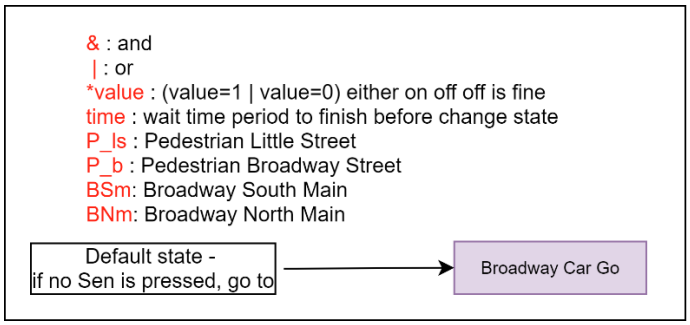
ELEC3042 2021 Defence 2 Questions:

There are some assumptions:

- Bus can turn left to go to little street (Bus Light on - can go straight or turn)
- In Broadway Car Go state (North), Car can turn left at all time
- Pedestrian in Broadway Street at North and South are actually two different States, however, properties of each are completely same, so I present it as 1 State only
- DIAG 1, DIAG 2, DIAG 3 have green, yellow, red color respectively.
- When Port D Pin 7 (hazard mode) is pressed, to terminate this state: wait for 10 second and go to default state or click on a next State to stop Hazard state.

There are also some notations:



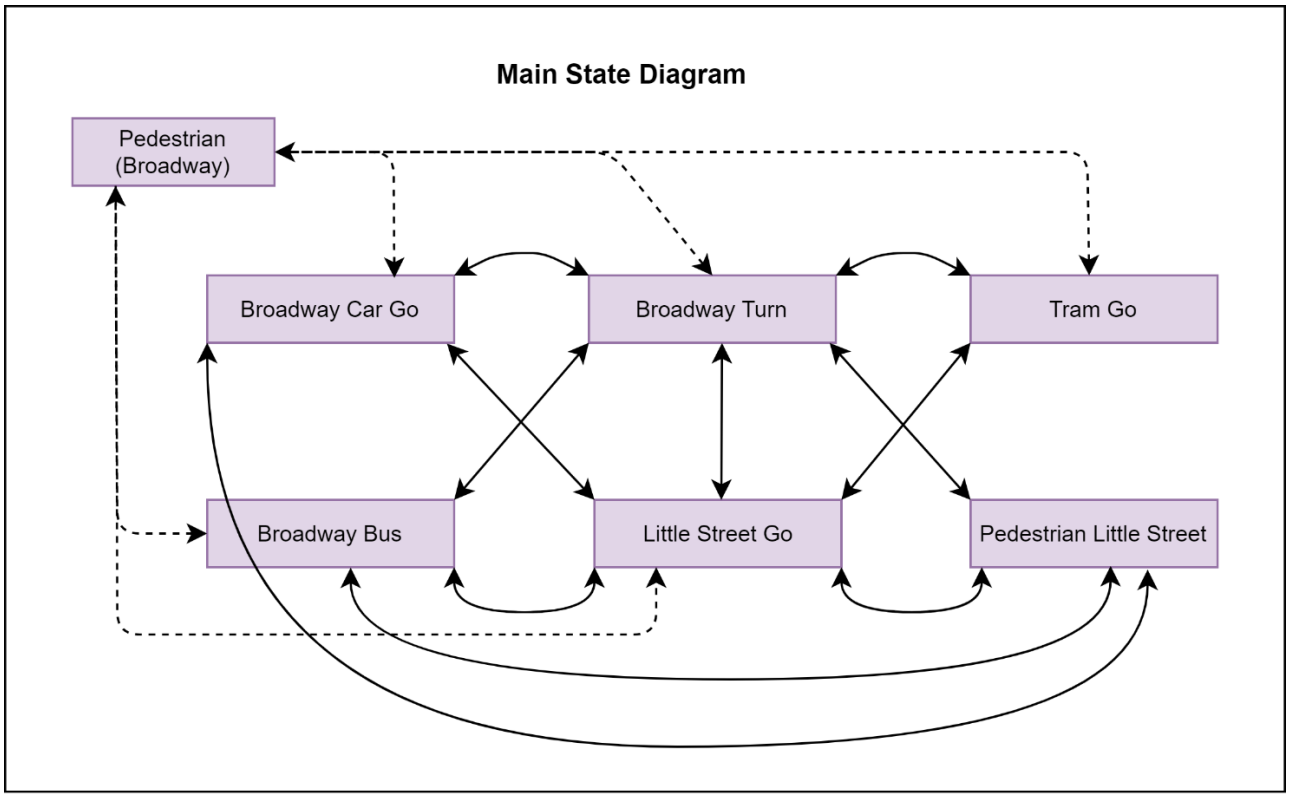


Figure 1 Main State Diagram

This is the Main State Diagram to show the relation between attributes. Personally, this step is an absolute vital process in designing system states, although, it is not mentioned in class,

it is necessary while coding, to give you an overall looking to all states. It brings confusion at first glance; however it is straightforward when each attribute is concentrated.

