```c
// Paul Valenzuela
// Project 3
// 12/06/19
#include <stdint.h>       // C99 data types
#include "TExaS.h"
#include "tm4c123gh6pm.h"

#define STEPPER_B (*((volatile unsigned long *)0x4000503C))     // Port B Pins 0-3
#define STEPPER_E (*((volatile unsigned long *)0x40024078))     // Port E Pins 1-4

// Global Variables
unsigned long Buttons;  // input from PF0, PF4

// Function Prototypes
void PortB_LED_Init(void) ;
void PortE_LED_Init(void) ;

void PortF_Int_Init(void);
void IRSensor_Init(void);

void WaitForInterrupt(void);
void EnableInterrupts(void);

void SysTick_Init(unsigned int);
void SysTick_Handler(void);

void GPIO_PortF_Handler(void);
void GPIO_PortA_Handler(void);


// Global Variables
uint32_t currentState;
uint32_t stepsRemaining;
uint32_t Input;

// State Machine Structure
struct State {
  uint32_t Out;           //Output
  uint32_t Delay;         //Delay
  const uint32_t Next[8]; //Array of states
};

typedef const struct State STyp;

// Pre Defined States
#define Stop_State 0
#define A_State 1
#define B_State 2
#define C_State 3
#define D_State 4

//State Machine
STyp FSM[5] = {
  {0x00, 2, {0,0,0,0,0,4,1,0}},     // Stop_State
  {0x93, 2, {0,0,0,0,0,4,2,0}},     // A_State
  {0xC6, 2, {0,0,0,0,0,1,3,0}},     // B_State
  {0x6C, 2, {0,0,0,0,0,2,4,0}},     // C_State
  {0x39, 2, {0,0,0,0,0,3,1,0}}      // D_State
};


// Main Method
int main(void){

  // Initialize Inputs
  PortF_Int_Init();       // Call Initialization of Port PF4 PF0 with Interrupts
  IRSensor_Init();        // Call Initialization of Port PA2 with Interrupts

  // Initialize Outputs
  PortB_LED_Init();       // Initialize PB3 PB2 PB1 PB0
  PortE_LED_Init();       // Initialize PE4 PE3 PE2 PE1

```

```
73      // Initialize Timer
74      SysTick_Init(1);
75
76      while(1){
77        // Wait For Interrupts To Occur
78        WaitForInterrupt();
79        }
80
81    }
82
83    // Method That Initializes SysTick Timer
84    void SysTick_Init(unsigned int amp){
85
86      NVIC_ST_CTRL_R &= 0x00;           // Turn off
87      NVIC_ST_RELOAD_R = 16e3 * amp;    // Set Timer Based on Paramter 1ms * amp
88      NVIC_ST_CURRENT_R = 0;            // Reset Counter
89
90      NVIC_ST_CTRL_R |= 0x07;           // Turn Timer On
91      NVIC_PRI7_R = (NVIC_PRI7_R & 0xFF00FFFF) | 0x00400000;  // Priority 2
92    }
93
94    // Initialize Edge Trigger Interrupt For PA3 PA2 Rising Edge
95    void IRSensor_Init(void)
96    {
97      uint32_t volatile delay;            // Declaring variable that will be used to setup PORT A
98
99      SYSCTL_RCGC2_R |= 0x00000001;       // Turn on A clock
100     delay = SYSCTL_RCGC2_R;             // Delay 3-5 bus cycles
101
102     GPIO_PORTA_DIR_R &= ~0x04;          // PA2 Input
103     GPIO_PORTA_DEN_R |= 0x04;           // Digital Enable PA2
104
105
106     GPIO_PORTA_IS_R &= ~0x04;           //  PA2 is edge-Sensitive
107     GPIO_PORTA_IBE_R &= ~0x04;          //  PA2 is both edges
108     GPIO_PORTA_IEV_R |= 0x04 ;          //  Port A is Positive Edge Triggered
109     GPIO_PORTA_ICR_R |= 0x04;           //  Clear flag3, flag2
110     GPIO_PORTA_IM_R |= 0x04;            //  Arm interrupt on PA3, PA2
111
112     NVIC_PRI7_R = (NVIC_PRI7_R&0xFFFFFF00) | 0x000000E0; // Priority 7
113     NVIC_EN0_R |= 0x00000001;                            // enable interrupt 30 NVIC
114
115   }
116
117   // Initialize Port B LEDs
118   void PortB_LED_Init(void)
119   {
120     uint32_t volatile delay;        // Declaring variable that will be used to setup PORT B
121     SYSCTL_RCGC2_R |= 0x00000002;   // Turn on B clock
122
123     delay = SYSCTL_RCGC2_R;         // Delay 3-5 bus cycles
124     GPIO_PORTB_AMSEL_R = 0x00;      // Disable analog function
125     GPIO_PORTB_PCTL_R = 0x00000000; // GPIO clear bit PCTL  111
126     GPIO_PORTB_DIR_R |= 0x0F;       // PB3, PB2, PB1, PB0 output
127     GPIO_PORTB_AFSEL_R = 0x00;      // No alternate function
128     GPIO_PORTB_DEN_R |= 0x0F;       // Enable digital pins PB3-PB0
129   }
130
131   // Initialize Port B LEDs 103
