```cpp
 * FlashLEDCodeCPP.cpp
#include <uTTCOS2013/uTTCOS.h>
#include "FlashLEDCodeCPP.h"

// State Machine Variables (global)
bool LED1_is_on = false;
enum LED3_State {JUST_STARTED, IS_OFF, IS_ON};
LED3_State currentState = JUST_STARTED;
enum LED5_State {LED5_JUST_STARTED, LED5_IS_OFF, LED5_IS_ON, LED5_STAYS_ON};
LED5_State currentState_LED5 = LED5_JUST_STARTED;

// Reset to Base States
void ResetStates_CPP(void) {
    LED1_is_on = false;
    currentState = JUST_STARTED;
    currentState_LED5 = LED5_JUST_STARTED;
}
#define LED1_BITMASK 0x01
void FlashLED1_CPP(void) {
    unsigned char LEDCurrentState = uTTCOS_ReadLED();
    unsigned char nxt_State = NULL;
    if (!LED1_is_on) {
        nxt_State = LED1_BITMASK | LEDCurrentState;
        uTTCOS_WriteLED(nxt_State);  // Turn on LED 1
        LED1_is_on = true;
    }
    else {
        nxt_State = ~LED1_BITMASK & LEDCurrentState;
        uTTCOS_WriteLED(nxt_State);  // Turn off LED 1
        LED1_is_on = false;
    }
}

#define LED3_BIT_MASK  0x04
void FlashLED3_CPP(void) {
    LED3_State nextStateToDo = currentState;
    unsigned char LEDCurrentState = uTTCOS_ReadLED();
    unsigned char nxt_State = NULL;

    switch (currentState) {
    case JUST_STARTED:
        nxt_State = ~LED3_BIT_MASK & LEDCurrentState;
        uTTCOS_WriteLED(nxt_State); // Turn off LED 3
        nextStateToDo = IS_OFF;
        break;
    case IS_OFF:
        nxt_State = LED3_BIT_MASK | LEDCurrentState;
        uTTCOS_WriteLED(nxt_State);  // Was off turn it on
        nextStateToDo = IS_ON;
        break;
    case IS_ON:
        nxt_State = ~LED3_BIT_MASK & LEDCurrentState;
        uTTCOS_WriteLED(nxt_State); // Turn off LED 3
        nextStateToDo = IS_OFF;
    break;
    }

    currentState = nextStateToDo;
}

// Making some C++ extensions to handle "LED state machine"

#define BEGIN_STATE case
#define END_STATE break;

#define LED5_BITMASK 0x10

void FlashLED5_CPP(void) {
    unsigned char LEDCurrentState = uTTCOS_ReadLED();
    unsigned char nxt_State = NULL;
    LED5_State nextStateToDo = currentState_LED5;

    switch (currentState_LED5) {
    BEGIN_STATE LED5_JUST_STARTED:
        nxt_State = ~LED5_BITMASK & LEDCurrentState;
        uTTCOS_WriteLED(nxt_State);              // Turn off LED 5
        nextStateToDo = LED5_IS_OFF;
    END_STATE

    BEGIN_STATE LED5_IS_OFF:
        nxt_State = LED5_BITMASK | LEDCurrentState;
        uTTCOS_WriteLED(nxt_State);  // Was off turn it on
        nextStateToDo = LED5_IS_ON;
    END_STATE

    BEGIN_STATE LED5_IS_ON:
        nextStateToDo = LED5_STAYS_ON;        // Just on -- keep on
    END_STATE

    BEGIN_STATE LED5_STAYS_ON:
        nxt_State = ~LED5_BITMASK & LEDCurrentState;
        uTTCOS_WriteLED(nxt_State);                 // Was off turn it on
        nextStateToDo = LED5_IS_OFF;
    END_STATE
    }

    currentState_LED5 = nextStateToDo;
}

```