Using M/G/1 Queueing Models with Vacations to Analyze Virtualized Logic Computations

Michael J. Hall
VelociData, Inc.
St. Louis, MO, USA

Roger D. Chamberlain
Washington University in
St. Louis, MO, USA

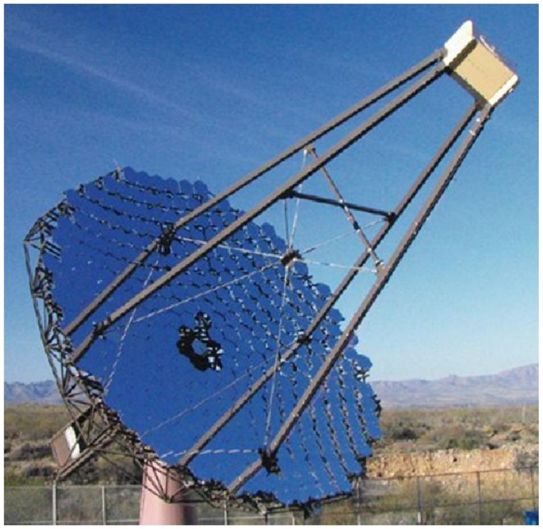
October 19, 2015

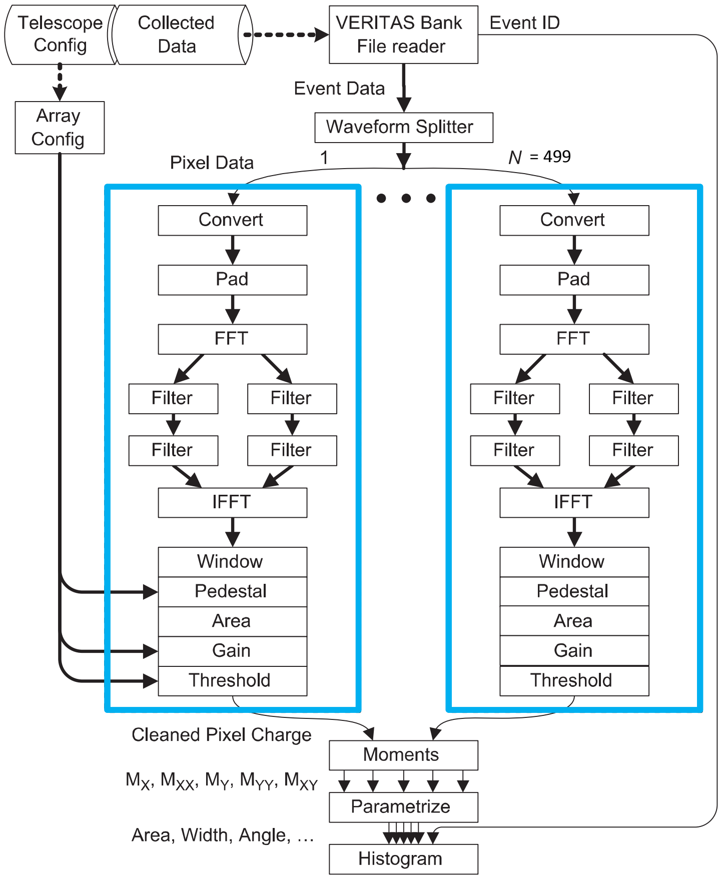
Funded by NSF, Exegy, Inc., and VelociData, Inc.

33rd IEEE
International
Conference on
Computer Design
ICCD 2015

Motivation for virtualized logic computations: An example big computation

- Telescope application with N=499 pixels
- Each pixel requires a channel of computation
- Replicating logic requires much hardware
- More hardware = more cost

