Chapter 1

INTRODUCTION

The project aims to bring about a changeover in the age-old office cabin. An office usually consists of a network of cabins and it would be quite a neck breaking task to monitor the person inside the cabin without shelling out quite a bit of money. The situation calls for a cheaper, smarter and efficient technology which could cater both to the needs of the employee and the employer. The system was developed to monitor the presence of a person inside the cabin. Once the person is present inside the cabin, it automatically controls the temperature inside the cabin by controlling the speed of the fan in it. Likewise, the light inside the cabin is maintained to an ambient setting which can be controlled by the user.

At present the project is being implemented in a teachers' staffroom. The workstation detects the presence of the teacher inside the cabin and displays the information in front of the staffroom in a LED display. This enables a student or any other person to know if the cabin is currently occupied, without having to enter inside. If the teacher is present inside, the workstation creates an ambient working atmosphere by controlling the lights and fan inside it.

The current test scenario consists of four cabins of the staffroom. Each cabin is considered as a node and is connected to a central server. Any number of nodes can be added to the system. The nodes send data such as temperature and other sensor readings to the server from where necessary control actions are taken up.

The microcontroller ATmega328 is used to control all the functions of the system. The communication between the nodes and the server is established using NRF24l01 transceiver, which operates at 2.4 GHz. Temperature readings are obtained using an LM35 temperature sensor and light readings using an LDR. The presence of the teacher inside the cabin is determined using a proximity sensor and the result is displayed using

an LED display. The user, in this case the teacher, can manually control or configure the settings of the system so as to cater to their needs.

Chapter 2

HARDWARE DETAILS

2.1. Block Diagram:

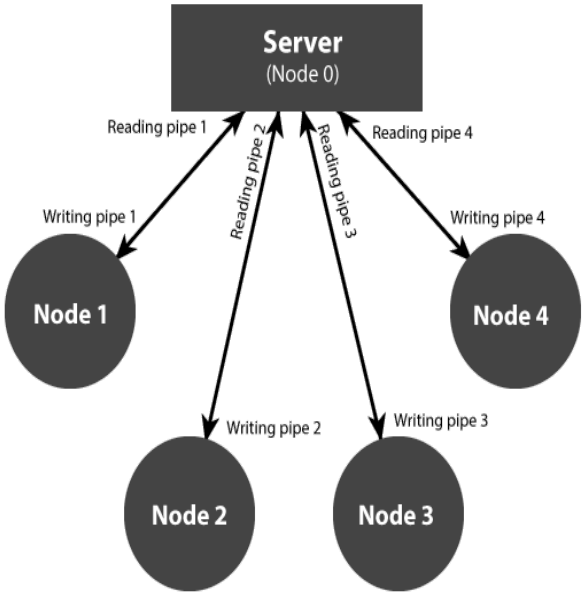


Fig 2.1: Shows the communication between the nodes and server

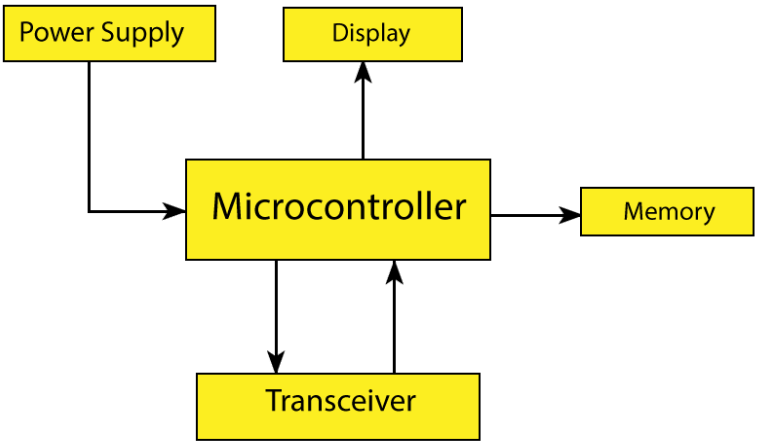


Fig 2.2: Shows block diagram of server

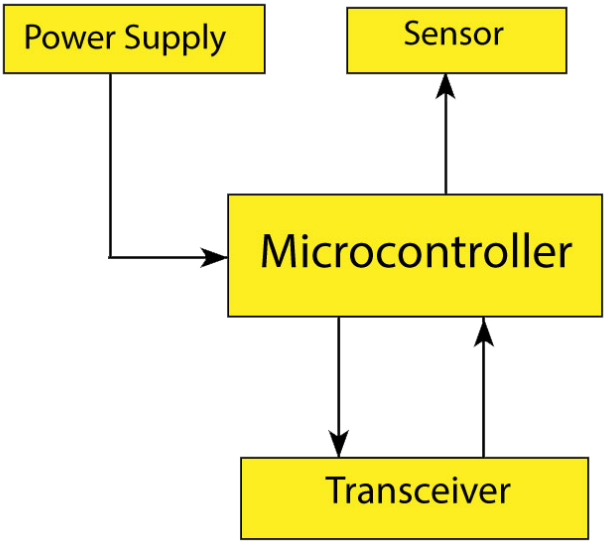


Fig 2.3: Shows the block diagram of node

2.1. Block Diagram: Main Hardware Description

2.1.1. ATMEGA 328P

The ATmega328P provides the following features: 32Kbytes of In-System Programmable Flash with Read-While-Write capabilities, 1Kbytes EEPROM, 2Kbytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), three flexible Timer/Counters with compare modes and PWM, 1 serial programmable USARTs , 1 byte-oriented 2-wire Serial Interface (I2C), a 6- channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages) , a programmable Watchdog Timer with internal Oscillator, an SPI serial port, and six software selectable power saving modes.

The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset.

In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In Extended Standby mode, both the main oscillator and the asynchronous timer continue to run.

