Developing the Indicators Framework for Creating Display Systems

Alisa Volk, Vera Ivanova, Alexey Syschikov,
Boris Sedov
State university of aerospace instrumentation
St. Petersburg, Russia
volk.alisa.v@yandex.ru

***Abstract*–Designing displays for embedded systems can be complicated. It requires the involvement of developers with programming skills in low-level languages.The API developed by the authors simplifies the design for creating and using devices, which requires less time to design indicators displays and can be performed by those specialists who do not code. The functionality of the created software includes support for rendering on the GPU, and the ability to control devices in runtime.**

I. INTRODUCTION

Industrial display modules (or indicator displays) usually show various information about the system in real time – from specific parameters like velocity or altitude to energy consumption and general health of the system. Industrial displays are often embedded systems interconnected with other modules. The displays allow monitoring car, railway or aircraft systems in real time and provide vital information for managing industrial facilities. Given the specifics of the domains, these indicators should have increased reliability and meet the requirements for safety-critical systems.

Industrial indicators as embedded systems have following features: increased reliability, operation in real-time, performing in a strictly defined time intervals. Besides, embedded devices are often designed as self-powered devices and their energy consumption is strictly limited. Hence, to develop embedded software we need specialized industrial application programming interfaces (APIs).

II. API FOR INDUSTRIAL INDICATORS

OpenGL ES is a multi-platform low-level API for rendering advanced graphics using GPU. It is designed specifically for embedded systems [1]. Writing programs using OpenGL ES is a complicated process for highly qualified specialists. In order to create a set of graphics for an indicator display system, the specialist should have a good sense of technical specifications of the system and the basics of display design. The specialist also should have programming skills in low-level and shader languages (special programs written for GPU) and the knowledge of computer graphics basics.

OpenGL instructions are abstract and not tied to any particular device.

Nowadays, industrial display development is mostly done manually. A set of indicators for each system is created by computer graphics specialists. Each simple element like scale or arrow is described by a large number of OpenGL ES primitives, thus developing graphics with the OpenGL ES API can be very challenging.

Authors propose an indicator framework as a solution to simplify the process of industrial indicator development.

III. INDICATORS FRAMEWORK

An indicator display is designed in web visual editor by creating set of devices, each consisting of a background element, mask determining the shape of the device, and smaller components bound to background element or a mask.

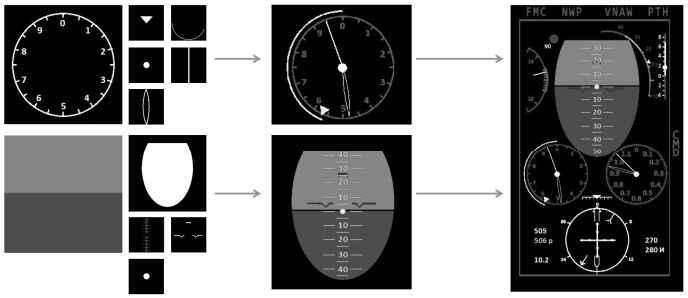

Fig. 1. Concept of the display design process

When the designing of elements on the display is completed, the developer can control display elements through a VIPE program.

VIPE (Visual programming environment) allows designing a fully functional, proved and algorithmically correct platform-independent program multicore/manycore embedded platforms, heterogeneous ones included [2].

VIPE has special visual DSL library (change parameters library in the fig below) for indicator displays that provides a number of functions to manipulate display primitives, such as: setting given coordinates, rotation at a given angle, changing radius, color or texture, changing text etc.

ISSN 2305-7254

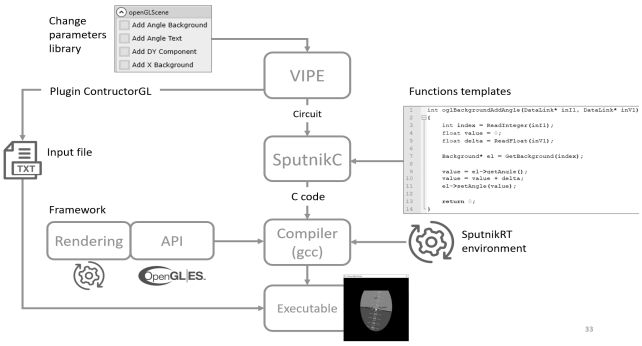

Fig. 2. Framework integration in VIPE

In addition, the developer can obtain data from peripheral devices (like gyroscope) using VIPE IO DSL library. The program shown below gets axis deviations from gyroscope sensor and changes the appearance of the indicator display to show the deviations on the screen.

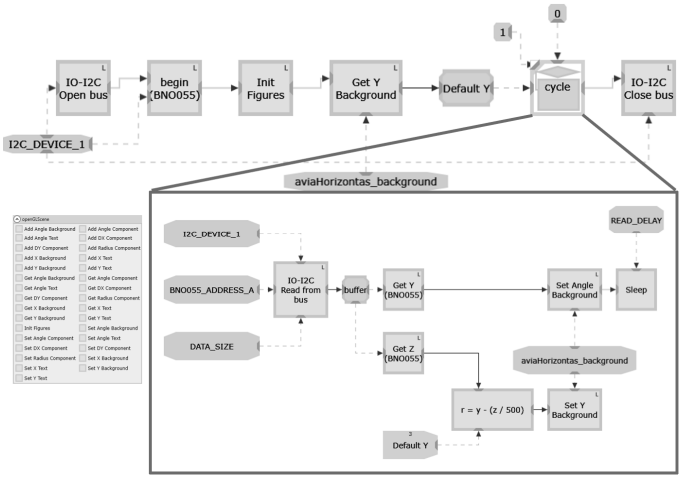

Fig. 3. VIPE program example with data coming from external sensor

Use case of the primary flight display was rendered on the Salyut-EL24PM1 processing module with the MALI-300 graphic core at the frequency of 70-75 fps [3]. On a Raspberry PI3 with Broadcom VC4 Graphics Core the same display was rendered with 75-80 fps.

IV. Conclusion

The authors developed a programming interface for creating graphic software for indicator displays. This interface provides methods to construct complex displays from a set of graphic primitives and render them on embedded GPU without writing an actual code; therefore, it does not require programming skills to design displays. In addition, this interface allows manipulating the elements of display in a runtime.

The developing process is simplified by moving it to the visual editor where user can drag and drop graphic primitives to construct an indicator. Graphic primitives include scales, pointers, text elements, arrows etc. Each primitive has an API to connect it with a dataset in runtime – for example, primitive can change angle depending on external value from sensor or computed value in program.

References

[1] H. Lee, N. Baek, “Implementing OpenGL ES on OpenGL”, *IEEE 13th International Symposium on Consumer Electronics,* 2009

[2] A. Syschikov, Y. Sheynin, B. Sedov, V. Ivanova, “Domain-specific programming environment for heterogeneous multicore embedded systems”, *International Journal of Embedded and Real-Time Communication Systems (IJERTCS)*, vol. 5(4), 2014, pp 1-23.

[3] Electronic VLSI engineering and Embedded Systems: Processor module Salyut-EL24PM.
Web: http://multicore.ru/index.php?id=1389