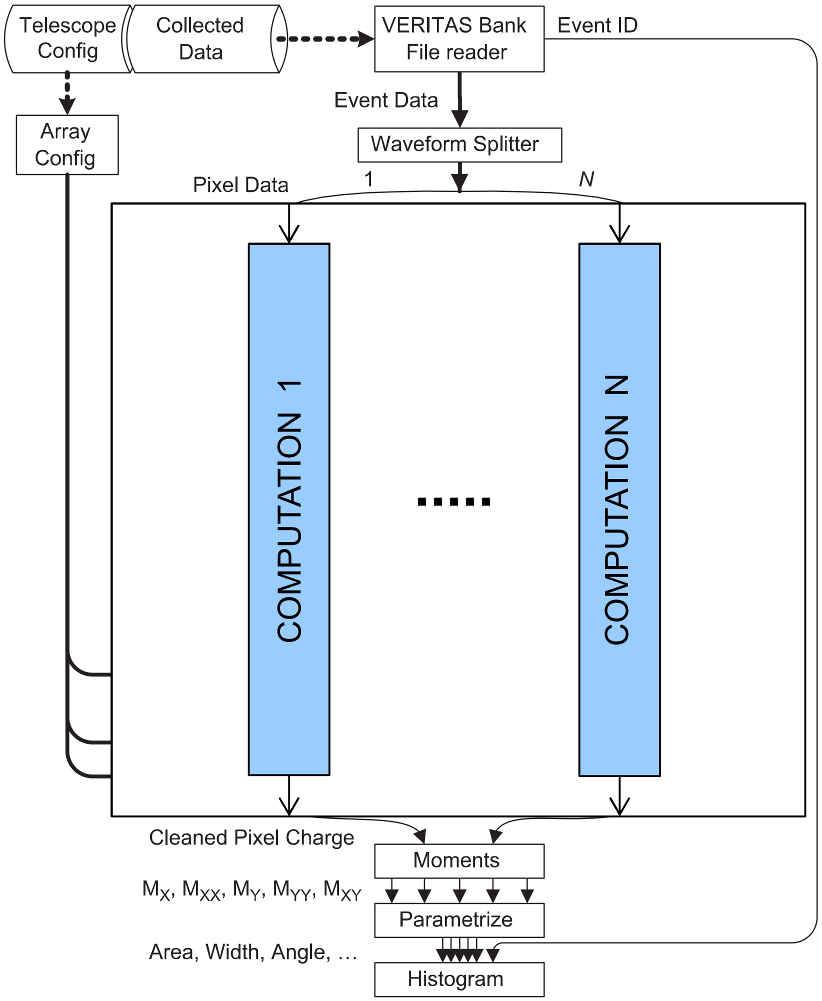




VERITAS gamma-ray signal processing pipeline in X.

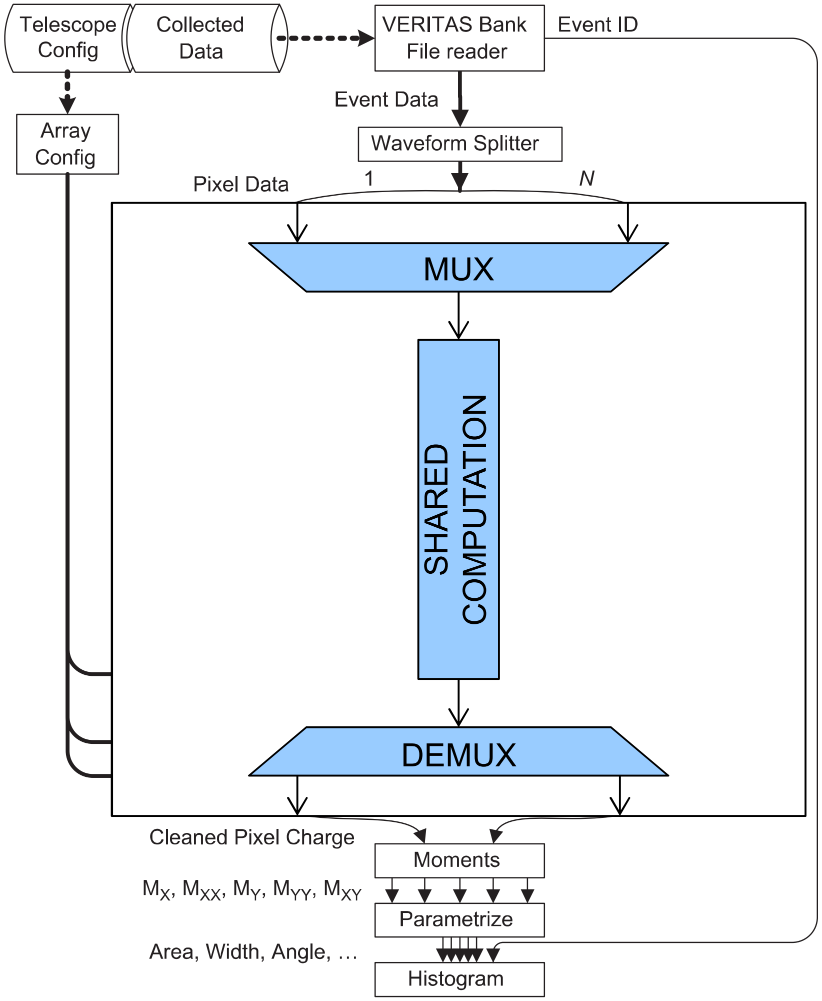
[Tyson et al. 2008]

Example big computation



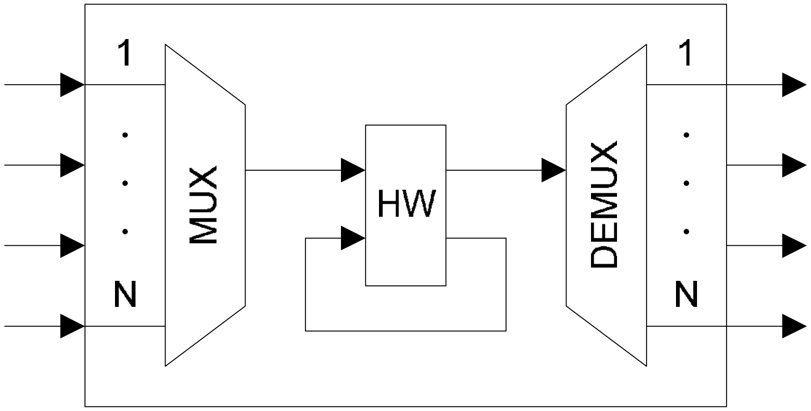
VERITAS gamma-ray signal processing pipeline in X.

