

## The LGP-30

The LGP-30 was a general-purpose binary computer using a drum for its memory

The word length is 32 bits although a memory word contains only 31 bits

The bits are numbered from 0 (the sign bit) to bit 31 for registers or bit 0 through bit 30 for memory words

There are 4096 words of memory - each word containing 31 bits

There are actually 32 bits in a word but the last bit in memory, called the "spacer" bit, is always zero

In addition to the memory there were 3 registers

C - Counter register

Keeps track of the address of the next instruction to be executed

I - Instruction register

Holds the instruction currently being executed

A - The accumulator register

Holds data being manipulated by the program

Each of the three registers are implemented as recirculating registers using two heads on the same track of the drum spaced apart by one word time

Actually the accumulator has an additional head to allow holding a 64-bit number during multiplication and division

The instruction register would remember the data from memory so it could be repeatedly added or subtracted from the (extended) accumulator

If bit 31 is set in the A register (by an input operation) it must be shifted out by a "n" or "d" instruction or it will be lost as memory and many operations reset this bit

## Normal Instruction Execution Cycle

The C register is continually compared with the current sector until there is a match

The I register is loaded from the correct track from the word in the next sector

The C register is incremented by 1

Depending on the instruction the address in the I register is compared with the current sector until there is a match

The next sector on the appropriate track is read/written from/to the A register or combined with the A register

The cycle then repeats (fetching and executing the next instruction)

## The Instruction Set

**zxx00**

stop unless track bit in binary matches depressed breakpoint switch

**bxxxx**

load the contents of memory location xxxx into A

**yxxxx**

store the address portion of A into location xxxx without affecting any other bits of xxxx

**rxxxx**

store the address of this instruction + 2 in the address portion of location xxxx without affecting any other bits of xxxx

**iaaxx**

read data from the selected input device 4 bits at a time unless the six-bit switch is depressed until a stopcode is read

The track address aa is used as the first character input (generally set to zero)

**dxxxx**

divide A by the contents of location xxxx and place the result in A

Halt if the division overflows

**nxxxx**

multiply A by the contents of location xxxx and place the right half of the product in A

multiplying by 2 will shift the contents of A left 1 bit

**mxxxx**

multiply A by the contents of location xxxx and place the left half of the product in A

multiplying by 4000000 will shift the contents of A right 1 bit

**paaxx**

print the character aa on the selected output device and send a start signal when done - no memory reference is done

**exxxx**

"and" A by the contents of location xxxx and place the result in A - "e" stands for extract

**uxxxx**

load C with XXXX causing the instruction in xxx to be the next instruction executed

unconditionally transfer to location xxxx

**txxxx**

transfer to location xxxx if the A register is negative

**800txxxx**

transfer to location xxxx if the A register is negative or the transfer control switch is depressed

**hxxxx**

store the contents of A in location xxxx and hold the contents of A unchanged

**cxxxx**

store the contents of A in location xxxx and clear the contents of A to zero

**axxxx**

add the contents of location xxxx to A and place the sum in A

halt if the addition overflows

**sxxxx**

subtract the contents of location xxxx from A and place the difference in A

halt if the subtraction overflows

## Modes of Operation

There are three modes that the LGP-30 can be in

### Manual Input

All keyboard input goes to the A register

If the "6 Bit Input" switch is not depressed then the A register is shifted left 4 bits and the top 4 bits of the character are inserted in the right 4 bits of the A register

If the "6 Bit Input" switch is depressed then the A register is shifted left 6 bits and the character is inserted in the right 6 bits of the A register

The start button copies the memory location specified by the C register to the I register and then increments the C register by one

### One Operation

pressing the start button executes the single instruction in the memory location specified by the C register and advances the C register

This is useful for stepping through a program to see it operate

### Normal

pressing start will start executing instructions starting with the instruction specified by the C register

## LGP-30 Hexadecimal

The LGP-30 uses a different hexadecimal code than we use today

Input from the keyboard (when not in 6 bit mode) will use this code

The digits are

0123456789fgjklqwvx

The characters for the opcodes of the LGP-30 have the proper character codes and form another 16 characters

zbyridnmp euthcas

## Track and Sector Notation

Coding everything in hexadecimal would require converting track and sector parts of the address to different hexadecimal values

The loader will convert instructions in the form  
zttss

where tt and ss are two-digit decimal numbers to the correct binary representation (z is the opcode)

The opcode is unchanged because (fortunately) the character values of the opcodes is the correct binary value

The sector part of the address becomes important when optimally arranging the data on the drum to minimize access time

## The boat loader 10.4

The standard load program for the LGP-30 is called 10.4 and reads a paper tape (or file in our simulator) and loads memory according to the following rules