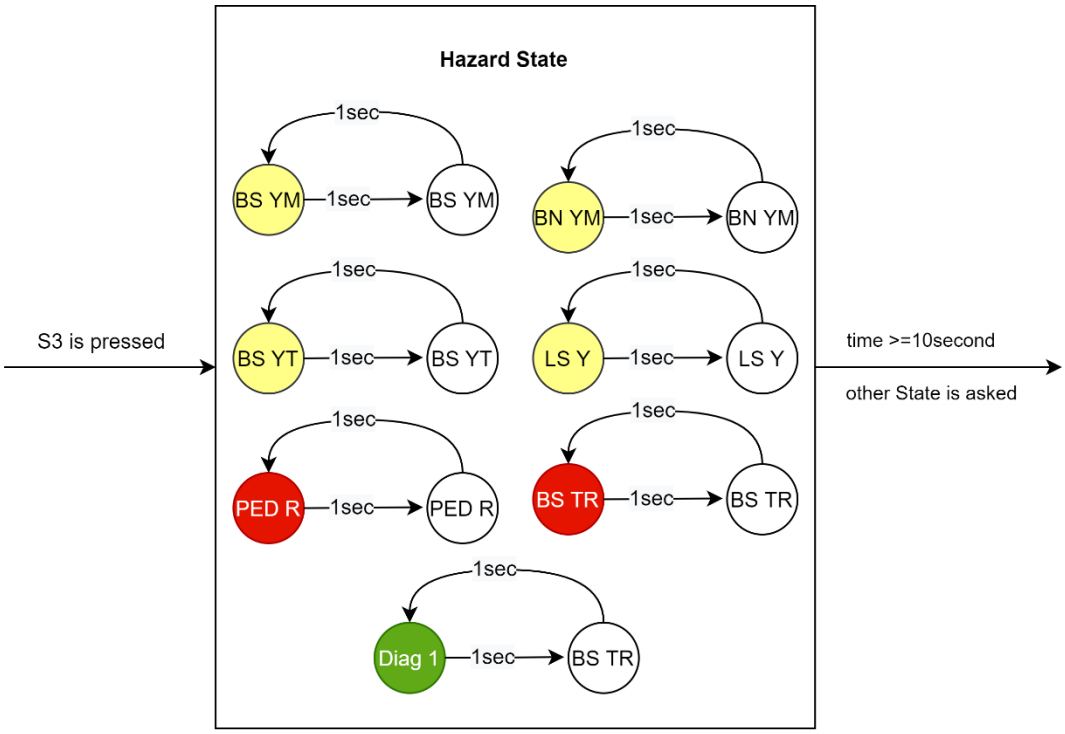


Figure 2 Hazard State

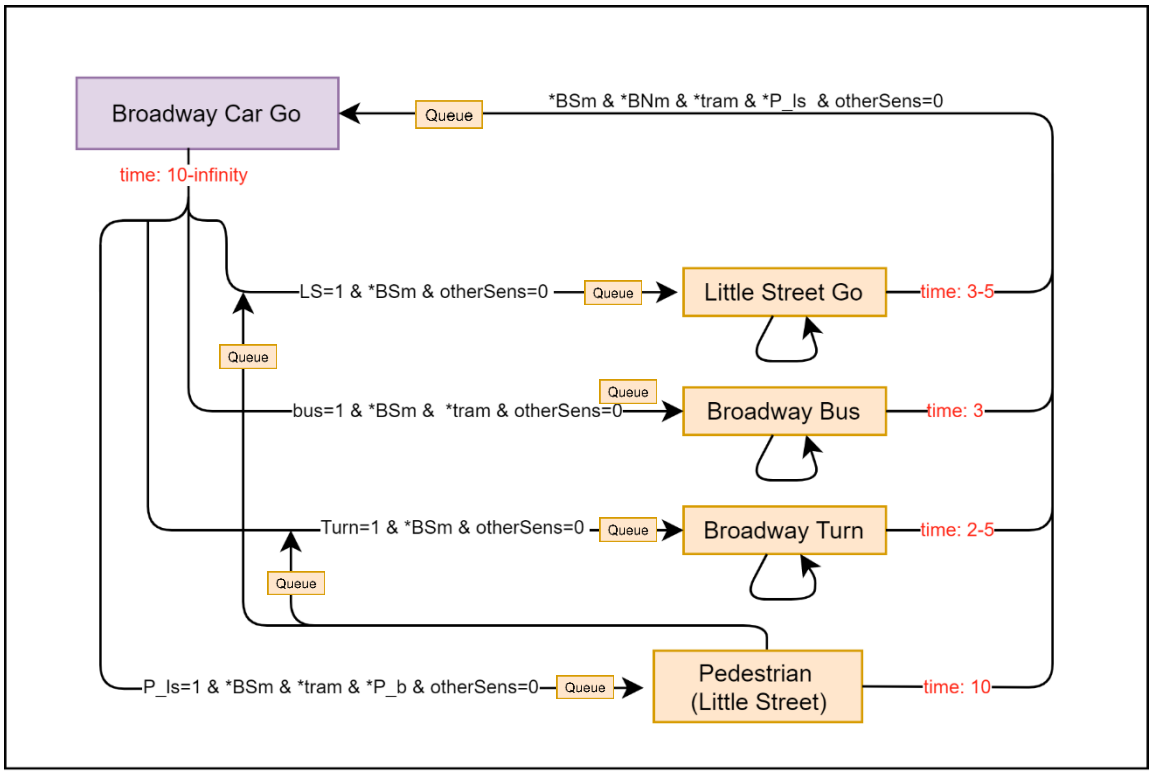


Figure 3 State Relation with Detail Conditions

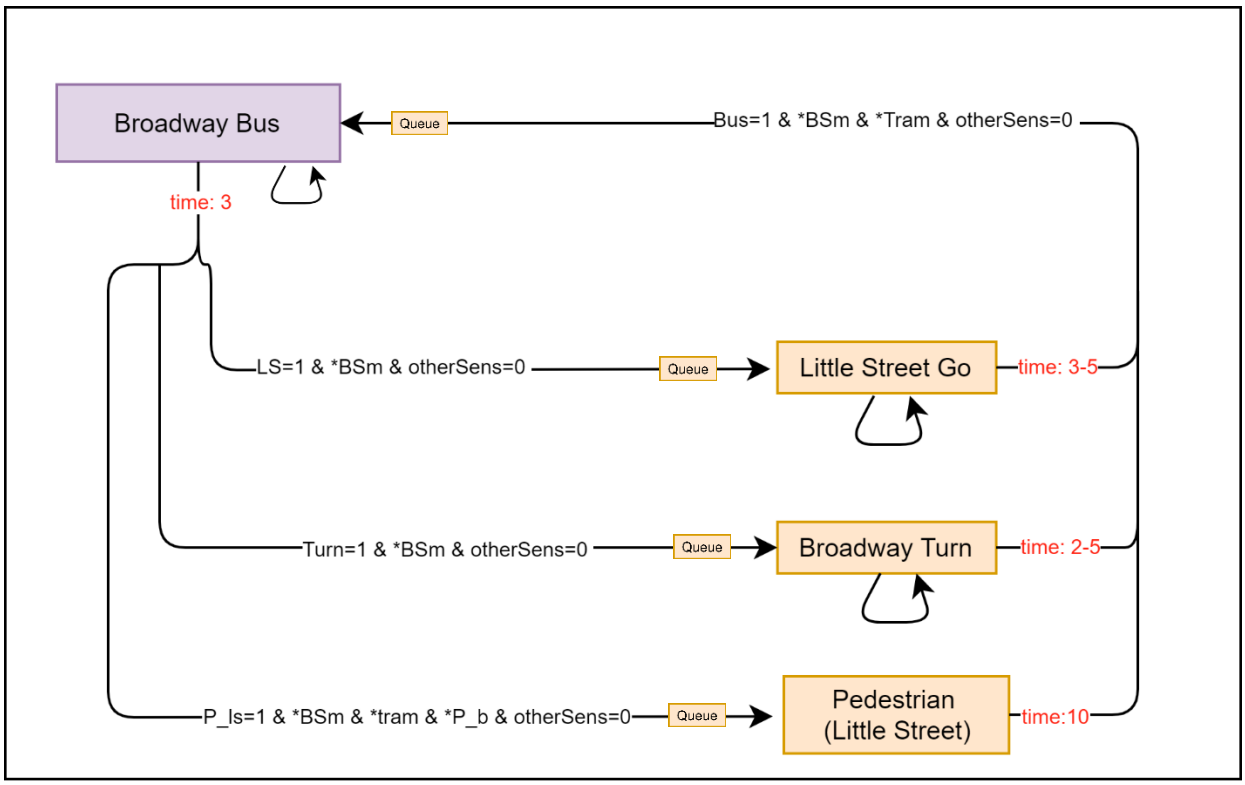


