Superscalar Performance without Superscalar Overheads

Ali Mustafa Zaidi & David Greaves

UNIVERSITY OF CAMBRIDGE
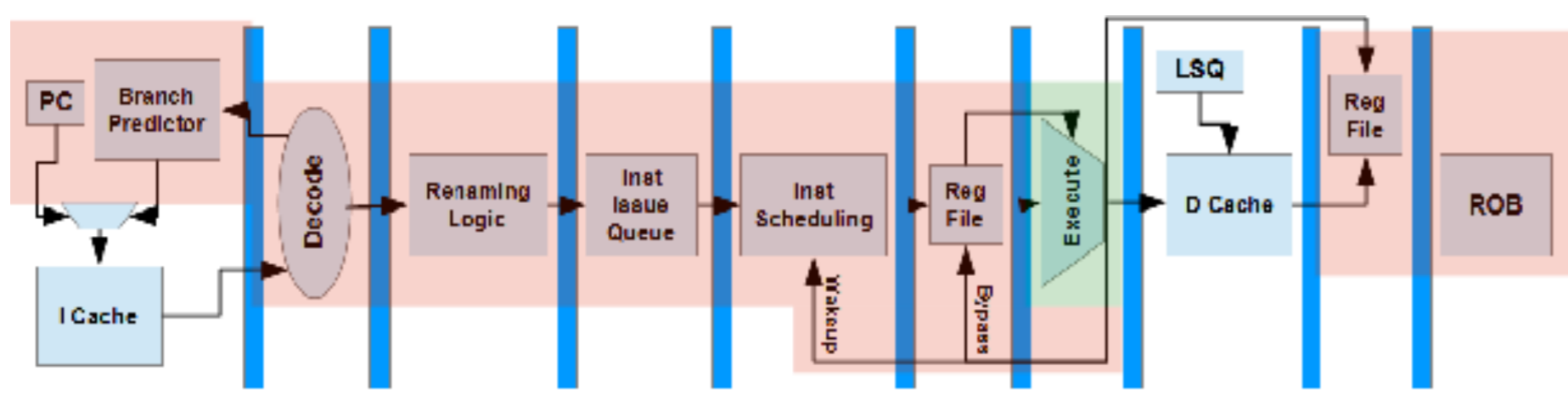Computer Laboratory

The 'Dark Silicon' Challenge

In the 'Dark Silicon Era', achievable speedups will be far below the exponential growth suggested by Moore's Law, primarily due to constrained power budgets. To mitigate the effects of Dark Silicon, we must dramatically improve the energy efficiency of computation.

Computer architects are increasingly relying on custom hardware to achieve high performance with very high efficiency. Unfortunately, there are several problems:

(1) **Amenability**: custom hardware currently exhibits poor performance on irregular, control-flow intensive applications, or those with low data parallelism.

(2) **Programmability**: developing custom hardware is far more difficult & time consuming than writing code for a sequential processor.

Thus to provide a balance between energy, performance and programmability, we must build heterogeneous designs: combining complex, power hungry superscalar processors with energy efficient custom hardware.

The Dark Silicon Problem

$$Speedup = \frac{1}{\dfrac{(1-f)}{S} + \dfrac{f}{N*S}}$$

f =
1
0.99
0.9
0.75
0.5

Number of Cores

2.1GHz @ 90nm (80W) — 18% → 5.2GHz @ 45nm (80W) — 7% → 7.3GHz @ 32nm (80W) — 3%

$$EPI_{CPU} = \frac{E_{I-CacheWarmup}}{InstructionCount_{Static}} + \frac{(E_{Execute} + E_{CPUOverheads})}{InstructionCount_{Dynamic}}$$

Superscalar processors achieve high performance by internally converting from the sequential Von Neumann to the highly concurrent Dataflow computational model: (1) Branch prediction is used to overcome control dependences, (2) Register Renaming is used to overcome name dependences (3) Out-of-order Issue, Bypass & forwarding logic is used to accelerate true dependences. However, performing this conversion at runtime in a general-purpose processor incurs over 10-1000x greater energy overheads than the equivalent custom / reconfigurable hardware implementation of an application.

