Department of Computer Science
National Tsing Hua University

CS 5244: Introduction to Cyber Physical Systems

Unit 14: Specification & Temporal Logic

(Ch. 12)

Instructor: Cheng-Hsin Hsu

Acknowledgement: The instructor thanks Profs. Edward A. Lee & Sanjit A. Seshia at UC Berkeley for sharing their course materials

When is a Design of a System “Correct”?

A design is correct when it meets its specification (requirements) in its operating environment

“A design without specification cannot be right or wrong, it can only be surprising!”

Simply running a few tests is not enough!

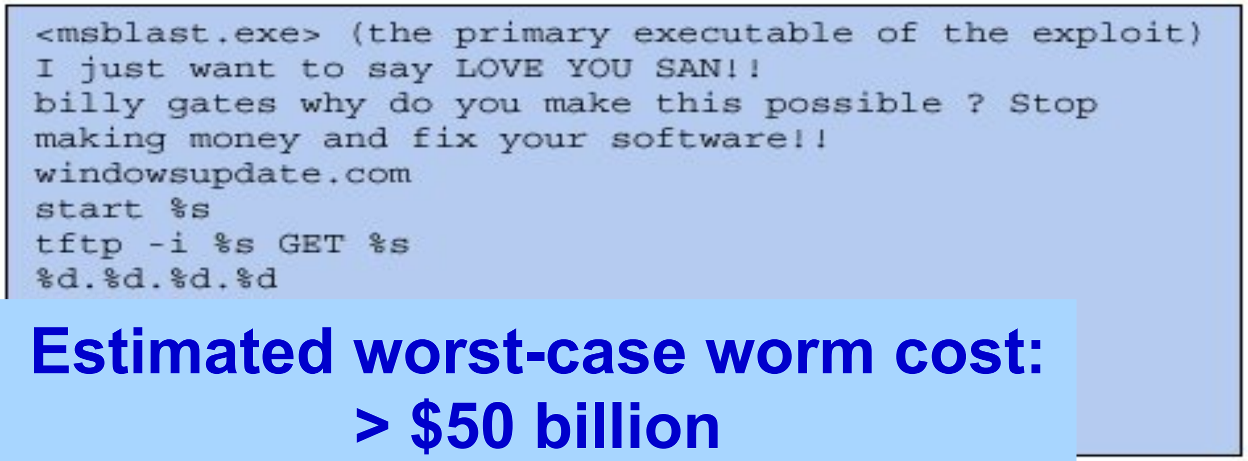
Many embedded systems are deployed in safety-critical applications (avionics, automotive, medical, …)

Ariane disaster, 1996

$500 million software failure

FDIV error, 1994

$500 million


```
<msblast.exe> (the primary executable of the exploit)
I just want to say LOVE YOU SAN!!
billy gates why do you make this possible ? Stop
making money and fix your software!!
windowsupdate.com
start %s
tftp -i %s GET %s
%d.%d.%d.%d
```

Estimated worst-case worm cost:

> $50 billion

Specification, Verification, and Control

Specification

A mathematical statement of the design objective (desired properties of the system)

