```
20  #include "Inner.h"
21  #include "Inner_private.h"
22
23  /* Block states (default storage) */
24  DW_Inner_T Inner_DW;
25
26  /* External inputs (root inport signals with default storage) */
27  ExtU_Inner_T Inner_U;
28
29  /* External outputs (root outports fed by signals with default storage) */
30  ExtY_Inner_T Inner_Y;
31
32  /* Real-time model */
33  static RT_MODEL_Inner_T Inner_M_;
34  RT_MODEL_Inner_T *const Inner_M = &Inner_M_;
35
36  /* Model step function */
37  void Inner_step(void)
38  {
39    real_T tmp_1[4];
40    real_T rtb_Aod[2];
41    real_T tmp[2];
42    real_T tmp_0[2];
43    real_T DiscreteFilterz2_tmp;
44    real_T rtb_Sum6;
45    real_T rtb_TSamp;
46    real_T tmp_2;
47    real_T tmp_3;
48    int32_T i;
49
50    /* Outputs for Atomic SubSystem: '<Root>/Inner-loop controller (2)' */
51    /* Sum: '<S1>/Sum6' incorporates:
52     *  Inport: '<Root>/gz'
53     *  Inport: '<Root>/z_s1'
54     */
55    rtb_Sum6 = Inner_U.z_s1 - Inner_U.gz;
56
57    /* SampleTimeMath: '<S2>/TSamp'
58     *
59     * About '<S2>/TSamp':
60     *  y = u * K where K = 1 / ( w * Ts )
61     */
62    rtb_TSamp = rtb_Sum6 * 800.0;
63
64    /* DiscreteFilter: '<S1>/Discrete Filter (z2)' incorporates:
```

```
65       *  Sum: '<S2>/Diff'
66       *  UnitDelay: '<S2>/UD'
67       *
68       * Block description for '<S2>/Diff':
69       *
70       *  Add in CPU
71       *
72       * Block description for '<S2>/UD':
73       *
74       *  Store in Global RAM
75       */
76      DiscreteFilterz2_tmp = ((rtb_TSamp - Inner_DW.UD_DSTATE) - (-0.975) *
77        Inner_DW.DiscreteFilterz2_states) / 1.0;
78
79      /* Saturate: '<S1>/Saturation (z2)' incorporates:
80       *  DiscreteFilter: '<S1>/Discrete Filter (z2)'
81       *  DiscreteIntegrator: '<S1>/Discrete-Time Integrator (z2)'
82       *  Gain: '<S1>/z_kd2'
83       *  Gain: '<S1>/z_ki2'
84       *  Gain: '<S1>/z_kp2'
85       *  Sum: '<S1>/Sum7'
86       */
87      Inner_Y.z_s2 = (0.025 * DiscreteFilterz2_tmp + 0.0 *
88                      Inner_DW.DiscreteFilterz2_states) * 0.0 + (1000.0 * rtb_Sum6 +
89        0.0 * Inner_DW.DiscreteTimeIntegratorz2_DSTATE);
90
91      /* Saturate: '<S1>/Saturation (z2)' */
92      if (Inner_Y.z_s2 > 800.0) {
93        /* Saturate: '<S1>/Saturation (z2)' */
94        Inner_Y.z_s2 = 800.0;
95      } else if (Inner_Y.z_s2 < (-800.0)) {
96        /* Saturate: '<S1>/Saturation (z2)' */
97        Inner_Y.z_s2 = (-800.0);
98      }
99
100     /* End of Saturate: '<S1>/Saturation (z2)' */
101
102     /* Gain: '<S1>/Ha' incorporates:
103      *  Inport: '<Root>/x_s1'
104      *  Inport: '<Root>/y_s1'
105      */
106     tmp[0] = 100.26787709298573 * Inner_U.x_s1 + 8.42201539480736E-15 *
107       Inner_U.y_s1;
108
109     /* Gain: '<S1>/Ga' incorporates:
110      *  DiscreteIntegrator: '<S1>/Discrete-Time Integrator (e_x)'
