Theme 3

Hardware Instrumentation and Control

George A. Constantinides

PRiME Industry Day

30/6/15

PRiME

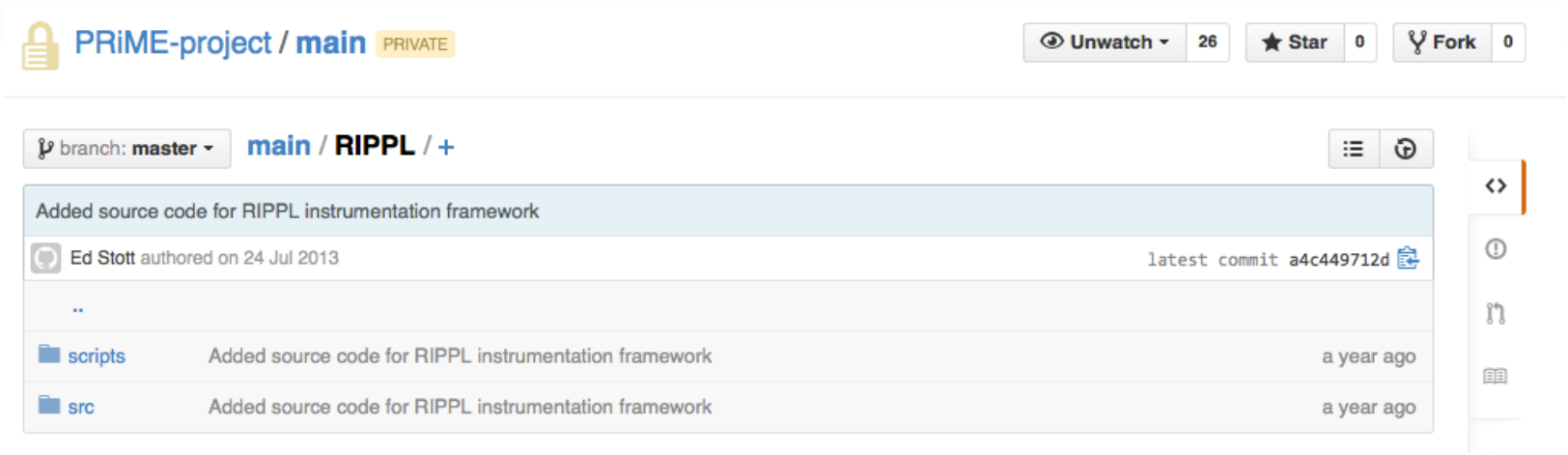
www.prime-project.org

Theme 3: The Big Idea(s)

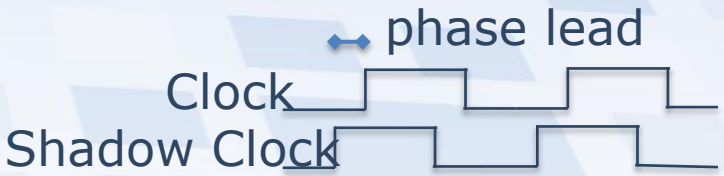
- What will future EDA/Compilation look like across the HW/SW boundary?
- What can be determined at compile (synthesis) time and what should be postponed to run-time?
- How can we develop hardware support for run-time monitoring and control of power and reliability?
- So:
  - Developing “knobs and monitors”
  - Compile-time and run-time specialisation of memory subsystems
  - Circuit level techniques for improved energy/reliability tradeoff
  - EDA flows and analysis