Figure 4 State Relation with Detail Conditions

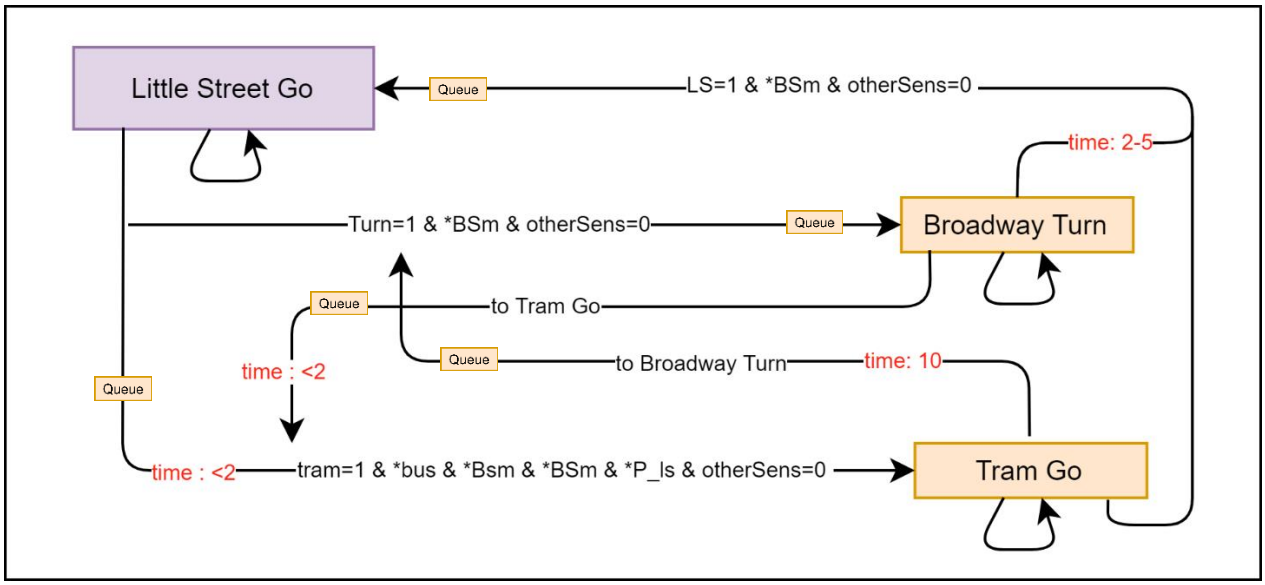


Figure 5 State Relation with Detail Conditions

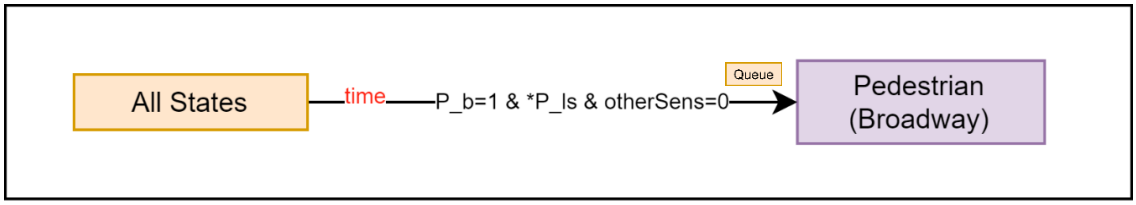


