**Abstract:**

Design and implementation of a power electronics circuit which converts fixed DC input voltage to AC output voltage whose frequency and amplitude can be appreciably varied and adequately controlled.

**Introduction:**

In this project, we designed an inverter that varies the properties of input DC voltage to AC voltage. A microcontroller was used to control the frequency and voltage. A buck converter was used to vary the amplitude of the output waveform. Full bridge configuration was used for the inverter implementation. Efficient coding was done to reduce the effect of harmonics and increase the efficiency and power factor.

**Complete Design Details and Performance Analysis:**

Components:

* IRF-540N (N-channel MOSFET)
* IRF-9405 (P-channel MOSFET)
* TIP 211 (Transistor)
* C945 (NPN Transistor)
* TLP 521 (Opto-Coupler)
* Arduino Microcontroller
* Resistances, Wires, Heat Sinks and Potentiometers

Circuit Design:

We implemented two circuits, one with opto-couplers without using a buck converter, and the other without using opto-couplers with a buck converter. The second technique yielded better results and amplitude control was implemented which was not achieved using technique one.

The Proteus circuit design is attached in the appendix A for both the techniques.

The basic inverter circuit consists of a H-bridge having P-channel MOSFETs at the high side and N-Channel MOSFETs at low side. The difference between two techniques is the circuitry used to turn on the MOSFETs. Two transistors (1 optocoupler and 1 NPN) in Darlington pair configuration are used to turn on the MOSFETs in technique 1. We are directly switching the NPN from Arduino in technique 2, which turns on the MOSFETs.

Sample Output Waveform:

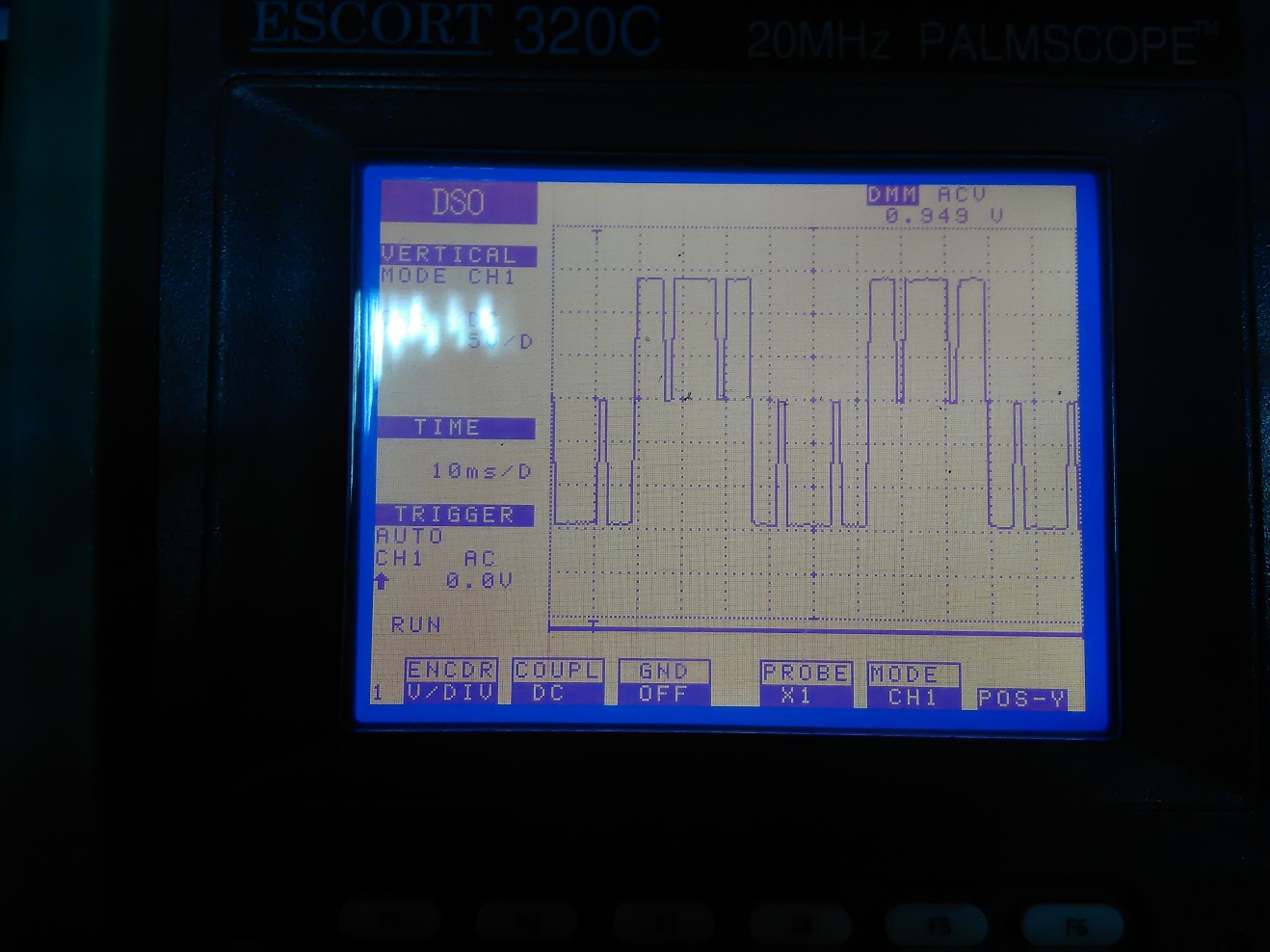


Fig.1: P=3 (Technique 2)

This is for P=3. We removed 2p-1=5 harmonics in this method. The efficiency was 96% and hence a high power factor. This technique proved quite efficient and was operated and verified at full load during testing.



Fig.2: P=7 (Technique 2)

This is for P=7. We removed 2p-1=13 harmonics in this method. The efficiency was 81%. There is a trade-off as removing more harmonics has resulted in more power losses, and hence lower power factor.

Variation in Amplitude and Frequency:

In technique one, we only varied the frequency. Arduino was used for this purpose in conjunction with a potentiometer. The 0-1023 analog values of the potentiometer were mapped into 10-200 divisions in Arduino. The range of frequency was 10 Hz to 285 Hz in both techniques.

The screenshot of Arduino code shows how the frequency control was achieved:

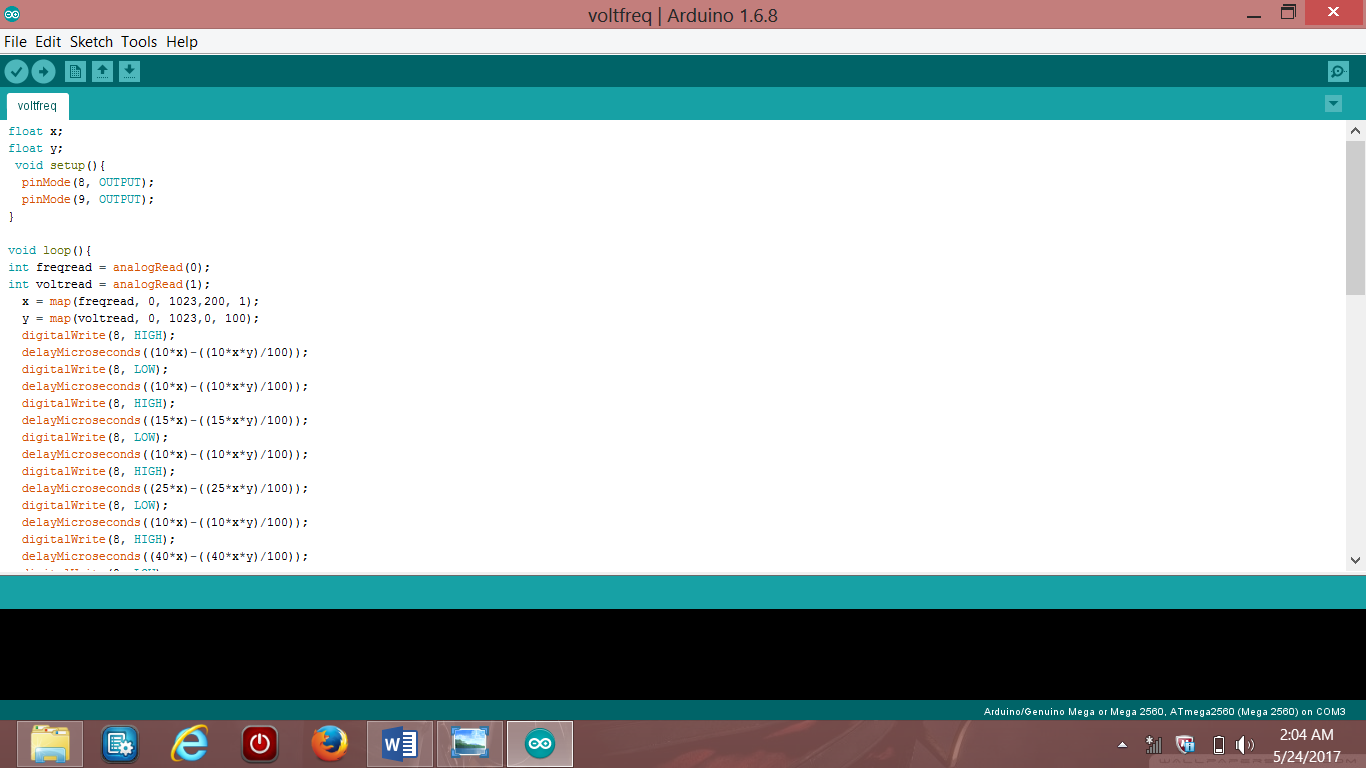


Fig.3: Code for frequency control

The analogread command reads analog input from the potentiometer at pin A0 and map function maps the value to variable x which is from 10-200 divisions as per requirement. The changing value of x changes the on time and off time of the pulses, which changes the time period of whole signal and hence the frequency is changed. (The complete code is given in Appendix B.)

The voltage in technique two was varied and controlled using a buck converter and Arduino.

The buck converter varied the amplitude. The buck converter is an efficient way of stepping down the DC Voltage. We supplied constant 15V at the buck input and achieved varying output 4-15 V DC that was fed to the inverter.

The Arduino controlled the voltage by varying the VRMS by chopping the driving signal.