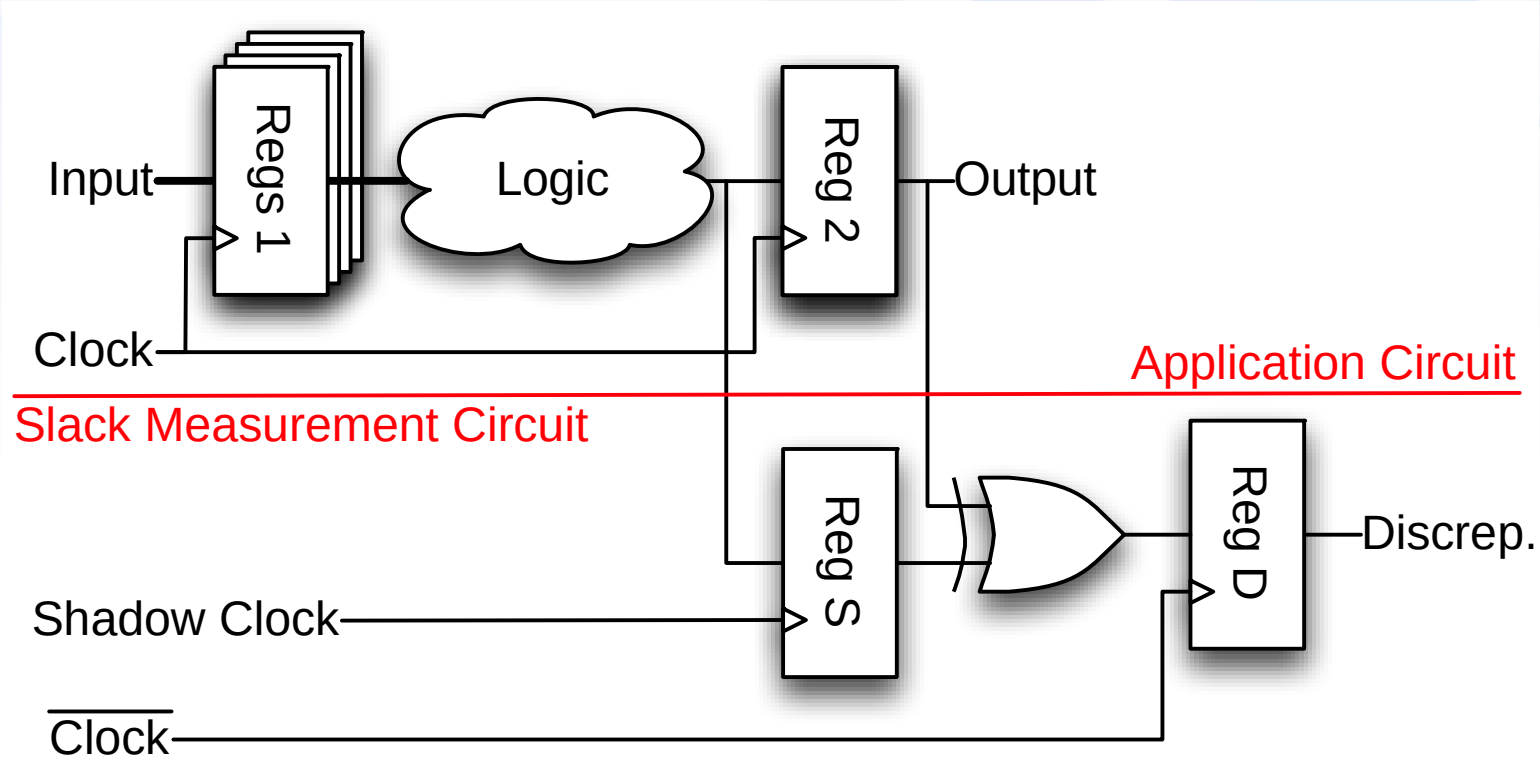
Theme Progress

- Timing Measurement and Control
  - RIPPL flow
  - Instrument RTL for real-time timing feedback
  - Port to Cyclone V and sufficient FPGA features for soft-core processors
  - Papers
    - FPGA 2015 “Delay-Bounded Routing for Shadow Registers”
    - FCCM 2014 "Timing Fault Detection in FPGA-based Circuits”
  - Development of infrastructure for whole-project demonstrator



