Simulation Overview

Chang-Gun Lee (cglee@snu.ac.kr)

Assistant Professor

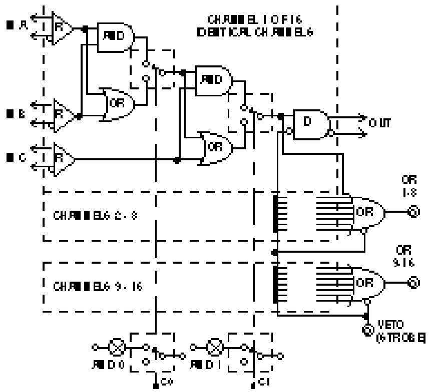
The School of Computer Science and Engineering

Seoul National University

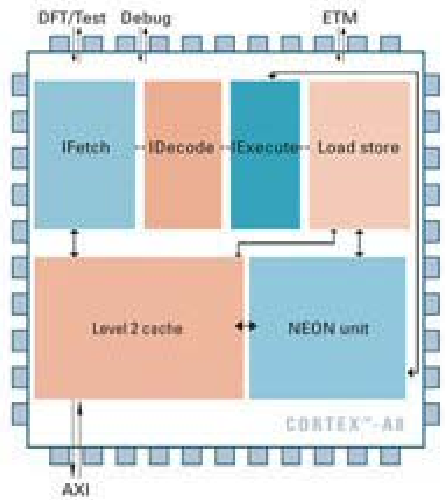
What is “Simulation”?

- Doing something without the real-system just like the real-system do
  - mentally?
  - using computer?
- What is “just like the real-system do”?
  - How much exact we want?
  - Level of abstraction
- Why?
  - We can evaluate the performance in advance before actual implementation
  - Easy to change
- Simulation is …
  - analysis tool for predicting the effect of changes
  - design tool to predict the performance of new system

What to Simulate?

