**Flightcontrolsystem.c**

**File:**[**flightControlSystem.c**](matlab:coder.internal.editUrlTextFile('file:///C:/Users/Welcome/MATLAB/Projects/examples/asbQuadcopter23/work/flightControlSystem_grt_rtw/flightControlSystem.c'))

|  |  |
| --- | --- |
| *1* | */\** |
| *2* | *\* flightControlSystem.c* |
| *3* | *\** |
| *4* | *\* Code generation for model "flightControlSystem".* |
| *5* | *\** |
| *6* | *\* Model version : 1.137* |
| *7* | *\* Simulink Coder version : 9.3 (R2020a) 18-Nov-2019* |
| *8* | *\* C source code generated on : Fri Nov 20 20:59:57 2020* |
| *9* | *\** |
| *10* | *\* Target selection: grt.tlc* |
| *11* | *\* Note: GRT includes extra infrastructure and instrumentation for prototyping* |
| *12* | *\* Embedded hardware selection: ARM Compatible->ARM 9* |
| *13* | *\* Code generation objectives: Unspecified* |
| *14* | *\* Validation result: Not run* |
| *15* | *\*/* |
| *16* |  |
| *17* | **#include "flightControlSystem.h"** |
| *18* | **#include "flightControlSystem\_private.h"** |
| *19* |  |
| *20* | **const** statesEstim\_t flightControlSystem\_rtZstatesEstim\_t = **{** |
| *21* | 0.0F, */\* X \*/* |
| *22* | 0.0F, */\* Y \*/* |
| *23* | 0.0F, */\* Z \*/* |
| *24* | 0.0F, */\* yaw \*/* |
| *25* | 0.0F, */\* pitch \*/* |
| *26* | 0.0F, */\* roll \*/* |
| *27* | 0.0F, */\* dx \*/* |
| *28* | 0.0F, */\* dy \*/* |
| *29* | 0.0F, */\* dz \*/* |
| *30* | 0.0F, */\* p \*/* |
| *31* | 0.0F, */\* q \*/* |
| *32* | 0.0F */\* r \*/* |
| *33* | **}** ; */\* statesEstim\_t ground \*/* |
| *34* |  |
| *35* | */\* Exported block signals \*/* |
| *36* | CommandBus cmd\_inport; */\* '<Root>/AC cmd' \*/* |
| *37* | SensorsBus sensor\_inport; */\* '<Root>/Sensors' \*/* |
| *38* | real32\_T motors\_outport[4]; */\* '<S1>/controller' \*/* |
| *39* | uint8\_T flag\_outport; */\* '<S3>/Merge' \*/* |
| *40* |  |
| *41* | */\* Block signals (default storage) \*/* |
| *42* | B\_flightControlSystem\_T flightControlSystem\_B; |
| *43* |  |
| *44* | */\* Block states (default storage) \*/* |
| *45* | DW\_flightControlSystem\_T flightControlSystem\_DW; |
| *46* |  |
| *47* | */\* External inputs (root inport signals with default storage) \*/* |
| *48* | ExtU\_flightControlSystem\_T flightControlSystem\_U; |
| *49* |  |
| *50* | */\* External outputs (root outports fed by signals with default storage) \*/* |
| *51* | ExtY\_flightControlSystem\_T flightControlSystem\_Y; |
| *52* |  |
| *53* | */\* Real-time model \*/* |
| *54* | RT\_MODEL\_flightControlSystem\_T flightControlSystem\_M\_; |
| *55* | RT\_MODEL\_flightControlSystem\_T \***const** flightControlSystem\_M = |
| *56* | &flightControlSystem\_M\_; |
| *57* |  |
| *58* | */\* System initialize for atomic system: '<S1>/Logging' \*/* |
| *59* | **void** flightControlSystem\_Logging\_Init(RT\_MODEL\_flightControlSystem\_T \* **const** |
| *60* | flightControlSystem\_M, DW\_Logging\_flightControlSystem\_T \*localDW) |
| *61* | **{** |
| *62* | */\* SetupRuntimeResources for ToWorkspace: '<S4>/To Workspace6' \*/* |
| *63* | **{** |
| *64* | **static** int\_T rt\_ToWksWidths[] = **{** 10 **}**; |
| *65* |  |
| *66* | **static** int\_T rt\_ToWksNumDimensions[] = **{** 1 **}**; |
| *67* |  |
| *68* | **static** int\_T rt\_ToWksDimensions[] = **{** 10 **}**; |
| *69* |  |
| *70* | **static** boolean\_T rt\_ToWksIsVarDims[] = **{** 0 **}**; |
| *71* |  |
| *72* | **static** **void** \*rt\_ToWksCurrSigDims[] = **{** (NULL) **}**; |
| *73* |  |
| *74* | **static** int\_T rt\_ToWksCurrSigDimsSize[] = **{** 4 **}**; |
| *75* |  |
| *76* | **static** BuiltInDTypeId rt\_ToWksDataTypeIds[] = **{** SS\_SINGLE **}**; |
| *77* |  |
| *78* | **static** int\_T rt\_ToWksComplexSignals[] = **{** 0 **}**; |
| *79* |  |
| *80* | **static** int\_T rt\_ToWksFrameData[] = **{** 0 **}**; |
| *81* |  |
| *82* | **static** RTWPreprocessingFcnPtr rt\_ToWksLoggingPreprocessingFcnPtrs[] = **{** |
| *83* | (NULL) |
| *84* | **}**; |
| *85* |  |
| *86* | **static** **const** char\_T \*rt\_ToWksLabels[] = **{** "" **}**; |
| *87* |  |
| *88* | **static** RTWLogSignalInfo rt\_ToWksSignalInfo = **{** |
| *89* | 1, |
| *90* | rt\_ToWksWidths, |
| *91* | rt\_ToWksNumDimensions, |
| *92* | rt\_ToWksDimensions, |
| *93* | rt\_ToWksIsVarDims, |
| *94* | rt\_ToWksCurrSigDims, |
| *95* | rt\_ToWksCurrSigDimsSize, |
| *96* | rt\_ToWksDataTypeIds, |
| *97* | rt\_ToWksComplexSignals, |
| *98* | rt\_ToWksFrameData, |
| *99* | rt\_ToWksLoggingPreprocessingFcnPtrs, |
| *100* |  |
| *101* | **{** rt\_ToWksLabels **}**, |
| *102* | (NULL), |
| *103* | (NULL), |
| *104* | (NULL), |
| *105* |  |
| *106* | **{** (NULL) **}**, |
| *107* |  |
| *108* | **{** (NULL) **}**, |
| *109* | (NULL), |
| *110* | (NULL) |
| *111* | **}**; |
| *112* |  |
| *113* | **static** **const** char\_T rt\_ToWksBlockName[] = |
| *114* | "flightControlSystem/Flight Control System/Logging/To Workspace6"; |
| *115* | localDW->ToWorkspace6\_PWORK.LoggedData = rt\_CreateStructLogVar( |
| *116* | flightControlSystem\_M->rtwLogInfo, |
| *117* | 0.0, |
| *118* | rtmGetTFinal(flightControlSystem\_M), |
| *119* | flightControlSystem\_M->Timing.stepSize0, |
| *120* | (&rtmGetErrorStatus(flightControlSystem\_M)), |
| *121* | "sensor", |
| *122* | 1, |
| *123* | 0, |
| *124* | 1, |
| *125* | 0.005, |
| *126* | &rt\_ToWksSignalInfo, |
| *127* | rt\_ToWksBlockName); |
| *128* | **if** (localDW->ToWorkspace6\_PWORK.LoggedData == (NULL)) |
| *129* | **return**; |
| *130* | **}** |
| *131* |  |
| *132* | */\* SetupRuntimeResources for ToWorkspace: '<S4>/To Workspace4' \*/* |
| *133* | **{** |
| *134* | **static** int\_T rt\_ToWksWidths[] = **{** 8 **}**; |
| *135* |  |
| *136* | **static** int\_T rt\_ToWksNumDimensions[] = **{** 1 **}**; |
| *137* |  |
| *138* | **static** int\_T rt\_ToWksDimensions[] = **{** 8 **}**; |
| *139* |  |
| *140* | **static** boolean\_T rt\_ToWksIsVarDims[] = **{** 0 **}**; |
| *141* |  |
| *142* | **static** **void** \*rt\_ToWksCurrSigDims[] = **{** (NULL) **}**; |
| *143* |  |
| *144* | **static** int\_T rt\_ToWksCurrSigDimsSize[] = **{** 4 **}**; |
| *145* |  |
| *146* | **static** BuiltInDTypeId rt\_ToWksDataTypeIds[] = **{** SS\_SINGLE **}**; |
| *147* |  |
| *148* | **static** int\_T rt\_ToWksComplexSignals[] = **{** 0 **}**; |
| *149* |  |
| *150* | **static** int\_T rt\_ToWksFrameData[] = **{** 0 **}**; |
| *151* |  |
| *152* | **static** RTWPreprocessingFcnPtr rt\_ToWksLoggingPreprocessingFcnPtrs[] = **{** |
| *153* | (NULL) |
| *154* | **}**; |
| *155* |  |
| *156* | **static** **const** char\_T \*rt\_ToWksLabels[] = **{** "" **}**; |
| *157* |  |
| *158* | **static** RTWLogSignalInfo rt\_ToWksSignalInfo = **{** |
| *159* | 1, |
| *160* | rt\_ToWksWidths, |
| *161* | rt\_ToWksNumDimensions, |
| *162* | rt\_ToWksDimensions, |
| *163* | rt\_ToWksIsVarDims, |
| *164* | rt\_ToWksCurrSigDims, |
| *165* | rt\_ToWksCurrSigDimsSize, |
| *166* | rt\_ToWksDataTypeIds, |
| *167* | rt\_ToWksComplexSignals, |
| *168* | rt\_ToWksFrameData, |
| *169* | rt\_ToWksLoggingPreprocessingFcnPtrs, |
| *170* |  |
| *171* | **{** rt\_ToWksLabels **}**, |
| *172* | (NULL), |
| *173* | (NULL), |
| *174* | (NULL), |
| *175* |  |
| *176* | **{** (NULL) **}**, |
| *177* |  |
| *178* | **{** (NULL) **}**, |
| *179* | (NULL), |
| *180* | (NULL) |
| *181* | **}**; |
| *182* |  |
| *183* | **static** **const** char\_T rt\_ToWksBlockName[] = |
| *184* | "flightControlSystem/Flight Control System/Logging/To Workspace4"; |
| *185* | localDW->ToWorkspace4\_PWORK.LoggedData = rt\_CreateStructLogVar( |
| *186* | flightControlSystem\_M->rtwLogInfo, |
| *187* | 0.0, |
| *188* | rtmGetTFinal(flightControlSystem\_M), |
| *189* | flightControlSystem\_M->Timing.stepSize0, |
| *190* | (&rtmGetErrorStatus(flightControlSystem\_M)), |
| *191* | "optical", |
| *192* | 1, |
| *193* | 0, |
| *194* | 1, |
| *195* | 0.005, |
| *196* | &rt\_ToWksSignalInfo, |
| *197* | rt\_ToWksBlockName); |
| *198* | **if** (localDW->ToWorkspace4\_PWORK.LoggedData == (NULL)) |
| *199* | **return**; |
| *200* | **}** |
| *201* |  |
| *202* | */\* SetupRuntimeResources for ToWorkspace: '<S4>/To Workspace5' \*/* |
| *203* | **{** |
| *204* | **static** int\_T rt\_ToWksWidths[] = **{** 8 **}**; |
| *205* |  |
| *206* | **static** int\_T rt\_ToWksNumDimensions[] = **{** 1 **}**; |
| *207* |  |
| *208* | **static** int\_T rt\_ToWksDimensions[] = **{** 8 **}**; |
| *209* |  |
| *210* | **static** boolean\_T rt\_ToWksIsVarDims[] = **{** 0 **}**; |
| *211* |  |
| *212* | **static** **void** \*rt\_ToWksCurrSigDims[] = **{** (NULL) **}**; |
| *213* |  |
| *214* | **static** int\_T rt\_ToWksCurrSigDimsSize[] = **{** 4 **}**; |
| *215* |  |
| *216* | **static** BuiltInDTypeId rt\_ToWksDataTypeIds[] = **{** SS\_SINGLE **}**; |
| *217* |  |
| *218* | **static** int\_T rt\_ToWksComplexSignals[] = **{** 0 **}**; |
| *219* |  |
| *220* | **static** int\_T rt\_ToWksFrameData[] = **{** 0 **}**; |
| *221* |  |
| *222* | **static** RTWPreprocessingFcnPtr rt\_ToWksLoggingPreprocessingFcnPtrs[] = **{** |
| *223* | (NULL) |
| *224* | **}**; |
| *225* |  |
| *226* | **static** **const** char\_T \*rt\_ToWksLabels[] = **{** "SensorCalibration" **}**; |
| *227* |  |
| *228* | **static** RTWLogSignalInfo rt\_ToWksSignalInfo = **{** |
| *229* | 1, |
| *230* | rt\_ToWksWidths, |
| *231* | rt\_ToWksNumDimensions, |
| *232* | rt\_ToWksDimensions, |
| *233* | rt\_ToWksIsVarDims, |
| *234* | rt\_ToWksCurrSigDims, |
| *235* | rt\_ToWksCurrSigDimsSize, |
| *236* | rt\_ToWksDataTypeIds, |
| *237* | rt\_ToWksComplexSignals, |
| *238* | rt\_ToWksFrameData, |
| *239* | rt\_ToWksLoggingPreprocessingFcnPtrs, |
| *240* |  |
| *241* | **{** rt\_ToWksLabels **}**, |
| *242* | (NULL), |
| *243* | (NULL), |
| *244* | (NULL), |
| *245* |  |
| *246* | **{** (NULL) **}**, |
| *247* |  |
| *248* | **{** (NULL) **}**, |
| *249* | (NULL), |
| *250* | (NULL) |
| *251* | **}**; |
| *252* |  |
| *253* | **static** **const** char\_T rt\_ToWksBlockName[] = |
| *254* | "flightControlSystem/Flight Control System/Logging/To Workspace5"; |
| *255* | localDW->ToWorkspace5\_PWORK.LoggedData = rt\_CreateStructLogVar( |
| *256* | flightControlSystem\_M->rtwLogInfo, |
| *257* | 0.0, |
| *258* | rtmGetTFinal(flightControlSystem\_M), |
| *259* | flightControlSystem\_M->Timing.stepSize0, |
| *260* | (&rtmGetErrorStatus(flightControlSystem\_M)), |
| *261* | "calib", |
| *262* | 1, |
| *263* | 0, |
| *264* | 100, |
| *265* | 0.005, |
| *266* | &rt\_ToWksSignalInfo, |
| *267* | rt\_ToWksBlockName); |
