Q1 - Embedded GUIs with Qt

Embedded GUIs with Qt

Objectives

- Installing Qt 5 and the needed components on an embedded platform
- Writing Qt applications
- Discovering Qt key components
- Understanding the proper use of threads in Qt applications

Labs are conducted on target boards, that can be:

Quad Cortex/A9-based "Sabre" boards from NXP, with Lauterbach JTAG probes.
Dual Cortex/A9-based "Panda ES" boards from Texas Instruments, with Lauterbach JTAG probes.
Atmel ARM9-based boards, with Lauterbach JTAG probes.

Course environment

- Printed course material (in English)
- One Linux PC for two trainees.
- One target platform (i.MX6 Sabre or PandaBoard ES) for two trainees

Prerequisite

- Good knowledge of C++ language

Environnement du cours

- Cours théorique
  - Support de cours imprimé et au format PDF (en anglais).
  - Le formateur répond aux questions des stagiaires en direct pendant la formation et fournit une assistance technique et pédagogique.
- Activités pratiques
  - Les activités pratiques représentent de 40% à 50% de la durée du cours.
  - Elles permettent de valider ou compléter les connaissances acquises pendant le cours théorique.
  - Exemples de code, exercices et solutions
  - Un PC (Linux ou Windows) par binôme de stagiaires (si plus de 6 stagiaires) pour les activités pratiques avec, si approprié, une carte cible embarquée.
  - Le formateur accède aux PC des stagiaires pour l'assistance technique et pédagogique.
- Une machine virtuelle préconfigurée téléchargeable pour refaire les activités pratiques après le cours
- Au début de chaque demi-journée une période est réservée à une interaction avec les stagiaires pour s'assurer que le cours répond à leurs attentes et l'adapter si nécessaire

Audience visée

- Tout ingénieur ou technicien en systèmes embarqués possédant les prérequis ci-dessus.

Course Outline

FIRST DAY

Introduction

- History
- Versions
- Licenses
- Components
- Qt Quick

Qt basics

- Hello world application
- Compiling and cross-compiling a Qt application
- The main mechanisms of Qt
  - MOC (Meta Object Compiler)
  - Main loop
  - Signals and slots
  - Introspection
  - Asynchronous calls
- Creation of a Qt application on Linux
  - Creation of Qt projects
  - Compiling
- Main classes
  - Base classes
  - Main widgets
  - Utilities
- The development tools for Qt
  - QT Designer
  - QtCreator
  - Qmake

Exercise : Hello world application

Qt widgets

- Basic widgets
  - Labels, buttons, …
- Layouts
- Dialogs
  - Custom dialogs
  - Standard dialogs

Exercise : Writing an application combining various widgets
Exercise : Writing a custom dialog

SECOND DAY

Threading

- Threading model
- Launching a worker thread
- Synchronization
- Queuing work to the GUI thread

- Timers

Exercise : Writing a multi-threaded application

Custom widget

- 2D drawing
- Handling mouse, touch screen and keyboard events

Exercise : Moving an image on screen

Model/view

- Model/View concept
- Model/View widget vs Standard Widget
- Standard models
- Writing a custom model
- Views

Exercise : Using a QListView

THIRD DAY

Install

- Low-level graphism
  - Frame buffer
  - Open GL ES and EGL
  - X Server
  - Wayland
- Qt platform plugins
  - EGLFS
  - LinuxFB
  - XCB (X server)
  - Wayland
- Low-level input subsystem
  - Input drivers
  - Tslib
  - Multi-touch protocol
- Configuring input in Qt
- Cross-compiling and installing Qt
  - Build system
  - Main options

Exercise : cross-compiling and installing Qt5 on an embedded board

Open GL ES 1.1 and 2.0

- Presentation of Open GL
  - Various Open GL versions
- Base concepts
  - Notion of state in OpenGL
  - Vertices and Triangles
  - Transformations
  - Drawing
  - Textures
  - Shaders
- OpenGL and OpenGL/ES
- Drawing in OpenGL
  - Vertices and index arrays
  - Drawing items

    - Projections
    - Viewpoints
- Transformation matrixes
    - Loading and initialization
    - Translations and rotations
    - Save and restore (Matrix stack)
- OpenGL shaders
    - OpenGL Shading Language (GLSL)
    - Vertex shaders
    - Fragment shaders
    - Applying transformations
- OpenGL rendering model
    - Surfaces
    - Rendering operators
    - Offscreen rendering
- OpenGL in Qt 5
    - Raw Opengl with Qt OpenGL module (QGLWidget class)
    - Qt wrappers above OpenGL ES of Qt GUI module (QOpenGLContext)

Exercise : Rotating cube

FOURTH DAY

Multimedia support

- Multimedia in Linux
    - Gstreamer
- Multimedia in Qt 5
    - QtMulimedia module
    - Audio, Video Camera, MediaPlayer and Radio

Exercise : Playing a video file

Qt Quick

- QML
    - Elements and properties
    - Javascript
- Main elements and properties
    - User input, state, model and views, …
- Interactions between C++ and QML

Exercise : Hello world application with Qt Quick