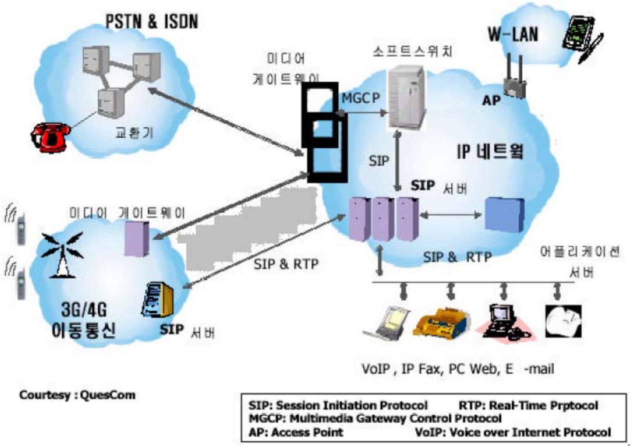


Digital Circuit

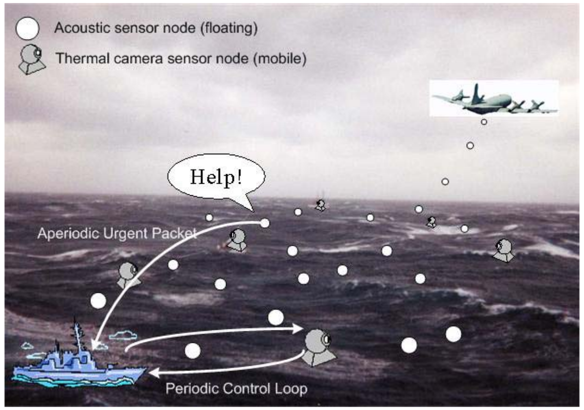


Processor Core

Embedded System

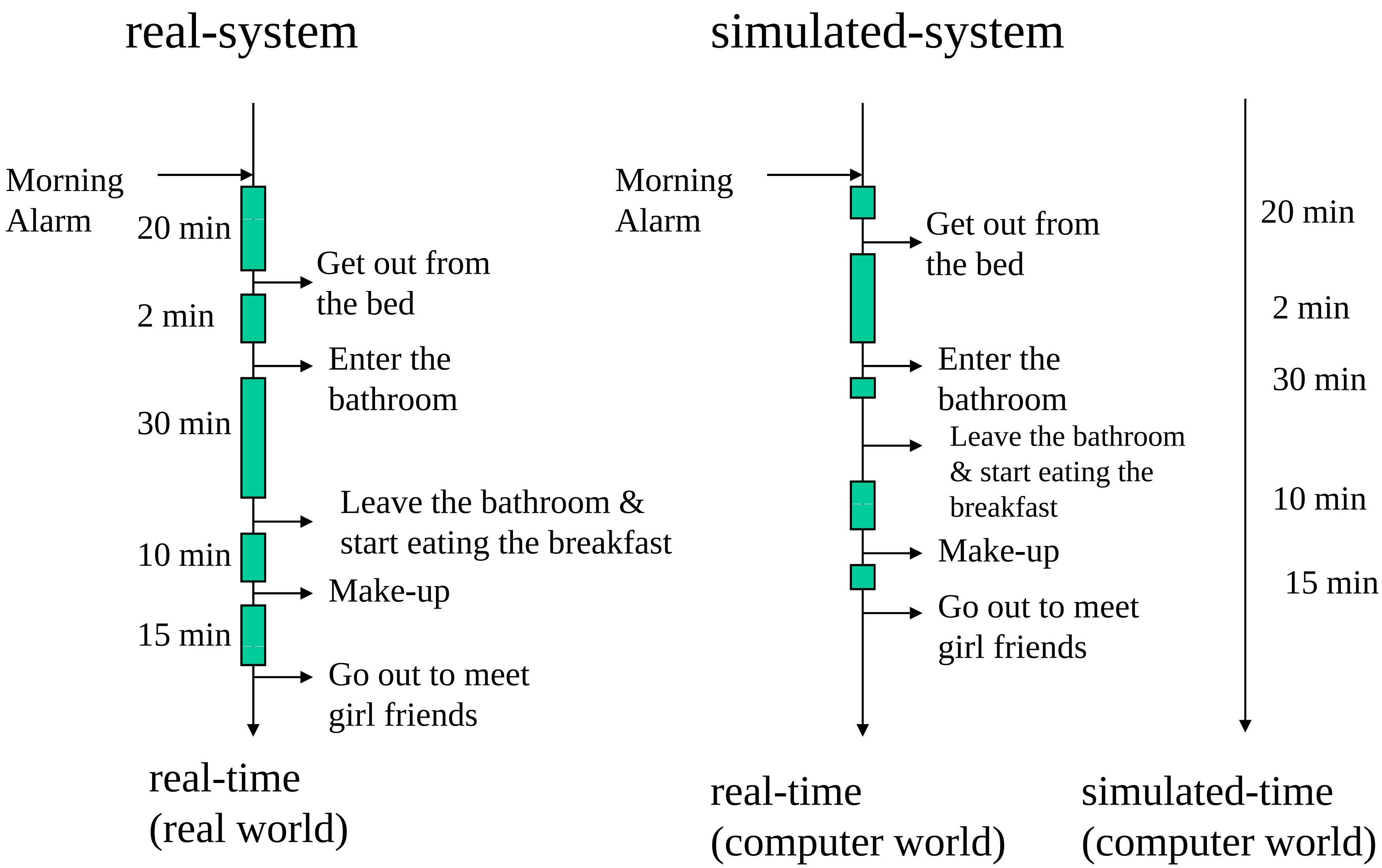


Internet



Wireless Sensor Network

Computer-based Simulation



Two things from the previous slide

- Real-Time vs. Simulated-Time
- Level of Abstraction

Real Time (Wall-Clock Time) vs. Simulated Time

- How the time goes in the simulated world?
  - Can be faster or slower than the real-time
  - The simulated-time progresses following the event occurrence simulating the time-progress in the real world

