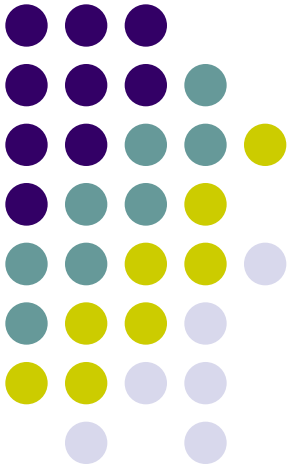
Circuit Modeling for Practical Many-core Architecture Design Exploration

Redefining design abstractions



Dean Truong

Bevan Baas

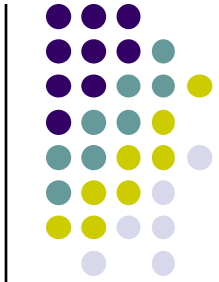
VLSI Computation Lab

University of California, Davis

Outline

- **Motivation**
- Circuit Model of Two Cores
- DVFS Control and Results
- Improved Model
- Tomasulo’s Algorithm
- Conclusions

Motivation



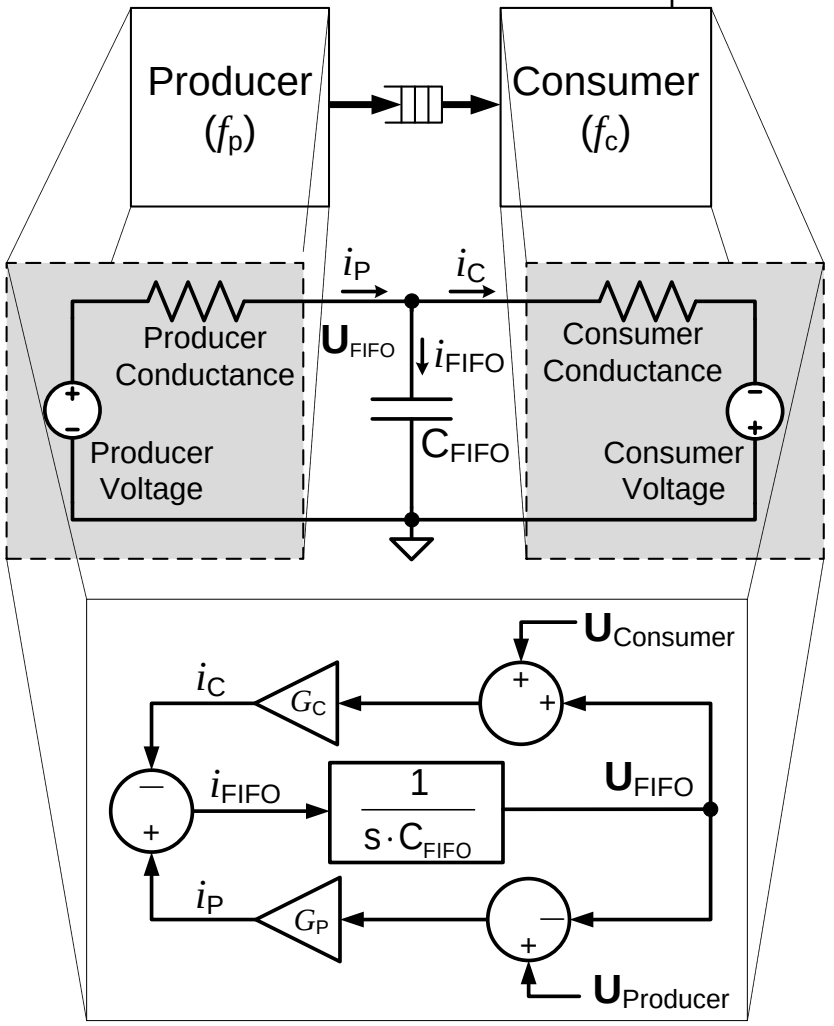
- We have the hardware
  - Many-core chip capable of per-core DVFS
  - Inter-core communication through FIFOs
- We need an accurate and intuitive modeling of the system to find an optimal DVFS scheme
- Achieved through appropriate analogies
  - Circuit analogy is useful because circuit analysis has well developed CAD and mathematical tools
  - Control analogy is useful because control systems analysis is very mature
- Model many-core systems like circuits?