Example big computation



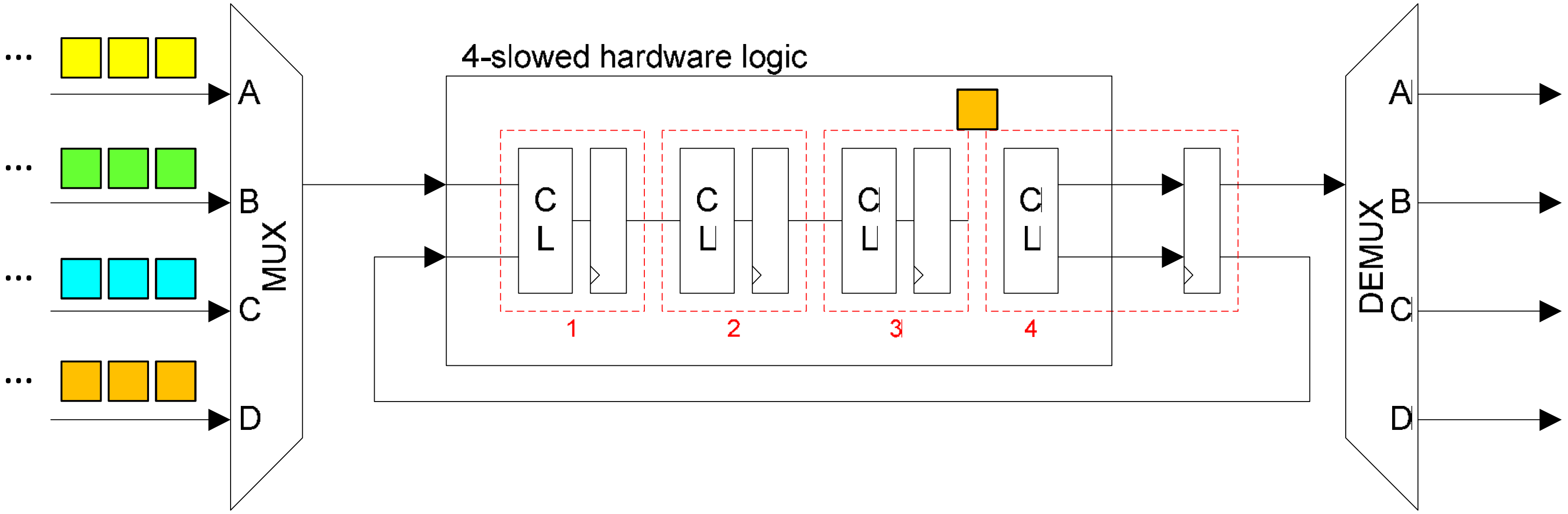
VERITAS gamma-ray signal processing pipeline in X.

Hardware virtualization



- Hardware virtualization for N distinct data streams that perform the same computation
- Interested in the case where there is feedback
- We will virtualize the hardware by applying a C-slow technique [Leiserson and Saxe 1991]
- We will derive queueing model equations to predict circuit performance
  - This is our main contribution

4-slow virtualization example



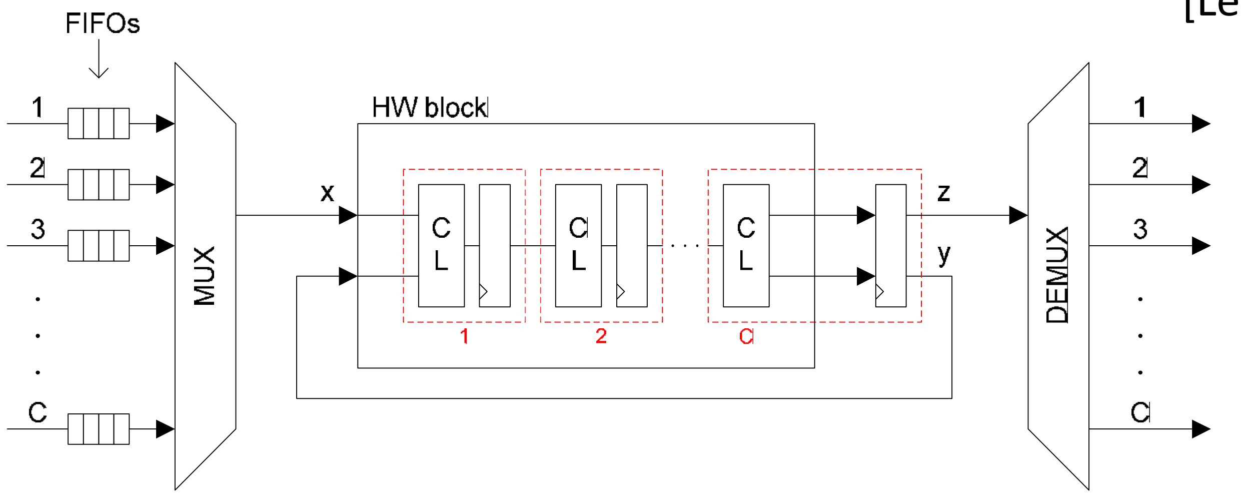
We will schedule the hardware using a *fixed, round-robin schedule*.

