KAntrak 3700

Software Development Kit

User's Guide

P04050A00

Table of Content

1. INTRODUCTION 3

2. HARDWARE SYSTEM 3

3. KANTRAK 3700 CONNECTIONS 4

4. FEATURES 5

5. SDK PACKAGE 5

6. DEVELOPMENT PROCESS 6

7. SOFTWARE PLATFORM 7

7.1 KANTRAK LIBRARY 9
7.1.1 KDISPLAY 10
7.1.2 KKEYPAD 10
7.2 CLUTTER 12
7.2.1 Main functionalities 12
7.2.2 Supported image format 12
7.2.3 Image/text positioning 13
7.2.4 Text 14
7.2.5 Image 16
7.2.6 Creating and animating a gauge 18
7.2.7 Animations 20
7.2.8 Additional Clutter resources 21
7.3 GLIB 22
7.3.1 Main functionalities 22
7.3.2 Additional GLib resources 22

8. KANTRAK 3700 APPLICATION 23

8.1 CREATING A NEW APPLICATION 23
8.2 DATA FOLDER 24
8.3 SIMULATOR 25

9. KANTRAK 3700 APPLICATION EXAMPLES 26

10. LICENSES 27

10.1 CLUTTER 27
10.1.1 Software 27
10.1.2 Documentation 27
10.2 GLIB 27

1. Introduction

This manual is intended to explain the basics of the KAntrak 3700 SDK. This should be interpreted as a good start point for a designer having to develop his own application as well as doing field support. The user must be aware that understanding the following concepts requires a minimum skill level in embedded systems and related software development tools. Thus, the following sections will not explain in depth the functionality of third party tools but will mainly deal with the KAntrak 3700 system particularities. You are also welcome to start your application using the supplied code while respecting the license agreement (*see Quick Start Guide).*

2. Hardware System

The circuit board can be divided in different blocks, as shown in the next diagram:

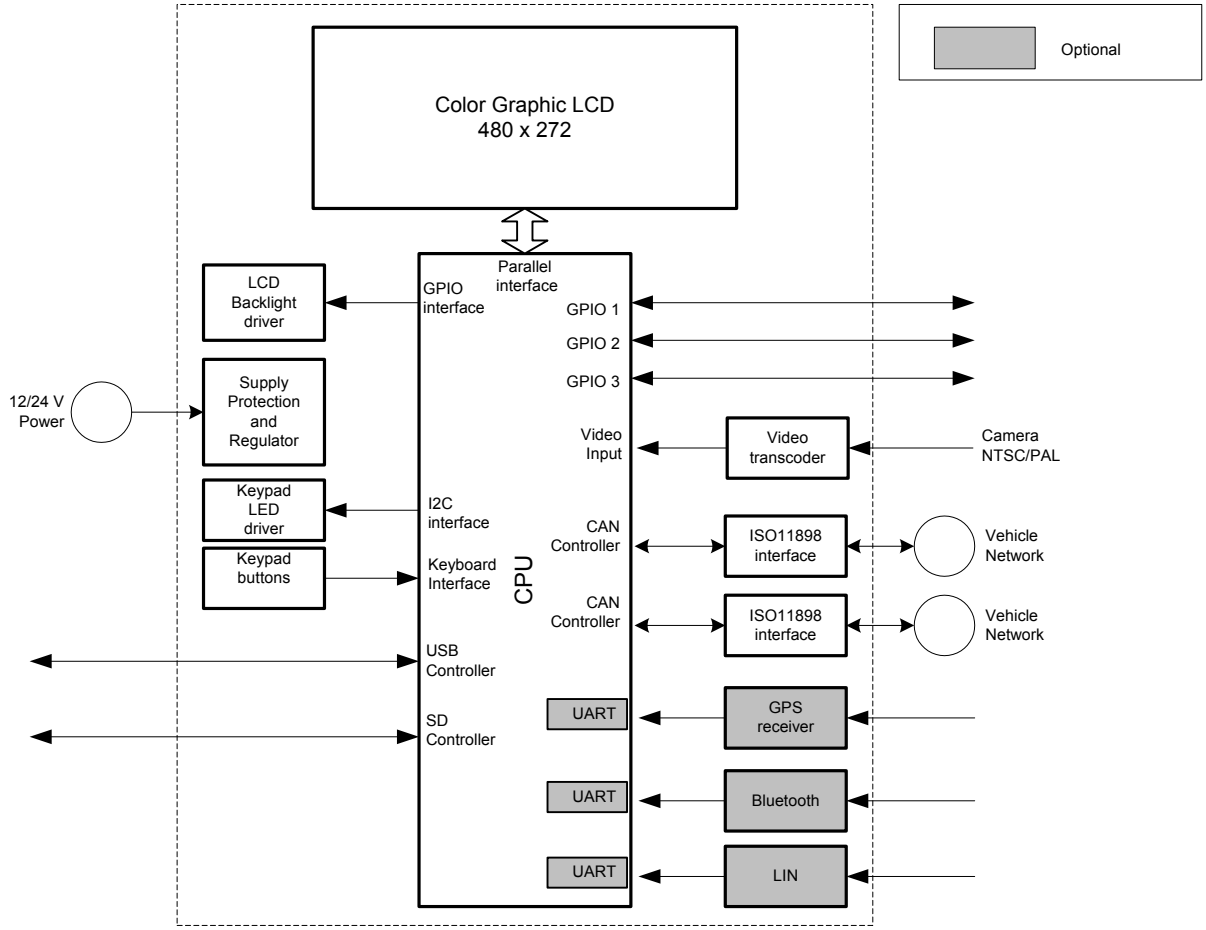

Figure 1 Software platform block diagram

3. KAntrak 3700 Connections

Figure 2 KAntrak 3700 connections

1. Micro SD port
2. Mini USB port
3. Deutsch 12 way connector:

    1. Power –ve
    2. Power +ve
    3. CAN HI (1)
    4. CAN LOW (1)
    5. Configurable I/O (1)
    6. Configurable I/O (2)
    7. CAN LOW (2)
    8. CAN HI (2)
    9. Ignition input
    10. Configurable I/O (3) – or LIN Bus (optional)
    11. Video In (-)
    12. Video In (+)

4. Features

Feature	Description
Processor	Freescale iMX534
RAM	256 Megs
Storage (Flash)	512 Megs
GPU	OpenGL® ES 2.0
OS	Linux (Freescale BSP)
Display	480x272 Color LCD
CAN ISO11898	2 ports up to 500 kbps
J1939/ NMEA200	SAE J1939/NMEA2000 Data Link Layers (J1939-21) and Network Management functions (J1939-81)
Configurable I/O	3 (2 Full, 1 Limited)
USB	Slave, Mini B
SD Card	MicroSD
GPS	optional
Video Input	1 NTSC/PAL
Bluetooth	optional
LIN	optional

5. SDK package

Included in the SDK package:

Folder	Description
\examples	KAntrak 3700 application examples
\docs	Libraries documentation
\virtual machine	KAntrak 3700 Oracle VirtualBox virtual machine
\klib	KAntrak 3700 library header files
RELEASE NOTES.txt	KAntrak 3700 SDK release notes

6. Development process

Typical software development process with the KAntrak 3700 SDK involve two entities: industrial or graphic designer and a software designer. At first, industrial or graphic designer creates screen layout/gauges and export it in the form of image files (PNG). Subsequently, the software designer import it into the application, animate it and connect gauges to live data.

Example: an industrial designer created this screen layout and exported all gauges and needle into separated PNG image files. The software designer took the gauges images, animated needles and connected it to the live data.

Figure 3 Screen layout

7. Software Platform

This section describes the software platform architecture. A particular care has been taken to make the software development as easy as possible. Figure 4 shows a block diagram of the implemented architecture. We will explain in more details the purpose of every block in the following sub-sections.

