[Q]: Отдача таймслайсов. Паскаль с ассемблером.

[A]: Vadim Rumyantsev (2:5030/301)

Более новая версия с пофиксенным зависанием при редком стечении обстоятельств в полночь в ДОСе :) И ещё чуть-чуть список операционных систем расширен.

———————————————————[Cut Here]———————————————————

{ Written by Vadim Rumyantsev, 2:5030/301. } { Generic DELAY unit – release timeslices } { if under OS/2 2.0, Windows 3.0, DesqView, } { DoubleDOS and probably DOS 5.0 (?!), else } { do nothing. } { It is assumed that program receives time } { quantums every day... so, don't run this } { unit on slow systems! } { Virtual Pascal compatible now! } { Delphi 2.0 compatible now. } { You may use this **without restrictions** }

UNIT USLDelay;

{$I-}

INTERFACE

type

```
OS_Type = (OS_MSDOS, OS_DOUBLEDOS, OS_TOPVIEW, OS_DESQVIEW,
           OS_OS2_1, OS_OS2_2, OS_WINDOWS, OS_WIN32, OS_MACOS);
```

const

```
AccessDenied : set of byte = [5 {$IFNDEF DOS} , 32 {$ENDIF} ];
```

var

```
Running_OS_Name : string;
```

{$IFDEF OS2} const

```
Running_OS = OS_OS2_2;
```

{$ENDIF} {$IFDEF WIN32} const

```
Running_OS = OS_WIN32;
```

{$ENDIF} {$IFDEF MSDOS} var

```
Running_OS : OS_Type;
```

{$ENDIF} {$IFDEF DPMI} var

```
Running_OS : OS_Type;
```

{$ENDIF}

procedure Delay (n : longint);

IMPLEMENTATION

{$IFDEF OS2}

uses {$IFDEF VIRTUALPASCAL} Os2base {$ELSE} Doscalls {$ENDIF};

var

```
Buf : packed array [5..12] of longint;
Sgn : string;
f : file;
fp : longint;
sp : longint;
p1, p2 : integer;
```

{$ENDIF}

{$IFDEF WIN32}

uses SysUtils, Windows;

const

```
UnknownPlatform = 'Win32';
UnknownWin95    = 'Win9x';
```

var

```
VersionInfo : TOsVersionInfoA;
vb : string [10];
```

{$ENDIF}

{$IFDEF MSDOS}

uses Dos;

{ Define Seg0040 for backward compatibility with TP 4.0 .. TP 6.0 }

const

```
Seg0040 = $0040;
```

var

```
r : Registers;
dosvh, dosvl : byte;
osvh, osvl : byte;
```

```
vendor : string [3];
```

```
{$DEFINE DOSMODE}
```

{$ENDIF}

{$IFDEF DPMI}

uses Dos;

{ Define Seg0040 for backward compatibility with TP 4.0 .. TP 6.0 }

var

```
r : Registers;
dosvh, dosvl : byte;
osvh, osvl : byte;
vendor : string [3];
```

```
{$DEFINE DOSMODE}
```

{$ENDIF}

function Version (vh, vl : longint) : string;

var

```
vhs, vls : string [2];
```

begin

```
str (vh, vhs);
str (vl, vls);
if length (vls) = 1 then
  vls := '0' + vls;
if vls [length (vls)] = '0' then
  dec (vls [0]);
Version := vhs + '.' + vls
```

end;

{$IFDEF OS2}

procedure Delay;

begin

```
if DosSleep (n) <> 0 then;
```

end;