Time

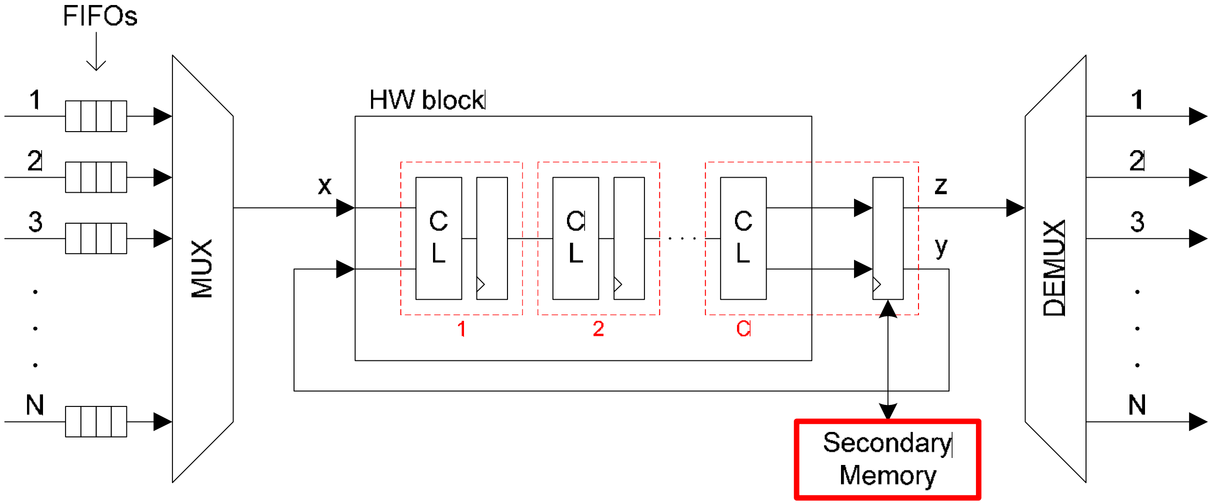
Stream	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	🟨				🟨											
B		🟩				🟩										
C			🟦				🟦									
D				🟧												

C-slow general virtualized hardware

[Leiserson and Saxe 1991]

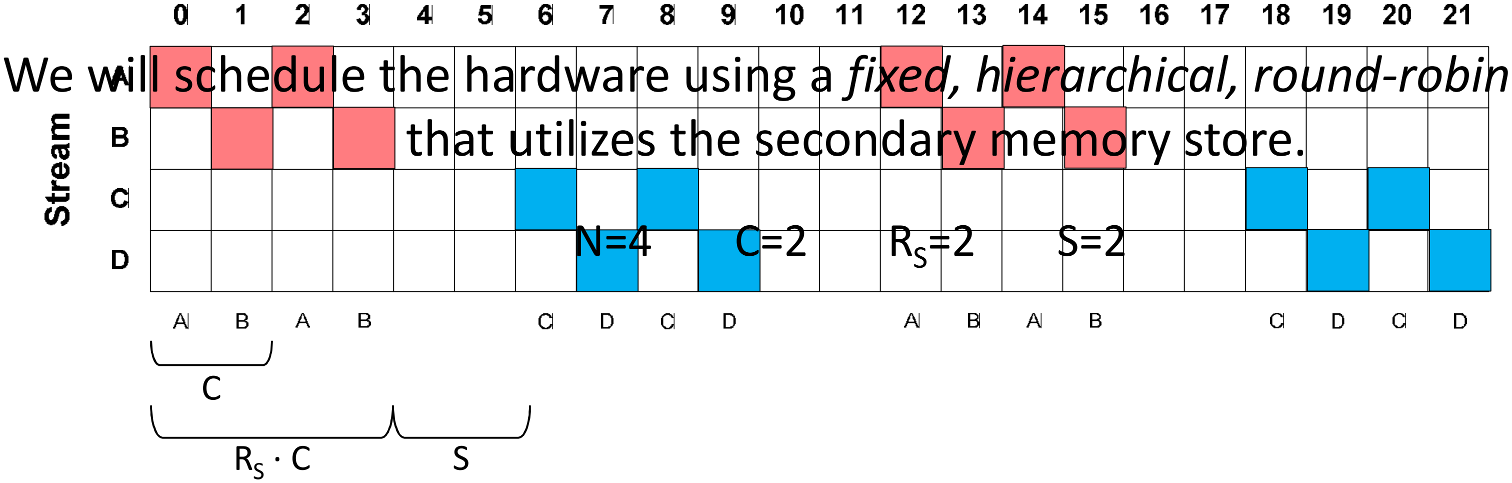


C-slow general virtualized hardware



Var.	Definition
N	Total contexts
C	Pipeline depth
R_S	Scheduling period
S	Cost of context switch

