

Pseudo-code for the JoystickXY module (a service that reads analog input form joystick)

Data private to the module: MyPriority, LastAD_Value[NUM_AD_CHANNELS]

#defines:

AD_CHANNEL (1<<11) | (1<<9) // Using Analong Port 11 and 9

NUM_AD_CHANNELS 2 // Using 2 Analog ports

InitJoystickXY

Takes a priority number, returns True.

Call ADC_ConfigAutoScan(AD_CHANNEL) to configure ADC_Scan

Post Event ES_Init to Joystick queue (this service)

End of InitJoystickXY

RunJoystickXY (implements the service that reads joystick xy values)

EventType field of ThisEvent will be one of: ES_INIT, ES_TIMEOUT

Returns ES_NO_EVENT

Based on the event received, choose one of the following blocks of code:

EventType of ThisEvent is ES_INIT

Setup a 20ms timer to scan AD value every 20ms

End ES_INIT block

EventType of ThisEvent is ES_TIMEOUT

Setup a 20ms timer to scan AD value every 20ms

Read AD values and store the values into LastAD_Value in this direction

ReadJOYSTICK_LR = LastAD_Value[1]; // Load the AD value in LR direction

If joystick has moved by user in LR direction

restart the IDLE timer

Set NewEvent.EventParam = ReadJOYSTICK_LR;

Post ES_JOYSTICK_LR to PostList01

End if

ReadJOYSTICK_LR = LastAD_Value[0]; // Load the AD value in UD direction

If joystick has moved by user in UD direction

restart the IDLE timer

Set NewEvent.EventParam = ReadJOYSTICK_UD;

Post ES_JOYSTICK_LR to PostList01

End if

End ES_TIMEOUT

Return ES_NO_EVENT

End of RunTargetLEDs

PostTargetLEDs & QueryTargetLEDs are not changed from the template

LEDtoDecoderInputs

Takes the current LED index, returns nothing

Apply the decoder inputs to light the current LED passed as input

if the The passed input index has its 0 bit high

Set the decoder 0 bit to high

else

Set the decoder 0 bit to low

if the The passed input index has its 1 bit high

Set the decoder 1 bit to high

else

Set the decoder 1 bit to low

if the The passed input index has its 2 bit high

```
    Set the decoder 2 bit to high
else
    Set the decoder 2 bit to low
End
```

TurnOnNextTargetLED

Takes nothing, returns nothing
Chooses which LED to be lit according to a predefined
sequences of LEDs apply this to the decoder inputs

```
if Counter1 is less than the number of Target LEDs
    Set LedtoBeLit according to the predefined sequence array and
    the current indices (Counter1 and Counter2)
    Increment counter 1
else
    Reset Counter1 to zero
Call LEDtoDecoderInputs(LedtoBeLit)
End
```

Spotlight_Positioning_Check (event checker)

Takes nothing, returns bool
See if the current servo positions match with the current target
coordinate and post ES_SPOT_REACHED or ES_ALL_SPOT_REACHED accordingly

Data private to this function: ReturnValue, EVENT_TO_POST

Initialize ReturnValue to false

Update the ServoLRPosition and ServoUDPosition from the values updated by PWM module

if LedtoBeLit is non zero (one LED is to be on)

```
if ServoLRPosition and ServoUDPosition carry values within bounds
    predefined inside ServosTargetAngles for the different LED positions
    Set ReturnValue to true
    if the lit LED count is still less than the number of LEDs available
        Post ES_SPOT_REACHED to EdMovSensModule and to SpotlightModule
    else
        Post ES_ALL_SPOTS_REACHED to EdMovSensModule and to SpotlightModule
    Increment Counter2 to utilize a new sequence of LEDs if the player chooses to replay
return ReturnValue
End
```