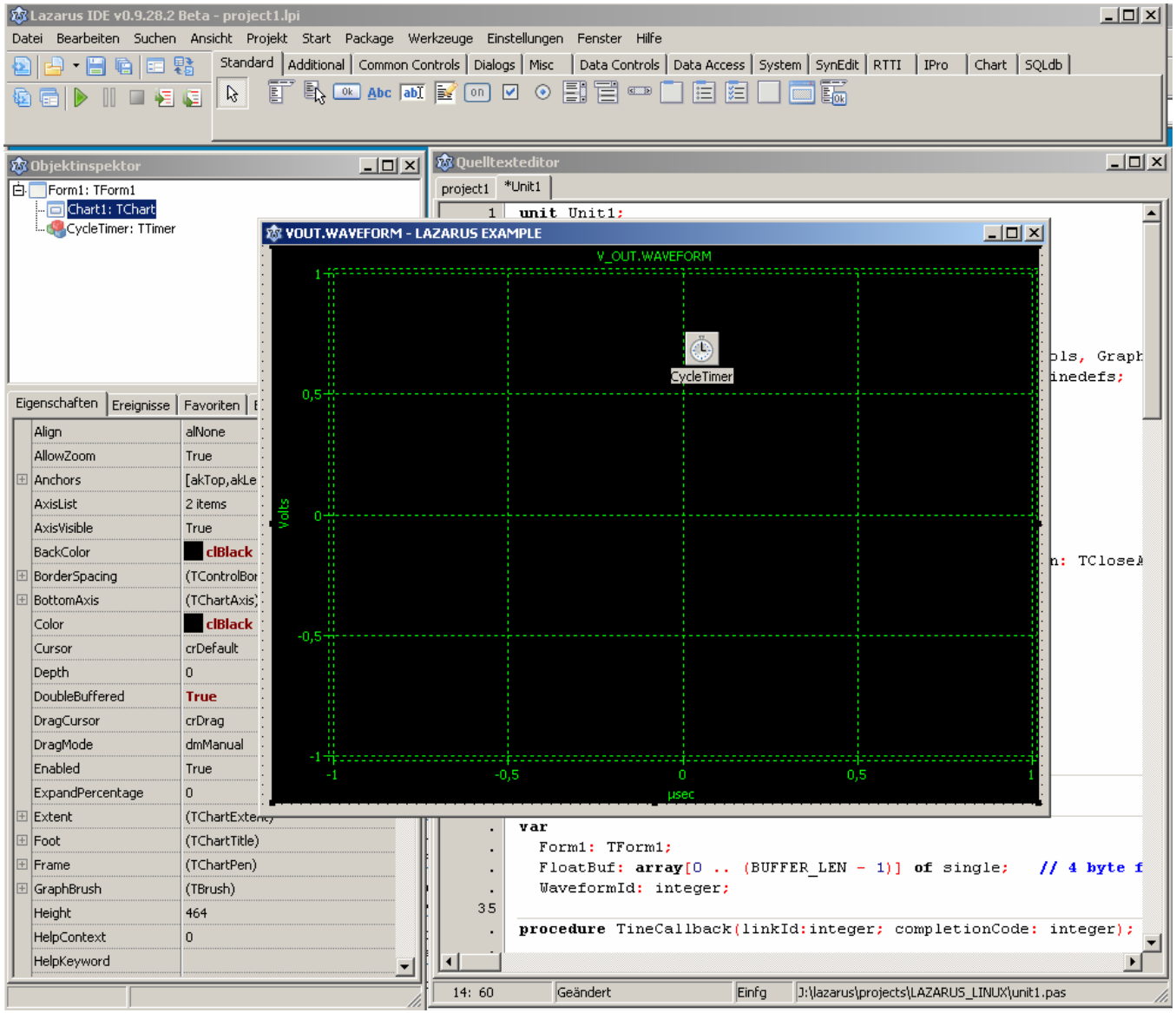
Using TINE with ObjectPascal

Short Introduction

- Why ObjectPascal?
  - Open Source (Lazarus + FPC) and commercial (Delphi)
  - Windows, GNU/Linux, FreeBSD and MacOSX with same source code if using Lazarus
  - Nearly same language in Lazarus and Delphi
  - GUI oriented programs, powerful IDE with integrated debugger, easy to learn, GUI design with a few clicks
  - Lot‘s of freeware components and extensions.
  - perfect for small clients
- No changes to TINE Libraries, just use them.
  - TINE ObjectPascal packet is a pascal translation of the TINE client C-API headers.
  - simply added support for additional language

Lazarus IDE


Lazarus IDE v0.9.28.2 Beta - project1.lpi
Datei Bearbeiten Suchen Ansicht Projekt Start Package Werkzeuge Einstellungen Fenster Hilfe
Standard Additional Common Controls Dialogs Misc Data Controls Data Access System SynEdit RTTI IPro Chart SQLdb
Objektinspektor
Form1: TForm1
Chart1: TChart
CycleTimer: TTimer
Eigenschaften Ereignisse Favoriten
Align alNone
AllowZoom True
Anchors [akTop,akLe
AxisList 2 items
AxisVisible True
BackColor clBlack
BorderSpacing (TControlBor
BottomAxis (TChartAxis)
Color clBlack
Cursor crDefault
Depth 0
DoubleBuffered True
DragCursor crDrag
DragMode dmManual
Enabled True
ExpandPercentage 0
Extent (TChartExtent)
Foot (TChartTitle)
Frame (TChartPen)
GraphBrush (TBrush)
Height 464
HelpContext 0
HelpKeyword
VOUT.WAVEFORM - LAZARUS EXAMPLE
V_OUT.WAVEFORM
CycleTimer
Volts