Figure 6 State Relation of Pedestrian Broadway with Condition

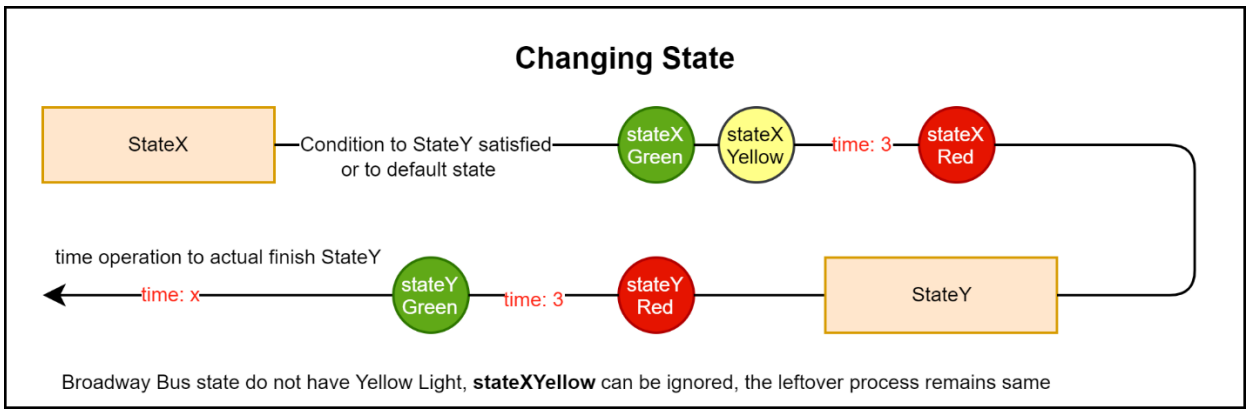


Figure 7 Changing State

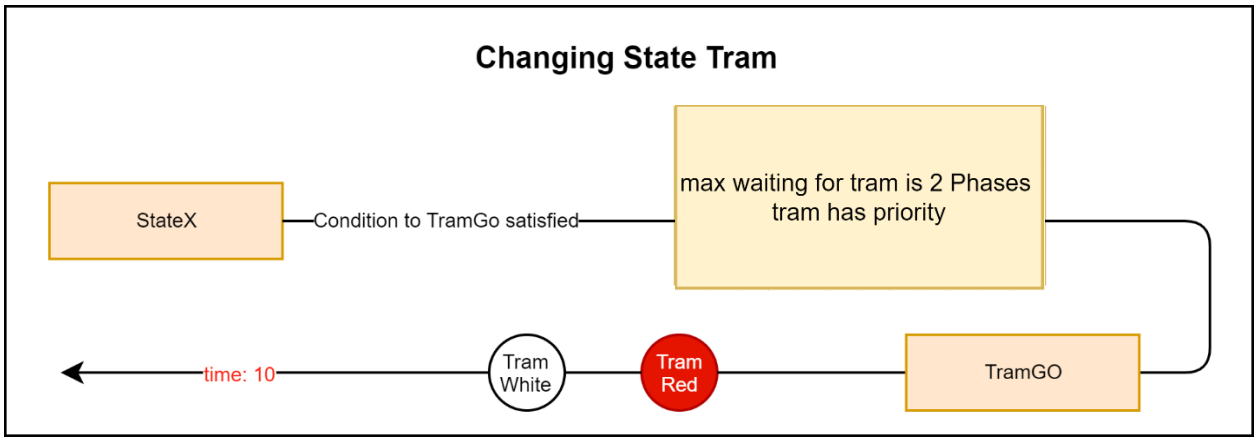


Figure 8 Changing State of Tram

