  ; apresenta mensagem
200   disp(2)(oct$(a%))  ; apresenta conteúdo em octal
210   delay_ms(2000)           ; aguarda 2 segundos
```

Exemplo 46 - Atribuindo um número atômico, massa e eletrosfera á uma variável inteira

Neste exemplo iremos atribuir as variáveis a%, b% e c% o número atômico, a massa e a eletrosfera do elmento químico Hidrogênio.

```
10    ; Programa de atribuição de número atômico,
20    ; massa e eletrosfera a uma variável
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    a%=z(H)           ; atribui a a% o Z de H
70    b%=mass(H)  ; atribui a massa de H
80    c%=elet(H)  ; atribui a eletrosfera de H
90    cld         ; limpa o display
100   disp(1)(    Hidrogenio   ) ; apresenta mensagem
110   delay_ms(2000)             ; espera 2 segundos
120
180   cld                ; limpa o display
130   disp(1)(Numero Atomico ) ; mostra mensagem
140   disp(2)(a%)        ; mostra conteúdo de a%
150   delay_ms(2000)           ; espera 2 segundos
```

```
160
170   cld                ; limpa o display
190   disp(1)(Massa)           ; mostra mensagem
200   disp(2)(b%)        ; mostra conteúdo de b%
210   delay_ms(2000)           ; espera 2 segundos
220
230   cld                ; limpa o display
240   disp(1)(Eletrosfera)     ; mostra mensagem
250   disp(2)(c%)              ; mostra conteúdo de c%
```

Exemplo 47 - Medidor de RPM Externo

Agora iremos apresentar um exemplo bem interessante, um medidor de RPM. Vamos utilizar o comando RPM para esta função.

```
10    ; Programa de Mediçaõ de
20    ; RPM Externo
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    aletas_default=10       ; define número de aletas
70    label1 for again  ; define label
80    input6 as rpm           ; define a entrada 6 como RPM
90
100   again
110         disp(1)(RPM$)     ; apresenta o RPM no display
120         goto again
```

Exemplo 48 - Atribuindo Seno, Cosseno, Tangente, Cossecante, Secante e Cotangente as Variáveis

Agora iremos atribuir os valores supracitados as variáveis. Observem.

```
10    ; Programa de Atribuição de
20    ; Seno, Cosseno, Tangente, Cossecante,
30    ; Secante e Cotangente as variáveis
40    ; Autor: Cerne Tecnologia
50    ; data: 29/01/2005
60
70    a@=sin(45)        ; atribui seno de 45
80    b@=cos(30)        ; atribui coseno de 30
90    c@=tan(10)        ; atribui tangente de 10
100   d@=sec(70)        ; atribui secante de 70
110   a@=cosec(80)            ; atribui cosecante de 80
120   a@=cotan(15)            ; atribui cotangente de 15
```

Exemplo 49 - Atribuindo Logarítmo, Logarítimo Neperiano e Exponencial as Variáveis

Agora iremos atribuir os valores supracitados as variáveis. Observem.

```
10    ; Programa de atribuição de
20    ; Exponencial, Logaritmo Neperiano e na Base 10
30
40    a@=exp(2)
50    b@=ln(2)
60    c@=log(10)
```

Exemplo 50 - Trabalhando com números randômicos

A PCL 1003 vem com um gerador de números randômicos. Vejamos um exemplo.

```
10    ; Programa para ler números
20    ; randômicos
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    a%=random$        ; lê o valor randômico
70    disp(1)(a%)       ; apresenta no display
80    delay_ms(300)     ; espera 300 ms
90    goto 0            ; volta para o início
```

Exemplo 51 - Voltímetro

A PCL 1003 tem um módulo de medição de tensão externa. Vamos ver como utiliza-lo.

```
10    ; Programa exemplo para demonstrar a
20    ; utilização do módulo voltímetro
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
110   disp(1)(ad_read$(voltage)) ; apresenta valor medido
120   goto 0
```

Exemplo 52 - Medição de Velocidade Angular

A PCL 1003 também pode medir velocidades angulares externas. Para isso, a entrada 6 deve estar configurada para o modo frequencímetro e a entrada de pulsos ligada neste. Vejamos o exemplo.

```
10    ; Programa para medição de
20    ; velocidade angular
30    ; Autor: Cerne Tecnologia
40    ; Data: 29/01/2005
50
60    input6 as frequecimeter ; entrada 6 como frequencímetro
70    label1 for again  ; define label1
80    again
90          a@=ang_speed$     ; variável recebe and_speed$
100         disp(1)(a@) ; apresenta o conteúdo da variável
110         goto again  ; salta para again
```