

Inhalt:

Wissenswertes zu den Time-Funktionen:	1
Source Code of Clock Application for eigerPanel:	2



ScreenShot der Applikation CLCK auf dem eigerPanel70C (WVGA)



ScreenShot der Applikation CLCK auf dem eigerPanel57H/C (VGA)

Wissenswertes zu den Time-Funktionen:

Die Time-Funktionen (-Methoden) finden Sie im eigerScript Software-Manual beschrieben:

www.eigergraphics.com/eigerScript-SoftwareManual.pdf

eigerScript stellt folgende Time-Funktionen zur Verfügung:

<code>Time.Get()</code>	ermittelt aktuelle Zeit und Datum von der RTC (Real Time Clock) und füllt die entsprechenden Time-Register (s.u.).
<code>Time.Set()</code>	stellt Zeit und Datum der RTC entsprechend den Werten, die zuvor allen Time-Registern zugewiesen wurden.
<code>Time.SetTime()</code>	stellt die Zeit der RTC entsprechend den Werten der Time-Register <code>eI.HOURS</code> , <code>eI.MIN</code> , <code>eI.SEC</code> und <code>eI.MSEC</code> .
<code>Time.SetDate()</code>	stellt das Datum der RTC entsprechend den Werten der Time-Register <code>eI.DATE</code> , <code>eI.MONTH</code> und <code>eI.YEAR</code> .
<code>Time.DayOfWeek()</code>	weist der RTC den Wochentag zu entsprechend dem Wert des Time-Registers <code>eI.DOW</code> .
<code>Str.Date(Date.\$, FormatDate_ISO8601_YYYY_MM_DD)</code>	weist einem String das Datum aus der RTC zu mit einem speziellen Format, z.B. 2010-08-12 (die Format-Auswahl finden Sie im eigerScript Software-Manual).
<code>Str.Time(Time.\$, FormatTime_HH_MM_SS)</code>	weist einem String die Zeit aus der RTC zu mit einem speziellen Format, z.B. 09:25:13 (die Format-Auswahl finden Sie im eigerScript Software-Manual).

Dazu gehören folgende Time-Register:

eI.YEAR	Calendar year, e.g. 2010, 2011 etc.
eI.MONTH	Month of year, 1 .. 12
eI.DOW	Day of week, e.g. 1 for Mondays .. 7 for Sundays
eI.DATE	Day of month, 1 .. 31
eI.HOURS	Hour of day, 0 .. 23
eI.MIN	Minute of hour, 0 .. 59
eI.SEC	Second of minute, 0 .. 59
eI.MSEC	Millisecond of second, 0 .. 999

Wenn in einer Applikation die bei Laufzeit aktuelle Zeit bzw. das aktuelle Datum verwendet werden soll, müssen diese zuerst mit der Methode `Time.Get()` aus der RTC ermittelt werden, d.h. die Register mit den aktuellen Werten gefüllt werden. Danach können die Werte der Time-Register z.B. für die Zeit-Anzeige auf dem Display verwendet werden. Für eine Zeit-Anzeige mit Sekunden-Angabe, ist also mindestens jede Sekunde eine Zeitabfrage von der RTC mit `Time.Get()` erforderlich.



