

Flatbuffer

July 20, 2015

1 Network

section *NetworkChannels* **parents** *scj_prelude, MissionId, MissionIds, SchedulableId, SchedulableIds, MissionChan, SchedulableChan, TopLevelMissionSequencerFWChan, FrameworkChan, SafeletChan*

channelset *TerminateSync* ==
 { *schedulables_terminated, schedulables_stopped, get_activeSchedulables* }

channelset *ControlTierSync* ==
 { *start_toplevel_sequencer, done_toplevel_sequencer, done_safeletFW* }

channelset *TierSync* ==
 { *start_mission . FlatBufferMission, done_mission . FlatBufferMission, done_safeletFW, done_toplevel_sequencer* }

channelset *MissionSync* ==
 { *done_safeletFW, done_toplevel_sequencer, register, signalTerminationCall, signalTerminationRet, activate_schedulables, done_schedulable, cleanupSchedulableCall, cleanupSchedulableRet* }

channelset *SchedulablesSync* ==
 { *activate_schedulables, done_safeletFW, done_toplevel_sequencer* }

channelset *ClusterSync* ==
 { *done_toplevel_sequencer, done_safeletFW* }

channelset *AppSync* ==
 { *SafeltAppSync, MissionSequencerAppSync, MissionAppSync, MTAppSync, OSEHSync, APEHSync, getSequencer, end_mission_app, end_managedThread_app, setCeilingPriority, requestTerminationCall, requestTerminationRet, terminationPendingCall, terminationPendingRet, handleAsyncEventCall, handleAsyncEventRet* }

section *Program* **parents** *scj_prelude, MissionId, MissionIds, SchedulableId, SchedulableIds, MissionChan, SchedulableMethChan, MissionFW, SafeletFW, TopLevelMissionSequencerFW, NetworkChannels, ManagedThreadFW, SchedulableMissionSequencerFW, PeriodicEventHandlerFW, OneShotEventHandlerFW, AperiodicEventHandlerFW, FlatBufferApp, FlatBufferMissionSequencerApp, FlatBufferMissionApp, ReaderApp, WriterApp*

process *ControlTier* $\hat{=}$

$$\left(\begin{array}{l} \text{SafeletFW} \\ \llbracket \text{ControlTierSync} \rrbracket \\ \text{TopLevelMissionSequencerFW}(\text{FlatBufferMissionSequencer}) \end{array} \right)$$

process *Tier0* $\hat{=}$

$$\left(\begin{array}{l} \text{MissionFW}(\text{FlatBufferMission}) \\ \llbracket \text{MissionSync} \rrbracket \\ \left(\left(\begin{array}{l} \text{ManagedThreadFW}(\text{Reader}) \\ \llbracket \text{SchedulablesSync} \rrbracket \end{array} \right) \right) \\ \left(\begin{array}{l} \text{ManagedThreadFW}(\text{Writer}) \\ \llbracket \text{SchedulablesSync} \rrbracket \end{array} \right) \end{array} \right)$$

process *Framework* $\hat{=}$

$$\left(\begin{array}{l} \text{ControlTier} \\ \llbracket \text{TierSync} \rrbracket \\ \text{Tier0} \end{array} \right)$$

process *Application* $\hat{=}$

$$\left(\begin{array}{l} \text{FlatBufferApp} \\ ||| \\ \text{FlatBufferMissionSequencerApp} \\ ||| \\ \text{FlatBufferMissionApp} \\ ||| \\ \text{ReaderApp} \\ ||| \\ \text{WriterApp} \end{array} \right)$$

process *Program* $\hat{=}$ *Framework* $\llbracket \text{AppSync} \rrbracket$ *Application*

2 ID Files

2.1 MissionIds

section *MissionIds* **parents** *scj_prelude*, *MissionId*

| *FlatBufferMission* : *MissionID*

distinct (*nullMissionId*, *FlatBufferMission*)

2.2 SchedulablesIds

section *SchedulableIds* **parents** *scj_prelude*, *SchedulableId*

| *FlatBufferMissionSequencer* : *SchedulableID*

| *Reader* : *SchedulableID*

| *Writer* : *SchedulableID*

distinct (*nullSequencerId*, *nullSchedulableId*, *Reader*,
Writer)

