Lecture 22 — **Case Study:** Microwave Oven

David J. Pearce

School of Engineering and Computer Science
Victoria University of Wellington

Microwave Ovens

“When used according to manufacturers’ instructions, microwave ovens are safe and convenient for heating and cooking a variety of foods. However, several precautions need to be taken, specifically with regards to potential exposure to microwaves, thermal burns and food handling.”

–World Health Organisation

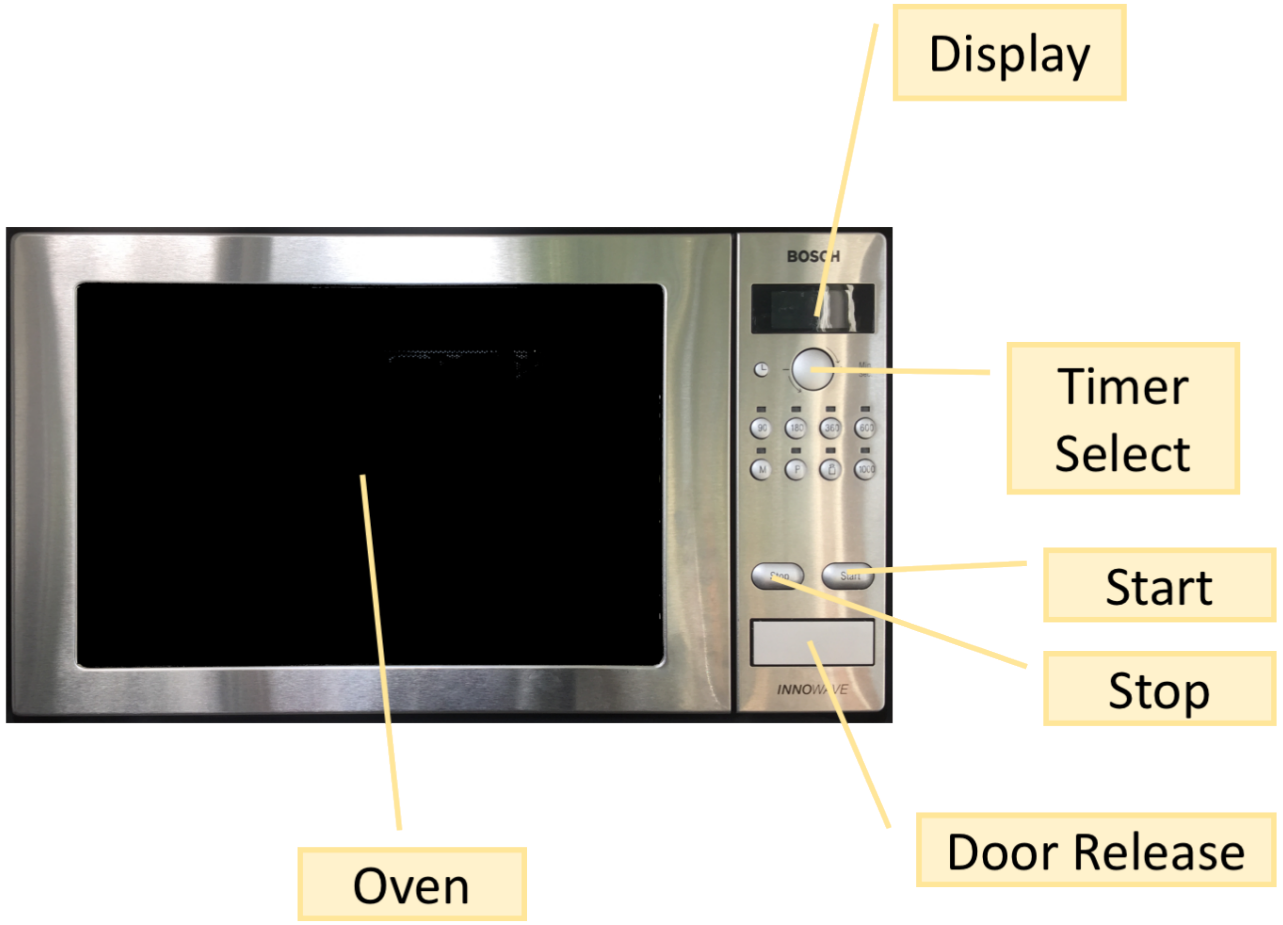
Microwave Ovens (are Safety-Critical Systems)

“On July 29, 1977, H.F., a 51-year-old teacher, was attempting to remove a casserole dish from her **new 600-watt microwave oven**. *The oven signaled the end of the heating cycle, but the light and the cooking blower were on. During retrieval of the dish, she inserted two thirds of her bare forearms into the oven, for a total time of about five seconds.* **The oven was still operating**. *She felt "hot pulsating sensation" and burning in fingers and fingernails and a sensation of "needles" over the exposed areas. Jabbing pain, swelling, and red-orange discoloration of dorsal sides of both hands and forearms appeared shortly afterwards. The next day she sought medical help.”*