Outline

- Motivation
- **Circuit Model of Two Cores**
- DVFS Control and Results
- Improved Model
- Tomasulo’s Algorithm
- Conclusions

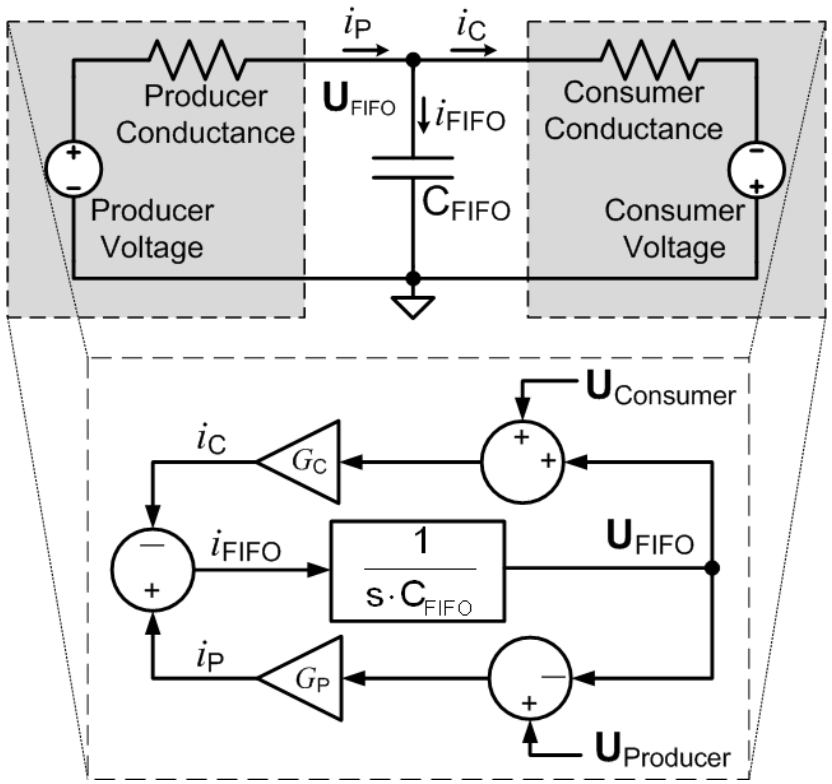
Circuit Model of Two Cores

- Producer→Consumer application
- Charge (*Q*) = sample
- Current (*I*) = samples/sec; data rate
- Conductance (*G*) = cycles/sec; (core) frequency
- Voltage (*V*) = samples/cycle; $(CPI \times IC)^{-1}$
  - CPI = cycles per instruction
  - IC = instructions per sample
- Ohms Law: $I = V \times G$
  - data rate = samples/cycle x cycles/sec
- Capacitance (C) = Q/V = cycles
- Time = C/G = **seconds!**



Circuit Model (cont.)

- Want $\mathbf{U}_{FIFO} = 0$
  - $\mathbf{U}_{FIFO} > 0$: FIFO full
  - $\mathbf{U}_{FIFO} < 0$: FIFO empty
  - When $\mathbf{U}_{FIFO} \neq 0 \rightarrow$ core(s) are stalling $(\mathbf{U}_{FIFO}/\mathbf{U}_{Core})\%$ of the time
- Continuous time model
  - DVFS algorithm only acts over long periods compared to core frequencies
  - $T_{control} >> 1/f_{Core}$

