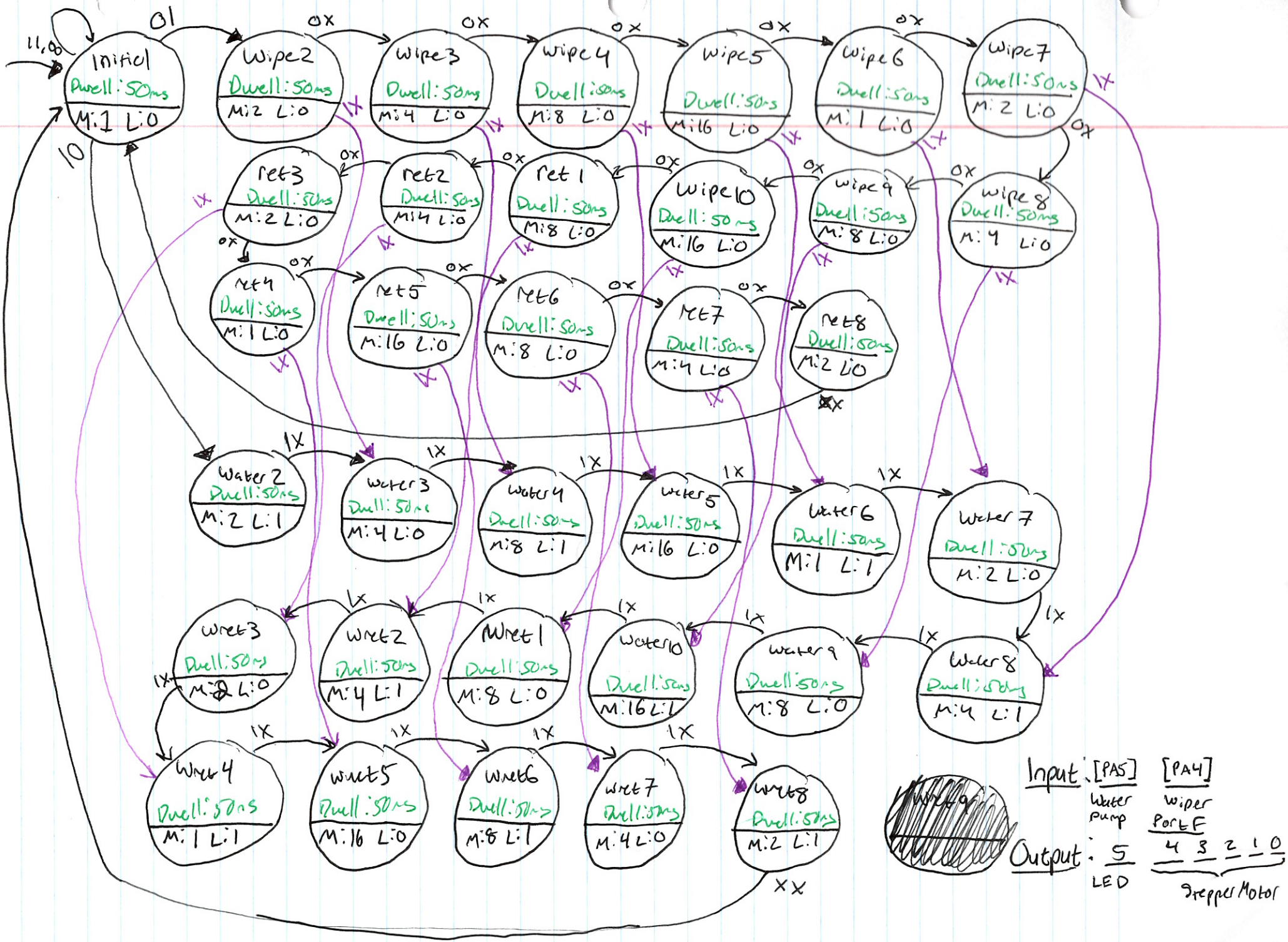
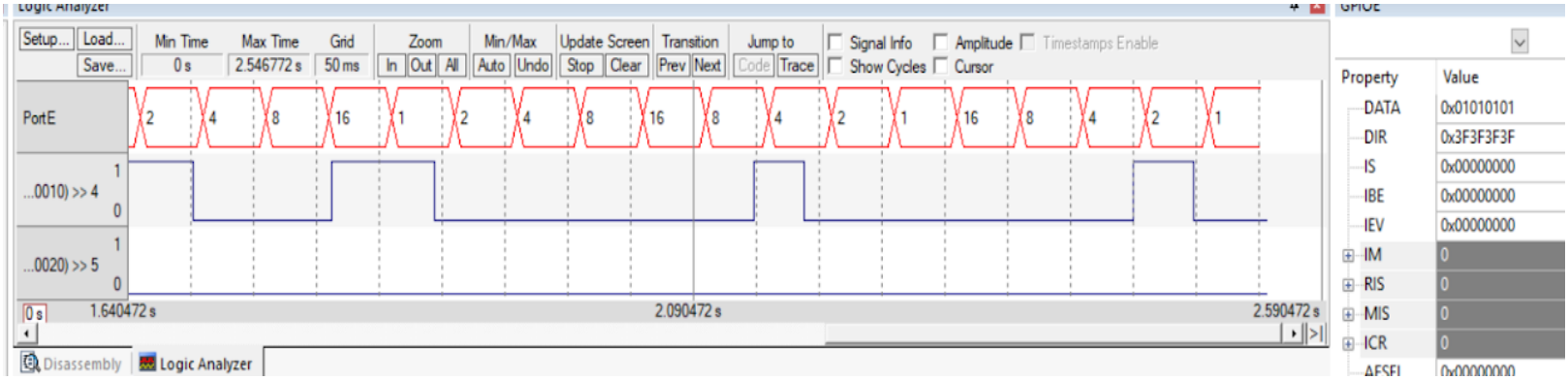
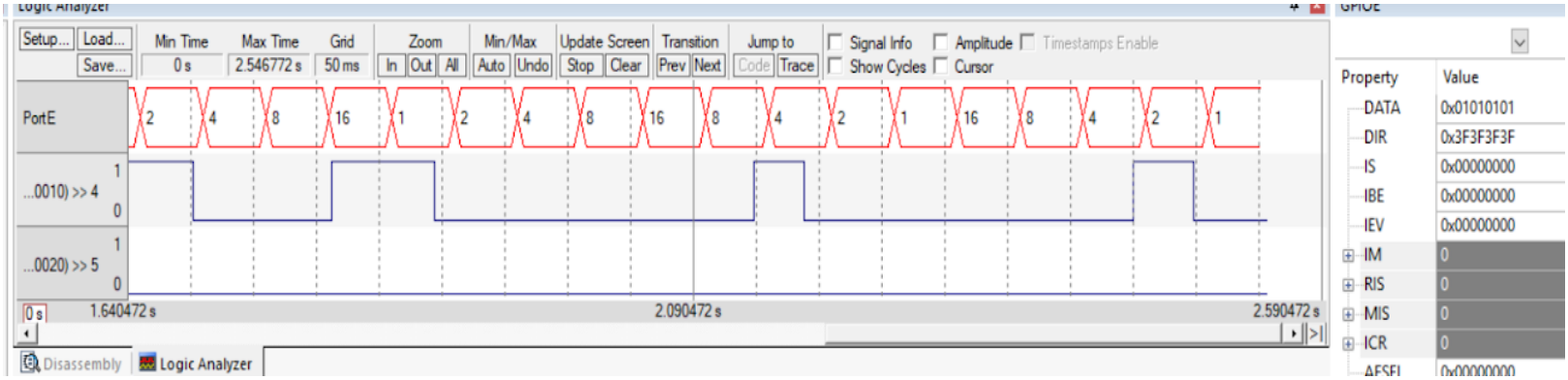

11,00
01
10
Initial
Dwell: 50ms
M:1 L:0
Wipe2
Dwell: 50ms
M:2 L:0
Wipe3
Dwell: 50ms
M:4 L:0
Wipe4
Dwell: 50ms
M:8 L:0
Wipe5
Dwell: 50ms
M:16 L:0
Wipe6
Dwell: 50ms
M:1 L:0
Wipe7
Dwell: 50ms
M:2 L:0
Wipe8
Dwell: 50ms
M:4 L:0
Wipe9
Dwell: 50ms
M:8 L:0
Wipe10
Dwell: 50ms
M:16 L:0
Ret1
Dwell: 50ms
M:8 L:0
Ret2
Dwell: 50ms
M:4 L:0
Ret3
Dwell: 50ms
M:2 L:0
Ret4
Dwell: 50ms
M:1 L:0
Ret5
Dwell: 50ms
M:16 L:0
Ret6
Dwell: 50ms
M:8 L:0
Ret7
Dwell: 50ms
M:4 L:0
Ret8
Dwell: 50ms
M:2 L:0
Water2
Dwell: 50ms
M:2 L:1
Water3
Dwell: 50ms
M:4 L:0
Water4
Dwell: 50ms
M:8 L:1
Water5
Dwell: 50ms
M:16 L:0
Water6
Dwell: 50ms
M:1 L:1
Water7
Dwell: 50ms
M:2 L:0
Water8
Dwell: 50ms
M:4 L:1
Water9
Dwell: 50ms
M:8 L:0
Water10
Dwell: 50ms
M:16 L:1
Wret1
Dwell: 50ms
M:8 L:0
Wret2
Dwell: 50ms
M:4 L:1
Wret3
Dwell: 50ms
M:2 L:0
Wret4
Dwell: 50ms
M:1 L:1
Wret5
Dwell: 50ms
M:16 L:0
Wret6
Dwell: 50ms
M:8 L:1
Wret7
Dwell: 50ms
M:4 L:0
Wret8
Dwell: 50ms
M:2 L:1
0X
1X
XX
Input: [PA5] Water Pump [PA4] Wiper
Output: 5 LED
PortF 4 3 2 1 0
Stepper Motor


Logic Analyzer
Setup...
