EE 319K
Introduction to Embedded Systems

Lecture 6: Finite State Machines, Data Structures

Announcements

- Feedback survey
  - On Blackboard
- Homework 5
  - Arrays and functions
  - Due next Monday
- Lab 4
  - Next Tuesday/Wednesday
  - Traffic light controller
    - Finite State Machine (FSM)

Agenda

- Recap
  - Pointers and indexed addressing
  - Functional debugging
  - Timer
  - Arrays and strings
- Outline
  - Data structures
  - Finite State Machines (FSMs)

Finite State Machines - FSM

Software abstraction

- define a problem with a set of basic abstract principles
- separate policies mechanisms

Finite **S**tate **M**achine (FSM.)

- inputs, outputs, states, and state transitions
- state graph defines relationships of inputs and outputs

The three advantages of this abstraction are

1. it can be faster to develop
2. it is easier to debug (prove correct) and
3. it is easier to change

What is a state?

- Description of current conditions

What is a state graph?

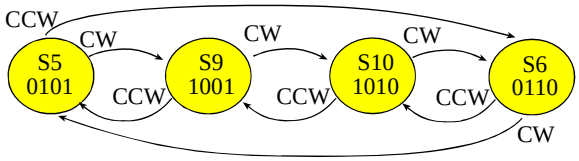
- Graphical interconnection between states

What is a controller?

- Software that inputs, outputs, changes state
- Accesses the state graph

Moore FSM

- Output value depends only on the current state, and
- Inputs affect the state transitions
- Significance is being in a state
- Input: when to change state
- Output: how to be in that state



Ramesh Yerraballi

6-5

Moore FSM Loop

mooreloop

Find o/p based on Current-state and write it out (or perform it)

Find Delay associated with this state and wait for that long

Get i/p

Change state based on Current-state and i/p

bra mooreloop

Moore FSM Loop

mooreloop

Find o/p based on Current-state and write it out (or perform it)

Find Delay associated with this state and wait for that long

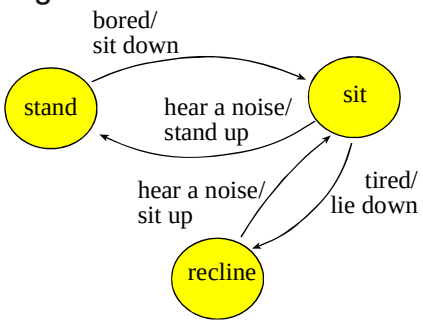
Get i/p

Change state based on Current-state and i/p

bra mooreloop

Mealy FSM

- Output value depends on input(s) and current state
- Inputs affect the state transitions
- Significance is the state transition
- Input: when to change state
- Output: how to change state



Ramesh Yerraballi

6-6

Mealy FSM Loop

Mealyloop

Find Delay associated with this state and wait for that long

Get i/p

Find o/p based on Current-state and i/p; Write it out (or perform it)

Change state based on Current-state and i/p

bra Mealyloop

Implementation of FSM

data structure embodies the FSM

- multiple identically-structured nodes
- statically-allocated fixed-size linked structures
- one-to-one mapping FSM state graph and linked structure
- one structure for each state

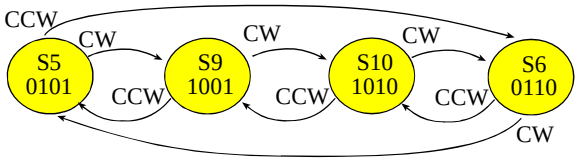
linked structure

- pointer (or link) to other nodes (define next states)

table structure

- indices to other nodes (define next states)

Stepper Motor FSM



CCW: PH6=1 & PH7=0
CW: PH6=0 & PH7=1
Outputs connected to PP3,PP2,PP1,PP0 Control Stepper Motor
For example, when in state S9 the output is 1001 which is written to the pins
PP3(1), PP2(0), PP1(0) and PP0(1)

Stepper Motor Controller

** RAM variables*

```
    org  $0800
Pt  rmb  2  ;pointer to current state
```

** ROM constants*

```
     org  $4000
out  equ  0    ;8-bit output
wait equ  1    ;time to wait
next equ  3    ;4 pointers to next state
S5   fcb $05   ;4-bit output
     fdb 4000
     fdb S5,S9,S6,S5 ;next for each in
S6   fcb $06
     fdb 4000
     fdb S6,S5,S10,S6
S10  fcb $0A
     fdb 4000
     fdb S10,S6,S9,S10
S9   fcb $09
     fdb 4000
     fdb S9,S10,S5,S9
```

** ROM program*

```
Main lds  #$4000
     bsr  Timer_Init  ; activate TCNT
     bclr DDRH,#$C0   ;PH6,7 input
     bset DDRP,#$0F   ;PP3-0 output
     movb #$05,PTP   ; initial output
     movw #S5,Pt     ; initial state
loop ldx  Pt
     movb out,X,PTP   ; step motor
     ldd  wait,X
     bsr  Timer_Wait  ; wait specified time
     ldaa PTH         ; read inputs
     anda #$C0        ; just CCW,CW
                      ; 0,40,80,C0
     lsra             ; 0,20,40,60
     lsra             ; 0,10,20,30
     lsra             ; 0,08,10,18
     lsra             ; 0,04,08,0C
     lsra             ; 0,02,04,06
     leax next,X      ; list of pointers
     ldx  A,X         ; next depends on in
     stx  Pt
     bra  loop
```