Birla Institute of Technology and Science

ELEVATOR CONTROL SYSTEM

Microprocessor, Programming & Interfacing

Submitted By:
GROUP NO. 60
PROJECT 26

SHANTOM KUMAR BORAH	**2016A3B50114P**
ADITYA AJAY GUPTA	**2016A3PS0132P**
JOY PARIKH	**2016A3PS0136P**
MAYANK AGGARWAL	**2016A3PS0137P**

Design Specifications

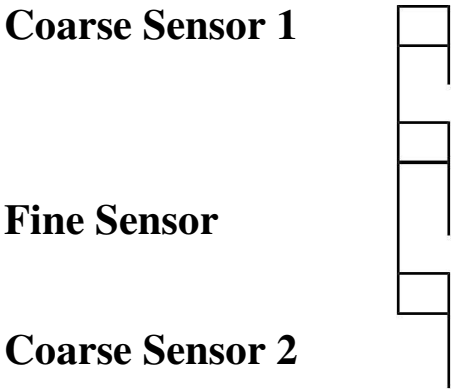
P26: System to be designed: Elevator control

System Requirements

- The elevator operates along 3 floors.
- When not in use the elevator is always on the ground floor.
- The elevator can be called by pressing any one of two buttons available on each floor. o One button is up and the other is down.
- Whether the elevator stops at the floor or not depends on the direction in which the lift moves. For eg., if the lift is moving in upward direction and the person on say the 2^{nd} floor presses the down button; the lift will not stop in the current journey. When the lift reaches the 3^{rd} floor and starts moving down then the lift will stop at the 2^{nd} floor.
- At every floor there is a 7-segement display that indicates the floor in which the lift is right now. The display can be any value from 0 - 3. '0' indicates the ground floor.
- Inside the lift - buttons are available for floor selection.
- The floor towards which the lift is moving is also displayed within the lift.
- Doors to the lift open and close automatically.
- When the lift reaches any floor where it has to stop it opens automatically, and it closes when a button called "Door Close" is pressed. Lift does not move until the door is closed.
- System runs from a standard power inlet.

System Specifications

- An Electro-magnetic system is used for open and close of the door. You just need to provide the on/off control.
- A heavy duty servo motor is used for lift movement. You just need to provide the input to the driver circuit.
- The inputs are direction (up/down) and a PWM input which control the speed at which the lift moves. The duty cycle can vary from 20% to 60%.
- The frequency of the PWM signal is 20 Hz.
- For detecting whether the lift has reached a floor, the system has a set of three sensors – two 'coarse' sensors and a 'fine' sensor. All the sensors are contact switches i.e. when the lift reaches the point where the sensors are placed, the contact switch gets pushed in. Output of contact switches are low when closed and high otherwise. The sensor arrangement is represented in the fig below



- On the ground floor – only Coarse Sensor1 and Fine Sensor will be available. On the 3rd floor only Coarse Sensor 2 and the Fine Sensor will be available.

o When the lift starts at the ground floor it starts at a low speed gradually accelerating to the maximum speed. It should operate at maximum speed when it reaches ‘Coarse Sensor 1”. As the lift moves up if it has to stop at floor ‘1’, when Coarse Sensor 2 is detected at that floor the lift starts moving at a low speed until it can stop when it reaches Fine sensor. When it starts again it moves at low speeds and reaches the maximum possible speed when it reaches the fine sensor. The same is done in the reverse direction with the appropriate sensors.

o Speed at which the lift moves is proportional to the duty cycle. For acceleration, duty cycle has to be gradually increased from 20 % to 60 %. And for deceleration, the duty cycle reduced from 60 % to 20 %. The increase is in steps of 20 %

o A 7447 chip (BCD to seven segment converter) is used for driving the 7-segment displays. o 7447 takes a 4-bit BCD value and converts into the corresponding 7-segement equivalent.

Assumptions made while implementing the design

- Coarse and fine sensors have been assumed to be SPDT switches and have to be manually pressed during simulation.
- The keypad for selecting the floor in the elevator also consists of SPDT switches.
- Door closes if door close button is pressed or 5 seconds after opening.
- Door control is monitored by using an LED. LED is on for closed door and off for open door.
- The heavy duty motor for lift movement is modelled as an oscilloscope and an LED. LED indicates direction: on for up and off for down and PWM indicates speed.
- 20% duty cycle is full stop, 40% is slow speed, 60% is fast speed.
- Contact with all sensors of a floor is lost before contact with sensors of the next floor is made.

