

Im Artikel *Lichtstärkemessung am PC* (Elektor EXTRA 2/98) wurde behauptet, daß eine präzise Zeitmessung in Visual BASIC nicht möglich wäre. Unser Leser Dipl. Ing. J. C. Feltes weiß es besser und will Ihnen die Lösung nicht vorenthalten.

Entwurf von J. C. Feltes (Luxemburg)

# Zeitmessung in Visual BASIC

## Pausenfunktion mit 100 µs Auflösung

Tatsächlich ist der Einsatz des Timers von Visual BASIC wegen der nur groben Auflösung nur für längere Intervalle geeignet. Die API-Funktion SLEEP scheint für eine höhere Genauigkeit auch bei kurzen Intervallen geeignet. Sie muß mit

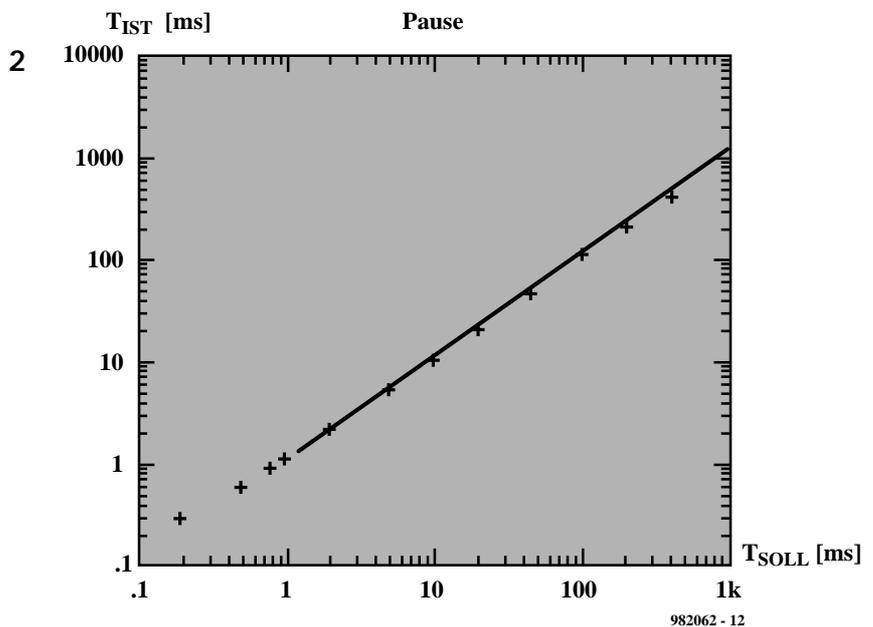
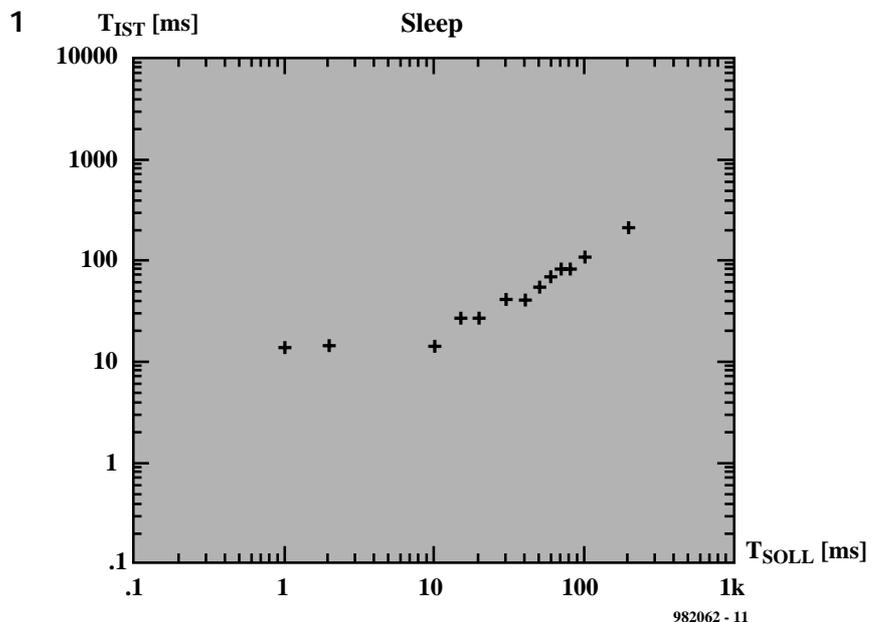
```
Declare Sub Sleep Lib "Kernel32"
    (ByVal Milliseconds As Long)
```

