Engineering Technology (ENGR 101)

FSM: Traffic lights

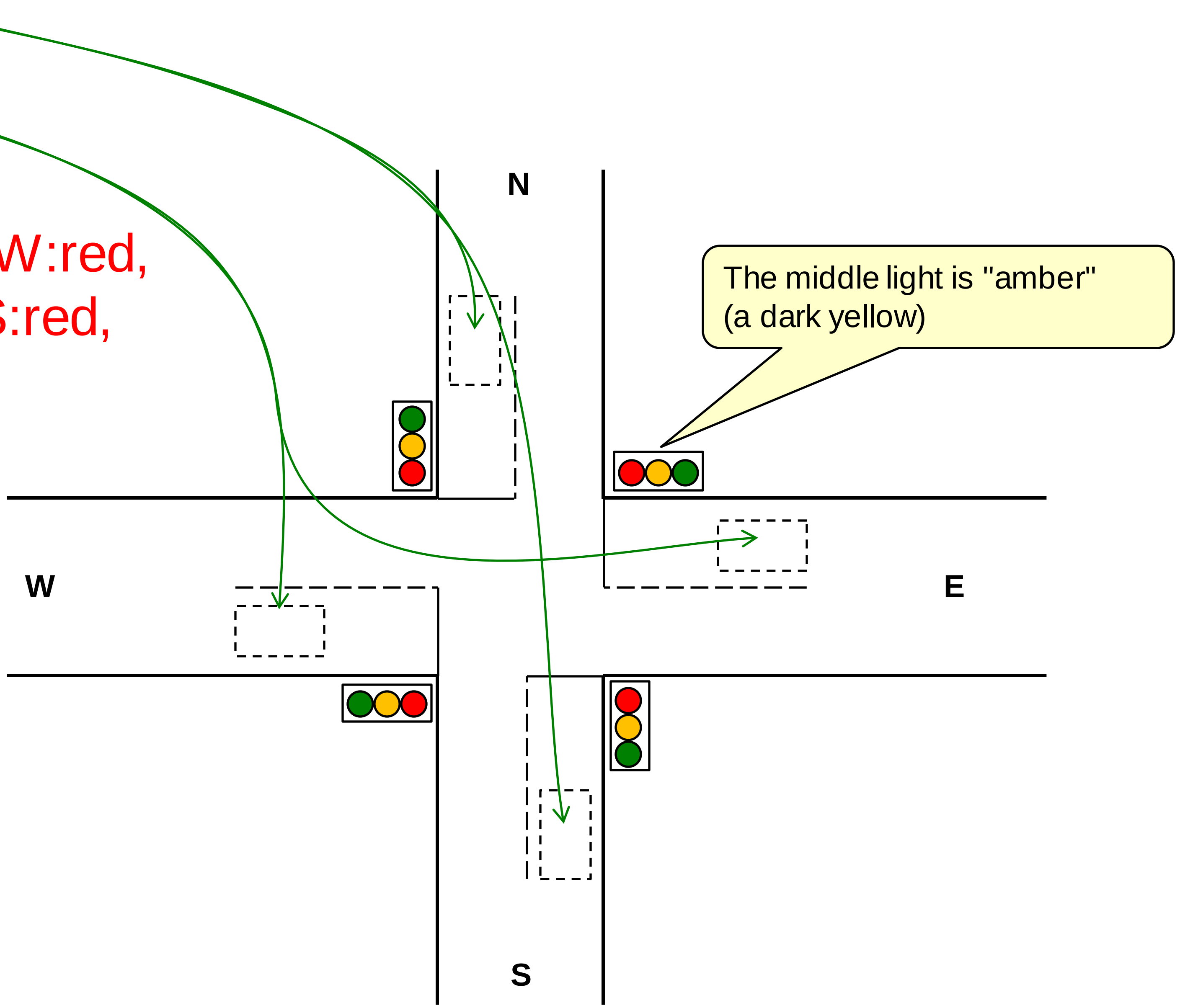
Example: Traffic light controller

- Sensors: carEW, carNS,
  timer

- Actions: EW:grn, EW:amb, EW:red,
  NS:grn, NS:amb, NS:red,
  setTimer(n)

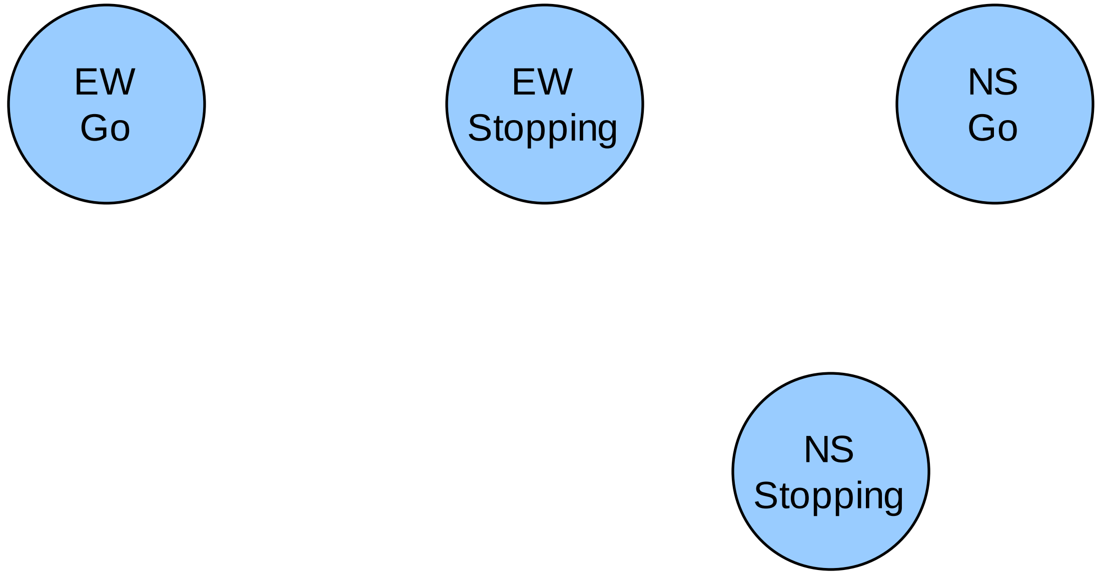
- States?
  - what are the different conditions where the Arduino should act differently?
  - eg:

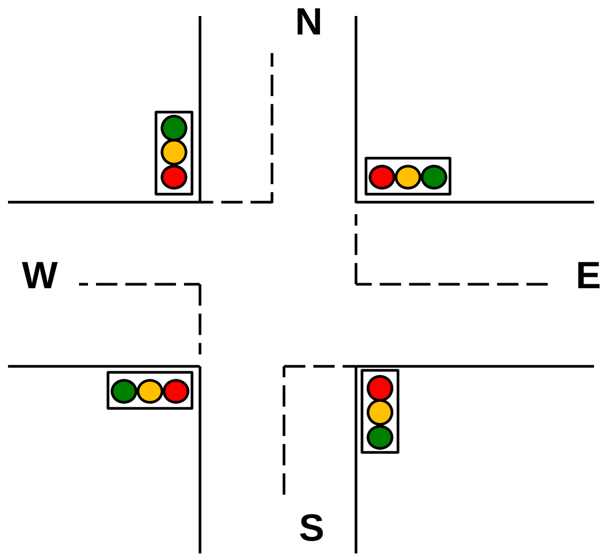
    should it always change the lights when a car drives up?



States for Arduino sketch for traffic light

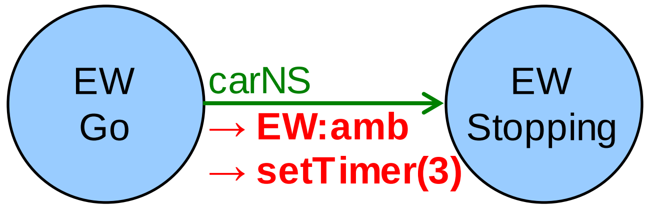
• Cars going East-West (and cars stopped North-South)	EW-Go
• Cars going North-South (and cars stopped East-West)	NS-Go
• Cars Stopping East-West (amber) (and cars stopped N-S)	EW-Stopping
• Cars Stopping North-South (amber) (cars stopped E-W)	NS-Stopping