Load...
Save...
Min Time
0 s
Max Time
2.546772 s
Grid
50 ms
Zoom
In
Out
All
Min/Max
Auto
Undo
Update Screen
Stop
Clear
Transition
Prev
Next
Jump to
Code
Trace
Signal Info
Show Cycles
Amplitude
Cursor
Timestamps Enable
PortE
2
4
8
16
1
2
4
8
16
8
4
2
1
16
8
4
2
1
...0010) >> 4
1
0
...0020) >> 5
1
0
0 s
1.640472 s
2.090472 s
2.590472 s
Disassembly
Logic Analyzer
GPIOE
Property
Value
DATA
0x01010101
DIR
0x3F3F3F3F
IS
0x00000000
IBE
0x00000000
IEV
0x00000000
IM
0
RIS
0
MIS
0
ICR
0
AFSEL
0x00000000

```
// StepperMotorController.c starter file EE319K Lab 5
// Runs on TM4C123
// Finite state machine to operate a stepper motor.
// Jonathan Valvano
// January 18, 2019

// Hardware connections (External: two input buttons and four outputs to stepper motor)
//  PA5 is Wash input  (1 means pressed, 0 means not pressed)
//  PA4 is Wiper input  (1 means pressed, 0 means not pressed)
//  PE5 is Water pump output (toggle means washing)
//  PE4-0 are stepper motor outputs
//  PF1 PF2 or PF3 control the LED on Launchpad used as a heartbeat
//  PB6 is LED output (1 activates external LED on protoboard)

#include "SysTick.h"
#include "TExaS.h"
#include <stdint.h>
#include "../inc/tm4c123gh6pm.h"

  struct State {
    uint32_t out;     // 2-bit output
    //uint32_t dwell;   // time to delay
    uint32_t next[4]; // next if 2-bit input is 0-3
  };
    typedef const struct State State_t;

void EnableInterrupts(void);
// edit the following only if you need to move pins from PA4, PE3-0
// logic analyzer on the real board
#define PA4       (*((volatile unsigned long *)0x40004040))
#define PE50      (*((volatile unsigned long *)0x400240FC))
#define initial   0x00;
#define wipe2     0x01;
#define wipe3     0x02;
#define wipe4     0x03;
#define wipe5     0x04;
#define ret4      0x05;
#define ret3      0x06;
#define ret2      0x07;
#define water2    0x08;
#define water3    0x09;
#define water4    0x0A;
#define water5    0x0B;
#define wret4     0x0C;
#define wret3     0x0D;
#define wret2     0x0E;


void SendDataToLogicAnalyzer(void){
  UART0_DR_R = 0x80|(PA4<<2)|PE50;
}

State_t fsm[37] = {
      //{/*initial*/   1, initial, wipe2, water2, initial},
  //    {/*wipe2*/   0x02, {wipe3, wipe3, wipe3, wipe3}},
  //    {/*wipe3*/   0x04, {wipe4, wipe4, wipe4, wipe4}},
  //    {/*wipe4*/   0x08, {wipe5, wipe5, wipe5, wipe5}},
    //  {/*wipe5*/   0x10, {ret4, ret4, ret4, ret4}},
      //{/*ret4*/      0x08, {ret3, ret3, ret3, ret3}},
      //{/*ret3*/      0x04, {ret2, ret2, ret2, ret2}},
      //{/*ret2*/     0x02, {initial, initial, initial, initial}},
      //{/*water2*/    0x22, {water3, water3, water3, water3}},
      //{/*water3*/    0x24, {water4, water4, water4, water4}},
    //{/*water4*/    0x28, {water5, water5, water5, water5}},
      //{/*water5*/    0x30, {wret4, wret4, wret4, wret4}},
      //{/*wret4*/   0x28, {wret3, wret3, wret3, wret3}},
