

**A Platform Independent Improved GUI for the PeANUt
Computer Simulator**

AVINASH G PRASAD

Supervisor: Dr Peter Strazdins

Software Engineering Project(COMP 8790)
The Department of Computer Science
The Australian National University

TABLE OF CONTENTS

Abstract.....	3
1. Introduction.....	4
1.1 Overview.....	4
1.1.1 Background.....	4
2 Requirements.....	6
2.1 Optional Requirements.....	7
2.2 Assumptions and Constraints.....	7
2.3 Project Deliverables.....	7
2.4 Risk Analysis.....	7
2. Scheduling.....	7
2.1 Project Plan.....	8
3. Reference.....	10

Abstract

This is the project plan for the implementation of a new GUI for the PeANUt Computer simulator and also to make it platform independent and maintainable.

PeANUt is a simple microprocessor used for the teaching purposes at the Australian National University. Currently it works only with the Linux Operating System. Its functionality was written in C and the scripting was done using Tcl/Tk. The use of Tcl/Tk has created maintainability problems along with portability.

This project rewrites the GUI of PeANUt in Java making it platform independent so that it works in all the major Operating Systems. The artifact provided will also be more maintainable and any changes or additions to the software can be easily made.

Introduction

Overview

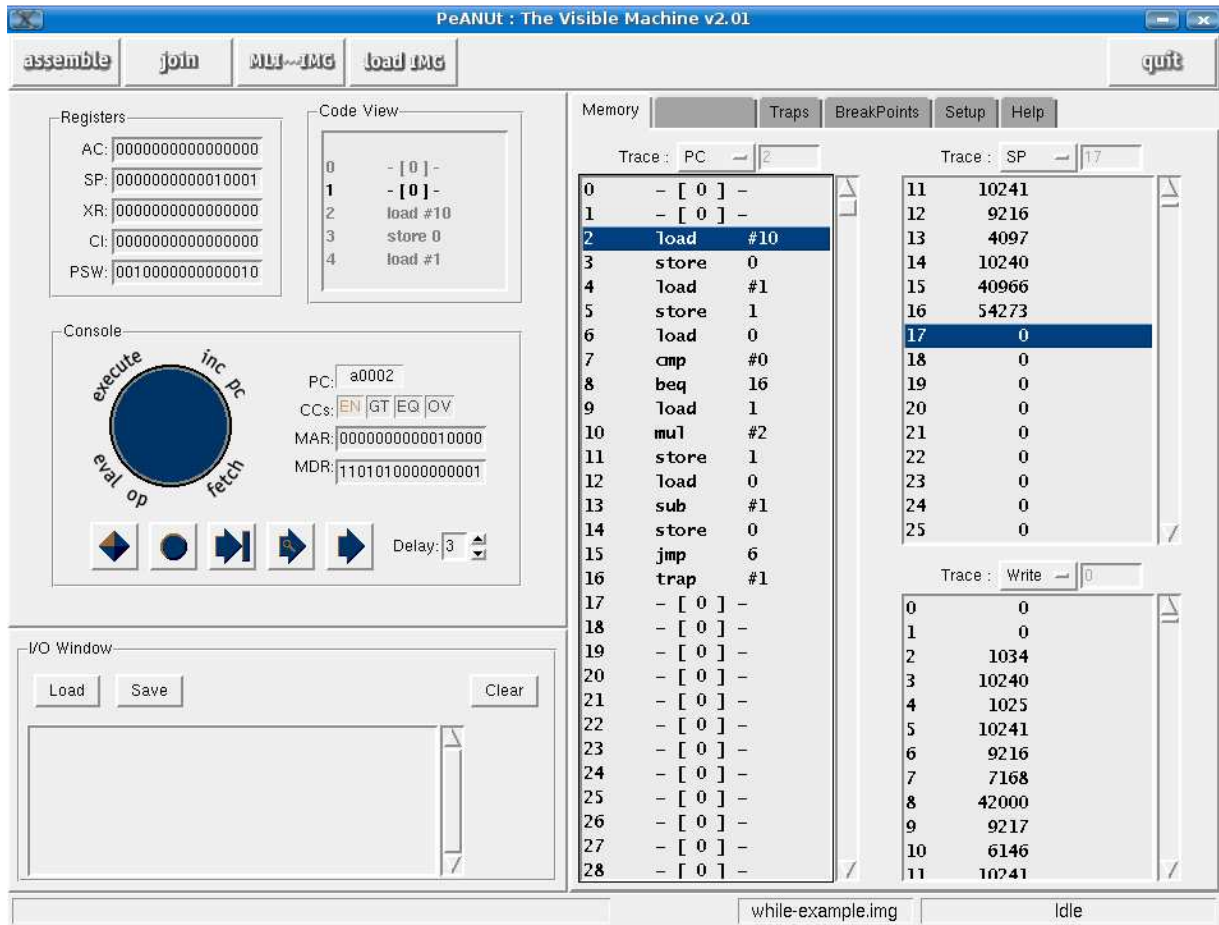
The GUI or the Graphical User Interface as we call it is a program's interface which takes the advantage of the computer's graphical capabilities to make the program easier to use. GUI Technology over the last couple of decades has seen steady incremental changes built on some core principles. Similar to other technologies, even the idea behind GUI was thought long before technology existed to build such a machine. Over the years GUI Technology has advanced considerably but the mouse interface has remained as the backbone. Much of the core functionality for GUI remains the same but the potential for adding new features remains limitless.