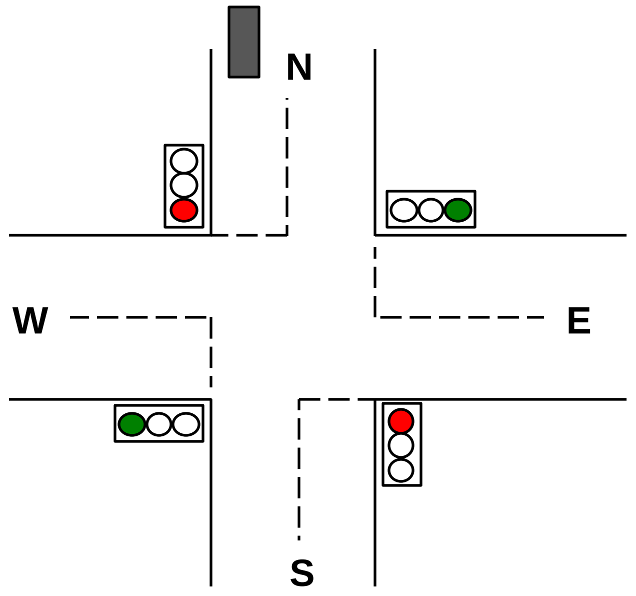




States for Arduino sketch for traffic light

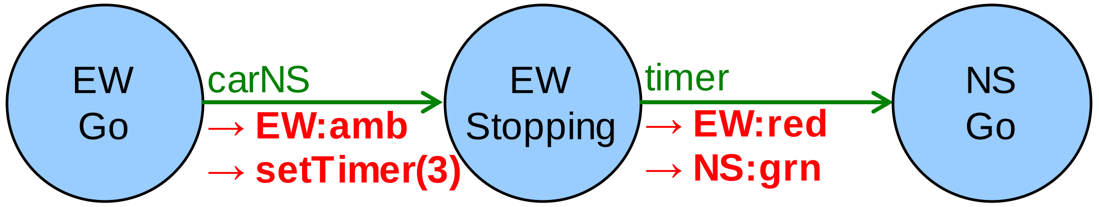
- Sensors: carEW, carNS, timer
- Actions: EW:grn, EW:amb, EW:red, NS:grn, NS:amb, NS:red, setTimer(n)

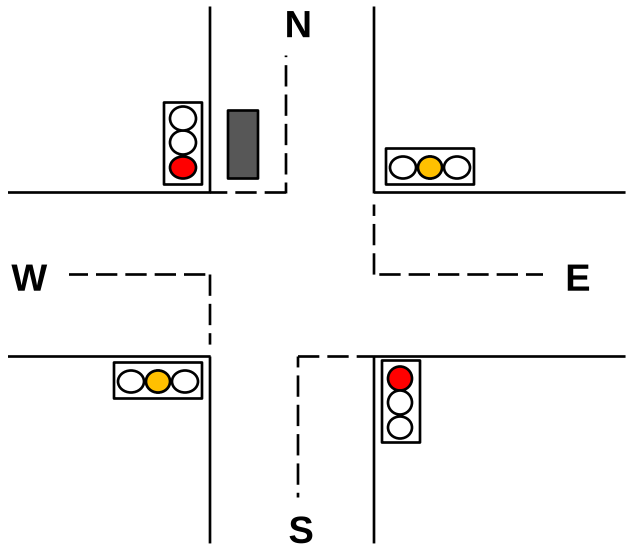




States for Arduino sketch for traffic light

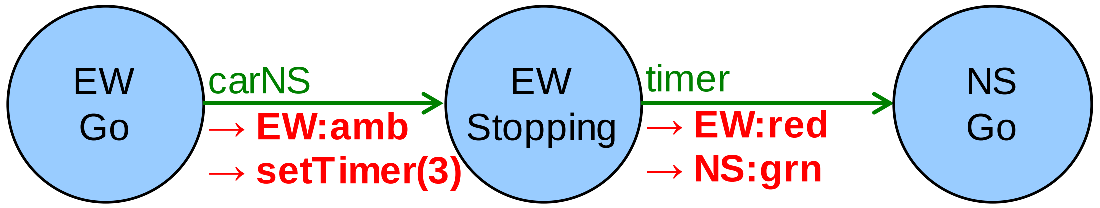
- Sensors: carEW, carNS, timer
- Actions: EW:grn, EW:amb, EW:red, NS:grn, NS:amb, NS:red, setTimer(n)

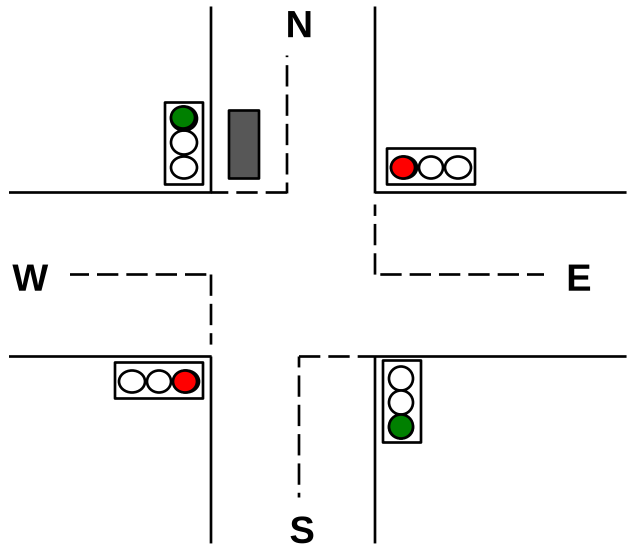




States for Arduino sketch for traffic light

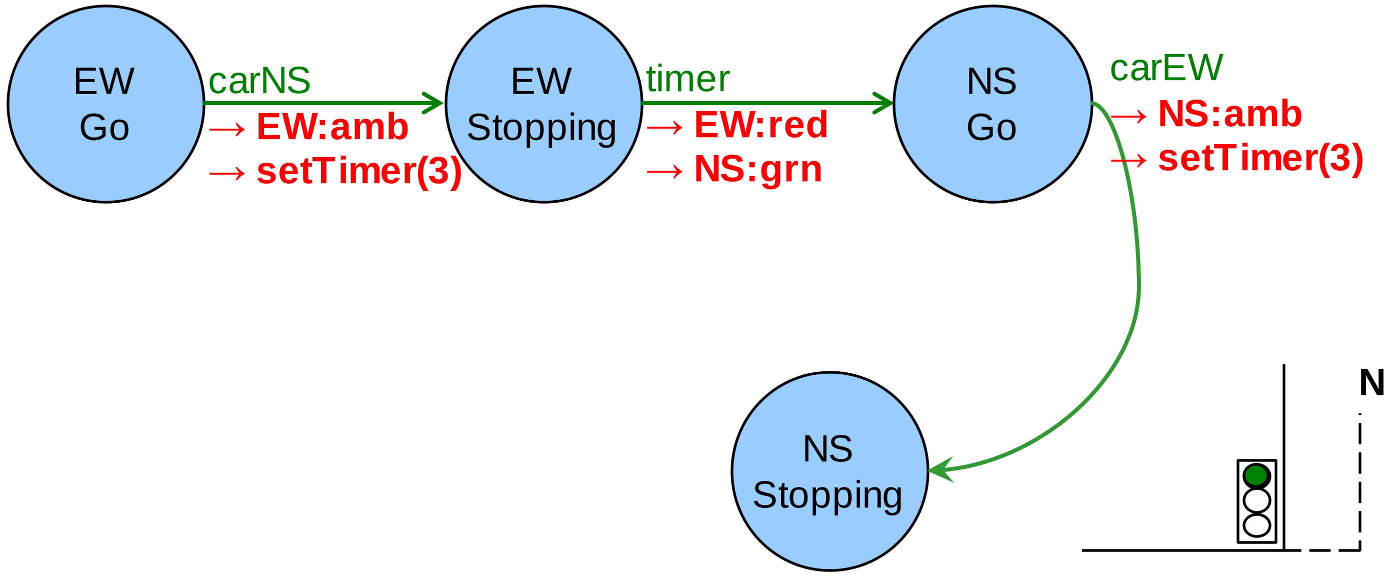
- Sensors: carEW, carNS, timer
- Actions: EW:grn, EW:amb, EW:red, NS:grn, NS:amb, NS:red, setTimer(n)