| *268* | **if** (localDW->ToWorkspace5\_PWORK.LoggedData == (NULL)) |
| *269* | **return**; |
| *270* | **}** |
| *271* |  |
| *272* | */\* SetupRuntimeResources for ToWorkspace: '<S4>/To Workspace3' \*/* |
| *273* | **{** |
| *274* | **static** int\_T rt\_ToWksWidths[] = **{** 9 **}**; |
| *275* |  |
| *276* | **static** int\_T rt\_ToWksNumDimensions[] = **{** 1 **}**; |
| *277* |  |
| *278* | **static** int\_T rt\_ToWksDimensions[] = **{** 9 **}**; |
| *279* |  |
| *280* | **static** boolean\_T rt\_ToWksIsVarDims[] = **{** 0 **}**; |
| *281* |  |
| *282* | **static** **void** \*rt\_ToWksCurrSigDims[] = **{** (NULL) **}**; |
| *283* |  |
| *284* | **static** int\_T rt\_ToWksCurrSigDimsSize[] = **{** 4 **}**; |
| *285* |  |
| *286* | **static** BuiltInDTypeId rt\_ToWksDataTypeIds[] = **{** SS\_SINGLE **}**; |
| *287* |  |
| *288* | **static** int\_T rt\_ToWksComplexSignals[] = **{** 0 **}**; |
| *289* |  |
| *290* | **static** int\_T rt\_ToWksFrameData[] = **{** 0 **}**; |
| *291* |  |
| *292* | **static** RTWPreprocessingFcnPtr rt\_ToWksLoggingPreprocessingFcnPtrs[] = **{** |
| *293* | (NULL) |
| *294* | **}**; |
| *295* |  |
| *296* | **static** **const** char\_T \*rt\_ToWksLabels[] = **{** "" **}**; |
| *297* |  |
| *298* | **static** RTWLogSignalInfo rt\_ToWksSignalInfo = **{** |
| *299* | 1, |
| *300* | rt\_ToWksWidths, |
| *301* | rt\_ToWksNumDimensions, |
| *302* | rt\_ToWksDimensions, |
| *303* | rt\_ToWksIsVarDims, |
| *304* | rt\_ToWksCurrSigDims, |
| *305* | rt\_ToWksCurrSigDimsSize, |
| *306* | rt\_ToWksDataTypeIds, |
| *307* | rt\_ToWksComplexSignals, |
| *308* | rt\_ToWksFrameData, |
| *309* | rt\_ToWksLoggingPreprocessingFcnPtrs, |
| *310* |  |
| *311* | **{** rt\_ToWksLabels **}**, |
| *312* | (NULL), |
| *313* | (NULL), |
| *314* | (NULL), |
| *315* |  |
| *316* | **{** (NULL) **}**, |
| *317* |  |
| *318* | **{** (NULL) **}**, |
| *319* | (NULL), |
| *320* | (NULL) |
| *321* | **}**; |
| *322* |  |
| *323* | **static** **const** char\_T rt\_ToWksBlockName[] = |
| *324* | "flightControlSystem/Flight Control System/Logging/To Workspace3"; |
| *325* | localDW->ToWorkspace3\_PWORK.LoggedData = rt\_CreateStructLogVar( |
| *326* | flightControlSystem\_M->rtwLogInfo, |
| *327* | 0.0, |
| *328* | rtmGetTFinal(flightControlSystem\_M), |
| *329* | flightControlSystem\_M->Timing.stepSize0, |
| *330* | (&rtmGetErrorStatus(flightControlSystem\_M)), |
| *331* | "cmd", |
| *332* | 1, |
| *333* | 0, |
| *334* | 1, |
| *335* | 0.005, |
| *336* | &rt\_ToWksSignalInfo, |
| *337* | rt\_ToWksBlockName); |
| *338* | **if** (localDW->ToWorkspace3\_PWORK.LoggedData == (NULL)) |
| *339* | **return**; |
| *340* | **}** |
| *341* |  |
| *342* | */\* SetupRuntimeResources for ToWorkspace: '<S4>/To Workspace2' \*/* |
| *343* | **{** |
| *344* | **static** int\_T rt\_ToWksWidths[] = **{** 12 **}**; |
| *345* |  |
| *346* | **static** int\_T rt\_ToWksNumDimensions[] = **{** 1 **}**; |
| *347* |  |
| *348* | **static** int\_T rt\_ToWksDimensions[] = **{** 12 **}**; |
| *349* |  |
| *350* | **static** boolean\_T rt\_ToWksIsVarDims[] = **{** 0 **}**; |
| *351* |  |
| *352* | **static** **void** \*rt\_ToWksCurrSigDims[] = **{** (NULL) **}**; |
| *353* |  |
| *354* | **static** int\_T rt\_ToWksCurrSigDimsSize[] = **{** 4 **}**; |
| *355* |  |
| *356* | **static** BuiltInDTypeId rt\_ToWksDataTypeIds[] = **{** SS\_SINGLE **}**; |
| *357* |  |
| *358* | **static** int\_T rt\_ToWksComplexSignals[] = **{** 0 **}**; |
| *359* |  |
| *360* | **static** int\_T rt\_ToWksFrameData[] = **{** 0 **}**; |
| *361* |  |
| *362* | **static** RTWPreprocessingFcnPtr rt\_ToWksLoggingPreprocessingFcnPtrs[] = **{** |
| *363* | (NULL) |
| *364* | **}**; |
| *365* |  |
| *366* | **static** **const** char\_T \*rt\_ToWksLabels[] = **{** "" **}**; |
| *367* |  |
| *368* | **static** RTWLogSignalInfo rt\_ToWksSignalInfo = **{** |
| *369* | 1, |
| *370* | rt\_ToWksWidths, |
| *371* | rt\_ToWksNumDimensions, |
| *372* | rt\_ToWksDimensions, |
| *373* | rt\_ToWksIsVarDims, |
| *374* | rt\_ToWksCurrSigDims, |
| *375* | rt\_ToWksCurrSigDimsSize, |
| *376* | rt\_ToWksDataTypeIds, |
| *377* | rt\_ToWksComplexSignals, |
| *378* | rt\_ToWksFrameData, |
| *379* | rt\_ToWksLoggingPreprocessingFcnPtrs, |
| *380* |  |
| *381* | **{** rt\_ToWksLabels **}**, |
| *382* | (NULL), |
| *383* | (NULL), |
| *384* | (NULL), |
| *385* |  |
| *386* | **{** (NULL) **}**, |
| *387* |  |
| *388* | **{** (NULL) **}**, |
| *389* | (NULL), |
| *390* | (NULL) |
| *391* | **}**; |
| *392* |  |
| *393* | **static** **const** char\_T rt\_ToWksBlockName[] = |
| *394* | "flightControlSystem/Flight Control System/Logging/To Workspace2"; |
| *395* | localDW->ToWorkspace2\_PWORK.LoggedData = rt\_CreateStructLogVar( |
| *396* | flightControlSystem\_M->rtwLogInfo, |
| *397* | 0.0, |
| *398* | rtmGetTFinal(flightControlSystem\_M), |
| *399* | flightControlSystem\_M->Timing.stepSize0, |
| *400* | (&rtmGetErrorStatus(flightControlSystem\_M)), |
| *401* | "estim", |
| *402* | 1, |
| *403* | 0, |
| *404* | 1, |
| *405* | 0.005, |
| *406* | &rt\_ToWksSignalInfo, |
| *407* | rt\_ToWksBlockName); |
| *408* | **if** (localDW->ToWorkspace2\_PWORK.LoggedData == (NULL)) |
| *409* | **return**; |
| *410* | **}** |
| *411* |  |
| *412* | */\* SetupRuntimeResources for ToWorkspace: '<S4>/To Workspace' \*/* |
| *413* | **{** |
| *414* | **static** int\_T rt\_ToWksWidths[] = **{** 4 **}**; |
| *415* |  |
| *416* | **static** int\_T rt\_ToWksNumDimensions[] = **{** 1 **}**; |
| *417* |  |
| *418* | **static** int\_T rt\_ToWksDimensions[] = **{** 4 **}**; |
| *419* |  |
| *420* | **static** boolean\_T rt\_ToWksIsVarDims[] = **{** 0 **}**; |
| *421* |  |
| *422* | **static** **void** \*rt\_ToWksCurrSigDims[] = **{** (NULL) **}**; |
| *423* |  |
| *424* | **static** int\_T rt\_ToWksCurrSigDimsSize[] = **{** 4 **}**; |
| *425* |  |
| *426* | **static** BuiltInDTypeId rt\_ToWksDataTypeIds[] = **{** SS\_SINGLE **}**; |
| *427* |  |
| *428* | **static** int\_T rt\_ToWksComplexSignals[] = **{** 0 **}**; |
| *429* |  |
| *430* | **static** int\_T rt\_ToWksFrameData[] = **{** 0 **}**; |
| *431* |  |
| *432* | **static** RTWPreprocessingFcnPtr rt\_ToWksLoggingPreprocessingFcnPtrs[] = **{** |
| *433* | (NULL) |
| *434* | **}**; |
| *435* |  |
| *436* | **static** **const** char\_T \*rt\_ToWksLabels[] = **{** "" **}**; |
| *437* |  |
| *438* | **static** RTWLogSignalInfo rt\_ToWksSignalInfo = **{** |
| *439* | 1, |
| *440* | rt\_ToWksWidths, |
| *441* | rt\_ToWksNumDimensions, |
| *442* | rt\_ToWksDimensions, |
| *443* | rt\_ToWksIsVarDims, |
| *444* | rt\_ToWksCurrSigDims, |
| *445* | rt\_ToWksCurrSigDimsSize, |
| *446* | rt\_ToWksDataTypeIds, |
| *447* | rt\_ToWksComplexSignals, |
| *448* | rt\_ToWksFrameData, |
| *449* | rt\_ToWksLoggingPreprocessingFcnPtrs, |
| *450* |  |
| *451* | **{** rt\_ToWksLabels **}**, |
| *452* | (NULL), |
| *453* | (NULL), |
| *454* | (NULL), |
| *455* |  |
| *456* | **{** (NULL) **}**, |
| *457* |  |
| *458* | **{** (NULL) **}**, |
| *459* | (NULL), |
| *460* | (NULL) |
| *461* | **}**; |
| *462* |  |
| *463* | **static** **const** char\_T rt\_ToWksBlockName[] = |
| *464* | "flightControlSystem/Flight Control System/Logging/To Workspace"; |
| *465* | localDW->ToWorkspace\_PWORK.LoggedData = rt\_CreateStructLogVar( |
| *466* | flightControlSystem\_M->rtwLogInfo, |
| *467* | 0.0, |
| *468* | rtmGetTFinal(flightControlSystem\_M), |
| *469* | flightControlSystem\_M->Timing.stepSize0, |
| *470* | (&rtmGetErrorStatus(flightControlSystem\_M)), |
| *471* | "motor", |
| *472* | 1, |
| *473* | 0, |
| *474* | 1, |
| *475* | 0.005, |
| *476* | &rt\_ToWksSignalInfo, |
| *477* | rt\_ToWksBlockName); |
| *478* | **if** (localDW->ToWorkspace\_PWORK.LoggedData == (NULL)) |
| *479* | **return**; |
| *480* | **}** |
| *481* |  |
| *482* | */\* SetupRuntimeResources for ToWorkspace: '<S4>/To Workspace1' \*/* |
| *483* | **{** |
| *484* | **static** int\_T rt\_ToWksWidths[] = **{** 8 **}**; |
| *485* |  |
| *486* | **static** int\_T rt\_ToWksNumDimensions[] = **{** 1 **}**; |
| *487* |  |
| *488* | **static** int\_T rt\_ToWksDimensions[] = **{** 8 **}**; |
| *489* |  |
| *490* | **static** boolean\_T rt\_ToWksIsVarDims[] = **{** 0 **}**; |
| *491* |  |
| *492* | **static** **void** \*rt\_ToWksCurrSigDims[] = **{** (NULL) **}**; |
| *493* |  |
| *494* | **static** int\_T rt\_ToWksCurrSigDimsSize[] = **{** 4 **}**; |
| *495* |  |
| *496* | **static** BuiltInDTypeId rt\_ToWksDataTypeIds[] = **{** SS\_SINGLE **}**; |
| *497* |  |
| *498* | **static** int\_T rt\_ToWksComplexSignals[] = **{** 0 **}**; |
| *499* |  |
| *500* | **static** int\_T rt\_ToWksFrameData[] = **{** 0 **}**; |
| *501* |  |
| *502* | **static** RTWPreprocessingFcnPtr rt\_ToWksLoggingPreprocessingFcnPtrs[] = **{** |
| *503* | (NULL) |
| *504* | **}**; |
| *505* |  |
| *506* | **static** **const** char\_T \*rt\_ToWksLabels[] = **{** "" **}**; |
| *507* |  |
| *508* | **static** RTWLogSignalInfo rt\_ToWksSignalInfo = **{** |
| *509* | 1, |
| *510* | rt\_ToWksWidths, |
| *511* | rt\_ToWksNumDimensions, |
| *512* | rt\_ToWksDimensions, |
| *513* | rt\_ToWksIsVarDims, |
| *514* | rt\_ToWksCurrSigDims, |
| *515* | rt\_ToWksCurrSigDimsSize, |
| *516* | rt\_ToWksDataTypeIds, |
| *517* | rt\_ToWksComplexSignals, |
| *518* | rt\_ToWksFrameData, |
| *519* | rt\_ToWksLoggingPreprocessingFcnPtrs, |
| *520* |  |
| *521* | **{** rt\_ToWksLabels **}**, |
| *522* | (NULL), |
| *523* | (NULL), |
| *524* | (NULL), |
| *525* |  |
| *526* | **{** (NULL) **}**, |
| *527* |  |
| *528* | **{** (NULL) **}**, |
| *529* | (NULL), |
| *530* | (NULL) |
| *531* | **}**; |
| *532* |  |
| *533* | **static** **const** char\_T rt\_ToWksBlockName[] = |
| *534* | "flightControlSystem/Flight Control System/Logging/To Workspace1"; |
| *535* | localDW->ToWorkspace1\_PWORK.LoggedData = rt\_CreateStructLogVar( |
| *536* | flightControlSystem\_M->rtwLogInfo, |
| *537* | 0.0, |
| *538* | rtmGetTFinal(flightControlSystem\_M), |
| *539* | flightControlSystem\_M->Timing.stepSize0, |
| *540* | (&rtmGetErrorStatus(flightControlSystem\_M)), |
| *541* | "posref", |
| *542* | 1, |
| *543* | 0, |
| *544* | 1, |
| *545* | 0.005, |
| *546* | &rt\_ToWksSignalInfo, |
| *547* | rt\_ToWksBlockName); |
| *548* | **if** (localDW->ToWorkspace1\_PWORK.LoggedData == (NULL)) |
| *549* | **return**; |
| *550* | **}** |
| *551* | **}** |
| *552* |  |
| *553* | */\* Output and update for atomic system: '<S1>/Logging' \*/* |
| *554* | **void** flightControlSystem\_Logging(RT\_MODEL\_flightControlSystem\_T \* **const** |