Background

PeANUt is a software microprocessor. It consists of a memory, 16 bit arithmetic and logical unit, an addressing unit, an execution unit, a primitive operating system, a set of 16 bit registers and connections to input and output devices. There are 102 memory cells, with address 0...1023. The PeANUt machine has 5 special registers. They are AC, CI, SP, XR and PSW.^[1]

PeANUt was developed in 1992 in the Department of Computer Science. The term PeANUt was coined in 1993. After 2002, PeANUt ceased working in the new Operating Systems. In 2005, a study concludes that GUI should be rewritten using Java or Python.

PeANUt's functionality is coded in C and it uses Tcl/Tk libraries for scripting and Tix for GUI.



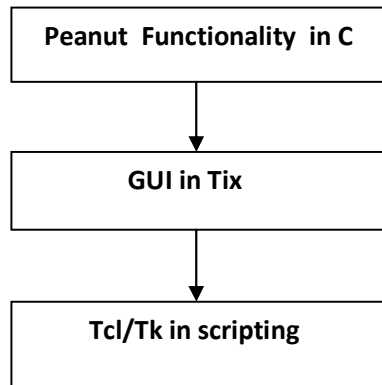
Example Program

```

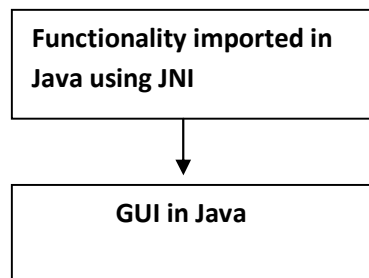
                                ; int main() {
n:      block      1            ;   int n;
pn:     block      1            ;   int pn;
start:  ;
        load       #10         ;   n = 10;
        store      n           ;
        load       #1          ;   pn = 1;
        store      pn         ;
while1: load       n           ;   while (n != 0) {
        cmp        #0         ;
        beq        endwh1     ;
        load       pn         ;       pn = pn * 2;
        mul        #2         ;
        store      pn         ;
        load       n           ;       n = n - 1;
        sub        #1         ;
        store      n           ;
        jmp        while1     ;   }
endwh1: trap       #1         ;   return 0;
        end        start     ;   }

```

Old PeANUt



New PeANUt



Requirements

1. Full support for Windows, Mac and Linux while maintaining the functionality of the PeANUt system.
2. Complete port of the GUI component of PeANUt in Java to make it platform independent.
3. Using Java Native Interface to import the C code in Java and maintain functionality.
4. Automated testing scripts like Abbot, Marathon to be used for GUI instead of manual GUI testing.

Optional Requirements

1. Create a better help file so that students can easily understand the functions of the software.
2. Display the registers in various formats (Decimal, Hexadecimal and the Octal System).
3. Automate the process of compiling the file instead of the user clicking on various tabs.

Assumptions and Constraints

1. It is assumed that the existing PeANUt functionality works and there will be no development or testing required confirming it.
2. The main developing language used will be Java and all development will be done using the Eclipse IDE.
3. The C code will be imported to Java and it is out of the scope of the project to do an actual translation of the code into Java.

Project Deliverables

1. A working artifact of the PeANUt microprocessor software and all the required documents.
2. Final Report and a presentation

Risk Analysis

1. Background research for GUI Design and Maintainability may require a long time as it is a specialized field in Application Design and Maintainability.

LOW RISK- To mitigate this risk, we can use the papers and resources available for Application Design and Maintainability (having a heavy emphasis on GUI) and follow a similar method followed in these papers .

2. Full code portability may not be possible i.e. Full C code may not be portable to Java resulting in partial or full implementation of the C code in Java.

HIGH RISK- From the early stages of the project, the parts which cannot be ported needs to be identified and if there is no alternative then we have to do an implementation in Java.

3. All the software aspects may not be tested using automated scripts. Some functionality may have to be manually tested.

MEDIUM RISK- We have to start writing test cases in parts along with the software implementation so that the parts which can be automated and which cannot be automated is identified.

4. Learning curve for implementing the automated test cases.

LOW RISK- Since this is an unknown area for me, finding good and useful resources may take some time

Project Plan

Week	Tasks	Notes
1-2	Requirements & Background research	<ul style="list-style-type: none">• Understanding requirements• Doing Risk Analysis• Background research on GUI•Decide Tools and software• Documentation

3-4	Modeling	<ul style="list-style-type: none"> • Analyze the existing artifact • Modeling using UML diagrams which consists of Use Cases, Class Diagrams
5-13	Implementation	<ul style="list-style-type: none"> • Coding for the PeANUt software which consists of creating a new GUI in Java and porting the functionality using JNI using software design methodologies • Testing consists of Automated test cases which automate the tests using scripts • Debugging the code and fix the errors
14-15	Report	<ul style="list-style-type: none"> • Completing the final report • Preparing slides for the final presentation

Approach

Many approaches were considered to fix this which included Python, Java or Jython. But eventually it was settled for Java as Java offers many advantages like more support for importing C codes using Java native Interface, multi platform compatibility and support for complex GUI.

The new version will be coded in Java with a new GUI using software design methodologies which make it more maintainable. All the previous functionalities will be imported in Java using Java Native Interface. JNI allows us to use the native code in Java without losing the functionality. To make the software more maintainable software design patterns like Facade, Decorator etc will be used.

Testing and debugging will be carried out using automated testing scripts. The testing scripts will be written using Abbot or Marathon.

Project Participants

Client and Supervisor- Dr Peter Strazdins

Developer- Avinash G Prasad

References

1. PeANUt Manual- The Australian National University
2. Java- www.java.sun.com
3. Eclipse- www.eclipse.org
4. Abbot- www.abbot.com
5. Marathon- www.marathontesting.com/
6. PeANUt website- <http://cs.anu.edu.au/student/comp2300/PeANUt/>