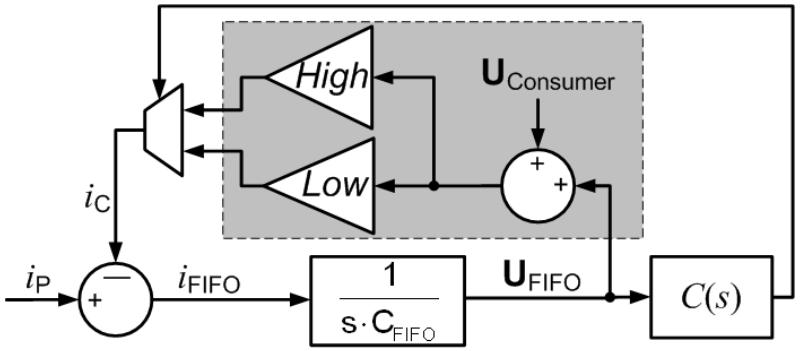


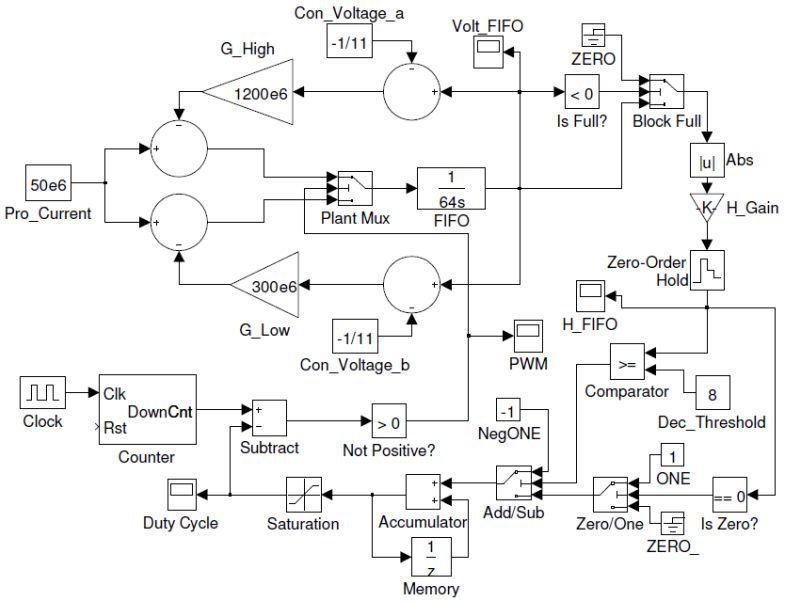
Outline

- Motivation
- Circuit Model of Two Cores
- **DVFS Control and Results**
- Improved Model
- Tomasulo’s Algorithm
- Conclusions

DVFS Control Strategy

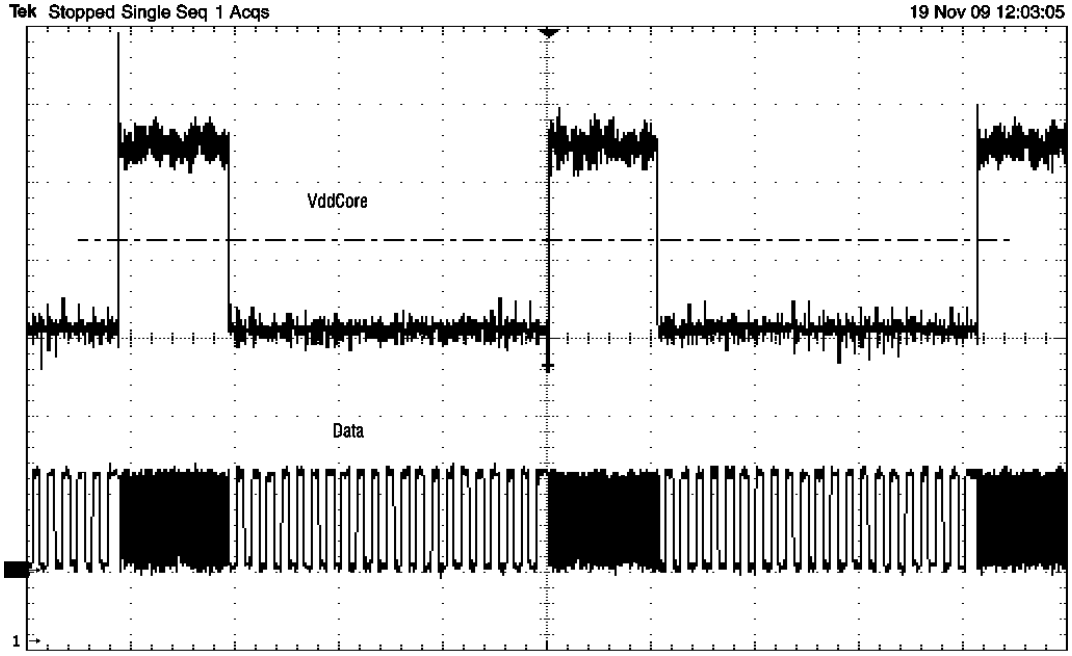
- Varying the conductance (frequency) makes control nonlinear
  - Approximate: two linear control systems selected by a mux
- Controller selects *High* and *Low* states through the mux
  - Voltage dithering!
- Controller samples $\mathbf{U}_{FIFO}$ every dither period
  - Controller is “discrete”
- Modeled in MATLAB and implemented in software using processor DVFS instructions





Measured Results

- Worst case:
  - 46.6 mW at *VddHigh* = 1.3 V; f_H = 1.2 GHz
- Ideal case:
  - 12.0 mW at *VddHigh* = 0.9 V; f_H = 550 MHz
- DVFS:
  - 15.6 mW at *VddHigh* = 1.3 V, *VddLow* = 0.8 V; f_H = 1.2 GHz, f_L = 300 MHz
- Settling time
  - Several hours!



