Batman Code

DefaultState (.c)

Module level variables – int basket, int timerInterval, uint8_t MyPriority, DefaultState CurrentState

InitDefaultSM
Takes a uint8_t and returns a Boolean
Set MyPriority to inputted parameter
Set pins 4, 5, and 6 on Port M to outputs
Set rest of the pins to inputs
Initialize basket to 0
Initialize timerInterval to 1
Initialize Servo to pin 7 (T7) by using ADS12_Init
Initialize all the other pins on AD to outputs by using ADS12_Init
Initialize currentState to InitDefault
Light LEDs 1, 2, and 3
Check if post ES_Init to DefaultSM returns true
    Return true if no error

RunDefaultSM
Takes an ES_Event and returns an ES_Event

Check if ThisEvent.EventType is ES_TIMEOUT and ThisEvent.EventParam is 1
    Turn off LED (timerInterval) and increment timerInterval by 1
    if timerInterval is greater than 3
        Set timerInterval back to 1
        Post Failure to DefaultSM
        Send message 6
    else
        Set timer 1 to 15000 ms and start timer 1

If EventType is Basket 1, set basket = 1
If EventType is Basket0, set basket = 0
If EventType is Failure
    Turn off all LEDs (OffLED)
    Set timerInterval to 1
    Stop timer 1
    Set nextState = Lockdown

switch(CurrentState)
    case InitDefault:
        if EventType is ES_Init
            Set nextState to Lockdown

```
        case Lockdown:
                if EventType is BATID_FOUND
                        Light LEDs 1, 2, and 3
                        Set timer 1 to 15s and start timer 1
                        Set nextState to WaitingForCode
                        Send Message1
        case WaitingForCode:
                if EventType is CodeSuccess
                        if basket is 1, light LED 4
                        if basket is 0, light LED 5
                        Set nextState to WaitingForIR
        case WaitingForIR:
                if EventType is IRDetected and EventParam is equal to basket
                        Turn off LED 4 and 5
                        send Message3
                        set nextState to WaitingForBall
                if EventType is IRDetected and EventParam is not equal to basket
                        send Message6
                        send Failure to DefaultSM
                        set timerInterval to 1
                        stop timer 1
        case WaitingForBall:
                if EventType is BallSuccess
                        set nextState to WaitingForAnalog1
        case WaitingForAnalog1:
                if EventType is AnalogCode and EventParam is 3
                        set nextState to WaitingForAnalog2
        case WaitingForAnalog2:
                if EventType is AnalogCode and EventParam is 1
                        set nextState to WaitingForAnalog3
        case WaitingForAnalog3:
                if EventType is AnalogCode and EventParam is 4
                        Light LEDs 1, 2, and 3
                        Send Message 4
                        Set timerInterval to 1 and stop timer 1
                        Call Celebration from Outputs.c
                        Set basket to 0
                        set nextState to Lockdown

set CurrentState to nextState
return ES_NO_EVENT
```

Outputs (.c)

Module level variables – uint8_t MyPriority

InitOutputs
Takes a uint8_t and returns a Boolean
Set MyPriority to inputted parameter
Set timer 6 to 500ms and start timer 6
Post ES_Init to OutputsSM
If successful, return true. Else return false.

RunOutputsSM
Takes an ES_Event and returns an ES_Event
If EventType is ES_TIMEOUT and EventParam is 6,
    Turn off all LEDs

PulseLED
Takes an integer (int i) and returns nothing
Set pin i to high
Set timer 6 to 500ms
Start timer 6

LightLED
Takes an integer (int i) and returns nothing
Set pin i to high (Port AD)

OffLED
Takes an integer (int i) and returns nothing
Set pin i to low (Port AD)

OffAll
Takes no parameters and returns nothing
Set all output pins to low (Port AD)

Celebration
Takes no parameters and returns nothing
Loop 5 times:
    Light LED 4 and 5
    Wait 700 ms
    Turn off LED 4 and 5
    Wait 300 ms

Wait (Blocking code only used for when no inputs should do anything)
Takes an integer (int ms) and returns nothing
Set time1 to current ES_Timer_GetTime
While (ES_Timer_GetTIme <= time1 + ms)

Event Checkers (.c)

checkMessage
Static variables – lastInputState
Local variables - currentInputState
Takes no arguments, returns a boolean

Input currentInputState by checking M pins.