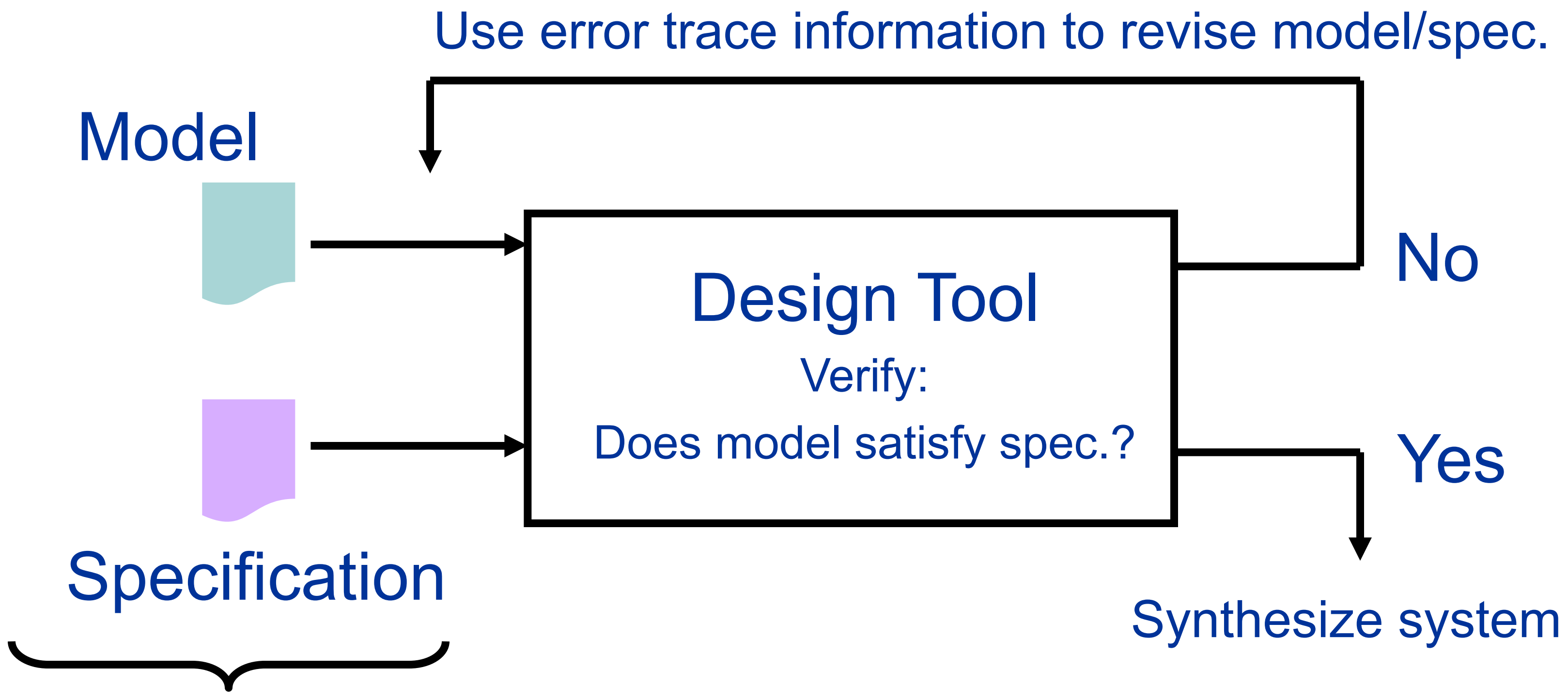
Verification

Does the designed system achieve its objective in the operating environment?

Controller Synthesis

Given an incomplete design, synthesize a strategy to complete the system so that it achieves its objective in the operating environment