P_L = 74.4%, P_H = 25.6%

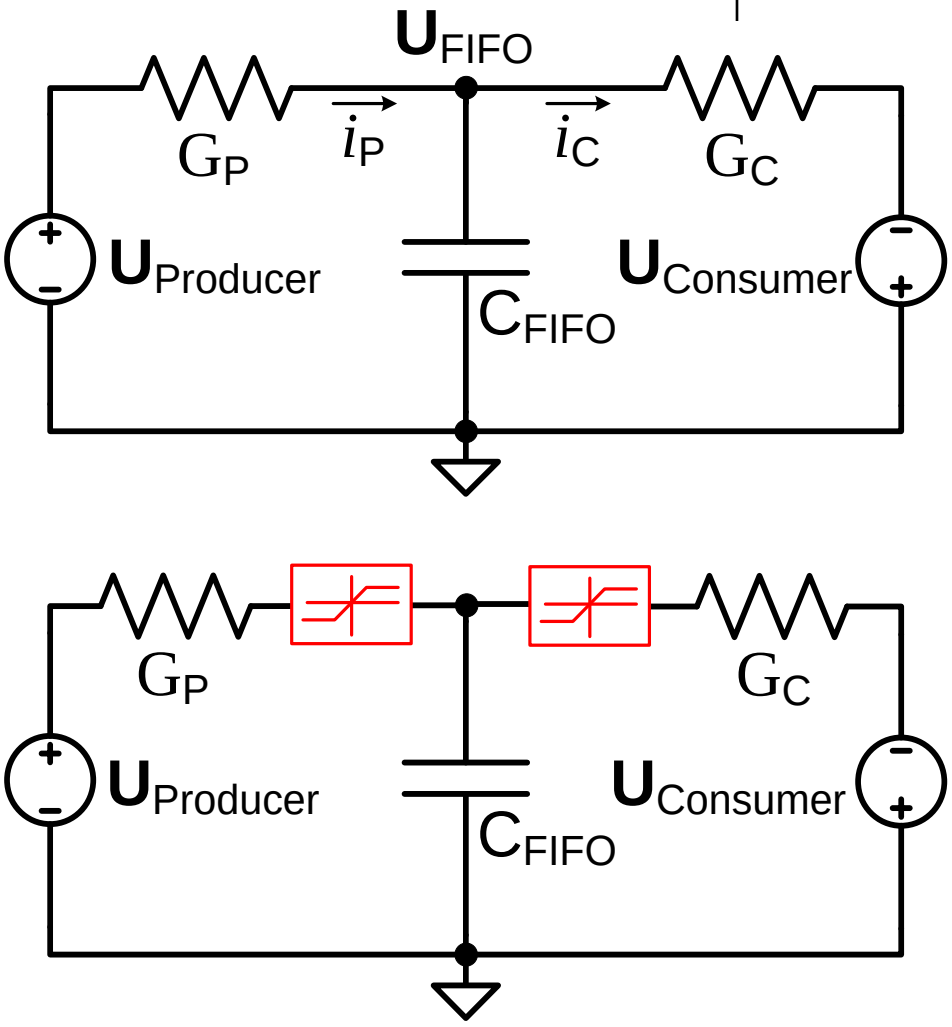
Ideal: P_L = 72.22%, P_H = 27.78%

Outline

- Motivation
- Circuit Model of Two Cores
- DVFS Control and Results
- **Improved Model**
- Tomasulo's Algorithm
- Conclusions