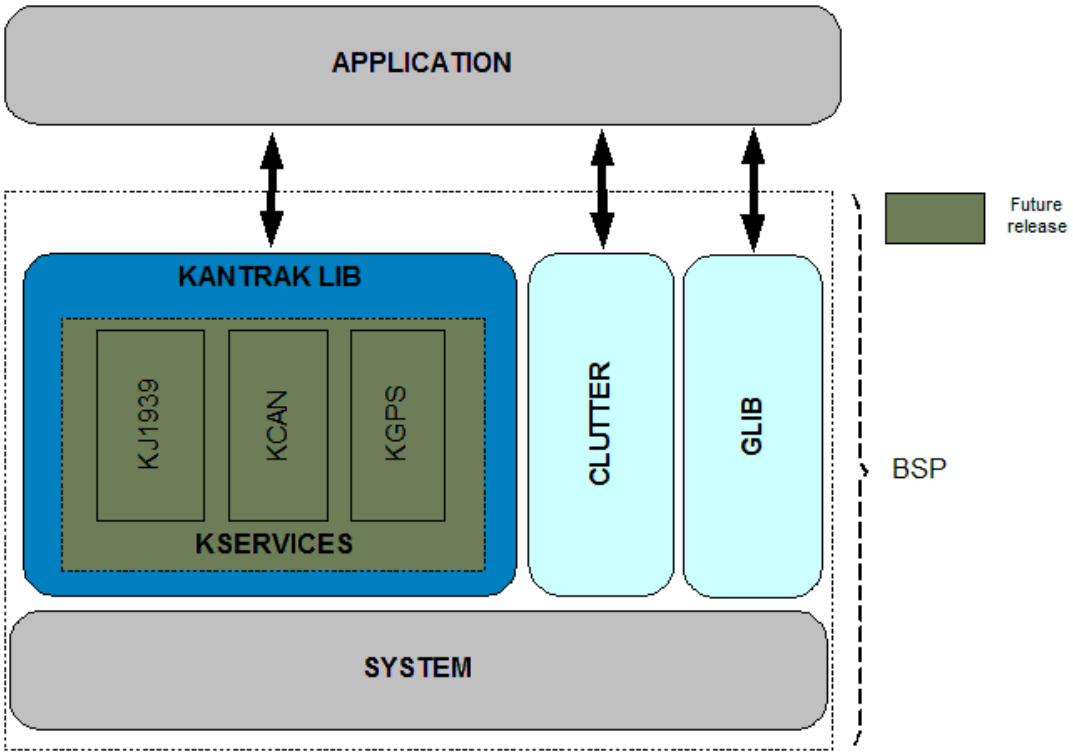

Figure 4 Software platform block diagram

Main features:

- KAntrak 3700 uses Clutter graphical library for all 3D effects and image manipulation
- KAntrak library provides access to devices like display, keypad, keypad LED, camera and some utilities methods.
- GLib is included in the SDK to provide additional data types, files utilities, thread, etc.

- The KAntrak 3700 SDK is built upon an event based framework. Events are registered at startup and are processed automatically in a main loop:

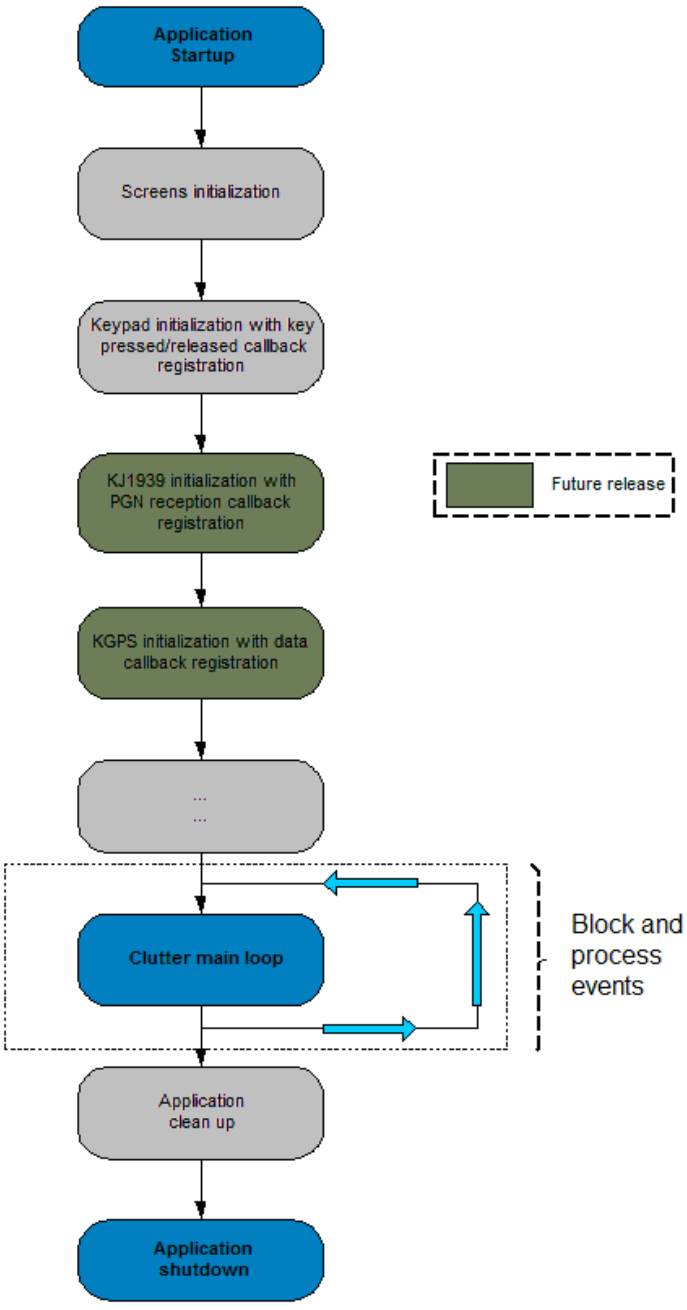

Figure 5 KAntrak 3700 event based framework

More information on main loop processing can be found at: docs/glib/glib-The-Main-Event-Loop.html

7.1 KAntrak library

The KAntrak library (KLIB) gives access to KAntrak 3700 devices and provide some utilities methods to facilitate software development.