BEGIN

```
Running_OS_Name := 'OS/2';
```

```
if DosQuerySysInfo (5, 12, Buf, sizeof (Buf)) = 0 then begin
```

```
  FileMode := open_access_ReadOnly + open_share_DenyNone;
  assign (f, chr (64 + Buf [5]) + ':\OS2KRNL');
  reset (f, 1);
  seek (f, $3C);
  blockread (f, fp, 4);
  seek (f, fp+$88);
  blockread (f, fp, 4);
  seek (f, fp);
  blockread (f, Sgn [0], 1);
  blockread (f, Sgn [1], length (Sgn));
  p1 := pos ('@#', Sgn);
  p2 := pos ('#@', Sgn);
  if (IoResult = 0) and
     (p1 <> 0) and (p2 <> 0) and (p2 > (p1+2))
  then begin
    Sgn := copy (Sgn, p1+2, p2-p1-2);
    p1 := pos (':', Sgn);
    if p1 <> 0 then
      Sgn := copy (Sgn, p1+1, 255);
    Running_OS_Name := Running_OS_Name + ' Revision ' + Sgn
  end
  else begin
    Buf [11] := Buf [11] div 10;
    if (Buf [11] = 2) and (Buf [12] >= 30) and (Buf [12] < 90) then begin
      Buf [11] := Buf [12] div 10;
      Buf [12] := Buf [12] mod 10
    end;
    Running_OS_Name := Running_OS_Name + ' ' + Version (Buf [11], Buf [12])
  end;
```

```
  close (f);
  if IoResult <> 0 then;
```

```
end;
```

{$ENDIF}

{$IFDEF WIN32}

procedure Delay;

begin

```
Sleep (n);
```

end;

BEGIN

```
with VersionInfo do begin
  dwOsVersionInfoSize := sizeof (VersionInfo);
  if not GetVersionExA (VersionInfo) then
    Running_OS_Name := UnknownPlatform
  else begin
    str (dwBuildNumber and $FFFF, vb);
    case dwPlatformId of
      VER_PLATFORM_WIN32_WINDOWS:
        if (dwMajorVersion = 4) and (dwMinorVersion = 0) then
          Running_OS_Name := 'Windows 95'
        else if (dwMajorVersion = 4) and (dwMinorVersion = 10) then
          Running_OS_Name := 'Windows 98'
        else if (dwMajorVersion = 4) and (dwMinorVersion = 90) then
          Running_OS_Name := 'Windows Me'
        else
          Running_OS_Name := UnknownWin95;
      VER_PLATFORM_WIN32_NT:
        if (dwMajorVersion = 5) then
          Running_OS_Name := 'Windows 2000'
        else
          Running_OS_Name := 'Windows NT'
      else
        Running_OS_Name := UnknownPlatform
    end;
    Running_OS_Name := Running_OS_Name + ' ' +
                       Version (dwMajorVersion, dwMinorVersion) +  '/' + vb;
    if szCsdVersion [0] <> #0 then
      Running_OS_Name := Running_OS_Name + ' ' + StrPas (@szCsdVersion [0])
  end
end;
```

{$ENDIF}

{$IFDEF DOSMODE}

procedure Delay (n : longint);

const

```
TicksPerDay = 1572480;
```

var

```
DelayQnt : longint;
DoneTime : longint;
LastTime : longint;
ThisTime : longint;
DateFlag : boolean;
nh, nl   : word;
```

begin

```
if Running_OS = OS_OS2_2 then begin
  {$IFDEF VER70}
    nh := n shr 8 shr 8;
  {$ELSE}
    nh := n shr 16;
  {$ENDIF}
  nl := n and $FFFF;
  asm
    mov    dx, nh;
    mov    ax, nl;
    hlt;
    db     $35,$CA
  end;
  exit
end;
```

```
DoneTime := MemW [Seg0040:$006C];                  { What time is it?     }
DelayQnt := round (n / 1000 * 18.2);               { How many ticks wait? }
DateFlag := (DoneTime + DelayQnt) >= TicksPerDay;  { Skip midnight?       }
DoneTime := (DoneTime + DelayQnt) mod TicksPerDay; { When we'll finish?   }
```

```
LastTime := MemW [Seg0040:$006C];
```

```
while (DateFlag or (LastTime < DoneTime)) do begin
```

```
  { probably fixed damned midnight freeze }
```

```
  ThisTime := MemW [Seg0040:$006C];
  if ThisTime < LastTime then   { A new day! }
    DateFlag := false;
  LastTime := ThisTime;
```

```
  { Release timeslice }
```

```
  case Running_OS of
```

```
    OS_TOPVIEW, OS_DESQVIEW:
      begin
        r.AX := $1000;
        Intr ($15, r)
      end;
```

```
    OS_DOUBLEDOS:
      begin
        r.AH := $EE;
        if DelayQnt > 767 then
          r.AL := $FF
        else
```

```
          r.AL := DelayQnt div 3;
        dec (DelayQnt, r.AL * 3);
        Intr ($21, r)
      end
```

```
    else
      begin
        r.AX := $1680;
        Intr ($2F, r)
      end;
  end
end
```

end;

