Flatbuffer

August 15, 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* ==
$\bigcup$ {*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 $ControlTier \mathrel{\widehat=}$
$$\left(\begin{array}{l} SafeletFW \\ \quad [\![ControlTierSync]\!] \\ TopLevelMissionSequencerFW\,(FlatBufferMissionSequencer) \end{array}\right)$$

process $Tier0 \mathrel{\widehat=}$
$$\left(\begin{array}{l} MissionFW\,(FlatBufferMission) \\ \quad [\![MissionSync]\!] \\ \left(\left(\begin{array}{l} ManagedThreadFW\,(Reader) \\ \quad [\![SchedulablesSync]\!] \\ ManagedThreadFW\,(Writer) \end{array}\right)\right) \\ \quad [\![SchedulablesSync]\!] \end{array}\right)$$

process $Framework \mathrel{\widehat=}$
$$\left(\begin{array}{l} ControlTier \\ \quad [\![TierSync]\!] \\ (\,Tier0\,) \end{array}\right)$$

process $Application \mathrel{\widehat=}$
$$\left(\begin{array}{l} FlatBufferApp \\ ||| \\ FlatBufferMissionSequencerApp \\ ||| \\ FlatBufferMissionApp \\ ||| \\ ReaderApp \\ ||| \\ WriterApp \end{array}\right)$$

process $Program \mathrel{\widehat=} Framework \;[\![\; AppSync \;]\!]\; Application$

2 ID Files

2.1 MissionIds

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

$$FlatBufferMission : MissionID$$

$$distinct\langle nullMissionId, FlatBufferMission\rangle$$

2.2 SchedulablesIds

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

$$FlatBufferMissionSequencer : SchedulableID$$
$$Reader : SchedulableID$$
$$Writer : SchedulableID$$

$$distinct\langle nullSequencerId, nullSchedulableId, Reader, Writer\rangle$$

3 Safelet

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

process $FlatBufferApp \mathrel{\widehat=}$ **begin**

$$InitializeApplication \mathrel{\widehat=} \begin{pmatrix} initializeApplicationCall \longrightarrow \\ initializeApplicationRet \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

$$GetSequencer \mathrel{\widehat=} \begin{pmatrix} getSequencerCall \longrightarrow \\ getSequencerRet\,!\,FlatBufferMissionSequencer \longrightarrow \\ \mathbf{Skip} \end{pmatrix}$$

$$Methods \mathrel{\widehat=} \begin{pmatrix} GetSequencer \\ \Box \\ InitializeApplication \end{pmatrix} ;\ Methods$$

- $(Methods) \triangle (end\_safelet\_app \longrightarrow \mathbf{Skip})$

end

4 Top Level Mission Sequencer

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

process $FlatBufferMissionSequencerApp \mathrel{\widehat=} \textbf{begin}$

State

$this : \textbf{ref}\ FlatBufferMissionSequencerClass$

state *State*

Init

$State'$

$this' = \textbf{new}\ FlatBufferMissionSequencerClass()$

$GetNextMission \mathrel{\widehat=} \textbf{var}\ ret : MissionID \bullet$
$\left(\begin{array}{l} getNextMissionCall\ .\ FlatBufferMissionSequencer \longrightarrow \\ ret := this\ .\ getNextMission(); \\ getNextMissionRet\ .\ FlatBufferMissionSequencer\ !\ ret \longrightarrow \\ \textbf{Skip} \end{array}\right)$

$Methods \mathrel{\widehat=}$
$\big(GetNextMission\big)\ ;\ Methods$

$\bullet\ (Init\ ;\ Methods) \triangle (end\_sequencer\_app\ .\ FlatBufferMissionSequencer \longrightarrow \textbf{Skip})$

end

class *FlatBufferMissionSequencerClass* $\widehat{=}$ **begin**

state *State*

$$returnedMission : boolean$$

state *State*

initial *Init*

$$State'$$

$$returnedMission' = false$$

protected *getNextMission* $\widehat{=}$ **var** *ret* : *MissionID* $\bullet$

$$\left(\begin{array}{l} \textbf{if } (\neg\ returnedMission = \textbf{True}) \longrightarrow \\ \quad \left(\begin{array}{l} returnedMission := \textbf{True}\,; \\ ret := FlatBufferMission \end{array}\right) \\ [\!]\ \neg\ (\neg\ returnedMission = \textbf{True}) \longrightarrow \\ \quad \big(ret := nullMissionId\big) \\ \textbf{fi} \end{array}\right)$$

