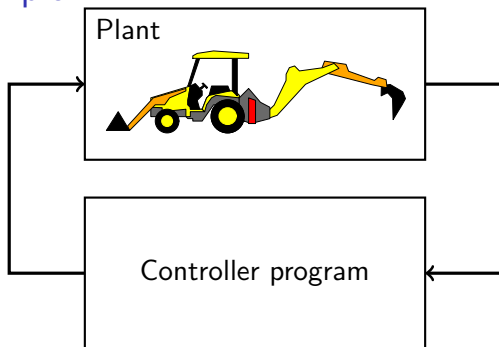


Backhoe example



- ▶ Intended for teaching reactive programming.
- ▶ The *controller* is a regular synchronous program.
- ▶ The *plant* dynamics are modeled using hybrid automata and first-order ODEs.
- ▶ The two interact via boolean flows and signals.
Signals can be compared with 'edge triggering' ($\lceil \cdot \rceil / \lfloor \cdot \rfloor$) or 'function-call triggers' in Simulink.

Modeling a segment

```
let hybrid segment ((min, max, i), maxf, (push, pull, go))
= ((segin, segout), angle) where
```

```
rec der angle = v init i
and error = v_r -. v
and der v = (0.7 /. maxf) *. error +. 0.3 *. z init 0.0
      reset hit(v0) → v0
and der z = error init 0.0 reset hit(_) → 0.0
and v_r = if go then rate else 0.0
```

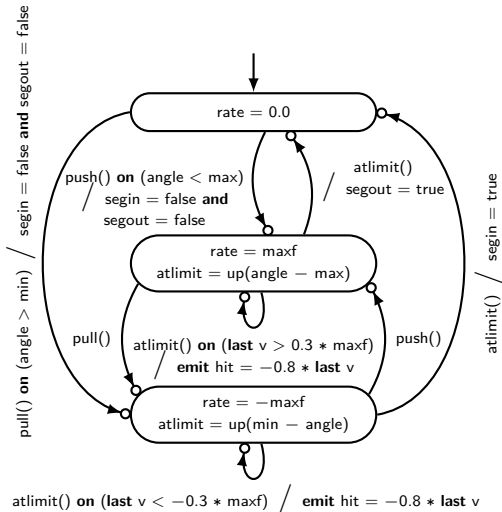
```
and init segin = angle <= min
and init segout = angle >= max
```

and automaton

```
| Stuck → do rate = 0.0
  until push() on (angle < max) then
    do segin = false and segout = false in Pushing
  else pull() on (angle > min) then
    do segin = false and segout = false in Pulling

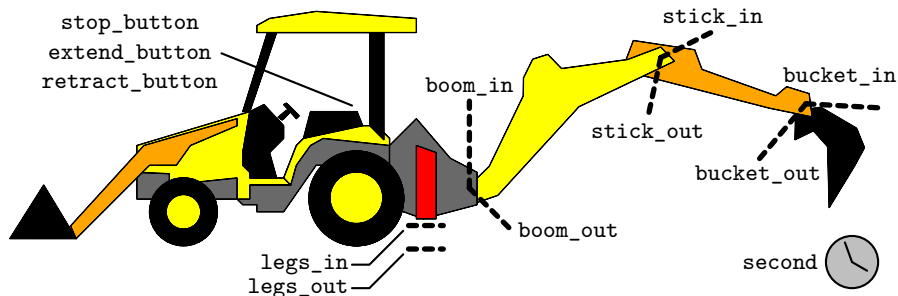
| Pushing → local atlimit in
  do rate = maxf and atlimit = up(angle -. max)
  until atlimit on (last v > 0.3 *. maxf) then do
    emit hit = -0.8 *. last v in Pushing
  else (atlimit) then do segout = true in Stuck
  else pull() then Pulling

| Pulling → local atlimit in
  do rate = -. maxf and atlimit = up(min -. angle)
  until atlimit on (last v < -0.3 *. maxf) then do
    emit hit = -0.8 *. last v in Pulling
  else (atlimit) then do segin = true in Stuck
  else push() then Pushing
```



Backhoe: sensors and actuators

Sensors (plant outputs / controller inputs)



Actuators (plant inputs / controller outputs)

alarm_lamp	legs_extend	boom_pull	stick_pull	bucket_pull
done_lamp	legs_retract	boom_push	stick_push	bucket_push
cancel_lamp	legs_stop	boom_drive	stick_drive	bucket_drive

(pull = in; push = out)