| *555* | flightControlSystem\_M, **const** real32\_T rtu\_motorCmds[4], **const** real32\_T |
| *556* | rtu\_posRef[8], **const** statesEstim\_t \*rtu\_states\_estim\_Inport\_3, **const** |
| *557* | CommandBus \*rtu\_ReferenceValueServerBus\_Inport\_4, **const** SensorsBus |
| *558* | \*rtu\_Sensors\_Inport\_5, real32\_T rtu\_sensordata\_datin\_Inport\_6, uint32\_T |
| *559* | rtu\_sensordata\_datin\_Inport\_6\_d, real32\_T rtu\_sensordata\_datin\_Inport\_6\_e, |
| *560* | real32\_T rtu\_sensordata\_datin\_Inport\_6\_c, real32\_T |
| *561* | rtu\_sensordata\_datin\_Inport\_6\_cg, real32\_T rtu\_sensordata\_datin\_Inport\_6\_i, |
| *562* | real32\_T rtu\_sensordata\_datin\_Inport\_6\_h, real32\_T |
| *563* | rtu\_sensordata\_datin\_Inport\_6\_b, real32\_T rtu\_sensordata\_datin\_Inport\_6\_dz, |
| *564* | real32\_T rtu\_sensordata\_datin\_Inport\_6\_n, DW\_Logging\_flightControlSystem\_T |
| *565* | \*localDW) |
| *566* | **{** |
| *567* | */\* local block i/o variables \*/* |
| *568* | real32\_T rtb\_DataTypeConversion1\_m; |
| *569* | real32\_T rtb\_DataTypeConversion2; |
| *570* | real32\_T rtb\_TmpSignalConversionAtToWorkspace6Inport1[10]; |
| *571* | real32\_T rtb\_TmpSignalConversionAtToWorkspace4Inport1[8]; |
| *572* | real32\_T rtb\_SigConversion\_InsertedFor\_Sensors\_BusSelector\_BusSelecto\_iq[8]; |
| *573* | real32\_T rtb\_TmpSignalConversionAtToWorkspace3Inport1[9]; |
| *574* | real32\_T rtb\_TmpSignalConversionAtToWorkspace2Inport1[12]; |
| *575* | int32\_T i; |
| *576* |  |
| *577* | */\* DataTypeConversion: '<S4>/Data Type Conversion1' \*/* |
| *578* | rtb\_DataTypeConversion1\_m = rtu\_sensordata\_datin\_Inport\_6; |
| *579* |  |
| *580* | */\* DataTypeConversion: '<S4>/Data Type Conversion2' \*/* |
| *581* | rtb\_DataTypeConversion2 = (real32\_T)rtu\_sensordata\_datin\_Inport\_6\_d; |
| *582* |  |
| *583* | */\* SignalConversion generated from: '<S4>/To Workspace6' \*/* |
| *584* | rtb\_TmpSignalConversionAtToWorkspace6Inport1[0] = |
| *585* | rtu\_sensordata\_datin\_Inport\_6\_e; |
| *586* | rtb\_TmpSignalConversionAtToWorkspace6Inport1[1] = |
| *587* | rtu\_sensordata\_datin\_Inport\_6\_c; |
| *588* | rtb\_TmpSignalConversionAtToWorkspace6Inport1[2] = |
| *589* | rtu\_sensordata\_datin\_Inport\_6\_cg; |
| *590* | rtb\_TmpSignalConversionAtToWorkspace6Inport1[3] = |
| *591* | rtu\_sensordata\_datin\_Inport\_6\_i; |
| *592* | rtb\_TmpSignalConversionAtToWorkspace6Inport1[4] = |
| *593* | rtu\_sensordata\_datin\_Inport\_6\_h; |
| *594* | rtb\_TmpSignalConversionAtToWorkspace6Inport1[5] = |
| *595* | rtu\_sensordata\_datin\_Inport\_6\_b; |
| *596* | rtb\_TmpSignalConversionAtToWorkspace6Inport1[6] = |
| *597* | rtu\_sensordata\_datin\_Inport\_6\_dz; |
| *598* | rtb\_TmpSignalConversionAtToWorkspace6Inport1[7] = |
| *599* | rtu\_sensordata\_datin\_Inport\_6\_n; |
| *600* | rtb\_TmpSignalConversionAtToWorkspace6Inport1[8] = rtb\_DataTypeConversion1\_m; |
| *601* | rtb\_TmpSignalConversionAtToWorkspace6Inport1[9] = rtb\_DataTypeConversion2; |
| *602* |  |
| *603* | */\* ToWorkspace: '<S4>/To Workspace6' \*/* |
| *604* | **{** |
| *605* | **double** locTime = flightControlSystem\_M->Timing.taskTime0; |
| *606* | ; |
| *607* | rt\_UpdateStructLogVar((StructLogVar \*)localDW->ToWorkspace6\_PWORK.LoggedData, |
| *608* | &locTime, |
| *609* | &rtb\_TmpSignalConversionAtToWorkspace6Inport1[0]); |
| *610* | **}** |
| *611* |  |
| *612* | */\* SignalConversion generated from: '<S4>/To Workspace4' incorporates:* |
| *613* | *\* DataTypeConversion: '<S4>/Data Type Conversion11'* |
| *614* | *\* SignalConversion generated from: '<S4>/Sensors\_BusSelector'* |
| *615* | *\*/* |
| *616* | rtb\_TmpSignalConversionAtToWorkspace4Inport1[0] = |
| *617* | rtu\_Sensors\_Inport\_5->VisionSensors.usePosVIS\_flag; |
| *618* | rtb\_TmpSignalConversionAtToWorkspace4Inport1[1] = |
| *619* | rtu\_Sensors\_Inport\_5->VisionSensors.opticalFlow\_data[0]; |
| *620* | rtb\_TmpSignalConversionAtToWorkspace4Inport1[2] = |
| *621* | rtu\_Sensors\_Inport\_5->VisionSensors.opticalFlow\_data[1]; |
| *622* | rtb\_TmpSignalConversionAtToWorkspace4Inport1[3] = |
| *623* | rtu\_Sensors\_Inport\_5->VisionSensors.opticalFlow\_data[2]; |
| *624* | rtb\_TmpSignalConversionAtToWorkspace4Inport1[4] = |
| *625* | rtu\_Sensors\_Inport\_5->VisionSensors.posVIS\_data[0]; |
| *626* | rtb\_TmpSignalConversionAtToWorkspace4Inport1[5] = |
| *627* | rtu\_Sensors\_Inport\_5->VisionSensors.posVIS\_data[1]; |
| *628* | rtb\_TmpSignalConversionAtToWorkspace4Inport1[6] = |
| *629* | rtu\_Sensors\_Inport\_5->VisionSensors.posVIS\_data[2]; |
| *630* | rtb\_TmpSignalConversionAtToWorkspace4Inport1[7] = |
| *631* | rtu\_Sensors\_Inport\_5->VisionSensors.posVIS\_data[3]; |
| *632* |  |
| *633* | */\* ToWorkspace: '<S4>/To Workspace4' \*/* |
| *634* | **{** |
| *635* | **double** locTime = flightControlSystem\_M->Timing.taskTime0; |
| *636* | ; |
| *637* | rt\_UpdateStructLogVar((StructLogVar \*)localDW->ToWorkspace4\_PWORK.LoggedData, |
| *638* | &locTime, |
| *639* | &rtb\_TmpSignalConversionAtToWorkspace4Inport1[0]); |
| *640* | **}** |
| *641* |  |
| *642* | */\* SignalConversion generated from: '<S4>/Sensors\_BusSelector' \*/* |
| *643* | **for** (i = 0; i < 8; i++) **{** |
| *644* | rtb\_SigConversion\_InsertedFor\_Sensors\_BusSelector\_BusSelecto\_iq[i] = |
| *645* | rtu\_Sensors\_Inport\_5->SensorCalibration[i]; |
| *646* | **}** |
| *647* |  |
| *648* | */\* ToWorkspace: '<S4>/To Workspace5' \*/* |
| *649* | **{** |
| *650* | **double** locTime = flightControlSystem\_M->Timing.taskTime0; |
| *651* | ; |
| *652* | rt\_UpdateStructLogVar((StructLogVar \*)localDW->ToWorkspace5\_PWORK.LoggedData, |
| *653* | &locTime, |
| *654* | &rtb\_SigConversion\_InsertedFor\_Sensors\_BusSelector\_BusSelecto\_iq |
| *655* | [0]); |
| *656* | **}** |
| *657* |  |
| *658* | */\* SignalConversion generated from: '<S4>/To Workspace3' incorporates:* |
| *659* | *\* DataTypeConversion: '<S4>/Data Type Conversion3'* |
| *660* | *\* DataTypeConversion: '<S4>/Data Type Conversion7'* |
| *661* | *\* DataTypeConversion: '<S4>/Data Type Conversion9'* |
| *662* | *\* SignalConversion generated from: '<S4>/ReferenceValueServerBus\_BusSelector'* |
| *663* | *\*/* |
| *664* | rtb\_TmpSignalConversionAtToWorkspace3Inport1[0] = |
| *665* | rtu\_ReferenceValueServerBus\_Inport\_4->controlModePosVSOrient; |
| *666* | rtb\_TmpSignalConversionAtToWorkspace3Inport1[1] = |
| *667* | rtu\_ReferenceValueServerBus\_Inport\_4->pos\_ref[0]; |
| *668* | rtb\_TmpSignalConversionAtToWorkspace3Inport1[2] = |
| *669* | rtu\_ReferenceValueServerBus\_Inport\_4->pos\_ref[1]; |
| *670* | rtb\_TmpSignalConversionAtToWorkspace3Inport1[3] = |
| *671* | rtu\_ReferenceValueServerBus\_Inport\_4->pos\_ref[2]; |
| *672* | rtb\_TmpSignalConversionAtToWorkspace3Inport1[4] = |
| *673* | rtu\_ReferenceValueServerBus\_Inport\_4->takeoff\_flag; |
| *674* | rtb\_TmpSignalConversionAtToWorkspace3Inport1[5] = |
| *675* | rtu\_ReferenceValueServerBus\_Inport\_4->orient\_ref[0]; |
| *676* | rtb\_TmpSignalConversionAtToWorkspace3Inport1[6] = |
| *677* | rtu\_ReferenceValueServerBus\_Inport\_4->orient\_ref[1]; |
| *678* | rtb\_TmpSignalConversionAtToWorkspace3Inport1[7] = |
| *679* | rtu\_ReferenceValueServerBus\_Inport\_4->orient\_ref[2]; |
| *680* | rtb\_TmpSignalConversionAtToWorkspace3Inport1[8] = (real32\_T) |
| *681* | rtu\_ReferenceValueServerBus\_Inport\_4->live\_time\_ticks; |
| *682* |  |
| *683* | */\* ToWorkspace: '<S4>/To Workspace3' \*/* |
| *684* | **{** |
| *685* | **double** locTime = flightControlSystem\_M->Timing.taskTime0; |
| *686* | ; |
| *687* | rt\_UpdateStructLogVar((StructLogVar \*)localDW->ToWorkspace3\_PWORK.LoggedData, |
| *688* | &locTime, |
| *689* | &rtb\_TmpSignalConversionAtToWorkspace3Inport1[0]); |
| *690* | **}** |
| *691* |  |
| *692* | */\* SignalConversion generated from: '<S4>/To Workspace2' incorporates:* |
| *693* | *\* SignalConversion generated from: '<S4>/states\_estim\_BusSelector'* |
| *694* | *\*/* |
| *695* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[0] = rtu\_states\_estim\_Inport\_3->X; |
| *696* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[1] = rtu\_states\_estim\_Inport\_3->Y; |
| *697* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[2] = rtu\_states\_estim\_Inport\_3->Z; |
| *698* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[3] = |
| *699* | rtu\_states\_estim\_Inport\_3->yaw; |
| *700* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[4] = |
| *701* | rtu\_states\_estim\_Inport\_3->pitch; |
| *702* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[5] = |
| *703* | rtu\_states\_estim\_Inport\_3->roll; |
| *704* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[6] = |
| *705* | rtu\_states\_estim\_Inport\_3->dx; |
| *706* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[7] = |
| *707* | rtu\_states\_estim\_Inport\_3->dy; |
| *708* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[8] = |
| *709* | rtu\_states\_estim\_Inport\_3->dz; |
| *710* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[9] = rtu\_states\_estim\_Inport\_3->p; |
| *711* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[10] = |
| *712* | rtu\_states\_estim\_Inport\_3->q; |
| *713* | rtb\_TmpSignalConversionAtToWorkspace2Inport1[11] = |
| *714* | rtu\_states\_estim\_Inport\_3->r; |
| *715* |  |
| *716* | */\* ToWorkspace: '<S4>/To Workspace2' \*/* |
| *717* | **{** |
| *718* | **double** locTime = flightControlSystem\_M->Timing.taskTime0; |
| *719* | ; |
| *720* | rt\_UpdateStructLogVar((StructLogVar \*)localDW->ToWorkspace2\_PWORK.LoggedData, |
| *721* | &locTime, |
| *722* | &rtb\_TmpSignalConversionAtToWorkspace2Inport1[0]); |
| *723* | **}** |
| *724* |  |
| *725* | */\* ToWorkspace: '<S4>/To Workspace' \*/* |
| *726* | **{** |
| *727* | **double** locTime = flightControlSystem\_M->Timing.taskTime0; |
| *728* | ; |
| *729* | rt\_UpdateStructLogVar((StructLogVar \*)localDW->ToWorkspace\_PWORK.LoggedData, |
| *730* | &locTime, &rtu\_motorCmds[0]); |
| *731* | **}** |
| *732* |  |
| *733* | */\* ToWorkspace: '<S4>/To Workspace1' \*/* |
| *734* | **{** |
| *735* | **double** locTime = flightControlSystem\_M->Timing.taskTime0; |
| *736* | ; |
| *737* | rt\_UpdateStructLogVar((StructLogVar \*)localDW->ToWorkspace1\_PWORK.LoggedData, |
| *738* | &locTime, &rtu\_posRef[0]); |
| *739* | **}** |
| *740* | **}** |
| *741* |  |
| *742* | */\* System initialize for atomic system: '<Root>/Flight Control System' \*/* |