Header files can be found into *'klib'* folder and sources documentation is located in '*docs/klib*' folder of the SDK: docs/klib/index.html

File Name	**Description**	**Usage examples**
kdisplay.h	Methods related to LCD peripheral management	examples/image/main.c examples/round_gauge/main.c
kkeypad.h	System keypad management methods	klib/kkeypad.h examples/round_gauge/screen.c
kled.h	System keypad LED management methods	examples/round_gauge/screen.c
kcamera.h	System camera management methods *Note*: camera not available in simulator mode	
kcrc.h	Methods used for CRC calculation	klib/kcrc.h
kiirfilter.h	Methods used for low pass (IIR) filter processing	klib/kiirfilter.h
kmath.h	General mathematic supplement related definitions and methods	examples/round_gauge/round_gauge.c
kutils.h	Software development utilities supplement	
kxml.h	libxml2 XML parsing method supplement	klib/kxml.h
kproduct_info.h	KAntrak 3700 product information method	

7.1.1 KDISPLAY

Supports all KAntrak 3700 display functionalities:

- Initialize display environment
- Methods for setting and getting LCD backlight value
- Method for getting LCD temperature

Main reference: docs/klib/kdisplay_8h.html

7.1.2 KKEYPAD

Supports for KAntrak keypad events.

Main reference: docs/klib/kkeypad_8h.html

Example: from *'klib/kkeypad.h':*

```
92  gint main(gint argc, gchar *argv[])
93  {
94      /** Initialize graphical library */
95      if(clutter_init(&argc, &argv) != CLUTTER_INIT_SUCCESS)
96      {
97          g_critical("main: unable to init graphical library");
98          return -1;
99      }
100
101     /** Initialize KAntrak display */
102     if(!kdisplay_init())
103     {
104         g_critical("main: unable to init kantrak display");
105         return -1;
106     }
107
109 1   if(!kkeypad_init(keypad_hit_cb))
111     {
111         g_critical("main: unable to initialise keypad");
112         return -1;
113     }
114
115     /** Wait for events */
116     clutter_main();
117
119 3   /** Close keypad */
        kkeypad_close();
121
121     return 0;
122 }
```

```
62  void keypad_hit_cb(guint8 key, guint8 state)
63  {
64      /** Print keypad key and state*/
65      switch(key)
66      {
67          case KKEYPAD_KEY1:
68              g_debug("KEY #1 %s", (state == KKEYPAD_KEY_PRESSED) ? "pressed": "released");
69              break;
70
71          case KKEYPAD_KEY2:
72              g_debug("KEY #2 %s", (state == KKEYPAD_KEY_PRESSED) ? "pressed": "released");
73              break;
74
75          case KKEYPAD_KEY3:
76              g_debug("KEY #3 %s", (state == KKEYPAD_KEY_PRESSED) ? "pressed": "released");
77              break;
78
79          case KKEYPAD_KEY4:
80              g_debug("KEY #4 %s", (state == KKEYPAD_KEY_PRESSED) ? "pressed": "released");
81              break;
82
83          case KKEYPAD_KEY5:
84              g_debug("KEY #5 %s", (state == KKEYPAD_KEY_PRESSED) ? "pressed": "released");
85              break;
86
87          default:
88              return;
89      }
90  }
```

Figure 6 Keypad example

Step-by-step:

1- Initialize keypad device and set callback function

2- Process keypad event and print a message

3- Close keypad device on exit

Example references:

Function name	Description	Documentation
kkeypad_init	Initialize keypad device and set event callback function	docs/klib/kkeypad_8h.html
kkeypad_close	Close keypad device	docs/klib/kkeypad_8h.html

7.2 Clutter

Clutter is a software library that uses OpenGL® ES 2.0 for rendering, but wraps an easy to use and flexible API around GL's complexity. KAntrak 3700 SDK uses Clutter for all graphics and 3D effects.

Supported version API documentation can be found into '*docs/clutter*' folder of the SDK:
docs/clutter/index.html

IMPORTANT NOTE: Some display functions have been overloaded by the SDK to ensure proper operation in all build target types. Please see *'klib/kdisplay.h'* for further information and the list of mandatory usage functions

7.2.1 Main functionalities

Name	Description	Documentation
ClutterTexture	Loading images methods	docs/clutter/ClutterTexture.html
ClutterActor	Moving, rotating, sizing, scaling images/text methods	docs/clutter/ClutterActor.html
ClutterText	Text methods	docs/clutter/ClutterText.html
ClutterAnimation	Animation methods	docs/clutter/ClutterAnimator.html
ClutterGroup	Grouping methods	docs/clutter/ClutterGroup.html

Please refer to documentation for more functionalities: docs/clutter/index.html

7.2.2 Supported image format

- Only *Portable Network Graphics (PNG)* image format is supported by KAntrak 3700 SDK.
- KAntrak 3700 resolution is 128 pixels per inch (PPI)

7.2.3 Image/text positioning

Here is a representation of the three axis in the KAntrak display. The origins are in the left top corner of the display.