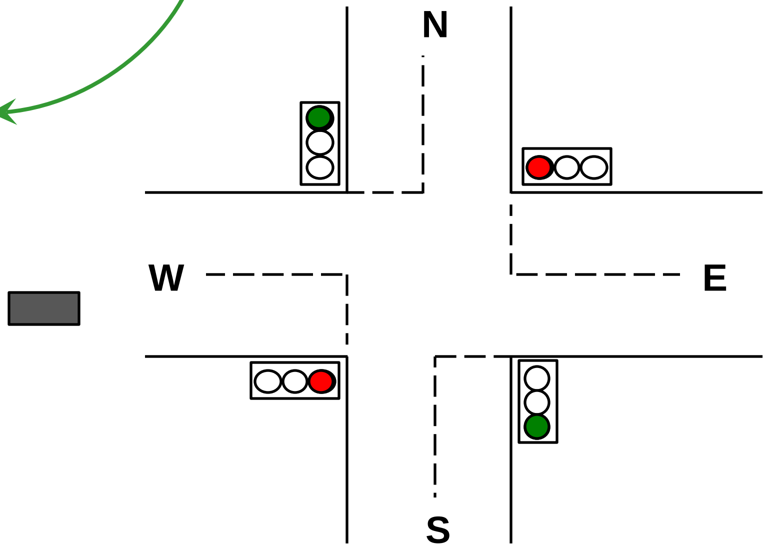




States for Arduino sketch for traffic light

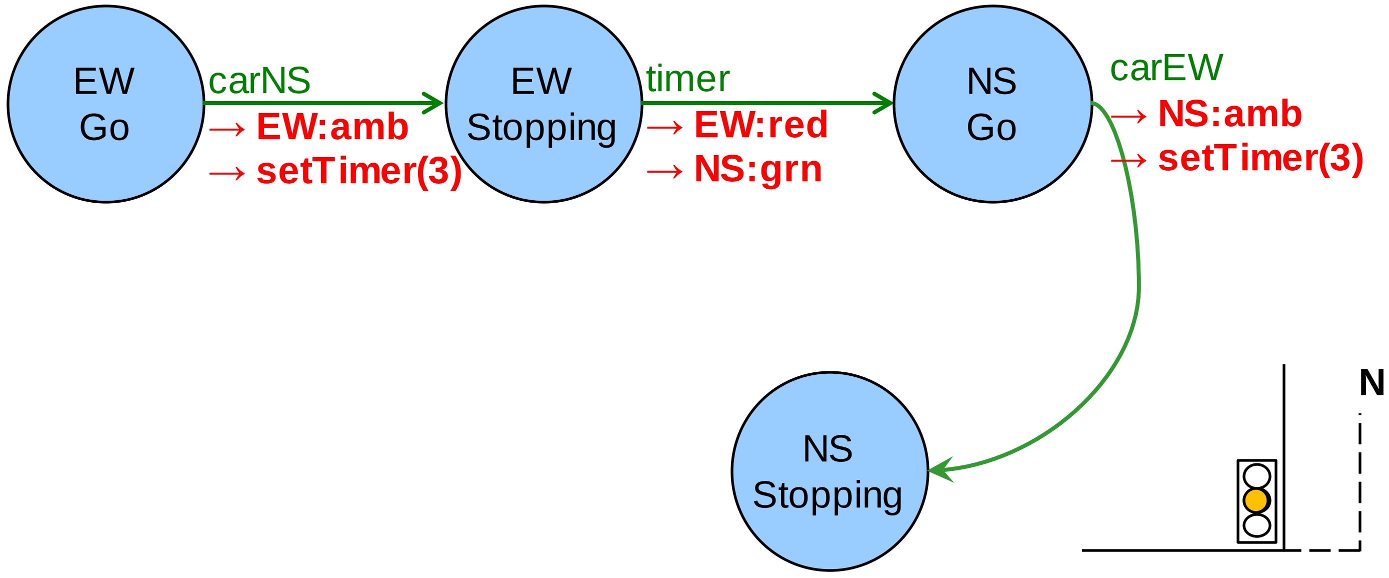
- Sensors: carEW, carNS, timer
- Actions: EW:grn, EW:amb, EW:red, NS:grn, NS:amb, NS:red, setTimer(n)

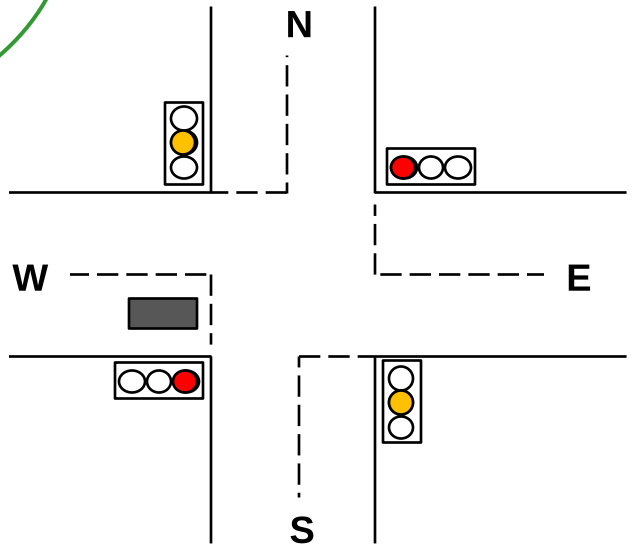




States for Arduino sketch for traffic light

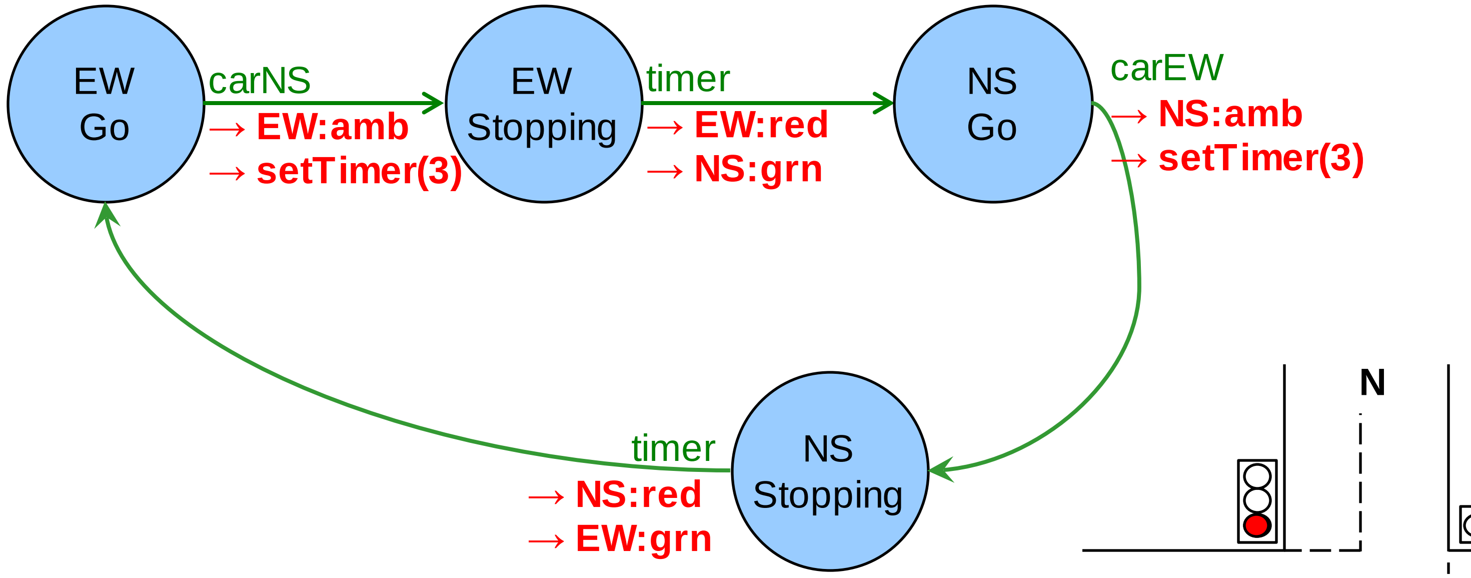
- Sensors: carEW, carNS, timer
- Actions: EW:grn, EW:amb, EW:red, NS:grn, NS:amb, NS:red, setTimer(n)

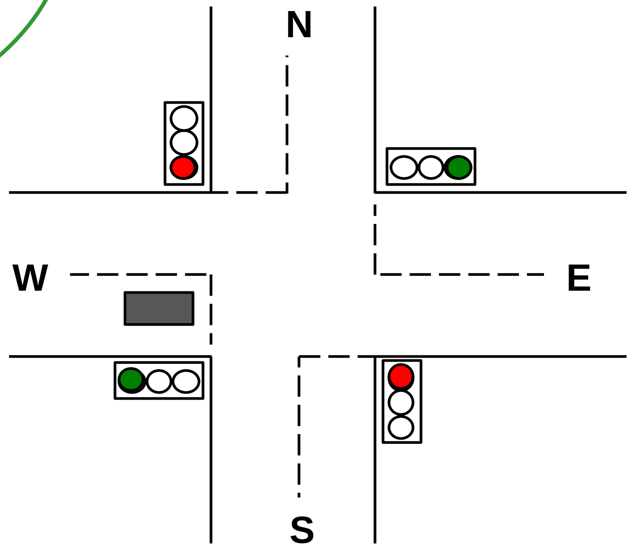




States for Arduino sketch for traffic light

- Sensors: carEW, carNS, timer
- Actions: EW:grn, EW:amb, EW:red, NS:grn, NS:amb, NS:red, setTimer(n)

