Heat Regulation of a Wood Stove

By Dean Devereaux
Bachelor of Electronic Engineering – Year 3
Dept of Electronic Engineering
Cork Institute of Technology,
Cork

1 Overview

Stove heat regulator is the project I choose to do for my 3rd year of Electronic Engineering. A wood stove is the second major cause of air pollution when it comes to household heating systems and also produces a lot of heat that isn't required once a house is heated. This can be controlled through the methods which will be discussed in this report.

I choose this as my home has a stove installed for a heating system and this project would help me in understanding the process the stove goes through along with how to control heat transfer. This project report will explain the objectives of the project. What components will be used and how these components will be programmed. The plan created to work on the pieces of the project and how well this was kept to throughout. What was accomplished in the 3 months working on this project and the results collected from testing with an evaluation on the work done through the overall project.

2 Objectives

The objective of this project is to control the air vent of a stove using two temperature sensors in order to regulate the intake of heat being produced by this wood fuel stove. In doing so the air vent will also be controlling the burn rate of the fuel. This project will cause the stove to have a reduced amount of manual labour as it will automatically change the position of the vent based on the environments condition allowing the house to have a comfortable temperature. The idea is to use the return pipe bringing the radiator water back to the stove and compare this to the temperature within the flow pipe which brings water from the stove to the radiator. When the return pipe is over a desired temperature difference with the flow pipe, the air vent will close gradually allowing the heat and burn rate to be reduced through the lack of air being supplied to the fuel. When the return pipe goes below the desired temperature difference the air vent will then gradually open allowing for more air flow within the stove to increase burn rate allowing more heat. Using an LCD the data retrieved from the return pipe will be displayed for the benefit of the end user.

3 Market Research

The majority of research done for this project was through online using Google to search for all data. For the PIC18F452 and C programming notes were reviewed from module 'Firmware Programming' of Electronic Engineering Year 2.

The stepper motor was suggested by my supervisor. [1]I viewed a video explaining the different movement possible with a stepper motor and stuck with half stepping. Bipolar stepper motor was selected instead of the unipolar stepper due to its use with an h-bridge and coil setup.

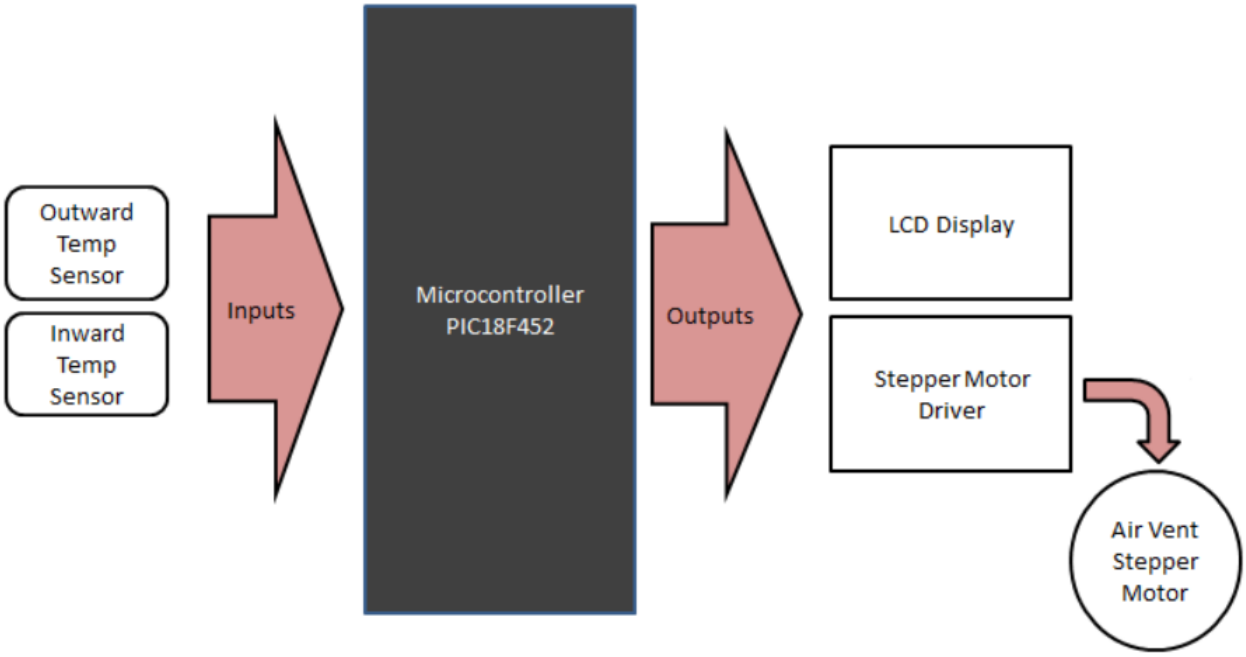
The LM35 was found on proteus when simulating and specs seemed adiquate to be used as a temperature sensor for this project as the temperature range suited and was easy use.

