Swansea University
Prifysgol Abertawe

Concepts of CS I

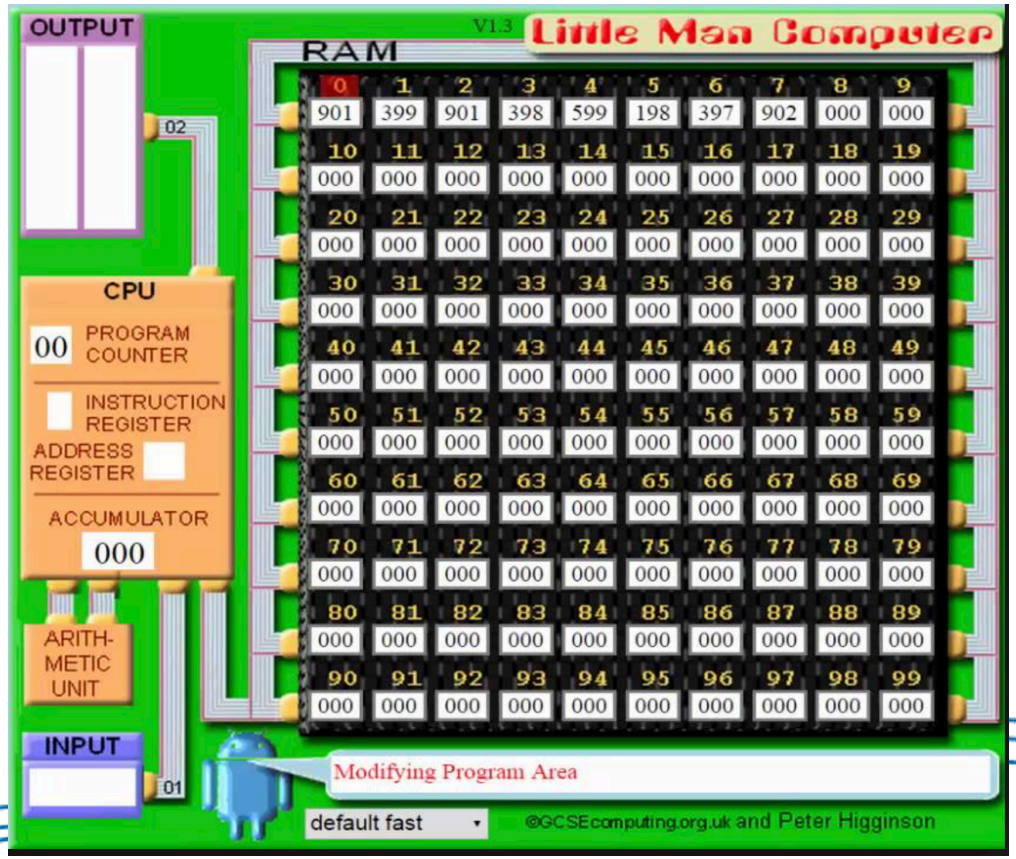
Little Man Computer

Mr. Stewart Powell s.w.powell@swansea.ac.uk Thursday 2nd November

Assembly Language

Swansea University
Prifysgol Abertawe

Little Man Computer is an assembly language simulator which gives an environment for users to program in an assembly language.



The Environment

Swansea University
Prifysgol Abertawe

- **Accumulator** - This is the like the active memory of the simulator. The majority of our instructions will modify the contents of the accumulator.
- **Program Counter** - This shows the current memory location that the processor is running.
- **MEM Address** - The current instruction type.
- **MEM Data** - The data being used for the current instruction.
- **In-Box** - The input box.
- **Out-Box** - The output box.

Taking Input

Swansea University
Prifysgol Abertawe

Name: Input
Mnemonic: INP
Code: *901*

Description:
The input instruction takes the value in the **In-Box** and puts the value into the **Accumulator**.

Next Action:
After the value has been copied the **Program Counter** will move onto the next (sequential) memory location.

Providing Output

Swansea University
Prifysgol Abertawe

Name: Output
Mnemonic: OUT
Code: *902*

Description:
The output instruction takes the value in the **Accumulator** and puts the value into the **Out-Box**.

Next Action:
After the value has been copied the **Program Counter** will move onto the next (sequential) memory location.

Swansea University
Prifysgol Abertawe

Stopping the Programming

Name: Halt
Mnemonic: HLT
Code: *000*

Description:
The halt instruction does not effect any of the memory locations and stop the program.

Next Action:
The execution of the program will stop.

Swansea University
Prifysgol Abertawe

Exercise: Input and Output

- Create a program which takes in an input from the user and outputs it.
- Analysis the memory locations and write down the instruction codes your program generated.

LMC Code Summary

Swansea University
Prifysgol Abertawe

___	**INP**	___	- Input
___	**OUT**	___	- Output
___	**HLT**	___	- Halt

Storing Data

Swansea University
Prifysgol Abertawe

Name: Store
Mnemonic: STA *variable*
Code: *3* _ _

Description:
The store instruction will take the data from the **Accumulator** and store it into an allocated memory location which will be referred to by the variable name given.

