

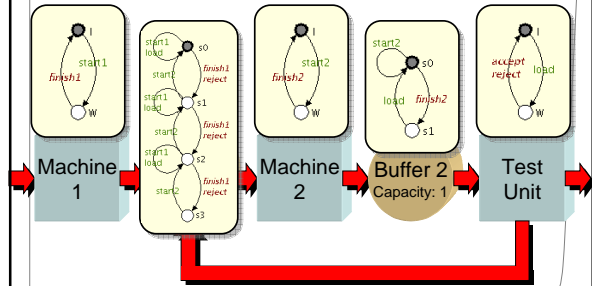
# COMP424/524-06A Topics in Software Engineering

Part I – Finite State Machines

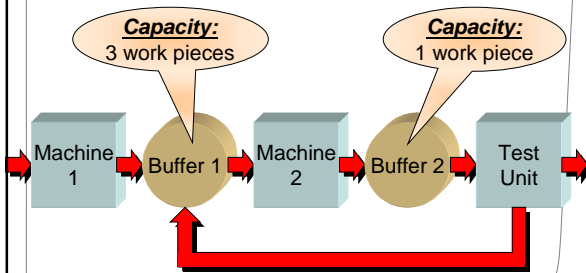
## 5. Blocking

Robi Malik

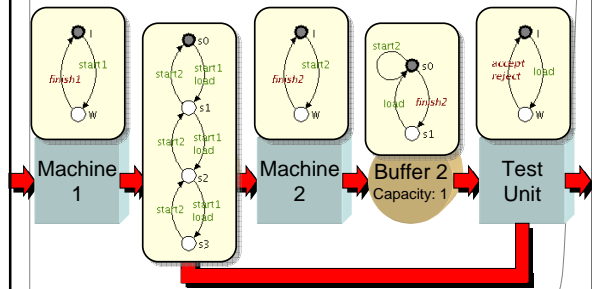
### Transfer Line: A Solution?



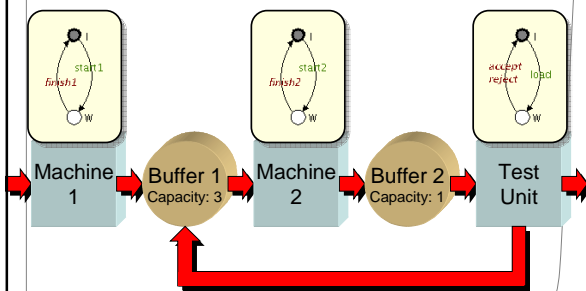
### Transfer Line – A New Example



### Transfer Line: A Solution?



### Transfer Line: Plants



### A Note on Controllability

*“If you never do anything,  
you can’t do anything wrong.”*

Controllability can easily be achieved by not permitting any controllable events.

## Nonblocking

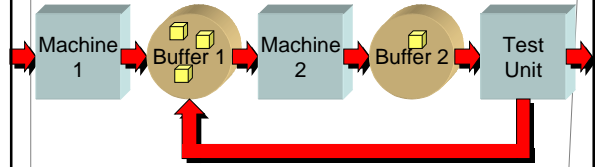
### Definition:

Let  $A = (Q, E, T, q_0, Q_m)$  be an automaton.  $A$  is called **nonblocking** if, for every reachable state  $q \in Q$ , there exists a path

$$q \xrightarrow{e_1} \dots \xrightarrow{e_n} q_m$$

that ends in a marked state  $q_m \in Q_m$ .

## A Deadlock



- Both buffers full.
- Test unit not started for fear it might *reject*.

## Nonconflicting

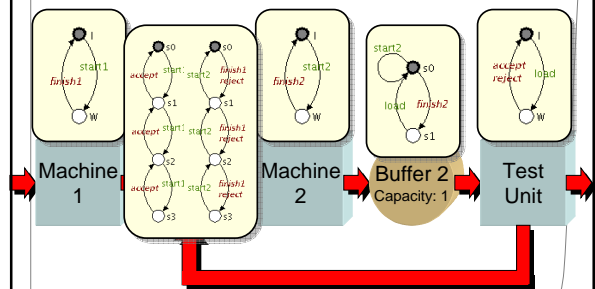
### Definition:

Let  $A_1$  and  $A_2$  be two automata.

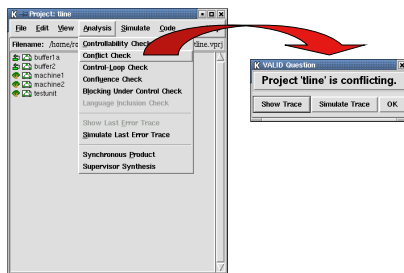
$A_1$  and  $A_2$  are called **nonconflicting**, if  $A_1 \parallel A_2$  is nonblocking.

Otherwise they are called **conflicting**.

## Transfer Line: A Solution?



## Checking for Conflicts



## Choosing the Marked States

The marked state is a configuration which the system always should be able to return to.

### Possibilities:

- Initial state
- All tasks completed
- All tasks in progress
- Some task is running

### Tip:

Sometimes you want to check several marking conditions ...