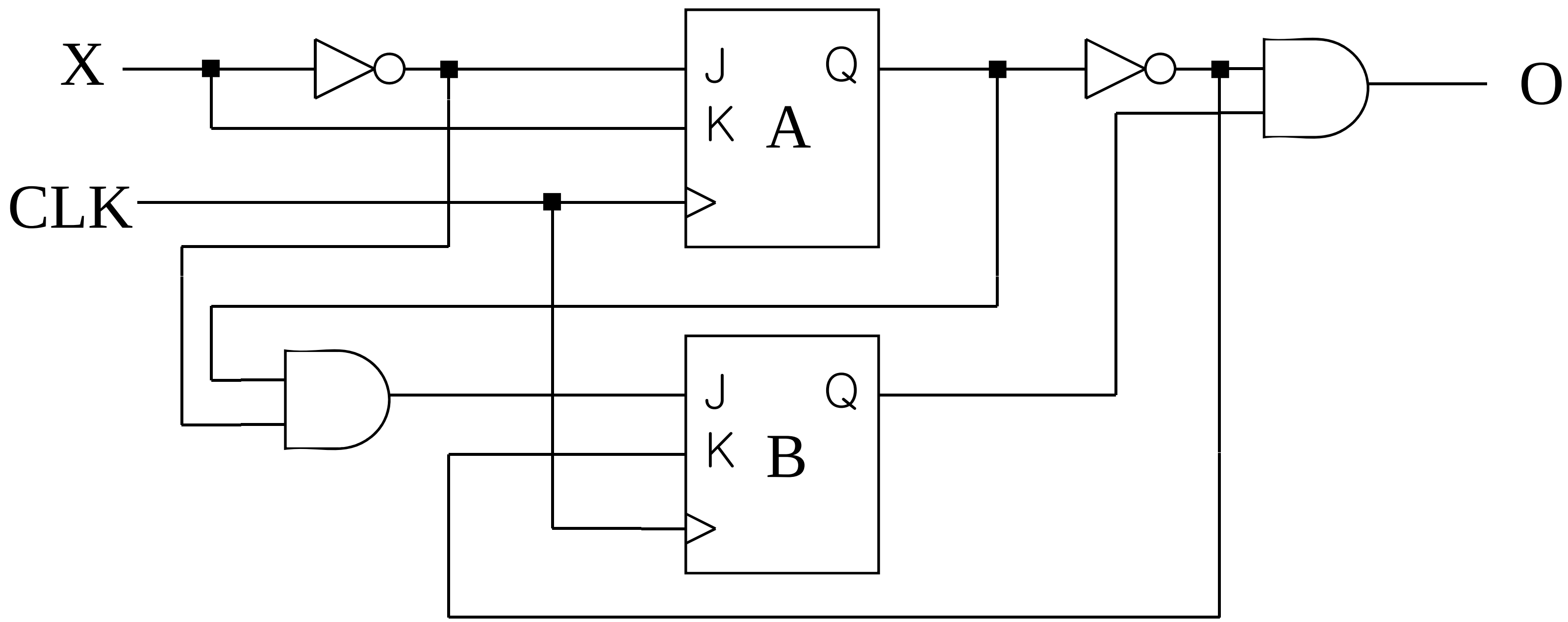
Level of Abstraction

- What you are doing in the bathroom
  - Don’t care
  - We want to model the detail of what happen in the bathroom.
- Depends on the needs of analysis
  - We only care the time spent at home before leaving
  - We want to know which action is most time consuming
- Typical levels of abstraction
  - Circuit-level simulation
  - Instruction-level simulation
  - Event-level simulation

Xilinx demo
(circuit-level simulation)

- A sequential circuit
  - Logical functionality
  - Timing

Example: Analysis of Clocked Synchronous Circuit



- A systematic way is necessary

Three Step Approach

- Step 1: Equation (excitation and output)
- Step 2: Table (state and output)
- Step 3: State diagram