| *743* | **void** flightControlSystem\_FlightControlSystem\_Init(RT\_MODEL\_flightControlSystem\_T |
| *744* | \* **const** flightControlSystem\_M, DW\_FlightControlSystem\_flightControlSystem\_T |
| *745* | \*localDW) |
| *746* | **{** |
| *747* | */\* InitializeConditions for Memory: '<S5>/Memory' \*/* |
| *748* | localDW->Memory\_PreviousInput = flightControlSystem\_rtZstatesEstim\_t; |
| *749* |  |
| *750* | */\* SystemInitialize for Atomic SubSystem: '<S1>/Logging' \*/* |
| *751* | flightControlSystem\_Logging\_Init(flightControlSystem\_M, &localDW->Logging); |
| *752* |  |
| *753* | */\* End of SystemInitialize for SubSystem: '<S1>/Logging' \*/* |
| *754* |  |
| *755* | */\* SystemInitialize for Merge: '<S3>/Merge' \*/* |
| *756* | flag\_outport = ((uint8\_T)0U); |
| *757* |  |
| *758* | */\* SystemInitialize for ModelReference: '<S1>/controller' \*/* |
| *759* | flightController\_Init(); |
| *760* |  |
| *761* | */\* SystemInitialize for ModelReference: '<S1>/estimator' \*/* |
| *762* | stateEstimator\_Init(); |
| *763* | **}** |
| *764* |  |
| *765* | */\* Output and update for atomic system: '<Root>/Flight Control System' \*/* |
| *766* | **void** flightControlSystem\_FlightControlSystem(RT\_MODEL\_flightControlSystem\_T \* |
| *767* | **const** flightControlSystem\_M, **const** CommandBus \*rtu\_ReferenceValueServerCmds, |
| *768* | **const** SensorsBus \*rtu\_Sensors, real\_T rtu\_VisionbasedData, |
| *769* | DW\_FlightControlSystem\_flightControlSystem\_T \*localDW) |
| *770* | **{** |
| *771* | */\* local block i/o variables \*/* |
| *772* | statesEstim\_t rtb\_estimator; |
| *773* | sensordata\_t rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1; |
| *774* | CommandBus rtb\_Switch\_o; |
| *775* | uint32\_T rtb\_vbat\_percentage; |
| *776* | real32\_T rtb\_x; |
| *777* | real32\_T rtb\_y; |
| *778* | real32\_T rtb\_z; |
| *779* | real32\_T rtb\_x\_c; |
| *780* | real32\_T rtb\_y\_n; |
| *781* | real32\_T rtb\_z\_p; |
| *782* | real32\_T rtb\_altitude; |
| *783* | real32\_T rtb\_pressure; |
| *784* | real32\_T rtb\_vbat\_V; |
| *785* | real32\_T rtb\_controller\_o2[8]; |
| *786* |  |
| *787* | */\* Switch: '<S5>/Switch' incorporates:* |
| *788* | *\* BusCreator: '<S5>/Bus Creator'* |
| *789* | *\* Constant: '<S16>/Constant'* |
| *790* | *\* Constant: '<S5>/Constant'* |
| *791* | *\* Constant: '<S5>/Constant1'* |
| *792* | *\* Constant: '<S5>/Constant2'* |
| *793* | *\* Constant: '<S5>/Constant3'* |
| *794* | *\* Logic: '<S5>/Logical Operator'* |
| *795* | *\* Memory: '<S5>/Memory'* |
| *796* | *\* RelationalOperator: '<S16>/Compare'* |
| *797* | *\* SignalConversion generated from: '<S5>/Bus Creator'* |
| *798* | *\*/* |
| *799* | **if** ((rtu\_VisionbasedData == 211.0) && true) **{** |
| *800* | rtb\_Switch\_o.controlModePosVSOrient = true; |
| *801* | rtb\_Switch\_o.pos\_ref[0] = localDW->Memory\_PreviousInput.X; |
| *802* | rtb\_Switch\_o.pos\_ref[1] = localDW->Memory\_PreviousInput.Y; |
| *803* | rtb\_Switch\_o.pos\_ref[2] = (-0.6F); |
| *804* | rtb\_Switch\_o.takeoff\_flag = rtu\_ReferenceValueServerCmds->takeoff\_flag; |
| *805* | rtb\_Switch\_o.orient\_ref[0] = 0.0F; |
| *806* | rtb\_Switch\_o.orient\_ref[1] = 0.0F; |
| *807* | rtb\_Switch\_o.orient\_ref[2] = 0.0F; |
| *808* | rtb\_Switch\_o.live\_time\_ticks = rtu\_ReferenceValueServerCmds->live\_time\_ticks; |
| *809* | **}** **else** **{** |
| *810* | rtb\_Switch\_o = \*rtu\_ReferenceValueServerCmds; |
| *811* | **}** |
| *812* |  |
| *813* | */\* End of Switch: '<S5>/Switch' \*/* |
| *814* |  |
| *815* | */\* SignalConversion generated from: '<S6>/Bus Selector3' \*/* |
| *816* | rtb\_x = rtu\_Sensors->HALSensors.HAL\_acc\_SI.x; |
| *817* |  |
| *818* | */\* SignalConversion generated from: '<S6>/Bus Selector3' \*/* |
| *819* | rtb\_y = rtu\_Sensors->HALSensors.HAL\_acc\_SI.y; |
| *820* |  |
| *821* | */\* SignalConversion generated from: '<S6>/Bus Selector3' \*/* |
| *822* | rtb\_z = rtu\_Sensors->HALSensors.HAL\_acc\_SI.z; |
| *823* |  |
| *824* | */\* SignalConversion generated from: '<S6>/Bus Selector4' \*/* |
| *825* | rtb\_x\_c = rtu\_Sensors->HALSensors.HAL\_gyro\_SI.x; |
| *826* |  |
| *827* | */\* SignalConversion generated from: '<S6>/Bus Selector4' \*/* |
| *828* | rtb\_y\_n = rtu\_Sensors->HALSensors.HAL\_gyro\_SI.y; |
| *829* |  |
| *830* | */\* SignalConversion generated from: '<S6>/Bus Selector4' \*/* |
| *831* | rtb\_z\_p = rtu\_Sensors->HALSensors.HAL\_gyro\_SI.z; |
| *832* |  |
| *833* | */\* SignalConversion generated from: '<S6>/Bus Selector5' \*/* |
| *834* | rtb\_altitude = rtu\_Sensors->HALSensors.HAL\_ultrasound\_SI.altitude; |
| *835* |  |
| *836* | */\* SignalConversion generated from: '<S6>/Bus Selector6' \*/* |
| *837* | rtb\_pressure = rtu\_Sensors->HALSensors.HAL\_pressure\_SI.pressure; |
| *838* |  |
| *839* | */\* SignalConversion generated from: '<S6>/Bus Selector1' \*/* |
| *840* | rtb\_vbat\_V = rtu\_Sensors->HALSensors.HAL\_vbat\_SI.vbat\_V; |
| *841* |  |
| *842* | */\* SignalConversion generated from: '<S6>/Bus Selector1' \*/* |
| *843* | rtb\_vbat\_percentage = rtu\_Sensors->HALSensors.HAL\_vbat\_SI.vbat\_percentage; |
| *844* |  |
| *845* | */\* BusCreator generated from: '<S1>/estimator' \*/* |
| *846* | rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1.ddx = rtb\_x; |
| *847* | rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1.ddy = rtb\_y; |
| *848* | rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1.ddz = rtb\_z; |
| *849* | rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1.p = rtb\_x\_c; |
| *850* | rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1.q = rtb\_y\_n; |
| *851* | rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1.r = rtb\_z\_p; |
| *852* | rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1.altitude\_sonar |
| *853* | = rtb\_altitude; |
| *854* | rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1.prs = |
| *855* | rtb\_pressure; |
| *856* | rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1.vbat\_V = |
| *857* | rtb\_vbat\_V; |
| *858* | rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1.vbat\_percentage |
| *859* | = rtb\_vbat\_percentage; |
| *860* |  |
| *861* | */\* ModelReference: '<S1>/estimator' \*/* |
| *862* | stateEstimator(&rtb\_Switch\_o.controlModePosVSOrient, |
| *863* | &rtb\_BusConversion\_InsertedFor\_estimator\_at\_inport\_1\_BusCreator1, |
| *864* | &rtu\_Sensors->VisionSensors.usePosVIS\_flag, |
| *865* | &rtu\_Sensors->VisionSensors.opticalFlow\_data[0], |
| *866* | &rtu\_Sensors->VisionSensors.posVIS\_data[0], |
| *867* | &rtu\_Sensors->SensorCalibration[0], &rtb\_estimator); |
| *868* |  |
| *869* | */\* ModelReference: '<S1>/controller' \*/* |
| *870* | flightController(&rtb\_Switch\_o, &rtb\_estimator, &motors\_outport[0], |
| *871* | &rtb\_controller\_o2[0]); |
| *872* |  |
| *873* | */\* Outputs for Atomic SubSystem: '<S1>/Logging' \*/* |
| *874* | flightControlSystem\_Logging(flightControlSystem\_M, motors\_outport, |
| *875* | rtb\_controller\_o2, &rtb\_estimator, rtu\_ReferenceValueServerCmds, rtu\_Sensors, |
| *876* | rtb\_vbat\_V, rtb\_vbat\_percentage, rtb\_x, rtb\_y, rtb\_z, rtb\_x\_c, rtb\_y\_n, |
| *877* | rtb\_z\_p, rtb\_altitude, rtb\_pressure, &localDW->Logging); |
| *878* |  |
| *879* | */\* End of Outputs for SubSystem: '<S1>/Logging' \*/* |
| *880* |  |
| *881* | */\* If: '<S3>/If' incorporates:* |
| *882* | *\* Abs: '<S3>/Abs'* |
| *883* | *\* Abs: '<S3>/Abs1'* |
| *884* | *\* Abs: '<S3>/Abs2'* |
| *885* | *\* Abs: '<S3>/Abs3'* |
| *886* | *\* Abs: '<S3>/Abs4'* |
| *887* | *\* Abs: '<S3>/Abs5'* |
| *888* | *\* Constant: '<S10>/Constant'* |
| *889* | *\* Constant: '<S11>/Constant'* |
| *890* | *\* Constant: '<S12>/Constant'* |
| *891* | *\* Constant: '<S7>/Constant'* |
| *892* | *\* Constant: '<S8>/Constant'* |
| *893* | *\* Constant: '<S9>/Constant'* |
| *894* | *\* Gain: '<S3>/Gain'* |
| *895* | *\* Gain: '<S3>/Gain1'* |
| *896* | *\* Logic: '<S3>/Logical Operator'* |
| *897* | *\* Logic: '<S3>/Logical Operator1'* |
| *898* | *\* Logic: '<S3>/Logical Operator2'* |
| *899* | *\* Logic: '<S3>/Logical Operator3'* |
| *900* | *\* RelationalOperator: '<S10>/Compare'* |
| *901* | *\* RelationalOperator: '<S11>/Compare'* |
| *902* | *\* RelationalOperator: '<S12>/Compare'* |
| *903* | *\* RelationalOperator: '<S7>/Compare'* |
| *904* | *\* RelationalOperator: '<S8>/Compare'* |
| *905* | *\* RelationalOperator: '<S9>/Compare'* |
| *906* | *\* Sum: '<S3>/Subtract'* |
| *907* | *\* Sum: '<S3>/Subtract1'* |
| *908* | *\*/* |
| *909* | **if** (((real32\_T)fabs(rtb\_estimator.X) > 10.0F) || ((real32\_T)fabs |
| *910* | (rtb\_estimator.Y) > 10.0F)) **{** |
| *911* | */\* Outputs for IfAction SubSystem: '<S3>/Geofencing error' incorporates:* |
| *912* | *\* ActionPort: '<S13>/Action Port'* |
| *913* | *\*/* |
| *914* | */\* SignalConversion generated from: '<S13>/Out1' incorporates:* |
| *915* | *\* Constant: '<S13>/Constant'* |
| *916* | *\*/* |
| *917* | flag\_outport = ((uint8\_T)1U); |
| *918* |  |
| *919* | */\* End of Outputs for SubSystem: '<S3>/Geofencing error' \*/* |
| *920* | **}** **else** **if** ((((real32\_T)fabs(rtu\_Sensors->VisionSensors.opticalFlow\_data[0]) > |
| *921* | 0.01F) && ((real32\_T)fabs(1.0F \* |
| *922* | rtu\_Sensors->VisionSensors.opticalFlow\_data[0] - |
| *923* | rtb\_estimator.dx) > 6.0F)) || (((real32\_T)fabs(1.0F \* |
| *924* | rtu\_Sensors->VisionSensors.opticalFlow\_data[1] - |
| *925* | rtb\_estimator.dy) > 6.0F) && ((real32\_T)fabs |
| *926* | (rtu\_Sensors->VisionSensors.opticalFlow\_data[1]) > 0.01F))) **{** |
| *927* | */\* Outputs for IfAction SubSystem: '<S3>/estimator//Optical flow error' incorporates:* |
| *928* | *\* ActionPort: '<S15>/Action Port'* |
| *929* | *\*/* |
| *930* | */\* SignalConversion generated from: '<S15>/Out1' incorporates:* |
| *931* | *\* Constant: '<S15>/Constant'* |
| *932* | *\*/* |
| *933* | flag\_outport = ((uint8\_T)99U); |
| *934* |  |
| *935* | */\* End of Outputs for SubSystem: '<S3>/estimator//Optical flow error' \*/* |
| *936* | **}** **else** **{** |
| *937* | */\* Outputs for IfAction SubSystem: '<S3>/Normal condition' incorporates:* |
| *938* | *\* ActionPort: '<S14>/Action Port'* |
| *939* | *\*/* |
| *940* | */\* SignalConversion generated from: '<S14>/Out1' incorporates:* |
| *941* | *\* Constant: '<S14>/Constant'* |
| *942* | *\*/* |
| *943* | flag\_outport = ((uint8\_T)0U); |
| *944* |  |
| *945* | */\* End of Outputs for SubSystem: '<S3>/Normal condition' \*/* |