–Wikipedia

See: `https://en.wikipedia.org/wiki/Microwave_burn`

Modelling a Microwave Oven



- **Also:** *heating element*, *door sensor*

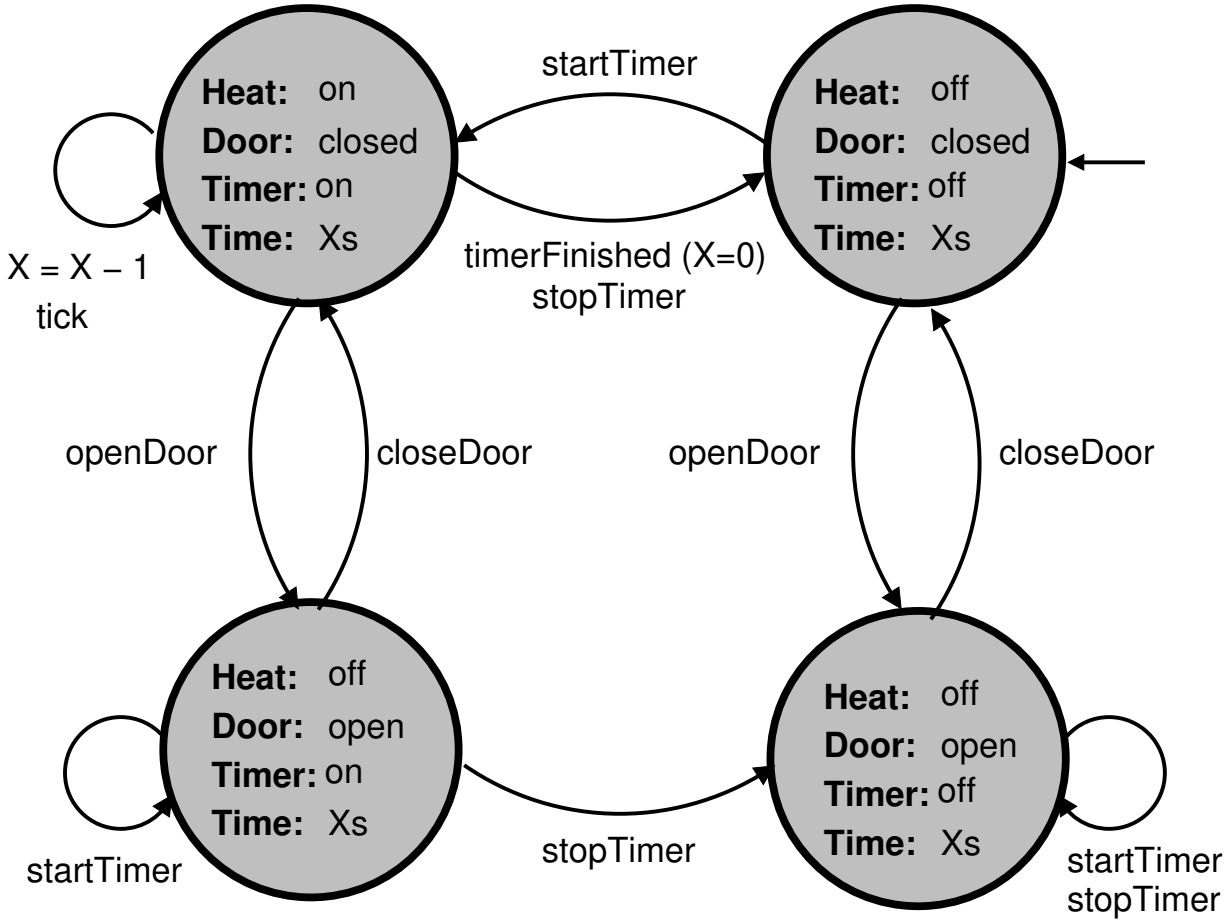
(image of Dave's microwave; public domain)

Safety Requirement

SR1

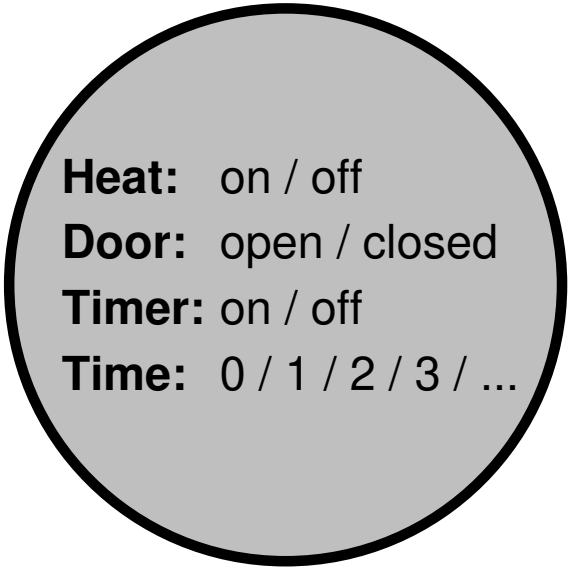
“When the door is **open***, the heating element must be* **off***”*