Step 1: Excitation and Output Equations

- Derive Excitation and Output Equations from the schematic

$$J_A = \overline{X}, K_A = X,$$

$$J_B = Q_A \overline{X}, K_B = \overline{Q_A},$$

$$O = \overline{Q_A} Q_B$$

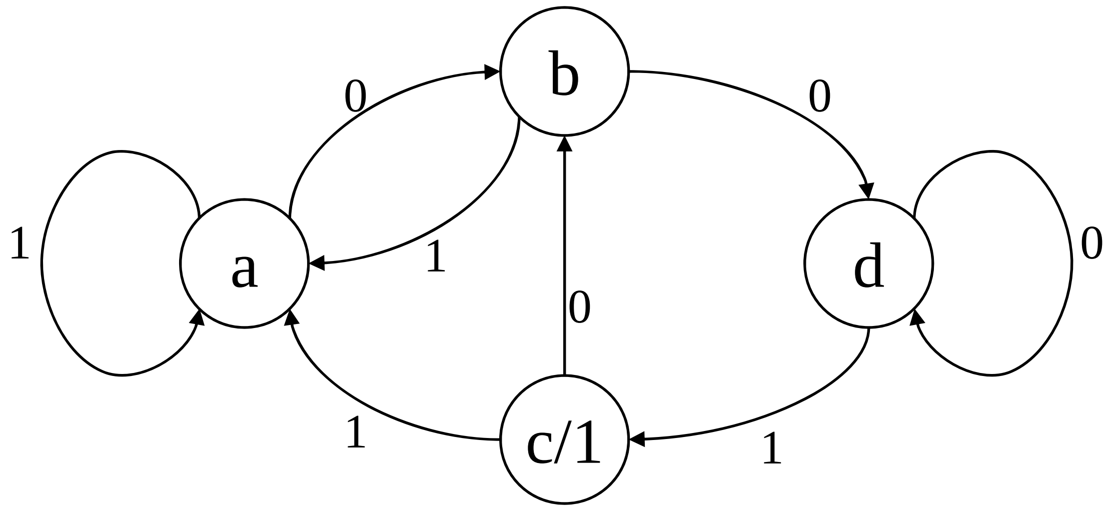
Step 2: State/Output Table

- Fill in the table from the previous equations

