José Antonio Ruiz-Domínguez

45 Circus Drive CB4 2BT Cambridge (UK)

ja_rd@outlook.com / **(+44) 07741188600**

EDUCATION AND TRAINING

BS in Telecommunications Engineering (Esp. Electronic Systems).

2005 - E.T.S.I.T, Universidad de Malaga (Malaga, Spain)

Final year project: Hopfield neural network on FPGA.

(Fixed point vectorial numeric computation with sparse matrices on Altera Stratix).

RELEVANT EMPLOYMENT HISTORY

Digital Design Contractor — Huawei — Apr/22 - Oct/22

I wrote some verification for an existing video projector chip project (SV, not UVM), on VCS.
I also did a little bit of 'hybrid verification' of an ASIC-FPGA interface over LVDS; hybrid in that the ASIC sources and the relevant FPGA sources were simulated together on a common TB under VCS.

I also wrote a few small video processing IP cores for a prototype for a future chip:

- Color space conversion, YcbCr to RGB
- Gamma correction with programmable gamma value
- Keystone distortion correction (Python fixed point model only)

The cores use an AXI4-Stream video protocol and have APB register blocks that I wrote manually... they include support scripts in Python and documentation along with some SV verification, all by me.

Digital Design Contractor — On Semiconductor — Sep/21 - Feb/22

I helped in the development of the digital side of a new power supply controller chip. I wrote several small modules for a CPU subsystem: GPIO, interrupts and a few others. For these cores I did design, implementation and verification in SV (not UVM), plus some documentation.
I also helped update a Perl script for the autogeneration of APB register blocks and did other ancillary tasks typical of a digital project of this kind, including some Python scripting.

Digital Design Engineer (senior) — Qualcomm — May/19 - Aug/21

I went back to my old job at Qualcomm where I did largely the same I used to do in 2013/16. In this second period, I helped implement a shared memory module for an Audio chip, that was cancelled before being finished.

I also built the power management subsystem for an IoT SoC. I was a one person team, supported by a multi-person verification team. I was responsible for gathering requirements, coming up with specs, thinking a design, implementing it and doing some basic verification (SV, not UVM). Plus all the relevant documentation.

A very simple power controller including only the digital side of the always-on core that powers the chip in all power saving modes. Only two clock domains, quite simple CDC, straightforward FSMs, interfaces to the rest of the chiphomogeneized to the extent it was sensible... plus the chip worked after the first tapeout.
All in all a satisfying end to my period at Qualcomm.

FPGA Engineer Kaleao Nov/18 - May/19

I helped maintain, develop and refactor an existing FPGA project. Vivado on Zynq.

In this short period I wrote a lone IP block (IP/TCP packet checksum patcher) in Verilog along with its test bench in System Verilog. I also reviewed and refactored existing code in Verilog and VHDL.

My day to day was spent mostly writing System Verilog and TCL code for Vivado and System Verilog test bench code.

Senior Engineer, Architecture ARM Nov/17 - Sep/18

I transcribed the specification of new architecture extensions from a human-readable document to a variety of proprietary machine-readable formats (pseudocode, register definitions, etc.).

I also did some C++ maintenance work (GEM5, trivial code mostly), wrote ad-hoc assembly code tests for the new extensions ('smoke tests' for model development, not part of any actual test suite) and did other ancillary tasks.

Digital Design Engineer (senior) CSR/Qualcomm Sep/13 - Oct/17

Mostly I designed and implemented IP blocks to be integrated in larger subsystems, in Verilog or System Verilog.
I did a moderate amount of testing too in SystemVerilog and C and/or assembler for our DSP.
Within the last year I also worked programming test utilities for our DSP in Python/C/Verilog.

Projects and roles

A necessarily brief summary of my 4 years in CSR/Qualcomm:

1. Developed several small blocks for audio processing chip (successfully taped out in 2016).
   a) Decryption module for Quad SPI flash data stream.
   b) Configurable arbitration module for the same.
   c) Including design, implementation, documentation, some testing, SW bring-up support.