Figure 7: KAntrak display axis

Here are main methods to position an image or a text field into the display:

Function name	Description	Documentation
clutter_actor_set_x	Set text/image position in x axis	docs/clutter/ClutterActor.html
clutter_actor_set_y	Set text/image position in y axis	docs/clutter/ClutterActor.html

7.2.4 Text

To display text, use ClutterText.

Main reference: docs/clutter/ClutterText.html

Example: from *'examples/round_gauge/screen.c'*:

```
186  static gboolean screen_text_bottom_set(void)
187  {
188      ClutterActor *default_stage;
189
190      /**
191      *   Get default stage from Kongsberg KLIB library,
192      *   this is mandatory for a proper execution in both simulator and kantrak
193      */
194  1   default_stage = kdisplay_stage_get_default();
195
196      /** Set font and text */
197  2   screen_text = clutter_text_new_with_text("LiberationSans Bold 12",
198                    "   SCALE          FLIP-Y      "
199                    "   FLIP-X         MOVE-X      "
200                    "   OPACITY");
201
202  3   clutter_text_set_color( CLUTTER_TEXT(screen_text),
203                              clutter_color_get_static(CLUTTER_COLOR_WHITE)
204                            );
205
206      /** Add text actor to default stage, i.e the main screen */
207  4   clutter_container_add_actor(CLUTTER_CONTAINER(default_stage), screen_text);
208
209  5   clutter_actor_set_y(screen_text, 256);
210
211      return TRUE;
212  }
```

Figure 8: Displaying text

Step-by-step:

1- Get KAntrak main screen stage

2- Create a text box with *'LiberationSans'* bold font size 12

3- Set text color to white

4- Add text to main screen stage

5- Set text y position at bottom screen

Example references:

Function name	Description	Documentation
kdisplay_stage_get_default	Get main screen stage from KAntrak library	docs/klib/kdisplay_8h.html
clutter_text_new_with_text	Create a new text box	docs/clutter/ClutterText.html
clutter_text_set_color	Set text color	docs/clutter/ClutterText.html
clutter_container_add_actor	Add text to main screen	docs/clutter/ClutterContainer.html
clutter_actor_set_y	Set text position in y axis	docs/clutter/ClutterActor.html

Available fonts:

Name	Modes
Mono (use 'LiberationMono' in code)	Regular/Italic/Bold/Bold Italic
Serif (use 'LiberationSerif' in code)	Regular/Italic/Bold/Bold Italic
Sans Serif (use 'LiberationSans' in code)	Regular/Italic/Bold/Bold Italic

More fonts will be added to the platform in future release

7.2.5 *Image*

To load an image, you need to create a clutter texture (ClutterTexture) from a PNG file contained in your data folder and add it to the main screen, i.e the KAntrak default stage.

Main reference: docs/clutter/ClutterTexture.html

Example: from '*examples/image/main.c*':

```
20      /** Initialize graphical library */
21  1   if(clutter_init(&argc, &argv) != CLUTTER_INIT_SUCCESS)
22      {
23          g_critical("main: unable to init graphical library");
24          return -1;
25      }
26
27      /** Initialize KAntrak display */
28  2   if(!kdisplay_init())
29      {
30          g_critical("main: unable to init kantrak display");
31          return -1;
32      }
33
34      /**
35      *   Get default stage from Kongsberg KLIB library,
36      *   this is mandatory for a proper execution in both simulator and kantrak
37      */
38  3   default_stage = kdisplay_stage_get_default();
39      if(!default_stage)
40      {
41          g_critical("main: unable to get default stage");
42          return -1;
43      }
44
45      /** Load image file */
46  4   image = clutter_texture_new_from_file("./data/image.png", NULL);
47      if(!image)
48      {
49          g_critical("main: unable to load image");
50          return -1;
51      }
52
53      /** Center image in the x axis in main screen */
54  5   clutter_actor_set_x( image,
55                           (clutter_actor_get_width(default_stage) / 2) -
56                           (clutter_actor_get_width(image) / 2)
57                         );
58
59  6   /** Add image to default stage, i.e the main screen */
60      clutter_container_add_actor(CLUTTER_CONTAINER(default_stage), image);
61
62      /** Everything is loaded and initialised, show main screen */
63  7   clutter_actor_show(default_stage);
64
```

