```
MACHINE
  Machine2
REFINES
  Machine1
SEES
  Context2
VARIABLES
  Flag
  Pressure
  Heater
  TimeStamp
  Delta
  DeltaTime
  SensorClock
  ControllerClock
  SensorAddress
  NextHeater
INVARIANTS
  inv2_1   :   DeltaTime ∈ ℕ      //   Used to update the SensorClock
  inv2_2   :   ControllerClock ∈ ℕ        //   Internal clock for Controller
  inv2_3   :   SensorClock ∈ ℕ      //   Internal Clock for sensor
  inv2_4   :   SensorAddress ∈ ℕ      //   Possible sensor addresses
EVENTS
  INITIALISATION   ≙
  STATUS
   ordinary
  BEGIN
   act1   :   Pressure ≔ 55
   act2   :   Heater ≔ High
   act3   :   TimeStamp ≔ 0      //   part of REQ 9
   act5   :   Delta ≔ 0
   act4   :   Flag ≔ Cont
   act6   :   NextHeater ≔ High
   act9   :   SensorAddress ≔ 0
   act10   :   DeltaTime ≔ 0
   act7   :   SensorClock ≔ 0      //   REQ 7
   act8   :   ControllerClock ≔ 0      //   REQ 8
  END

  PressureSens   ≙
  STATUS
   ordinary
  REFINES
   PressureSens
  WHEN
   grd1   :   Flag = Sens
   grd2   :   (Heater = High) ⇒ (Delta ∈ {0, 1, 2, 3})
   grd3   :   (Heater = Low) ⇒ (Delta ∈ {−2, −1, 0})
   grd4   :   (Heater = Off) ⇒ (Delta ∈ {−1, −2})
   grd5   :   Pressure + Delta ∈ ℕ
   grd6   :   DeltaTime ∈ 1‥TMAX      //   Time between 1 and TMAX
   grd7   :   SensorClock + DeltaTime ∈ ℕ
  THEN
   act1   :   Flag ≔ Cont
   act2   :   Pressure ≔ Pressure + Delta
   act4   :   SensorClock ≔ SensorClock + DeltaTime      //   Update SensorClock
   act5   :   TimeStamp ≔ SensorClock      //   Update Timestamp, part of REQ 9
   act6   :   SensorAddress ≔ 1      //   Valid sensorAdress, REQ 10
  END

  SetHeater   ≙
  STATUS
   ordinary
  REFINES
   SetHeater
  WHEN
   grd1   :   Pressure ∈ ℕ
   grd2   :   (Pressure ≥ 61) ⇒ (NextHeater = Off)
   grd3   :   (Pressure ∈ {56, 57, 58, 59, 60}) ⇒ NextHeater = Low
   grd4   :   (Pressure ∈ {50, 51, 52, 53, 54, 55}) ⇒ NextHeater = High
   grd5   :   Flag = Cont
   grd8   :   SensorAddress ∈ LegitimateAddresses      //   Assure adress from sensor is Legitimate, REQ 11
   grd7   :   TimeStamp > ControllerClock      //   Should be a new Timestamp
  THEN
```

```
   act1   :   Heater ≔ NextHeater
   act2   :   Flag ≔ Sens
   act3   :   ControllerClock ≔ TimeStamp      //   Update controller clock
  END

  SafeShutDown   ≙
  STATUS
   ordinary
  REFINES
   SafeShutDown
  WHEN
   grd1   :   Flag = Cont
   grd2   :   (TimeStamp < ControllerClock) ∨ (SensorAddress ∉ LegitimateAddresses)      //   REQ 12, 13, If a non valid timestamp
  THEN
   act1   :   Heater ≔ Off
   act2   :   Flag ≔ Sens
  END

END
```