ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Integrated Systems Laboratory

Group Project at the
Department of Information Technology and
Electrical Engineering

Towards the Ariane Desktop: Display Output for Ariane on FPGA under Linux

Students: Aron Szakacs
Toni Tanner

Advisors: Georg Rutishauser
Nils Wistoff

Professor: Prof. Dr. Luca Benini

Handout Date:
Due Date:

1 Introduction and Project Goals

1.1 Introduction

Ariane is an open-source, general-purpose RISC-V CPU architecture which implements the RV64GC instruction set and is capable of running Linux. Intended as a research and demonstration platform, it has been taped out in several incarnations and can be mapped to an FPGA device for low-cost evaluation and testing. The supported target for FPGA emulation is the **Genesys II** board, and the mapping flow and auxiliary hardware environment provided by the Ariane maintainers offers a rich feature set, making it possible to run RISC-V programs on custom hardware in real time.

On Ariane, user interaction with the programs running on the CPU was until recently limited to an UART link, which only allows for text input and output and requires a computer to be connected. In a previous project, a display peripheral (called PAPER) was integrated into the FPGA platform and tested.

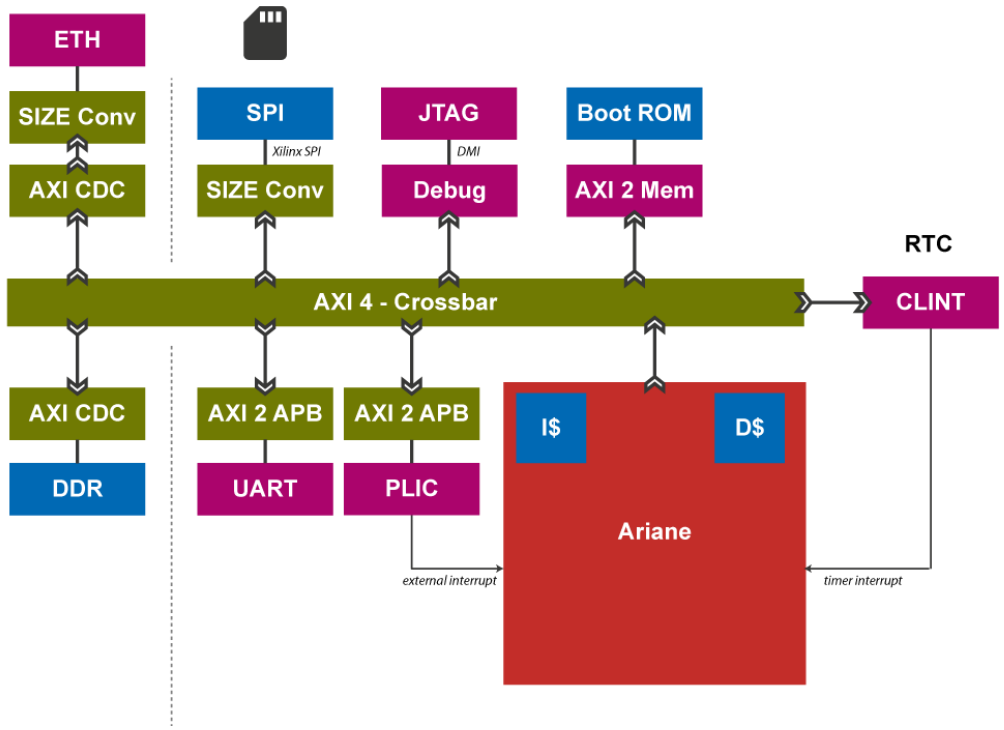

Figure 1
Ariane system for mapping on FPGA, not including PAPER

PAPER is an AXI-compliant peripheral module which supports the streaming of image (framebuffer) data from memory, as well as possessing the capability of streaming a text buffer in VGA text format – such as the Linux Virtual Terminal's internal text representation – from memory and graphically rendering the text it

contains. While PAPER has been successfully integrated into the FPGA platform and tested using simple bare-metal examples, driver support under Linux has not yet been implemented, and the current hardware implementation still has a prototype character.

The overall goal of this project is to take a step towards a fully-featured autonomous Linux system based on Ariane with extensive user interaction support. To this end, you will work to provide usable display output, directly from the FPGA's HDMI port, on the FPGA port of Ariane. This will involve both hardware design tasks to improve maturity and extend the current feature set, as well as software development in the form of Linux driver development.