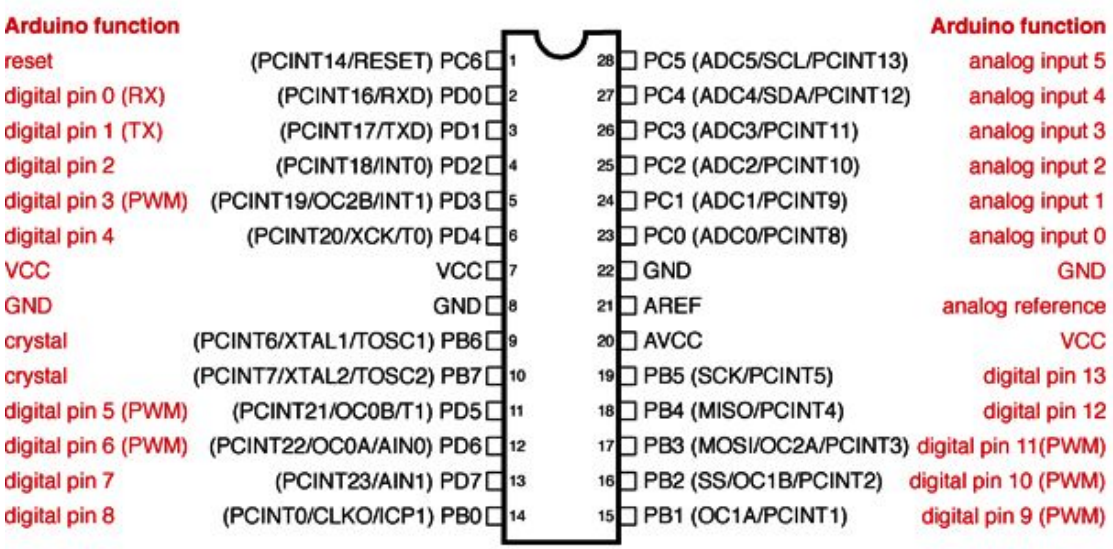


Fig 2.4: Shows the pin Diagram of Atmega328P

2.1.2 LM35

One of the main features of this project is maintaining the temperature inside the particular cabin by controlling the speed of the fan. This is done with the help of a temperature sensor, LM35.

The sensor picks up temperature readings within the cabin and this information is sent to the central server where decision is taken on regulating the fan speed.

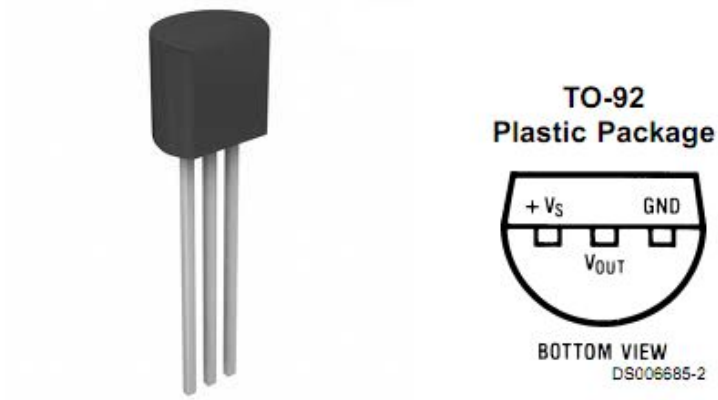


Fig 2.5: LM35 [Wikipedia] *Fig 2.6: LM35 Pin Diagram [Wikipedia]*

General Description:

The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. The LM35 thus has an advantage over linear temperature sensors calibrated in ˚ Kelvin, as the user is not required to subtract a large constant voltage from its output to obtain convenient Centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of ±1⁄4˚C at room temperature and ±3⁄4˚C over a full −55 to +150˚C temperature range.

The LM35's low output impedance, linear output, and precise inherent calibration make interfacing to readout or control circuitry especially easy. It can be used with single power supplies, or with plus and minus supplies. As it draws only 60 µA from its supply, it has very low self-heating, less than 0.1˚C in still air. The LM35 is rated to operate over a −55˚ to +150˚C temperature range, while the LM35C is rated for a −40˚ to +110˚C range (−10˚ with improved accuracy). The LM35 series is available packaged in hermetic TO-46 transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package.

Specifications:

- Calibrated directly in ˚ Celsius (Centigrade)
- Linear + 10.0 mV/˚C scale factor
- 0.5˚C accuracy guarantee-able (at +25˚C)
- Rated for full −55˚ to +150˚C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Less than 60 µA current drain
- Low self-heating, 0.08˚C in still air
- Nonlinearity only ±1⁄4˚C typical
- Low impedance output, 0.1 Ω for 1 mA load

2.1.3. NRF24L01

The communication for the project is established through a transmitter and a receiver. NRF24l01 is a transceiver. That is, it transmits and receives data between the nodes and the server thus enabling communication.

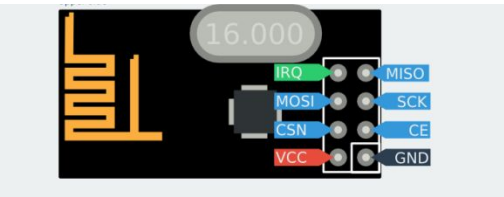


Fig 7:Pin Diagram of NRF24L01 Module [Google Images]

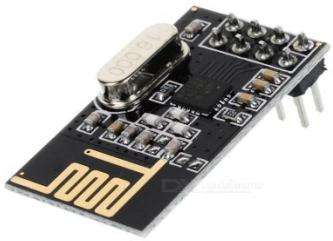


Fig 8:NRF Transceiver module [Google Images]

General Description:

The nRF24L01 is a single chip 2.4GHz transceiver designed for ultra low power wireless applications. The nRF24L01 is designed for operation in the world wide ISM frequency band at 2.400 - 2.4835GHz.The nRF24L01 is configured and operated through a Serial Peripheral Interface (SPI.) Through this interface the register map is available. Internal FIFOs ensure a smooth data flow between the radio front end and the system's MCU.