Model-Based Design: Verification & Synthesis



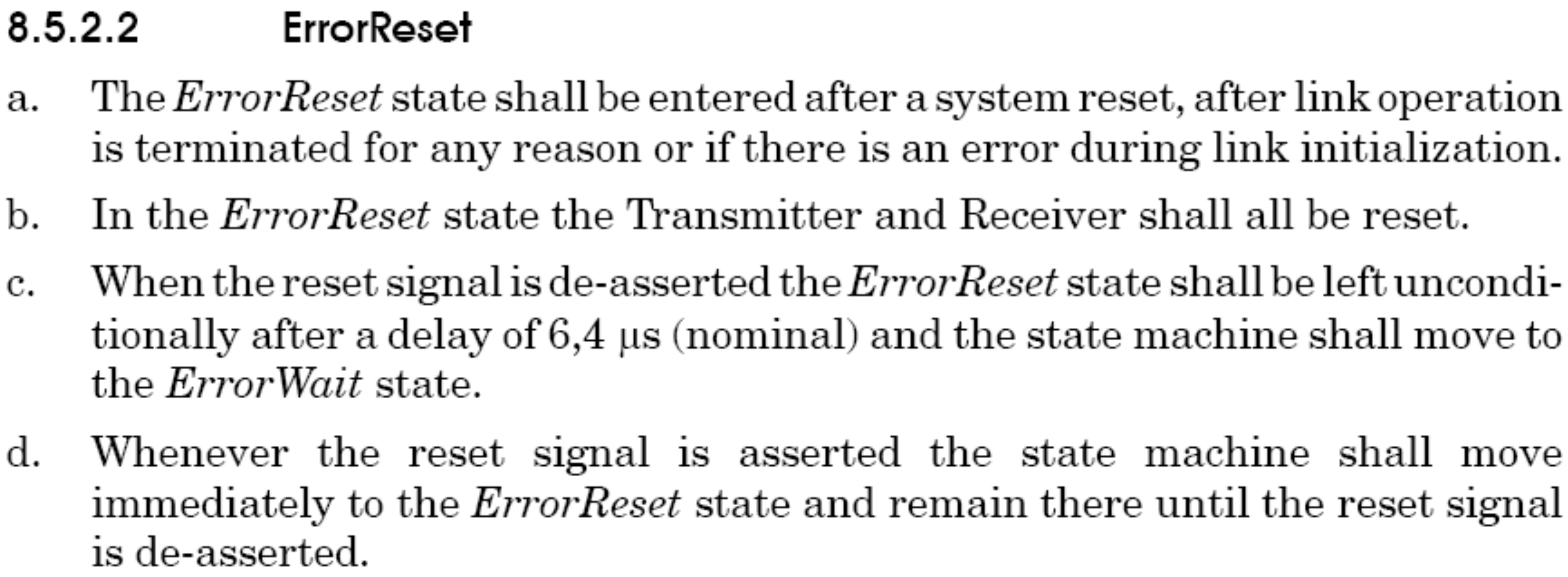
Use error trace information to revise model/spec.
Model
Design Tool
Verify:
Does model satisfy spec.?
No
Yes
Specification
Synthesize system

Need a mathematical way to write models and specifications so that an algorithm can process it

Temporal Logic

- A mathematical way to express properties of a system over time
  - E.g., Behavior of an FSM or Hybrid System
- Many flavors of temporal logic
  - Propositional temporal logic (we will study this)
  - Real-time temporal logic
- Amir Pnueli won ACM Turing Award, in part, for the idea of using temporal logic for specification

Example: Specification of the *SpaceWire* Protocol (European Space Agency standard)



8.5.2.2 ErrorReset

a. The *ErrorReset* state shall be entered after a system reset, after link operation is terminated for any reason or if there is an error during link initialization.

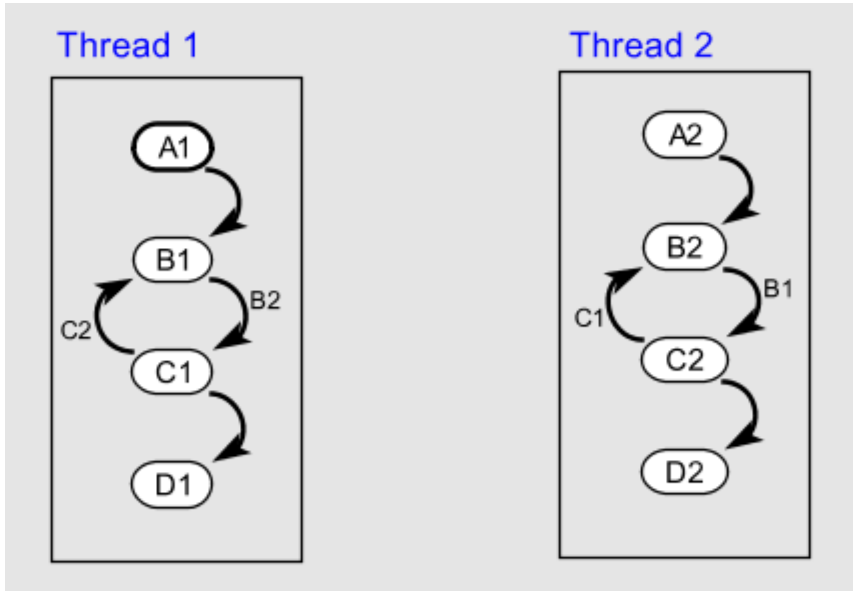
b. In the *ErrorReset* state the Transmitter and Receiver shall all be reset.

