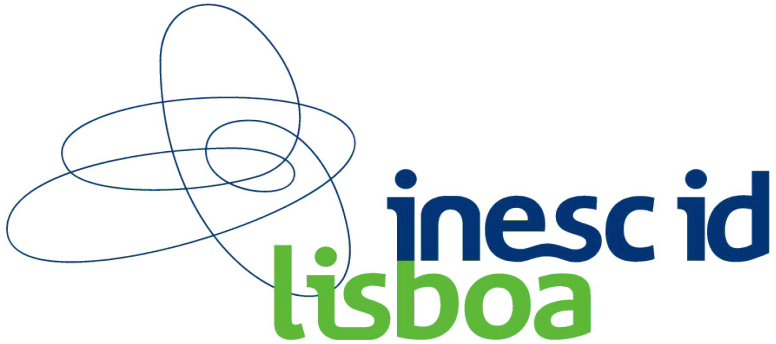
VERSAT, a Minimal Fully and Partially Reconfigurable Coarse-Grain Reconfigurable Array

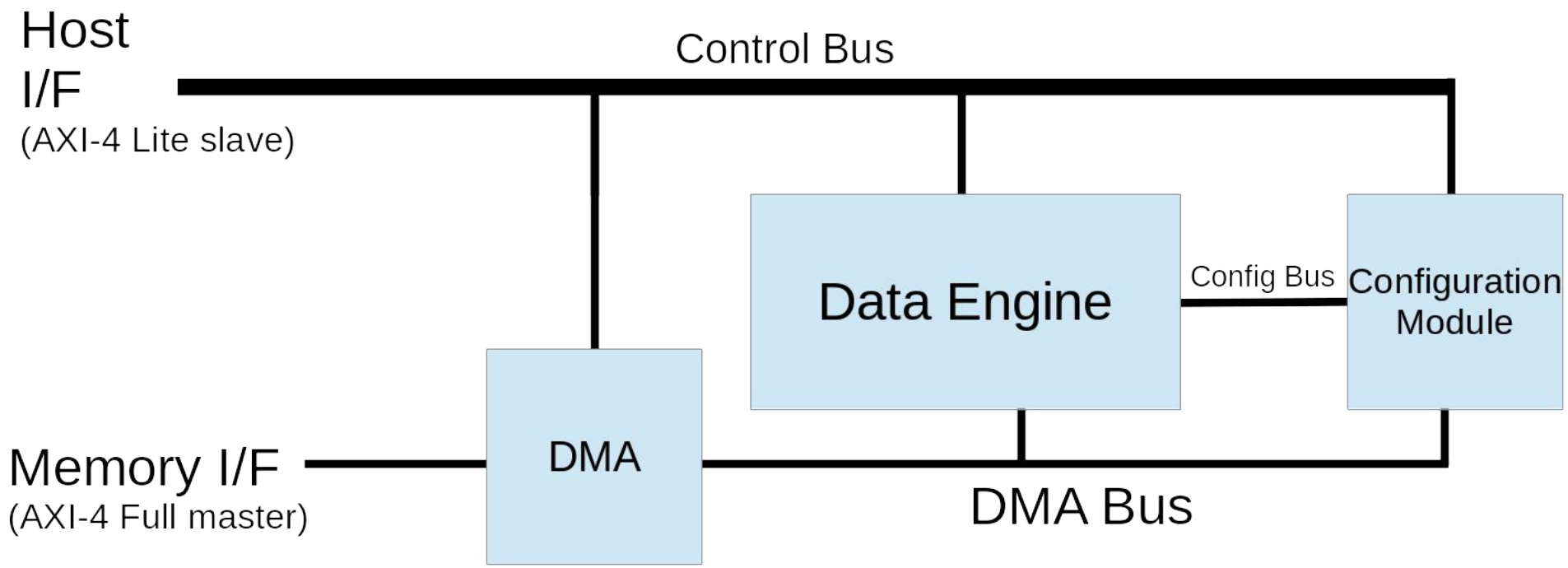
João D. Lopes



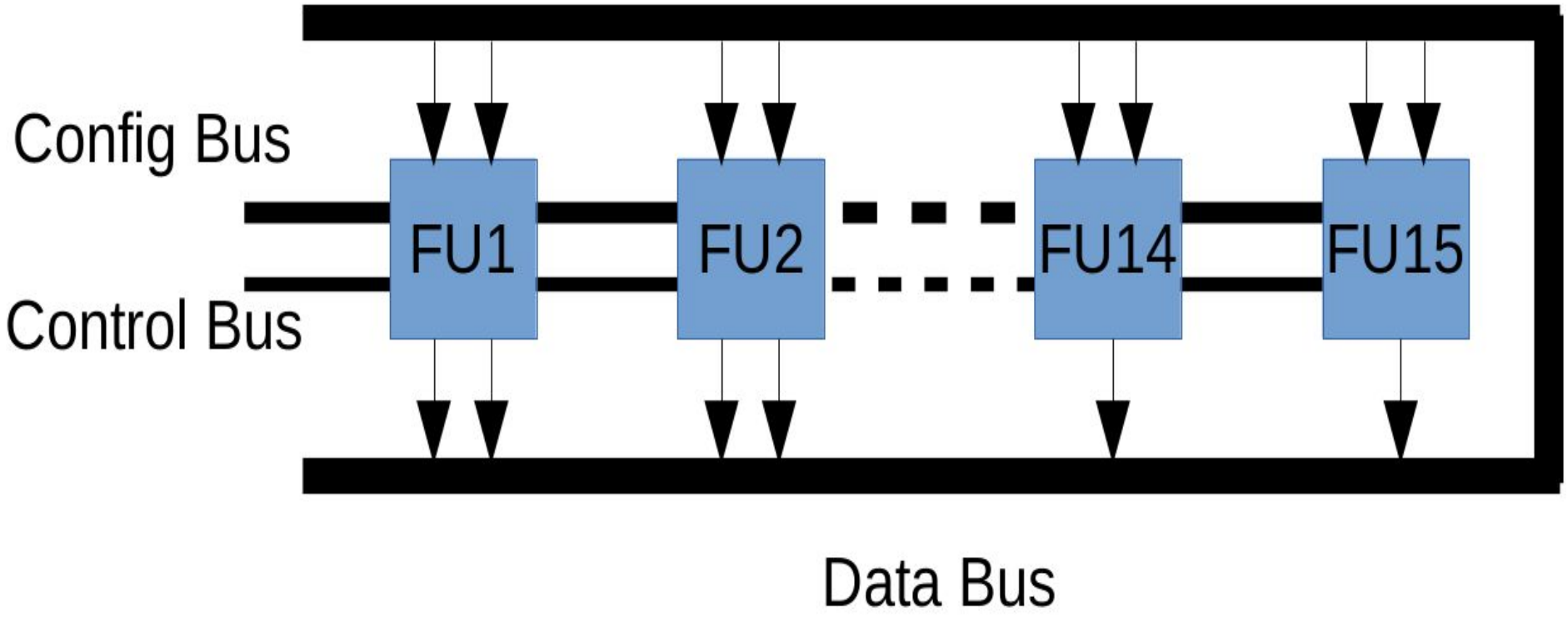
Introduction

- Modern embedded systems are becoming more and more power hungry
- Internet of Things (IoT): small devices, battery operated (eg. sensor networks)
- Need local processing to avoid IoT data deluge
- Use reconfigurable hardware for high performance at low clock rates
- Field Programmable Logic Arrays (FPGAs) are a possibility but consume large silicon area and high power
- We consider Coarse-Grain Reconfigurable Arrays (**CGRAs**) used as co-processors