```

```
111       *  DiscreteIntegrator: '<S1>/Discrete-Time Integrator (e_y)'
112       */
113      tmp_0[0] = 100.00000000000009 * Inner_DW.DiscreteTimeIntegratore_x_DSTAT +
114        (-1.563050990616535E-14) * Inner_DW.DiscreteTimeIntegratore_y_DSTAT;
115
116      /* Gain: '<S1>/Ha' incorporates:
117       *  Inport: '<Root>/x_s1'
118       *  Inport: '<Root>/y_s1'
119       */
120      tmp[1] = 3.8634117533128733E-15 * Inner_U.x_s1 + 100.26787709298584 *
121        Inner_U.y_s1;
122
123      /* Gain: '<S1>/Ga' incorporates:
124       *  DiscreteIntegrator: '<S1>/Discrete-Time Integrator (e_x)'
125       *  DiscreteIntegrator: '<S1>/Discrete-Time Integrator (e_y)'
126       */
127      tmp_0[1] = (-2.0771705684243972E-14) *
128        Inner_DW.DiscreteTimeIntegratore_x_DSTAT + 100.00000000000017 *
129        Inner_DW.DiscreteTimeIntegratore_y_DSTAT;
130
131      /* Gain: '<S1>/Cod' incorporates:
132       *  UnitDelay: '<S1>/Unit Delay (State estimator)'
133       */
134      for (i = 0; i < 4; i++) {
135        tmp_1[i] = Inner_ConstP.Cod_Gain[i + 4] *
136          Inner_DW.UnitDelayStateestimator_DSTATE[1] + Inner_ConstP.Cod_Gain[i] *
137          Inner_DW.UnitDelayStateestimator_DSTATE[0];
138      }
139
140      /* End of Gain: '<S1>/Cod' */
141
142      /* SignalConversion generated from: '<S1>/Fa' */
143      tmp_2 = tmp_1[0];
144      tmp_3 = tmp_1[1];
145
146      /* Sum: '<S1>/Sum3' incorporates:
147       *  Gain: '<S1>/Fa'
148       *  Inport: '<Root>/gx'
149       *  Inport: '<Root>/gy'
150       */
151      for (i = 0; i < 2; i++) {
152        rtb_Aod[i] = (((Inner_ConstP.Fa_Gain[i + 2] * tmp_3 + Inner_ConstP.Fa_Gain[i]
153                        * tmp_2) + Inner_ConstP.Fa_Gain[i + 4] * Inner_U.gx) +
154                      Inner_ConstP.Fa_Gain[i + 6] * Inner_U.gy) + (tmp[i] + tmp_0[i]);
155      }
156
```

```
157     /* End of Sum: '<S1>/Sum3' */
158
159     /* Saturate: '<S1>/Saturation (y2)' */
160     if (rtb_Aod[1] > 800.0) {
161       /* Saturate: '<S1>/Saturation (y2)' */
162       Inner_Y.y_s2 = 800.0;
163     } else if (rtb_Aod[1] < (-800.0)) {
164       /* Saturate: '<S1>/Saturation (y2)' */
165       Inner_Y.y_s2 = (-800.0);
166     } else {
167       /* Saturate: '<S1>/Saturation (y2)' */
168       Inner_Y.y_s2 = rtb_Aod[1];
169     }
170
171     /* End of Saturate: '<S1>/Saturation (y2)' */
172
173     /* Saturate: '<S1>/Saturation (x2)' */
174     if (rtb_Aod[0] > 800.0) {
175       /* Saturate: '<S1>/Saturation (x2)' */
176       Inner_Y.x_s2 = 800.0;
177     } else if (rtb_Aod[0] < (-800.0)) {
178       /* Saturate: '<S1>/Saturation (x2)' */
179       Inner_Y.x_s2 = (-800.0);
180     } else {
181       /* Saturate: '<S1>/Saturation (x2)' */
182       Inner_Y.x_s2 = rtb_Aod[0];
183     }
184
185     /* End of Saturate: '<S1>/Saturation (x2)' */