$\bullet$ **Skip**

end

5 Missions

5.1 FlatBufferMission

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

process $FlatBufferMissionApp \mathrel{\widehat=}$ **begin**

State

$this : \mathbf{ref}\ FlatBufferMissionClass$

state *State*

Init

$State'$

$this' = \mathbf{new}\ FlatBufferMissionClass()$

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

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

$writeSyncMeth \mathrel{\widehat=}$
$\left(\begin{array}{l} writeCall\,.\,FlatBufferMission\,?\,thread \longrightarrow \\ startSyncMeth\,.\,FlatBufferMission\,.\,thread \longrightarrow \\ lockAcquired\,.\,FlatBufferMission\,.\,thread \longrightarrow \\ \left(\begin{array}{l} buffer := update\,; \\ notify\,.\,FlatBufferMissionObject\,?\,thread \longrightarrow \\ \mathbf{Skip} \end{array}\right); \\ endSyncMeth\,.\,FlatBufferMission\,.\,thread \longrightarrow \\ writeRet\,.\,FlatBufferMission\,!\,ret\,!\,thread \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$readSyncMeth \mathrel{\widehat=} \mathbf{var}\ ret : int \bullet$
$\left(\begin{array}{l} readCall\,.\,FlatBufferMission\,?\,thread \longrightarrow \\ startSyncMeth\,.\,FlatBufferMission\,.\,thread \longrightarrow \\ lockAcquired\,.\,FlatBufferMission\,.\,thread \longrightarrow \\ \left(\begin{array}{l} \mathbf{var}\ out : int \bullet out := buffer\,; \\ buffer := 0\,; \\ notify\,.\,FlatBufferMissionObject\,?\,thread \longrightarrow \\ \mathbf{Skip}; \\ ret := out \end{array}\right); \\ endSyncMeth\,.\,FlatBufferMission\,.\,thread \longrightarrow \\ readRet\,.\,FlatBufferMission\,!\,ret\,!\,thread \longrightarrow \\ \mathbf{Skip} \end{array}\right)$

$$
\begin{array}{l}
waitOnMissionSyncMeth \mathrel{\widehat{=}} \\
\left(\begin{array}{l}
waitOnMissionCall\,.\,FlatBufferMission\,?\,thread \longrightarrow \\
startSyncMeth\,.\,FlatBufferMission\,.\,thread \longrightarrow \\
lockAcquired\,.\,FlatBufferMission\,.\,thread \longrightarrow \\
\quad \left(\begin{array}{l} waitCall\,.\,FlatBufferMissionObject\,?\,thread \longrightarrow \\ waitRet\,.\,FlatBufferMissionObject\,?\,thread \longrightarrow \\ \textbf{Skip} \end{array}\right); \\
endSyncMeth\,.\,FlatBufferMission\,.\,thread \longrightarrow \\
waitOnMissionRet\,.\,FlatBufferMission\,!\,ret\,!\,thread \longrightarrow \\
\textbf{Skip}
\end{array}\right)
\end{array}
$$

$$
Methods \mathrel{\widehat{=}} \left(\begin{array}{l} InitializePhase \\ \Box \\ CleanupPhase \\ \Box \\ writeSyncMeth \\ \Box \\ readSyncMeth \\ \Box \\ waitOnMissionSyncMeth \end{array}\right); \; Methods
$$

$\bullet$ $(Init \; ; \; Methods) \triangle (end\_mission\_app\,.\,FlatBufferMission \longrightarrow \textbf{Skip})$

end

class *FlatBufferMissionClass* $\widehat{=}$ **begin**

state *State*

$buffer : int$

state *State*

initial *Init*

$State'$

$buffer' = 0$

public *bufferEmpty* $\widehat{=}$ **var** $ret : \mathbb{B}$ •

$$\begin{pmatrix} \textbf{if } (buffer = 0) \longrightarrow \\ \quad ret := \textbf{True} \\ [] \neg (buffer = 0) \longrightarrow \\ \quad ret := \textbf{False} \\ \textbf{fi} \end{pmatrix}$$

• **Skip**

end

section *FlatBufferMissionMethChan* **parents** *scj_prelude*, *GlobalTypes*

channel *writeCall*
channel *writeRet*

channel *readCall*
channel *readRet* : *int*

channel *waitOnMissionCall*
channel *waitOnMissionRet*

5.2 Schedulables of FlatBufferMission

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

process $ReaderApp \mathrel{\widehat=}$ **begin**

$$Run \mathrel{\widehat=} \begin{pmatrix} runCall\,.\,Reader \longrightarrow \\ this\,.\,run(); \\ runRet\,.\,Reader \longrightarrow \\ \textbf{Skip} \end{pmatrix}$$