In this project, you will learn to work with and extend an advanced processing system all the way from the RTL/hardware level to the Linux kernel and userspace levels. This will provide you with unique insights into the workings of such a platform, and these insights will apply readily to the Linux systems you will encounter in your academic, professional and private lives. You will learn how to hardware peripherals are designed and integrated into processing systems, as well as how to establish software support for them both in bare-metal applications and under Linux.

1.2 Hardware Design Tasks

- **Support for higher resolutions:** Due to the current structure of the AXI infrastructure, output resolutions are limited to a maximum of SVGA (800x600) at a 60 hertz refresh rate. By refactoring the AXI infrastructure, you will eliminate this restriction and allow resolutions of Full HD and beyond.
- **Resource reclamation:** Currently, FIFOs are implemented in HDL, which results in relatively high flip-flop utilization. By replacing them with Xilinx primitives, the FF utilization of the peripheral can be reduced by ~66%.
- **Support for Runtime-Configurable Resolution:** Currently, the frequency of the output pixel clock is fixed at compile time. In order to make the resolution runtime-configurable, you will replace the fixed-frequency clock generator with a configurable one. This will make it possible to change the output resolution while the system is running and without requiring time-intensive re-implementation.
- ***Feature Extensions:*** *As an example for an optional goal, the PAPER module's functionality could be extended to allow for runtime-customizable fonts in text mode.*
- ***Input Peripheral Support:*** *In order to completely remove dependence on serial UART input, support for keyboard input over PS/2 or USB could be implemented. This may be realized with a dedicated hardware peripheral (re-using an existing*

open-source core is the most time-effective option), or by using GPIO pins to "bit-bang" the PS/2 protocol. The Genesys II board has both a dedicated USB HID host which converts USB to PS/2 and a general-purpose USB 2.0 controller which is attached to the FPGA with an ULPI interface.

- **Verification:** The assembled platform must be thoroughly tested to ensure complete and correct functionality. Furthermore, it must be ensured that the modifications don't impact the system's critical path.

1.3 Software Design Tasks

- **Bare-Metal Driver Adaptation:** The existing bare-metal software framework is kept as platform-agnostic as possible, but certain specifics must be adapted to correspond to the specific platform. The goal is to achieve an easy-to-use library which provides user-friendly support for both image and text display in bare-metal applications.
- **Linux Device Driver:** In order to interact with the PAPER peripheral under Linux, it must be abstracted by a simple device driver kernel module. This driver will allow higher-level drivers (e.g. a console driver) to interact with the peripheral through a unified interface.
- **Linux Console Driver:** A prototype console driver for Linux exists, which allows the output of the linux console to be displayed from the first line on PAPER's original target platform, the ZEDBoard. The objective is to develop this driver to provide a more streamlined integration into the Linux kernel and to port the resulting software to source tree for the Linux distribution targeting Ariane.
- ***Linux display driver:*** *In order to provide graphical output under Linux, a framebuffer driver or a KMS/DRM display driver can be implemented.*
- ***Input Peripheral Support:*** *If support for PS/2 keyboard input is implemented, this will require software support under Linux.*

2 Tasks

The project will be split into four distinct phases, detailed below. Software-focused tasks are colored green, hardware-focused tasks are colored blue. Tasks in black are applicable to both software and hardware focus.

Phase 1: Familiarization and Planning (3-4 weeks)

To familiarize with the project's subject matter and establish a plan for the project's successful execution, the students will perform the following steps:

1. Reading the documentation for PAPER, Ariane and the Genesys II FPGA board, taking note of the aspects relevant to each sub-task (e.g., how is PAPER connected to the system? How is it configured for different resolutions?)
2. Implementing the baseline platforms of Ariane with PAPER integrated and running example software on this platform to get acquainted with the operation of the baseline hardware.
3. Compilation of the baseline Linux system and booting your very own "Linux from Source".
4. Familiarization with the Ariane-on-FPGA system's layout. This will involve getting familiar with the AMBA AXI protocol and exploring the structure of the peripheral system and the AXI infrastructure.
5. Familiarization with the fundamentals of digital video signals. After this step, you should be able to answer the following questions:
    a. What does a raw digital video signal look like? What is the function of the control signals HSYNC, VSYNC and DE?
    b. What information is required to completely specify an output resolution and refresh rate?
6. Creation of high-level block diagrams for the hardware modifications:
    a. Refactoring of AXI infrastructure
    b. Addition of configurable pixel clock generator
    c. Support for runtime-configurable fonts
7. Familiarization with the details of Linux peripheral management, the Virtual Terminal infrastructure and the baseline console driver (`gcon`)

