**REQUIREMENT**

1. **Input:**  3 inputs ứng với 3 Sensor:
   1. South\_sensor: Khi có xe ở đường phía Nam (South Road) thì South sensor: logic dương

(Triển khai) Bấm và giữ button -> South sensor = 1. Nếu button không được bấm -> South sensor = 0

* 1. West\_sensor: Khi có xe ở đường phía Tây (West Road) thì West sensor: logic dương

(Triển khai) Bấm và giữ button -> West sensor = 1. Nếu button không được bấm -> West sensor = 0

* 1. Pedestrian\_sensor: Khi người đi bộ đi qua đường (ở bất kì hướng nào) thì pedestrian sensor: logic dương

(Triển khai): bấm và giữ button -> Pedestrian sensor = 1. Nếu button không được bấm -> Pedestrian sensor = 0

1. **Output**: Outputs ra các LED và LCD
   1. 6 LEDs: red, yellow, green cho đèn ở hướng South và West, ứng với **3 trạng thái:**
      1. **Go**: LED chuyển Green khi cho xe đi qua (trong tối thiểu 10s)
      2. **Warning**: LED chuyển Yellow (trong 5s)
      3. **Stop**: LED chuyển RED
   2. 3-color LED: Đèn cho người đi bộ với **3 trạng thái:**
      1. **Go**: LED chuyển White khi người đi bộ qua đường (trong tối thiểu 10s)
      2. **Warning**: (white) – red – off – red – off trong 5s (1s white, 1s red rồi 1s off rồi 1s red rồi 1s off).
      3. **Stop**: LED chuyển RED, người đi bộ dừng lại
   3. Hiển thị ra LCD thời gian đã chờ, ứng với đèn đang ở chế độ Go | Warning:
      1. Go: Hiển thị thời gian đã ở trạng thái Go (đơn vị: s, LCD cập nhật theo giây)
      2. Warning: Hiển thị thời gian đã ở trạng thái Warning (đơn vị: s, LCD cập nhật theo giây)

Khi tất cả các đèn đều ở trạng thái **Stop**, LCD hiển thị: 00

1. **Light sequence:**

**Yêu cầu:**

* Đèn giao thông ở các hướng khác nhau phải chuyển từ trạng thái Go sang Warning trước khi chuyển sang Stop. Khi một đèn ở trạng thái Go và Warning thì 2 đèn còn lại phải ở trạng thái Stop.
* **Thứ tự ưu tiên**: Được xác định bằng việc nhập EID

VD: West, South, Walk: nếu sensor ở các hướng cùng ở logic 1 thì sẽ ưu tiên cho xe hướng West di chuyển trước rồi đến South, rồi đến người đi bộ.

* **Nhường đường luân phiên:** Khi có nhiều sensor = 1, đèn các hướng sẽ sáng theo **thứ tự ưu tiên** và **nhường đường luân phiên**. VD ban đầu ưu tiên hướng West trước, sau đó là South và Walk (Pedestrian). Nếu West\_light Green thì sau đó nếu hướng South có xe, South\_light sẽ Green (bất kể hướng West có xe hay không).

**Các kịch bản:**

* 1. **Không button nào được ấn** (3 sensor đều bằng 0) -> 3 đèn cho 3 hướng ở trạng thái Safe Pattern\_0: Tất cả đều đỏ.

Khi 1 đèn trạng thái Go mà các sensor = 0 thì đèn đấy sẽ chuyển sang trạng thái Warning trước rồi mới chuyển về trạng thái Stop.

* 1. **Chỉ có 1 button được ấn/ giữ (input bằng 1)**: Đèn ở hướng đó sẽ theo Safe Pattern\_1:

Đèn tương ứng ở trạng thái Go trong 10s. Nếu sau 10s mà vẫn chỉ có button đấy được ấn thì tiếp tục ở trạng thái Go. Khi đó nếu có button khác được ấn thì lập tức chuyển trạng thái Warning (trong 5s) rồi sang trạng thái Stop

VD: Hành vi các đèn khi đèn hướng West đang Green, có xe ở hướng South và không có xe ở hướng West, vẫn không có người đi bộ (South sensor = 1, West sensor = 0, Pedestrian sensor vẫn bằng 0):

t = 0-: West\_light: Green, South\_light: Red, Pedestrian\_light : RED (đèn hướng West chuẩn bị chuyển Yellow đến t = 5).