2. Part of a 2 man team that implemented the 'applications subsystem' in the aforementioned chip.
   a) Integration of several IP blocks and two custom 32-bit DSPs.
   b) Memory interconnect fabric (custom bus), arbiters, interrupt logic...
   c) Subsystem-level tests (DSP SW cooperating with test RTL).

3. Developed a proprietary *random instruction sequence* test scheme for our custom DSP.
   a) Generic system that can be retargeted to any common ISA within certain constraints.
   b) ISA is described using custom DSL: inclusing all encoding and semantic information.
   c) Auto-generates instruction set simulator in C from ISA description (bit-perfect golden model).
   d) Constrained random sequence generator also auto-generated from ISA description.
   e) Strong complement to directed test suite.
   f) Mostly Python, C and auto-generated C.
   g) A one-person effort that yielded a small system that is largely state of the art.

In my last stage in this position I worked mostly on the maintenance of our custom 32-bit fixed-point DSP and the elaboration of its test suite, including the random test scheme mentioned above.

Prior Work in Embedded Software | Various Spanish Companies | Sep/07 - Sep/13

In my previous life back in Spain I worked on a lot of things, including some stints doing Java/JSP, but most of that period is not worth going over in detail after all these years.

I did have a few interesting jobs doing embedded software, in GCM (now extint startup) and Sepsa. The work included the following, among many other things:

- I maintained several embedded applications in C running on bare-metal C166 processors.
  I developed a small application (C on XE164) for which I made the software specification and design, the implementation and the SW test environment. This was a SIL-2 level project involving extensive documentation and testing.
  Please note that we relied on an external consultant for the SIL-2 paperwork and rule checking, I just followed the instructions to build the software as per the rules.

- A couple of laboratory demonstration projects on small TI DSPs, in C and assembly for the C2000 family. These demos applied a proprietary signal processing technology to the application domain of a potential customer (power line communications and others). Not real products meant for tough real life work conditions but mere demos meant to be shown at meetings.
  The point is, I can do very low level C and assembly in tricky architectures.

- A uClinux (2.4) driver running on a MMU-less ARM C4 MCU. The device in question was a signal processing pipeline implemented on an FPGA. I wrote a userland program that used the driver to update data in the device in real time (calculation of equalization filter for a channel every few seconds). My first and last Linux driver so far.

KEY SKILLS

Digital design and verification

- ✓ Can design for FPGA or ASIC technology using vendor-agnostic VHDL or Verilog/SystemVerilog.
- ✓ Experience building designer-friendly block-level test benches in System Verilog (***not UVM***).
- ✓ Can develop custom ad-hoc SW tools when required, including models for hardware devices.
- ✓ Experience in ISA modelling with ISS (*including auto-generation of ISS from pseudocode*).

Embedded programming in C and assembler

- ✓ Knowledgeable in low-level C programming, plus assembler for several CPUs.
- ✓ Have worked to MISRA-C standard and can adapt to tighter rule sets and methodologies.
- ✓ Can build test harnesses with C, Python or Matlab as appropriate, on host or on target.
- ✓ Familiar with Test Driven Development techniques.

PERSONAL OPEN-HARDWARE PROJECTS ON FPGA

1. Synthesizable CPU core: Intel 8051 compatible CPU:
   (https://github.com/jaruiz/light52). *(Includes datasheet)*.
   - May be smallest free 8051 core around, not far behind low-end commercial cores.
   - Dhrystone benchmark score equivalent to a 6-clocker MCS51.
   - Includes ISS for cosimulation, made for this project in C.

2. Synthesizable CPU core: Intel 8080 compatible CPU:
   (https://github.com/jaruiz/light8080).
   - Tested for 100% binary compatibility to original silicon.
   - Tried with original, vintage software – Microsoft's Altair 8080 Basic.

There’s a few more pet repos in my Github page that can be useful as (admittedly very old) code samples.