| *946* | **}** |
| *947* |  |
| *948* | */\* End of If: '<S3>/If' \*/* |
| *949* |  |
| *950* | */\* Update for Memory: '<S5>/Memory' \*/* |
| *951* | localDW->Memory\_PreviousInput = rtb\_estimator; |
| *952* | **}** |
| *953* |  |
| *954* | */\* Model step function for TID0 \*/* |
| *955* | **void** flightControlSystem\_step0(**void**) */\* Sample time: [0.005s, 0.0s] \*/* |
| *956* | **{** |
| *957* | */\* local block i/o variables \*/* |
| *958* | real\_T rtb\_RateTransition; |
| *959* |  |
| *960* | */\* RateTransition: '<Root>/Rate Transition' \*/* |
| *961* | rtb\_RateTransition = |
| *962* | flightControlSystem\_DW.RateTransition\_Buffer[flightControlSystem\_DW.RateTransition\_ActiveBufIdx]; |
| *963* |  |
| *964* | */\* Outputs for Atomic SubSystem: '<Root>/Flight Control System' \*/* |
| *965* |  |
| *966* | */\* Inport: '<Root>/AC cmd' incorporates:* |
| *967* | *\* Inport: '<Root>/Sensors'* |
| *968* | *\*/* |
| *969* | flightControlSystem\_FlightControlSystem(flightControlSystem\_M, &cmd\_inport, |
| *970* | &sensor\_inport, rtb\_RateTransition, |
| *971* | &flightControlSystem\_DW.FlightControlSystem); |
| *972* |  |
| *973* | */\* End of Outputs for SubSystem: '<Root>/Flight Control System' \*/* |
| *974* |  |
| *975* | */\* Outport: '<Root>/Actuators' \*/* |
| *976* | flightControlSystem\_Y.Actuators[0] = motors\_outport[0]; |
| *977* | flightControlSystem\_Y.Actuators[1] = motors\_outport[1]; |
| *978* | flightControlSystem\_Y.Actuators[2] = motors\_outport[2]; |
| *979* | flightControlSystem\_Y.Actuators[3] = motors\_outport[3]; |
| *980* |  |
| *981* | */\* Outport: '<Root>/Flag' \*/* |
| *982* | flightControlSystem\_Y.Flag = flag\_outport; |
| *983* |  |
| *984* | */\* Matfile logging \*/* |
| *985* | rt\_UpdateTXYLogVars(flightControlSystem\_M->rtwLogInfo, |
| *986* | (&flightControlSystem\_M->Timing.taskTime0)); |
| *987* |  |
| *988* | */\* signal main to stop simulation \*/* |
| *989* | **{** */\* Sample time: [0.005s, 0.0s] \*/* |
| *990* | **if** ((rtmGetTFinal(flightControlSystem\_M)!=-1) && |
| *991* | !((rtmGetTFinal(flightControlSystem\_M)- |
| *992* | flightControlSystem\_M->Timing.taskTime0) > |
| *993* | flightControlSystem\_M->Timing.taskTime0 \* (DBL\_EPSILON))) **{** |
| *994* | rtmSetErrorStatus(flightControlSystem\_M, "Simulation finished"); |
| *995* | **}** |
| *996* | **}** |
| *997* |  |
| *998* | */\* Update absolute time \*/* |
| *999* | */\* The "clockTick0" counts the number of times the code of this task has* |
| *1000* | *\* been executed. The absolute time is the multiplication of "clockTick0"* |
| *1001* | *\* and "Timing.stepSize0". Size of "clockTick0" ensures timer will not* |
| *1002* | *\* overflow during the application lifespan selected.* |
| *1003* | *\* Timer of this task consists of two 32 bit unsigned integers.* |
| *1004* | *\* The two integers represent the low bits Timing.clockTick0 and the high bits* |
| *1005* | *\* Timing.clockTickH0. When the low bit overflows to 0, the high bits increment.* |
| *1006* | *\*/* |
| *1007* | **if** (!(++flightControlSystem\_M->Timing.clockTick0)) **{** |
| *1008* | ++flightControlSystem\_M->Timing.clockTickH0; |
| *1009* | **}** |
| *1010* |  |
| *1011* | flightControlSystem\_M->Timing.taskTime0 = |
| *1012* | flightControlSystem\_M->Timing.clockTick0 \* |
| *1013* | flightControlSystem\_M->Timing.stepSize0 + |
| *1014* | flightControlSystem\_M->Timing.clockTickH0 \* |
| *1015* | flightControlSystem\_M->Timing.stepSize0 \* 4294967296.0; |
| *1016* | **}** |
| *1017* |  |
| *1018* | */\* Model step function for TID1 \*/* |
| *1019* | **void** flightControlSystem\_step1(**void**) */\* Sample time: [0.2s, 0.0s] \*/* |
| *1020* | **{** |
| *1021* | */\* local block i/o variables \*/* |
| *1022* | real32\_T rtb\_BlobAnalysis\_o2[100]; |
| *1023* | int32\_T rtb\_BlobAnalysis\_o1[50]; |
| *1024* | int32\_T rtb\_BlobAnalysis\_o3[200]; |
| *1025* | uint32\_T pos; |
| *1026* | uint32\_T neg; |
| *1027* | uint32\_T yDiff; |
| *1028* | real\_T rtb\_Switch; |
| *1029* | int32\_T pixListNinc; |
| *1030* | uint32\_T BlobAnalysis\_NUM\_PIX\_DW[50]; |
| *1031* | int32\_T j\_pixListNinc; |
| *1032* | int32\_T ns; |
| *1033* | int32\_T ms; |
| *1034* | int32\_T numLoops; |
| *1035* | int32\_T p; |
| *1036* | boolean\_T maxNumBlobsReached; |
| *1037* | uint8\_T currentLabel; |
| *1038* | uint32\_T stackIdx; |
| *1039* | uint32\_T walkerIdx; |
| *1040* | uint32\_T pixIdx; |
| *1041* | uint32\_T padIdx; |
| *1042* | int32\_T idx; |
| *1043* | int32\_T n; |
| *1044* | int32\_T loop; |
| *1045* | real32\_T centroid\_idx\_1; |
| *1046* | real32\_T centroid\_idx\_0; |
| *1047* |  |
| *1048* | */\* Outputs for Atomic SubSystem: '<Root>/Image Processing System' \*/* |
| *1049* | */\* ModelReference: '<S2>/Y1UY2V to YUV' incorporates:* |
| *1050* | *\* Inport: '<Root>/Image Data'* |
| *1051* | *\*/* |
| *1052* | conversionYUV(&flightControlSystem\_U.Y1UY2V[0], |
| *1053* | &flightControlSystem\_B.Y1UY2VtoYUV\_o1[0], |
| *1054* | &flightControlSystem\_B.Y1UY2VtoYUV\_o2[0], |
| *1055* | &flightControlSystem\_B.Y1UY2VtoYUV\_o3[0], |
| *1056* | &(flightControlSystem\_DW.Y1UY2VtoYUV\_InstanceData.rtb)); |
| *1057* |  |
| *1058* | */\* S-Function (svipcolorconv): '<S2>/Color Space Conversion' \*/* |
| *1059* | */\* Precompute constants \*/* |
| *1060* | stackIdx = ((uint16\_T)26149U) \* 128U; |
| *1061* | pixIdx = (((uint16\_T)6419U) \* 128U + ((uint16\_T)13320U) \* 128U) + 8192U; |
| *1062* | padIdx = ((uint16\_T)33050U) \* 128U; |
| *1063* | **for** (loop = 0; loop < 19200; loop++) **{** |
| *1064* | */\* Convert YcbCr to RGB; apply coefficients and offsets \*/* |
| *1065* | */\* derived from the ITU BT.601-5 recommendation; all of the \*/* |
| *1066* | */\* coefficients and offsets are scaled (by 2^14) such that \*/* |
| *1067* | */\* the conversion can be done using integer arithmetic; this \*/* |
| *1068* | */\* routine relies on the user supplying the data in proper \*/* |
| *1069* | */\* ranges: Y [16..235], Cb & Cr [16..240] \*/* |
| *1070* | */\* Note that all of the operations are grouped in such a way \*/* |
| *1071* | */\* that the arithmetic can be done using unsigned integers \*/* |
| *1072* | */\* Given that Y is in the proper range, yDiff should \*/* |
| *1073* | */\* always be positive \*/* |
| *1074* | yDiff = flightControlSystem\_B.Y1UY2VtoYUV\_o1[loop] - 16U; |
| *1075* |  |
| *1076* | */\* Red; 8192 in this equations is 0.5\*2^14 or 2^13; adding 0.5 \*/* |
| *1077* | */\* before truncation will result in rounding \*/* |
| *1078* | pos = (yDiff \* ((uint16\_T)19077U) + (uint32\_T) |
| *1079* | flightControlSystem\_B.Y1UY2VtoYUV\_o3[loop] \* ((uint16\_T)26149U)) + |
| *1080* | 8192U; |
| *1081* | **if** (pos > stackIdx) **{** |
| *1082* | pos -= stackIdx; |
| *1083* | **}** **else** **{** |
| *1084* | pos = 0U; |
| *1085* | **}** |
| *1086* |  |
| *1087* | walkerIdx = pos >> 14; |
| *1088* |  |
| *1089* | */\* limit to avoid wrapping \*/* |
| *1090* | **if** (walkerIdx > 255U) **{** |
| *1091* | walkerIdx = 255U; |
| *1092* | **}** |
| *1093* |  |
| *1094* | */\* Compute green channel \*/* |
| *1095* | pos = yDiff \* ((uint16\_T)19077U) + pixIdx; |
| *1096* | neg = (uint32\_T)flightControlSystem\_B.Y1UY2VtoYUV\_o2[loop] \* ((uint16\_T) |
| *1097* | 6419U) + (uint32\_T)flightControlSystem\_B.Y1UY2VtoYUV\_o3[loop] \* ((uint16\_T) |
| *1098* | 13320U); |
| *1099* |  |
| *1100* | */\* scale back \*/* |
| *1101* | **if** (pos > neg) **{** |
| *1102* | pos -= neg; |
| *1103* | **}** **else** **{** |
| *1104* | pos = 0U; |
| *1105* | **}** |
| *1106* |  |
| *1107* | neg = pos >> 14; |
| *1108* | **if** (neg > 255U) **{** |
| *1109* | neg = 255U; |
| *1110* | **}** |
| *1111* |  |
| *1112* | */\* Compute blue channel \*/* |
| *1113* | pos = (yDiff \* ((uint16\_T)19077U) + (uint32\_T) |
| *1114* | flightControlSystem\_B.Y1UY2VtoYUV\_o2[loop] \* ((uint16\_T)33050U)) + |
| *1115* | 8192U; |
| *1116* |  |
| *1117* | */\* scale back \*/* |
| *1118* | **if** (pos > padIdx) **{** |
| *1119* | pos -= padIdx; |
| *1120* | **}** **else** **{** |
| *1121* | pos = 0U; |
| *1122* | **}** |
| *1123* |  |
| *1124* | yDiff = pos >> 14; |
| *1125* | **if** (yDiff > 255U) **{** |
| *1126* | yDiff = 255U; |
| *1127* | **}** |
| *1128* |  |
| *1129* | */\* assign back the results \*/* |
| *1130* | flightControlSystem\_B.ColorSpaceConversion\_o1[loop] = (uint8\_T)walkerIdx; |
| *1131* | flightControlSystem\_B.ColorSpaceConversion\_o2[loop] = (uint8\_T)neg; |
| *1132* | flightControlSystem\_B.ColorSpaceConversion\_o3[loop] = (uint8\_T)yDiff; |
| *1133* | **}** |
| *1134* |  |
| *1135* | */\* End of S-Function (svipcolorconv): '<S2>/Color Space Conversion' \*/* |
| *1136* |  |
| *1137* | */\* RelationalOperator: '<S2>/Relational Operator1' incorporates:* |
| *1138* | *\* Constant: '<S2>/Constant'* |
| *1139* | *\* Sum: '<S2>/Sum'* |
| *1140* | *\*/* |
| *1141* | **for** (loop = 0; loop < 19200; loop++) **{** |
| *1142* | */\* Sum: '<S2>/Sum' \*/* |
| *1143* | idx = (flightControlSystem\_B.ColorSpaceConversion\_o1[loop] - |
| *1144* | flightControlSystem\_B.ColorSpaceConversion\_o2[loop]) - |
| *1145* | flightControlSystem\_B.ColorSpaceConversion\_o3[loop]; |
| *1146* | **if** (idx < 0) **{** |
| *1147* | idx = 0; |
| *1148* | **}** |
| *1149* |  |
| *1150* | flightControlSystem\_B.RelationalOperator1[loop] = (idx > ((uint8\_T)100U)); |
| *1151* | **}** |
| *1152* |  |
| *1153* | */\* End of RelationalOperator: '<S2>/Relational Operator1' \*/* |
| *1154* |  |
| *1155* | */\* S-Function (svipblob): '<S2>/Blob Analysis' \*/* |
| *1156* | maxNumBlobsReached = false; |
| *1157* | memset(&flightControlSystem\_DW.BlobAnalysis\_PAD\_DW[0], 0, 123U \* **sizeof** |
| *1158* | (uint8\_T)); |
| *1159* | currentLabel = 1U; |
| *1160* | loop = 0; |
| *1161* | idx = 123; |
| *1162* | **for** (n = 0; n < 160; n++) **{** |
| *1163* | **for** (pixListNinc = 0; pixListNinc < 120; pixListNinc++) **{** |
| *1164* | flightControlSystem\_DW.BlobAnalysis\_PAD\_DW[idx] = (uint8\_T) |
| *1165* | (flightControlSystem\_B.RelationalOperator1[loop] ? 255 : 0); |
| *1166* | loop++; |
| *1167* | idx++; |
| *1168* | **}** |
| *1169* |  |
| *1170* | flightControlSystem\_DW.BlobAnalysis\_PAD\_DW[idx] = 0U; |
| *1171* | flightControlSystem\_DW.BlobAnalysis\_PAD\_DW[idx + 1] = 0U; |
| *1172* | idx += 2; |
| *1173* | **}** |
| *1174* |  |
| *1175* | memset(&flightControlSystem\_DW.BlobAnalysis\_PAD\_DW[idx], 0, 121U \* **sizeof** |
| *1176* | (uint8\_T)); |