Components Used

- INTEL 8086 Microprocessor –1
- Octal D-Type latch (74LS373) – 3
- Bidirectional Buffer (74LS245) – 2
- Unidirectional Buffer (74LS244) – 1
- INTEL 8255A Programmable Peripheral Interface chip – 2
- 2732 ROM 4KB – 2
- 6116 RAM 2KB – 2
- 8253A Programmable Interval Timer –1
- 7-Segment Display –5
- 7447 BCD to 7-segment display convertor –2
- LEDs – 12
- 74LS138 3-8 Decoder – 1
- Interactive SPDT Switch (Momentary action) – 12
- Interactive SPDT Switch (Latched action) – 10
- 7407 Hex NOT gate – 1
- 7432 Quad 2 input OR Gate – 3
- 7427 Triple 3 input NOR Gate – 1

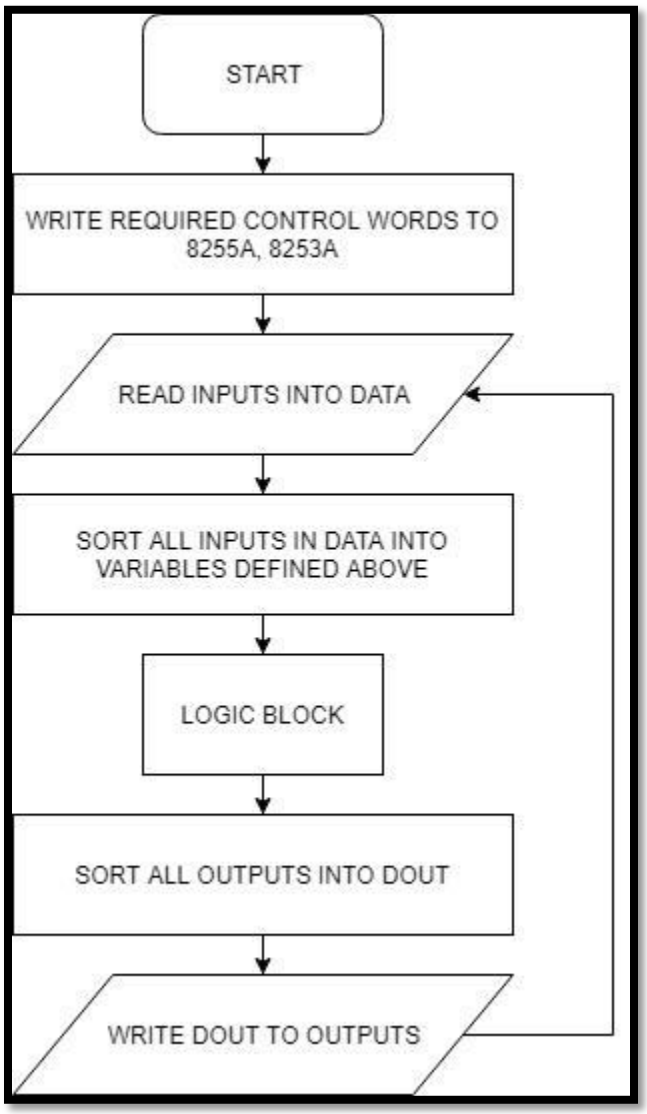
Address and I/O Mapping

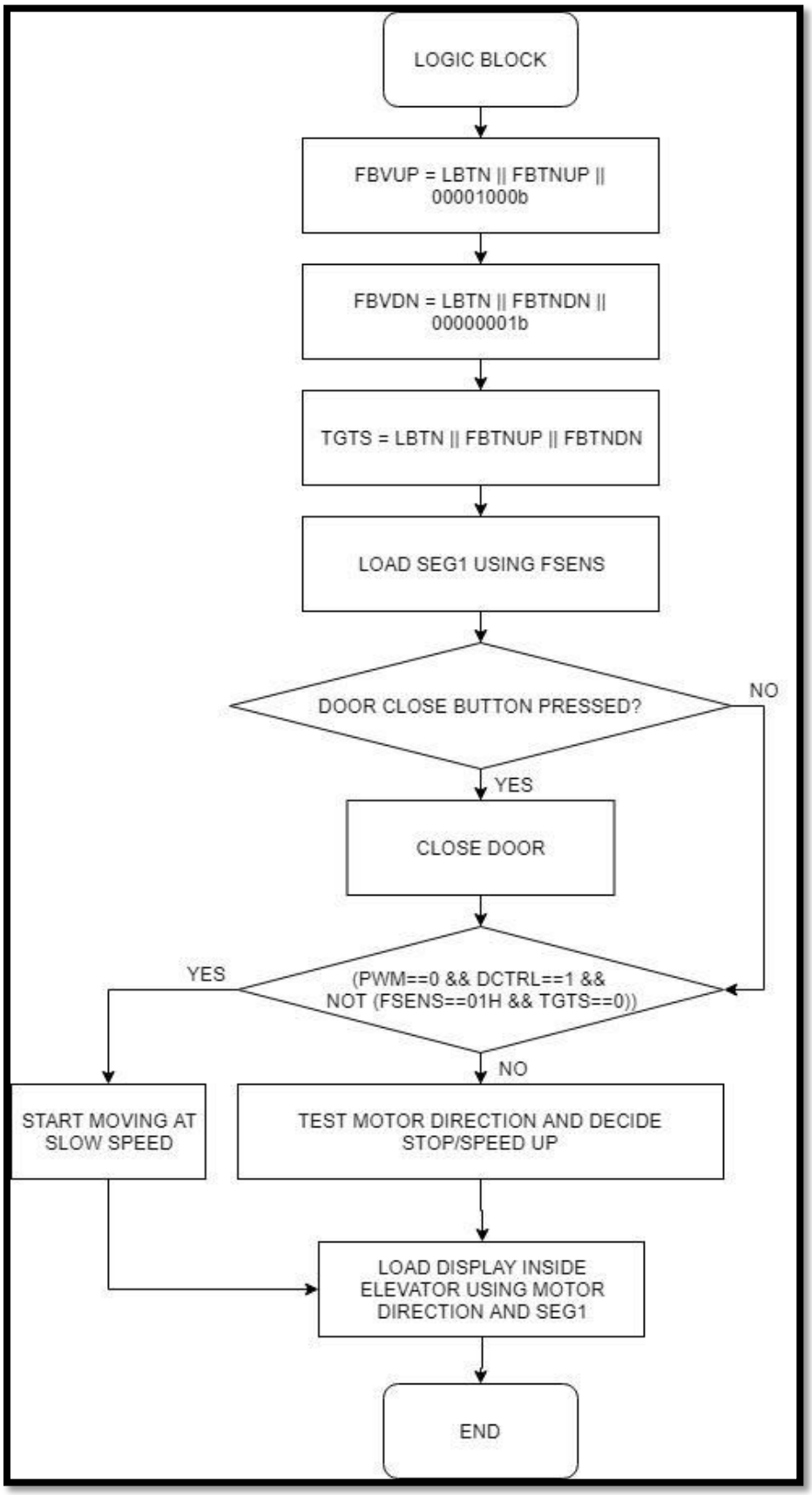
- ROM : 0x00000 to 0x01FFF (8 kB distributed in even and odd banks)
- RAM : 0x02000 to 0x02FFF (4 kB distributed in even and odd banks)
- 8255A (for inputs): 0x03000, 0x03002, 0x03004, 0x03006
- 8255A (for outputs): 0x04000, 0x04002, 0x04004, 0x04006
- 8253A: 0x05000, 0x05002, 0x05004, 0x05006