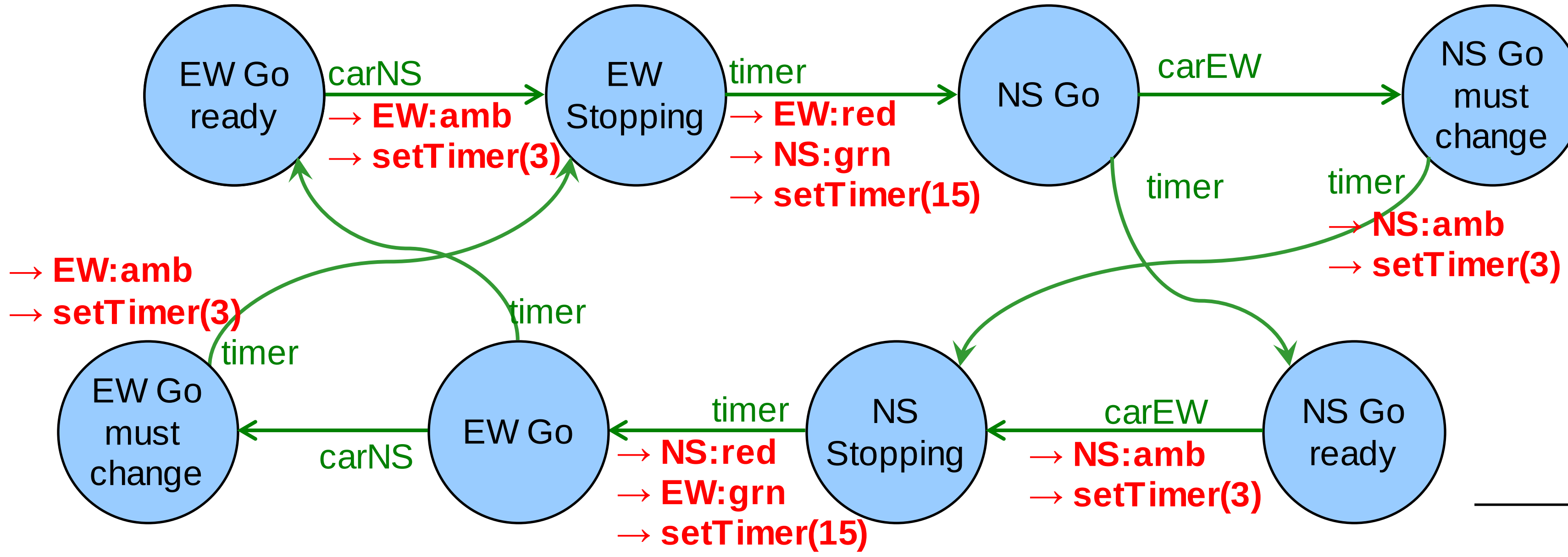


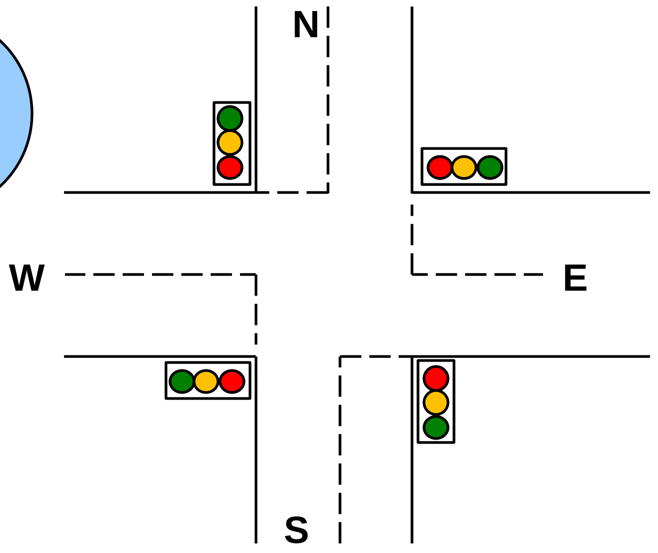


- What's the problem with this?

States for Arduino sketch for traffic light

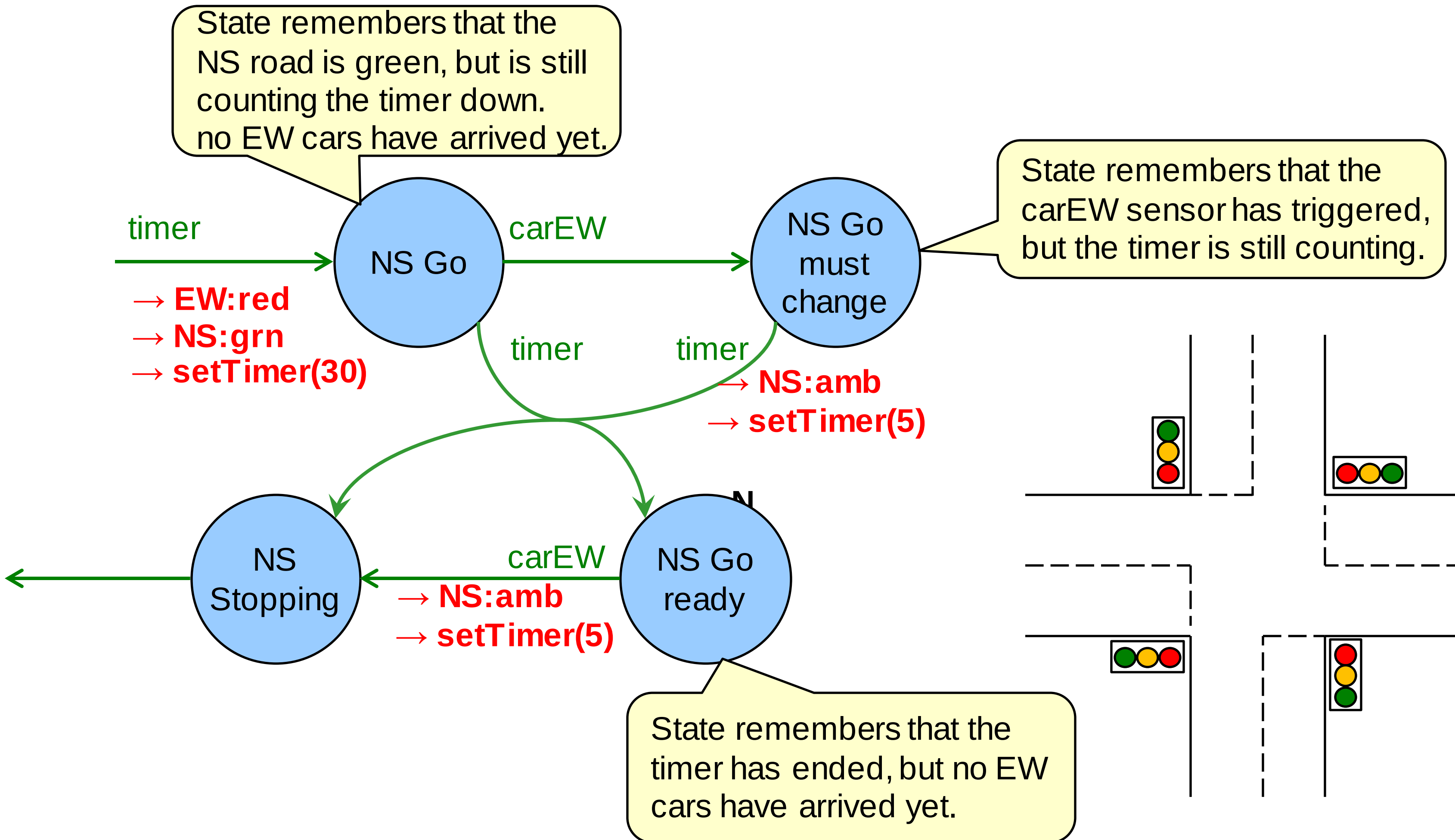
- Sensors: carEW, carNS, timer
- Actions: EW:grn, EW:amb, EW:red, NS:grn, NS:amb, NS:red, setTimer(n)

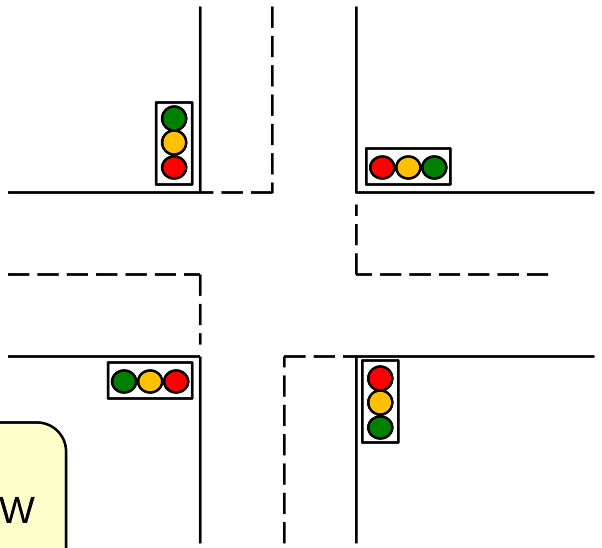




States remember sensor values.

