Lab 5 FSM

Walk PF3, Don't Walk PF1

PE2, PE1, PE0

Walk West

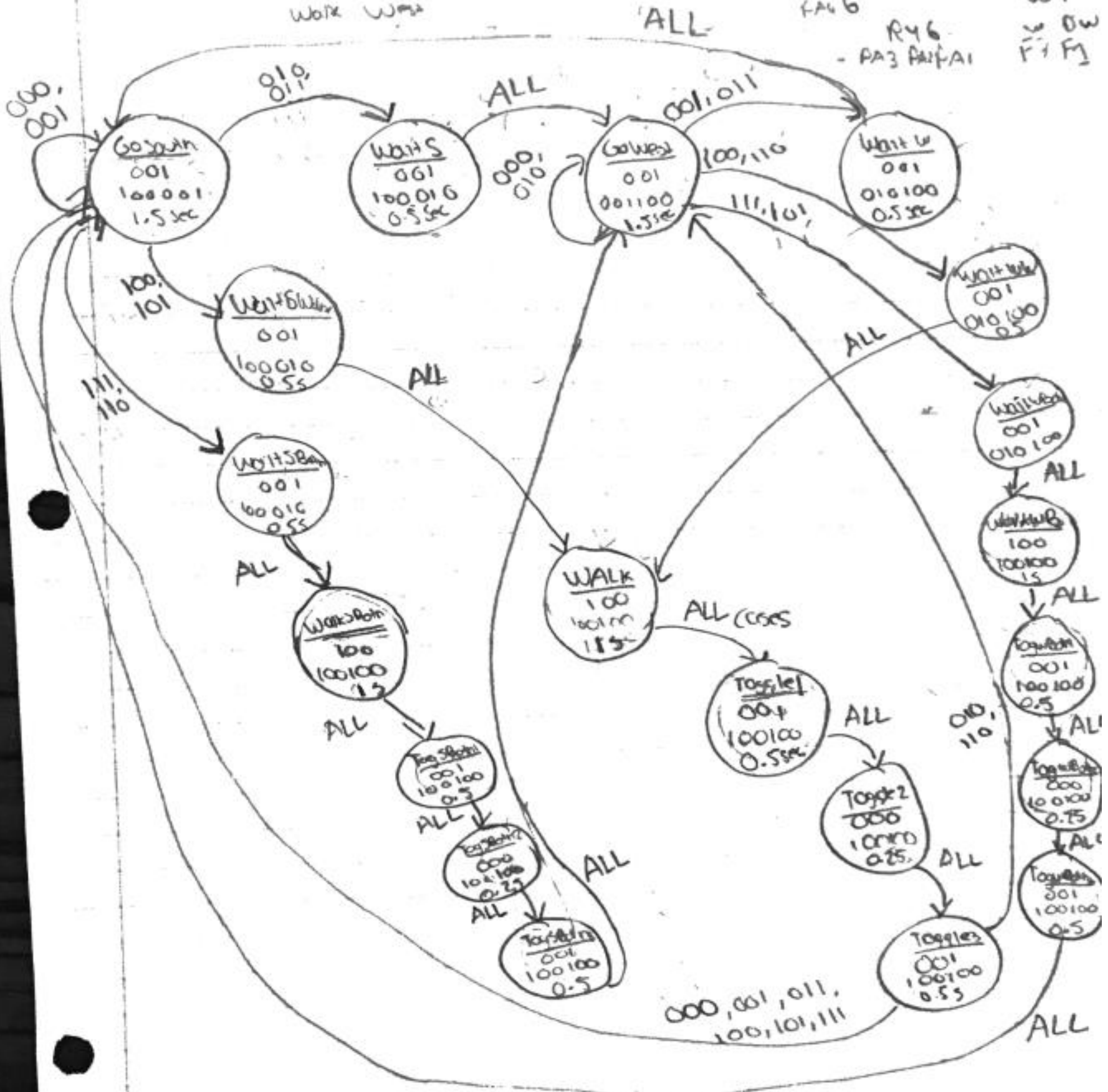
PE2 PE0

Walk S

PAGR
PACY
PA4 6

RY6
- PA3 PA2 PA1

W PE1
W DW
F1 F1



```
#include <stdint.h>
#include "tm4c123gh6pm.h"
#include "SysTick.h"
#include "TExaS.h"

// Declare your FSM linked structure here
struct State {
 uint32_t Output;
 uint32_t Time;
 uint32_t Next[8];
};
typedef const struct State Stype;
#define GoS      0
#define waitS     1
#define waitSWalk   2
#define waitSBoth   3
#define GoW      4
#define waitW     5
#define waitWWalk   6
#define waitWBoth   7
#define walkReg    8
#define TogReg1    9
#define TogReg2    10
#define TogReg3    11
#define walkSBoth   12
#define TogSBoth1   13
#define TogSBoth2   14
#define TogSBoth3   15
#define walkWBoth   16
#define TogWBoth1   17
#define TogWBoth2   18
#define TogWBoth3   19
Stype FSM[20]={
 {0x85, 150, {GoS, GoS, waitS, waitS, waitSWalk, waitSWalk, waitSBoth, waitSBoth}},
 {0x89, 50, {GoW, GoW, GoW, GoW, GoW, GoW, GoW, GoW}},
 {0x89, 50, {walkReg, walkReg, walkReg, walkReg, walkReg, walkReg, walkReg, walkReg}},
 {0x89, 50, {walkSBoth, walkSBoth, walkSBoth, walkSBoth, walkSBoth, walkSBoth, walkSBoth, walkSBoth}},
 {0x31, 150, {GoW, waitW, GoW, waitW, waitWWalk, waitWBoth, waitWWalk, waitWBoth}},
 {0x51, 50, {GoS, GoS, GoS, GoS, GoS, GoS, GoS, GoS}},
 {0x51, 50, {walkReg, walkReg, walkReg, walkReg, walkReg, walkReg, walkReg, walkReg}},
 {0x51, 50, {walkWBoth, walkWBoth, walkWBoth, walkWBoth, walkWBoth, walkWBoth, walkWBoth,
walkWBoth}},
 {0x92, 100, {TogReg1, TogReg1, TogReg1, TogReg1, TogReg1, TogReg1, TogReg1, TogReg1}},
 {0x91, 50, {TogReg2, TogReg2, TogReg2, TogReg2, TogReg2, TogReg2, TogReg2, TogReg2}},
 {0x90, 25, {TogReg3, TogReg3, TogReg3, TogReg3, TogReg3, TogReg3, TogReg3, TogReg3}},
 {0x91, 50, {GoS, GoS, GoW, GoS, GoS, GoS, GoW, GoS}},
 {0x92, 100, {TogSBoth1, TogSBoth1, TogSBoth1, TogSBoth1, TogSBoth1, TogSBoth1, TogSBoth1,
TogSBoth1}},
 {0x91, 50, {TogSBoth2, TogSBoth2, TogSBoth2, TogSBoth2, TogSBoth2, TogSBoth2, TogSBoth2,
TogSBoth2}},
 {0x90, 25, {TogSBoth3, TogSBoth3, TogSBoth3, TogSBoth3, TogSBoth3, TogSBoth3, TogSBoth3,
TogSBoth3}},
 {0x91, 50, {GoW, GoW, GoW, GoW, GoW, GoW, GoW, GoW}},
 {0x92, 100, {TogWBoth1, TogWBoth1, TogWBoth1, TogWBoth1, TogWBoth1, TogWBoth1, TogWBoth1,