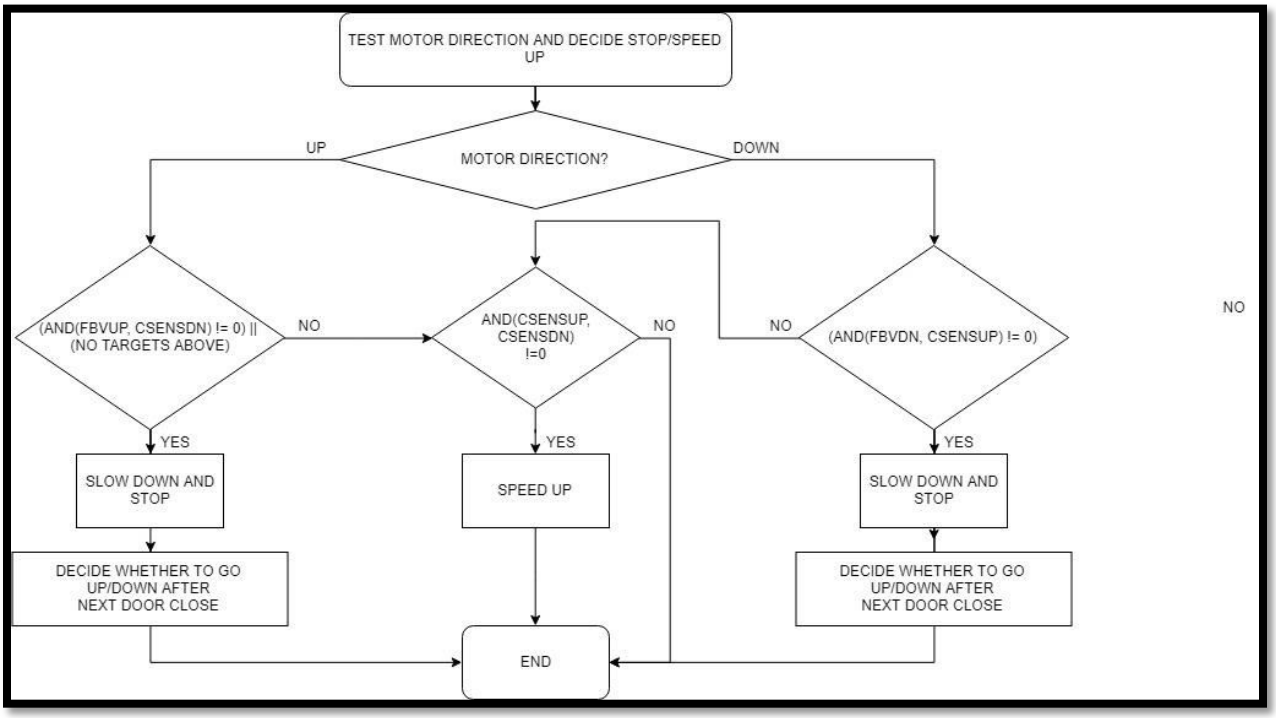
List of Variables

DATA	Input to be read from 8255
DOUT	Output to be written to 8255
LBTN	Internal buttons for lift destinations
FBTNUP	Up buttons for all floors
FBTNDN	Down buttons for all floors
FSENS	Fine sensors
CSENSUP	Upper coarse sensors
CSENSDN	Lower coarse sensors
DBTN	Door close button
FBVUP	Floors to stop at while moving up
FBVDN	Floors to stop at while moving down
TGTS	Floors to visit
MOTOR	Motor Direction
DCTRL	Door close/open control
PWM	PWM pulse to be set (0 – 20%/1 – 40%/2 – 60%)
PWM0	Current PWM pulse
SEG1	Value to load in display on floors
SEG2	Value to load in display in elevator

Flow chart of the software




LOGIC BLOCK
FBVUP = LBTN || FBTNUP || 00001000b
FBVDN = LBTN || FBTNDN || 00000001b
TGTS = LBTN || FBTNUP || FBTNDN
LOAD SEG1 USING FSENS
DOOR CLOSE BUTTON PRESSED?
NO
YES
CLOSE DOOR
(PWM==0 && DCTRL==1 && NOT (FSENS==01H && TGTS==0))
YES
NO
START MOVING AT SLOW SPEED
TEST MOTOR DIRECTION AND DECIDE STOP/SPEED UP
LOAD DISPLAY INSIDE ELEVATOR USING MOTOR DIRECTION AND SEG1
END


TEST MOTOR DIRECTION AND DECIDE STOP/SPEED UP
MOTOR DIRECTION?
UP
DOWN
(AND(FBVUP, CSENSDN) != 0) || (NO TARGETS ABOVE)
NO
AND(CSENSUP, CSENSDN) !=0
NO
NO
(AND(FBVDN, CSENSUP) != 0)
NO
YES
YES
YES
SLOW DOWN AND STOP
SPEED UP
SLOW DOWN AND STOP
DECIDE WHETHER TO GO UP/DOWN AFTER NEXT DOOR CLOSE
DECIDE WHETHER TO GO UP/DOWN AFTER NEXT DOOR CLOSE
END