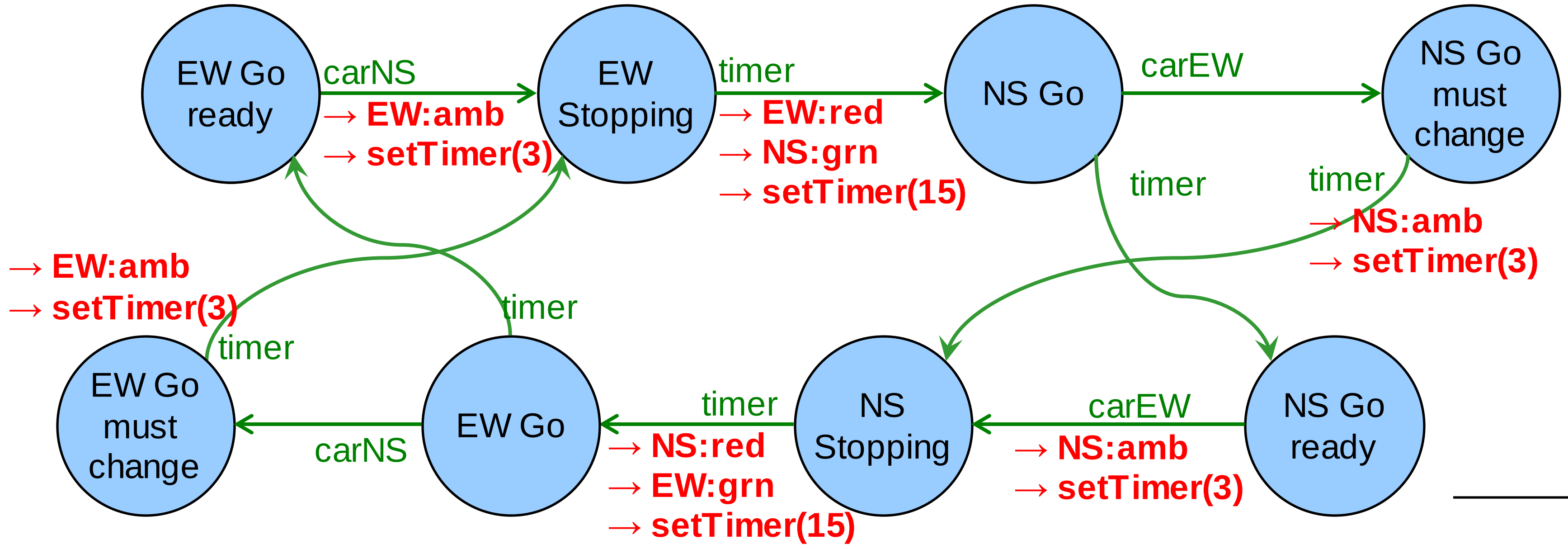
- Sensors: carEW, carNS, timer
- NS Go represents the state in which the N-S roads have the green.

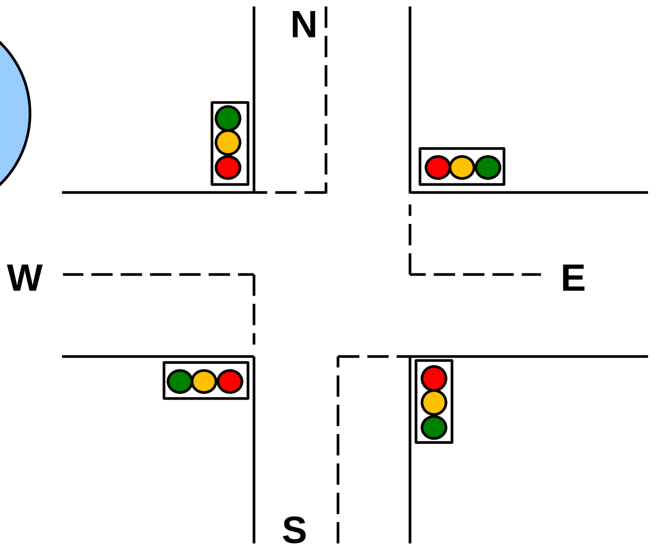

State remembers that the NS road is green, but is still counting the timer down. no EW cars have arrived yet.
timer
→ EW:red
→ NS:grn
→ setTimer(30)
NS Go
carEW
NS Go must change
State remembers that the carEW sensor has triggered, but the timer is still counting.
timer
timer
→ NS:amb
→ setTimer(5)
NS Stopping
carEW
→ NS:amb
→ setTimer(5)
NS Go ready
State remembers that the timer has ended, but no EW cars have arrived yet.



States for Arduino sketch for traffic light

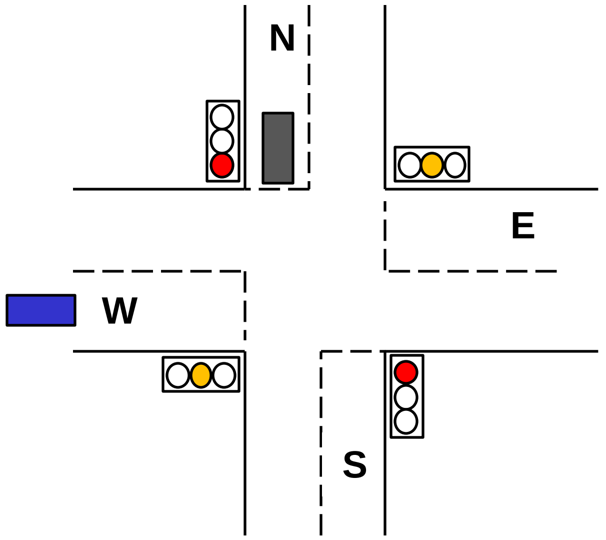
- Sensors: carEW, carNS, timer
- Actions: EW:grn, EW:amb, EW:red, NS:grn, NS:amb, NS:red, setTimer(n)

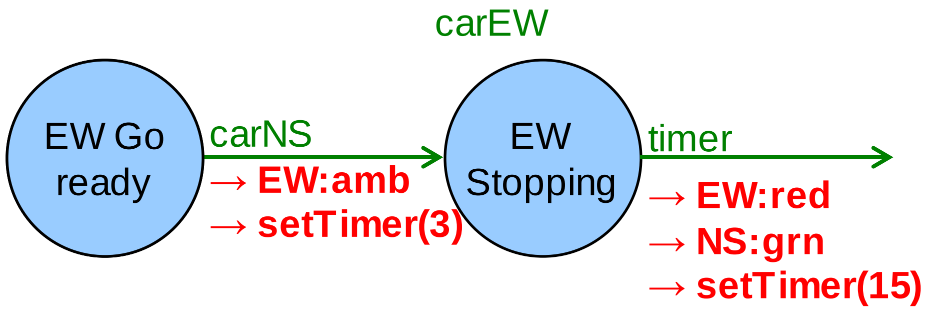




- What if an EW car arrives (and stops) during the EW-Stopping? Would it ever be noticed?

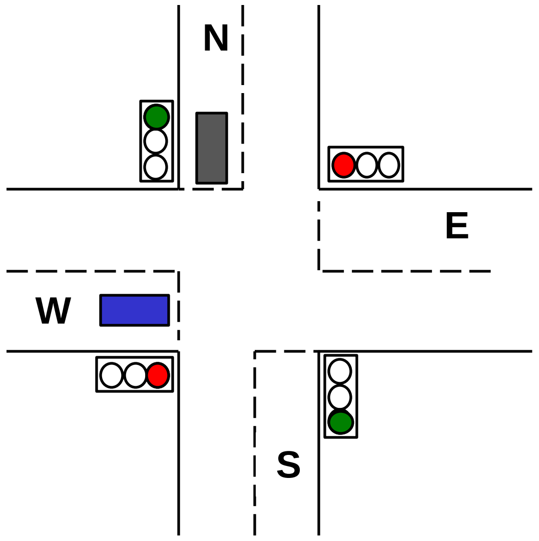
States for Arduino sketch for traffic light

