**Design Manual**

**Flow:**

Lift.asm

lcd.asm

macros.asm

**Data Structures:**

These are the memory mapped data.

|  |  |  |
| --- | --- | --- |
| Label | Size (.byte) | Comment |
| TempCounter | 2 | Temporary Counter for Timer0 |
| SecondCounter | 2 | Used to count seconds (when TempCounter reaches 7812) |
| DebounceCounter | 2 | Debouncing. Debounces for approx. ½ second |
| LightCounter | 2 | Counts time for strobe light flashes to determine when to turn strobe on/off |
| FloorNumber | 1 | Holds the current floor number |
| FloorBits | 2 | Holds the current floor number represented as a bit |
| FloorQueue | 2 | Holds all the called floors as set bits in the low 10 bits of 2 bytes |
| CurrentPattern | 2 | Holds the pattern needed to out for the LEDs for Lift/Door moving/close/open/etc. |

Register 22 is used as the Lift “STATUS” register.

; Status Bits: LSB [ 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 ] MSB

; 0 => EMERGENCY

; 1 => DEBOUNCE FLAG

; 2 => LIFT MOVING

; 3 => LIFT DIRECTION

; 4 => DOOR MOVING

; 5 => DOOR OPEN/CLOSE

Set/clear bits to designate the status of the lift and it's components. For example when traversing between floors Bit 2 is set or when a key is pressed the Bit 1 is set.

Some operations require more complex combinations of the bits to be set; such as when the lift reaches a floor that has been called to perform the open/close sequence Bits 4 and 5 need to be set and cleared in a specific order to get the “doors” to display properly on the LEDs.

Most registers used are between r16 – r31. This is due to certain operations being restricted to these registers. However a few times when the operation was simple, a lower register was used.

The only register lower than r16 that was defined was r9 as the strobe\_pattern since the 'pattern' was either 0xFF (on) or 0x00 (off).

The following is a list of the defined registers that were used.

.def strobe\_pattern = r9

.def row = r16 ; current row number

.def col = r17 ; current column number

.def rmask = r18 ; mask for current row during scan

.def cmask = r19 ; mask for current column during scan

.def temp1 = r20

.def temp2 = r21

.def status = r22

.def temp3 = r23

FloorQueue and FloorBits are representations of the 10 floors using 2 bytes each. Each bit represents a floor.

So, in FloorQueue if the user calls for floors 2, 5, and 9 the bits should appear as follows:

Bit Position [7 6 5 4 3 2 1 0] [7 6 5 4 3 2 1 0]

Bits [0 0 0 0 0 0 1 0] [0 0 1 0 0 1 0 0]

The top 6 bits of the 'higher' byte are wasted as they are never used. However this way it is much easier to keep track of the Queue and also allows me to add items to this queue while traversing and visiting the floors like a lift would in reality.

Similarly, FloorBits represents the current floor in the same way except it would only ever have ONE bit set at a time.

**Algorithm Descriptions:**

1. Multiple Requests

Do normal operations to get the value of the key pressed.

If a number 0-9:

If number is not current floor number:

get the representation of that floor as bits.

'OR' the floor's representation with FloorQueue

loop back to main

2. Get floor in bits

Get value from keypad

Initialise floor's bits to [0b00000000 0b00000001]

While the value is > 0

dec the value

left shift the bits

3. Enter Emergency Mode

Clear LCD, write Emergency message

If already at Floor 0:

open doors

Else

If door is open

close it

set lift to moving

if door is opening

close it

reset SecondCounter

set FloorQueue to Floor\_0

set lift direction to DOWN

4. Leave Emergency Mode

clear prompt

write current floor number

turn off emergency bit

turn off strobe

5. Open button

If lift is not moving (i.e. it is stopped on a floor)

if door is already open return

else

set door status to closed

set door status to moving

else return

6. Close button

If door is closed, return

else

set door status to open

set door status to moving

7. MoveLift (Activate the lift when a floor is queued)

Immediately set the lift status to moving

if floor queue value is less than current floor's bit value

then queued floor(s) must be below so set lift direction to DOWN

else

some floors must be above us

need to know if there are also any floors below us that are q'd

subtract 1 from current floor's bit value

now all bits lower than the current floor's bit are set

'AND' those bits with the FloorQueue

if this results in 0 in the that means that only q'd floors are above us only

set direction to UP

else

there are some queue'd floors above AND below so continue in current direction

8. Once a floor has been reached

inc/dec floor based on lift direction

turn off motor

set lift status to not moving

if floor in queue start the open/close process

**Module Specification:**

lift.asm is the main project file. It holds the main code for Timer0, and the other interrupt routines.

All operations are routed through lift.asm. It includes operations for debouncing, flashing the strobe, operating the patterns and writing them to the LEDs at appropriate times, etc.

lcd.asm holds the macros needed for LCD operations with a few extra macros used for specific purposes such as a *do\_lcd\_data\_reg* macro that does the same thing as *do\_lcd\_data* except it takes in a register as argument instead of a raw value. It also contains the .equ definitions needed for LCD operations

macros.asm contains miscellaneous macros used for various operations. These include macros to print and update the FloorQueue, get the bit representation of a floor, set and clear flags from the custom Lift “STATUS” register, and the Timer macros to clear one or two bytes from SRAM memory.