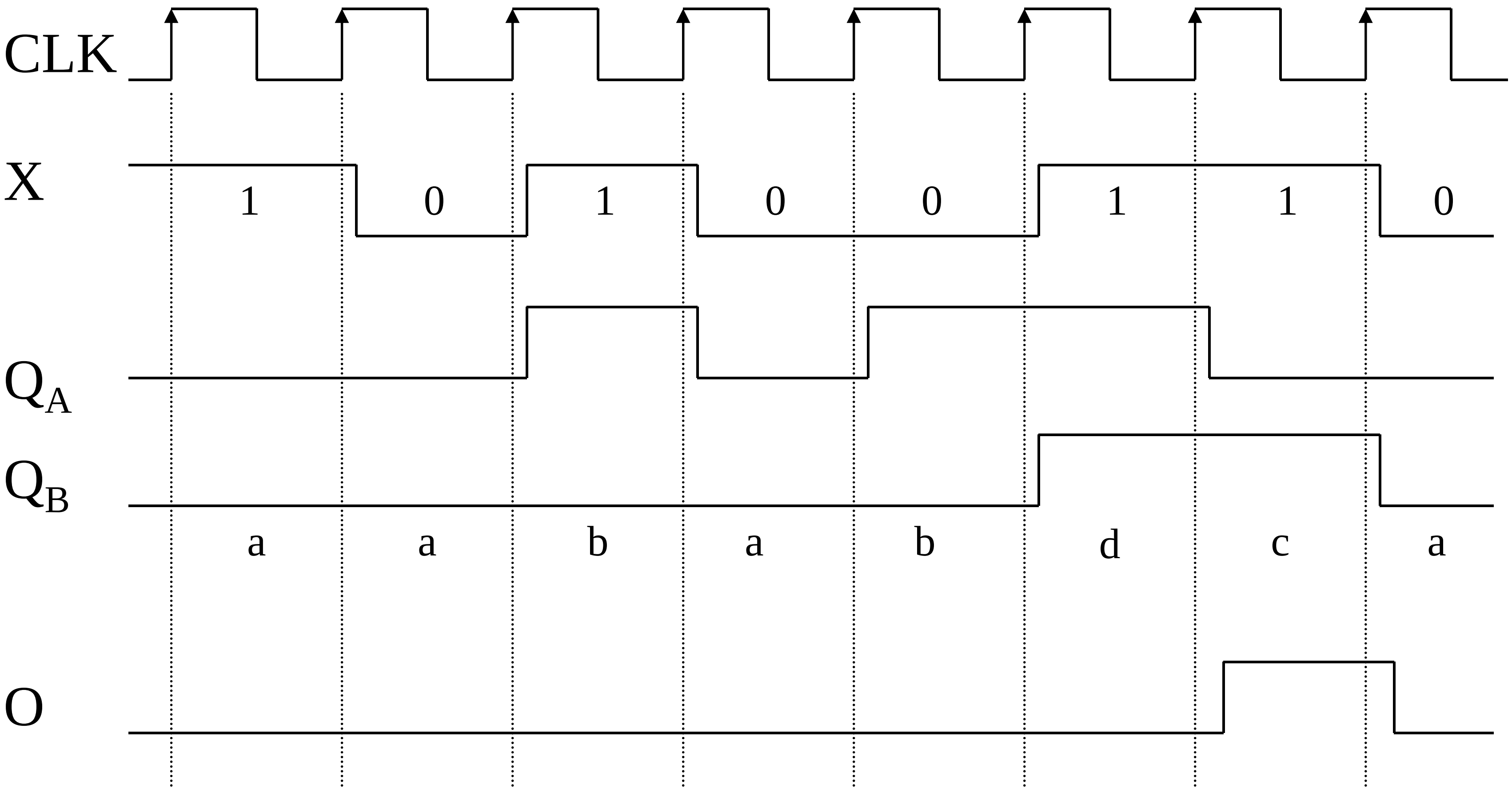
	P.S.		Input	Output	Excitation				N.S.	
	QB	QA	X	O	JB	KB	JA	KA	QB	QA
a	0	0	0	0	0	1	1	0	0	1
	0	0	1	0	0	1	0	1	0	0
b	0	1	0	0	1	0	1	0	1	1
	0	1	1	0	0	0	0	1	0	0
c	1	0	0	1	0	1	1	0	0	1
	1	0	1	1	0	1	0	1	0	0
d	1	1	0	0	1	0	1	0	1	1
	1	1	1	0	0	0	0	1	1	0

Step 3: State Diagram

- Can you tell what this machine is doing?



