Part I

We can model this system with the tuple $S = (Q, \Sigma_1, \Sigma_2, q_0, \vee, \wedge)$, where:

$Q = \{ \text{dormant, exit, init, idle, monitoring, error\_diagnosis, safe\_shutdown} \}$

$\Sigma_1 = \{ \text{kill, start, init\_ok, begin\_monitoring, init\_crash, retry\_init, shutdown, sleep, idle\_crash, idle\_rescue, moni\_rescue} \}$

$\Sigma_2 = \{ \text{retry++, broadcast init\_err\_msg, broadcast idle\_err\_msg, broadcast moni\_err\_msg} \}$

$q_0 = \text{dormant}$

$\vee : \text{retry} : \mathbb{N}_0 \; ; \; \text{init\_err\_msg} : \text{string} \; ; \; \text{idle\_err\_msg} : \text{string} \; ; \; \text{moni\_err\_msg} : \text{string}$

$\wedge : \text{transition specification} :$

1. $\longrightarrow dormant$

2. $dormant \xrightarrow{\quad kill \quad} exit$

3. $dormant \xrightarrow{\quad start \quad} init$

4. $init \xrightarrow{\quad kill \quad} exit$

5. $init \xrightarrow{\quad init\_ok \quad} idle$

6. $idle \xrightarrow{\quad kill \quad} exit$

7. $idle \xrightarrow{\quad begin\_monitoring \quad} monitoring$

8. $init \xrightarrow{\quad init\_crash/broadcast\,init\_err\_msg \quad} error\_diagnosis$

9. $error\_diagnosis \xrightarrow{\quad retry\_init[retry<3]/retry++ \quad} init$

10. $error\_diagnosis \xrightarrow{\quad kill \quad} exit$

11. $error\_diagnosis \xrightarrow{\quad shutdown[retry>2] \quad} safe\_shutdown$

12. $safe\_shutdown \xrightarrow{\quad kill \quad} exit$

13. $safe\_shutdown \xrightarrow{\quad sleep \quad} dormant$

14. $idle \xrightarrow{\quad idle\_crash/broadcast\,idle\_err\_msg \quad} error\_diagnosis$

15. $error\_diagnosis \xrightarrow{\quad idle\_rescue \quad} idle$

16. $monitoring \xrightarrow{\quad monitor\_crash/broadcast\,moni\_err\_msg \quad} error\_diagnosis$

17. $error\_diagnosis \xrightarrow{\quad moni\_rescue \quad} monitoring$

18. $monitoring \xrightarrow{\quad kill \quad} exit$

Part II

As **init** is a composed state, we define it as the tuple $S = (Q, \Sigma_1, q_0, \wedge)$, where:

$$Q = \{ boot\_hw, \ senchk, \ tchk, \ psichk, \ ready \}$$

$$\Sigma_1 = \{ hw\_ok, \ senok, \ psi\_ok \}$$

$$q_0 = boot\_hw$$

$\wedge : transition \ specification :$

1. $\longrightarrow boot\_hw$

2. $boot\_hw \xrightarrow{\ hw\_ok\ } senchk$

3. $senchk \xrightarrow{\ senok\ } tchk$

4. $tchk \xrightarrow{\ t\_ok\ } psichk$

5. $psichk \xrightarrow{\ psi\_ok\ } ready$

Part III

As **monitoring** is a composed state, we define it as the tuple $S = \left(Q, \Sigma_1, \Sigma_2, q_0, \vee, \wedge \right)$, where:

$Q = \left\{ \ monidle, \ regulate\_enironment, \ lockdown \ \right\}$

$\Sigma_1 = \left\{ \ no\_contagion, \ after\_100ms, \ contagion\_alert, \ purge\_succ \ \right\}$

$\Sigma_2 = \left\{ \ broadcast \ FACILITY\_CRIT\_MESG \ \right\}$

$q_0 = monidle$

$\vee : FACILITY\_CRIT\_MESG : string \ ; \ inlockdown : Boolean$

$\wedge : transition \ specification :$

1. $\longrightarrow monidle$

2. $monidle \xrightarrow{\ no\_contagion\ } regulate\_environment$

3. $regulate\_environment \xrightarrow{\ after\_100ms\ } monidle$

4. $regulate\_environment \xrightarrow{\ contagion\_alert\,/\,broadcast\,FACILITY\_CRIT\_MESG,\ inlockdown:=true\ } psichk$

5. $lockdown \xrightarrow{\ purge\_succ\,/\,inlockdown:=false\ } monidle$

super :

6. $monitoring \xrightarrow{\ monitor\_crash\,[!inlockdown]\,/\,broadcast\,moni\_err\_msg\ } error\_diagnosis$

7. $monitoring \xrightarrow{\ kill\,[!inlockdown]\ } exit$

Part IV

As **lockdown** is a composed state, we define it as the tuple $S = (Q, \Sigma_1, \Sigma_2, q_0, \vee, \wedge)$, where

$Q = \{ \; prep\_vpurge, \; alt\_temp, \; alt\_psi, \; risk\_assess, \; safe\_status, \; exit \; \}$

$\Sigma_1 = \{ \; initiate\_purge, \; tcyc\_comp, \; psicyc\_comp \; \}$

$\Sigma_2 = \{ \; lock\_doors, \; unlock\_doors \; \}$

$q_0 = prep\_vpurge$

$\vee : risk : \{ risk \in Q \mid 0 \le risk \le 1 \}$

$\wedge : transition \; specification :$

1. $\longrightarrow prep\_vpurge$

2. $prep\_vpurge \xrightarrow{\quad initiate\_purge/lock\_doors \quad} alt\_temp$

3. $prep\_vpurge \xrightarrow{\quad initiate\_purge/lock\_doors \quad} alt\_psi$

4. $alt\_temp \xrightarrow{\quad tcyc\_comp \quad} risk\_assess$

5. $alt\_psi \xrightarrow{\quad psicyc\_comp \quad} risk\_assess$

6. $risk\_assess \xrightarrow{\quad [risk>0.01] \quad} prep\_vpurge$

7. $risk\_assess \xrightarrow{\quad [risk<0.01]/unlock\_doors \quad} safe\_status$

8. $safe\_status \longrightarrow exit$

Part V

As **error_diagnosis** is a composed state, we define it as the tuple $S = \left(Q, \Sigma_1, q_0, \vee, \wedge\right)$, where

$Q = \left\{\ error\_rcv,\ applicable\_rescue,\ exit,\ reset\_module\_data\ \right\}$

$\Sigma_1 = \left\{\ apply\_protocol\_issue,\ reset\_to\_stable\ \right\}$

$q_0 = error\_rcv$

$\vee : err\_protocol\_def : Boolean$

$\wedge : transition\ specification :$

1. $\longrightarrow error\_rcv$

2. $error\_rcv \xrightarrow{\ [err\_protocol\_def]\ } applicable\_rescue$

3. $applicable\_rescue \xrightarrow{\ apply\_protocol\_rescue\ } exit$

4. $error\_rcv \xrightarrow{\ [!err\_protocol\_def]\ } reset\_module\_data$

5. $reset\_module\_data \xrightarrow{\ reset\_to\_stable\ } exit$