DEMO: Power Management using Game State Detection on Android Smartphones

Benedikt Dietrich, Samarjit Chakraborty
Institute for Real-Time Computer Systems, TU Munich, Germany
{benedikt.dietrich, samarjit}@tum.de

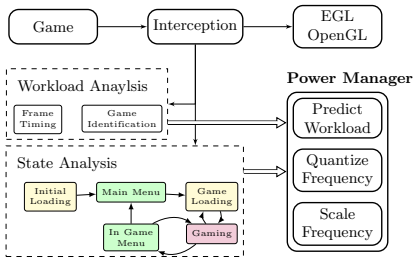

Figure 1: System Architecture Overview

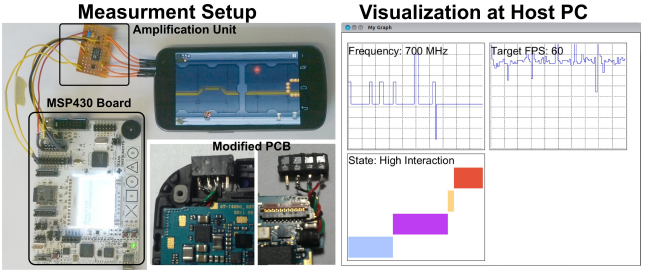

Figure 2: Demo Setup

Categories and Subject Descriptors

D.4.8 [**Operating Systems**]: Performance Measurements, Modeling and Prediction

Keywords

Android; Power Management; Workload Prediction; Mobile Games

1. INTRODUCTION

Compute intensive games currently represent the class of most popular and at the same time most power consuming applications on mobile phones. To reduce the power consumption of games we have developed a game state specific power management technique. Games typically consist of several *states* such as the game loading, main menu, in-game menu and gaming state. Each of these states has its specific processing requirements, e.g., the game loading state is likely to be memory bound and menu scenes are less interactive than gaming states and hence do not require high frame rates to satisfy the user's perception. Our `game state specific` governor (i) recognizes these game states by intercepting and analyzing calls made by the game application to the graphics library, and (ii) exploits these state-specific characteristics to enable power management strategies targeted to these individual states at runtime. Thereby, we achieve significant power savings of up to 50.8 % compared to Android's default `interactive` governor. In the following we describe the system architecture and the setup used during the demo.

2. SYSTEM ARCHITECTURE

We detect the game's current state based on the textures used to render the current frame. To identify the textures we intercept the game's texture related calls to the OpenGL library like shown in Figure 1. The gained state information is forwarded to our `game state specific` governor. To reduce the processor's power consumption we lower the processor's voltage and frequency whenever possible. If the current state is a memory bound loading state, the governor selects the processing frequency that minimizes the time the CPU is waiting for the memory. During interactive states like the gaming or menu state the governor maintains a state specific frame rate r_{state}. To predict the next frame's workload any suitable predictor can be used as the prediction is completely independent from the proposed technique to measure frame timings and detect states. In this work we predict the next frame's workload based on the timing of previous frames using an Autoregressive model. The frame timing of previous frames is measured by intercepting the `eglSwapBuffers()` call to the EGL library which is made by the game at the end of every frame. Based on the prediction result the governor selects a CPU frequency that allows finishing the processing of the next frame within $1/r_{state}$. As both, the state detection and the profiling mechanism are based on instrumenting Android's graphics libraries the game's source code is not required and hence the method can be applied to any Android based game.

3. DEMO SETUP

The demo setup consists of a Samsung Galaxy Nexus which has been modified to allow accurate CPU power measurements, a MSP430 based power measurement device and a monitor for live visualization purposes. During the demo we plan to invite the audience to play games like Cut the Rope, Jetpack Joyride or Temple Run using either the default `interactive` governor or our `game state specific` governor. While playing, the game's current state and the used processing frequencies are plotted. After the game the power measurement results for each game state and the total average power consumption are shown (see [1] for a video). This allows the audience to compare the performance of the two governors in terms of power savings and perceived gaming quality.

4. REFERENCES

[1] *Demo Video of Power Management using Game State Detection (http://www.rcs.ei.tum.de/en/staff/dietrich).*

MobiSys'13 June 25-28, 2013, Taipei, Taiwan
ACM 978-1-4503-1672-9/13/06.