If (currentInputState is not equal to lastInputState)

```
switch(currentInputState)

        case Message1:
                Post CodeSuccess to DefaultSM

        case Message2:
                Post BallSuccess to DefaultSM

        case Message3:
                Post Basket1 to DefaultSM

        case Message4:
                Post Basket0 to DefaultSM

        case Message6:
                Post Failure to DefaultSM
```

Set lastInputState to currentInputState.
return false

checkMorseEvents
Static variables – lastInputState
Local variables - currentInputState
Takes no arguments, returns a boolean
Input currentInputState by checking BatID pin (T0)

If (currentInputState is not equal to lastInputState)

```
        If(currentInputState is high)
                Post RisingEdge to MorseElementsSM and DecodeMorseSM

        If(currentInputState is low)
                Post FallingEdge to MorseElementsSM and DecodeMorseSM
```

Set lastInputState to currentInputState
return false

checkIREvents
Static variables – lastIR1State, lastIR2State
Local variables – currentIR1State, currentIR2State
Takes no arguments, returns a Boolean

Input CurrentIR1State and CurrentIR2State by checking IR1 and IR2 pins (T2 & T3)

If (currentIR1State is not equal to lastIR1State)
    Post IRDetected with EventParam = 0 to DefaultSM

If (currentIR2State is not equal to lastIR2State)
    Post IRDetected with EventParam = 1 to DefaultSM

checkAnalogEvents
Takes no arguments, returns a Boolean

Read analog pin (T7), put value in x

If(x is between 100-150)
    Post AnalogCode with EventParam = 1 to DefaultSM

If(x is between 350-400)
    Post AnalogCode with EventParam = 2 to DefaultSM

If(x is between 600-650)
    Post AnalogCode with EventParam = 3 to DefaultSM

If(x is between 850-900)
    Post AnalogCode with EventParam = 4 to DefaultSM

MorseElements and DecodeMorse

Taken from Ed's pseudo code given in lab 4
Changes listed here –

TestCalibration in MorseElements
if(**(100*FirstDelta/SecondDelta) <= 37 && (100*FirstDelta/SecondDelta) >= 27**)

and

else if**((100*FirstDelta/SecondDelta) >= 250 && (100*FirstDelta/SecondDelta) <= 400**)

DecodeMorse in DecodeMorse
Completely changed to
if(MorseString is "-." using strcmp)

Post BATID_FOUND to DefaultSM

MessageSender (.c)

sendMessage
Takes an integer (int i) and does not return anything

switch(i)

case 1: Raise PTM bits to Message 1 (0x1) and then add a very short delay
case 2: Raise PTM bits to Message 2 (0x2) and then add a very short delay
case 3: Raise PTM bits to Message 3 (0x4) and then add a very short delay
case 4: Raise PTM bits to Message 4 (0x3) and then add a very short delay
case 5: Raise PTM bits to Message 5 (0x5) and then add a very short delay
case 6: Raise PTM bits to Message 6 (0x6) and then add a very short delay
case 7: Raise PTM bits to Message 7 (0x7) and then add a very short delay

Robin Code

Event Checkers (.c)

checkMessage
Static variables – lastInputState
Local variables - currentInputState
Takes no arguments, returns a Boolean

Set MyPriority to inputted parameter
Input currentInputState by checking M pins.

If (currentInputState is not equal to lastInputState)

switch(currentInputState)

case Message1:
Post BATID_FOUND to DefaultSM and LEDCodeSM

case Message2:
Post StartButton to DefaultSM

case Message3:
Post IRDetected to DefaultSM

case Message4:
Post AnalogCodeSuccess to DefaultSM

case Message5:
Post Buzzer to OutputsSM

case Message6:

Post Failure to DefaultSM and OutputsSM

Set lastInputState to currentInputState.
return false

checkButton1Events
Static variables – lastButton1State
Local variables – currentButton1State
Takes no arguments, returns a boolean

Input currentButton1State by checking Button1 pin (T0)

If (QueryButton1SM returns READY2SAMPLE)
    Post START_DEBOUNCE to ButtonFSM with EventParam = 0
    If(CurrentButton1State is not equal to LastButton1State)
        If(CurrentButton1State is not equal to 0)
            Post ButtonDown to DefaultSM and LEDCodeSM with EventParam = 1

Set lastButton1State to currentButton1State
return false

checkButton2Events
Static variables – lastButton2State
Local variables – currentButton2State
Takes no arguments, returns a boolean

Input currentButton2State by checking Button2 pin (T1)

If (QueryButton2SM returns READY2SAMPLE)
    Post START_DEBOUNCE to ButtonFSM with EventParam = 1
    If(CurrentButton2State is not equal to LastButton2State)
        If(CurrentButton2State is not equal to 0)
            Post ButtonDown to DefaultSM and LEDCodeSM with EventParam = 2

Set lastButton2State to currentButton2State
return false

checkButton3Events
Static variables – lastButton3State
Local variables – currentButton3State
Takes no arguments, returns a boolean

Input currentButton3State by checking Button3 pin (T2)

If (QueryButton1SM returns READY2SAMPLE)
    Post START_DEBOUNCE to ButtonFSM with EventParam = 2