Code (as compiled in emu8086)

```
#MAKE_BIN#

#LOAD_SEGMENT=FFFFH#
#LOAD_OFFSET=0000H#

#CS=0000H#
#IP=0000H#

#DS=0000H#
#ES=0000H#

#SS=0000H#
#SP=FFFEH#

#AX=0000H#
#BX=0000H#
#CX=0000H#
#DX=0000H#
#SI=0000H#
#DI=0000H#
#BP=0000H#

;DATA
      JMP    START
```

```
DB     1021 DUP(0)

    PORTINA         EQU 3000H
        PORTINB             EQU 3002H
        PORTINC             EQU 3004H
        CREGIN          EQU 3006H

        PORTOUTA    EQU 4000H
        PORTOUTB    EQU 4002H
        PORTOUTC    EQU 4004H
        CREGOUT             EQU 4006H

        CLKA            EQU 5000H
        CLKB            EQU 5002H
        CLKC            EQU 5004H
        CREGCLK     EQU 5006H

        DATA            DB 3 DUP(0FFH)
        DOUT            DB 3 DUP(00H)

        LBTN            DB 00H
        FBTNUP          DB 00H
        FBTNDN          DB 00H
        FSENS           DB 00H
        CSENSUP             DB 00H
        CSENSDN             DB 00H
```

```
        DBTN        DB 00H
        DSENS       DB 00H

        FBVUP       DB 00H
        FBVDN       DB 00H
        TGTS        DB 00H

        MOTOR       DB 00H
        DCTRL       DB 00H
        PWM             DB 00H
        PWM0        DB 00H
        SEG1        DB 00H
        SEG2        DB 00H

;CODE
START:  MOV AX,0200H
    MOV DS,AX
    MOV ES,AX
    MOV SS,AX
    MOV SP,0FFEH
        ;PORT INITIALIZE
        MOV AL, 9BH
        MOV DX, CREGIN
        OUT DX, AL
        MOV AL, 80H
        MOV DX, CREGOUT
```

```
        OUT DX, AL
        ;PWM INITIALIZE
        MOV AL, 00010100B
        MOV DX, CREGCLK
        OUT DX, AL
        MOV DX, CLKA
        MOV AL, 04H
        OUT DX, AL
        MOV AL, 01010010B
        MOV DX, CREGCLK
        OUT DX, AL
        MOV DX, CLKB
        MOV AL, PWM
        INC AL
        OUT DX, AL
        MOV AL, 10011000B
        MOV DX, CREGCLK
        OUT DX, AL

;LOOP
X1: ;INPUT
        MOV DX, PORTINA
        IN    AL, DX
        LEA BX, DATA
        MOV [BX], AL
        INC BX
```

```
        MOV DX, PORTINB
        IN    AL, DX
        MOV [BX], AL
        INC BX
        MOV DX, PORTINC
        IN    AL, DX
        MOV [BX], AL

;SORT OUT THE INPUTS

        ;LOAD DATA + 0
        LEA SI, DATA
        MOV AL, [SI]
        NOT AL

        ;DSENS
        MOV BL, 10000000B
        AND BL, AL
        MOV CL, 1
        ROL BL, CL
        LEA DI, DSENS
        MOV [DI], BL

        ;DBTN
        MOV BL, 01000000B
        AND BL, AL
```

```
MOV CL, 2
ROL BL, CL
LEA DI, DBTN
MOV [DI], BL

;CSENSDN
MOV BL, 00111000B
AND BL, AL
MOV CL, 2
ROR BL, CL
LEA DI, CSENSDN
MOV [DI], BL

;CSENSUP
MOV BL, 00000111B
AND BL, AL
LEA DI, CSENSUP
MOV [DI], BL

;LOAD DATA + 1
INC SI
MOV AL, [SI]
NOT AL

;LBTN
MOV BL, 11110000B
```

```
AND BL, AL
MOV CL, 4
ROR BL, CL
LEA DI, LBTN
MOV DL, [DI]
OR   BL, DL
MOV [DI], BL

;FSENS
MOV BL, 00001111B
AND BL, AL
LEA DI, FSENS
MOV [DI], BL

;LOAD DATA + 2
INC SI
MOV AL, [SI]
NOT AL

;FBTNDN
MOV BL, 00111000B
