JoustingSM Pseudocode

```
RunJoustingSM( )
{
    Check for state transition
    Set NextState to CurrentState
    default to normal entry to new state
    assume we are not consuming event

if define PRINT_JOUSTING_SM_CALLS

    print RunJoustingSM

 end if

    switch  CurrentState

        If current state is state one

            Execute During function for state one. ES_ENTRY & ES_EXIT are
            processed here allow the lowere level state machines to re-map
            or consume the event
           Set CurrentEvent to DuringLance_Up(CurrentEvent)
           process any events

if define PRINT_JOUSTING_SM_STATES

           print Lance_Up

end if

           If an event is active

                switch EventType


                   If event is Timeout

                      If EventParameter is LANCE_TIMER


                            set cleared_to_lance_flag as 1

                   break


                   If event is LowerLance command


                      if  cleared_to_lance_flag is 1

                            Set NextState as Lance_Down

                            if strategy is set to "drive by" mode

                                  Close loading gate and open firing gate
                                Update Lance Servo to  LANCE_SERVO_DOWN_POSITION

                           Set lance down timer
```

```
                    Mark that we are taking a transition
                    Set EntryEventKind.EventType as ES_ENTRY
                    Set ReturnEvent.EventType as ES_NO_EVENT;
                   Set cleared_to_lance_flag as 0, denoting that we are NOT clear to lance
            Break
    Break
    If current state is lance down
        Set CurrentEvent as DuringLance_Down(CurrentEvent)
If define PRINT_JOUSTING_SM_STATES
        Print Lance_Down
End if
        If an event is active
            switch EventType
              If event is Timeout
                if  EventParam is LANCE_TIMER
                    Set NextState as Lance_Up
                    Update Lance Servo to LANCE_SERVO_UP_POSITION)
                    Set lance up timer
                    Set ReturnEvent.EventType as ES_NO_EVENT                    }
                else
                   Set NextState as Lance_Down
                   mark that we are taking a transition
                    Set EntryEventKind.EventType as ES_ENTRY
                    Set ReturnEvent as CurrentEvent
If define PRINT_JOUSTING_SM_STATES
                    Print Timer parameter error
End if
            break;
            If event is Raise Lance command
```

```
                    Set NextState as Lance_Up

                    Update Lance Servo to LANCE_SERVO_UP_POSITION
                    Set MakeTransition as true
                    Set EntryEventKind.EventType as ES_ENTRY
                    Set ReturnEvent.EventType as ES_NO_EVENT

                break

        break

    if we are making a transition

        Set CurrentEvent.EventType as ES_EXIT
         Call RunJoustingSM(CurrentEvent)

        Set CurrentState as NextState

         Call RunJoustingSM(EntryEventKind)

    Set CurrentState as NextState

    Return ReturnEvent




StartJoustingSM ( )

    Set LocalEvent as CurrentEvent

If define PRINT_JOUSTING_SM_CALLS

    Print StartJoustingSM

End if

    if there is no entry history

        Set CurrentState as Lance_Up
        Set cleared_to_lance_flag as 1, denoting that our robot is again cleared to lance

    Ensure that ball servo gates are initialized to proper position:
   Close firing gate and open loading gate
    Update Lance Servo to LANCE_SERVO_UP_POSITION

    Start spinning the pitching wheel


If def JOUSTING_TEST_HARNESS
   init lance servo up position

    Configure Timer 1, Channel 5 to provide a polling interval to check for
   new input commands during servo calibration

  Configure Channel 5 as output compare
```

```
    Configure Channel 5 to leave pin disconnected
    Initialize output compare register
    Clear interrupt flags
    Enable interrupts for channel 5
   Enable Timer
    Initialize Port AD0 to be an analog input port
    Initialize last_Pot_Value to present value of Port AD0
End if
  Call RunJoustingSM(CurrentEvent)


StopJoustingSM ( )
Set LocalEvent as CurrentEvent
If def PRINT_JOUSTING_SM_CALLS
   Print StopJoustingSM
End if
   Ensure that ball servo gates are returned to proper position:
  Close firing gate and open loading gate
   Update Lance Servo to LANCE_SERVO_UP_POSITION
   Set cleared_to_lance_flag as 1, denoting that our robot can lance again
   Call RunJoustingSM(CurrentEvent)


QueryJoustingSM ( )
 If def PRINT_JOUSTING_SM_CALLS
   Print QueryJoustingSM
End if
   return(CurrentState)


DuringLance_Up( )
Set ReturnEvent as Event
If define PRINT_JOUSTING_SM_CALLS
```

```
    Print DuringLanceUp
 End if

   process ES_ENTRY, ES_ENTRY_HISTORY & ES_EXIT events
    if  Event.EventType is ES_ENTRY) or Event.EventType is ES_ENTRY_HISTORY

    else if Event.EventType is ES_EXIT

    else

    return(ReturnEvent)



DuringLance_Down( )

Set ReturnEvent as Event

If def PRINT_JOUSTING_SM_CALLS

    Print DuringLanceDown

End if

    if  EventType is ES_ENTRY or EventType is ES_ENTRY_HISTORY

    else if  EventType is ES_EXIT

    else

    return(ReturnEvent)



 if define JOUSTING_TEST_HARNESS

poll pushbutton input for fire ball command
PollLanceButton( )

    clear lance button flag

    Enable Interrupts

   Set Button_Value as current value of port E, bit 1

    Print button value

    if Button_Value is 0

            Set Button_State as 0

    else

            switch Button_State
```

```
            if Button_State is 0:
                Button_State = 1;
            break;
            If Button_State is 1:
                Button_State = 2;
                Set ThisEvent EventType as LowerLance
                Call PostJoustingHSM(ThisEvent)
                Print Drop Lance!
            break
            If Button_State is 2:
                do nothing now that button is debounced
            break;

    Set current_pot_input as current value of testing potentiometer pin
    Reset button check timer
End if
```