The radio front end uses GFSK modulation. It has user configurable parameters like frequency channe ,output power and air data rate.The air data rate supported by the nRF24L01 is configurable to 2 Mbps. The high air data rate combined with two power saving modes makes the nRF24L01 very suitable for ultra low power designs. Internal voltage regulators ensure a high Power Supply Rejection Ratio (PSRR) and a wide power supply range.

Specifications:

- Power supply :1.9V~3.6V
- Working current:13.5mA at 2Mbps / 11.3mA at 0dBm output power
- IO counts :8
- Sensitivity :-85dBm at 1Mbps

- Emission distance :70~100 meter at 256kbps
- Data rate :256kbps / 1Mbps / 2Mbps
- Communication mode :Enhanced ShockBurst TM / ShockBurst TM
- Working mode :Power Down Mode / Standby Mode / RX Mode / TX Mode
- Temperatures :Operating:-40°C ~ 85°C / Storage:-40°C ~ 125°C

2.1.4. PROXIMITY SENSOR

A proximity sensor is used to to detect the presence of nearby objects. Inside the cabin, to detect the presence of the teacher, an IR (infrared sensor) is used.The result is displayed on the display monitor outside.

General Description:

The maximum distance that this sensor can detect is defined "nominal range". Some sensors have adjustments of the nominal range or means to report a graduated detection distance. Some know these process as "thermo sensation".

Proximity sensors can have a high reliability and long functional life because of the absence of mechanical parts and lack of physical contact between sensor and the sensed object.

Specifications:

- Number of Pins - 4 (AO,DO,VCC,GND)
- Output Type - Analog and/or Digital output
- Operating Voltage - 4.5V to 6V
- Lead Pitch- 0.1th inch (2.54 mm) breadboard compatible

2.1.5. LDR (Light Dependent Resistor)

Fig 2.9: LDR [Wikipedia]

An LDR is a device whose resistivity is a function of electromagnetic radiation. They are also called photo conductors or photo cells.

The purpose on an LDR in our project is to maintain ambient lighting in a room automatically. i.e. to adjust the intensity of light depending on the daylight.

General Description

A photoresistor is made of a high resistance semiconductor. In the dark, a photoresistor can have a resistance as high as several megohms (MΩ),while in the light, a photoresistor can have a resistance as low as a few hundred ohms.When light falls i.e. when the photons fall on the device, the electrons in the valence band of the semiconductor material are excited to the conduction band. These photons in the incident light should have energy greater than the band gap of the semiconductor material to make the electrons jump from the valence band to the conduction band.

Hence when light having enough energy strikes on the device, more and more electrons are excited to the conduction band which results in large number of charge carriers. The result of this process is more and more current starts flowing through the device when the circuit is closed and hence it is said that the resistance of the device has been decreased. This is the most common working principle of LDR.

Specifications

- resistance : 400 ohm to 400 Kohm
- normal resistance variation: 1 Kohm to 10Kohm
- sensitivity: about 3 msec

- Voltage ratings: I used it on 3V,5V and 12V

2.1.6. RELAY

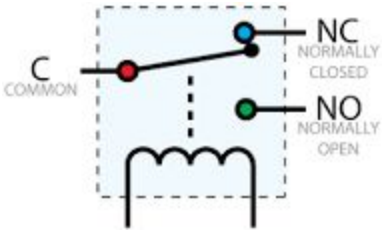


Figure 2.10: Relay [Wikipedia] *Fig 2.11 Relay Pinout Diagram[Wikipedia]*

A **relay** is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a separate low-power signal, or where several circuits must be controlled by one signal.

Chapter 3

SOFTWARE DETAILS

3.1. Software's Used

3.1.1 Arduino IDE Software

The Arduino project provides the Arduino integrated development environment (IDE), which is a cross-platform application written in the programming language Java. It originated from the IDE for the languages *Processing* and *Wiring*. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and provides simple one-click mechanism to compile and load programs to an Arduino board. A program written with the IDE for Arduino is called a "sketch".

The Arduino IDE supports the languages C and C++ using special rules to organize code. The Arduino IDE supplies a software library called Wiring from the Wiring project, which provides many common input and output procedures. A typical Arduino C/C++ sketch consist of two functions that are compiled and linked with a program stub *main()* into an executable cyclic executive program:

- *setup()*: a function that runs once at the start of a program and that can initialize settings.
- *loop()*: a function called repeatedly until the board powers off.

After compiling and linking with the GNU toolchain, also included with the IDE distribution, the Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal coding that is loaded into the Arduino board by a loader program in the board's firmware.

The version used in this project is 1.6.5

3.1.2. Fritzing

Fritzing is an open source initiative to develop amateur or hobby CAD software for the design of electronics hardware, to support designers and artists ready to move from experimenting with a prototype to building a more permanent circuit. It was developed at the University Of Applied Sciences Of Potsdam.

The software is created in the spirit of the Processing programming language and the Arduino microcontroller and allows a designer, artist, researcher, or hobbyist to document their Arduino-based prototype and create a PCB layout for manufacturing. The associated website helps users share and discuss drafts and experiences as well as to reduce manufacturing costs.

Fritzing can be seen as an electronic design automation (EDA) tool for non-engineers: the input metaphor is inspired by the environment of designers (the breadboard-based prototype), the output is offering nearly no options and is focused on accessible means of production. As of December 2, 2014 Fritzing has made a code view option, where one can modify code and upload it directly to an Arduino device.