AND BL, AL
MOV CL, 2
ROR BL, CL
LEA DI, FBTNDN
MOV DL, [DI]
```

```
        OR   BL, DL
        MOV [DI], BL

        ;FBTNUP
        MOV BL, 00000111B
        AND BL, AL
        LEA DI, FBTNUP
        MOV DL, [DI]
        OR   BL, DL
        MOV DL, [DI]
        OR   BL, DL
        MOV [DI], BL

;LOGIC

        ;FBVUP
        LEA SI, LBTN
        MOV AL, [SI]
        LEA SI, FBTNUP
        MOV AH, [SI]
        OR   AL, AH
        OR   AL, 00001000B
        LEA DI, FBVUP
        MOV [DI], AL

        ;FBVDN
```

```
LEA SI, LBTN
MOV AL, [SI]
LEA SI, FBTNDN
MOV AH, [SI]
OR   AL, AH
OR   AL, 00000001B
LEA DI, FBVDN
MOV [DI], AL

;TGTS
MOV BL, TGTS
LEA SI, LBTN
MOV AL, [SI]
LEA SI, FBTNUP
MOV AH, [SI]
OR   AL, AH
LEA SI, FBTNDN
MOV AH, [SI]
OR   AL, AH
LEA DI, TGTS
MOV [DI], AL
CMP BL, 0
JNZ SLEEP
CMP AL, 0
JZ  SLEEP
MOV AL, FSENS
```

```
CMP AL, 01H
JNZ SLEEP
MOV DX, CLKC
MOV AL, 32H
OUT DX, AL
SLEEP:

;SEG1
LEA DI, SEG1
MOV AL, FSENS
CMP AL, 00H
JZ DEND
CMP AL, 01H
JZ D0
CMP AL, 02H
JZ D1
CMP AL, 04H
JZ D2
CMP AL, 08H
JZ D3
D0:  MOV BL, 0
     MOV [DI], BL
     JMP DEND
D1:  MOV BL, 1
     MOV [DI], BL
     JMP DEND
```

```
D2:   MOV BL, 2
      MOV [DI], BL
      JMP DEND
D3:   MOV BL, 3
      MOV [DI], BL
      JMP DEND
DEND:

;DCTRL
LEA DI, DCTRL
MOV AL, [DI]
MOV AH, DBTN
OR   AL, AH
MOV [DI], AL

;STARTUP DECIDE
MOV AL, PWM
MOV AH, 0
CMP AL, AH
JNZ NOSTART
MOV AL, DSENS
MOV AH, 0
CMP AL, AH
JZ NOSTART
MOV AL, DCTRL
MOV AH, 0
```

```
CMP AL, AH
JZ NOSTART
MOV AL, FSENS
MOV AH, 01H
CMP AL, AH
JNZ DOSTART
MOV AL, TGTS
MOV AH, 00H
CMP AL, AH
JNZ DOSTART
JMP NOSTART
DOSTART:    CALL DIRECT
            CALL SLOW
            JMP AEND
NOSTART:

;TEST MOTOR DIRECTION AND DECIDE STOP/SPEED UP
;JUST STARTING?

LEA SI, MOTOR
MOV AL, [SI]
MOV AH, 0
CMP AH, AL
JNZ A1
JZ  A2
```

```
A1:  ;STOP?
     MOV AL, FBVUP
     MOV AH, CSENSDN
     AND AL, AH
     MOV AH, 0
     CMP AL, AH
     JZ A3
     A3X:
     CALL STOP
     JMP AEND
     A3:          ;NO TARGETS ABOVE?
     MOV AL, CSENSDN
     MOV CL, 1
     ROL AL, CL
     DEC AL
     MOV AH, TGTS
     CMP AH, AL
     JLE A3X
     ;SPEED UP?
     MOV AL, CSENSUP
     MOV AH, CSENSDN
     MOV BL, FSENS
     AND BL, 00000001B
     OR   AH, BL
     AND AL, AH
     MOV AH, 0
```

```
        CMP AL, AH
        JNZ A5
        CALL FAST
        A5:
        JMP AEND

A2:     ;STOP?
        MOV AL, FBVDN
        MOV AH, CSENSUP
        AND AL, AH
        MOV AH, 0
        CMP AL, AH
        JZ A4
        CALL STOP
        JMP AEND
        A4:
        ;SPEED UP?
        MOV AL, CSENSDN
        MOV AH, CSENSUP
        MOV BL, FSENS
        AND BL, 00001000B
        OR   AH, BL
        AND AL, AH
        MOV AH, 0
        CMP AL, AH
        JNZ A6
```

```
        CALL FAST
        A6:
        JMP AEND
AEND:

;SEG2
LEA SI, MOTOR
LEA DI, SEG2
MOV BL, SEG1
MOV AL, [SI]
MOV AH, 0
CMP AH, AL
JNZ S1
JZ  S2
S1:   CMP BL, 3
        JZ S3
        INC BL
        S3:
        JMP SEND