t = 0-> 5: West\_light: Yellow, South\_light: Red, Pedestrian\_light : RED. LCD hiển thị 00 -> 01 -> 02 -> 03 -> 04-> 05

t = 5-> 15: West\_light: RED, South\_light: GREEN, Pedestrian\_light : RED. LCD hiển thị: 00-> 10

Giả sử vẫn có xe hướng South, không có xe hướng West và không có người đi bộ thì trạng thái ở t = 15 giữ nguyên. Đến khi t = 20, Pedestrian\_sensor = 1

t = 15-> 20: West\_light: RED, South\_light: GREEN, Pedestrian\_light : RED. LCD hiển thị: 10-> 15

t = 20-> 25: West\_light: RED, South\_light: Yellow, Pedestrian\_light : RED. LCD hiển thị: 00-> 05

t = 25: West\_light: RED, South\_light: RED, Pedestrian\_light : Green. LCD hiển thị: 00

* 1. **Có 2 button được ấn / giữ 1 lúc:** Sẽ tuân theo **Thứ tự ưu tiên, Nhường đường luân phiên**

VD: khi mới bắt đầu, đèn cả 3 hướng đều đỏ, có xe ở hướng South và có xe ở hướng West, không có người đi bộ (South sensor = 1, West sensor = 1, Pedestrian sensor = 0). Thứ tự ưu tiên: West, South, Walk

t = 0-: West\_light: RED, South\_light: RED, Pedestrian\_light : RED (đèn hướng West chuẩn bị chuyển xanh đến t = 10 do hướng West được ưu tiên)

t = 0-> 10: West\_light: GREEN, South\_light: RED, Pedestrian\_light : RED. LCD hiển thị 00 -> 10

Nếu ở giây thứ 10 không có xe hướng South và có xe hướng West (South\_sensor = 1, West\_sensor = 0) thì West\_light Green. Đến giây thứ 15, đồng thời có xe hướng South và West (South\_sensor = West\_sensor = 1) thì nhường đường xe hướng South **(nhường đường luân phiên)**

t = 10 -> 15: West\_light: GREEN, South\_light: Red, Pedestrian\_light : RED. LCD hiển thị 10 -> 15

t = 15 -> 20: West\_light: Yellow, South\_light: RED, Pedestrian\_light : RED. LCD hiển thị 00 -> 05

t = 20: West\_light: RED, South\_light: GREEN, Pedestrian\_light : RED. LCD hiển thị 00

* 1. **Có 3 button được ấn / giữ 1 lúc:** sẽ tuân theo **Thứ tự ưu tiên, Nhường đường luân phiên**

VD: Ban đầu khi chưa ấn button, đèn cả 3 hướng đều đỏ, có xe ở hướng South và có xe ở hướng West, không có người đi bộ (South sensor = 1, West sensor = 1, Pedestrian sensor = 0). Thứ tự ưu tiên: West, South, Walk

t = 0-: West\_light: RED, South\_light: RED, Pedestrian\_light : RED (đèn hướng West chuẩn bị chuyển xanh đến t = 10 do hướng West được ưu tiên)

t = 0-> 10: West\_light: GREEN, South\_light: RED, Pedestrian\_light : RED. LCD hiển thị 00 -> 10

Nếu vẫn có xe hướng South và có xe hướng West thì nhường đường cho xe hướng South do đèn West vừa xanh rồi. Đến giây thứ 15, đồng thời có xe hướng South và West (South\_sensor = West\_sensor = 1) thì nhường đường xe hướng South **(nhường đường luân phiên)**

t = 10 -> 15: West\_light: GREEN, South\_light: Red, Pedestrian\_light : RED. LCD hiển thị 10 -> 15

t = 15 -> 20: West\_light: Yellow, South\_light: RED, Pedestrian\_light : RED. LCD hiển thị 00 -> 05

t = 20 -> 30: West\_light: RED, South\_light: GREEN, Pedestrian\_light : RED. LCD hiển thị 00 -> 10

