

# Marek Funtowicz

Embedded (C/C++) Software / FPGA Engineer (VHDL)

## Personal Details

**Phone:** +48 724 235 978

**E-mail:** ice.marek@yahoo.com

**Git:** [github.com/IceDefcon](https://github.com/IceDefcon)

**Web:** [icedefcon.github.io](https://icedefcon.github.io)

## Education

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

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

## Skill

**FPGA:** Intel, Xilinx

**CPU:** x86, ARM, PowerPC

**RTL:** VHDL, Verilog

**Code:** ASM, C, C++, Python

**OS:** FreeRTOS, VxWorks,  
Linux, Autosar OS

**Build:** Makefile, Yocto

**Debug:** J-Link (Ozone), gdb

**Com:** UART, SPI, I2C, JTAG,  
GPIO, TCP/UDP, CAN,  
ASK, FSK

**Circuit:** Mentor graphics, Altium,  
Multisim

**Test:** Oscilloscope, Multimeter,  
Logic Analyzer

**Matrix:** Matlab, Octave

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

**DB:** MySQL

**CD/CI:** GIT, Bitbucket,  
Gerrit, Jenkins

**Mgmt:** JIRA, JAZZ

## Personal Research and Development

### CPU & FPGA Embedded Computing Platform with GPU Acceleration (C, C++, VHDL)

- ❖ LNX → ARM Cortex-A57 powered by Ubuntu
- ❖ FPGA → Intel Cyclone IV FPGA for parallel computation

### AI Drone System Specification → Project Repository

- ❖ LNX → Qt5 Graphical User Interface for the overall master control
- ❖ 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
- ❖ LNX → Spinlock and mutex kernel space synchronization
- ❖ LNX → FPGA controlled scheduler for Kernel space RTOS operations
- ❖ LNX → Memory allocation monitor in Kernel space
- ❖ FPGA → Interrupt vector handler for Kernel commands
- ❖ FPGA → Watchdog interface for Kernel/FPGA synchronization
- ❖ FPGA → SPI interface for data serialization and parallelization
- ❖ FPGA → FIFO for redundant control data from LNX Kernel
- ❖ FPGA → Offload controller and packet switch for FIFO data flow
- ❖ FPGA → Parametric I2C and SPI controllers driven from LNX Kernel
- ❖ FPGA → PWM controller for the motor throttle control
- ❖ FPGA → UART assembler and transmission controller for logs capture
- ❖ FPGA → I2C and SPI Gyro/Accel devices for measurements
- ❖ FPGA → SDRAM Data capture controller for measurements
- ❖ FPGA → SDRAM Measurement storage controller :: **Development**
- ❖ FPGA → Accelerated PID Controller for DC motor feedback :: **Pending**
- ❖ GPU → Investigation of libraries @ 128-Cuda cores :: **Development**
- ❖ GPU → Accelerated measurements processing from FPGA :: **Pending**
- ❖ GPU → Accelerated camera signal video drivers :: **Pending**

### Upgrade to Xilinx Zynq™ UltraScale+™ MPSoC

- ❖ Boot Petalinux system using HW description to link with FPGA logic
- ❖ Port FPGA Driver kernel module into Xilinx system :: **Pending**
- ❖ Convert the link between CPU and FPGA from SPI to DMA :: **Pending**

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

- ❖ Read and analyze prototype PCB schematics for debugging purposes
- ❖ Customer defect analysis, computation and testing solutions
- ❖ Develop interrupt monitor to monitor latency of ISR
- ❖ Investigate communication problems between the charger and the receiver using ASK and FSK protocols
- ❖ Adaptation of existing versions of the protocol into newer stack
- ❖ Process various number of ADC measurements: voltage, current and temperatures signals
- ❖ Diagnostics system control with respect to collected measurements
- ❖ Integration with the Autosar OS (Davinci code generator)
- ❖ Integration with RTD from NXP (Tresos)
- ❖ Integration with Qi-Library from NXP
- ❖ Code reviews

**Software Engineer****Complex CPU & FPGA system to emulate the behavior of cellular network (C, C++, VHDL)**

- ❖ x86 HTML interface to control FPGA and CPU binaries required for specific configuration of cellular network
- ❖ x86 Applications to configure parameters of LTE/5G cells
- ❖ Store and process cell parameters from x86 Applications by the processor boards using instances of C++ classes
- ❖ Configure LTE/5G cells inside FPGA using cell parameters from the instances of C++ classes
- ❖ Communication interface between FPGA and CPU over SRIO
- ❖ Process internal and customer issues using JIRA management system
- ❖ Issues varying from invalid configurations of LTE/5G cells in Applications with respect to LTE/5G specifications
- ❖ Up to the low level PHY problems like: Thread crashes, code disassembly and investigation of instruction code to find the bug

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

- ❖ Integration of the LWIP stack drivers with the current CPU stack
- ❖ Investigation of ASCLIN UART drivers for logging and communication
- ❖ Research Microblaze firmware in ARTIX 7 FPGA
- ❖ Investigate and analyze circuit schematics and PCB layout

**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
- ❖ 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 and live PCB
- ❖ Nios II firmware for communication with FPGA over Avalon bus
- ❖ Debug Nios II firmware on live targets
- ❖ Integrate BSP and Intel HAL drivers
- ❖ Investigate booting codes for embedded Intel Nios II processor
- ❖ FPGA Pin and Chip planning → IO Banks, Voltages, Transceivers
- ❖ 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 and modify U-boot source → Configure MAC addressing
- ❖ Research XEN Hypervisor → Register XEN watchdog
- ❖ Develop UART communication for Zynq Ultrascale MPSoC → register UARTE 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