Figure 9: Loading an image

Step-by-step:

1- Initialize Clutter library

2- Initialize KAntrak display

3- Get KAntrak main screen stage

4- Load a PNG image into a clutter texture

5- Move image in the X axis to center

6- Add image to main screen

7- Show main screen

Example references:

Function name	Description	Documentation
clutter_init	Initialize Clutter library	docs/clutter/clutter-General.html
kdisplay_init	Initialize KAntrak 3700 display	docs/klib/kdisplay_8h.html
kdisplay_stage_get_default	Get main screen stage from KAntrak library	docs/klib/kdisplay_8h.html
clutter_texture_new_from_file	Load a PNG file into Clutter texture	docs/clutter/ClutterTexture.html
clutter_actor_set_x	Set image position in the X axis. '0' value is the left side of the screen	docs/clutter/ClutterActor.html
clutter_container_add_actor	Add texture to main screen	docs/clutter/ClutterContainer.html
clutter_actor_show	Display main screen	docs/clutter/ClutterActor.html

7.2.6 *Creating and animating a gauge*

To create and animate a simple gauge, developer needs to create a group, load a background and a needle image in PNG format then animate needle.

Main references: docs/clutter/ClutterTexture.html, docs/clutter/ClutterActor.html

Example: from *'examples/round_gauge/round_gauge.c'*:

1- Load a PNG images into textures.

```
65      /**
66      *   Create a group for the gauge. This is useful to show and animate both
67      *   gauge background and needle at the same time
68      */
69  1   new_round_gage->group = clutter_group_new();
70
71      /** Load gauge background image file */
72  2   new_round_gage->background = clutter_texture_new_from_file(background_image, &error);
73      if(!new_round_gage->background)
74      {
75          g_critical("gauge_init: unable to load gauge background image %s", error->message);
76          g_free(new_round_gage);
77          return NULL;
78      }
79
80      /** Set gauge background opacity to 255, i.e opaque */
81      clutter_actor_set_opacity(new_round_gage->background, 255);
82
83      /** Add background to gauge group */
84  3   clutter_container_add_actor(CLUTTER_CONTAINER(new_round_gage->group), new_round_gage->background);
85
86      /** Load needle background image file */
87  4   new_round_gage->needle = clutter_texture_new_from_file(needle_image, &error);
88      if(!new_round_gage->needle)
89      {
90          g_critical("gauge_init: unable to load needle image %s", error->message);
91          g_free(new_round_gage);
92          return NULL;
93      }
94
95      /** Add needle to gauge group */
96  5   clutter_container_add_actor(CLUTTER_CONTAINER(new_round_gage->group), new_round_gage->needle);
97
98      /** Add gauge group to default stage, i.e the main screen */
99  6   clutter_container_add_actor(CLUTTER_CONTAINER(default_stage), new_round_gage->group);
100
```

Figure 10 Clutter gauge loading image

Step-by-step:

1- Create a clutter group to be able to manipulate background and needle image at the same time. For example, if you want to rotate or move the entire gauge, you only need to manipulate the group

2- Load background PNG image into clutter texture

3- Add background texture to the gauge group

4- Load needle PNG image into a clutter texture

5- Add needle texture to the gauge group

6- Add the gauge group to the default stage, i.e the main screen

2 - Rotating needle. From *'examples/round_gauge/round_gauge.c':*

```
183  void round_gauge_set_value(round_gauge_t *gauge, guint16 raw_value)
184  {
185      gint16 rotation_point;
186
187      if(!gauge)
188          return;
189
190      /** Interpolate value */
1911     rotation_point = kmath_interpolation_guint16_gint16( gauge->needle_raw_value_min_max,
192                                                          gauge->needle_angle_min_max,
193                                                          sizeof(raw_value),
194                                                          raw_value
195                                                        );
196
197      /** Set needle angle */
1982     clutter_actor_set_z_rotation_from_gravity( gauge->needle,
199                                                rotation_point,
200                                                CLUTTER_GRAVITY_CENTER
201                                              );
202  }
```

Figure 11 Clutter gauge rotating needle

Step-by-step:

1- Calculate rotation value depending on raw value, see *'klib/kmath.h'*

2- Set image rotation from center point in the Z axis

Example references:

Function name	Description	Documentation
clutter_group_new	Initialize Clutter library	docs/clutter/ClutterGroup.html
clutter_texture_new_from_file	Load a PNG file into Clutter texture	docs/clutter/ClutterTexture.html
clutter_container_add_actor	Add texture to main screen	docs/clutter/ClutterContainer.html
kmath_interpolation_guint16_gint16	Translate and interpolate raw value to a rotation point	docs/klib/kmath_8h.html
clutter_actor_set_z_rotation_from_gravity	Display main screen	docs/clutter/ClutterActor.html