Example

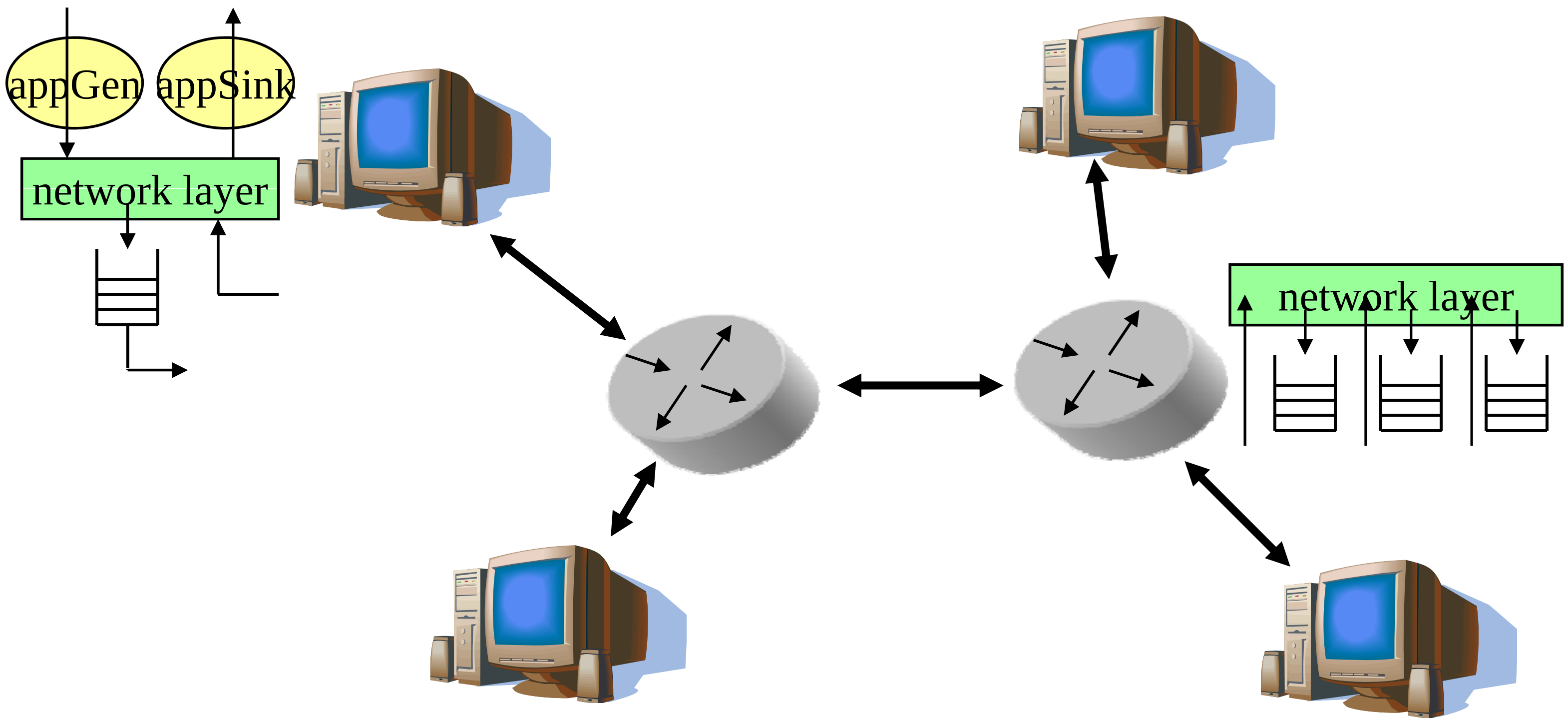


68HC11 Simulator demo (Instruction-level simulation)

- Read 4bit switches and display the read through 7-Segment element.
  - Instruction-level trace
  - Memory monitoring
  - Register monitoring

OMNet++ demo (“mySamples/psNetwork”) (Event-level simulation)