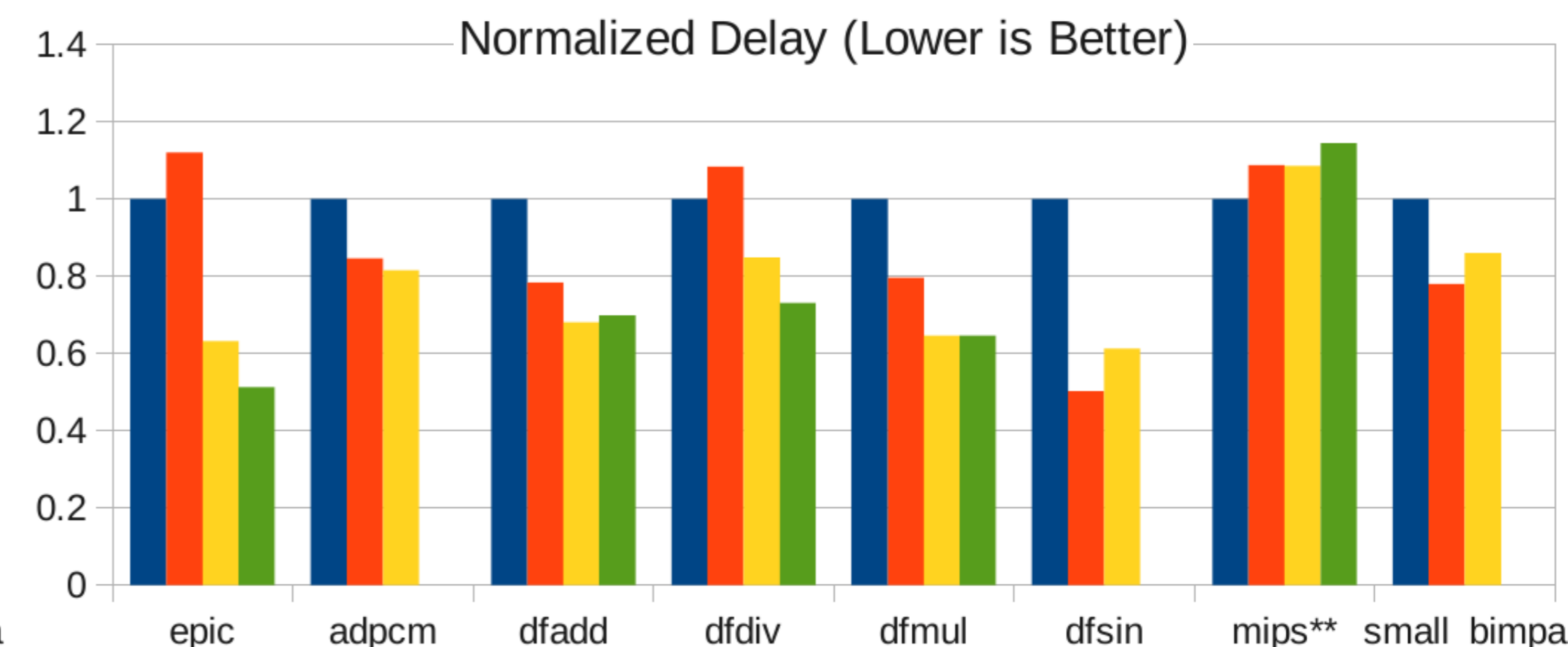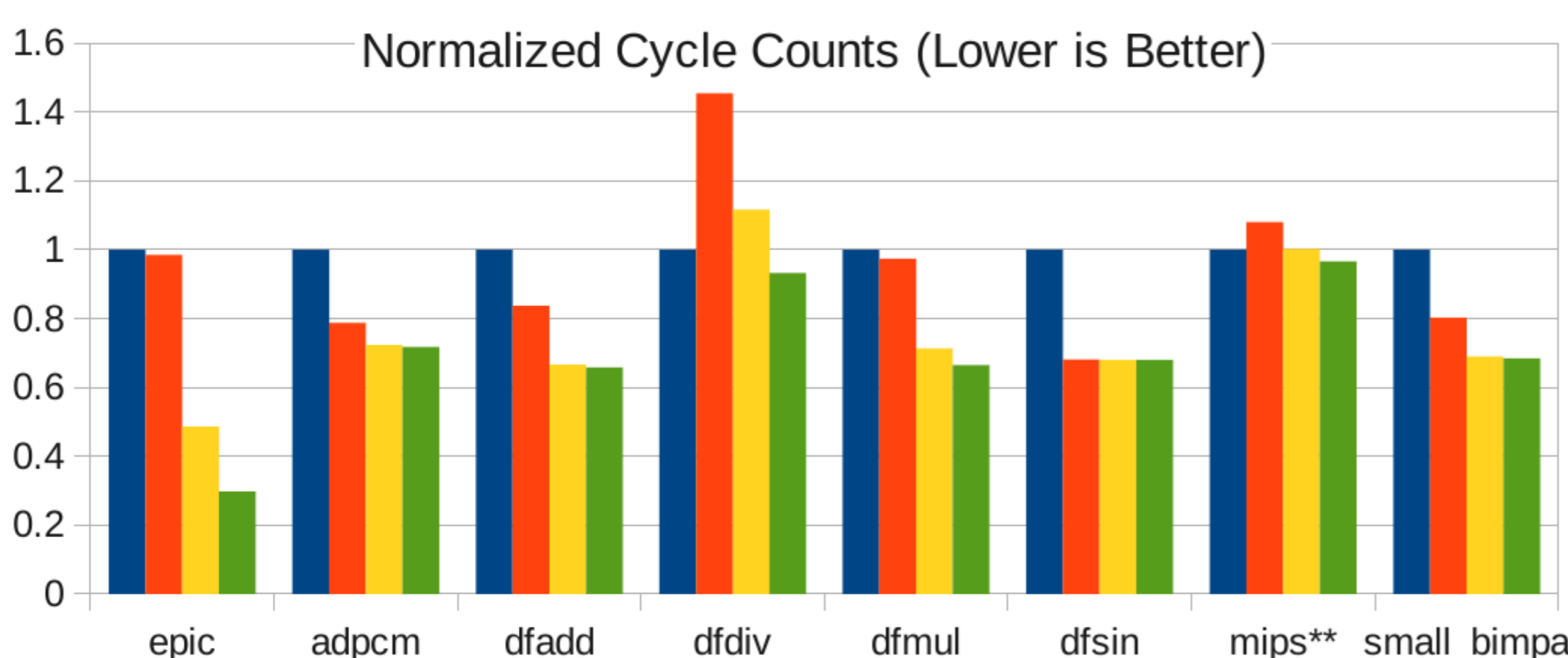