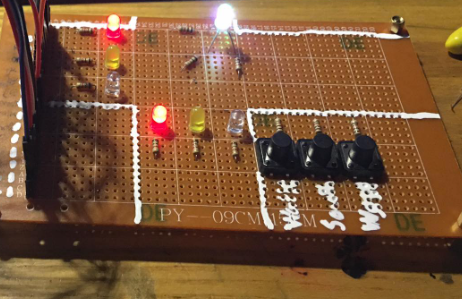
Nếu tại t = 30, West\_sensor = South\_sensor = Pedestrian\_sensor = 1, thì nhường đường người đi bộ do hướng West, South vừa xanh rồi **(nhường đường luân phiên)**

t = 30-> 35: West\_light: RED, South\_light: Yellow, Pedestrian\_light : RED. LCD hiển thị 00 -> 05

t = 35: West\_light: RED, South\_light: RED, Pedestrian\_light : Green. LCD hiển thị 00

1. **Lưu ý: Chỉ cho phép sử dụng thư viện sẵn cho LCD**

* Lắp mạch trên bo như hình vẽ bên dưới, kẻ lề, chia khối để dễ quan sát. **Chú ý chọn Pin cho khéo để đủ pin cho LCD (hoặc mua bộ mở rộng IO)**



*Hình 1: Mạch đèn giao thông (chưa có LCD) (Nguồn: N.T.G)*

* Triển khai các input, output bằng config thanh ghi, không dùng hàm có sẵn. Các bạn chia nhau ra làm nhiệm vụ này coi như ôn lại 1 lần config thanh ghi GPIO
* Yêu cầu thực hiện ngắt GPIO với 3 chân input ứng với 3 sensor để detect và xử lý khi button được bấm
  + Ngắt được thực hiện bằng cách khai báo, config thanh ghi. **KHÔNG** sử dụng thư viện ngắt có sẵn
  + Triển khai ngắt: Xem phần 1 của **TUTORIAL**

**TUTORIAL**

1. **Interrupt:**

* Mục đích: Để các bạn biết triển khai interrupt bằng config thanh ghi

Thông thường, interrupt có thể được tạo để tránh hỏi vòng. VD khi đổi trạng thái chân GPIO thì interrupt được gọi và xử lý, thay vì phải hỏi vòng while để kiểm tra xem khi nào trạng thái GPIO được thay đổi

Tuy nhiên, trong hệ thống này khi đổi trạng thái GPIO thì state của đèn tín hiệu chưa chắc đã thay đổi, lí do trình bày ở phần dưới. Do đó, khi triển khai, vẫn cần hỏi vòng trong chương trình chính, hoặc triển khai thêm ngắt event (ngắt khi có sự kiện. VD triển khai ngắt timer, đếm hết 10s hoặc đếm hết 5s thì quay về kiểm tra trạng thái GPIO để cập nhật trạng thái LED) để tránh hỏi vòng (Lưu ý: Để đơn giản, bài Lab không bắt buộc thực hiện ngắt event)

**Câu hỏi:** Nếu bỏ interrupt (ngắt cứng GPIO) đi, kiểm tra giá trị GPIO trong vòng while thì chương trình có chạy không?

* Lý do cần tạo ra các cờ: West\_pressed, South\_pressed, Walk\_pressed:

(VD: West\_pressed = 1 khi West button được ấn)

* + Hàm xử lý ngắt cần xử lý nhanh rồi quay về chương trình chính. (Các cờ được cập nhật trạng thái trong Hàm xử lý ngắt, sẽ là “trung gian” giao tiếp giữa ngắt với chương trình chính)
  + Hết thời gian đếm ngược (10s cho đèn xanh, 5s cho đèn vàng) **thì mới cập nhật** lại các đèn dựa trên button được bấm:

VD: Khi đèn hướng West đang Green mà có button được ấn (ngắt được gọi) thì hệ thống sẽ xử lý ngắt trước. **Nhưng** nếu chưa hết 10s thì phải chờ hết 10s mới được cập nhật trạng thái mới, tức là button được ấn hay không thì cũng chả để làm gì vì chưa hết 10s.

