Representation of Read Operation of Memory Bit-Cells in Word Counting Program for a Set String

Christopher Monks
Department of Electrical and Computer Engineering
University of Central Florida
Orlando, FL 32816-2362

Abstract— **Read Memory is based on the fluency of a program, how often it is accessed, and by what means is the read address stored. The objective is to identify different designs that can be used in read memory and their effect on the energy requirement of the program. The code is from Project 3, a task to create a program that can read a preset string, and identify the amount of times a word from the keyboard appears in it. The only stipulation for the input word is that it cannot contain spaces, and it cannot be larger than nine characters. The inputs consist of a hardcoded string and the user input once the program is run. Once it has concluded, it should print the value of the word's appearances. After numerous tests, the total energy consumption of the design that used the least amount of energy, the EASA (Energy Aware Sense Amplifiers), was 115340.24 fJ.**

Keywords—syscall, directive, register, branch, null, asciiz, character, string, MRAM, anisotropy,

I. Project Design

The program starts by designating a string to a label with an asciiz directive. Using syscall, a string is read from the keyboard and placed in a space created for that label with a capacity of ten characters. The strings are then loaded into addresses which are then copied to separate registers to be referred to later. A series of loops are initiated that load bytes into a register, checks whether they're upper case, and if they are, turns them lower case by taking their values and increasing them by a constant of 32. This allows the program to count words from the string regardless of whether they are capital or not. This procedure is done for both the string and the input. To count the amount of times the inputted word appears in the given string, the first bytes of string and input are loaded and checked for similarities. If the values are not similar, then the address of the string is incremented by one while the address of the input stays the same. The code loops until a similar value is found. Once one is, a branch is initiated that increments the address of the string and input in tandem, checking whether the letters match until the input reaches its end. If they do, then a counter is incremented and the addresses are set back to their original places, except the address of the string is increased by one in order to avoid an infinite loop and keep the program going. This process

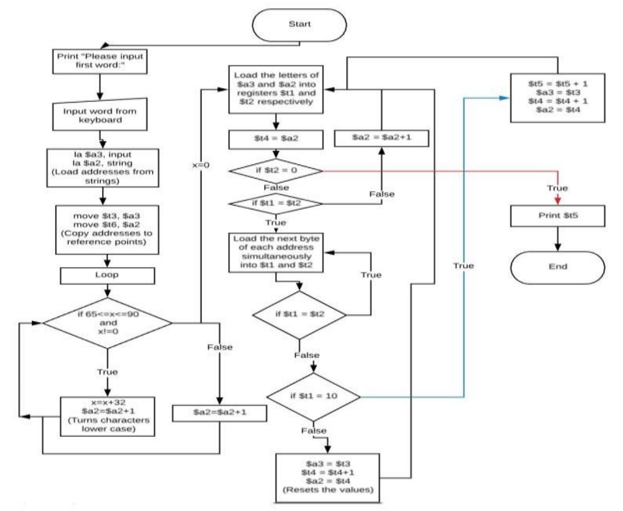
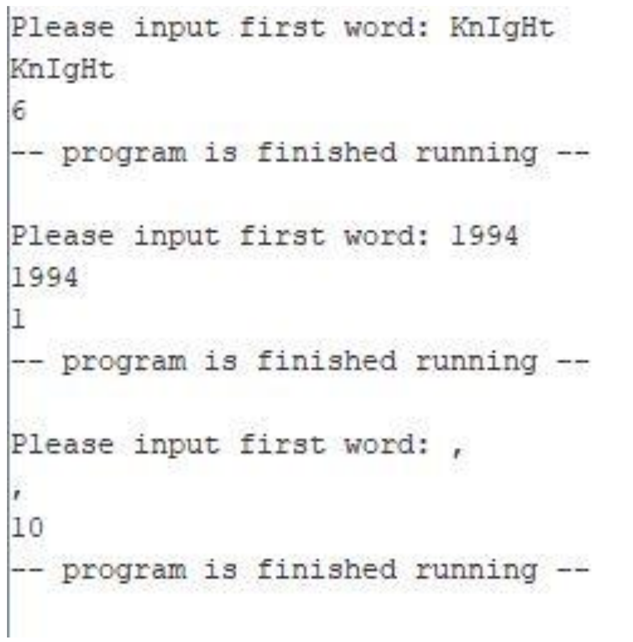


Fig.1: Flowchart of the assembly program.



```
Please input first word: KnIgHt
KnIgHt
6
-- program is finished running --

Please input first word: 1994
1994
1
-- program is finished running --

Please input first word: ,
,
10
-- program is finished running --
```

Fig.2: Sample outputs of assembly program.

continues until the string reaches a null character, then the final count is printed.

To test the program, three inputs were entered to test the accuracy of the code. The first input was “KnIgHt”, used to test the ability of the program to convert the word and string into lowercase letters, and then count the number of appearances that word makes in the string. The output concluded with an amount of six, which is the proper number of appearances any variation of the word “knight” makes in the string. The second input, “1994” is used to test that no errors are made in checking numbers to the string. When the letters in the string are converted to lowercase, 1994 is not altered in any way, allowing for the following correct output of one. The final input is a simple comma, which achieves two tasks; confirming the program works with single byte inputs, and that punctuation is allowed to be checked by the code. With a final value of ten, it is conclusive that all steps in checking words to the string are running as they should.