If(CurrentButton3State is not equal to LastButton3State)
If(CurrentButton3State is not equal to 0)
Post ButtonDown to DefaultSM and LEDCodeSM with EventParam = 3

Set lastButton3State to currentButton3State
return false

checkSwitch1Events
Static variables – lastSwitch1State
Local variables – currentSwitch1State
Takes no arguments, returns a boolean

Input current Switch1State by checking Switch1 pin (T3)

If (Query Switch1SM returns READY2SAMPLE)
Post START_DEBOUNCE to ButtonFSM with EventParam = 3
If(CurrentSwitch1State is not equal to LastSwitch1State)
If(CurrentSwitch1State is not equal to 0)
Post BallSuccess to DefaultSM with EventParam = 0

Set lastSwitch1State to currentSwitch1State
return false

checkSwitch2Events
Static variables – lastSwitch2State
Local variables – currentSwitch2State
Takes no arguments, returns a boolean

Input currentSwitch2State by checking Switch2 pin (T4)

If (QuerySwitch2SM returns READY2SAMPLE)
Post START_DEBOUNCE to ButtonFSM with EventParam =4
If(CurrentSwitch2State is not equal to LastSwitch2State)
If(CurrentSwitch2State is not equal to 0)
Post BallSuccess to DefaultSM with EventParam = 1

Set lastSwitch2State to currentSwitch2State
return false

checkLeverEvents
Static variables – lastLeverState
Local variables – currentLeverState
Takes no arguments, returns a boolean

Input currentLeverState by checking Lever pin (T5)

If (currentLeverState is not equal to 0)
    Post LeverDown to DefaultSM

If (QueryLeverSM returns READY2SAMPLE)
    Post START_DEBOUNCE to ButtonFSM with EventParam = 5
    If(CurrentLeverState is not equal to LastLeverState)
        If(CurrentLeverState is not equal to 0)
            Post LeverDown to DefaultSM and LEDCodeSM with EventParam = 0

Set lastLeverState to currentLeverState
return false

Outputs (.c)

Module level variables – uint8_t MyPriority
InitOutputs
Takes a uint8_t and returns a Boolean
Set MyPriority to inputted parameter
Set timer 6 to 500ms and start timer 6
Post ES_Init to OutputsSM
If successful, return true. Else return false.

RunOutputsSM
Takes an ES_Event and returns an ES_Event
if EventType is Failure
    PulseBuzzer for 2000ms
if EventType is ES_TIMEOUT and EventParam is 6,
    Turn off buzzer (AD7)

PulseBuzzer
Takes an integer (int i) and returns nothing
Set pin 7 to high (AD7)
Set timer 6 to i ms
Start timer 6

LightLED
Takes an integer (int i) and returns nothing
Set pin i to high (Port AD)

OffLED
Takes an integer (int i) and returns nothing
Set pin i to low (Port AD)

AllOff
Takes no parameters and returns nothing
Set all output pins to low (Port AD)

AllOn
Takes no parameters and returns nothing
Set all output pins to high (Port AD)

Celebration
Takes no parameters and returns nothing
Loop 5 times:
    Turn all LEDs on
    Wait 700 ms
    Turn all LEDs off
    Wait 300 ms

Wait (Blocking code only used for when no inputs should do anything)
Takes an integer (int ms) and returns nothing
Set time1 to current ES_Timer_GetTime
While (ES_Timer_GetTIme <= time1 + ms)

Servo
Takes an integer (int angle) and returns nothing
If angle is less than SERVOMIN, set angle to equal SERVOMIN
If angle is larger than SERVOMAX, set angle to equal SERVOMAX
Use Servo12_SetPulseWidth on the servo pin with 600+10*angle

MessageSender (.c)

void sendMessage(int i)
Takes an integer and does not return anything

switch(i)
    case 1: Raise PTM bits to Message 1 (0x1) and then add a very short delay
    case 2: Raise PTM bits to Message 2 (0x2) and then add a very short delay
    case 3: Raise PTM bits to Message 3 (0x4) and then add a very short delay
    case 4: Raise PTM bits to Message 4 (0x3) and then add a very short delay
    case 5: Raise PTM bits to Message 5 (0x5) and then add a very short delay
    case 6: Raise PTM bits to Message 6 (0x6) and then add a very short delay
    case 7: Raise PTM bits to Message 7 (0x7) and then add a very short delay

LEDCode (.c)

Module level variables – int n1, n2, n3, n4, LEDState CurrentState
n1, n2, n3, n4 are values of random LED code
InitLEDCode
Takes a uint8_t and returns a Boolean
Set MyPriority to inputted parameter
Set n1, n2, n3, and n4 to random numbers from 1-3

Initialize currentState to InitLEDCode
Post ES_Init to LEDCodeSM, and return true if successful, return false if else.

