ROBOTS IN ASSISTED
LIVING ENVIRONMENTS

UNOBTRUSIVE, EFFICIENT, RELIABLE AND MODULAR
SOLUTIONS FOR INDEPENDENT AGEING

Research Innovation Action

Project Number: 643892 Start Date of Project: 01/04/2015 Duration: 36 months

DELIVERABLE 4.5

Robust and Energy Efficient Hardware Components II

Dissemination Level	**Public**
Due Date of Deliverable	Project Month 30, 30 September 2017
Actual Submission Date	12 March 2018
Work Package	WP4*: Physical home architecture development and integration of cost-effective, reliable and power-efficient RADIO components for elder monitoring and caring*
Task	T4.2: *Embedded device design and development*
Lead Beneficiary	AVN
Contributing beneficiaries	TWG, RUB
Type	DEM
Status	Submitted
Version	Final

Project funded by the European Union's Horizon 2020 Research and Innovation Actions

Abstract

The deliverable is the final library of dedicated SW-HW components required to integrate the Zynq FPGA board to the on-robot processing unit, to setup the FPGA with a fully-functional embedded Linux distribution, to bring up the ROS operating system on top of the embedded Linux, and to build the framework for developing FPGA HW accelerators (and connect them efficiently with the ROS system).

History and Contributors

Ver	Date	Description	Contributors
00	1 Sep 2017	Document structure, assuming D4.4 as a starting point.	NCSR-D
01	25 Sep 2017	img_proc_accel: Hardware acceleration components for motion detection.	AVN
02	20 Feb 2018	Picozed_Python: direct connection between camera and FPGA	RUB
03	6 Mar 2018	img_proc_accel: Updates to the hardware acceleration components for motion detection.	AVN
04	8 Mar 2018	SDSoC: hardware implementation of vision-based ADL recognition algorithms.	TWG
05	9 Mar 2018	Internal review.	NCSR-D
06	10 Mar 2018	Addressed internal review comments.	TWG
Fin	12 Mar 2018	Final preparation and submission	NCSR-D

Abbreviations and Acronyms

SoC	System on Chip
ROS	Robotic Operating System
ADL	Activities of Daily Life
HDMI	High-Definition Multimedia Interface
DHCP	Dynamic Host Configuration Protocol

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Purpose and scope

This deliverable is a demonstration of the RADIO approach for increasing the power efficiency and robustness of the system by using dedicated hardware components to implement data processing methods from WP3. The scope of delivery is the source code (Section 2) and that this code has been physically prototyped on a PicoZed FPGA and evaluated. The evaluation results are reported in D4.3.

1.2 Approach

This deliverable is prepared within T4.3 that designs and prototypes embedded hardware components that implement visual processing methods. During the first phase (D4.4) the hardware architecture for accelerating image processing algorithms was improved, making more appropriate for image processing algorithms. Also, the Zynq FPGA board was integrated into the Linux/ROS framework of the RADIO Robot. Initial experiments were also conducted regarding the energy and run-time efficiency gained by accelerating the motion detection algorithms; these indicated that significant gains are expected by directly connecting the camera to the FPGA, bypassing the on-board CPU.

During the second phase (D4.5, this deliverable), work focused on: (a) directly connecting the camera to the FPGA; and (b) implementing and evaluating visual motion detection algorithms for ADL recognition using three different approaches/tools: hardware-only implementation based on Verilog HDL code, through Xilinx Vivado HLS (high-level synthesis), and through HW-SW partitioning using the SDSoC framework. In all cases, several performance optimizations have been performed in order to maximize the speedup of the provided algorithms.

Special emphasis was given to prepare a library of dedicated SW-HW components required to integrate the Zynq FPGA board to the on-robot processing unit and attach the camera and processing on the FPGA boards i.e., to enable the use of the PYTHON1300 FMC camera and the related HDMI input and output ports.

