****

**TABLE OF CONTENTS**

[**USER REQUIREMENTS AND TECHNICAL SPECIFICATIONS**](#_hcbvd8f3s4xx) **3**

[User Requirements](#_z299hbsi61w6) 3

[Technical Specifications](#_ya4dgket1sa4) 3

[**ASSUMPTIONS AND JUSTIFICATION**](#_l08bj53tb39m) **3**

[Assumptions](#_rumcj8vme28c) 3

[Justification](#_orhfziu2th5z) 3

[**COMPONENTS USED WITH JUSTIFICATION (IF REQUIRED)**](#_agpzo8h5u5kq) **4**

[**ADDRESS MAP**](#_gilrf0nthxj5) **6**

[Memory Map](#_v1odzdxciidk) 6

[I/O Map](#_1p3axqwnh5gy) 6

[**DESIGN**](#_mn92gejiazw3) **6**

[**FLOWCHART**](#_gbkoo23m9akd) **7**

[**Table of States and “Process the Input” from Flowchart**](#_xk98rqfc52ep) **8**

[**VARIATIONS IN PROTEUS IMPLEMENTATION WITH JUSTIFICATION**](#_goxbns51ag84) **9**

[**LIST OF ATTACHMENTS**](#_xylfxiaaxumf) **10**

# 

# **USER REQUIREMENTS AND TECHNICAL SPECIFICATIONS**

## Problem 27 - Fan Speed Sensing and Control

## User Requirements

Design a system that senses the speed of the fan in real time and adjusts the fan speed so that the speed remains constant. The user can set fan speed using keypad and arrows. The system should also have an auto timer which turns the fan off after a set number of hours.

## Technical Specifications

* The fan has 5 speed levels
* Initial speed level is inputted by the keypad. The speed can be changed by up and down arrows.
* The auto mode must accept the number of hours through the keypad. Fan will be turned off after those many hours.

# **ASSUMPTIONS AND JUSTIFICATION**

## Assumptions

1. Sensor is calibrated to read a maximum value of 500 RPM.
2. The number of hours for auto mode are from 1 to 9.
3. The same keypad is used to take input of hours if auto mode is pressed. Otherwise it is used to take input of speed level setting.
4. The change in rpm levels of the fan takes around 500ms to 1s to physically manifest the change in levels.

## Justification

1. A fast ceiling fan has a maximum RPM of 400. So the RPM sensor is so programmed that the maximum value it reads is 500. ( Fan chosen to control - [Havells Fan](https://www.havells.com/en/consumer/fans/ceiling-fans/regular/pacer-(13).html#gref) , 400RPM, 75w, 220-240V AC, 50Hz. )
2. Since the average sleeping time for adults is from 7 to 9 hours, such a timer is adequate to suit the needs.
3. The number of hours as well as fan speed level is a single digit. So there is only the need of one 7-segment display.

# 

# **COMPONENTS USED WITH JUSTIFICATION (IF REQUIRED)**

1. **8086**
2. **8284**
3. **Hall Effect Sensor SS49E (Datasheet Attached)**
   * The fan will have a small magnet attached to the circumference. It will be placed such that for every rotation of fan, the hall sensor will be triggered once.
4. **LM2907 (Datasheet Attached)**
   * LM2907 is a frequency-to-voltage converter and is attached to the sensor on one side and to the ADC on the other. The hall effect sensor triggers the input of this IC once per rotation of fan. Thus we get the fan frequency as an analog voltage.
   * VOUT = VCC x R1 x C1 x f
5. **BTA16 (Datasheet Attached)**
   * BTA16 is a TRIAC. The TRIAC is used to chop the 220V AC waveform according to the 8254 output. Thus the fan speed can be controlled with the Microprocessor using the TRIAC.
6. **PC817 (Datasheet Attached)**
   * PC817 is an opto-isolator. It is used to protect the circuitry from high voltage.
7. **ADC 0801** (Datasheet Attached)
   * ADC 0801 is a 1-channel 8 bit parallel ADC.
8. **Common Anode 7 segment display**
   * 1 nos as only 1 digit needs to be displayed.
9. **IC 7447**
   * BCD to Common Anode 7 segment display driver.
10. **8255**
    * 2 nos. To interface Keypad, 7 segment display, ADC.
11. **8254**
    * 2 nos. To generate 50Hz PWM (for AC Motor),
    * -n-hour timed pulses (for auto mode), clock for ADC
12. **2716**
    * 4 nos. Smallest available ROM chip is 2K. We need even and odd banks. We need ROM at location 00000h-00FFFh & FF000h-FFFFFh.
13. **6116**
    * 2 nos. Minimum size of RAM is 2K. We need odd and even banking. Ram is needed for stack, variables, etc.
14. **LS 138**
    * 1 nos.
15. **LS 373, LS 245, LS 244** and required gates

# **ADDRESS MAP**

## Memory Map

| Component | Starting Address | Ending Address |
| --- | --- | --- |
| ROM1 | 00000H | 00FFFH |
| RAM1 | 01000H | 01FFFH |
| ROM2 | FF000H | FFFFFH |

## I/O Map

| Component | Starting Address | Ending Address |
| --- | --- | --- |
| 8255 #1 | 00H | 06H |
| 8255 #2 | 08H | 0EH |
| 8254 #1 | 10H | 16H |
| 8254 #2 | 18H | 1EH |
| 8259 | 20H | 22H |

# **DESIGN**

The complete design with proper labelling is attached.

# **FLOWCHART**

# **Table of States and “Process the Input” from Flowchart**

| **IP Key** | **Current State** | **Change to State** | **Action** |
| --- | --- | --- | --- |
| Off | Any | Off | Disable timers, counters, motor |
| On | Off | Normal | Enable motor, counters |
| Wait\_for\_auto | Auto | Enable timer,motor, counters |
| Auto, Normal | - | - |
| Auto | Off , Normal | Wait\_for\_Auto | The next input number taken is used to programme timer |
| Wait\_for\_Auto, Auto | - | - |
| Up/Down | Any | - | Increase or decrease target RPM to required level |
| Number from 1-5 | Wait\_for\_Auto | Auto | Enable timer,motor, counters. Set i/p Number as hours in timer |
| Off, Auto, Normal | - | Increase or decrease target RPM to required level |
| Number from 6-9 | Wait\_for\_Auto | Auto | Enable timer,motor, counters. Set i/p Number as hours in timer |
| Off, Auto, Normal | - | - |

# **VARIATIONS IN PROTEUS IMPLEMENTATION WITH JUSTIFICATION**

1. The sensor LM2907 is not available in Proteus. So, the Proteus file will have a DC voltage source to mimic the output of the sensor.
2. Since, 2716 isn’t available on Proteus for simulation, we need to use 2732. So, the address map changes accordingly
3. Use of 8253, since 8254 is not available for simulation on Proteus.
4. The motor controller that we have used isn’t available on Proteus, so we have to simulate our system using a DC motor.

# 

# **LIST OF ATTACHMENTS**

1. Real world hardware design - Design.pdf
2. Datasheets-
   1. LM2907
   2. BTA16
   3. PC817
   4. ADC 0801
   5. SS49E
3. Proteus File - Assignment.DSN
4. EMU8086 ASM FIle - fan\_control.asm
5. Binary file after assembly - fan\_control.bin