7.2.7 Animations

Clutter provides an easy way to make 3D animation

Main reference: docs/clutter/ClutterAnimator.html

Example: from '*examples/round_gauge/animations.c*', here is a simple scale in and out animation:

```
1  animation = clutter_animator_new();

   /** Set animation duration */
2  clutter_animator_set_duration(animation, interval);

   /** Set scaling in the center */
3  clutter_actor_set_scale_with_gravity( actor,
                                         0,
                                         0,
                                         CLUTTER_GRAVITY_CENTER
                                        );

   /** Scale up and down actor */
4  clutter_animator_set( animation,

                         actor, "scale-x", CLUTTER_LINEAR, 0.0, 1.0,
                         actor, "scale-y", CLUTTER_LINEAR, 0.0, 1.0,

                         actor, "scale-x", CLUTTER_LINEAR, 0.5, 0.5,
                         actor, "scale-y", CLUTTER_LINEAR, 0.5, 0.5,

                         actor, "scale-x", CLUTTER_EASE_OUT_ELASTIC, 1.0, 1.0,
                         actor, "scale-y", CLUTTER_EASE_OUT_ELASTIC, 1.0, 1.0,

                         NULL
                        );

   /** Start animation */
5  clutter_animator_start(animation);
```

Figure 12 Clutter animation scale in and out

Step-by-step:

1- Allocate a new animation

2- Set animation duration in ms

3- Set scaling from the center of the image

4- Define animation:

- At animation start time 0.0, set X and Y scales to original scales: 1.0
- At animation middle time 0.5, set X and Y scales to half: 0.5
- At animation end time 1.0, set X and Y scales to original scales: 1.0

4- Start animation

Example references:

Function name	Description	Documentation
clutter_animator_new	Create a new animation	docs/clutter/ClutterAnimator.html
clutter_animator_set_duration	Set animation duration	docs/clutter/ClutterAnimator.html
clutter_actor_set_scale_with_gravity	Set scaling point in the middle of the image	docs/clutter/ClutterActor.html
clutter_animator_set	Define animation	docs/clutter/ClutterAnimator.html
clutter_animator_start	Start animation	docs/clutter/ClutterAnimator.html

7.2.8 Additional Clutter resources

- https://clutter-project.org/.
- http://developer.gnome.org/clutter/1.8/
- http://tuxradar.com/content/clutter-beginners-tutorial
- http://www.openismus.com/documents/clutter_tutorial/0.8/docs/tutorial/html/

7.3 GLib

GLib is a general-purpose and portable utility library, which provides many useful data types, macros, type conversions, string utilities, file utilities, thread, etc. for C programming. KAntrak 3700 SDK and Clutter uses GLib for all data types. We encourage using Glib to facilitates your application development.

Supported version API documentation can be found into the '*docs/glib*' subfolder of the SDK: docs/glib/index.html

7.3.1 Main functionalities

Name	Description	Documentation
GType	Portable types	docs/glib/glib-Basic-Types.html
GList/GSLit	Generic single and double linked list methods	docs/glib/glib-Singly-Linked-Lists.html docs/glib/glib-Doubly-Linked-Lists.html
GTimer	Timer methods	docs/glib/glib-Timers.html
GFile	File reading/writing methods	docs/glib/glib-File-Utilities.html
GHastable	Hashtable methods	docs/glib/glib-Hash-Tables.html
GThread	Thread methods	docs/glib/glib-Threads.html
GKeyFile	.INI configuration file methods	docs/glib/glib-Key-value-file-parser.html
GSlice	Memory allocator methods	docs/glib/glib-Memory-Slices.html

7.3.2 Additional GLib resources

- http://developer.gnome.org/glib/2.28/index.html

8. KAntrak 3700 application

8.1 Creating a new application

1- In Code::blocks, from the File ->New->Project menu, select 'Kantrak 3700 Project':