c. When the reset signal is de-asserted the *ErrorReset* state shall be left unconditionally after a delay of 6,4 μs (nominal) and the state machine shall move to the *ErrorWait* state.

d. Whenever the reset signal is asserted the state machine shall move immediately to the *ErrorReset* state and remain there until the reset signal is de-asserted.

Example from the Threads Lecture

States or transitions represent atomic instructions



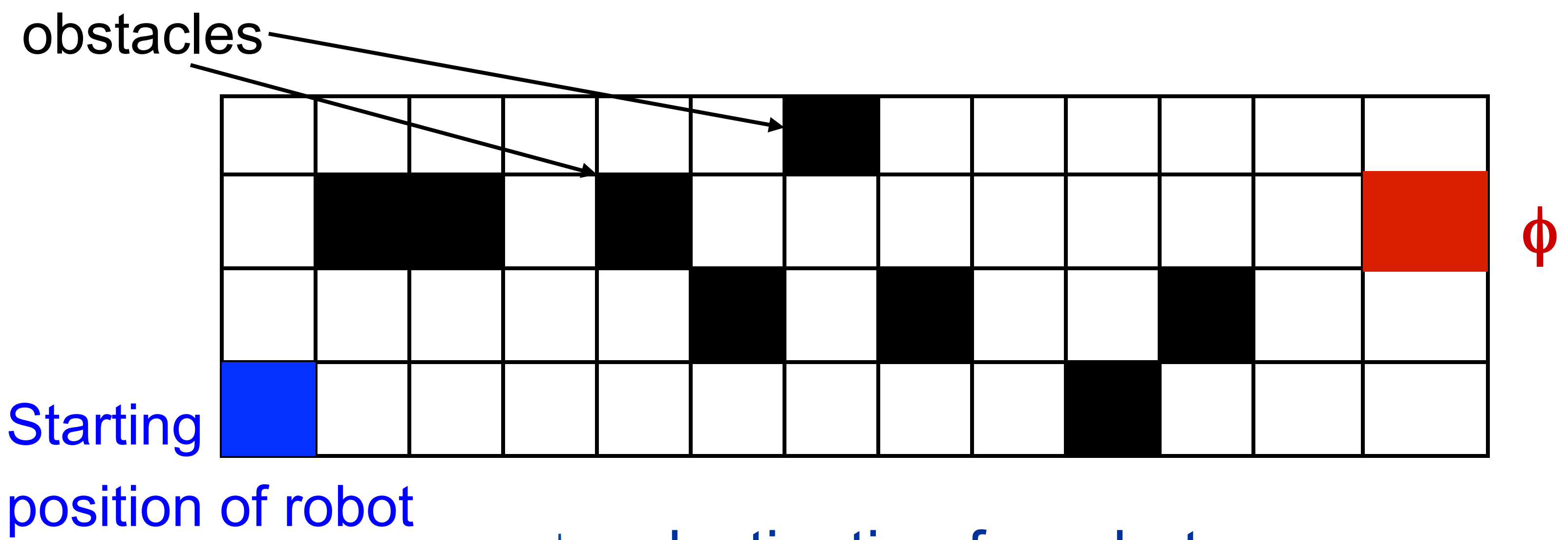
Interleaving semantics:

- Choose one machine, arbitrarily.
- Advance to a next state if guards are satisfied.
- Repeat.

We hand-computed the set of reachable states.