$$runMeth \mathrel{\widehat=} \left(\begin{array}{l} runCall\,.\,Reader \longrightarrow \\ \left(\begin{array}{l} \textbf{var}\ terminationPending : boolean \bullet terminationPending := fbMission.terminationPending() ; \\ \left(\begin{array}{l} \mu X \bullet \\ \left(\begin{array}{l} \textbf{if}\ (\neg\ terminationPending = \textbf{True}) \longrightarrow \\ \quad \left(\left(\left(\left(\begin{array}{l} \textbf{var}\ bufferEmpty : boolean \bullet bufferEmpty := fbMission.bufferEmpty() ; \\ \left(\begin{array}{l} \mu X \bullet \\ \left(\begin{array}{l} \textbf{if}\ bufferEmpty = \textbf{True} \longrightarrow \\ \quad \left(\left(\begin{array}{l} waitOnMissionCall\,.\,fbMission\,.\,Reader\,! \longrightarrow \\ waitOnMissionRet\,.\,fbMission\,.\,Reader \longrightarrow \\ \textbf{Skip} \end{array}\right)\right) ; X \\ [\!]\ \neg\ bufferEmpty = \textbf{True} \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right) \end{array}\right) ; \\ readCall\,.\,fbMission\,.\,Reader \longrightarrow \\ readRet\,.\,fbMission\,.\,Reader\,?\,value \longrightarrow \\ \textbf{Skip}; \end{array}\right)\right)\right)\right) ; X \\ [\!]\ \neg\ (\neg\ terminationPending = \textbf{True}) \longrightarrow \textbf{Skip} \\ \textbf{fi} \end{array}\right) \end{array}\right) ; \\ runRet\,.\,Reader \longrightarrow \\ \textbf{Skip} \end{array}\right)$$

$$Methods \mathrel{\widehat=} \begin{pmatrix} Run \\ \Box \\ runMeth \end{pmatrix} ;\ Methods$$

$\bullet\ (Methods) \triangle (end\_managedThread\_app\,.\,Reader \longrightarrow \textbf{Skip})$

end

section *ReaderMethChan* **parents** *scj_prelude*, *GlobalTypes*

channel *runCall*
channel *runRet*

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

process *WriterApp* $\widehat{=}$ **begin**

State

$i : int$

state *State*

Init

$State'$

$i' = 1$

$$Run \mathrel{\widehat{=}} \left(\begin{array}{l} runCall\,.\,Writer \longrightarrow \\ this\,.\,run(); \\ runRet\,.\,Writer \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$runMeth \mathrel{\widehat{=}} \left(\begin{array}{l} runCall\,.\,Writer \longrightarrow \\ \left(\begin{array}{l} \mathbf{var}\ terminationPending : boolean \bullet terminationPending := fbMission.terminationPending() ; \\ \left(\begin{array}{l} \mu X \bullet \\ \left(\begin{array}{l} \mathbf{if}\ (\neg\ terminationPending = \mathbf{True}) \longrightarrow \\ \left(\begin{array}{l} \left(\begin{array}{l} \mathbf{var}\ bufferEmpty : boolean \bullet bufferEmpty := fbMission.bufferEmpty() ; \\ \left(\begin{array}{l} \mu X \bullet \\ \left(\begin{array}{l} \mathbf{if}\ bufferEmpty = \mathbf{True} \longrightarrow \\ \left(\left(\begin{array}{l} waitOnMissionCall\,.\,fbMission\,.\,Writer\,! \longrightarrow \\ waitOnMissionRet\,.\,fbMission\,.\,Writer \longrightarrow \\ \mathbf{Skip} \end{array}\right)\right) ; X \\ [] \neg\ bufferEmpty = \mathbf{True} \longrightarrow \mathbf{Skip} \\ \mathbf{fi} \end{array}\right) \end{array}\right) ; \\ writeCall\,.\,fbMission\,.\,Writer\,!\,i \longrightarrow \\ writeRet\,.\,fbMission\,.\,Writer \longrightarrow \\ \mathbf{Skip}; \\ i := i + 1 \end{array}\right) \\ [] \neg\ (\neg\ terminationPending = \mathbf{True}) \longrightarrow \mathbf{Skip} \\ \mathbf{fi} \end{array}\right) ; X \end{array}\right) ; \\ runRet\,.\,Writer \longrightarrow \\ \mathbf{Skip} \end{array}\right)$$

$$Methods \mathrel{\widehat{=}} \left(\begin{array}{l} Run \\ \Box \\ runMeth \end{array}\right) ;\ Methods$$

$$\bullet\ (Init\ ;\ Methods) \triangle (end\_managedThread\_app\,.\,Writer \longrightarrow \mathbf{Skip})$$

end

section *WriterMethChan* **parents** *scj_prelude*, *GlobalTypes*

channel *runCall*
channel *runRet*