| *1177* | idx = 0; |
| *1178* | pixIdx = 0U; |
| *1179* | n = 0; |
| *1180* | **while** (n < 160) **{** |
| *1181* | ns = 0; |
| *1182* | ms = (idx + 1) \* 122; |
| *1183* | pixListNinc = 0; |
| *1184* | **while** (pixListNinc < 120) **{** |
| *1185* | padIdx = (uint32\_T)((ms + ns) + 1); |
| *1186* | **if** (flightControlSystem\_DW.BlobAnalysis\_PAD\_DW[padIdx] == 255) **{** |
| *1187* | flightControlSystem\_DW.BlobAnalysis\_PAD\_DW[padIdx] = currentLabel; |
| *1188* | flightControlSystem\_DW.BlobAnalysis\_N\_PIXLIST\_DW[pixIdx] = (int16\_T)idx; |
| *1189* | flightControlSystem\_DW.BlobAnalysis\_M\_PIXLIST\_DW[pixIdx] = (int16\_T)ns; |
| *1190* | pixIdx++; |
| *1191* | numLoops = currentLabel - 1; |
| *1192* | BlobAnalysis\_NUM\_PIX\_DW[numLoops] = 1U; |
| *1193* | flightControlSystem\_DW.BlobAnalysis\_STACK\_DW[0U] = padIdx; |
| *1194* | stackIdx = 1U; |
| *1195* | **while** (stackIdx != 0U) **{** |
| *1196* | stackIdx--; |
| *1197* | padIdx = flightControlSystem\_DW.BlobAnalysis\_STACK\_DW[stackIdx]; |
| *1198* | **for** (loop = 0; loop < 8; loop++) **{** |
| *1199* | walkerIdx = padIdx + rtCP\_BlobAnalysis\_WALKER\_RTP[loop]; |
| *1200* | **if** (flightControlSystem\_DW.BlobAnalysis\_PAD\_DW[walkerIdx] == 255) **{** |
| *1201* | flightControlSystem\_DW.BlobAnalysis\_PAD\_DW[walkerIdx] = |
| *1202* | currentLabel; |
| *1203* | flightControlSystem\_DW.BlobAnalysis\_N\_PIXLIST\_DW[pixIdx] = |
| *1204* | (int16\_T)((int16\_T)(walkerIdx / 122U) - 1); |
| *1205* | flightControlSystem\_DW.BlobAnalysis\_M\_PIXLIST\_DW[pixIdx] = |
| *1206* | (int16\_T)(walkerIdx % 122U - 1U); |
| *1207* | pixIdx++; |
| *1208* | BlobAnalysis\_NUM\_PIX\_DW[numLoops]++; |
| *1209* | flightControlSystem\_DW.BlobAnalysis\_STACK\_DW[stackIdx] = walkerIdx; |
| *1210* | stackIdx++; |
| *1211* | **}** |
| *1212* | **}** |
| *1213* | **}** |
| *1214* |  |
| *1215* | **if** (currentLabel == 50) **{** |
| *1216* | maxNumBlobsReached = true; |
| *1217* | n = 160; |
| *1218* | pixListNinc = 120; |
| *1219* | **}** **else** **{** |
| *1220* | currentLabel++; |
| *1221* | **}** |
| *1222* | **}** |
| *1223* |  |
| *1224* | ns++; |
| *1225* | pixListNinc++; |
| *1226* | **}** |
| *1227* |  |
| *1228* | idx++; |
| *1229* | n++; |
| *1230* | **}** |
| *1231* |  |
| *1232* | stackIdx = (uint32\_T)(maxNumBlobsReached ? (int32\_T)currentLabel : (int32\_T) |
| *1233* | (uint8\_T)(currentLabel - 1U)); |
| *1234* | n = 0; |
| *1235* | pixListNinc = 0; |
| *1236* | **for** (loop = 0; loop < (int32\_T)stackIdx; loop++) **{** |
| *1237* | rtb\_BlobAnalysis\_o1[loop] = (int32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; |
| *1238* | ns = 0; |
| *1239* | ms = 0; |
| *1240* | **if** (BlobAnalysis\_NUM\_PIX\_DW[loop] <= 260000U) **{** |
| *1241* | **for** (idx = 0; idx < (int32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; idx++) **{** |
| *1242* | ns += flightControlSystem\_DW.BlobAnalysis\_N\_PIXLIST\_DW[idx + pixListNinc]; |
| *1243* | ms += flightControlSystem\_DW.BlobAnalysis\_M\_PIXLIST\_DW[idx + n]; |
| *1244* | **}** |
| *1245* |  |
| *1246* | centroid\_idx\_0 = (real32\_T)ms / (real32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; |
| *1247* | centroid\_idx\_1 = (real32\_T)ns / (real32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; |
| *1248* | **}** **else** **{** |
| *1249* | numLoops = (int32\_T)((real32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop] / 260000.0F); |
| *1250* | centroid\_idx\_0 = 0.0F; |
| *1251* | centroid\_idx\_1 = 0.0F; |
| *1252* | **for** (p = 0; p < numLoops; p++) **{** |
| *1253* | ns = 0; |
| *1254* | ms = 0; |
| *1255* | **for** (idx = 0; idx < 260000; idx++) **{** |
| *1256* | ns += flightControlSystem\_DW.BlobAnalysis\_N\_PIXLIST\_DW[(pixListNinc + |
| *1257* | idx) + p \* 260000]; |
| *1258* | ms += flightControlSystem\_DW.BlobAnalysis\_M\_PIXLIST\_DW[(n + idx) + p \* |
| *1259* | 260000]; |
| *1260* | **}** |
| *1261* |  |
| *1262* | centroid\_idx\_0 += (real32\_T)ms / (real32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; |
| *1263* | centroid\_idx\_1 += (real32\_T)ns / (real32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; |
| *1264* | **}** |
| *1265* |  |
| *1266* | ns = 0; |
| *1267* | ms = 0; |
| *1268* | numLoops \*= 260000; |
| *1269* | p = (int32\_T)(BlobAnalysis\_NUM\_PIX\_DW[loop] - numLoops); |
| *1270* | **for** (idx = 0; idx < p; idx++) **{** |
| *1271* | ns += flightControlSystem\_DW.BlobAnalysis\_N\_PIXLIST\_DW[(idx + |
| *1272* | pixListNinc) + numLoops]; |
| *1273* | ms += flightControlSystem\_DW.BlobAnalysis\_M\_PIXLIST\_DW[(idx + n) + |
| *1274* | numLoops]; |
| *1275* | **}** |
| *1276* |  |
| *1277* | centroid\_idx\_0 += (real32\_T)ms / (real32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; |
| *1278* | centroid\_idx\_1 += (real32\_T)ns / (real32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; |
| *1279* | **}** |
| *1280* |  |
| *1281* | rtb\_BlobAnalysis\_o2[loop] = centroid\_idx\_1 + 1.0F; |
| *1282* | pixIdx = loop + 50U; |
| *1283* | rtb\_BlobAnalysis\_o2[pixIdx] = centroid\_idx\_0 + 1.0F; |
| *1284* | ns = 160; |
| *1285* | ms = 120; |
| *1286* | numLoops = 0; |
| *1287* | p = 0; |
| *1288* | **for** (idx = 0; idx < (int32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; idx++) **{** |
| *1289* | j\_pixListNinc = idx + pixListNinc; |
| *1290* | **if** (flightControlSystem\_DW.BlobAnalysis\_N\_PIXLIST\_DW[j\_pixListNinc] < ns) |
| *1291* | **{** |
| *1292* | ns = flightControlSystem\_DW.BlobAnalysis\_N\_PIXLIST\_DW[j\_pixListNinc]; |
| *1293* | **}** |
| *1294* |  |
| *1295* | **if** (flightControlSystem\_DW.BlobAnalysis\_N\_PIXLIST\_DW[j\_pixListNinc] > |
| *1296* | numLoops) **{** |
| *1297* | numLoops = |
| *1298* | flightControlSystem\_DW.BlobAnalysis\_N\_PIXLIST\_DW[j\_pixListNinc]; |
| *1299* | **}** |
| *1300* |  |
| *1301* | j\_pixListNinc = idx + n; |
| *1302* | **if** (flightControlSystem\_DW.BlobAnalysis\_M\_PIXLIST\_DW[j\_pixListNinc] < ms) |
| *1303* | **{** |
| *1304* | ms = flightControlSystem\_DW.BlobAnalysis\_M\_PIXLIST\_DW[j\_pixListNinc]; |
| *1305* | **}** |
| *1306* |  |
| *1307* | **if** (flightControlSystem\_DW.BlobAnalysis\_M\_PIXLIST\_DW[j\_pixListNinc] > p) **{** |
| *1308* | p = flightControlSystem\_DW.BlobAnalysis\_M\_PIXLIST\_DW[j\_pixListNinc]; |
| *1309* | **}** |
| *1310* | **}** |
| *1311* |  |
| *1312* | rtb\_BlobAnalysis\_o3[loop] = ns + 1; |
| *1313* | rtb\_BlobAnalysis\_o3[pixIdx] = ms + 1; |
| *1314* | rtb\_BlobAnalysis\_o3[loop + 100] = (numLoops - ns) + 1; |
| *1315* | rtb\_BlobAnalysis\_o3[loop + 150] = (p - ms) + 1; |
| *1316* | n += (int32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; |
| *1317* | pixListNinc += (int32\_T)BlobAnalysis\_NUM\_PIX\_DW[loop]; |
| *1318* | **}** |
| *1319* |  |
| *1320* | **for** (loop = (int32\_T)stackIdx; loop < 50; loop++) **{** |
| *1321* | rtb\_BlobAnalysis\_o1[loop] = (-1); |
| *1322* | **}** |
| *1323* |  |
| *1324* | **for** (loop = (int32\_T)stackIdx; loop < 50; loop++) **{** |
| *1325* | rtb\_BlobAnalysis\_o2[loop] = (-1.0F); |
| *1326* | **}** |
| *1327* |  |
| *1328* | **for** (loop = (int32\_T)stackIdx + 50; loop < 100; loop++) **{** |
| *1329* | rtb\_BlobAnalysis\_o2[loop] = (-1.0F); |
| *1330* | **}** |
| *1331* |  |
| *1332* | **for** (loop = (int32\_T)stackIdx; loop < 50; loop++) **{** |
| *1333* | rtb\_BlobAnalysis\_o3[loop] = (-1); |
| *1334* | **}** |
| *1335* |  |
| *1336* | **for** (loop = (int32\_T)stackIdx + 50; loop < 100; loop++) **{** |
| *1337* | rtb\_BlobAnalysis\_o3[loop] = (-1); |
| *1338* | **}** |
| *1339* |  |
| *1340* | **for** (loop = (int32\_T)stackIdx + 100; loop < 150; loop++) **{** |
| *1341* | rtb\_BlobAnalysis\_o3[loop] = (-1); |
| *1342* | **}** |
| *1343* |  |
| *1344* | **for** (loop = (int32\_T)stackIdx + 150; loop < 200; loop++) **{** |
| *1345* | rtb\_BlobAnalysis\_o3[loop] = (-1); |
| *1346* | **}** |
| *1347* |  |
| *1348* | */\* Switch: '<S2>/Switch' incorporates:* |
| *1349* | *\* Constant: '<S2>/One'* |
| *1350* | *\* Constant: '<S2>/One1'* |
| *1351* | *\* Constant: '<S2>/One2'* |
| *1352* | *\* RelationalOperator: '<S2>/Relational Operator'* |
| *1353* | *\* S-Function (svipblob): '<S2>/Blob Analysis'* |
| *1354* | *\*/* |
| *1355* | **if** ((int32\_T)stackIdx >= ((uint8\_T)1U)) **{** |
| *1356* | rtb\_Switch = 211.0; |
| *1357* | **}** **else** **{** |
| *1358* | rtb\_Switch = 0.0; |
| *1359* | **}** |
| *1360* |  |
| *1361* | */\* End of Switch: '<S2>/Switch' \*/* |
| *1362* | */\* End of Outputs for SubSystem: '<Root>/Image Processing System' \*/* |
| *1363* |  |
| *1364* | */\* RateTransition: '<Root>/Rate Transition' \*/* |
| *1365* | flightControlSystem\_DW.RateTransition\_Buffer[flightControlSystem\_DW.RateTransition\_ActiveBufIdx |
| *1366* | == 0] = rtb\_Switch; |
| *1367* | flightControlSystem\_DW.RateTransition\_ActiveBufIdx = (int8\_T) |
| *1368* | (flightControlSystem\_DW.RateTransition\_ActiveBufIdx == 0); |
| *1369* | **}** |
| *1370* |  |
| *1371* | */\* Model step wrapper function for compatibility with a static main program \*/* |
| *1372* | **void** flightControlSystem\_step(int\_T tid) |
| *1373* | **{** |
| *1374* | **switch** (tid) **{** |
| *1375* | **case** 0 : |
| *1376* | flightControlSystem\_step0(); |
| *1377* | **break**; |
| *1378* |  |
| *1379* | **case** 1 : |
| *1380* | flightControlSystem\_step1(); |
| *1381* | **break**; |
| *1382* |  |
| *1383* | **default** : |
| *1384* | **break**; |
| *1385* | **}** |
| *1386* | **}** |
| *1387* |  |
| *1388* | */\* Model initialize function \*/* |
| *1389* | **void** flightControlSystem\_initialize(**void**) |
| *1390* | **{** |
| *1391* | */\* Registration code \*/* |
| *1392* |  |
| *1393* | */\* initialize non-finites \*/* |
| *1394* | rt\_InitInfAndNaN(**sizeof**(real\_T)); |
| *1395* |  |
| *1396* | */\* initialize real-time model \*/* |
| *1397* | (**void**) memset((**void** \*)flightControlSystem\_M, 0, |
| *1398* | **sizeof**(RT\_MODEL\_flightControlSystem\_T)); |
| *1399* | (flightControlSystem\_M)->Timing.TaskCounters.cLimit[0] = 1; |
| *1400* | (flightControlSystem\_M)->Timing.TaskCounters.cLimit[1] = 40; |
| *1401* | rtmSetTFinal(flightControlSystem\_M, 30.0); |
| *1402* | flightControlSystem\_M->Timing.stepSize0 = 0.005; |
| *1403* |  |
| *1404* | */\* Setup for data logging \*/* |
| *1405* | **{** |
| *1406* | **static** RTWLogInfo rt\_DataLoggingInfo; |
| *1407* | rt\_DataLoggingInfo.loggingInterval = NULL; |
| *1408* | flightControlSystem\_M->rtwLogInfo = &rt\_DataLoggingInfo; |