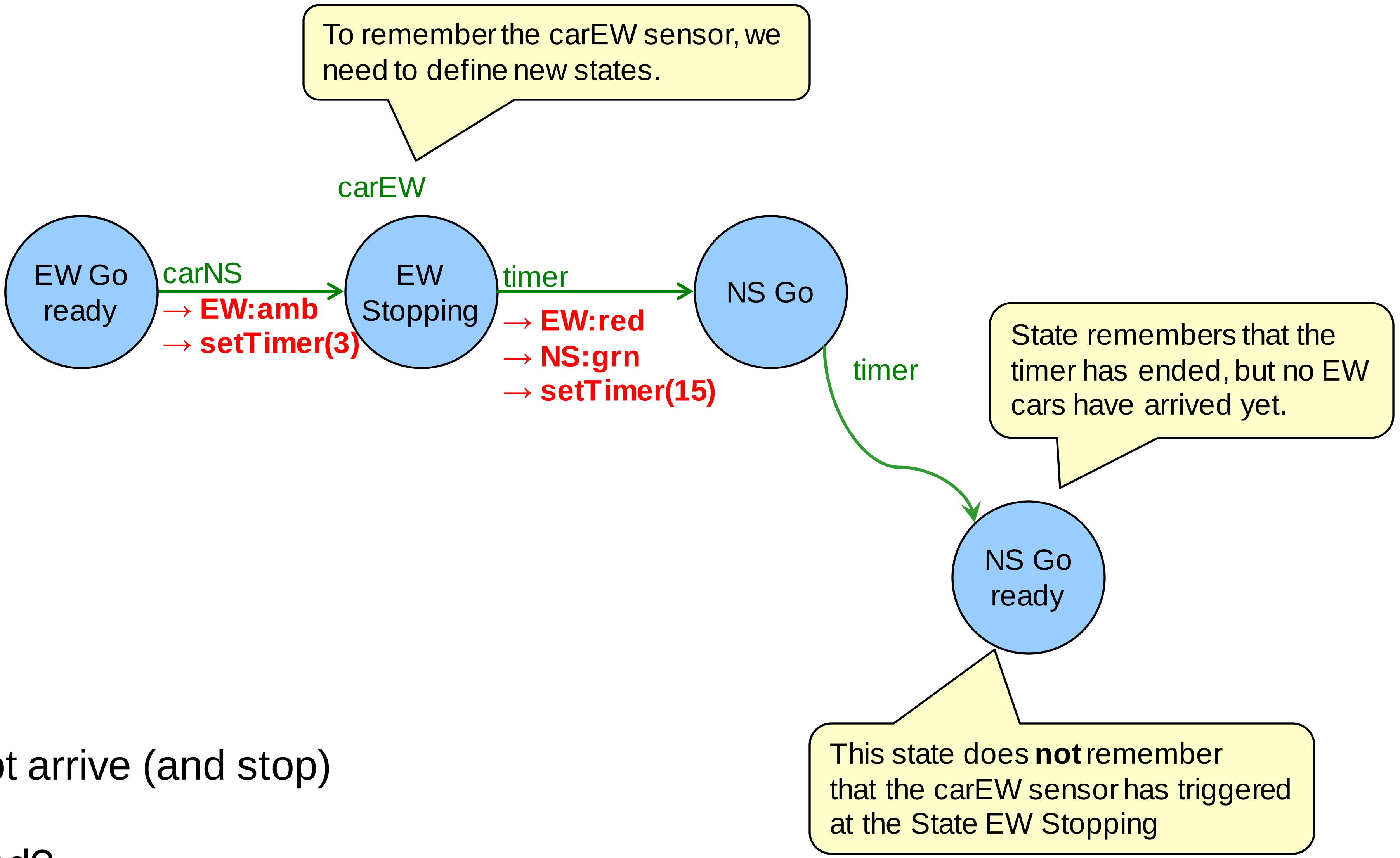




What if an EW car does not arrive (and stop) during the EW-Stopping?
Would timer ever be noticed?

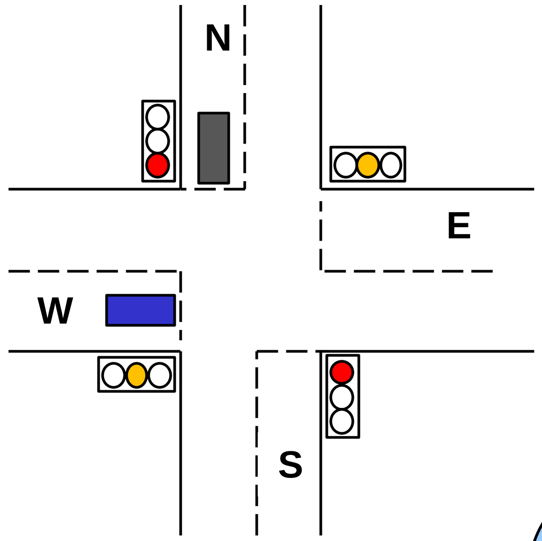
States for Arduino sketch for traffic light

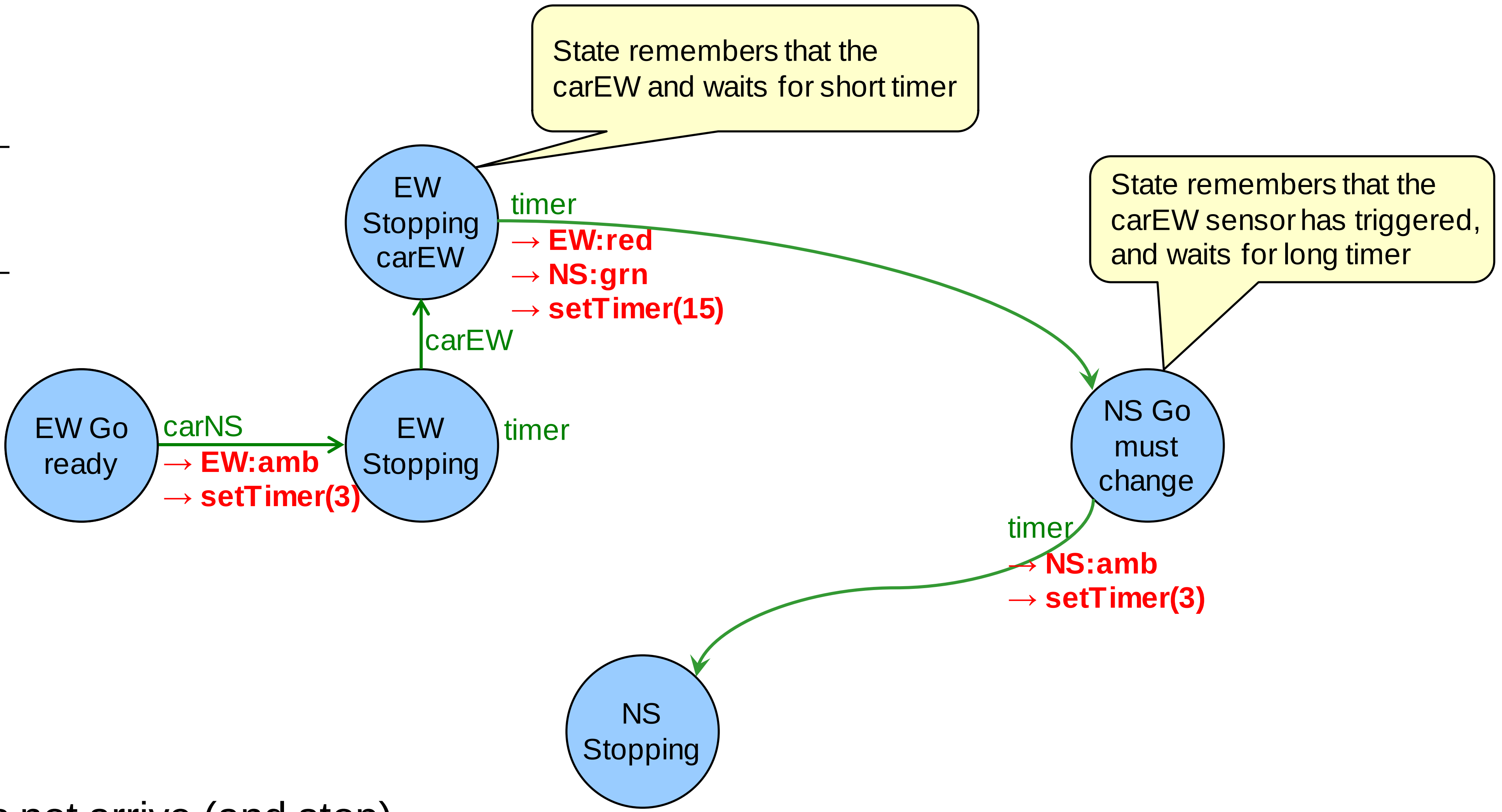



To remember the carEW sensor, we need to define new states.
carEW
EW Go ready
carNS
→ EW:amb
→ setTimer(3)
EW Stopping
timer
→ EW:red
→ NS:grn
→ setTimer(15)
NS Go
timer
State remembers that the timer has ended, but no EW cars have arrived yet.
NS Go ready
This state does **not** remember that the carEW sensor has triggered at the State EW Stopping

What if an EW car does not arrive (and stop) during the EW-Stopping?
Would timer ever be noticed?