Do đó, giải quyết bằng các cờ. Làm như thế nào ? (Lưu ý: hàm xử lý ngắt sẽ chỉ cập nhật các cờ)

* Khi triển khai Interrupt (các ngắt ngoài) cho các GPIO:
  + 3 chân input tương ứng với 3 ngắt ngoài. Lưu ý, không phải chân GPIO nào làm input cũng được. Vì sao?
  + Đọc **Reference Manual** (197) để biết cách config thanh ghi ngắt
    - Chú ý: Cần config cả thanh ghi **NVIC**, tuy nhiên trong **Reference Manual** không chỉ cách config. Các bạn xem trong **Cortex M3 – Programming Manual** (118)
  + Khi config thanh ghi ngắt, chú ý rằng mình sẽ **detect cả sườn lên và sườn xuống** của tín hiệu tại chân GPIO. (Do khi giữ button thì sensor =1, nhả ra thì sensor = 0. VD, nếu mình chỉ detect sườn lên của West\_sensor, cờ West\_pressed = 1 khi mình nhấn button West. Nhưng khi mình nhả button, nhẽ ra West\_pressed = 0 nhưng cái này không được cập nhật)

1. **FSM**

* Có 2 hướng làm FSM:
  + 1 sơ đồ FSM cho toàn bộ hệ thông, điều khiển cả 3 đèn 1 lúc. Cách này mình survey thấy nhiều nhóm đã làm thành công. Đặc điểm là dễ kiểm soát và code (vẽ đúng là code đúng). Tuy nhiên, cách này rất nhiều state và các nhóm dễ bỏ qua các state. Cách làm đã được hướng dẫn rất kỹ tại: <https://users.ece.utexas.edu/%7Evalvano/Volume1/IntroToEmbSys/Ch5_FiniteStateMachines.html>
  + 3 sơ đồ FSM cho 3 đèn West\_light, South\_light, Pedestrian\_light:
    - 3 sơ đồ giao tiếp với nhau qua các tín hiệu enable: W\_en (West enable), S\_en (South enable), Wa\_en (Walk enable). VD khi đèn West\_light RED, W\_en = 1: cho phép các hướng South và Pedestrian
    - Cách này đặc điểm là dễ vẽ, dễ quan sát. Tuy nhiên cần bước tiền xử lý đầu vào để tính hết các kịch bản **(thứ tự ưu tiên, nhường đường luân phiên** trong **REQUIREMENT).**
* Hướng dẫn làm theo cách 3 sơ đồ FSM:
  + Phân rã hệ thống: (Coi như đã triển khai được ngắt ngoại cho 3 button đầu vào)
    - Khối xử lý đầu vào: Tính đến các kịch bản trong **REQUIREMENT**, trả về: thực hiện theo Input nào (chuẩn bị cho xe hướng nào di chuyển).

VD khi ấn cả 2 button thì thực hiện theo button nào.

* + - Khối FSM: Thực hiện sơ đồ trạng thái ứng với 3 đầu vào: Car\_S, Car\_W và Walk
  + Khối xử lý đầu vào: TUÂN THỦ THEO CÁC KỊCH BẢN ĐỀ RA TRONG **REQUIREMENT**
    - Input: 3 cờ: West\_pressed, South\_pressed, Walk\_pressed
      * West\_pressed = 1 khi button ứng với West\_sensor được ấn
      * West\_pressed = 0 khi button ứng với West\_sensor được thả
      * Tương tự với South\_pressed và Walk\_pressed
      * Việc cập nhật 3 cờ này: Được xử lý trong hàm xử lý ngắt!!
    - Output: Car\_S, Car\_W, Walk
      * Car\_S = 1: Sẵn sàng cho xe hướng South di chuyển. Tương tự với Car\_W và Walk
      * Chú ý: khi Car\_S = 1 thì Car\_W = 0 và Walk = 0
    - VD: Ban đầu, West\_pressed = 1, South\_pressed = 1, Walk\_pressed = 0 thì cho xe hướng West di chuyển **(Thứ tự ưu tiên).** Do đó, Car\_W = 1, Car\_S = Walk = 0
      * Lúc sau (thời điểm t = 15 theo **kịch bản c ở REQUIREMENT**) thì dù West\_pressed = 1, South\_pressed = 1 nhưng do West\_light GREEN rồi nên nhường xe hướng South chuẩn bị di chuyển