Operating system : Mac OS X, Linux, Windows

License : GNU GPL v3 (software)

Developer(s) : Interaction Design Lab Potsdam

3.3. FLOWCHART

3.3.1. Node

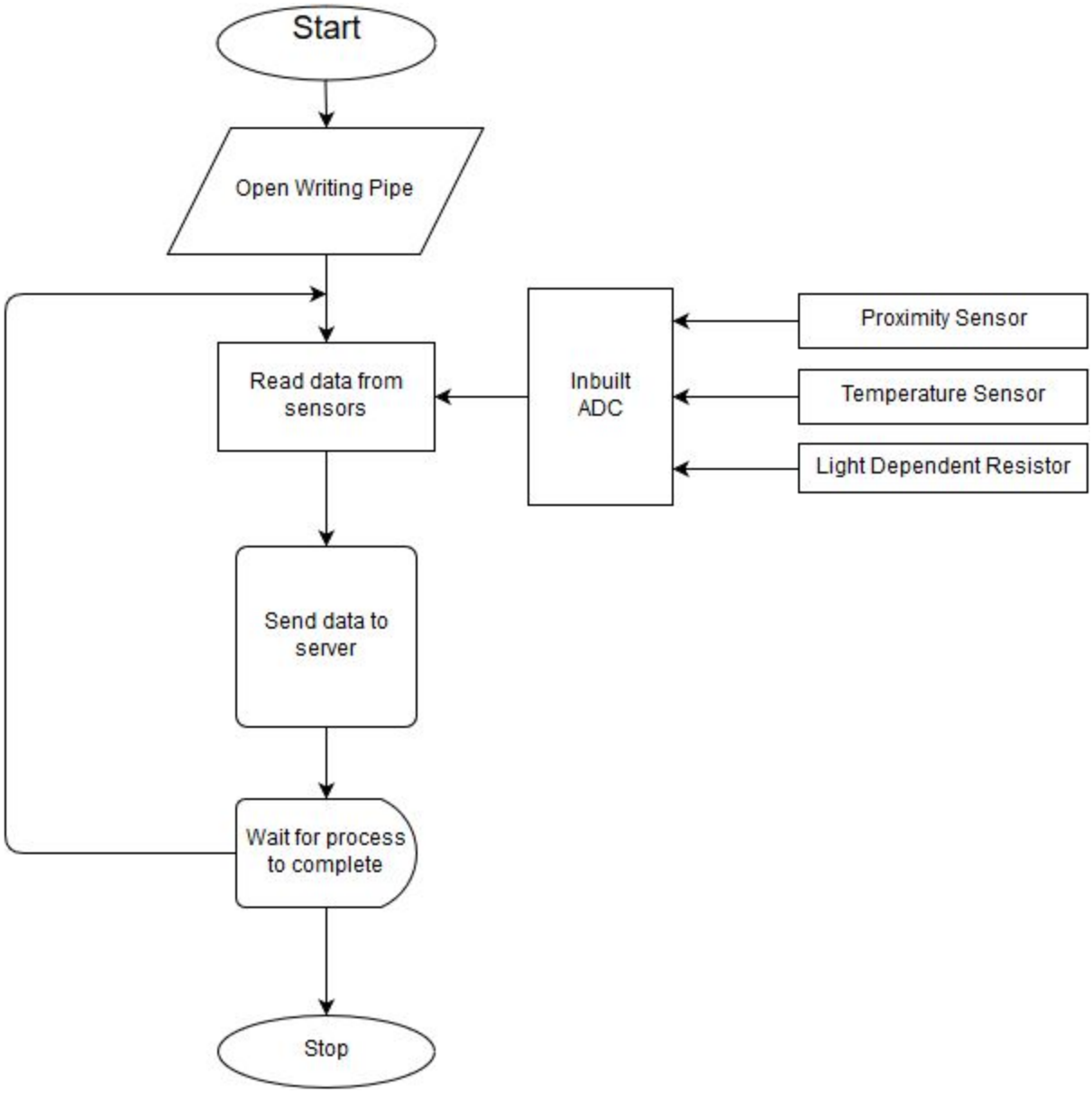


Fig 3.1: Shows the flowchart of Node program

3.3.2. Server

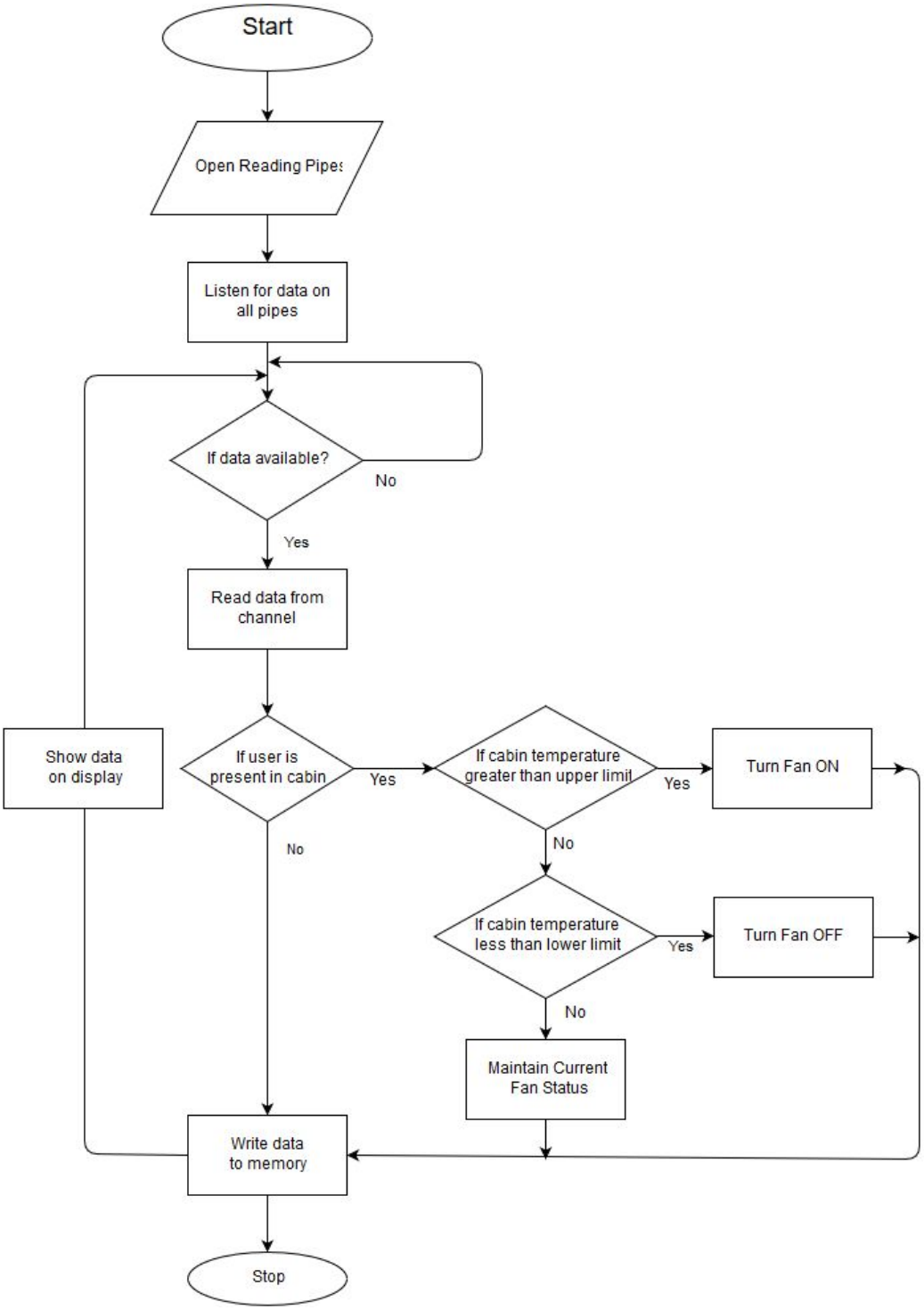

Start
Open Reading Pipes
Listen for data on all pipes
If data available?
No
Yes
Read data from channel
Show data on display
If user is present in cabin
Yes
If cabin temperature greater than upper limit
Yes
Turn Fan ON
No
If cabin temperature less than lower limit
Yes
Turn Fan OFF
No
Maintain Current Fan Status
No
Write data to memory
Stop

Fig 3.2: Flowchart of Server

Chapter 4

IMPLEMENTATION

4.1. Project development stages

PHASE 1:

To develop an intelligent workstation, initially the communication between the various nodes was established by interfacing arduino and RF transceiver modules. The components were studied and their feasibility assessed.

Finally the transceiver module selected was *NRF24L01* due to low power modes and multiple channels. The basics of Arduino programming were studied. The programming was developed using C.

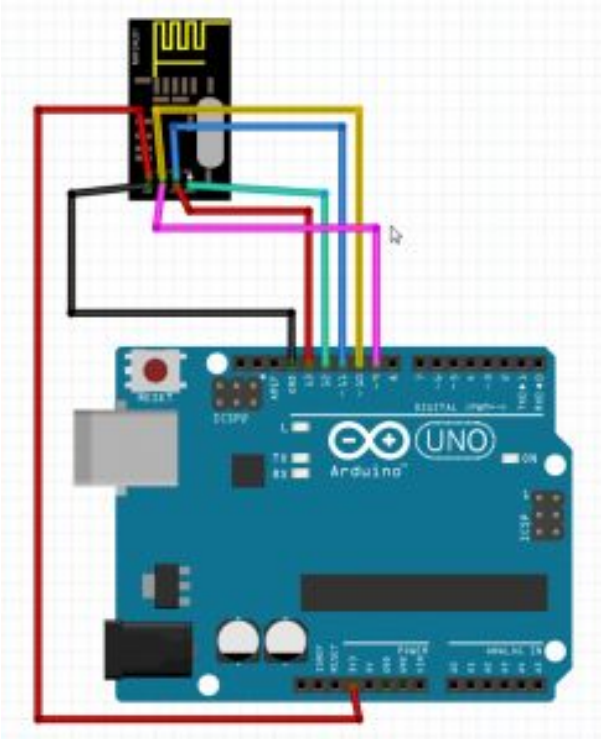


Fig4.1 Interfacing arduino and NRF24L01[Google Images]

PHASE 2:

Next we interfaced a temperature sensing circuit using LM35 to sense the temperature and the data transmitted to the power control unit. Thereby controlling the speed of the fan according to the data received.

PHASE 3:

Further we modified the code for the node by enabling the pipes to listen for any data. Once the data is available the node sends the data which is received by the server. Based on this data the presence of a person within a cabin is displayed on the monitor and the fan is controlled.

PHASE 4:

The code for the server was developed and a display monitor setup. Once the server receives the data, the procured information is stored into an external SD card.