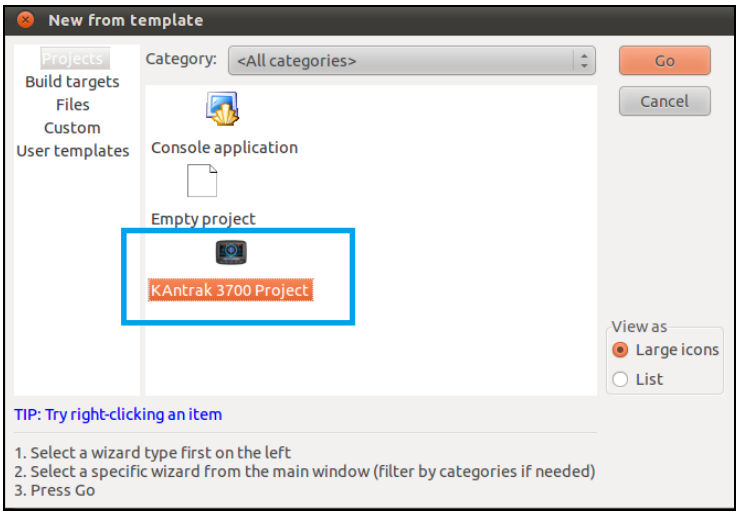

Figure 13 Creating a new KAntrak application

2- Follow on screen instructions to create application

IMPORTANT NOTE: All projects and source code should be created and saved into your shared folders on your host PC *(/media/sf_FolderName/XXXX*). This will avoid any data lost when updating Kongsberg KAntrak 3700 SDK virtual machine

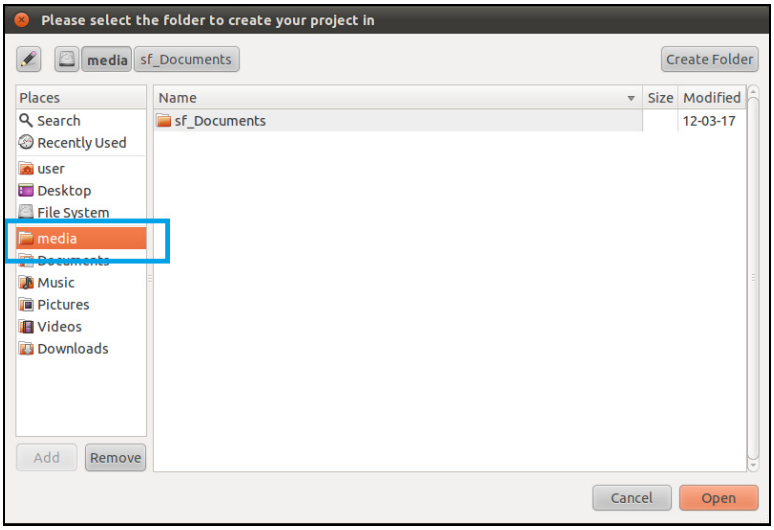

Figure 14 Media folder

Code::blocks build targets:

Build target	Description
Debug KAntrak Simulator	Run locally within KAntrak 3700 simulator with debug symbols and without optimisation
Release KAntrak Simulator	Run locally within KAntrak 3700 simulator without debug symbols and with optimisation
Debug KAntrak 3700	Run on KAntrak 3700 hardware with debug symbols and without optimisation
Release KAntrak 3700	Run on KAntrak 3700 hardware without debug symbols and with optimisation

8.2 Data folder

All data for application like gauge images, screen background, icons and configuration files ***must*** be located into the *'data'* folder contained of the code project in order to be installed on KAntrak.

- This folder is automatically created when using the KAntrak 3700 project wizard.
- Using build targets *'Debug KAntrak 3700'* and *'Release KAntrak 3700'*, this folder and all its subfolders will be copied into KAntrak 3700 storage and available for your application
- In your code, to refer to your files, uses relative path *'./data/XXXX'*

8.3 Simulator

The KAntrak 3700 simulator starts automatically when running target is *'Debug KAntrak Simulator'* or *' Release KAntrak Simulator'*

Figure 15 KAntrak 3700 simulator

- Mouse click on simulator keypad button will emulate KAntrak 3700 keypad events for button 1 to 5
- Numbers 1 to 5 on your PC keyboard emulate KAntrak 3700 keypad events for button 1 to 5

9. KAntrak 3700 application examples

Into the SDK root folder, you will find a folder name *'examples'* with KAntrak 3700 application examples. Open it and run it via your shared folder.

Example name	Description
image	• Demonstrate how to load an image, center and display it • Demonstrate KAntrak display usage
round_gauge	• Demonstrate KAntrak display usage • Demonstrate Keypad usage • Demonstrate Keypad LED usage • Demonstrate on to display text • Demonstrate how to add and place a gauge image into screen • Demonstrate how to set background color and image • Demonstrate how to animate gauge (scaling, rotation, moving, opacity)

10. Licenses

10.1 Clutter

10.1.1 Software

GNU Library General Public License (GNU LGPL) 2.1: http://www.gnu.org/licenses/lgpl-2.1.html

10.1.2 Documentation

10.2 GLib

GNU Library General Public License (GNU LGPL): http://www.gnu.org/licenses/lgpl.html