Next Action:
After the value has been copied the **Program Counter** will move onto the next (sequential) memory location.

Retrieving Data

Swansea University
Prifysgol Abertawe

Name: Load
Mnemonic: LDA variable
Code: *5 _ _*

Description:
The load instruction will put the value stored at the variable location into the **Accumulator**.

Next Action:
After the value has been loaded into the **Accumulator** ,the **Program Counter** will move onto the next (sequential) memory location.

Swansea University
Prifysgol Abertawe

Data Memory Locations

Name: Data
Mnemonic: *variable* DAT
Code: *(the data)*

Description:
The data instruction will reserve a memory location to store data. This location can then be referred to by the given name.

Next Action:
After the memory location has been reserved, the **Program Counter** will move onto the next (sequential) memory location.

Exercise: Storing and Loading

- Create a program which takes and stores in 2 inputs from the user and outputs the first input followed by the second input.

- Create a program which takes and stores 4 inputs from the user and always outputs the third input to the user.

LMC Code Summary

Swansea University
Prifysgol Abertawe

___	**INP**	___	- Input
___	**OUT**	___	- Output
___	**HLT**	___	- Halt
___	**STA**	***var***	- *Store*
___	**LDA**	***var***	- Load
var	**DAT**	___	- Data

Addition

Name: Addition
Mnemonic: *ADD variable*
Code: 1 _ _

Description:

The add instruction adds the value stored in the given memory location to the accumulator.

Next Action:

After the value has been loaded into the **Accumulator** ,the **Program Counter** will move onto the next (sequential) memory location.

Subtraction

Name: Subtraction
Mnemonic: *SUB variable*
Code: *2 _ _*

Description:

The subtraction instruction subtracts the value stored in the given memory location away from the accumulator.

Next Action:

After the value has been loaded into the **Accumulator** ,the **Program Counter** will move onto the next (sequential) memory location.

Exercise: Addition Subtraction

- Create a program which takes and stores in 2 inputs from the user and outputs the sum of them.

- Create a program which take in three numbers and stores them and then outputs the sum of the first 2 numbers with the third subtracted.

LMC Code Summary

Swansea University
Prifysgol Abertawe

___	**INP**	___	- Input
___	**OUT**	___	- Output
___	**HLT**	___	- Halt
___	**STA**	***var***	- *Store*
___	**LDA**	***var***	- Load
var	**DAT**	___	- Data
___	**ADD**	***var***	- Addition
___	**SUB**	***var***	- *Subtraction*

Go to (Branch Always)

Swansea University
Prifysgol Abertawe

Name: Branch Always
Mnemonic: *BRA variable*
Code: *6 _ _*

Description:

Moves the **Program Counter** to the memory location stored within the variable memory location.

Next Action:

After the memory location has been loaded that memory location will be executed.

Exercise: Looping

- Create a program which allows the user to input numbers indefinitely and outputs the number which was entered last.
- Create a program which allows the user to input numbers indefinitely and outputs the running total after each entry.
- (Extension) Create a program which allows the user to input numbers indefinitely and outputs the sum at each entry.

Swansea University
Prifysgol Abertawe

LMC Code Summary

___	**INP**	___	- Input
___	**OUT**	___	- Output
___	**HLT**	___	- Halt
___	**STA**	***var***	- *Store*
___	**LDA**	***var***	- Load
var	**DAT**	___	- Data
___	**ADD**	***var***	- Addition
___	**SUB**	***var***	- *Subtraction*
___	***BRA***	***var***	- Branch Always

Go to (Branch If Zero)

Swansea University
Prifysgol Abertawe

Name: Branch If Zero
Mnemonic: *BRZ variable*
Code: *7 _ _*

Description:

Moves the **Program Counter** to the memory location stored within the variable memory location if the accumulator is equal to zero.

Next Action:

After the memory location has been loaded that memory location will be executed.

Go to (Branch If Zero or Positive)

Name: Branch If Zero or Positive
Mnemonic: *BRP variable*
Code: *8 _ _*

Description:

Moves the **Program Counter** to the memory location stored within the variable memory location if the accumulator is zero or positive.

Next Action:

After the memory location has been loaded that memory location will be executed.

LMC Code Summary

Swansea University
Prifysgol Abertawe

___	**INP**	___	- Input
___	**OUT**	___	- Output
___	**HLT**	___	- Halt
___	**STA**	***var***	- *Store*
___	**LDA**	***var***	- Load
var	**DAT**	___	- Data
___	**ADD**	***var***	- Addition
___	**SUB**	***var***	- *Subtraction*
___	***BRA***	***var***	- Branch Always
___	***BRZ***	***var***	- Branch If Zero
___	***BRP***	***var***	- Branch If Positive

Exercise: Conditional Branching

- Create a program which allows the user to input two numbers and outputs the multiplication of the two numbers.

- Create a program which allows the user to input two numbers and outputs the smallest number.

  Hint: if you do a - b and the number is positive then a is bigger than b.