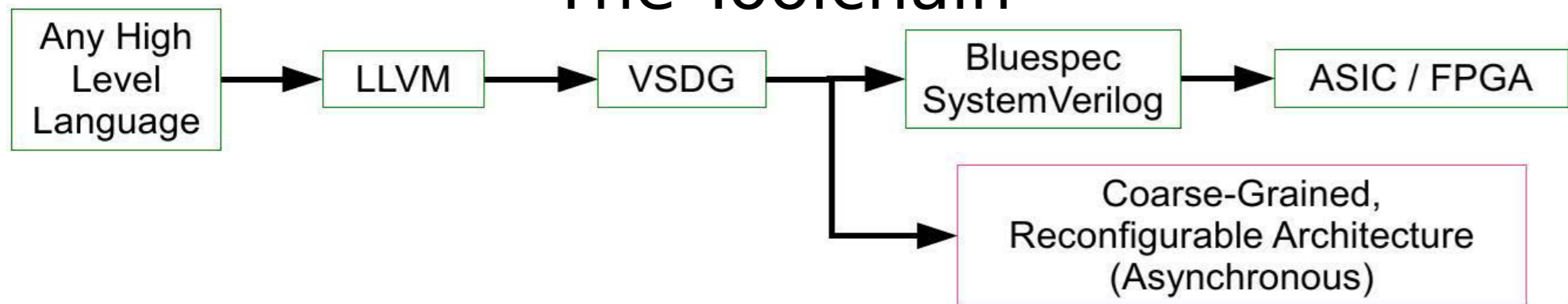## Dataflow Computing for Efficiency and Performance