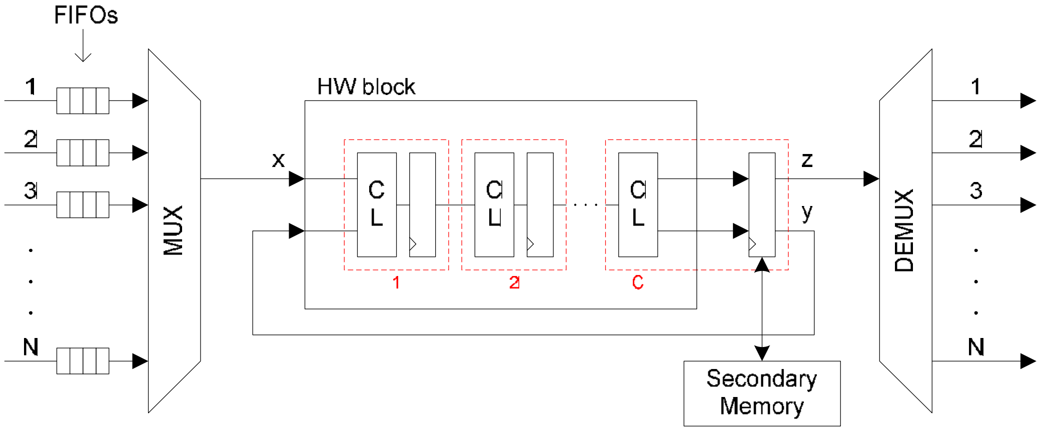
We will schedule the hardware using a *fixed, hierarchical, round-robin schedule* that utilizes the secondary memory store.



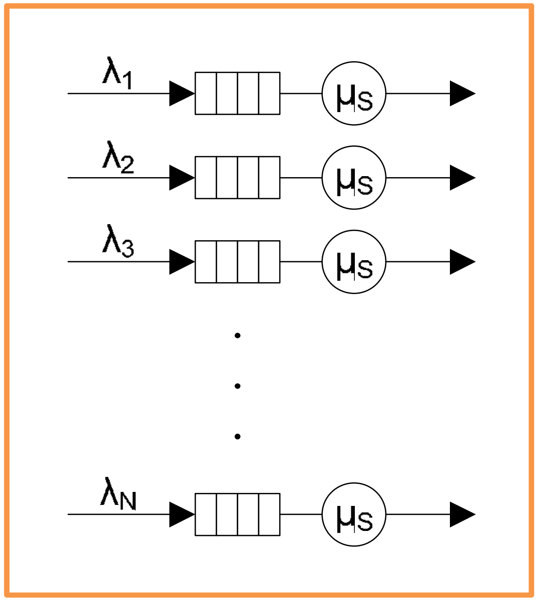
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
A	🟥		🟥										🟥		🟥							
B		🟥		🟥										🟥		🟥						
C							🟦		🟦										🟦		🟦	
D								🟦		🟦										🟦		🟦
	A	B	A	B			C	D	C	D			A	B	A	B			C	D	C	D

N=4 C=2 R_S=2 S=2

Queueing model

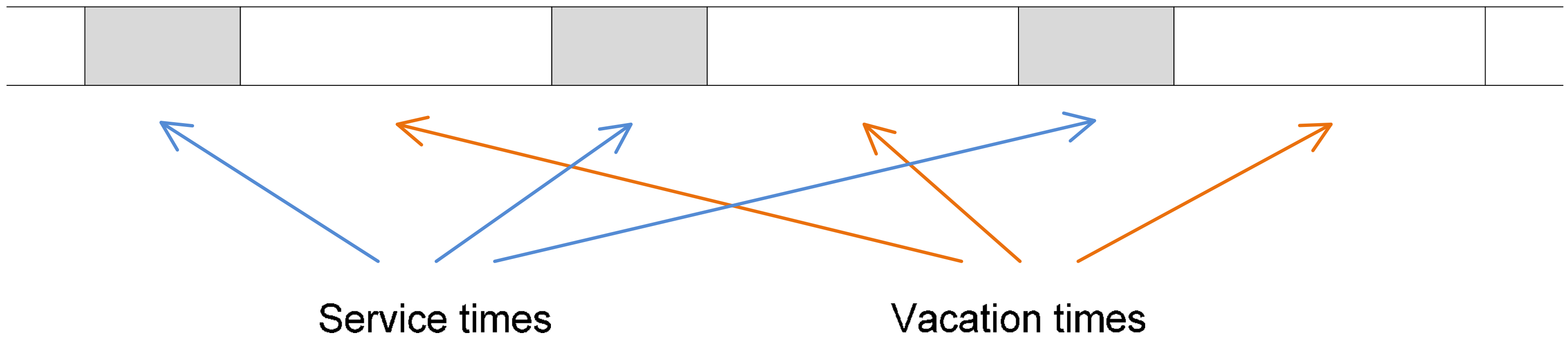


Queueing model



- Each queueing station is modeled as an M/G/1 queueing model with vacations
- M/G/1 is *M*arkovian, or memoryless, arrival process; *G*eneral service process; and *1* server