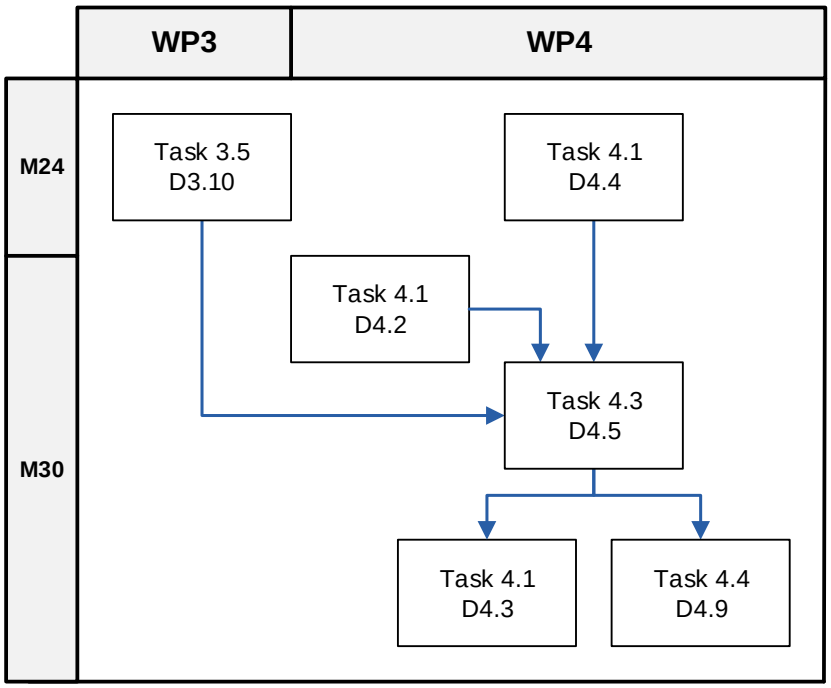

Figure 1: Relation to other Work Packages and Deliverables

1.3 Relation to other Work Packages and Deliverables

This deliverable is prepared following the *Architecture for extending smart homes with robotic platforms II* (D4.2). D4.5 provides efficient hardware implementations for the ADL recognition methods from WP3.

The robust and energy efficient hardware components described in this deliverable are evaluated in D4.3, the final report of WP4. The components are also used by *Task 4.4: Smart home design and integration* in order to prepare the *Integrated smart home with robotic platform* (D4.9).

2 PROTOTYPE

The hwaccel_radio folder in the git repository contains code that enables acceleration of RADIO functions in the Xilinx PicoZed with a Zynq-7000 all programmable System on Chip (APSoC) platform. The code can be user-space software code (C/C++ and Python), hardware code (verilog), and Linux kernel code (C/C++).

More specifically, the Picozed_RADIO and Picozed_Python subfolder have the code used for attaching the camera and processing on the FPGA board e.g., the implementation for video direct memory access (VDMA), the software that enables the use of the PYTHON1300 FMC camera and the HDMI input and output ports.

The SDSoC_design folder includes code that accelerates a Centre of Gravity algorithm for ADL detection using an optimal software-hardware partitioning approach.

The img_proc_accel folder contains four building blocks in verilog, three that accelerate various key image processing functions of generic usage and a hardware-only implementation of a monitoring system that triggers further analysis only if a significant activity is seen. In all cases, the PYTHON1300FMC camera sends its data directly to the programmable logic. The data is converted to the AXI STREAM protocol and then sent for further processing to the DDR memory. Then, the data is sent via the same VDMA to the HDMI output. Figure 2 depicts an example architecture of a FPGA HW accelerator generated using the HLS (high-level synthesis) approach.

Table 1 depicts the HW-SW projects that make up this deliverable. The changes with respect to D4.4 *Robust and Energy Efficient Hardware Components I* are also indicated.