132   void PortE_LED_Init(void)
133   {
134     uint32_t volatile delay;        // Declaring variable that will be used to setup PORT B
135     SYSCTL_RCGC2_R |= 0x00000010;   // Turn on B clock
136
137     delay = SYSCTL_RCGC2_R;         // Delay 3-5 bus cycles
138     GPIO_PORTE_AMSEL_R = 0x00;      // Disable analog function
139     GPIO_PORTE_PCTL_R = 0x00000000; // GPIO clear bit PCTL  111
140     GPIO_PORTE_DIR_R |= 0x1E;       // PE4, PE3, PE2, PE1 output
141     GPIO_PORTE_AFSEL_R = 0x00;      // No alternate function
142     GPIO_PORTE_DEN_R |= 0x1E;       // Enable digital pins PE4-PE1
143   }
144
```

```
145   // Initialize edge trigger interrupt for PF0 PF4 rising edge
146   void PortF_Int_Init() {
147     uint32_t volatile delay;          // Declaring variable that will be used to setup PORT F
148
149     SYSCTL_RCGC2_R |= 0x00000020;     // Turn on F clock
150     delay = SYSCTL_RCGC2_R;           // Delay 3-5 bus cycles
151
152     GPIO_PORTF_LOCK_R |= 0x4C4F434B;   // Unlock PortF PF0
153     GPIO_PORTF_CR_R |= 0x11;           // Allow changes to PF0 PF4
154     GPIO_PORTF_DIR_R &= ~0x11;         // PF0 PF4 Input
155     GPIO_PORTF_PUR_R |= 0x11;          // Enable pullup resistors on PF0  PF4
156     GPIO_PORTF_DEN_R |= 0x11;          // Digital Enable PF0 PF4
157
158     GPIO_PORTF_IS_R &= ~0x11;          // PF0 PF4 is edge-Sensitive
159     GPIO_PORTF_IBE_R &= ~0x11;         // PF0 PF4 is not both edges
160     GPIO_PORTF_IEV_R |= 0x00;          // PF0 PF4 Falling edge event
161     GPIO_PORTF_ICR_R |= 0x11;          // Clear flag0 flag4
162     GPIO_PORTF_IM_R |= 0x11;           // arm interrupt on PF0 PF4
163
164     NVIC_PRI7_R = (NVIC_PRI7_R&0xFF00FFFF) | 0x00600000; // Priority 1
165     NVIC_EN0_R |= 0x40000000;                            // Enable interrupt 30 NVIC
166   }
167
168   // Method That Handles SysTick Enabled Interrupts
169   void SysTick_Handler(){
170     // Decrement stepsRemaining
171     if(stepsRemaining > 0)
172     { stepsRemaining--; }
173
174     // 'Move' Variable is Still On If There Are Steps Remaining
175     if(stepsRemaining > 0)
176     { Input |= 0x04;  }
177     // Turn Off 'Move' Variable If There Are No Steps Remaining
178     else
179     { Input &= ~0x04; }
180
181     // Current State Gets Next State
182     currentState = FSM[currentState].Next[Input];
183     // Stepper B Gets Output
184     STEPPER_B = FSM[currentState].Out & 0x0F;
185     // Stepper E Gets Output
186     STEPPER_E = (FSM[currentState].Out & 0xF0) >> 3;
187
188     // Reload SysTick Timer With New Value
189     SysTick_Init(FSM[currentState].Delay);
190
191   }
192
193   // Handle GPIO Port F interrupts. When Port F interrupt triggers, do what's necessary then send output
      signals to the circuit.
194   void GPIOPortF_Handler(void) {
195
196       // Buttons Variable Gets The Value of PF4 PF0
197       Buttons = GPIO_PORTF_DATA_R;
198       // Acknowledge Flag0 Flag4
199       GPIO_PORTF_ICR_R = 0x11;
200
201       // If Button 1 Pressed, Go Forward
202       if(Buttons == 0x01)
203       {
204         // Drives Approximately 36 Inches
205         stepsRemaining = 9000;
206
207         // 'Backward' Off
208         Input &= ~0x01;
209         // 'Forward' On
210         Input |= 0x02;
211       }
212       else if(Buttons == 0x10)
213       {
214         // Drives Approximately 3 Inches
215         stepsRemaining = 750;
```

```
216         // 'Forward' Off
217         Input &= ~0x02;
218         // 'Backward' On
219         Input |= 0x01;
220       }
221       else {
222         // Do Not Drive
223         stepsRemaining = 0;
224         // 'Move' Off
225         Input &= ~0x03;
226       }
227
228   }
229
230   // Handle GPIO Port A interrupts
231   void GPIOPortA_Handler(void) {
232       // Acknowledge PA2 Flag
233       GPIO_PORTA_ICR_R |= 0x04;
234
235       // If The IR Sensor Is On
236       if((GPIO_PORTA_DATA_R & 0xFF) == 0x00)
237       {
238         // 'Forward' Off
239         Input &= ~0x02;
240       }
241   }
242
```