deklariert werden. Allerdings zeigen konkrete Messungen mit dem einfachen Testprogramm

```
Sub Rect(t As Long, t1 As Long)
    For i = 1 To t1
        Sleep t
        PortOut &H278, 1
        Sleep t
        PortOut &H278, 0
    Next i
End Sub
```

daß auch die Auflösung von SLEEP nicht besser ist, als dies die Ticks des Timer-Interrupts vorgeben, also in der gleichen Größenordnung wie die Timer-Steuerung liegt. Das Ergebnis der Messung ist in **Bild 1** zu sehen. Auf der horizontalen Achse ist der Sollwert, auf der vertikalen der Istwert eingetragen (beides in Millisekunden). Gemessen wurde auf einem Pentium 133 MHz unter Windows 95. Das Diagramm zeigt, daß die Dauer der Pausen nur dann annähernd übereinstimmt, wenn sie nicht kürzer als etwa 14 ms sind.

Will man aber beispielsweise Schrittmotoren ansteuern, so benötigt man eine wesentlich kürzere, einigermaßen reproduzierbare und rechnerunabhängige Verzögerungszeit im



Bereich einiger Millisekunden. Mit dem Timer oder SLEEP kommt man jedenfalls nicht weiter. Ein brauchbarer Kurzzeit-Timer nutzt dagegen einen der drei Zähler, die der in jedem PC mindestens einmal vorhandene Intervall-Timer 8253/8254 zur Verfügung stellt. Denn während ein Zähler für den Speicher-Refresh und ein zweiter für die Systemuhr unabdingbar ist, wirkt der dritte Zähler nur bei der Tonerzeugung mit, ist aber ansonsten frei verfügbar.

Dieser freie Zähler 2 zählt im Modus 0 von einem geladenen Startwert bis auf Null zurück. Die Länge jedes Zählschritts und deshalb des Intervalls ist genau definiert. Nach Abschluß des Zählvorgangs geht der Zählerausgang von Null nach Eins. Leider ist es nicht bei jedem PC möglich, den Ausgang des Zählers permanent zu lesen, wenn nämlich der Zähler nur 8253-kompatibel ist und den Readback-Befehl nicht unterstützt. So wird statt des Ausgangs der Zählerstand fortwährend gelesen. Wenn der neue Wert größer wird als der alte Zählwert, bedeutet dies, daß der Zähler übergelaufen ist, also das Intervallende erreicht hat.

Ein Programm wie das folgende springt in diesem Augenblick aus einer Warteschleife:

```
Public Sub Sleepshort (ByVal SDelay As Long)
    SDelay As Long
    SDelay als Vielfaches von 0.838us
    Dim x As Long, y As Long, i As Long, zold As Long, Z As Long

    x = PortIn(&H61)
    x = x Or 1
    PortOut &H61, x

    PortOut &H43, &H80
    x = SDelay And &HFF
    PortOut &H42, x
    x = SDelay \ &H100
    PortOut &H42, x

    zold = 100000
    Do
        PortOut &H43, &H80
        x = PortIn(&H42)
        y = PortIn(&H42)
        Z = x + y * &H100
        If Z > zold Then Exit Do
        zold = Z
    Loop

End Sub
```