BEGIN

```
r.AX := $3000;
MsDos (r);
dosvh := r.AL;
dosvl := r.AH;
if r.BH = $00 then
  vendor := 'PC'
else if r.BH = $66 then
  vendor := 'PTS'
else if r.BH = $FF then
  vendor := 'MS'
else
  vendor := 'OEM';
```

```
{ Check for Novell NetWare to eliminate conflict with DoubleDOS detection }
```

```
r.AX := $DC00;
Intr ($21, r);
```

```
if r.AL = 0 then begin
  { NetWare is not installed, so we can check for DoubleDOS }
  r.AX := $E400;
  Intr ($21, r);
  if r.AL <> 0 then begin   { Yes, DoubleDos }
    Running_OS := OS_DOUBLEDOS;
    Running_OS_Name := 'DoubleDos';
    exit
  end;
end;
```

```
{ Check for DesqView }
```

```
r.AX := $1022;
r.BX := $0000;
```

```
Intr ($15, r);
```

```
if r.BX <> 0 then begin   { Yes, DesqView or TopView }
  if r.BX <> $0A01 then begin
    Running_OS := OS_TOPVIEW;
    Running_OS_Name := 'TopView ' + Version (r.BL, r.BH)
  end
  else begin
    Running_OS := OS_DESQVIEW;
    r.CX := $4445;   { 'DE', Serg Projzogin uses it }
    r.DX := $5351;   { 'SQ', Serg Projzogin uses it }
    r.AX := $2B01;
    Intr ($21, r);
    Running_OS_Name := 'DesqView ' + Version (r.BH, r.BL)
  end;
  exit
end;
```

```
{ Check for OS/2 }
```

```
r.AX := $4010;
r.BX := $0000;
Intr ($2F, r);
```

```
if r.BX <> 0 then begin   { Yes, OS/2 }
  if r.BH >= 20 then
    Running_OS := OS_OS2_2
  else
    Running_OS := OS_OS2_1;
  Include (AccessDenied, 162);
  if (r.BH <> dosvh) or (r.BL <> dosvl) then begin { DOS VMB under OS/2 }
    osvh := r.BH div 10;
    osvl := r.BL;
    if (osvh = 2) and (osvl >= 30) and (osvl < 90) then begin
      osvh := osvl div 10;
      osvl := osvl mod 10
    end;
    Running_OS_Name := vendor + ' DOS ' + Version (dosvh, dosvl) +
                       ' under OS/2 ' + Version (osvh, osvl);
    exit
  end;
  dosvh := dosvh div 10;
  if (dosvh = 2) and (dosvl >= 30) and (dosvl < 90) then begin
    dosvh := dosvl div 10;
    dosvl := dosvl mod 10
  end;
  Running_OS_Name := 'OS/2 ' + Version (dosvh, dosvl);
  exit
end;
```

```
r.AX := $1600;
```

```
Intr ($2F, r);
```

```
if r.AL <> 0 then begin   { Yes, Windows }
  Running_OS := OS_WINDOWS;
  if r.AX = $0004 then
    Running_OS_Name := 'Windows 95'
  else if r.AX = $0A04 then
    Running_OS_Name := 'Windows 98'
  else if r.AX = $5A04 then
    Running_OS_Name := 'Windows Me'
  else
    Running_OS_Name := 'Windows ' + Version (r.AL, r.AH);
  exit
end;
```

```
Running_OS := OS_MSDOS;
Running_OS_Name := vendor + ' DOS ' + Version (dosvh, dosvl);
```

```
{$ENDIF}
```

```
END.
```

————————————————————————————[Cut Here]————————————————————————————