usec
Quelltexteditor
project1 *Unit1
unit Unit1;
var
Form1: TForm1;
FloatBuf: array[0 .. (BUFFER_LEN - 1)] of single; // 4 byte f
WaveformId: integer;
procedure TineCallback(linkId:integer; completionCode: integer);
14: 60 Geändert Einfg J:\lazarus\projects\LAZARUS_LINUX\unit1.pas

C to ObjectPascal translation

	TINE C header	TINE ObjectPascal
constant	#define ENSTAG "ENS"	const ENSTAG = 'ENS';
unsigned 16-bit	UINT16, short	UINT16, word
pointer to unsigned short	short *	Pshort, pUINT16
4byte float	float	single, float
8byte float	double	double
pointer to 4byte float	float *	Pfloat, ^single
structure type	typedef struct { .. };	type record begin ..end;
union type	union	record case Integer of ..
address	&	@
dereference	<var>*	<var>^

- TINE types have the same names:
  - DUNION, DUSTRING, DUTYPE, SPECTRUM, INTINTINT, FWINDOW, ..
- TINE constants have the same names:
  - CA_READ, CM_POLL, ..
- Pointer types are named starting with an „p“ or „P“ (i.e. pDUSTRING)
- Functions are named equally and take the same arguments as in C client API
- C source documentation will mostly fit

Basic usage – mini howto

- Add to your apps Mainform.OnCreate:

```
InitializeTineclientDll('C:\TINE\system32');
```

- Add a Timer to your Mainform and add to Timer.OnTimer:

```
retry:=true;
while retry do
      retry:=SystemCycle(true);
```

- Add to your apps Mainform.OnClose:

```
UnInitializeTineclientDll();
```

- Add a Callback Function for AttachLink. Note, it‘s declared as cdecl;