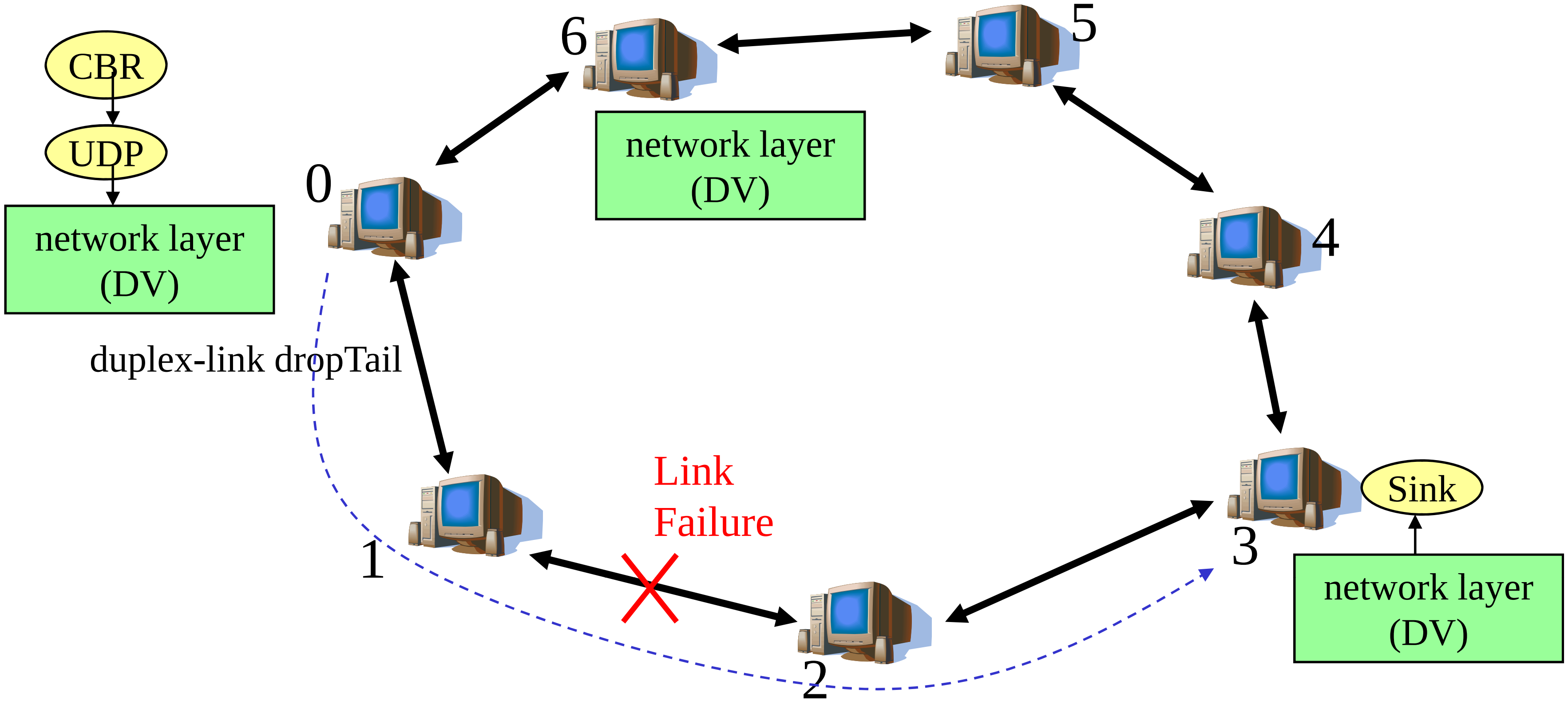
- A packet switching network with 2 routers and 4 hosts
  - Host: AppGen + AppSink + NetworkLayer + OutputQueue
  - Router: NetworkLayer + OutputQueues for ouput ports



- Observe output queue length, message hop count, and end-to-end delay
- Post visualization of data using “scalars” and “plove” tools
  - “scalars omnetpp.sca”, “plove omnetpp.vec”