| *1409* | **}** |
| *1410* |  |
| *1411* | */\* Setup for data logging \*/* |
| *1412* | **{** |
| *1413* | rtliSetLogXSignalInfo(flightControlSystem\_M->rtwLogInfo, (NULL)); |
| *1414* | rtliSetLogXSignalPtrs(flightControlSystem\_M->rtwLogInfo, (NULL)); |
| *1415* | rtliSetLogT(flightControlSystem\_M->rtwLogInfo, "tout"); |
| *1416* | rtliSetLogX(flightControlSystem\_M->rtwLogInfo, ""); |
| *1417* | rtliSetLogXFinal(flightControlSystem\_M->rtwLogInfo, ""); |
| *1418* | rtliSetLogVarNameModifier(flightControlSystem\_M->rtwLogInfo, "rt\_"); |
| *1419* | rtliSetLogFormat(flightControlSystem\_M->rtwLogInfo, 2); |
| *1420* | rtliSetLogMaxRows(flightControlSystem\_M->rtwLogInfo, 1000); |
| *1421* | rtliSetLogDecimation(flightControlSystem\_M->rtwLogInfo, 1); |
| *1422* |  |
| *1423* | */\** |
| *1424* | *\* Set pointers to the data and signal info for each output* |
| *1425* | *\*/* |
| *1426* | **{** |
| *1427* | **static** **void** \* rt\_LoggedOutputSignalPtrs[] = **{** |
| *1428* | &flightControlSystem\_Y.Actuators[0], |
| *1429* | &flightControlSystem\_Y.Flag |
| *1430* | **}**; |
| *1431* |  |
| *1432* | rtliSetLogYSignalPtrs(flightControlSystem\_M->rtwLogInfo, |
| *1433* | ((LogSignalPtrsType)rt\_LoggedOutputSignalPtrs)); |
| *1434* | **}** |
| *1435* |  |
| *1436* | **{** |
| *1437* | **static** int\_T rt\_LoggedOutputWidths[] = **{** |
| *1438* | 4, |
| *1439* | 1 |
| *1440* | **}**; |
| *1441* |  |
| *1442* | **static** int\_T rt\_LoggedOutputNumDimensions[] = **{** |
| *1443* | 1, |
| *1444* | 1 |
| *1445* | **}**; |
| *1446* |  |
| *1447* | **static** int\_T rt\_LoggedOutputDimensions[] = **{** |
| *1448* | 4, |
| *1449* | 1 |
| *1450* | **}**; |
| *1451* |  |
| *1452* | **static** boolean\_T rt\_LoggedOutputIsVarDims[] = **{** |
| *1453* | 0, |
| *1454* | 0 |
| *1455* | **}**; |
| *1456* |  |
| *1457* | **static** **void**\* rt\_LoggedCurrentSignalDimensions[] = **{** |
| *1458* | (NULL), |
| *1459* | (NULL) |
| *1460* | **}**; |
| *1461* |  |
| *1462* | **static** int\_T rt\_LoggedCurrentSignalDimensionsSize[] = **{** |
| *1463* | 4, |
| *1464* | 4 |
| *1465* | **}**; |
| *1466* |  |
| *1467* | **static** BuiltInDTypeId rt\_LoggedOutputDataTypeIds[] = **{** |
| *1468* | SS\_SINGLE, |
| *1469* | SS\_UINT8 |
| *1470* | **}**; |
| *1471* |  |
| *1472* | **static** int\_T rt\_LoggedOutputComplexSignals[] = **{** |
| *1473* | 0, |
| *1474* | 0 |
| *1475* | **}**; |
| *1476* |  |
| *1477* | **static** RTWPreprocessingFcnPtr rt\_LoggingPreprocessingFcnPtrs[] = **{** |
| *1478* | (NULL), |
| *1479* | (NULL) |
| *1480* | **}**; |
| *1481* |  |
| *1482* | **static** **const** char\_T \*rt\_LoggedOutputLabels[] = **{** |
| *1483* | "motors", |
| *1484* | "flag" **}**; |
| *1485* |  |
| *1486* | **static** **const** char\_T \*rt\_LoggedOutputBlockNames[] = **{** |
| *1487* | "flightControlSystem/Actuators", |
| *1488* | "flightControlSystem/Flag" **}**; |
| *1489* |  |
| *1490* | **static** RTWLogDataTypeConvert rt\_RTWLogDataTypeConvert[] = **{** |
| *1491* | **{** 0, SS\_SINGLE, SS\_SINGLE, 0, 0, 0, 1.0, 0, 0.0 **}**, |
| *1492* |  |
| *1493* | **{** 0, SS\_UINT8, SS\_UINT8, 0, 0, 0, 1.0, 0, 0.0 **}** |
| *1494* | **}**; |
| *1495* |  |
| *1496* | **static** RTWLogSignalInfo rt\_LoggedOutputSignalInfo[] = **{** |
| *1497* | **{** |
| *1498* | 2, |
| *1499* | rt\_LoggedOutputWidths, |
| *1500* | rt\_LoggedOutputNumDimensions, |
| *1501* | rt\_LoggedOutputDimensions, |
| *1502* | rt\_LoggedOutputIsVarDims, |
| *1503* | rt\_LoggedCurrentSignalDimensions, |
| *1504* | rt\_LoggedCurrentSignalDimensionsSize, |
| *1505* | rt\_LoggedOutputDataTypeIds, |
| *1506* | rt\_LoggedOutputComplexSignals, |
| *1507* | (NULL), |
| *1508* | rt\_LoggingPreprocessingFcnPtrs, |
| *1509* |  |
| *1510* | **{** rt\_LoggedOutputLabels **}**, |
| *1511* | (NULL), |
| *1512* | (NULL), |
| *1513* | (NULL), |
| *1514* |  |
| *1515* | **{** rt\_LoggedOutputBlockNames **}**, |
| *1516* |  |
| *1517* | **{** (NULL) **}**, |
| *1518* | (NULL), |
| *1519* | rt\_RTWLogDataTypeConvert |
| *1520* | **}** |
| *1521* | **}**; |
| *1522* |  |
| *1523* | rtliSetLogYSignalInfo(flightControlSystem\_M->rtwLogInfo, |
| *1524* | rt\_LoggedOutputSignalInfo); |
| *1525* |  |
| *1526* | */\* set currSigDims field \*/* |
| *1527* | rt\_LoggedCurrentSignalDimensions[0] = &rt\_LoggedOutputWidths[0]; |
| *1528* | rt\_LoggedCurrentSignalDimensions[1] = &rt\_LoggedOutputWidths[1]; |
| *1529* | **}** |
| *1530* |  |
| *1531* | rtliSetLogY(flightControlSystem\_M->rtwLogInfo, "yout"); |
| *1532* | **}** |
| *1533* |  |
| *1534* | */\* block I/O \*/* |
| *1535* | (**void**) memset(((**void** \*) &flightControlSystem\_B), 0, |
| *1536* | **sizeof**(B\_flightControlSystem\_T)); |
| *1537* |  |
| *1538* | */\* exported global signals \*/* |
| *1539* | motors\_outport[0] = 0.0F; |
| *1540* | motors\_outport[1] = 0.0F; |
| *1541* | motors\_outport[2] = 0.0F; |
| *1542* | motors\_outport[3] = 0.0F; |
| *1543* | flag\_outport = 0U; |
| *1544* |  |
| *1545* | */\* states (dwork) \*/* |
| *1546* | (**void**) memset((**void** \*)&flightControlSystem\_DW, 0, |
| *1547* | **sizeof**(DW\_flightControlSystem\_T)); |
| *1548* |  |
| *1549* | */\* external inputs \*/* |
| *1550* | (**void**)memset(&flightControlSystem\_U, 0, **sizeof**(ExtU\_flightControlSystem\_T)); |
| *1551* | (**void**)memset(&cmd\_inport, 0, **sizeof**(CommandBus)); |
| *1552* | (**void**)memset(&sensor\_inport, 0, **sizeof**(SensorsBus)); |
| *1553* |  |
| *1554* | */\* external outputs \*/* |
| *1555* | (**void**) memset((**void** \*)&flightControlSystem\_Y, 0, |
| *1556* | **sizeof**(ExtY\_flightControlSystem\_T)); |
| *1557* |  |
| *1558* | */\* Model Initialize function for ModelReference Block: '<S1>/controller' \*/* |
| *1559* | flightController\_initialize(rtmGetErrorStatusPointer(flightControlSystem\_M)); |
| *1560* |  |
| *1561* | */\* Model Initialize function for ModelReference Block: '<S1>/estimator' \*/* |
| *1562* | stateEstimator\_initialize(rtmGetErrorStatusPointer(flightControlSystem\_M)); |
| *1563* |  |
| *1564* | */\* Model Initialize function for ModelReference Block: '<S2>/Y1UY2V to YUV' \*/* |
| *1565* | conversionYUV\_initialize(rtmGetErrorStatusPointer(flightControlSystem\_M), |
| *1566* | &(flightControlSystem\_DW.Y1UY2VtoYUV\_InstanceData.rtm)); |
| *1567* |  |
| *1568* | */\* Matfile logging \*/* |
| *1569* | rt\_StartDataLoggingWithStartTime(flightControlSystem\_M->rtwLogInfo, 0.0, |
| *1570* | rtmGetTFinal(flightControlSystem\_M), flightControlSystem\_M->Timing.stepSize0, |
| *1571* | (&rtmGetErrorStatus(flightControlSystem\_M))); |
| *1572* |  |
| *1573* | */\* InitializeConditions for RateTransition: '<Root>/Rate Transition' \*/* |
| *1574* | flightControlSystem\_DW.RateTransition\_Buffer[0] = 0.0; |
| *1575* |  |
| *1576* | */\* SystemInitialize for Atomic SubSystem: '<Root>/Flight Control System' \*/* |
| *1577* |  |
| *1578* | */\* SystemInitialize for Inport: '<Root>/Sensors' \*/* |
| *1579* | flightControlSystem\_FlightControlSystem\_Init(flightControlSystem\_M, |
| *1580* | &flightControlSystem\_DW.FlightControlSystem); |
| *1581* |  |
| *1582* | */\* End of SystemInitialize for SubSystem: '<Root>/Flight Control System' \*/* |
| *1583* | **}** |
| *1584* |  |
| *1585* | */\* Model terminate function \*/* |
| *1586* | **void** flightControlSystem\_terminate(**void**) |
| *1587* | **{** |
| *1588* | */\* (no terminate code required) \*/* |
| *1589* | **}** |
| *1590* |  |

**Header file**

**File:**[**flightControlSystem.h**](matlab:coder.internal.editUrlTextFile('file:///C:/Users/Welcome/MATLAB/Projects/examples/asbQuadcopter23/work/flightControlSystem_grt_rtw/flightControlSystem.h'))

|  |  |
| --- | --- |
| *1* | */\** |
| *2* | *\* flightControlSystem.h* |
| *3* | *\** |
| *4* | *\* Code generation for model "flightControlSystem".* |
| *5* | *\** |
| *6* | *\* Model version : 1.137* |
| *7* | *\* Simulink Coder version : 9.3 (R2020a) 18-Nov-2019* |
| *8* | *\* C source code generated on : Fri Nov 20 20:59:57 2020* |
| *9* | *\** |
| *10* | *\* Target selection: grt.tlc* |
| *11* | *\* Note: GRT includes extra infrastructure and instrumentation for prototyping* |
| *12* | *\* Embedded hardware selection: ARM Compatible->ARM 9* |
| *13* | *\* Code generation objectives: Unspecified* |
| *14* | *\* Validation result: Not run* |
| *15* | *\*/* |
| *16* |  |
| *17* | **#ifndef** RTW\_HEADER\_flightControlSystem\_h\_ |
| *18* | **#define** RTW\_HEADER\_flightControlSystem\_h\_ |
| *19* | **#include <stddef.h>** |
| *20* | **#include <math.h>** |
| *21* | **#include <string.h>** |
| *22* | **#include <float.h>** |
| *23* | **#ifndef** flightControlSystem\_COMMON\_INCLUDES\_ |
| *24* | **# define** flightControlSystem\_COMMON\_INCLUDES\_ |
| *25* | **#include <stdlib.h>** |
| *26* | **#include "rtwtypes.h"** |
| *27* | **#include "rtw\_continuous.h"** |
| *28* | **#include "rtw\_solver.h"** |
| *29* | **#include "rt\_logging.h"** |
| *30* | **#endif** */\* flightControlSystem\_COMMON\_INCLUDES\_ \*/* |
| *31* |  |
| *32* | **#include "flightControlSystem\_types.h"** |
| *33* |  |
| *34* | */\* Shared type includes \*/* |
| *35* | **#include "multiword\_types.h"** |
| *36* |  |
| *37* | */\* Child system includes \*/* |
| *38* | **#include "conversionYUV.h"** |
| *39* | **#define** stateEstimator\_MDLREF\_HIDE\_CHILD\_ |
| *40* | **#include "stateEstimator.h"** |
| *41* | **#define** flightController\_MDLREF\_HIDE\_CHILD\_ |
| *42* | **#include "flightController.h"** |
| *43* | **#include "rt\_nonfinite.h"** |
| *44* |  |
| *45* | */\* Macros for accessing real-time model data structure \*/* |
| *46* | **#ifndef** rtmGetFinalTime |
| *47* | **# define** rtmGetFinalTime(rtm) ((rtm)->Timing.tFinal) |
| *48* | **#endif** |
| *49* |  |
| *50* | **#ifndef** rtmGetRTWLogInfo |
| *51* | **# define** rtmGetRTWLogInfo(rtm) ((rtm)->rtwLogInfo) |
| *52* | **#endif** |
| *53* |  |
| *54* | **#ifndef** rtmCounterLimit |
| *55* | **# define** rtmCounterLimit(rtm, idx) ((rtm)->Timing.TaskCounters.cLimit[(idx)]) |
| *56* | **#endif** |
| *57* |  |
| *58* | **#ifndef** rtmGetErrorStatus |
| *59* | **# define** rtmGetErrorStatus(rtm) ((rtm)->errorStatus) |
| *60* | **#endif** |
| *61* |  |
| *62* | **#ifndef** rtmSetErrorStatus |
| *63* | **# define** rtmSetErrorStatus(rtm, val) ((rtm)->errorStatus = (val)) |
| *64* | **#endif** |
| *65* |  |
| *66* | **#ifndef** rtmGetErrorStatusPointer |
| *67* | **# define** rtmGetErrorStatusPointer(rtm) ((**const** char\_T \*\*)(&((rtm)->errorStatus))) |