3 Safelet

section *FlatBufferApp* **parents** *scj_prelude, SchedulableId, SchedulableIds, SafeletChan*

process *FlatBufferApp* $\hat{=}$ **begin**

InitializeApplication $\hat{=}$
 $\left(\begin{array}{l} \textit{initializeApplicationCall} \longrightarrow \\ \textit{initializeApplicationRet} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

GetSequencer $\hat{=}$
 $\left(\begin{array}{l} \textit{getSequencerCall} \longrightarrow \\ \textit{getSequencerRet} ! \textit{FlatBufferMissionSequencer} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

Methods $\hat{=}$
 $\left(\begin{array}{l} \textit{GetSequencer} \\ \square \\ \textit{InitializeApplication} \end{array} \right); \textit{Methods}$

$\bullet (\textit{Methods}) \triangle (\textit{end_safelet_app} \longrightarrow \mathbf{Skip})$

end

4 Top Level Mission Sequencer

section *FlatBufferMissionSequencerApp* **parents** *TopLevelMissionSequencerChan*,
MissionId, *MissionIds*, *SchedulableId*

process *FlatBufferMissionSequencerApp* $\hat{=}$ **begin**

<i>State</i> <i>returnedMission</i> : <i>boolean</i>

state *State*

<i>Init</i> <i>State'</i>
<i>returnedMission'</i> = <i>false</i>

GetNextMission $\hat{=}$
 $\left(\begin{array}{l} \text{getNextMissionCall} . \text{FlatBufferMissionSequencer} \longrightarrow \\ \text{getNextMissionRet} . \text{FlatBufferMissionSequencer} ! \text{FlatBufferMission} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

getNextMissionMeth $\hat{=}$ **var** *ret* : *MissionId* •
 $\left(\begin{array}{l} \text{getNextMissionCall} . \text{FlatBufferMissionSequencer} \longrightarrow \\ \left(\begin{array}{l} \text{if } (\neg \text{returnedMission} = \mathbf{True}) \longrightarrow \\ \quad \left(\begin{array}{l} \text{returnedMission} := \mathbf{True}; \\ \text{ret} := \text{FlatBufferMission} \end{array} \right) \\ \square \neg (\neg \text{returnedMission} = \mathbf{True}) \longrightarrow \\ \quad (\text{ret} := \mathbf{null}) \end{array} \right) \\ \text{fi} \\ \text{getNextMissionRet} . \text{FlatBufferMissionSequencer} ! \text{ret} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

Methods $\hat{=}$
 $\left(\begin{array}{l} \text{GetNextMission} \\ \square \\ \text{getNextMissionMeth} \end{array} \right) ; \text{Methods}$

• (*Methods*) \triangle (*end_sequencer_app* . *FlatBufferMissionSequencer* \longrightarrow **Skip**)

end

5 Missions

5.1 FlatBufferMission

section *FlatBufferMissionApp* **parents** *scj_prelude, MissionId, MissionIds, SchedulableId, SchedulableIds, MissionChan, SchedulableMethChan*

process *FlatBufferMissionApp* $\hat{=}$ **begin**

<i>State</i> <i>buffer</i> : <i>int</i> <i>writer</i> : <i>Writer</i> <i>reader</i> : <i>Reader</i>
--

state *State*

<i>Init</i> <i>State'</i>
<i>buffer'</i> = 0 <i>writer'</i> = <i>init_placeholder</i> <i>reader'</i> = <i>init_placeholder</i>

InitializePhase $\hat{=}$
 $\left(\begin{array}{l} \textit{initializeCall} . \textit{FlatBufferMission} \longrightarrow \\ \textit{register} ! \textit{Reader} ! \textit{FlatBufferMission} \longrightarrow \\ \textit{register} ! \textit{Writer} ! \textit{FlatBufferMission} \longrightarrow \\ \textit{initializeRet} . \textit{FlatBufferMission} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