3.1 Stakeholders

Name/Person/Organization	Impact/Importance
Dean Devereaux	Researching and producing the project
Fergus O'Reilly	Supervisor of the project giving advice and helping with providing equipment if necessary
Stove Manufacturers	Improve/alter their branded stove with a controlled heating device
Consumers	If this project became reality then individuals who buy stoves with this device can maintain the temperature without manually changing the air vents

4 System Design

4.1 Block Diagram



4.1.1 Inputs

Two LM35D temperature sensors were used for the inputs and were connected onto pin 0 and pin 1 of PORTA to be used with the ADC

4.1.2 Outputs

A. Air Vent Motor

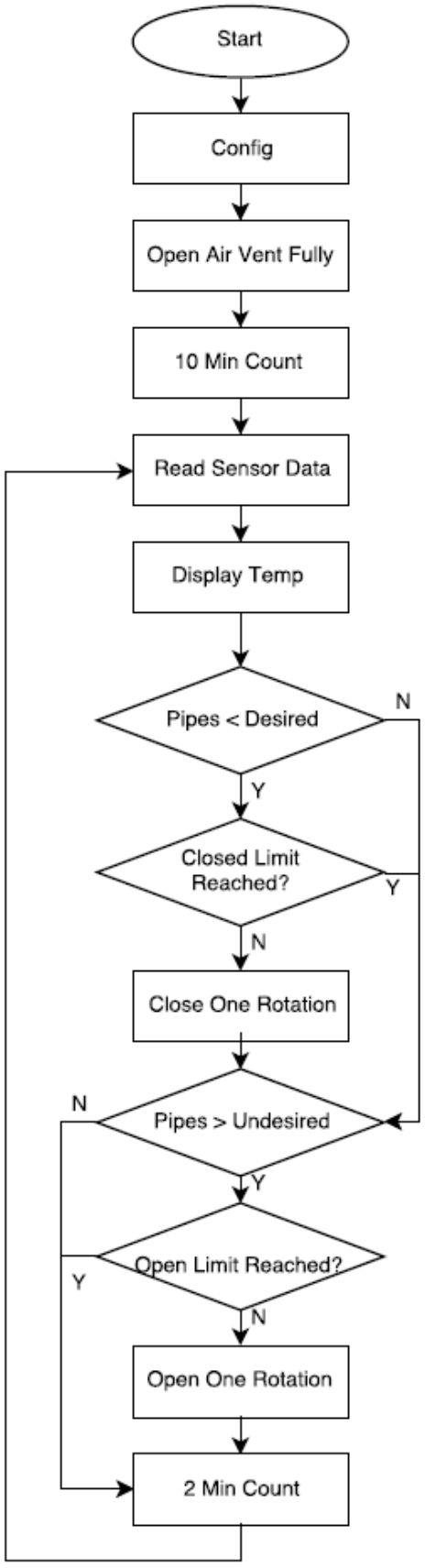
Bipolar stepper motor used on pins 0-3 of PORTB to control movement of air vent

B. LCD Display

The LCD will be used to display the return pipe temperature through the use of the ADC. Four data lines will be needed to display this and will be used through pin 0-3 of PORTD. The control lines RS, R/W and E will be used through pin 0-2 of PORTE.

