Lab 5
Microprocessor Architectures [ELEC-H-473]
Tessellated Intelligence System

v1.2.1

Introduction

Using an exclusive architecture, we want to show you:

- how to train to quickly digest a new architecture documentation and get productive quickly;
- how to optimize your code under heavy hardware constraints;
- that it is possible to learn and play at the same time.

What you will need:

- A copy of TIS-100 (DRM-free from the lab network drive (Z:/))
- The TIS-100 reference manual
- RTFM

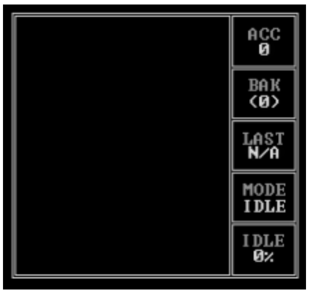
If you like this lab and want to continue playing, consider supporting the developers by buying the game on Steam (DRM-free key available at Humble Bundle)

1 Tessellated Intelligence System: TIS-100

You are a system engineer charged with completing a set of tasks on an old computer: the TIS-100. Its architecture is composed of twelve interconnected nodes that can exchange data with their direct neighbours. Each node can be one out of the three possible types:

- Regular computing node
- Memory stack node
- Corrupted and unusable node

1.1 Computing node



ACC
0

BAK
<0>

LAST
N/A

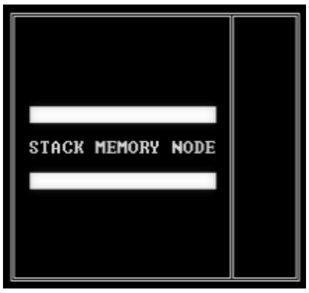
MODE
IDLE

IDLE
0%

The node is composed of a 15-lines programmable area surrounded by its register and various information. It has two registers: `ACC` and `BAK`, the former being directly addressable whereas the later and can only be modified using the `SAV` and `SWP` commands. The `LAST` register tells you which port has been used last, `MODE` what the node is doing and `IDLE` the percentage of time the node has spent doing nothing.

Using the `MOV` instruction, it can communicate with the surrounding computing or memory stack nodes.

1.2 Memory stack



The memory stack node, as its name indicates is a stack in which the first element added is the last that can be accessed. Such a structure is also called “last in, first out”, or LIFO.

The content of the node can be accessed using the `MOV` instruction.

1.3 Instruction set

The instruction set only contains a handful instructions such as addition, substraction and a bunch of comparison against zero. The full set is detailed in the reference manual, but the general idea is

```
<instruction> <source>, <destination>
```

Please note that although the reference manual says so, there is no “model-specific manual” nor “Tessellated Intelligence System Best Practices *” extra documents.

1.4 Use custom specifications

The game comes with a bunch of levels you need to solve in order to complete the game. However, you can also create your own levels using the specification editor.

You can import custom levels by copying the corresponding `.lua` files into the right folder depending on your operating system:

⊞ `My Documents\My Games\TIS-100\<UID>\custom`
🐧 `$HOME/.local/share/TIS-100/<UID>/custom`
 `$HOME/Library/Application Support/TIS-100/<UID>/custom`

where `<UID>` is either `0` if you are using the DRM-free version of the game or your Steam ID if you are playing on Steam.

For this lab, you will need to import three levels:

- “Decode the image” (`14173819.lua`)
- “Coprime detector” (`69396829.lua`)
- “Multiplication” (`73121466.lua`)

2 Your mission

2.1 Basics

We strongly advise you to first train on the pre-existing levels of the game before tackling the custom specifications (see Assignment). You should get the hang of it by the end of the fourth level: “Signal comparator”.

2.2 Assignment

We ask you to solve two custom levels: “Decode the image” and “Coprime detector”, **not** “Multiplication” which is only there as an optional non-graded bonus. The grade will be based on the performance of benchmarks and weighted across the three measures: node, instruction and cycle count. If you performed better than the benchmark for at least one of the three performance measures, you will get a perfect grade for the exercise, that is five out of five. Otherwise, you will be awarded two points if the problem is solved and each of the remaining three will be linearly weighted against the benchmark bounds for each metrics (e.g. if you performed the worst in all three measures, you won't get any more point, if you are right at the average, you will get half a point each, i.e. 3.5/5). Table 1 gives you the benchmark bounds.

Note: (1) For "Decode the image", you **have to** work with the inputs. (2) Also, only *Computing nodes* count toward the *node count* metric, meaning you can use memory stacks without impacting this part of your score.

	Cycles		Nodes		Instructions	
	min	max	min	max	min	max
Image	850	1500	4	10	40	60
Coprime	7000	25000	4	10	25	55

Table 1: Score bounds for evaluation. If you submit both assignments working with just 4 nodes each, you would have scored 10/10 regardless of you other metrics, since one of them (node count) is at least as good as the benchmark bound for each exercise.

You will need to send us two .txt files, one for each solution.

Those save files can be found in:

⊞ `My Documents\My Games\TIS-100\<UID>\save\<solution_file>`

🐧 `$HOME/.local/share/TIS-100/<UID>/save/<solution_file>`

🍎 `$HOME/Library/Application Support/TIS-100/<UID>/save/<solution_file>`

where `<solution_file>` follows this syntax: `SPEC<specification ID>.<solution number>.txt`.

For example, a solution to the "Decode the image" problem can be found under the name `SPEC14173819.0.txt`.

Do *not* change the file name when sending it to us. If you are not sure you are selecting the right file, just open it and check its content.