

**COMP2300**

- [Home](#)
- [StudyAt](#)
- [Assessment](#)
- [Schedule](#)
- [Lecture Notes](#)
- [Tute / Labs](#)
- [Assignments](#)
- [Announcements](#)
- [Discussion](#)
- [Reading material](#)
- [Old Exams](#)
- [Links](#)
- [PeANUt@Home](#)
- [Getting Help](#)

Quick Links

- [ANU Home](#)
- [FEIT Home](#)
- [DCS Home](#)
- [Search](#)

**COMP2300****Tutorial / Laboratory 04 - Introduction to PeANUt**

Semester 1, 2009

Week 5 (23 - 27 March)

Preparation

It is expected that you would have looked over your lecture notes of the PeANUt module as well as read the relevant parts of the *Specification of the PeANUt Computer* before doing this session.

Don't worry if you have just a partial understanding of PeANUt. It is the purpose of this session to help you acquire familiarity with the fundamentals of the PeANUt machine.

You should bring the COMP2300 reading brick *Specification of the PeANUt Computer* to this and all subsequent tutorial / laboratory sessions (as well as all the PeANUt lectures).

Tutorial Exercises

In the following box you can see a simple PeANUt machine language initialisation file called lab4a.mli.

```

; lab4a.mli - Simple PeANUt machine language initialisation file
;
START a10          ; initialise the PC to a10.

AT a10             ; from address a10 on, initialise with the following:
000 001 0 000 000 001 ; a10: this goes in a10
000 011 0 000 000 010 ; a11: this goes in a11
110101 0 000 000 001 ; a12: this goes in a12 (this instruction must always
; be at the end of an instruction sequence)

```

1. Examining a .mli file

All text to the right of the ;'s are comments. The PeANUt conversion to image file process ignores these comments, they are only there to help humans to read the file. The remaining information (the text to the left of the ;'s) will be used to initialise the PeANUt.

- The first line (START a10) will result in the PeANUt initialising its **PC** (program counter) with a10 (memory address 10_g, 10 octal).
- The **PC** holds the location (in memory) of the next instruction to be executed, so initialising the PeANUt's **PC** to a10 will result in the instruction at location a10 in memory being executed first when the machine is started.
- The second line (AT a10) will result in the PeANUt to initialise its memory cells from a10 onwards with the data in the lines that follow.
- The next three lines are the information (16 bits each) that the PeANUt will initialise the memory cells a10, a11 and a12 with.

2. Understanding PeANUt instructions

Write down the answer to the following questions.

In the program lab4a.mli, the information to be stored at memory locations a10 onwards are binary numbers (with 16 digits, i.e. 16 bits) that the PeANUt will interpret as instructions.

1. Open your PeANUt manual on page 3.
2. Since the **PC** (program counter) will be initialised to a10, when the PeANUt begins executing it will fetch the first instruction from a10. The data at a10 should thus be viewed as a PeANUt instruction. Examine the data that is to be used to initialise memory a10:

```
000 001 0 000 000 001
```

Note that the PeANUt ignores spaces, it treats "000 001 0 000 000 001" as "0000010000000001". The spaces are only in the instruction to help make it easier for human readers to understand.

 - What **format** is this instruction? (Section 2.4)
 - What is the **mode** of this instruction? (Section 2.4)
 - What is the **opcode** of this instruction? (Section 2.4)
 - What is the **opspec** of this instruction? (Section 2.4)
 - What is the **operand** (OP) for this instruction? (Sections 2.4, 2.5)
3. What will be the effect of this instruction, when executed?
4. Write an appropriate comment for this instruction (similar to the comments given in the programs presented in the lectures).
5. Repeat this process (steps 2, 3 and 4) for the data that will be used to initialise a11 and a12.
6. What is the effect of executing the three instructions in this machine language initialisation file?

3. Writing a simple PeANUt machine language program

Write - on paper - a PeANUt machine language program that does the following:

- The memory cell a0 should be initialised to contain the decimal value 42.
- The memory cell a1 should be initialised to contain the decimal value 3.
- Your program should start at address a20.
- Your program should multiply the number stored in memory cell a0 with the number stored in memory cell a1, and then store the result into memory cell a3.
- Your program should be correctly terminated using the appropriate trap instruction.
- Write appropriate comments for each instruction of your program.

The program [addition.mli](#) from lecture P2 may be useful as a guide.

4. Referring to [Lecture P1](#) (np. 9) and [Lecture P2](#) (e.g. p 9), write down the sequence of internal steps in the PeANUt instruction cycle for the multiply instruction in the above program. Assume the address of the first instruction is at a10. Make sure you include the instruction fetch stage. Give the contents of the **CI** (in binary), and the **MAR** and **MDR** each time these change.
5. Suppose it is desired to increase the PeANUt memory size to 4096 cells. What problems would this create in the instruction set? Discuss ways in which this problem could be mitigated.

Part 2 - Laboratory Exercises (using the PeANUt simulator)

The PeANUt simulator is a tool that allows detailed viewing of the operation of the PeANUt computer. The objective of this laboratory is to introduce you to the PeANUt simulator and the basic operation of the PeANUt computer. Try to answer all questions requiring explanation yourself first, but don't spend long on any one - rather ask your tutor for help.

Twenty minutes or so before the end of your session, if you have not done so already, get to the [Your multiplication program](#) part and complete that before your session finishes. Any remaining parts can be completed in your own time.

Note that you may take this opportunity to ask any (last!) questions on Assingment 1.

1. Preliminaries