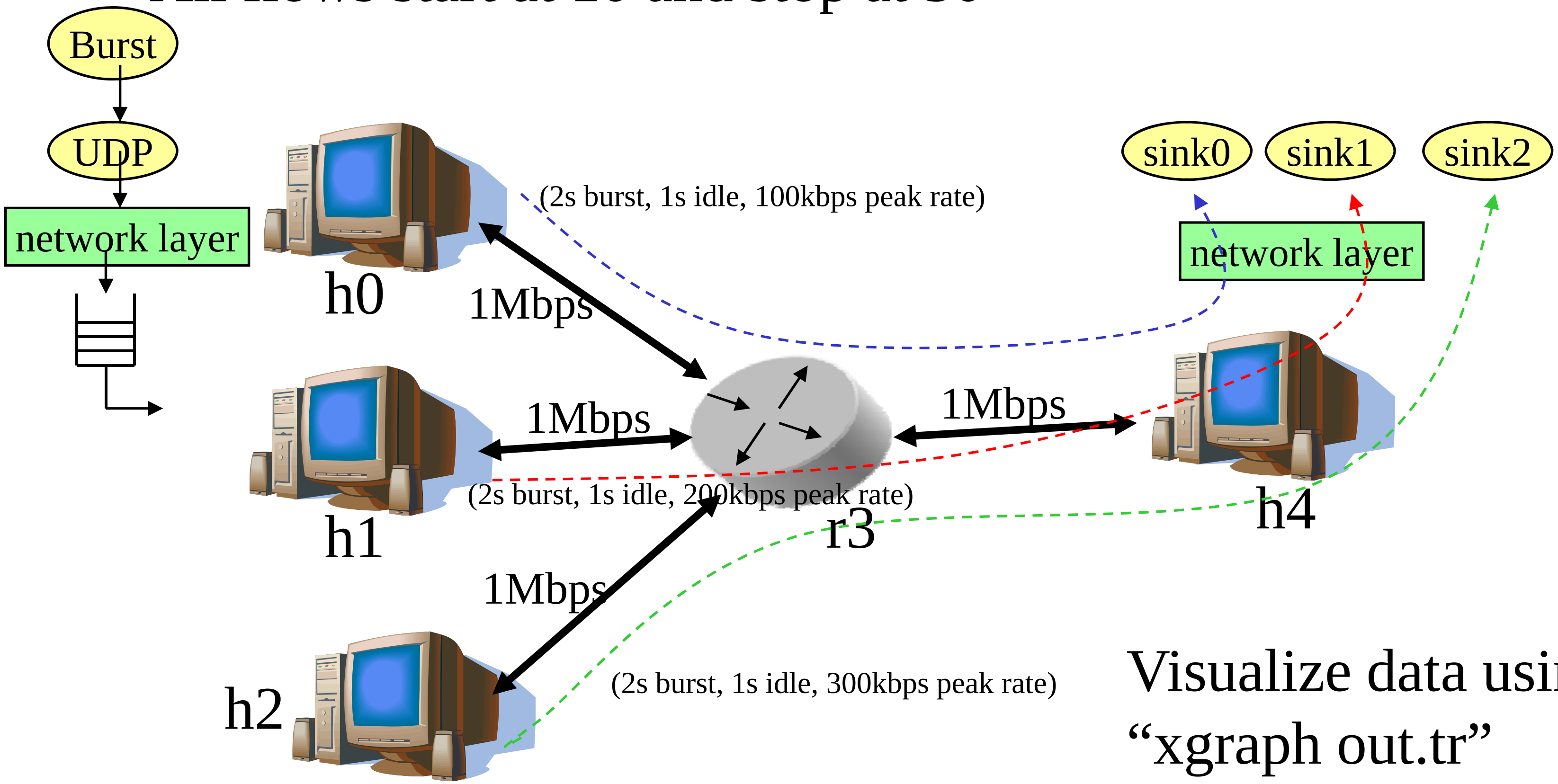
ns-2 demo ("ns example3.tcl") (Event-level simulation)

- A ring topology with 7 nodes
- Node 0 starts CBR traffic over UDP at time 0.5 sec, whose destination is Node 3
- Link between Node 1 and Node 2 downs during [1 sec, 2sec]
- Node 0's traffic stops at 4.5 sec



ns-2 demo ("ns example4.tcl")

- Three hosts (h0, h1, h2) send burst traffic to one sink (h4) via a router (r3)
  - Host: BurstTraffic + UDP + NetworkLayer + DropTailQueue
  - Router: NetworkLayer + DropTailQueues for ouput ports
  - Sink: Sink(LossMonitor) + NetworkLayer
- All flows start at 10 and stop at 50



Visualize data using "xgraph out.tr"

Workload Model

- In event-driven simulation, we have to characterize the workload
  - arrival rate of jobs
  - jobs' execution time in each resource
  - interactions among jobs
- Deterministic model
  - periodic arrivals
  - constant execution times
- Stochastic model
  - random arrivals (e.g., exponential inter-arrival time distribution)
  - random execution times (e.g., exponential execution times)
  - We use a random number generator for that

Disadvantage of Simulation

- Simulation requires special training
- There is an inevitable deviation from the reality
  - How to make the deviation acceptable?
- It is hard to validate the correctness of your simulation
  - How can you judge that your simulation is correct?

- Solution
  - Work hard in this class