Microwave State Diagram



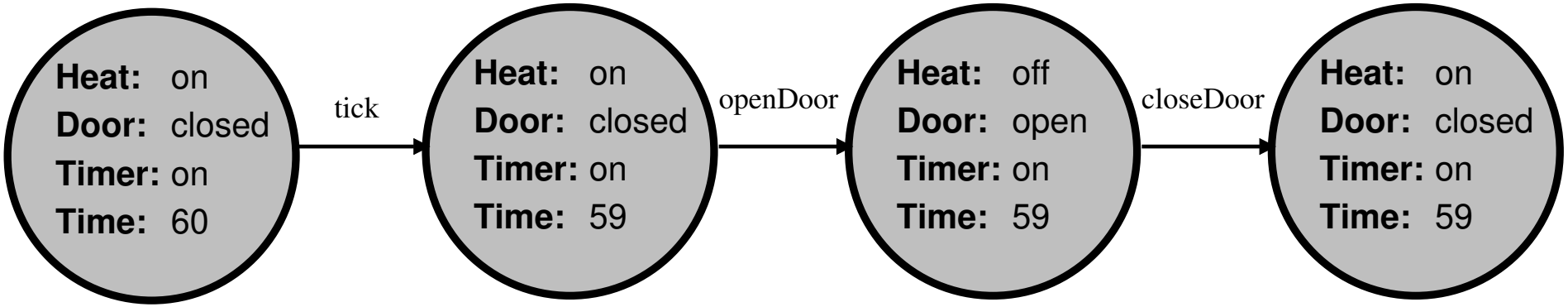
- **Objective:** check this with Whiley!

Microwave State (in Whiley)




```
type State is {
  bool heat,
  bool door,
  bool timer,
  int time
} where time >= 0
```

Microwave Execution Traces



Abstract Transitions

```
// Initial state of microwave
final State INITIAL_STATE = {
  heat: false,
  door: false,
  timer: false,
  time: 0
}

// Transition from one state to another
property transition(State from, State to)
where openDoor(from,to) || closeDoor(from,to)
      || startTimer(from,to) || stopTimer(from,to)
      || timerFinished(from,to) || setTimer(from,to)
      || tick(from,to)
```

Transitions: Open / Close Door

```
property openDoor(State from, State to)
// Door can only be opened if already closed (physics)
where !from.door && to.door
// Door opening doesn't affect timer
where (from.timer == to.timer) && (from.time == to.time)
// Heat must be off when door opened (safety)
where !to.heat
```

```
property closeDoor(State from, State to)
// Door can only be closed if already open
where from.door && !to.door
// Door closing doesn't affect timer
where (from.timer == to.timer) && (from.time == to.time)
// Door closing doesn't affect heat either
where from.heat == to.heat
```

Doesn't handle **both** *transitions in state diagram! Why Not?*

Transitions: Start / Stop Timer

```
property startTimer(State from, State to)
// Pressing start doesn't affect door or time
where (from.door == to.door) && (from.time == to.time)
// Heat will come on if and only if door closed
where !from.door <==> to.heat
// Timer will begin if and only if door closed
where !from.door <==> to.timer
```

```
property stopTimer(State from, State to)
// Pressing stop doesn't affect door or time
where (from.door == to.door) && (from.time == to.time)
// Heat is off after stop pressed
where !to.heat && !to.timer
```

Transitions: Timer Up

```
property timerFinished(State from, State to)
// Timer up only occurs when timer on and zero
where (from.time == 0) && from.timer
// Timer up doesn't affect door or time value
where (from.door == to.door) && (from.time == to.time)
// Heat is off after timer up
where !to.heat
```

Something is definitely **wrong** *with this!*

Transitions: Timer Stuff

```
property setTimer(State from, State to)
// Setting timer doesn't affect door or time value
where (from.door == to.door) && (from.heat == to.heat)
// Setting timer doesn't affect timer status, but does set timer value
where (from.timer == to.timer) && (to.time == 60)

property tick(State from, State to)
// Ticking doesn't affect door or time value
where (from.door == to.door) && (from.heat == to.heat)
// Ticking doesn't affect timer status
where (from.timer == to.timer)
// Ticking decreases time
where (from.time > 0) && (to.time == from.time - 1)
```

Models **user input** *using non-deterministic choice! When does time tick? Does time go up or down?*

Checking the **Safety Property!**

SR1

"When the door is **open***, the heating element must be* **off***"*

```
property isSafe(State s)
// When door open, heat must be off
where s.door ==> !s.heat
```

How do we **check** *this property in Whiley?*

Checking the **Safety Property!**

```
method lemma(State[] states) -> (int r)
// Always at least one state
requires |states| > 0
// Initial states always fixed
requires states[0] == INITIAL_STATE
// Remaining states connected by transitions
requires all { k in 1 .. |states|
               | transition(states[k-1],states[k]) }
// When door open, heat must be off
ensures all { k in 0..|states| | isSafe(states[k]) }:
    // Done
    return 1
```

Good to **break** *model and check it catches this!*