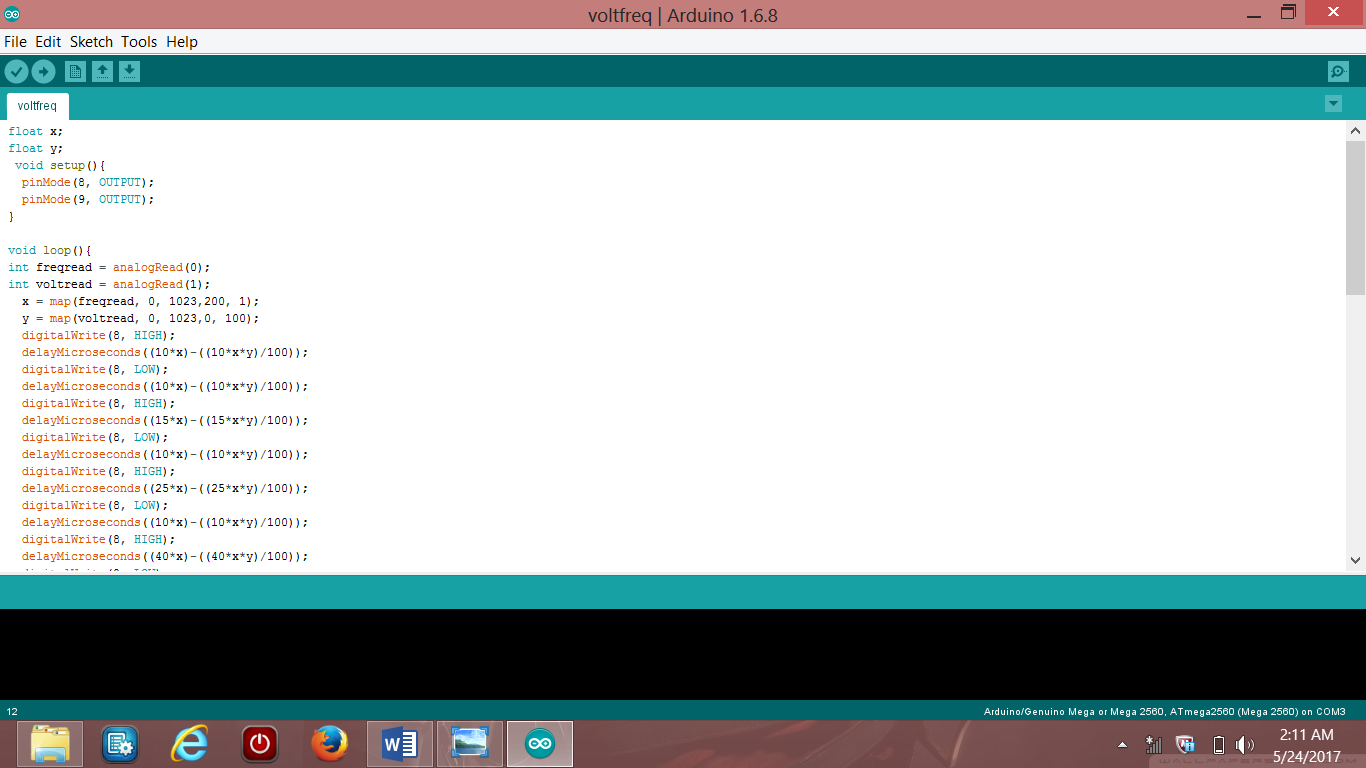


Fig.4: Code for voltage control

The voltread command took input from pin A1 about the required voltage and mapped it to the corresponding value as shown in code. The range achieved was 6.7-12.3 V. (The complete code is given in Appendix B.)

Harmonics Reduction:

We use SPWM (sine pulse width modulation) technique to reduce the harmonics. This varies the width if each pulse in proportion to the amplitude of sine wave evaluated at the centre of same pulse. If pulses in half the cycle are p, this technique removes 2p-1 harmonics from the wave, making it more sinusoidal. This also reduces the distortion factor greatly. And we achieved this using efficient coding from Arduino as shown in the Appendix B.

Power Comparison:

Table 1: Technique 1

|  |  |
| --- | --- |
| Technique 1 | P=7 f=50Hz R=75Ω |
| Vin | 15 V |
| Iin | 200 mA |
| Vout | 12.3 V |
| Pin | 3 W |
| Pout | 2.02 W |
| Efficiency | 67 % |
|  |  |

Table 2: Technique 2

|  |  |
| --- | --- |
| Technique 2 | P=7 f=50Hz R=75Ω |
| Vin | 15.4 V |
| Iin | 154 mA |
| Vout | 12 V |
| Pin | 2.37 W |
| Pout | 1.92 W |
| Efficiency | 81 % |
|  |  |
| Technique 2 | P=3 f=50Hz R=75Ω |
| Vin | 15 V |
| Iin | 178 mA |
| Vout | 14.04 V |
| Pin | 2.67 W |
| Pout | 2.61 W |
| Efficiency | 97 % |
|  |  |

Power Loss Areas:

In technique one, power was mainly lost in driving opto-couplers.

Technique two was quite efficient in terms of switching losses.

There were general losses due to breadboard resistances, MOSFET switching and heating of load resistance.

Issues Encountered:

We initially used 4 P-channel MOSFETs in the design but the circuit did not work due to large switching losses.

We moved to MOSFET driver circuits but this was also not much efficient way.

Then we finally moved to the combination of N & P channel MOSFETs and efficiency improved.

Opto-coupler for switching still gave some considerable losses.

Then we finally used C945 transistors in place of opto-coupler for switching and this gave us great efficiency exceeding 95%.

**Conclusion and Future Work:**We wish to make the output waveform purely sinusoidal in the future.   
And we also desire to design the inverter for inductive and capacitive type of loads.

This project helped us understanding the concepts of H-bridge, PWM and MOSFETs.

It cleared our theory concepts a lot.