Phase 2a: Hardware Design and Implementation (6-9 weeks)

In this phase, the necessary modifications and extensions to the existing hardware are planned, implemented and tested. **Before and after each modification, you should take note of the resource usage and timing characteristics of the design, so you can track the impact of your changes! This is often neglected or forgotten in the "heat of the battle", but will make for extremely valuable content for your report and presentation.**

1. **PAPER resource reclamation:** As a first step in hardware development, you will replace the flip flop-based FIFOs by Xilinx FIFO primitives. This replacement should be parametrizable, i.e. the designer should be able to switch between instantiation of FF-based FIFOs (for use in ASIC designs) and Xilinx FIFO primitives.
2. **Support for Runtime-Adjustable Resolution:** In this task, the fixed-frequency pixel clock generator (a Xilinx IP) will be reparametrized to be runtime-reconfigurable. The configurable clock generator will then be attached to the AXI bus, so the system user can reconfigure the pixel clock (and with it, the output resolution) at runtime.

3. **Support for Higher Resolutions:** This task involves refactoring the AXI infrastructure to support pixel clocks at higher frequencies than the system bus clocks.
4. ***Support for Runtime-Configurable Fonts:*** *The font memory in the PAPER module contains the pixel-by-pixel representations of the glyphs with which text buffers are rendered in text mode. Its contents are currently hard coded. In this step, you will attach the Font memory to the AXI infrastructure, so the font face can be changed at runtime.*
5. ***Input Peripheral:*** *If there is enough time, implement a PS/2 hardware peripheral to accept keyboard input directly to the FPGA board, or remap Ariane GPIO pins to enable bit-banging the PS/2 protocol in software via GPIO pins.*

Phase 2b: Driver Support (6-9 weeks)

1. **Bare-Metal Drivers:** Using the existing cross-platform driver framework, a user-friendly library for text and image display is created. This library will provide functions to:
    a. Configure the PAPER hardware (operating mode, display mode, etc)
    b. Display and manipulate text buffers interactively
    c. Display framebuffers stored in the main memory

    This functionality is already almost complete, but will require some verification and debugging.
2. **Linux device driver:** To be able to systematically interact with the PAPER peripheral from within the Linux kernel, you will implement a simple device driver with analogous functionality to that of the bare-metal drivers. This will also require the integration of the Xilinx Clock Wizard Linux driver in order to configure the pixel clock.
3. **Linux Console Driver:** Starting from the prototype console driver implemented for the ZEDBoard Linux, a more sophisticated version of this driver is developed and integrated into the Ariane Linux source tree. The goal is to keep the disturbance of the upstream kernel source code to a minimum and conform to Linux contribution guidelines.
4. ***Linux Display Driver:*** *To enable graphical output using PAPER's video mode, you will implement either a basic framebuffer driver or a simplified KMS/DRM driver supporting PAPER.*
5. ***Input Peripheral Support:*** *If direct keyboard input is to be supported, this will require driver support – either in form of a kernel module supporting a dedicated hardware PS/2 peripheral, or the integration and activation of the* `ps2-gpio` *driver into the kernel source tree.*

Phase 3: Conclusion (3-4 weeks)

1. Documentation and cleanup of code
2. Compilation and analysis of results:
    a. What impact did your hardware changes have on the critical path (if any)?
    b. How did resource utilization change as a result of your changes?
3. Writing and finalization of report
4. Preparation of the final presentation (15 minutes + 5 minutes Q&A)

3 Milestones

By the end of **Phase 1** (2-3 weeks after project start) the following should be completed and presented to the supervisors:

- Running example programs on the Ariane/PAPER baseline hardware platform: Displaying images with the provided software
- Successfully booting self-compiled Linux on Ariane
- High-level block diagrams for the extensions to PAPER and the integration into Ariane

By the end of **Phase 2** (9-12 weeks after project start) the following should be completed:

- Upstreamed modifications to PAPER:
    - Parametrized instantiation of FPGA FIFO primitives
    - *Support for runtime-reconfigurable fonts*
- Fully verified modifications to the Ariane FPGA System:
    - Runtime-adjustable output resolution/pixel clock
    - AXI infrastructure refactoring to support higher resolutions/pixel clocks
    - *Support for keyboard input*
- Working example program demonstrating both image and text display mode as bare-metal code
- Linux device driver enabling interaction with PAPER from Linux kernel space
- Linux console driver integrated in a fork of the source tree, allowing to display the boot console in runtime-configurable resolutions and text buffer sizes directly from FPGA

- *Linux Display driver allowing the user to display images via a framebuffer device or KMS/DRM*
- *Software/driver support for keyboard input*
- Demo applications showcasing the implemented features

By the end of **Phase 4** (14 weeks after project start) the following should be completed:

- Finalisation of the software and hardware documentation
- Final presentation
- Final report, including final results.