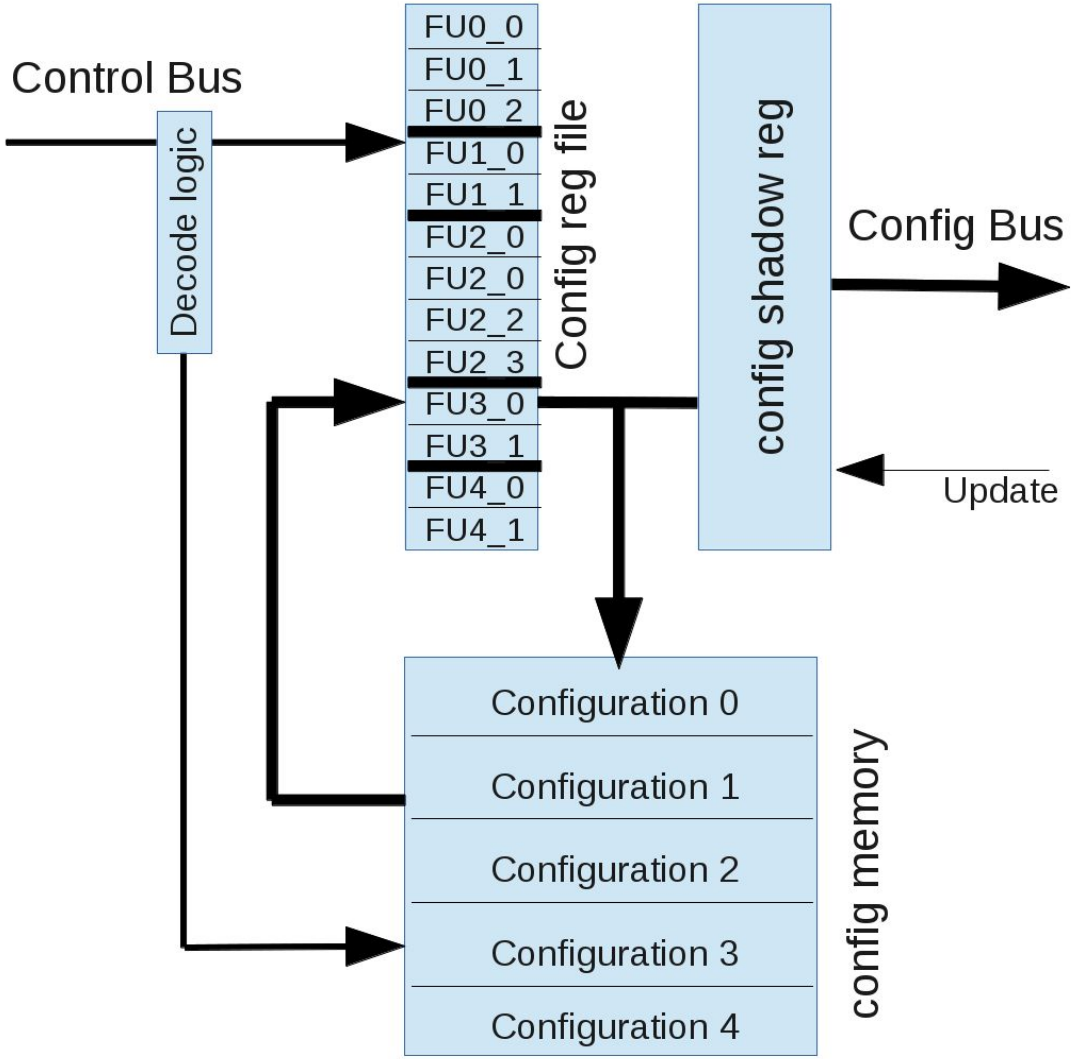
Top-Level Entity Overview



Data Engine



Configuration Module



Host processor program: vector addition

```
#include “versat.h”
…
versat.dma.load(data_ptr, mem0, 1000); // load data

versat.de.clear();

versat.de.mem0.portA.setIter(500);
versat.de.mem0.portB.setIter(500);
versat.de.mem0.portA.setIncr(1);
versat.de.mem0.portB.setIncr(1);
versat.de.mem0.portB.setStart(500);

versat.de.add0.setOpA(mem0A);
versat.de.add0.setOpB(mem0B);
versat.de.add0.setFunc(ADD);

versat.de.mem1.portA.setSel(add0);
versat.de.mem1.portA.setIter(500);
versat.de.mem1.portA.setIncr(1);
versat.de.mem1.portA.setDelay(3);

while (!versat.dma.done()); // wait for DMA to finish
versat.run(); // run a Versat configuration
…
while (!versat.done()); // wait for Versat to finish
versat.dma.store(data_ptr, mem1, 500); // store data
…
while (!versat.dma.done()); // wait for DMA to finish
…
```

ASIC implementation results (ref. 40nm)

Core	Area (mm^2)	RAM (KB)	Power density (uW/MHz)
ARM Cortex A9	4.6	65.54	62
Morphosys	2.2	6.14	914
ADRES	0.8	65.54	60
Versat	0.4	46.34	55

Versat vs Morphosys

- Morphosys is 4.6x faster than Versat running a 1024-point FFT kernel
- Morphosys is 5x the size of Versat, in the same technology
- Morphosys consumes 16.6x more power than Versat, at the same frequency
- Morphosys consumes 3.6x more energy than Versat

Is an increased area and power consumption justified when the CGRA is integrated in a real system?

Conclusion

- Versat, a minimal CGRA with 4 embedded memories and 11 FUs is presented
- The CPU can generate configurations and uses partial reconfiguration whenever possible; it is also in charge of data transfers and basic algorithmic flows
- It is programmed in C++
- Versat is competitive in terms of silicon area and energy efficiency (11x smaller, 17x faster and more energy efficient on a 1024-point FFT compared to ARM Cortex A9)
- Compared to another CGRA, Versat has same performance per functional unit but consumes (5x) less silicon area and (3.6x) less energy
- Versat is easy to program and do not need any complex compiler tool