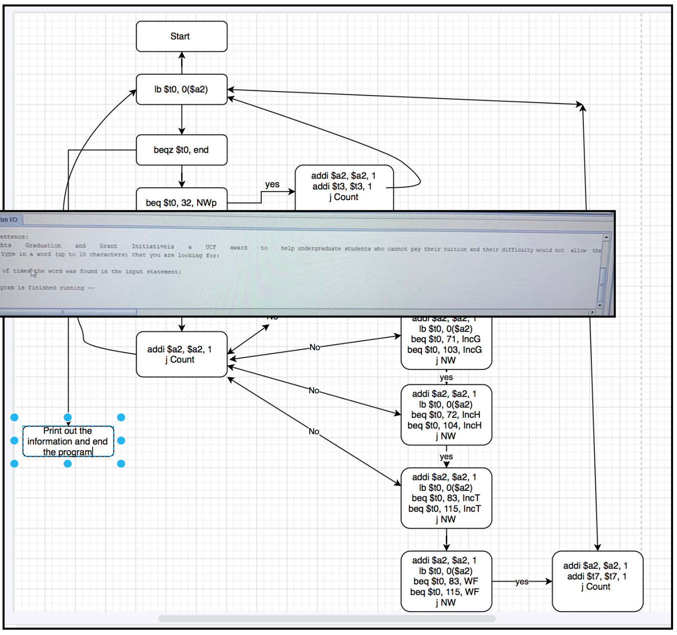
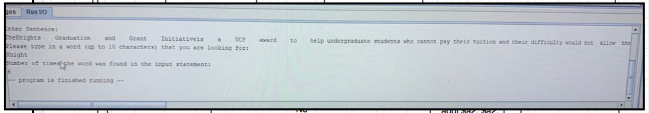
<DBN LAYERING: Stochastic And Latent Variables>

Abel Assefa
Department of Electrical and Computer Engineering
University of Central Florida
Orlando, FL 32816-2362

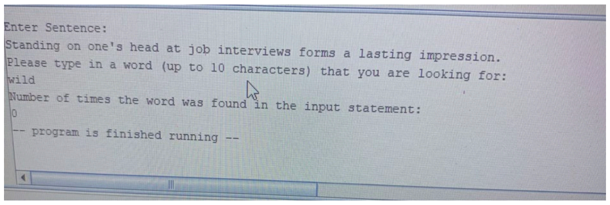
***Abstract*— Through this report the understanding and the use of energy consumption as well as the understanding of DBN (Deep Belief Network). DBN is a concept of learning and hierarchy of multiple layers and variables that are both random that are referred to as stochastic variables and another input variable that is often a hidden. binary value called latent variables. The structure of the DBN follows a top down structure with each layer receiving from the previous. Through this the understanding of how DBN's work can be derived through mathematical expressions using a a restricted Boltzmann Machine. Lastly in terms of energy consumption, Design 1 provided the least amount of energy being outputted.





```
Enter Sentence:
TheKnights   Graduation   and   Grant   Initiativeis   a   UCF   award   to   help undergraduate students who cannot pay their tuition and their difficulty would not  allow  the
Please type in a word (up to 10 characters) that you are looking for:
Knight
Number of times the word was found in the input statement:
6
-- program is finished running --
```

Keywords— DBN (Deep Belief Network), Stochastic Variable, Latent Variable, Boltzmann Machine, Energy Consumption, Hierarchy, Hidden binary Value, Layers

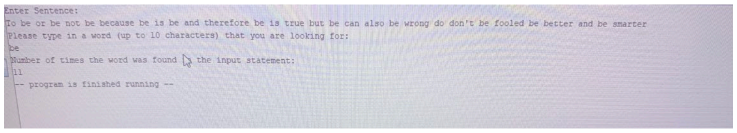

```
Enter Sentence:
Standing on one's head at job interviews forms a lasting impression.
Please type in a word (up to 10 characters) that you are looking for:
wild
Number of times the word was found in the input statement:
0
-- program is finished running --
```

I. Introduction

A. *Project Design*

The objective of the assembly code was to take a user input sentence and find all instances of a specified word by the user and return the number of occurrences of said word along with the index position of the word in the sentence. In the .data section there was the creation of the labels that would be used in guiding the framework of the output, things such as the questions asking the user for the input sentence and word. Along with that there was the label for the sentence that would tell the user the information that they seek in a presentable format.

Following the .data section is the .text which starts by reading the string the user put for the sentence as well as the word of interest into two different registers. Following this is the Label Count that holds all the conditional branches, the first thing done here is a lb instruction to get the first bit of the input sentence. Following this is a series of beq instructions with the first one being a conditional to branch to the end if the sentence is done along with that is conditional statement to check if the current bit is in a space character and if so to move the bit to the next bit in the sentence. After running through those conditional statements, another beq instruction is used to check if the current bit is the first letter of the specified word, if this is true it proceeds to the next label. This is done until the final word of the specified word is reached and that last conditional statement branches to the label WF which states the the word has been found and increments the counter holding the number of occurrences, along with adding the current index of the word in


```
Enter Sentence:
To be or be not be because be is be and therefore be is true but be can also be wrong do don't be fooled be better and be smarter
Please type in a word (up to 10 characters) that you are looking for:
be
Number of times the word was found   the input statement:
11
-- program is finished running --
```

a blank register and finally adds one to the bit counter to continue along the sentence. If at any point through this process the bit is not equal, then it jumps to NW which increments the bit value.