Service and vacation time modeling



- Service and vacation times are regular and deterministic
- This is for 1 queueing station, or virtual instance of the hardware logic

Model definition

Tput, Latency, Occupancy = f (Circuit, Tech, C, N, S, R_S, λ)

Variable	Definition
Circuit	Logical circuit description (e.g. AES-256)
Tech	Target technology (e.g. FPGA or ASIC)
C	Pipeline depth (also represents the number of fine-grain contexts)
N	Total number of contexts (requires secondary memory if N > C)
S	Cost of a context switch (to/from secondary memory)
R_S	Scheduling period (number of rounds of C contexts that execute before doing a context switch to secondary memory)
λ	Arrival rate (e.g. data elements per second)

Performance model

Total achievable throughput:

$$T_{TOT} = \frac{R_S}{(R_S + S/C) \cdot t_{CLK}}$$

Total wait time (latency):

$$W_T = \frac{\lambda \overline{X^2}}{2\left(1-\rho\right)} + \frac{\overline{V}}{1-\rho} + \overline{X}$$

Number in queue:

$$N_q = \frac{\lambda^2 \overline{X^2}}{2\left(1-\rho\right)} + \frac{\lambda \overline{V}}{1-\rho}$$

Variable	Definition
C	Pipeline depth
N	Number of streams
S	Context switch cost
R_S	Scheduling period
λ	Mean arrival rate
ρ	Utilization
$\overline{X}$	Mean service time
$\overline{X^2}$	Service time second moment
$\overline{V}$	Mean vacation waiting

Ways to use the model in design

Model definition:

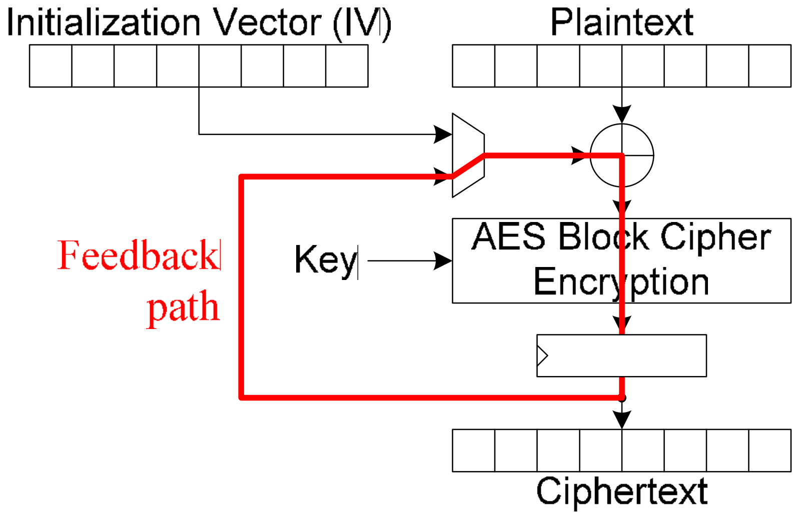
$$\text{Tput, Latency, Occupancy} = f\,(\text{Circuit, Tech}, C, N, S, R_S, \lambda)$$

- Subset of parameters are given
  - E.g., Circuit, Tech, N, S
- Remainder under control of designer
  - E.g., C, R_S, λ
- Design goal
  - E.g., Latency

Variable	Definition
C	Pipeline depth
N	Number of contexts
S	Cost of a context switch
R_S	Scheduling period
λ	Arrival rate

Experimental setup

- AES-256 encryption application in CBC mode
- Fully unrolled, N_r = 14 rounds
- Targeting Xilinx Virtex-4 XC4VLX100 FPGA



Calibrated t_{CLK} model:

$$t_{CLK}\left(N_r, C\right) = \left[\frac{1.8 + 5.2 \cdot N_r}{C} + (2.56 - 0.038 \cdot N_r) \cdot (\ln C)^{0.7}\right] \text{ns}$$