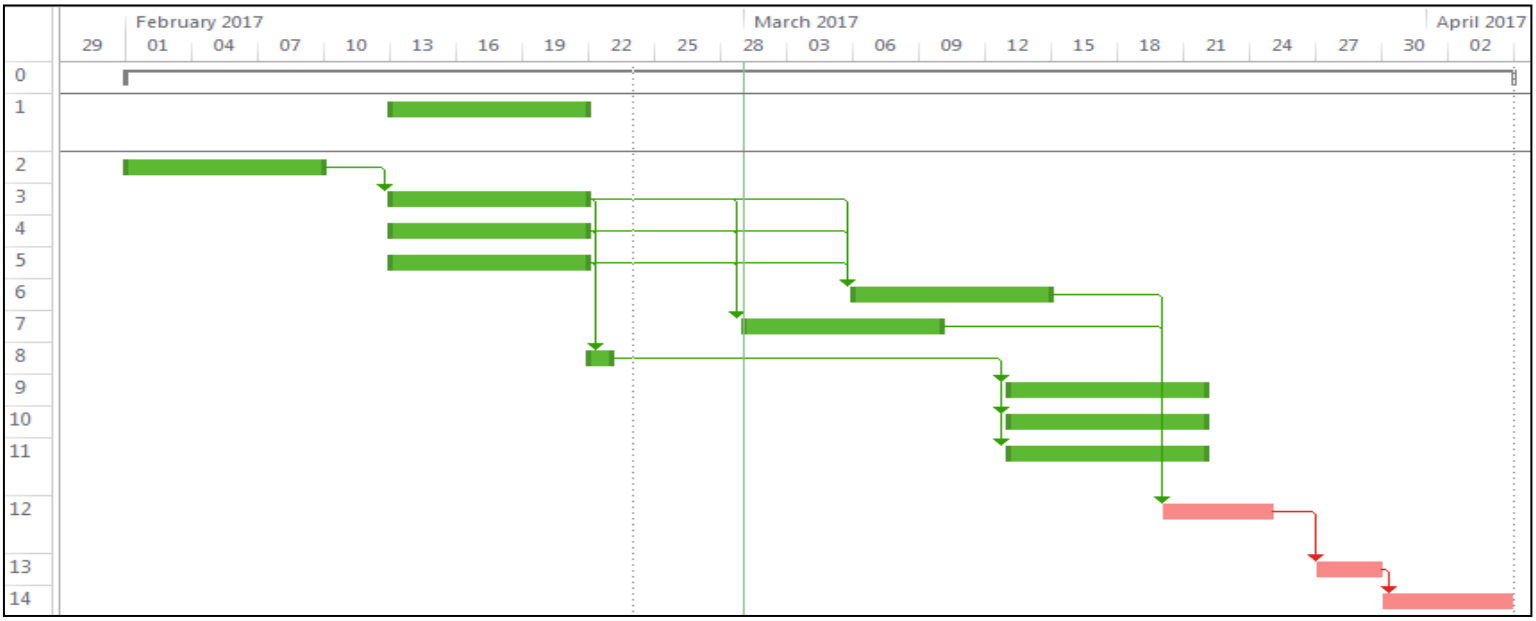
4.2 Flow Chart

This was the final flow chart created to implement the use of the temperature sensors in order to move the air vent. Seen in the diagram the air vent is open fully when started and given a delay of 10 min to give the stove time to transfer heat. In the first decision 'Pipes' is the Flow and Return pipes value subtracted by each other to reveal if the house has enough heat transferred. 'Desired' is the temperature difference at which we want the pipes to be at and 'Undesired' being the max temperature difference which if passed allows the vent to open for more air. The limits are the count value for the amount of rotations the vent has completed. The limit starts at 10 when fully opening the air vent.