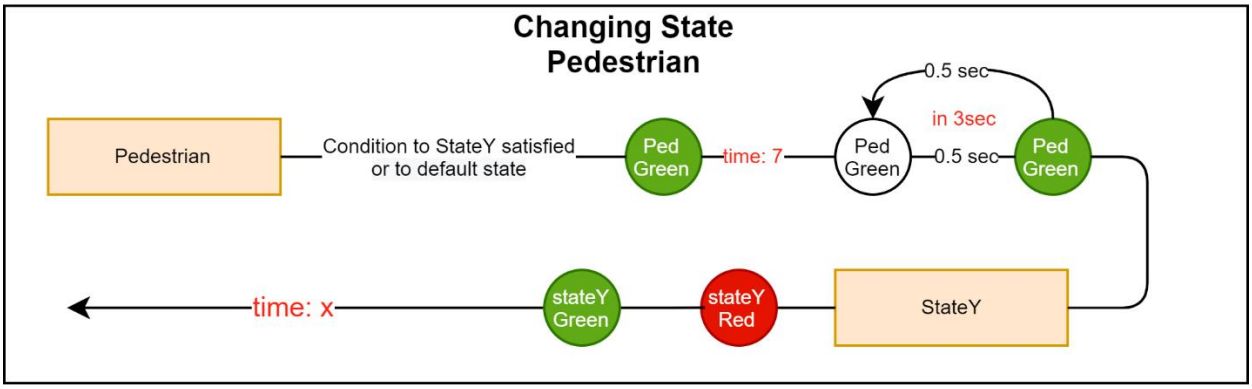


Figure 9 Changing State for Pedestrian

State Diagram Questions:

1) List the inputs, number of states, and outputs of your state diagrams.