Source Code of Clock Application for eigerPanel:

```
-----  
; Titel      : Time functions   View 1  
; File      : CLCK_001.EVS  
-----  
; Compiler  : eigerScript  
;  
; System    : eigergraphics.com; FOXS embedded computer  
;  
; Description: Example of implementing time functions  
;  
; Version   : Initial version: 03.01.2011
```



```

;
; Autor      : Christoph Angst
;
;-----
; (c) 2005-2011      S-TEC electronics AG, CH-6314 Unterägeri; +41 41 754 50 10
;-----

EIGERPROJECT 'CLCK'          ; Project name; first part of EVI file name
EIGERVIEW 1                  ; View number; second part of EVI file name: CLCK_001.EVI

IMPORT      'DEF/DEF_eVM_OpCodes.h'          ; Token
IMPORT      'DEF/DEF_eVM_Registers.h'        ; Register
FUNCLIB     'DEF/DEF_eVM_Functions.lib'      ; Functions library

INCLUDEFILE 'DEF/DEF_eiger_Colors.INC'      ; Color definitions 12.03.2006
INCLUDEFILE 'DEF/DEF_eiger_Types.INC'       ; eiger definitions
INCLUDEFILE 'DEF/DEF_eiger_StackMachine.INC' ; Stack Machine

; D E C L A R A T I O N S _____

; Geometrie für Zeitanzeige .....
CONST      TimeWidth          = 220 ; Label width
CONST      TimeHeight         = 60  ; Label height
CONST      Time_Y             = 200 ; Label Y-position
CONST      DayAndDateWidth    = 190 ; Label width
CONST      DayAndDateHeight   = 40  ; Label height
CONST      DayAndDate_Y       = Time_Y + TimeHeight + 100 ; Label Y-position

CONST      Label_X            = 50   ; Label X-position

; Variables for time display .....

INTEGER    FontColor.I = 0
INTEGER    DOW_old.I = 0
STRING [15] Weekday.$ = ''
STRING [10] Date.$ = ''
STRING [50] DayAndDate.$ = ''
STRING [10] Time.$ = ''

; S U B R O U T I N E S _____

```



```

; Displaying date and time -----
SUB   Time_Timer
Timer.InstallLocal(1, GetTime)
Timer.Load(1,1000)           ; actualizing each second
Timer.StartContinuous(1)
ENDSUB

SUB   GetTime           ; distinguish office hours from spare time
Time.Get()             ; get actual time and date from the panels RTC (Real Time Clock)

IF eI.DOW < 6 THEN      ; DayOfWeek: Mo-Fr that means 1 to 5; Sa = 6, So = 7
  IF eI.HOURS >= 17 THEN ; after closing time
    FontColor.I := lightblue ; blue for outside office hours
  ELSIF eI.HOURS >= 13 THEN ; lunch time up to 1 pm
    FontColor.I := white     ; red for office hours
  ELSIF eI.HOURS >= 12 THEN ; start of lunch time
    FontColor.I := lightblue ; blue for outside office hours
  ELSIF eI.HOURS >= 8 THEN  ; start of work at 8 am
    FontColor.I := white     ; red for office hours
  ELSE                   ; other time out outside office hours
    FontColor.I := lightblue ; blue for outside office hours
  ENDIF
ELSE
  FontColor.I := lightgreen ; green for weekend
ENDIF

IF DOW_old.I <> eI.DOW THEN
  CallSubroutine(Display_DayAndDate)
ENDIF
CallSubroutine(Display_Time)

ENDSUB

SUB   Display_DayAndDate

DOW_old.I := eI.DOW
IF eI.DOW == 1 THEN
  Weekday.$ := 'Monday'
ELSIF eI.DOW == 2 THEN
  Weekday.$ := 'Tuesday'
ELSIF eI.DOW == 3 THEN
  Weekday.$ := 'Wednesday'