5 Schedule

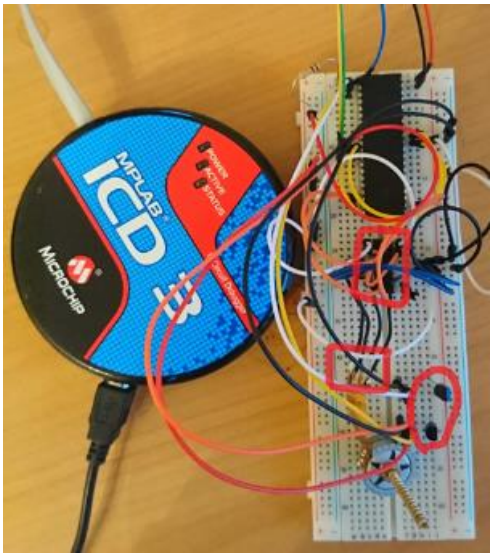
Work Tasks	Description	Duration
1	Look over previous notes on PIC18F452 and pick key functions needed for project	7 days
2	Research heat transfer based on the movement of heat within pipes	7 days
3	Interfacing temperature sensors with PIC	4 days
4	Interfacing stepper motor with PIC	7 days
5	Interfacing LCD with PIC	7 days
6	Purchase components that will be adequate for the project objective	7 days
7	Create circuit on proteus and simulate	7 days
8	Flow chart for program	1 days
9	Section of C program for temperature sensors	7 days
10	Section of C program for displaying temperature on LCD	7 days
11	Section of C program for precise movement of a bipolar stepper motor based on temperature values	7 days
12	Produce real-world circuit by connecting peripherals to their designated ports of the PIC18F452	5 days
13	Program real-world circuit	3 days
14	Revise program for improvements	4 days



6 Achievements

6.1 PIC18F4520

Through MP Lab I was able to program the PIC to read data from temperature sensors and based on these results change the state of LEDs and a stepper motor.



6.2 LM35D

The temperature sensors were successfully programmed to alter the state of components when both sensors

```
OpenADC( ADC_FOSC_RC & ADC_RIGHT_JUST & ADC_4_TAD,ADC_CH0
 & ADC_INT_OFF & ADC_VREFPLUS_VDD & ADC_VREFMINUS_VSS,0);
SetChanADC(ADC_CH0);     // select channel
Delay10TCYx(50);         // acquisition time
ConvertADC();            // Start conversion
while( BusyADC() );      // Wait for completion
flowpipe = ReadADC();    // Retrieve Measured Temp
Delay10KTCYx(50);
CloseADC();
```

values were subtracted and compared with the desired temperature difference. The sensors were first used to light LEDs to mimic half stepping movement of the a motor.

```
if(DIFF > BigDiff && limit<10)
    {
        limit++;
        OPEN_VENT();
    }
if(DIFF < SmallDiff && limit!=0 && limit>0)
    {
        limit--;
        CLOSE_VENT();
    }
for(minutes=0;minutes>1;minutes++)
    {
        for(seconds=0;seconds>29;seconds++)
            {
                Delay10KTCYx(200);
            }
```

6.3 Bipolar Stepper Motor

The bipolar stepper motor was simulated on Proteus VSM using the PIC18F452 and L293D. The code produced was able to give a 45°C movement in both directions within the simulation and was ready to be used with a real world stepper motor.

Two stepper motors were extracted from individual floppy disk drives. The first stepper motor had the ribbon wire damaged and soldering was attempted on the motor itself. The outcome was unfortunate as the motor was not operational. The second motor was extracted carefully with the ribbon wire

```
void OPEN_VENT(void)
{
    PORTD = 0x04;        //Starting Point
    Delay10KTCYx(100);   //1s Delay
    PORTD = 0x05;        //Move 45 Degrees
    Delay10KTCYx(100);
    PORTD = 0x01;
    Delay10KTCYx(100);
    PORTD = 0x09;
    Delay10KTCYx(100);
    PORTD = 0x08;
    Delay10KTCYx(100);
    PORTD = 0x0A;
    Delay10KTCYx(100);
    PORTD = 0x02;
    Delay10KTCYx(100);
    PORTD = 0x06;
    Delay10KTCYx(100);
    PORTD = 0x04;
    Delay10KTCYx(100);
    PORTD = 0x00;
    Delay10KTCYx(100);
    return;
}
```