Zunächst wird an Port 61<sub>H</sub> Bit 0 auf logisch Eins gesetzt und damit der Gate-Eingang des Zählers 2 aktiviert, anschließend das Steuerwort 80<sub>H</sub> = 10110000<sub>B</sub> an Port 43<sub>H</sub> ausgegeben, danach wird das niederwertige, dann höherwertige Byte des Startwerts über Port 42<sub>H</sub> an den Zähler übertragen. Da

10	11	000	0
Select counter 2	Read/Write-Modus Zunächst Low Byte, dann High Byte	Counter Mode 0: Interrupt bei Zählende	BCD/Binär Zählung im Binärcode

die Eingangsfrequenz des Zählers unabhängig vom PC immer 1,19318 MHz beträgt, ist die zu erwartende Verzögerungszeit immer ein Vielfaches von 0,838  $\mu$ s. So ergibt sich eine maximale Pause bei SDelay = 65535 von 54,9 ms. Der zu übertragende Wert ist eigentlich eine vorzeichenlose ganze Zahl. Da es diesen Zahlentyp in Visual BASIC nicht gibt und der maximale Integerwert 32767 beträgt, wird SDelay als LONG übergeben und intern in ein High Byte und ein Low Byte zerlegt.

In der Schleife wird fortwährend mit 80<sub>H</sub> = 1000000<sub>B</sub> der Zählerwert ermittelt:

10	00	000	0
Select Counter 2	Counter Latch Zustand intern speichern	ohne Bedeutung	ohne Bedeutung

Wie Messungen (Bild 2) zeigen, wird die Pausendauer von Sleepshort im Bereich 0,5...50 ms mit einer Toleranz von etwa  $\pm 10$  % eingehalten. Eine weitere Verbesserung wäre durch die Verlagerung des SUBs in Assemblercode innerhalb der DLL möglich. Aufgrund der Zählerkapazität von 16 Bit ist keine längere Pause als 54,9 ms möglich. Es bietet sich an, je nach gewünschter Pausendauer Sleepshort beziehungsweise Sleep zu benutzen. Dies erledigt das SUB Pause automatisch. Bei größeren Intervallen wird zudem zwischendurch DoEvents aufgerufen, um eine Reaktion auf Events zu ermöglichen.

```
Public Sub Pause (Milliseconds As Single)

    Dim Rest As Long, M1 As Long, i As Long

    ganz kurz?
    If Milliseconds < 50 Then
        Sleepshort Milliseconds * 1000 / 0.838
        Exit Sub
    End If

    länger als 50 ms
    BreakFlag = False
    If Milliseconds <= 500 Then
        Sleep Milliseconds
    Else
```

```
M1 = Milliseconds \ 500
    Ganzzahl division!!!
    Rest = Milliseconds - M1 * 500
    For i = 1 To M1
        Sleep 500
        DoEvent
        If BreakFlag Then Exit For
    Next i
    Sleep Rest
End If

End Sub
```

Ein längeres Intervall sollte durch den Aufruf einer Breakpause, die das Breakflag setzt, unterbrochen werden

können

```
Public Sub BreakPause()
    BreakFlag = True
End Sub
```

Allgemein muß betont werden, daß echte Echtzeitanwendungen unter Windows eigentlich nicht möglich sind, da jederzeit das Betriebssystem dazwischenfunken kann. Der etwas unregelmäßige Programmablauf kann gut beobachtet werden, wenn man mit obigem Programm einen D/A-Wandler steuert und das Ergebnis der Wandlung im Oszilloskop betrachtet. Es zeigt sich die erwartete Sägezahnkurve, allerdings mit mehr oder weniger langen Pausen in Form von horizontalen Abschnitten. Die Pausen werden breiter, wenn die Maus oder die Tastatur aktiv ist oder ein Zugriff auf die Festplatte erfolgt. Bei einem modernen PC liegen die Pausen allerdings nur im Millisekundenbereich, so daß sie sich bei nicht allzuschleunigen Anwendungen kaum auswirken.

(982062)rg

Der Artikel ist in erweiterter Form in der Januar Ausgabe '98 der Zeitschrift Basic Pro erschienen.