CleanupPhase $\hat{=}$
 $\left(\begin{array}{l} \textit{cleanupMissionCall} . \textit{FlatBufferMission} \longrightarrow \\ \textit{cleanupMissionRet} . \textit{FlatBufferMission} ! \mathbf{False} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

bufferEmptyMeth $\hat{=}$ **var** *ret* : *char* •
 $\left(\begin{array}{l} \textit{bufferEmptyCall} . \textit{FlatBufferMission} ? \textit{name} \longrightarrow \\ (\textit{ret} := (\textit{buffer} = 0)) ; \\ \textit{bufferEmptyRet} . \textit{FlatBufferMission} ! \textit{ret} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

writeSyncMeth $\hat{=}$
 $\left(\begin{array}{l} \textit{writeCall} . \textit{FlatBufferMission} ? \textit{thread} ? \textit{update} \longrightarrow \\ \textit{startSyncMeth} . \textit{FlatBufferMission} . \textit{thread} \longrightarrow \\ \textit{lockAcquired} . \textit{FlatBufferMission} . \textit{thread} \longrightarrow \\ \\ \textit{endSyncMeth} . \textit{FlatBufferMission} . \textit{thread} \longrightarrow \\ \textit{writeRet} . \textit{FlatBufferMission} . \square . \textit{thread} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$

$$\text{readSyncMeth} \hat{=} \left(\begin{array}{l} \text{readCall} . \text{FlatBufferMission} ? \text{thread} \longrightarrow \\ \text{startSyncMeth} . \text{FlatBufferMission} . \text{thread} \longrightarrow \\ \text{lockAcquired} . \text{FlatBufferMission} . \text{thread} \longrightarrow \\ \\ \text{endSyncMeth} . \text{FlatBufferMission} . \text{thread} \longrightarrow \\ \text{readRet} . \text{FlatBufferMission} . \text{out} . \text{thread} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

$$\text{waitOnMissionSyncMeth} \hat{=} \left(\begin{array}{l} \text{waitOnMissionCall} . \text{FlatBufferMission} ? \text{thread} ? \text{name} \longrightarrow \\ \text{startSyncMeth} . \text{FlatBufferMission} . \text{thread} \longrightarrow \\ \text{lockAcquired} . \text{FlatBufferMission} . \text{thread} \longrightarrow \\ \\ \text{endSyncMeth} . \text{FlatBufferMission} . \text{thread} \longrightarrow \\ \text{waitOnMissionRet} . \text{FlatBufferMission} . [] . \text{thread} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

$$\text{Methods} \hat{=} \left(\begin{array}{l} \text{InitializePhase} \\ \square \\ \text{CleanupPhase} \\ \square \\ \text{bufferEmptyMeth} \\ \square \\ \text{writeSyncMeth} \\ \square \\ \text{readSyncMeth} \\ \square \\ \text{waitOnMissionSyncMeth} \end{array} \right) ; \text{Methods}$$

- $(\text{Methods}) \triangle (\text{end_mission_app} . \text{FlatBufferMission} \longrightarrow \mathbf{Skip})$

end

5.2 Schedulables of FlatBufferMission

section *ReaderApp* **parents** *ManagedThreadChan*, *SchedulableId*, *SchedulableIds*

process *ReaderApp* $\hat{=}$ **begin**

<i>State</i> <i>fbMission</i> : <i>FlatBufferMission</i>

state *State*

<i>Init</i> <i>State</i> ′ <i>fbMission</i> ′ = <i>fbMission</i>
--

Run $\hat{=}$
$$\left(\begin{array}{l} \text{runCall} . \text{Reader} \longrightarrow \\ \text{runRet} . \text{Reader} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

Methods $\hat{=}$
 $(\text{Run}) ; \text{Methods}$

• $(\text{Methods}) \triangle (\text{end_managedThread_app} . \text{Reader} \longrightarrow \mathbf{Skip})$

end

section *WriterApp* **parents** *ManagedThreadChan, SchedulableId, SchedulableIds*

process *WriterApp* \triangleq **begin**

<i>State</i>
<i>fbMission</i> : <i>FlatBufferMission</i>
<i>i</i> : <i>int</i>

state *State*

<i>Init</i>
<i>State</i> ′
<i>fbMission</i> ′ = <i>fbMission</i>
<i>i</i> ′ = 1

Run \triangleq

$$\left(\begin{array}{l} \text{runCall} . \text{Writer} \longrightarrow \\ \text{runRet} . \text{Writer} \longrightarrow \\ \mathbf{Skip} \end{array} \right)$$

Methods \triangleq
 $(\text{Run}) ; \text{Methods}$

• $(\text{Methods}) \triangle (end_managedThread_app . \text{Writer} \longrightarrow \mathbf{Skip})$

end