and the soldered ends. These were cut into the individual ribbon cables and soldered onto wires. When programming this motor there was small movement in one direction but not in the 45°C movement anticipated.

```
void CLOSE_VENT(void)
{
    PORTD = 0x04;        //Starting Point
    Delay10KTCYx(100);   //1s Delay
    PORTD = 0x06;        //Move 45 Degrees
    Delay10KTCYx(100);
    PORTD = 0x02;
    Delay10KTCYx(100);
    PORTD = 0x0A;
    Delay10KTCYx(100);
    PORTD = 0x08;
    Delay10KTCYx(100);
    PORTD = 0x09;
    Delay10KTCYx(100);
    PORTD = 0x01;
    Delay10KTCYx(100);
    PORTD = 0x05;
    Delay10KTCYx(100);
    PORTD = 0x04;
    Delay10KTCYx(100);
    PORTD = 0x00;
    Delay10KTCYx(100);
    return;
}
```

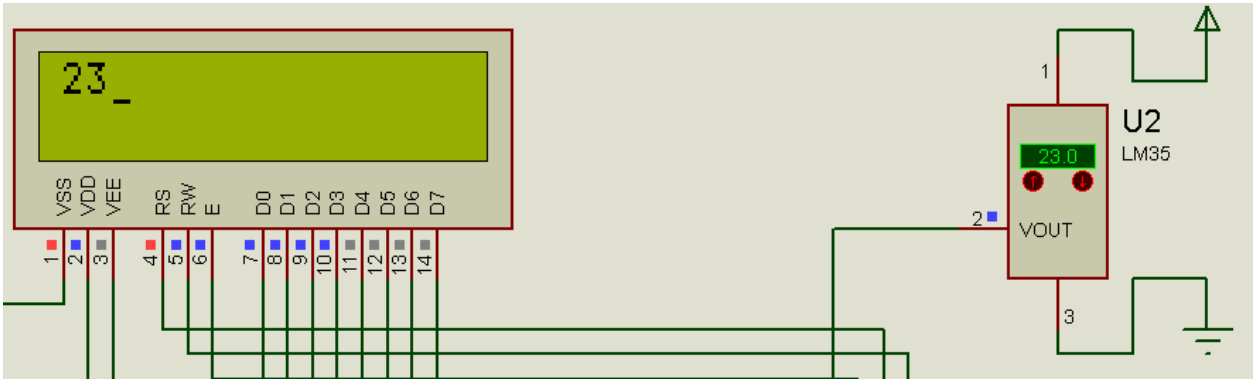
The code below was produced to be used with the real world circuit and was the same code used in the simulation minus the “Return Pipe” and “Temp” displaying. The code was able to build and be programmed to the PIC however the real world LCD could not display the reading.

```
OpenADC( ADC_FOSC_RC & ADC_RIGHT_JUST & ADC_4_TAD,ADC_CH0
& ADC_INT_OFF & ADC_VREFPLUS_VDD & ADC_VREFMINUS_VSS,0);
SetChanADC(ADC_CH0);     // select channel
Delay10TCYx(50);         // acquisition time
ConvertADC();            // Start conversion
while( BusyADC());       // Wait for completion
flowpipe = ReadADC()/2;      // Retrieve Data
Delay10KTCYx(50);
CloseADC();
Delay10TCYx(50);


while(BusyXLCD());       // Wait if LCD busy
WriteCmdXLCD(LCDclear); // Clear display
putrsXLCD((const rom char*) "Return Pipe" );
SetDDRamAddr(0x040);     // moves cursor to second line
putrsXLCD( "Temp:" );
ultoa( flowpipe, str );  // convert return pipe temp to string
while (BusyXLCD());
putsXLCD( str );         // display return pipe temp
Delay10KTCYx(100);
```

6.4 LCD Display

Previous work from the LCD helped with attempting to display a value from the sensors. Through this work I was able to simulate a value from a temperature sensor onto the LCD as shown below.



7 Testing – Results

7.1 Temperature Sensors

[2]LM35D was the sensor used in this project to measure the temperature. Ranged from 0°C to 100°C with 1 = 10mV. The sensors had just one output and needed a supply voltage. Two of these sensors were used with AN0 and AN1 pins on PORTA for analogue to digital conversion. This was first tested with MP Lab and Proteus VSM to simulate the use of the sensor and once i received the components the sensors were physically tested.