S2:   CMP BL, 0
        JZ S4
        DEC BL
        S4:
        JMP SEND
SEND:
MOV [DI], BL
```

```
;SORT OUT THE OUTPUTS

    ;LOAD DOUT + 0
    LEA DI, DOUT
    MOV AL, FBTNUP
    MOV BL, FBTNDN
    MOV CL, 2
    ROL BL, CL
    OR   AL, BL
    MOV BL, DCTRL
    MOV CL, 2
    ROR BL, CL
    OR   AL, BL
    MOV BL, MOTOR
    MOV CL, 1
    ROR BL, CL
    OR   AL, BL
    MOV [DI], AL

    ;LOAD DOUT + 1
    INC DI
    MOV AL, LBTN
    MOV BL, SEG1
    MOV CL, 4
    ROL BL, CL
```

```
        OR   AL, BL
        MOV BL, SEG2
        MOV CL, 2
        ROR BL, CL
        OR   AL, BL
        MOV [DI], AL

    ;OUTPUT
        LEA BX, DOUT
        MOV AL, [BX]
        MOV DX, PORTOUTA
        OUT DX, AL
        INC BX
        MOV AL, [BX]
        MOV DX, PORTOUTB
        OUT DX, AL
        INC BX
        MOV AL, [BX]
        MOV DX, PORTOUTC
        OUT DX, AL
        JMP X1

;SUBROUTINES
        STOP    PROC NEAR
                MOV AL, CSENSUP
                MOV BL, FSENS
```

```
        AND BL, 00001000B
        OR  AL, BL
        MOV AH, CSENSDN
        MOV BL, FSENS
        AND BL, 00000001B
        OR  AH, BL
        AND AL, AH
        MOV AH, 0
        CMP AL, AH
        JNZ B1
        JZ B2
        B1:  CALL FSTOP
             JMP BEND
        B2:  CALL SLOW
             JMP BEND
        BEND:
        RET
        ENDP

SLOW    PROC NEAR
        LEA DI, PWM
        MOV AL, 1
        MOV [DI], AL
        CALL PWMSET
        RET
        ENDP
```

```
FAST    PROC NEAR
        LEA DI, PWM
        MOV AL, 2
        MOV [DI], AL
        CALL PWMSET
        RET
        ENDP

FSTOP   PROC NEAR
        MOV AL, PWM
        CMP AL, 0
        JZ   NOTIMER
        MOV DX, CLKC
        MOV AL, 32H
        OUT DX, AL
        NOTIMER:
        LEA DI, PWM
        MOV AL, 0
        MOV [DI], AL
        CALL PWMSET
        LEA DI, DCTRL
        MOV AL, 0
        MOV [DI], AL
        MOV AH, FSENS
        NOT AH
```

```
        LEA DI, FBTNUP
        MOV AL, [DI]
        AND AL, AH
        MOV [DI], AL
        LEA DI, FBTNDN
        MOV AL, [DI]
        AND AL, AH
        MOV [DI], AL
        LEA DI, LBTN
        MOV AL, [DI]
        AND AL, AH
        MOV [DI], AL
        RET
        ENDP

DIRECT  PROC NEAR
        MOV AL, FSENS
        MOV AH,01H
        CMP AL, AH
        JNZ C1
        LEA DI, MOTOR
        MOV BL, 01H
        MOV [DI], BL
        C1:
        MOV AL, FSENS
        MOV AH,08H
```

```
        CMP AL, AH
        JNZ C2
        LEA DI, MOTOR
        MOV BL, 00H
        MOV [DI], BL
        C2:
        MOV AL, FSENS
        MOV AH, TGTS
        CMP AH, AL
        JG C3
        LEA DI, MOTOR
        MOV BL, 00H
        MOV [DI],BL
        C3:
        RET
        ENDP

PWMSETPROC NEAR
        MOV AL, PWM0
        MOV AH, PWM
        CMP AL, AH
        JZ ENDPWM
        LEA DI, PWM0
        MOV [DI], AH
        MOV DX, CLKB
        MOV AL, PWM
```

```
INC AL
OUT DX, AL
ENDPWM:
RET
ENDP
```

Hardware Circuit Diagram