| *68* | **#endif** |
| *69* |  |
| *70* | **#ifndef** rtmStepTask |
| *71* | **# define** rtmStepTask(rtm, idx) ((rtm)->Timing.TaskCounters.TID[(idx)] == 0) |
| *72* | **#endif** |
| *73* |  |
| *74* | **#ifndef** rtmGetStopRequested |
| *75* | **# define** rtmGetStopRequested(rtm) ((rtm)->Timing.stopRequestedFlag) |
| *76* | **#endif** |
| *77* |  |
| *78* | **#ifndef** rtmSetStopRequested |
| *79* | **# define** rtmSetStopRequested(rtm, val) ((rtm)->Timing.stopRequestedFlag = (val)) |
| *80* | **#endif** |
| *81* |  |
| *82* | **#ifndef** rtmGetStopRequestedPtr |
| *83* | **# define** rtmGetStopRequestedPtr(rtm) (&((rtm)->Timing.stopRequestedFlag)) |
| *84* | **#endif** |
| *85* |  |
| *86* | **#ifndef** rtmGetT |
| *87* | **# define** rtmGetT(rtm) ((rtm)->Timing.taskTime0) |
| *88* | **#endif** |
| *89* |  |
| *90* | **#ifndef** rtmGetTFinal |
| *91* | **# define** rtmGetTFinal(rtm) ((rtm)->Timing.tFinal) |
| *92* | **#endif** |
| *93* |  |
| *94* | **#ifndef** rtmGetTPtr |
| *95* | **# define** rtmGetTPtr(rtm) (&(rtm)->Timing.taskTime0) |
| *96* | **#endif** |
| *97* |  |
| *98* | **#ifndef** rtmTaskCounter |
| *99* | **# define** rtmTaskCounter(rtm, idx) ((rtm)->Timing.TaskCounters.TID[(idx)]) |
| *100* | **#endif** |
| *101* |  |
| *102* | */\* Block states (default storage) for system '<S1>/Logging' \*/* |
| *103* | **typedef** **struct** **{** |
| *104* | **struct** **{** |
| *105* | **void** \*LoggedData; |
| *106* | **}** ToWorkspace6\_PWORK; */\* '<S4>/To Workspace6' \*/* |
| *107* |  |
| *108* | **struct** **{** |
| *109* | **void** \*LoggedData; |
| *110* | **}** ToWorkspace4\_PWORK; */\* '<S4>/To Workspace4' \*/* |
| *111* |  |
| *112* | **struct** **{** |
| *113* | **void** \*LoggedData; |
| *114* | **}** ToWorkspace5\_PWORK; */\* '<S4>/To Workspace5' \*/* |
| *115* |  |
| *116* | **struct** **{** |
| *117* | **void** \*LoggedData; |
| *118* | **}** ToWorkspace3\_PWORK; */\* '<S4>/To Workspace3' \*/* |
| *119* |  |
| *120* | **struct** **{** |
| *121* | **void** \*LoggedData; |
| *122* | **}** ToWorkspace2\_PWORK; */\* '<S4>/To Workspace2' \*/* |
| *123* |  |
| *124* | **struct** **{** |
| *125* | **void** \*LoggedData; |
| *126* | **}** ToWorkspace\_PWORK; */\* '<S4>/To Workspace' \*/* |
| *127* |  |
| *128* | **struct** **{** |
| *129* | **void** \*LoggedData; |
| *130* | **}** ToWorkspace1\_PWORK; */\* '<S4>/To Workspace1' \*/* |
| *131* | **}** DW\_Logging\_flightControlSystem\_T; |
| *132* |  |
| *133* | */\* Block states (default storage) for system '<Root>/Flight Control System' \*/* |
| *134* | **typedef** **struct** **{** |
| *135* | statesEstim\_t Memory\_PreviousInput; */\* '<S5>/Memory' \*/* |
| *136* | DW\_Logging\_flightControlSystem\_T Logging;*/\* '<S1>/Logging' \*/* |
| *137* | **}** DW\_FlightControlSystem\_flightControlSystem\_T; |
| *138* |  |
| *139* | */\* Block signals (default storage) \*/* |
| *140* | **typedef** **struct** **{** |
| *141* | uint8\_T ColorSpaceConversion\_o1[19200];*/\* '<S2>/Color Space Conversion' \*/* |
| *142* | uint8\_T ColorSpaceConversion\_o2[19200];*/\* '<S2>/Color Space Conversion' \*/* |
| *143* | uint8\_T ColorSpaceConversion\_o3[19200];*/\* '<S2>/Color Space Conversion' \*/* |
| *144* | uint8\_T Y1UY2VtoYUV\_o2[19200]; */\* '<S2>/Y1UY2V to YUV' \*/* |
| *145* | uint8\_T Y1UY2VtoYUV\_o3[19200]; */\* '<S2>/Y1UY2V to YUV' \*/* |
| *146* | uint8\_T Y1UY2VtoYUV\_o1[19200]; */\* '<S2>/Y1UY2V to YUV' \*/* |
| *147* | boolean\_T RelationalOperator1[19200];*/\* '<S2>/Relational Operator1' \*/* |
| *148* | **}** B\_flightControlSystem\_T; |
| *149* |  |
| *150* | */\* Block states (default storage) for system '<Root>' \*/* |
| *151* | **typedef** **struct** **{** |
| *152* | uint32\_T BlobAnalysis\_STACK\_DW[19200];*/\* '<S2>/Blob Analysis' \*/* |
| *153* | int16\_T BlobAnalysis\_M\_PIXLIST\_DW[19200];*/\* '<S2>/Blob Analysis' \*/* |
| *154* | int16\_T BlobAnalysis\_N\_PIXLIST\_DW[19200];*/\* '<S2>/Blob Analysis' \*/* |
| *155* | uint8\_T BlobAnalysis\_PAD\_DW[19764]; */\* '<S2>/Blob Analysis' \*/* |
| *156* | **volatile** real\_T RateTransition\_Buffer[2];*/\* '<Root>/Rate Transition' \*/* |
| *157* | **volatile** int8\_T RateTransition\_ActiveBufIdx;*/\* '<Root>/Rate Transition' \*/* |
| *158* | uint8\_T ColorSpaceConversion\_DWORK1[19200];*/\* '<S2>/Color Space Conversion' \*/* |
| *159* | MdlrefDW\_conversionYUV\_T Y1UY2VtoYUV\_InstanceData;*/\* '<S2>/Y1UY2V to YUV' \*/* |
| *160* | DW\_FlightControlSystem\_flightControlSystem\_T FlightControlSystem;*/\* '<Root>/Flight Control System' \*/* |
| *161* | **}** DW\_flightControlSystem\_T; |
| *162* |  |
| *163* | */\* External inputs (root inport signals with default storage) \*/* |
| *164* | **typedef** **struct** **{** |
| *165* | uint8\_T Y1UY2V[38400]; */\* '<Root>/Image Data' \*/* |
| *166* | **}** ExtU\_flightControlSystem\_T; |
| *167* |  |
| *168* | */\* External outputs (root outports fed by signals with default storage) \*/* |
| *169* | **typedef** **struct** **{** |
| *170* | real32\_T Actuators[4]; */\* '<Root>/Actuators' \*/* |
| *171* | uint8\_T Flag; */\* '<Root>/Flag' \*/* |
| *172* | **}** ExtY\_flightControlSystem\_T; |
| *173* |  |
| *174* | */\* Real-time Model Data Structure \*/* |
| *175* | **struct** tag\_RTM\_flightControlSystem\_T **{** |
| *176* | **const** char\_T \*errorStatus; |
| *177* | RTWLogInfo \*rtwLogInfo; |
| *178* |  |
| *179* | */\** |
| *180* | *\* Timing:* |
| *181* | *\* The following substructure contains information regarding* |
| *182* | *\* the timing information for the model.* |
| *183* | *\*/* |
| *184* | **struct** **{** |
| *185* | time\_T taskTime0; |
| *186* | uint32\_T clockTick0; |
| *187* | uint32\_T clockTickH0; |
| *188* | time\_T stepSize0; |
| *189* | **struct** **{** |
| *190* | uint32\_T TID[2]; |
| *191* | uint32\_T cLimit[2]; |
| *192* | **}** TaskCounters; |
| *193* |  |
| *194* | time\_T tFinal; |
| *195* | boolean\_T stopRequestedFlag; |
| *196* | **}** Timing; |
| *197* | **}**; |
| *198* |  |
| *199* | */\* Block signals (default storage) \*/* |
| *200* | **extern** B\_flightControlSystem\_T flightControlSystem\_B; |
| *201* |  |
| *202* | */\* Block states (default storage) \*/* |
| *203* | **extern** DW\_flightControlSystem\_T flightControlSystem\_DW; |
| *204* |  |
| *205* | */\* External inputs (root inport signals with default storage) \*/* |
| *206* | **extern** ExtU\_flightControlSystem\_T flightControlSystem\_U; |
| *207* |  |
| *208* | */\* External outputs (root outports fed by signals with default storage) \*/* |
| *209* | **extern** ExtY\_flightControlSystem\_T flightControlSystem\_Y; |
| *210* |  |
| *211* | */\* External data declarations for dependent source files \*/* |
| *212* | **extern** **const** statesEstim\_t flightControlSystem\_rtZstatesEstim\_t;*/\* statesEstim\_t ground \*/* |
| *213* |  |
| *214* | */\** |
| *215* | *\* Exported Global Signals* |
| *216* | *\** |
| *217* | *\* Note: Exported global signals are block signals with an exported global* |
| *218* | *\* storage class designation. Code generation will declare the memory for* |
| *219* | *\* these signals and export their symbols.* |
| *220* | *\** |
| *221* | *\*/* |
| *222* | **extern** CommandBus cmd\_inport; */\* '<Root>/AC cmd' \*/* |
| *223* | **extern** SensorsBus sensor\_inport; */\* '<Root>/Sensors' \*/* |
| *224* | **extern** real32\_T motors\_outport[4]; */\* '<S1>/controller' \*/* |
| *225* | **extern** uint8\_T flag\_outport; */\* '<S3>/Merge' \*/* |
| *226* |  |
| *227* | */\* Model entry point functions \*/* |
| *228* | **extern** **void** flightControlSystem\_initialize(**void**); |
| *229* | **extern** **void** flightControlSystem\_step(int\_T tid); |
| *230* | **extern** **void** flightControlSystem\_terminate(**void**); |
| *231* |  |
| *232* | */\* Real-time Model object \*/* |
| *233* | **extern** RT\_MODEL\_flightControlSystem\_T \***const** flightControlSystem\_M; |
| *234* |  |
| *235* | */\*-* |
| *236* | *\* The generated code includes comments that allow you to trace directly* |
| *237* | *\* back to the appropriate location in the model. The basic format* |
| *238* | *\* is <system>/block\_name, where system is the system number (uniquely* |
| *239* | *\* assigned by Simulink) and block\_name is the name of the block.* |
| *240* | *\** |
| *241* | *\* Use the MATLAB hilite\_system command to trace the generated code back* |
| *242* | *\* to the model. For example,* |
| *243* | *\** |
| *244* | *\* hilite\_system('<S3>') - opens system 3* |
| *245* | *\* hilite\_system('<S3>/Kp') - opens and selects block Kp which resides in S3* |
| *246* | *\** |
| *247* | *\* Here is the system hierarchy for this model* |
| *248* | *\** |
| *249* | *\* '<Root>' : 'flightControlSystem'* |
| *250* | *\* '<S1>' : 'flightControlSystem/Flight Control System'* |
| *251* | *\* '<S2>' : 'flightControlSystem/Image Processing System'* |
| *252* | *\* '<S3>' : 'flightControlSystem/Flight Control System/Crash Predictor Flags'* |
| *253* | *\* '<S4>' : 'flightControlSystem/Flight Control System/Logging'* |
| *254* | *\* '<S5>' : 'flightControlSystem/Flight Control System/landing logic'* |
| *255* | *\* '<S6>' : 'flightControlSystem/Flight Control System/sensordata\_group'* |
| *256* | *\* '<S7>' : 'flightControlSystem/Flight Control System/Crash Predictor Flags/Compare To Constant'* |
| *257* | *\* '<S8>' : 'flightControlSystem/Flight Control System/Crash Predictor Flags/Compare To Constant1'* |
| *258* | *\* '<S9>' : 'flightControlSystem/Flight Control System/Crash Predictor Flags/Compare To Constant2'* |
| *259* | *\* '<S10>' : 'flightControlSystem/Flight Control System/Crash Predictor Flags/Compare To Constant3'* |
| *260* | *\* '<S11>' : 'flightControlSystem/Flight Control System/Crash Predictor Flags/Compare To Constant4'* |
| *261* | *\* '<S12>' : 'flightControlSystem/Flight Control System/Crash Predictor Flags/Compare To Constant5'* |
| *262* | *\* '<S13>' : 'flightControlSystem/Flight Control System/Crash Predictor Flags/Geofencing error'* |
| *263* | *\* '<S14>' : 'flightControlSystem/Flight Control System/Crash Predictor Flags/Normal condition'* |
| *264* | *\* '<S15>' : 'flightControlSystem/Flight Control System/Crash Predictor Flags/estimator//Optical flow error'* |
| *265* | *\* '<S16>' : 'flightControlSystem/Flight Control System/landing logic/Compare To Constant'* |
| *266* | *\*/* |
| *267* | **#endif** */\* RTW\_HEADER\_flightControlSystem\_h\_ \*/* |
| *268* |  |