With the 18F4520 there was a bit of tweaking in the programming of the ADC which was different to the ADC programming of the 18F452 but once adjusted the temp sensors worked accordingly with producing measurements. On the day at which i tested the sensors they were reading 233mV on the DMM suggesting the temperature was 23°C within the room. Debugging the program to read the values received through the ADC both sensors read a 46 decimal value.

7.2 Bipolar Stepper Motor

To move the motor I first had to research how exactly to control the movement of the stepper motor. Precise control was done through the use of four coils. These coils heat up based on signals sent through them and attract the magnet which moves the motor. With this info i found that the motor could be moved through half stepping which allows for 45° movement. The L293D driver was chosen to drive the direction of the motor for opening and closing the vent.

Firstly I simulated the motor with the PIC18F452 on Proteus VSM and half stepped the motor successfully. With the physical circuit LEDs were used with the LM293D before acquiring the motor and showed the powering of the coil in the half stepping motion. Two motors were removed from individual floppy disk drives. The first motor was poorly soldered and couldn't be used. The second motor was soldered more delicately with the ribbon wire attached however when setting up the motor with the circuit one wire detached and had to be manually held against the input when motor movement was in progress. The motor was able to move but it wasn't in 45°C. It gave little twitches when being programmed to move. This could be for two reasons, either the wiring was incorrect or the extracted motor had two coils and not four.

7.3 LCD Display

In the module 'Firmware programming' I had already simulated a LCD Display with the PIC18F452 and had this simulation and module notes within my USB device. The LCD was simulated with the ADC of the PIC and by reviewing the notes and program I retrieved what was needed from the past work and implemented it into this project.

Using the simulation of the LCD with the ADC I changed the POT's being used with the LM35 sensors and displayed their read value on the LCD successfully. Displaying the correct temperature however was more difficult. The real world LCD was soldered and worked however I could not get anything to display. Another LCD was used which was operational however again could not get it to display therefore the program for the simulation could not be implemented onto the real world LCD. The program as well as the header was

altered to comply with the PORTS and PIC I wanted to use but nothing else could be found to suggest why the LCD wasn't displaying the sensors data.

8 Project Progress Assessment

The schedule was never kept to once the flow chart was started. Giving a day to do the flow chart was very optimistic as it turned out to be about 5 days thinking about what ways to program temperature sensors to activate the motor movement. Another fault with the schedule was planning to complete programming of the sensors, LCD and motor all in one week. These should have really been all spaced out within 3 weeks to allow for certain parts to be focused on instead of completing the program in one go. The idea of simulating the circuit on proteus was removed and instead went straight to programming the circuit using an ICD3. Simulating the different parts of the circuit was very successful however only the temperature sensors worked as required on the physical circuit. The motor movement was mimicked correctly with LEDs however the Motor wasn't capable of performing 45°C angle movement. LCD could display on simulation but not on physically circuit due to complications with program being used which was created for a different LCD

9 Next Tasks

Possible developments to this project can be to implement a carbon monoxide detector to open the air vent when detection occurs allowing enough air in the stove to stop carbon monoxide from being produced. Another idea would be to have two modes which could be toggled with a button. One mode would be what the program is doing now, comparing the flow pipe and return pipe to alter the air vent. The second mode would then add an additional temperature sensor to read the temperature of the room in which the stove is situated and change position of the air vent based on a threshold value which can be altered to a desired value.

10 Conclusion

The project was successful with moving a stepper motor through the use of two temperature sensors however the movement wasn't as preferred. It was also very disappointing that the real LCD wouldn't display a value for the sensors and I feel if there was more time this problem could have be resolved. Simulations and programming were both satisfactory but the program could have been improved if I got the motor and LCD working appropriately in time. Due to MP lab IDC3 not working on my laptop this also limited me to working on the real world circuit in college with limited computers having the software with c compiler. The flow chart and problems persisting with the hardware were the major draw backs with the project and with extra time the final project could have been accomplished.

REFERENCES

[1] https://www.youtube.com/watch?v=TWMai3oirnM

[2] http://www.hep.upenn.edu/SNO/daq/parts/lm35dm.pdf