PHASE 5:

PCB layout was done for both server and the nodes. PCB fabrication and soldering of components on the board was done.

Chapter 5

PCB LAYOUT AND FABRICATION

Chapter 5 details the preparation of PCB layout and its subsequent fabrication procedures. The materials required for PCB fabrication are copper clad, ferric chloride solution, and drilling machine. The PCB fabrication includes the following steps:

5.1. Preparation of the PCB layout

First the circuit is drawn using Fritzing. The image of the layout of is printed on an A4 size translucent tracing sheet or butter paper.

5.2. Transferring the layout to copper clad sheet

First the copper clad sheet of required dimension is cut by using a cutting machine. Then the sheet is cleaned by using a metal scrubber. The next step is to form an image of the layout on the copper clad sheet. The copper clad sheet is exposed to heat by ironing using an iron box till the design is printed on the board. After that, the board is washed gently in water for about 1mnt so that the image is copied into the board.In case the process is not successful ,the process is repeated.

5.3. Etching of the board

When the board is ready for etching, it is placed in the ferric chloride solution of required concentration. It is checked in regular intervals to prevent over etching and successive damage to the part. After etching is complete, the board is taken out of the solution and is washed in water to remove the excess ferric chloride. The D13X NC thinner is applied to remove any dew or paint material on copper tracks. Then the sheet is

cleaned by using steel scrubber and washed again in water. Now the copper lines are exposed and the body is checked with the magnifying glass to see whether all the lines in the layout are clearly formed. Now the board is ready for tinning.

