

Marek Funtowicz

Embedded Software / FPGA Firmware Engineer

Personal Details

Address: Stanisława Ligionia 14/37
44-280 Rydułtowy

Phone: +48 724 235 978

E-mail: ice.marek@yahoo.com

Git: github.com/IceDefcon

Web: icedefcon.github.io

Education

2014 – 2017 The University of Sheffield
Electronics and Communication Engineering
Master of Engineering

2013 – 2014 Open Study College
Pure Mathematics
A-Level

2012 – 2014 Central College Nottingham
Electrical and Electronic Engineering
Higher National Diploma

Skill

FPGA: Intel, Xilinx

CPU: x86_64, ARM, PowerPC

RTL: Verilog, VHDL

Code: ASM, C, C++, Python,
Fortran

OS: FreeRTOS, VxWorks,
Linux, Autosar

Comms: USB, UART, SPI, I2C,
JTAG, GPIO, TCP, UDP,
CAN

Circuit: Mentor graphics, Spice,
Multisim, Altium

Test: Oscilloscope, Multimeters
Logic Analyzer

Matrix: Matlab, Octave

Web: HTML, PHP, CSS,
Java script, Bootstrap

DB: MySQL

Lang: Polish (Native),
English (Fluent)

Personal Research and Development

Research and Develop CPU & FPGA Computer Platform for Robotics (C, C++, VHDL, Python)

- ❖ x86 powered by Ubuntu
- ❖ ARM Cortex-A8 powered by Debian LNX
- ❖ Intel Cyclone IV FPGA for high speed computation

System specification:

- ❖ x86 → Graphical User Interface for the overall master control
- ❖ x86 → Network Client (Python GUI)
- ❖ LNX → Network Server (C++ Application)
- ❖ LNX → Char device for IO between Kernel and User space
- ❖ LNX → Block RAM device for chip configuration in FPGA
- ❖ LNX → SPI driver for bidirectional FPGA communication over DMA
- ❖ LNX → Bidirectional GPIO signaling for FPGA/Kernel ISR processing
- ❖ LNX → Work queues and kthread design techniques
- ❖ FPGA → Watchdog interface for Kernel synchronization
- ❖ FPGA → SPI interface for data serialization and parallelization
- ❖ FPGA → Parametrized I2C controller driven from LNX Kernel
- ❖ FPGA → Parametrized FIFO controller driven from LNX Kernel
- ❖ FPGA → Offload controller and packet switch for FIFO data flow
- ❖ FPGA → PWM controller for the motor throttle control
- ❖ FPGA → UART controller for the logs acquisition
- ❖ FPGA → PID controllers for feedback control :: Pending R&D
- ❖ FPGA → Gyroscope/Accelerometer devices for measurement acquisition

Platform will be upgraded to Xilinx Zynq UltraScale+™ MPSoC

- ❖ System will be powered by the Ultra96-V2
- ❖ ARM Cortex-A53, ARM Cortex-R5 and a Mali-400 MP2 GPU
- ❖ More powerful FPGA chip fully integrated with CPU over DMA
- ❖ The SPI interface for CPU<->FPGA comms replaced with a DMA
- ❖ Adaptation of the the current LNX kernel module to Xilinx system
- ❖ 40-pin general-purpose I/O and low-speed peripheral interfaces
- ❖ 60-pin high-speed connector (handling higher data rates interfaces)

Software Engineer**Prototyping of the new generation wireless charging systems in the automotive industry (ASM, C)**

- ❖ Reading/analyzing prototype PCB schematics
- ❖ Adaptation of the PCB hardware with the low level drivers
- ❖ Licensed code generation tools (Tresos, Da-Vinci, CANoe)
- ❖ BPP, EPP and MPP wireless charging protocols
- ❖ Backward compatibility of the existing stack with the latest generation of wireless charging protocol
- ❖ Low level coding and debugging on the live targets (ASM, C)
- ❖ Introduction to the software architecture design
- ❖ Coding standards (MISRA, HIS) in the automotive applications