Note:

- if another State currently is running, put the new State into IFOF Queue to process later.
- Tram has priority

State Diagram Name	Inputs	Outputs	Number of States
Broad Way Car Go	-South Main button -North Main button -Either Tram Buttons -Others not trigged	-North Main Green -South Main Green -Either Tram Light -Others are Red	1
Broad Way Turn	-Broad Way Turn button -Others not Trigged	-Turn LED Green -Others are Red	2
Little Street Turn	-Little Street button -Others not Trigged	-Little Street Green -Others are Red	3
Bus South Go	-Bus buttons trigged -Either Tram -Either South Main Go -Others not Trigged	-Bus LED Turn on -Either Tram LED -Either South Main Go -Others are Red	4
Tram Broad Way Go (has priority)	-Tram button Trigged -Either South Main button -Either North Main button -Either BUS -Others not Trigged	-Tram light on -Either South Main Go -Either North Main Go -Either Bus Status	5
Pedestrian	-Little Street Ped -South Ped -North Ped -Other not Trigged	-PED lights are all On -Others are Red	6

2) How does your state machine handle external inputs (i.e. sensor inputs)? Detail when and how they are handled.

ANS: **SPI** and **Interrupt** within MCP23S17 and Atmega328p are using to handler the external inputs.

In detail:

- **SPI-** MCP23s17 are using SPI to read and write data,
- **Interrupt-**When a button or buttons are pressed, INTA or INTB are Trigged (using interrupt within mcp23s17). As soon as buttons are pressed, Interrupt within Atmega328p notices the changes in portD2 and portD3 and handle in INT0_vect and INT1_vect perspective. To terminate the interrupt event in MCP23S17, simply read the INTCAP or GPIO using SPI.

3) How does your state machine generate external outputs (i.e. traffic light sequences, pedestrian sounds)? Detail when and how these are created.

There will be two timers to do all the jobs

- **First Timer (1KHz):** mainly for time counter and some repeatably actions such as pedestrian sound every 5 seconds (which does not consume many resources)
- **Second Timer (1MHz):** State Processing including lights, lcd.
- **SPI, I2c** will use CPU clock to generate

My system is quite straightforward, I process everything (I much as I can) in two timers (hardware) and avoid using main loop (software).

4) Create a timeline of the state transitions that occur for each of the sequences of events listed below. Assume that the system time period is set at 1 s, and prior to time 0 the system is in the hazard state. You MUST list state transitions according to your state diagrams. A vague description of what happens will not suffice for this question. (Note: Current Time column is time passed, not duration.)

Time (seconds)	Input Sensor Event	Duration before sensor clears (seconds)
0	S3 (bus)	2
4	S1 (BW turn)	2
6	S0 (BW southbound)	10
7	S6 (little)	4
8	S5 (tram)	5
10	S7 (pedestrian)	5

Current Time (seconds)	Input Sensor Event	States and transitions	Output(s)
0	S3 (bus)	BUS State	LED BUS ON
1		BUS State	
2		Bus extra waiting Time	
3	S5 (tram)	Tram State	Bus Led OFF Tram White Led ON
4 …		Tram State …	
7		Tram State	
8	S1 (BW turn)	waiting before turn Green	Tram White Led OFF Tram Red Led ON
9		waiting before turn Green	
10		waiting before turn Green	
11		Turn State	Broad Way Turn Led ON
12		Turn State	

13		yellow Turn State	Broad Way Turn Green OFF Broad Way Turn Yellow ON
14		yellow Turn State	
15		yellow Turn State	
16	S0 (BW southbound)	Waiting before Green	Broad Way Turn Yellow OFF Broad Way Turn Red ON
17		Waiting before Green	
18		Waiting before Green	
19 …		Broad Way Car Go State …	Broad Way Green ON
28		Broad Way Car Go State	
29		Yellow Broad Way Car Go State	Broad Way Green OFF Broad Way Yellow ON
30		Yellow Broad Way Car Go State	
31		Yellow Broad Way Car Go State	
32	S6 (little)	Waiting before Green	Broad Way Red ON
33		Waiting before Green	
34		Waiting before Green	
35		Little Street State	Little Street Green ON
36		Little Street State	
37		Little Street State	
38		Little Street State	
39		Yellow Little Street State	Little Street Green OFF Little Street Yellow On
40		Yellow Little Street State	
41		Yellow Little Street State	
42	S7 (pedestrian)	Waiting before Green	Little Street Yellow OFF Little Street Red ON
43		Waiting before Green	
44		Waiting before Green	
45 …		Pedestrian State …	Pedestrian Green ON
51		Pedestrian State	