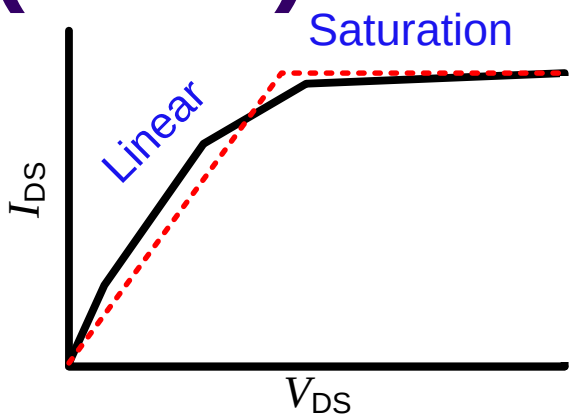
Saturation Modeling

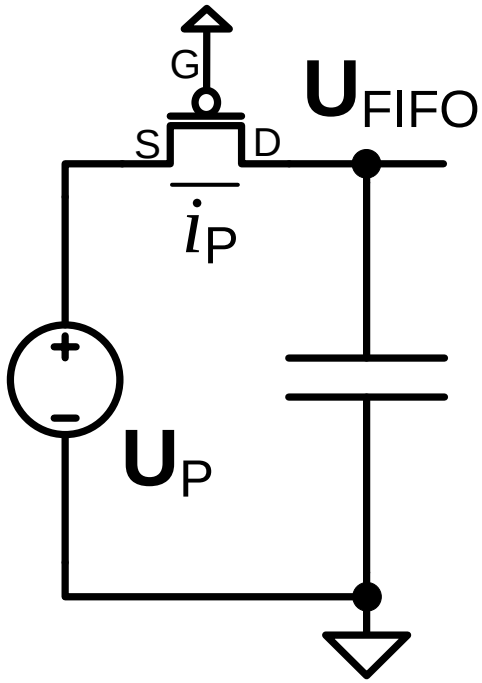
- So far our circuit model does not saturate the data rate when $\mathbf{U}_{FIFO} \neq 0$
  - If $\mathbf{U}_{FIFO} < 0$,
    $i_P > \mathbf{U}_{Producer} \bullet G_P$
  - If $\mathbf{U}_{FIFO} > 0$,
    $i_C > \mathbf{U}_{Consumer} \bullet G_C$
- Have to add current limiters



Saturation Modeling (cont.)

- MOSFETs have linear and saturation regions
  - Linear = resistor
  - Saturation = current limiter
- PMOS: set $V_{th} = 0$, if $\mathbf{U}_{FIFO} \le 0$
  - $i_P = -I_{DS} = (k/2)V_{GS}^2$,
    saturation: $V_{DS} \le V_{GS} \rightarrow V_{GS} = -\mathbf{U}_P$
  - Want: $i_P = \mathbf{U}_P G_P \rightarrow k = 2G_P/\mathbf{U}_P$
- Else linear: $V_{DS} > V_{GS}$
  - $i_P = k(V_{GS}V_{DS} - (V_{DS}^2)/2) =$
    $-(2G_P V_{DS} - (V_{DS}^2)/(2\mathbf{U}_P))$
  - Want: $i_P = (\mathbf{U}_P - \mathbf{U}_{FIFO})G_P = -V_{DS}G_P$



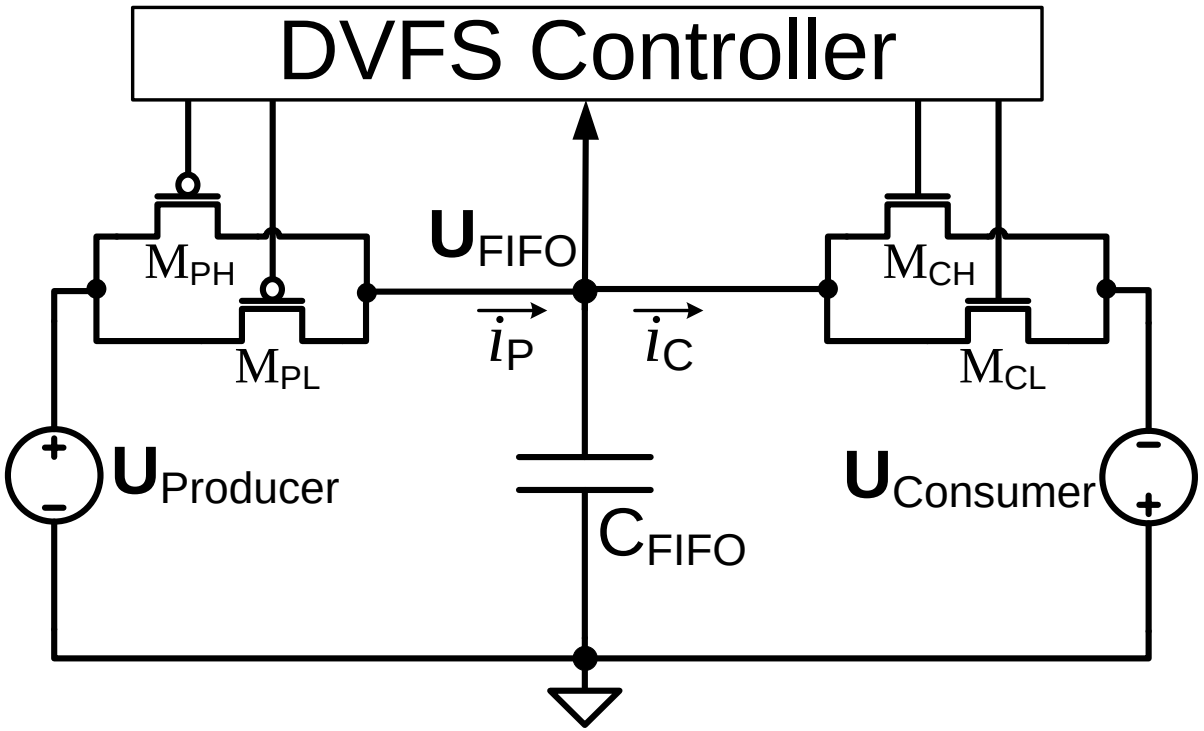


Improved Model

- Make a new MOS model in SPICE to accommodate modified linear region
- Can adjust conductance by changing W/L
  - Switch between two MOSFETs—*High* and *Low*