Software Engineer**Research and design the components of LTE communication protocol**

- ❖ Physical L1 abstraction layer of an LTE protocol stack
- ❖ DMA communication with FPFA → Data and control
- ❖ Low-level DMA problems
- ❖ Low-level multithreading problems → Race conditions
- ❖ Low-level CPU stack problems (ASM) → Thread crashes
- ❖ Live kernel debugging @ VxWorks RTOS
- ❖ Code disassembly and investigation (ASM)
- ❖ Thread crash analysis → Intel and PowerPC registry dump analysis
- ❖ Analyzing data logs using custom made scripts
- ❖ Research and investigation of FPGA RTL
- ❖ Research and investigation of non-FPGA parts of the system → Higher layers of the LTE protocol, SCPI communication with RF boards
- ❖ Various number of tests → LTE protocol functionality

Embedded Systems Engineer**Design Firmware and RTL for Power Electronic Device using AURIX 32-bit Tri-Core Microcontroller and ARTIX 7 Xilinx FPGA (Soft Power Bridge)**

- ❖ Research the Infineon Low Level Drivers for the Tri-Core
- ❖ Develop Tri-Core Firmware drivers for ASCLIN (UART), QSPI and CAN peripherals to communicate with PCB
- ❖ Research Tri-Core Firmware for TCP/IP Stack (LWIP)
- ❖ Research ARTIX 7 RTL to communicate with CPU over QSPI and PCB peripherals
- ❖ Research Microblaze firmware in ARTIX 7 FPGA
- ❖ Investigate and analyze circuit schematics and PCB layout
- ❖ Test the behavior of the Firmware on the Tri-Core development board and the actual product PCB (Soft Power Bridge)

Junior Hardware Engineer**Design computer hardware for capturing, processing and streaming live video using Intel Altera FPGA technology and firmware control**

- ❖ Design FPGA hardware modules in VHDL and Verilog
- ❖ Design QSYS networks for Nios II processing
- ❖ On-chip memory management for Nios II processing
- ❖ Test RTL modules with Modelsim
- ❖ Test and debug RTL modules with signal tap
- ❖ Test and debug RTL modules with live PCB
- ❖ Test functionality of live PCB with measurement equipment
- ❖ Design Nios II firmware for graphical signal processing in FPGA
- ❖ Design Nios II firmware for peripheral devices mounted on PCB
- ❖ Test and debug Nios II firmware on live FPGA using Eclipse
- ❖ Integrate firmware with BSP and HAL drivers for Intel IP definitions
- ❖ Investigate booting codes for embedded Intel Nios II processor
- ❖ FPGA pin planning → IO Banks, Voltages, Transceivers
- ❖ FPGA chip planning → Module's connectivity and Logic congestion
- ❖ Timing analysis for RTL → Clock Cross Domain, False paths
- ❖ Investigate and analyze circuit schematics and PCB layout
- ❖ Integrate FPGA hardware code and CPU firmware with the circuit schematic and PCB Layout

Prototyping embedded graphical processing system using Xilinx FPGA technology and Linux firmware control

- ❖ Develop booting sequence for Zynq Ultrascale MPSoC in QSPI → hand over FPGA control to Linux kernel via U-Boot
- ❖ Research bare metal codes → PMUFW, FSBL, ATF
- ❖ Research and modify U-boot source → Configure MAC addressing
- ❖ Research bare metal XEN Hypervisor → Register XEN watchdog
- ❖ Develop UART communication for Zynq Ultrascale MPSoC → register UARTLITE driver in Linux to communicate with external chip for TMDS processing and HDCP control
- ❖ Research kernel → Version control
- ❖ Research device tree → Petalinux overlay
- ❖ Research root file system → Network configuration
- ❖ Customize Petalinux → Kernel, Root file system and device tree
- ❖ Debug kernel in OS awareness mode → Eclipse environment