52		Pedestrian State	Pedestrian Green FLASH + Siren
53		Pedestrian State	Pedestrian Green FLASH + Siren
53		Pedestrian State	Pedestrian Green FLASH + Siren
54			Pedestrian Green OFF Pedestrian Red ON

(Extend as necessary)

Implementation Questions:
5) What occurs in your main loop? Justify why each of these actions is in the main loop.

ANS: as my idea, no actions are needed to run in my main Loop to save CPU resources, I put my main into sleep_mode() when is not in use to reduce power consumption. However, core clock sometimes is used to do tasks (eg. SPI, i2C, calculation)

6) (a) Describe in detail how your timing system works. Does your system run at a regular interval? If so, what is the frequency of that interval?

ANS: the timing system or software clock is generated in the core CPU, this can run UP TO 16Mhz (regular interval) however, timing (frequency) is usually way less accurate than hardware and not flexible.

(b) Which hardware timers are you using, in which mode are they configured and how often (if at all) do they interrupt?

ANS:
Hardware built-in timers are available in Atmega328p

- **Timer 0-** will run at 1kHz which indicates that every interval will take 1 millisecond
- **Timer 1-** will run at 1Mhz which indicates that every interval will take 1 microsecond since this timer 2 has more actions to work on.
- They will not interrupt, two timers are running at the same time in Atmega328p

7) How many interrupts do you use? List each interrupt and describe its purpose is within your system.

Interrupt	Purpose
INT0_vect (PD2)	This interrupt is for tracking sensors (from 0 to 7) are trigged
INT1_vect (PD3)	This interrupt is for tracking sensors (8 and 9) are trigged

8) *Describe how your system identifies which, if any, of the buttons on the two port expanders are pressed. How often are these checked? How do you handle buttons being pressed on both port expanders simultaneously?*

ANS:

- There are two different Interrupt handlers in this particular system, first to handle the Expander A, particularly INTB (connected to PD2 ~ INT0_vect) because all buttons are allocated in GPIOB of Expander B
- Similar to Expander 2, only first 2 bits in GPIOB are used. INTB (connected to PD2 ~ INT1_vect) are used.
- Either INT0_vect or INT1_vect is trigged, so we can tell which is the one of which expander is pressed
- They check as soon as sensors are trigged
- Read values from CAPINT ram or GPIO ram to indicates which sensors are pressed

9) A busy loop is any piece of code that loops waiting for an event, such as a flag being made ready. Do you use busy loop? If yes describe each busy loop. (Check in non-obvious places, like SPI or TWI waiting for a flag to be ready on byte send.)

ANS: yes I am using a busy loop but for very very very short of time for each.

- State Handler that checks which sensors are pressed (similar to 10 if-else statement);
- A busy Loops for main **while(1){} ?**
- While loop for waiting a flag for Serial
- While loop until all characters are send in Serial
- While loop for waiting a flag in SPI
- While loop for I2C operations
- While loop until all characters are send in I2C
- While loop for ADC read

10) How do you generate the following timing events:

Timing Event	Generation
Single Time Period	Using potentiometer, ranging from 0v to 5v corresponding from 500milliseconds to 2000milliseconds
Speaker Tone Timings	- Not in Pedestrian State: Speaker Tone set at ~ 600Hz every 5 period of time lasting for 0.5 period of time - set frequency to 1500Hz, and play it at every 1 period of time (on for 0.5 period of time and off for 0.5 period of time)
Hazard Timing	10 Seconds – each required Led flashing at 1Hz
Pedestrian Flashing	Flashing green for 3 time periods in half duty cycle, For example: - *1 period of time = 1 second: flashing at 1Hz.*