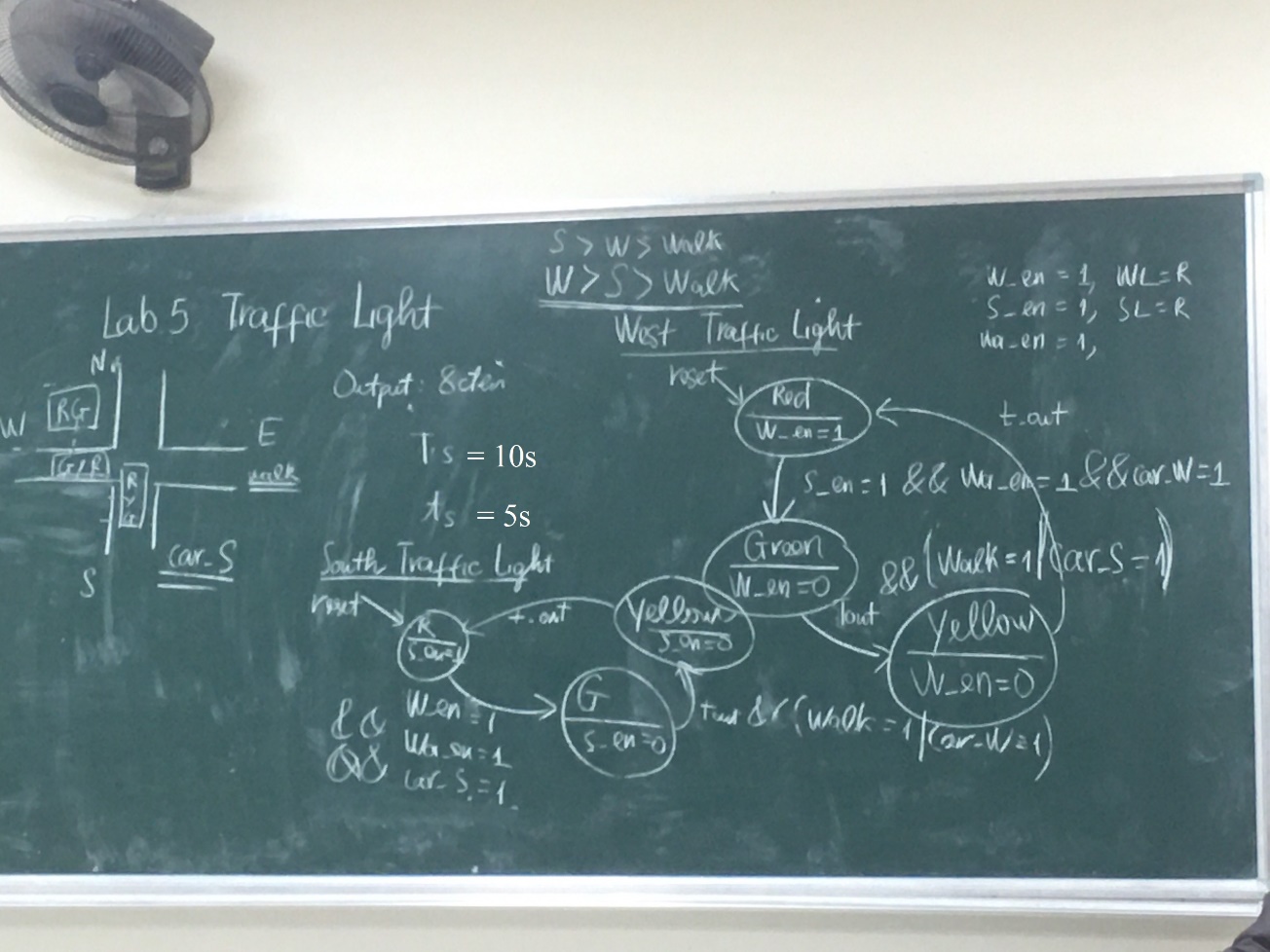
=> Lúc này, Car\_S = 1, Car\_W = Walk = 0

*(!! Cần thêm biến để check xem các đèn vừa GREEN hay chưa)*

* + Khối FSM:
    - Input: 3 tín hiệu: Car\_S, Car\_W và Walk
    - Output: Trạng thái của các đèn và các tín hiệu enable
    - Sơ đồ trạng thái tham khảo: (West\_light và South\_light)

*(chú ý: Output tại các state ngoài các tín hiệu enable, còn* ***output ra LCD****)*

A chalkboard with writing on it

Description automatically generated*Hình 2: Sơ đồ trạng thái FSM cho West light và South* 

**Lưu ý cuối cùng: Các bạn phải tự code và hiểu code. Nếu không rất khó để bảo vệ thành công**

**Thi cuối kỳ: Ngoài Final Lab trên, sẽ hỏi cả những kiến thức đã học trên lớp. Cụ thể mình sẽ thông báo sau. Cố gắng lên nhá sắp xong rồi**

Chúc các bạn A+ Vi xử lý