5.4. Tinning

For tinning, the PCB is cleaned well and flux is applied to surface. Then it is passed through the tinning machine. In tinning, the copper lines are plotted with an alloy of TIN and LEAD.

5.5. Drilling

After tinning, the next process is drilling. In this the holes of required sizes are drilled in the PCB wherever needed, using a PCB drilling machine.

5.6. Finishing

In the process after drilling the holes on the PCB, the board is taken and a light coat of air dyeing varnish insulating varnish is applied to the bottom side carefully avoiding the PAD areas. The PCB is then left till the insulating varnish dry up.

Chapter 7

RESULT AND DISCUSSION

The project 'Intelligent Workstation' was successfully completed achieving all the requirements which were proposed. Once the person (teacher) occupied the cabin, the result was displayed on the LED display.

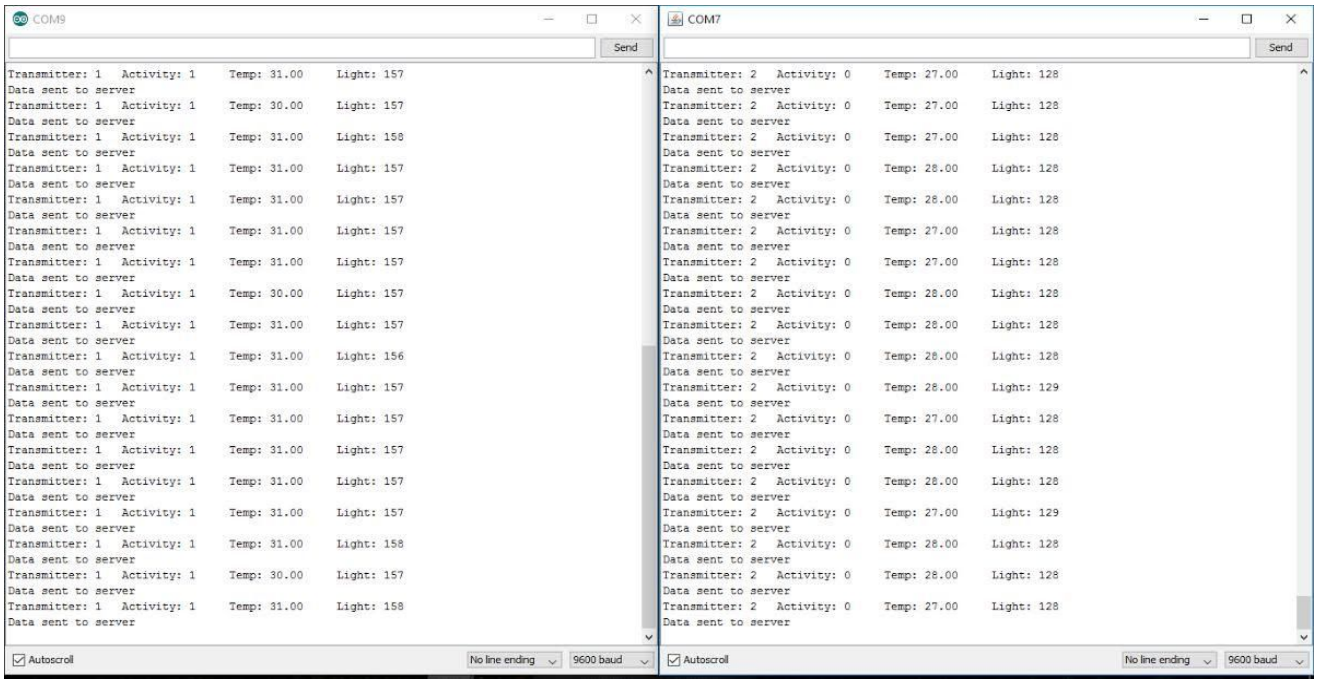


Fig 7.1: Output data sent from the server obtained from the sensors displayed on the serial port

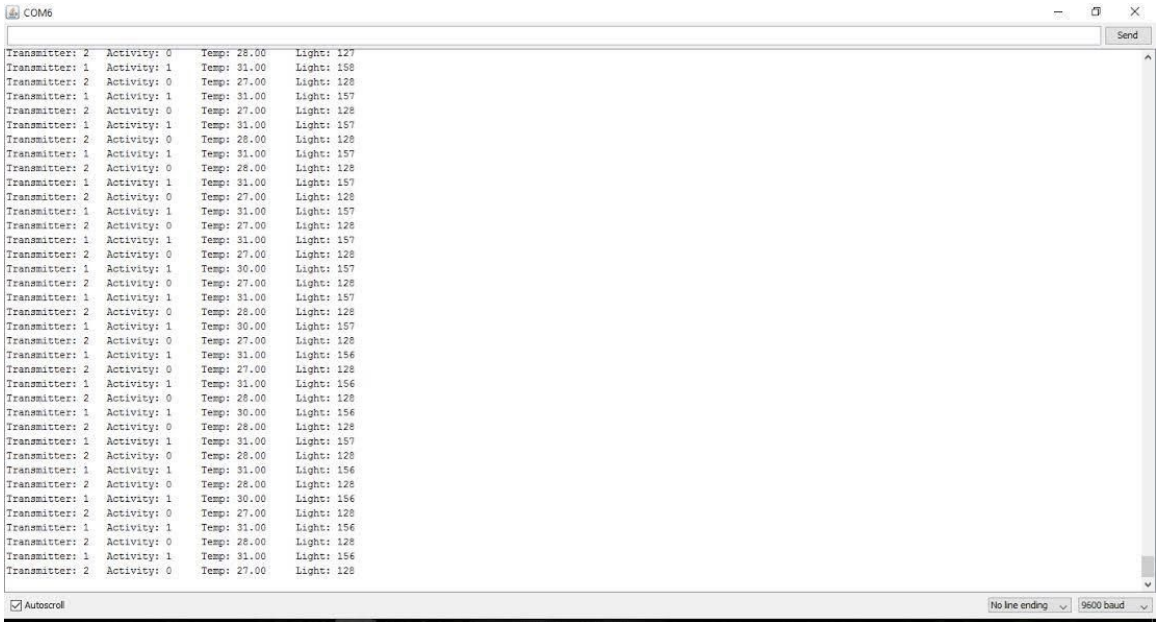


Fig 7.2: Data received by the server displayed on the serial port

Chapter 8

CONCLUSION AND FUTURE SCOPE

The project 'Intelligent Workstation' was developed successfully and tested. The project has been able to meet up with all the goals which were initially proposed. The project has been successfully implemented in four cabins inside the staffroom. The main features of the project include:

- The presence of the person (teacher) inside the cabin is detected and the result is displayed in front of the staffroom.
- Once a presence inside the cabin is detected, the temperature is regulated by taking necessary temperature readings inside the cabin using the temperature sensor LM35.

The future advancements of this particular project are numerous. Some of the possible developments which we had initially looked upon are:

- Controlling the light inside the cabin using a dimmer circuit.
- Integrating the schedule of the particular teacher into the cabin.
- Uses of RFID tags to determine which teacher was present in each cabin at a particular time and also use the tags to mark their attendance each day.

REFERENCES

[1] Gulnara Z. Karimova & Amir Shirkhanbeik, "Society of things: An alternative vision of Internet of things", *Cogent Social Sciences (2015)*, 1: 1115654
DOI: 10.1080/23311886.2015.111565, Nov 2015

[2] "Society of Things", *https://hackaday.io/project/2160-a-society-of-things* July 27,2014

[3] [online] http://forcetronic.blogspot.in/2015/05/creating-nrf24l01-transceiver

[4] [online] http://shanes.net/simple-nrf24l01-with-arduino-sketch-and-setup/

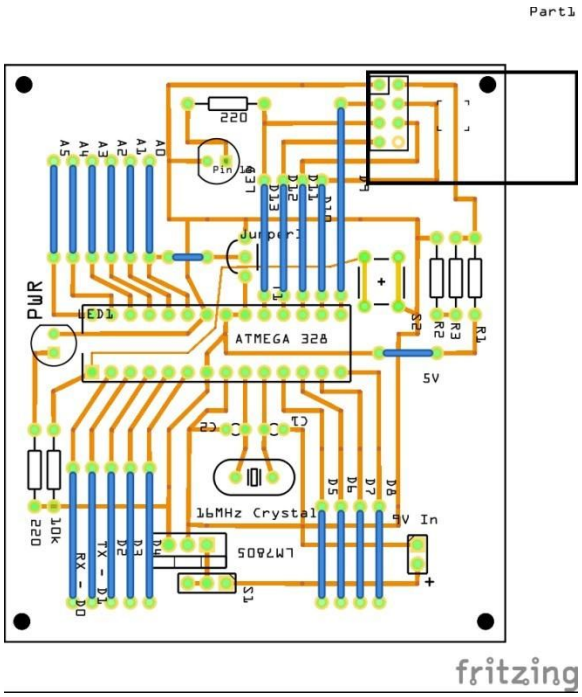
[5] [online] http://playground.arduino.cc/InterfacingWithHardware/Nrf24L01

[6] [online] https://en. wikipedia.org/wiki/Internetofthings

[7] [online] http://fritzing.org/learning/tutorials

APPENDIX

1. PCB LAYOUT



2. PROGRAM CODE

Server

```
#include <SPI.h> //Call SPI library so you can communicate with the nRF24L01+
#include <nRF24L01.h>
#include <RF24.h>
```

```
const int pinCE = 9; //This pin is used to set the nRF24 to standby (0) or active mode (1)
const int pinCSN = 10;
RF24 radio(pinCE, pinCSN); // Declare object from nRF24 library (Create your wireless SPI)
const uint64_t rAddress = 0xE8E8F0F0E1LL;
typedef struct{
int node;
int activity;
float temp;
int light;
}data;

data tr;

void setup()
{
Serial.begin(9600);  //start serial to communication
Serial.println("Server On. Listening for transmissions ");
radio.begin();  //Start the nRF24 module
radio.openReadingPipe(1,rAddress);
radio.startListening();            // Start listening for messages
}

void loop()
{
byte pipeNum = 0; //variable to hold which reading pipe sent data
byte gotByte = 0; //used to store payload from transmit module
```

```
while(radio.available(&pipeNum)){ //Check if recieved data
radio.read( &tr, sizeof(tr) ); //read one byte of data and store it in gotByte variable
Serial.print("Transmitter: ");
Serial.print(tr.node); //print which pipe or transmitter this is from
Serial.print("\t Activity: ");
Serial.print(tr.activity); //print payload or the number the transmitter guessed
Serial.print("\t Temp: ");
Serial.print(tr.temp); //print payload or the number the transmitter guessed
Serial.print("\t Light: ");
Serial.print(tr.light); //print payload or the number the transmitter guessed
Serial.println();
}
delay(100);
}
```