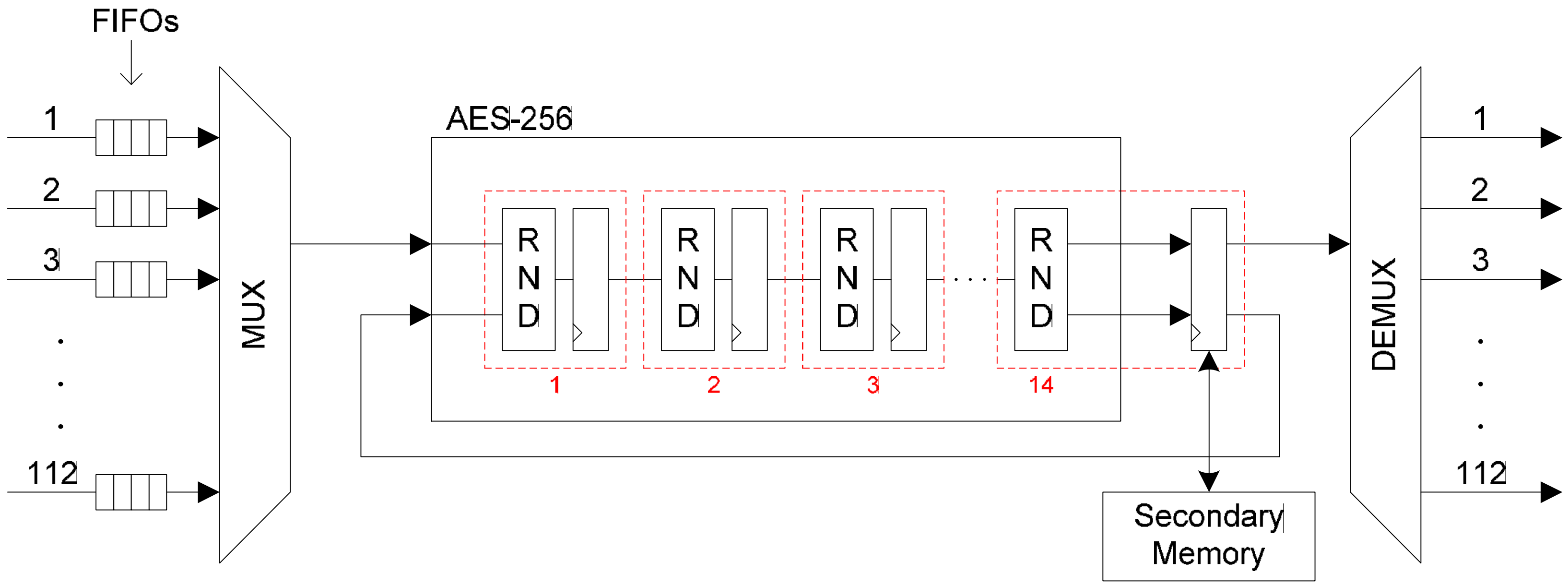
Fixed model parameters:

Pipeline depth, C = 14

Total contexts, N = 8C (or 112)

Context switch cost, S = 120 clock cycles

14-slow AES-256



- Each round (RND) performs a series of operations on a block of data propagating through to the output
  - Substitutions, shifts, multiplications, and logical operations
- We have 112 virtual copies

Total achievable throughput

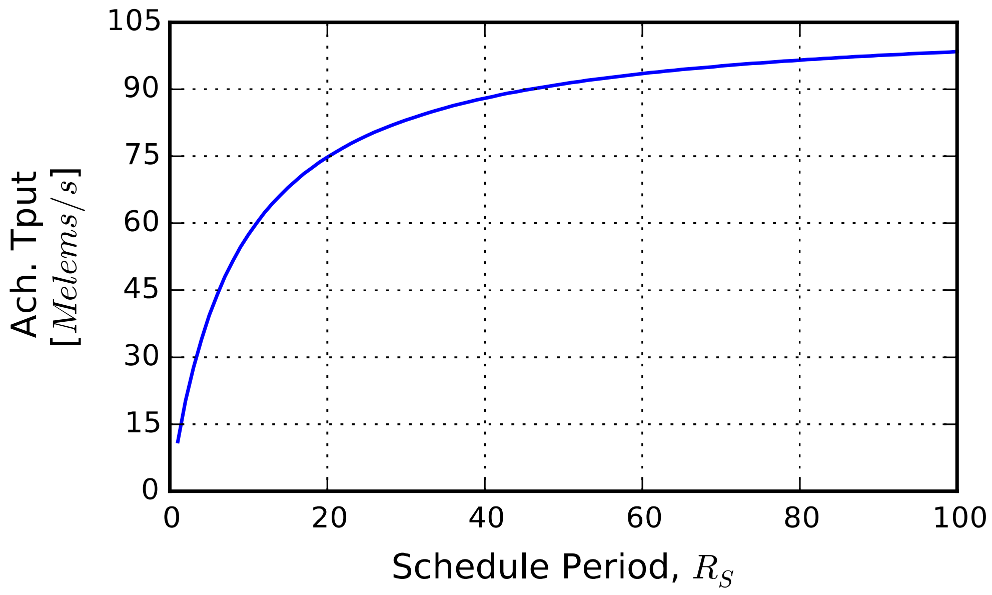
- Model parameters:

  C = 14

  N = 8C

  S = 120 clock cycles

- Sweep R_S



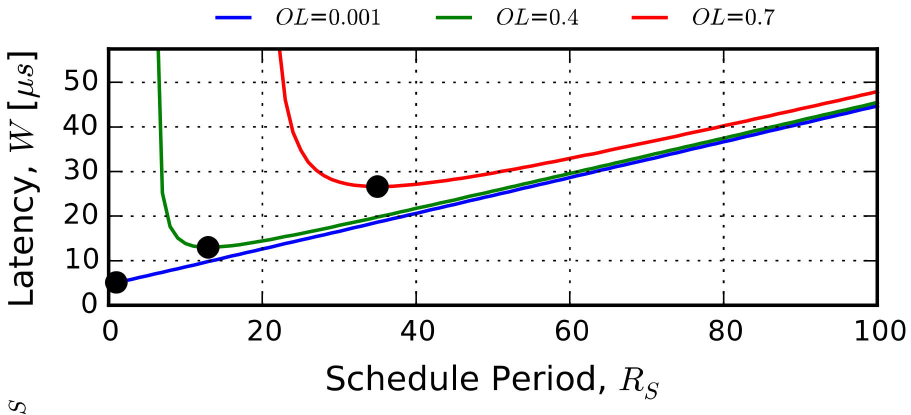
Latency prediction and optimization

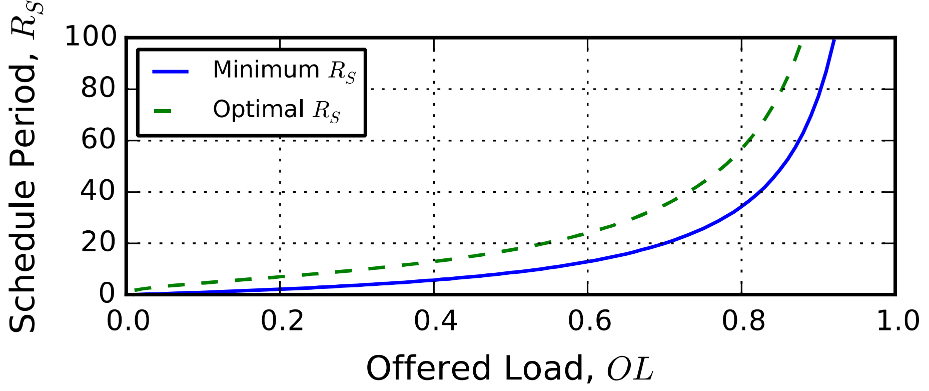
- Model parameters:

  C = 14
  N = 8C
  S = 120 clock cycles

- Sweep R_S

- Optimize Latency





$\mathrm{OL} \propto \lambda$

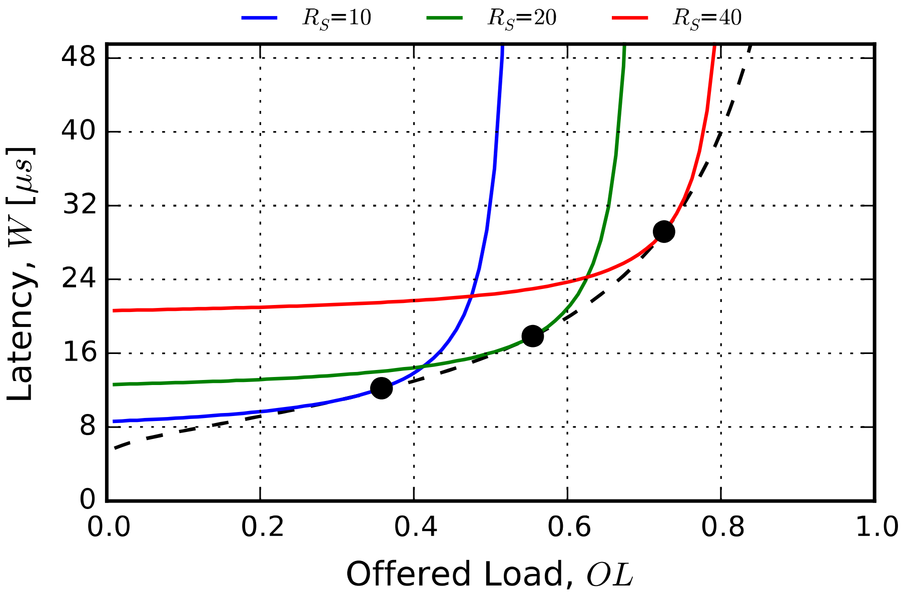
Latency prediction vs. offered load

- Model parameters:

  C = 14
  N = 8C
  S = 120 clock cycles

- Sweep *OL*

- Optimize Latency



OL ∝ λ

Conclusion

- Developed a vacation-based M/G/1 queueing model for virtualized custom logic functions
- The model predicts throughput, latency, and queue occupancy
- Inputs to the model are the circuit, technology, clock period, pipeline depth, number of contexts, schedule period, input arrival rate, and overhead of a context switch

Future directions

- Evaluate the assumption that the input process is Poisson; many real systems may act differently by buffering up data and sending in bursts

- Extend the model to support additional scheduling algorithms; the current schedule is not work-conserving, meaning that an empty input queue will still get scheduled

Questions?