Timing Measurement and Control





Theme Progress

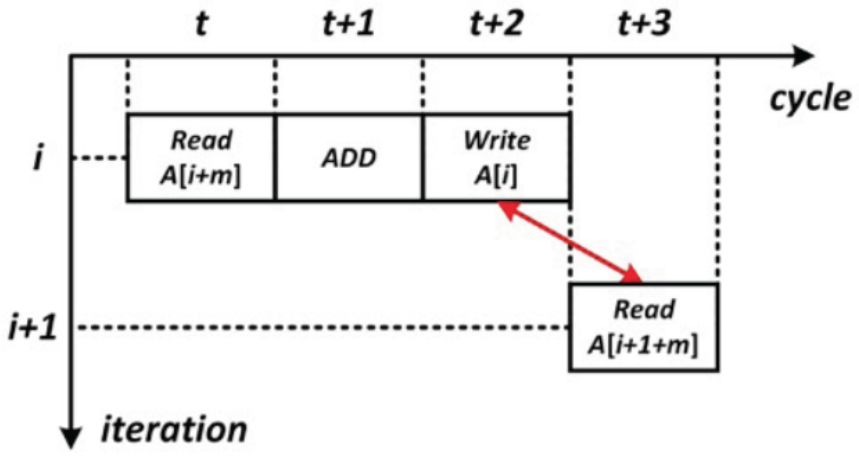
- Static analysis for dynamic memory access patterns
  - How can static analysis be extended to uncertain memory patterns?
- Initial approach: loop nests characterised by a parameter vector
- Working compiler based on clang/LLVM infrastructure and iSL. Uses VivadoHLS as a back-end.

```
22  // 2D nested loop
23  #pragma scop
24  for (j=N; j<2*N; j++){
25    for (i=LB; i<N; i++){
26      # pragma HLS PIPELINE
27      A[j][i-m] = A[j][i] + j;
28    }
29  }
30  #pragma endscop
```

```
22  // 2D nested loop
23 /* Begin Accelerated Scop */
24 float* A_flt = A;
25 {
26   if (m <= -15 || m >= 0) {
27     for (int c1 = 100; c1 <= 199; c1 += 1)
28       for (int c3 = 20; c3 <= 99; c3 += 1)
29         #pragma HLS PIPELINE
30         #pragma HLS DEPENDENCE variable=A_flt array inter false
31         A_flt[-m + 200 * c1 + c3] = A_flt[200 * c1 + c3] + c1;
32   } else
33     for (int c1 = 100; c1 <= 199; c1 += 1)
34       for (int c3 = 20; c3 <= 99; c3 += 1)
35         #pragma HLS PIPELINE
36         A_flt[-m + 200 * c1 + c3] = A_flt[200 * c1 + c3] + c1;
37 }
38
39 /* End Accelerated Scop */
```

Loop Pipelining

```
for (int i=LB; i<=UB; i++) {
    A[i] = A[i+m] + 1;
}
```



II = 3 is safe

Theme Progress

- Number Representation
  - Paper in DAC 2014, “Datapath Synthesis for Overclocking: Online Arithmetic for Latency-Accuracy Trade-offs”
  - Paper in FPT 2014, “Efficient FPGA Implementation of Digit Parallel Online Arithmetic Operators”
  - Paper in WAPCO 2015, “Evaluation of Design Tradeoffs for Adders in Approximate Datapath”

(e) $1.25 f_0$, SNR:26.7dB

(f) $1.25 f_0'$, SNR:8.5dB

Theme Progress

- Low Overhead Fault Detection
  - For matrix multiplication based algorithms
  - Paper in FPL 2014, “Achieving Low-overhead Fault Tolerance for Parallel Accelerators with Dynamic Partial Reconfiguration”

Say

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 4 & 6 \end{pmatrix} \times \begin{pmatrix} 5 & 6 & 11 \\ 7 & 8 & 15 \end{pmatrix} = \begin{pmatrix} 19 & 22 & 41 \\ 43 & 50 & 93 \\ 62 & 72 & 134 \end{pmatrix}$$

instead results in

DPR!

$$\begin{pmatrix} \textcircled{21} & 22 & 41 \\ \textcircled{45} & 50 & 93 \\ \textcircled{63} & 72 & 134 \end{pmatrix}, \begin{pmatrix} 19 & \textcircled{23} & 41 \\ 43 & \textcircled{51} & 93 \\ 62 & \textcircled{73} & 134 \end{pmatrix} \text{ or } \begin{pmatrix} 19 & 22 & \textcircled{43} \\ 43 & 50 & \textcircled{95} \\ 62 & 72 & \textcircled{135} \end{pmatrix}$$

Future Plans / Work in Progress

- Knobs / Monitors
  - Public release of timing instrumentation tools
  - Low overhead power estimation instrumentation
  - Providing knobs to switch numerical representation
- Analysis
  - Lifetime extension (NBTI degradation)
- Systems
  - Rollback FIFO implementation
  - Integration with machine learning