II. Memory Bit-Cells

Energy Aware Sense Amplifiers and Variation Immune Sense Amplifiers achieve similar outcomes. In a VISA circuit, leaked energy is kept to a minimum by keeping the output of its inverters low so that the Transmission Gates will be off. [1] This is only in the case that the signal in the VISA is low. The EASA circuit acts similarly. [5] By turning the Transmission Gates off, leaked energy is reduced in the case of a low signal. For high signals on both the EASA and VISA circuits, there are no extra costs as the Transmission Gates are on, but other parts remain off to keep energy leakage at a minimum. [1]

The Preread and Write Sense Amplifier is used in Magnetoresistive RAM to increase read speeds and support nonvolatility. Although it has not been perfected yet, making use of spin torque transfer in combination with MRAM is said to increase these read speeds even more. [4] [6] By using Voltage-Controlled Magnetic Anisotropy, memory cells experience a 10x size decrease of their access transistors. [2] The VCMA effect is achieved by using Magnetic Tunnel Junctions, allowing the PWSA circuit to utilize low switching energy by having a single pulse shape, accentuating its simplicity.

In the Body-Voltage Sensing Circuit, emphasis is placed on minimizing power usage instead of maximizing the read margin. However, BVSC still shows an overwhelming amount of RM over SPSC and CMSC at various levels of sensing times. [3]

Every value that is read out consists of a bit-line and a word-line. The bit-lines are represented as columns while the word-lines are represented as rows. Where the two intersect is the address of a given memory cell. In order to access a value to be read, the CPU needs to first access the memory location of that value. If the address already resides in the cache, then the words can be delivered immediately to the CPU. If it is not in the cache, then the main memory needs to be accessed and space in the cache needs to be reserved for the memory. Once the memory is loaded into the cache, the Read Address is delivered to the CPU. This second option is much less efficient as it takes much more time to search the main memory for an address rather than accessing a readily available cache.

III. Results and Discussion

To calculate the total energy output using each of the designs represented in Table II, the power consumption involving the read energy went uncalculated while the remaining instruction energies were added up to a single constant. The energies used in this calculation are as follows:

1) *ALU = 1 fJ*
2) *Branch = 3 fJ*
3) *Jump = 2 fJ*
4) *Write Energy = 50 fJ*
5) *Other = 5 fJ*

The total instruction count was tallied at 9992 using the word “kNiGhT” with overall values of:

1) *ALU = 3656*
2) *Branch = 3068*
3) *Jump = 1259*
4) *Memory = 1988*

Table I: Energy consumption for a single bit-cell read operation in the designs provided in [1-3].

Design	Energy Consumption For Each Bit-cell’s Read Operation
EASA [1]	0.23 fJ
VISA [1]	1.86 fJ
PWSA [2]	36.0 fJ
BVSC [3]	195.5 fJ

5) *Other = 21*

By multiplying the respective energies by their overall values and adding them together, the total energy before read energy is calculated is 114,883 fJ.

By using the above table for the read energy values involving different circuit designs, the read energy consumption by multiplying the memory instruction count by these values are:

1) *EASA= 457.24 fJ*
2) *VISA = 3,697.68 fJ*
3) *PWSA = 71,568 fJ*
4) *BVSC = 388,654 fJ*

By adding the final read energy values to the constant, the total energy consumption for each design is equivalent to those represented in Table II.

Table II: Total Energy consumption for the assembly program using designs provided in [1-3].

Design	Total Energy Consumption
EASA [1]	115,340.24 fJ
VISA [1]	118,580.68 fJ
PWSA [2]	186,451 fJ
BVSC [3]	503,537 fJ

IV. Conclusion

` The program was highly branch based, using multiple decision making procedures to arrive at the final value. When there are more capital letters for a single word in the user's input, the ALU and Memory values increase while all other instruction counts remain consistent. In this sense, the increase of capitals letters has an effect on the specific energy associated with the given designs. Other observations include that the EASA design consumes much less power than the remaining designs for read energy, while BVSC consumes the most. The total energy for EASA resulted in 115,340.24 fJ, about 4.37 times less than the BVSC circuit design.

References

[1] S. Salehi and R. F. DeMara, "Process variation immune and energy aware sense amplifiers for resistive non-volatile memories," 2017 IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, MD, 2017, pp. 1-4.

[2] H. Lee *et al*., "Design of a Fast and Low-Power Sense Amplifier and Writing Circuit for High-Speed MRAM," in *IEEE Transactions on Magnetics*, vol. 51, no. 5, pp. 1-7, May 2015.

[3] F. Ren, H. Park, R. Dorrance, Y. Toriyama, C. K. K. Yang and D. Marković, "A body-voltage-sensing-based short pulse reading circuit for spin-torque transfer RAMs (STT-RAMs)," Thirteenth International Symposium on Quality Electronic Design (ISQED), Santa Clara, CA, 2012, pp. 275-282.

[4] J. Kim, T. Na, J. P. Kim, S. H. Kang and S. O. Jung, "A Split-Path Sensing Circuit for Spin Torque Transfer MRAM," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 61, no. 3, pp. 193-197, March 2014.

[5] I. Arsovski and R. Wistort, "Self-referenced sense amplifier for across-chip-variation immune sensing in high-performance Content-Addressable Memories," *IEEE Custom Integrated Circuits Conference 2006*, San Jose, CA, 2006, pp. 453-456.

[6] S. Tehrani, J. M. Slaughter, E. Chen, M. Durlam, J. Shi and M. DeHerren, "Progress and outlook for MRAM technology," in *IEEE Transactions on Magnetics*, vol. 35, no. 5, pp. 2814-2819, Sep 1999.