Table 1: Code Repositories

Project	**Description**
In repository https://github.com/radio-project-eu/Zedboard_Linaro_Radio (from D4.4)	
	A fully functional Linaro embedded Linux distribution with support for HDMI screens, with keyboard and mouse for the FPGA Zedboards. Network connection is configured to DHCP protocol. Server-side ssh and sftp is also installed (username and password: linaro)
In repository https://github.com/radio-project-eu/hwaccel_radio (from D4.4)	
BOOT/	Bitstreams (FSBL, u-boot, FPGA bitstream) and Linux (Linaro) kernel for booting the Zynq-7000 all programmable System on Chip (APSoC) platform.
PicoZed_RADIO/	The SW, HW, and driver code used for attaching the robot's camera to the FPGA board. This approach uses the on-board CPU to feed images into the FPGA.
driver_test/	Source code (at Linux driver level) to test the correct and efficient (DMA transfers) communication between the PS (ARM cores) and PL (programmable logic) parts in the Picozed APSoC platform.
In repository https://github.com/radio-project-eu/hwaccel_radio (new in D4.5)	
img_proc_accel/	Source code in HDL (verilog) that contain four building blocks; three blocks that accelerate various key image processing functions of generic usage and a hardware-only implementation of a monitoring system that triggers further analysis only if a significant activity is seen (new in D4.5).
SDSoC_design/	Acceleration of the Centre of Gravity algorithm for ADL detection in the Picozed APSoC platform following an optimal hardware-software partitioning approach. The source code includes the components in hardware HDL (verilog), the software components (running on ARM), and the interconnection components (Linux drivers) required to synchronize the software and hardware parts.
PicoZed_Python/	The HW and driver code used for attaching ON Semiconductor's PYTHON-1300 color image sensor directly to the FPGA. The on-board CPU only receives the processing results.

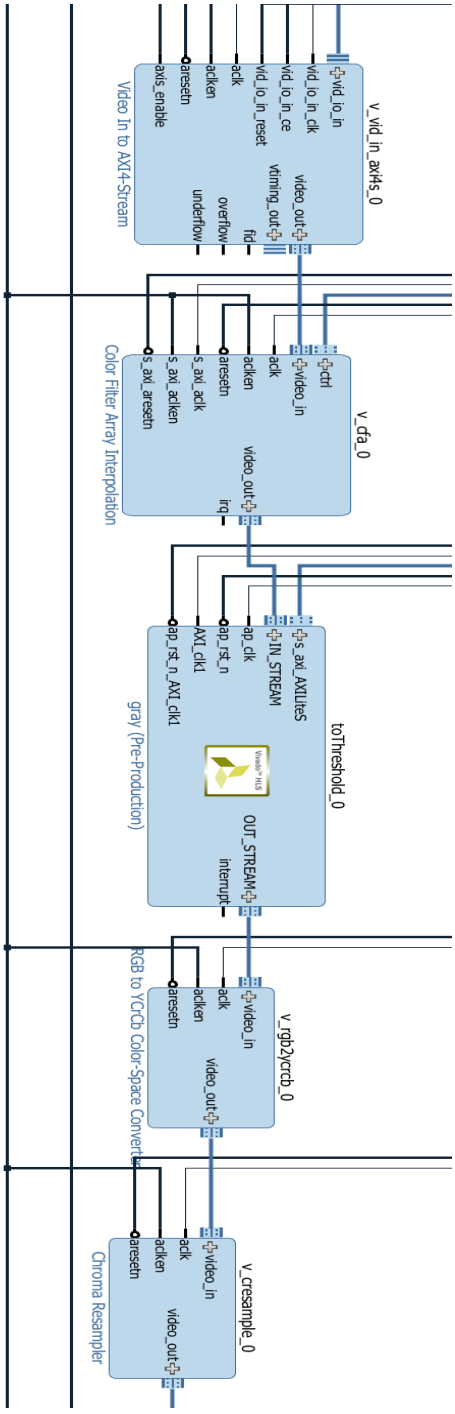

Figure 2: Example Architecture of a FPGA HW Accelerator generated using HLS techniques