The 2-threaded program should never be in state (C1,C2)
Thread i must eventually reach Di

A Robot delivery service, with moving obstacles



ϕ = destination for robot

Specification:

The robot eventually reaches ϕ

Suppose there are n destinations $\phi_1, \phi_2, \ldots, \phi_n$

The new specification could be that

The robot visits $\phi_1, \phi_2, \ldots, \phi_n$ in that order

Propositional Logic

Atomic formulas: Statements about an input, output, or state of a state machine. Examples:

formula	meaning
x	x is *present*
$x = 1$	x is *present* and has value 1
s	machine is in state s

These are predicates (true or false statements) about a state machine with input or output x and state s.

Propositional Logic

Propositional logic formulas: More elaborate statements about an input, output, or state of a state machine. Examples:

formula	meaning
$p_1 \wedge p_2$	p_1 and p_2 are both true
$p_1 \vee p_2$	either p_1 or p_2 is true
$p_1 \implies p_2$	if p_1 is true, then so is p_2
$\neg p_1$	true if p_1 is false

Here, p_1 and p_2 are either atomic formulas or propositional logic formulas.

Execution Trace of a State Machine

An **execution trace** is a sequence of the form

$$q_0,\ q_1,\ q_2,\ q_3,\ \ldots,$$

where $q_j = (x_j, s_j, y_j)$ where s_j is the state at step j, x_j is the input valuation at step j, and y_j is the output valuation at step j. Can also write as

$$s_0 \xrightarrow{x_0/y_0} s_1 \xrightarrow{x_1/y_1} s_2 \xrightarrow{x_2/y_2} \cdots$$

Propositional Logic on Traces

A propositional logic formula p **holds** for a trace

$$q_0,\, q_1,\, q_2,\, q_3,\, \ldots,$$

if and only if it holds for q_0.

This may seem odd, but we will provide temporal logic operators to reason about the trace.

Linear Temporal Logic (LTL)

LTL formulas: Statements about an execution trace

$$q_0,\ q_1,\ q_2,\ q_3,\ \ldots,$$

formula	meaning
p	p holds in q_0
$\mathbf{G}\phi$	ϕ holds for every suffix of the trace
$\mathbf{F}\phi$	ϕ holds for some suffix of the trace
$\mathbf{X}\phi$	ϕ holds for the trace $q_1, q_2, \cdots$
$\phi_1 \mathbf{U} \phi_2$	ϕ_1 holds for all suffixes of the trace until a suffix for which ϕ_2 holds.

Here, p is propositional logic formula and ϕ is either a propositional logic or an LTL formula.

Linear Temporal Logic (LTL)

LTL formulas: Statements about an execution trace

$$q_0,\ q_1,\ q_2,\ q_3,\ \ldots,$$

formula	mnemonic
p	proposition
$\mathbf{G}\phi$	globally
$\mathbf{F}\phi$	finally, future, eventually
$\mathbf{X}\phi$	next state
$\phi_1\mathbf{U}\phi_2$	until

Here, p is propositional logic formula and ϕ is either a propositional logic or an LTL formula.

First LTL Operator: G (Globally)

The LTL formula $\mathbf{G}p$ **holds** for a trace

$$q_0,\ q_1,\ q_2,\ q_3,\ \ldots,$$

if and only if it holds for every suffix of the trace:

$$\begin{array}{r}
q_0,\ q_1,\ q_2,\ q_3,\ \ldots \\
q_1,\ q_2,\ q_3,\ \ldots \\
q_2,\ q_3,\ \ldots \\
q_3,\ \ldots
\end{array}$$

If p is a propositional logic formula, this means it holds for each q_i.

Second LTL Operator: F (Eventually, Finally)

The LTL formula $\mathbf{F}p$ **holds** for a trace

$$q_0,\ q_1,\ q_2,\ q_3,\ \ldots,$$

if and only if it holds for some suffix of the trace:

$$\begin{array}{r}
q_0,\ q_1,\ q_2,\ q_3,\ \ldots \\
q_1,\ q_2,\ q_3,\ \ldots \\
q_2,\ q_3,\ \ldots \\
q_3,\ \ldots
\end{array}$$

If p is a propositional logic formula, this means it holds for some q_i.

Propositional Linear Temporal Logic

LTL operators can apply to LTL formulas as well as to propositional logic formulas.