LEDCodeSM
Takes an ES_Event and returns an ES_Event
If EventType is Finished
    nextState = waitToStart

switch (currentState)
    case InitLEDCode:
        if EventType is ES_Init
            Set nextState to WaitToStart
    case WaitToStart:
        if EventType is BATID_FOUND
            Get new random numbers for n1, n2, n3, n4
            Call LEDSequence(n1, n2, n3, n4)
            Set nextState to WaitForFirstButton

    case WaitForFirstButton:
        if EventType is ButtonDown and EventParam is equal to n1
            Set nextState to WaitForSecondButton
        Else if EventParam is not equal to n1
            PulseBuzzer for 500 ms
            Set nextState to WaitForFirstButton

    case WaitForSecondButton:
        if EventType is ButtonDown and EventParam is equal to n2
            Set nextState to WaitForThirdButton
        Else if EventParam is not equal to n2
            PulseBuzzer for 500 ms
            Set nextState to WaitForFirstButton

    case WaitForThirdButton:
        if EventType is ButtonDown and EventParam is equal to n3
            Set nextState to WaitForFourthButton
        Else if EventParam is not equal to n3
            PulseBuzzer for 500 ms
            Set nextState to WaitForFirstButton

    case WaitForFourthButton:
        if EventType is ButtonDown and EventParam is equal to n4
            Set nextState to Idle
            Send CodeSuccess to DefaultSM
        Else if EventParam is not equal to n3
            PulseBuzzer for 500 ms
            Set nextState to WaitForFirstButton

case Idle:
return ES_NO_EVENT;

getNewRand
Takes no parameters and returns nothing

Set n1, n2, n3, and n4 to new random numbers from 1 to 3

DefaultState (.c)

Module level variables – DefaultState CurrentState, uint8_t MyPriority, int angleServo, int basket, int timerActive
InitDefaultSM
Takes a uint8_t and returns a Boolean
Set MyPriority to inputted parameter
Set pins 4, 5, and 6 on Port M to outputs
Set rest of the pins on T and M to inputs
Initialize basket to random number between 0 and 1
Initialize angleServo to SERVOMIN
Initialize timerActive to 0
If basket is 1, send Message 3
If basket is 0, send Message 4
Initialize currentState to InitDefault
Initialize all AD pins outputs by using ADS12_Init
Light LEDs 1, 2, and 3
Check if post ES_Init to DefaultSM returns true
    Return true if no error

RunDefaultSM
Takes an ES_Event and returns an ES_Event

Check if ThisEvent.EventType is ES_TIMEOUT and ThisEvent.EventParam is 7
    Set timerActive to 0

If EventType is Failure
    Turn off all LEDs (OffLED)
    PulseBuzzer for 2000ms
    Set nextState = Lockdown
    Post FINISHED to LEDCodeSM

switch(CurrentState)
    case InitDefault:
        if EventType is ES_Init
            Set nextState to Lockdown

```
        case Lockdown:
                if EventType is BATID_FOUND
                        Randomize basket to 0 or 1
                        Set angleServo to SERVOMIN and call Servo(angleServo)
                        If basket is 1, send message 3
                        If basket is 0, send message 4
                        Set nextState to WaitingForCode
                        Light LED 4
        case WaitingForCode:
                if EventType is CodeSuccess
                        Send message 1
                        Turn LED 4 off
                        Set nextState to WaitingForIR
        case WaitingForIR:
                if EventType is IRDetected
                        if basket is 1, light LED 5
                        if basket is 0, light LED 6
                        set nextState to WaitingForBall
        case WaitingForBall:
                if EventType is BallSuccess and EventParam is equal to basket
                        Turn off LED 5 and 6
                        Light LED 3
                        Send message 2
                        Set nextState to WaitingForAnalog
                if EventType is BallSuccess and EventParam is not equal to basket
                        Send message 6
                        Turn off all LEDs
                        PulseBuzzer for 2000ms
                        Set nextState to Lockdown
                        Post FINISHED to LEDCodeSM
        case WaitingForAnalog:
                if EventType is LeverDown and timerActive is 0
                        Set timer 7 to 70ms and start timer 7
                        Set timerActive to 1
                        Set Servo to angleServo and increment angleServo
                        If angleServo is smaller than SERVOMIN or bigger than SERVOMAX, set angleServo to
SERVOMIN/SERVOMAX
                If EventType is AnalogCodeSuccess
                        Call Celebration
                        Turn off all LEDs
                        Set nextState to Lockdown
                        Post FINISHED to LEDCodeSM
                        Set angleServo and Servo to SERVOMIN

set CurrentState to nextState
return ES_NO_EVENT
```