B. *Test Cases*

There were three different inputs chosen for this code. The first input was a test input defined by the rubric to find the word knight in a sentence, this was first used to show the desired output was being attainted and the code was functioning smoothly. Next a phrase was used with more than 10 occurrences of a specific word, the phrase contained the word of interest every other word. This was done to ensure that the branch statements were operating successfully and accurately. Finally the last case was one of a phrase that did not contain the desired word at all, this again was to test that the code was functioning adequately and not returning false values.

Table I: Energy consumption for a branch instruction in the designs provided in [1-4].

Design	Energy Consumption For Each Branch Instruction
[1]	0.2 pJ
[2]	$0.03e^{+5}$ pJ
[3]	$6e^{+5}$ pJ
[4]	$5e^{+5}$ pJ

II. DBN CIRCUIT

The Deep Belief network is a a structure of multiple layers of variables that are random based on a probability or a hidden variable. The inputs in the DBN are referred to as stochastic and latent variables. The stochastic variable is that of a more randomly determined occurrence while the latent variable is often a hidden binary value. The structure of the DBN is multiple layers with the top layers specifically the top 2 having a connection between them forming a memory of sorts. Following these layers are the lower layers that receive orders from the top and moves down with the lowest layer containing some sort of data. The Structure of the DBN and the layer by layer process it goes through, allows for a high level of control and efficiency to certain degree. In the layers due to the top-down nature, each layer is built upon based on the previous layers variables and this will continue on until the final layer is reached. This final layer is represented usually as the data vector which incorporates all the variables and data passed through each layer, however with certain measures there can be forced variable insertions to better improve data spreads and the efficiency of the circuit as a whole

The big understanding of DBN's are though the variables that are seen and the unseen. DBN's are typically understood more through the understanding of a restricted Boltzmann Machine. Now this machine is a collection of connected units that make probable and statistically decisions on whether or not to turn on or off. This machine is typically a two layer system with seen and unseen units composing of these layers and there is a connection by these two layers through a symmetrical weight, W, concept. This concept leads to the mathematical works of the DBN's as knowing the weight allows the use of the summation of the weight and general distribution and derive and understand the visible and hidden vectors over a given distribution layer.

III. RESULTS AND DISCUSSION

*All values kept in pJ

Design 1 Energy: (2*2876) + (0.2*2563) + (4*1288) + (100 *745) + (5 * 9)

Table II: Total Energy consumption for the assembly program using designs provided in [1-4].

Design	Total Energy Consumption
[1]	85961.6 pJ
[2]	96860.4878 pJ
[3]	2367746.561 pJ
[4]	1987363.634 pJ

Design 2 Energy: (2*2876) + (0.03e^5*2563) + (4*1288) + (100 *745) + (5 * 9)

Design 3 Energy: (2*2876) + (6e^5*2563) + (4*1288) + (100 *745) + (5 * 9)

Design 4 Energy: (2*2876) + (5e^5*2563) + (4*1288) + (100 *745) + (5 * 9)

Using the instruction statistics in the MARS program, the total energy of the ALU, Jump, Branch, Memory and other is recorded by this feature and outputted. Using the energy value given in this section along with the branch information in table 1, the total energy of the code can be calculated. Each design option has a differing branch value associated with it, and the results can be seen in table 2. In the case of this code the Design with the lowest energy consumption is Design 1. This can be expected because this Design has the lowest value for the branch instruction and in this code the branch instruction is utilized frequently

IV. CONCLUSION

Through this paper many technical skills were developed and or enhanced through this process. The skills were

1. Better utilization and understanding of MIPS assembly language
2. Understanding and calculation of energy consumption and dissipation
3. Understanding the use of DBN's along with learning and understanding the differences in the variable types
4. Deriving and understanding DBN's through mathematical concepts and with the idea of the restricted Boltzmann Machine
5. Ability to parse through research, and express said knowledge in written form

Lastly the design with the best energy is Design 1 with an efficient 85961.6 pJ

REFERENCES

1. H. Pourmeidani, S. Sheikhfaal, R. Zand, and R. F. DeMara, "Probabilistic Interpolation Recoder for Energy-Error-Product Efficient DBNs with p-bit Devices," IEEE Transactions on Emerging Topics in Computing, 2020.
2. A. Roohi, S. Sheikhfaal, S. Angizi, D. Fan, and R. F. DeMara, "ApGAN: Approximate GAN for Robust Low Energy Learning from Imprecise Components," IEEE Transaction on Computer, vol. 69, no. 3, 2020.
3. D. Le Ly and P. Chow, "High-performance reconfigurable hardware architecture for restricted boltzmann machines," IEEE Transactions on Neural Networks, vol. 21, no. 11, pp. 1780–1792, 2010.

4. A. Ardakani, F. Leduc-Primeau, N. Onizawa, T. Hanyu, and W. J. Gross, “Vlsi implementation of deep neural network using integral stochastic computing,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, no. 10, pp. 2688–2699, 2017.

5. Geoffrey E. Hinton (2009) Deep belief networks. Scholarpedia, 4(5):5947

6. A. Mohamed, G. E. Dahl and G. Hinton, "Acoustic Modeling Using Deep Belief Networks," in *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14-22, Jan. 2012.