Our research goal is to address the Programmability and Amenability issues of custom hardware, enabling a broader, more energy-efficient design space for heterogeneous systems. We do this by:

(1) Developing a new program representation (IR) - the 'Value-State Flow Graph (VSFG)' that is inherently dataflow oriented, exposing much greater concurrency than existing IRs like the conventional Control Flow Graph (CFG).

(2) Providing a new High-level synthesis (HLS) toolchain that converts input high-level language code into our dataflow IR and then implements it as custom dataflow hardware.
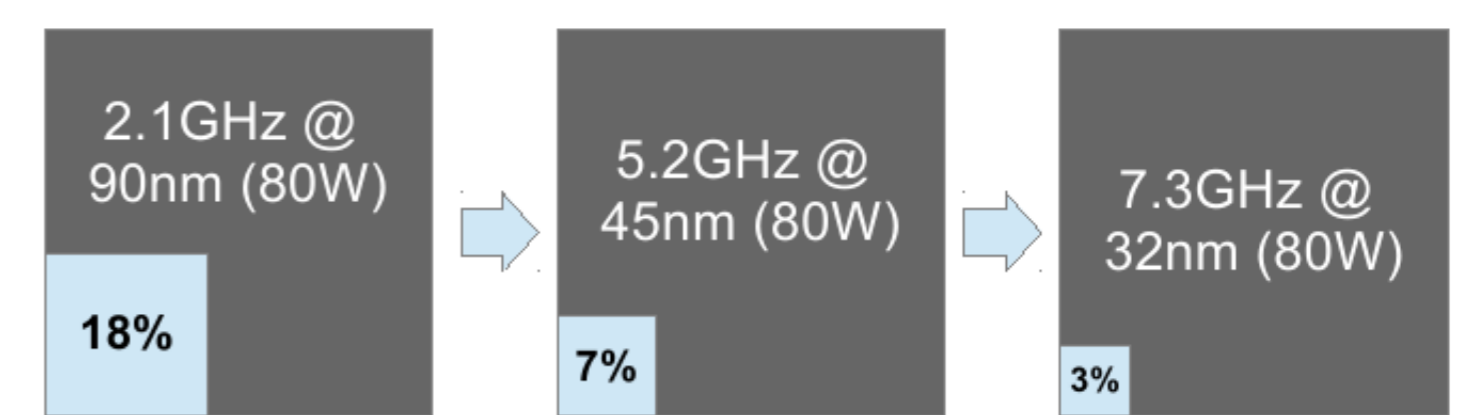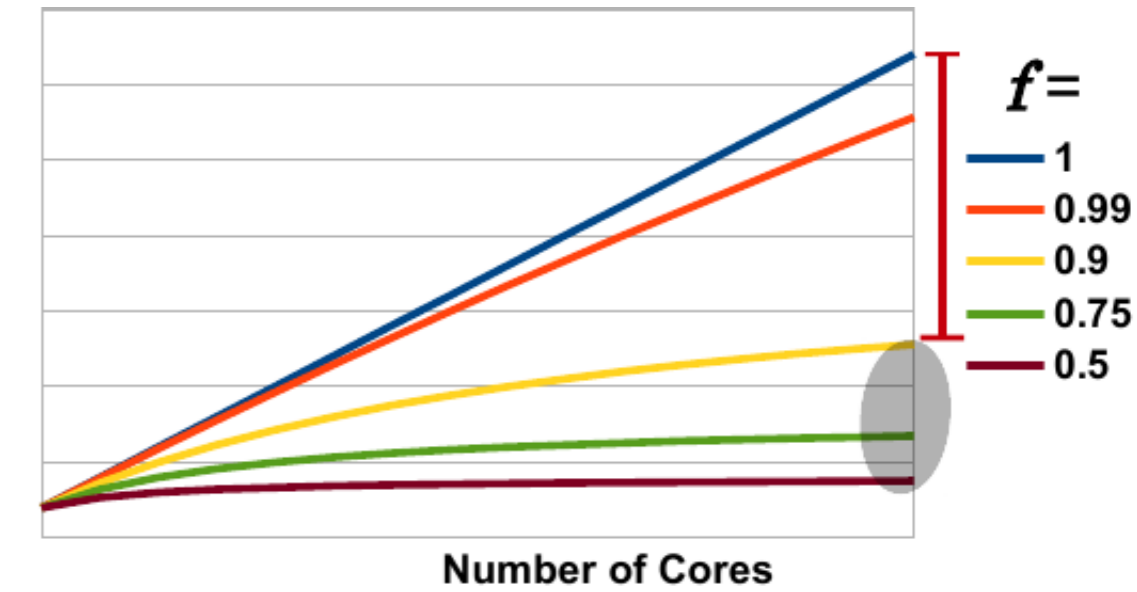
(3) Unlike conventional HLS tools, but like superscalar processors, our generated custom hardware uses *dynamic execution scheduling* to provide further performance improvements on control-flow intensive code.