186
187     /* Update for DiscreteIntegrator: '<S1>/Discrete-Time Integrator (z2)' */
188     Inner_DW.DiscreteTimeIntegratorz2_DSTATE += 0.00125 * rtb_Sum6;
189     if (Inner_DW.DiscreteTimeIntegratorz2_DSTATE >= 2.0) {
190       Inner_DW.DiscreteTimeIntegratorz2_DSTATE = 2.0;
191     } else if (Inner_DW.DiscreteTimeIntegratorz2_DSTATE <= (-2.0)) {
192       Inner_DW.DiscreteTimeIntegratorz2_DSTATE = (-2.0);
193     }
194
195     /* End of Update for DiscreteIntegrator: '<S1>/Discrete-Time Integrator (z2)' */
196
197     /* Update for UnitDelay: '<S2>/UD'
198      *
199      * Block description for '<S2>/UD':
200      *
201      *  Store in Global RAM
202      */
```

```
203     Inner_DW.UD_DSTATE = rtb_TSamp;
204
205     /* Update for DiscreteFilter: '<S1>/Discrete Filter (z2)' */
206     Inner_DW.DiscreteFilterz2_states = DiscreteFilterz2_tmp;
207
208     /* Update for DiscreteIntegrator: '<S1>/Discrete-Time Integrator (e_x)' incorporates:
209      *  Inport: '<Root>/gx'
210      *  Inport: '<Root>/x_s1'
211      *  Sum: '<S1>/Sum2'
212      */
213     Inner_DW.DiscreteTimeIntegratore_x_DSTAT += (Inner_U.x_s1 - Inner_U.gx) *
214       0.00125;
215     if (Inner_DW.DiscreteTimeIntegratore_x_DSTAT >= 20.0) {
216       Inner_DW.DiscreteTimeIntegratore_x_DSTAT = 20.0;
217     } else if (Inner_DW.DiscreteTimeIntegratore_x_DSTAT <= (-20.0)) {
218       Inner_DW.DiscreteTimeIntegratore_x_DSTAT = (-20.0);
219     }
220
221     /* End of Update for DiscreteIntegrator: '<S1>/Discrete-Time Integrator (e_x)' */
222
223     /* Update for DiscreteIntegrator: '<S1>/Discrete-Time Integrator (e_y)' incorporates:
224      *  Inport: '<Root>/gy'
225      *  Inport: '<Root>/y_s1'
226      *  Sum: '<S1>/Sum4'
227      */
228     Inner_DW.DiscreteTimeIntegratore_y_DSTAT += (Inner_U.y_s1 - Inner_U.gy) *
229       0.00125;
230     if (Inner_DW.DiscreteTimeIntegratore_y_DSTAT >= 20.0) {
231       Inner_DW.DiscreteTimeIntegratore_y_DSTAT = 20.0;
232     } else if (Inner_DW.DiscreteTimeIntegratore_y_DSTAT <= (-20.0)) {
233       Inner_DW.DiscreteTimeIntegratore_y_DSTAT = (-20.0);
234     }
235
236     /* End of Update for DiscreteIntegrator: '<S1>/Discrete-Time Integrator (e_y)' */
237
238     /* Saturate: '<S1>/Saturation (x2,se)' */
239     if (Inner_Y.x_s2 > 60.0) {
240       tmp_2 = 60.0;
241     } else if (Inner_Y.x_s2 < (-60.0)) {
242       tmp_2 = (-60.0);
243     } else {
244       tmp_2 = Inner_Y.x_s2;
245     }
246
247     /* Saturate: '<S1>/Saturation (y2,se)' */
248     if (Inner_Y.y_s2 > 60.0) {
```

```
249       tmp_3 = 60.0;
250     } else if (Inner_Y.y_s2 < (-60.0)) {
251       tmp_3 = (-60.0);
252     } else {
253       tmp_3 = Inner_Y.y_s2;
254     }
255
256     for (i = 0; i < 2; i++) {
257       /* Gain: '<S1>/Bod' incorporates:
258        *  Inport: '<Root>/gx'