PMOS (producer)
M_{PH}: $(W/L)_{High}$
M_{PL}: $(W/L)_{Low}$

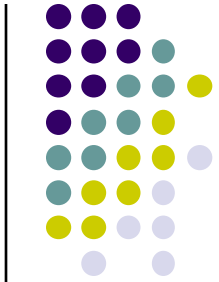
NMOS (consumer)
M_{CH}: $(W/L)_{High}$
M_{CL}: $(W/L)_{Low}$



Outline

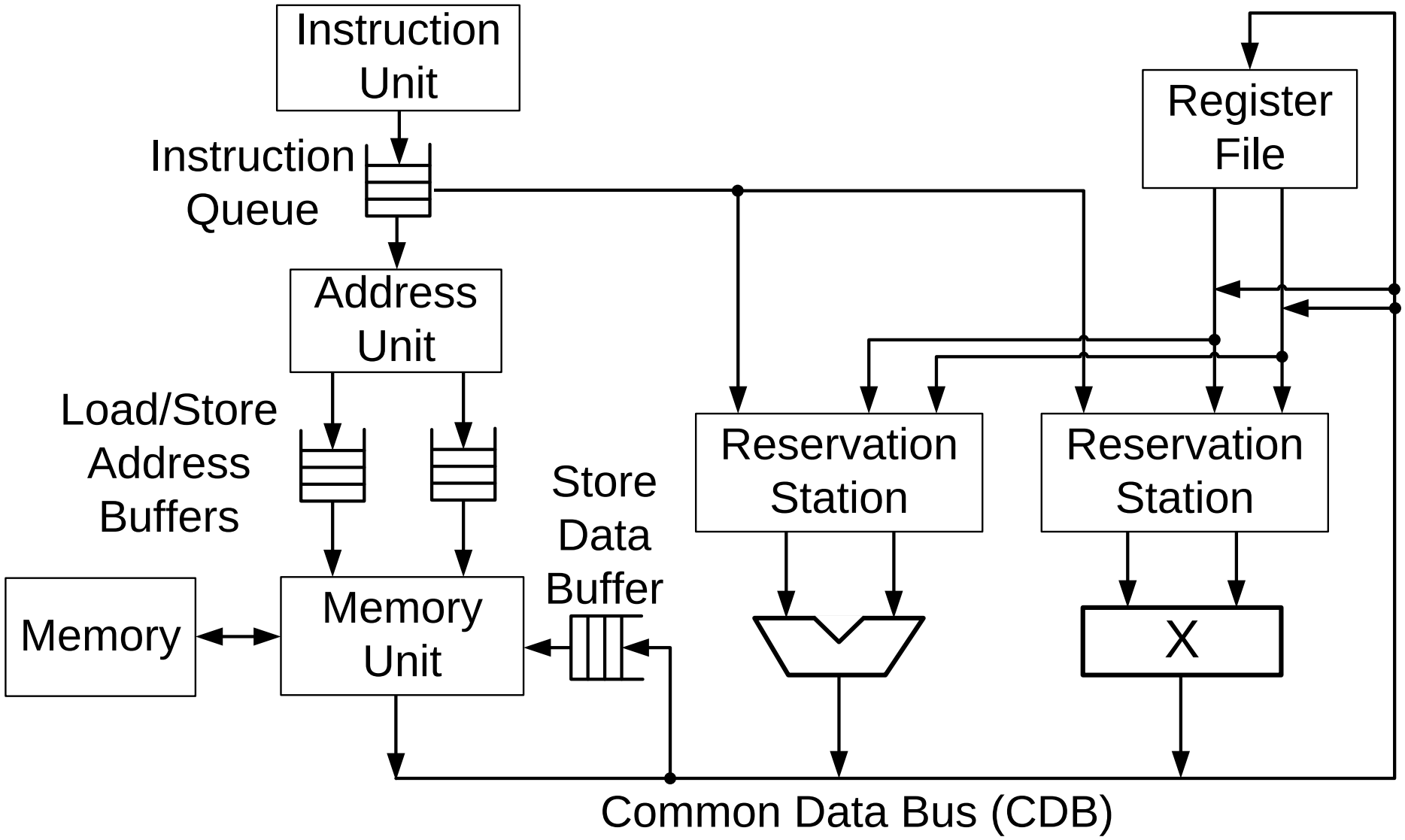
- Motivation
- Circuit Model of Two Cores
- DVFS Control and Results
- Improved Model
- **Tomasulo's Algorithm**
- Conclusions