4 Project Organization

4.1 Weekly Report and Weekly Meeting

At weekly meetings with the supervisors, project progression and next steps can be discussed.

4.2 Final Report

A pdf copy of the report has to be turned in and remains the property of the Integrated Systems Laboratory. A copy of the developed software and hardware code needs to be stored on the assigned account's home folder.

4.3 Final Presentation

At the end of the project, the outcome of the thesis will be presented in a 15-minute presentation with an additional 5 minutes allotted to a question and answer session, again during a group meeting of the Integrated Systems Laboratory. The presentation will be held using an FPGA platform with the PAPER hardware instantiated to display the slides.

5 Resources and Links

5.1 Ariane Project

- Project Repository: https://github.com/openhwgroup/cva6
- Core Documentation: https://cva6.readthedocs.io/en/latest/

5.2 PAPER

- Development Repository – use this for simulation and reproduction of the original system: https://iis-git.ee.ethz.ch/georgr/master_thesis

- Hardware Repository – to be used when PAPER is included in a different repository: https://iis-git.ee.ethz.ch/georgr/PAPER_hw
- Software/Driver Repository: https://iis-git.ee.ethz.ch/georgr/PAPER_sw
- Master Thesis containing documentation: https://iis-git.ee.ethz.ch/georgr/master_thesis_report
- Linux console driver source for original ZEDBoard PAPER integration: https://iis-git.ee.ethz.ch/georgr/gcon

5.3 Protocol Specifications & Documentation

- DVI: http://www.cs.unc.edu/Research/stc/FAQs/Video/dvi_spec-V1_0.pdf
- AMBA AXI4/AXI4 Lite: http://www.gstitt.ece.ufl.edu/courses/fall15/eel4720_5721/labs/refs/AXI4_specification.pdf
- Introduction to AMBA AXI4 Lite: https://www.realdigital.org/doc/a9fee931f7a172423e1ba73f66ca4081
- VESA Display Monitor Timings: http://caxapa.ru/thumbs/361638/DMTv1r11.pdf
- VGA Text Specification:
  - https://wiki.osdev.org/Text_UI
  - http://www.scs.stanford.edu/17wi-cs140/pintos/specs/freevga/vga/vgatext.htm
- PS/2 Protocol: https://www.avrfreaks.net/sites/default/files/PS2%20Keyboard.pdf
- USB Transceiver Macrocell Interface: https://www.intel.com/content/dam/www/public/us/en/documents/technical-specifications/usb2-transceiver-macrocell-interface-specifications.pdf

5.4 Hardware Platform Documentation:

- ZEDBoard (PAPER development platform): http://zedboard.org/support/documentation/1521
- Genesys II (Ariane FPGA reference target): https://reference.digilentinc.com/reference/programmable-logic/genesys-2/start?redirect=1
- Report on integration of PAPER into Ariane: https://polybox.ethz.ch/index.php/s/GSDMzCAhz4o6M0w

5.5 Ariane / Linux Resources

- Linux port for Ariane: https://github.com/pulp-platform/ariane-sdk

- High-Level Info on KMS/DRM display drivers:
  https://events.static.linuxfound.org/sites/events/files/slides/brezillon-drm-kms.pdf
- Framebuffer driver development:
  http://ftp.osuosl.org/pub/nslu2/sources/cvs/linux-input/ruby/web/htdocs/fbdev/HOWTO/4.html
- Linux Virtual Terminals and console drivers:
  https://en.wikipedia.org/wiki/Virtual_console
  https://www.kernel.org/doc/html/latest/driver-api/console.html
  `vgacon` driver – `gcon` is based on this:
  https://elixir.bootlin.com/linux/latest/source/drivers/video/console/vgacon.c
- `ps2-gpio`: A PS/2 driver which implements the protocol in software, requiring only GPIO pins for PS/2 communication.
  https://github.com/torvalds/linux/blob/master/drivers/input/serio/ps2-gpio.c