In the 10.4 loader all addresses are track and sector in decimal

:000aaaa' - start fil

start loading instructions at location aaaa

/000aaaa' - modifier

add aaaa (track and sector) to all instructions not preceded by an "x"

.000aaaa' - start location

start execution at location aaaa

halt first unless breakpoint 32 is depressed

,0000nn' - load hex

the next nn words on the tape are hexadecimal constants to be loaded into the next memory locations

### Instructions

Instructions have one of the following forms

z1234'

xz1234'

800z1234'

80xz1234'

The "x" means don't add the modifier

The initial 8 means set the sign bit on the instruction useful for the t instruction that senses the transfer control switch

If you want to just specify absolute addresses you can set the modifier to zero

/0000000'

## Example program

The following program, when loaded with the 10.4 loader, will print "hi" on the output

```
/0000000' ; 0000300'  
p4900' z0000' p1700' z0000' z0000' u0300'  
.0000300'
```

The stop instructions after the print instructions stop the computer until the character is printed

## The Simulator

The simulator is a java program located in the course account

[www.cs.rit.edu/~swm/history/LGP30](http://www.cs.rit.edu/~swm/history/LGP30)

All of the source code is contained in a single source file LGP30.java

Many, many class files result from compiling this single source file

For your convenience all of the files you need are packed in a LGP30.jar file

all source code and LGP-30 files are included too

To run the simulator either compile the source file and type

```
java LGP30
```

or download the jar file and type

```
java -jar LGP30.jar
```

The buttons that latch are colored yellow when depressed

The other buttons are momentary

The File menu

Clear/new clears drum memory

load memory initializes memory from an image file

save memory saves memory to a memory image file

The LGP 30 State menu is currently not functional

the Reader/Punch menu

Close Input File closes the input file so input will come from the keyboard

Close output file closes the output file - currently not functional

Load Input Tape opens up a file and directs all "i" commands to read from this file

Punch tape currently not functional

## The Simulator Display

The top bar is the pulldown menu bar with useful tools

Below this are two "lights" "STOP" and "COMPUTE"

The stop light lights when the computer halts or waits for input

The compute light lights when executing instructions

Below this are the 18 buttons and switches

The first three on the top row latch and only one can be pressed at a time

Normal sets the machine to operate continuously

One Operation sets the machine to execute one instruction at a time

Manual Input connects the keyboard to the A register so any character typed goes into the right end of A after shifting A left 4 or 6 bits

The rest of the buttons on the top row are not functional

The Start button executes the next instruction located at C and continues execution in normal mode

In Manual Input mode fetches the instruction at C into I but does not execute it so you can examine memory this way

Clear Counter (unless in normal mode) clears the C register

Fill Instruction (unless in normal mode) loads the I register from A

Execute Instruction in One Operation mode executes the instruction in the I register

Power On and Power Off are not simulated

The lower row of buttons are all latching buttons and effect the operation of the machine as described by the instruction descriptions

There is a typing window that displays all output and allows keyboard input

The bottom line is the feedback line where messages are displayed

When the machine is running it displays an estimated running time in seconds

## Loading the 10.4 loader into memory

To load the 10.4 loader into memory do the following:

run the simulator

make sure that "6 Bit Input" is off

press "Manual Input"

select the typing area and type

c0000

press "Fill Instruction"

select the typing area and type

i0000

press "One Operation", Execute Instruction", "Manual Input"

select the typing area and type

c0004

press "Fill Instruction"

select the typing area and type

c000j

press "One Operation", Execute Instruction", "Manual Input"

c0008

press "Fill Instruction"

select the typing area and type

i0000

press "One Operation", Execute Instruction"

On the Reader/Punch menu select "Load Input Tape ..." and select the 104.boot file

press "One Operation", Clear Counter", "Normal", "Start"

The simulator should then proceed to load the loader into memory

When it finishes select "Close Input File" in the Reader/Punch menu

## Loading the example program

(You can also load the 10.4 loader into memory from the file 104.lgp30)

Now that the loader is in memory you can run it by branching to location zero so -

On the Reader/Punch menu select "Load Input Tape ..." and select the example.104 file

press "One Operation", Clear Counter", "Normal", "Start"

The program should now be loaded into memory and the machine will halt

Pressing "Start" will transfer to the example program and run it

The example program prints "hi" and halts

Pressing "Start" again will run the example program again

## Limitations of the Simulator

possible bugs still exist

all lower case on output

only lower case on input

letter "L" doesn't work

p0000 command ignored but i command will read properly (just like the photoreader)

display in normal hexadecimal 0123456789abcdef

selecting an output file doesn't work yet

printing too fast works instead of garbaging the output

power off, power on, standby etc. are not implemented

Not all of the menus are implemented yet