      //{/*wret3*/   0x24, {wret2, wret2, wret2, wret2}},
      //{/*wret2*/   0x22, {initial, initial, initial, initial}}
  {0x01, {0x00, 0x01, 0x13, 0x00}},
  {0x02, {0x02, 0x02, 0x14, 0x14}},
  {0x04, {0x03, 0x03, 0x15, 0x15}},
  {0x08, {0x04, 0x04, 0x16, 0x16}},
```

```
    {0x10, {0x05, 0x05, 0x17, 0x17}},
    {0x01, {0x06, 0x06, 0x18, 0x18}},
    {0x02, {0x07, 0x07, 0x19, 0x19}},
    {0x04, {0x08, 0x08, 0x1A, 0x1A}},
    {0x08, {0x09, 0x09, 0x1B, 0x1B}},
    {0x10, {0x0A, 0x0A, 0x1C, 0x1C}},
    {0x08, {0x0B, 0x0B, 0x1D, 0x1D}},
    {0x04, {0x0C, 0x0C, 0x1E, 0x1E}},
    {0x02, {0x0D, 0x0D, 0x1F, 0x1F}},
    {0x01, {0x0E, 0x0E, 0x20, 0x20}},
    {0x10, {0x0F, 0x0F, 0x21, 0x21}},
    {0x08, {0x10, 0x10, 0x22, 0x22}},
    {0x04, {0x11, 0x11, 0x23, 0x23}},
    {0x02, {0x12, 0x12, 0x24, 0x24}},
    {0x01, {0x00, 0x00, 0x00, 0x00}},
    //water pump/led flash
    {0x22, {0x02, 0x02, 0x14, 0x14}},
    {0x04, {0x03, 0x03, 0x15, 0x15}},
    {0x28, {0x04, 0x04, 0x16, 0x16}},
    {0x10, {0x05, 0x05, 0x17, 0x17}},
    {0x21, {0x06, 0x06, 0x18, 0x18}},
    {0x02, {0x07, 0x07, 0x19, 0x19}},
    {0x24, {0x08, 0x08, 0x1A, 0x1A}},
    {0x08, {0x09, 0x09, 0x1B, 0x1B}},
    {0x30, {0x0A, 0x0A, 0x1C, 0x1C}},
    {0x08, {0x0B, 0x0B, 0x1D, 0x1D}},
    {0x24, {0x0C, 0x0C, 0x1E, 0x1E}},
    {0x02, {0x0D, 0x0D, 0x1F, 0x1F}},
    {0x21, {0x0E, 0x0E, 0x20, 0x20}},
    {0x10, {0x0F, 0x0F, 0x21, 0x21}},
    {0x28, {0x10, 0x10, 0x22, 0x22}},
    {0x04, {0x11, 0x11, 0x23, 0x23}},
    {0x22, {0x12, 0x12, 0x24, 0x24}},
    {0x01, {0x00, 0x00, 0x00, 0x00}},
};

  volatile uint8_t k =0;
  uint8_t Curr_State = 0;
  uint8_t input;

int main(void){
  TExaS_Init(&SendDataToLogicAnalyzer);    // activate logic analyzer and set system clock to 80 MHz
  SysTick_Init();
// you initialize your system here
  SYSCTL_RCGCGPIO_R |= 0x31;
  k = 1;
  k = 1;                                  // waiting for clock to start
  GPIO_PORTE_DIR_R |= 0x3F;
  GPIO_PORTE_DEN_R |= 0x3F;
  GPIO_PORTA_DIR_R &= ~(0x30);
  GPIO_PORTA_DEN_R |= 0x30;
  GPIO_PORTF_DIR_R |= 0x02;
  GPIO_PORTF_DEN_R |= 0x02;

  EnableInterrupts();
  while(1){
// output
    GPIO_PORTE_DATA_R = fsm[Curr_State].out;
// wait
    SysTick_Wait10ms(5);
// input
    input = GPIO_PORTA_DATA_R & 0x30;
    input >>= 4;
// next
    Curr_State = fsm[Curr_State].next[input];
  }
}
```


Logic Analyzer
Setup...
Load...
Save...
Min Time
0 s
Max Time
2.546772 s
Grid
50 ms
Zoom
In
Out
All
Min/Max
Auto
Undo
Update Screen
Stop
Clear
Transition
Prev
Next
Jump to
Code
Trace
Signal Info
Show Cycles
Amplitude
Cursor
Timestamps Enable
PortE
2
4
8
16
1
2
4
8
16
8
4
2
1
16
8
4
2
1
...0010) >> 4
1
0
...0020) >> 5
1
0
0 s
1.640472 s
2.090472 s
2.590472 s
Disassembly
Logic Analyzer
GPIOE
Property
Value
DATA
0x01010101
DIR
0x3F3F3F3F
IS
0x00000000
IBE
0x00000000
IEV
0x00000000
IM
0
RIS
0
MIS
0
ICR
0
AFSEL
0x00000000