```
// Module Variables
// static uint16_t motorPeriod = 25000; // 20 ms
// static uint8_t group = 0; // Channels 0 and 1
// static uint8_t MyPriority;
// static uint8_t HealthBar = 0;
// static Game_States_t CurrentState = Inactivity;


// bool InitGameService(uint8_t Priority)
      // Initialize all hardware
// end



// bool PostGameService(ES_Event_t ThisEvent)
      // Post to this service
// end



// bool Check4GameStart(void)
      // Set CurrentSwitchState to state read from port pin
      // if there is a change of state
            // if the change of state is a falling edge
            // post game start event
      // endif
// end



// ES Event t RunGameService(ES Event t ThisEvent)
      // return ES_NO_EVENT if no errors

      // switch CurrentState
      // case Inactivity
            // if GAME_START event
            // Initialize Health LEDs to all off
            // Turn off water LEDs
            // Turn off weather LEDs
            // Post to needle service so it moves to leftmost position
            // Post to WaterService to start
            // Move to Watering State
            // end if

            // If game is not started, stay in welcome mode
            // LEDs will flash at time intervals of WELCOME_TIMER

            // if WELCOME_TIMEOUT
                  // Every other time flash LEDs on
                  // Start Welcome Timer
                  // Self-transition so stay in current state
            // end if
      // end case

      // case Watering
            // if WATER_TIMEOUT
                  // Post to LightService to start
                  // Move to Lighting State
            // end if
      // end case

      // case Lighting
            // if LIGHT_TIMEOUT
                  // Turn on Harvest Button LED
```

```
                  // Start the ButtonWait Timer
                  // Move to Harvesting State
            // end if
      // end case

      // case Harvesting
            // if BUTTON_WAIT_TIMEOUT
                  // Init Harvest Timer
                  // Decrease Health -5
                  // Switch off Harvest LED
                  // Bad potato display if health too low (< 6)
                        // Turn on bad potato motor
                  // Good potato display if health good (>= 6)
                        // Turn on good potato motor
            // end if

            // if BUTTON_PRESSED
                  // Add 1 to Health Bar
                  // Init Harvest timer
                  // Switch off Harvest LED
                  // Bad potato display if health too low (< 6)
                        // Turn on bad potato motor
                  // Good potato display if health good (>= 6)
                        // Turn on good potato motor
            // end if
            // Move to PotatoDisplay State
      // end case

      // case Celebrating
            // If CELEB_TIMEOUT
                  // Post to Needle Service so it moves to rightmost position
                  // Displace TOT
                  // write high on PD0 = Turn solenoid on
                  // Init Welcome Timer
            // end if

            // if WELCOME_TIMEOUT
                  // write low on PD0 = Turn solenoid off
                  // Move to Inactivity state
            // end if
      // end case

      // set CurrentState to NextState
// end function


// uint8_t GetHealth(void)
      // return HealthBar
// end


// void AddHealth(void)
      // Add 1 to health if less than 8
      // else keep health at 8

      // post to LEDWriteService with correct health LED command
// end


// void SubtractHealth(void)
      // Subtract 1 from health if greater than 0
      // else keep health at 0

      // post to LEDWriteService with correct health LED command
```

```
// end


// PRIVATE FUNCTIONS

// static void HarvestAndSwitchInit(void)
      // Initialize ports A, B, D, E
      // Initialize PWM and ADMulti
      // THIS IS DONE IN MAIN
// end


// static void VibeMotorInit(void)
      // Initialize pin for vibration motor (PB3)
      // Set as output
// end


// static void HarvestButtonLEDOn(void)
      // set harvest button LED pin (PA3) high
// end


// static void HarvestButtonLEDOff(void)
      // set harvest button LED pin (PA3) low
// end


// static void DisplayMotorOn(uint8_t potato_channel)
      // Set the period and pulse width for potato display
// end
```