States for Arduino sketch for traffic light

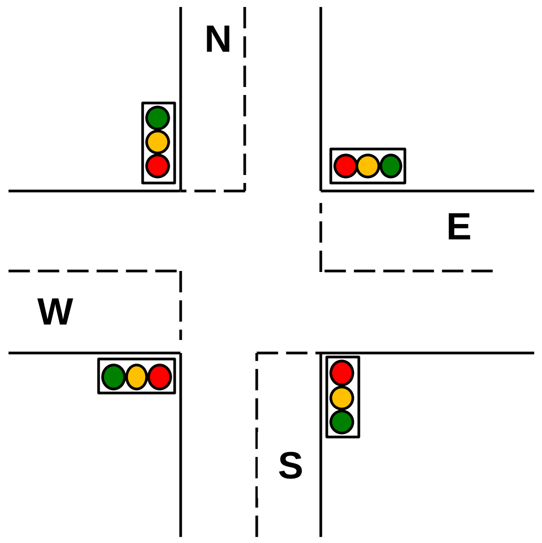


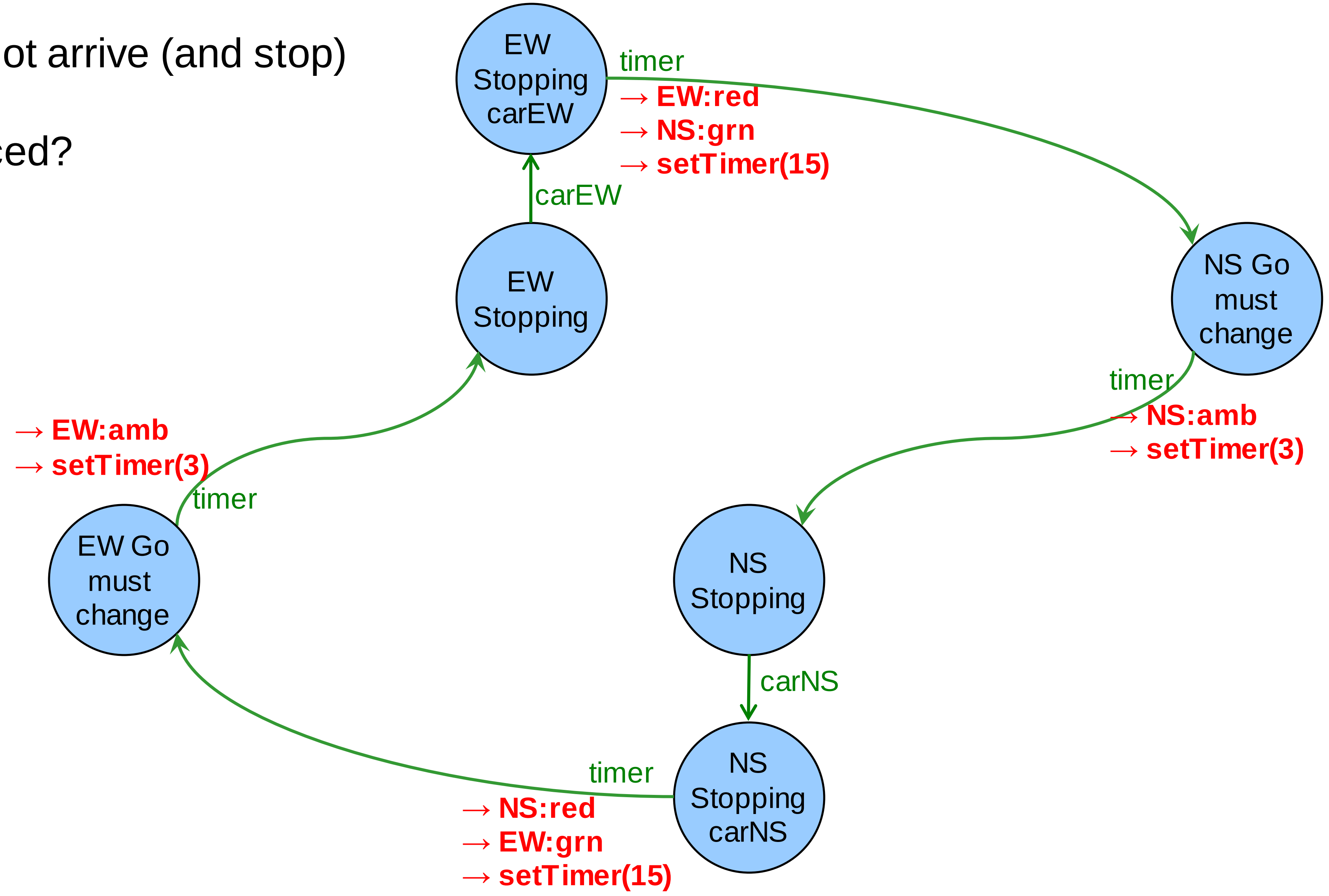


What if an EW car does not arrive (and stop) during the EW-Stopping?
Would timer ever be noticed?

States for Arduino sketch for traffic light

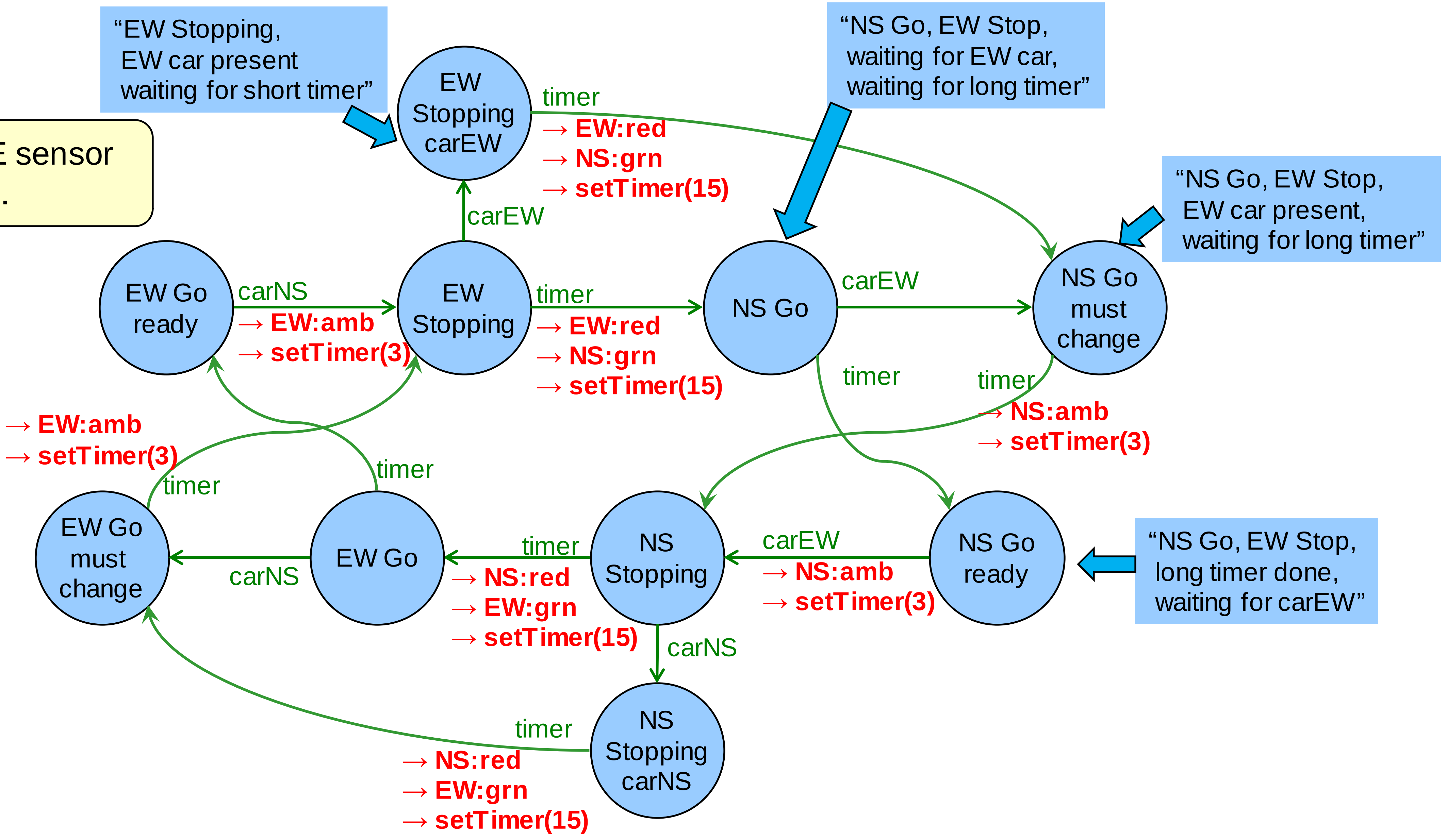
What if an EW car does not arrive (and stop) during the EW-Stopping?
Would timer ever be noticed?