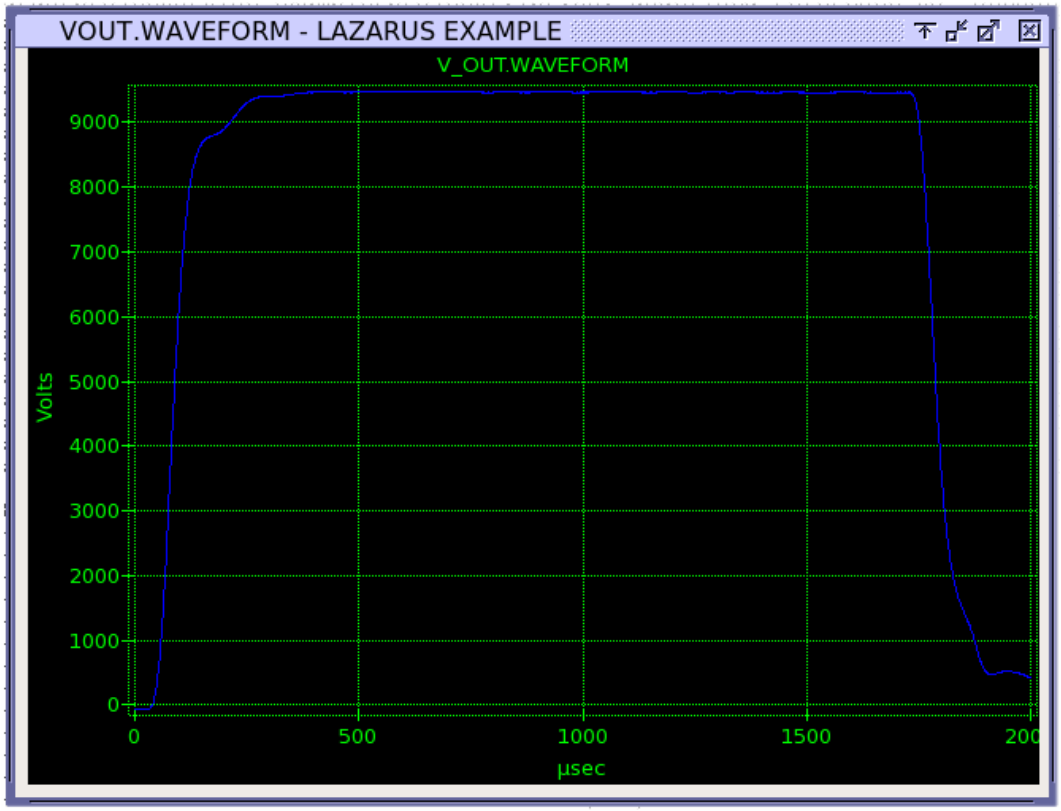
```
procedure TineCallback(linkId:integer; completionCode: integer); cdecl;
begin
   if (completionCode <> 0) then
        ShowMessage('TRC error: ' +GetLastLinkErrorAsString(completionCode))
   else
       begin
       // your data arrived, check the linkId
       end;
end;
```

Basic usage – mini howto (contd.)

- Initialize async link using Attachlink:

```
procedure TForm1.InitTineLink();
   var dout : DTYPE;
   DeviceName    : array[0 .. (DEVICE_NAME_SIZE-1)]  of char;
   PropertyName : array[0 .. (PROPERTY_NAME_SIZE-1)] of char;
begin
  StrPCopy(@DeviceName,   'MTF.RF/MODDATA/KLYMOD2');
  StrPCopy(@PropertyName, 'V_OUT.WAVEFORM');
  dout.dArrayLength := BUFFER_LEN;
  dout.dFormat        := CF_FLOAT;
  dout.data.fptr      := @FloatBuf[0];
  WaveformId := AttachLink(@DeviceName, @PropertyName, @dout, NIL,
    CA_READ, 1000, @TineCallback, CM_POLL);
  if (WaveformId < 0) then
    ShowMessage('ERROR initializing tine link:
    '"+GetLastLinkErrorAsString(WaveformId)+"")
  else
    { start calling SystemCycle periodically }
    Timer.Enabled:=true;
end;
```

Basic usage – mini howto (contd.)



Lazarus example running on GNU/Linux using GTK+, showing data from Modulator Test Facility.
(Source code included in package as sample)

Thank you for attention.