259        *  Inport: '<Root>/gy'
260        *  Saturate: '<S1>/Saturation (x2,se)'
261        *  Saturate: '<S1>/Saturation (y2,se)'
262        */
263       tmp[i] = ((Inner_ConstP.Bod_Gain[i + 2] * tmp_3 + Inner_ConstP.Bod_Gain[i] *
264                  tmp_2) + Inner_ConstP.Bod_Gain[i + 4] * Inner_U.gx) +
265         Inner_ConstP.Bod_Gain[i + 6] * Inner_U.gy;
266
267       /* Gain: '<S1>/Aod' incorporates:
268        *  UnitDelay: '<S1>/Unit Delay (State estimator)'
269        */
270       tmp_0[i] = Inner_ConstP.Aod_Gain[i + 2] *
271         Inner_DW.UnitDelayStateestimator_DSTATE[1] + Inner_ConstP.Aod_Gain[i] *
272         Inner_DW.UnitDelayStateestimator_DSTATE[0];
273     }
274
275     /* Update for UnitDelay: '<S1>/Unit Delay (State estimator)' incorporates:
276      *  Sum: '<S1>/Sum8'
277      */
278     Inner_DW.UnitDelayStateestimator_DSTATE[0] = tmp[0] + tmp_0[0];
279     Inner_DW.UnitDelayStateestimator_DSTATE[1] = tmp[1] + tmp_0[1];
280
281     /* Outport: '<Root>/motor1_pwm' incorporates:
282      *  Gain: '<S1>/C_dist_dir'
283      *  Gain: '<S1>/C_dist_lat'
284      *  Gain: '<S1>/C_dist_lon'
285      *  Gain: '<S1>/C_dist_thr'
286      *  Inport: '<Root>/motor_thr'
287      *  Sum: '<S1>/Sum1'
288      */
289     Inner_Y.motor1_pwm = (((-1.0) * Inner_Y.x_s2 + (-1.0) * Inner_Y.y_s2) + 1.0 *
290                           Inner_Y.z_s2) + 1.0 * Inner_U.motor_thr;
291
292     /* Outport: '<Root>/motor2_pwm' incorporates:
293      *  Gain: '<S1>/C_dist_dir'
294      *  Gain: '<S1>/C_dist_lat'
```

```
295       *  Gain: '<S1>/C_dist_lon'
296       *  Gain: '<S1>/C_dist_thr'
297       *  Inport: '<Root>/motor_thr'
298       *  Sum: '<S1>/Sum1'
299       */
300      Inner_Y.motor2_pwm = ((1.0 * Inner_Y.x_s2 + (-1.0) * Inner_Y.y_s2) + (-1.0) *
301                            Inner_Y.z_s2) + 1.0 * Inner_U.motor_thr;
302
303      /* Outport: '<Root>/motor3_pwm' incorporates:
304       *  Gain: '<S1>/C_dist_dir'
305       *  Gain: '<S1>/C_dist_lat'
306       *  Gain: '<S1>/C_dist_lon'
307       *  Gain: '<S1>/C_dist_thr'
308       *  Inport: '<Root>/motor_thr'
309       *  Sum: '<S1>/Sum1'
310       */
311      Inner_Y.motor3_pwm = ((1.0 * Inner_Y.x_s2 + 1.0 * Inner_Y.y_s2) + 1.0 *
312                            Inner_Y.z_s2) + 1.0 * Inner_U.motor_thr;
313
314      /* Outport: '<Root>/motor4_pwm' incorporates:
315       *  Gain: '<S1>/C_dist_dir'
316       *  Gain: '<S1>/C_dist_lat'
317       *  Gain: '<S1>/C_dist_lon'
318       *  Gain: '<S1>/C_dist_thr'
319       *  Inport: '<Root>/motor_thr'
320       *  Sum: '<S1>/Sum1'
321       */
322      Inner_Y.motor4_pwm = (((-1.0) * Inner_Y.x_s2 + 1.0 * Inner_Y.y_s2) + (-1.0) *
323                            Inner_Y.z_s2) + 1.0 * Inner_U.motor_thr;
324
325      /* End of Outputs for SubSystem: '<Root>/Inner-loop controller (2)' */
326    }
```