E.g. Every input x is eventually followed by an output y

$$\underbrace{\mathbf{G}}_{\text{Globally (at any point in time)}}(\underbrace{x}_{\text{If } x \text{ occurs}} \implies \underbrace{\mathbf{F}y}_{\text{It is eventually followed by } y})$$

Globally

(at any point in time)

If x occurs

It is *eventually* followed by y

Every input x is eventually followed by an output y

The LTL formula $\mathbf{G}(x \implies \mathbf{F}y)$ holds for a trace

$$q_0, q_1, q_2, q_3, \ldots,$$

if and only if it holds for any suffix of the trace where x holds, there is a suffix of that suffix where y holds:

$$\begin{array}{rrrrr}
q_0, & q_1, & q_2, & q_3, & \ldots \\
 & q_1, & q_2, & q_3, & \ldots \\
 & & q_2, & q_3, & \ldots \\
 & & & q_3, & \ldots
\end{array}$$

y holds

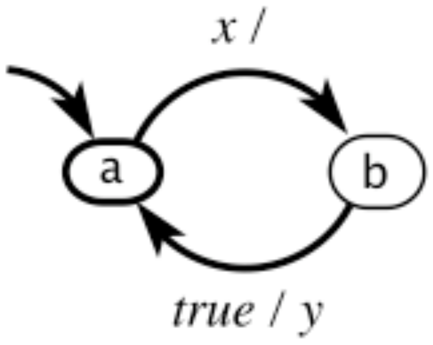
x holds

Propositional Temporal Logic

Does the following hold?

$$\mathbf{G}(x \implies \mathbf{F}y)$$

input: x: pure
output: y: pure



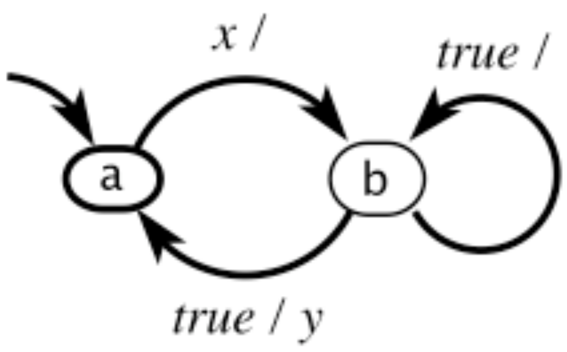
yes

Propositional Temporal Logic

Does the following hold?

$$\mathbf{G}(x \implies \mathbf{F}y)$$

input: x: pure
output: y: pure



no

Third LTL Operator: X (Next)

The LTL formula $\mathbf{X}p$ **holds** for a trace

$$q_0,\ q_1,\ q_2,\ q_3,\ \ldots,$$

if and only if it holds for the suffix $q_1,\ q_2,\ q_3,\ \ldots$

$$\begin{array}{r} q_0,\ q_1,\ q_2,\ q_3,\ \ldots \\ q_1,\ q_2,\ q_3,\ \ldots \\ q_2,\ q_3,\ \ldots \\ q_3,\ \ldots \end{array}$$

Fourth LTL Operator: U (Until)

The LTL formula $p_1 \mathbf{U} p_2$ **holds** for a trace

$$q_0, q_1, q_2, q_3, \ldots,$$

if and only if p_2 holds for some suffix of the trace, and p_1 holds for all previous suffixes:

$$q_0, q_1, q_2, q_3, \ldots$$
$$q_1, q_2, q_3, \ldots$$
$$q_2, q_3, \ldots$$
$$q_3, \ldots$$