TogWBoth1}},
 {0x91, 50, {TogWBoth2, TogWBoth2, TogWBoth2, TogWBoth2, TogWBoth2, TogWBoth2, TogWBoth2,
TogWBoth2}},
 {0x90, 25, {TogWBoth3, TogWBoth3, TogWBoth3, TogWBoth3, TogWBoth3, TogWBoth3, TogWBoth3,
TogWBoth3}},
 {0x91, 50, {GoS, GoS, GoS, GoS, GoS, GoS, GoS, GoS}},
};

uint32_t state;
uint32_t input;
void EnableInterrupts(void);

int main(void){ volatile unsigned long delay;
  TExaS_Init(SW_PIN_PE210, LED_PIN_PB543210); // activate traffic simulation and set system clock to 80
MHz
  SysTick_Init();

 SYSCTL_RCGC2_R |= 0x31; //A,E,F clock on
 delay = SYSCTL_RCGC2_R;

 GPIO_PORTA_AMSEL_R &= ~0xFC; // PA 7-2 are the output lights
 GPIO_PORTA_PCTL_R &= ~0x0000000FF;
 GPIO_PORTA_DIR_R |= 0xFC;
 GPIO_PORTA_AFSEL_R &= ~0xFC;
 GPIO_PORTA_DEN_R |= 0xFC;

 GPIO_PORTE_AMSEL_R &= ~0x07; // PE 2-0 are the inputs
 GPIO_PORTE_PCTL_R &= ~0x0000000FF;
 GPIO_PORTE_DIR_R &= ~0x07;
 GPIO_PORTE_AFSEL_R &= ~0x07;
 GPIO_PORTE_DEN_R |= 0x07;

 GPIO_PORTF_AMSEL_R &= ~0x0A;
 GPIO_PORTF_PCTL_R &= ~0x0000000FF;
 GPIO_PORTF_DIR_R |= 0x0A; // PF 1 and 3 are outputs for walk, don't walk
 GPIO_PORTF_AFSEL_R &= ~0x0A;
 GPIO_PORTF_DEN_R |= 0x0A;

 state = GoS; // first state is just go south

  EnableInterrupts();
  //FSM Engine
  while(1){
  GPIO_PORTA_DATA_R = (FSM[state].Output & 0xFC);
  GPIO_PORTF_DATA_R = (((FSM[state].Output & 0x01) <<1) | ((FSM[state].Output & 0x02) << 2));
  SysTick_Wait10ms(FSM[state].Time);
  input = GPIO_PORTE_DATA_R & 0x07;
  state = FSM[state].Next[input];
  }
}
```

```
#include <stdint.h>
#include "tm4c123gh6pm.h"

// Initialize SysTick with busy wait running at bus clock.
void SysTick_Init(void){
  NVIC_ST_CTRL_R = 0;                   // disable SysTick during setup
  NVIC_ST_CTRL_R = 0x00000005;     // enable SysTick with core clock
}

// The delay parameter is in units of the core clock. (units of 12.5 nsec for 80 MHz clock)
void SysTick_Wait(uint32_t delay){
 NVIC_ST_RELOAD_R = delay-1;  // number of counts to wait
 NVIC_ST_CURRENT_R = 0;   // any vlaue written clears the CURRENT
  while((NVIC_ST_CTRL_R&0x00010000)==0){ // wait for count flag
 }
}

// This assumes 80 MHz system clock.
void SysTick_Wait10ms(uint32_t delay){
  uint32_t i;
  for(i=0; i<delay; i++){
    SysTick_Wait(800000);  // wait 10ms (assumes 80 MHz clock)
  }
}
```