Tomasulo's Algorithm

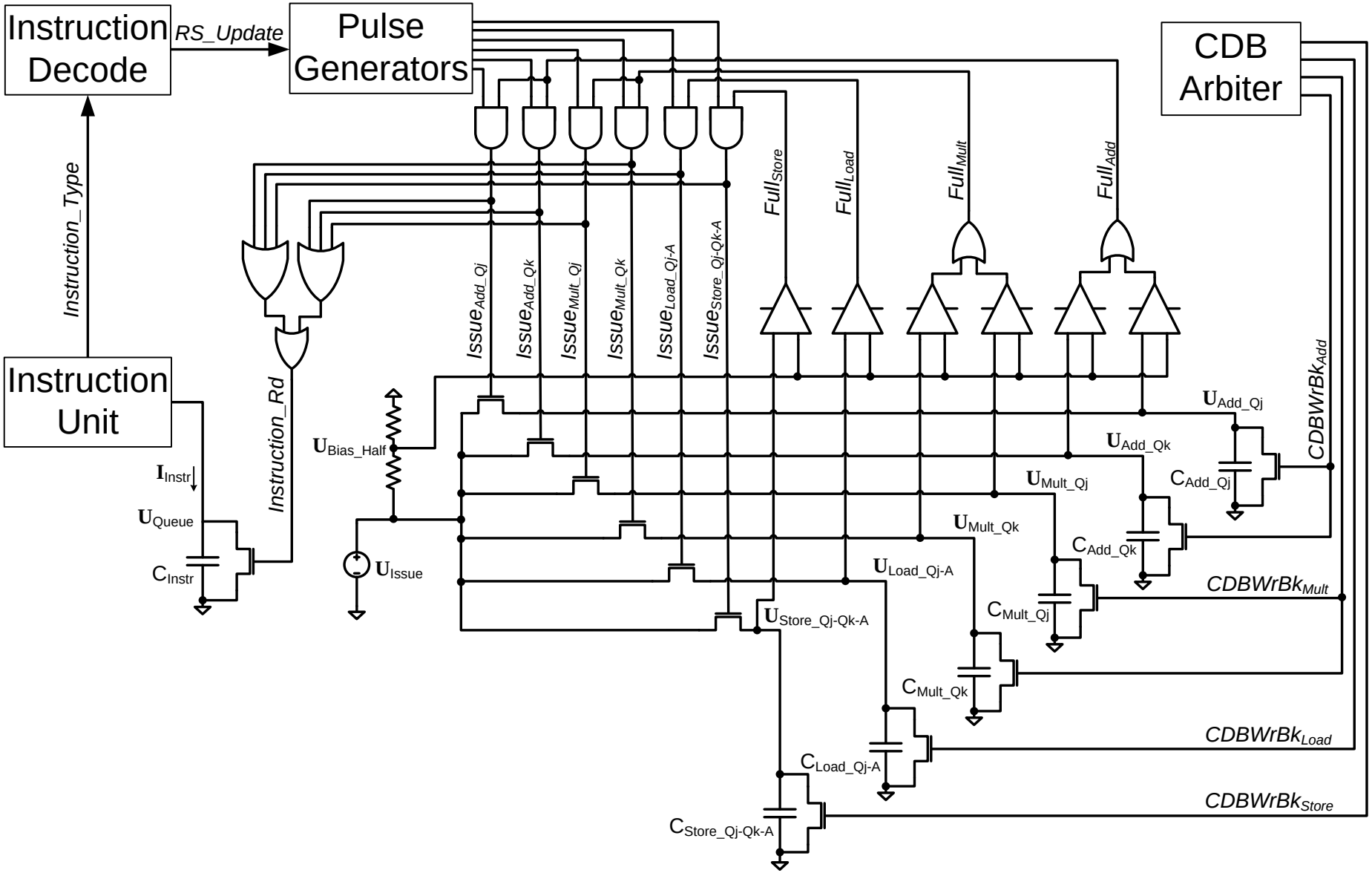


- Used in the IBM 360/91 floating-point unit to allow out-of-order execution
- Tomasulo's algorithm enables out-of-order execution through register renaming via reservation stations and load/store buffers
- Let us see if we can model this algorithm…
  - How much can be abstracted into a circuit element or logic gate?