Node

```
#include <SPI.h> //Call SPI library so you can communicate with the nRF24L01+
#include <nRF24L01.h>
#include <RF24.h>

#define node_id 1;
int act_sens = 3;
int temp_sens = A0;
int ldr = A1;



const int pinCE = 9; //This pin is used to set the nRF24 to standby (0) or active mode (1)
```

```
const int pinCSN = 10;
RF24 radio(pinCE, pinCSN); // Declare object from nRF24 library (Create your wireless SPI)
const uint64_t wAddress = 0xE8E8F0F0E1LL;

typedef struct{
int node ;
int activity;
float temp;
int light;
}data;

data tr;



void setup()
{
pinMode(act_sens,INPUT);
tr.node = node_id;
Serial.begin(9600);  //start serial to communication
Serial.print("Initialised Node ");
Serial.println(tr.node);
radio.begin();  //Start the nRF24 module
radio.openWritingPipe(wAddress);    // setup pipe to transmit over
radio.stopListening();
}

void loop()
{
```

```
tr.activity = digitalRead(act_sens);
tr.temp = (analogRead(temp_sens))*500/1024;
tr.light = analogRead(ldr);
Serial.print("Transmitter: ");
Serial.print(tr.node); //print which pipe or transmitter this is from
Serial.print("\t Activity: ");
Serial.print(tr.activity); //print payload or the number the transmitter guessed
Serial.print("\t Temp: ");
Serial.print(tr.temp); //print payload or the number the transmitter guessed
Serial.print("\t Light: ");
Serial.print(tr.light); //print payload or the number the transmitter guessed

Serial.println();
if (!radio.write( &tr, sizeof(tr) ))
{Serial.println("Data sending failed");}
else
{Serial.println("Data sent to server ");}
radio.powerDown();
delay(1000);
radio.powerUp();
}
```

3. COST ESTIMATION

Sl. No.	COMPONENT	Quantity	TOTAL PRICE(Rs)
1	ATmega328P	6	180 X 6
2	Nrf24L01	5	150 X 5
3	LM35	5	45 X 5

4	Ferric Chloride	1	45 X 1
5	PCB Drill	1	120 X 1
6	Thinner	1	65 X 1
7	Jumper Wires	20	20 X 4
8	Copper Clad	1	120 X 1
9	IR Proximity Sensor	5	5 X 40
10	Relay Module	1	1 X 450
TOTAL			2685

4. ATMEGA328 -Features

Features	ATmega328/P
Pin Count	28/32
Flash (Bytes)	32K
SRAM (Bytes)	2K
EEPROM (Bytes)	1K
Interrupt Vector Size (instruction word/vector)	1/1/2
General Purpose I/O Lines	23
SPI	2
TWI (I^2C)	1
USART	1
ADC	10-bit 15kSPS
ADC Channels	8
8-bit Timer/Counters	2
16-bit Timer/Counters	1

5. Nrf24L01

Operating conditions	Minimum	Maximum	Units
Supply voltages			
VDD	-0.3	3.6	V
VSS		0	V
Input voltage			
V_I	-0.3	5.25	V
Output voltage			
V_O	VSS to VDD	VSS to VDD	
Total Power Dissipation			
P_D (T_A=85°C)		60	mW
Temperatures			
Operating Temperature	-40	+85	°C
Storage Temperature	-40	+125	°C

6. LM35

Absolute Maximum Ratings (Note 10)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Supply Voltage	+35V to −0.2V
Output Voltage	+6V to −1.0V
Output Current	10 mA
Storage Temp.;	
TO-46 Package,	−60˚C to +180˚C
TO-92 Package,	−60˚C to +150˚C
SO-8 Package,	−65˚C to +150˚C
TO-220 Package,	−65˚C to +150˚C
Lead Temp.:	
TO-46 Package, (Soldering, 10 seconds)	300˚C
TO-92 and TO-220 Package, (Soldering, 10 seconds)	260˚C
SO Package (Note 12)	
Vapor Phase (60 seconds)	215˚C
Infrared (15 seconds)	220˚C
ESD Susceptibility (Note 11)	2500V
Specified Operating Temperature Range: T_{MIN} to T_{MAX} (Note 2)	
LM35, LM35A	−55˚C to +150˚C
LM35C, LM35CA	−40˚C to +110˚C
LM35D	0˚C to +100˚C

Electrical Characteristics

(Notes 1, 6)

Parameter	Conditions	LM35A			LM35CA			Units (Max.)
		Typical	Tested Limit (Note 4)	Design Limit (Note 5)	Typical	Tested Limit (Note 4)	Design Limit (Note 5)	
Accuracy (Note 7)	T_A=+25˚C	±0.2	±0.5		±0.2	±0.5		˚C
	T_A=−10˚C	±0.3			±0.3		±1.0	˚C
	T_A=T_{MAX}	±0.4	±1.0		±0.4	±1.0		˚C
	T_A=T_{MIN}	±0.4	±1.0		±0.4		±1.5	˚C
Nonlinearity (Note 8)	T_{MIN}≤T_A≤T_{MAX}	**±0.18**		**±0.35**	**±0.15**		**±0.3**	˚C
Sensor Gain (Average Slope)	T_{MIN}≤T_A≤T_{MAX}	**+10.0**	**+9.9, +10.1**		**+10.0**		**+9.9, +10.1**	mV/˚C
Load Regulation (Note 3) 0≤I_L≤1 mA	T_A=+25˚C	±0.4	±1.0		±0.4	±1.0		mV/mA
	T_{MIN}≤T_A≤T_{MAX}	**±0.5**		**±3.0**	**±0.5**		**±3.0**	mV/mA
Line Regulation (Note 3)	T_A=+25˚C	±0.01	±0.05		±0.01	±0.05		mV/V
	4V≤V_S≤30V	**±0.02**		**±0.1**	**±0.02**		**±0.1**	mV/V
Quiescent Current (Note 9)	V_S=+5V, +25˚C	56	67		56	67		µA
	V_S=+5V	**105**		**131**	**91**		**114**	µA
	V_S=+30V, +25˚C	56.2	68		56.2	68		µA
	V_S=+30V	**105.5**		**133**	**91.5**		**116**	µA
Change of Quiescent Current (Note 3)	4V≤V_S≤30V, +25˚C	0.2	1.0		0.2	1.0		µA
	4V≤V_S≤30V	**0.5**		**2.0**	**0.5**		**2.0**	µA
Temperature Coefficient of Quiescent Current		**+0.39**		**+0.5**	**+0.39**		**+0.5**	µA/˚C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, I_L=0	+1.5		+2.0	+1.5		+2.0	˚C
Long Term Stability	T_J=T_{MAX}, for 1000 hours	±0.08			±0.08			˚C