```



```

ELSIF eI.DOW == 4 THEN
    Weekday.$ := 'Thursday'
ELSIF eI.DOW == 5 THEN
    Weekday.$ := 'Friday'
ELSIF eI.DOW == 6 THEN
    Weekday.$ := 'Saturday'
ELSIF eI.DOW == 7 THEN
    Weekday.$ := 'Sunday'
ENDIF

Str.Date(Date.$, FormatDate_ISO8601_YYYY_MM_DD) ; fill string with actual date e.g. 2010-08-12
DayAndDate.$ := Weekday.$ ; Thursday

Str.Concat(DayAndDate.$, ', ') ; Thursday,
Str.Concat(DayAndDate.$, Date.$) ; Thursday, 2010-12-09

; other possibility to get the same result:
; Str.Concat(DayAndDate.$, ', ') ; Thursday,
; Str.Cvt_Integer(DayAndDate.$, eI.YEAR, 4) ; Thursday, 2010
; Str.Concat(DayAndDate.$, '-') ; Thursday, 2010-
; eI.FillChar := "0" ; Zero ahead of Numbers below 10, e.g. 09 instead of 9
; Str.Cvt_Integer(DayAndDate.$, eI.MONTH, 2 ) ; Thursday, 2010-12
; Str.Concat(DayAndDate.$, '-') ; Thursday, 2010-12-
; Str.Cvt_Integer(DayAndDate.$, eI.DATE, 2) ; Thursday, 2010-12-09

Fill.LabelParameter(TimeStamp_Style) ; Loading the properties for the Label according to TimeStamp_Style (see below)
Load.Geometry_XYWH(Label_X, DayAndDate_Y, DayAndDateWidth, DayAndDateHeight)
eI.TextColor := FontColor.I)

Label.Text(DayAndDate.$)
ENDSUB

SUB Display_Time

Display.Prepare ; prepare in AVR (Accessible Video Ram) before displaying
eI.FillChar := "0" ; Zero ahead of Numbers below 10, e.g. 09 instead of 9
Time.$ := '' ; clear string
Str.Time(Time.$, FormatTime_HH_MM_SS) ; fill string with actual time e.g. 09:25:13

; other possibility to get the same result:
; Str.Cvt_Integer(Time.$, eI.HOURS, 2) ; 09
; Str.Concat(Time.$, ':') ; 09:
; Str.Cvt_Integer(Time.$, eI.MIN, 2) ; 09:25
; Str.Concat(Time.$, ':') ; 09:25:

```



```

;      Str.Cvt_Integer(Time.$, eI.SEC, 2)      ; 09:25:13

Fill.LabelParameter(TimeStamp_Style)      ; Loading the properties for the Label according to TimeStamp_Style (see below)
Load.Geometry_XYWH(Label_X,Time_Y,TimeWidth,TimeHeight)
; Label.Color(FontColor.I)                ; Color of eI.FillColor, eI.LineColor, eI.BackColor (autocolor for eI.TextColor)
eI.TextColor := FontColor.I)
eI.FontNumber := Font_DigitalNumbers_48

Label.Text(Time.$)
Display.ShowWindow()                      ; copying the window (Geometry_XYWH) from AVR to RVR (Refresh Video Ram)
ENDSUB

; S T Y L E S _____

SUB Styles

TimeStamp_Style:
INLINWORDS (0)                ; corresponds to eI.Pos_X1
INLINWORDS (0)                ; corresponds to eI.Pos_Y1
INLINWORDS (as_DisplayWidth) ; corresponds to eI.Width
INLINWORDS (18)               ; corresponds to eI.Height
INLINWORDS (20)               ; corresponds to eI.SpaceLeft
INLINWORDS (8)                ; corresponds to eI.SpaceRight
INLINWORDS (0)                ; corresponds to eI.HorizontalAdjust
INLINWORDS (0)                ; corresponds to eI.VericalAdjust
INLINWORDS (black)            ; corresponds to eI.FillColor
INLINWORDS (black)            ; corresponds to eI.BackColor
INLINWORDS (black)            ; corresponds to eI.LineColor
INLINWORDS (lightyellow)      ; corresponds to eI.TextColor
INLINWORDS (Pos_left)         ; corresponds to eI.Position
INLINWORDS (Orientation_0deg) ; corresponds to eI.Orientation
INLINWORDS (normal)           ; corresponds to eI.Appearance
INLINWORDS (no_border)        ; corresponds to eI.BorderStyle
INLINWORDS (Font_Arial_24n)   ; corresponds to eI.FontNumber

ENDSUB

; M A I N P R O G R A M _____
BEGINVIEW
EVE.Init()                      ; initialize EVE ANNA
Load.Pos_X1Y1(0,0)              ; up left corner of screen

```



```
Display.Prepare()           ; prepares subsequent Layout first in AVR (Accessible Video Ram) before copying it to RVR
and Screen
File.Read_EGI('C:\\CLCK\\PICT\\UNIVERSE.EGI') ; zeichnet Hintergrund-Bild "UNIVERSE.EGI"

CallSubroutine(GetTime)

Display.Show() ; display Layout on the screen by copying it into the RVR (Refresh Video Ram)

CallSubroutine(Time_Timer)

;HotSpot.TableEnable()
Timer.TableEnable()

LOOP
ENDLOOP
ENDVIEW
```