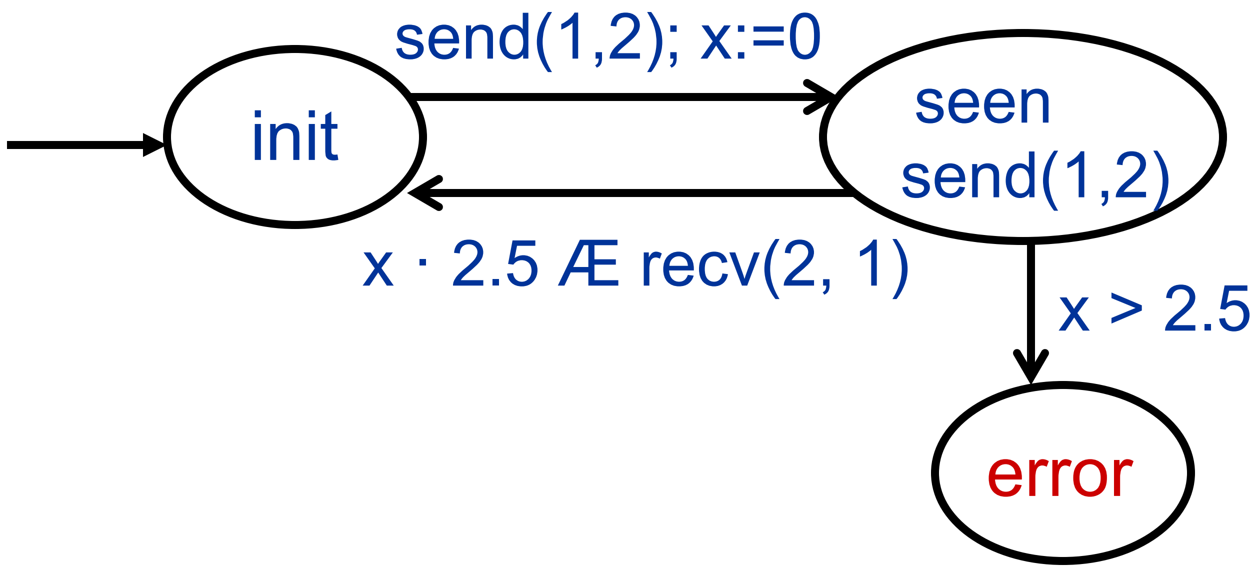
🔵 p_1 holds

🩷 p_2 holds (and maybe p_1 also)

Real-Time Temporal Logic

Every send(1, 2) is followed by a recv(2, 1) within 2.5 ms

G { send(1,2)) $\mathbf{F}_{\cdot 2.5}$ recv(2, 1) }



Alternate Notation

Sometimes you' ll see alternative notation in the literature:

G	¤
F	◊
X	○

Examples: What do they mean?

Remember:

Gp	p holds in all states
Fp	p holds eventually
Xp	p holds in the next state

- $\mathbf{G}\,\mathbf{F}\,p$

  p holds infinitely often

- $\mathbf{F}\,\mathbf{G}\,p$

  Eventually, p holds henceforth

- $\mathbf{G}(\,p \Rightarrow \mathbf{F}\,q\,)$

  Every p is eventually followed by a q

- $\mathbf{F}(\,p \Rightarrow (\mathbf{X}\,\mathbf{X}\,q)\,)$

  Every p is followed by a q two reactions later

Examples: Write in Temporal Logic

1. "Whenever the iRobot is at the ramp-edge (cliff), eventually it moves 5 cm away from the cliff."
   - p – iRobot is at the cliff
   - q – iRobot is 5 cm away from the cliff

2. "Whenever the distance between cars is less than 2m, cruise control is deactivated"
   - p – distance between cars is less than 2 m
   - q – cruise control is active

Temporal Operators & Relationships

G, F, X, U: All express properties along system traces

- Can you express G p purely in terms of F, p, and Boolean operators ?

$$\mathbf{G}\phi = \neg\mathbf{F}\neg\phi$$

- How about F in terms of U?

$$\mathbf{F}\phi = \mathit{true}\ \mathbf{U}\ \phi$$

- What about X in terms of G, F, or U?

Cannot be done

Some Points to Ponder

- A mathematical specification only includes properties that the system must or must not have
- It requires human judgment to decide whether that specification constitutes “correctness”
- Getting the specification right is often as hard as getting the design right!

Exercises

Write the SpaceWire specs. in Temporal Logic

Also write the specification for the Robot and Thread examples in Temporal Logic