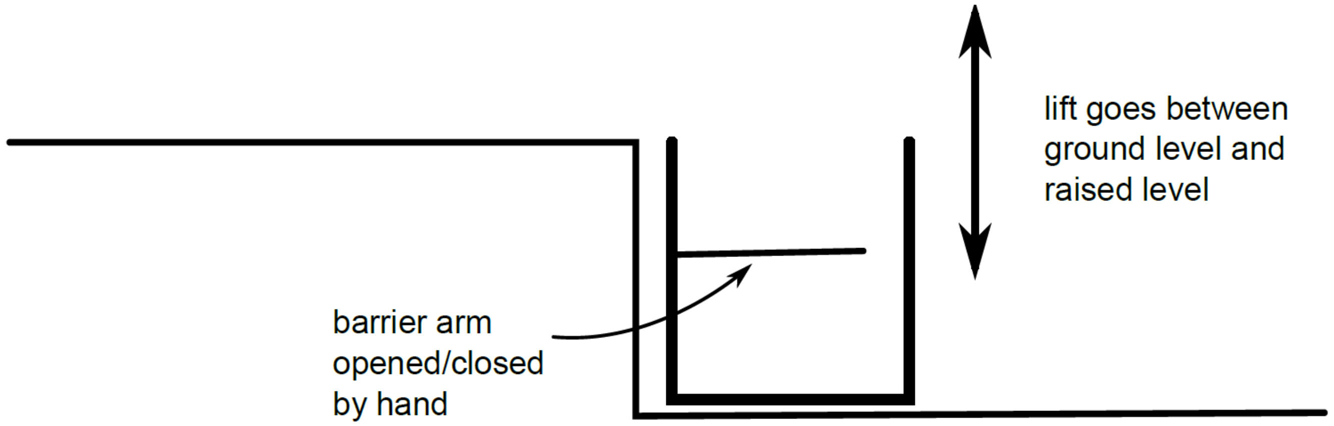
traffic light controller for Lab 3

Always need ONE sensor on each transition.


"EW Stopping, EW car present waiting for short timer"
"NS Go, EW Stop, waiting for EW car, waiting for long timer"
"NS Go, EW Stop, EW car present, waiting for long timer"
"NS Go, EW Stop, long timer done, waiting for carEW"
EW Stopping carEW
timer → EW:red → NS:grn → setTimer(15)
carEW
EW Go ready
carNS → EW:amb → setTimer(3)
EW Stopping
timer → EW:red → NS:grn → setTimer(15)
NS Go
carEW
NS Go must change
timer
timer → NS:amb → setTimer(3)
→ EW:amb → setTimer(3)
timer
timer
EW Go must change
carNS
EW Go
timer → NS:red → EW:grn → setTimer(15)
NS Stopping
carEW → NS:amb → setTimer(3)
NS Go ready
carNS
timer → NS:red → EW:grn → setTimer(15)
NS Stopping carNS

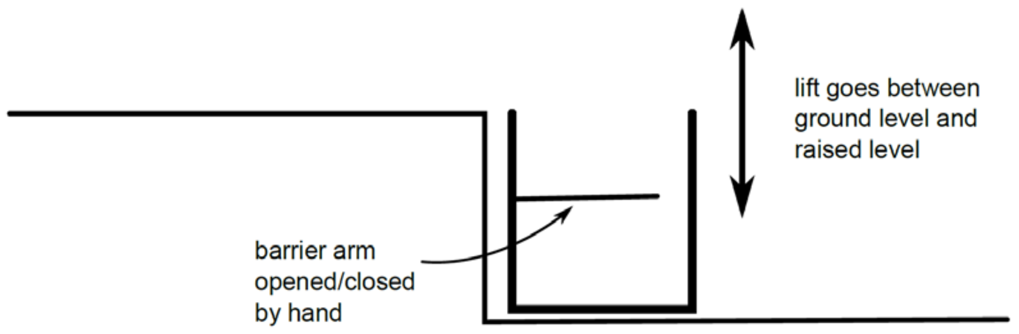
Example: Wheelchair lift

- The wheelchair lift has a barrier arm in front of the platform.
- When the barrier arm is closed, the lift will immediately start moving to the other level.
- The lift will lock the barrier arm while the lift is moving, and only unlocks the arm (allowing the rider to get off) when the lift reaches the other level
- If the lift detects an overload, it will sound a warning buzzer, and will not move when the barrier arm is closed


lift goes between ground level and raised level
barrier arm opened/closed by hand

Example: Wheelchair lift

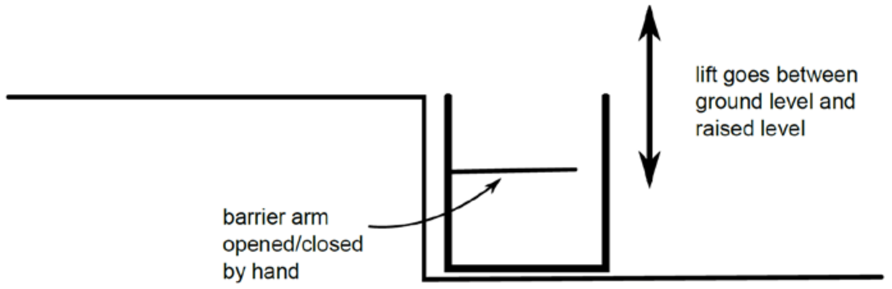
- Sensors:
  - atGround when the lift arrives at ground level
  - atRaised when the lift arrives at the raised level
  - barrierClosed when the barrier arm is closed
  - overload when a rider gets on the lift, taking the load above the limit.
  - withinload when a rider gets off the lift, bringing the load below the limit

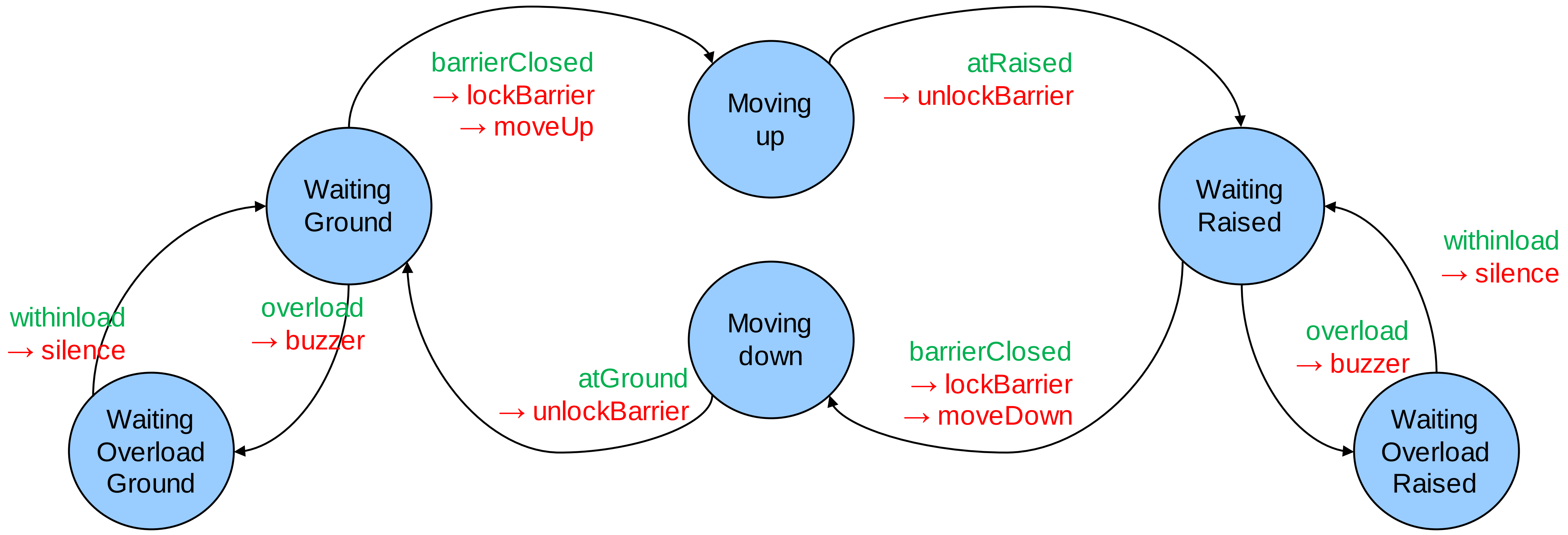

lift goes between ground level and raised level
barrier arm opened/closed by hand

- Actions:
  - moveUp to make the lift start moving up
  - moveDown to make the lift start moving down
  - lockBarrier to lock the barrier arm
  - unlockBarrier to unlock the barrier arm
  - buzzer to turn on the overload warning buzzer
  - silence to turn off the overload warning buzzer

Example: Wheelchair lift

- Sensors:
  - atGround, atRaised, barrierClosed, overload, withinload.
- Actions:
  - moveUp, moveDown, lockBarrier, unlockBarrier, buzzer, silence


lift goes between ground level and raised level
barrier arm opened/closed by hand


Waiting Ground
Moving up
Waiting Raised
Moving down
Waiting Overload Ground
Waiting Overload Raised
barrierClosed → lockBarrier → moveUp
atRaised → unlockBarrier
withinload → silence
overload → buzzer
barrierClosed → lockBarrier → moveDown
atGround → unlockBarrier
withinload → silence
overload → buzzer