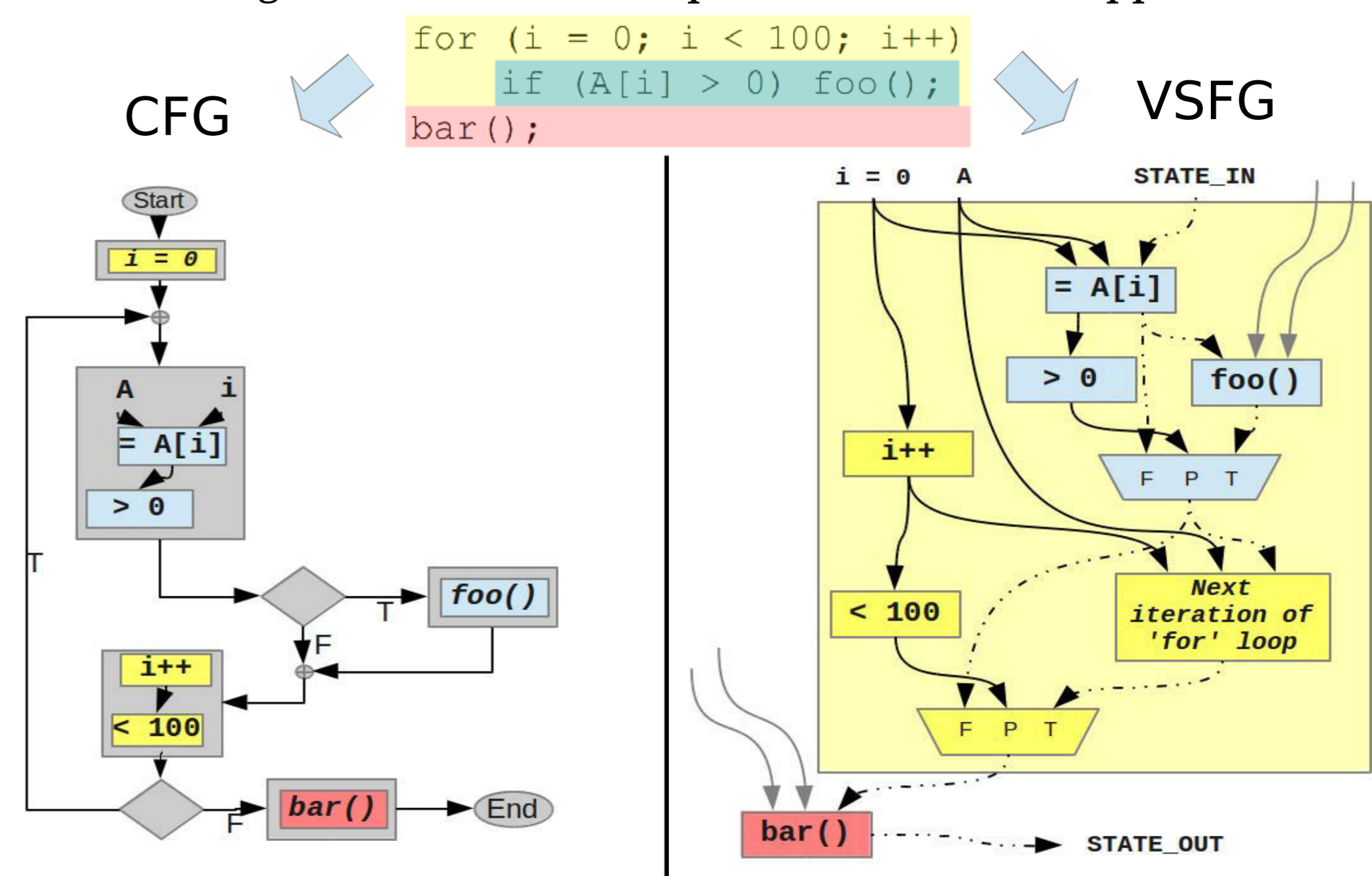
```
for (i = 0; i < 100; i++)
    if (A[i] > 0) foo();
bar();
```

CFG   VSFG

The VSFG exposes more concurrency by statically converting control flow to dataflow

The Toolchain

Any High Level Language → LLVM → VSDG → Bluespec SystemVerilog → ASIC / FPGA
→ Coarse-Grained, Reconfigurable Architecture (Asynchronous)

The Results

We compare against LegUp, an existing HLS tool from Toronto. The VSFG allows us to perform aggressive loop unrolling. VSFG results shown for 0, 1, & 3 degrees of unrolling.

Legend: LegUp (CFG), VSFG_0, VSFG_1, VSFG_3

Normalized Cycle Counts (Lower is Better)

Normalized Delay (Lower is Better)

Ali-Mustafa.Zaidi@cl.cam.ac.uk
David.Greaves@cl.cam.ac.uk

Computer Architecture Group