**FiringAtGoalHSM States Pseudo Code**

**Key:**

/\* comments \*/

// place holder for code

Code that can be ignored for lowest level states (no internal SM)

**Start FiringAtGoalHSMS (CurrentEvent)**

// Handle history or irregular entry here

If not history entry

Current state = CheckingForGoal

End

/\* Call ENTRY function for FiringAtGoalHSMS \*/

Run FiringAtGoalHSMS(CurrentEvent)

End Start

**Run FiringAtGoalHSMS (CurrentEvent)**

Init MakeTransition, NextState, EntryEventKind, ReturnEvent

/\* React differently based on the state you’re in \*/

Switch (CurrentState)

Case CheckingForGoal

/\* Execute During to respond to event \*/

CurrentEvent = DuringCheckingForGoalHSMS(CurrentEvent)

/\* Handle all events relevant to CheckingForGoal \*/

If ES\_NO\_EVENT does not == CurrentEvent

Switch (CurrentEvent)

Case EV\_GOAL\_DETECTED or EV\_OPPONENT\_DETECTED

// Any transition action here

NextState = WaitingToShoot

MakeTransition = True EntryEventKind = EV\_ENTRY

ReturnEvent = CurrentEvent

Break

End switch

End if

Break

Case WaitingToShoot

/\* Execute During to respond to event \*/

CurrentEvent = DuringWaitingToShootHSMS(CurrentEvent)

/\* Handle all events relevant to WaitingToShoot \*/

If ES\_NO\_EVENT does not == CurrentEvent

Switch (CurrentEvent)

Case EV\_ROUND\_2\_START // Any transition action here

NextState = Shooting

MakeTransition = True EntryEventKind = EV\_ENTRY

ReturnEvent = CurrentEvent

Break

End switch

End if

Break

Case Shooting

/\* Execute During to respond to event \*/

CurrentEvent = DuringShootingHSMS(CurrentEvent)

Break

End switch

/\* Handle transitions between states \*/

If (MakeTransition)

/\* Exit this state \*/

CurrentEvent.EventType = ES\_EXIT

Run FiringAtGoalHSMS(CurrentEvent)

/\* Enter next state \*/

CurrentState = NextState; // Change to next state

CurrentEvent.EventType = EventEntryKind

Run FiringAtGoalHSMS(CurrentEvent)

End if

Return ReturnEvent

End Run

**QueryFiringAtGoalHSMS**

return CurrentState

End query

**During CheckingForGoalHSMS (currentEvent)**

Init returnEvent

If (EV\_ENTRY = currentEvent)

// check for beacon

/\* Start lower level SM(s) \*/

// StartLowerLevelSM(currentEvent)

Else if (EV\_EXIt = currentEvent)

/\* Exit lower level SM(s) \*/

// RunLowerLevelSM(currentEvent)

// Execute Exit actions here

Else

/\* Pass event to lower level SM(s) \*/

// ReturnEvent = RunLowerLevelSM(currentEvent)

// Execute any internal event response here

End if

return ReturnEvent // or currentEvent to avoid event consumption

End During

**During WaitingToShootHSMS (currentEvent)**

Init returnEvent

If (EV\_ENTRY = currentEvent)

// do nothing

/\* Start lower level SM(s) \*/

// StartLowerLevelSM(currentEvent)

Else if (EV\_EXIt = currentEvent)

/\* Exit lower level SM(s) \*/

// RunLowerLevelSM(currentEvent)

// do nothing

Else

/\* Pass event to lower level SM(s) \*/

// ReturnEvent = RunLowerLevelSM(currentEvent)

// Execute any internal event response here

End if

return ReturnEvent // or currentEvent to avoid event consumption

End During

**During ShootingHSMS (currentEvent)**

Init returnEvent

If (EV\_ENTRY = currentEvent)

// Execute Entry actions here

Turn on ball launcher fan

/\* Start lower level SM(s) \*/

// StartShootingHSMS(currentEvent)

Else if (EV\_EXIt = currentEvent)

/\* Exit lower level SM(s)2 \*/

// RunShootingHSMS(currentEvent)

// Execute Exit actions here

Turn off ball launcher fan

Else

/\* Pass event to lower level SM(s) \*/

// ReturnEvent = RunLowerLevelSM(currentEvent)

// Execute any internal event response here

End if

return ReturnEvent // or currentEvent to avoid event consumption

End During