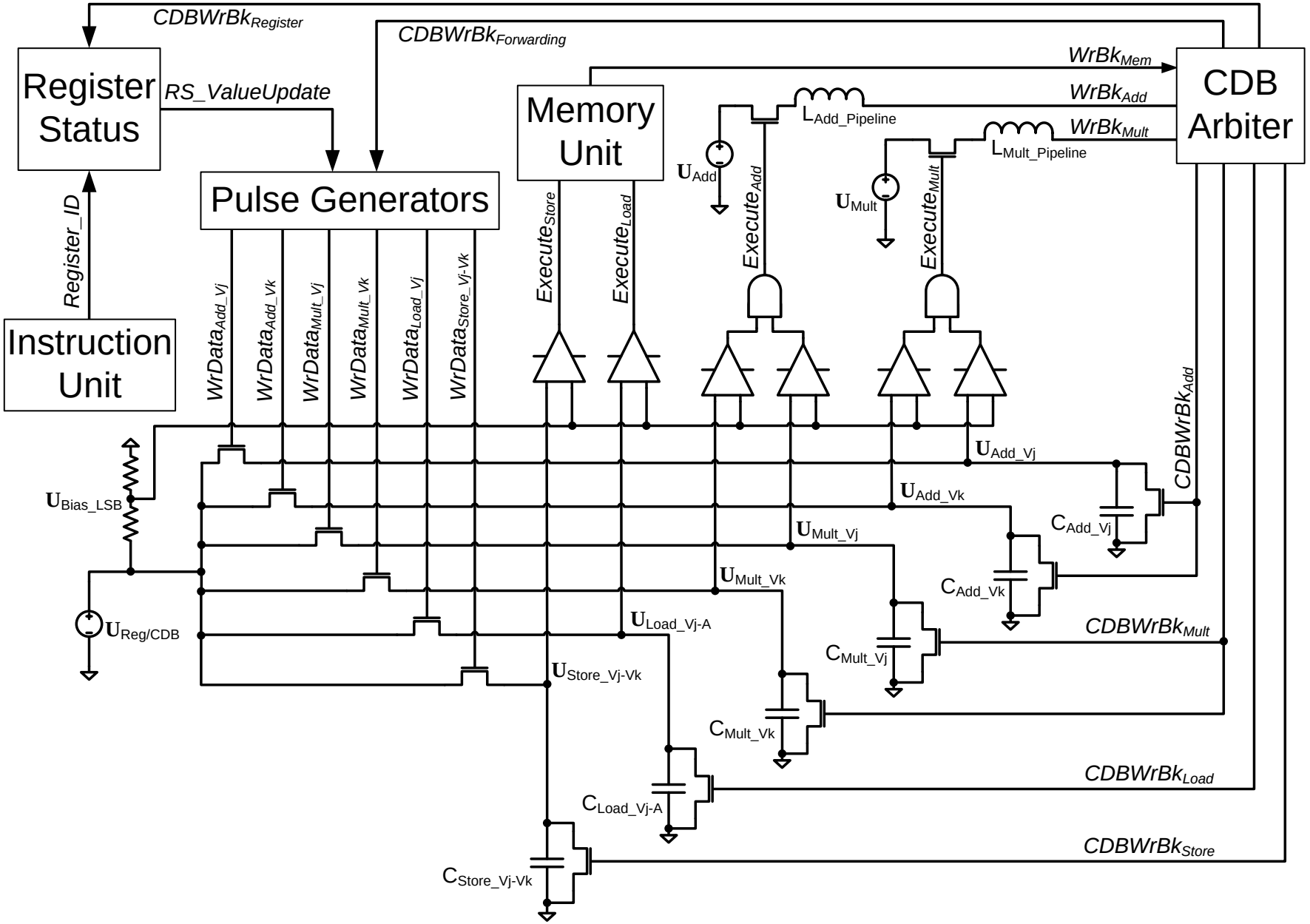
Tomasulo's Algorithm (cont.)



Tomasulo's Circuit



Tomasulo's Circuit (cont.)



Outline

- Motivation
- Circuit Model of Two Cores
- DVFS Control and Results
- Improved Model
- Tomasulo’s Algorithm
- **Conclusions**

Conclusions

- System level modeling with circuit equivalents
  - Reuse circuit design principles and CAD
  - Easily translates into control theory or signals and systems analysis
  - Reuse mixed-signal simulation tools to develop higher level systems
- Optimal DVFS requires a **holistic** approach from the circuit layer to application layer
  - Circuit modeling can ease translation between layers
- The physical world can be modeled as circuits
  - Streamlines development of cyber-physical systems

Acknowledgements

- ST Microelectronics
- NSF Grant 430090, 0903549 and CAREER award 546907
- Intel
- SRC GRC Grant 1598, 1971 and CSR Grant 1659
- Intellasys
- UC Micro
- SEM
- J.-P. Schoellkopf, K. Torki, S. Dumont, Y.-P. Cheng, R. Krishnamurthy and M. Anders